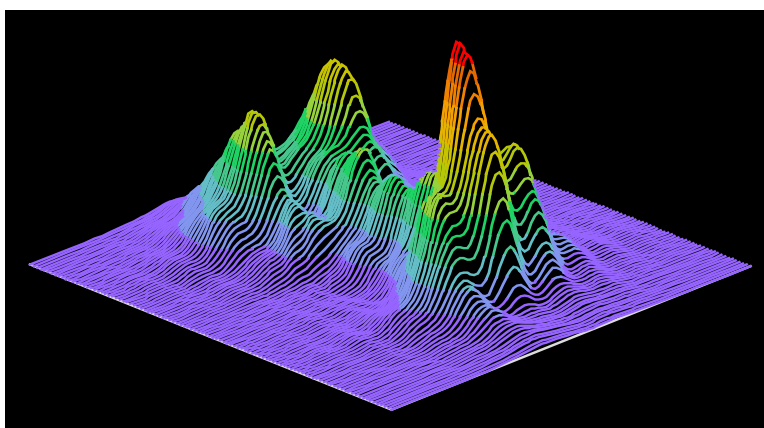


GLAD Examples Manual

Ver. 6.1



Applied Optics Research

Applied Optics Research
1087 Lewis River Rd. #217, Woodland, WA 98674
Tel: (360) 225 9718, Fax: (360) 225 0347, email: glad@aor.com
www.aor.com


Copyright 1986-2018 Applied Optics Research. All Rights Reserved. 
Reproduction of this manual is prohibited.

Table of Contents

Introduction

Ex1: INPUT Conventions and RTF command files

Ex1a: Input conventions

Ex1b: Rich text format (RTF) command files

Ex2: Beam initialization and automatic units control

Ex2a: Gaussian and supergaussian beam generation, automatic units

Ex2b: Calculation of beam and array size through the waist

Ex2c: Calculation of beam and array size with grating

Ex2d: Calculation of beam and array size with shallow focus

Ex3: Unit selection

Ex4: Variables, expressions, and mathematical surfaces

Ex5: Simple lenses and mirrors

Ex6: Conic mirrors and three-dimensional orientation

Ex7: Command: `mirror/global`

Ex8: Conic mirrors

Ex8a: Confocal parabolas, alternate surface

Ex8b: Single off-axis parabola

Ex8c: Elliptical mirror

Ex8d: Off-axis parabola with high NA effects

Ex8e: Eigen mode treatment for an array of elliptical mirrors

Ex9: Beam steering with flat mirrors in three-dimensions

Ex10: Macros, variables and the `udata` command

Ex11: Confocal unstable resonator

Ex11a: Unstable, bare cavity resonator

Ex11b: Unstable, bare cavity resonator with apodization

Ex11c: Unstable resonator with diverging output

Ex11d: Bare Cavity Resonator, seeded with reverse mode

Ex11e: Finding first six modes of an unstable resonator.

Ex11f: Gain sheets in an unstable resonator

Ex11g: 1-D Unstable Resonator

Ex12: Misaligned confocal unstable resonator

Ex13: Phase aberrations

Ex13a: Display of various types of aberration

Ex13b: Bitmap displays of Zernike aberration

Ex14: Beam fitting

Ex15: General obscuration

Ex16: General apertures and obscurations

Ex17: Raman amplifier

Ex18: Raman amplification with multiple Stokes beams

Ex19: Raman process with converging beams: simple kinetics step

- Ex20: Raman amplification with wave4: collimated beams
- Ex21: Four-wave mixing with wave4: collimated, geometric propagation
- Ex22: Raman gain and four-wave mixing, collimated
- Ex23: Four-wave mixing wave4, converging beam
- Ex24: Atmospheric aberration and adaptive optics
 - Ex24a: Atmospheric turbulence
 - Ex24b: Atmospheric aberration in a collimated path
 - Ex24c: Atmospheric aberration in a converging path
- Ex25: Ground-to-space laser communication system
- Ex26: Ground-to-space laser propagation system with atmospheric aberration
- Ex27: Ground-to-space laser with atmospheric aberration and jitter
 - Ex27a: Effects of atmosphere and jitter just before relay mirror
 - Ex27b: Effects of atmospheric aberration and jitter observed just after the relay mirror
 - Ex27c: Effects of atmospheric aberration, jitter, and adaptive optics after the relay mirror
- Ex28: Phased array
 - Ex28a: Phased Array
 - Ex28b: c## ex28a
- Ex29: Atmospheric aberration with wind shear
- Ex30: Specklon, near- and far-field
- Ex31: Thermal blooming
 - Ex31a: Propagation with no thermal blooming
 - Ex31b: Thermal blooming, no kinetic cooling
 - Ex31c: Thermal blooming with kinetic cooling and aberration
- Ex32: Phase conjugate mirror or correcting aberration plate
 - Ex32a: Conjugate mirror in near-field
 - Ex32b: Conjugating mirror or aberration plate in either near- or far-field
- Ex33: Stable resonator
 - Ex33a: Half-symmetric stable cavity
 - Ex33b: Half-symmetric stable cavity with 1:1 intercavity telescope, ideal lenses
 - Ex33c: c## ex33b
 - Ex33d: c## ex33c
 - Ex33e1: Coherent injection, decentered input, part 1
 - Ex33e2: Coherent injection, decentered input, part 2
 - Ex33f: Global definition of half-symmetric resonator
 - Ex33g: Example of a wire obscuration to induce TEM₁₀
 - Ex33h: Example of half-symmetric resonator with rotating end mirror
 - Ex33i: Two-wavelength flat-flat resonator
 - Ex33j: Multiple beams through the same resonator
 - Ex33k: Extended cavity and spurious reflections
 - Ex33l: Coupled resonator
 - Ex33m: Finding multiple modes by orthogonality
 - Ex33n: Bowtie resonator
- Ex34: Stable-unstable resonator

- Ex35: Distributed propagation through a refractive surface
 - Ex35a: Distributed propagation, method of aperture division
 - Ex35b: Distributed propagation, phase grating in incident beam
 - Ex35c: Distributed propagation, phase grating on refracting surface
 - Ex35d: Propagation to tilted surface for a phase grating.
 - Ex35e: Propagation to tilted surface for a circular aperture.
- Ex36: Finite-difference propagator
 - Ex36a: Finite-Difference Propagator (FDP), Soft Aperture
 - Ex36b: Finite-Difference Propagator, Hard Aperture with FFT
- Ex37: Polarization, Jones matrices
 - Ex37a: Polarization, Jones matrices
 - Ex37b: Polarization, surface polarization effects
 - Ex37c: Transmission and reflection coefficients for light incident at Brewster's angle
 - Ex37d: Polarization, Goos-Hanchen, Part 1
 - Ex37e: Polarization, Goos-Hanchen with `jsurf/goos`, Part 2.
 - Ex37f: Birefringent wedge using 3d addressing of polarization sheets
 - Ex37g: Calculate polarization properties through a Dove prism
- Ex38: Shearing Interferometer
- Ex39: Gaussian phase factor in propagation, Gouy shift
- Ex40: Phase conjugation, limited interaction length
- Ex41: Effect of spatial filter on polarization
- Ex42: Waveguide grating coupler, mode matched input
- Ex43: Waveguide grating coupler, reversed mode input
- Ex44: Waveguide grating coupler, reversed mode input with aberration
- Ex45: Unstable ring resonator with 90 degree rotation and lensing medium
- Ex46: Beam shaping filter
- Ex47: Gain sheet modeling
 - Ex47a: Unstable, Loaded Cavity Resonator with Beer's Law Gain
 - Ex47b: Unstable, loaded cavity resonator with gain sheet
 - Ex47c: Unstable, loaded cavity resonator with gain sheet, single step
 - Ex47d: Point-by-point control of gain and saturation
 - Ex47e: Point-by-point control of gain and saturation, multiple beam saturation
- Ex48: Frequency doubling
 - Ex48a: Frequency Doubling: Tuned and Detuned
- Ex49: Frequency doubling, closed form model
- Ex50: Out-coupled polarization for TE and TM waveguide modes
- Ex51: Calculation of TE and TM outcoupling with the Induced Dipole Model
 - Ex51a: Waveguide grating incoupling, TE
 - Ex51b: Waveguide grating incoupling, TM
- Ex52: Cone (axicon) aberration
- Ex53: Hermite-gaussian functions
 - Ex53a: Some Hermite-Gaussian polynomials
 - Ex53b: Construction of donut mode from HG(1,0) and HG(0,1), orthogonal polarizations.

- Ex54: Laguerre functions
- Ex55: Speckle pattern in the far-field
- Ex56: Fabry-Perot cavity with coherent injection
 - Ex56a: Find Gouy phase of ideal gaussian mode
 - Ex56b: Scan coherent injected signal around Gouy phase, peak at 140 deg.
 - Ex56c: Orthogonalization to show build-up and Gouy phase of second lowest loss mode
 - Ex56d: Coherent laser injection tuned for gaussian mode (real apertures)
 - Ex56e: Coherent laser injection tuned for gaussian mode (real apertures)
 - Ex56f: Scan injected signal over longitudinal mode spacing
- Ex57: Stable resonator with obscuration to generate higher order modes
- Ex58: Effect of absorption and self-focusing in a gaussian beam
 - Ex58a: Penetration depth in a Beer's Law absorber, no absorption
 - Ex58b: Penetration depth in a Beer's Law absorber, absorption included
 - Ex58c: Penetration depth in a Beer's Law absorber, Beer's Law and self-focusing
 - Ex58d: Penetration depth in a Beer's Law absorber, absorption, self-focusing, aberration
- Ex59: Through-focus image of spherical aberration with central obscuration
 - Ex59a: Through-focus spherical aberration, with obscuration
 - Ex59b: Through-focus spherical aberration, no aberration
 - Ex59c: Two-focal-length lens, through-focus scan
- Ex60: Optimization of size and location of an elliptical pinhole
 - Ex60a: Simple optimization of astigmatic focus
 - Ex60b: Numerical check of optimization, scalar targets
 - Ex60c: Numerical check of optimization, array targets
 - Ex60d: Reverse optimization of aperture shape, array targets, numerical check
 - Ex60e: Reverse optimization of aperture shape, array targets, built-in functions
- Ex61: Optimization for accelerated mode estimation
- Ex62: Linear grating with a small defect
 - Ex62a: Square wave gratings, effect of small obscuration
 - Ex62b: Square wave gratings, effect of second grating
- Ex63: Comparison of Beer's Law and CO2 gain
- Ex64: Lens array using single apertures
- Ex65: Incoherent imaging using OTF
- Ex66: Roof mirrors and corner cubes
- Ex67: Lens and laser diode arrays
 - Ex67a: Hexagonal lens array
 - Ex67b: Rectangular lens array
 - Ex67c: Lensarray used as optical integrator
 - Ex67d: Four cylindrical lenses
 - Ex67e: Array of lenslets with 25 cm focal length
 - Ex67f: Two lensarrays creating afocal 1:1 imager
 - Ex67g: Array of optical fibers collimated by lensgroup
 - Ex67h: Array of N x N laser diodes, gaussian overall envelope
- Ex68: Resonator with Brewster windows

- Ex68a: Resonator with polarization by JSURF, 1 micron wavelength
- Ex68b: Resonator with polarization by JSURF, 100 micron wavelength
- Ex69: Rate equations, transient response
 - Ex69a: Rate equation gain and mode competition
 - Ex69b: Rate equation gain for ruby laser
 - Ex69c: Rate equations, single step
 - Ex69d: Semiconductor gain
 - Ex69e: Three-level gain, single upper state
 - Ex69f: Numerical example of rate equation gain
 - Ex69g: Numerical example of single level, three level gain
 - Ex69h: Rate equations for ruby
 - Ex69i: Rate equations for general, three level laser
 - Ex69j: Steady-state rate equation solution
 - Ex69k: Rate equations for single level, three-level laser, multiple steps
- Ex70: Udata display
- Ex71: Schlieren system
- Ex72: Test ABCD equivalent systems
- Ex73: Tests of dynamic memory
- Ex74: More checks of dynamic memory allocation
- Ex75: Axicons
 - Ex75a: Afocal axicon left out and left back
 - Ex75b: Afocal axicon with shift on return, both left
 - Ex75c: Afocal axicon left and right with shifts
 - Ex75d: Afocal axicon with large positive shift
 - Ex75e: Stable resonator with focal axicons
 - Ex75f: Unstable resonator with focal axicons
 - Ex75g: Form annular beam from radial beam, reflaxicon
 - Ex75h: Form annular beam from radial beam, waxicon
 - Ex75i: Reflaxicon-waxicon pair with intermediate ring focus
 - Ex75j: Effect of decenter and tilt on beam in radial mode
 - Ex75k: Waxicon resonator with inverting optics (simple treatment)
 - Ex75l: Waxicon resonator with inverting optics (exact mirror treatment)
- Ex76: Stable resonator with coherent injection
 - Ex76a: Example of coherent injection, bare cavity analysis
 - Ex76b: Example of coherent injection, automatic frequency control
- Ex77: Hollow waveguide with reflecting walls
 - Ex77a: Hollow waveguide with reflecting walls
 - Ex77b: Tapered waveguide with a converging beam centered at the perspective point
 - Ex77c: Inject collimated beam into tapered waveguide
 - Ex77d: Inject collimated beam into curved waveguide
 - Ex77e: Waveguide optical integrator
 - Ex77f: Waveguide in a stable resonator
 - Ex77g: Half waveguide in unstable resonator

- Ex77h: Resonator with waveguide in place
- Ex77i: Incoherent treatment of reflecting-wall waveguide, converging beam
- Ex78: Automatic optimization of resonator design
- Ex79: Transient Raman
 - Ex79a: Effect of wide angle noise, 64 x 64 array
 - Ex79b: Effect of wide angle noise, 256x 256 array
 - Ex79c: Transient behavior of the Raman process for a gaussian temporal waveform
 - Ex79d: Transient behavior of the Raman process with doubled irradiance
 - Ex79e: Effect of weak hot spots in the laser to create strong Stokes spikes
- Ex80: Q-switch laser
 - Ex80a: Q-switch YAG laser
 - Ex80b: Q-switch YAG laser, full polarization
 - Ex80c: Q-switch, saturable absorber
 - Ex80d: Q-switch, time-limited Beer's Law gain
 - Ex80e: Q-switch, YAG laser using a relatively slow Q-switch
 - Ex80f: Q-switch, YAG laser pulsed laser diode pumping
 - Ex80g: An example of a gain/absorber, numerical check.
 - Ex80h: An example of a saturable absorber based on gain/absorber
- Ex81: Illustration of zone control of propagation
 - Ex81a: Illustration of zone command
 - Ex81b: Through-focus image of circular aperture
 - Ex81c: Lensarray used as optical integrator
- Ex82: Table building feature
- Ex83: Partial coherence
 - Ex83a: Partial coherent imaging of a three-bar pattern
 - Ex83b: Two sets of seven bars, above and below the coherent resolution limit
 - Ex83c: Thirteen-bar pattern to compare with theory
- Ex84: Effects of laser heating of a window, convolution method
- Ex85: Geometrical optics using lensgroup
 - Ex85a: Simple lens with a tilted reflector
 - Ex85b: Cooke triplet
 - Ex85c: Tilted Cook triplet
 - Ex85d: Cooke triplet with mirror, reversed lens
 - Ex85e: Cooke triplet with 180 rotation
 - Ex85f: Cooke triplet with 45 tilt mirror and 90 vertex rotation
 - Ex85g: Example to illustrate propagation through Brewster angle window
 - Ex85h: Propagation of a decentered beam through a microsphere
 - Ex85i: 60-60-60 prism used at minimum deviation
- Ex86: Fiber optics and waveguides
 - Ex86a: Straight waveguide
 - Ex86b: Sinusoidal waveguide
 - Ex86c: Waveguide with two close cores
 - Ex86d: Multimode fiber

- Ex86e: Array of 11 x 11 fiber cores, detailed analysis
- Ex86f: Response function for 11 x 11 array of fibers by overlap integral
- Ex86g: Focused beam into straight fiber
- Ex86h: Direct observation of the propagation constant
- Ex86i: Slab waveguide treating both guided and free-space direction
- Ex86j: Analytical calculation of slab waveguide eigenmode
- Ex86k: Slab waveguide treating both guided and free-space direction
- Ex86l: Comparison of BPM mode and gaussian approximation
- Ex86m: Comparison of BPM and gaussian at critical frequency
- Ex86n: Fiber with long radius bend
- Ex86o: A pencil of light injected into a dielectric waveguide at near-TIR angle.
- Ex86p: A pencil of light injected into a tapered waveguide.
- Ex87: Slab waveguides using extrude and slab/waveguide
 - Ex87a: Two straight waveguides, extruded
 - Ex87b: Curved waveguides forming a directional coupler, equal division
 - Ex87c: Curved waveguides forming a directional coupler, 100% conversion
 - Ex87d: Y-splitter
 - Ex87e: Y-combiner, input into a single leg
 - Ex87f: Y-combiner, input into both legs
 - Ex87g: Optical switch, ON
 - Ex87h: Optical switch, OFF
 - Ex87i: Waveguide lens
 - Ex87j: Double directional coupler
 - Ex87k: Three directional couplers producing a fan-out of five equal beams
- Ex88: Speckle smoothing with a lens array integrator
 - Ex88a: Perfect pupil with lensarray
 - Ex88b: Perfect pupil with lensarray, interference pattern
 - Ex88c: Independent random phase plates or random amplitude with lens array
 - Ex88d: Random phase grating, lens array, mirror dither.
 - Ex88e: Random phase grating, lens array, circular motion.
 - Ex88f: Calculation of speckle smoothing by linear motion
 - Ex88g: Smoothing of diffraction ringing by partial coherence
- Ex89: Binary optics
 - Ex89a: Binary grating surface calculation
 - Ex89b: Binary grating, direct phase calculation
 - Ex89c: Binary lens, positive element
 - Ex89d: Binary lens, negative element
 - Ex89e: Binary lens, positive-negative element
 - Ex89f: Binary lens, dispersion
 - Ex89g: Binary fractioning of an arbitrary surface
- Ex90: High numerical aperture lens with vector diffraction calculation
 - Ex90a: High numerical aperture objective lens
 - Ex90b: High numerical aperture spatial filter, includes recollimation step

Ex91: Measuring beam width and M-squared

- Ex91a: Measuring spot width by mode fitting
- Ex91b: Measuring spot width using `fitgeo` with noise
- Ex91c: Power-in-the-bucket
- Ex91d: Fitting Hermite-Gaussian
- Ex91e: Finding and displaying the region containing specified energy
- Ex91f: Fitting embedded gaussian to data set
- Ex91g: Fitting embedded gaussian, noise and aberration
- Ex91h: Fitting embedded gaussian, noise and aberration
- Ex91i: Calculation of best-fit gaussian as a function of lens spacing

Ex92: Thermal changes in refractive elements

- Ex92a: Two-dimensional heat flow, window, metal mount, air contact, internal heat source
- Ex92b: Two-dimensional heat flow, window, air contact, internal heat source
- Ex92c: Three-dimensional heat flow, point source of heat
- Ex92d: Thermally induced aberrations in a window
- Ex92e: Thermally induced change in optical power in a lens
- Ex92f: Three-dimensional heat flow, YAG material
- Ex92g: Thermally-induced stress birefringence
- Ex92h: Simple model of thermal array and optical aberrations
- Ex92i: Thermal distortion due to end pumping

Ex93: Design of far-field distribution by phase-retrieval methods

- Ex93a: Design of far-field distribution by phase-retrieval methods
- Ex93b: Recovery of pupil aberrations from spherically aberrated image

Ex94: Fiber optic laser

- Ex94a: Single centered core
- Ex94b: Single decentered core
- Ex94c: Four cores

Ex95: Optical parametric oscillator

- Ex95a: Interference between a straight and tilted beam
- Ex95b: Propagation in a uniaxial crystal
- Ex95c: Optical parametric amplifier: tuned and detuned beams
- Ex95d: Optical parametric amplifier: aligned and misaligned beams
- Ex95e: Optical parametric amplifier, misaligned, various crystal lengths
- Ex95f: Use of `mult/tensor` for three-wave interactions
- Ex95g: Use of `mult/tensor` for four-wave interactions
- Ex95h: Interference between a straight and tilted beam, in glass
- Ex95i: Resonator with OPA included

Ex96: Circular propagator

- Ex96a: Example of one-dimensional, circular array
- Ex96b: Compare diffraction propagation of square and circular beams
- Ex96c: General propagation of circular beams

Ex97: Volume holograms and GRIN lens arrays

- Ex97a: Volume hologram showing mode conversion versus propagation length

- Ex97b: Propagation through three-beam interference pattern
- Ex97c: Propagation through four-beam interference pattern
- Ex98: Design of far-field distribution by simulated annealing
 - Ex98a: Initialize array
 - Ex98b: Run this example to convergence, about 100,000 times
 - Ex98c: Final run to make phase plot
- Ex99: Michelson interferometer and point diffracting interferometer
 - Ex99a: Initialize array
 - Ex99b: Michelson interferometer, relative tilt
 - Ex99c: Michelson interferometer, finite spectral width
 - Ex99d: The point diffracting interferometer
- Ex100: Cavity mode and power spectrum for flat-flat resonator
- Ex101: Measurement of excimer laser with Moire fringes
 - Ex101a: Crossed Ronci rulings with slight rotation, coherent input
 - Ex101b: Crossed Ronci rulings with slight rotation, excimer with 200 speckle
- Ex102: Vector addition of beams to make lenslet array
- Ex103: Circular and pentagonal reflecting wall waveguide
 - Ex103a: Example of cylindrical rod analyzed by the method of images
 - Ex103b: Circular rod, two reflecting walls
 - Ex103c: Circular rod, small memory model
 - Ex103d: Circular rod, large memory model
 - Ex103e: Pentagonal rod
- Ex104: Phase gratings: resolved and nonresolved
 - Ex104a: Phase grating with `abr/lrip`, square wave and blazed, resolved model
 - Ex104b: Phase grating with `grating/*/phase`, square wave and blazed, resolved model
 - Ex104c: Cosine phase grating, comparison of resolved and nonresolved models.
 - Ex104d: Global grating with vertex tilt
 - Ex104e: `Global/grating` used with a global spherical mirror
 - Ex104f: Including aberrations of the grating rulings
 - Ex104g: Comparison of side lobe intensity for `global/grating` and resolved models
- Ex105: Three-dimensional arrays
 - Ex105a: Conversion between $N \times M \times 2$ and $N \times M$ polarized arrays
 - Ex105b: Transpose `/xyz`. Swaps y- and z-directions
 - Ex105c: `/zxy` (right circular) transpose followed by `/yzx` (left circular)
 - Ex105d: `/yzx` (left circular) transpose followed by `/zxy` (right circular)
 - Ex105e: `/xzy` transpose of non-cubic 3D array
 - Ex105f: Transpose of 3D array, left circular permutation
 - Ex105g: Transpose of 3D array, right circular permutation
- Ex106: Fiber-to-fiber coupling
 - Ex106a: Fiber-to-fiber coupling with a single ideal lens.
 - Ex106b: Fiber-to-fiber coupling with a single aspheric lens, using `lensgroup`.
 - Ex106c: `c## ex106b`
 - Ex106d: Fiber-to-fiber coupling with decenter at input

- Ex106e: More complex example of fiber-to-fiber coupling
- Ex106f: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), no apertures, through-focus
- Ex106g: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), no apertures, best focus
- Ex106h: Fiber-to-focus GRIN, aberration ($\alpha = 1.8$), no apertures, through-focus
- Ex106i: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), apertures implemented, through-focus.
- Ex106j: Fiber-to-focus GRIN, aberration ($\alpha = 1.8$), apertures implemented
- Ex106k: Optimization of fiber-to-fiber GRIN lens system (paraxial)
- Ex106l: Optimization of fiber-to-fiber GRIN lens system (phase aberration model)
- Ex106m: Optimization of fiber-to-fiber GRIN lens system
- Ex106n: Example of multi-mode diode laser
- Ex107: Sum frequency generation (SFG)
 - Ex107a: SFG, plane wave case
 - Ex107b: `c## ex107a`
 - Ex107c: `c## ex107b`
- Ex108: Fan-out grating
- Ex109: Flat-flat and polygon resonators
 - Ex109a: Flat-flat bare cavity resonator
 - Ex109b: `c## ex109a`
- Ex110: Beam reshaping optics
- Ex111: Guide star, ground-to-space
- Ex112: Data reduction of interferograms by Fourier Transform Method
- Ex113: Optical limiting
- Ex114: Various plot styles
- Ex115: Pulse compression with grating rhombs
- Ex116: CGH test plates for aspheric mirror testing: Burge method
 - Ex116a: Scan over one free spectral range
 - Ex116b: Scan over full separation
- Ex117: Transverse pumping with an array of laser diodes
 - Ex117a: Side pumping with geometric expansion
 - Ex117b: Side pumping with `slab/pump`
 - Ex117c: Side pumping with `slab/pump` and three rotations
 - Ex117d: Resonator with complex side pumping
- Ex118: Partial coherence for a 3D object
- Ex119: Sub-round-trip sampling of resonator
- Ex120: Multi-pass amplifier
 - Ex120a: Multipass amplifier, pumping
 - Ex120b: Multipass amplifier, propagation
- Ex121: Zigzag amplifier
 - Ex121a: Equal length mirror pair
 - Ex121b: Prism configuration
 - Ex121c: Interaction of strongly slanted beams
- Ex122: Continuous Evolution of Random Processes (movie)
 - Ex122a: Continuous evolution of smoothed random distribution

- Ex122b: Continuous evolution of atmospheric aberration
- Ex123: Encryption and decryption with a hologram
 - Ex123a: Encryption/decryption, forming sources
 - Ex123b: Encryption/decryption, complex object and point sources
 - Ex123c: Encryption/decryption with hologram, two point objects
 - Ex123d: Encryption/decryption with hologram, noise and point objects
 - Ex123e: Encryption/decryption with hologram, noise and complex objects
- Ex124: Feedback coupling into a laser from external optics
 - Ex124: Feedback coupling into a laser from external optics
- Ex125: Coherent gain model resonator
 - Ex125a: Transient short pulse behavior modeled with coherent gain
 - Ex125b: Coherent gain, short pulse, atomic linewidth
- Index

1. Introduction

This document contains examples of the use of GLAD. Input decks for each of the examples are included. Command files for all examples are provided with each sale of GLAD. Output decks and plots are included for many of the examples. Certain examples require the options sold as GLAD Pro. See Section 4.0 Options, Chapter 1, Commands Manual for a description of which commands are activated by options.

To run GLAD, Start GLAD IDE, select “GladEdit” from the “Interactive Input” window to start the editor, start a new file or open one of the existing *.inp files. Use “Init-Run” to reinitialize GLAD and run the command file from GladEdit. Examples are located in the \glad57\examples folder. See the online document Introduction to GLAD for more details.

Electronic versions of all manuals are on the distribution CD ROM. The most current version of the manuals may be downloaded from the Demo/Download section of <http://www.aor.com>.

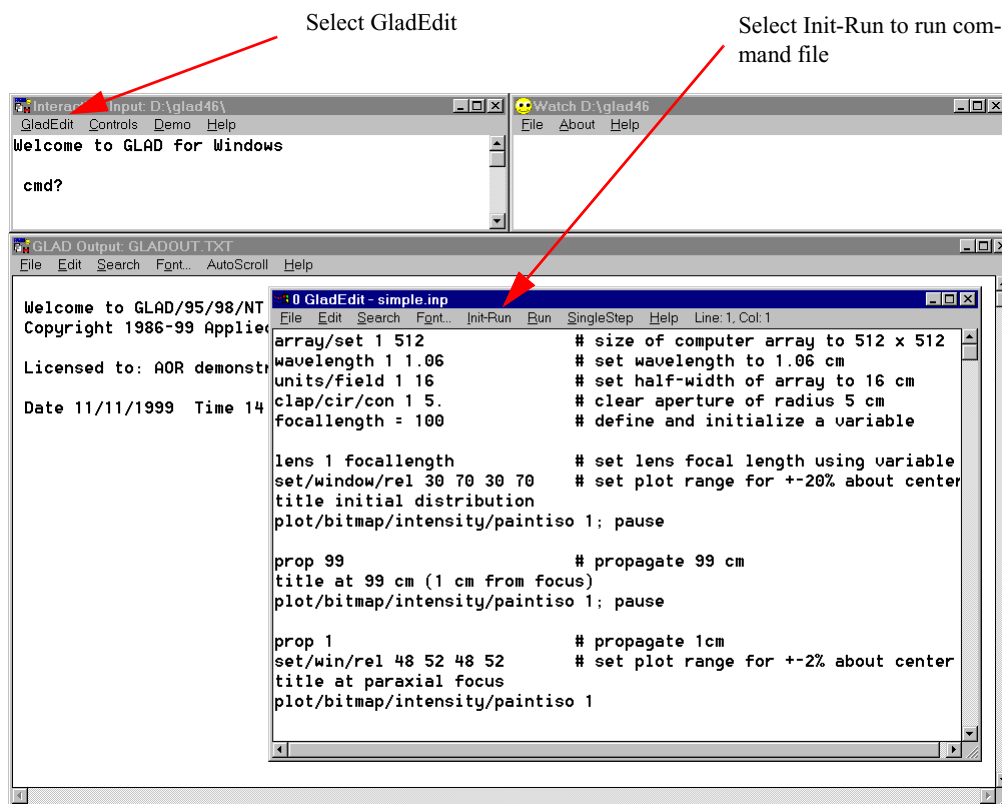


Fig. 1. Opening GladEdit and running a command file.

Ex1: INPUT Conventions and RTF command files

Table. 1.1. Table of Ex1 examples

Ex1a: Input conventions	1
Ex1b: Rich text format (RTF) command files	4

This example demonstrates some aspects of the command language, input-output control, array initialization, command prompting, the help file, and calculating a diffraction pattern of Fresnel number 5. See Chap. 1 of the GLAD Commands Manual for information on the command language.

Ex1a: Input conventions

GLAD may be used interactively, by reading command files, or by using both methods. In the Examples Manual the command files are listed. These command files are supplied with the GLAD program. To run the program interactively enter:

```
glad
```

To read a command file in the local subdirectory called ex1.inp

```
read/disk ex1.inp
```

Alternatively the file name could be given on the command line,

```
glad ex1.inp
```

Comments are used to annotate and explain the program. The user is likely to find that a generous use of comments in his or her command files will help understanding greatly and reduce mistakes. A comment line may start with “c” followed by a space of “#”. Also, any characters to the left of “#” will be ignored. The first command `echo/on` writes the commands to the screen as they are processed. The `pause` command causes GLAD to wait for input. In this example `pause` is used to allow the user to read the information on the screen. This example uses the `array` command to illustrate several different ways of entering commands and their associated data values. GLAD will also prompt for data values if a “?” is entered in the numeric field. Entering the command followed by `help` will bring up the HELP program (`gladhelp`), giving a summary of the information in the GLAD Command Manual. Graphics are displayed with the `Watch` program. See the installation instructions that came with GLAD and the `read.me` file for details on using graphics on your system. You will be able to make graphics interactively, to make off-line graphics (metafiles), to print graphics and to convert the metafiles into other commonly used graphic protocols.

In this example, the array is set to 128×128 , the wavelength is set to 1.06μ , and the units are set to give the array representing the beam a half-width of 5 cm. Most calculations will begin by setting the array size, the units (or accepting the default condition), and the wavelength.

To calculate a diffraction pattern with a Fresnel number of 5 a variable, `radius`, is defined to have the value 5. All dimensions are in centimeters, except wavelength which is in microns. Variables may be used to represent any data in GLAD and may also be used in mathematical expressions. The propagation distance, `zdist`, needed to have a Fresnel number of 5, is calculated using the definition of Fresnel number:

$$F_n = \frac{a^2}{\lambda z} = \frac{\text{radius} * \text{radius}}{\text{wavelength} * \text{zdist}} \quad (1.1)$$

where F_n is the Fresnel number, a is the aperture radius, λ is the wavelength, and z is the propagation distance. Solving this equation, we have the line:

```
zdist = radius^2/1.06e-4/5 list
```

The word `list` causes the value to be displayed after it is calculated.

This example demonstrates some aspects of the command language, input-output control, array initialization, command prompting, the help file, and calculating a diffraction pattern of Fresnel number 5. See Chap. 1 of the GLAD Commands Manual for information on the command language.

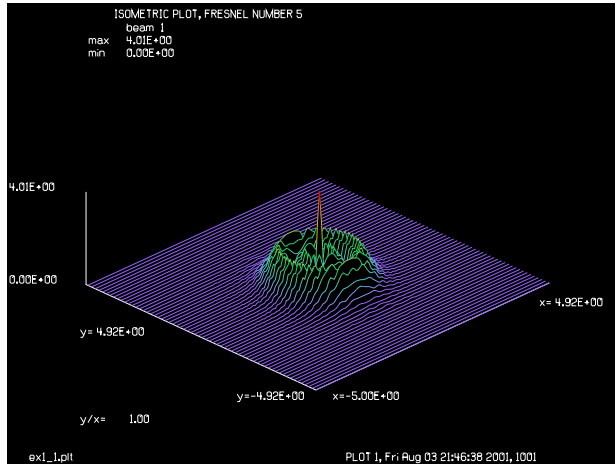


Fig. 1.1. Isometric plot of the beam irradiance, Fresnel number 5, plot/1.

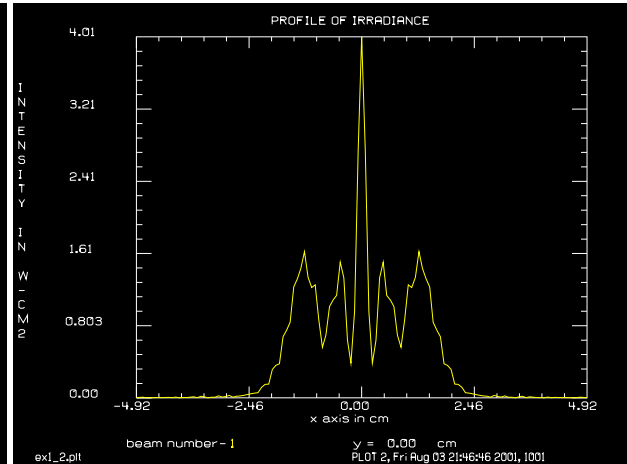


Fig. 1.2. Profile plot of the beam irradiance, plot/x/i, of Fresnel number 5.

Input: ex1a.inp

```
c## ex1
c echo/on
c
c Example 1: Input Conventions
c
c Run this example file with the commands
c
c      glad                      (system command)
c      read/disk ex1.inp        (Reads from file EX1.INP)
c
c Return to reading from the terminal by the GLAD command
c
c      read/screen
c
c pause
c A command line is illustrated in the examples and user's
c manual in the following format:
c
c      command/mod1/mod2 string value1 value2 ... value20 param/mod1/mod2
c
c      command      - command word (lower case).
c      mod1         - a modifying character string for commands
c                   or parameters.
c      mod2         - a second modifying character string.
```

Jump to: [Commands](#), [Theory](#)


```

c      string      - a character string to input a title or file name
c                    enclose string in single quotes if blanks or slashes
c                    are to be included.
c      param       - a character string which controls command execution.
c                    parameters must follow values
c      value(n)    - numerical data: numbers, variables, or assignments.
c
c pause
c Values may be entered in the sequence shown in the users manual,
c separated by blanks. An alternate method of entering values is with
c numeric assignments in the form:
c
c valuenam=number
c
c The name of the value is entered as character data followed with the
c symbol and the number to be assigned to the value. Assignments may be
c entered in any order. Value names may also be abbreviated.
c
c Notice that a parameter can be located anywhere after the command.
c A mixture of sequential and assigned values can be used, providing
c the sequential values are entered first.
c
c pause
c The beam's arrays may be initialized to a different size using
c array/set:
c
c array/set 1 64 128 1 # numbers as values
c array/set ibeams=1 nlinx=64 nliny=128 ipol=1 # numeric assignments
c arr/s i=1 nlinx=64 nliny=128 i=1 # with abbreviations
c
c These are all equivalent forms
c
c array/set 1 64 128 1
c
c The array is now dimensioned 64 x 128 with an initial beam radius of
c 2 cm. The following command line will allocate two polarization
c states for Beam 1.
c
c pause
c If want GLAD to prompt you for the values to be entered, use the
c command followed by "?"
c
c array ?
c
c You may use the prompt form for commands and modifiers as well.
c
c pause
c The default modifier for ARRAY is /LIST
c
c array
c
c Let us calculate the diffraction pattern of a collimated beam and a
c circular aperture. We will use the capability of GLAD to calculate

```

Jump to: [Commands](#), [Theory](#)

```

c mathematical expressions to simplify our setup.
c
array/s 1 128          # array size for beam 1 is 128 x 128
units/field 1 5        # array half-width is 5 cm
radius = 2             # define a variable
clap/c/c 1 radius      # clear aperture radius is 2 cm
wavelength/set 1 1.06  # wavelength is 1.06 microns
                        # calculate distance for Fresnel number = 5
zdist = radius^2/1.06e-4/5 list
pause
prop zdist             # propagate calculated distance
pause
title ISOMETRIC PLOT, FRESNEL NUMBER 5
plot/screen/pause 8
plot/w ex1_1.plt
plot/l 1 ns=64
title PROFILE OF IRRADIANCE
plot/w ex1_2.plt
plot/x/i 1
end

```

Ex1b: Rich text format (RTF) command files

This is an example of rich text (RTF) format command file. The user may select font, color, and other text formatting. Pictures may be embedded. Here we have a WMF file generated by GLAD and an equation generated by MatyType 5® as a WMF file. A hypelink command is shown that calls the command manual/destination commands.pdf gaussian. Double clicking the hyperlink will start the Adobe Acrobat Reader on the page describing the gaussian command. GLAD creates a text “shadow” file with file extension “txt” and reads commands from the shadow file, so that formatting and other RTF features are preserved. Text files may be converted to RTF files from GladEdit by using Save As and selecting RTF format. The file extension INP may be used for either TXT or RTF file. The GladEdit displays the file type in the lower left hand corner of the GLAEdit window.

Input: ex1b.inp

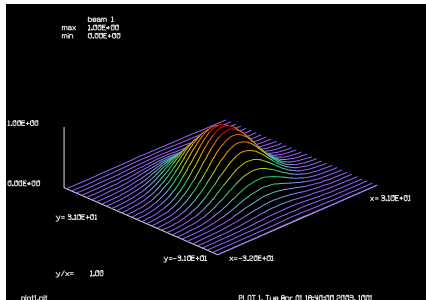
c## **ex1b.inp**

c

c This example illustrates some of the rich text format (RTF) capability.

c

c Here is an example of an embedded picture that shows a gaussian beam.



Jump to: [Commands](#), [Theory](#)

c

c Here is an example of an equation made in MathType an inserted as
c an embedded WMF file.

$$a(x, y) = \sqrt{pkflu} \exp \left[- \left(\frac{x^2 + y^2}{ro^2} \right)^{sgxp} \right]$$

c Here is a hyperlink reference to a manual page for the [gaussian](#) command

```
variable/dec/integer Pass
array/s 1 64
macro/def step/o
    Pass = Pass + 1
    gaus/c/c 1 1 [10 + Pass]
    plot/l 1; pause 2
macro/end
macro/run step/4
```


Ex2: Beam initialization and automatic units control

Table. 2.1. Table of Ex2 examples

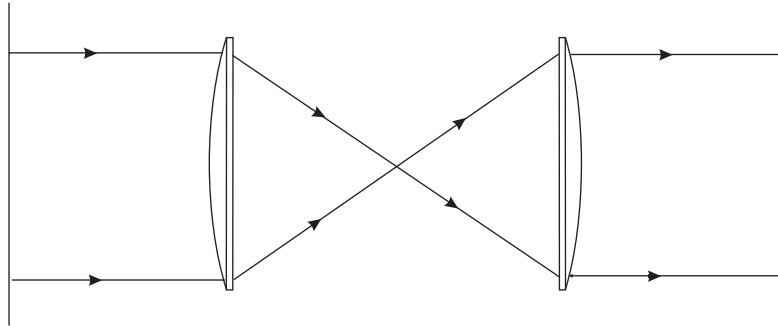
Ex2a: Gaussian and supergaussian beam generation, automatic units	3
Ex2b: Calculation of beam and array size through the waist.	5
Ex2c: Calculation of beam and array size with grating.	11
Ex2d: Calculation of beam and array size with shallow focus	14

This example shows further methods of beam initialization and introduces the subject of automatic units control. This example demonstrates the use of lenses and diffraction propagation. Profile plots are used to display the distribution.

Beam 1 is set to a gaussian distribution. Beam 2 is set to a top hat distribution. The code chooses somewhat different units to optimize the filling of the array. Beam 1 has units of 0.078 cm and Beam 2 has units of 0.057 cm.

As shown in Fig. 2.1, the optical configuration consists of 100 cm propagation from the starting distribution to a lens of 100 cm focal length. The beams are then propagated 100 cm to the intermediate focus. A further propagation brings the beams 100 cm to a recollimating lens. The last 100 cm propagation occurs after the recollimating lens. This configuration is similar to a Fourier transform lens arrangement sometimes used for coherent spatial filtering. The optical fields at the end of this sequence of operations should be identical to the starting fields (except that they are inverted). Inspection of the final field data, as displayed by the `field` command, shows that the starting fields are reproduced to high accuracy.

The word `list` causes the value to be displayed after it is calculated.



Plot 1. Gaussian beam at start.

Plot 3. Gaussian beam at focus.

Plot 5. Gaussian beam after.

Plot 2. Top hat beam at start.

Plot 4. Top hat beam at focus.

Plot 6. Top hat beam after lens.

Fig. 2.1. Optical schematic for Ex. 2. The beams proceed from the front focal point of the first lens, to the focus and finally to the rear focal plane of the second lens.

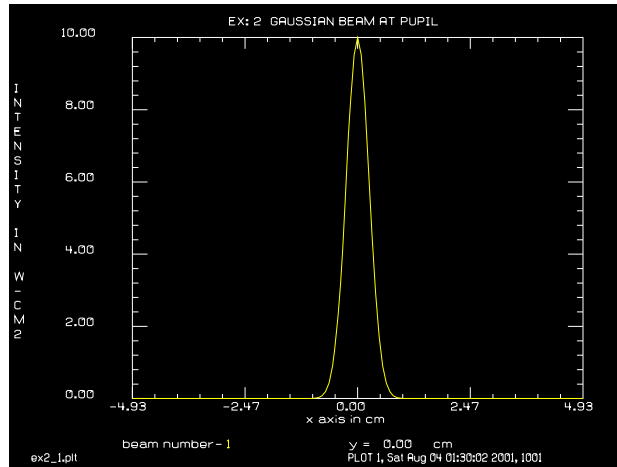


Fig. 2.2. Starting gaussian beam.

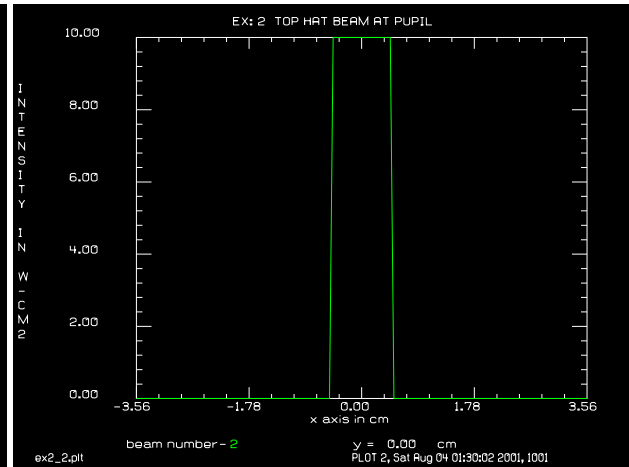


Fig. 2.3. Starting flat top beam.

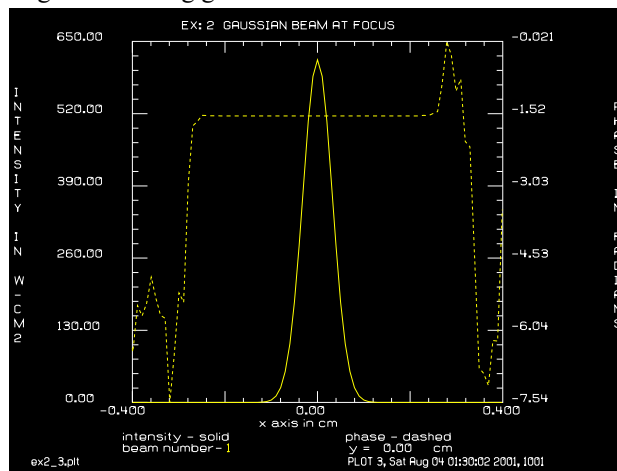


Fig. 2.4. Gaussian beam at focus.

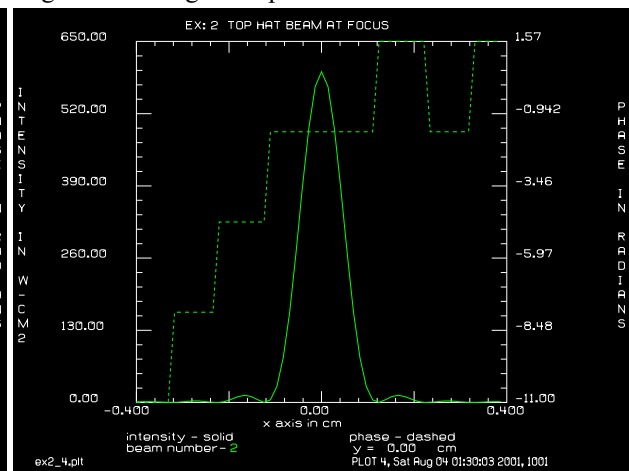
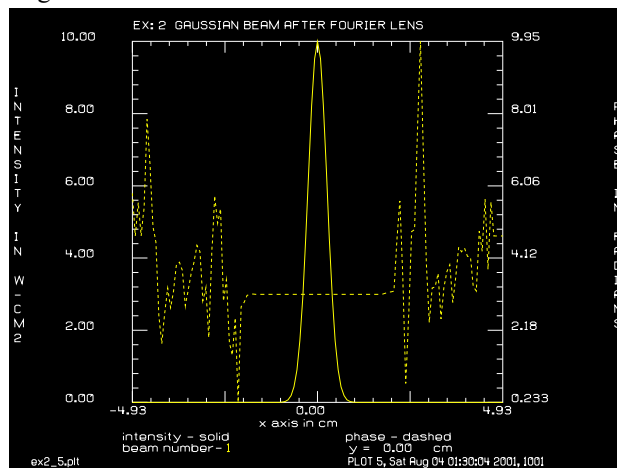
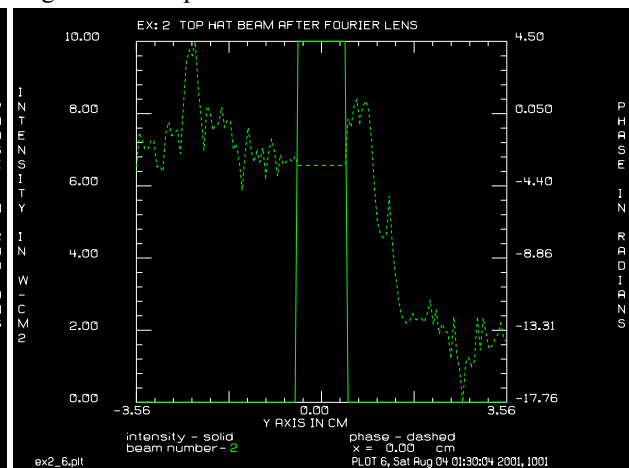


Fig. 2.5. Flat top beam at focus.

Fig. 2.6. Gaussian beam at the end of the Fourier lens.
Same as the starting distribution, Fig. 2.2.Fig. 2.7. Gaussian beam at the end of the Fourier lens.
Same as the starting distribution, Fig. 2.3.

Ex2a: Gaussian and supergaussian beam generation, automatic units

Input: ex2a.inp

```

c## ex2a
c
c Example 2a: Gaussian and supergaussian beam generation, automatic units.
c
c Initialize two beams with array sizes of 128x128 and wavelengths
c of 10 microns.
c
array/set 1 128 128          # Set Beam 1 to 128 x 128
nbeam 2                      # Expand to 2 beams, with Beam 1 size
wavelength/s 0 10.
units 1 .01
c
c Generate Beam 1 as a gaussian beam with radius .5 cm and peak
c fluence of 10. Generate Beam 2 as a top hat beam with a peak
c fluence of 10. and a radius of .5 cm.
c
c For a Gaussian beam default units are :
c     Units = sqrt(pi*r0*r0/Nline)
c           = .07833213 cm
c
c At the pupil the ratio of the field size to beam size is 9.87
c
c gaussian/[ushap]/[uscal] ibeams pkflu r0x r0y sgxp r0y decx decy
c
gaussian/cir/res 1 10. .5 1.
c
clear 2 1
mult 2 10.
c
c For top hat beams default units are:
c     Units = sqrt((rad*rad)/(.61*nline))
c           = .05658484 cm.
c
c At the pupil the ratio of the field size to beam size is 7.13
c
clap/cir/res 2 .5
c
c Show a sample of each intensity array
c
c field kbeam jyline ixstart ixend istep
c
field 1 65 65 128 5
pause
field 2 0 65 80 2
c
c Generate a single line slice plot of both beams. After each beam is
c plotted to the screen GLAD will wait for a carriage return.
c
c PLOT/XSLICE/INTENSITY Kbeam Slice Left Right Fmin Fmax

```

Jump to: [Commands](#), [Theory](#)

```

c
c Define a title for the plots
c
title EX: 2 GAUSSIAN BEAM AT PUPIL
plot/w ex2_1.plt
plot/xslice/intensity 1
title EX: 2 TOP HAT BEAM AT PUPIL
plot/w ex2_2.plt
plot/xslice/intensity 2
pause
c
c Propagate both beams 100. cm and apply a lens of 100cm focal length,
c then propagate to the focus. Check beam status.
c
dist 100.
c
c LENS Ibeams Flx (Fly) Decx Decy
c
lens 0 100.
dist 100.
status/parax
c
c For a Gaussian beam the new beam size should be :
c       $r_0' = \text{Lambda} * \text{fl} / (\pi * r_0)$ 
c      = .0010*100/(pi*.5)
c      = .06366198 cm.
c
c For a top hat beam the first dark ring should have a radius :
c       $rd = 1.22 * \text{lambda} * \text{fl} / (2. * \text{rad})$ 
c      = 1.22*.0010*100./1.
c      = .122 cm.
c
c Notice that the radii of the beams match the above predictions.
c
c At the focus the ratio of the field size to "beam" size for the
c Gaussian and top hat beam are the same as at the pupil.
c
c Generate a single line plot of each beam
c
title EX: 2 GAUSSIAN BEAM AT FOCUS
plot/w ex2_3.plt
plot/xslice 1 0 -.4 .4 0. 650.
title EX: 2 TOP HAT BEAM AT FOCUS
plot/w ex2_4.plt
plot/xslice 2 0 -.4 .4 0. 650.
pause
c
c Repeating the previous propagation sequence in reverse order should
c generate an image of the original beam. Repeating the sampling will
c provide data to check the accuracy of the propagator. Any differences
c will be a result of roundoff or other numerical errors.
c
dist 100
lens 0 100

```



```

dist 100
title EX: 2  GAUSSIAN BEAM AFTER FOURIER LENS
plot/w ex2_5.plt
plot/xslice 1
title EX: 2  TOP HAT BEAM AFTER FOURIER LENS
plot/w ex2_6.plt
plot/yslice 2
field 1 65 65 128 5
pause
field 2 0 65 80 2
pause
end

```

Ex2b: Calculation of beam and array size through the waist

The command `fitegaus/udata` may be used to calculate the beam size and array size or array filling (with `/fillx`) through the waist region. In the region of the waist the units are held constant. Outside the waist region, the units expand linearly according to a line drawn through the center of the waist. Under `set/propcontrol/on` (default) the transition is at the point where the units are equal for the constant and linearly expanding region. This transition point is at a distance of

$$zswitchx = N * units^2 / \lambda$$

where

N	the number of pixels on one side of the array
units	pixel spacing at the waist
lambda	wavelength in cm

Alternatively

$$zswitchx = \lambda * zdistx^2 / (N * units^2)$$

where

N	the number of pixels on one side of the array
units	pixel spacing at a position outside the waist region
lambda	wavelength in cm
zdistx	distance between current position and waist

In Ver. 4.7 and earlier versions of GLAD, the transition point was placed at the Rayleigh range of the surrogate gaussian. This resulted in a sudden change in units size across the boundary. A more general definition of Rayleigh range has been added at Ver. 4.8 to include an approximate expression for M^2 for the purpose of defining an effective wavelength of $M^2 * \lambda$ model the expansion of the beam in the presence of aperture clipping and/or aberration.

For a uniformly filled aperture, an M^2 or about 3 work reasonably well. Taking the waist of the equivalent gaussian to be the same as the aperture radius Apt , the equivalent gaussian Rayleigh range

Jump to: [Commands](#), [Theory](#)

```
zray = pi*Apt^2/(Msq*Lambda)
```

Prior to Ver. 4.8, the equivalent gaussian was constructed with a smaller waist than the aperture but the same wavelength. The new method using the M^2 concept is comparable in result but somewhat more intuitive.

In this case, we examine a uniformly filled aperture under the new control using continuous units and the old control based on the Rayleigh range. We also show the degree of array filling.

Input: ex2b.inp

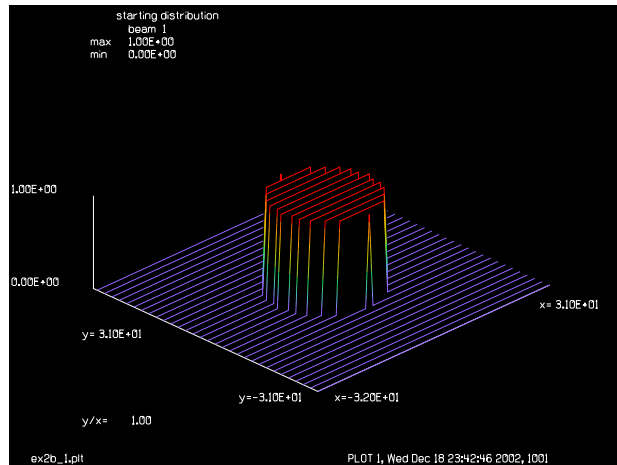


Fig. 2.8. Starting distribution for through-waist test.

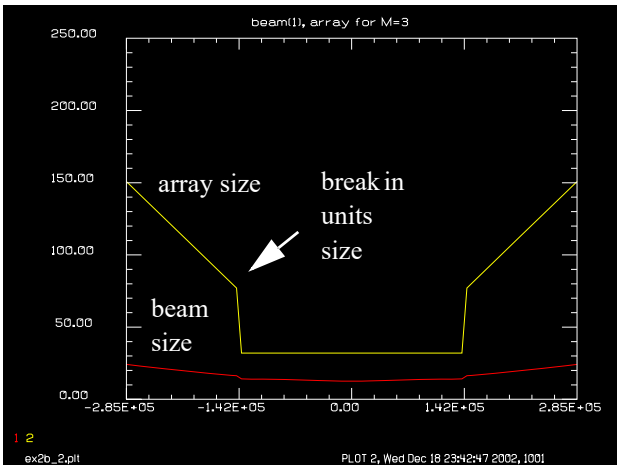


Fig. 2.9. Beam half width (red) and array half width (yellow) showing a break in units size at the transition point for the choice $M^2 = 3$. The break point is the Rayleigh range of the surrogate gaussian beam with effective wavelength $M^2\lambda$. The break point is stretched too far into the far-field.

```
c## ex2b
#
# Example 2b: Calculation of beam and array size through the waist
#
# The command fitegaus/udata may be used to calculate the beam size
# and array size or array filling (with /fillx) through the waist region.
#
# In the region of the waist the units are held constant. Outside the
# waist region, the units expand linearly according to a line drawn
# through the center of the waist.
#
# At the point of transition between the constant units and linearly
# expanding units there may be a jump in the size of the units.
#
# If the transition is too far from the waist, the beam may be aliased
# in the plane reference region (waist region) close to the transition
# region, because the transition to expanding units is made too late.
#
# If the transition is too close to the waist, the beam may be aliased
```

Jump to: [Commands](#), [Theory](#)

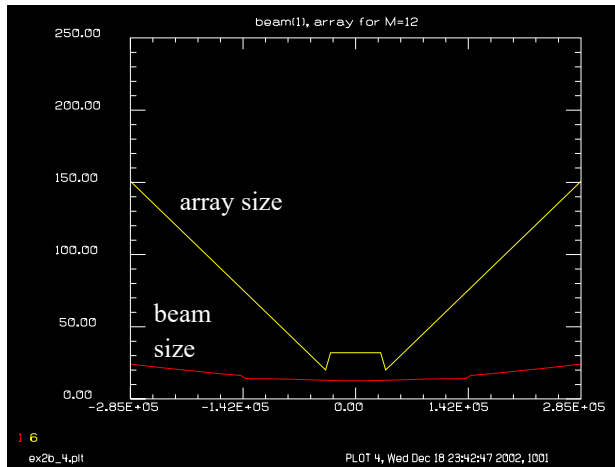


Fig. 2.10. Beam half width (red) and array half width (yellow) for $M^2 = 12$. The break point with this choice of M^2 is too close into the near field.

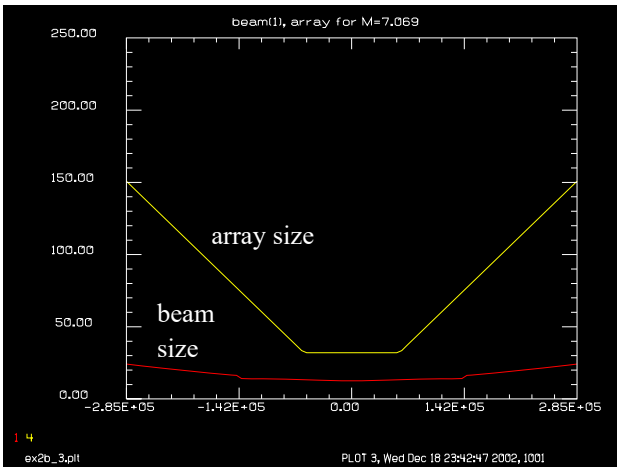


Fig. 2.11. Beam half width (red) and array half width (yellow) for the continuous unit choice of about $M^2 = 7$. This choice appears to be a good point to transition from constant to expanding units.

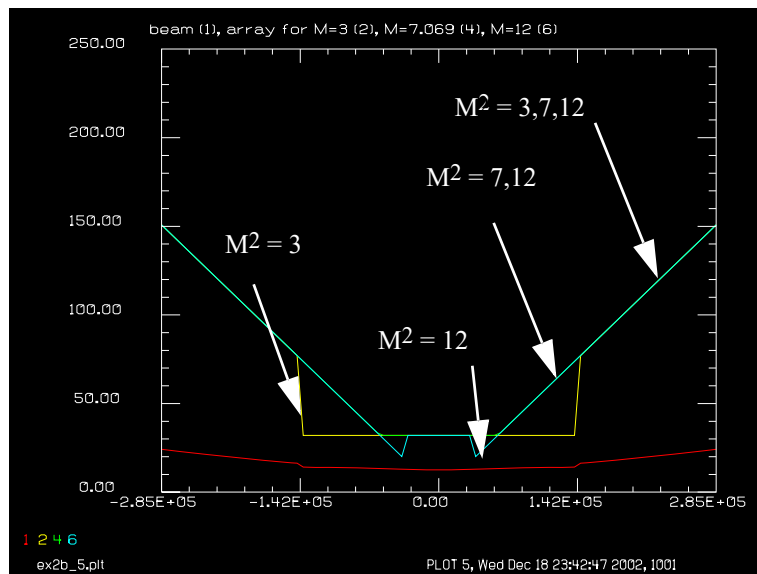


Fig. 2.12. Overlay of beam half width (red) and array half widths for $M^2 = 3, 7$, and 12 . For this example of a clear aperture, $M^2 = 3$ extends the unit transition region too far creating a sampling problem at about $z = 1.4e5$. $M^2 = 12$ makes the unit transition region too early so that a sampling problem exists at about $z = 4.5e4$. For this case, the solution for continuous units occurs at approximately $M^2 = 7$. Note that the $M^2 = 7$ and 12 overlap in the mid-field and $M^2 = 3, 7$, and 12 overlap in the far-field.

in the curved reference region close to the transition point, because
the transition to expanding units is made too early and the units
are too small.

The jump in units may be disturbing for some calculations through
the focus.

The point at which the near- and far-field units are equal is

Jump to: [Commands](#), [Theory](#)

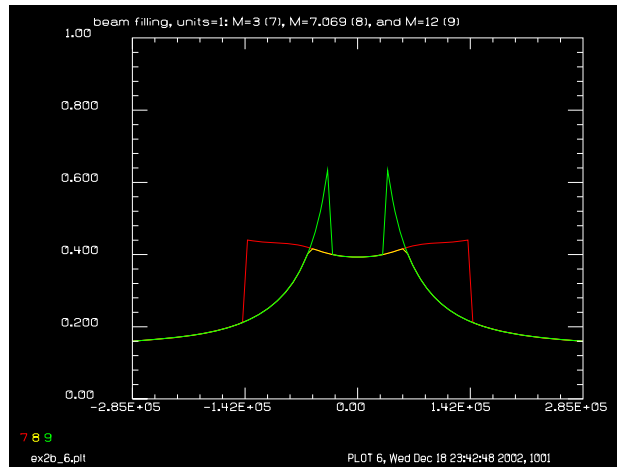


Fig. 2.13. Relative array filling for initial units of 1 cm per pixel for the same cases as Fig. 2.12: $M^2 = 3$ (red), $M^2 = 7$ (yellow), and $M^2 = 12$ (green). Sharp peaks of high array filling for $M^2 = 12$ indicate positions of higher aliasing.

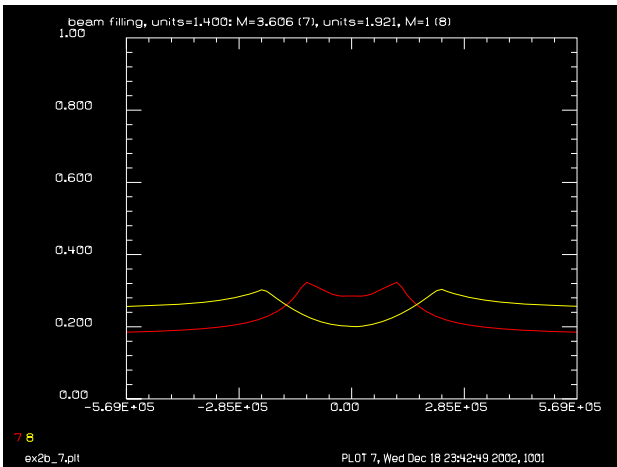


Fig. 2.14. Array filling for units = 1.4 (red) and units = 1.9 (yellow), selected by the rescale selection of the `clap` command. Both choices have lower array filling than for units = 1 cm.

```
# zswitchx = N*units^2/lambda

# where
# N          the number of pixels on one side of the array
# units      pixel spacing at the waist
# lambda     wavelength in cm

# Alternatively

# zswitchx = lambda*zdistx^2/(N*units^2)

# where
# N          the number of pixels on one side of the array
# units      pixel spacing at a poistion outside the waist region
# lambda     wavelength in cm
# zdistx     distance between current position and waist

# The actual transition in GLAD is based on the effective Rayleigh
# distance from the waist using the effective wavelength,  $M^2 \cdot \lambda$ 

# Zray = pi*Wsx^2/M^2/Lambda

# The jump in units at the transition point may be eliminated by
# setting the  $M^2$  value to make the effective Rayleigh distance the
# same as zswitchx, the point of equal units. This condition is

# Msq = pi*Apt^2/N/Units^2 list      #  $M^2$  for continuous units

# This condition for  $M^2$  results in both a continuous solution for units
```

Jump to: [Commands](#), [Theory](#)

```

# (no jumps in unit value) and relatively low aliasing near the
# transition region.

# In the special case of a uniformly filled aperture, an  $M^2$  or about 3
# works reasonably well.
# Taking the waist of the equivalent gaussian to be the same as the aperture
# radius, the equivalent gaussian Rayleigh range

#  $z_{ray} = \pi \cdot Apt^2 / (Msq \cdot \Lambda)$ 

# Prior to Ver. 4.8, the equivalent gaussian was constructed with a
# smaller waist than the aperture but the same wavelength. The new method
# using the  $M^2$  concept is comparable in result but somewhat more intuitive

# In this case, we examine a uniformly filled aperture under the new
# control using continuous units and the old control based on the Rayleigh
# range. We also show the degree of array filling.

variable/dec/int N
N = 64
Units = 1
Lambda = 1.06e-3
Apt = 12
zswitchx = N*Units^2/Lambda list      # distance for continuous units
Msq = pi*Apt^2/N/Units^2 list         #  $M^2$  for continuous units
Times=2
array/set 1 N
units 1 Units
wavelength 1 Lambda*1e4
clap/c 1 Apt
nbeam 3
copy 1 2
copy 1 3
geodata/set/msquared 2 m2x=Msq
geodata/set/msquared 3 m2x=12
plot/watch ex2b_1.plt
title starting distribution
plot/l 1
variable/set Zrayx 1 geodata/zrayx
fitegaus/udata kbeam=1 steps=91 zcenter=0 halfwidth=Times*Zrayx coll=1 col2=2
fitegaus/udata kbeam=2 steps=91 zcenter=0 halfwidth=Times*Zrayx coll=3 col2=4
fitegaus/udata kbeam=3 steps=91 zcenter=0 halfwidth=Times*Zrayx coll=5 col2=6
plot/watch ex2b_2.plt
plot/udata/set y01 y02
title beam(1), array for M=3
plot/udata/seq min=0 max=250
plot/watch ex2b_3.plt
plot/udata/set y01 y04
title beam(1), array for M=@Msq
plot/udata/seq min=0 max=250
plot/watch ex2b_4.plt
plot/udata/set y01 y06
title beam(1), array for M=12

```

Jump to: [Commands](#), [Theory](#)

```

plot/udata/seq min=0 max=250
plot/watch ex2b_5.plt
plot/udata/set y01 y02 y04 y06
title beam (1), array for M=3 (2), M=@Msq (4), M=12 (6)
plot/udata/seq min=0 max=250
fitegaus/udata/fillx kbeam=1 steps=91 zcenter=0 halfwidth=Times*Zrayx coll=1
col2=7
fitegaus/udata/fillx kbeam=2 steps=91 zcenter=0 halfwidth=Times*Zrayx coll=1
col2=8
fitegaus/udata/fillx kbeam=3 steps=91 zcenter=0 halfwidth=Times*Zrayx coll=1
col2=9
plot/watch ex2b_6.plt
plot/udata/set y07 y08 y09
title beam filling, units=1: M=3 (7), M=@Msq (8), and M=12 (9)
plot/udata/seq min=0 max=1
clear 0 1
c
c Set up beam 1 with units of 1.4
c
Units1 = 1.4
units/set 1 Units1
clap/c/c 1 Apt
Msq = pi*Apt^2/N/Units1^2 list # M^2 for continuous units
geodata/set/msquared 1 m2x=Msq
c
c Set up beam 2 with optimum units and M=1
c
clap/c/res 2 Apt
variable/set Units2 2 units
c
c Optimum units should be Apt*sqrt(1/.61/N) = 1.920553 for Apt=12
c
Msq2 = pi*Apt^2/N/Units2^2 list # M^2 for continuous units
geodata/set/msquared 2 m2x=Msq2
Times=4 # Extend the range to be displayed
fitegaus/udata/fillx kbeam=1 steps=91 zcenter=0 halfwidth=Times*Zrayx coll=1
col2=7
fitegaus/udata/fillx kbeam=2 steps=91 zcenter=0 halfwidth=Times*Zrayx coll=1
col2=8
plot/watch ex2b_7.plt
plot/udata/set y07 y08
title beam filling, units=@Units1: M=@Msq (7), units=@Units2, M=1 (8)
plot/udata/seq min=0 max=1
plot/w ex2b_8.plt
title array filling for units=@Units1
plot/l 1
plot/w ex2b_9.plt
title array filling for units=@Units2
plot/l 2

```

Ex2c: Calculation of beam and array size with grating

When a relatively high frequency grating is added, the beam spreads at a larger angle. Even if the beams are well sampled in the near-field and the extreme far-field, they may become aliased near the boundary between constant units and expanding units. In this example a grating with vertical lines, causes the beam to expand rapidly in the x-direction. The default choice of $M^2 = 3$ is adequate for the near- and far-field condition, but is not sufficient for the mid-field condition shown in Fig. 2.15. At the distance of $1e5$ cm, the x-direction shows significant aliasing.

Selecting the continuous units choice from `geodata/set/msquared/continuousunits` eliminates the unit discontinuity and also provides a good choice for the transition between near- and far-field units as shown in Fig. 2.17. Because the halfwidth of the field remains larger than the beam width, the aliasing is reduced at the mid-field region for $z = 1e5$.

A full through-waist fit of the best surrogate gaussian beam is shown for the x-direction in Fig. 2.19 and y-direction in Fig. 2.19. The associated beam display at $z = 1e5$ in Fig. 2.21 shows little aliasing.

The condition for continuous units is easy to calculate and serves well for many cases where the mid-field region needs to be displayed.

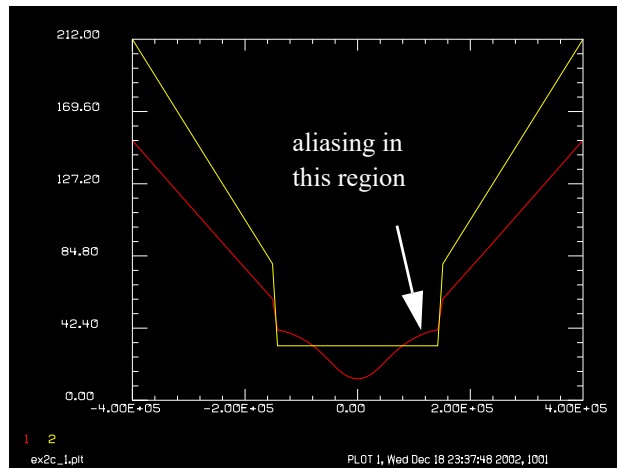


Fig. 2.15. Beam size (red) versus array half width (yellow) for the default choice of $M^2 = 3$. In this case transition to expanding units for the far-field is made somewhat too late resulting in aliasing at the intermediate position indicated.

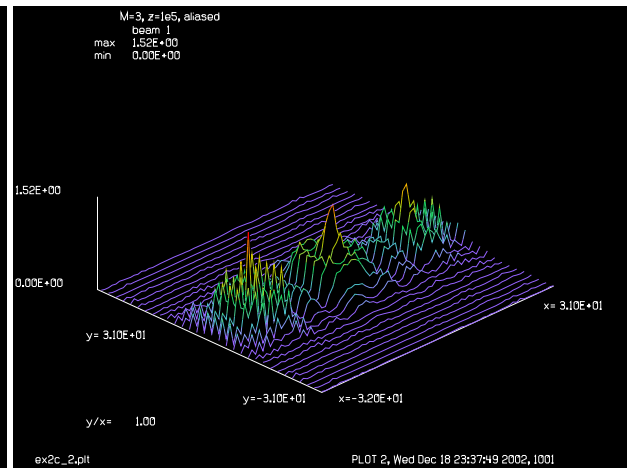


Fig. 2.16. Beam with phase grating in the intermediate region at $z = 1e5$, showing some aliasing.

Input: `ex2c.inp`

```
c## ex2c
#
# Example 2c: Calculation of beam and array size with grating
#

variable/dec/int N
N = 64
Units = 1
Lambda = 1.06e-3
```

Jump to: [Commands](#), [Theory](#)

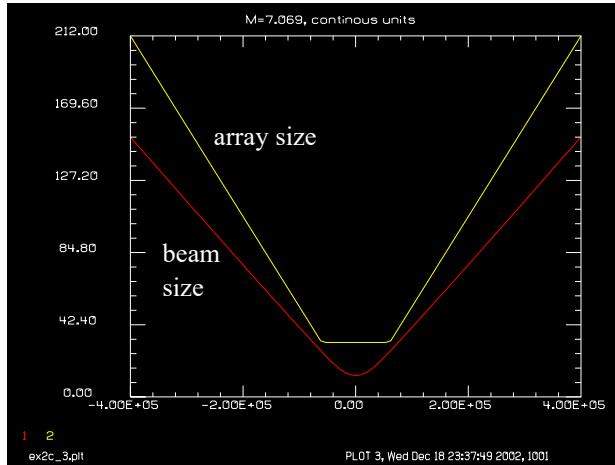


Fig. 2.17. Through-waist display of beam (red) and halfwidth of array (yellow) for a value of $M^2 = 7$ that gives continuous units. This choice also gives good sampling throughout the mid region. Both directions have the same propagation M-squared value of control.

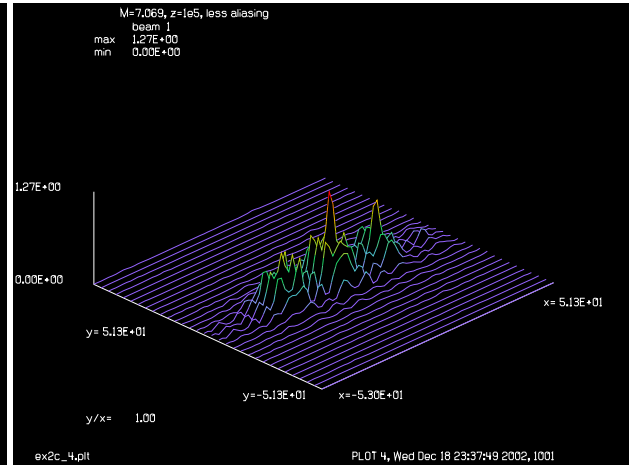


Fig. 2.18. Beam with phase grating in the intermediate region at $z = 1e5$, under control for continuous units as shown in Fig. 2.17. Aliasing is significantly reduced as compared with Fig. 2.16.

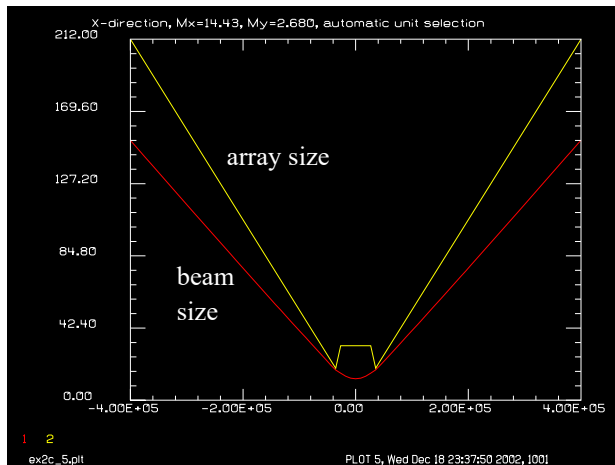


Fig. 2.19. Through-waist display in the x-direction of beam (red) and halfwidth of array (yellow) for fitegauss/automatic choice of M^2 . Beam filling is relatively good at the point of interest $z = 1e5$.

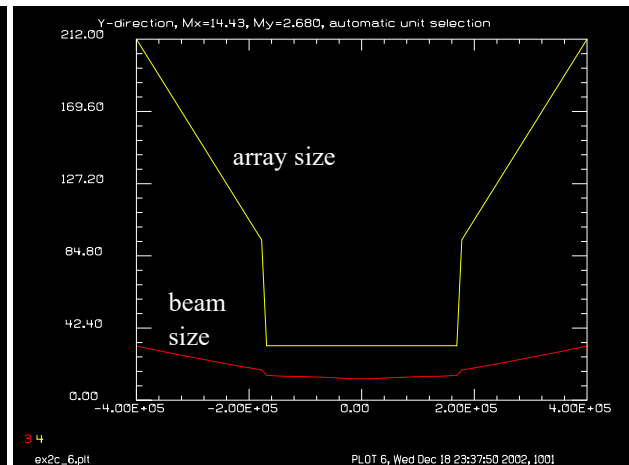


Fig. 2.20. Through-waist display in the y-direction of beam (red) and halfwidth of array (yellow) for fitegauss/automatic choice of M^2 . Because the grating only affects the x-direction, the y-direction is still in the near-field region for $z = 1e5$.

```
Apt = 12
zswitchx = N*Units^2/Lambda list # distance for continuous units
Msq = pi*Apt^2/N/Units^2 list # M^2 for continuous units
Times=2
array/set 1 N
units 1 Units
wavelength 1 Lambda*1e4
clap/c 1 Apt
#
# Add phase grating
```

Jump to: [Commands](#), [Theory](#)

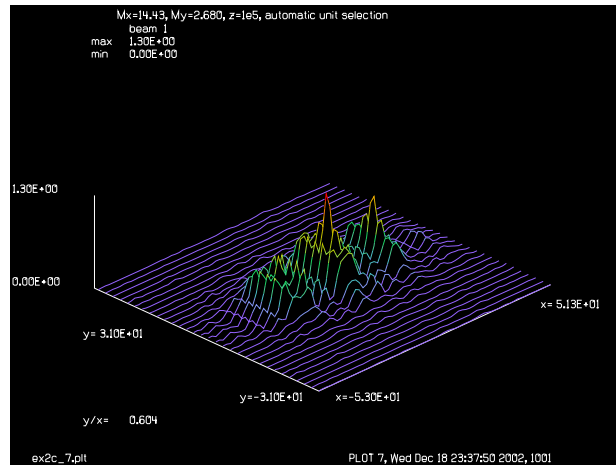


Fig. 2.21. Beam at the point $z = 1e5$ after fitegauss/automatic. Aliasing is low. The y-direction is still in the near-field with the original units of 1 cm for this case.

```
#
grating/cosine/phase 1 .2 5 az=90
#
# Calculate and display through-waist properties with default settings
#
fitegaus/udata kbeam=1 steps=91 zcenter=0 halfwidth=4e5
plot/watch ex2c_1.plt
plot/udata 1 2 min=0
prop 1e5                                # propagate to an intermediate point
plot/watch ex2c_2.plt
title M=3, z=1e5, aliased
plot/l 1
prop -1e5                                # propagate back to z = 0
#
# Set M^2 to have continuous units
#
geodata/set/msquared/continuousunits 1
variable/set M2x geodata/m2x list
variable/set M2y geodata/m2y list
#
# Calculate and display through-waist display with continuous units
#
fitegaus/udata kbeam=1 steps=91 zcenter=0 halfwidth=4e5
plot/watch ex2c_3.plt
title M=@M2x, continous units
plot/udata 1 2 min=0
prop 1e5                                # propagate to an intermediate point
plot/watch ex2c_4.plt
title M=@M2x, z=1e5, less aliasing
plot/l 1
prop -1e5                                # propagate back to z = 0
#
# automatic selection of surrogate gaussian
#
fitegaus/automatic/set
```

Jump to: [Commands](#), [Theory](#)

```

variable/set M2x geodata/m2x list
variable/set M2y geodata/m2y list
#
# Calculate and display both x- and y-through-waist characteristics
#
fitegaus/udata/x kbeam=1 steps=91 zcenter=0 halfwidth=4e5 coll=1 col2=2
fitegaus/udata/y kbeam=1 steps=91 zcenter=0 halfwidth=4e5 coll=3 col2=4
plot/watch ex2c_5.plt
title X-direction, Mx=@M2x, My=@M2y, automatic unit selection
plot/udata 1 2 min=0
plot/watch ex2c_6.plt
title Y-direction, Mx=@M2x, My=@M2y, automatic unit selection
plot/udata/set y03 y04
plot/udata/seq min=0
prop 1e5
plot/watch ex2c_7.plt
title Mx=@M2x, My=@M2y, z=1e5, automatic unit selection
plot/l 1

```

Ex2d: Calculation of beam and array size with shallow focus

For a shallow focused beam, the M^2 value determines the degree to which the waist is shifted from the paraxial focus position—generally more aberration shifts the focus more in the direction of the incoming converging beam. The condition for continuous units is set by the command

```
geodata/set/msquared/continuousunits 2
```

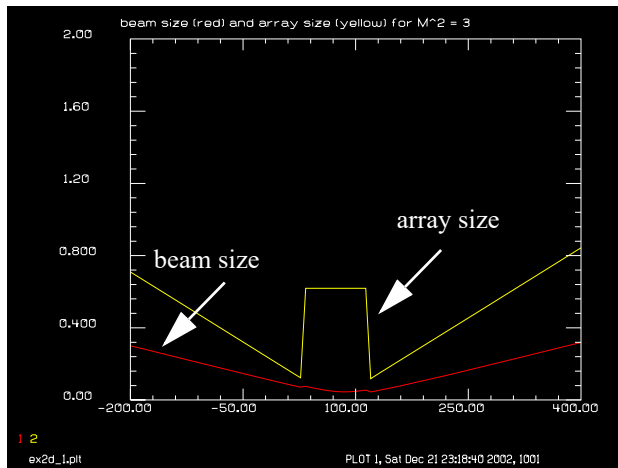


Fig. 2.22. Beam (red) and array (yellow) half widths for $M^2 = 3$ (default) for a shallow converging beam. The lens is located at $z = 0$. The waist position is shifted slightly inward from the geometrical position at $z = 100$.

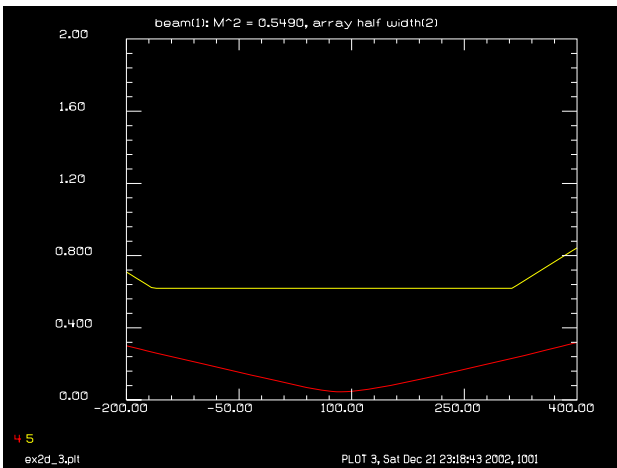


Fig. 2.23. Beam (red) and array (yellow) half widths for $M^2 = 0.55$ to have continuous units for a shallow converging beam. The lens is located at $z = 0$.

Input: ex2d.inp

```

c## ex2d
#
# Example 2d: Calculation of beam and array size with shallow focus
#

```

Jump to: [Commands](#), [Theory](#)

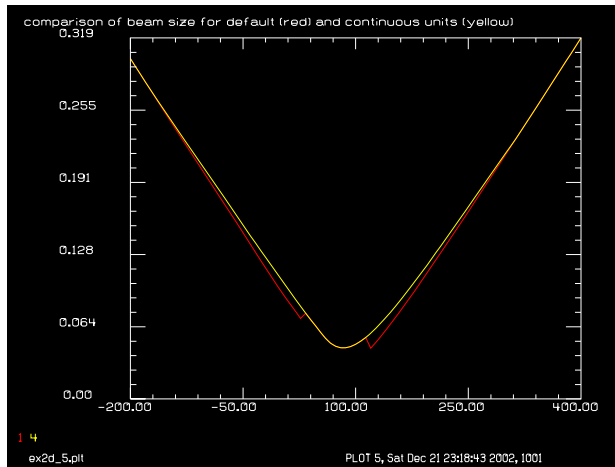


Fig. 2.24. Comparison of beam half width for default M^2 of 3 for a clear aperture (red) and for $M^2 = 0.55$ for continuous units for this case. The agreement is good.

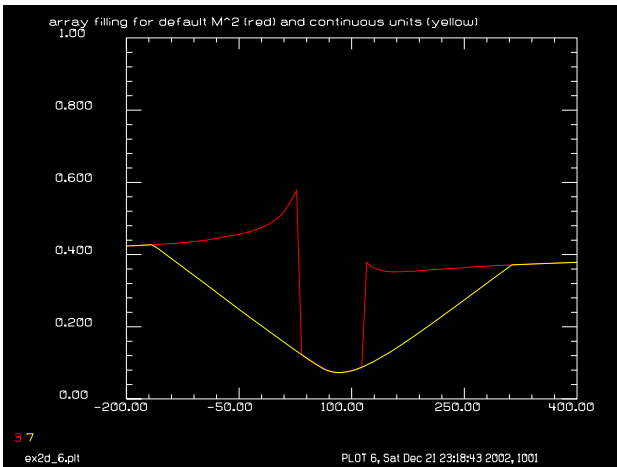


Fig. 2.25. Relative array filling for the default setting (red) and the setting for continuous units (yellow) for the shallow focus case.

```
# For a shallow focused beam, the M^2 value determines the degree to
# which the waist is shifted from the paraxial focus position --
# generally more aberration shifts the focus more in the direction of
# the incoming converging beam.
#
variable/dec/int N
N = 128
Units = .003
Lambda = .5e-4
Apt = .09
Msq = 3
zswitchx = N*Units^2/Lambda list
zray = pi*Apt^2/(Msq*Lambda) list
Steps=91
Max=2
array/set 1 N
units 1 Units
wavelength 1 Lambda*1e4
clap/c 1 Apt
nbeam 2
copy 1 2
lens 0 100
#
# Find M^2 value for continuous units, Beam 2.
#
#
# Calculate through-waist beams size (col1=1)
# and corresponding array size (col2=2) and store in udata
#
fitegaus/udata/x 1 steps=Steps col1=1 col2=2 zcenter=100 halfwidth=300
#
# Calculate through-waist beams size (col1=1)
# and corresponding array filling (col2=3) and store in udata
#
```

Default M^2 for "clap"
Transition point for continuous units
Rayleigh distance
Number of steps in through-waist analysis
Maximum for plot

Set aperture, $M^2 = 3$ by default.
Add a second beam.
Copy Beam 1 to 2
Add lens to both beams.

Jump to: [Commands](#), [Theory](#)

```

fitegaus/udata/fillx 1 steps=Steps coll=1 col2=3 zcenter=100 halfwidth=300
#
# Plot stored values for beam size and array size, Beam 1.
#
plot/watch ex2d_1.plt
title beam size (red) and array size (yellow) for  $M^2 = 3$ 
plot/udata/set y01 y02
plot/udata/seq left=-200 right=400 min=0 max=Max
#
# Propagate to focus
#
focus/apply/waist 1
variable/set Zreff1 1 zreff
plot/watch ex2d_2.plt
title at waist,  $M^2 = 3$ , Zreff = @Zreff1
plot/l 1
#
# Calculate through-waist properties for Beam 2 with  $M^2$  set for continuous
# units. Beam size (coll=4) and array size (col2=5) for udata.
#
geodata/set/msquared/cont 2
variable/set M2x 2 geodata/m2x
fitegaus/udata 2 steps=Steps coll=4 col2=5 zcenter=100 halfwidth=300
#
# Calculate through-waist properties for Beam 2 with  $M^2$  set for continuous
# units. Beam size (coll=6) and array filling (col2=7) for udata.
#
fitegaus/udata/fillx 2 steps=Steps coll=6 col2=7 zcenter=100 halfwidth=300
#
# Plot stored values for beam size and array size, Beam 2, continuous units
#
plot/watch ex2d_3.plt
plot/udata/set y04 y05
title beam(1):  $M^2 = @M2x$ , array half width(2)
plot/udata/seq left=-200 right=400 min=0 max=Max
#
# Propagate to focus, Beam 2.
#
focus/apply/waist 2
variable/set Zreff2 2 zreff
plot/watch ex2d_4.plt
title at waist,  $M^2 = @M2x$ , Zreff = @Zreff2
plot/l 2
#
# Comparison of beam size of Beam 1 (red) and Beam 2 (yellow).
#
plot/watch ex2d_5.plt
plot/udata/set y01 y04
title comparison of beam size for default (red) and continuous units (yellow)
plot/udata/seq min=0
#
# Comparison of array filling for Beam 1 and Beam 2.
# Constant unit case of Beam 2 is generally better near transition region.
#

```

Jump to: [Commands](#), [Theory](#)

```
plot/watch ex2d_6.plt
plot/udata/set y03 y07
title array filling for default  $M^2$  (red) and continuous units (yellow)
plot/udata/seq left=-200 right=400 min=0 max=1
```


Ex3: Unit selection

This example shows some of the possible forms of unit selection. The theory describing unit selection and sampling considerations is contained in GLAD Theoretical Description. If the units are chosen to be small, the distribution fills the array more fully in the near-field but may be under-resolved in the far-field. If the units are chosen large, the near-field is under-resolved and the far-field distribution is spread out. This may lead to aliasing errors in the far-field distribution. By default, the code will choose units to make the relative filling of the array equal in the near- and far-fields. If the user plans to work only in the near-field, the distribution may be spread out by choosing the units smaller. However, the near-field may alias if the units are chosen too small.

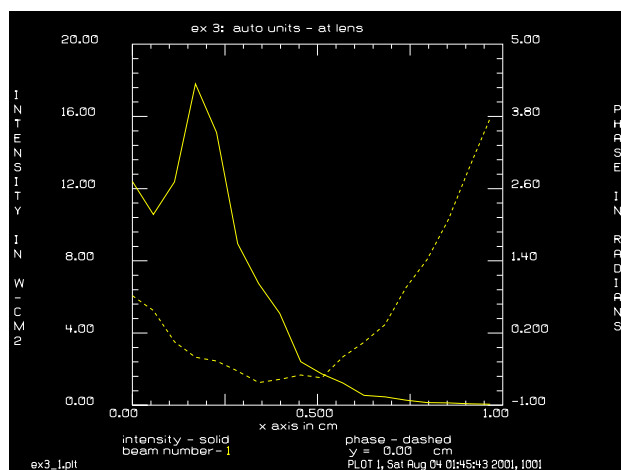


Fig. 3.1. Case 1, automatic units, at the lens.

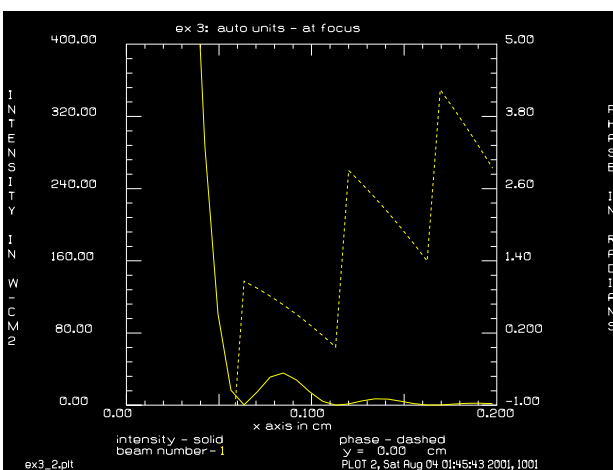


Fig. 3.2. Case 1, automatic units, at the focus.

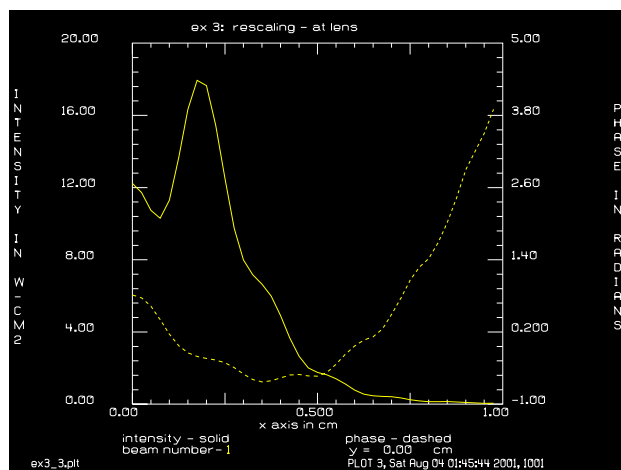


Fig. 3.3. Case 2, units = 0.25, (about half the automatic units selection), with lens rescaling, at the lens.

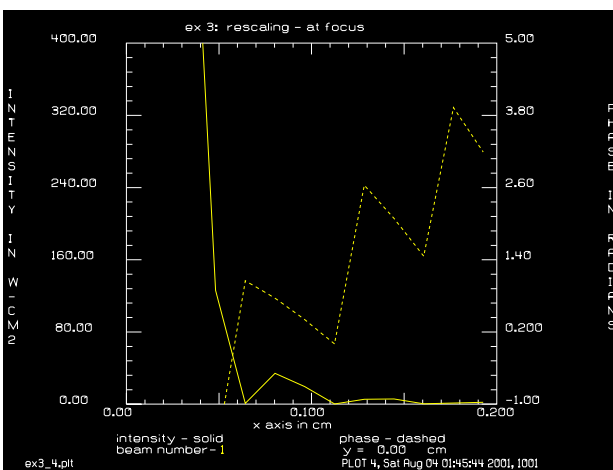


Fig. 3.4. Case 2, units = 0.25, (about half the automatic units selection), with lens rescaling, at focus.

Input: `ex3.inp`

```
c## ex3
c
c Example 3: Selection of Matrix Units
```

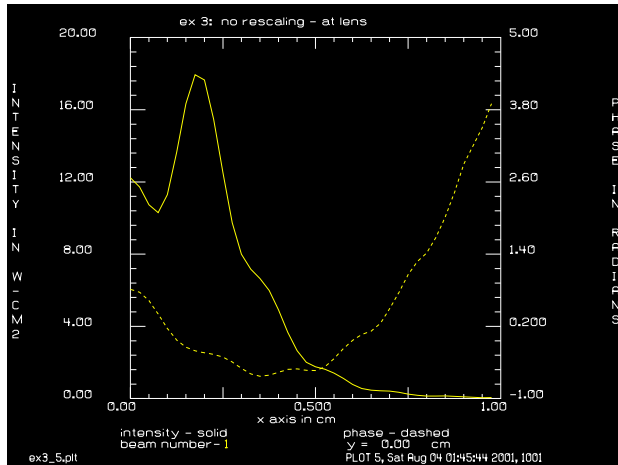


Fig. 3.5. Case 3, units = 0.25, no lens rescaling, at the lens.

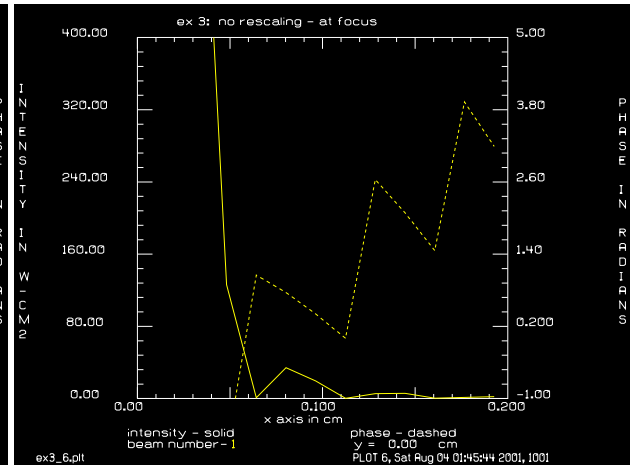


Fig. 3.6. Case 3, units = 0.25, with no lens rescaling, at focus.

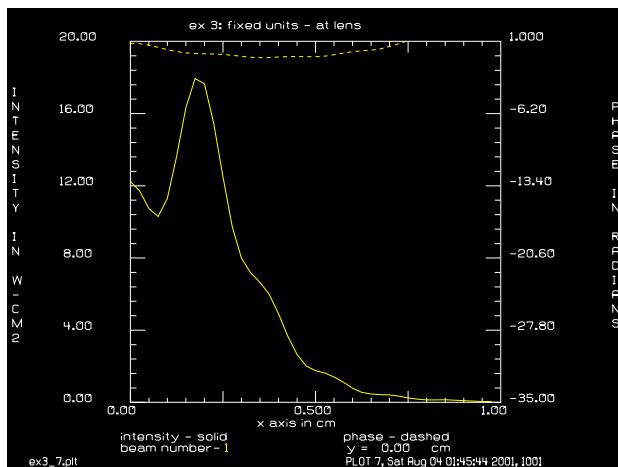


Fig. 3.7. Case 4, units = 0.25, no lens rescaling, no rescaling in propagation, at the lens.

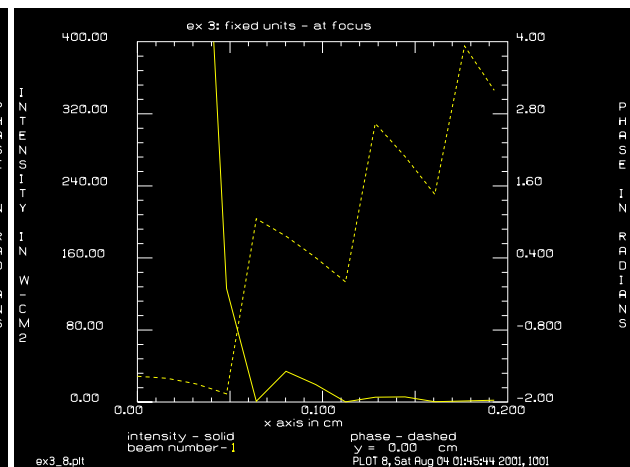


Fig. 3.8. Case 4, units = 0.25, with no lens rescaling, no rescaling in propagation, at focus.

c
c This example propagates a supergaussian beam of order 10 to a lens
c and then to the focus of the lens with different means of controlling
c the units. For comparison, data will be shown at the lens and
c the focus for four different techniques.

c
c Cases
c ----
c 1) fully automatic unit control
c 2) initial units set to half the automatic units to
c show more resolution in the near-field.
c 3) user defined units without rescaling
c 4) user defined fixed units, no rescaling
c or far-field propagation

c
c Unit changes at the Rayleigh range boundary are described
c in the GLAD Theoretical Description Manual.
c Once the units have been defined, there are two ways that GLAD
c may change the units. When the geometry of the beam changes,

Jump to: [Commands](#), [Theory](#)


```

c  such as at an aperture or a lens, GLAD may rescale the units.
c  When the beam position is outside the Rayleigh range, and the
c  units have not been 'fixed', any propagation step will change
c  the units.
c
wavelength/set 1 10.
array/set 1 128
pause
c
c  Case #1: fully automatic unit selection.
c
c  Generate Beam 1 as a tenth order supergaussian beam with
c  radius .5 cm and peak fluence of 10.
c  The units will be defined by the code to be .05684 cm., based
c  on the algorithm given in GLAD Theoretical Description.
c
gaussian/cir/res 1 10. .5 10.          # form gaussian distribution
plot/x 1 0 0 1 0 20 -1 5.
read/scr
dist 150.                             # propagate 150 cm. to the lens
lens 1 50.                             # apply lens with f=50 cm.
title ex 3:  auto units - at lens
plot/w ex3_1.plt
plot/x 1 0 0 1 0 20 -1 5.
read/scr
dist 50.                              # propagate to lens focal plane.
field 1 65 65 70 1
title ex 3:  auto units - at focus
plot/w ex3_2.plt
plot/x 1 0 0 .2 0. 400. -1. 5.
c
c  Case #2: user defined units with rescaling.
c
c  Define units to be .025 cm.  Reset beam position to 0.  Regenerate
c  top hat beam with the /CON modifier to conserve units and propagate
c  to lens.  The units are chosen to be about half of those used
c  in Case #1.
c
units/s 1 .025                         # define units to be .025 cm.
zreff/se 1 0.                         # reset z-reference to 0.
gaussian/cir/con 1 10. .5 10.         # gaussian formed with no rescale.
dist 150
lens/sph/res 1 50                     # allow lens to rescale distribution.
title ex 3:  rescaling - at lens
plot/w ex3_3.plt
plot/x 1 0 0 1 0 20 -1 5.
dist 50
title ex 3:  rescaling - at focus
plot/w ex3_4.plt
plot/x 1 0 0 .2 0. 400. -1. 5.
c
c  Case #3: user defined units without rescaling.
c
c  This case is the same as #2 except that the rescaling at the lens
c  is prohibited.
c
units/s 1 .025
zreff/se 1 0.
gaussian/cir/con 1 10. .5 10.

```

Jump to: [Commands](#), [Theory](#)

```
dist 150
c
c The /con modifier is the default
c
lens/sph/con 1 50          # prohibit rescale at lens (default).
title ex 3: no rescaling - at lens
plot/w ex3_5.plt
plot/x 1 0 0 1 0 20 -1 5.
dist 50
field 1 65 65 70 1
title ex 3: no rescaling - at focus
plot/w ex3_6.plt
plot/x 1 0 0 .2 0. 400. -1. 5.
c
c Case #4: fixed units, no rescaling or far field propagation.
c
units/s 1 .025
zreff/se 1 0.
units/fix          # prohibit units change with diffraction.
c
c With units fixed the /con modifier is not needed
c
gaussian 1 10. .5 10.
dist 150
lens 1 50
title ex 3: fixed units - at lens
plot/w ex3_7.plt
plot/x 1 0 0 1 0 20 -35 1
dist 50
field 1 65 65 70 1
title ex 3: fixed units - at focus
plot/w ex3_8.plt
plot/x 1 0 0 .2 0. 400. -2. 4.
end
```

Ex4: Variables, expressions, and mathematical surfaces

Variables and mathematical expressions may be used to simplify the development of command files. Variables are classified as real unless specifically declared as integer by the `variables/declare/integer` command. Refer to Chap. 1, Commands Manual for a complete description.

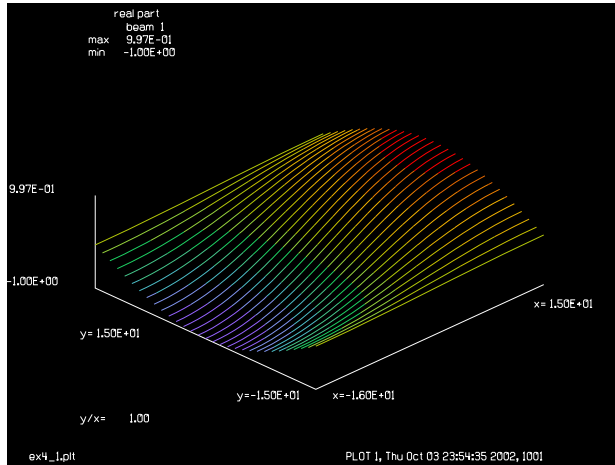


Fig. 4.1. Real part of mathematical surface.

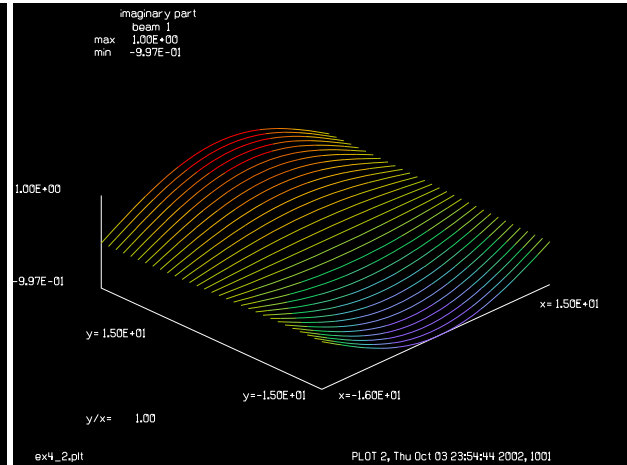


Fig. 4.2. Imaginary part of mathematical surface.

Input: ex4.inp

```
c## ex4
echo/on
c
c Example 4: Mathematical expressions and mathematical pupil function
c
c
c This example illustrates how to define a pupil function using the
c equation feature of glad
variab/declare/int i j nsize nc      # declare integer variables
                                     # real variables will be automatically
                                     # declared

c
c first test the equation parser
c
x = sin(cos(5e-1)) list               # should be .7692
pause
x = max(3,min(11,9),1,3) list         # should be 9
pause
x = max(3,min(11,9/2-1),1,3) list     # should be 3.5
pause
x = 180*atan2(1,1)/pi list            # should be 45
pause
x = 2+3*4 list                        # should be 14
pause
x = ( 2 + 3 ) * 4 list                # should be 20
pause
x = (2+3)*4+power(3/2,.1+.2) list     # should be 21.1294
pause
x = 2*-3 list                         # should be -6
pause
```

```

x = srand(11) list
                                # srand sets random number generator
                                # and returns 0
pause
x = rand() list                 # should be .4924198
pause
c
c  variables may be defined as part of command lines
c  note the distinction between variables x and y and the
c  numerical assignments xunit and yunit abbreviated to x and y
c
x = sin(pi/2) list              # should be 1
units 1 x=[y^2] y=[y=x+1.5]
y=                               # should be 2.5
units                           # should be unitsx=6.25 unitsy=2.5
pause
nsize = 32
arra/s 1 nsize                  # set array size
echo/off
clear 1 0
nc = [(nsize/2)+1]
j = 1
macro/def inner/o
c
c  define real and imaginary parts of mathematical aperture
c
    x = i-nc
    t1 = sin(x/10.)*cos(y/10.))
    t2 = cos(x/10.)*sin(y/10.))
    point/set 1 i j t1 t2
    i = i+1
macro/end
macro/def outer/o
    y = nc-j
    i = 1
    macro inner/nsize
    j = j+1
macro/end
echo/on
c
c  The next operation defines each point of a 32 x 32 array
c  It takes about 2 minutes on a 486/66 MHz machine
c
echo/off
time/i
macro outer/nsize
time
plot/screen/pause 8
title real part
plot/watch ex4_1.plt
plot/l/r 1
title imaginary part
plot/watch ex4_2.plt
plot/l/a 1
end

```

Ex5: Simple lenses and mirrors

In this example, some additional capability to use lenses and mirrors off-axis is illustrated. Many laser systems may be represented by the paraxial commands, although some “unfolding” of the system may be required.

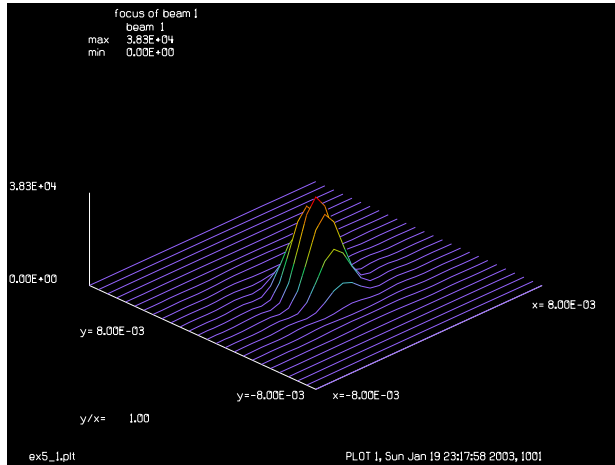


Fig. 5.1. Focus of centered beam.

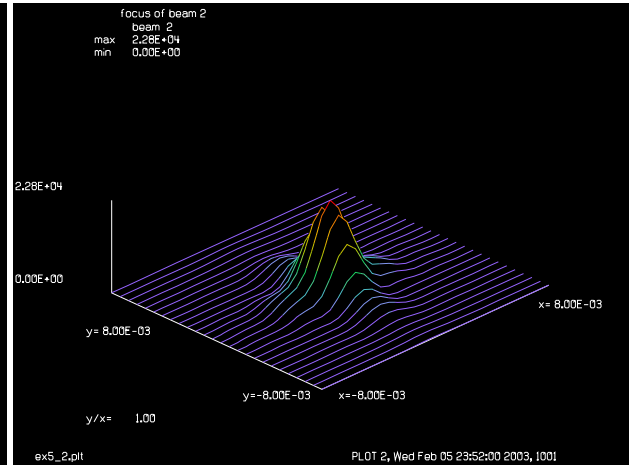


Fig. 5.2. Focus of offset beam.

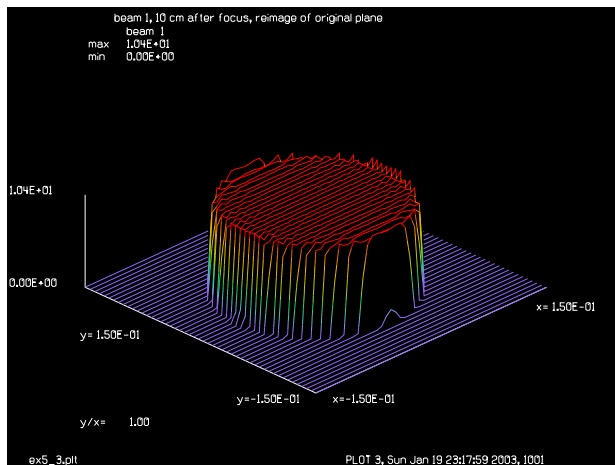


Fig. 5.3. Reimaged pupil of centered beam.

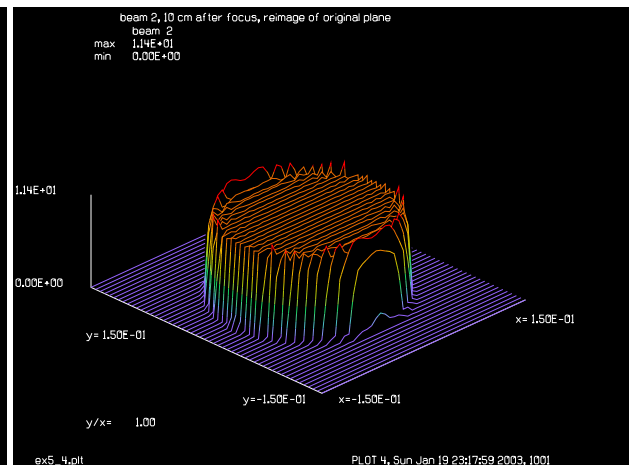


Fig. 5.4. Reimaged pupil of offset beam.

Input: `ex5.inp`

```
c## ex5
c
c Example 5: Paraxial Mirrors, On- and Off-axis.
c
c Two beams are used in this example. Beam 1 is on-axis,
c Beam 2 displaced off-axis. Both beams are propagated to a
c converging mirror then propagated through the focus to the image.
c
c Generate initial top-hat beams, and propagate to mirror.
c
array/s 1 128          # set 128 x 128 array
nbeam 2                # define 2 beams with default size of
```

```

                                # of 64 x 64
wavelength/set 0 .5
c
c Since the propagation goes through the focus the default units will
c not be required for this example. Define the units with a field
c size of .15 for the top-hat beams of radius .1.
c
units/field 0 .4
gaussian/cir/con 1 10. .1 50
copy 1 2                                # copy Beam 1 to Beam 2
global/def 2 4.7                        # decenter Beam 2 by 4.7 cm.
pause
vertex/locate/abs 0 0 20
mirror/global -20.                      # apply mirror with r=-20 cm.
status/global
c
c Notice that the beams have reversed direction and each has the same
c Z-position.
c
prop 10.                                # propagate 10 cm to focus
c
c Notice that the propagation lengths for both beams are identical,
c the distance added by the slightly diagonal path for the offset
c beam is ignored.
status/global
plot/w ex5_1.plt
title focus of beam 1
plot/l 1 xrad=8e-3 ns=128
plot/w ex5_2.plt
title focus of beam 2
plot/l 2 xrad=8e-3 ns=64
geodata
pause
prop 10.                                # propagate 10 cm to focus
c
c Note that Beam 2 has moved to x = -4.7 cm.
c
status/global
plot/w ex5_3.plt
title beam 1, 10 cm after focus, reimage of original plane
plot/l 1 xrad=.15 ns=64
plot/w ex5_4.plt
title beam 2, 10 cm after focus, reimage of original plane
plot/l 2 xrad=.15 ns=64

```

Ex6: Conic mirrors and three-dimensional orientation

This example illustrates the use of the GLOBAL commands to perform the same problem as Example 5. Accurate optical path Lents will be observed for the decentered beam. These features were developed to model grazing incidence optics and to calculate accurate round trip transit times of the optical pulse in free electron laser (FEL). The global commands add considerable power to capability of complex configurations.

Among the commands used in this example are

```
global/define
vertex/locate
vertex/rotate
mirror/global/conic
global/list
prop/zproj/absolute
```

Input: ex6.inp

```
c## ex6
c
c Example 6: Conic Mirrors and Three-Dimensional Orientation
c
c This example illustrates three-dimensional orientation of
c the beams and mirrors using conic section surfaces. The same
c beams and geometry from Example 5 will be modeled with accurate
c three-dimensional treatment. The three-dimensional features are
c helpful in determining the orientation of the beam for complex
c systems.
c
c Initialize the beam arrays and fields and define global position of
c each beam.
c
nbeam 2                                # define 2 beams with default size
                                      # of 64 x 64
wavelength 0 .5
units/field 1 .15                      # define units by field half-width.
c
c GLOBAL/DEFINE Ibeams X Y Z
c
global/define 1 0 0 0                  # define Beam 1 global coordinates.
gaussian/cir/con 1 10. .1 50
copy 1 2
global/define 2 5.                    # define Beam 2 with offset.
c
c Define the position of the vertex of the parabolic mirror. A
c propagation step is not needed to position the beam at a mirror
c surface using the MIRROR/GLOBAL command. GLAD computes the intercept
c of the chief ray of each active beam and propagates to that point.
c The BEAMS command is used to define the active beams that function
c with the MIRROR/GLOBAL and the PROP commands. By default all
c beams are active.
c
c VERTEX/LOCATE/ABSOLUTE X Y Z
c
vertex/locate/absolute 0 0 10          # locate vertex at z = 10 cm.
status
c
```

```
c Notice that the STATUS command lists direction cosines and the three
c dimensional position of each beam.
c
c MIRROR/GLOBAL/CONIC Rad CC = -1
  mirror/global/conic -20 -1.          # Apply the parabolic mirror.
  status
  status/gaus
  global/list
c
c Notice that the Z-position of the two beams are different because the
c sag of the mirror has been accounted for and each beam has a different
c path length accordingly. Also note the difference in the distances
c to the waist.
c
c Propagate to the Z = -10 plane and note the differences in path lengths
c and field data.
c
c PROP/ZPROJ/ABSOLUTE Projpl
c
  prop/zproj/absolute -10.          # propagate 10 cm. along the z-axis.
  global/list
  intens 1
  intens 2
  end
```


Ex7: Command: `mirror/global`

This example illustrates two modes of the `mirror/global` command. The global commands extend the usual physical optics capability to include some features of geometrical optics codes which are able to treat very complex configurations with large misalignments. The astigmatic mode gives improved accuracy over the paraxial approximation. The exact mode gives accurate OPD calculations but takes longer to calculate. Isometric plotting is also illustrated with control of the plot window. Numerical assignments are demonstrated as well as the use of parameters.

In this example, we examine the aberrations of a spherical mirror in a collimate beam. The astigmatism mode show the reflected beam without aberration, except for slight focus error resulting from the GLAD code choosing a gaussian beam waist as the center of the reference sphere rather than the geometric focus of the mirror.

The mirror is spherical with radius = 50 cm. To clearly illustrate the phase effects, we use a top hat function. The radius of the top hat function is 2.5. The units are chosen to be 0.125 to put 20 points across the radius. The code chooses a surrogate gaussian beam radius to 2.5 cm. The surrogate gaussian beam will come to a focus at a distance z .

$$z = \frac{f}{1 + \left(\frac{\lambda f}{\pi \omega^2}\right)^2} = 24.9995901666 \text{ cm} \quad (7.1)$$

$$\omega_0 = \frac{\omega}{\left[1 + \left(\frac{\pi \omega^2}{\lambda f}\right)^2\right]^{1/2}} = 0.01012217112 \text{ cm} \quad (7.2)$$

where

f = mirror focal length = 25 cm

λ = wavelength = 0.00106 cm

ω_0 = equivalent gaussian radius = 2.5 cm

The center of the phase front is a point 25.0 cm from the mirror at the geometrical focus, so there is a slight focus error. This error expressed in radians of phase is

$$\text{phase} = \frac{\pi}{\lambda} r^2 \left(\frac{1}{z} - \frac{1}{f} \right) = r^2 \frac{\lambda f}{\pi \omega^4} = 0.0121467049188 \text{ rad} \quad (7.3)$$

where $r = 2.5$ cm.

The value of 0.012154 radians (with some numerical error) show in the intensity listing at the edge of the aperture. This effect will be less at shorter wavelength.

The `exact` parameter invokes exact ray tracing. Spherical aberration shows up in this calculation. The additional aberration at a beam radius of 2.5 cm is

$$\text{spherical aberration} = -\frac{2\pi}{\lambda} \left[2(R - \sqrt{R^2 - r^2}) - \frac{r^2}{R} \right] = -0.463668319389 \text{ rad} \quad (7.4)$$

The total aberration is calculated by these equations to be $0.0121467049188 - 0.463668319389 = -0.463668319389$ which compares well with the value calculated with the `exact` parameter. GLAD issues TROOT failures where the ray tracing has failed. Failures occur in the vicinity of the aperture edge where the ray directions are ill-determined and generally do not indicate a significant error in the calculation.

$$z = \frac{cr^2}{1 + \sqrt{1 - (1 + \kappa)c^2 r^2}} + \sum_{i=4}^{\infty} a_i r^i \quad (7.5)$$

$$z_{\text{sphere}} = \frac{cr^2}{1 + \sqrt{1 - c^2 r^2}} \approx \frac{1}{8} c^3 r^4 + \frac{5}{64} c^5 r^6 + \dots = 1 \times 10^{-6} r^4 + 2.5 \times 10^{-10} r^6 + \dots \quad (7.6)$$

This example shows four cases:

- 1) spherical mirror with `astigmatic` parameter (paraxial)
- 2) spherical mirror with `exact` parameter
- 3) parabolic mirror with `exact` parameter
- 4) spherical mirror with aspheric coefficients to simulate parabolic surface with `exact` parameter

Case 1 shows a slight defocus due to the choice of reference surface at the surrogate gaussian waist. Case 2 shows use of the `exact` parameter which calculates the spherical aberration. Case 3 shows a parabolic mirror with use of the `exact` parameter. There is no spherical aberration in this case, so the phase error is due only to focus as in Case 1. It is best to use the simplest model which is accurate—for this problem Case 1 is better than Case 3 since it is quicker. The calculations in this example will show variations, particularly on 32 bit machines, because of differences in the accuracy of floating point calculations on different computers. Case 4 adds aspheric terms to a spherical mirror to simulate the parabolic mirror.

Table. 7.1. This is a title

Case	Description	analytical calculation (radians)	GLAD calculation (radians)
1	Idealized model, using astigmatic option. Shows defocus due to difference between beam wavefront and reference surface.	0.0121467049188	0.012154
2	Spherical mirror, exact model.	-0.451521614470	-0.450238
3	Parabolic mirror, exact model. Phase is essentially the same as the ideal model, Case 1.	0.0121467049188	0.012242
4	Spherical mirror, corrected to parabola with aspheric coefficients. Similar to Case 1.	0.0121467049188	0.012331

Input: ex7.inp

```
c## ex7
c
c Example 7: Spherical, Conic, and Aspheric Mirrors, Exact Ray Tracing
```

Jump to: [Commands](#), [Theory](#)

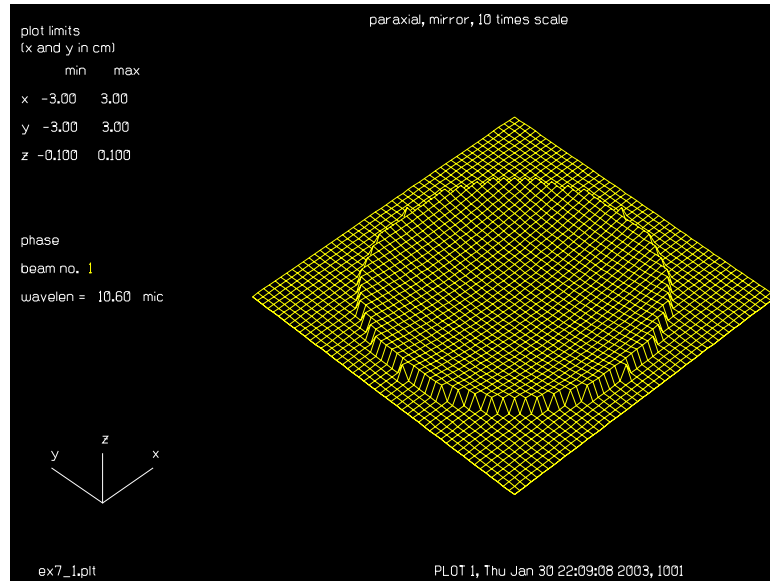


Fig. 7.1. Phase for Case 1, small defocus error due to difference of reference surface and true radius.

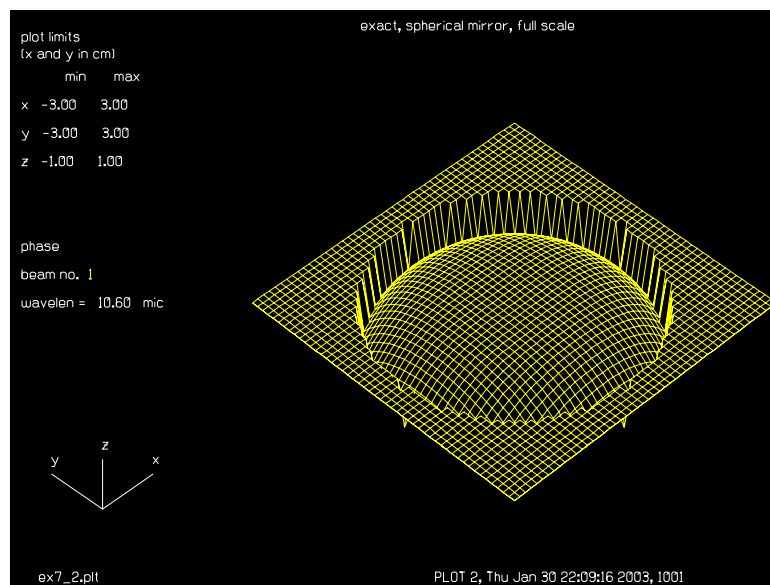


Fig. 7.2. Phase of exact spherical mirror showing spherical aberration.

```

c
c This calculation illustrates the use of MIRROR/GLOBAL with the
c ASTIGMATIC and EXACT parameters.
c
c Case 1: spherical mirror, ASTIGMATIC model
c
c This model implements global positioning with paraxial optics. This
c is an accurate representation if the aberration of the component
c is small.
c
c Case 2: Spherical mirror, EXACT model
c
c The EXACT parameter with CC=0 implements a spherical

```

Jump to: [Commands](#), [Theory](#)

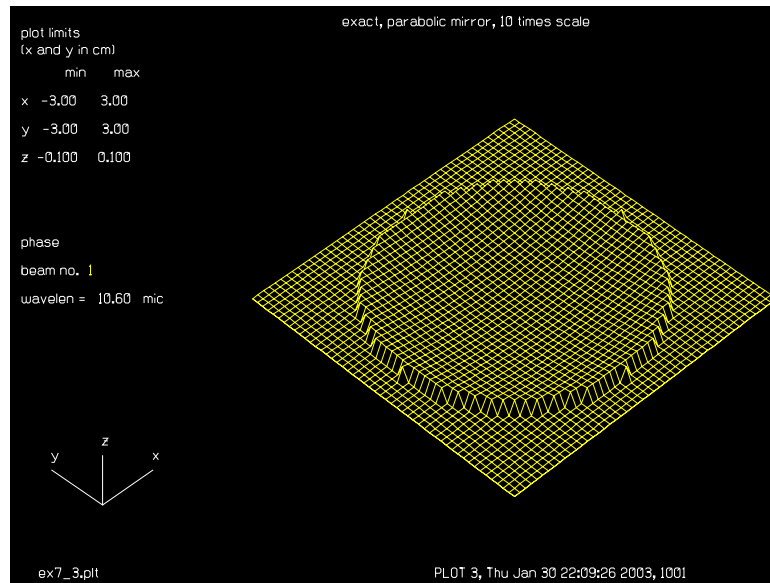


Fig. 7.3. Phase of exact parabolic mirror showing negligible aberration.

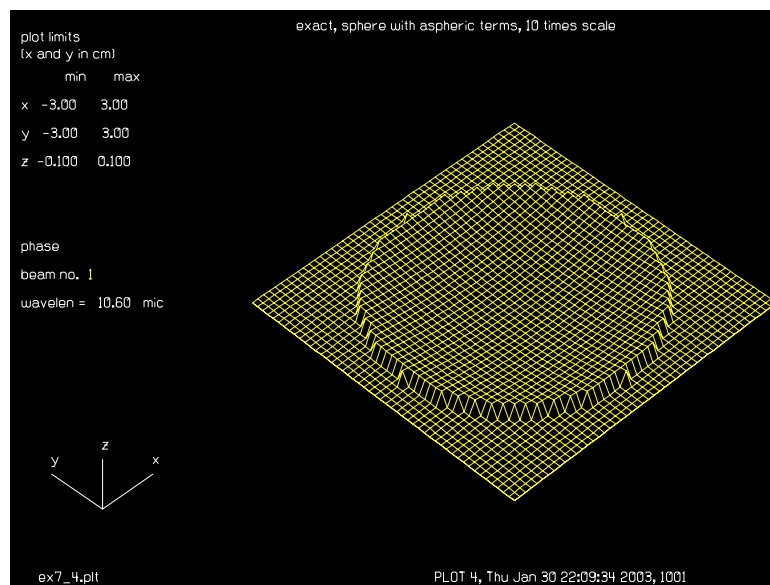


Fig. 7.4. Phase of exact spherical mirror with 4th and 6th order terms showing negligible aberration.

```

c mirror which has about .46 waves of spherical aberration.
c
c Case 3: Parabolic mirror, EXACT model
c
c The EXACT parameter with CC=-1. gives a parabola which is free of
c spherical aberration and, therefore, gives almost exactly the
c same characteristics as paraxial calculation using ASTIGMATIC.
c
c Case 4: Spherical mirror with aspheric terms, EXACT model
c
c Aspheric terms are added to a spherical mirror (CC=0.) to
c achieve the same surface profile as the parabolic mirror.
c

```

Jump to: [Commands](#), [Theory](#)

```

c One should generally use the simplest model that is appropriate
c for the problem under study.
c
  echo/on
  write/disk/on ex7.out/o
  set/strict/on
  variable/dec/real Apt F Z R Phase Waves Lambda SphericalAbr NetPhase
  variable/dec/real Omega0 M2
  nbeam 2
  array/set 0 64 64
  wavelength/set 0 Lambda*1e4           # set wavelength to 10.6 microns
  global/define 1 0. 0. 0.             # set Beam 1 to (0,0,0).
  Apt = 2.5
  F = 25
  R = 2.*F
  Lambda = 10.6e-4
  M2 = 3                               # default value for aperture
  units/s 1 .125
  clap/cir/con 1 Apt
  copy 1 2

c ----Case 1-----

  beams/off 2                          # limit calculation to Beam 1
c
c Position the vertex and apply conic astigmatic model
c
  vertex/locate/absolute x=0. y=0. z=0.
  vertex/rotate/set
  status
  mirror/global/conic rad=-R cc=0. astigmatic # ASTIGMATIC model, sphere
  status/gauss
  Z = F/(1.+((M2*Lambda*F)/(pi*Apt^2))^2) list
  set/strict/off
  Omega0 = Apt/sqrt(1.+((pi*Apt^2)/(M2*Lambda*F))^2) list
  intensity/xslice/phase 1 0. 0. 2.5 11
C Case 1
  Phase = (pi*Apt^2/Lambda)*(1./Z - 1./F) list
  set/window/abs -3 3 -3 3
  set/density 32 32
  title paraxial, mirror, 10 times scale
  plot/watch ex7_1.plt
  plot/i/phase first=1 last=1 max=.1 min=-.1
  pause
c
c Because of the choice of reference surface centered at the
c surrogate gaussian beam waist, there is a slight amount of
c focus phase shown in the display.
c
c ----Case 2-----
c
  copy 2 1
  global/define 1 0. 0. 0.
  vertex/locate/abs
  mirror/global/conic rad=-R cc=0. exact/rays # EXACT model, sphere
c
c Note the added spherical aberration
c
  intensity/xslice/phase 1 0. 0. 2.5 11

```

Jump to: [Commands](#), [Theory](#)

```

C Case 2
  SphericalAbr = -(2*pi/Lambda)*(2*(R-sqrt(R^2-Apt^2))-(Apt^2/R)) list
  NetPhase = Phase + SphericalAbr list
  title exact, spherical mirror, full scale
  plot/watch ex7_2.plt
  plot/i/phase first=1 last=1 max=1. min=-1.
  pause
c
c ----Case 3-----
c
  copy 2 1
  global/define 1 0. 0. 0.
  vertex/loc/abs
  mirror/global/conic rad=-R cc=-1. exact/rays # EXACT, parabola
c
c Note that there is almost no spherical aberration
c
  intensity/xslice/phase 1 0. 0. 2.5 11
C Case 3
  Phase=
  title exact, parabolic mirror, 10 times scale
  plot/watch ex7_3.plt
  plot/i/phase first=1 last=1 max=.1 min=-.1
  pause
c
c ----Case 4-----
c
  copy 2 1
  global/define 1 0. 0. 0.
  vertex/loc/abs
                                     # EXACT, sphere, aspheric coefficients
  mirror/global/conic rad=-R cc=0. a4=1e-6 a6=2.5e-10 exact/rays
c
c Note that there is almost no spherical aberration
c
  intensity/xslice/phase 1 0. 0. 2.5 11
  Phase=
C Case 4
  pause
  title exact, sphere with aspheric terms, 10 times scale
  plot/watch ex7_4.plt
  plot/i/phase first=1 last=1 max=.1 min=-.1

```

Ex8: Conic mirrors

Table. 8.1. Table of Ex8 examples

Ex8a: Confocal parabolas, alternate surface	1
Ex8b: Single off-axis parabola.	3
Ex8c: Elliptical mirror	4
Ex8d: Off-axis parabola with high NA effects	4
Ex8e: Eigen mode treatment for an array of elliptical mirrors	6

This example illustrates several examples of conic mirror system.

Ex8a: Confocal parabolas, alternate surface

In this example, two confocal parabolas are used to form an afocal telescope. Two configurations are considered. Both use the same parabolic primary. Both also use the same parabolic secondary, except that in the first case the secondary is struck on the concave side by the beam passing through the secondary surface and exiting in the reverse direction from the incident beam. In the second case, the alternate intersection point is selected and the beam strikes the on convex side and exits in the same direction as the incident beam. The first case has a much lower magnification value and has higher peak intensity than the second case.

If propagation in the forward direction results in two possible intersection points, the primary intersection point is the one closest to the vertex.

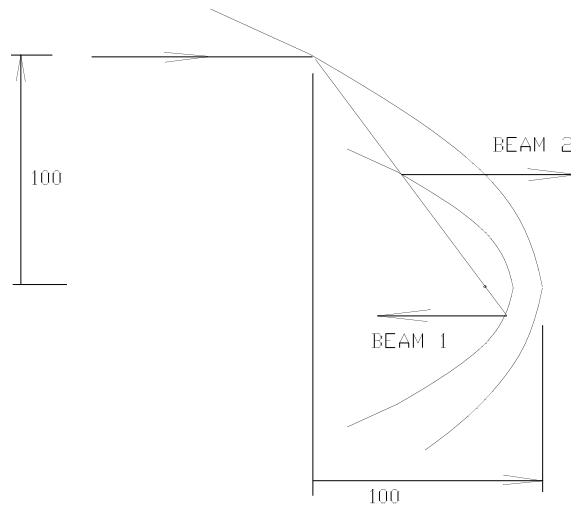


Fig. 8.1. In this application a set of confocal parabolas is decentered by 100 cm. The incident ray is reflected toward the focal point of the first parabola. Beam 1 takes the route which leads to the closest surface. Beam 2 is set to use the alternate surface intersection. Beam 1 is much smaller than Beam 2 leading to a higher irradiance.

Input: `ex8a.inp`

c## ex8a

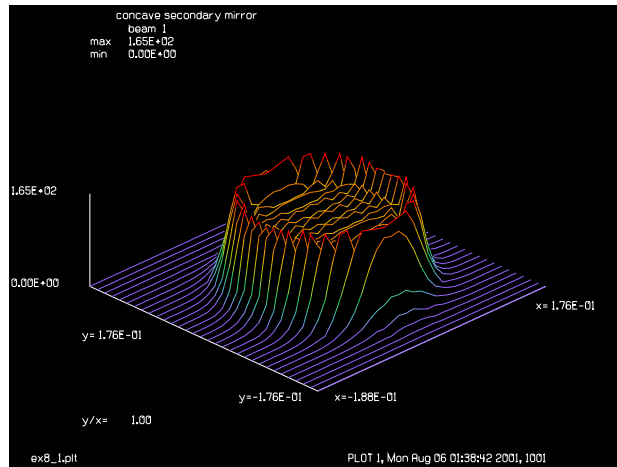


Fig. 8.2. Intensity after hitting concave secondary.

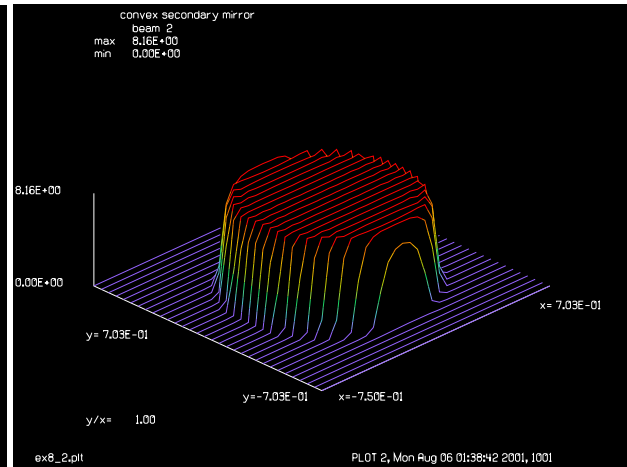


Fig. 8.3. Intensity after hitting convex secondary.

```

c
c Example 8a: Off-axis Confocal Parabolas, Alternate Surface
c
c Two confocal parabolic mirrors are used with the exiting beam
c parallel to the input beam. Input beams are off-axis with a large
c incidence angle. Beam 1 selects the first intercept on the second
c mirror. Beam 2 selects the alternate intercept and exits to the -z
c direction. The primary mirror has a vertex radius of 50. cm.
c A magnification of .5 is obtained with a secondary mirror of radius
c 25.0 displaced 12.5 cm. from the primary vertex position.
c
c array/set 1 32 32                # define Beam 1, 32 x 32
c nbeam 2                          # define Beam 2 of same size
c units/field 1 1.5
c wavelength/set 0 .5
c gaussian/cir/con 1 2. 1. 20
c global/define 0 0. 0. 0.
c
c Apply new exact ray trace model to Beam 1 then copy Beam 1 to Beam 2.
c
c beams/off 2
c vertex/locate/absolute x=0. y=-100. z=100.
c vertex/rotate/set
c mirror/global/conic rad=-50. cc=-1. ast
c copy 1 2
c status
c
c Apply second mirror halfway to focus.
c
c vertex/locate/absolute x=0. y=-100. z=87.5
c vertex/rotate/set
c mirror/global/conic rad=-25. cc=-1. ast
c
c Select Beam 2 and apply second mirror with alternate intercept
c
c beams/all/off

```



```

beams/on 2
vertex/locate/absolute x=0. y=-100. z=87.5
vertex/rotate/set
mirror/global/conic radius=-25 cc=-1. ast alternate
c
c Display results
c
status
status/gaus
global/l
intensity/xslice/phase 1
intensity/yslice/phase 1
title concave secondary mirror
plot/watch ex8_1.plt
plot/l 1
intensity/xslice/phase 2
intensity/yslice/phase 2
title convex secondary mirror
plot/watch ex8_2.plt
plot/l 2
end

```

Ex8b: Single off-axis parabola

Input: `ex8b.inp`

```

c## ex8b
c
c Example 8b: Off-axis Parabola
c
array/set 1 32 32 # define Beam 1, 32 x 32
nbeam 2 # define Beam 2 of same size
units/field 1 1.5
wavelength/set 0 .5
global/define 0 0. 0. 0.
clap/c/n 1 1
c
c Apply exact ray trace model to Beam 1 then copy Beam 1 to Beam 2.
c
beams/off 2
vertex/locate/absolute x=0. y=-100. z=100.
vertex/rotate/set
mirror/global/conic rad=-50. cc=-1. ast
variab/set Xrad 1 prad list # phase radius in x-direction
variab/set Yrad 1 yprad list # phase radius in y-direction
#
#
# Focus point lies at (-100,0,75)
# The hypotenuse of the 3-4-5 triangle is 125
# matching the phase radius

prop 125 # propagate to focus
title Airy pattern

```

Jump to: [Commands](#), [Theory](#)

plot/1 1

Ex8c: Elliptical mirror

Input: ex8c.inp

```
c## ex8c
c
c calculating reflection from an ellipsoidal mirror
c The beam hits the ellipsoid from inside at the end of semi-minor axis
c The tangential plane of incidence (X-Z) has elliptical cross-section (longitud
c                                     (A = major axis, B = minor axis)
c The array beam has minimum waist at the focal point of ellipsoid
c and it should focus to the second focal point
c
html/start          # start html browser display
html/svg/on         # turn on Scalable Vector Graphics (SVG)
html/location/current # set browser to make current line visible
array/set 1 256 256
units/field 1 1
wavelength/set 0 .5
gaussian/cir/con 1 1. 1.
global/define 0 0. 0. 0.
rad = 10 list          # radius of curvature of mirror at vertex
                        # f< radius <2f
cc = -0.36 list        # Conic Constant -1<cc<0 for ellipse
ecn = sqrt(-cc) list   # eccentricity
f = rad / (1+ecn) list  # focal distance
c1 = -f + (rad / (1+cc)) list # half the distance between two foci
A = f + c1 list        # semi-major axis of ellipse
B = A*sqrt(1+cc) list  # semi-minor axis of ellipse
Angle = 5              # angle in degrees
global/def 1 0 0 0 ry=Angle
deltaz = 2*c1 + f
vertex/locate/relative 1 0. 0. deltaz
vertex/rotate/set 0 0 0 vertex
mirror/global/conic rad cc exact/ray
end
focus/apply/abcd 1
zreff
```

Ex8d: Off-axis parabola with high NA effects

This is an example of an off-axis parabola with high NA features considered. A collimated beam is incident on an off-axis parabola. The mirror is located such that the chief ray intercept obscures at $z = 0$. After the mirror, an ideal lens was used to remove the curvature and the curvature was put back with highna. This yields E_x , E_y , and E_z fields. Highna is, strictly speaking, built for aplanatic objectives, i.e., an objective free of coma. The parabola, of course, has coma so there is some slight pupil distortion that would not exist in a truly aplanatic objective lens of the same numerical aperture. The error should be quite slight for this low NA and would primarily have a very slight change in image size with virtually no change in polarization cross coupling. The calculation is started with a collimated beam in horizontal linear

Jump to: [Commands](#), [Theory](#)

polarization, TE. At the image plane, a linear polarizer is used to block all the TE polarization to determine the degree of cross coupling of TE into TM. The cross coupled power in TM is on the order of $1\text{E-}7$. The TM is back-propagated back to the original pupil to display the pupil distribution. It has the expected azimuthal dependency. This system has a relatively low NA and the cross coupling would be expected to be low as it is.

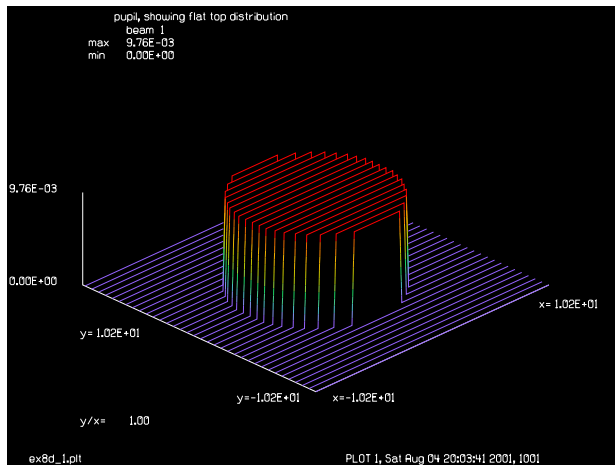


Fig. 8.4. Starting distribution for Ex8d.

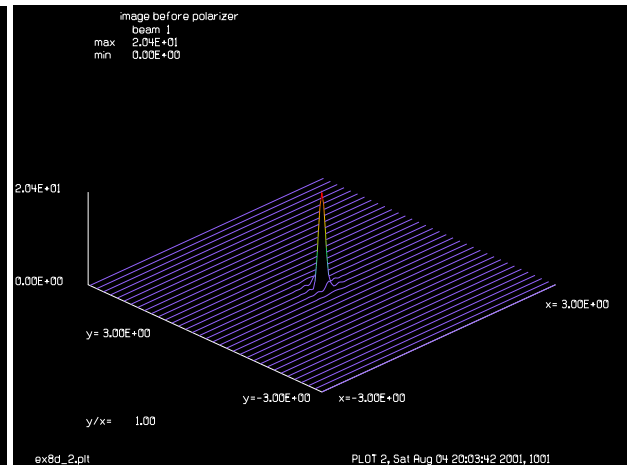


Fig. 8.5. Image created by off-axis parabola with high NA effects.

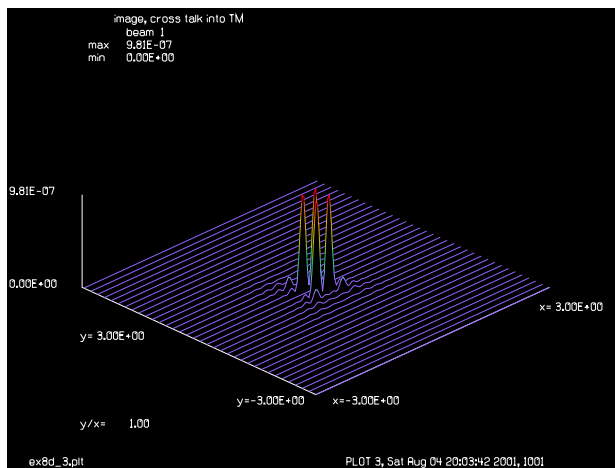


Fig. 8.6. Image after filtering out TE light.

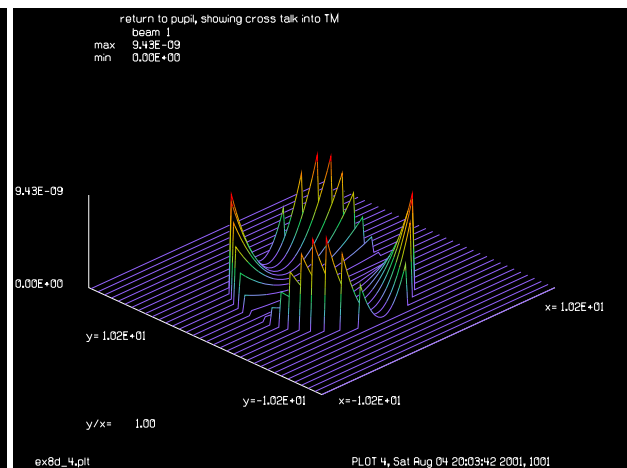


Fig. 8.7. Pupil recreated from TM by reverse high NA operation.

Input: `ex8d.inp`

```
c## ex8d
#
# Example 8d: Off-axis Parabola
#
# Beam is initially horizontally polarized
html/start          # start html browser display
html/svg/on         # turn on Scalable Vector Graphics (SVG)
html/location/current # set browser to make current line visible
```

Jump to: [Commands](#), [Theory](#)

```

variable/dec/int Line
Nline = 256
Lambda = 250e-4
array/set 1 Nline Nline 1          # define Beam 1
wavelength/set 1 Lambda*1e4
units/field 1 10*2.54
global/define 0 0. 0. 0.
clap/c/n 1 4.5/2*2.54
energy/norm 1 1
# locate mirror so ray intercept is at z=0
# this requires just a bit more than 19.5571
vertex/locate/absolute x=0. y=-74. z=19.5572
vertex/rotate/set
mirror/global/conic rad=-140. cc=-1. ast
global
title pupil, showing flat top distribution
plot/w ex8d_1.plt
plot/l 1 1 xrad=4*2.54
variab/set Xrad 1 prad list         # phase radius in x-direction
variab/set Yrad 1 yrad list         # phase radius in y-direction
vertex/locate/rel 1                 # locate vertex at current position
vertex/rotate/set 1 beam            # rotate vertex to align with beam
lens 1 Xrad                          # remove radius of curvature
                                     # with ideal lens. Radius will
                                     # be put back in with "highna"

nbeam 2 Nline Nline 0
wavelength/set 2 Lambda*1e4
highna/focus 1 2 -Xrad              # high NA puts
global 1                            # focus should be at (0,-74,30)
title image before polarizer
plot/w ex8d_2.plt
plot/l 1 1 xrad=3
jones/linpol 1 0                    # linear polarizer to block TE
energy 1                            # energy in TM polarization indicates cross talk
title image, cross talk into TM
plot/w ex8d_3.plt
plot/l 1 1 xrad=3
highna/collimate 1 2 Xrad
title return to pupil, showing cross talk into TM
plot/w ex8d_4.plt
plot/l 1 1 xrad=4*2.54

```

Ex8e: Eigen mode treatment for an array of elliptical mirrors

For an array of ellipses we get the best optical performance if we match the input beam to the eigen mode of the periodic structure defined by the array of ellipses. Matching the foci for neighboring ellipses is the proper structure for a geometric point source located at the foci; but for a gaussian wave solution, the foci need to be slightly offset. We choose the waist position so that when the gaussian wave strikes the mirror its radius of curvature is centered on one of the mirror foci as shown in Fig. 8.8. The separation between the gaussian waist and the foci may be small if the beam gaussian waist is small, as it will be at the output of an optical fiber, but it should be accounted for by proper alignment. If the angle of the chief ray is arbitrary then the offset is different for the left and right foci, as indicated in Fig. 8.8.

Jump to: [Commands](#), [Theory](#)

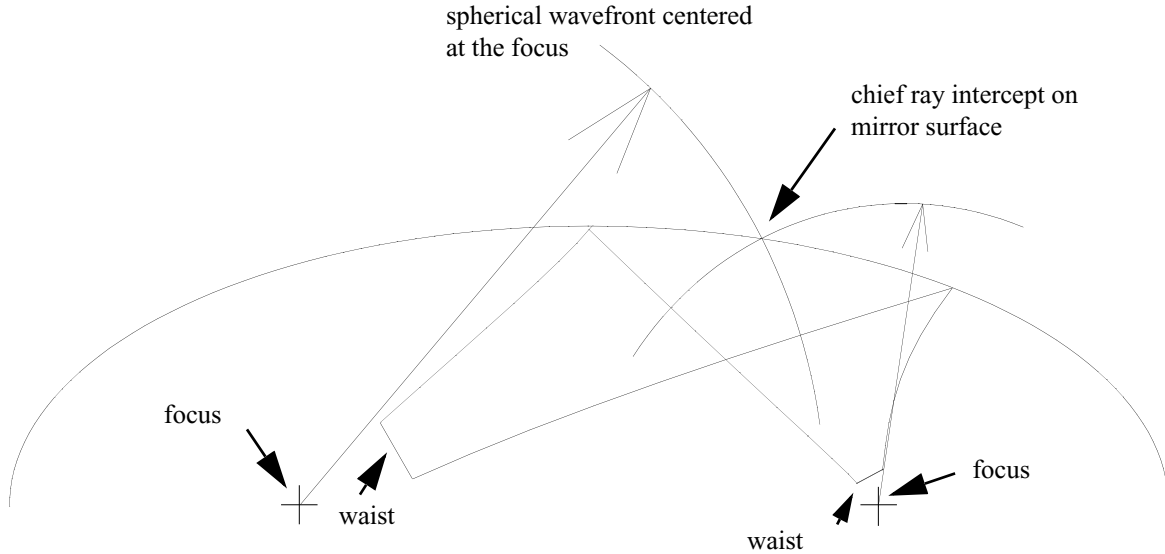


Fig. 8.8. Perfect imaging is obtained when the gaussian spherical wave incident on the mirror has a wavefront with center at one of the foci. The reflected gaussian spherical wave will then be centered at the second focus point. The waist radii are located on a chief ray through the foci but on the inside of the focus point.

Figure 8.9 shows the system in unfolded form (without the flat mirrors). The first mirror of Fig. 8.9 is the same as Fig. 8.8. To make the second mirror of Fig. 8.9, we simply take the system of Fig. 8.8, flip it end-for-end and top-to-bottom, and batch up the corresponding waist positions. We continue in this way for as many mirrors as we want in the array. The unfolded system is slightly skewed with respect to a line through the foci of each mirror for arbitrary chief ray angle. We can of course redefine our concept of the optical axis and consider the elliptical mirrors to be slightly rotated with respect to this redefined optical axis.

If you use this eigen function solution, then you will get nearly perfect beam transfer through the system, even for many elements. Let me know if this is what you want.

Figure 8.10 indicates the variables and points. The ellipse is defined by its radius at the vertex and eccentricity ϵ (or equivalently by the conic constant κ). The semi-major axis is a and the semi-minor axis is b . The variable c is defined to be half the separation of the foci. We have the following relationships

$$\epsilon = \frac{c}{a} \quad (8.1)$$

$$\kappa = -\epsilon^2 \quad (8.2)$$

$$a^2 = b^2 + c^2 \quad (8.3)$$

$$f = \frac{r_v}{1 + \epsilon}, \text{ focus-to-vertex distance (not focal length)} \quad (8.4)$$

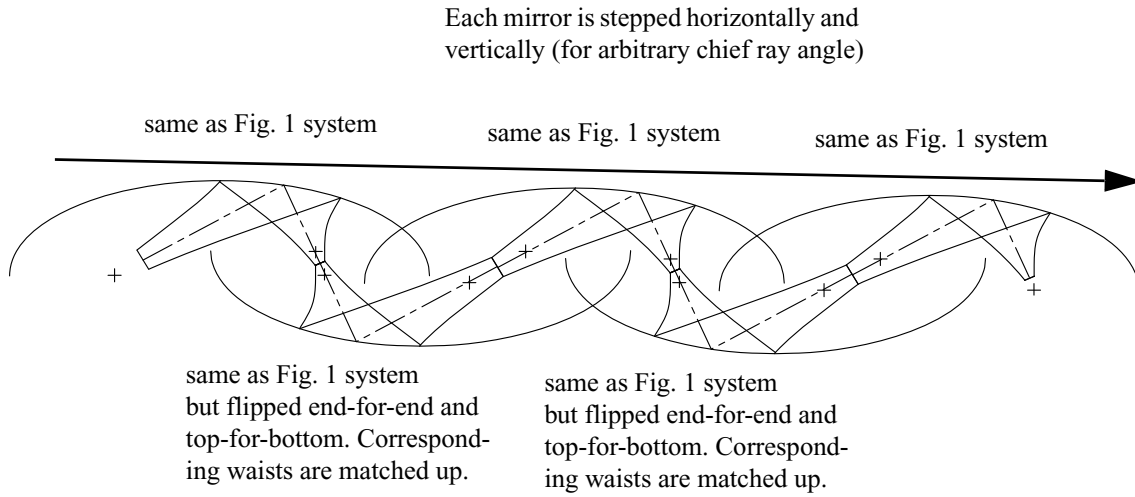


Fig. 8.9. The “unfolded” system (equivalent to system with flat mirrors). This is an array of mirror systems—each identical to the system of Fig. 8.8 with appropriate flips and shifts. For arbitrary chief ray angle the mirrors are stepped horizontally and vertically. This offset can be accommodated by slight translation of the flat mirrors in the folded system.

$$r_v = f(1 + \epsilon) = \frac{b^2}{a} \quad (8.5)$$

$$c = \frac{r_v}{1 - \epsilon^2} - f \quad (8.6)$$

$$a = f + c \quad (8.7)$$

$$b = a\sqrt{1 - \epsilon^2} \quad (8.8)$$

$$r_1 + r_2 = 2c + 2f \quad (8.9)$$

As the ray will go through the focal point, its angle may be defined by the height y from the ellipse axis to the ray intercept on the surface. We choose y to be a fraction of the semi-minor axis so that only physically meaningful values are selected. Given y , we find x :

$$x = a \sqrt{1 - \frac{y^2}{b^2}} \quad (8.10)$$

$$\theta = \tan^{-1}\left(\frac{y}{c + x}\right) \quad (8.11)$$

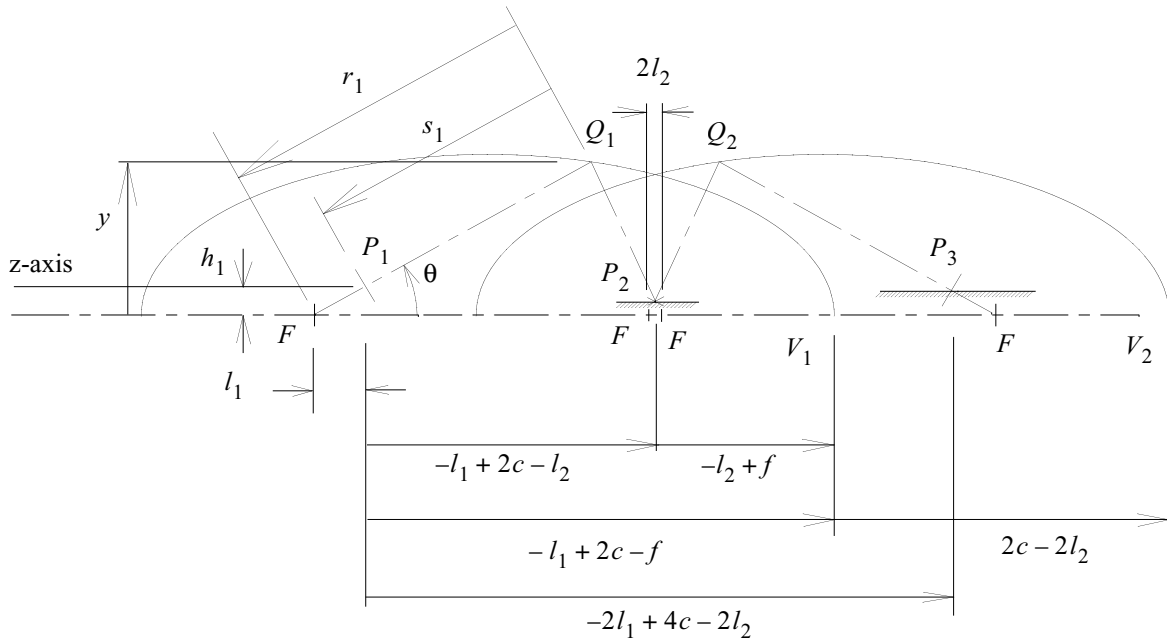


Fig. 8.10. An array of elliptical mirrors with fold mirrors. Only two mirrors are shown, but the sequence may be continued indefinitely, just as with Fig. 8.9. The initial waist is at point P_1 . The beam is directed at angle θ relative to the z -axis and proceeds through the first focal point. The ray intercepts the mirror at point Q_1 . If the gaussian beam meets the eigen condition, the center of curvature of the incident wavefront at point Q_1 will be r_1 . The distance from the intercept point at Q_1 to the waist at P_1 is s_1 . The separation of the waist from the focus is Δs_1 . The projections of Q_1 onto the z - and y -axis are l_1 and h_1 . All focal points F are on the same center line. The first vertex V_1 is advanced along the z -axis a distance $-l_1 + 2c - f$ from the initial waist, where $2c$ is the distance between the foci of one elliptical mirror. The second vertex V_2 is advanced along the z -axis the distance $2c - 2l_2$ relative to V_1 . The third vertex V_3 is advanced along the z -axis the distance $2c - 2l_2$ relative to V_2 . The flat mirrors are located at the waist points and are alternately dropped h_2 (at P_2) and h_1 at the next point P_3 .

We select the radius to match the focus-to-mirror distance.

$$r_1 = \sqrt{y^2 + (c^2 + x^2)} \quad (8.12)$$

For a beam of initial waist radius ω_0 and wavelength λ , the Rayleigh range is $z_R = \frac{\pi \omega_0^2}{\lambda}$.

The radius obeys the expression

$$r_1 = s_1 + \frac{z_R^2}{s_1} \quad (8.13)$$

The propagation distance s_1 may be solved:

$$s_1 = \frac{r_1 + \sqrt{r_1^2 - 4z_R^2}}{2} \quad (8.14)$$

The gaussian beam size at the mirror can be found by:

$$\omega_1 = \omega \sqrt{1 + \left(\frac{\lambda s_1}{\pi \omega_0^2} \right)^2} \quad (8.15)$$

$$\Delta_1 = r_1 - s_1 \quad (8.16)$$

$$h_1 = \Delta_1 \sin \theta \quad (8.17)$$

$$l_1 = \Delta_1 \cos \theta \quad (8.18)$$

From the properties of the ellipse, we have

$$r_2 = 2(c + f) - r_1 \quad (8.19)$$

The propagation distance s_2 may be found

$$s_2 = \frac{r_2}{1 + \left(\frac{\lambda r^2}{\pi \omega_0^2} \right)} \quad (8.20)$$

$$\Delta_2 = r_2 - s_2 \quad (8.21)$$

$$h_2 = \Delta_2 \sin \theta \quad (8.22)$$

$$l_2 = \Delta_2 \cos \theta \quad (8.23)$$

If the initial waist is at $(0, 0, 0)$, then the vertex of the first elliptical mirror is at $(0, -h_1, -l_1 + 2c + f)$. The first flat mirror is at $(0, h_2 - h_1, -l_1 + 2c - l_2)$. The vertex of the second elliptical mirror is at $(0, -h_1, -l_1 + 4c + f - 2l_2)$. The second flat mirror is located at $(0, 0, -2l_1 + 4c - 2l_2)$.

The second elliptical mirror is advanced $-2l_2 + 2c$ relative to the first elliptical vertex and the third elliptical mirrors is advanced $-2l_1 + 2c$ relative to the second elliptical vertex. This pattern repeats for additional mirrors. Each additional flat mirror is advanced $-l_1 + 2c - l_2$, but the heights have alternating values of $h_2 - h_1$ and 0.

Input: ex8e.inp

```
c## ex8e
c
c Example: Eigenmode treatment for an array of elliptical mirrors
c
```

Jump to: [Commands](#), [Theory](#)


```

c Calculating reflection from an array of ellipsoidal mirrors.
c we find the eigen function of the for an array of elliptical
c mirrors for an arbitrary input angle "y" defined by the height
c of the intercept of the chief ray relative to the ellipse
c major axis.
c
c The eigen mode is found by solving for the propagation distance
c "s1" such that when the gaussian beam intercepts the mirror
c the radius of curvature is equal to the focus-to-mirror distance
c This eigen mode gaussian will be reflected by the mirror with
c no aberrations. A second distance "s2" is calculated to locate
c the waist in the reflected optical space. A flat mirror is placed
c at this new waist position. Additional elliptical mirrors can
c be added with every other mirror interchanging the roles of
c "s1" and "s2".
c
c Since no aberration is included the "exact" parameter on the
c mirror/global command can be deleted for cleaner and faster
c calculations, as the detailed ray tracing is not needed.
c
w0 = 0.005 # initial waist of gaussian
r_vertex = .5 # radius of mirror at vertex
Lambda = 1.55e-4
eccentricity = .8 # eccentricity of ellipse
cc = -1*eccentricity^2 # conic constant
f = r_vertex/(1+eccentricity)
c_ellipse = r_vertex/(1-eccentricity^2) - f # half the separation of foci
a_ellipse = f + c_ellipse # semi-major axis
b_ellipse = a_ellipse*sqrt(1 - eccentricity^2) # semi-minor axis
y = .9*b_ellipse # height of ray intercept relative to vertex
x = a_ellipse*sqrt(1.-y^2/b_ellipse^2) # horizontal displacement of intercept re
                                     # to the the center of the ellipse

theta1 = atan(y/(x + c_ellipse)) # angle of ray (radians)
theta_degrees1 = theta1*180/pi # angle of ray (degrees)
r1 = sqrt(y^2 + (c_ellipse + x)^2) list # desired radius of beam at mirror inter
z_rayleigh1 = pi*w0^2/Lambda # Rayleigh range for start
s1 = (r1 + sqrt(r1^2 - 4*z_rayleigh1^2))/2 list # distance to propagate to yield
Check = s1 + z_rayleigh1^2/s1 - r1 list # check that s1 is calculated properly
w1 = w0*sqrt(1 + (Lambda*s1/(pi*w0^2))^2) # gaussian transverse radius at mirror
delta1 = r1 - s1 # separation between of waist from focus
h1 = delta1*sin(theta1) # y-projection of separation
l1 = delta1*cos(theta1) # z-projection of separation
theta2 = -atan(y/(x - c_ellipse)) # angle of exiting beam (radians)
r2 = 2*(c_ellipse+f) - r1 # radius after reflection
s2 = r2/(1 + (Lambda*r2/(pi*w1^2))^2) list # distance to next waist
delta2 = r2 - s2 # separation of new waist from focus
h2 = delta2*sin(theta2) # y-projection of separation
l2 = delta2*cos(theta2) # z-projection of separation
array/set 1 256 256 # set number of pixels
wavelength/set 0 Lambda*1e4 # set wavelength
gaussian/cir/res 1 1. w0 # define starting gaussian
global/define 1 rx=-theta1*180/pi # define starting ray angle
# Ellipse 1

```

Jump to: [Commands](#), [Theory](#)

```

Y_ellipse = -h1
Z_ellipse = -l1 + 2*c_ellipse + f
vertex/locate/abs 0. Y_ellipse Z_ellipse
vertex/rotate/set 0 0 0
mirror/global/conic -r_vertex cc=cc order=4 exact # ellipse 1
# Flat 1
Y_flat = h2 - h1
Z_flat = -l1 + 2*c_ellipse - l2
vertex/locate/abs 0 Y_flat Z_flat
vertex/rotate/set 90 0 0
mirror/global/flat # flat 1
#Ellipse 2
y_ellipse = -h1
Z_ellipse = Z_ellipse + -2*l2 + 2*c_ellipse
vertex/locate/abs 0. Y_ellipse Z_ellipse
vertex/rotate/set 0 0 0
mirror/global/conic -r_vertex cc=cc order=4 exact # ellipse 2
# Flat 2
Y_flat = 0
Z_flat = Z_flat -l1 + 2*c_ellipse - l2
vertex/locate/abs 0 Y_flat Z_flat
vertex/rotate/set 90 0 0
mirror/global/flat # flat 2
# Ellipse 3
y_ellipse = -h1
Z_ellipse = Z_ellipse + -2*l1 + 2*c_ellipse
vertex/locate/abs 0. Y_ellipse Z_ellipse
vertex/rotate/set 0 0 0
mirror/global/conic -r_vertex cc=cc order=4 exact # ellipse 3
# Flat 3
Y_flat = h2 - h1
Z_flat = Z_flat -l1 + 2*c_ellipse - l2
vertex/locate/abs 0 Y_flat Z_flat
vertex/rotate/set 90 0 0
mirror/global/flat # flat 3
# check that Strehl ration is still good
strehl # check Strehl ratio
plot/x 1 le=-.01 ri=.01
# propagate same distance s2 again and compare with
# radx and rady from geodata with r2 calculated
# analytically
prop s2
geodata
r2=

```

Ex9: Beam steering with flat mirrors in three-dimensions

This example illustrates azimuthal rotation of a beam by three flat mirrors in the form of a “k”.

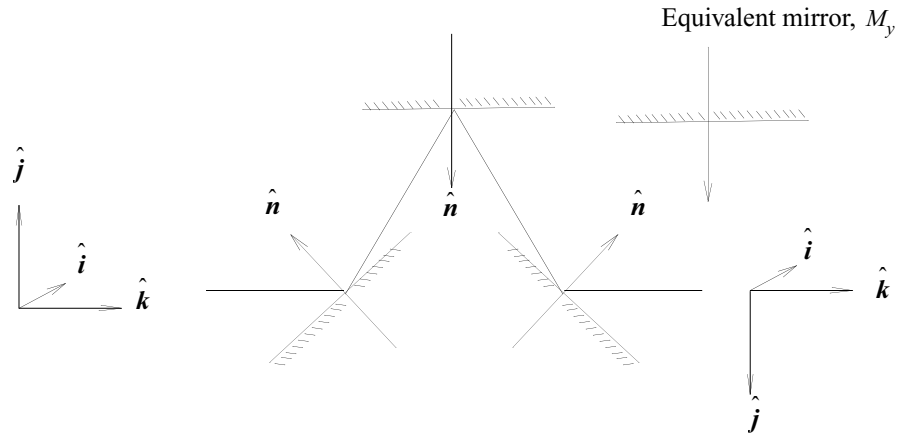


Fig. 9.1. K-mirror system. It is equivalent to the elementary y-mirror system M_y . This system is rotated 22.5 degrees in Ex 9.

Input: ex9.inp

```
c## ex9
c
c Example 9: Beam Steering with Flat Mirrors in Three-Dimensions
c
c Tilted flat surfaces are used to direct the beam off-axis. A series
c of mirrors are used to return the beam in line with the original
c direction with an azimuthal rotation of the field.
c
c Initialize beam field and position.
c
units/field 1 1.5
wavelength/set 0 .5
gaussian/cir/con 1 2. 1. 30
c
c The GLOBAL commands allow three-dimensional component positioning.
c
global/define 0 0. 0. 0. # initialize global coordinates.

C C position vector(0,0,0)
C C k-vector (0,0,1
C C j-vector (0,1,0)
C
vertex/locate/absolute x=0. y=0. z=0.
vertex/rotate/set rx=45+22.5.
c Vertex is tilted to direct beam upward and forward
mirror/global # Mirror 1
global/list

C C position vector (0,0,0)
C C k-vector (0,0.707107,0.707104)
C
```

```
vertex/locate/absolute x=0. y=10. z=10.
vertex/rotate/set rx=-90.
c Vertex is tilted face down
mirror/global # Mirror 2
global/list

C C position vector (0,10,10)
C C k-vector (0,-0.707107,0.707104)
C
vertex/locate/absolute x=0 y=0 z=20.
vertex/rotate/set rx=-45-22.5
c Vertex is tilted to direct the beam to +Z direction
mirror/global # Mirror 3
global/list

C C position vector (0,0,20)
C C k-vector (0,0,1)
C C j-vector (0,-1,0)
```

Ex10: Macros, variables and the udata command

GLAD has powerful features to facilitate construction of complex data decks. Command sequences which are to be used repeatedly may be stored in macros. GLAD has variables which may be used to access internal information and which may be used to perform mathematical operations. Summary of data accumulated during long calculations may be displayed at the end of the calculation to convey a clearer picture of the evolution of the data. I

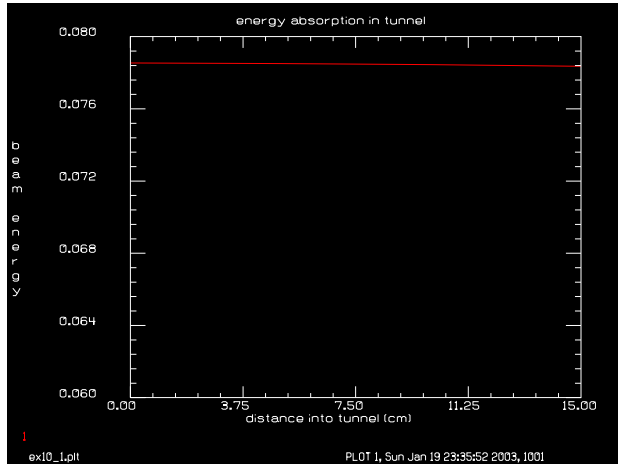


Fig. 10.1. Energy absorption in tunnel.

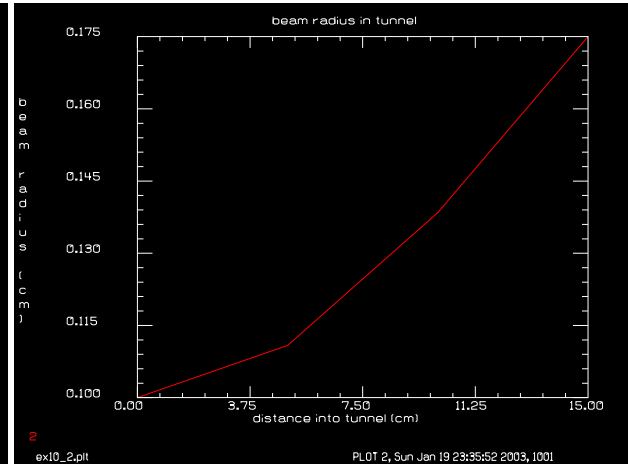


Fig. 10.2. Beam radius versus distance.

Input: ex10.inp

```
c## ex10
c
c Example 10: Macros, Variables, and User Data.
c
c This example demonstrates the capabilities in GLAD to use macros that
c control the execution of GLAD. Macros are also the basis for the
c iterative analysis needed for analysis and design of resonators.
c
c Variables may be defined with mnemonic names, making the command files
c much easier to read and allowing command values to be calculated.
c
c Variable names may be used for any numerical entry in a command.
c
c A macro is a file containing a sequence of command lines.
c
c The user data is a set of arrays from which the user may define and
c generate plots.
c
c In this example a simple macro is generated that propagates the beam
c a short distance then applies an aperture. Data will be saved
c including the beam energy and beam size. The macro is called several
c times generating a table of data for a beam propagating through a
c tunnel.
c
c Initialize the beam
c
c wavelength/set 1 10
c gaussian/cir/res 1 5 .1
```

```

c
  variable/declare/integer udata_nstore
c
  udata_nstore = 0          # initialize user data storage value
  distance = 5.             # propagation length
  total_length = 0.         # initialize variable
c
c  To obtain the current list of parameters available:
c
c#  variable/set
c
c  Define the macro 'tunnel'. The overwrite modifier will overwrite
c  a local file if it exists.
c
macro/def tunnel/overwrite
  prop distance              # propagation step
  clap/cir 1 .2             # clear aperture
  total_length = total_length+distance] # increment total length
  variab/set bm_energy kbeam=1 energy # energy of beam 1
  variab/set bm_size kbeam=1 bmsize # current beam size
  udata_nstore = udata_nstore+1
c store set of user data
  udata/set udata_nstore total_length bm_energy bm_size
macro/end
c
c  Set first point of user data
c
  variab/set bm_energy kbeam=1 energy # energy of beam 1
  variab/set bm_size kbeam=1 bmsize # current beam size
  udata_nstore = udata_nstore+1
c store set of user data
  udata/set udata_nstore total_length bm_energy bm_size
c
c  Turn off writing to screen and execute macro 3 times
c
  write/screen/off
  macro/run tunnel/3
  write/screen/on
c
c  Define labels and generate plots.
c
  udata/xlabel distance into tunnel (cm)
  udata/ylabel beam energy
  title energy absorption in tunnel
  plot/watch ex10_1.plt
  plot/udata 1 min=.06 max=.08
  title beam radius in tunnel
  udata/ylabel beam radius (cm)
  plot/watch ex10_2.plt
  plot/udata 2
end

```

Ex11: Confocal unstable resonator

Table. 11.1. Table of Ex11 examples

Ex1a: Unstable, bare cavity resonator	1
Ex1b: Unstable, bare cavity resonator with apodization	5
Ex1c: Unstable resonator with diverging output	8
Ex1d: Bare Cavity Resonator, seeded with reverse mode	10
Ex1e: Finding first six modes of an unstable resonator.	13
Ex1f: Gain sheets in an unstable resonator	15
Ex1g: 1-D Unstable Resonator	17

Example 11a models a confocal unstable resonator with circular mirrors. The configuration is the same as one described in by Siegman and Miller [1]. The resonator collimated and equivalent Fresnel numbers are

$$N_c = \frac{Ma^2}{L\lambda}, N_{eq} = \frac{M^2 - 1}{2M} \frac{a^2}{L\lambda} \quad (11.1)$$

where a is the aperture radius, L is the resonator length, λ is the wavelength, and M is the magnification. The parameters that are used are: $L = 90$ cm, $a = 0.3$ cm, $M = 2$, $\lambda = 10$ μ . This results in $N_c = 2$ and $N_{eq} = 0.75$.

After one round trip the units of the distribution are twice those of the starting distribution. To start the next round trip the distribution must be rescaled to the original units. According to Siegman and Miller, the loss per cycle should be 44%.

The GLAD calculations are in close agreement.

Reference

1. A. E. Siegman and H. Y. Miller, "Unstable Optical Resonator Loss Calculations Using Prony Method," Appl. Opt. Vol. 9, No. 12, p. 2729 (1970).

Ex11a: Unstable, bare cavity resonator

Input: ex11a.inp

```

c## ex11a
c
c Example 11a: Unstable, Bare Cavity Resonator
c
c Resonator parameters
c -----
c equivalent Fresnel number      0.5
c magnification                  2.0
c length                        90.0 cm
c aperture 1                    0.3 cm
c aperture 2                    0.6 cm

```

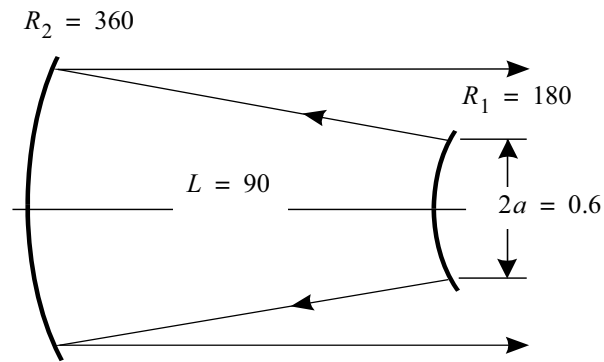


Fig. 11.1. Unstable confocal resonator configuration with tilt.

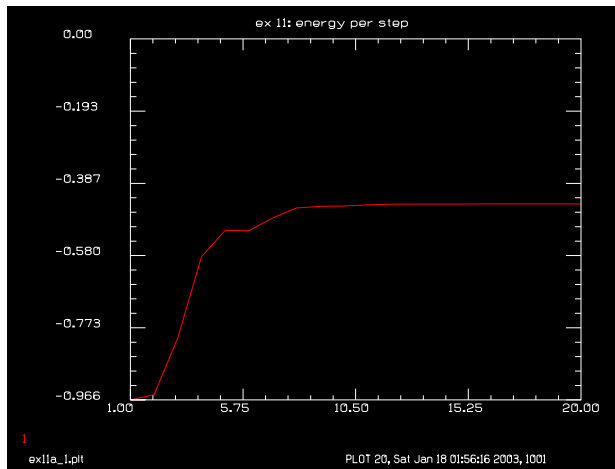


Fig. 11.2. Plot of energy loss per pass as a function of the pass number.

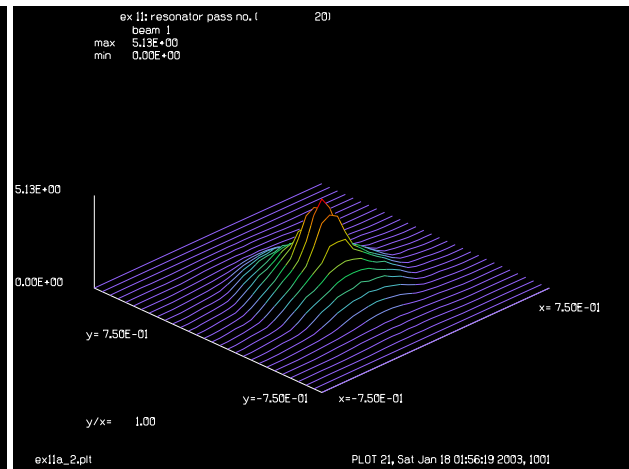


Fig. 11.3. Converged transverse mode before scraper mirror.

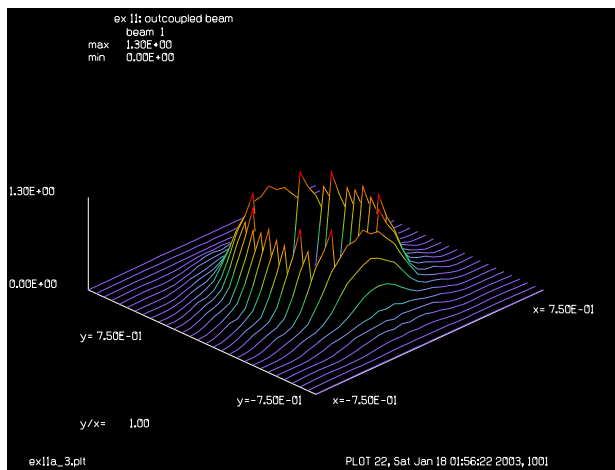


Fig. 11.4. Converged transverse mode after scraper mirror.

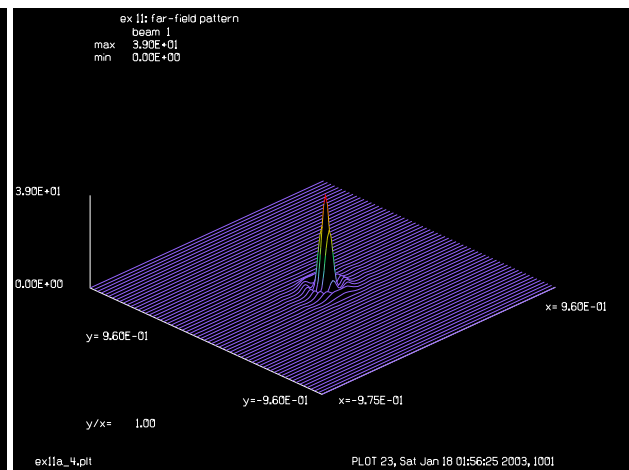


Fig. 11.5. Far-field distribution of aligned resonator.

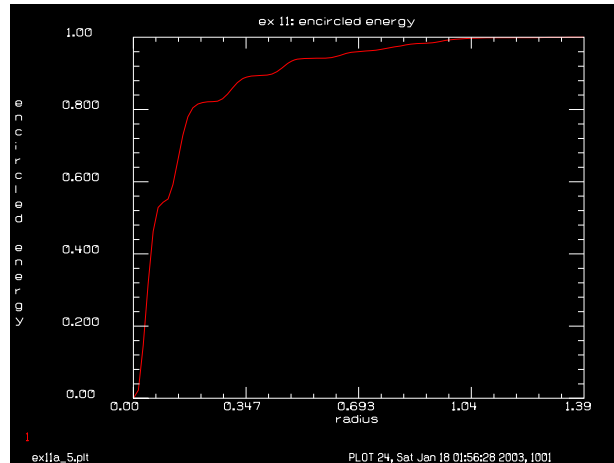


Fig. 11.6. Plot of encircled energy using the PIB command.

```

c
c  ENERGY/GNORM normalizes the beam energy to the assigned value after
c  each pass.
c
c  WRITE/OFF and WRITE/SCREEN are used to reduce terminal printout.
c
c  The variable pass_number is used to count the macro pass numbers and
c  is input to the title. The variable STOP is used to test for
c  convergence and pass the value to the IF statement for exiting the
c  macro. The variable field_radius is used to initialize the array
c  and rescale the array at the end of the macro.
c
c  Variables
c  -----
c  STOP                switch to determine convergence
c  pass_number         counts macro passes (integer)
c  field_radius        half-width of array for rescale
c
variable/dec/int pass_number          # declare pass_number as integer
variable/dec/int STOP TEST            # stop switch
c
c  Define a macro which is a series of commands which represents one loop
c  through the bare resonator cavity
c
macro/def conres/over
    pass_number = pass_number + 1 list      # increment pass counter
c#    write/screen/off                    # dispose unwanted output
    clap/cir/no 1 .3                       # aperture of .3 cm
    mirror rad=180                          # convex mirror
    prop 90                                # propagate 90 cm backward
    mirror rad=360.                         # concave mirror
    clap/cir/no 1 .7                       # aperture of .7 cm
    prop 90                                # forward propagation
    variable/set Energy 1 energy
    write/screen/on
    udata/set pass_number pass_number Energy-1

```

Jump to: [Commands](#), [Theory](#)

```

gain/converge/test ibeams=1 nstore=STOP # store convergence test in STOP
gain/eigenvalue/show 1                  # display eigenvalues
energy/norm 1 1
if STOP macro/exit                      # conditional exit
if [!TEST] then
    title resonator mode pass = @pass_number
    plot/l xrad=.75
endif
macro/end
c
c Inialize variables
c
    pass_number = 0                      # Macro pass number
    field_radius = 1.6                  # Rescale field radius
c
c Establish initial units and a gaussian field distribution
c
    array/set 1 128                    # set array size
    units/field 1 field_radius          # define units
    wavelength/set 1 10.                # define wavelength
    gain/converge/set eps1=.005 eps2=.001 npoints=3
    gain/eigenvalue/set 1
c
c eps1= fractional difference between min and max energy in last
c       npoints passes for convergence.
c eps2= fraction field change for convergence.
c
c Call the macro requesting up to 30 passes with conditional exit on
c convergence.
c
    plot/screen/pause 3
    TEST = 1
    resonator/name conres
    resonator/eigen/test 1
    TEST = 0
    pass_number = 0
    clear 1 0
    noise 1 1
    resonator/run 30
    title ex 11: energy per step
    plot/watch ex11a_1.plt
    plot/udata max=0
c
c plot converged field distribution.
c
    title ex 11: resonator pass no. @pass_number
    plot/watch ex11a_2.plt
    plot/liso 1 xrad=.75 ns=64
c
c Use obscuration of .3 cm. radius and display outcoupled beam.
c
    obs 1 .3
    title ex 11: outcoupled beam
    plot/watch ex11a_3.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/liso 1 xrad=.75 ns=64
c
c Apply a lens and propagate to the far-field.
c
c Plot far-field pattern
c
lens/sph 1 100
prop 100
title ex 11: far-field pattern
plot/watch ex11a_4.plt
plot/liso 1 ns=64
c
c Generate a power-in-the-bucket table.
c
encircled/calculate/energy 1
encircled/udata 1
title ex 11: encircled energy
plot/watch ex11a_5.plt
plot/udata 1 min=0. max=1.
end

```

Ex11b: Unstable, bare cavity resonator with apodization

Example Ex 11b illustrates a general unstable resonator with optional aperture apodization using the split command. It is a half-symmetric resonator with the following prescription: $L = 7$ cm, $a = 0.24$ cm, $M = 1.687$, $\lambda = 0.58 \mu$.

We can not find analytical eigenmodes of unstable resonators, but we can find eigenradii which define a paraxial beam which would have the same radius after one round trip. See Section 8.1.3 of the Theory Manual. Eq. 8.34 gives an expression for the eigenradii. In this case, the eigenradii are 34.37 (expanding wave) and -20.37 cm (shrinking wave). The resonator will operate in the expanding wave. The resonator/eigen/test command determines the eigenradii and resonator/eigen/set sets up a beam up with the proper radius of curvature. The clear and noise commands setup the beam to start from noise.

Input: ex11b.inp

```

c## ex11b
c
c Example 11b: Unstable, Bare Cavity Resonator with Apodization
c
c ENERGY/NORM normalizes the beam energy to the assigned value after
c each pass.
c
c WRITE/OFF is used to reduce terminal
c
c The variable pass_number is used to count the macro pass numbers and
c is input to the title. The variable STOP is used to test for
c convergence and pass the value to the IF statement for exiting the
c macro. The variable field_radius is used to initialize the array
c and rescale the array at the end of the macro.

```

Jump to: [Commands](#), [Theory](#)

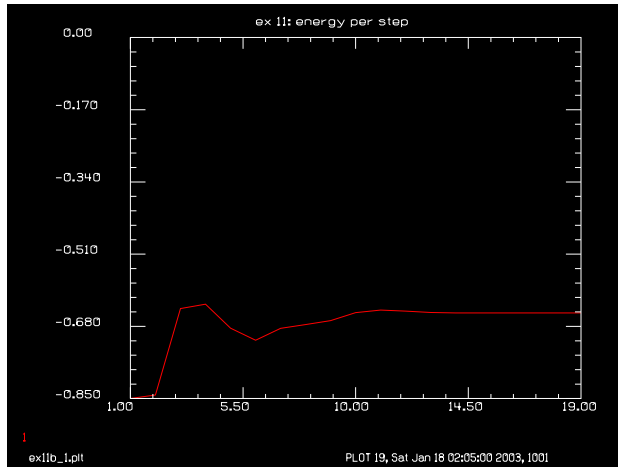


Fig. 11.7. Apodized scraper. Plot of energy loss per pass as a function of the pass number.

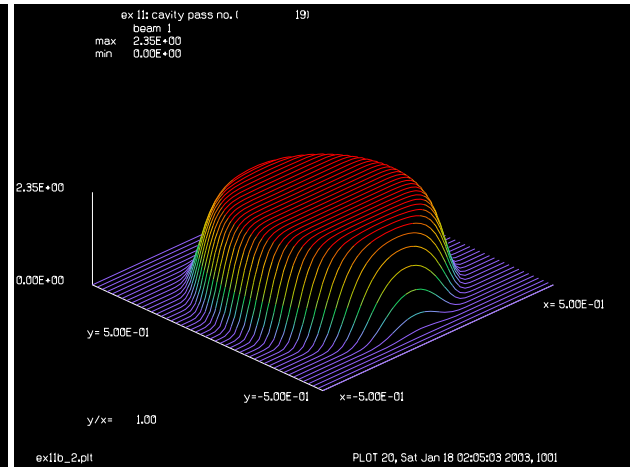


Fig. 11.8. Apodized scraper. Converged transverse mode before scraper mirror.

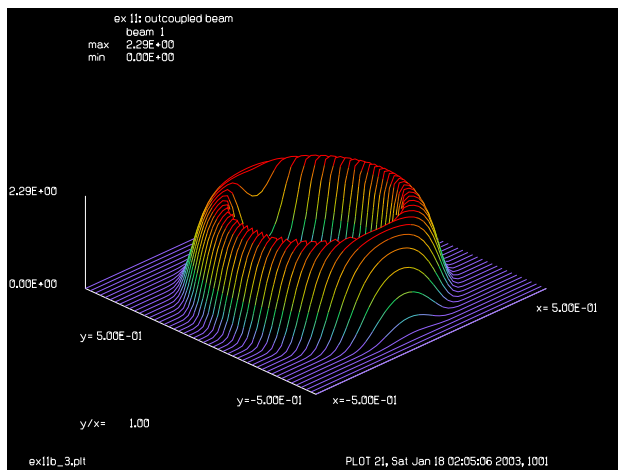


Fig. 11.9. Apodized scraper. Converged transverse mode after scraper mirror.

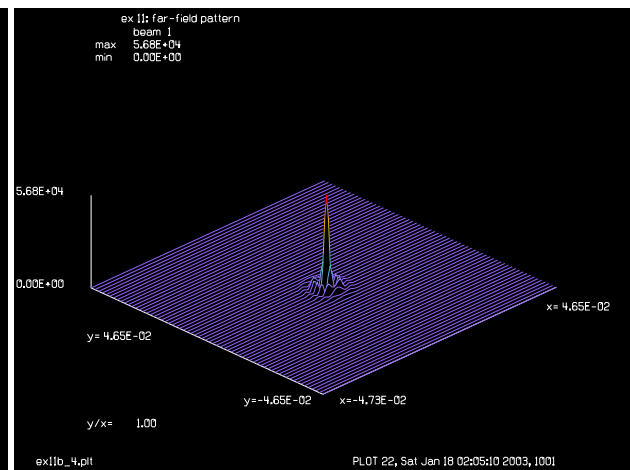


Fig. 11.10. Apodized scraper. Far-field distribution of aligned resonator.

```

c
c Variables
c -----
c STOP          switch to determine convergence
c pass_number   counts macro passes (integer)
c field_radius  half-width of array for rescale
c
variable/dec/int pass_number          # declare pass_number as integer
variable/dec/int apodize STOP TEST
c
c Define a macro which is a series of commands which represents one loop
c through the bare resonator cavity
c
macro/def res/over
    pass_number = pass_number + 1      # increment pass counter
    write/off                                         # dispose unwanted output
    if apodize then

```

Jump to: [Commands](#), [Theory](#)

```

        split/c/in 1 .24 6                # aperture of .24 cm
    else
        clap/c/no 1 .24
    endif
    mirror 1 rad=100                      # concave mirror
    prop 7                                # propagate 90 cm backward
    mirror/flat 1                          # flat mirror
    prop 7
    variable/set Energy 1 energy
    write/screen
    udata/set pass_number pass_number Energy-1
    gain/converge/test ibeams=1 nstore=STOP # store convergence test in STOP
    gain/eigenvalue/show 1                 # display eigenvalues
    energy/norm 1 1
    if [!TEST] then
        title cavity mode pass_number = @pass_number
        plot/1 1 xrad=.5 ns=64
    endif
    if STOP macro/exit                      # conditional exit
macro/end
c
c  Inialize variables
c
    apodize = 1
    pass_number = 0                        # Macro pass number
    field_radius = .8                      # Rescale field radius
c
c  Establish initial units and a gaussian field distribution
c
    array/s 1 128                          # set array size
    units/field 1 field_radius              # define units
    wavelength/set 1 .58                   # define wavelength
    plot/screen/pause 3
    TEST = 1
    resonator/name res                     # resonator macro name
    resonator/eigen/test 1                 # determine eigenradii
    TEST = 0
    clear 1 0                              # clear array
    noise 1 1                              # start from noise
    gain/converge/set eps1=.005 eps2=.001 npoints=3
    gain/eigenvalue/set 1
c
c  eps1= fractional difference between min and max energy in last
c         npoints passes for convergence.
c  eps2= fraction field change for convergence.
c
c  Call the macro requesting up to 30 passes with conditional exit on
c  convergence.
c
    resonator/run 30                       # run a maximum of 30 times
    title ex 11: energy per step
    plot/watch ex11b_1.plt
    plot/udata max=0
c

```

Jump to: [Commands](#), [Theory](#)

```

c plot converged field distribution.
c
c title ex 11: cavity pass no. @pass_number
c plot/watch ex11b_2.plt
c plot/liso 1 xrad=.5 ns=64
c if apodize then
c     split/c/out 1 .24 6          # aperture of .24 cm
c else
c     obs/c 1 .24
c endif
c
c Display outcoupled beam.
c
c title ex 11: outcoupled beam
c plot/watch ex11b_3.plt
c plot/liso 1 xrad=.5 ns=64
c focus/apply/abcd 1              # propagate to waist, Beam 2
c
c Plot far-field pattern
c
c title ex 11: far-field pattern
c plot/watch ex11b_4.plt
c plot/liso 1 ns=64
c
c Generate a power-in-the-bucket table.
c
c encircled/calculate/energy 1
c encircled/udata
c title ex 11: power-in-the-bucket
c plot/watch ex11b_5.plt
c plot/udata 1 min=0. max=1.
c end

```

Ex11c: Unstable resonator with diverging output

This is an example of an unstable resonator with diverging output.

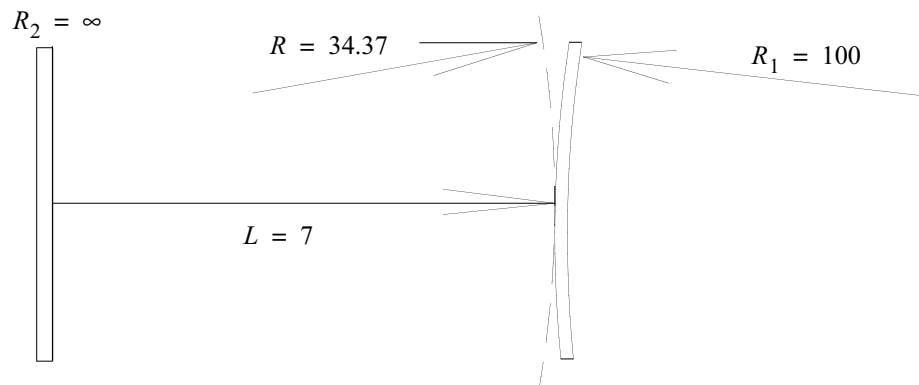


Fig. 11.11. Unstable resonator with diverging output.

Jump to: [Commands](#), [Theory](#)

Input: ex11c.inp

```

c## ex11c
#
# Example 11c: Unstable resonator with diverging output
#
# This example illustrates an unstable resonator consisting
# of a convex mirror and a flat mirror. The resonator command
# is used to find the eigenradius  $r=34.3678$  cm. The distribution
# is initially set to noise to simulate a start from simulated
# emission.
c

variable/dec/int pass_number MaxTimes TEST Nline
Nline = 128
array/s 1 Nline
nbeam 2 data                # specify attribute "data" so beam 2
                             # does not propagate. Saves time
nbeam 3 Nline 16 data        # array to show mode cross sections
clear 3 0
x=srand(3)                   # set random number generator seed
macro/def conres/over
  if [!TEST] pass_number = pass_number + 1
  write/disk output.dat/o
  clap/c/c 1 .86
  mirror 1 rad=100
  mult/beam 1 2
  prop 7
  mirror/flat 1
  prop 7
  variable/set Energy 1 energy
  write/screen
  udata/set pass_number pass_number Energy-1
  energy/norm 1 1
  if [!TEST] then
    plot/w plot1.plt
    plot/l xrad=.6 ns=128
    if [pass_number==5] then
      title 'Mode at 5th pass, start from noise'
      plot/w plot3.plt
      plot/l xrad=.6 ns=128
      title ' '
    endif
  endif
# copy center row of beam 1 to a series of rows in beam 3
  if [pass_number==1] then
    variab/set Units1 1 units
    units/set 3 Units1 1 # X-units match Beam1. Y-units=1
  endif
  copy/row 1 3 65 pass_number
  plot/w plot2.plt
  plot/l 3 xrad=.6 yrad=8
endif
  if [pass_number!=MaxTimes] rescale 1 field_radius
macro/end
pass_number = 0
field_radius = .865
wavelength/set 1 .58
wavelength/set 2 .58
units/field 1 field_radius

```

Jump to: [Commands](#), [Theory](#)

```

units/field 2 field_radius
units/field 3 field_radius 8
TEST=1                      # set test mode on
gaus/c/c 1 1 .3             # guess at starting distribution
resonator/name conres
resonator/eigen/test 1
# use "eigen/set" to set to the eigenradius to 34.367 cm
# for unstable resonators only the eigenradius is set.
# The beam size is left unchanged
resonator/eigen/set 1
TEST=0                      # done with test mode
# fill beam 2 with noise and copy into beam 1
clear 2 0
noise 2 1
copy/con 2 1
# make a supergaussian mask
gaus/c/c 2 .32 .24 4        # make the gaussian after units are
                             # defined

MaxTimes = 16
reson/run MaxTimes

```

Ex11d: Bare Cavity Resonator, seeded with reverse mode

This example illustrates injection of a converging mode in the confocal resonator. All unstable resonators have two possible modes. The expanding mode which is usually selected and the converging mode. A converging mode is created by creating a beam with focus aberration to give it a wavefront radius of 180 cm. The beam begins converging but as it collapses into the axis diffraction ultimately causes the beam to begin expanding again.

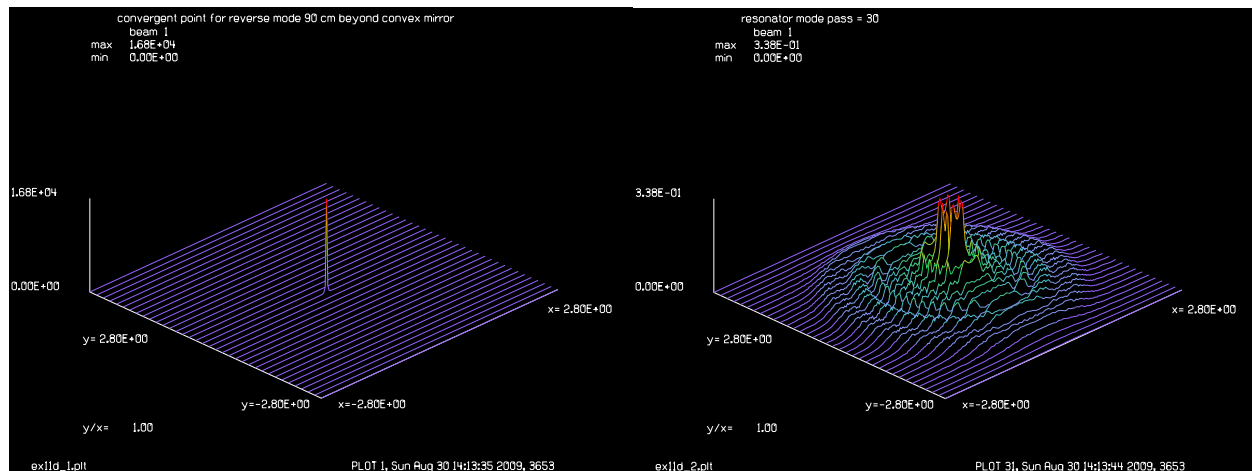


Fig. 11.12. Convergent point for reverse mode 90 cm beyond convex mirror.

Fig. 11.13. Resonator transverse mode at completion.

Input: ex11d.inp

```

c## ex11d!420779647016833
c
c Example 11d: Unstable, Bare Cavity Resonator, seeded with reverse mode

```

Jump to: [Commands](#), [Theory](#)

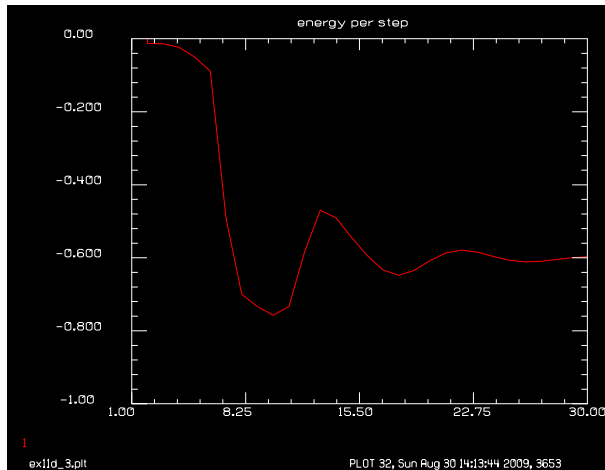


Fig. 11.14. Energy per step.

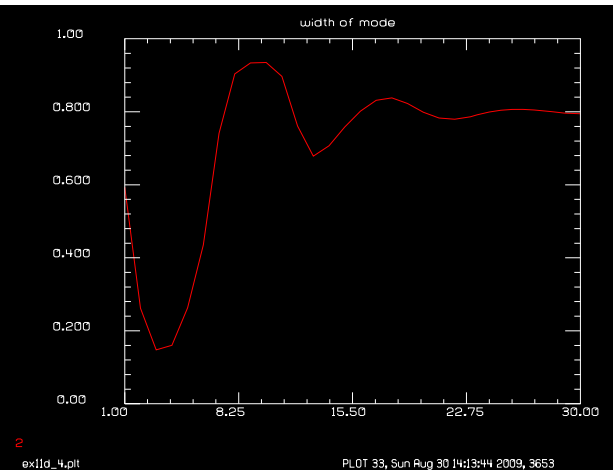


Fig. 11.15. Width of mode per step.

```

c
variable/dec/int pass_number          # declare pass_number as integer
variable/dec/int TEST                 # switch in after setup
c
c Unstable resonators support both expanding or collapsing modes
c We can seed the resonator with a collapsing mode by adding the
c appropriate convergence as a wavefront error. This mode will
c collapse into the center and then seed the expanding mode.
c plot4.plt shows the beam size just before the scrapper mirror
c as a function of pass number. Initially the mode decreases in size
c and then expands and stabilizes to the steady-state condition.
c
alias Name ex11d
set/alias_stop/off
macro/def conres/over
    if [!TEST] then
        fitgeo 1
        variab/set Xwidth 1 fitxrad
    endif
    clap/cir/no 1 1.2
    mirror 1 rad=180                      # convex mirror
    prop 90                             # propagate 90 cm backward
    mirror 1 rad=360.                    # concave mirror
    clap/cir/no 1 2.8
    prop 90                             # forward propagation
    variable/set Energy 1 energy
    energy/norm 1 1
    if [!TEST] then
        title resonator mode pass = @pass_number
        plot/w @Name_2.plt
        plot/l xrad=2.8
        udata/set pass_number pass_number Energy-1 Xwidth
    endif
    pass_number = pass_number + 1        # increment pass counter
macro/end
c

```

Jump to: [Commands](#), [Theory](#)

```

c  Inialize variables
c
pass_number = 0                # Macro pass number
field_radius = 2               # Rescale field radius
c
c  Establish initial units and a gaussian field distribution
c
array/set 1 256                # set array size
nbeam 2 256 16 data
units/field 1 field_radius     # define units
units/field 2 field_radius 8
clear 2 0
Lambda = 10.6e-4
wavelength/set 1 Lambda*10000  # define wavelength
c
c  eps1= fractional difference between min and max energy in last
c      npoints passes for convergence.
c  eps2= fraction field change for convergence.
c
c  Call the macro requesting up to 30 passes with conditional exit on
c  convergence.
c
TEST = 1
resonator/name conres
resonator/eigen/test 1
TEST = 0
pass_number = 1

c Simulate injection of collimated beam injected backward into
c the resonator so it strikes the concave mirror first.
c Add the convergence as a wavefront error keeping the geometry
c confiured for the forward mode
c
prop -90                      # back up to concave mirror
clear 1 1
apt_rad = 2.8
clap/c/c 1 apt_rad
c
c simulate convergence due to concave mirror
c
Wave = apt_rad^2/2/180/Lambda  # wavefront error to create
                                # phase radius of 180 cm
abr/foc 1 Wave rnorm=apt_rad   # implement wavefront error
prop 180                      # propagate to virtual focus
title convergent point for reverse mode 90 cm beyond convex mirror
plot/w @Name_1.plt
plot/l 1 xrad=2.8
prop -90                      # propagate backward to scrapper
c
c  Start resonator here
c
resonator/run 30
title energy per step
plot/watch @Name_3.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/udata 1 max=0 min=-1
title width of mode
plot/watch @Name_4.plt
plot/udata 2 min=0 max=1

```

Ex11e: Finding first six modes of an unstable resonator.

Find the first six modes of an unstable resonator. After each pass, the next mode is found by orthogonalizing with all earlier modes.

Input: ex11e.inp

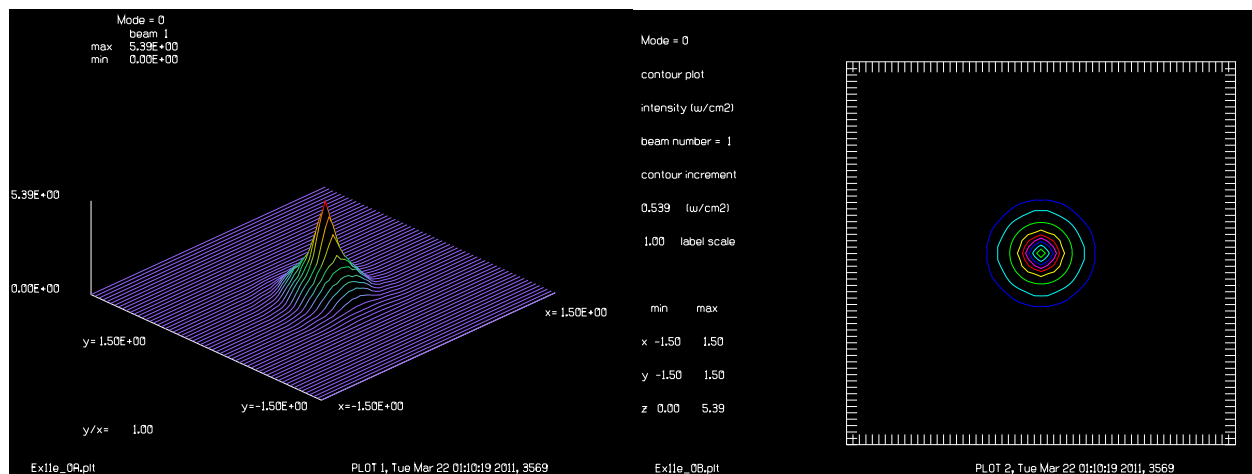


Fig. 11.16. Shape for mode 0.

Fig. 11.17. Shape for mode 0.

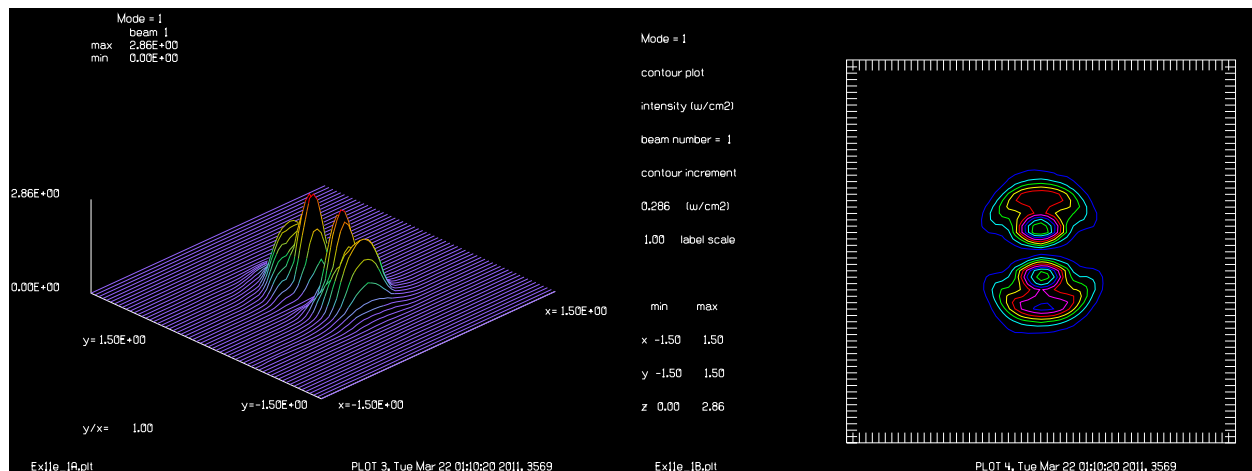


Fig. 11.18. Shape for mode 1.

Fig. 11.19. Shape for mode 1.

Jump to: [Commands](#), [Theory](#)

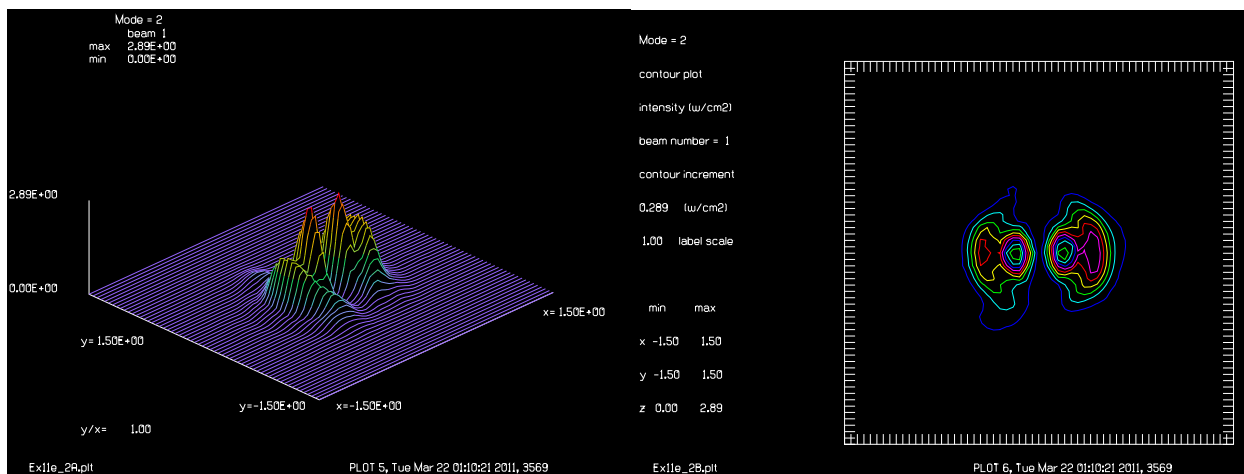


Fig. 11.20. Shape for mode 2.

Fig. 11.21. Shape for mode 2.

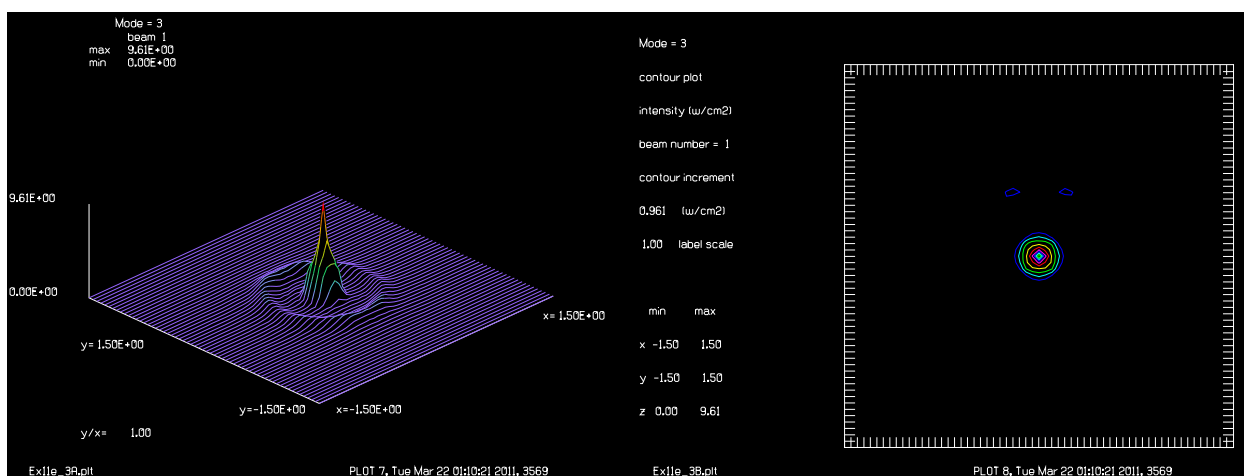


Fig. 11.22. Shape for mode 3.

Fig. 11.23. Shape for mode 3.

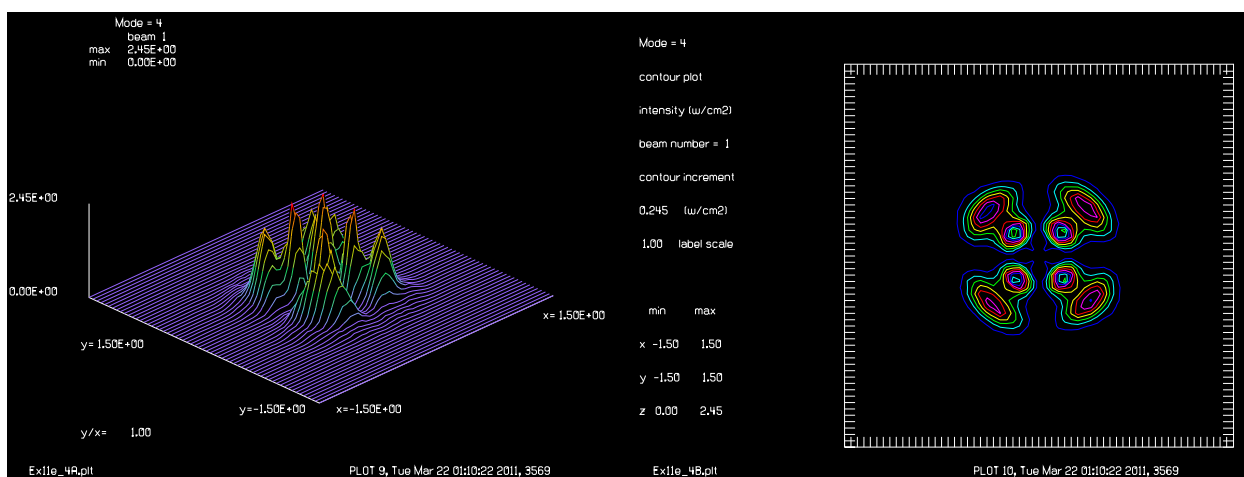


Fig. 11.24. Shape for mode 4.

Fig. 11.25. Shape for mode 4.

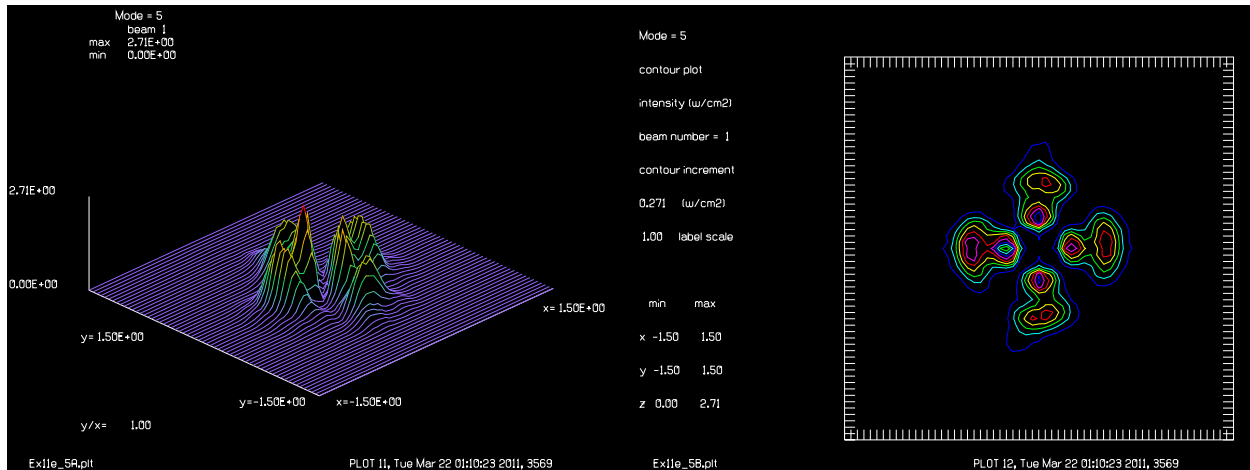


Fig. 11.26. Shape for mode 5.

Fig. 11.27. Shape for mode 5.

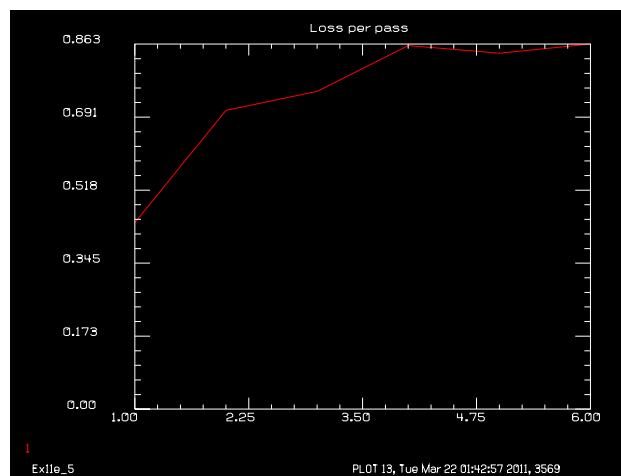


Fig. 11.28. Loss per pass for the first six modes.

Ex11f: Gain sheets in an unstable resonator

Examples 11f illustrates the best procedure to include gain sheets in an unstable resonator. Since the forward and backward paths usually have different sizes, we must either rescale the gain sheet to match the optical array or rescale the optical array to match the size of the gain sheet. Interpolation inevitably results in loss of accuracy with the rescale that most occur with each pass. It is usually best to leave the gain array constant and interpolate the optical array as the small errors due to interpolation are scattered out of the optical array at a rate determined by the photon lifetime. In the initial, backward pass of Ex11f, there is a rescale of 4/3 for the optical array to match its units to the gain array and immediately after a the gain sheet a rescale of 3/4 for the optical array to restore it to its original units. The forward pass requires no rescale of the optical array as it was set up to match the gain in the forward pass.

Input: `ex11f.inp`

```
c## ex11f
c
```

Jump to: [Commands](#), [Theory](#)

```

c  Example 11f: Unstable, Bare Cavity Resonator
c
c  Resonator parameters
c  -----
c  equivalent Fresnel number      0.5
c  magnification                  2.0
c  length                        90.0 cm
c  aperture 1                    0.3 cm
c  aperture 2                    0.6 cm
c
c  ENERGY/NORM normalizes the beam energy to the assigned value after
c  each pass.
c
c  WRITE/OFF and WRITE/SCREEN are used to reduce terminal printout.
c
c  The variable pass_number is used to count the macro pass numbers and
c  is input to the title.
c
c  Variables
c  -----
c  pass_number      counts macro passes (integer)
c  field_radius     half-width of array for rescale
c
alias Name Ex11f
variable/dec/int pass_number      # declare pass_number as integer
variable/dec/int TEST             # switch for setup
c
c  Define a macro which is a series of commands which represents one loop
c  through the bare resonator cavity
c
macro/def conres
  pass_number = pass_number + 1 list      # increment pass counter
  mirror 1 rad=180                        # convex mirror
  prop 45 1                               # propagate 90 cm backward
  if TEST=0 then
    energy/list 1
  endif
  rescale/scale 1 4/3 # set units to that of the forward beam for Mag=2
  pack/in
  gain/sheet 4          # 4 cm of gain sheet
  copy 1 2              # copy intensity for forward pass
  pack/out
  rescale/scale 1 3/4 # restore to units
  prop 45 1
  mirror 1 rad=360.      # concave mirror
  clap/cir/no 1 .7      # aperture of .7 cm
  prop 45 1              # forward propagation
  pack/in
  gain/sheet 4          # 4cm gain sheet
  copy 1 2              # copy intensity for backward pass
  pack/out
  prop 45 1
  variable/set Energy 1 energy
  write/screen/on

```

Jump to: [Commands](#), [Theory](#)

```

udata/set pass_number pass_number Energy
c Form output beam
copy 1 3
obs 3 .3
clap/cir/no 1 .3                # aperture of .3 cm
if [!TEST] then
    title resonator mode pass = @pass_number
    plot/1 xrad=.75
endif
macro/end
c
c Inialize variables
c
pass_number = 0                  # Macro pass number
field_radius = 1.6              # Rescale field radius
c
c Establish initial units and a gaussian field distribution
c
array/set 1 128                 # set array size
nbeam 3 data
beams/off 2
units/field 0 field_radius      # define units
wavelength/set 0 10.           # define wavelength
beer/set .25 100
pack/set 1 2
TEST = 1
resonator/name conres
resonator/eigen/test 1
resonator/eigen/list
geodata 1
geodata 2
TEST = 0
pass_number = 0
clear 2 0                      # set up array 2 for first pass
energy/norm 1 1
plot/w @Name_1.plt
resonator/run 10
udata
plot/w @Name_2.plt
plot/udata 1 min=0

```

Ex11g: 1-D Unstable Resonator

Examples 11g illustrates a 1-D confocal, negative branch, unstable resonator.

Input: ex11g.inp

```

c## ex11g
c
c Example 11g: Unstable, Bare Cavity Resonator
c
c Confocal, negative branch unstable 1D resonator

```

Jump to: [Commands](#), [Theory](#)

```

c
c Resonator parameters
c -----
c equivalent Fresnel number      0.5
c magnification                  2.0
c length                        90.0 cm
c aperture 1                    0.3 cm
c aperture 2                    0.6 cm
c
c Variables
c -----
c STOP                          switch to determine convergence
c pass_number                   counts macro passes (integer)
c field_radius                  half-width of array for rescale
c
alias Name ex11g
variable/dec/int pass_number      # declare pass_number as integer
variable/dec/int TEST             # stop switch
variable/dec/int Nline
Nline = 512
c
c Define a macro which is a series of commands which represents one loop
c through the bare resonator cavity
c
Apt1 = 2.7
Apt2 = Apt1/2
macro/def conres/over
    pass_number = pass_number + 1 list      # increment pass counter
    clap/cir/no 1 Apt2                     # aperture of M2
convex mirror
    mirror rad=-180                        # convex mirror
    prop 180+90                            # propagate 90 cm backward
    mirror rad=360.                        # concave mirror
    clap/cir/no 1 Apt1                     # aperture of .7 cm
    prop 180+90                            # forward propagation
    variable/set Energy 1 energy
    write/screen/on
    udata/set pass_number pass_number Energy-1
    energy/norm 1 1
    if [!TEST] then
        plot/w @Name_1.plt
        title resonator mode pass = @pass_number
        plot/x/i left=-Apt1*1.07 right=Apt1*1.07
    endif
macro/end
c
c Initialize variables
c
pass_number = 0                          # Macro pass number
field_radius = 3.2                       # Rescale field radius
c
c Establish initial units and a gaussian field distribution
c
c Make 1D array

```

Jump to: [Commands](#), [Theory](#)


```

c
array/set 1 Nline 1           # set array size
units/field 1 field_radius    # define units
wavelength/set 1 10.          # define wavelength
TEST = 1
resonator/name conres
resonator/eigen/test 1
TEST = 0
pass_number = 0
clear 1 0
noise 1 1
write/off
resonator/run 100
write/on
title ex 11: energy per step
plot/watch @Name_2.plt
plot/udata max=0
c
c  plot converged field distribution.
c
title ex 11: resonator pass no. @pass_number
plot/watch @Name_3.plt
plot/x/i 1 left=-Apt1*1.07 right=Apt1*1.08
c
c  Use obscuration of .3 cm. radius and display outcoupled beam.
c
obs 1 .3
title ex 11: outcoupled beam
plot/watch @Name_4.plt
plot/x/i 1 left=-Apt1*1.07 right=Apt1*1.07
title ex 11: Loss per pass
plot/watch @Name_5.plt
plot/udata 1.

```


Ex12: Misaligned confocal unstable resonator

This example is the same as Example 11, except that the resonator is misaligned by 0.1 wavelength. The resonator takes significantly longer to converge and the loss increases to about 55%.

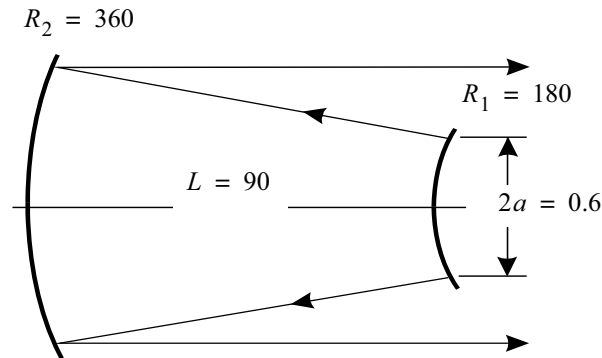


Fig. 12.1. Unstable confocal resonator configuration with tilt.

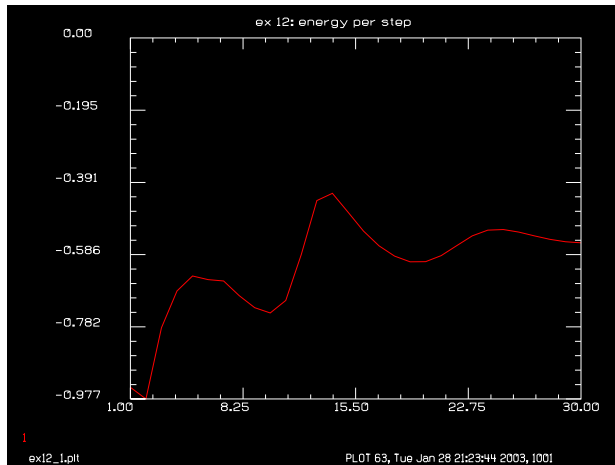


Fig. 12.2. Plot of energy loss per pass as a function of the pass number.

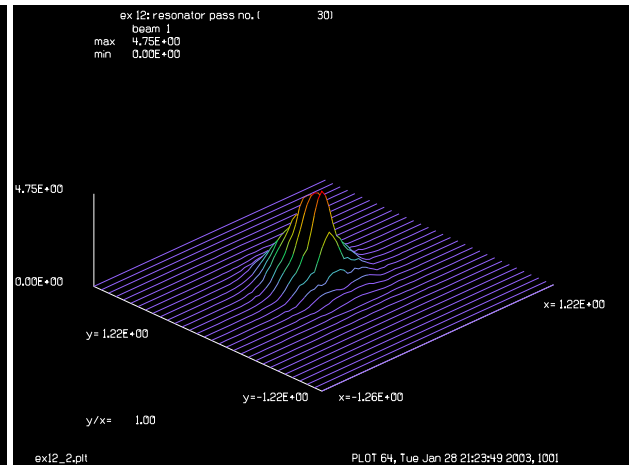


Fig. 12.3. Converged transverse mode before scraper mirror.

Input: ex12.inp

```
c## ex12
c
c Example 12: Misaligned, Unstable, Bare Cavity Resonator
c
c Resonator parameters
c -----
c equivalent Fresnel number      0.5
c magnification                  2.0
c length                        90.0 cm
c aperture 1                    0.3 cm
```

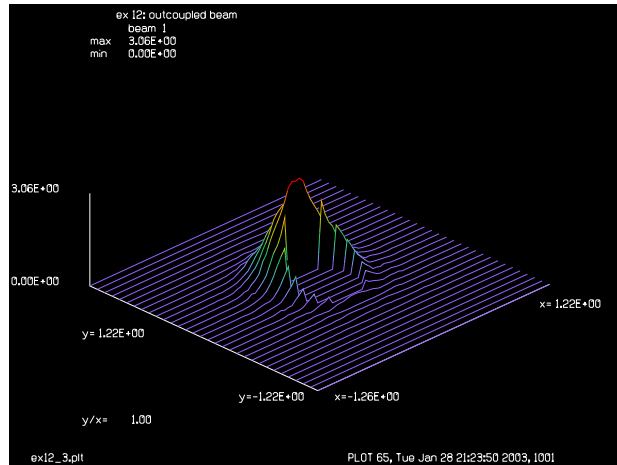


Fig. 12.4. Converged transverse mode after scraper mirror.

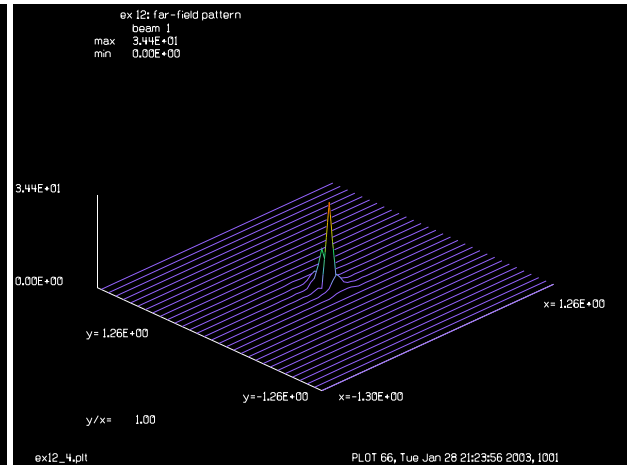


Fig. 12.5. Far-field distribution of aligned resonator.

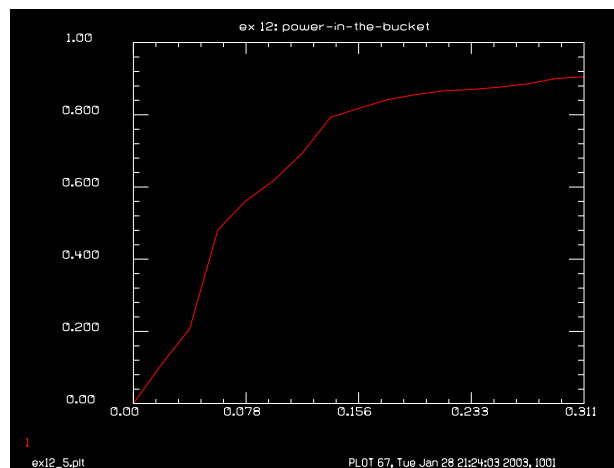


Fig. 12.6. Plot of encircled energy using the PIB command.

```

c aperture 2                0.6 cm
c tilt aberration           -0.1 wave
c
c The misalignment consists of a -0.1 wavelength tilt on the smaller
c mirror. This misalignment causes the cavity to be more lossey and
c results in slower convergence and a poorer far-field spot than
c Example 11.
c
c RESCALE compresses the field size to a radius which is sized to minimize
c aliasing, but may. For this example the rescale size is entered as a
c variable value to facilitate adjusting the parameter.
c
c ENGNORM normalizes the beam energy to the assigned value after
c each pass.
c
c WRITE/DISK, and WRITE/TTY are used to reduce terminal
c printout.
c
c The variable pass_number is used to count the macro pass numbers and

```

Jump to: [Commands](#), [Theory](#)

```

c is input to the title. The variable STOP is used to test for
c convergence and pass the value to the IF statement for exiting the
c macro. The variable field_radius is used to initialize the array
c and rescale the array at the end of the macro.
c
c Variables
c -----
c STOP          switch to determine convergence
c pass_number   counts macro passes (integer)
c field_radius  half-width of array for rescale
c
poptext/compose ex12.txt
Unstable confocal resonator with misalignment.

A slight tilt misalignment causes the mode to be
skewed to the top of the annular outcoupled beam
poptext/end
poptext ex12.txt 8
variable/dec/int pass_number          # declare pass_number as integer
variable/dec/int STOP
c
c Define a macro which is a series of commands which represents one loop
c through the bare resonator cavity
c
macro/def misres/over
    pass_number = pass_number + 1      # increment pass counter
    write/screen/off                   # dispose unwanted output
    clap/cir/con 1 .3                  # aperture of .3 cm
    abr/tilt 1 -.1                     # misalignment tilt
    mirror rad=180                     # convex mirror
    prop 90                            # propagate 90 cm backward
    mirror rad=360.                    # concave mirror
    clap/cir/con 1 .7                  # aperture of .7 cm
    prop 90                            # forward propagation
    variable/set Energy 1 energy
    write/screen/on
    udata/set pass_number pass_number Energy-1
    gain/converge/test ibeams=1 nstore=STOP # store convergence test in STOP
    gain/eigenvalue/show 1             # display eigenvalues
    energy/norm 1 1
poptext/compose ex12.txt
Unstable confocal resonator with misalignment.

A slight tilt misalignment causes the mode to be
skewed to the top of the annular outcoupled beam
pass @pass_number of 30
poptext/end
poptext ex12.txt 12
    plot/liso 1
    pause 5
    plot/b/i/b 1
    pause 5
    if STOP macro/exit                 # conditional exit
macro/end
c
c Inialize variables
c
    pass_number = 0                    # Macro pass number
    field_radius = .6                  # Rescale field radius

```

Jump to: [Commands](#), [Theory](#)

```

c
c Establish initial units and a gaussian field distribution
c
units/field 1 field_radius          # define units
wavelength/set 1 10.                # define wavelength
gaus/cir/con 1 1 1 field_radius      # set initial distribution
gain/converge/set eps1=.005 eps2=.001 npoints=3
gain/eigenvalue/set 1

c
c eps1= fractional difference between min and max energy in last
c         npoints passes for convergence.
c eps2= fraction field change for convergence.
c
c Call the macro requesting up to 30 passes with conditional exit on
c convergence.
c
resonator/name misres                # set resonator name
resonator/eigen/test 1               # probe resonator
resonator/eigen/set 1                # set eigenradius
pass_number = 0                      # initialize pass number
clear 1 0                            # clear array
noise 1 1                            # seed with noise
resonator/run 30                     # run 30 time or to convergence
title ex 12: energy per step
plot/watch ex12_1.plt
poptext/compose ex12.txt
Unstable confocal resonator with misalignment.

Energy loss versus pass number
poptext/end
poptext ex12.txt 8
plot/udata max=0
pause 5

c
c plot converged field distribution.
c
title ex 12: resonator pass no. @pass_number
plot/watch ex12_2.plt
plot/liso 1

c
c Use obscuration of .3 cm. radius and display outcoupled beam.
c
obs 1 .3
title ex 12: outcoupled beam
plot/watch ex12_3.plt
poptext/compose ex12.txt
Unstable confocal resonator with misalignment.

Outcoupled beam shows mode skewed toward the top
poptext/end
poptext ex12.txt 8
plot/liso 1
pause 5

c
c Apply a lens and propagate to the far-field.
c
c Plot far-field pattern
c
lens/sph/res 1 100

```

Jump to: [Commands](#), [Theory](#)

```

prop 100
title ex 12: far-field pattern
plot/watch ex12_4.plt
poptext/compose ex12.txt
Unstable confocal resonator with misalignment.

```

```

Far-field of outcoupled beam
poptext/end
poptext ex12.txt 8
plot/liso 1
pause 5
c
c Generate a power-in-the-bucket table.
c
pib kbeam=1 nbrads=4 udata
title ex 12: power-in-the-bucket
plot/watch ex12_5.plt
poptext/compose ex12.txt
Unstable confocal resonator with misalignment.

```

```

Encircled energy of far-field irradiance
poptext/end
poptext ex12.txt 8
plot/udata 1 min=0. max=1.
pause 5
end

```


Ex13: Phase aberrations

Table. 13.1. Table of Ex13 examples

Ex13a: Display of various types of aberration.	1
Ex13b: Bitmap displays of Zernike aberration	5

This example shows some of the many types of aberration which may be added to the beam.

Ex13a: Display of various types of aberration

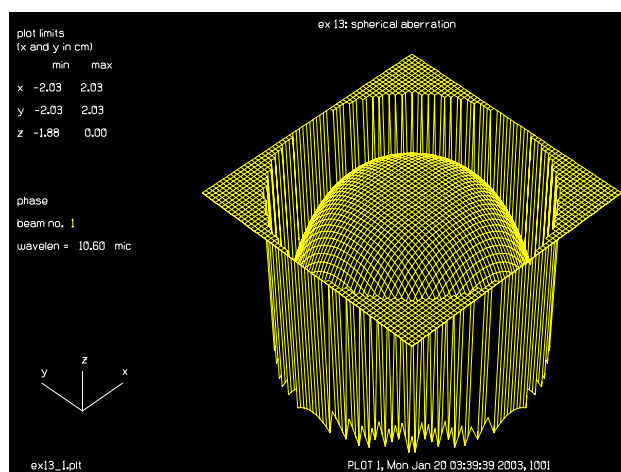


Fig. 13.1. Spherical aberration.

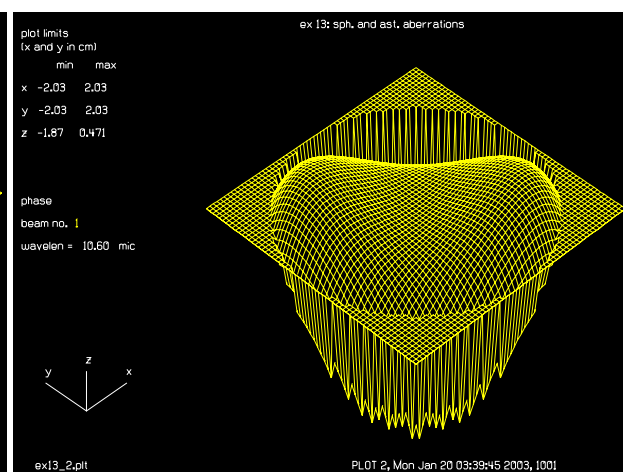


Fig. 13.2. Spherical aberration and astigmatism.

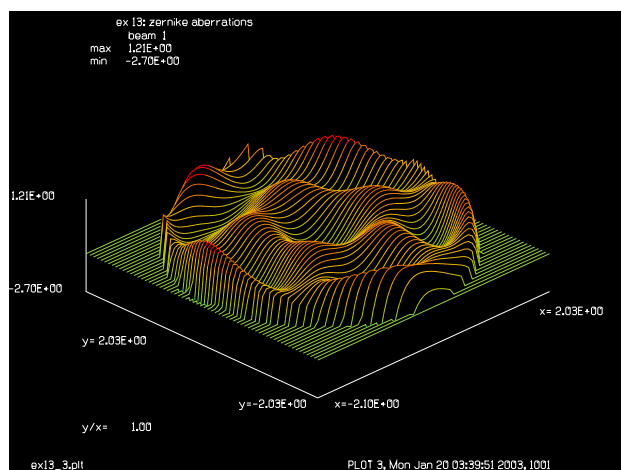


Fig. 13.3. Higher order Zernike aberrations.

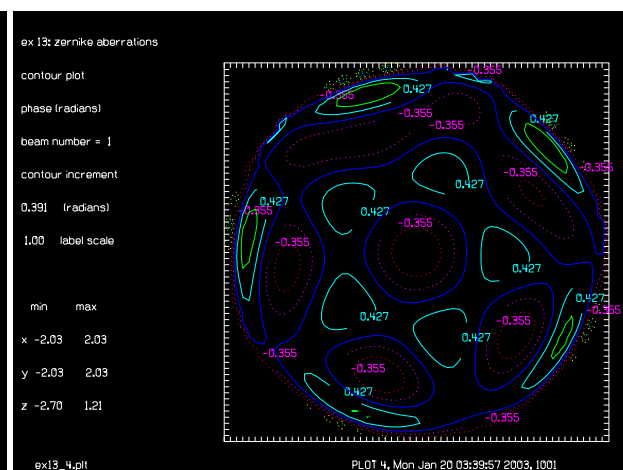


Fig. 13.4. Zernike aberrations, contour phase plot.

Input: `ex13.inp`

```
c## ex13
c
```

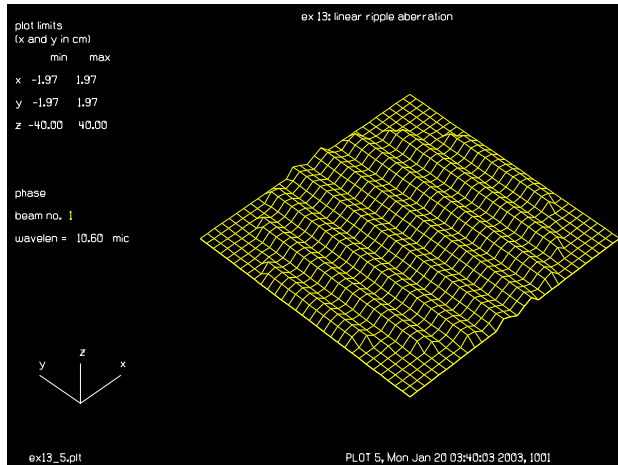


Fig. 13.5. Linear phase ripple for phase grating.

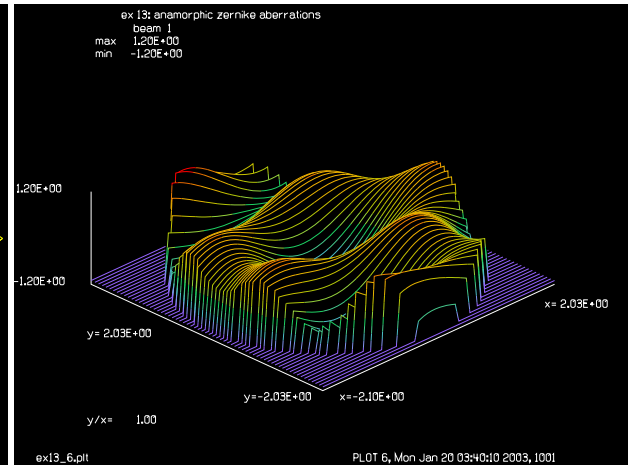


Fig. 13.6. Higher order anamorphic Zernike aberrations.

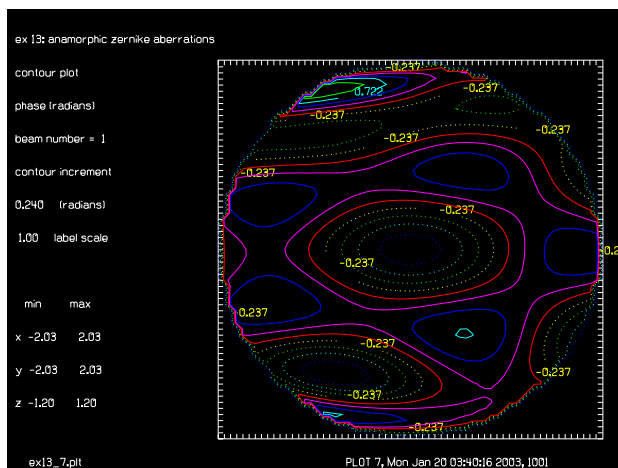


Fig. 13.7. Anamorphic Zernike aberrations, contour phase plot.

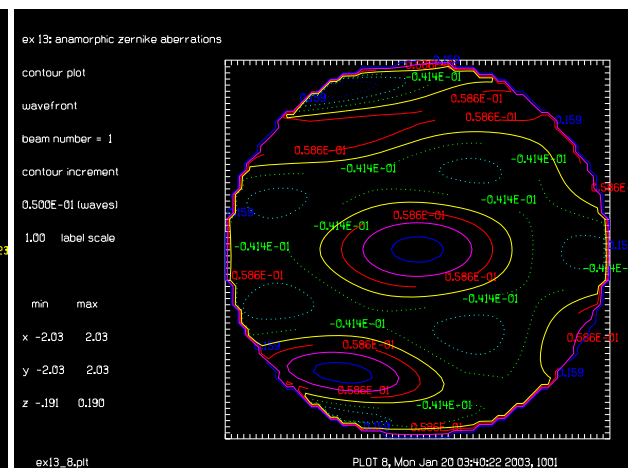


Fig. 13.8. Anamorphic Zernike aberrations, contour phase plot (wavefront form with even increments of 0.05).

```

c Example 13: Phase Aberrations
c
poptext/compose ex13.txt
GLAD can generate several types of aberrations, including the
Seidel and Zernike aberrations and smoothed random aberration.
poptext/end
poptext ex13.txt 8
    pause 5
c
c Generate a top hat beam of radius 2.
c
    units/field 1 2.1
    clear 1 1.
    clap/c/c 1 2.
c
c Generate sperical aberration of .3 waves at beam edge

```

Jump to: [Commands](#), [Theory](#)

```

c  ABR/SPH      Ibeams Ewav  Rnorm
c
  abr/sph      1      .3
  title ex 13: spherical aberration
  title/write
  plot/watch ex13_1.plt
poptext/compose ex13.txt
Spherical aberration.
poptext/end
poptext ex13.txt 5
  plot/iso/phase
  pause 5
c
c  Add astigmatism of .3 waves oriented at -45 degrees.
c  ABR/AST Ibeams  Ewav Azdeg Rnorm
c
  abr/ast      1  -.3  -45.
  title ex 13: sph. and ast. aberrations
  title/write
  plot/watch ex13_2.plt
poptext/compose ex13.txt
Spherical aberration and astigmatism.
poptext/end
poptext ex13.txt 5
  plot/iso/phase
  pause 5
c
c  Generate a few terms of Zernike aberrations
c
  clear 1 1.
  clap/c/c 1 2.
c
c  ABR/ZERN/Type Ibeams Nrad (Mazi) Coef Rnorm Rignore
c
  z = 1./-2./pi
  abr/zern/rad   1      8      -1.1*z
  abr/zern/cos   1      9      5      1.2*z
  abr/zern/sin   1      8      6      0.8*z
  set/density 64
  title ex 13: zernike aberrations
  plot/watch ex13_3.plt
poptext/compose ex13.txt
Complex mixture of high order Zernike aberrations
poptext/end
poptext ex13.txt 5
  plot/liso/phase ns=64
  pause 5
plot/watch ex13_4.plt
poptext ex13.txt 5
  plot/contour/phase
  pause 5
  clear 1 1.
  clap/c/c 1 2.
c

```

Jump to: [Commands](#), [Theory](#)

```

c  ABR/LRIP Ibeams   Ewav   Wnbr   Azdeg   Phi   Rnorm
c
c  abr/lrip   1       .3      4       90     90
c  set/density 32
c  title ex 13: linear ripple aberration
c  plot/watch ex13_5.plt
poptext/compose ex13.txt
Linear phase ripple (phase grating)
poptext/end
poptext ex13.txt 5
  plot/iso/phase max=40 min=-40
  pause 5
  clear 1 1.
  clap/c/c 1 2.
c
c  Anamorphic Zernike polynomialz using Xnorm and Ynorm to define normalized
c  pupil coordinates and Xignore and Yignore to define an elliptical
c  active region, outside of which the phase is not added
c
c  ABR/ZERN/Type      Ibeams Nrad   (Mazi) Coef   Xnorm  Ynorm  Xignore Yignore
c
c  echo/on
c  abr/zern/rad/ana   1       8       -1.1*z   4       2       2       2
c  abr/zern/cos/ana   1       9        5    1.2*z   4       2       2       2
c  abr/zern/sin/ana   1       8        6    0.8*z   4       2       2       2
c  set/density 64
c  title ex 13: anamorphic zernike aberrations
c  set/density 64
c  plot/watch ex13_6.plt
poptext/compose ex13.txt
Mixture of high order Zernike aberrations

anamorphic form
poptext/end
poptext ex13.txt 5
  plot/liso/phase ns=64
  pause 5
  plot/watch ex13_7.plt
poptext ex13.txt 5
  plot/contour/phase
  pause 5
c
c  contour plot in the form of wavefront error
c
c  wavefront = -phase/(2 pi)
c
c  plot with increments of .05 waves
c
c  plot/watch ex13_8.plt
poptext/compose ex13.txt
Mixture of high order Zernike aberrations

anamorphic form (displayed in waves of aberration)
poptext/end

```

Jump to: [Commands](#), [Theory](#)

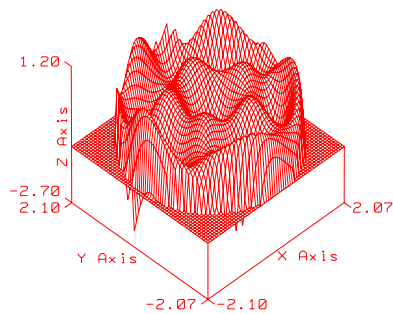
```

poptext ex13.txt 5
  plot/contour/wavefront coninc=.05 incre
  pause 5
end

```

Ex13b: Bitmap displays of Zernike aberration

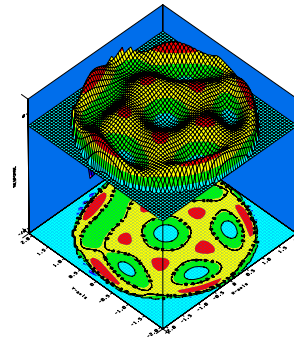
ex 13b: zernike aberrations



phase, Beam 1
Z-scale 1.000000, xy-scale 1.000000

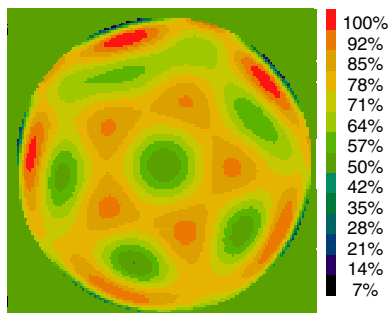
Fig. 13.9. Ex13b anamorphic with
plot/bitmap/phase/wire.

ex 13b: zernike aberrations



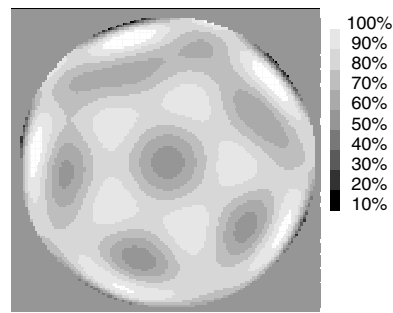
phase, Beam 1
Z-scale 1.000000, xy-scale 1.000000

Fig. 13.10. Ex13b, plot/bitmap/phase/coniso.



ex 13b: zernike aberrations
phase, Beam 1
range: -2.908E+00, 1.202E+00,
x-limits: -2.100E+00, 2.067E+00,
y-limits: -2.067E+00, 2.100E+00,
PLOT 3, Mon Jan 20 03:44:52 2003, 1001

Fig. 13.11. Ex13b, plot/bitmap/phase/burn.



ex 13b: zernike aberrations
phase, Beam 1
range: -2.908E+00, 1.202E+00,
x-limits: -2.100E+00, 2.067E+00,
y-limits: -2.067E+00, 2.100E+00,
PLOT 4, Mon Jan 20 03:44:53 2003, 1001

Fig. 13.12. Ex13b. plot/bitmap/phase/gray.

Input: ex13b.inp

```

c## ex13b
c
c Example 13b: Various displays of phase aberrations
c
alias Name ex13b
array/s 1 128
units/field 1 2.1

```

Jump to: [Commands](#), [Theory](#)

```
clear 1 1.
clap/c/c 1 2.
c
c  ABR/ZERN/Type Ibeams Nrad (Mazi) Coef Rnorm Rignore
c
z = 1./-2./pi
abr/zern/rad      1      8      -1.1*z
abr/zern/cos      1      9      5    1.2*z
abr/zern/sin      1      8      6    0.8*z
set/density 64
title ex 13b: zernike aberrations
plot/watch @Name_1.plt
c plot/l/ph 1 ns=64
plot/b/phase/wire
plot/w @Name_2.plt
set/den 64 64
plot/b/ph/c 1
set/den 128 128
plot/w @Name_3.plt
plot/b/ph/b 1
plot/w @Name_4.plt
plot/b/ph/g 1
```

Ex14: Beam fitting

This example illustrates the capability of GLAD to fit a surrogate gaussian beam to a beam intensity distribution and to remove focus and tilt from the wavefront. The example also illustrates fitting Zernike polynomial coefficients to a smoothed random wavefront.

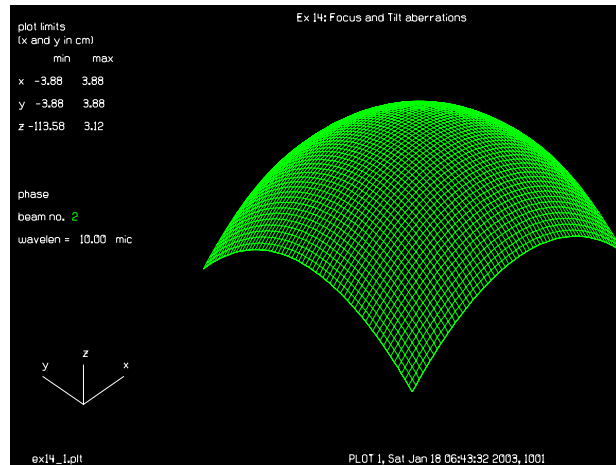


Fig. 14.1. Focus and tilt aberration.

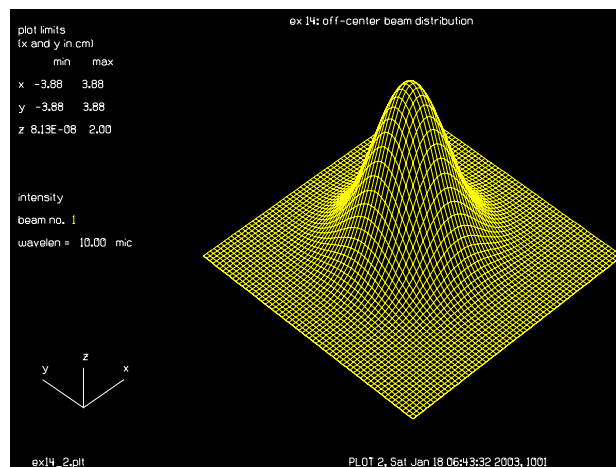


Fig. 14.2. Off-center beam distribution.

Input: ex14.inp

```
c## ex14
c
c Example 14: Beam Fitting, Intensity and Phase
c
c This example illustrates the capability of GLAD to fit a surrogate
c gaussian beam to a beam intensity distribution and to remove focus
c and tilt from the wavefront. The example also illustrates
c fitting Zernike polynomial coefficients to a smoothed random
c wavefront.
c
```

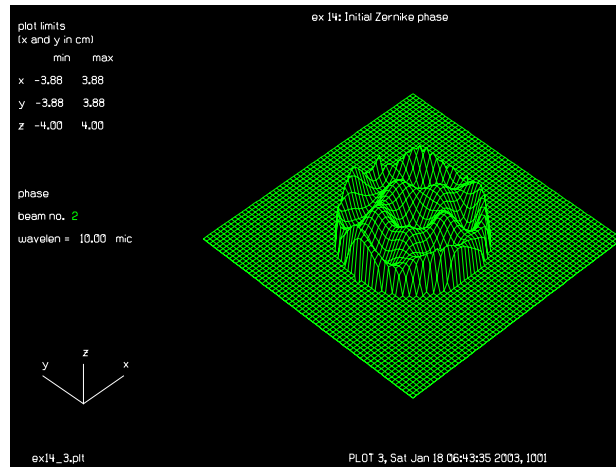


Fig. 14.3. Initial Zernike phase.

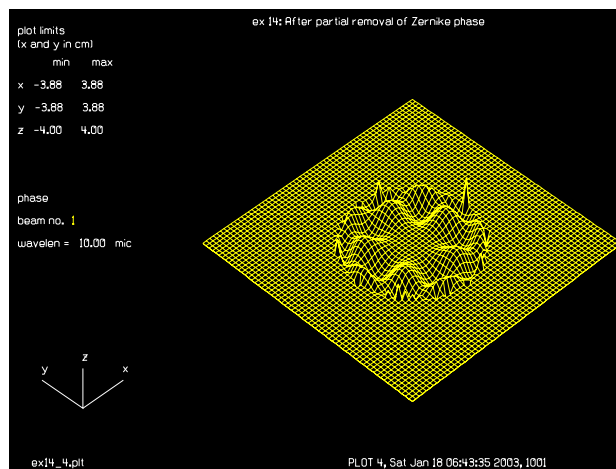


Fig. 14.4. After partial removal of Zernike phase.

```

c Initialize two beams with a wavelength of 10. microns
c
c   echo/on
c   nbeam 2
c   wavelength/set 0 10.
c   array/set 0 128
c
c Generate an offset gaussian beam in Beam 2 and apply focus and tilt
c phase aberrations.
c
c The aberration coefficient for tilt error is:
c   Et = Tilt*Rnorm/Wavelength
c   2.0 = .001 * 2. / .0010
c
c The aberration coefficient for a focus error is:
c   Ef = focallength*Rnorm**2/(2.*Wavelength)
c   2.0 = 1000. *2.**2/(2*0.0010)
c
c   units/field 2 4.

```

Jump to: [Commands](#), [Theory](#)


```

gaussian/cir/con 2 2 2. 1. .2 .3
abr/tilt ibeams=2 ewav=2. azdeg=30. xdec=.2 ydec=.3
abr/focus ibeams=2 ewav=2. xdec=.2 ydec=.3
c
c Display phase distribution
c
  set/density 43
  title Ex 14: Focus and Tilt aberrations
  plot/watch ex14_1.plt
  plot/iso/phase first=2 last=2
c
c Add Beam 2 into Beam 1 coherently with the field of Beam 1 remaining
c at the origin. Beam 1 must be cleared to zero intensity first since
c the ADD command accumulates beams into the output beam number
c
  clear 1 0
  units/field 1 4.
  add/coh/con 1 2
  energy
c
c Since the array points do not match a linear interpolation is used
c in the beam addition that results in some change in total energy.
c Also a point does not exist on the center of the beam. Display a
c plot of the intensity of the off-center beam distribution.
c
  title ex 14: off-center beam distribution
  plot/watch ex14_2.plt
  plot/iso/int last=1
c
c Use the FITGEO command to find the intensity centroid.
c The PEAK command will find peak intensity for both beams.
c
  fitgeo/spatial/list 1
  peak
c
c Use the FITGEO command to center the beam at the point nearest the
c centroid. Another factor that effects the centroid calculation
c is the roundoff at the edge. Display the edge points of beam 1 and 2
c
  field 1 61 1 6
  field 1 61 123 128
  field 2 0 1 5
  field 2 0 125 128
  fitgeo 1
  status/parax
c
c Test the intensity fitting for a top-hat beam
c
  gaussian/cir/con 0 2 2 50
  clear 1 0
  add/coh/con 1 2
  fitgeo 1
  status/parax
  status/haus

```

```
c
c Generate the Zernike aberrations from the previous example and fit
c the phase with FITZERN
c
  gaus/c/c 1 2 2.5 50
  clear 1 1
  clap/cir/con 1 2
  z = 1./-2/pi
  abr/zern/rad 1 8 -1.1*z
  abr/zern/cos 1 9 5 1.2*z
  abr/zern/sin 1 8 6 .8*z
  copy/con 1 2
  fitzern/list kbeam=1 nrad=9 nsin=6 ncos=5
c
c Reduce the order of the fitting polynomial to fit only the lower two
c terms.
c
  fitzern/remove 1 8 6 4
c
c Display plot of the residual phase and the fitted phase.
c
  title ex 14: Initial Zernike phase
  plot/watch ex14_3.plt
  plot/iso/phase first=2 last=2 max=4 min=-4
  title ex 14: After partial removal of Zernike phase
  plot/watch ex14_4.plt
  plot/iso/phase first=1 last=1 max=4 min=-4
  end
```

Ex15: General obscuration

This example illustrates the general obscuration feature of GLAD for modeling a spider. A file called 'spoke.inp' contains the points defining a single spoke. The obs/gen command is then used to position and rotate these points to generate the spider.

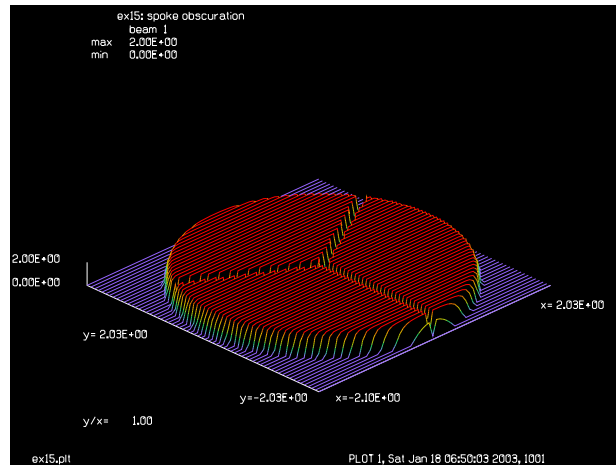


Fig. 15.1. Off-center beam distribution.

Input: ex15.inp

```
c## ex15
c
c Example 15: General Obscurations: Spider
c
c This example illustrates the general obscuration feature of GLAD for
c modeling a spider. A file called 'spoke.dat' contains the
c points defining a single spoke. The OBS/GEN command is then used to
c position and rotate these points to generate the spider.
c
c Initialize the beam and field array.
c
units/field 1 2.1
gaussian/cir/con 1 2 2 50
clap/cir/con 1 2.
c
c Generate a file called 'spoke.inp' that contains the following information.
c
c      6
c      -.1, 0.
c      -.1, 1.
c      .1, 1.
c      .1, -1.
c      -.1, -1.
c      -.1, 0.
c
c Generate a spider with half widths of .05 cm. Use three equally spaced
c spokes with one directed in the -y direction.
c
c OBS/GEN      (Fname)   Ibeams Xdec Ydec Xscale Yscale Theta (deg)
obs/gen/disk spoke.dat   1      0.  -1.   .5      1.      0.
```

```
obs/gen/disk spoke.dat    1    .866  .5    .5    1.    -60.
obs/gen/disk spoke.dat    1   -.866  .5    .5    1.     60.
c
c Display result with an integer map
c
  intmap 1
  title ex15: spoke obscuration
  plot/watch ex15.plt
  plot/l 1 1 h=.1 ns=64
```

Ex16: General apertures and obscurations

This is an example of an unusual aperture, the use of a very large array (512×512), and some of the plotting capabilities. In addition to the standard aperture shapes (circular, elliptical, square, and rectangular) GLAD can make apertures of arbitrary shape with `clap/gen` and `obs/gen`.

This example illustrates an aperture consisting of the letters AOR. A large array of 512×512 is chosen to provide high sampling. The effects of diffraction are illustrated by propagating the beam after the aperture. Several different graphic displays are used to show all of the beam and selected portions. Since there are so many points, it is difficult to resolve all of the data if the whole beam is displayed. `plot/iso` is limited to an array of 64×64 and GLAD interpolates the data to fit into this format. The details of the array can be examined by using the `set/window.plot/liso` uses maximum resolution in one direction with a variable number of slices.

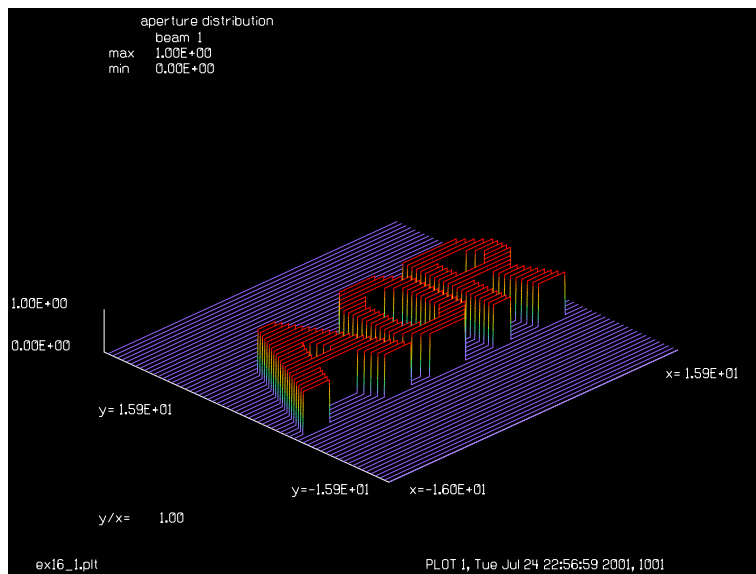


Fig. 16.1. Representation of the letters AOR with apertures and obscurations.

Input: ex16.inp

```
c## ex16 !574081459751386
c
c Example 16: General Apertures and Obscurations
c
c Example of unusual aperture and some of the plotting capabilities.
c In addition to the standard aperture shapes (circular, elliptical,
c square, and rectangular) GLAD can make apertures of arbitrary
c shape with CLAP/GEN and OBS/GEN. This example illustrates
c a sample aperture consisting of the letters AOR.
c
c A large array size of 512 X 512 is chosen to provide high sampling.
c The effects of diffraction are illustrated by propagating the
c beam after the aperture. Several different graphic displays are
c used to show all of the beam and selected portions. Since there
c are so many points, it is difficult to resolve all of the data
c if the whole beam is displayed.
c
```

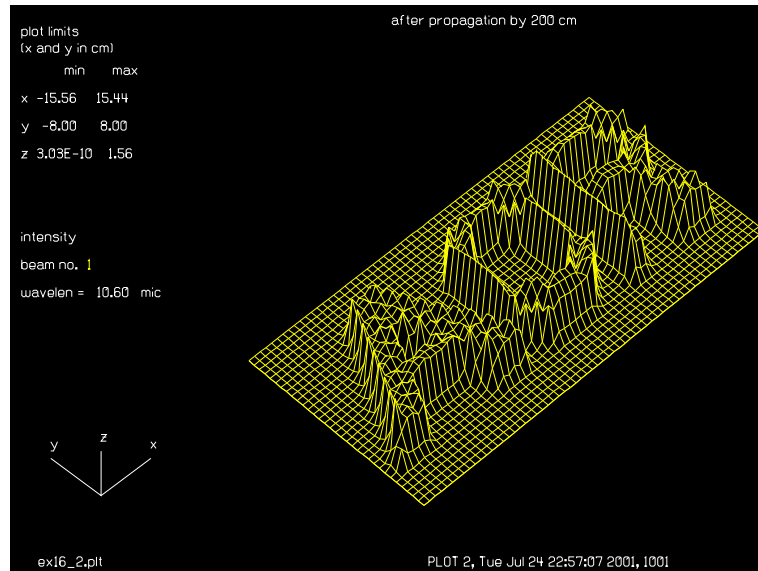
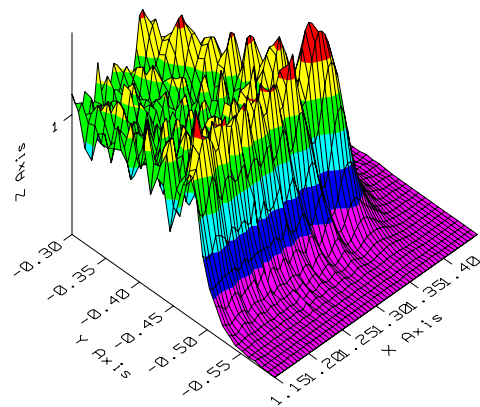


Fig. 16.2. Aperture function after propagation by 200 cm. The detail in the array is far greater than may be readily displayed when viewing the full array.

foot of r expanded



intensity, Beam 1
z-scale 1.000000, xy-scale 10.000000

Fig. 16.3. A detailed view of the foot of the R is shown to illustrate the level of detail in the 512×512 array.

```
poptext/compose ex16.txt
Diffraction of light through the AOR logo.
```

A combination of general apertures and
general obscurations is used to spell out A O R

An optical beam is then propagated a short distance
after the aperture.

```
poptext/end
poptext ex16.txt 8
mem/set/b 2
array/set 1 512          # set array size
wavelength 1 10.6        # set wavelength to 10.6 microns
```

Jump to: [Commands](#), [Theory](#)

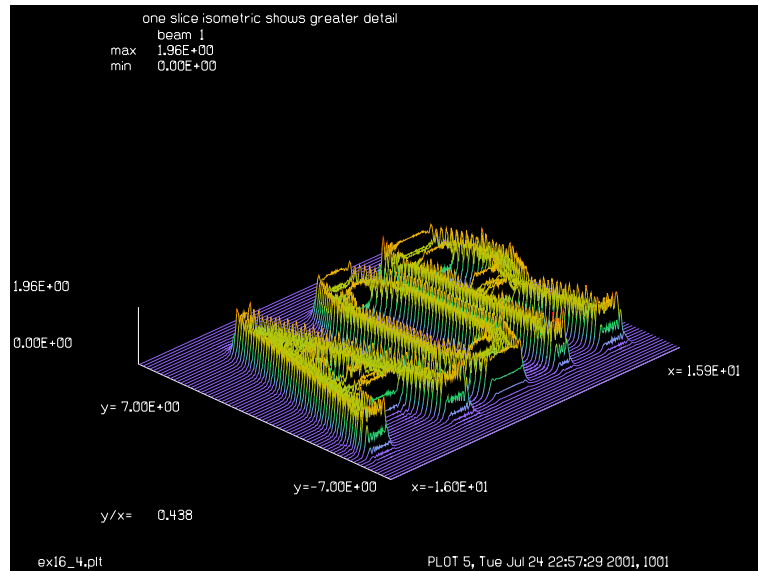


Fig. 16.4. The `plot/liso` command shows greater detail in the horizontal slices than `plot/iso`.

```

units 1 .0625          # set array units
clap/gen 1             # outline all letters with general aperture
4
-15 -5
14 -5
14 5
-15 5
obs/gen 1              # cut out left side of A
3
-15 -5
-10 5
-15 5
obs/gen 1              # cut out bottom of A
4
-12 -5
-11 -3
-7 -3
-6 -5
obs/gen 1              # cut out hole in A
3
-10 -1
-9 1
-8 -1
obs/gen 1              # cut out area between A and O
6
-3 -5
-1 -5
-3 -3
-3 3
-1 5
-8 5
obs/gen 1              # cut out inside of O
8
0 -3
2 -3
3 -2

```

```

3 2
2 3
0 3
-1 2
-1 -2
obs/gen 1          # cut out between O and R
6
3 -5
6 -5
6 5
3 5
5 3
5 -3
obs/gen 1          # cut out bottom of R
4
8 -5
8 -1
10 -1
11 -5
obs/gen 1          # cut out right side of R
3
14 -5
12 -1
14 1
obs/gen 1          # cut out top right corner of R
3
14 3
12 5
14 5
obs/gen 1          # cut out hole in R
5
8 1
8 3
11 3
12 2
11 1
intmap             # make simple integer map
set/density 64 32
set/window/abs -16 16 -8 8          # Set plot windows
title aperture distribution
plot/watch ex16_1.plt              # Plot 1 to disk
poptext/compose ex16.txt
Diffraction of light through the AOR logo.

Aperture for A O R
poptext/end
poptext ex16.txt 8
plot/iso height=.2 elevation=50
pause 5
dist 200                      # Take diffraction step
title after propagation by 200 cm
plot/watch ex16_2.plt          # Plot 2 to disk
poptext/compose ex16.txt
Diffraction of light through the AOR logo.

Aperture for A O R

After short propagation
poptext/end

```

Jump to: [Commands](#), [Theory](#)


```

poptext ex16.txt 8
plot height=.2 elevation=50          # Plot after diffraction
pause 5
set/window/abs 11.5 14.5 -6 -3      # Zoom in on foot of R
title foot of r expanded
plot/watch ex16_3.plt               # Plot 3 to disk
poptext/compose ex16.txt
Diffraction of light through the AOR logo.

Aperture for A O R after short propagation

windowed to show foot of "R"
poptext/end
poptext ex16.txt 8
plot height=.2 elevation=30
pause 5
title one slice isometric shows greater detail
plot/watch ex16_4.plt               # Plot 4 to disks
poptext/compose ex16.txt
Diffraction of light through the AOR logo.
after short propagation

'plot/1' style shows high horizontal resolution
poptext/end
poptext ex16.txt 8
plot/liso nslice=64 max=5 xrad=16 yrad=7    # High resolution plot
pause 5
end

```


Ex17: Raman amplifier

This example illustrates Raman amplification of a seed beam by a shorter wavelength pump beam. The pump beam is at 1.06 microns and the seed beam is at 1.54 microns. Both the seed and pump beam are aberrated. The seed beam goes through a spatial filter which cleans up the beam somewhat. The seed and pump beams are combined and passed through a Raman amplifier. The amplifier depletes the pump beam and amplifies the seed beam. In this model none of the phase of the pump beam is imposed on the seed beam. The irradiance distribution of both the pump and seed beams show effects due to diffraction from the aperture edges and due to the aberration in the beams. The output of the seed beam is brought to a line focus by a cylindrical lens to illustrate the use of nonrotationally symmetrical optical elements and the use of rectangular arrays. Note that the code automatically determines the matrix units to achieve good resolution in both directions.

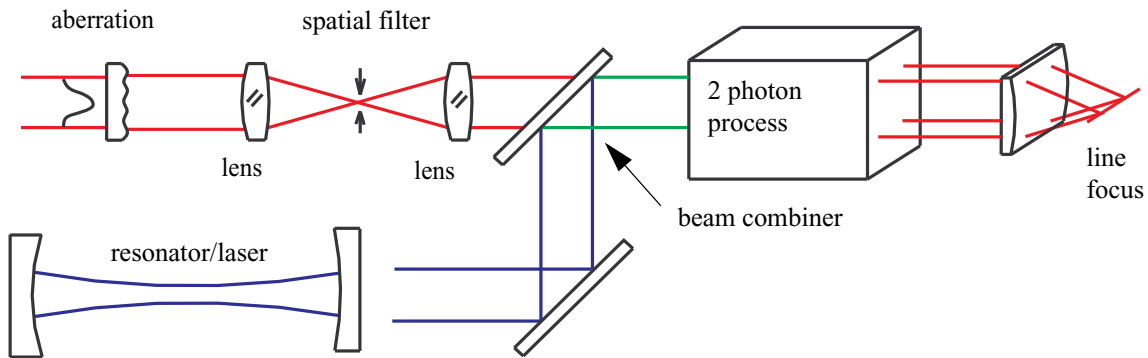


Fig. 17.1. Schematic for Raman amplification (not true scale).

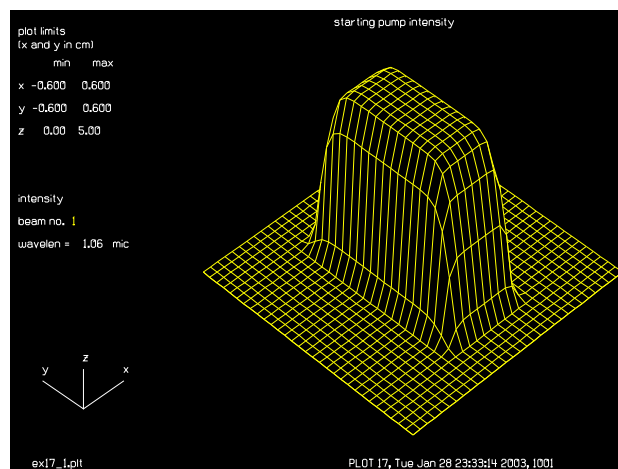


Fig. 17.2. Initial pump intensity.

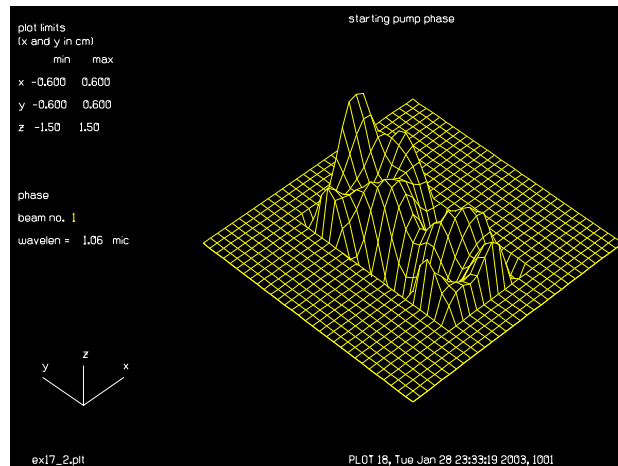


Fig. 17.3. Initial pump phase showing random aberration.

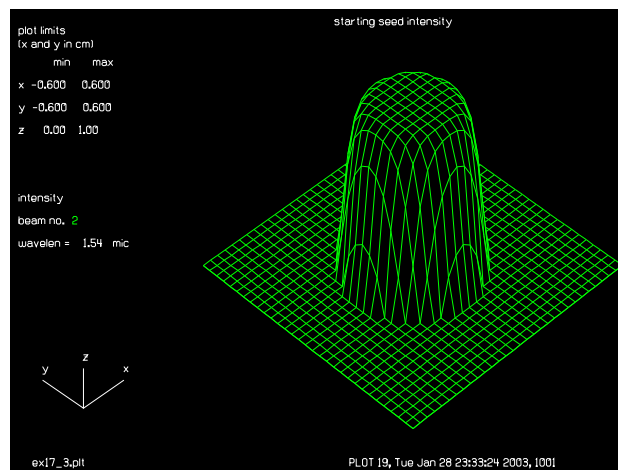


Fig. 17.4. Initial seed intensity.

Input: `ex17.inp`

```
c## ex17
c
c Example 17: Raman Amplification
c
c This example illustrates Raman amplification of a seed beam by a shorter
c wavelength pump beam. In this example the pump beam is at 1.06 microns and
c the seed beam is at 1.54 microns. Both the seed and pump beam are
c aberrated. The seed beam goes through a spatial filter which cleans up the
c beam somewhat. The seed and pump beams are combined and passed through a
c Raman amplifier. The amplifier depletes the pump beam and amplifies the
c seed beam. In this model none of the phase of the pump beam is imposed on
```

Jump to: [Commands](#), [Theory](#)

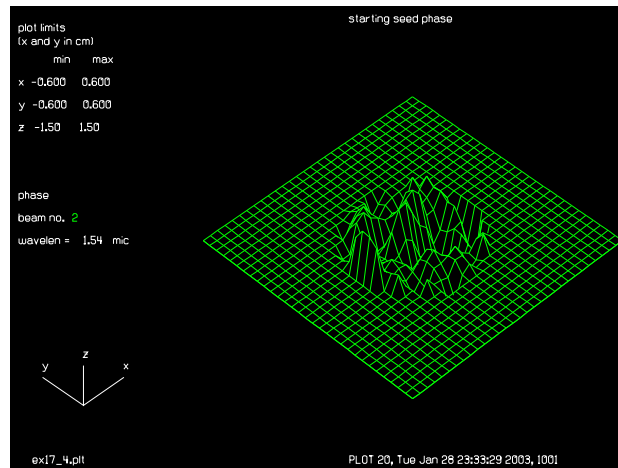


Fig. 17.5. Initial seed phase showing random aberration.

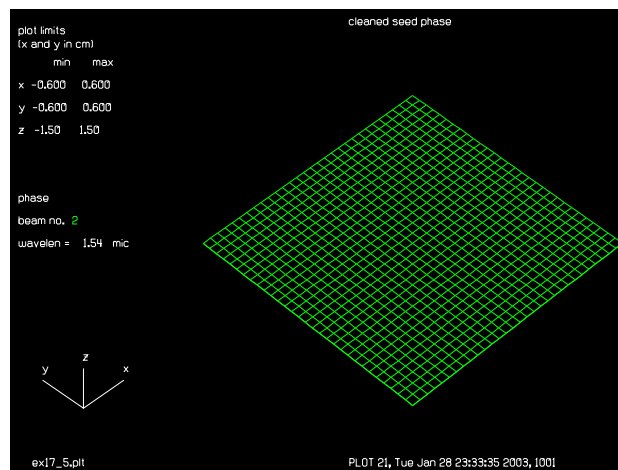


Fig. 17.6. Phase of seed after spatial filter. Note, residual low-order aberration.

c the seed beam. The irradiance distribution of both the pump and seed beams
 c show effects due to diffraction from the aperture edges and due to the
 c aberration in the beams.

c
 c The the seed beam is brought to a line focus by a cylindrical lens
 c to illustrate the use of nonrotationally symmetrical optical elements and
 c the use of rectangular arrays. Note that the code automatically determines
 c the matrix units/s to achieve good resolution in both directions.

c
 popntext/compose ex17.txt
 Example of Raman application

The pump beam (yellow) has aberrations but pumps
 the seed beam (green)

Jump to: [Commands](#), [Theory](#)

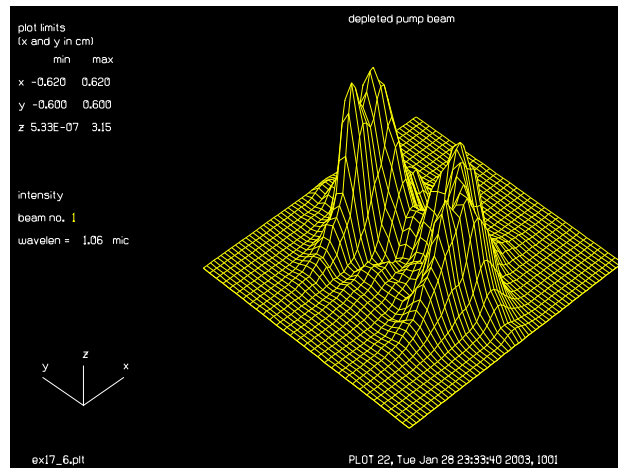


Fig. 17.7. Intensity of depleted pump after Raman amplification showing that the power in the center has been reduced in the center by the amplification of the seed.

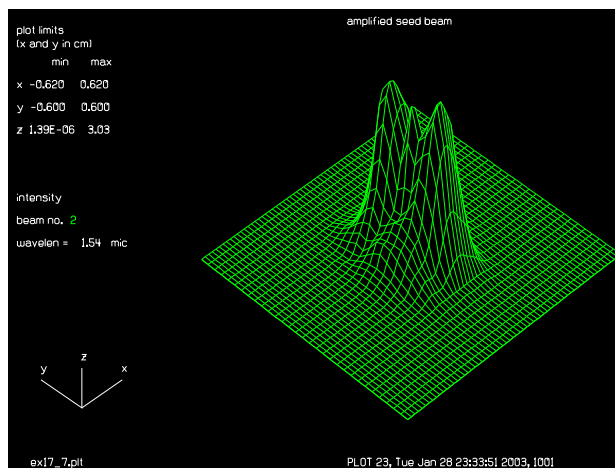


Fig. 17.8. Amplified seed.

The seed beam is initially aberrated but is cleaned by a spatial filter. It is then amplified by the pump with no transfer of aberration.

The pump beam shows depletion in the center.
The seed beam shows transfer of irradiance nonuniformity which ultimately converts into some phase aberration.

```
poptext/end
echo/on
poptext ex17.txt 8
pause 5
set/density 32
```

Jump to: [Commands](#), [Theory](#)

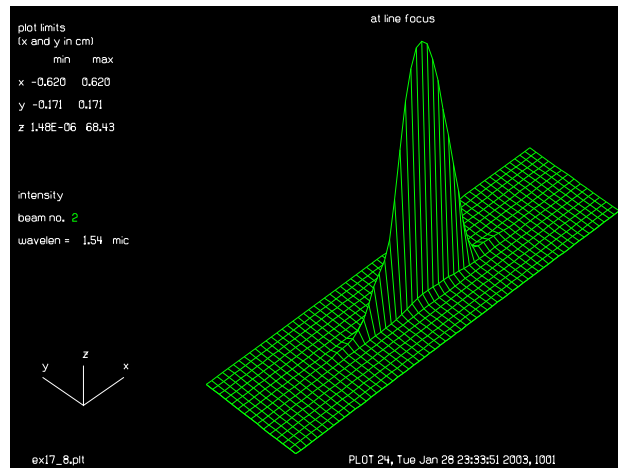


Fig. 17.9. Seed after focusing by the cylindrical lens.

```

nbeam 2                                # set up number of beams
array/set 0 64                          # define array size
units/set 0 .02                         # define units
c
c ** Initialize pump, Beam 1
c
  global/def 1 50. 0 300.                # define start of pump beam
  wavelength/set 1 1.06                  # define pump beam wavelength
  gaussian/rec/con 1 5. .2 .4 5 5        # pump beam shape
  clap/rec/con 1 .2 .4                   # rectangular aperture on pump
  title starting pump intensity
c                                          # Plot 1, starting pump intensity
  plot/watch ex17_1.plt
  plot/iso/intensity first=1 last=1
poptext/compose ex17.txt
Example of Raman application

Pump beam starting irradiance
poptext/end
poptext ex17.txt 4
  pause 4
  phase/random 1 .1 .1                   # smoothed random aberration
  title starting pump phase
c                                          # Plot 2, starting pump phase
  plot/watch ex17_2.plt
  plot/iso/phase first=1 last=1 max=1.5 min=-1.5
poptext/compose ex17.txt
Example of Raman application

Pump beam aberrations
poptext/end
poptext ex17.txt 4
  pause 4
c
c ** Initialize seed, Beam 2
c

```

Jump to: [Commands](#), [Theory](#)

```

global/def 2 0. 0. 0.          # define start of seed beam
wavelength/set 2 1.54          # define seed beam wavelength
gaussian/cir/con 2 1. .3 5     # seed beam shape
clap/cir/con 2 .3              # circular aperture on seed
title starting seed intensity

c                               # Plot 3, starting seed intensity
    plot/watch ex17_3.plt
    plot/iso/intensity first=2 last=2
poptext/compose ex17.txt
Example of Raman application

Seed beam starting irradiance distribution
poptext/end
poptext ex17.txt 4
    pause 4
    phase/random 2 .05 .05      # smoothed random aberration
    title starting seed phase

c                               # Plot 4, starting seed phase
    plot/watch ex17_4.plt
    plot/iso/phase first=2 last=2 max=1.5 min=-1.5
poptext/compose ex17.txt
Example of Raman application

Seed beam starting aberrations
poptext/end
poptext ex17.txt 4
    pause 4

c
c ** Pump beam path, Beam 1
c
    beams/on 1                  # turn on only pump
    beams/off 2
    vertex/locate/absolute 50. 0. 400. # fold mirror
    vertex/rotate/set 0. 45. 0.
    mirror/global/flat
    vertex/locate/absolute 0. 0. 400. # beam splitter
    vertex/rotate/set 0. -135. 0.
    mirror/global/flat          # attenuate at beam splitter
    mult 1 .5
    prop 100                    # propagate to Raman cell

c
c ** Seed beam path, Beam 2
c
    beams/on 2                  # turn on only seed beam
    beams/off 1
    prop 100                    # spatial filter
    lens 2 100
    prop 100
    status/parax
    clap/cir/con 2 .03
    prop 100
    lens 2 100
    clap/cir/con 2 .3           # aperture at end of spatial filter
    fitphase/both/list 2
    title cleaned seed phase

c                               # Plot 5, cleaned seed phase
    plot/watch ex17_5.plt
    plot/iso/phase first=2 last=2 max=1.5 min=-1.5
echo/on

```

Jump to: [Commands](#), [Theory](#)


```

poptext/compose ex17.txt
Example of Raman application

Seed beam aberrations after cleanup in spatial filter
poptext/end
poptext ex17.txt 4
    pause 4
    prop 100
    mult 2 .5                                # attenuation at beam splitter
    beams/on 1 2                             # turn on both beams
c
c ** Joint path
c
    energy
    raman 1 2 100. .005                     # Raman cell
    energy
    title depleted pump beam
c                                           # Plot 6, depleted pump
    set/density 64 32
    plot/watch ex17_6.plt
    plot/iso/intensity first=1 last=1
poptext/compose ex17.txt
Example of Raman application

Partially depleted pump beam
poptext/end
poptext ex17.txt 4
    pause 4
    title amplified seed beam
poptext/compose ex17.txt
Example of Raman application

Amplified seed beam showing pump irradiance printthrough
from pump beam
poptext/end
poptext ex17.txt 4
    pause 4
c                                           # Plot 7, amplified seed
    plot/watch ex17_7.plt
    plot/iso/intensity first=2 last=2
    lens/ycyl 2 50.                          # cylindrical lens
    prop 50.                                 # propagation to line focus
    title at line focus
c                                           # Plot 8, at line focus
    plot/watch ex17_8.plt
    set/density 64 16
    plot/iso/intensity first=2 last=2
poptext/compose ex17.txt
Example of Raman application

Line focus of seed beam
poptext/end
poptext ex17.txt 4
    pause 5
end

```


Ex18: Raman amplification with multiple Stokes beams

This example illustrates Raman amplification of a seed beam by a shorter wavelength pump beam. The pump beam is at 1.06 microns and the seed beam is at 1.54 microns. Both the seed and pump beam are aberrated. The seed beam goes through a spatial filter which cleans up the beam somewhat. The seed and pump beams are combined and passed through a Raman amplifier. The amplifier depletes the pump beam and amplifies the seed beam. In this model none of the phase of the pump beam is imposed on the seed beam. The irradiance distribution of both the pump and seed beams show effects due to diffraction from the aperture edges and due to the aberration in the beams. The output of the seed beam is brought to a line focus by a cylindrical lens to illustrate the use of nonrotationally symmetrical optical elements and the use of rectangular arrays. Note that the code automatically determines the matrix units to achieve good resolution in both directions.

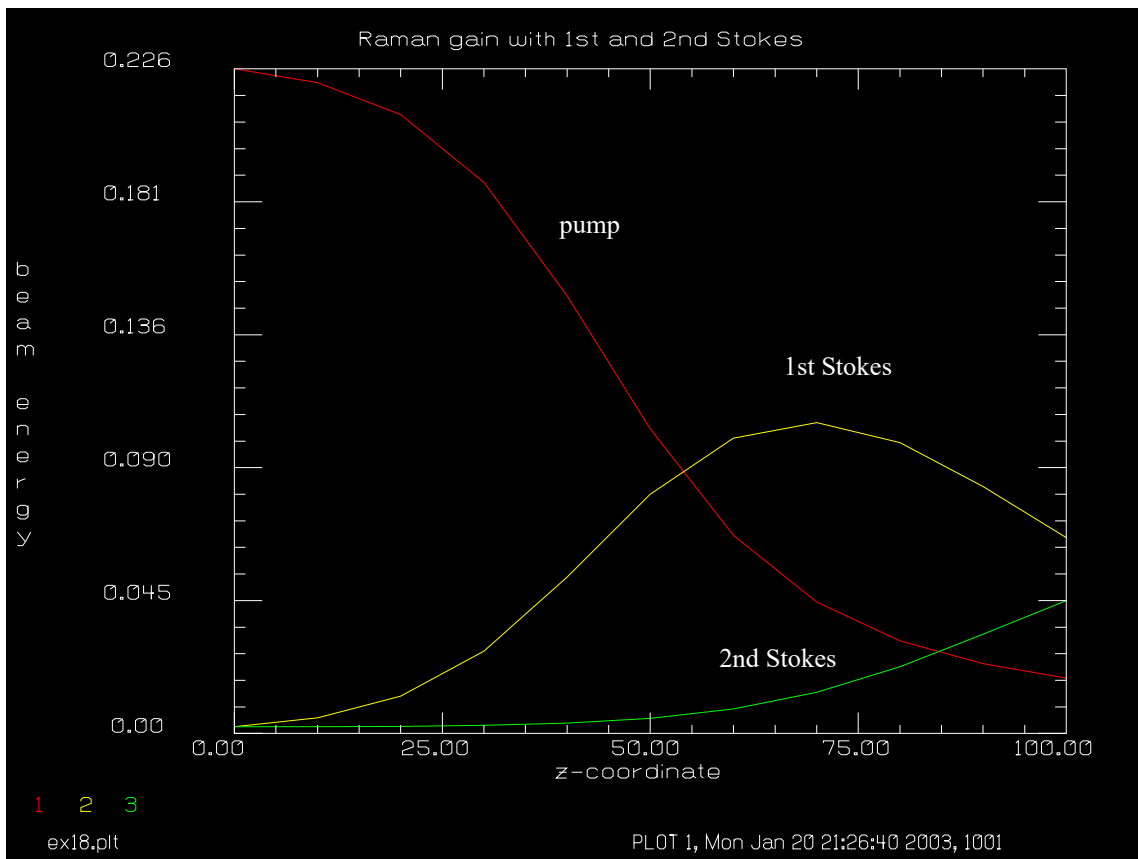


Fig. 18.1. Energy cascades from the pump beam into successive Stokes beams.

Input: ex18.inp

```
c## ex18
c
c Example 18: Raman Amplification with Multiple Stokes Beams
c
c This example demonstrates the Raman amplification with first and
c second Stokes beams.
c
```

```

variab/dec/int pass
macro/def ramstep/overwrite                                # define macro
    pass = pass + 1                                         # increment pass count
    z = z + 10.                                             # increment z-coord
    pack/in
        raman/collimated/noprop 1 2 10. .1                # pump and 1st Stokes gain
        raman/collimated/noprop 2 3 10. .1                # 1st and 2nd Stokes gain
    pack/out
    prop 10.
    variab/set energy1 1 energy                             # Beam 1 energy saved
    variab/set energy2 2 energy                             # Beam 2 energy saved
    variab/set energy3 3 energy                             # Beam 3 energy saved
    # Store plot/udata info
    udata/set pass z energy1 energy2 energy3
macro/end
echo/on                                                     # turn on echo
nbeam 3                                                     # define three beams
wavelength/set 1 1.06                                       # pump beam, YAG laser
wavelength/set 2 1.54                                       # 1st Stokes beam
wavelength/set 3 2.81                                       # 2nd Stokes beam
array/set 0 64                                              # initialize array size
units/set 0 .02                                             # set units
gaussian/cir/con 1 1.00 .3 5                               # set pump beam
gaussian/cir/con 2 .01 .3 5                                # set 1st Stokes
gaussian/cir/con 3 .01 .3 5                                # set 2nd Stokes
c
c Initialize PLOT/UDATA info
c
pass = 1                                                     # initialize data point count
z = 0.                                                       # initialize z-coord
variab/set energy1 1 energy                                 # Beam 1 energy saved
variab/set energy2 2 energy                                 # Beam 2 energy saved
variab/set energy3 3 energy                                 # Beam 3 energy saved
# Store plot/udata info
udata/set pass z energy1 energy2 energy3
c
c Gain steps
c
pack/set k1beam=1 k2beam=2 k3beam=3                        # set up pack arrays
write/screen/off                                           # turn off output
macro/run ramstep/10
write/screen/on                                             # restore output to screen
c
c Set up plot of energy of beams
c
udata/xlabel z-coordinate
udata/ylabel beam energy
title Raman gain with 1st and 2nd Stokes
udata/list                                                  # list udata information
echo/off
plot/w ex18.plt
plot/udata 1 3 min=0.
end

```

Ex19: Raman process with converging beams: simple kinetics step

This example illustrates application of GLAD to Raman interaction through focus. It is necessary to take many steps through focus to correctly calculate the Raman interactions. The interaction in the focus region requires correct treatment of the different sizes of the arrays and a method of choosing the position of the steps.

In this problem the pump and starting Stokes seed distributions are made identical. The optimum array sizes in the region of the focus are different by the ratio of the wavelengths. GLAD will interpolate information in arrays of different sizes so that kinetics may be applied. The interpolation introduces errors particularly when the wavelengths are separated widely as in this problem with the pump at 1.06 microns and the Stokes at 1.54 microns. The zone command allows the user to control the algorithms so that arrays that are initially identical may be kept identical through focus. The pack routines detect the identical sizes and skip the interpolation step.

The zone command may be used to replace the automatic control of diffraction algorithms by the Rayleigh ranges of the individual beams. The `zone/fix` or `zone/xyfix` command specifies the center of the zone analogous to the Rayleigh waist and a halfwidth similar to the Rayleigh range. The zone parameters hold for all beams. Consequently if the beams are started identically they will remain so. While interpolation errors are removed, the user should be aware that aliasing errors are greater because each of the array sizes is not chosen for minimum aliasing. In this problem the zone is defined with the center at the focus of the lens and the halfwidth is chosen to be the Rayleigh distance for the 1.06 wavelength. The 1.54 beam will overfill the array near the focus. However, the effects are relatively small.

The method of selecting the step positions is outlined below. The method used here is to take a step at each point that the 1.06 micron (in the absence of the Raman effect) beam changes its intensity by 20%. Our starting conditions are a gaussian waist of 1 cm for the 1.06 beam and a lens of 30 cm focal length. We must first calculate the properties of the waist of the 1.06 microns in the absence of the Raman effects.

Given the initial transverse radius and the phase radius of curvature, we can find the waist radius and the distance to the waist. We begin from the well known properties of gaussian beams.

We can find transverse radius and the phase radius of curvature by,

$$\omega = \omega_0 \left[1 + \left(\frac{\lambda z}{\pi \omega_0^2} \right)^2 \right]^{1/2} \quad (19.1)$$

$$R = z \left[1 + \left(\frac{\pi \omega_0^2}{\lambda z} \right)^2 \right] \quad (19.2)$$

$$\frac{\pi \omega^2}{\lambda R} = \frac{\pi \omega_0^2 \left[1 + \left(\frac{\lambda z}{\pi \omega_0^2} \right)^2 \right]^{-1/2}}{\lambda z \left[1 + \left(\frac{\pi \omega_0^2}{\lambda z} \right)^2 \right]} = \frac{\lambda z}{\pi \omega_0^2} \quad (19.3)$$

$$\omega_0 = \omega \left[1 + \left(\frac{\pi \omega^2}{\lambda R} \right)^2 \right]^{-1/2} \quad (19.4)$$

$$z_\omega = R \left[1 + \left(\frac{\lambda R}{\pi \omega^2} \right)^2 \right]^{-1} \quad (19.5)$$

$$\omega_0 = 22 \text{ cm} \quad (19.6)$$

$$z_\omega = 29.999997 \text{ cm} \quad (19.7)$$

$$z_R = 0.0303652 \text{ cm} \quad (19.8)$$

where z_ω is the location of the waist from the lens and z_R is the Rayleigh distance.

From the equation for the gaussian radius, we can find the positions for each n step increase of intensity of 20%.

$$\left[\left(\frac{\omega}{\omega_0} \right)^2 - 1 \right]^{1/2} = \frac{z_\omega}{z_R} \quad (19.9)$$

$$z_n = z_R \sqrt{(1.2)^n - 1} \quad (19.10)$$

where z_n is the distance from the waist for the n th increase of intensity.

The `udata` feature, shown in Fig. 19.1 is used to build a plot of the energy in the tho beam through focus. Conversion is nearly complete and the ratio of the final energies are close to the ratio of the wavelengths as expected. Detailed examination of the photon count shows that there is an error of about 5 percent. This error is due primarily to the `rraman` command, which does not explicitly preserve photons.

Input: ex19.inp

```
c## ex19
c
c Example 19: Raman Process with Converging Beams: Simple Kinetics Step
c -----
c
c This example consists of calculation of Raman gain in a converging beam
c using simple kinetics steps.
c
c The configuration consists of a 1.06 micron pump beam and a 1.54 micron
c 1st Stokes beam. Both beams are brought to a focus by a 30 cm focal length
c lens. The initial radius of both beams is 1. cm.
c
c This example uses the ZONE command to keep the units identical for
c the two beams.
c
c ZONE/FIX Center Halfw
c
```

Jump to: [Commands](#), [Theory](#)

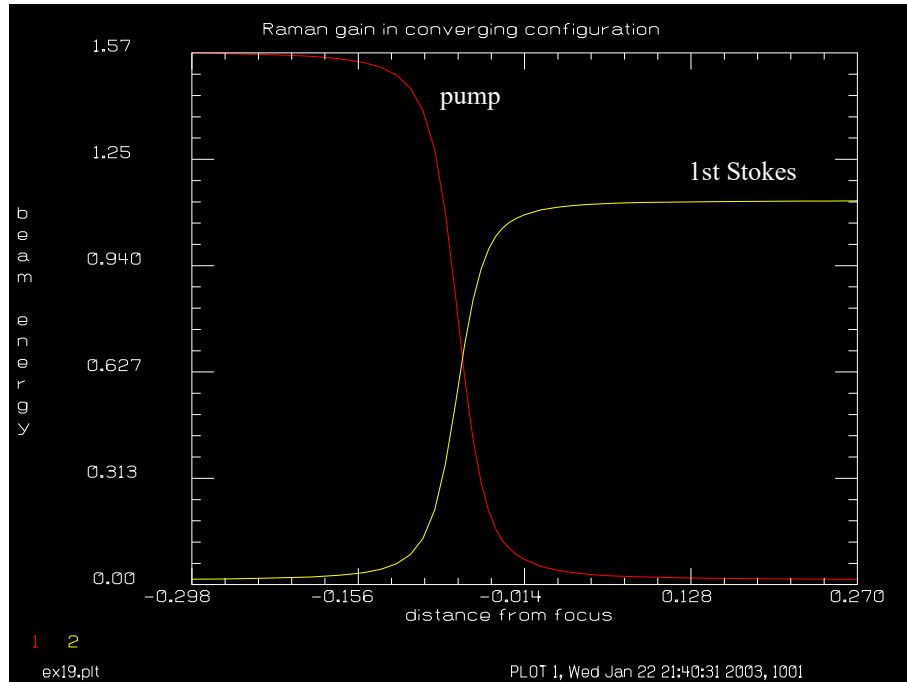


Fig. 19.1. Through-focus Raman conversion.

```

c   where Center is the center of the zone and Halfw is the halfwidth
c   of the zone. In general, place Center at the focal length of the
c   lens and make Halfw equal to the Rayleigh distance for the pump.
c
c   Given the initial waist radius (1 cm) and converging radius of curvature
c   (30 cm), the gaussian beam properties are found by
c
c   Parameters
c   -----
c   rad      = 30 cm      # initial phase radius of curvature
c   w        = 1 cm       # initial transverse gaussian radius
c   lambda   = 1.06E-4 cm # pump wavelength
c   w0       = gaussian waist of pump beam = .0010122
c   rad = 30
c   w = 1
c   lambda = 1.06e-4
c   w0 = w/sqrt(1 + (pi*w^2/(lambda*rad))^2) list
c   z      = distance from lens to gaussian waist of pump beam = 29.99997
c   z      = rad/(1 + (lambda*rad/(pi*w^2))^2) list
c   zray   = Rayleigh distance = .0303652
c   zray = pi*w0^2/lambda list
c
c   If the primary interaction occurs at the waist of the pump beam, then
c   the area far from the focus may be neglected. At a distance of 10
c   Rayleigh lengths from the focus the intensities are only .01 of
c   the values at the waist. Consequently the Raman calculation will begin
c   at .3 cm from the focus.
c
c   The diffraction step size is chosen so that the gaussian waist changes
c   by .1 at each step. This results in a .21 change in intensity per step
c
c   For each diffraction step, 4 kinetics steps will be taken.

```

Jump to: [Commands](#), [Theory](#)

```

c
c
c ##### Define macros
c
variab/dec/int n pass
macro/def ramstep/over
    n = n+1
    z1= zray*sqrt(1.1^(2*abs(n)) - 1)
    zstep = abs(z2 - z1)
    z2 = z1
    dz = zstep/4.
    pass = pass + 1                # Increment pass count
    raman/step/noprop 1 2 dz .0003 # Kinetics steps
    raman/step/noprop 1 2 dz .0003
    raman/step/noprop 1 2 dz .0003
    raman/step/noprop 1 2 dz .0003
    energy
    z = z + zstep                  # Increment z-coordinate
    variab/set energy1 1 energy    # Beam 1 energy saved
    variab/set energy2 2 energy    # Beam 2 energy saved
    udata/set pass z energy1 energy2
    prop zstep
macro/end
c
c ##### end of macro definitions
c
nbeam 2                          # Define two beams
wavelength/set 1 1.06            # Define wavelength of pump beam
wavelength/set 2 1.54            # Define wavelength of seed beam
array/set 0 16                   # Set arrays to 16x16
units/s 0 .35                    # Set units
gauss/cir/con 1 1.0 1.0          # Define pump distribution
gauss/cir/con 2 0.01 1.0         # Define seed distribution
lens 0 30.
zone/fix 30. zray                # Define waist zone
zone/in 1
zone/in 2
c
c Compute first point
c
n = -24
c zstep(23) = .02734679
z2 = zray*sqrt(1.1^(2*abs(n)) - 1) list
z = -z2
prop/zproj/abs [30 + z]
c
c Begin kinetics calculations.
c
pass = 1                          # Initialize pass counter
variab/set energy1 1 energy        # Beam 1 energy saved
variab/set energy2 2 energy        # Beam 2 energy saved
udata/set pass z energy1 energy2   # Store plot/udata info
c
c ##### step to waist
c
macro/run ramstep/47
c
c Set up plot of energy of beams
c

```

Jump to: [Commands](#), [Theory](#)


```
udata/xlabel distance from focus
udata/ylabel beam energy
title Raman gain in converging configuration
udata/list                                # list udata information
pause 2
plot/watch ex19.plt
plot/udata 1 2 min=0.
end
```


Ex20: Raman amplification with wave4: collimated beams

This example and Examples 21, 22, and 23 illustrate the use of wave4 to model both four-wave mixing effects and Raman amplification. Example 20 performs the Raman calculation for collimated light with no gain for four-wave mixing. Note that the 2nd Stokes does not achieve significant values until about 800 cm of propagation.

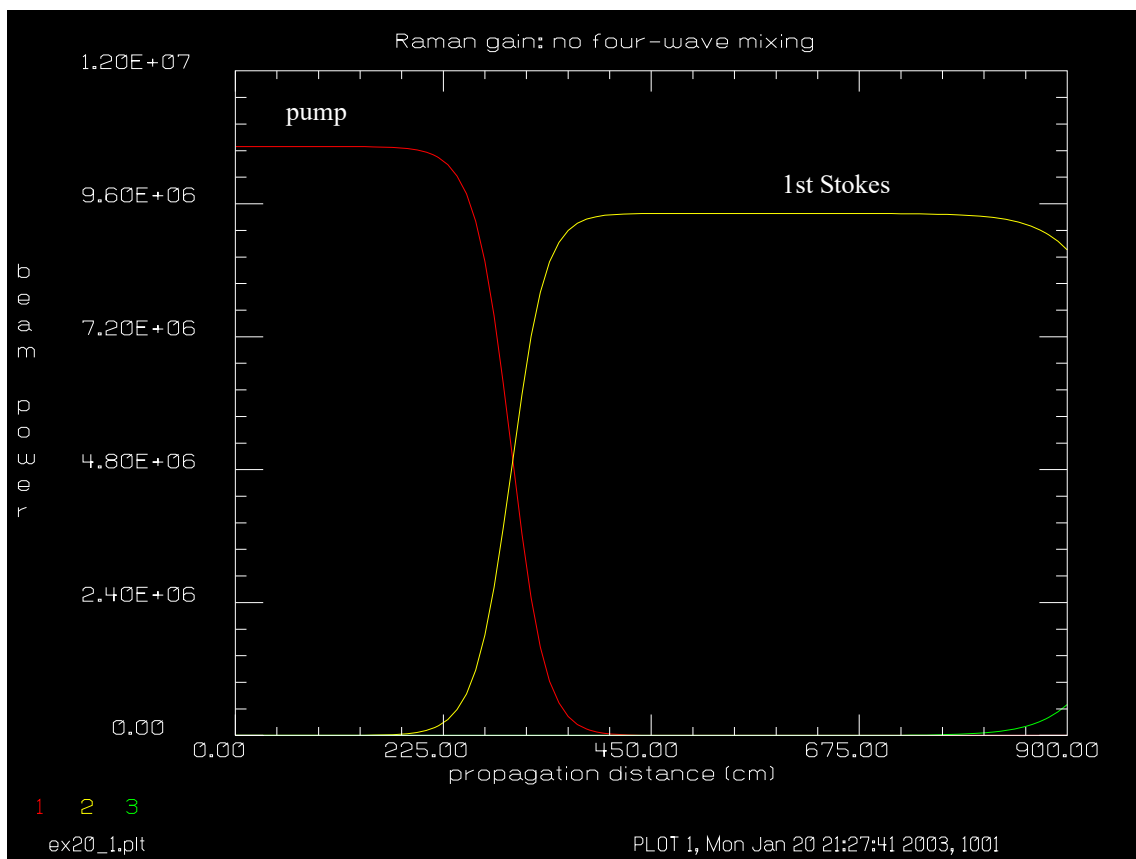


Fig. 20.1. Multiple Stokes Raman amplification: no four-wave mixing.

Input: `ex20.inp`

```
c## ex20
c
c Example 20: Raman Amplification with "WAVE4": Collimated Beams,
c           Geometric Propagation
c -----
c
c This example illustrates Raman amplification for pump and
c 1st and 2nd Stokes with no four-wave mixing.
c
nbeam 3                                # Define three beams
wavelength/set 1 .353                  # Define wavelength of pump beam
wavelength/set 2 .414                  # Define wavelength of 1st Stokes
wavelength/set 3 .499                  # Define wavelength of 2nd Stokes
array/set 0 16                         # Set arrays to 16x16
units/field 0 .75                      # Set units
```

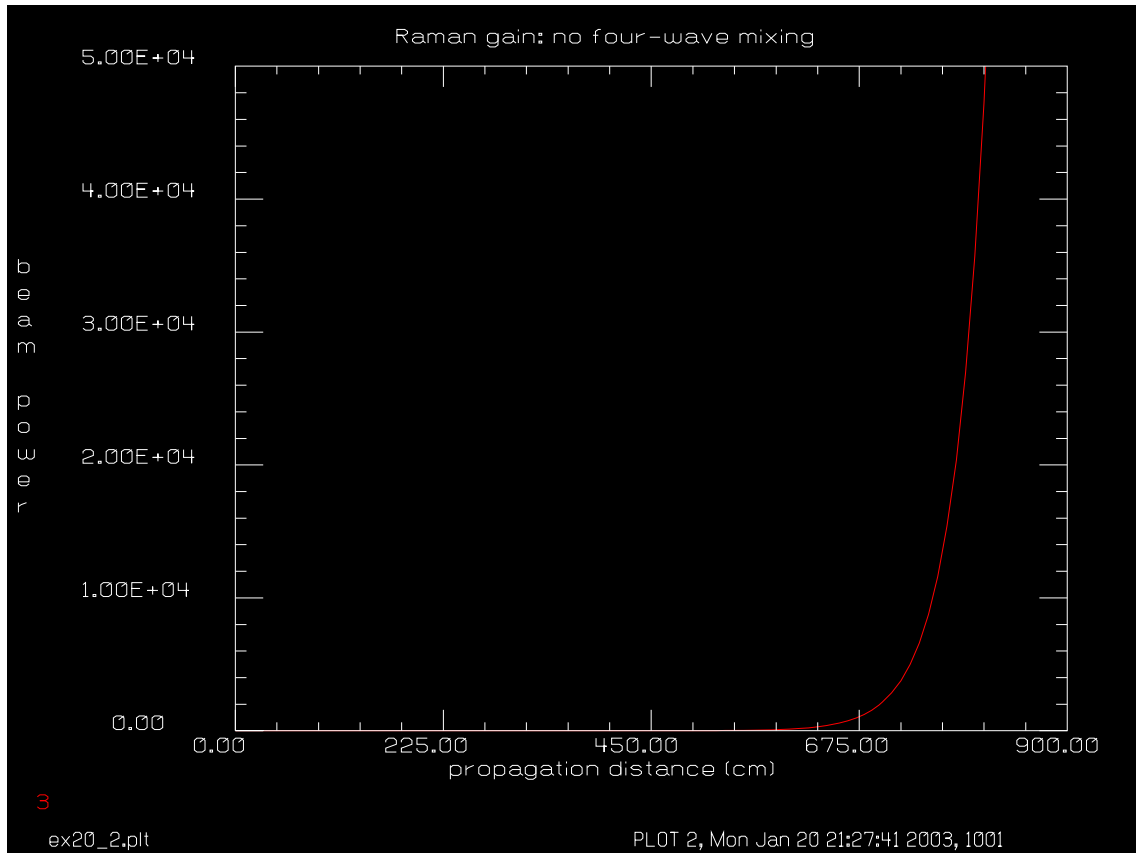


Fig. 20.2. 2nd Stokes growth plotted with expanded scale.

```

clear 1 1e7                                # Define pump intensity
clear 2 2.5                                # Define 1st Stokes intensity
clear 3 .025                               # Define 2nd Stokes intensity
clap/sqr/con 0 .5                          # Square aperture
udata/clear                                # Clear user data variables
wave4/set 0.                               # Set zstart for four-wave mixing
pack/set 1 2 3                             # Define arrays for memory processing
variab/set energy1 1 energy                # Define Param 1 = Energy of Beam 1
variab/set energy2 2 energy                # Define Param 2 = Energy of Beam 2
variab/set energy3 3 energy                # Define Param 3 = Energy of Beam 3
udata/set 1 0. energy1 energy2 energy3     # First call to user data
zstep = 10.                                # Set step length
z = 0.                                     # Initialize z-axis
pass = 1                                   # Initialize udata index
c
c Define macro for kinetic step
c
macro/def kinstep/over
  pack/in
    wave4/kinet zstep=zstep g4w=0. gps=5.16e-9 gst=3.25e-9 nstep2=10
  pack/out
  pass = pass + 1
  z = z + zstep
  variab/set energy1 1 energy
  variab/set energy2 2 energy
  variab/set energy3 3 energy
  udata/set pass z energy1 energy2 energy3

```

Jump to: [Commands](#), [Theory](#)

```
macro/end
c
c  Run calculations
c
time/init          # Initialize time step
macro/run kinstep/90 # Propagate 900 cm in 90 steps
time/show
c
c  Set up plot titles
c
udata/xlab propagation distance (cm)
udata/ylab beam power
title Raman gain: no four-wave mixing
udata/list
plot/watch ex20_1.plt
plot/udata 1 3 min=0 max=1.2e7
plot/watch ex20_2.plt
plot/udata 3 3 min=0 max=5e4
end
```


Ex21: Four-wave mixing with wave4: collimated, geometric propagation

This example shows four-wave mixing without Raman amplification. This is done by setting the Raman gain coefficients to zero in the command, wave4. The effective gain is influenced by the phase matching coefficient. In Fig. 21.1, the phase matching factor is $\Delta k = 0$, with the tilt angle set to $\theta = 0.002853$. The modulation is a maximum for this case. In Fig. 21.2, which detunes the phase matching factor. The modulation is decreased and the period of modulation is shortened. In Fig. 21.3, the beams are colinear. This case is the same as Examples 20 and 22. Figure 21.4 shows the colinear case at expanded scale to show the modulation of the 2nd Stokes.

Note that in Figs. 21.2 and 21.3, there is a slight increase in the average energy in the pump beam. This is a numerical error and may be reduced by setting nstep2 to larger values. .

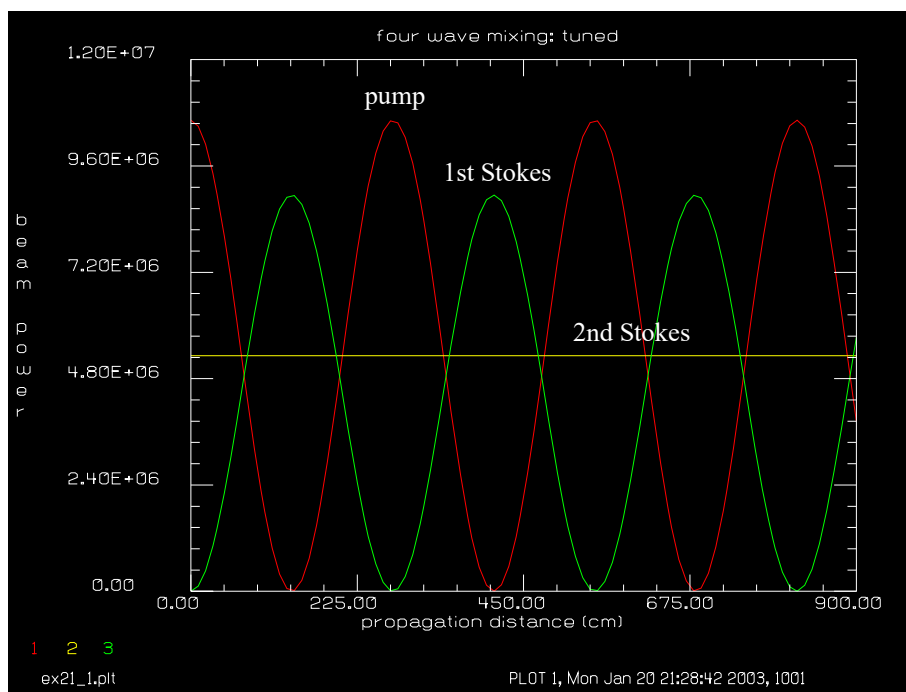


Fig. 21.1. Four-wave mixing with perfect phase matching at $\theta = 0.002853$. Pump (red), 1st Stokes (green), 2nd Stokes (yellow).

Input: ex21.inp

```
c## ex21
c
c Example 21: Four-Wave Mixing "WAVE4": Collimated Beams,
c           Geometric Propagation
c -----
c
c This example illustrates four-wave mixing with no Raman amplification.
c The four-wave mixing oscillates. Because the beams are colinear,
c the process is significantly detuned by the phase-matching factor.
c
variab/dec/int pass
nbeam 3                                # Define three beams
wavelength/set 1 .353                  # Define wavelength of pump beam
```

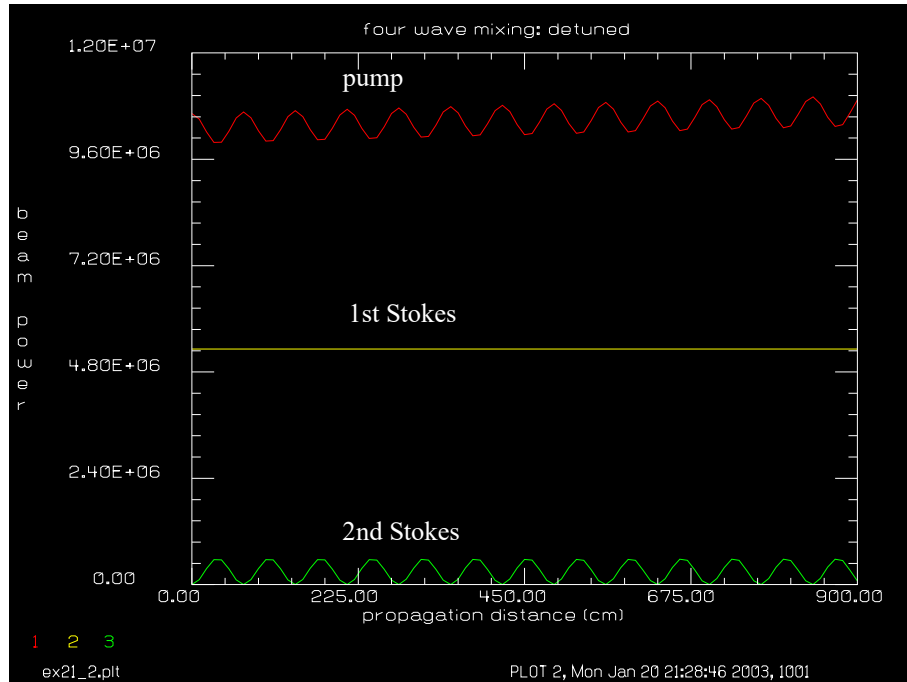


Fig. 21.2. Four-wave mixing with the 2nd Stokes at $t \theta = 0.0026$.

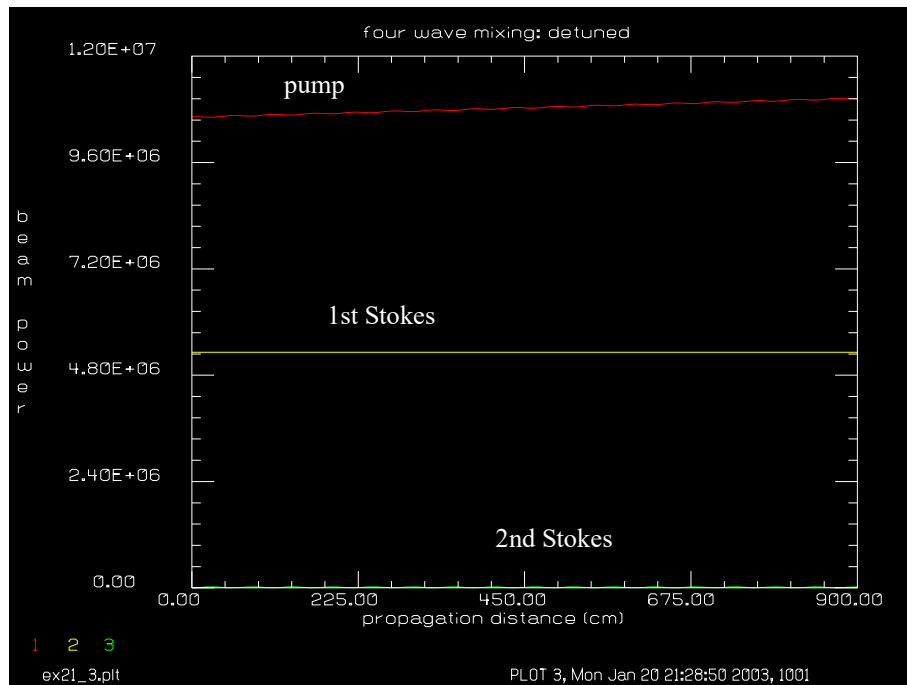


Fig. 21.3. Four-wave mixing with collinear beams.

```
wavelength/set 2 .414          # Define wavelength of 1st Stokes
wavelength/set 3 .499          # Define wavelength of 2nd Stokes
array/set 0 16                 # Set arrays to 16x16
units/field 0 .75              # Set units
clear 1 1e7                    # Define pump intensity
clear 2 .5e7                   # Define 1st Stokes intensity
```

Jump to: [Commands](#), [Theory](#)

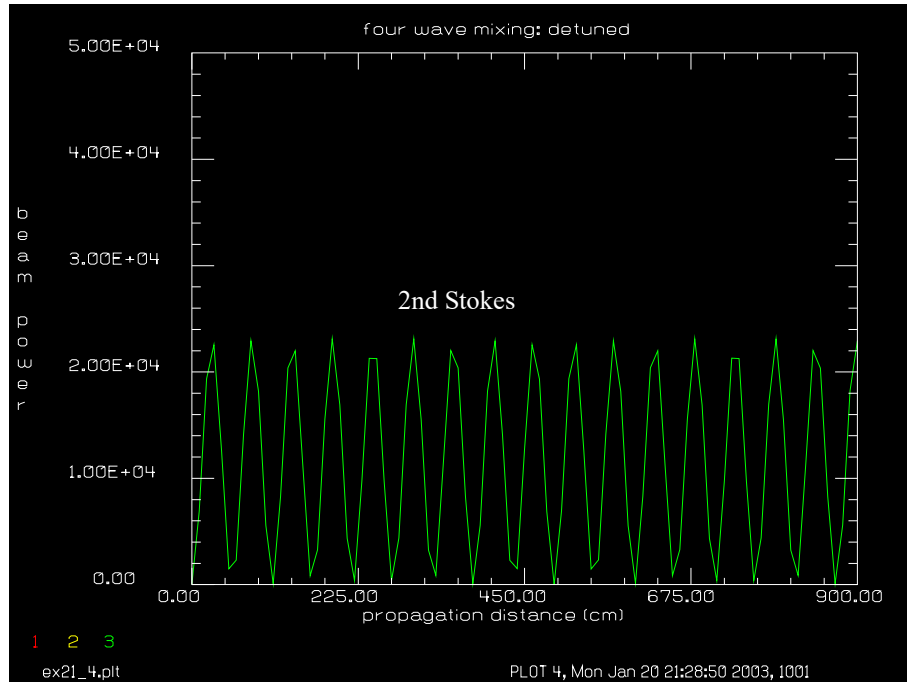


Fig. 21.4. Four-wave mixing with colinear beams (expanded scale).

```

clear 3 .025                                # Define 2nd Stokes intensity
clap/sqr/con 0 .5                            # Square aperture
tilt 3 2.853                                # Tilt for deltak=.002853 mr
                                              # Also run with, .0026 and 0.0

udata/clear                                  # Clear user data variables
wave4/set 0.                                # Set zstart for four-wave mixing
pack/set 1 2 3                              # Define arrays for memory processing
variab/set energy1 1 energy                 # Define Param 1 = Energy of Beam 1
variab/set energy2 2 energy                 # Define Param 2 = Energy of Beam 2
variab/set energy3 3 energy                 # Define Param 3 = Energy of Beam 3
udata/set 1 0. energy1 energy2 energy3      # First call to user data
zstep = 10.                                # Set kinetics step to 20 cm.
z = 0.                                       # Initialize z-axis
pass = 1                                    # Initialize udata index
c
c Define macro for kinetic step
c
macro/def kinstep/over
  pack/in
    wave4/kinet zstep=zstep g4w=4.22e-9 nstep2=10
  pack/out
    pass = pass + 1
    z = z + zstep
    variab/set energy1 1 energy
    variab/set energy2 2 energy
    variab/set energy3 3 energy
    udata/set pass z energy1 energy2 energy3
macro/end
c
c Run calculations
c
macro/run kinstep/90                        # Propagate 900 cm in 90 steps

```

Jump to: [Commands](#), [Theory](#)

```

c
c  Set up plot titles
c
udata/xlab propagation distance (cm)
udata/ylab beam power
title four wave mixing: tuned
plot/watch ex21_1.plt
plot/udata 1 3 min=0 max=1.2e7
clear 1 1e7                                # Define pump intensity
clear 2 .5e7                                # Define 1st Stokes intensity
clear 3 .025                                # Define 2nd Stokes intensity
clap/sqr/con 0 .5                           # Square aperture
zreff = 0.
z = 0.
tilt 3 2.6-2.853                             # Tilt for deltak=.0026 and 0.0
pass = 1
macro/run kinstep/90                         # Propagate 900 cm in 90 steps
plot/watch ex21_2.plt
title four wave mixing: detuned
plot/udata 1 3 min=0 max=1.2e7
clear 1 1e7                                # Define pump intensity
clear 2 .5e7                                # Define 1st Stokes intensity
clear 3 .025                                # Define 2nd Stokes intensity
clap/sqr/con 0 .5                           # Square aperture
zreff = 0.
z = 0.
tilt 3 -2.6                                 # Tilt for deltak=0.0
pass = 1
macro/run kinstep/90                         # Propagate 900 cm in 90 steps
plot/watch ex21_3.plt
plot/udata 1 3 min=0 max=1.2e7
plot/watch ex21_4.plt
plot/udata 1 3 min=0 max=5e4
end

```

Ex22: Raman gain and four-wave mixing, collimated

This example shows the combined effect of four-wave mixing and Raman amplification. The four-wave mixing seeds the Raman amplification and 2nd Stokes shows up much more quickly than was the case in Example 20.

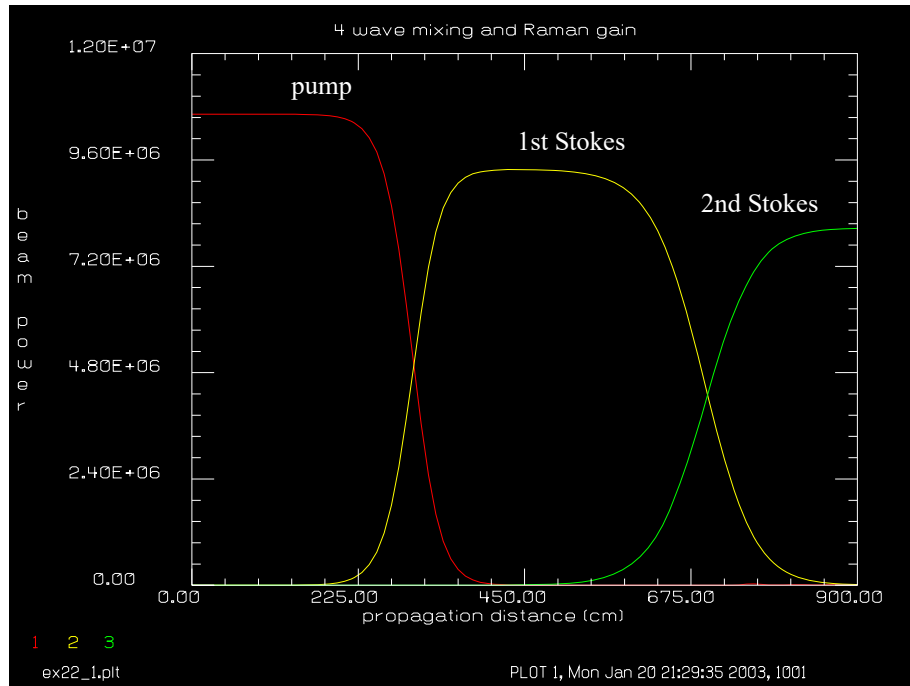


Fig. 22.1. Multiple Stokes gain with four-wave mixing.

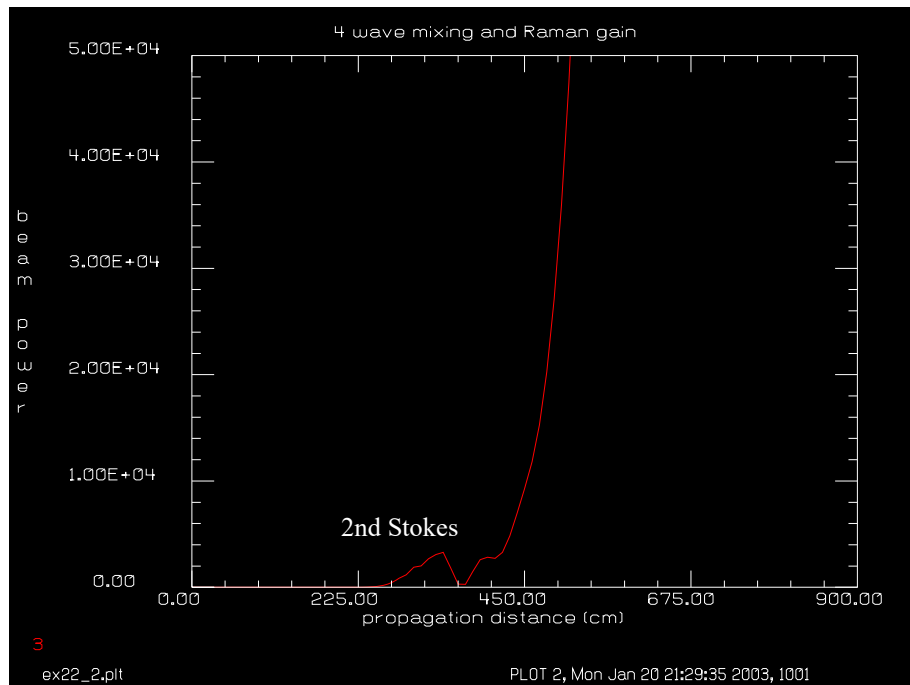


Fig. 22.2. Energy plot with expanded scale.

Input: ex22.inp

```

c## ex22
c
c Example 22: Raman Gain and Four-Wave Mixing, Collimated
c
c This example illustrates four-wave mixing with Raman amplification.
c
array/set 1 16 # Set array to 16x16
nbeam 3 # Define two more beams of the same
# size
wavelength/set 1 .353 # Define wavelength of pump beam
wavelength/set 2 .414 # Define wavelength of 1st Stokes
wavelength/set 3 .499 # Define wavelength of 2nd Stokes
units/field 0 .75 # Set units
clear 1 1e7 # Define pump intensity
clear 2 2.5 # Define 1st Stokes intensity
clear 3 .025 # Define 2nd Stokes intensity
clap/sqr/con 0 .5 # Square aperture
c *** tilt 3 2.853 # Tilt for deltak=0, not used.
udata/clear # Clear user data variables
wave4/set 0. # Set zstart=0 for four-wave mixing
pack/set 1 2 3 # Define arrays for memory processing
variab/set energy1 1 energy # Define Param 1 = Energy of Beam 1
variab/set energy2 2 energy # Define Param 2 = Energy of Beam 2
variab/set energy3 3 energy # Define Param 3 = Energy of Beam 3
udata/set 1 0. energy1 energy2 energy3 # First call to user data
zstep = 10. # Set step length
z = 0. # Initialize z-axis
pass = 1 # Initialize udata index
c
c Define macro for kinetic step
c
macro/def kinstep/over
  pack/in
  wave4/kinet zstep=zstep g4w=4.22e-9 gps=5.16e-9 gst=3.25e-9 nstep2=10
  pack/out
  pass = pass + 1
  z = z + zstep
  variab/set energy1 1 energy
  variab/set energy2 2 energy
  variab/set energy3 3 energy
  udata/set pass z energy1 energy2 energy3
macro/end
c
c Run calculations
c
time/init # Initialize time step
macro/run kinstep/90 # Propagate 900 cm in 90 steps
time/show
c
c Set up plot titles
c
udata/xlab propagation distance (cm)
udata/ylab beam power
title 4 wave mixing and Raman gain
udata/list
plot/watch ex22_1.plt
plot/udata 1 3 min=0 max=1.2e7

```

Jump to: [Commands](#), [Theory](#)

```
plot/watch ex22_2.plt
plot/udata 3 3 min=0 max=5e4
end
```


Ex23: Four-wave mixing wave4, converging beam

This example shows the effects of four-wave mixing and Raman amplification with the 2nd Stokes beam made to be converging. The converging beam is specified with a radius of curvature of 105.3 cm, such that at the 0.3 cm aperture zone the phase matching condition is achieved. Consequently the $r = 0.3$ cm zone has the most significant growth of four-wave mixing. Figure 23.1 shows the initial intensity distribution of the 2nd Stokes beam. Figure 23.2 shows the 2nd Stokes intensity after a 5 cm propagation. A ring located approximately at the $r = 0.3$ cm zone is evident. Also, there are secondary rings evident in the outer zones. This calculation requires interpolation between the pump and 1st Stokes beams which are the same size and the 2nd Stokes which is decreasing in size as it propagates toward the 105.2 cm focus. .

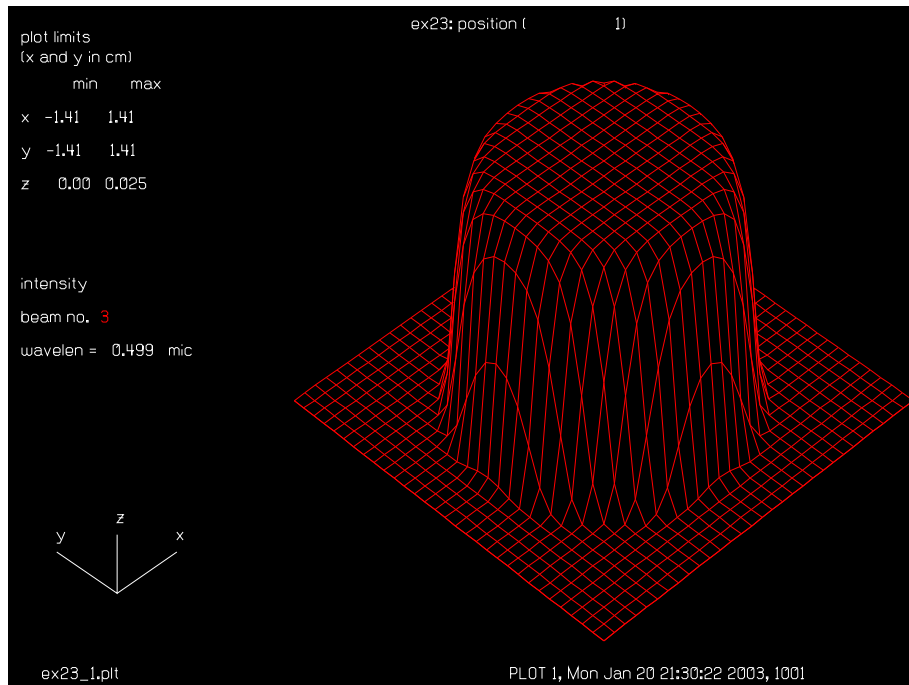


Fig. 23.1. Initial 2nd Stokes distribution.

Input: ex23.inp

```
c## ex23!292299492932935
c
c Example 23: Four-Wave Mixing "WAVE4", Converging Beam
c -----
c
c This example illustrates four-wave mixing with Raman amplification.
c The beam is converging so that the phase matching factor is zero at
c aperture zone  $r=.3$ .
c
c variab/dec/int pass
c nbeam 3                                # Define three beams
c wavelength/set 1 .353                  # Define wavelength of pump beam
c wavelength/set 2 .414                  # Define wavelength of 1st Stokes
c wavelength/set 3 .499                  # Define wavelength of 2nd Stokes
c array/set 0 64                         # Set arrays to 64x64
```

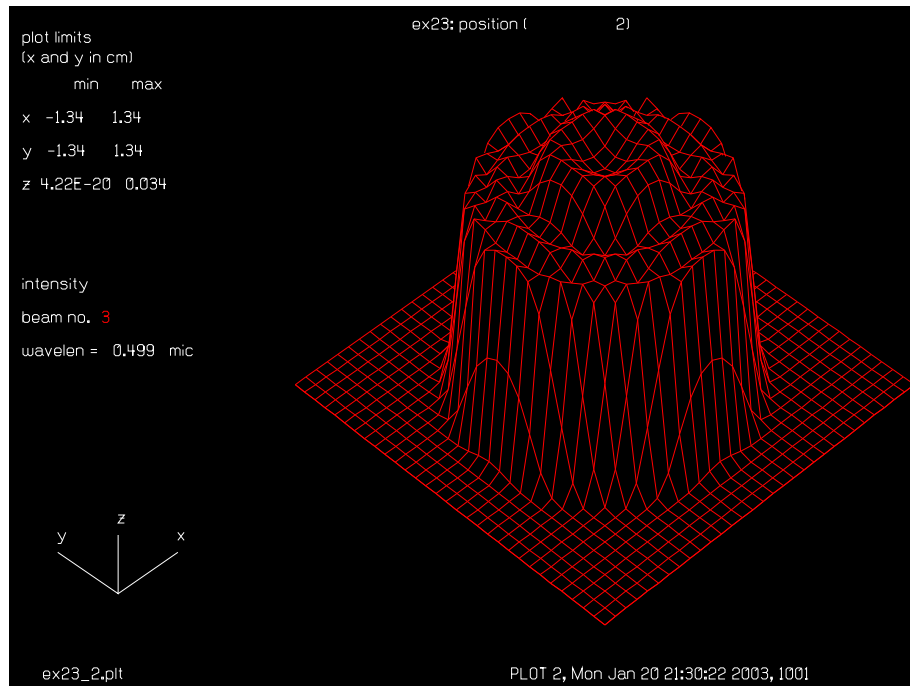


Fig. 23.2. 2nd Stokes distribution after 15 cm propagation.

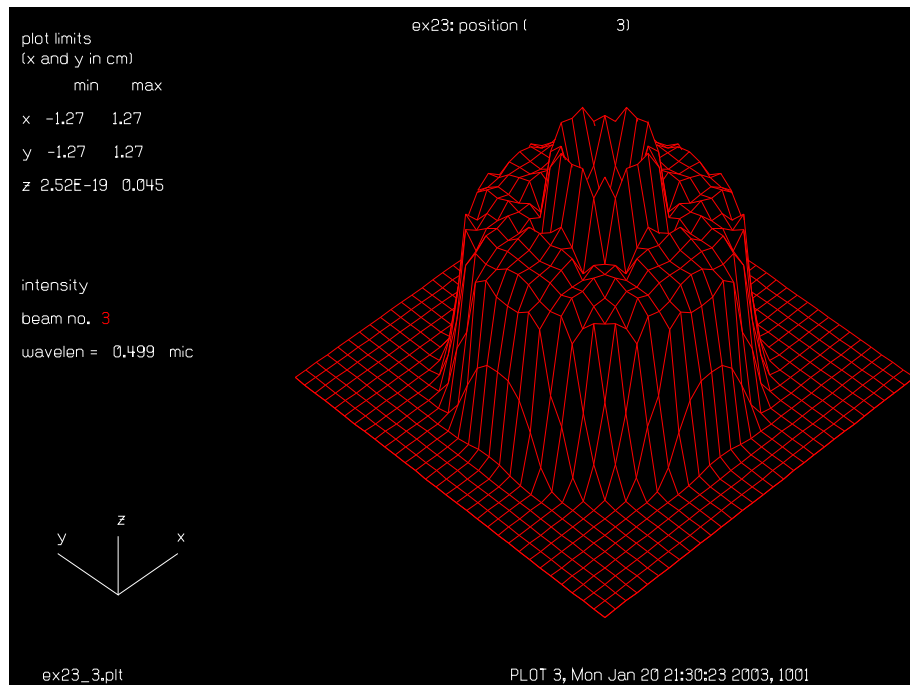


Fig. 23.3. 2nd Stokes distribution after 10 cm propagation.

```

units/field 0 1.5
gauss/cir/con 1 1e7 1. 8
gauss/cir/con 2 5e2 1. 8
gauss/cir/con 3 .025 1. 8
udata/clear
wave4/set 0.
# Set units
# Define pump intensity
# Define 1st Stokes intensity
# Define 2nd Stokes intensity
# Clear user data variables
# Set zstart=0 for four-wave mixing

```

Jump to: [Commands](#), [Theory](#)

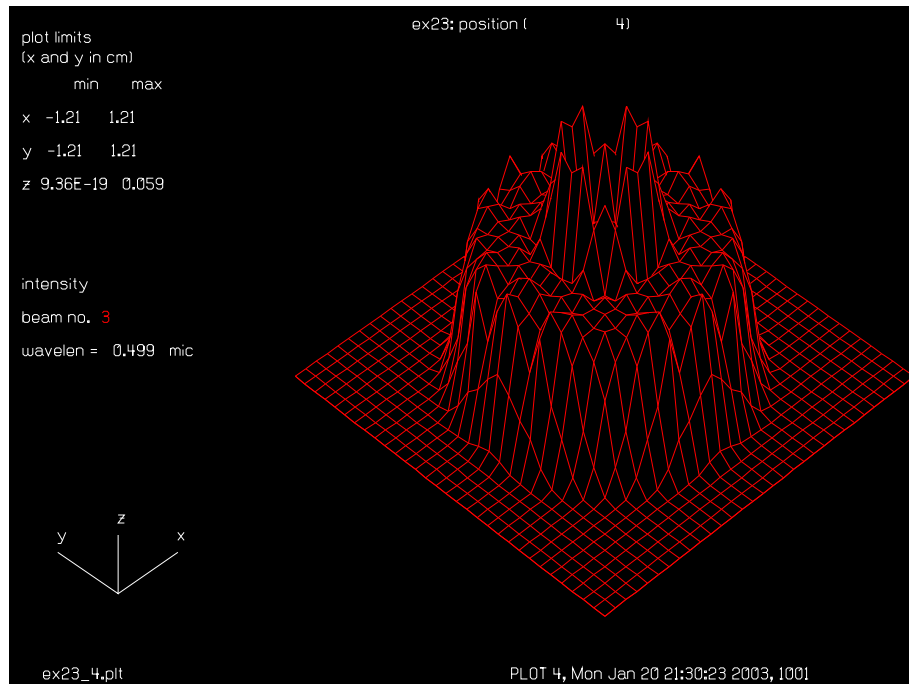


Fig. 23.4. 2nd Stokes distribution after 15 cm propagation.

```

pack/set 1 2 3                                     # Define arrays for memory processing
variab/set energy1 1 energy                         # Define Param 1 = Energy of Beam 1
variab/set energy2 2 energy                         # Define Param 2 = Energy of Beam 2
variab/set energy3 3 energy                         # Define Param 3 = Energy of Beam 3
udata/set 1 0. energy1 energy2 energy3             # First call to user data
zstep = 5.                                          # Set step length
z = 0.                                             # Initialize z-axis variable
pass = 1                                           # Initialize udata index
set/density 32
c
c Define macro for kinetic step
c
macro/def kinstep/over
  dist zstep
  pack/in
    wave4/kinet zstep=zstep g4w=4.22e-9 gps=5.16e-9 gst=3.25e-9 nstep2=10
  pack/out
  pass = pass + 1
  z = z + zstep
  variab/set energy1 1 energy
  variab/set energy2 2 energy
  variab/set energy3 3 energy
  udata/set pass z energy1 energy2 energy3
  plot/watch ex23_@pass.plt
  plot first=3 last=3
macro/end
c
c Run calculations
c
title ex23: position @pass
plot/watch ex23_1.plt
plot first=3 last=3
lens 3 105.2                                     # lens for 2nd Stokes beam

```

Jump to: [Commands](#), [Theory](#)

```
macro/run kinstep/3          # Propagate 15 cm in 3 steps
end
```

Ex24: Atmospheric aberration and adaptive optics

Table. 24.1. Table of Ex24 examples

Ex24a: Atmospheric turbulence	1
Ex24b: Atmospheric aberration in a collimated path	2
Ex24c: Atmospheric aberration in a converging path	4

This example shows the use of the atmospheric and adaptive optics models. The atmospheric model assumes the power spectrum of the wavefront is

$$W^2(f) = \frac{0.023 e^{-f^2 L_i^2}}{r_0^{5/3} \left(f^2 + \frac{1}{L_o^2} \right)^{11/3}}, \quad (24.1)$$

where $W^2(f)$ is the power spectrum of the wavefront, r_0 is the seeing parameter, f is the spatial frequency, L_o is the outer scale, and L_i is the inner scale. These parameters are in radians, meters, and inverse meters respectively. Examples of collimated and converging beams are illustrated in Ex24b.inp and Ex24c.inp.

The adaptive model assumes that all actuators are identical and equally spaced in a square grid. The user may specify the width of the actuator influence function. In this example the influence function width is 3 cm. The initial wavefront is shown in Fig. 24.1 and the corrected wavefront is shown in Fig. 24.2. The Strehl ratio is improved from 0.04 to 0.87 by the adaptive optic. The residual error consists of high frequency components and errors localized at the edge of the aperture due to the edge discontinuities.

Ex24a: Atmospheric turbulence

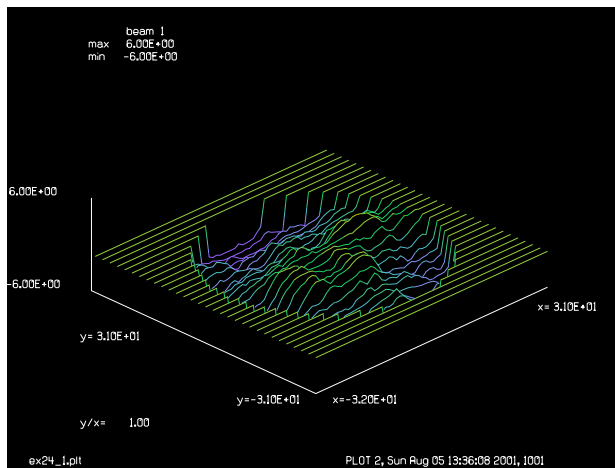


Fig. 24.1. Wavefront before correction.

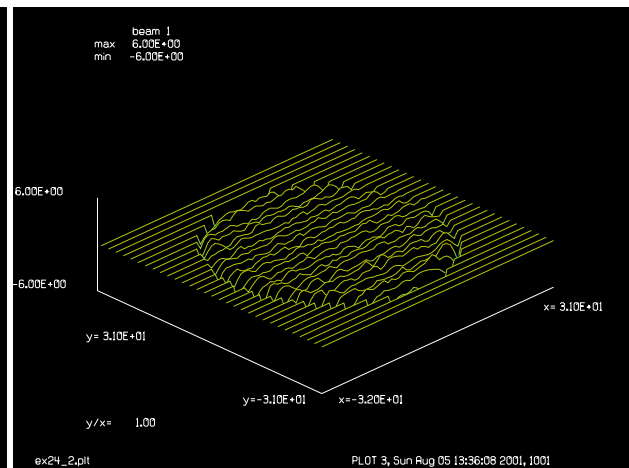


Fig. 24.2. Wavefront after correction.

Input: **ex24a.inp**

c## ex24a!816655357930262

```

c
c Example 24a: Atmospheric Turbulence
c
c This is an example of the modeling atmospheric aberration and the
c adaptive optic model. The atmosphere is modeled with  $r_0 = 10$ . cm. The
c adaptive optic has an impulse response width of 3 cm. The resulting
c wavefront has the Strehl ratio improved from .09 to .97.
c
units 1 1                                # Select units to be 1. cm.
clap/c/c 1 25.                            # Clear aperture 50. cm.
plot/1
phase/random/kolmogorov 1 10.              # Create atmospheric aberration
strehl                                     # Calculate Strehl ratio.
pause
plot/watch ex24_1.plt
plot/liso/phase max=6. min=-6.             # Plot phase.
adapt 1 3.                                # Adaptive mirror, 3 cm. actuator spacing
plot/watch ex24_2.plt
plot/liso/phase max=6. min=-6.             # Plot phase.
strehl                                     # Calculate Strehl ratio.
pause
end

```

Ex24b: Atmospheric aberration in a collimated path

Input: `ex24b.inp`

```

c## ex24b
#
# Atmospheric aberration in a collimated air path
#
# Parameters
# -----
# beam diameter, 4 inches
# wavelength range, 1 micron to 2 microns
# collimated beam
# propagation length, 10 km
# atmospheric seeing,  $r_0 = 5$  cm over 1 km distance
#
# The beam is assumed to be aberration-free and collimated.
# The 10 km path is broken up into ten 1 km sections.
# The seeing of each of the full 10 km path is
#
#  $r = r_0 / N^{(3/5)} = 5 / 10^{.6} = 1.256$  cm
#
# where r is the seeing of the seeing of the total path
# and N = 10 is the number of subsections.
#
# The  $r_0 = 5$  cm applies to  $\lambda = 1$ . micron.
#
# Assume a frozen atmosphere, although the
# atmosphere could be temporally sheared to represent

```

Jump to: [Commands](#), [Theory](#)

```

# wind or beam steering.
# Note that the aberration at the beginning of the
# propagation path has the most effect on beam breakup.
# The primary effect at the 10 km distance is high
# spatial frequency beam breakup. Longer propagation
# will start to show lower spatial frequency effects
# also.
#
# The wavelength range is broken up into five wavelength
# samples: 1.00, 1.25, 1.50, 1.75, 2.00
# We want each wavelength to see the same atmospheric turbulence
# but the effect on each particular wavelength is scaled
# by the wavelength
#
# phase aberration =  $2\pi \cdot dN \cdot L / \lambda$ 
#
# where dN is the index disturbance, L is the length and
# lambda is the wavelength.
#
# Beams used
# -----
# Beam 1      Propagating beam
# Beam 2      phase screen
# Beam 3      Incoherent sum of irradiance due to wavelength
#              samples
#
variab/dec/int count_axis count_lambda max_axis max_lambda
max_axis = 10
max_lambda = 5
lambda_start = 1.0          # initial wavelength
lambda_step = .25           # increment wavelength
count_lambda = 0
r0 = 5                      # Fried's seeing parameter
                           # for 1 km distance
delta_z = 1e5               # axial step size (cm)
count_axis = 0
x=srand(11)                 # set random number generator
                           # change argument to get
                           # different screens
                           # half-width of array
Field = 20
macro/def make_screens/overwrite
# makes the phase screens for later use
    count_axis = count_axis + 1
    clear 2 1                # initialize beam 2
    units/field 2 Field      # choose sampling to cover 10 x 10 cm
    phase/kol 2 r0           # make phase screen
# form file name: screen + @count_axis (integer variable)
# write out phase screen, no header, unformatted, array 2
    outfile/beam screen@count_axis.dat/no/unf 2
    title wavefront @count_axis
    plot/w ex24b_1.plt
    plot/l/wave 2
macro/end

```

Jump to: [Commands](#), [Theory](#)

```

macro/def path/overwrite
# propagate along optical path
count_axis = count_axis + 1
infile/beam screen@count_axis.dat/no/unf 2 # read in phase screen
mult/phase 2 [lambda_start/lambda] # scale for wavelength
mult/beam 1 2 # invoke phase screen
prop delta_z # diffraction propagation
title step: lambda = @count_lambda, axis step = @count_axis
plot/1 1 xrad=10 ns=64 # let's look at irradiance
macro/end

macro/def lambda/overwrite
count_lambda = count_lambda + 1
lambda = (count_lambda-1)*lambda_step + lambda_start
count_axis = 0
units/field 1 Field
gaus/c/c 1 1 5.08 # set gaussian beam to 4" dia.
wavelength 1 [lambda*.0001]
#
# put aperture here if we have one
# put beam expander aberration here if we have any
#
macro/run path/max_axis # propagate to target
add/inc/con 3 1 # sum over wavelengths
macro/end
#
# main work starts here
#
array/set 1 256 # set array size to 256 x 256
nbeam 3 data # make two more data arrays, same size
clear 3 0. # clear array 3 for forming sum
units/field 0 Field
wavelength 0 lambda_start
macro/run make_screens/max_axis # make phase screens
count_lambda = 0
macro/run lambda/max_lambda # scan over wavelengths
title sum over wavelengths
plot/1 3 xrad=10 ns=64
plot/w ex24b_2.plt
plot/x/i 3 left=-10 right=10

```

Ex24c: Atmospheric aberration in a converging path

Input: ex24c.inp

```

c## ex24c!052477178026991
#
# Atmospheric aberration in a converging path
#
# Parameters
# -----
# beam diameter, 4 inches
# wavelength range, 1 micron to 2 microns
# propagation length, 1 km

```

Jump to: [Commands](#), [Theory](#)

```

# beam converges at 1 km distance
# atmospheric seeing, r0 = 5 cm over 1 km distance
#
# The beam is assumed to be aberration-free and collimated.
# The 1 km path is broken up into four .25 km sections.
# The seeing of each .25 km of the path is
#
#  $r = r_0 * N^{(3/5)} = 5 * N^{.6} = 11.49$  cm
#
# where r is the seeing of the subsections and N = 4
# is the number of subsections.
#
# It is assumed that r0 = 5 cm applies to lambda = 1. micron.
#
# assume a frozen atmosphere, although the
# atmosphere could be temporally sheared to represent
# wind or beam steering.
#
# For this converging beam, the maximum irradiance disturbance
# occurs at about 0.5 km. The far-field shows mostly beam
# tilt.
#
# The wavelength range is broken up into five wavelength
# samples: 1.00, 1.25, 1.50, 1.75, 2.00
# We want each wavelength to see the same atmospheric turbulence
# but the effect on each particular wavelength is scaled
# by the wavelength
#
# phase aberration =  $2\pi * dN * L / \lambda$ 
#
# where dN is the index disturbance, L is the length and
# lambda is the wavelength.
#
# Beams used
# -----
# Beam 1      Propagating beam
# Beam 2      phase screen
# Beam 3      Incoherent sum of irradiance due to wavelength
#             samples
variab/dec/int count_axis count_lambda max_axis max_lambda
max_axis = 4
max_lambda = 5
lambda_start = 1.0          # initial wavelength
lambda_step = .25           # increment wavelength
count_lambda = 0
r0 = 5.                      # Fried's seeing parameter
                           # for 1 km distance
r = r0 * max_axis^(3/5)     # seeing of the subsections
total_z = 1e5               # 1 km
delta_z = total_z/max_axis  # axial step size (cm)
count_axis = 0
x=srand(11)                 # set random number generator
                           # change argument to get
                           # different screens
Field = 50                  # half-width of array

rad      = 1e5              # initial phase radius of curvature
w        = 5.08             # initial transverse gaussian radius

```

Jump to: [Commands](#), [Theory](#)

```

lambda_cm      = 2e-4      # pump wavelength (cm)
w0 = w/sqrt(1 + (pi*w^2/(lambda_cm*rad))^2) # far field radius
zray = pi*w0^2/lambda_cm list # Rayleigh range
variab
macro/def make_screens/overwrite
# makes the phase screens for later use
    count_axis = count_axis + 1
    clear 1 1          # initialize beam 1
    phase/kol 1 r      # make phase screen
# form file name: screen + @count_axis (integer variable)
# write out phase screen, no header, unformatted, array 2
    outfile/beam screen@count_axis.dat/no/unf 1
    title wavefront @count_axis
    plot/1/wave 1
    prop delta_z      # propaate to get right units
macro/end

macro/def path/overwrite
# propagate along optical path
    count_axis = count_axis + 1
    infile/beam screen@count_axis.dat/no/unf 2 # read in phase screen
    mult/phase 2 lambda_start/lambda_micron    # scale for wavelength
    mult/beam 1 2                               # invoke phase screen
    prop delta_z                                # diffraction propagation
    title step: lambda = @count_lambda, axis step = @count_axis
    plot/1 1 xrad=10 ns=64                     # let's look at irradiance
macro/end

macro/def lambda/overwrite
    count_lambda = count_lambda + 1
    lambda_micron = (count_lambda-1)*lambda_step + lambda_start
    count_axis = 0
    units/field 1 Field
    gaus/c/c 1 1 5.08          # set gaussian beam to 4" dia.
    energy/norm 1 1
    wavelength/set 1 lambda_micron
    zreff/se 1 0
    lens 1 1e5                # lens focal length
    zone/fix 1e5 zray          # define propagation zone
    zone/in 1                  # implement zone control
    pause 3
#
# put aperture here if we have one
# put beam expander aberration here if we have any
#
    macro/run path/max_axis      # propagate to target
    add/inc/con 3 1              # sum over wavelengths
    plot/w plot2.plt
    title step: lambda = @count_lambda, axis step = @count_axis
    plot/1 3 xrad=4 ns=128
    plot/w plot1.plt            # return to plot1.plt
    zone/out 1
    zone/unfix
macro/end
#
# main work starts here
#
array/set 1 256                # set array size to 256 x 256
nbeam 3 data                    # make two more data arrays, same size

```

Jump to: [Commands](#), [Theory](#)


```

units/field 0 Field
wavelength/set 0 lambda_start    # lambda in microns
lens 1 1e5                        # lens focal length
zone/fix 1e5 zray                # define propagation zone
zone/in 1                        # implement zone control
pause 3
macro/run make_screens/max_axis   # make phase screens
# done with making phase screens
# now set up units of array 3 to match array 1 in far-field
copy 1 3                          # copy 1 to 3 to set units in 3
clear 3 0.                        # clear array 3 for forming sum
zone/out 1
zone/unfix
count_lambda = 0
macro/run lambda/max_lambda       # scan over wavelengths
title sum over wavelengths
plot/w plot2.plt
plot/l 3 xrad=4 ns=64
plot/w plot3.plt
plot/x/i 3 left=-4 right=4

```


Ex25: Ground-to-space laser communication system

This example shows the propagation of a laser beam from a ground-based laser to a relay mirror in low earth orbit. The relay mirror sends the beam to a focusing mirror which projects the beam onto a low altitude endo-atmospheric target. The aberrations of the atmosphere are calculated according to the Kolmogorov model, assuming a seeing value of $r = 10$. cm. This example includes beam expander aberrations and aberrations on the relay mirror and focusing mirror. Example 26 repeats the calculation with an adaptive optic model.

The atmospheric model assumes the power spectrum of the wavefront is (neglecting inner and outer scale limitations)

$$W^2(f) = \frac{0.023}{r_0^{5/3} f^{11/3}}, \quad (25.1)$$

where $w^2(f)$ is the power spectrum of the wavefront, r_0 is the seeing parameter, and f is the spatial frequency. These parameters are in radians, meters, and inverse meters respectively. The Strehl ratio after the addition of atmospheric aberration and the beam expander astigmatism is $SR = 0.34$. After propagation through the full path 10% of the energy is deposited within a 50 cm diameter spot on the target. No attenuation for atmospheric scattering optical element efficiencies has been considered, so the actual transmission will be lower.

Table. 25.1. Key parameters

beam expander diameter	50 cm
initial power	1 watt
atmospheric seeing	10 cm
relay orbit	500 km
relay to focusing mirror distance	3,000 km

Input: ex25.inp

```
c## ex25
c
c Example 25: Ground-to-Space Laser Propagation with Atmospheric Aberration
c
echo/on
set/alias/off
write/disk ex25.out/overwrite
wavelength/set 1 .48          # set wavelength of laser
array/set 1 256              # set computer array size
units/s 1 .1
gauss/c/c 1 1 1.25           # define gaussian starting distribution
clap/cir/con 1 1.25
energy/norm 1 1.             # normalize energy to 1.0
set/density 64               # set density of plot lines
title 1: starting laser distribution
plot/watch ex25_1.plt
plot/liso nsl=64 xr=1.5 yr=1.5 # plot intensity
c
c Use a 20X beam expander
c
```

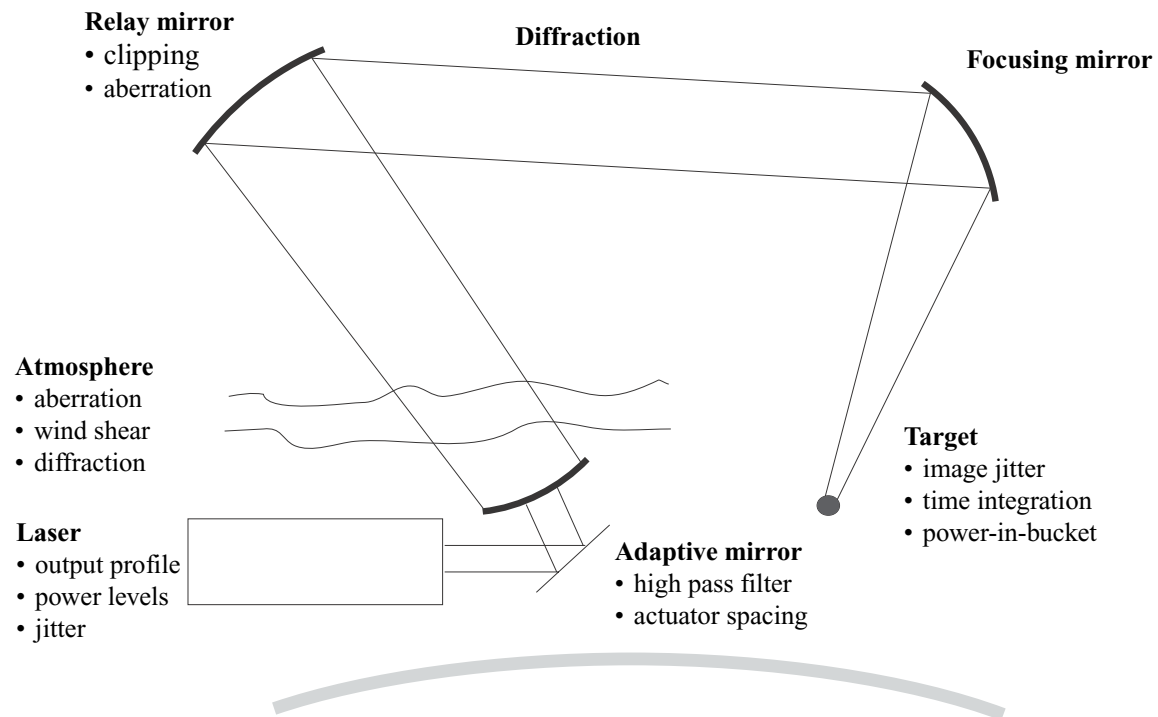


Fig. 25.1. Overview of ground-to-space laser communication system.

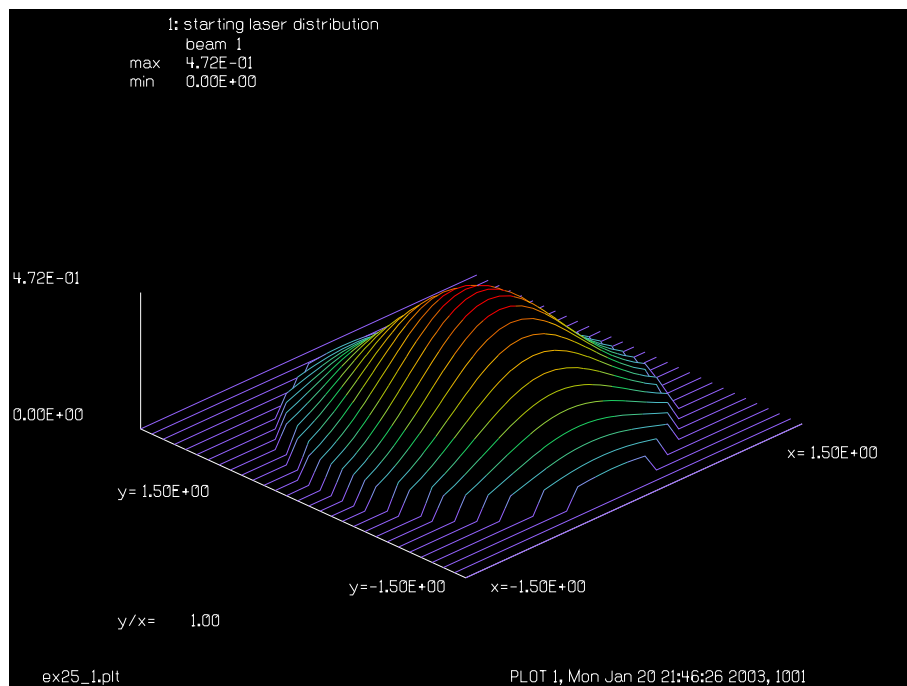


Fig. 25.2. Starting laser distribution.

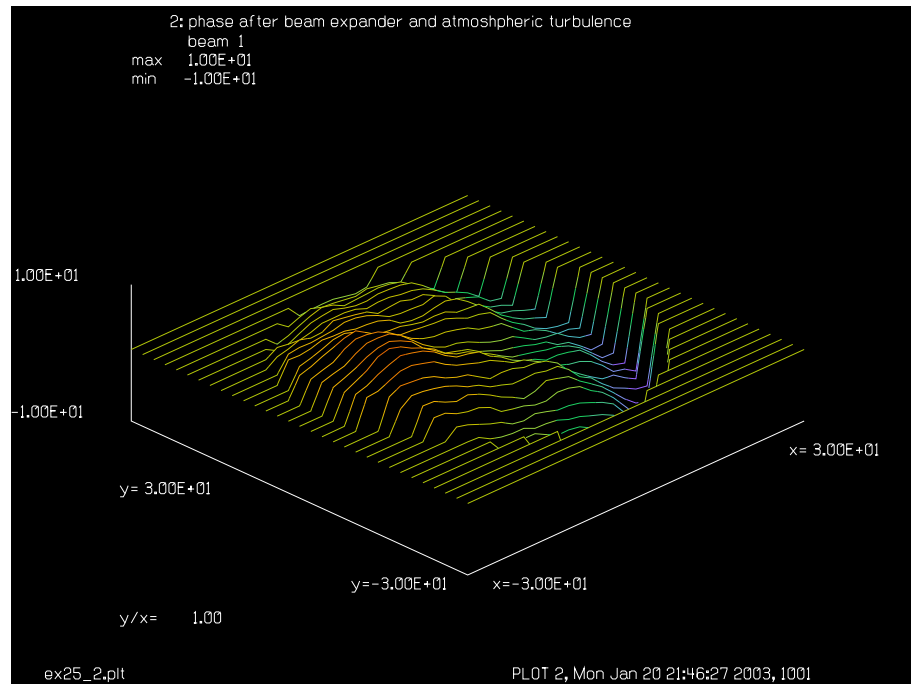


Fig. 25.3. .Phase after beam expander and atmospheric turbulence.

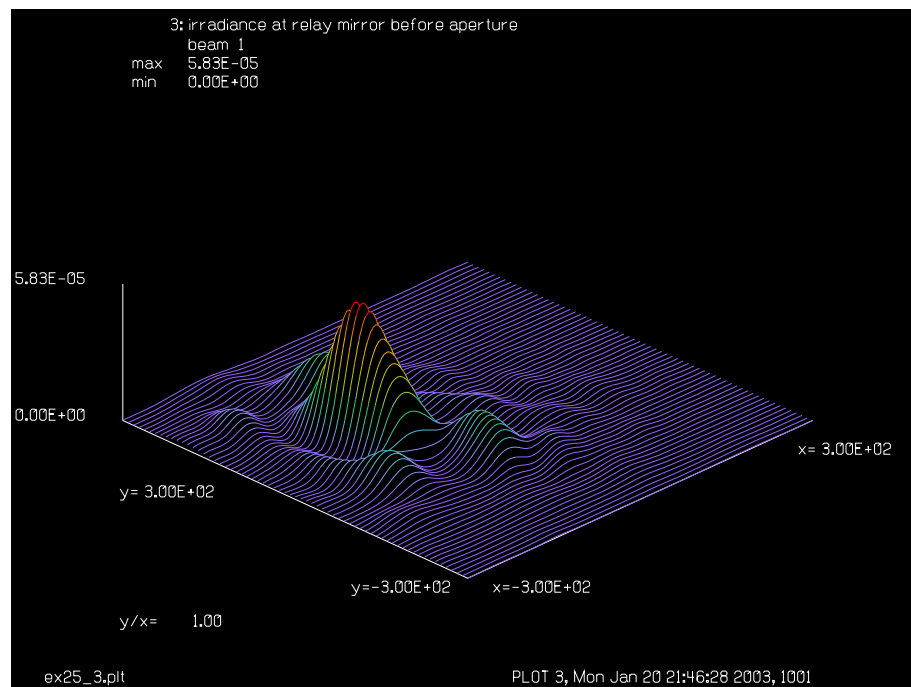


Fig. 25.4. Irradiance at relay mirror before aperture.

```

mirror 1 20 focallength      # mirror of focal length = 20.
dist -420                    # separation of mirrors
mirror 1 400 focallength     # mirror of focal length = 400.
abr/ast 1 .2 45              # astigmatism of .3 waves peak to valley
clap/cir/con 1 25            # aperture of 50 cm. diameter
phase/random/kolmogorov 1 10. 7 # atmospheric aberration, r0 = 10. cm.

```

Jump to: [Commands](#), [Theory](#)

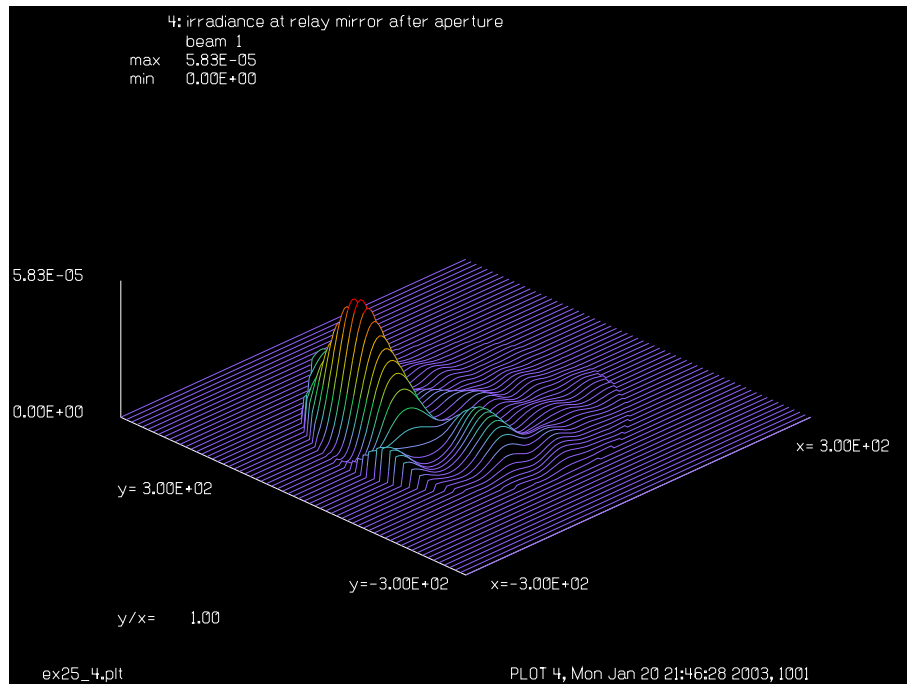


Fig. 25.5. Irradiance at relay mirror after aperture. The outline of the aperture can be seen in the wings.

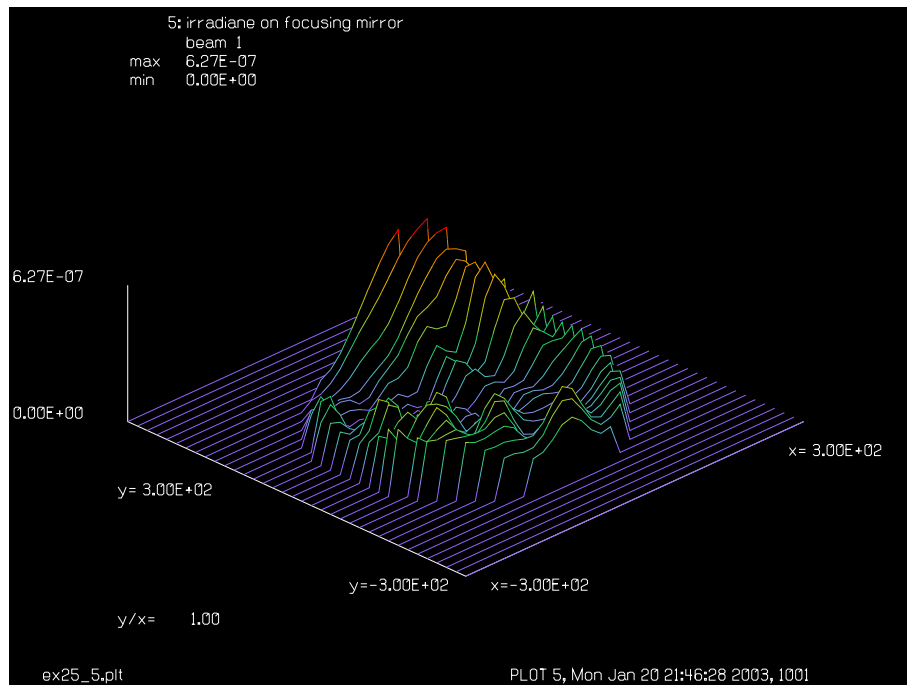


Fig. 25.6. Irradiance on focusing mirror.

```

c                                     # random seed=7.
strehl
title 2: phase after beam expander and atmospheric turbulence
c
c  Phase plots show spurious effects where the intensity is very low.
c

```

Jump to: [Commands](#), [Theory](#)

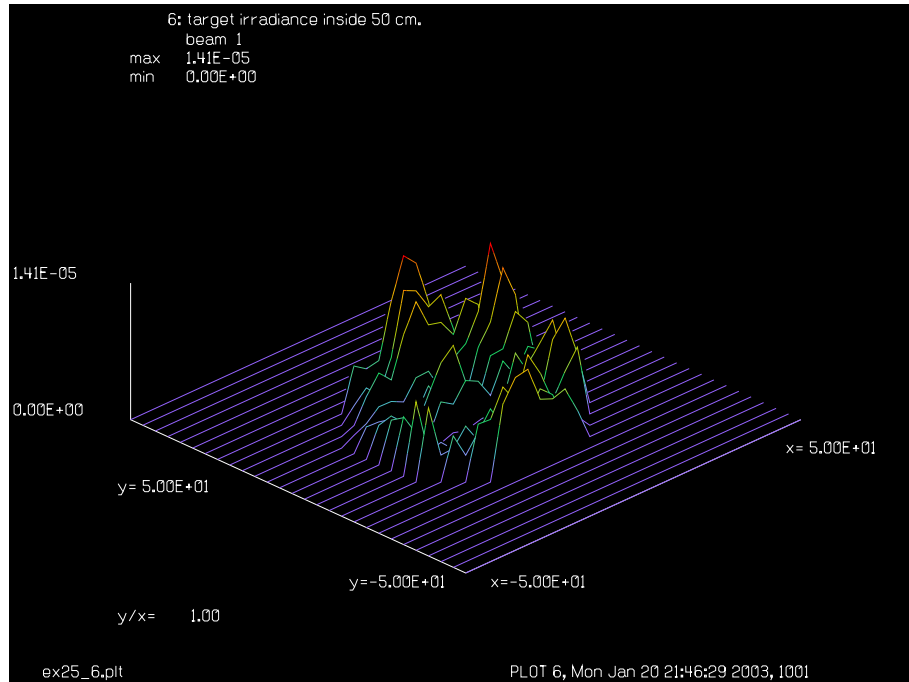


Fig. 25.7. Target irradiance inside 50 cm.

```

plot/watch ex25_2.plt
plot/liso/phase xr=30 yr=30 nsl=64 min=-10 max=10
dist 5e7                                # propagate 500 km to relay mirror
title 3: irradiance at relay mirror before aperture
plot/watch ex25_3.plt
plot/liso xr=300 yr=300 nsl=64
energy 1
clap/cir/con 1 200                      # aperture of 4 m. dia. for relay
energy 1
title 4: irradiance at relay mirror after aperture
plot/watch ex25_4.plt
plot/liso xr=300 yr=300 nsl=64
mirror 1 3e8 foc
abr/ast 1 .3 90                        # astigmatism from relay
dist 3e8                                # propagate 3000 km to focusing mirror
clap/cir/con 1 200                      # 4 meter diameter focusing mirror
energy
title 5: irradiane on focusing mirror
plot/watch ex25_5.plt
plot/liso xr=300 yr=300 nsl=64
phase/random 1 .1 50                   # add aberration on focusing mirror
mirror 1 5e7 focallength               # focus at 500 km
dist 5e7                               # propagate to target
clap/cir/con 1 25                      # aperture to find energy in 50 cm. dia.
energy
title 6: target irradiance inside 50 cm.
plot/watch ex25_6.plt
plot/liso xr=50 yr=50 nsl=64
end

```


Ex26: Ground-to-space laser propagation system with atmospheric aberration

This example shows the propagation of a laser beam from a ground-based laser to a relay mirror in low earth orbit. The relay mirror sends the beam to a fighting mirror which focuses the beam onto a low altitude endoatmospheric target. The aberrations of the atmosphere are calculated according to the Kolmogorov model, assuming a seeing value of $r = 10$. cm. This example includes beam expander aberrations and aberrations on the relay mirror and fighting mirror. The Strehl ratio after the addition of atmospheric aberration and the beam expander astigmatism is $SR = 0.34$. An adaptive mirror with actuator influence radius of 4. cm corrects the Strehl ratio to 0.87. After propagation through the full path, 56 percent of the energy is deposited within a 50 cm diameter spot on the target—a significant improvement from the 22 percent on target without the adaptive optic. No attenuation for atmospheric scattering or optical element efficiencies has been considered, so the actual transmission will be lower.

Table. 26.1. Parameters

beam expander diameter	50 cm
initial power	1 watt
atmospheric seeing	10 cm
actuator influence radius	4 cm
relay orbit	500 km
relay mirror diameter	1.5 m

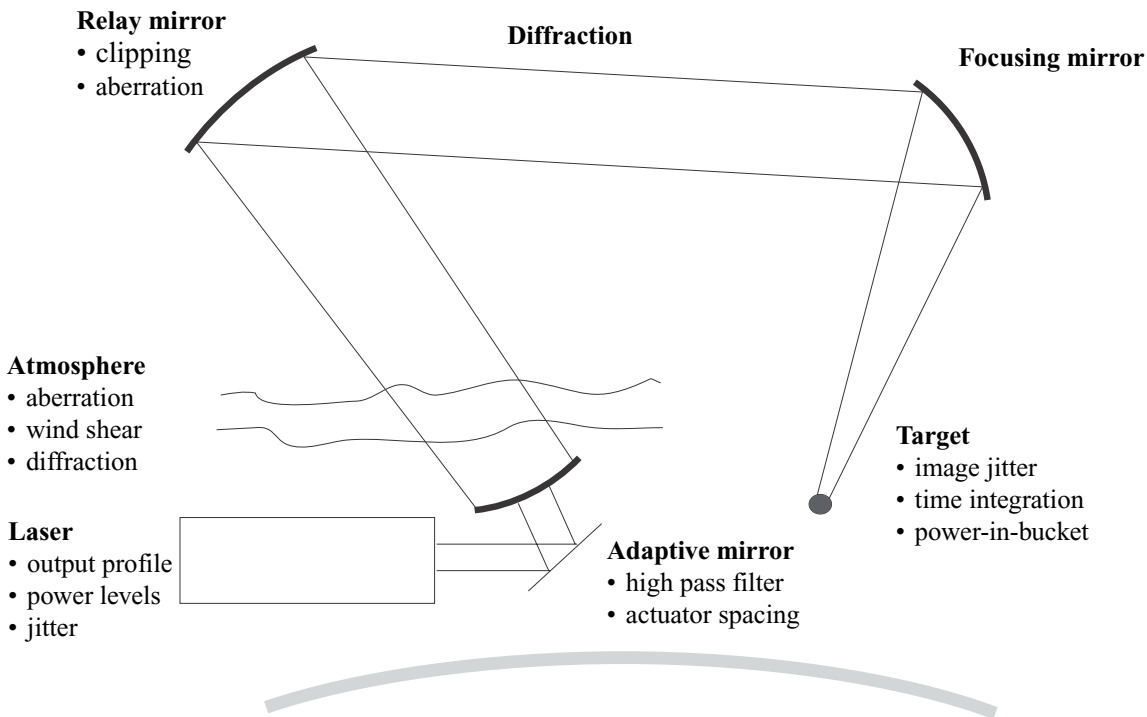


Fig. 26.1. Overview of ground-to-space laser communication system.

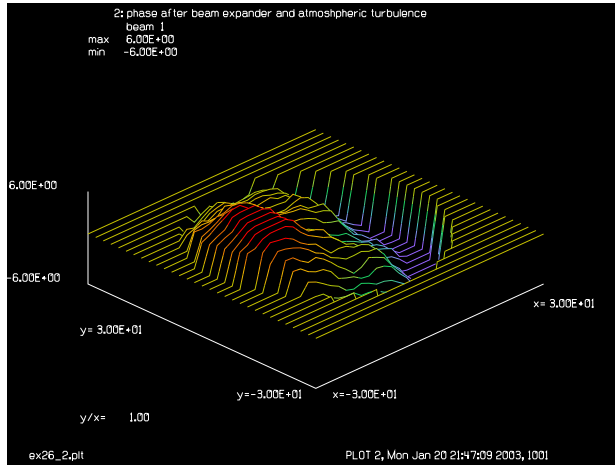


Fig. 26.2. Atmospheric aberration.

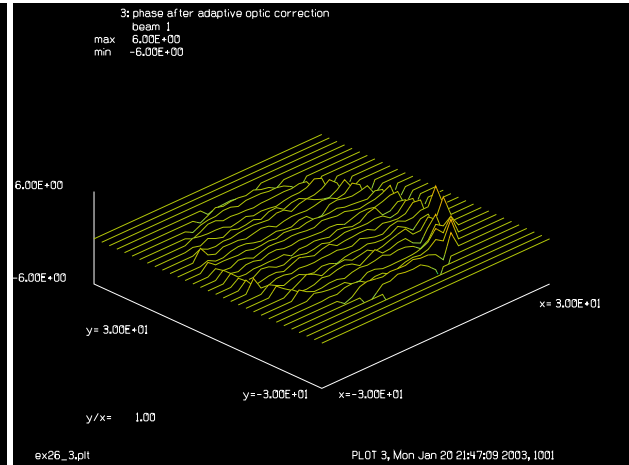


Fig. 26.3. After correction by the adaptive mirror.

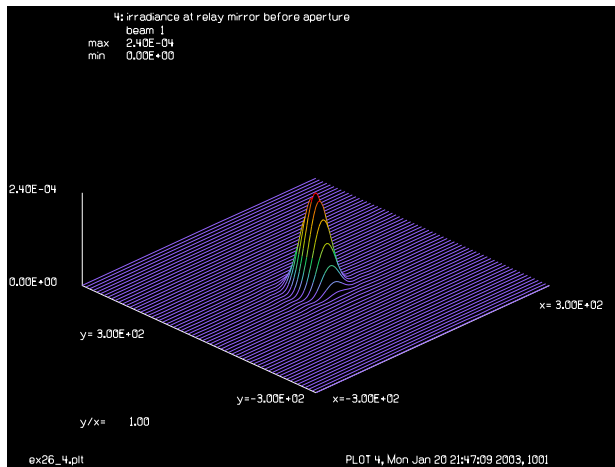


Fig. 26.4. Before the relay mirror.

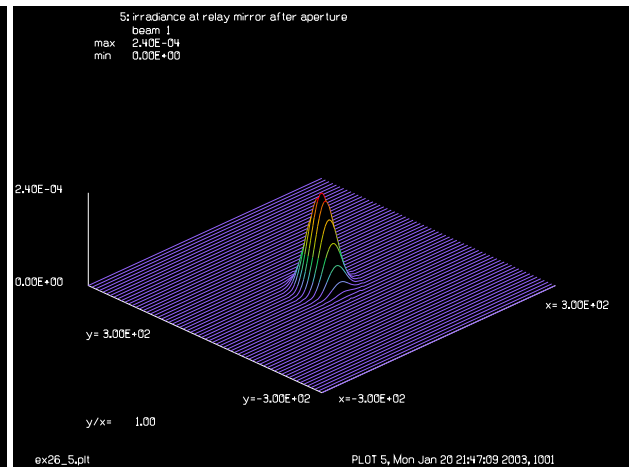


Fig. 26.5. After the relay mirror.

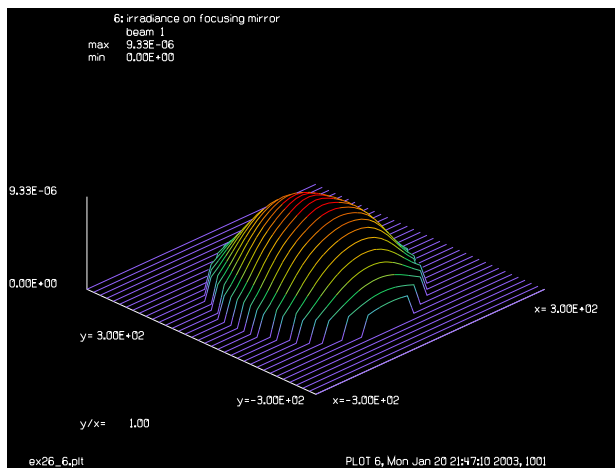


Fig. 26.6. Before the relay mirror.

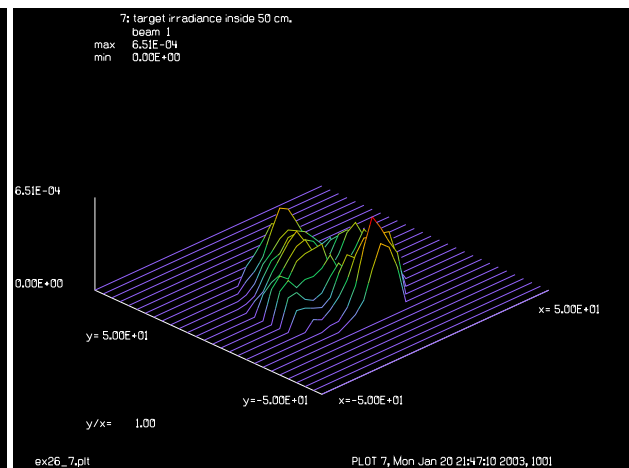


Fig. 26.7. After the relay mirror.

Input: ex26.inp

```

c## ex26
c
c Example 26: Ground-to-Space Laser Propagation with Atmospheric Aberration
c             and Adaptive Optic Correction
c
c This example shows the propagation of a laser beam from a ground-based
c laser to a relay mirror in low earth orbit. The relay mirror sends the
c beam to a focusing mirror which focuses the beam onto a low altitude
c endoatmospheric target. The aberrations of the atmosphere are calculated
c according to the Kolmogorov model, assuming a seeing value of r = 10. cm.
c This example includes beam expander aberrations and aberrations on the
c relay mirror and fighing mirror.
c
c The Strehl ratio after the addition of atmospheric aberration and the beam
c expander astigmatism is SR = .34. An adaptive mirror with actuator
c influence radius of 4. cm. corrects the Strehl ratio to .87. After
c propagation through the full path, 56 percent of the energy is deposited
c within a 50 cm. diameter spot on the target -- a significant improvement
c from the 22 percent on target without the adaptive optic. No attenuation
c for atmospheric scattering or optical element efficiencies has been
c considered, so the actual transmission will be lower.
c
c Key Parameters
c -----
c Beam expander diameter           50 cm.
c Initial power                    1 watt
c Atmospheric seeing               10. cm.
c Actuator influence radius        4. cm.
c Relay orbit                      500 km.
c Relay diameter                   1.5 m.
c Relay to focusing mirror distance 3,000 km.
c Focusing mirror diameter         4 m.
c Focusing mirror to target distance 500 km.
c Target area                      50 cm.
c
set/alias/off
wavelength/set 1 .48              # set wavelength of laser
array/set 1 256                   # set computer array size
units/s 1 .1
gauss/c/c 1 1 1.25               # define gaussian starting distribution
clap/cir/con 1 1.25
energy/norm 1 1.                  # normalize energy to 1.0
set/density 64                   # set density of plot lines
title 1: starting laser distribution
plot/watch ex26_1.plt
plot/liso nsl=64 xr=1.5 yr=1.5   # plot intensity
c
c Use a 20X beam expander
c
mirror 1 20 focallength           # mirror of focal length = 20.
dist -420                         # separatio of mirrors
mirror 1 400 focallength          # mirror of focal length = 400.
abr/ast 1 .2 45                  # astigmatism of .2 waves peak to valley
clap/cir/con 1 25                # aperture of 50 cm. diameter
phase/random/kolmogorov 1 10. 7  # atmospheric aberration, r0 = 10. cm.
c                                # random seed=7.
strehl

```

Jump to: [Commands](#), [Theory](#)

```

title 2: phase after beam expander and atmospheric turbulence
c
c Phase plots show spurious effects where the intensity is very low.
c
plot/watch ex26_2.plt
plot/liso/phase xr=30 yr=30 nsl=64 min=-6 max=6
adapt 1. 4. # adaptive optic: influence radius = 4 cm.
strehl
title 3: phase after adaptive optic correction
plot/watch ex26_3.plt
plot/liso/phase xr=30 yr=30 nsl=64 min=-6 max=6
dist 5e7 # propagate 500 km to relay mirror
title 4: irradiance at relay mirror before aperture
plot/watch ex26_4.plt
plot/liso xr=300 yr=300 nsl=64
clap/cir/con 1 75 # aperture of 1.5 m. dia. for relay
title 5: irradiance at relay mirror after aperture
plot/watch ex26_5.plt
plot/liso xr=300 yr=300 nsl=64
mirror 1 3e8 foc
abr/ast 1 .3 90 # astmatism from relay
dist 3e8 # propagate 3000 km to focusing mirror
clap/cir/con 1 200 # 4 meter diameter focusing mirror
energy
title 6: irradiance on focusing mirror
plot/watch ex26_6.plt
plot/liso xr=300 yr=300 nsl=64
phase/random 1 .1 50 # add aberration on focusing mirror
mirror 1 5e7 focallength # focus at 500 km
dist 5e7 # propagate to target
clap/cir/con 1 25 # aperture to find energy in 50 cm. dia.
energy
title 7: target irradiance inside 50 cm.
plot/watch ex26_7.plt
plot/liso xr=50 yr=50 nsl=64
end

```

Ex27: Ground-to-space laser with atmospheric aberration and jitter

Table. 27.1. Table of Ex27 examples

Ex27a: Effects of atmosphere and jitter just before relay mirror	1
Ex27b: Effects of atmospheric aberration and jitter observed just after the relay mirror	4
Ex27c: Effects of atmospheric aberration, jitter, and adaptive optics after the relay mirror	6

This example shows the use of a random tilt error and beam integration. The random tilt may be used to model beam jitter. Each call to rand() produces a new randomly chosen tilt value. The beam is propagated to the relay mirror as with Example 26. Time integration of the beam is done with the command add/inc.

Three cases are given in Table 27.2.

The jitter has an RMS value of 0.5 wavelengths at the edge of the exit pupil of the telescope. This results in a substantial amount of image motion at the relay mirror. In EX27A.INP, the motion is observed just before the aperture and the distribution simply moves without shape change. In Ex27b.inp, the distribution is shown after clipping by the aperture. In EX27C.INP, adaptive optics are used to clean up the beam. The motion due to jitter is substantial and results in significant bumpiness in the distribution even after six pulses.

Table. 27.2. Parameters

beam expander diameter	50 cm
initial power	1 watt
atmospheric seeing	10 cm
actuator influence radius	4 cm
relay orbit	500 km
relay mirror diameter	1.5 m

Ex27a: Effects of atmosphere and jitter just before relay mirror

Input: ex27a.inp

```

c## ex27a
c
c Example 27a: Ground-to-Space Laser Propagation with Atmospheric Aberration
c              and Jitter Observed just Before the Relay Mirror
c
variab/dec/int pass
echo/on
write/disk ex27a.out/overwrite
wavelength/set 1 .48           # set wavelength of laser
array/set 1 256                # set computer array size
nbeam 3
pass = 1                       # set pass counter
units/s 0 2.0
gauss/c/c 1 1 25.              # define gaussian starting distribution
clap/cir/con 1 25.
energy/norm 1 1.               # normalize energy to 1.0
set/density 64                 # set density of plot lines
abr/ast 1 .2 45                # astigmatism of .3 waves peak to valley

```

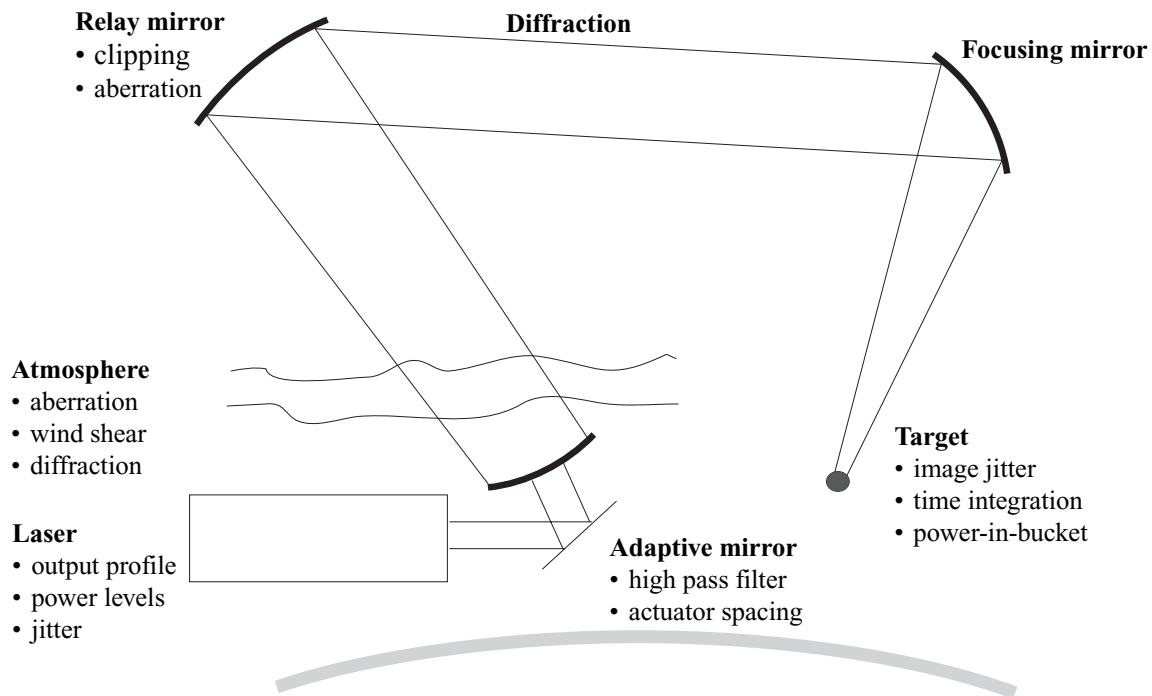


Fig. 27.1. Overview of ground-to-space laser communication system.

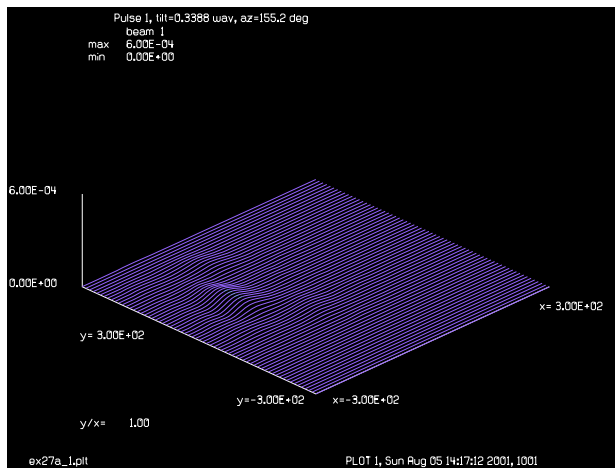


Fig. 27a.1. Pulse 1, before aperture.

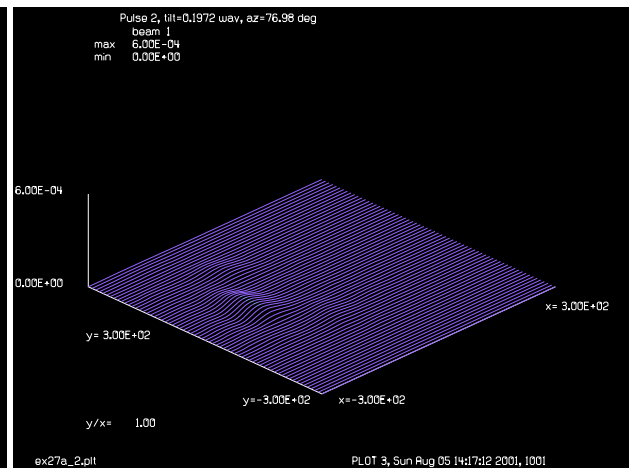


Fig. 27a.2. Pulse 2, before aperture.

```
clap/cir/con 1 25
phase/random/kolmogorov 1 10. 7
intmap 1
strehl 1
copy 1 3
copy 1 2
```

```
# aperture of 50 cm. diameter
# atmospheric aberration, r0 = 10. cm.
# store distribution in Beam 3
# store distribution in Beam 2
```

Jump to: [Commands](#), [Theory](#)

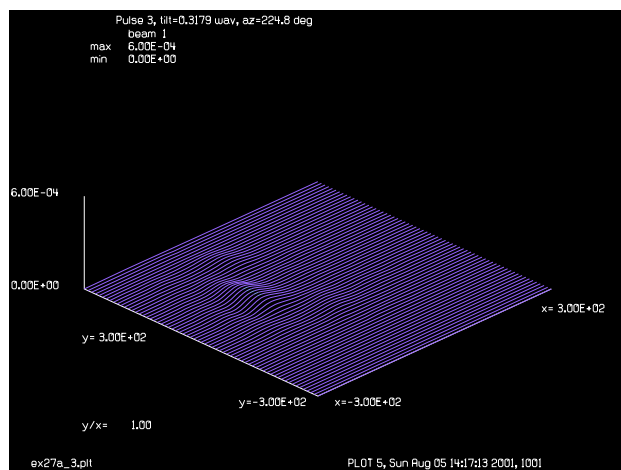


Fig. 27a.3. Pulse 3, before aperture.

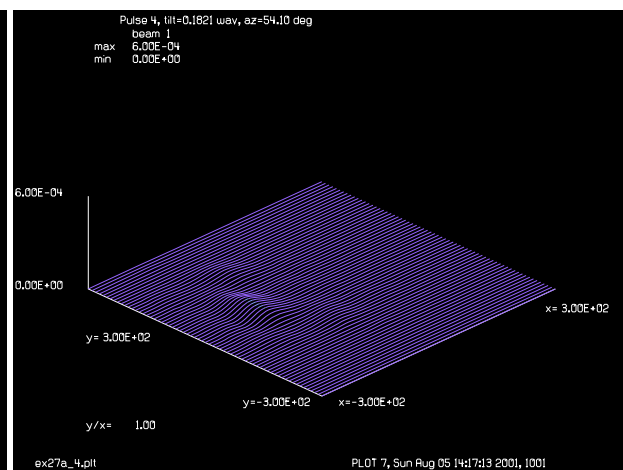


Fig. 27a.4. Pulse 4, before aperture.

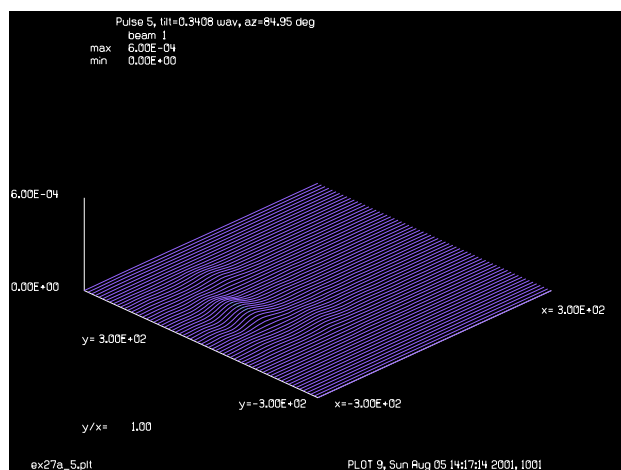


Fig. 27a.5. Pulse 5, before aperture.

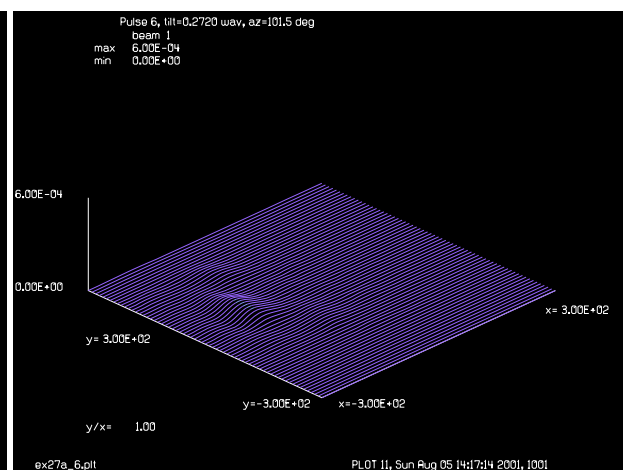


Fig. 27a.6. Pulse 6, before aperture.

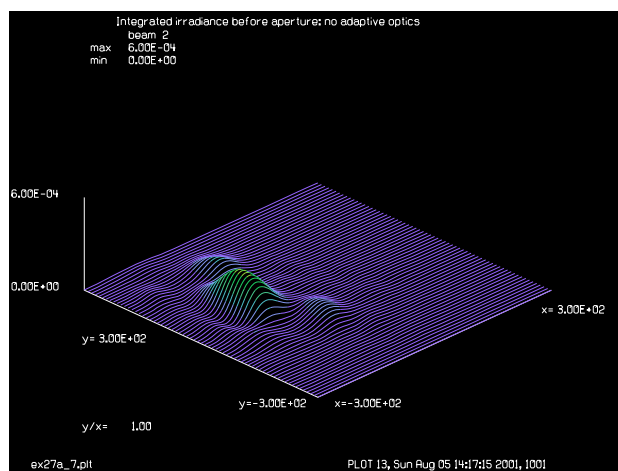


Fig. 27a.7. Integrated irradiance, before aperture.

```
dist 5e7 2
```

```
# propagate Beam 2 to set up beam size
```

Jump to: [Commands](#), [Theory](#)

```

intmap 1
clear 2 0.                                # empty Beam 2
variab/set/random tilt 1 11                # set random seed to 11
macro/def pass/overwrite
  units/s 1 2.
  copy 3 1                                # reset distribution of Beam 1
  tilt = .5*rand()                         # random tilt of RMS .5 wavelengths
  phase = 360.*rand()                      # random azimuthal angle
  abr/tilt 1 tilt phase                    # random tilt representing jitter
  dist 5e7 1                              # propagate 500 km to relay mirror
  title Pulse @pass, tilt=@tilt wav, az=@phase deg
  plot/watch ex27a_@pass.plt
  plot/liso ibeams=1 ibeame=1 xr=300 yr=300 nsl=64 min=0 max=6e-4
  intmap 1
  add/inc 2 1                             # add beam incoherently to Beam 2
  plot/w plot1.plt
  plot/l 2
  energy
  pass = pass + 1
macro/end
macro/run pass/6
title Integrated irradiance before aperture: no adaptive optics
plot/watch ex27a_@pass.plt
plot/liso ibeams=2 ibeame=2 xr=300 yr=300 nsl=64 min=0 max=6e-4
end

```

Ex27b: Effects of atmospheric aberration and jitter observed just after the relay mirror

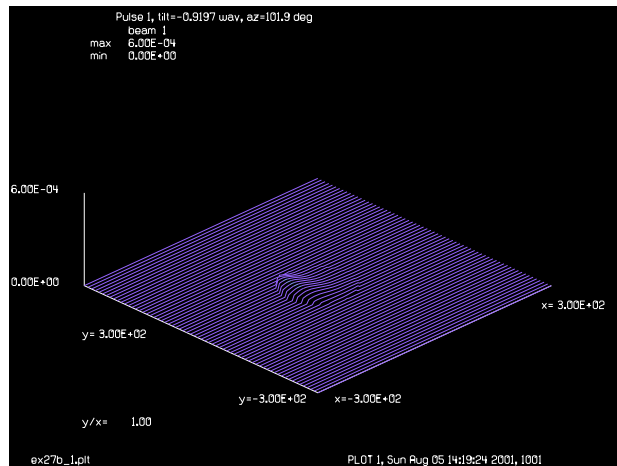


Fig. 27b.1. Pulse 1, before aperture.

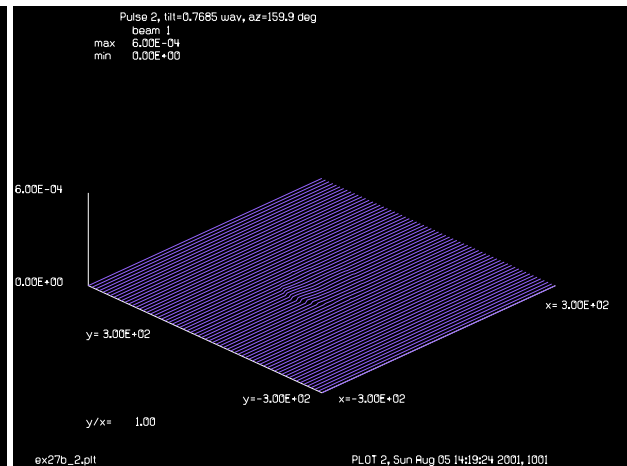


Fig. 27b.2. Pulse 2, before aperture.

Input: ex27b.inp

```

c## ex27b
c

```

Jump to: [Commands](#), [Theory](#)

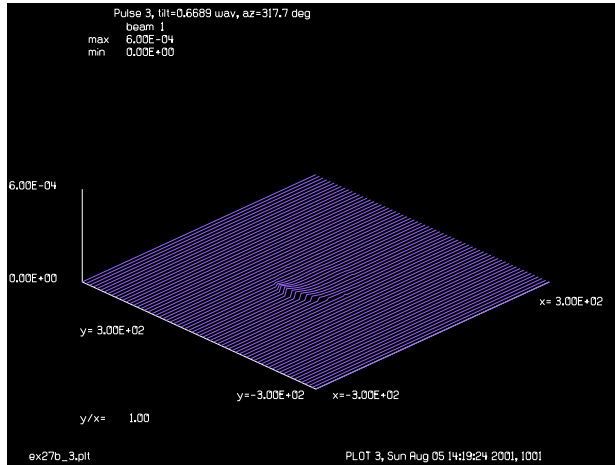


Fig. 27b.3. Pulse 3, before aperture.

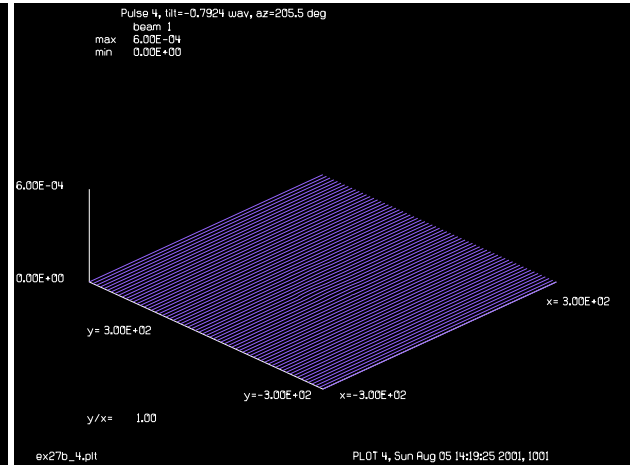


Fig. 27b.4. Pulse 4, before aperture.

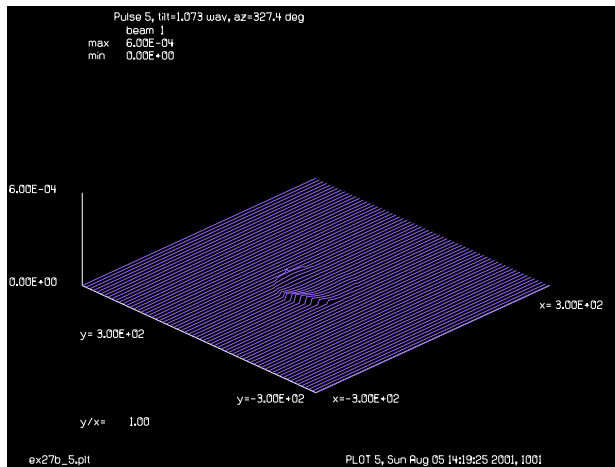


Fig. 27b.5. Pulse 5, before aperture.

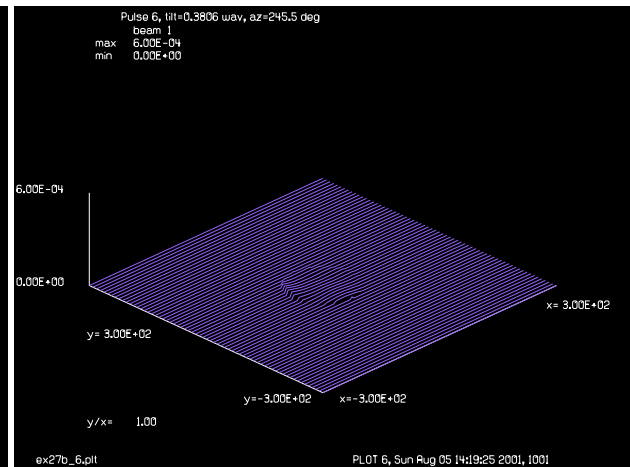


Fig. 27b.6. Pulse 6, before aperture.

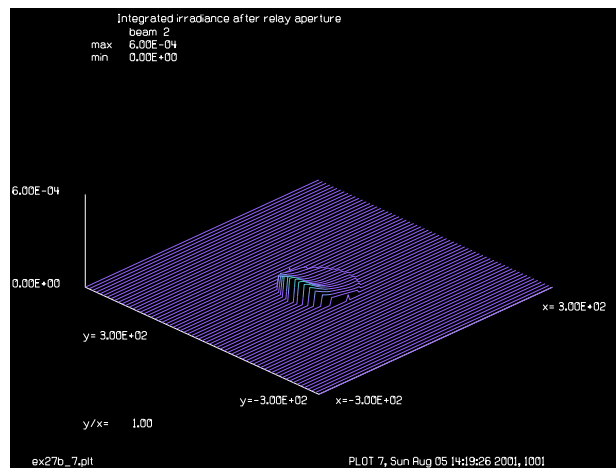


Fig. 27b.7. Integrated irradiance, before aperture.

c Example 27b: Ground-to-Space Laser Propagation with Atmospheric Aberration

Jump to: [Commands](#), [Theory](#)

```

c          and Jitter Observed just after the Relay Mirror
c
set/alias_stop/off
variab/dec/int pass
echo/on
write/disk ex27b.out/overwrite
wavelength/set 1 .48          # set wavelength of laser
array/set 1 256              # set computer array size
nbeam 3
pass = 1                     # set pass counter
units/s 0 2.0
gauss/c/c 1 1 25.           # define gaussian starting distribution
clap/cir/con 1 25.
energy/norm 1 1.            # normalize energy to 1.0
set/density 64              # set density of plot lines
abr/ast 1 .2 45             # astigmatism of .3 waves peak to valley
clap/cir/con 1 25           # aperture of 50 cm. diameter
phase/random/kolmogorov 1 10. 7 # atmospheric aberration, r0 = 10. cm.
strehl 1
copy 1 3                    # store distribution in Beam 3
copy 1 2                    # store distribution in Beam 2
dist 5e7 2                  # propagate Beam 2 to set up beam size
clear 2 0.                  # empty Beam 2
variab/set/random tilt 1 3  # set random seed to 3
macro/def pass/overwrite
    units/s 1 2.
    copy 3 1                # reset distribution of Beam 1
    variab/set/random tilt .5 # random tilt of RMS .5 wavelengths
    variab/set/phaser phase  # random phaser 0 < X < 2pi
    abr/tilt 1 tilt phase    # random tilt representing jitter
    dist 5e7 1              # propagate 500 km to relay mirror
    clap/cir/con 1 75       # aperture of 1.5 m. dia. for relay
    title Pulse @pass, tilt=@tilt wav, az=@phase deg
    plot/watch ex27b_@pass.plt
    plot/liso ibeams=1 ibeame=1 xr=300 yr=300 nsl=64 min=0 max=6e-4
    add/inc 2 1             # add beam incoherently to Beam 2
    energy
    pass = pass + 1
macro/end
macro/run pass/6
title Integrated irradiance after relay aperture
plot/watch ex27b_@pass.plt
plot/liso ibeams=2 ibeame=2 xr=300 yr=300 nsl=64 min=0 max=6e-4
end

```

Ex27c: Effects of atmospheric aberration, jitter, and adaptive optics after the relay mirror

Input: `ex27c.inp`

```

c## ex27c
c

```

Jump to: [Commands](#), [Theory](#)

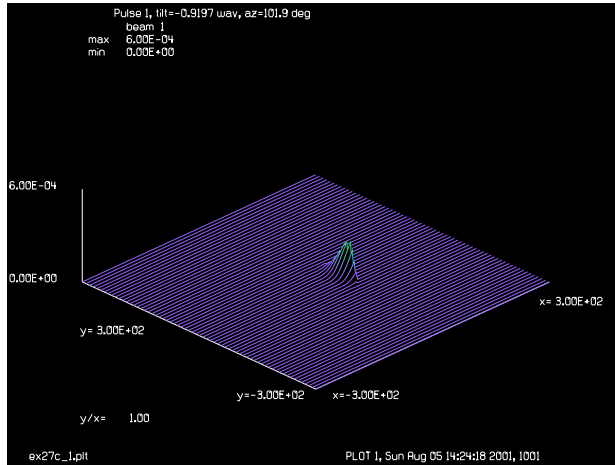


Fig. 27c.1. Pulse 1, before aperture.

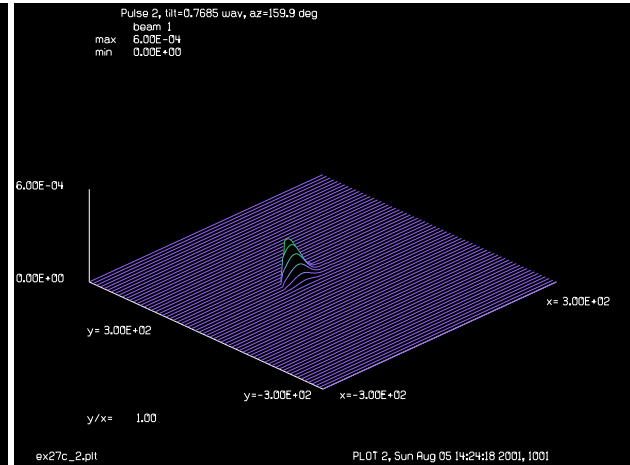


Fig. 27c.2. Pulse 2, before aperture.

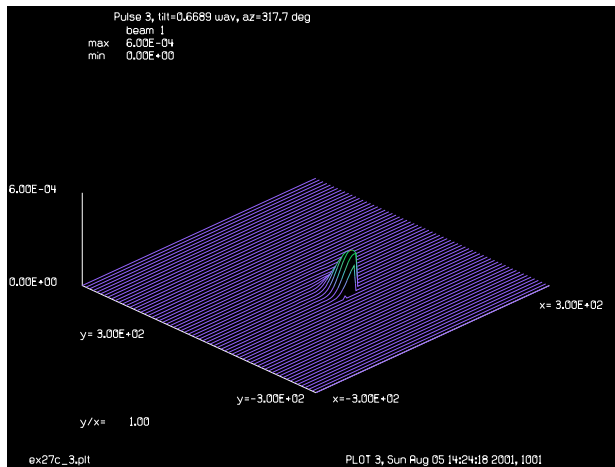


Fig. 27c.3. Pulse 3, before aperture.

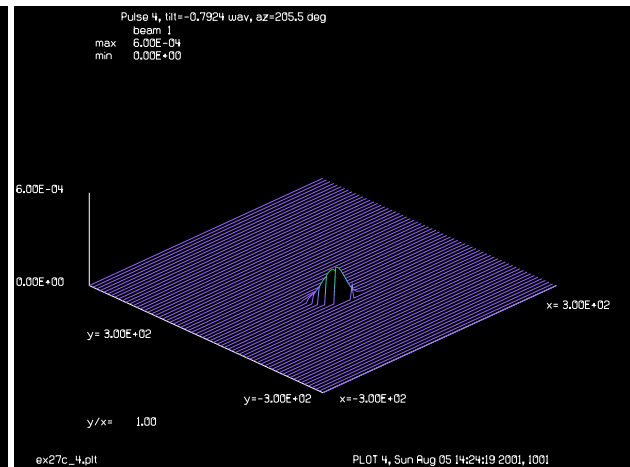


Fig. 27c.4. Pulse 4, before aperture.

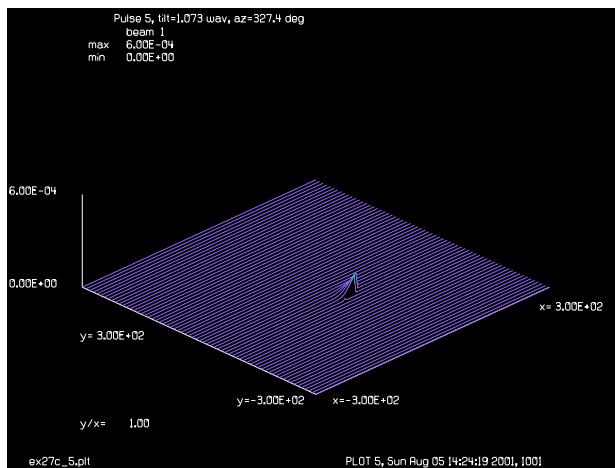


Fig. 27c.5. Pulse 5, before aperture.

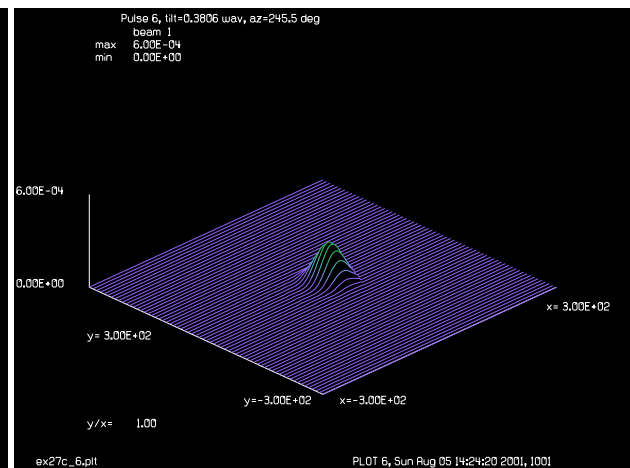


Fig. 27c.6. Pulse 6, before aperture.

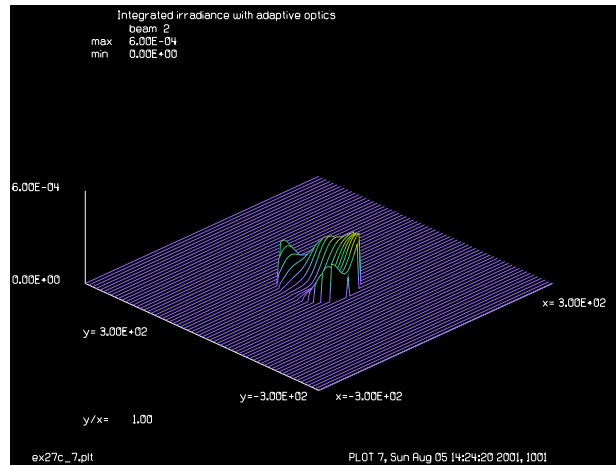


Fig. 27c.7. Integrated irradiance, before aperture.

```

c Example 27c: Ground-to-Space Laser Propagation with Atmospheric Aberration,
c Jitter, and Adaptive Optics after the Relay Mirror
c
variab/dec/int pass
echo/on
write/disk ex27c.out/overwrite
wavelength/set 1 .48 # set wavelength of laser
array/set 1 256 # set computer array size
nbeam 3
pass = 1 # set pass counter
units 0 2.0
gauss/c/c 1 1 25. # define gaussian starting distribution
clap/cir/con 1 25.
energy/norm 1 1. # normalize energy to 1.0
set/density 64 # set density of plot lines
abr/ast 1 .2 45 # astigmatism of .3 waves peak to valley
clap/cir/con 1 25 # aperture of 50 cm diameter
phase/random/kolmogorov 1 10. 7 # atmospheric aberration, r0 = 10. cm
strehl
adapt 1. 4. # adaptive optic: inf. radius = 4 cm.
strehl 1
copy 1 3 # store distribution in Beam 3
copy 1 2 # store distribution in Beam 2
dist 5e7 2 # propagate Beam 2 to set up beam size
clear 2 0. # empty Beam 2
variab/set/random tilt 1 3 # set random seed to 3
macro/def pass/overwrite
  units 1 2.
  copy 3 1 # reset distribution of Beam 1
  variab/set/random tilt .5 # random tilt of RMS .5 wavelengths
  variab/set/phaser phase # random phaser 0 < X < 2pi
  abr/tilt 1 tilt phase # random tilt representing jitter
  dist 5e7 1 # propagate 500 km to relay mirror
  clap/cir/con 1 75 # aperture of 1.5 m. dia. for relay
  title Pulse @pass, tilt=@tilt wav, az=@phase deg
  plot/watch ex27c_@pass.plt
  plot/liso ibeams=1 ibeame=1 xr=300 yr=300 nsl=64 min=0 max=6e-4
  add/inc 2 1 # add beam incoherently to Beam 2
  energy
  pass = pass + 1

```

```
macro/end
macro/run pass/6
title Integrated irradiance with adaptive optics
plot/watch ex27c_@pass.plt
plot/liso ibeams=2 ibeame=2 xr=300 yr=300 nsl=64 min=0 max=6e-4
end
```


Ex28: Phased array

Table. 28.1. Table of Ex28 examples

Ex28a: Phased Array	1
Ex28b: 11 x 11 laser array with steering by piston terms on each laser	3

This example shows how to create a segmented aperture using two beams and coherent addition to combine various subapertures.

Ex28a: Phased Array

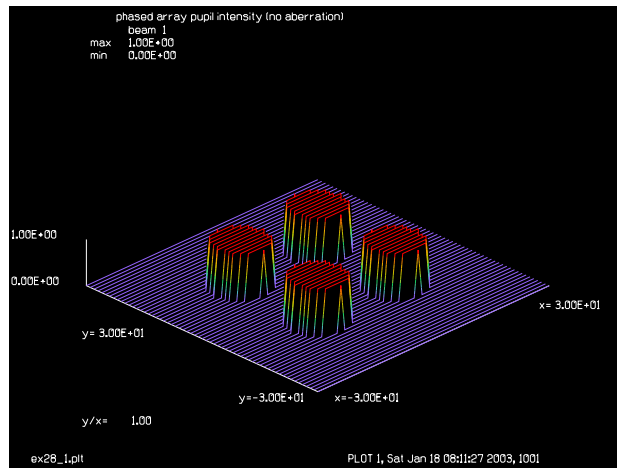


Fig. 28a.1. Pupil irradiance.

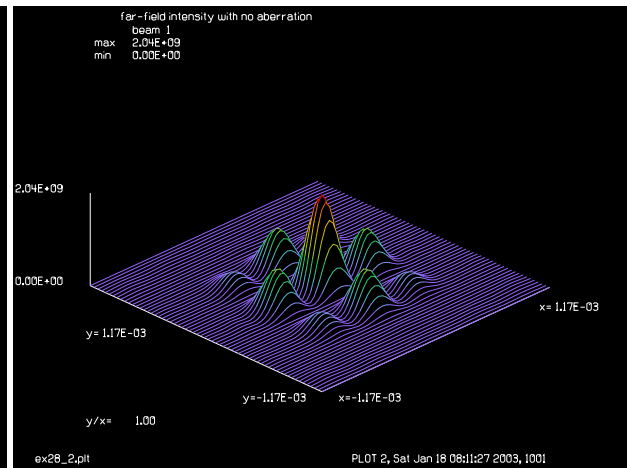


Fig. 28a.2. Far-field pattern with no aberration.

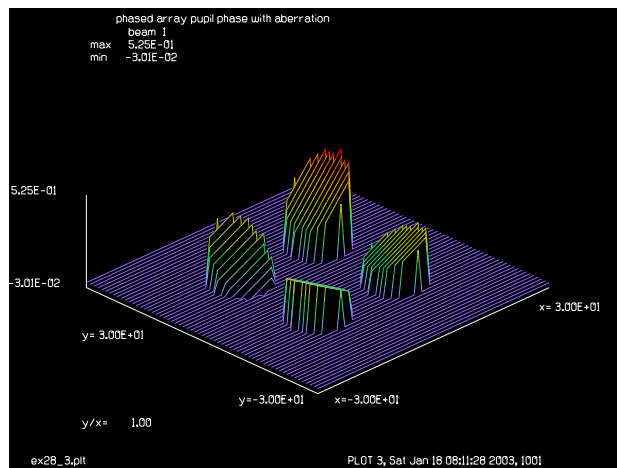


Fig. 28a.3. Phase in pupil after tilt and piston addition.

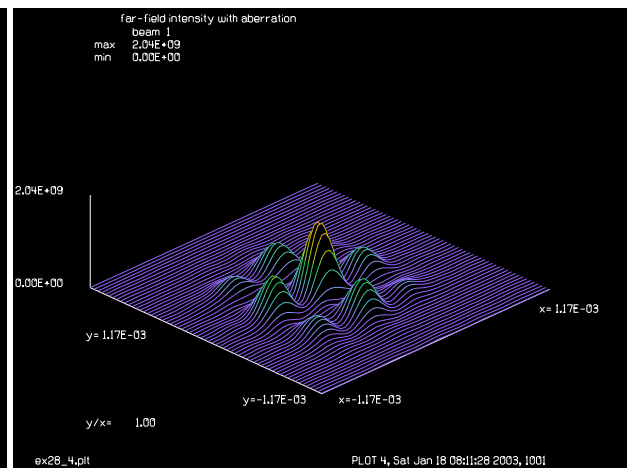


Fig. 28a.4. Far-field pattern with aberration.

Input: `ex28a.inp`

Ex28b: c## ex28a

```

c
c Example 28: Phased Array
c
array/s 1 256
nbeam 2                # expand the number of beams
units/s 0 1            # set units to 0
wavelength/set 0 1     # set wavelength to 1
c
c Aberration-free calculation

```

Jump to: [Commands](#), [Theory](#)


```

c
clear 1 1          # set entire field to unity
c                  # make Subaperture 1 in Beam 1
clap/cir/con 1 6 10 10      # define an off-set aperture
clear 2 1          # make Subaperture 2 in Beam 2
clap/cir/con 2 6 -10 10
add/coh 1 2        # add Beam 2 to Beam 1 coherently
clear 2 1          # make Subaperture 3 in Beam 2
clap/cir/con 2 6 10 -10
add/coh 1 2
clear 2 1          # make Subaperture 4 in Beam 2
clap/cir/con 2 6 -10 -10
add/coh 1 2
title phased array pupil intensity (no aberration)
plot/watch ex28_1.plt
plot/l/intensity 1 ns=128 h=.2 xrad=30 yrad=30 # plot pupil intensity
lens 1 100
dist 100 1
title far-field intensity with no aberration
variab/set units 1 units
variab/set peak 1 peak
plot/watch ex28_2.plt
plot/l/intensity 1 max=peak ns=128 xrad=units*30 yrad=units*30
# display far field pattern
c
c Repeat calculation with aberration
c
units/s 0 1        # set units to 0
clear 2 0.          # clear Beam 2
clear 1 1          # set entire field to unity
c                  # make Subaperture 1 in Beam 1
clap/cir/con 1 6 10 10      # define an off-set aperture
abr/tilt 1 1 85 32        # add tilt, 1 wav at edge, az=85 deg
c                  rnorm=32
clear 2 1          # make Subaperture 2 in Beam 2
clap/cir/con 2 6 -10 10
abr/tilt 2 1 25 32
add/coh 1 2        # add Beam 2 to Beam 1 coherently
clear 2 1          # make Subaperture 3 in Beam 2
clap/cir/con 2 6 10 -10
abr/tilt 2 .5 120 32
add/coh 1 2
clear 2 1          # make Subaperture 4 in Beam 2

```

```

clap/cir/con 2 6 -10 -10
abr/tilt 2 -.5 70 32
phase/piston 2 .5          # add piston error
add/coh 1 2
title phased array pupil phase with aberration
plot/watch ex28_3.plt
plot/l/wave 1 ns=128 xrad=30 yrad=30  # plot pupil wavefront
lens 1 100
dist 100 1
title far-field intensity with aberration
plot/watch ex28_4.plt
plot/l/intensity 1 max=peak ns=64 xrad=units*30 yrad=units*30
  # display far-field pattern
end
11 x 11 laser array with steering by piston terms on each laser

```

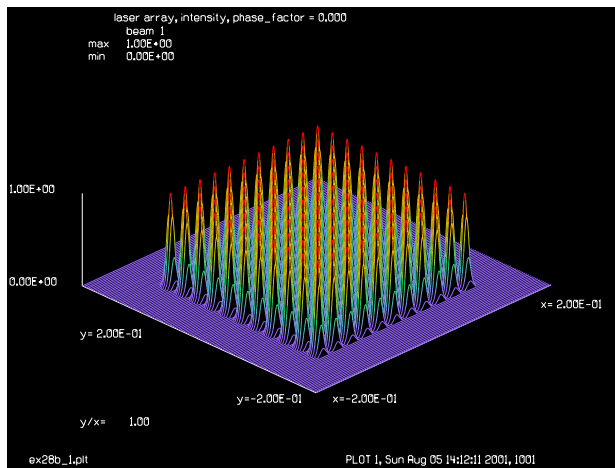


Fig. 28b.1. Pupil irradiance of an 11 x 11 array of laser diodes.

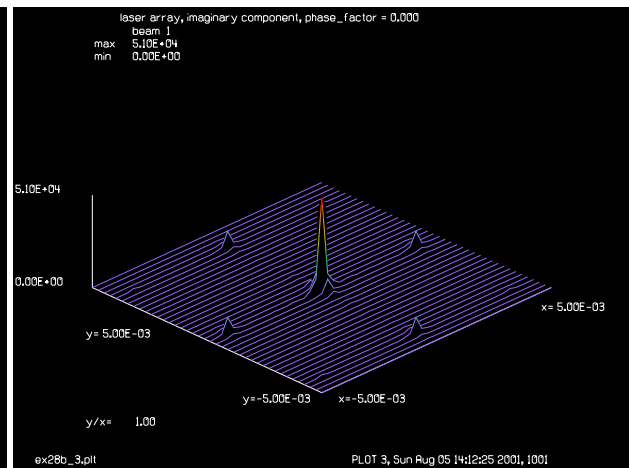


Fig. 28b.2. Far-field pattern with no tilt.

Input: ex28b.inp

```

c## ex28b
#
# Example of an array of lasers (such as vecsel's) with controlled
# piston values to introduce tilt.
#
mem/set/b 40
variable/dec/int Nlinw iplot
iplot = 1
Nline = 2048
array/s 1 Nline
nbeam 2 data
num_pixels_per_beam = 128          # number of pixels per side
array/s 2 num_pixels_per_beam data # generating array
lambda = 1.04e-4                   # wavelength in cm
wavelength/set 0 1e4*lambda         # wavelength has units/s [microns]

```

Jump to: [Commands](#), [Theory](#)

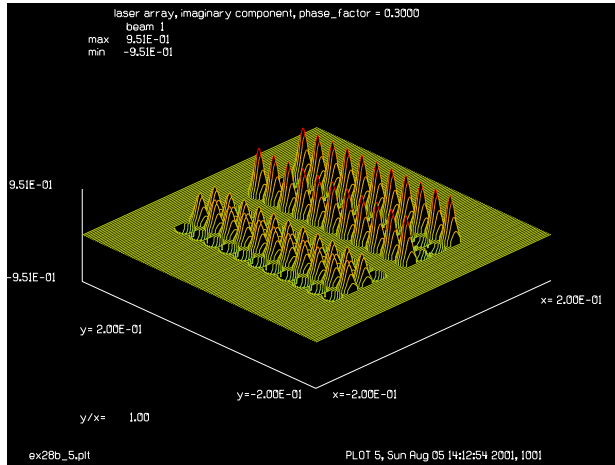


Fig. 28b.3. Imaginary distribution for diode array with tilt introduced by piston terms.

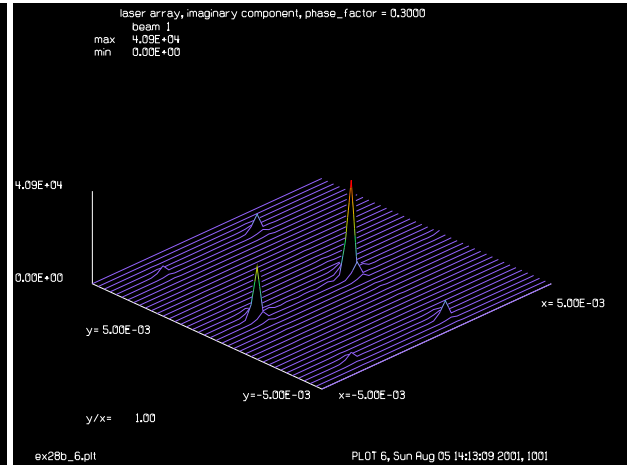


Fig. 28b.4. Far-field pattern with tilt-piston phase. Note shift in x-direction and change in sidelobes.

```
laser_sep = 250e-4
unitset = laser_sep/num_pixels_per_beam
units/s 2 unitset

azimuthal_angle = 90      # in degrees.  0.0 refers to y-axis

nsize = 11
msize = nsize      # to make square array

macro/def inner/o
  row = row + 1 list
  macro make_beam
  lensa/rec/paste 1 2 row col laser_sep laser_sep
  title row = @row, col = @col
macro/end

macro/def outer/o
  col = col + 1 list
  row = -(nsize + 1)/2
  macro/run inner/nsize
macro/end

macro/def make_beam/o
  global/def 2 0 0 0 0 0 0
  array/s 2 num_pixels_per_beam
  zreff/se 2 0
  gaussian/c/c 2 1 .0083 1
  Piston_r = sqrt(col^2 + row^2)      # radius of point (col,row)
  Piston_theta = atan2(row,col)      # azimuthal angle of point
  Piston_offset_angle = azimuthal_angle*(pi/180) # equiv to azdeg in abr/
  Piston_tilt = phase_factor*Piston_r*cos(Piston_theta-Piston_offset_angle)
  phase/piston 2 Piston_tilt*360
  clap/cir/no 2 laser_sep/2.0
macro/end

macro/def system/o
```

Jump to: [Commands](#), [Theory](#)

```
array/s 1 Nline
units/s 1 unitset
zreff/se 1 0
clear 1 0
col = -(msize + 1)/2
macro/run outer/msize
plot/w ex28b_@iplot.plt; iplot = iplot + 1
title laser array, intensity, phase_factor = @phase_factor
plot/l 1 ns=128
plot/w ex28b_@iplot.plt; iplot = iplot+1
title laser array, imaginary component, phase_factor = @phase_factor
plot/l/a 1 ns=128
lens 1 1
prop 1
plot/w ex28b_@iplot.plt; iplot = iplot + 1
plot/l/i xrad=.005 ns=128
macro/end
phase_factor = .0
macro system
phase_factor = .3
macro system
```

Ex29: Atmospheric aberration with wind shear

This example shows wind shear. The atmospheric aberration moves in the positive x-direction in this example. The movement is actually cyclical.

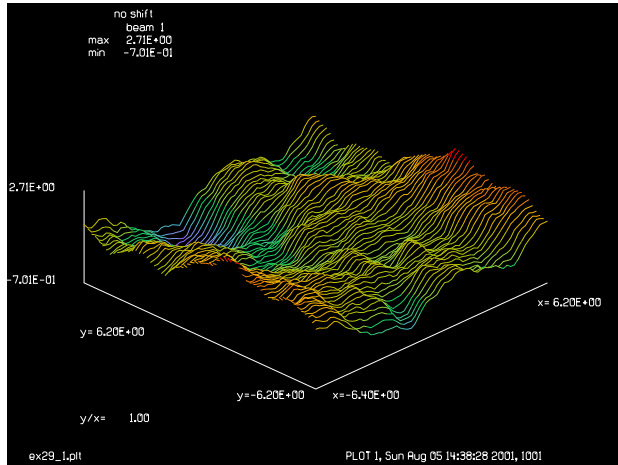


Fig. 29a.1. No shift.

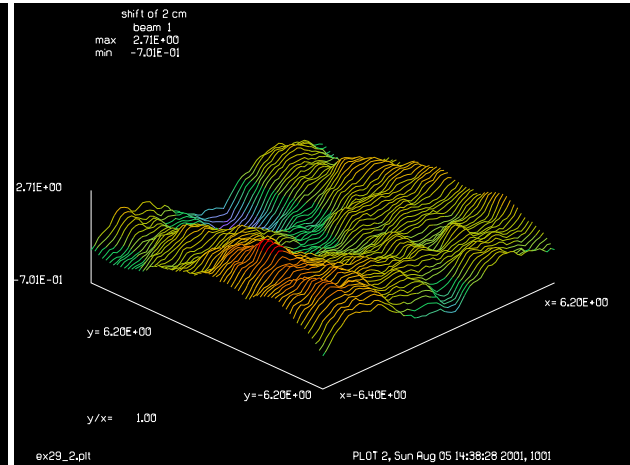


Fig. 29a.2. Shift of 2 cm.

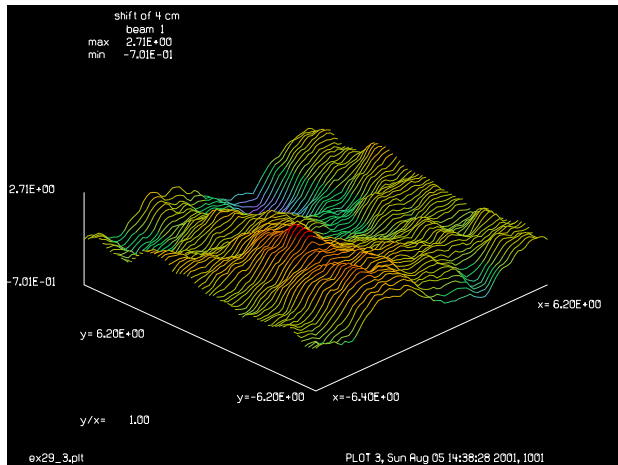


Fig. 29a.3. Shift of 4 cm.

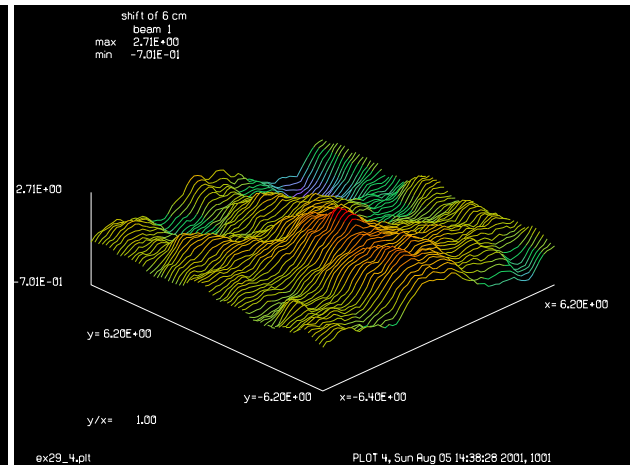


Fig. 29a.4. Shift of 6 cm.

Input: ex29.inp

```
c## ex29!047120255126340
c
c Example 29: Wind Shear
c
echo/on
write/disk ex29.out/overwrite
wavelength/set 1 .48          # set wavelength of laser
array/set 1 64                # set computer array size
units/s 1 .2
energy/norm 1 1.              # normalize energy to 1.0
set/density 64                # set density of plot lines
phase/random/kolmogorov 1 10.3 # atmospheric aberration, r0 = 3. cm.
```

```
title no shift
plot/watch ex29_1.plt
plot/liso/phase nsl=64
shift 1 2. 90.          # shift 2 cm in x-direction.
title shift of 2 cm
plot/watch ex29_2.plt
plot/liso/phase nsl=64
title shift of 4 cm
shift 1 2. 90.          # shift 2 cm in x-direction.
plot/watch ex29_3.plt
plot/liso/phase nsl=64
title shift of 6 cm
shift 1 2. 90.          # shift 2 cm in x-direction.
plot/watch ex29_4.plt
plot/liso/phase nsl=64
end
```

Ex30: Specklon, near- and far-field

This example is a comparison of the far field patterns of two beams with sinusoidal phase aberrations which are 180 different in phase. This illustrates characteristics of the “specklon” described by Zel'dovich as playing an important role in phase conjugation by stimulated Brillouin scattering in a focused geometry. The irradiance patterns must be different for phase conjugation to occur. This begins at about 3 cm from focus for an aperture of 1 cm dia., focal length of 100 cm, and frequency of 8 cycles/aperture.

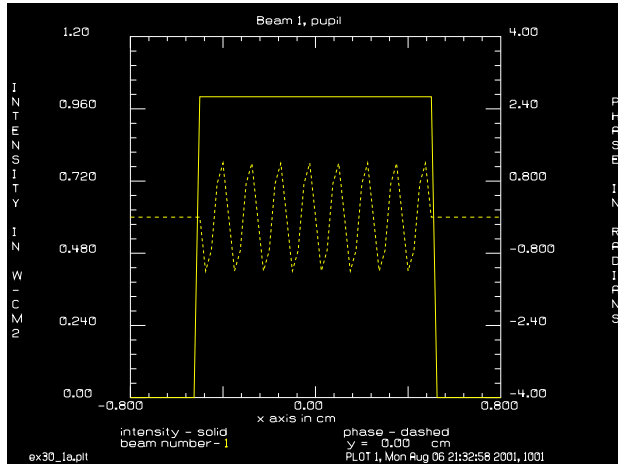


Fig. 30.1. Beam 1, at pupil showing linear phase modulation.

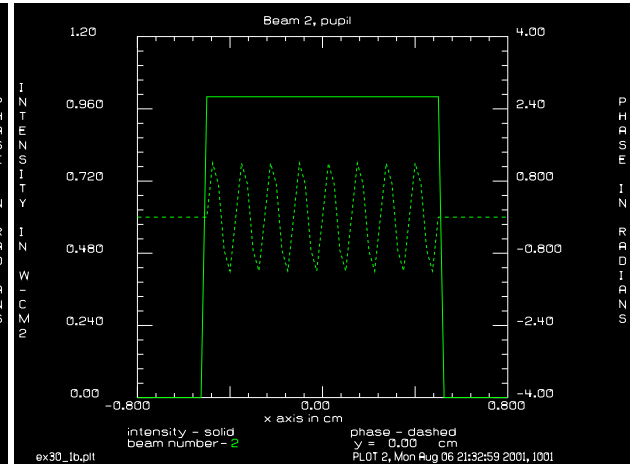


Fig. 30.2. Beam 2, at pupil. Phase modulation is 180 degrees to Fig. 30.1.

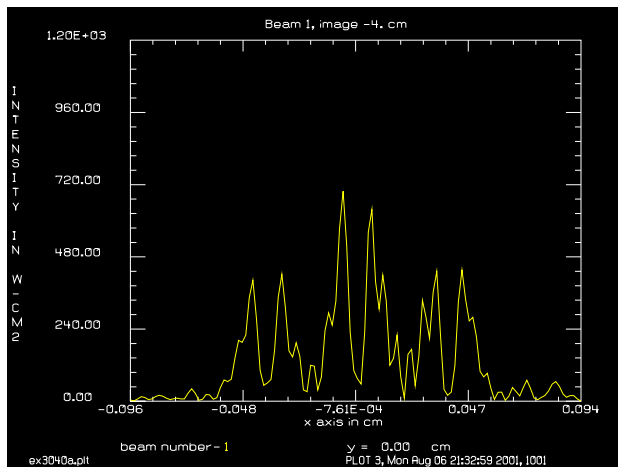


Fig. 30.3. Beam 1, -4 cm from focus.

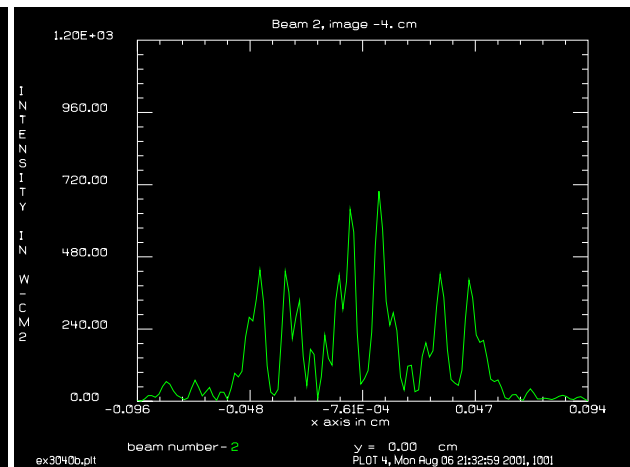


Fig. 30.4. Beam 2, -4 cm from focus. Intensity is different from Fig. 30.3.

Input: `ex30.inp`

```
c## ex30
c
c Example 30: Specklon
c
c Comparison of the far field patterns of two linear, sinusoidal phase
c aberrations. Comparison is made in the vicinity of the focus. This
```

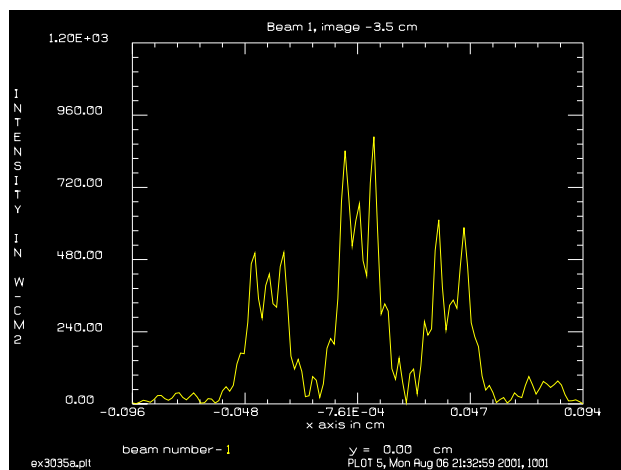


Fig. 30.5. Beam 1, -3.5 cm from focus.

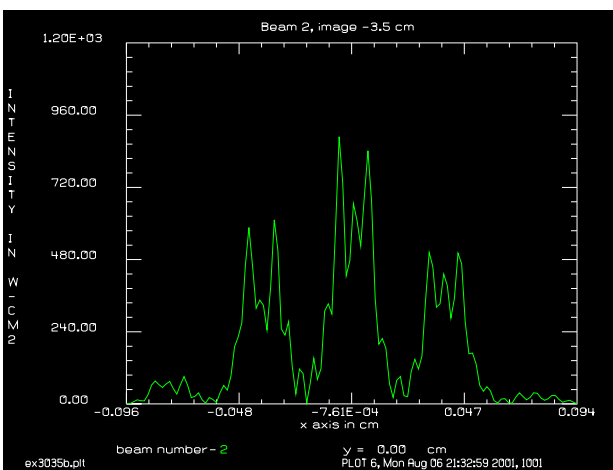


Fig. 30.6. Beam 2, -3.5 cm from focus.

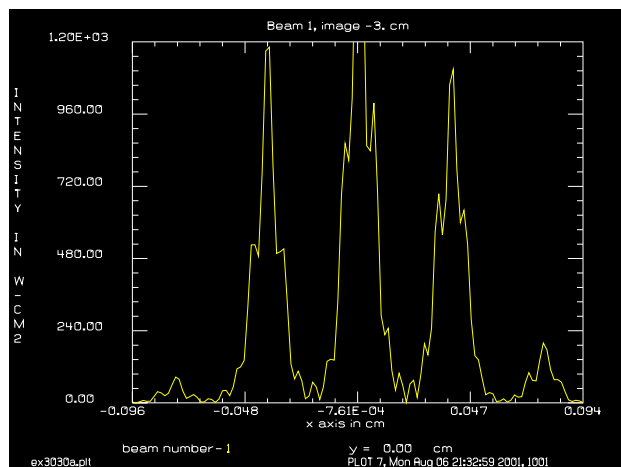


Fig. 30.7. Beam 1, -3 cm from focus.

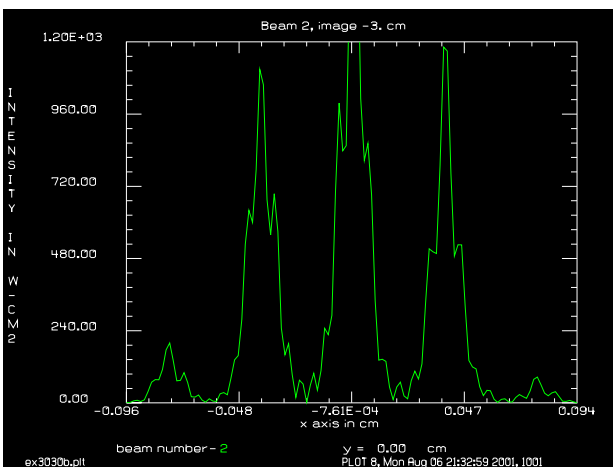


Fig. 30.8. Beam 2, -3 cm from focus.

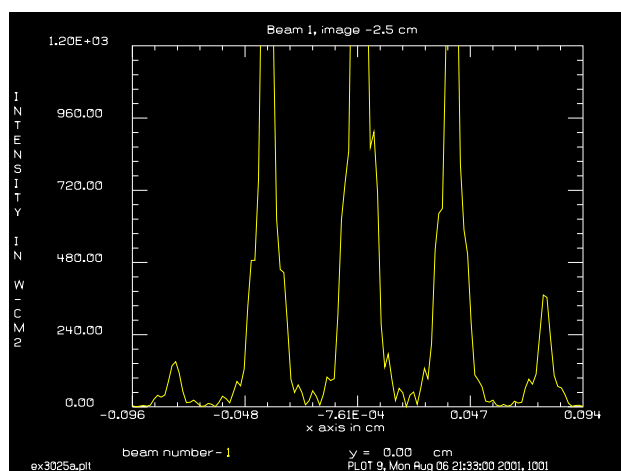


Fig. 30.9. Beam 1, -2.5 cm from focus (with slight clipping to keep same scale).

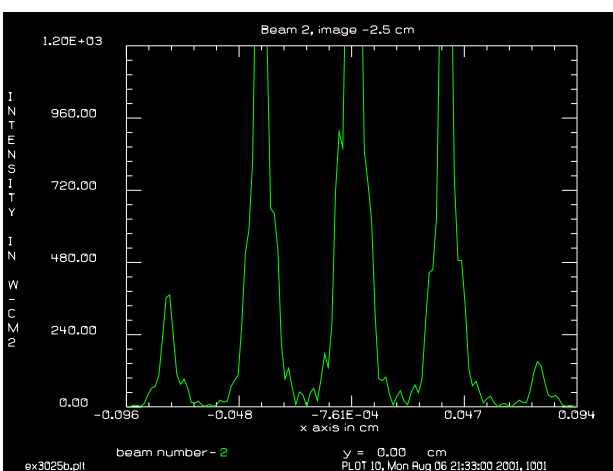


Fig. 30.10. Beam 2, -2.5 cm from focus. Intensity is very similar to Fig. 30.9.

c example illustrates characteristics of the "specklon" described by

Jump to: [Commands](#), [Theory](#)

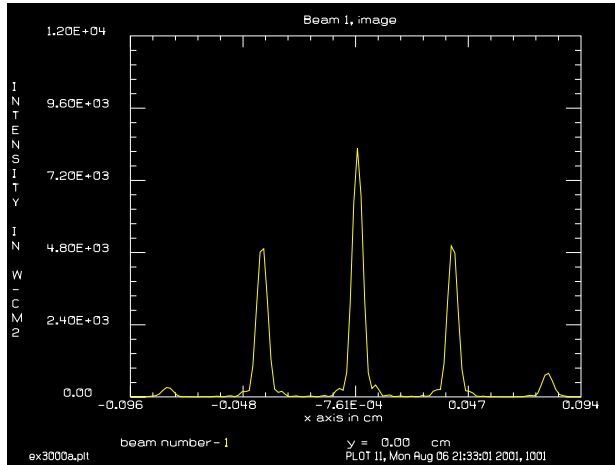


Fig. 30.11. Beam 1, at focus (10 X scale).

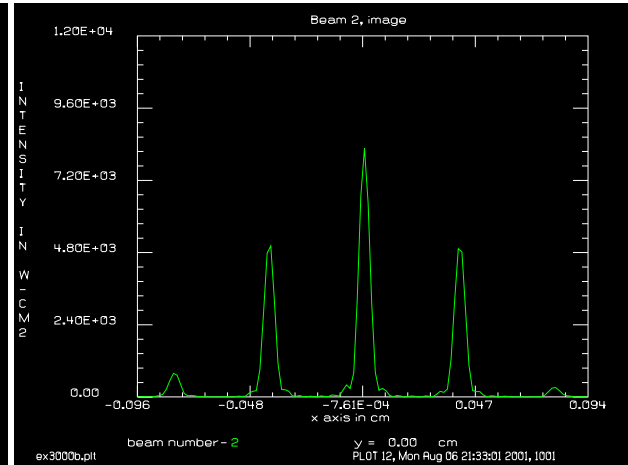


Fig. 30.12. Beam 2, at focus (10 X scale).

```

c Zel'dovich as playing an important in phase conjugation by stimulated
c Brillouin scattering in a focused geometry.
c
echo/on
write/disk ex30.out/over
array/set 1 128
nbeam 2
units/s 0 .025
wavelength/set 0 .5
clear 1 1
clear 2 1
clap/cir/con 0 .5
abr/lrip 1 .2 4 90
title Beam 1, pupil
plot/watch ex30_1a.plt
plot/x 1 0 -.8 .8 0 1.2 -4 4
abr/lrip 2 -.2 4 90
title Beam 2, pupil
plot/watch ex30_1b.plt
plot/x 2 0 -.8 .8 0 1.2 -4 4
lens 0 100
zone/fix 100 6.1
zone/in 1
zone/in 2
dist 96.
c
c Propagate succesively to -4, -3.5, -3, -2.5, and 0 cm from focus.
c
title Beam 1, image -4. cm
plot/watch ex3040a.plt
plot/x/int 1 fmax=1.2e3
title Beam 2, image -4. cm
plot/watch ex3040b.plt
plot/x/int 2 fmax=1.2e3
dist .5
title Beam 1, image -3.5 cm
plot/watch ex3035a.plt
plot/x/int 1 fmax=1.2e3
title Beam 2, image -3.5 cm
plot/watch ex3035b.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/x/int 2 fmax=1.2e3
dist .5
title Beam 1, image -3. cm
plot/watch ex3030a.plt
plot/x/int 1 fmax=1.2e3
title Beam 2, image -3. cm
plot/watch ex3030b.plt
plot/x/int 2 fmax=1.2e3
dist .5
title Beam 1, image -2.5 cm
plot/watch ex3025a.plt
plot/x/int 1 fmax=1.2e3
title Beam 2, image -2.5 cm
plot/watch ex3025b.plt
plot/x/int 2 fmax=1.2e3
dist 3.
title Beam 1, image
plot/watch ex3000a.plt
plot/x/int 1 fmax=1.2e4
title Beam 2, image
plot/watch ex3000b.plt
plot/x/int 2 fmax=1.2e4
end
c## ex39!445369246937236
c
c Example 39: Gaussian Phase Correction Factor
c
macro/def angle/o
  point/list/ij 1 33 33
  write/off
  variab/set x point/sr
  variab/set y point/si
  angle = 180*atan2(y,x)/pi
  write/on
macro/end
gaussian/c/res 1 1 1          # Gaussian beam with waist = 1
zbound                       # Rayleigh distance, Zr = 2.964E3
dist 2.963e3                  # Move to Rayleigh distance
echo/on
macro angle
c peak amplitude (.5,-.5), phase = -45 deg
angle=
pause
dist -2.963e3                 # Move back to waist
macro angle
c peak amplitude (1.,0.), phase = 0 deg
angle=
pause
dist 1e9                      # Move to 1e9
macro angle
c peak amplitude (.0,-2.964E-6), phase = -90 deg
angle=
pause
dist -1e9                     # Move to waist
macro angle
c peak amplitude (1.,0.), phase = 0 deg
angle=
pause
dist -2.963e3                 # To negative Rayleigh distance

```

Jump to: [Commands](#), [Theory](#)

```
macro angle
c peak amplitude (.5,.5), phase = 45 deg
angle=
pause
dist 2.963e3                # Move to waist
macro angle
c peak amplitude (1.,0.), phase = 0 deg
angle=
pause
dist -1e9                   # Move to -1e9 from waist
macro angle
c peak amplitude (0.,2.964E-6), phase = 90 deg
angle=
pause
end
```


Ex31: Thermal blooming

Table. 31.1. Table of Ex31 examples

Ex31a: Propagation with no thermal blooming	1
Ex31b: Thermal blooming, no kinetic cooling	4
Ex31c: Thermal blooming with kinetic cooling and aberration	5

This example shows examples of thermal blooming calculations. Example 31a shows the far field pattern with no thermal blooming and no atmospheric aberration, as shown in Fig. 31.1. Fig. 31.2 shows the phase effect of the wind blowing the hot air toward the viewer. Figure 31.3 shows the far field pattern for thermal blooming with no atmospheric aberration. The characteristic secondary blip and “sugar scoop” effect is characteristic of thermal blooming. We show the peak irradiance versus distance (Figure 4). The thermal blooming causes the beam to steer many centimeters and to diverge beyond about 2.8 km. GLAD automatically finds the best tilt and focus curvature after each blooming step. This helps keep the array size matched to the beam as it converges or diverges. We may see a jump in the apparent size of the array if the beam passes inside the Rayleigh range. In the last calculation, Example 5 and 6, atmospheric aberration is added. A seeing constant of $r_0 = 10$ cm for the 4 km path was used. The contribution at each of 40 steps is

$$r_i = r_0(10)^{3/5} = 92 \text{ cm} \quad (31.1)$$

where r_i is the seeing for each step. The addition of random aberration can occasionally improve performance. This would not be expected in a linear system where errors generally add but with the nonlinear thermal blooming such effects as this may occur.

Table. 31.2. Parameters

Beam profile	gaussian $r_0 = 25$ cm
Peak intensity	1000 w/cm
Focus point	4 km
Wavelength	10.6 μ
Target velocity	(4,-4) m/sec

Ex31a: Propagation with no thermal blooming

Input: `ex31a.inp`

```
c## ex31a!316564773899758
c
c Example 31a: Propagation with no Thermal Blooming
c
c Example of thermal blooming propagation. The example consists
c of three input files: EX31A.INP, EX31B.INP, EX3AC.INP
c
c In this example we calculate the intensity at the target at
c a distance of 4 km.
c
write/d ex31a.out/o          # write to output file
```

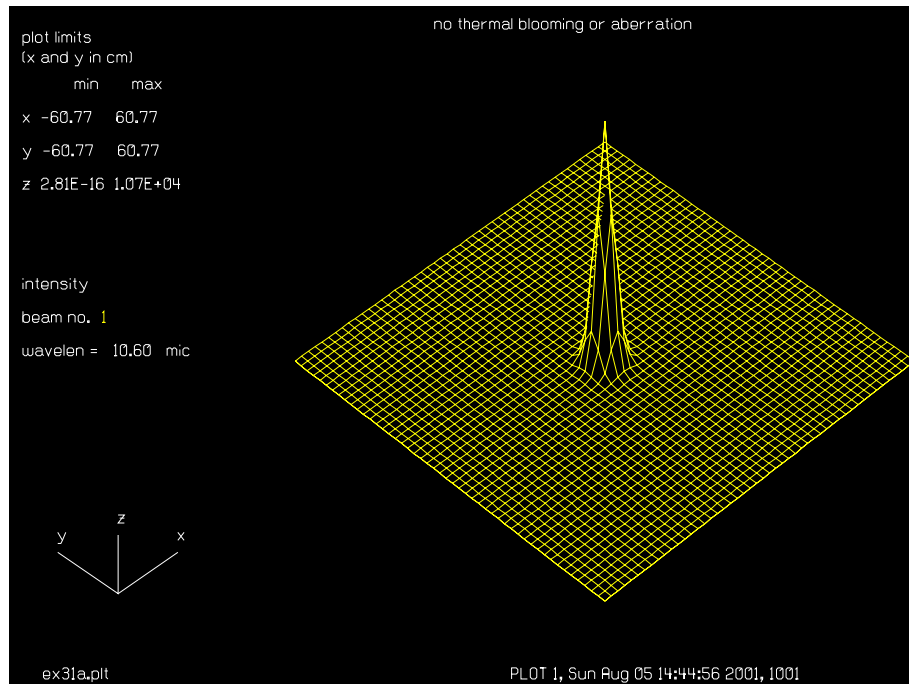


Fig. 31.1. Far-field pattern at 4 km with no thermal blooming.

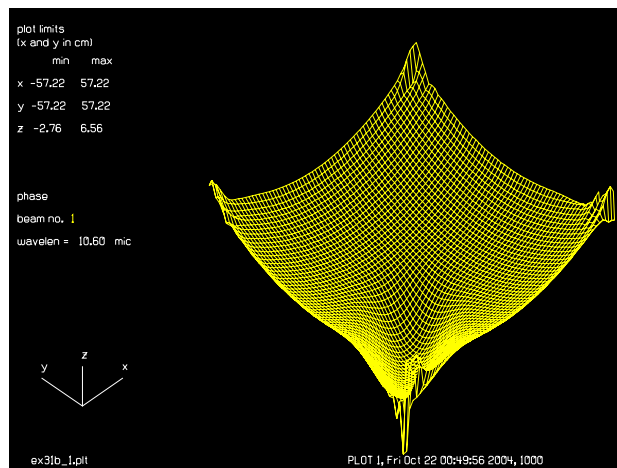


Fig. 31.2. Phase at a distance of 1 km showing the region of heated error caused by the wind blowing toward the viewer.

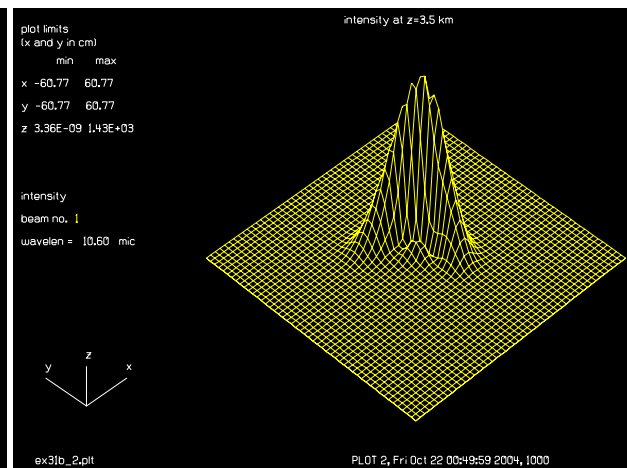


Fig. 31.3. Intensity in the far-field at 3.5 km showing a pronounced sugar scoop effect and a secondary peak on the upstream side which is characteristic. The pattern is not completely symmetrical because the wind changed direction in the course of the propagation.

```
units/s 1 2.5
gaus/c/c 1 500 25.
energy
lens 1 4e5
dist 4e5
peak
plot/watch ex31a.plt
set/density 32
```

```
# set gaussian beam to 500 w/cm**2 peak
# list energy
# focus beam to 4 km.

# set plot density
```

Jump to: [Commands](#), [Theory](#)

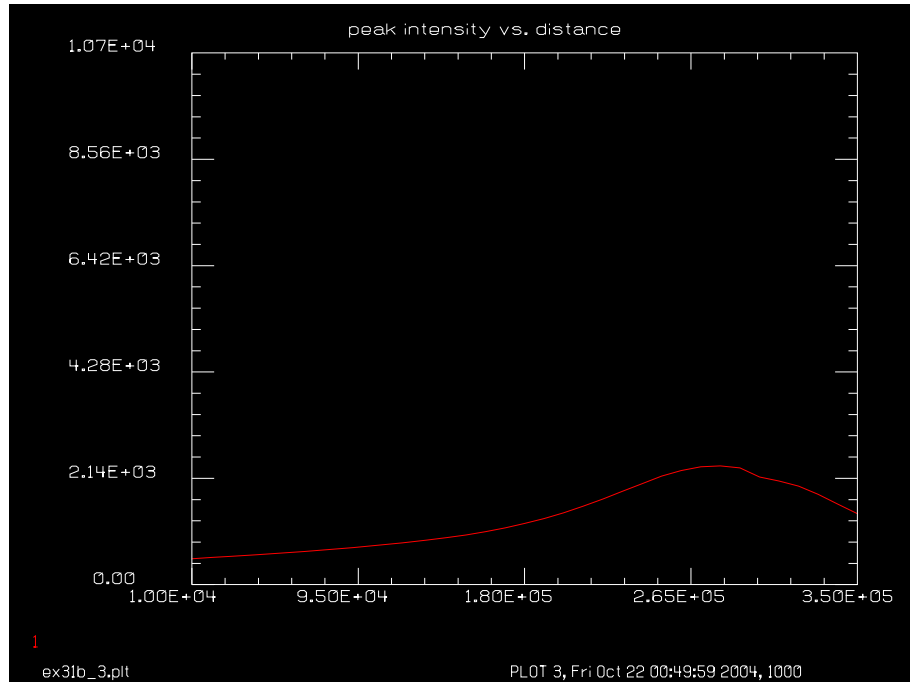


Fig. 31.4. Plot of the peak intensity as a function of position. Without thermal blooming the beam would peak at $z = 4$. With blooming, the peak occurs at about $z = 2.8$ km and the beam diverges after that point.

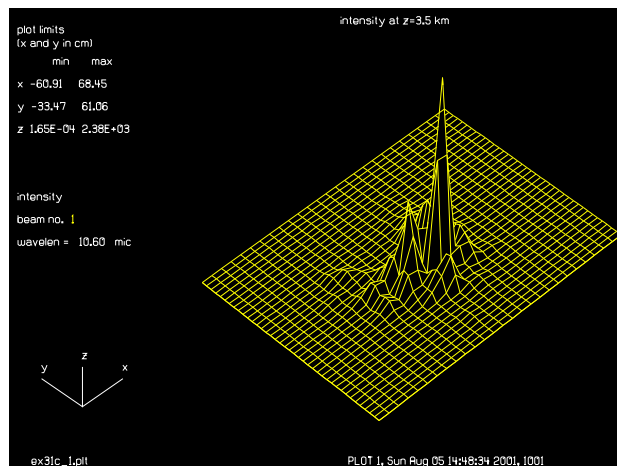


Fig. 31.5. Far-field pattern at 3.5 km after thermal blooming and atmospheric aberration, assuming $r_0 = 10$ cm. The peak intensity is slightly higher than Fig. 31.3.

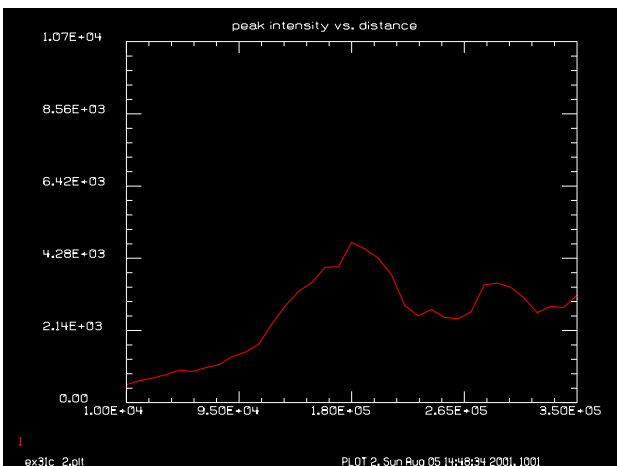


Fig. 31.6. Plot of peak irradiance versus position with thermal blooming and atmospheric aberration. The atmospheric aberration can increase or decrease the peak irradiance. In this case there is a slight improvement at the 2.5 km point.

```
set/window/abs -60 60 -60 60 # set plot window
title no thermal blooming or aberration
plot
end
```

Jump to: [Commands](#), [Theory](#)

Ex31b: Thermal blooming, no kinetic cooling**Input: ex31b.inp**

```

echo/on
c## ex31b
c
c Example 31b: Theramal Blooming, no Kinetic Cooling
c
c This example shows a simple thermal blooming case without
c atmospheric aberration.
c
c The target is moving at v = (4,-4) m/sec, range = 4 km.
c The wind is moving at v = (-10,-10) m/sec. The
c effect will be strongest when wind and targer movement are
c at the same velocity. The effective target velocity changes
c at the beam tracks the target.
c
c The thermal blooming causes the beam to diverge at about
c 2.5 km rather than come to a focus at 4 km where the beam
c if focused.
c
variab/dec/int pass
write/d ex31b.out/o          # write to output file
target/set 4. -4. 4          # v = (4,-4) m/sec, range = 4 km.
units/s 1 2.5
gaus/c/c 1 500 25.           # set gaussian beam to 500 w/cm**2 peak
energy                       # list energy
lens 1 4e5                    # focus beam to 4 km.
bloom/set x=-10 y=-10 tn=0    # wind is (-10,-10) m/sec, cooling off.
bloom/altitude/set 1          # set altitude to 1 km
pass = 0                      # initialize step counter
z = 0.                        # initialize total length
zstep = 1e4                   # set step length
macro/def blmstep/o          # define macro for steps
    write/screen/on
    pass = pass + 1           # increment step counter
    write/screen/off
    z = z + zstep             # increment total length
    bloom/prop 1 zstep 1      # blooming propagation
    variab/set peak 1 peak    # store peak
    udata/set pass z peak     # set User data
    if pass=10 then
        plot/watch ex31b_1.plt
        plot/i/phase
    endif
macro/end                     # end of macro
macro/run blmstep/35          # execute macro 35 times
title intensity at z=4.0 km
plot/watch ex31b_2.plt        # define plot file
set/density 32                 # set plot density
set/window/abs -60 60 -60 60  # set plot window
plot                           # ploy intensity isometric

```

Jump to: [Commands](#), [Theory](#)


```

title peak intensity vs. distance
plot/watch ex31b_3.plt
plot/udata 1 1 min=0 max=1.07e4 # plot User data
end

```

Ex31c: Thermal blooming with kinetic cooling and aberration

Input: ex31c.inp

```

c## ex31c!750182376588420
c
c Example 31c: Thermal Blooming with Kinetic Cooling and Aberration.
c
c This example shows a simple thermal blooming case with
c atmospheric aberration with r0 = 10 cm accomplished in
c 40 steps of r0=92 cm for each step.
c
c The target is moving at v = (4,-4) m/sec, range = 4 km.
c The wind is initially moving at v = (-10,-10) m/sec and after
c 2 km is changed to v = (-10,10).
c
c The effect will be strongest when wind and target movement are
c at the same velocity. Effective target velocity changes
c at the beam approaches the target assuming the beam is
c tracking the target.
c
c The thermal blooming causes the beam to diverge at about
c 2.5 km rather than come to a focus at 4 km where the beam
c is focused.
c
variab/dec/int pass
target/set 4. -4. 4 # v = (4,-4) m/sec, range = 4 km.
units 1 2.5
gaus/c/c 1 500 25. # set gaussian beam to 500 w/cm**2 peak
energy # list energy
lens 1 4e5 # focus beam to 4 km.
bloom/set x=-10 y=-10 tn=0 # wind is (-10,-10) m/sec, cooling off.
bloom/altitude/set 1 # set altitude to 1 km
pass = 0 # initialize step counter
z = 0. # initialize total length
zstep = 1e4 # set step length
macro/def blmstep/o # define macro for steps
    write/screen/on
    pass = pass + 1 # increment step counter
    write/screen/off
    z = z + zstep # increment total length
    phase/random/kolmog 1 92 pass # atmospheric aberration, r0 = 184 cm.
    bloom/prop 1 zstep 1 # blooming propagation
    variab/set peak 1 peak # store peak
    udata/set pass z peak # set User data
macro/end # end of macro
macro/run blmstep/20 # execute macro 20 times
bloom/set x=-10 y=10 # change wind direction
macro/run blmstep/15 # execute macro 15 more times
title intensity at z=3.5 km
plot/watch ex31c_1.plt # define plot file
set/density 20 # set plot density

```

Jump to: [Commands](#), [Theory](#)

```
set/window/abs -60 60 -60 60    # set plot window
plot                               # ploy intensity isometric
title peak intensity vs. distance
plot/watch ex31c_2.plt
plot/udata 1 1 min=0 max=1.07e4 # plot User data
end
```

Ex32: Phase conjugate mirror or correcting aberration plate

Table. 32.1. Table of Ex32 examples

Ex32a: Conjugate mirror in near-field.	1
Ex32b: Conjugating mirror or aberration plate in either near- or far-field	3

Ex32a: Conjugate mirror in near-field

Example 32a illustrates a phase conjugate mirror used in the near-field. The phase conjugate mirror reverses the direction of propagation and the phase. In this example aberration is imposed on a 20 cm radius beam which is then brought to a focus at a 100 cm focal length lens. At the focus, a phase conjugate mirror reverses the phase effects of the original aberration and propagation. After propagation along the reverse path and imposing the same aberration, the original phase and intensity distributions are recovered.

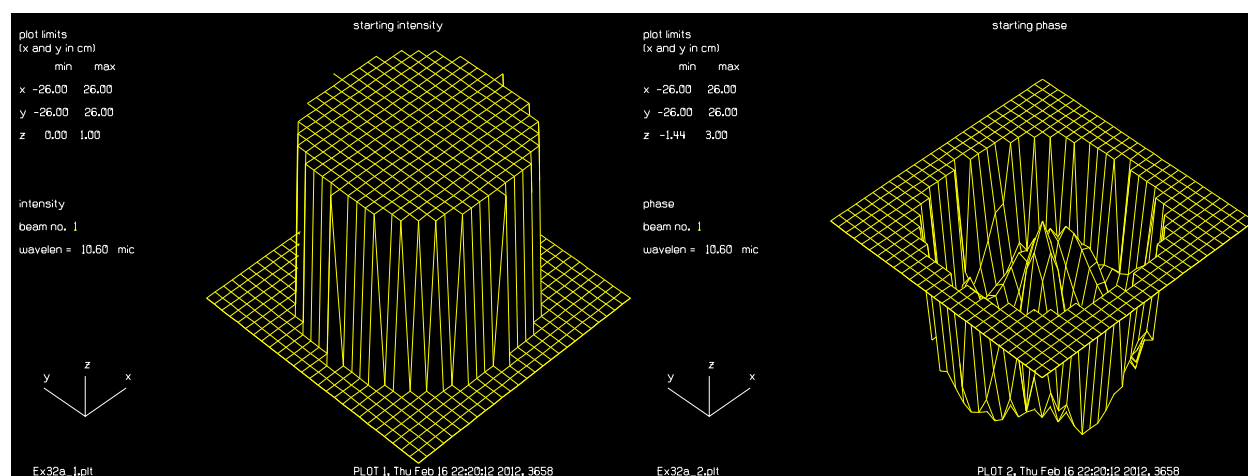


Fig. 32.1. Intensity (left) and phase (right) of a beam with Zernike polynomial aberrations. The wavefront error is 0.135 waves of aberration.

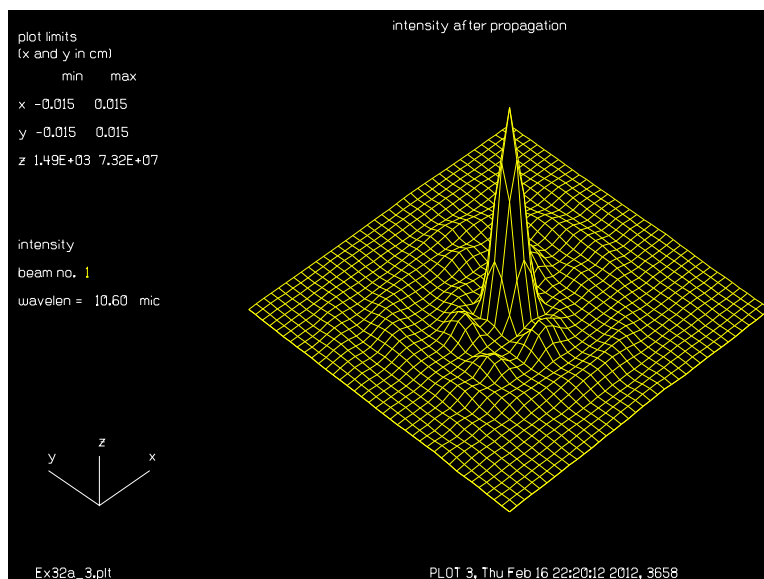


Fig. 32.2. Intensity of the beam after propagation to the focus.

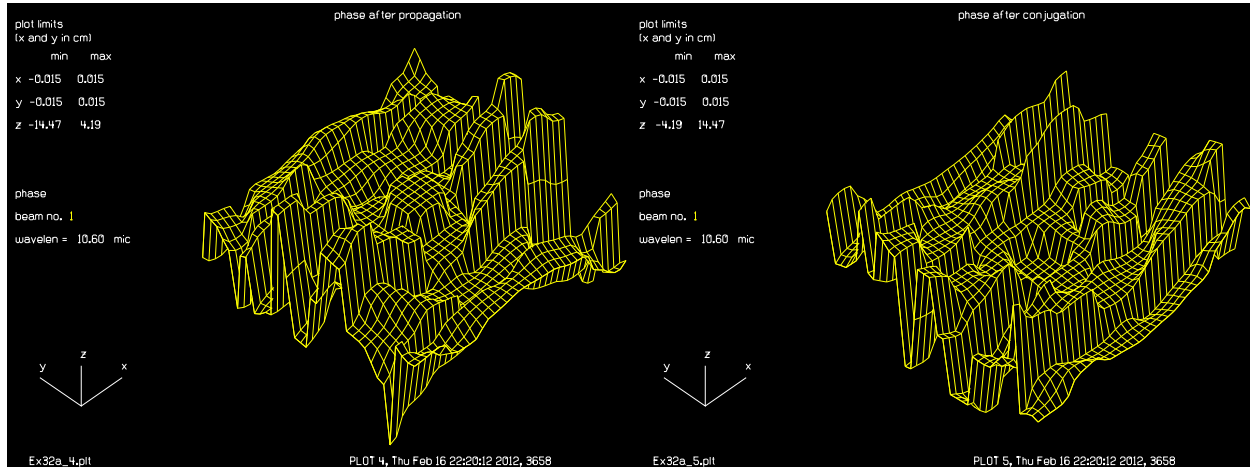


Fig. 32.3. Phase after propagation to the far-field: before conjugation (left) and after conjugation (right).

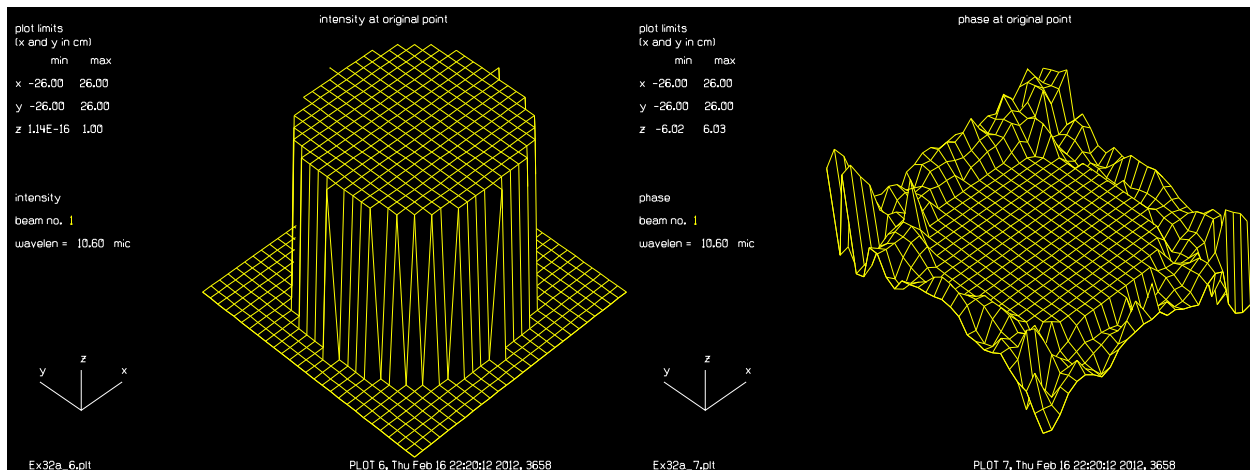


Fig. 32.4. Intensity (left) and phase (right) of the beam after return to original point and addition of the original aberration. The phase excursions occur only well outside the intensity distribution and are due to numerical accuracy in the Fourier transforms. The residual wavefront error is 0.0001 waves.

Input: ex32a.inp

```

c## ex32a
c
c Example 32a: Phase Conjugate Mirror
c
c This is an example of a phase conjugate mirror. The phase conjugate mirror
c reverses the direction of propagation and the phase. In this example
c aberration is imposed on a 20 cm radius beam which is then propagated to
c the focus of a lens. At the focus a phase conjugate mirror reverses the
c phase effects of the original aberration and propagation. Then propagation
c along the reverse path and imposing the same aberration recreates
c original phase and intensity distributions.
c
alias Name Ex32a
echo/on                                # turn on echo
units/s 1 2                            # set units
clap/c/c 1 20                          # define 40 cm diameter aperture

```

Jump to: [Commands](#), [Theory](#)

```

z = 1./-2/pi
abr/zern/rad 1 8 1.5*z           # impose Zernike aberrations
abr/zern/rad 1 10 -1.5*z
abr/zern/cos 1 5 5 z
abr/zern/sin 1 5 3 -1.5*z
str
set/density 32                   # set plot density
title starting intensity
set/window/abs -25 25 -25 25
plot/watch @Name_1.plt
plot/iso                         # Plot 1: starting intensity
title starting phase
plot/watch @Name_2.plt          # Plot 2: starting phase
plot/iso/phase
lens 1 100
dist 100                         # propagate to focus
title intensity after propagation
set/window/abs -.015 .015 -.015 .015
plot/watch @Name_3.plt
plot/iso                         # Plot 3: intensity after propagation
title phase after propagation
plot/watch @Name_4.plt
plot/iso/phase                  # Plot 4: phase after propagation
conjug 1                         # phase conjugate mirror
title phase after conjugation
plot/watch @Name_5.plt
plot/iso/phase                  # Plot 5: phase after conjugation
dist -100                       # propagate in the reverse direction
lens 1 100
abr/zern/rad 1 8 1.5*z           # impose original aberration again
abr/zern/rad 1 10 -1.5*z
abr/zern/cos 1 5 5 z
abr/zern/sin 1 5 3 -1.5*z
title intensity at original point
set/window/abs -25 25 -25 25
plot/watch @Name_6.plt
plot/iso                        # Plot 6: intensity at original point
title phase at original point
plot/watch @Name_7.plt
plot/iso/phase                  # Plot 7: phase at original point
str

```

Ex32b: Conjugating mirror or aberration plate in either near- or far-field

Example 32b illustrates modeling of either a conjugate mirror or a conjugating aberration plate in the near- or far-field. In the far-field GLAD uses a curved reference surface. For the purpose of conjugation, the reference surface may temporarily be set to be plane, the conjugation operation applied, and then the bias radius restored. An aberration plate is treated by converting the wavefront to a real value and multiplying by -2 with “mult/complex 3 -2”. A real aberration plate that may have imperfections may be substituted for the multiplication step. For the aberration plate model the diffraction is not reversed although “mirror/flat” may be added.

Input: ex32b.inp

```

c## Ex32b, Conjugation mirror or plate in near- or far-field.
c

```

Jump to: [Commands](#), [Theory](#)

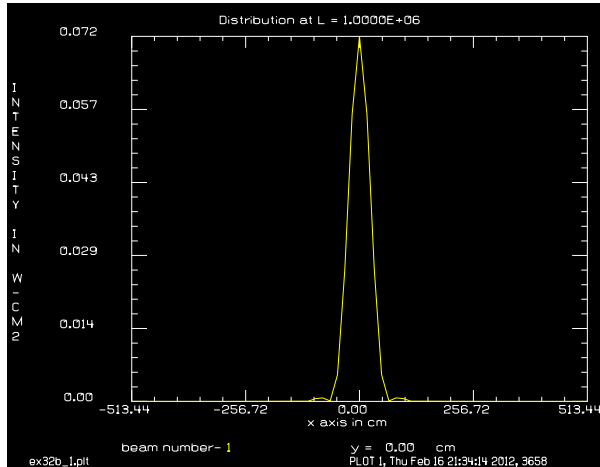
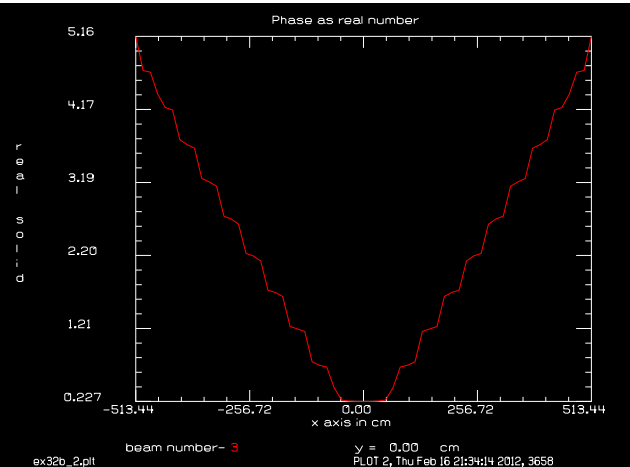
Fig. 32.5. Far-field intensity for $L = 1e6$.

Fig. 32.6. Phase at point of conjugation.

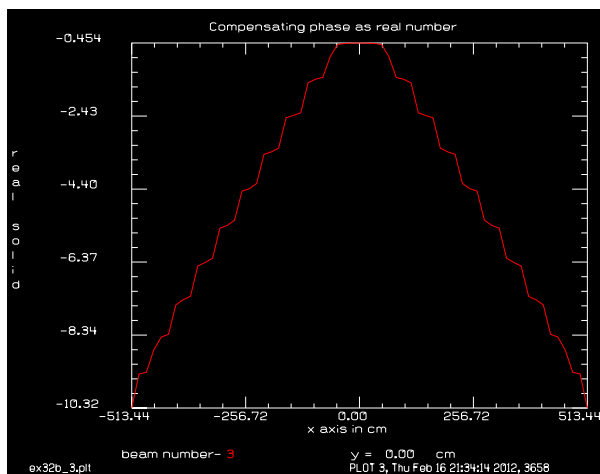


Fig. 32.7. Phase after conjugation.

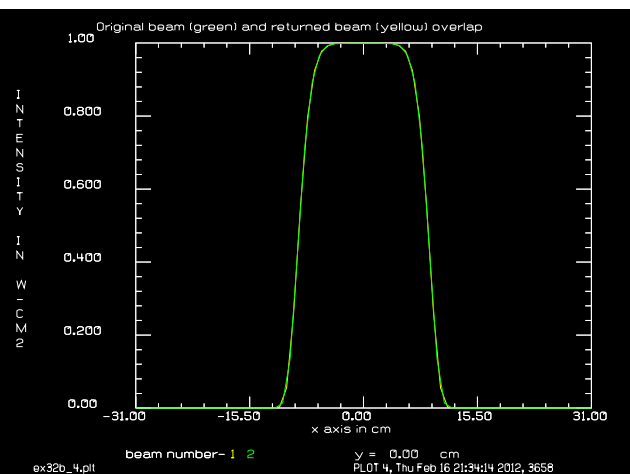


Fig. 32.8. Initial supergaussian distribution (green) and distribution after conjugation (yellow) match closely.

c Conjugation in near- or far-field. In the far-field, the bias radius c is removed with an ideal lens, conjugation is performed, and the bias c radius is restored.

c

c If `CONJUGATE_MIRROR = 1`, the conjugate mirror command is used.

c If `CONJUGATE_MIRROR = 0`, a correction plate is simulated

c

c The conjugating correction plate is simulated by "mult/complex 3 -2".

c This reverses the wavefront sign.

c

c If the beam is initially in the far-field, the bias reference surface c should first be removed, wavefront conjugation performed, and the bias c reference surface restored.

c

c The steps for `CONJUGATE_MIRROR=0` are:

c 1) If the array is in the far-field, remove reference bias radius.

c 2) The wavefront is transformed to a real value.

c 3) The real values are multiplied by -2 to conjugate the wavefront.

c 4) The real values are transformed back to wavefront form.

Jump to: [Commands](#), [Theory](#)

```

c 5) If the array was initially in the far-field, the bias radius is restored.
c
c Note the distance is advanced for this case.
c
c The steps for CONJUGATE_MIRROR=1 are:
c 1) If the array is in the far-field, remove reference bias radius.
c 2) The "conjugate" command is applied to stimulate a conjugate mirror.
c 3) If the array was initially in the far-field, the bias radius is restored.
c
c Note the beam returns to the initial position for this case.
c
c The beam is restored to the initial profile.
c
alias Name ex32b
variable/dec/int Iplane CONJUGATE_MIRROR
c
CONJUGATE_MIRROR = 0 # set to 0 or 1 for aberration plate or mirror

c set to short or long path
c try L=1e3 for near-field and L=1e6 for far-field
L = 1e6 # near-field or far-field
gaus/c/c 1 1 10 4
nbeam 3 data
copy/c 1 2
prop L
plot/w @Name_1.plt
title Distribution at L = @L
plot/x/i 1
C C starting surrogate beam values
geodata 1
c iplanx yields 1 for near-field (plane reference) or
c 0 for far-field (curved reference)
variable/set Iplane 1 geodata/iplanx list
c
c ----- Construct conjugate -----
c
if [Iplane==0] then
  c Add a lens to put beam temporarily in near-field if necessary
  variable/set Radius 1 geodata/zdistx
  lens 1 Radius
endif
if CONJUGATE_MIRROR=0 then
  copy 1 3
  c Convert wavefront to real values
  waves2real 3
  plot/w @Name_2.plt
  title Phase as real number
  plot/x/r 3
  c
  c Make phase compensation by multiplying by -2
  c
  mult/complex 3 -2. # simulated an ideal conjugating phase plate.
  plot/w @Name_3.plt
  title Compensating phase as real number
  plot/x/r 3
  c
  c Convert real values to wavefront.
  c
  real2waves 3 1.

```

```
    mult/beam 1 3
else
    conjugate 1
endif
if [Iplane==0] then
    c Add a lens to return beam to far-field if necessary
    lens 1 Radius
endif
c ----- End construction of conjugate phase -----
prop L
C C return to starting position
geodata 1
plot/w @Name_4.plt
title Original beam (green) and returned beam (yellow) overlap
plot/x/i fi=1 la=2
L=
CONJUGATE_MIRROR=
zreff
```


Ex33: Stable resonator

Table. 33.1. Table of Ex33 examples

Ex33a: Half-symmetric stable cavity.	1
Ex33b: Half-symmetric stable cavity with 1:1 intercavity telescope, ideal lenses	4
Ex33c: Half-symmetric stable cavity with 1:1 intercavity telescope, lensgroup lenses	6
Ex33d: Analysis of polygon resonator.	8
Ex33e1: Coherent injection, decentered input, part 1	9
Ex33e2: Coherent injection, decentered input, part 2	10
Ex33f: Global definition of half-symmetric resonator	12
Ex33g: Example of a wire obscuration to induce TEM ₁₀	14
Ex33h: Example of half-symmetric resonator with rotating end mirror	14
Ex33i: Two-wavelength flat-flat resonator	14
Ex33j: Multiple beams through the same resonator	14
Ex33k: Extended cavity and spurious reflections	14
Ex33l: Coupled resonator	15
Ex33m: Finding multiple modes by orthogonality	22
Ex33n: Bowtie resonator	22

Ex33a: Half-symmetric stable cavity

This example is a stable resonator with circular mirrors. The configuration consists of a flat mirror and a concave spherical mirror of radius 50 cm. The mirrors are separated by 45 cm. The parameters are summarized below.

Table. 33.2. Parameters

length	45 cm
mirror radius	50 cm
wavelength	1.064 microns
Rayleigh range	15 cm

It is important to establish a geometrically stable resonator before attempting to calculate the diffraction performance. A surrogate gaussian beam is used to determine the switching of the propagation algorithms. If the surrogate gaussian is not set to the geometric resonant condition, the parameters will oscillate without damping. In principal the diffractive performance would be unaffected. Because of sampling limitations an oscillation in the selection of propagation algorithms will perturb the diffractive mode significantly preventing convergence. .

The resonator command may be used to calculate the paraxial stability and eigenmode properties the resonator. The `resonator/set` command is used to set the surrogate beam to be an eigenmode of the resonator and to establish the complex amplitude to be the eigenmode value as well. Having the surrogate beam be an eigenmode of the resonator assures stable numerical algorithms and the eigenmode also serves as a good starting point for the complex amplitude.

Beam 1 is kept as the eigenmode. However, Beam 2 is reinitializes to be a flat-top function to observe convergence from a non-optimum starting point. Beam 1 converges almost immediately to 0.09 percent loss per pass for the aperture radius of 0.14 cm Beam 2 takes about 90 passes to converge to within 0.1 percent.

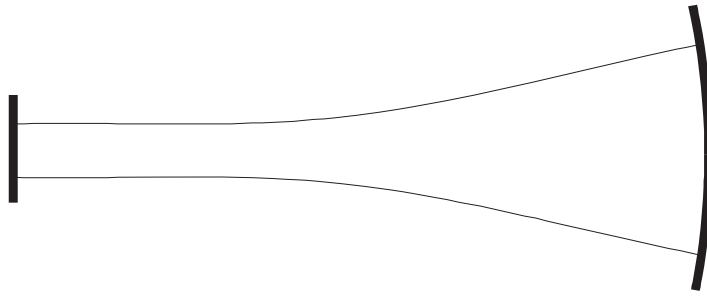


Fig. 33.1. Stable resonator configuration. The waist will form on the flat mirror and the phase radius will match the radius of the concave mirror in the ideal geometric mode.

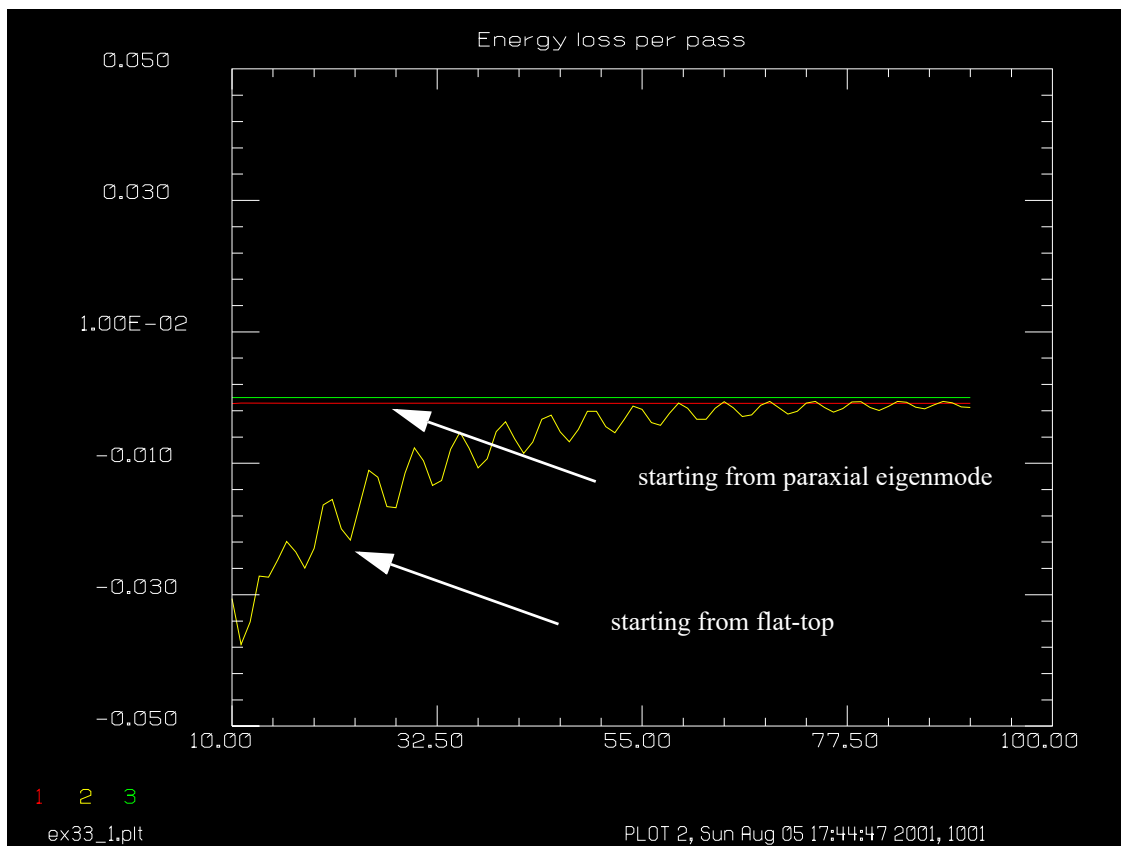


Fig. 33.2. Plot of energy loss per pass as a function of the pass number. The plot is from Pass 10 to Pass 91. The lower horizontal line is actually the loss for Beam 1 which has already converged from the geometric mode with in 10 passes. The oscillating curve shows the convergence of Beam 2 from the initial uniform intensity.

Input: `ex33a.inp`

```
c## ex33a
c
c Example 33a
c
```

Jump to: [Commands](#), [Theory](#)

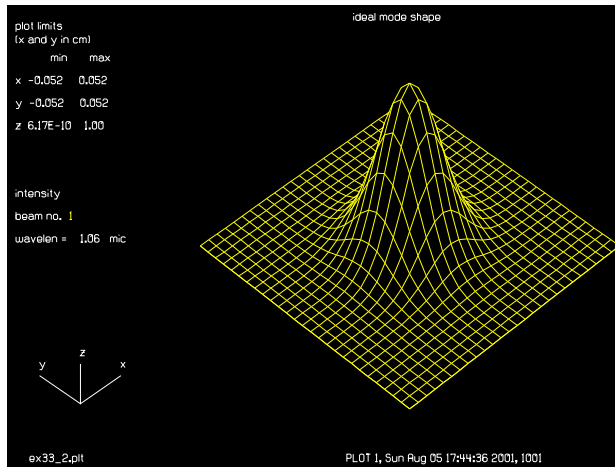


Fig. 33.3. Ideal resonator mode.

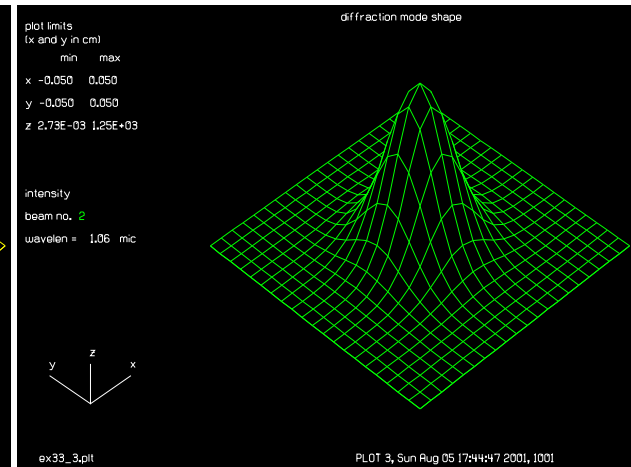


Fig. 33.4. Converged diffractive mode.

```

c Bare Cavity Stable Resonator
c
c The configuration consists of a flat mirror and a spherical mirror.
c Surrogate gaussian beam is initially set-up to exactly match the
c stable geometric mode. Beam 1 is left with the ideal mode and
c Beam 2 is modified to have a uniform intensity with the CLEAR
c command which does not change the surrogate gaussian beam
c parameters.
c
c The waist of the ideal resonator mode is w0=.02253936. At
c the curved mirror the beam radius is w=.07128. The mirror aperture
c is selected to be twice the beam radius at the mirror at .14 cm.
c
c The steady-state mode depends on the aperture losses. Both Beam 1
c and Beam 2 are affected by the aperture and converge to the same
c steady-state solution. Beam 1 converges quickly because the starting
c condition is the ideal geometric mode which is very close to the
c the steady-state mode. Beam 2 starts from uniform intensity and
c takes about 100 iterations to converge.
c
c
variab/dec/int pass STOP
macro/def reson/o
    pass = pass + 1 list          # increment pass counter
    step = step + 1              # increment step number
    prop 45                      # propagate 45 cm.
    mirror/sph 0 -50             # mirror of 50 cm. radius
    clap/c/n 0 .14               # .14 cm. radius aperture
    prop 45                      # propagate 45 cm. along beam
    mirror/sph 0 1.e15           # flat mirror
    energy                      # calculate energy in the beams
    variab/set energy1 1 energy
    variab/set energy2 2 energy
    energy1 = energy1 - 1        # calculate energy difference
    energy2 = energy2 - 1
    udata/set pass step energy1 energy2 # store energy differences

```

Jump to: [Commands](#), [Theory](#)

```

    gain/converge/test ibeams=2 nstore=STOP      # test for convergence
    if STOP macro/exit                          # exit from macro on convergence
    energy/norm 1 1                             # renormalize energy
    energy/norm 2 1
macro/end
nbeam 2                                         # establish 2 beams
wavelength/set 0 1.064                         # set wavelengths
units/s 0 .005
resonator/name reson
resonator/eigen/test 2
resonator/eigen/set 2                         # set beam 2 to eigen mode
gaus/c/c 1 1 .02253936                       # set beam 1 to ideal mode
clear 2 1                                     # start with a plane wave in beam 2
title ideal mode shape
set/density 32
set/window/abs -.05 .05 -.05 .05
plot/watch ex33_2.plt
plot first=1 last=1
energy/norm 1 1                               # normalize energies
energy/norm 2 1
status/p
pass = 0                                       # initialize variables
step = 0                                      # for pass counters
gain/converge/set eps1=.001 npoints=5        # set convergence criterion to
                                              # .1 percent energy change

reson/run 100
title Energy loss per pass
plot/watch ex33_1.plt
plot/udata first=1 last=3 left=10 right=100 min=-.05 max=.05
title diffraction mode shape
plot/watch ex33_3.plt
set/window/abs -.05 .05 -.05 .05
plot/iso first=2 last=2

```

Ex33b: Half-symmetric stable cavity with 1:1 intercavity telescope, ideal lenses

Input: `ex33b.inp`

Ex33c: c## ex33b

```

c
c This is an example of a half-symmetric resonator with a
c 1:1 intercavity telescope.
c
c The effect of the 1:1 telescope is to reduce the cavity length
c by four times the length = 10.16 cm
c
variab/dec/int pass Equivalent Test
Equivalent = 1          # select equivalent system
F=2.54

```

Jump to: [Commands](#), [Theory](#)

```

macro/def reson/o
  pass = pass + 1 list          # increment pass counter
  mirror/flat 1
  if Equivalent then
    prop 492-4*F
  else                          # 65% efficiency
    mult 1 1 .65
    lens 1 F
    prop 2*F
    lens 1 F
    prop 492-2*F
  endif
  mirror/sph 1 1500            # mirror of 15 m radius
  clap/c/n 1 .625              # aperture dia. 12.5
  if Equivalent then
    prop 492-4*F
  else
    mult 1 .95                  # 95% efficiency
    prop 492-2*F
    lens 1 F
    prop 2*F
    lens 1 F
  endif
  if [!Test] then
    variab/set Energy 1 energy
    Energy = Energy - 1 list    # calculate energy difference
    count1 = count1 + 1
    udata/set count1 pass Energy # store energy differences
    title flat, pass @pass
    plot/w ex33b_1.plt
    plot/udata
    plot/w ex33b_2.plt
    plot/l 1 xrad=1.5
  endif
  energy/norm 1 1
  clap/c/n 1 .575              # aperture dia. 11.5 mm
macro/end
array/s 1 256
wavelength/set 0 10.6          # set wavelengths
units/field 1 3.0
Test = 1
resonator/name reson
resonator/eigen/test 1

```

```

Test = 0
# start with noise to seed all possible modes
clear 1 0          # start with a plane wave in beam 2
noise 1 1
convol/haus 1 .05    # smooth just a bit
clap/c/n 1 .575      # aperture dia. 11.5 mm
title initial distribution
plot/w ex33b_2.plt
plot/l 1 xrad=1.5
reson/run 20
Half-symmetric stable cavity with 1:1 intercavity telescope, lensgroup lenses

```

Input: ex33c.inp

Ex33d: c## ex33c

```

c
c This is an example of a half-symmetric resonator with a
c 1:1 intercavity telescope using lensgroup
c
variab/dec/int pass Equivalent Test
F=25.4
Rindex = 2.403083 # irtran4 at 10.6 microns
R1_1 = 40
R1_2 = 1./(1./R1_1 - 1./F/(Rindex-1))
T1=.1
R2_1 = -1000
T2=.1
R2_2 = 1./(1./R2_1 - 1./F/(Rindex-1))
variab
lensgroup/def ex33b_1f/o
  object
  surface_lensgroup radius=R1_1 thick=T1 irtran4
  surface_lensgroup radius=R1_2 thick=.0 air
  image focus
lensgroup/end
lensgroup/def ex33b_2f/o
  object
  surface_lensgroup radius=R2_1 thick=T2 irtran4
  surface_lensgroup radius=R2_2 thick=.0 air
  image focus
lensgroup/end
lensgroup/def ex33b_2b/o

```

Jump to: [Commands](#), [Theory](#)

```

object
surface_lensgroup radius=-R2_2 thick=T2 irtran4
surface_lensgroup radius=-R2_1 thick=.0 air
image focus
lensgroup/end
lensgroup/def ex33b_1b/o
  object
  surface_lensgroup radius=-R1_2 thick=T1 irtran4
  surface_lensgroup radius=-R1_1 thick=.0 air
  image focus
lensgroup/end
macro/def reson/o
  pass = pass + 1 list          # increment pass counter
  mirror/flat 1
  mult 1 .65
  vertex/locate/abs 0 0 0
  vertex/rotate/add 180 0 0
  lensgroup/run/radial ex33b_2b
  # lens 1 F
  vertex/locate/abs 0 0 -(2*F)
  vertex/rotate/add 180 0 0
  # lens 1 F
  lensgroup/run/radial ex33b_1b
  vertex/locate/abs 0 0 -(492-4*F)
  prop/vertex
  mirror/sph 1 1500             # mirror of 15 m radius
  clap/c/n 1 .625              # aperture dia. 12.5
  mult 1 .95                   # 95% efficiency
  vertex/locate/abs 0 0 -(2*F)
  lensgroup/run/radial ex33b_1f
  # lens 1 F
  vertex/locate/abs 0 0 -T1
  lensgroup/run/radial ex33b_2f
  vertex/locate/abs 0 0 0
  vertex/rotate/add 0 0 0
  prop/vertex
  # lens 1 F
  if [!Test] then
    variab/set Energy 1 energy
    Energy = Energy - 1 list     # calculate energy difference
    count1 = count1 + 1
    udata/set count1 pass Energy # store energy differences
    title flat, pass @pass

```

```

    plot/w ex33c_1.plt
    plot/udata
    plot/w ex33c_2.plt
    plot/l 1 1 xrad=1.5
endif
energy/norm 1 1
clap/c/n 1 .575          # aperture dia. 11.5 mm
macro/end
array/s 1 256
wavelength/set 0 10.6    # set wavelengths
units/field 1 3.0
Test = 1
resonator/name reson
resonator/eigen/test 1
Test = 0
# start with noise to seed all possible modes
clear 1 0                # start with a plane wave in beam 2
noise 1 1
convol/haus 1 .05        # smooth just a bit
clap/c/n 1 .575          # aperture dia. 11.5 mm
title initial distribution
plot/w ex33c_2.plt
plot/l 1 1 xrad=1.5
reson/run 20
Analysis of polygon resonator

```

Input: ex33d.inp

```

c## ex33d
#
# Analysis of polygon resonator.
#
# The beam reflects off the interior faces of a polygon.
# With flat surfaces, the resonator is on the boundary
# between stability and instability. Slight optical power
# is added by including a phase lens at each surface.
#
mem/set/b 2
variab/dec/int Array Pass Nsides SideNo
Pass = 0
MirrorApertur = 0.2      # aperture radius at each phase
FocalLength= 5000        # weak converging power
LengthPerSide= 20        # propagation length between surface
                        # intercepts

Array = 256
Field = .75
WaveL = 1.064e-4

```

Jump to: [Commands](#), [Theory](#)


```

array/set 1 Array
units/field 0 Field
wavelength/set 0 WaveL*1e4
Nsides = 8
Theta = 2.*pi/Nsides
macro/def side/o
    SideNo = SideNo + 1      # increment side number counter
    prop LengthPerSide      # propagate length between sides
    vertex/locate/rel       # relocate vertex to current beam position
    vertex/rotate/set ry=(SideNo-.5)*360/Nsides
    mirror/global/flat 1    # global flat mirror
    clap/c/n 1 MirrorApertur
c phase lens adds effect of slightly curved mirror
    lens/toric/ele/phase 1 FocalLength/cos(Theta) FocalLength
    zreff/set 1 0.
macro/end

macro/def reson/o
    Pass = Pass + 1          # increment pass counter
    SideNo = 0              # initialize side number counter
    macro side/Nsides        # call side macro
    plot/w ex33d_1.plt
    plot/l 1
macro/end
reson/name reson
reson/eigen/test 1
reson/eigen/set 1
clear 1 0
noise 1 1
reson/run 100

```

Ex33e1: Coherent injection, decentered input, part 1

We want to inject a decentered beam into the cavity and observe the interference pattern built up in the cavity mode. For good cavity buildup it is necessary to tune the resonator length by a fraction of a wavelength so the lowest order gaussian transverse mode is exactly on the input wavelength. We do this by finding the Gouy shift numerically and cancelling it each pass. This is equivalent to making a fine adjustment of frequency. Fig. 33.5 illustrates the position of the beam after the final pass. In ex33e1 we test the beam position vs. pass number. The center of the beam is determined by `fitxgeo`, as shown in Fig. 33.6.

Input: ex33e1.inp

```

c## ex33e1
#
# Example to show interference fringes for coherent injection of
# an offset beam into a stable resonator.
#
# We want to inject a decentered beam into the cavity and observe
# the interference pattern build up. For good cavity buildup it is
# necessary to tune the resonator length by a fraction of a wavelength
# so the lowest order gaussian transverse mode is exactly on the input

```

Jump to: [Commands](#), [Theory](#)

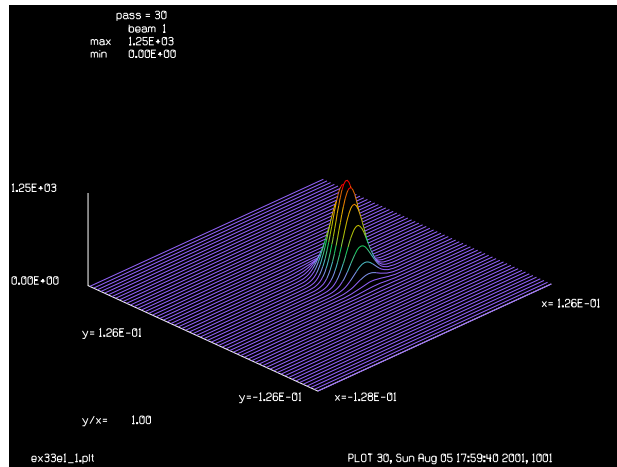


Fig. 33.5. Decentered signal injected into resonator. The beam oscillates left and right.

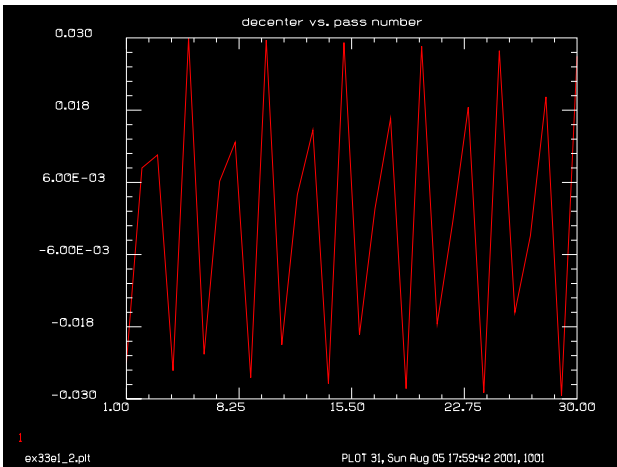


Fig. 33.6. Locus of center of beam as it oscillates left and right.

```
# wavelength. We do this by finding the Gouy shift numerically and
# cancelling it each pass. This is equivalent to making a fine adjustment
# of frequency.
```

```
#
# In ex33e1 we test the beam position vs. pass number.
#
```

```
variab/dec/int pass TEST
```

```
macro/def reson/o
```

```
    pass = pass + 1 list
```

```
    prop 45
```

```
    mirror 1 rad=-50
```

```
    prop 45
```

```
    mirror/flat 1
```

```
    fitgeo 1
```

```
    variab/set Xcen fitxcen
```

```
    udata/set pass pass Xcen
```

```
    if [!TEST] then
```

```
        title pass = @pass
```

```
        plot/l 1 ns=64
```

```
        pause 2
```

```
    endif
```

```
macro/end
```

```
array/set 1 128
```

```
wavelength/set 0 1.064
```

```
units/set 0 .002
```

```
gaussian/cir/con 1 1 .025 1 # decx=.03 # set gaussian beam with off-axis
```

```
set/density 32
```

```
wid=.05
```

```
set/window/abs -wid wid -wid wid
```

```
plot/w ex33e1_1.plt
```

```
TEST = 1
```

```
resonator/name reson
```

```
resonator/eigen/test 1
```

```
resonator/eigen/set 1
```

```
energy/norm 1 1
```

```
# increment pass counter
```

```
# mirror of 50 cm. radius
```

```
# flat mirror
```

```
# record decenter
```

```
# store decenter in udata
```

```
# set array size
```

```
# set wavelength
```

```
# set gaussian beam with off-axis
```

```
# make a plot at each pass
```

```
# set width of plots
```

```
# set name of resonator macro
```

```
# find resonator properties
```

```
# set surrogate beam to eigen mode
```

Jump to: [Commands](#), [Theory](#)

```

Xdec = .03
rescale/shift 1 x=Xdec          # start with beam off-set
TEST = 0
pass = 0
reson/run 30
plot/w ex33e1_2.plt
title decenter vs. pass number
plot/udata min=-Xdec max=Xdec

```

Ex33e2: Coherent injection, decentered input, part 2

In Part 2 of ex33e, we tune the resonator for coherent injection by finding the Gouy shift and compensating for it. This is equivalent to making a fine tuning of the length of the resonator so that the frequency of the lowest loss transverse mode exactly coincides with the injected frequency. Beam 1 is the cavity mode where we see the build up of the interference pattern and Beam 2 is the injected beam. See Fig. 33.7.

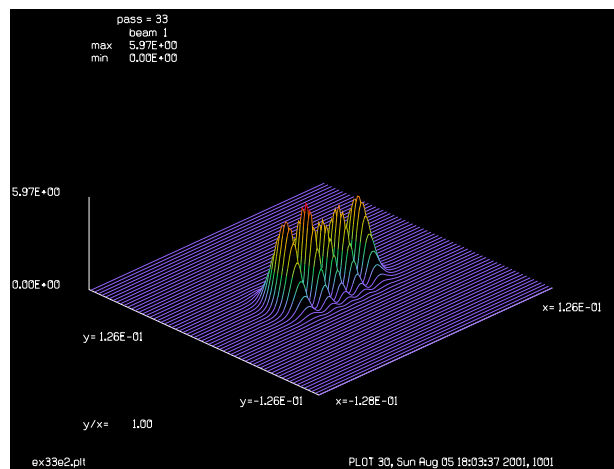


Fig. 33.7. Interference pattern of cavity beam formed with decentered input.

Input: ex33e2.inp

```

c## ex33e2
#
# Example to show interference fringes for coherent injection of
# an offset beam into a stable resonator.
#
# We want to inject a decentered beam into the cavity and observe
# the interference pattern build up. For good cavity buildup it is
# necessary to tune the resonator length by a fraction of a wavelength
# so the lowest order gaussian transverse mode is exactly on the input
# wavelength. We do this by finding the Gouy shift numerically and
# cancelling it each pass. This is equivalent to making a fine adjustment
# of frequency.
#
# It is important to assign some efficiency factor for partially
# reflecting mirror output so the cavity field builds up to a steady-state
# value. More interference effects will be seen if the efficiency is

```

Jump to: [Commands](#), [Theory](#)

```

# higher but convergence is lower.
#
# Beams
# 1      cavity mode (builds up over many passes)
# 2      injected beam
#
# Steps in the analysis
# 1) Find eigen value of resonator
# 2) Find Gouy shift and compensate (equivalent to tuning frequency exactly)
# 3) Injeject signal and coherently add to cavity mode each pass.
#
variab/dec/int pass TEST
macro/def reson/o
    pass = pass + 1 list          # increment pass counter
    phase/piston 1 GouyPhase
    if [!TEST] add/coh 1 2        # add input to cavity mode
    prop 45
    mirror 1 rad=-50              # mirror of 50 cm. radius
c   clap/c/n 1 .14               # .14 cm. radius aperture
    prop 45
    mult 1 Efficiency
    mirror/flat 1                 # flat mirror
    variab/set Energy 1 energy    # set variable to energy value
    udata/set pass pass Energy-1 # store energy differences
    if [!TEST] then
        title pass = @pass
        plot/l 1 ns=64
        pause 2
    endif
macro/end
array/set 1 128                  # set array size
nbeam 2 data
wavelength/set 0 1.064           # set wavelength
units/set 0 .002
gaussian/cir/con 1 1 .025 1 # decx=.03 # set gaussian beam with off-axis
set/density 32                   # make a plot at each pass
wid=.05                          # set width of plots
set/window/abs -wid wid -wid wid
TEST = 1
GouyPhase = 0
Efficiency = .9
resonator/name reson             # set name of resonator macro
resonator/eigen/test 1           # find resonator properties
resonator/eigen/set 1            # set surrogate beam to eigen mode

# Find Gouy shift by checking correlation change after one round trip

copy 1 2
reson/run 1
mult/mode/corr 1 2
variab/set Phase angcorr list
GouyPhase = -GouyPhase           # determine compensation for Gouy shift
                                  # equivalent to slight frequency tuning

```

Jump to: [Commands](#), [Theory](#)

```

# Test Gouy compensation

copy 2 1
reson/run 1
mult/mode/corr 1 2
#
C Note that round-trip phase is zero, indicating compensation
#
bell
pause 2
TEST = 0                                # exit test mode
Xdec = .03
rescale/shift 2 x=Xdec                  # start with beam off-set
clear 1 0                                # zero out cavity
Ntimes = 3/(1-Efficiency)
plot/w ex33e2.plt
reson/run Ntimes

```

Ex33f: Global definition of half-symmetric resonator

The global component description may be used to define a resonator. It is necessary to specifically reset zreff to zero for proper performance.

Input: ex33f.inp

```

c## ex33f
c
c Example 33f
c
c Complare reversed global resonator with simple resonator.
c
c The global resonator has the concave mirror on the left and the beam is started
c backward with parity=-1
c
c For comparison, the simple resonator has the concave mirror on the right.
c
variab/dec/int pass GLOBAL
c
c GLOBAL=0 selects non-global resonator. GLOBAL=1 selects global resonator.
c
GLOBAL = 1
macro/def reson_simple/o
    pass = pass + 1 list                # increment pass counter
    prop 45                             # propagate 45 cm.
    mirror/sph 0 -50                    # mirror of 50 cm. radius
    clap/c/n 0 .14                      # .14 cm. radius aperture
    prop 45                             # propagate 45 cm. along beam
    mirror/sph 0 1.e15                   # flat mirror
macro/end
macro/def reson_global/o
    pass = pass + 1 list                # increment pass counter
    vertex/locate/abs 0 0 -45           # mirror at -45 cm, (0,0,-45)
    vertex/rotate/set 0 180 0           # mirror face is toward +z
    mirror/global rad=-50               # mirror of 50 cm. radius, concave
    vertex/locate/abs 0 0 0             # back to (0,0,0)
    vertex/rotate/set 0 0 0             # not rotation

```

Jump to: [Commands](#), [Theory](#)

```

    mirror/global/flat 1                # flat mirror at (0,0,0)
    zreff/set 1 0                      # Reset zreff in each path
macro/end
wavelength/set 0 1.064                # set wavelengths
units 0 .005
if GLOBAL=0 then
    c
    c Simple, non-global definition
    c
    resonator/name reson_simple
    resonator/eigen/test 1
    resonator/eigen/list 1
    C C before reson/run
    units
    geodata
    global/list
    write/off
    reson/run 1
    write/on
    C C after reson/run
    units
    geodata
    global/list
else
    c
    c Global definition
    c
    global/def parity=-1                # start beam backward for backward-defined resonator
    reson/name reson_global
    reson/eigen/test 1
    reson/eigen/list 1
    C C before reson/run
    units
    geodata
    global/list
    write/off
    reson/run 1
    write/on
    C C after reson/run
    units
    geodata
    global/list
endif

```

Ex33g: Example of a wire obscuration to induce TEM₁₀

Illustrates a resonator driven to running in near-single mode TEM₁₀ in the x-direction and multi-mode in the y-direction.

Input: `ex33g.inp`

Ex33h: Example of half-symmetric resonator with rotating end mirror

Illustration of stable resonator that lases off a rotating end mirror.

Input: `ex33h.inp`

Ex33i: Two-wavelength flat-flat resonator

Example of two-beams of different wavelengths in flat-flat resonator, including absolute round-trip phase included by fractional wavefront.

Input: `ex33i.inp`

Ex33j: Multiple beams through the same resonator

An example to illustrate putting multiple beams through the same resonator. `Copy/geodata` is used to preserve the exact surrogate gaussian beam parameters. The example can be modified to set up the complex amplitude of the different beams differently.

Input: `ex33j.inp`

Ex33k: Extended cavity and spurious reflections

Objects external to the laser cavity can reflect light back into the cavity. Consequently the cavity can potentially lase on both the intended cavity mirror and on one or more external reflectors. The simplest case consists of only two paths. Consider the example illustrated in of Fig. 33.8. The desired resonator is formed by end mirrors M1 and M2. If mirror M2 is only partially reflecting, a second resonator path can be formed between M1 and M3. For the M1-M2 resonator, mirror M2 is modeled as a partial reflector. For the M1-M3 resonator, mirror M2 is modeled as a partially transmitting element. Both resonator paths compete for the same gain, although the mode size will generally be different.

Each resonator path will run on a frequency that forms an exact integer number of wavelengths in its round trip path. The laser paths will run on wavelengths $\lambda_{1-2} = L_{1-2}/N_{1-2}$ and $\lambda_{1-3} = L_{1-3}/N_{1-3}$. In principle, there are wavelengths for all choices of N_{1-2} and N_{1-3} to form an ensemble of longitudinal modes, but only the longitudinal modes that fall under the atomic line width will receive gain. The two families of longitudinal modes are separated by frequencies $c/2L_{1-2}$ and $c/2L_{1-3}$ respectively. Only modes that fall under the atomic line width receive amplification. While the modes of each family are separated by a fixed amount the separation between wavelengths λ_{1-2} and λ_{1-3} and associated frequencies f_{1-2} and f_{1-3} have no fixed separation.

Jump to: [Commands](#), [Theory](#)

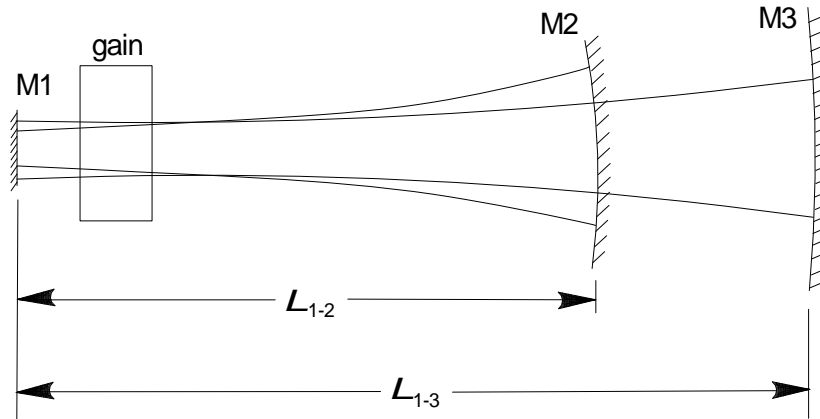


Fig. 33.8. Mirrors M1 and M2 form the desired resonator. Mirror M2 is assumed to be partially transmitting. An alternate path from M1 to M3 competes for gain.

$N_{1-2} = \lfloor (L_{1-2}/\lambda) \rfloor$ and $N_{1-3} = \lfloor (L_{1-3}/\lambda) \rfloor$, where “ \lfloor ” is the lowest integer function. Each path Unlike longitudinal modes that have frequencies separated by $c/2L$.

Input: `ex33k.inp`

Ex33I: Coupled resonator

This example illustrates the essential concepts needed to model coupled resonators. This model provides for 1) mode coupling optics between the two resonators to allow independent configurations for the two resonators including different radii of curvature and different lengths and 2) a method to keep the two resonators temporally synchronized to within the shortest round trip time.

Fig. 33.9 illustrates two coupled resonators. It is assumed that the gain is in Resonator 1 and that some unspecified elements or material may be in Resonator 2 such as a nonlinear optics element. The treatment may employ two polarization states if required simply by specifying that the optical beam arrays be polarized. For reasons explained by the resonant dipole model (and commonly observed experimentally) a free running laser will quickly adjust the exact wavelengths of the longitudinal modes to have zero round trip phase. Mirror M2 is partially transmitting so that the fields of Resonator 1 and 2 are cross coupled.

We have assumed that the design of the system may incorporate mode coupling optics which creates an image of M2' of mirror M2 such that the radius R2' and position of M2' may be set independently of M2. With no mode coupling optics, $R2 = R2'$ constituting a single ideal partially reflecting surface. For resonators having reflecting end mirrors the running modes will have their radii of curvature match the radii of the respective mirrors (or the image M2').

If we assume a transverse radius at M2 and M2' (unity magnification) and specify radi R2 and R2', then radi R1 and R3 may be calculated (see Sect 4.4.1 in GLAD Theory Manual) if L1 and L2 are specified.

On an instantaneous basis, Resonator 1 will see the returned field from M2 and the contribution of Resonator 2 as a complex reflectivity. Ideally we would make sure the modes of the two resonators are coupled (see Fig. 33.9) so that the complex reflectivity consists only of reflectivity and a phase advance or retardation without a mode mismatch. To include a mode mismatch, a mode coupling operation using `mult/mode/parallel` could be implemented. In any case, the exact running wavelength will quickly

Jump to: [Commands](#), [Theory](#)

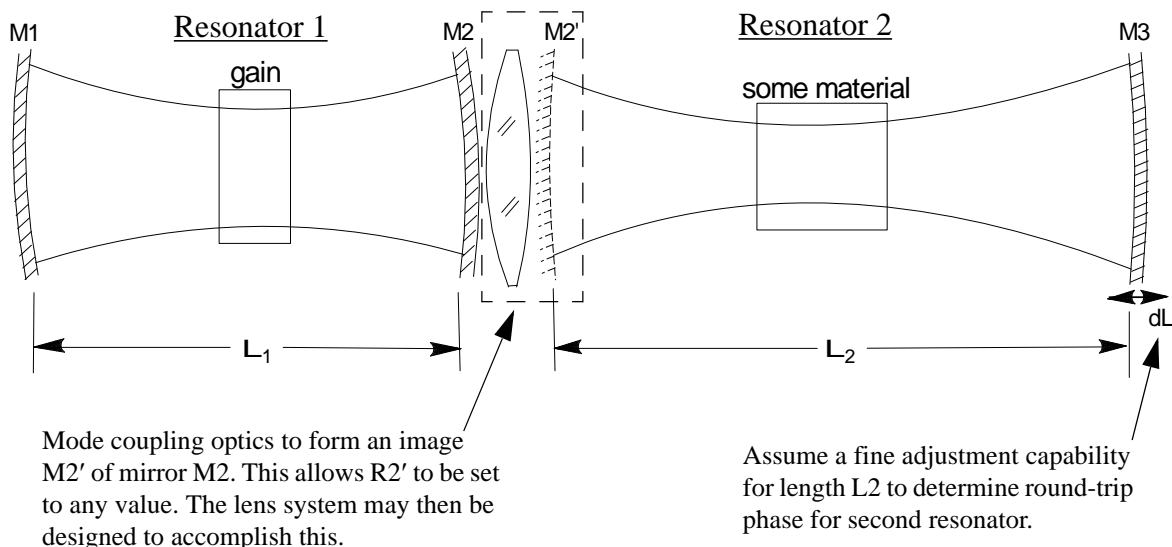


Fig. 33.9. Illustration of coupled resonators. Resonator 1 contains the gain medium. Resonator 2 may contains some other material such as a nonlinear optics element. Mirror M2 is partially reflecting and partially transmitting such the the field of Resonators 1 and 2 are cross coupled.

The combination of gain and a resonator cavity will cause the exact running wavelength to adjust to have zero round trip phase for Resonator 1. It is assumed that mode coupling optics will image mirror M2 into M2' such that R2' may take on any desired value. Thus Resonator 2 may be configured as desired. In Ex33.1, we assume that the imaging of M2 into M2' a unity magnification, but this is not required in the general case and may be modeled by using a magnification adjustment of the transferred fields.

A fine adjustment of length L2 is indicated as dL that may be used to adjust the round trip phase of Resonator 2. For most efficient coupling and stable phase performance in Resonator 2 the round trip phase should be zero, so any departure from zero round trip phase is considered a misalignment.

adjust because of the gain and spontaneous emission so that the phase of the complex reflectivity at M2 is canceled to achieve zero round trip phase.

On an instantaneous basis, Resonator 2 will see the effect of Resonator 1 to be a form of coherent injection with feedback in the Resonator 2 cavity. Ideally the exact length L2 is adjusted in the configuration to have zero round trip phase so that the feedback in Resonator 2 achieves maximum value. The model includes a round trip phase factor for Resonator 2 which may be set to zero for perfect adjustment. Note that even if there is a round trip phase error in Resonator 2 which is feeded back into Resonator 1, the wavelength adjustment that is due to gain in Resonator 1 will keep the round trip phase in Resonator 1 to be zero.

Assuming the optical path lengths L1 and L2 are different, the round trip times T1 and T2 are different.

$$T_1 = \frac{L_1}{c} \quad T_2 = \frac{L_2}{c} \quad (33.1)$$

In the typical (and convenient) method of modeling, a single complex amplitude array is used to represent the full 3D distribution of the field in a resonator over the full round trip pass. However if L2 is not equal to L1, the round trip times are different. We can maintain approximate synchronization in both resonators by alternately advancing the time in Resonators 1 and 2. For example having taken one round trip in Resonator 1 to advance t_1 to $t_1 + T_1$. If the new t_1 is such that $t_1 > t_2$, we then advance Resonator 2

one or more times until $t_2 > t_1$. Resonator 1 is then advanced again and the time difference is checked again. Provided the fields in the resonators change at a slower pace than the shorter of T1 or T2, this alternating procedure will result in a reasonably good numeric approximation of the continuous result.

In Ex33l, a simple partially reflecting mirror is used for M2, although coupling optics could easily be included. We build both resonators into a single macro but with the provision that part of the macro applies to Array 1 and part to Array 2. Consequently a single macro name may be used and both arrays may be controlled.

Input: ex33l.inp

```
c## Ex33l
c
c An example of a coupled resonator.
c
c Resonator 1 is developed between mirror M1 and M2.
c Resonator 2 is developed between M2p (the image of M2) and M3.
c
c In this example, for simplicity we assume 1:1 imaging of M2 into M2p.
c A generalization to include non-unity magnification is straightforward.
c
c Resonator 2 is set up first to stabilize the appropriate surrogate
c gaussian beam and to determine its round trip time.
c
c Resonator 1 is then set up and will call Resonator 2 as required to
c keep the total times of the two resonators synchronized to within the
c the round trip times.
c
c Beams
c -----
c 1                # propagation beam for Resonator 1
c 2                # propagation beam for Resonator 2
c 3                # output after M3
c 4                # temporary beam
c 5                # temporary beam
c
variable/dec/int Ntimes TEST Pass1 Pass2 Pass3
TEST = 1
TotalTime1 = 0.
TotalTime2 = 0.
Lambda = 1.064E-4
W2 = .02                # transverse radius at M2
Mag = 1.0                # magnification between M2 and M2p
R2 = 0.9                # radius of curvature of M2
L1 = 1.4                # length of Resonator 1
L2 = 0.8                # length of Resonator 1
R2p = 0.8                # Radius of image M2p
PhaseError2 = 0          # round trip phase for Resonator2
Apt1 = .08
Apt2 = .08
Apt3 = .10
Reflectance_M2 = .7
Transmission_M2 = 1 - Reflectance_M2
```

Jump to: [Commands](#), [Theory](#)

```

Reflectance_M3 = .995
Transmission_M3 = 1 - Reflectance_M3
c
c  Beers's Law gain
c
g0 = .18
Es = 1000
Lgain = .8
beer/set g0 Es
c
c  Set up other resonator parameters
c
Z2 = R2/(1 + (Lambda*R2/pi/W2^2)^2)      # distance to waist from M2.
Z2=
Z1 = L1 - Z2                             # distance from waist to M1 for length L1
Z1=
W10 = W2/sqrt(1 + (pi*W2^2/Lambda/R2)^2) # waist radius of Resonator 1
W10=
Z1_ray = pi*W10^2/Lambda                 # Raylieght length for Resonator 1
Z1_ray=
R1 = Z1 + Z1_ray^2/Z1                    # Radius of M1 to yield R2 at L1
R1=
W2p = Mag*W2                             # transverse radius at M2p for
magnification Mag
W2p=
Z2p = R2p/(1 + (Lambda*R2p/pi/W2p^2)^2)  # distance to waist from M2p.
Z2p=
Z3 = L2 - Z2p                             # distance from waist to M3 for length L2
Z3=
W20 = W2p/sqrt(1 + (pi*W2p^2/Lambda/R2p)^2) # waist radius of Resonator 2
W20=
Z2_ray = pi*W20^2/Lambda                 # Raylieght length for Resonator 2
Z2_ray=
R3 = Z3 + Z2_ray^2/Z3                    # Radius of M1 to yield R1 at L1
Z2=
Z1=
Z2p=
Z3=
R3=

array/s 1 128
nbeam 2
nbeam 5 data
wavelength/set 0 Lambda*1e4
units/set 0 .003

macro/def Resonator1
  c
  c Resonator 1, start at M2
  c
  if [TEST==0] then
    Pass1 = Pass1 + 1
  endif
  mirror 1 -R2

```

Jump to: [Commands](#), [Theory](#)

```

clap/c/no 1 Apt1
prop L1 1
beer/noprop 1 Lgain
mirror 1 R1
clap/c/no 1 Apt2
beer/noprop 1 Lgain
prop L1 1
c
c Delete piston term from Beam 1. Assumes wavelength has shifted.
c
if [TEST==0] then
  variable/set Sr 1 point/sr
  if [Sr<0] then
    phase/piston 1 180
  endif
  copy 1 5
  conjugate 5
  mult/mode/corr 1 5
  variable/set Angcorr angcorr list
  phase/piston 1 -Angcorr/2
  c
  c Exchange fields
  c
  Pass1=
  variab/set Energy1 1 energy list
  variab/set Energy2 2 energy list
  EnergyTotal = Energy1 + Energy2 list
  c clear 1 .2
  c clear 2 .8
  copy/con 1 4 # copy for use in Resonator 2
  copy/con 2 5
  A1 = -sqrt(Reflectance_M2)
  A2 = sqrt(Transmission_M2)
  A3 = sqrt(Reflectance_M2)
  A4 = sqrt(Transmission_M2)
  mult/complex 1 A1
  mult/complex 4 A2
  mult/complex 2 A3
  mult/complex 5 A4
  add/coh/con 2 4
  add/coh/con 1 5
  if [mod(Pass1,10)==0] then
    Pass3 = Pass3 + 1
    variab/set Energy1 1 energy
    variab/set Energy2 2 energy
    EnergyTotal = Energy1 + Energy2
    udata/set Pass3 Pass3 Energy1 Energy2 EnergyTotal
  endif
  TotalTime1 = TotalTime1 + T1
  if [TotalTime1>TotalTime2] then
    Ntimes = ceil((TotalTime1-TotalTime2)/T2)
    macro/run Resonator2/Ntimes
  endif
  if [mod(Pass1,10)==0] then

```

Jump to: [Commands](#), [Theory](#)

```

        plot/w Plot5.plt
        title Pass1 = @Pass1
        plot/l 2
    endif
endif
macro/end
macro/def Resonator2
    c
    c Resonator 2, start at M2p
    c
    Pass2 = Pass2 + 1
    if [TEST==0] then
c        units 2
c        geodata 2
        units/set 2 Units2
        geodata/set 2 waistx=W2p waisty=W2p
        geodata/set 2 zwastx=0 zwasty=0
        zreff/set 2 0.
c        geodata 2
c        read/scr
    endif
    clap/c/n 2 Apt1
    prop L2 2
    copy 2 3
    mult 3 Transmission_M3
    mirror 2 -R3
    clap/c/no 2 Apt3
    mult 2 Reflectance_M3
    prop L2 2
    c
    c Delete piston term from Beam 2. Assumes length of Resonator 2 has been
    adjusted.
    c
    if [TEST==0] then
        variable/set Sr 2 point/sr list
        variable/set Si 2 point/si list
        if [Sr<0] then
            phase/piston 2 180
        endif
        variable/set Energy2 2 energy
        if [Energy2>0] then
            copy 2 5
            conjugate 5
            mult/mode/corr 2 5
            variable/set Angcorr angcorr
            phase/piston 2 -Angcorr/2
        endif
    endif
    mirror 2 R2p
    if [TEST==0] then
        TotalTime2 = TotalTime2 + T2
    endif
    phase/piston 2 PhaseError2
macro/end

```

Output after M3 is Beam 3

Jump to: [Commands](#), [Theory](#)

```

gaussian 1 1 W2 radx=R2
gaussian 2 1 W2p radx=-R2p
clear 4 0
if 0 then
  c
  c Check Resonator 1 with setup gaussian
  c
  geodata 1
  macro/run Resonator1
  geodata 1
  read/scr
endif
if 0 then
  c
  c Check Resonator 2 with setup gaussian
  c
  geodata 2
  macro/run Resonator2
  geodata 2
  read/scr
endif
reson/name Resonator2
reson/eigen/test 2
reson/eigen/list
variab/set T2 2 reson/roundtriptime
variab/set Units2 2 units
reson/name Resonator1
reson/eigen/test 1
reson/eigen/list
variab/set T1 1 reson/roundtriptime
T1=
T2=
Pass1 = 1
Pass2 = 1
clear 1 0
clear 2 0
clear 4 0
gaussian/c/n 1 1 W2
energy/norm 1 4
c gaussian/c/n 2 1 W2p
c energy/norm 2 1
variab/set Energy1 1 energy
variab/set Energy2 2 energy
EnergyTotal = Energy1 + Energy2
udata/set Pass1 Pass1 Energy1 Energy2 EnergyTotal
udata/list
point/list 1
TEST = 0
macro/def system
  resonator/run 1
  c Ntimes=
  c Pass1=
  c Pass2=
  c TotalTime1=

```

Jump to: [Commands](#), [Theory](#)

```

c TotalTime2=
c udata/list
mac/end
c macro/run system/400
reson/run 400
plot/w plot6.plt
plot/udata 1 3
Pass1=
Pass2=

```

Ex33m: Finding multiple modes by orthogonality

This example illustrates a method of finding multiple modes using the Fox-Li to find the first, lowest loss mode and then performing Fox-Li again with orthogonalization with the first mode. This process is repeated as many times as needed with orthogonalization against all the previously solved modes. In this example TEM(0,0), TEM(1,0), TEM(0,1), and TEM(1,1) are found through sequential orthogonalization.

Input: [ex33m.inp](#)

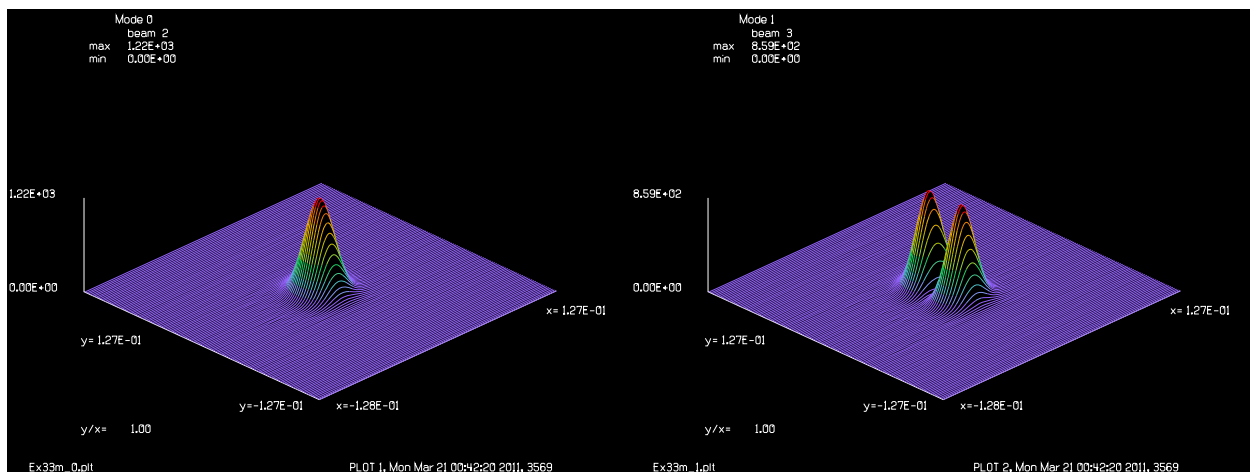


Fig. 33.10. First mode, TEM(0,0), found by Fox-Li method.

Fig. 33.11. Second mode, TEM(0,1), found by Fox-Li and orthogonalizing with first mode. See Fig. 33.10.

Ex33n: Bowtie resonator

This example illustrates a method of finding multiple modes using the Fox-Li to find the first, lowest loss mode and then performing Fox-Li again with orthogonalization with the first mode. This process is

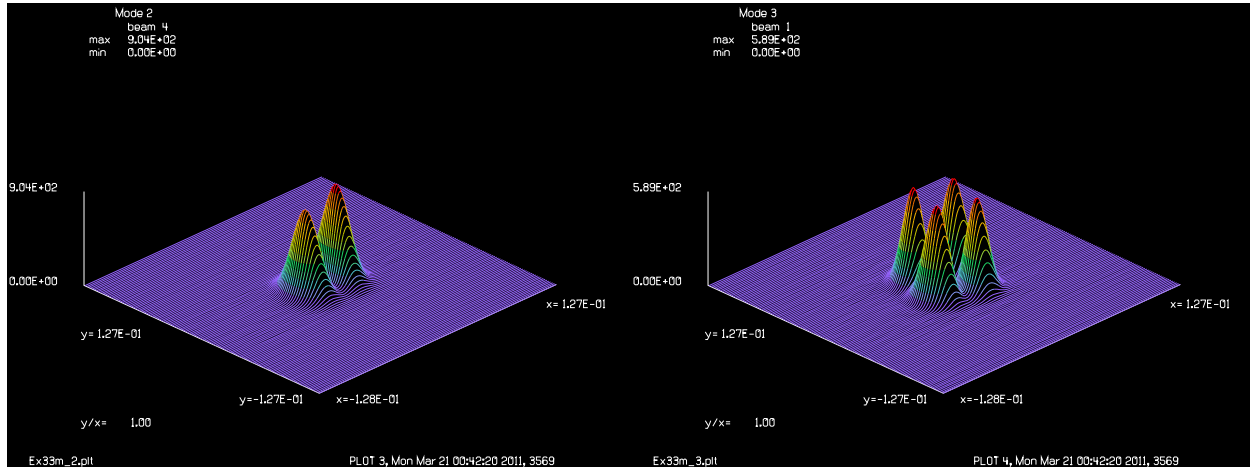


Fig. 33.12. Third mode, TEM(1,0), by Fox-Li and orthogonalizing with first and second modes. See Figs. 33.10 and 33.11.

Fig. 33.13. Fourth mode, TEM(1,1), found Fox-Li and orthogonalizing with first through third modes. See Figs. 33.10 through 33.12.

repeated as many times as needed with orthogonalization against all the previously solved modes. In this example TEM(0,0), TEM(1,0), TEM(0,1), and TEM(1,1) are found through sequential orthogonalization.

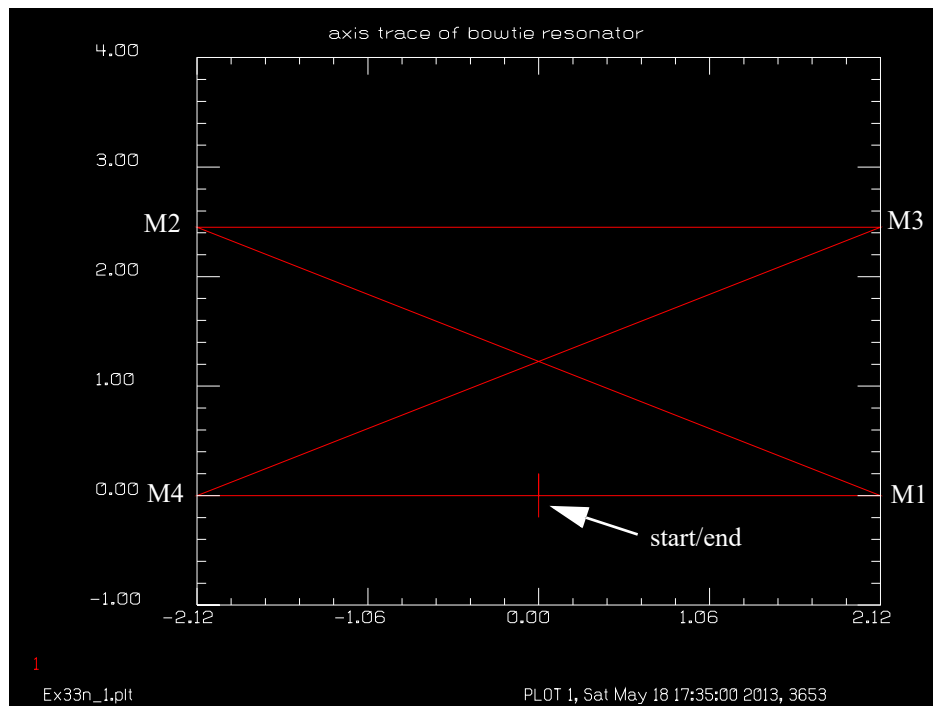


Fig. 33.14. Path of center ray for bowtie resonator. The calculation emanates from “start” and proceeds to M1, M2, M3, and M4 and finally returns to “end”.

Input: `ex33n.inp`

```
c## Ex33n
c
```

Jump to: [Commands](#), [Theory](#)

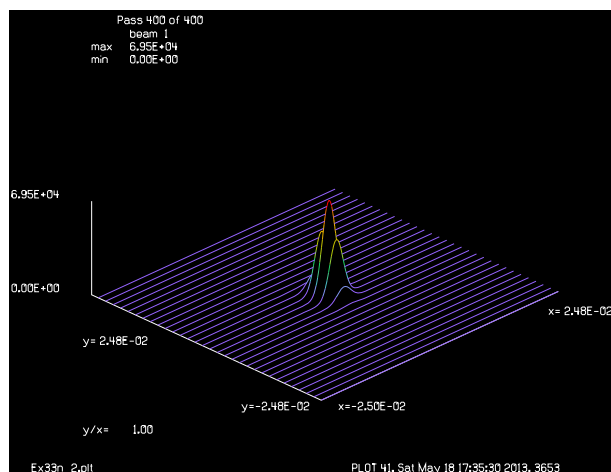


Fig. 33.15. Converged mode after 400 passes.

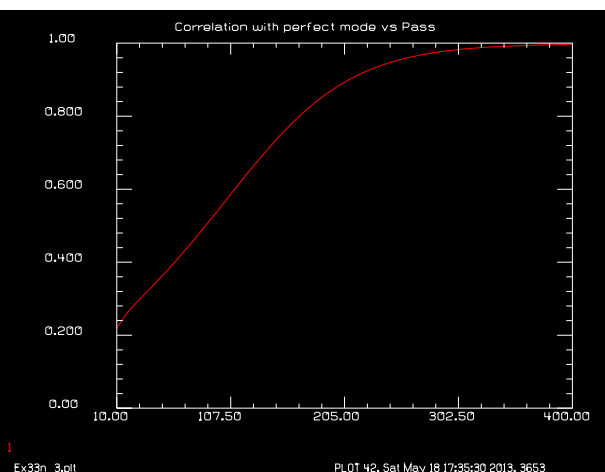


Fig. 33.16. Correlation of cavity mode with ideal mode versus pass number, showing a high degree of convergence to the ideal mode.

```

c Example of Bowtie resonator
c
c Four mirrors are arranged in an "X" configuration.
c
alias Name Ex33n
variable/dec/int TEST Pass Count Ntimes
Ntimes = 400          # number of resonator iterations
Lambda = 0.400e-4    # wavelength
diag = 4.9           # diagonal lengthcm
MirDeg = 15.         # beam incident angle on mirrors
height = diag*sind(MirDeg*2) list
width = sqrt(diag^2 - height^2) list

x1 = 0
z1 = 0
x2 = 0
z2 = width/2
x3 = height
z3 = -width/2
x4 = height
z4 = width/2
x5 = 0.
z5 = -width/2
x6 = 0.
z6 = 0.
c
c Draw bowtie ray path with "udata"
c
      X      Y
udata/set 1  z1  x1
udata/set 2  z2  x2
udata/set 3  z3  x3
udata/set 4  z4  x4
udata/set 5  z5  x5
udata/set 6  z6  x6

```

```

udata/set 7 0. 0.2
udata/set 8 0. -0.2

plot/w @Name_1.plt
title axis trace of bowtie resonator
plot/udata min=-1 max=4
c
c Reset "udata"
c
udata/clear

Radius = -2.6 # concave mirrors

Apt = .02 # Need to make aperture small enough to clip the beam

array/set 0 256
nbeam 2 data
units/field 0 .025
wavelength/set 0 Lambda*1e4
gaussian/cir/con 1 1. .2

global/define 0 0. 0. 0.

# now wrap those mirrors in a macro to form resonator

macro/def reson
  Pass = Pass + 1
  vertex/locate/absolute x=0. y=0. z=width/2
  vertex/rotate/set ry=-MirDeg
  mirror/global/conic radius=Radius
  if [TEST==1&&Pass==1] then
    CC C Mirror: M1
    variable/set Grx 1 global/grx
    variable/set Gry 1 global/gry
    variable/set Grz 1 global/grz
    CC C Expected (@x2,0,@z2), calculated (@Grx,@Gry,@Grz)
  endif
  clap/cir/noadjust 0 rad=Apt

  vertex/locate/absolute x=height y=0. z=-width/2
  vertex/rotate/set ry=180-MirDeg
  mirror/global/conic radius=Radius
  if [TEST==1&&Pass==1] then
    CC C Mirror: M2
    variable/set Grx 1 global/grx
    variable/set Gry 1 global/gry
    variable/set Grz 1 global/grz
    CC C Expected (@x3,0,@z3), calculated (@Grx,@Gry,@Grz)
  endif
  clap/cir/noadjust 0 rad=Apt

  vertex/locate/absolute x=height y=0. z=width/2
  vertex/rotate/set ry=MirDeg
  mirror/global/conic radius=Radius

```

Jump to: [Commands](#), [Theory](#)

```

if [TEST==1&&Pass==1] then
  CC C Mirror: M3
  variable/set Grx 1 global/grx
  variable/set Gry 1 global/gry
  variable/set Grz 1 global/grz
  CC C Expected (@x4,0,@z4), calculated (@Grx,@Gry,@Grz)
endif
clap/cir/noadjust 0 rad=Apt

vertex/locate/absolute x=0. y=0. z=-width/2
vertex/rotate/set ry=180.+MirDeg
mirror/global/conic radius=Radius
if [TEST==1&&Pass==1] then
  CC C Mirror: M4
  variable/set Grx 1 global/grx
  variable/set Gry 1 global/gry
  variable/set Grz 1 global/grz
  CC C Expected (@x5,0,@z5), calculated (@Grx,@Gry,@Grz)
endif
clap/cir/noadjust 0 rad=Apt

vertex/locate/absolute x=0. y=0. z=0.
prop/vertex
if [TEST==1&&Pass==1] then
  CC C Return to start
  variable/set Grx 1 global/grx
  variable/set Gry 1 global/gry
  variable/set Grz 1 global/grz
  CC C Expected (@x6,0,@z6), calculated (@Grx,@Gry,@Grz)
endif
energy/norm 1 1
zreff/set 0 0.
if [mod(Pass,10)==0] then
  c Run diagnostics
  Count = Count + 1
  plot/w @Name_2.plt
  title Pass @Pass of @Ntimes
  plot/l 1
  c Check correlation with ideal modes in Array 2
  mult/mode/corr 1 2
  variab/set Abscorr 1 abscorr
  udata/set Count Pass Abscorr
endif
macro/end

resonator/name reson

write/off
TEST = 1
resonator/eigen/test 1      # calculate eigen mode put into beam 2
TEST = 0
write/on
resonator/eigen/list
c

```

Jump to: [Commands](#), [Theory](#)

```
c  Save ideal mode in Array 2
c
Pass = 0
copy 1 2

c inject noise beam

clear 1 0
noise 1 1
energy/norm 1 1
write/off
reson/run Ntimes    # run until almost converged
write/on
plot/w @Name_3.plt
title Correlation with perfect mode vs Pass
plot/udata min=0 max=1
```

Ex34: Stable-unstable resonator

This is an example of a resonator which is stable in one direction and unstable in the other. The resonator consists of a spherical mirror at one end and a cylindrical mirror at the other. The cylindrical mirror is curved about an axis in the x-direction. A circular scraper mirror is used to extract energy from the cavity. .

Table. 34.1. Parameters

length	117.5 cm
spherical mirror radius	400 cm
spherical mirror aperture	elliptical: 3 cm \times 7 cm
cylindrical mirror radius	165 cm
scraper mirror projected size	2.5 cm dia.
wavelength	1.73 microns

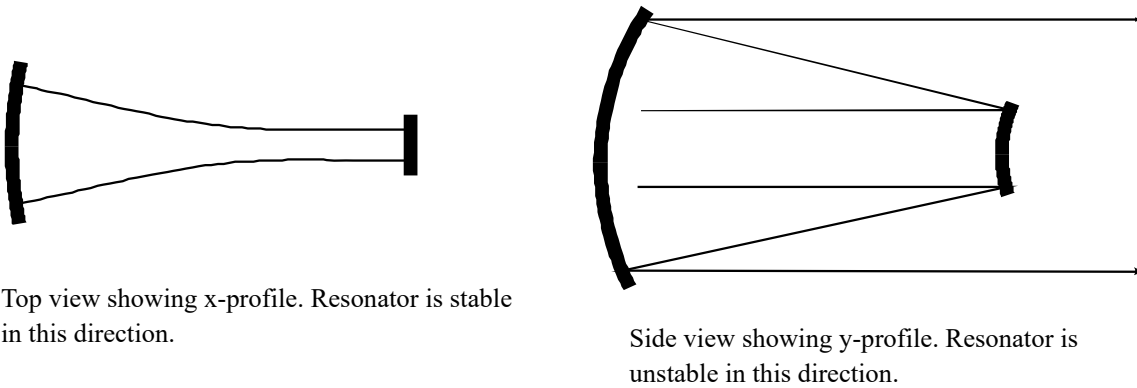


Fig. 34.1. The resonator consists of a spherical mirror at one end of radius 400 cm and a convex cylindrical mirror of radius 165 cm. The mirrors are separated by 117.5 cm. A scraper mirror of diameter of 2.5 cm is located 3 cm in front of the cylindrical mirror. An elliptical aperture 2.6 cm by 7.0 cm is located in front of the spherical mirror to give mode selection.

The resonator will operate in a separable manner, unless the two directions are coupled by nonlinear gain. The cylindrical mirror acts as a plane mirror in the x-direction and a waist of radius 0.1 cm is formed because of interaction with the spherical mirror. In the y-direction, the convex cylindrical mirror works with the spherical mirror to form an unstable resonator of magnification 2.42.

The geometric parameters of the surrogate gaussian beam control the selection of diffraction algorithm. Any starting distribution may be used but it is important to set the surrogate gaussian parameters up correctly so the resonator is geometrically stable in the x-direction. This is done by calculating the waist size of the gaussian mode for the spherical mirror and the wavelength of 1.73 microns. The y-direction is unstable and we rescale the size of the array and reset the geometric data to be sure the resonator parameters are reset to the same values for each cycle.

Any starting distribution may be used. In this example, a constant amplitude array was formed in Beam 2 and copied into Beam 1, using the `copy/con` command which does not alter the geometric data in Beam 1. This data could also have been directly set with the `geodata` command.

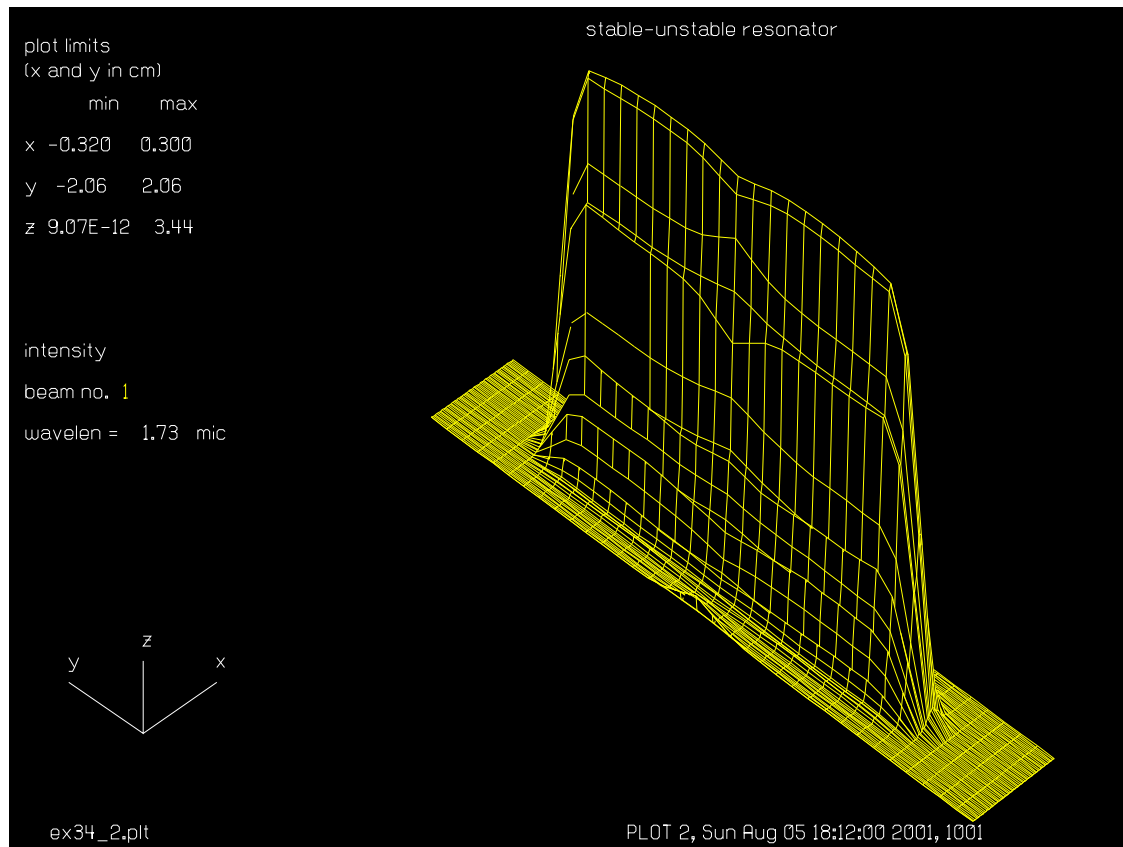


Fig. 34.2. Plot of intensity profile of converged mode showing significantly greater width in the y-direction, which is limited by the width of the scraper mirror.

Input: ex34.inp

```

c## ex34
c
c Example 34: Stable-unstable Anamorphic Resonator
c
c This resonator illustrates the use of a stable geometry in the x-direction
c and an unstable resonator in the y-direction. The resonator was designed
c by W. Sweatt and D. Neal of Sandia National Laboratory. A combination of a
c concave spherical and convex cylindrical mirrors is used. In the x-
c direction, the cylindrical mirror acts as a plane mirror and a small
c waist would be formed at this mirror if the convex mirror were flat.
c In the y-direction, the convex mirror acts as a diverging element to
c form an unstable confocal resonator in that direction.
c
c As with the stable resonator of Ex. 33, the geometry in the x-direction
c should be set up to have the surrogate gaussian beam waist of the proper
c size initially. This will ensure that the propagation algorithms are
c constant for each path and are in accord with the geometrical stable
c condition.
c
c Beam 2 is used to build any desired starting condtion. It is copied

```

Jump to: [Commands](#), [Theory](#)

```

c using COPY/CONSTANT into Beam 1 so as to not change the surrogate gaussian
c beam parameters of Beam 1. Beam 2 is then deactivated.
c
c The surrogate gaussian beam parameters which set the choice of propagation
c algorithm are constant for the x-direction which is stable. This is
c because the initial beam size is chosen to match the stable geometric
c gaussian mode. The y-direction is unstable and the surrogate gaussian
c waist size and location must be reset each path. Note that the beam
c has a plane reference surface (iplany=1) at the beginning and end of
c each path so no correction of the reference surface is required.
c
variab/dec/int pass
macro/def ex34/o
    pass = pass + 1          # increment pass counter
    step = step + 1          # increment step number
    mirror/ycyl 1 165
    dist -117.5
    mirror/sph 1 400.
    clap/ell/c 1 .15 3.5
    dist 114.5
    clap/cir/con 1 1.25
    dist 3.
    variab/set energy 1 energy
    energy = energy - 1      # calculate energy difference
    udata/set pass step energy # store energy differences
    rescale 1 .32 3.875
    energy/norm 1 1
    geodata/set 1 zwaisty=0 waisty=1.25 # reset geometric data
macro/end
wavelength/set 1 1.73
units/field 1 .32 3.875    # initialize units
gaus/ell/con 1 1 .100164 1.25 # initialize Beam 1 to set up surrogate
                                # gaussian beam parameters
nbeam 2                    # add second beam which is
                                # a constant field of 1's.
copy/con 2 1              # copy Beam 2 to Beam 1
nbeam 1                    # revert to Beam 1
energy/norm 1 1
pass = 0                  # initialize pass counter
step = 0                  # for pass counters
geodata/set 1 zwaisty=0 waisty=1.25 # set geometric data
write/d ex34.out/o
mac ex34/20
udata/list
title stable-unstable resonator
plot/watch ex34_1.plt
plot/udata
plot/watch ex34_2.plt
set/window/abs -.4 .4 -2. 2.
plot/iso first=1 last=1
end

```


Ex35: Distributed propagation through a refractive surface

Table. 35.1. Table of Ex35 examples

Ex35a: Distributed propagation, method of aperture division.	3
Ex35b: Distributed propagation, phase grating in incident beam	8
Ex35c: Distributed propagation, phase grating on refracting surface	10
Ex35d: Propagation to tilted surface for a phase grating.	13
Ex35e: Propagation to tilted surface for a circular aperture.	17

Light propagates differently in vacuum or air than in glass or other material with index of refraction not equal to 1. The effective wavelength is $1/n$ times the wavelength in vacuum. This means that the effective propagation distance is different inside glass than in vacuum. Consider Fig. 35.1 which shows an incident collimated beam and a refractive surface. When the beam is partly inside the medium, the outer part of the beam has the original wavefront curvature and the inner part has the curvature of the converging wavefront. Figure 35.2 shows the more general case. Let P_1 be the center of the object wavefront and P_2 be the center of the image wavefront. The effective propagation distance for a movement along the axis of $n_i = 2$ is different in the two axes.

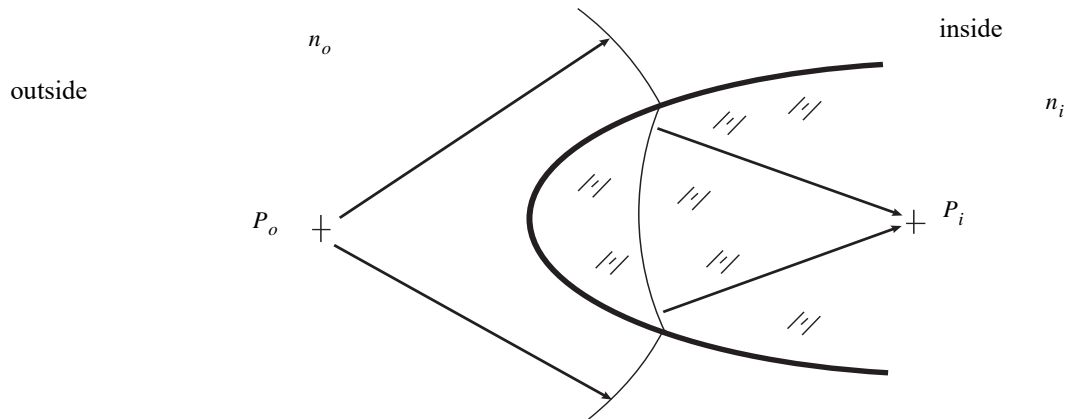


Fig. 35.1. Schematic of a beam partially into a medium with a spherically refracting lens surface. The wavefront in the outer portion is the incident collimated beam. The wavefront in the inner portion is converging to the focus in the medium. In representing the wavefront in an intermediate stage, we must consider the fact that the complex amplitude seems to have to distinct references surfaces and that the effective propagation distances are different in different portions of the beam.

$$z_{eff_o} = \frac{R_o}{R_o + z} \frac{\Delta z}{n_o} \quad (35.1)$$

$$z_{eff_i} = \frac{R_i}{R_i + z} \frac{\Delta z}{n_i} \quad (35.2)$$

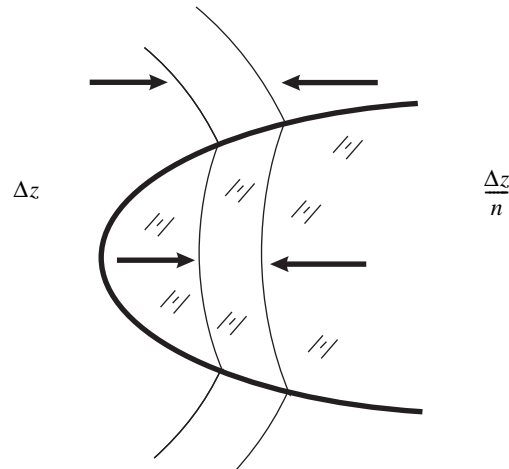


Fig. 35.2. Effective propagation distances outside and inside the medium.

For an incident collimated beam in air being refracted into an index of $n_i = 2$, and a radius of curvature of the surface of $R_s = 8$. The effective distances are

$$z_{eff_o} = \Delta z \quad (35.3)$$

$$z_{eff_i} = \frac{8}{16 - \Delta z} \Delta z \quad (35.4)$$

where $R_o = \infty$ and $R_i = -16$.

$$\frac{n_o}{R_o} = \frac{n_i}{R_i} - \frac{n_o - n_i}{R_s} \quad (35.5)$$

where the sign convention for wavefronts wavefront radii is positive if the wavefront is diverging and negative if converging. This differs from the conventions of the usual Lens Maker's equation.

Applying Eq. (35.4), three steps of 0.5 result in effective propagations in the medium as shown in Table 35.2.

Table. 35.2. Effective propagation distances outside and inside the medium.

Step	Zeff outside	Zeff inside	Incremental
1	0.5	0.2581	0.2581
2	1.0	0.5333	0.2752
3	1.5	0.8276	0.2943

To illustrate the application of GLAD, we consider the case of a collimated beam incident on the end of a rod of index $n_i = 2$, with a curved surface on the end of 8 micron radius. If the incident beam is of 12 microns diameter, then the maximum surface sag is 2.25 micron. The inner part of the beam propagates for a longer time in the material and, if aberrations are introduced at the surface, the inner part of the beam has a longer path for the aberrations to be affected by diffraction. In order to correctly treat the problem, we need to have different parts of the beam have different effective propagation distances. We can do this in steps

Jump to: [Commands](#), [Theory](#)

by splitting the amplitude distribution into inside and outside zones. After the split the inside portion is propagated by the effective distance based on the medium index and convergent beam properties. Through the calculation we can carry a complex distribution for the outside, ψ_o , and for the inside, ψ_i , the process consists of

$$\psi_o(z + \Delta z) = \psi_o(z) - \psi(\Delta x) \quad (35.6)$$

$$\psi_i(z + \Delta z) = \psi_i(z) + \psi(\Delta x) \quad (35.7)$$

$$\psi_o(z + \Delta z) = \mathbf{F}\mathbf{F}^{-1}[H(\Delta z_{eff_o})\mathbf{F}\mathbf{F}[\psi_o(z + \Delta z)]] \quad (35.8)$$

$$\psi_i(z + \Delta z) = \mathbf{F}\mathbf{F}^{-1}[H(\Delta z_{eff_i})\mathbf{F}\mathbf{F}[\psi_i(z + \Delta z)]] \quad (35.9)$$

where $\psi(\Delta x)$ is the part of the aperture to be converted from outside space to inside space. After splitting the aperture, the outside and inside beams are propagated by different amounts. $H(\Delta z)$ is the transfer function for free space and $\mathbf{F}\mathbf{F}$ and $\mathbf{F}\mathbf{F}^{-1}$ are the forward and inverse Fourier transform.

To reduce diffraction effects at the interface between the aperture zones, the split should be done with a soft aperture. The supergaussian function is a suitable choice. The supergaussian splitting function is

$$\psi_i = \psi e^{-\left(\frac{r}{r_o}\right)^{2N}} \quad (35.10)$$

$$\psi_i = \psi \left[1 - e^{-\left(\frac{r}{r_o}\right)^{2N}} \right] \quad (35.11)$$

In Eqs. (35.10) and (35.11), r_o is the radius of the split, N is the supergaussian factor and ψ_i and ψ_o are the inside and outside complementary parts of the aperture. Figure 35.3 shows schematically how the complex amplitude is split into outside and inside portions in going through the refractive surface.

Ex35a: Distributed propagation, method of aperture division

To evaluate the procedure we first consider a simple input beam and compare single step propagation against propagation in steps of different effective length. Figure 35.4a shows the beam after propagation of 1.5 microns. Figure 35.4b shows the beam after propagation, in steps, into the medium. Note the slight perturbations in the amplitude due to the finite steps. This may be reduced by using more steps and/or wider supergaussian splits.

The actual beams are shown after the two splits at $z = 0.5$ and $z = 1.0$ in Fig. 35.5.

The effects of different effective propagation distances can be readily seen by adding a relatively high spatial frequency phase ripple to the beam. Consider a linear phase aberration of sinusoidal profile of frequency = 1.414 cycles/micron. The complex amplitude is

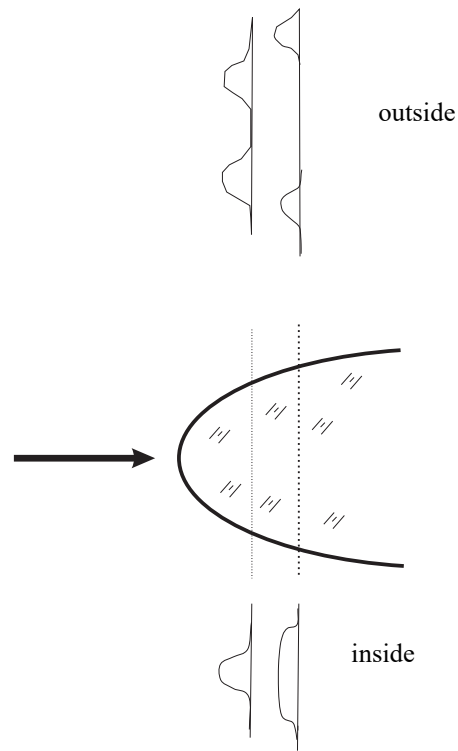


Fig. 35.3. The aperture is split into inside and outside regions with a soft aperture such as a supergaussian function and its complement. The inner portion is extended until all of the incident beam is inside the medium.

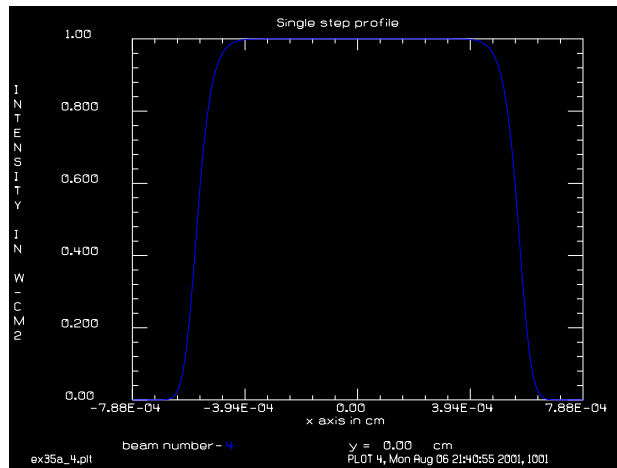


Fig. 35.4a. This is a profile of the intensity after propagation of 1.5 microns in a single step.

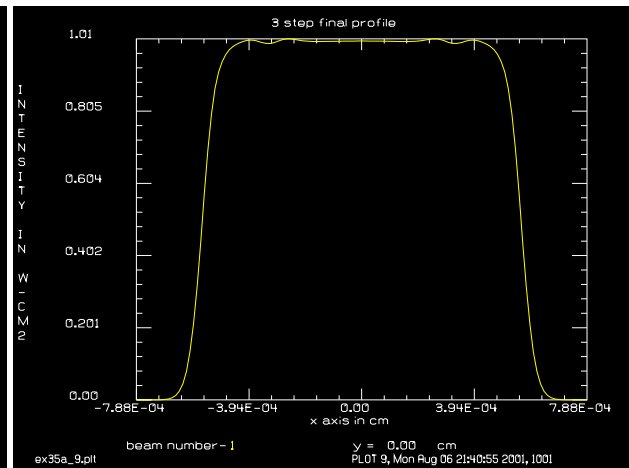


Fig. 35.4b. This figure shows the beam after propagation into the medium, in 3 steps. There are slight perturbations in the amplitude due to the finite steps. This may be reduced by using more steps and/or wider supergaussian splits.

$$a(x, y) = e^{jkC \cos(2\pi fx)} e^{-j\pi\lambda\Delta z f^2} \quad (35.12)$$

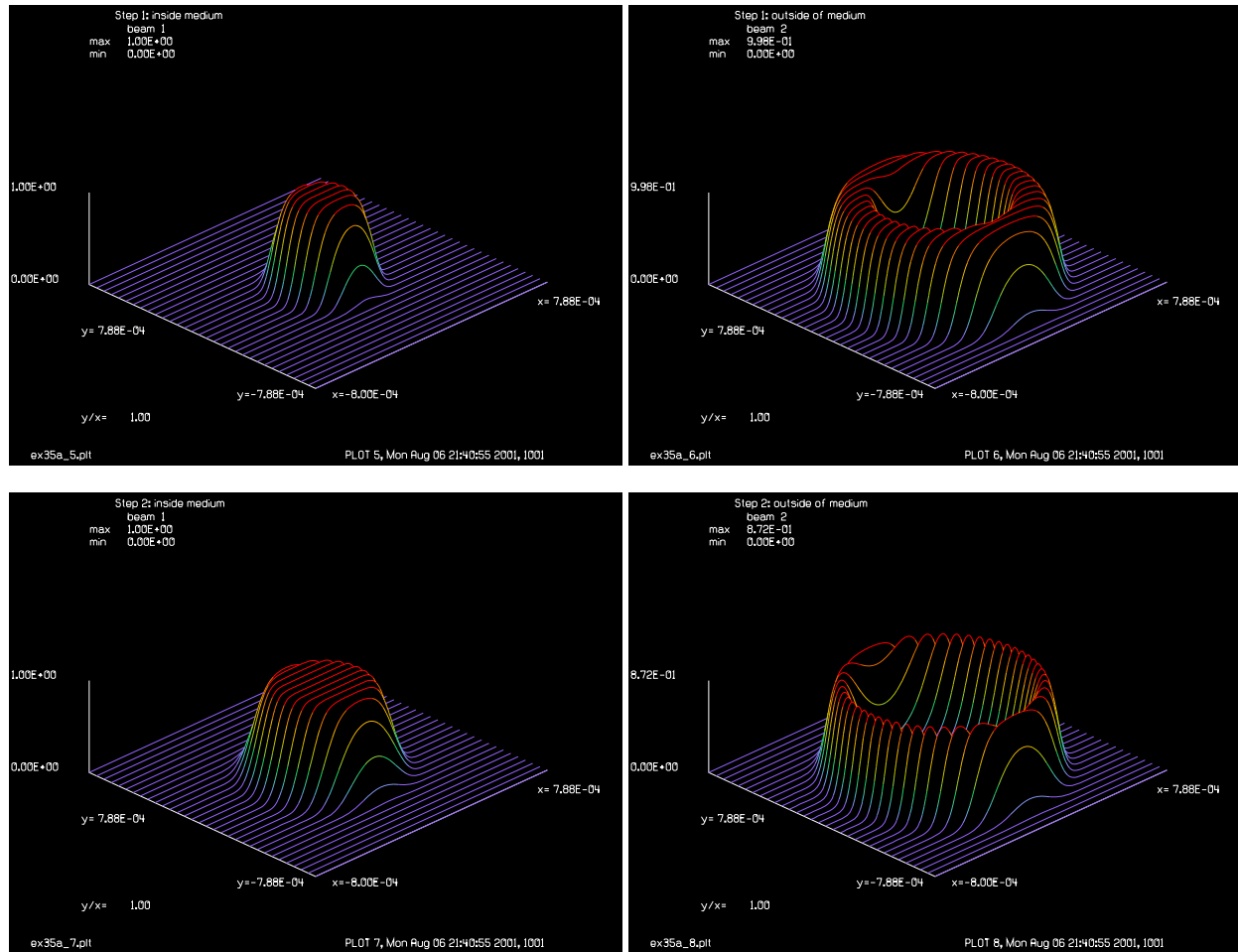


Fig. 35.5. The top figures show the inside (left) and outside (right) distribution at the first step and the lower figures show the inside and outside figures after the second step. The distribution inside the medium grows as the distribution outside the medium shrinks until all of the beam is inside the medium.

where C is the modulation of the phase ripple, f is the spatial frequency and the second term represents the diffraction factor. The characteristic diffraction length for a specific spatial frequency in vacuum is

$$\text{characteristic length} = \frac{2}{\lambda f^2} \quad (35.13)$$

based on one full period of the diffraction factor. For $f = 1.414$ cycles/micron, the characteristic length is 2 microns. We shall show the appearance of the beam as it propagates 1.5 microns into the medium which is 75% of the characteristic length.

Sinusoidal phase aberration of $f = 1.414$ will be completely converted to amplitude ripples at this point. The propagation in the medium with $n=2$, is 0.83 microns, equivalent to 41% of the characteristic length where the amplitude errors are small.

Input: `ex35a.inp`

c## `ex35a`

Jump to: [Commands](#), [Theory](#)

```

c
c Example 35a: Distributed Propagation through a Refractive Surface
c
c Light propagates differently in glass or other material with index of
c refraction not equal to 1. The effective wavelength is 1/n times the
c wavelength in vacuum. This means that the effective propagation distance
c is different inside glass than in vacuum.
c
c Step      Z eff. Beam 1      Z eff. Beam 2      Zone radius
c ----      -
c 1          .5                .5                2.8284
c 2          .5                .2753             4.0000
c 3          .5                .2943             4.8990
c
c All Steps  1.5                1.070
c
c BEAMS
c ----
c Beam 1    - net beam amplitude for distributed propagation
c Beam 2    - beam amplitude in portion of array outside refractive surface
c Beam 3    - temporary storage
c Beam 4    - temporary storage for adding aberration at surface refraction
c
echo/on
array/s 1 128          # Set Beam 1 to 128 x 128
nbeam 4                # Set number of beams to 4
                        # accepting default size of Beam 1
units/s 0 .125e-4      # Set units to .2 microns
wavelength/set 0 .5    # Set wavelengths to .5 microns
clear 1 0              # Initialize Beam 1 to 0
c
c Process Beam 4 as a non-distributed refraction
c
gauss/cir/con 4 1. 6e-4 8      # Gaussian starting distribution
copy/con 4 2                  # Make copy in Beam 2 for later use
title starting intensity
plot/watch ex35a_1.plt
plot/l 4
title starting phase
plot/watch ex35a_2.plt
plot/l/ph 4
dist 1.070e-4 4              # Propagate effective distance equivalent
title Single, step final intensity
plot/watch ex35a_3.plt
plot/l 4
title Single step profile
plot/watch ex35a_4.plt
plot/x/i 4
c                          # to 1.5 micron in air
c Begin distributed steps
c
c Beam 2 contains the initial distribution in air
c
dist .5e-4 2

```

Jump to: [Commands](#), [Theory](#)

```

copy/con 2 3
split/cir/out 2 2.8284e-4 3
c
split/cir/in 3 2.8284e-4 3
add/coh 1 3
dist .5e-4 2
dist .2753e-4 1
c
title Step 1: inside medium
plot/watch ex35a_5.plt
plot/l/i 1
title Step 1: outside of medium
plot/watch ex35a_6.plt
plot/l/i 2
c
c Step 2
c
copy/con 2 3
split/cir/out 2 4e-4 3
c
split/cir/in 3 4e-4 3
add/coh 1 3
dist .5e-4 2
dist .2943e-4 1
c
title Step 2: inside medium
plot/watch ex35a_7.plt
plot/l/i 1
title Step 2: outside of medium
plot/watch ex35a_8.plt
plot/l/i 2
c
c Step 3, last step
c
add/coh 1 2
dist .2943e-4 1
c
title 3 step final profile
plot/watch ex35a_9.plt
plot/x/i 1
gauss/cir/con 2 1. 5e-4 8
abr/lrip 2 .01 1.414 90 90 1e-4
copy/con 2 3
dist 1.5e-4 2
dist 1.070e-4 3
title 1 step with aberration, B 2
plot/watch ex35a_10.plt
plot/x/i 2
title 1 step with aberration, B 3
plot/watch ex35a_11.plt
plot/x/i 3
end
# Copy to Beam 3
# Gaussian aperture, r=2.8284
# supergaussian exponent, n=4
# Complementary aperture
# Add aperture patch to inside beam
# Propagation of Beam 2 by .5 microns
# Propagation of Beam 1 by equivalent
# distance to .5 micron in air

# Copy to Beam 3
# Gaussian aperture, r=4.
# supergaussian exponent, n=4
# Complementary aperture
# Add aperture patch to inside beam
# Propagation of Beam 2 by .5 microns
# Propagation of Beam 1 by equivalent
# distance to .5 micron in air

# Add aperture patch to inside beam
# Propagation of Beam 1 to get to
# a net distance of 1.5 microns

# Gaussian starting distribution
# Impose linear phase ripple

```

Ex35b: Distributed propagation, phase grating in incident beam

Figure 35.6 illustrates the case of an incident collimated beam with a phase grating in the beam. We expect different propagation effects in the inside and outside parts of the beam.

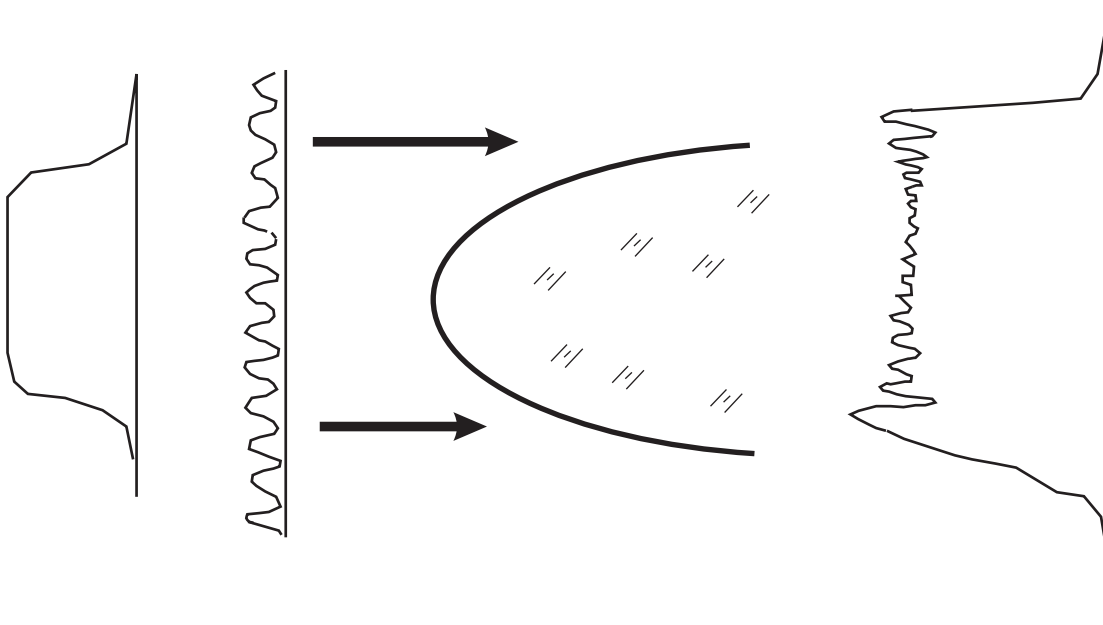


Fig. 35.6. Incident collimated beam with a phase grating. The effective propagation distance is less in the center than the outer zone.

The innermost part of the beam in Example 35a has an effective propagation distance of 1.07 microns and the outermost part has an effective propagation distance of 1.5 microns. Figure 35.7a and 35.7b show the appearance of the beam when propagated 1.5 and 1.07 microns respectively. We expect the propagation through the medium to show aspects of both propagations based on position in the aperture.

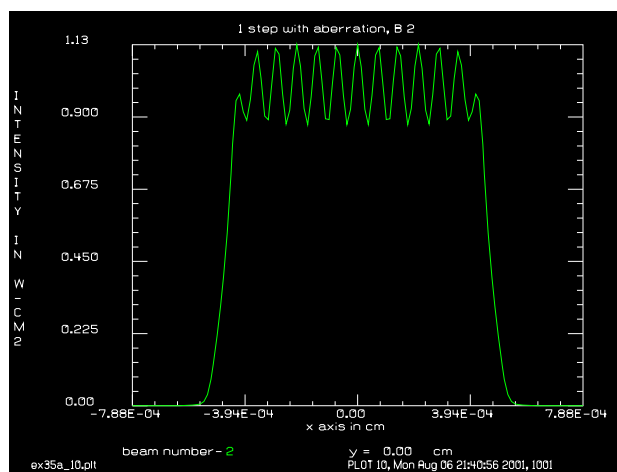


Fig. 35.7a. After 1.5 micron propagation strong variations in intensity are exhibited.

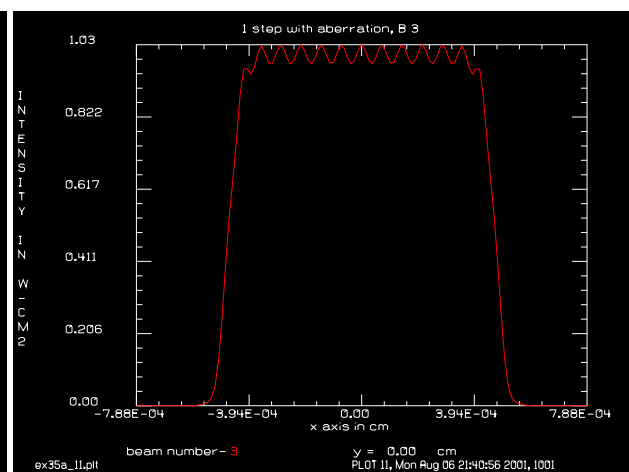


Fig. 35.7b. After 1.07 micron propagation only weak variations in intensity are evident.

The results of propagating the beam with phase effects is shown in Figs. 35.8a and 35.8b.

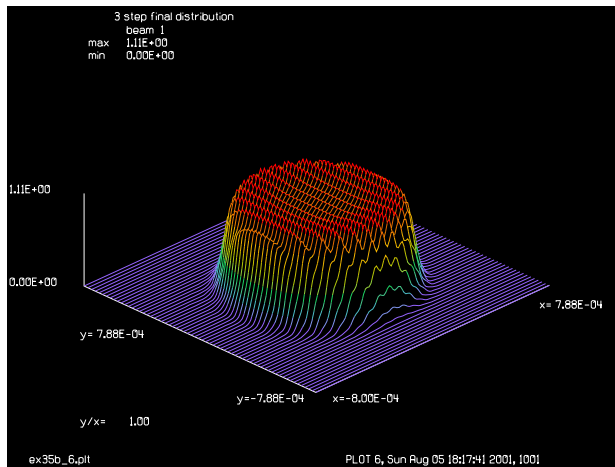


Fig. 35.8a. Isometric plot of the propagation, in 3 steps, of the beam with phase aberrations. The variations in intensity are more pronounced in the outer part of the aperture where the effective propagation length is greater.

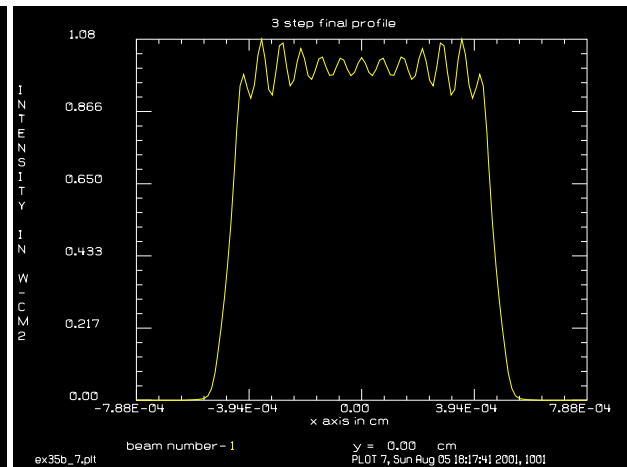


Fig. 35.8b. Profile of the beam after 3 steps of propagation. The diffraction ripples are clearly greater in the outer part of the aperture which has an effective propagation length of about $3/4$ of a characteristic length.

Input: ex35b.inp

```
c## ex35b
c
c Example 35b: Distributed Propagation through a Refractive Surface
c
c In Ex35b, the phase grating is applied to the beam before it
c enters the refractive medium.
c
c BEAMS
c -----
c Beam 1 - net beam amplitude for distributed propagation
c Beam 2 - beam amplitude in portion of array outside refractive surface
c Beam 3 - temporary storage
c
echo/on
array/set 1 128 # Set array size of Beam 1 to 128 x 128
nbeam 3 # Set number of beams to 3
units/s 0 .125e-4 # Set units to .2 microns
wavelength/set 0 .5 # Set wavelengths to .5 microns
clear 1 0 # Initialize Beam 1 to 0
c
c Process Beam 2 as a non-distributed refraction
c
gauss/cir/con 2 1. 5e-4 8 # Gaussian starting distribution
abr/lrip 2 .01 7.071 90 90 5e-4 # Impose linear phase ripple
title starting phase
plot/watch ex35b_1.plt
plot/l/ph 2
c # to 1.5 micron in air
c Begin distributed steps
```

Jump to: [Commands](#), [Theory](#)

```

c
c Beam 2 contains the initial distribution in air
c
dist .5e-4 2
copy/con 2 3                                # Copy to Beam 3
split/cir/out 2 2.8284e-4 3                  # Gaussian aperture, r=2.8284
c                                              # supergaussian exponent, n=4
split/cir/in 3 2.8284e-4 3                   # Complementary aperture
add/coh 1 3                                  # Add aperture patch to inside beam
dist .5e-4 2                                # Propagation of Beam 2 by .5 microns
dist .2753e-4 1                              # Propagation of Beam 1 by equivalent
c                                              # distance to .5 micron in air
title Step 1: inside medium
plot/watch ex35b_2.plt
plot/x/i 1
title Step 1: outside of medium
plot/watch ex35b_3.plt
plot/x/i 2
c
c Step 2
c
copy/con 2 3                                # Copy to Beam 3
split/cir/out 2 4e-4 3                      # Gaussian aperture, r=4.
c                                              # supergaussian exponent, n=4
split/cir/in 3 4e-4 3                       # Complementary aperture
add/coh 1 3                                  # Add aperture patch to inside beam
dist .5e-4 2                                # Propagation of Beam 2 by .5 microns
dist .2943e-4 1                              # Propagation of Beam 1 by equivalent
c                                              # distance to .5 micron in air
title Step 2: inside medium
plot/watch ex35b_4.plt
plot/x/i 1
title Step 3: outside of medium
plot/watch ex35b_5.plt
plot/x/i 2
c
c Step 3, last step
c
add/coh 1 2                                  # Add aperture patch to inside beam
title 3 step final distribution
plot/watch ex35b_6.plt
plot/l/i 1 nsl=64
title 3 step final profile
plot/watch ex35b_7.plt
plot/x/i 1
end

```

Ex35c: Distributed propagation, phase grating on refracting surface

We can add the aberration on the surface of the component, using the distributed propagation technique. Figure 35.9 shows schematically the refractive surface with a high frequency ripple in the surface that acts

Jump to: [Commands](#), [Theory](#)

similarly to the phase grating of the previous example. Example 35c performs this calculation by adding the aberration at the time the aperture is split into the two sections.

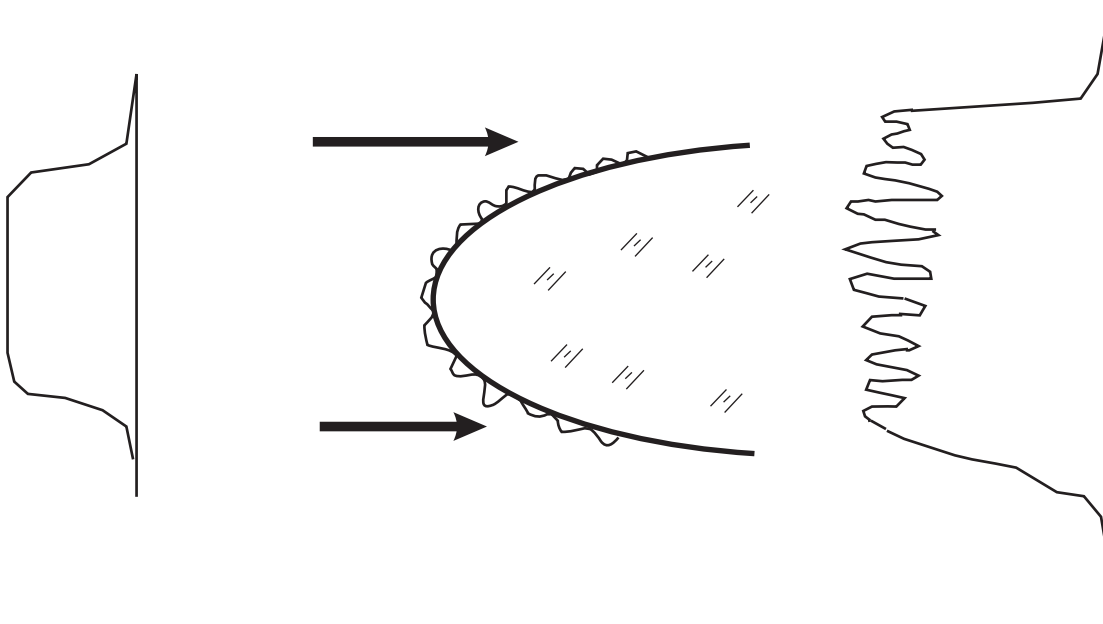


Fig. 35.9. Refractive surface with a phase grating on the surface. The effects of the phase grating begin at the surface so the outer portions show a shorter effective propagation distance.

$$\psi_o(z + \Delta z) = \psi_o(z) - \psi(\Delta x) \quad (35.14)$$

$$\psi_i(z + \Delta z) = \psi_i(z) + \psi(\Delta x)e^{jkW(x,y)} \quad (35.15)$$

where $W(x, y)$ is the aberration. Figure 35.9 shows schematically a refractive surface with a phase grating on the surface.

Performing a similar calculation in several steps, we find the final distribution has a shorter effective propagation length on the outside of the distribution because the aberration is added later in the propagation. Isometric and profile plots are shown in Figs. 35.10a and 35.10b respectively.

The variations in intensity are more pronounced in the outer part of the aperture where the effective propagation length is greater.

Input: ex35c.inp

```
c## ex35c
c
c Example 35c: Distributed Propagation through a Refractive Surface
c
c In this Ex35c, the phase grating is on the refractive surface.
c The propagation is less in the outer parts of the aperture.
c
c All Steps .8277
```

Jump to: [Commands](#), [Theory](#)

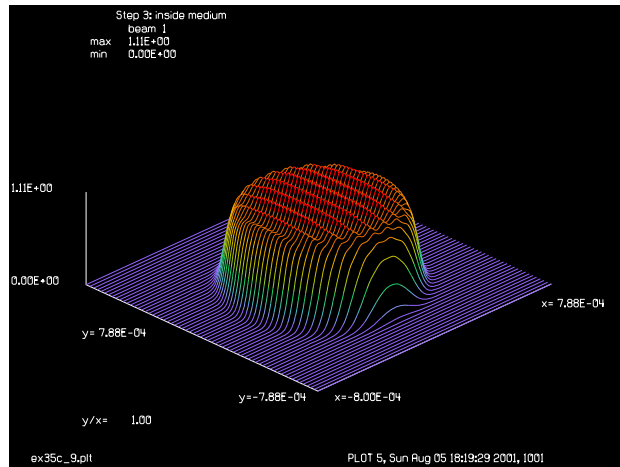


Fig. 35.10a. Isometric plot of the propagation, in 3 steps, of the beam with phase aberrations.

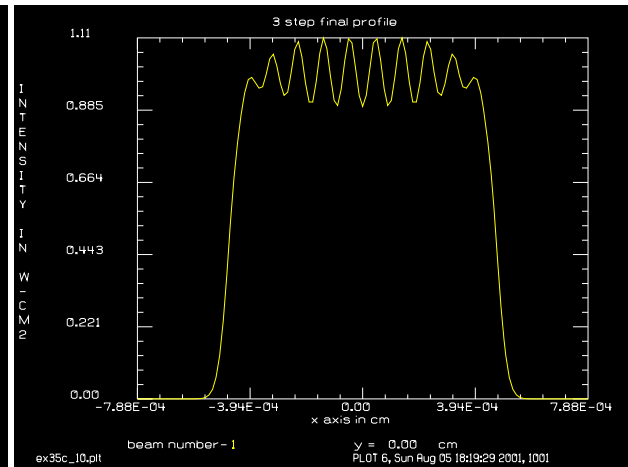


Fig. 35.10b. Profile of the beam after 3 steps of propagation. The diffraction ripples are clearly greater in the outer part of the aperture which has an effective propagation length of about 3/4 of a characteristic length.

```

c
c  BEAMS
c  ----
c  Beam 1   - net beam amplitude for distributed propagation
c  Beam 2   - beam amplitude in portion of array outside refractive surface
c  Beam 3   - temporary storage
c
echo/on
array/set 1 128                                # Set Beam 1 to 128 x 128
nbeam 3                                         # Set number of beams to 3
units/s 0 .125e-4                             # Set units to .2 microns
wavelength/set 0 .5                           # Set wavelengths to .5 microns
clear 1 0                                       # Initialize Beam 1 to 0
c
c  Process Beam 4 as a non-distributed refraction
c
gauss/cir/con 2 1. 5e-4 8                      # Gaussian starting distribution
c                                              #   to 1.5 micron in air
c  Begin distributed steps
c
c  Beam 2 contains the initial distribution in air
c
copy/con 2 3                                   # Copy to Beam 3
split/cir/out 2 2.8284e-4 3                     # Gaussian aperture, r=2.8284
c                                              #   supergaussian exponent, n=4
split/cir/in 3 2.8284e-4 3                      # Complementary aperture
abr/lrip 3 .01 5.496 90 90 5e-4                # Add aberrations
add/coh 1 3                                     # Add aperture patch to inside beam
dist .5e-4 2                                   # Propagation of Beam 2 by .5 microns
dist .2581e-4 1                                # Propagation of Beam 1 by equivalent
c                                              #   distance to .5 micron in air
c
title Step 1: inside medium
plot/watch ex35c_5.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/l 1
title Step 1: outside of medium
plot/watch ex35c_6.plt
plot/l 2
c
c Step 2
c
copy/con 2 3 # Copy to Beam 3
split/cir/out 2 4e-4 3 # Gaussian aperture, r=4.
c # supergaussian exponent, n=4
split/cir/in 3 4e-4 3 # Complementary aperture
abr/lrip 3 .01 5.496 90 90 5e-4 # Add aberrations
add/coh 1 3 # Add aperture patch to inside beam
dist .5e-4 2 # Propagation of Beam 2 by .5 microns
dist .2753e-4 1 # Propagation of Beam 1 by equivalent
c # distance to .5 micron in air
title Step 2: inside medium
plot/watch ex35c_7.plt
plot/l 1
title Step 3: outside of medium
plot/watch ex35c_8.plt
plot/l 2
c
c Step 3, last step
c
abr/lrip 2 .01 5.496 90 90 5e-4 # Add aberrations to last of Beam 2
add/coh 1 2 # Add aperture patch to inside beam
dist .2943e-4 2
c
title Step 3: inside medium
plot/watch ex35c_9.plt
plot/l 1 nsl=64
title 3 step final profile
plot/watch ex35c_10.plt
plot/x/i 1
end

```

Ex35d: Propagation to tilted surface for a phase grating.

Calculation of propagation to a tilted surface for a phase grating. This example first illustrates transfer from a surface normal to the direction of propagation transferred to a tilted surface. Secondly it illustrates propagation to a point 1/3 of the total deflection of the tilted surface.

Input: ex35d.inp

```

c## ex35d
c
c Use Babinet's principle to calculate diffraction to a tilted
c surface for a phase grating.
c
c Set up phase grating and propagate to tilted plane
c Adjust the period of the grating such that the edge of the

```

Jump to: [Commands](#), [Theory](#)

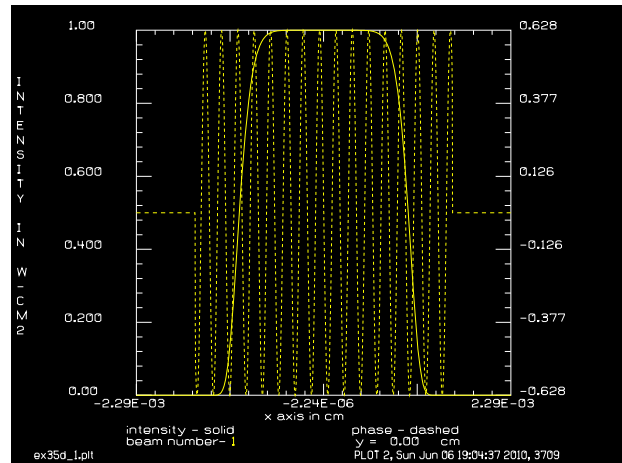


Fig. 35.11. Starting intensity and phase distribution. The phase modulation will create Talbot imaging effects indicating the local propagation distance.

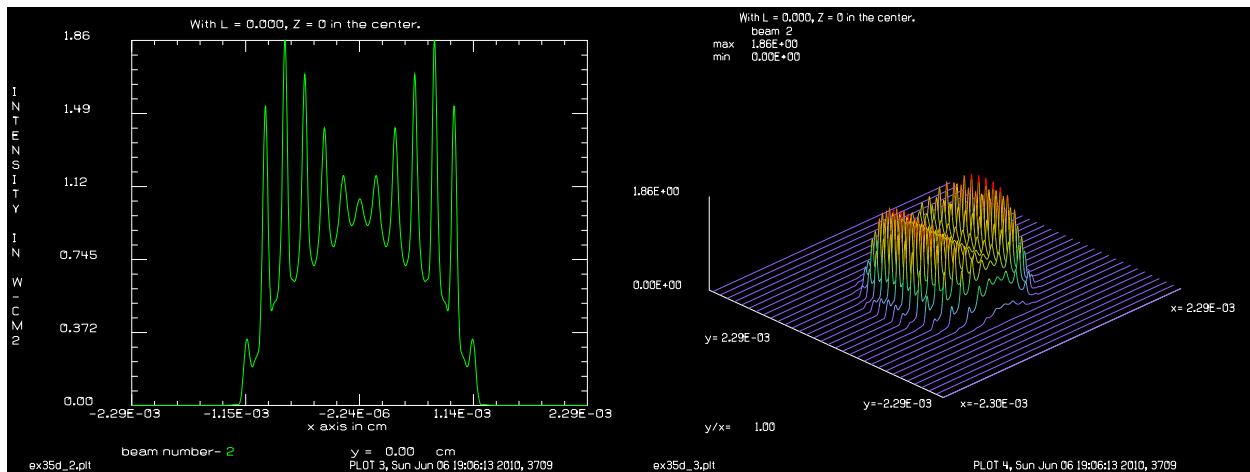


Fig. 35.12a. The phase modulation is transformed into intensit modulation at the left and right side because of the propagation to the plane tilted in the x-plane.

Fig. 35.12b. Isometric plot of the distribution after propagation to the tilted plane.

```

c array is at one quarter Talbot cycle.
c
c Propagate to plane tilted at angle ThetaDeg -- backward for the
c top of the plane and forward for the bottom.
c
c Observe intensity modulation due to propagation of the light
c from the phase grating. The modulation should be large at the edges
c where the propagation length is greatest and essentially zero in the
c center.
c
c Copy one column at a time to an empty array. Propagate the required
c distance for this required distance. Coherently sum into an array.
c
c Case 1: L = 0, just transfer to tilted plane. No net propagation.
c Case 2: L = Propagate to 1/3 slope excursion.
c

```

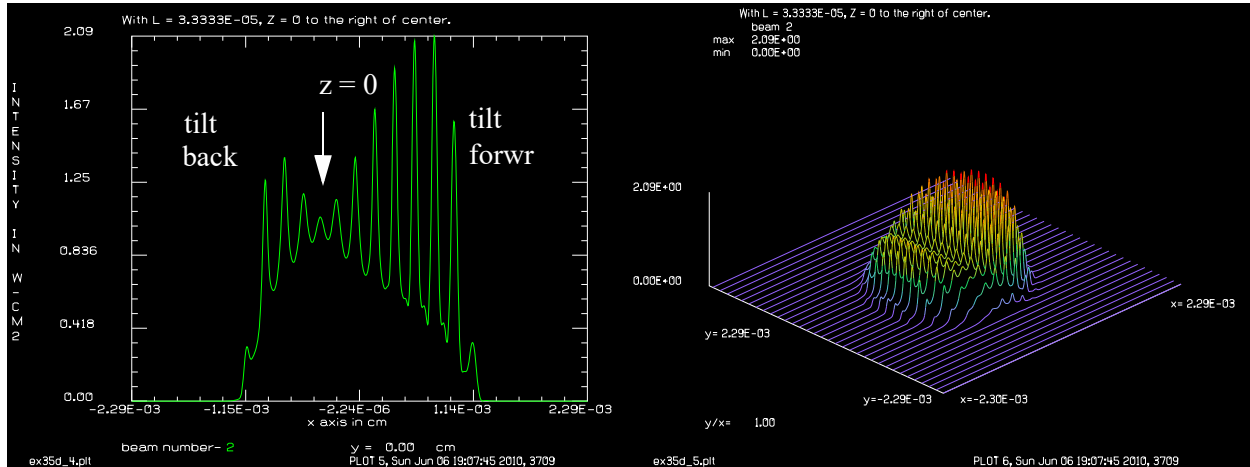


Fig. 35.13a. After propagation to a point such that $z = 0$ at the $1/3$ radius point.

Fig. 35.13b. Isometric plot showing propagation to a tilted plane such that $z = 0$ at the $1/3$ radius point.

```
alias Name ex35d
set/cpu 2
variable/dec/int Nline count
set/highlight/off
Nline = 1024
Ncen = Nline/2+1
ThetaDeg = 5
ThetaRad = ThetaDeg/180*pi
Slope = atan(ThetaRad) list
Lambda = 1e-4 list
Period = 2.*Lambda list
Talbot = Period^2/2./Lambda list
dX = 2.*Talbot/Nline/Slope list
PointsPerCycle = Period/dX list
Units = dX
Chycles = Nline/PointsPerCycle list

array/s 1 Nline
array/s 2 Nline
array/s 3 Nline
units/set 0 Units
wavelength/set 0 Lambda*1e4
Apt = .5*dX*Nline/2 list
gaus/c 1 1 Apt 5
grating/cosine/phase 1 .1 period=Period 90
plot/w @Name_1.plt
plot/x/w 1
plot/x 1
Xmax = dX*Nline/2 list

L = 0
clear 2 0
Lmax = Xmax*Slope list
count = 0
write/off
```

Jump to: [Commands](#), [Theory](#)

```

time/i
macro/run step/Nline
time/h
write/on
time/s
c
c Rescale to get intensity on tilted surface
c
Scale = cos(ThetaRad)
rescale/scale 2 Scale 1.
units/set 2 Units Units
mult 2 Scale
title With L = @L, Z = 0 in the center.
plot/w @Name_2.plt
plot/x/i 2
plot/w @Name_3.plt
plot/l 2

c
c Repeat with a propagation distance of L
c
units/set 2 Units
clear 2 0.
L = Slope*Apt/3      # Propage so one third of radius is at z =0.
count = 0
write/off
time/i
macro/run step/Nline
time/h
write/on
time/s
c
c Rescale to get intensity on tilted surface
c
Scale = cos(ThetaRad)
rescale/scale 2 Scale 1.
units/set 2 Units Units
mult 2 Scale
title With L = @L, Z = 0 to the right of center.
plot/w @Name_4.plt
plot/x/i 2
plot/w @Name_5.plt
plot/l 2
end
macro/def step/o
    count = count + 1
    x = Units*(count - Ncen)
    dZ = x*Slope
    array/reset 3
    clear 3 0
    zreff 3 0.
    copy/col 1 3 count count
    LocalL = L + dZ
    prop LocalL 3

```

Jump to: [Commands](#), [Theory](#)


```

add/coh/con 2 3
if [mod(count,100)==0] then
    CC Step @count of @Nline
endif
macro/end

```

Ex35e: Propagation to tilted surface for a circular aperture.

Calculation of propagation to a tilted surface from a circular aperture. An initial propagation puts the left-hand side at $z = 0$.

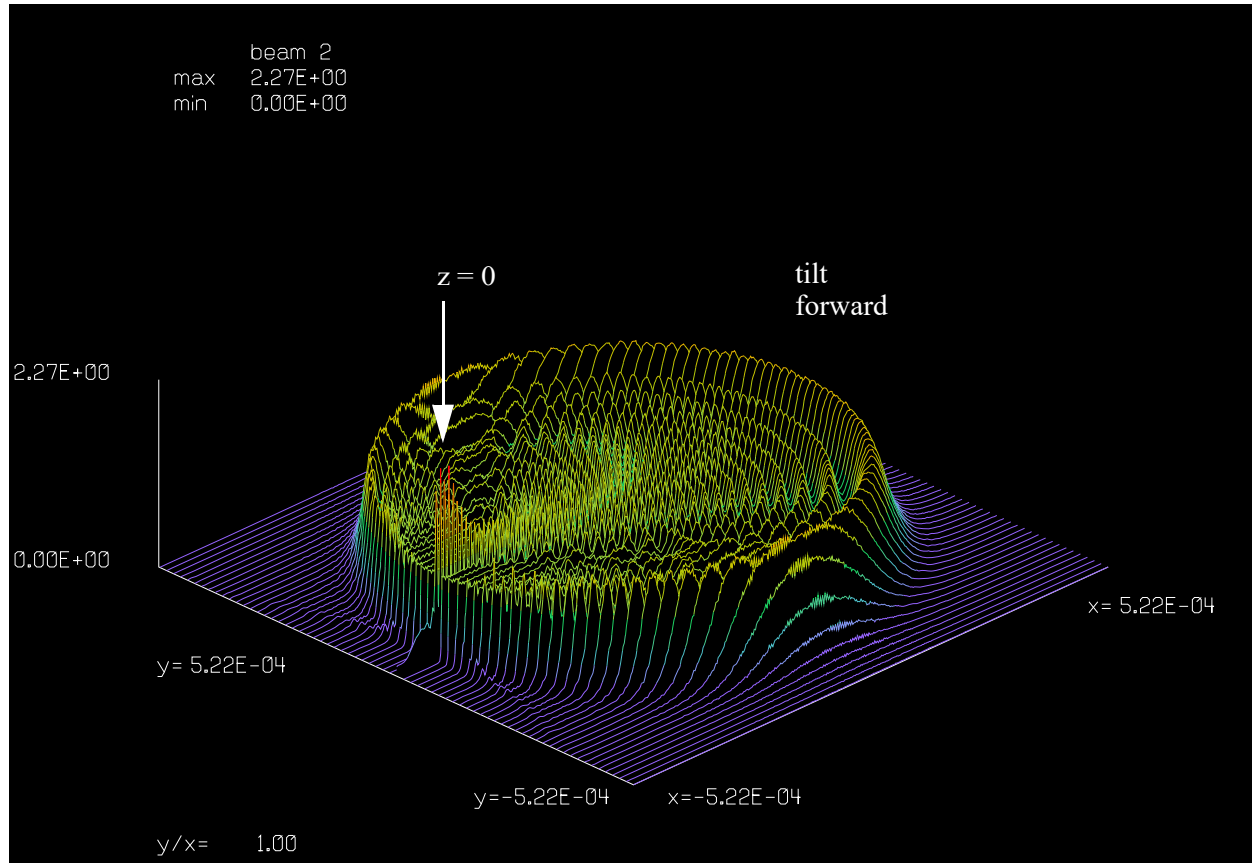


Fig. 35.14. Propagation of a circular aperture to a tilted plane pivoted about the left point at $z = 0$ has zero propagation distance and the tilt causes all points to the right to be forward propagated. Note propagation rings are quite narrow at the left because of short propagation distance and are wider on the right because of greater propagation distance.

Input: ex35e.inp

```

c## tilt
c
c Use Babinet's principle to calculate diffraction to a tilted
c surface.
c
c Set up phase grating and propagate to tilted plane
c Adjust the period of the grating such that the edge of the

```

Jump to: [Commands](#), [Theory](#)

```

c array is at one quarter Talbot cycle.
c
c Propagate to plane tilted at angle ThetaDeg -- backward for the
c top of the plane and forward for the bottom.
c
c Observe intensity modulation due to propagation of the light
c from the phase grating. The modulation should be large at the edges
c where the propagation length is greatest and essentially zero in the
c center.
c
c Copy one column at a time to an empty array. Propagate the required
c distance for this position. Coherently sum into an array.
c
set/cpu 2
alias Name ex35e
variable/dec/int Nline count
set/highlight/off
Nline = 1024
Ncen = Nline/2+1
ThetaDeg = 5
ThetaRad = ThetaDeg/180*pi
Slope = atan(ThetaRad) list
Lambda = 1e-4 list
Cycles = 2.5
Fraction = 1./4.      # fraction of quarter Talbot cycle
c Z_talbot = Period^2/(2.*Lambda)
c Apt = Z_talbot*Fraction/Slope*2.Lambda      # 1/4 of 1/4 Talbot cycle
c Period = Apt/Cycles
c Apt = Cycles/Slope
c Apt = Cycles^2*2.*Lambda*Slope/Fraction
Apt = Cycles^2*2.*Lambda*Slope/Fraction list      # 1/4 of 1/4 Talbot cycle
Apt = 4.35229738E-04
dZMax = Apt*Slope list
Period = Apt/Cycles list
dX = 3*Apt/Nline*2 list
Fn = Apt^2/(Lambda*dZMax) list
Z_talbot = Period^2/(2.*Lambda) list
dX=
Apt=
H = Slope*Apt list
L = H
Units = dX

array/s 1 Nline
array/s 2 Nline
array/s 3 Nline
units/s 0 Units
wavelength/set 0 Lambda*1e4

clap/cir 1 Apt
c grating/cosine/phase 1 .02 period=Period 90

c set/label/off
plot/w @Name_1.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/x/w 1
plot/x/i 1 le=-1.2*Apt ri=1.2*Apt
Xmax = dX*Nline/2 list
macro/def step/o
    count = count + 1
    x = Units*(Ncen-count)
    dZ = -x*Slope
    udata/set count count x dZ
    array/reset 3
    clear 3 0
    zreff 3 0.
    copy/col 1 3 count count
    LocalL = L + dZ
    prop LocalL 3
    add/coh/con 2 3
    if [mod(count,100)==0] then
        CC Step @count of @Nline
    endif
macro/end
clear 2 0
Lmax = Xmax*Slope list
count = 0
write/off
time/i
macro/run step/Nline
time/h
write/on
time/s
c
c Rescale to get intensity on tilted surface
c
Scale = cos(ThetaRad)
rescale/scale 2 Scale 1.
units/s 2 Units Units
mult 2 Scale
plot/w @Name_2.plt
plot/x/i 2 le=-1.2*Apt ri=1.2*Apt
plot/w @Name_3.plt
plot/l 2 xrad=1.2*Apt ns=64

```


Ex36: Finite-difference propagator

Table. 36.1. Table of Ex36 examples

Ex36a: Finite-Difference Propagator (FDP), Soft Aperture	1
Ex36b: Finite-Difference Propagator, Hard Aperture with FFT	4

This example illustrates use of the finite difference propagator (FDP). The FDP has the advantage of being much faster than the FFT propagator but is highly subject to numerical instability at discontinuities. The FDP directly solves the parabolic scalar wave equation.

$$\frac{\partial \psi}{\partial z} = \frac{1}{2jk} \left[\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right] \psi(x, y, z) \quad (36.1)$$

The second derivatives are taken from the discrete complex amplitude distribution by

$$\frac{\partial^2 A}{\partial x^2} \approx \frac{A(I+1, J) + A(I-1, J) - 2A(I, J)}{2\Delta x^2} \quad (36.2)$$

$$\frac{\partial^2 A}{\partial y^2} \approx \frac{A(I, J+1) + A(I, J-1) - 2A(I, J)}{2\Delta y^2} \quad (36.3)$$

where $A(I, J)$ is an element of the complex amplitude array and I and J are the indices. The second derivative is taken by considering only the neighboring points. In the area of a discontinuity the numerical procedure blows up.

In this example we shall consider propagation with the following aspects

The maximum distance allowed with the FDP according to Rench is

$$\Delta z_{max} = \frac{k}{2} \Delta x^2 = 1.852354 \quad (36.4)$$

for the conditions of Table 36.2.

To resolve a Fresnel number of 10 we need, $8F_n$ points across the diameter. With 80 points across the diameter, the sampling is $\Delta x = 0.025$. Choosing an array size of 128×128 gives a reasonable guard band. To go the full distance of 94.339623 cm, we need at least 50 FDP steps, each of length 1.886792.

Ex36a: Finite-Difference Propagator (FDP), Soft Aperture

In Example 36a, we will use the supergaussian function to make the soft amplitude roll-off. We show the amplitude distribution after 10 propagation steps and 20 propagation steps, each step is of length 1.886792. The initial distribution shows the rounded edges of the intensity profile. After 10 steps to go to $z = 18.86792$, the FDP shows somewhat odd diffraction phenomena at the aperture edge. At 20 steps, $z =$

37.735849, the FDP propagation shows very bad errors and further propagation will make them much worse.

Table. 36.2. Effective propagation distances outside and inside the medium.

Parameter	Value
wavelength	1.06×10^{-3}
aperture radius	1 cm
Fresnel number	10

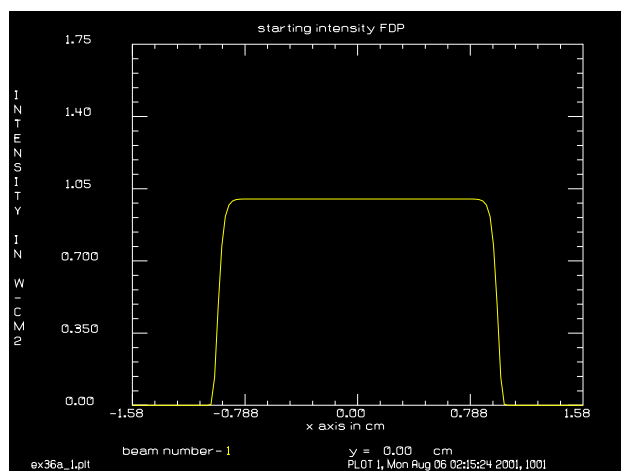


Fig. 36.1a. Starting distribution for the FDP propagator.

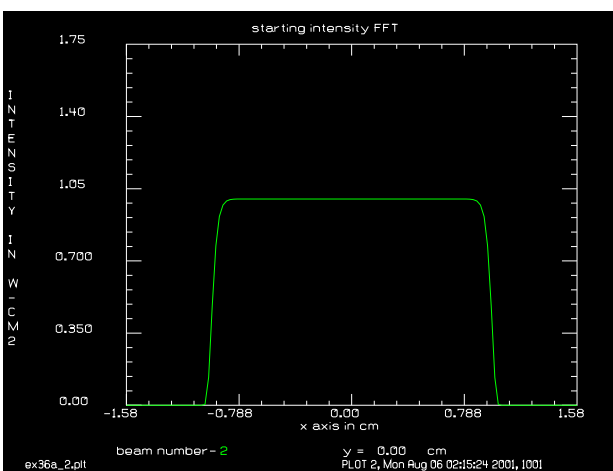


Fig. 36.1b. Starting distribution for the FFT propagator (the same as for the FDP).

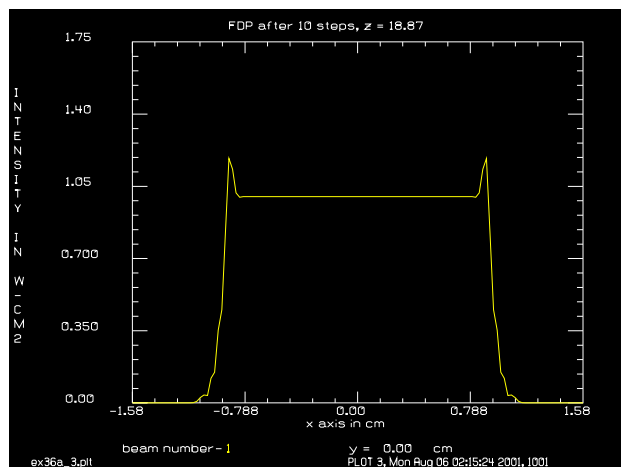


Fig. 36.2a. Distribution at 18.87 cm after 10 FDP steps. The diffraction effects show some accuracies at the edge.

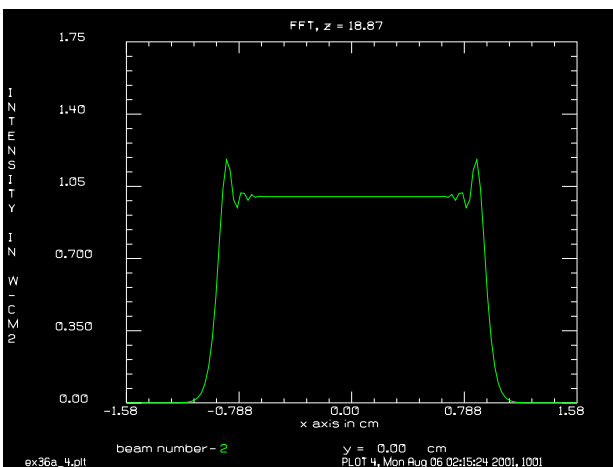


Fig. 36.2b. Distribution at 18.87 cm after FFT propagation.

Input: `ex36a.inp`

```
c## ex36a
c
c Example 36a: Finite-Difference Propagator (FDP), Soft Aperture
c
```

Jump to: [Commands](#), [Theory](#)

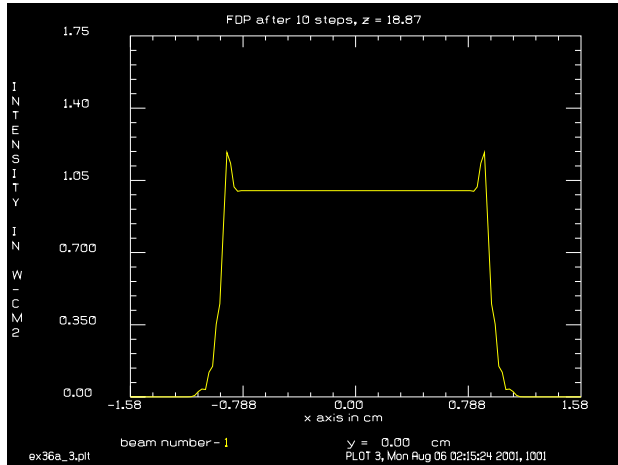


Fig. 36.3a. Distribution at 37.74 cm after 20 FDP steps. The errors are very bad and will only get worse with further propagation.

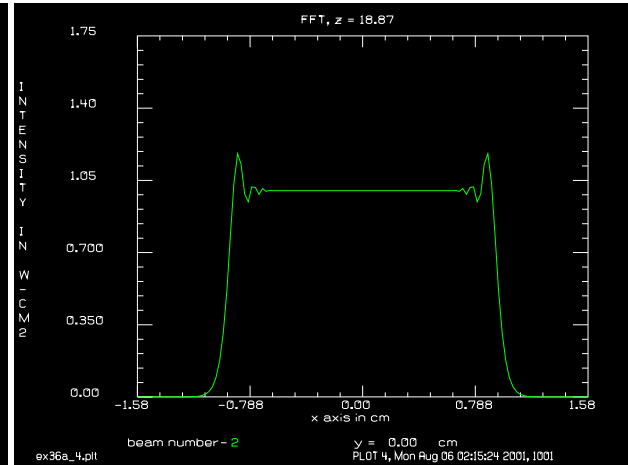


Fig. 36.3b. Distribution at 37.74 cm after FFT propagation.

```

c The parameter used in the example are
c
c wavelength          1.06 x 10-3 cm
c aperture radius     1 cm
c
c The maximum distance allowed with the FDP according to Rench is
c
c    $\text{deltaz}(\text{max}) = k / (2 * \text{deltax}^2) = 1.852354$ 
c
echo/on
array/set 1 128          # Set all beams to 128 x 128
nbeam 2                  # Set number of beams to 2
units/s 0 .025           # Set units/s to .025 cm
gauss/cir/con 0 1 1 20   # Gaussian starting distribution
c                         # to make soft aperture

title starting intensity FDP
plot/watch ex36a_1.plt
plot/x/in 1 fmin=0 fmax=1.75
title starting intensity FFT
plot/watch ex36a_2.plt
plot/x/in 2 fmin=0 fmax=1.75
macro/def fdp/o          # define macro for FDP
    dist/finitdif 1.886792 1
macro/end
macro/run fdp/10
dist 18.86792 2
title FDP after 10 steps, z = 18.87
plot/watch ex36a_3.plt
plot/x/in 1 fmin=0 fmax=1.75
title FFT, z = 18.87
plot/watch ex36a_4.plt
plot/x/in 2 fmin=0 fmax=1.75
macro/run fdp/10
dist 18.86792 2

```

Jump to: [Commands](#), [Theory](#)

```

title FDP after 10 steps, z = 37.74
plot/watch ex36a_5.plt
plot/x/in 1 fmin=0 fmax=1.75
title FFT, z = 37.74
plot/watch ex36a_6.plt
plot/x/in 2 fmin=0 fmax=1.75
end

```

Ex36b: Finite-Difference Propagator, Hard Aperture with FFT

Example 36b, uses a hard aperture, but propagates $z = 94.339623 - 1.886792$ with the FFT propagator and the last step with the FDP. The results are very similar, illustrating the fact that after propagation away from the hard aperture the second derivatives are well behaved and the FDP works relatively well.

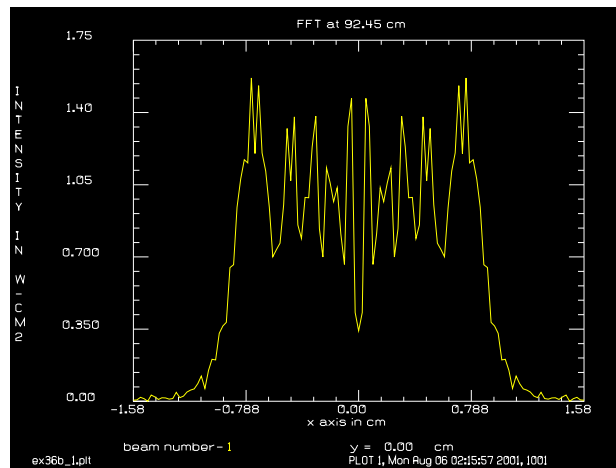


Fig. 36.4. Distribution at 92.45 after FFT propagation.

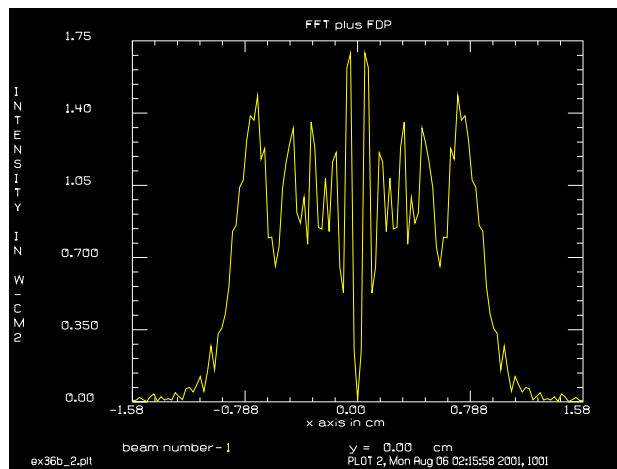


Fig. 36.5a. Distribution at 94.34 cm after 1 FFT step to 92.45 and the last 1.88 cm with the FDP steps. The results are close to the FFT propagation.

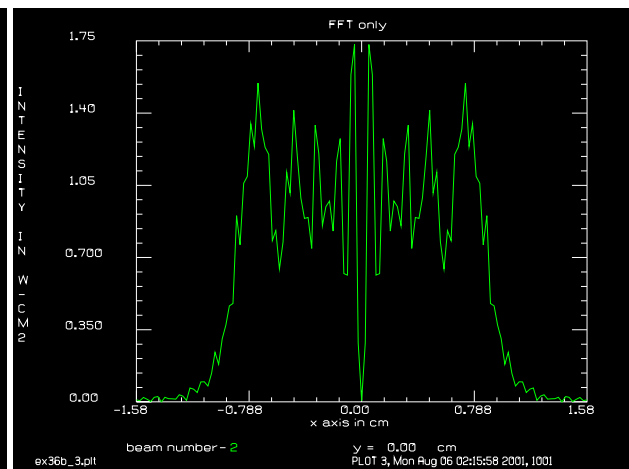


Fig. 36.5b. Distribution at 94.34 cm after FFT propagation.

Input: ex36b.inp

```

c## ex36b
c
c Example 36b: Finite-Difference Propagator, Hard Aperture with FFT
c
c Finite-difference propagator (FDP), hard aperture with FFT propagation
c to soften discontinuities. In this example, both beams are propagated
c to a Fresnel number of 10. Beam 1 is propagated 92.45 cm by the FFT
c and then the remaining distance of 1.88 cm by the FDP. Beam 2 is
c propagated directly to 94.33 in one step.
c
c By first propagating with the FFT, the discontinuities are removed and
c the FDP is more accurate.
c
echo/on
array/set 1 128 # Set all beams to 128 x 128
nbeam 2 # Set number of beams to 2
units/s 0 .025 # Set units to .025 cm
clap/cir/con 0 1 # hard aperture
dist 92.452831 1 # FFT propagation to 92.45, Beam 1
title FFT at 92.45 cm
plot/watch ex36b_1.plt
plot/x/in 1 fmin=0 fmax=1.75
macro/def fdp/o # define macro for FDP
    dist/finitdif 1.886792 1
macro/end
macro/run fdp/1 # FDP propagation to 94.33, Beam 1
dist 94.339623 2 # FFT propagation to 94.33, Beam 2
title FFT plus FDP
plot/watch ex36b_2.plt
plot/x/in 1 fmin=0 fmax=1.75
title FFT only
plot/watch ex36b_3.plt
plot/x/in 2 fmin=0 fmax=1.75
end

```


Ex37: Polarization, Jones matrices

Table. 37.1. Table of Ex37 examples

Ex37a: Polarization, Jones matrices	1
Ex37b: Polarization, surface polarization effects	5
Ex37c: Transmission and reflection coefficients for light incident at Brewster's angle	8
Ex37d: Polarization, Goos-Hanchen, Part 1	10
Ex37e: Polarization, Goos-Hanchen with <code>j surf/goos</code> , Part 2.	11
Ex37f: Birefringent wedge using 3d addressing of polarization sheets.	12
Ex37g: Calculate polarization properties through a Dove prism.	15

This example illustrates polarization modeling. Effects included in this example are:

```
jones/set          ar ai br bi cr ci dr di
jones/list
jones/multiply     kbeam
jones/left         kbeam
jones/right        kbeam
jones/retard       kbeam azimuth phi
```

Ex37a: Polarization, Jones matrices

The beam is initialized to a gaussian distribution of with $\omega_0 = 10$ and linearly polarized in the x-direction. The starting distribution in the center is

$$E = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \quad (37.1)$$

The E_x and E_y components are shown in real-imaginary space by the command `plot/vector`. A more conventional elliptical plot is displayed by the command `plot/elliptical`.

The linear x-polarization is rotated 45° clockwise by a Faraday rotator with the command `jones/faraday`. The rotated vector is

$$\begin{bmatrix} (0.707, 0) \\ (-0.707, 0) \end{bmatrix} = \begin{bmatrix} (0.707, 0) & (0.707, 0) \\ (-0.707, 0) & (0.707, 0) \end{bmatrix} \begin{bmatrix} (1, 0) \\ (0, 0) \end{bmatrix} \quad (37.2)$$

The beam then goes through a retarder with the fast axis vertical and with retardance of $\pi/3$. The elliptical plot shows that the beam is elliptically polarized. The primary polarization is left circular.

$$\begin{bmatrix} (0.354, 0.612) \\ (-0.354, 0.612) \end{bmatrix} = \begin{bmatrix} e^{j\pi/3} & (0, 0) \\ (0, 0) & e^{-j\pi/3} \end{bmatrix} \begin{bmatrix} (0.707, 0) \\ (-0.707, 0) \end{bmatrix} \quad (37.3)$$

A right circular polarizer is used to extract the right circular polarization. Right handed circular polarization is displayed as a clockwise rotation.

$$\begin{bmatrix} (-0.129, 0.129) \\ (0.129, 0.129) \end{bmatrix} = \frac{1}{2} \begin{bmatrix} (1, 0) & (0, 1) \\ (0, -1) & (1, 0) \end{bmatrix} \begin{bmatrix} (0.354, 0.612) \\ (-0.354, 0.612) \end{bmatrix} \quad (37.4)$$

The beam is reinitializes and put into linear polarization at 45° again. A clear aperture of radius 10 limits the beam. The beam then passes through a refractive surface of radius $R = 12$. The incident index is 1 and the index of the refractive medium is 4.

These extreme values are used to enhance the polarization effects for purposes of illustration. A lens command would have to be added as well to add the appropriate convergence.

The elliptical polarization plot shows asymmetry characteristic of linear polarization on a curved surface with Fresnel reflection properties. The isometric plot of intensity also shows asymmetry.

The intensity varies across the pupil because of the differences in the reflectivity of the s - and p -polarizations. Fig. 37.6 shows a sketch of the polarization state and the local polarization coordinate system across the pupil. The polarization state was made linear at 45° . The polarization coordinate system rotates so that p always lies on a radius drawn from the center of the aperture. The transmission coefficients for p and s are both 0.566 at normal incidence and 0.626 and 0.429 at the edge of the aperture. The example was chosen to have extreme values of an index of 4, an aperture radius of 10, and a surface radius of 12. This exaggerates the effects. In the NE and SW quadrants, the polarization of the light is parallel to p and the higher transmission p -coefficient applies. In the NW and SE quadrants, the polarization of the light is parallel to s and the lower s -coefficient applies.

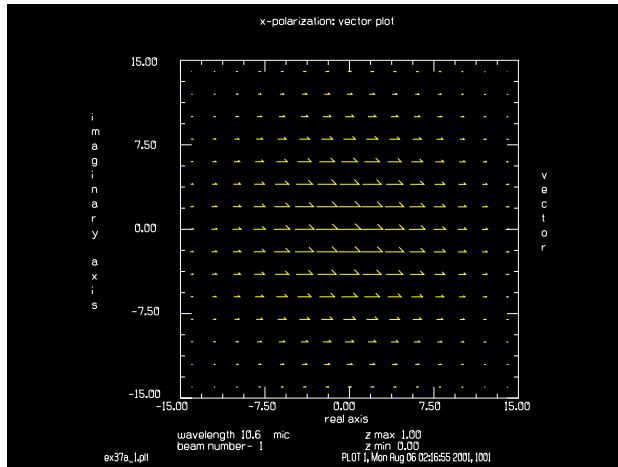


Fig. 37.1a. Plot in real-imaginary form, with the horizontal axis real and the imaginary axis vertical. E_x is the vector with the half arrow head on the counterclockwise side.

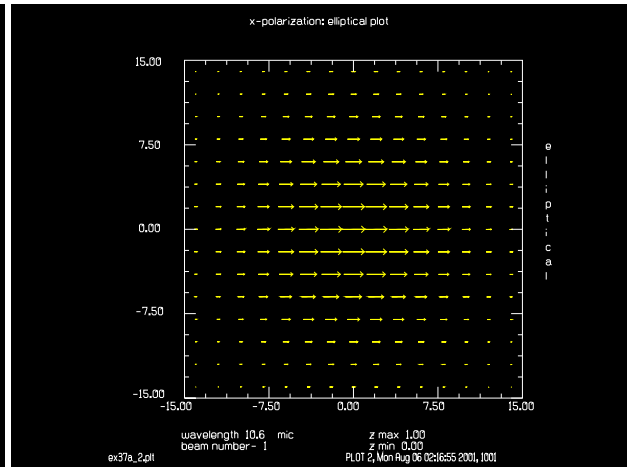


Fig. 37.1b. Elliptical polarization plot. The size of the ellipse indicates the complex amplitude. The plots indicate the locus swept out over one optical time cycle. Clockwise rotation indicates a component of right handed circular polarization.

Input: `ex37a.inp`

c## `ex37a!553560762282639`

Jump to: [Commands](#), [Theory](#)

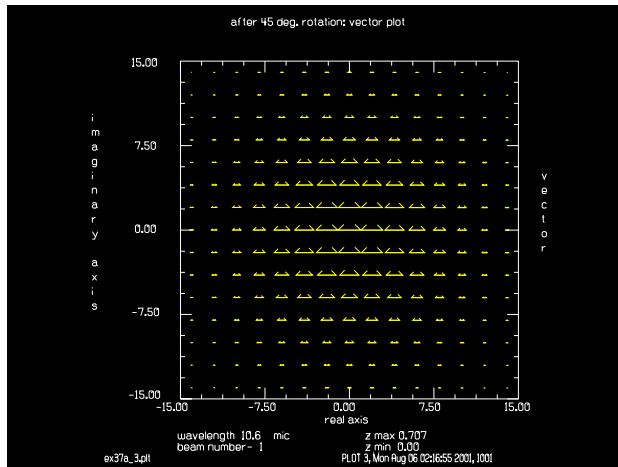


Fig. 37.2a. Plot in real-imaginary form showing the linear polarization at 45° .

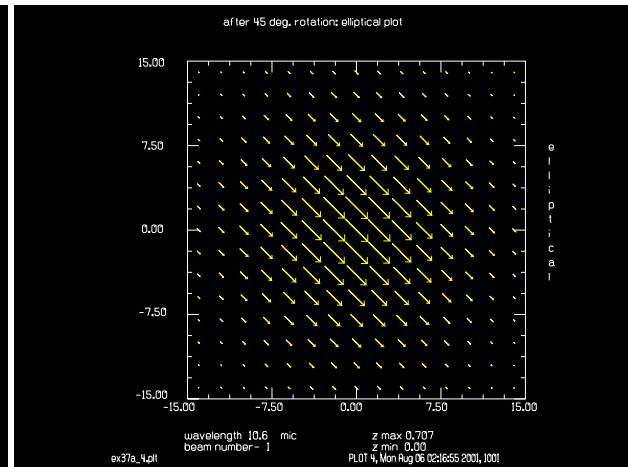


Fig. 37.2b. Elliptical polarization plot showing the linear polarization at 45° .

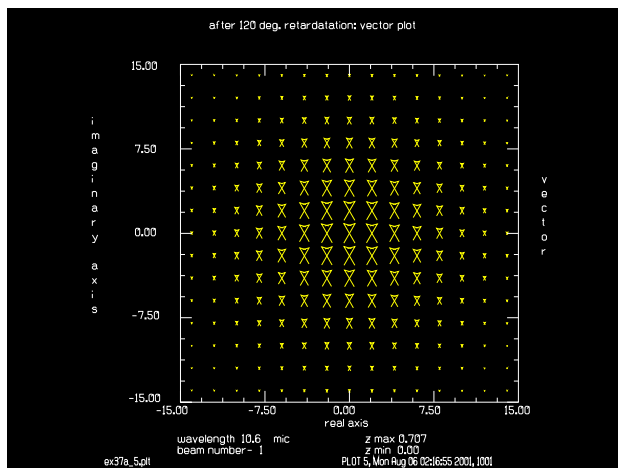


Fig. 37.3a. Plot in real-imaginary form after a retarder is used to put the beam into elliptical polarization—mostly left handed circular.

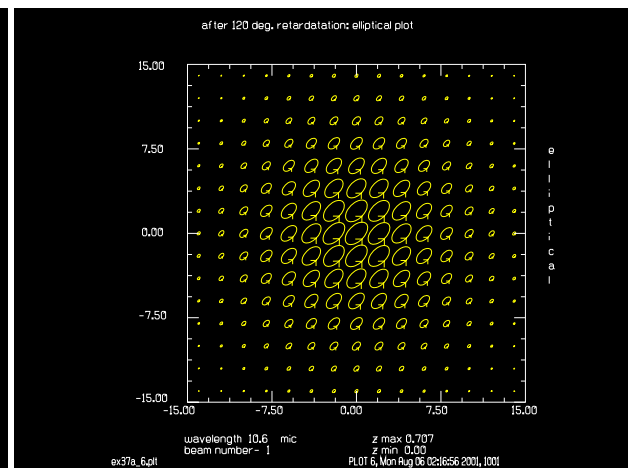


Fig. 37.3b. Elliptical polarization plot showing mostly left handed circular polarization.

```
c
c Example 37a: Polarization
c
echo/on
array/s 1 32 ip=1
units/s 1 1
gaus/c/c 1 1 10
set/density 13 13
title x-polarization: vector plot
field
plot/watch ex37a_1.plt
plot/vector 1
title x-polarization: elliptical plot
plot/watch ex37a_2.plt
plot/elliptical 1
```

```
# 32 x 32 array with polarization
# set units to 1
# gaussian of width 10

# real-imaginary plot

# elliptical plot
```

Jump to: [Commands](#), [Theory](#)

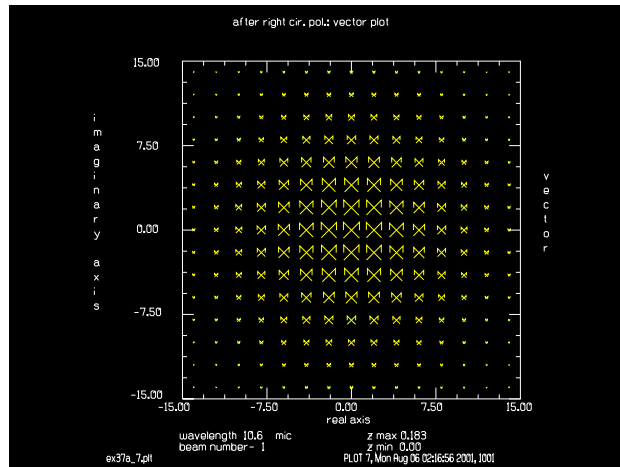


Fig. 37.4a. Plot in real-imaginary form after right handed circular polarizer.

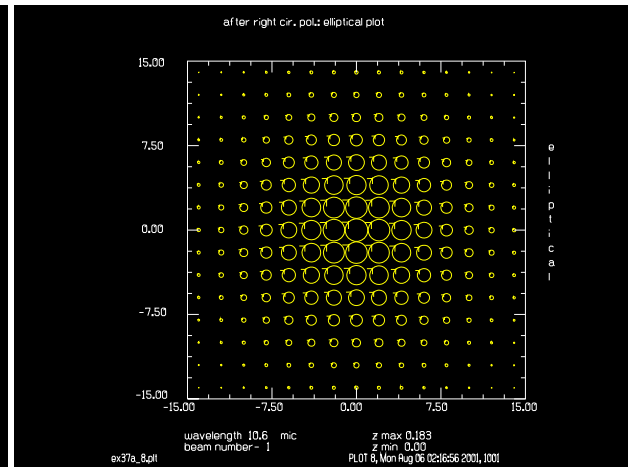


Fig. 37.4b. Elliptical polarization plot showing right handed circular polarization.

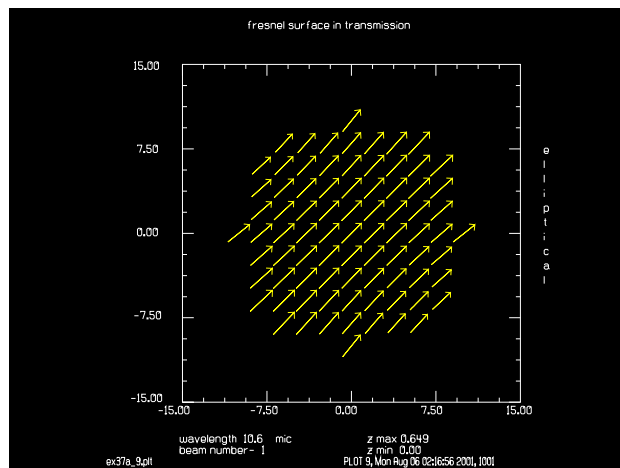


Fig. 37.5a. Elliptical polarization plot after Fresnel transmission effects. Slight asymmetric changes are noticeable in the outer parts of the beam.

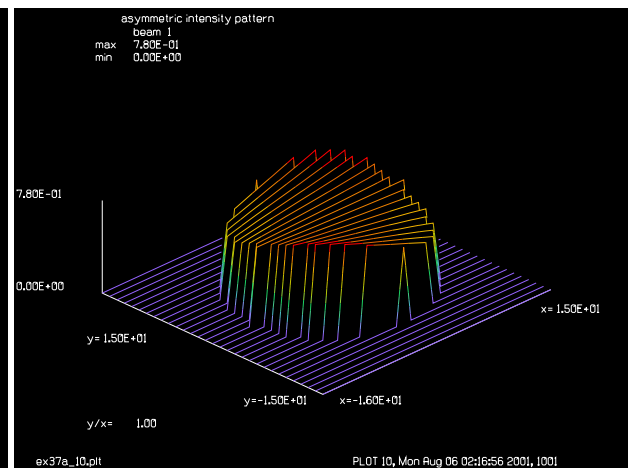


Fig. 37.5b. The plot of intensity also shows slight asymmetric changes due to Fresnel transmission effects.

```
jones/faraday 1 45                                # 1/8 wave faraday rotator
title after 45 deg. rotation: vector plot
plot/watch ex37a_3.plt
plot/vector 1
title after 45 deg. rotation: elliptical plot
plot/watch ex37a_4.plt
plot/elliptical 1
jones/retard 1 0 120                               # 1/3 wave retarder
c                                                    # causes elliptical polarization
title after 120 deg. retardation: vector plot
plot/watch ex37a_5.plt
plot/vector 1
title after 120 deg. retardation: elliptical plot
plot/watch ex37a_6.plt
plot/elliptical 1
```

Jump to: [Commands](#), [Theory](#)

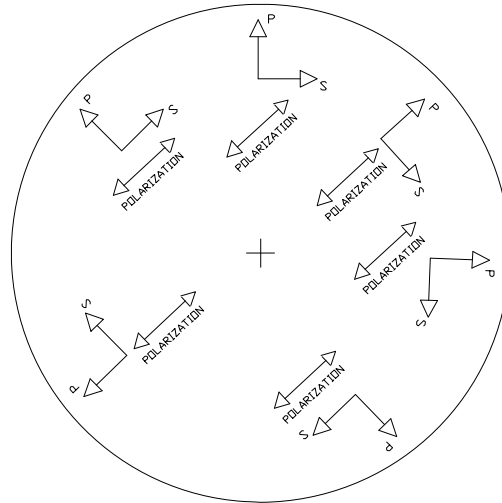


Fig. 37.6. The polarization state of the incident light is constant over the aperture — linear at 45° . The p-transmission coefficient rises away from normal incidence and the s coefficient falls. The light is largely in the p state in the NE and SW quadrants and in the s state in the NW and SE quadrants. The intensity plot in Fig. 37.5b shows the asymmetric effect of the Fresnel transmission coefficients on the transmitted beam.

```
jones/right 1
title after right cir. pol.: vector plot
plot/watch ex37a_7.plt
plot/vector 1
title after right cir. pol.: elliptical plot
plot/watch ex37a_8.plt
plot/elliptical 1
clear 1 1                                # reinitialize array
clap/c/c 1 10
jones/set ar=.707 bi=0 cr=.707 ci=0 dr=0 # define jones operator
jones/multiply 1                          # apply jones operator
c
c Array is now linearly polarized at 45 deg
c
jsurf/fresnel 1 1. 4. 12 12              # define air-glass surface
title fresnel surface in transmission
plot/watch ex37a_9.plt
plot/elliptical 1
title asymmetric intensity pattern
plot/watch ex37a_10.plt
plot/l
end
```

Ex37b: Polarization, surface polarization effects

The command `jsurf` models Fresnel transmission and reflection and single layer coatings, such as antireflection coatings. The surface may be curves and tilted. The treatment, described in GLAD Theory Manual, Chapt. 14, is a combination of paraxial angles for the surface curvature and real angles for vertex tilt. Figs. 37.7 and Fig. 37.8 show Fresnel transmission and reflection for an air-to-glass interface with the glass having an index of refraction of 4.0. Fig. 37.9 shows recombination of the transmitted and reflected

Jump to: [Commands](#), [Theory](#)

beam to show energy conservation. The transmitted beam, Beam 1, is elliptical because the vertex is tilted by 15, so the irradiance for Beam 1 must be multiplied by the inverse of the one-dimensional magnification factor before taking the irradiance sum.

A single layer antireflection coating will have perfect performance if the film layer with index N_f conforms to the equation

$$N_f = \sqrt{\frac{N_s}{N_i}} \quad (37.5)$$

where N_i is the initial index and N_s is the substrate index. For a substrate index of 4 and incident index of 1, the single-layer antireflection coating consists of a quarter-wave of a film of index 2. Fig. 37.10 shows transmission for a curved, tilted surface. The transmission has a peak value of 100%. Fig. 37.11 shows the reflected energy. Fig. 37.12 shows the incoherent sum of transmitted and reflected energy, demonstrating energy conservation. It is necessary to correct the irradiance of the transmitted beam for its elliptical profile, as described above, before making the irradiance sum.

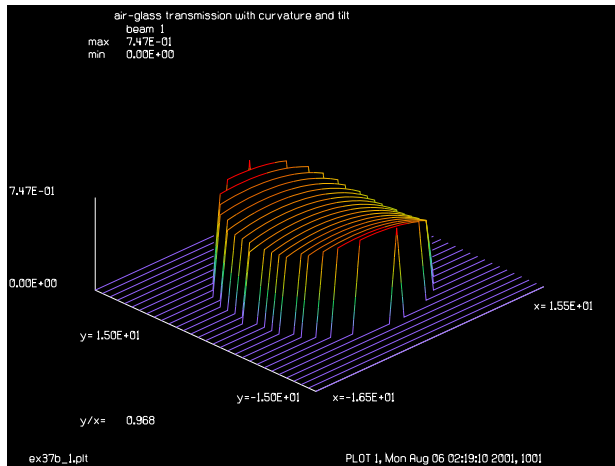


Fig. 37.7. Fresnel transmission, curved surface, $T_y = 15^\circ$.

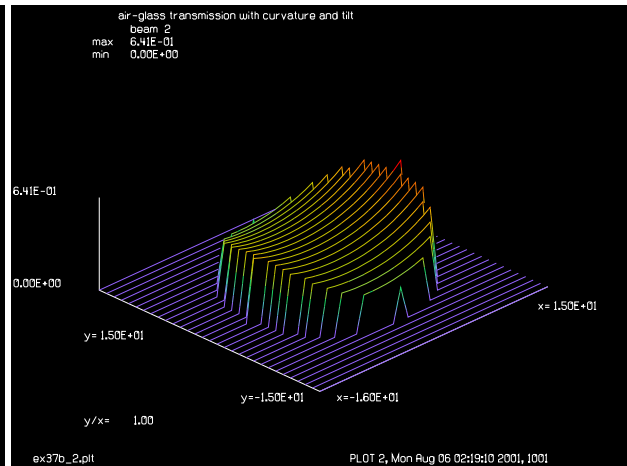


Fig. 37.8. Fresnel reflection, curved surface, $T_y = 15^\circ$. Renormalized.

Input: `ex37b.inp`

```
c## ex37b!519359051645297
c
c Example 37b: Surface and single layer thin film effects
c
echo/on
array/s 1 32 ip=1                                # 32 x 32 array with polarization
nbeam 3
wavelength/set 0 1
clear 1 1                                           # reinitialize array
clap/c/c 1 10
c
c Establish vertical polarization
c
```

Jump to: [Commands](#), [Theory](#)

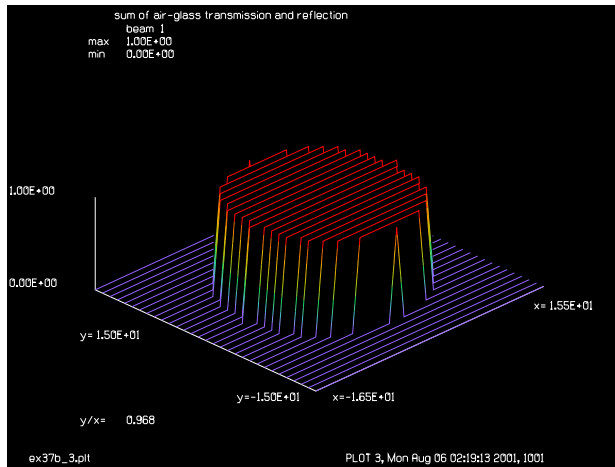


Fig. 37.9. Recombination of Fresnel transmission and reflection, showing conservation of energy.

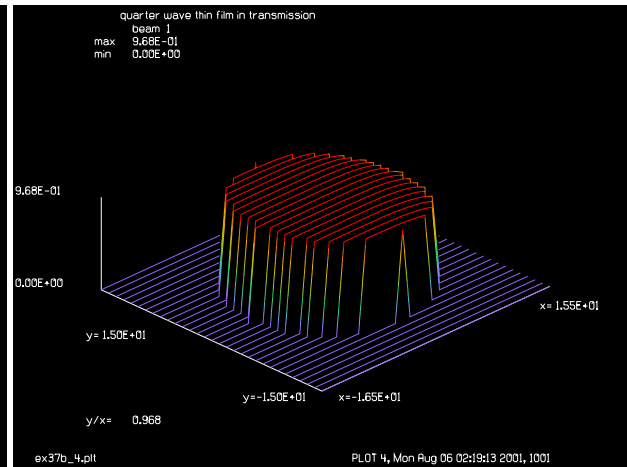


Fig. 37.10. Transmission through antireflection coating, curved surface, $T_y = 15^\circ$.

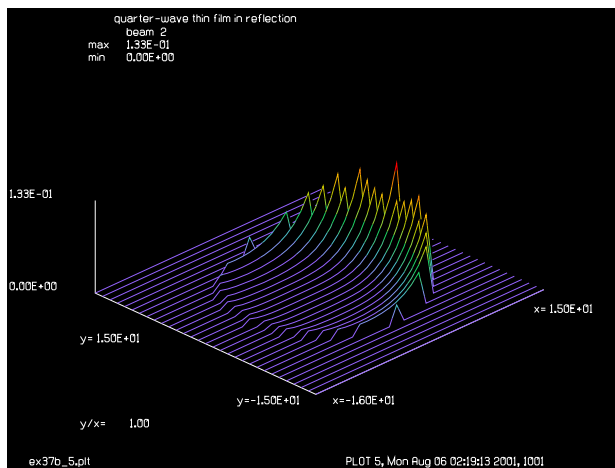


Fig. 37.11. Reflection from antireflection coating, curved surface, $T_y = 15^\circ$.

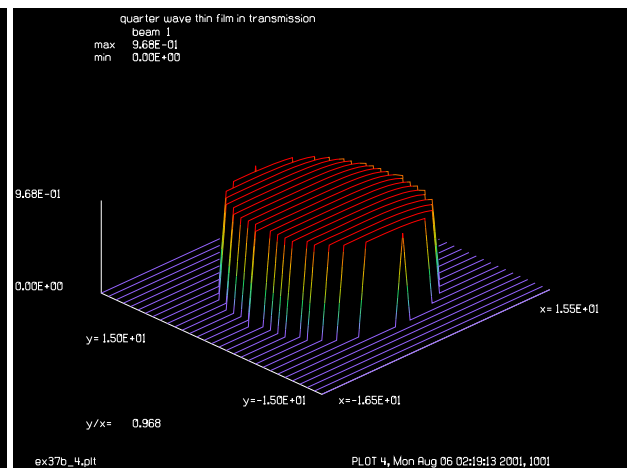


Fig. 37.12. Recombination of transmission and reflection from antireflection coating.

```
jones/set ar=0. bi=0 cr=1. ci=0 dr=0      # define jones operator
jones/multiply 1                          # apply jones operator
copy 1 3
copy 1 2
c
c Transmission of air-glass interface
c rotation about y-axis of 10 degrees
c
jsurf/fresnel/trans 1 1. 4. 12. ty=15
title air-glass transmission with curvature and tilt
plot/watch ex37b_1.plt
plot/l 1
c
c Transmission of air-glass interface
c rotation about y-axis of 10 degrees
c
```

Jump to: [Commands](#), [Theory](#)

```

jsurf/fresnel/refl 2 1. 4. 12. ty=15
title air-glass transmission with curvature and tilt
plot/watch ex37b_2.plt
plot/l 2
variab/set units 1 units
mult 1 units                # compensate beam 1 for x-magnification
add/inc/con 1 2
point/list/xy 1             # check point in center is 1.0000
pause
title sum of air-glass transmission and reflection
plot/watch ex37b_3.plt
plot/l 1
copy 3 1
copy 3 2
jsurf/single/trans 1 1. 2. 4. 1e-4/2/4 12. 12. ty=15 # air-glass surface
title quarter wave thin film in transmission
plot/watch ex37b_4.plt
plot/l 1
jsurf/single/refl 2 1. 2. 4. 1e-4/2/4 12. 12. ty=15. # air-glass surface
title quarter-wave thin film in reflection
plot/watch ex37b_5.plt
plot/l 2
variab/set units 1 units
mult 1 units                # compensate beam 1 for x-magnification
add/inc/con 1 2
point/list/xy 1             # check point in center is 1.0000
pause
title sum of quarter-wave transmission and reflection
plot/watch ex37b_6.plt
plot/l 1
end

```

Ex37c: Transmission and reflection coefficients for light incident at Brewster's angle

Input: `ex37c.inp`

```

c## ex37c
c
c Example 37c: Brewster's angle, calculate rs, rp, ts, tp
c
c
theta_b = 180./pi*atan(1.82) list          # Brewster's angle
echo/on
array/s 1 32 ip=1                          # 32 x 32 array with polarization
nbeam 1
wavelength/set 0 1
units/s 1 1
clear 1 1                                  # reinitialize array
clap/c/c 1 10
c
c Establish vertical polarization

```

Jump to: [Commands](#), [Theory](#)

```

c
jones/set ar=1. bi=0 cr=0. ci=0 dr=0      # define jones operator
jones/multiply 1                          # apply jones operator
energy/norm 1
jsurf/fresnel/trans 1 1. 1.82 ty=theta_b
variab/set/par Energy_tp 1 energy

units/s 1 1
clear 1 1                                # reinitialize array
clap/c/c 1 10
c
c  Establish vertical polarization
c
jones/set ar=1. bi=0 cr=0. ci=0 dr=0      # define jones operator
jones/multiply 1                          # apply jones operator
energy/norm 1
jsurf/fresnel/refl 1 1. 1.82 ty=theta_b
energy
variab/set/par Energy_rp 1 energy
Energy_tp=
Energy_rp=
Energy = Energy_tp + Energy_rp list
c Should be 1.0 exactly
pause

c -----
c Start a second beam
c
units/s 1 1
clear 1 1                                # reinitialize array
clap/c/c 1 10
c
c  Establish horizontal polarization
c
jones/set ar=0. bi=0 cr=1. ci=0 dr=0      # define jones operator
jones/multiply 1                          # apply jones operator
energy/norm 1
jsurf/fresnel/trans 1 1. 1.82 ty=theta_b
variab/set/par Energy_ts 1 energy

units/s 1 1
clear 1 1                                # reinitialize array
clap/c/c 1 10
c
c  Establish vertical polarization
c
jones/set ar=0. bi=0 cr=1. ci=0 dr=0      # define jones operator
jones/multiply 1                          # apply jones operator
energy/norm 1
jsurf/fresnel/refl 1 1. 1.82 ty=theta_b
energy
variab/set/par Energy_rs 1 energy

Energy_ts=

```

Jump to: [Commands](#), [Theory](#)

```

Energy_rs=
Energy = Energy_ts + Energy_rs list
c Should be 1.0 exactly
end

```

Ex37d: Polarization, Goos-Hanchen, Part 1

Input: ex37d.inp

```

c## ex37d
c
c Example 37d: Polarization, Goos-Hanchen, part 1
c
c This example illustrates the Goos-Hanchen shift calculated by applying
c the Fresnel reflection coefficients in the far-field. The phase variation
c with angle of incidence in TIR reflection creates a component of
c linear phase tilt in the far-field. This phase tilt creates a
c slight shift in the distribution at the TIR surfaced.
c
c It is only necessary to implement the Fresnel reflection coefficients
c correctly with a complex calculation of the cosine of the refracted
c angle. In this example, this is implemented by propagating 1E6 cm
c to the far-field, applying the ordinary Fresnel reflection
c routine jsurf/fresnel/refl, and propagating back to the original
c surface. The phase distribution is shown in the far-field after
c reflection showing both the linear phase term as well as second
c and third order phase aberrations effects that cause the reflected
c beam in the near-field to be wider.
c
echo/on
array/s 1 128 ip=1 # 32 x 32 array with polarization
units/s 1 .00002 # set units/s to 1
wavelength/set 1 .5 1.5 1
clear 1 1 # reinitialize array
Length = 1e6
Width=8e-5
gaus/c/c 1 1e6 Width
dist Length
jsurf/fresnel/refl 1 1.5 1 ty=60 # define glass-air surface
plot/w ex37d_1.plt
title beam in far-field
plot/l
plot/w ex37d_2.plt
title phase after reflection at glass-air interface
plot/l/w
nbeam 2 ip=1 data
wavelength/set 2 .5 1.5 1
units/beam 2 1
dist -Length
units/beam 2 1
gaus/c/c 2 1e6 Width
plot/w ex37d_3.plt
title comparison of shifted and unshifted beams

```

Jump to: [Commands](#), [Theory](#)

```

plot/x/i fi=1 la=2
fitgeo 1
variab/set Xcen fitxcen list

```

Ex37e: Polarization, Goos-Hanchen with jsurf/goos, Part 2.

Examples Ex37d and Ex37e illustrate the Goos-Hanchen effect. As explained in Chapt. 14, GLAD Theory Manual, to include Goos-Hanchen effects it is only necessary to apply the Fresnel reflection coefficients (properly implemented with complex math) in the far-field. In Ex37d, we explicitly add a propagation step of $1e6$ cm before the `jsurf` command, with back-propagation after reflection. For small diameter beams at the glass-air interface, the angular divergence is large and the linear change in phase with angle results in a translation of the beam upon back propagation. Figure 37.13 illustrates the incident beam, Fig. 37.14 illustrates the phase due to application of Fresnel reflection in the far-field, and Fig. 37.15 illustrates the slight shift due to Goos-Hanchen relative to the incident beam. Second, third, and higher order components of the variation of TIR phase with angle result in beam distortions. Ex37e uses the `jsurf/goos` command to accomplish both the far-field and Fresnel reflection calculations.

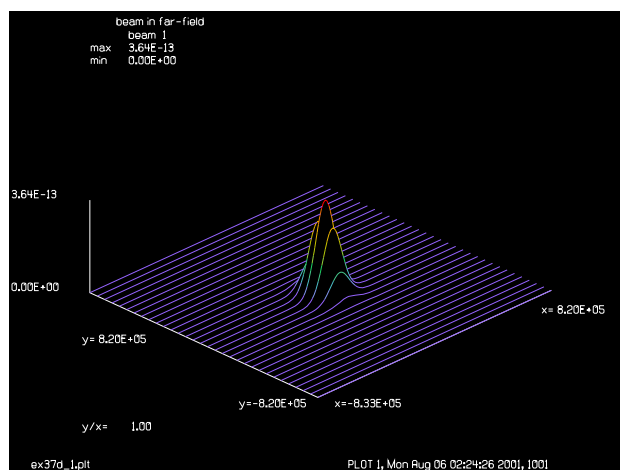


Fig. 37.13. Gaussian beam in far-field incident on glass-air interface at TIR angle.

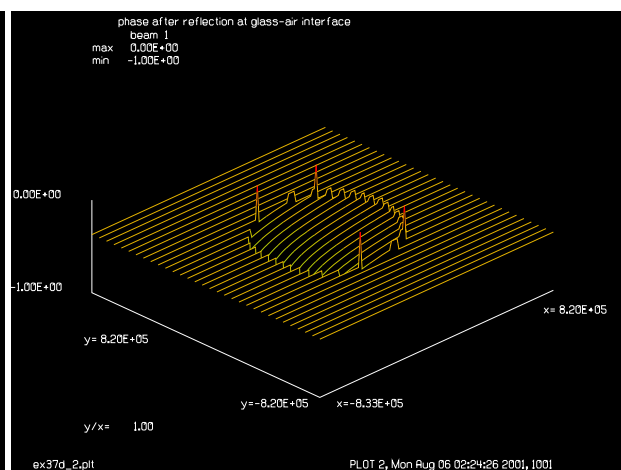


Fig. 37.14. Phase after Fresnel reflection calculations showing linear and higher order terms.

Input: `ex37e.inp`

```

c## ex37e
c
c Example 37e: Polarization, Goos-Hanchen, part 2
c
c This example illustrates the Goos-Hanchen shift calculated by applying
c the Fresnel reflection coefficients in the far-field. The phase variation
c with angle of incidence in TIR reflection creates a component of
c linear phase tilt in the far-field. This phase tilt creates a
c slight shift in the distribution at the TIR surfaced.
c
c It is only necessary to implement the Fresnel reflection coefficients
c correctly with a complex calculation of the cosine of the refracted
c angle. In this example, this is implemented by the command

```

Jump to: [Commands](#), [Theory](#)

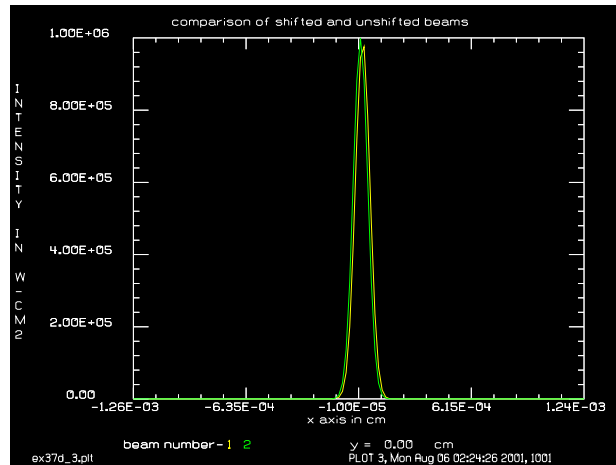


Fig. 37.15. After return to real glass-air interface. Original beam (green) and shifted beam (yellow). The shift is due to the Goos-Hanchen effect. The beam is also slightly wider due to high order terms.

```
c jsurf/gooshanchen that accomplishes the same functions as the
c the method of ex37d.
c
echo/on
array/s 1 128 ip=1                                # 32 x 32 array with polarization
units/s 1 .00002                                    # set units to 1
wavelength/set 1 .5 1.5 1
clear 1 1                                           # reinitialize array
Width=8e-5
gaus/c/c 1 1e6 Width
jsurf/goos 1 1.5 1 ty=60                            # define glass-air surface
nbeam 2 ip=1 data
wavelength/set 2 .5 1.5 1
units/beam 2 1
units/beam 2 1
gaus/c/c 2 1e6 Width
plot/w ex37e.plt
plot/x/i fi=1 la=2
fitgeo 1
variab/set Xcen fitxcen list
```

Ex37f: Birefringent wedge using 3d addressing of polarization sheets

If a birefringent prism is applied to the beam, it is possible to have a different prism effect on the x- and y-polarization states. In this case, the calculation follows the average angle. Over a sufficiently long propagation distance, the two polarization states will be seen to separate. It is convenient to use work with the polarization array as a 3d array, work with the two sections of the beam and then convert the array back to a polarized beam. To more clearly illustrate the effects, we choose a very extreme example with index differences that are likely to be impossible to achieve.

Input: ex37f.inp

```
c## ex37f
```

Jump to: [Commands](#), [Theory](#)

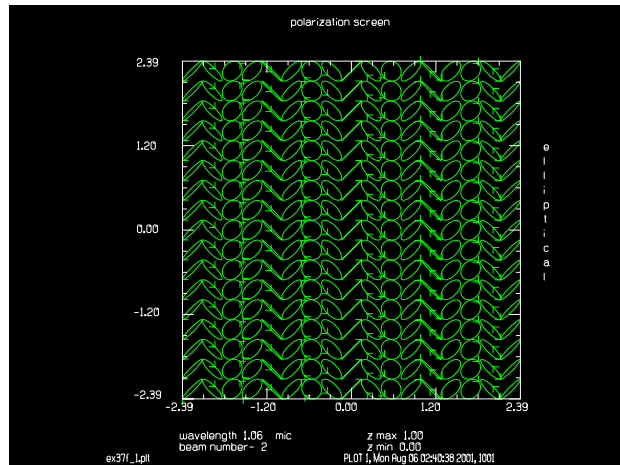


Fig. 37.16. Polarization phase screen for Ex37f showing different tilt in x and y.

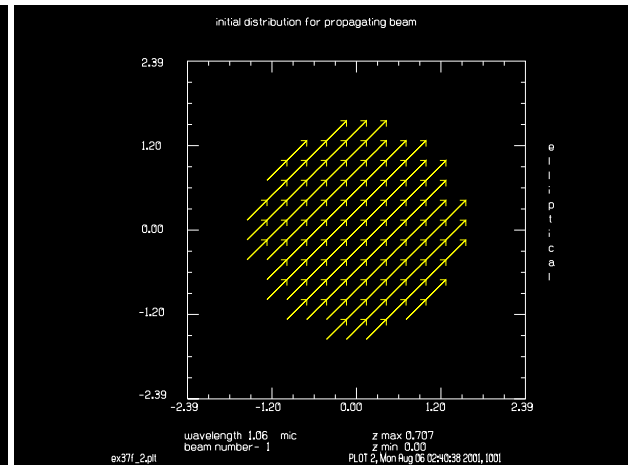


Fig. 37.17. Starting distribution for propagating beam.

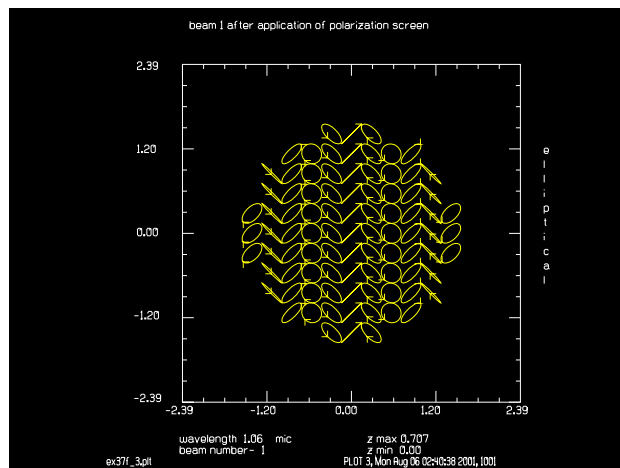


Fig. 37.18. Starting distribution after application of polarization screen.

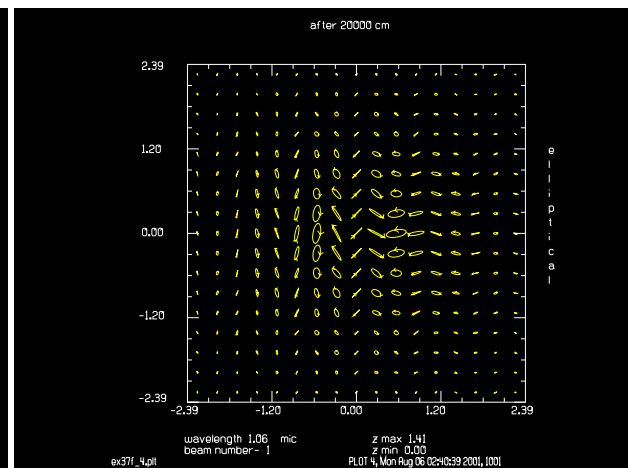


Fig. 37.19. Polarization plot after propagation of 20,000 cm shows separation of polarization states.

```
#
# Example of birefringent tilt
#
# If a birefringent prism is applied to the beam, it is possible to have a
# different
# prism effect on the x- and y-polarization states. In this case, the calculation
# follows the average angle. Over a sufficiently long propagation distance, the
# polarization states will be seen to separate.
#
# It is convenient to work with the polarization array as a 3d array, work with
# the two sections of the beam, and then convert the array back to a polarized be
#
# To more clearly illustrate the effects, we choose a very extreme example with
# index differences that are likely to be impossible to achieve
#
alias Name ex37f
```

Jump to: [Commands](#), [Theory](#)

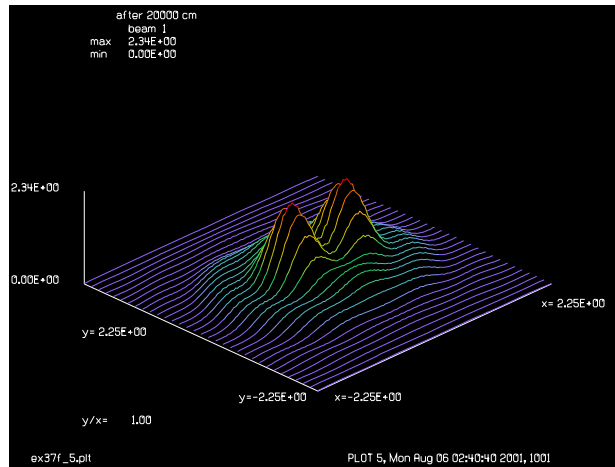


Fig. 37.20. Separation of polarization states after propagation of 20,000 cm.

```

echo/on
mem/set/b 16
variab/dec/int Nline
Nline = 512
array/s 1 Nline Nline 1
array/s/3d 2 Nline Nline 2 data      # make a 3D beam
Apt = 1.5                          # aperture radius
Waves = 4                          # waves of tilt
Azdeg = 90
AlphaDeg = 0.1                     # prism apex angle in degrees
AlphaRad = pi*AlphaDeg/180         # prism apex angle in radians
Lambda = 1.06e-4                    # wavelength
N0 = 2.0                            # average index
Nx = 2.5                            # x-index
Ny = 1.5                            # y-index

# propagate beam along direction caused by average tilt
# so average tilt is ignored

WavesX = AlphaRad*(Nx - N0)*Apt/Lambda
WavesY = AlphaRad*(Ny - N0)*Apt/Lambda

units/field 0 Apt*6
wavelength/set 0 Lambda*1e4
set/density 16 16
#
# a 3D array can be addressed in terms of 2D sections
# The usual 2D operations can be applied to any section.
# Specify the section to be addressed by changing the section number.
#
set/section 2 1                     # set the section number to 1
abr/tilt 2 WavesX rnorm=Apt azdeg=90
set/section 2 2                     # set the section number to 2
abr/tilt 2 WavesY rnorm=Apt azdeg=90

# convert array to polarized beam
array/convert/polarize 2           # convert N x N x 2 3D array to polarized beam

```

Jump to: [Commands](#), [Theory](#)


```

plot/w @Name_1.plt
title polarization screen
set/win/abs -1.5*Apt 1.5*Apt -1.5*Apt 1.5*Apt
plot/e 2 # plot elliptical polarization
clap 1 Apt
jones/set ar=1./sqrt(2.) br=0. cr=1./sqrt(2.) dr=0.
jones/mult 1
plot/w @Name_2.plt
title initial distribution for propagating beam
plot/e 1
mult/beam 1 2
plot/w @Name_3.plt
title beam 1 after application of polarization screen
plot/e 1 # plot elliptical polarization
prop 20000
plot/w @Name_4.plt
title after 20000 cm
plot/e 1
plot/w @Name_5.plt
plot/l 1 xrad=Apt*1.5
Prism with Brewster's Angle windows and multiple side bounces

```

Ex37g: Calculate polarization properties through a Dove prism

Input: `ex37g.inp`

```

c## ex37g
c
c Calculate polarization properties through a Dove prism with
c entrance and exit faces set to Brewster's Angle and three
c TIR bounces from the walls

variable/dec/int P_Polarization
P_Polarization = 1
Index = 1.818
theta_b = 180/pi*atan(Index) # Brewster angle in radians
RefractedAngle = 180/pi*asin(sin(pi*theta_b/180.)/Index) list

array/s 1 64 64 ipol=1 # both polarization states
units/field 1 1.5
wavelength/set 0 1.064
gaussian/cir/con 1 2. 1. 30 # super-gaussian

if [P_Polarization] then
  jones/set ar=0 br=0 cr=1 dr=0 # set Jones operator
  jones/mult 1
endif
variable/set EnergyStart 1 energy list

lensgroup/define front/overwrite
index Index # define index and glass before use
surface_lensgroup radius=1e20 glass

```

Jump to: [Commands](#), [Theory](#)

```

    image focus
lensgroup/end

lensgroup/define back/overwrite
    surface_lensgroup radius=1e20 air
    image focus
lensgroup/end

prop 10.
vertex/locate/absolute x=0. y=0. z=10.
vertex/rotate/set rx=theta_b          # corrected angle
lensgroup/run front          # start the slab with a brewster angle surface

jsurf/fresnel/tran 1 n1=1 n2=Index tx=theta_b          # tip angle

# z-position does not do anything for 90 deg mirror
# but does no harm

vertex/locate/absolute x=0. y=-.29 z=10.54169
vertex/rotate/set rx=90.
variable/set Xrot 1 xrot
X_IncidenceAngle = 90-Xrot list          # incidence angle for mirror
mirror/global/flat          # 1st bounce
variable/set Xrot 1 xrot list
jsurf/fresnel/refl 1 n1=Index n2=1 tx=X_IncidenceAngle

vertex/locate/absolute x=0. y=.29 z=0
vertex/rotate/set rx=-90.
mirror/global/flat          # 2nd bounce
jsurf/fresnel/refl 1 n1=Index n2=1 tx=X_IncidenceAngle

vertex/locate/absolute x=0. y=-.29 z=0
vertex/rotate/set rx=90.
mirror/global/flat          # 3rd bounce
jsurf/fresnel/refl 1 n1=Index n2=1 tx=X_IncidenceAngle

vertex/locate/absolute x=0. y=0. z=13.25139          # out of the slab
vertex/rotate/set rx=-theta_b
lensgroup/run back          # end the slab with a brewster angle surface

jsurf/fresnel/trans 1 n1=Index n2=1 tx=RefractedAngle
status
global/list
variable/set EnergyFinish 1 energy
C energy loss
LossPercent = 100*(EnergyStart - EnergyFinish)/EnergyStart list

```

Ex38: Shearing Interferometer

This example illustrates modeling of a shearing interferometer. An amplitude grating is moved across an image and the pupil distribution and total energy is observed. A field lens is used to reimage the pupil. In addition to the total energy, we observe the energy in a small aperture at the center of the aperture and at the edge of the aperture. In the first case, the pupil is aberration-free. Different areas of the pupil are modulated by different amounts, but always at the same temporal frequency. A summary plot at the end shows the total energy and the local energy modulation over one cycle of movement of the grating.

Table. 38.1. Parameters.

Parameter	Value
aperture radius	20 cm
lens focal length	100 cm
wavelength	10.6 micron

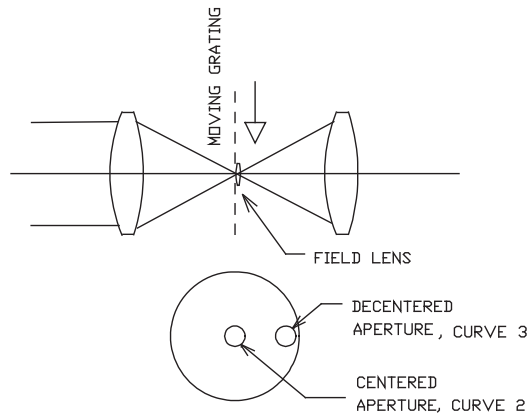


Fig. 38.1. Schematic of shearing interferometer. An image of the lens is formed and an amplitude grating is moved past the image. A field lens is used to reimage the original pupil to avoid near-field diffraction affects. The reimaged pupil shows shifted pupils for the +1 and -1 orders in addition to the 0 order pupil. As the grating is moved through one period, the pupil is modulated. Different areas of the pupil are modulated by different amounts depending on the order overlap but all have the same frequency. Two small apertures are used to sample the energy, one at the center and the other at the edge of the aperture — in addition to measuring the total energy.

Input: ex38.inp

```

c## ex38
c
c Example 38: Shearing interferometer
c
c This example illustrates the modeling of a shearing interferometer by
c using a moving amplitude grating. A pupil of 40 cm diameter
c is brought to a focus with a 100 cm lens. An amplitude grating is moved
c past the image, which causes a modulation in the reimaged pupil.
c We calculate the total energy and the energy in a 2 cm radius circle
c at the center of the aperture and in a 2 cm radius circle at the edge
c aperture. The total energy and energy in isolated
c areas is, of course, always in phase. The amplitude grating is moved
c in 45 degree phase increments over one full cycle.

```

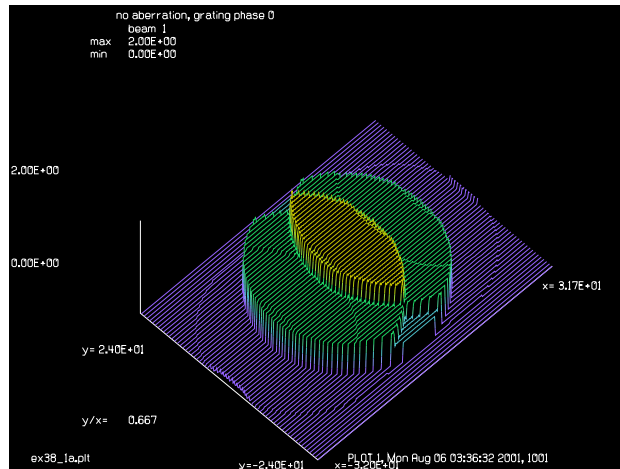


Fig. 38.2a. Isometric, 0 deg.

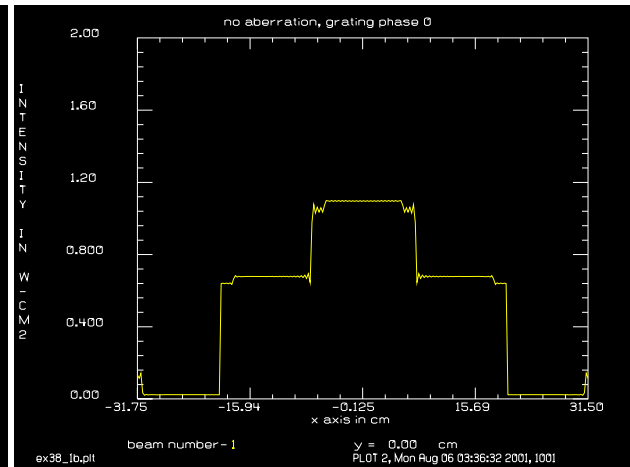


Fig. 38.2b. Profile, 0 deg

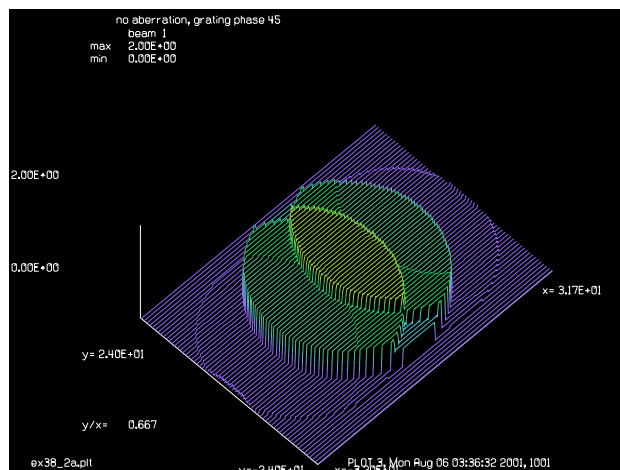


Fig. 38.3a. Isometric, 45 deg.

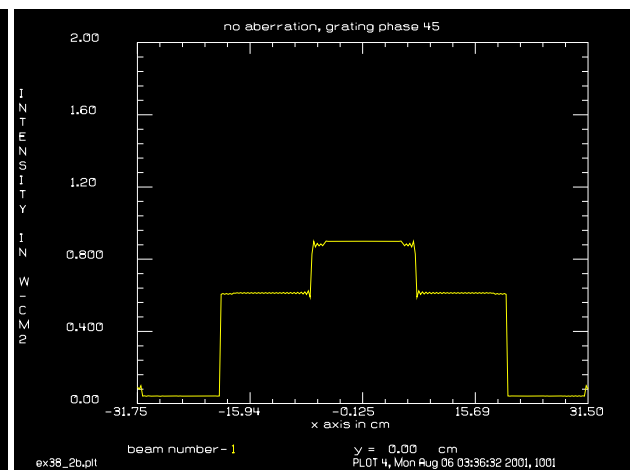


Fig. 38.3b. Profile, 45 deg.

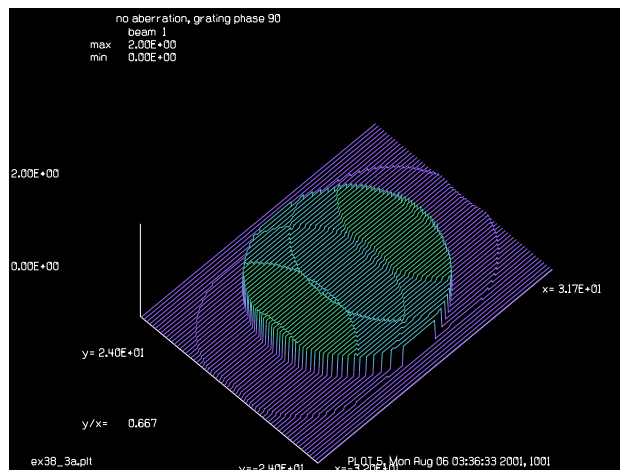


Fig. 38.4a. Isometric, 90 deg.

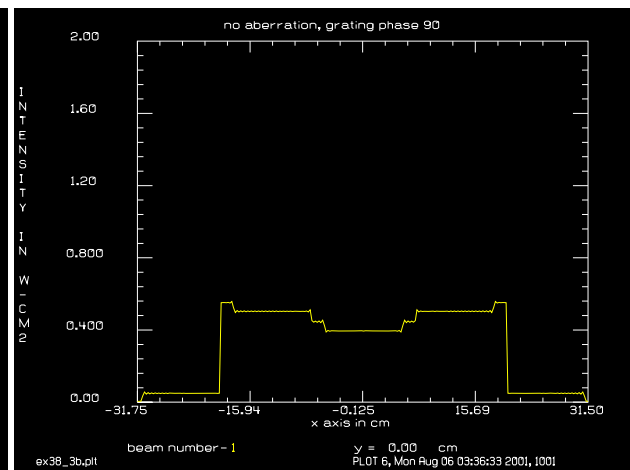


Fig. 38.4b. Profile, 90 deg.

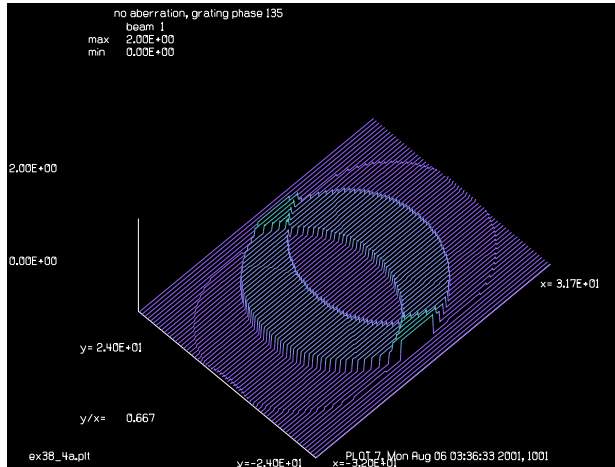


Fig. 38.5a. Isometric, 135 deg.

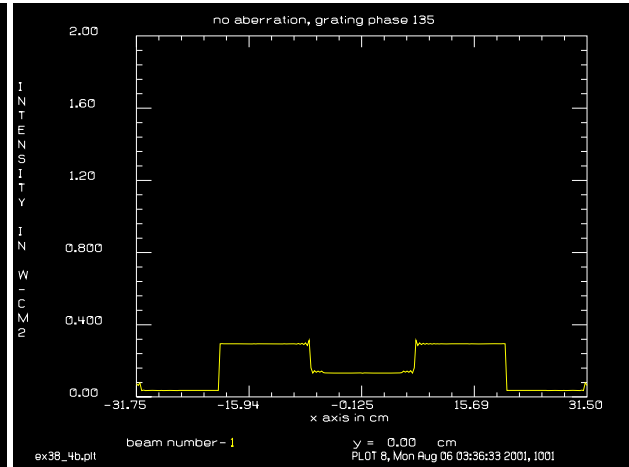


Fig. 38.5b. Profile, 135 deg.

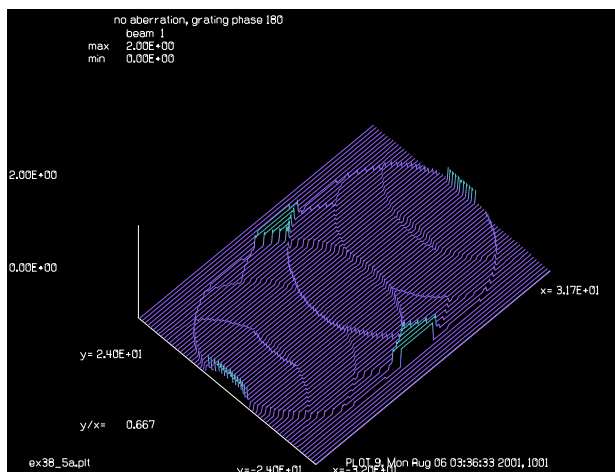


Fig. 38.6a. Isometric, 180 deg.

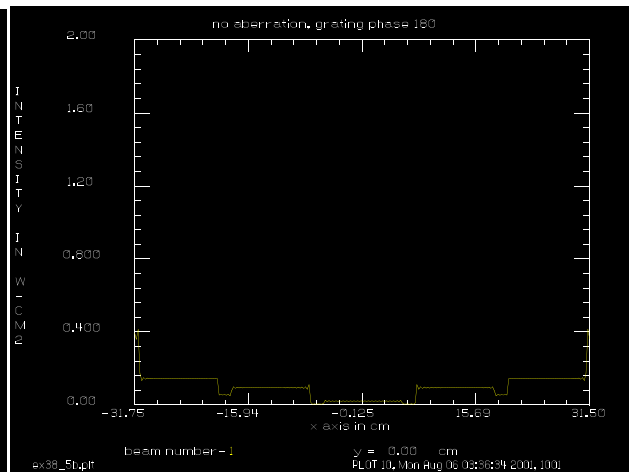


Fig. 38.6b. Profile, 180 deg.

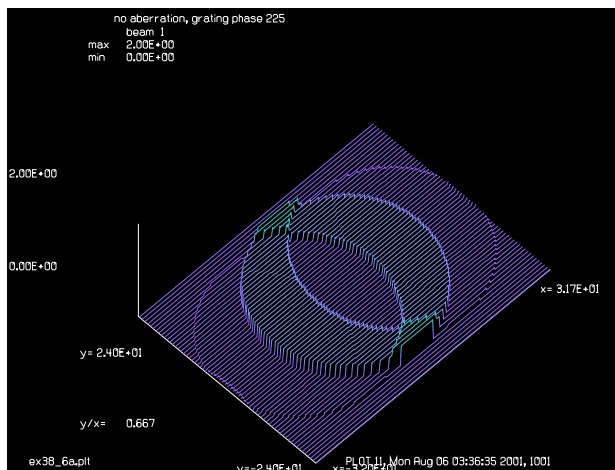


Fig. 38.7a. Isometric, 225 deg.

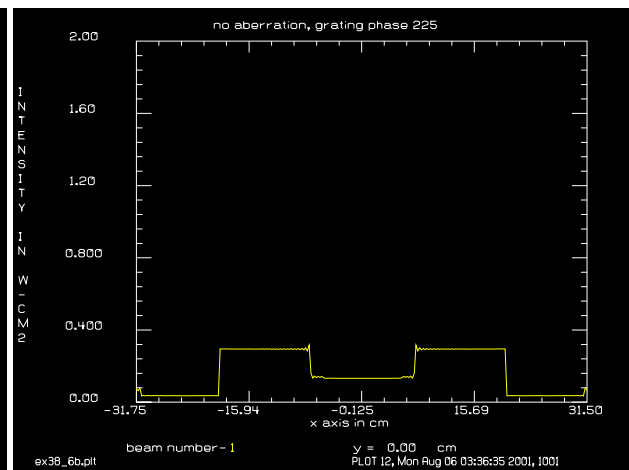


Fig. 38.7b. Profile, 225 deg.

Jump to: [Commands](#), [Theory](#)

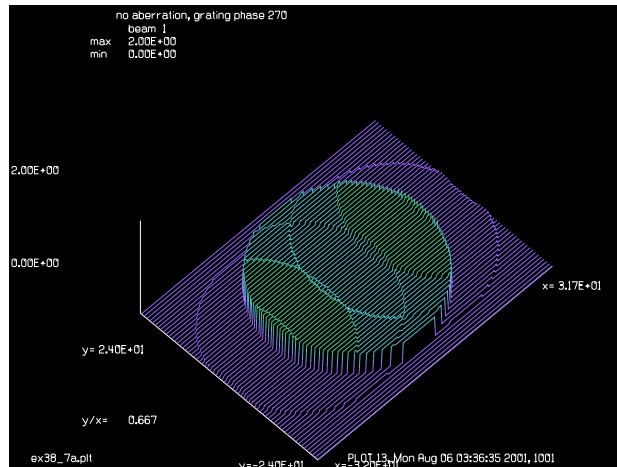


Fig. 38.8a. Isometric, 270 deg.

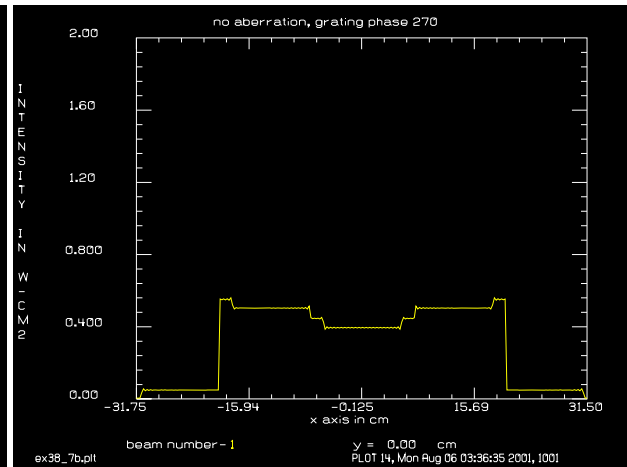


Fig. 38.8b. Profile, 270 deg.

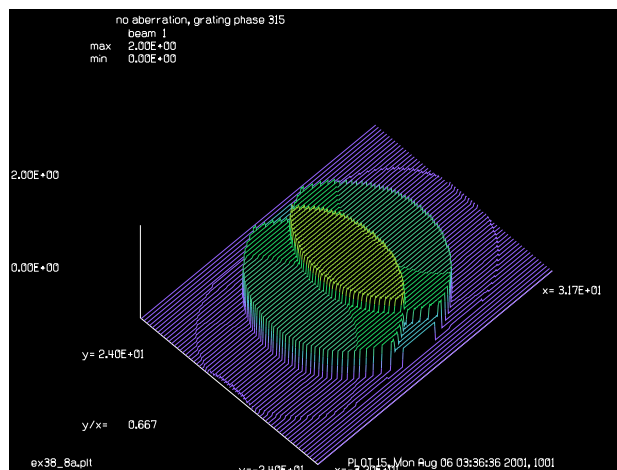


Fig. 38.9a. Isometric, 315 deg.

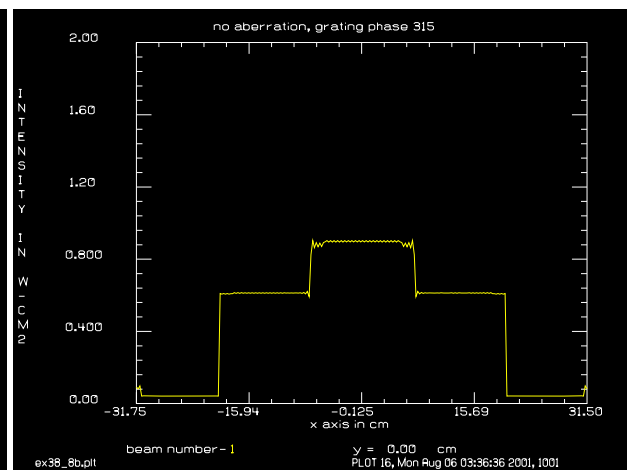


Fig. 38.9b. Profile, 315 deg.

```

c
variab/dec/int pass phase
nbeam 3                                # Set up 3 beams, only 1 is active
array/s 0 256                          # Use 256 X 256 array
pass = 0                                # Initialize pass counter
phase = -45                             # Initialize grating phase
units/s 0 .25                           # Set units
clap/c/c 1 20                           # 40 cm diameter aperture
c -- abr/focus 1 6.5                    # (insert this command to see aberration)
energy
lens 1 100                              # lens of 100 cm focal length
dist 100 1                              # propagate to focus
copy 1 2                                # save distribution in Beam 2
macro/define grat/o                     # define macro
  pass = pass + 1                        # increment pass counter
  phase = phase + 45                     # increment grating phase
  copy 2 1                               # restore image distribution
  grat 1 8.281e-3 phi=phase              # grating of period = .0082813 cm

```

Jump to: [Commands](#), [Theory](#)

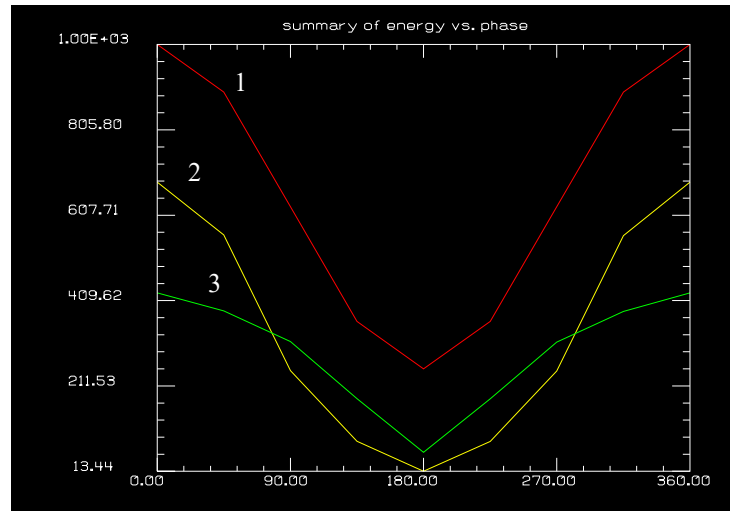


Fig. 38.10. Energy for the total aperture versus grating phase. Zero phase puts the peak of a cosine transmission pattern at the center of the image distribution. The energy in a small centered aperture (Plot 2) and one at the edge of the aperture (Plot 3) are also plotted. The energy from the smaller apertures is multiplied by 50 to make plotting easier. It can be seen that all energy curves are of the same frequency and phase.

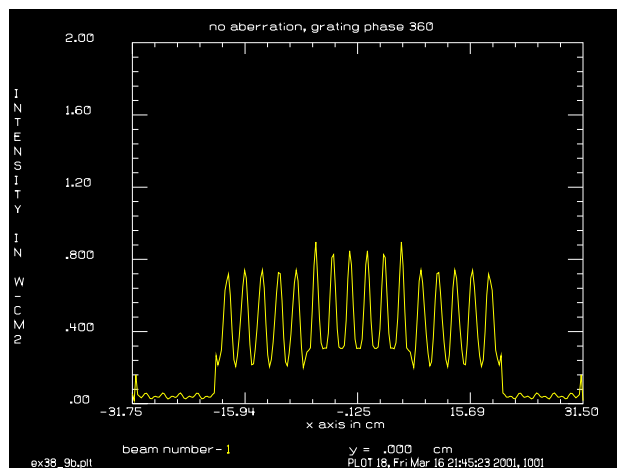


Fig. 38.11. Profile with 6.5 waves of focus aberration. The contrast of temporal modulation is very low and the pattern hardly changes through a grating period.

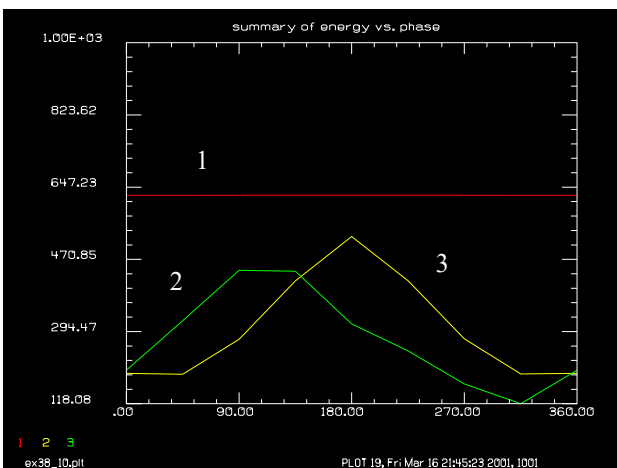


Fig. 38.11a. Curve of energy versus grating phase for 4 waves of focus error in the pupil, similar to Fig. 38.10. Note that all curves have the same period but show different phases. This is the general result for aberration in the pupil.

```

variab/set energy1 1 energy # Set energy1 to energy in Beam 1
lens 1 50 # field lens to reimage pupil
dist 100 1 # propagate to pupil image
plot/watch ex38_@passa.plt # set plot file name
plot/l 1 xr=36 yr=24 ns=64 max=2 thet=40
plot/watch ex38_@passb.plt
plot/x/i 1 fmax=2
copy 1 3 # make a copy of the pupil in Beam 3
clap/c/c 1 .31 # 2 cm radius aperture in pupil center
clap/c/c 3 .31 xdec=8 # 2 cm radius aperture, edge of pupil
variab/set energy2 1 energy

```

```
variab/set energy3 3 energy
energy2 = energy2*2000      # linear scaling for PLOT/UDATA
energy3 = energy3*2000
udata/set pass phase energy1 energy2 energy3
macro/end
title/format f 3 0
title no aberration, grating phase @phase
macro/run grat/9
title summary of energy vs. phase
plot/watch ex38_10.plt
plot/udata 1 3
write/disk ex38.out/o
udata/list
end
```


Ex39: Gaussian phase factor in propagation, Gouy shift

This example illustrates the phase shift of a spherical gaussian beam. The representation of a gaussian beam at the waist is,

$$A(r, 0) = e^{-\frac{r^2}{\omega_0^2}} \quad (39.1)$$

The definition of the gaussian spherical wave adds a quadratic phase factor:

$$A(r, 0) = \frac{\sqrt{2}}{\pi} \frac{1}{\omega} e^{-\frac{r^2}{\omega^2}} e^{-jk \frac{r^2}{R}} \quad (39.2)$$

where k is the wave number and R is the radius of the phase front. At the waist $R = \infty$, and the phase factor disappears. The gaussian propagation equations are

Table. 39.1. Principle terms.

radius of beam	$\omega(z) = \omega_0 \sqrt{1 + \frac{z^2}{z_R^2}}$
phase radius	$R(z) = z + \frac{z_R^2}{z}$

To the accuracy of scalar Fresnel diffraction, the gaussian beam propagates according to the equation.

$$A(r, 0) = \frac{\sqrt{2}}{\pi} \frac{1}{\omega(z)} e^{-j(kz - \theta(z))} e^{-\frac{r^2}{\omega^2}} e^{-jk \frac{r^2}{R}} \quad (39.3)$$

Table. 39.2. Factors.

piston term	$e^{-j(kz - \theta(z))}$
local transverse distribution	$e^{-\frac{r^2}{\omega^2}}$
quadratic phase factor	$e^{-jk \frac{r^2}{R}}$

The amplitude drops inversely to the increase in beam radius. There is a piston term which contains a phase term specific to gaussian beams. The remainder of the function is a spherical gaussian wave.

Let us consider each of the terms above, as shown in Table 39.2. At the Rayleigh distance, the amplitude drops to $\sqrt{2}/2$ and the phase correction is -45° , giving a value of (.5,-.5). At a distance of 1E9, the phase correction is -90° . Negative propagation from the waist results in positive phase correction factors.

The amplitude at a distance of 1E9 is

$$\frac{1}{\sqrt{1 + \frac{z^2}{z_R^2}}} = \frac{1}{\sqrt{1 + \left(\frac{1 \times 10^9}{2.964 \times 10^3}\right)^2}} = 2.964 \times 10^{-6} \quad (39.4)$$

Input: ex39.inp

```
c## ex39
c
c Example 39: Gaussian Phase Correction Factor
c
macro/def angle/o
  point/list/ij 1 33 33
  write/off
  variab/set x point/sr
  variab/set y point/si
  angle = 180*atan2(y,x)/pi
  write/on
macro/end
gaussian/c/res 1 1 1          # Gaussian beam with waist = 1
zbound                        # Rayleigh distance, Zr = 2.964E3
dist 2.963e3                  # Move to Rayleigh distance
echo/on
macro angle
c peak amplitude (.5,-.5), phase = -45 deg
angle=
pause
dist -2.963e3                  # Move back to waist
macro angle
c peak amplitude (1.,0.), phase = 0 deg
angle=
pause
dist 1e9                       # Move to 1e9
macro angle
c peak amplitude (.0,-2.964E-6), phase = -90 deg
angle=
pause
dist -1e9                      # Move to waist
macro angle
c peak amplitude (1.,0.), phase = 0 deg
angle=
pause
dist -2.963e3                  # To negative Rayleigh distance
macro angle
c peak amplitude (.5,.5), phase = 45 deg
angle=
pause
```

Jump to: [Commands](#), [Theory](#)

```
dist 2.963e3                # Move to waist
macro angle
c peak amplitude (1.,0.), phase = 0 deg
angle=
pause
dist -1e9                   # Move to -1e9 from waist
macro angle
c peak amplitude (0.,2.964E-6), phase = 90 deg
angle=
pause
end
```


Ex40: Phase conjugation, limited interaction length

This example considers the case of a 32 cm diameter beam passing through a turbulence layer. The beam will be phase conjugate by a nonlinear optic conjugator having a finite length. The finite length of the conjugator results in incomplete conjugation of the higher spatial frequencies. The bandwidth of the conjugation process is

$$\frac{\sin(\Delta kz)}{\Delta kz} \quad (40.1)$$

where Δk defines the angle of interaction. The expression above can be well approximated by an Airy pattern. The first zero of the Airy pattern is matched to the first zero of the function.

Figure 40.1 shows a probe beam and a signal beam passing through aberration. We form the conjugate of one of the beams but limit the angular frequencies present in the conjugated beam.

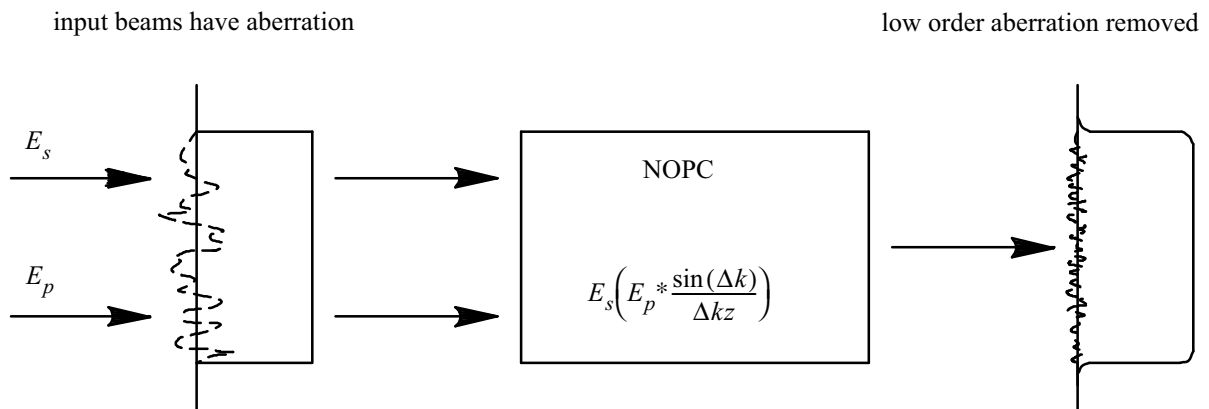


Fig. 40.1. Sketch of beam conjugation configuration. A probe beam E_p and a signal beam E_s pass through common aberration. The nonlinear optic phase conjugator (NOPC) calculated the product $E_s E_p^*$ with damping of the high spatial frequencies. The resulting beam is not perfectly corrected in the high spatial frequencies.

The wavelength is 0.5 micron. The aberration is modeled as 0.16 waves RMS with an autocorrelation radius of 2 cm, as shown in Fig. 40.2. The Strehl ratio of the aberration is 0.35. The far-field of the probe beam is shown in Fig. 40.3. The filter function to damp the high frequencies of the conjugate is shown in Fig. 40.4. The corrected beam is shown in Fig. 40.5, where one can see mostly high spatial frequency aberration remaining.

Input: ex40.inp

```
c## ex40
c
c Example 40: Nonlinear Optical Phase Conjugation
c
c Phase conjugation may be incomplete for nonlinear optics if the interaction
c region is too short. The higher spatial frequencies are reduced in the
c conjugate wave by the equation
```

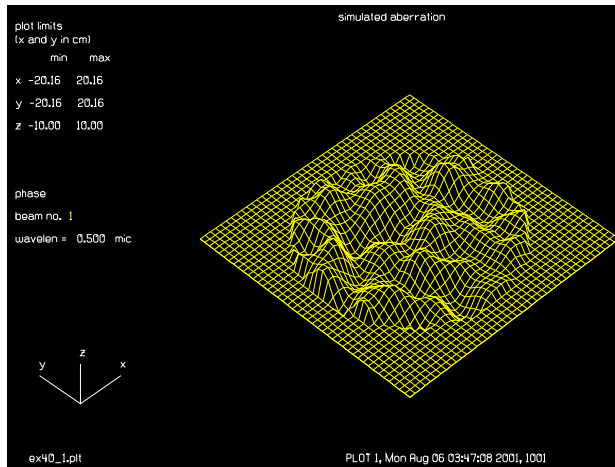


Fig. 40.2. Initial aberration.

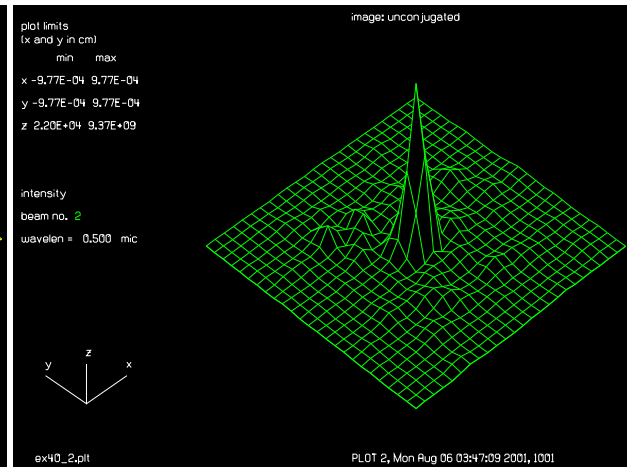


Fig. 40.3. Image with initial aberration.

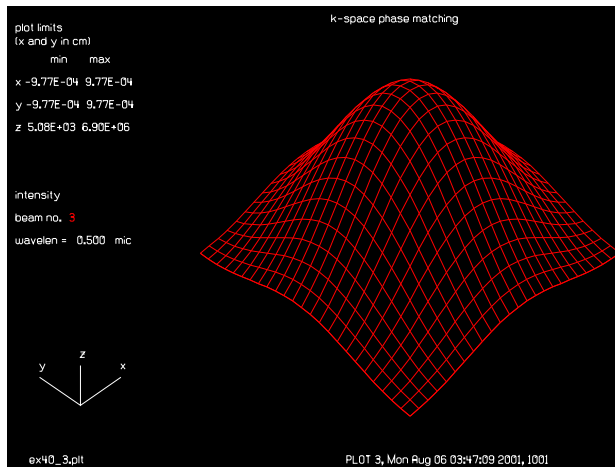


Fig. 40.4. Band limiting function for conjugation.

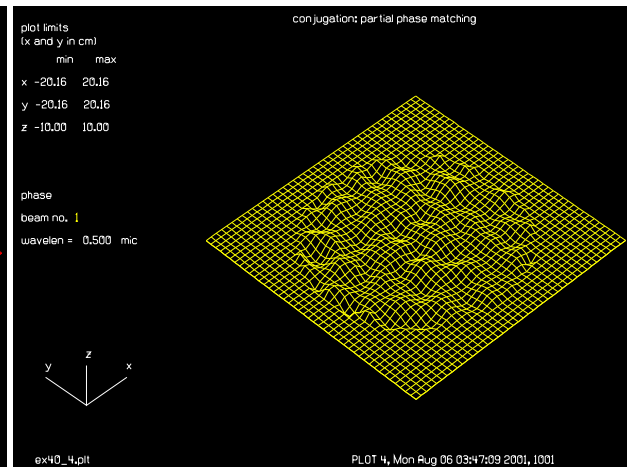


Fig. 40.5. Aberration residuals after conjugation.

```

c
c  sin(dkz) / dkz
c
c  where dk is the transverse radius in k-vector space and z is the length in
c  the nonlinear medium.  For convenience we will approximate the function by
c
c  J1(dkz) / dkz
c
c  where J1 is the first order Bessel function.  The above equation is the
c  Airy pattern.
c
c  The phase-matching angle of the first zero of the sin(x)/x function is
c  assumed to be 3.05E-5 radians.  This equivalent to the angle of the first
c  zero of an Airy pattern of 2 cm radius at lambda = .5 micron.
c
echo/on                                # turn on echo
array/s 1 128                          # set Beam 1 to 128 x 128
nbeam 3                                # establish 2 more beams of the same size

```

Jump to: [Commands](#), [Theory](#)

```

wavelength/set 0 .5          # set wavelength to .5 micron
units/s 0 .48                # set units to .48 cm
clap/c/c 1 16                # define 32 cm diameter aperture
phase/ran 1 .16 2            # random aberration, RMS = .16 waves,
                              # autocorrelation radius = 2 cm

set/density 32
set/window/abs -20 20 -20 20
title simulated aberration
plot/watch ex40_1.plt
plot/i/ph f=1 l=1 max=10 min=-10
strehl 1
copy 1 2                      # copy Beam 1 to Beam 2
status/p
conjug 2                      # form the conjugate of Beam 2
mirror/flat 2                 # add a mirror to change the direction
status/p
lens 2 100                    # 100 cm focal length lens for Beam 2
dist 100 2                    # propagate to focus of Beam 2
set/window/abs -.001 .001 -.001 .001
title image: unconjugated
plot/watch ex40_2.plt
plot/i f=2 l=2
clap/c/c 3 2                  # simulate sin(x)/x function with J1(x)/x
lens 3 100
dist 100 3                    # form image of Beam 3
title k-space phase matching
plot/watch ex40_3.plt
plot/i f=3 l=3
mult/beam 2 3                 # multiply Beam 2 by Beam 3
                              # to simulate damping by sin(x)/x
dist -100 2                   # back up to pupil
mult/beam 1 2                 # multiply Beam 1 by frequency-filtered
                              # conjugated
                              # apply aperture
clap/c/c 1 17
set/window/abs -20 20 -20 20
title conjugation: partial phase matching
plot/watch ex40_4.plt
plot/i/ph f=1 l=1 max=10 min=-10
strehl
end

```


Ex41: Effect of spatial filter on polarization

This example illustrates the effect of a spatial filter on polarization variation in the pupil. An arbitrary distribution of polarization is established in the pupil. The polarization is linear along the x- and y-axes and becomes circular along the 45 degree diagonals. The polarization variation reduces the Strehl ratio to about 0.5. The spatial filter smooths out pupil variations including polarization. Generalized polarization filters can be used. The beam is separated into parts which are parallel and perpendicular to the input distribution with the command `jones/orthog`.

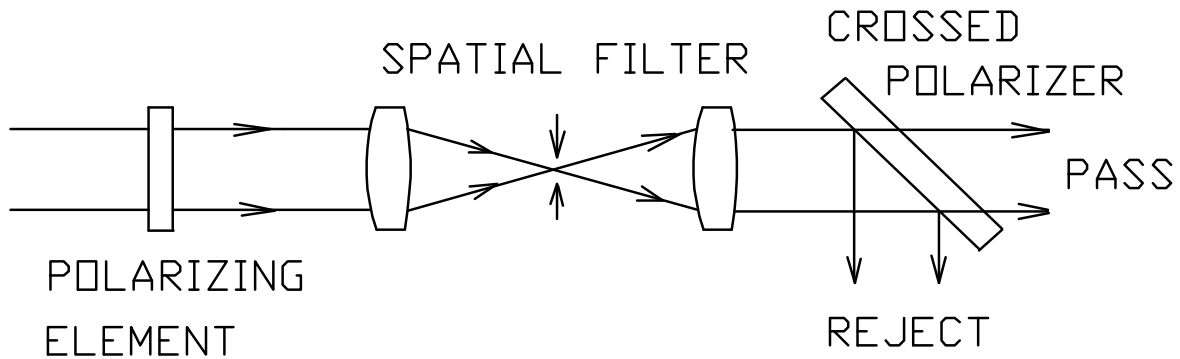


Fig. 41.1. Configuration for the example. A polarizing element is inserted in the beam. The spatial filter smooths out the polarization in the pupil. A polarizing filter which exactly matches the input is used to select the part of the output common to the input and the part orthogonal to the input.

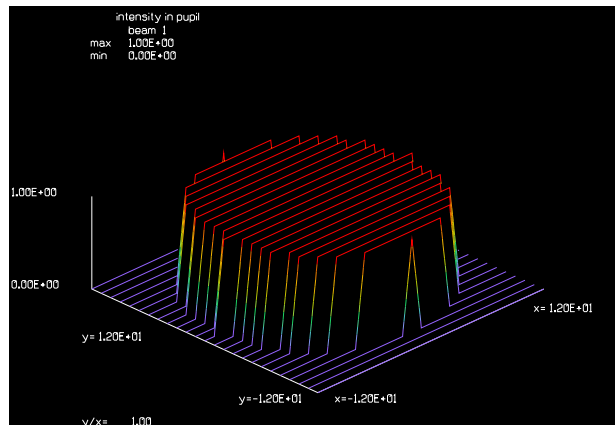


Fig. 41.2. Initial irradiance distribution.

Input: ex41.inp

```
c## ex41!556554515378499
c
c Example 41: Effects of Spatial Filter on Polarization
c
c This example illustrates the effect of a spatial filter on polarization
c in the pupil. The example illustrates the use of generalized crossed
c polarizers to select polarization states which are either of identical
```

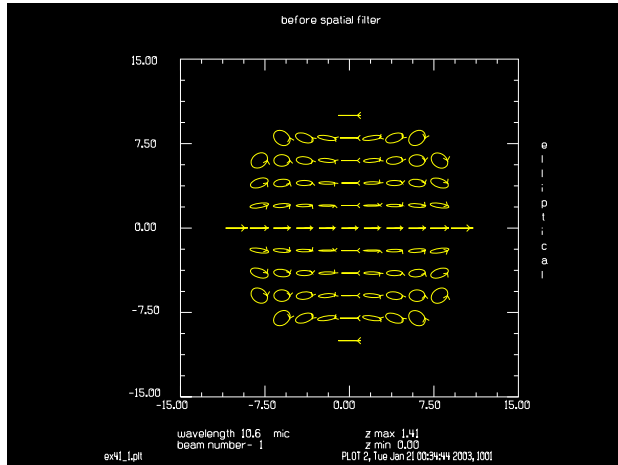


Fig. 41.3. Initial pupil polarization.

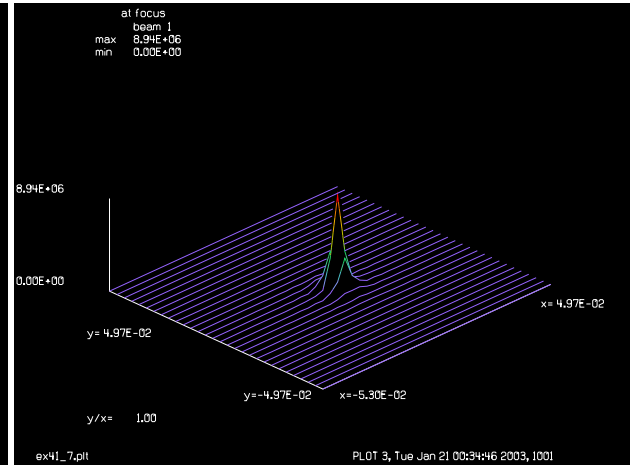


Fig. 41.4. Image irradiance.

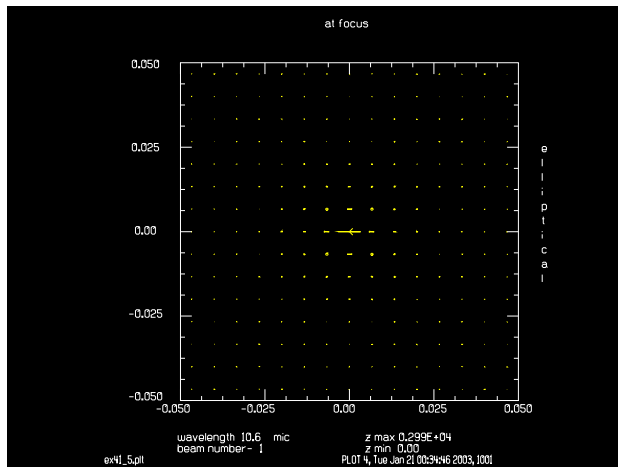


Fig. 41.5. Polarization in the far-field.

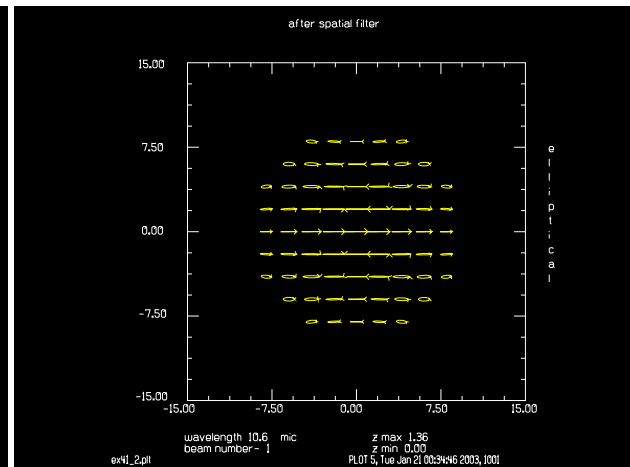


Fig. 41.6. Polarization after spatial filter.

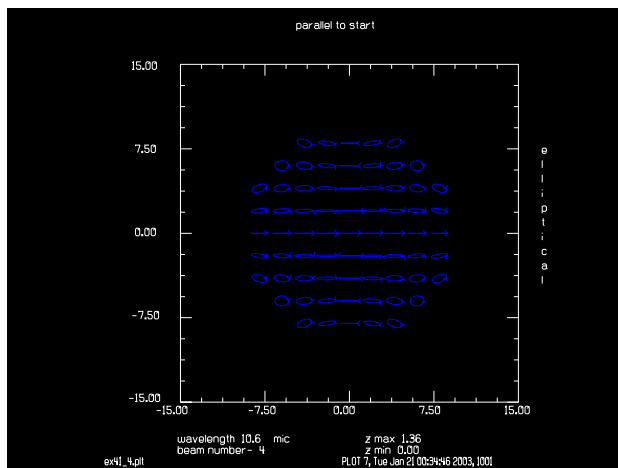


Fig. 41.7. Part of output parallel to input polarization.

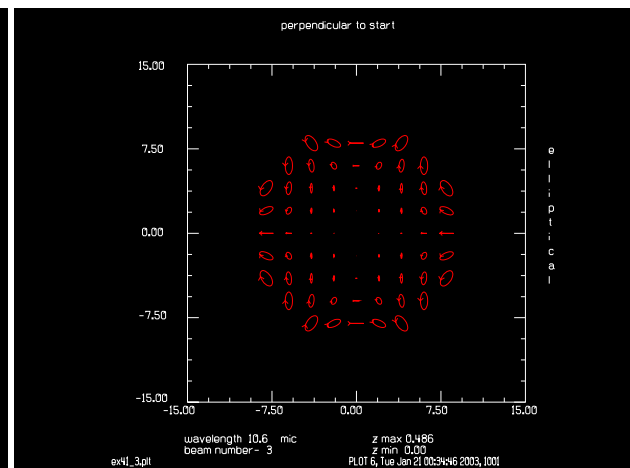


Fig. 41.8. Part of output orthogonal to input polarization.

```

c  polarization or of orthogonal polarization to the original beam.
c
echo/on
set/density 16 16
array/s 1 32 32 1          # establish 32 x 32 array
nbeam 3
jaberr/radius 10            # set radius of aberration surface
jaberr/second/jones ar=0 ai=1 dr=0 di=-1 # define second 2nd order jones coef.
clap/c/c 1 10              # clear aperture radius = 10
title intensity in pupil
plot/watch ex41_6.plt
plot/1 1 ns=32 xrad=12     # plot intensity in pupil
energy
jaberr/surf
energy
strehl
copy/con 1 2               # save beam for later use
title before spatial filter
plot/watch ex41_1.plt
plot/ell 1                 # elliptical polarization plot
lens 1 100                 # start spatial filter
dist 100 1
title at focus
plot/watch ex41_7.plt
plot/1 ns=32 1             # plot intensity at image
plot/watch ex41_5.plt
plot/ell 1                 # plot polarization at image
clap/c/c 1 .007            # aperture for spatial filter
dist -100 1
clap/c/c 1 10              # final aperture in spatial filter
title after spatial filter
plot/watch ex41_2.plt
plot/ell 1                 # polarization after spatial filter
copy/con 1 3               # save Beam 1 in Beam 3
nbeam 4
copy/con 2 4               # Copy Beam 2 to Beam 4
                           # only the inner part of the beam is
                           # copied to the smaller array
                           # splits B3 into parts parallel and
                           # perpendicular to B4

jones/orthog 3 4

title perpendicular to start
plot/watch ex41_3.plt
plot/ell 3
title parallel to start
plot/watch ex41_4.plt
plot/ell 4
end

```


Ex42: Waveguide grating coupler, mode matched input

This is the first of several examples on waveguide couplers. Fig. 42.1 shows a schematic of a typical waveguide grating coupler.

This example illustrates outcoupling of waveguide grating and subsequent incoupling with the identical mode. The grating is of constant coupling constant in a circular grating region. The input guided mode is of unit amplitude. The guided mode is mostly diffracted into the radiated mode in the region of the grating. The undiffracted guided mode residuals are shown in Fig. 42.2. The radiated mode is in the form of a “sugar scoop” pattern because of the relative strong coupling constant of 1 cm⁻¹ and the circular grating region (Fig. 42.3).

The incoupling is accomplished by forming there inner product of the of the conjugate of the normalized radiation mode with the incident radiation mode. In this example, the incident radiation mode is exactly mode matched. However, the transmitted guided mode (residual from the radiation mode formation) is not included so the initial unit amplitude guided mode is not recreated. In particular, the intensity rolls off at the edge of the aperture, as shown in Fig. 42.4. The transmitted radiation is shown in Fig. 42.5. Even though the radiation modes are exactly matched, the absorption of radiation is not perfect because there is no incident back coupled guided mode. The transmitted radiation is strongest on the sides of the circular grating patch where most of the radiation went past the grating in the outcoupled beam without being diffracted.

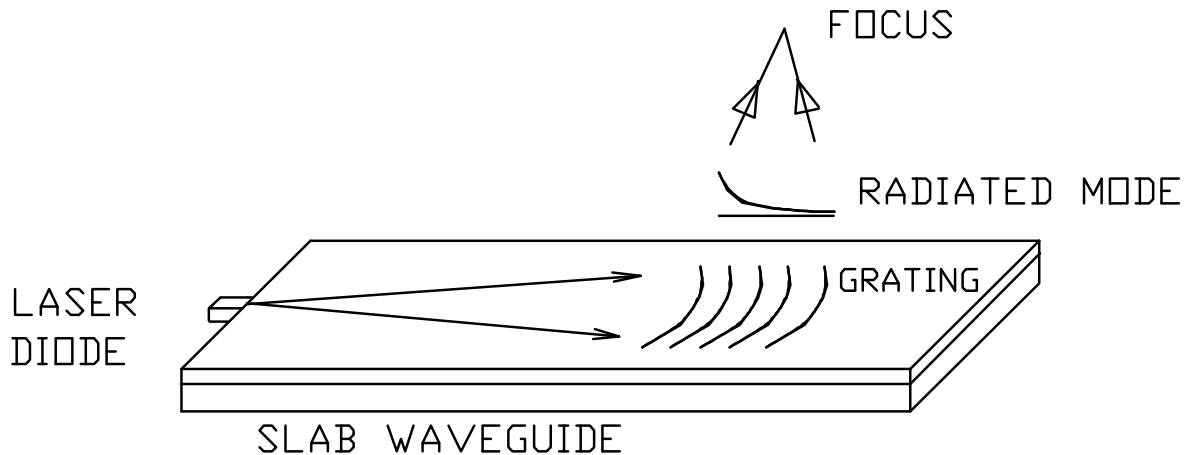


Fig. 42.1. Schematic of grating coupler as used in optical data storage. Light from a laser diode is injected into a slab waveguide. The light propagates as a guided mode in the vertical direction—trapped in a boundary layer of higher index. A grating of variable index or surface relief is put on the grating. The grating lines are curved to form a diffracted beam which converges to the focus. The guided mode decays as it passes under the grating, as the light is scattered out.

Input: ex42.inp

```
c## ex42
c
c Example 42: Waveguide Grating Coupler, Mode-Matched Input
c
c This example illustrates outcoupling of waveguide grating and
```

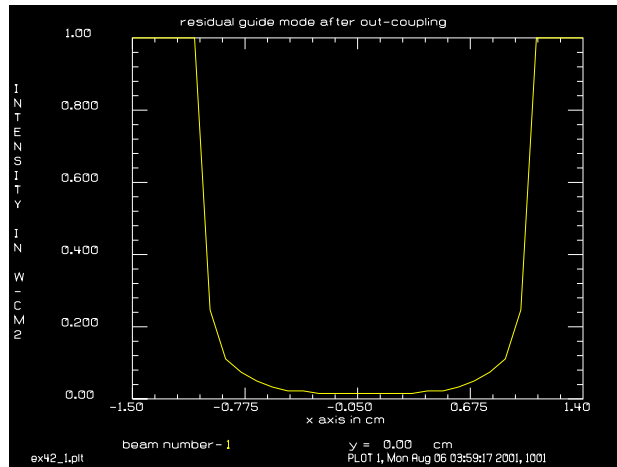


Fig. 42.2. Guide mode after passing under the outcoupling grating. The mode is strongly depleted in the center.

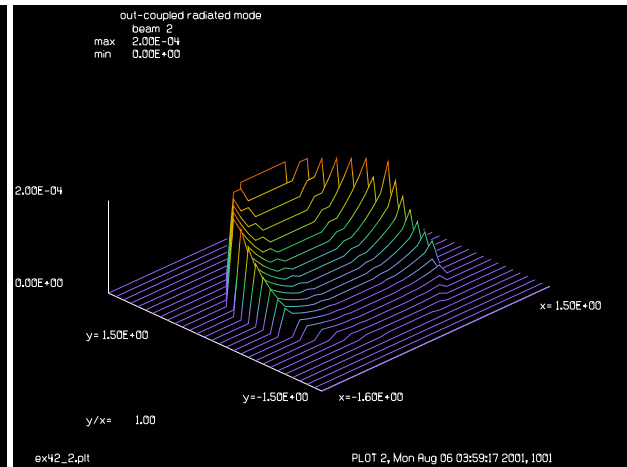


Fig. 42.3. Radiated mode after outcoupling.

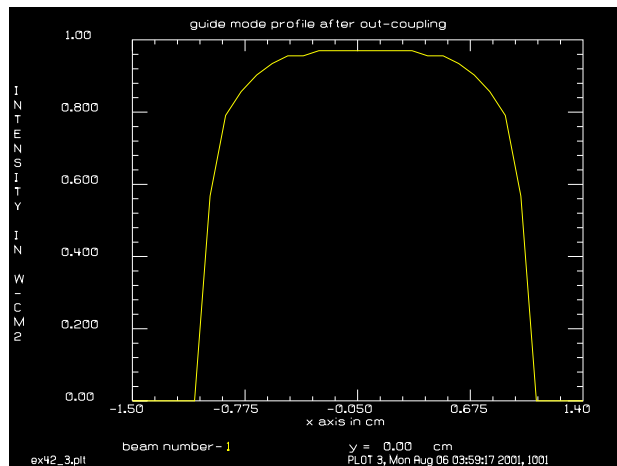


Fig. 42.4. Guide mode after incoupling. Incident beam and ideal radiation mode are identical.

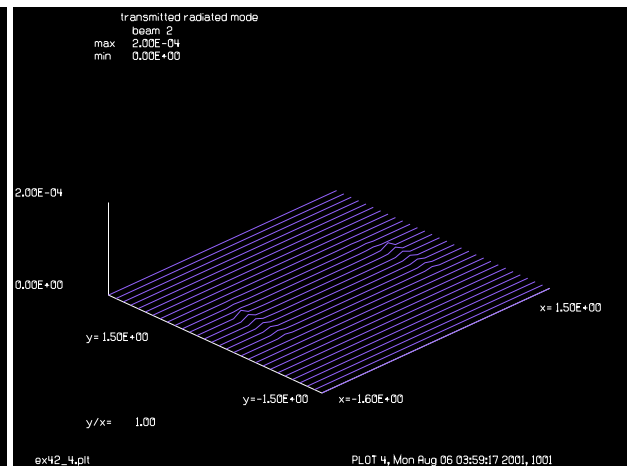


Fig. 42.5. Transmitted radiation mode. Even with ideal mode matching the absorption is not perfect at the aperture edges because there is no incident backward guided mode.

c subsequent incoupling with the identical mode. The grating is
 c of constant coupling constant in a circular grating region. The
 c input guided mode is of unit amplitude. The guided mode is mostly
 c diffracted into the radiated mode in the region of the grating.
 c The undiffracted guided mode residuals are shown in Plot 1.
 c The radiated mode is in the form of a "sugar scoop" pattern
 c because of the relative strong coupling constant of 1 cm⁻¹ and
 c the circular grating region (Plot 2).
 c
 c The incoupling is accomplished by forming the inner product of the
 c of the conjugate of the normalized radiation mode with the incident
 c radiation mode.
 c

c In Ex 42, the incident radiation mode is exactly mode matched. However,
 c the transmitted guided mode (residual from the radiation mode
 c formation) is not included so the initial unit amplitude guided mode
 c is not recreated. In particular, the intensity rolls off at the
 c edge of the aperture, as shown in Plot 3. The transmitted radiation
 c is shown in Plot 4. Even though the radiation modes are exactly
 c matched, the absorption of radiation is not perfect because the
 c there is no incident back coupled guided mode. The transmitted
 c radiation is strongest on the sides of the circular grating patch
 c where most of the radiation went past the grating in the outcoupled
 c beam without being diffracted.

```
c
echo/on
nbeam 2                                # form the guided and radiated beams
array/set 1 32 1 1                     # guided mode is 32 X 1
array/set 2 32 32 1                   # radiated mode is 32 X 32
clear 1 1
clear 2 0
units/s 1 .1 .0001                    # width of guided beam is 1 micron
units/s 2 .1 .1                       # units of radiated mode
wavelength/set 0 .78                  # set wavelength to .78 micron
status/p
energy
debug hoe
slab/hoe/cir 1.05 0. 0. 1.            # define circular aperture to be 1.05 radius
slab/hoe/ecoef 1.                     # set coupling constant to 1 cm-1
slab/out 1 2
title residual guide mode after out-coupling
plot/watch ex42_1.plt
plot/x/in 1 fmax=1. fmin=0.
intmap 1
intmap 2
title out-coupled radiated mode
plot/watch ex42_2.plt
plot/l 2 max=2.e-4 min=0.
energy
nbeam 3                                # form a beam for the normalized mode
array/s 3 32 32 1
units 3 .1 .1
wavelength/set 3 .78
copy 2 3                               # copy Beam 2 to Beam 3 to make normalized mode
c                                       # initial beam 1 was of unit amplitude
slab/in 2 3 1                          # input Beam 2 using Beam 3 as normalized mode
c                                       # and using Beam 1 as the guided mode
intmap 1
intmap 2
intmap 3
title guide mode profile after out-coupling
plot/watch ex42_3.plt
plot/x/in 1 fmax=1. fmin=0.
title transmitted radiated mode
plot/watch ex42_4.plt
plot/l 2 max=2.e-4 min=0.
end
```

Jump to: [Commands](#), [Theory](#)

Ex43: Waveguide grating coupler, reversed mode input

This example illustrates outcoupling of waveguide grating and subsequent incoupling with the reversed mode obtained from focusing onto the optical data storage surface, reflection, and return to the grating for incoupling. The incident mode is not only reversed in direction, with the peak of the “sugar scoop” on the upstream side, but there are some diffraction effects because of the diffraction propagation of the beam. The incoupled radiation is less than half the initial value. The transmitted incident beam shows that mode matching is worst at the upstream and downstream parts of the grating aperture.

Input: ex43.inp

```
c## ex43
c
c Example 43: Waveguide grating coupler, reversed mode input
c
c This example illustrates outcoupling of waveguide grating and
c subsequent incoupling with the reversed mode obtained from
c focusing onto the optical data storage surface, reflection,
c and return to the grating for incoupling. The incident mode is
c not only reversed in direction, with the peak of the 'sugar
c scoop" on the upstream side, but there are some diffraction
c effects because of the diffraction propagation of the beam
c
nbeam 2                                # form the guided and radiated beams
array/set 1 32 1 1                      # guided mode is 32 X 1
array/set 2 32 32 1                    # radiated mode is 32 X 32
clear 2 0
units/s 1 .1 .0001                     # width of guided beam is 1 micron
units/s 2 .1 .1                         # units of radiated mode
wavelength/set 0 .78                    # set wavelength to .78 micron
status/p
energy
slab/hoe/cir 1.05 0. 0. 1.             # define circular aperture to be 1.05 radius
slab/hoe/ecoef 1.                       # set coupling constant to 1 cm-1
slab/out 1 2
title outcoupled radiation mode
plot/watch ex43_1.plt
plot/l 2 max=2e-4 min=0.
energy
nbeam 3                                # form a beam for the normalized mode
array/s 3 32 32 1
units/s 3 .1 .1
wavelength/set 3 .78
copy 2 3                                # copy Beam 2 to Beam 3 to make normalized mode
c                                        # initial beam 1 was of unit amplitude
lens 2 1                                # form lens with focal length f=1
dist 1 2
mirror/flat 2
dist -1 2
lens 2 1
title returned radiation mode
plot/watch ex43_2.plt
```

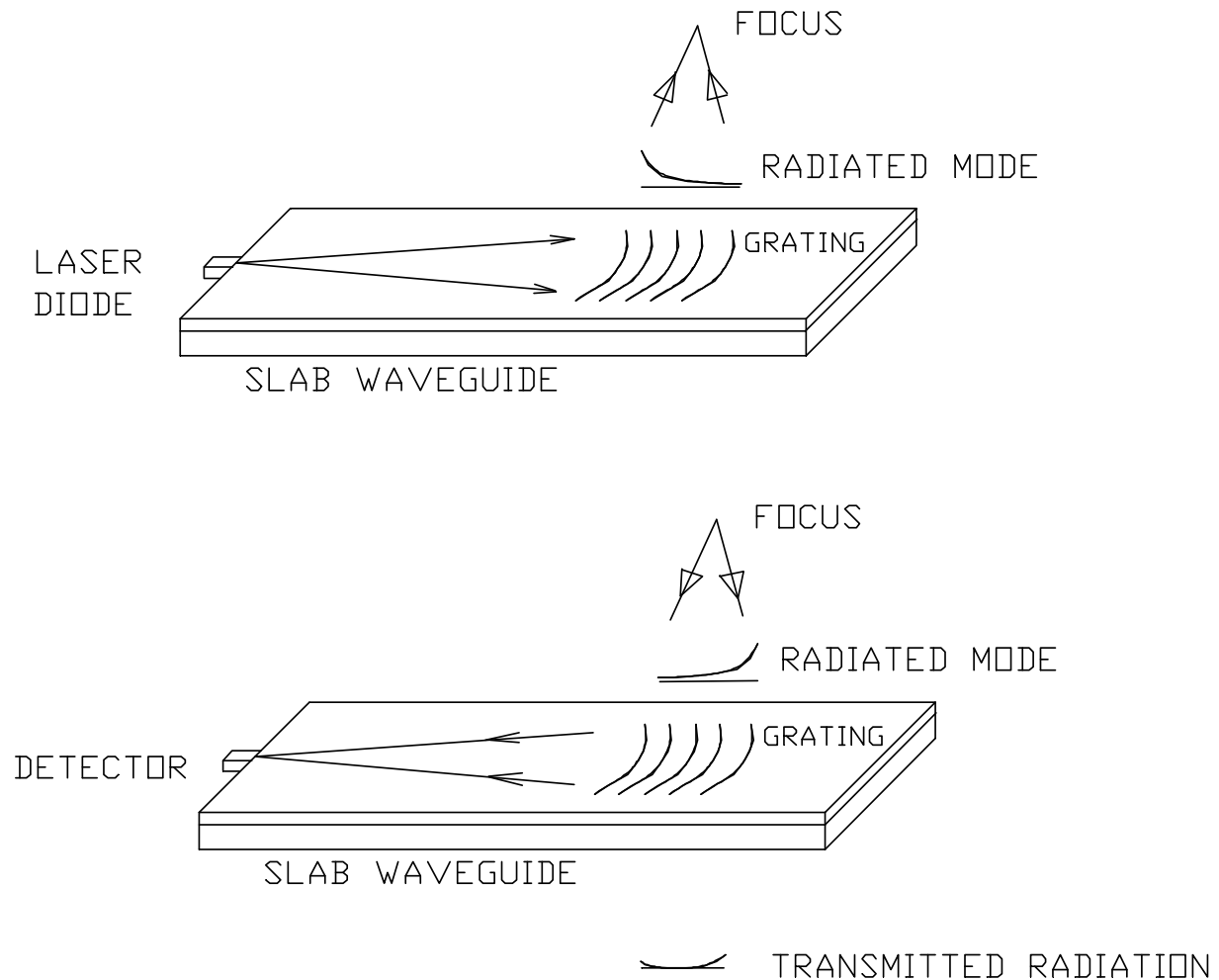


Fig. 43.1. Schematic illustrating outcoupling of radiation, focusing on an optical data storage surface, reflection, and return of the radiation with the distribution reversed. In outcoupling (top), light from a laser diode is injected into a slab waveguide. The light propagates as a guided mode in the vertical direction — trapped in a boundary layer of higher index. A grating of variable index or surface relief is put on the grating. The grating lines are curved to form a diffracted beam which converges to the focus. The guided mode decays as it passes under the grating, as the light is scattered out. The beam is reflected at the focus by the optical data storage surface and returns to the grating (bottom). The returned, incident mode is flipped with respect to the outgoing beam and is, therefore, not mode matched. The light is partly scattered into the grating and partly transmitted, because the mode is not perfectly matched. The guided mode is focused onto a detector. A more detailed arrangement — not shown for simplicity — allows optical separation of the laser diode and the detector paths.

```
plot/1 2 max=2e-4 min=0.
slab/in 2 3 1          # input Beam 2 using Beam 3 as normalized mode
c                      # and using Beam 1 as the guided mode
intmap 1
intmap 2
intmap 3
title waveguide mode after in-coupling
plot/watch ex43_3.plt
```

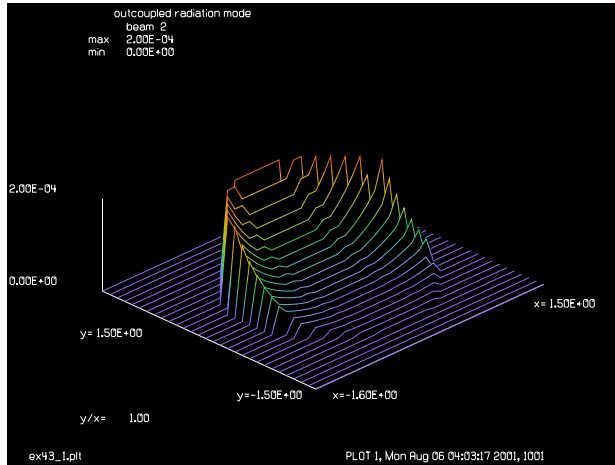


Fig. 43.2. Radiated mode after outcoupling.

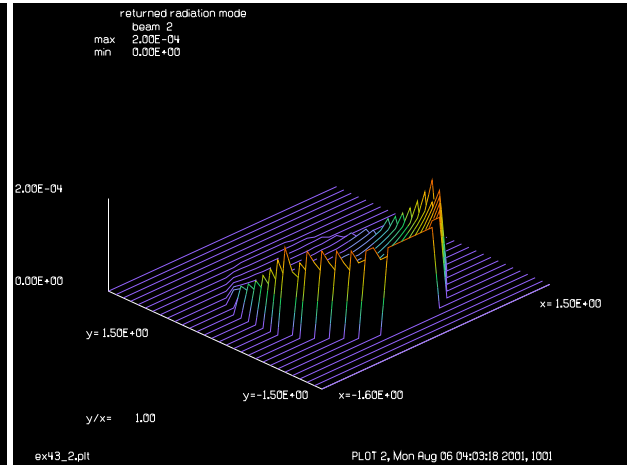


Fig. 43.3. Incident radiation mode after reflection and return from the optical data storage surface.

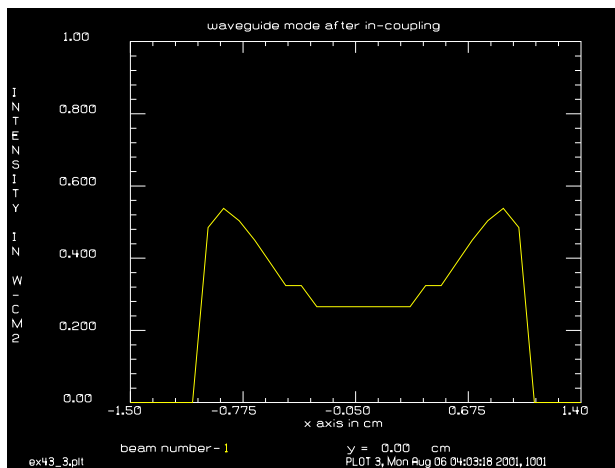


Fig. 43.4. Guide mode after incoupling. The round trip efficiency is lower than Ex. 42 because of the incident mode is reversed.

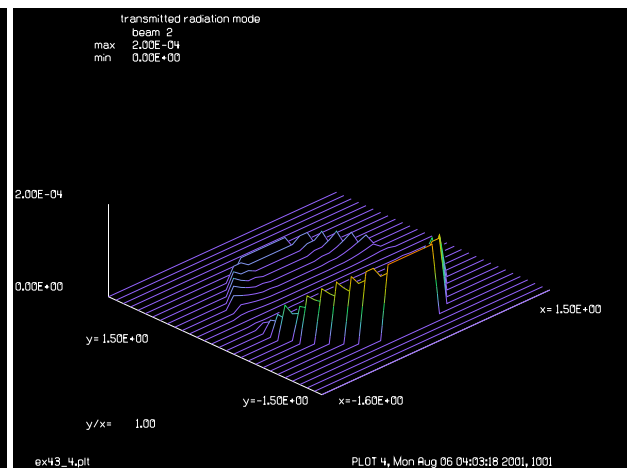


Fig. 43.5. Transmitted radiation mode. Note that the radiation is most poorly matched at the upstream and down stream parts of the grating aperture.

```

plot/x/in 1 fmax=1 fmin=0.
title transmitted radiation mode
plot/watch ex43_4.plt
plot/l 2 max=2e-4 min=0.
end

```


Ex44: Waveguide grating coupler, reversed mode input with aberration

This example illustrates outcoupling of waveguide grating and subsequent incoupling with the reversed mode obtained from focusing onto the optical data storage surface, reflection, and return to the grating for incoupling. This example is the same as Ex. 43, except for the addition of aberration.

Aberration is added to show further decrease in the input efficiency. Odd aberrations cause a decrease in efficiency, but even aberrations do not because the aberration is not affected by the pupil reversal.

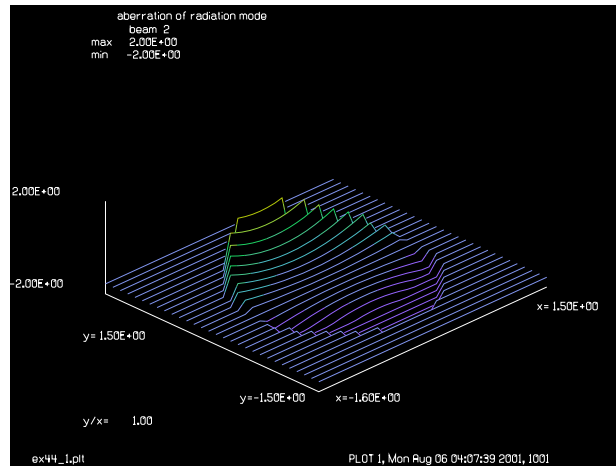


Fig. 44.1. Coma added to the grating.

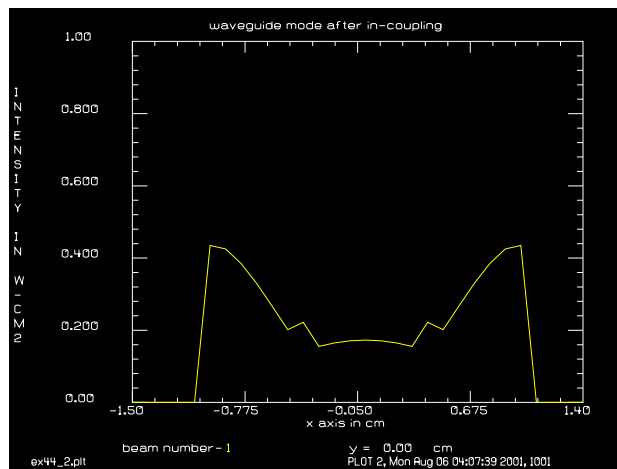


Fig. 44.2. Guide mode after incoupling. The input efficiency is lower than Ex. 43 because the coma reduces the coupling.

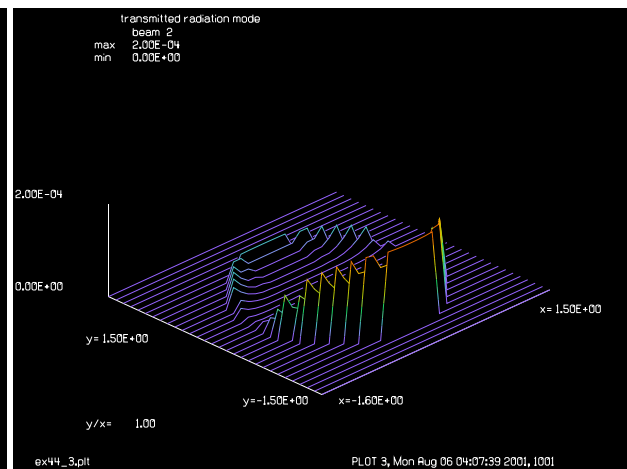


Fig. 44.3. Transmitted radiation, showing the effects of mode reversal and aberrations.

Input: `ex44.inp`

```
c## ex44
c
c Example 44: Waveguide grating coupler, reversed mode input
c               with aberration
```

```

c
c This example illustrates outcoupling of waveguide grating and
c subsequent incoupling with the reversed mode obtained from
c focusing onto the optical data storage surface, reflection,
c and return to the grating for incoupling. This example is the
c same as Ex. 43, except for the addition of aberration.
c
c Aberration is added to show further decrease in the input
c efficiency. Odd aberrations cause a decrease in efficiency, but
c even aberrations do not because the aberration is not affected by
c the pupil reversal.
c
nbeam 2                                # form the guided and radiated beams
array/set 1 32 1 1                      # guided mode is 32 X 1
array/set 2 32 32 1                    # radiated mode is 32 X 32
clear 2 0
units/s 1 .1 .0001                     # width of guided beam is 1 micron
units/s 2 .1 .1                         # units/s of radiated mode
wavelength/set 0 .78                   # set wavelength to .78 micron
status/p
energy
slab/hoe/cir 1.05 0. 0. 1.             # define circular aperture to be 1.05 radius
slab/hoe/ecoef 1.                       # set coupling constant to 1 cm-1
slab/out 1 2
energy
nbeam 3                                # form a beam for the normalized mode
array/s 3 32 32 1
units/s 3 .1 .1
wavelength/set 3 .78
copy 2 3                                # copy Beam 2 to Beam 3 to make normalized mode
c                                        # initial beam 1 was of unit amplitude
abr/coma 2 -1                           # coma of -1 wave added
title aberration of radiation mode
plot/watch ex44_1.plt
plot/l/ph 2 max=2 min=-2
lens 2 1                                # form lens with focal length f=1
dist 1 2
mirror/flat 2
dist -1 2
lens 2 1
slab/in 2 3 1                           # input Beam 2 using Beam 3 as normalized mode
c                                        # and using Beam 1 as the guided mode
energy
intmap 1
intmap 2
intmap 3
title waveguide mode after in-coupling
plot/watch ex44_2.plt
plot/x/in 1 fmax=1 fmin=0.
title transmitted radiation mode
plot/watch ex44_3.plt
plot/l 2 max=2e-4 min=0.
end

```

Ex45: Unstable ring resonator with 90 degree rotation and lensing medium

This example illustrates a ring resonator with a 90° rotation of the beam. The active medium has a relatively strong lensing effect which is corrected by an active mirror capable of putting astigmatism into the beam. The resonator is modeled schematically as shown in Fig. 45.1.

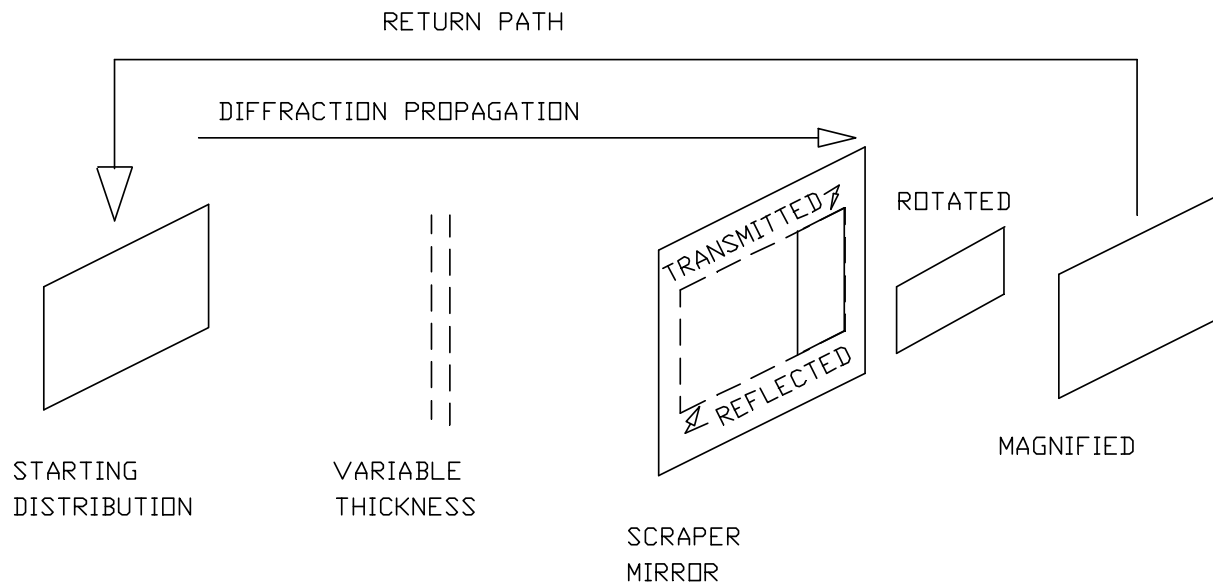


Fig. 45.1. Schematic of the ring resonator in unfolded form. The initial distribution propagates 550 cm, is apertured by the scraper mirror, magnified by 1.732, rotated by 90° and fed back at the original point. The medium, when active, adds a slight magnification change and one cm of length.

The cavity begins with a rectangular aperture and a propagation distance to achieve Fresnel numbers of 10.5 and 3.5 in the wide and narrow directions with a propagation of 550.51 cm. The scraper is a large mirror with a rectangular hole of 0.66×1.15 cm, decentered by -0.67 cm. The energy reflected is outcoupled. The energy transmitted is fed back into the resonator after being decentered, rotated 90°, and magnified by 1.732.

It is assumed that an active mirror is imaged into the center of the active medium by relay optics. The adaptive optic can maintain the output of the active medium to be a collimated beam but there will be a residual effect of a slight increase in magnification (a factor of 1.00158) and an increase in the effective length of the resonator of 1 cm. Some of the parameters are summarized in Table 45.1.

Table. 45.1. Principle terms.

rectangular aperture	2×1.1547
length	550.51
feedback aperture	0.66666×0.1547 cm, decentered -0.66666 cm
magnification (after scrapper)	1.732
rotation	90°
λ	1.73

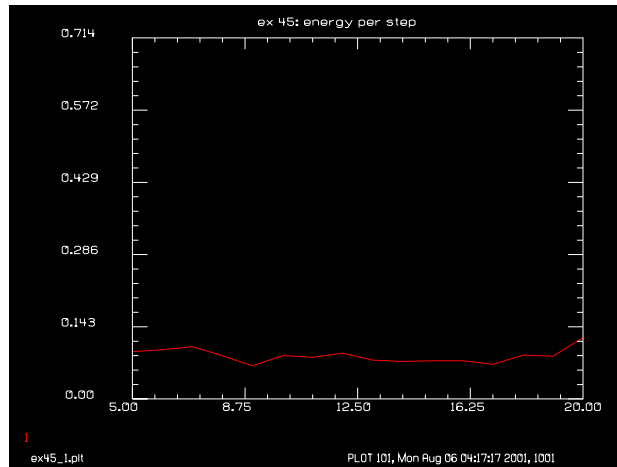


Fig. 45.2. Energy loss per pass showing convergence of the resonator. The calculation was done with no medium effects.

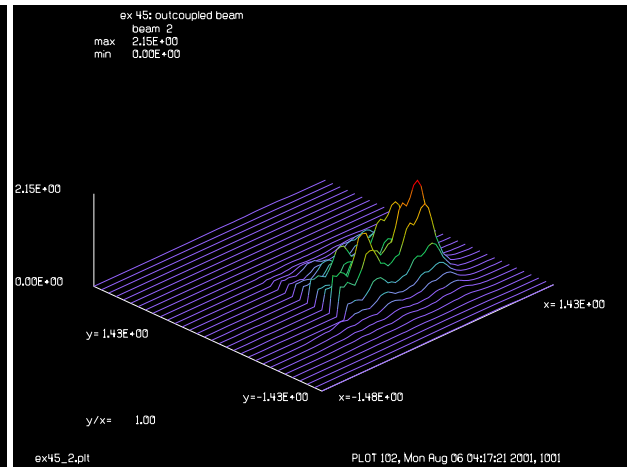


Fig. 45.3. Outcoupled beam. Scrapper mirror has a rectangular hole to allow feedback into the resonator. Note the tendency to form to lobes. The medium is neglected in this calculation.

Input: ex45.inp

```
c## ex45!57707509173297
c
c Example 45: Ring Resonator with 90 deg. Rotation, and Lensing Medium
c
c rectangular aperture      2 X 1.1547 cm, a
c length                    550.51
c feedback aperture        .66666 X 1.1547 cm, decentered -.66666 cm
c magnification             1.732
c (after scrapper)
c 90 deg. rotation
c lambda                    1.732
c Fresnel number, X        10.5
c Fresnel number, Y        3.5
c
```

poptext/compose ex45.txt

Example of a ring resonator with a 90 degree rotation.

Outcoupling occurs in a rectangular region on the left side of the distribution.

The device is almost converged after 20 passes.

poptext/end

poptext ex45.txt 8

variab/dec/int pass STOP

macro/def ex45/over

pass = pass + 1

step = step + 1

dist 551.51 1

magnify 1 1.001

copy/con 1 2

obs/rec 2 .33333 .57735 -.66666

increment pass counter

increment step number

with extra 1 cm of length

with increase in magnification

obscuration applied to output

Jump to: [Commands](#), [Theory](#)


```

intmap 2
clap/rec/c 1 .33333 .57735 -.66666      # aperture for feedback
shift 1 .66666 90.
variab/set energy 1 energy
udata/set pass step energy
plot/watch plot1.plt
plot/udata first=1 last=1 min=0
gain/converge/test 1 STOP              # store convergence test in test.
gain/eigenvalue/show 1                # display eigenvalues.
energy/norm 1 1
poptext/compose ex45.txt
Example of a ring resonator with a 90 degree rotation.

```

Outcoupling occurs in a rectangular region on the left side of the distribution.

```

pass @pass of 20
poptext/end
poptext ex45.txt 12
  plot/watch plot2.plt
  plot/l 2 ns=64
  set/density 64 64
  plot/bit/i/burn 2
  set/density 32 32
  plot/bit/i/paint 2
  plot/bit/i/con 2
  if STOP macro/exit                  # conditional exit
  transpose 1                        # 90 deg. rotation
  magnify 1 1.732
  rescale 1 size
  gain/eigenvalue/set 1
macro/end
  echo/on
  nbeam 2
  pass = 0                            # Macro pass number
  size = 1.47619                      # Define field radius
  units/field 0 size
  wavelength/set 0 1.73
  clap/rec/c 1 1. .57735
  gain/converge/set eps1=.005 eps2=.001 npoints=3
  gain/eigenvalue/set 1
c
c eps1= fractional difference between min and max energy in last
c      npoints passes for convergence.
c eps2= fraction field change for convergence.
c
c Call the macro requesting up to 30 passes with conditional exit on
c convergence.
c
  plot/screen/pause 3
  macro ex45/20
  title ex 45: energy per step
  plot/watch ex45_1.plt
  plot/udata first=1 last=1 left=5 min=0.

```

Jump to: [Commands](#), [Theory](#)

```
c
c plot converged field distribution.
c
  title ex 45: outcoupled beam
  plot/watch ex45_2.plt
  plot/liso 2
  pause 5
  end
```

Ex46: Beam shaping filter

This examples illustrates a beam shaping element based on a complex amplitude filter. It is possible to convert a gaussian beam into a flat top function by multiplying by a complex amplitude function $t(x,y)$ of the form,

$$t(x, y) = \frac{f(x, y)}{g(x, y)t_{max}} \quad (46.1)$$

where $t(x, y)$ is the complex amplitude transmission function, $f(x, y)$ is the function desired, t_{max} is the maximum transmission value, and $g(x, y)$ is the gaussian function to be converted. It is advantageous to work in the frequency domain of the desired function $f(x, y)$.

High fidelity can be achieved by designing the filter to work with a wide gaussian in the focal plane, but the efficiency is very poor. Making the gaussian smaller improves the energy efficiency at the cost of a loss of fidelity.

This calculation investigates a proposed method of making a beam shaping filter suggested by W. B. Veldkamp in "Laser beam profile shaping with interlaced binary diffraction gratings," Appl. Opt. 21, 3209 (1982).

GLAD enables one to calculate the complex amplitude filter function. The filter function, in this example, is contained in Beam 2 and may be output to a disk file using the output command. The output information can be used in the detailed design of the binary optic filter.

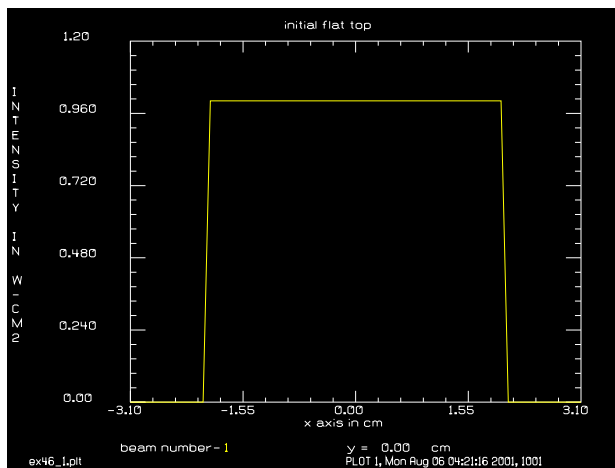


Fig. 46.1. Initial flat top distribution.

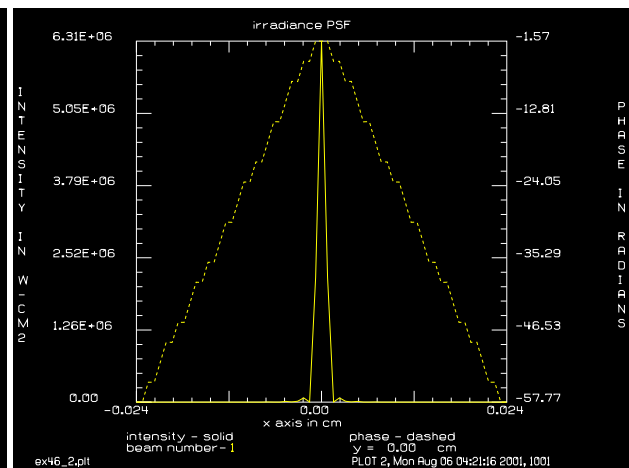


Fig. 46.2. Intensity point spread function of flat top.

Input: ex46.inp

```
c## ex46
c
c Example 46: Beam Shaping Filter
c
c A complex amplitude filter may be used to provide intensity shaping.
c In this example, the filter is placed at a focus plane.
c To determine the optimum complex amplitude filter, form the point
```

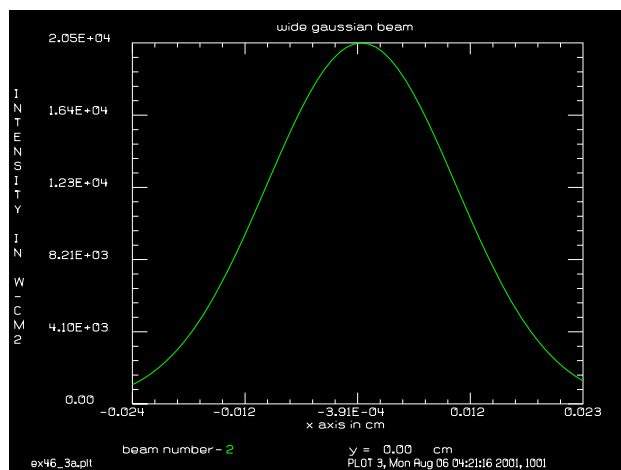


Fig. 46.3. Wide gaussian.

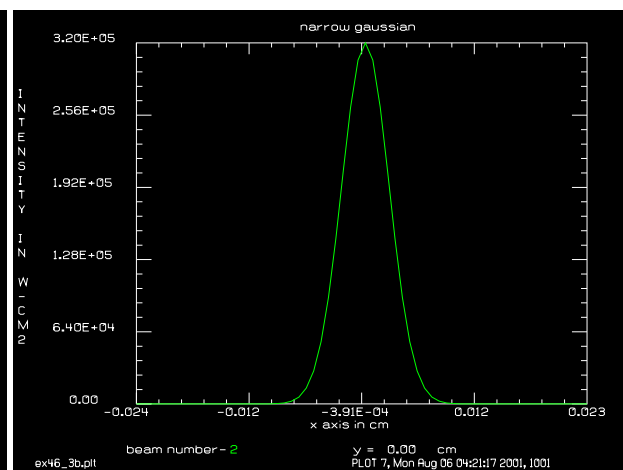


Fig. 46.4. Narrow gaussian.

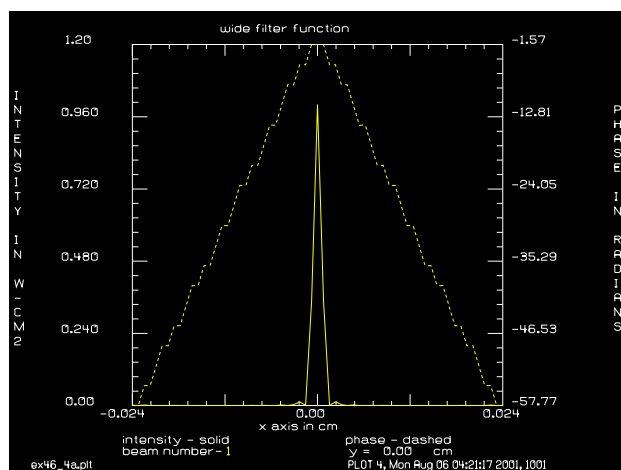


Fig. 46.5. Filter function for wide gaussian.

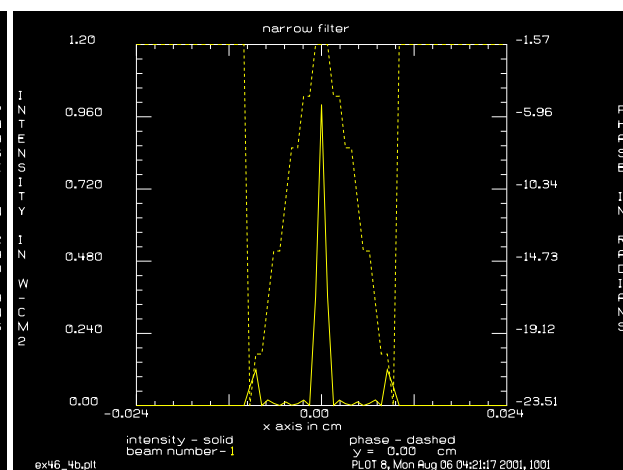


Fig. 46.6. Filter function for narrow gaussian.

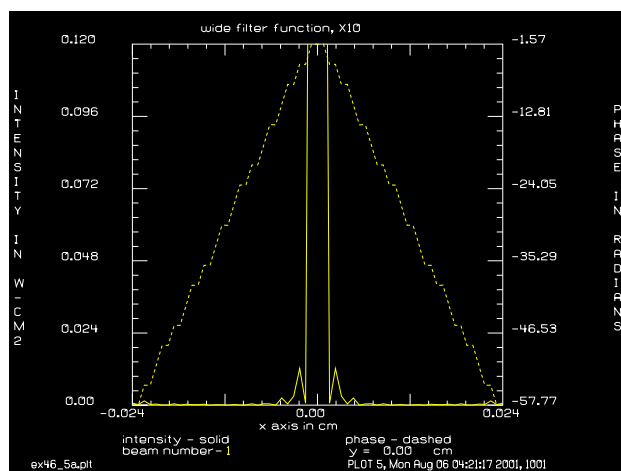


Fig. 46.7. Filter function for wide gaussian, 10X.

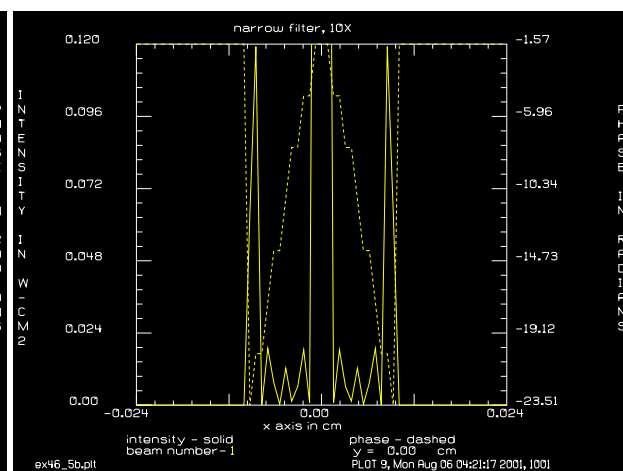


Fig. 46.8. Filter function for narrow gaussian, 10X.

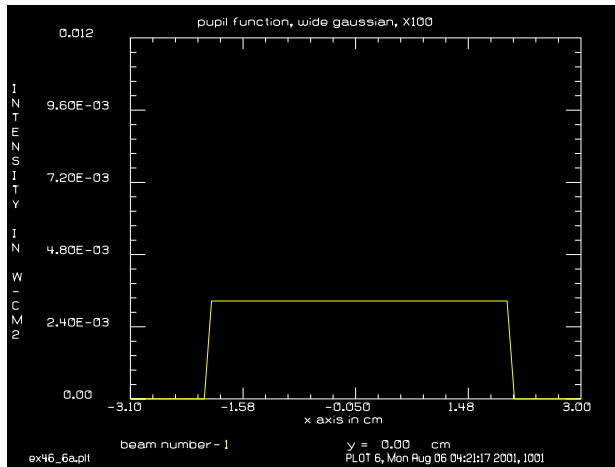


Fig. 46.9. Pupil function, wide gaussian, 100X.

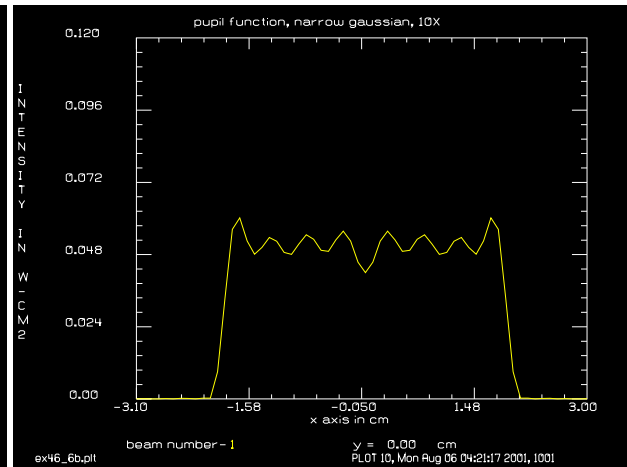


Fig. 46.10. Pupil function, narrow gaussian, 10X.

c response function of the desired beam shape -- a flat top function
 c in this example -- at the focal plane. The function which is to be
 c transformed into a top hat is a gaussian beam. Let $a(x,y)$ be the
 c Fourier transform of the desired function and $g(x,y)$ be the gaussian
 c distribution in the focal plane, which is to be filtered. The transmission
 c function is

$$t(x,y) = a(x,y) / g(x,y) / t_{\max}$$

c where t_{\max} is a normalizing factor to keep $|t(x,y)| \leq 1$.

c We first form $a(x,y)$, then $g(x,y)$, then $1./g(x,y)$, then $a(x,y)/g(x,y)$,
 c then normalize to a maximum value of 1. The amplitude transmission
 c is then stored in Beam 2 and multiplied into gaussians of different sizes
 c and the far-field patterns formed to assess how close to a flat top
 c the final beam is. The best match to a flat top function is achieved when
 c the gaussian is the size for which the filter was designed.

c The first case uses a gaussian function of radius $2.e-2$. This gives
 c excellent match to the flat top function but is very inefficient.

c The second case uses a gaussian function of radius $0.25e-2$. The match
 c is not perfect but the efficiency is better. The filter function is
 c clipped by an aperture to improve the normalization.

c echo/on

```
wavelength/set 1 .5           # wavelength is .5 microns
array/s 1 64                  # use 64 x 64 array
units/s 1 .1                  # set the units to .1
clap/c/c 1 2                  # clear aperture radius is 2
energy/norm 1 12.57           # normalize energy to give
                                # intensity = 1.

title initial flat top
plot/watch ex46_1.plt         # define external plot file
plot/x/int fmax=1.2           # plot beam profile
lens 1 100                    # lens of 100 focal length
```

Jump to: [Commands](#), [Theory](#)

```

dist 100                                # propagate 100 to the focus
                                         # this is a(x,y)

title irradiance PSF
plot/watch ex46_2.plt
plot/x                                  # plot of PSF of image of flat top
nbeam 3                                # initialize two more beams
copy 1 3                               # make a copy of Beam 1 for later use
units/s 2 7.812e-4                     # set the units the same as Beam 1
width = 2e-2
gaus/c/c 2 1 width                     # define the gaussian beam
energy/norm 2 12.57                    # normalize gaussian energy

title wide gaussian beam
plot/watch ex46_3a.plt
plot/x/int 2                           # plot wide gaussian function
inverse 2                              # form the inverse of the gaussian
mult/beam 1 2                          # form  $t(x,y) = a(x,y)/g(x,y)$ 
peak/norm 1                            # normalize so  $|t_{max}| = 1$ .

title wide filter function
plot/watch ex46_4a.plt
plot/x fmax=1.2                        # plot wide filter function

title wide filter function, X10
plot/watch ex46_5a.plt
plot/x fmax=.12                        # plot wide filter function, X10
copy/con 1 2                           # store filter in Beam 2
gaus/c/c 1 1 width                     # recreate gaussian
energy/norm 1 12.57
mult/beam 1 2                          # form  $g(x,y)t(x,y)$ 
energy
dist -100 1                            # recreate pupil
title pupil function, wide gaussian, X100
plot/watch ex46_6a.plt
plot/x/int fmax=.012

copy 3 1                                # Make a new run with small gaussian
width = .5e-2                           # recreate original image
gaus/c/c 2 1 width                       # make new filter
energy/norm 2 12.57
title narrow gaussian
plot/watch ex46_3b.plt
plot/x/int 2
inverse 2
mult/beam 1 2
clap/c/c 1 1.e-2
peak/norm 1
title narrow filter
plot/watch ex46_4b.plt
plot/x 1 fmax=1.2
title narrow filter, 10X
plot/watch ex46_5b.plt
plot/x fmax=.12
copy/con 1 2
gaus/c/c 1 1 width
energy/norm 1 12.57
mult/beam 1 2

```

Jump to: [Commands](#), [Theory](#)

```
energy
dist -100 1
title pupil function, narrow gaussian, 10X
plot/watch ex46_6b.plt
plot/x/int fmax=.12
end
```


Ex47: Gain sheet modeling

Table. 47.1. Table of Ex47 examples

Ex47a: Unstable, Loaded Cavity Resonator with Beer's Law Gain	2
Ex47b: Unstable, loaded cavity resonator with gain sheet	4
Ex47c: Unstable, loaded cavity resonator with gain sheet, single step	6
Ex47d: Point-by-point control of gain and saturation	7
Ex47e: Point-by-point control of gain and saturation, multiple beam saturation	8

This examples illustrates gain sheet modeling. For multiple passes through a gain medium the gain may saturate according to the sum of all the intensities of all passes. Fig. 47.1 illustrates a confocal resonator similar to Ex11. The gain sheet is located about 30 centimeters left of the convex mirror. On the collimated rightward pass the inner part of the beam is more strongly saturated in the inner part of the beam where the two passes overlap. The gain saturation has the following equation,

$$\text{gain} = \frac{g_0(x, y)}{\left(1 + \frac{\sum I_p}{I_s}\right)^q} \quad (47.1)$$

where $g_0(x, y)$ is the small signal gain which may vary spatially, $\sum I_p$ is the sum of the intensities of all passes, I_s is the saturation intensity, and q takes the value 1 for homogeneous broadening and 1/2 for inhomogeneous broadening.

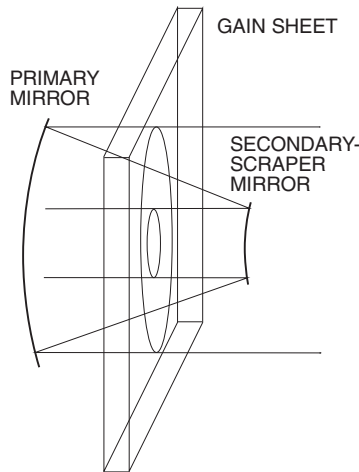


Fig. 47.1. Resonator with gain region. The intensity of both passes is used to saturate the intensity.

The effect of pass overlap is illustrated first by an idealized example shown in Fig. 47.2. A flat top distribution, of intensity 1 and with saturation intensity set to 1, is injected going to the left using the commands of ex47c.inp. On the left pass the intensity for the right pass is assumed to be zero, so the gain is uniform. On the right pass, the intensity of both passes are assumed to determine the saturation. The intensity is lower in the center because of the higher saturation due to the intensity of the left pass.

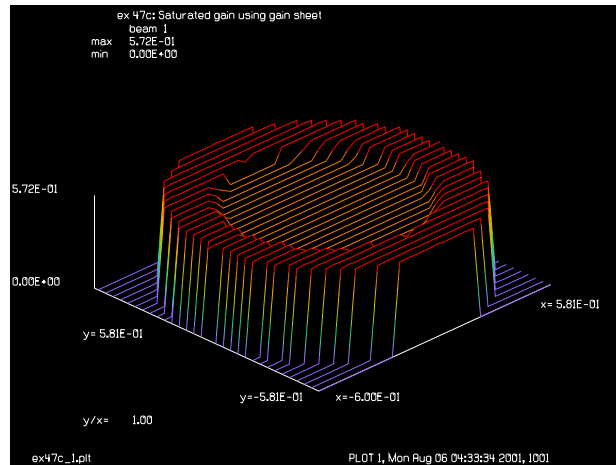


Fig. 47.2. Intensity after amplification of a flat top function after one pass through the resonator. The gain in the center is suppressed because of the intensity in the center from the first pass.

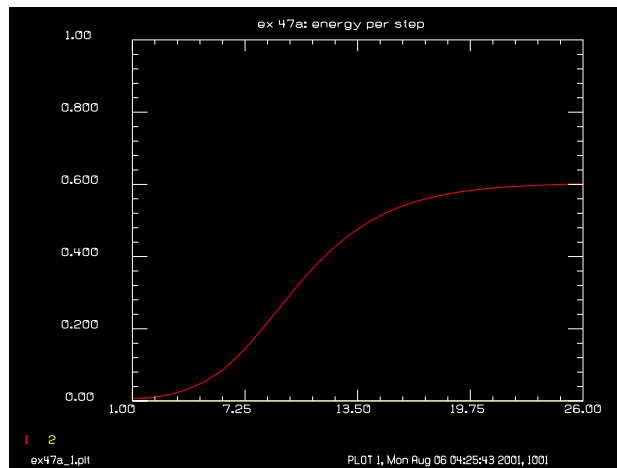


Fig. 47.3a. Energy convergence for the Beer's Law gain model which takes its saturation only from the current pass.

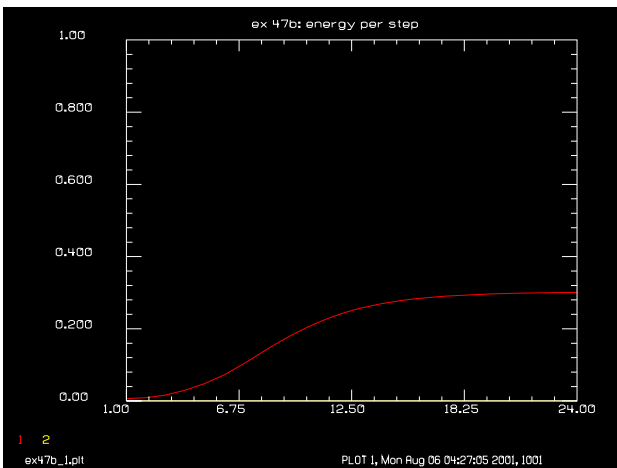


Fig. 47.3b. Energy convergence for gain sheet model. The final energy is significantly reduced by taking into account the intensity of both paths.

More realistic cases are illustrated in Ex47a with saturation based on only one pass and Ex47b with saturation based on both passes. The energies of the two examples are illustrated in Fig. 47.3a and Fig. 47.3b. The gain sheet model exhibits lower output energy. The transverse modes for the two cases are shown in Fig. 47.4a and Fig. 47.4b. The transverse modes are visually identical.

Ex47a: Unstable, Loaded Cavity Resonator with Beer's Law Gain

Input: `ex47a.inp`

```
c## ex47a!01026748758913
c
c Example 47A: Unstable, Loaded Cavity Resonator with Beer's Law Gain
c
```

Jump to: [Commands](#), [Theory](#)

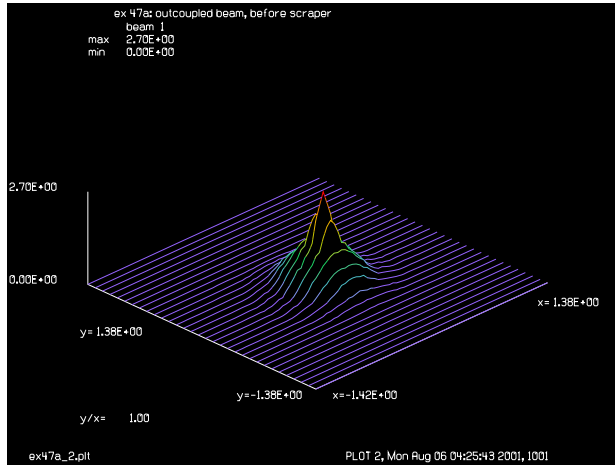
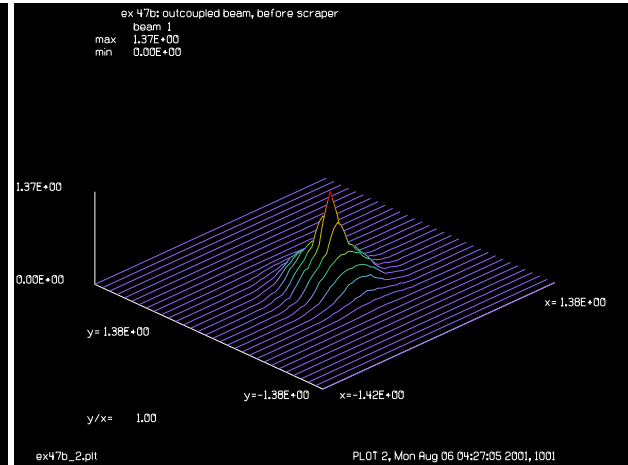


Fig. 47.4a. Transverse mode for one beam saturation.

Fig. 47.4b. Transverse mode for two-beam saturation.
The transverse modes are essentially identical.

```

c Resonator parameters
c -----
c equivalent Fresnel number      0.5
c magnification                  2.0
c length                        90.0 cm
c aperture 1                    0.3 cm
c aperture 2                    0.6 cm
c
c RESCALE compresses the field size to a radius which is sized to minimize
c aliasing, but may. For this example the rescale size is entered as a
c variable to facilitate adjusting the parameter.
c
c Variable pass is used to count the macro pass numbers and
c is input to the title.
c
c Define a macro which is a series of commands which represents one loop
c through the bare resonator cavity
c
variab/dec/int pass STOP
macro/def ex47a/over
    pass = pass + 1                # increment pass counter
    step = step + 1               # increment step number
    clap/cir/con 1 .3             # aperture of .3 cm
    mirror rad=180                # convex mirror
    status/parax
    dist -30.                     # propagate backward
    beer/noprop 1 30.
    dist -60.
    mirror rad=360.               # concave mirror
    clap/cir/con 1 .7             # aperture of .7 cm
    status/parax
    dist 60                       # forward propagation
    beer/noprop 1 30.
    dist 30.
    status/parax

```

```

variab/set energy 1 energy
udata/set pass step energy
gain/converge/test 1 STOP          # store convergence test in test.
gain/eigenvalue/show 1            # display eigenvalues.
if STOP macro/exit                # conditional exit
rescale 1 size
peak
geodata/set 1 0 0 .3 .3 1 1 1 1
gain/eigenvalue/set 1
macro/end
c
c Set variables and define convergence criterion
c
    pass = 0                      # Macro pass number
    size = .6                     # Rescale field radius
c
c Establish initial units and a gaussian field distribution
c
    units/field 1 size
    wavelength/set 1 10.
    gaus/cir/con 1 .01 size
    geodata/set 1 0 0 .3 .3 1 1 1 1
    beer/set .02 1
    gain/converge/set eps1=.005 eps2=.001 npoints=3
    gain/eigenvalue/set 1
c
c eps1= fractional difference between min and max energy in last
c       npoints passes for convergence.
c eps2= fraction field change for convergence.
c
c Call the macro requesting up to 30 passes with conditional exit on
c convergence.
c
    macro ex47a/30
    title ex 47a: energy per step
    plot/watch ex47a_1.plt
    plot/udata first=1 last=2 min=0 max=1
c
c plot converged field distribution.
c
    title ex 47a: outcoupled beam, before scraper
    plot/watch ex47a_2.plt
    plot/liso 1
end

```

Ex47b: Unstable, loaded cavity resonator with gain sheet

Input: ex47b.inp

```

c## ex47b!594743611032182
c
c Example 47B: Unstable, Loaded Cavity Resonator with Gain Sheet
c

```

Jump to: [Commands](#), [Theory](#)

```

c Resonator parameters
c -----
c equivalent Fresnel number      0.5
c magnification                  2.0
c length                        90.0 cm
c aperture 1                    0.3 cm
c aperture 2                    0.6 cm
c
c RESCALE compresses the field size to a radius which is sized to minimize
c aliasing, but may. For this example the rescale size is entered as a
c variable to facilitate adjusting the parameter.
c
c variable pass is used to count the macro pass numbers and
c is input to the title.
c
c Define a macro which is a series of commands which represents one loop
c through the bare resonator cavity
c
variab/dec/int pass STOP
macro/def ex47b/over
    pass = pass + 1                # increment pass counter
    step = step + 1               # increment step number
    clap/cir/con 1 .3             # aperture of .3 cm
    mirror rad=180                # convex mirror
    status/parax 1
    dist -30. 1                   # propagate backward
    pack/in
    gain/sheet 30.                # apply gain sheet
    pack/out
    copy 1 2                      # store intensity
    dist -60. 1
    mirror rad=360.               # concave mirror
    clap/cir/con 1 .7            # aperture of .7 cm
    status/parax
    dist 60 1                     # forward propagation
    pack/in
    gain/sheet 30.                # apply gain sheet
    pack/out
    copy 1 2                      # store intensity
    dist 30. 1
    status/parax 1
    variab/set energy 1 energy
    udata/set pass step energy
    gain/converge/test 1 STOP      # store convergence result in test
    gain/eigenvalue/show 1        # display eigenvalues.
    if STOP macro/exit            # conditional exit
    rescale 1 size
    peak 1
    geodata/set 1 0 0 .3 .3 1 1 1 1
    gain/eigenvalue/set 1
macro/end
c
c Set variables and define convergence criterion
c

```

Jump to: [Commands](#), [Theory](#)

```

pass = 0                                # Macro pass number
size = .6                               # Rescale field radius
c
c Establish initial units and a gaussian field distribution
c
nbeam 2
units/field 1 size
wavelength/set 1 10.
gaus/cir/con 1 .01 size
clear 2 0
geodata/set 1 0 0 .3 .3 1 1 1 1
beer/set .02 1
gain/converge/set eps1=.005 eps2=.001 npoints=3
gain/eigenvalue/set 1
c
c eps1= fractional difference between min and max energy in last
c npoints passes for convergence.
c eps2= fraction field change for convergence.
c
c Call the macro requesting up to 30 passes with conditional exit on
c convergence.
c
pack/set 1 2
macro ex47b/30
title ex 47b: energy per step
plot/watch ex47b_1.plt
plot/udata first=1 last=2 min=0 max=1.
c
c plot converged field distribution.
c
title ex 47b: outcoupled beam, before scraper
plot/watch ex47b_2.plt
plot/liso 1
end

```

Ex47c: Unstable, loaded cavity resonator with gain sheet, single step

Input: ex47c.inp

```

c## ex47c!180346708427979
c
c Example 47C: Unstable, Loaded Cavity Resonator with Gain Sheet
c
c Resonator parameters
c -----
c equivalent Fresnel number      0.5
c magnification                  2.0
c length                        90.0 cm
c aperture 1                    0.3 cm
c aperture 2                    0.6 cm
c
c Define a macro which is a series of commands which represents one loop
c through the bare resonator cavity

```

Jump to: [Commands](#), [Theory](#)

```

c
c This example uses the macro EX47B created in that example.
c
c Set values for variables and define convergence criterion
c
macro/def ex47c/over
    pass = pass + 1                # increment pass counter
    step = step + 1                # increment step number
    clap/cir/no 1 .3              # aperture of .3 cm
    mirror rad=180                # convex mirror
    status/parax 1
    dist -30. 1                   # propagate backward
    pack/in
        gain/sheet 30.            # apply gain sheet
    pack/out
    copy 1 2                      # store intensity
    dist -60. 1
    mirror rad=360.               # concave mirror
    clap/cir/no 1 .7              # aperture of .7 cm
    status/parax
    dist 60 1                     # forward propagation
    pack/in
        gain/sheet 30.            # apply gain sheet
    pack/out
    copy 1 2                      # store intensity
    dist 30. 1
    status/parax 1
    variab/set energy 1 energy
    udata/set pass step energy
    gain/converge/test 1 STOP      # store convergence result in test
    gain/eigenvalue/show 1        # display eigenvalues.
    if STOP macro/exit            # conditional exit
    peak 1
    gain/eigenvalue/set 1
macro/end
variab/dec/int pass STOP
pass = 0                         # Macro pass number
size = .6                        # Rescale field radius
c
c Establish initial units and a gaussian field distribution
c
nbeam 2
units/field 1 size
wavelength/set 1 .01
clear 1 1.                       # initialize beam one to uniform intensity
clear 2 0
geodata/set 1 0 0 .3 .3 1 1 1 1
beer/set .02 1
gain/converge/set eps1=.005 eps2=.001 npoints=3
gain/eigenvalue/set 1
c
c eps1= fractional difference between min and max energy in last
c      npoints passes for convergence.
c eps2= fraction field change for convergence.

```

Jump to: [Commands](#), [Theory](#)

```

c
pack/set 1 2
reson/name ex47c
reson/eigen/test 1
reson/run 1
c
c plot field after one pass.
c
title ex 47c: Saturated gain using gain sheet
plot/watch ex47c_1.plt
plot/liso 1

```

Ex47d: Point-by-point control of gain and saturation

With the `gain/sheet/point` command the small signal gain and saturation may be defined for each (x,y) point.

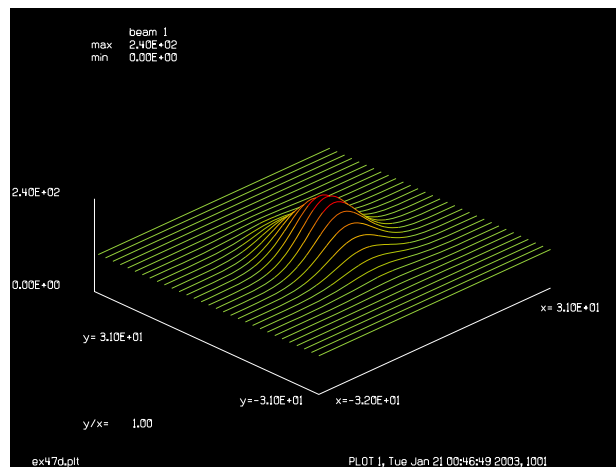


Fig. 47.5. Point definition of small signal gain.

Input: ex47d.inp

```

c## ex47d
#
# Gain sheet with point definition of small signal gain
# and saturation intensity.
#
# The first beam in pack/set is always the beam to be amplified.
# The last two beams are always g0(x,y) and es(x,y).
# If more than three beams are included in pack/set, then the
# 2nd and additional beams between the first and last two,
# are beams to contribute to saturation as with the regular
# form of gain/sheet.
#
# Arrays
# 1      beam to be amplified
# 2      small signal gain distribution
# 3      saturation intensity distribution
#

```

Jump to: [Commands](#), [Theory](#)


```

g0 = .07          # small signal gain
es = 100          # saturation intensity
nbeam 8 data      # beams 2, 3, 4 are data beams
clear 1 100       # uniform intensity for propagating beam
gaus 7 g0 20      # gaussian distribution for small signal gain
irradiance 7
gaus 8 es 20      # gaussian distribution for saturation intensity
pack/set 1 7 8    # beam 1, g0, es distributions
pack/in
  gain/sheet/point 25    # 25 cm. of gain
pack/out
prop 10           # diffraction propagation of 10 cm.
intens 1
plot/watch ex47d.plt
plot/l 1

```

Ex47e: Point-by-point control of gain and saturation, multiple beam saturation

The point-by-point definition of small signal gain and saturation may be combined with an additional optical beam. In this example, Beam 4 is treated as a second optical beam so that the saturation is due to the net effects of both Beam 1 and Beam 4.

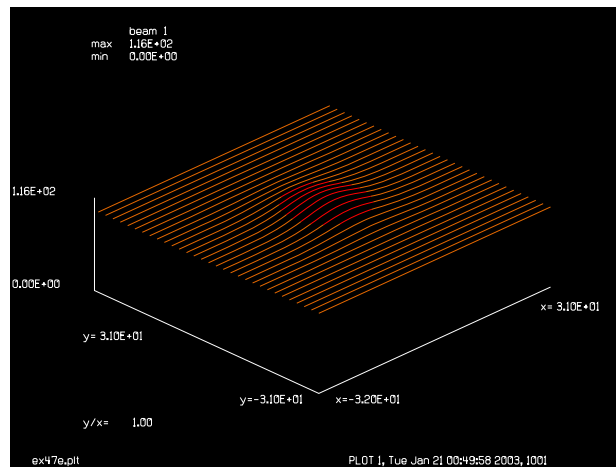


Fig. 47.6. Point definition of small signal gain with saturation due to beam 4 as well.

Input: ex47e.inp

```

c## ex47e
#
# Gain sheet with point definition of small signal gain
# and saturation intensity. Includes saturation due to combined
# irradiance of array 1 and array 4.
#
# The first beam in pack/set is always the beam to be amplified.
# The last two beams are always g0(x,y) and es(x,y).
# If more than three beams are included in pack/set, then the
# 2nd and additional beams between the first and last two,
# are beams to contribute to saturation as with the regular

```

Jump to: [Commands](#), [Theory](#)

```
# form of gain/sheet.
#
# Arrays
# 1      beam to be amplified
# 2      small signal gain distribution
# 3      saturation intensity distribution
# 4      beam to be summed with beam 1 for saturation intensity
#
g0 = .07      # small signal gain
es = 100      # saturation intensity
nbeam 4 data  # beams 2, 3, 4 are data beams
clear 1 100   # uniform intensity for propagating beam
gaus 2 g0 20  # gaussian distribution for small signal gain
irradiance 2
gaus 3 es 20  # gaussian distribution for saturation intensity
clear 4 1000  # beam to sum with beam 1 to form saturation
              # intensity
pack/set 1 4 2 3 # beam 1, beam 4, g0, es distributions
pack/in
    gain/sheet/point 25      # 10 cm. of gain
pack/out
prop 10              # diffraction propagation of 10 cm.
intens 1
plot/watch ex47e.plt
plot/l 1
```

Ex48: Frequency doubling

Table. 48.1. Table of Ex48 examples

Ex48a: Frequency Doubling: Tuned and Detuned.	1
Ex48b: Frequency doubling, single point check	3

Ex48a: Frequency Doubling: Tuned and Detuned

This example shows frequency doubling with no detuning and with detuning.

Table. 48.2. Parameters for frequency doubling.

Parameter	Value
array size	32 x 32
pump wavelength	1.06 micron
harmonic wavelength	0.53 micron
pump mode shape	gaussian, $\omega_0 = 0.5$ cm
peak pump power	10 megawatts
kappa	5.8×10^{-4} cm/photon

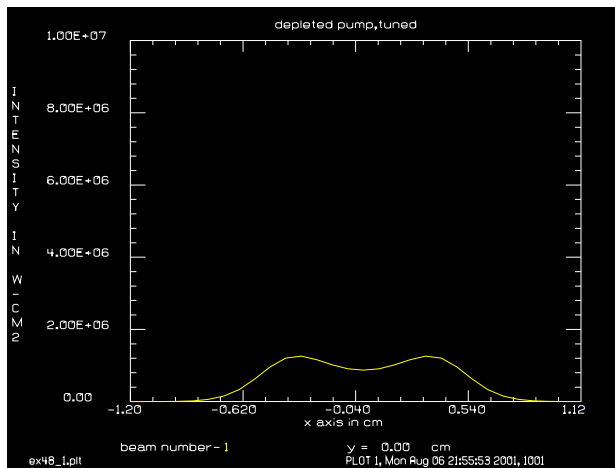


Fig. 48.1. Depleted pump (no detuning).

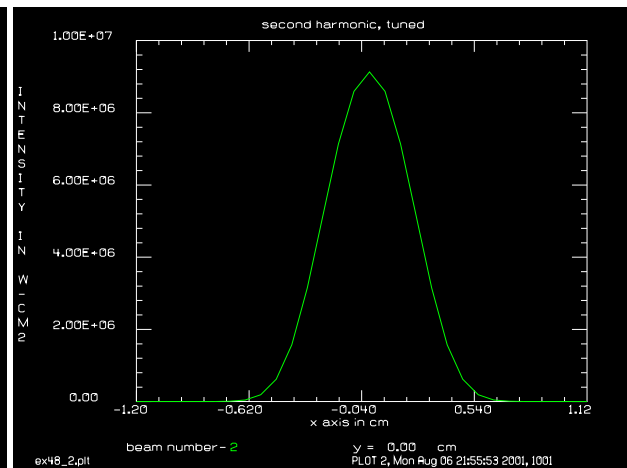


Fig. 48.2. Second harmonic (no detuning).

Input: ex48a.inp

```

c## ex48a f:\glad61_64
c
c Example 48a: Frequency Doubling
c
variable/dec/real Gain Lambda Kappa
PlankC = h*c list
Lambda1 = 1.06e-4
Lambda2 = Lambda/2
Kappa = 5.8e-4
array/set 1 32 # set Beam 1 size to 32 x 32
nbeam 2 # set up 2 beams
wavelength/set 1 Lambda1*1e4 # wavelength for Beam 1 is 1.06 microns

```

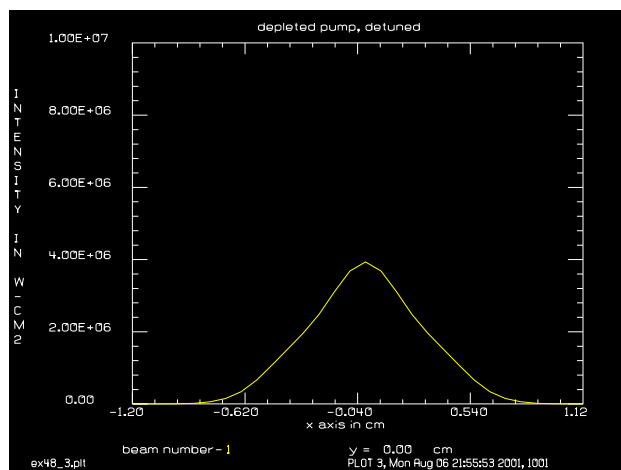


Fig. 48.3. Depleted pump with detuning.

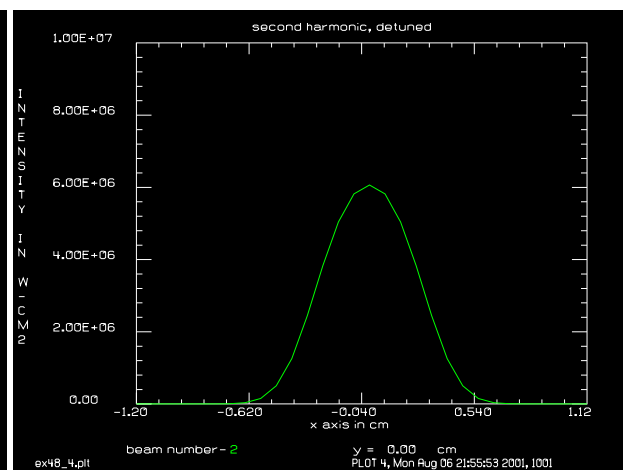


Fig. 48.4. Second harmonic with detuning showing a reduction of about 33 percent in efficiency.

```

wavelength/set 2 Lambda2*1e4 # wavelength for Beam 2 is .53 microns
units/s 0 .08
gaus/c/c 1 1e7 .5 # gaussian, peak 10 Mwatts, w0=.5 cm
clear 2 0 # Beam 2 is zero
peak
pack/set 1 2 # identify beams to be packed
pack/in # pack beams
double/steps 1. kappa=Kappa ns=30 deltak=0. list # doubling crystal
pack/out # unpack beams
energy # check that total energy is unchanged
plot/watch ex48a_1.plt
title depleted pump,tuned
plot/x/i 1 fmax=1.e7 fmin=0.
plot/watch ex48a_2.plt
title second harmonic, tuned
plot/x/i 2 fmax=1.e7 fmin=0.
c
c Repeat calculation with pi/2 detuning over 1 cm
c
zreff/se 1 0
units/s 0 .08
gaus/c/c 1 1e7 .5 # gaussian, peak 10 Mwatts, w0=.5 cm
clear 2 0 # Beam 2 is zero
energy
pack/set 1 2 # identify beams to be packed
pack/in # pack beams
double/steps 1. kappa=Kappa ns=30 deltak=1.57 list # doubling crystal
# detuning of pi/2 over 1 cm
pack/out # unpack beams
energy # check that total energy is unchanged
# note that detuning reduces efficiency
# by about 80 percent

plot/watch ex48a_3.plt
title depleted pump, detuned
plot/x/i 1 fmax=1.e7 fmin=0.

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch ex48a_4.plt
title second harmonic, detuned
plot/x/i 2 fmax=1.e7 fmin=0.
Frequency doubling, single point check

```

Input: ex48b.inp

```

c## ex48b
c
c Example 48b: Frequency Doubling, single point check
c
variable/dec/real Gain Lambda1 Lambda2 Coef1 Coef2 Plankc Sqrt2
variable/dec/real Gain2
variable/dec/complex A1 A2 dA1 dA2 A1_end A2_end dA2
Kappa = 1.e-4
Lambda1 = 1.06e-4
Lambda2 = Lambda1/2.
Dstep = .1 list
Sqrt2 = sqrt(2.)
A1 = 1e4
A2 = 10
CenterIntensity1 = A1*conj(A1) list
CenterAmplitude1 = sqrt(CenterIntensity1)
CenterIntensity2 = A2*conj(A2) list
CenterAmplitude2 = sqrt(CenterIntensity2)
gain_length = 1./Kappa/CenterAmplitude1 list
array/set 1 32 # set Beam 1 size to 32 x 32
nbeam 2 # set up 2 beams
wavelength/set 1 Lambda1*1e4 # wavelength for Beam 1 is 1.06 microns
wavelength/set 2 Lambda2*1e4 # wavelength for Beam 2 is .53 microns
units/s 0 .08
gaus/c/c 1 CenterIntensity1 .5 # gaussian pump
c plot/w plot1.plt
c title Pump start
c plot/x/i 1 fmax=CenterIntensity1
clear 2 0 # Beam 2 is zero
clear/complex 2 0. CenterAmplitude2
energy
pack/set 1 2 # identify beams to be packed
point/list/ij/sr 1
point/list/ij/si 2
variab/set A1_start 1 point/sr
variab/set A2_start 2 point/si
A1 = A1_start list
A2 = 1i*A2_start list
pack/in # pack beams
double/steps zstep=Dstep kappa=Kappa nstep=1 list # doubling crystal
pack/out
C Command Language Calculations
C -----
C Starting pump amplitude, A1 = @A1
C Startingg harmonic amplitude, A2 = @A2
E1 = A1*conj(A1) + A2*conj(A2)
dA2 = (0.+1i)*Kappa*A1*A1*Dstep

```

Jump to: [Commands](#), [Theory](#)

```
A2 = A2 + dA2
dA1 = (0.0+1i)*Kappa*A2*conj(A1)*Dstep
A1 = A1 + dA1
E2 = A1*conj(A1) + A2*conj(A2)
C Incremental pump amplitude, dA1 = @dA1
C Incremental harmonic amplitude, dA2 = @dA2
C Ending pump amplitude, A1 = @A1
C Ending harmonic amplitude, A2 = @A2
Scale = E2/E1
A1 = A1/sqrt(Scale)
A2 = A2/sqrt(Scale)
E3 = A1*conj(A1) + A2*conj(A2)
C Local starting energy, E1 = @E1
C Local ending energy, E2 = @E2
C Energy error ratio E2/E1, Scale = @Scale
C Corrected local ending energy, E3 = @E3
C
```

Ex49: Frequency doubling, closed form model

This example frequency doubling with no detuning and with detuning using a closed form model as described in Chap. 9, Theory Manual. The closed form accurately predicts the gain in the tuned case, Figs. 49.1 and 49.2. The effect of detuning, Figs. 49.3 and 49.4, in the closed form model is somewhat less than in the full model illustrated in Ex. 48.

Table. 49.1. Parameters for frequency doubling.

Parameter	Value
array size	32 x 32
pump wavelength	1.06 micron
harmonic wavelength	0.53 micron
pump mode shape	gaussian, $\omega_0 = 0.5$ cm
peak pump power	10 megawatts
kappa	5.8×10^{-4} cm/photon

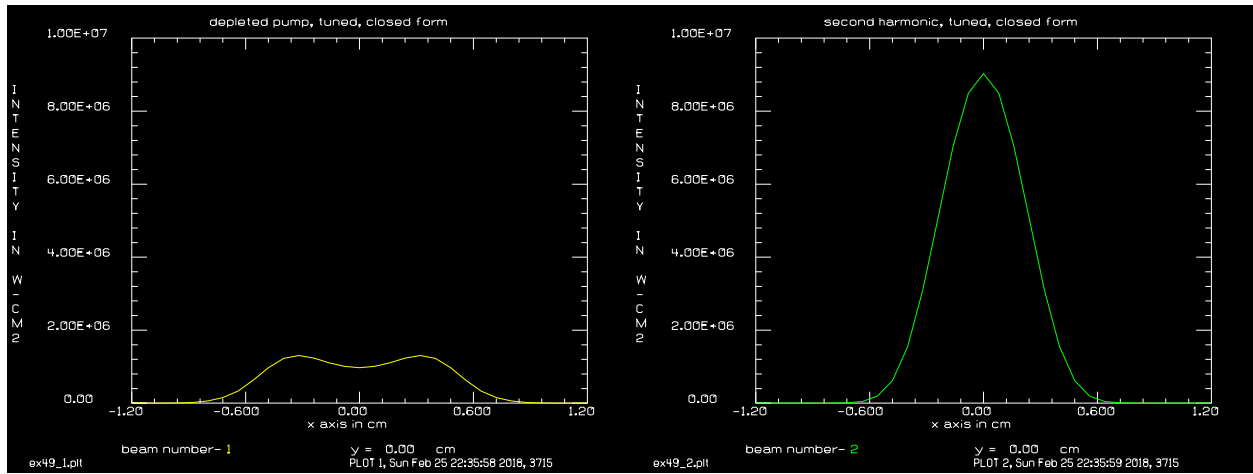


Fig. 49.1. Depleted pump (no detuning).

Fig. 49.2. Second harmonic (no detuning).

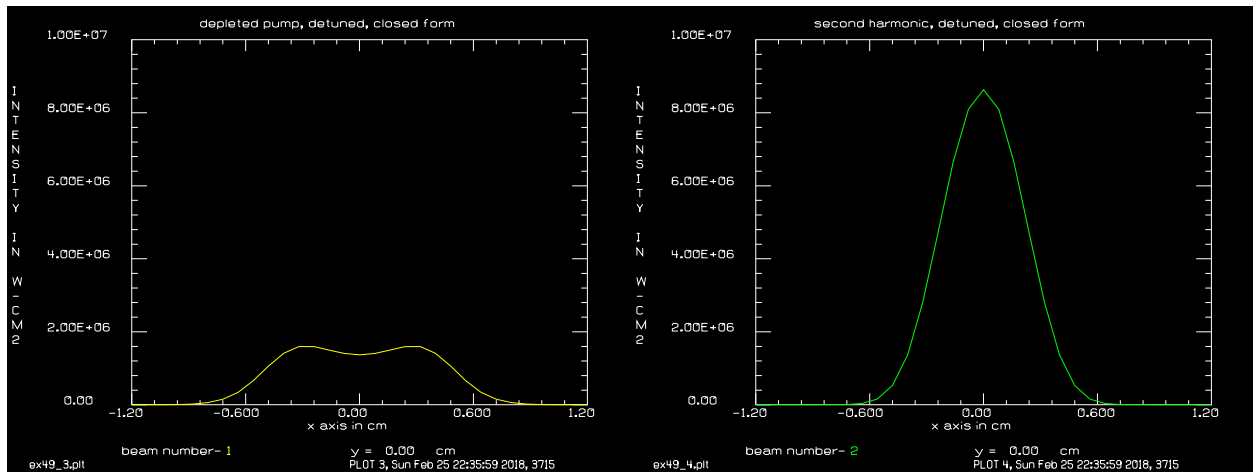


Fig. 49.3. Depleted pump with detuning.

Fig. 49.4. Second harmonic with detuning showing a reduction of about 10 percent in efficiency.

Input: ex49.inp

```

c## ex49
c
c Example 49: Frequency Doubling, Closed Form Model
c
variable/dec/int Line Kbeam1 Kbeam2 Ncys Ncxs Nline
declare/complex minus_one amp
variables/set/parameter Energy 1 energy
variables/dec/complex CenterAmplitude AmplitudeFinal
variables/dec/complex Gain Amplification_P Amplification_H Cfac
variables/dec/complex Amplification_P Amplification_H
variables/dec/complex PAmplitudeFinal HAmplitudeFinal
c
Kbeam1=1          # Pump beam number
Kbeam2=2          # Harmonic beam number
Kappa = 5.8e-4    # Irradiance gain coefficient
Zstep = 1.        # Step length. Here it is the same as crystal length [cm].
DeltaK = 0.       # Detuning [cm^-1]
Lambda1 = 1.06E-4 # Wavelength of pump [cm]
Lambda2 = Lambda1/2
CenterIntensity = 1E7 # Starting center intensity of pump [w/cm^2]
CenterAmplitude = sqrt(CenterIntensity)
c
Nline = 32
Ncxs = Nline/2+1
Ncys = Nline/2+1
array/set 1 32          # set Beam 1 size to 32 x 32
wavelength/set Kbeam1 Lambda1
units/s 0 .08
gaus/c/c Kbeam1 CenterIntensity .5          # gaussian, peak 10 Mwatts, w0=.5 cm
nbeam 2
clear Kbeam2 0.
energy
c
c Make a copy of the incident beam for later calculation of
c pump depletion.
c
c
c pump depletion, tuned
c
variab/set Line line; Line=
CenterAmplitude = sqrt(CenterIntensity)

c doubling crystal
peak
double/closed/pump Kbeam1 zstep=Zstep kappa=Kappa deltak=DeltaK list
gain_length = 1/Kappa/sqrt(CenterIntensity) list
Fac = 0; DetuningRatio = 1.
if [abs(DeltaK)>0] then
    Fac = 0.5*DeltaK*Zstep
    DetuningRatio = sin(Fac)/Fac          # Detuning ratio
endif
Coef2_Kappa = Kappa*Zstep*DetuningRatio

```

Jump to: [Commands](#), [Theory](#)


```

Amplification_P = 1./cosh(Coef2_Kappa*CenterAmplitude)
PAmplitudeFinal = Amplification_P*CenterAmplitude
C
C -----
C Kbeam1, @Kbeam1:      C Number of pump beam
C Step length, Zstep: @Zstep      C Step length
C Irradiance gain coefficient, @Kappa      C Kappa
C Wavelength, @Lambda1      C Wavelength of pump
C Detuning factor, DeltaK, @DeltaK      C Detuning factor
C DetuningRatio, @DetuningRatio:      C DetuningRatio
C Coef2_Kappa, @Coef2_Kappa:      C Kappa-Length_Detuning factor
C Center Point, (@Ncxs,@Ncys)
C Pump center amplitude, Initial: @CenterAmplitude      C Starting amplitude
C Pump center amplification, @Amplification_P      C Amplification with Kappa
C Pump center amplitude, Final, @PAmplitudeFinal      C final amplitude
C -----

plot/watch ex49_1.plt
title depleted pump, tuned, closed form
plot/x/i 1 fmax=CenterIntensity fmin=0.
c
c harmonic amplification
c
variab/set Line line; Line=
gaus/c/c Kbeam2 CenterIntensity .5          # gaussian, peak 10 Mwatts, w0=.5 cm
clear Kbeam1 0.
double/closed/harmonic Kbeam2 zstep=Zstep kappa=Kappa deltak=0. list
gain_length = 1/Kappa/sqrt(CenterIntensity) list
Fac = 0; DetuningRatio = 1.
if [abs(DeltaK)>0] then
    Fac = 0.5*DeltaK*Zstep
    DetuningRatio = sin(Fac)/Fac          # Detuning factor
endif
Coef2_Kappa = Kappa*Zstep*DetuningRatio
wavelength/set Kbeam2 Lambda2
peak
plot/watch ex49_2.plt
title second harmonic, tuned, closed form
plot/x/i 2 fmax=1.e7 fmin=0.
Cfac = -1i # square root of -1
Amplification_H = Cfac*tanh(Coef2_Kappa*CenterAmplitude)
HAmplitudeFinal = Amplification_H*CenterAmplitude
C
C -----
C Kbeam2, @Kbeam2,      C Number of harmonic beam
C Step length, Zstep: @Zstep:      C Step length
C Irradiance gain coefficient, Kappa: @Kappa      C Kappa
C Wavelength: @Lambda2      C Wavelength of harmonic
C Detuning factor, DeltaK: @DeltaK      C Detuning factor
C DetuningRatio, @DetuningRatio:      C DetuningRatio
C Coef2_Kappa, @Coef2_Kappa:      C Kappa-Length_Detuning factor
C Center Point, (@Ncxs,@Ncys)
C Pump center amplitude, Initial: @CenterAmplitude      C Starting amplitude
C Harmonic center amplification: @Amplification_H      C Amplification with Kappa

```

Jump to: [Commands](#), [Theory](#)

```

C Harmonic center amplitude, Final, @HAmplitudeFinal      C final amplitude
C -----
C Energy Conservation Check
PplusTot = PAmplitudeFinal*conj(PAmplitudeFinal)
PplusTot = PplusTot + HAmplitudeFinal*conj(HAmplitudeFinal)
C C CenterIntensity = @CenterIntensity, PplusTot = @PplusTot
C -----

c Start same analysis as above but with detuning

gaus/c/c Kbeam1 CenterIntensity .5          # gaussian, peak 10 Mwatts, w0=.5 cm
energy
c
c Make a copy of the incident beam for later calculation of
c pump depletion.
c
copy Kbeam1 Kbeam2
c
c pump depletion with detuning
c
variab/set Line line; Line=
DeltaK = 1.57
double/closed/pump Kbeam1 zstep=Zstep kappa=Kappa deltak=DeltaK list
                                # doubling crystal
                                # detuning of pi/2 over 1 cm

Fac = 0.; Detuning = 0.
if [DeltaK>0.] then
    Fac = 0.5*DeltaK*Zstep
    DetuningRatio = sin(Fac)/Fac          # Detuning factor
endif
Coef2_Kappa = Kappa*Zstep*DetuningRatio
Amplification_P = 1./cosh(Coef2_Kappa*CenterAmplitude)
AmplitudeFinal = Amplification_P*CenterAmplitude
C
C -----
C Kbeam1, @Kbeam1,      C Number of pump beam
C Zstep, @Zstep,      C Step length
C Kappa, @Kappa      C Irradiance gain coefficient
C Wavelength, @Lambda1      C Wavelength of pump
C Detuning factor, DeltaK: @DeltaK      C Detuning factor
C DetuningRatio, @DetuningRatio:      C DetuningRatio
C Coef2_Kappa, @Coef2_Kappa      C Kappa-Length-detuning factor
C CenterAmplitude, @CenterAmplitude      C Starting amplitude
C Amplification_P, @Amplification_P      C Amplification with Kappa
C AmplitudeFinal, @AmplitudeFinal      C final amplitude
C -----

plot/watch ex49_3.plt
title depleted pump, detuned, closed form
plot/x/i 1 fmax=1.e7 fmin=0.
c
c harmonic amplification with detuning
c
variab/set Line line; Line=

```

Jump to: [Commands](#), [Theory](#)

```

double/closed/harmonic Kbeam2 zstep=Zstep kappa=Kappa deltak=DeltaK list
Fac = 0.; Detuning = 0.
if [DeltaK>0.] then
    Fac = 0.5*DeltaK*Zstep
    DetuningRatio = sin(Fac)/Fac          # Detuning factor
endif
Coef2_Kappa = Kappa*Zstep*DetuningRatio
Amplification_H = Cfac*tanh(Coef2_Kappa*CenterAmplitude)
AmplitudeFinal = Amplification_H*CenterAmplitude
C
C -----
C Kbeam2, @Kbeam2,      C Number of harmonic beam
C Fac, @Fac  C Intermedate variable
C Detuning, @Detuning    C Detuning factor
C Zstep, @Zstep,      C Step length
C Kappa, @Kappa      C Kappa
C Wavelength, @Lambda2    C Wavelength of harmonic
C Detuning factor, DeltaK: @DeltaK    C Detuning factor
C DetuningRatio, @DetuningRatio:    C DetuningRatio
C Coef2_Kappa, @Coef2_Kappa    C Kappa-Length-detuning factor
C CenterAmplitude, @CenterAmplitude    C Starting amplitude
C Amplification_H, @Amplification_H    C Amplification with Kappa
C AmplitudeFinal, @AmplitudeFinal    C final amplitude
C -----
C Energy Conservation Check
PplusTot = PAmplitudeFinal*conj(PAmplitudeFinal)
PplusTot = PplusTot + HAmplitudeFinal*conj(HAmplitudeFinal)
C C CenterIntensity = @CenterIntensity, PplusTot = @PplusTot
C -----
plot/watch ex49_4.plt
title second harmonic, detuned, closed form
plot/x/i 2 fmax=1.e7 fmin=0.
end

```


Ex50: Out-coupled polarization for TE and TM waveguide modes

This example illustrates qualitatively the type of polarization produced by the TE and TM modes when out-coupled by a focusing grating. TE produces a constant linear polarization in the focused beam. TM, because it is elliptical in the waveguide, produces elliptical polarization in the out-coupled beam depending on the projection of the guided TM polarization and coupling of the two TM electric vector components.

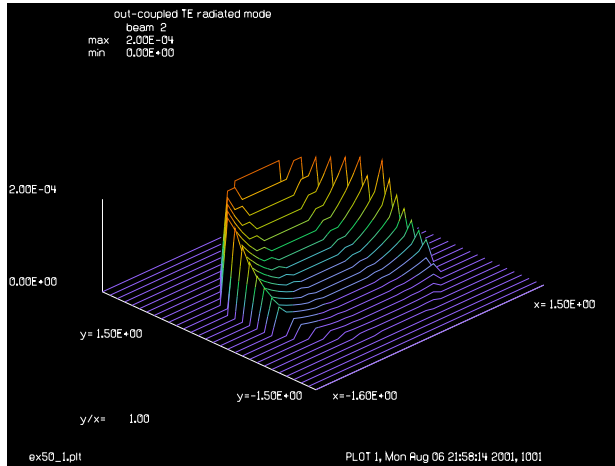


Fig. 50.1. Out-coupled intensity from TE.

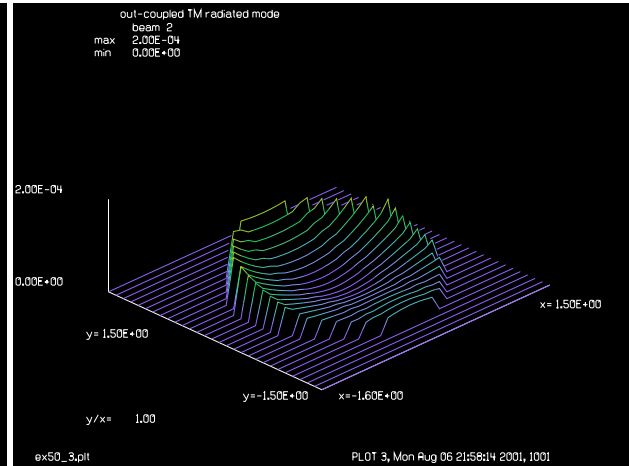


Fig. 50.2. Out-coupled intensity from TM.

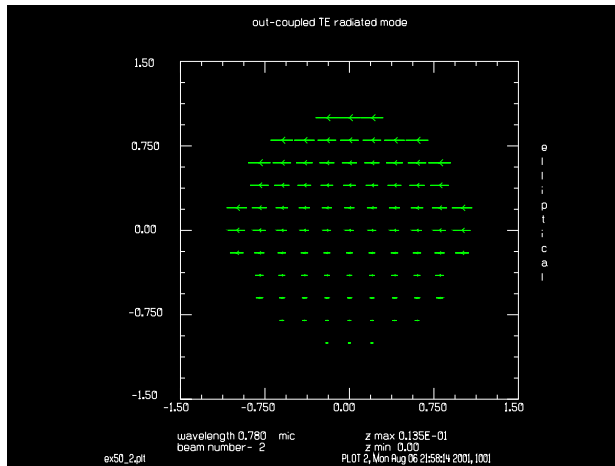


Fig. 50.3. Pupil polarization from TE.

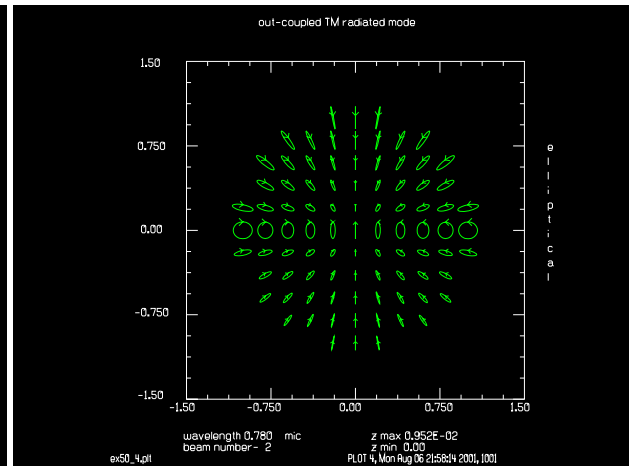


Fig. 50.4. Pupil polarization from TM.

Input: `ex50.inp`

```
c## ex50
c
c Example 50: Waveguide Grating Coupler: TE and TM Polarizations
c
c This example illustrates outcoupling of a waveguide grating
c coupler for both TE and TM polarization states of the guided
c mode. The TE mode produces a focusing beam with uniform
c linear polarization and with exponential decay. The TM
```

```

c  guided mode shows pronounced elliptical polarization in the
c  exit pupil of the focused beam.  It shows a departure from
c  simple exponential decay as well with the center being less
c  efficient in outcoupling than the edges of the grating aperture.
c
echo/on
nbeam 2                                # form the guided and radiated beams
array/set 1 32 1 1                     # guided mode is 32 X 1
array/set 2 32 32 1                   # radiated mode is 32 X 32
c
c  Set up for TE mode
c
set/density 13 13
clear 1 1
clear 2 0
units/s 1 .1 .0001                    # width of guided beam is 1 micron
units/s 2 .1 .1                       # units of radiated mode
wavelength/set 0 .78                  # set wavelength to .78 micron
field 1
slab/hoe/cir 1.05 0. 0. 1.            # define circular aperture to be 1.05 radius
slab/hoe/ecoef 1.                     # set TE coupling constants to 1 cm-1
slab/out 1 2
intmap 1
intmap 2
title out-coupled TE radiated mode
plot/watch ex50_1.plt
plot/l 2 max=2.e-4 min=0.
plot/watch ex50_2.plt
plot/ell 2
c
c  Set up for TM mode
clear 1 1
c
c  Model two components of TM, which are really z and y, as y and z
c  in the one-dimensional beam.
c
clear 2 0
c
c  Set in-plane and out-of-plane components of TM
c
jones/set ar=.979795897 ci=.2 dr=0.
jones/mult 1
field 1
slab/hoe/mcoef 1. 1.                  # set coupling constants to 1 cm-1
slab/out 1 2
intmap 1
intmap 2
title out-coupled TM radiated mode
plot/watch ex50_3.plt
plot/l 2 max=2.e-4 min=0.
plot/watch ex50_4.plt
plot/ell 2
energy
end

```

Jump to: [Commands](#), [Theory](#)

Ex51: Calculation of TE and TM outcoupling with the Induced Dipole Model

Table. 51.1. Table of Ex51 examples

Ex51a: Waveguide grating incoupling, TE	1
Ex51b: Waveguide grating incoupling, TM	6

The induced dipole model is very simple but gives a clear idea of the type of polarization to be expected. It is accurate up to modest numerical apertures — perhaps as large as 0.5. A generalized dipole model which includes the divergence of the dipoles provides shows some small polarization rotations and slight changes in the elliptical polarization at high numerical apertures. The concepts are explained in Chap. 15, GLAD Theory Manual.

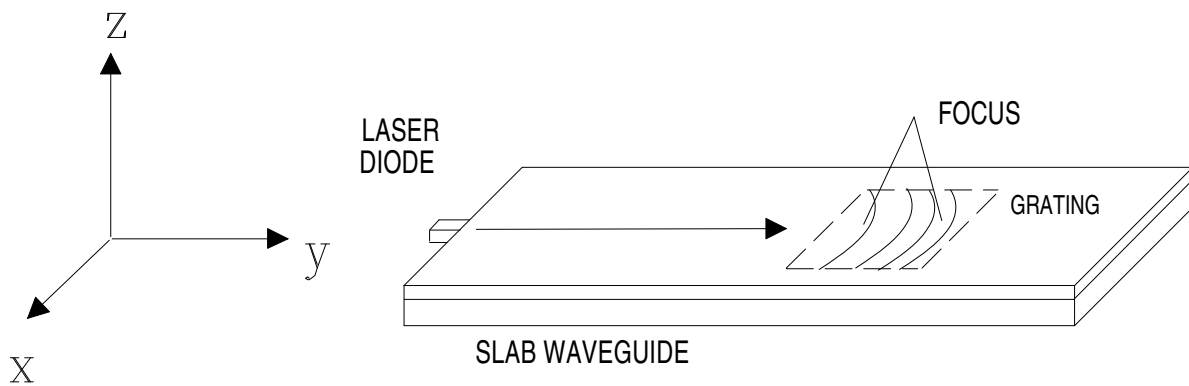


Fig. 51.1. Slab waveguide consisting of a planar substrate surface with a thin, high index layer on the top. The light will propagate as a guided wave when considered in a direction parallel to the surface normal and according to free-space propagation in the direction parallel to the surface.

Ex51a: Waveguide grating incoupling, TE

Input: [ex51a.inp](#)

```
c## ex51a!882614074019883
c
c Example 51a: Waveguide grating incoupling, TE
c
echo/on
nbeam 2                                # form the guided and radiated beams
array/set 1 32 1 1                      # guided mode is 32 X 1
array/set 2 32 32 1                    # radiated mode is 32 X 32
c
c Set up for TE mode
c
set/density 13 13
clear 1 1
clear 2 0
units/s 1 .1 .0001                    # width of guided beam is 1 micron
```

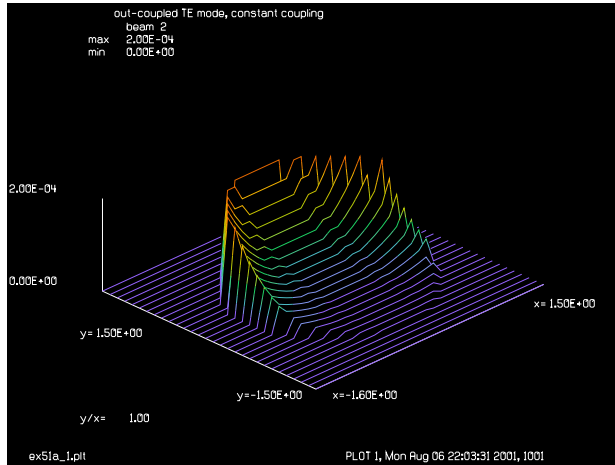


Fig. 51.2a. Isometric plot of TE radiation output. The guided mode is propagating down from the top. The pupil exhibits simple exponential decay. The grating is circular, causing the outer edges to begin the exponential decay later than the center.

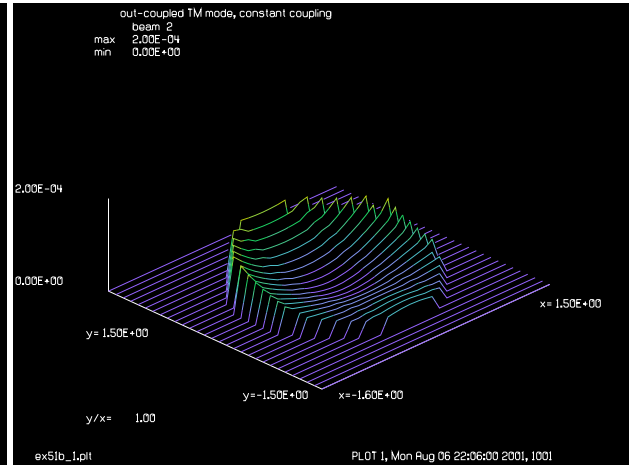


Fig. 51.2b. Isometric plot of TM radiation showing pronounced departure from the exponential decay. The outcoupling is much lower in the center because the vertical component of TM does not contribute.

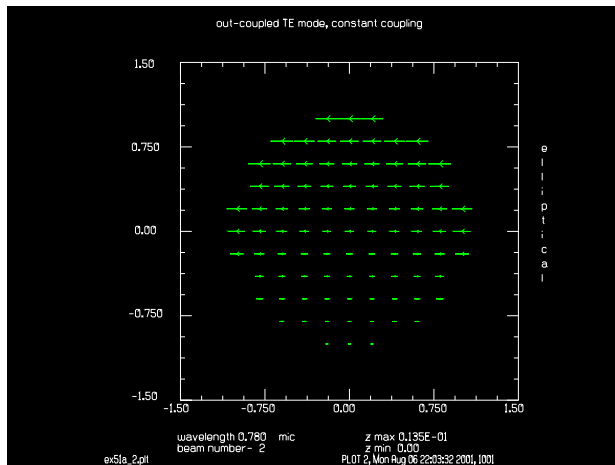


Fig. 51.3a. Polarization plot of outcoupled TE radiation. The polarization is uniform linear polarization. The slight increase in amplitude in the outer parts of the aperture is due to the effect of the circular grating aperture.

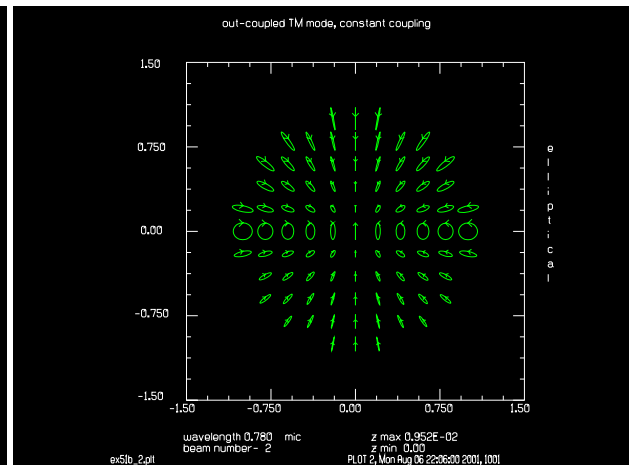


Fig. 51.3b. The polarization varies across the pupil of a focusing grating because the elliptical dipoles are viewed from different directions as seen from the focus point.

```
units/s 2 .1 .1          # units of radiated mode
wavelength/set 0 .78     # set wavelength to .78 micron
field 1
slab/hoe/cir 1.05 0. 0. 1. # define circular aperture to be 1.05 radius
slab/hoe/ecoef 1.         # set TE coupling constants to 1 cm-1
slab/out 1 2
intmap 1
intmap 2
title out-coupled TE mode, constant coupling
plot/watch ex51a_1.plt
```

Jump to: [Commands](#), [Theory](#)

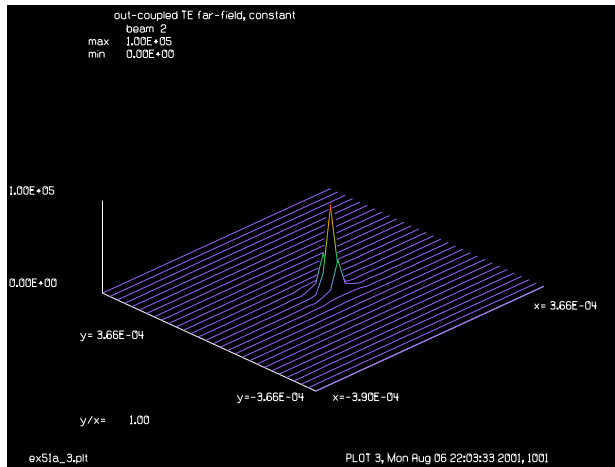


Fig. 51.4a. Far-field intensity of outcoupled TE radiation. The distribution is fairly close to an Airy pattern. The Strehl ratio is 1.00.

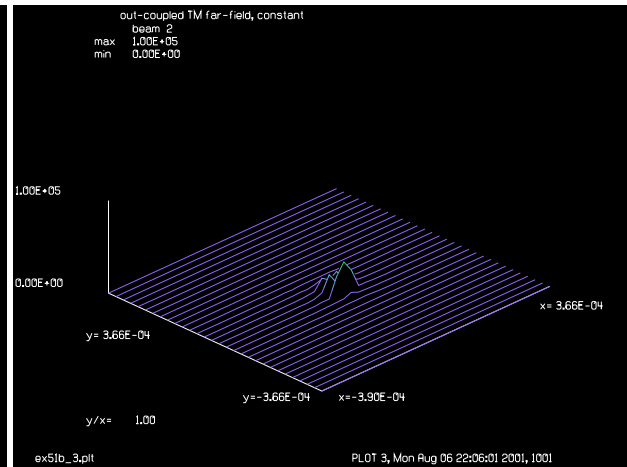


Fig. 51.4b. Far-field intensity of outcoupled TM radiation. The Strehl ratio is 0.1 because of the polarization variation across the pupil.

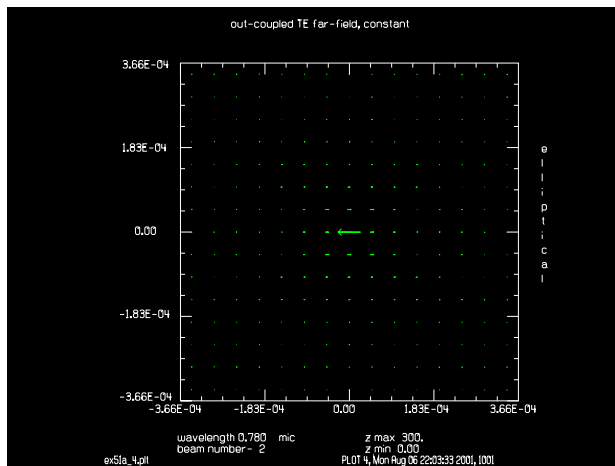


Fig. 51.5a. Far-field polarization map of outcoupled TE radiation.

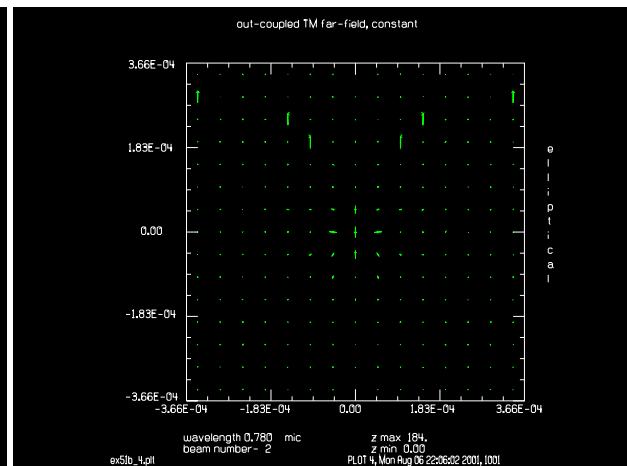


Fig. 51.5b. Far-field polarization map of outcoupled TM radiation. The central lobe shows orthogonal polarization to the central lobe of the outcoupled far-field TE plot. The Strehl ratio is less and information on the pupil polarization variation is contained in the secondary lobes.

```
plot/l 2 max=2.e-4 min=0.
plot/watch ex51a_2.plt
plot/ell 2
strehl 2
lens 2 1
dist 1 2
mirror/flat 2
peak 2
title out-coupled TE far-field, constant
plot/watch ex51a_3.plt
plot/l 2 max=1e5
plot/watch ex51a_4.plt
```

Jump to: [Commands](#), [Theory](#)

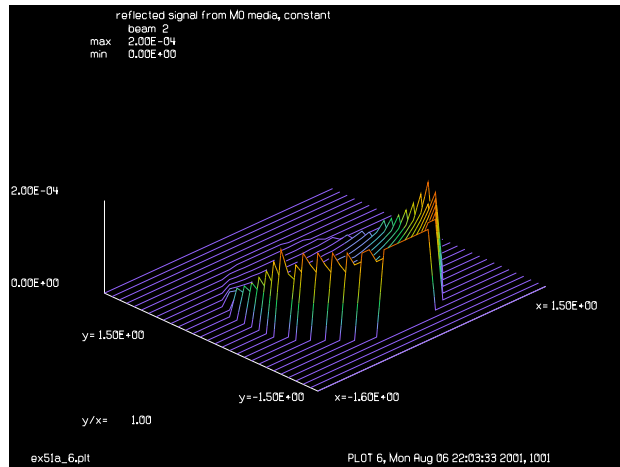


Fig. 51.6a. Returned signal prior to incoupling into the grating.

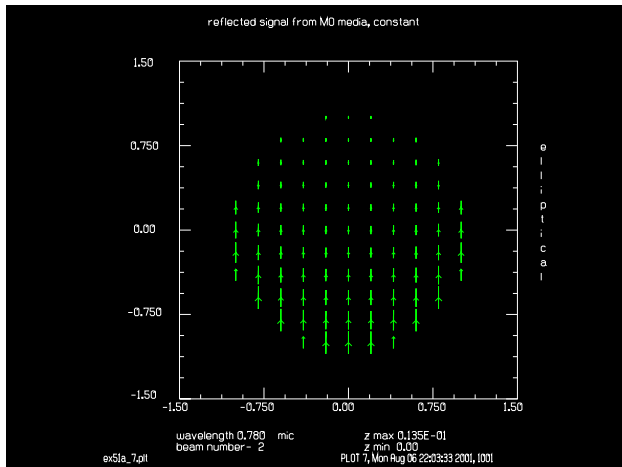


Fig. 51.6b. Polarization after return from MO medium.

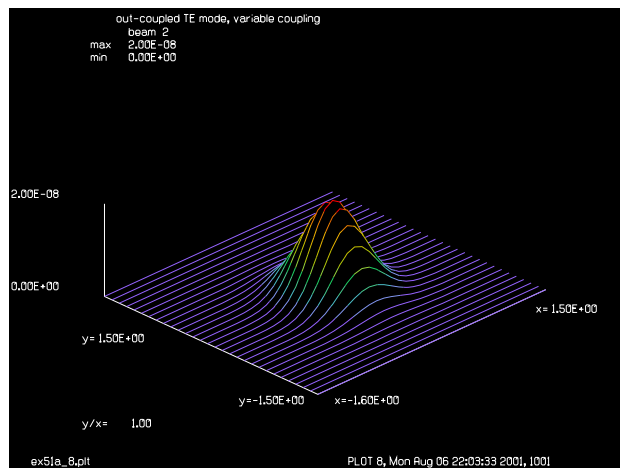


Fig. 51.7a. Outcoupled TE mode with apodized pupil, intensity.

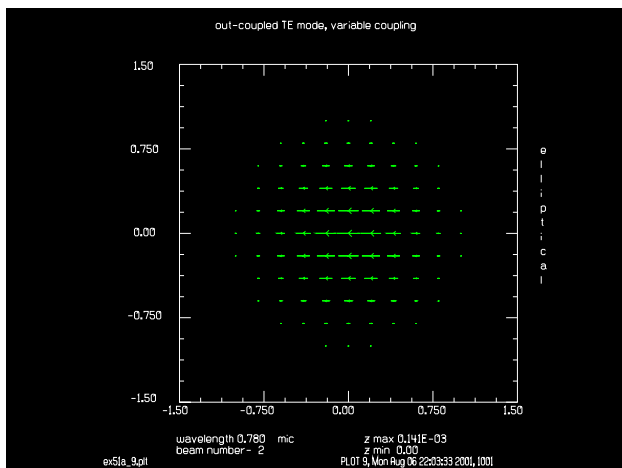


Fig. 51.7b. Outcoupled TE mode with apodized pupil, polarization.

```

plot/ell 2
jones/set ar=0 br=1 cr=1 dr=0
jones/multiply 2
title far-field, after MO media, constant
plot/watch ex51a_5.plt
plot/ell 2
dist -1 2
lens 2 1
title reflected signal from MO media, constant
plot/watch ex51a_6.plt
plot/l 2 max=2.e-4 min=0.
plot/watch ex51a_7.plt
plot/ell 2
clear 1 1
clear 2 0
units/s 1 .1 .0001          # width of guided beam is 1 micron

```

Jump to: [Commands](#), [Theory](#)

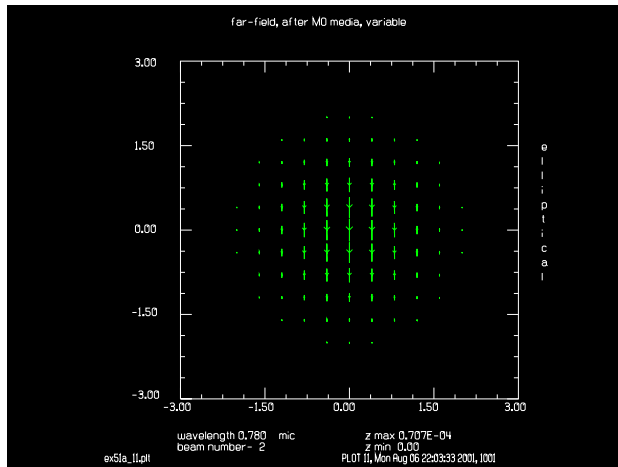


Fig. 51.8. Reflected signal from MO media is identical to the outcoupled TE mode (Fig. 51.9), except rotated 90°.

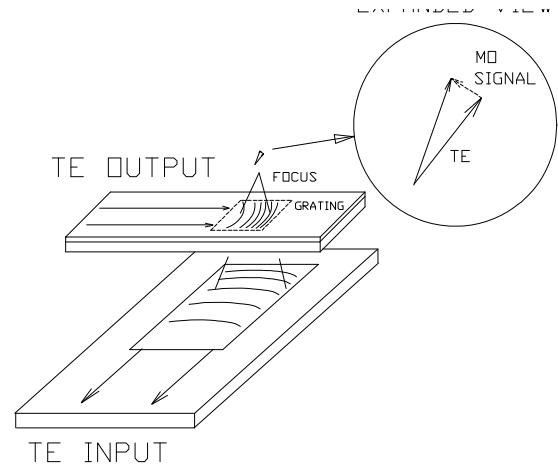


Fig. 51.9. Elementary crossed grating concept to detect signal from MO medium. In practice we would use two gratings in the detection leg rotated 90 degrees apart to do differential phase detection.

```

units/s 2 .1 .1                # units of radiated mode
slab/hoe/ecoef .0001
slab/out 1 2
split/cir/in 2 .6
intmap 1
intmap 2
title out-coupled TE mode, variable coupling
plot/watch ex51a_8.plt
plot/l 2 min=0.
plot/watch ex51a_9.plt
plot/ell 2
lens 2 1
dist 1 2
mirror/flat 2
title out-coupled TE far-field, variable
plot/watch ex51a_10.plt
plot/ell 2
jones/set ar=0 br=1 cr=1 dr=0
jones/multiply 2
title far-field, after MO media, variable
plot/watch ex51a_11.plt
plot/ell 2
dist -1 2
lens 2 1
title reflected signal from MO media, variable
plot/watch ex51a_12.plt
plot/l 2 min=0.
plot/watch ex51a_13.plt
plot/ell 2
end

```

Ex51b: Waveguide grating incoupling, TM**Input: ex51b.inp**

```

c## ex51b!636718095684801
c
c  Example 51b: Waveguide grating incoupling, TM
c
echo/on
nbeam 2                                # form the guided and radiated beams
array/set 1 32 1 1                     # guided mode is 32 X 1
array/set 2 32 32 1                   # radiated mode is 32 X 32
c
c  Set up for TM mode
c
set/density 13 13
clear 1 1
clear 2 0
units/s 1 .1 .0001                    # width of guided beam is 1 micron
units/s 2 .1 .1                       # units of radiated mode
wavelength/set 0 .78                  # set wavelength to .78 micron
field 1
jones/set ar=.979795897 dr=0 ci=.2
jones/multiply 1
field
pause
slab/hoe/cir 1.05 0. 0. 1.            # define circular aperture to be 1.05 radius
slab/hoe/mcoef 1. 1.                  # set TM coupling constants to 1 and 1 cm-1
slab/out 1 2
intmap 1
intmap 2
title out-coupled TM mode, constant coupling
plot/watch ex51b_1.plt
plot/l 2 max=2.e-4 min=0.
plot/watch ex51b_2.plt
plot/ell 2
strehl 2
pause
lens 2 1
dist 1 2
mirror/flat 2
peak 2
title out-coupled TM far-field, constant
plot/watch ex51b_3.plt
plot/l 2 max=1e5
plot/watch ex51b_4.plt
plot/ell 2
end

```

Ex52: Cone (axicon) aberration

A type of cone aberration may be generated by conical elements such as axicons or waxicons. The cone aberration produces a well-defined ring structure and small peak in the center of far-field with a center structure. In this example, GLAD is used to calculate the far-field and rough estimates of the expected intensity are compared with the calculated values. The parameters of the problem are listed in Table 52.1.

Table. 52.1. Principle terms.

aperture radius	20 cm
cone aberration	4 waves at aperture edge
wavelength	10.6 micron

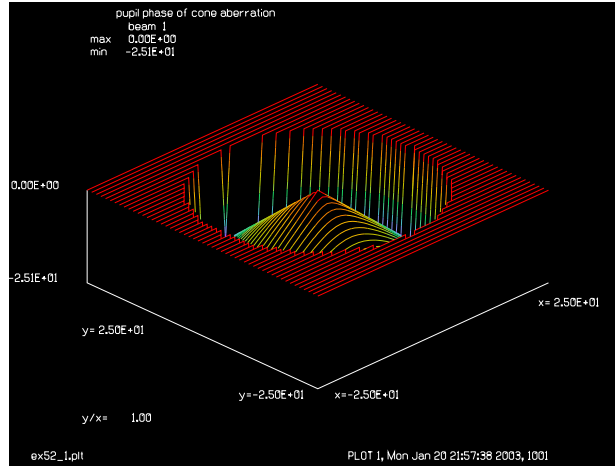


Fig. 52.1. Isometric plot of cone aberration.

The wavefront aberration varies according to,

$$W = 2\pi W_{cone} \frac{r}{a} \quad (52.1)$$

where W is the wavefront error (in radians), W_{cone} is the cone aberration coefficient (in waves), r is the aperture variable, and a is the aperture radius.

The far-field illustrated in Fig. 52.1 and Fig. 52.2 shows a ring structure with a small central peak. The ring structure can be considered to derived from the diffraction width of a line function of length which has been rotated about a radius in the image. The ring function takes the approximate form,

$$2\pi \frac{\lambda^2 f^2}{a^2} \text{sinc}^2 \left[\frac{\pi}{\lambda f} (ar' - r'_0) \right] \quad (52.2)$$

where r'_0 is the rotation radius and f is the focal length of the lens.

$$r'_0 = W_{coma} \frac{\lambda f}{a} \quad (52.3)$$

The intensity of the sinc^2 function was determined by taking the approximate width of the ring to be $\frac{\lambda f}{a}$, such that the area of the ring is

$$\text{ring area} = 2\pi\Delta r' r'_0 = 2\pi\left(\frac{W_{coma}f}{a}\right)\frac{\lambda f}{a} = 1.70 \times 10^6 \quad (52.4)$$

in good agreement with the calculated value of 1.60×10^6 . The peak intensity for the center pattern may also be calculated. Using the integral relationship

$$I(0, 0) = \frac{1}{\lambda^2 f^2} \left| \int_0^1 2\pi r W_{coma} \frac{r}{a} dr \right|^2 \quad (52.5)$$

$$\int_0^a r e^{br} dr = \frac{e^{br}}{b^2} (br - 1) \Big|_0^a = \frac{e^{ba}}{b^2} (ba - 1) + \frac{1}{b^2} \quad (52.6)$$

In the special case where W_{coma} is an integer value, the peak intensity takes the form

$$I(0, 0) = \frac{1}{\lambda^2 f^2} \left| \frac{a^2}{iW_{coma}} \right|^2 = 8.9 \times 10^5 \quad (52.7)$$

in good agreement with the calculated value of 8.73×10^5 .

For strong cone aberration — many wavelengths — the far-field will be well approximated by a rotated sinc^2 function, except for some fine structure in the center of the far-field.

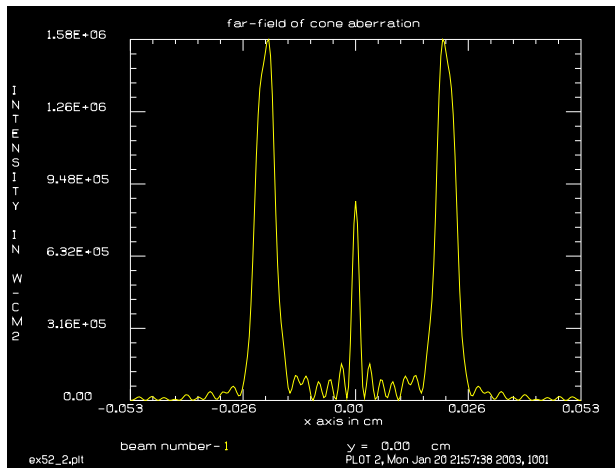


Fig. 52.2. Profile of the far-field of cone aberration after focusing by a lens.

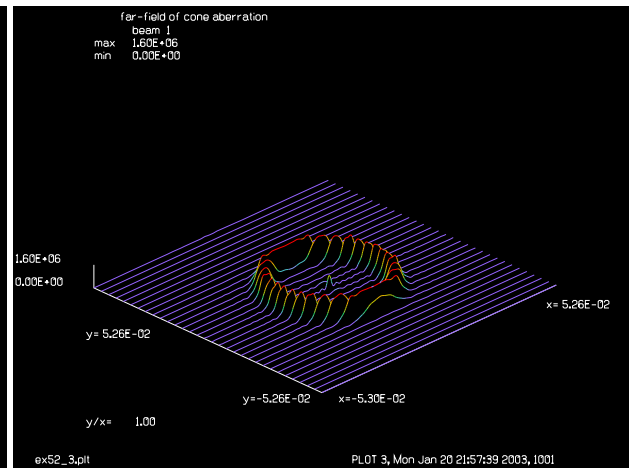


Fig. 52.3. Isometric plot of intensity of far-field illustrating the strong ring function and small central peak.

Input: ex52.inp

```

c## ex52
c
c Example 52: Cone Phase Function
c
c The far-field of the cone function exhibits to structures.
c A ring is formed with radius equal to the amount of cone
c considered as tilt aberration. The center structure arises
c from the edge diffraction forming a zero order Bessel function.
c
array/s 1 256                # use large array
units/s 1 1
clap/c/c 1 20                # 20 cm radius aperture
energy
abr/cone 1 4                 # four waves of cone aberration
title pupil phase of cone aberration
plot/watch ex52_1.plt
plot/l/ph 1 xrad=25 ns=128
lens 1 100                   # lens, f=100 cm
dist 100
title far-field of cone aberration
plot/watch ex52_2.plt
plot/x/i 1                   # plot profile
plot/watch ex52_3.plt
plot/l 1 h=.1                # isometric plot
peak 1
intens
end

```


Ex53: Hermite-gaussian functions

Table. 53.1. Table of Ex53 examples

Ex53a: Some Hermite-Gaussian polynomials	1
Ex53b: Construction of donut mode from HG(1,0) and HG(0,1), orthogonal polarizations. . .	4

The transverse modes of ideal stable resonators take the form of Hermite-gaussian polynomials or Laguerre-gaussian. The general polynomial form of the Hermite-gaussian functions is

$$u_n(x) = \left(\frac{2}{\pi}\right)^{1/4} \left(\frac{1}{2^n n! \omega_0}\right)^{1/2} H_n\left(\frac{\sqrt{2}x}{\omega(z)}\right) \exp\left[-\frac{x^2}{\omega(z)^2}\right] \quad (53.1)$$

where n is the order of the polynomial, ω_0 is a waist radius parameter similar to the gaussian beam and $H_n(x)$ are the Hermite functions. The two-dimensional functions may be described by multiplying two one-dimensional functions. The order and waist parameters may be different for the two directions.

Example 53b illustrates construction of a donut mode consisting of equal proportions of HG(1,0) and HG(0,1) with orthogonal polarizations.

Ex53a: Some Hermite-Gaussian polynomials

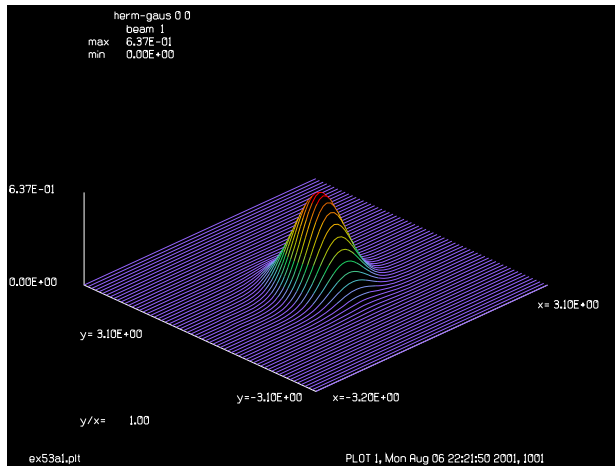


Fig. 53.1a. HG 0,0.

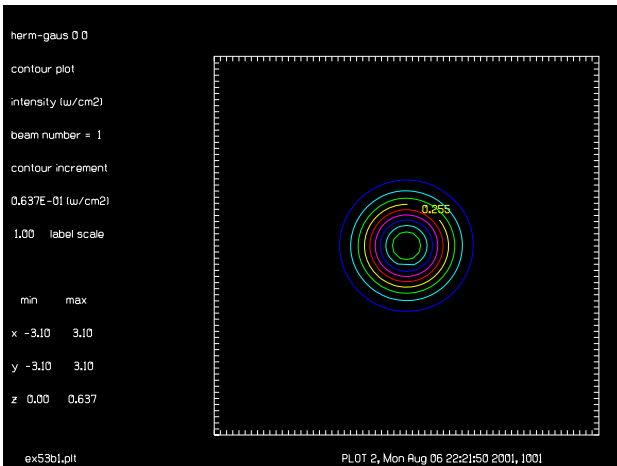


Fig. 53.1b. HG 0,0.

Input: `ex53a.inp`

```
c## ex53a!143825927762552
c
c Example 53a: Some Hermite-Gaussian Polynomials
c
echo/on
variab/dec/int max_order x_order y_order plot_num
macro/def loop2/o
  y_order = y_order + 1
```

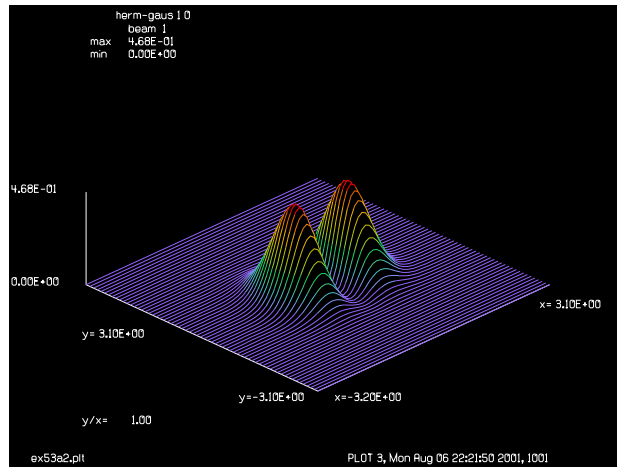


Fig. 53.2a. HG 1,0.

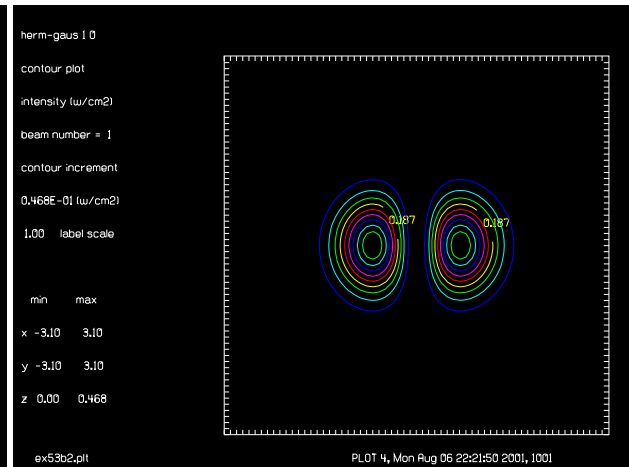


Fig. 53.2b. HG 1,0.

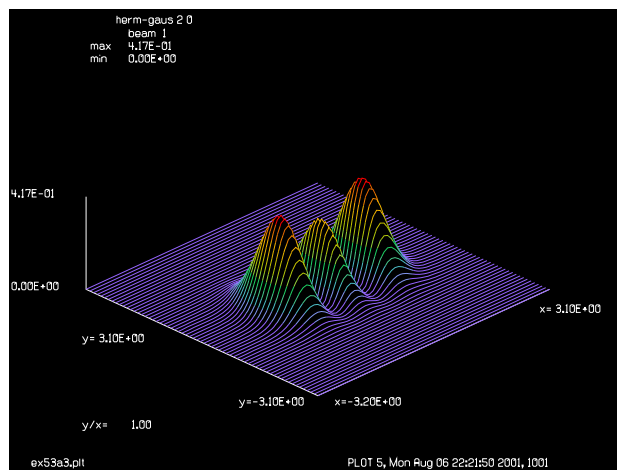


Fig. 53.3a. HG 2,0.

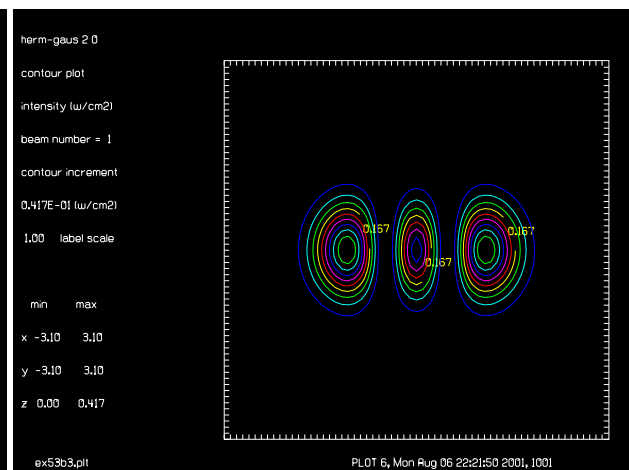


Fig. 53.3b. HG 2,0.

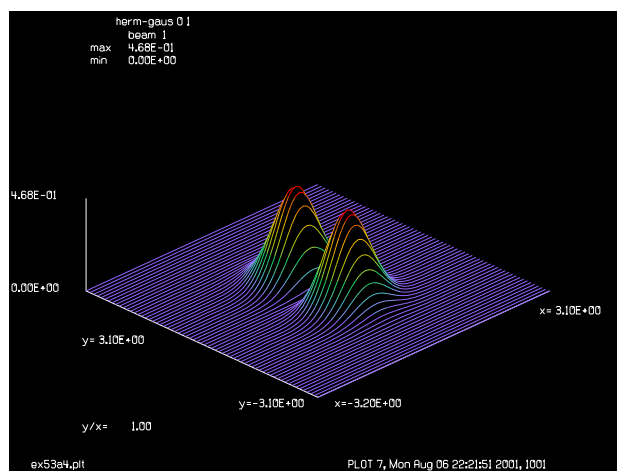


Fig. 53.4a. HG 0,1.

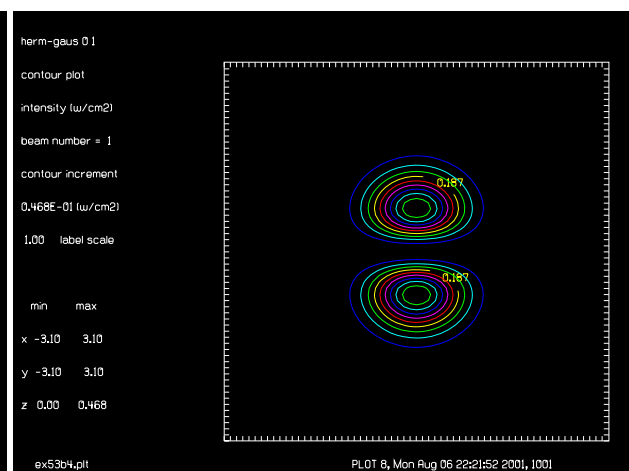


Fig. 53.4b. HG 0,1.

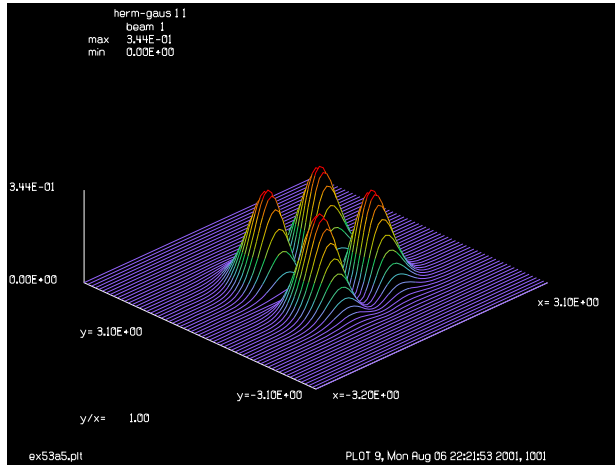


Fig. 53.5a. HG 1,1.

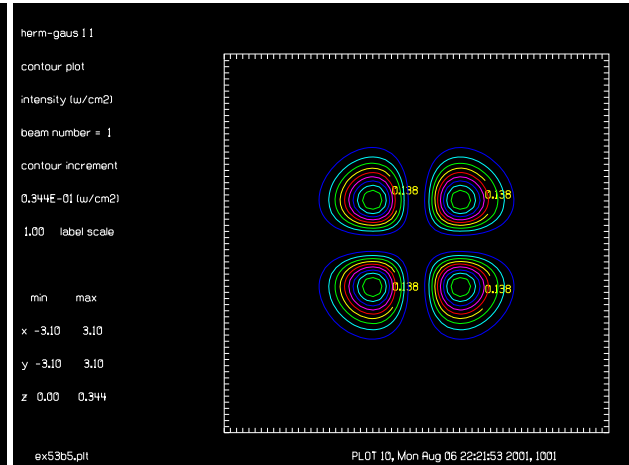


Fig. 53.5b. HG 1,1.

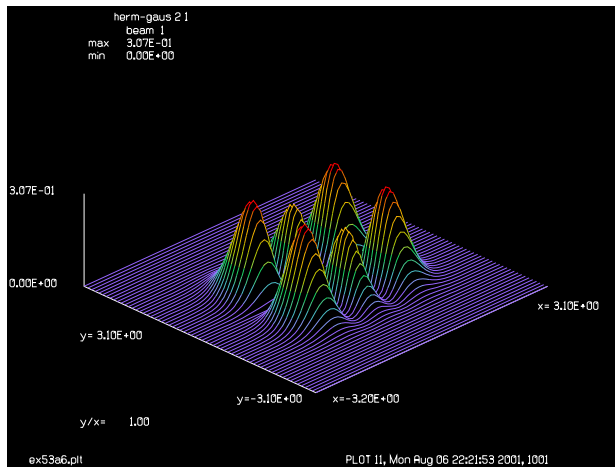


Fig. 53.6a. HG 2,1.

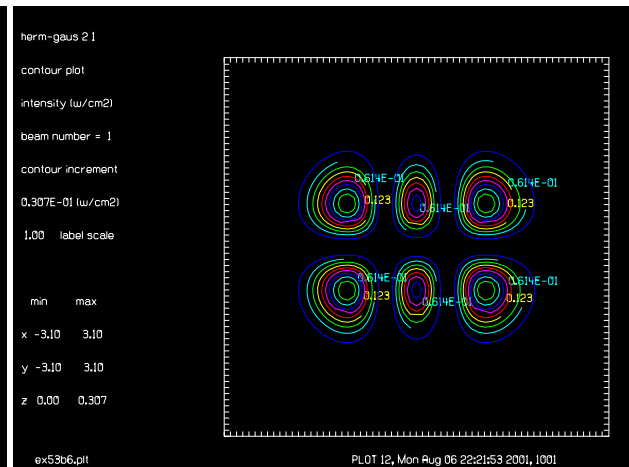


Fig. 53.6b. HG 2,1.

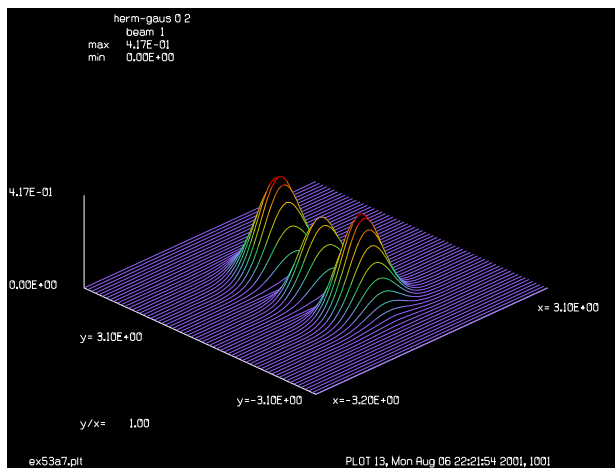


Fig. 53.7a. HG 0,2.

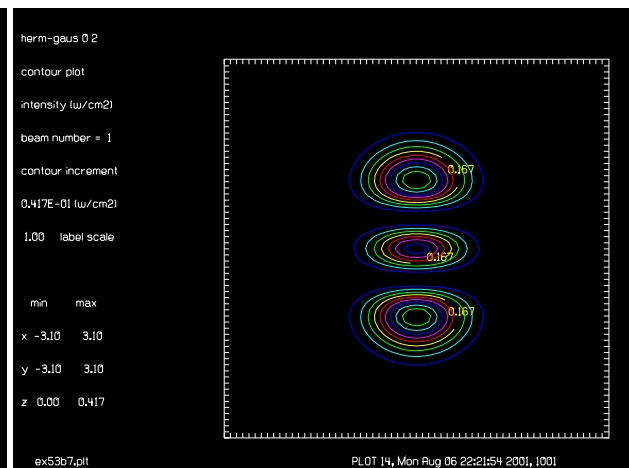


Fig. 53.7b. HG 0,2.

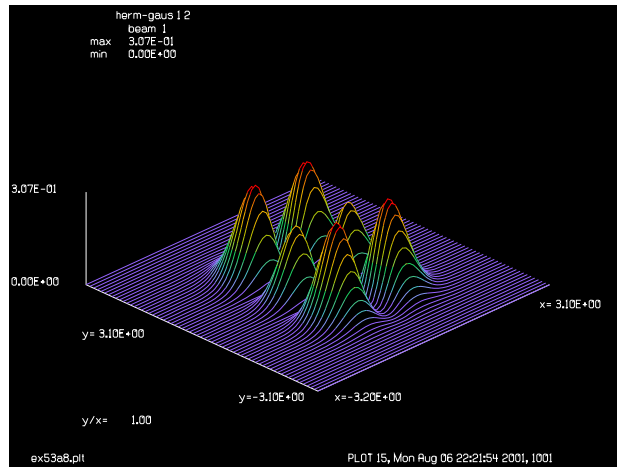


Fig. 53.8a. HG 0,2.

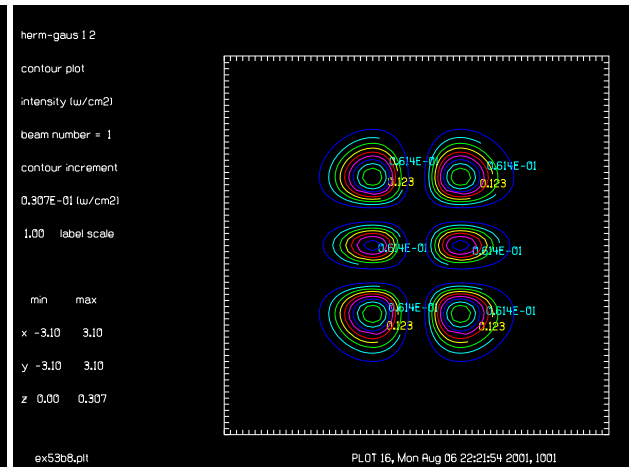


Fig. 53.8b. HG 0,2.

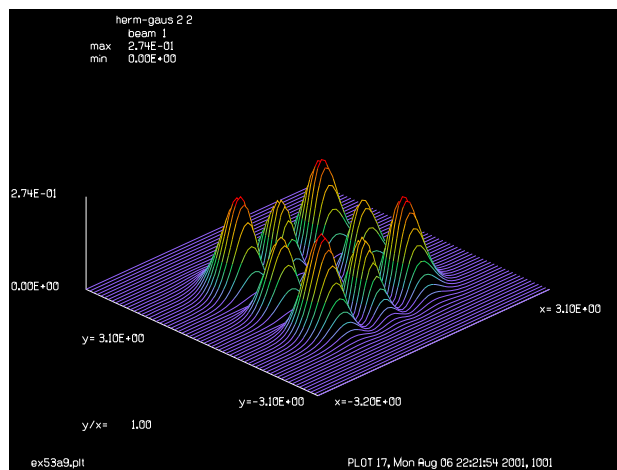


Fig. 53.9a. HG 2,2.

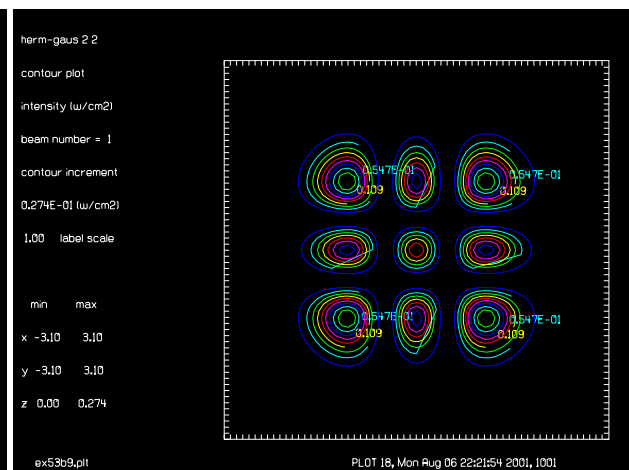


Fig. 53.9b. HG 2,2.

```

x_order = -1
macro loop1/max_order
macro/end
macro/def loop1/o
  x_order = x_order + 1
  hermite/con 1 1 1 1 x_order y_order
  plot_num = plot_num + 1
  title herm-gaus @x_order @y_order
  plot/watch ex53a@plot_num.plt
  plot/l ns=64
  plot/watch ex53b@plot_num.plt
  plot/con
macro/end
units/s 1 .1
y_order = -1
max_order = 3
plot_num = 0
macro loop2/max_order

```

set maximum order number

Jump to: [Commands](#), [Theory](#)

end

Ex53b: Construction of donut mode from HG(1,0) and HG(0,1), orthogonal polarizations.

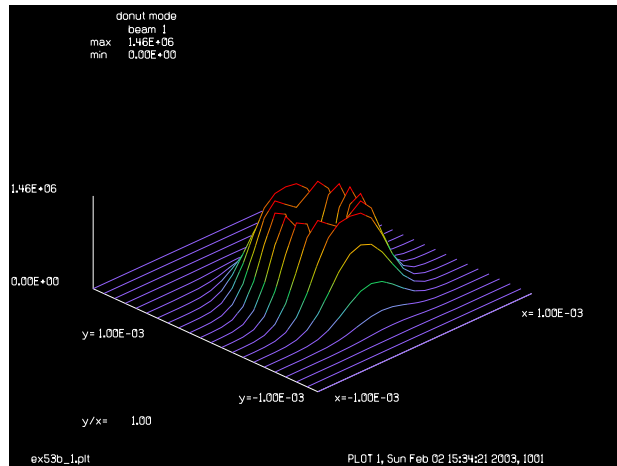


Fig. 53.10. HG 1,0 and HG 0,1, isometric.

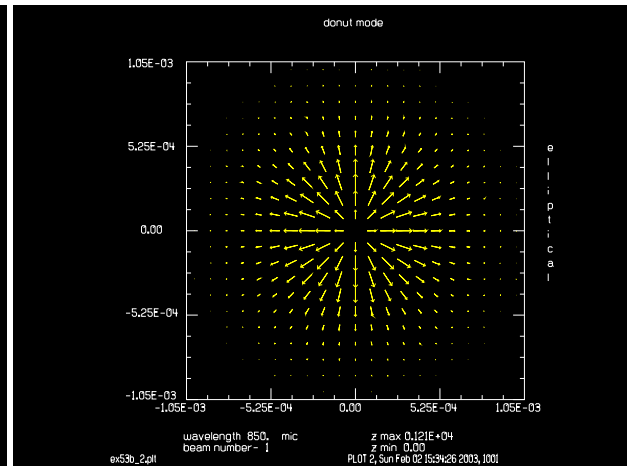


Fig. 53.11. HG 1,0 and HG 0,1, polarization.

Input: ex53b.inp

```
c## ex53b!571070855974324
#
# Example 53a: Donut mode
#
# This example illustrates formation of the "donut" mode consisting
# of equal amounts of HG(1,0) and HG(0,1) and opposite linear
# polarization for the two modes.
#
# Although the donut mode is perfectly constructed, there is zero intensity
# at the center point of the far-field, so the Strehl ratio is zero
# and the calculation is correct.
#
# Interestingly, application of aberration to the donut mode and
# similar HG modes will cause the Strehl ratio to rise from zero --
# its optimum value.
#
lambda=850
nline=128
field=64e-4
waist=4e-4
array/s 1 nline nline 1          # beam 1, propagating beam
nbeam 2 nline nline 1
units/field 0 field              # beam 1 field half-width of 256 microns
wavelength/set 1 lambda 1        # set wavelength and refractive index of the mold
hermite/rec/c 1 .5 waist waist 1 0 # generate hermit-gaussian (1,0)
```

Jump to: [Commands](#), [Theory](#)

```
hermite/rec/c 2 .5 waist waist 0 1      # generate hermit-gaussian (0,1)
jones/set ar=0. bi=0 cr=1. ci=0 dr=0     # define vertical plarization
jones/mult 2                             # apply vertical polarization to beam 1
add/c 1 2                                 # combine two beam into beam 1
title donut mode
plot/watch ex53b_1.plt
plot/l 1 1 xrad=1e-3 ns=128
pause 5
set/win/abs -1e-3 1e-3 -1e-3 1e-3
#
# Display polarization to verify correct donut mode
#
plot/watch ex53b_2.plt
plot/ell
echo/on
#
# Note that Strehl ratio is zero for donut mode, as it is for
# HG(1,0) and HG(0,1) considered separately.
#
strehl 1
pause 5
end
```


Ex54: Laguerre functions

The Laguerre-gaussian functions are described in cylindrical coordinates.

$$u_{pm}(r, \theta) = \sqrt{\frac{2p!}{(1 + \delta_{0m})\pi(m+p)!}} \frac{e^{j(2p+m+1)}}{\omega_0} \left(\frac{\sqrt{2}r}{\omega(z)}\right)^m L_p^m\left(\frac{2r^2}{\omega_0^2}\right) e^{-\frac{r^2}{\omega_0^2}} e^{jm\theta} \quad (54.1)$$

where θ is the azimuthal variation, p is the radial order, m is the azimuthal order, δ_{0m} , takes the value 1 if $m = 0$ and zero otherwise, and L_p^m is the Laguerre polynomial.

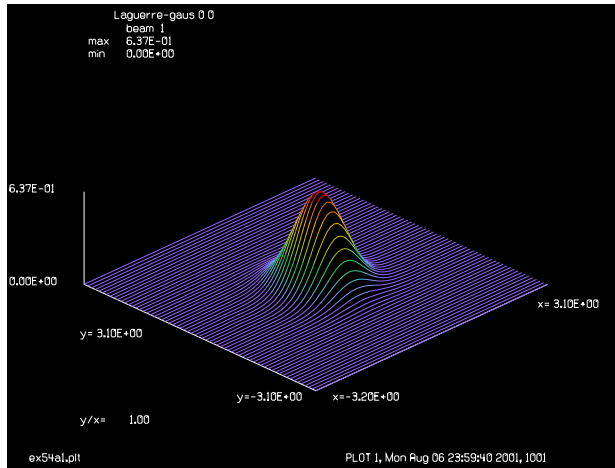


Fig. 54.1. LG 0,0.

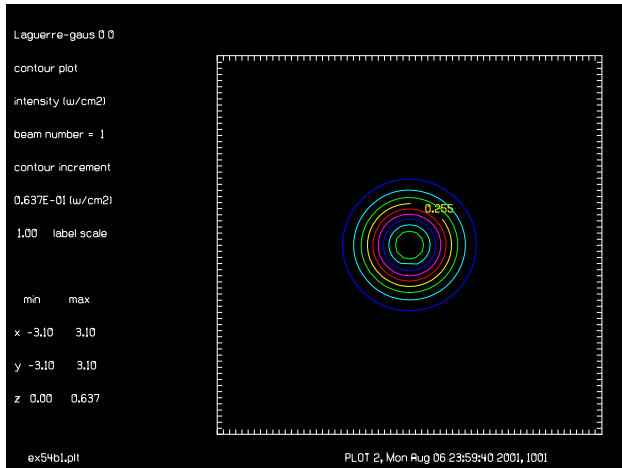


Fig. 54.2. LG 0,0.

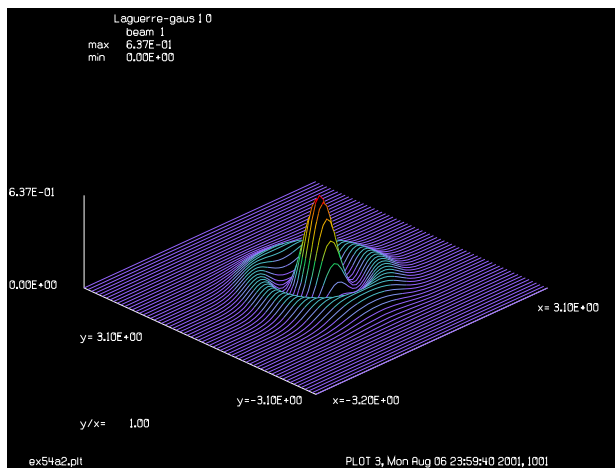


Fig. 54.3. LG 1,0.

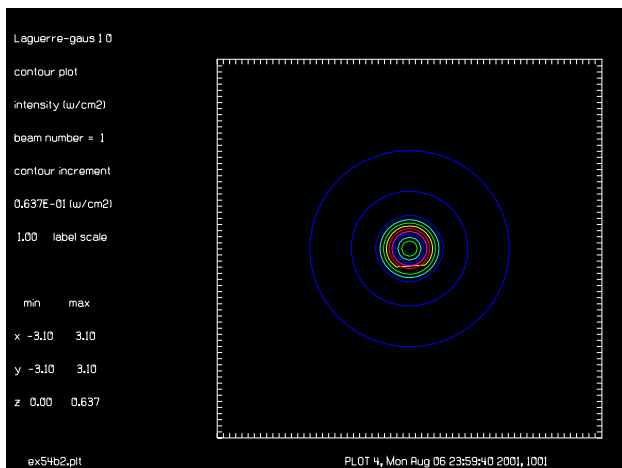


Fig. 54.4. LG 1,0.

Input: `ex54.inp`

c## ex54!361026644174045

c

Jump to: [Commands](#), [Theory](#)

Ex54.1

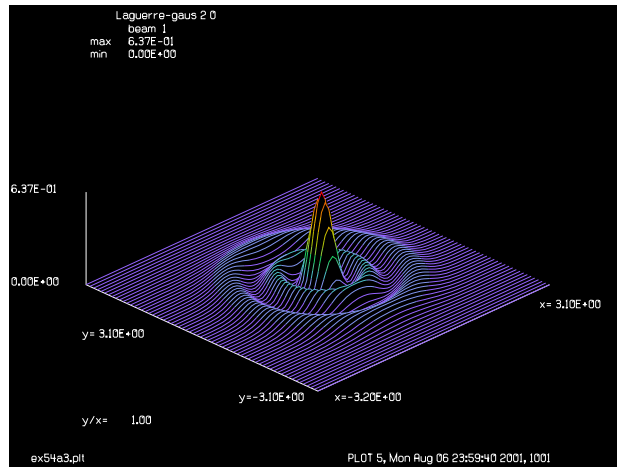


Fig. 54.5. LG 2,0.

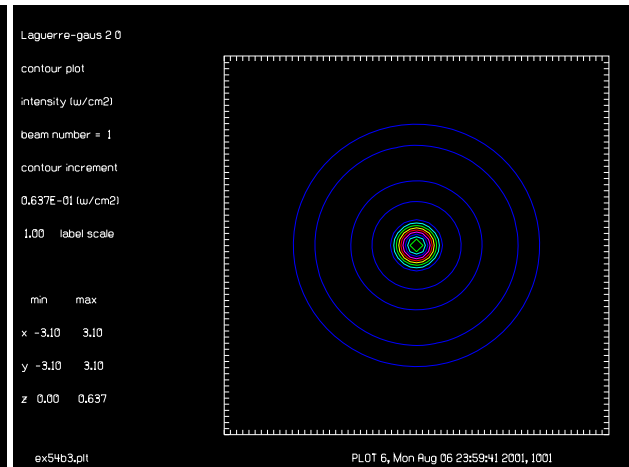


Fig. 54.6. LG 2,0.

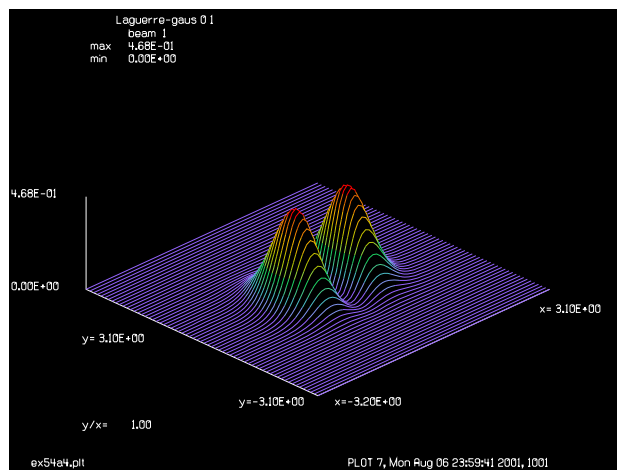


Fig. 54.7. LG 0,1.

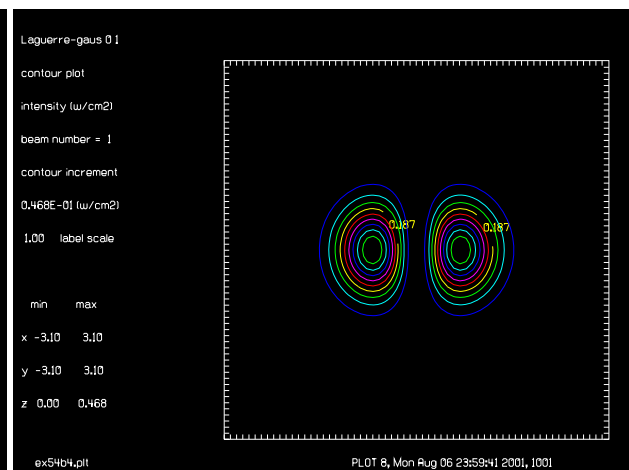


Fig. 54.8. LG 0,1.

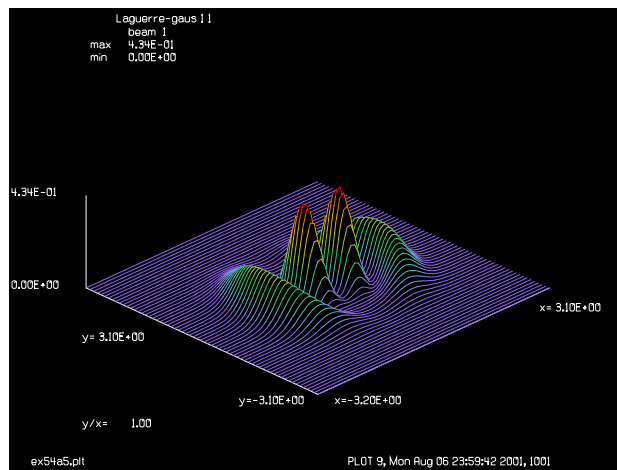


Fig. 54.9. LG 1,1.

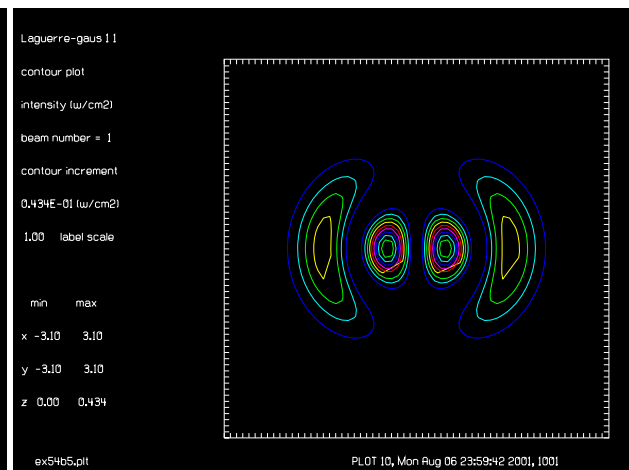


Fig. 54.10. LG 1,1.

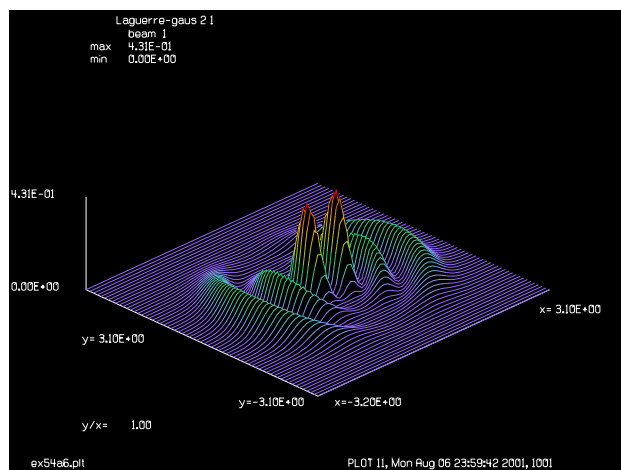


Fig. 54.11. LG 2,1.

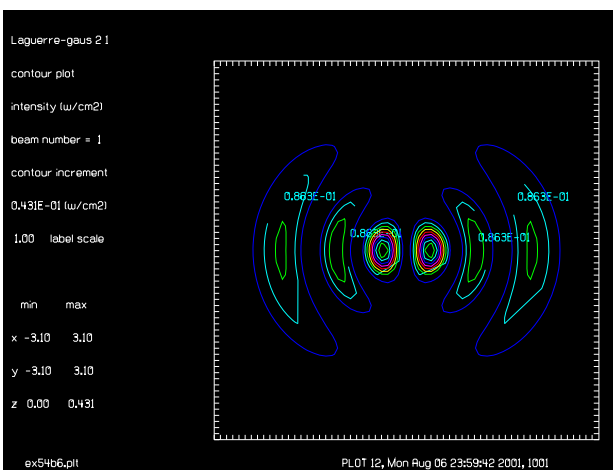


Fig. 54.12. LG 2,1.

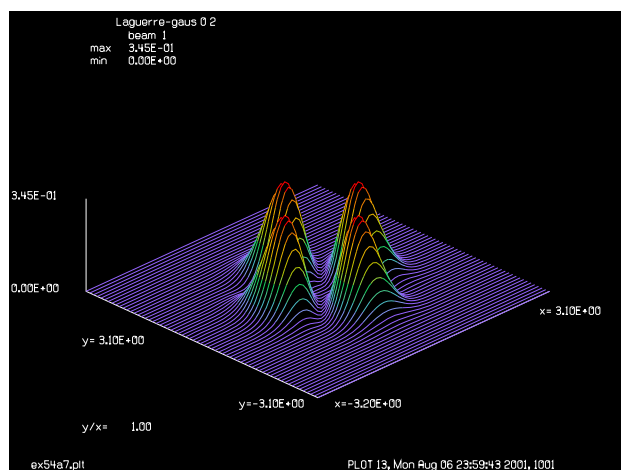


Fig. 54.13. LG 0,2.

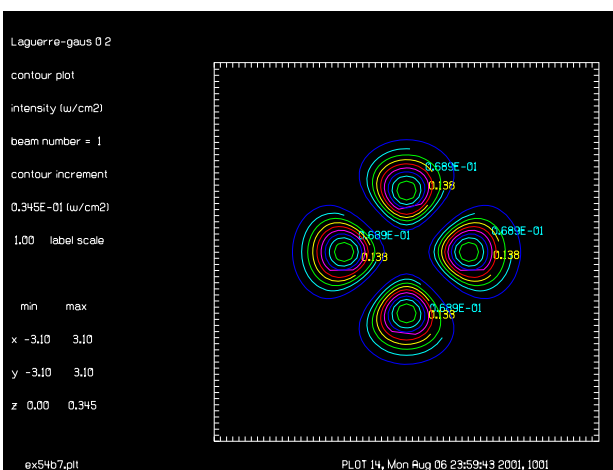


Fig. 54.14. LG 0,2.

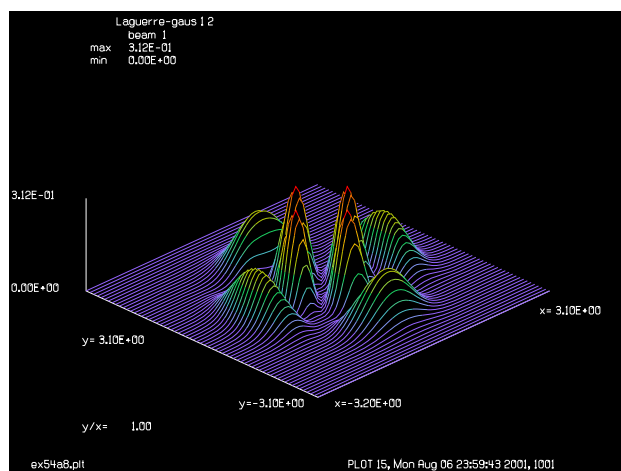


Fig. 54.15. LG 1,2.

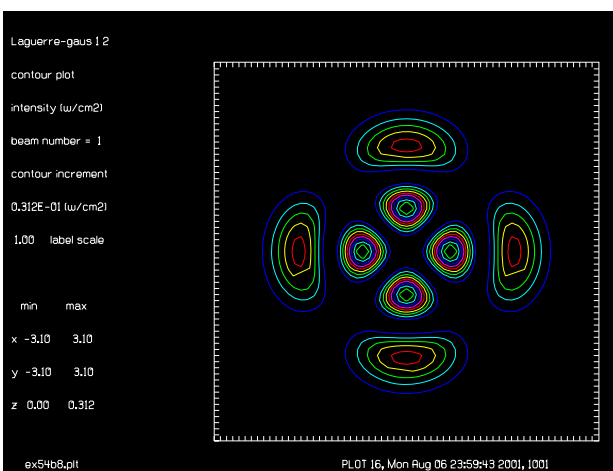


Fig. 54.16. LG 1,2.

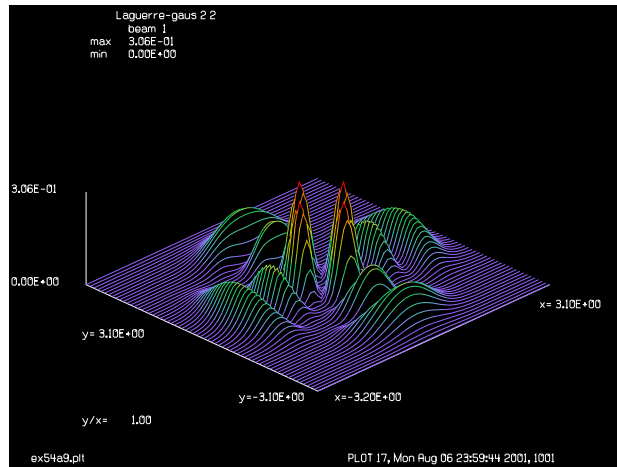


Fig. 54.17. LG 2,2.

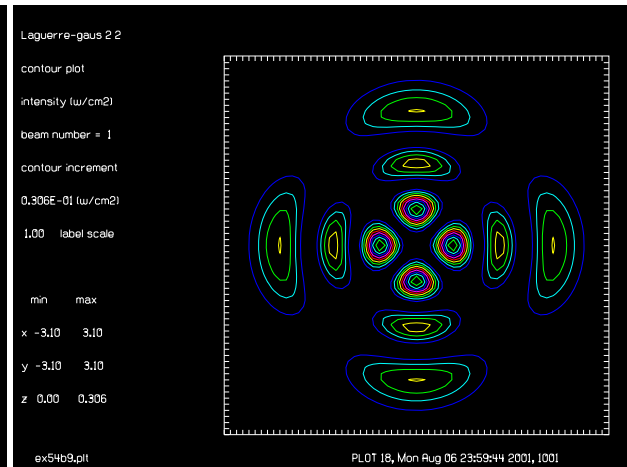


Fig. 54.18. LG 2,2.

```

c   Example 54: Some Laguerre-Gaussian
c
echo/on
variab/dec/int plot_num max_order x_order y_order
macro/def loop2/o
    y_order = y_order + 1
    x_order = -1
    macro loop1/max_order
macro/end
macro/def loop1/o
    x_order = x_order + 1
    laguerre/ell/con 1 1 1 1 x_order y_order
    plot_num = plot_num + 1
    title Laguerre-gaus @x_order @y_order
    plot/watch ex54a@plot_num.plt
    plot/l ns=64
    plot/watch ex54b@plot_num.plt
    plot/con
macro/end
units/s 1 .1
y_order = -1
max_order = 3                      # set maximum order number
plot_num = 0
macro loop2/max_order
end

```

Ex55: Speckle pattern in the far-field

This example illustrates the nature of the speckle pattern in the far-field. A smoothed random wavefront is focused to create the speckle pattern. A series of scans across the image are taken at different axial points. A total of 64 axial scans are taken. Each axial scan is displayed in Fig. 55.3, which displays the contorted, snake-like speckle pattern. The scale in the transverse direction has been expanded to more clearly show the speckle structure.

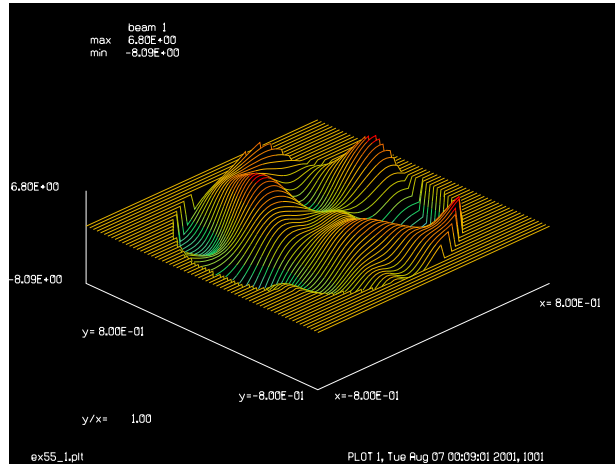


Fig. 55.1. Pupil aberration exhibiting smoothed random aberration of moderately high spatial frequency.

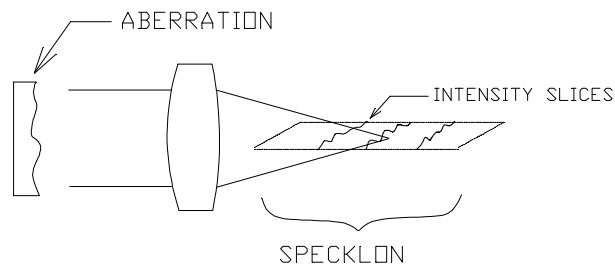


Fig. 55.2. A series of scans across the image are taken at 64 axial positions. A series of scans across the image are taken at 64 axial positions to make a through-focus image.

Input: ex55.inp

```
c## ex55
c
c Example 55: Speckle Pattern at the Image Plane
c
c This example illustrates the formation of the far-field pattern
c of rather high-spatial frequency aberration. A series of
c propagations through the image are taken. At each axial point,
c The center row of Beam 1 is copied to a specific row of Beam 2
c with an off-set. This allows an overview of the through-focus
c pattern.
c
c variab/dec/int row
```

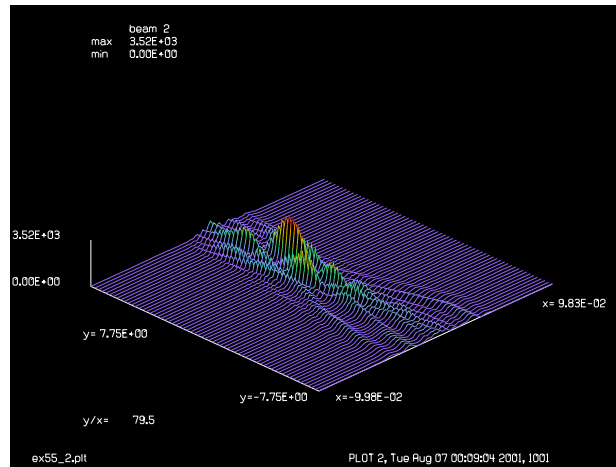


Fig. 55.3. The speckle pattern shows twisted structure in a snake-like pattern. The scale in the transverse dimension is expanded to more clearly illustrate the structure.

```

nbeam 2
array/set 1 128
units/s 1 .025                                # Set units.
array/set 2 128 64
units/s 2 .00156 .25
wavelength/set 0 .5                            # Set wavelength.
clear 1 1                                       # Initialize arrays.
clap/cir/con 1 .7
phase/ran 1 .5 .15 is=1
plot/watch ex55_1.plt
plot/l/ph 1 xrad=.8 ns=128
clear 2 0
lens 1 100
zone/fix 100 8.1
zone/in 1
dist 94 1                                     # Move within 4 cm of focus.
row = 0
macro/def ex55/o
  row = row + 1 list
  dist .25 1
  copy/row 1 2 65 row
macro/end
macro ex55/64
plot/watch ex55_2.plt
plot/l 2 h=.2 ns=64
end

```

Ex56: Fabry-Perot cavity with coherent injection

Table. 56.1. Table of Ex56 examples

Ex56a: Find Gouy phase of ideal gaussian mode	1
Ex56b: Scan coherent injected signal around Gouy phase, peak at 140 deg.	2
Ex56c: Orthogonalization to show build-up and Gouy phase of second lowest loss mode	4
Ex56d: Coherent laser injection tuned for gaussian mode (real apertures)	6
Ex56e: Coherent laser injection tuned for gaussian mode (real apertures)	8
Ex56f: Scan injected signal over longitudinal mode spacing	11

This example is similar to Example 33 except that the resonator is driven by coherent injection of a laser beam. The exact frequency of the injected signal is important in causing discrimination between transverse modes. In Example 33, the aperture caused different diffraction losses for the different modes. If the coherent injected signal is close to the frequency of a transverse mode, that mode will be build up preferentially.

Five runs are used to investigate the coherent injection.

Table. 56.2. Parameters

length	45 cm
mirror radius	50 cm
wavelength	1.064 microns
Rayleigh range	15 cm

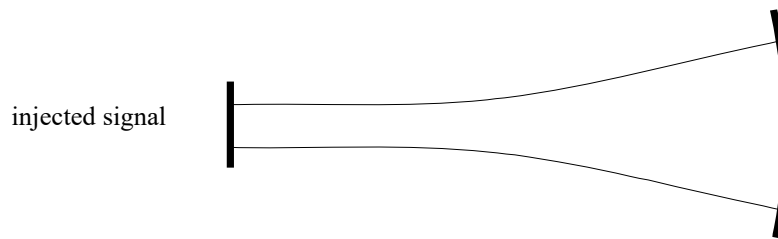


Fig. 56.1. Stable resonator configuration. The waist will form on the flat mirror and the phase radius will match the radius of the concave mirror in the ideal geometric mode.

Ex56a: Find Gouy phase of ideal gaussian mode

Input: `ex56a.inp`

```

c## ex56a
c
c Example 56A: Bare Cavity Stable Resonator with Coherent Injection
c
c The configuration consists of a flat mirror and a spherical mirror.
c Surrogate gaussian beam is initially set-up to exactly match the
c stable geometric mode. Beam 1 is left with the ideal mode and
c Beam 2 is modified to have a uniform intensity with the CLEAR
c command which does not change the surrogate gaussian beam
c parameters.

```

```

c
c The waist of the ideal resonator mode is w0=.02253936. At
c the curved mirror the beam radius is w=.07128. The mirror aperture
c is selected to be twice the beam radius at the mirror at .14 cm.
c
c The steady-state mode depends on the aperture losses. Both Beam 1
c and Beam 2 are affected by the aperture and converge to the same
c steady-state solution. Beam 1 converges quickly because the starting
c condition is the ideal geometric mode which is very close to the
c the steady-state mode. Beam 2 starts from uniform intensity and
c takes about 100 iterations to converge.
c
nbeam 2                                # establish 2 beams
array/s 0 32
wavelength/set 0 1.064                 # set wavelengths
units/s 0 .01
c
c Use ideal mode shape, since Ex33 proved the converged mode is
c almost identical to the ideal mode.
c
gaus/c/c 1 1 .02253936                 # set to geometric mode
energy/norm 1 1                       # normalize energies
copy 1 2
beam/off 2
c
c Make one round-trip of resonator
c
prop 45                                # propagate 45 cm.
mirror/sph 1 -50                      # mirror of 50 cm. radius
clap/c/c 1 .14                        # .14 cm. radius aperture
prop 45                                # propagate 45 cm. along beam
mirror/sph 1 1.e15                    # flat mirror
c
echo/on
c Calculate correlation after one round trip
c Phase rotation (Guoy phase) is -143.1301
c
mult/mode/corr 1 2                    # find complex correlation
end

```

Ex56b: Scan coherent injected signal around Gouy phase, peak at 140 deg.

Input: ex56b.inp

```

c## ex56b!213052764335507
c
c Example 56B: Bare Cavity Stable Resonator with Coherent Laser
c             Injection
c
c The configuration consists of a flat mirror and a spherical mirror.
c Surrogate gaussian beam is initially set-up to exactly match the

```

Jump to: [Commands](#), [Theory](#)

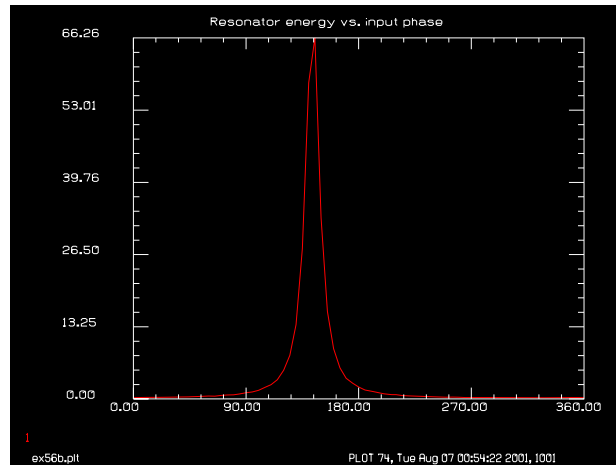


Fig. 56b.1. Resonator energy versus phase of the injected signal. The maximum field occurs at about 140° — the Gouy shift for the gaussian beam.

```

c  stable geometric mode.  Beam 1 is left with the ideal mode and
c  Beam 2 is modified to have a uniform intensity with the CLEAR
c  command which does not change the surrogate gaussian beam
c  parameters.
c
c  The waist of the ideal resonator mode is w0=.02253936.  At
c  the curved mirror the beam radius is w=.07128.  The mirror aperture
c  is selected to be twice the beam radius at the mirror at .14 cm.
c
c  The steady-state mode depends on the aperture losses.  Both Beam 1
c  and Beam 2 are affected by the aperture and converge to the same
c  steady-state solution.  Beam 1 converges quickly because the starting
c  condition is the ideal geometric mode which is very close to the
c  the steady-state mode.  Beam 2 starts from uniform intensity and
c  takes about 100 iterations to converge.
c
c
variab/dec/int pass
macro/def reson56b/o
    add/coh 1 2                # coherent addition
    prop 45                    # propagate 45 cm.
    mirror/sph 1 -50           # mirror of 50 cm. radius
    prop 45                    # propagate 45 cm. along beam
    mirror/sph 1 1.e15         # flat mirror
    mult/scalar 1 .8           # round-trip efficiency
    phase/piston 1 phase      # adjust for phase delay
    energy
    geodata
macro/end
macro/def out56b/o
    pass = pass + 1            # increment pass counter
    phase = phase + 5          # increment phase
    copy 2 1
    mult/scalar 1 50           # make a guess at solution
    write/d xxx.out/o

```

```

resonator/run 50
write/d ex56b.out
energy
variab/set energy 1 energy
udata/set pass phase energy
title Resonator energy vs. input phase, pass = @pass
plot/udata first=1 last=1 min=0.
macro/end
nbeam 2                                # establish 2 beams
array/s 0 32
wavelength/set 0 1.064                  # set wavelengths
units/s 0 .01
beam/off 2
pass = 0                                # initialize pass counter
phase = -5                              # offset phase
resonator/name reson56b
resonator/eigen/test 1
resonator/eigen/set 1
resonator/eigen/list
pass = 0                                # initialize pass counter
phase = -5                              # offset phase
energy/norm 1 1                         # normalize energies
copy 1 2
status/p
mac/run out56b/73
title Resonator energy vs. input phase
plot/watch ex56b.plt
plot/udata first=1 last=1 min=0.
udata/list/disk ex56b.uda
end

```

Ex56c: Orthogonalization to show build-up and Gouy phase of second lowest loss mode

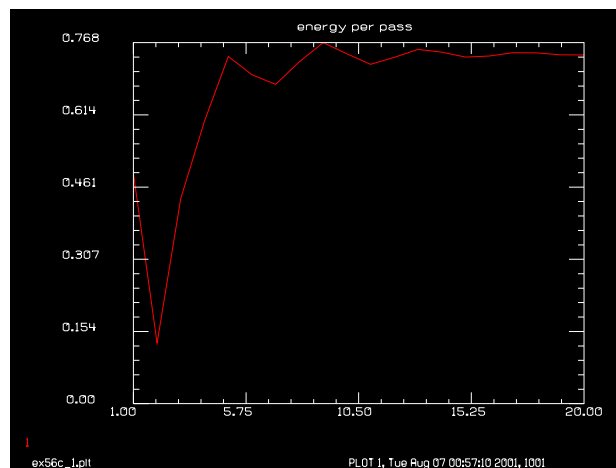


Fig. 56c.1. Build up of second lowest loss mode with orthogonalization against the Gaussian mode at each round trip.

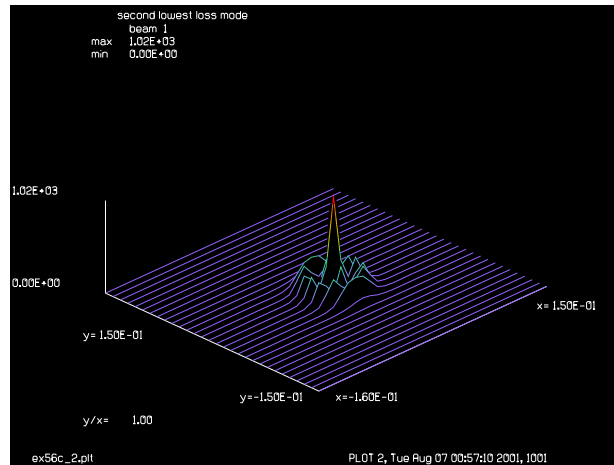


Fig. 56c.2. Second lowest loss eigenmode. Note the similarity to 1,0 Laguerre-Gaussian in Ex. 54.

Input: ex56c.inp

```

c## ex56c
c
c Example 56C: Bare Cavity Stable Resonator with Coherent Laser
c Injection
c
c The configuration consists of a flat mirror and a spherical mirror.
c Surrogate gaussian beam is initially set-up to exactly match the
c stable geometric mode. Beam 1 is left with the ideal mode and
c Beam 2 is modified to have a uniform intensity with the CLEAR
c command which does not change the surrogate gaussian beam
c parameters.
c
c The waist of the ideal resonator mode is w0=.02253936. At
c the curved mirror the beam radius is w=.07128. The mirror aperture
c is selected to be twice the beam radius at the mirror at .14 cm.
c
c The steady-state mode depends on the aperture losses. Both Beam 1
c and Beam 2 are affected by the aperture and converge to the same
c steady-state solution. Beam 1 converges quickly because the starting
c condition is the ideal geometric mode which is very close to the
c the steady-state mode. Beam 2 starts from uniform intensity and
c takes about 100 iterations to converge.
c
variab/dec/int pass
macro/def reson56c/o
    pass = pass + 1                # increment pass counter
    mult/mode/orthogonal 1 2
    prop 45                        # propagate 45 cm.
    mirror/sph 1 -50              # mirror of 50 cm. radius
    clap/c/c 1 .14                # .14 cm radius aperture
    prop 45                        # propagate 45 cm. along beam
    mirror/sph 1 1.e15             # flat mirror
    mult/scalar 1 .8              # round-trip efficiency
    phase/piston 1 143.1301       # Apply Guoy phase for gaussian

```

Jump to: [Commands](#), [Theory](#)

```

                                # mode
    variab/set energy 1 energy
    udata/set pass pass energy
    energy/norm 1 1
macro/end
nbeam 2                                # establish 2 beams
array/s 0 32
wavelength/set 0 1.064                # set wavelengths
units/s 0 .01
energy/norm 1 1                        # normalize energy, Beam 1
zbound/set 1 .02253936                # define surrogate gaussian
                                        # for Beam 1
gaus/c/c 2 1 .02253936                # set to ideal mode
energy/norm 2 1                        # normalize energy, Beam 2
status/p
pass = 0                                # initialize pass counter
beam/off 2
mac/run reson56c/20                    # make 20 passes to converge
mult/mode/corr 1 2                    # check orthogonality
title energy per pass
plot/watch ex56c_1.plt
plot/udata first=1 last=1 min=0.
title second lowest loss mode
plot/watch ex56c_2.plt
plot/l 1
c
c make another pass to calculate Guoy phase for second lowest
c loss mode
c
copy 1 2                                # store Beam 1
prop 45                                # propagate 45 cm.
mirror/sph 1 -50                       # mirror of 50 cm. radius
clap/c/c 1 .14                         # .14 cm radius aperture
prop 45                                # propagate 45 cm. along beam
mirror/sph 1 1.e15                     # flat mirror
mult/scalar 1 .8                       # round-trip efficiency
echo/on
c
c Guoy shift should be -69 degrees
c
mult/mode/corr 1 2                    # correlation after one round-trip
                                        # check Guoy phase
end

```

Ex56d: Coherent laser injection tuned for gaussian mode (real apertures)

Input: `ex56d.inp`

```

c## ex56d
c
c Example 56D: Bare Cavity Stable Resonator with Coherent Laser
c Injection

```

Jump to: [Commands](#), [Theory](#)

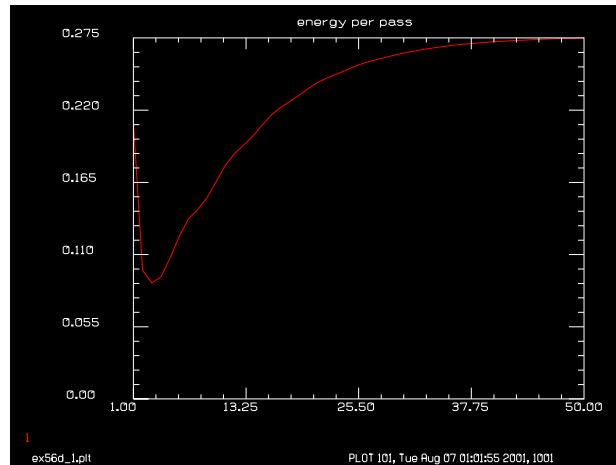


Fig. 56d.1. Build up of field in resonator with tuning for the gaussian Gouy phase delay.

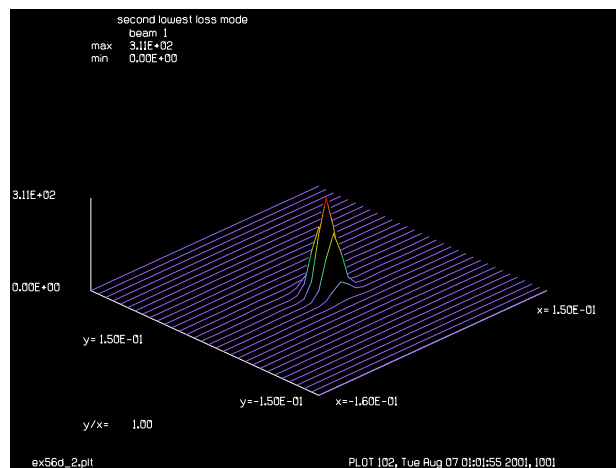


Fig. 56d.2. Gaussian mode is formed for properly tuned injected signal.

```

c
c The configuration consists of a flat mirror and a spherical mirror.
c Surrogate gaussian beam is initially set-up to exactly match the
c stable geometric mode. Beam 1 is left with the ideal mode and
c Beam 2 is modified to have a uniform intensity with the CLEAR
c command which does not change the surrogate gaussian beam
c parameters.
c
c The waist of the ideal resonator mode is w0=.02253936. At
c the curved mirror the beam radius is w=.07128. The mirror aperture
c is selected to be twice the beam radius at the mirror at .14 cm.
c
c The steady-state mode depends on the aperture losses. Both Beam 1
c and Beam 2 are affected by the aperture and converge to the same
c steady-state solution. Beam 1 converges quickly because the starting
c condition is the ideal geometric mode which is very close to the
c the steady-state mode. Beam 2 starts from uniform intensity and
c takes about 100 iterations to converge.
c

```

Jump to: [Commands](#), [Theory](#)

```

variab/dec/int pass
macro/def reson56d/o
    pass = pass + 1          # increment pass counter
    add/coh 1 2
    prop 45                  # propagate 45 cm.
    mirror/sph 1 -50         # mirror of 50 cm. radius
    clap/c/c 1 .14          # .14 cm radius aperture
    prop 45                  # propagate 45 cm. along beam
    mirror/sph 1 1.e15       # flat mirror
    mult/scalar 1 .8         # round-trip efficiency
    phase/piston 1 143.1301  # Apply Guoy phase for lowest
                             # mode

    variab/set energy 1 energy
    udata/set pass pass energy
    plot/l 1
    plot/udata first=1 last=1 min=0.
macro/end
nbeam 2                     # establish two beams
array/s 0 32
wavelength/set 0 1.064      # set wavelengths
units/s 0 .01
zbound/set 1 .02253936     # define surrogate gaussian
                             # for Beam 1

copy 1 2
status/p
pass = 0                    # initialize pass counter
beam/off 2
mac/run reson56d/50         # make 50 passes to converge
title energy per pass
plot/watch ex56d_1.plt
plot/udata first=1 last=1 min=0.
title second lowest loss mode
plot/watch ex56d_2.plt
plot/l 1
udata/list/disk ex56d.uda
end

```

Ex56e: Coherent laser injection tuned for gaussian mode (real apertures)

Input: ex56e.inp

```

c## ex56e
c
c Example 56E: Bare Cavity Stable Resonator with Coherent Laser
c Injection
c
c The configuration consists of a flat mirror and a spherical mirror.
c Surrogate gaussian beam is initially set-up to exactly match the
c stable geometric mode. Beam 1 is left with the ideal mode and
c Beam 2 is modified to have a uniform intensity with the CLEAR
c command which does not change the surrogate gaussian beam
c parameters.
c

```

Jump to: [Commands](#), [Theory](#)

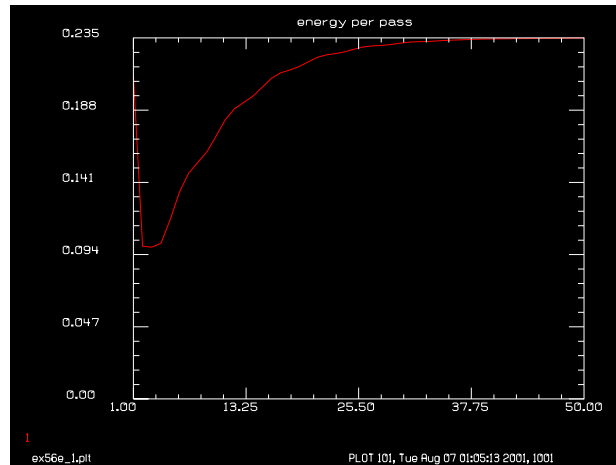


Fig. 56e.1. Build up of field in resonator with tuning for the second lowest loss mode at 68° phase delay.

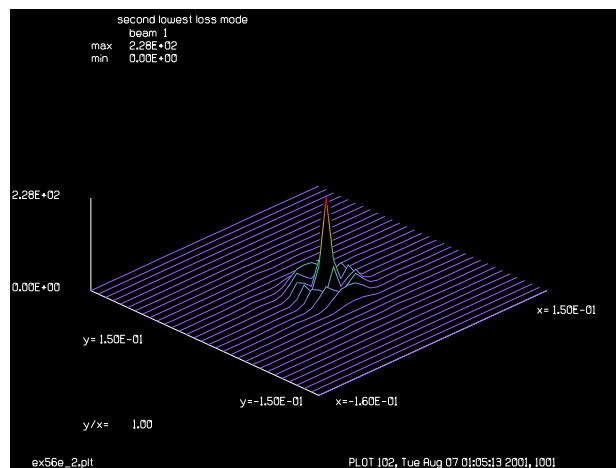


Fig. 56e.2. Second lowest loss mode is formed as predicted.

```

c The waist of the ideal resonator mode is w0=.02253936. At
c the curved mirror the beam radius is w=.07128. The mirror aperture
c is selected to be twice the beam radius at the mirror at .14 cm.
c
c The steady-state mode depends on the aperture losses. Both Beam 1
c and Beam 2 are affected by the aperture and converge to the same
c steady-state solution. Beam 1 converges quickly because the starting
c condition is the ideal geometric mode which is very close to the
c the steady-state mode. Beam 2 starts from uniform intensity and
c takes about 100 iterations to converge.
c
variab/dec/int pass
macro/def reson56e/o
    pass = pass + 1                # increment pass counter
    add/coh 1 2
    prop 45                        # propagate 45 cm.
    mirror/sph 1 -50              # mirror of 50 cm. radius
    clap/c/c 1 .14                # .14 cm radius aperture
    prop 45                        # propagate 45 cm. along beam

```

Jump to: [Commands](#), [Theory](#)

```

mirror/sph 1 1.e15          # flat mirror
mult/scalar 1 .8           # round-trip efficiency
phase/piston 1 68.9117     # Apply Guoy phase for second
                             # lowest mode

variab/set energy 1 energy
udata/set pass pass energy
plot/l 1
plot/udata first=1 last=1 min=0.
macro/end
nbeam 2                    # establish two beams
array/s 0 32
wavelength/set 0 1.064     # set wavelengths
units/s 0 .01
zbound/set 1 .02253936    # define surrogate gaussian
                             # for Beam 1

copy 1 2
status/p
pass = 0                   # initialize pass counter
beam/off 2
mac/run reson56e/50        # make 50 passes to converge
title energy per pass
plot/watch ex56e_1.plt
plot/udata first=1 last=1 min=0.
title second lowest loss mode
plot/watch ex56e_2.plt
plot/l 1
udata/list/disk ex56e.uda
end
c## ex56d
c
c Example 56D: Bare Cavity Stable Resonator with Coherent Laser
c Injection
c
c The configuration consists of a flat mirror and a spherical mirror.
c Surrogate gaussian beam is initially set-up to exactly match the
c stable geometric mode. Beam 1 is left with the ideal mode and
c Beam 2 is modified to have a uniform intensity with the CLEAR
c command which does not change the surrogate gaussian beam
c parameters.
c
c The waist of the ideal resonator mode is  $w_0=.02253936$ . At
c the curved mirror the beam radius is  $w=.07128$ . The mirror aperture
c is selected to be twice the beam radius at the mirror at .14 cm.
c
c The steady-state mode depends on the aperture losses. Both Beam 1
c and Beam 2 are affected by the aperture and converge to the same
c steady-state solution. Beam 1 converges quickly because the starting
c condition is the ideal geometric mode which is very close to the
c the steady-state mode. Beam 2 starts from uniform intensity and
c takes about 100 iterations to converge.
c
variab/dec/int pass
macro/def reson56d/o
    pass = pass + 1        # increment pass counter

```

Jump to: [Commands](#), [Theory](#)


```

add/coh 1 2
prop 45                                # propagate 45 cm.
mirror/sph 1 -50                       # mirror of 50 cm. radius
clap/c/c 1 .14                         # .14 cm radius aperture
prop 45                                # propagate 45 cm. along beam
mirror/sph 1 1.e15                     # flat mirror
mult/scalar 1 .8                       # round-trip efficiency
phase/piston 1 143.1301               # Apply Guoy phase for lowest
                                      # mode

variab/set energy 1 energy
udata/set pass pass energy
plot/l 1
plot/udata first=1 last=1 min=0.
macro/end
nbeam 2                                # establish two beams
array/s 0 32
wavelength 0 1.064                    # set wavelengths
units 0 .01
zbound/set 1 .02253936                # define surrogate gaussian
                                      # for Beam 1

copy 1 2
status/p
pass = 0                               # initialize pass counter
beam/off 2
mac/run reson56d/50                   # make 50 passes to converge
title energy per pass
plot/watch ex56d_1.plt
plot/udata first=1 last=1 min=0.
title second lowest loss mode
plot/watch ex56d_2.plt
plot/l 1
udata/list/disk ex56d.uda
end

```

Ex56f: Scan injected signal over longitudinal mode spacing

Input: ex56f.inp

```

c## ex56f!882808332431668
c
c Example 56F: Bare Cavity Stable Resonator with Coherent Laser
c Injection
c
c The configuration consists of a flat mirror and a spherical mirror.
c Surrogate gaussian beam is initially set-up to exactly match the
c stable geometric mode. Beam 1 is left with the ideal mode and
c Beam 2 is modified to have a uniform intensity with the CLEAR
c command which does not change the surrogate gaussian beam
c parameters.
c
c The waist of the ideal resonator mode is w0=.02253936. At
c the curved mirror the beam radius is w=.07128. The mirror aperture
c is selected to be twice the beam radius at the mirror at .14 cm.

```

Jump to: [Commands](#), [Theory](#)

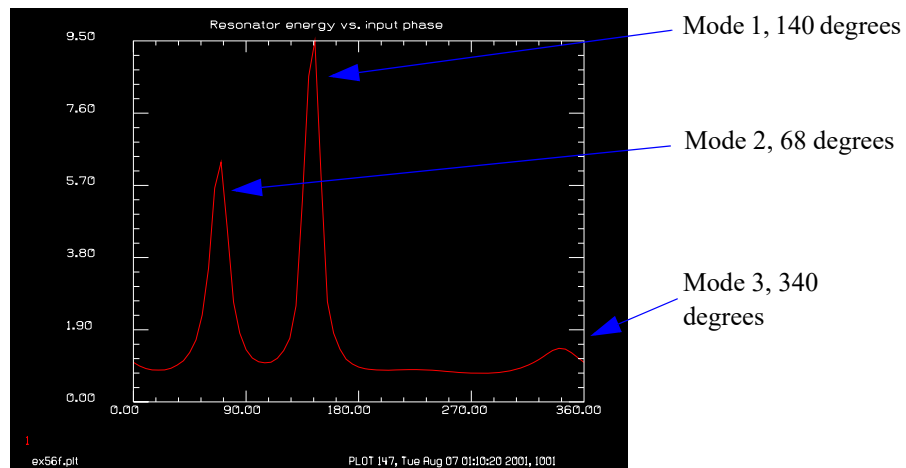


Fig. 56f.1. Scan of phase of injected signal from 0° to 360° . This figure shows the transverse mode frequency shifts with respect to the paraxial longitudinal mode. Finite aperture size causes transverse mode discrimination in addition to the frequency tuning.

```

c
c The steady-state mode depends on the aperture losses. Both Beam 1
c and Beam 2 are affected by the aperture and converge to the same
c steady-state solution. Beam 1 converges quickly because the starting
c condition is the ideal geometric mode which is very close to the
c the steady-state mode. Beam 2 starts from uniform intensity and
c takes about 100 iterations to converge.
c
variab/dec/int pass STOP
macro/def reson56f/o
    add/coh/con 1 2                # coherent addition
    prop 45                        # propagate 45 cm.
    mirror/sph 1 -50               # mirror of 50 cm. radius
    clap/c/c 1 .14                 # .14 cm radius aperture
    prop 45                        # propagate 45 cm. along beam
    mirror/sph 1 1.e15             # flat mirror
    mult/scalar 1 .8               # round-trip efficiency
    phase/piston 1 phase           # adjust for phase delay
    gain/converge/test 1 STOP
    if STOP macro/exit
macro/end
macro/def out56f/o
    pass = pass + 1                # increment pass counter
    phase = phase + 5              # increment phase
    copy 2 1
    write/off
    gain/converge/set eps1=.01 np=5
    mac/run reson56f/50
    write/on
    gain/converge/list
    write/d ex56f.out
    variab/set energy avgeng
    udata/set pass phase energy
    plot/l 1

```

```

    plot/udata first=1 last=1 min=0.
macro/end
nbeam 2                                # establish 2 beams
array/s 0 32
wavelength/set 0 1.064                 # set wavelengths
units/s 0 .01
gaus/c/c 1 1 .1 4
energy/norm 1 1
zbound/set 1 .02253936
c
c Copy beams to make all parameters identical
c
copy 1 2
c
c Start with plane waves to stimulate all modes
c
status/p
pass = 0                               # initialize pass counter
phase = -5                             # offset phase
beam/off 2
title phase angle @phase deg
mac/run out56f/73
title Resonator energy vs. input phase
plot/watch ex56f.plt
plot/udata first=1 last=1 min=0.
end

```


Ex57: Stable resonator with obscuration to generate higher order modes

If the resonator configuration is significantly modified, the simple gaussian mode may no longer be the lowest-loss mode. Consider the resonator from the previous example with the addition of an obscuration at the center of the flat mirror. If this obscuration is very small, the mode will be largely unaffected but if the hole is increased sufficiently, a mode which has low intensity in the center may prevail in the mode competition. The parameters of the resonator are listed in Table 57.1. The aperture size has been reduced to make convergence faster. In the numerical experiment, the obscuration is increased by 0.005 cm 10 times and the resonator is allowed to settle for 25 round trips. Where there is a clearly defined mode, the laser stabilizes easily in the 25 passes but where two modes have nearly the same round-trip loss, 25 passes are insufficient for settling.

Table. 57.1. Parameters for resonator with hole outcoupling, i.e., central obscurations.

length	45 cm
mirror radius	50 cm
wavelength	1.064 microns

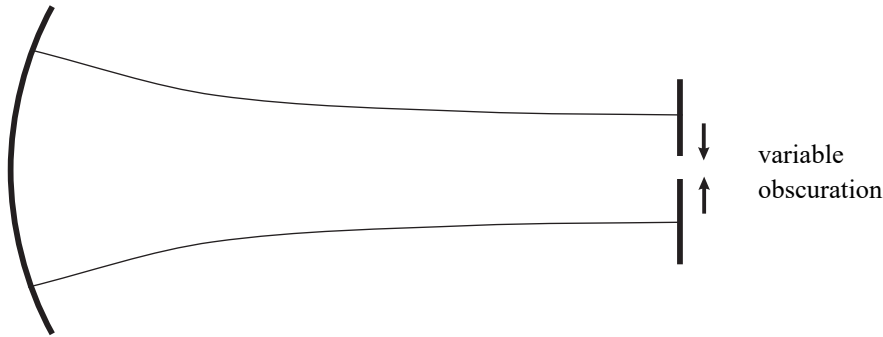


Fig. 57.1. Concave and flat mirror resonator with variable size hole in the flat which forms an obscuration in the beam.

The round-trip energy loss is plotted against the number of passes. At the first set of 25 passes, the obscuration is so small that only the center point is obscured. After the second significant increase in obscuration—at 50 passes—there is significant disruption of the settling, indicating modes keeping. Examination of the isometric plots on the following pages can be compared with the regions of this plot to see the mode changes.

The resonator stayed in the lowest order gaussian mode until about the 100th pass—after the obscuration had been increased three times. We see the formation of a different mode after 150 passes.

At higher obscuration values, we start to get significant azimuthal variation. The finite accuracy of the numerical calculations plays an increasing role as the obscuration increases because so many modes have similar eigenvalues. It is interesting to note that the azimuthal direction tends to break up in to lumps of about the size of the Airy pattern.

Input: `ex57.inp`

```
c## ex57!061893911346998
c
```

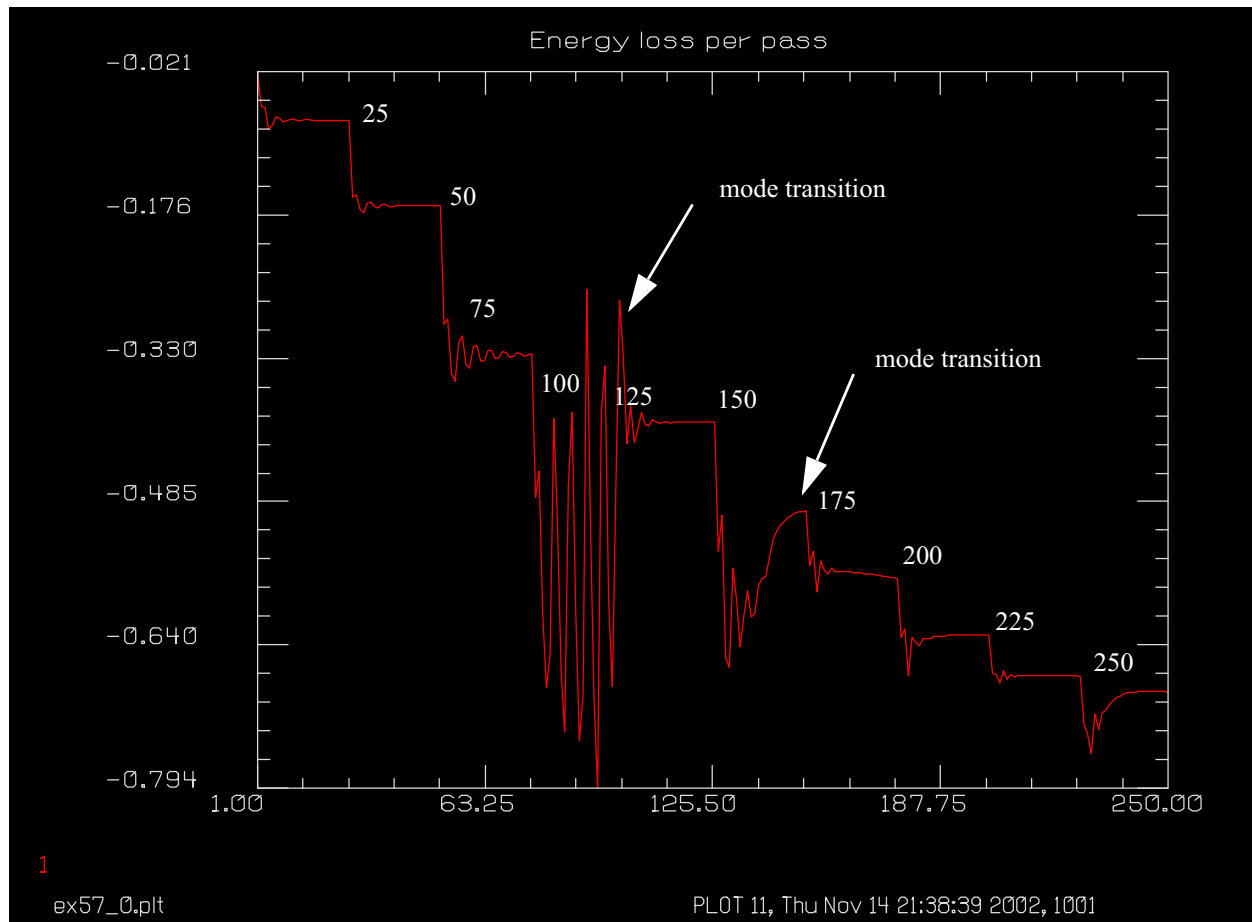


Fig. 57.2. Energy loss as a function of number of passes. Note the first region of poor convergence after 50 passes.

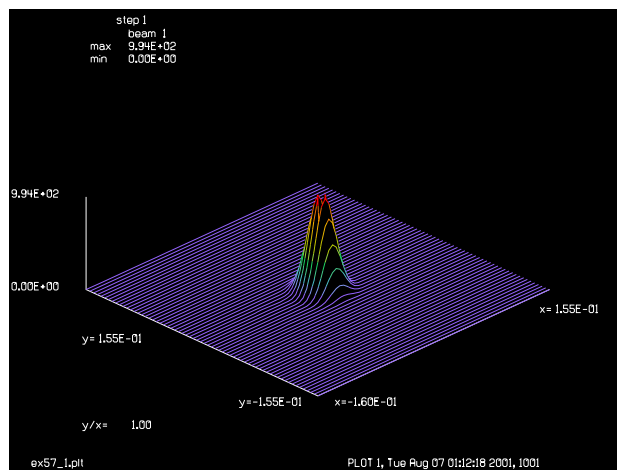


Fig. 57.3. After 25 passes.

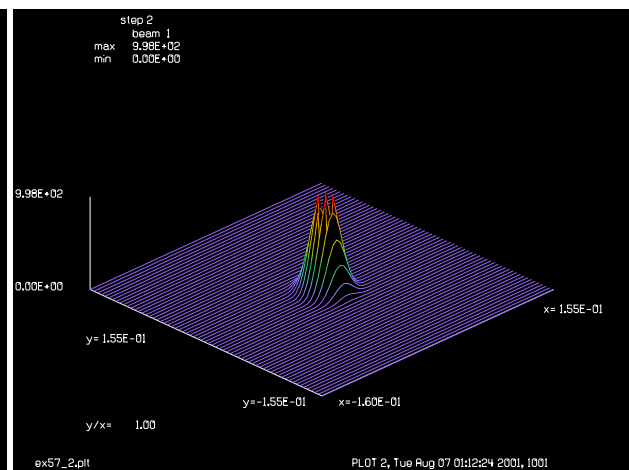


Fig. 57.4. After 50 passes.

c Example 57: Bare Cavity Stable Resonator with Obscuration

c

c This example is the same as Example 33 except that a central
c obscuration is used. The objective is to generate higher order

Jump to: [Commands](#), [Theory](#)

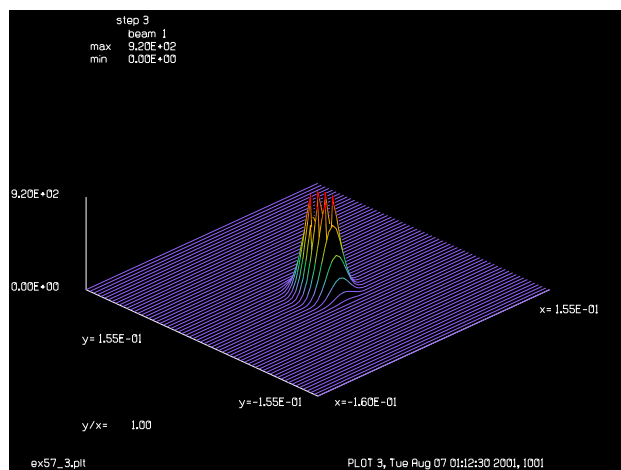


Fig. 57.5. After 75 passes.

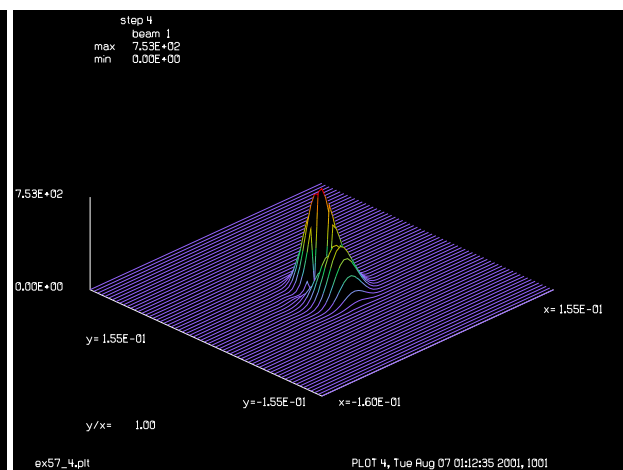


Fig. 57.6. After 100 passes.

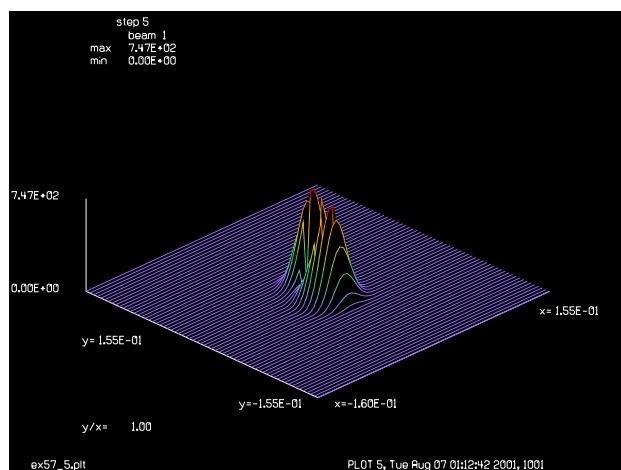


Fig. 57.7. After 125 passes.

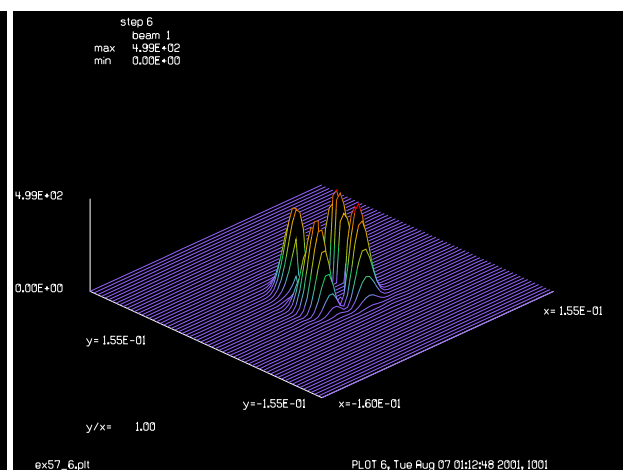


Fig. 57.8. After 150 passes.

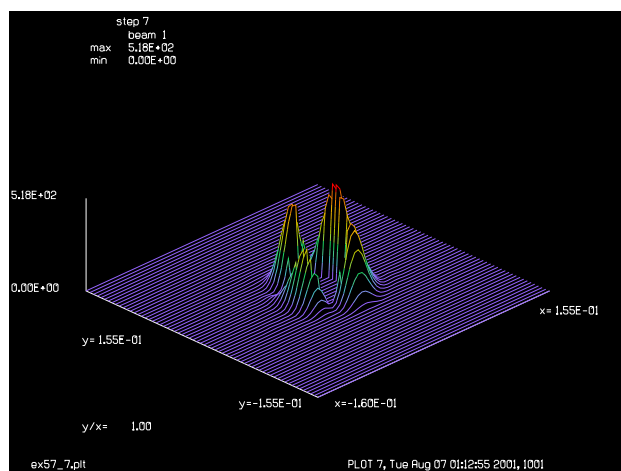


Fig. 57.9. After 175 passes.

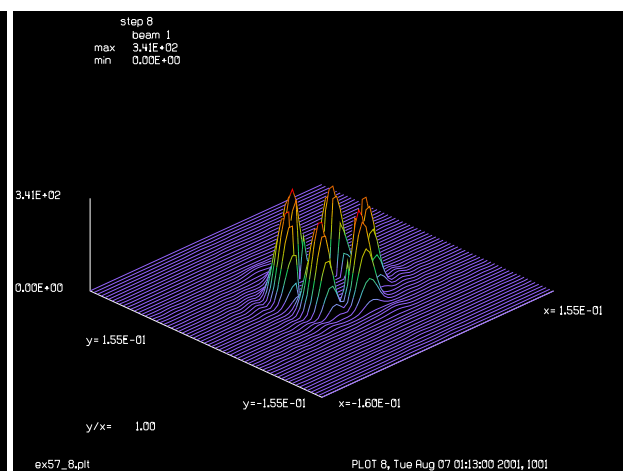


Fig. 57.10. After 200 passes.

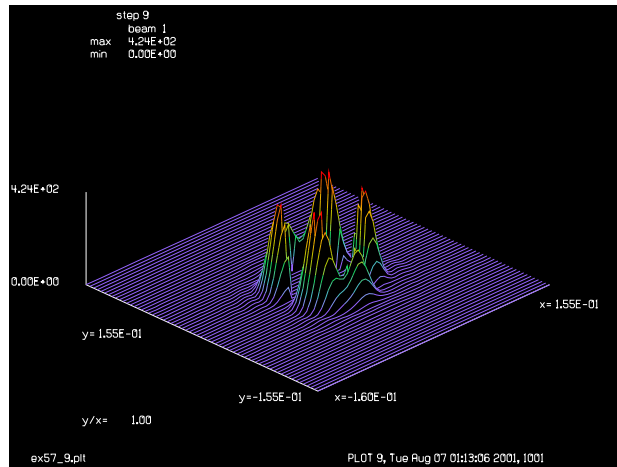


Fig. 57.11. After 225 passes.

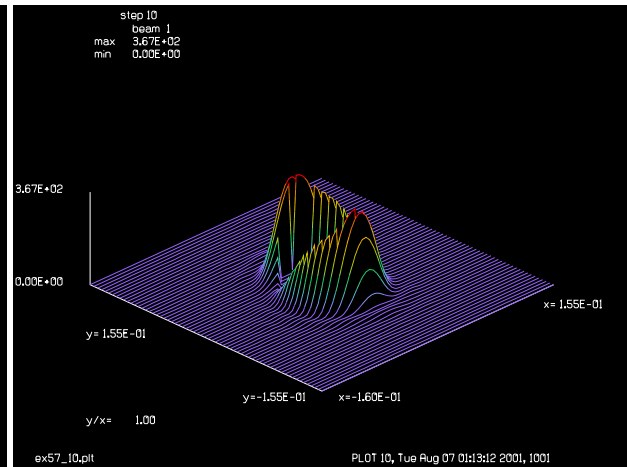


Fig. 57.12. After 250 passes.

```

c transverse modes. The resonator exhibits the problems expected
c of hole outcoupling.
c
variab/dec/int pass plot_num
macro/def reson/o
    pass = pass + 1                # increment pass counter
    prop 45                        # propagate 45 cm.
    mirror/sph 0 -50               # mirror of 50 cm. radius
    clap/c/n 0 .1                  # .1 cm. radius aperture
    prop 45                        # propagate 45 cm. along beam
    mirror/sph 0 1.e15             # flat mirror
    variab/set energy 1 energy
    energy = energy - 1            # calculate energy difference
    udata/set pass pass energy    # store energy differences
    obs/c 0 obs
    energy/norm 1 1                # renormalize energy
macro/end
plot/screen/pause 4
wavelength/set 0 1.064            # set wavelengths
units/s 0 .005
obs = 0.
pass = 0
resonator/name reson
resonator/eigen/test 1
resonator/eigen/set 1
energy/norm 1 1
status/p
pass = 0                           # initialize pass counter
obs = -.0049
macro/def reson1/o
    obs = obs + .005
    resonator/run 25
    plot_num = plot_num + 1
    plot/watch ex57_@plot_num.plt
    title step @plot_num
    plot/1 ns=64

```

Jump to: [Commands](#), [Theory](#)


```
macro/end  
macro reson1/10  
title Energy loss per pass  
plot/watch ex57_0.plt  
plot/udata first=1 last=1  
end
```


Ex58: Effect of absorption and self-focusing in a gaussian beam

Table. 58.1. Table of Ex58 examples

Ex58a: Penetration depth in a Beer's Law absorber, no absorption.	1
Ex58b: Penetration depth in a Beer's Law absorber, absorption included.	3
Ex58c: Penetration depth in a Beer's Law absorber, Beer's Law and self-focusing	4
Ex58d: Penetration depth in a Beer's Law absorber, absorption, self-focusing, aberration	5

This example illustrates the intensity profile of a gaussian beam as it is absorbed by a material. The light is absorbed according to Beer's Law and self-focusing may be present. It has been suggested that self-focusing will cause narrowing of the penetration profile. Four cases are considered,

- 1) ideal gaussian beam
- 2) ideal gaussian beam with absorption
- 3) ideal gaussian beam with absorption and self-focusing
- 4) aberrated gaussian beam with absorption and self-focusing

The configuration of the numerical experiment is shown in Fig. 58.1. A gaussian beam is focused by a lens. The image is scanned transversely and the intensity profile is plotted for a series of longitudinal positions beginning with the paraxial focus and stepping backward. Fig. 58.2 shows the pupil distribution which is the same for all cases. Fig. 58.3 shows the image profile slices for the ideal gaussian beam. The beam diverges slightly because of diffraction. Fig. 58.4 includes absorption according to Beer's Law. Fig. 58.5 includes the effects of self-focusing as well as absorption. Self-focusing causes narrowing of the pulse. Fig. 58.6 illustrates the far-field distribution of a gaussian beam with aberration. The far-field exhibits a considerable degree of speckle. Fig. 58.7 shows the longitudinal slices of the aberrated beam with absorption and self-focusing. Over a sufficiently long distance a three-dimensional specklon would be formed similar to Example 55. In this example, the absorption cuts off the intensity within a shorter distance than the longitudinal speckle length. The primary effect of aberration is to drop the peak intensity by about a factor of 4 and to generate side lobes which are visible in Fig. 58.6 and Fig. 58.7. Since the power is lower, the self-focusing is also less for Case 4.

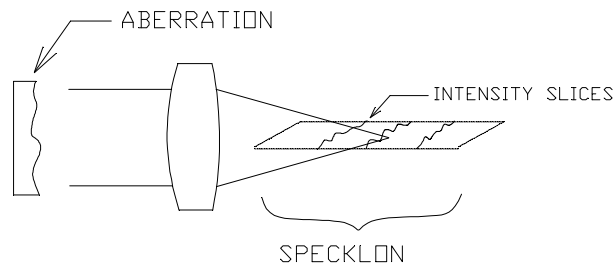


Fig. 58.1. Configuration for examining a series of transverse scans of the image. Figures 58.3, 58.4, 58.5, and 58.7 show the intensity profiles of these transverse scans.

Ex58a: Penetration depth in a Beer's Law absorber, no absorption

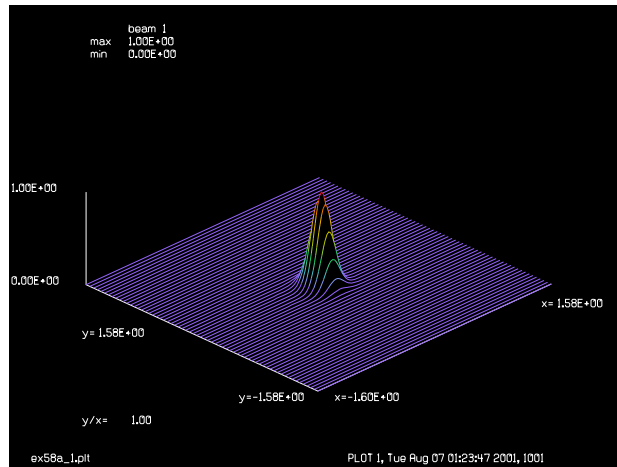


Fig. 58.2. Pupil distribution. Same for all cases.

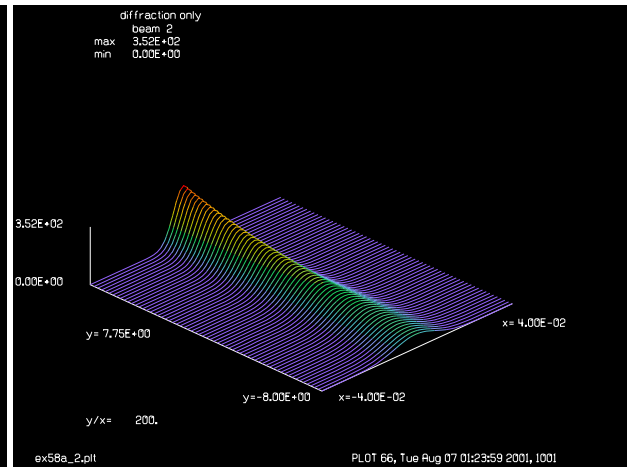


Fig. 58.3. Case 1, image distribution, diffraction spreads the beam but no energy is loss.

Input: ex58a.inp

```

c## ex58a
c
c Example 58a: Penetration depth in a Beer's Law absorber,
c
c Part A: no absorption
c
c The first calculation illustrates the ideal gaussian beam with no
c absorption, self-focusing, or aberration
c
variab/dec/int row
echo/on
nbeam 2
array/set 1 128
units/s 1 .025
array/set 2 128 64
units/s 2 .00156 .25
wavelength/set 0 .5
gaus/c/c 1 1 .2
plot/watch ex58a_1.plt
plot/l 1 ns=64
plot/watch plot1.plt
clear 2 0
lens 1 100
zone/fix 100 8.1
zone/in 1
dist 99.9 1
row = 0
beer/set g0=-.0
beam/off 2
row = 0
macro/def ex58a/o
    row = row + 1
    prop .1

```

Set units.

Set wavelength.

Jump to: [Commands](#), [Theory](#)

```

copy/row 1 2 65 row
plot/1 2 h=.25 xrad=.04 yrad=8 ns=64
macro/end
macro ex58a/64
plot/watch ex58a_2.plt
title diffraction only
plot/1 2 h=.25 xrad=.04 yrad=8 ns=64
end

```

Ex58b: Penetration depth in a Beer's Law absorber, absorption included

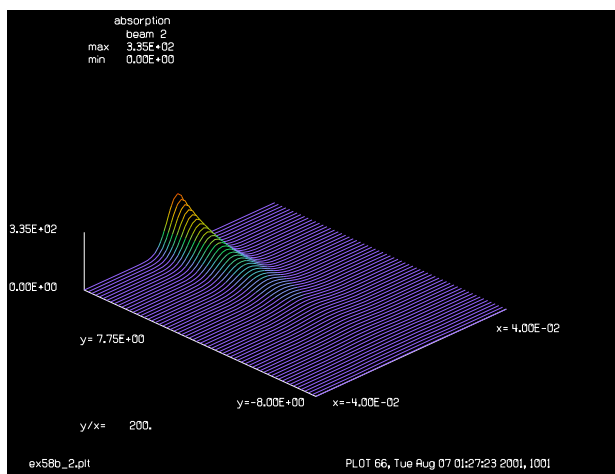


Fig. 58.4. Case 2, effects of absorption.

Input: ex58b.inp

```

c## ex58b
c
c Example 58b: Penetration depth in a Beer's Law absorber,
c
c Part B: absorption included
c
c The penetration depth of radiation in a Beer's Law medium is calculated
c by applying the Beer's Law absorber
c
variab/dec/int row
nbeam 2
array/set 1 128
units/s 1 .025 # Set units.
array/set 2 128 64
units/s 2 .00156 .25 # Set wavelength.
wavelength/set 0 .5
gaus/c/c 1 1 .2
plot/watch ex58b_1.plt
plot/1 1 ns=64
plot/watch plot1.plt
clear 2 0

```

Jump to: [Commands](#), [Theory](#)

```

lens 1 100
zone/fix 100 8.1
zone/in 1
dist 99.9 1
row = 0
beer/set g0=-.5 es=1e6
macro/def ex58b/o
    row = row + 1
    beer/dist 1 .1
    copy/row 1 2 65 row
    plot/1 2 xrad=.04 yrad=8 h=.25 ns=64
macro/end
macro ex58b/64
plot/watch ex58b_2.plt
title absorption
plot/1 2 xrad=.04 yrad=8 h=.25 ns=64
end

```

Ex58c: Penetration depth in a Beer's Law absorber, Beer's Law and self-focusing

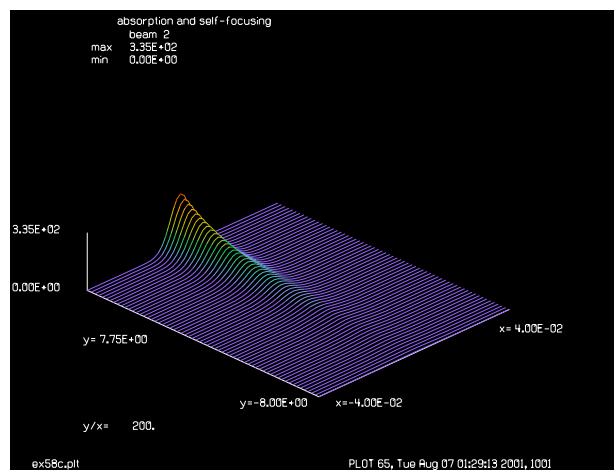


Fig. 58.5. Case 3, absorption and self-focusing. The beam is pinched-in by self-focusing until depleted.

Input: ex58c.inp

```

c## ex58c
c
c Example 58c: Penetration depth in a Beer's Law absorber,
c
c Part C: Beer's Law and self-focusing
c
c The penetration depth of radiation in a Beer's Law medium is calculated
c by applying the Beer's Law absorber and self-focusing
c
variab/dec/int row
nbeam 2
array/set 1 128

```

Jump to: [Commands](#), [Theory](#)

```

units/s 1 .025                                # Set units.
array/set 2 128 64
units/s 2 .00156 .25
wavelength/set 0 .5                            # Set wavelength.
gaus/c/c 1 1 .2
clear 2 0
lens 1 100
zone/fix 100 8.1
zone/in 1
dist 99.9 1
row = 0
beer/set g0=-.5 es=1e6
macro/def ex58c/o
    row = row + 1
    beer/dist 1 .1
    sfocus 1 1e-8 .1
    copy/row 1 2 65 row
    plot/1 2 xrad=.04 yrad=8 h=.25 ns=64
macro/end
macro ex58c/64
title absorption and self-focusing
plot/watch ex58c.plt
plot/1 2 xrad=.04 yrad=8 h=.25 ns=64
end

```

Ex58d: Penetration depth in a Beer's Law absorber, absorption, self-focusing, aberration

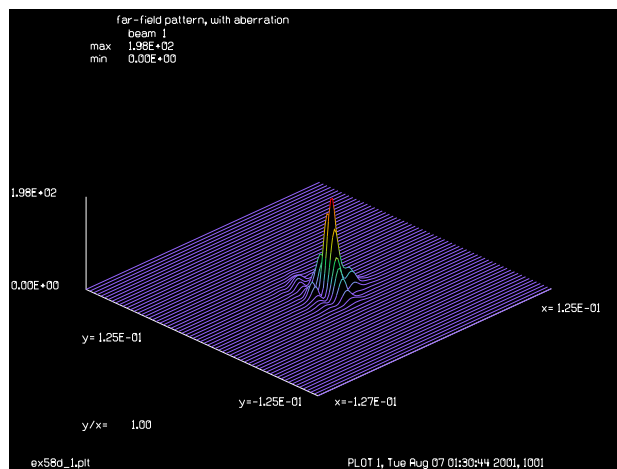


Fig. 58.6. Case 4, far-field showing aberration. Strehl ratio is about 0.25.

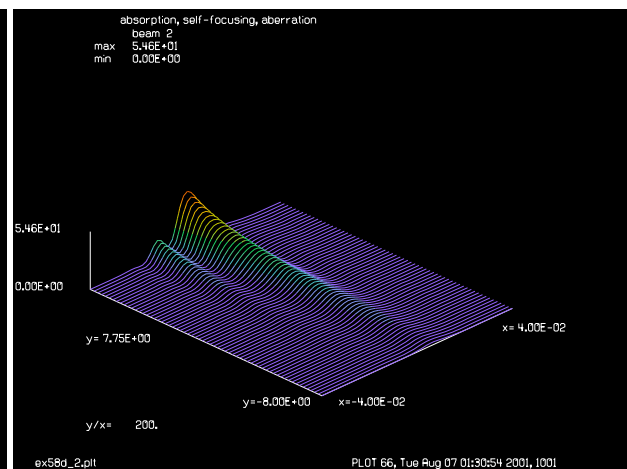


Fig. 58.7. Case 4: aberration, absorption, self-focusing.

Input: ex58d.inp

```

c## ex58d
c
c Example 58D: Penetration depth in a Beer's Law absorber,

```

Jump to: [Commands](#), [Theory](#)

```

c
c Part D: absorption, self-focusing, initial aberration
c
c The penetration depth of radiation in a Beer's Law medium is calculated
c by applying the Beer's Law absorber, self-focusing. The aberration
c causes some offset and twist of the through-focus image distribution.
c
variab/dec/int row
nbeam 2
array/set 1 128
units/s 1 .025 # Set units.
array/set 2 128 64
units/s 2 .00156 .25
wavelength/set 0 .5 # Set wavelength.
gaus/c/c 1 1 .2
nbeam 3
array/set 3 128
units/s 3 .025
clap/c/c 3 1.25
phase/ran 3 .2 .1
mult/beam 1 3
strehl 1
nbeam 2
clear 2 0
lens 1 100
zone/fix 100 8.1
zone/in 1
dist 99.9 1
title far-field pattern, with aberration
plot/watch ex58d_1.plt
plot/l 1 ns=64
plot/watch plot1.plt
row = 0
beer/set g0=-.5 es=1e6
title absorption, self-focusing, aberration
macro/def ex58d/o
    row = row + 1
    beer/dist 1 .1
    sfocus 1 1e-8 .1
    copy/row 1 2 65 row
    plot/l 2 xrad=.04 yrad=8 h=.25 ns=64
macro/end
macro ex58d/64
plot/watch ex58d_2.plt
title absorption, self-focusing, aberration
plot/l 2 xrad=.04 yrad=8 h=.25 ns=64
end

```


Ex59: Through-focus image of spherical aberration with central obscuration

Table. 59.1. Table of Ex59 examples

Ex59a: Through-focus spherical aberration, with obscuration	1
Ex59b: Through-focus spherical aberration, no aberration	4
Ex59c: Two-focal-length lens, through-focus scan	5

This example illustrates the appearance of the image of a wavefront having a significant amount of spherical aberration and a relatively large central obscuration.

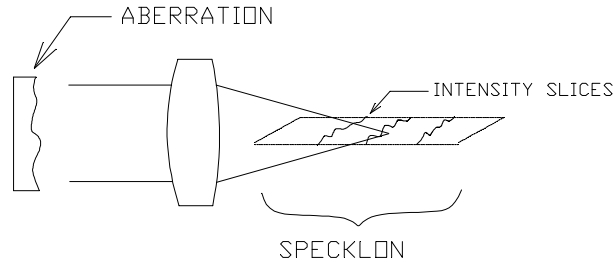


Fig. 59.1. Configuration for examining a series of transverse scans of the image. Figures 4 and 5 show the intensity profiles of these transverse scans.

The spherical aberration was 3 waves of Zernike fourth-order spherical aberration resulting in about 0.2 RMS waves of aberration. Fig. 59.3 shows the through-focus image profiles for the annular aperture. Fig. 59.4 shows the through-focus scan for an unobscured aperture. The spherical aberration is overcorrected so the caustic forms away from the lens. The image is scanned from -0.08 to +0.08 cm, centered at the paraxial focus. Note that in both cases the peak of the image does not occur at the paraxial focal point as might be expected from Zernike spherical aberration which is balanced by defocus. The usual rules for balancing aberration are based on minimizing the wavefront variance. For small amounts of aberration, the minimizing the wavefront variance maximizes the Strehl ratio, but for the large amount of aberration of this example, the optimum balance of defocus is different for minimizing wavefront variance and Strehl ratio. Note also that the through-focus scan for the unobscured aperture, as shown in Fig. 59.5 has the peak much further forward. In addition, the obscured aperture has an image which is more constant through the focus. Spherical aberration in the outer parts of the aperture, is nearly linearly varying radially—like cone aberration as shown in Example 55. Parameters of the system are given in Table 59.2.

Table. 59.2. Parameters.

aperture diameter	120 cm
obscuration diameter	52.8 cm
wavelength	0.5 micron
focal length	2400 cm

Ex59a: Through-focus spherical aberration, with obscuration

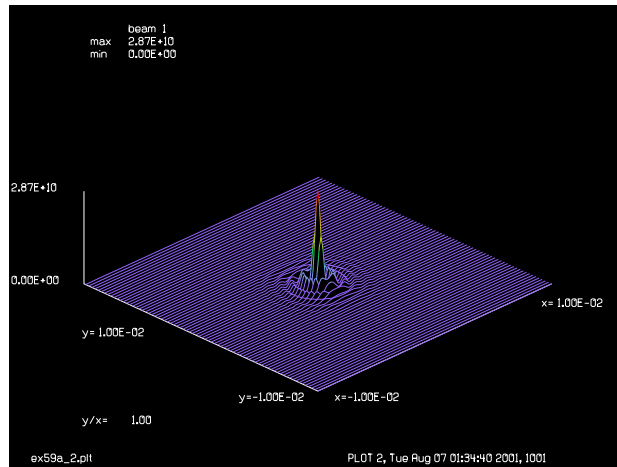


Fig. 59.2. Isometric of intensity at paraxial image.

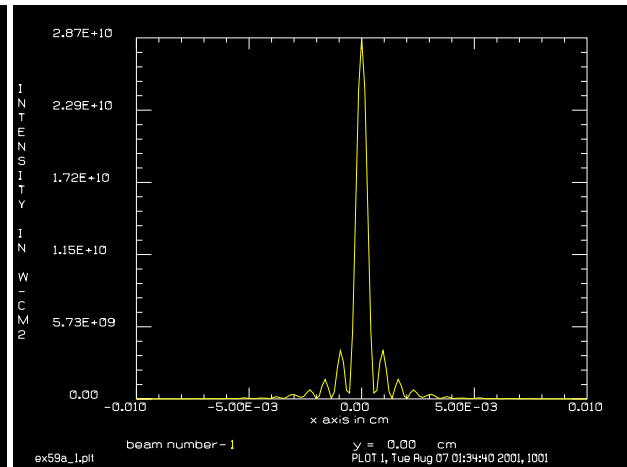


Fig. 59.3. Single slice through paraxial image.

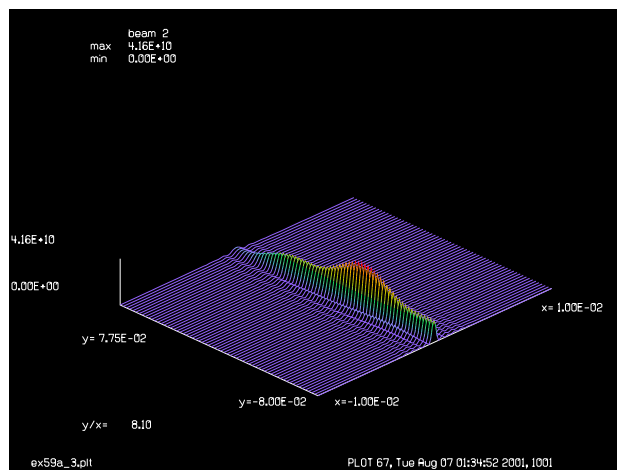


Fig. 59.4. Through-focus image (with obscuration). The lens is to the left. The through-focus scan shows that the image is slowly varying along the axis—similar to cone aberration.

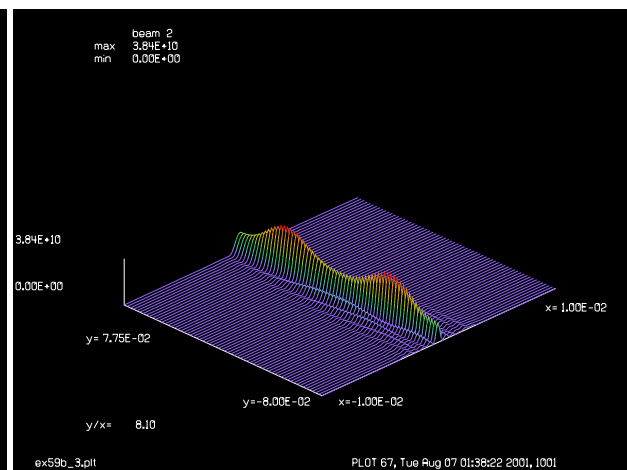


Fig. 59.5. Through-focus image (without obscuration). The peak is forward towards the lens. The spherical aberration is overcorrected and the start of the caustic is evident to the right of the picture.

Input: `ex59a.inp`

```

c## ex59a
c
c Example 59a: Through-focus spherical aberration, with obscuration
c
variab/dec/int row
nbeam 2
array/set 1 256
units/s 1 4
array/set 2 256 64
wavelength/set 0 .5
clap/c/c 1 120
obs 1 52.8
# Set wavelength.

```

Jump to: [Commands](#), [Theory](#)

```

z = 1./-2/pi
abr/zern/rad 1 4 3.*z rn=120
variance 1
clear 2 0
clear 2 0
lens 1 2400
zone/fix 2400 .081
zone/in 1
beam/off 2
dist 2400 1
units 1
geodata/set 1 waistx=3e-3 waisty=3e-3
plot/watch ex59a_1.plt
plot/x/i 1 left=-.01 right=.01
plot/watch ex59a_2.plt
plot/l 1 ns=64 xrad=.01 yrad=.01
plot/watch plot1.plt
variab/set units 1 units
units/s 2 units .0025
dist -.08 1
row = 0
macro/def ex59a/o
    row = row + 1
    dist .0025 1
    copy/row 1 2 129 row
    plot/l 2 h=.2 ns=64 xrad=.01 yrad=.081
macro/end
macro ex59a/64
plot/watch ex59a_3.plt
plot/l 2 h=.2 ns=64 xrad=.01 yrad=.081
end

```

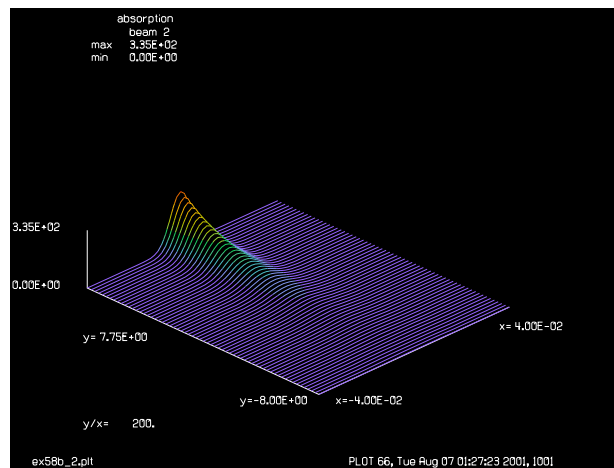
Ex59b: Through-focus spherical aberration, no aberration

Fig. 59.6. Case 2, effects of absorption.

Input: ex59b.inp

```

c## ex59b
c
c Example 59b: Through-focus spherical aberration, no obscuration
c
variab/dec/int row
nbeam 2
array/set 1 256
units/s 1 4
array/set 2 256 64
wavelength/set 0 .5 # Set wavelength.
clap/c/c 1 120
z = 1./-2/pi
abr/zern/rad 1 4 3.*z rn=120
variance 1
clear 2 0
lens 1 2400
zone/fix 2400 .081
zone/in 1
beam/off 2
dist 2400 1
units 1
geodata/set 1 waistx=3e-3 waisty=3e-3
plot/watch ex59b_1.plt
plot/x/i 1 left=-.01 right=.01
plot/watch ex59b_2.plt
plot/l 1 ns=64 xrad=.01 yrad=.01
plot/watch plot1.plt
variab/set units 1 units
units/s 2 units .0025
dist -.08 1
row = 0
macro/def ex59b/o

```

Jump to: [Commands](#), [Theory](#)

```

row = row + 1
dist .0025 1
copy/row 1 2 129 row
plot/1 2 h=.2 ns=64 xrad=.01 yrad=.081
macro/end
macro ex59b/64
plot/watch ex59b_3.plt
plot/1 2 h=.2 ns=64 xrad=.01 yrad=.081
end

```

Ex59c: Two-focal-length lens, through-focus scan

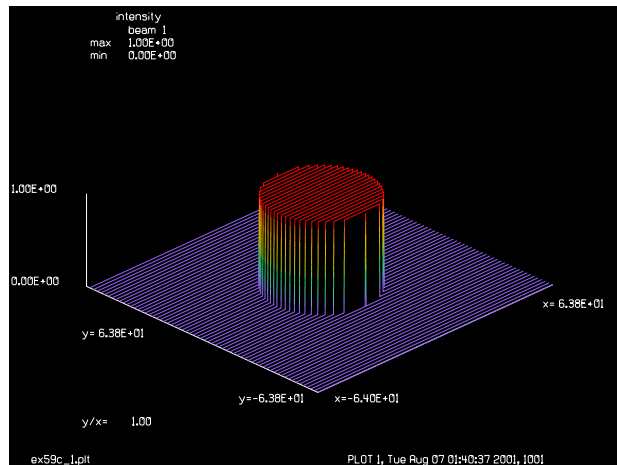


Fig. 59.7. Pupil function for two-focal-length lens.

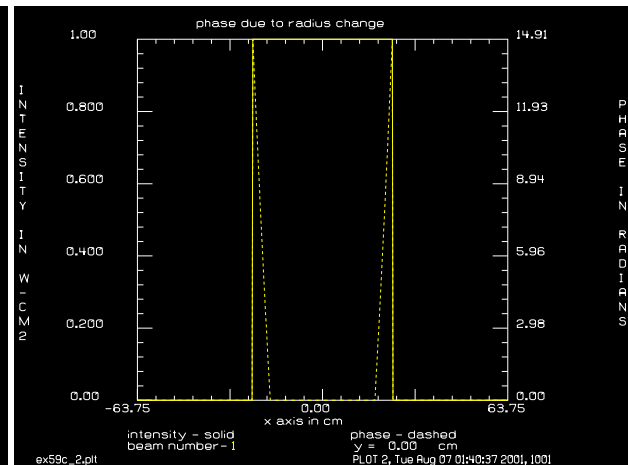


Fig. 59.8. Difference phase in outer zone of pupil. Discontinuity is due to coherent sum.

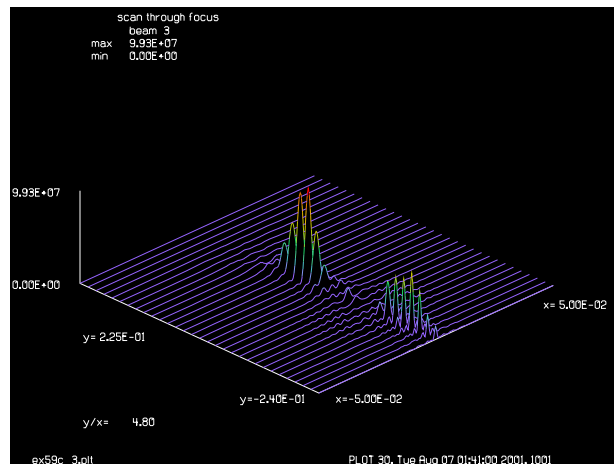


Fig. 59.9. Through-focus image of two-focal-length lens showing two peaks. Y-axis is through-focus distance.

Input: ex59c.inp

```

c## ex59c!713971180145715
c
c Example 59c: Two-focal-length lens, through-focus scan

```

Jump to: [Commands](#), [Theory](#)

```

c
c This example illustrate a lens which has focal length f1=100 over the
c inner 18 cm aperture radius and f2=100.2 in the outer zone.
c
c Two focal spots are produced in the far-field. The zone command is used
c two extend the region of constant units.
c
c Variables
c -----
c row          number of row for beam 3
c first        logic switch for first pass
c dz           axial step
c lambda       wavelength
c f1,f2        two focal lengths associated with lens
c clap        division point between f1 and f2 focal lengths
c waves       number of wavelengths of aberration representing
c             difference of two focal lengths
c phase       piston error correction to have smooth transition from
c units       units in zone
c             f1 to f2
c
variab/dec/int row first          # declare integer variables
first = 1                        # set logic switch
dz = .015                        # axial step size
mem/set/b 5                      # set memory
lambda = .00106                  # wavelength variable
array/s 1 512                    # set array size
nbeam 2 data                     # make a second beam
units 0 .25                      # set units
wavelength/set 0 lambda*1e4      # set wavelength
nbeam 3 512 32 data              # make array for data
f1 = 100                         # set focal length 1
f2 = 100.2                      # set focal length 2
clap = 18                       # set size of inner region
waves = .5*(f1-f2)/f1/f2/lambda list # calculate wavefront error
                                # for radius = 1
abr/foc 2 waves rn=1             # set error due to focal length
                                # differences
phase = 360*waves*clap*clap      # set piston error correction
phase/piston 2 phase             # make piston error correction
obs/c 2 clap                     # eliminate inner region, beam 2
clap/c/c 1 clap                 # eliminate outer region, beam 1
add/coh/con 1 2                 # sum coherently
clap/c/c 1 24                   # clear aperture
title intensity
plot/watch ex59c_1.plt
plot/l 1 ns=64
title phase due to radius change
plot/watch ex59c_2.plt
plot/x 1
lens 1 f1                        # lens with f1=100
zone/fix f1 10.1*dz             # set through-focus zone
zone/in 1                       # beam 1 under zone control
dist f1-10*dz 1                 # propagate close to focus

```

Jump to: [Commands](#), [Theory](#)

```

row = 6                                # starting row number
clear 3 0                              # clear beam 3

title scan through focus

plot/watch plot1.plt

mac/def step/o
    copy/row 1 3 257 row                # copy center row of beam 1 to
                                         # specified row of beam 3
    row = row + 1                      # increment row number
    if first = 1 then                  # set units of beam 3
        variab/set/param units 1 units
        units 3 units dz
        first = 0
    endif
    plot/1 3 xrad=.05 yrad=16*dz ns=32
    dist dz 1
macro/end
macro step/27
plot/watch ex59c_3.plt
plot/1 3 xrad=.05 yrad=16*dz ns=32
end

```


Ex60: Optimization of size and location of an elliptical pinhole

Table. 60.1. Table of Ex60 examples

Ex60a: Simple optimization of astigmatic focus	1
Ex60b: Numerical check of optimization, scalar targets	3
Ex60c: Numerical check of optimization, array targets.	4
Ex60d: Reverse optimization of aperture shape, array targets, numerical check	5
Ex60e: Reverse optimization of aperture shape, array targets, built-in functions	5

Ex60a: Simple optimization of astigmatic focus

This example illustrates optimization in a very simple configuration. GLAD Pro is required. The system to be optimized is a spatial filter. There is a pinhole at the focal region of elliptical shape with the x- and y-radii variables. The location is also allowed to vary. Astigmatism is introduced into at the first lens and the objective of the optimization is to optimize simultaneously the Strehl ratio and the energy. The aperture will be moved and resized by the optimization process to roughly match the line foci caused by the astigmatism. The merit function is

$$\varphi = (SR - 1)^2 + (Energy - 1)^2 \quad (60.1)$$

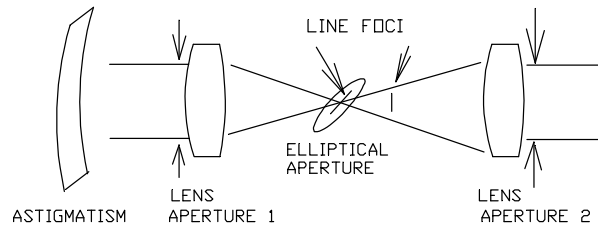


Fig. 60.1. Configuration to be optimized. The pinhole at the line focus is to be moved and its size changed in the x- and y-directions to simultaneously optimize the Strehl ratio and energy transmitted through Aperture 2.

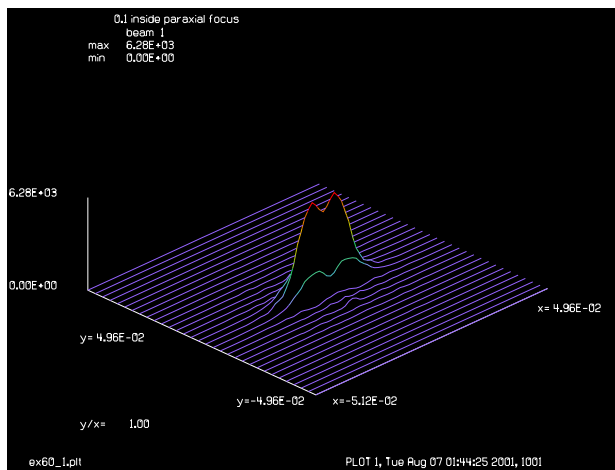


Fig. 60.2. Line focus 0.1 inside the paraxial focus.

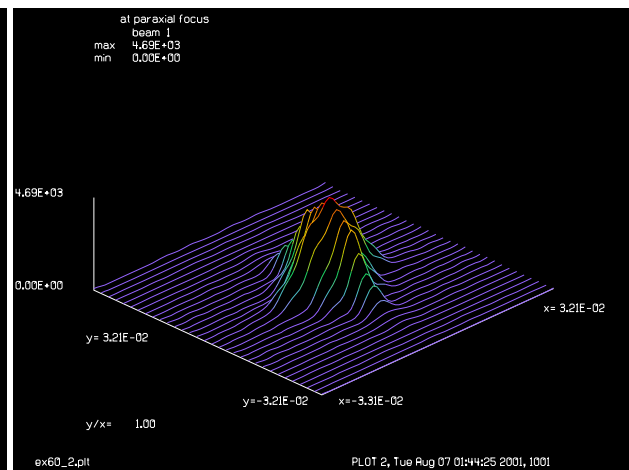


Fig. 60.3. Paraxial focus.

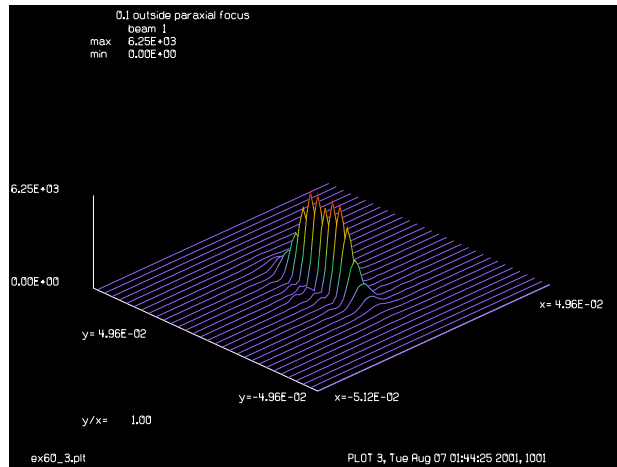


Fig. 60.4. Line focus 0.1 outside the paraxial focus.

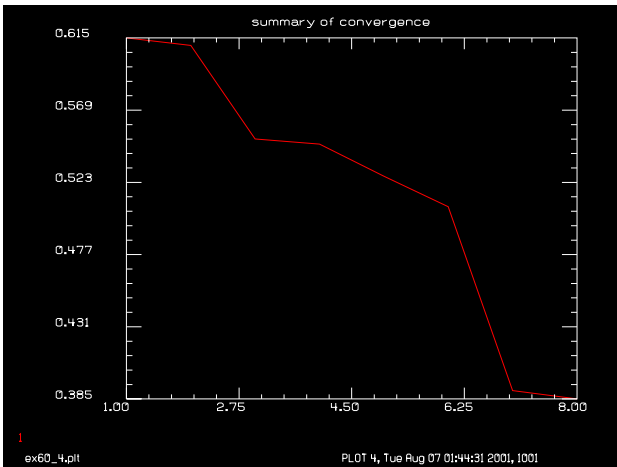


Fig. 60.5. Merit function change per optimization cycle.

Input: ex60a.inp

```

c## ex60a
c
c Example: Optimization
c
c This example illustrates simple optimization.
c
alias Name ex60a
set/alias_stop/off
variab/dec/int pass pass1
macro/def system/o
c
c system macro
c
pass1 = pass1 + 1 list
copy 2 1
dist var3 1
clap/e/c 1 var1 var2
dist 100-var3 1
clap/c/c 1 10
variab/set tar1 1 strehl
variab/set tar2 1 energy
macro/end
macro/def opt1/o
c
c the optimization macro
c
pass = pass + 1
x_save = var1
y_save = var2
opt/run
variab/set merit merit
udata/set pass pass merit x_save y_save
macro/end
# copy Beam 2 to 1 for fresh start
# propagate to pinhole
# elliptical pinhole
# propagate remaining distance to last lens
# aperture of Lens 2
# evaluate Strehl ratio and energy
# increment pass counter
# UDATA variable: x aperture size
# UDATA variable: y aperture size
# implement optimization cycle
# store merit function value
# store UDATA values

```

Jump to: [Commands](#), [Theory](#)

```

echo/on                      # set echo on for commands
array/s 1 64                 # define array size
nbeam 2                       # define two beams
clap/c/r 1 10                # initialize problem
abr/ast 1 1
abr/focus 1 -.5              # establish two separated line focii with
energy/norm 1 1              # astigmatism
lens 1 100
dist 99.9 1
title 0.1 inside paraxial focus
plot/watch @Name_1.plt       # plot paraxial focus
plot/l 1
dist .1 1
title at paraxial focus
plot/watch @Name_2.plt       # plot outside focus
plot/l 1
copy 1 2
dist .1 1
title 0.1 outside paraxial focus
plot/watch @Name_3.plt       # plot inside focus
plot/l 1
opt/nam system               # define macro name
var1 = .005                  # guess at initial radii for x and y
var2 = .005                  # to be .005
var3 = -.1                   # change to +.1 to start near other line focus
pass = 0                     # set udata counter
c          Reg      increment
opt/var/add var1   .002      # x radius variable, derivative increment .001
opt/var/add var2   .002      # y radius variable, derivative increment .001
opt/var/add var3   .01       # optimize with variable distance
c          Reg      Tar
opt/tar/add tar1    1         # target: strehl ratio
opt/tar/add tar2    1         # target: energy
opt/damp/mul 1      # set damping factor
opt/reject/on      # reject solutions if merit function increases
macro opt1/8        # run optimization macro 9 times
title summary of convergence
plot/watch @Name_4.plt
plot/udata min=0

```

Ex60b: Numerical check of optimization, scalar targets

The numerical methods for a simple optimization of scalar values are illustrated by working through the details of a simple optimization using the GLAD commands language. The result is compared with the GLAD optimization. See Ex60b.inp.

Input: `ex60b.inp`

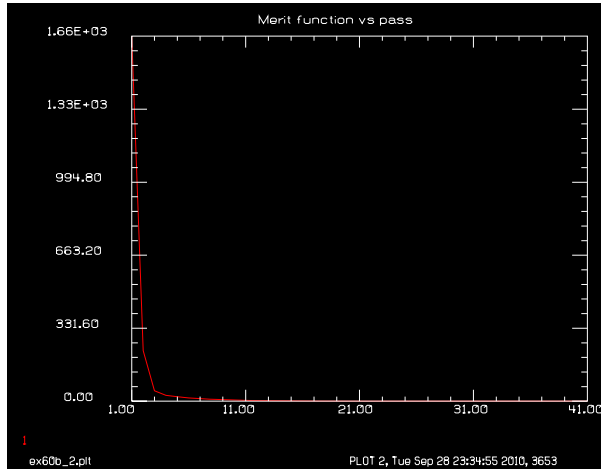


Fig. 60.6. Change of merit function vs. path number.

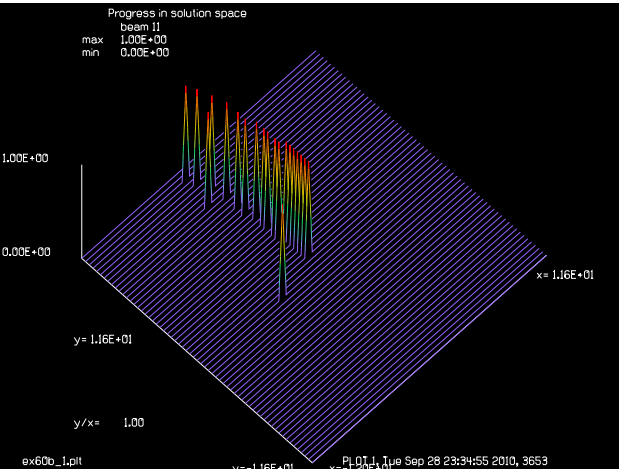


Fig. 60.7. Change in X- and Y-variables in solution space. Ultimately the X- and Y-variables converge to the desired values at the center of the figure.

Ex60c: Numerical check of optimization, array targets

The numerical methods for a simple optimization of array values are illustrated by working through the details of a simple optimization using the GLAD commands language. In this case the arrays are set to constant values so that they match the scalar problem of Ex60b.inp. The result is compared with the GLAD optimization. See Ex60c.inp.

Input: `ex60c.inp`

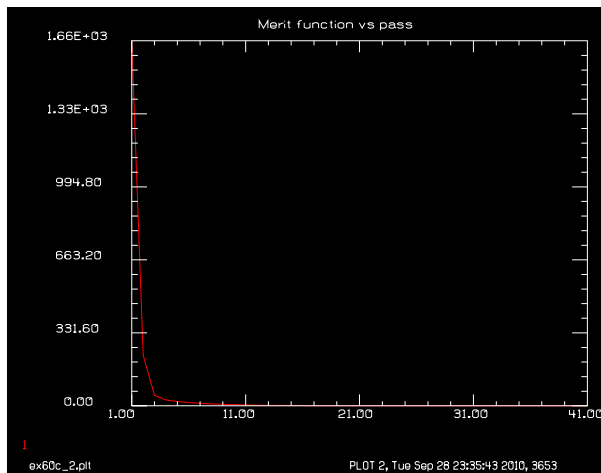


Fig. 60.8. Change of merit function vs. path number.

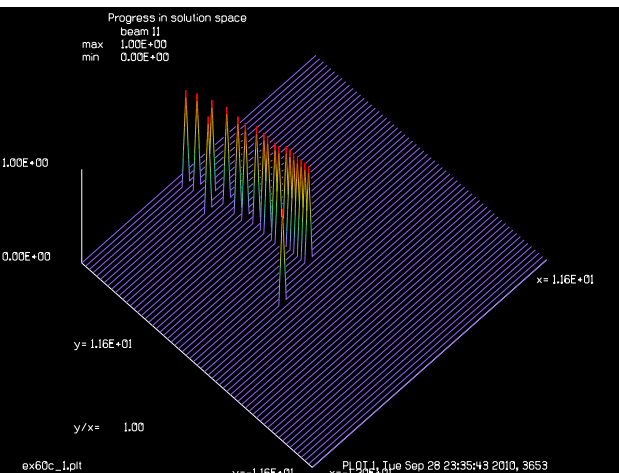


Fig. 60.9. Change in X- and Y-variables in solution space. Ultimately the X- and Y-variables converge to the desired values at the center of the figure.

Ex60d: Reverse optimization of aperture shape, array targets, numerical check

Optimization to recover actual aperture size using the far-field irradiance as a target array. The numerical methods for a simple optimization of scalar values are illustrated by working through the details of a simple optimization using the GLAD commands language. The result is compared with the GLAD optimization. See Ex60d.inp.

Input: ex60d.inp

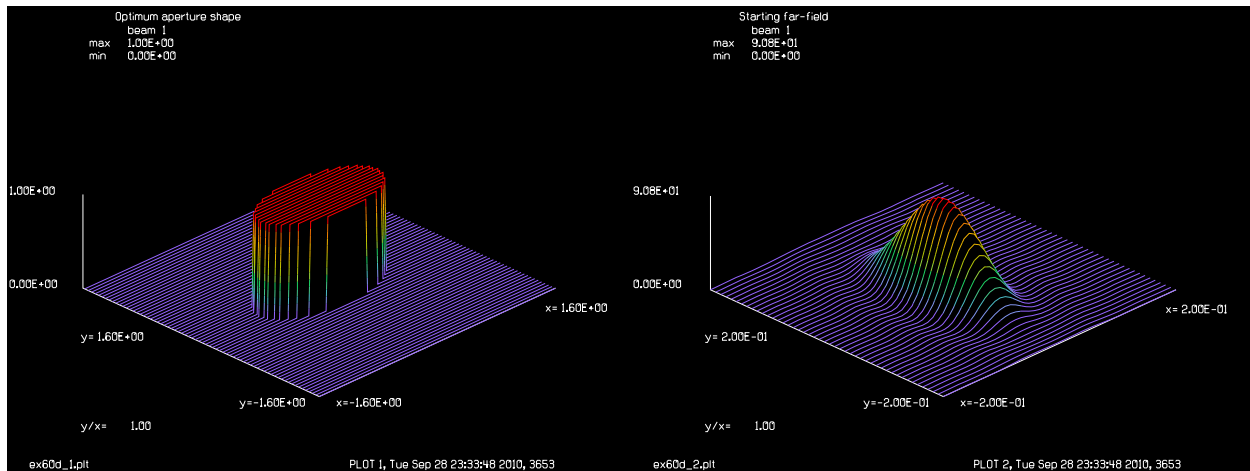


Fig. 60.10. Initial elliptical aperture.

Fig. 60.11. Far-field irradiance of initial elliptical aperture.

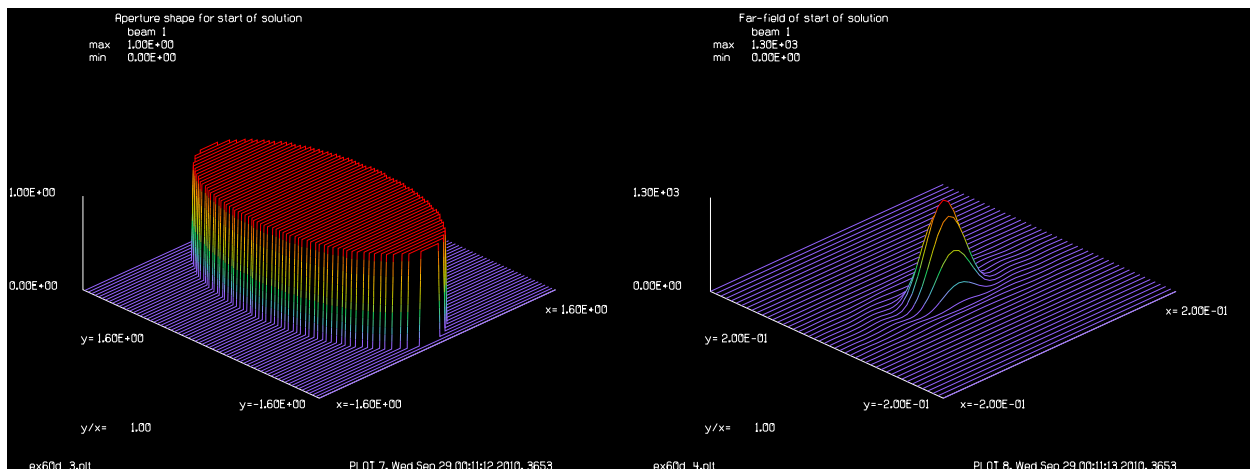


Fig. 60.12. Elliptical aperture used to start optimization.

Fig. 60.13. Far-field irradiance of elliptical aperture used to start optimization.

Ex60e: Reverse optimization of aperture shape, array targets, built-in functions

Optimization to recover actual aperture size using the far-field irradiance as a target array. The numerical methods for a simple optimization of scalar values are illustrated by working through the details of a simple optimization using the GLAD commands language. A single array target is sufficient to support the two performance variables consisting of the x- and y-widths of the elliptical aperture. The result is

Jump to: [Commands](#), [Theory](#)

compared with the GLAD optimization. The recovered aperture shape is within the pixel spacing and is effectively perfect. See Ex60e.inp.

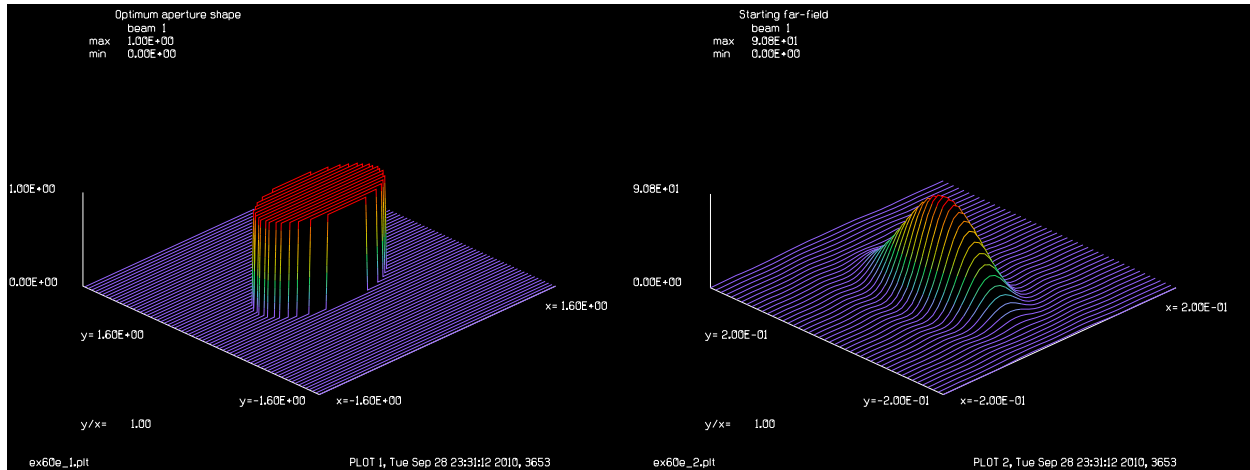


Fig. 60.14. Initial elliptical aperture. With $X = 0.8\text{cm}$ and $Y = 0.4\text{cm}$.

Fig. 60.15. Far-field irradiance of initial elliptical aperture with $X = 0.8\text{cm}$ and $Y = 0.4\text{cm}$.

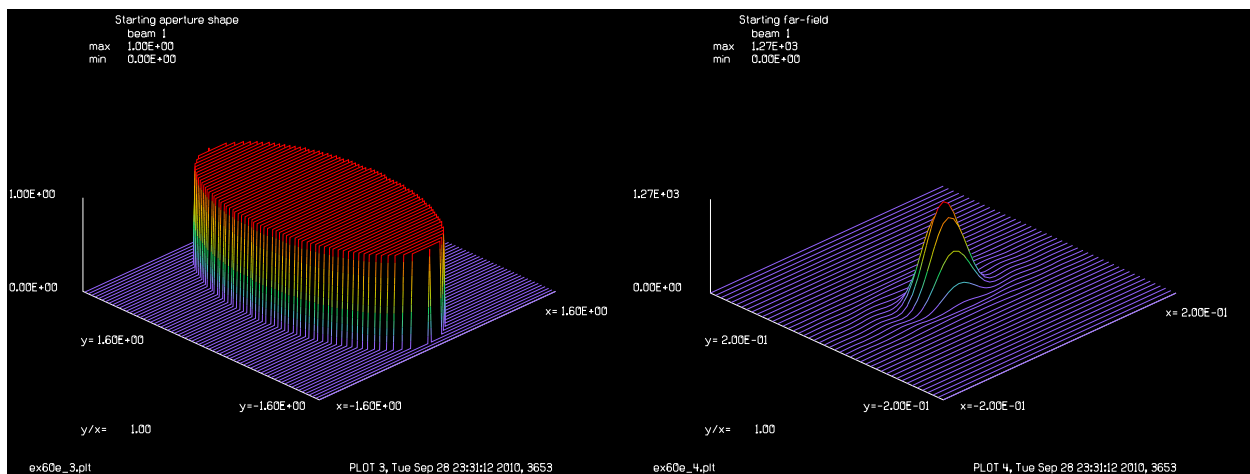


Fig. 60.16. Elliptical aperture used to start optimization with $X = 0.8\text{cm}$ and $Y = 1.5\text{cm}$.

Fig. 60.17. Far-field irradiance of elliptical aperture used to start optimization with $X = 0.8\text{cm}$ and $Y = 1.5\text{cm}$. The optimization varies the aperture widths until the far-field distribution matches that of Fig. 60.15.

Input: ex60e.inp

```
c## ex60e
c
c Example to show numerical details in a simple optimization
c using arrays having constant values to match ex60b.inp
c
c See GLAD Theory Manual, Chap. 16 for details.
c
alias Name ex60e
write/disk @Name.out/o
```

Jump to: [Commands](#), [Theory](#)

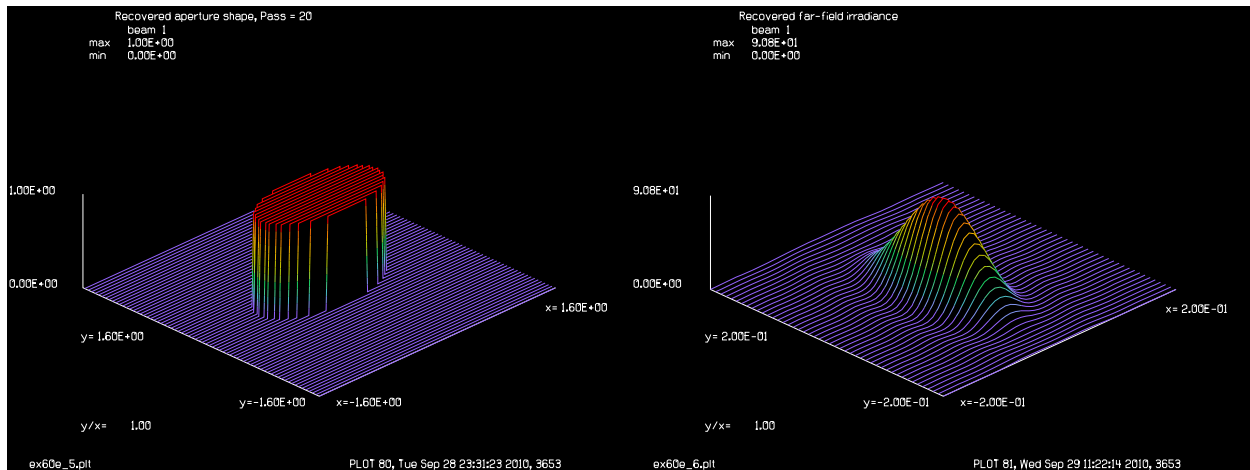


Fig. 60.18. Recovered aperture shape matches Fig. 60.14. Fig. 60.19. Recovered far-field irradiance matches Fig. 60.15.

```

C name: @Name
variable/dec/int Pass Nline FIRST Nline6 Ncx Ncy Ivar I J
Nline = 512
FIRST = 1
array/s 1 Nline
nbeam 5 data
array/variable/declare Value 1          # Propagating beam, complex amplitude
array/variable/declare Target 2         # Target for Value, intensity, t
array/variable/declare Work0 3          # error function, intensity, f0
array/variable/declare Work1 4          # Increment for variable 1, intensity, f1
array/variable/declare Work2 5          # Increment for variable 2, intensity, f2

array/list
Field = 6
units/field 0 Field
FocalLength = 100
Pass = -1
macro/def system
  array/reset Value
  unit/field Value Field
  clap/ell/no 1 X Y
  if [Pass== -1] then
    plot/w @Name_1.plt
    title Optimum aperture shape
    plot/l 1 xrad=1.6 ns=64
  endif
  if [Pass==0] then
    plot/w @Name_3.plt
    title Starting aperture shape
    plot/l 1 xrad=1.6 ns=64
  endif
  if [Pass>1] then
    plot/w @Name_5.plt
    title Recovered aperture shape, Pass = @Pass
    plot/l 1 xrad=1.6 ns=64
  endif

```

Jump to: [Commands](#), [Theory](#)

```

endif
lens Value FocalLength
prop FocalLength
C macro system: X = @X, Y = @Y
if FIRST then
  c
  c Set up units for all arrays in the far-field
  variab/set Units 1 units
  units/set 2:5 Units
  FIRST = 0
endif
macro/end
c
c Starting points to be recovered as optimum points.
c This is an arbitrary choice.
c
X_optimum = 0.8
Y_optimum = 0.4

X = X_optimum
Y = Y_optimum
clear 0 0
macro/run system
plot/w @Name_2.plt
title Starting far-field
plot/l 1 1 xrad=.2
copy/con Value Target
Xmdamp = 1
Pass = 0
X_start = .8
Y_start = 1.5
X = X_start
Y = Y_start
macro/run system
plot/w @Name_4.plt
title Starting far-field
plot/l 1 1 xrad=.2
udata/clabel 'Pass' 1
udata/clabel 'Merit' 2
udata/clabel 'X' 3
udata/clabel 'Y' 4
DvX = .02
DvY = .02
Wt1 = 1 # will be scaled for array size below
Wt1_Nline = Wt1/Nline

opt/name system
opt/damp/mul Xmdamp
opt/var/add X DvX
opt/var/add Y DvY
opt/tar/add/intensity Value Target Wt1_Nline Work0 Work1 Work2
opt/tar/list
variab/set Merit merit
Pass = 1

```

Jump to: [Commands](#), [Theory](#)


```
udata/set Pass Pass Merit X Y
macro/def Opt
  Pass = Pass + 1 list
  opt/run
  variab/set Merit merit
  udata/set Pass Pass Merit X Y
macro/end
macro Opt/1
  udata
  C
  C Compare with Ex60d.inp
  C
  pause/label `Compare with Ex60d.inp
  macro Opt/18
  udata
  plot/w @Name_6.plt
  title Recovered far-field irradiance
  plot/1 1 xrad=.2
```


Ex61: Optimization for accelerated mode estimation

This example illustrates the use of optimization to find the steady-state gaussian waist size for the resonator described in Example 33. For this simple resonator, the waist size can be calculated exactly. This example shows how the optimization is implemented, and could be used for more sophisticated resonators where the answer is not readily apparent. In the more general case, where the mode is not necessarily a gaussian, this optimization procedure could be used to find a best-fit gaussian to be used as a seed. The GLAD Pro is required for the optimization features.

Table. 61.1. Parameters for resonator with hole outcoupling, i.e., central obscurations.

length	45 cm
mirror radius	50 cm
wavelength	1.064 microns
Rayleigh range	15 cm
waist radius	0.02253936

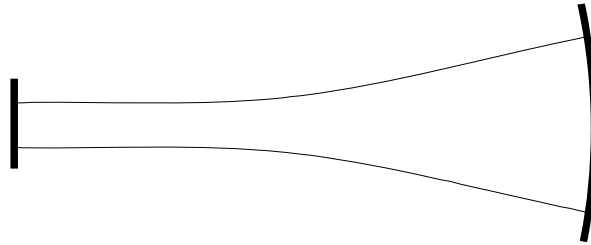


Fig. 61.1. Stable resonator configuration. The waist will form on the flat mirror and the phase radius will match the radius of the concave mirror in the ideal geometric mode.

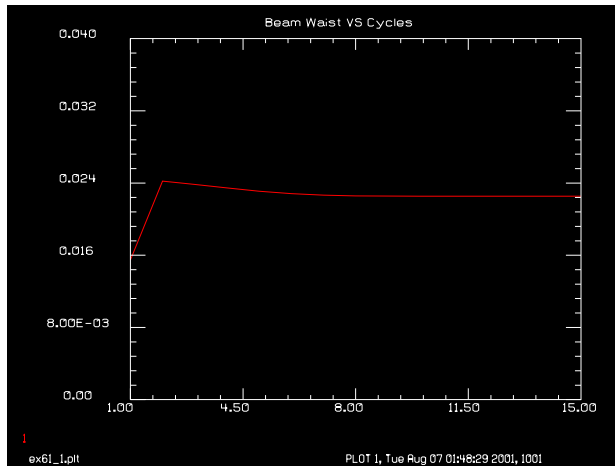


Fig. 61.2. Beam Waist vs. optimization cycles. The expected value of 0.0225 cm is quickly achieved. Note that the waist parameter is underestimated slightly initially. In more extreme cases the damping could be increased, but it is not required here.

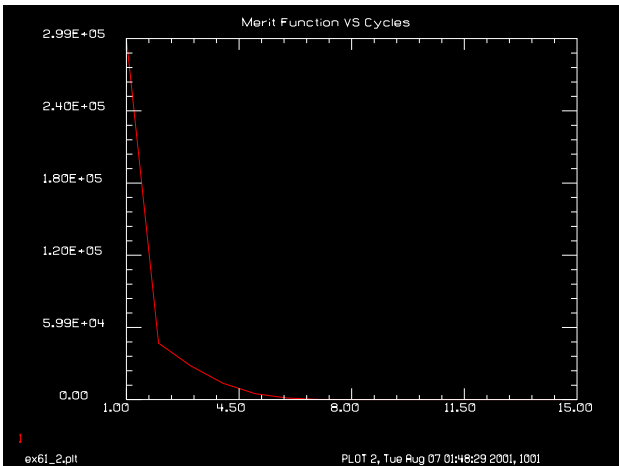


Fig. 61.3. Merit Function vs. optimization cycles. The merit function here is the square of the difference between successive peak intensities in the seed beam. In only three attempts the merit function is essentially zero. For rough seed estimation, a single cycle would probably be sufficient.

Input: ex61.inp

```

c## ex61!16749978392057
c
c   Example 61: Bare Cavity Stable Resonator Gaussian Mode Optimization
c
c   This example works for GLAD/OPTION
c
c   Contributed by Kenneth Moore
c
c   This example uses the difference between successive peak intensities as
c   a criterion for establishing mode convergence in the plano-concave
c   resonator of Example 33. The parameter to be optimized is the seed
c   gaussian waist size. This example shows how optimization can be used
c   to accelerate mode convergence. Although for this particular example
c   the answer is readily calculable, the method is general.
c
c   Variable          Application
c   pass              Loop Counter
c   waist             Gaussian beam waist
c   peak_diff         Change in successive peak intensities
c   merit             Merit function for UDATA
c   done              set to 1 if optimization done.
c
variab/dec/int pass done
macro/def cavity/o
  write/off
c
c   This is the system macro that describes the laser cavity.
c   There are two successive propagations to calculate the change in
c   the peak intensities.
c
  units/s 0 .005                # redefine units
  gaus/c/c 1 1 waist            # form a gaussian with beam
  zbound/s 1 waistx=.03 zwaistx=0. # force surrogate gaussian to be
                                   # constant for each pass
  energy/norm 1                 # normalize energy
  prop 45                       # propagate 45 cm.
  mirror/sph 0 -50              # mirror of 50 cm. radius
  prop 45                       # propagate back to center
  mirror/sph 0 1.e15
  variab/set      peak1 1 peak    # store first peak value
c
c   now propagate beam through again and calculate change.
c
  prop 45                       # propagate 45 cm.
  mirror/sph 0 -50              # mirror of 50 cm. radius
  prop 45                       # propagate back to center
  mirror/sph 0 1.e15
  variab/set peak2 1 peak        # store second peak value
  peak_diff = peak2 - peak1
  write/on
macro/end
c

```

Jump to: [Commands](#), [Theory](#)

```

macro/def optimize/o
c
c This is the macro that actually performs the optimization.
c
    pass = pass + 1                # increment pass counter
    opt/run                        # implement optimization cycle
    variab/set done optdone        # check optimization finished
    variab/set merit merit         # store merit function value
    udata/set pass pass waist peak_diff merit
    # store UDATA values: counter, counter, beam waist, energy, merit
    if done > 0 then
        write/on
        mac/exit
    endif
macro/end
c
c All the macros are defined, now initialize system
c
echo/on                          # set echo on for commands
array/s 1 64                     # define array size
wavelength/set 0 1.064           # define wavelength
units/s 0 .005                   # define units
opt/nam cavity                   # define the macro name
waist = .04                      # initial guess for beam waist
pass = 0                         # initialize udata counter
c          Reg          increment
opt/var/add waist          .0002    # variable: beam waist
c          Reg          Tar
opt/tar/add peak_diff 0          # target : 0 ,
                                # (no change on successive passes)
c          Reg      Con      Type    # constraint: waist greater 0
opt/con/add waist 0      greater
macro optimize/15              # run optimization macro 5 times
c
c now plot data to see if we made any progress.
c
title Beam Waist VS Cycles
plot/watch ex61_1.plt
plot/udata 1 min=0. max=.04
title Merit Function VS Cycles
plot/watch ex61_2.plt
plot/udata 3
plot/screen/pause 3
end

```


Ex62: Linear grating with a small defect

Table. 62.1. Table of Ex62 examples

Ex62a: Square wave gratings, effect of small obscuration	3
Ex62b: Square wave gratings, effect of second grating.	6

This example illustrates calculations of the performance of two diffraction gratings separated by the Talbot distance. In addition, a small defect is included. The defect is representative of “metalization” error.

A linear grating of period 10μ imposed on a collimated beam. The grating is assumed to have an aperture of $10\mu \times 10\mu$. An obscuration of $1.5\mu \times 5\mu$ is used to illustrate an opaque defect. A defect of only a few microns is not directly visible because the scattering angle is large and the total energy effected is small. The subtraction method illustrated in Example 62a can display the effects of extremely small defects.

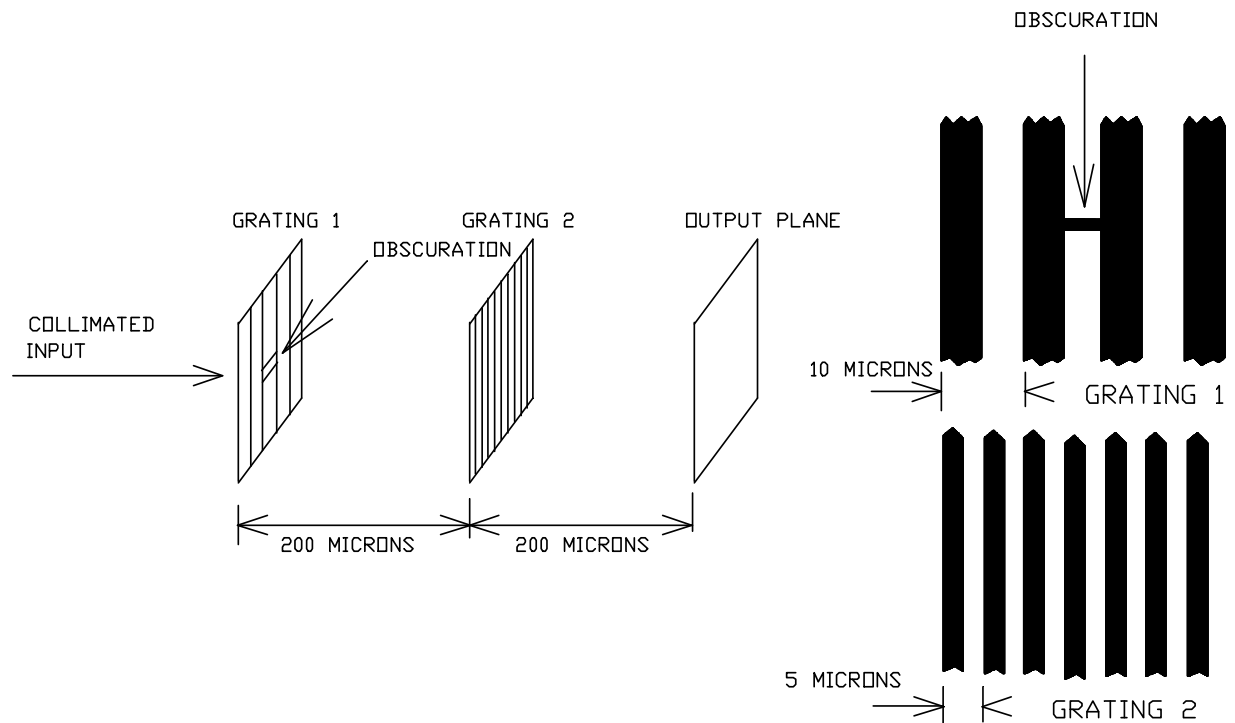


Fig. 62.1. Optical configuration for Ex. 62. The initial grating is placed one-half Talbot cycle from the second grating and has a small square obscuration on one clear grating line. The second grating is of half the period of the first grating. The small obscuration scatters a small amount of light at a large angle.

Table. 62.2. Parameters of the grating configuration.

wavelength	0.5μ
grating 1 period	10μ
grating 2 period	5μ , offset 1.5μ
aperture of gratings 1 and 2	$60\mu \times 60\mu$
obscuration	$1.5\mu \times 5\mu$
propagation to grating 2	200μ
propagation to final plane	200μ
array size	1024×1024

The grating is illustrated in Fig. 62.1. The second grating is separated from the first by 200μ and the final observation plane is separated from the second grating by 200μ . The Talbot cycle for self-imaging is,

$$\text{Talbot length} = \frac{2T^2}{\lambda} \approx 400\mu \quad (62.1)$$

where T is the period and λ is the wavelength. A length of 200μ is a half-cycle, which results in a reimaging of the original distribution with a 180° shift in the linear grating modulation. Figure 62.2 shows the initial grating with no obscuration, note the large guardband to allow for wide-angle scattering. Figure 62.3 shows the initial distribution after 200μ propagation. The grating is reasonably well reimaged and is shifted by 180° , as shown in Fig. 62.4.

The obscuration blocks part of one of the grating lines. Its width was selected to be a $1.5\mu \times 5\mu$ rectangle, as shown in a close up view in Fig. 62.5. This small obscuration introduces a wide but very low amplitude change in the diffraction pattern after 200μ propagation. The close up view of the diffraction pattern of the grating with the obscuration in Fig. 62.7 shows no perceptible change from Fig. 62.6 which is from the grating with no obscuring defect. We can observe the effect by subtracting the complex amplitudes of the two diffraction patterns: one with the defect and one without. Both were propagated 200μ . The difference of the two beams shows the diffraction pattern associated with the small square obscuration, as shown in Fig. 62.8 at $25\mu \times 15\mu$ scale and in Fig. 62.9 at $100\mu \times 100\mu$. The effect of the single tiny defect is evident as a very wide angle diffraction pattern—the same pattern as from a diffracting aperture of the same size. The diffraction pattern shows the characteristics of a rectangular aperture of the same size. This derives from Babinet's Principle. The difference distribution is lower in intensity by about 2 orders of magnitude. The remainder of the calculation for the second grating is essentially unaffected by the obscuration.

A second grating is imposed after the first 200μ propagation. Figure 62.10 shows the outline of the aperture of the second grating and Fig. 62.11 shows the profile after the second grating. The second grating is of twice the period of the first grating but aligned on the Talbot image of the first grating so that the principle effect simply to reduce the duty cycle of the 10μ grating. Because the Talbot imaging is not perfect, the second grating allows some small amount of light at the period of 5μ . The final observation plane is 200μ from the second grating. The final pattern is a Talbot image with a strong frequency component at the period 10μ so we see the half-period shift in the peak, as shown in Fig. 62.12, similar to Fig. 62.4. Sampling errors are evident. The second grating is diffracting at a relatively large angle, as shown in Fig. 62.13. Parts of the complex amplitude which diffract to the boundaries of the computer array scatter back into the array as aliasing errors. Aliasing errors affect the center parts of the beam where the signal is highest much more than the edges of the computer array where the signal is low.

The sampling requirements of this problem are severe. We wish to resolve the 5μ obscuration, leading to a choice of matrix units of 0.24μ . We elect to start with a 60μ width of the grating and enclose it a computer array of about 246μ — a total of 1024 computer points. If we allow a smaller guard band, aliasing errors will become unacceptably large. If we did not want to resolve the obscuration we could use a one-dimensional array of 1024 points, which would execute very quickly. To resolve the square obscuration, we have used the same sampling in both directions leading to 1024×1024 .

Ex62a: Square wave gratings, effect of small obscuration

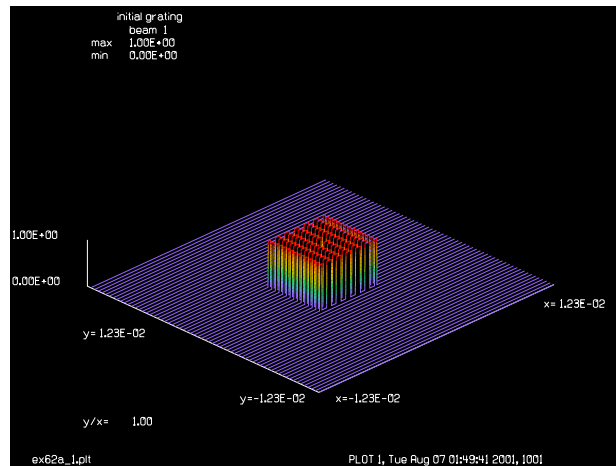


Fig. 62.2. Intensity distribution for grating. The period of the grating is 10μ and a square region of $246\mu \times 246\mu$ is impeded in a large array to give a generous guardband. view.

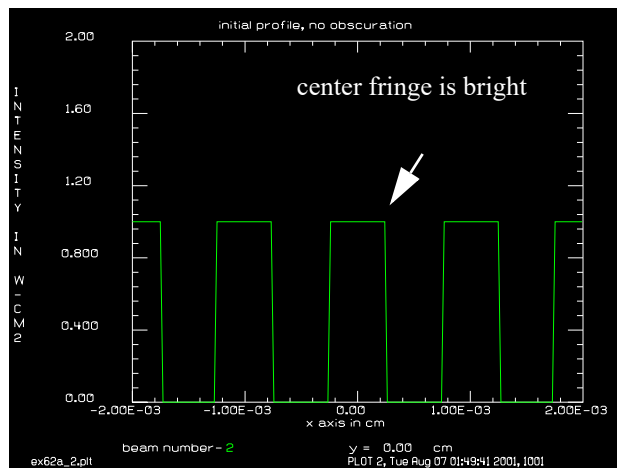


Fig. 62.3. Initial grating profile with no obscuration. 40μ of the grating are displayed.

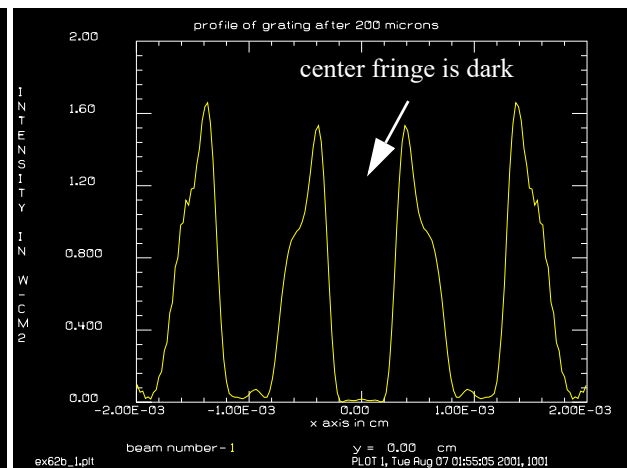


Fig. 62.4. Unobscured grating profile after 200μ propagation. Note shift of half-cycle. 40μ of the grating are displayed.

Input: `ex62a.inp`

```

c## ex62a
c
c Example 62a: Square wave gratings, effect of small obscuration
c
echo/on
mem/set/b 8 # try to get 8 megabytes
nbeam 2
array/s 0 1024 1024 # define array size
wavelength/set 1 .5 # define wavelength in microns
units/s 1 .24e-4 .24e-4 # define spacing of matrix points

```

Jump to: [Commands](#), [Theory](#)

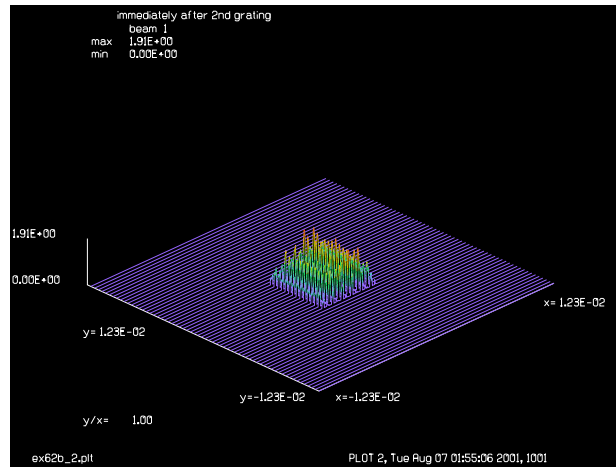


Fig. 62.5. Close up view of region with obscuration before propagation. Obscuration is $1.5\mu \times 5\mu$ rectangle. $25\mu \times 25\mu$ view

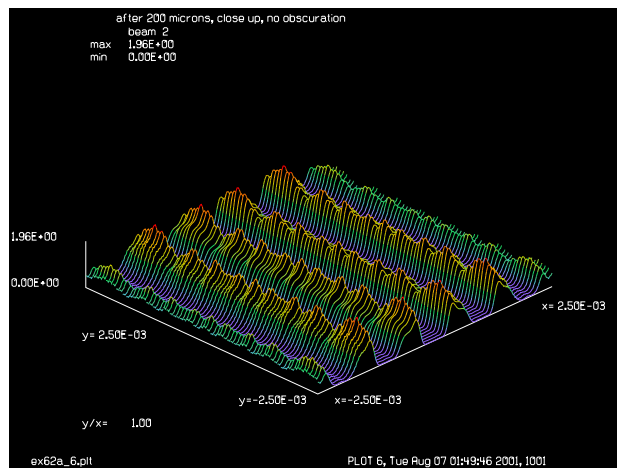


Fig. 62.6. Close up view of region with obscuration after 200μ propagation. It is difficult to see a difference. $25\mu \times 25\mu$ view.

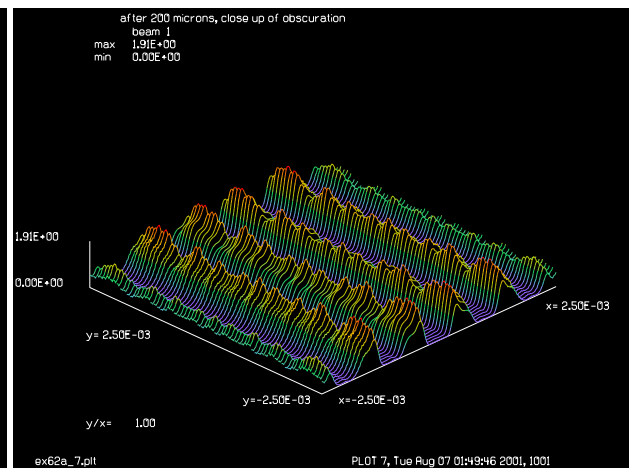


Fig. 62.7. Close up view after 200μ propagation. This is a Talbot image of the first grating, $25\mu \times 25\mu$ view. The effect of the obscuration can not be detected — it is too faint.

```

grat/s 1 pe=10e-4           # grating, G1
clap/s/c 1 30e-4           # 60 x 60 square aperture
copy 1 2                   # Beam 2 is unobscured
obs/rec 1 2.5e-4 .75e-4    # Beam 1 has obscuration
title initial grating
plot/watch ex62a_1.plt
plot/l 1 h=.2 ns=64
title initial profile, no obscuration
plot/watch ex62a_2.plt
plot/x/i 2 0 -20e-4 20e-4 fmax=2.
title initial close up of obscuration
plot/watch ex62a_3.plt
plot/l 1 xrad=25e-4 h=.2 ns=64
dist 200e-4                # 200 micron propagation
title after 200 microns, no obscuration

```

Jump to: [Commands](#), [Theory](#)

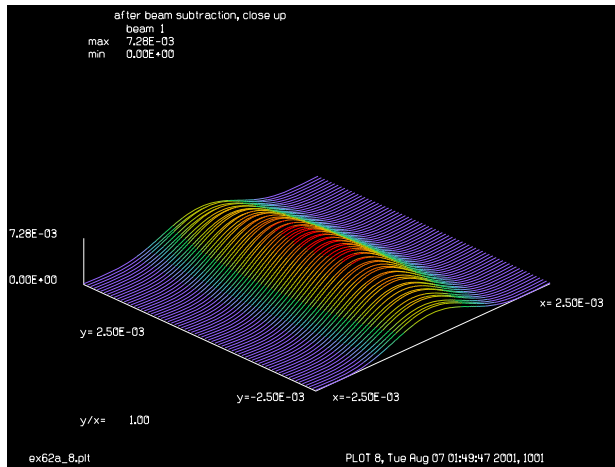


Fig. 62.8. After subtracting the diffraction pattern of a beam with no obscuration from the beam with the obscuration. The wide distribution is down by 2 orders of magnitude. $25\mu \times 25\mu$ view.

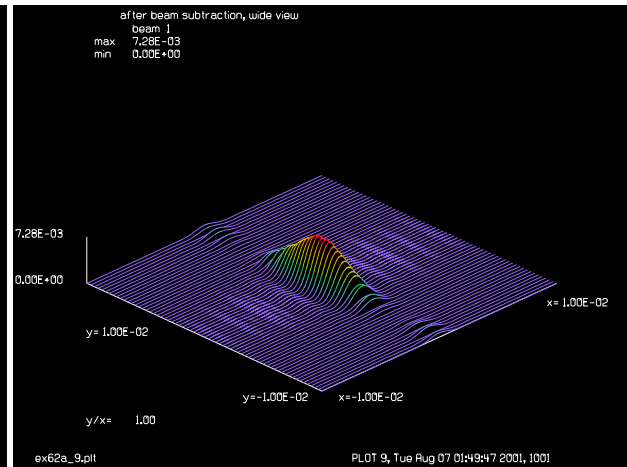


Fig. 62.9. Same distribution as Fig. 62.8 except shown in $100\mu \times 100\mu$ view.

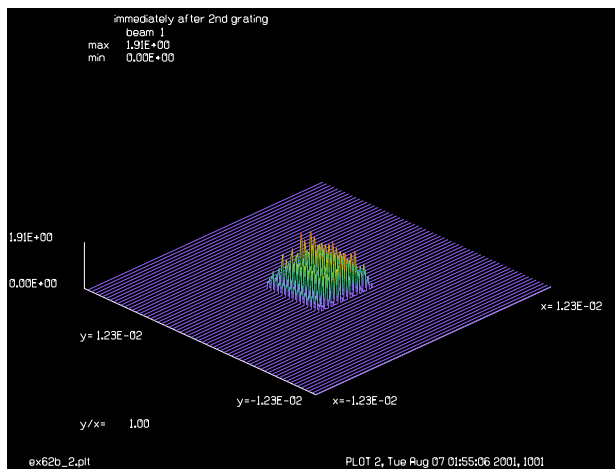


Fig. 62.10. Distribution after second grating. $246\mu \times 246\mu$ view.

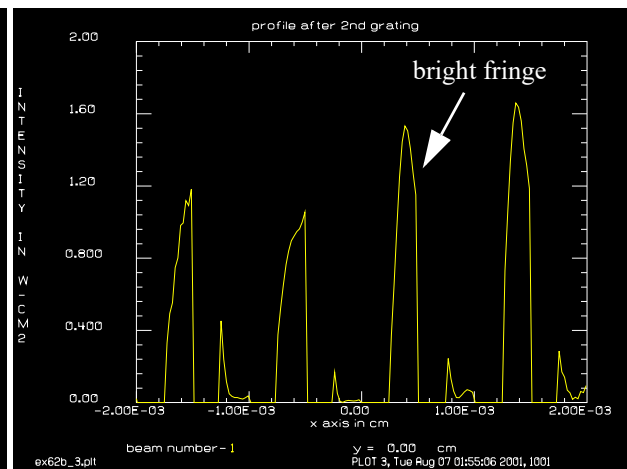


Fig. 62.11. Final distribution showing wide angle scattering. This was the reason for the choice of a very large guardband. $246\mu \times 246\mu$.

```
plot/watch ex62a_4.plt
plot/l 2 h=.2 ns=64
title profile after 200 microns, no obscuration
plot/watch ex62a_5.plt
plot/x/i 2 0 -20e-4 20e-4 fmax=2.
title after 200 microns, close up, no obscuration
plot/watch ex62a_6.plt
plot/l 2 xrad=25e-4 h=.2 ns=64
title after 200 microns, close up of obscuration
plot/watch ex62a_7.plt
plot/l 1 xrad=25e-4 h=.2 ns=64
phase/piston 2 180
add/coh 1 2
```

Jump to: [Commands](#), [Theory](#)

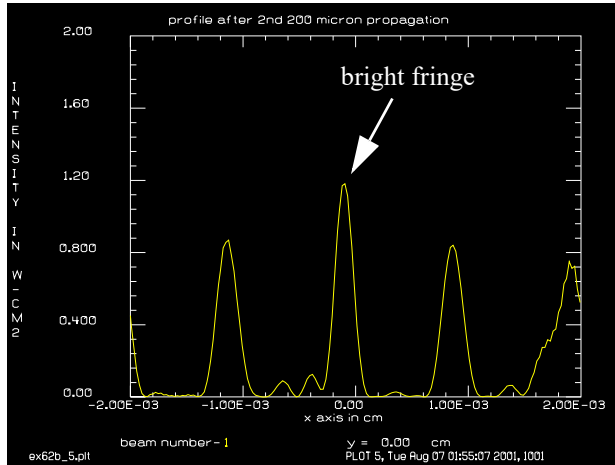


Fig. 62.12. Diffraction profile after second 200 μ propagation. This is a Talbot image of Fig. 62.11. Note again the half-cycle shift of the strong $T = 10\mu$ component.

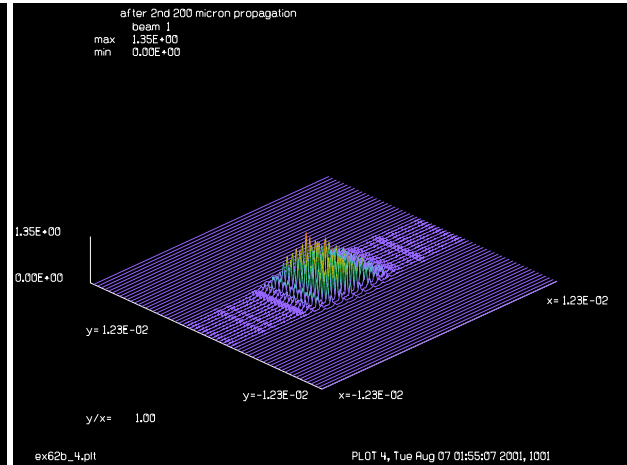


Fig. 62.13. Final distribution showing wide angle scattering. This was the reason for the choice of a very large guardband. $246\mu \times 246\mu$ view.

```
title after beam subtraction, close up
plot/watch ex62a_8.plt
plot/1 1 xrad=25e-4 h=.2 ns=64
title after beam subtraction, wide view
plot/watch ex62a_9.plt
plot/1 1 xrad=100e-4 h=.2 ns=64
end
```

Ex62b: Square wave gratings, effect of second grating

Input: ex62b.inp

```
c## ex62b
c
c Example 62b: Square wave gratings, effect of second grating
c
echo/on
mem/set/b 8                                # request 8 megabytes of memory
array/s 1 1024 1024                        # define array size
wavelength/set 1 .5                         # define wavelength in microns
units/s 1 .24e-4 .24e-4                   # define spacing of matrix points
grat/s 1 pe=10e-4                           # grating, G1
obs/rec 1 2.5e-4 .75e-4
clap/s/c 1 30e-4                            # 60 x 60 square aperture
dist 200e-4                                # 200 micron propagation
title profile of grating after 200 microns
plot/watch ex62b_1.plt
plot/x/i 1 0 -20e-4 20e-4 fmax=2.
grat/s 1 pe=5e-4 xd=-1.25e-4
clap/s/c 1 30e-4
title immediately after 2nd grating
plot/watch ex62b_2.plt
```

Jump to: [Commands](#), [Theory](#)

```
plot/1 ns=64 h=.2
title profile after 2nd grating
plot/watch ex62b_3.plt
plot/x/i 1 0 -20e-4 20e-4 fmax=2.
dist 200e-4                                # 2nd 200 micron propagation
title after 2nd 200 micron propagation
plot/watch ex62b_4.plt
plot/1 ns=64 h=.2
title profile after 2nd 200 micron propagation
plot/watch ex62b_5.plt
plot/x/i 1 0 -20e-4 20e-4 fmax=2.
end
```


Ex63: Comparison of Beer's Law and CO₂ gain

This example compares Beer's Law gain and Frantz-Nodvik gain for pulsed gain. In general, the Frantz-Nodvik gain saturates less at small irradiance values and more at high irradiance values. In the example the small signal gain and saturation irradiance are set to be the same for both types of gain functions.

The example uses character registers to set the command name and the `udata` parameter to be saved.

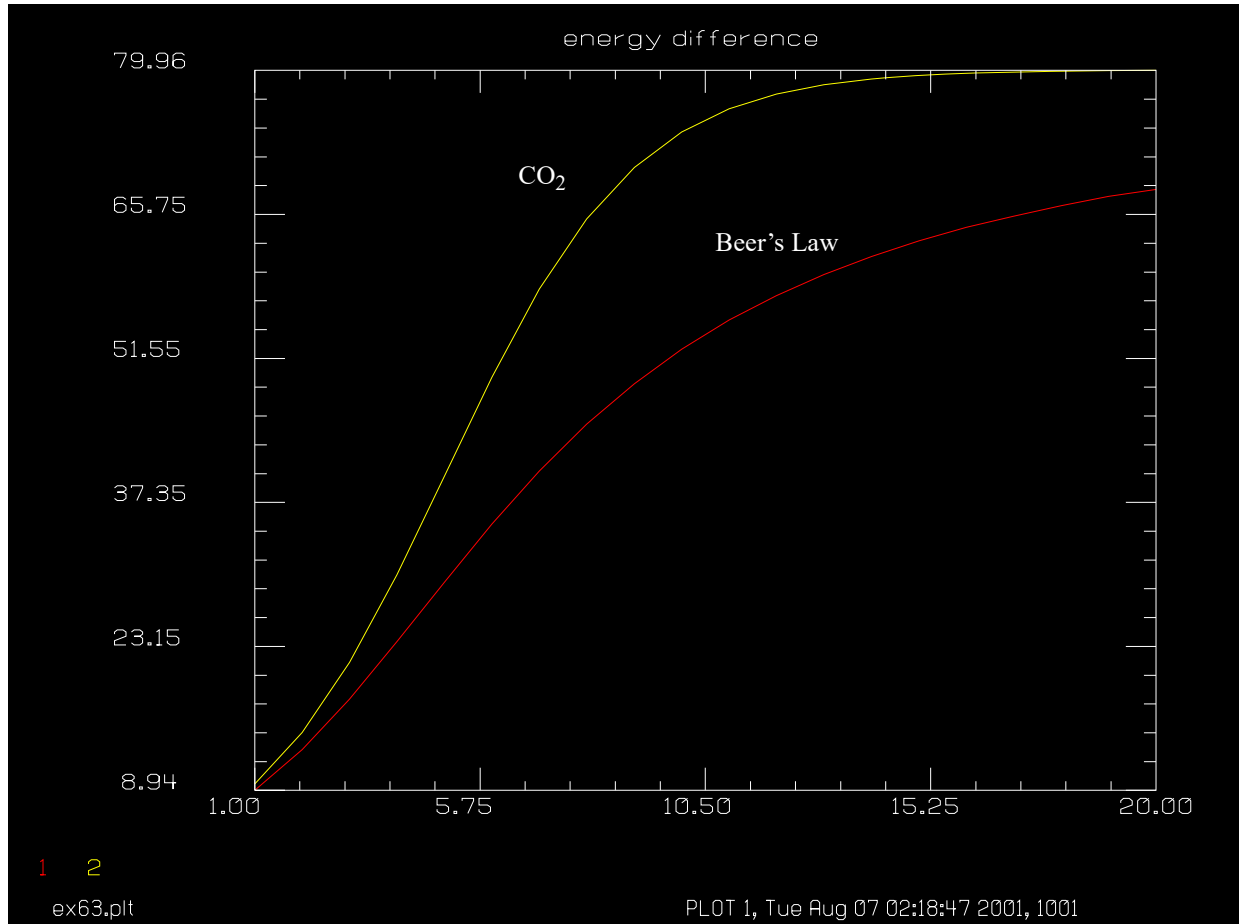


Fig. 63.1. Comparison of Beer's Law and pulsed gain using Frantz-Nodvik theory.

Input: ex63.inp

```
c## ex63
c
c Example 63: Comparison of Beer's Law and CO2 gain
c
c The CO2 gain function has different saturation behavior.
c
array/s 1 4
macro/def ex63a/o
c
c loop macro
c
```

```
pass = pass + 1
@cmd/prop 1 5                                # use alias to set function name
variab/set new_energy energy
diff_energy = new_energy - old_energy
udata/set pass pass @y=diff_energy          # set udata value, y1 or y2
old_energy = new_energy
macro/end
macro/def ex63b/o
c
c  initialization and execution macro
c
  clear 1 1
  pass = 0
  variab/set old_energy energy
  @cmd/set g0=.1 es=10                        # initialize gain variables
  macro/run ex63a/20                          # execute loop macro
macro/end
c
c  run beer gain
c
alias cmd beer                               # set alias to 'beer'
alias y y01                                  # set alias to 'y01' for udata
macro/run ex63b
c
c  run co2 gain
c
alias cmd co2                               # set character data to 'co2'
alias y y02                                  # set character data to 'y02' for udata
macro/run ex63b
title energy difference
plot/watch ex63.plt
plot/udata first=1 last=2
end
```


Ex64: Lens array using single apertures

This example shows illustrates construction of a lenslet array. A lenslet array of 4 x 4 elements is focused by a lens of 100 cm focal length. Each lens element has a focal length of 2500 cm.

Table. 64.1. Parameters.

array elements	4×4 lenses, 1.2 cm diameter, $f = 2500$ cm
wavelength	1 micron
focusing lens	$f = 100$ cm

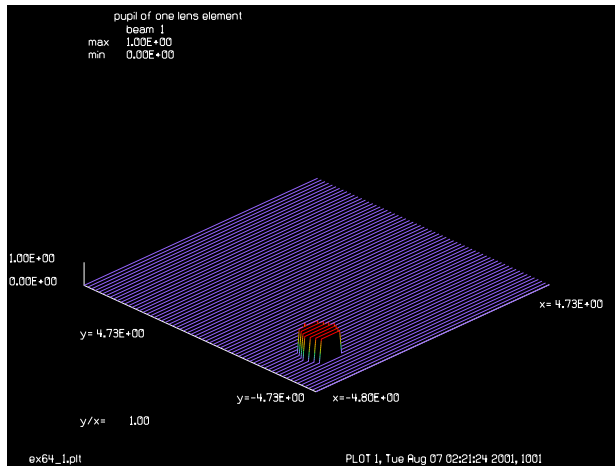


Fig. 64.1. Irradiance distribution of single lenslet.

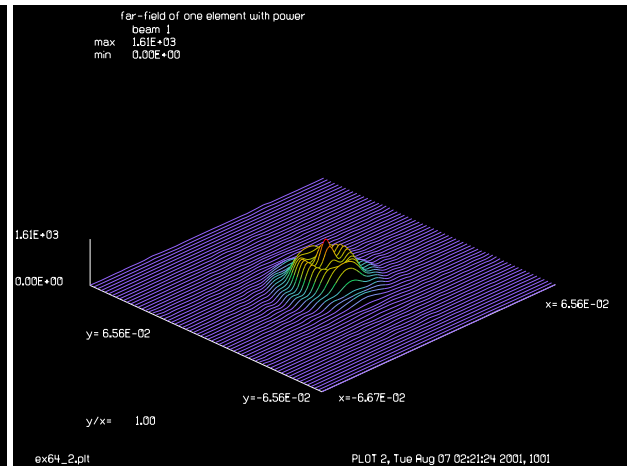


Fig. 64.2. Far-field of single lenslet with focal length of 2500 cm.

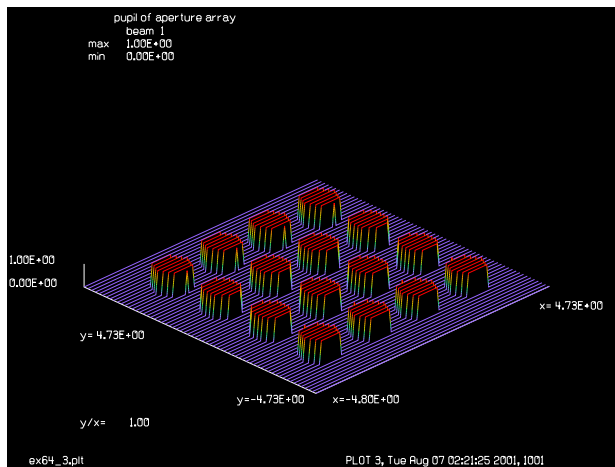


Fig. 64.3. Irradiance distribution of lenslet array.

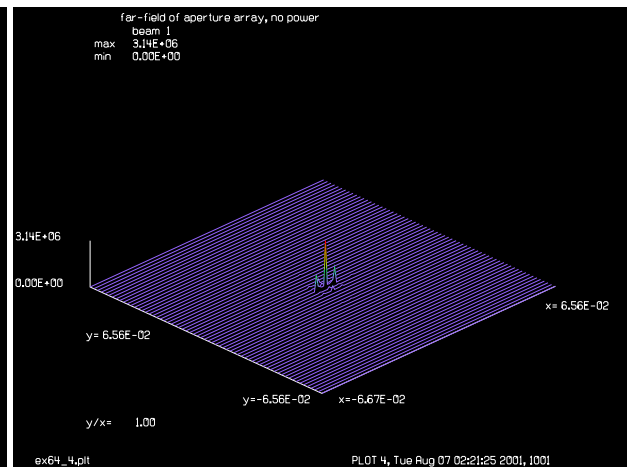


Fig. 64.4. Far-field of lenslet array with no optical power in the lenses.

Input: `ex64.inp`

```
c## ex64
c
c Example 64: Lens Array
c
```

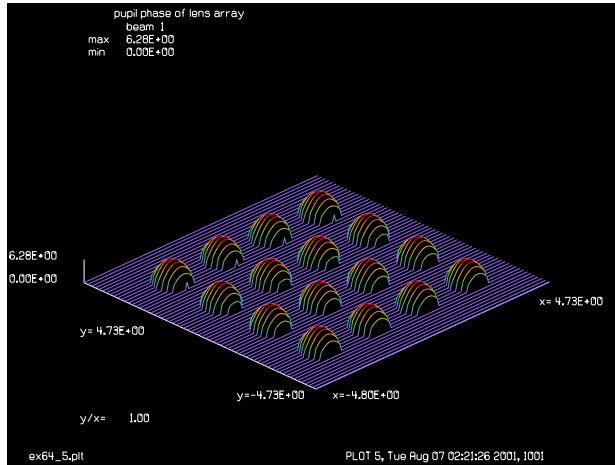


Fig. 64.5. Phase distribution of lenslet array. Note artifacts of the phase plotting where the intensity is zero.

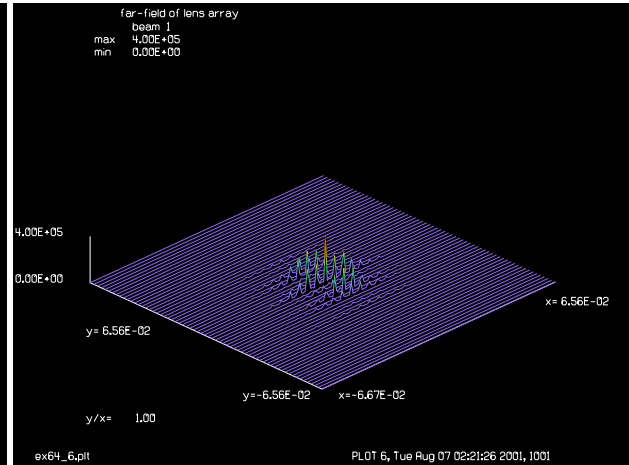


Fig. 64.6. Far-field of lenslet array, each element has a focal length of 2500 cm.

```

c This example illustrates creation of a lenslet array.
c 16 lenslets are made each with a focal length of 2500 cm.
c The array is created by using two macros to create a loop
c for the y- and x-directions.
c
set/alias_stop/off
variab/dec/int n_times
c
c Define inner loop for aperture creation
c
macro/def inner/o
    x_dec = x_dec + 2.0                # increment x-decenter
    clear 2 1                          # initialize beam 2
    clap/cir/con 2 .6 x_dec y_dec
    abr/focus 2 werr rn=.5 x=x_dec y=y_dec # define lens of focal length
    add/coh/con 1 2                    # combine beams
macro/end
c
c Define outer loop for aperture creation
c
macro/def outer/o
    y_dec = y_dec + 2.0                # increment y-decenter
    x_dec = -5.0                      # initialize x-decenter
    macro/run inner/n_times
macro/end
nbeam 2                               # expand the number of beams
array/set 0 128                       # initialize the arrays
units/s 0 .075                        # set units to .075
clear 1 0                             # set entire field to unity
wavelength/set 0 1                    # wavelength 1 micron
echo/on
c
c calculate with single lenslet
c
n_times = 1                           # set number of elements per side

```

Jump to: [Commands](#), [Theory](#)

```

y_dec = -5.0                # initialize y-decenter
werr = 0.5                  # f=2500 cm
macro/run outer/n_times
title pupil of one lens element
plot/watch ex64_1.plt
plot/l/intensity 1 1 ns=64 h=.1      # plot pupil intensity
lens 1 100
dist 100 1
title far-field of one element with power
plot/watch ex64_2.plt
plot/l/intensity 1 1 ns=64 h=.2      # plot far-field intensity
c
c  calculate with all elements but no power
c
zreff/se 1 0
array/s 1 128
units/s 0 .075              # set units to .075
clear 1 0                   # set entire field to unity
n_times = 4
y_dec = -5.0                # initialize y-decenter
werr = 0
macro/run outer/n_times
title pupil of aperture array
plot/watch ex64_3.plt
plot/l/intensity 1 1 ns=64 h=.1      # plot pupil intensity
lens 1 100
dist 100 1
title far-field of aperture array, no power
plot/watch ex64_4.plt
plot/l/intensity 1 1 ns=64 h=.2      # plot far-field intensity
c
c  now with power in lens array elements
c
zreff = 0
array/set 1 128
units/s 0 .075              # set units to .075
clear 1 0                   # set entire field to unity
n_times = 4
y_dec = -5.0                # initialize y-decenter
werr = 0.5                  # f=5000 cm
macro/run outer/n_times
title pupil phase of lens array
plot/watch ex64_5.plt
plot/l/phase 1 1 ns=64 h=.1          # plot pupil intensity
lens 1 100
dist 100 1
title far-field of lens array
plot/watch ex64_6.plt
plot/l/intensity 1 1 ns=64 h=.2      # plot far-field intensity
end

```


Ex65: Incoherent imaging using OTF

This example shows illustrates incoherent imaging. GLAD performs coherent propagation. We can form the coherent impulse response, take the absolute value squared, and convolve that with the irradiance distribution which is to be image.

In this example, the incoherent image of a cosine wave pattern is to be formed by a 1:1 relay lens. The working f-number is $f/5$. We proceed by forming the optical transfer function (OTF) from the ideal image of an $f/5$ beam. We use the `convolve/beam` command to calculate the convolution of the cosine bar pattern with the impulse response. The cosine wave pattern is selected to have a period equal to the diffraction-limited period and twice the diffraction-limited period.

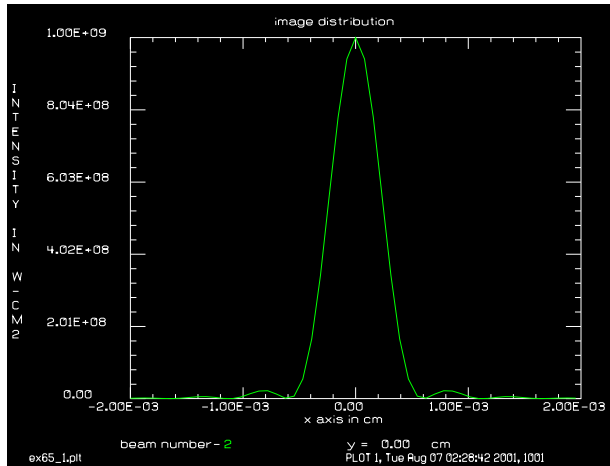


Fig. 65.1. Diffraction-limited image of $f/5$ beam.

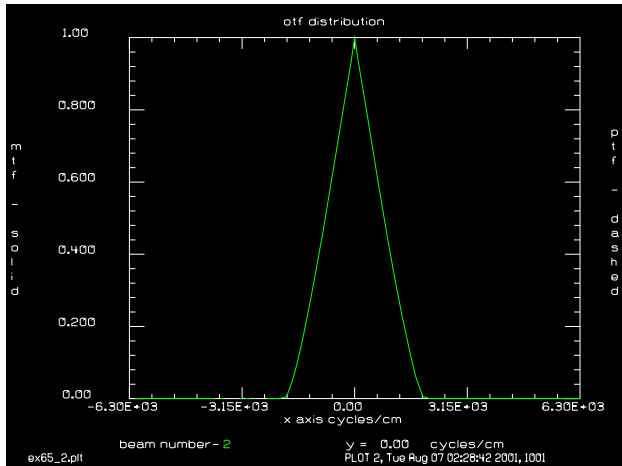


Fig. 65.2. OTF of $f/5$ image.

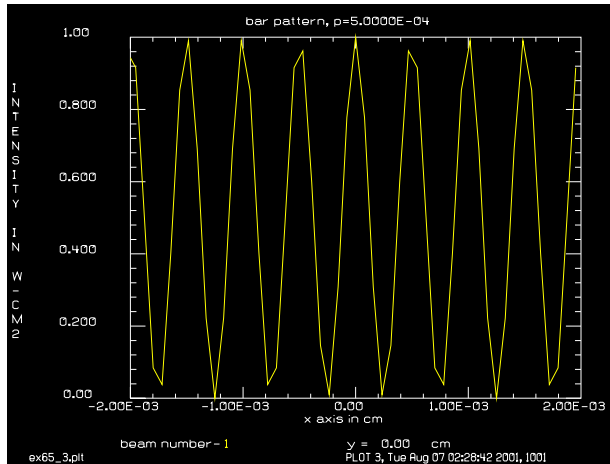


Fig. 65.3. Cosine bar pattern object at the diffraction-limited period.

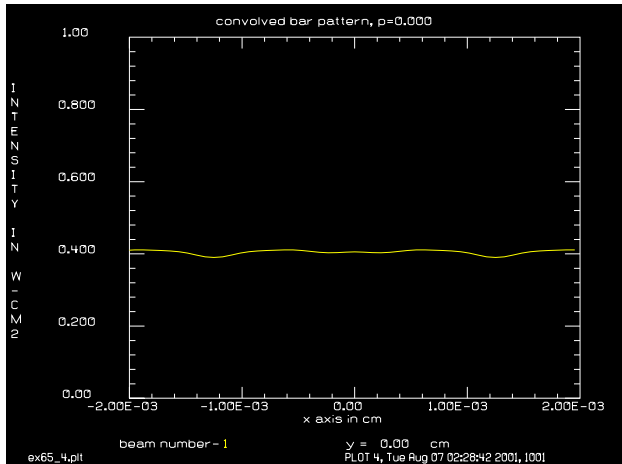


Fig. 65.4. Image of cosine bar pattern at the diffraction-limited period shows no modulation.

Input: `ex65.inp`

c## ex65

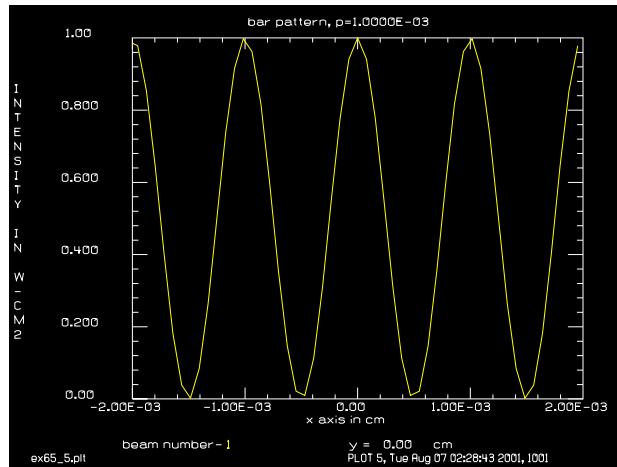


Fig. 65.5. Cosine bar pattern object at twice diffraction-limited period.

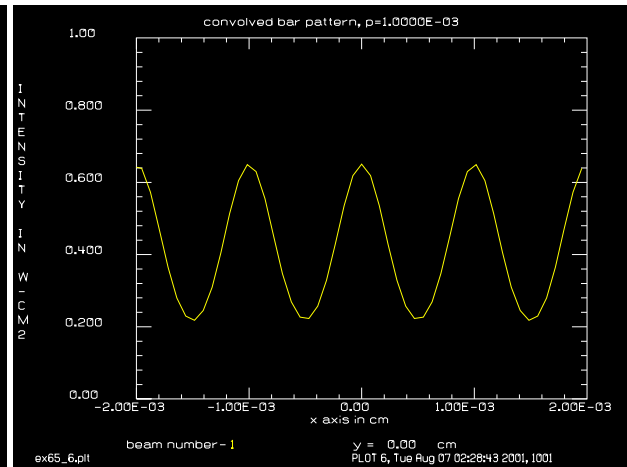


Fig. 65.6. Image of cosine bar pattern of twice diffraction-limited period.

```

c
c Example 65: Incoherent imaging
c
c This example illustrates incoherent imaging.
c A bar pattern is imaged by a lens of finite numerical aperture.
c
c Assume a 1:1 relay lens and a bar pattern of half the
c cutoff frequency as determined by the numerical aperture
c
c The lens has focal length of 50 cm and an aperture of
c 10 cm radius. At 1:1 the cutoff frequency will be
c
c  $f_{max} = 20/(\lambda \cdot 100) = 2000 \text{ cycles/cm}$ 
c
c twice the minimum grating period is  $p = .001$ 
c only half the system is modeled so the lens focal length
c is set to  $f = 100$  rather than  $f = 50$ .
c
echo/on
nbeam 2
array/s 0 128 # set array to 128 x 128
wavelength/set 0 1 # wavelength 1 micron
units/s 2 1 # pupil units are 1
clap/c/c 2 10 # clear aperture is 10 cm radius
lens 2 100 # lens is 100 cm length
dist 100 2 # propagate to focus
title image distribution
plot/watch ex65_1.plt
plot/x/i 2 left=-.002 right=.002 fmin=0.
variab/set Units 2 units # store actual units
otf/image 2 # find OTF of image
title otf distribution
plot/watch ex65_2.plt
plot/x/r 2 fmin=0. fmax=1.
zreff/se 2 0

```

Jump to: [Commands](#), [Theory](#)

```

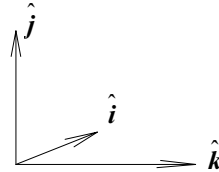
clear 1 1                                # initialize beam 1
units/s 1 Units                          # set up units for convolution
units/s 2 Units
period = .0005                           # choose grating period
grat/c 1 pe=period                        # cosine grating at maximum frequency
title bar pattern, p=@period
plot/watch ex65_3.plt
plot/x/i 1 left=-.002 right=.002 fmin=0. fmax=1.
convol/beam/nofft 1 2                    # convolution without FFTing beam 2
title convolved bar pattern, p=%r1
plot/watch ex65_4.plt
plot/x/i 1 left=-.002 right=.002 fmin=0 fmax=1.
clear 1 1                                # reinitialize beam 1
period = .001
grat/c 1 pe=period                        # grating at half limiting frequency
title bar pattern, p=@period
plot/watch ex65_5.plt
plot/x/i 1 left=-.002 right=.002 fmin=0. fmax=1.
convol/beam/nofft 1 2                    # convolution without FFTing beam 2
title convolved bar pattern, p=@period
plot/watch ex65_6.plt
plot/x/i 1 left=-.002 right=.002 fmin=0 fmax=1.

```


Ex66: Roof mirrors and corner cubes

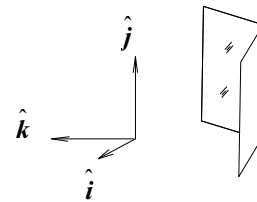
Roof mirrors are an important class of optical elements. Roof mirrors and cube corner reflectors send the light nominally back on itself. One may define roof mirrors with the roof oriented in the north-south, east-west, northeast-southwest, northwest-southeast directions. The beam may be described by a set of \hat{i} -, \hat{j} -, and \hat{k} -vectors with \hat{k} being in the direction of propagation and \hat{i} and \hat{j} being the transverse coordinates. The transformations due to mirrors may be described by a set of 3×3 matrix operations. The upper lefthand cofactor describes the rotation properties of the matrix. The polarization properties are affected by these operations.

$$\text{free space} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



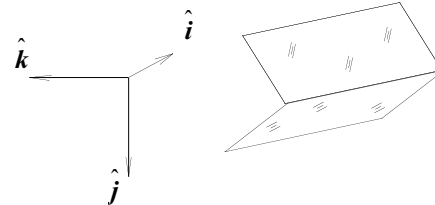
$$90^\circ \text{ y-roof} \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

insensitive to y-rotation, 2θ for x- or z-rotations, parity 1



$$90^\circ \text{ x-roof} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

insensitive to x-rotation, 2θ for y- or z-rotations, parity 1

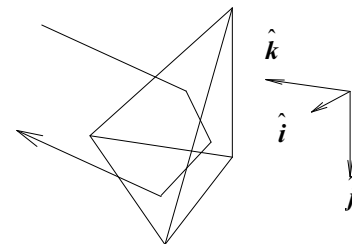


$$\text{NESW roof} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}, \text{NWSE roof} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

(not shown)

$$\text{after reflection from a cube corner} \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

insensitive to all rotations, retroreflects, parity 1



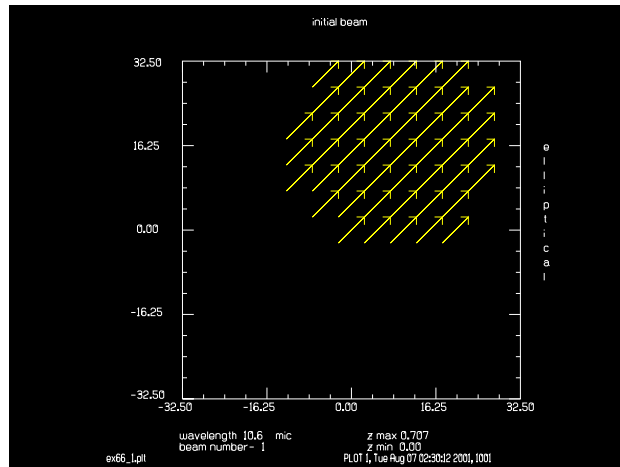


Fig. 66.1. Initial distribution with linear polarization rotated at 45° .

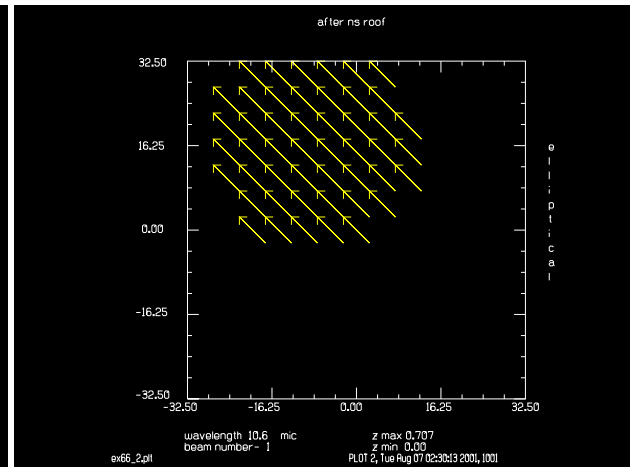


Fig. 66.2. Pupil after reflection from a NS roof. Both aperture and polarization are affected.

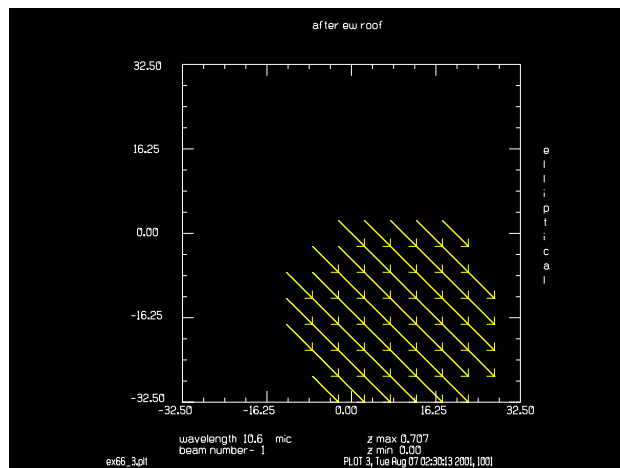


Fig. 66.3. After reflection from an EW roof.

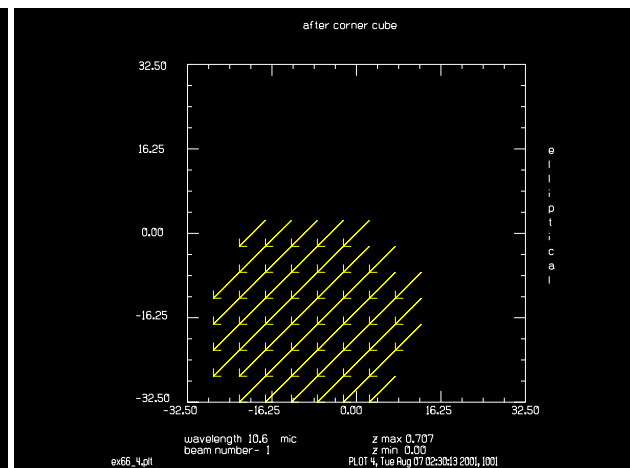


Fig. 66.4. After reflection from a corner cube reflector.

Input: ex66.inp

```

c## ex66
c
c Example 66: Roof mirrors and corner cube
c
c This example illustrates roof mirrors and a corner cube. All of these
c elements send the beam nominally back on itself but the beam
c and the polarization state is rotated differently for the different
c mirror systems.
c
c roof/ns      roof axis is north-south
c roof/ew      roof axis is east-west
c roof/nsw     roof axis is northeast-southwest
c roof/nwse    roof axis is northwest-southeast
c corner       corner cube reflector

```

Jump to: [Commands](#), [Theory](#)

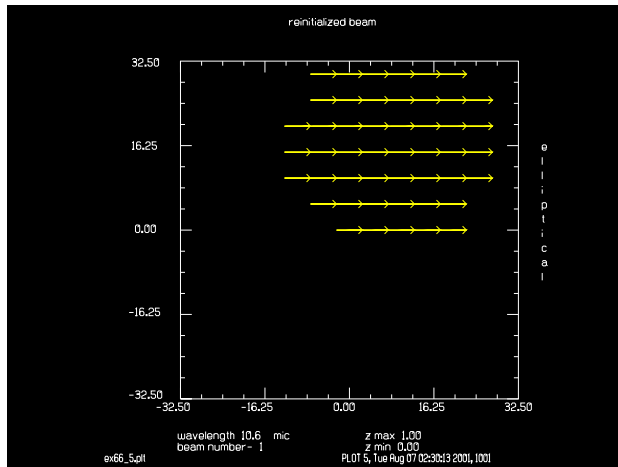


Fig. 66.5. Reinitialized beam to better display roofs at angles 45° .

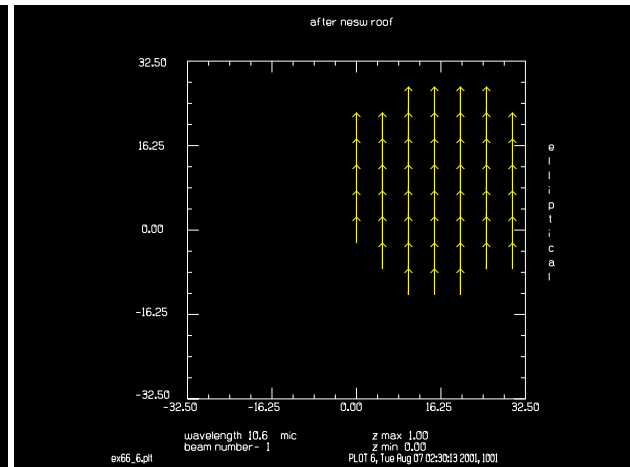


Fig. 66.6. After reflection from NESW roof, starting with Fig. 66.5.

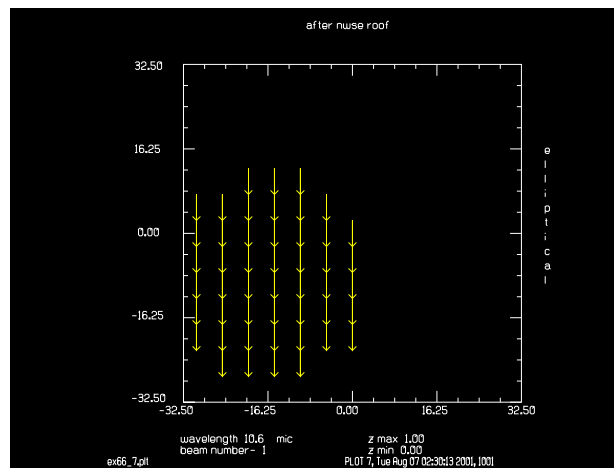


Fig. 66.7. After reflection from a NWSE roof.

```

c
echo/on
nbeam 2
array/s 0 64 64 1
global/def
set/density 12 12
vertex/loc/abs
clap/c/c 1 20. 8. 16.
jones/set ar=.707 br=-.707 cr=.707 dr=.707 # rotate linear polarization
jones/multiply 1 # 45 degrees
copy/con 1 2 # store beam for later use
title initial beam
plot/watch ex66_1.plt
plot/ell 1 # plot elliptical polarization
roof/ns 1 # NS roof
global # list beam coordinates
title after ns roof
plot/watch ex66_2.plt

```

Jump to: [Commands](#), [Theory](#)

```
plot/ell 1          # plot elliptical polarization
copy 2 1
roof/ew 1           # EW roof
global              # list beam coordinates
title after ew roof
plot/watch ex66_3.plt
plot/ell 1          # plot elliptical polarization
copy 2 1
corner 1            # corner cube
title after corner cube
plot/watch ex66_4.plt
plot/ell 1          # plot elliptical polarization
clear 1 1           # reinitialize for horizontal
clap/c/c 1 20. 8. 16. # linear polarization
copy 1 2            # save beam again
title reinitialized beam
plot/watch ex66_5.plt
plot/ell 1          # plot elliptical polarization
roof/nesw 1         # NESW roof
global              # list beam coordinates
title after nesw roof
plot/watch ex66_6.plt
plot/ell 1          # plot elliptical polarization
copy 2 1
roof/nwse 1         # NWSE roof
global
title after nwse roof
plot/watch ex66_7.plt
plot/ell 1          # plot elliptical polarization
clear 1 1           # reinitialize for horizontal
clap/c/c 1 20. 8. 16. # linear polarization
jones/set ar=.707 br=-.707 cr=.707 dr=.707 # rotate linear polarization
jones/multiply 1     # 45 degrees
copy/con 1 2         # store beam for later use
title initial beam, again
plot/watch ex66_8.plt
plot/ell 1          # plot elliptical polarization
roof/nesw 1         # NESW roof
global              # list beam coordinates
title after nesw roof, unchanged
plot/watch ex66_9.plt
plot/ell 1          # plot elliptical polarization
copy 2 1
roof/nwse 1         # NWSE roof
global
title after nwse roof, flipped
plot/watch ex66_10.plt
plot/ell 1          # plot elliptical polarization
```

Ex67: Lens and laser diode arrays

Table. 67.1. Table of Ex67 examples

Ex67a: Hexagonal lens array	1
Ex67b: Rectangular lens array	6
Ex67c: Lensarray used as optical integrator	7
Ex67d: Four cylindrical lenses	9
Ex67e: Array of lenslets with 25 cm focal length	11
Ex67f: Two lensarrays creating afocal 1:1 imager	13
Ex67g: Array of optical fibers collimated by lensgroup	15
Ex67h: Array of N x N laser diodes, gaussian overall envelope	18

GLAD can create hexagonal or rectangular arrays of lenses. The width of each element, the focal length, and random piston, tilt, and defocus errors may be included for the whole array. Lens elements may also be set individually. The lensarray command may be used to define a lens array or a multilevel array with random piston, tilt, or defocus assigned in a specified number of levels.

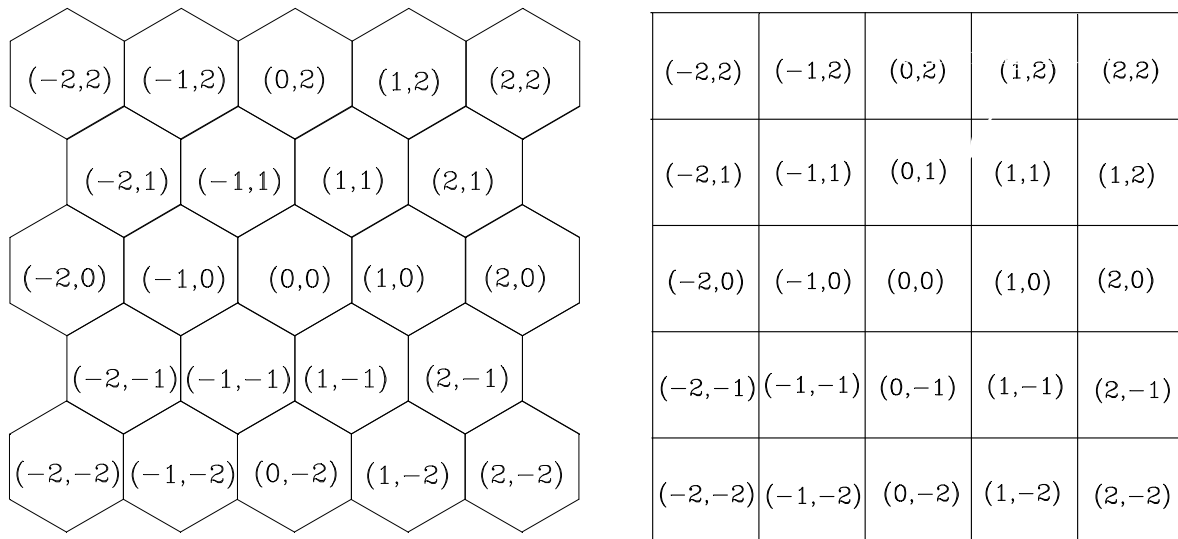


Fig. 67.1a. Hexagonal array showing numbering system. Fig. 67.1b. Rectangular array showing numbering system.

Ex67a: Hexagonal lens array

Input: ex67a.inp

```

c## ex67a
c
c Example 67A: hexagonal lens array
c
c This is a test of hexagonal lens arrays
c
alias Name ex67a

```

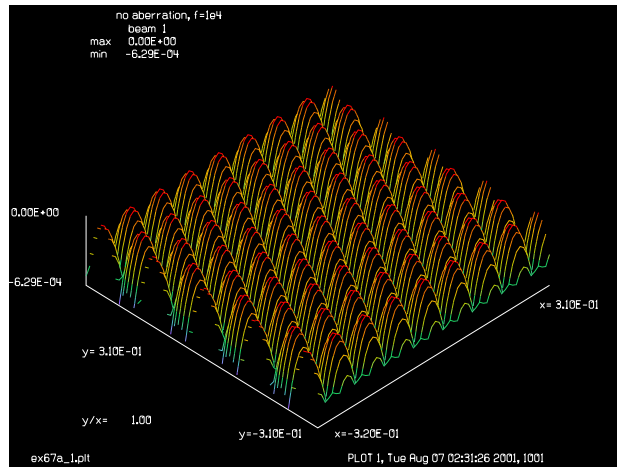


Fig. 67.2a. Phase of hexagonal lens array.

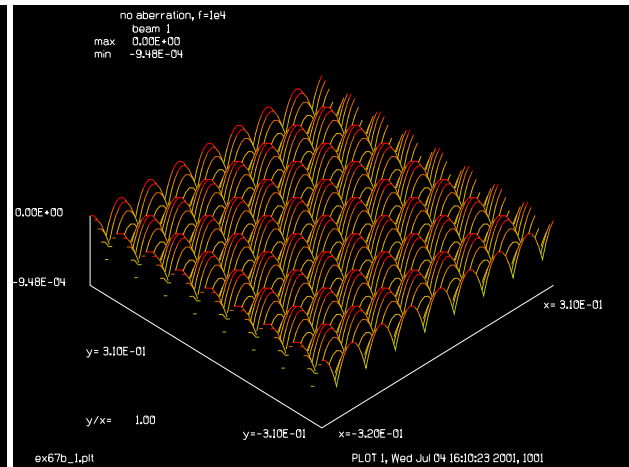


Fig. 67.2b. Phase of rectangular lens array.

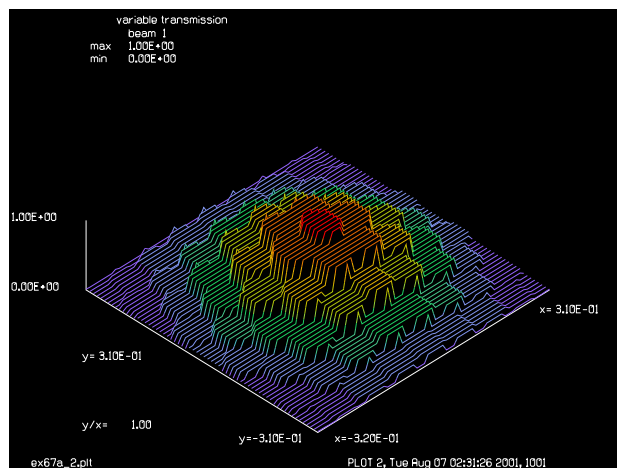


Fig. 67.3a. Hexagonal variable intensity, constant over each lens element.

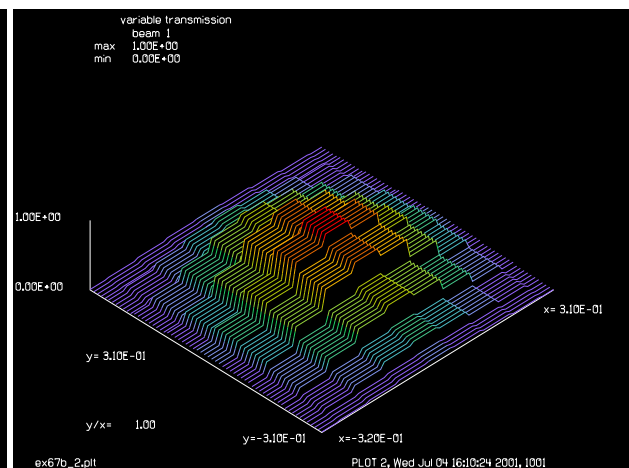


Fig. 67.3b. Rectangular variable intensity, constant over each lens element.

```

Theta = 32                                # tilt angle for plot/liso
array/s 1 64
units/s 1 .01
set/density 64 64                         # set plotting density
lensarray/hex/set omega=.2                # set gaussian transmission function
lensarray/hex/all 1 .08 1e4               # hexagonal lenses of .08 width
title no aberration, f=1e4
plot/watch @Name_1.plt
plot/l/ph h=.3 ns=64 theta=Theta
title variable transmission
plot/watch @Name_2.plt
plot/l h=.3 ns=64 theta=Theta
lensarray/hex/clear 1 -1 -2 .08 value=1   # clear single element
title intensity after clearing (-1,-2)
plot/watch @Name_3.plt
plot/l h=.3 ns=64 theta=Theta
lensarray/hex/element 1 -1 -2 .08 2e3    # set single element (-1,-2)

```

Jump to: [Commands](#), [Theory](#)

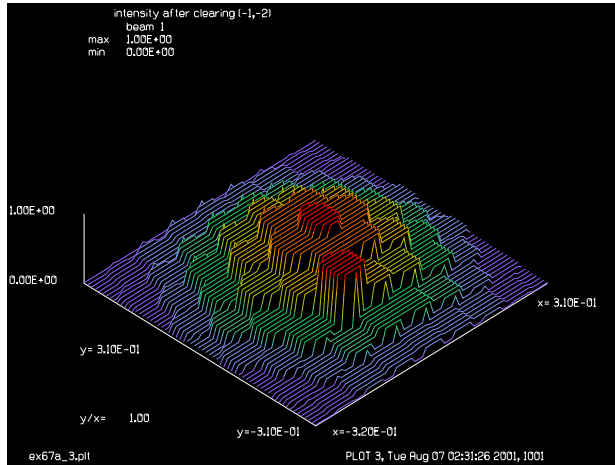


Fig. 67.4a. Hexagonal elements, reset (-1,-2).

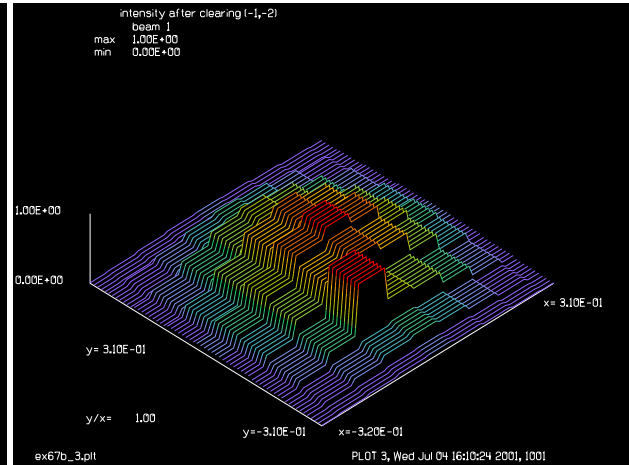


Fig. 67.4b. Rectangular variable intensity, constant over each lens element.

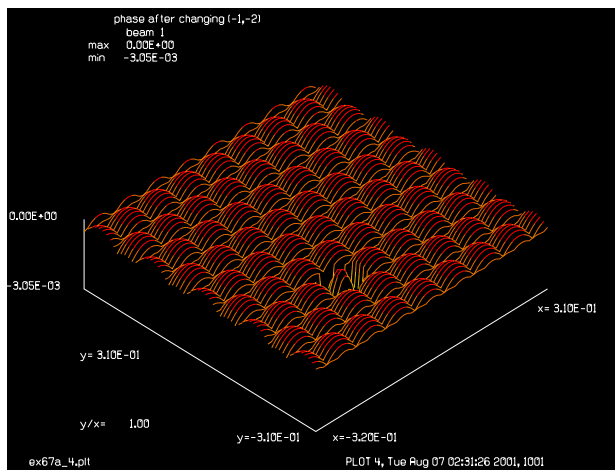


Fig. 67.5a. Phase of hexagonal lens array with (-1,-2) reset.

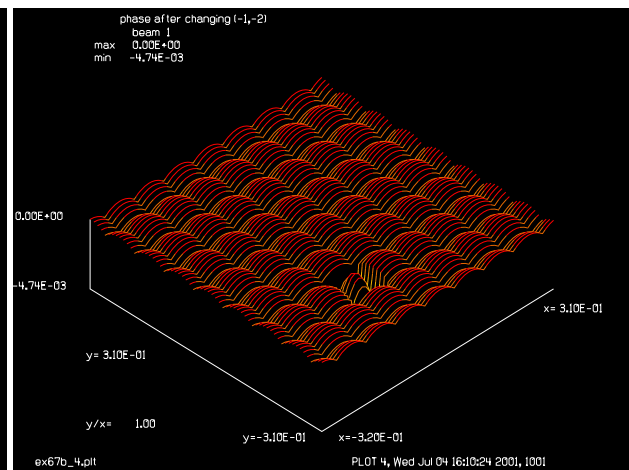


Fig. 67.5b. Phase of rectangular lens array with (-1,-2) reset.

```

title phase after changing (-1,-2)
plot/watch @Name_4.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/hex/set piston=.05      # set random piston peak-to-valley
lensarray/hex/all 1 .08 1e10      # hexagonal array with very large focal
c                                  # length to illustrate random piston
title random piston aberration
plot/watch @Name_5.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/hex/set focus=.05       # random focal length error
lensarray/hex/all 1 .08 1e10
title random focus aberration
plot/watch @Name_6.plt
plot/l/ph h=.3 ns=64 theta=Theta

```

Jump to: [Commands](#), [Theory](#)

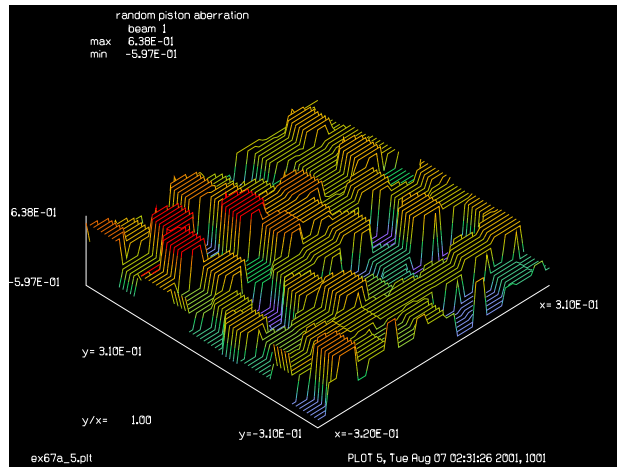


Fig. 67.6a. Phase of hexagonal lens array, long focal length, random phase error dominates.

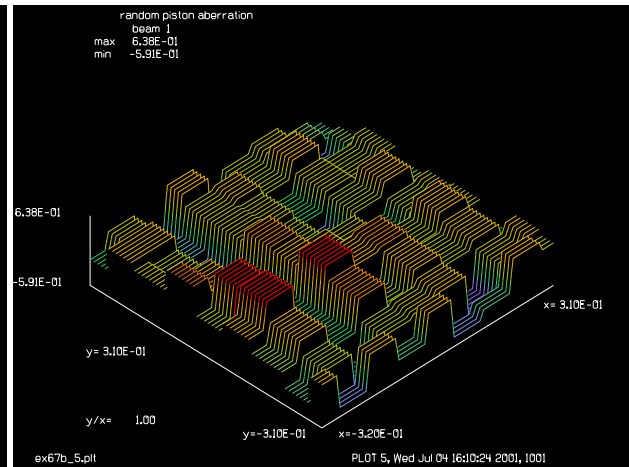


Fig. 67.6b. Phase of rectangular lens array, long focal length, random phase error dominates.

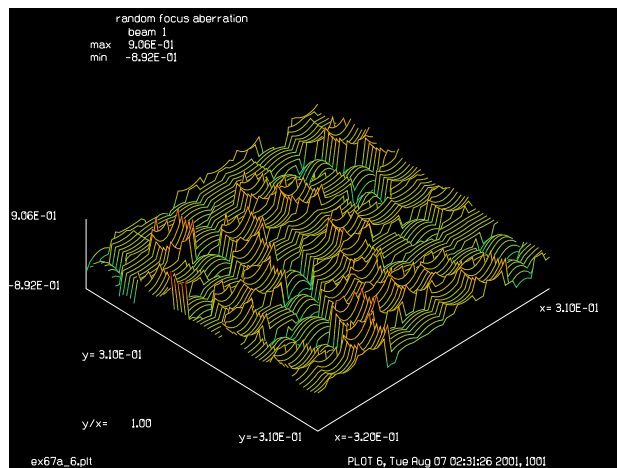


Fig. 67.7a. Phase of hexagonal lens array, long focal length, random focus error dominates.

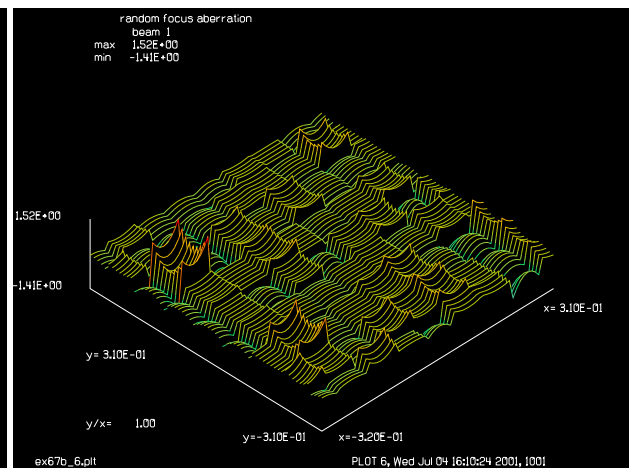


Fig. 67.7b. Phase of rectangular lens array, long focal length, random focus error dominates.

```
clear 1 1
lensarray/hex/set piston=.05
lensarray/hex/multilevel 1 .08 n1=2
title multilevel piston, n1=2
plot/watch @Name_7.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/hex/set piston=.05
lensarray/hex/multilevel 1 .08 n1=4
c
title multilevel piston, n1=4
plot/watch @Name_8.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/hex/set xtilt=.05
c
```

```
# multilevel random piston
# set for two levels
```

```
# multilevel random piston with
# four levels
```

```
# multilevel random tilt with
# random seed set, 100 levels
```

Jump to: [Commands](#), [Theory](#)

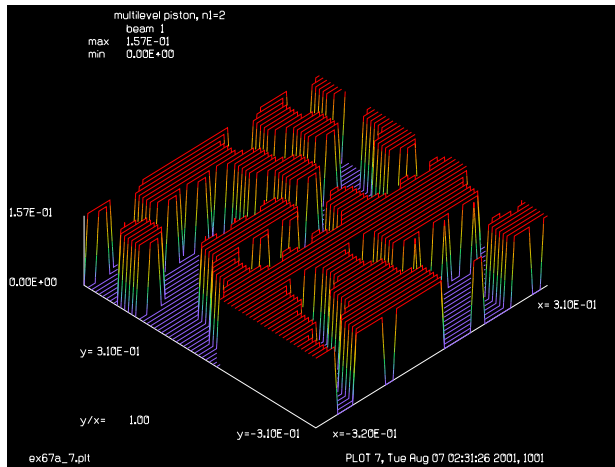


Fig. 67.8a. Multilevel mode, hexagonal 2 level random tilt.

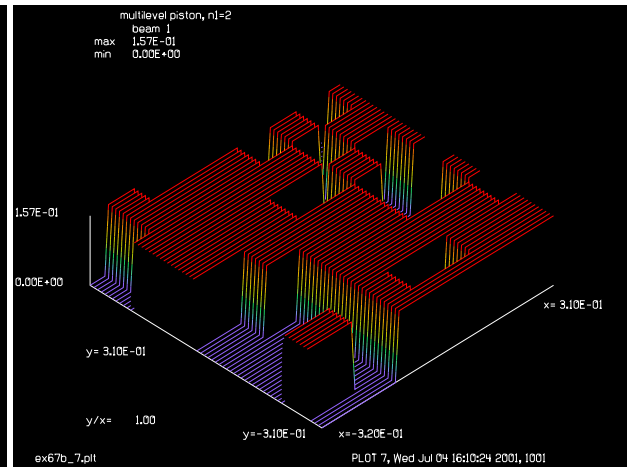


Fig. 67.8b. Multilevel mode, rectangular 2 level random tilt.

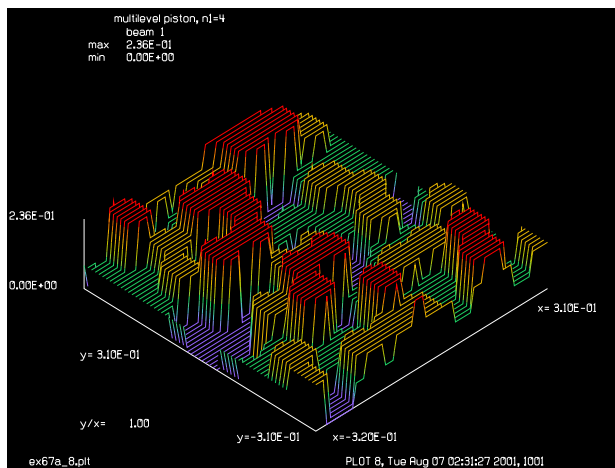


Fig. 67.9a. Multilevel mode, hexagonal 4 level random piston.

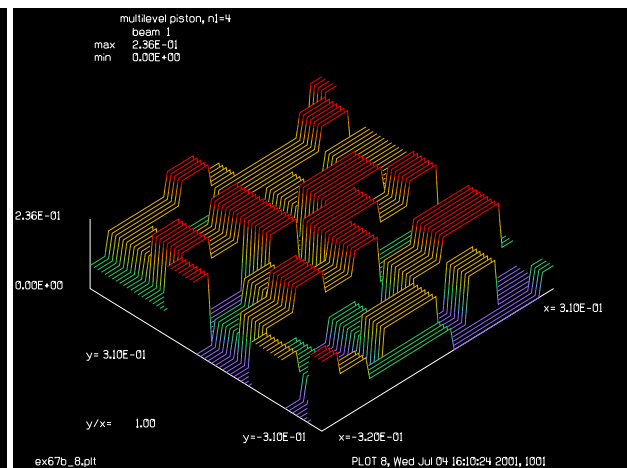


Fig. 67.9b. Multilevel mode, rectangular 4 level random piston.

```
lensarray/hex/multilevel 1 .08 n2=100 iseed=5
title multilevel xtlt, n2=100, iseed=5
plot/watch @Name_9.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/hex/set xtlt=.05                                # multilevel random tilt with same
c                                                            # seed, distribution is the same
lensarray/hex/multilevel 1 .08 n2=100 iseed=5
title multilevel xtlt, n3=100, iseed=5
plot/watch @Name_10.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/hex/set xtlt=.05                                # same seed with four levels
lensarray/hex/multilevel 1 .08 n2=4 iseed=5
title multilevel xtlt, n3=4, iseed=5
plot/watch @Name_11.plt
```

Jump to: [Commands](#), [Theory](#)

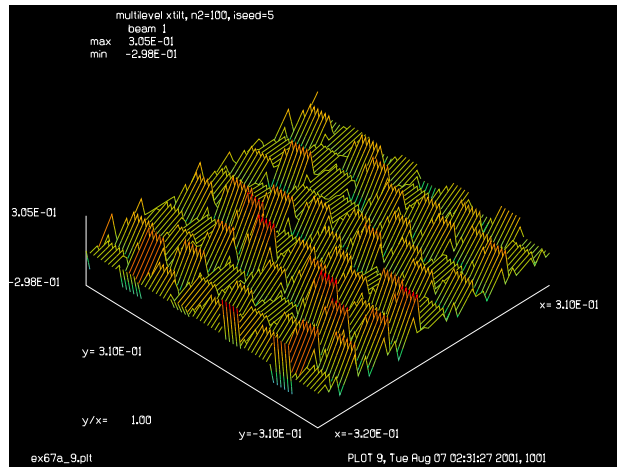


Fig. 67.10a. Multilevel mode, hexagonal 100 level random tilt.

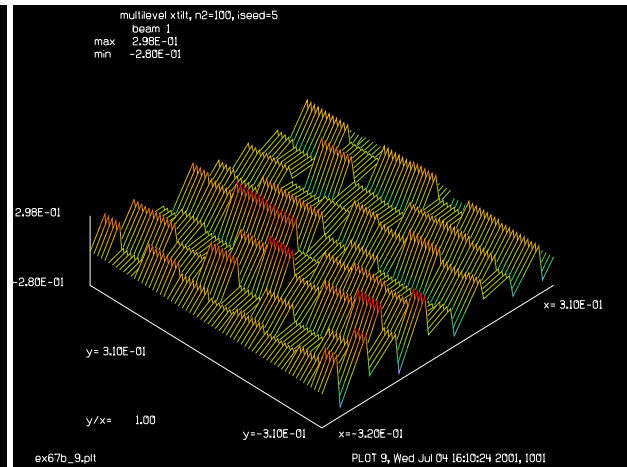


Fig. 67.10b. Multilevel mode, rectangular 100 level random tilt.

plot/l/ph h=.3 ns=64 theta=Theta

Ex67b: Rectangular lens array

Input: `ex67b.inp`

```
c## ex67b
c
c Example 67B: rectangular lens array
c
c This is a test of rectangular lens arrays
c
alias Name ex67b
Theta = 32
array/s 1 64
units/s 1 .01
set/density 64 64 # set plotting density
lensarray/rec/set omega=.2 # set gaussian transmission function
lensarray/rec/all 1 .08 f=1e4 # rectangular lenses of .08 width
title no aberration, f=1e4
plot/watch @Name_1.plt
plot/l/ph h=.3 ns=64 theta=Theta
title variable transmission
plot/watch @Name_2.plt
plot/l h=.3 ns=64 theta=Theta
lensarray/rec/clear 1 -1 -2 .08 value=1 # clear single element
title intensity after clearing (-1,-2)
plot/watch @Name_3.plt
plot/l h=.3 ns=64 theta=Theta
lensarray/rec/element 1 -1 -2 .08 f=2e3 # set single element (-1,-2)
title phase after changing (-1,-2)
plot/watch @Name_4.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
```

Jump to: [Commands](#), [Theory](#)

```

lensarray/rec/set piston=.05      # set random piston peak-to-valley
lensarray/rec/all 1 .08 f=1e10    # rectangular array with very large focal
c                                # length to illustrate random piston
title random piston aberration
plot/watch @Name_5.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/rec/set focus=.05      # random focal length error
lensarray/rec/all 1 .08 f=1e10
title random focus aberration
plot/watch @Name_6.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/rec/set piston=.05      # multilevel random piston
lensarray/rec/multilevel 1 .08 n1=2 # set for two levels
title multilevel piston, n1=2
plot/watch @Name_7.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/rec/set piston=.05
lensarray/rec/multilevel 1 .08 n1=4 # multilevel random piston with
c                                # four levels
title multilevel piston, n1=4
plot/watch @Name_8.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/rec/set xtilt=.05      # multilevel random tilt with
c                                # random seed set, 100 levels
lensarray/rec/multilevel 1 .08 n2=100 iseed=5
title multilevel xtilt, n2=100, iseed=5
plot/watch @Name_9.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/rec/set xtilt=.05      # multilevel random tilt with same
c                                # seed, distribution is the same
lensarray/rec/multilevel 1 .08 n2=100 iseed=5
title multilevel xtilt, n3=100, iseed=5
plot/watch @Name_10.plt
plot/l/ph h=.3 ns=64 theta=Theta
clear 1 1
lensarray/rec/set xtilt=.05      # same seed with four levels
lensarray/rec/multilevel 1 .08 n2=4 iseed=5
title multilevel xtilt, n3=4, iseed=5
plot/watch @Name_11.plt
plot/l/ph h=.3 ns=64 theta=Theta

```

Ex67c: Lensarray used as optical integrator

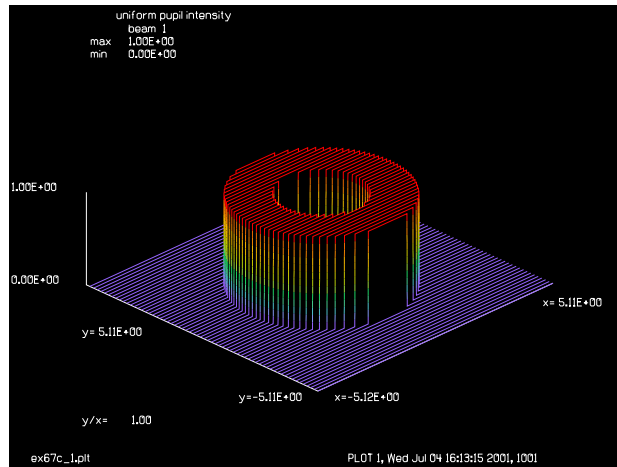


Fig. 67.11. Far-field uniform intensity annular pupil for ex67c.

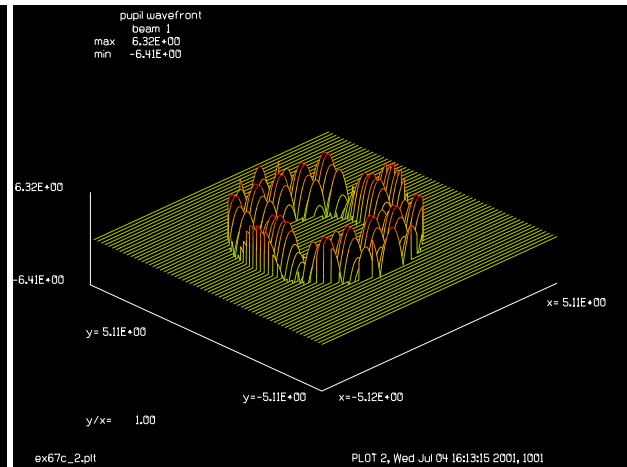


Fig. 67.12. Phase in the pupil showing effect of lenslet array.

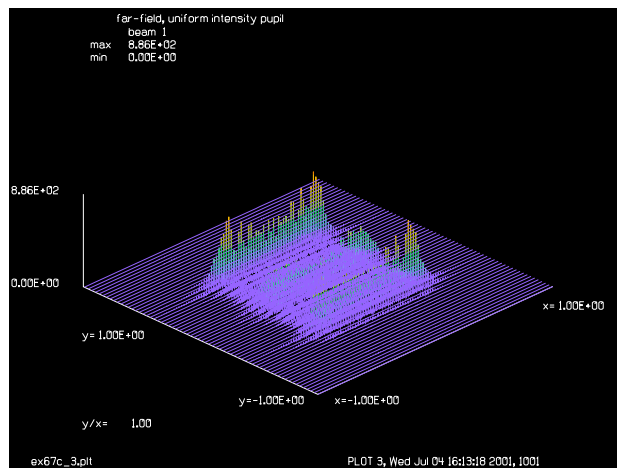


Fig. 67.13. Far-field showing fine speckle.

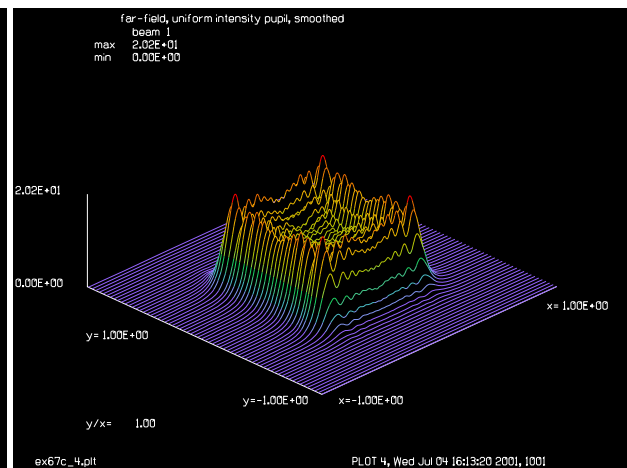


Fig. 67.14. Far-field after smoothing showing a high degree of integration.

Input: ex67c.inp

```
c## ex67c!065720358826569
c
c Example 67c: Lensarray used as optical integrator
c
c A combination of a lens array with negative elements and an ordinary
c positive lens may be used to simulate an optical integrator.
c
c An annular aperture is incident on the integrator. The output is a
c nearly square distribution with extremely fine speckle structure.
c To better illustrate the envelope of the irradiance, the speckle
c structure is smoothed by convolution with a gaussian of 1/e**2 radius
c of .02.
c
c Variables
```

Jump to: [Commands](#), [Theory](#)

```

c -----
c  dist          focal lengths and propagation distance
c
alias Name ex67c
dist = 20
mem/set/b 16
array/s 1 1024
wavelength/set 1 10.6
units/s 1 .01
lensarray/rec/all 1 1 1 -dist          # lens array of negative elements
c
c  Make annular aperture to be integrated
c
clap/c/c 1 3
obs/c/c 1 1.5
title uniform pupil intensity
plot/watch @Name_1.plt
plot/l/i 1 ns=64
title pupil wavefront
plot/watch @Name_2.plt
plot/l/w 1 ns=64
lens 1 dist                          # full aperture lens
dist dist                            # propagate to far-field
title far-field, uniform intensity pupil
plot/watch @Name_3.plt
plot/l 1 xrad=1 ns=64
amplitude/abs 1                      # remove phase for better convolution
convol/gaus 1 .02                    # convolution to smooth speckles
title far-field, uniform intensity pupil, smoothed
plot/watch @Name_4.plt
plot/l 1 xrad=1 ns=64
end

```

Ex67d: Four cylindrical lenses

Input: ex67d.inp

```

c## ex67d
c
c  Example 67d: four cylindrical lenses
c
alias Name ex67d
FocalLength = 200          # focal length of 200 cm
mem/set/b 8               # set memory for 8 megabytes
array/s 1 1024
wavelength/set 1 1.0      # set wavelength to 1 micron
Field = 4                 # variable for field half-width
#
# make an array of lenses, each lens 1 cm wide of
# focal length = FocalLength

```

Jump to: [Commands](#), [Theory](#)

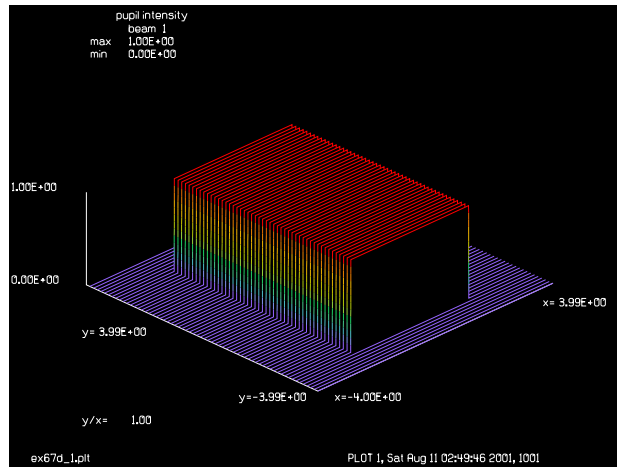


Fig. 67.15. Rectangular uniformly filled aperture for Ex67d.

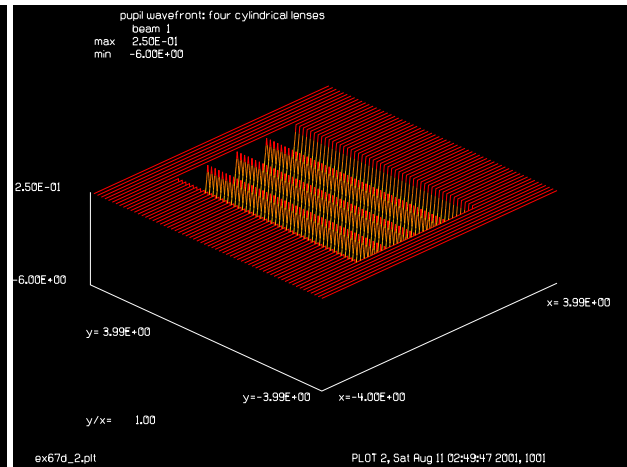


Fig. 67.16. For cylindrical lenses created by lensarray.

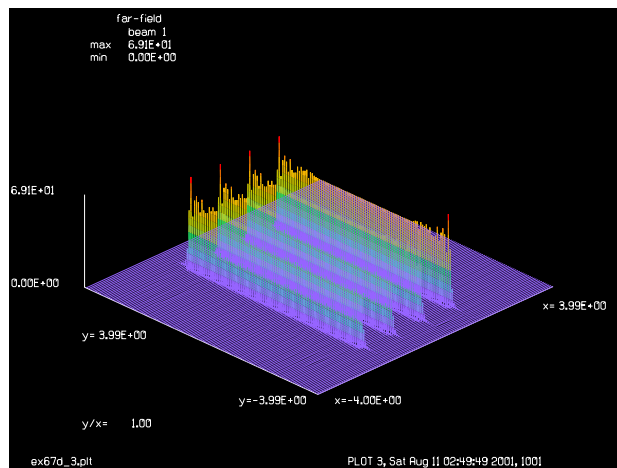


Fig. 67.17. Focus plane of cylindrical lenses.

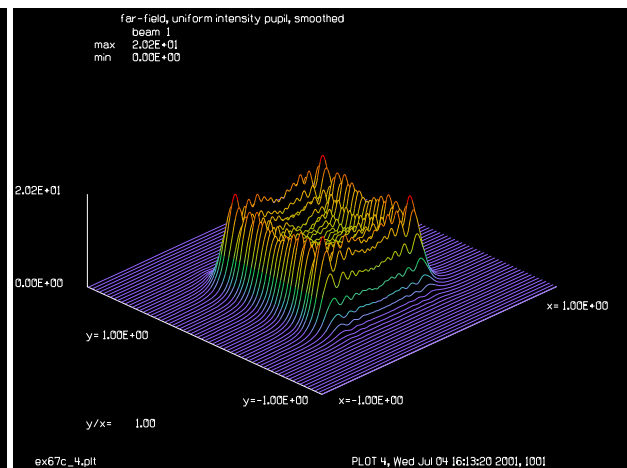


Fig. 67.18. Profile across focus plane.

```
# to make a cylindrical lens, just make the spacing per pixel
# very small in the y-direction and restore after the lensarray
#
units/field 1 Field 1e-6
lensarray/rec/all 1 1 1 FocalLength          # lens array
# restore to proper units
units/field 1 Field Field
#
# shift by 1/2 lens width so the four cylindrical lenses
# are symmetrical about x=0
#
shift/cart 1 x=-.5                          # ship to make four lenses in the cent
#
# make an aperture 2 x 6 cm
#
clap/r/c 1 2 3                               # make an aperture of 2 x 6 cm
# normalize the beam to remove any small artifacts from
```

Jump to: [Commands](#), [Theory](#)

```

# the shift command
norm 1 .01
title pupil intensity
plot/watch @Name_1.plt
plot/l/i 1 ns=64
title pupil wavefront: four cylindrical lenses
plot/watch @Name_2.plt
plot/l/w 1 ns=64
prop FocalLength
title far-field
plot/watch @Name_3.plt
plot/l 1 ns=128
plot/watch @Name_4.plt
title scan in x-direction
plot/x/i 1
plot/watch @Name_5.plt
title scan in y-direction
plot/y/i 1

```

Ex67e: Array of lenslets with 25 cm focal length

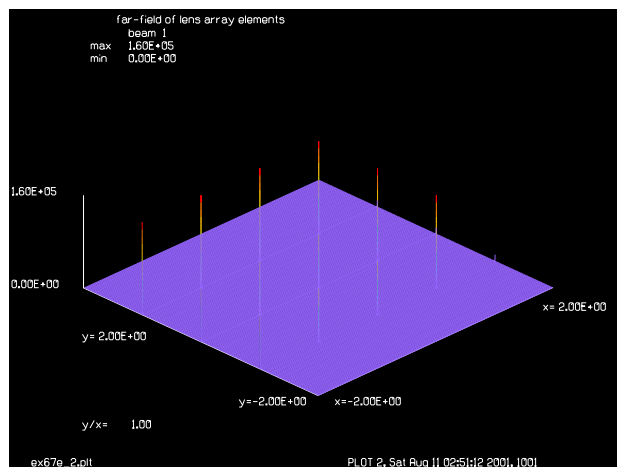


Fig. 67.19. Focused spots after lenslet array.

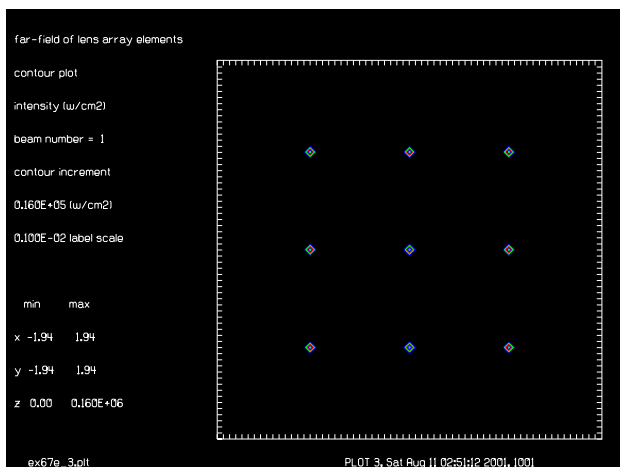


Fig. 67.20. Contour plots of focus plane.

Input: `ex67e.inp`

```

c## ex67e
c
c Example 67e: images of a single lensarray
c
alias Name ex67e
variab/dec/int Nline
Nline = 2048
FocalLength = 25          # focal length
mem/set/b 32             # set memory
array/s 1 Nline
Lambda = 1e-4

```

Jump to: [Commands](#), [Theory](#)

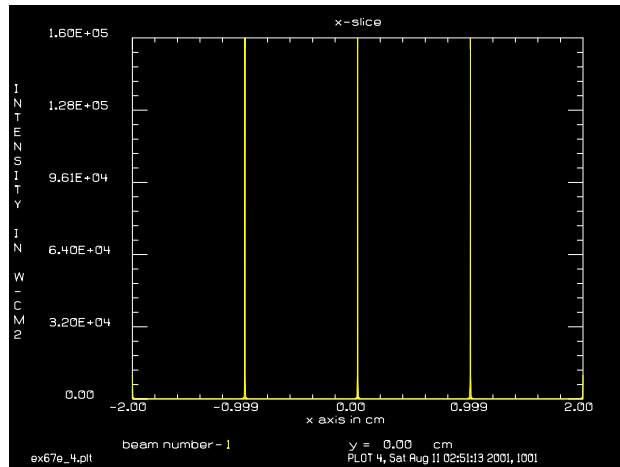


Fig. 67.21. Profile across x-direction.

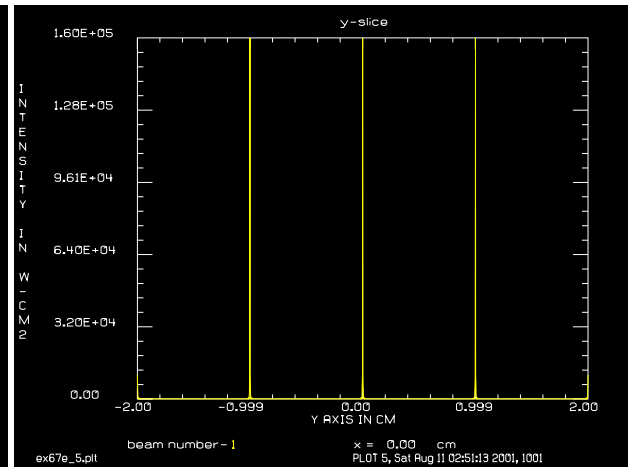


Fig. 67.22. Profile across y-direction.

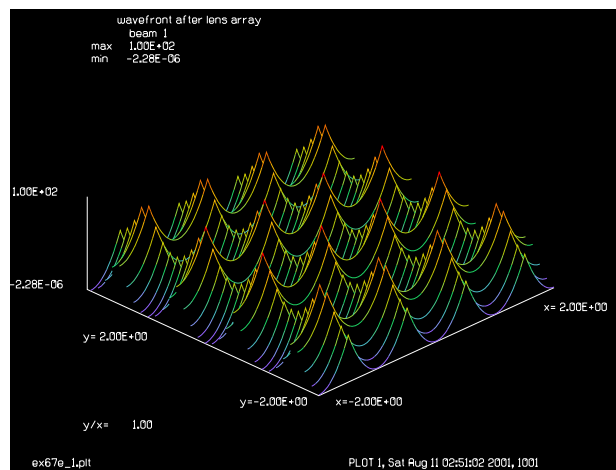


Fig. 67.23. Phasor from lenslet array.

```

wavelength/set 1 Lambda*1e4 # set wavelength to 1 micron
Field = 2                    # variable for field half-width
#
# make an array of lenses, each lens 1 cm wide of
# focal length = FocalLength
#
units/field 1 Field Field
Units = 2*Field/Nline
units/s 1 Units
geodata/set 1 waistx=1 waisty=1 # set surrogate radius large so
                                # we will stay in the near-field
                                # for propagation

clear 1 1
lensarray/rec/all 1 1 1 FocalLength # lens array of negative elements
plot/watch @Name_1.plt
title wavefront after lens array
plot/l/w 1
prop FocalLength
plot/watch @Name_2.plt

```

Jump to: [Commands](#), [Theory](#)


```

title far-field of lens array elements
plot/1 1 ns=256
set/den 256
plot/watch @Name_3.plt
plot/c
plot/watch @Name_4.plt
title x-slice
plot/x/i 1
plot/watch @Name_5.plt
title y-slice
plot/y/i 1

```

Ex67f: Two lensarrays creating afocal 1:1 imager

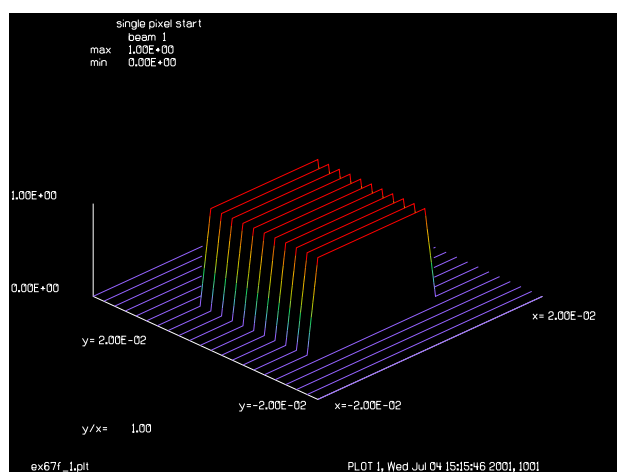


Fig. 67.24. Starting distribution to be imaged by lens array.

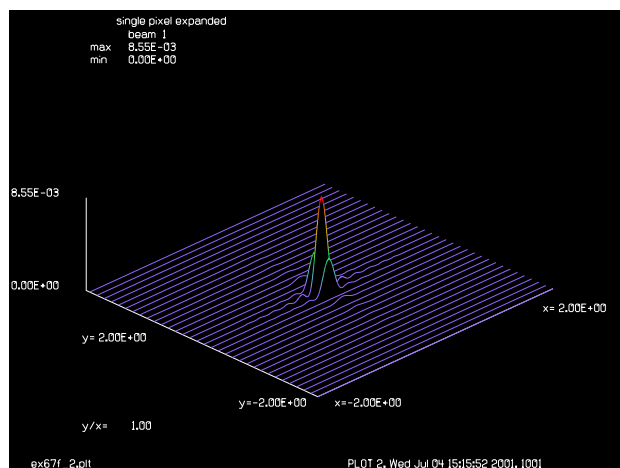


Fig. 67.25. Far-field of square aperture at plane of first lens array.

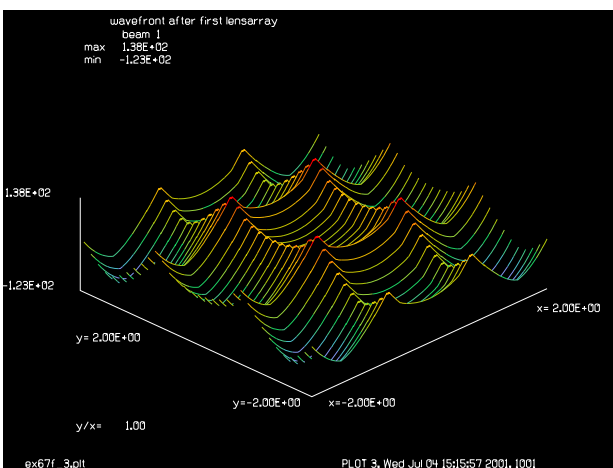


Fig. 67.26. Phase after lenarray.

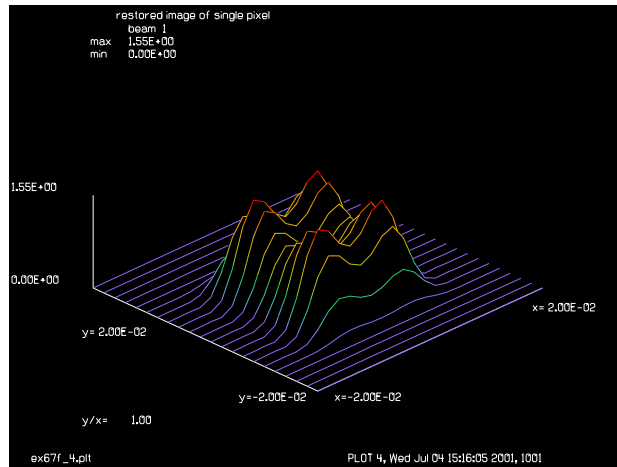


Fig. 67.27. Internal image of square aperture. Resolution is limited by low NA.

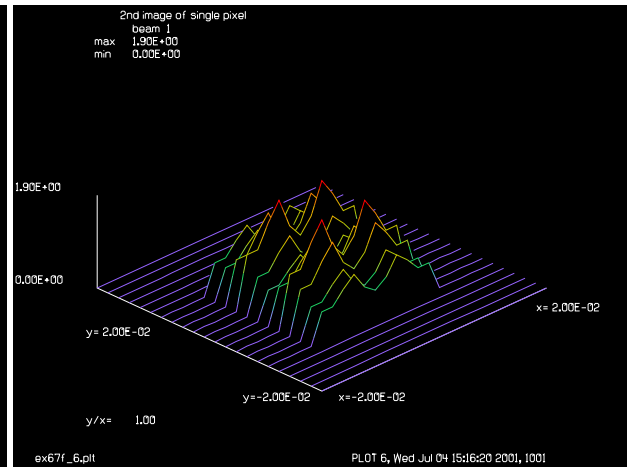


Fig. 67.28. Second afocal image.

Input: ex67f.inp

```

c## ex67f
c
c Example 67f: two lensarrays creating afocal 1:1 imager
c
alias Name ex67f
variab/dec/int Nline PltCnt
Nline = 2048
FocalLength = 25          # focal length
mem/set/b 32             # set memory
array/s 1 Nline
Lambda = 1e-4
wavelength/set 1 Lambda*1e4 # set wavelength to 1 micron
Field = 2                # variable for field half-width
#
# make an array of lenses, each lens 1 cm wide of
# focal length = FocalLength
#
units/field 1 Field Field
Units = 2*Field/Nline
units/s 1 Units
clap/s/c 1 .01           # make pixels of .02 x .02 cm
geodata/set 1 waistx=1 waisty=1
PltCnt = PltCnt + 1; plot/watch @Name_@PltCnt.plt
title single pixel start
plot/1 1 xrad=.02
#
# propagate to lensarray
#
prop 2*FocalLength
PltCnt = PltCnt + 1; plot/watch @Name_@PltCnt.plt
title single pixel expanded
plot/1 1

```

Jump to: [Commands](#), [Theory](#)

```

lensarray/rec/all 1 1 1 FocalLength          # lens array of negative elements
PltCnt = PltCnt + 1; plot/watch @Name_@PltCnt.plt
title wavefront after first lensarray
plot/1/w 1
#
# propagate to image plane
#
prop 2*FocalLength
PltCnt = PltCnt + 1; plot/watch @Name_@PltCnt.plt
title restored image of single pixel
plot/1/i 1 xrad=.02
#
# propagate to 2nd lens array
#
prop 2*FocalLength
PltCnt = PltCnt + 1; plot/watch @Name_@PltCnt.plt
title single pixel expanded, 2
plot/1 1
lensarray/rec/all 1 1 1 FocalLength          # lens array of negative elements
#
# propagate to 2nd image plane
#
prop 2*FocalLength
PltCnt = PltCnt + 1; plot/watch @Name_@PltCnt.plt
title 2nd image of single pixel
plot/1/i 1 xrad=.02

```

Ex67g: Array of optical fibers collimated by lensgroup

This example simulates an array of real lenses with aberrations calculated by lensgroup. The lensgroup is decentered by 1 micron.

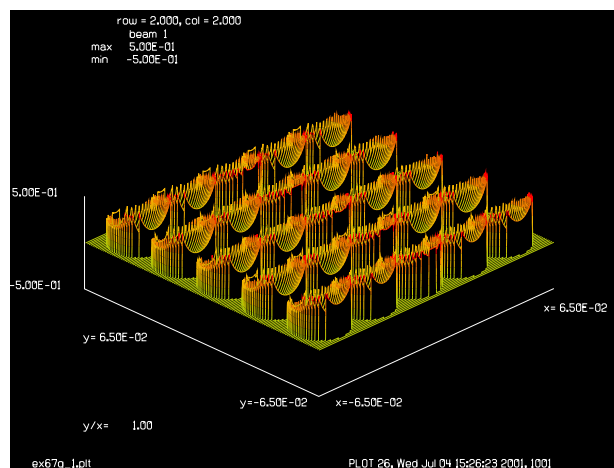


Fig. 67.29. Phase after lenarray.

Input: ex67g.inp

```

c## ex67g
#

```

Jump to: [Commands](#), [Theory](#)

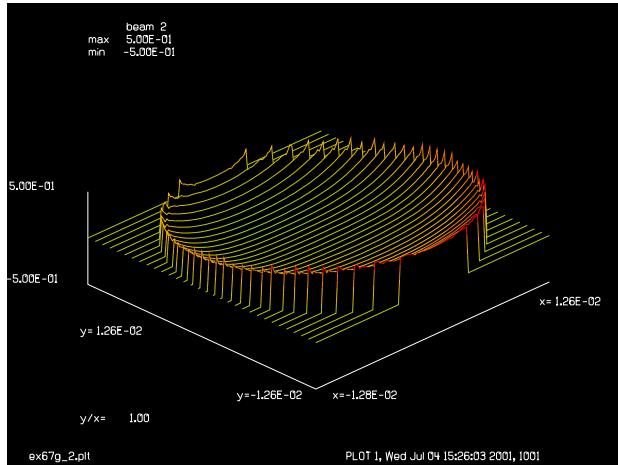


Fig. 67.30. Internal image of square aperture. Resolution is limited by low NA.

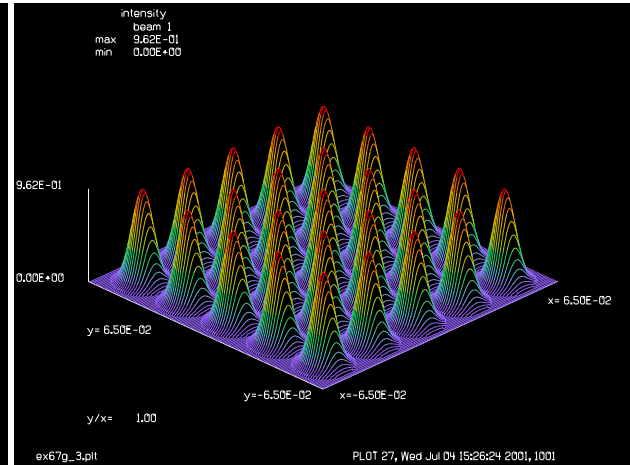


Fig. 67.31. Second afocal image.

```
# Example: Array of optical fibers collimated by lensgroup
#
# Example to calculate an array of optical fibers followed by an array
# of collimating lens elements using lensgroup and including aberrations
# due to decenter.
#
variable/dec/int LENSGROUP LOCAL
# LENSGROUP: 1 for lensgroup, 0 for ideal lens
# LOCAL: 1 for beam 2 initialization in macro "inner", 0 for
#         external initialization

alias Name ex67g
array/s 1 1024
nbeam 2
num_pix_per_beam = 128          # no. of pixels in each beamlet dimension
array/s 2 num_pix_per_beam data
lambda = 1.04e-4
wavelength/set 0 lambda*1e4
centre_sep = 250e-4             # 250 micron in cm
unitset = centre_sep/num_pix_per_beam # units of beam 1 determined
units/s 0 unitset               # set the units of beam 1

w0 = .0008/3.0                  #waist at fibre
w1 = .025/3.0                   #waist size at lens array
Zr = pi*w0^2/lambda              #rayleigh distance
Z = sqrt(Zr^2*(w1^2/w0^2 - 1))  #distance to lens array
R = Z + Zr^2/Z                  #phase radius of beam at lens
Yshift = .0001
array
nsize = 5                       # size of array (nsize x nsize)
clear 1 0

lensgroup/def scratch/o         #simple biconvex
```

Jump to: [Commands](#), [Theory](#)

```

        surface_lensgroup 1.02*R .004 bk7
        clap/cir/no centre_sep/2
        surface_lensgroup -1.02*R 0. air
        image focus
lensgroup/end

macro/def inner/o
    row = row + 1
    if LOCAL macro make_beam
        lensa/rec/paste 1 2 row col centre_sep centre_sep
        title row = @row, col = @col
        plot/w @Name_1.plt
        plot/l/w 1 xrad=.065 ns=128 min=-.5 max=.5
macro/end

macro/def outer/o
    col = col + 1
    row = -(nsize + 1)/2
    macro/run inner/nsize
macro/end

col = -(nsize + 1)/2

macro/def make_beam/o
    global/define 2 0 0 0 0 0 0 #reset beam 2 to z=0
    array/set 2 num_pix_per_beam
    zreff/se 2 0
    gaussian/c/c 2 1 w1 1 radx=R rady=R #define gaussian
    if LENSGROUP then
        vertex/locate/abs 0 Yshift 0 #y offset of 50 microns
        lensgroup/build/beam scratch 2
        lensgroup/run scratch 2
        global/def 2 0 0 0 0 0 0
    else
        lens 1 R
    endif
    Waves = -Yshift/R/lambda
    abr/tilt 2 Waves rnorm=1
    clap/cir/no 2 centre_sep/2
    plot/w @Name_2.plt
    plot/l/w 2 min=-.5 max=.5
macro/end

LENSGROUP = 1 # select lensgroup or ideal lens
LOCAL = 0

if [!LOCAL] macro make_beam

macro/run outer/nsize
title intensity
plot/w @Name_3.plt

```

Jump to: [Commands](#), [Theory](#)

```
plot/1 1 xrad=.065 ns=128
```

Ex67h: Array of $N \times N$ laser diodes, gaussian overall envelope

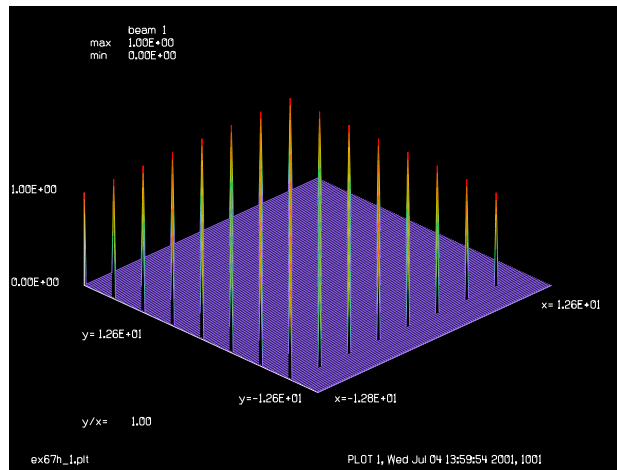
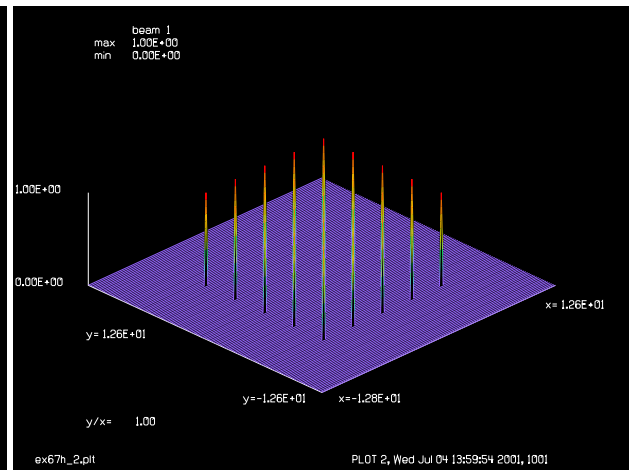
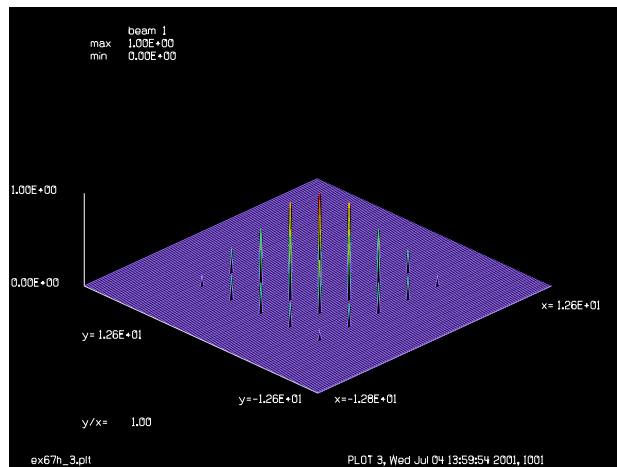
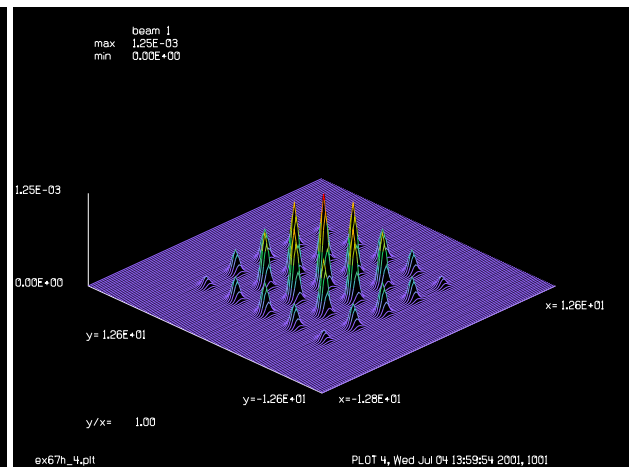
Fig. 67.32. Array of $N \times N$ delta functionsFig. 67.33. Clipped array of $N \times N$ delta functions.

Fig. 67.34. With gaussian envelope.

Fig. 67.35. After convolution with gaussian to form coherent $N \times N$ array, simulating coherent laser diodes.

Input: ex67h.inp

```
c## ex67h
#
# Example: Array of N x N laser diodes, gaussian overall envelope
#
alias Name ex67h
variable/dec/int Center Nsmall Nbig
Nsmall = 16
Nbig = 8*Nsmall
Units = .2          # pixel-to-pixel spacing
nbeam 2
array/s 1 Nbig      # make a 512 x 512 array
array/s 2 Nsmall    # small array representing each
```

Jump to: [Commands](#), [Theory](#)

```

                                # cell of the array
units/s 0 Units
#
# begin by making a field of delta functions
#

clear 2 0
Center = Nsmall/2 + 1 # pixel of center of small array
point/set/ij 2 Center Center 1
lensarray/rec/paste/all 1 2 Nsmall*Units Nsmall*Units
plot/w @Name_1.plt
plot/l 1 1 ns=128
#
# Trim the field to the inner 5 X 5 array
#
clap/s 1 .35*Nbig*Units
plot/w @Name_2.plt
plot/l 1 1 ns=128

#
# make a gaussian envelope over the array of delta functions
#
array/s 2 Nbig          # redefine Array 2
units/s 2 Units
gauss/cir 2 1 Nbig*Units/3
mult/beam 1 2
plot/w @Name_3.plt
plot/l 1 1 ns=128

#
# Now convolve with a gaussian function (other shapes may be used
# by defining the beam shape in Array 2 and using convol/beam 1 2).
#
convol/gaus 1 3*Units
plot/w @Name_4.plt
plot/l 1 1 ns=128

```


Ex68: Resonator with Brewster windows

Table. 68.1. Table of Ex68 examples

Ex68a: Resonator with polarization by JSURF, 1 micron wavelength	1
Ex68b: Resonator with polarization by JSURF, 100 micron wavelength	4

This example illustrates a resonator with an amplifier rod which uses Brewster windows. Example Ex68a illustrates a laser operating in the near IR at 1 microns. Ex68b.inp works in the long wavelength IR at 100 microns. The Brewster window works perfectly for a collimated beam at the proper angle of incidence. This may be checked by setting `switch = 1`, which switches out all the apertures, mirrors, and diffraction. With a divergent beam, the efficiency is not 100%.

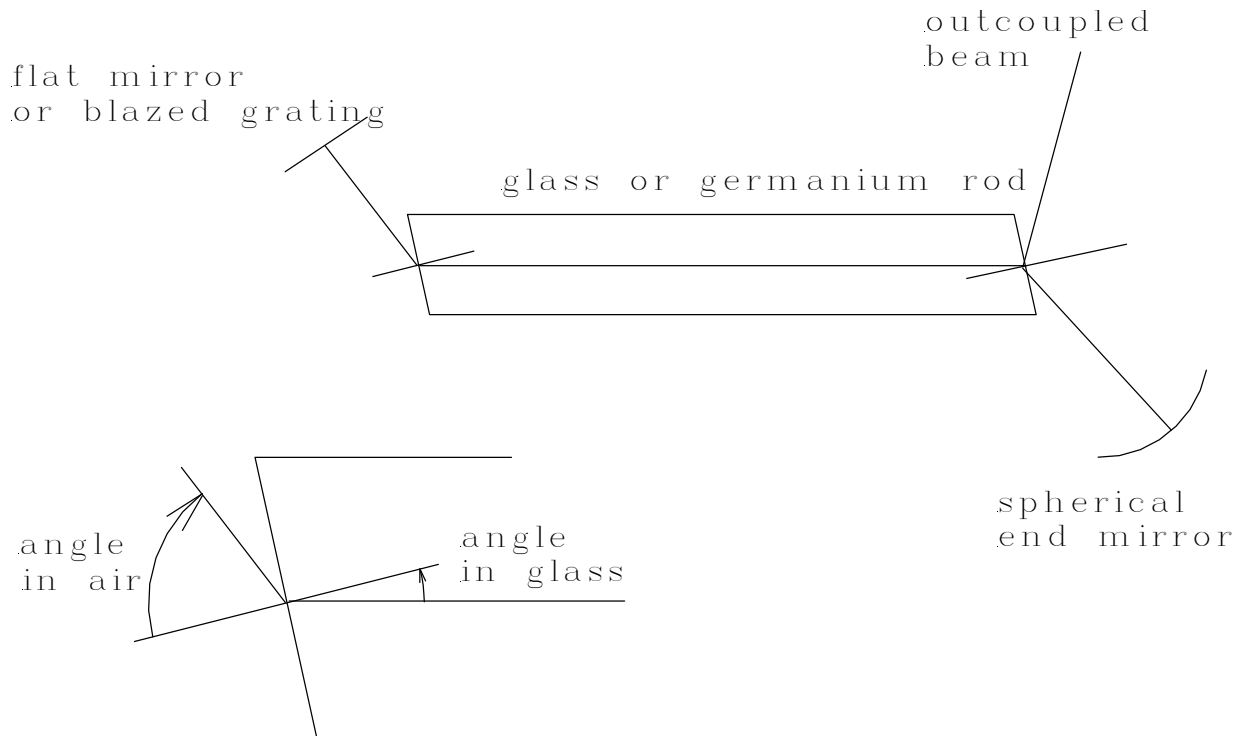


Fig. 68.1. Configuration for laser with glass or germanium rod with Brewster angle windows.

Ex68a: Resonator with polarization by JSURF, 1 micron wavelength

Input: `ex68a.inp`

```

c## ex68a!382705658285276
c
c Example 68a: Polarization with JSURF, nonnormal incidence
c
echo/on
variab/dec/int switch pass
arra/s 1 64 64 1                                # set array size
nbeam 2                                           # beam 1 to be transmitted

```

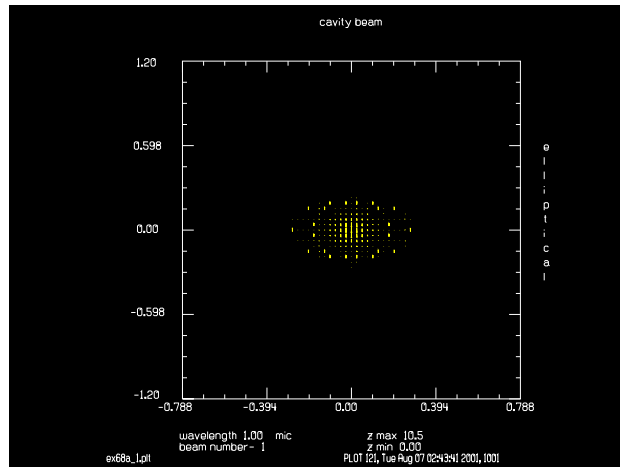


Fig. 68.2a. Polarization plot of cavity mode — Case a.

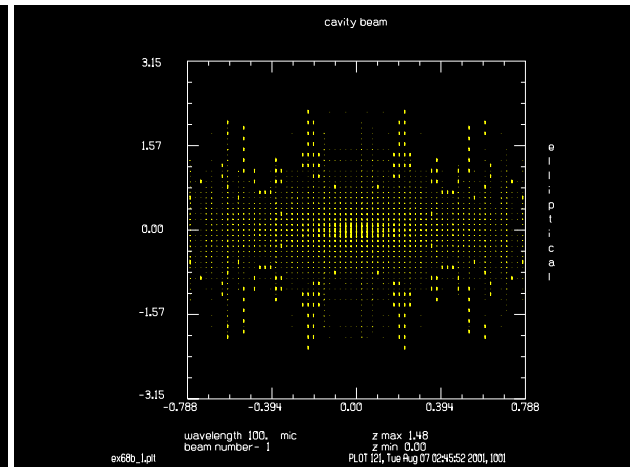


Fig. 68.2b. Polarization plot of cavity mode — Case b.

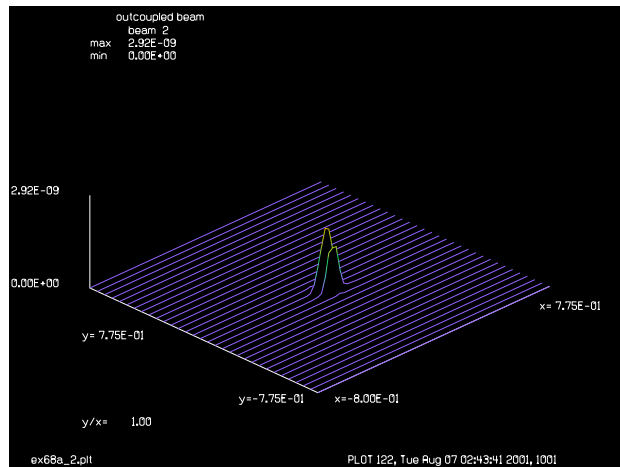


Fig. 68.3a. Polarization plot of outcoupled beam—Case a.

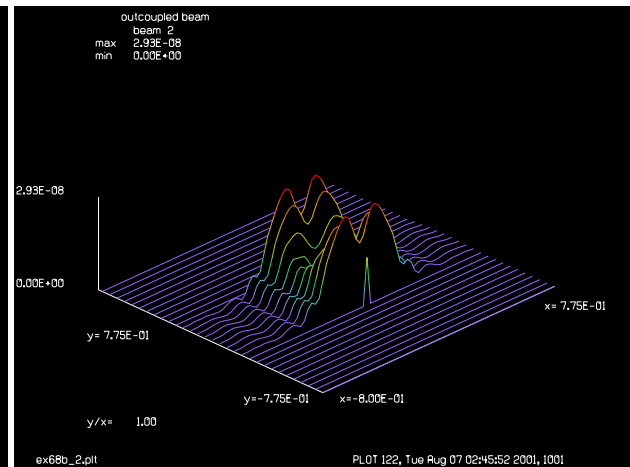


Fig. 68.3b. Polarization plot of outcoupled beam—Case b.

```

# beam 2 to be reflected

arra/s 2 64 64 1 data

lambda = 1                # wavelength 1 microns
air = 1                   # index of air
glass = 1.52              # index of glass
inair = 56.659            # input incidence angle
inglass = 33.340581       # output incidence angle
claprad = .25             # rod aperture
eclaprad = .3             # elliptical aperture dimension
sphclap = .10            # clear aperture for spherical mirror
c
wavelength/set 1 lambda
units/s 1 .025 .025      # set units/s to .025 cm
clear 2 0.
energy/norm 1
switch = 1               # switch = 0 switches out apertures

```

Jump to: [Commands](#), [Theory](#)

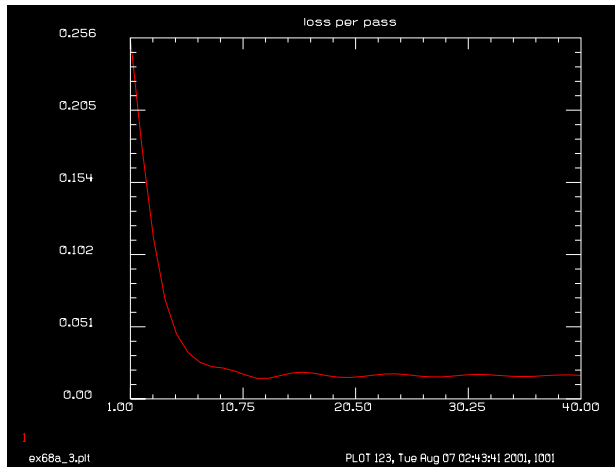


Fig. 68.4a. Convergence history—Case a.

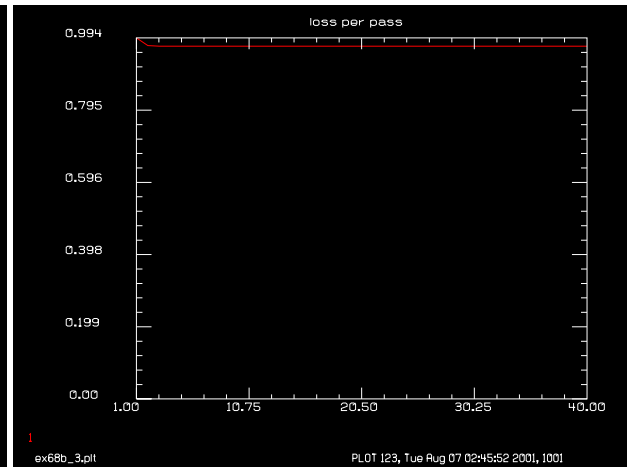


Fig. 68.4b. Convergence history—Case b. Very high losses because of Fresnel reflection.

```

c
jones/far 1 90

beams/off 2
macro/def nearir/o
    pass = pass + 1
    if switch != 0 then
        mirror/flat 1      # flat mirror represents blazed grating
        prop 2
    endif
    copy 1 2
c
c Fresnel transmission and reflection for beams 1 and 2
c
jsurf/fresnel/trans 1 air glass tx=inair    # into glass
jsurf/fresnel/refl 2 air glass tx=inair    # output beam
if switch != 0 then
    clap/c/n 1 claprad
    clap/e/n 2 eclaprad claprad
    prop 50
    title cavity beam
    plot/watch ex68a_1.plt
    plot/e 1
    title outcoupled beam
    plot/watch ex68a_2.plt
    plot/l 2
endif
c
c leave the glass rod
c
jsurf/fresnel/trans 1 glass air tx=inglass  # out of germanium
if switch != 0 then
    prop 2

```

```

    mirror/sph 1 r=500    # spherical end mirror
    clap/c/n 1 sphclap
    prop 2
endif
jsurf/fresnel/trans 1 air glass tx=inair # into glass
if switch != 0 then
    clap/c/n 1 claprad
    prop 50
    clap/c/n 1 claprad
endif
jsurf/fresnel/trans 1 glass air tx=inglass # out of germanium
if switch != 0 then
    prop 2
    variab/set energy 1 energy
    udata/set pass pass 1-energy
    plot/watch ex68a_3.plt
    title loss per pass
    plot/udata min=0
endif
energy/norm 1 list          # normalize energy each pass
macro/end
if switch = 0 then
    macro nearir
    end
endif
resonator/name nearir      # define name of resonator macro
resonator/eigen/test 1     # test macro to determine properties
resonator/eigen/set 1      # set resonator to eigenmode
pass = 0                   # reset pass counter
jones/far 1 45             # rotate polarization 45 deg.
                           # to seed both polarizations
energy/norm 1              # normalize energy to start
reson/run 40               # run resonator 40 times
end

```

Ex68b: Resonator with polarization by JSURF, 100 micron wavelength

Input: ex68b.inp

```

c## ex68b!293200403953352
c
c Example 68b: Polarization with JSURF, nonnormal incidence
c
echo/on
variab/dec/int switch pass
arra/s 1 64 64 1          # set array size
nbeam 2                   # beam 1 to be transmitted
                           # beam 2 to be reflected

arra/s 2 64 64 1 data

lambda = 100              # wavelength 100 microns
air = 1                   # index of air
germanium = 4             # index of germanium

```

Jump to: [Commands](#), [Theory](#)

```

inair = 75.963757      # input incidence angle
inger = 14.036244      # output incidence angle
claprad = .25          # rod aperture
eclaprad = 2.061012    # elliptical aperture dimension
sphclap = .15          # clear aperture for spherical mirror
c
wavelength/set 1 lambda
units/s 1 .025 .025    # set units to .025 cm
clear 2 0.
energy/norm 1
switch = 1             # switch = 0 switches out apertures
                        # diffraction, lenses to test
                        # Brewster's angle

c
jones/far 1 90         # select vertical polarization for
                        # first pass

beams/off 2
macro/def longir/o
    pass = pass + 1
    if switch != 0 then
        mirror/flat 1    # flat mirror represents blazed grating
        prop 2
    endif
    copy 1 2
c
c Fresnel transmission and reflection for beams 1 and 2
c
jsurf/fresnel/trans 1 air germanium tx=inair # into germanium
jsurf/fresnel/refl 2 air germanium tx=inair  # output beam
if switch != 0 then
    clap/c/n 1 claprad
    clap/e/n 2 eclaprad claprad
    prop 50
    title cavity beam
    plot/watch ex68b_1.plt
    plot/e 1
    title outcoupled beam
    plot/watch ex68b_2.plt
    plot/l 2
endif
c
c leave the germanium rod
c
jsurf/fresnel/trans 1 germanium air tx=inger # out of germanium
if switch != 0 then
    prop 2
    mirror/sph 1 r=500
    clap/c/n 1 sphclap
    prop 2
endif
jsurf/fresnel/trans 1 air germanium tx=inair # into germanium
if switch != 0 then
    clap/c/n 1 claprad
    prop 50

```

```

        clap/c/n 1 claprad
    endif
    jsurf/fresnel/trans 1 germanium air tx=inger # out of germanium
    if switch != 0 then
        prop 2
        variab/set energy 1 energy
        udata/set/ pass pass 1-energy
        plot/watch ex68b_3.plt
        title loss per pass
        plot/udata min=0
    endif
    energy/norm 1 list # normalize energy each pass
macro/end
if switch = 0 then
    macro longir
    end
endif
resonator/name longir # define name of resonator macro
resonator/eigen/test 1 # test macro to determine properties
resonator/eigen/set 1 # set resonator to eigenmode
pass = 0 # reset pass counter
jones/far 1 45 # rotate polarization 45 deg.
# to seed both polarizations
energy/norm 1 # normalize energy to start
reson/run 40 # run resonator 40 times
end

```

Ex69: Rate equations, transient response

Table. 69.1. Table of Ex69 examples

Ex69a: Rate equation gain and mode competition	1
Ex69b: Rate equation gain for ruby laser	5
Ex69c: Rate equations, single step	8
Ex69d: Semiconductor gain.	9
Ex69e: Three-level gain, single upper state.	13
Ex69f: Numerical example of rate equation gain	18
Ex69g: Numerical example of single level, three level gain	21
Ex69h: Rate equations for ruby	21
Ex69i: Rate equations for general, three level laser	21
Ex69j: Steady-state rate equation solution.	21
Ex69k: Rate equations for single level, three-level laser, multiple steps	22

This example illustrates rate equation gain and longitudinal mode competition.

Ex69a: Rate equation gain and mode competition

Example 69a shows a simple calculation of the time response of power in a resonator considering the pumping of the medium. The array size has been reduced to 4 x 4 and diffraction has been neglected to study only the time response and to increase the speed of the calculations.

The pumping is started at $t = 0$. Initially the population inversion builds up with no appreciable depletion by stimulated emission because the optical field is too small. At about 20 cycles the optical field builds up enough to extract the energy stored in the population inversion. This causes the strong initial spike. Two longitudinal modes are modeled simultaneously. The Fabry-Perot line width causes the mode which is off line center to have larger round-trip losses and it begins to lose power while the mode on line center grows. At pass 100, the cavity length is changed so that Mode 2 is now on line center. Mode 2 now begins to dominate. The average power is essentially constant after the initial transient when the medium becomes saturated.

Example 69b shows the transient response of a ruby laser which may be described by 3-level kinetics. See GLAD Theory Manual, Chap. 9 for a discussion of 3-level kinetics and a sample calculation.

Example 69c shows an elementary step to illustrate how small signal gain may be calculated. It illustrates the steps found in the description of `gain/rate` in the GLAD Commands Manual.

For a single mode operating on line center,

line width function for zero offset

$$g(\Delta v_k = 0) = \frac{2}{\pi \text{Width}} = \frac{2}{\pi 1.45 \times 10^{13}} = 4.3905 \times 10^{-14} \text{ sec}^{-1} \quad (69.1)$$

For $v_k = 3.57 \times 10^{14}$ Hz, $\lambda_k = c/v_k = 0.84 \times 10^{-5}$, media wavelength $\lambda_k/n = c/(v_k n) = 2.50673 \times 10^{-5}$. The photon energy conversion is $h\nu = 2.365482 \times 10^{-19}$

Einstein B-coefficient (cross section)

$$B_k = \frac{\lambda^2 g(\Delta\nu_k)}{8\pi n^2 \tau_{\text{spont}}} = \frac{(0.84 \times 10^{-4})^2 4.3905 \times 10^{-14}}{8\pi 3.35^2 3 \times 10^{-9}} = 3.659034 \times 10^{-16} \text{ cm}^2 \quad (69.2)$$

excitations per cm³

$$\frac{1}{2} \Delta N^0 = \frac{1}{2} R \Delta t = \frac{1}{2} \frac{1.93 \times 10^6}{h\nu} \times 10^{-10} \text{ sec} = 4.0795 \times 10^{14} \quad (69.3)$$

half of the population inversion for a pumping rate of 1.93e6 watts per cm and 1e-10 sec pumping. As explained in the GLAD Commands Manual for `gain/rate`, half the pumping is assumed to occur before the gain and half after. Since we are taking only one step in Ex69c, the pumping is half the specified rate.

decrease in upper level due to spontaneous decay and increase in lower level due to spontaneous decay.

$$\frac{1}{2} \Delta N^0 e^{-\frac{2\Delta t}{\tau_{\text{spont}}}} = 4.0795 e^{-\frac{2e-10}{3e-9}} = 3.8164e14 \quad (69.4)$$

small signal gain coefficient

$$\gamma(0) = B_k \Delta N^0 = 3.66e-16 \times 3.8164e14 = 0.1397 \quad (69.5)$$

amplification

$$e^{\gamma(0)\Delta z} = e^{0.1397 \times 0.89} = 1.132 \quad (69.6)$$

The amplification value is in good agreement with Ex69c.inp. The saturation value is

$$I_s = \frac{h\nu}{Bt_2} = \frac{(6.626e-34)(3.57e14)}{(3.66e-16)(3e-9)} = 2.154e5 \text{ w-cm}^{-2} \quad (69.7)$$

The population inversion of 3.816e14 excitations per cm³ yields an energy density according to energy density $\times (h\nu)/2 = 4.509e-5 \text{ J/cm}$. The maximum intensity that may obtained for a temporal pulse of length 1e-10 and a gain length of 0.89 cm is 4.509e-5 J/cm = 4.017e5 w/cm, in good agreement with Ex. 69c.inp.

Consider the χ' factors that introduce a change in index for longitudinal lines that are off line center. A population inversion of 3.812e14 excitations per cm³ yields the values of $\chi''(\nu)$ and $\chi'(\nu)$

$$\chi''_m(\nu) = (N_1 - N_2) \frac{\lambda^3}{16\pi^2 \tau_{\text{spont}} n} g(\nu_m) \quad (69.8)$$

$$\chi'_m(\nu) = \frac{2(\nu_{off} + m\Delta\nu_c)}{\Delta\nu} \chi''_m(\nu) \quad (69.9)$$

For a population inversion of 3.812×10^{14} excitations per cm^3 , a line off-center by $\delta\nu = 5 \times 10^{11}$ Hz, line width of $\Delta\nu = 1.45 \times 10^{13}$ Hz, $\lambda = 2.506731 \times 10^{-5}$ cm, $\tau_{spont} = 3 \times 10^{-9}$ sec, $n = 3.35$, we have

$$g(\nu) = \frac{\Delta\nu}{2\pi[(\nu_{off} + m\Delta\nu_c)^2 + (\frac{\Delta\nu}{2})^2]} = \frac{1.45 \times 10^{13}}{2\pi[(5 \times 10^{11})^2 + (\frac{1.45 \times 10^{13}}{2})^2]} = 4.369 \times 10^{-14} \quad (69.10)$$

$$\chi''(\nu) = (N_1 - N_2) \frac{\lambda^3}{16\pi^2 \tau_{spont} n} g(\nu) = (-3.812 \times 10^{14}) \frac{\lambda^3}{16\pi^2 3 \times 10^{-9} (3.35)} 4.369 \times 10^{-14} = -1.6533 \times 10^{-7} \quad (69.11)$$

$$\chi'_m(\nu) = \frac{2(\nu_{off} + m\Delta\nu_c)}{\Delta\nu} \chi''_m(\nu) = \frac{2(5 \times 10^{11})}{1.45 \times 10^{13}} -1.6533 \times 10^{-7} = -1.14018 \times 10^{-8} \quad (69.12)$$

The optical phase is

$$\frac{k\chi'(\nu)}{2n^2L} = \frac{\pi\chi'(\nu)}{n\lambda} L = \frac{\pi(-1.14018 \times 10^{-8})}{(3.35) 2.56073 \times 10^{-5}} 0.89 = 3.7963 \times 10^{-4} \text{ radians} \quad (69.13)$$

Input: ex69a.inp

```
c## ex69a
c
c Example 69a: Rate equations
c
c This is an example of the use of rate equations
c The resonator is similar to that of a laser diode
c with flat faces.
c
c The device will be allowed to converge for 100 passes
c and then the length changed to put Beam 2 on the line
c center causing Beam 1 to decay and Beam 2 to increase.
c
c Establish initial units and a gaussian field distribution
c
c Beams Use
c 1 longitudinal mode on line center
c 2 slightly off line center
c 3 pump distribution array
c
c variab/dec/int pass switch
c array/s 1 4
c nbeam 3 # set number of beams
```

Jump to: [Commands](#), [Theory](#)

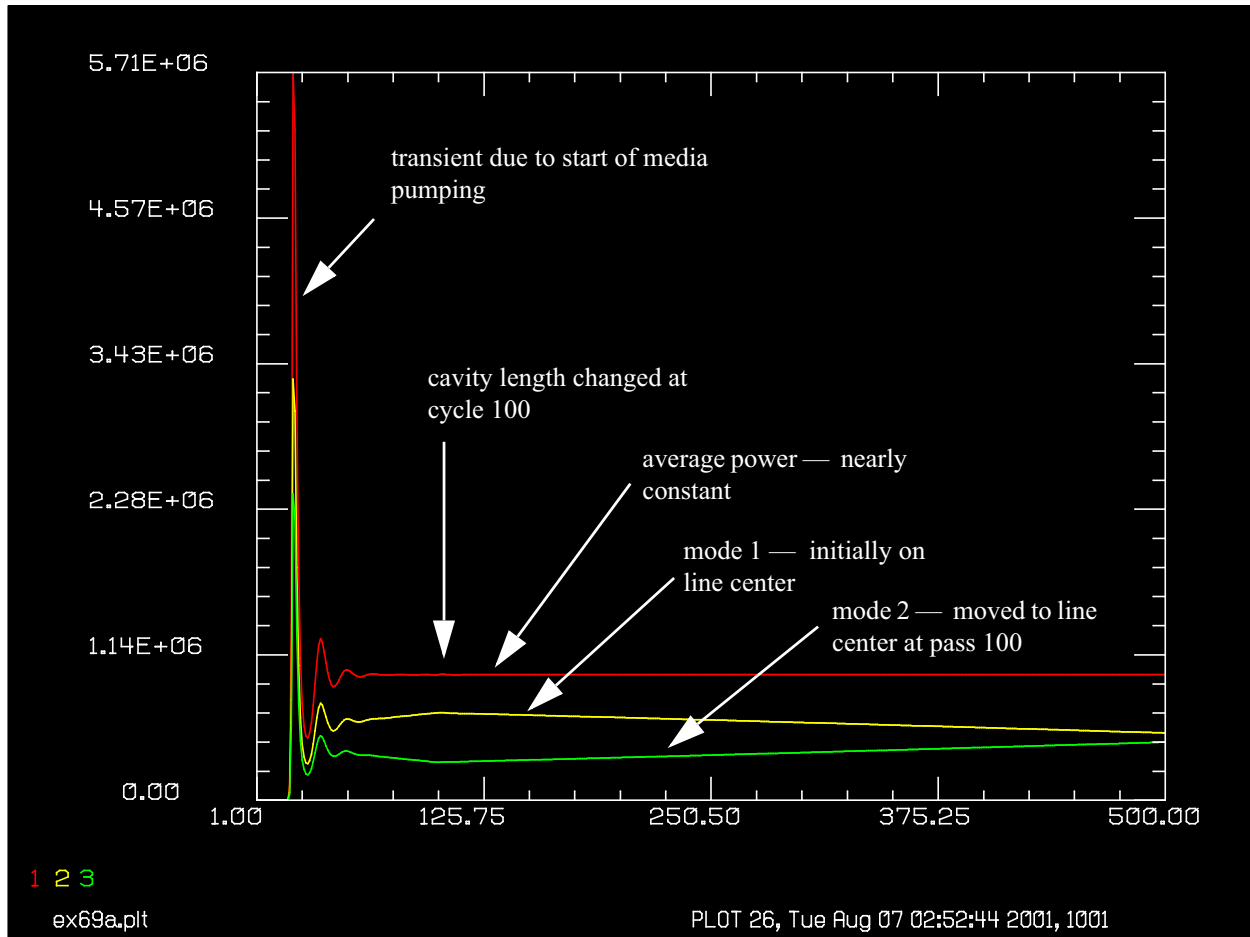


Fig. 69.1. Temporal response of competing modes from the start of medium pumping. The cavity length is changed at pass 100 causing the relative loss on the two longitudinal modes to reverse.

```

array/s 3 4 4 1 data      # make medium array polarized
units/s 0 1
wavelength/set 0 .84 3.35 # set wavelength and index
clear 1 1e-4              # set initial irradiance
clear 2 1e-4
clear 3 1.93e6            # set pump rate in
                           # watts per cm**2
gain/rate/n2pump 3 3      # Modify beam 3 to put
                           # the pump rate in the real
                           # word
                           # inversion will go in the
                           # imaginary word

c
width = 1.45e13            # Set line width, hz
center = 3.57e14           # Set line center, hz
tspont = 3e-9             # Set spontaneous emission time, sec
t20 = 3e-1                # Set decay time for level 2, sec
t10 = 1e-11               # Set decay time for level 1, sec
mode_sep = 5e11           # Set longitudinal mode separation, hz
c
c Set offset from line center of first mode, 0 hz

```

Jump to: [Commands](#), [Theory](#)

```

c Set fractional pumping into level 1, nlpump, 0
c
  gain/rate/set width center tspont t20 t10 mode_sep 0 0
  gain/rate/list
  pack/set 1 2 3          # pack all 3 beams
  pass = 0                 # initialize pass counter
macro/def ex69a/o
  pack/in                 # pack beams
  pass = pass + 1         # increment pass counter
c
c Set gain length of .89 cm
c Set pump time of 1e-10
c Set number of steps over which distance is to be taken
c
  gain/rate/step .89 1e-10 10 # implement rate eq. gain
  pack/out                 # unpack beams
  mult 1 .396              # Fresnel reflection losses
  mult 2 .396
  variab/set peak1 1 peak  # record peak irradiance
  variab/set peak2 2 peak
  sum = peak1 + peak2       # record sum of Beam 1 and 2
  udata/set pass pass sum peak1 peak2 # store for the record
  switch = mod(pass,20)    # plot every 20th pass
  if switch = 0 then
    plot/udata/seq         # make udata plot
  endif
macro/end
  plot/udata/set y01 y02 y03 # specify beams to be displayed
  plot/watch ex69a.plt      # display record
  mac ex69a/100             # stabilize for 100 passes
c
c shift cavity length
c
c Set frequency offset so Beam 2 is now on line center
c
  gain/rate/set width center tspont t20 t10 mode_sep -5e11 0
  mac ex69a/400             # run 400 passes to let
                           # Beam 2 to begin to grow

  plot/udata/seq
end

```

Ex69b: Rate equation gain for ruby laser

Input: ex69b.inp

```

c## ex69b
c
c Example 69b: Rate equations for ruby laser
c
c This is an example of the use of rate equations
c for a ruby laser.
c

```

Jump to: [Commands](#), [Theory](#)

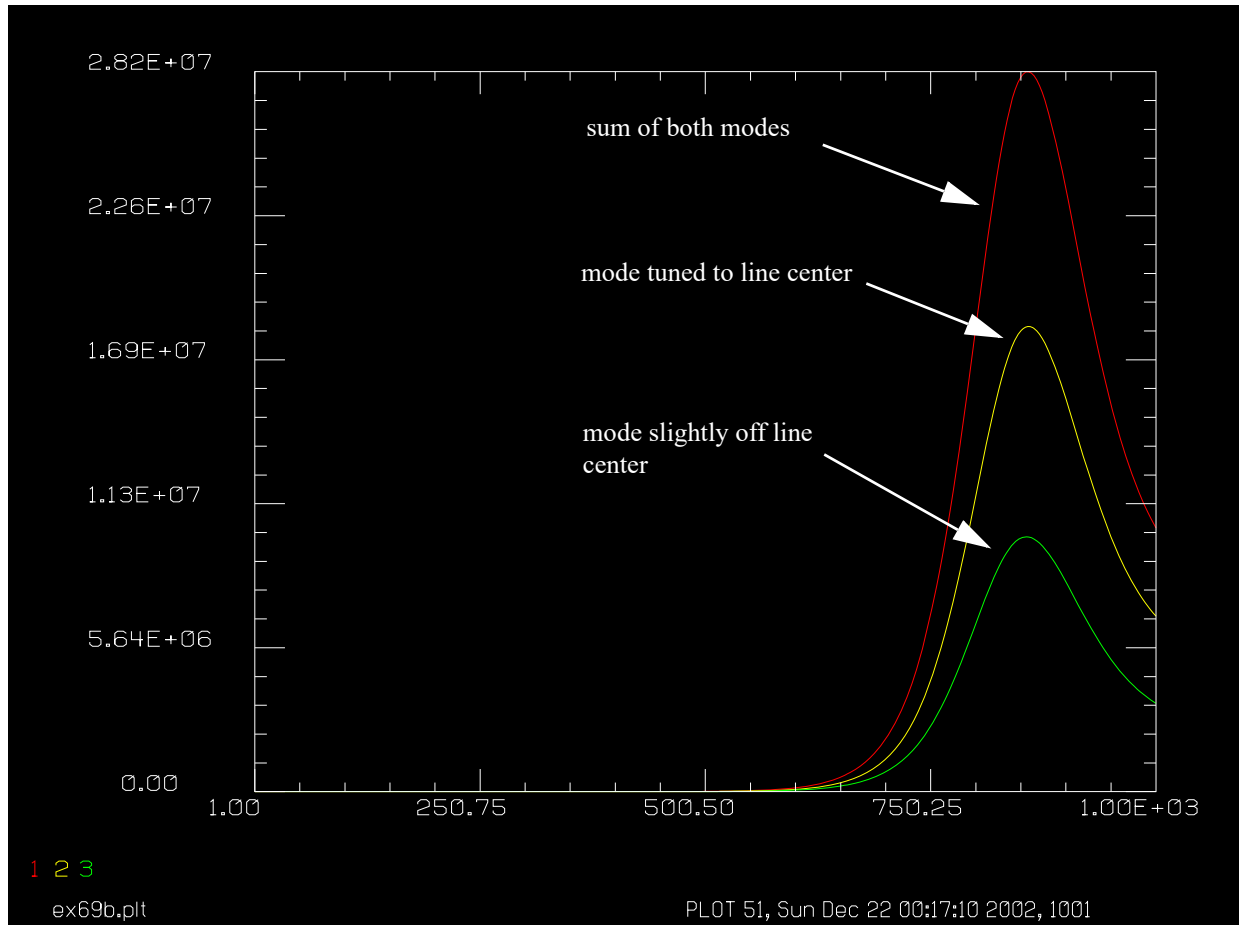


Fig. 69.2. Transient response of ruby laser showing 3-level kinetics. X-axis is in nanoseconds. Two modes are shown: one on line center, and one slightly off line center. The sum of the two modes is also shown.

```

c Beams      Use
c 1          longitudinal mode on line center
c 2          slightly off line center
c 3          media distribution array
c 4          work array
c
c variab/dec/int pass switch
c lambda = .694325e-4          # wavelength
c v = c/lambda                 # frequency of radiation
c hv = h*v                     # Joules/photon
c Ntot = 1.58e19               # total potential population density
c Jtot = Ntot*hv               # total potential energy density
c Temp = 293.                  # temperature
c Index = 1.77                 # index of refraction
c N2level = Ntot               # set to maximum inversion
c J2level = N2level*hv         # energy density of N2 level
c array/s 1 4
c nbeam 4                      # set number of beams
c array/s 3 4 4 1 data        # make medium array polarized
c units/s 0 .1
c wavelength/set 0 lambda*10000 1.77 # set wavelength and index

```

Jump to: [Commands](#), [Theory](#)

```

clear 1 1e-2          # set initial irradiance
clear 2 1e-2
sigma = 2e-20         # pump cross section, cm^2
Pump_int = 1e8        # pump irradiance, watts/cm^2
r_value = Pump_int*sigma
clear 4 r_value
gain/ruby/pump 4 3     # Modify beam 3 to put
clear 4 J2level        # set N2 level, Joules per cm**3
gain/ruby/n2level 4 3  # Modify beam 3 to set N2level

clear 4 Jtot           # set N-total, Joules per cm**3
gain/ruby/ntotal 4 3   # Modify beam 3 to put

clear 4 Temp
gain/ruby/temperature 4 3 # set temperature distribution

# the pump rate in the real
# word
# inversion will go in the
# imaginary word

c
width = 3.3e11         # Set line width, hz
mode_sep = 1.e10       # Set longitudinal mode separation, hz
c
    gain/ruby/set width mode_sep 0.
    gain/ruby/list
    pack/set 1 2 3      # pack all 3 beams
    pass = 0            # initialize pass counter
macro/def ex69b/o
    pack/in             # pack beams
    pass = pass + 1     # increment pass counter
c
c    Set gain length of .89 cm
c    Set pump time of 1e-10
c    Set number of steps over which distance is to be taken
c
    gain/ruby/step 1. 1e-9 # implement rate eq. gain
    pack/out              # unpack beams
    Trans = 1 - 2*((Index-1.)/(Index+1.))^2 # two interfaces
    mult 1 Trans          # Fresnel reflection losses
    mult 2 Trans
    variab/set peak1 1 peak # record peak irradiance
    variab/set peak2 2 peak
    sum = peak1 + peak2    # record sum of Beam 1 and 2
    udata/set pass pass sum peak1 peak2 # store for the record
    switch = mod(pass,20)  # plot every 20th pass
    if switch = 0 then
        plot/udata/seq    # make udata plot
    endif
macro/end
    plot/udata/set y01 y02 y03 # specify beams to be displayed
    plot/watch ex69b_1.plt    # display record
    mac ex69b/1000           # stabilize for 400 passes
c

```

Jump to: [Commands](#), [Theory](#)

```

c  shift cavity length
c
c  Set frequency offset so Beam 2 is now on line center
c      2
c      plot/udata/seq
c      end

```

Ex69c: Rate equations, single step

Input: ex69c.inp

```

c## ex69c
c
c  Example 69c: Rate equations, single step
c
c  This is an example of the use of rate equations
c  for a single elementary step. See notes in Ex69 for
c  calculations of small signal gain and saturation gain
c
c
c  variab/dec/int pass switch
c  array/s 1 4
c  nbeam 2 # set number of beams
c  array/s 2 4 4 1 data # make medium array polarized
c  units/s 0 1
c  wavelength/set 0 .84 3.35 # set wavelength and index
c  clear 1 1e-4 # set initial irradiance
c  clear 2 1.93e6 # set pump rate in
c                  # watts per cm**3
c  gain/rate/n2pump 2 2 # Modify beam 3 to put
c                  # the pump rate in the real
c                  # word
c                  # inversion will go in the
c                  # imaginary word
c
c  width = 1.45e13 # Set line width, hz
c  center = 3.57e14 # Set line center, hz
c  tspond = 3e-9 # Set spontaneous emission time, sec
c  t20 = 3e-1 # Set decay time for level 2, sec
c  t10 = 1e-11 # Set decay time for level 1, sec
c  mode_sep = 5e11 # Set longitudinal mode separation, hz
c
c
c  Set offset from line center of first mode, 0 hz
c  Set fractional pumping into level 1, n1pump, 0
c
c  gain/rate/set width center tspond t20 t10 mode_sep 0 0
c  gain/rate/list
c  pack/set 1 2 # pack all 3 beams
c  variab/set Peak1 1 peak
c  pack/in # pack beams
c  gain/rate/step .89 1e-10 1 # implement rate eq. gain

```

Jump to: [Commands](#), [Theory](#)

```

pack/out                                # unpack beams

variab/set Peak2 1 peak
Amplification = Peak2/Peak1 list # small signal amplification
c
c now try very high incident irradiance to test saturated gain
c
  clear 1 1e8                            # set initial irradiance
  clear 2 1.93e6                          # set pump rate in
                                          # watts per cm**2
gain/rate/n2pump 2 2                    # Modify beam 3 to put
                                          # the pump rate in the real
                                          # word
                                          # inversion will go in the
                                          # imaginary word

variab/set Peak1 1 peak
pack/in                                # pack beams
  gain/rate/step .89 1e-10 1 # implement rate eq. gain
pack/out                                # unpack beams

variab/set Peak2 1 peak
DeltaI = Peak2 - Peak1 list # saturated irradiance increase

```

Ex69d: Semiconductor gain

Semiconductor gain is described in the GLAD Theory Manual, [Sect. 9.4](#). This example provides a simple calculation for verification showing all steps.

Input: ex69d.inp

```

c## ex69d
c
c Example to illustrate and verify semiconductor gain
c based on a model from Coldren and Corzine
c
# write file containing calculations
write/disk temp1.txt/o
# Nonradiative recombination coefficient for decay rate of inversion. [sec^-1]
A = 1e7      list

# Bimolecular recombination coefficient for decay rate of inversion. [cm^2/sec]
B = 1e-10    list

# Auger recombination rate for decay rate of inversion per Eq. (135). [cm^6/sec]
C = 1e-30    list

# Initial population density. [cm^-3]
N = 1e19     list

# Small signal gain coefficient. [cm^-1]
g0 = 2000    list

```

Jump to: [Commands](#), [Theory](#)

```

# Transparency population density of excitations. [cm^-3].
Ntr = 2e18      list

# Numerical adjustment to keep the argument of the logarithm positive. [cm^-3]
Ns = 1e18      list

# Length. [cm]
L = 200e-7

# Ratio of mode bandwidth to spontaneous emission bandwidth, bsp.
Beta = .1

# Spontaneous emission rate [cm^-3/sec]
Rsp = B*N^2      list

# Wavelength/ [cm]
Lambda = 1e-4

# Index of refraction. (Just a guess).
Index = 3.567

# Material wavelength.
xlambda = Lambda/Index

# Frequency. [cm^-1]. "c" is the speed of light -- predefined in GLAD.
nu = c/Lambda      list

# Quantity h*nu. "h" is Plank's constant -- predefined in GLAD.
hnu = h*nu      list

# Pixel-to-pixel spacing, x-direction.
dX = .01

# Pixel-to-pixel spacing, y-direction.
dY = dX

# Time for one pass. [sec]
dT = L*Index/c      list

# Solid angle subtended by array.
Omega = Lambda^2/dX/dY      list

# Spontaneous emission is removed from population inversion and added
# into optical field as a delta-correlated random complex amplitude.
SpontEmmisionLoss = Beta*Omega*hnu*B*N^2      # w^2/s/cm

# Generation rate. (Just a guess.)
Pgen = .1      list # 1.0 w/cm^3

# Generation for
dN = Pgen*dT/hnu/2      list # half of generation

# Add half of generation at beginning of calculation and half at end.

```

Jump to: [Commands](#), [Theory](#)


```

N0 = N+dN                                list

# Alternate variable name.
dpop = dN                                list

# Reset to N, so expression looks like equations in text.
N = N0

# Calculate inverse of Tau. (Might want to check that this is non-zero).
InvTau = A + B*N + C*N^2                list

# Calculate Tau.
Tau = 1./InvTau                          list

# Population density loss rate.
N_Tau = N/Tau                            list

# Loss of population over time dT.
dp = N0*(1.-exp(-dT/Tau))               list

# Correct population by spontaneous emission loss.
N0 = N0 -dp                              list

# Calculate gain as a function of N.
gN = g0*log((N+Ns)/(Ntr+Ns))             list

# Calculate cross section.
sigma = gN/N                             list
# Set initial optical field. Also, will grow from spontaneous emission.
I0 = 1

# Set to names of internal variables, for easy code check.
flux0 = I0                               list

# Calculate maximum possible population density.
Nmax = N0 + (I0*dT)/(hnu*L)              list

# Calculate maximum possible optical intensity.
Imax = I0 + N0*hnu*L/dT                  list

# Calculate an internal variable for code check. Same as Nmax.
coef3 = N0 + I0*dT/L/hnu

# Calculate the coefficient to convert population to intensity.
xa = dT/L/hnu                            list

# Calculate an internal variable for code check. Same as Nmax.
ximax = coef3/xa                          list

# Calculate internal variable for code check.
args = sigma*coef3*L                      list

# Calculate intensity after propagation L, based on Frantz-Nodvik.
I1 = Imax*I0/(I0 + (Imax - I0)*exp(-sigma*Nmax*L))

```

Jump to: [Commands](#), [Theory](#)

```

# Set variable for code check.
flux2 = I1                                list

# Calculate depletion of population by conservation of energy.
N1 = Nmax - I1*dt/hnu/L                    list

# Calculate change in intensity.
dint2 = I1 - I0

# Calculate change in popylation.
dpopw2 = dint2*xa                            list

# Calculate intensity amplification.
arg1 = 1. + dint2/flux0                      list

# Calculate amplitude amplification.
arg1 = sqrt(arg1)                            list

# display half of generation increase of population. (Calcualted above.)
dpop=                                     # half of noise

# Set to an internal variable name.
dpopsum = dpopw2                            list

# Spectral width. (Just a guess. We will work at line center for this
calculation.)
Width = 1.45e13                             # Set line width, hz

# Calculate line width coefficient for internal code check.

gaincv = 2./(pi*Width)                      list
write/disk/off/close

# Set up a 2 x 2 array for convenience in checking the calculation.
array/s 1 2 data
# Beams          Use
# -----      ---
# 1              Optical array
# 2              Gain array, unpolarized.
nbeam 2 data
units/set 0 dX dY
# Set line center. This calculation is assumed to be at line center,
# so this is not a real issue.
Center = nu                                # Set line center, hz

# Guess at longitudinal mode separation. Not used here. User should
# calculate proper values.
Cavity = 5e11                             # Frequency separation between longitudinal modes

# Set semiconductor cavity parameters.
gain/semi/cavity width=Width center=Center cavity=Cavity offset=0

# Set semiconductor gain parameters.

```

Jump to: [Commands](#), [Theory](#)

```

gain/semi/set ntr=Ntr ns=Ns gain=g0 a=A b=B c=C beta=Beta

set/definitions/on      # turn definitions on or off
                        # A 5.1 feature only implemented for gain/semi in 5.0
# List current settings.
gain/semi/list

clear 2 0                # Clear gain array
clear 1 Pgen             # Set generation rate
                        # Apply aperturing and other customization
                        # to control generation distribution
gain/semi/pgen 1 2       # Set generation into gain array (beam 2).
field 2                  # List real and imaginary parts of gain array
clear 1 N*hnv            # Set initial population inversion. Could be zero.
gain/semi/nlevel 1 2     # Set inversion into gain array.
field 2                  # List real and imaginary parts of gain array
clear 1 I0               # Set initial optical field.
wavelength/set 0 Lambda*1e4 Index  # Set wavelength
# Implement gain
pack/set 1 2
pack/in
    gain/semi/step L dT
pack/out
field 1                  # List optical field to show correct gain.
write/disk/off/close

```

Ex69e: Three-level gain, single upper state

See GLAD Theory Manual, [Sect. 9.5](#). This example provides a simple calculation for verification showing all steps.

Input: ex69e.inp

```

c## ex69e
c
c Example to illustrate and verify semiconductor rate equation gain
c

c Beams    Use
c 1        longitudinal mode on line center
c 2        slightly off line center
c 3        pump distribution array
c

array/s 1 4
nbeam 3                # set number of beams
array/s 3 4 4 1 data   # make medium array polarized
units/s 0 1
Index = 3.35
wavelength/set 0 .84 Index  # set wavelength and index
clear 1 1e-4            # set initial irradiance
clear 2 1e-4

```

Jump to: [Commands](#), [Theory](#)

```

clear 3 1.93e6          # set pump rate in
                        # watts per cm**2
gain/rate/n2pump 3 3    # Modify beam 3 to put
                        # the pump rate in the real
                        # word
                        # inversion will go in the
                        # imaginary word

c
width = 1.45e13          # Set line width, hz
center = 3.57e14         # Set line center, hz
tspont = 3e-9            # Set spontaneous emission time, sec
t20 = 3e-1              # Set decay time for level 2, sec
t10 = 1e-11             # Set decay time for level 1, sec
mode_sep = 5e11          # Set longitudinal mode separation, hz
time = 1.e-10            # Set incremental time
length = .89             # Length of gain region
c
c  Set offset from line center of first mode, 0 hz
c  Set fractional pumping into level 1, nlpump, 0
c
freq = center
gaincv = 2./pi/width list
lambda = c/freq/Index list
Beinstein = lambda*2/(8*pi*tspont)*gaincv list
hnu = h*freq/c list
xa = 2.*time/zstep/hnu

gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/list
pack/set 1 2 3          # pack all 3 beams
pack/in                 # pack beams
c
c  Set gain length of .89 cm
c  Set pump time of 1e-10
c  Set number of steps over which distance is to be taken
c
    gain/rate/step length time 10 # implement rate eq. gain
pack/out                # unpack beams
set peak1 1 peak        # record peak irradiance
variab/set peak2 2 peak
# write file containing calculations
write/disk temp1.txt/o
# Nonradiative recombination coefficient for decay rate of inversion. [sec^-1]
A = 1e7                list

# Bimolecular recombination coefficient for decay rate of inversion. [cm^2/sec]
B = 1e-10              list

# Auger recombination rate for decay rate of inversion per Eq. (135). [cm^6/sec]
C = 1e-30              list

# Initial population density. [cm^-3]
N = 1e19               list

```

Jump to: [Commands](#), [Theory](#)

```

# Small signal gain coefficient. [cm^-1]
g0 = 2000      list

# Transparency population density of excitations. [cm^-3].
Ntr = 2e18     list

# Numerical adjustment to keep the argument of the logarithm positive. [cm^-3]
Ns = 1e18      list

# Length. [cm]
L = 200e-7

# Ratio of mode bandwidth to spontaneous emission bandwidth, bsp.
Beta = .1

# Spontaneous emission rate [cm^-3/sec]
Rsp = B*N^2     list

# Wavelength/ [cm]
Lambda = 1e-4

# Index of refraction. (Just a guess).
Index = 3.567

# Material wavelength.
xlambda = Lambda/Index

# Frequency. [cm^-1]. "c" is the speed of light -- predefined in GLAD.
nu = c/Lambda   list

# Quantity h*nu. "h" is Plank's constant -- predefined in GLAD.
hnu = h*nu      list

# Pixel-to-pixel spacing, x-direction.
dX = .01

# Pixel-to-pixel spacing, y-direction.
dY = dX

# Time for one pass. [sec]
dT = L*Index/c  list

# Solid angle subtended by array.
Omega = Lambda^2/dX/dY  list

# Spontaneous emission is removed from population inversion and added
# into optical field as a delta-correlated random complex amplitude.
SpontEmmisionLoss = Beta*Omega*hnu*B*N^2  # w^2/s/cm

# Generation rate. (Just a guess.)
Pgen = .1       list # 1.0 w/cm^3

# Generation for
dN = Pgen*dT/hnu/2  list # half of generation

```

Jump to: [Commands](#), [Theory](#)

```

# Add half of generation at beginning of calculation and half at end.
N0 = N+dN                                list

# Alternate variable name.
dpop = dN                                list

# Reset to N, so expression looks like equations in text.
N = N0

# Calculate inverse of Tau. (Might want to check that this is non-zero).
InvTau = A + B*N + C*N^2                list

# Calculate Tau.
Tau = 1./InvTau                          list

# Population density loss rate.
N_Tau = N/Tau                            list

# Loss of population over time dT.
dp = N0*(1.-exp(-dT/Tau))               list

# Correct population by spontaneous emission loss.
N0 = N0 -dp                              list

# Calculate gain as a function of N.
gN = g0*log((N+Ns)/(Ntr+Ns))            list

# Calculate cross section.
sigma = gN/N                             list
# Set initial optical field. Also, will grow from spontaneous emission.
I0 = 1

# Set to names of internal variables, for easy code check.
flux0 = I0                               list

# Calculate maximum possible population density.
Nmax = N0 + (I0*dT)/(hnu*L)             list

# Calculate maximum possible optical intensity.
Imax = I0 + N0*hnu*L/dT                 list

# Calculate an internal variable for code check. Same as Nmax.
coef3 = N0 + I0*dT/L/hnu

# Calculate the coefficient to convert population to intensity.
xa = dT/L/hnu                           list

# Calculate an internal variable for code check. Same as Nmax.
ximax = coef3/xa                        list

# Calculate internal variable for code check.
args = sigma*coef3*L                    list

```

Jump to: [Commands](#), [Theory](#)

```

# Calculate intensity after propagation L, based on Frantz-Nodvik.
I1 = Imax*I0/(I0 + (Imax - I0)*exp(-sigma*Nmax*L))

# Set variable for code check.
flux2 = I1                                list

# Calculate depletion of population by conservation of energy.
N1 = Nmax - I1*dT/hnu/L                    list

# Calculate change in intensity.
dint2 = I1 - I0

# Calculate change in popylation.
dpopw2 = dint2*xa                          list

# Calculate intensity amplification.
arg1 = 1. + dint2/flux0                    list

# Calculate amplitude amplification.
arg1 = sqrt(arg1)                          list

# display half of generation increase of population. (Calcualted above.)
dpop=                                     # half of noise

# Set to an internal variable name.
dpopsum = dpopw2                           list

# Spectral width. (Just a guess. We will work at line center for this
calculation.)
Width = 1.45e13                            # Set line width, hz

# Calculate line width coefficient for internal code check.

gaincv = 2./(pi*Width)                     list
write/disk/off/close

# Set up a 2 x 2 array for convenience in checking the calculation.
array/s 1 2
# Beams                                Use
# ----                                ---
# 1                                    Optical array
# 2                                    Gain array, unpolarized.
nbeam 2 data
units/set 0 dX dY
# Set line center. This calculation is assumed to be at line center,
# so this is not a real issue.
Center = nu                                # Set line center, hz

# Guess at longitudinal mode separation. Not used here. User should
# calculate proper values.
Cavity = 5e11                              # Frequency separation between longitudinal modes

# Set semiconductor cavity parameters.
gain/semi/cavity width=Width center=Center cavity=Cavity offset=0

```

Jump to: [Commands](#), [Theory](#)

```

# Set semiconductor gain parameters.
gain/semi/set ntr=Ntr ns=Ns gain=g0 a=A b=B c=C beta=Beta

set/definitions/on          # turn definitions on or off
                             # A 5.1 feature only implemented for gain/semi in 5.0

# List current settings.
gain/semi/list

clear 2 0                   # Clear gain array
clear 1 Pgen                # Set generation rate
                             # Apply aperturing and other customization
                             # to control generation distribution

gain/semi/pgen 1 2         # Set generation into gain array (beam 2).
field 2                    # List real and imaginary parts of gain array
clear 1 N*hnv              # Set initial population inversion. Could be zero.
gain/semi/nlevel 1 2       # Set inversion into gain array.
field 2                    # List real and imaginary parts of gain array
clear 1 I0                 # Set initial optical field.
wavelength/set 0 Lambda*1e4 Index  # Set wavelength
# Implement gain
pack/set 1 2
pack/in
    gain/semi/step L dT
pack/out
field 1                    # List optical field to show correct gain.
write/disk/off/close

```

Ex69f: Numerical example of rate equation gain

Numerical example of rate equation gain. See GLAD Theory Manual, [Sect. 9.3](#). This example provides a simple calculation for verification showing all steps.

Input: ex69f.inp

```

c## ex69f
c
c Example to illustrate and verify rate equation gain
c
c Beams    Use
c 1        longitudinal mode on line center (frequency)
c 2        gain array
c
array/s 1 4
nbeam 2 4 4 1 data          # set number of beams
units/s 0 1
Index = 3.35 list
Lambda = .84e-4 list
wavelength/set 0 Lambda*1e4 Index  # set wavelength and index
flux1 = 1 list              # starting irradiance
clear 1 flux1               # set initial irradiance
P2 = 1.93e6                 # pump rate in watts per cm^2
clear 2 P2                  # set pump rate in

```

Jump to: [Commands](#), [Theory](#)


```

gain/rate/n2pump 2 2      # watts per cm^2
                           # Modify beam 2 to put
                           # the pump rate in the real
                           # word
                           # inversion will go in the
                           # imaginary word

field 2
c
width = 1.45e13           # Set line width, hz
C Line width:             @width
frequency = c/Lambda      # Set line center, hz
C Center frequency:      @frequency
Lambda = c/frequency
C Wavelength:            @Lambda
tspont = 3e-9             # Set spontaneous emission time, sec
C Spontaneous emission:  @tspont
t20 = 3e-1                # Set decay time for level 2, sec
C t20:                   @t20
t10 = 1e-11              # Set decay time for level 1, sec
C t10:                   @t10
t2 = 1./(1/tspont + 1/t20)
C t2:                    @t2

mode_sep = 5e11           # Set longitudinal mode separation, hz
t = 1.e-10                # Set incremental time
C Incremental time:      @t
length = .89              # Length of gain region
C Length:               @length
c
c Set offset from line center of first mode, 0 hz
c Set fractional pumping into level 1, nlpump, 0
c
gaincv = 2./pi/width
C Gain spectrum factor:  @gaincv
lambda = c/frequency/Index
C Material wavelength:   @lambda
bk = lambda^2/(8*pi*tspont)*gaincv
C Einstein B-coefficient: @bk
hnu = h*frequency
C Energy per excitation: @hnu
xa = 2.*t/length/hnu
C Time-length factor:    @xa

R2_pump = P2/hnu
P1 = 0.
R1 = P1/hnu

C Initial intensity      @flux1

N2_0 = 0.
C Initial upper level    @N2_0
SigmaPump = 0.           # pump saturation term
c Apply pump saturation term
R2 = max(0,R2_pump-SigmaPump*N2_0/t)

```

Jump to: [Commands](#), [Theory](#)

```

C Pump rate, R2: @R2
N1_0 = 0.
C Pumping applied at top

N2 = t2*R2 + (N2_0 - t2*R2)*exp(-t/t2)
C New upper level: @N2

N1 = (R1 + R2*t2/tspont)*(1-exp(-t/t10))*t10
N1 = N1 + 1/tspont*(N2_0-R2*t2)*(exp(-t/t2)-exp(-t/t10))/(1/t10-1/t2)
N1 = N1 + N1_0*exp(-t/t10)
C New lower level: @N1

dpopw0 = N2 - N1
C Population difference: @dpopw0
gain0 = dpopw0*bk
C Small signal gain: @gain0
coef3 = dpopw0 + flux1*2.*t/length/hnu
C Intermediate coefficient: @coef3
imax = coef3/xa
C Maximum possible intensity: @imax
args = bk*coef3*length
C Exponential argument: @args
flux2 = imax*flux1/(flux1 + (imax - flux1)*exp(-args))
C Amplified intensity: @flux2
amplification = flux2/flux1
C Amplification: @amplification
dint2_check = flux2 - flux1
C Intensity increase: @dint2_check

gain/rate/set width frequency tspond t20 t10 mode_sep 0 0

pack/set 1 2 # pack both beams
variable/set peak1 1 peak # record peak irradiance
C Starting peak intensity @peak1
pack/in # pack beams
c
c Set gain length of .89 cm
c Set pump time of 1e-10
c Set number of steps over which distance is to be taken
c
gain/rate/step length t 1 list # implement rate eq. gain
C after gain/rate/step
pack/out # unpack beams
field 2

C calculate population change after calculation of amplification
variable/set peak2 1 peak
C Peak intensity output: @peak2
dint2 = peak2 - peak1
C Peak intensity change: @dint2
dpopsum = dint2*xa
C Change in inversion, dpopsum: @dpopsum
N2 = N2 - dpopsum/2.
C Final N2: @N2 1/cm^3

```

Jump to: [Commands](#), [Theory](#)

```

N1 = N1 + dpopsum/2.
C Final N1:                                @N1 1/cm^3
N2_check = N2*hnu      # scaled N2
N1_check = N1*hnu      # scaled N1
C Scaled N2:                                @N2 j/cm^3
C Scaled N1:                                @N1 j/cm^3
C find scaled population values from numerical model
point/list/ij 2
variab/set N2_glad point/si
variab/set N1_glad point/pr
C -----
C Check flux increase: dint2 = @dint2 (GLAD), dint2_check = @dint2_check
(analytical)
C Check upper state calculation: N2 = @N2_glad (GLAD), N2_check = @N2_check
(analytical)
C Check lower state calculation: N1 = @N1_glad (GLAD), N1_check = @N1_check
(analytical)
C -----

```

Ex69g: Numerical example of single level, three level gain

Numerical example of rate equation gain. See GLAD Theory Manual, [Sect. 9.6](#). This example provides a simple calculation for verification showing all steps.

Input: [ex69g.inp](#)

Ex69h: Rate equations for ruby

Numerical example of rate equation gain for the ruby laser. See GLAD Theory Manual, [Sect. 9.5](#). This example provides a simple calculation for verification showing all steps.

Input: [ex69h.inp](#)

Ex69i: Rate equations for general, three level laser

Numerical example of rate equation gain for the general three laser. See GLAD Theory Manual, [Sect. 9.7](#). This example provides a simple calculation for verification showing all steps.

Input: [ex69i.inp](#)

Ex69j: Steady-state rate equation solution

Numerical example of rate equation gain for the general three laser. See GLAD Theory Manual, [Sect. 9.3](#).

Input: ex69j.inp

Ex69k: Rate equations for single level, three-level laser, multiple steps

This example is similar to the single step example 69g but with multiple steps.

Input: ex69k.inp

Ex70: Udata display

The functions `udata` and `plot/udata` may be used to display the history of GLAD activities. Up to 12 functions may be displayed.

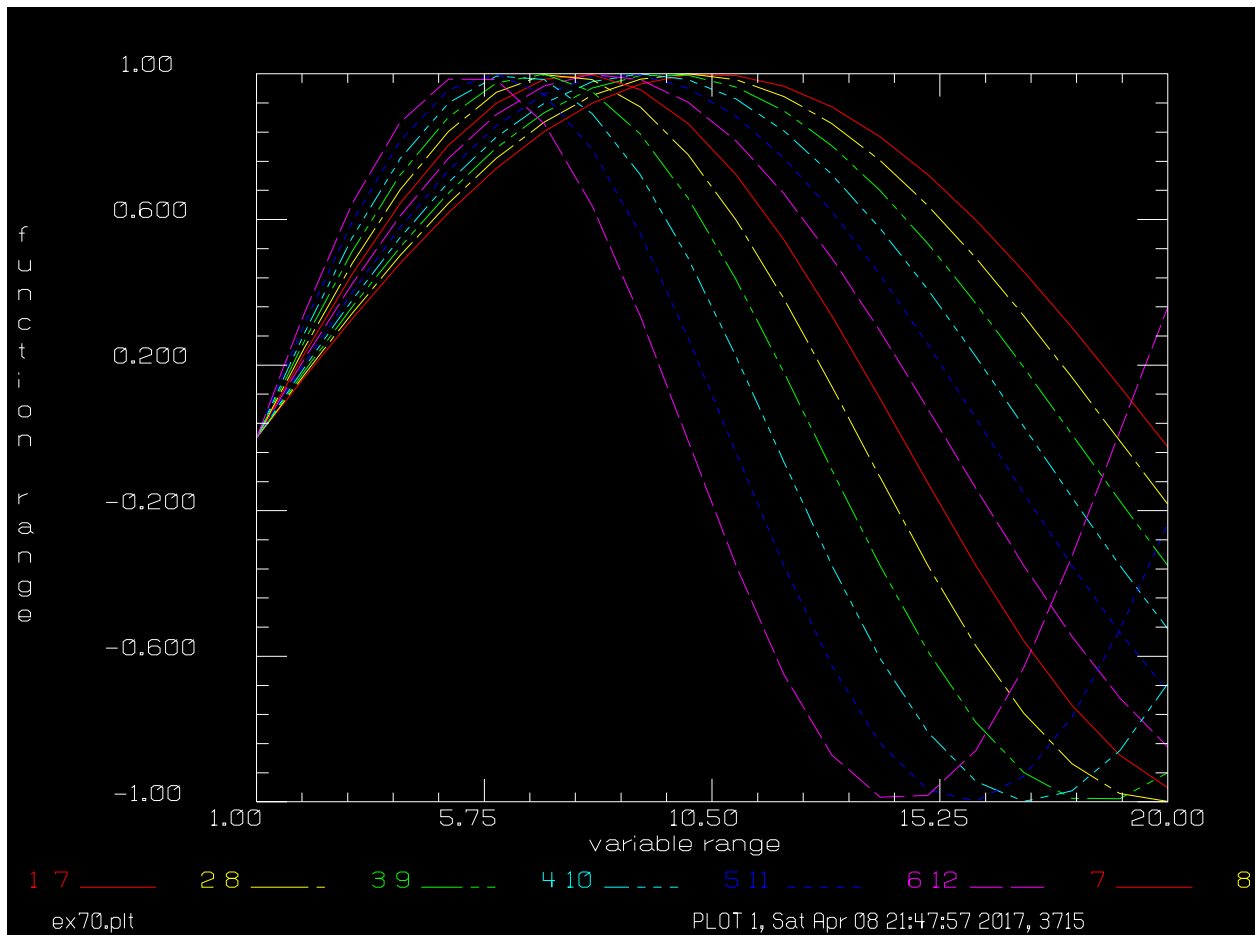


Fig. 70.1. 12 sinewaves of varying period plotted with `udata`.

Input: `ex70.inp`

```
c## ex70
c
c Example 70: Some extended features of UDATA
c
variab/dec/int pass i1
macro/def ex70/o
c
c Calculate 12 sinewaves of varying period.
c
pass = pass + 1
i1 = 0
macro inner/12
c
step = step + 7          # advance phase of sinewaves
```

```
macro/end
macro/def inner/o
c
c  Use register value to calculate variable indices and lvalues
c  for udata/set
c
    i1 = i1 + 1
    y@i1 = sin(step/(44-2*i1))
    if i1 < 10 then
        udata/set pass pass y0@i1=y@i1
    endif
    if i1 >= 10 then
        udata/set pass pass y@i1=y@i1
    endif
macro/end
c
c  Initialize values
c
pass = 0
step = 0
c
c  Run macro 20 times
c
macro ex70/20
c
c  Define sequence of functions to be plotted -- in this case
c  the maximum number of 12 is used.
c
plot/udata/set y01 y02 y03 y04 y05 y06 y07 y08 y09 y10 y11 y12
c
c  Define plot labels
c
plot/udata/xlabel variable range
plot/udata/ylabel function range
c
c  Plot from function definition.
c
plot/watch ex70.plt
plot/udata/sequence min=-1 max=1 dash
end
```

Ex71: Schlieren system

This example exhibits conversion of phase aberrations to intensity modulation by means of an obscuration at the frequency plane of a spatial filter. The phase aberration is a checker board piston error made by using the `lensarray` command.

The obscuration blocks the center lobe and first few rings of the Airy pattern. The intensity pattern in the far field clearly shows the an intensity modulation corresponding to the initial phase modulation. Adapted from a an example written by Shiow-Hwei Hwang.

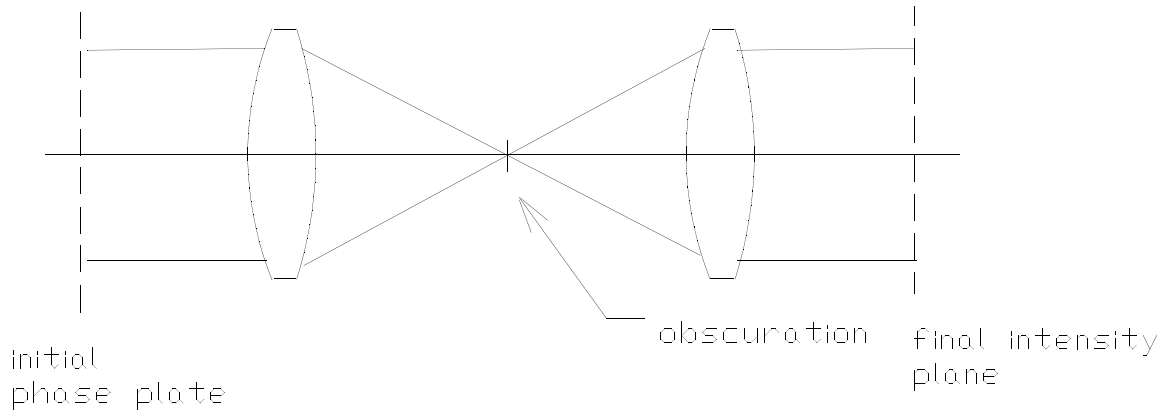


Fig. 71.1. Configuration of spatial filter with obscuration to block central lobe and inner rings of diffraction pattern.

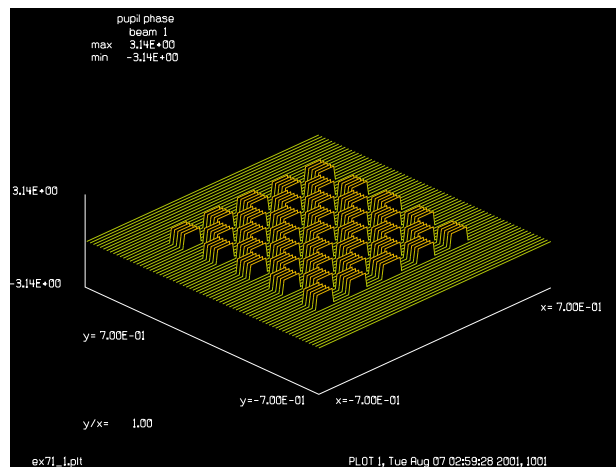


Fig. 71.2. Phase distribution at start of spatial filter.

Input: ex71.inp

```
c## ex71!829835379432407
c
c Example 71: Schlieren system
c
c This example exhibits conversion of phase aberrations to intensity
```

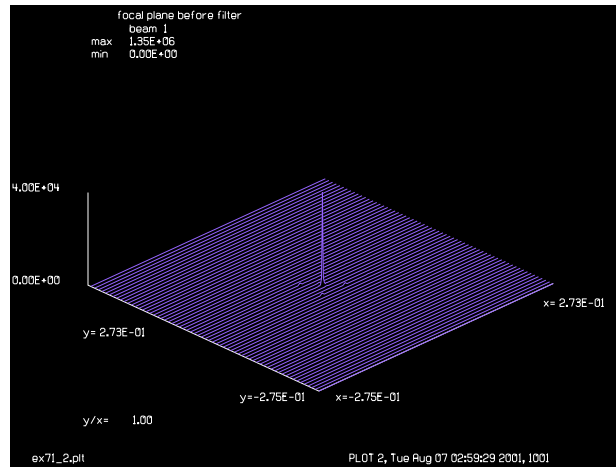


Fig. 71.3. Image of checkerboard phase distribution.

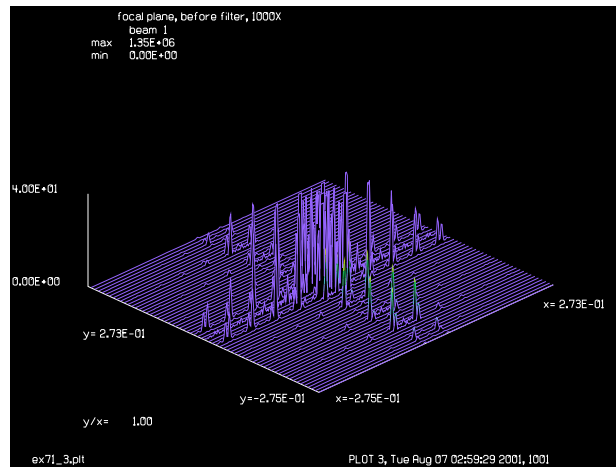


Fig. 71.4. Image of checkerboard phase distribution, 1000X.

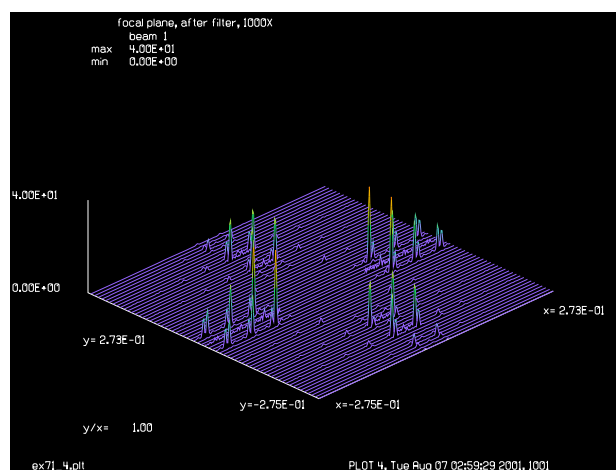


Fig. 71.5. Image of checkerboard phase distribution after obscuration, 1000X.

c modulation by means of an obscuration at the frequency plane of
 c a spatial filter. The phase aberration is a checker board piston

Jump to: [Commands](#), [Theory](#)

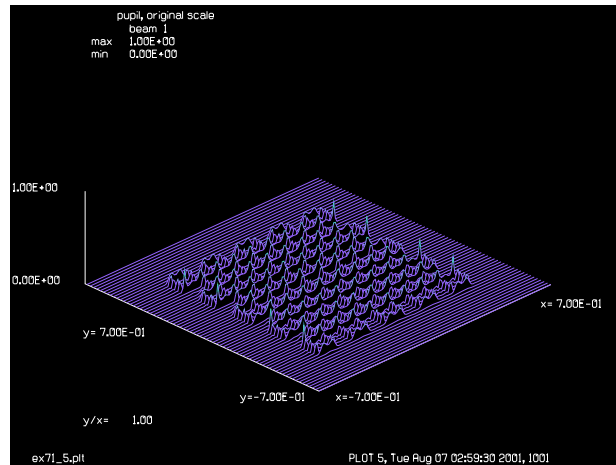


Fig. 71.6. Intensity distribution after Schlieren system showing intensity distribution similar to initial phase modulation.

```

c error made by using the LENSARRAY command.
c
c The obscuration blocks the center lobe and first few rings of the Airy
c pattern. The intensity pattern in the far field clearly shows the
c an intensity modulation corresponding to the initial phase modulation
c
c Courtesy of Shioh-Hwei Hwang
c
variab/dec/int row col element switch size start
macro/def inner/o
    col = col + 1                # increment column number
    element = element + 1        # increment element number
    switch = -1 ^ element        # -1**element
c process every other element
    if switch < 0 lensa/rec/ele 1 row col width width
macro/end
macro/def outer/o
    row = row - 1                # decrement row number
    col = -start                 # initialize column number
    macro/run inner/size
macro/end
echo/on
array/set 1 256
units/s 1 .01
wavelength/set 1 .55
size = 9                        # set array size to 9 x 9, use odd value
piston = .15                    # set piston value
width = .1                      # rectangle width, 1 mm
start = size/2 + 1              # starting row or column number
row = start                     # initialize row number
col = 0                         # column number
element = 0                     # element counter
lensa/rec/set piston=piston     # set piston value
macro/run outer/size
title pupil phase

```

Jump to: [Commands](#), [Theory](#)

```
plot/watch ex71_1.plt
plot/l/ph 1 min=-3.14 max=3.14 xrad=.7 ns=64
dist 100
lens 1 100
dist 100
title focal plane before filter
plot/watch ex71_2.plt
plot/l 1 min=0 max=4e4 ns=64 xrad=.3
title focal plane, before filter, 1000X
plot/watch ex71_3.plt
plot/l 1 min=0 max=40 ns=64 xrad=.3
obs 1 .1
title focal plane, after filter, 1000X
plot/watch ex71_4.plt
plot/l 1 min=0 max=40 ns=64 xrad=.3
dist 100
lens 1 100
dist 100
title pupil, original scale
plot/watch ex71_5.plt
plot/l 1 min=0 max=1 ns=64 xrad=.7
end
```

Ex72: Test ABCD equivalent systems

This example illustrates the use of optically equivalent systems. In the first case a propagation followed by a lens and a second propagation to the focus is shown to be equivalent to a magnification change and an equivalent lens or an ABCD operation. The latter two methods have only one diffraction step.

Input: ex72.inp

```
c## ex72
c
c Example 72: Test ABCD equivalents to system.
c
echo/on
c
c System 1: original
c
array/s 1 64
zreff/se 1 0
units/s 1 .01
clap/c/c 1 .2
dist 10
lens 1 20
dist 20
geodata 1
title original system
plot/watch ex72_1.plt
plot/x/i
c
c Equivalent to System 1
c
array/s 1 64
zreff/se 1 0
units/s 1 .01
clap/c/c 1 .2
lens 1 10
magnify 1 2
dist 40
geodata 1
title equivalent system
plot/watch ex72_2.plt
plot/x/i
c
c ABCD for System 1
c
array/s 1 64
zreff/se 1 0
units/s 1 .01
clap/c/c 1 .2
abcd/set 1
abcd/operator 1 ax=0 bx=20 cx=-.05 dx=.5
geodata 1
title equivalent system
plot/watch ex72_3.plt
```

```
plot/x/i
c
c  System 2: original
c
array/s 1 64
zreff/se 1 0
units/s 1 .01
clap/c/c 1 .2
dist 20
lens 1 20
dist 20
geodata 1
title original system
plot/watch ex72_4.plt
plot/x/i
c
c  Equivalent to System 2
c
array/s 1 64
zreff/se 1 0
units/s 1 .01
clap/c/c 1 .2
lens 1 20
dist 20
lens 1 20
geodata 1
title equivalent system
plot/watch ex72_5.plt
plot/x/i
c
c  ABCD for System 2
c
array/s 1 64
zreff/se 1 0
units/s 1 .01
clap/c/c 1 .2
abcd/set 1
abcd/operator 1 ax=0. bx=20 cx=-.05 dx=0.
geodata 1
title equivalent system
plot/watch ex72_6.plt
plot/x/i
end
```

Ex73: Tests of dynamic memory**Input: ex73.inp**

```

c## ex73!963961812736928
c
c Example 73: Tests of dynamic memory
c
c This example tests various operations which are sensitive to the
c size of the memory. There are 3 basic modes of operation
c
c blocking          memory is not sufficient to hold the whole array
c                   being worked on. Blocks of memory are
c                   transferred. An out-of-memory transpose is used
c                   during diffracton.
c                   Slowest mode because of the disk IO required.
c
c swapping          memory is sufficient for a whole array but not
c                   sufficient to hold all arrays. The arrays are
c                   swapped in and out of memory as required.
c                   This mode is fine if only beam is being used,
c                   but requires disk IO if several beams are being
c                   used. The transpose is done in-memory which
c                   is much more efficient. Copies and other multiple
c                   beam operations require blocking until the operation
c                   is complete.
c
c sharing           memory is sufficient to hold all arrays as defined
c                   by Nbeam. This is the most efficient mode.
c
c The initial memory size is set to 2**16 = .5 megabytes on most
c computers. You can increase or decrease the memory during program
c execution. It is generally best to choose a size appropriate to
c your problem and the resources of your computer and leave it alone
c to avoid inefficiencies associated with the transitions.
c
echo/on
mem/set 16          # set to 2**16 = .5 megabytes
array/s 1 128       # define an array of 128 x 128
energy              # test energy, should be 16384
nbeam 4             # define 4 beams of 128 x 128
clear 2 1           # initialize arrays
clear 3 1
clear 4 1
debug rdwr          # turn on debug mode to see disk IO
                    # check energy, no disk IO is used
energy              # because beams are sharing memory
pause
memory/content      # display current memory contents
nbeam 5             # add one more beam so disk swapping is
                    # required
clear 5 1
energy              # test energy, note disk IO

```

```

pause
nbeam 1                                # reduce to 1 beam, GLAD releases other
                                        # beams
energy                                  # note no more disk IO after the first
                                        # call which reads the array back from disk
energy                                  # no disk IO here
pause
memory/set 12                           # reduce memory so the 128 x 128 array
                                        # is processed in 4 blocks
energy                                  # note disk IO even for one beam
pause
c
c Now, we will examine the transpose which is the most difficult
c operation because the information is accessed in highly non-
c sequential fashion.
c
clap/c/c 1 10 -20                       # make an eccentric aperture
debug/delete rdwr                       #      turn off debug to see display better
intmap 1                                # display aperture
pause
debug rdwr                             #      turn on debug mode to see disk IO
transpose 1                             # note disk IO
debug/delete rdwr                       #      turn off debug to see display better
intmap 1                                # display transpose
pause
memory/set 16                           # increase memory to original value
debug rdwr                             #      turn on debug mode to see disk IO
transpose 1                             # note no more disk IO in transpose
debug/delete rdwr                       #      turn off debug to see display better
intmap 1                                # should be back to original shape
pause
debug rdwr                             #      turn on debug mode to see disk IO
c
c Now, we will try the transpose operation on rectangular arrays
c under different memory conditions.
c
array/s 1 128 256                       # set to 128 columns by 256 rows
clap/c/c 1 10 -20                       # make the same aperture
debug/delete rdwr                       #      turn off debug to see display better
intmap 1                                # display intensity map
pause
debug rdwr                             #      turn on debug mode to see disk IO
transpose 1                             # transpose array, no disk IO
memory/set 12                           # array now requires 8 blocks
transpose 1                             # note disk IO now required
debug/delete rdwr                       #      turn off debug to see display better
intmap 1                                #
pause
debug rdwr                             #      turn on debug mode to see disk IO
c
c Now, we will exercise the remap function which is used in the
c diffraction propagation. It shifts the beam by a half the array width
c in the x- and y-directions
c

```

Jump to: [Commands](#), [Theory](#)

```

array/s 1 64          # square array of 64 x 64, fits in memory
clap/c/c 1 20         # centered aperture of radius 10
intmap 1              # display
pause
remap 1               # should see energy in corners
intmap 1              # display
pause
remap 1               # should be restored
intmap 1              # display
pause
c
c  Now, test scratch array and reblocking during copying
c
memory/set 14         # set memory to just hold 128 x 128
array/s 1 128         # reset array
energy                # note no disk IO after restoring state
pause
copy 1 -1             # copy to scratch array
                     # note reblocking to 2 blocks for each beam
                     # so both beams are in memory
                     # simultaneously
clear 1 0             # clear Beam 1 to 0
energy                # display zero energy
copy -1 1             # copy from scratch array
energy                # energy is recovered
pause
c
c  When reducing the array size, GLAD may be left in swapping or blocking
c  modes.  Run MEMORY/RESET to see if there is enough memory to use
c  memory sharing.  MEMORY/RESET is run automatically when the memory
c  size is changed by MEMORY/SET or when NBEAM is run.
c
memory/set 12         # set memory to just hold 64 x 64
                     # beam is still in blocking mode because
                     # of the current array size of 128 x 128
array/s 1 64          # reduced size
energy                # note disk IO still being used
pause
memory/reset          # try to run in memory sharing mode
energy                # no more disk IO
pause
c
c  Now, we will test the polarization properties
c
array/s 1 64 64 1     # set array with polarization so tht
                     # each polarization array fits in memory
                     # but full beam is twice the memory
                     # allocation
clap/c/c 1 10 -10     # decentered aperture
c
c  set equal energy in each polarization state
c
jones/set ar=.54772256 cr=.83666003 dr=0
jones/mult 1

```

Jump to: [Commands](#), [Theory](#)

```

energy                                # note blocking required to calculate energy
                                     # energy
debug/delete rdwr                     #      turn off debug to see display better
intmap 1                             # display beam
pause
debug rdwr                           #      turn on debug mode to see disk IO
transpose 1                          # transpose array, note blocking
                                     # is not needed during transpose
debug/delete rdwr                     #      turn off debug to see display better
intmap 1                             # display beam
pause
debug rdwr                           #      turn on debug mode to see disk IO
mem/set 11                           # reduce memory to force blocking
                                     # during matrix transpose
transpose 1                          # note increased IO
debug/delete rdwr                     #      turn off debug to see display better
intmap 1                             # restore original distribution
pause
array/s 1 64 128 1                   # polarized rectangular array, blocking
clap/c/c 1 10 -10
c
c  set equal energy in each polarization state
c
jones/set ar=.54772256 cr=.83666003 dr=0
jones/mult 1
intmap 1                             # display polarized, rectangular array
transpose 1
intmap 1                             # display transposed rectangular array
pause
c
c  test copy routines with memory set to hold only two of three arrays
c
array/s 1 8                          # set array 1 to 8 x 8
mem/set 7
nbeam 2                             # create two beams of 8 x 8
mem/cont                             # display memory, beams 1 and 2 in memory
nbeam 3
mem/cont                             # display memory, beams 3 and 2 in memory
pause
clear 2 0
intmap 2
clear 3 0
mem/cont
intmap 3
mem/cont                             # display memory, beams 3 and 2 in memory
pause
point/set 2 3 5 1                    # put a point at (3,5), for beam 2
intmap 2
copy/con 2 3                         # copy into beam 3
intmap 3
clear 1 0
intmap 1
point/set 1 5 5 1                    # put a point at (5,5), for beam 1
intmap 1

```

Jump to: [Commands](#), [Theory](#)


```
mem/cont                # display memory, beams 1 and 3 in memory
add/coh/c 3 1           # coherently add beam 1 to beam 3
mem/cont                # display memory, beams 1 and 3
intmap 3
intmap 1
point/set 3 7 5 1       # put a point at (7,5), for beam 3
intmap 3
c
c  should see three points at (-2,0), (0,0), (2,0)
c
pause
end
```


Ex74: More checks of dynamic memory allocation**Input: ex74.inp**

```

c## ex74
c
c Example 74: More checks of dynamic memory allocation
c
echo/on
mem/set 12                # set memory to 4096 complex words to exaggerate
                           # the disk IO
                           # GLAD is usually set with a default of 65536 for
                           # IBM/DOS and 262144 for worstations so the current
                           # example is quite extreme
                           # 4096 will hold a maximum array of 64 x 64 so
                           # out-of-memory transposes for rectangular polarized
                           # arrays are tested
                           # turn on disk IO debug mode to see reads and writes

debug rdwr                #
nbeam 3
array/s 1 64 128          # rectangular, unpolarized arrays
array/s 2 128 64
array/s 3 64 64
mem/cont
pause                     # display memory contents
clap/c/c 0 10 -10
debug/delete rdwr         # turn off debug mode for displays
intmap 1
pause                     # beam 1 start
debug rdwr
transpose 1
debug/delete rdwr
intmap 1
pause                     # beam 1 after transpose
intmap 2
pause                     # beam 2 start
debug rdwr
transpose 2
debug/delete rdwr
intmap 2
pause                     # beam 2 after transpose
intmap 3
pause                     # beam 3 start
debug rdwr
transpose 3
debug/delete rdwr
intmap 3
pause                     # beam 3 after transpose
debug rdwr
dist 10000
debug/delete rdwr
intmap 1
pause                     # beam 1 after propagation
intmap 2

```

```
pause                                # beam 2 after propagation
intmap 3
pause                                # beam 3 after propagation
array/s 1 64 128 1                   # make beam 1 polarized
array/s 2 128 64 1                   # make beam 2 polarized
array/s 3 64 64                       # reset beam 3 unpolarized
mem/cont
pause                                # display memory contents
clap/c/c 0 10 -10
jones/set ar=0 cr=1 dr=0              # put array in mixed polarization step to
                                      # be sure both polarization sheets are modified
                                      # correctly
jones/mult 1                          # implement polarization
jones/mult 2
intmap 1
pause                                # beam 1, polarized start
intmap 2
pause                                # beam 2, polarized start
intmap 3
pause                                # beam 3, unpolarized start
dist 10000
intmap 1
pause                                # beam 1, polarized, after propagation
intmap 2
pause                                # beam 2, polarized, after propagation
intmap 3
pause                                # beam 3, unpolarized, after propagation
end
```

Ex75: Axicons

Table. 75.1. Table of Ex75examples

Ex75a: Afocal axicon left out and left back.	2
Ex75b: Afocal axicon with shift on return, both left.	4
Ex75c: Afocal axicon left and right with shifts.	6
Ex75d: Afocal axicon with large positive shift.	9
Ex75e: Stable resonator with focal axicons.	11
Ex75f: Unstable resonator with focal axicons.	13
Ex75g: Form annular beam from radial beam, reflaxicon.	15
Ex75h: Form annular beam from radial beam, waxicon.	17
Ex75i: Reflaxicon-waxicon pair with intermediate ring focus.	22
Ex75j: Effect of decenter and tilt on beam in radial mode.	26
Ex75k: Waxicon resonator with inverting optics (simple treatment).	27
Ex75l: Waxicon resonator with inverting optics (exact mirror treatment).	35

This example shows several applications of axicon mirrors. Axicon mirrors may be characterized by their effect on the optical axis. An axicon can change a conventional beam with a simple line axis (called an axial beam) into a radially expanding cylindrical wave (called a radial beam) with an optical axis in the form of a plane perpendicular to the original axis. A second axicon may convert the radial beam into an annular beam with the optical axis taking the form of a cylinder. The annular beam propagates parallel to the original axis. In special cases the radius of the axis of the annular beam may be zero, corresponding to an axial beam.

Several examples of axicons are presented.

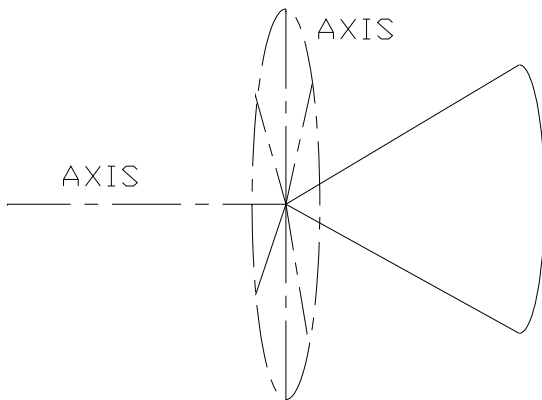


Fig. 75.1. An axicon mirror splits the optical axis of an axial beam into a plane perpendicular to the axis in forming a radial beam.

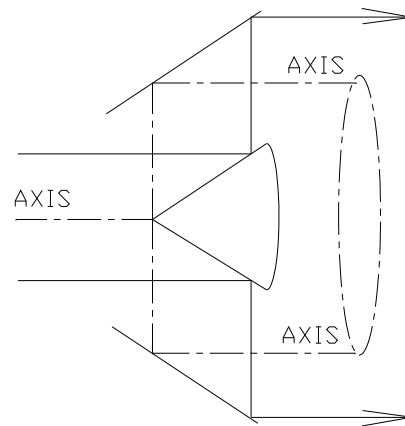


Fig. 75.2. A second axicon can generate an annular beam from a radial beam with the axis taking the form of a cylinder.

Ex75a: Afocal axicon left out and left back

Example 75.1 illustrates expansion of a beam from axial mode to radial mode by an axicon. In this case the axicon is a left axicon. The radial beam is reflected by a cylindrical mirror and reformed into an axial beam by hitting the same axicon a second time. Fig. 75a.1 illustrates schematically how a flat top beam is transformed into a radial beam by an axicon. The cylindrical wavefront is represented on a rectangular array with the x-direction corresponding to the radius and the y-direction corresponding to the azimuthal direction. The optical axis, which is now expanded into a plane, is represented in the computer array as a vertical line in the center of the array. Fig. 75a.2. shows how the radius of the exiting radial beam is specified.

GLAD propagates, in axial mode, the specified radial distance and then does a geometric transformation to radial mode. Fig. 75a.3 shows the intensity in the radial beam. The radial beam goes to zero at the optical axis. Fig 75a.4 shows the polarization in radial mode. The polarization which was linear in axial mode, has now been rotated in the azimuthal direction. Fig. 75a.5 shows that the flat top function is accurately recreated after the radial mode is reflected by a cylindrical mirror and the same axicon is used to retransform the beam to axial mode. There are interpolation errors and the center point, where the cone tip lies, is made zero to avoid a singularity. The linear polarization is also restored, as shown in Fig. 75a.6.

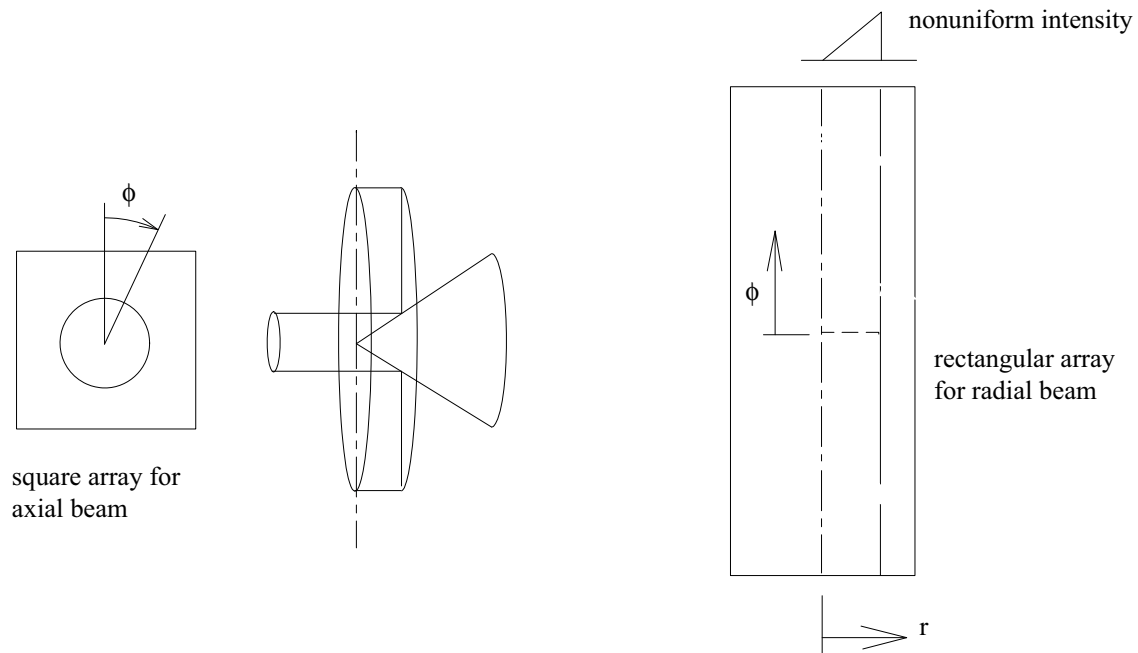


Fig. 75a.1. Schematic of transformation of a flat top axial beam represented on a square array, into a radial beam with nonuniform intensity on a rectangular array. In radial mode, the x-direction represents the radius and the y-direction represents the azimuthal direction. The optical axis is a vertical line in the center of the rectangular array. By using no guardband in the vertical direction, cylindrical propagation is treated correctly by FFT techniques.

Input: ex75a.inp

c## ex75a!026357993764086

Jump to: [Commands](#), [Theory](#)

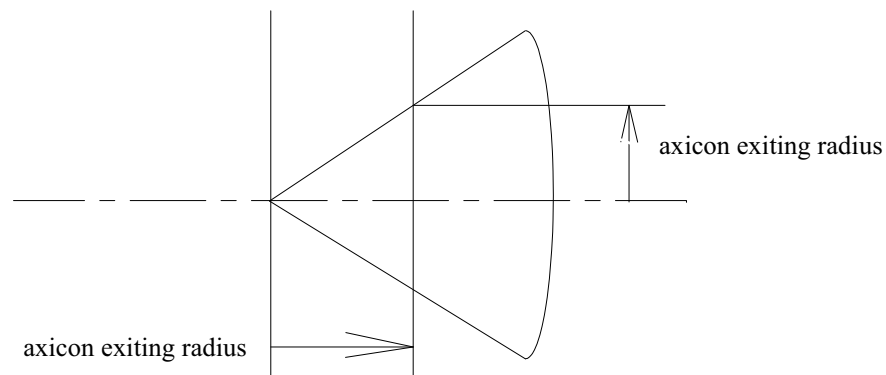


Fig. 75a.2. A radius is specified for the exiting radius of the radial beam. GLAD propagates the distance past the cone tip and then does a geometric transformation. to the radial mode.

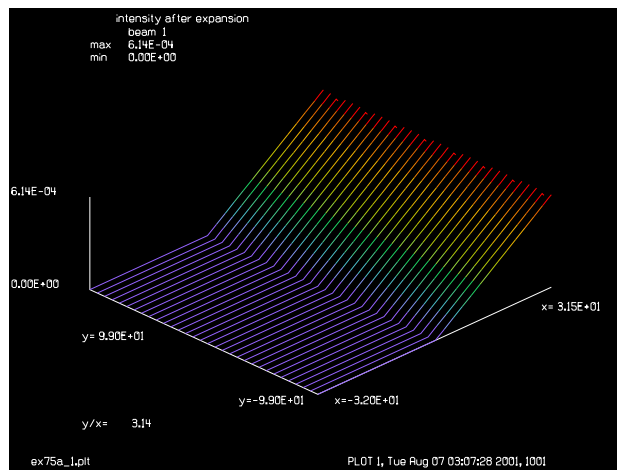


Fig. 75a.3. Intensity distribution in radial beam showing the nonlinear effect on intensity. The radius is in the x-direction, the azimuth in y.

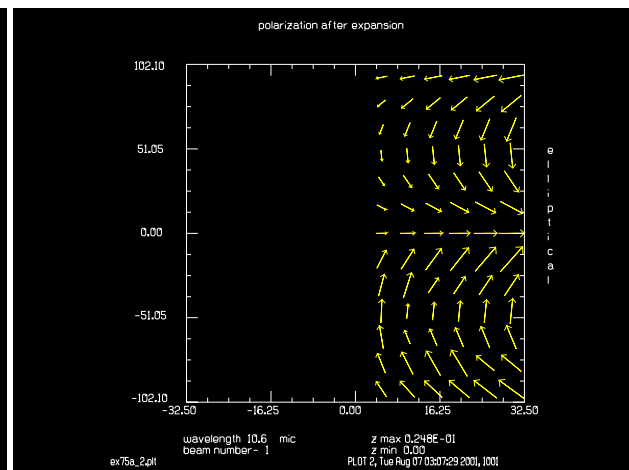


Fig. 75a.4. Polarization in the radial beam. Note that the polarization rotates in the azimuthal direction.

```

c
c Example 75a: afocal axicon left out and left back
c
echo/on
array/s 1 128 128 1          # 128 x 128 array selected
units/set 1 .5 .5
clap/c/c 1 32.              # top hat function selected
obs 1 .001                  # block center point
energy/norm 1                # normalize energy to 1
set/density 12 12
c
c axicon/radial transforms the beam from axial to radial mode
c this will result in an expanding cylindrical wave
c propagate out to a distance of 32 from the axis
c
axicon/radial/left/afocal 1 32.

```

Jump to: [Commands](#), [Theory](#)

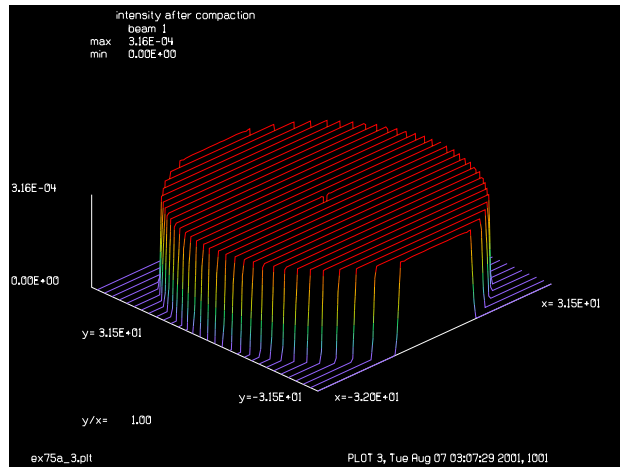


Fig. 75a.5. Intensity after return to axial mode by the same axicon. The original distribution is reproduced except for a hole in the center.

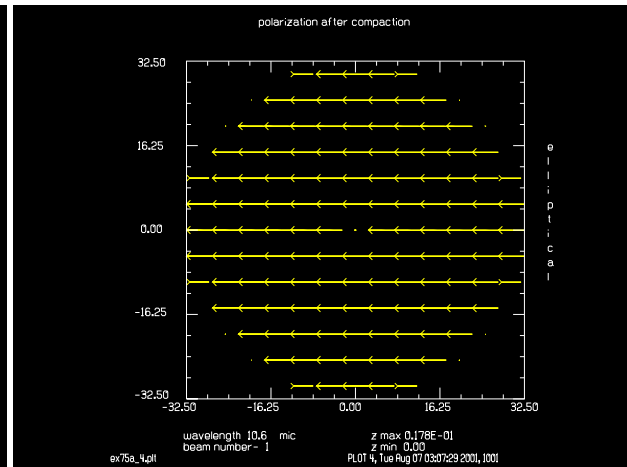


Fig. 75a.6. The polarization returns to the original linear polarization after return from the axicon.

```

title intensity after expansion
plot/watch ex75a_1.plt
plot/l 1                                # note intensity profile in radial
                                         # mode

title polarization after expansion
plot/watch ex75a_2.plt
plot/ell 1                              # note polarization rotation in
                                         # radial mode

mirror/flat 1 xonly                     # mirror reflects radial beam
                                         # back toward the axis
                                         # "xonly" makes beam work only in
                                         # the x-direction, equivalent to the
                                         # radial direction, saves
                                         # defining a toric mirror

c
c Now recompact the beam
c
axicon/axial/left 1 32.
title intensity after compaction
plot/watch ex75a_3.plt
plot/l 1                                # plot intensity after recompactation
title polarization after compaction
plot/watch ex75a_4.plt
plot/ell 1                              # plot polarization after recompactation
energy

```

Ex75b: Afocal axicon with shift on return, both left

This example is similar to Ex75a except the axicon is shifted when the beam hits it the second time. The axicon is shifted to $z = -1$ for the return path, causing the axial beam to have a donut shape.

Jump to: [Commands](#), [Theory](#)

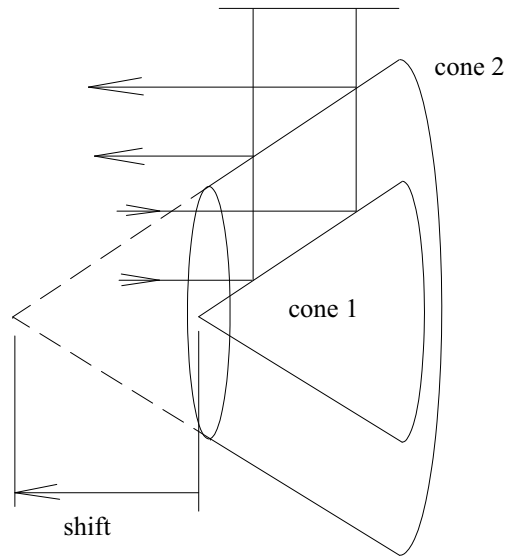


Fig. 75b.1. The return axicon is shifted to the left causing a donut shaped beam to be formed.

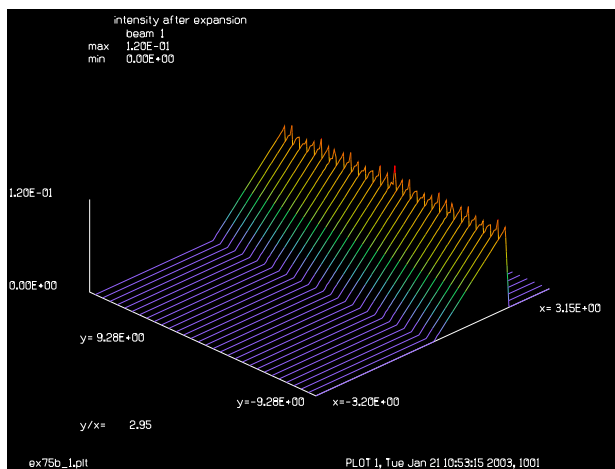


Fig. 75b.2. Intensity in the radial beam is the same as Ex75a. Note small diffraction effects.

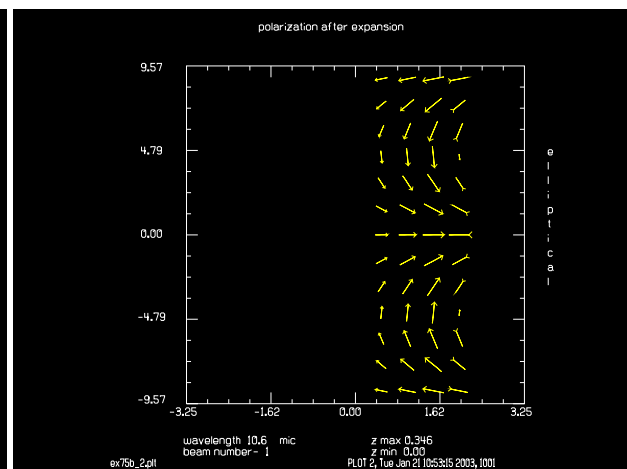


Fig. 75b.3. Polarization in the radial beam is the same as Ex75a.

Input: ex75b.inp

```
c## ex75b!158220266229740
c
c Example 75b: afocal axicon with shift on return, both left
c
echo/on
array/s 1 128 128 1
units/set 1 .05 .05
clap/c/c 1 2
obs 1 .001
```

Jump to: [Commands](#), [Theory](#)

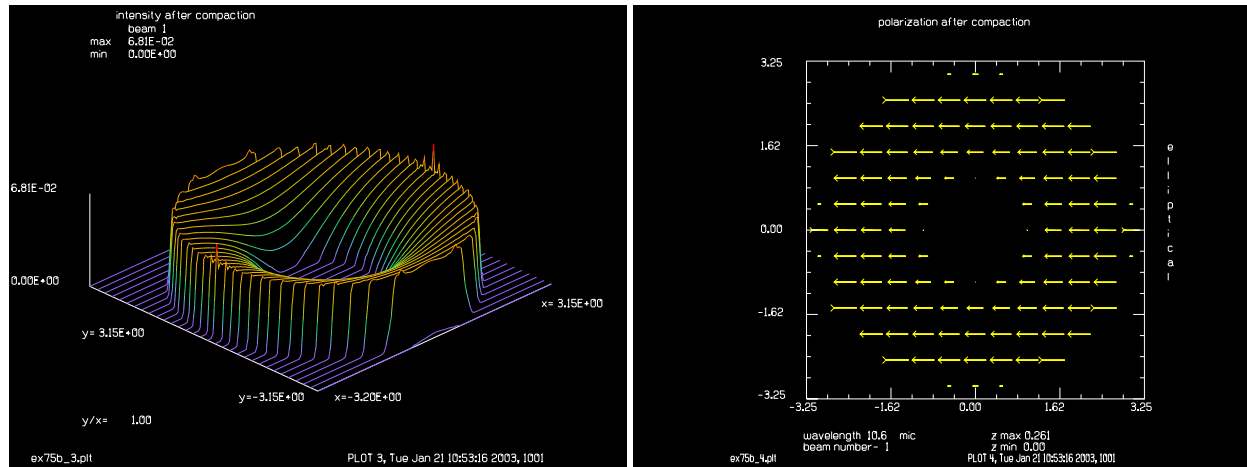


Fig. 75b.4. Donut shaped beam formed by the shifted axicon. Note the small diffraction effects.

Fig. 75b.5. Polarization is returned to linear form.

```
energy/norm 1
set/density 12 12
axicon/radial/left/afocal 1 3      # expand to radial mode with no
                                     # shift
title intensity after expansion
plot/watch ex75b_1.plt
plot/l 1
title polarization after expansion
plot/watch ex75b_2.plt
plot/ell 1
mirror/flat 1 xonly                # mirror, only works in x-direction
axicon/axial/left/afocal 1 3 -1    # recompack with axicon shift of -1
title intensity after compaction
plot/watch ex75b_3.plt
plot/l 1                          # not intensity is not uniform
title polarization after compaction
plot/watch ex75b_4.plt
plot/ell 1                        # polarization is well behaved
energy
```

Ex75c: Afocal axicon left and right with shifts

The radial beam is returned to axial mode by a right axicon. Both the intensity and polarization are disrupted by this configuration.

Input: ex75c.inp

```
c## ex75c
c
c Example 75c: afocal axicon left and right with shifts
c
echo/on
array/s 1 128 128 1
units/set 1 .05 .05
```

Jump to: [Commands](#), [Theory](#)

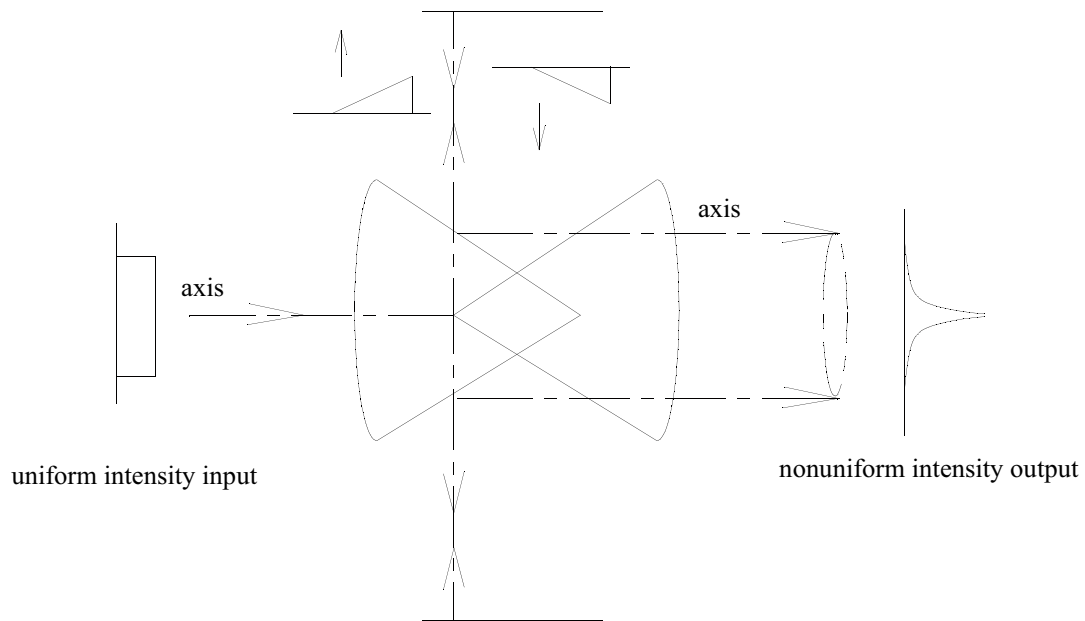


Fig. 75c.1. In this example, a left axicon expands the beam into radial mode and the beam is returned to axial form by a right cone, leading to nonuniform intensity and disrupted polarization.

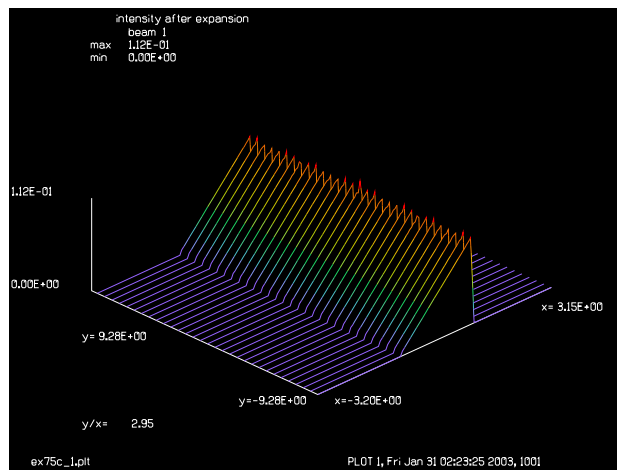


Fig. 75c.2. Intensity in the radial beam is the same as Ex75a.

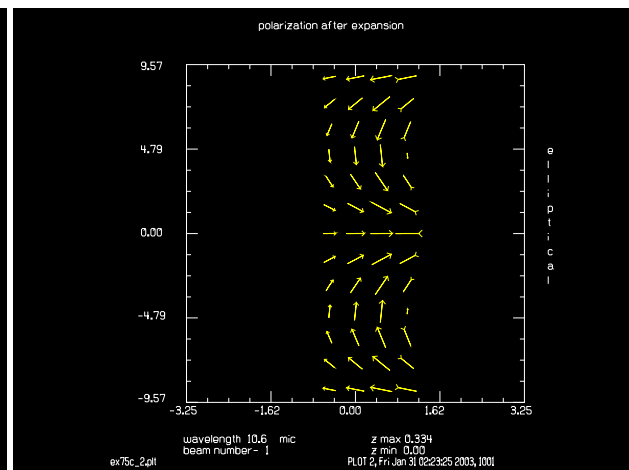


Fig. 75c.3. Polarization in the radial beam is the same as Ex75a.

```
clap/c/c 1 2
obs 1 .001
energy/norm 1
set/density 12 12
axicon/radial/left/afocal 1 3 -1 # left axicon with -1 shift
title intensity after expansion
plot/watch ex75c_1.plt
plot/1 1
title polarization after expansion
plot/watch ex75c_2.plt
```

Jump to: [Commands](#), [Theory](#)

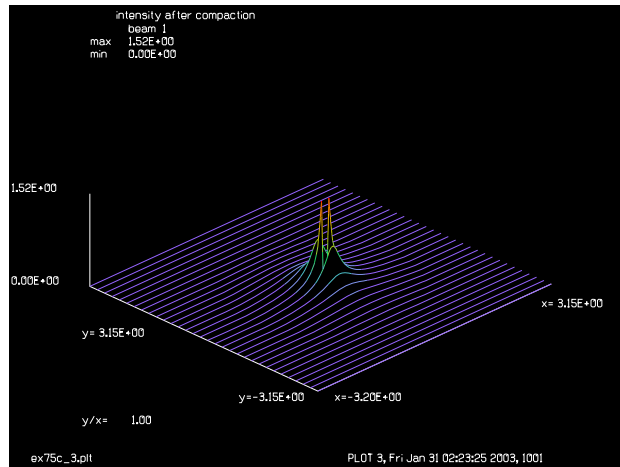


Fig. 75c.4. The intensity is strongly peaked in the center because of the use of the right cone rather than the same left cone.

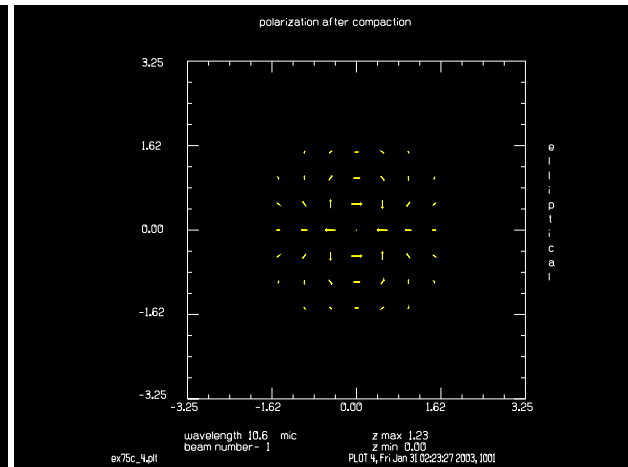


Fig. 75c.5. The polarization is strongly disrupted by the use of a right axicon in the return path.

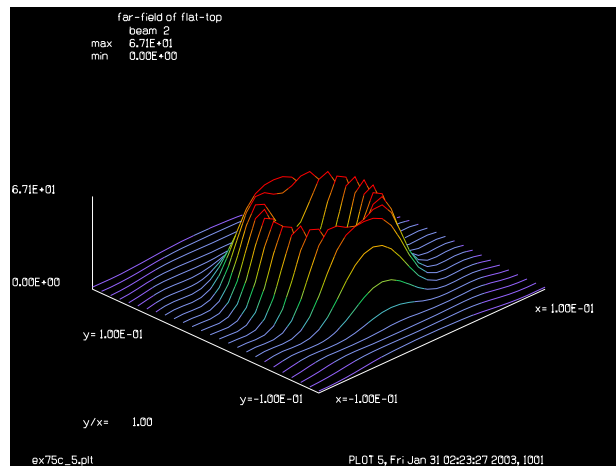


Fig. 75c.6. Expanded view of far-field image showing ring image due to polarization aberrations.

```

plot/ell 1
mirror/flat 1 xonly
axicon/axial/right/afocal 1 3 1    # right axicon with +1 shift
title intensity after compaction
plot/watch ex75c_3.plt
plot/l 1                            # sever disruption of intensity
fitgeo 1
title polarization after compaction
plot/watch ex75c_4.plt
plot/ell 1                          # note polarization rotation
energy
nbeam 2 256 256 1                  # create larger beam for calculating
                                     # far field

clear 2 0
copy/con 1 2                       # copy beam 1 into larger beam
lens 2 100                         # 100 cm focal length lens

```

Jump to: [Commands](#), [Theory](#)

```

dist 100 2                                # propagate beam 2 only
title far-field of flat-top
plot/watch ex75c_5.plt
plot/1 2 xrad=.1                          # far field intensity
                                           # note ring structure

```

Ex75d: Afocal axicon with large positive shift

Two left axicons are used but the return axicon is given a strong positive shift along the axis. The intensity is disrupted but the polarization returns to linear form.

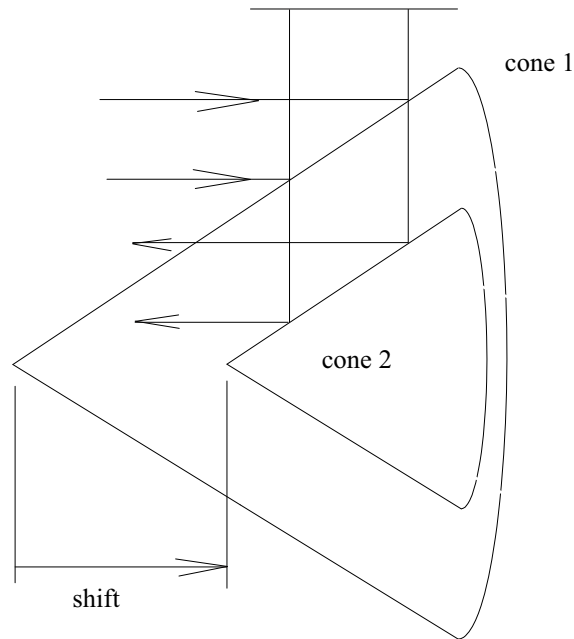


Fig. 75d.1. In this example, a left axicon expands the beam into radial mode and the beam is returned to axial form by a right cone, leading to nonuniform intensity and disrupted polarization.

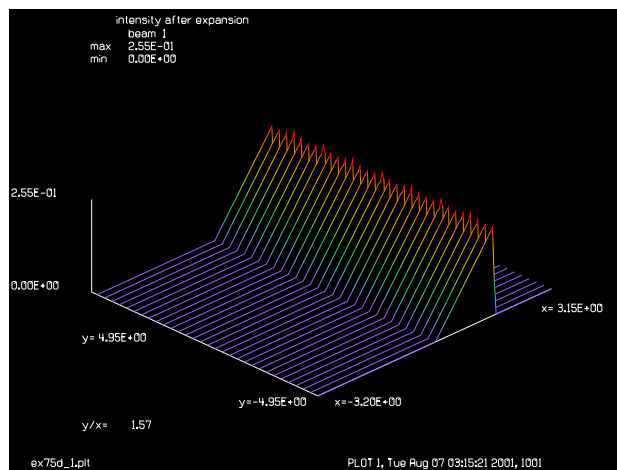


Fig. 75d.2. Intensity in radial beam.

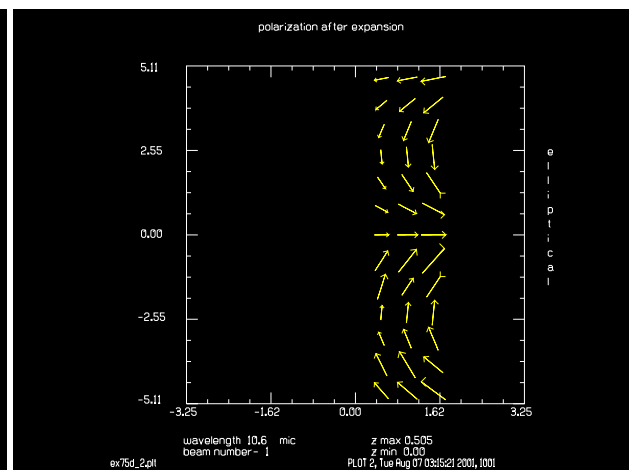


Fig. 75d.3. Polarization in radial mode.

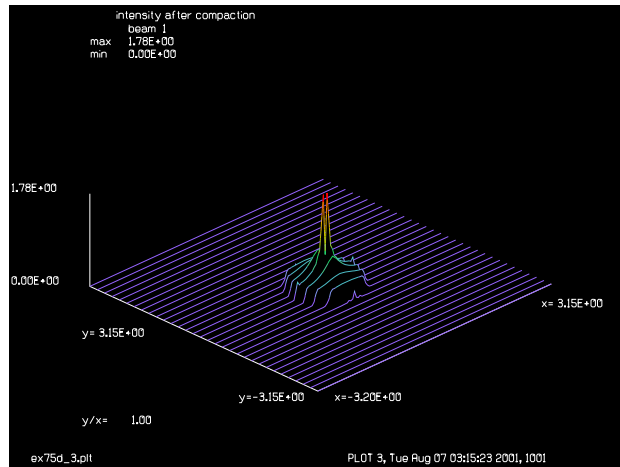


Fig. 75d.4. Intensity after second left axicon. The shift to the right has resulted in nonuniform intensity.

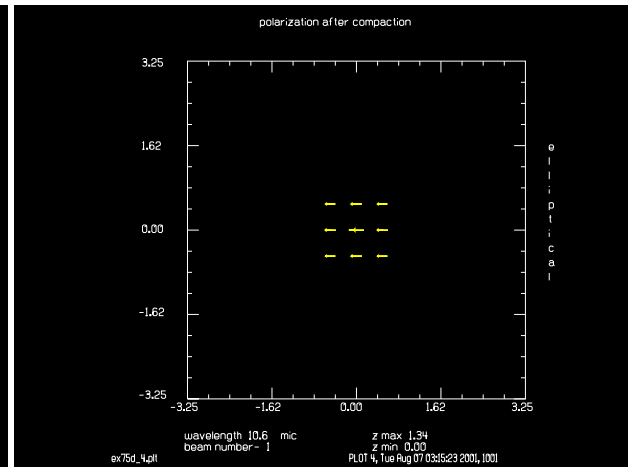


Fig. 75d.5. The polarization is well behaved, unlike the case of left and right axicon mirrors.

Input: ex75d.inp

```

c## ex75d
c
c Example 75d: afocal axicon with large positive shift
c
echo/on
radius = 1.6                                # axicon exiting radius
array/s 1 128 128 1
units/set 1 .05 .05
clap/c/c 1 radius
obs 1 .001
energy/norm 1
set/density 12 12
axicon/radial/left/afocal 1 radius          # form radial mode
geodata
mirror/xcyl 1 xonly                         # reflect radial beam back to axis
title intensity after expansion
plot/watch ex75d_1.plt
plot/l 1                                    # plot radial beam intensity
title polarization after expansion
plot/watch ex75d_2.plt
plot/ell 1                                 # polarization in radial beam
axicon/axial/left/afocal 1 radius .8        # left axicon with positive shift
title intensity after compaction
plot/watch ex75d_3.plt
plot/l 1                                    # recompacted beam
title polarization after compaction
plot/watch ex75d_4.plt
plot/ell 1
energy

```

Ex75e: Stable resonator with focal axicons

This is an example of a stable resonator with an axicon. One of the resonator modes is a flat mirror in the axial leg and the other is a toroidal mirror in the radial leg. This system behaves similar to the stable resonator of Example 33.

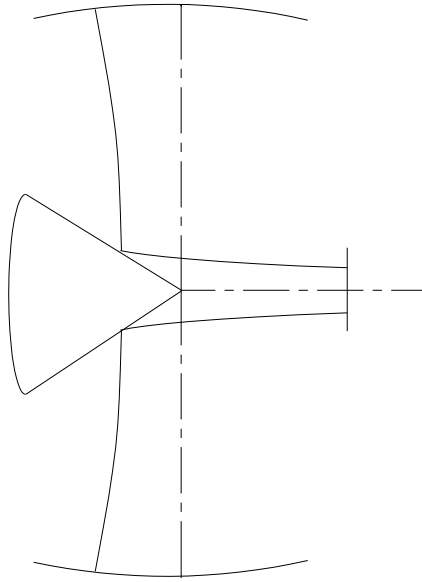


Fig. 75e.1. Configuration for stable resonator.

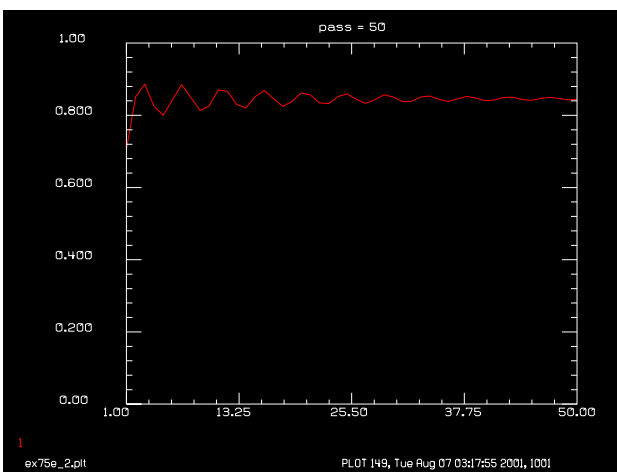
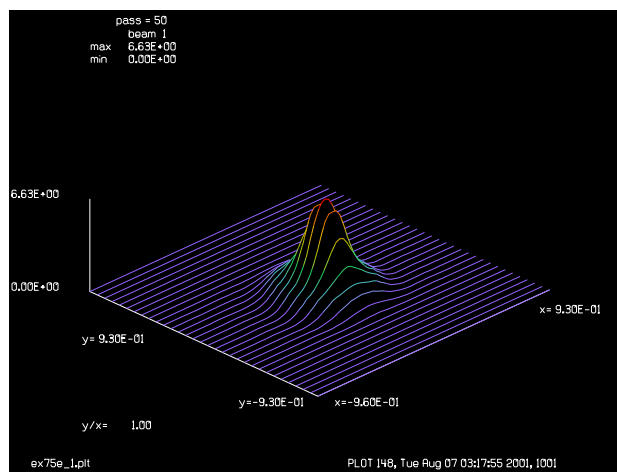


Fig. 75e.2. Output of stable resonator with axicon mirror. Fig. 75e.3. Convergence history of stable resonator.

Input: `ex75e.inp`

```
c## ex75e
c
c Example 75e: stable resonator with focal axicons
c
```

Jump to: [Commands](#), [Theory](#)

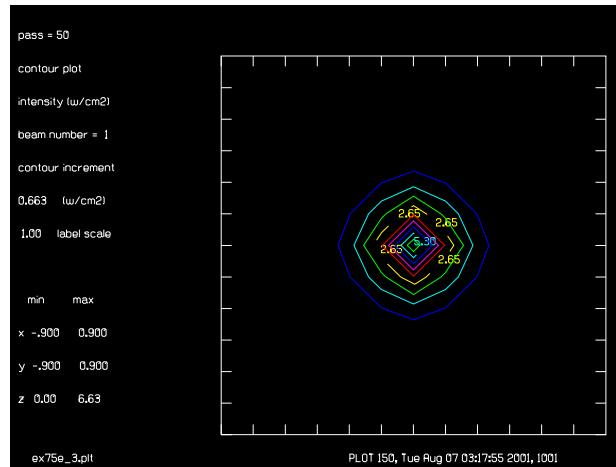


Fig. 75e.4. Contour plot of output of stable resonator with axicon.

```

variab/dec/int pass
wavelength/set 1 10.6
array/s 1 64
units/s 1 .03
set/density 12 12
c
c set up variable names using alias feature
c
startrad = .29          # stating beam radius
rad2 = 1                # radius in radial beam to be used after
                        # axicon

clap/c/no 1 startrad    # make an aperture
zbound/s 1 startrad     # establish surrogate gaussian data
obs 1 .0001             # block center point
energy/norm 1           # normalize energy
pass = 0                # initialize pass counter
macro/def ex75e/o
  pass = pass + 1
  mirror/flat 1          # flat end mirror
  zbound/s 1 startrad    # set surrogate gaussian radius
  zreff/se 1 0           # reinitialize axis
  dist -50               # propagate 50 cm to axicon
  clap/c/no 1 .6         # clear aperture for axicon
  axicon/radial 1 rad2    # expand to radial mode
  dist 40                # propagate 40 cm to mirror
  mirror/xcyl 1 -750. xonly # cylinder mirror with "xonly"
                        # to form other end mirror
  dist -40              # back to axicon
  axicon/axial 1 rad2    # recompact beam
  clap/c/no 1 .6
  dist 50                # back to flat end mirror
  status/p
  variab/set energy 1 energy
  udata/set pass pass energy
  title pass = @pass
c

```



```

c  make some pictures
c
  plot/watch ex75e_1.plt
  plot/1                                     # intensity plot
  plot/watch ex75e_2.plt
  plot/udata min=0 max=1                     # convergence history plot
  plot/watch ex75e_3.plt
  plot/con 1                                 # contour plot
  energy/norm 1
macro/end
macro/run ex75e/50

```

Ex75f: Unstable resonator with focal axicons

This example shows an unstable resonator with an axicon similar to Example 11.

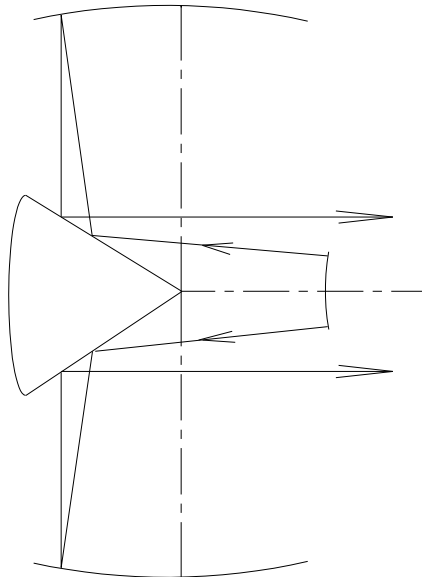


Fig. 75f.1. Configuration for unstable resonator with axicon. Having a limiting aperture in the compact leg is helpful in establishing good mode control.

Input: ex75f.inp

```

c## ex75f
c
c  Example 75f: unstable resonator with focal axicons
c
variab/dec/int pass
wavelength/set 1 10.6
array/s 1 64
nbeam 2
units/field 1 .6
set/density 12 12
c

```

Jump to: [Commands](#), [Theory](#)

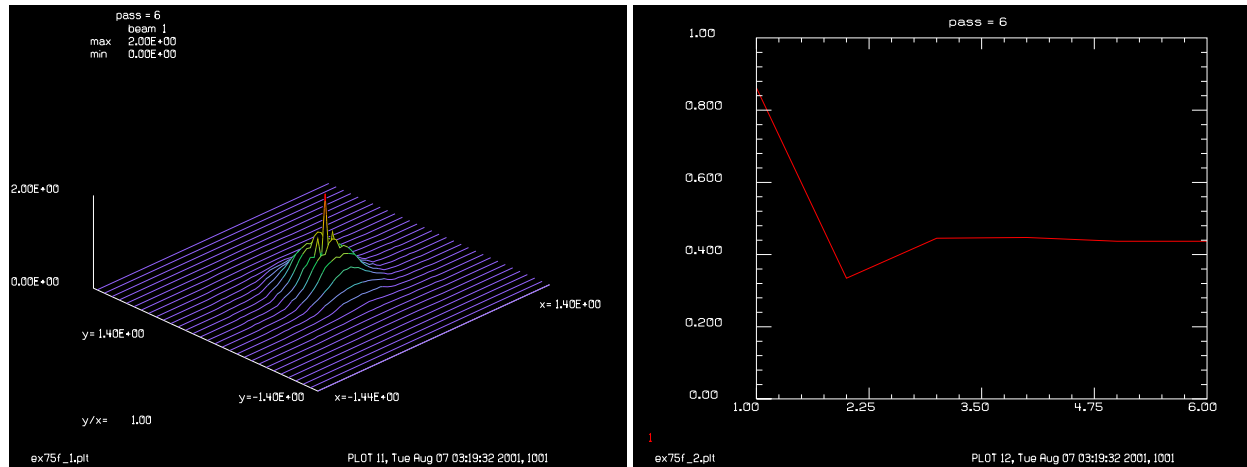


Fig. 75f.2. Output of stable resonator with axicon mirror. Fig. 75f.3. Convergence history of unstable resonator.

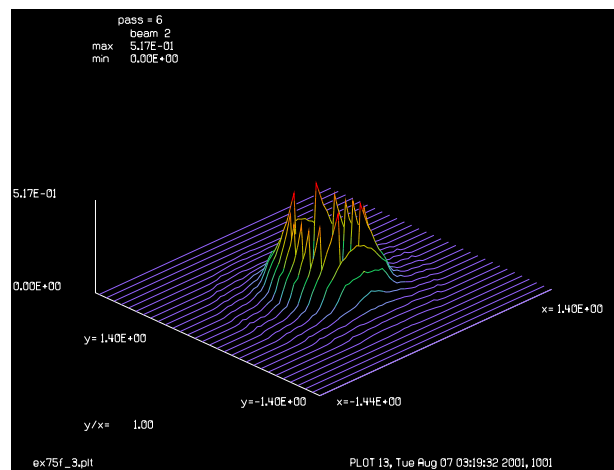


Fig. 75f.4. Output after scraper mirror.

```

c  set up variable names using alias feature
c
startrad = .3          # starting beam radius
rad2 = 1              # radius in radial beam to be used after
                        # axicon
clap/c/n 1 startrad    # make an aperture
zbound/s 1 startrad    # establish surrogate gaussian data
obs 1 .0001           # block center point
energy/norm 1          # normalize energy
pass = 0              # initialize pass counter
macro/def ex75e/o
  pass = pass + 1
  zbound/s 1 startrad
  mirror/sph 1 rad=180 # convex scraper mirror
  clap/c/n 1 startrad
  dist -5              # propagate -5 cm to axicon
  clap/c/n 1 rad2      # aperture of axicon, r2 = 1
  axicon/radial 1 rad2 # form radial mode
  dist 83              # propagate 83 cm to concave mirror

```

Jump to: [Commands](#), [Theory](#)

```

mirror/xcyl 1 -360 xonly # concave mirror, r=-360
clap/r/n 1 .7 1e8 -.7    # add an aperture in radial mode
dist -83                 # return to axicon
axicon/axial 1 rad2      # reform axial mode
clap/c/n 1 rad2          # aperture of axicon, r2 = 1
dist 5
title pass = @pass
copy 1 2
variab/se energy 1 energy
udata/set pass pass energy
plot/watch ex75f_1.plt
plot/l 1
plot/watch ex75f_2.plt
plot/udata min=0 max=1
rescale 1 .6             # rescale beam for next pass
energy/norm 1
macro/end
macro/run ex75e/6
obs 2 startrrad
plot/watch ex75f_3.plt
plot/l 2
end

```

Ex75g: Form annular beam from radial beam, reflaxicon

A reflaxicon consists of two cones facing in the same direction. Opposing left and right reflaxicons may be used to generate an annular beam and to reconstruct a well corrected axial beam. Both intensity and polarization properties are well behaved. The representation of the annular beam is similar to that in the radial beam with a small change of the intensity.

Input: ex75g.inp

```

c## ex75g
c
c Example 75g: Form annular beam from radial beam, reflaxicon
c
c An axicon converts an axial beam to a radial beam. A second
c axicon may be used to convert the radial beam to an annular
c beam with the optical axis consisting of a cylinder.
c
c In this example, a two left cones are used to form a reflaxicon
c which converts an axial beam into a annular beam. A second
c reflaxicon consisting of two right cones reconverts the beam
c to axial form.
c
set/density 12 12
echo/on
radius = 2                # exit radius for radial mode
array/s 1 128 128 1
units/set 1 .05 .05
clap/c/n 1 radius
obs 1 .001
energy/norm 1

```

Jump to: [Commands](#), [Theory](#)

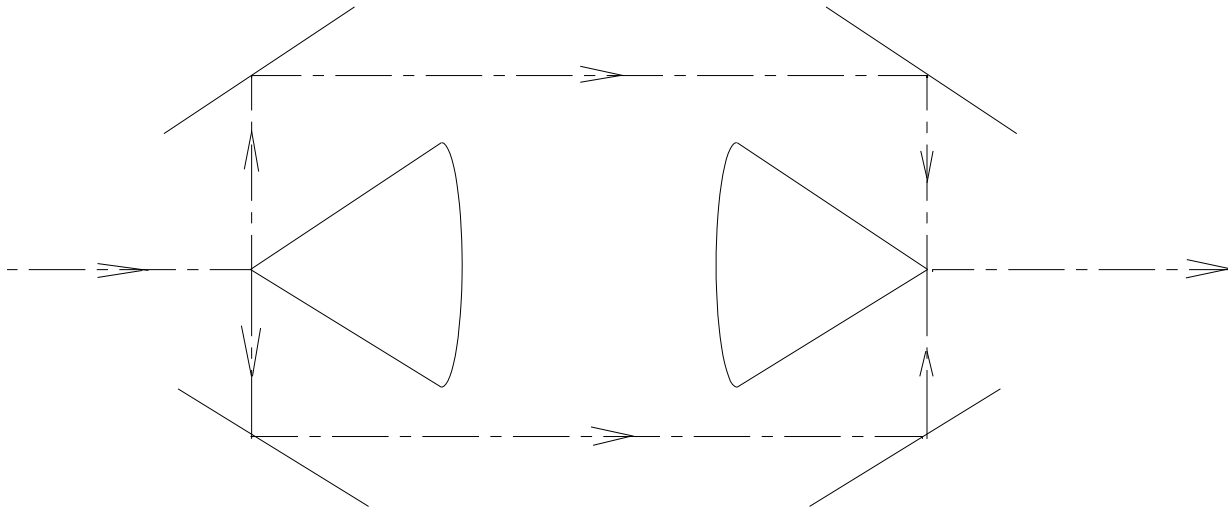


Fig. 75g.1. Configuration of two reflexicons which generates an annular beam and after recompaction reforms as an axial beam.

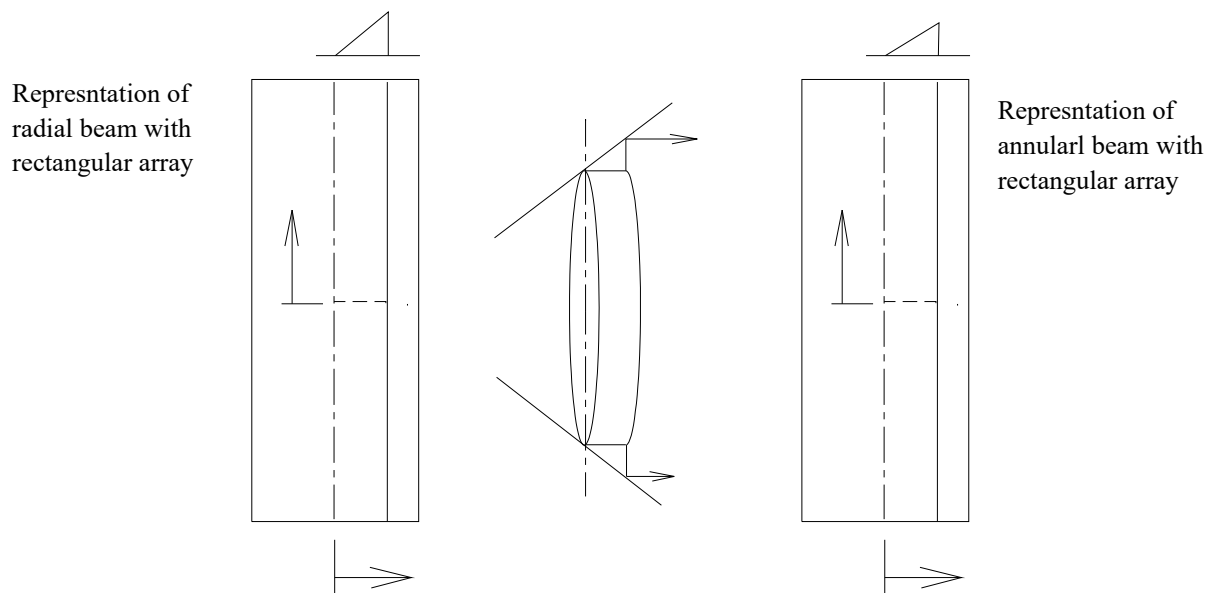


Fig. 75g.2. Both radial and annular beams are represented in radial-azimuthal form with the azimuthal direction corresponding to the y-direction and filling the rectangular array with no guard band.

```
title starting axial beam
plot/watch ex75g_1.plt
plot/ell 1
dist 10
```

Jump to: [Commands](#), [Theory](#)

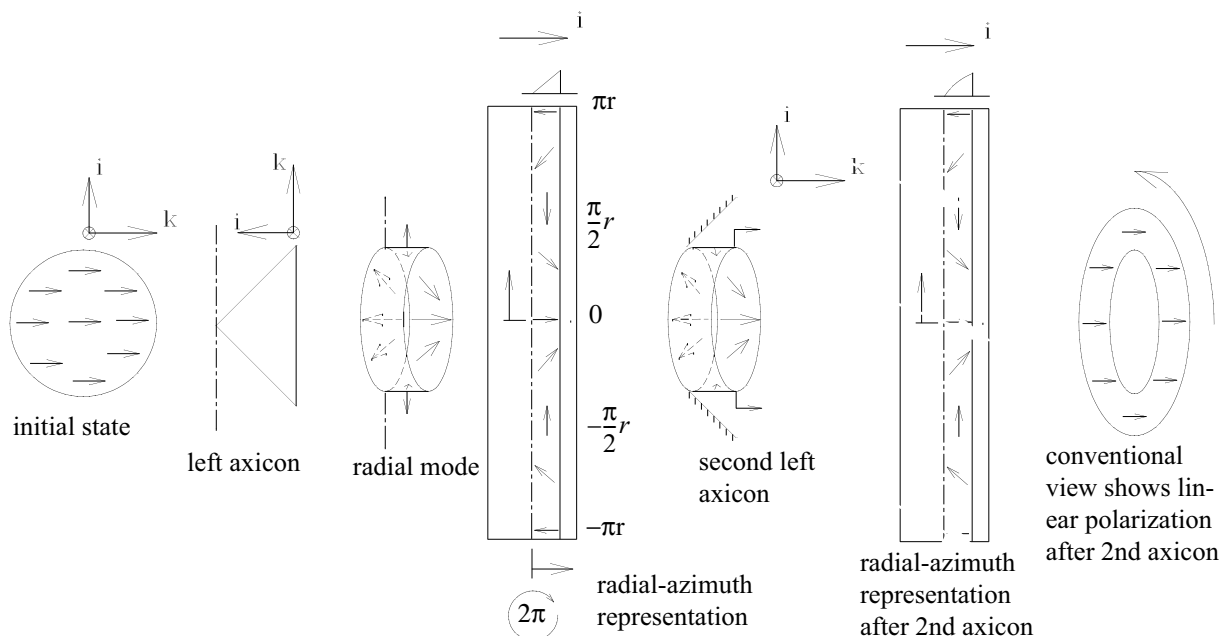


Fig. 75g.3. Polarization mapping for a left reflexicon. A circular beam with horizontal linear polarization is reflected off a left axicon mirror, forming a radially expanding beam (radial mode). For radial-azimuth display, the beam is stretched into a rectangular array with the vertical direction spanning $-\pi r$ to $+\pi r$. The polarization state will be rotated clockwise for one cycle in the $+y$ -direction. After reflection from a second left axicon, the polarization state retains the same state of rotation and is linear in the conventional view.

```

axicon/radial/left/focal 1 radius      # axial to radial
title expanding radial path
plot/watch ex75g_2.plt
plot/e 1
axicon/annular/left 1                  # radial to annular
title annular beam
plot/watch ex75g_3.plt                  # intensity in annular beam
plot/l 1
plot/watch ex75g_4.plt                  # intensity in annular beam
plot/e 1
axicon/radial/right 1                  # begining of second reflexicon
jones/set ar=-1. dr=1.
jones/mult 1
title contracting radial beam
plot/watch ex75g_5.plt
plot/ell 1                             # polarization in 2nd radial beam
axicon/axial 1 2                       # reform axial beam
title recompact beam
plot/watch ex75g_6.plt
plot/ell 1                             # recompact axial beam

```

Ex75h: Form annular beam from radial beam, waxicon

A waxicon consists of a pair of left and right axicon mirrors. Two waxicons of opposite orientation may be used to form an annular beam and then reconstruct a well behaved axial beam.

Jump to: [Commands](#), [Theory](#)

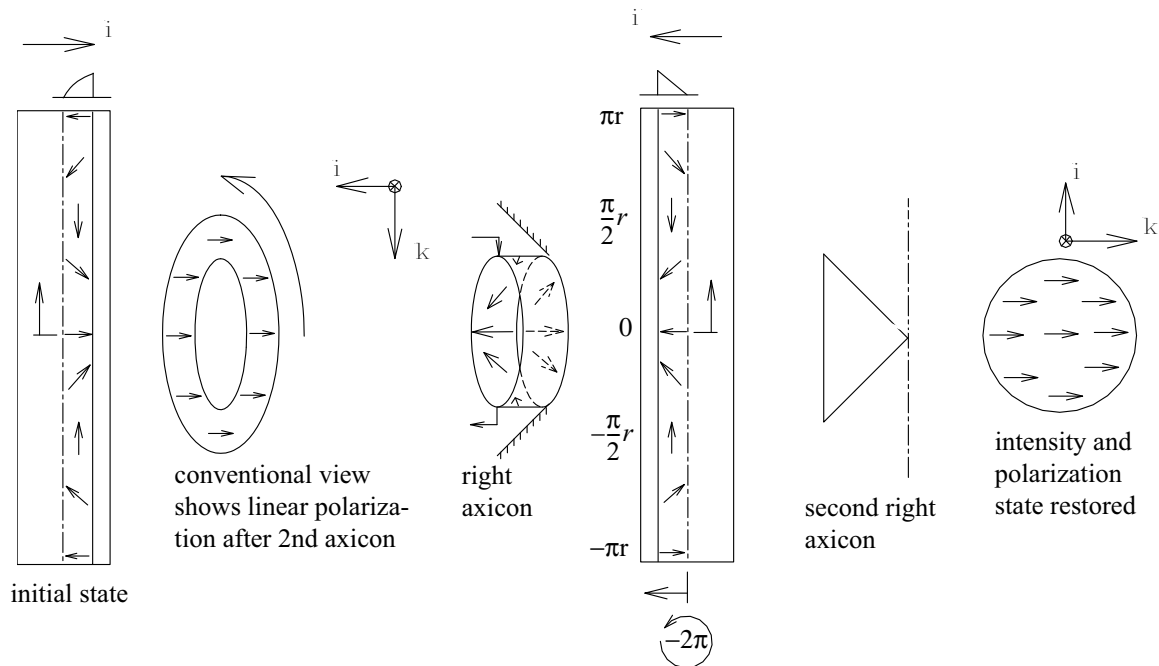


Fig. 75g.4. Polarization mapping for a right reflexicon following the left reflexicon. An annular beam with horizontal linear polarization is reflected off a right axicon mirror, forming a radially collapsing beam (radial mode). For radial-azimuth display, the beam is stretched into a rectangular array with the vertical direction spanning $-\pi r$ to $+\pi r$. The polarization state will be rotated clockwise for one cycle in the $+y$ -direction. After reflection from a second right axicon, the polarization state returns to the original linear state.

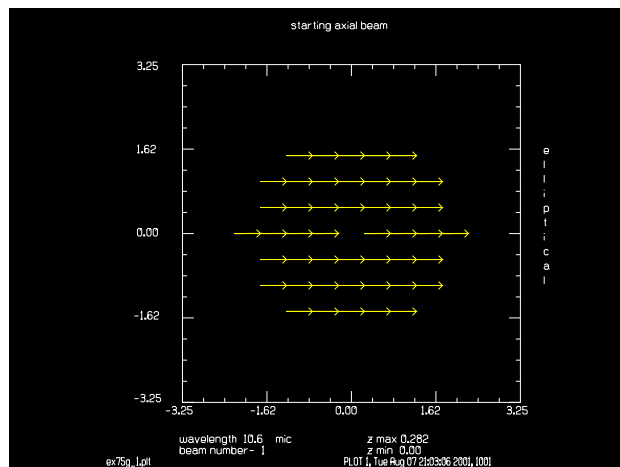


Fig. 75g.5. Polarization at start.

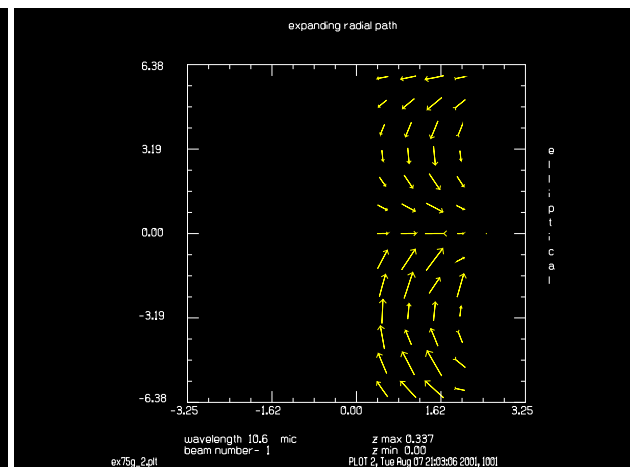


Fig. 75g.6. Intensity in expanding radial path.

Input: `ex75h.inp`

```
c## ex75h
c
c Example 75h: Conversion of axial beam to annular beam by waxicons
c
c Similar to Ex75g except that waxicons are used to create the annular
```

Jump to: [Commands](#), [Theory](#)

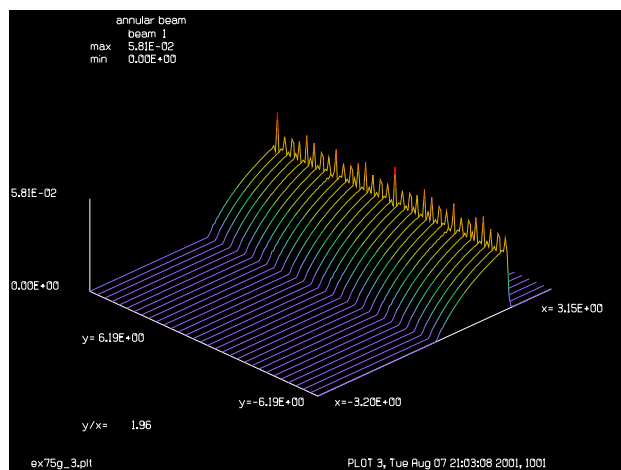


Fig. 75g.7. Intensity in annular beam. Note change in intensity due to conversion to annular form.

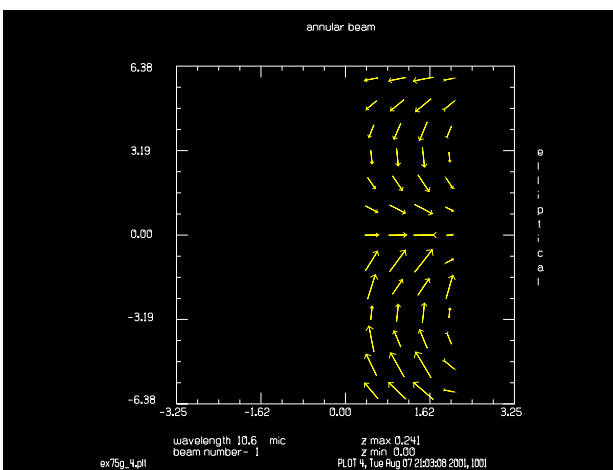


Fig. 75g.8. Polarization is well behaved in annular path.

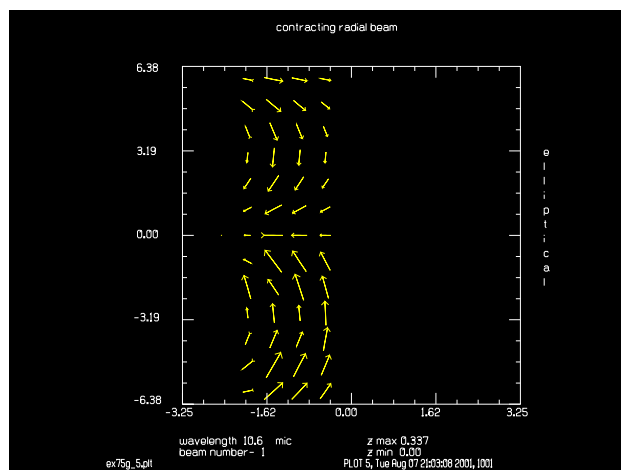


Fig. 75g.9. Polarization in contracting radial beam.

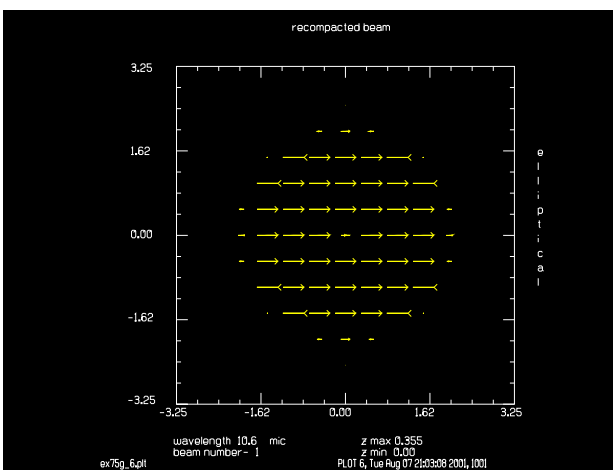


Fig. 75g.10. Polarization at end shows well corrected behavior.

c beam and to reform the axial beam. The polarization of the annular beam c is disrupted but the reformed axial beam has well behaved properties.

c

echo/on

radius = 2.5

array/s 1 128 128 1

units/set 1 .05 .05

clap/c/c 1 2

obs 1 .001

energy/norm 1

set/density 12 12

title starting axial beam

plot/watch ex75h_1.plt

plot/ell 1

dist 5

axicon/radial 1 radius

display starting distribution

form radial beam with left axicon

Jump to: [Commands](#), [Theory](#)

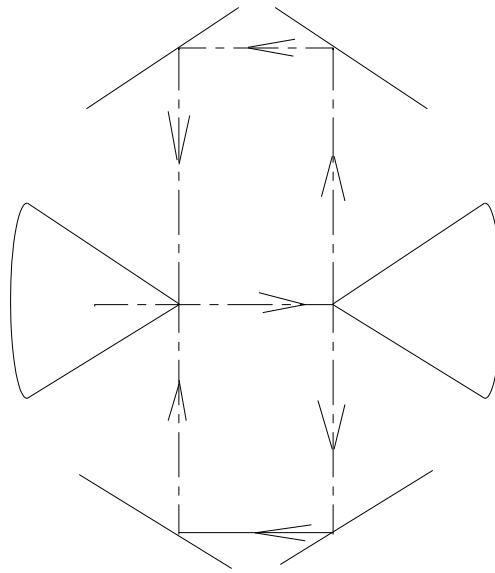


Fig. 75h.1. Configuration of two waxicons of opposite orientation. Note that the optical axis is on the outside of the annular region. A resonator could be made by putting concave or convex mirrors in the axial region.

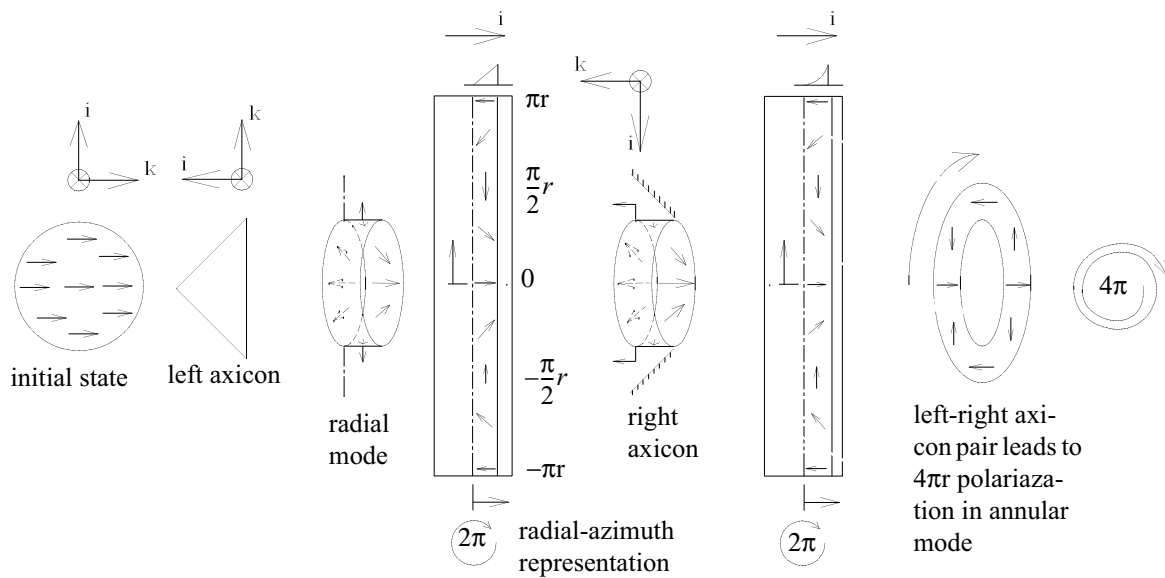


Fig. 75h.2. Polarization mapping for a waxicon. Same as Fig. 75k3 except that the 2nd axicon is a right axicon and reverses the propagation direction. The true aperture shape now rotates counterclockwise leading to a 4π rotation when viewed in the conventional view for one cycle in the $+y$ direction.

```
title expanding axial beam
plot/watch ex75h_2.plt      # plot radial beam
plot/ell 1
```

Jump to: [Commands](#), [Theory](#)

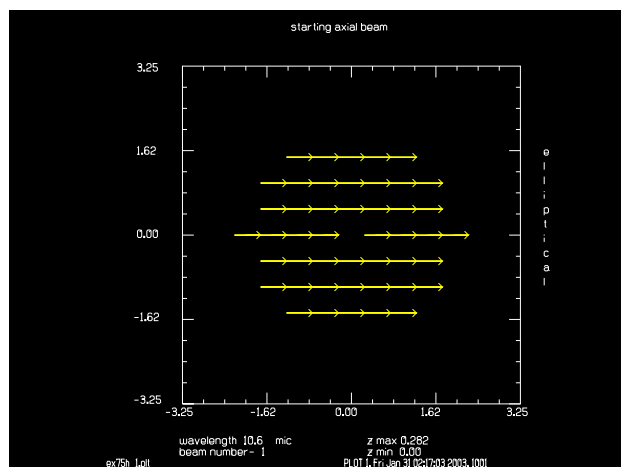


Fig. 75h.3. Starting polarization distribution in axial beam.

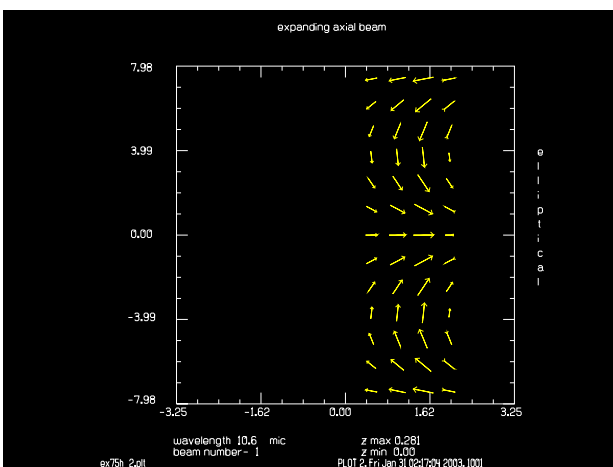


Fig. 75h.4. Intensity in radial region.

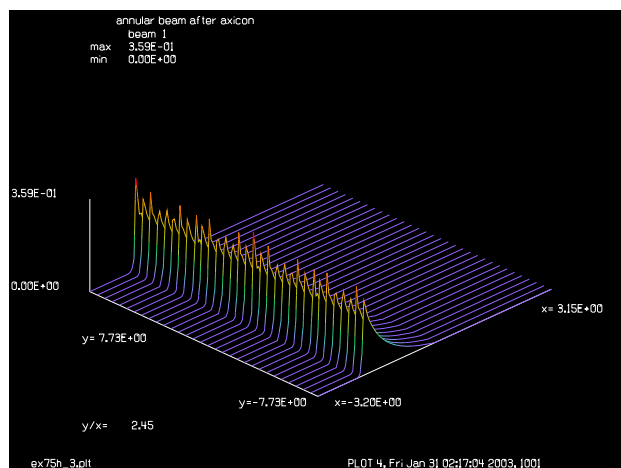


Fig. 75h.5. Irradiance in annular region.

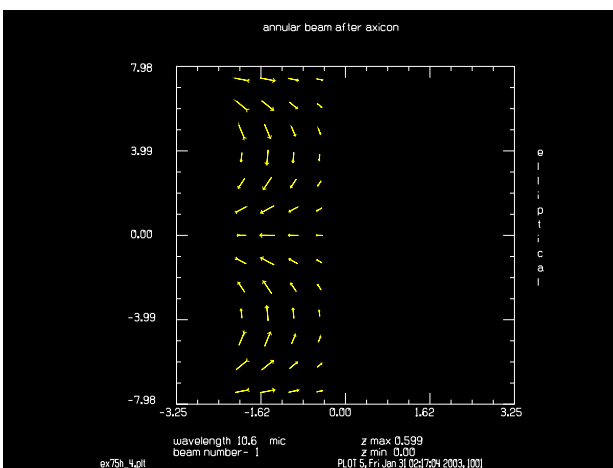


Fig. 75h.6. Polarization in annular leg is well behaved.

```

c
c This is a patch
c
flip/x
jones/set ar=-1 dr=1
jones/mult 1
plot/w ex75hx_2.plt
plot/e 1
c
c End of patch
c
axicon/annular/right 1           # make annular beam with right axicon
                                # second part of first waxicon

title annular beam after axicon
plot/watch ex75h_3.plt          # note polarization rotation in annular beam
plot/l 1
plot/watch ex75h_4.plt          # note polarization rotation in annular beam

```

Jump to: [Commands](#), [Theory](#)

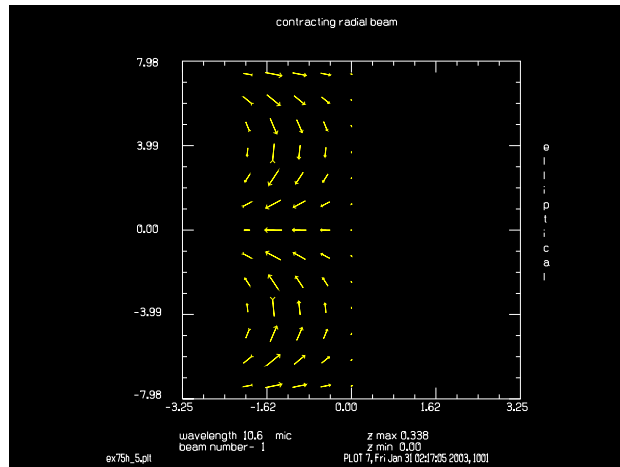


Fig. 75h.7. Polarization in contracting radial region is flipped to align with initial coordinate system.

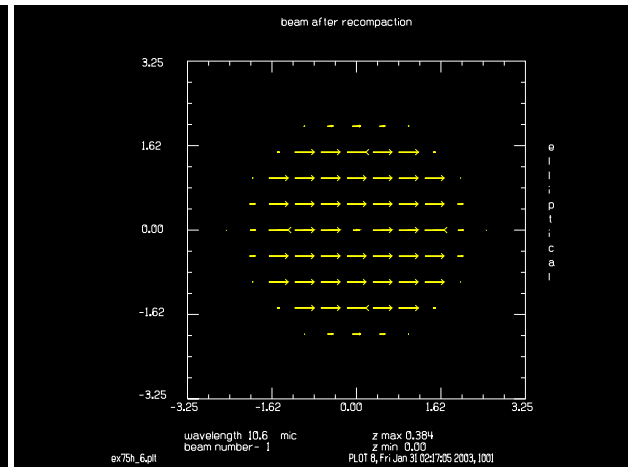


Fig. 75h.8. Final polarization distribution in axial beam, showing well corrected structure.

```

plot/ell 1
dist -20
plot/watch ex75hx_4.plt          # note polarization rotation in annular beam
plot/ell 1
axicon/radial/left/focal 1      # form radial beam with left axicon
                                # the first part of second waxicon

title contracting radial beam
plot/watch ex75h_5.plt
plot/ell 1
axicon/axial 1 radius           # for axial beam to conclude second waxicon
title beam after recombination
plot/watch ex75h_6.plt
plot/ell 1

```

Ex75i: Reflaxicon-waxicon pair with intermediate ring focus

Generally using a reflaxicon and a waxicon will result in the same nonuniform intensity and polarization scrambling as the use of a left and right axicon, as illustrated in Ex75c. By using toroidal optics to form a ring focus in the annular path it is possible to flip the irradiance distribution with respect to the cylindrical axis. The recompact axial beam will have well behaved intensity, the axis will be correctly returned to a line, but the polarization will be disrupted. This is because the ring focus does not flip the polarization when it flips the intensity. It is necessary to area-weight the intensity to have correct treatment of the toroidal ring focus. In this example, the diffraction propagation in the azimuthal direction is only approximately correct.

Input: ex75i.inp

```

c## ex75i
c
c Example 75i: Reflaxicon-waxicon pair with intermediate ring focus
c
c A pair of reflaxicons (or a pair of waxicons) will result in a
c well formed axial beam. Using a reflaxicon with a waxicon will
c cause undesirable intensity and polarization properties.

```

Jump to: [Commands](#), [Theory](#)

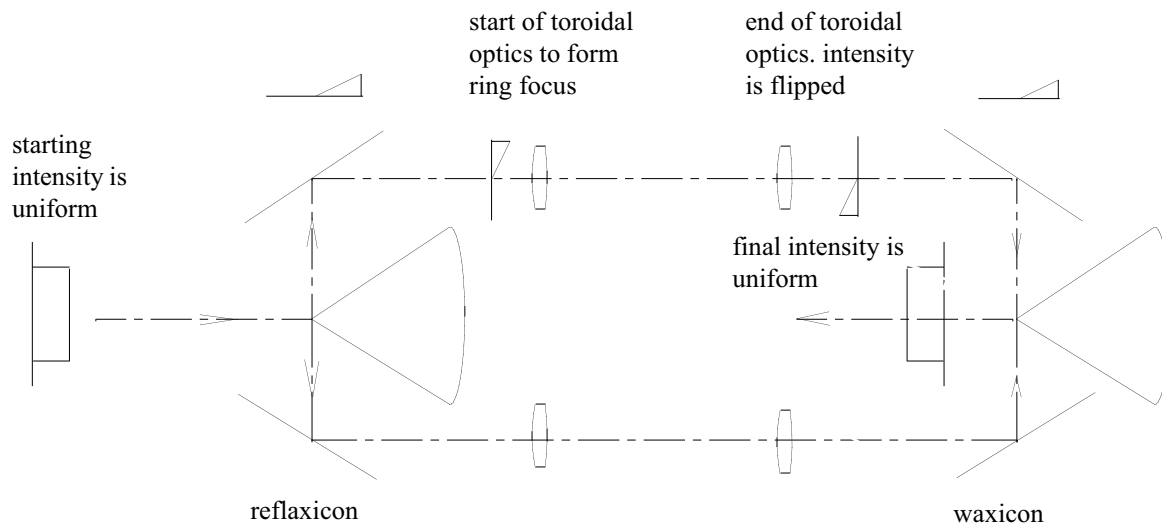


Fig. 75i.1. Refluxicon-waxicon with intermediate ring focus. The refractive optics in the annular path are toroidal lenses which flip the intensity about the cylindrical axis but do not change the polarization.

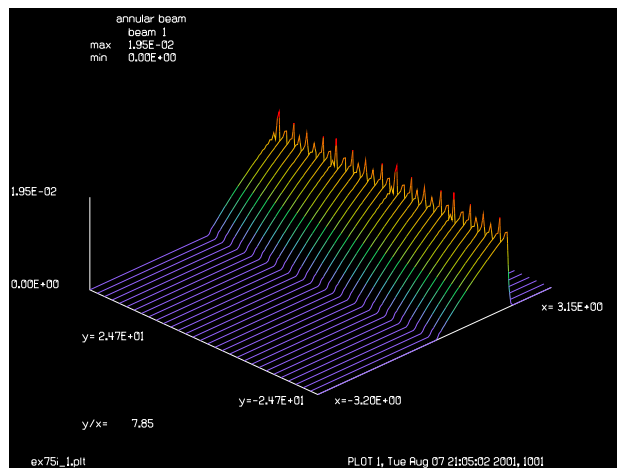


Fig. 75i.2. Annular beam just before ring focus.

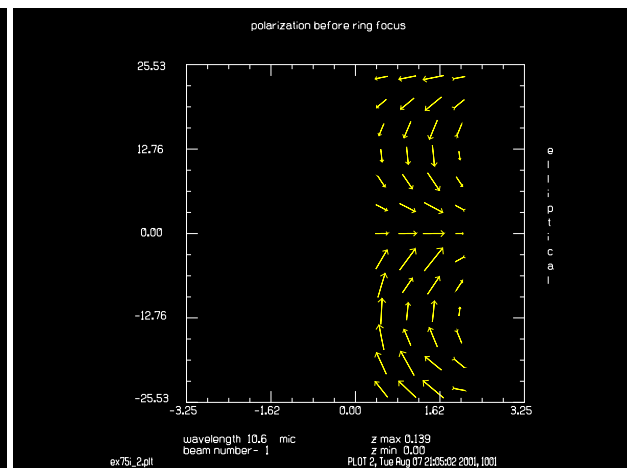


Fig. 75i.3. Polarization before ring focus.

```

c
c We can use an intermediate ring focus to correct the intensity pattern,
c but the polarization will remain disrupted.
c
c set switch to: 0 for normal refluxicon,
c                 gives good intensity and polarization
c                 1 for toroidal image and reversed compaction
c                 cone, gives good intensity but poor polarization
c
variab/dec/int switch
switch = 1

```

Jump to: [Commands](#), [Theory](#)

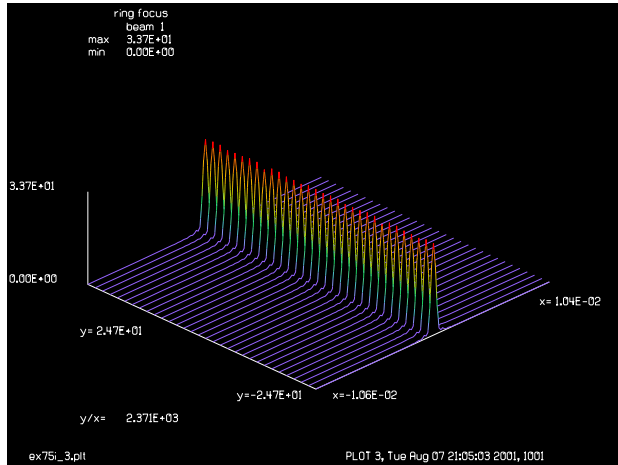


Fig. 75i.4. Intensity at ring focus.

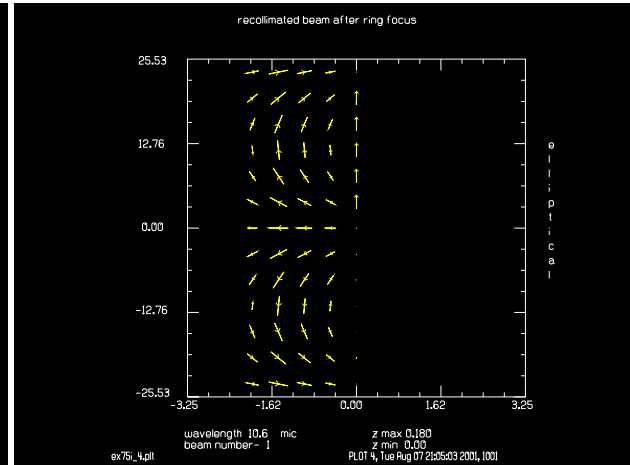


Fig. 75i.5. Polarization after ring focus.

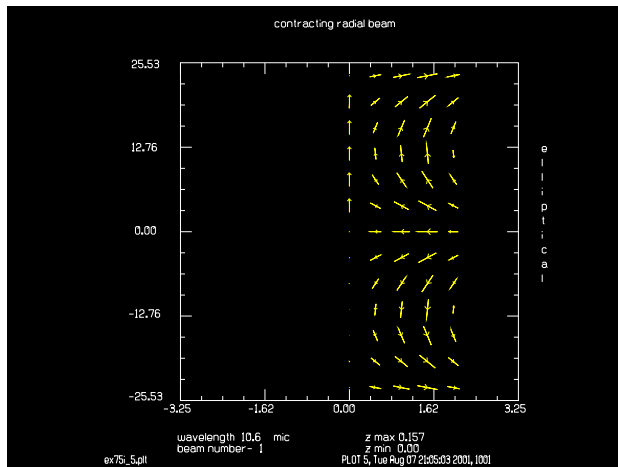


Fig. 75i.6. Polarization in contracting radial leg.

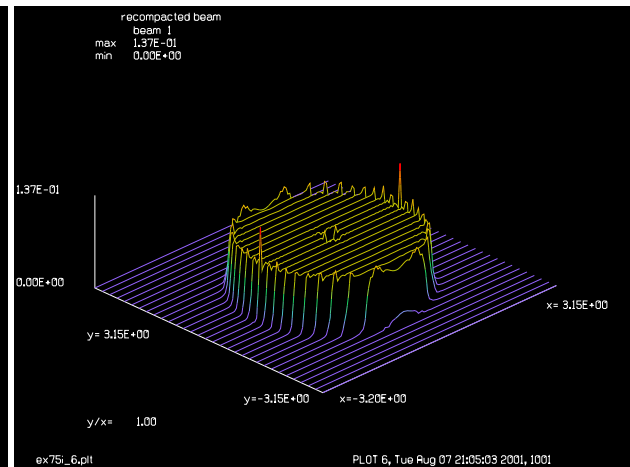


Fig. 75i.7. Intensity in recompacked axial beam is close to the starting flat top form.

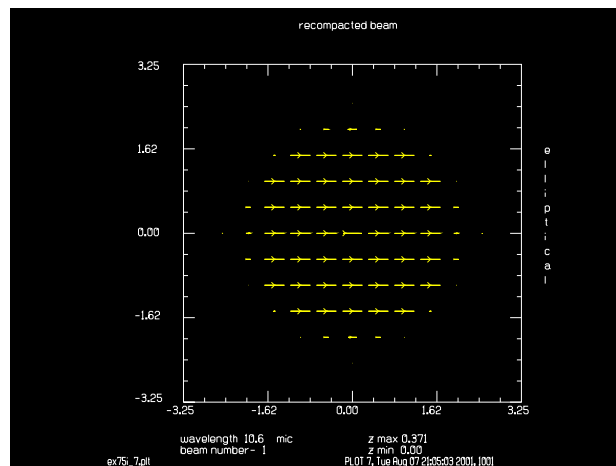


Fig. 75i.8. The polarization in the recompacked axial beam is disrupted.

```

c
echo/on
wavelength/set 1 10.6
radius = 8          # set axicon exit radius
array/s 1 128 128 1
units/set 1 .05 .05
clap/c/c 1 2
obs 1 .001
set/density 12 12
energy/norm 1
c
c Begin with reflaxicon
c
axicon/radial/left/focal 1 8
energy
axicon/annular/left 1
title annular beam
plot/watch ex75i_1.plt
plot/l 1
status/p
c
c The next lines are switchable logic to examine the normal
c reflaxicon-waxicon case and a reflaxicon-waxicon with internal ring
c focus.
c
if switch = 0 then
c
c option 0: simple propagation in annular region
c
    dist 2
endif
if switch = 1 then
c
c option 1: beam inversion by torioidal lens system
c
    title polarization before ring focus
    plot/watch ex75i_2.plt
    plot/ell 1
    lens/xcyl 1 1          # cylindrical lens of 1 cm focal length
    dist 1                 # propagate to ring focus
    plot/watch ex75i_3.plt
    title ring focus
    plot/l 1               # ring focus
    dist 1
    lens/xcyl 1 1          # recollimating cylindrical lens
endif
energy
plot/watch ex75i_4.plt
title recollimated beam after ring focus
plot/ell 1
field 1
axicon/radial/right/focal 1 # return to radial mode
title contracting radial beam
plot/watch ex75i_5.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/ell 1
energy
axicon/axial/left 1 radius      # return to axial mode
energy
title recompacked beam
plot/watch ex75i_6.plt
plot/l 1                        # note flat hat intensity
plot/watch ex75i_7.plt
plot/ell 1                      # note polarization is disrupted

```

Ex75j: Effect of decenter and tilt on beam in radial mode

Tipping or decentering of an axicon or of toroidal or cylindrical optics in the radial path will result in a special form of tilt aberration and piston aberration which varies azimuthally. These aberrations are defined by `abr/radial/tilt` and `abr/radial/decenter` and may only be applied when the beam is in radial mode. Tilt is applied with the optical axis being the center of the aberration. This corresponds to tipping an axicon about its vertex.

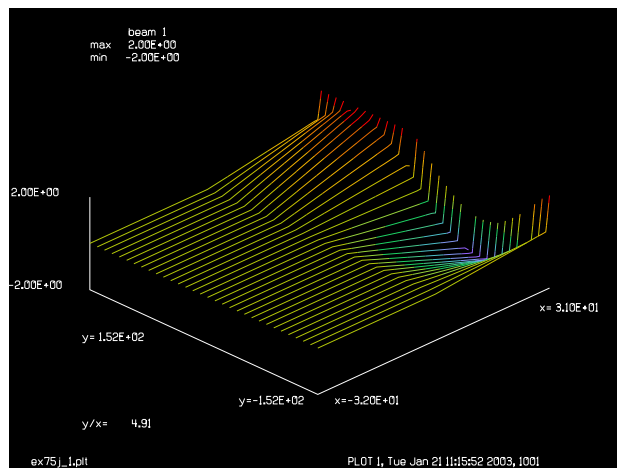


Fig. 75j.1. Phase plot of tilt aberration on a radial beam. The vertical direction corresponds to the azimuthal direction.

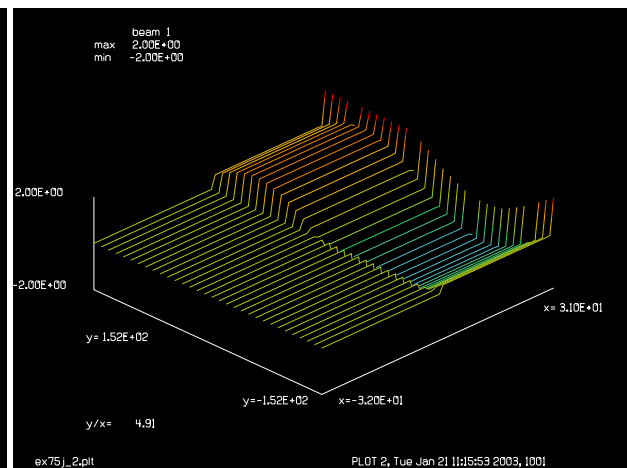


Fig. 75j.2. Phase plot of decenter aberration on a radial beam.

Input: ex75j.inp

```

c## ex75j
c
c Example 75j: effect of decenter and tilt on beam in radial mode
c
radius = 30
clap/c/c 1 radius
axicon/radial 1 50
set/density 12 12
c
c tilt aberration for radial mode
c
c          kbeam waves azimuth normalizing radius

```

Jump to: [Commands](#), [Theory](#)

```

abr/radial/tilt 1      -.2      90      20
plot/watch ex75j_1.plt
plot/l/ph 1 min=-2 max=2

array/set 1 64 64
units/set 1 1
clap/c/c 1 radius
axicon/radial 1 50
c
c decenter aberration for radial mode
c
c          kbeam  waves  azimuth
abr/radial/decenter 1      -.2      90
plot/watch ex75j_2.plt
plot/l/ph 1 min=-2 max=2

```

Ex75k: Waxicon resonator with inverting optics (simple treatment)

A more complex a waxicon is shown below. It consists of two waxicon end mirrors and a reflaxicon consisting of a figure-of-rotation rhomb mirror with cylindrical internal inverting to flip the irradiance distribution in the radial direction. If perfectly constructed the waxicon, reflaxicon, and inverting optics result in perfect round-trip behavior: a collimated beam injected at the left axicon will exit from the reflaxicon as uniform phase, uniform polarization state beam. This configuration is shown in Fig. 75k2. Ordinary flat mirrors are used as scraper and feedback optics and the beam is a simple collimated beam in this path. A pair of figure-of-rotation cylindrical lenses are optically equivalent to the reflecting inversion optics, as shown in Fig. 75k2.

While this configuration is drawn with all cones having 45 degree apex angles, its performance will be representative of arrangements with angles at other than 45 degrees as the only effect will be slight changes in the radial remapping of intensity. The inversion optics will, in general, require some aspheric coefficient correction from a pure toroidal shape to achieve aberration-free performance. If the aberration correction is done properly, the performance will approach the performance of the idealized system.

The polarization mapping is somewhat complex and is illustrated in Figs. 75k3 and 75k4. After reflection from the inner left cone, the beam is essentially expanding as a cylindrical shell, which in GLAD is called radial mode. Cones are called "left" if they point to the left and right otherwise. We show radial mode on a radial-azimuth plot with the x-direction showing the radial variation and the y-direction spanning $-\pi$ to $+\pi$. The positive radial direction corresponds to the i-vector as shown in Figs. 75k1 and 75k2 where an ijk coordinate system is propagated through the system. The annular path is also displayed in a radial-azimuth plot, although it represents an annular section.

The irradiance in the radial path has a strong wedge profile because of the nonlinear mapping of the radial zones. This wedge is partially compensated by the outer cone of a reflaxicon pair and accentuated by the outer cone of a waxicon pair as indicated schematically in the Fig. 75k3 and 75k4.

To illustrate the effect a collimated beam was injected at Point 1 as shown in Fig. Ex75k2 and followed around the system. The beam had a round flat-top profile with an offset obscuration to illustrate the beam orientation. The inversion optics flip the beam in the radial direction so that the wedge irradiance is corrected (see Figs. Ex75k17-18), even though the beam is outcoupled by a right cone rather than a left cone as was used initially. The performance of the nominal design is essentially perfect, as demonstrated by Figs.

Jump to: [Commands](#), [Theory](#)

Ex 75k19-20, although the irradiance is substantially wedged in the annular path which may reduce the efficiency of energy extraction or have some other reduction of efficiency.

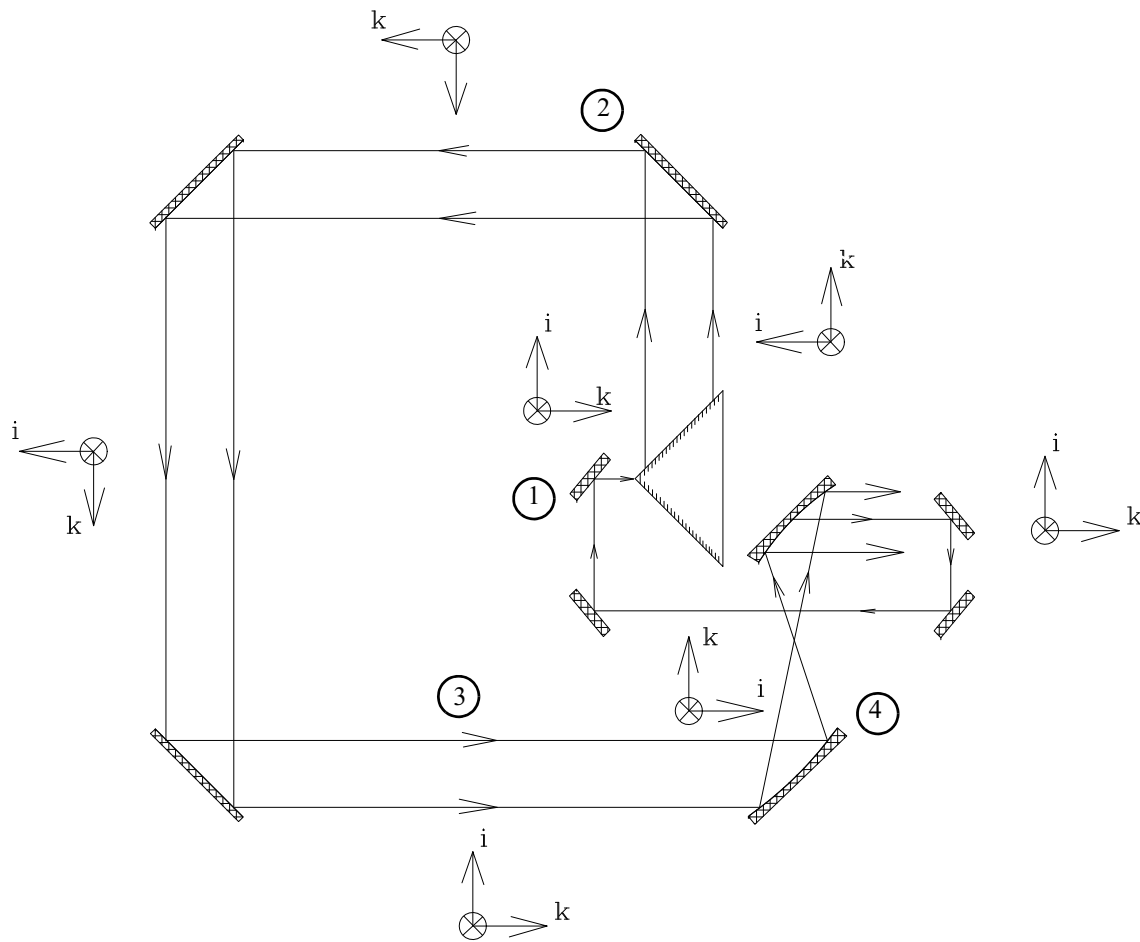


Fig. 75k.1. Starting from the center and moving right (1), a nominally collimated beam intercepts a left inner cone and right outer cone forming a waxicon. Moving backward (2), the light intercepts a second waxicon consisting of two outer cones, with the light moving forward again (3), the light intercepts a reflaxicon pair with internal inverting optics (4). The beam is recompact by the reflaxicon and returns to its collimated, flat-top form. A set of ordinary flat mirrors act as scrappier mirrors and reinject the collimated beam back at (1).

Input: `ex75k.inp`

```
c## ex75k
c
c Example 75k: Waxicon resonator with internal focus in radial path
c
c Simulation with simple axicons.
c
c In this example, a left cone creates a radial path
c followed by a right cone, forming a waxicon configuration
```

Jump to: [Commands](#), [Theory](#)

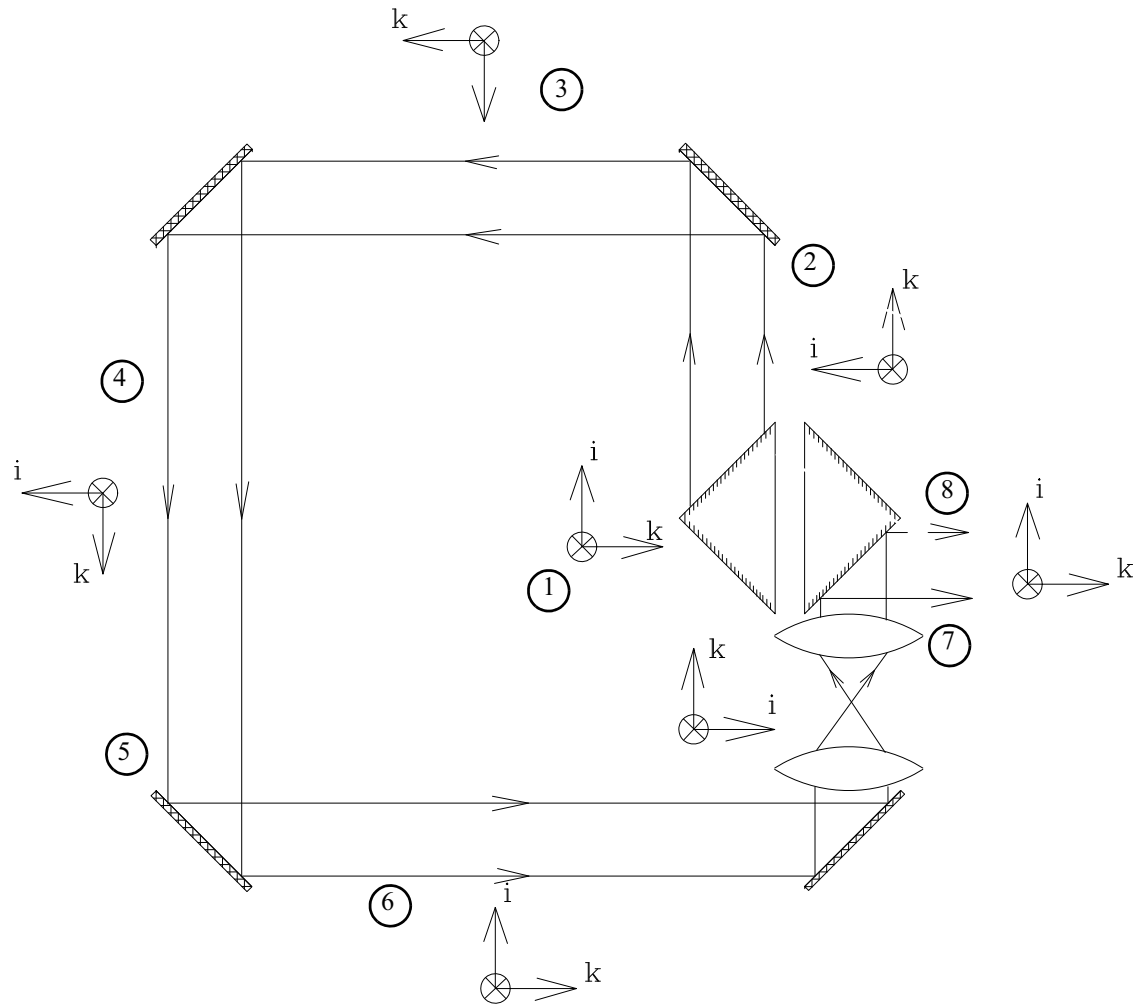


Fig. 75k.2. The reflecting inversion optics function the same, in the ideal case, as a pair of figure-of-rotation cylindrical lenses. The feedback optics are omitted in this view. The ijk vectors are drawn to indicate both the propagation direction and the azimuthal orientation of the beam. The view is of the x - z plane and the j -vector points out of the page. Numbers help identify plots ex75k5-20.

```

c  and creating an annular beam moving backward.
c
c  A second pair of left cones forms a second waxicon arrangement
c  with the beam crossing over the axis.
c
c  A final pair of right cones forms a reflaxicon configuration which
c  recompacts the beam.
c
c  An internal focus in the radial path flips the irradiance distribution
c  so the final right axicon recompacts the beam with flat irradiance
c  and well-behaved polarization.
c
c  Optical power and aberrations may be added by using MIRROR/GLOBAL
c  in radial mode. The aberrations are computed only along the x-direction
c  and applied in rotationally symmetric fashion for array attribute

```

Jump to: [Commands](#), [Theory](#)

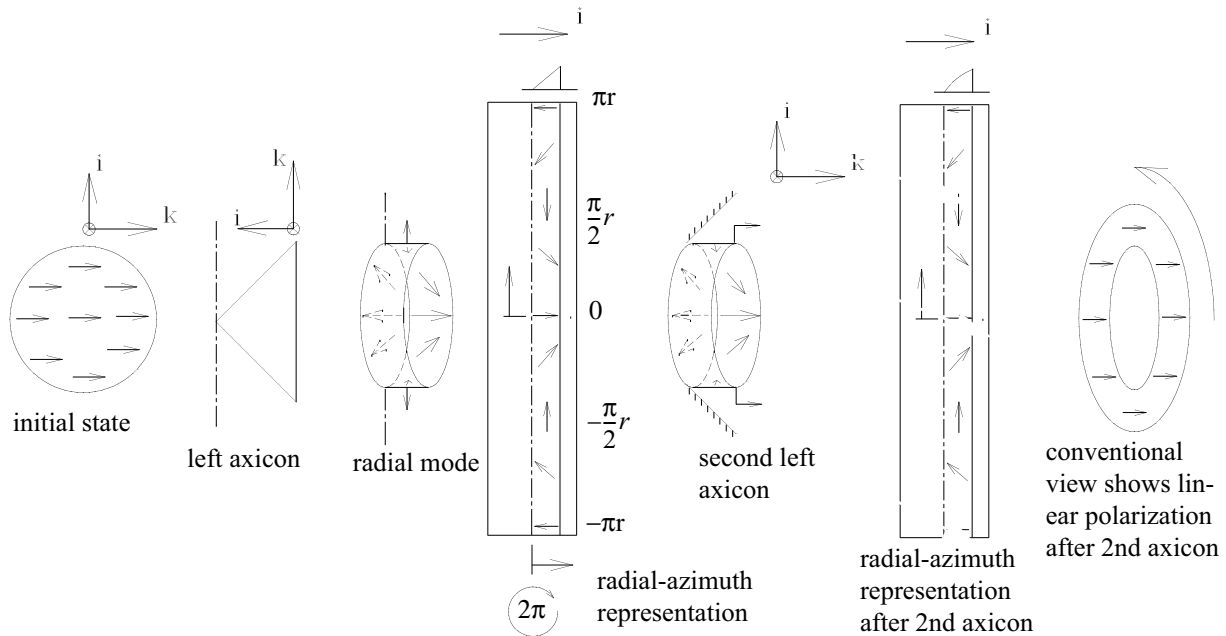


Fig. 75k.3. Polarization mapping for a left reflexicon. A circular beam with horizontal linear polarization is reflected off a left axicon mirror, forming a radially expanding beam (radial mode). For radial-azimuth display, the beam is stretched into a rectangular array with the vertical direction spanning $-\pi r$ to $+\pi r$. The polarization state will be rotated clockwise for one cycle in the $+y$ -direction. After reflection from a second left axicon, the polarization state retains the same state of rotation and is linear in the conventional view.

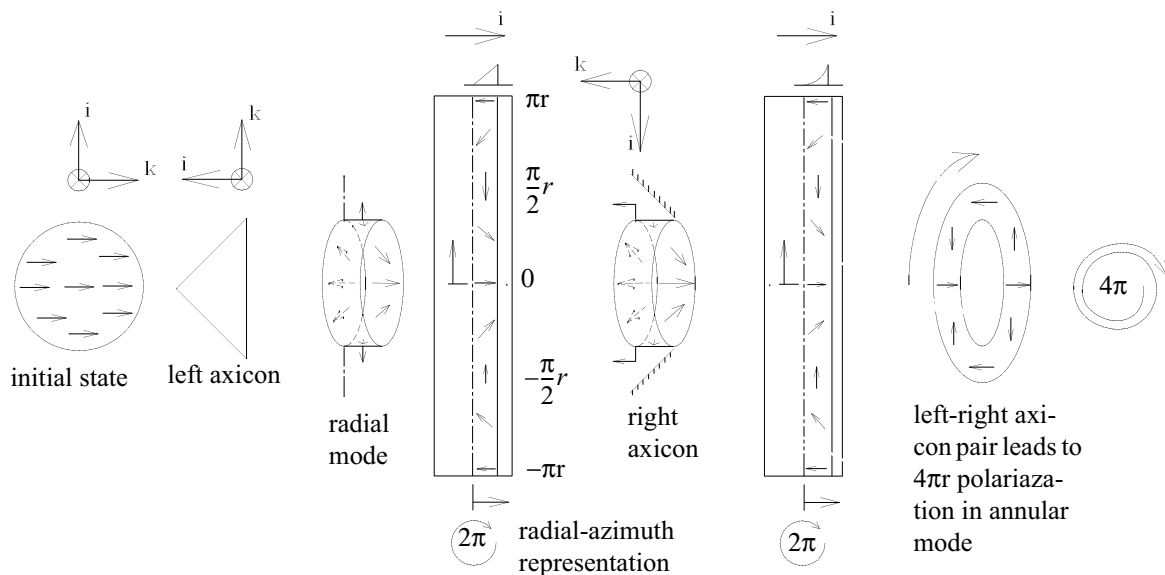


Fig. 75k.4. Polarization mapping for a waxicon. Same as Fig. 75k.3 except that the 2nd axicon is a right axicon and reverses the propagation direction. The true aperture shape now rotates counterclockwise leading to a 4π rotation when viewed in the conventional view for one cycle in the $+y$ direction.

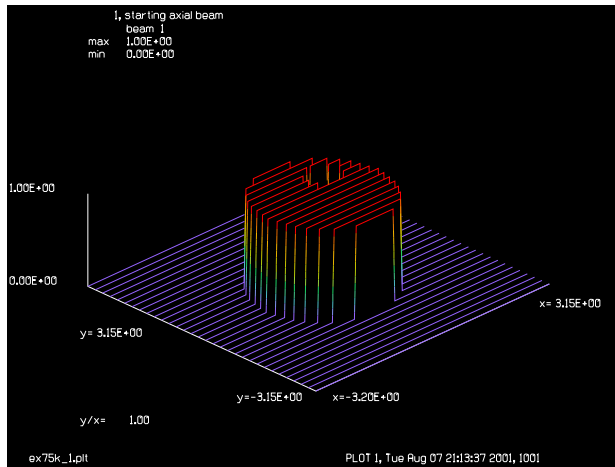


Fig. 75k.5. Point 1. Collimated beam with offset obscuration to illustrate beam orientation.

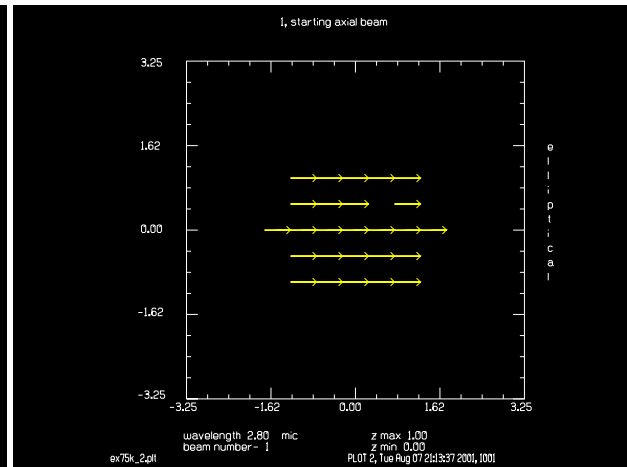


Fig. 75k.6. Point 1. Horizontal linear polarization.

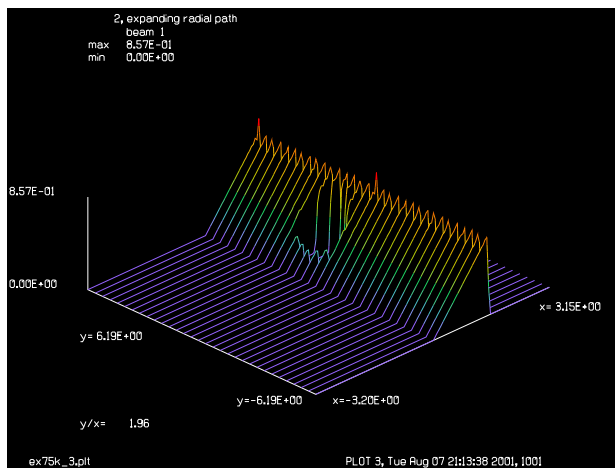


Fig. 75k.7. Point 2. Radial-azimuth intensity from $-\pi r$ to $+\pi r$ in vertical.

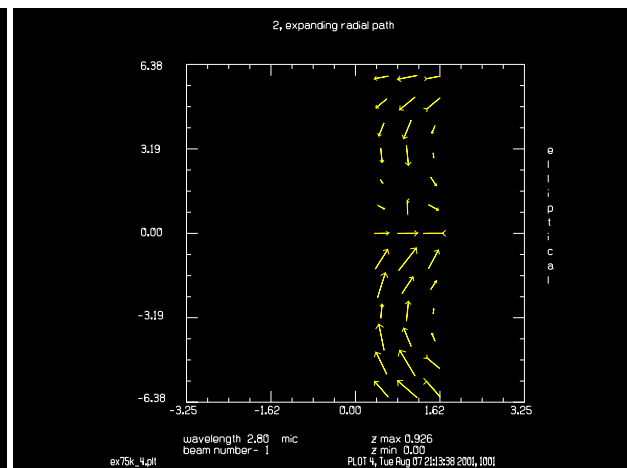


Fig. 75k.8. Point 2. Radial-azimuth polarization from $-\pi r$ to $+\pi r$ in vertical.

c of beam and projected in the y-direction for radial and annular modes.

c

c Aberration must be in the form of ordinary even powers of radius
c and must be centered about the axis. The axis is
c on the outside of the annular path, so the mirrors are centered
c on the top of the annular path.

c

c A decentered obscuration is introduced to display the beam orientation
c at the various positions.

c

Apt_inner = 2 # radius of inner axicon

Apt_outer = 4 # radius of outer axicon

Length1 = 2 # length of resonator

Length2 = 4 # end mirror to outcoupling axicon

set/density 12 12

echo/on

Jump to: [Commands](#), [Theory](#)

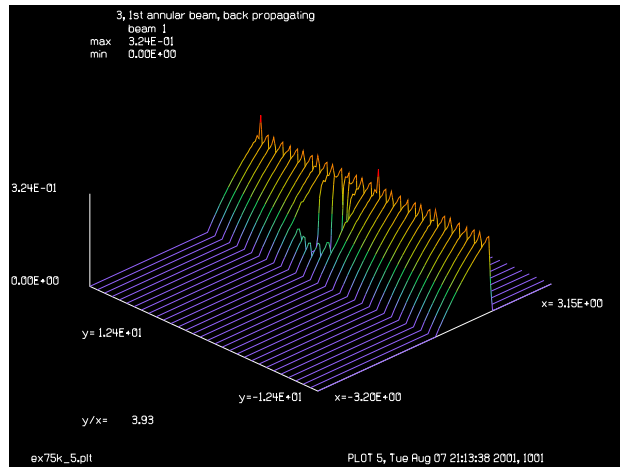


Fig. 75k.9. Point 3. Radial-azimuth intensity after first outer cone. Annular mode.

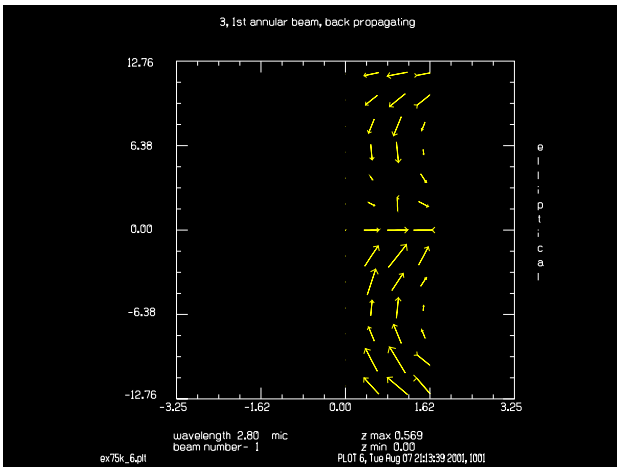


Fig. 75k.10. Point 3. Radial-azimuth polarization after first outer cone. Annular mode.

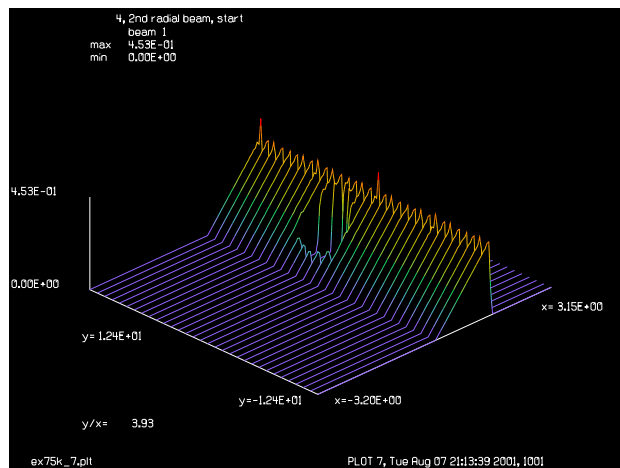


Fig. 75k.11. Point 4. Radial-azimuth intensity after 2nd outer cone. Inward radial mode.

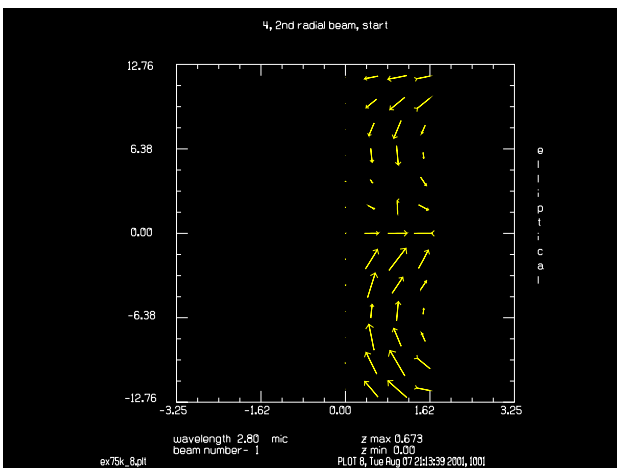


Fig. 75k.12. Point 4. Radial-azimuth polarization after 2nd outer cone. Inward radial mode.

```
array/s 1 128 128 1
units/set 1 .05 .05
wavelength/set 1 2.8
clap/c/c 1 Apt_inner
obs 1 .001
energy/norm 1
clear 1 1
clap/c/c 1 1.5
obs/c 1 .4 xdec = .6 ydec=.6
title 1, starting axial beam
plot/watch ex75k_1.plt
plot/l 1
plot/watch ex75k_2.plt
plot/e 1
dist 10
B0=0.
```

Jump to: [Commands](#), [Theory](#)

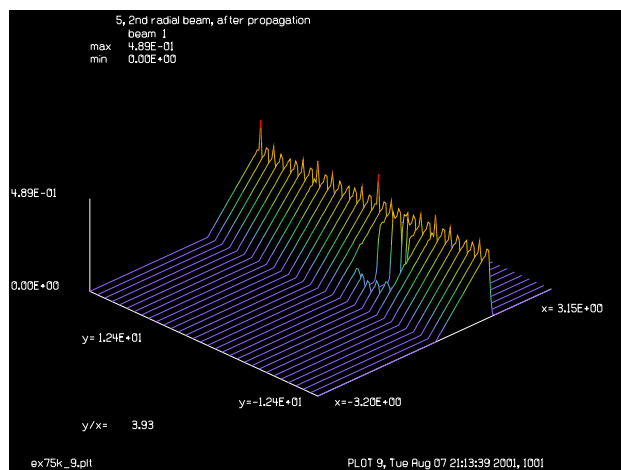


Fig. 75k.13. Point 5. Radial-azimuth intensity after 3rd outer cone. Note flip in azimuth direction.

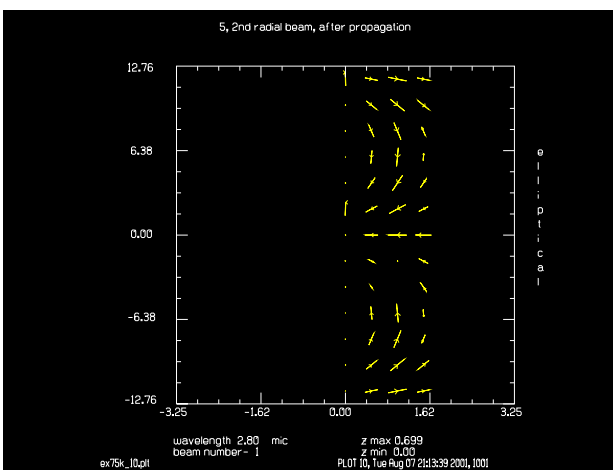


Fig. 75k.14. Point 5. Radial-azimuth polarization after 3rd outer cone. Note flip in azimuth direction.

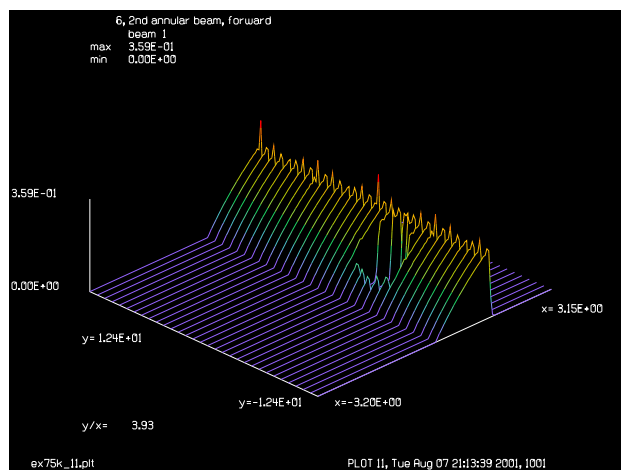


Fig. 75k.15. Point 6. Radial-azimuth intensity after 4th outer cone.

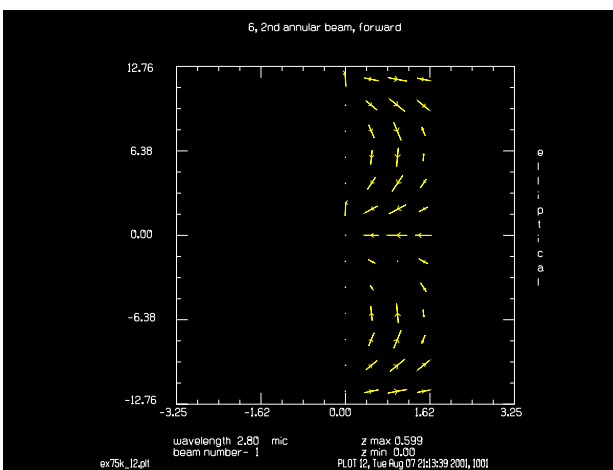


Fig. 75k.16. Point 6. Radial-azimuth polarization after 4th outer cone.

```

B1=0.
B2=1e-10.    # should not be exactly zero
B4=0.
B6=0.
B8=0.
c Propagate to first axicon, WIC
axicon/radial/left/focal 1 Apt_inner    # axial to radial
title 2, expanding radial path
plot/watch ex75k_3.plt
plot/l 1
plot/watch ex75k_4.plt
plot/e 1
global
c Propagate to outer edge of waxicon, WOC
prop Apt_outer-Apt_inner
axicon/annular/right 1                # radial to annular

```

Jump to: [Commands](#), [Theory](#)

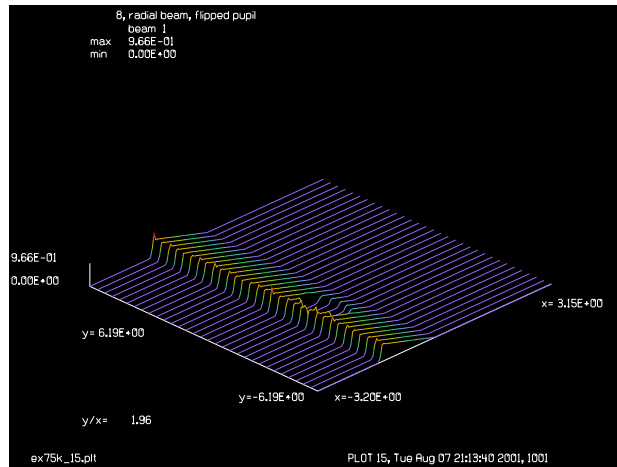


Fig. 75k.17. Point 7. Radial-azimuth intensity after 4th inversion optics. Note flip of intensity.

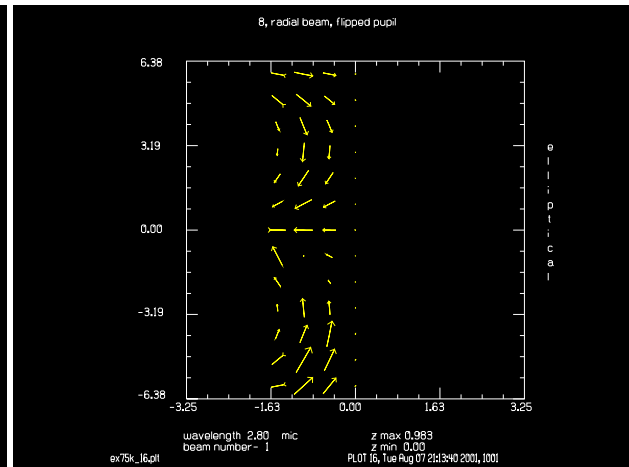


Fig. 75k.18. Point 7. Radial-azimuth polarization after 4th inversion optics. Note flip of intensity.

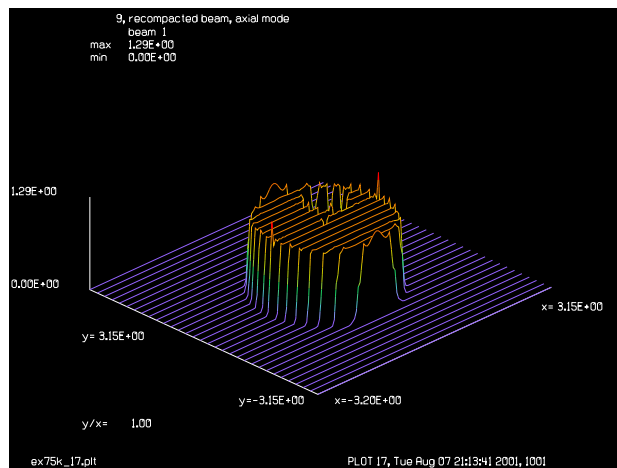


Fig. 75k.19. Point 8. Reformed pupil in collimated beam, shows offset obscuration.

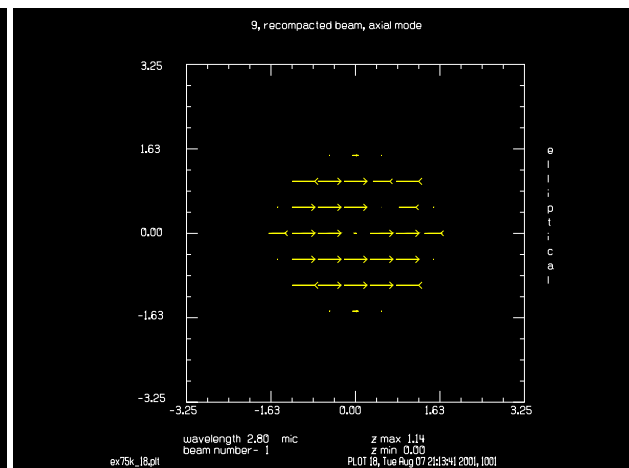


Fig. 75k.20. Point 8. Reformed pupil in collimated beam, shows polarization preserved.

```

title 3, 1st annular beam, back propagating
plot/watch ex75k_5.plt          # intensity in annular beam
plot/l 1
plot/watch ex75k_6.plt          # intensity in annular beam
plot/e 1
c Propagate to rear cone, RC
prop Length1
axicon/radial/left 1            # 2nd outer axicon reflection
title 4, 2nd radial beam, start
plot/watch ex75k_7.plt          # intensity in annular beam
plot/l 1
plot/watch ex75k_8.plt          # intensity in annular beam
plot/e 1
c Propagate to other side of rear cone, RC
prop 2*Apt_outer
title 5, 2nd radial beam, after propagation

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch ex75k_9.plt          # intensity in annular beam
plot/l 1
plot/watch ex75k_10.plt         # intensity in annular beam
plot/e 1
axicon/annular/left 1
title 6, 2nd annular beam, forward
plot/watch ex75k_11.plt         # intensity in annular beam
plot/l 1
plot/watch ex75k_12.plt         # intensity in annular beam
plot/e 1
global
c Propagate to outer reflaxicon, ROC
prop Length2
axicon/radial/left 1
lens/xcyl 1 Apt_outer/2
global
title 7, radial beam, start
plot/watch ex75k_13.plt         # intensity in annular beam
plot/l 1
plot/watch ex75k_14.plt         # intensity in annular beam
plot/e 1
global
c Inner reflaxicon, RIC
c Propagate to axis
prop Apt_outer
lens/xcyl 1 Apt_outer/2.
prop -Apt_inner
title 8, radial beam, flipped pupil
plot/watch ex75k_15.plt         # intensity in annular beam
plot/l 1 h=.1
plot/watch ex75k_16.plt         # intensity in annular beam
plot/e 1
axicon/axial/right 1 -Apt_inner
title 9, recompact beam, axial mode
plot/watch ex75k_17.plt         # intensity in annular beam
plot/l 1
plot/watch ex75k_18.plt         # intensity in annular beam
plot/e 1
global

```

Ex75l: Waxicon resonator with inverting optics (exact mirror treatment)

In Ex75l, the problem is repeated with aspheric mirrors using the `mirror/global` form. Each mirror is positioned so that the vertex of the mirror intercepts the optical axis and the figure of the mirror is calculated along the x-radius and applied radially. The `axicon` command is invoked with the "nomirror" switch, so that only the intensity remapping occurs. When the distribution is reimaged across the center of the system, the distribution is not flipped as azimuthal angle is measured with respect to the axis in the "exact" case. Utilities `wic.txt`, `woc.txt`, `rc.txt`, `roc.txt`, `ric.txt` are included for conversion of global mirror aspheric coefficients to vertex form.

Jump to: [Commands](#), [Theory](#)

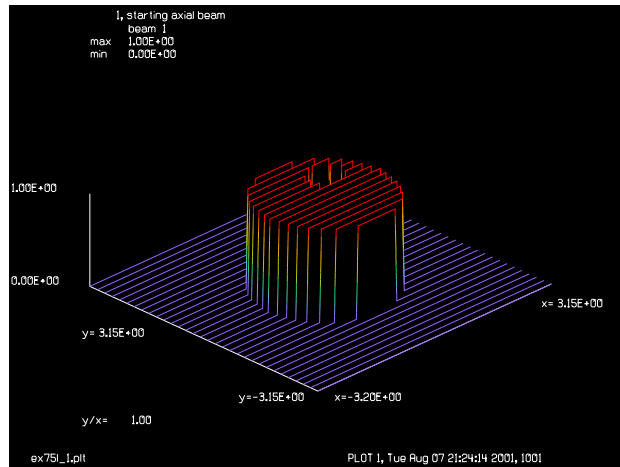


Fig. 75l.1. Point 1. Collimated beam with offset obscuration to illustrate beam orientation.

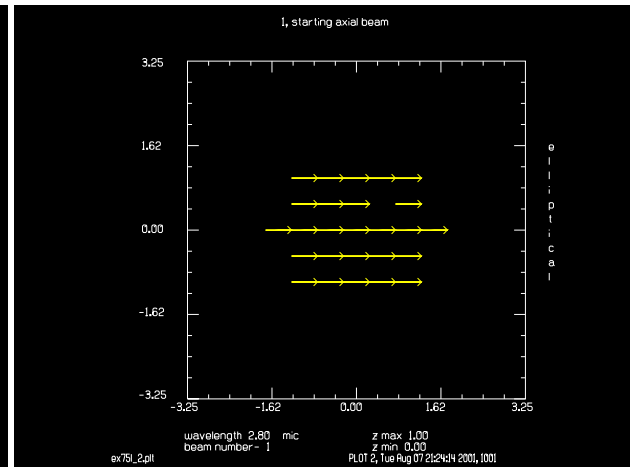


Fig. 75l.2. Point 1. Horizontal linear polarization.

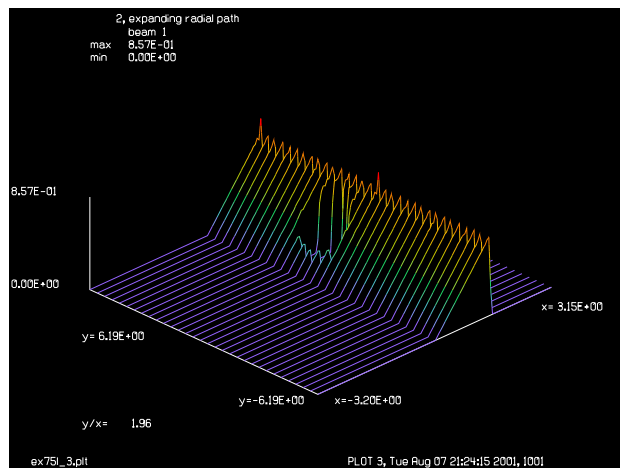


Fig. 75l.3. Point 2. Radial-azimuth intensity from $-\pi r$ to $+\pi r$ in vertical.

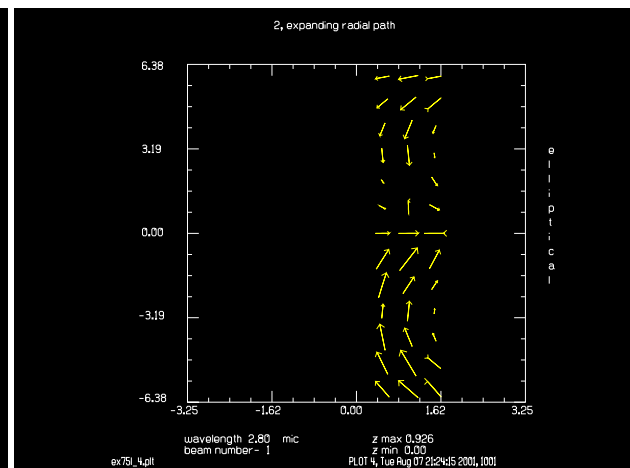


Fig. 75l.4. Point 2. Radial-azimuth polarization from $-\pi r$ to $+\pi r$ in vertical.

Input: `ex75l.inp`

```
c## ex75l
c
c Example 75l: Waxicon resonator with internal focus in radial path
c
c Exact mirror treatment.
c
c In this example, a left cone creates a radial path
c followed by a right cone, forming a waxicon configuration
c and creating an annular beam moving backward.
c
c A second pair of left cones forms a second waxicon arrangement
c with the beam crossing over the axis.
c
```

Jump to: [Commands](#), [Theory](#)

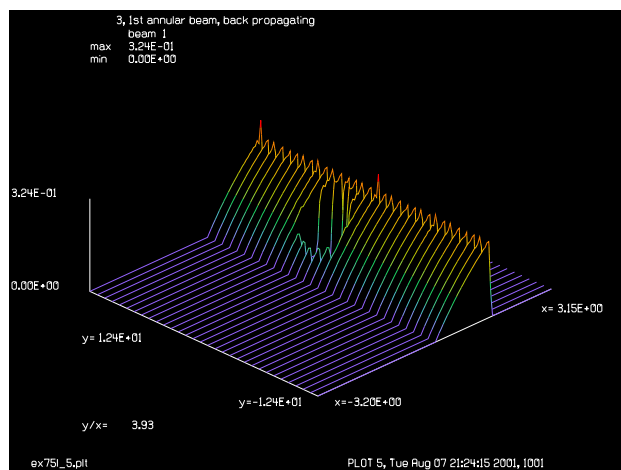


Fig. 75l.5. Point 3. Radial-azimuth intensity after first outer cone. Annular mode.

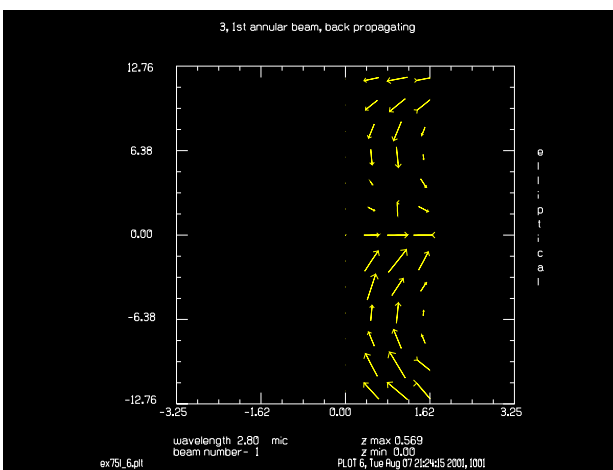


Fig. 75l.6. Point 3. Radial-azimuth polarization after first outer cone. Annular mode.

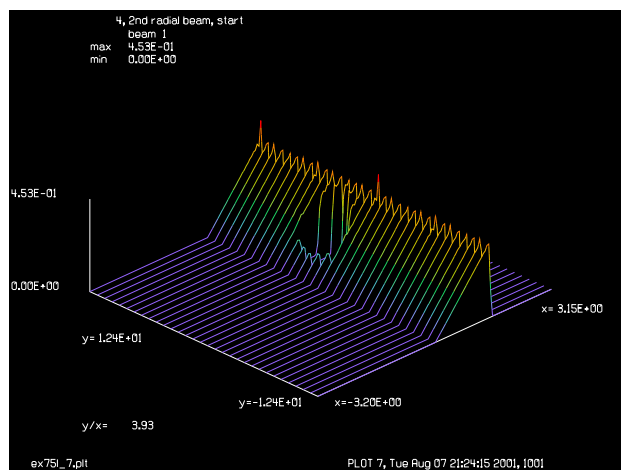


Fig. 75l.7. Point 4. Radial-azimuth intensity after 2nd outer cone. Inward radial mode.

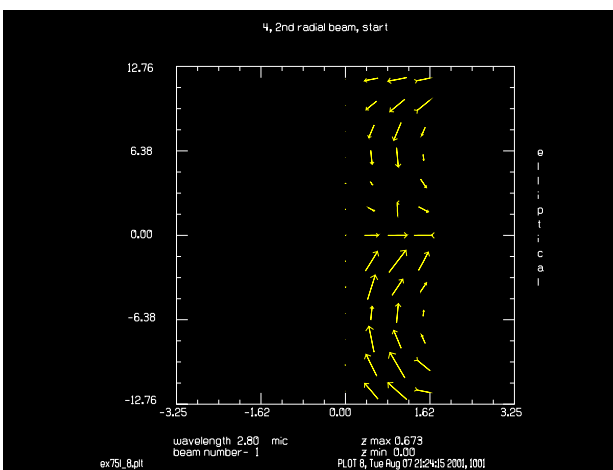


Fig. 75l.8. Point 4. Radial-azimuth polarization after 2nd outer cone. Inward radial mode.

- c A final pair of right cones forms a reflexicon configuration which
- c recompacks the beam.
- c
- c An internal focus in the radial path flips the irradiance distribution
- c so the final right axicon recompacks the beam with flat irradiance
- c and well-behaved polarization.
- c
- c Optical power and aberrations may be added by using MIRROR/GLOBAL
- c in radial mode. The aberrations are computed only along the x-direction
- c and applied in rotationally symmetric fashion for array attribute
- c of beam and projected in the y-direction for radial and annular modes.
- c
- c Aberration must be in the form of ordinary even powers of radius
- c and must be centered about the axis. The axis is
- c on the outside of the annular path, so the mirrors are centered
- c on the top of the annular path.

Jump to: [Commands](#), [Theory](#)

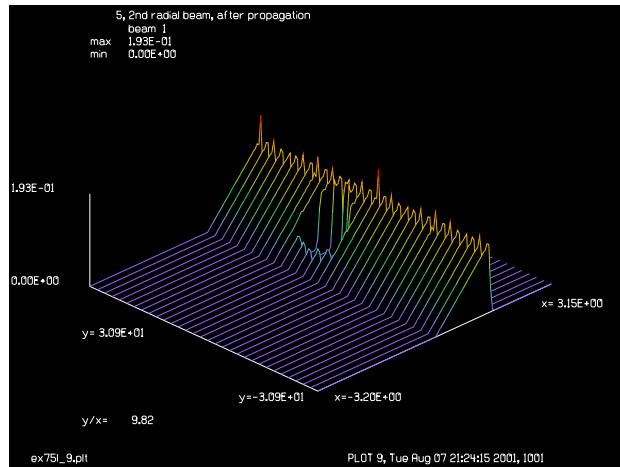


Fig. 75l.9. Point 5. Radial-azimuth intensity after 3rd outer cone. Note flip in azimuth direction.

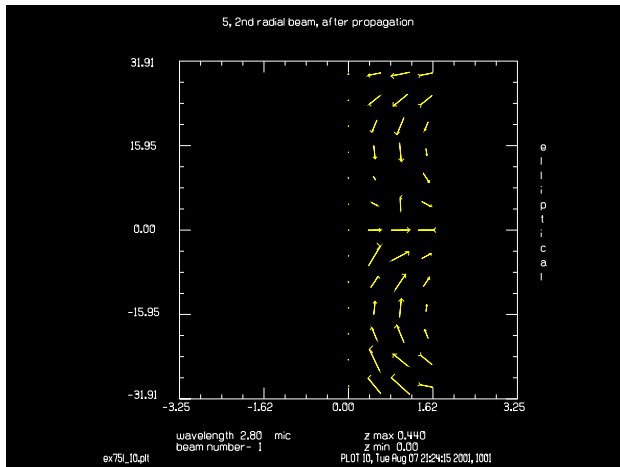


Fig. 75l.10. Point 5. Radial-azimuth polarization after 3rd outer cone. Note flip in azimuth direction.

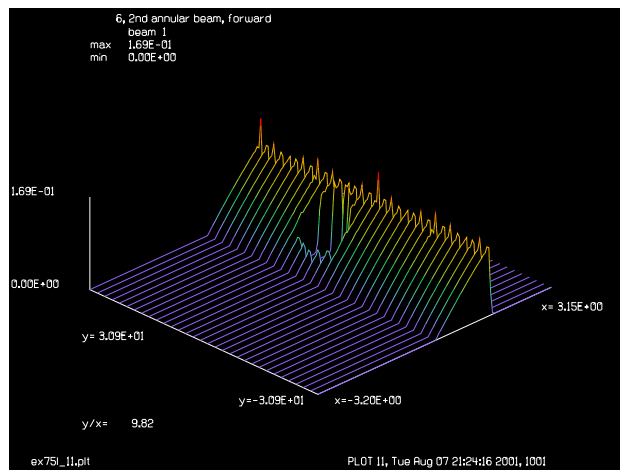


Fig. 75l.11. Point 6. Radial-azimuth intensity after 4th outer cone.

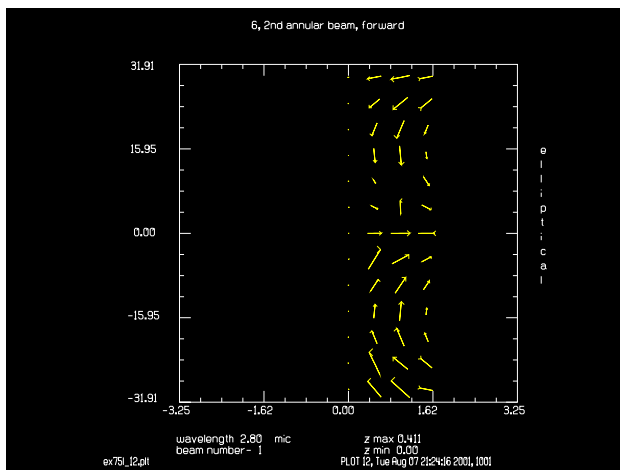


Fig. 75l.12. Point 6. Radial-azimuth polarization after 4th outer cone.

c
c A decentered obscuration is introduced to display the beam orientation
c at the various positions.

```
c
variab/dec/int EXACT READ_COEFF
EXACT = 0      # 0, use guess for image inversion mirrors
READ_COEFF = 0 # 0, do not read coefficients from table
Apt_inner = 2  # radius of inner axicon
Apt_outer = 4  # radius of outer axicon
Length1 = 2    # length of resonator
Length2 = 4    # end mirror to outcoupling axicon
set/density 12 12
echo/on
array/s 1 128 128 1
units/set 1 .05 .05
wavelength/set 1 2.8
```

Jump to: [Commands](#), [Theory](#)

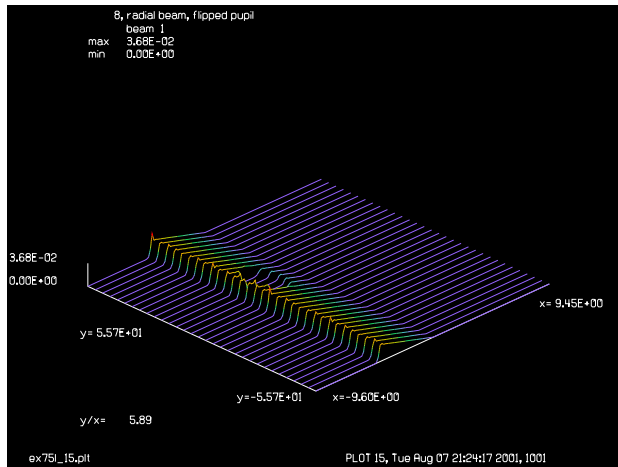


Fig. 75l.13. Point 7. Radial-azimuth intensity after 4th inversion optics. Note flip of intensity.

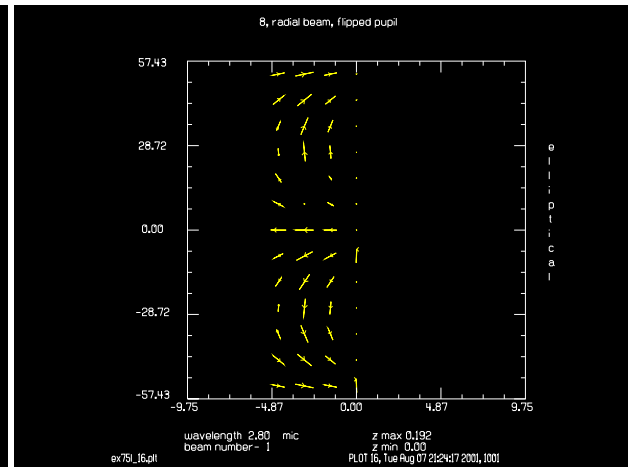


Fig. 75l.14. Point 7. Radial-azimuth polarization after 4th inversion optics. Note flip of intensity.

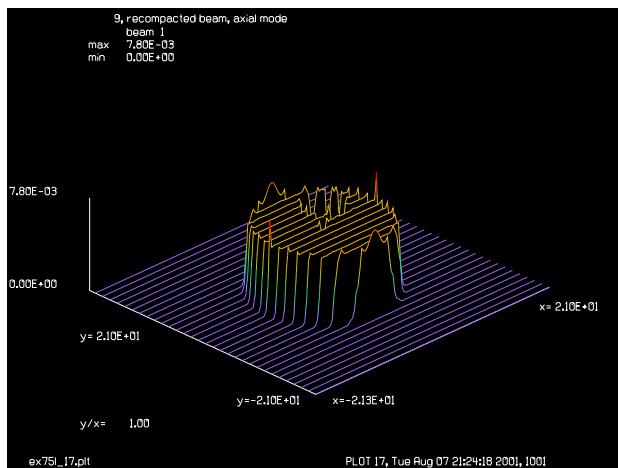


Fig. 75l.15. Point 8. Reformed pupil in collimated beam, shows offset obscuration.

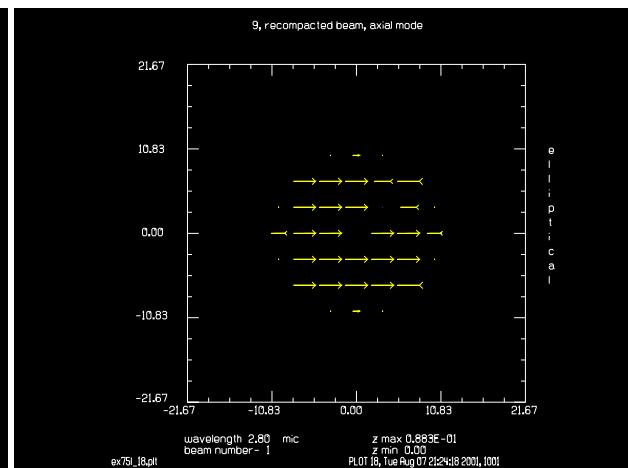


Fig. 75l.16. Point 8. Reformed pupil in collimated beam, shows polarization preserved.

```
clap/c/c 1 Apt_inner
obs 1 .001
energy/norm 1
clear 1 1
clap/c/c 1 1.5
obs/c 1 .4 xdec = .6 ydec=.6
title 1, starting axial beam
plot/watch ex75l_1.plt
plot/l 1
plot/watch ex75l_2.plt
plot/e 1
dist 10
B0=0.
B1=0.
B2=1e-10. # should not be exactly zero
B4=0.
```

Jump to: [Commands](#), [Theory](#)

```
B6=0.
B8=0.
c Propagate to first axicon, WIC
vertex/rotate/set 0 -45 0
if READ_COEFF read/disk wic.out
Rad = 1./2/B2
mirror/global/conic Rad cc=-1 a4=B4 a6=B6 a8=B8 exact/polynomials/radial
axicon/radial/left/focal 1 Apt_inner nomirror      # axial to radial
title 2, expanding radial path
plot/watch ex75l_3.plt
plot/l 1
plot/watch ex75l_4.plt
plot/e 1
global
c Propagate to outer edge of waxicon, WOC
vertex/locate/abs Apt_outer 0 0
vertex/rotate/set 0 45 0
if READ_COEFF read/disk woc.out
Rad = 1./2/B2
mirror/global/conic Rad cc=-1 a4=B4 a6=B6 a8=B8 exact/polynomials/radial
axicon/annular/right 1 nomirror      # radial to annular
global
title 3, 1st annular beam, back propagating
plot/watch ex75l_5.plt                # intensity in annular beam
plot/l 1
plot/watch ex75l_6.plt                # intensity in annular beam
plot/e 1
c Propagate to rear cone, RC
vertex/locate/abs Apt_outer 0 -Length1
vertex/rotate/set 0 135 0
if READ_COEFF read/disk rc.out
Rad = 1./2/B2
mirror/global/conic Rad cc=-1 a4=B4 a6=B6 a8=B8 exact/polynomials/radial
axicon/radial/left 1 nomirror      # 2nd outer axicon reflection
title 4, 2nd radial beam, start
plot/watch ex75l_7.plt                # intensity in annular beam
plot/l 1
plot/watch ex75l_8.plt                # intensity in annular beam
plot/e 1
c Propagate to other side of rear cone, RC
vertex/locate/abs -Apt_inner 0 -Length1
vertex/rotate/set 0 -135 0
Rad = 1./2/B2
mirror/global/conic/radial Rad cc=-1 a4=B4 a6=B6 a8=B8 exact/polynomials/radial
title 5, 2nd radial beam, after propagation
plot/watch ex75l_9.plt                # intensity in annular beam
plot/l 1
plot/watch ex75l_10.plt              # intensity in annular beam
plot/e 1
axicon/annular/left 1 nomirror
title 6, 2nd annular beam, forward
plot/watch ex75l_11.plt              # intensity in annular beam
plot/l 1
plot/watch ex75l_12.plt              # intensity in annular beam
```

Jump to: [Commands](#), [Theory](#)

```

plot/e 1
global
c Propagate to outer reflaxicon, ROC
vertex/locate/abs -Apt_inner 0 Length2
vertex/rotate/set 0 -45 0
if EXACT then
    if READ_COEFF read/disk roc.out
    Rad = 1./2/B2
    mirror/global/conic/radial Rad cc=-1 a4=B4 a6=B6 a8=B8 exact/polynomials/radi
else
    Rad=-Apt_outer
    Rad_eff = Rad/cos(pi*45/180.)
    mirror/global/conic Rad_eff astigmatic
endif
axicon/radial/left 1 nomirror
global
title 7, radial beam, start
plot/watch ex75l_13.plt          # intensity in annular beam
plot/l 1
plot/watch ex75l_14.plt          # intensity in annular beam
plot/e 1
global
c Inner reflaxicon, RIC
c Propagate to axis
vertex/locate/abs 0 0 4
vertex/rotate/set 0 135 0
if EXACT then
    if READ_COEFF read/disk ric.out
    Rad = 1./2/B2
    mirror/global/conic/radial Rad cc=-1 a4=B4 a6=B6 a8=B8 exact/polynomials/radi
else
    prop 2*Apt_outer # propagate just to make a plot
    title 8, radial beam, flipped pupil
    plot/watch ex75l_15.plt          # intensity in annular beam
    plot/l 1 h=.1
    plot/watch ex75l_16.plt          # intensity in annular beam
    plot/e 1
    prop -2*Apt_outer
    Rad=-Apt_outer
    Rad_eff = Rad/cos(pi*45/180.)
    mirror/global/conic/radial Rad_eff astigmatic
endif
c Propagate back to edge of aperture
prop -Apt_inner
axicon/axial/right 1 -Apt_inner nomirror
title mirror option, at z=-2, after axicon
prop Apt_inner
title 9, recompact beam, axial mode
plot/watch ex75l_17.plt          # intensity in annular beam
plot/l 1
plot/watch ex75l_18.plt          # intensity in annular beam
plot/e 1
global
end

```

Jump to: [Commands](#), [Theory](#)

Input: WIC

Input: wic.inp

```

c## wic.inp
# x, z are in the x-z plane
# z = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
#       + C5*(x-x0)^4 + C6*(x-x0)^5
#
# here are the coefficients
C1 = 0.
C2 = 1.34650191
C3 = -8.80673268e-3
C4 = -2.461295e-5
C5 = -1.9268059e-7
C6 = 0.
Apt = 5.79
x0 = 0.
Xdec = 0.
# variables are x0, C1, C2, C3, C4, C5, C6 (seven variables)
#
# t, v are the coordinates in the vertex coordinate system
#
# v = B0 + B1*t + .5*t^2/Rad + B4*t^4 + B6*t^6 + B8*t^8
#
# A = angle
# v = x*cos(A)
# z1 is the surface calculated with the vertex polynomial
#
# z1 = t*sin(A) + v*cos(A)
#
# abr/tilt Kbeam [-2*B1/Lambda] rn=1
# mirror/global/conic/radial Kbeam [1./2./B2] cc=0. a4=B4 a6=B6 a8=B8
#
Angle = atan(C2)
A = Angle
variab/dec/int count
macro/def step/o
    count = count + 1
    row = .1*(count-1)
    x = x0 + row*Apt
    z_x = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
    z_x = z_x + C5*(x-x0)^4 + C6*(x-x0)^5
    t = x*cos(A) + z_x*sin(A)
    v = B2*t^2 + B4*t^4 + B6*t^6 + B8*t^8
    z_t = t*sin(A) + v*cos(A)
    dZ@count = z_x - z_t
    udata/set count row z_x z_t dZ@count
macro/end
macro/def system/o
    count = 0
    macro step/11
macro/end

```

Jump to: [Commands](#), [Theory](#)

```

B0 = 0.
B1 = 0.
B2 = 0.
B4 = 0.
B6 = 0.
B8 = 0.

macro/run system
opt/tar/add dZ1
opt/tar/add dZ2
opt/tar/add dZ3
opt/tar/add dZ4
opt/tar/add dZ5
opt/tar/add dZ6
opt/tar/add dZ7
opt/tar/add dZ8
opt/tar/add dZ9
opt/tar/add dZ10
opt/tar/add dZ11
opt/name system
variab/dec/int Ntimes
Ntimes = 4

opt/var/add B2
opt/run Ntimes
plot/w plot1.plt
title B2
plot/udata 3

opt/var/add B4
opt/run Ntimes
plot/w plot2.plt
title B4
plot/udata 3 min=-1e-3 max=1e-3

opt/var/add B6
opt/run Ntimes
plot/w plot3.plt
title B6
plot/udata 3 min=-1e-3 max=1e-3

opt/var/add B8
opt/run Ntimes
plot/w plot4.plt
title B8
plot/udata 3 min=-1e-3 max=1e-3

write/d wic.out/o
B0=
B1=
B2=
B4=
B6=
B8=

```

Jump to: [Commands](#), [Theory](#)

write/tty

Input: WOC

Input: woc.inp

```

c## woc!255347614281518
# x, z are in the x-z plane
# z = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
#       + C5*(x-x0)^4 + C6*(x-x0)^5
#
# here are the coefficients
C1 = -6.3188727e-1
C2 = -9.01395217e-1
C3 = 3.38595748e-2
C4 = -1.86880569e-3
C5 = 7.2953038e-5
C6 = 0.
Apt = 3.0261941
x0 = 55.1498897
Xdec = x0 + Apt
# variables are x0, C1, C2, C3, C4, C5, C6 (seven variables)
#
# t, v are the coordinates in the vertex coordinate system
#
# v = B0 + B1*t + .5*t^2/Rad + B4*t^4 + B6*t^6 + B8*t^8
#
# A = angle
# v = x*cos(A)
# z1 is the surface calculated with the vertex polynomial
#
# z1 = t*sin(A) + v*cos(A)
#
# abr/tilt Kbeam [-2*B1/Lambda] rn=1
# mirror/global/conic/radial Kbeam [1./2./B2] cc=0. a4=B4 a6=B6 a8=B8
#
Angle = atan(C2)
A = Angle
x = Xdec
z_x = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
z_x = z_x + C5*(x-x0)^4 + C6*(x-x0)^5
Z0 = z_x
variab/dec/int count
macro/def step/o
    count = count + 1
    row = .1*(count-1)
    x = x0 + row*Apt
    z_x = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
    z_x = z_x + C5*(x-x0)^4 + C6*(x-x0)^5
    t = (x-Xdec)*cos(A) + (z_x-Z0)*sin(A)
    v = B1*t + B2*t^2 + B4*t^4 + B6*t^6 + B8*t^8
    z_t = t*sin(A) + v*cos(A) + Z0
    dZ@count = z_x - z_t

```

Jump to: [Commands](#), [Theory](#)


```

    udata/set count row z_x z_t dZ@count
macro/end
macro/def system/o
    count = 0
    macro step/11
macro/end
B0 = 0.
B1 = 0.
B2 = 0.
B4 = 0.
B6 = 0.
B8 = 0.

macro/run system
opt/tar/add dZ1
opt/tar/add dZ2
opt/tar/add dZ3
opt/tar/add dZ4
opt/tar/add dZ5
opt/tar/add dZ6
opt/tar/add dZ7
opt/tar/add dZ8
opt/tar/add dZ9
opt/tar/add dZ10
opt/tar/add dZ11
opt/name system
variab/dec/int Ntimes
Ntimes = 6

opt/var/add B1
opt/var/add B2
opt/run Ntimes
plot/w plot1.plt
title B2
plot/udata 3
c read/screen

opt/var/add B4
opt/run Ntimes
plot/w plot2.plt
title B4
plot/udata 3
c read/screen

opt/var/add B6
opt/run Ntimes
plot/w plot3.plt
title B6
plot/udata 3
c read/screen

opt/var/add B8
opt/run Ntimes
plot/w plot4.plt

```

Jump to: [Commands](#), [Theory](#)

```

title B8
plot/udata 3
c read/screen

write/d woc.out/o
B0=
B1=
B2=
B4=
B6=
B8=
write/screen

```

Input: RC

Input: rc.inp

```

c## rc!255347614259582
# x, z are in the x-z plane
# z = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
#       + C5*(x-x0)^4 + C6*(x-x0)^5
#
# here are the coefficients
C1 = -2.54e-5
C2 = -9.98504522e-1
C3 = 4.22754657e-4
C4 = 3.6236338e-7
C5 = 7.2953038e-5
C6 = 0.
Apt = 5.7097150
x0 = 55.1498897
Xdec = x0 + Apt
# variables are x0, C1, C2, C3, C4, C5, C6 (seven variables)
#
# t, v are the coordinates in the vertex coordinate system
#
# v = B0 + B1*t + .5*t^2/Rad + B4*t^4 + B6*t^6 + B8*t^8
#
# A = angle
# v = x*cos(A)
# z1 is the surface calculated with the vertex polynomial
#
# z1 = t*sin(A) + v*cos(A)
#
# abr/tilt Kbeam [-2*B1/Lambda] rn=1
# mirror/global/conic/radial Kbeam [1./2./B2] cc=0. a4=B4 a6=B6 a8=B8
#
Angle = atan(C2)
A = Angle
x = Xdec
z_x = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
z_x = z_x + C5*(x-x0)^4 + C6*(x-x0)^5

```

Jump to: [Commands](#), [Theory](#)

```

Z0 = z_x
variab/dec/int count
macro/def step/o
    count = count + 1
    row = .1*(count-1)
    x = x0 + row*Apt
    z_x = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
    z_x = z_x + C5*(x-x0)^4 + C6*(x-x0)^5
    t = (x-Xdec)*cos(A) + (z_x-Z0)*sin(A)
    v = B1*t + B2*t^2 + B4*t^4 + B6*t^6 + B8*t^8
    z_t = t*sin(A) + v*cos(A) + Z0
    dZ@count = z_x - z_t
    udata/set count row z_x z_t dZ@count
macro/end
macro/def system/o
    count = 0
    macro step/11
macro/end
B0 = 0.
B1 = 0.
B2 = 0.
B4 = 0.
B6 = 0.
B8 = 0.

macro/run system
opt/tar/add dZ1
opt/tar/add dZ2
opt/tar/add dZ3
opt/tar/add dZ4
opt/tar/add dZ5
opt/tar/add dZ6
opt/tar/add dZ7
opt/tar/add dZ8
opt/tar/add dZ9
opt/tar/add dZ10
opt/tar/add dZ11
opt/name system
variab/dec/int Ntimes
Ntimes = 6

opt/var/add B1
opt/var/add B2
opt/run Ntimes
plot/w plot1.plt
title B2
plot/udata 3
c read/screen

opt/var/add B4
opt/run Ntimes
plot/w plot2.plt
title B4
plot/udata 3

```

Jump to: [Commands](#), [Theory](#)

```

c read/screen

opt/var/add B6
opt/run Ntimes
plot/w plot3.plt
title B6
plot/udata 3
c read/screen

write/d rc.out/o
B0=
B1=
B2=
B4=
B6=
B8=
write/screen

```

Input: RIC

Input: ric.inp

```

c## ric!25534761423331
# x, z are in the x-z plane
# z = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
#       + C5*(x-x0)^4 + C6*(x-x0)^5
#
# here are the coefficients
C1 = 1.e-8
C2 = -1.339678521
C3 = 1.36223789e-2
C4 = -6.8025241e-8
C5 = 0.
C6 = 0.
Apt = 13.825220
x0 = 0.
Xdec = 0.
# variables are x0, C1, C2, C3, C4, C5, C6 (seven variables)
#
# t, v are the coordinates in the vertex coordinate system
#
# v = B0 + B1*t + .5*t^2/Rad + B4*t^4 + B6*t^6 + B8*t^8
#
# A = angle
# v = x*cos(A)
# z1 is the surface calculated with the vertex polynomial
#
# z1 = t*sin(A) + v*cos(A)
#
# abr/tilt Kbeam [-2*B1/Lambda] rn=1
# mirror/global/conic/radial Kbeam [1./2./B2] cc=0. a4=B4 a6=B6 a8=B8
#
Angle = atan(C2)

```

Jump to: [Commands](#), [Theory](#)

```

A = Angle
variab/dec/int count
macro/def step/o
    count = count + 1
    row = .05*(count-1)
    x = x0 + row*Apt
    z_x = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
    z_x = z_x + C5*(x-x0)^4 + C6*(x-x0)^5
    t = x*cos(A) + z_x*sin(A)
    v = B2*t^2 + B4*t^4 + B6*t^6 + B8*t^8
    z_t = t*sin(A) + v*cos(A)
    dZ@count = z_x - z_t
    udata/set count row z_x z_t dZ@count
macro/end
macro/def system/o
    count = 0
    macro step/21
macro/end
B0 = 0.
B1 = 0.
B2 = 0.
B4 = 0.
B6 = 0.
B8 = 0.

macro/run system
opt/tar/add dZ1
opt/tar/add dZ2
opt/tar/add dZ3
opt/tar/add dZ4
opt/tar/add dZ5
opt/tar/add dZ6
opt/tar/add dZ7
opt/tar/add dZ8
opt/tar/add dZ9
opt/tar/add dZ10
opt/tar/add dZ11
opt/tar/add dZ12
opt/tar/add dZ13
opt/tar/add dZ14
opt/tar/add dZ15
opt/tar/add dZ16
opt/tar/add dZ17
opt/tar/add dZ18
opt/tar/add dZ19
opt/tar/add dZ20
opt/tar/add dZ21
opt/name system
variab/dec/int Ntimes
Ntimes = 4

opt/var/add B2
opt/run Ntimes
plot/w plot1.plt

```

Jump to: [Commands](#), [Theory](#)

```

title B2
plot/udata 3

opt/var/add B4
opt/run Ntimes
plot/w plot2.plt
title B4
plot/udata 3 min=-1e-3 max=1e-3

opt/var/add B6
opt/run Ntimes
plot/w plot3.plt
title B6
plot/udata 3 min=-1e-3 max=1e-3

write/d ric.out/o
B0=
B1=
B2=
B4=
B6=
B8=
write/screen

```

Input: ROC

Input: roc.inp

```

c## roc
#
# x, z are in the x-z plane
#  $z = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3$ 
#  $+ C5*(x-x0)^4 + C6*(x-x0)^5$ 
#
# here are the coefficients
C1 = 5.182e-5
C2 = -1.01709225
C3 = -1.30609608e-1
C4 = 2.49164254e-4
C5 = 2.2300356e-5
C6 = 5.123966e-7
Apt = 1.4499971
x0 = 55.1498897
Xdec = x0 + Apt
# variables are x0, C1, C2, C3, C4, C5, C6 (seven variables)
#
# t, v are the coordinates in the vertex coordinate system
#
#  $v = B0 + B1*t + .5*t^2/Rad + B4*t^4 + B6*t^6 + B8*t^8$ 
#
# A = angle
#  $v = x*\cos(A)$ 

```

Jump to: [Commands](#), [Theory](#)

```

# z1 is the surface calculated with the vertex polynomial
#
# z1 = t*sin(A) + v*cos(A)
#
# abr/tilt Kbeam [-2*B1/Lambda] rn=1
# mirror/global/conic/radial Kbeam [1./2./B2] cc=0. a4=B4 a6=B6 a8=B8
#
Angle = atan(C2)
A = Angle
x = Xdec
z_x = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
z_x = z_x + C5*(x-x0)^4 + C6*(x-x0)^5
Z0 = z_x
variab/dec/int count
macro/def step/o
    count = count + 1
    row = .1*(count-1)
    x = x0 + row*Apt
    z_x = C1 + C2*(x-x0) + C3*(x-x0)^2 + C4*(x-x0)^3
    z_x = z_x + C5*(x-x0)^4 + C6*(x-x0)^5
    t = (x-Xdec)*cos(A) + (z_x-Z0)*sin(A)
    v = B1*t + B2*t^2 + B4*t^4 + B6*t^6 + B8*t^8
    z_t = t*sin(A) + v*cos(A) + Z0
    dZ@count = z_x - z_t
    udata/set count row z_x z_t dZ@count
macro/end
macro/def system/o
    count = 0
    macro step/11
macro/end
B0 = 0.
B1 = 0.
B2 = 0.
B4 = 0.
B6 = 0.
B8 = 0.

macro/run system
opt/tar/add dZ1
opt/tar/add dZ2
opt/tar/add dZ3
opt/tar/add dZ4
opt/tar/add dZ5
opt/tar/add dZ6
opt/tar/add dZ7
opt/tar/add dZ8
opt/tar/add dZ9
opt/tar/add dZ10
opt/tar/add dZ11
opt/name system
variab/dec/int Ntimes
Ntimes = 6

opt/var/add B1

```

Jump to: [Commands](#), [Theory](#)

```
opt/var/add B2
opt/run Ntimes
plot/w plot1.plt
title B2
plot/udata 3
c read/tty
```

```
opt/var/add B4
opt/run Ntimes
plot/w plot2.plt
title B4
plot/udata 3
c read/tty
```

```
opt/var/add B6
opt/run Ntimes
plot/w plot3.plt
title B6
plot/udata 3
c read/tty
```

```
opt/var/add B8
opt/run Ntimes
plot/w plot4.plt
title B8
plot/udata 3
c read/tty
```

```
write/d roc.out/o
B0=
B1=
B2=
B4=
B6=
B8=
write/tty
```


Ex76: Stable resonator with coherent injection

Table. 76.1. Table of Ex76 examples

Ex76a: Example of coherent injection, bare cavity analysis	2
Ex76b: Example of coherent injection, automatic frequency control	7

This examples illustrates coherent injection of a stable resonator. The resonator is shown schematically in Figs. 76.1 and 76.2. Two mirrors of radius 75 cm are spaced 149 cm apart. The resonator is marginally stable: if the spacing were 1 cm longer at 150 cm, it would be concentric and not stable. The device, is therefore, prone to resonate in off-axis modes. As will be shown, the apertures of 0.3 cm radius are so large that they allow multi-mode performance. Consequently, the bare-cavity performance shows very slow convergence to the lowest loss mode. However, with coherent injection of a beam matched to the lowest-loss mode, the injected signal will induce a high degree of mode selectivity, leading to very stable performance.

It is assumed that the coherent injection is exactly phase matched to the cavity mode. In an experiment, it takes a sophisticated detection and control system to exactly match the input beam to the phase of the cavity mode. This implies both frequency and phase matching. In this numerical model, the process is treated in an idealized manner. The complex correlation of the input beam (before injection) with the cavity mode is calculated and the phase of the input beam is corrected exactly match the phase of the cavity mode. With frequency correction but without phase correction, the cavity mode would not build to the maximum possible value. Without frequency correction, a beat frequency would develop due to the mismatch of frequencies. These effects may be studied, if desired, by making modifications to the command decks used here.

A second consideration is that there is a hole in the left mirror of approximately 150 micron radius. An obscuration of approximately this size may cause the TEM00 mode to cease to be lowest loss mode.

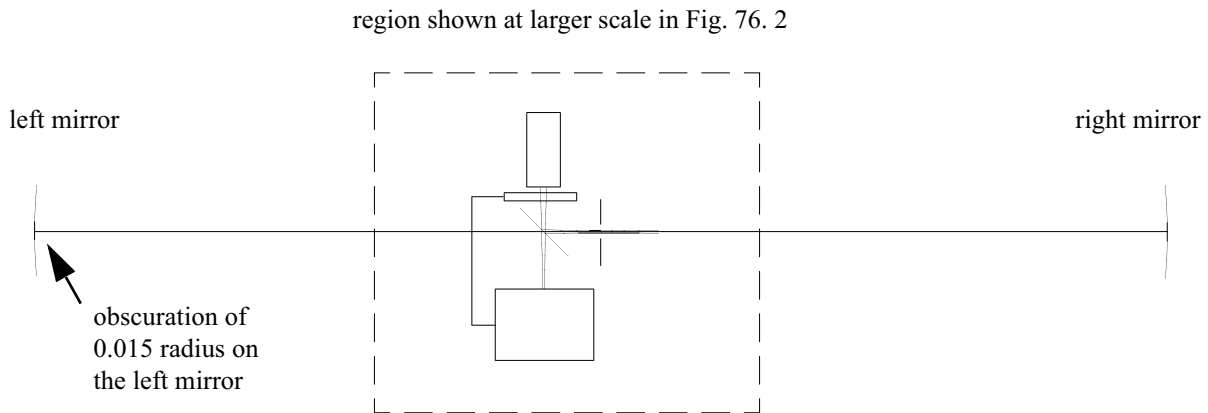


Fig. 76.1. Schematic of coherent injection experiment. Two spherical mirrors of radius 75 cm are separated by 149 cm. Because of symmetry the waist lies in the middle. A beam is injected in to the resonator by a beam combiner. A control system measures and corrects the phase of the input signal.

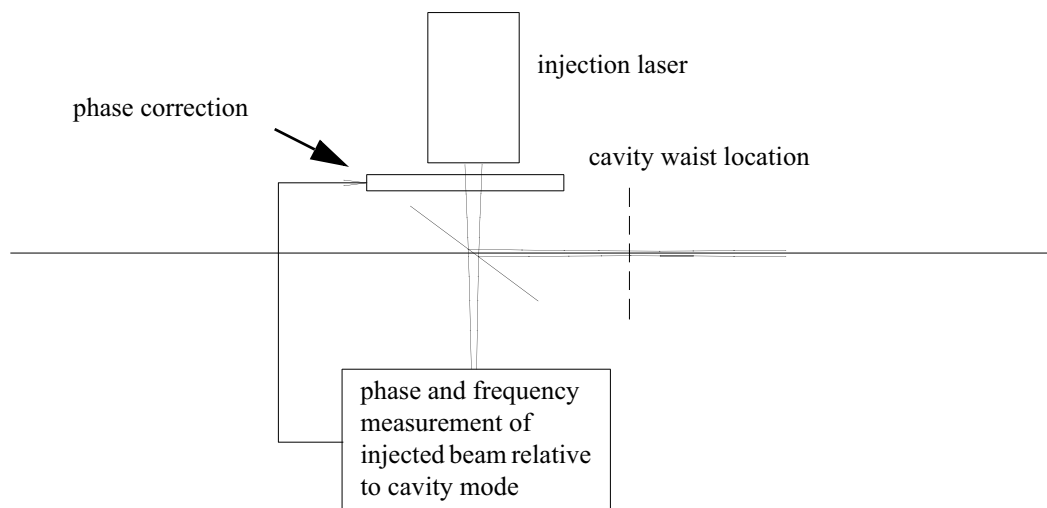


Fig. 76.2. Schematic of coherent injection experiment. The injection signal should be spatially and frequency mode-matched to the cavity mode. The injected beam is designed to put its gaussian waist at exactly the same position as the cavity waist and to have the same size. The optical frequency and phase must also be matched to obtain maximum cavity power. The control system is simulated by measuring the relative phase between injected and cavity modes by calculating the correlation between the beams and then using the phase of the complex correlation coefficient to correct the injected signal.

Ex76a: Example of coherent injection, bare cavity analysis

To investigate this effect, a bare-cavity mode analysis was conducted with command deck `ex87a.inp`. This example used a range of obscuration radii from 0.0 to 600 microns in steps of 150 microns. For each obscuration value, the resonator was started from noise and allowed 200 passes to stabilize. The loss per pass and mode convergence per pass is plotted. The loss per pass drops quickly to a low value as the higher spatial frequencies are eliminated by the clear apertures. The mode convergence was measured by calculating the correlation of the current mode with the mode at the 4th previous pass. It is interesting to note that, although the beam distribution changes greatly over several passes, the convergence criteria based on correlation increases monotonically.

Figures 76.3a through 76.7b illustrate the distribution after 200 cycles for the obscuration values 0.0 through 600 micron radius, in steps of 150 microns. The (a) figures show the left mirror and the (b) figures show the right mirror at the end of 200 cycles of stabilization. At each value of obscuration, the resonator analysis was restarted with the same noise distribution. It is quite interesting to observe the gradual increase in speckle size as the high spatial frequencies are filtered out. For obscuration radii of 0.0 to through 300, the primary modes are TEM00 and a combination of TEM01 and TEM10 (the donut mode combination). From 450 to 600 microns, the TEM00 mode is not evident and there are various high order modes as well as the TEM01 and TEM10 modes which cause a side-to-side thrashing.

None of the distributions are stabilized, even after 200 cycles, and all show considerable variation of mode shape every few cycles as the modes beat together. Figure 8 shows the history of round-trip loss and convergence (by our particular means of measurement) over the full 1000 passes covering the 5 choices of obscuration radius. The convergence plots (dashed) show that while the loss becomes quite low, there are

several modes left in the resonator. If the device were to be used as a conventional laser, it would be wise to reduce the clear aperture radius from 0.3 cm to some appreciable smaller value. However, in an injected device, the injected mode is a powerful force to control the transverse mode structure.

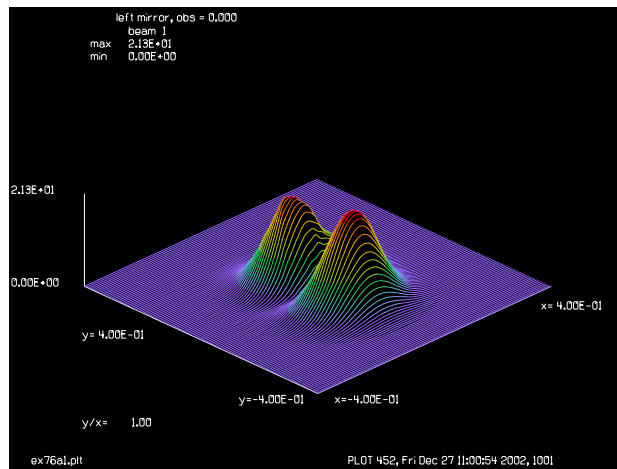


Fig. 76.3a. Left mirror. No obscuration. Modes are TEM00, TEM10 and TEM01.

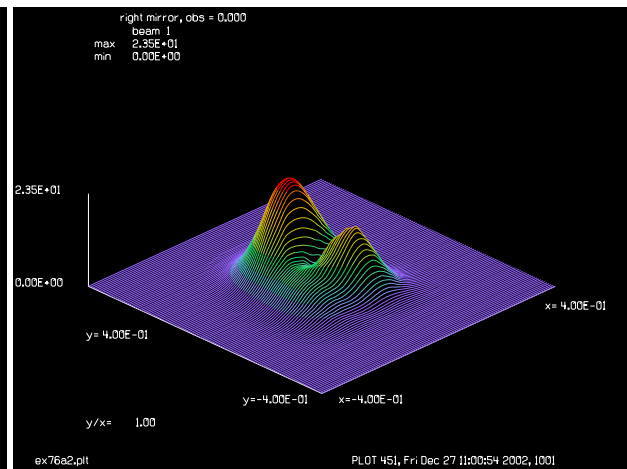


Fig. 76.3b. Right mirror. No obscuration. Modes are TEM00, TEM10 and TEM01.

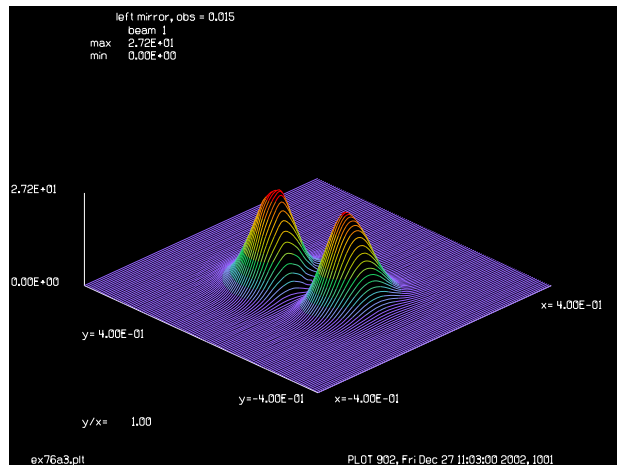


Fig. 76.4a. Left mirror. Obscuration 0.015. Modes are primarily TEM01 and TEM10.

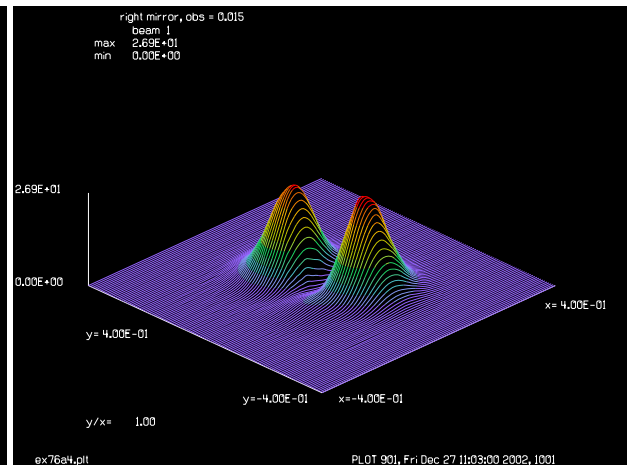


Fig. 76.4b. Right mirror. Obscuration 0.015. Modes are primarily TEM01 and TEM10.

Input: `ex76a.inp`

```
c## ex76a
c
c Example 76a: example of coherent injection, bare cavity analysis
c
c This example illustrated a bare cavity analysis of the cavity
c to be used for a coherent injection experiment as described in
c Example 87b.
c
c variables
```

Jump to: [Commands](#), [Theory](#)

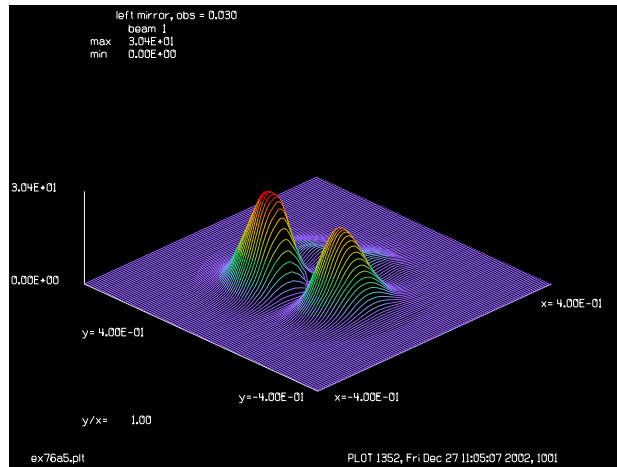


Fig. 76.5a. Left mirror. Obscuration 0.030. Modes TEM01, TEM10 and higher.

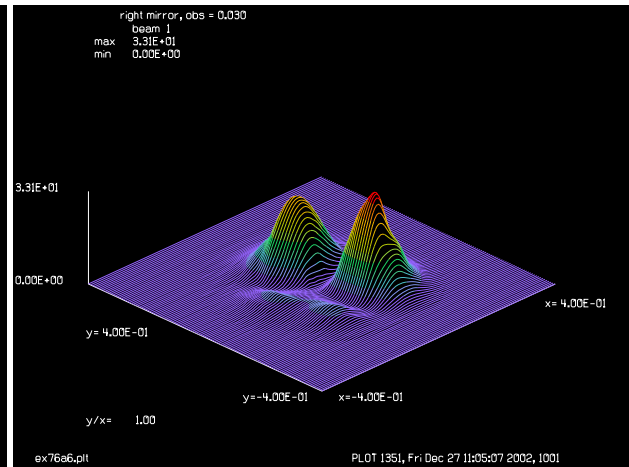


Fig. 76.5b. Right mirror. Obscuration 0.030. Modes TEM01, TEM10 and higher.

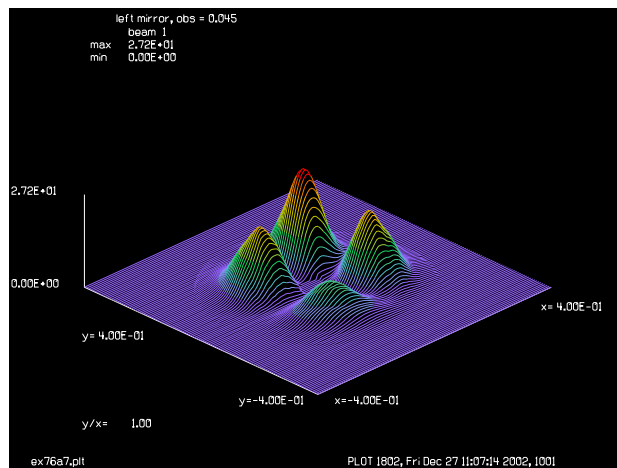


Fig. 76.6a. Left mirror. Obscuration 0.045. Modes TEM01, TEM10 and higher order modes dominate.

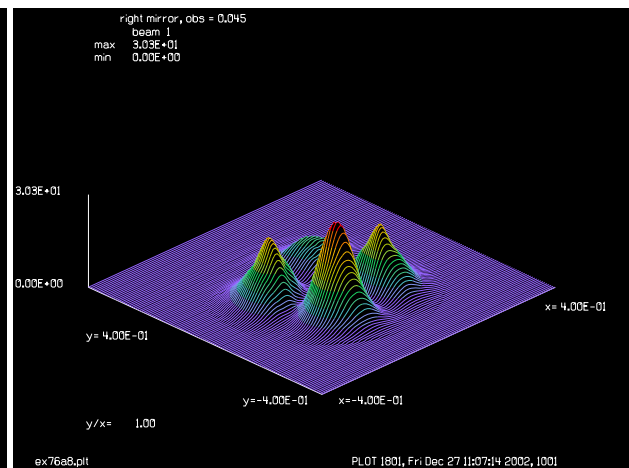


Fig. 76.6b. Right mirror. Obscuration 0.045. Modes TEM01, TEM10 and higher order modes dominate.

```

c -----
c power          - power in beam after each pass
c units          - units of the beam
c abscorr        - modulus of correlation of beam with 4 th pass
c pass           - pass number
c Ntimes         - number of passes to stabilize
c clap           - clear aperture radius
c obs            - obscuration radius
c set_obs        - logical switch to activate obscuration
c display        - counts 1 to 4 and displays plots on 4
c point          - number of point in user data
c plot_num       - plot number
c obs_store      - number of plot1.plt, plot2.plt, or plot3.plt
c
variab/dec/int pass Ntimes set_obs step display point plot_num obs_store
variab/dec/int plot_num1

```

Jump to: [Commands](#), [Theory](#)

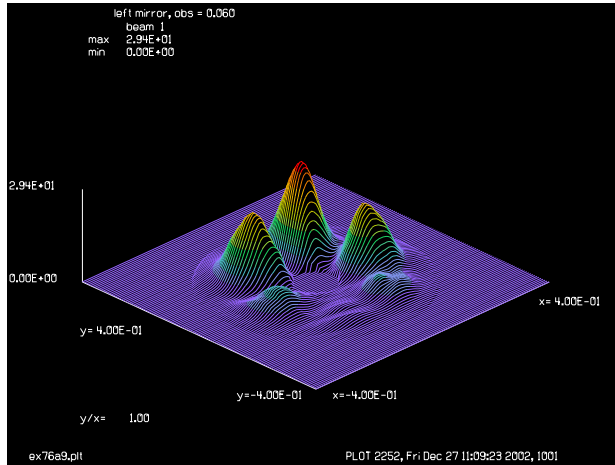


Fig. 76.7a. Left mirror. Obscuration 0.060. Modes TEM01, TEM10 and higher order modes dominate.

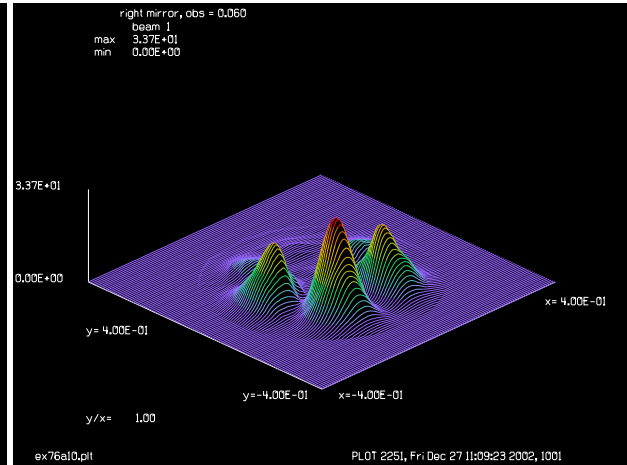


Fig. 76.7b. Right mirror. Obscuration 0.060. Modes TEM01, TEM10 and higher order modes dominate.

```

c
c bare cavity analysis
c
macro/def reson/o
  plot_num = obs_store*2
  pass = pass+1
  display = display+1
  prop 83
  mirror/sph 1 -75
  clap/c/n 1 clap
  title right mirror, obs = @obs
  plot_num1 = plot_num + 2
  plot/watch ex76a_@plot_num1.plt
  plot/l 1 ns=128 xrad=.4
  prop 149
  mirror/sph 1 75
  clap/c/n 1 clap
  if set_obs = 1 obs 1 obs
  if display = 4 then
    mult/mode/corr 1 2
    copy/con 1 2
  endif
  variab/set abscorr abscorr
  title left mirror, obs = @obs
  plot_num1 = plot_num + 1
  plot/watch ex76a_@plot_num1.plt
  plot/l 1 ns=128 xrad=.4
  prop 66
  if display = 4 then
    point = point+1
    variab/set power 1 energy
    udata/set point pass 1-power abscorr
    title loss (solid) and convergence (dash) per pass
    plot/watch ex76a_11.plt
  endif
endmacro

```

increment pass counter
increment display counter
propagate to next mirror
right mirror, 75 cm. radius
.3 cm. radius aperture

propagate 149 cm. along beam
left mirror, 75 cm. radius
.3 cm. radius aperture
obscuration radius

propagate to waist

calculate the current power
store the loss and convergence

Jump to: [Commands](#), [Theory](#)

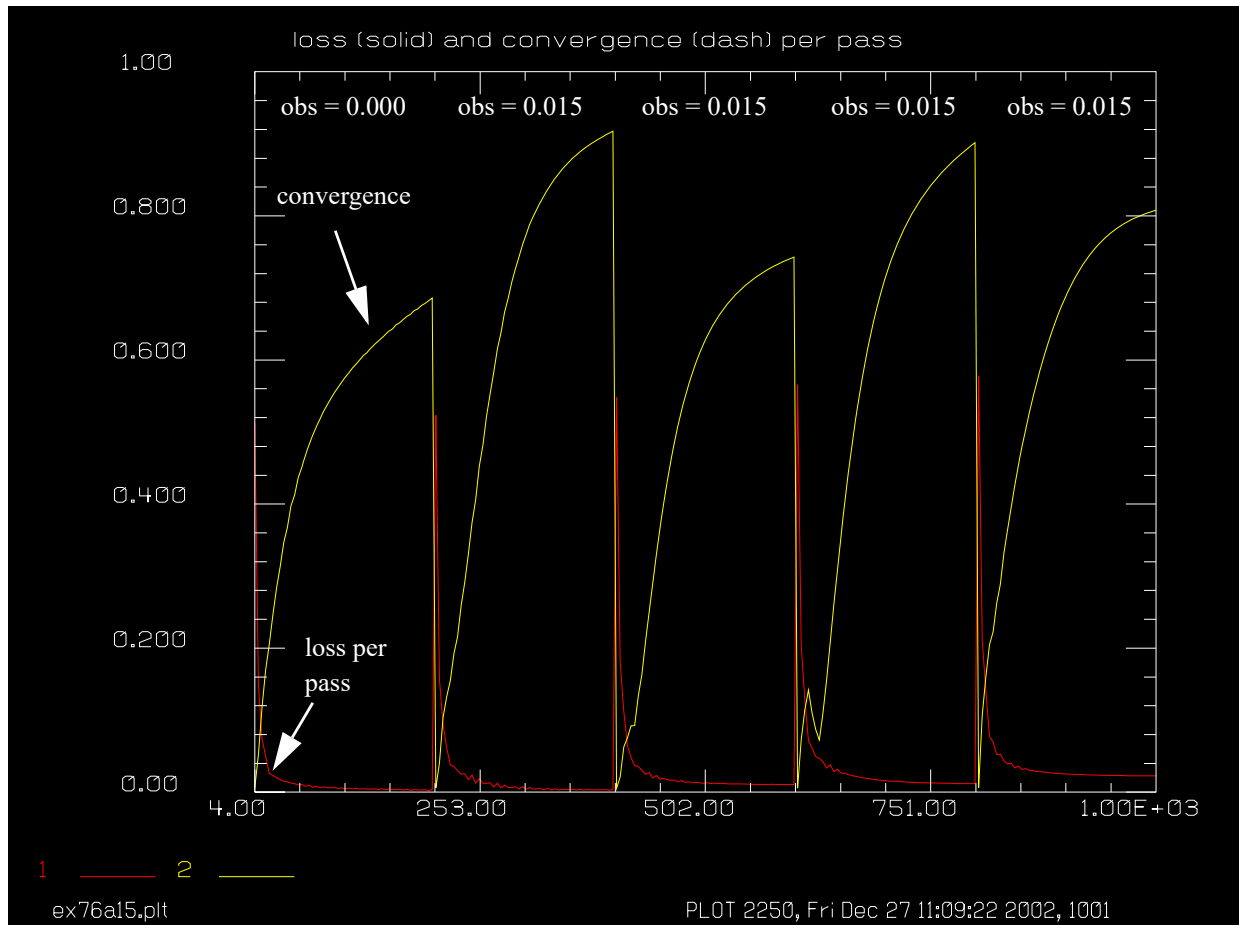


Fig. 76.8. Bare cavity analysis. History of loss (solid line) and convergence (dashed line) for obscuration radii of 0.0, 0.015, 0.030, 0.045, and 0.060. The large apertures lead to very low losses but insufficient mode selectivity. If the mode selectivity were better the convergence would go to unit in a reasonable number of passes. For no obscuration, modes TEM00 and the donut mode beat together. For an obscuration of 0.015, the donut mode dominates, so convergence is better. For an obscuration of 0.030, the donut mode and TEM11 beat together. The alteration of the components of the donut mode, TEM01 and TEM10, cause a side-to-side thrashing. At an obscuration of 0.45 the TEM11 mode is stronger but the donut mode is still present. At an obscuration of 0.060, we see a six-lobe mode in the region of the annular ring defined on the outside by the 0.3 cm radius aperture and on the inside by the 0.06 cm radius obscuration. This mode does directly correspond to any of the Hermite-Gaussian or Laguerre-Gaussian modes.

```

plot/udata 1 2 min=0 max=1. dash
display = 0
endif
energy/norm 1
macro/end
macro/def obs_step/o
  obs = obs+.015
  clear 1 0
  noise 1 1 3
  copy/con 1 2
  energy/norm 1 1
  reson/run Ntimes
  # start from same noise source

```

Jump to: [Commands](#), [Theory](#)

```

    obs_store = obs_store+1
macro/end
macro/def store/o
macro/end
array/s 1 256                                # set array size to 128 x 128
nbeam 2
beams/off 2
title/format f 5 3
wavelength/set 0 .6328                        # set wavelengths
units/s 0 .0008
Ntimes = 200
clap = .3
obs = 0
reson/name reson                             # set resonator name
reson/eigen/test 1                           # run once to calculate eigen mode
reson/eigen/set 1                             # set distribution to ideal eigenmode
obs = -.015
point = 0
set_obs = 1
obs_store = 0
macro obs_step/5                             # step obscuration 5 times

```

Ex76b: Example of coherent injection, automatic frequency control

Example 76b models coherent injection. Ideally the injected signal should exactly match the optical frequency, optical phase, and transverse mode structure of the cavity mode to be drive, assumed to be TEM00. After the resonator is defined, the TEM00 mode is calculated by the `resonator/eigen/test` and `resonator/eigen/set` commands. The characteristics of the TEM00 mode are used to define the injected mode. While we may reasonably use the ideal TEM00 mode as the transverse distribution, we should be precise in controlling the optical frequency. If we are not precise, then even a small error, given many passes will cause great power fluctuations at the beat frequency between the injected frequency and the cavity frequency. Note, that it is necessary to match the frequency of the actual cavity mode. There may well be enough difference between the Gouy shift of the ideal mode and the Gouy shift of the true mode, calculated numerically, to cause beating effects over the convergence time scale. While it is necessary to also exactly match the phase, if we start with a random noise distribution that is random in both power and phase, then the optical mode which grows out of the noise will be phase matched to the input.

To get an exact frequency and phase match in the laboratory, a control system, of some sort, must be included in the configuration. In GLAD, it is easy to calculate the phase difference between cavity mode and injected signal, using the `mult/mode/correlation` command, and simply correct the phase of the injected signal. If desired, GLAD can also be used to model the actual correction hardware, including the effects of finite response time of the instruments.

In this particular configuration, the signal is injected by a beam combiner mirror at a distance of 7.5 cm from the waist, located in the center of the device. Injection of the ideal TEM00 mode causes the TEM00 mode to completely dominate the mode competition, illustrated in Fig. 76. 9. If the phase were not matched, this would not be the case. Example 56 illustrates how variation of the injected frequency can cause any of the transverse modes to dominate based on the exact frequency tuning.

Although the transverse mode is essentially fixed, it takes time for the resonator power to converge to a steady-state solution. To avoid a very tedious number of passes to allow the power to converge, an

Jump to: [Commands](#), [Theory](#)

optimization loop is used to find the steady-state power. Before optimization, 10 cycles are run to show the power gradually rising from the initial guess of 400 (see Fig. 76. Ex10). Then six optimization cycles are used. Optimization requires about 20 passes through the resonator, due to the need to calculate partial derivatives and to determine the best damping. After optimization, 50 cycles of the resonator are run, showing no apparent change in the power, verifying that the steady-state power has been found.

After the first 50 cycles of the nominal system (with no obscuration), the configuration is modified by a 20% increase in injected mode width, a defocus of the injected mode by 0.3 waves, and by including the obscuration. Each of the three effects are considered separately for 50 passes, as shown in Fig. 76. 12. It is apparent that slight changes in mode matching efficiency have a large effect on cavity mode power.

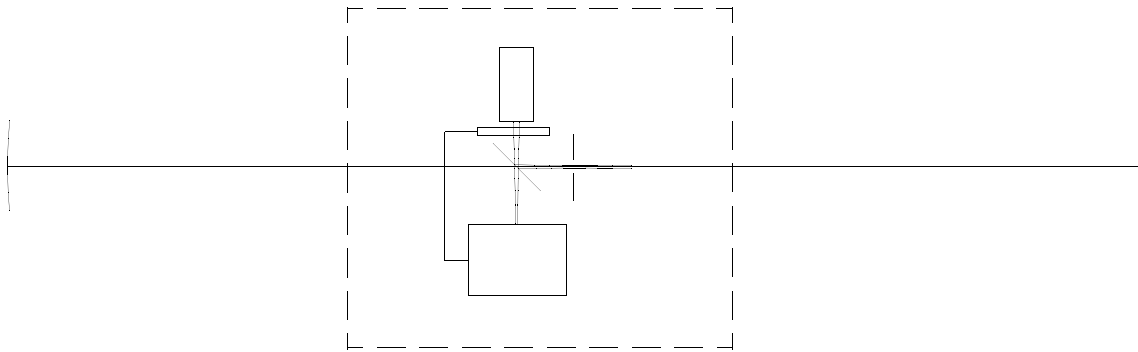


Fig. 76.9. Coherent injection of Fabry-Perot cavity. Steady-state transverse mode with obscuration of 0.015 cm radius, shown at the left mirror just after the obscuration. The mode is close to being a gaussian but the effects of the obscuration and the clear aperture are both visible.

Input: ex76b.inp

```
c## ex76b
c
c Example 76b: Example of coherent injection, automatic frequency control
c
c This example illustrated coherent injection of a laser beam into a
c Fabry-Perot cavity. The model includes a variable cavity length
c adjustment which is controlled to minimize the phase error between
c the injected signal and the cavity mode. The cavity length is adjusted
c by including a variable piston value in the cavity.
c
mem/set/b 1      # set memory to 1 megabyte
time/i          # set time check
c
c variables
c -----
c power          - power in beam after each pass
c ezwaist        - waist location
c piston         - phase error due to detuning between injected and
c                 round trip mode
c enorm          - energy normalization
```

Jump to: [Commands](#), [Theory](#)

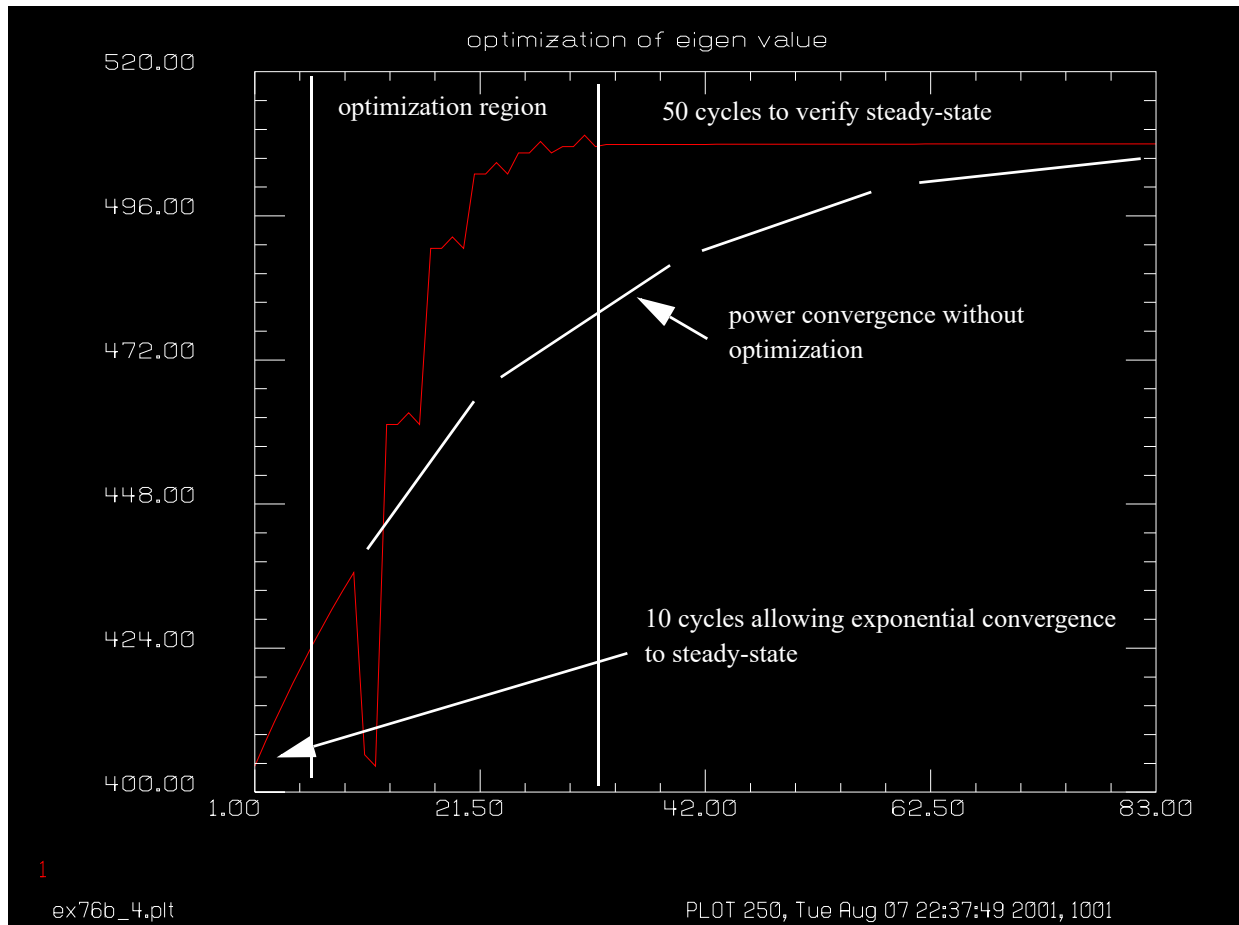


Fig. 76.10. Coherent injection of Fabry-Perot cavity. With coherent injection, the transverse mode is forced by the injected signal. However because of the low losses in the resonator, the power stabilizes with a slowly varying exponential response function. This is shown by 10 cycles after starting with an initial guess of 400 for the cavity power. The dashed line shows approximately how the power in the resonator would exponentially approach steady-state. To find the steady-state mode of the unperturbed resonator quickly, six optimization cycles are used to solve for the eigen power. The bumps in the curve show the changes in power used in determining the numerical partial derivative. After optimization, the resonator is given 50 passes to show that the power determined by optimization does, indeed, give a steady-state solution.

```

c  eng_dif      - round-trip gain or loss
c  units       - units of the beam
c  angcorr     - angle of the correlation between input and cavity mode
c  pass       - pass number
c  ntimes      - number of passes to stabilize
c
c  injected mode variables
c  -----
c  radius      - phase radius of beam
c  focus       - focus error measured in waves
c  waist       - transverse radius of waist of injected beam
c
c  logic switches
c  -----

```

Jump to: [Commands](#), [Theory](#)

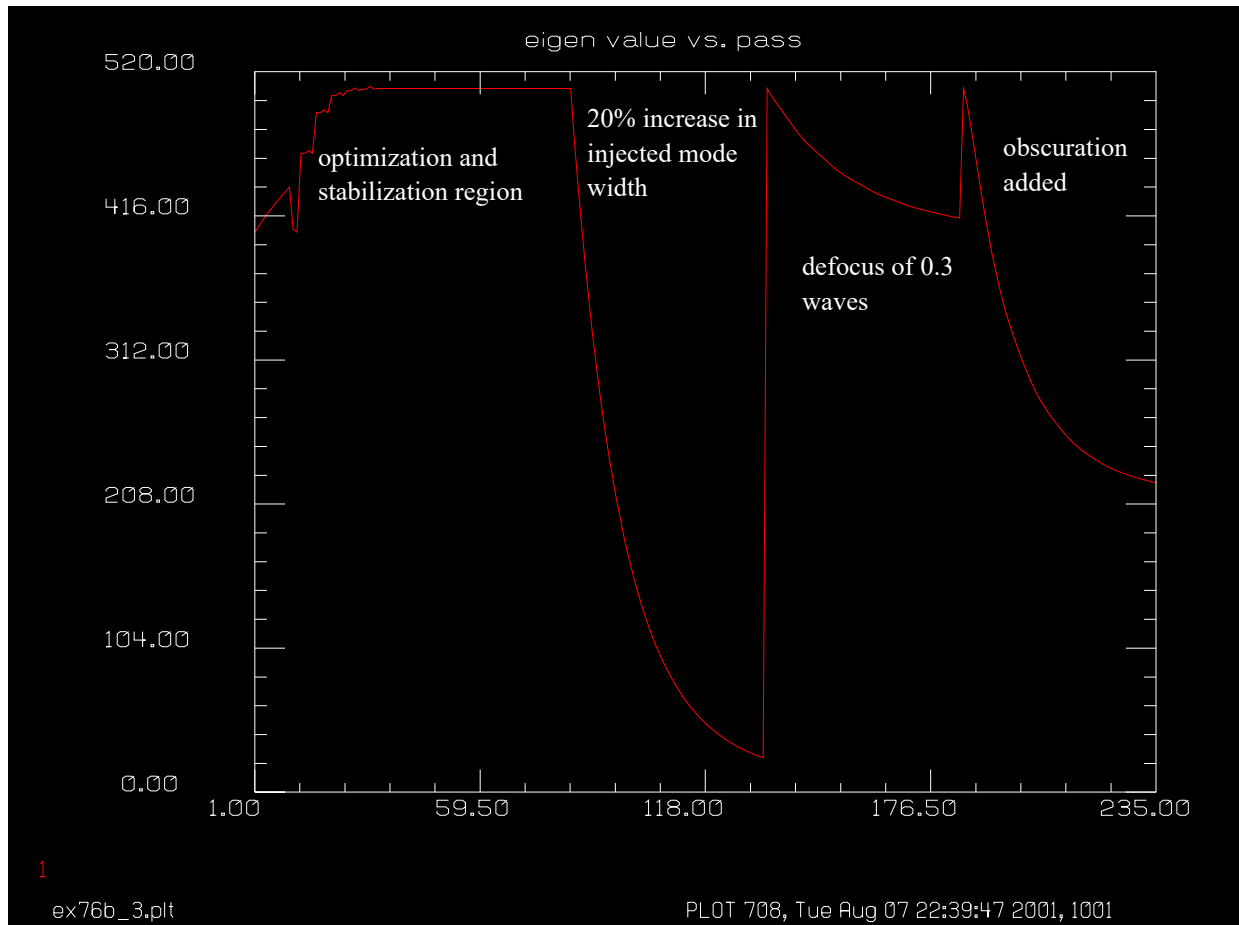


Fig. 76.11. Coherent injection of Fabry-Perot cavity. After the initial 10 cycles, optimization solution, and 50 cycles of steady-state operation, as shown in Fig. 76. 9, The width of the injected signal is increased by 20%, while maintaining the same injected power. The mode mismatch causes a sharp drop in cavity power as soon as the change is made. After 50 cycles of wider injected mode, the resonator is returned to steady-state (which was saved) and a defocus of 0.3 waves was placed on the injected signal, leading to another drop in cavity power in the next 50 cycles. The last test is to include the obscuration.

```

c inject      - logical switch to turn on injected beam
c set_piston  - logical switch to turn on cavity length correction
c set_obs     - logical switch to turn on obscuration
c
variab/dec/int pass inject set_piston set_obs ntimes
set/alias_stop/off
c
c Coherent injection
c
macro/def reson/o
    pass = pass+1                # increment pass counter
    if inject = 1 then
c
c This is the injection code. The mode is calculated below
c
    units/s 2 units

```

Jump to: [Commands](#), [Theory](#)

```

    gaus/c/c 2 1 r0=waist      # change waist to change size of gaussian
    abr/focus 2 focus rn=.015 # change focus to change phase radius
    energy/norm 2 1           # normalize injected signal
    dist -8.5 2               # back up to injected beam location
    zreff/se 2 0              # reset axis
    if set_piston = 1 then
        mult/mode/corr 1 2      # calculate correlation
        variab/set angcorr angcorr list # calculate correlation
        piston = angcorr list   # set piston adjustment
                                # to correlation phase
        phase/piston 2 piston
    endif
    add/coh/con 1 2           # coherent injection
endif
mult 1 .9207                  # apply efficiency for beam combiner
prop 83                       # propagate to next mirror
mirror/sph 1 -75              # right mirror, 75 cm. radius
clap/c/n 1 .3                 # .3 cm. radius aperture
mult 1 .998                   # efficiency of mirror 2
title right mirror
if inject = 1 then
    plot/watch plot1.plt
    plot/l 1 ns=128 xrad=.4
endif
prop 149                      # propagate 149 cm. along beam
mirror/sph 1 75               # left mirror, 75 cm. radius
clap/c/n 1 .3                 # .3 cm. radius aperture
mult 1 .998                   # efficiency of mirror 1
if set_obs = 1 obs 1 .015     # 150 micron obscuration radius
title left mirror
if inject = 1 then
    plot/watch plot1.plt
    plot/l 1 ns=128 xrad=.4
endif
prop 66                       # propagate to waist
if set_piston = 1 phase/piston 1 -piston
variab/set power 1 energy      # calculate the current power
if inject = 1 then
    udata/set pass pass power piston # store the power
    plot/watch plot2.plt
    title eigen value vs. pass
    plot/udata 1 1 min=0 max=520
endif
macro/end
array/s 1 256                 # set array size to 256 x 256
nbeam 3                       # establish 3 beams
beams/off 2 3
wavelength/set 0 .6328        # set wavelengths
units/s 0 .0008
inject = 0
ntimes = 50
reson/name reson              # set resonator name
reson/eigen/test 1            # run once to calculate ideal eigenmode
reson/eigen/set 1             # set distribution to ideal eigenmode

```

Jump to: [Commands](#), [Theory](#)

```

energy/norm 1
reson/run 1
variab/set waist 1 resonator/ewaist list # ideal eigen waist size
variab/set radius 1 resonator/radius list # ideal phase radius
variab/set ezwaist 1 resonator/ezwaist list # ideal waist location
c
c calculate Guoy shift with one pass
c
dist 8.5 1
variab/set units 1 units # store units of eigen mode at the waist
dist -8.5 1
c
c Let's run the cavity once to get a good guess and cavity mode
c Otherwise we spend many passes building up cavity power
c
reson/run 1 # run once to set up correlation
inject = 1 # switch injection on
focus = 0.
set_piston = 1 # turn on piston correction
enorm = 400 # starting condition, deliberately not a good guess
energy/norm 1 enorm
pass = 0
macro reson/10 # observe bad guess for power
macro/def optres
    energy/norm 1 enorm
    reson/run 1
    eng_dif = power-enorm
macro/end
opt/name optres
opt/var/add enorm 2
opt/tar/add eng_dif
opt/run 6
copy 1 3
reson/run ntimes
plot/watch ex76b_4.plt
title optimization of eigen value
plot/udata 1 1 min=400 max=520
c
c We should now have a good approximation to steady-state
c
copy 3 1
waist = waist*1.2 # increase injected waist size 20%
reson/run ntimes
waist = waist/1.2
copy 3 1
reson/run 1 # reestablish original condition
focus = .3 # .3 waves of defocus
reson/run ntimes
focus = .0
copy 3 1
reson/run 1 # reestablish original condition
set_obs = 1 # include obscuration
reson/run ntimes
set_obs = 0

```

Jump to: [Commands](#), [Theory](#)

```
system/copy plot1.plt ex76b_1.plt
plot/watch ex76b_2.plt
title eigen value vs. pass
plot/udata 1 1 min=400 max=520
plot/watch ex76b_3.plt
title eigen value vs. pass
plot/udata 1 1 min=0 max=520
time
```


Ex77: Hollow waveguide with reflecting walls

Table. 77.1. Table of Ex77 examples

Ex77a: Hollow waveguide with reflecting walls	1
Ex77b: Tapered waveguide with a converging beam centered at the perspective point	7
Ex77c: Inject collimated beam into tapered waveguide	10
Ex77d: Inject collimated beam into curved waveguide	12
Ex77e: Waveguide optical integrator	14
Ex77f: Waveguide in a stable resonator	16
Ex77g: Half waveguide in unstable resonator	19
Ex77h: Resonator with waveguide in place	23
Ex77i: Incoherent treatment of reflecting-wall waveguide	27
Ex77j: Incoherent treatment of reflecting-wall waveguide, converging beam	32

The natural aliasing feature of discrete Fourier transforms provides a convenient means of modeling a hollow waveguide with reflecting walls. A reflecting wall reflects the light back into the beam path but aliasing “wraps” the distribution back into the array on the opposite side. If the distribution were an even function, then the wrapping would be identical to reflecting. We can force arbitrary distribution to be effectively even by putting the array of interest in the upper left quadrant of an array of twice the width. We then fill the other three quadrants with mirror images of the upper left quadrant. The overall array is even and the aliasing will correctly model wall reflections.

To illustrate this effect, a circular beam was injected into the waveguide at an offset position and with a tilt. The beam is successively reflected off all four walls.

Ex77a: Hollow waveguide with reflecting walls

Input: `ex77a.inp`

```

c##  ex77a
c
c Example Ex77a: Hollow waveguide with reflecting walls
c
c
c This example illustrates how to model a hollow waveguide with
c reflecting walls. We take advantage of aliasing intrinsic to
c discrete Fourier transforms to model the reflections at the walls.
c
c The distribution of interest is a circular aperture with a
c phase tilt of 6 waves at the edge of the aperture.
c
c We copy the array of 128 x 128 into the upper quadrant of
c a 512 x 512 array and then copy images as formed by reflecting
c walls into the other three quadrants. The FLIP command is used
c to form the images.
c
c The resulting distribution shows the beam propagating toward
c the lower mirror and then bounces off the left mirror as well.

```

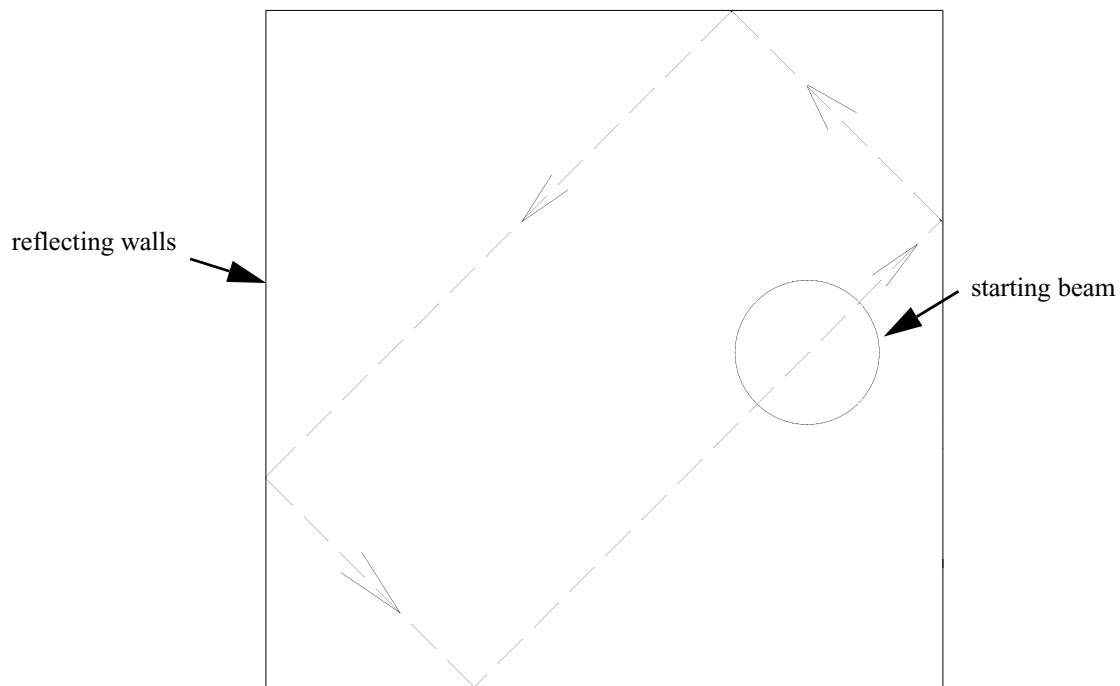


Fig. 77.1. Consider a beam in a hollow waveguide with reflecting walls. The beam is given a tilt which sends it toward the upper right. The beam will reflect around the walls while expanding because of diffraction.

```
c
c Any distribution may be selected for the starting function
c and will be properly modeled.
c
poptext/compose ex77.txt
Example of a circular beam propagating in a square waveguide
with reflecting walls.
```

The beam is injected with a slight tilt so it bounces off all four walls.

```
Look for self-interference near the walls.
poptext/end
poptext ex77.txt 10
variab/dec/int pass
variab/dec/int pass pass1
mem/set/b 3
array/s 1 256 256
units/s 1 .125
c
c Any distribution may be selected for beam 1
c
clap/c/c 1 6 8
abr/tilt 1 -6 45
nbeam 2 512 512
clear 2 0
```

Jump to: [Commands](#), [Theory](#)

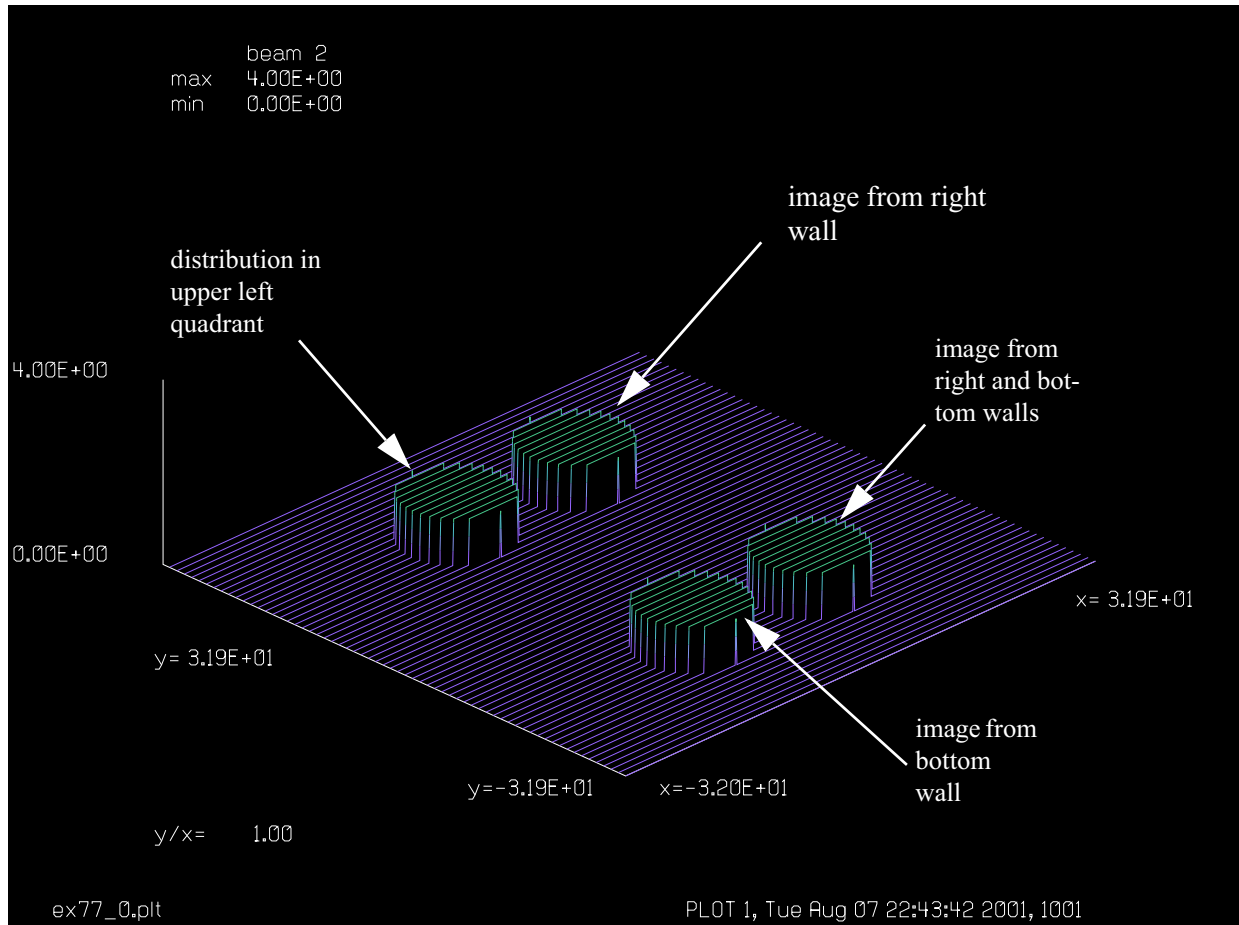


Fig. 77.2. A hollow waveguide may be represented by placing the distribution in one quadrant (here the upper left quadrant) and placing images as formed by the walls in the other three quadrants. The starting distribution is offset and has a tilt which directs the beam initially toward the upper right.

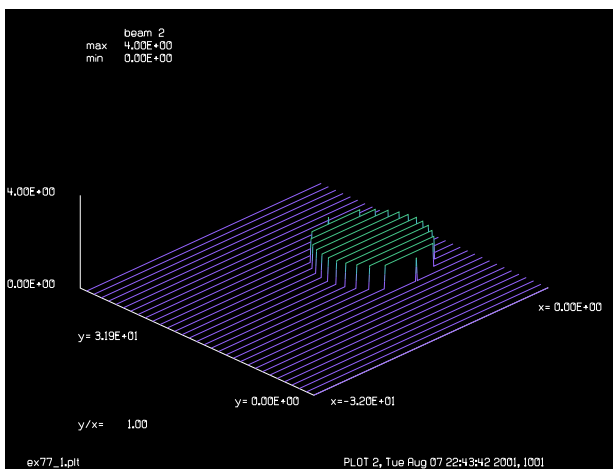


Fig. 77.3. Start with tilt aberration, showing only upper left quadrant, as with rest of figures.

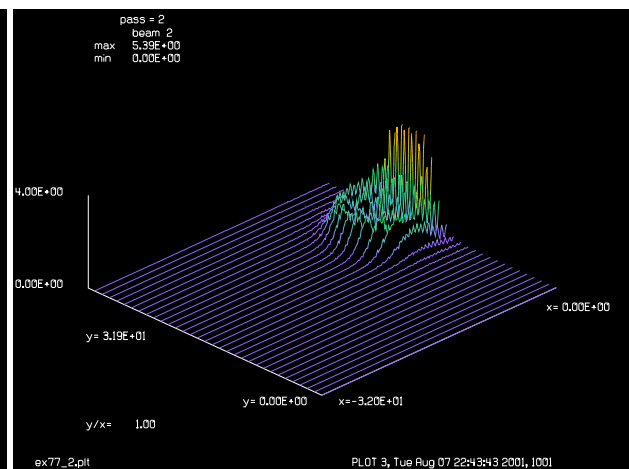


Fig. 77.4. Beam is tilted to upper right and hits the right wall.

units/s 2 .125

Jump to: [Commands](#), [Theory](#)

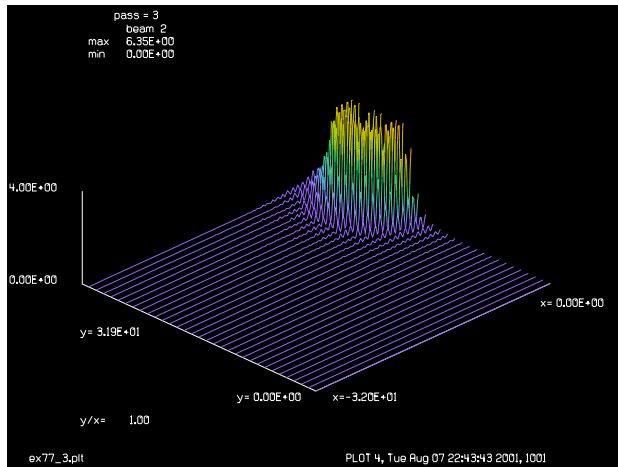


Fig. 77.5. Beam collides with right wall and is deflected toward top wall.

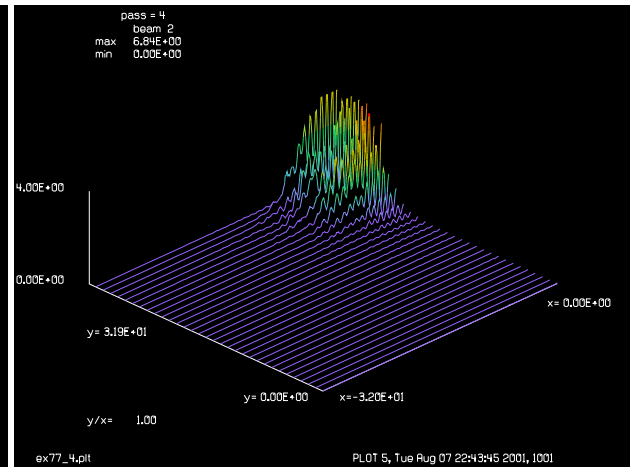


Fig. 77.6. Beam is passing from right to top wall.

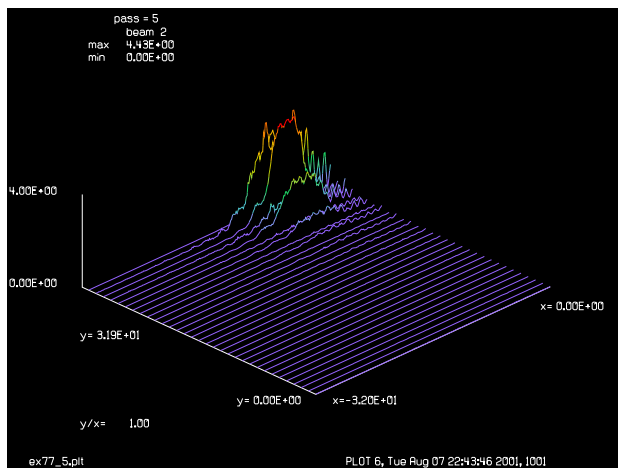


Fig. 77.7. Beam is now reflecting off top wall.

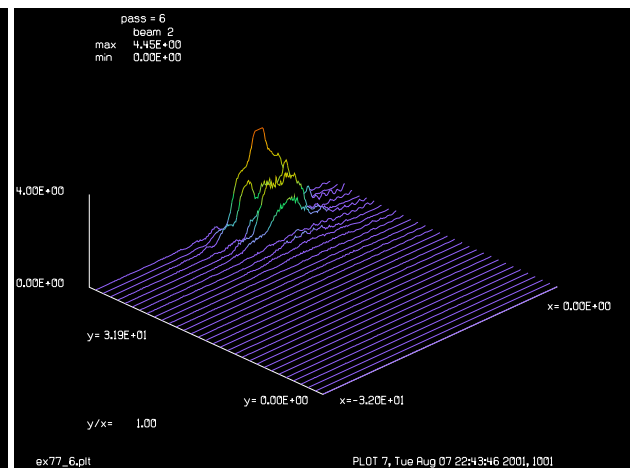


Fig. 77.8. Beam is now headed for left wall after reflection from top wall.

```

c
c copy images formed by walls into quadrants of beam 2
c
copy/con 1 2 -128 -128
flip/x 1
copy/con 1 2 128 -128
flip/y 1
copy/con 1 2 128 128
flip/x 1
copy/con 1 2 -128 128
units/fix 2
c
c Display upper left quadrant, which represents the array of
c interest.
c
macro/def ex77/o

```

Jump to: [Commands](#), [Theory](#)

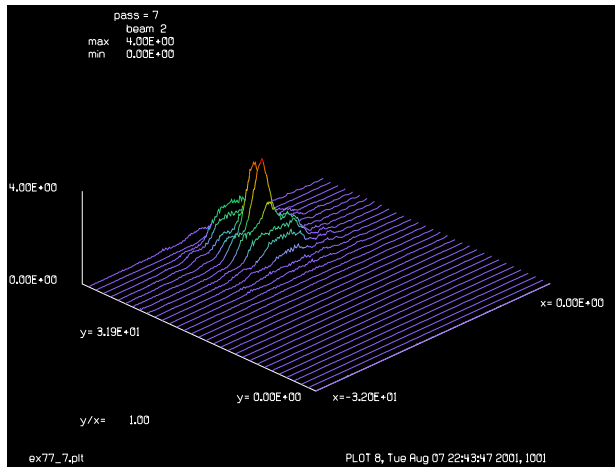


Fig. 77.9. Beam is coming off top wall.

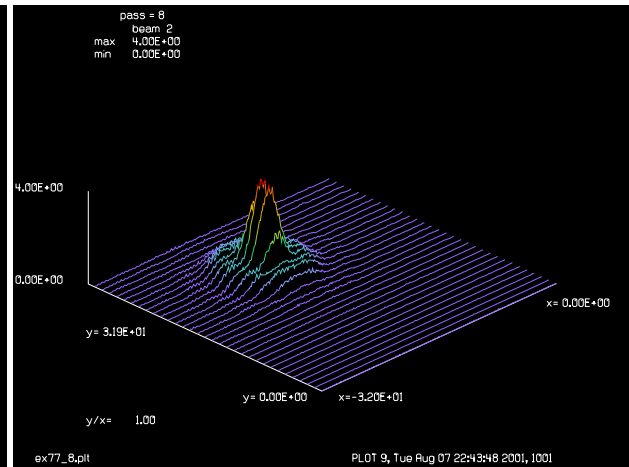


Fig. 77.10. Beam is headed for left wall. Note, how diffraction has spread the flat-top function.

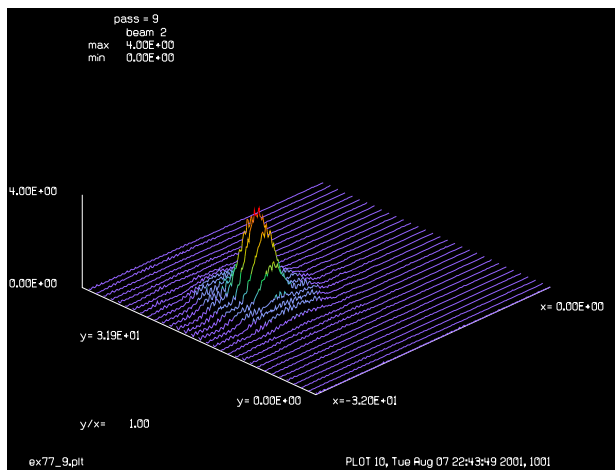


Fig. 77.11. Still headed for left wall.

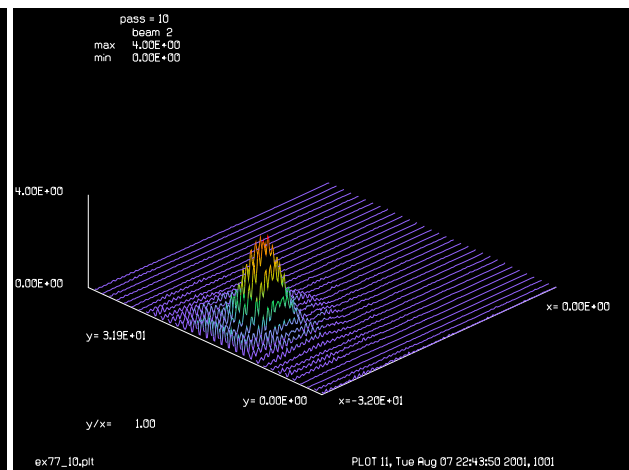


Fig. 77.12. Beam impacts at left wall.

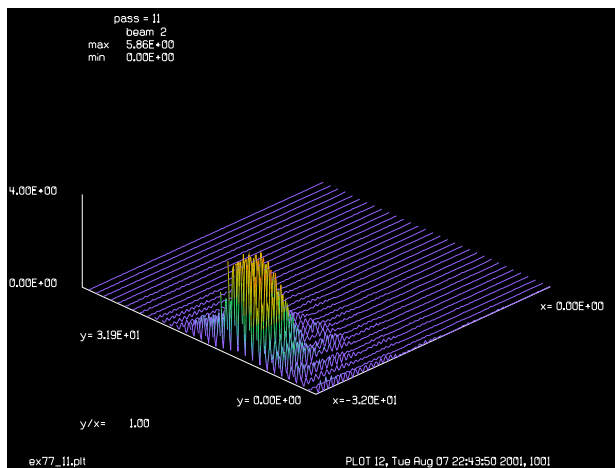


Fig. 77.13. Beam has its center at the left wall.

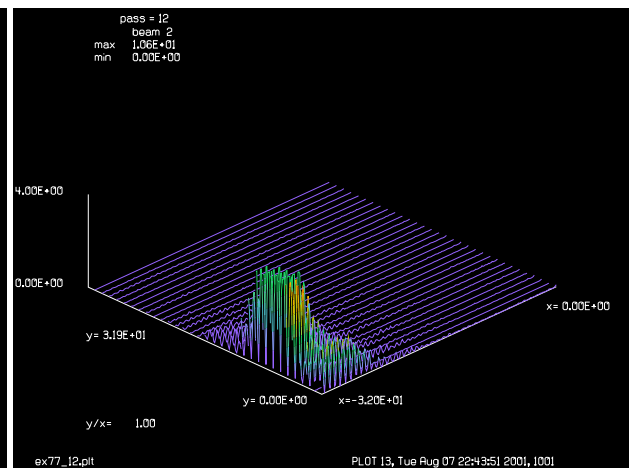


Fig. 77.14. Beam is now headed into bottom wall.

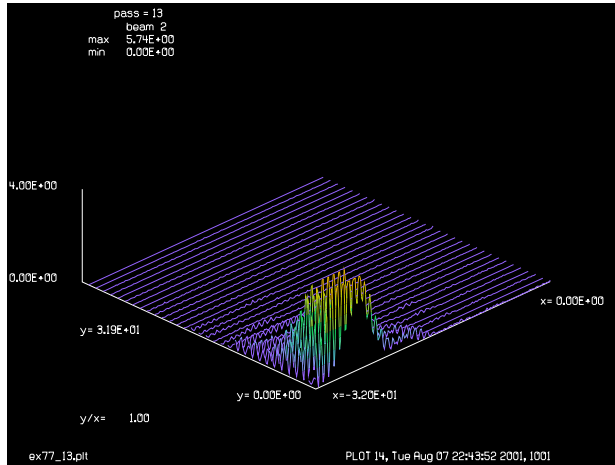


Fig. 77.15. Beam is part way into bottom wall.

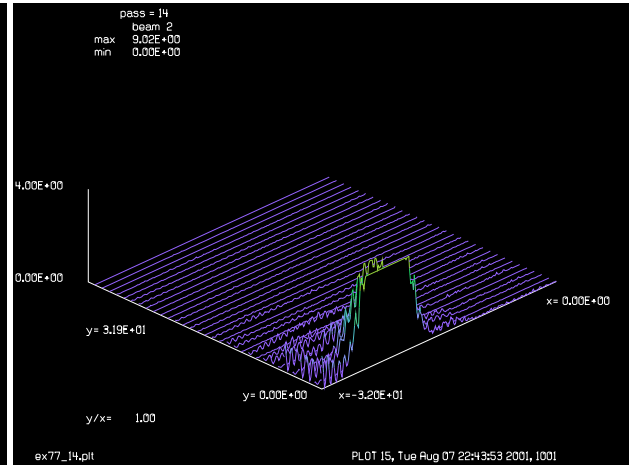


Fig. 77.16. Beam is half way into bottom wall.

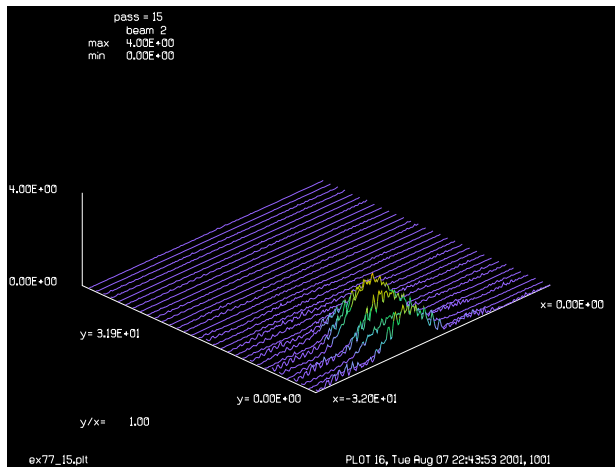


Fig. 77.17. Beam is now reflecting off bottom wall.

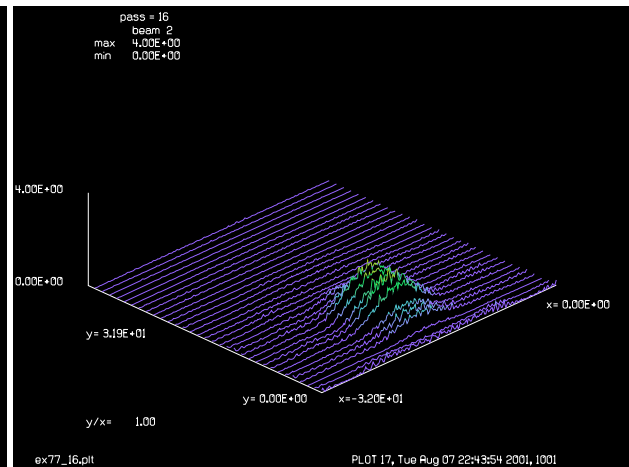


Fig. 77.18. Beam is headed for original position.

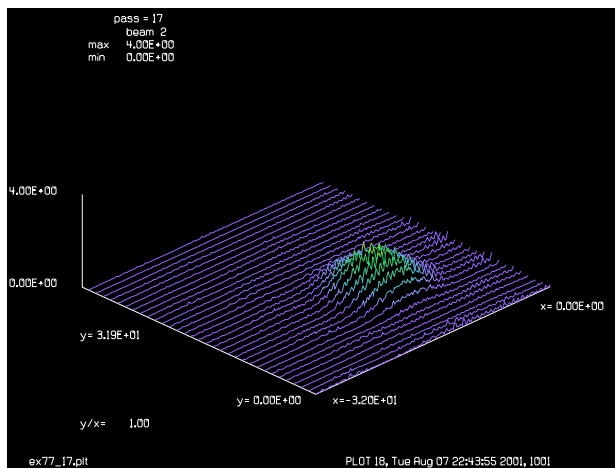


Fig. 77.19. Note far-field diffraction pattern.

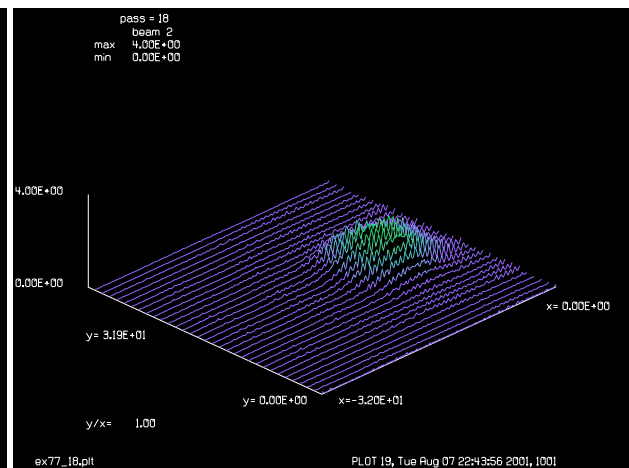


Fig. 77.20. Beam is now at the starting position, but is wider so it is already hitting the right wall.

```

pass = pass + 1
pass1 = pass - 1
dist 5000 2
title pass = @pass
poptext/compose ex77.txt

```

Example of a gaussian beam propagating in a square waveguide with reflecting walls.

The beam is injected with a slight tilt so it bounces off all four walls.

Look for self-interference near the walls.

```

pass @pass1 of 17
poptext/end
  poptext ex77.txt 5
  plot/watch ex77_@pass.plt
  plot/1 2 xrad=16 xdec=-16 ydec=16 max=4.
macro/end
pass = 0
plot/screen/pause 3
plot/watch ex77_@pass.plt
plot/1 2 max=4 ns=64
pass = pass + 1
plot/watch ex77_@pass.plt
plot/1 2 xrad=16 xdec=-16 ydec=16 max=4.
mac ex77/17
pause 5
end

```

Ex77b: Tapered waveguide with a converging beam centered at the perspective point

Example Ex. 77b illustrates a tapered waveguide with a converging beam centered at the perspective point of the waveguide. Fig. 77.21 shows a straight waveguide. The points of intersection with the waveguide are evenly spaced and the angles are constant. Fig. 77.22 shows a tapered waveguide. The intersection points become closer together and the angles steeper. If a ray is too steep for the length and taper of the guide it will be reflected back out the front. A sphere formed by rotating the rear face about the perspective point indicates the sphere of acceptance (see Fig. 77.22). A tapered waveguide is achieved by applying a lens to the composite array containing the four imaged quadrants (see Fig. 77.23). The perspective point of the waveguide is set at the focus point of the lens. Because of the lens, the curvature of the beam will be centered at the perspective point.

Input: ex77b.inp

```

c## ex77b!211800595633675
c
c Example Ex77b: inject converging beam into tapered waveguide
c
c Hollow waveguide, square cross section, reflecting walls,
c injection of converging beam (converging to the perspective point

```

Jump to: [Commands](#), [Theory](#)

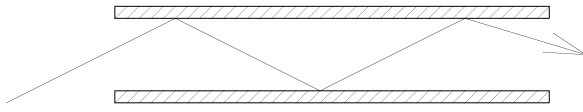


Fig. 77.21. Beam injected into straight wall waveguide makes equally spaced intersections with the wall.

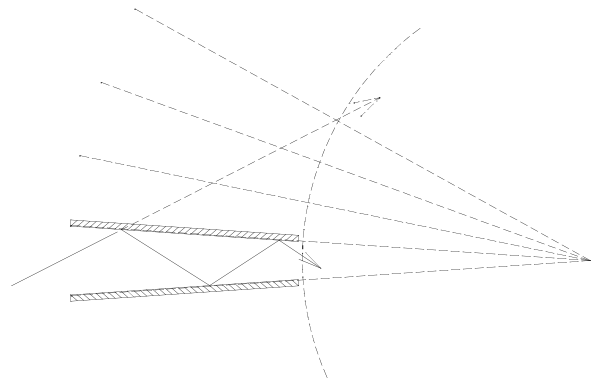


Fig. 77.22. Tapered waveguide. The intersections points become closer together and the angles steeper as the ray moves into the waveguide. The intersection points are “chirped”. We may consider the multiple reflections of the waveguide faces. A ray that is too steep will miss the sphere formed by rotating the grating about the perspective point.

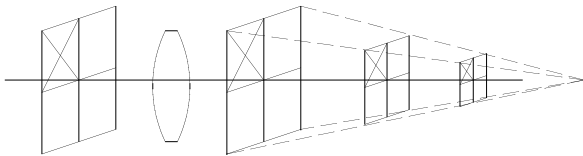


Fig. 77.23. The taper is achieved by adding a lens to the composite array containing the four imaged quadrants. The coordinate system converges to the focus point of the lens which becomes the perspective point of the tapered waveguide.

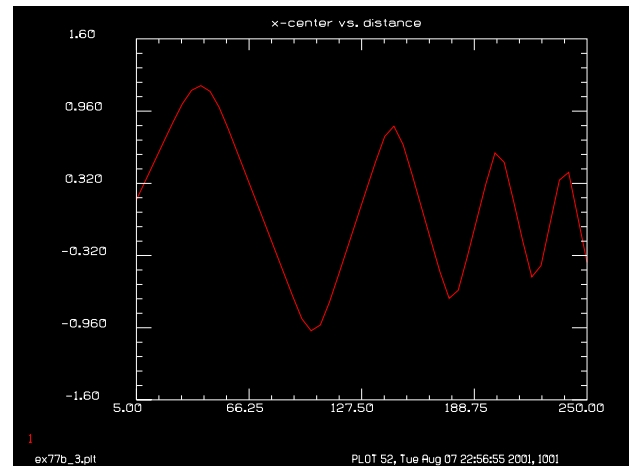


Fig. 77.24. Trajectory of beam hitting waveguide walls shows chirped behavior for Ex77b.inp.

c of the waveguide taper) .

c

c Convergence of waveguide is established by lens of focal length 400
 c -- same as convergence of injected beam. Note that the plot of x-center
 c consists of straight-line segments with bends when the beam
 c hits the wall. The angle of the beam is increased with each
 c bounce as indicated by the “chirp” shown in plot2.plt.

c

Jump to: [Commands](#) , [Theory](#)

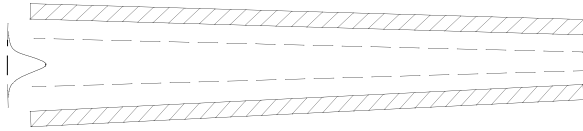


Fig. 77.25. The lens tends to produce a converging beam with width decreasing with distance. Ex77b.

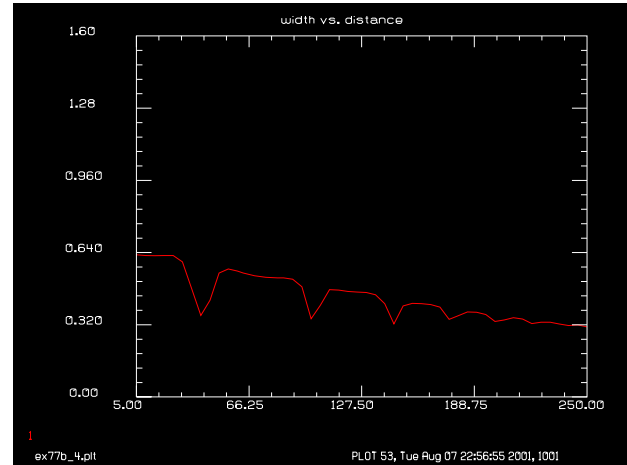


Fig. 77.26. Width decreases for convergent input beam. Ex 77b.inp.

```

c The beam width decreases linearly with the decrease in
c waveguide width.
c
variab/dec/int pass
mem/set/b 3
array/s 1 256 256
units/s 1 .0125
c
c Any distribution may be selected for beam 1
c
clap/c/c 1 .6
Lambda = 10.6e-4
focal_length = 400
#
# put diverging phase curvature to cancel curvature
# of subsequent lens of same focal length.
#
# injected collimated beam maintains beam width (approximately)
# but shows similar translation as a function. Translation
# shows "chirp" because angle of phase tilt increases with
# each reflection.
#
abr/tilt 1 -20 90
nbeam 2 512 512
clear 2 0
units/s 2 .0125
c
c copy images formed by walls into quadrants of beam 2
c
copy/con 1 2 -128 -128
flip/x 1
copy/con 1 2 128 -128
flip/y 1
copy/con 1 2 128 128
flip/x 1

```

Jump to: [Commands](#), [Theory](#)

```

copy/con 1 2 -128 128
c
c Display upper left quadrant, which represents the array of
c interest.
c
lens 2 focal_length      # apply lens to four-quadrant array
z = 0.
zstep = 5
macro/def ex77c/o
    pass = pass + 1
    dist zstep 2
    z = z + zstep
    title pass = @pass
    variab/set Units 2 units
    Field = 128*Units
    plot/watch ex77b_2.plt
    plot/1 2 xrad=Field xdec=-Field ydec=-Field
    copy/con 2 1 128 -128  # extract quadrant
    units/beam 1 2
    fitgeo 1
    variab/set X 1 fitxcen list
    variab/set Rad 1 fitxomega
    udata/set pass z X Rad
macro/end
pass = 0
variab/set Units 2 units
Field = 128*Units
plot/w ex77b_1.plt
plot/1 2 xrad=Field xdec=-Field ydec=-Field
mac ex77c/50
plot/w ex77b_3.plt
title x-center vs. distance
plot/udata 1 min=-1.6 max=1.6
plot/w ex77b_4.plt
title width vs. distance
plot/udata 2 min=0 max=1.6

```

Ex77c: Inject collimated beam into tapered waveguide

Input: `ex77c.inp`

```

c## ex77c!768434051563639
c
c Example Ex77c: inject collimated beam into tapered waveguide
c
c Hollow waveguide, square cross section, reflecting walls,
c injection of collimated beam.
c
c The beam is given an incident tilt. At each reflection the
c beam angle increases giving the "chirp" shown in plot2.plt.
c The beam width is nearly constant vs. distance because it
c is initially collimated as shown in plot3.plt.
c

```

Jump to: [Commands](#), [Theory](#)

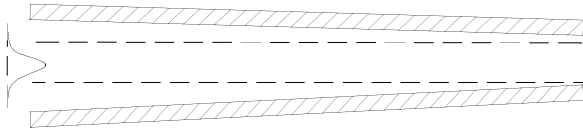


Fig. 77.27. A collimated input may be modeled by adding appropriate divergence to exactly compensate for the focusing of the lens. Ex 77c.

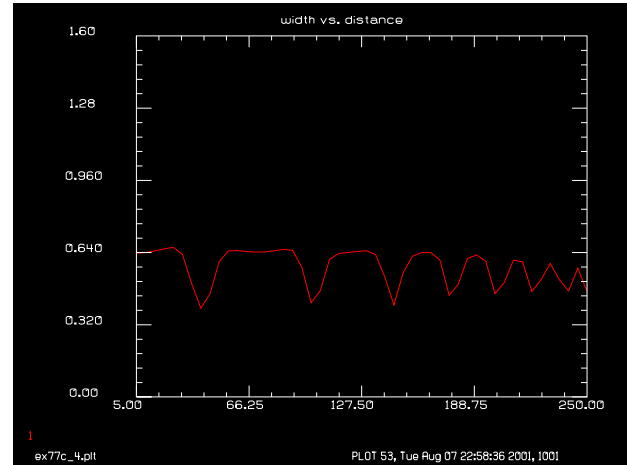


Fig. 77.28. Width remains constant for collimated input, except at the intersection points. Ex 77c.inp.

```

variab/dec/int pass
mem/set/b 3
array/s 1 256 256
units/s 1 .0125
c
c Any distribution may be selected for beam 1
c
clap/c/c 1 .6
Lambda = 10.6e-4
focal_length = 400
#
# put diverging phase curvature to cancel curvature
# of subsequent lens of same focal length.
#
# injected collimated beam maintains beam width (approximately)
# but shows similar translation as a function. Translation
# shows "chirp" because angle of phase tilt increases with
# each reflection.
#
waves = -1*.6^2/2./Lambda/focal_length
abr/foc 1 waves rn=.6
abr/tilt 1 -20 90
nbeam 2 512 512
clear 2 0
units/s 2 .0125
c
c copy images formed by walls into quadrants of beam 2
c
copy/con 1 2 -128 -128
flip/x 1
copy/con 1 2 128 -128
flip/y 1
copy/con 1 2 128 128
flip/x 1

```

Jump to: [Commands](#), [Theory](#)

```

copy/con 1 2 -128 128
c
c Display upper left quadrant, which represents the array of
c interest.
c
lens 2 focal_length
z = 0.
zstep = 5
macro/def ex77d/o
    pass = pass + 1
    dist zstep 2
    z = z + zstep
    variab/set Units 2 units
    Field = 128*Units
    title pass = @pass
    plot/watch ex77c_2.plt
    plot/1 2 xrad=Field xdec=-Field ydec=-Field
    copy/con 2 1 128 -128    # extract quadrant
    units/beam 1 2
    fitgeo 1
    variab/set X 1 fitxcen list
    variab/set Rad 1 fitxomega
    udata/set pass z X Rad
macro/end
pass = 0
variab/set Units 2 units
Field = 128*Units
plot/w ex77c_1.plt
plot/1 2 xrad=Field xdec=-Field ydec=-Field
mac ex77d/50
plot/w ex77c_3.plt
title x-center vs. distance
plot/udata 1 min=-1.6 max=1.6
plot/w ex77c_4.plt
title width vs. distance
plot/udata 2 min=0 max=1.6

```

Ex77d: Inject collimated beam into curved waveguide

Input: ex77d.inp

```

c## ex77d!319742795271032
c
c Example Ex77d: inject collimated beam into curved waveguide
c
c Hollow waveguide, square cross section, reflecting walls,
c waveguide curved with large radius, injection of collimated beam.
c
c The input beam will bounce off the walls striking one wall --
c the left or right wall depending on the sign of radius -- and back
c to the center.
c
c In this case we made the waveguide have a half-width of .01 cm.

```

Jump to: [Commands](#), [Theory](#)

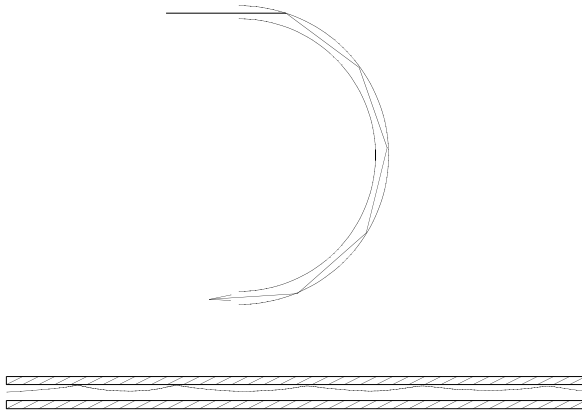


Fig. 77.29. Light injected into a curved waveguide bounces around the circumference in a “square-the-circle” manner. We may treat this with a straight waveguide model where the coordinate system of the beams is constantly changed by adding tilt aberration.

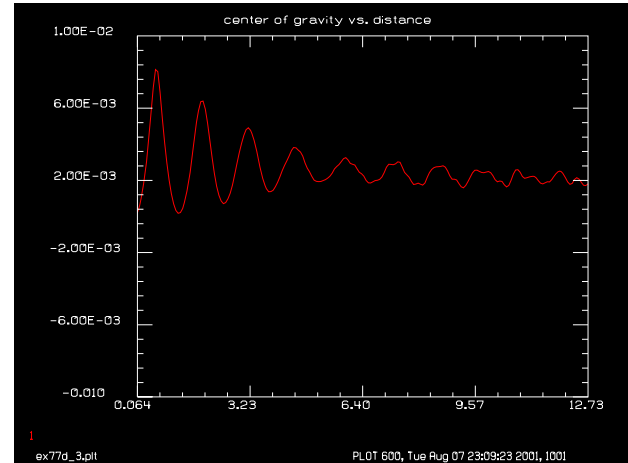


Fig. 77.30. Motion of the center of the beam relative to the walls. See Fig. 77.29. Ex 77d.inp. The apparent motion decreases as the beam diameter increases.

```

c the period of oscillation is found from the sag formula.
c
c period = sqrt(Radius*Width/2) = 1.11 cm
c
c where Width = .01 is the half width and Radius = 250.
c
c The period measured from the numerical calculation agrees well
c with the expected value of 1.11.
c
set/cpu 2
variab/dec/int pass
mem/set/b 32
Nline = 1024
Nline2 = Nline/2
Nline4 = Nline/4
array/s 1 Nline2 data
Width = .0100 # 100 micron wide guide
units/field 1 Width # will make 100 micron wide guide
gaus/c/c 1 1 .0030
Lambda = .5e-4
nbeam 2 Nline
wavelength/set 0 Lambda*10000
clear 2 0
units/field 2 Width*2
variab/set/par Units 1 units
Nsteps = 200
Radius = 250
AzDeg = 90
MaxAlpha = Lambda/Units/2. # maximum angle that array can accommodate
MaxZ = Radius*MaxAlpha/2./pi/2 # arc length corresponding to half of
                                # max angle

```

Jump to: [Commands](#), [Theory](#)

```

zstep = MaxZ/Nsteps          # compute incremental step size
AlphaPerStep = MaxAlpha/Nsteps # angle per step
Theta = AlphaPerStep/Lambda  # waves of tilt for rn=1
macro/def ex77d/o
  pass = pass + 1
  z = z + zstep
  abr/tilt 1 -Theta rn=1 azdeg=AzDeg
  copy/con 1 2 -Nline4 -Nline4
  flip/x 1
  copy/con 1 2 Nline4 -Nline4
  flip/y 1
  copy/con 1 2 Nline4 Nline4
  flip/x 1
  copy/con 1 2 -Nline4 Nline4
  zreff/se 2 0
  dist zstep 2
  title pass = @pass
  plot/watch ex77d_1.plt
  variab/set Units 2 units
  copy/con 2 1 Nline4 -Nline4      # extract quadrant
  plot/w plot1.plt
  title single beam, pass = #pass
  plot/l 1
  plot/w ex77d_2.plt
  title doubled beam, pass = @pass
  plot/l 2
  pass1 = pass1 + 1
  units/beam 1 2
  fitgeo 1
  variab/set X 1 fitxcen list
  variab/set Rad 1 fitxomega
  udata/set pass1 z X Rad
  plot/w ex77d_3.plt
  title center of gravity vs. distance
  plot/udata min=-.01 max=.01
macro/end
mac ex77d/Nsteps

```

Ex77e: Waveguide optical integrator

Input: ex77e.inp

```

c## ex77e
c
c Example Ex77e: Waveguide optical integrator
c
c This example illustrates how to model a hollow waveguide with
c reflecting walls. We take advantage of aliasing intrinsic to
c discrete Fourier transforms to model the reflections at the walls.
c
c We copy the array of Nline x Nline into the upper quadrant of
c a Nline*2 x Nline*2 array and then copy images as formed by reflecting
c walls into the other three quadrants. The FLIP command is used

```

Jump to: [Commands](#), [Theory](#)

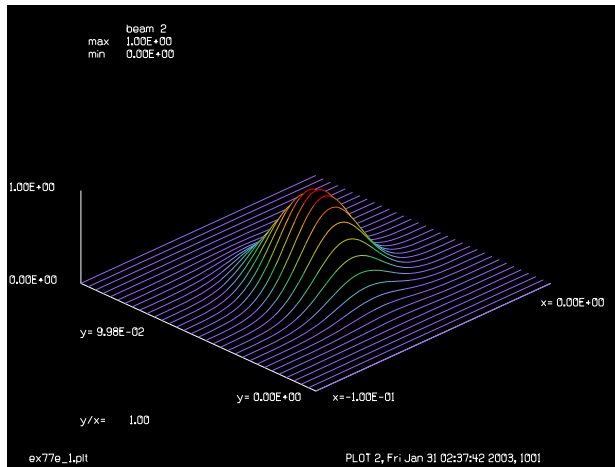


Fig. 77.31. Starting gaussian for waveguide homogenizer 0.1 x 0.1 cm square and 32 cm long.

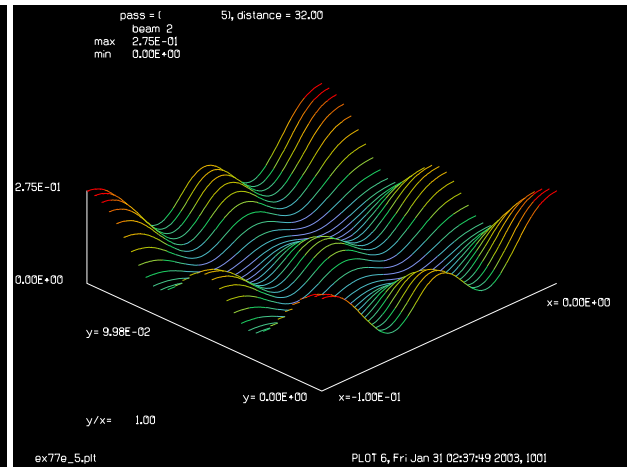


Fig. 77.32. Output of homogenizer. Shows moderate smoothing. Ex77e.inp.

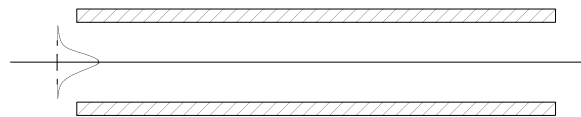


Fig. 77.33. Waveguide used as homogenizer for gaussian beam. Ex 77e.

```
c to form the images.
c
c The resulting distribution shows the beam propagating toward
c the lower mirror and then bounces off the left mirror as well.
c
c Any distribution may be selected for the starting function
c and will be properly modeled.
c
variab/dec/int pass Nline
Nline = 512
# A, vertical width
# B, horizontal width
A = .1
B = .1
mem/set/b 10
array/s 1 Nline
units/field 1 B/2 A/2      # set half-width of optical beam
c
c Any distribution may be selected for beam 1
c
gaus/c/c 1 1 .025          # input beam of .05 mm diameter at 1/e^2
nbeam 2 Nline*2
wavelength/set 0 10.6      # six
```

Jump to: [Commands](#), [Theory](#)

```

clear 2 0
geodata/set/waist 2 waistx=1 waisty=1 # set waist large to extend Rayleigh rang
units/field 2 B A
c
c copy images formed by walls into quadrants of beam 2
c
copy/con 1 2 -Nline/2 -Nline/2
flip/x 1
copy/con 1 2 Nline/2 -Nline/2
flip/y 1
copy/con 1 2 Nline/2 Nline/2
flip/x 1
copy/con 1 2 -Nline/2 Nline/2
units/fix 2
c
c Display upper left quadrant, which represents the array of
c interest.
c
macro/def ex77e/o
    pass = pass + 1
    dist L 2
    Ztotal = Ztotal + L
    title pass = @pass, distance = @Ztotal
    plot/watch ex77e_@pass.plt
    plot/1 2 xrad=B/2 xdec=-B/2 ydec=A/2
macro/end
pass = 0
plot/watch ex77e_@pass.plt
plot/1 2 max=4 ns=64
pass = pass + 1
plot/watch ex77e_@pass.plt
plot/1 2 xrad=B/2 xdec=-B/2 ydec=A/2
L = 8
mac ex77e/4

```

Ex77f: Waveguide in a stable resonator

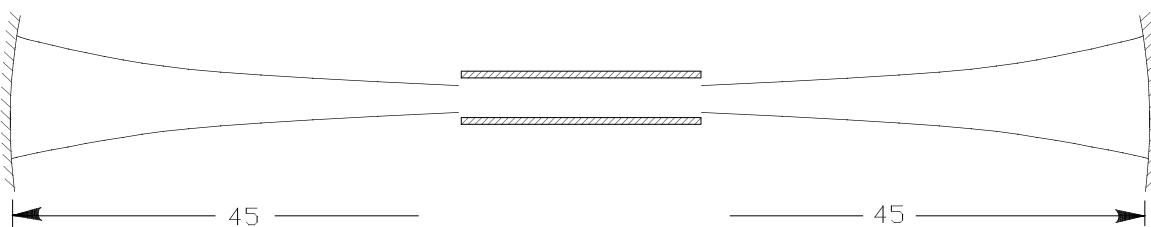


Fig. 77.34. Waveguide with an internal waveguide. The resonator ABCD properties are the same as if the waveguide had zero length.

Input: `ex77f.inp`

c## ex77f!52036906275057

Jump to: [Commands](#), [Theory](#)

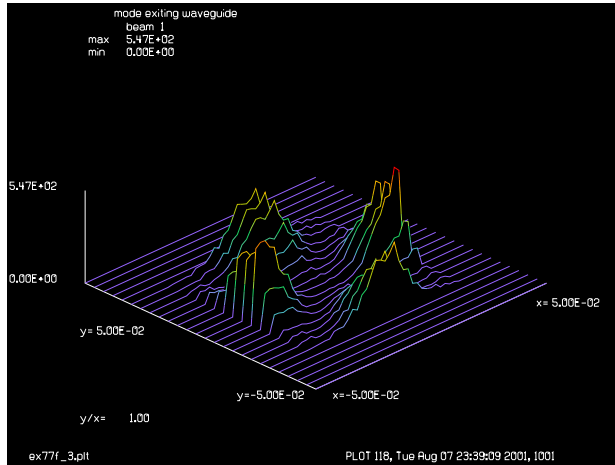


Fig. 77.35. Internal mode of resonator with internal waveguide. Ex 77f.inp.

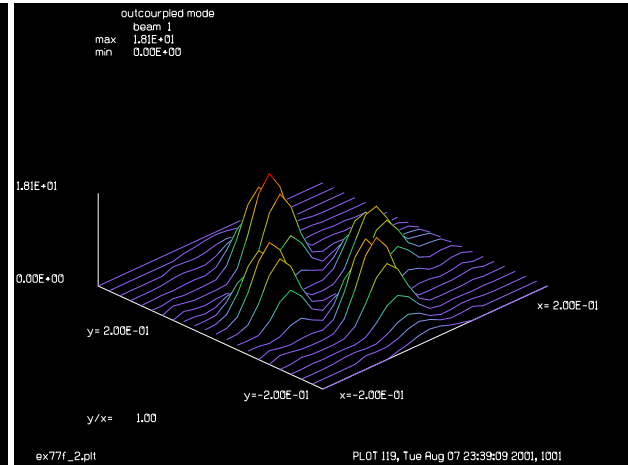


Fig. 77.36. Outcoupled mode from resonator with internal waveguide. Ex 77f.inp.

```
#
# Example of reflecting wall waveguide in a stable resonator.
#
# Treat the waveguide as having zero length when setting up the
# resonator. A symmetric cavity was selected, similar to the
# half-symmetric resonator of Ex33.
#
# In this example, the waveguide is modeled as a 32 x 32 array (beam 2)
# which, with units of 0.002 cm, has a size of .64 x .64 mm.
#
# The waveguide is implemented as in other Ex77 examples by copying
# the 32 x 32 array into a 64 x 64 array (beam 3) with imaging into
# the three other quadrants.
#
# At the end of the waveguide the upper left quadrant of beam 3 is
# copied back into beam 2 and then beam 2 is copied into the center
# of beam 1. The rest of the resonator follows.
#
# The waist sizes of beams 2 and 3 are set very large so these beams
# stay in the near-field. The reference position of beam 3 is reset
# each time to zero. Beam 2 is actually never propagated.
#
variab/dec/int count TEST
macro/def waveguide/o
  copy/con 1 2          # copy cavity mode array to waveguide array
  zreff/se 3 0
  clear 3 0             # make 2 x 2 array for waveguide propagation
  copy/con 2 3 -16 -16
  flip/x 2
  copy/con 2 3 16 -16
  flip/y 2
  copy/con 2 3 16 16
  flip/x 2
  copy/con 2 3 -16 16    # copy back to 32 x 32 array
  prop WaveguideLength 3
```

Jump to: [Commands](#), [Theory](#)

```

    copy/con 3 2 16 16
    clear 1 0
    copy/con 2 1                                # copy back to cavity mode array
macro/end
macro/def reson/o
    if [!TEST] count = count + 1
    mirror/sph 1 -50                            # mirror of 50 cm. radius
    clap/c/n 1 Apt                             # radius aperture
    if TEST then
        prop 90 1                             # propagate 45 cm. along beam
    else
        prop 45 1
        macro waveguide
        prop 45 1
    endif
    mirror/sph 1 50                             # flat mirror
    clap/c/n 1 .14                             # .14 cm. radius aperture
    if TEST then
        prop 90 1                             # propagate 45 cm.
    else
        prop 45 1
        macro waveguide
        title mode exiting waveguide
        plot/w ex77f_3.plt
        plot/l 1 xrad=.05
        prop 45 1
    endif
    clap/c/n 1 Apt                             # radius aperture
    if [!TEST] then
        variab/set Energy 1 energy
        Energy = 1 - Energy
        udata/set count count Energy
        title outcoupled mode
        plot/w ex77f_2.plt
        plot/l 1 xrad=.2
        plot/w ex77f_1.plt
        plot/udata
    endif
    energy/norm 1 1                             # renormalize energy
macro/end
array/s 1 128                                # cavity mode
nbeam 2 32                                  # waveguide of precisely 1/4, cavity mode array size
nbeam 3 64                                  # 2 x 2 array for waveguide propagation
wavelength/set 0 1.064                      # set wavelengths
gaus/c 1 1 .0225                            # guess at mode size
units/s 0 .002
prop 45 1                                    # propagate forward to start
geodata/set 2 waistx=10 waisty=10          #
geodata/set 3 waistx=10 waisty=10
Apt = .2                                    # aperture for resonator mirrors
WaveguideLength = 30                       # length of waveguide
TEST = 1
resonator/name reson
resonator/eigen/test 1

```

Jump to: [Commands](#), [Theory](#)


```

resonator/eigen/set 1          # set beam 2 to eigen mode
clear 1 1                     # start with a plane wave in beam 2
energy/norm 1 1               # normalize energies
reson/run 1
TEST = 0
reson/run 40

```

Ex77g: Half waveguide in unstable resonator

This example demonstrates an unstable resonator with a half waveguide, consisting of upper and lower mirror walls. Figs. 77.35 and 77.36 illustrate the top and side views. Parameters of the system are defined in Table 77.2. The total length of the resonator is 86.2 cm and the radii of the two mirrors are 92.03 cm and 81.07 cm. The stability parameter based on the g-parameters:

$$g_1 g_2 = \left(1 - \frac{L}{R_1}\right) \left(1 - \frac{L}{R_2}\right) = -0.004 \quad (77.1)$$

The negative value indicates that the resonator is a negative branch unstable resonator due to the relatively strong curvature of mirror 2. In GLAD we used the more general ABCD-based stability criterion of $|A + D|/2 < 1$ which also indicates that this resonator is slightly unstable. The ABCD stability parameter is -1.008. The magnification is 1.134 per pass for the expanding mode.

Outcoupling is accomplished by shifting the aperture at Mirror 1, so that the left side is exposed. However loss will still occur at all areas in a semi-annular ring because of the expansion of the beam in the unstable resonator. Hence, the device will not be ideally efficient. Some tilt might be used so the mode slides toward the left side as it expands so that there is very little light leaking out the right side.

The half waveguide consists of horizontal mirrors above and below the beam separated by 0.2 cm. The waveguide is most easily implemented in a collimated geometry with plane reference surfaces. In this case, the curved mirror may be correctly modeled as a flat mirror and the appropriate amount of focusing optical power. We could have put both the converging power of the mirrors into the model as flat mirrors plus optical power, but in the examples, we made the mirrors cylindrical using the normal mirror properties to allow the growth of the beam because of the unstable resonator properties in the x-direction. The `abr/ast` command is used to implement the optical power in the y-direction in combination with the collimated geometry.

Example 77g.inp illustrates the resonator without the waveguide and with centered apertures. Cylindrical mirrors are used to define the unstable resonator in the x-direction and astigmatic power establishes the optical power in the y-direction. This example serves to demonstrate that the flat mirror and astigmatic power is, indeed, numerically comparable to the curved mirror case as the x- and y-directions are shown to be very close in performance. Fig. 77.386 shows the cavity mode after 30 passes. Fig. 77.39 shows the outcoupled beam after 30 passes. The apertures were centered in this examples. Fig. 77.40 shows the width of the mode in the x- and y-directions. We see that flat mirror plus astigmatic power in the y-direction and the cylindrical mirror in the x-direction are, indeed, comparable.

Input: ex77g.inp

```

c## ex77g
#

```

Jump to: [Commands](#), [Theory](#)

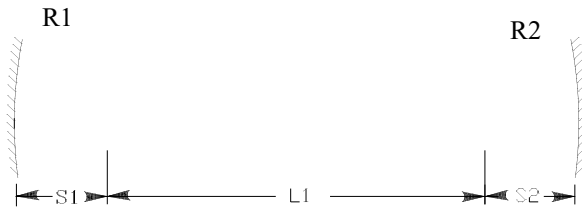


Fig. 77.37. Resonator looking from the top appears as a simple two-mirror system.

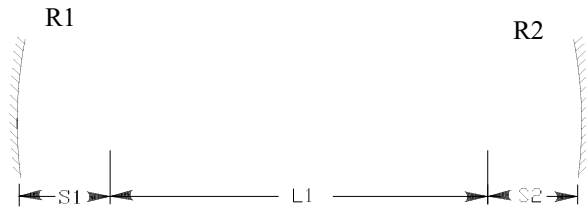


Fig. 77.38. Resonator viewed from the side. The waveguide is formed by top and bottom reflecting walls. The radii of the mirrors are identical in both directions, but in the vertical plane are formed by flat mirrors plus positive power.

Table. 77.2. Parameters used in the calculation.

Parameter	Value
waveguide	10.6 micron
mirror-to-waveguide, S1	1.9 cm
length of waveguide, L1	82.4 cm
total length of resonator, L2	86.2 cm
waveguide-to-mirror, S2	1.9 cm
thickness of waveguide	0.2 cm
R1	92.03 cm
R2	81.07 cm
stability parameter	-0.004, unstable
$g_1 g_2 = \left(1 - \frac{L}{R_1}\right) \left(1 - \frac{L}{R_2}\right)$	
aperture diameter, W1	4.4 cm
offset of first aperture, W2	0.56 cm

```
# Example of unstable laser. The mirrors are spherical but are
# represented in the model by cylindrical mirrors and astigmatic
# optical power in the y-direction. Comparison of the x- and
# y-directions shows that the beam size is, indeed, the same.
#
alias Name ex77g
variab/dec/int count
Nline = 1024
mem/set/b 5*8
set/alias/off
macro/def reson/o
    count = count + 1
    prop L2 1                                # propagate 45 cm. along beam
    mirror/xcyl 1 R2                         # flat mirror
    abr/ast 1 Waves2 rnorm=1
```

Jump to: [Commands](#), [Theory](#)

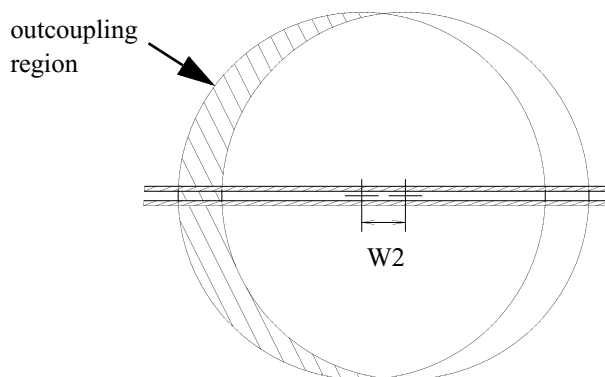


Fig. 77.39. Outcoupling occurs on the left side because the aperture associated with mirror 1 is shifted by $W2$.

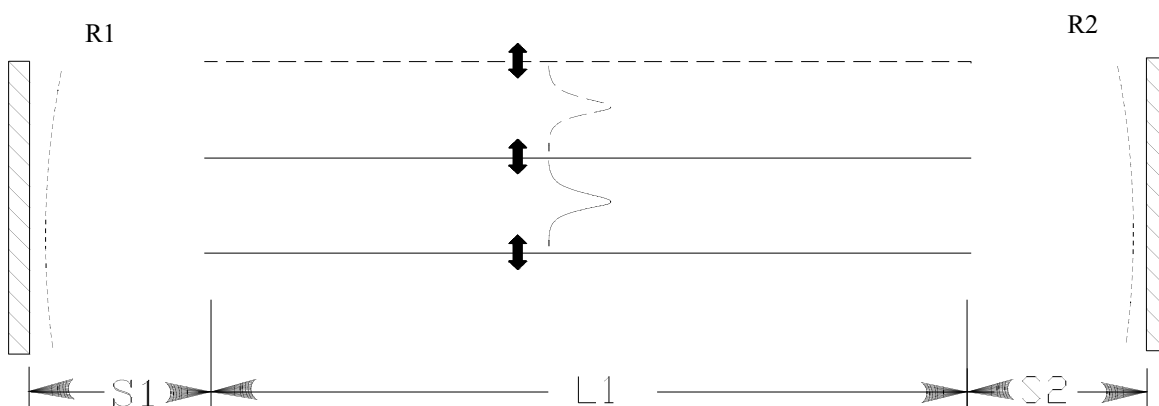


Fig. 77.40. Resonator viewed from the side. The waveguide is formed by top and bottom reflecting walls. The waveguide is modeled by creating a temporary array that contains both the internal distribution and its image. The aliasing properties of the numerical FFT will exactly model the effect of reflection from the walls.

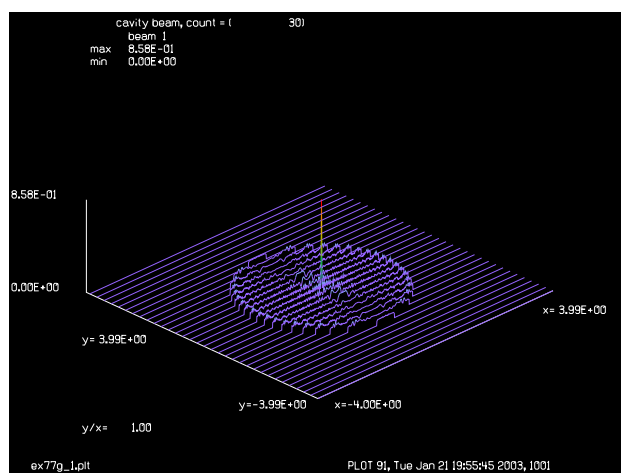


Fig. 77.41. Cavity mode for Ex77ga.inp after 30 passes.

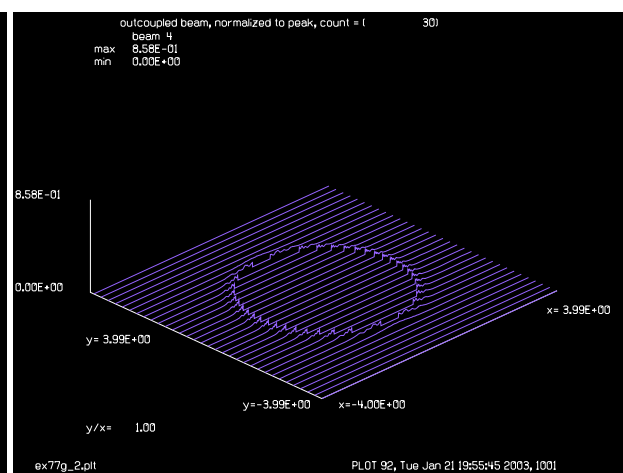


Fig. 77.42. Outcoupled mode for Ex77ga.inp after 30 passes.

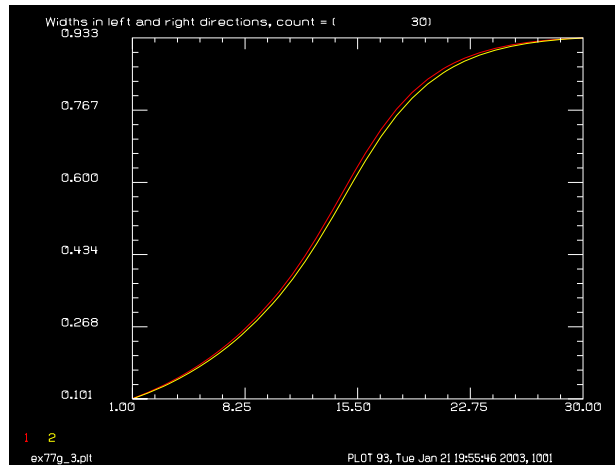


Fig. 77.43. Comparison of average radius in the x-(red) and y-direction (yellow) showing the flat mirror plus astigmatic power in the y-direction is essentially the same as the cylindrical mirror model in the x-direction.

```

clap/c/n 1 Apt                # radius aperture
prop L2 1                    # propagate 45 cm.
mirror/xcyl 1 R1             # mirror of 50 cm. radius
rescale/field 1 Field Field
copy 1 4
variab/set Peak 1 peak
clap/c/n 1 Apt xdec=Xdec     # radius aperture
obs/c 4 Apt xdec=Xdec
abr/ast 1 Waves1 rnorm=1
plot/w @Name_1.plt
title cavity beam, count = @count
plot/l 1 max=Peak
plot/w @Name_2.plt
title outcoupled beam, normalized to peak, count = @count
plot/l 4 max=Peak
fitgeo 1
variab/set Xrad 1 fitxrad
variab/set Yrad 1 fityrad
udata/set count count Xrad Yrad
plot/w @Name_3.plt
title Widths in left and right directions, count = @count
plot/udata 1 2
energy/norm 1 1              # renormalize energy
macro/end
array/s 1 Nline              # cavity mode
nbeam 2 Nline Nline/4        # waveguide of precisely 1/4, cavity mode array size
nbeam 3 Nline Nline/2        # 1 x 2 array for waveguide propagation
nbeam 4 Nline Nline          # outcoupled beam
Lambda = .00106              # wavelength
R1 = 92.03
R2 = -81.07
Waves1 = 1/R1/Lambda
Waves2 = -1/R2/Lambda
W1 = 4.4                     # aperture for resonator mirrors
W2 = .56

```

Jump to: [Commands](#), [Theory](#)

```

Apt = W1/2
Xdec = W2          # normal decenter
Xdec = 0           # set decenter to zero
S1 = 1.9
S2 = 1.9
L1 = 82.4          # length of waveguide
L2 = L1+S1+S2
WaveguideLength = L1      # length of waveguide
Field = 4
wavelength/set 0 Lambda*1e4  # set wavelengths
units/field 0 Field Field
gaus/c/c 1 1 .225          # guess at mode size
geodata/set/waist 1 waistx=100 waisty=100      #
geodata/set/waist 2 waistx=100 waisty=100      #
geodata/set/waist 3 waistx=100 waisty=100
TEST = 1
echo/on
resonator/name reson
resonator/eigen/test 1
resonator/eigen/set 1          # set beam 2 to eigen mode
count = 0
nbeam 5 Nline Nline
gaus/c 5 1 .225          # guess at mode size
copy/con/c 5 1
nbeam 4
reson/run 30

```

Ex77h: Resonator with waveguide in place

Example 77h.inp illustrates the resonator with the waveguide in place. Cylindrical mirrors are combined with astigmatic power so the resonator appears to be a standard resonator with two curved mirrors in the x-direction and two flat mirrors with converging astigmatic power in the y-direction. The waveguide is formed of two horizontal mirrors above and below the cavity mode separated by 2 cm. The propagating beam is Beam 1. Beam 1 is set to 1024 x 1024 with a half-width of 4 cm. Beam 2 is set to 1024 x 256 with a half-width of 1 cm such that Beam 2 exactly covers the width of the wave guide in the y-direction. Beam 1 is copied to Beam 2 for waveguide propagation. Beam 3 is set to 1024 x 512 and a half-width of 2 cm such that Beam 3 is twice the width of Beam 2 in the vertical direction. Beam 3 is created by copying Beam 2 to the upper half plane and Beam 2 with a vertical flip and 180 deg. phase change to the bottom half-plane. Propagation of Beam 3 will cause aliasing to exactly create the effects of reflecting upper and lower walls. Beam 3 is copied back to Beam 2 at the end of propagation through the waveguide to exactly clip the beam to the waveguide format. In the last step, Beam 1 is cleared and Beam 2 is copied into Beam 1 to recreate the cavity mode at the end of the waveguide.

Input: ex77h.inp

```

c## ex77h
#
# Example of reflecting wall waveguide in a resonator.
#
# Cylindrical mirrors are combined with astigmatic power so the
# resonator appears to be a standard resonator with two curved

```

Jump to: [Commands](#), [Theory](#)

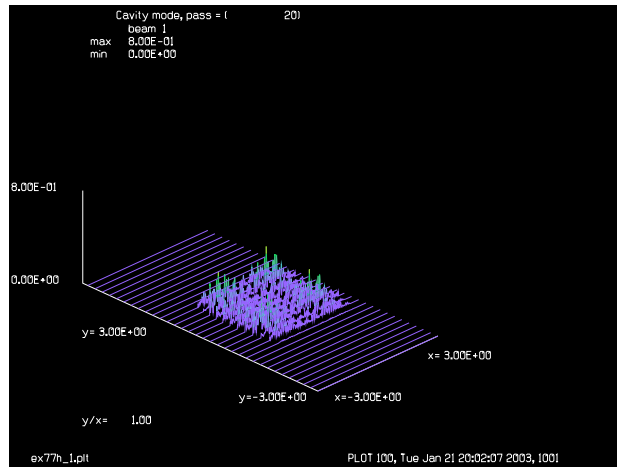


Fig. 77.44. Cavity mode for Ex77gb.inp.

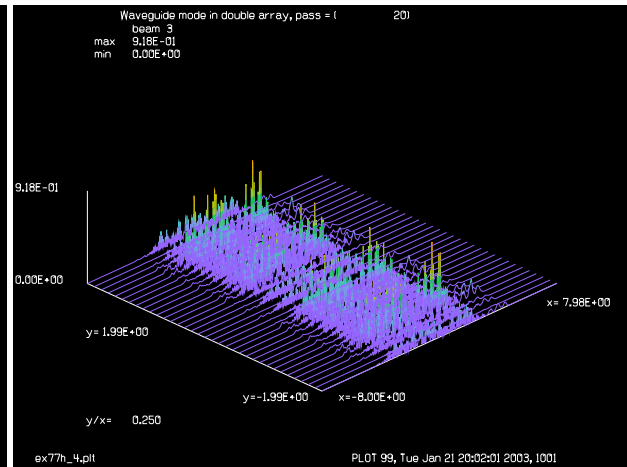


Fig. 77.45. Double waveguide mode showing two halves simulating the waveguide by aliasing.

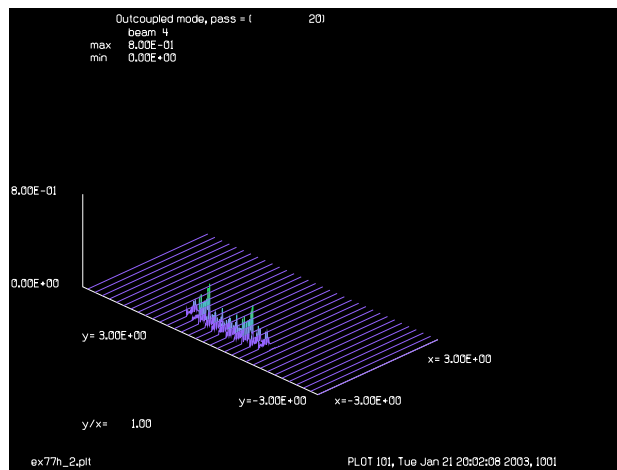


Fig. 77.46. Outcoupled beam.

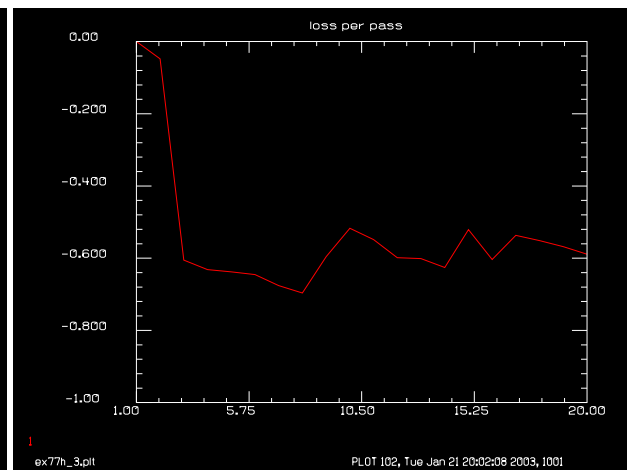


Fig. 77.47. Loss per step.

```
# mirrors in the x-direction and two flat mirrors with converging
# astigmatic power in the y-direction.
#
# The waveguide is formed of two horizontal mirrors above and below
# the cavity mode separated by 2 cm. The propagating beam is Beam 1.
#
# Beam 1 is set to 1024 x 1024 with a half-width of 4 cm.
#
# Beam 2 is set to 1024 x 256 with a half-width of 1 cm such that
# Beam 2 exactly covers the width of the wave guide in the y-direction.
# Beam 1 is copied to Beam 2 for waveguide propagation.
#
# Beam 3 is set to 1024 x 512 and a half-width of 2 cm such that Beam 3
# is twice the width of Beam 2 in the vertical direction. Beam 3 is created
# by copying Beam 2 to the upper half plane and Beam 2 with a vertical
# flip and 180 deg. phase change to the bottom half-plane. Propagation
# of Beam 3 will cause aliasing to exactly create the effects of
```

Jump to: [Commands](#), [Theory](#)

```

# of reflecting upper and lower walls. Beam 3 is copied back to Beam 2
# at the end of propagation through the waveguide to exactly clip the
# beam to the waveguide format. In the last step, Beam 1 is cleared and
# and Beam 2 is copied into Beam 1 to recreate the cavity mode at the
# end of the waveguide.
#
alias Name ex77h
variab/dec/int count TEST
Nline = 1024
mem/set/b 5*8
set/alias/off
macro/def waveguide/o
    clap/r/n 1 1000 1          # aperture at start of waveguide
    copy/con 1 2              # copy cavity mode array to waveguide array
    zreff/se 3 0              # reset z-coordinate to zero each time
    clear 3 0                 # make 1 x 2 array for waveguide propagation
    copy/con 2 3 0 Nline/8    # copy waveguide mode to top of double array
    flip/y 2                  # flip the waveguide mode vertically
    phase/piston 2 180        # 180 deg phase change models reflection effect
    copy/con 2 3 0 -Nline/8   # copy waveguide mode to bottom of double array
    prop L1 3                 # propagate in waveguide
    plot/w @Name_4.plt
    title Waveguide mode in double array, pass = @count
    plot/l 3
    copy/con 3 2 0 -Nline/8   # copy double array to waveguide array
    clear 1 0                 # zero the cavity mode
    copy/con 2 1              # copy waveguide mode to cavity array
    clap/r/n 1 1000 1        # slit aperture at end of waveguide
macro/end
macro/def reson/o
    if [TEST==0] count = count + 1
    if TEST then
        prop L2 1             # propagate in free-space for test
    else
        prop S1 1
        macro waveguide       # propagate in waveguide
        prop S2 1
    endif
    mirror/xcyl 1 R2          # cylindrical mirror
    abr/ast 1 Waves2 rnorm=1  # converging power for y-direction
    clap/c/n 1 Apt           # radius aperture
    if TEST then
        prop L2 1             # propagate in free-space for test
    else
        prop S2 1
        macro waveguide       # propagate in waveguide
        prop S1 1
    endif
    mirror/xcyl 1 R1          # mirror 1
    rescale/field 1 Fieldx Fieldy # rescale for unstable resonator
    abr/ast 1 Waves1 rnorm=1  # converging power for y-direction
    copy 1 4                  # form outcoupled mode in Beam 4
    variab/set Peak 1 peak    # find peak of cavity mode
    clap/c/n 1 Apt xdec=Xdec  # decenter aperture for outcoupling

```

Jump to: [Commands](#), [Theory](#)

```

obs/c 4 Apt xdec=Xdec          # obscuration for outcoupled mode
plot/w @Name_1.plt
title Cavity mode, pass = @count
plot/l 1 1 max=Peak xrad=3 yrad=3
plot/w @Name_2.plt
title Outcoupled mode, pass = @count
plot/l 4 4 max=Peak xrad=3 yrad=3
if [TEST==0] then
    variab/set Energy 1 energy
    udata/set count count Energy-1
    plot/w @Name_3.plt
    title loss per pass
    plot/udata max=0 min=-1
endif
energy/norm 1 1                # renormalize energy
macro/end
array/s 1 Nline                # cavity mode
nbeam 2 Nline Nline/4          # waveguide of precisely 1/4, cavity mode array size
nbeam 3 Nline Nline/2          # 1 x 2 array for waveguide propagation
nbeam 4 Nline Nline            # outcoupled beam
Lambda = .00106                # wavelength
R1 = 92.03                     # radius for mirror 1
R2 = -81.07                    # radius for mirror 2
Waves1 = 1/R1/Lambda           # astigmatic optical power for mirror 1
Waves2 = -1/R2/Lambda          # astigmatic optical power for mirror 2
W1 = 4.4                       # aperture for resonator mirrors
W2 = .56                       # decenter of aperture
Apt = W1/2
Xdec = W2
S1 = 1.9
S2 = 1.9
L1 = 82.4                      # length of waveguide
L2 = L1+S1+S2
WaveguideLength = L1           # length of waveguide
Fieldx = 8
Fieldy = 4
wavelength/set 0 Lambda*1e4    # set wavelengths
units/field 1 Fieldx Fieldy
units/field 2 Fieldx Fieldy/4
units/field 3 Fieldx Fieldy/2
units/field 4 Fieldx Fieldy
gaus/c/c 1 1 .4                # guess at mode size
geodata/set/waist 1 waistx=100 waisty=100 # large apertures to stay
geodata/set/waist 2 waistx=100 waisty=100 # in near-field
geodata/set/waist 3 waistx=100 waisty=100
TEST = 1
resonator/name reson
resonator/eigen/test 1
nbeam 5 Nline Nline
gaus/c 5 1 .225                # guess at mode size
copy/con 5 1
nbeam 4
TEST = 0
energy/norm 1 1

```

Jump to: [Commands](#), [Theory](#)

reson/run 20

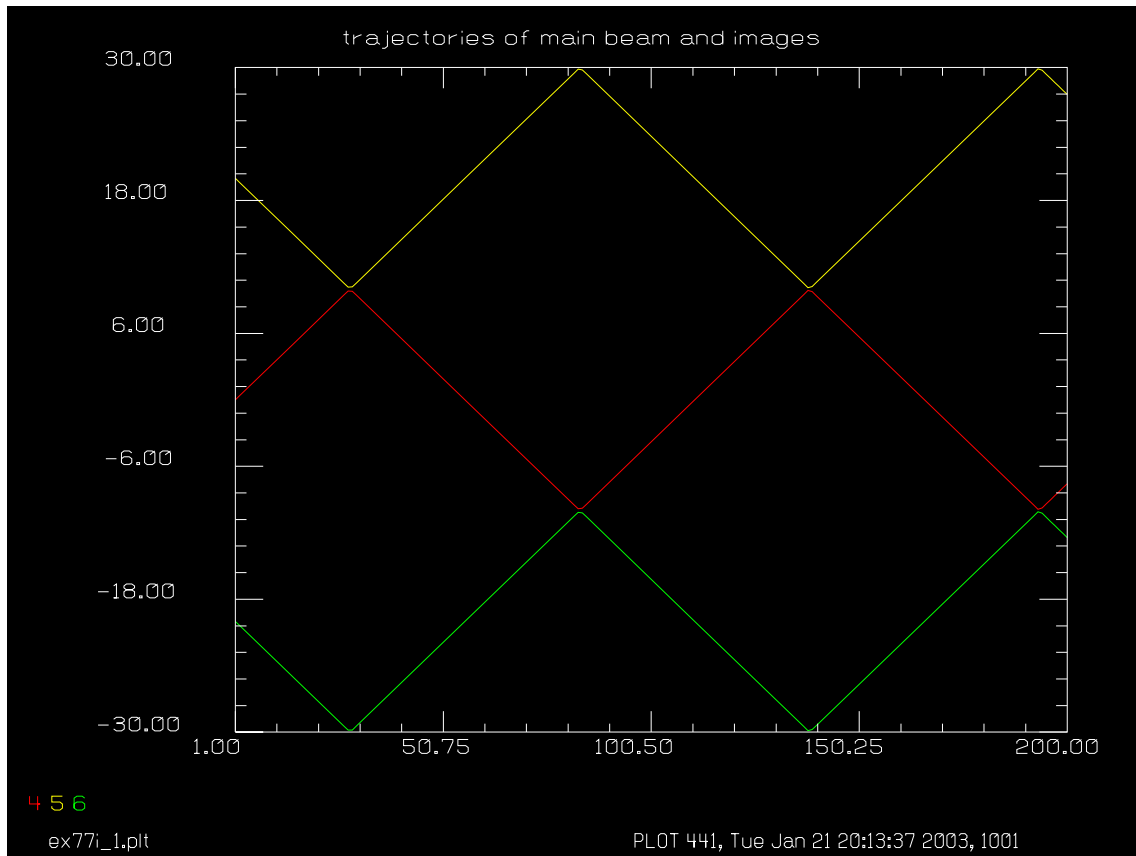
Ex77i: Incoherent treatment of reflecting-wall waveguide

Fig. 77.48. Trajectory of center ray of main beam (red) and that of incoherent images (yellow and green) for ex77i, resulting in a correct model of incoherent interaction in a two-mirror reflecting wall waveguide.

Input: ex77i.inp

```

c## ex77i
#
# Example 77i: incoherent treatment of waveguide
#
# This example treats reflecting wall waveguides where
# the incidence angle is sufficiently large that the
# self-interference fringes may be ignored as they are of
# too fine scale.
#
# The incoherent overlap of the beams will enable proper treatment
# of gain saturation if "gain/sheet" is used.
#
# In this example the waveguide consists of two mirrors located
# on the right and left sides at  $x = \text{Width}/2$  and  $x = -\text{Width}/2$  respectively.
# The vertical direction is left open.
#
# The method is to calculate the trajectory of the beam and the

```

Jump to: [Commands](#), [Theory](#)

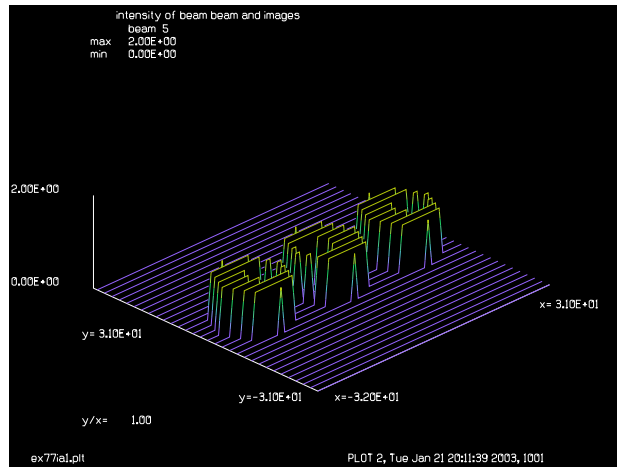


Fig. 77.49. Parent system for incoherent treatment of two-mirror waveguide. Starting condition.

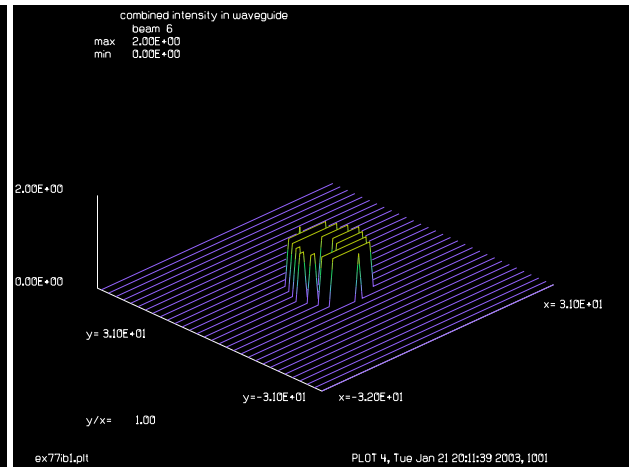


Fig. 77.50. Beam distribution derived from parent system. Starting condition.

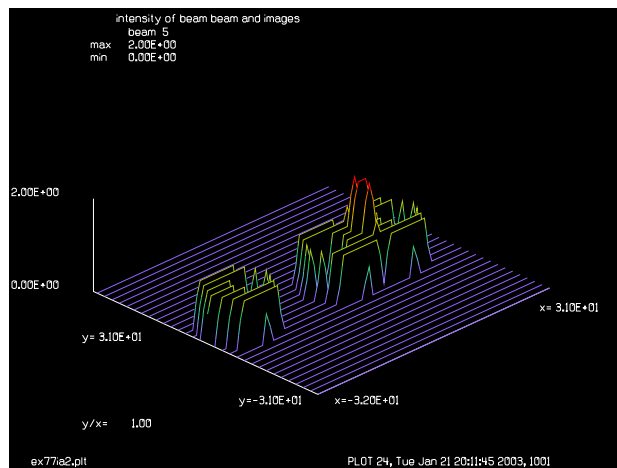


Fig. 77.51. Beam is angled to the left and begins to intercept mirror on right.

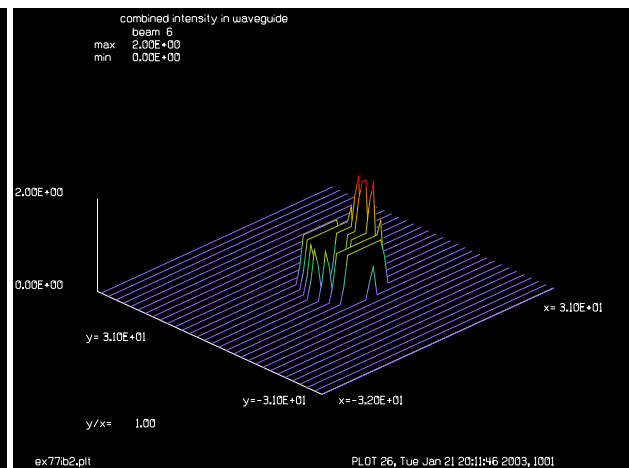


Fig. 77.52. Beam begins to overlap with itself.

```
# trajectories of the images from the right and left mirrors.
# The combined intensity is formed by the incoherent sum of
# the main beam and the two images.
#
set/highlight/off
variab/dec/int count count1 Nflip
Width = 20                                # full width of reflecting wall waveguide
HalfWidth = Width/2                       # half-width of waveguide
Angle = 20                                # angle in degrees
s = tan(Angle*pi/180)                     # slope of beam
DeltaZ = 1.                               # incremental z-displacement
nbeam 6 data
clap/c/n 1 8
obs/c 1 3 xdec=-3
macro/def step/o
    count = count + 1
```

Jump to: [Commands](#), [Theory](#)

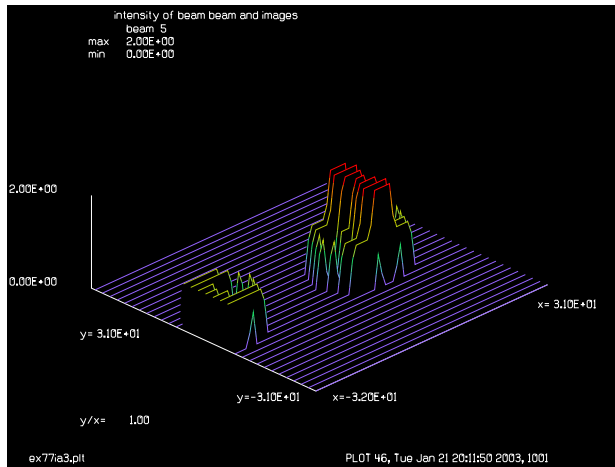


Fig. 77.53. Parent system shows collision with wall.

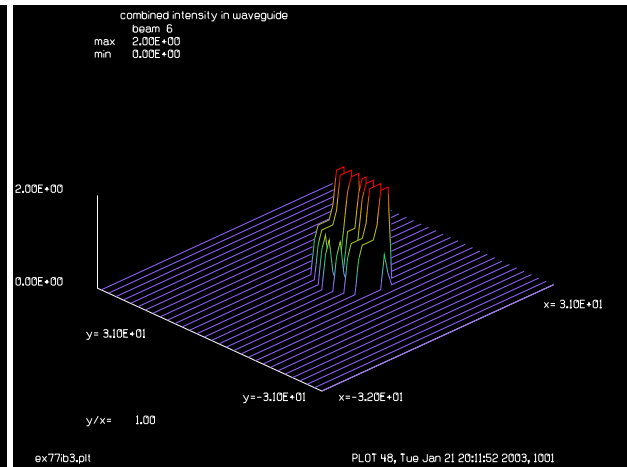


Fig. 77.54. Beam colliding with wall.

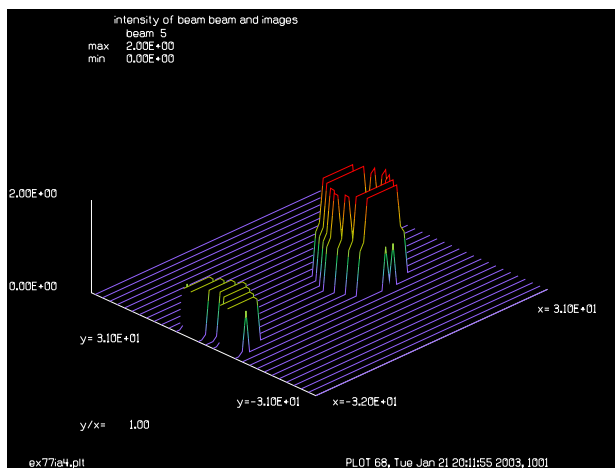


Fig. 77.55. Parent system. Beam proceeds to right.

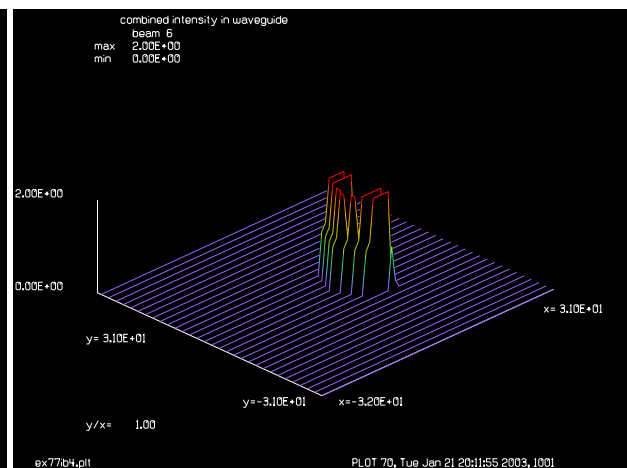


Fig. 77.56. Beam proceeds to right.

```

prop DeltaZ/cos(Angle*pi/180) # propagate along slant distance
Z = (count-1)*DeltaZ        # Z-position
x = Z*s                      # unfolded x-location of beam
#
# compute trajectories of beam 1 and images
#
x1 = x + HalfWidth          # shift coordinates to left side
x2 = mod(x1,Width*2)
x3 = x2/Width/2
if [x3<.5] then
    y = x2
    Nflip = 0
else
    y = 2.*Width - x2
    Nflip = 1
endif
right = 2*Width - y
left = - y

```

Jump to: [Commands](#), [Theory](#)

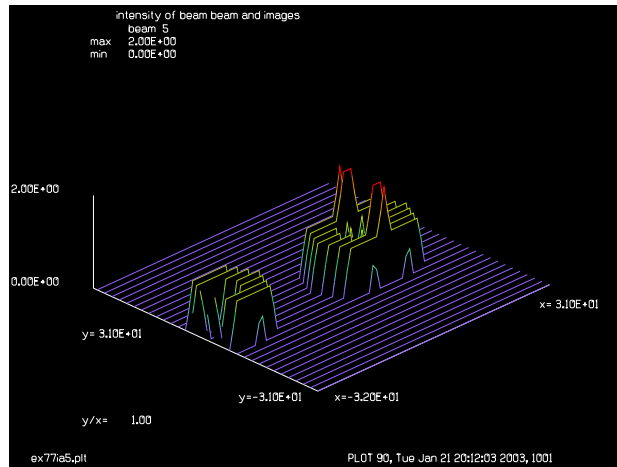


Fig. 77.57. Parent system. Beam begins to return from right.

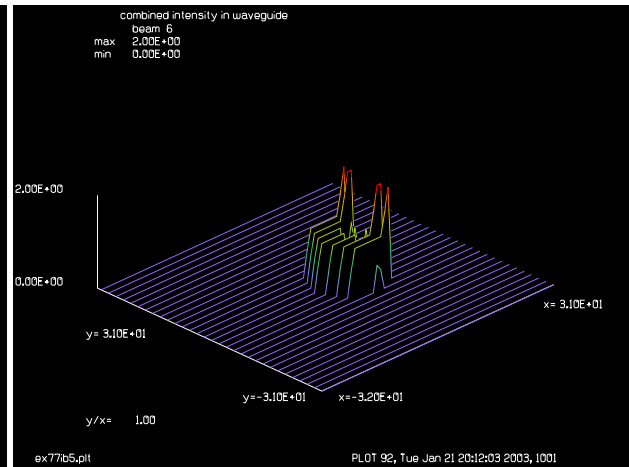


Fig. 77.58. Beam begins to return from right.

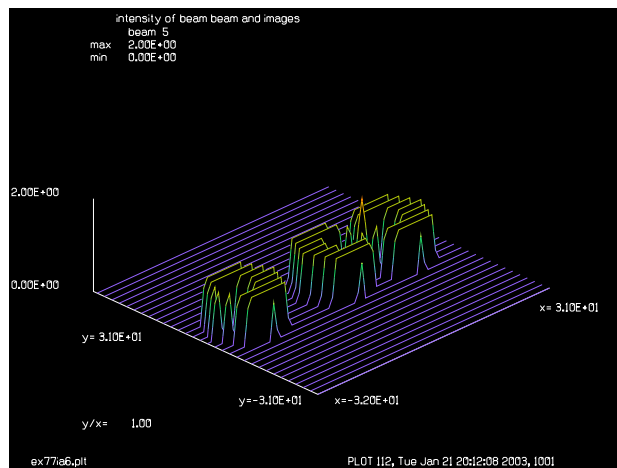


Fig. 77.59. Parent system. Beam continues to return from right.

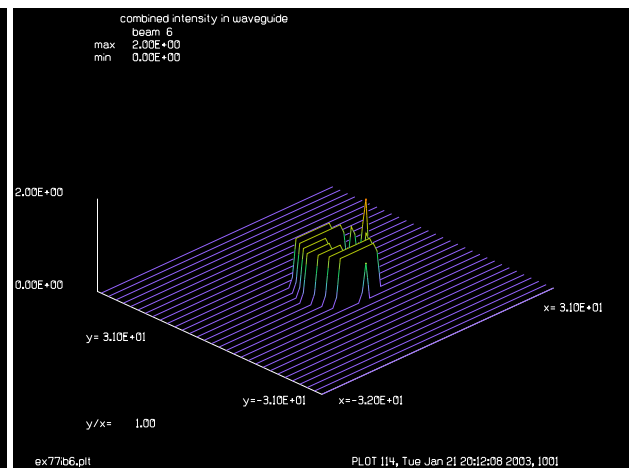


Fig. 77.60. Beam continues to return from right.

```

y = y - HalfWidth          # shift back to centered coordinates
right = right - HalfWidth   # shift back to centered coordinates
left = left - HalfWidth     # shift back to centered coordinates
udata/set count count x x1 x2 y right left
# form center
copy/con 1 2
if [Nflip] flip/x 2
rescale/shift 2 y
# form right side
copy/con 1 3
if [!Nflip] flip/x 3
rescale/shift 3 right
# form left side
copy/con 1 4
if [!Nflip] flip/x 4
rescale/shift 4 left

```

Jump to: [Commands](#), [Theory](#)

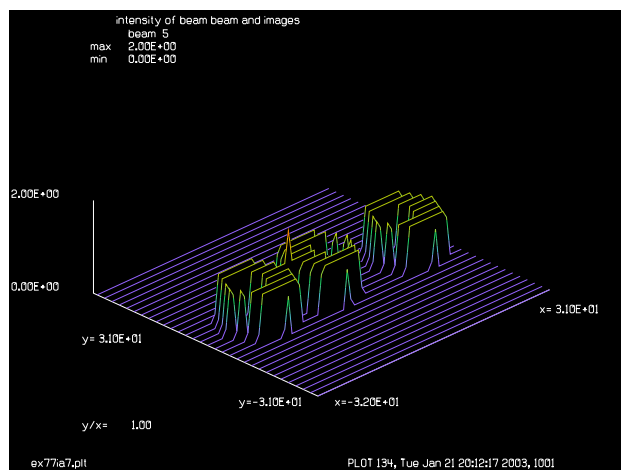


Fig. 77.61. Parent system. Beam heads to left wall.

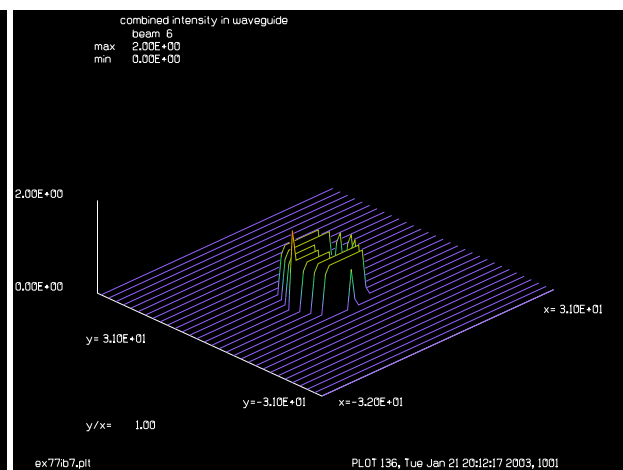


Fig. 77.62. Beam heads to left wall.

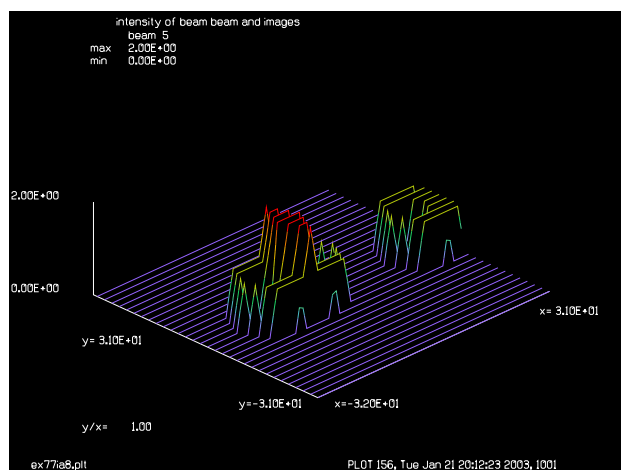


Fig. 77.63. Parent system. Beam strikes left wall.

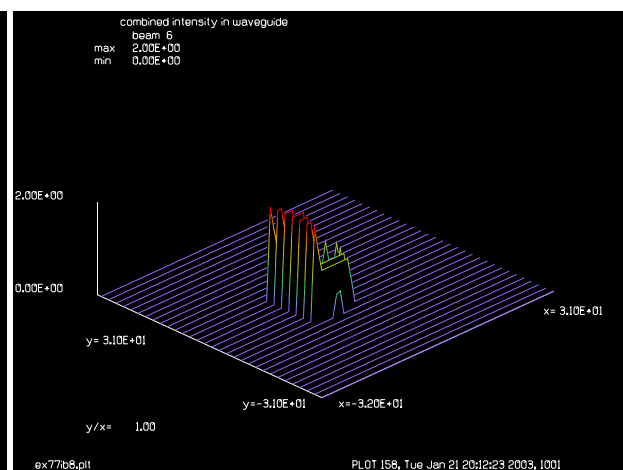


Fig. 77.64. Beam strikes left wall.

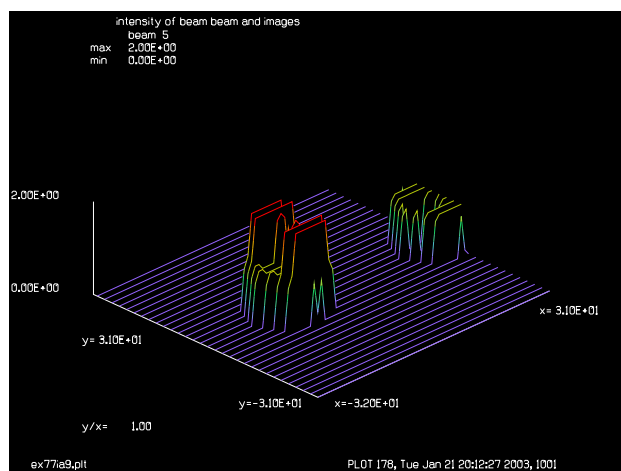


Fig. 77.65. Parent system. Beam is centered on left wall.

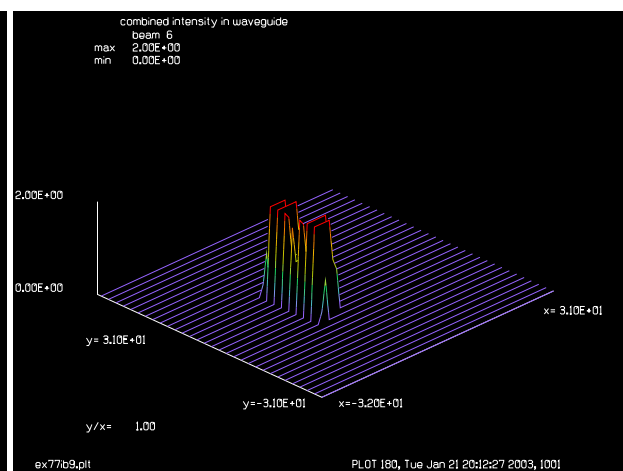


Fig. 77.66. Beam is centered on left wall.

```

clear 5 0
add/inc/c 5 2 3 4
plot/w plot1.plt
title intensity of beam beam and images
plot/l 5 max=2
if [mod(count-1,10)==0] then
    count1 = count1 + 1
    plot/d ex77ia@count1.plt
    plot/l 5 max=2
endif
copy/con 5 6
clap/sqr/n 6 HalfWidth
title combined intensity in waveguide
plot/w plot2.plt
plot/l 6 max=2
if [mod(count-1,10)==0] then
    plot/d ex77ib@count1.plt
    plot/l 6 max=2
endif
macro/end
macro step/200
plot/udata/set y04 y05 y06
plot/w ex77i_1.plt
title trajectories of main beam and images
plot/udata/seq min=-1.5*Width max=1.5*Width
udata

```

Ex77j: Incoherent treatment of reflecting-wall waveguide, converging beam

Input: ex77j.inp

```

c## ex77j
#
# Example 77j: incoherent treatment of waveguide, converging beam
#
# This example treats reflecting wall waveguides where
# the incidence angle is sufficiently large that the
# self-interference fringes may be ignored as they are of
# too fine scale.
#
# The incoherent overlap of the beams will enable proper treatment
# of gain saturation if "gain/sheet" is used.
#
# In this example the waveguide consists of two mirrors located
# on the right and left sides at x=Width/2 and x=-Width/2 respectively.
# The vertical direction is left open.
#
# The method is to calculate the trajectory of the beam and the
# trajectories of the images from the right and left mirrors.
# The combined intensity is formed by the incoherent sum of
# the main beam and the two images.
#
# Use the method of images to "unfold" the system. The variable "x" is

```

Jump to: [Commands](#), [Theory](#)

```

# the unfolded shift.
#
# Beam 1 is in beam coordinates (unshifted).
# Beam 2 is shifted relative to walls.
# Beam 3 is reflection from right wall
# Beam 4 is reflection from left wall
# Beam 5 is intensity considering overlap
# Beam 6 is clipped to show beam between walls (what actually happens)
#
variab/dec/int count count1 Nflip
Width = 20                # full width of reflecting wall waveguide
HalfWidth = Width/2        # half-width of waveguide
Angle = 20                # angle in degrees
s = tan(Angle*pi/180)      # slope of beam
DeltaZ = 1.                # incremental z-displacement
FocalLength = 600
nbeam 6 data
clap/c/n 1 8
obs/c 1 3 xdec=-3          # add an obscuration so we can see flip of reflection
lens 1 FocalLength          # lens makes a convergent beam
macro/def step/o
    count = count + 1
    prop DeltaZ/cos(Angle*pi/180) # propagate along slant distance
    Z = (count-1)*DeltaZ          # Z-position
    x = Z*s                      # unfolded x-location of beam
#
# compute trajectories of beam 1 and images
#
    x1 = x + HalfWidth           # shift coordinates to left wall
    x2 = mod(x1,Width*2)         # figure which two-element block to use
    x3 = x2/Width/2              # which half of the two-element block are we in
    if [x3<.5] then
        y = x2
        Nflip = 0                # Logic to dictate flipping distribution
    else
        y = 2.*Width - x2
        Nflip = 1
    endif
    right = 2*Width - y
    left = - y
    y = y - HalfWidth            # shift back to centered coordinates
    right = right - HalfWidth    # shift back to centered coordinates
    left = left - HalfWidth      # shift back to centered coordinates
    udata/set count count x x1 x2 y right left
# form center
    copy 1 2
    if [Nflip] flip/x 2
    rescale/shift 2 y
# form right side
    copy 1 3
    if [!Nflip] flip/x 3
    rescale/shift 3 right
# form left side
    copy 1 4

```

Jump to: [Commands](#), [Theory](#)

```

if [!Nflip] flip/x 4
rescale/shift 4 left
clear 5 0
units/beam 5 2
add/inc/c 5 2 3 4
plot/w plot1.plt
title intensity of beam beam and images
Peak = 2*(FocalLength/(FocalLength-Z))^2
plot/l 5 max=Peak
if [mod(count-1,10)==0] then
    count1 = count1 + 1
    plot/d ex77ja@count1.plt
    plot/l 5 max=Peak
endif
copy 5 6
clap/rec/n 6 HalfWidth 1000
title combined intensity in waveguide
plot/w plot2.plt
plot/l 6 xrad=HalfWidth max=Peak
if [mod(count-1,10)==0] then
    plot/d ex77jb@count1.plt
    plot/l 6 xrad=HalfWidth max=Peak
endif
macro/end
macro step/256
plot/udata/set y04 y05 y06
plot/w ex77j_1.plt
title trajectories of main beam and images
plot/udata/seq min=-1.5*Width max=1.5*Width
udata

```


Ex78: Automatic optimization of resonator design

With the optimization option, the resonator command may be used to optimize all aspects of resonator design. This example illustrates predesign of a resonator to achieve a waist radius of 0.04 cm by adjusting the radii of the two end mirrors. The analysis proceeds in two phases. The first task is to use the `resonator/test` and `resonator/set` command to establish the size of the eigenmode for a given choice of resonator mirror. Both radii are the same, making the resonator symmetrical. The `resonator/test` command exercises the resonator defined in the macro EX78 and calculates the round-trip ABCD properties, from which the stability criteria and eigenmode are determined. The radius of the eigenmode is stored in real register 2, where it is accessible to the optimization target table.

Once the specified waist of the eigenmode is achieved. The diffraction transverse mode is calculated using the gaussian eigenmode as a starting point. Convergence is very rapid because the calculated eigenmode is close to the exact diffraction solution.

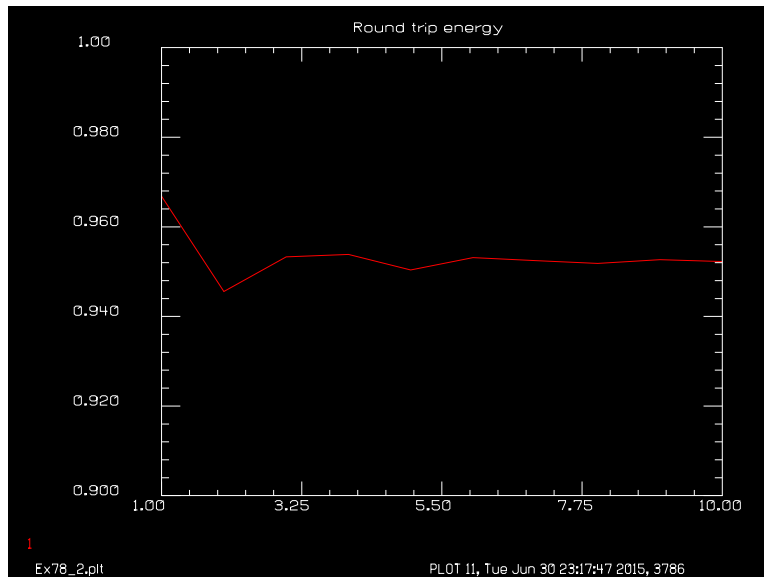


Fig. 78.1. Convergence of symmetrical resonator after injection of analytical eigenmode calculated from ABCD properties. The rapid convergence shows that the analytical eigenmode was a good choice.

Input: ex78.inp

```
c## ex78
c
c Example 78: Automatic optimization of resonator design
c
c This example illustrates optimization of the configuration of a
c resonator to achieve a specific waist size of .04 cm.
c
c The RESONATOR command is used to do the predesign of the laser.
c Once the optimization is complete, a Fox-Li analysis is used to
c establish the transverse mode.
c
alias Name Ex78
variab/dec/int pass done switch
```

```

switch = 0
wavelength/set 1 1.06
units/set 1 .01
macro/def ex78/o
c
c Optical configuration to be optimized
c
  energy/norm 1 1
  mirror 1 -radius
  dist -25
  mirror 1 radius
  dist 25
  clap/c/n 1 .06
  if switch = 1 then
c
c code to be switched in for Fox-Li analysis
c
    pass = pass + 1
    plot/1
    variab/set energy 1 energy
    udata/set pass pass energy
  endif
macro/end
radius = 100
resonator/name ex78          # name macro for resonator
macro/def optmac1/o
c
c Run resonator to setup paraxial properties
c
  resonator/eigen/test      # list properties
  variab/set ewaist 1 reson/ewaist list
macro/end
macro/def optmac/o
c
c Macro to control optimization
c
  pass = pass + 1
  opt/run
  variab/set done optdone    # check for optimization finished
  udata/set pass pass radius ewaist
  if done > 0 then
    macro/exit
  endif
macro/end
pass = 0
switch = 0
opt/name optmac1            # select macro to be optimized
opt/var/add radius          # make variable table
opt/tar/add ewaist .04      # make target table
c
c Run optimization macro to find desired eigen waist
c
write/off
plot/watch @Name_1.plt

```

Jump to: [Commands](#), [Theory](#)

```
title Resonator mode
macro optmac/5
write/on
udata/clabel 'radius' 2
udata/clabel 'ewaist' 3
C
C C optimize to ewaist = 0.04
C
udata/list
c
c Paraxial properties are now set.
c Set switch for Fox-Li operation.
c
switch = 1
udata/clear
udata/clabel 'energy' 2
pass = 0
write/off
resonator/run 10
write/on
plot/watch @Name_2.plt
title Round trip energy
plot/udata min=.9 max=1.
```


Ex79: Transient Raman

Table. 79.1. Table of Ex79 examples

Effect of wide angle noise, 64 x 64 array	3
Effect of wide angle noise, 256x 256 array	5
Transient behavior of the Raman process for a gaussian temporal waveform	7
Transient behavior of the Raman process with doubled irradiance	12
Effect of weak hot spots in the laser to create strong Stokes spikes	16

This example illustrates some effects of transient Raman physics. We assume that a high power laser pulse of short duration is to be propagated through a gas such as the atmosphere. The temporal pulse length may be much less than the time constant of the Raman process, although the equations will converge properly to the steady-state gain if a long pulse is used. The complete set of equations describing transient Raman physics are found in Chap. 9, Theory Manual.

A key issue in the propagation of very high power laser beams is the question of the angle of the spontaneous emission which needs to be treated. Finer spatial sampling allows a greater solid angle to be considered. We shall investigate arrays of 64 x 64 and 256 x 256 for a beam meeting the conditions shown in Table 79.1. In this example we consider only the first 10 picoseconds, so the transient gain is much less than the steady-state gain. For each size array we take 30 steps of 1 km through the medium, considering spontaneous Raman, stimulated Raman, and diffraction. Figures 1a and 1b show the growth of spontaneous emission for the case of no stimulated Raman amplification, in terms of the logarithm of the ratio of Stokes to laser pump power. Initially, the noise grows linearly according but after about 10 km for the 64 beam and after about 2.5 km for the 256 beam, the effect of light scattering out of the beam path causes the spontaneous emission to reach a constant value. The larger array has more wide angle spontaneous emission and the effect of scattering out of the beam path occurs at a comparably shorter distance.

Table. 79.2. Parameters used in transient Raman calculations, for Ex79a and Ex79b.

Parameter	Value
laser wavelength	0.3511 μ
Stokes wavelength	0.35204 μ
initial laser irradiance	1×10^8 w/cm ²
initial Stokes irradiance	0
laser pulse width	10 picoseconds
Raman bandwidth	2.4×10^9 sec ⁻¹
stimulated Raman gain g_1	3×10^{-12} cm ⁻¹ w ⁻¹
spontaneous Raman gain g_2	1×10^{-10} cm ⁻¹
beam diameter	30 cm
propagation step length	1 km

The Rayleigh range of the 30 cm diameter beam is about 200 km so the 30 km distance is well in the near-field. To observe the far-field characteristics of the beams a lens was added after the 30 km propagation. Figures 2a and 2b illustrate the far-field patterns for the small and large arrays. At this distance, only the very narrow angle spontaneous emission has had the maximum gain-length, so the Stokes beam is well confined in angle. The beams are about 5 to 10 times diffraction limited, as shown.

Figures 3a and 3b show the logarithm of the Stokes-to-pump ratio as a function of propagation distance. Initially, the large array has higher power because it covers a larger solid angle. However, for this case the wider angle scatter out of the beam path so that after about 10 km the Stokes beam is about the same power for both sizes of array.

This particular example was designed to illustrate the limits of the narrow-angle model. It illustrates that including wide angle spontaneous emission is very important for short propagation distances. The shorter the distance the larger the angles which must be considered. For low power laser beams the Raman threshold occurs at very long distances (10's of km in this example) and the modest angular requirements may be accommodated by reasonable sized arrays. High power beams with Raman thresholds on the order of 10's of meters, will require consideration of angles of many degrees—much more than can be treated by reasonable size arrays.

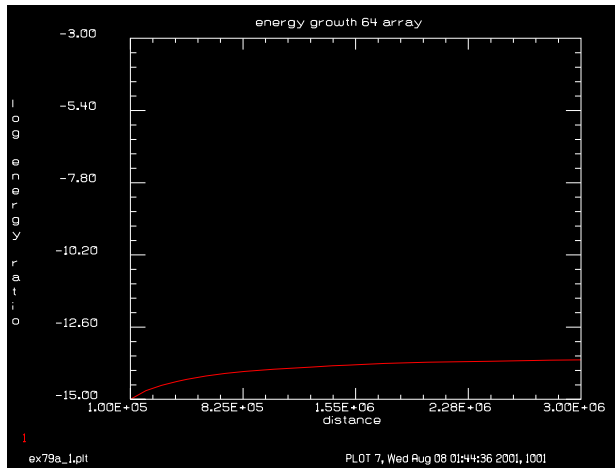


Fig. 79.1a. Spontaneous emission growth, assuming no Raman amplification, but scattering losses out of the beam (64 array).

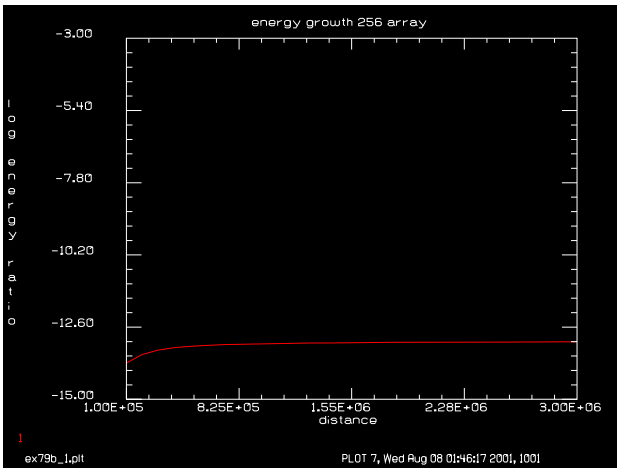


Fig. 79.1b. Spontaneous emission growth, assuming no Raman amplification, but scattering losses out of the beam (256 array).

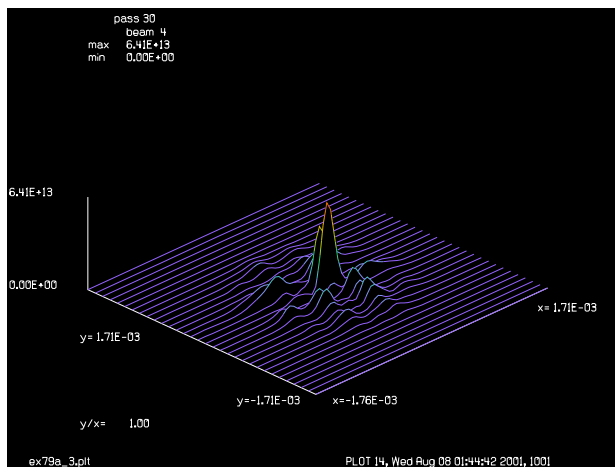


Fig. 79.2a. Far-field of Stokes beam at 30 km showing the beam is 5 to 10 times diffraction limited (64 array).

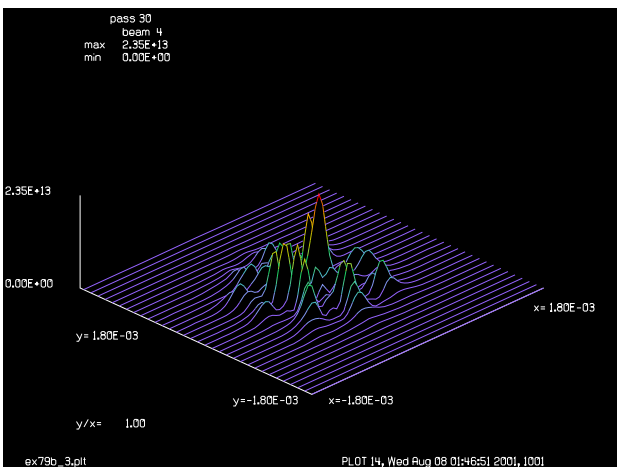


Fig. 79.2b. Far-field of Stokes beam at 30 km showing the beam is 5 to 10 times diffraction limited (256 array).

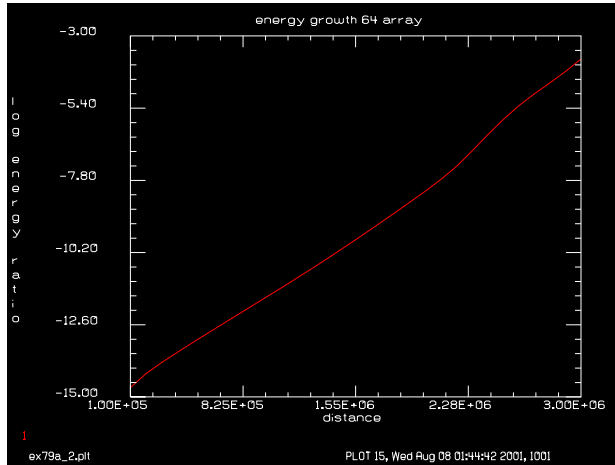


Fig. 79.3a. Energy growth over the 30 km path as a function of the log of the Stokes-to-laser ratio (64 array). Note the nearly exponential growth.

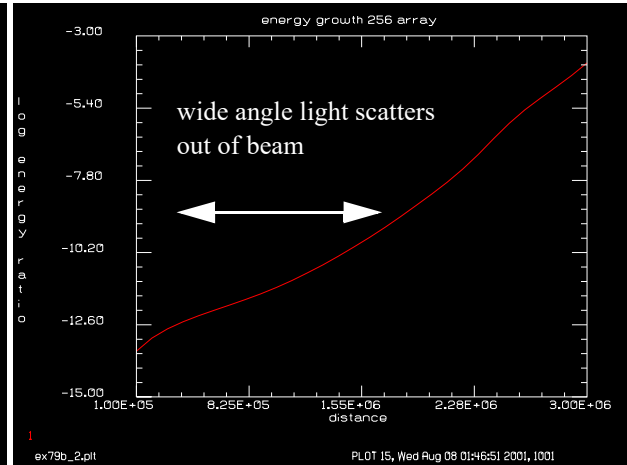


Fig. 79.3b. Energy growth over the 30 km path as a function of the log of the Stokes-to-laser ratio (64 array). Initially the Stokes is higher but the wide angle light scatters out giving the same value at 30 km.

Ex79a: Effect of wide angle noise, 64 x 64 array

Input: ex79a.inp

```
c## ex79a
c
c Example to illustrate the effect of wide angle noise
c on transient Raman growth
c
c In this example the array is 64 x 64, giving a
c a maximum noise angle of .352e-4 rad
c
variab/dec/int pass make_plot
array/s 1 64
nbeam 4
beams/off 4
array/s 3 64 64 1 data
units/s 0 1.
wavelength/set 1 .3511
wavelength/set 2 .35204
c
c pump is beam 1
c Stokes is beam 2
c stimulated gain is 3e-12 per cm-watt
c spontaneous gain is 1e-18 cm-1
c gamma is 2.4e9 sec-1
c
pack/set 1 2 3
c
c initial laser pump is 1 w/cm**2
c the laser pulse width is 10 picoseconds
c the transient gain length is 8.33e4
```

Jump to: [Commands](#), [Theory](#)

```

c
macro/def step/o
  pass = pass + 1
c
c instead of using 30 separate media screens, for 1 single
c time step, we can just reinitialize Beam 3 30 times
c
  raman/trans/rest 3          # initialize media array
  pack/in
    raman/trans/step 3 zstep 10e-12 # take Raman step
  pack/out
  prop zstep                  # diffraction step
  z = z + zstep
  clap/c/c 2 25              # aperture to clean off wide angle diffraction
  variab/set energy_now 2 energy list
  amplification = energy_now/energy_start
  log_amp = log10(amplification)
  udata/set pass z log_amp amplification
c
c make a plot every 5 passes
c
  make_plot = mod(pass,5)
  if make_plot = 0 then
    copy 2 4
    beams/on 4
    lens 4 100
    dist 100 4
    beams/off 4
    title pass @pass
    plot/l 4
  endif
macro/end
zstep = 1e5
c
c Step with negligible gain
c
clear 1 1e8
clap/c/c 1 15.              # aperture diameter is 30 cm
variab/set energy_start 1 energy
clear 2 0.
pass = 0
z = 0
gamma = 1.
raman/trans/set 1 2 3.e-12 1.e-10 gamma 11
title far-field for 64 array
macro step/30              # run 30 steps to get to 30 km
plot/watch ex79a_1.plt
title energy growth 64 array
plot/udata/xlabel distance
plot/udata/ylabel log energy ratio
plot/udata 1 1 min=-15 max=-3
plot/watch plot1.plt
c
c Repeat calculations with correct gain value

```

Jump to: [Commands](#), [Theory](#)


```

c
clear 1 1e8
clap/c/c 1 15.          # aperture diameter is 30 cm
clear 2 0.
pass = 0
z = 0
zreff/se 0 0.
udata/clear
gamma = 2.4e9
raman/trans/set 1 2 3.e-12 1.e-10 gamma 11
title far-field for 64 array
macro step/30          # run 30 steps to get to 30 km
plot/watch ex79a_3.plt
plot/l 4
plot/watch ex79a_2.plt
title energy growth 64 array
plot/udata/xlabel distance
plot/udata/ylabel log energy ratio
plot/udata 1 1 min=-15 max=-3
udata/list/disk ex79a.out
end

```

Ex79b: Effect of wide angle noise, 256x 256 array

Input: ex79b.inp

```

c## ex79b!556470256081842
c
c Example to illustrate the effect of wide angle noise
c on transient Raman growth
c
c In this example the array is 256 x 256, giving a
c a maximum noise angle of 1.408e-4 rad, 4 times the
c example of ex79a.inp
c
variab/dec/int pass make_plot
array/s 1 256
nbeam 4
beams/off 4
array/s 3 256 256 1 data
units/s 0 .25
wavelength/set 1 .3511
wavelength/set 2 .35204
c
c pump beam is 1
c Stokes beam is 2
c stimulated gain is 3e-12 per cm-watt
c spontaneous gain is 1e-10 cm-1
c gamma is 2.4e9 sec-1
c
pack/set 1 2 3
c

```

Jump to: [Commands](#), [Theory](#)

```

c  initial laser pump is 1 w/cm**2
c  the laser pulse width is 10 picoseconds
c  the transient gain length is 8.33e4
c
macro/def step/o
  pass = pass + 1
c
c  instead of using 30 separate media screens, for 1 single
c  time step, we can just reinitialize Beam 3 30 times
c
  raman/trans/rest 3          # initialize media array
  pack/in
    raman/trans/step 3 zstep 10e-12 # take Raman step
  pack/out
  prop zstep                  # diffraction step
  z = z + zstep
  clap/c/c 2 25              # aperture to clean off wide angle diffraction
  variab/set energy_now 2 energy list
  amplification = energy_now/energy_start
  log_amp = log10(amplification)
  udata/set pass z log_amp amplification
c
c  make a plot every 5 passes
c
  make_plot = mod(pass,5)
  if make_plot = 0 then
    copy 2 4
    beams/on 4
    lens 4 100
    dist 100 4
    beams/off 4
    title pass @pass
    plot/1 4 xrad=1.8e-3
  endif
macro/end
zstep = 1e5
c
c  Step with negligible gain
c
clear 1 1e8
clap/c/c 1 15.              # aperture diameter is 30 cm
variab/set energy_start 1 energy
clear 2 0.
pass = 0
z = 0
gamma = 1.
raman/trans/set 1 2 3.e-12 1.e-10 gamma 11
title far-field for 256 array
macro step/30              # run 30 steps to get to 30 km
plot/watch ex79b_1.plt
title energy growth 256 array
plot/udata/xlabel distance
plot/udata/ylabel log energy ratio
plot/udata 1 1 min=-15 max=-3

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch plot1.plt
c
c Repeat calculations with correct gain value
c
clear 1 1e8
clap/c/c 1 15.          # aperture diameter is 30 cm
clear 2 0.
pass = 0
z = 0
zreff/se 0 0.
udata/clear
gamma = 2.4e9
raman/trans/set 1 2 3.e-12 1.e-10 gamma 11
title far-field for 256 array
macro step/30          # run 30 steps to get to 30 km
plot/watch ex79b_3.plt
plot/1 4 xrad=1.8e-3
plot/watch ex79b_2.plt
title energy growth 256 array
plot/udata/xlabel distance
plot/udata/ylabel log energy ratio
plot/udata 1 1 min=-15 max=-3
udata/list/disk ex79b.out
end

```

Ex79c: Transient behavior of the Raman process for a gaussian temporal waveform

Example 79c illustrates the transient behavior of the Raman process for a gaussian temporal waveform for the laser input. Fig. 79.4 shows the waveform of both the laser and the generated Stokes beam. Since temporal change of the laser occurs in the exponent of the amplification equation, the time response of the Stokes beam is much narrower. The peak of the Stokes pulse also follows the laser pulse. Example 79d is exactly the same as Ex79c, except that the power level is 4 rather than 2 gigawatts per sq. cm.

Table. 79.3. Parameters used in transient Raman calculations, for Ex79a and Ex79b.

Parameter	Value
laser wavelength	0.3511 μ
Stokes wavelength	0.35204 μ

Table. 79.3. Parameters used in transient Raman calculations, for Ex79a and Ex79b.

Parameter	Value
initial laser irradiance	$1 \times 10^8 \text{ w/cm}^2$
initial Stokes irradiance	0
laser pulse width	gaussian profile with 1/e width of 100 picoseconds
Raman bandwidth	$7.52 \times 10^9 \text{ sec}^{-1}$
stimulated Raman gain g_1	$7.45 \times 10^{-12} \text{ cm}^{-1} \text{ w}^{-1}$
spontaneous Raman gain g_2	$1.754 \times 10^{-20} \text{ cm}^{-1} \times 637 \times 10^4$ due to the ratio of the geometrical solid angle to the maximum solid angle represented by the array
beam diameter	30 cm
propagation step length	30 m

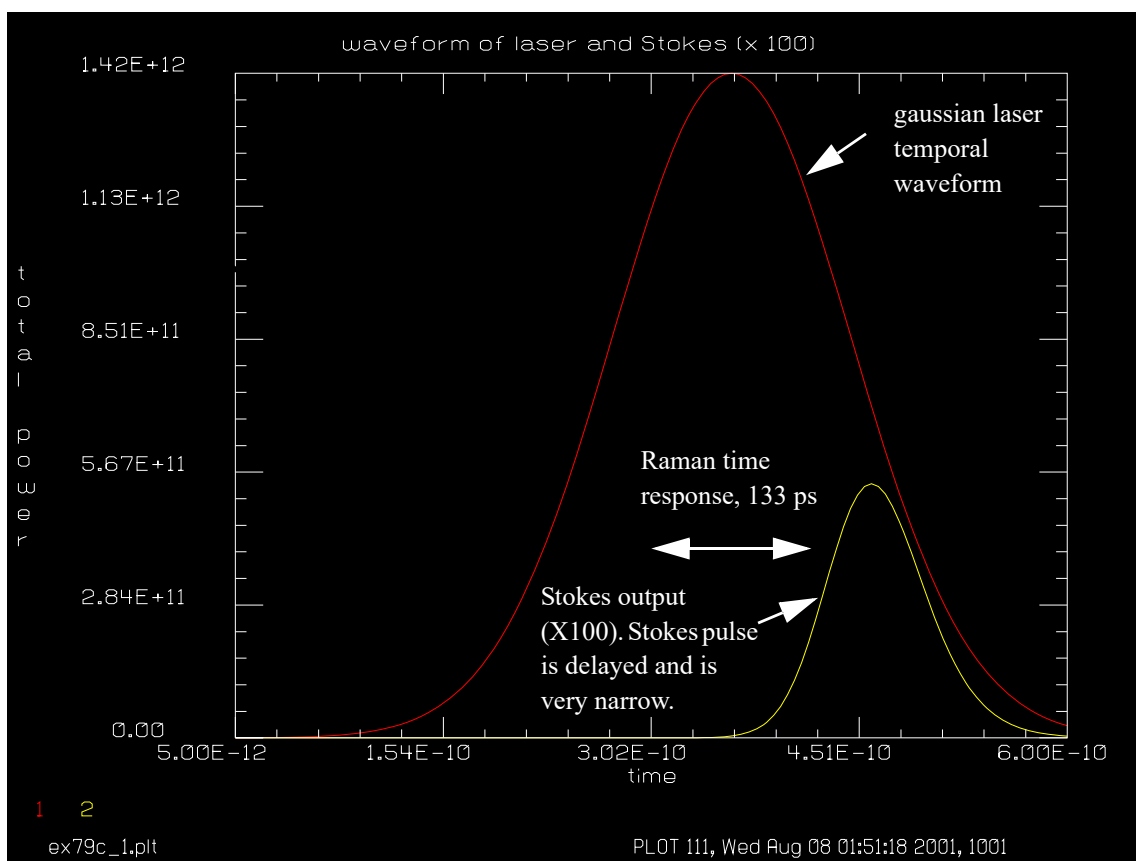


Fig. 79.4. A gaussian temporal waveform for the laser input produces a Stokes pulse which may be delayed less than the 133 picosecond response of the Raman medium. The Stokes pulse may also be narrower than the Raman response time because of the exponential amplification.

Input: ex79c.inp

c## ex79c
c

Jump to: [Commands](#), [Theory](#)

```

c Example to illustrate the effect of the temporal waveform
c of the laser pulse on the transient Raman growth
c
c Parameters
c pulse width          gaussian temporal width, 240 ps wide at 1/e points
c laser irradiance     2 gigawatts per sq. cm.
c
mem/set/b 1
variab/dec/int pass data medium
array/s 1 64
nbeam 12
array/s 3 64 64 1 data
array/s 4 64 64 1 data
array/s 5 64 64 1 data
array/s 6 64 64 1 data
array/s 7 64 64 1 data
array/s 8 64 64 1 data
array/s 9 64 64 1 data
array/s 10 64 64 1 data
array/s 11 64 64 1 data
array/s 12 64 64 1 data
units/s 0 1.
wavelength/set 1 .3511
wavelength/set 2 .35204
c
c pump is beam 1
c Stokes is beam 2
c stimulated gain is 7.45e-12 per cm-watt
c spontaneous gain is 6.4e-9 x 6.37e4 = 4.08e-4 cm-1
c gamma is 7.52e9 sec-1
c
raman/trans/set 1 2 7.45e-12 4.08e-4 7.52e9 11
c
c the time step is 10 picoseconds
c
c integer variables
c pass          time counter
c data          data point
c data1         temporary data point
c medium        counter for medium array
c
c real variables
c time          cumulative time
c peak          gaussian relative amplitude
c energy_s      instantaneous energy in the laser beam
c energy_p      instantaneous energy in the Stokes beam
c
pass = 0          # time step counter
time = 0          # cumulative time
tstep = 5e-12
macro/def outer/o
c
c macro to increment time steps
c

```

Jump to: [Commands](#), [Theory](#)

```

pass = pass + 1
time = time + tstep
clear 1 2.e9          # initialize laser two 2 gigawatts per sq. cm.
c
c use 240 ps wide time pulse delayed by 360 ps
c
peak = gauss(time,360e-12,120e-12,1)
mult 1 peak
clap/c/c 1 15.        # aperture diameter is 30 cm
variab/set energy_p 1 energy
clear 2 0.
macro/run inner1
variab/set energy_s 2 energy list
energy_s = 100*energy_s
udata/set pass time energy_p energy_s
macro/end
macro/def inner1/o
  if pass = 1 then
c
c initialize medium array to reset state, if first time step
c
    raman/trans/rest 3 # initialize media array
    raman/trans/rest 4 # initialize media array
    raman/trans/rest 5 # initialize media array
    raman/trans/rest 6 # initialize media array
    raman/trans/rest 7 # initialize media array
    raman/trans/rest 8 # initialize media array
    raman/trans/rest 9 # initialize media array
    raman/trans/rest 10 # initialize media array
    raman/trans/rest 11 # initialize media array
    raman/trans/rest 12 # initialize media array
  endif
  medium = 2
  data = 0
  macro/run inner2/10
  if pass > 10 then
    title pass @pass
    plot/udata 1 2
  endif
macro/end
macro/def inner2/o
  medium = medium + 1
  data = data + 1
  pack/set 1 2 medium
  pack/in    # propagate 3 meters
    raman/trans/step medium 300 tstep # take Raman step
  pack/out
  pack/close
  if pass = 5 then
c
c save beam growth values for time step 5
c
    variab/set energy_s 2 energy
    log_energy = log10(energy_s)

```

Jump to: [Commands](#), [Theory](#)

```

        udata/set data y03=log_energy
    endif
    if pass = 30 then
c
c  save beam growth values for time step 30
c
        variab/set energy_s 2 energy
        log_energy = log10(energy_s)
        udata/set data y04=log_energy
    endif
    if pass = 120 then
c
c  save beam growth values for time step 120
c
        variab/set energy_s 2 energy
        log_energy = log10(energy_s)
        udata/set data y05=log_energy
    endif
        variab/set energy_s 2 energy
        log_energy = log10(energy_s)
        udata/set data y06=log_energy
macro/end
macro/def temp/o
c
c  macro to reset udata x-axis values
c
    temp1 = temp1 + temp2
    data1 = data1 + 1
    udata/set data1 temp1
macro/end
macro/run outer/120
plot/watch ex79c_1.plt
title waveform of laser and Stokes (x 100)
plot/udata/xlabel time
plot/udata/ylabel total power
plot/udata 1 2
plot/watch ex79c_2.plt
title energy growth vs. distance 25ps
plot/udata/xlabel distance (cm)
plot/udata/ylabel log energy
pass = 0
temp1 = 0
temp2 = 300
data1 = 0
c
c  reset x-axis
c
macro temp/10
udata/maxrows 10
plot/udata/set y03
plot/udata/seq
plot/watch ex79c_3.plt
udata/maxrows 10
title energy growth vs. distance 150ps

```

Jump to: [Commands](#), [Theory](#)

```

plot/udata/set y04
plot/udata/seq
plot/watch ex79c_4.plt
udata/maxrows 10
title energy growth vs. distance 600ps
plot/udata/set y05
plot/udata/seq
udata/list/disk ex79c.out
end

```

Ex79d: Transient behavior of the Raman process with doubled irradiance

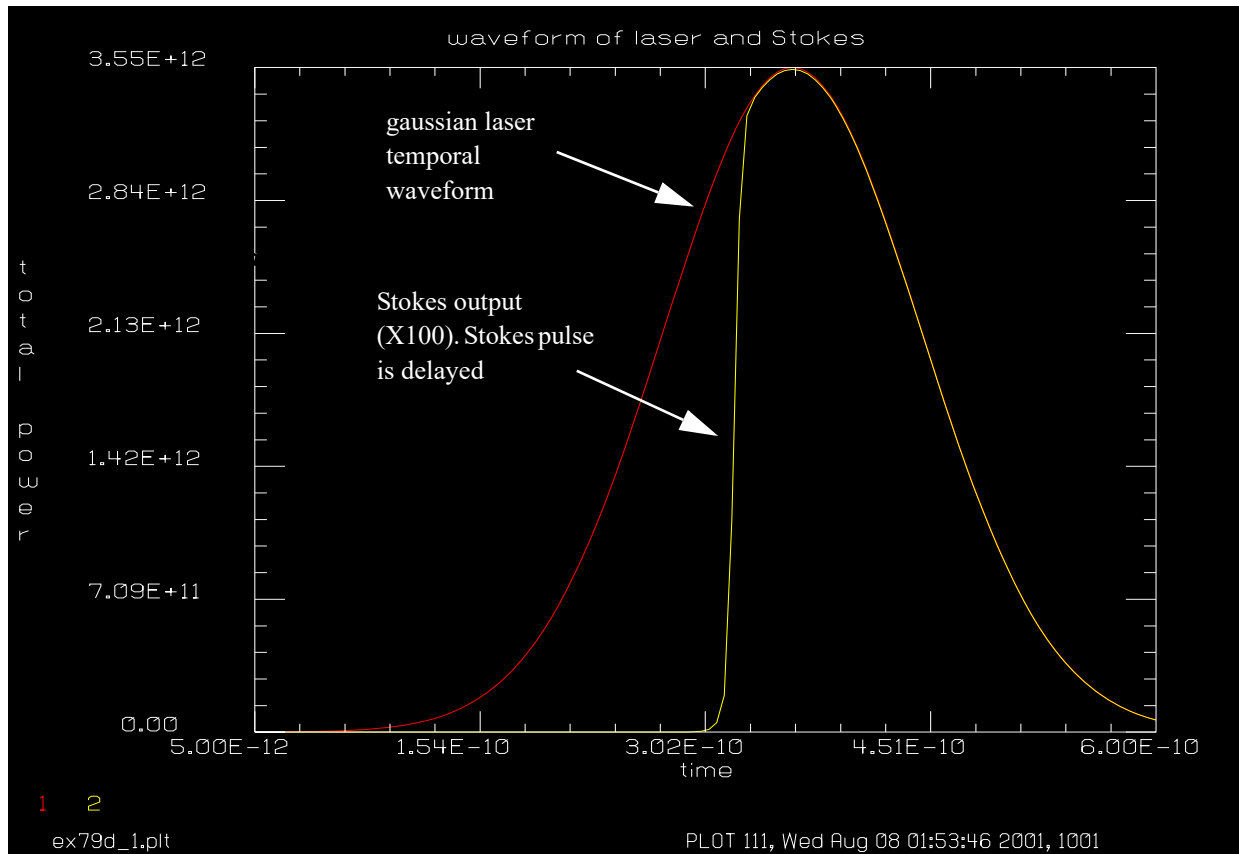


Fig. 79.5. This figure shows the results of Table 79.2 with 4 gigawatts per sq. cm instead of 2. The laser is essentially completely depleted.

Input: ex79d.inp

```

c## ex79d!701534752079307
c
c Example to illustrate the effect of the temporal waveform
c of the laser pulse on the transient Raman growth
c
c Parameters
c pulse width          gaussian temporal width, 240 ps wide at 1/e points

```

Jump to: [Commands](#), [Theory](#)


```

c  laser irradiance  5 gigawatts per sq. cm.
c
c  Example 79d is the same as Example 79c except the irradiance is
c  twice as high at 5 gigawatts per sq. cm. This causes the pump to be
c  strongly depleted. This is the saturated regime for Stokes
c  amplification.
c
mem/set/b 1
variab/dec/int pass data medium
array/s 1 64
nbeam 12
array/s 3 64 64 1 data
array/s 4 64 64 1 data
array/s 5 64 64 1 data
array/s 6 64 64 1 data
array/s 7 64 64 1 data
array/s 8 64 64 1 data
array/s 9 64 64 1 data
array/s 10 64 64 1 data
array/s 11 64 64 1 data
array/s 12 64 64 1 data
units/s 0 1.
wavelength/set 1 .3511
wavelength/set 2 .35204
c
c  pump is beam 1
c  Stokes is beam 2
c  stimulated gain is 7.45e-12 per cm-watt
c  spontaneous gain is 6.4e-9 x 6.37e4 = 4.08e-4 cm-1
c  gamma is 7.52e9 sec-1
c
raman/trans/set 1 2 7.45e-12 4.08e-4 7.52e9 11
c
c  the time step is 10 picoseconds
c
c  integer variables
c  pass          time counter
c  data          data point
c  data1         temporary data point
c  medium        counter for medium array
c
c  real variables
c  time          cumulative time
c  peak          gaussian relative amplitude
c  energy_s      instantaneous energy in the laser beam
c  energy_p      instantaneous energy in the Stokes beam
c
pass = 0          # time step counter
time = 0          # cumulative time
tstep = 5e-12
macro/def outer/o
c
c  macro to increment time steps
c

```

Jump to: [Commands](#), [Theory](#)

```

pass = pass + 1
time = time + timestep
clear 1 5.e9          # initialize laser two 2 gigawatts per sq. cm.
c
c use 240 ps wide time pulse delayed by 360 ps
c
peak = gauss(time,360e-12,120e-12,1)
mult 1 peak
clap/c/c 1 15.        # aperture diameter is 30 cm
variab/set energy_p 1 energy
clear 2 0.
macro/run inner1
variab/set energy_s 2 energy list
udata/set pass time energy_p energy_s
macro/end
macro/def inner1/o
  if pass = 1 then
c
c initialize medium array to reset state, if first time step
c
    raman/trans/rest 3 # initialize media array
    raman/trans/rest 4 # initialize media array
    raman/trans/rest 5 # initialize media array
    raman/trans/rest 6 # initialize media array
    raman/trans/rest 7 # initialize media array
    raman/trans/rest 8 # initialize media array
    raman/trans/rest 9 # initialize media array
    raman/trans/rest 10 # initialize media array
    raman/trans/rest 11 # initialize media array
    raman/trans/rest 12 # initialize media array
  endif
  medium = 2
  data = 0
  macro/run inner2/10
  if pass > 10 then
    title pass @pass
    plot/udata 1 2
  endif
macro/end
macro/def inner2/o
  medium = medium + 1
  data = data + 1
  pack/set 1 2 medium
  pack/in      # propagate 3 meters
    raman/trans/step medium 300 @tstep # take Raman step
  pack/out
  pack/close
  if pass = 5 then
c
c save beam growth values for time step 5
c
    variab/set energy_s 2 energy
    log_energy = log10(energy_s)
    udata/set data y03=log_energy

```

Jump to: [Commands](#), [Theory](#)

```

endif
if pass = 30 then
c
c save beam growth values for time step 30
c
    variab/set energy_s 2 energy
    log_energy = log10(energy_s)
    udata/set data y04=log_energy
endif
if pass = 120 then
c
c save beam growth values for time step 120
c
    variab/set energy_s 2 energy
    log_energy = log10(energy_s)
    udata/set data y05=log_energy
endif
    variab/set energy_s 2 energy
    log_energy = log10(energy_s)
    udata/set data y06=log_energy
macro/end
macro/def temp/o
c
c macro to reset udata x-axis values
c
    temp1 = temp1 + temp2
    data1 = data1 + 1
    udata/set data1 temp1
macro/end
macro/run outer/120
plot/watch ex79d_1.plt
title waveform of laser and Stokes
plot/udata/xlabel time
plot/udata/ylabel total power
plot/udata 1 2
plot/watch ex79d_2.plt
title energy growth vs. distance 25ps
plot/udata/xlabel distance (cm)
plot/udata/ylabel log energy
pass = 0
temp1 = 0
temp2 = 300
data1 = 0
c
c reset x-axis
c
macro temp/10
udata/maxrows 10
plot/udata/set y03
plot/udata/seq
plot/watch ex79d_3.plt
udata/maxrows 10
title energy growth vs. distance 150ps
plot/udata/set y04

```

Jump to: [Commands](#), [Theory](#)

```

plot/udata/seq
plot/watch ex79d_4.plt
udata/maxrows 10
title energy growth vs. distance 900ps
plot/udata/set y05
plot/udata/seq
udata/list/disk ex79d.out
end

```

Ex79e: Effect of weak hot spots in the laser to create strong Stokes spikes

Example 79e illustrates the effect of weak hot spots in the laser to create very strong spikes in the Stokes field. The hot spots in the Stokes beam constitute an increase in the spatial coherence of the Stokes beams and allow the resulting Speckle pattern to be resolved by a 512 x 512 array at a distance of 1200 meters. In order to extend the distance for Raman conversion, the laser intensity was reduced and a pulse of 50 picoseconds was used.

Table. 79.4. Parameters for Ex79e.

Parameter	Value
laser wavelength	0.3511 μ
Stokes wavelength	0.35204 μ
initial laser irradiance	1×10^8 w/cm ²
initial Stokes irradiance	0
laser pulse width	50 picoseconds
Raman bandwidth	7.52×10^9 sec ⁻¹
stimulated Raman gain g_1	7.45×10^{-12} cm ⁻¹ w ⁻¹
spontaneous Raman gain g_2	1.754×10^{-20} cm ⁻¹ $\times 996 \times 10^4$ due to the ratio of the geometrical solid angle to the maximum solid angle represented by the array
beam diameter	30 cm
propagation step length	30 m

Input: ex79e.inp

```

c## ex79e!666665526461732
c
c Example 79e: illustrates the effect of weak hot spots in the laser
c to create strong hot spots in the Stokes beam, which act to increase
c the spatial coherence of the Stokes beam. This leads to relatively
c large speckles which may be resolved at a distance of 1.2 km by
c a 512 x 512 array.
c
c In this particular example, only one time step of 50 ps is considered.
c Since only one time step is considered, the same array is used
c for each of the spatial steps with reinitialization each time.
c
echo/on
mem/set 20

```

Jump to: [Commands](#), [Theory](#)

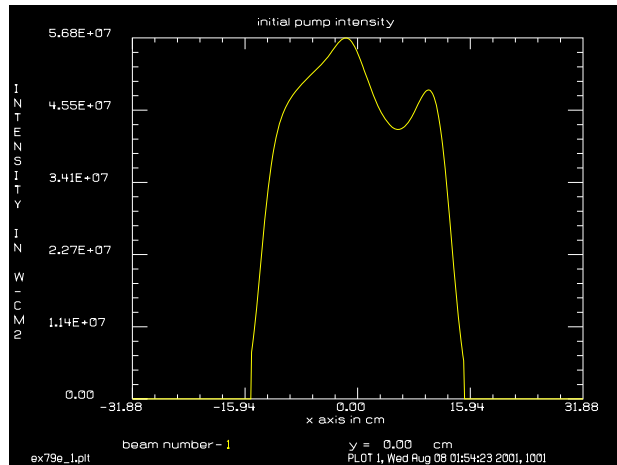


Fig. 79.6. Starting pump distribution with a weak hot spot.

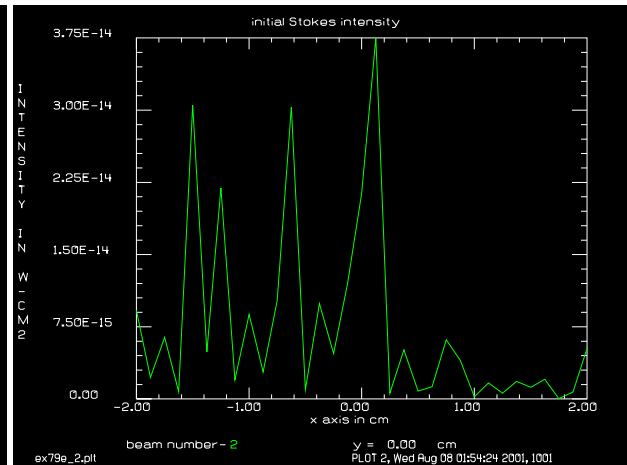


Fig. 79.7. A 4 cm slice of the starting Stokes distribution showing uncorrelated irradiance.

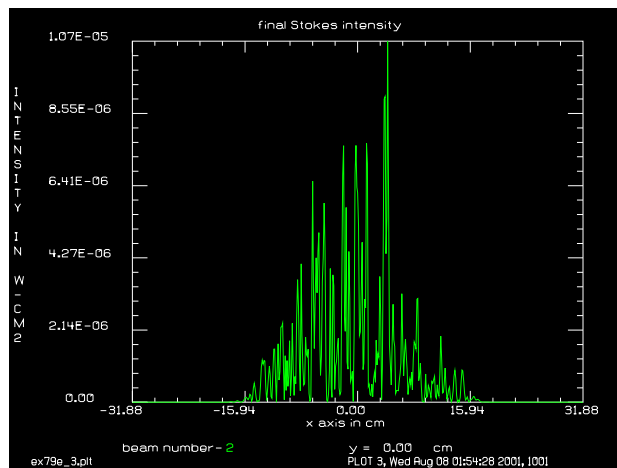


Fig. 79.8. Final Stokes distribution showing strong hot spot due to the strong amplification.

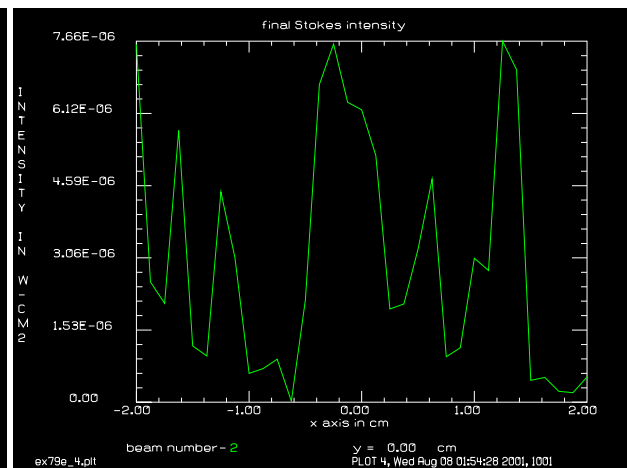


Fig. 79.9. A 4 cm slice of the final Stokes distribution showing relatively large speckles.

```
array/s 1 512
nbeam 3
units/s 0 .125
wavelength/set 1 .3511
wavelength/set 2 .35204
c
c pump beam is 1
c Stokes beam is 2
c stimulated gain is 7.45e-12 per cm-watt
c spontaneous gain is 1.574e-20 x 996 = 1.57e-17 cm-1
c gamma is 7.52e9 sec-1
c
gauss/c/c 1 5e7 15 4
clap/c/c 1 25. # aperture diameter is 30 cm
phase/random 1 .3 4 3
dist 3e4 1
```

Jump to: [Commands](#), [Theory](#)

```
clap/c/c 1 15.
title initial pump intensity
plot/watch ex79e_1.plt
plot/x/i 1
array/s 3 512 512 1 data
units/s 0 .125
raman/trans/set 1 2 7.45e-12 1.57e-17 7.52e9 11
pack/set 1 2 3
clear 2 0.
raman/trans/rest 3 # initialize media array
pack/in
    raman/trans/step 3 1500 50e-12 # take Raman step
pack/out
title initial Stokes intensity
plot/watch ex79e_2.plt
plot/x/i 2 left=-2 right=2
dist 1500 2
clap/c/c 2 20
raman/trans/rest 3 # initialize media array
pack/in
    raman/trans/step 3 38500 50e-12 # take Raman step
pack/out
dist 38500 2
clap/c/c 2 20
raman/trans/rest 3 # initialize media array
pack/in
    raman/trans/step 3 40000 50e-12 # take Raman step
pack/out
dist 40000 2
clap/c/c 2 20
raman/trans/rest 3 # initialize media array
pack/in
    raman/trans/step 3 40000 50e-12 # take Raman step
pack/out
dist 40000 2
clap/c/c 2 20
title final Stokes intensity
plot/watch ex79e_3.plt
plot/x/i 2
plot/watch ex79e_4.plt
plot/x/i 2 left=-2 right=2
end
```

Ex80: Q-switch laser

Table. 80.1. Table of Ex80 examples

Ex80a: Q-switch YAG laser.	5
Ex80b: Q-switch YAG laser, full polarization.	9
Ex80c: Q-switch, saturable absorber	14
Ex80d: Q-switch, time-limited Beer's Law gain	17
Ex80e: Q-switch, YAG laser using a relatively slow Q-switch.	20
Ex80f: Q-switch, YAG laser pulsed laser diode pumping.	23
Ex80g: An example of a gain/absorber, numerical check.	26
Ex80h: An example of a saturable absorber based on gain/absorber	28

The objective of this example is to investigate and model the a Q-switch laser to determine the output power, beam divergence, and the intensity profile. The laser is a Q-switched YAG laser with about a 60 cm round-trip path an approximately a 10 ns pulse. The device is characterized by the following components

- 1) electro-optic Q-switch
- 2) a 6 mm diameter YAG rod
- 3) a polarizing output coupler
- 4) crossed roof mirrors as end reflectors
- 5) an off-axis cube corner reflector to fold the system
- 6) alignment wedges
- 7) a periscope-type, stronglink “on-off” switch
- 8) a quarter-wave plate

It is anticipated that the divergence will be between 3 and 7 milliradians (about 20 times diffraction-limited).

Conceptual Discussion of the Operation of the Device

The optical system is shown schematically in Fig. 80.1. The optical system is bent in the form of a “U” to provide more compact packaging. An extended corner cube is used to make the 180° bend. The corner cube is insensitive to all three forms of rotational misalignment. A rhomb is used to provide a complete, positive optical isolation, called a stronglink prism. The rhomb is insensitive to all rotational misalignments and has no other significant effect on optical performance. It may be represented as a simple on-off shutter.

A pair of thin prisms are provided which may be rotated azimuthally to align the optical axis. This is a fairly crude adjustment, necessary only to center the beam with respect to the apertures. As discussed below, the use of crossed roof prisms as end mirrors makes the resonator self-aligning so the angular alignment of the resonator is independent of the alignment prisms. Hence, the alignment prisms may be neglected in the optical analysis.

A Brewster window is used as a polarizing beam splitter. It is nearly perfectly efficient in transmission for p-polarization electric vector parallel to the plane of the paper) and reflects a small percentage of the s-polarization. Since only the s-polarization is outcoupled, we assume a half-wave plate is added in the optical path to cross-couple s- and p-polarizations, so that all of the light is outcoupled.

It is very helpful to write the polarization states and polarization operators using Jones calculus.

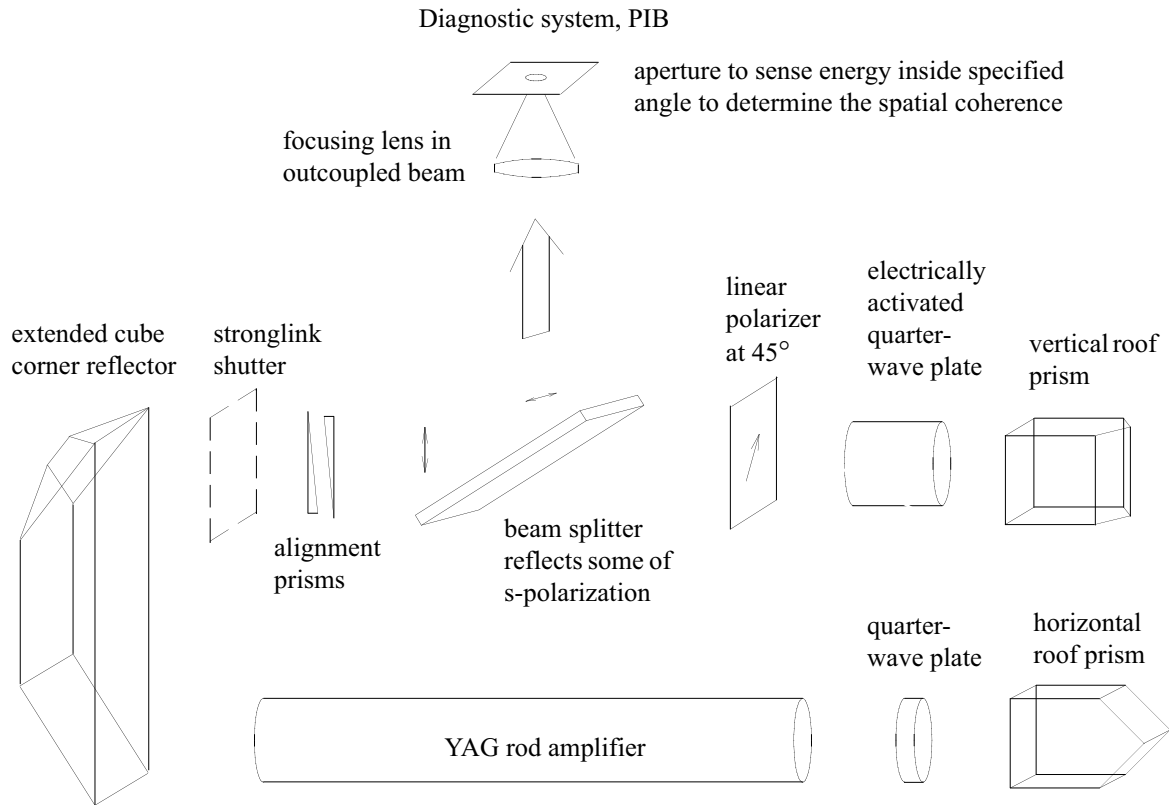


Fig. 80.1. Schematic of the Q-switch laser. The resonator is defined by the two crossed roof prism end reflectors, which assure that the optical axis is effectively perfectly aligned. The system is folded by an extended corner cube. A “stronglink” shutter consisting of a mirror rhomb provides a positive “off” condition. Alignment prisms are provided for rough centering of the beam with respect to the limiting apertures. A beam divider outcouples a percentage of the s-polarization but passes all of the p-polarization. The Q-switch consists of a 45° linear polarizer and an electrically activated quarter-wave plate. When activated the combination of quarter-wave plate, in double pass, and roof mirror transmits the polarization component. When deactivated the combination blocks all light. In Ex80b.inp the outcoupler is assumed to be rotated 45° to line up with the linear polarizer. Alternately the roofs could be rotated 45°.

$$\text{x-polarization} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (80.1)$$

$$\text{y-polarization} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad (80.2)$$

A half-wave plate is oriented with the fast axis vertical and the roof mirrors are oriented so that one has a horizontal roof line and the other is vertical. The Jones equation for a pass through the quarter-wave plate, horizontal roof mirror (roof line is horizontal, in GLAD referred to as EW), and a second pass through the half-wave plate:

$$e^{j\pi/4} \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} e^{j\pi/4} \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} = i \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (80.3)$$

This allows all polarization states to pass the roof mirror with no change in state.

A Q-switch is used to inhibit the build-up of the optical field while the medium is being optically pumped. The traditional Q-switch consists of a linear polarizer, an electrically activated quarter-wave plate (with fast-axis at 45° degrees with respect to the linear polarization, and the end mirror. When the quarter-wave plate is not activated, one polarization is passed through the polarizer, the mirror, and the polarizer again without change. When the quarter-wave plate is activated, the linear polarization is changed to circular polarization. A simple mirror end reflector will flip the state of circular polarization, resulting in linear polarization after the quarter-wave plate which is orthogonal to the linear polarizer and is not passed. In the “off” condition the Q-switch has the equation,

$$\frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} e^{j\pi/4} \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} e^{j\pi/4} \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad (80.4)$$

The activated Q-switch, shuts off the end reflector. The loss per round-trip is essentially complete and the device can not reach threshold. When the Q-switch is deactivated, the polarization is unchanged and passes through the linear polarizer with no loss.

Since a roof mirror does not flip the state of circular polarization, it is necessary to redesign the Q-switch. We find that the on-off state to be exactly reversed with respect to the simple end mirror. The activated quarter-wave plate allows the light to pass. In the off condition, the Q-switch blocks light. The on-state for a roof mirror with vertical roof line is

$$\frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} e^{j\pi/4} \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} e^{j\pi/4} \begin{bmatrix} 1 & 0 \\ 0 & -i \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = -i \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (80.5)$$

The device will operate with linear polarization in the 45° azimuthal direction. We could as readily have rotated both roof mirrors by 45° to have the linear polarization oriented in the horizontal direction. In this example, the polarization outcoupler is rotated to be aligned with the linear polarization of the resonator so that all of the light may be outcoupled.

Because the end reflectors are crossed roof mirrors, the rays can not “walk off” the resonator axis as is the case with flat mirrors that are even slightly misaligned.

The YAG rod is the source of optical energy and is pumped by a flash lamp. The device is operated by turning on the voltage to the electrically operated quarter-wave plate to effectively turn off the associated end reflector. The flash lamp is activated and a population inversion is stored in the YAG rod. In the absence of an optical field, the population inversion is depleted at the spontaneous decay rate, which is approximately $\tau_{spont} = 3 \times 10^{-4}$ sec. This is generally a negligible effect in terms of depleting the population inversion, for the fast operation times typical of a Q-switch. However the spontaneous decay of the population inversion is the source of light for stimulated emission. With the Q-switch on, the cavity losses are very high and the net round-trip gain is strongly negative.

When the optical pumping is essentially complete, the Q-switch is turned off to make the associated end reflector essentially 100% efficient. The net round-trip gain is strongly positive and the spontaneous emission becomes amplified. The spontaneous emission is emitted into 4π steradians. However, only the radiation which is in the forward direction will successfully pass through the system apertures and the YAG rod to become amplified. Only very low angle rays will successfully pass through the series of apertures encountered during numerous passes through the device. Of course, Spontaneous emission occurs as long as the population inversion exists, so spontaneous emission is continuously added to the optical beam. However, the spontaneous emission which is in the cavity when the Q-switch is turned off has greater amplification than spontaneous emission which occurs later. Consequently the “early” spontaneous emission dominates the process. Hypothetically, if we were to look back into the device from the output window (please do not look into the Q-switched laser to try this) and if we could see $1.06\ \mu$ radiation, we would see the optical system as an unfolded tunnel with a series of limiting apertures. The spontaneous emission would appear as a self-luminous “fog”. The self-luminous fog which is at a distance in an absorbing atmosphere. In effect the source of the radiation is the spontaneous emission which is present in the cavity when the Q-switch is turned off. This source is a delta-correlated coherent object of diameter equal to the YAG rod diameter at a distance which is the product of the time from the Q-switch initiation and the speed of light. The spatial coherence or beam quality of the device may be understood by considering the output of the laser to be the amplified speckle pattern produced by the coherent source of finite angular subtense. Clearly, the spatial coherence will vary with pulse time. The number of times diffraction limited will be approximately,

$$\frac{d^2}{\lambda t c} \quad (80.6)$$

where d is the diameter of the device and t is the time from Q-switch initiation. The angular filtering process which improves the spatial correlation occurs even during the very early stages of the pulse, when the beam still has very low power and is rising exponentially.

Gain rate equations

The details of the rate equation gain are described in Section 9.3, GLAD Theory Manual. If we consider the optical field to be of intensity I and of duration, Δt , the round-trip time: then the optical field contains a well-defined photon flux. The potential photon flux increase due to the population inversion is

$$\frac{1}{2}\Delta N(0)L \quad (80.7)$$

where L is the length of the gain region, as illustrated in Fig. 80.2. The net energy of an incident square pulse of irradiance $I(z)$ and temporal length Δt , giving the energy density as $I(0)\Delta t$. The energy density in a gain sheet representing a length of L is $\Delta N(0)h\nu L/2$. The sum of these two energy densities is a constant by conservation of energy.

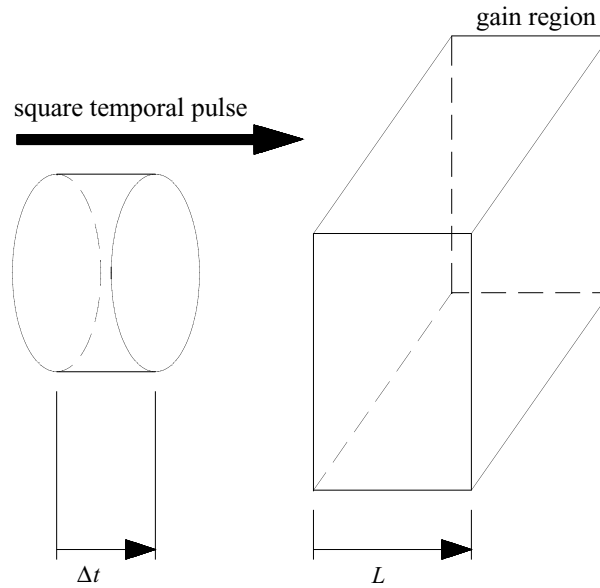


Fig. 80.2. A time slice of the pulse of length Δt interacts with a gain region of length L , having energy density $\Delta N(0)h\nu L/2$.

Q-Switch pulse decay

The rise of the Q-switch pulse is largely driven by the value of the initial population inversion, the decay of the pulse is essentially completely determined by the photon lifetime which is calculated from,

$$t_c = \frac{nL}{c(1-R)} \quad (80.8)$$

where L is the round-trip cavity length and R is the round trip efficiency (often the reflectivity of the output mirror).

Ex80a: Q-switch YAG laser

Example ex80a.inp shows a simplified Q-switch which is functionally similar to the full system shown in Fig. 80.1, but which does not explicitly treat the polarization effects and the Q-switch. Fig. 80.3 shows the output at pass 10, about 20 nanoseconds after Q-switch initiation. The spontaneous emission is treated as being delta-correlated when it is added, but this radiation becomes correlated during propagation as the widest angle radiation is diffracted out of the beam and is clipped by the apertures. The population inversion is initially set to 0.1875 j/cm^3 . With this high gain, the output power peaks after two passes when the population inversion is completely depleted. The optical field then decays because of the outcoupling and the clipping of wide angle radiation by the apertures. Initially the loss of the wide angle light is the strongest loss. Fig. 80.4 shows the peak outcoupled power and the power within 5 milliradians. Fig. 80.5 shows the relative ratio of power within 5 milliradians. This is a rough measure of spatial coherence. We can see that the divergence meets the design goal of 5 milliradians within the first 10 nanoseconds.

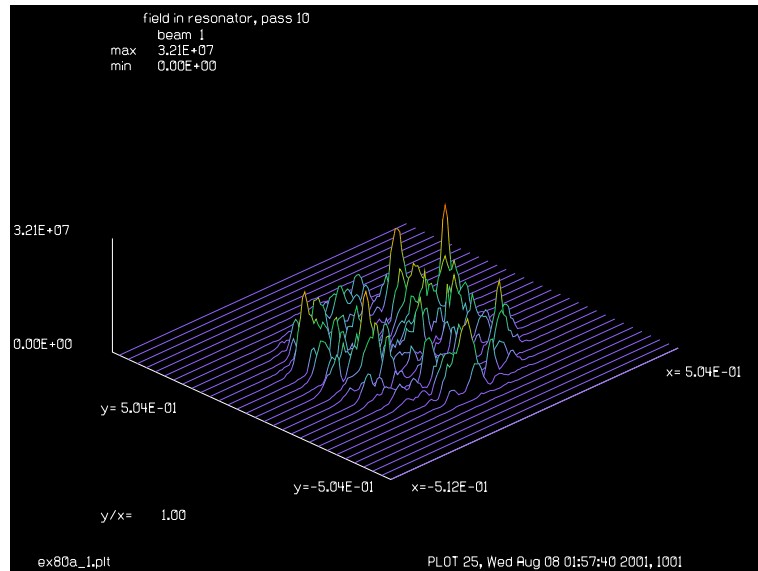


Fig. 80.3. Outcoupled beam distribution at pass 10 of ex80a.inp corresponding to 20 nanoseconds after the Q-switch. This beam is approximately 20 times diffraction limited.

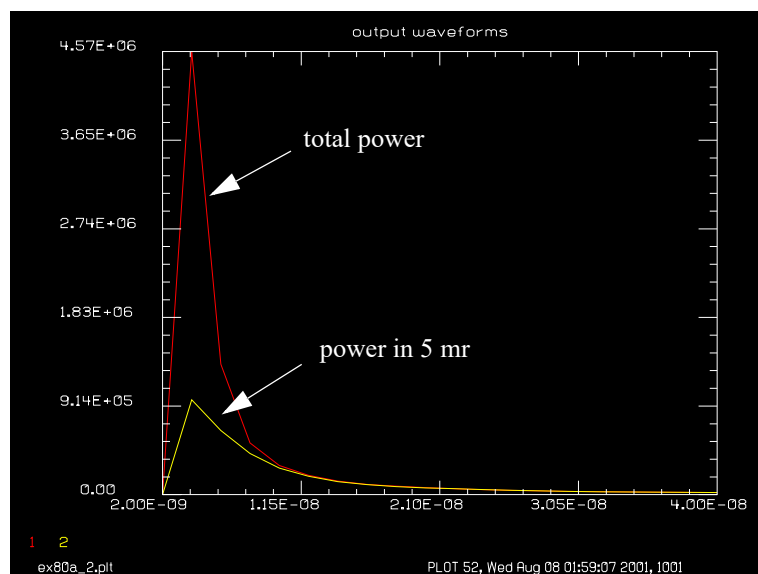


Fig. 80.4. Output power of Q-switch versus time and the output power falling within 5 milliradians.

Input: ex80a.inp

```

c## ex80a
c
c Example 80a: Q-switch, YAG laser
c
c This is an example of a Q-switched, YAG laser.
c This model is simplified, a more detailed model is given in
c ex80b.inp
c
c Beams

```

Jump to: [Commands](#), [Theory](#)

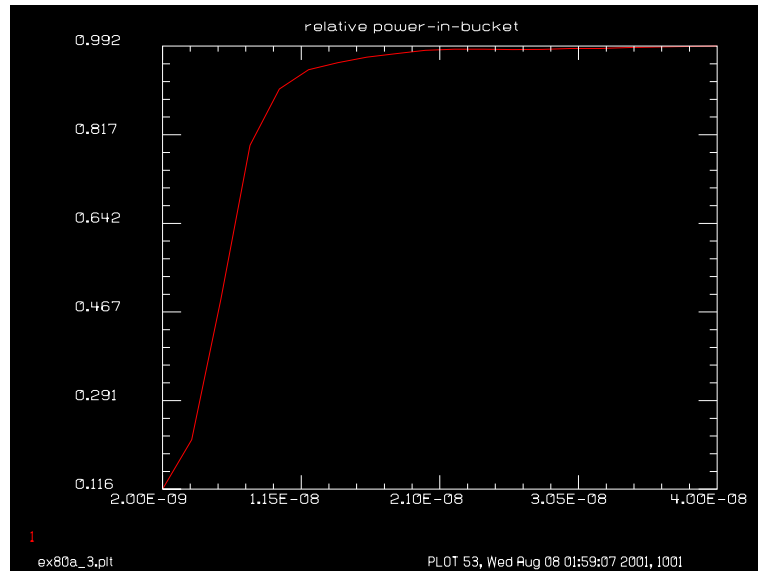


Fig. 80.5. Relative output power falling within 5 milliradians, showing the increase in spatial correlation.

```

c  -----
c  Beam 1      optical field in the cavity
c  Beam 2      medium array
c  Beam 3      outcoupled beam
c
variab/dec/int pass switch nstep
array/s 1 128 128
nbeam 3                      # set number of beams
array/s 2 128 128 1 data      # make medium array polarized
units/s 0 .008
wavelength/set 0 1.06         # set wavelength and index
width = 1.2e10                # set line width, hz
center = 2.83e14              # set line center, hz
tspont = 3e-4                 # set spontaneous emission time, sec
t20 = 3e-1                   # set decay time for level 2, sec
t10 = 1e-5                   # set decay time for level 1, sec
mode_sep = 5e8                # set longitudinal mode separation, hz
c
c  Set offset from line center of first mode, 0 hz
c  Set fractional pumping into level 1, nlpump, 0
c
gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
pack/set 1 2                  # pack both beams
plot/watch plot1.plt
macro/def ex80a/o
  mirror/flat 1
  pass = pass + 1             # increment pass counter
  clap/c/n 1 .3               # 3 mm radius clear aperture on rod
  wavelength/set 1 1.06 1.6
  pack/in                     # pack beams
  gain/rate/step 10 time nstep

```

Jump to: [Commands](#), [Theory](#)

```

pack/out                                # unpack beams
prop 10
wavelength/set 1 1.06 1.
clap/c/n 1 .3                          # 3 mm radius clear aperture on rod
prop 20
c ----- outcoupling section
copy 1 3
mult 1 trans                          # Fresnel reflection losses
mult 3 refl list
title near-field of output, pass @pass
plot/l 3
pause 5
lens 3 100
beams/off 1
beams/on 3
prop 100
beams/off 3
beams/on 1
title far-field of output, pass @pass
plot/l 3
variab/set energy1 3 energy list      # record peak irradiance
clap/c/n 3 .25                        # clear aperture subtending 5 mr
variab/set energy2 3 energy list      # record peak irradiance
if [energy1!= 0] then
    diff = energy2/energy1
    udata/set pass tot_time energy1 energy2 diff  # store for the record
endif
tot_time = tot_time + time
if pass = 10 then
    plot/watch ex80a_1.plt
    title field in resonator, pass 10
    plot/l 1
    plot/watch plot1.plt
endif
switch = mod(pass,5)                  # plot every 5th pass
if switch = 0 then
    title output waveforms
    plot/udata/123 1 2                # make plot output energy
    pause 8
    title relative power-in-bucket
    plot/udata/123 3                  # make plot of power-in-the-bucket
endif
c ----- continue with resonator field section
mirror/flat 1
prop 20
clap/c/n 1 .3                          # 3 mm radius clear aperture on rod
wavelength/set 1 1.06 1.6
pack/in                                # pack beams
    gain/rate/step 10 time nstep
pack/out                                # unpack beams
prop 10
wavelength/set 1 1.06 1.
clap/c/n 1 .3                          # 3 mm radius clear aperture on rod
macro/end

```

Jump to: [Commands](#), [Theory](#)

```

time = 2e-9
tot_time = 0
trans = .95
refl = 1. - trans
nstep = 1
pass = 0
beams/all/off
beams/on 1
resonator/name ex80a
gaus/c/c 1 1 .3
resonator/eigen/test 1
clear 1 0.                # set initial irradiance
c
c Set initial population inversion as N x h x frequency
c
c N x h x fre = 1e18 X 6.626e-34 X 2.83e14 = .1875 J/cm**3
c
clear 2 .1875              # set population inversion in J/cm**3
irradiance 2              # Modify beam 2 to put
                          # the inversion in the
                          # real word

c
jones/set ar=0 ai=1 dr=0   # now put inversion in the imaginary word
jones/mult 2
plot/udata/set y01         # specify data to be displayed
time = 2e-9
pass = 0
resonator/run 20           # run for 20 round-trips
plot/watch ex80a_2.plt
title output waveforms
plot/udata/123 1 2        # make plot output energy
plot/watch ex80a_3.plt
title relative power-in-bucket
plot/udata/123 3          # make plot of power-in-the-bucket

```

Ex80b: Q-switch YAG laser, full polarization

Example ex80b.inp includes the detailed polarization description. The macro ex80b begins at the lower roof mirror as shown in Fig. 80.1 and goes through the gain region, to the corner cube, through the outcoupler, to the Q-switch, back through the corner cube, and finally back through the gain region. The Q-switch includes a linear polarizer which filters the spontaneous emission to a single polarization state at 45°. The subsequent amplification works only on the remaining polarization state and essentially all of the population inversion goes into this polarization state. The outcoupling actually occurs during each pass through the splitter, but the problem has been simplified to include outcoupling only on one pass. The detailed model gives essentially identical performance to the simple model used in ex80a.inp.

Input: ex80b.inp

```

c## ex80b
c
c Example 80b: Q-switch, YAG laser, full polarization

```

Jump to: [Commands](#), [Theory](#)

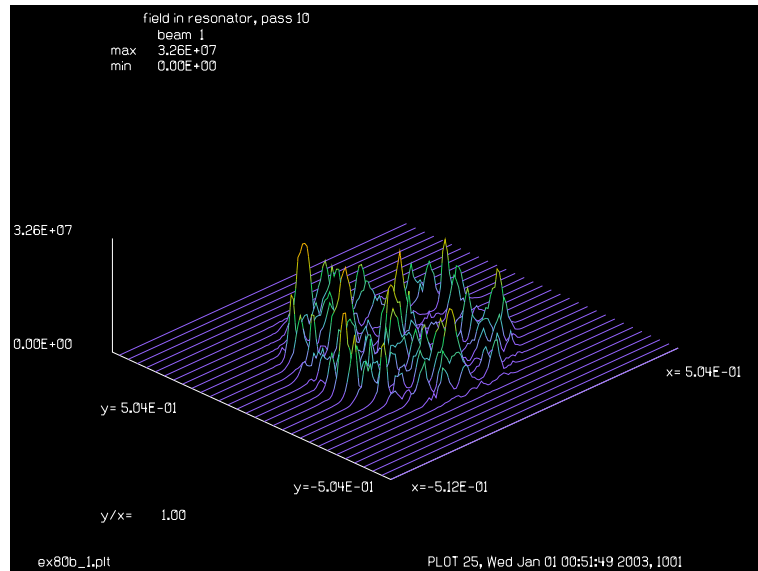


Fig. 80.6. Outcoupled beam distribution at pass 10 for ex80b.inp corresponding to 20 nanoseconds after the Q-switch. This beam is approximately 20 times diffraction limited.

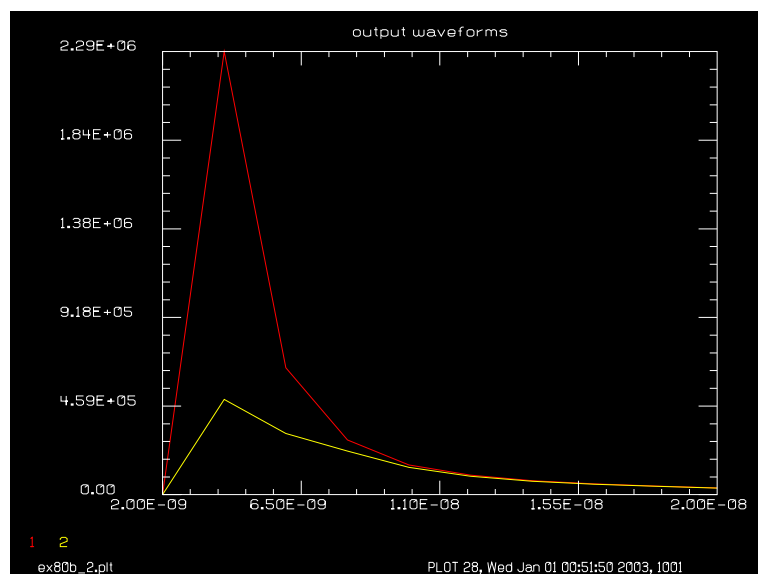


Fig. 80.7. Output power of Q-switch versus time and the output power falling within 5 milliradians, for ex80b.inp.

```

c
c This is an example of a Q-switched, YAG laser.
c
c Establish initial units and a gaussian field distribution
c
c Beams
c -----
c Beam 1    optical field in the cavity
c Beam 2    medium array
c Beam 3    outcoupled beam
c

```

Jump to: [Commands](#), [Theory](#)

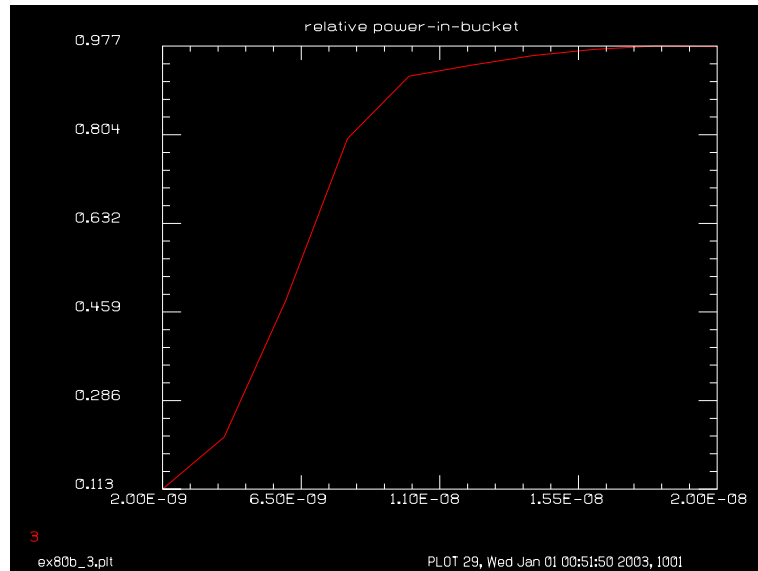


Fig. 80.8. Relative output power falling within 5 milliradians, showing the increase in spatial correlation, for ex80b.inp.

```

variab/dec/int pass switch nstep plot_now
array/s 1 128 128 1
nbeam 3                                # set number of beams
array/s 2 128 128 1 data                # make medium array polarized
units/s 0 .008
wavelength/set 0 1.06                   # set wavelength and index
width = 1.2e10                           # Set line width, hz
center = 2.83e14                         # Set line center, hz
tspont = 3e-4                           # Set spontaneous emission time, sec
t20 = 3e-1                              # Set decay time for level 2, sec
t10 = 1e-5                              # Set decay time for level 1, sec
mode_sep = 5e8                          # Set longitudinal mode separation, hz
c
c Set offset from line center of first mode, 0 hz
c Set fractional pumping into level 1, nlpump, 0
c
gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
pack/set 1 2                            # pack both beams
pass = 0                                # initialize pass counter
plot/watch plot1.plt
macro/def ex80b/o
    jones/retard 1 0 90                  # fixed quarter-wave plate to compensate
    roof/ew 1
    jones/retard 1 0 90                  # second pass of fixed quarter-wave plate
    beams/all/off
    beams/on 1
    pass = pass + 1                      # increment pass counter
    clap/c/n 1 .3                       # 3 mm radius clear aperture on rod
    wavelength/set 1 1.06 1.6
    pack/in                              # pack beams

```

Jump to: [Commands](#), [Theory](#)

```

    gain/rate/step 10 dt nstep
pack/out                                # unpack beams
prop 10
wavelength/set 1 1.06 1.
clap/c/n 1 .3                          # 3 mm radius clear aperture on rod
prop 5
corner 1
prop 5
c ----- outcoupling section
copy 1 3
c
c polarization beam splitter at +45 deg.
c
c attenuate resonator field
c
jones/set ar=1 ai=0 br=0 bi=0 cr=0 ci=0 dr=1 di=0
jones/set ar=refl dr=refl
jones/mult 1
c
c attenuate outcoupled field
c
jones/set ar=1 ai=0 br=0 bi=0 cr=0 ci=0 dr=0 di=0
jones/set ar=trans dr=trans
jones/mult 3
title near-field of output, pass @pass
plot/1 3
pause 5
lens 3 100
beams/off 1
beams/on 3
prop 100
beams/off 3
beams/on 1
title far-field of output, pass @pass
plot/1 3
variab/set energy1 3 energy list      # record peak irradiance
clap/c/n 3 .25                        # clear aperture subtending 5 mr
variab/set energy2 3 energy list      # record peak irradiance
if [energy1 != 0] then
    eng_ratio = energy2/energy1
    udata/set pass time energy1 energy2 eng_ratio # store for the record
endif
time = time + dt
plot_now = mod(pass,5)                # plot every 5th pass
if pass = 10 then
    plot/watch ex80b_1.plt
    title field in resonator, pass 10
    plot/1 1
    plot/watch plot1.plt
endif
if plot_now = 0 then
    title output waveforms
    plot/udata/123 1 2                # make plot output energy
    title relative power-in-bucket

```

```

    plot/udata/123 3          # make plot of power-in-the-bucket
endif
c ----- continue with resonator field section
prop 10
c ----- start of q-switch
c
c if switch = 1, on
c if switch = 0, off
c
jones/linpol 1 45            # linear polarizer at +45 deg.
if switch = 1 jones/retard 1 0 90 # switchable quarter-wave plate
roof/ns 1                    # roof mirror end reflector
if switch = 1 jones/retard 1 0 90 # second pass of switchable reflector
jones/linpol 1 45            # second pass of linear polarizer
c ----- end of q-switch
beams/all/off
beams/on 1
prop 15
corner 1
prop 5
clap/c/n 1 .3                # 3 mm radius clear aperture on rod
wavelength/set 1 1.06 1.6
pack/in                      # pack beams
    gain/rate/step 10 dt nstep
pack/out                    # unpack beams
prop 10
wavelength/set 1 1.06 1.
clap/c/n 1 .3                # 3 mm radius clear aperture on rod
zreff/se 1 0
zreff/se 3 0
macro/end
beams/all/off
beams/on 1
dt = 2e-9
time = 0
refl = .95
refl = refl^.5 list
trans = 1. - refl
trans = trans^.5 list
resonator/name ex80b
gaus/c/n 1 1 .3
resonator/eigen/test 1
clear 1 0.                  # set initial irradiance
c
c Set initial population inversion as N x h x frequency
c
c  $N \times h \times \text{fre} = 1e18 \times 6.626e-34 \times 2.83e14 = .1875 \text{ J/cm}^3$ 
c
clear 2 0.
clear 3 .1875                # set population inversion in  $\text{J/cm}^3$ 
gain/rate/n2level 3 2       # Modify beam 2 to put
                             # the inversion in the
                             # imaginary word
c

```

Jump to: [Commands](#), [Theory](#)

```

plot/udata/set y01          # specify data to be displayed
pass = 0
time = 0
nstep = 1
switch = 1                  # open q-switch
resonator/run 10            # run for 20 round-trips
plot/watch ex80b_2.plt
title output waveforms
plot/udata/123 1 2          # make plot output energy
plot/watch ex80b_3.plt
title relative power-in-bucket
plot/udata/123 3            # make plot of power-in-the-bucket

```

Ex80c: Q-switch, saturable absorber

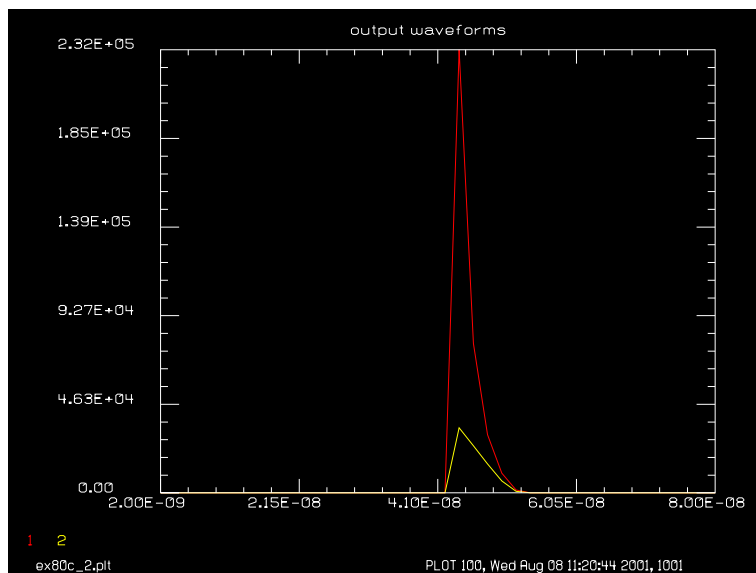


Fig. 80.9. Output power of Q-switch versus time for a saturable absorber Q-switch and the output power falling within 2.5 milliradians, from ex80c.inp. It is assumed that the medium is pumped at a constant rate. The delay of the pulse is due to the build up of the optical field to the point where the saturable absorber becomes saturated.

Input: ex80c.inp

```

c## ex80c
c
c Example 80c: Q-switch, YAG laser
c
c This is an example of a Q-switched, YAG laser, using a saturable
c absorber as the Q-switch.
c
c Beams
c -----
c Beam 1    optical field in the cavity
c Beam 2    medium array

```

Jump to: [Commands](#), [Theory](#)

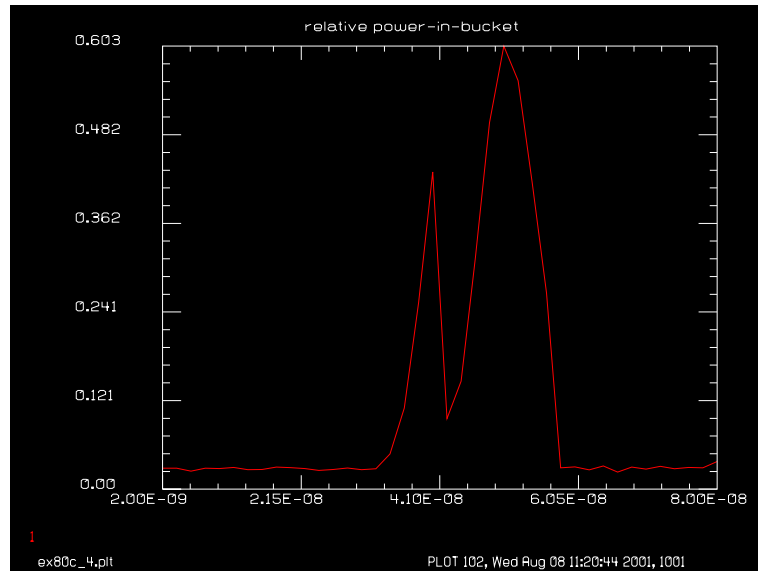


Fig. 80.10. Relative output power falling within 2.5 milliradians, showing the increase in spatial correlation, for ex80c.inp.

```

c Beam 3      outcoupled beam
c
variab/dec/int pass switch nstep
array/s 1 128 128
nbeam 3                      # set number of beams
array/s 2 128 128 1 data      # make medium array polarized
units/s 0 .008
wavelength/set 0 1.06         # set wavelength and index
width = 1.2e10                # set line width, hz
center = 2.83e14              # set line center, hz
tspont = 3e-4                 # set spontaneous emission time, sec
t20 = 3e-1                    # set decay time for level 2, sec
t10 = 1e-5                    # set decay time for level 1, sec
mode_sep = 5e8                # set longitudinal mode separation, hz
c
c Set offset from line center of first mode, 0 hz
c Set fractional pumping into level 1, nlpump, 0
c
gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
pack/set 1 2                  # pack both beams
plot/watch plot1.plt
macro/def ex80c/o
  mirror/flat 1
  pass = pass + 1              # increment pass counter
  clap/c/n 1 .3                # 3 mm radius clear aperture on rod
  wavelength/set 1 1.06 1.6
  pack/in                      # pack beams
    gain/rate/step 10 time nstep
  pack/out                    # unpack beams
  wavelength/set 1 1.06 1.

```

Jump to: [Commands](#), [Theory](#)

```

beer/noprop 1 10
prop 10
clap/c/n 1 .3          # 3 mm radius clear aperture on rod
prop 20
c ----- outcoupling section
copy 1 3
mult 1 trans           # Fresnel reflection losses
mult 3 refl list
title near-field of output, pass @pass
plot/l 3
pause 5
lens 3 100
beams/off 1
beams/on 3
prop 100
beams/off 3
beams/on 1
title far-field of output, pass @pass
plot/l 3
variab/set energy1 3 energy list    # record peak irradiance
clap/c/n 3 .125                    # clear aperture subtending 2.5 mr
variab/set energy2 3 energy list    # record peak irradiance
if [energy1 != 0] then
    diff = energy2/energy1
    udata/set pass tot_time energy1 energy2 diff    # store for the record
endif
tot_time = tot_time + time
if pass = 10 then
    plot/watch ex80c_1.plt
    title field in resonator, pass 10
    plot/l 1
    plot/watch plot1.plt
endif
switch = mod(pass,5)              # plot every 5th pass
if switch = 0 then
    title output waveforms
    plot/udata/123 1 2            # make plot output energy
    bell
    pause 5
    title relative power-in-bucket
    plot/udata/123 3 min=0        # make plot of power-in-the-bucket
endif
c ----- continue with resonator field section
mirror/flat 1
prop 20
clap/c/n 1 .3          # 3 mm radius clear aperture on rod
wavelength/set 1 1.06 1.6
pack/in                # pack beams
    gain/rate/step 10 time nstep
pack/out               # unpack beams
wavelength/set 1 1.06 1.
beer/noprop 1 10
prop 10
clap/c/n 1 .3          # 3 mm radius clear aperture on rod

```

```

macro/end
time = 2e-9
tot_time = 0
trans = .95
refl = 1. - trans
nstep = 1
pass = 0
beams/all/off
beams/on 1
resonator/name ex80c
gaus/c/c 1 1 .3
beer/set -1 1e5
resonator/eigen/test 1
pass = 0
clear 1 0.
clear 2 1e5                                # set pumping rate
gain/rate/n2pump 2 2                        # Modify beam 2 to put
                                           # the pump rate in the real word
resonator/run 40                            # run for 40 round-trips
plot/watch ex80c_2.plt
title output waveforms
plot/udata/123 1 2                          # make plot output energy
plot/watch ex80c_3.plt
title output waveforms, expanded scale
plot/udata/123 1 2 1e=4e-8 ri=6e-8 min=0    # make plot of power-in-the-bucket
plot/watch ex80c_4.plt
title relative power-in-bucket
plot/udata/123 3 min=0                      # make plot of power-in-the-bucket
end

```

Ex80d: Q-switch, time-limited Beer's Law gain

Input: ex80d.inp

```

c## ex80d
c
c Example 80d: Q-switch, YAG laser
c
c This is an example of a Q-switched, YAG laser,
c approximated with time-limited Beer's Law gain.
c
c Beams
c -----
c Beam 1    optical field in the cavity
c Beam 2    outcoupled beam
c Beam 3    integrated outcoupled beam
c
alias Name ex80d
variab/dec/int pass TEST nstep Nline
variab/dec/rea Ppl_start
Nline = 256
Lambda = 1.06e-4

```

Jump to: [Commands](#), [Theory](#)

```

array/s 1 Nline
nbeam 2                                # add a second propagating beam
nbeam 3 data                            # add a data beam
units/field 0 .4
wavelength/set 0 1.96                  # set wavelength and index
macro/def ex80d/o
  mirror/flat 1
  pass = pass + 1                      # increment pass counter
  clap/c/n 1 Apt                       # 2 mm radius clear aperture on rod
  prop 5                               # guess at mirror-to-rod distance
  variab/set/par Ppl 1 ppl             # physical path length
  total_time = total_time + Ppl/clight
  Ppl_Last = Ppl
  if [total_time < time_stop] beer/noprop 1 Length
  wavelength/set 1 1.06 1.6
  prop Length
  wavelength/set 1 1.06 1.
  clap/c/n 1 Apt                       # 3 mm radius clear aperture on rod
  prop 5                               # guess at mirror-to-rod distance
  c ----- outcoupling section
  mirror 1 80
  prop 5
  clap/c/n 1 Apt                       # 3 mm radius clear aperture on rod
  variab/set/par Ppl 1 ppl             # physical path length
  total_time = total_time + (Ppl-Ppl_Last)/clight
  if [total_time < time_stop] beer/noprop 1 Length
  wavelength/set 1 1.06 1.6
  prop Length
  wavelength/set 1 1.06
  prop 5
  plot/w @Name_1.plt
  plot/l 1 xrad=1.1*Apt
  clap/c/n 1 Apt                       # 3 mm radius clear aperture on rod
  c ----- far-field section
  if [!TEST] then
    copy 1 2
    mult 1 trans                       # Fresnel reflection losses
    mult 2 refl list
    title near-field of output, pass @pass
    plot/watch @Name_2.plt
    plot/l 2 xrad=Apt
c  pause 5
    lens 2 focallength
    beams/off 1
    beams/on 2
    prop focallength
    beams/off 2
    beams/on 1
    title far-field of output, pass @pass
    plot/w @Name_3.plt
    plot/l 2
    fitgeo 2
    variab/set Width1 fitxrad
    variab/set Energy 1 energy

```



```

divergence1 = 2*Width1/focallength
add/inc/con 3 2
fitgeo 3
variab/set Width2 fitxrad
divergence2 = 2*Width2/focallength
udata/set pass total_time Energy divergence1 divergence2    # store for the
plot/w @Name_4.plt
title energy vs. time
plot/udata 1 min=0
title instantaneous beam size vs. time, pass = @pass
plot/w @Name_5.plt
plot/udata 2 min=0
title integrated beam size vs. time, pass = @pass
plot/w @Name_6.plt
plot/udata 3 min=0
endif
c ----- continue with resonator field section
macro/end
Length = 5.5
focallength = 20
variab/set/par Units1 1 units
Units2 = Lambda*focallength/Nline/Units1
units/set 3 Units2
clear 3 0
Apt = .2
time_stop = 10e-9 # time at which to turn off gain
clight = 3e10
refl = .65
trans = 1. - refl
nstep = 1
pass = 0
beams/all/off
beams/on 1
G0 = .5          # small signal gain .5/cm
Es = 1e4         # saturation intensity 10,000 cm^2
beer/set g0=G0 es=Es
resonator/name ex80d
gaus/c/c 1 1 .3
TEST = 1
resonator/eigen/test 1
clear 1 0.          # set initial irradiance
noise 1 1e-8
TEST = 0
c
c Set initial population inversion as N x h x frequency
c
c  $N \times h \times \text{fre} = 1e18 \times 6.626e-34 \times 2.83e14 = .1875 \text{ J/cm}^3$ 
c
pass = 0
variab/set/par Ppl_start 1 ppl
resonator/run 20          # run for 20 round-trips

```

Ex80e: Q-switch, YAG laser using a relatively slow Q-switch

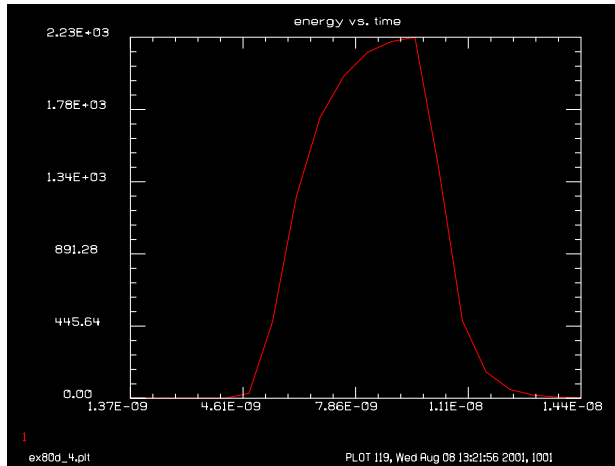


Fig. 80.11. Temporal power pulse for Ex80d, Q-switch with saturable absorber.

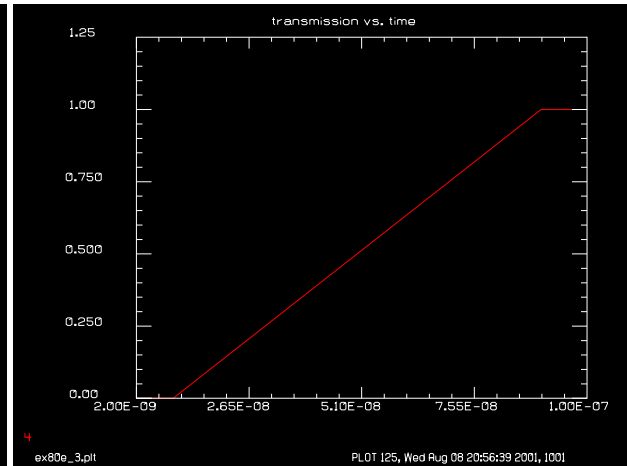


Fig. 80.12. Waveform of transmission function for slow Q-switch of Ex80e.

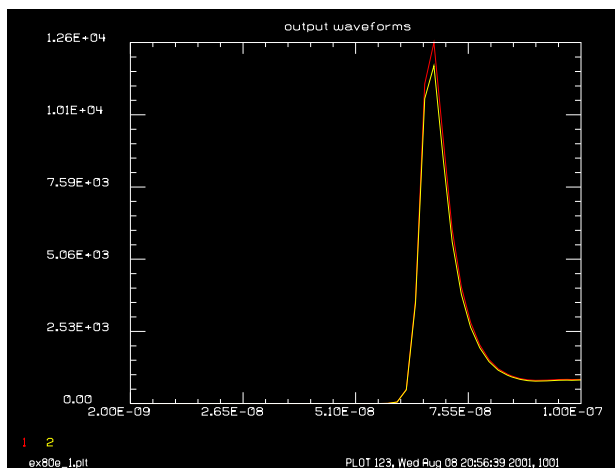


Fig. 80.13. Waveform of population inversion (red) and optical pulse (yellow) for Ex80e. The primary effect of slow turn on as shown in Fig. 80.12 is to delay the onset of the pulse.

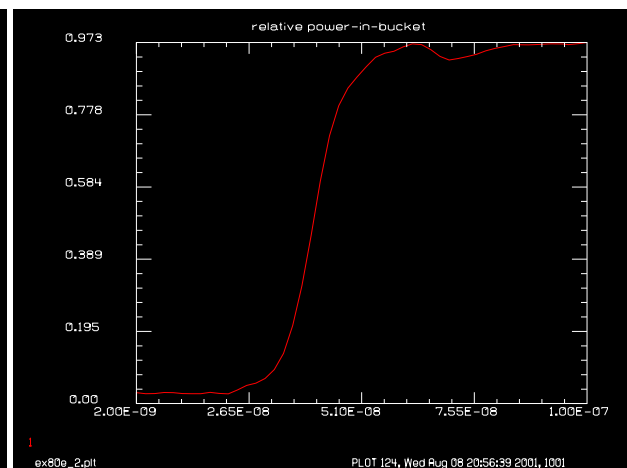


Fig. 80.14. Waveform of power-in-bucket for Ex80e.inp

Input: ex80e.inp

```
c## ex80e
c
c Example 80e: Q-switch, YAG laser
c
c This is an example of a Q-switched, YAG laser, using a relatively
c slow Q-switch, taking 80 nanoseconds to switch
c
c Beams
c -----
```

Jump to: [Commands](#), [Theory](#)

```

c Beam 1    optical field in the cavity
c Beam 2    medium array
c Beam 3    outcoupled beam
c
variab/dec/int pass switch nstep
array/s 1 128 128
nbeam 3          # set number of beams
array/s 2 128 128 1 data # make medium array polarized
units/s 0 .008
wavelength/set 0 1.06 # set wavelength and index
width = 1.2e10 # set line width, hz
center = 2.83e14 # set line center, hz
tspont = 3e-4 # set spontaneous emission time, sec
t20 = 3e-1 # set decay time for level 2, sec
t10 = 1e-5 # set decay time for level 1, sec
mode_sep = 5e8 # set longitudinal mode separation, hz
c
c Set offset from line center of first mode, 0 hz
c Set fractional pumping into level 1, nlpump, 0
c
gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
pack/set 1 2 # pack both beams
plot/watch plot1.plt
macro/def ex80e/o
  mirror/flat 1
  pass = pass + 1 # increment pass counter
  clap/c/n 1 .3 # 3 mm radius clear aperture on rod
  pack/in # pack beams
    wavelength/set 1 1.06 1.6
    gain/rate/step 1 time nstep
    wavelength/set 1 1.06 1.
  pack/out # unpack beams
  trans = ramp(tot_time,start,slope)
  trans = min(1.,trans)
  pass=
  tot_time=
  trans=
  prop 10
  clap/c/n 1 .3 # 3 mm radius clear aperture on rod
  prop 20
c ----- outcoupling section
copy 1 3
mult 1 trans # Fresnel reflection losses
mult 3 refl list
title near-field of output, pass @pass
plot/l 3
pause 5
lens 3 100
beams/off 1
beams/on 3
prop 100
beams/off 3

```

```

beams/on 1
title far-field of output, pass @pass
plot/1 3
variab/set energy1 3 energy list      # record peak irradiance
clap/c/n 3 .125                        # clear aperture subtending 2.5 mr
variab/set energy2 3 energy list      # record peak irradiance
if [energy1 != 0] then
    diff = energy2/energy1
    udata/set pass tot_time energy1 energy2 diff trans # store for the record
endif
tot_time = tot_time + time
switch = mod(pass,5)                    # plot every 5th pass
if switch = 0 then
    title output waveforms
    plot/udata/123 1 2                  # make plot output energy
    bell
    pause 5
    title relative power-in-bucket
    plot/udata/123 3 min=0              # make plot of power-in-the-bucket
endif
c ----- continue with resonator field section
mirror/flat 1
prop 20
clap/c/n 1 .3                          # 3 mm radius clear aperture on rod
pack/in                                # pack beams
    wavelength/set 1 1.06 1.6
    gain/rate/step 1 time nstep
    wavelength/set 1 1.06 1.
pack/out                                # unpack beams
mult 1 trans
prop 10
clap/c/n 1 .3                          # 3 mm radius clear aperture on rod
macro/end
time = 2e-9
tot_time = 0
trans = .95
refl = 1. - trans
nstep = 1
pass = 0
start = 10e-9                          # time to turn on Q-switch
ramp_time = 80e-9                       # time to ramp up
ramp_time = max(time,ramp_time)
stop = start+ramp_time                  # time to finish ramp-up
slope = 1./ramp_time                   # slope of transmission function
beams/all/off
beams/on 1
resonator/name ex80e
gaus/c/c 1 1 .3
resonator/eigen/test 1
pass = 0
clear 1 0.
clear 2 1e3                            # set pumping rate
gain/rate/n2pump 2 2                   # Modify beam 2 to put
                                        # the pump rate in the real word

```

Jump to: [Commands](#), [Theory](#)

```

c
c Pump for 10 microsecond before running macro
c
pack/in                                # pack beams
    wavelength/set 1 1.06 1.6
    gain/rate/step 1e-10 10e-6 10
    wavelength/set 1 1.06 1.
pack/out                                # unpack beams
resonator/run 50                        # run for 50 round-trips
plot/watch ex80e_1.plt
title output waveforms
plot/udata/123 1 2                      # make plot output energy
plot/watch ex80e_2.plt
title relative power-in-bucket
plot/udata/123 3 min=0                  # make plot of power-in-the-bucket
plot/watch ex80e_3.plt
title transmission vs. time
plot/udata/set y04
plot/udata/seq min=0 max=1.25

```

Ex80f: Q-switch, YAG laser pulsed laser diode pumping

Input: `ex80f.inp`

```

c## ex80f
c
c Example 80f: Q-switch, YAG laser, pulsed diode pumping
c
c This is an example of a Q-switched, YAG laser, using a pulsed diode
c laser for pumping.
c
c Beams
c -----
c Beam 1    optical field in the cavity
c Beam 2    medium array
c Beam 3    outcoupled beam
c
variab/dec/int TEST pass switch nstep
array/s 1 128 128
nbeam 3                                # set number of beams
array/s 2 128 128 1 data                # make medium array polarized
units/s 0 .008
wavelength/set 0 1.06                   # set wavelength and index
width = 1.2e10                          # set line width, hz
center = 2.83e14                         # set line center, hz
tspont = 3e-4                           # set spontaneous emission time, sec
t20 = 3e-1                              # set decay time for level 2, sec
t10 = 1e-5                              # set decay time for level 1, sec
mode_sep = 5e8                          # set longitudinal mode separation, hz
length = .5
c
c Set offset from line center of first mode, 0 hz

```

Jump to: [Commands](#), [Theory](#)

```

c Set fractional pumping into level 1, n1pump, 0
c
gain/rate/set width center tspont t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
pack/set 1 2 # pack both beams
plot/watch ex80f_1.plt
macro/def gain_step/o
    tot_time = tot_time + time
    CenterPump = 1e3*gauss(tot_time, CenterTime, half_pulse_width, 1)
    gaus/c 2 CenterPump DiodeWidth # set pumping rate

    gain/rate/n2pump 2 2 # Modify beam 2 to put
                        # the pump rate in the real word
    pack/in # pack beams
        gain/rate/step length time/2
    pack/out # unpack beams
macro/end
macro/def out_coupling/o
    copy 1 3
    mult 1 trans # Fresnel reflection losses
    mult 3 refl list
    title near-field of output, pass @pass
    plot/w ex80f_2.plt
    plot/l 3
    lens 3 100
    beams/off 1
    beams/on 3
    prop 100
    beams/off 3
    beams/on 1
    title far-field of output, pass @pass
    plot/w ex80f_3.plt
    plot/l 3
    variab/set energy1 3 energy list # record peak irradiance
    clap/c/n 3 .125 # clear aperture subtending 2.5 mr
    variab/set energy2 3 energy list # record peak irradiance
    if [energy1 != 0] then
        diff = energy2/energy1
        udata/set pass tot_time energy1 energy2 diff # store for the record
    endif
    switch = mod(pass,5) # plot every 5th pass
    if switch = 0 then
        title output waveforms
        plot/w ex80f_4.plt
        plot/udata/123 1 2 # make plot of output energy
        title relative power-in-bucket
        plot/w ex80f_5.plt
        plot/udata/123 3 min=0 # make plot of power-in-the-bucket
    endif
macro/end
macro/def ex80f/o
    mirror/flat 1
    pass = pass + 1 # increment pass counter

```

Jump to: [Commands](#), [Theory](#)

```

clap/c/n 1 Apt          # clear aperture on rod
if [!TEST] macro gain_step
prop 10
clap/c/n 1 Apt          # clear aperture on rod
prop 20
clap/c/n 1 Apt          # clear aperture on mirror
if [!TEST] macro gain_step
wavelength/set 1 1.06 1.6
prop length
wavelength/set 1 1.06 1.
c ----- outcoupling section
if [!TEST] macro out_coupling
c ----- continue with resonator field section
mirror/flat 1
prop 20
clap/c/n 1 Apt          # clear aperture on rod
if [!TEST] macro gain_step
wavelength/set 1 1.06 1.6
prop length
wavelength/set 1 1.06 1.
mult 1 trans
prop 10
clap/c/n 1 Apt          # clear aperture on rod
macro/end
tot_time = 0
trans = .95
refl = 1. - trans
Apt = .2
nstep = 1
pass = 0
start = 10e-9           # time to turn on Q-switch
full_pulse_width=20e-9  # full width, 1/e, of gaussian time pulse
half_pulse_width=full_pulse_width/2
CenterTime = 40e-9      # time for peak pumping
DiodeWidth = .2         # half-width of diode spatial profile, 1/e
beams/all/off
beams/on 1
resonator/name ex80f
gaus/c/c 1 1 Apt
TEST = 1
resonator/eigen/test 1
resonator/eigen/list
c
c GLAD will calculate roundtrip time
c
variab/set time 1 reson/roundtriptime list
TEST = 0
pass = 0
clear 1 0.
resonator/run 50         # run for 50 round-trips
plot/watch ex80f_4.plt
title output waveforms
plot/udata/123 1 2       # make plot output energy
plot/watch ex80f_5.plt

```

Jump to: [Commands](#), [Theory](#)

```

title relative power-in-bucket
plot/udata/123 3 min=0           # make plot of power-in-the-bucket

```

Ex80g: An example of a gain/absorber, numerical check.

The gain/absorber command models a four-level saturable absorber capable of modeling Cr^{4+} . It may be used in a passive Q-switch. This example compares analytical calculation with the GLAD calculation.

Input: ex80g.inp

```

c## ex80g
c
c Example 80g: Passive Q-switch using gain/absorb
c
c This is an example of a saturable absorber based on gain/absorb.
c
c Case 1: N2 = 0, Es_sigma = 0
c Case 2: N2 = .9*N0, Es_sigma = 0
c Case 3: N2 = .9*N0, Es_sigma = .5*Gs_sigma
c
c Beams
c -----
c Beam 1      optical field in the cavity
c Beam 2      absorber medium array
c
variab/dec/int pass switch TEST
array/s 1 128 128
array/s 2 128 128           # absorber medium array
wavelength/set 0 1.06      # set wavelength and index
width = 1.2e10             # set line width, hz
center = 2.83e14           # set line center, hz
freq = center
hnu = h*freq list
time = 2e-9
N0 = 1e20
N1 = N0
Gs_sigma = 2e-20
Gamma = 1
Transmission = 1
c
C Case 1: N2 = 0, Es_sigma = 0
c
Es_sigma = 0e-20
gain/absorb/set Gs_sigma Es_sigma Gamma Transmission
gain/absorb/list
A1 = 1
I1 = A1^2
N2 = 0
clear/complex 1 A1
clear/complex 2 N0 N2      # N0 in real word, N2 in imaginary word
N1 = N0 - N2

```

Jump to: [Commands](#), [Theory](#)


```

N2 = N2 + Gamma*Gs_sigma*N1*I1/hnu
N1 = N0 - N2
Arg = (-Gs_sigma + Es_sigma)*N1 - Es_sigma*N0
Amp = sqrt(Transmission)*exp(.5*Arg)
A2 = A1*Amp
I2 = A2^2
pack/set 1 2                # pack both beams
pack/in                     # pack beams
    gain/absorber/length 1 time list
pack/out                    # unpack beams
variable/set GLAD_N2 2 64 64 point/si
GLAD_N1 = N0 - GLAD_N2
variable/set GLAD_A2 1 65 65 point/sr
GLAD_I2 = GLAD_A2^2
C
C Case 1: N2 = 0, Es_sigma = 0
C N1 = @N1, GLAD_N1 = @GLAD_N1
C N2 = @N2, GLAD_N2 = @GLAD_N2
C A2 = @A2, GLAD_A2 = @GLAD_A2
C I2 = @I2, GLAD_I2 = @GLAD_I2
C
c
C Case 2: N2 = .9*N0, Es_sigma = 0
c
Es_sigma = 0e-20
gain/absorb/set Gs_sigma Es_sigma Gamma Transmission
gain/absorb/list
A1 = 1
I1 = A1^2
N2 = .9*N0
clear/complex 1 A1
clear/complex 2 N0 N2        # N0 in real word, N2 in imaginary word
N1 = N0 - N2
N2 = N2 + Gamma*Gs_sigma*N1*I1/hnu
N1 = N0 - N2
Arg = (-Gs_sigma + Es_sigma)*N1 - Es_sigma*N0
Amp = sqrt(Transmission)*exp(.5*Arg)
A2 = A1*Amp
I2 = A2^2
pack/set 1 2                # pack both beams
pack/in                     # pack beams
    gain/absorber/length 1 time list
pack/out                    # unpack beams
variable/set GLAD_N2 2 64 64 point/si
GLAD_N1 = N0 - GLAD_N2
variable/set GLAD_A2 1 65 65 point/sr
GLAD_I2 = GLAD_A2^2
C
C Case 2: N2 = .9*N0, Es_sigma = 0
C N1 = @N1, GLAD_N1 = @GLAD_N1
C N2 = @N2, GLAD_N2 = @GLAD_N2
C A2 = @A2, GLAD_A2 = @GLAD_A2
C I2 = @I2, GLAD_I2 = @GLAD_I2
C

```

Jump to: [Commands](#), [Theory](#)

```

C
C Case 3: N2 = .9*N0, Es_sigma = .5*Gs_sigma
C
Es_sigma = .5*Gs_sigma
gain/absorb/set Gs_sigma Es_sigma Gamma Transmission
gain/absorb/list
A1 = 1
I1 = A1^2
N2 = .9*N0
clear/complex 1 A1
clear/complex 2 N0 N2          # N0 in real word, N2 in imaginary word
N1 = N0 - N2
N2 = N2 + Gamma*Gs_sigma*N1*I1/hnu
N1 = N0 - N2
Arg = (-Gs_sigma + Es_sigma)*N1 - Es_sigma*N0
Amp = sqrt(Transmission)*exp(.5*Arg)
A2 = A1*Amp
I2 = A2^2
pack/set 1 2                  # pack both beams
pack/in                       # pack beams
    gain/absorber/length 1 time list
pack/out                      # unpack beams
variable/set GLAD_N2 2 64 64 point/si
GLAD_N1 = N0 - GLAD_N2
variable/set GLAD_A2 1 65 65 point/sr
GLAD_I2 = GLAD_A2^2
C
C Case 3: N2 = .9*N0, Es_sigma = .5*Gs_sigma
C N1 = @N1, GLAD_N1 = @GLAD_N1
C N2 = @N2, GLAD_N2 = @GLAD_N2
C A2 = @A2, GLAD_A2 = @GLAD_A2
C I2 = @I2, GLAD_I2 = @GLAD_I2
C

```

Ex80h: An example of a saturable absorber based on gain/absorber

The gain/absorber command models a four-level saturable absorber capable of modeling Cr^{4+} . It may be used in a passive Q-switch. This example illustrates a simple Q-switched resonator with passive saturable absorber as the switch.

Input: ex80h.inp

```

c## ex80h
c
c Example 80h: Passive Q-switch using gain/absorb
c
c This is an example of a Q-switched, YAG laser, using a saturable
c absorber based on gain/absorb.
c
c Beams
c ----
c Beam 1      optical field in the cavity

```

Jump to: [Commands](#), [Theory](#)

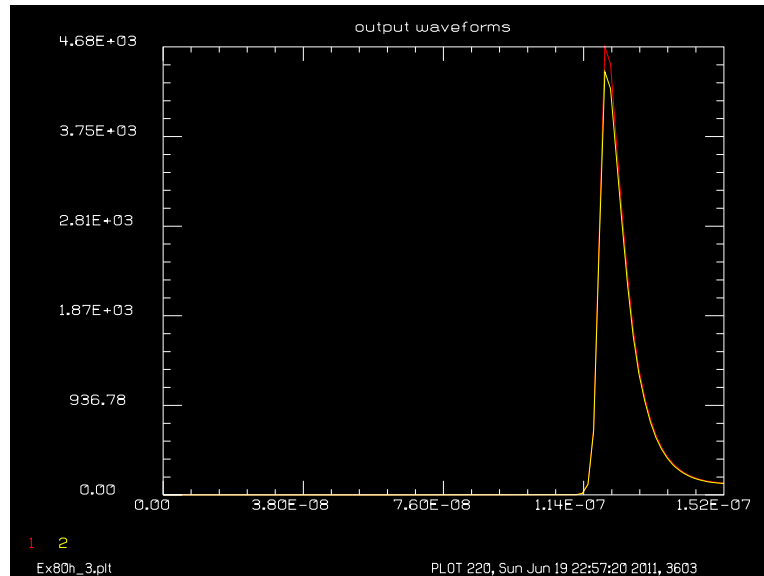


Fig. 80.15. Output power of Q-switch versus time for a saturable absorber Q-switch and the output power falling within 2.5 milliradians, from ex80c.inp. It is assumed that the medium is pumped at a constant rate. The delay of the pulse is due to the build up of the optical field to the point where the saturable absorber becomes saturated.

```

c Beam 2    gain medium array
c Beam 3    absorber medium array
c Beam 4    outcoupled beam
c
alias Name Ex80h
variab/dec/int pass SWITCH TEST SWITCH1
SWITCH1 = 0                # 0 passive absorber, 1 ideal switch
array/s 1 128 128
array/s 2 128 128 1 data   # make gain medium array polarized
array/s 3 128 128 data     # absorber array
nbeam 4                    # set number of beams
units/s 0 .008
Lambda = 1.06e-4
wavelength/set 0 Lambda*1e4 # set wavelength and index
width = 1.2e10              # set line width, hz
center = c/Lambda list
tspont = 4e-4               # set spontaneous emission time, sec
t20 = 3e-1                  # set decay time for level 2, sec
t10 = 1e-5                  # set decay time for level 1, sec
mode_sep = 5e8              # set longitudinal mode separation, hz
Pump = 0.8e4
trans = .90
refl = 1. - trans
Apt = .1
L_gain = 6
L_air = 40
Gs_sigma = .2e-21
Es_sigma = 0.1*Gs_sigma
Gamma = 1.
Transmission = 1.
N0_start = 2E21

```

Jump to: [Commands](#), [Theory](#)

```

N2_start = 0.
c
c Set offset from line center of first mode, 0 hz
c Set fractional pumping into level 1, nlpump, 0
c
gain/rate/set width center tspont t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
gain/absorb/set Gs_sigma Es_sigma Gamma Transmission
gain/absorb/list
macro/def ex80c/o
  mirror/flat 1
  pass = pass + 1          # increment pass counter
  clap/c/n 1 .3           # 3 mm radius clear aperture on rod
  wavelength/set 1 1.06 1.6
  pack/set 1 2            # pack both beams
  pack/in                 # pack beams
    gain/rate/step L_gain-1 time
  pack/out
  if SWITCH1 = 0 then
    pack/set 1 3          # pack both beams
    pack/in              # pack beams
    gain/absorber/length 1 time
    pack/out             # unpack beams
  else
    if pass<19 then
      mult 1 0.
    endif
  endif
  endif
  wavelength/set 1 1.06 1.
  prop L_gain
  clap/c/n 1 Apt          # clear aperture on rod
  prop L_air
  c ----- outcoupling section
  copy 1 4
  mult 1 trans            # Fresnel reflection losses
  mult 4 refl list
  plot/w @Name_1.plt
  title near-field of output, pass @pass
  plot/l 4
  lens 4 100
  beams/off 1
  beams/on 4
  prop 100
  beams/off 4
  beams/on 1
  plot/w @Name_2.plt
  title far-field of output, pass @pass
  plot/l 4
  variab/set energy1 4 energy list # record peak irradiance
  clap/c/n 4 .125             # clear aperture subtending 2.5 mr
  variab/set energy2 4 energy list # record peak irradiance
  if [energy1 != 0] then
    diff = energy2/energy1

```

Jump to: [Commands](#), [Theory](#)

```

    udata/set pass tot_time energy1 energy2 diff    # store for the record
endif
if TEST=0 tot_time = tot_time + time
SWITCH = mod(pass,5)                                # plot every 5th pass
if SWITCH = 0 then
    plot/w @Name_3.plt
    title output waveforms
    plot/udata/123 1 2                                # make plot output energy
endif
c ----- continue with resonator field section
mirror/flat 1
prop L_air
clap/c/n 1 Apt                                # 3 mm radius clear aperture on rod
wavelength/set 1 1.06 1.6
pack/set 1 2                                # pack both beams
pack/in                                # pack beams
    gain/rate/step L_gain-1 time
pack/out                                # unpack beams
if SWITCH1 = 0 then
    pack/set 1 3                                # pack both beams
    pack/in                                # pack beams
    gain/absorber/length 1 time
    pack/out                                # unpack beams
else
    if pass<19 then
        mult 1 0.
    endif
endif
wavelength/set 1 1.06 1.
prop L_gain
clap/c/n 1 .3                                # 3 mm radius clear aperture on rod
macro/end
tot_time = 0
pass = 0
beams/all/off
beams/on 1
resonator/name ex80c
gaus/c/c 1 1 .3
TEST = 1
resonator/eigen/test 1
TEST = 0
resonator/eigen/list
variable/set TimeFull reson/roundtriptime
time = TimeFull/2
pass = 0
variable/set Units 1 units
c gaus/c 1 1e-4 10*Units
clear 1 0
clear 2 Pump                                # set pumping rate
gain/rate/n2pump 2 2                        # Modify beam 2 to put
                                            # the pump rate in the real word
clear/complex 3 N0_start N2_start          # total ion density in real word
write/off
resonator/run 100

```

Jump to: [Commands](#), [Theory](#)

write/on
udata/list

Ex81: Illustration of zone control of propagation

Table. 81.1. Table of Ex81 examples

Ex81a: Illustration of zone command	1
Ex81b: Through-focus image of circular aperture.....	5
Ex81c: Lensarray used as optical integrator	8

This example illustrates the use of the zone command. Normally GLAD determines the propagation algorithms to minimize the aliasing. This is done independently for each beam and if the wavelength and focusing properties are different for each beam, then the array sizes for various beams may be different. Interactions between beams such as is required for Raman kinetics may be done using the `pack/in` and `pack/out` feature, which will automatically invoke interpolation to bring the beams into a common coordinate system. When this process is done many times, as in Example 19, the interpolation errors contribute substantial errors. The problem is exacerbated because the Raman wavelengths are different so the Rayleigh zone is different and one beam may be propagated with the near-field propagator and the other with the far-field propagator. Putting both beams under zone control will assure the array sizes and propagation algorithms are always identical.

In the first part of Example 81, the zone is centered at the paraxial focus. See Fig. 81.1. The zone width is set to 3.5, greater than the 3.1 width normal zone determined by the Rayleigh distance. This extends the zone of constant units and also increases the array size. Consequently the beam appears smaller in the array as shown in Fig. 81. 2. In the second part of Example 81, the zone is shifted backward 1 cm and GLAD interpolates during the `zone/in` command and `zone/out` command to transfer from gaussian to zone control and return, respectively. GLAD determines the limits of the zone by matching the two methods at the negative limit of the zone.

Ex81a: Illustration of zone command

Input: `ex81a.inp`

```
c## ex81a
c
c Example 81a: Illustration of zone command
c
c The zone command allows the user to override the surrogate gaussian
c beam control. With the ZONE command the user may specify
c the boundaries for plane-to-curved reference surface interchange
c which are normally placed at the Rayleigh range of the surrogate
c gaussian beam.
c
c In case 1, the zone is centered at the paraxial focus.
c In case 2, the zone is shifted 1 cm from paraxial focus.
c
macro/def zone/o
  pass = pass + 1
  z = z + dz
  dist dz
  variab/set units1 1 units
```

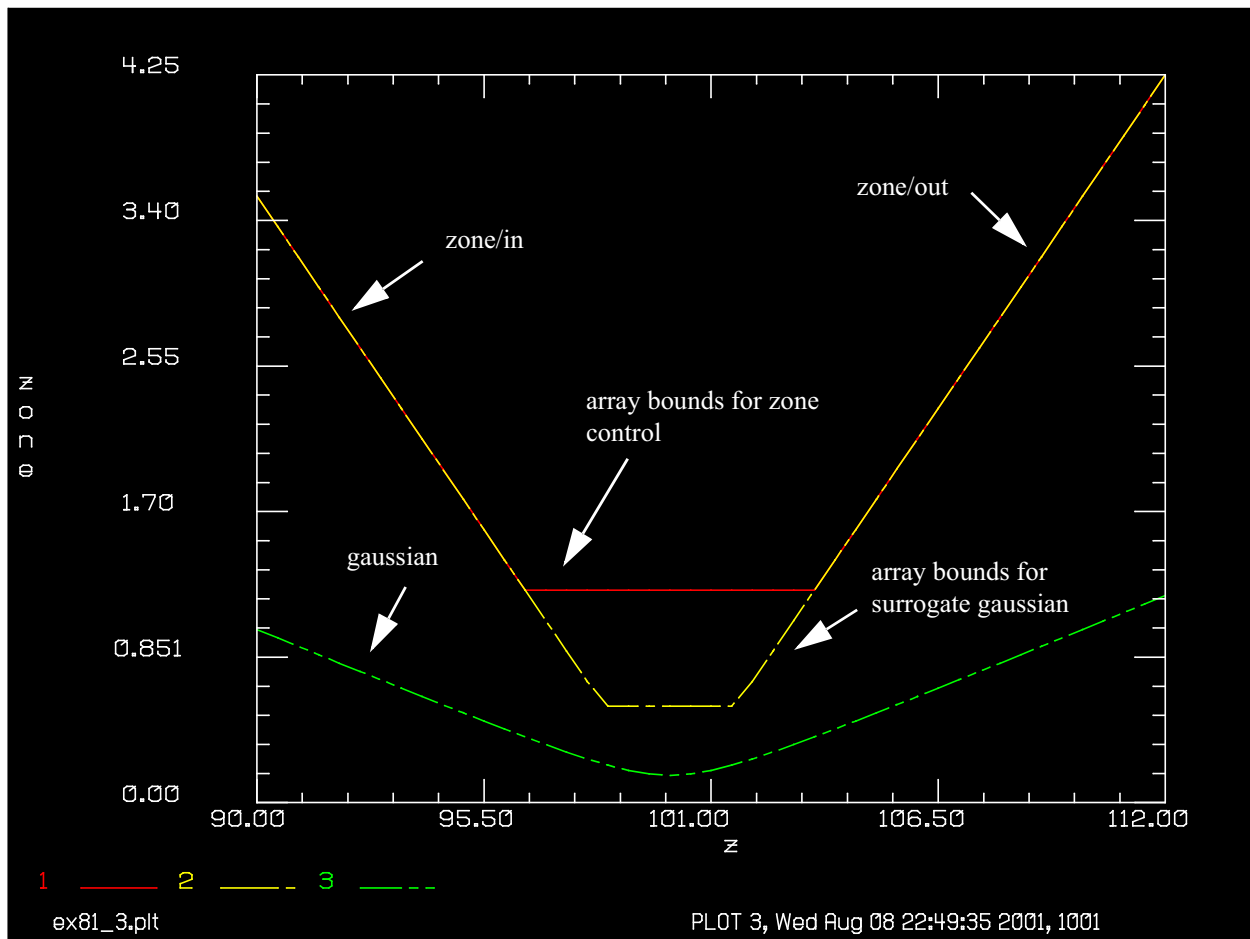


Fig. 81.1. The array bounds for zone control are shown as a solid line. The normal array bounds, as determined by the surrogate gaussian beam are shown as a dashed line. The radius of a gaussian beam is shown as it propagates along the optical axis.

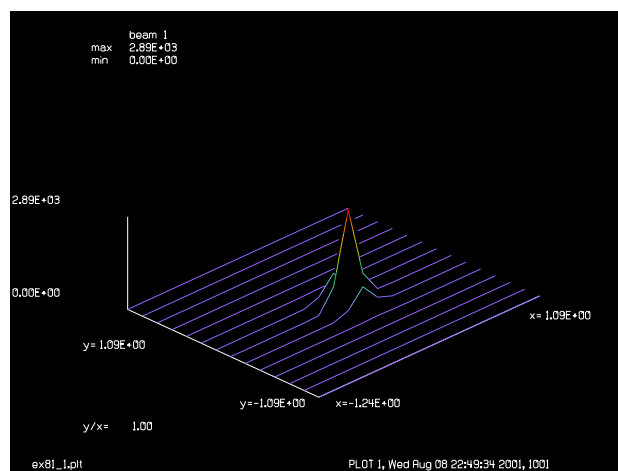


Fig. 81.2. The array is large under zone control.

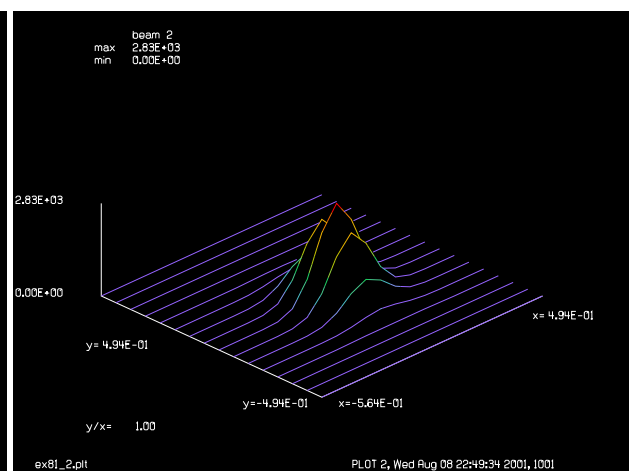


Fig. 81.3. The array, under normal control, is smaller than the zonal control of Fig. 81.2.

variab/set units2 2 units

Jump to: [Commands](#), [Theory](#)

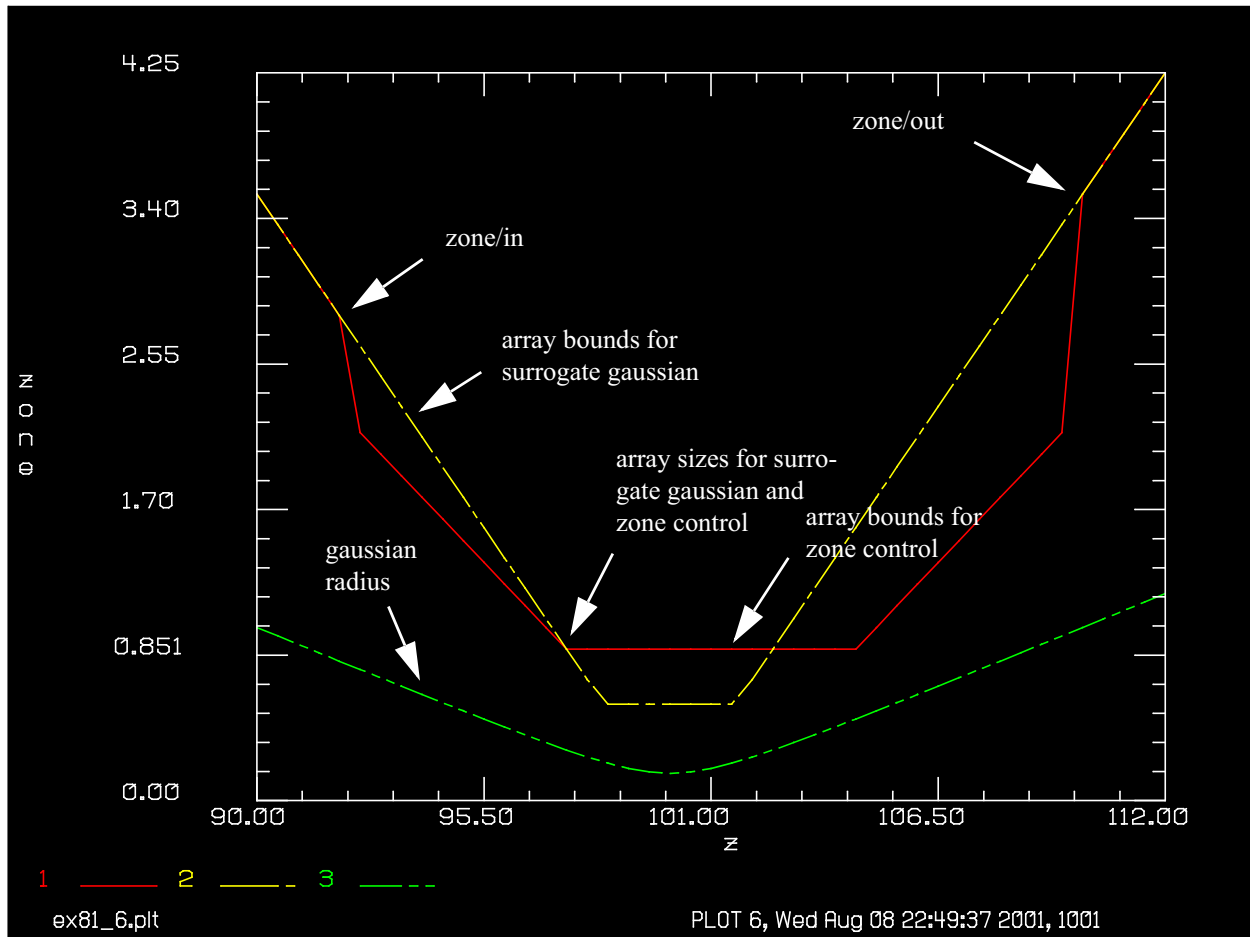


Fig. 81.4. When the zone is shifted backward, GLAD interpolates to the zone boundary for zone/in and back to the regular boundary for zone/out.

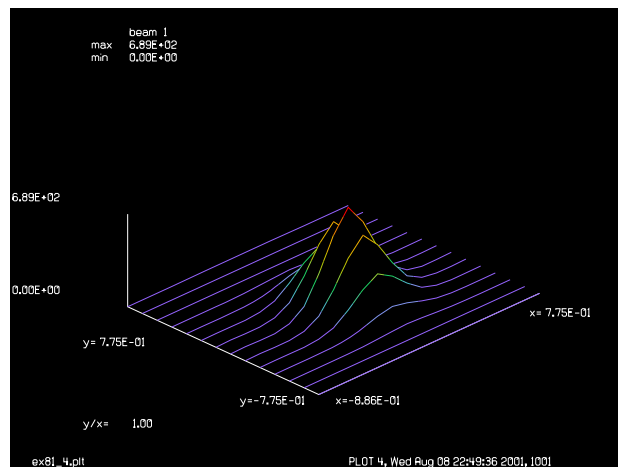


Fig. 81.5. The array is somewhat larger under zone control, but not as much as in Fig. 81.2.

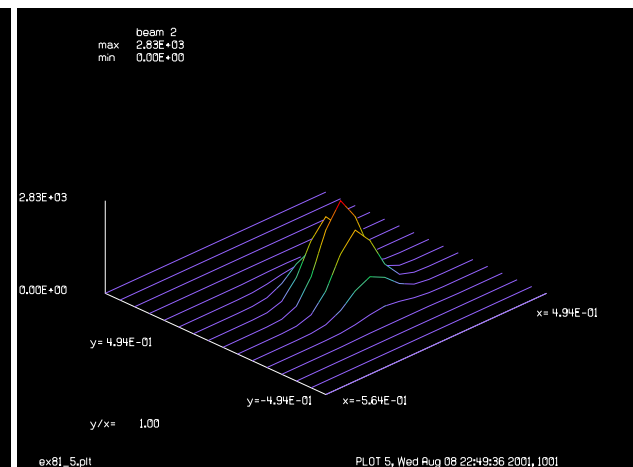


Fig. 81.6. The array, under normal control, is the same as Fig. 81.3.

```
variab/set bmsize 1 bmsize
udata/set pass z units1*size/2 units2*size/2 bmsize
```

Jump to: [Commands](#), [Theory](#)

```

macro/end
dz = .5           # define incremental propagation distance
start = 89.5      # define propagation start distance, 87.5
size = 16         # define array size
nbeam 2
array/s 0 size    # establish array size
wavelength/set 0 500 # set wavelength to 500 microns
gauss/c/r 0 1 10
dist 100
lens 0 100
dist start
zone/fix 200 3.5  # define zone at paraxial focus
pass = 0
z = start
macro/run zone/5
zone/in 1         # enter zone control
macro/run zone/18
plot/watch ex81a_1.plt
plot/l 1
plot/watch ex81a_2.plt
plot/l 2
macro/run zone/17
zone/out 1        # exit from zone control
macro/run zone/5
plot/watch ex81a_3.plt
udata/xlabel z axis
udata/ylabel zone size and gau. radius
plot/udata 1 3 min=0 dash
c
c Re run with shifted zone.
c
udata/clear
zreff/se 1 0.
zreff/se 2 0.
array/s 0 size    # establish array size
wavelength/set 0 500 # set wavelength to 500 microns
gauss/c/r 0 1 10
dist 100
lens 0 100
dist start
zone/unfix
zone/fix 201 3.5  # define zone at paraxial focus
pass = 0
z = start
macro/run zone/5
zone/in 1         # enter zone control
macro/run zone/18
plot/watch ex81a_4.plt
plot/l 1
plot/watch ex81a_5.plt
plot/l 2
macro/run zone/17
zone/out 1        # exit zone control
macro/run zone/5

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch ex81a_6.plt
plot/udata 1 3 min=0 dash
end

```

Ex81b: Through-focus image of circular aperture

The through-focus diffraction pattern of a circular aperture is calculated to illustrate the tendency of a focused beam to have the most narrow width in front of the paraxial focus position. Higher M^2 numbers may result from aperture clipping, an apodized pupil, strong and/or high frequency aberration, diffraction gratings, complex apertures, etc. A surrogate gaussian beam with effective wavelength $\lambda_{\text{eff}} = M^2 \lambda$ serves as an approximate indicator of the beam width versus distance. Higher M^2 numbers indicate a more shallow focus that is shifted more strongly from the paraxial focus toward the aperture. Somewhat lower aliasing may be achieved by shifting the center of the collapsing coordinate system to the center of the surrogate gaussian waist rather than the paraxial focus.

Example 81b shows the case of a uniformly filled circular aperture located relatively close to the paraxial focus. The M^2 is approximately 3 which in this case is shifted from $z = 100$ cm to approximately $z = 74.2$ cm.

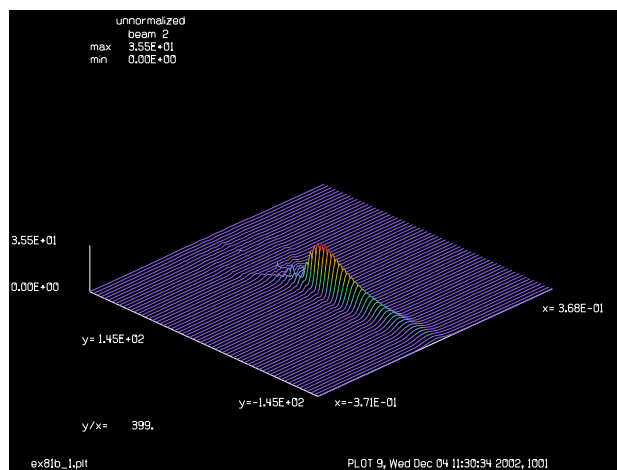


Fig. 81.7. Through-focus plot of uniformly filled circular aperture. Display is centered about the surrogate gaussian waist at $z = 74.2$ cm.

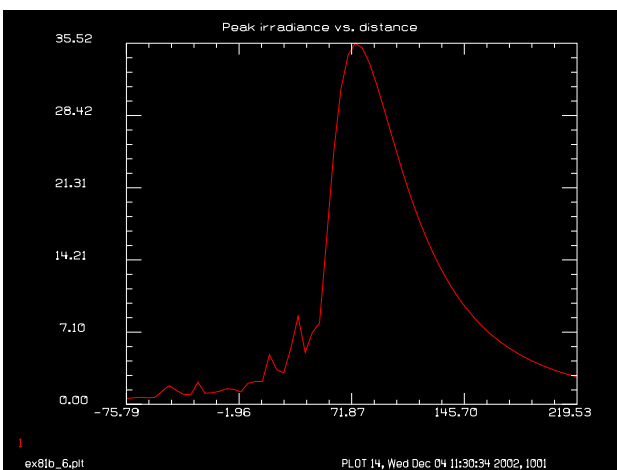


Fig. 81.8. Peak intensity vs. focus position, showing the peak very close to the surrogate gaussian waist at $z = 74.2$ cm.

Input: ex81b.inp

```

c## ex81b
c
c Example 81b: Through-focus image of circular aperture
c
c This example illustrates the formation of the far-field pattern
c of for a uniformly filled clear aperture placed relatively close
c to the focus.
c
variable/dec/int row Nline Kprop Kslice1 Kslice2 Ksum1 Ksum2 Ksum3 Khalf Kgauss
variable/dec/int Kgauss1

```

Jump to: [Commands](#), [Theory](#)

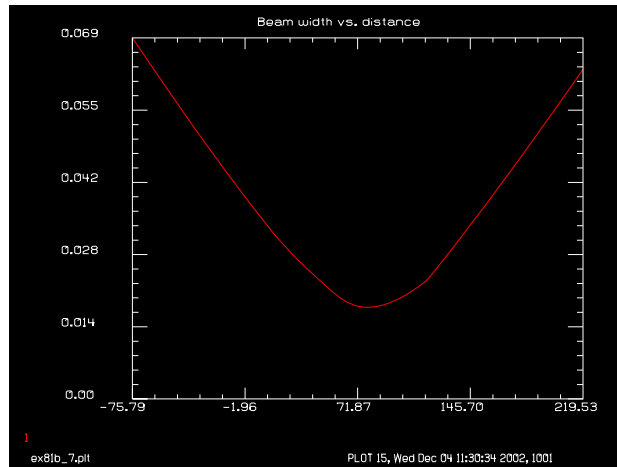


Fig. 81.9. Plot of average radius versus distance from lens. The geometric focus is at $z = 100$ cm. The minimum waist position is very close to the surrogate gaussian waist at $z = 74.2$ cm.

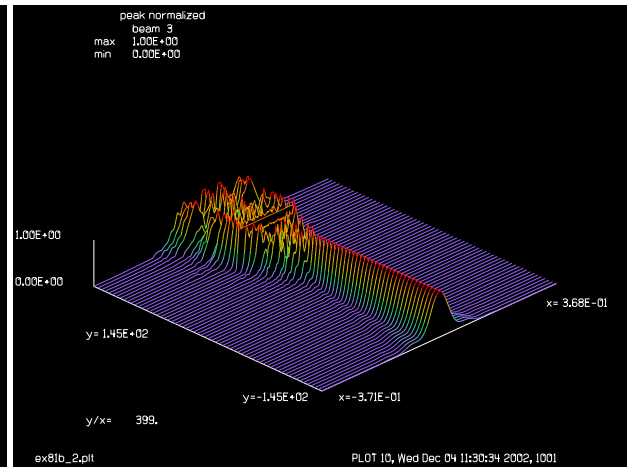


Fig. 81.10. Through-focus plot with peak normalization of each row to better show beam shape.

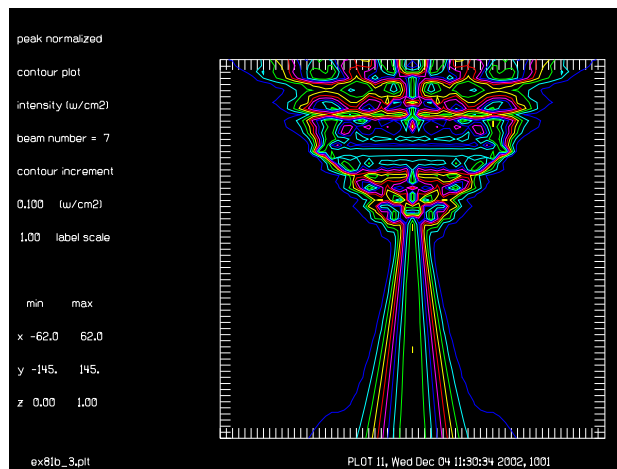


Fig. 81.11. Contour display of through-focus plot with peak normalization and showing only the inner half width in the x-direction. The plane of the lens appears as a straight line.

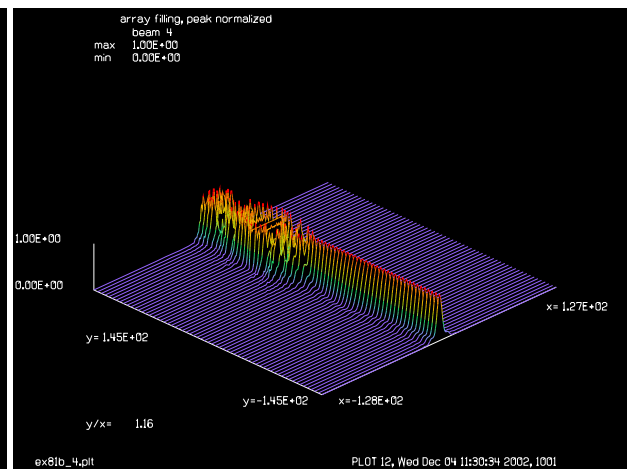


Fig. 81.12. Through-focus plot with peak normalized rows and scaled in the x-direction to the full array width at that x-position. This display shows the degree of filling of the array is relatively constant versus distance.

```
nbeam 9
Kprop = 1
Ksum1 = 2
Ksum2 = 3
Ksum3 = 4
Kslice1 = 5
Kslice2 = 6
Khalf = 7
Kgauss = 8
Kgauss1 = 9
Nline = 256
```

Jump to: [Commands](#), [Theory](#)

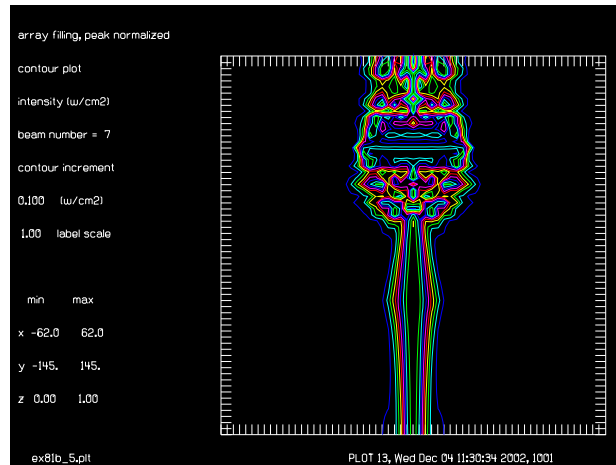


Fig. 81.13. Through-focus plot with peak normalized rows and scaled in the x-direction to the full array width at that x-position. Same data as Fig. 81.12. This display shows the degree of filling of the array is relatively constant versus distance.

```

array/set Kprop Nline
array/set Ksum1 Nline 64 data
array/set Kslice1 Nline 1 data
array/set Ksum2 Nline 64 data
array/set Ksum3 Nline 64 data
array/set Kslice2 Nline 1 data
array/set Khalf Nline/2 64 data
array/set Kgauss Nline
Apt = .09
Units = .005
Lambda = .5e-4
units/s Kprop Units           # Set units.
units/s Kgauss Units
units/s Kgauss1 Units
wavelength/set 0 Lambda*1e4    # Set wavelength.
clap/cir/con Kprop Apt
variab/set Msqx Kprop msqx
wavelength/set Kgauss Msqx*Lambda*1e4
gaus/c Kgauss 1 Apt
lens Kprop 100
lens Kgauss 100
copy Kprop Kgauss1
fitegaus/automatic/set Kgauss1
variable/set Zwaistx Kprop geodata/zwaistx
prop Zwaistx
variab/set Units2 Kprop units
dZ = 300/64
units/s Ksum1 Units2 dZ
units/s Ksum2 Units2 dZ
units/s Ksum3 1 dZ
units/s Kslice1 Units2 1
units/s Kslice2 1 1
clear Ksum1 0
prop -33*dZ                    # Move within 4 cm of focus.

```

Jump to: [Commands](#), [Theory](#)

```

macro/def ex81b/o
  prop dZ
  variab/set Peak Kprop peak
  fitgeo Kprop
  variab/set Width Kprop fitxrad
  fitgeo Kgauss
  variab/set Width_gauss Kgauss fitxrad
  fitgeo Kgauss1
  variab/set Width_gauss1 Kgauss fitxrad
  row = row + 1 list
  zreff
  variab/set Zreff 1 zreff
  udata/set row Zreff Peak Width Width_gauss Width_gauss1
  copy/row Kprop Kslice1 Nline/2+1 1
  copy/con Kslice1 Kslice2
  variab/set Units3 Kprop units
  units/set Kslice1 Units3 1
  rescale/units Kslice1 Units2 1
  copy/row Kslice1 Ksum1 1 row
  peak/norm Kslice1 1
  copy/row Kslice1 Ksum2 1 row
  peak/norm Kslice2 1
  copy/row Kslice2 Ksum3 1 row
macro/end
row = 0
macro ex81b/64
plot/watch ex81b_1.plt
title unnormalized
plot/l Ksum1 h=.2 ns=64
plot/watch ex81b_2.plt
copy/con Ksum2 Khalf
title peak normalized
plot/l Ksum2 h=.2 ns=64
plot/watch ex81b_3.plt
plot/c Khalf ilab=0
copy/con Ksum3 Khalf
plot/watch ex81b_4.plt
title array filling, peak normalized
plot/l Ksum3 h=.2 ns=64
plot/watch ex81b_5.plt
plot/c Khalf ilab=0
plot/watch ex81b_6.plt
title Peak irradiance vs. distance
plot/udata 1 min=0
plot/watch ex81b_7.plt
title Beam width vs. distance: (1) top hat, (2) surrogate gaussian
plot/udata/set y02 y03 y04
plot/udata/seq min=0

```

Ex81c: Lensarray used as optical integrator

Jump to: [Commands](#), [Theory](#)

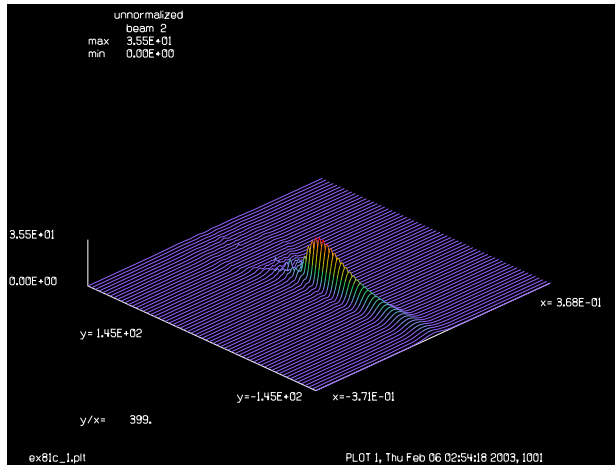


Fig. 81.14. Through-focus plot of uniformly filled circular aperture. Display is centered about the surrogate gaussian waist at $z = 74.2$ cm.

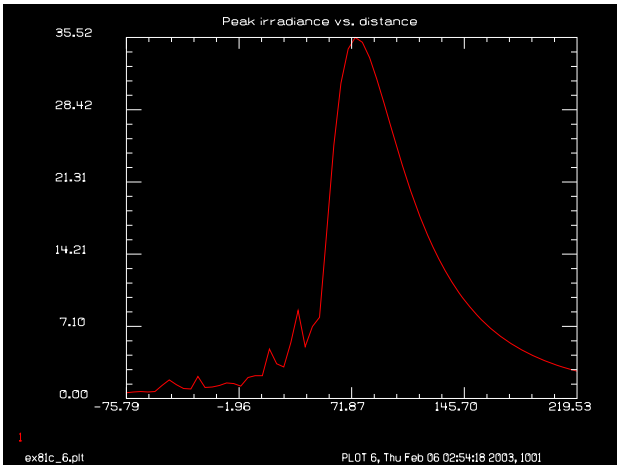


Fig. 81.15. Through-focus plot with peak normalization of each row to better show beam shape.

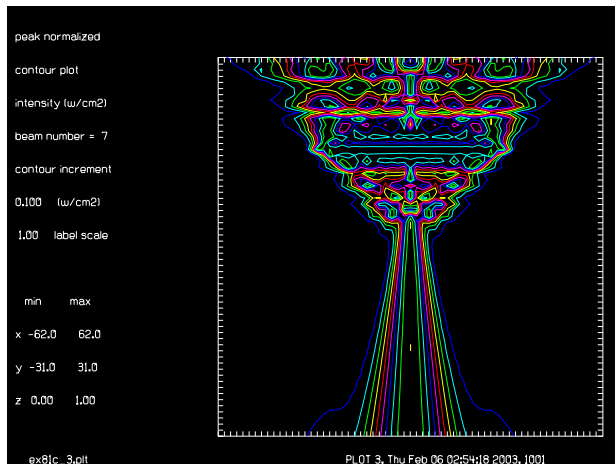


Fig. 81.16. Contour display of through-focus plot with peak normalization and showing only the inner half width in the x-direction. The plane of the lens appears as a straight line.

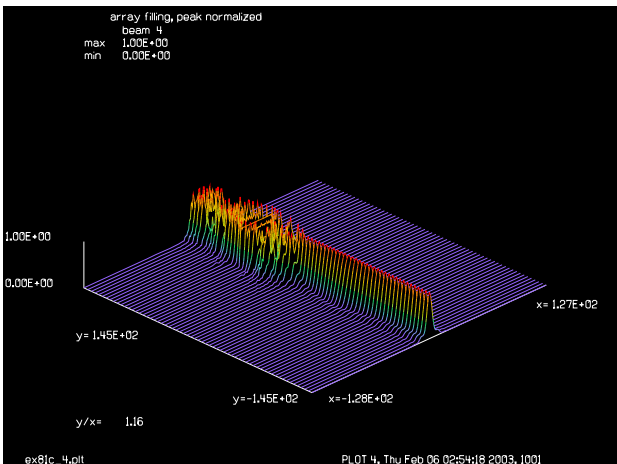


Fig. 81.17. Through-focus plot with peak normalized rows and scaled in the x-direction to the full array width at that x-position. This display shows the degree of filling of the array is relatively constant versus distance.

Input: `ex81c.inp`

```
c## ex81c
c
c Example 81c: Through-focus image of circular aperture
c
c This example illustrates the formation of the far-field pattern
c of for a uniformly filled clear aperture with random aberration
c placed relatively close to the focus.
c
variab/dec/int row Nline Kprop Kslice1 Kslice2 Ksum1 Ksum2 Ksum3 Khalf Kgauss
nbeam 8
```

Jump to: [Commands](#), [Theory](#)

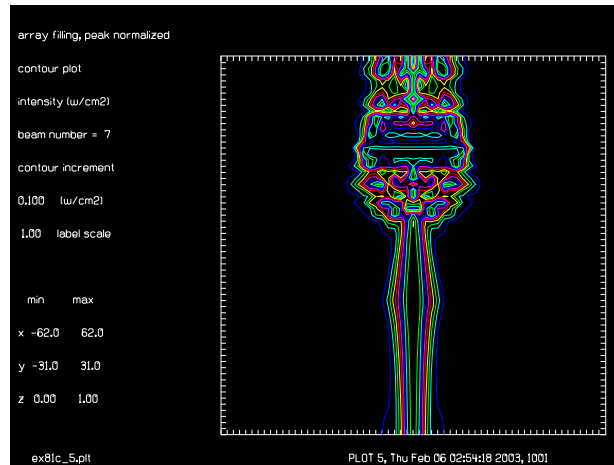


Fig. 81.18. Through-focus plot with peak normalized rows and scaled in the x-direction to the full array width at that x-position. Same data as Fig. 81.12. This display shows the degree of filling of the array is relatively constant versus distance.

```

Kprop = 1
Ksum1 = 2
Ksum2 = 3
Ksum3 = 4
Kslice1 = 5
Kslice2 = 6
Khalf = 7
Kgauss = 8
Nline = 256
array/set Kprop Nline
array/set Ksum1 Nline 64 data
array/set Kslice1 Nline 1 data
array/set Ksum2 Nline 64 data
array/set Ksum3 Nline 64 data
array/set Kslice2 Nline 1 data
array/set Khalf Nline/2 64 data
array/set Kgauss Nline
Apt = .09
Units = .005
Lambda = .5e-4
units/s Kprop Units           # Set units.
units/s Kgauss Units
wavelength/set 0 Lambda*1e4   # Set wavelength.
clap/cir/con Kprop Apt
if 0 then
    fitegaus/auto/list/separate 1 averageradius
    variable/set W0 fitegaus/waist list
    variable/set Zwaist fitegaus/zwaist list
    variable/set Msqx fitegaus/msqx list
    Fitxomega = W0
else
    fitgeo Kprop
    variab/set Fitxomega fitxomega
    variable/set Msqx Kprop msqx

```

Jump to: [Commands](#), [Theory](#)


```

endif
gaus/c Kgauss 1 Fitxomega
wavelength/set Kgauss Msqx*Lambda*1e4
lens Kprop 100
lens Kgauss 100
variable/set Zwaistx Kprop geodata/zwaistx
prop Zwaistx
variab/set Units2 Kprop units
dZ = 300/64
units/s Ksum1 Units2 dZ
units/s Ksum2 Units2 dZ
units/s Ksum3 1 dZ
units/s Kslice1 Units2 1
units/s Kslice2 1 1
clear Ksum1 0
prop -33*dZ                                # Move within 4 cm of focus.
macro/def ex81c/o
    prop dZ
    variab/set Peak Kprop peak
    fitgeo Kprop
    variab/set Width Kprop fitxomega
    fitgeo Kgauss
    variab/set Width_gauss Kgauss fitxomega
    row = row + 1 list
    zreff
    variab/set Zreff 1 zreff
    copy/row Kprop Kslice1 Nline/2+1 1
c copy to Kslice2 for uniform pixel display to show array filling
    copy/con Kslice1 Kslice2
c Update Units3 to have current units for Kprop
    variab/set Units3 Kprop units
    units/set Kslice1 Units3 1
c Rescale Kslice1 to have Units2
    Ratio = Units3/Units2
    udata/set row Zreff Peak Width Width_gauss Ratio
    rescale/units Kslice1 Units2 1
    copy/row Kslice1 Ksum1 1 row
    peak/norm Kslice1 1
    copy/row Kslice1 Ksum2 1 row
    peak/norm Kslice2 1
    copy/row Kslice2 Ksum3 1 row
macro/end
row = 0
macro ex81c/64
plot/watch ex81c_1.plt
title unnormalized
plot/1 Ksum1 h=.2 ns=64
plot/watch ex81c_2.plt
copy/con Ksum2 Khalf
title peak normalized
plot/1 Ksum2 h=.2 ns=64
plot/watch ex81c_3.plt
plot/c Khalf ilab=0
copy/con Ksum3 Khalf

```

Jump to: [Commands](#), [Theory](#)

```
plot/watch ex81c_4.plt
title array filling, peak normalized
plot/l Ksum3 h=.2 ns=64
plot/watch ex81c_5.plt
plot/c Khalf ilab=0
plot/watch ex81c_6.plt
title Peak irradiance vs. distance
plot/udata 1 min=0
plot/watch ex81c_7.plt
title Beam width vs. distance: (1) top hat, (2) surrogate gaussian
plot/udata 2 3 min=0
```

Ex82: Table building feature

This example illustrates the table command of GLAD. This command may be used to read user-defined input into the GLAD program. Some sample input follows:

```
1,2,3,4,5,6,7,8,9
10 11 12      13, 14,      15,16,17,18
19 20 21      22 23  24      25 26 27
28 29 , 30, 31, 32, 33, 34 35 36
```

Input: ex82.inp

```
c## ex82!505854722147895
c
c Example 82: Table Reading Features
c
c one can read in values from a table in a data file
c using table/read.
c
c define table file name, file is always rewound
c
table/name table.dat
c
c read in nine column values into matching variables
c
table/read c1=c1 c2=c2 c3=c3 c4=c4 c5=c5 c6=c6 c7=c7 c8=c8 c9=c9
c
c display the 9 values
variab
pause
c
c read in 9 more values in reverse order
c
table/read c9 c8 c7 c6 c5 c4 c3 c2 c1
variab
pause
c
c skip a table record
c
table/skip 1
c
c read column 8 into c99
c
table/read c8=c99
echo/on
c99=                                # should be 35
pause
end
c
c make a file called table.dat out of the following lines
1,2,3,4,5,6,7,8,9
10 11 12      13, 14,      15,16,17,18
19 20 21      22 23  24      25 26 27
```

28 29 , 30, 31, 32, 33, 34 35 36

Ex83: Partial coherence

Table. 83.1. Table of Ex83 examples

Ex83a: Partial coherent imaging of a three-bar pattern	4
Ex83b: Two sets of seven bars, above and below the coherent resolution limit	8
Ex83c: Thirteen-bar pattern to compare with theory	15

This example illustrates how partial coherence may be modeled using a full diffraction analysis. It is assumed that the system is configured as a Kohler illumination system with condenser optics to relay an incoherent source into the pupil of a relay lens, as illustrated in Fig. 83.1. As in a classical Kohler illumination system, a source point is imaged by a condenser mirror into the pupil of a relay optic. The source illuminates an object mask which is relayed to the desired magnification at the final image plane. In order to properly sample the optical beam, the beam is initiated at the conjugate of the object mask so that there is a source point of finite dimension at the source plane, rather than an ideal point source which would induce aliasing. For a source of finite size, the image is partially coherent. If the source is of sufficient size such that it fills the pupil of the relay lens, then the imaging is said to be incoherent.

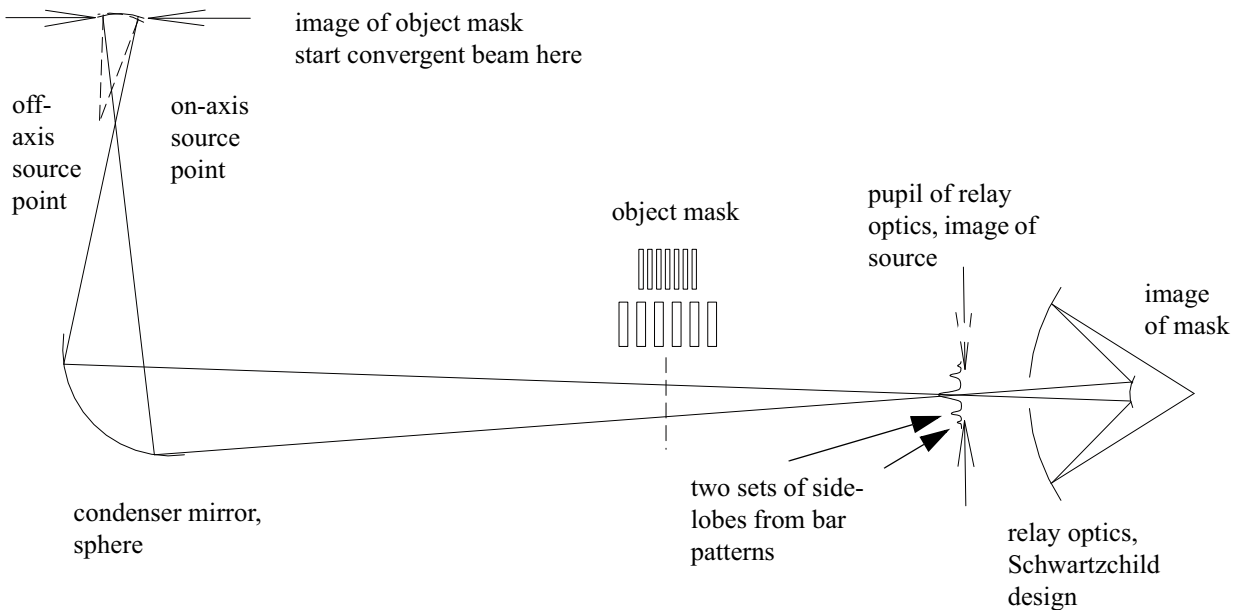


Fig. 83.1. The optical system consists of a condenser mirror, a tipped sphere, and a relay mirror of Schwartzchild design. The source is imaged into the pupil of the relay mirror. The beam is started at the conjugate of the object mask and is imaged by the condenser mirror to the object mask, where the transmission mask is located. The beam is converging at the object mask and forms a point image at the pupil of the relay lens. A multi-bar pattern will form a central lobe and side lobes at the pupil of the relay lens. Different source points are simulated by tilting the initial beam where it is created at the conjugate of the object mask, as shown by the off-axis source point shown as dotted lines.

The partially coherent system may be modeled by treating the source as a series of points, each of which is traced through the system and the resulting image is summed incoherently. Since the light must pass through the full system for each source point, it is important to minimize the number of source points. A

completely coherent system requires only source point, a partially coherent configuration may require about 10 points and an incoherent system may require up to 50 points. Figure 83.2 may help to explain the source sampling requirements. Consider two sets of bars which are to be located at the object mask. We select narrow bars which are just outside the coherent MTF and wider bars which are just inside. The bar pattern has a point spread function (PSF) in the pupil of the relay consisting of a central lobe and side lobes. For the narrow bars (with spatial frequency greater than the coherent MTF). The side lobes are separated by more than the radius of the pupil. An on-axis source will place the central lobe in the middle of the pupil and the side lobes will not pass through the pupil. There will, therefore, be no modulation in the image—zero modulation. The wider bar pattern was specified so that the PSF for an on-axis point will pass through the pupil—100% modulation for the on-axis point, the coherent imaging case. A source point which is sufficiently off-axis, will have the central lobe and one side lobe pass through the pupil, yielding some modulation. Such off-axis source points will contribute to partial coherent imaging and will provide better imaging than the on-axis point. An average over a finite-sized source will yield a partial coherent image as defined by the relative pupil filling factor.

For the narrow bars, the locus of points which will pass the center lobe and one side lobe form two lens-like regions in the pupil. If the central lobe falls within the regions marked 0.5, they will contribute “half” modulation. Locating the central lobe outside these regions will not contribute to the modulation of the images of the bars. In order to adequately represent the degree of coherence, it is only necessary to adequately represent the various areas in the pupil. For a fully incoherent source, i.e., one which filled the pupil, the areas marked 0 and 0.5 must be adequately represented (remembering that these regions strictly apply only to a specific spatial frequency). Perhaps 50 source points will suffice. For partial filling of the source, as indicated by the dashed circle, a smaller region need be sampled, reducing the number of sample points correspondingly.

A diffraction calculation has difficulty modeling a true point. Instead of beginning the beam as diverging from a source point, we model the beam as it appears at the image of the mask in the first condenser optical space. The beam is, of course, in the same optical space as the source. When the beam is initiated at the image of the mask, it is apertured, given appropriate convergence to be directed toward the source plane, and tilted to the proper angle for the corresponding off-axis point. Because the beam is apertured when it is created, it represents a source region of finite size thus avoiding aliasing problems.

The beam is then propagated through the condenser system. The condenser system is modeled according to its exact surface shape and three-dimensional position. The aberrations of the condenser included in the calculation. The beam then is propagated to the mask. The mask is implemented by multiplying the beam array by the mask array which is constructed at the beginning of the program. Multiplying the two arrays is much faster than building the mask for each pass. A Schwartzchild relay design is used to reimage the mask with a 1:20 reduction ratio. The incoherent image of each point is summed until all source points have been processed.

GLAD may be used to model a source of any shape. In this example we used simple circular patches corresponding to 0, 0.5, 1. Since the source points corresponding to 0 and 0.5 are subsets of the full source needed for incoherent illumination, we trace the center point ($\sigma = 0$), an annular region of half radius ($\sigma = 0.5$), and an outer annular region which completes the incoherent plot. These annular regions are generated by setting up inner and outer loops of a rectangular scan over x- and y-directions, but allowing only those points corresponding to the current annular region to be traced.

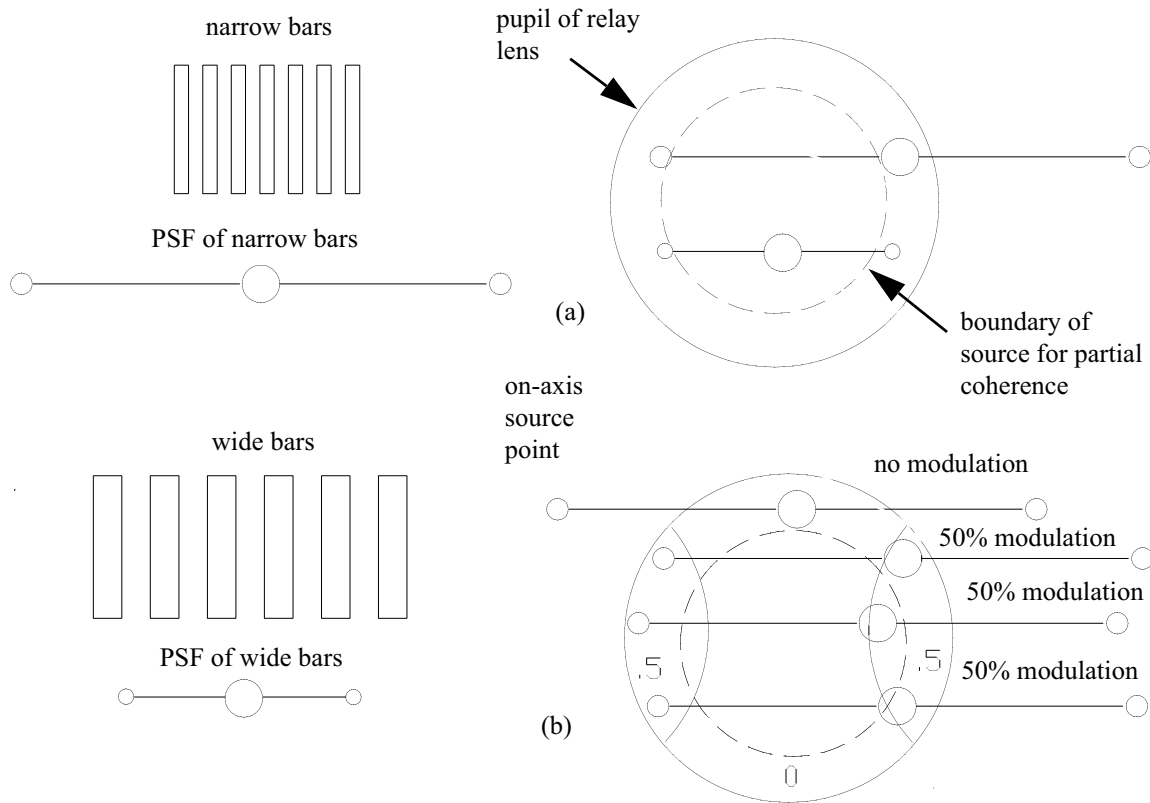


Fig. 83.2. A set of narrow bars and wide bars in the object mask creates a PSF in the pupil of the relay lens which has a center lobe and two side bands, as shown in (a). The narrow bars have a wide PSF and the wide bars have a narrow PSF. For very narrow bars, the PSF may be so wide that it is impossible to get all three lobes through the pupil. This causes zero modulation for the on-axis source point. Wider bars, with the narrower PSF may have all three lobes passing through the pupil yielding good modulation in the image of the bars. For off-axis source points the PSF is shifted, and when two lobes pass through the aperture, the image is partially modulated. Figure (b) shows the locus of points which just pass both lobes.

The source points which put the central lobe in the lens-like regions will yield approximately 50% modulation as marked in the sketch. Locating the center lobe inside or outside of the lens-like regions will yield zero modulation. Since off-axis points may be at least partially modulated, a source of finite extent has greater image modulation. A source which fills the pupil yields incoherent imaging. A source which partially fills the pupil, as shown by the dashed circle defines the region over which the central lobe will move in the pupil. For the partial source size shown, the off-axis source points at the extreme edge of the source will contribute some image modulation.

A full incoherent calculation may require up to 50 system passes. Fortunately, we generally want to work at about $\sigma = 0.5$ so that only about 10 are needed. The calculation may also be speeded up by simplifying the modeling of the condenser system. The process is relatively insensitive to the aberrations of the condenser. It may be possible to ignore the condenser aberrations or to use a polynomial aberration formulation to avoid calculating the aberrations for each point in the array. However Ex. 83a gives a full system model to illustrate all features.

Ex83a: Partial coherent imaging of a three-bar pattern

Figure 83.3. shows the simple three-bar mask. Figure 83.4 shows the appearance of the image of the source as it appears in the pupil of the relay lens. The fully coherent image is shown in Fig. 83. 5. Figure

83.6 shows the image of the source points at $\sigma = 0.5$, note that the region is “spanned” by the source points but not filled. Experiments with GLAD do not show changes in the imaging when using more source points. In fact fewer points may be acceptable. For a series of bars, more source points may be concentrated along the diameter. Figure 83.7 shows the resulting partially coherent image. The pupil of the relay lens is fully spanned as shown in Fig. 83. 8. The resulting image is completely incoherent as shown in Fig. 83. 9.

Ex83a: Partial coherent imaging of a three-bar pattern

Figure 83.3. shows the simple three-bar mask. Figure 83.4 shows the appearance of the image of the source as it appears in the pupil of the relay lens. The fully coherent image is shown in Fig. 83.5. Figure 83.6 shows the image of the source points at $\sigma = 0.5$, note that the region is “spanned” by the source points but not filled. Experiments with GLAD do not show changes in the imaging when using more source points. In fact fewer points may be acceptable. For a series of bars, more source points may be concentrated along the diameter. Figure 83.7 shows the resulting partially coherent image. The pupil of the relay lens is fully spanned as shown in Fig. 83. 8. The resulting image is completely incoherent as shown in Fig. 83. 9.

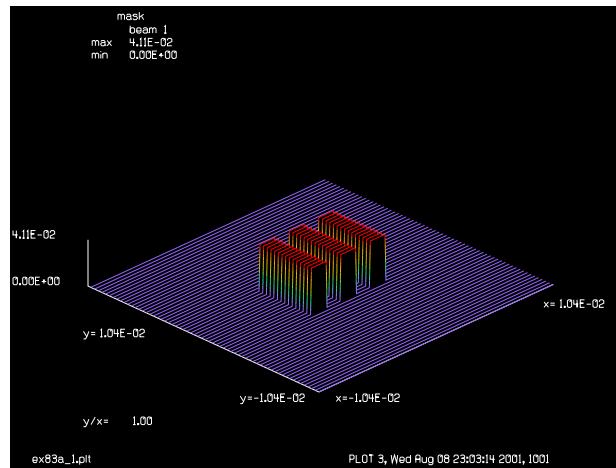


Fig. 83.3. Object mask consisting of three bars.

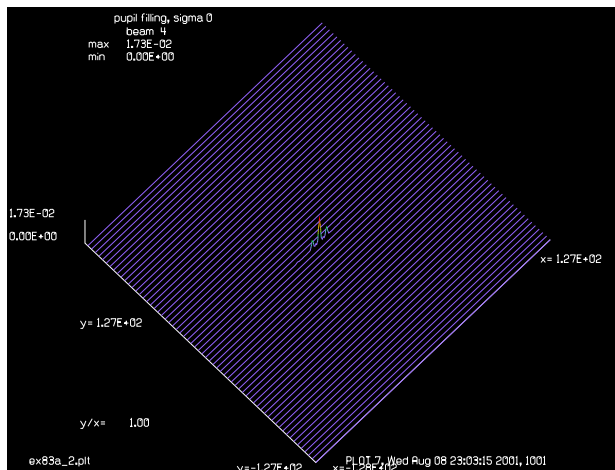


Fig. 83.4. Image of single source point in the pupil of the relay lens. This will result in full coherence, $\sigma = 0$.

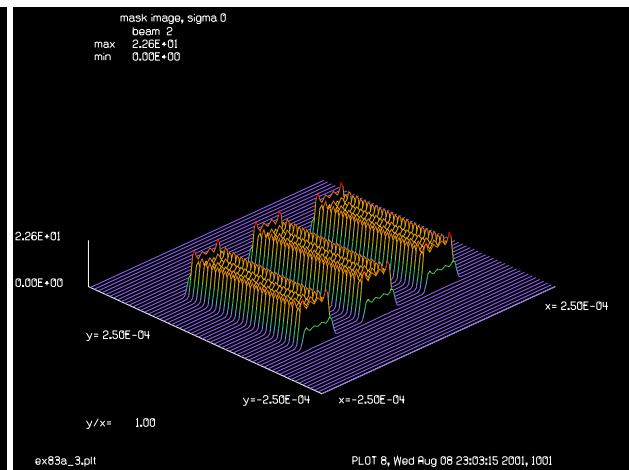


Fig. 83.5. Image of three bars at full coherence, $\sigma = 0$.

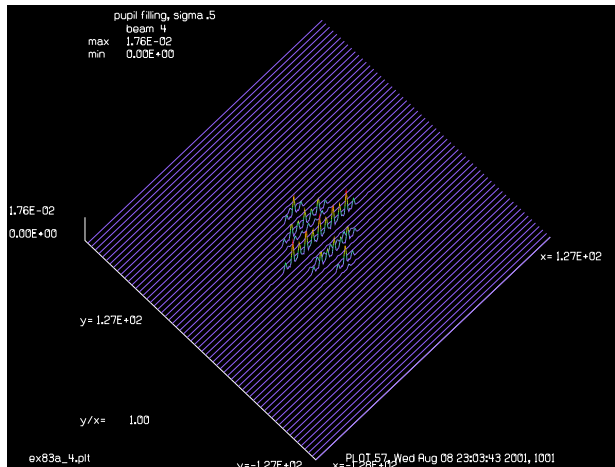


Fig. 83.6. Image of source points in the pupil of the relay lens at $\sigma = 0.5$, partial coherence.

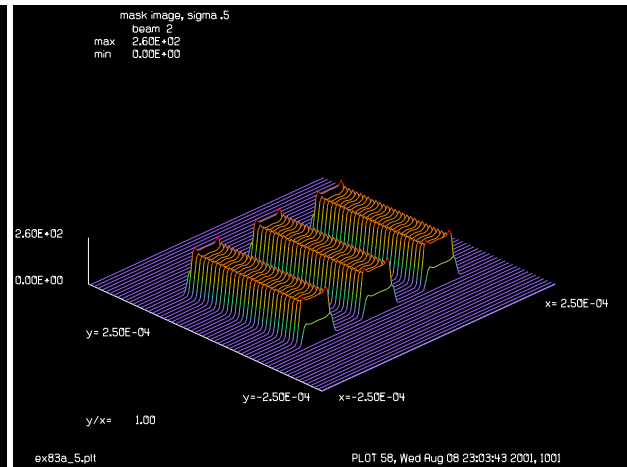


Fig. 83.7. Image of three bars at $\sigma = 0.5$, partial coherence.

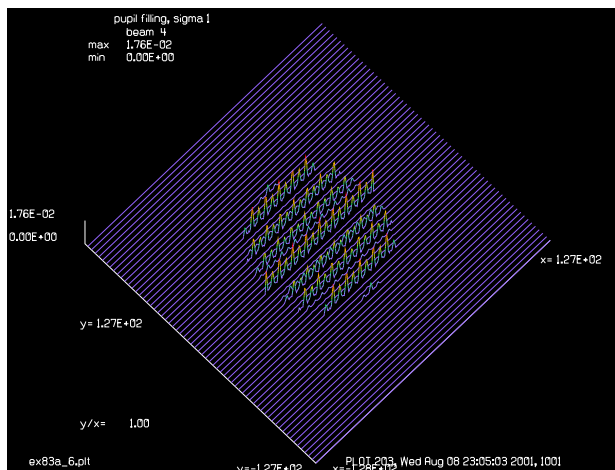


Fig. 83.8. Image of source points filling the pupil of the relay lens to give full incoherence, $\sigma = 1$.

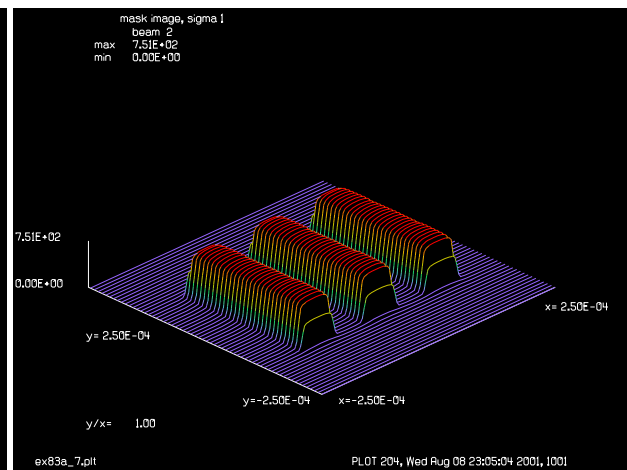


Fig. 83.9. Incoherent imaging of three bars, $\sigma = 1$.

Input: `ex83a.inp`

```
c## ex83a!191357809573566
c
c Example 83a: Partial coherence, 256 x 256
c
c A three bar pattern is imaged with 0, 0.5 and 1.0 source filling
c of the pupil.
c
set/alias_stop/off
macro/def system/o
  sys_pass = sys_pass + 1
  units/s 1 1.6534e-5
  clear 1 1
  global/def 1 0 0 -.7706      # conjugate 25.7706, m=4.9352
```

Jump to: [Commands](#), [Theory](#)

```

zreff/se 1 -.7706
abr/foc 1 1.267 rn=1.6534e-3
clap/c/c 1 1.2e-3
geodata/set 1 waistx=1e-4 waisty=1e-4
abr/tilt 1 tilt az=theta rn=1.
vertex/locate/abs 0 0 0
vertex/rotate/set -45
mirror/global/flat
vertex/locate/abs 0 1.8849 -11.3095
vertex/rotate/set 9.462322
mirror/global/conic r=21.4286 cc=-.755102
prop 127.182
mult/beam 1 3
plot/l 1 ns=64
if sys_pass = 1 then
    title mask
    plot/watch ex83a_1.plt
    plot/l 1 ns=64 h=.2
    plot/watch plot1.plt
endif
prop 22.818
clap/c/n 1 .089
add/inc/con 4 1
plot/l 4 ns=64 h=.1 theta=44
vertex/locate/abs 0 -(25+dec) 154.8271
vertex/rotate/set 0 0 0
mirror/global/conic 1.8313 cc=0
title after primary
vertex/locate/abs 0 -(25+dec) 151.5986
vertex/rotate/set 0 0 0
mirror/global/conic r=5.0324 cc=0
title after secondary
prop/chief/rel zimage
variab/set Units1 1 units
units/set 2 Units1
add/inc/con 2 1
plot/x/i 2
plot/l 2 ns=64 xrad=2.5e-4 h=.2
macro/end
macro/def coherent/o
c
c loop over source values
c
    reltilt = (tiltx*tiltx)+(tilty*tilty) list
    reltilt = sqrt(reltilt)
    pass = 1
    if reltilt > max then
        pass = 0
    endif
    if reltilt < min then
        pass = 0
    endif
    if pass = 1 then
        tilt = (4.9352*reltilt*(.089/22.818))/lambda

```

Jump to: [Commands](#), [Theory](#)

```

    theta = (57.2957795*atan2(tilty,tiltx))+90
macro system
  if tiltx != 0 then
    tiltx = -tiltx
    tilt = (4.9352*reltilt*(.089/22.818))/lambda
    theta = (57.2957795*atan2(tilty,tiltx))+90
    macro system
      tiltx = -tiltx
    endif
  endif
  tiltx = tiltx+.25 list
macro/end
macro/def outer/o
  tiltx = 0.
  mac coherent/ntimes
  if tilty != 0 then
    tilty = -tilty
    tiltx = 0.
    mac coherent/ntimes
    tilty = -tilty
  endif
  tilty = tilty+.25
macro/end
mem/set/b 2
dec = 0. # do not decenter Schwartzchild
zimage = 6.543-.06623
lambda = 14e-7
nbeam 4
array/s 1 256
array/s 2 256 data
array/s 3 256 data
array/s 4 256 data
clear 2 1
clear 4 0
variab/set units3 3 units
clear 3 0
clap/rec/c 2 units3*8 units3*30 units3*32
copy/con/c 2 3
clear 2 1
clap/rec/c 2 units3*8 units3*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units3*8 units3*30 -units3*32
add/coh/con 3 2
normalize 3
plot/screen/pause 3
plot/l 3 ns=64 h=.2
clear 2 0
wavelength/set 0 .014
beams/off 2
beams/off 3
beams/off 4
set/window/rel 20 80 20 80
set/density 64

```

Jump to: [Commands](#), [Theory](#)

```

clear 2 0
max = .1
min = 0.
tilty = 0
ntimes = 1
mac outer/ntimes
plot/watch ex83a_2.plt
title pupil filling, sigma 0
plot/1 4 ns=64 h=.1 theta=44
plot/watch ex83a_3.plt
title mask image, sigma 0
plot/1 2 ns=64 xrad=2.5e-4 h=.2
plot/watch plot1.plt
max = .5
min = .1
tilty = 0
ntimes = 3
mac outer/ntimes
plot/watch ex83a_4.plt
title pupil filling, sigma .5
plot/1 4 ns=64 h=.1 theta=44
plot/watch ex83a_5.plt
title mask image, sigma .5
plot/1 2 ns=64 xrad=2.5e-4 h=.2
plot/watch plot1.plt
max = 1.
min = .51
tilty = 0
ntimes = 5
mac outer/ntimes
plot/watch ex83a_6.plt
title pupil filling, sigma 1
plot/1 4 ns=64 h=.1 theta=44
plot/watch ex83a_7.plt
title mask image, sigma 1
plot/1 2 ns=64 xrad=2.5e-4 h=.2
plot/watch plot1.plt
end

```

Ex83b: Two sets of seven bars, above and below the coherent resolution limit

Example 83b models imaging of two sets of seven bars, as shown in Fig. 83.10. The wide bars have $2\ \mu$ feature size (just within the coherent MTF) and the narrow bars have $1.5\ \mu$ feature size (just outside the coherent MTF). In order to provide adequate sampling the 512 arrays were used. Figure 83.11 shows the image of a single on-axis source point in the pupil of the relay optic, just before the aperture. Figure 83.12 shows the effect of the relay aperture stop to clip the outer lobes which were generated by the narrow bars. Figure 83.13 shows the resulting partially coherent imaging. The narrow bars are not resolved, while the wide bars are well resolved. Figure 83.14 shows $\sigma = 0.5$, partial coherent imaging with the narrow bars partially resolved and Fig. 83.15 shows how the source at $\sigma = 0.5$ fills the pupil. Figure 83.16 shows the imaging and pupil filling for $\sigma = 1$.

Jump to: [Commands](#), [Theory](#)

The effect of 5 waves of astigmatism in the condenser mirror was modeled. Figures 83.18 and 83.19 show coherent imaging and pupil filling respectively. Note that the source image in the relay pupil is wider in the horizontal direction because of the astigmatism in Fig. 83.19. There is no observable effect on the image observable in Fig. 83.18. Similar figures are shown for $\sigma = 0.5$ imaging with astigmatism in Figures 83.20 and 83.21. Again, there is negligible effect of the condenser astigmatism on imaging.

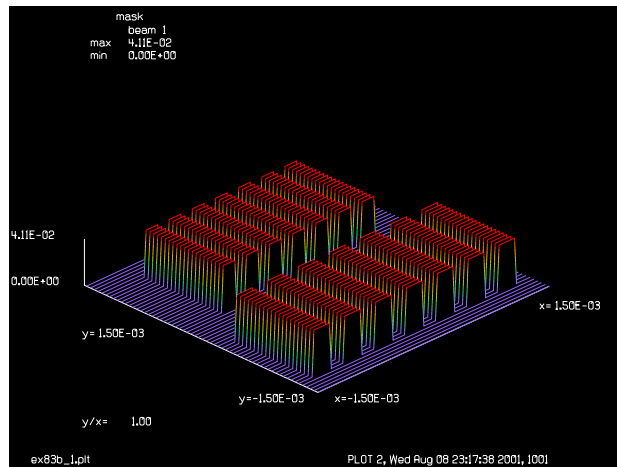


Fig. 83.10. Object mask consisting of seven bars of 1.5μ and 2μ .

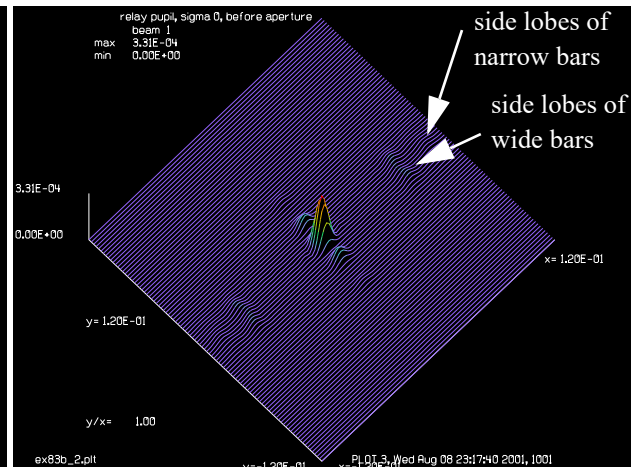


Fig. 83.11. Image of single source point in pupil of relay lens, before aperture.

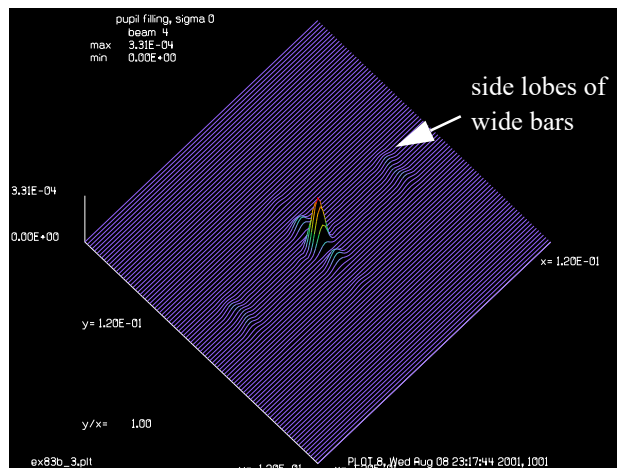


Fig. 83.12. Image of single source point in pupil of relay lens, after aperture. Note the lobes from the narrow pattern are eliminated.

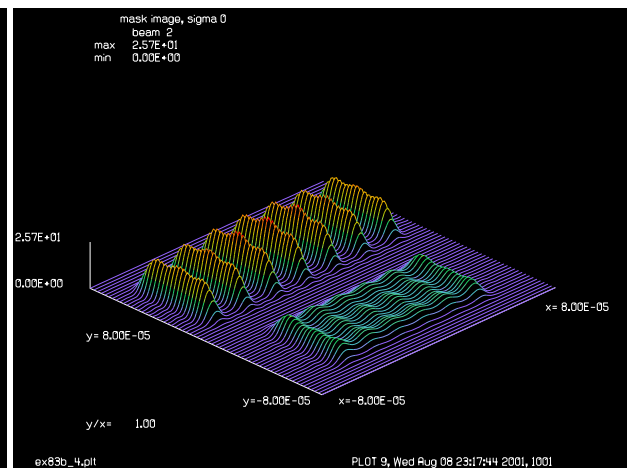


Fig. 83.13. Image of seven bar patterns for a single on-axis source point, full coherence. Wide bars are resolved. Narrow lobes are not resolved because the side lobes did not pass.

Input: `ex83b.inp`

```
c## ex83b!564896180339624
c
c Partial coherence, 512 x 512
c
c Two seven bar patterns are imaged with 0, 0.5 and 1.0 source filling
```

Jump to: [Commands](#), [Theory](#)

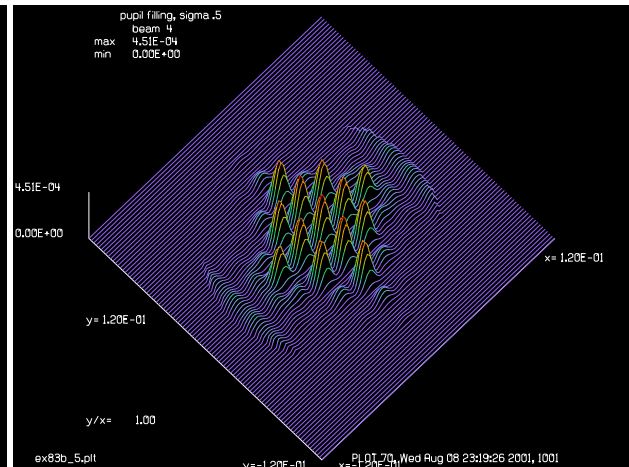
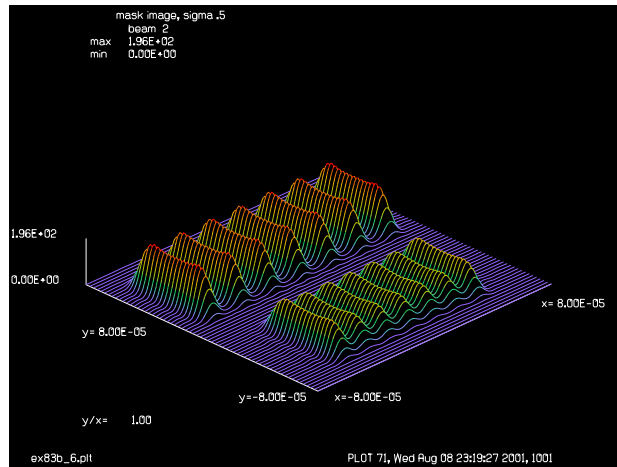


Fig. 83.14. Image of mask at $\sigma = 0.5$, partial coherence. Fig. 83.15. Pupil of relay showing partial filling, $\sigma = 0.5$.

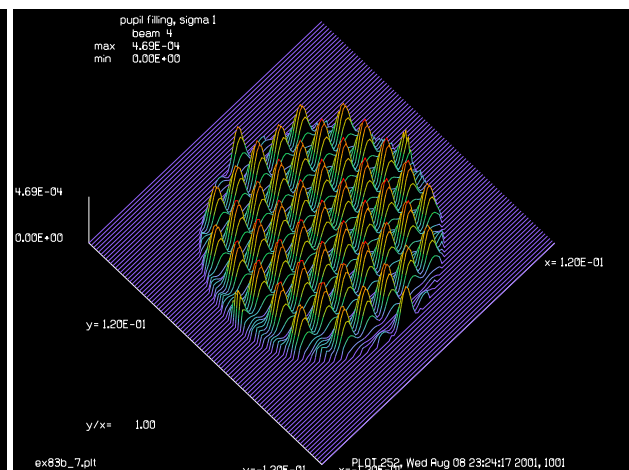
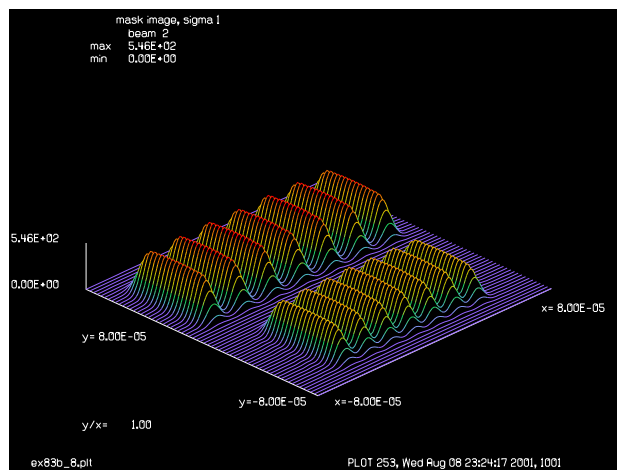


Fig. 83.16. Image of mask at $\sigma = 1$, full incoherence.

Fig. 83.17. Pupil of relay showing full filling, $\sigma = 1.0$, full incoherence.

```

c of the pupil.
c
echo/on
variab/dec/int pass sys_pass ntimes
macro/def system/o
  sys_pass = sys_pass + 1
  units/s 1 5.0657e-6
  clear 1 1
  global/def 1 0 0 -.7706      # conjugate 25.7706, m=4.9352
  zreff/set 1 -.7706
  abr/foc 1 1.267 rn=1.6534e-3
  clap/c/c 1 1.2e-3
c# abr/ast 1 5 az=90 rn=1.2e-3 # include this line to add aberration
geodata/set 1 waistx=1e-4 waisty=1e-4
abr/tilt 1 tilt az=theta rn=1.
vertex/locate/abs 0 0 0
vertex/rotate/set -45

```

Jump to: [Commands](#), [Theory](#)

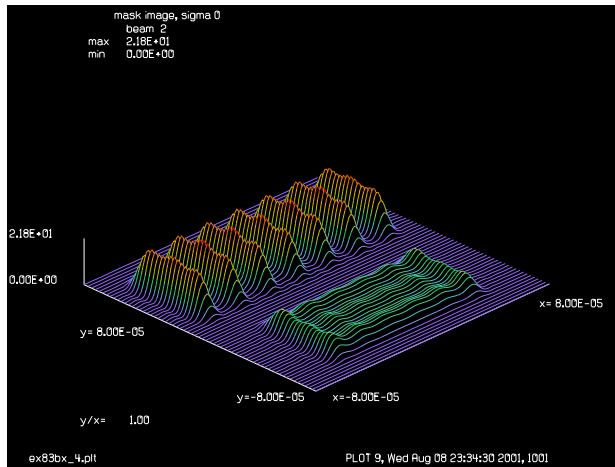


Fig. 83.18. Image of mask at $\sigma = 0.0$, full coherence, with 5 waves of astigmatism. Essentially identical with Fig. 83.14, which had no aberration.

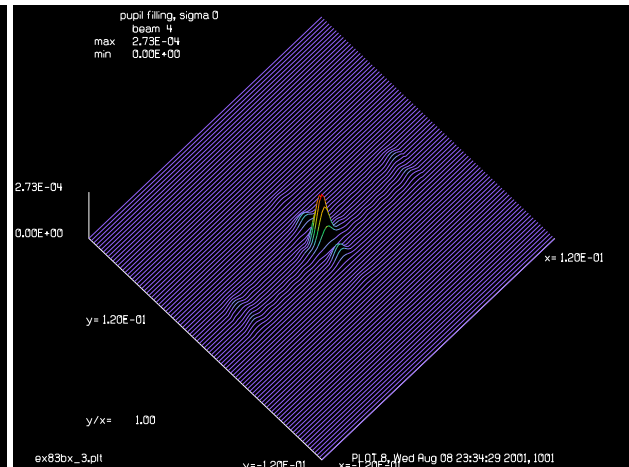


Fig. 83.19. Pupil of relay showing effects of 5 waves of astigmatism, $\sigma = 0.0$. Compare with Fig. 83.15.

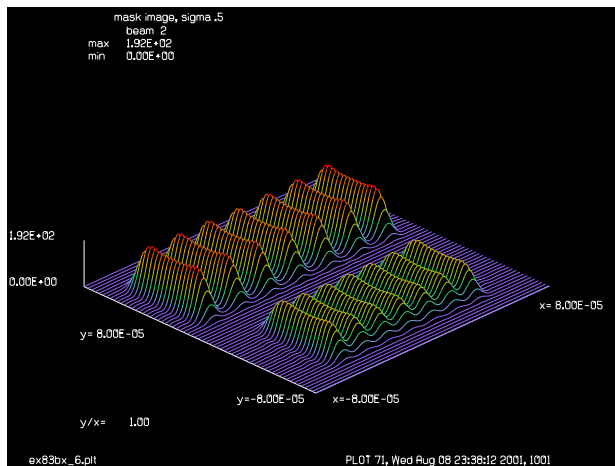


Fig. 83.20. Image of mask at $\sigma = 0.5$, partial coherence, with 5 waves of astigmatism. Essentially identical to Fig. 83.16, with no aberration.

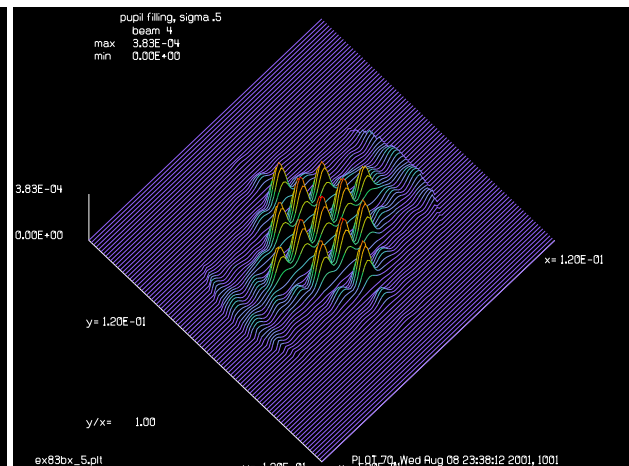


Fig. 83.21. Pupil of relay showing effects of 5 waves of astigmatism, $\sigma = 0.5$. Compare with Fig. 83.17.

```
mirror/global/flat
vertex/locate/abs 0 1.8849 -11.3095
vertex/rotate/set 9.462322
mirror/global/conic r=21.4286 cc=-.755102
prop 127.182
mult/beam 1 3
plot/l 1 ns=64
c pause 4
if sys_pass = 1 then
  title mask
  plot/watch ex83b_1.plt
  plot/l 1 ns=64 xrad=1.5e-3 h=.2
  plot/watch plot1.plt
endif
```

Jump to: [Commands](#), [Theory](#)

```

prop 22.818
if sys_pass = 1 then
    variab/set units 1 units
    units/set 4 units
endif
if sys_pass = 1 then
    title relay pupil, sigma 0, before aperture
    plot/watch ex83b_2.plt
    plot/1 1 ns=64 xrad=.12 h=.2 theta=44
    plot/watch plot1.plt
endif
clap/c/n 1 .089
add/inc/con 4 1
plot/1 4 ns=64 xrad=.12 h=.2 theta=44
c    pause 5
vertex/locate/abs 0 -25-dec 154.8271
vertex/rotate/set 0 0 0
mirror/global/conic 1.8313 cc=0
title after primary
vertex/locate/abs 0 -25-dec 151.5986
vertex/rotate/set 0 0 0
mirror/global/conic r=5.0324 cc=0
title after secondary
prop/chief/rel zimage
variab/set units 1 units
units/set 2 units
add/inc/con 2 1
title wide bars
plot/x/i 2 left=-8e-5 right=8e-5 slice=3.75e-5
title narrow bars
plot/x/i 2 left=-8e-5 right=8e-5 slice=-3.75e-5
plot/1 2 ns=64 xrad=8e-5 h=.2
c    pause 6
macro/end
macro/def coherent/o
c
c    loop over source values
c
    reltilt = (tiltx*tiltx)+(tilty*tilty) list
    reltilt = sqrt(reltilt)
    pass = 1
    if reltilt > max then
        pass = 0
    endif
    if reltilt < min then
        pass = 0
    endif
    if pass = 1 then
        tilt = (4.9352*reltilt*(.089/22.818))/lambda
        theta = (57.2957795*atan2(tilty,tiltx))+90
        macro system
        if tiltx != 0 then
            tiltx = -tiltx
            tilt = (4.9352*reltilt*(.089/22.818))/lambda

```

Jump to: [Commands](#), [Theory](#)


```

        theta = (57.2957795*atan2(tilty,tiltx))+90
        macro system
            tiltx = -tiltx
        endif
    endif
    tiltx = tiltx+.25 list
c    pause 4
macro/end
macro/def outer/o
    tiltx = 0.
    mac coherent/ntimes
    if tilty != 0 then
        tilty = -tilty
        tiltx = 0.
        mac coherent/ntimes
        tilty = -tilty
    endif
    tilty = tilty+.25
macro/end
mem/set 20
dec = 0.                                # do not decenter Schwartzchild
zimage = 6.543-.06623
lambda = 14e-7
nbeam 4
array/s 1 512
array/s 2 512 data
array/s 3 512 data
array/s 4 512 data
clear 2 1
clear 4 0
variab/set units 3 units
clear 3 0
clap/rec/c 2 units*3 units*20 units*36 units*30
copy/con/c 2 3
clear 2 1
clap/rec/c 2 units*3 units*20 units*24 units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*3 units*20 units*12 units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*3 units*20 0 units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*3 units*20 -units*12 units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*3 units*20 -units*24 units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*3 units*20 -units*36 units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*4 units*20 units*48 -units*30

```

Jump to: [Commands](#), [Theory](#)

```
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*4 units*20 units*32 -units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*4 units*20 units*16 -units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*4 units*20 0 -units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*4 units*20 -units*16 -units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*4 units*20 -units*32 -units*30
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*4 units*20 -units*48 -units*30
add/coh/con 3 2
normalize 3
clear 2 0
wavelength/set 0 .014
beams/off 2
beams/off 3
beams/off 4
set/window/rel 20 80 20 80
set/density 64
clear 2 0
max = .1
min = 0.
tilty = 0
ntimes = 1
mac outer/ntimes
plot/watch ex83b_3.plt
title pupil filling, sigma 0
plot/1 4 ns=64 xrad=.12 h=.2 theta=44
plot/watch ex83b_4.plt
title mask image, sigma 0
plot/1 2 ns=64 xrad=8e-5 h=.2
plot/watch plot1.plt
max = .5
min = .1
tilty = 0
ntimes = 3
mac outer/ntimes
plot/watch ex83b_5.plt
title pupil filling, sigma .5
plot/1 4 ns=64 xrad=.12 h=.2 theta=44
plot/watch ex83b_6.plt
title mask image, sigma .5
plot/1 2 ns=64 xrad=8e-5 h=.2
plot/watch plot1.plt
max = 1.
min = .51
```

Jump to: [Commands](#), [Theory](#)

```

tilty = 0
ntimes = 5
mac outer/ntimes
plot/watch ex83b_7.plt
title pupil filling, sigma 1
plot/1 4 ns=64 xrad=.12 h=.2 theta=44
plot/watch ex83b_8.plt
title mask image, sigma 1
plot/1 2 ns=64 xrad=8e-5 h=.2
plot/watch plot1.plt
end

```

Ex83c: Thirteen-bar pattern to compare with theory

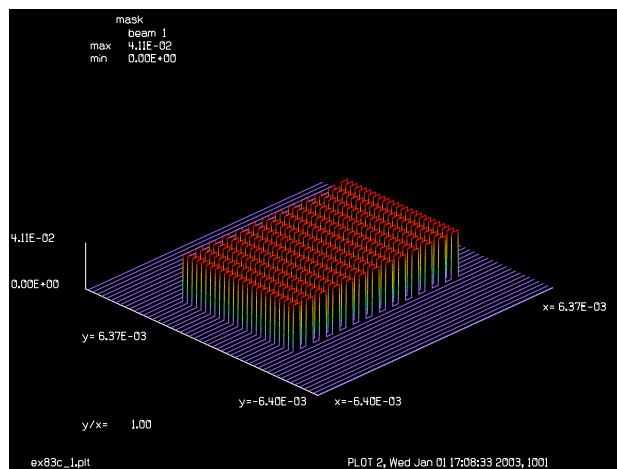


Fig. 83.22. Thirteen bar target set to a period of 0.9 of the coherent diffraction limited spatial frequency.

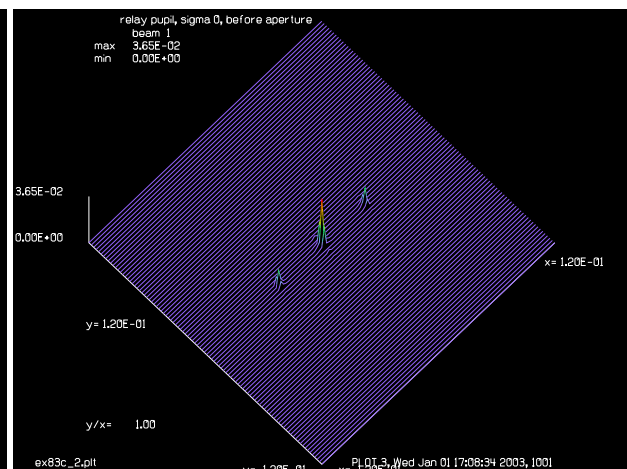


Fig. 83.23. Irradiance at relay pupil due to a single source point and the thirteen bar pattern with spatial frequency equal to 0.5 of the coherent diffraction limit.

Input: ex83c.inp

```

c## ex83c!373463083250224
c
c Partial coherence, 512 x 512
c
c A thirteen bar pattern, with period approximately equal to the
c coherent MTF limit. Coherence associated with .0, .7 and 1.0 source
c filling of the pupil. This example combines three curves into a
c udata plot. Results may be directly compared with H. H. Hopkins
c calculations according to the GLAD Theory Manual, Chap 18.
c
variab/dec/int isum istep switch sys_pass pass
fmax = 16.3*1.3
units = 1.26136e-6
lambda = 14e-7
right = 28.71*units

```

Jump to: [Commands](#), [Theory](#)

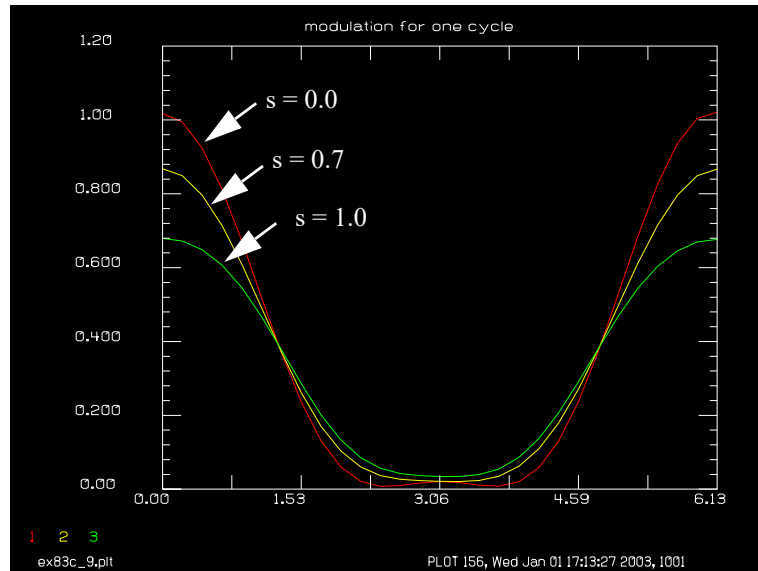


Fig. 83.24. Modulation at $\sigma = 0.0, 0.7, 1.0$ for 0.5 of the coherent diffraction limited frequency ($\omega = 0.5$), for a thirteen bar pattern. This compares very well with the $\omega = 0.5$ bar pattern of infinite length calculated by Hopkins [Ref. 1]. See Chap. 18. GLAD Theory Manual.

```

echo/on
set/grid 8 0 5 5
macro/def system/o
  sys_pass = sys_pass + 1
  units/s 1 5.0657e-6
  clear 1 1
  global/def 1 0 0 -.7706      # conjugate 25.7706, m=4.9352
  zreff/se 1 -.7706
  abr/foc 1 1.267 rn=1.6534e-3
  clap/s/n 1 1.0e-3
  geodata/set/waist 1 waistx=1e-4 waisty=1e-4
  abr/tilt 1 tilt az=theta rn=1.
  vertex/locate/abs 0 0 0
  vertex/rotate/set -45
  mirror/global/flat
  vertex/locate/abs 0 1.8849 -11.3095
  vertex/rotate/set 9.462322
  mirror/global/conic r=21.4286 cc=-.755102
  prop 127.182
  mult/beam 1 3
  plot/l 1 ns=64
c  pause 4
  if sys_pass = 1 then
    title mask
    plot/watch ex83c_1.plt
    plot/l 1 ns=64 xrad=8.e-3 h=.2
    plot/watch plot1.plt
  endif
  prop 22.818
  if sys_pass = 1 then
    variab/set units 1 units

```

Jump to: [Commands](#), [Theory](#)

```

    units/set 4 units
endif
if sys_pass = 1 then
    title relay pupil, sigma 0, before aperture
    plot/watch ex83c_2.plt
    plot/1 1 ns=64 xrad=.12 h=.2 theta=44
    plot/watch plot1.plt
endif
clap/c/n 1 .089
add/inc/con 4 1
plot/1 4 ns=64 xrad=.12 h=.2 theta=44
c  pause 5
vertex/locate/abs 0 -25-dec 154.8271
vertex/rotate/set 0 0 0
mirror/global/conic 1.8313 cc=0
title after primary
vertex/locate/abs 0 -25-dec 151.5986
vertex/rotate/set 0 0 0
mirror/global/conic r=5.0324 cc=0
title after secondary
prop/chief/rel zimage
variab/set units 1 units
units/set 2 units
if sys_pass = 1 then
    variab/set efirst 1 energy list
endif
variab/set enow 1 energy list
isum = isum+1
rsum = enow/efirst list
rsum = isum/rsum
mult 2 [1.-1./rsum]
mult 1 [1./rsum]
add/inc/con 2 1
title bars
plot/x/i 2 left=0.e-5 right=right fmax=fmax fmin=0.
macro/end
macro/def coherent/o
c
c  loop over source values
c
    reltilt = (tiltx*tiltx)+(tilty*tilty) list
    reltilt = sqrt(reltilt)
    pass = 1
    if reltilt > max then
        pass = 0
    endif
    if reltilt < min then
        pass = 0
    endif
    if pass = 1 then
        tilt = (4.9352*reltilt*(.089/22.818))/lambda
        theta = (57.2957795*atan2(tilty,tiltx))+90
        macro system
        if tiltx != 0 then

```

```

        tiltx = -tiltx
        tilt = (4.9352*reltilt*(.089/22.818))/lambda
        theta = (57.2957795*atan2(tilty,tiltx))+90
        macro system
            tiltx = -tiltx
        endif
    endif
    tiltx = tiltx+.25 list
macro/end
macro/def outer/o
    tiltx = 0.
    mac coherent/ntimes
    if tilty != 0 then
        tilty = -tilty
        tiltx = 0.
        mac coherent/ntimes
        tilty = -tilty
    endif
    tilty = tilty+.25
macro/end
macro/def scan/o
c macro to compute intensity at points along x-direction
point/list/ij 2 [257+istep+29] 257
variab/set point point/i
udata/set istep+1 [2.*pi*istep/28.72] @y=point
istep = istep+1
macro/end
mem/set 20
dec = 0.                                # do not decenter Schwartzchild
zimage = 6.543-.06623
nbeam 4
array/s 1 512
array/s 2 512 data
array/s 3 512 data
array/s 4 512 data
clear 2 1
clear 4 0
variab/set units 3 units
c
c omega = 1, period = 2 ( lambda * Fn )
c omega = .5, period = 4 ( lambda * Fn ) = 4. * .014E-4 * 22.818 / .178
c period = 7.179E-4
c units = 2.5E-5
c period = 28.71 units
c rectangle half-width = 7.179 units
c
units = units*1.026
clear 3 0
clap/rec/c 2 units*7 units*120 units*28*6
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 units*28*5
add/coh/con 3 2
clear 2 1

```

Jump to: [Commands](#), [Theory](#)

```

clap/rec/c 2 units*7 units*120 units*28*4
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 units*28*3
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 units*28*2
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 units*28
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 0
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 -units*28
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 -units*28*2
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 -units*28*3
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 -units*28*4
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 -units*28*5
add/coh/con 3 2
clear 2 1
clap/rec/c 2 units*7 units*120 -units*28*6
add/coh/con 3 2
normalize 3
clear 2 0
wavelength/set 0 .014
beams/off 2
beams/off 3
beams/off 4
set/window/rel 20 80 20 80
set/density 64
clear 2 0
max = .1
min = 0.
tilty = 0
ntimes = 1
mac outer/ntimes
plot/watch ex83c_3.plt
title pupil filling, sigma .0
plot/1 4 ns=64 xrad=.12 h=.2 theta=44
plot/watch ex83c_4.plt
title mask image, sigma .0
plot/x/i 2 left=right right=[2*right] fmax=fmax fmin=0.
plot/watch plot1.plt
istep = 0

```

Jump to: [Commands](#), [Theory](#)

```
alias y y01
macro scan/29
udata/list
plot/watch plot1.plt
max = .7
min = .1
tilty = 0
ntimes = 3
mac outer/ntimes
plot/watch ex83c_5.plt
title pupil filling, sigma .7
plot/1 4 ns=64 xrad=.12 h=.2 theta=44
plot/watch ex83c_6.plt
title mask image, sigma .7
plot/x/i 2 left=right right=[2*right] fmax=fmax fmin=0.
istep = 0
alias y y02
macro scan/29
udata/list
plot/watch plot1.plt
max = 1.
min = .7001
tilty = 0
ntimes = 5
mac outer/ntimes
plot/watch ex83c_7.plt
title pupil filling, sigma 1
plot/1 4 ns=64 xrad=.12 h=.2 theta=44
plot/watch ex83c_8.plt
title mask image, sigma 1
plot/x/i 2 left=right right=[2*right] fmax=fmax fmin=0.
plot/watch plot1.plt
istep = 0
alias y y03
macro scan/29
udata/list
udata/calc 1 [1./fmax]
udata/calc 2 [1./fmax]
udata/calc 3 [1./fmax]
set/grid 8 0 6 5
title modulation for one cycle
plot/watch ex83c_9.plt
plot/udata 1 3 max=1.2 min=0.
```


Ex84: Effects of laser heating of a window, convolution method

This example illustrates the case of a laser beam passing through a window and experiencing the aberrations induced in the window due to absorption of the laser beam. The instantaneous heat deposition is proportional to the irradiance distribution on the window. The distribution of heat in the windows experiences conduction, which causes the heat to spread laterally. We assume the conduction to the air is negligible and that the only heat loss is due to the edges of the window. We assume the mount, which is attached to the edge of the window, is infinitely conducting.

The transverse flow of heat in the window may be modeled by using the influence function of heat conduction. Given the heat deposition function at time $H(x, y, t)$. The heat deposition at time $P + \Delta t$,

$$H(x, y, t + \Delta t) = H(x, y, t) ** s(x, y, \Delta t) \quad (84.1)$$

$$s(x, y, \Delta t) = \exp \left[-\frac{\rho c}{\kappa z} (x^2 + y^2) \Delta t \right] \quad (84.2)$$

where $s(x, y, \Delta t)$ is the influence function of heat conduction, c is the specific heat, ρ is the density, κ is the heat conductivity, and Δt is the time increment. The width of the heat conduction influence function depends on the value of the increment in time per step. In this example, we simple set $\Delta t = 1$.

To study the window heating effects, a circular laser distribution with central obscuration as shown in Fig. 84.1 and Fig. 84.2. Initially, the heat deposition function, shown at step 40 in Fig. 84.3 and Fig. 84.4, roughly following the laser distribution. The far-field distribution at time step 40, shown in Fig. 84.5, is essentially diffraction-limited. After 300 steps, the heat distribution has achieved nearly a steady-state condition with the heat deposition per step balanced by the heat loss at the edges. The heat distribution in steady-state is very different (and much smoother) than the laser distribution, as shown in Figs. 84.6 and 84.7. The peak value of the heat distribution, as a function of time, is shown in Fig. 84.9.

The heat distribution alters the index of refraction. This is modeled by using the self-focusing routine `sfocus`. This routine generates phase errors in Beam 1 due to the irradiance in Beam 2. The far-field distribution, shown in Fig. 8, is considerably degraded with respect to distribution at step 40, Ex5. The Strehl ratio is plotted as a function of time in Ex10. Note that the Strehl ratio assumes a steady-state value as the heat distribution becomes stable.

Input: ex84.inp

```
c## ex84!210884743827859
c
c Example 84: Example of heating of a refractive plate including the effects
c of edge cooling of the plate
c
c A gaussian beam is incident on a glass plate. Beam 1.
c
c The heat flows transversely in the plate due to conduction.
c This is modeled by convolution with the thermal impulse response function.
c The heat deposited in the plate due to bulk absorption is saved in
c Beam 2.
c
```

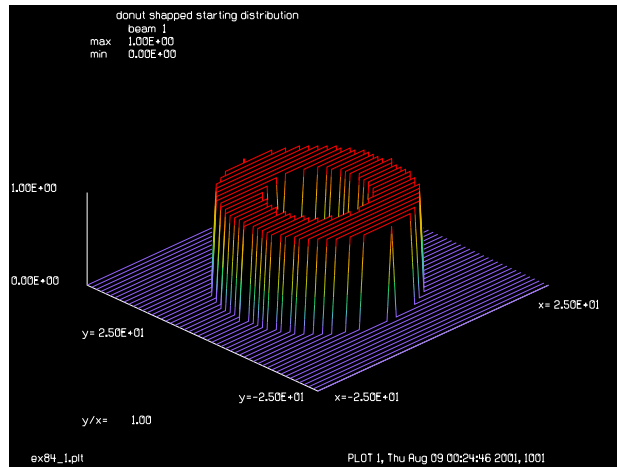


Fig. 84.1. Starting distribution for laser.

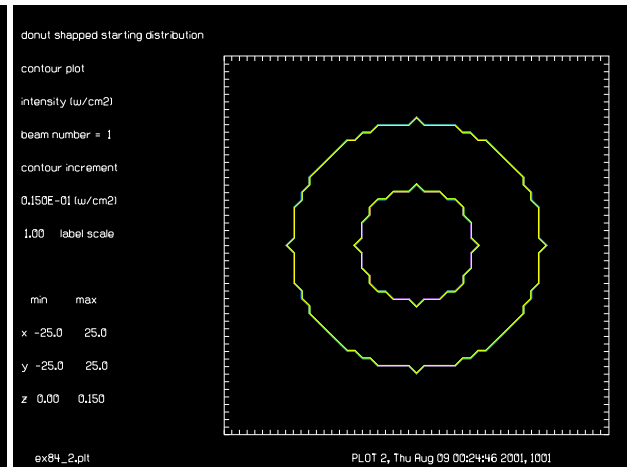


Fig. 84.2. Contour plot of initial laser distribution.

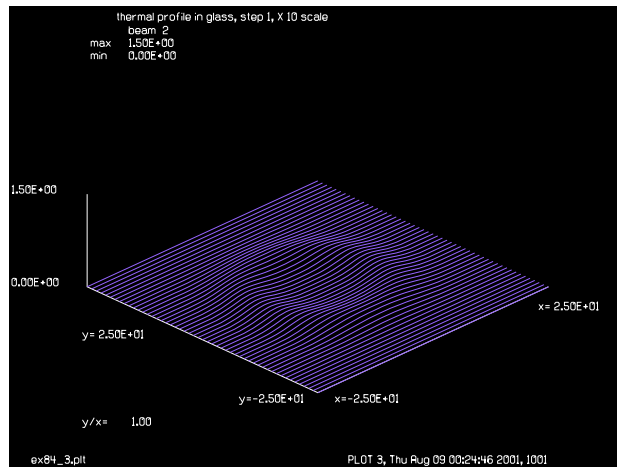


Fig. 84.3. Heat deposition after 40 steps, showing a similarity to the laser beam.

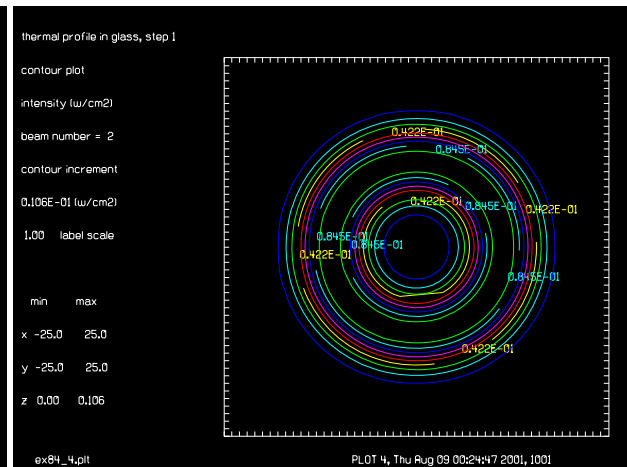


Fig. 84.4. Contour plot of heat deposition after 40 steps.

c The edges of the plate are assumed to be a heat sink. This is modeled
 c by truncating beam 2 with the aperture function

c
 c Aberrations due to the heating of the plate are calculated by
 c converting the heat distribution into a phase distribution
 c and multiplying the original beam by the phase screen, thus
 c adding aberrations. Beam 3.

```
c
mem/set 20
nbeam 4
array/s 1 128          # select array size for optical beam
array/s 2 128 data     # specify array for heat distribution
array/s 3 128 data     # specify array for aberrations
array/s 4 128 data     # influence function for heat impulse function
lambda = 1.06e-4
wavelength/set 0 lambda*1e4      # specify 1.06 microns
set/density 64 64
```

Jump to: [Commands](#), [Theory](#)

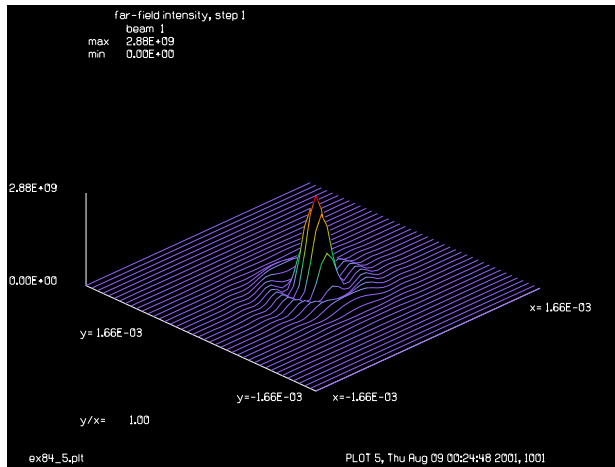


Fig. 84.5. Far-field pattern, showing near diffraction-limited performance.

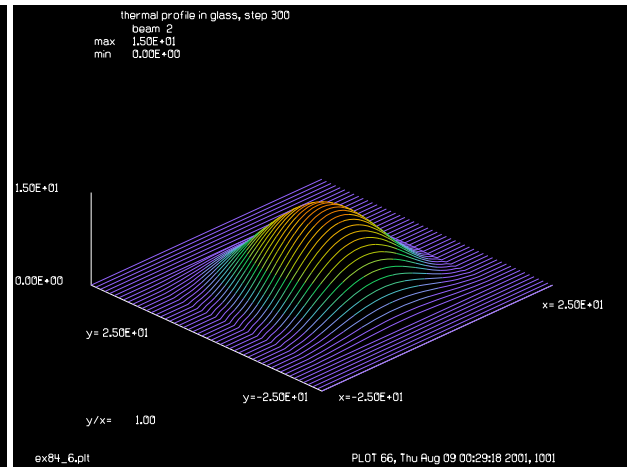


Fig. 84.6. Heat deposition after 300 steps -- near steady-state condition.

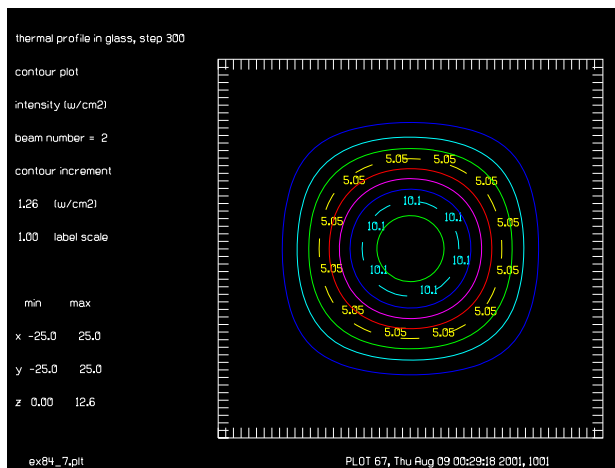


Fig. 84.7. Contours of heat deposition, 300 steps, showing the influence of the square edge heat sink.

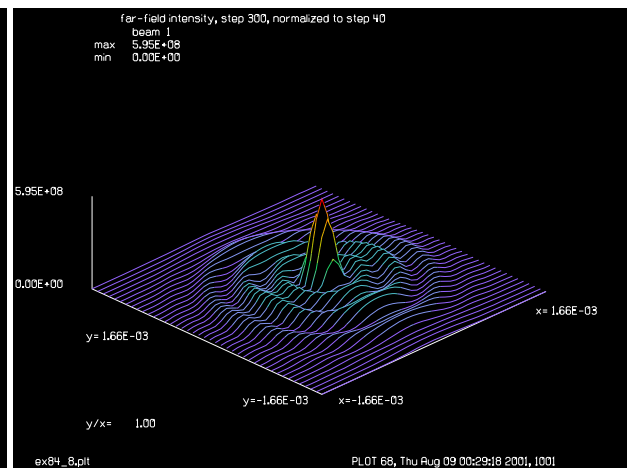


Fig. 84.8. Far-field at 300 steps plotted to same scale as Fig. 84.5, showing effects of aberration.

```
set/window/abs -25 25 -25 25
variab/dec/int pass count data
c
c Define variables
c
absorb = .1                # absorption per time step
trans = [1 - absorb]      # transmission function
clear 2 0                  # initialize Beam 2 to zero (zero stored energy)
spread = 15                # width of thermal influence function is 15
alpha = 2.*pi*1e-5/lambda # coefficient to convert from heat to phase
gaus/c/c 4 1 5            # influence function
energy/norm 4 1
macro/def thermal/o
    pass = pass+1
    count = count + 1
    units/s 1 1            # initialize beam 1 every time
```

Jump to: [Commands](#), [Theory](#)

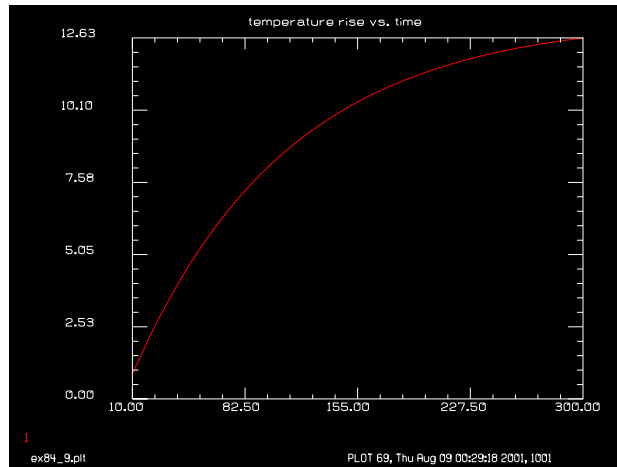


Fig. 84.9. Peak value of heat deposition per pass. At 300 passes the system is near steady-state.

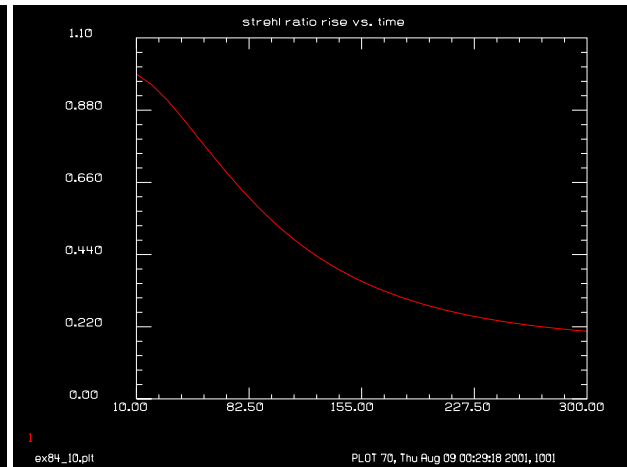


Fig. 84.10. Strehl ratio as a function of pass number. The Strehl ratio is near steady-state.

```

array/s 1 128
clear 1 1
clap/c/c 1 16
obs 1 8
if pass = 1 then
    title donut shapped starting distribution
    plot/watch ex84_1.plt
    plot/l 1 xrad=25 ns=64
    plot/watch ex84_2.plt
    plot/con 1 peak=15 ilab=0
    plot/tty
endif
copy/con 1 3          # store a temporary copy of the beam
mult 1 trans          # calculate transmission
mult 3 absorb         # calculate the absorbed energy
add/inc 2 3           # add heat to Beam 2
variab/set energy 2 energy list
convolve/beam 2 4     # convolve with thermal influence function
energy/norm 2 energy  # renormalize energy
clap/s/c 2 18         # size of window, to include heat absorbing walls
c plot/watch plot1.plt
c title temperature distribution pass @pass
c plot/l 2 max=15 xrad=25
clear 3 1.            # initialize beam 3 to 1's
int2phase/two 1 2 -alpha
if count = 10 then
c
c maximum heat deposition array every 10th pass to screen
c
    data = data + 1
    variab/set peak 2 peak
    variab/set str 1 str list
    udata/set data pass peak str
    title temperature rise vs. time
    plot/watch plot2.plt

```

```

    plot/udata 1 1 min=0
    title strehl ratio rise vs. time
    plot/watch plot3.plt
    plot/udata 2 2 min=0 max=1.1
    count = 0
endif
if pass = 1 then
    title thermal profile in glass, step 1, X 10 scale
    plot/watch ex84_3.plt
    plot/l 2 xrad=25 ns=64 max=1.5
    plot/watch ex84_4.plt
    title thermal profile in glass, step 1
    plot/con 2
    lens 1 100          # make a lens to go to far field
    dist 100 1          # propagate to focal point
    title far-field intensity, step 1
    variab/set units 1 units
    variab/set peak 1 peak
    plot/watch ex84_5.plt
    plot/l 1 xrad=[20*units] ns=64
endif
if pass = 300 then
    title thermal profile in glass, step 300
    plot/watch ex84_6.plt
    plot/l 2 xrad=25 ns=64 max=15
    plot/watch ex84_7.plt
    plot/con 2
    lens 1 100          # make a lens to go to far field
    dist 100 1          # propagate to focal point
    title far-field intensity, step 300, normalized to step 40
    variab/set units 1 units
    plot/watch ex84_8.plt
    plot/l 1 xrad=[20*units] ns=64
endif
macro/end
macro thermal/300
title temperature rise vs. time
plot/watch ex84_9.plt
plot/udata 1 1 min=0
title strehl ratio rise vs. time
plot/watch ex84_10.plt
plot/udata 2 2 min=0 max=1.1
end

```


Ex85: Geometrical optics using lensgroup

Table. 85.1. Table of Ex85 examples

Ex85a: Simple lens with a tilted reflector	1
Ex85b: Cooke triplet	2
Ex85c: Tilted Cook triplet	4
Ex85d: Cooke triplet with mirror, reversed lens	6
Ex85e: Cooke triplet with 180 rotation	7
Ex85f: Cooke triplet with 45 tilt mirror and 90 vertex rotation.	9
Ex85g: Example to illustrate propagation through Brewster angle window	10
Ex85h: Propagation of a decentered beam through a microsphere	12
Ex85i: 60-60-60 prism used at minimum deviation	15

This example illustrates the use of the lensgroup features. These features require the implementation of the geometrical optics option. A lens may be described in terms of radius, thickness, glass type, coordinate break, apertures, and obscurations. GLAD will calculate the aberrations, propagation length, and change in divergence of the beam for the lens group.

Ex85a: Simple lens with a tilted reflector

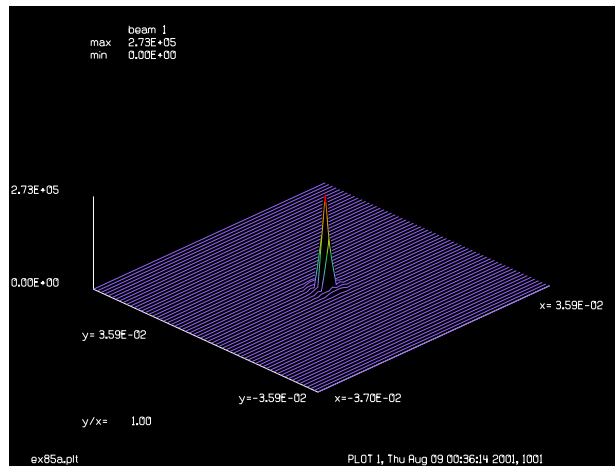


Fig. 85.1. Far-field for Ex. 85a.

Input: `ex85a.inp`

```

c## ex85a
c
c Example to illustrate use of the geometrical optics feature.
c
c A simple single lens is defined
c
lensgroup/def ex85a/o
c
c define object space: wavelength, field height, numerical aperture,

```

```

c distance from object to first surface and pupil distance
c
  object .5876 .8 .008 100. 0. air      # 0, surface
  surface_lensgroup 1.e20 .2 bk7        # 1, plane glass surface
  vertex/rot/set 2                      # rotate vertex 2 deg.
  surface_lensgroup 1e20 -.2 refl       # 2, tipped mirror
  vertex/rot/add 2                      # reset vertex to follow chief ray
  surface_lensgroup 1e20 0. bk7         # 3, another flat refractive surface
  surface_lensgroup 20. -.2 air         # 4
  surface_lensgroup 1.e20 0. air        # 5
  image focus                          # 6
lensgroup/end
wavelength/set 1 .5876
units 1 .05
clap/c/c 1 .8
lens 1 -100                          # use a conventional lens to start beam
c
c list vertex coordinates
c
lensgroup/trace/oneray/default ex85a vertex
c
c trace single rays, listing global coordinates of the rays
c
lensgroup/trace/oneray/default ex85a 0. 0. 0. 1. relative
lensgroup/trace/oneray/default ex85a 0. 0. 0. 0. relative
lensgroup/trace/oneray/default ex85a 0. 0. 0. -1. relative
c
c trace a ray fan
c
lensgroup/trace/yfan/default ex85a 0. 0.
c
c implement the lens
c
lensg/run ex85a 1                     # implement lens with aberrations
strehl                                # list Strehl ratio
geodata
global/list
focus/a 1                           # propagate to paraxial focus
plot/watch ex85a.plt
plot/l 1 ns=64                        # display image at paraxial focus

```

Ex85b: Cooke triplet

Input: ex85b.inp

```

c## ex85b
c
c Example to illustrate a Cooke triplet
c
c This particular lens was designed to work at 20 deg. half angle field
c
array/s 1 128
lensgroup/def cooke/o

```

Jump to: [Commands](#), [Theory](#)

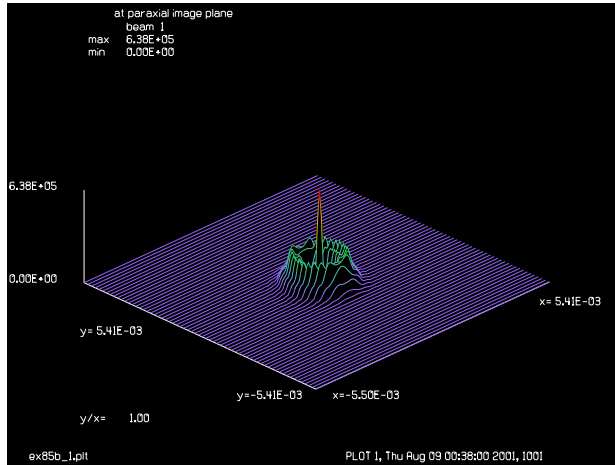


Fig. 85.2. Far-field for Ex. 85b, paraxial focus.

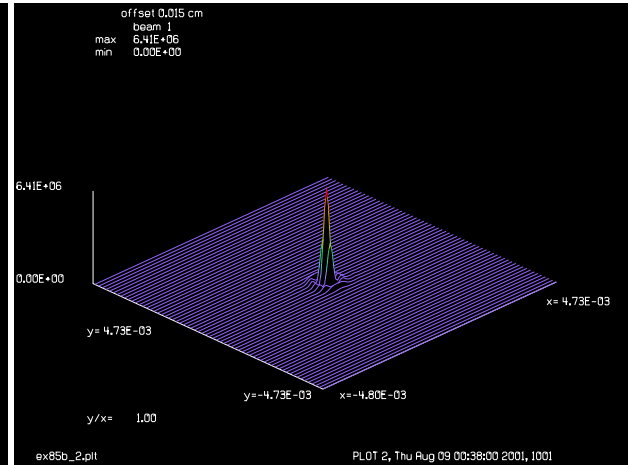


Fig. 85.3. Far-field for Ex. 85b, best focus.

```

object lambda=.55 yfield=y0 yna=.5e-10 zsurf=obj_dist zpupil=pupil_dist air
surface_lensgroup 1e10 0. air
surface_lensgroup 2.18490397 .326326343 sk16
surface_lensgroup -17.2730754 .516337239 air
surface_lensgroup -2.01031718 .099769834 f2
surface_lensgroup 2.33565226 .424584729 air
surface_lensgroup 22.8173221 .1956842 sk16
surface_lensgroup -1.70906459 4.17485291 air
image
lensgroup/end
uc = tan(pi*20/180)          # chief ray paraxial angle
obj_dist = 1e10             # set object distance, to a large number
yp = -.3711                 # set chief ray height at 1st surface
pupil_dist = -yp/uc         # calculate pupil displacement from
                             # 1st surface
yo = uc*(obj_dist + pupil_dist) # calculate object height
variab                      # list variables
wavelength/set 1 .55        # select wavelength
units/set 1 .025            # specify units
clap/c/c 1 .5001           # specify clear aperture
c
c Trace a single ray, in global coordinates
c
lensgroup/trace/oneray/beam cooke 1 0 1 global
c
c Trace a ray fan, using 5 rays
c
lensgroup/trace/yfan/beam cooke nrays=5
c
c Trace a spot diagram and display x- and y-fans
c
lensgroup/trace/spot/beam cooke
c
c Implement the lens. The beam propagates to the last surface of the lens.
c
lensgroup/run cooke 1

```

Jump to: [Commands](#), [Theory](#)

```

geodata
focus/a 1                                # propagate to paraxial image
c
title at paraxial image plane
plot/watch ex85b_1.plt
plot/l 1 ns=64
title offset 0.016 cm
prop .016
plot/watch ex85b_2.plt
plot/l 1 ns=64

```

Ex85c: Tilted Cook triplet

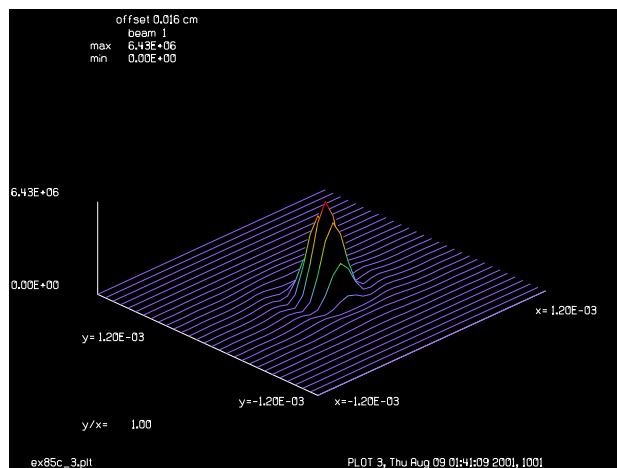


Fig. 85.4. Far-field for Ex. 85c, best focus, no tilt.

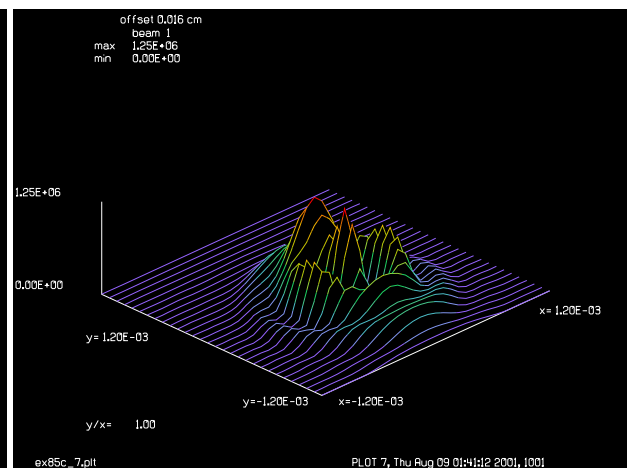


Fig. 85.5. Far-field for Ex. 85c, best focus, 4 deg. tilt.

Input: ex85c.inp

```

c## ex85c
c
c Example to illustrate a Cooke triplet
c
c This particular lens was designed to work at 20 deg. half angle field
c
array/s 1 128
lensgroup/def cooke/o
  object lambda=.55 yfield=y0 yna=.5e-10 zsurf=obj_dist zpupil=pupil_dist air
  surface_lensgroup 1e10 0. air
  surface_lensgroup 2.18490397 .326326343 sk16
  surface_lensgroup -17.2730754 .516337239 air
  surface_lensgroup -2.01031718 .099769834 f2
  surface_lensgroup 2.33565226 .424584729 air
  surface_lensgroup 22.8173221 .1956842 sk16
  surface_lensgroup -1.70906459 4.17485291 air
  image
lensgroup/end
  uc = tan(pi*20/180)                                # chief ray paraxial angle

```

Jump to: [Commands](#), [Theory](#)

```

obj_dist = 1e10          # set object distance, to a large number
yp = -.3711             # set chief ray height at 1st surface
pupil_dist = -yp/uc      # calculate pupil displacement from
                          # 1st surface
yo = uc*(obj_dist + pupil_dist) # calculate object height
variab                  # list variables
wavelength/set 1 .55    # select wavelength
units/s 1 .025          # specify units
clap/c/c 1 .5001        # specify clear aperture
c
c Implement the lens. The beam propagates to the last surface of the lens.
c

if 1 then

Angle = 0
vertex/rotate/set Angle 0 0
vertex/locate/rel 1
lensgroup/run cooke 1
focus/a 1              # propagate to paraxial image
c
title at paraxial image plane
plot/watch ex85c_1.plt
plot/l 1 ns=64
title offset 0.016 cm
prop .016
plot/watch ex85c_2.plt
plot/l 1 ns=64
plot/watch ex85c_3.plt
plot/l 1 xrad=.0012
plot/watch ex85c_4.plt
set/win/abs -.0012 .0012 -.0012 .0013
plot/c 1 ilab=0

endif
c read/scr
array/s 1 128
global/def
zreff/se 1 0
units/s 1 .025
clap/c/c 1 .5001

Angle = 4
vertex/rotate/set Angle 0 0
vertex/locate/rel 1
lensgroup/def cooke/o
  object lambda=.55 yfield=yo yna=.5e-10 zsurf=obj_dist zpupil=pupil_dist air
  surface_lensgroup 1e10 0. air
  surface_lensgroup 2.18490397 .326326343 sk16
  surface_lensgroup -17.2730754 .516337239 air
  surface_lensgroup -2.01031718 .099769834 f2
  surface_lensgroup 2.33565226 .424584729 air
  surface_lensgroup 22.8173221 .1956842 sk16

```

Jump to: [Commands](#), [Theory](#)

```

surface_lensgroup -1.70906459 4.17485291 air
image
lensgroup/end
lensgroup/run cooke 1
focus/a 1                                # propagate to paraxial image
c
title at paraxial image plane
plot/watch ex85c_5.plt
plot/l 1 ns=64
title offset 0.016 cm
prop .016
plot/watch ex85c_6.plt
plot/l 1 ns=64
plot/watch ex85c_7.plt
plot/l 1 xrad=.0012
plot/watch ex85c_8.plt
set/win/abs -.0012 .0012 -.0012 .0013
plot/c 1 ilab=0

```

Ex85d: Cooke triplet with mirror, reversed lens

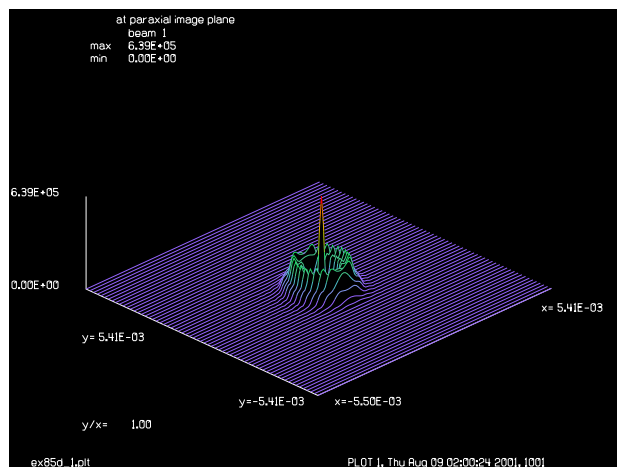


Fig. 85.6. Far-field for Ex. 85d, paraxial focus.

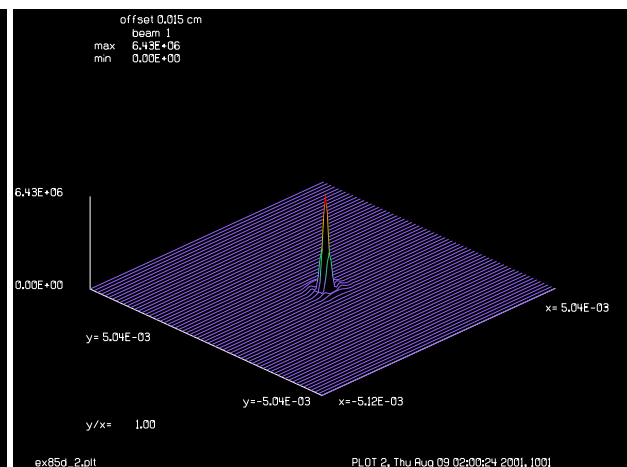


Fig. 85.7. Far-field for Ex. 85d, best focus.

Input: `ex85d.inp`

```

c## ex85d
c
c Example to illustrate a Cooke triplet
c
c In Ex85d a mirror is inserted just in front of the lens
c and the lens is defined backward.
c
array/s 1 128
lensgroup/def cooke/o
  object lambda=.55 yfield=y0 yna=.5e-10 zsurf=obj_dist zpupil=pupil_dist air
  surface_lensgroup 1e10 0. refl

```

Jump to: [Commands](#), [Theory](#)

```

surface_lensgroup -2.18490397 -.326326343 sk16
surface_lensgroup 17.2730754 -.516337239 air
surface_lensgroup 2.01031718 -.099769834 f2
surface_lensgroup -2.33565226 -.424584729 air
surface_lensgroup -22.8173221 -.1956842 sk16
surface_lensgroup 1.70906459 -4.17485291 air
image
lensgroup/end
uc = -tan(pi*20/180)           # chief ray paraxial angle
obj_dist = -1e10              # set object distance, to a large number
yp = -.3711                   # set chief ray height at 1st surface
pupil_dist = -yp/uc           # calculate pupil displacement from
                               # 1st surface
yo = uc*(obj_dist + pupil_dist) # calculate object height
variab
wavelength/set 1 .55          # select wavelength
units/s 1 .025                # specify units
clap/c/c 1 .5001              # specify clear aperture
c
c Trace a single ray, in global coordinates
c
lensgroup/trace/oneray/beam cooke 1 0 1 global
c
c Trace a ray fan, using 5 rays
c
lensgroup/trace/yfan/beam cooke nrays=5
c
c Trace a spot diagram and display x- and y-fans
c
lensgroup/trace/spot/beam cooke
c
c Implement the lens. The beam propagates to the last surface of the lens.
c
lensgroup/run cooke 1
geodata
focus/a 1                     # propagate to paraxial image
c
title at paraxial image plane
plot/watch ex85d_1.plt
plot/l 1 ns=64
title offset 0.015 cm
prop .016
plot/watch ex85d_2.plt
plot/l 1 ns=64

```

Ex85e: Cooke triplet with 180 rotation

Input: ex85e.inp

```

c## ex85e
c
c Example to illustrate a Cooke triplet

```

Jump to: [Commands](#), [Theory](#)

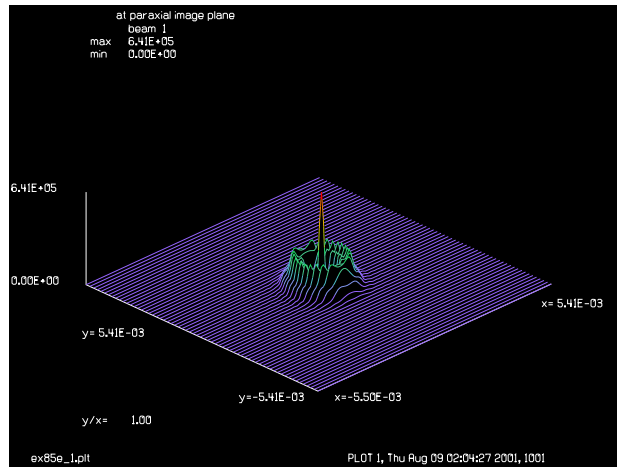


Fig. 85.8. Far-field for Ex. 85e, paraxial focus.

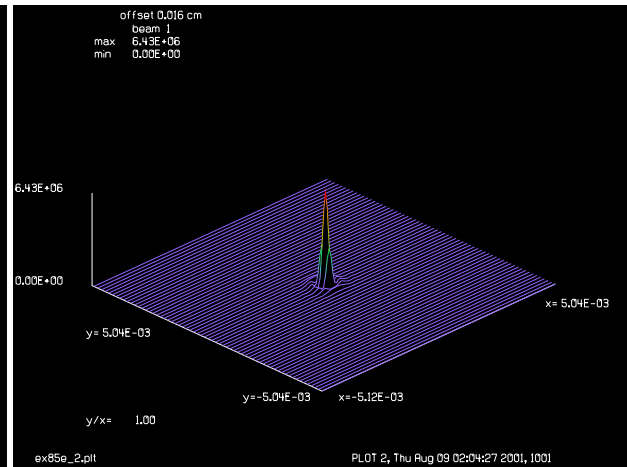


Fig. 85.9. Far-field for Ex. 85e, best focus.

```

c
c In this example a mirror is defined external to the lensgroup and
c the vertex associated with the lensgroup is rotated 180 deg. so the
c lens does not need to be written backward as with Ex85c.
c
array/s 1 128
lensgroup/def cooke/o
  object lambda=.55 yfield=y0 yna=.5e-10 zsurf=obj_dist zpupil=pupil_dist air
  surface_lensgroup 1e10 0. air
  surface_lensgroup 2.18490397 .326326343 sk16
  surface_lensgroup -17.2730754 .516337239 air
  surface_lensgroup -2.01031718 .099769834 f2
  surface_lensgroup 2.33565226 .424584729 air
  surface_lensgroup 22.8173221 .1956842 sk16
  surface_lensgroup -1.70906459 4.17485291 air
  image
lensgroup/end
  uc = tan(pi*20/180)
  obj_dist = 1e10
  yp = -.3711
  pupil_dist = -yp/uc
  y0 = uc*(obj_dist + pupil_dist)
variab
wavelength/set 1 .55
units/s 1 .025
clap/c/c 1 .5001
mirror/global/flat 1
vertex/rotate/add 180 0 0
lensgroup/trace/oneray/beam cooke 1 0 1 global
lensgroup/trace/yfan/beam cooke nrays=5
lensgroup/run cooke 1
geodata
focus/a 1 # propagate to paraxial image
c
title at paraxial image plane
plot/watch ex85e_1.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/1 1 ns=64
title offset 0.016 cm
prop .016
plot/watch ex85e_2.plt
plot/1 1 ns=64

```

Ex85f: Cooke triplet with 45 tilt mirror and 90 vertex rotation

Input: `ex85f.inp`

```

c## ex85f
c
c Example to illustrate a Cooke triplet
c
c In this example a 45 deg. mirror reflects the beam upward.
c The vertex is rotated 90 deg. so the lens looks down into the beam.
c
array/s 1 128
lensgroup/def cooke/o
  object lambda=.55 yfield=yo yna=.5e-10 zsurf=obj_dist zpupil=pupil_dist air
  surface_lensgroup 1e10 0. air
  surface_lensgroup 2.18490397 .326326343 sk16
  surface_lensgroup -17.2730754 .516337239 air
  surface_lensgroup -2.01031718 .099769834 f2
  surface_lensgroup 2.33565226 .424584729 air
  surface_lensgroup 22.8173221 .1956842 sk16
  surface_lensgroup -1.70906459 4.17485291 air
  image
lensgroup/end
  uc = tan(pi*20/180)
  obj_dist = 1e10
  yp = -.3711
  pupil_dist = -yp/uc
  yo = uc*(obj_dist + pupil_dist)
variab
wavelength/set 1 .55
units/s 1 .025
clap/c/c 1 .5001
vertex/rotate/set 45 0 0
global/list
mirror/global/flat 1
global/list
vertex/rotate/set -90 0 0
vertex/list
lensgroup/trace/oneray/beam cooke 1 0 1 global
lensgroup/trace/yfan/beam cooke nrays=5
lensgroup/run cooke 1
geodata
focus/a 1 # propagate to paraxial image
c
title at paraxial image plane
plot/watch ex85f_1.plt
plot/1 1 ns=64

```

Jump to: [Commands](#), [Theory](#)

```

title offset 0.016 cm
prop .016
plot/watch ex85f_2.plt
plot/1 1 ns=64

```

Ex85g: Example to illustrate propagation through Brewster angle window

Input: ex85g.inp

```

c## ex85g
#
# Example to illustrate propagation through windows tilted
# at Brewster's angle
#
variable/declare/integer pass
variable/declare/real lambda nmed nglass inang
variable/declare/real thickness
array/set 1 128 128 1
unit/set 1 0.0015 0.0015
lambda = 0.633
nglass = 1.458                                # index, fused silica
inang=180*atan(nglass)/pi                      # Brewster's angle
L1=50.0                                         # left size of 1st window
L2=100.0                                       # right side of 2nd window
thickness= 0.635                              # window thickness
wavelength/set 1 lambda

# define four windows: two for forward pass and two for backward pass

lensgroup/define bplate1/overwrite
c surface_lensgroup 1e10 0. air
surface_lensgroup radius=1e20 thickness=thickness silica
surface_lensgroup radius=1e20 thickness=0.0 air
image focus
lensgroup/end

lensgroup/define bplate2/overwrite
c surface_lensgroup 1e10 0. air
surface_lensgroup radius=1e20 thickness=thickness silica
surface_lensgroup radius=1e20 thickness=0.0 air
image focus
lensgroup/end

lensgroup/define bplate3/overwrite
c surface_lensgroup 1e10 0. air
surface_lensgroup radius=1e20 thickness=thickness silica
surface_lensgroup radius=1e20 thickness=0.0 air
image focus

lensgroup/end
lensgroup/define bplate4/overwrite
c surface_lensgroup 1e10 0. air
surface_lensgroup radius=1e20 thickness=thickness silica

```

Jump to: [Commands](#), [Theory](#)


```

surface_lensgroup radius=1e20 thickness=0.0 air
image focus
lensgroup/end

#
# start in front of mirror
#
clap/c/c 1 .045
vertex/locate/absolute 0. 0 L1
vertex/rotate/set inang 0. 0. global

lensgroup/trace/xfan/beam bplate4 1
lensgroup/trace/yfan/beam bplate4 1
pause 3

lensgroup/run bplate1 1

variab/set Yray1p 1 yray list # set refracted ray coordinates
variab/set Zray1p 1 zray list
pause 3

title bplate1
plot/w ex85f_1.plt
plot/y
global

Yray2p = Yray1p list # locate 2nd window
Zray2p = L1 + 10 - (Zray1p-L1) list
pause 3

vertex/locate/absolute 0. Yray2p Zray2p
vertex/rotate/set -inang 0. 0. global

lensgroup/run bplate2 1

title bplate2
plot/w ex85f_2.plt
plot/y

vertex/locate/absolute 0. 0. L2 # locate 1st mirror
vertex/rotate/set 0. 0. 0.
mirror/global/conic -100 exact

geodata
global
c clap/cir/no 1 sclaprad
Yray2 = 0. list
Zray2 = L1 + 10 list
pause 3

vertex/locate/absolute 0. Yray2 Zray2 # locate 2nd window, reverse
vertex/rotate/set -inang 0. 0. global
vertex/rotate/add 180. 0. 0.

```

```

geodata
global
lensgroup/run bplate3 1
title bplate3
plot/w ex85f_3.plt
plot/y
global

vertex/locate/absolute 0. Yray1p Zray1p      # locate 1st window, reverse
vertex/rotate/set inang 0. 0. global
vertex/rotate/add 180. 0. 0. global

lensgroup/trace/xfan/beam bplate4 1
lensgroup/trace/yfan/beam bplate4 1
pause 3

lensgroup/run bplate4 1

title bplate4
plot/w ex85f_4.plt
plot/y
global
pause 3

vertex/locate/absolute 0. 0. 0.              # 2nd mirror at z=0
vertex/rotate/set 0. 0. 0. global
mirror/global/conic 100 exact
end

```

Ex85h: Propagation of a decentered beam through a microsphere

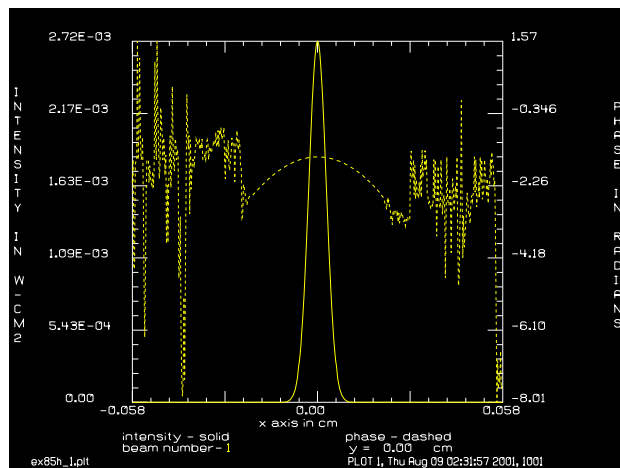


Fig. 85.10. Starting distribution for Ex85h.

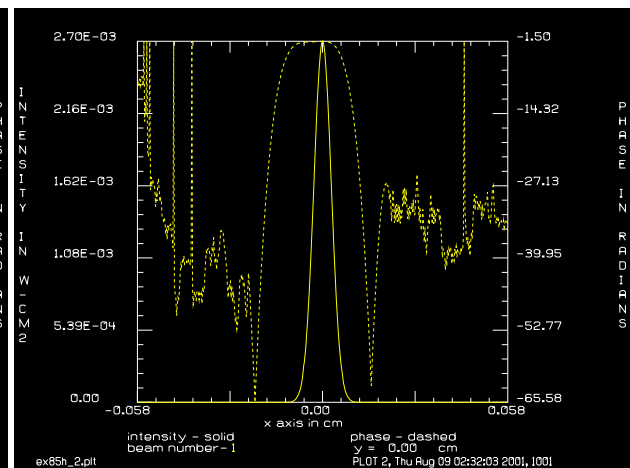


Fig. 85.11. After first surface of ball lens.

Input: ex85h.inp

c## ex85h

Jump to: [Commands](#), [Theory](#)

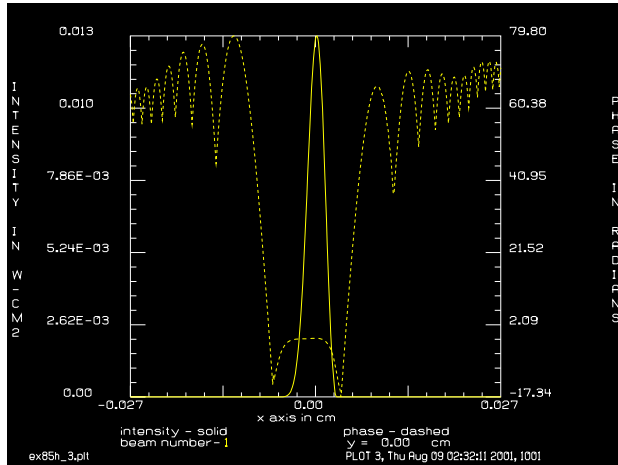


Fig. 85.12. After second surface of ball lens.

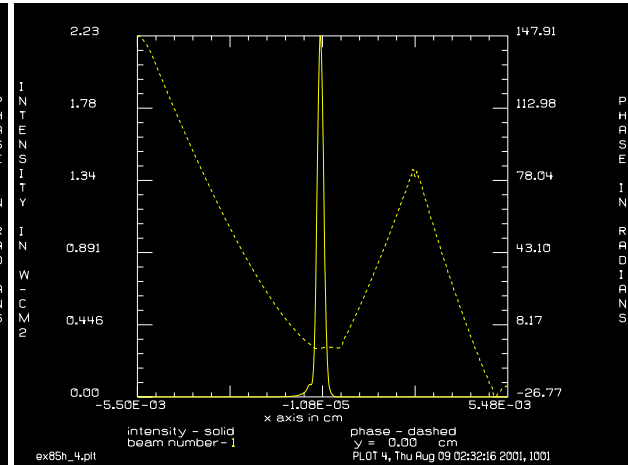


Fig. 85.13. At best focus.

```
#
# Propagation through a sphere where the front and back surfaces
# are treated as separate lens groups to allow propagation between
# the elements.
#
# The beam is injected with a decentration of 0.0040 cm (40 microns).
#
# Propagation of a gaussian with 4um waist through a ball lens.
# Lens radius is .025 cm, refractive index is 1.789 at the wavelength
# of .85 microns. The beam is located .08 cm from the first surface.
#
# Paraxial ray trace
#
# Plane          z          y          u (following plane)
# -----
# object          0.000      .0040000    .00000
# first surface    0.080      .0040000    .00000
# second surface   0.130      .0004718   -.07056
# image plane      0.144     -.0013628   -.14112
#
mem/set/b 8
c
c *****Set variables for fiber*****
c
variab/dec/int pass count nline          # define variable names
variab/dec/r prop
c
decx=40e-4                                # laser decentering
field =150e-4                             # field size for the propagating beam
nline =512                                # propagating array width
c n_glass=1.7988                          # glass refractive index
n_air=1                                   # air refractive index
waist=4e-4                               # generatig gaussian beam waist
lambda=.85                               # laser wavelength in vacuum in um
c
c ***** Set arrays and units *****
```

Jump to: [Commands](#), [Theory](#)

```

c
array/s 1 nline nline 1      # beam 1, propagating beam
# no need to propagate beam 2, set to "data"
mem/cont
units/field 0 field          # beam 1 field half-width of 256 microns
zsurf=.08                    # lens first surface location
lens_rad=.025
lens_thick=2*lens_rad
back_surf=zsurf+lens_thick
fiber_position =100e-4
c
c ***** Lensgroups*****
c
lensgroup/def ball_front/o      # defining the front surface of
    surface_lensgroup rad=lens_rad th=0 LASFN30# defining front surface .25mm
    image focus
lensgroup/end
c
lensgroup/def ball_back/o      # defining the baack surface of
    surface_lensgroup rad=-lens_rad th=0 air    # defining second surface .25mm
    image focus
lensgroup/end
c
c *****Propagation*****
c
wavelength/set 0 lambda        # set wavelength and refractive
global/define 1 x=decx         # decenter the beam reference su
gaussian/c/c 1 .5 waist        # generate hermit-gaussian (1,0)
vertex/locate/abs 0 0 zsurf    # locate lens vertex
prop/vertex
plot/w ex85h_1.plt
plot/x
global
pause 3
lensgroup/run ball_front 1      # propagate beam through the len
global
plot/w ex85h_2.plt
plot/x
pause 3
back_surf=
vertex/locate/abs 0 0 back_surf # locate back surface of the len
lensgroup/run ball_back
global
plot/w ex85h_3.plt
plot/x
pause 3
prop .013                      # propagate to "best" image plan
global
plot/w ex85h_4.plt
plot/x
fitfwhm

```

Ex85i: 60-60-60 prism used at minimum deviation**Input: ex85i.inp**

```

c## ex85i
#
# Example to illustrate use of the geometrical optics feature.
#
# A simple 60-60-60 prism used at minimum deviation. Glass is BK7.
# Convergence of the beam induces 3rd order coma and 5th order
# elliptical coma.
#
lensgroup/def prism/o
#
# define object space: wavelength, field height, numerical aperture,
# distance from object to first surface and pupil distance
#
object .5876 .8 .008 100. 0. air      # 0, surface
vertex_lensgroup/rot/set I0          # rotate vertex to incident
                                     # angle for minimum deviation
surface_lensgroup 1e20 .0 bk7        # plane surface into glass
vertex_lensgroup/locate 0. [-.2*sin(pi*D/180)] [.2*cos(pi*D/180)]
vertex_lensgroup/rot/add -60         # reset vertex to follow chief r
surface_lensgroup 1e20 0. air        # plane surface into air
image focus
lensgroup/end
wavelength/set 1 .5876
Index = 1.5168
I1 = 30
I0 = 180*asin(Index*sin(pi*I1/180))/pi
D = I0-I1
I2 = 60-I1
I3 = 180*asin(Index*sin(pi*I2/180))/pi
Final_Deflect = 2*D                 # Relative to global z-axis
variab
units/s 1 .1
clap/c/c 1 2
lens 1 500                          # add weak convergence
#
# Weak convergence will generate common coma Z(3,1)
# and elliptical coma (3,3)
#
# list vertex coordinates
#
lensgroup/trace/oneray/beam prism vertex
#
# list global coordinates of center ray
#
lensgroup/trace/oneray/beam prism global
lensgroup/run prism 1
global/list

```

Jump to: [Commands](#), [Theory](#)

Ex86: Fiber optics and waveguides

Table. 86.1. Table of Ex86 examples

Ex86a: Straight waveguide	2
Ex86b: Sinusoidal waveguide	6
Ex86c: Waveguide with two close cores	9
Ex86d: Multimode fiber.	12
Ex86e: Array of 11 x 11 fiber cores, detailed analysis	15
Ex86f: Response function for 11 x 11 array of fibers by overlap integral.	17
Ex86g: Focused beam into straight fiber	19
Ex86h: Direct observation of the propagation constant.	20
Ex86i: Slab waveguide treating both guided and free-space direction	23
Ex86j: Analytical calculation of slab waveguide eigenmode	27
Ex86k: Slab waveguide treating both guided and free-space direction.	27
Ex86l: Comparision of BPM mode and gaussian approximation	31
Ex86m: Comparision of BPM and gaussian at critical frequency.	34
Ex86n: Fiber with long radius bend.	34
Ex86o: A pencil of light injected into a dielectric waveguide at near-TIR angle.	36
Ex86p: A pencil of light injected into a tapered waveguide.	42

This example illustrates the some simple cases of circular fiber optics. The optical fiber consists of a core of index 1.532 immersed in a cladding of index 1.5. The higher index core, in these examples, is 1.6 microns in diameter. The wavelength is 0.6328 microns. Without the high index core, an optical beam would simply diverge with propagation. The high index core acts as a positive lens to add convergence to the beam. The combined effects of the high index core and divergence incorporated into a split-step solution as illustrated schematically in Ex86.1. The effect on the phase of the beam is illustrated in Fig. 86.2. Fig. 86.2a shows the first part of the first split-step, where

$$a_1(x, y, z + \Delta z) = a_0(x, y, z + \Delta z)e^{i2\pi\Delta n(x, y)\Delta z} \quad (86.1)$$

In this example, a gaussian beam is started into the fiber. If the gaussian beam has a narrow waist then, initially, the beam will diverge. If the starting gaussian is larger, then the beam converges initially. In the transient regime, the beam tends to go through cycles of blooming and pinching. The cyclical behavior damps out in the about the first 20 microns, for this example, as the higher index modes diffract out of the waveguide and single mode operation takes over.

In numerical modeling, it is necessary to provide some absorption at the edges of the array to emulate the loss of light out of the sides of the cladding. The absorption is represented as a supergaussian distribution with radius of 6 microns. The half-width of the array is set to 8 microns. The radius of the core region of 0.8 microns keeps the optical mode well confined within the 6 micron absorbing region. However the absorbing region, although it is well outside the optical mode, represents the only source of loss. This treatment is useful for determining the mode shape, but a larger array may be required to accurately determine the loss per unit length of the lowest loss mode.

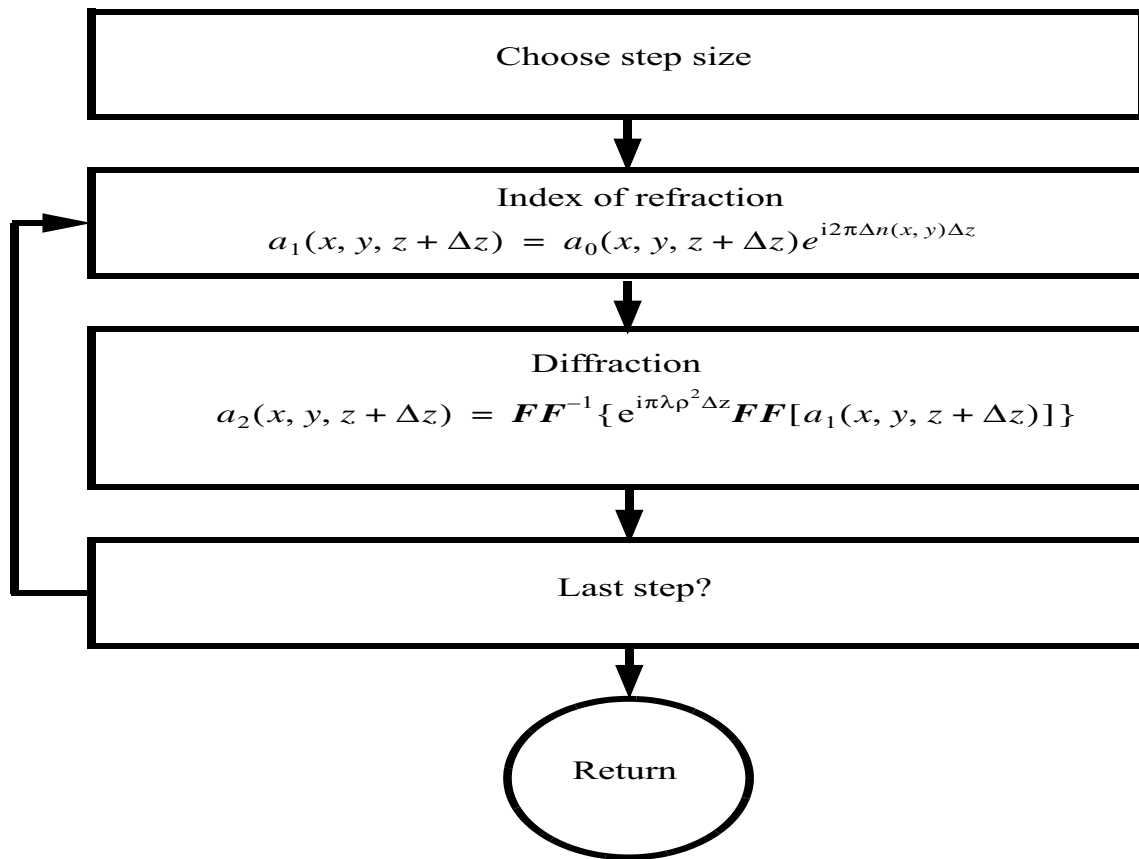


Fig. 86.1. Flow chart for split-step method of treating diffraction and the refractive index function $\Delta n(x, y)$. For a small step Δz , the effect of the refractive index is implemented as a phase screen of the form $e^{i2\pi\Delta n(x, y)\Delta z}$ to change the initial complex amplitude $a_0(x, y, z + \Delta z)$ into the intermediate result $a_1(x, y, z + \Delta z)$. A diffraction step is applied to the intermediate result, implemented by FFT methods, to create the full split-step procedure.

Ex86a: Straight waveguide

The first example, Ex86a, is a simple straight fiber.

Input: `ex86a.inp`

```

c## ex86a
c
c Example 86a: fiber optics propagation, straight fiber
c
c This example illustrates propagation in an optical fiber of
c circular cross section. A gaussian beam is injected into the
c fiber. The initial distribution is transformed into the lowest
c loss mode of the fiber after propagation.
c
c The fiber consists of a cladding of index 1.5 and a core of index
c 1.532, giving a difference in index of 0.032. The modeling directly

```

Jump to: [Commands](#), [Theory](#)

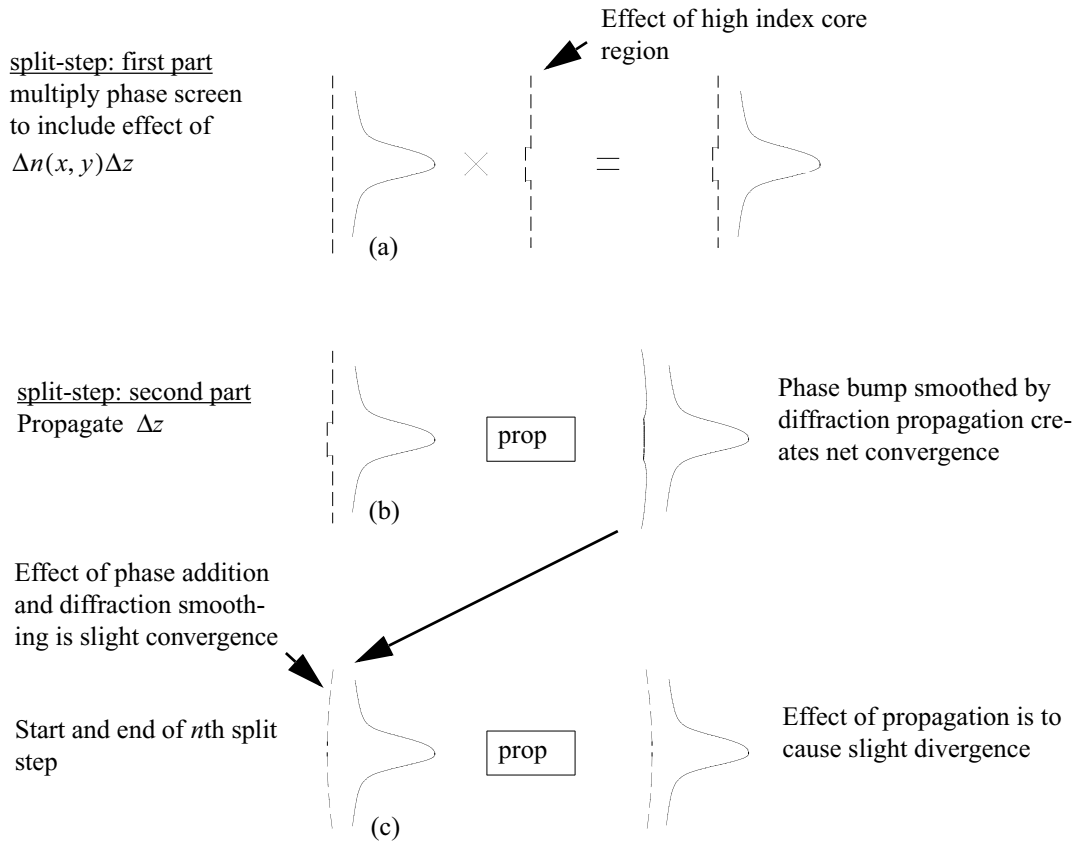


Fig. 86.2. Split step method. Fig. (a) shows the first part of the first split-step. The high index region creates a phase bump on the wavefront. Fig. (b) shows the second part of the first step where both the phase pump and the amplitude are smoothed out somewhat. Fig. (c) shows that the accumulated effect of phase addition and diffraction spreading creates a very slightly converging wavefront that becomes very slightly diverging after the propagation step of Δz . Converging and diverging effects exactly balance for an eigen mode.

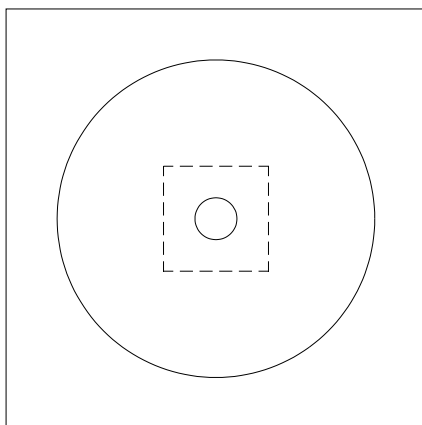


Fig. 86.3. Core region dia. 1.6μ , absorbing wall dia. 12μ , and array of width 16μ .

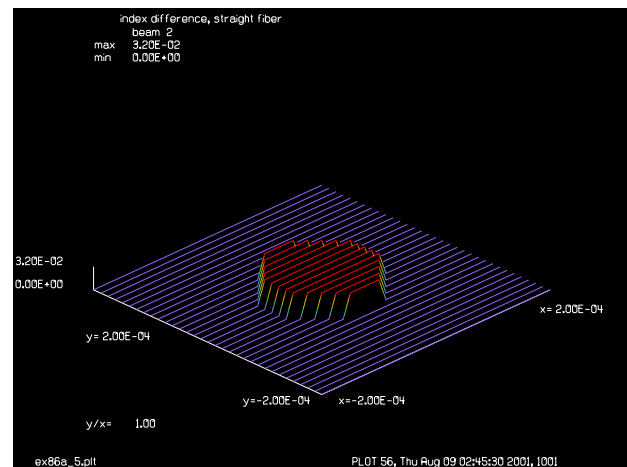


Fig. 86.4. Index distribution of 0.032 to scale of dashed square in Fig. 86.3.

c treats the phase build-up versus distance. Beam 2 contains the

Jump to: [Commands](#), [Theory](#)

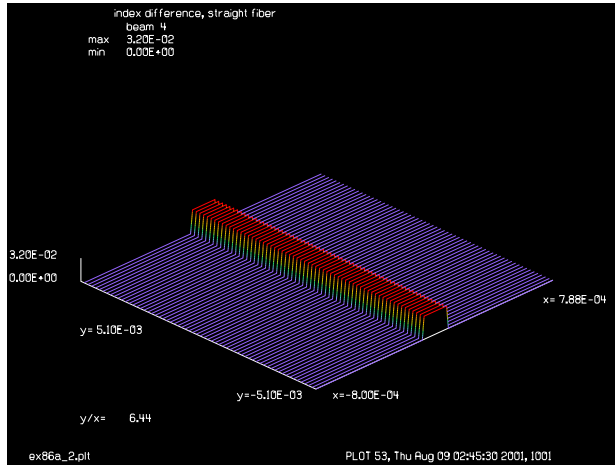


Fig. 86.5. Index difference distribution. Array is $100\ \mu$ long by $16\ \mu$ wide and $\Delta n = 0.032$.

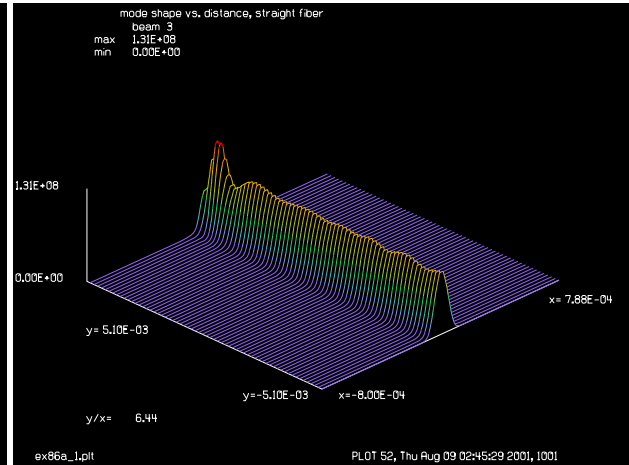


Fig. 86.6. History of irradiance profiles. Mode stabilizes in about $20\ \mu$.

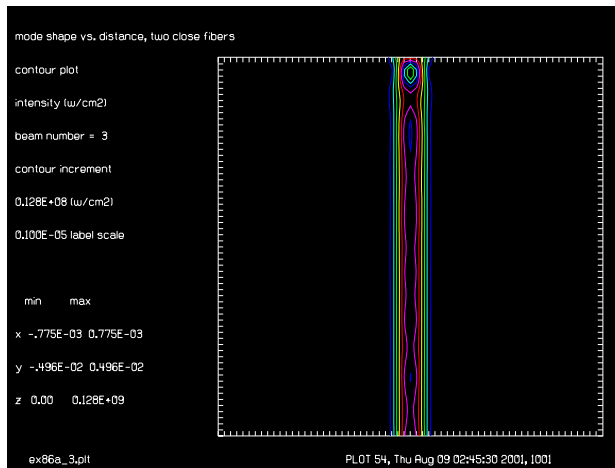


Fig. 86.7. Contour plot of irradiance profiles.

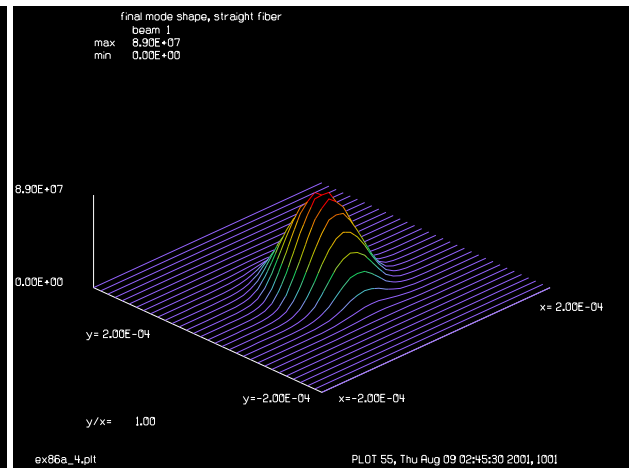


Fig. 86.8. Final mode shape. Same scale as Fig. 86.4.

c pattern of index change associated with the core. The INT2PHAS command
 c is used to implement the incremental index difference defined on
 c Beam 2 as a phase effect on Beam 1.

c
 c It is necessary to have light scattered into the sides of the array
 c be absorbed. The SPLIT/CIR/IN command implements a supergaussian
 c windowing function to absorb light near the edge of the array.

c
 c The optical fiber has a circular cross section, so this is a three-
 c dimensional calculation. To display the mode shape, the irradiance
 c profile, taken across the diameter of the fiber is displayed at
 c each .2 microns of propagation.

```
c
variab/dec/int pass count nline      # define variable names
dist = 2e-5                          # step length
nline = 128                          # propagating array width
title mode shape vs. distance, straight fiber
```

Jump to: [Commands](#), [Theory](#)

```

plot/watch ex86a_1.plt
array/s 1 nline          # beam 1, propagating beam
nbeam 2 data             # beam 2, index distribution
nbeam 3 nline 512 data   # beam 3, history of mode shape
nbeam 4 nline 512 data   # beam 4, history of index profile
units/field 0 8e-4       # field half-width of 8 microns
variab/set units 3 units
units/set 3 units dist   # set units of history arrays
units/set 4 units dist
wavelength/set 0 .6328 1.5 # set wavelength and cladding index
clear 1 1
clear 3 0
clear 4 0
alpha = 2.*pi*dist/.6328e-4 # phase coefficient per step
macro/def step/o
  pass = pass+1
  count = count+1
  zreff/se 1 0
  geodata/set/waist waistx=1e-4 waisty=1e-4
  clear 2 .032           # set index difference for core
  clap/c/n 2 .8e-4
  int2phas/two 1 2 alpha # implement index in beam 2 on beam 1
  split/cir/in 1 6e-4 6  # absorbing boundary for the array
  dist dist 1            # propagation
  if pass = 1 energy/norm 1
  copy/row 1 3 [nline/2+1] pass # store mode in history array
  copy/row 2 4 [nline/2+1] pass # store index profile in history array
  if count = 10 then
    plot/l 3 ns=64        # plot every 10 steps
    count = 0
  endif
macro/end
gaus/c/c 1 1 1e-4        # inject a gaussian mode of 1 micron
                          # radius

pass = 0
macro step/512           # propagate 512 times
plot/watch ex86a_1.plt
plot/l 3 ns=64
plot/watch ex86a_2.plt
title index difference, straight fiber
plot/l 4 ns=64 h=.1
set/density 64
plot/watch ex86a_3.plt
title mode shape vs. distance, straight fiber
plot/con 3 ilab=0
plot/watch ex86a_4.plt
title final mode shape, straight fiber
plot/l 1 xrad=2e-4 ns=64
plot/watch ex86a_5.plt
title index difference, straight fiber
plot/l 2 xrad=2e-4 ns=64 h=.1
energy 1

```

Ex86b: Sinusoidal waveguide

In Ex. 86b the fiber has an “s” curve with beginning and ending straight sections. The mode stabilizes in the initial straight section of 20 microns. The mode is disrupted by the next 62 microns of “s” bend but restabalizes in the last 20 microns of straight fiber. There is a loss of a few percent in this relatively sharp bend. Note that the mode follows the fiber through the “s” curve but there is blooming and pinching because of multiple mode propagation in the curved region.

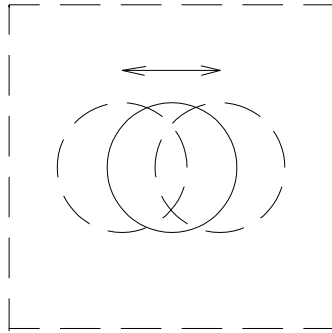


Fig. 86.9. Index difference distribution: first and last 20 μ are straight, inner 60 μ in “s” curve.

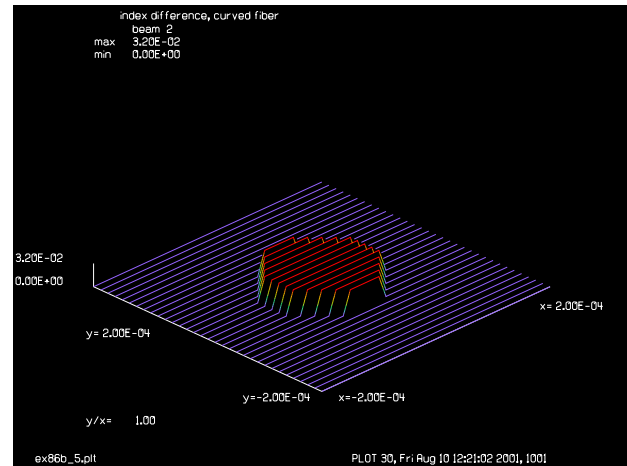


Fig. 86.10. History of irradiance profiles. Mode breaks up in curved regions.

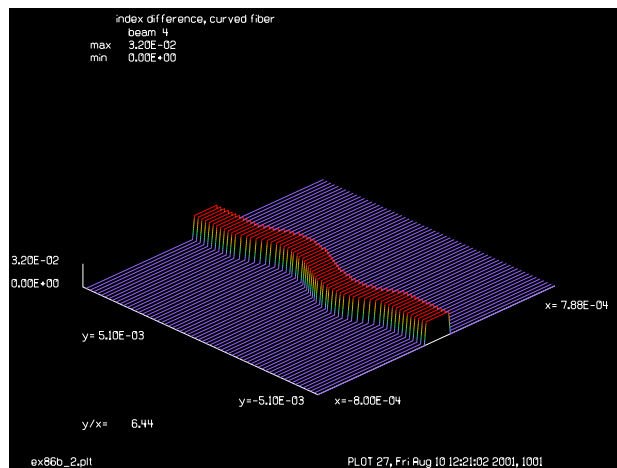


Fig. 86.11. Index difference distribution: first and last 20 μ are straight, inner 60 μ in “s” curve.

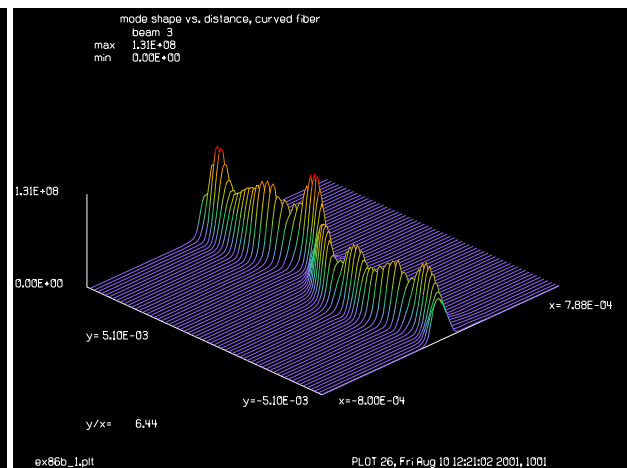


Fig. 86.12. History of irradiance profiles. Mode breaks up in curved regions.

Input: `ex86b.inp`

```
c## ex86b!223063765322918
c
c Example 86b: fiber optics propagation, sinusoidal fiber
c
c This example illustrates propagation in an optical fiber of
```

Jump to: [Commands](#), [Theory](#)

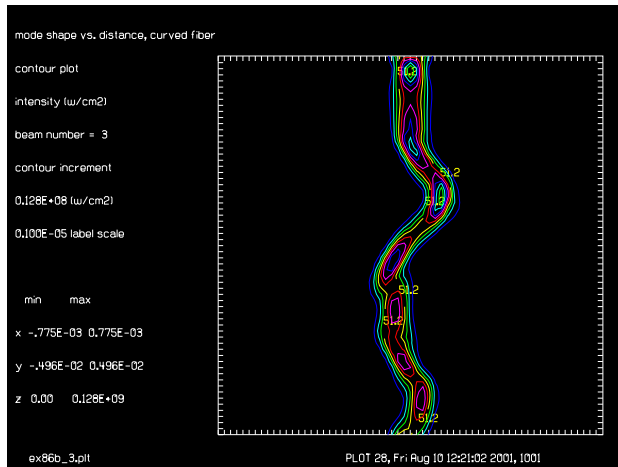


Fig. 86.13. Contour plot of irradiance profiles. Note mode break-up in curves.

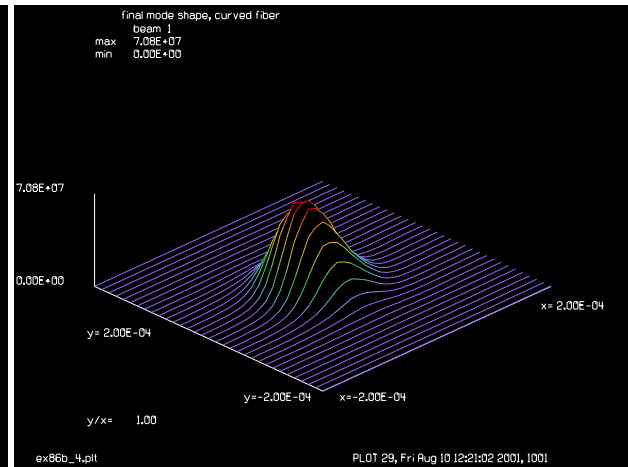


Fig. 86.14. Final mode shape. Same scale as Fig. 86.10.

```

c circular cross section. This example is the same as Ex. 86a,
c except the fiber is bent into a sinusoidal shape. The first
c 20 microns of fiber are straight, the next 60 microns are curved
c in an "s" shape, the last 20 microns are straight.
c
c A gaussian beam is injected into the fiber. The initial distribution
c is transformed into the lowest loss mode of the fiber after propagation.
c The "s" curve disrupts the mode structure but the mode restabalizes
c in the final straight section.
c
c The fiber consists of a cladding of index 1.5 and a core of index
c 1.532, giving a difference in index of 0.032. The modeling directly
c treats the phase build-up versus distance. Beam 2 contains the
c pattern of index change associated with the core. The INT2PHAS command
c is used to implement the incremental index difference defined on
c Beam 2 as a phase effect on Beam 1.
c
c It is necessary to have light scattered into the sides of the array
c be absorbed. The SPLIT/CIR/IN command implements a supergaussian
c windowing function to absorb light near the edge of the array.
c
c The optical fiber has a circular cross section, so this is a three-
c dimensional calculation. To display the mode shape, the irradiance
c profile, taken across the diameter of the fiber is displayed at
c each .2 microns of propagation.
c
variab/dec/int pass count nline      # declare variable names
dist = 2e-5                          # step length
nline = 128                          # propagating array width
title mode shape vs. distance, curved fiber
plot/watch ex86b_1.plt
array/s 1 nline                      # beam 1, propagating beam
nbeam 2 data                         # beam 2, index distribution
nbeam 3 nline 512 data               # beam 3, history of mode shape
nbeam 4 nline 512 data               # beam 4, history of index profile

```

Jump to: [Commands](#), [Theory](#)

```

units/field 0 8e-4                                # field half-width of 8 microns
variab/set units 3 units
units/set 3 units dist                            # set units of history arrays
units/set 4 units dist
wavelength/set 0 .6328 1.5                        # set wavelength and cladding index
clear 1 1
clear 3 0
clear 4 0
alpha = 2.*pi*dist/.6328e-4                      # phase coefficient per step
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0
    geodata/set 1 0 0 1e-4 1e-4 1 1
    clear 2 .032                                  # set index difference for core
    xdec = 0.
    if pass > 100 then
        if pass < 412 then
c
c      add sinewave bending of fiber
c
            xdec = 6*units*sin(2.*pi*(pass-100)/312.)
        endif
    endif
    clap/c/c 2 .8e-4 xdec=xdec                    # build index with variable shift
    int2phas/two 1 2 alpha                        # implement index in beam 2 on beam 1
    split/cir/in 1 6e-4 6                        # absorbing boundary for the array
    dist dist 1
    if pass = 1 energy/norm 1
    copy/row 1 3 [nline/2+1] pass                 # store mode in history array
    copy/row 2 4 [nline/2+1] pass                 # store index profile in history array
    if count = 20 then
        plot/1 3 ns=64                            # plot every 10 steps
        count = 0
    endif
macro/end
gaus/c/c 1 1 1e-4                                # inject gaussian mode into core
pass = 0
macro step/512                                    # propagate 512 times
plot/watch ex86b_1.plt
plot/1 3 ns=64
plot/watch ex86b_2.plt
title index difference, curved fiber
plot/1 4 ns=64 h=.1
set/density 64
plot/watch ex86b_3.plt
title mode shape vs. distance, curved fiber
plot/con 3 ilab=0
plot/watch ex86b_4.plt
title final mode shape, curved fiber
plot/1 1 xrad=2e-4 ns=64
plot/watch ex86b_5.plt
title index difference, curved fiber
plot/1 2 xrad=2e-4 ns=64 h=.1

```

Jump to: [Commands](#), [Theory](#)

```
energy 1
end
```

Ex86c: Waveguide with two close cores

Example 86c shows two cores of diameter 1.6 microns separated by 0.4 microns. A gaussian beam of transverse radius of 1 micron is injected into one of the cores. During 200 microns of propagation the mode moves into the other fiber and then moves back. The mode continues to oscillate back and forth between the cores. This cyclical behavior is typical of identical modes with weak coupling.

GLAD uses scalar Fresnel diffraction theory and the parabolic wave approximation to do diffraction calculations. The small feature sizes used in these examples this theory is not completely accurate. However the results are still indicative of the actual situation.

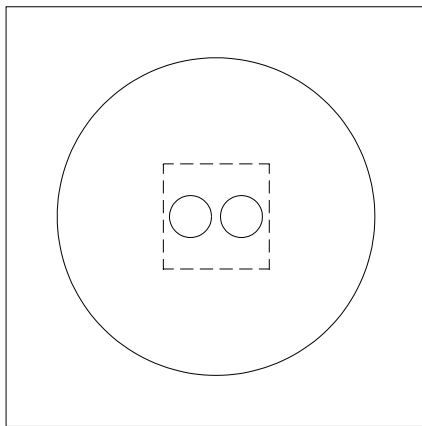


Fig. 86.15. Two core regions of dia. 1.6 μ , separated by 4 μ .

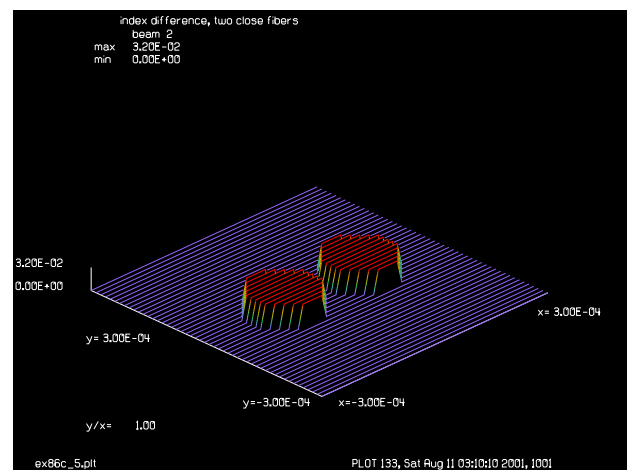


Fig. 86.16. Index distribution of 0.032 to scale of dashed square in Fig. 86.15.

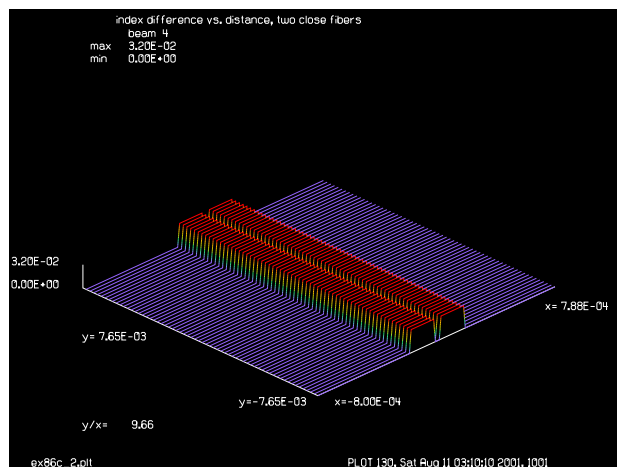


Fig. 86.17. Index difference distribution. Array is 300 μ long by 16 μ wide and $\Delta n = 0.032$.

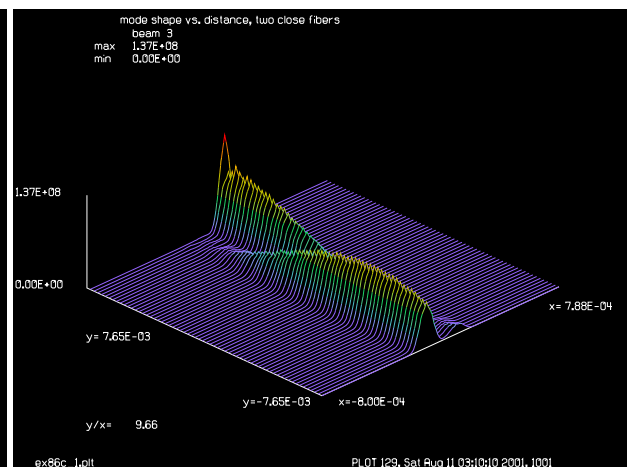


Fig. 86.18. History of irradiance profiles. Mode oscillates between the two core regions.

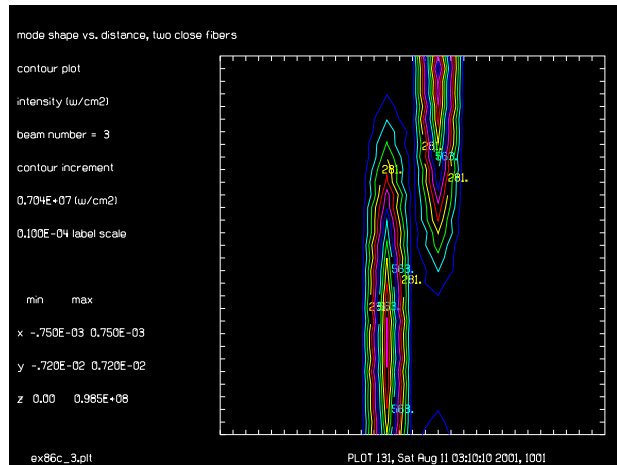


Fig. 86.19. Contour plot of irradiance profiles, showing oscillation of mode between two cores.

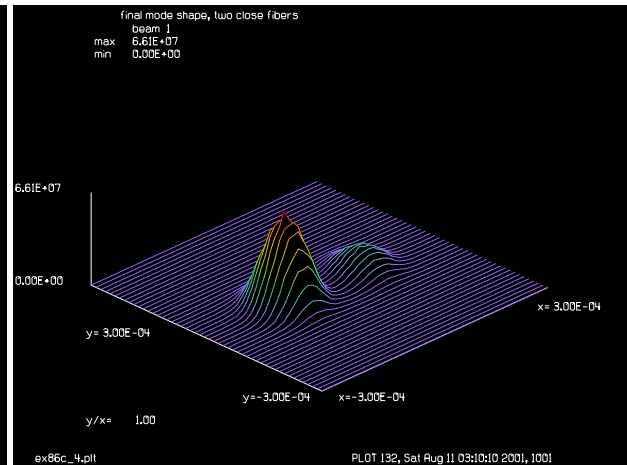


Fig. 86.20. Final mode shape, showing mode partly in each of the cores. Same scale as Fig. 86.16.

Input: ex86c.inp

```
c## ex86c
c
c Example: fiber optics propagation, two close fibers
c
c This example illustrates propagation in an optical fiber
c consisting of two closely spaced circular core regions.
c A gaussian beam is injected into one of the core regions. As
c the beam propagates it moves to the other core and then back
c to the original core. The propagation distance is about 200
c microns. Examples 86a and 86b propagated about 100 microns.
c
c The fiber consists of a cladding of index 1.5 and a core of index
c 1.532, giving a difference in index of 0.032. The modeling directly
c treats the phase build-up versus distance. Beam 2 contains the
c pattern of index change associated with the core. The INT2PHAS command
c is used to implement the incremental index difference defined on
c Beam 2 as a phase effect on Beam 1.
c
c It is necessary to have light scattered into the sides of the array
c be absorbed. The SPLIT/CIR/IN command implements a supergaussian
c windowing function to absorb light near the edge of the array.
c
poptext/compose ex86c.txt
Propagation in a waveguide with two closely spaced
core regions.
```

A gaussian beam is injected in the upper core but gradually crossovers to the lower core because of slight coupling between the cores.

In due course, the beam begins to cross back to the upper core.

Jump to: [Commands](#), [Theory](#)


```

poptext/end
poptext ex86c.txt 30
c echo/on
variab/dec/int pass count nlinex nliney # define variable names
dist = 3e-5                                # step length
nlinex = 128                               # propagating array width
nliney = 512                               # length of history arrays
shift = 1.e-4
title mode shape vs. distance, two close fibers
plot/watch ex86c_1.plt
array/s 1 nlinex                          # beam 1, propagating beam
nbeam 2 data                              # beam 2, index distribution
nbeam 3 nlinex nliney data                # beam 3, history of mode shape
nbeam 4 nlinex nliney data                # beam 4, history of index profile
units/field 0 8e-4                        # field half-width of 8 microns
variab/set units 3 units
units/set 3 units dist                    # set units of history arrays
units/set 4 units dist
wavelength/set 0 .6328 1.5                # set wavelength and cladding index
clear 1 .032
clear 2 .032
clap/c/c 1 .8e-4 xdec=-shift               # make left core
clap/c/c 2 .8e-4 xdec=shift               # make right core
add/inc/con 2 1                           # sum the two cores
clear 1 1
clear 3 0
clear 4 0
set/density 64
alpha = 2.*pi*dist/.6328e-4               # phase coefficient per step
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0
    geodata/set/waist 1 waistx=1e-4 waisty=1e-4 1 1
c
c take two steps together
c
    int2phas/two 1 2 alpha                 # implement index in beam 2 on beam 1
    split/cir/in 1 6e-4 6                 # absorbing boundary for the array
    dist dist 1
    int2phas/two 1 2 alpha                 # implement index in beam 2 on beam 1
    split/cir/in 1 6e-4 6                 # absorbing boundary for the array
    dist dist 1
    if pass = 1 energy/norm 1
    copy/row 1 3 [nlinex/2+1] pass         # store mode in history array
    copy/row 2 4 [nlinex/2+1] pass         # store index profile in history array
    if count = 4 then
        plot/watch plot1.plt
        plot/l 3 ns=64                    # plot every 10 steps
    endif
    if count = 8 then
        poptext/compose ex86c.txt
Energy transfer between two close waveguides

```

```

pass @pass of @nliney
poptext/end
poptext ex86c.txt 8
    plot/watch plot1.plt
    plot/c 3 ilab=0                # plot every 10 steps
    count = 0
endif
macro/end
set/density 32 32
gaus/c/c 1 1 1e-4 decx=shift      # inject gaussian mode into one core
pass = 0
macro step/nliney                # propagate nliney times
plot/watch ex86c_1.plt
plot/l 3 ns=64
plot/watch ex86c_2.plt
title index difference vs. distance, two close fibers
plot/l 4 ns=64 h=.1
plot/watch ex86c_3.plt
title mode shape vs. distance, two close fibers
plot/con 3 con=14 ilab=0
plot/watch ex86c_4.plt
title final mode shape, two close fibers
plot/l 1 xrad=3e-4 ns=64
plot/watch ex86c_5.plt
title index difference, two close fibers
plot/l 2 xrad=3e-4 ns=64 h=.1
energy 1
pause 5
end

```

Ex86d: Multimode fiber

Example 86d shows a multimode fiber supporting about 100 modes. Random noise is injected into the fiber to seed all spatial modes. With a multimode waveguide, the distribution will change rapidly during propagation as the modes beat together due to different propagation constants. However, after about 100 μ of propagation the statistics of the process are stationary. This is evaluated by taking the correlation between the n th and $n-3$ mode. For free-space propagation, the correlation would go from 0 to 1, after sufficient propagation distance. However, in a waveguide a set of spatial modes are propagating with essentially no losses so the correlation asymptotically approaches a value less than 1.

Input: ex86d.inp

```

c## ex86d
c
c Example: fiber optics propagation, multimode fiber
c
c This example illustrates a multimode fiber. The fiber is
c 20 microns in diameter with index difference of 0.032. About a hundred
c spatial modes are supported. A noise distribution was injected
c into the fiber to seed all spatial modes.
c

```

Jump to: [Commands](#), [Theory](#)

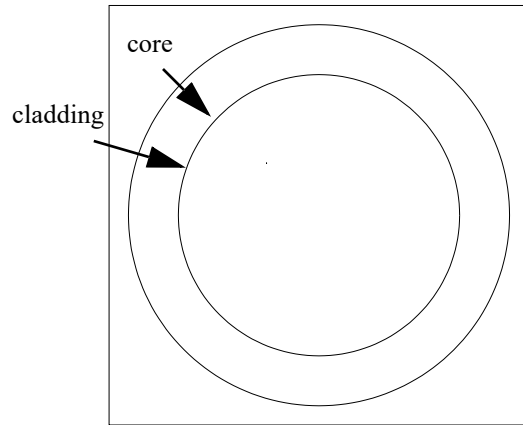


Fig. 86.21. Ex86d models a multimode fiber of 20 μ diameter. The 256 x 256 array spans 32 μ . The inner circle is the core; the outer the absorbing band.

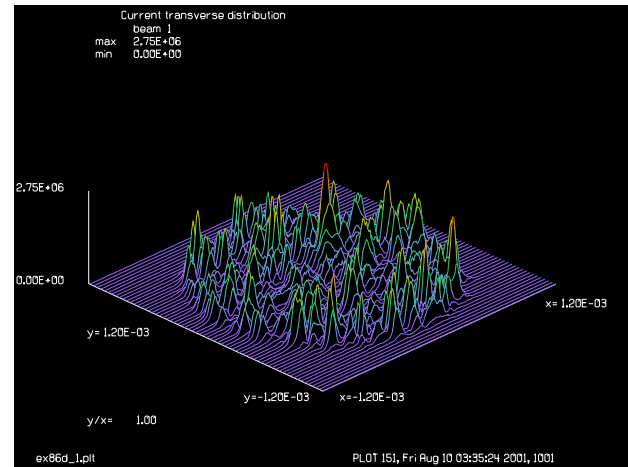


Fig. 86.22. Speckle pattern after 300 μ propagation. The speckles change but are statistically stationary.

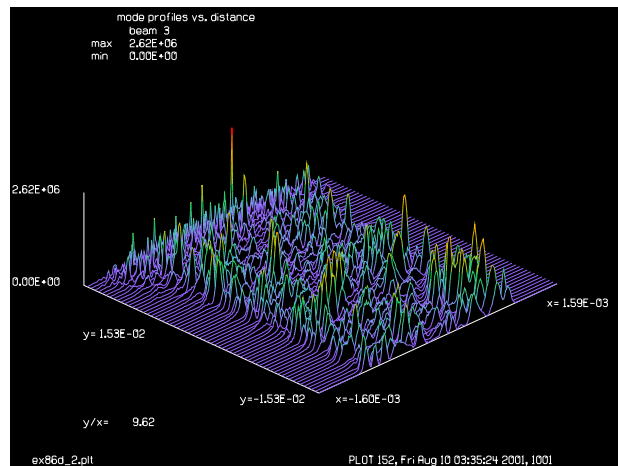


Fig. 86.23. Ex86d models a multimode fiber of 20 μ diameter. The 256 x 256 array spans 32 μ . The inner circle is the core; the outer the absorbing band.

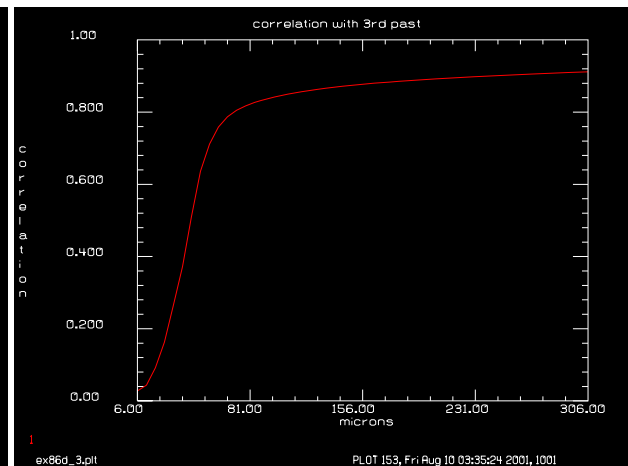


Fig. 86.24. Speckle pattern after 300 μ propagation. The speckles change but are statistically stationary.

c With a multimode waveguide, the distribution will change rapidly
 c during propagation as the modes beat together due to different
 c propagation constants. However after about 100 microns of propagation
 c the statistics of the process are stationary. This is evaluated
 c by taking the correlation between the nth and n-3 mode.

c
 c For free-space propagation, the correlation would go from 0 to 1,
 c after sufficient propagation distance. However, in a waveguide
 c a set of spatial modes are propagating with essentially no losses
 c so the correlation asymptotically approaches a value less than 1.

```
c
variab/dec/int pass count nline
dist = 6.e-5
nline = 256
```

```
# define variable names
# step length
# propagating array width
```

Jump to: [Commands](#), [Theory](#)

```

title mode shape vs. distance, straight fiber
plot/watch exx_1.plt
array/s 1 nline                                # beam 1, propagating beam
nbeam 2 data                                    # beam 2, index distribution
nbeam 3 nline 512 data                          # beam 3, history of mode shape
nbeam 4 nline 512 data                          # beam 4, history of index profile
nbeam 5 nline nline data
mem/cont
units/field 0 16e-4                            # field half-width of 16 microns
variab/set units 3 units
units/set 3 units dist                         # set units of history arrays
units/set 4 units dist
wavelength/set 0 .6328 1.5                     # set wavelength and cladding index
clear 1 0
clear 3 0
clear 4 0
udata/ylabel correlation of nth step with n-3
udata/xlabel microns of propagation
alpha = 2.*pi*dist/.6328e-4                    # phase coefficient per step
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/set 1 0
    geodata/set/waist 1 waistx=1e-4 waisty=1e-4
    clear 2 .032                                # set index difference for core
    clap/c/n 2 10e-4
    int2phas/two 1 2 alpha                      # implement index in beam 2 on beam 1
    split/cir/in 1 15e-4 8                     # absorbing boundary for the array
    dist dist 1                                # propagation
    energy/norm 1
    copy/row 1 3 [nline/2+1] pass               # store mode in history array
    copy/row 2 4 [nline/2+1] pass               # store index profile in history array
    if count = 7 copy/con 1 5
    if count = 10 then
        udata = udata+1
        title Current transverse distribution
        plot/watch ex86d_1.plt
        plot/l 1 ns=64 xrad=12e-4
        plot/watch ex86d_2.plt
        title mode profiles vs. distance
        plot/l 3 ns=64                          # plot every 10 steps
        mult/mode/corr 1 5
        variab/set abscorr abscorr
        udata/set udata [pass*dist*1e4] abscorr
        plot/watch ex86d_3.plt
        title correlation with 3rd past
        plot/udata min=0 max=1
        count = 0
    endif
macro/end
noise 1 1                                       # add delta-correlated noise
pass = 0
macro step/512                                # propagate 512 times

```

Jump to: [Commands](#), [Theory](#)

Ex86e: Array of 11 x 11 fiber cores, detailed analysis

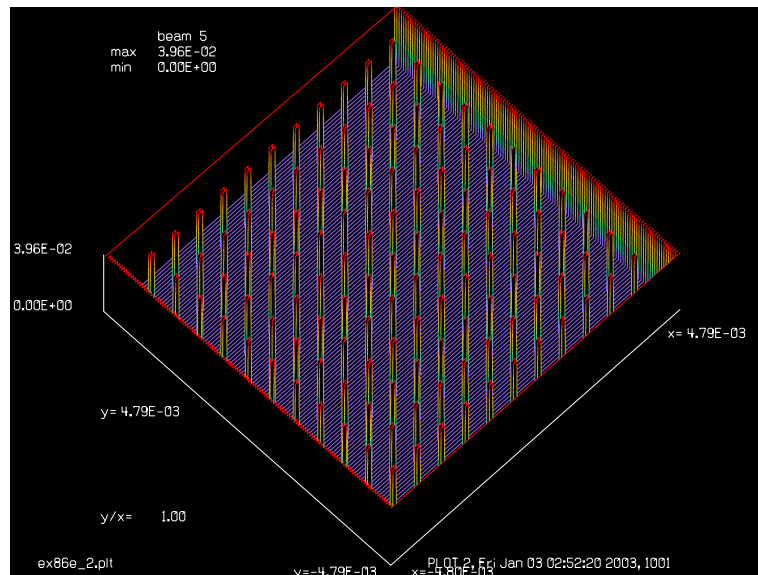


Fig. 86.25. 11 x 11 array of fiber cores.

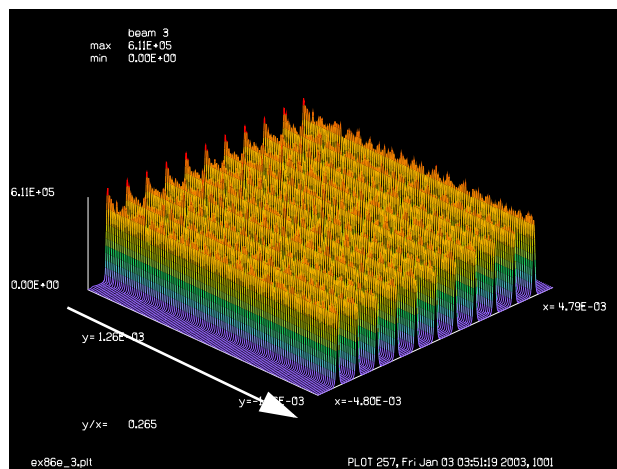


Fig. 86.26. X-slice showing history of propagation of 11 x 11 array of fiber cores. Propagation is from left to right.

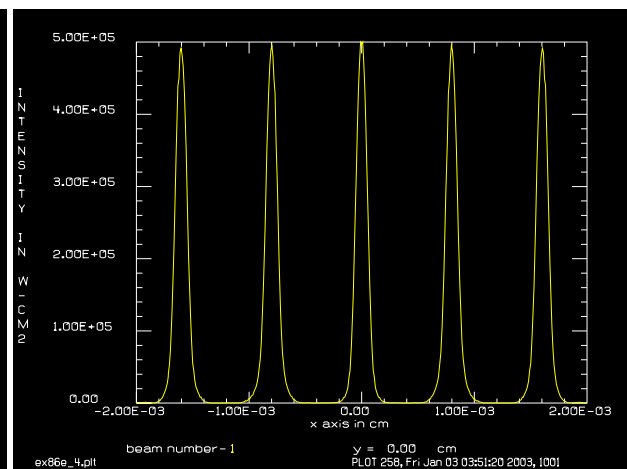


Fig. 86.27. X-slice at conclusion of propagation calculation for 11 x 11 array of fiber cores.

Input: `ex86e.inp`

```
c## ex86e
c
c Example: 11 x 11 array of cores, detailed analysis
c
c This example illustrates propagation in an optical fiber of
c circular cross section. A gaussian beam is injected into the
c fiber. The initial distribution is transformed into the lowest
c loss mode of the fiber after propagation.
```

Jump to: [Commands](#), [Theory](#)

```

c
c The fiber consists of a cladding of index 1.5 and a core of index
c 1.520, giving a difference in index of 0.020. The modeling directly
c treats the phase build-up versus distance. Beam 2 contains the
c pattern of index change associated with the core. The INT2PHAS command
c is used to implement the incremental index difference defined on
c Beam 2 as a phase effect on Beam 1.
c
c The optical fiber has a circular cross section, so this is a three-
c dimensional calculation. To display the mode shape, the irradiance
c profile, taken across the diameter of the fiber is displayed at
c each 3.2 microns of propagation.
c
set/cpu 2
variab/dec/int pass count count1 nline Ntimes Ns # define variable names
variab/dec/int SWITCH
SWITCH = 1 # 0, uniform beam start; 1, array of gaussians
Dist = 2e-5 # step length
Ntimes = 8
nline = 1024 # propagating array width
Width = 8e-4
Field = 6*Width
Units = Field/nline*2
Nz=128
array/s 1 nline # beam 1, propagating beam
nbeam 2 data # beam 2, index distribution
nbeam 3 nline Nz data # beam 3, history of mode shape
nbeam 4 nline Nz data # beam 4, history of index profile
nbeam 5 nline data
nbeam 6 [nline/Ntimes] data # array for forming one cell
units/s 0 Units # set units
variab/set units 3 units
units/set 3 units Dist # set units of history arrays
units/set 4 units Dist
Lambda = .635e-4
wavelength/set 0 Lambda*1e4 1.5 # set wavelength and cladding index
clear 1 1
clear 3 0
clear 4 0
alpha = 2.*pi*Dist/Lambda # phase coefficient per step
dN = .020
clear 6 dN # set index difference for core
clap/c/c 6 .8e-4
c
c Build lensarray in beam 1 as temporary storage
c
Apt = ,8e-4 # radius of phase region
clap/c/c 6 Apt
clear 1 0
lensarray/rec/paste/all 1 6 Width # copy beam 6 into all cells
clap/s/c 1 5.5*Width # clip region outside cells
c
c Build phase array
c

```

Jump to: [Commands](#), [Theory](#)

```

int2phas/two 5 1 alpha          # implement index in beam 2 on beam 1
clap/s/c 5 .98*Field            # absorbing boundary for the array
plot/w ex86e_1.plt
plot/l 5 ns=128
plot/w ex86e_2.plt
plot/l/ph 5 ns=128 theta=42 h=.2
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0
    geodata/set/waist 1 waistx=1e-4 waisty=1e-4
    mult/beam 1 5
    dist Dist 1                  # propagation
    if pass = 1 energy/norm 1
    if count = 16 then
        count1 = count1 + 1
        copy/row 1 3 [nline/2+1] count1      # store mode in history array
        copy/row 2 4 [nline/2+1] count1      # store index profile in history arr
        plot/w ex86e_3.plt
        plot/l 3 ns=Nz                    # plot every 20 steps
        plot/w ex86e_4.plt
        plot/x/i 1 left=[-2.5*Width] right=[2.5*Width]
        count = 0
    endif
macro/end
if SWITCH = 1 then
    gaus/c/c 6 1 1.1e-4            # inject a gaussian mode of 1 micron
    lensarray/rec/paste/all 1 6 Width # make an array of gaussians
endif
pass = 0
macro step/2048                  # propagate 2048 times

```

Ex86f: Response function for 11 x 11 array of fibers by overlap integral

A simple overlap integral treatment of each fiber in the array against the distribution that lands on the fiber array will give a good approximation for the response of an array of fibers. This example uses a simple gaussian as a test input function and extracts array square regions centered about the elements of an 11 x 11 array. The command file computes the overlap integral that represents the response of the fiber to the local neighborhood of the incident test function and then copies this response function back into the array so the original functions is replaced by the array of appropriately weighted response functions.

Input: ex86f.inp

```

c## ex86f
#
# Response function for an 11 x 11 array of fiber optic cores
#
# The mode response of each fiber optic element is assumed to be
# a gaussian function. Each element of the 11 x 11 array is selected
# and copied to Beam 2. The overlap integral is calculated for the
# fiber optic mode stored in Beam 3 and the component parallel to

```

Jump to: [Commands](#), [Theory](#)

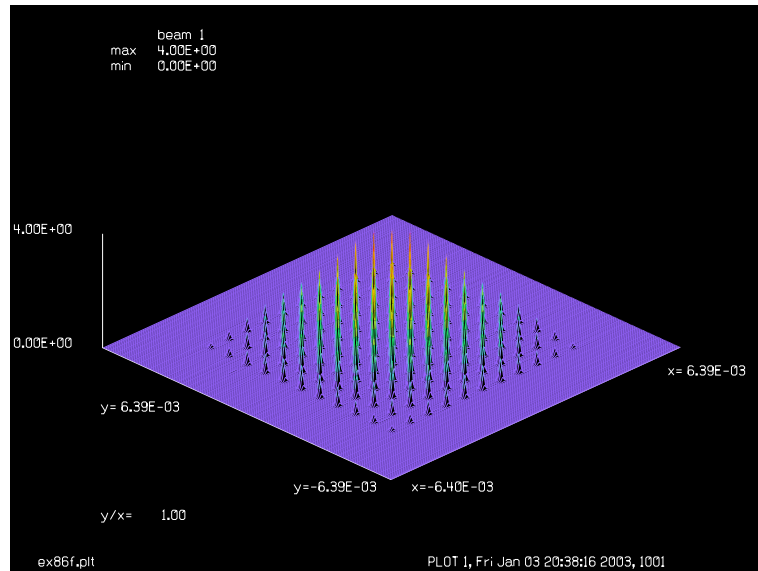


Fig. 86.28. Response function for an 11 x 11 array of fiber optic cores.

```
# Beam 3 is left in Beam 2 and then copies back into Beam 1.
#
variab/dec/int Nline1 Nline2 Ntimes icent jcent
Nline1 = 1024
Nline2 = 64
Ntimes = Nline1/Nline2
Width = 8e-4
Field = Ntimes/2*Width
Units = Field/Nline1*2
array/s 1 Nline1
nbeam 3 Nline2
macro/def outer/o
    j = j + 1
    icent = Nline2/2
    i = 0
    mac inner/[Ntimes+1]
    jcent = jcent + Nline2
macro/end
macro/def inner/o
    i = i + 1
    clear 2 0
    copy/border 1 2 icent jcent
    mult/mode/parallel 2 3
    copy/border 2 1 -icent -jcent
    icent = icent - Nline2
macro/end
jcent = -Nline2/2
j = 0
units/s 0 Units
gaus/c/c 1 1 Field*.7           # set up a gaussian in Beam 1 as
                                # a test function

gaus/c/c 3 1 1.1e-4
clap/s/c 1 5.5*Width           # clip to the 11 x 11 array
```

Jump to: [Commands](#), [Theory](#)


```

mac outer/[Ntimes+1]
plot/w plot1.plt
plot/l 1 ns=256

```

Ex86g: Focused beam into straight fiber

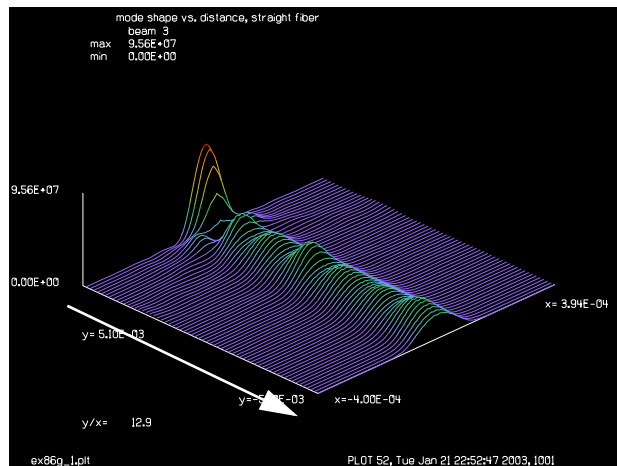


Fig. 86.29. Transient region of waveguide after injection of a focused beam into the waveguide.

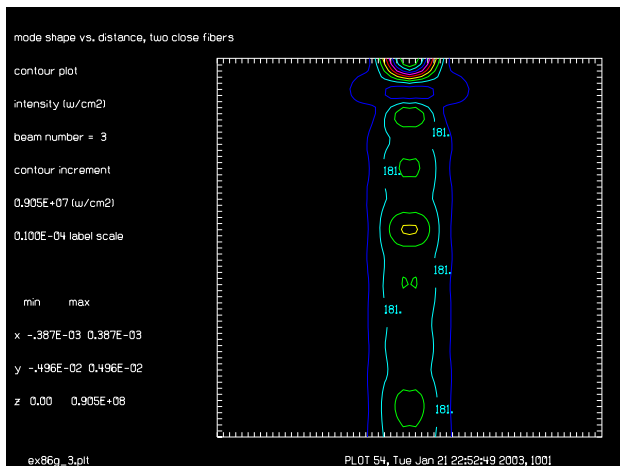


Fig. 86.30. Transient region of waveguide after injection of a focused beam into the waveguide.

Input: ex86g.inp

```

c## ex86g
#
# Response function for an 11 x 11 array of fibers by overlap integral
#
# The mode response of each fiber optic element is assumed to be
# a gaussian function. Each element of the 11 x 11 array is selected
# and copied to Beam 2. The overlap integral is calculated for the
# fiber optic mode stored in Beam 3 and the component parallel to
# Beam 3 is left in Beam 2 and then copies back into Beam 1.
#
mem/set/b 10
variab/dec/int Nline1 Nline2 Ntimes icent jcent
Nline1 = 1024
Nline2 = 64
Ntimes = Nline1/Nline2
Width = 8e-4
Field = Ntimes/2*Width
Units = Field/Nline1*2
array/s 1 Nline1
nbeam 3 Nline2
macro/def outer/o
    j = j + 1
    icent = Nline2/2

```

Jump to: [Commands](#), [Theory](#)

```

i = 0
mac inner/[Ntimes+1]
jcent = jcent + Nline2
macro/end
macro/def inner/o
i = i + 1
clear 2 0
copy/border 1 2 icent jcent
mult/mode/parallel 2 3
copy/border 2 1 -icent -jcent
icent = icent - Nline2
macro/end
jcent = -Nline2/2
j = 0
units/s 0 Units
gaus/c/c 1 1 Field*.7          # set up a gaussian in Beam 1 as
                                # a test function
gaus/c/c 3 1 1.1e-4
clap/s/c 1 [5.5*Width]        # clip to the 11 x 11 array
mac outer/[Ntimes+1]
plot/w plot1.plt
plot/l 1 ns=256

```

Ex86h: Direct observation of the propagation constant

Typically we use a coordinate system that moves with the beam, e^{jkz} , based on the index of the cladding, n_0 . The phase of the eigenmode advances faster than coordinate system. We can see this by displaying the real part of the distribution $\text{Re}\{\exp[jk(\beta - k)z]\} = \cos[(\beta - k)z]$. We display this in a history plot. With no piston correction (the phase/piston line below is out) a modulation of about 2.75 cycles is observed over the 512 steps. The value of 2.75 cycles per 512 steps indicates that the phase advance is 1.93 degrees per step. By trial and error we find that -1.96 degrees per step almost perfectly compensates the phase advance.

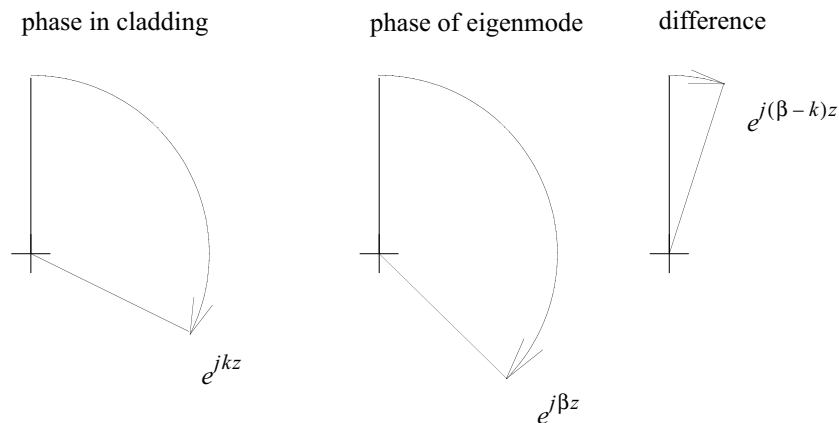


Fig. 86.31. Phase advance of eigenmode relative to base index propagation.

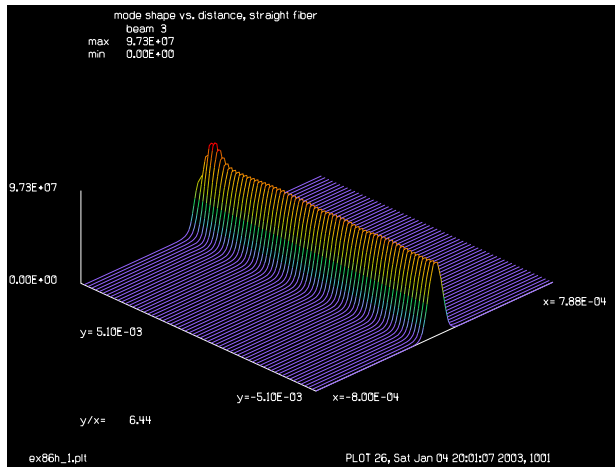


Fig. 86.32. Irradiance vs. distance for straight fiber.

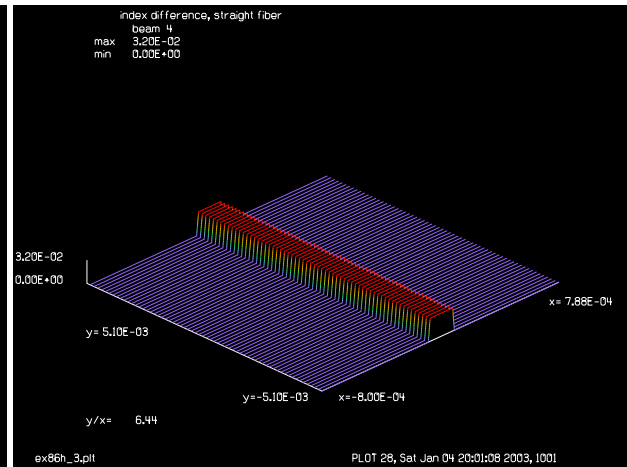


Fig. 86.33. Higher index path due to rib waveguide.

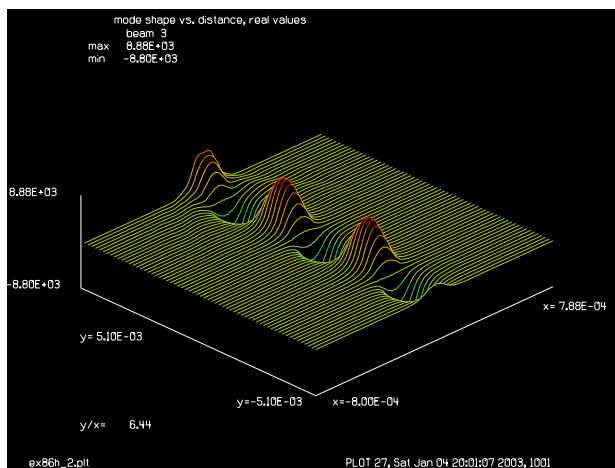


Fig. 86.34. Real component of mode shape vs. distance. No piston compensation.

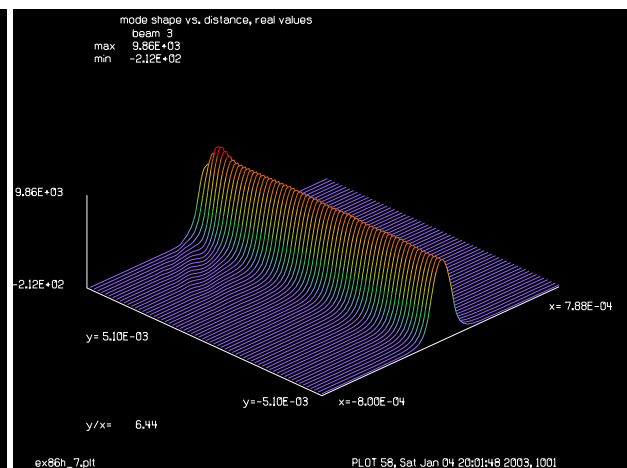


Fig. 86.35. Real component of mode shape vs. distance with piston compensation of -1.96 degrees.

Input: `ex86h.inp`

```
c## ex86h
c
c Example 86h: Direct observation of the propagation constant
c
c Typically we use a coordinate system that moves with the beam.
c The the phase advance of the actual eigenmode proceeds at
c a slightly faster rate because the effective index is higher
c than the base index of the waveguide but lower than the index
c of the rib waveguide.
c
c The phase advance may be observed by displaying the real part
c of the eigenmode in a history plot. With no phase piston correction
c (phase/piston line below is commented out) a modulation of 2.75 cycles
c is observed over the 512 steps.
```

Jump to: [Commands](#), [Theory](#)

```

c
c The value of 2.75 cycles per 512 steps indicates that the phase advance
c is 1.93 degrees per step. By trial and error we find that -1.96
c degrees per step almost perfectly compensates the phase advance
c
alias Name ex86h
variab/dec/int pass count nline          # define variable names
dist = 2e-5                               # step length
nline = 128                               # propagating array width
nliney = 512
title mode shape vs. distance, straight fiber
plot/watch @Name_1.plt
array/s 1 nline                           # beam 1, propagating beam
nbeam 2 data                              # beam 2, index distribution
nbeam 3 nline nliney data                 # beam 3, history of mode shape
nbeam 4 nline nliney data                 # beam 4, history of index profile

units/field 0 8e-4                        # field half-width of 8 microns

variab/set units 3 units
units/set 3 units dist                    # set units of history arrays
units/set 4 units dist
wavelength/set 0 .6328 1.5                # set wavelength and cladding index
clear 1 1
clear 3 0
clear 4 0
alpha = 2.*pi*dist/.6328e-4               # phase coefficient per step
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0
    geodata/set/waist 1 waistx=1e-4 waisty=1e-4
    clear 2 .032                           # set index difference for core
    clap/c/n 2 .8e-4
    int2phas/two 1 2 alpha                 # implement index in beam 2 on beam 1
    split/cir/in 1 6e-4 6                  # absorbing boundary for the array
    dist dist 1                            # propagation
    phase/piston 1 Phase
    if pass = 1 energy/norm 1
    copy/row 1 3 [nline/2+1] pass           # store mode in history array
    copy/row 2 4 [nline/2+1] pass           # store index profile in history array
    if count = 20 then
        plot/l/r 3 ns=64                   # plot every 10 steps
        count = 0
    endif
macro/end
gaus/c/c 1 1 1e-4                         # inject a gaussian mode of 1 micron
Phase = 0                                  # radius
pass = 0
write/off
macro step/nliney                          # propagate 512 times
write/on
plot/watch @Name_1.plt
plot/l 3 ns=64

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch @Name_2.plt
title mode shape vs. distance, real values
plot/l/r 3 ns=64
plot/watch @Name_3.plt
title index difference, straight fiber
plot/l 4 ns=64 h=.1
set/density 64
plot/watch @Name_4.plt
title mode shape vs. distance, two close fibers
plot/con 3 ilab=0
plot/watch @Name_5.plt
title final mode shape, straight fiber
plot/l 1 xrad=2e-4 ns=64
plot/watch @Name_6.plt
title index difference, straight fiber
plot/l 2 xrad=2e-4 ns=64 h=.1
energy 1
Phase = -1.96
pass = 0
clear 3 0
gaus/c/c 1 1 1e-4                                # inject a gaussian mode of 1 micron
plot/watch @Name_7.plt
write/off
macro step/nliney                                # propagate 512 times
write/on
title mode shape vs. distance, real values
plot/l/r 3 ns=64

```

Ex86i: Slab waveguide treating both guided and free-space direction

This example illustrates a slab waveguide with calculation of both the guided performance in the y-direction and the free-space performance in the x-direction. We first consider a simplified index profile for the guided mode for the y-direction, as indicated in Table 86.2. This case may be solved analytically, as indicated in Fig. 86.36 and is in near-perfect agreement with the numerical result in Fig. 86.37. A gaussian of radius 16×0.5 micron is injected into the waveguide. In the guided direction (y-direction), the guided mode is formed during propagation and quickly achieves a stable width.

Table. 86.2. Simplified waveguide layers in y-direction.

layer	depth (microns)	Index
top	unlimited	3.20
middle	0.265	3.30
bottom	unlimited	3.20

Input: ex86i.inp

```

c## ex86i
c
c Example: slab waveguide
c
alias Name ex86i
set/cpu 2

```

Jump to: [Commands](#), [Theory](#)

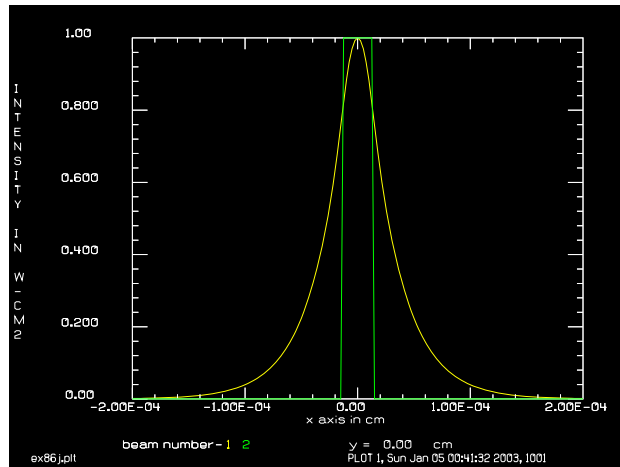


Fig. 86.36. Analytical calculation of mode for simplified slab waveguide.

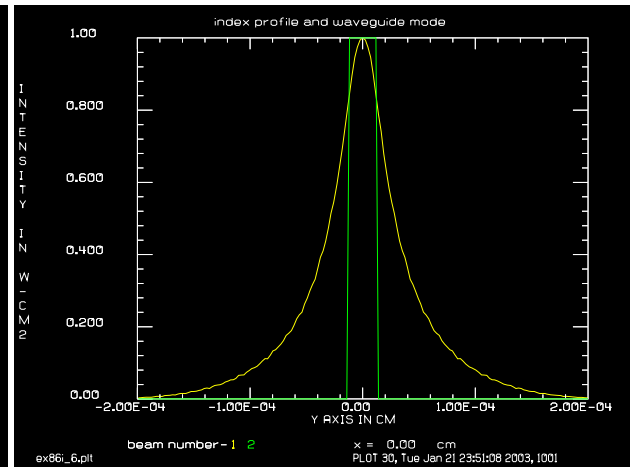


Fig. 86.37. Numerical calculation of mode for simplified slab waveguide from Ex86i. Compare with analytical calculation of Fig. 86.36.

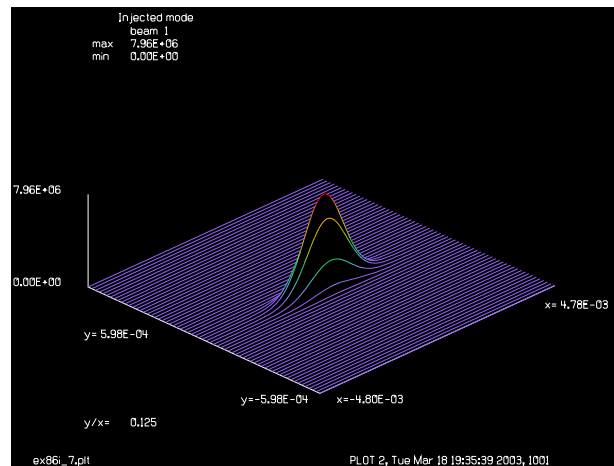


Fig. 86.38. Injected mode is 12μ radius in x-direction and 0.5μ radius in the y-direction.

```

variab/dec/int pass count count1 NlineX NlineY Ntimes    # define variable names
NlineX = 512
NlineY = 512
Ntimes = 128
Dist = .5e-5                                     # step length
FieldX = .0048
FieldY = .0006
UnitsX = FieldX/NlineX*2
UnitsY = FieldY/NlineY*2

array/s 1 NlineX NlineY                          # beam 1, propagating beam
nbeam 4 data                                       # beam 2, index distribution
nbeam 5 NlineY Ntimes data                        # history array
nbeam 6 1 NlineY
nbeam 7 NlineX Ntimes data                        # history array
clear 5 0
clear 7 0

```

Jump to: [Commands](#), [Theory](#)

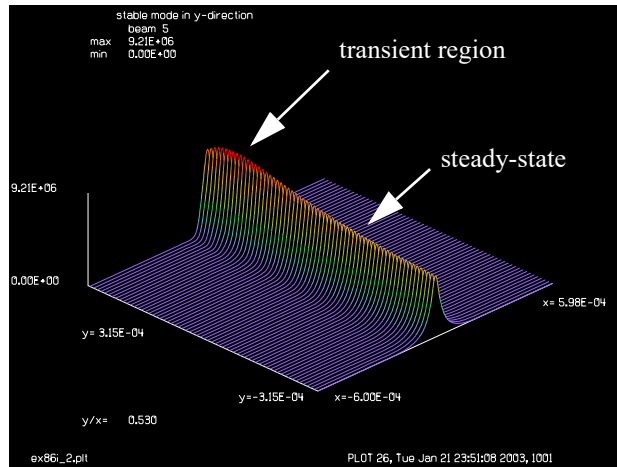


Fig. 86.39. Mode shape vs. length for the guided mode (y-direction).

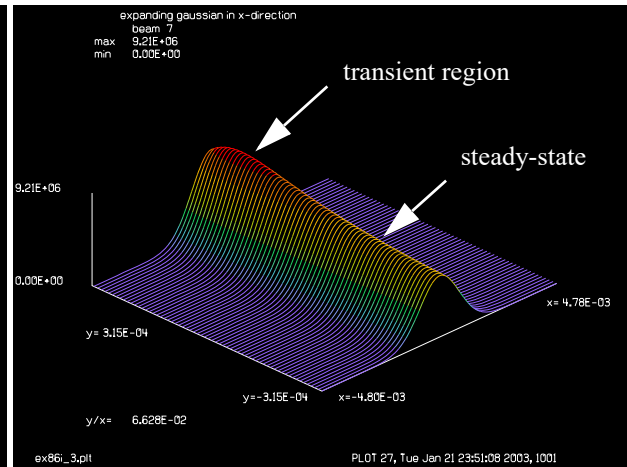


Fig. 86.40. Mode shape vs. length for the free-space mode (x-direction) for a $12\ \mu$ radius starting beam so the width is nearly constant.

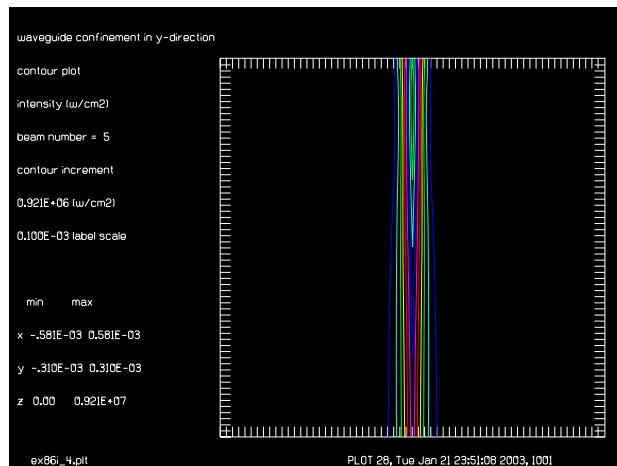


Fig. 86.41. Contour plot guided mode indicating constant width.

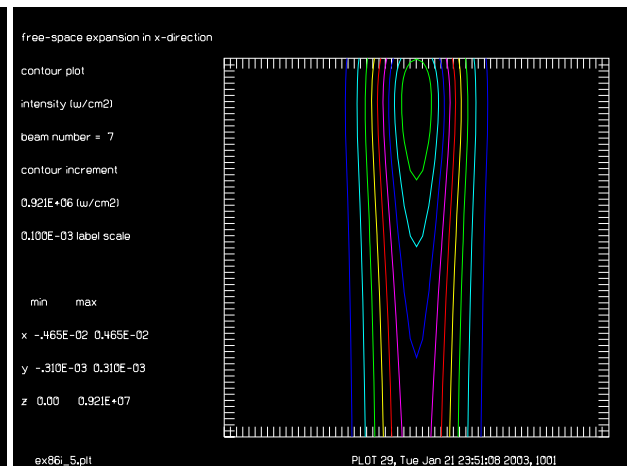


Fig. 86.42. Contour plot free-space mode indicating constant width for a $12\ \mu$ radius starting beam.

```
units/s 1 UnitsX UnitsY
units/s 2 UnitsX UnitsY
units/s 3 UnitsX UnitsY
units/s 4 UnitsX UnitsY
units/s 5 UnitsY Dist
units/s 6 UnitsY Dist
units/s 7 UnitsX Dist
Lambda = 1.3e-4
wavelength/set 0 Lambda*1e4 3.2
alpha = 2.*pi*Dist/Lambda
clear 3 0.
clear 4 .1
clap/r/c 4 1. .265e-4/2.
add/inc/con 3 4
plot/w @Name_1.plt
```

```
# set units
# set units
# set units
# set units
# set units
```

```
# set wavelength and cladding index
# phase coefficient per step
```

```

plot/y/i 3
int2phas/two 2 3 alpha          # implement index in beam 2 on beam 1
split/rec/in 2 FieldX*.95 FieldY*.95 24 24      # absorbing boundary for the
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0
    geodata/set/waist 1 waistx=1e-4 waisty=1e-4
    mult/beam 1 2
    dist Dist 1                  # propagation
    if pass = 1 energy/norm 1
c
c save center column in beam 5
c
    array/s 6 1 NlineY
    copy/col 1 6 NlineY/2+1 1
    transpose 6
    copy/row 6 5 1 pass          # store in history array 5
c
c save center row in beam 7
c
    copy/row 1 7 NlineX/2+1 pass  # store in history array 7
    if count = 10 then
        plot/w @Name_2.plt
        plot/l 5 ns=64
        plot/w @Name_3.plt
        plot/l 7 ns=64
        count = 0
    endif
macro/end
gaus/r/c 1 1 16e-4 .5e-4        # inject a gaussian mode of 1 micron
energy/norm 1 1
plot/w @Name_7.plt
title Injected mode
plot/l 1 1 ns=64
pass = 0
write/off
macro step/Ntimes              # propagate Ntimes
write/on
plot/w @Name_2.plt
title stable mode in y-direction
plot/l 5 ns=64
plot/w @Name_3.plt
title expanding gaussian in x-direction
plot/l 7 ns=64
plot/w @Name_4.plt
title waveguide confinement in y-direction
plot/c 5 ilab=0
plot/w @Name_5.plt
title free-space expansion in x-direction
plot/c 7 ilab=0
copy 3 2
peak/norm 2 1
energy 1

```

Jump to: [Commands](#), [Theory](#)


```

peak/norm 1 1
plot/w @Name_6.plt
title index profile and waveguide mode
plot/y/i fi=1 la=2 le=-2e-4 ri=2e-4

```

Ex86j: Analytical calculation of slab waveguide eigenmode

Calculation of the ideal eigenmode using slab/eigenmode. See Fig. 86.36.

Input: ex86j.inp

```

c## ex86j
alias Name ex86j
echo/on
array/s 1 512 1
nbeam 2
a = .265e-4/2
units/s 0 .25e-5
wavelength/set 0 1.3
slab/eigen/mode 1 a 3.3 3.2 3.2
clap/c/c 2 a
plot/w @Name.plt
plot/x/i fi=1 la=2 le=-2e-4 ri=2e-4

```

Ex86k: Slab waveguide treating both guided and free-space direction

This example illustrates a slab waveguide with calculation of both the guided performance in the y-direction and the free-space performance in the x-direction. A complex index profile is used for the guided mode for the y-direction, as indicated in Table 86.3. The added thin layers above and below the middle layer do not change the guided mode significantly.

Table. 86.3. Complex waveguide layers in y-direction.

layer	depth (microns)	Index
top	unlimited	3.20
top, middle	0.15	3.24
middle	0.265	3.30
bottom, middle	0.05	3.24
bottom	unlimited	3.20

Input: ex86k.inp

```

c## ex86k
c
c Example: slab waveguide
c
alias Name ex86k
set/cpu 2
variab/dec/int pass count count1 NlineX NlineY Ntimes # define variable names
NlineX = 512

```

Jump to: [Commands](#), [Theory](#)

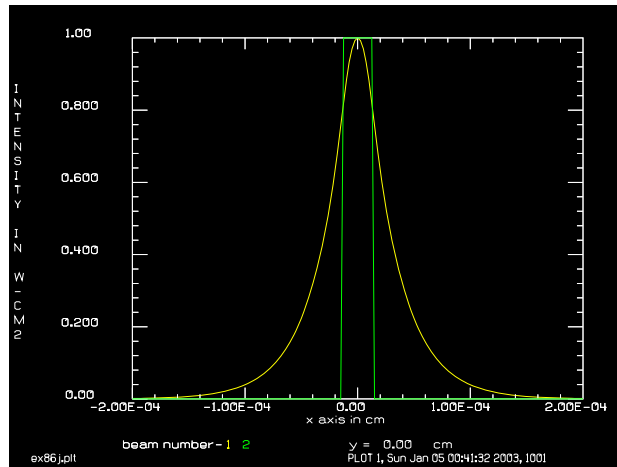


Fig. 86.43. Analytical calculation of mode for simplified slab waveguide.

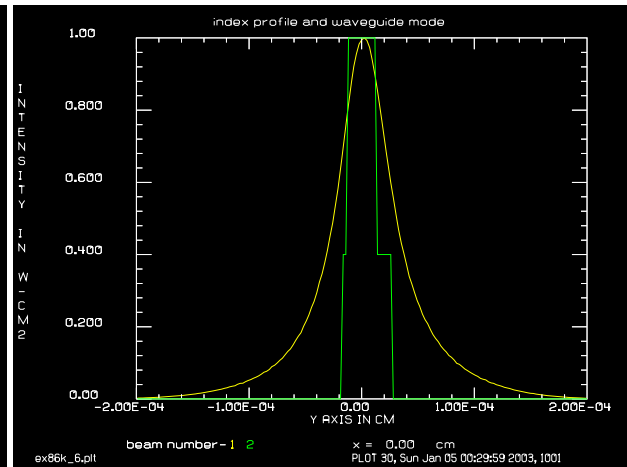


Fig. 86.44. Numerical calculation of mode for simplified slab waveguide from Ex86k. Compare with analytical calculation of Fig. 86.43.

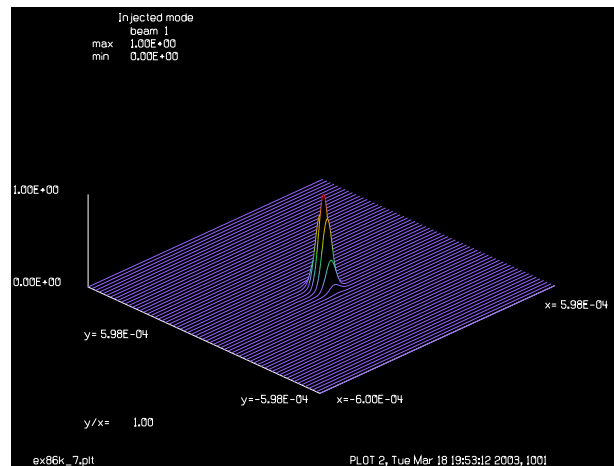


Fig. 86.45. Injected mode is 12μ radius in x-direction and 0.5μ radius in the y-direction.

```

NlineY = 512
Ntimes = 128
Dist = .5e-5
FieldX = .0006
FieldY = .0006
UnitsX = FieldX/NlineX*2
UnitsY = FieldY/NlineY*2

array/s 1 NlineX NlineY
nbeam 4 data
nbeam 5 NlineY Ntimes data
nbeam 6 1 NlineY
nbeam 7 NlineX Ntimes data
clear 5 0
clear 7 0
units/s 1 UnitsX UnitsY
units/s 2 UnitsX UnitsY

# step length
# beam 1, propagating beam
# beam 2, index distribution
# history array
# history array
# set units
# set units

```

Jump to: [Commands](#), [Theory](#)

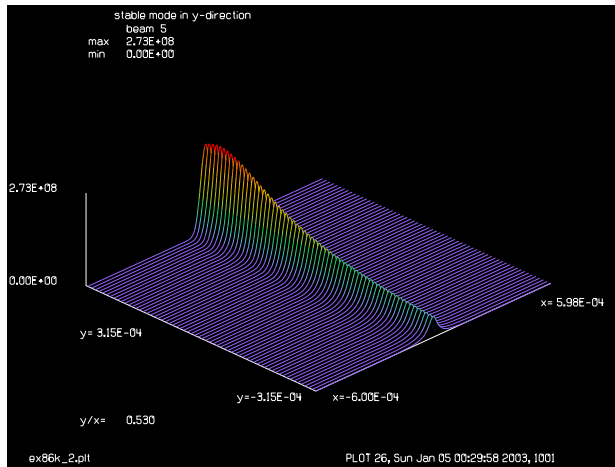


Fig. 86.46. Mode shape vs. length for the guided mode (y-direction). Decay is due to gaussian spreading in the x-direction.

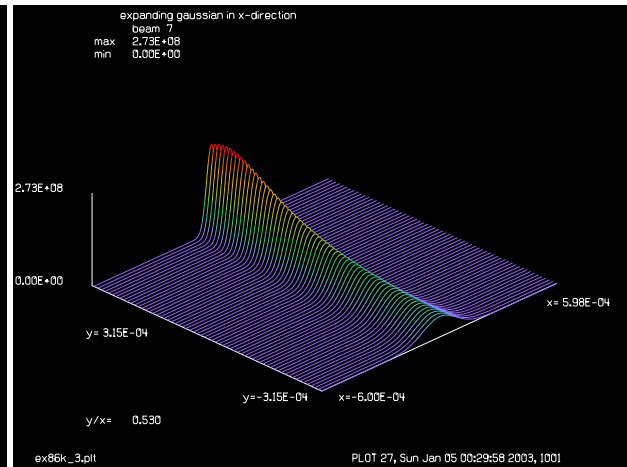


Fig. 86.47. Mode shape vs. length for the free-space mode (x-direction), showing the spread of the injected gaussian beam.

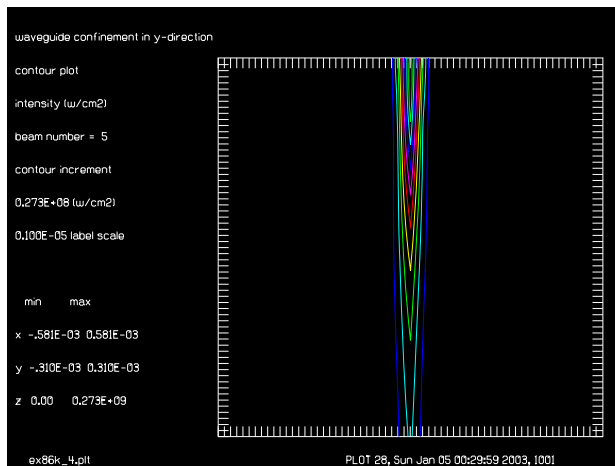


Fig. 86.48. Contour plot of guided mode indicating constant width and decaying peak.

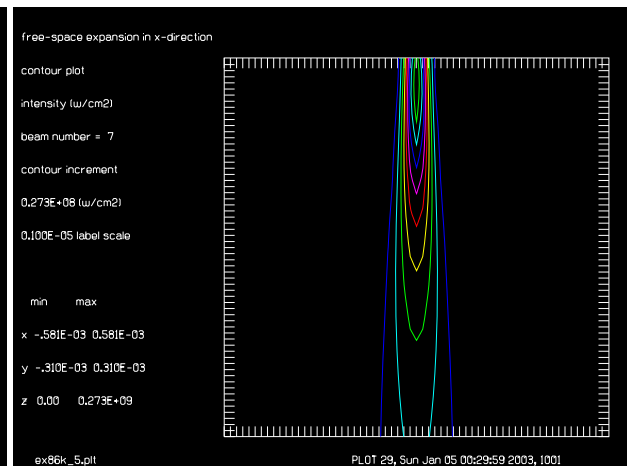


Fig. 86.49. Contour plot of free-space mode indicating spreading width and decaying peak.

```
units/s 3 UnitsX UnitsY
units/s 4 UnitsX UnitsY
units/s 5 UnitsY Dist
units/s 6 UnitsY Dist
units/s 7 UnitsX Dist
Lambda = 1.3e-4
wavelength/set 0 Lambda*1e4 3.2
clear 3 .04
alpha = 2.*pi*Dist/Lambda
clap/r/c 3 1. .233e-4 ydec=.05e-4
clear 4 .06
clap/r/c 4 1. .265e-4/2.
add/inc/con 3 4
plot/w @Name_1.plt
plot/y/i 3
```

Jump to: [Commands](#), [Theory](#)

```
# set units
# set units
# set units
```

```
# set wavelength and cladding index
# set index difference for core
# phase coefficient per step
```

```

int2phas/two 2 3 alpha                # implement index in beam 2 on beam 1
split/rec/in 2 FieldX*.95 FieldY*.95 24 24    # absorbing boundary for the
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0
    geodata/set/waist 1 waistx=1e-4 waisty=1e-4
    mult/beam 1 2
    dist Dist 1                        # propagation
    if pass = 1 energy/norm 1
c
c save center column in beam 5
c
    array/s 6 1 NlineY
    copy/col 1 6 NlineY/2+1 1
    transpose 6
    copy/row 6 5 1 pass                # store in history array 5
c
c save center row in beam 7
c
    copy/row 1 7 NlineX/2+1 pass        # store in history array 7
    if count = 10 then
        plot/w @Name_2.plt
        plot/l 5 ns=64
        plot/w @Name_3.plt
        plot/l 7 ns=64
        count = 0
    endif
macro/end
gaus/r/c 1 1 .5e-4 .5e-4                # inject a gaussian mode of 1 micron
plot/w @Name_7.plt
title Injected mode
plot/l 1 ns=64
pass = 0
write/off
macro step/Ntimes                        # propagate Ntimes
write/on
plot/w @Name_2.plt
title stable mode in y-direction
plot/l 5 ns=64
plot/w @Name_3.plt
title expanding gaussian in x-direction
plot/l 7 ns=64
plot/w @Name_4.plt
title waveguide confinement in y-direction
plot/c 5 ilab=0
plot/w @Name_5.plt
title free-space expansion in x-direction
plot/c 7 ilab=0
copy 3 2
peak/norm 2 1
peak/norm 1 1
plot/w @Name_6.plt
title index profile and waveguide mode

```

Jump to: [Commands](#), [Theory](#)

```
plot/y/i fi=1 la=2 le=-2e-4 ri=2e-4
```

Ex86l: Comparison of BPM mode and gaussian approximation

A gaussian approximation is frequently used to determine the optical mode of a step index fiber. The formula is expressed in terms of the normalized frequency

$$v^2 = a^2 k^2 (n_2^2 - n_1^2) \quad (86.2)$$

where a is the fiber radius, k is the wave number, n_2 is the index of the core, and n_1 is index of the cladding. the gaussian expression for an equivalent gaussian beam is

$$\omega_0 = a(0.65 + 1.619v^{-1.5} + 2.879v^{-6}) \quad (86.3)$$

For a given value of v , the radius may be found:

$$a = \frac{v}{2\pi} \frac{\lambda}{\sqrt{n_2^2 - n_1^2}} \quad (86.4)$$

In this example values of $v = 4.0, 3.0, 2.4$, and 1.2 . The gaussian approximation agrees best with the BPM analysis for a “normal” value of normalized frequency, $v = 2.4$.

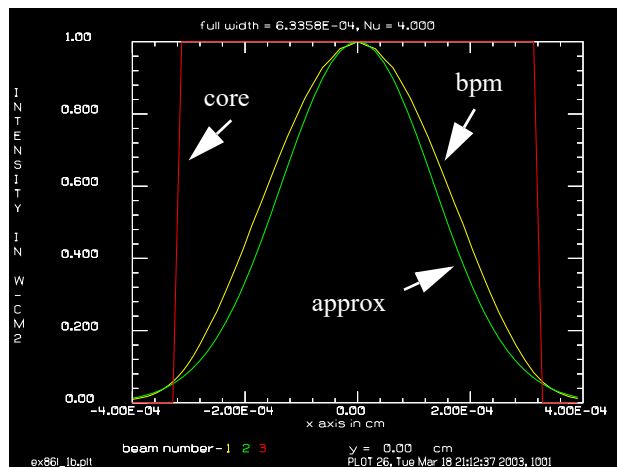


Fig. 86.50. $v = 4.0$, well confined, width = 6.32μ .

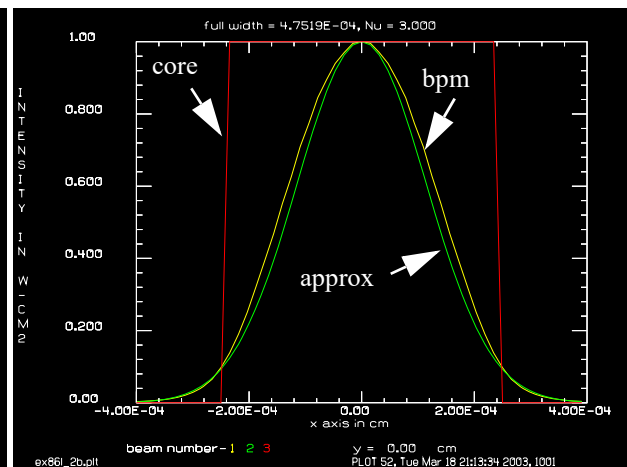
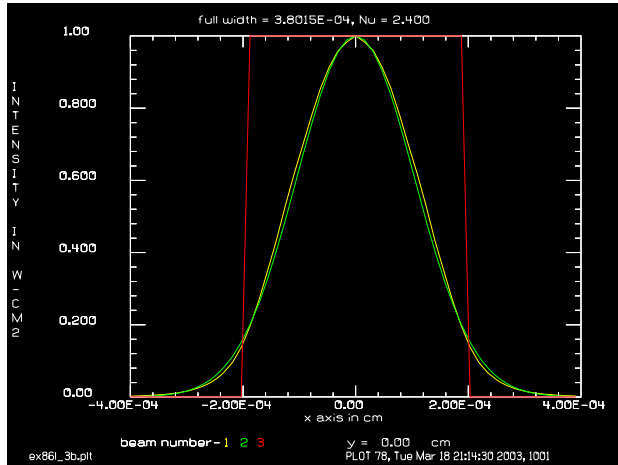
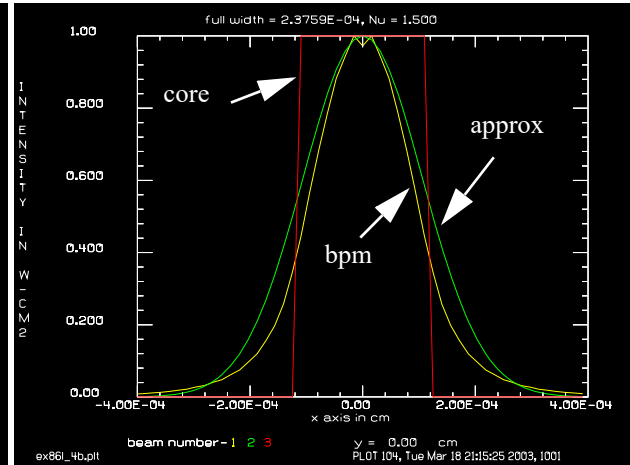
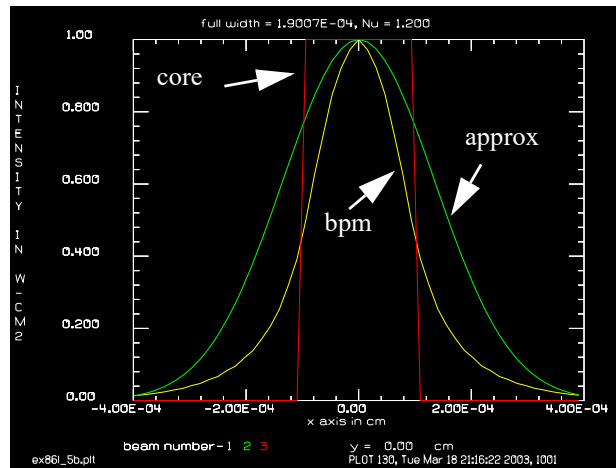


Fig. 86.51. $v = 3.0$, well confined, width = 4.75μ .

Input: ex86l.inp

```
c## ex86l
c
c Example: Comparison of BPM and approximate gaussian modes
c
c echo/on
```

Jump to: [Commands](#), [Theory](#)

Fig. 86.52. $\nu = 2.4$, good agreement, width = 3.8μ .Fig. 86.53. $\nu = 1.5$, separation of BPM and gaussian, width = 2.4μ .Fig. 86.54. $\nu = 1.2$, strong separation of BPM and gaussian, width = 1.9μ .

```

variab/dec/int i pass count nline nline2 # define variable names
dist = 2e-5                               # step length
nline = 256                               # propagating array width
title mode shape vs. distance, straight fiber
array/s 1 nline                           # beam 1, propagating beam
nbeam 2 data                               # beam 2, index distribution
nline2 = 256
nbeam 3 nline nline2 data                 # beam 3, history of mode shape
nbeam 4 nline nline2 data                 # beam 4, history of index profile
nbeam 5 nline 1 data
Field = 20e-4
units/field 0 Field                       # field half-width of 8 microns
variab/set units 3 units
units/set 3 units dist                    # set units of history arrays
units/set 4 units dist
c Lambda = .6328e-4
Lambda = 1.55e-4
wavelength/set 0 Lambda*1e4 1.5          # set wavelength and cladding index

```

Jump to: [Commands](#), [Theory](#)

```

clear 1 1
clear 3 0
clear 4 0
alpha = 2.*pi*dist/Lambda          # phase coefficient per step
x1 = 4
x2 = 3
x3 = 2.4
x4 = 1.5
x5 = 1.2
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0
    geodata/set 1 0 0 1e-4 1e-4 1 1
    clear 2 .032                    # set index difference for core
    clap/c/c 2 Apt
    int2phas/two 1 2 alpha          # implement index in beam 2 on beam 1
    split/cir/in 1 Field 6          # absorbing boundary for the array
    dist dist 1                     # propagation
    if pass = 1 energy/norm 1
    copy/row 1 3 [nline/2+1] pass    # store mode in history array
c   copy/row 1 5 [nline/2+1] 1      # store mode in history array
c   amplitude/abs 5
c   copy/row 5 3 1 pass
    copy/row 2 4 [nline/2+1] pass    # store index profile in history array
    if count = 10 then
        plot/l/i 3 ns=64            # plot every 10 steps
        count = 0
    endif
macro/end
macro/def outer/o
    i = i + 1
c   v = 2.*pi/Lambda*Apt*sqrt(1.532^2 - 1.50^2)
    v = x@i
    Apt = v/2./pi*Lambda/sqrt(1.532^2 - 1.50^2)
    Apt=
    w0 = Apt*(.65 + 1.619*v^-1.5 + 2.879*v^-6)
    array/set 3 nline nline2 data    # beam 3, history of mode shape
    units/field 3 Field              # field half-width of 8 microns
    clear 3 0
    clear 4 0
    gaus/c/c 1 1 w0                  # inject a gaussian mode of 1 micron
    zreff/se 1 0.
c   clap/c/c 1 Apt                    # radius
    pass = 0
    count = 0
    plot/watch ex86l_@ia.plt
    write/off
    macro step/nline2                # propagate nline2 times
    write/on
c   amp/abs 1
    peak/norm 1 1
    gaus/c/c 2 1 w0
    array/set 3 128 128

```

```

units/beam 3 1
clap/c/n 3 Apt
FullWidth=2*Apt
title full width = @FullWidth, Nu = @v
plot/watch ex86l_@ib.plt
plot/x/i fi=1 la=3 le=-.0004 ri=.0004
fitgeo
macro/end
i = 0
macro outer/5

```

Ex86m: Comparision of BPM and gaussian at critical frequency

This example shows an expanded view of the case $\nu = 1.2$.

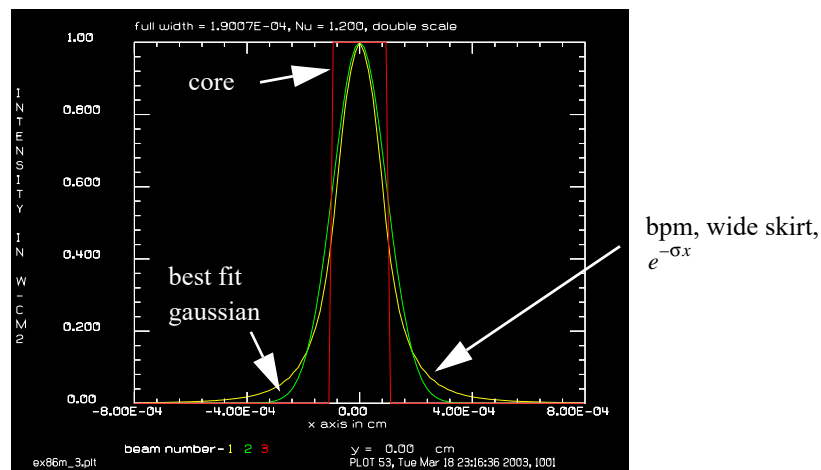


Fig. 86.55. $\nu = 1.2$, at critical frequency, good match, width = 1.9μ ., double scale, with "best fit" gaussian. Note the true curve calculated by BPM method has a much wider skirt. The true curve is close to the form $\exp(-\sigma x)$ outside the core and decays slower than the gaussian.

Input: `ex86m.inp`

Ex86n: Fiber with long radius bend

This example illustrates techniques for calculating the loss for long radius bends using the BPM method.

Input: `ex86n.inp`

```

c## ex86n
c
c Example 86n: Fiber with long radius bend
c
c Determine loss due to fiber bend of radius, Radius
c Let the fiber stablize and then calculate energy loss
c in remaining Ntest steps.
c
c Radius of about 1 cm and shorter gives a noticeable loss

```

Jump to: [Commands](#), [Theory](#)


```

c   relative to infinite radius.
c
c   Run once with Radius set to a very large value. Repeat with
c   shorter radius and compute extra loss due to bend.
c
set/highlight/off
variab/dec/int Pass Count Nline Ntest Ntimes # define variable names
variab/dec/int Limit
Limit = 10
Dist = 1e-5 # step length
Nline = 128 # propagating array width
Ntimes = 1024 # total number of steps
title mode shape vs. distance, straight fiber
plot/watch ex86a_1.plt
array/s 1 Nline # beam 1, propagating beam
nbeam 2 data # beam 2, index distribution
nbeam 3 Nline Ntimes data # beam 3, history of mode shape
nbeam 4 Nline Ntimes data # beam 4, history of index profile
units/field 0 6e-4 # field half-width of 8 microns
variab/set Units 3 units
units/set 3 Units Limit*Dist # set units of history arrays
units/set 4 Units Limit*Dist
Lambda = 0.6328e-4
BulkIndex = 1.5
wavelength/set 0 Lambda*1e4 BulkIndex # set wavelength and cladding index
Radius = .1 # radius of fiber bend
Ntest = 400 # number of steps to test loss
clear 1 1
clear 3 0
clear 4 0
Alpha = 2.*pi*Dist/.6328e-4 # phase coefficient per step
macro/def step/o
    Pass = Pass+1
    Count = Count+1
    zreff/se 1 0
    geodata/set/waist waistx=1e-4 waisty=1e-4
    clear 2 .032 # set index difference for core
    clap/c/n 2 .8e-4
    int2phas/two 1 2 Alpha # implement index in beam 2 on beam 1
    abr/tilt 1 Dist*BulkIndex/Lambda rnorm=Radius az=90 # tilt for fiber bend ra
    split/cir/in 1 5e-4 6 # absorbing boundary for the array
    dist Dist 1 # propagation
    if pass = 1 energy/norm 1
    copy/row 1 3 [Nline/2+1] Pass # store mode in history array
    copy/row 2 4 [Nline/2+1] Pass # store index profile in history array
    if [Count==Limit] then
        plot/1 3 ns=64 # plot every 10 steps
        Count = 0
    endif
macro/end
gaus/c/c 1 1 1e-4 # inject a gaussian mode of 1 micron
# radius

pass = 0
write/off

```

Jump to: [Commands](#), [Theory](#)

```

macro step/[Ntimes-Ntest]                # propagate 512 times
variable/set Energy1 1 energy
macro step/Ntest
write/on
variable/set Energy2 1 energy
plot/watch ex86a_1.plt
plot/1 3 ns=64
plot/watch ex86a_2.plt
title index difference, straight fiber
plot/1 4 ns=64 h=.1
set/density 64
plot/watch ex86a_3.plt
title mode shape vs. distance, straight fiber
plot/con 3 ilab=0
plot/watch ex86a_4.plt
title final mode shape, straight fiber
plot/1 1 xrad=2e-4 ns=64
plot/watch ex86a_5.plt
title index difference, straight fiber
plot/1 2 xrad=2e-4 ns=64 h=.1
RelativeEnergyLoss = (Energy1-Energy2)/Energy1/Ntest list
RelativeLossPerCm = RelativeEnergyLoss/Dist list

```

Ex86o: A pencil of light injected into a dielectric waveguide at near-TIR angle.

This example illustrates a pencil of light injected into a dielectric waveguide at near-TIR angle. A small pencil of light provides a very valuable way to investigate the accuracy of the analysis. First we test what happens if we just send the pencil toward the left side of the array without a high index core and without an absorbing boundary. Figures 86.56 and 86.57 illustrate the aliased behavior that exists if an absorbing boundary is not used. It is necessary to suppress light near the edges of the array to prevent aliasing errors.

Figure 86.58 a rolled off transmission function that forms an absorption boundary. Figures 86.59 and 86.60 show that the pencil of light is suppressed at the absorbing boundary.

Figure 86.61 shows an intermediate point at pass 3,800. Reflections R1 and R2 are shown as well as the transmitted and refracted beams T1 and T2. Figure 86.62 the intermediate point at pass 5,600. Reflections R1, R2 and R3 are shown as well as associated transmitted beams T1, T2, and T3. The strong transmitted beam T1, for near-grazing exiting ray, appears to be interfering with reflection R2.

Figure 86.63 shows the distribution at the final point of the calculation which is at the start of reflection R5. The display covers 125% \times 125% of the waveguide width showing that substantial light is outside the waveguide. This figure also shows the standing wave formed at the reflection point. Figures 86.64 and 86.65 show the history of the beam progress through the waveguide for 8,192 passes. Figures 86.66 shows the far field distribution. As the calculation ends approximately at R5, both the internal left and right reflecting beams are present as well as a combination of all transmitted beams.

Input: ex86o.inp

```

c## Ex86o
c
c Example: Pencil of light hitting high-to-low index boundary

```

Jump to: [Commands](#), [Theory](#)

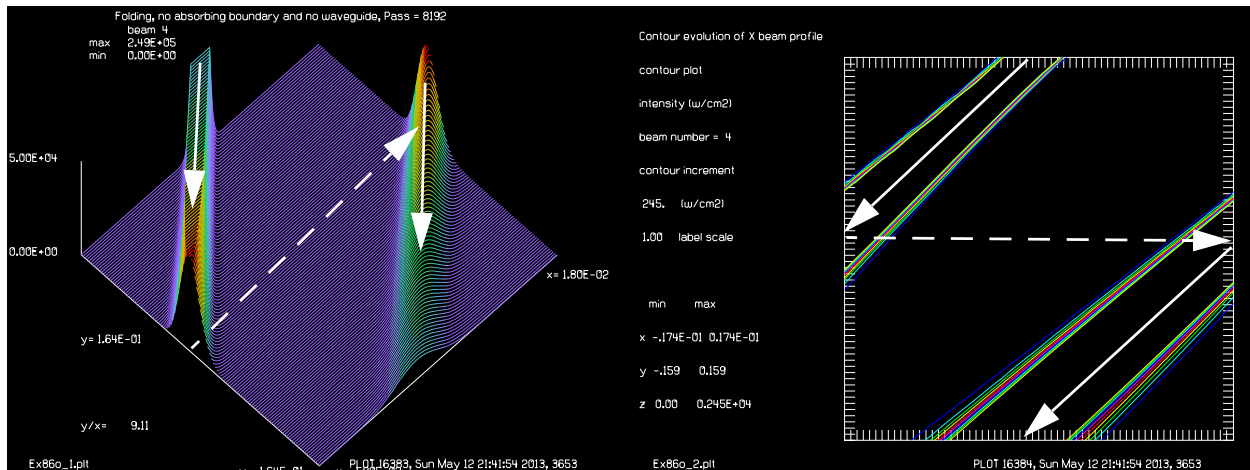


Fig. 86.56. A pencil of light is injected at an angle toward the left wall. In this case there is no high index core and no absorbing boundary. The pencil exits at the left wall and reappears at the right wall at the same angle. Stray light reaching the edge of the array will cause an error in the calculation if not absorbed.

Fig. 86.57. A contour plot for the case shown in Fig. 86.61. The display shows detail below 1% of the peak to give an outline of the expanding beam.

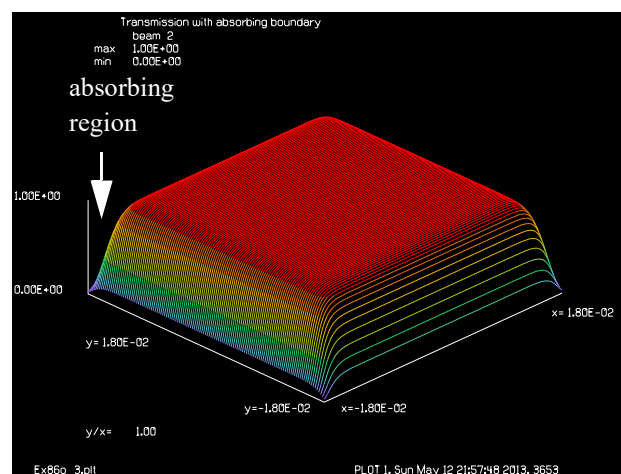


Fig. 86.58. The absorbing boundaries appear as a rolloff in the transmission distribution of the waveguide. The absorption boundary is intended to absorb light which is propagating well away from the high index core region.

```

c          showing refracted and reflecting beams for four reflections.
c
alias Name Ex86o
set/cpu 4

variab/dec/int Pass SWITCH NlineX NlineY Ntimes      # define variable names

c SWITCH = 1, just test light hitting boundary and coming in from other side
c SWITCH = 2, test absorbing boundary without dielectric waveguide
c SWITCH = 3, both absorbing boundary and dielectric waveguide
SWITCH = 3

```

Jump to: [Commands](#), [Theory](#)

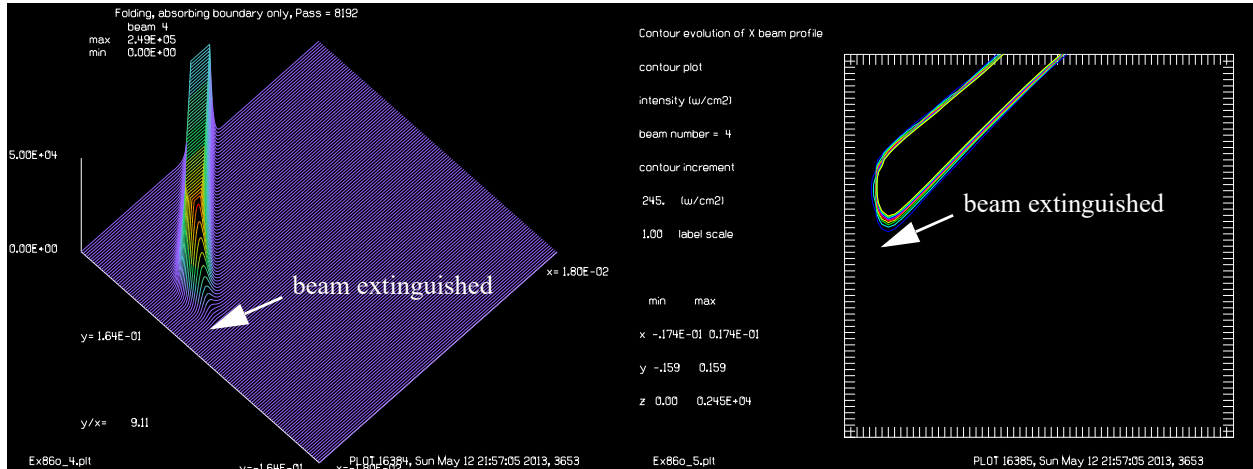


Fig. 86.59. A pencil headed into the left wall is extinguished by the absorbing boundary illustrated in Fig. 86.58. Compare with Fig. 86.56. The aliased, wrapped beam is extinguished.

Fig. 86.60. A contour plot showing the pencil beam has been extinguished by the absorbing boundary. Compare with Fig. 86.57.

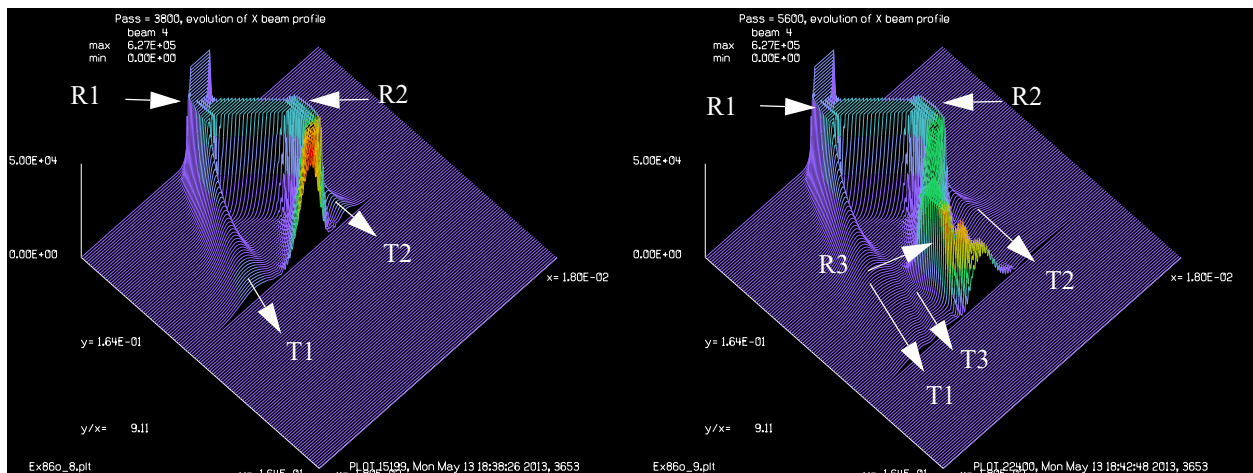


Fig. 86.61. A pencil of light, just slightly steeper than the TIR angle, is injected into a dielectric waveguide. It is initially headed to the left and making bounces at points R1 and R2. The display was made at step 3,800. Beams marked T1 and T2 are the transmitted beams which are refracted at a near-grazing angle.

Fig. 86.62. A pencil of light, just slightly steeper than the TIR angle, is injected into a dielectric waveguide. It is initially headed to the left and making bounces at points R1, R2, and R3. The display was made at step 5,600. Beams marked T1, T2, and T3 are the transmitted beams which are refracted at a near-grazing angle.

```
NlineX = 1024
NlineY = NlineX
Ntimes = 8192
propagated in z
```

```
step = 0.4e-4
FieldX = 180e-4
FieldY = FieldX
UnitsX = 2*FieldX/NlineX
```

```
# number of sections field is to be divided in x
# number of sections field is to be divided in y
# number of steps field is to be
# step length
# 1/2 width of field in x [cm]
# 1/2 width of field in y [cm]
# length of each section in x [cm]
```

Jump to: [Commands](#), [Theory](#)

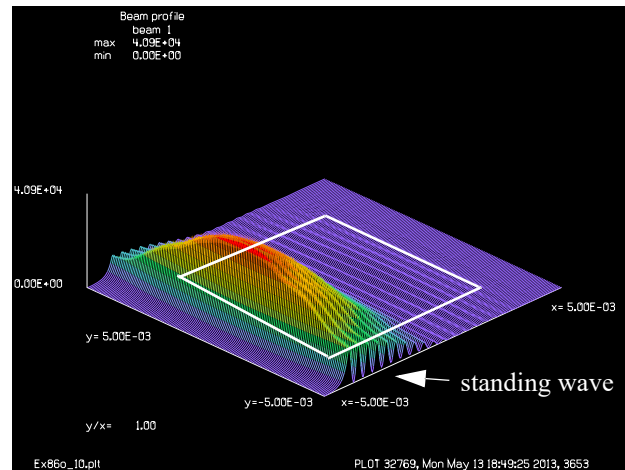


Fig. 86.63. A square section of the beam distribution at 125% \times 125% of the waveguide width, shown by the white square, at the end of the calculation at about the point of the fifth reflection R5. The standing wave implies that both reflected and transmitted beams are present at this point as shown in the far field in Fig. 86.66.

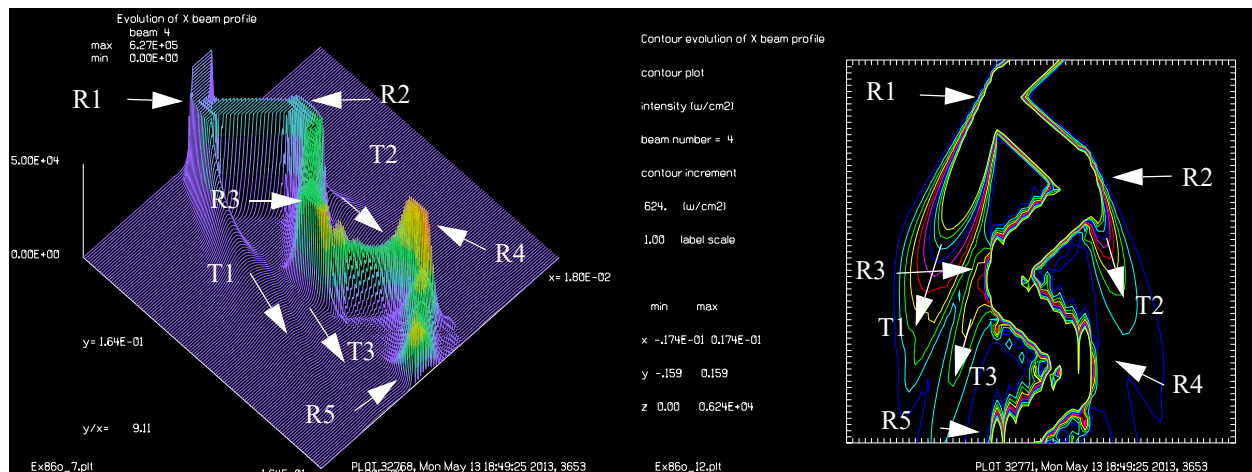


Fig. 86.64. A pencil of light is injected into a dielectric waveguide, initially headed to the left and making bounces at points R1 through R5. Reflection R3 appears to be distorted by the presence of the strong transmitted beam T1. The internal mode follows a zigzag path. Light transmitted through the boundaries is strongly refracted in accord with near-TIR condition.

Fig. 86.65. Countour plot saturated to 1% of peak for the same situation as displayed in Fig. 86.64. It appears that the strong transmitted beam from R1 is affecting the reflection behaviour at R3.

```

UnitsY = 2*FieldY/NlineX      # length of each section in y [cm]
UnitsZ = step                  # length of step in z [cm]
array/s 1 NlineX              # array dimensions of propagating beam
and index beams
nbeam 2 NlineX data           # Index beam
nbeam 3 NlineX NlineY data    # beam 3, complete index distribution
nbeam 4 NlineX Ntimes data    # array dimensions of storage beams

units/s 1:3 UnitsX            # Dimension of each pixel in X and Y [cm]
units/s 4 UnitsX UnitsZ

```

Jump to: [Commands](#), [Theory](#)

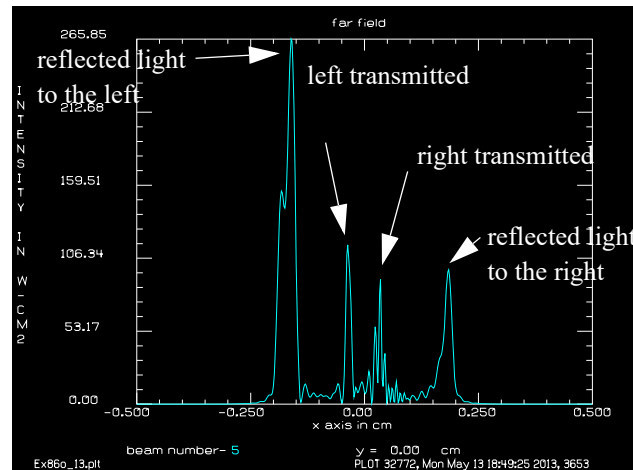


Fig. 86.66. A profile of the far field from the finishing condition which is a reflection R5 shown in Fig. 86.64 and Fig. 86.65. Because we are in mid-reflection both the left and right reflections are shown with substantial magnitudes.

```

Lambda = 1.4e-4                                # wavelength of light [microns]
wavelength/set 0 Lambda*1e4 1.5                # sets wavelength of light to be lambda and
refractive index of                             # all beams to be n_0
CoreWidthX = 40e-4                             # 1/2 Core width in x [cm]

clear 3 0                                       # set base refractive index to 0 (N_0 offset
defined by wavelength definition)
clear 4 0

xborder = FieldX
yborder = xborder list

if SWITCH!=1 then
  c
  c Put absorbing boundary at edge of array
  c
  split/rec/in 2 xborder yborder 10 10        # define border where field absorbs
endif
if SWITCH=2 then
  plot/w @Name_3.plt
  title Transmission with absorbing boundary
  plot/1 2 ns=128
endif
if SWITCH=3 then
  c
  c Implement higher index core
  c
  DeltaIndex = .012
  clear 3 DeltaIndex                          # set inner clad refractive index
difference between N_clad_1 and base
  clap/r/c 3 CoreWidthX 2*CoreWidthX          # set rectangular core aperture of half
widths
  alpha = 2.*pi*step/Lambda                   # phase coefficient per step

```

Jump to: [Commands](#), [Theory](#)

```

    int2phas/two 2 3 alpha          # create phase screen based on refractive index
profile
    plot/w @Name_6.plt
    title Refractive index difference profile
    plot/l 3 ns=128
endif

macro/def step/o                   # define macro for propagating beam
    Pass = Pass+1
    zreff/set 1 0                  # initailizes z reference to 0
    geodata/set/waist 1 CoreWidthX # define surrogate gaussian beam
    mult/beam 1 2                  # multiplies beam 1 by phase screen
    prop step 1                    # propagates beam 1 by distance, step
    copy/row 1 4 NlineX/2+1 Pass   # store in history array
    if SWITCH=1 then
        plot/w @Name_1.plt
        title Folding, no absorbing boundary and no waveguide, Pass = @Pass
        plot/l 4 ns=128 max=.5e5 theta=42
        plot/w @Name_2.plt
        title Contour evolution of X beam profile
        plot/c 4 ilab=0 peak=1
    endif
    if SWITCH=2 then
        plot/w @Name_4.plt
        title Folding, absorbing boundary only, Pass = @Pass
        plot/l 4 ns=128 max=.5e5 theta=42
        plot/w @Name_5.plt
        title Contour evolution of X beam profile
        plot/c 4 ilab=0 peak=1
    endif
    if SWITCH=3 then
        plot/w @Name_7.plt
        title Pass = @Pass, Evolution of X beam profile
        plot/l 4 ns=128 max=.5e5 theta=42
        if Pass=3800 then
            plot/w @Name_8.plt
            title Pass = @Pass, evolution of X beam profile
            plot/l 4 ns=128 max=.5e5 theta=42
        endif
        if Pass=5600 then
            plot/w @Name_9.plt
            title Pass = @Pass, evolution of X beam profile
            plot/l 4 ns=128 max=.5e5 theta=42
        endif
        plot/w @Name_10.plt
        title Beam profile
        plot/l 1 1 xrad=CoreWidthX*1.25 ns=128
        plot/w @Name_11.plt
        title Contour evolution of X beam profile
        plot/c 4 ilab=0 peak=1
    endif
macro/end
c

```

Jump to: [Commands](#), [Theory](#)


```

c Start with gaussian beam
c
gaus/c 1 1 16e-4
c
c Start pencil beam at an angle
c
AngleRad = 1.2e-1
AngleDeg = AngleRad*180/pi list
abr/tilt/angle 1 AngleRad az=90
energy/norm 1 1
variable/set Peak 1 peak          # Store peak value
write/off
macro step/Ntimes          # Propagate beam in waveguide
write/on
if SWITCH!=3 read/scr
c
c Set up a new beam at twice the size for far field for good far field sampling.
c
array/set 5 2*NlineX 2*NlineX
wavelength/set 5 Lambda*1e4 1
units/beam 5 1
c
c Set beam 5 to zero before copy
c
clear 5 0
c
c copy Beam 1 to Beam 5
c
copy/con 1 5
c
c Turn of propagation of Beam 1
c
beams/off 1
c
c Make a lens and propagate to focus
c
FocalLength = 1
lens 5 FocalLength
prop FocalLength
plot/w @Name_12.plt
title far field
plot/x/i 5 left=-.5 right=.5

```

Ex86p: A pencil of light injected into a tapered waveguide.

This example illustrates a pencil of light injected into a tapered dielectric waveguide. A small pencil of light provides a very valuable way to investigate the accuracy of the analysis. See the related example Ex86o which models a straight waveguide at near-grazing incidence and explores the requirements of the absorbing boundary. The angle for Ex86p has been made slightly lower than Ex86o.

In Ex86p.inp both a straight waveguide and a 2:1 tapered waveguide are modeled. For SWITCH=1, the core is a constant 0.0040 cm half width. The index vs length is shown in Fig. 86.67 and the BPM results are

Jump to: [Commands](#), [Theory](#)

shown in Fig. 86.69. For this constant width case the transmission loss remains consistently very low for the reflections R1 to R4.

For SWITCH=2, the waveguide tapers down to 0.0020 cm half width over the length of 0.328 cm. For the tapered waveguide shown in Fig. 86.68, the pencil of light has little transmission loss on the first bounce R1. However, the incident angle increases with each bounce off the tapered wall and the transmission loss increases accordingly from R2 to R4.

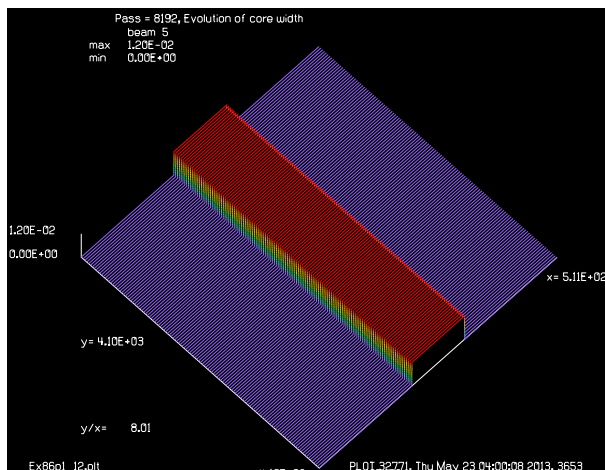


Fig. 86.67. For SWITCH=1, the index profile from start to finish at the length of 0.328 cm is a constant 0.0040 cm half width.

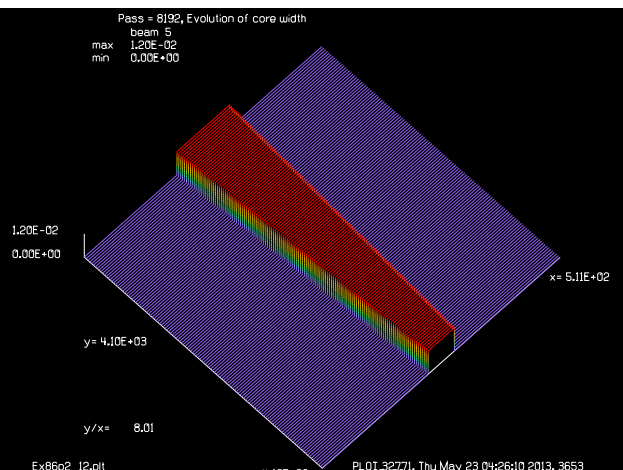


Fig. 86.68. For SWITCH=2, the index profile starts at 0.0040 half width and finishes at 0.0020 cm half width over the length 0.328 cm.

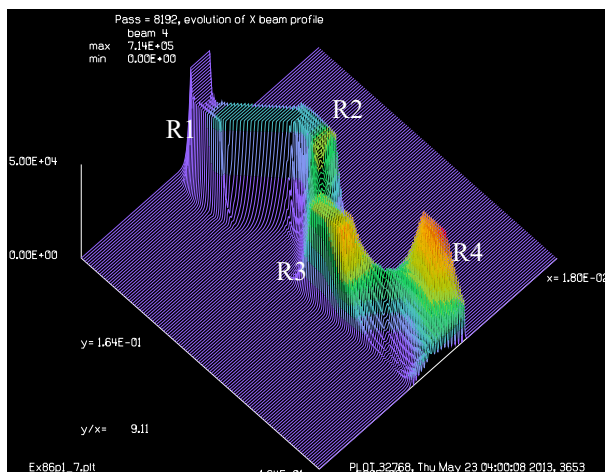


Fig. 86.69. For SWITCH=1 the core has constant width as shown in Fig. 86.67. A pencil of light is injected at an angle toward the left wall. The angle is set to be slightly less than the TIR angle. Leakage is very low through all reflections from R1 to R4. The primary change is due to the expansion of the gaussian beam.

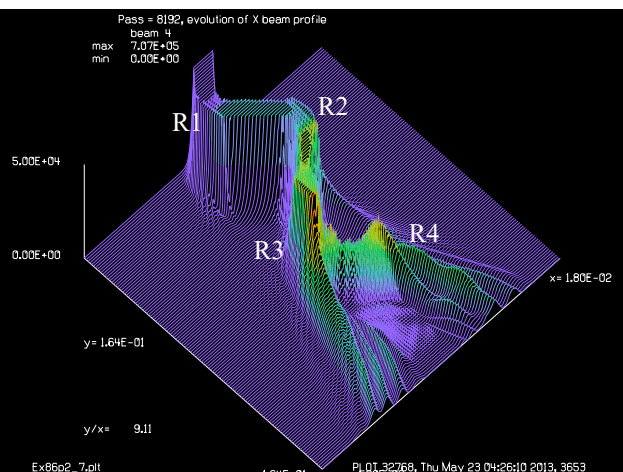


Fig. 86.70. For SWITCH=2 the core tapers from 0.0040 half width to 0.0020 half width as shown in Fig. 86.68. A pencil of light is injected at the same angle as Fig. 86.69—an angle slightly less than the TIR angle of the straight waveguide. There is almost no leakage at the first reflection R1, but with each reflection the angle increases and leakage increases quickly with each reflection from R2 to R4.

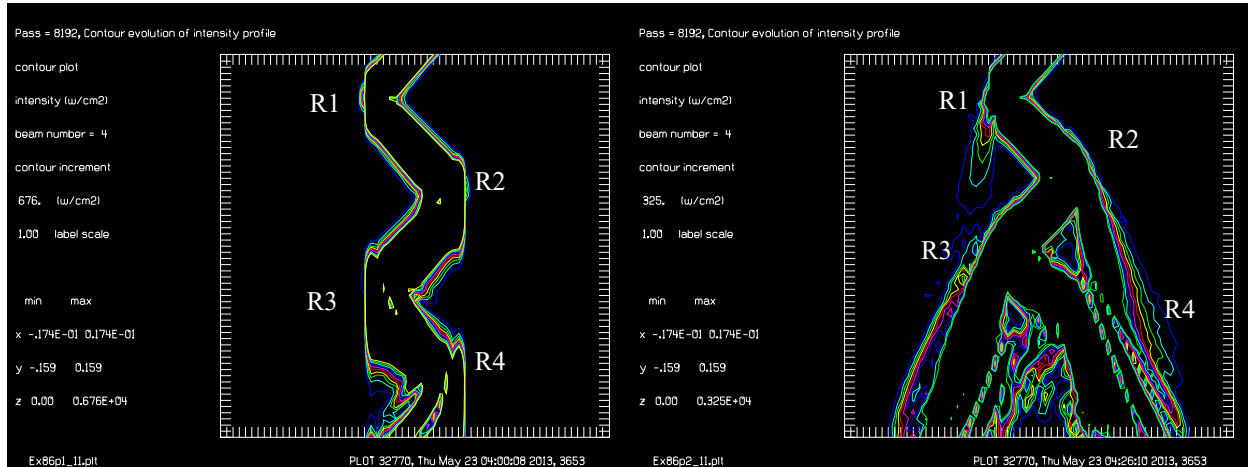


Fig. 86.71. Contour plot saturated to 1% of peak for SWITCH=1 where the core has constant width as shown in Fig. 86.67. A pencil of light is injected at an angle toward the left wall. The angle is set to be slightly less than the TIR angle. Leakage is very low through all reflections from R1 to R4. The primary change is due to the expansion of the gaussian beam.

Fig. 86.72. Contour plot saturated to 1% of peak for SWITCH=2 where the core tapers from 0.0040 half width to 0.0020 half width as shown in Fig. 86.68. A pencil of light is injected at an angle slightly less than the TIR angle for the tapered wall. There is a little leakage at the first reflection R1, but with each reflection from R2 to R4 the angle increases and leakage increases correspondingly.

Input: ex86p.inp

```
c## ex86p
c
c Example 86p: fiber optics propagation, straight fiber with gap
c
c This example is the same as ex86a except for a gap in the center.
c
c This example illustrates propagation in an optical fiber of
c circular cross section. A gaussian beam is injected into the
c fiber. The initial distribution is transformed into the lowest
c loss mode of the fiber after propagation.
c
c The fiber consists of a cladding of index 1.5 and a core of index
c 1.532, giving a difference in index of 0.032. The modeling directly
c treats the phase build-up versus distance. Beam 2 contains the
c pattern of index change associated with the core. The INT2PHAS command
c is used to implement the incremental index difference defined on
c Beam 2 as a phase effect on Beam 1.
c
c It is necessary to have light scattered into the sides of the array
c be absorbed. The SPLIT/CIR/IN command implements a supergaussian
c windowing function to absorb light near the edge of the array.
c
c The optical fiber has a circular cross section, so this is a three-
c dimensional calculation. To display the mode shape, the irradiance
c profile, taken across the diameter of the fiber is displayed at
c each .2 microns of propagation.
```

Jump to: [Commands](#), [Theory](#)

```

c
c A gap is introduced at the middle of the waveguide.
c
alias Name ex86p
variab/dec/int pass count nline GapNum # define variable names
dist = 2e-5 # step length
Gap = 3e-4 # set the gap
GapNum = Gap/dist # number of steps for the gap
nline = 128 # propagating array width
title mode shape vs. distance, straight fiber
plot/watch @Name_1.plt
array/s 1 nline # beam 1, propagating beam
nbeam 2 data # beam 2, index distribution
nbeam 3 nline 512 data # beam 3, history of mode shape
nbeam 4 nline 512 data # beam 4, history of index profile
units/field 0 8e-4 # field half-width of 8 microns
variab/set units 3 units
units/set 3 units dist # set units of history arrays
units/set 4 units dist
Lambda = 1.064e-4
Index = 1.4332
dIndex = 0.068
wavelength/set 0 Lambda*1e4 Index # set wavelength and cladding index
clear 1 1
clear 3 0
clear 4 0
alpha = 2.*pi*dist/.6328e-4 # phase coefficient per step
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0
    geodata/set/waist waistx=1e-4 waisty=1e-4
    if [pass==256-GapNum/2] then
        wavelength/set 0 Lambda*1e4 1. # set to air
        dIndex = 0.
    endif
    if [pass==256+GapNum/2] then
        wavelength/set 0 Lambda*1e4 Index # set wavelength and cladding index
        dIndex = .068 # set index difference for core
    endif
    clear 2 dIndex
    clap/c/n 2 .8e-4
    int2phas/two 1 2 alpha # implement index in beam 2 on beam 1
    split/cir/in 1 6e-4 6 # absorbing boundary for the array
    dist dist 1 # propagation
    if pass = 1 energy/norm 1
    copy/row 1 3 [nline/2+1] pass # store mode in history array
    copy/row 2 4 [nline/2+1] pass # store index profile in history array
    if count = 10 then
        plot/l 3 ns=64 # plot every 10 steps
        count = 0
    endif
macro/end
gaus/c/c 1 1 1e-4 # inject a gaussian mode of 1 micron

```

Jump to: [Commands](#), [Theory](#)

```
pass = 0
write/off
macro step/512
write/on
plot/watch @Name_1.plt
plot/l 3 ns=64
plot/watch @Name_2.plt
title index difference, straight fiber
plot/l 4 ns=64 h=.1
set/density 64
plot/watch @Name_3.plt
title mode shape vs. distance, straight fiber
plot/con 3 ilab=0
plot/watch @Name_4.plt
title final mode shape, straight fiber
plot/l 1 xrad=2e-4 ns=64
plot/watch @Name_5.plt
title index difference, straight fiber
plot/l 2 xrad=2e-4 ns=64 h=.1
energy 1
plot/watch @Name_6.plt
plot/y/i 3
Gap=
```

Ex87: Slab waveguides using extrude and slab/waveguide

Table. 87.1. Table of Ex87 examples

Ex87a: Two straight waveguides, extruded	1
Ex87b: Curved waveguides forming a directional coupler, equal division	4
Ex87c: Curved waveguides forming a directional coupler, 100% conversion	7
Ex87d: Y-splitter	10
Ex87e: Y-combiner, input into a single leg	13
Ex87f: Y-combiner, input into both legs	16
Ex87g: Optical switch, ON	19
Ex87h: Optical switch, OFF	22
Ex87i: Waveguide lens	27
Ex87j: Double directional coupler	29
Ex87k: Three directional couplers producing a fan-out of five equal beams	33

The examples of Ex86 use macros and the command language to implement waveguide structures. This approach allows a very high degrees of control, but significant time is spent in parsing the command lines. Special features are described in this example to calculate 2D slab waveguides quickly. The extrude command allows a great variety of 2D slab waveguides to be defined in terms of the curved paths of rib waveguide structures. The full 2D index profile is created before waveguide analysis and the command slab/waveguide performs the waveguide propagation using entirely built-in operations. Calculations which would take many minutes using the command line implementation, as used in Ex86, take a second or so with the slab/waveguide command.

Ex87a: Two straight waveguides, extruded

Two straight waveguides are separated by 14 microns. The waveguide is created by defining the path as shown in Fig. 87a.1. Fig. 87a.2 shows the result of using the extrude command to pass an index disturbance distribution along the path. Fig. 87a.3 shows the result of injecting light into the upper waveguide from the top. The coupling is weak but sufficient for the upper waveguide to leak into the lower guide. In due course, the lower guide acquires nearly 100% of the energy. In due course, the light begins leaking back into the upper waveguide. Fig. 87a.3 and Fig. 87a.4 show the waveform vs. distance.

Input: ex87a.inp

```

c## ex87a
c
c Example 87a: of two straight cores with extruded method
c
alias Name ex87a
variab/dec/int pass count nlinex nliney # define variable names
Field = 50e-4
Lambda = 1.55e-4
N0 = 1.455

dist = 20e-5 # step length

```

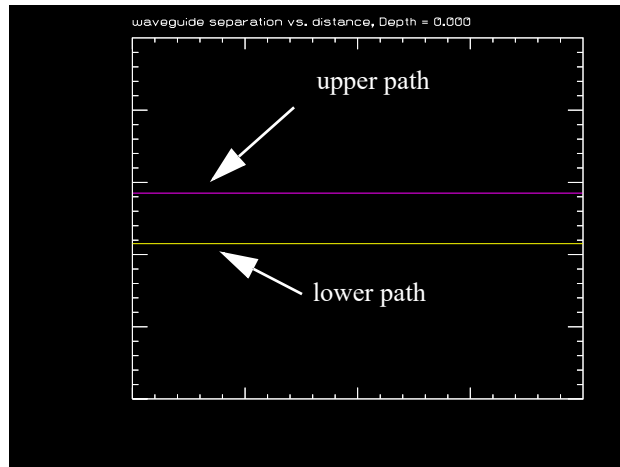


Fig. 87a.1. Paths of two close waveguides on a slab.

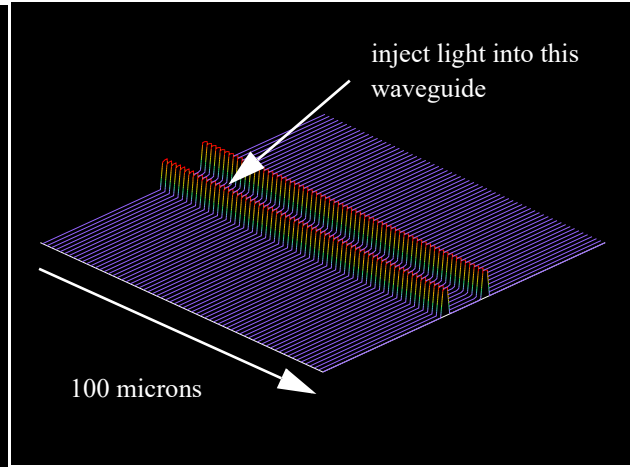


Fig. 87a.2. Index distribution for two close waveguides on a slab.

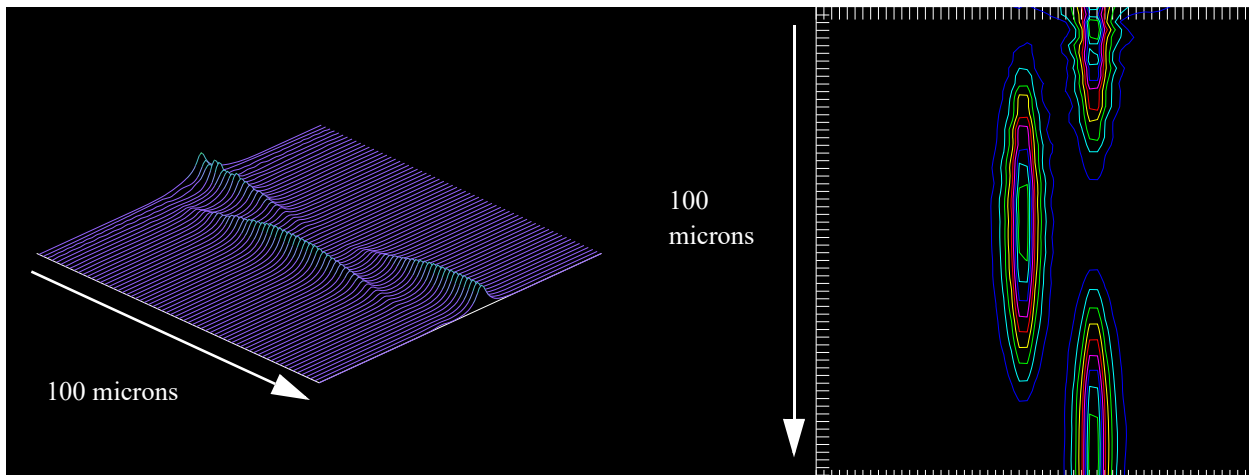


Fig. 87a.3. Light injected into the upper guide is coupled to the lower guide and then back.

Fig. 87a.4. Contour plot of coupled waveguides shows power coupling between the waveguides.

```

Fieldy = dist*2048
alpha = 2.*pi*dist/Lambda      # phase coefficient per step
alpha=
Width = 1700*dist  # was 900
Smooth = 600*dist
Apt = 1.4e-4      # was 3.8e-4
DeltaN = .003    # .3%
c dN = .032
dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list
c Depth = 12e-4 # gives 100% conversion
c Depth = 8e-4
Depth=0.
nlinex = 128                      # propagating array width
nliney = 4096                    # length of history arrays
units = 2*Field/nlinex
SeparationHalfWidth = 7e-4

```

Jump to: [Commands](#), [Theory](#)

```

set/label/off
mem/set/b 12 # request 3 megabytes
variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1 # beam 1, propagating beam
nbeam 3 data # beam 2, index distribution
nbeam 4 nlinex nliney data # beam 3, history of mode shape
nbeam 5 nlinex nliney data # beam 4, history of index profile
nbeam 7 nliney 1 data
nbeam 8 nlinex 1 data
units/field 0 Field # field half-width of 8 microns
variab/set units 3 units
units/set 4 units dist # set units of history arrays
units/set 5 units dist
wavelength/set 0 Lambda*1e4 N0 # set wavelength and cladding index
clear 1 1
clear 4 0
clear 5 0
set/density 64

units/s 6 dist
echo/on
clear 6 Depth

clap/c/c 6 Width/2
irradiance 6

convol/gaus 6 Smooth/2
phase/piston 6 180

clear 7 SeparationHalfWidth
irradiance 7

add/coh/con 6 7
copy 6 7
phase/piston 7 180
plot/w @Name_1.plt
title waveguide separation vs. distance, Depth = @Depth
plot/x/r fi=6 la=7 fmin=-Field fmax=Field

transpose 6
transpose 7

clear 8 dN
irradiance 8
clap/c/n 8 Apt

extrude/row/path 8 5 6
extrude/row/path 8 4 7
add/coh/con 5 4

```

Jump to: [Commands](#), [Theory](#)

```

c set/label/on

clear 4 0
plot/w @Name_2.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 5 ns=64 h=.1

gaus/c/c 1 1 Apt*1.2 s=1 decx=SeparationHalfWidth      # inject gaussian mode int

time/i
slab/wave 1 5 4
time
title mode shape vs. distance, Depth = @Depth
plot/w @Name_3.plt
plot/l 4 ns=64
plot/w @Name_4.plt
plot/c 4 ilab=0

```

Ex87b: Curved waveguides forming a directional coupler, equal division

The generalized extrude command can project an index of refraction profile along a path with specified deflections. The curves must be smooth to avoid mode disruption leading to radiation losses. The paths for upper and lower waveguides is constructed from rectilinear generating path (dashed) and convolved by a gaussian (shown above as dashed figure) to create smooth waveguide paths, as shown in Fig. 87b1. The width of the gaussian smoothing function may be made narrower or wider for tighter or looser curved regions. Depth and length of coupling region controls coupling efficiency. In the case of Ex87b, the beam is divided equally into both output waveguides by using a minimum separation of 10.

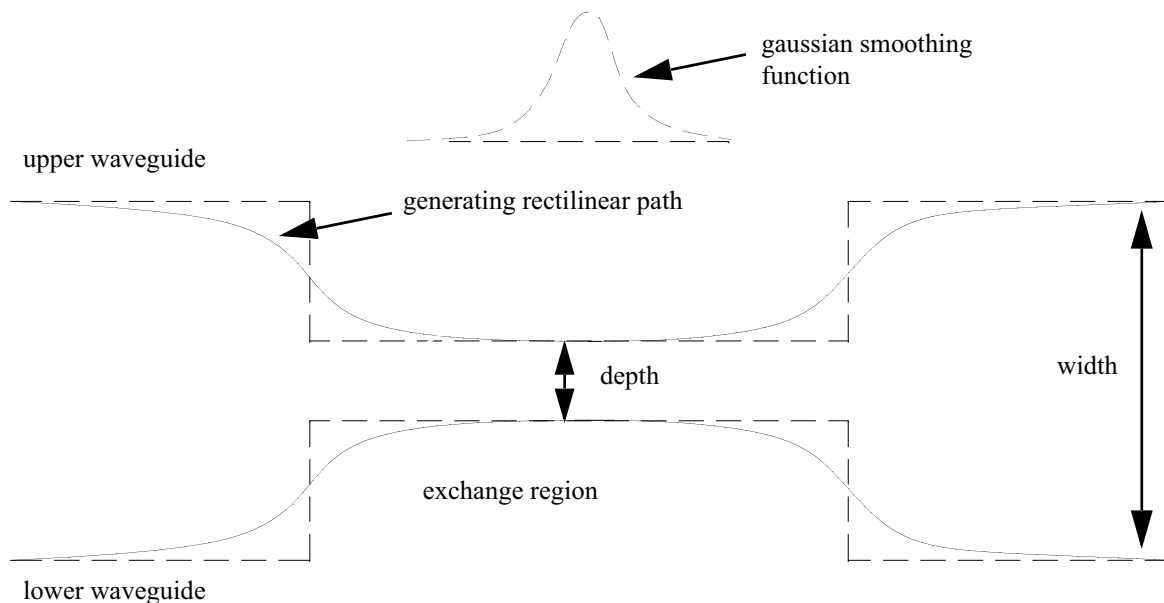


Fig. 87b.1. Generating a smooth curve for a waveguide directional coupler to minimize losses in the bend region.

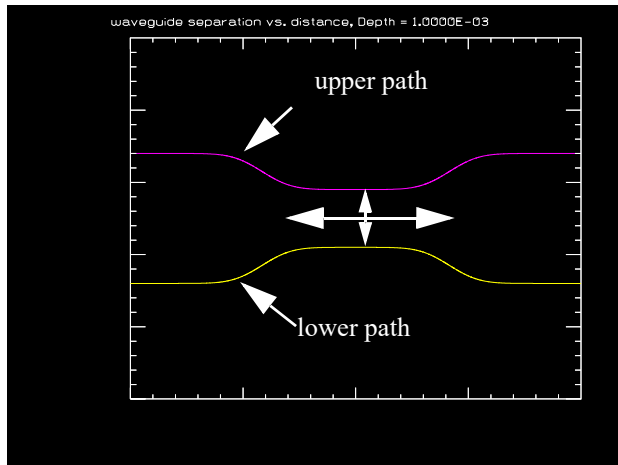


Fig. 87b.2. Change depth and length to control coupling to 50%. Total width $18\ \mu$, depth $10\ \mu$.

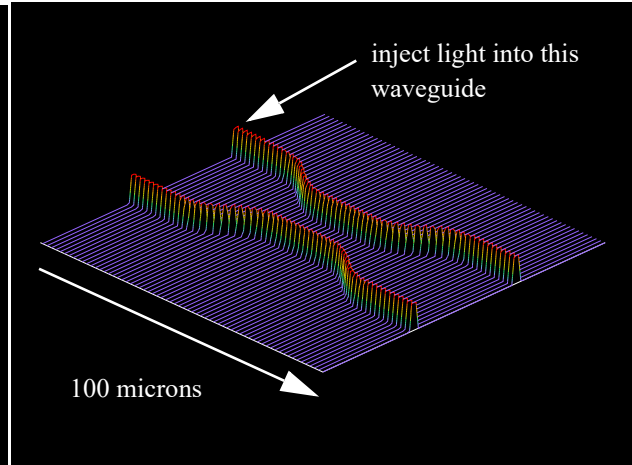


Fig. 87b.3. Index distribution for directional coupler.

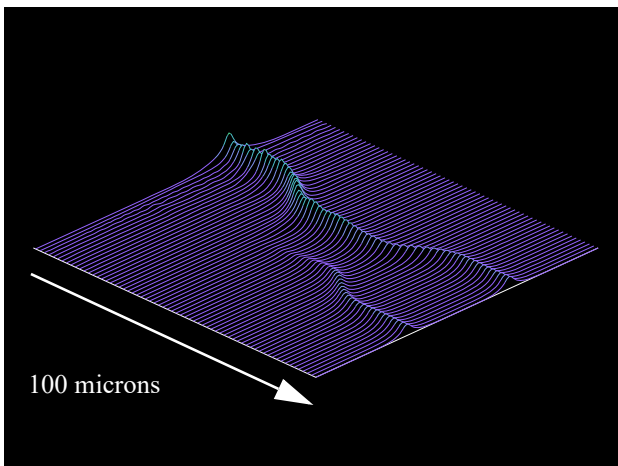


Fig. 87b.4. Light injected into the upper guide is evenly divided at the output.

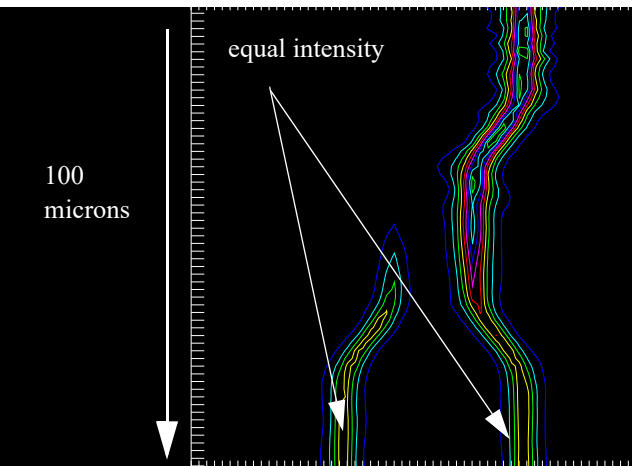


Fig. 87b.5. Contour of coupled waveguides.

Input: `ex87b.inp`

```
c## ex87b
c
c Examples 87b: Curved waveguides forming a directional coupler,
c equal division
c
alias Name ex87b

variab/dec/int pass count nlinex nliney # define variable names
Field = 50e-4
Lambda = 1.55e-4
N0 = 1.455

dist = 20e-5 # step length
```

Jump to: [Commands](#), [Theory](#)

```

Fieldy = dist*2048
alpha = 2.*pi*dist/Lambda      # phase coefficient per step
alpha=
Width = 1700*dist  # was 900
Smooth = 600*dist
Apt = 1.4e-4      # was 3.8e-4
DeltaN = .003    # .3%
c dN = .032
dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list
c Depth = 12e-4 # gives 100% conversion
Depth = 10e-4
c Depth=0.
nlinex = 128                      # propagating array width
nliney = 4096                    # length of history arrays
units = 2*Field/nlinex
SeparationHalfWidth = 18e-4

c read/d ex86chxx.inp

set/label/off
mem/set/b 12                      # request 3 megabytes
variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1                # beam 1, propagating beam
nbeam 3 data                      # beam 2, index distribution
nbeam 4 nlinex nliney data        # beam 3, history of mode shape
nbeam 5 nlinex nliney data        # beam 4, history of index profile
nbeam 7 nliney 1 data
nbeam 8 nlinex 1 data
units/field 0 Field              # field half-width of 8 microns
variab/set units 3 units
units/set 4 units dist          # set units of history arrays
units/set 5 units dist
wavelength/set 0 Lambda*1e4 NO  # set wavelength and cladding index
clear 1 1
clear 4 0
clear 5 0
set/density 64

units/s 6 dist
echo/on
clear 6 Depth

clap/c/c 6 Width/2
irradiance 6

convol/gaus 6 Smooth/2
phase/piston 6 180

clear 7 SeparationHalfWidth
irradiance 7

```

Jump to: [Commands](#), [Theory](#)

```

add/coh/con 6 7
copy 6 7
phase/piston 7 180
plot/w @Name_1.plt
title waveguide separation vs. distance, Depth = @Depth
plot/x/r fi=6 la=7 fmin=-Field fmax=Field

transpose 6
transpose 7

clear 8 dN
irradiance 8
clap/c/n 8 Apt

extrude/row/path 8 5 6
extrude/row/path 8 4 7
add/coh/con 5 4

c set/label/on

clear 4 0
plot/w @Name_2.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 5 ns=64 h=.1

gaus/c/c 1 1 Apt*1.2 s=1 decx=SeparationHalfWidth      # inject gaussian mode int

time/i
slab/wave 1 5 4
time
title mode shape vs. distance, Depth = @Depth
plot/w @Name_3.plt
plot/l 4 ns=64
plot/w @Name_4.plt
plot/c 4 ilab=0

```

Ex87c: Curved waveguides forming a directional coupler, 100% conversion

Ex87c shows a directional coupler with depth set to 12 to achieve 100% conversion as shown in Figs. 87c.1-87c.4.

Input: ex87c.inp

```

c## ex87c
c
c Examples 87c: Curved waveguides forming a directional coupler,
c              100% conversion
c
variab/dec/int pass count nlinex nliney # define variable names
Field = 50e-4
Lambda = 1.55e-4

```

Jump to: [Commands](#), [Theory](#)

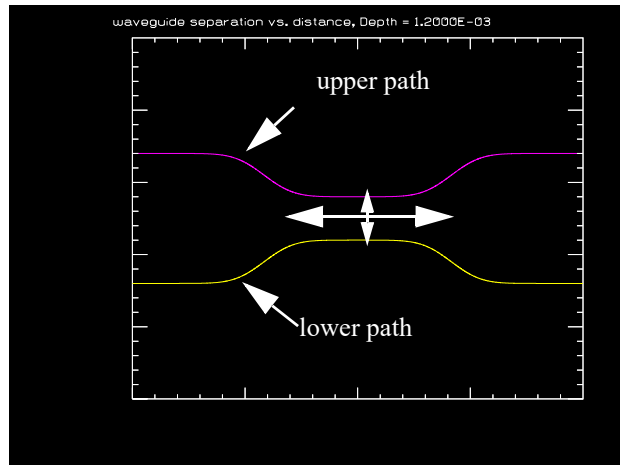


Fig. 87c.1. Change depth and length to control coupling to 50%. Total width $18 \mu\text{m}$, depth $12 \mu\text{m}$.

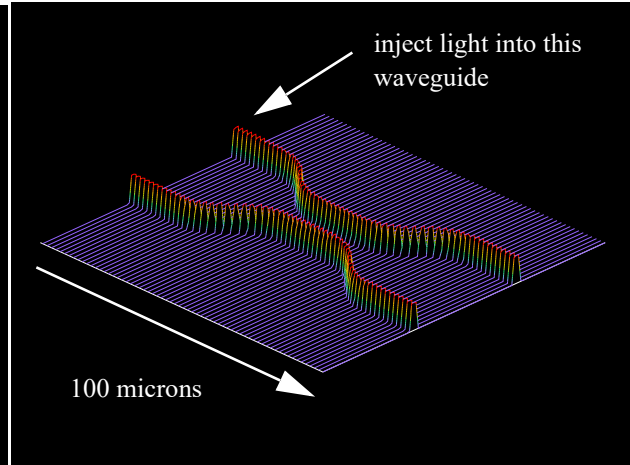


Fig. 87c.2. Index distribution for directional coupler.

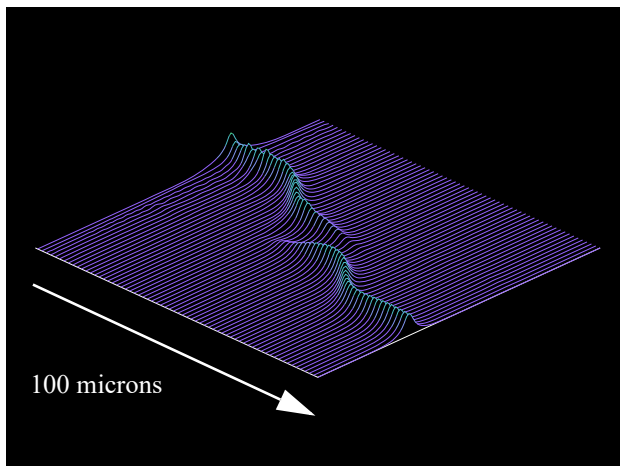


Fig. 87c.3. Light injected into the upper guide is completely converted to the lower guide.

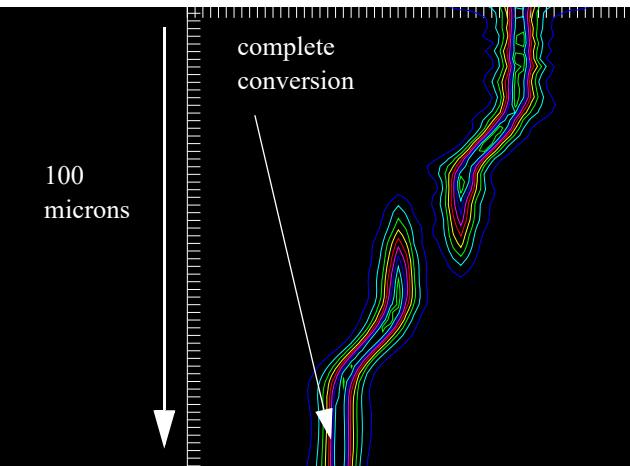


Fig. 87c.4. Contour of coupled waveguides.

$N_0 = 1.455$

$\text{dist} = 20\text{e-}5$ # step length

$\text{Fieldy} = \text{dist} * 2048$

$\alpha = 2 * \pi * \text{dist} / \text{Lambda}$

phase coefficient per step

$\alpha =$

$\text{Width} = 1700 * \text{dist}$ # was 900

$\text{Smooth} = 600 * \text{dist}$

$\text{Apt} = 1.4\text{e-}4$ # was $3.8\text{e-}4$

$\Delta N = .003$ # .3%

$c \Delta N = .032$

$\Delta N = \sqrt{1.455^2 / (1 - 2 * \Delta N)}$ - N_0 list

$\text{Depth} = 12\text{e-}4$ # gives 100% conversion

$n_{\text{linex}} = 128$

propagating array width

$n_{\text{liney}} = 4096$

length of history arrays

Jump to: [Commands](#), [Theory](#)

```

units = 2*Field/nlinex
SeparationHalfWidth = 18e-4

alias Name ex87c

c read/d ex86cixx.inp
set/label/off
mem/set/b 12 # request 3 megabytes
variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1 # beam 1, propagating beam
nbeam 3 data # beam 2, index distribution
nbeam 4 nlinex nliney data # beam 3, history of mode shape
nbeam 5 nlinex nliney data # beam 4, history of index profile
nbeam 7 nliney 1 data
nbeam 8 nlinex 1 data
units/field 0 Field # field half-width of 8 microns
variab/set units 3 units
units/set 4 units dist # set units of history arrays
units/set 5 units dist
wavelength/set 0 Lambda*1e4 N0 # set wavelength and cladding index
clear 1 1
clear 4 0
clear 5 0
set/density 64

units/s 6 dist
echo/on
clear 6 Depth

clap/c/c 6 Width/2
irradiance 6

convol/gaus 6 Smooth/2
phase/piston 6 180

clear 7 SeparationHalfWidth
irradiance 7

add/coh/con 6 7
copy 6 7
phase/piston 7 180
plot/w @Name_1.plt
title waveguide separation vs. distance, Depth = @Depth
plot/x/r fi=6 la=7 fmin=-Field fmax=Field

transpose 6
transpose 7

clear 8 dN
irradiance 8
clap/c/n 8 Apt

```

Jump to: [Commands](#), [Theory](#)

```

extrude/row/path 8 5 6
extrude/row/path 8 4 7
add/coh/con 5 4

c set/label/on

clear 4 0
plot/w @Name_2.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 5 ns=64 h=.1

gaus/c/c 1 1 Apt*1.2 s=1 decx=SeparationHalfWidth      # inject gaussian mode int

time/i
slab/wave 1 5 4
time
title mode shape vs. distance, Depth = @Depth
plot/w @Name_3.plt
plot/l 4 ns=64
plot/w @Name_4.plt
plot/c 4 ilab=0

```

Ex87d: Y-splitter

Ex87d demonstrates a y-splitter. Two paths are made to be initially coincident and then spread apart, as indicated in Fig. 87d.1.

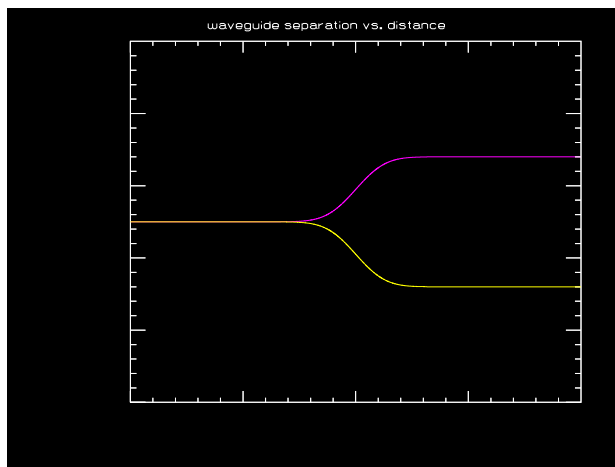


Fig. 87d.1. Path of split waveguide.

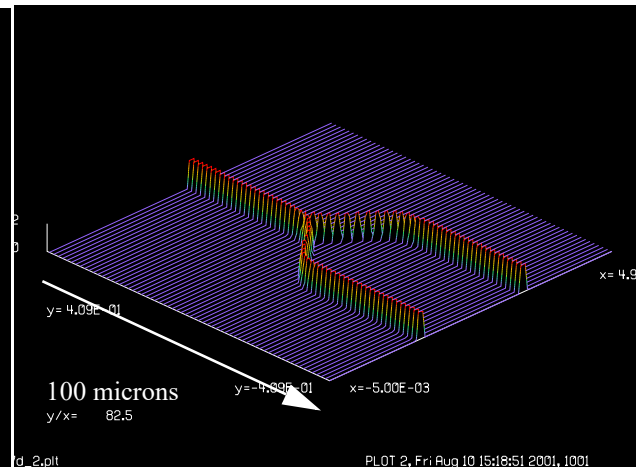


Fig. 87d.2. Index distribution for y-splitter.

Input: ex87d.inp

```

c## ex87d
c
c Example 87d: Y-splitter
c

```

Jump to: [Commands](#), [Theory](#)

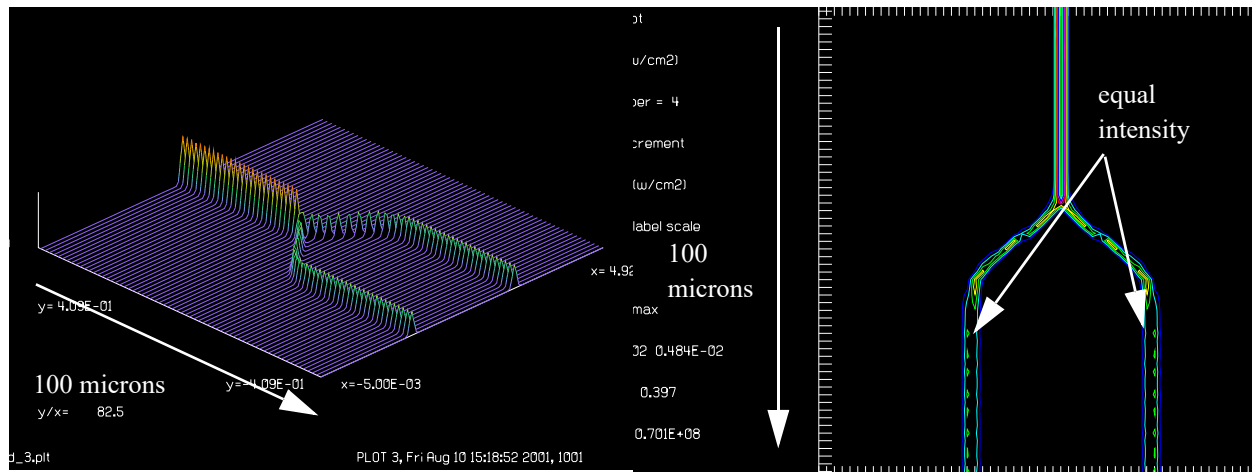


Fig. 87d.3. The propagating modes for the y-junction. Efficiency is about 98%.

Fig. 87d.4. Contour of y-junction.

```
alias Name ex87d
variab/dec/int pass count nlinex nliney # define variable names
set/label/off
Field = 50e-4
Lambda = 1.55e-4
N0 = 1.455

dist = 20e-5 # step length

Fieldy = dist*2048
alpha = 2.*pi*dist/Lambda # phase coefficient per step
alpha=
Width = 2800*dist # was 900
Smooth = 600*dist
Apt = 1.4e-4 # was 3.8e-4
DeltaN = .003 # .3%
c dN = .032
dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list
nlinex = 128 # propagating array width
nliney = 4096 # length of history arrays
units = 2*Field/nlinex
SeparationHalfWidth = 18e-4
Depth = SeparationHalfWidth

set/label/off
mem/set/b 12 # request 3 megabytes
variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1 # beam 1, propagating beam
nbeam 3 data # beam 2, index distribution
nbeam 4 nlinex nliney data # beam 3, history of mode shape
nbeam 5 nlinex nliney data # beam 4, history of index profile
nbeam 7 nliney 1 data
nbeam 8 nlinex 1 data
```

Jump to: [Commands](#), [Theory](#)

```

units/field 0 Field                                # field half-width of 8 microns
variab/set units 3 units
units/set 4 units dist                             # set units of history arrays
units/set 5 units dist
wavelength/set 0 Lambda*1e4 N0                     # set wavelength and cladding index
clear 1 1
clear 4 0
clear 5 0
set/density 64

units/s 6 dist
echo/on
clear 6 Depth

clap/c/c 6 Width/2
irradiance 6

convol/gaus 6 Smooth/2

rescale/shift 6 -Width/2

phase/piston 6 180

clear 7 SeparationHalfWidth
irradiance 7

add/coh/con 6 7
copy 6 7
phase/piston 7 180
plot/w @Name_1.plt
title waveguide separation vs. distance
plot/x/r fi=6 la=7 fmin=-Field fmax=Field

transpose 6
transpose 7

clear 8 dN
irradiance 8
clap/c/n 8 Apt

extrude/row/path 8 5 6
extrude/row/path 8 4 7
add/coh/con 5 4
peak/norm/rows 5 dN

clear 4 0
plot/w @Name_2.plt
title index difference vs. distance

plot/l/r 5 ns=64 h=.1

gaus/c/c 1 1 Apt*1.2 s=1      # inject gaussian mode into one core
energy/norm 1 1

```

Jump to: [Commands](#), [Theory](#)


```

time/i
slab/wave 1 5 4
time
title mode shape vs. distance
plot/w @Name_3.plt
plot/l 4 ns=64 h=.2
plot/w @Name_4.plt
plot/c 4 ilab=0

plot/w @Name_5.plt
title final mode shape
plot/x/i 1
energy

```

Ex87e: Y-combiner, input into a single leg

Ex87e demonstrates a y-combiner. Two paths are initially separated and made to combined, as indicated in Fig. 87e.1. Light is injected into only one leg. The beam is not mode matched because of the single-leg injection causing significant loss at the junction.

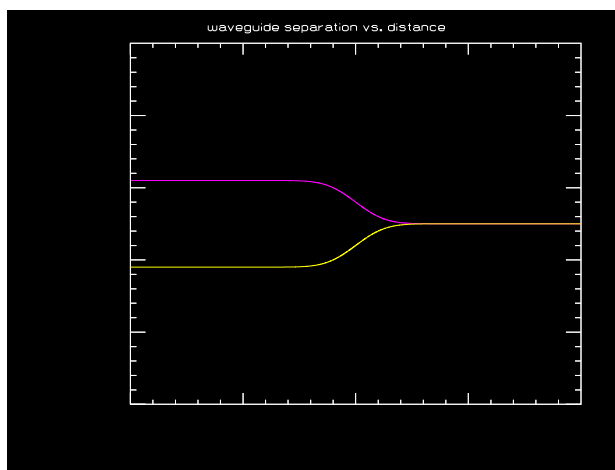


Fig. 87e.1. Path of y-combiner waveguide.

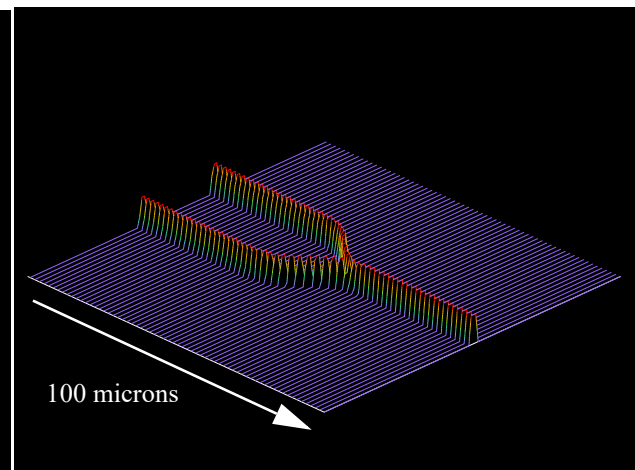


Fig. 87e.2. Index distribution for y-combiner.

Input: ex87e.inp

```

c## ex87e
c
c Examples 87e: Y-combiner, input into a single leg
c
alias Name ex87e
variab/dec/int pass count nlinex nliney # define variable names
Field = 50e-4
Lambda = 1.55e-4
Lambda = 3e-4
N0 = 1.455

dist = 5e-5 # step length

```

Jump to: [Commands](#), [Theory](#)

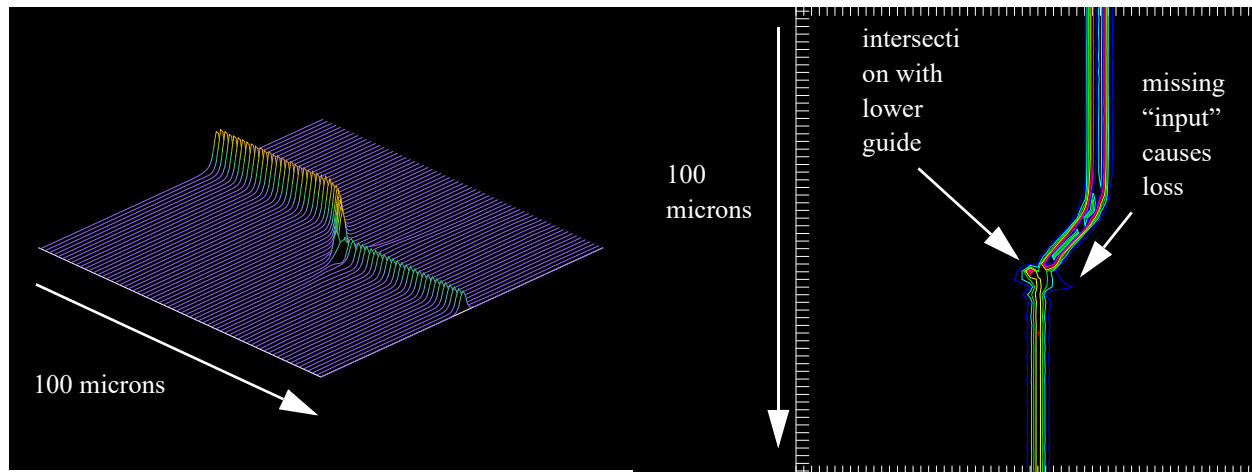


Fig. 87e.3. Injection into only the upper guide. Imperfect coupling. Efficiency is about 50%.

Fig. 87e.4. Mode coupling mismatch after intersection with empty waveguide.

```

Fieldy = dist*2048
alpha = 2.*pi*dist/Lambda          # phase coefficient per step
alpha=
Width = 2800*dist  # was 900
Smooth = 600*dist
Apt = 1.4e-4      # was 3.8e-4
DeltaN = .003    # .3%
c dN = .032
dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list
nlinex = 128                      # propagating array width
nliney = 4096                     # length of history arrays
units = 2*Field/nlinex
SeparationHalfWidth = 12e-4
Depth = SeparationHalfWidth
variab/dec/int TWO
TWO = 1

set/label/off
mem/set/b 12                      # request 3 megabytes
variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers

array/s 1 nlinex 1                # beam 1, propagating beam
nbeam 3 data                       # beam 2, index distribution
nbeam 4 nlinex nliney data        # beam 3, history of mode shape
nbeam 5 nlinex nliney data        # beam 4, history of index profile
nbeam 7 nliney 1 data
nbeam 8 nlinex 1 data
units/field 0 Field               # field half-width of 8 microns
variab/set units 3 units
units/set 4 units dist            # set units of history arrays
units/set 5 units dist
wavelength/set 0 Lambda*1e4 N0    # set wavelength and cladding index

```

Jump to: [Commands](#), [Theory](#)

```

clear 1 1
clear 4 0
clear 5 0
set/density 64

units/s 6 dist
echo/on
clear 6 Depth

clap/c/c 6 Width/2
irradiance 6

convol/gaus 6 Smooth/2

rescale/shift 6 Width/2

phase/piston 6 180

clear 7 SeparationHalfWidth
irradiance 7

add/coh/con 6 7
copy 6 7
phase/piston 7 180
plot/w @Name_1.plt
title waveguide separation vs. distance
if TWO then
    plot/x/r fi=6 la=7 fmin=-Field fmax=Field
else
    plot/x/r 7 fmin=-Field fmax=Field
endif

transpose 6
transpose 7

clear 8 dN
irradiance 8
clap/c/n 8 Apt

extrude/row/path 8 5 6
if TWO extrude/row/path 8 4 7
add/coh/con 5 4
peak/norm/rows 5 dN

c set/label/on

clear 4 0
plot/w @Name_2.plt
title index difference vs. distance
plot/l/r 5 ns=64 h=.1

gaus/c/c 1 1 Apt*1.2 s=1 SeparationHalfWidth    # inject gaussian mode into one
energy/norm 1 1

```

Jump to: [Commands](#), [Theory](#)

```

time/i
slab/wave 1 5 4
time
title mode shape vs. distance
plot/w @Name_3.plt
plot/l 4 ns=64 h=.2
plot/w @Name_4.plt
plot/c 4 ilab=0
energy 1

```

Ex87f: Y-combiner, input into both legs

Ex87f demonstrates a y-combiner with the same input into both legs. The beam is mode matched because it is the exact reverse of the y-splitter.

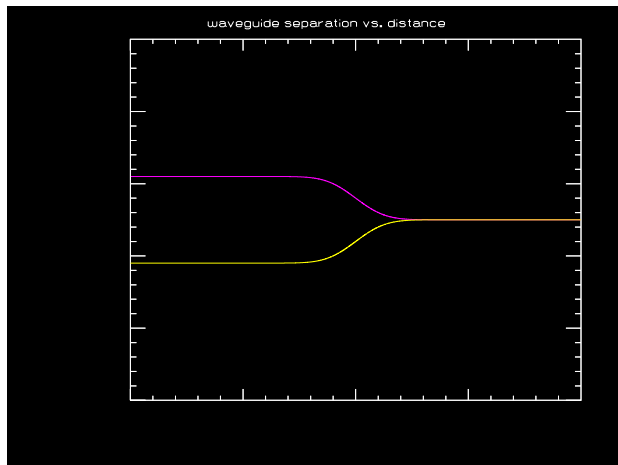


Fig. 87f.1. Path of y-combiner waveguide.

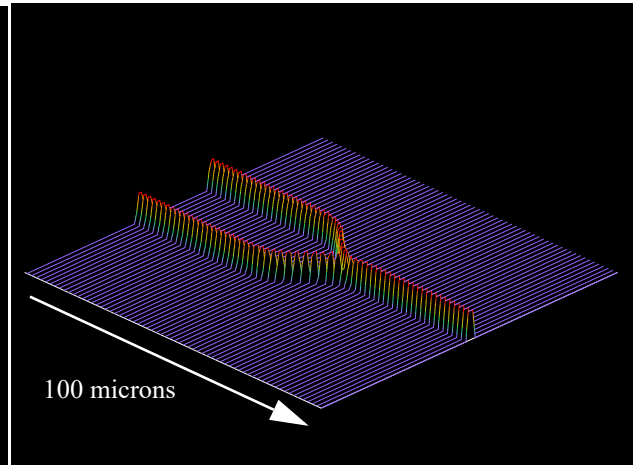


Fig. 87f.2. Index distribution for y-combiner.

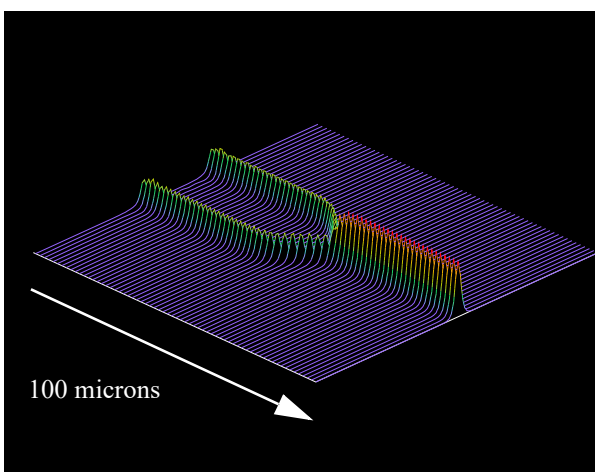


Fig. 87f.3. Injection of identical modes into both guides. Efficiency is about 100%.

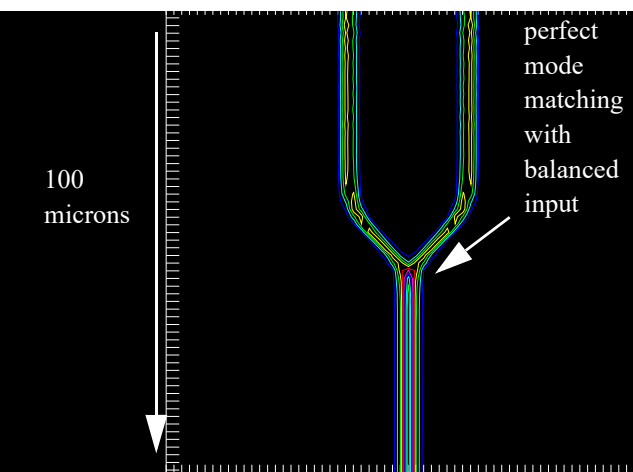


Fig. 87f.4. Mode coupling mismatch after intersection with empty waveguide.

Input: ex87f.inp

```

c## ex87f
c
c Example 87f: Y-combiner, input into both legs
c
alias Name ex87f
variab/dec/int pass count nlinex nliney # define variable names
Field = 50e-4
Lambda = 1.55e-4
Lambda = 3e-4
N0 = 1.455

dist = 5e-5 # step length

Fieldy = dist*2048
alpha = 2.*pi*dist/Lambda # phase coefficient per step
alpha=
Width = 2800*dist # was 900
Smooth = 600*dist
Apt = 1.4e-4 # was 3.8e-4
DeltaN = .003 # .3%
c dN = .032
dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list
nlinex = 128 # propagating array width
nliney = 4096 # length of history arrays
units = 2*Field/nlinex
SeparationHalfWidth = 12e-4
Depth = SeparationHalfWidth

set/label/off
mem/set/b 12 # request 3 megabytes
variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1 # beam 1, propagating beam
nbeam 3 data # beam 2, index distribution
nbeam 4 nlinex nliney data # beam 3, history of mode shape
nbeam 5 nlinex nliney data # beam 4, history of index profile
nbeam 7 nliney 1 data
nbeam 8 nlinex 1 data
units/field 0 Field # field half-width of 8 microns
variab/set units 3 units
units/set 4 units dist # set units of history arrays
units/set 5 units dist
wavelength/set 0 Lambda*1e4 N0 # set wavelength and cladding index
clear 1 1
clear 4 0
clear 5 0
set/density 64

units/s 6 dist
echo/on
clear 6 Depth

```

Jump to: [Commands](#), [Theory](#)

```
clap/c/c 6 Width/2
irradiance 6

convol/gaus 6 Smooth/2

rescale/shift 6 Width/2

phase/piston 6 180

clear 7 SeparationHalfWidth
irradiance 7

add/coh/con 6 7
copy 6 7
phase/piston 7 180
plot/w @Name_1.plt
title waveguide separation vs. distance
variab/dec/int TWO
TWO = 1
if TWO then
    plot/x/r fi=6 la=7 fmin=-Field fmax=Field
else
    plot/x/r 7 fmin=-Field fmax=Field
endif

transpose 6
transpose 7

clear 8 dN
irradiance 8
clap/c/n 8 Apt

extrude/row/path 8 5 6
if TWO extrude/row/path 8 4 7
add/coh/con 5 4
peak/norm/rows 5 dN

c set/label/on

clear 4 0
plot/w @Name_2.plt
title index difference vs. distance
plot/l/r 5 ns=64 h=.1

gaus/c/c 1 1 Apt*1.2 s=1 SeparationHalfWidth # inject gaussian mode into one
gaus/c/c 8 1 Apt*1.2 s=1 -SeparationHalfWidth
add/inc/con 1 8
energy/norm 1 1

time/i
slab/wave 1 5 4
time
```

Jump to: [Commands](#), [Theory](#)

```

title mode shape vs. distance
plot/w @Name_3.plt
plot/l 4 ns=64 h=.2
plot/w @Name_4.plt
plot/c 4 ilab=0
energy 1

```

Ex87g: Optical switch, ON

Ex87g demonstrates an optical switch in the ON position. The switch may be thought of as a y-splitter followed by a y-combiner or a Mach-Zender interferometer with both legs of equal length.

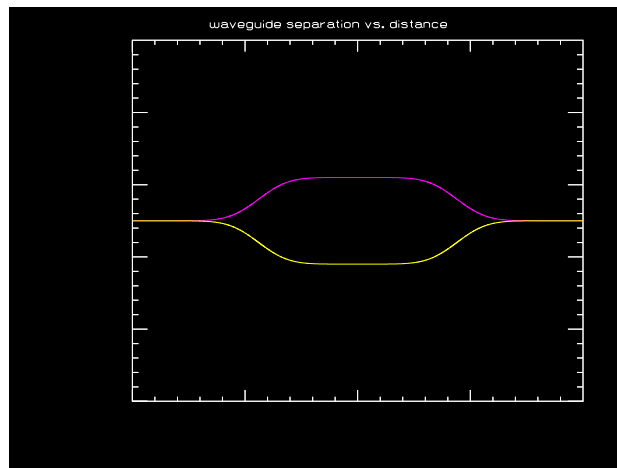


Fig. 87g.1. Path of optical switch.

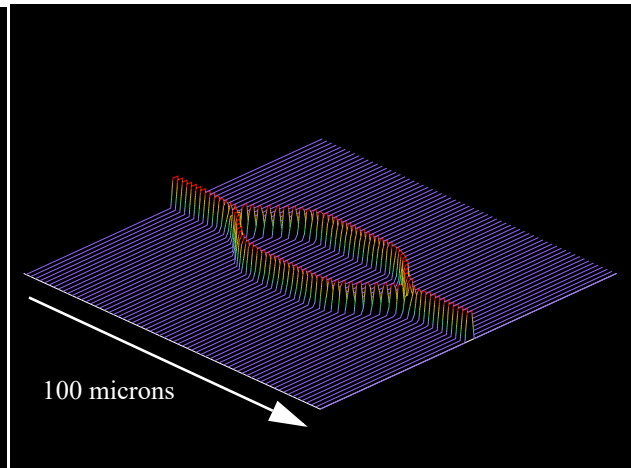


Fig. 87g.2. Index distribution for optical switch. Both legs are identical for 100% efficiency.

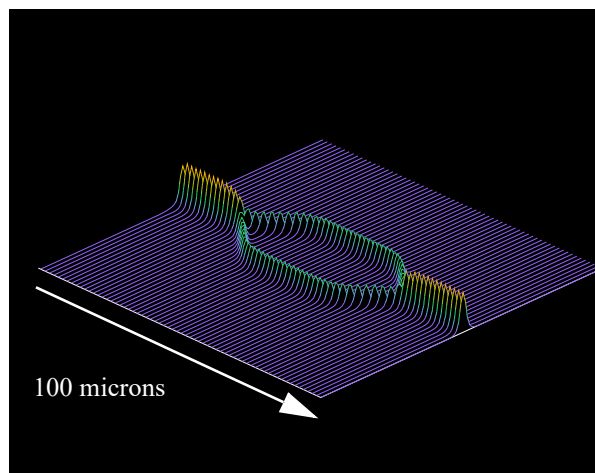


Fig. 87g.3. Optical switch in the ON position.

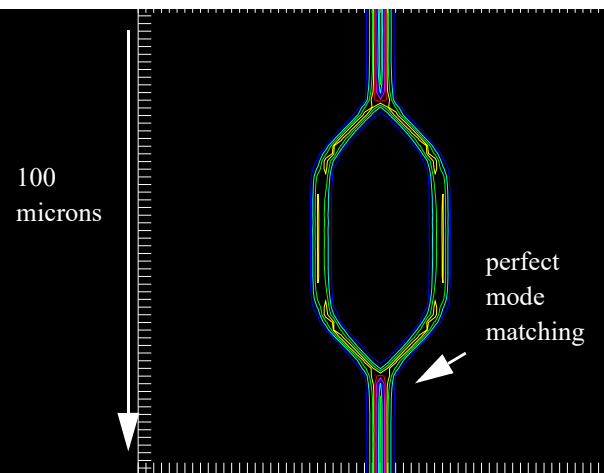


Fig. 87g.4. Optical switch in the ON position.

Input: ex87g.inp

```

c## ex87g
c

```

Jump to: [Commands](#), [Theory](#)

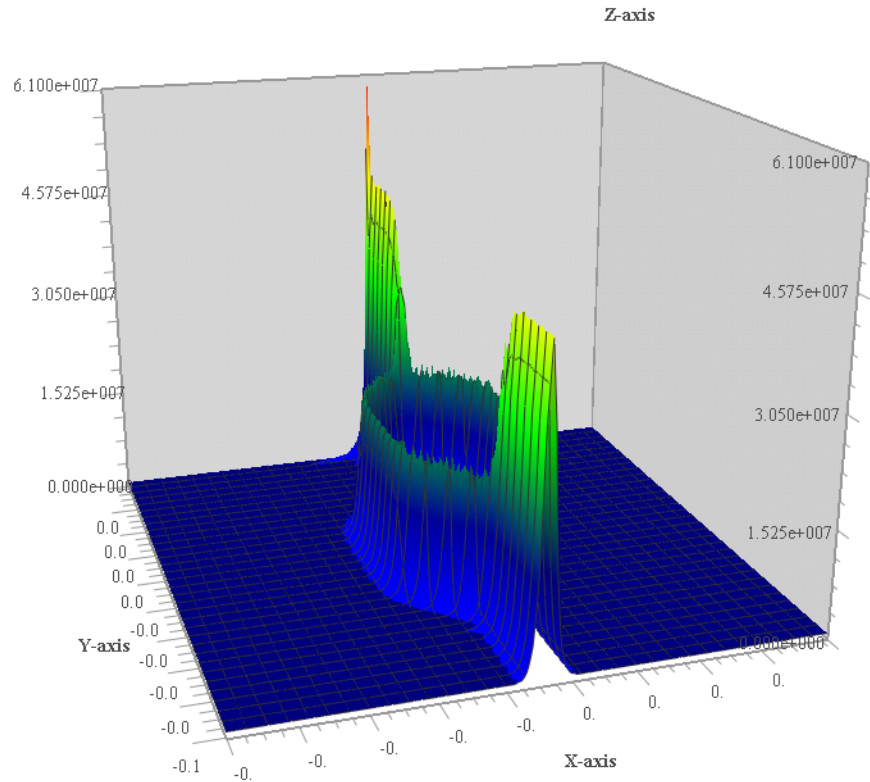


Fig. 87g.5. Optical switch in the ON position, shown with plot/bitmap/intensity/arrayvisualizer.

c Examples 87g: Optical switch, ON

c

alias Name ex87g

variab/dec/int pass count nlinex nliney # define variable names

Field = 50e-4

Lambda = 1.55e-4

Lambda = 3e-4

N0 = 1.455

dist = 5e-5 # step length

Fieldy = dist*2048

alpha = 2.*pi*dist/Lambda # phase coefficient per step

alpha=

Width = 1800*dist # was 900

Smooth = 600*dist

Apt = 1.4e-4 # was 3.8e-4

DeltaN = .003 # .3%

c dN = .032

dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list

nlinex = 128 # propagating array width

nliney = 4096 # length of history arrays

units = 2*Field/nlinex

SeparationHalfWidth = 12e-4

Jump to: [Commands](#), [Theory](#)

Depth = SeparationHalfWidth

```

set/label/off
mem/set/b 12                                # request 3 megabytes
variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1                          # beam 1, propagating beam
nbeam 3 data                                # beam 2, index distribution
nbeam 4 nlinex nliney data                  # beam 3, history of mode shape
nbeam 5 nlinex nliney data                  # beam 4, history of index profile
nbeam 7 nliney 1 data
nbeam 8 nlinex 1 data
units/field 0 Field                        # field half-width of 8 microns
variab/set units 3 units
units/set 4 units dist                     # set units of history arrays
units/set 5 units dist
wavelength/set 0 Lambda*1e4 N0             # set wavelength and cladding index
clear 1 1
clear 4 0
clear 5 0
set/density 256

units/s 6 dist
echo/on
clear 6 Depth

clap/c/c 6 Width/2
irradiance 6

convol/gaus 6 Smooth/2

c rescale/shift 6 Width/2

c phase/piston 6 180

c clear 7 SeparationHalfWidth
c irradiance 7

c add/coh/con 6 7

copy 6 7
phase/piston 7 180
plot/w @Name_1.plt
title waveguide separation vs. distance
variab/dec/int TWO
TWO = 1
if TWO then
    plot/x/r fi=6 la=7 fmin=-Field fmax=Field
else
    plot/x/r 7 fmin=-Field fmax=Field
endif

```

Jump to: [Commands](#), [Theory](#)

```

transpose 6
transpose 7

clear 8 dN
irradiance 8
clap/c/n 8 Apt

extrude/row/path 8 5 6
if TWO extrude/row/path 8 4 7
add/coh/con 5 4
peak/norm/rows 5 dN

c set/label/on

clear 4 0
plot/w @Name_2.plt
title index difference vs. distance
plot/l/r 5 ns=64 h=.1

gaus/c/c 1 1 Apt*1.2 s=1      # inject gaussian mode into one core
add/inc/con 1 8
energy/norm 1 1

time/i
slab/wave 1 5 4
time
title mode shape vs. distance
plot/w @Name_3.plt
plot/l 4 ns=64 h=.2
plot/w @Name_5.plt
plot/bit/i/arr 4

plot/w @Name_4.plt
plot/c 4 ilab=0
energy 1

```

Ex87h: Optical switch, OFF

Ex87h demonstrates an optical switch in the OFF position. The switch may be thought of as a y-splitter followed by a y-combiner or a Mach-Zender interferometer. A patch of higher index material is added to the index array after has been formed by the extrude command to induce a π phase change in the optical path length.

Input: ex87h.inp

```

c## ex87h
c
c Example 87h: Opticla switch, OFF
c
alias Name ex87h
variab/dec/int pass count nlinex nliney # define variable names

```

Jump to: [Commands](#), [Theory](#)

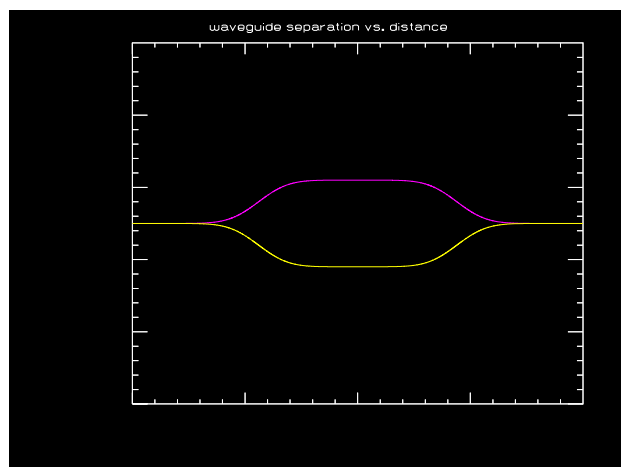


Fig. 87h.1. Path of optical switch.

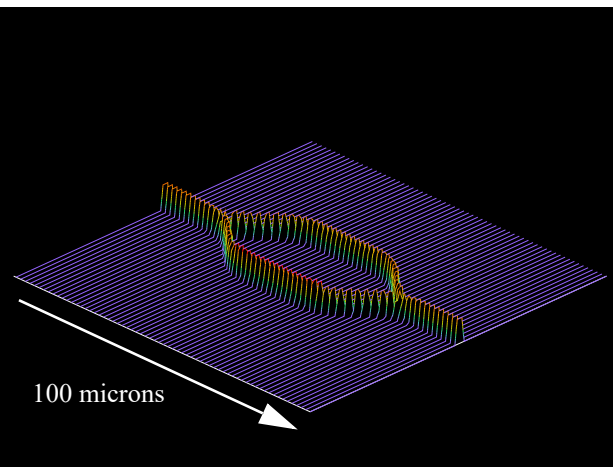


Fig. 87h.2. Index distribution for optical switch. Both legs are identical for 100% efficiency.

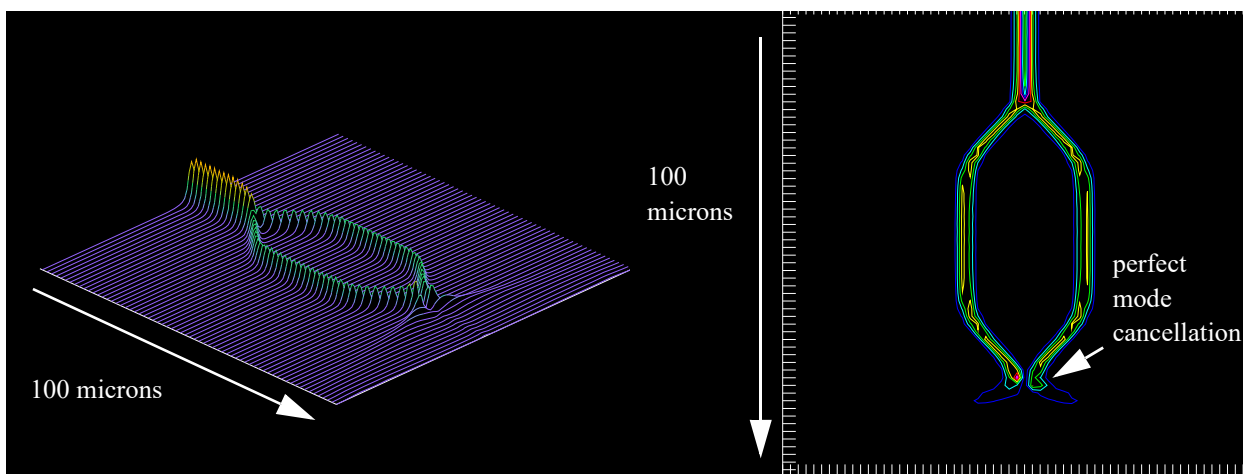


Fig. 87h.3. Optical switch in the OFF position.

Fig. 87h.4. Optical switch in the OFF position.

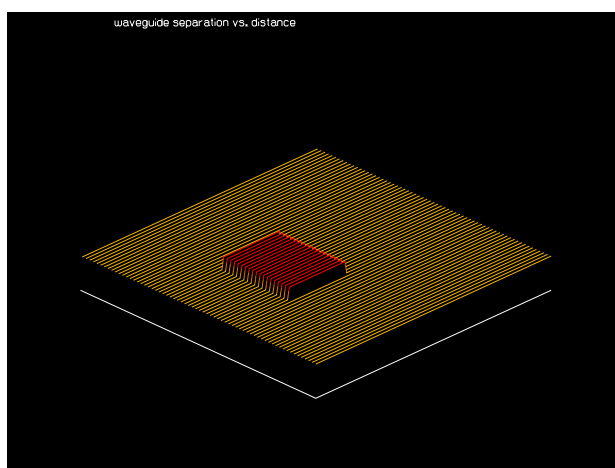


Fig. 87h.5. Region of waveguide given higher index to cause 180 deg. phase change in the lower leg.

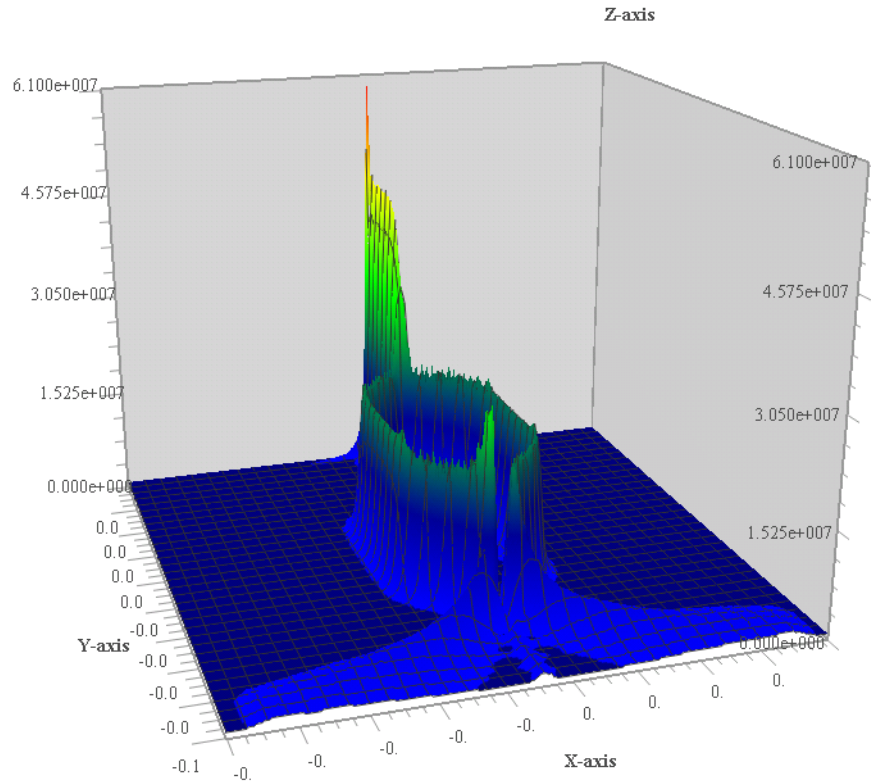


Fig. 87h.6. Optical switch in the OFF position, shown with plot/bitmap/intensity/arrayvisualizer.

```
Field = 50e-4
Lambda = 1.55e-4
Lambda = 3e-4
N0 = 1.455
```

```
dist = 5e-5 # step length
```

```
Fieldy = dist*2048
alpha = 2.*pi*dist/Lambda      # phase coefficient per step
alpha=
Width = 1800*dist # was 900
Smooth = 600*dist
Apt = 1.4e-4 # was 3.8e-4
DeltaN = .003 # .3%
c dN = .032
dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list
nlinex = 128                    # propagating array width
nliney = 4096                  # length of history arrays
units = 2*Field/nlinex
SeparationHalfWidth = 12e-4
Depth = SeparationHalfWidth
```

```
set/label/off
mem/set/b 12                    # request 3 megabytes
```

Jump to: [Commands](#), [Theory](#)

```

variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1 # beam 1, propagating beam
nbeam 3 data # beam 2, index distribution
nbeam 4 nlinex nliney data # beam 4, history of index profile
nbeam 5 nlinex nliney data # beam 5, history of index profile
nbeam 7 nliney 1 data
nbeam 8 nlinex 1 data
nbeam 9 nlinex nliney data
units/field 0 Field # field half-width of 8 microns
variab/set units 3 units
units/set 4 units dist # set units of history arrays
units/set 5 units dist
wavelength/set 0 Lambda*1e4 N0 # set wavelength and cladding index
clear 1 1
clear 4 0
clear 5 0
set/density 256

units/s 6 dist
echo/on
clear 6 Depth

clap/c/c 6 Width/2
irradiance 6

convol/gaus 6 Smooth/2

c rescale/shift 6 Width/2

c phase/piston 6 180

c clear 7 SeparationHalfWidth
c irradiance 7

c add/coh/con 6 7

copy 6 7
phase/piston 7 180
plot/w @Name_1.plt
title waveguide separation vs. distance
variab/dec/int TWO
TWO = 1
if TWO then
    plot/x/r fi=6 la=7 fmin=-Field fmax=Field
else
    plot/x/r 7 fmin=-Field fmax=Field
endif

transpose 6
transpose 7

```

Jump to: [Commands](#), [Theory](#)

```
clear 8 dN
irradiance 8
clap/c/n 8 Apt

extrude/row/path 8 5 6
if TWO extrude/row/path 8 4 7
add/coh/con 5 4
peak/norm/rows 5 dN

clear 9 .353
clap/r/c 9 Depth Width/2 -Depth*1.2
clear 4 1

add/inc/c 4 9
mult/beam 5 4
plot/w @Name_5.plt
plot/l 4 ns=64 h=.2

c set/label/on

clear 4 0
plot/w @Name_2.plt
title index difference vs. distance
plot/l/r 5 ns=64 h=.1

gaus/c/c 1 1 Apt*1.2 s=1      # inject gaussian mode into one core
energy/norm 1 1

time/i
slab/wave 1 5 4
time
title mode shape vs. distance
plot/w @Name_3.plt
plot/l 4 ns=64 h=.2
plot/w @Name_6.plt
plot/bit/i/arr 4
plot/w @Name_4.plt
plot/c 4 ilab=0

energy 1
peak 1
nbeam 10 128 256
copy/border 4 10 jskip1=16
units/s 10 1 1
peak/norm 10 1
set/den 256 256
plot/w @Name_7.plt
plot/b/i/b 10

read/scr
flip/y 10
plot/l 10
outfile/intens ex86cn.txt/no/comma 10
```

Jump to: [Commands](#), [Theory](#)

Ex87i: Waveguide lens

Ex87i demonstrates a waveguide lens. In this example a parabolic index distribution is created directly into the index array (the `extrude` command is not used for this example). The `slab/waveguide` command processes the index array quickly.

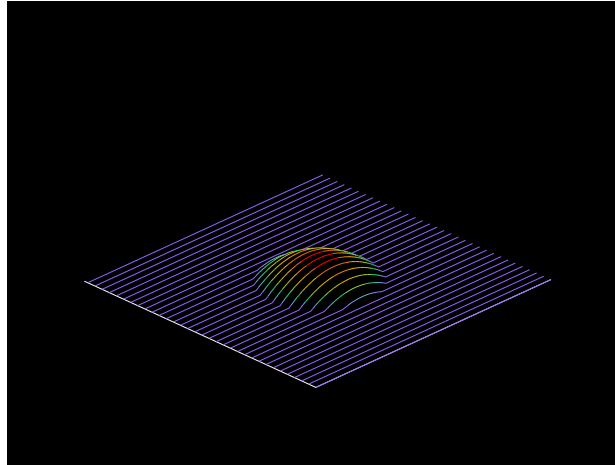


Fig. 87i.1. Index distribution with parabolic higher index region.

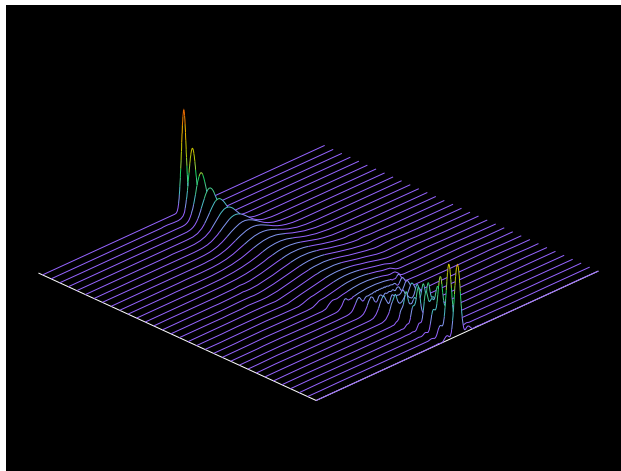


Fig. 87i.2. Optical mode vs. distance starting from a gaussian and forming a near-gaussian image.

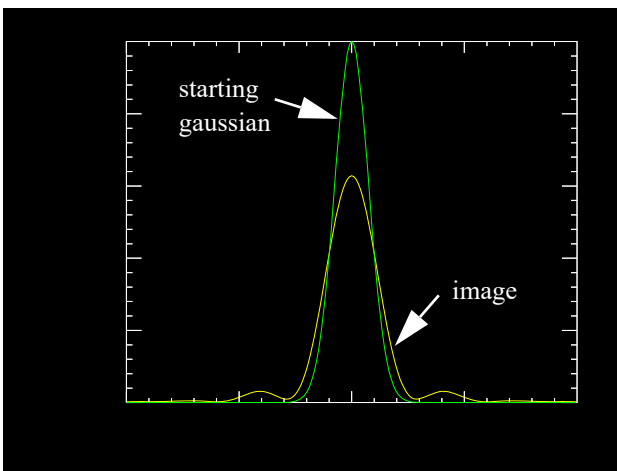


Fig. 87i.3. Comparison of starting gaussian with ending beam. Note sidelobes in image due to truncation of parabolic index hill.

Input: ex87i.inp

```
c## ex87i
c
c Example 87i: Waveguide lens
c
alias Name ex87i
variab/dec/int pass count nlinex nliney # define variable names
Field = 50e-4
Lambda = 1.55e-4
Lambda = 3e-4
```

Jump to: [Commands](#), [Theory](#)

```

N0 = 1.455

Apt = 1.4e-4    # was 3.8e-4
Width = Apt
DeltaN = .003  # .3%
c dN = .032
dN = .225
nlinex = 1024          # propagating array width
nliney = 1024          # length of history arrays
SeparationHalfWidth = 12e-4
Depth = SeparationHalfWidth
units = 2*Field/nlinex
dist = units
alpha = 2.*pi*dist/Lambda    # phase coefficient per step
alpha=

set/label/off
c mem/set/b 12
variab/dec/int pass count nlinex nliney # define variable names

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1          # beam 1, propagating beam
nbeam 3 nlinex nliney data  # beam 3, history of mode shape
C 3
mem/cont
nbeam 4 nlinex nliney data  # beam 4, history of index profile
C 4
mem/cont
units/field 0 Field          # field half-width of 8 microns
variab/set units 3 units
units/set 3 units dist       # set units of history arrays
units/set 4 units dist
wavelength/set 0 Lambda*1e4 N0 # set wavelength and cladding index
C 5
mem/cont
clear 1 1
clear 3 0
clear 4 0
gaus/c/c 1 1 .00015
gaus/c/c 2 1 .00015

parabola 4 dN dN .4*Field
c set/label/on
plot/w @Name_1.plt
plot/l 4 h=.1
array
debug slabwave rdwr
slab/wave 1 4 3
plot/w @Name_2.plt
plot/l 3
peak 1
plot/w @Name_3.plt
plot/x/i fi=1 la=2 le=-.0010 ri=.0010 fmin=0

```

Jump to: [Commands](#), [Theory](#)


```

time
title mode shape vs. distance
plot/w @Name_3.plt
plot/l 4 ns=64 h=.2
plot/w @Name_4.plt
plot/c 4 ilab=0
energy 1

```

Ex87j: Double directional coupler

Ex87j demonstrates a double directional coupler.

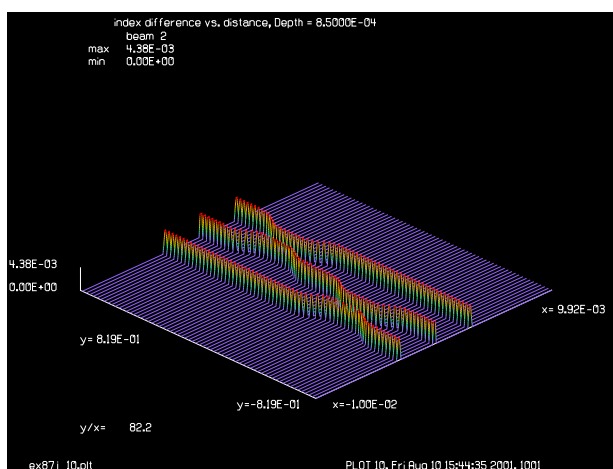


Fig. 87j.1. Index distribution for double directional coupler.

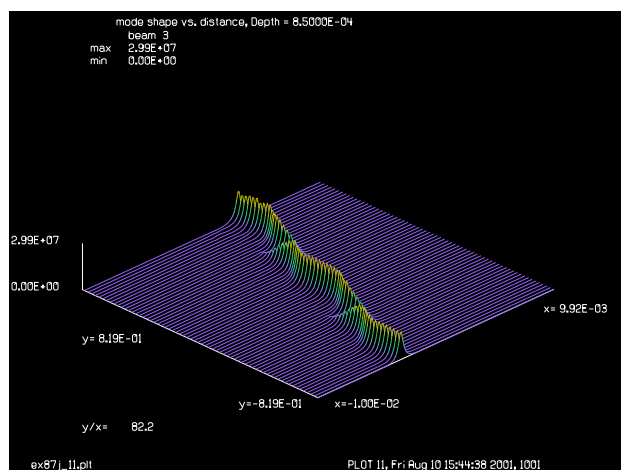


Fig. 87j.2. Propagating mode through two directional couplers.

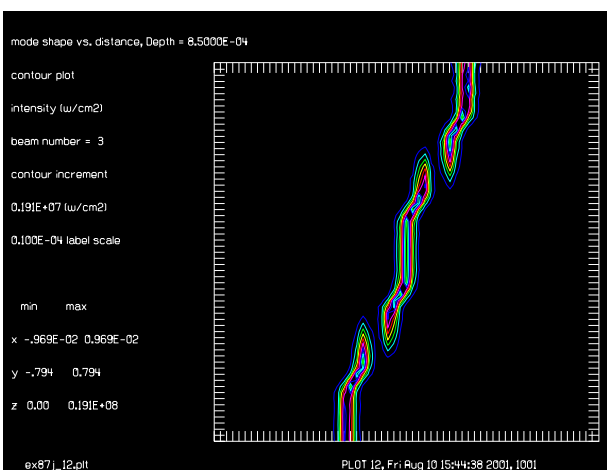


Fig. 87j.3. Propagating mode through two directional couplers.

Input: `ex87j.inp`

```

c## ex87j
#

```

Jump to: [Commands](#), [Theory](#)

```

# Example of two directional couplers in sequence
#
c set/label/off
alias Name ex87j
mem/set/b 20                                # request 3 megabytes
variab/dec/int pass count nlinex nliney Kbeam1 Ktemp # define variable names
Kbeam1 = 4                                # Extruded path
Ktemp = 6                                # Temporary beam
Kshape = 7                                # Shape to be projected
BumpWidth = .34
Smooth = .12
Decx = 30e-4
Offset1 = Decx
Offset2 = 0.
Offset3 = -Decx

Field = 100e-4
Lambda = 1.55e-4
N0 = 1.455

dist = 20e-5 # step length

Fieldy = dist*2048
alpha = 2.*pi*dist/Lambda                # phase coefficient per step
alpha=

Width = 1700*dist # was 900

Apt = 1.4e-4 # was 3.8e-4
DeltaN = .003 # .3%

dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list
Depth = 8.5e-4 # gives 100% conversion

nlinex = 256                                # propagating array width
nliney = 8192                                # length of history arrays

# beams
# 1 propagating array, nlinex x 1
# 2 index array used in extrude, nlinex x nliney
# 3 history of beam, nlinex x nliney
# 4 temporary array, nlinex x 1
# 5 temporary array, nlinex x 1
# 6 temporary array, nlinex x 1

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1                        # propagating beam
nbeam 2 nlinex nliney data                # index array
nbeam 3 nlinex nliney data                # history of mode shape
nbeam 4 nliney 1 data                     # temporary array
nbeam 5 nliney 1 data                     # temporary array
nbeam 6 nliney 1 data                     # temporary array
nbeam 7 nlinex 1 data

```

Jump to: [Commands](#), [Theory](#)

```

units/field 0 Field                # field half-width of 8 microns
variab/set Units 1 units
units/set 2 Units dist            # set units of history arrays
units/set 3 Units dist
units/set 4 dist
units/set 5 dist
units/set 6 dist
wavelength/set 0 Lambda*1e4 N0    # set wavelength and cladding index
set/density 64

```

```

variable/dec/int Kbeam1 KbeamSum
macro/def make_bump/o
#
# make a smoothed bump in Kbeam1
# Ktemp is temporary beam
# width: BumpWidth
# center: Center
# height: Deflection
# smoothing: Smooth
# offset: Offset
  clear/complex Ktemp Deflection
  clap Ktemp BumpWidth/2 xdec=Center
  convol/gaus Ktemp Smooth/2
  add/coh/con Kbeam1 Ktemp
macro/end

```

```

macro/def make_offset/o
#
# make an offset in Kbeam1
# Ktemp is temporary beam
# offset: Offset
  clear/complex Ktemp Offset
  add/coh/con Kbeam1 Ktemp
macro/end

```

```

# Make shape to be extruded, assumed the same for all paths
clear Kshape dN
irradiance Kshape
clap/c/n Kshape Apt

```

```
clear 2 0
```

```
# Make 1st path
```

```

array/set Kbeam1 nliney 1 data    # temporary array
units/set Kbeam1 dist
clear Kbeam1 0
Offset = Offset1

Center = -.36
Deflection = -Depth
macro make_bump

```

Jump to: [Commands](#), [Theory](#)

```

plot/w @Name_1.plt
plot/x/r Kbeam1

macro make_offset
plot/w @Name_2.plt
plot/x/r Kbeam1 fmin=-Field fmax=Field

transpose Kbeam1

extrude/row/path/add Kshape 2 Kbeam1
plot/w @Name_3.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 2 ns=64 h=.1

# Make 2nd path

array/set Kbeam1 nliney 1 data          # temporary array
units/set Kbeam1 dist
clear Kbeam1 0
Offset = Offset2

Center = -.36
Deflection = Depth
macro make_bump

plot/w @Name_4.plt
plot/x/r Kbeam1

Center = .36
Deflection = -Depth
macro make_bump
plot/w @Name_5.plt
plot/x/r Kbeam1

macro make_offset
plot/w @Name_6.plt
plot/x/r Kbeam1 fmin=-Field fmax=Field

transpose Kbeam1

extrude/row/path/add Kshape 2 Kbeam1
plot/w @Name_7.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 2 ns=64 h=.1

# Make 3rd path

array/set Kbeam1 nliney 1 data          # temporary array
units/set Kbeam1 dist
clear Kbeam1 0
Offset = Offset3

Center = .36
Deflection = Depth

```

Jump to: [Commands](#), [Theory](#)

```

macro make_bump

plot/w @Name_8.plt
plot/x/r Kbeam1

macro make_offset
plot/w @Name_9.plt
plot/x/r Kbeam1 fmin=-Field fmax=Field

transpose Kbeam1

extrude/row/path/add Kshape 2 Kbeam1
plot/w @Name_10.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 2 ns=64 h=.1

c set/label/on

gaus/c/c 1 1 Apt*2.6 s=1 decx=Decx      # inject gaussian mode into one core
energy/norm 1 1

time/i
slab/wave 1 2 3
time
title mode shape vs. distance, Depth = @Depth
plot/w @Name_11.plt
plot/l 3 ns=64 h=.2
plot/w @Name_12.plt
plot/c 3 ilab=0
energy 1

```

Ex87k: Three directional couplers producing a fan-out of five equal beams

Ex87k demonstrates a configuration of three directional couplers creating five beams of nearly equal intensity.

Input: ex87k.inp

```

c## ex87k
c
c Example 87k: Three directional couplers producing a fan-out
c           of five equal beams
c
alias Name ex87k
set/label/off
mem/set/b 20                      # request 3 megabytes
variab/dec/int pass count nlinex nliney Kbeam1 Ktemp # define variable names
variab/dec/int PlotNo
Kbeam1 = 4          # Extruded path
Ktemp = 6           # Temporary beam
Kshape = 7          # Shape to be projected

```

Jump to: [Commands](#), [Theory](#)

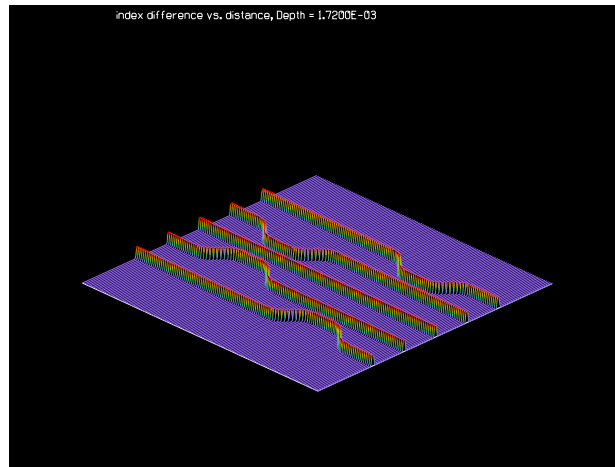


Fig. 87k.1. Index distribution for three-coupler configuration.

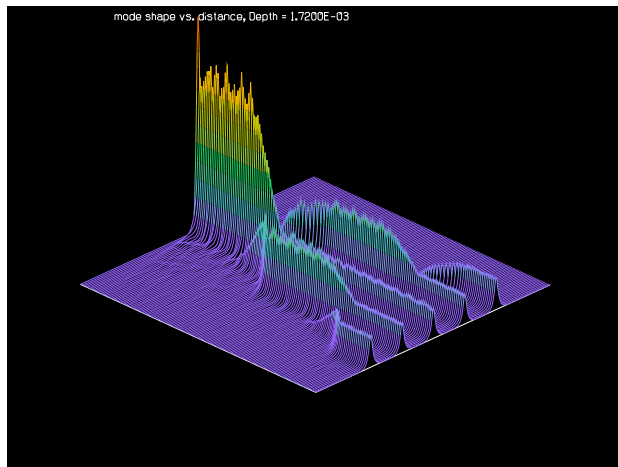


Fig. 87k.2. Propagating mode through three directional couplers.

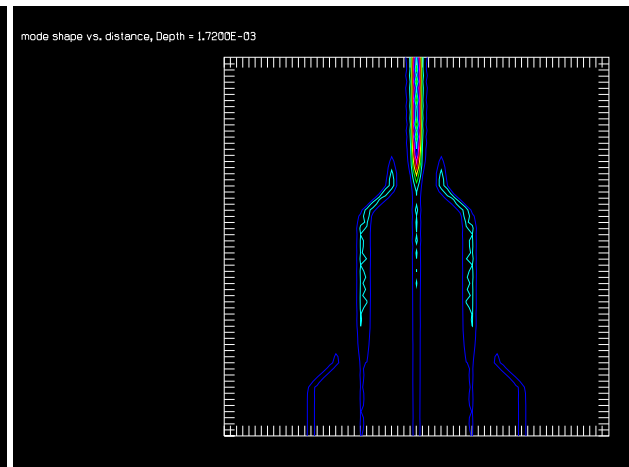


Fig. 87k.3. Five nearly equal beams created by three tuned couplers.

```

BumpWidth = .34
Smooth = .12
Separation = 32e-4
Decx = 0.
Offset1 = 2*Separation
Offset2 = Separation
Offset3 = 0.
Offset4 = -Separation
Offset5 = -2*Separation
Depth = 17.2e-4 # gives 100% conversion

```

```

Field = 120e-4
Lambda = 1.55e-4
N0 = 1.455

```

```

dist = 20e-5 # step length

```

Jump to: [Commands](#), [Theory](#)

```

Fieldy = dist*2048
alpha = 2.*pi*dist/Lambda      # phase coefficient per step
alpha=

Width = 1700*dist  # was 900

Apt = 1.4e-4      # was 1.4e-4
DeltaN = .003     # .3%

dN = sqrt(1.455^2/(1-2*DeltaN)) -N0 list

nlinex = 256      # propagating array width
nliney = 8192     # length of history arrays

PlotNo = 0
Hmax = .05

# beams
# 1  propagating array, nlinex x 1
# 2  index array used in extrude, nlinex x nliney
# 3  history of beam, nlinex x nliney
# 4  temporary array, nlinex x 1
# 5  temporary array, nlinex x 1
# 6  temporary array, nlinex x 1

c title mode shape vs. distance, two close fibers
array/s 1 nlinex 1      # propagating beam
nbeam 2 nlinex nliney data # index array
nbeam 3 nlinex nliney data # history of mode shape
nbeam 4 nliney 1 data    # temporary array
nbeam 5 nliney 1 data    # temporary array
nbeam 6 nliney 1 data    # temporary array
nbeam 7 nlinex 1 data

units/field 0 Field      # field half-width of 8 microns
variab/set Units 1 units
units/set 2 Units dist   # set units of history arrays
units/set 3 Units dist
units/set 4 dist
units/set 5 dist
units/set 6 dist
wavelength/set 0 Lambda*1e4 N0 # set wavelength and cladding index
set/density 64

variable/dec/int Kbeam1 KbeamSum
macro/def make_bump/o
#
# make a smoothed bump in Kbeam1
# Ktemp is temporary beam
# width: BumpWidth
# center: Center
# height: Deflection
# smoothing: Smooth

```

Jump to: [Commands](#), [Theory](#)

```

# offset: Offset
  clear/complex Ktemp Deflection
  clap Ktemp BumpWidth/2 xdec=Center
  convol/gaus Ktemp Smooth/2
  add/coh/con Kbeam1 Ktemp
macro/end

macro/def make_offset/o
#
# make an offset in Kbeam1
# Ktemp is temporary beam
# offset: Offset
  clear/complex Ktemp Offset
  add/coh/con Kbeam1 Ktemp
macro/end

# Make shape to be extruded, assumed the same for all paths
clear Kshape dN
irradiance Kshape
clap/c/n Kshape Apt

clear 2 0

# Make 1st path

  array/set Kbeam1 nliney 1 data          # temporary array
  units/set Kbeam1 dist
  clear Kbeam1 0
  Offset = Offset1

  Center = .36
  Deflection = -Depth
  macro make_bump
  PlotNo = PlotNo + 1
  plot/w @Name_@PlotNo.plt
  plot/x/r Kbeam1

  macro make_offset
  PlotNo = PlotNo + 1
  plot/w @Name_@PlotNo.plt
  plot/x/r Kbeam1 fmin=-Field fmax=Field

  transpose Kbeam1

  extrude/row/path/add Kshape 2 Kbeam1
  PlotNo = PlotNo + 1
  plot/w @Name_@PlotNo.plt
  title index difference vs. distance, Depth = @Depth
  plot/l/r 2 ns=128 h=Hmax

# Make 2nd path

  array/set Kbeam1 nliney 1 data          # temporary array

```

Jump to: [Commands](#), [Theory](#)


```

units/set Kbeam1 dist
clear Kbeam1 0
Offset = Offset2

Center = -.36
Deflection = -Depth
macro make_bump
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
plot/x/r Kbeam1

macro make_offset
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
plot/x/r Kbeam1 fmin=-Field fmax=Field

transpose Kbeam1

extrude/row/path/add Kshape 2 Kbeam1
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 2 ns=128 h=Hmax

# Make 3rd path

array/set Kbeam1 nliney 1 data          # temporary array
units/set Kbeam1 dist
clear Kbeam1 0
Offset = Offset3

PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
plot/x/r Kbeam1 fmin=-Field fmax=Field

transpose Kbeam1

extrude/row/path/add Kshape 2 Kbeam1
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 2 ns=128 h=Hmax

# Make 4th path
array/set Kbeam1 nliney 1 data          # temporary array
units/set Kbeam1 dist
clear Kbeam1 0
Offset = Offset4

Center = -.36

```

Jump to: [Commands](#), [Theory](#)

```

Deflection = Depth
macro make_bump
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
plot/x/r Kbeam1

macro make_offset
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
plot/x/r Kbeam1 fmin=-Field fmax=Field

transpose Kbeam1

extrude/row/path/add Kshape 2 Kbeam1
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 2 ns=128 h=Hmax

# Make 5th path
array/set Kbeam1 nliney 1 data          # temporary array
units/set Kbeam1 dist
clear Kbeam1 0
Offset = Offset5

Center = .36
Deflection = Depth
macro make_bump
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
plot/x/r Kbeam1

macro make_offset
plot/w @Name_@PlotNo.plt
plot/x/r Kbeam1 fmin=-Field fmax=Field

transpose Kbeam1

extrude/row/path/add Kshape 2 Kbeam1
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
title index difference vs. distance, Depth = @Depth
plot/l/r 2 ns=128 h=Hmax

c set/label/on

gaus/c/c 1 1 Apt*2.6 s=1 decx=Decx      # inject gaussian mode into one core
energy/norm 1 1

time/i
slab/wave 1 2 3

```

Jump to: [Commands](#), [Theory](#)

```
time
title mode shape vs. distance, Depth = @Depth
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
plot/l 3 ns=128 h=1
PlotNo = PlotNo + 1
plot/w @Name_@PlotNo.plt
plot/c 3 ilab=0
energy 1
```


Ex88: Speckle smoothing with a lens array integrator

Table. 88.1. Table of Ex88 examples

Ex88a: Perfect pupil with lensarray.	2
Ex88b: Perfect pupil with lensarray, interference pattern	4
Ex88c: Independent random phase plates or random amplitude with lens array	6
Ex88d: Random phase grating, lens array, mirror dither.	9
Ex88e: Random phase grating, lens array, circular motion.	11
Ex88f: Calculation of speckle smoothing by linear motion	13
Ex88g: Smoothing of diffraction ringing by partial coherence	18

This example illustrates a variety of issues in speckle smoothing of an lens array integrator. A lenslet array may be used to homogenize an optical beam by overlapping different areas of the pupil. Fig. 88.1 shows a typical configuration. The elements of the lens array, in combination with a recollimating lens, create overlapping pupils at a target plane. The overlapping beams are formed, as shown in Fig. 88.2, with different tilt angles based on the chief ray angle of each element as it arrives at the target plane. Since the overlapping pupils add coherently, a strong beat frequency or interference fringe is formed. The interference pattern is complex because all lenslet elements interfere with each other. The fringe contrast is objectionable. Fringe contrast may be significantly reduced by using a moving aberration plate in front of the lenslet array. The target plane is not conjugate to the plane of the lenslet array but, because of the very large beam diameter, the effective Fresnel number is very high and edge diffraction and diffraction effects other than the beat frequency due to chief ray angles may be ignored.

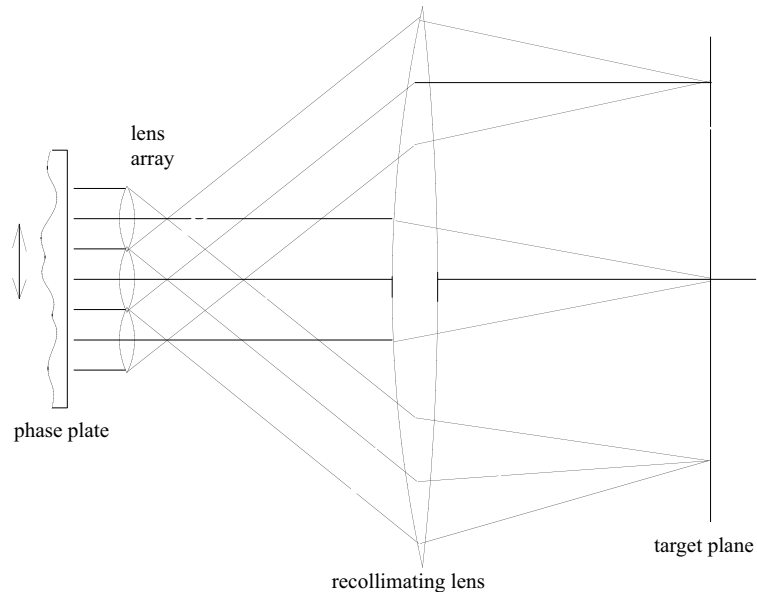


Fig. 88.1. A lens array is positioned in an optical beam. A phase plate is used to reduce the spatial coherence. A recollimating lens results in the beams overlapping at the target plane. The target plane is not conjugate to the plane of the phase plate. However, the effects of propagation are negligible at the target plane. The principal diffraction effect is due to the relative angles of the overlapping beam determined by the angle of the chief rays from the various apertures.

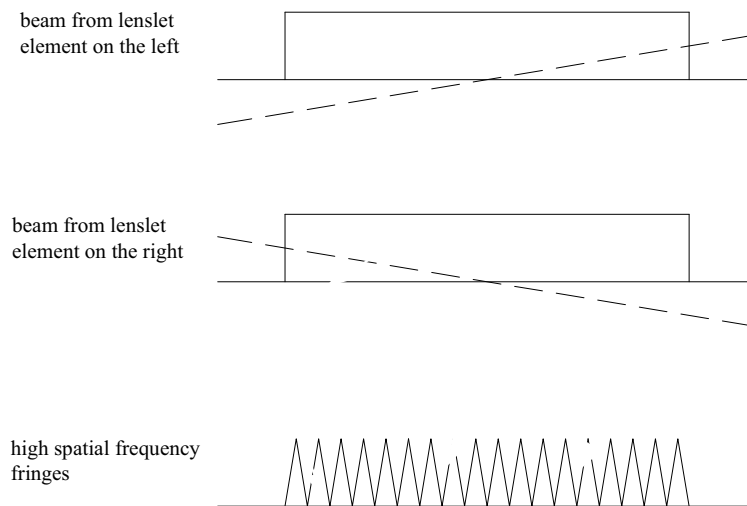


Fig. 88.2. The overlapping pupils from the lens array elements arrive at the target plane with angles corresponding to the angles of the chief rays. High spatial frequency fringes are produced, which are 100% modulated.

Ex88a: Perfect pupil with lensarray

Figure 88.3 shows the pupils of a 10 x 10 array of lenses. In this example, the diameter of each element is assumed to be 0.4 cm and the focal length of the recollimating lens is assumed to be 30 cm. The wavelength is assumed to be 0.254 micron.

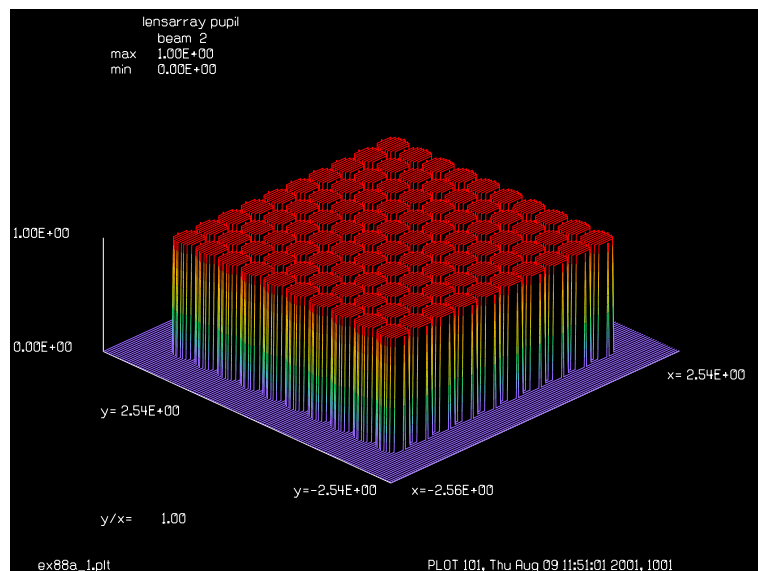


Fig. 88.3. Array of 10 x 10 lens elements, as calculated by Ex88a.inp.

Input: ex88a.inp

c## ex88a

Jump to: [Commands](#), [Theory](#)

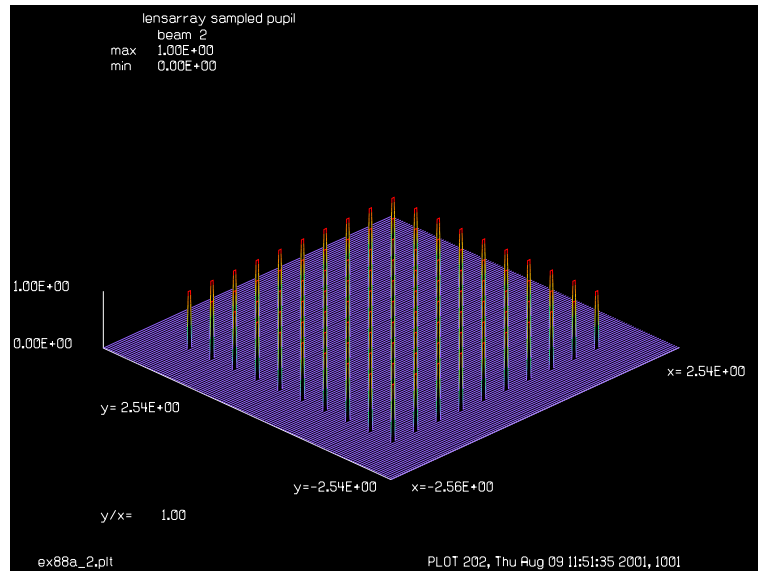


Fig. 88.4. Samples of the center of the lens elements in the 10 x 10 array used to determine the beat frequency, as calculated by Ex88a.inp.

```

c
c Perfect pupil with lensarray, pupil distribution
c
variab/dec/int switch
mem/set/b 2
array/s 1 256
nbeam 2 256
lambda = .254e-4
wavelength/set 0 1e4*lambda
units/s 1 .02
units/s 2 .02
plot/screen/pause 1
macro/def outer/o
    y = y + dy
    x = -2.2
    mac inner/10
macro/end
macro/def inner/o
    x = x + dx
    clear 1 1
    clap/c/c 1 clap xdec=x ydec=y
    add/coh/con 2 1
    if switch = 1 normalize 2
    plot/watch plot1.plt
    plot/l 2 ns=128 h=.2
macro/end
dx = .4
dy = .4
y = -2.2
clap = dx/2.
switch = 1
clear 2 0.

```

Jump to: [Commands](#), [Theory](#)

```

mac outer/10
plot/watch ex88a_1.plt
title lensarray pupil
plot/1 2 min=0 ns=128
y = -2.2
clap = .02
switch = 0.
clear 2 0.
mac outer/10
plot/watch ex88a_2.plt
title lensarray sampled pupil
plot/1 2 min=0 ns=128 h=.2

```

Ex88b: Perfect pupil with lensarray, interference pattern

The observation region corresponds to an essentially infinitesimal region in each of the lens elements. To build the interference pattern at the target plane, single points are determined for each lens element. Then a uniform amplitude is specified for the target plane based on the complex amplitude of the center point of the corresponding lenslet element. The tilt value is then added and the complex amplitudes from all lens elements are summed coherently.

The largest feature size of the interference pattern at the target plane is found by dividing the wavelength by the angle between the centers of neighboring elements.

$$\text{largest feature size} = \lambda/\theta = 19 \mu, \text{ where } \theta = 0.4/30 \quad (88.1)$$

Interference between the extreme elements create feature sizes of about 4μ , but all fringe patterns add coherently as shown in Fig. 88.5. To resolve the finest features, the units must be about 0.000032 cm at the target plane. With a 128×128 array we can sample only about $0.004 \times 0.004 \text{ cm}$.

Input: ex88b.inp

```

c## ex88b
c
c Perfect pupil with lensarray, interference pattern
c
set/alias_stop/off
mem/set/b 2
array/s 1 256
nbeam 2 128
nbeam 3 128
clear 3 0
lambda = .254e-4
wavelength/set 0 1e4*lambda
units/s 1 .02
units/s 2 .000032
units/s 3 .000032
plot/screen/pause 4
macro/def scan/o
    y = -2.2
    clear 3 0.

```

Jump to: [Commands](#), [Theory](#)

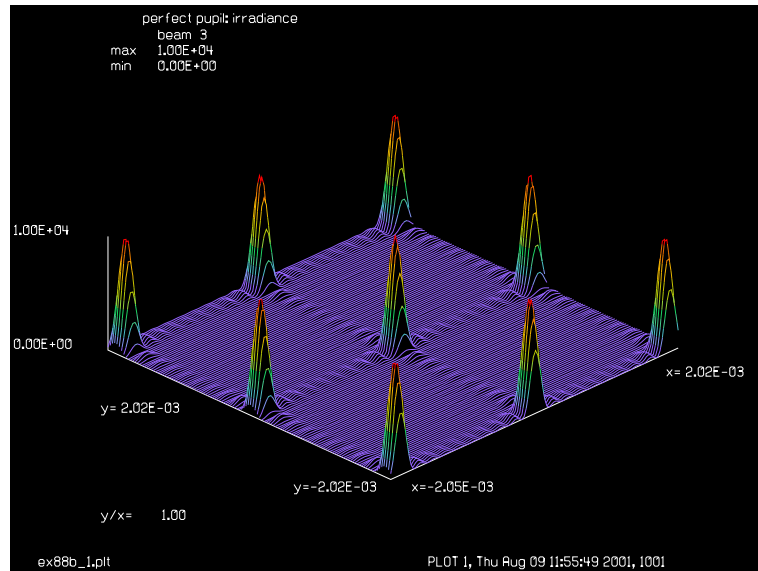


Fig. 88.5. The 10 x 10 array of lens elements produce crossed interference fringes which add coherently. This display shows a region of about 40 x 40 microns. Peaks appear at the crossing points. The entire target plane is covered with this pattern. The fringes are separated by 19 microns. See Ex88b.inp.

```

mac outer/10
macr/end
macro/def outer/o
  y = y + dy
  x = -2.2
  mac inner/10
macro/end
macro/def inner/o
  x = x + dx
  point/list/xy 1 x y
  variab/set sr point/sr
  variab/set si point/si
  clear 2 1
  mult/complex 2 sr si
  azimuth = 180*atan2(y,x)/pi + 90
  rtilt = sqrt(x^2 + y^2)/focallength
  abr/tilt 2 rtilt/lambda azimuth rn=1
  add/coh/con 3 2
macro/end
dx = .4
dy = .4
focallength = 30
mac scan/1
plot/watch ex88b_1.plt
title perfect pupil: irradiance
plot/1 3 min=0 ns=128

```

Ex88c: Independent random phase plates or random amplitude with lens array

Adding a strong phase aberration plate, as illustrated in Fig. 88.1, breaks up the regular peaks as shown in Fig. 88.6. Random phase of 0.5 waves RMS and autocorrelation width of 0.3 cm at the 1/e points was used. While Fig. 88.6 is much better than Fig. 88.5, the irradiance is still very nonuniform. We can measure irradiance nonuniformity by

$$\text{nonuniformity (percent)} = \frac{1}{\langle I \rangle} (\langle I^2 \rangle - \langle I \rangle^2)^{1/2} \times 100 \quad (88.2)$$

where $\langle \rangle$ indicates a spatial average and the irradiance is I . This definition of irradiance nonuniformity can give a value greater than 100%, A random speckle pattern such as is approximated in Fig. 88.6 has a nonuniformity of about 100%.

The irradiance pattern may be improved by incoherently summing the speckle due to different random phase distributions. Figures 88.7 and 88.8 show the results of using 100 different phase plates, and incoherently summing the results. The nonuniformity is reduced by a factor of 10 in agreement with the $1/\sqrt{N}$ rule, where N is the number of independent phase plates. Using 10,000 samples should give 1% nonuniformity.

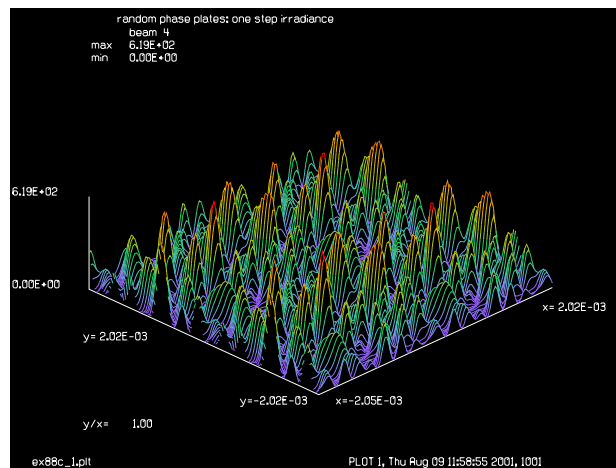


Fig. 88.6. Use of a stationary aberration plate breaks up the fringe pattern into a speckle distribution. This distribution was calculated by Ex88c.inp.

Input: ex88c.inp

```
c## ex88c
c
c Example 88c: Independent random phase plates or random amplitude
c             with lens array
c
c Set switch = 0 for random phase plates
c Set switch = 1 for smoothed complex amplitude noise similar to
c excimer laser
c
set/alias_stop/off
variab/dec/int ntimes switch
```

Jump to: [Commands](#), [Theory](#)

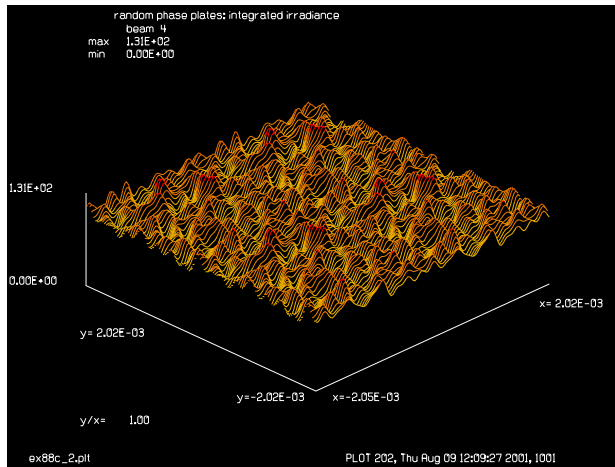


Fig. 88.7. Integrated irradiance for 100 different random phase plates, as calculated in Ex88c.inp.

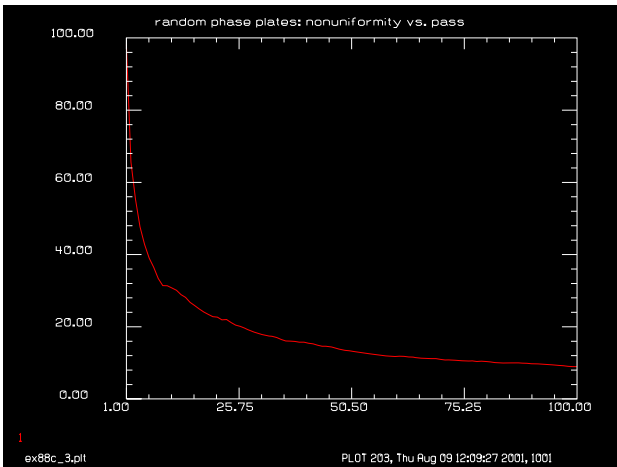


Fig. 88.8. Nonuniformity in percent vs. number of different random phase plates. Agrees with $1/\sqrt{N}$, as calculated in Ex88c.inp. After 100 steps the nonuniformity is improved by a factor of ten.

```

mem/set/b 2
array/s 1 256
nbeam 2 128
nbeam 3 128
nbeam 4 128
clear 3 0
lambda = .254e-4
wavelength/set 0 1e4*lambda
units/s 1 .02
units/s 2 .000032
units/s 3 .000032
units/s 4 .000032
clear 4 0
plot/screen/pause 4
variab/dec/int pass
switch = 0          # set to 0, random phase plates, 1 for excimer laser
if switch = 0 alias type phase,plates
if switch = 1 alias type amplitude
macro/def scan/o
    pass = pass + 1
    y = -2.2
    time = time + dt
    if switch = 0 then
c use this logic for random phase plate
        clear 1 1
        phase/ran 1 .5 .15
    endif
    if switch = 1 then
c use this logic for random complex amplitude similar to excimer laser
        clear 1 0
        noise 1 1
        convol/gaus 1 .15
    endif
endif

```

Jump to: [Commands](#), [Theory](#)

```

clear 3 0.
mac outer/10
add/inc/con 4 3
if pass = 1 then
  plot/watch ex88c_1.plt
  title random @type: one step irradiance
  plot/1 4 min=0 ns=128
endif
plot/watch plot1.plt
title random @type: integrated irradiance
plot/1 4 min=0 ns=128
uniform 4
variab/set uniform uniform
udata/set pass pass uniform
plot/watch plot2.plt
title random @type: nonuniformity vs. pass
plot/udata min=0 max=100
macr/end
macro/def outer/o
  y = y + dy
  x = -2.2
  mac inner/10
macro/end
macro/def inner/o
  x = x + dx
  point/list/xy 1 x y
  variab/set sr point/sr
  variab/set si point/si
  clear 2 1
  mult/complex 2 sr si
  azimuth = 180*atan2(y,x)/pi + 90
  rtilt = sqrt(x^2 + y^2)/focallength
  abr/tilt 2 rtilt/lambda azimuth rn=1
  add/coh/con 3 2
macro/end
dx = .4
dy = .4
focallength = 30
amplitude = 1.0
ntimes = 100
dt = 1/ntimes
time = -dt
mac scan/ntimes
mult 4 .01
plot/watch ex88c_2.plt
title random @type: integrated irradiance
plot/1 4 min=0 ns=128
plot/watch ex88c_3.plt
title random @type: nonuniformity vs. pass
plot/udata min=0 max=100

```

Ex88d: Random phase grating, lens array, mirror dither.

We can approximate independent phase plates by moving a single plate. If we move the plate by one phase autocorrelation width, we will have an effectively independent phase distribution. Figure 88.9 shows how a phase region is moved by harmonic motion due to mirror dither. The harmonic motion retraces itself so it is two-fold redundant. In addition the sample spacing is nonuniform, so samples may be too widely spaced in the center of motion and too closely spaced at the end of the motion. Figures 88.10 and 88.11 show the nonuniformity due to 20 samples of mirror dither. The range of the dither was 2 cm peak-to-valley. The irradiance nonuniformity is reduced only to about 50%, much higher than the expected value of 22% expected for 20 independent phase plates.

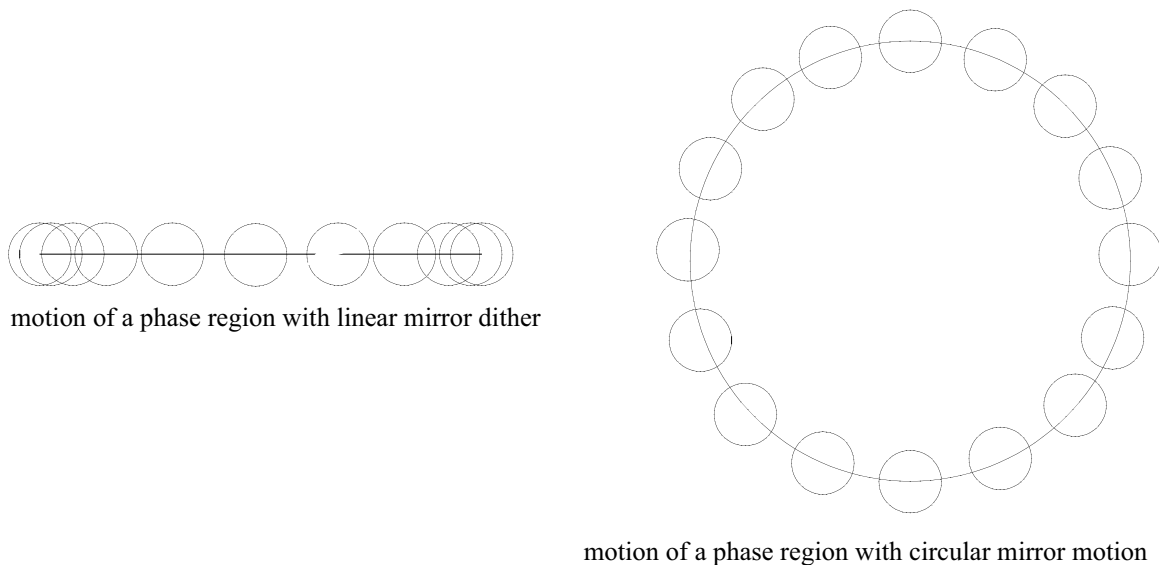


Fig. 88.9. A vibrating mirror may be used to scan the aberration across the lenslet array, as used in Ex88d.inp. The linear harmonic motion is not perfectly efficient because each position is repeated on the return pass and positions at the extreme part of the motion overlap so that these points do not generate independent aberration. Circular motion, as shown on the right and as implemented in Ex88e.inp, generates nearly independent aberration for all points separated by an autocorrelation distance.

Input: ex88d.inp

```
c## ex88d
c
c Example 88d: Random phase grating, lens array, mirror dither
c
variab/dec/int ntimes
mem/set/b 2
array/s 1 256
nbeam 2 128
nbeam 3 128
nbeam 4 128
nbeam 5 256
```

Jump to: [Commands](#), [Theory](#)

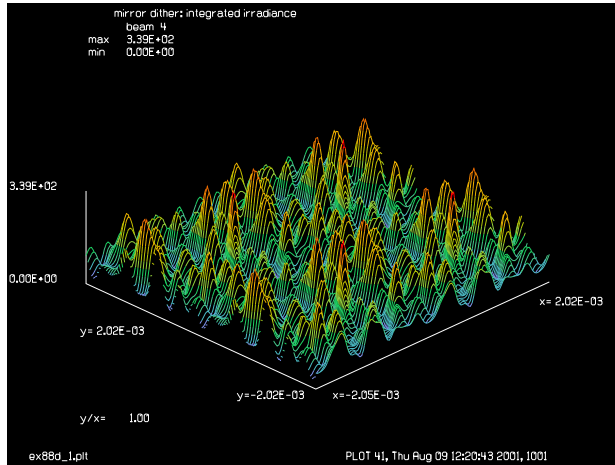


Fig. 88.10. Integrated irradiance for mirror dither with 20 sample points of the motion. Smoothing is not as good as 20 random samples because of two-fold redundancy and sample overlaps at the ends of the motion.

```
clear 3 0
lambda = .254e-4
wavelength/set 0 1e4*lambda
units/s 1 .02
units/s 2 .000032
units/s 3 .000032
units/s 4 .000032
units/s 5 .02
phase/ran 5 .5 .15
clear 4 0
plot/screen/pause 4
variab/dec/int pass
macro/def scan/o
    pass = pass + 1
    y = -2.2
    time = time + dt
    copy 5 1
    shift = amplitude*sin(2*pi*time) list
    shift 1 shift
    clear 3 0.
    mac outer/10
    mult 4 [1.-1./pass]
    mult 3 [1./pass]
    add/inc/con 4 3
    plot/watch plot1.plt
    title mirror dither: integrated irradiance
    plot/1 4 min=0 ns=128
    uniform 4
    variab/set uniform uniform
    udata/set pass pass uniform
    plot/watch plot2.plt
    title mirror dither: nonuniformity vs. pass
    plot/udata 1 1 min=0 max=100
```

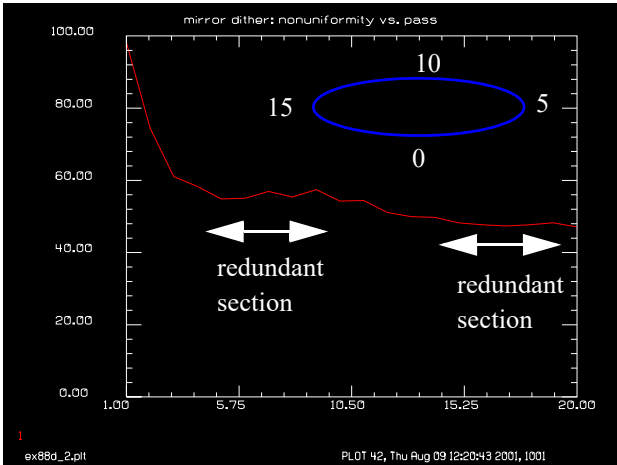


Fig. 88.11. Nonuniformity in percent vs. number of samples of mirror dither. The mirror retraces each sample point and end points are not independent so the improvement does not follow $1/\sqrt{N}$.

```

macr/end
macro/def outer/o
  y = y + dy
  x = -2.2
  mac inner/10
macro/end
macro/def inner/o
  x = x + dx
  point/list/xy 1 x y
  variab/set sr point/sr
  variab/set si point/si
  clear 2 1
  mult/complex 2 sr si
  azimuth = 180*atan2(y,x)/pi + 90
  rtilt = sqrt(x^2 + y^2)/focallength
  abr/tilt 2 rtilt/lambda azimuth rn=1
  add/coh/con 3 2
macro/end
dx = .4
dy = .4
focallength = 30
amplitude = 1.0
ntimes = 20
dt = 1/ntimes
time = -dt
mac scan/ntimes
plot/watch ex88d_1.plt
title mirror dither: integrated irradiance
plot/1 4 min=0 ns=128
plot/watch ex88d_2.plt
title mirror dither: nonuniformity vs. pass
plot/udata 1 1 min=0 max=100

```

Ex88e: Random phase grating, lens array, circular motion.

Figure 88.9 shows a circular sample distribution achieved by nutating the mirror. It is actually mechanically easier to nutate a mirror (which is done simply by tilting a mirror slightly and then rotating it at high speed) than it is to dither it, because unlike dithering with nutation there is no change of momentum. Figures 88.12 and 88.13 show the results of rotation in 20 steps with a nutation diameter of 2 cm, resulting in about 0.31 cm between sample points. The 0.31 step size is comparable to the 0.3 cm autocorrelation diameter so the steps result in nearly, but not exactly, independent phase at each step.

Figure 88.13 shows that the nonuniformity is 28% with 20 steps—not as good as the 22% for the strictly uncorrelated steps but much better than the 50% nonuniformity for linear motion. Presumably any type of motion that gave 10,000 independent samples would improve the nonuniformity to 1%.

Input: ex88e.inp

```

c## ex88e
c
c Random phase grating, lens array, circular motion

```

Jump to: [Commands](#), [Theory](#)

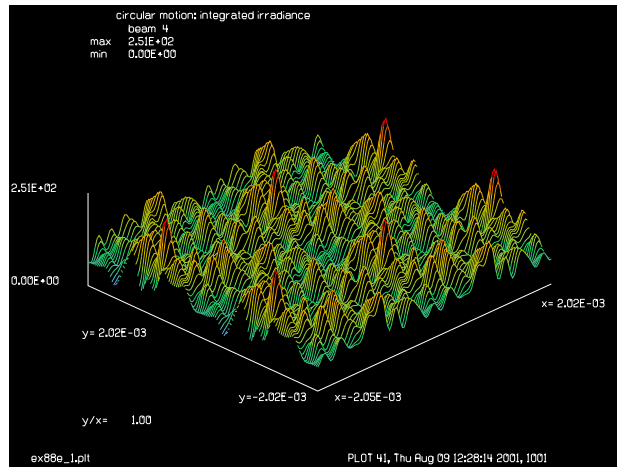


Fig. 88.12. Integrated irradiance for mirror nutation to give a circular motion with 20 sample points. Sample point separation is about 0.31 cm—about the phase autocorrelation diameter. Smoothing is nearly as good as for 20 different phase plates.

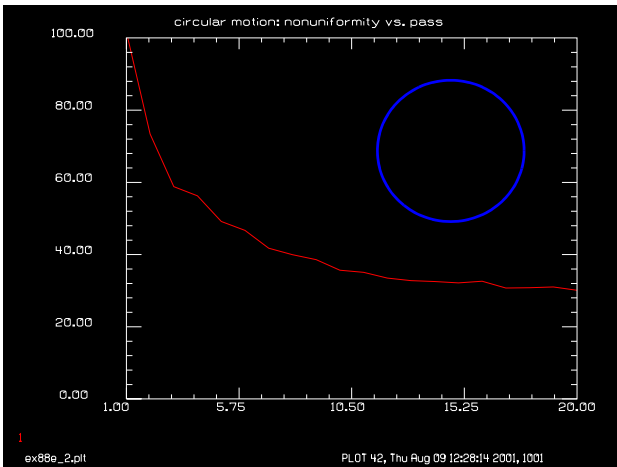


Fig. 88.13. Nonuniformity in percent vs. number of samples of mirror nutation to give circular sampling. Sample points are nearly independent and nearly follow $1/\sqrt{N}$. The nonuniformity is 28% after 20 samples as compared with the expected value of 22% for 20 completely independent samples.

```

C
variab/dec/int ntimes
mem/set/b 2
array/s 1 256
nbeam 2 128
nbeam 3 128
nbeam 4 128
nbeam 5 256
clear 3 0
lambda = .254e-4
wavelength/set 0 1e4*lambda
units/s 1 .02
units/s 2 .000032
units/s 3 .000032
units/s 4 .000032
units/s 5 .02
phase/ran 5 .5 .15
clear 4 0
plot/screen/pause 4
variab/dec/int pass
macro/def scan/o
    pass = pass + 1
    y = -2.2
    time = time + dt
    copy 5 1
    shift 1 amplitude 360*time
    clear 3 0.
    mac outer/10
    mult 4 [1.-1./pass]
    mult 3 [1./pass]
    add/inc/con 4 3

```

Jump to: [Commands](#), [Theory](#)


```

plot/watch plot1.plt
title circular motion: integrated irradiance
plot/1 4 min=0 ns=128
uniform 4
variab/set uniform uniform
udata/set pass pass uniform
plot/watch plot2.plt
title circular motion: nonuniformity vs. pass
plot/udata 1 1 min=0 max=100
udata/list
pause 4
macr/end
macro/def outer/o
  y = y + dy
  x = -2.2
  mac inner/10
macro/end
macro/def inner/o
  x = x + dx
  point/list/xy 1 x y
  variab/set sr point/sr
  variab/set si point/si
  clear 2 1
  mult/complex 2 sr si
  azimuth = 180*atan2(y,x)/pi + 90
  rtilt = sqrt(x^2 + y^2)/focallength
  abr/tilt 2 rtilt/lambda azimuth rn=1
  add/coh/con 3 2
macro/end
dx = .4
dy = .4
focallength = 30
amplitude = 1.0
ntimes = 20
dt = 1/ntimes
time = -dt
mac scan/ntimes
plot/watch ex88e_1.plt
title circular motion: integrated irradiance
plot/1 4 min=0 ns=128
plot/watch ex88e_2.plt
title circular motion: nonuniformity vs. pass
plot/udata 1 1 min=0 max=100

```

Ex88f: Calculation of speckle smoothing by linear motion

In Example 88f a linear motion is used to smooth a speckle distribution and both the nonuniformity of irradiance and the autocorrelation function of the irradiance are calculated. This example illustrates several points:

Jump to: [Commands](#), [Theory](#)

1) Reduction in nonuniformity follows inverse square root N , where N is the number of independent speckles traversed. In fact, any trajectory for smoothing may be used, including complex curves. Multiple crossings of a speckle do not contribute to nonuniformity reduction.

2) The effect of smoothing direction is to increase the speckle size in that direction.

3) The autocorrelation of the irradiance provides a good description of the effects.

The `autocorrelation` command calculates the autocorrelation of the complex amplitude. By first transforming the array to irradiance form with the `irradiance` command, the autocorrelation of the irradiance is calculated. The departure of the autocorrelation function of the speckle irradiance from the autocorrelation function from the ideal pupil, when measured at a distance greater than the typical speckle size gives the asymptotic value of $1/\sqrt{N}$.

For this example, the noise amplitude is smoothed by a gaussian function with of 10 between the $1/e$ points. The speckle pattern is shifted and summed over a distance of 20, covering about 3 speckles, as measured between the full width half max (FWHM).

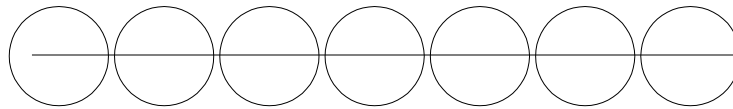


Fig. 88.14. Linear motion which is crossing a number of speckles.

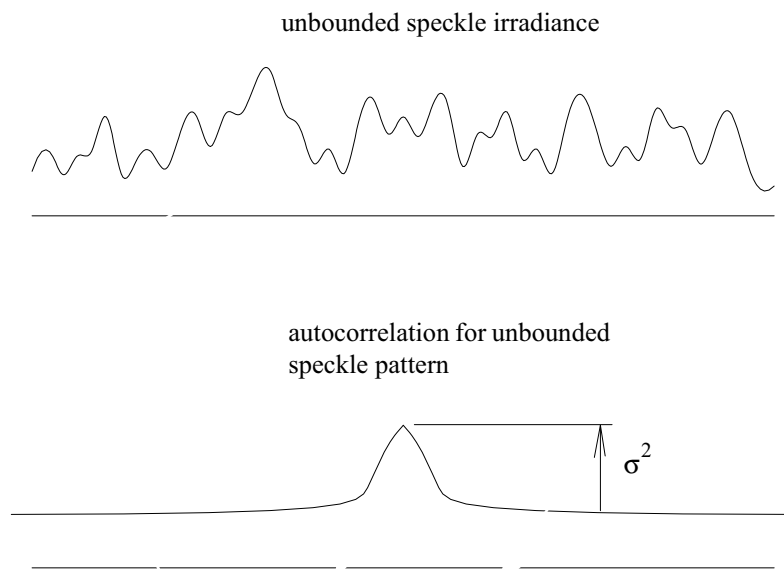


Fig. 88.15. The autocorrelation function of an unbounded speckle pattern has a DC level determined by the σ^2 , the standard deviation of irradiance nonuniformity and a bump in the center which is determined by the typical speckle size.

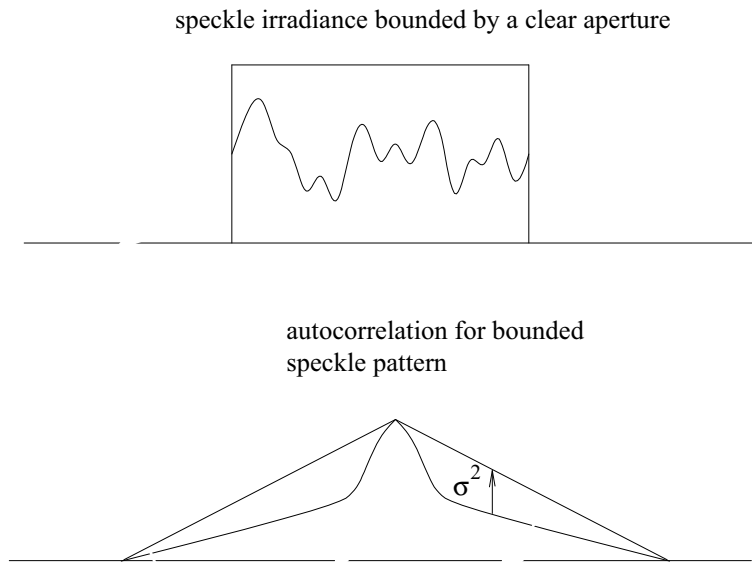


Fig. 88.16. For a finite size clear aperture the irradiance nonuniformity manifests itself as a drop from the autocorrelation of the uniformly filled aperture.

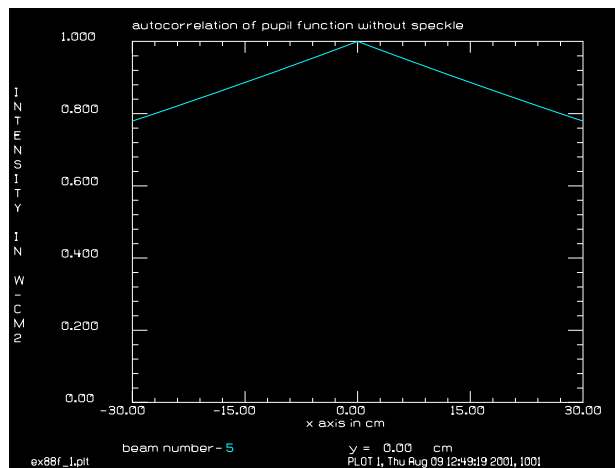


Fig. 88.17. Autocorrelation function of ideal pupil of 128 x 128 uniform irradiance. Shown over 20 x 30 region.

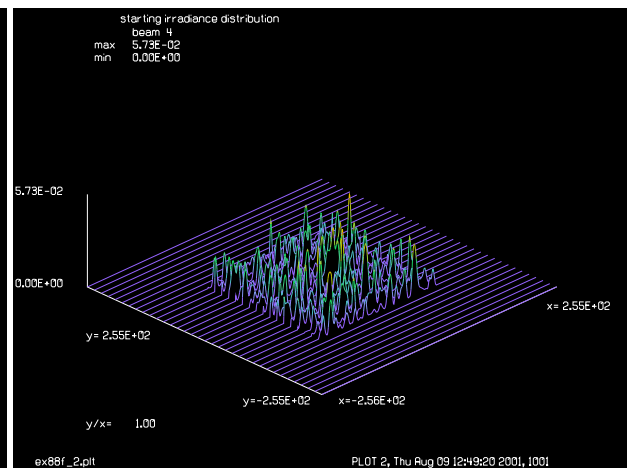


Fig. 88.18. Noise distribution after convolution with 5 cm radius gaussian. Region is 128 x 128.

Input: `ex88f.inp`

```

c## ex88f
c
c Calculation of speckle smoothing by linear motion
c
variab/dec/int ntimes
mem/set/b 6
c
c beam 1 starting random speckle pattern

```

Jump to: [Commands](#), [Theory](#)

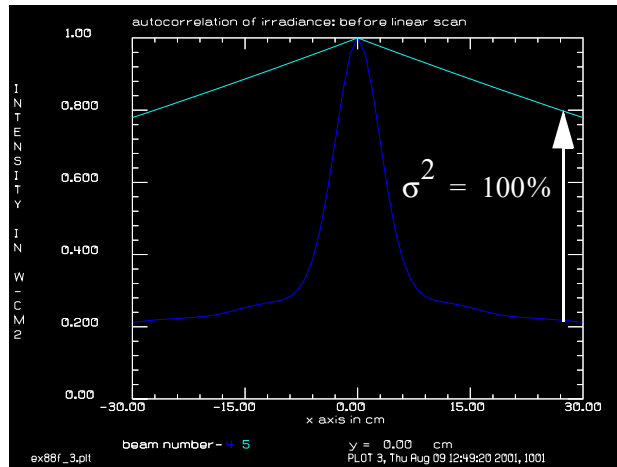


Fig. 88.19. Autocorrelation function for starting, smoothed noise distribution.

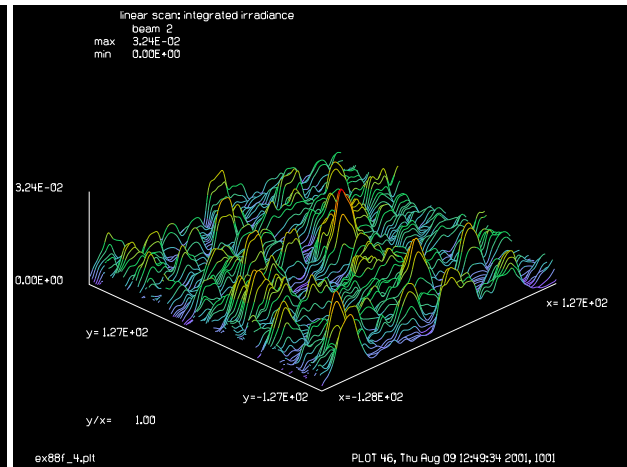


Fig. 88.20. Irradiance pattern after linear scan of 20 length in the x-direction.

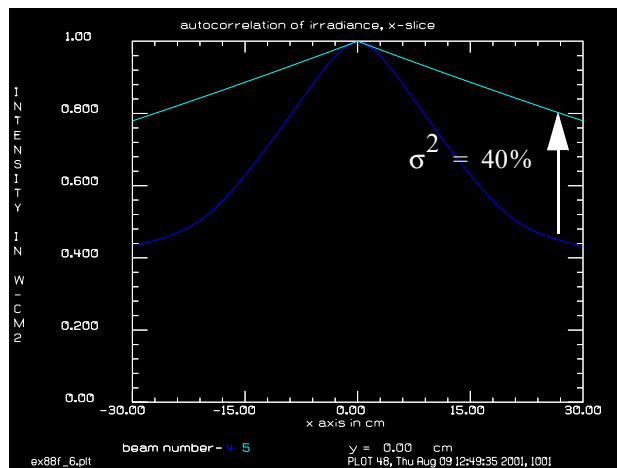


Fig. 88.21. Autocorrelation function (x-direction) after linear scan of 20 in the x-direction. Note considerable increase in the “bump” indicating larger speckles. Irradiance nonuniformity is improved according to inverse square root law.

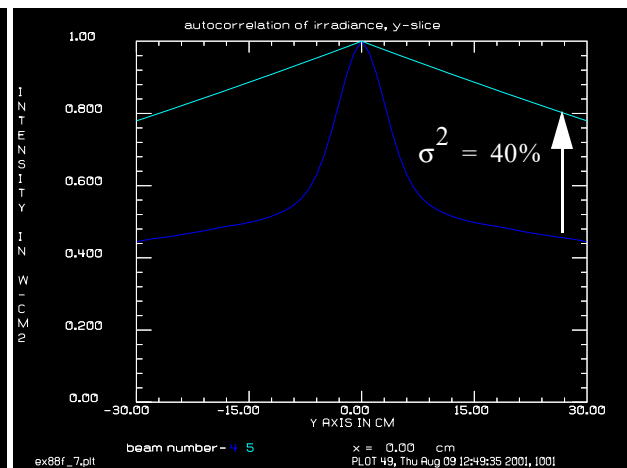


Fig. 88.22. Autocorrelation function (y-direction) after linear scan of 20 in the x-direction. Note modest increase in the “bump” indicating larger speckles.

```

c beam 2      shifted speckle pattern
c beam 3      integrated irradiance pattern
c beam 4      autocorrelation of integrated irradiance from speckle
c beam 5      autocorrelation of aperture function
c
array/s 1 256
nbeam 3 256
nbeam 4 512
nbeam 5 512
lambda = .254e-4
wavelength/set 0 1e4*lambda
units/s 0 1
clear 5 0

```

Jump to: [Commands](#), [Theory](#)

```

copy/con 3 5
autocorrelation 5          # calculate autocorrelation function of irradiance
title autocorrelation of pupil function without speckle
plot/watch ex88f_1.plt
plot/x/i 5 le=-30 ri=30 fmin=0
clear 3 0
c
c  Create speckle pattern with autocorrelation radius of 5
c
noise 3 1
convol/gaus 3 5
clear 4 0
copy/con 3 4
title starting irradiance distribution
plot/watch ex88f_2.plt
plot/l 4
irradiance 4
c
c  Calculate autocorrelation function of irradiance before scan
c
autocorrelation 4
title autocorrelation of irradiance: before linear scan
plot/watch ex88f_3.plt
plot/x/i le=-30 ri=30 fi=4 la=5 fmin=0
clear 2 0
variab/dec/int pass
macro/def scan/o
    pass = pass + 1
    x = x + dx list
    copy 3 1
    shift 1 x 90
    mult 2 [1.-1./pass]
    mult 1 [1./pass]
    add/inc/con 2 1
    plot/watch plot1.plt
    title linear scan: integrated irradiance
    plot/l 2 min=0 ns=128
    uniform 2
    variab/set uniform uniform
    udata/set pass pass uniform
    plot/watch plot2.plt
    title linear scan: nonuniformity vs. pass
    plot/udata 1 1 min=0 max=100
macr/end
amplitude = 10
ntimes = 21
dx = 2*amplitude/(ntimes-1) list
x = -(amplitude+dx)
mac scan/ntimes
plot/watch ex88f_4.plt
title linear scan: integrated irradiance
plot/l 2 min=0 ns=128
plot/watch ex88f_5.plt
title linear scan: nonuniformity vs. pass

```

Jump to: [Commands](#), [Theory](#)

```

plot/udata 1 1 min=0 max=100
clear 4 0
copy/con 2 4
irradiance 4
autocorrelation 4
plot/watch ex88f_6.plt
title autocorrelation of irradiance, x-slice
plot/x/i le=-30 ri=30 fi=4 la=5 fmin=0
plot/watch ex88f_7.plt
title autocorrelation of irradiance, y-slice
plot/y/i le=-30 ri=30 fi=4 la=5 fmin=0
c
c About 3 speckles should produce about 57% nonuniformity
c
udata/list 21          # display value of nonuniformity

```

Ex88g: Smoothing of diffraction ringing by partial coherence

In Example 88g, an LED's with M^2 values of 60 and 15 illuminate a square aperture. The light is then propagated a short distance to form a high Fresnel diffraction pattern. The partially coherent illumination reduces the region of the aperture that acts coherently and therefore yields diffraction somewhat like a small region with soft edges. The number of independent instances determines the residual speckle by the square root N rule. Ten thousand instances leave about 1% residual error.

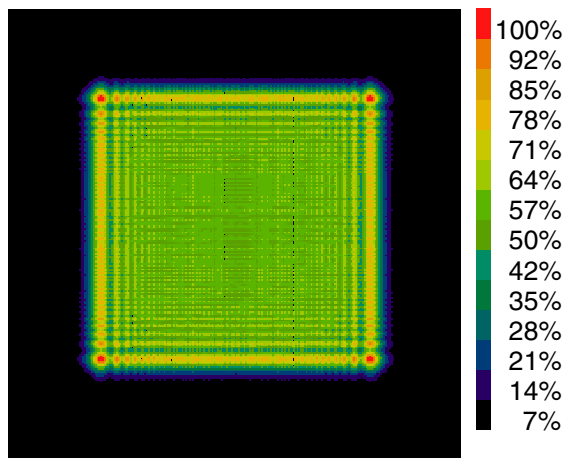


Fig. 88.23. Diffraction pattern for fully coherent uniform illumination of a square aperture after propagation a short distance.

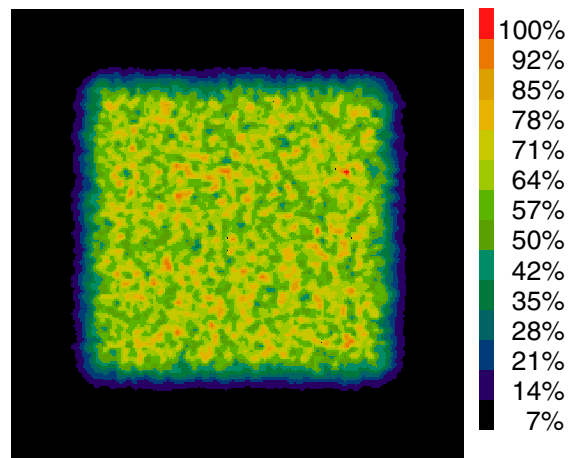


Fig. 88.24. Diffraction distribution of $M^2 = 60$ after 50 instances showing that the characteristic speckle size for $M^2 = 60$. With 50 instances, the nonuniformity is about $1/\sqrt{50} \approx 14\%$.

Input: `ex88g.inp`

```
c## ex88g.inp
```

```
c
```

Jump to: [Commands](#), [Theory](#)

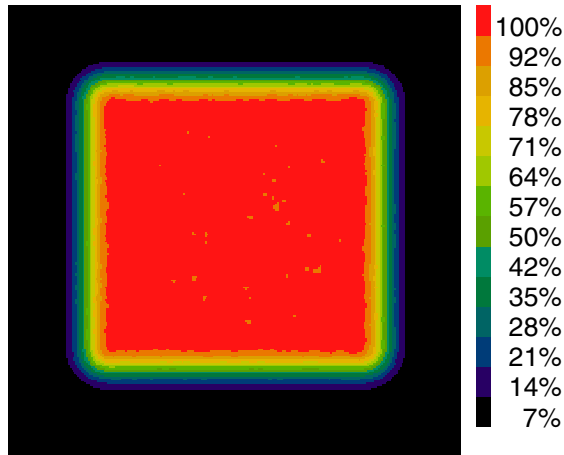


Fig. 88.25. Diffraction distribution of $M^2 = 60$ after 10,000 instances showing that the diffraction ringing has been almost completely smoothed out. The residual nonuniformity is on the order of $1/\sqrt{10,000} \approx 1\%$.

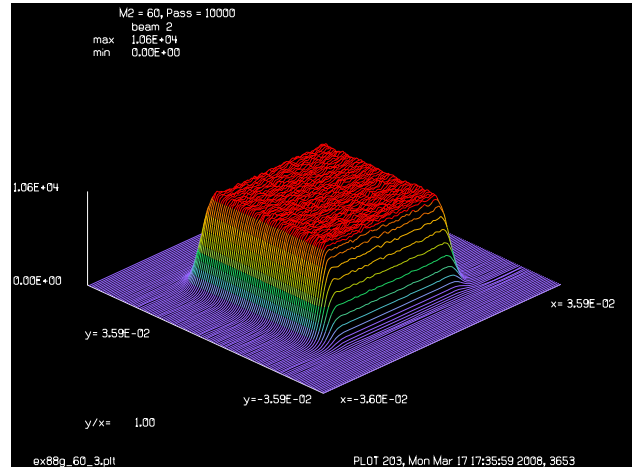


Fig. 88.26. Diffraction distribution of $M^2 = 60$ after 10,000 instances showing that the diffraction ringing has been almost completely smoothed out. Note that coherent “rabbit ears” are almost completely absent.

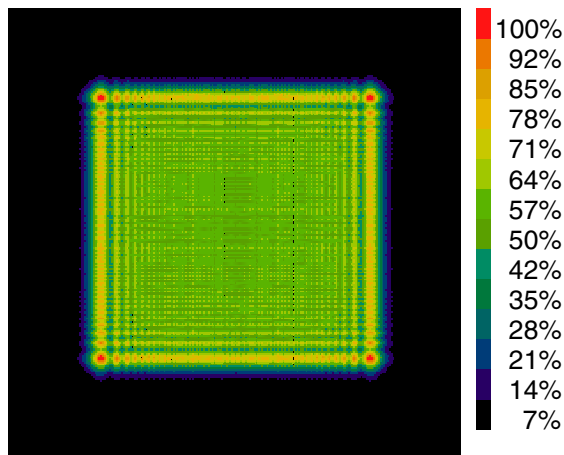


Fig. 88.27. Diffraction pattern for fully coherent uniform illumination of a square aperture after propagation a short distance.

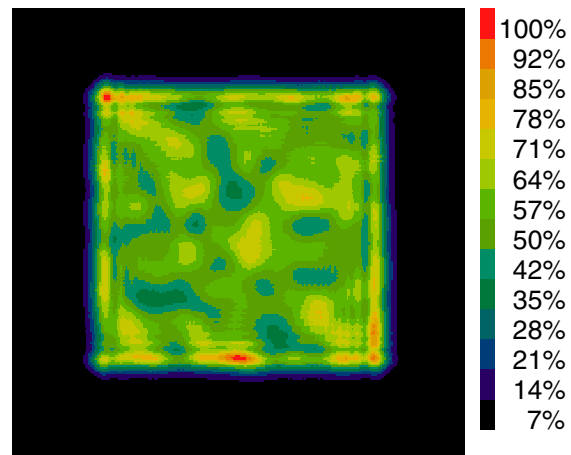


Fig. 88.28. Diffraction distribution of $M^2 = 15$ after 50 instances showing that the characteristic speckle size for $M^2 = 15$. With 50 instances, the nonuniformity is about $1/\sqrt{50} \approx 14\%$.

```

c Calculate effect of high M2 LED beam on suppression of
c diffraction ringing after a square aperture with:
c M^2 = 60, very short spatial coherence suppresses diffraction ringing
c M^2 = 15, mid-range spatial coherence partially suppresses ringing.
c
c The partial coherence of the LED affects the degree of suppression.
c
c A beam of given M2 value is propagated a distance L and
c incoherently summed.
c
c The macro should be run about 10,000 times to get 1% accuracy
c

```

Jump to: [Commands](#), [Theory](#)

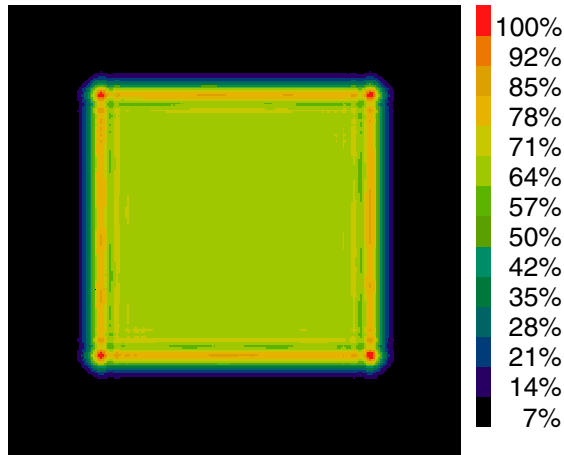


Fig. 88.29. Diffraction distribution of $M^2 = 15$ after 10,000 instances showing that the diffraction ringing has been reduced but not eliminated because the finite size of the speckle allows a finite spatial coherence. Compare with Fig. 88.25.

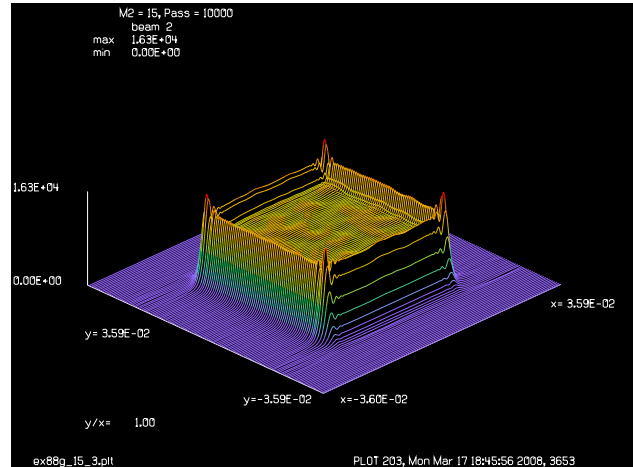


Fig. 88.30. Diffraction distribution of $M^2 = 15$ after 10,000 instances showing that some diffraction ringing remains because the larger speckle leaves a finite coherence size. Note the presence of “rabbit ears” due to partial coherence. Compare with Fig. 88.26.

```
alias name ex88g
variable/dec/int SWITCH Pass Ntimes Nlinex Nliney FIRST M2

SWITCH = 1    # choose 1 for M^2 = 60, else M^2 = 15

Nlinex = 512
array/s 1 Nlinex
nbeam 2 data      # beam for summing irradiance pattern

wavelength/set 1 0.81 # 810nm
Xwidth = .2/10      # guess at a suitable width

units/field 0 1.8*Xwidth # setup sampling and a guardband
clap/sqr 1 Xwidth      # square aperture

L = 1/12            # guess at some distance
if [SWITCH==1] then
    M2 = 60          # approximate M-squared value
else
    M2 = 15
endif

set/density 256 256
set/window/abs -1.5*Xwidth 1.5*Xwidth -1.6*Xwidth 1.5*Xwidth

c
c Calculate fully coherent diffraction pattern
c
```

Jump to: [Commands](#), [Theory](#)


```

clap/sqr 1 Xwidth
prop L
plot/w @name_@M2_1.plt
title fully coherent diffraction pattern
plot/b/i/b 1

macro/def step
  Pass = Pass + 1
  zreff/set 1 0
  array/reset 1
  noise/smooth 1 1 2*Xwidth/M2
  clap/sqr 1 Xwidth
  prop L
  add/inc/con 2 1
  if [mod(Pass,50)==0] then
    plot/w @name_@M2_2.plt
    title M2 = @M2, Pass = @Pass
    plot/b/i/b 2
  endif
  if [Pass==50] then
    plot/w @name_@M2_4.plt
    title M2 = @M2, Pass = @Pass
    plot/b/i/b 2
  endif
endif

macro/end
clear 2 0
time/i
write/off
macro/run step/10000
write/on
time
plot/w @name_@M2_3.plt
plot/l 2 ns=128

```


Ex89: Binary optics

Table. 89.1. Table of Ex89 examples

Ex89a: Binary grating surface calculation.	2
Ex89b: Binary grating, direct phase calculation	6
Ex89c: Binary lens, positive element	8
Ex89d: Binary lens, negative element	11
Ex89e: Binary lens, positive-negative element	13
Ex89f: Binary lens, dispersion.	15
Ex89g: Binary fractioning of an arbitrary surface.	16

Binary optics consist of discrete layers of identical thickness—grouped into major levels of thickness $\lambda/(n-1)$ and sublevels. Figure 89.1 shows a continuous surface after being divided into major levels and sublevels with 8 sublevels per major level. Since the major levels contribute exact multiples of 2π to the phase at the nominal wavelength, these layers may be subtracted from the surface. The lower figure of Fig. 89.1 shows the surface with major levels subtracted. Fig. 89.2 shows the binary fractioning as illustrated in Example 89g. In binary optics, the number sublevels per major level is generally set to be a power of 2, e.g., 2, 4, 8, etc. The maximum height of the binary optic is

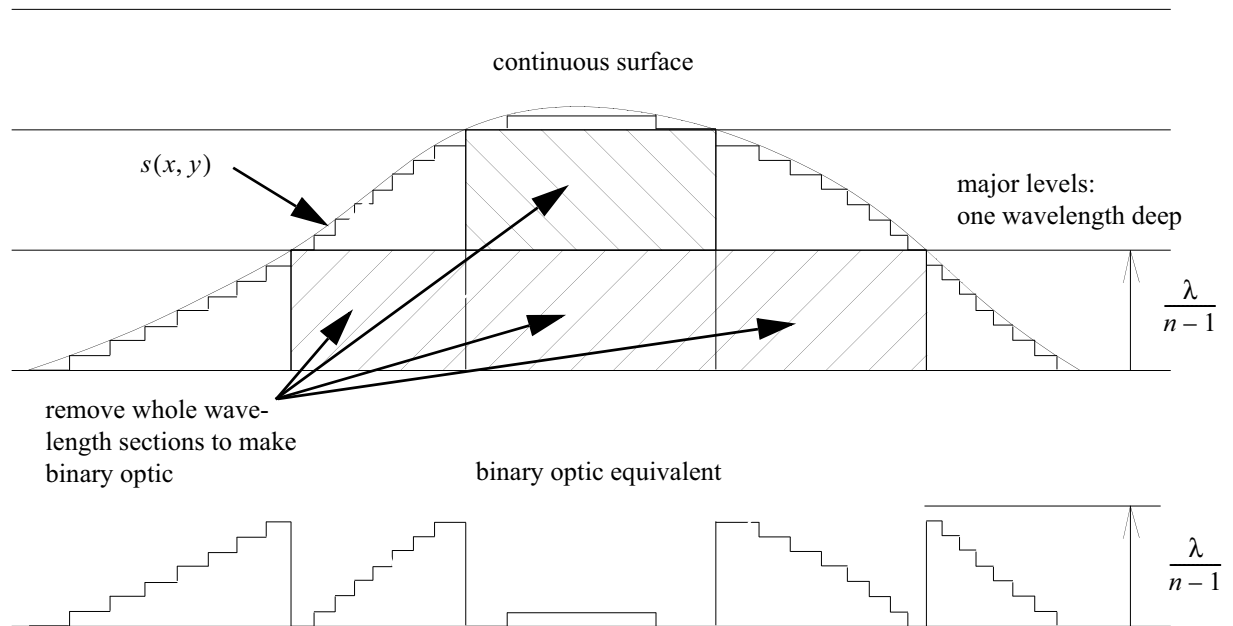


Fig. 89.1. A surface divided into major levels of thickness $\lambda/n-1$ and into 8 sub levels to create a binary optic. The major levels may be subtracted without optical effect at wavelength λ , i.e., by removing the hatched areas. The sublevels produce some diffractive loss into high angle scattering. Note that the maximum height of the binary optic is $\frac{N-1}{N} \frac{\lambda}{n-1}$, where N is the number of sublevels.

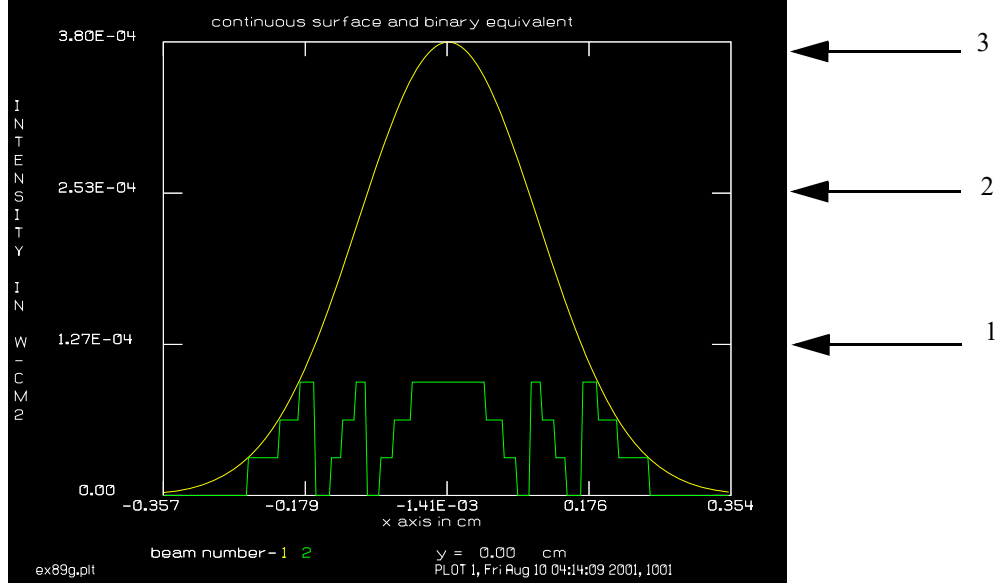


Fig. 89.2. A binary surface created from a gaussian surface by dividing the surface into three major levels and four sublevels. Note that one of the sublevels is zero so that the peak value of the binary surface is three sublevels high. Compare with Fig. 89.1.

$$\frac{N-1}{N} \frac{\lambda}{n-1}, \quad (89.1)$$

where N is the number of sublevels. For $N = 2$, phase steps of 0 and π are created.

In GLAD, the binary lens is defined by the design wavelength, radius, index of refraction, and size of the major levels and the number of sublevels. GLAD will then calculate performance at any wavelength. Binary gratings of the Dammann type may also be specified in terms of grating period, azimuth angle, major level size and number of sublevels.

The mathematical description of the binary optic surface is

$$S'(x, y) = \frac{L}{N} \text{mod} \left\{ \left\lfloor \left(\frac{Ns(x, y)}{L} \right) \right\rfloor - N \left[\frac{1}{N} \left\lfloor \left(\frac{Ns(x, y)}{L} \right) \right\rfloor \right], N \right\} \quad (89.2)$$

$$\text{optical phase induced} = \frac{2\pi}{\lambda} (n-1) S'(x, y) \quad (89.3)$$

where $\lfloor \cdot \rfloor$ is the least integer function (rounds down to next integer value).

Ex89a: Binary grating surface calculation

Example 89a illustrates a binary grating made by constructing the binary surface and then applying it using the `sfocus` command. The surface is represented as an irradiance of the appropriate height in cm. The surface height may be displayed with any of the usual plot commands for irradiance. `sfocus` applies

Jump to: [Commands](#), [Theory](#)

the surface height as a phase error. The surface is multiplied by $\alpha = n - 1$ and z , the propagation distance, is set to 1. `SFOCUS` includes the term $2\pi/\lambda$. Sublevel numbers of 2, 4, 8, and 16 are used. With this choice of units, 16 sublevels are not well resolved, but the surface is well represented as a nearly smooth distribution.

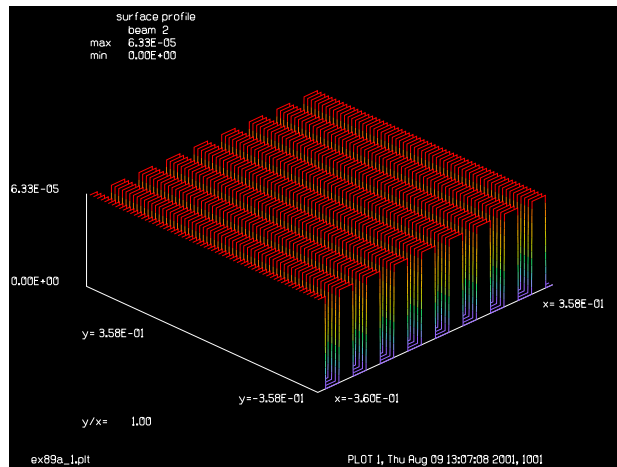


Fig. 89a.1. Binary grating with 2 sublevels: 0 and $\lambda/2(n - 1)$.

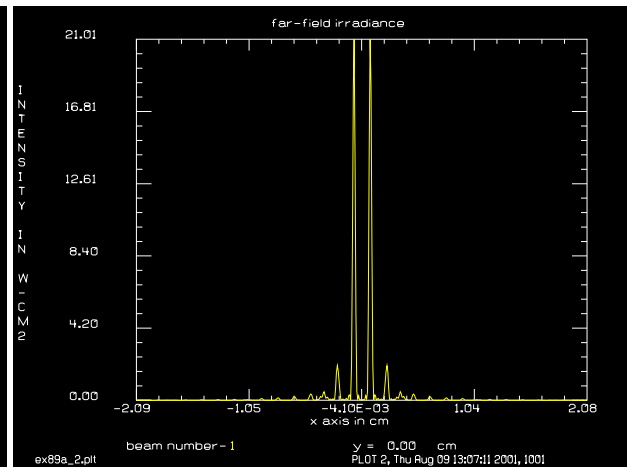


Fig. 89a.2. Far-field of 2-sublevel grating. Two equal sidelobes are created.

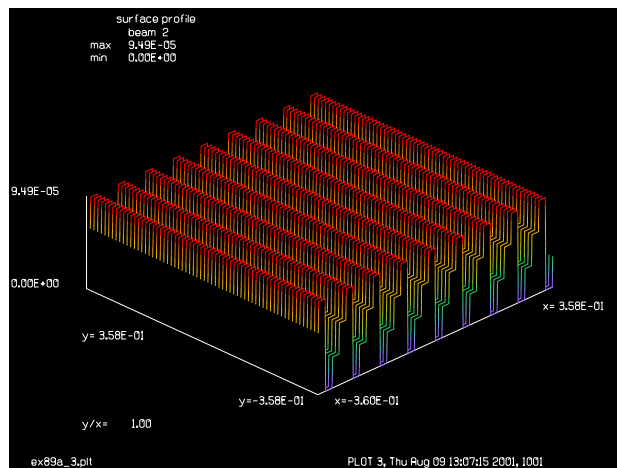


Fig. 89a.3. Binary grating with 4 sublevels.

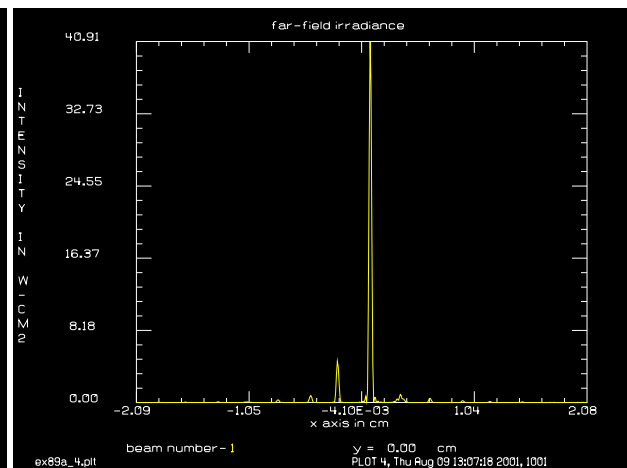


Fig. 89a.4. Far-field of 4-sublevel grating. The right sidelobe is dominant, showing blazing effect.

Input: ex89a.inp

```
c## ex89a
c
c Example 89a: Binary grating surface calculation
c
c This example implements a binary grating, designed for .6328 micron
c a 512 by 512 array is used and levels 2, 4, 8, 16 are used.
c The period of the grating is .0843 cm.
c
```

Jump to: [Commands](#), [Theory](#)

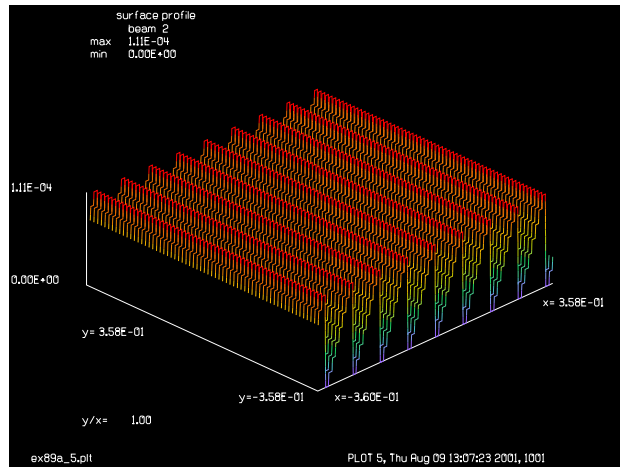


Fig. 89a.5. Binary grating with 8 sublevels.

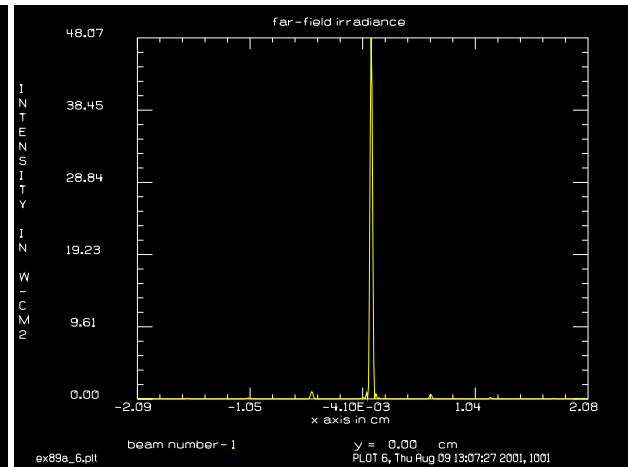


Fig. 89a.6. Far-field of 8-sublevel grating. Efficiency is very good.

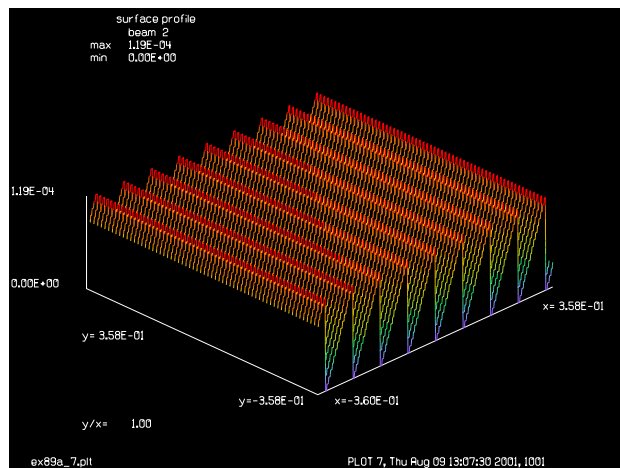


Fig. 89a.7. Binary grating with 16 sublevels.

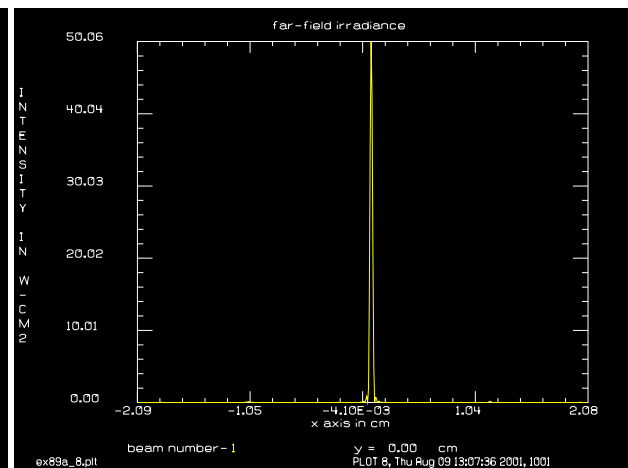


Fig. 89a.8. Far-field of 16-sublevel binary grating. Blazing is essentially perfect.

c The peak irradiance is calculated for each choice of number of levels.
 c For 2 levels, the center irradiance is zero and only two side bands are
 c present. As the number of levels is increased from 4 to 16, the
 c blazing efficiency increases to the point where almost all of the light
 c goes into the right-hand order.

c
 c In Ex89a, the grating surface is calculated first and stored in beam 2.
 c The surface is implemented as a phase screen by means of SFOCUS.

```
c
variab/dec/int count plotno
mem/set/b 4
color = .6328e-4
array/s 1 512
nbeam 2 512 data
wavelength/set 0 color*1e4
units/s 0 .001405
rindex = 1.5
```

Jump to: [Commands](#), [Theory](#)

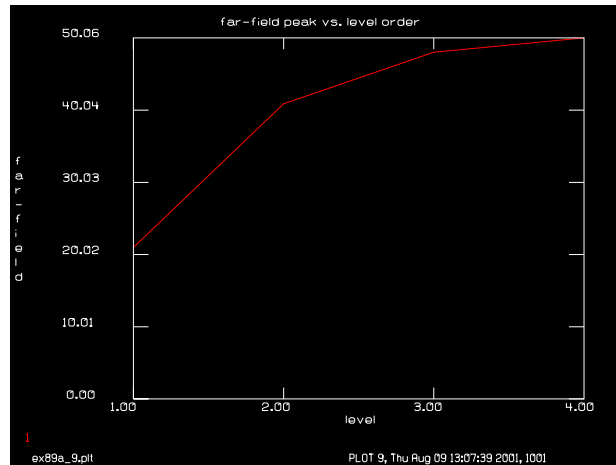


Fig. 89a.9. Far-field peak as a function of the number of sublevels expressed as a power of 2. Other gratings and lenses show similar improvements in efficiency with number of levels.

```

level = color/(rindex-1.) list
macro/def binary/o
  count = count+1
  units/s 1 .001405
  array/reset 1
  clear 1 1
  clap/c/c 1 .12
  binary/grating/surface 2 .0843 level 2^count 90
  plotno = plotno+1
  plot/watch ex89a_@plotno.plt
  title surface profile
  int2phas/two 1 2 [2.*pi*(rindex-1)/color]
  lens 1 100
  dist 100
  plotno = plotno+1
  plot/watch ex89a_@plotno.plt
  title far-field irradiance
  plot/x/i 1
  variab/set peak 1 peak
  udata/set count count peak
macro/end
count = 0
plot/tty/pause 3
mac binary/4
udata/xlabel level order
udata/ylabel far-field peak
title far-field peak vs. level order
plotno = plotno+1
plot/watch ex89a_@plotno.plt
set/grid 3 1 5 1
plot/udata min=0

```

Ex89b: Binary grating, direct phase calculation

Ex89b is the same problem done by applying the distribution directly as a phase screen. This method is faster but does not allow the surface phase to be displayed, since the graphic phase algorithm can not remove the 2π levels from such a discontinuous function. In both, examples the peak intensity is plotted as a function of the power of 2 of the number of sublevels. The plots of the far-field irradiance show the suppression of sidelobes due to higher sublevel numbers.

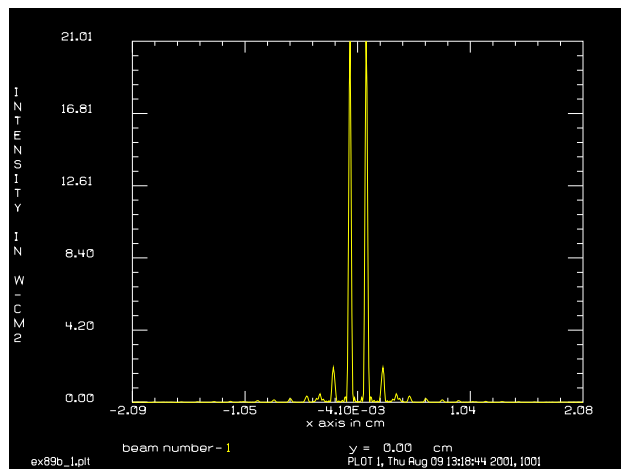


Fig. 89b.1. Far-field with binary phase grating with 2 sublevels. Compare with Fig. 89a.2.

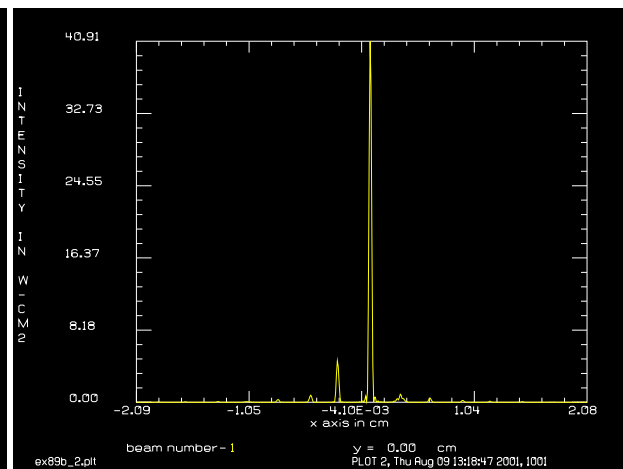


Fig. 89b.2. Far-field with binary phase grating with 4 sublevels. Compare with Fig. 89a.4.

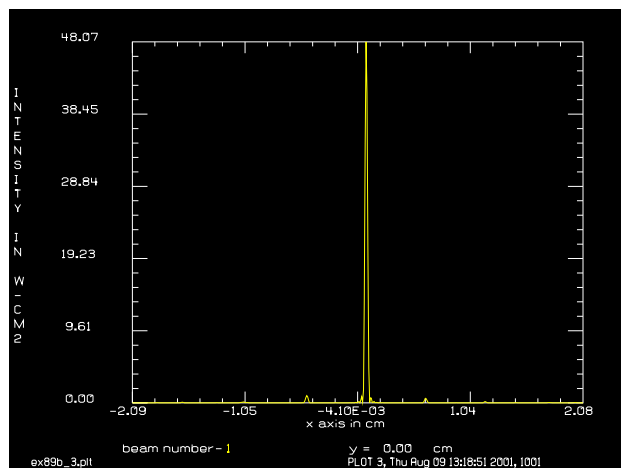


Fig. 89b.3. Far-field with binary grating with 8 sublevels. Compare with Fig. 89a.6.

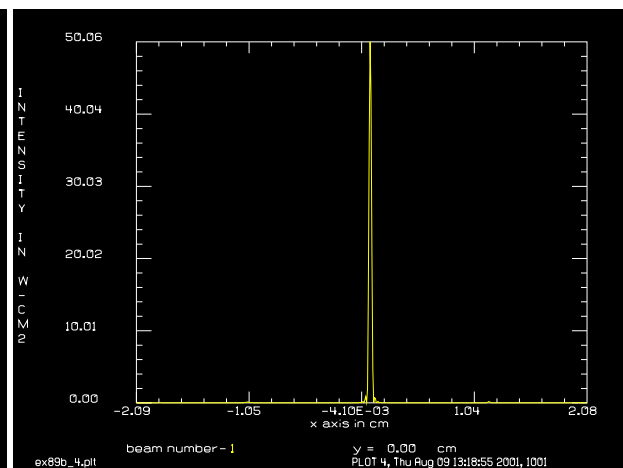


Fig. 89b.4. Far-field with binary phase grating with 16 sublevels. Compare with Fig. 89a.8.

Input: `ex89b.inp`

```
c## ex89b
c
c Example 89b: Binary grating, direct phase calculation
c
c This example implements a binary grating, designed for .6328 micron
c a 512 by 512 array is used and levels 2, 4, 8, 16 are used.
```

Jump to: [Commands](#), [Theory](#)

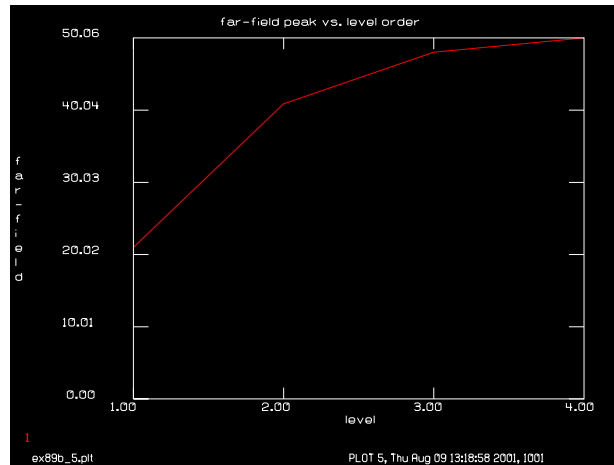


Fig. 89b.5. For binary phase grating. Far-field peak as a function of the number of sublevels expressed as a power of 2. This variation of efficiency of gratings and other lenses. Compare with Fig. 89a.9.

```

c The period of the grating is .0843 cm.
c
c The peak irradiance is calculated for each choice of number of levels.
c For 2 levels, the center irradiance is zero and only two side bands are
c present. As the number of levels is increased from 4 to 16, the
c blazing efficiency increases to the point where almost all of the light
c goes into the right-hand order.
c
variab/dec/int count
mem/set/b 2
color = .6328e-4
array/s 1 512
wavelength/set 0 color*1e4
units/s 0 .001405
rindex = 1.5
level = color/(rindex-1.) list
macro/def binary/o
    count = count+1
    zreff/se 1 0
    array/reset 1
    units/s 1 .001405
    clap/c/c 1 .12
    binary/grating/phasescreen 1 rindex .0843 level 2^count 90
    lens 1 100
    dist 100
    plot/watch ex89b_@count.plt
    plot/x/i 1
    variab/set peak 1 peak
    udata/set count count peak
macro/end
count = 0
plot/screen/pause 3
mac binary/4
udata/xlabel level order
udata/ylabel far-field peak

```

Jump to: [Commands](#), [Theory](#)

```

title far-field peak vs. level order
count = count+1
plot/watch ex89b_@count.plt
set/grid 3 1 5 1
plot/udata min=0

```

Ex89c: Binary lens, positive element

Example 89c shows a positive lens of index 1.5 and radius 100, giving a focal length of 200 cm. The clear aperture of the lens is 0.225 cm and the design wavelength is 0.6328 microns. The peak intensity in the far-field is displayed against sublevel order number showing similar asymptotic behavior as with the grating.

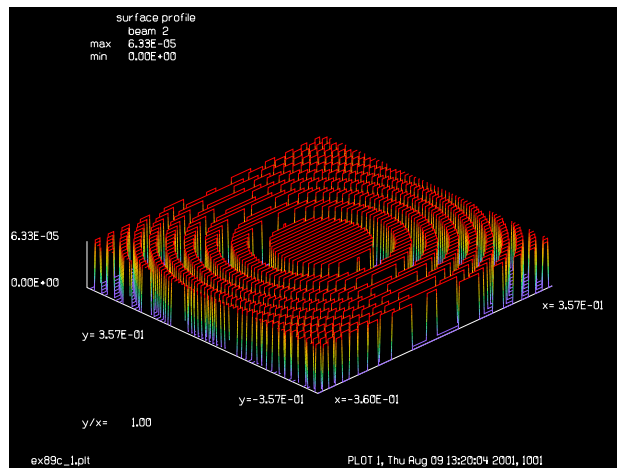


Fig. 89c.1. Positive lens with 2 levels.

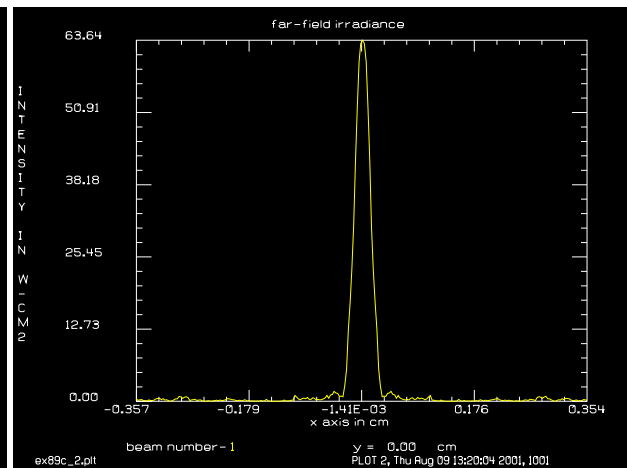


Fig. 89c.2. Far-field of 2-level binary lens. Equal power is diffracted into the divergent mode.

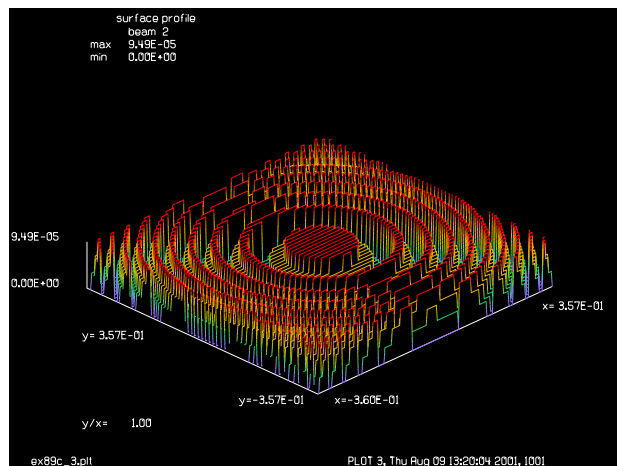


Fig. 89c.3. Positive lens with 4 levels.

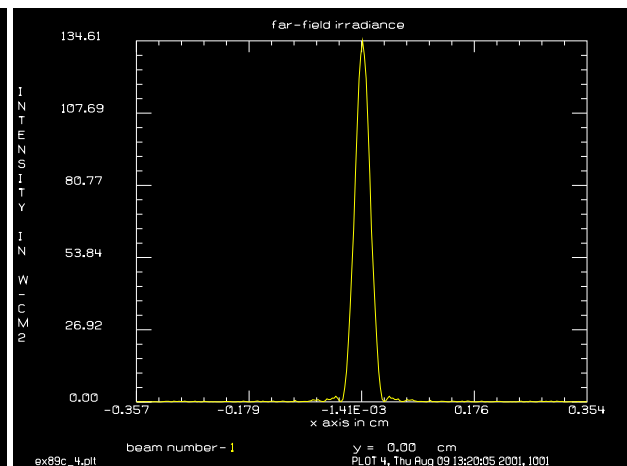


Fig. 89c.4. Far-field of 4-level binary lens.

Input: `ex89c.inp`

c## `ex89c!26850989402773`

Jump to: [Commands](#), [Theory](#)

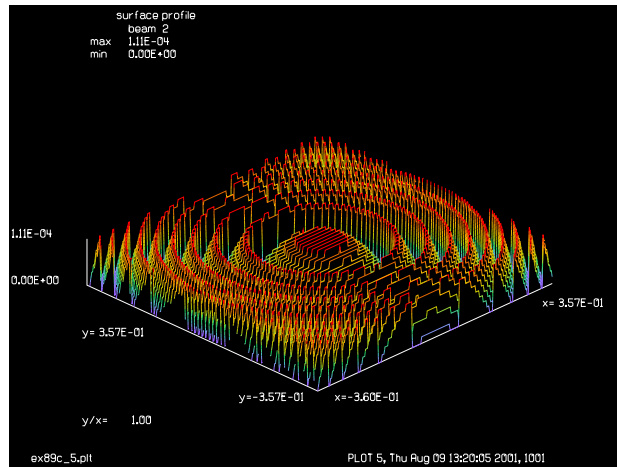


Fig. 89c.5. Positive lens with 8 levels.

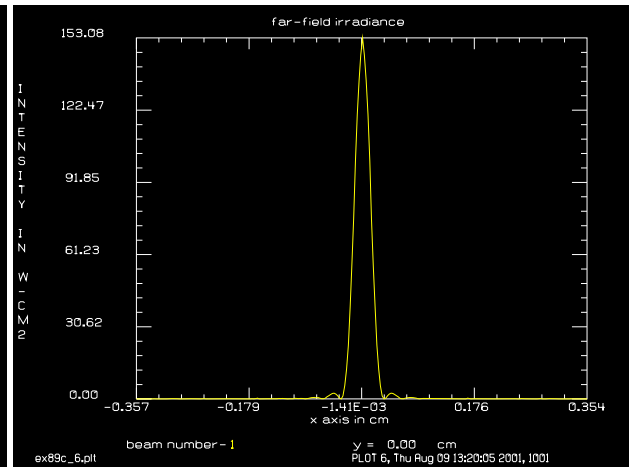


Fig. 89c.6. Far-field of 8-level binary lens.

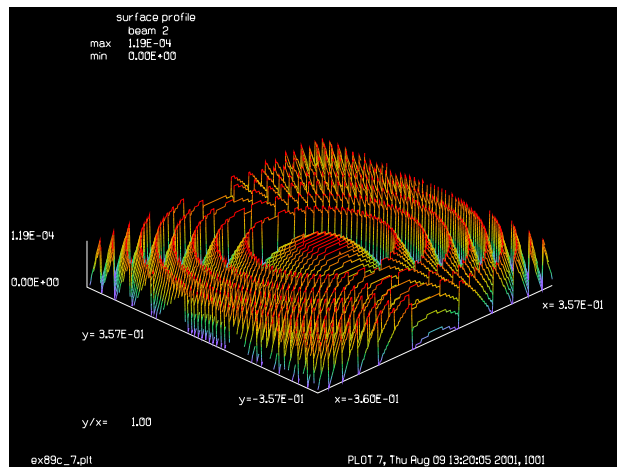


Fig. 89c.7. Positive lens with 16 levels.

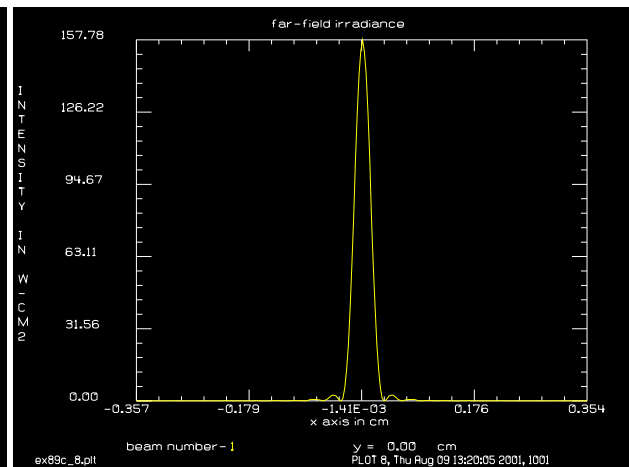


Fig. 89c.8. Far-field of 16-level binary lens. Equal power is diffracted into the divergent mode.

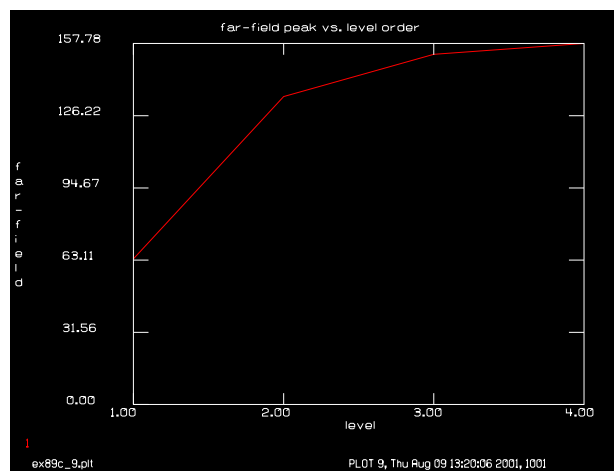


Fig. 89c.9. Far-field peak as a function of the number of sublevels expressed as a power of 2.

C

Jump to: [Commands](#), [Theory](#)

```
c Example 89c: Binary lens, positive element
c
poptext/compose ex89c.txt
Examples of some positive binary lenses with
2, 4, 8 and 16 binary levels.
```

Note decrease in sidelobes with more binary levels showing improved diffraction efficiency.

```
poptext/end
poptext ex89c.txt 8
variab/dec/int count plotno
color = .6328e-4
radius = 100
mem/set/b 1
array/s 1 256
nbeam 2 256 data
wavelength/set 0 color*1e4
units/s 0 .00281
rindex = 1.5 # set index of refraction to 1.5
level = color/(rindex-1.) list
macro/def binary/o
  count = count+1
  array/reset 1
  units/s 1 .00281
  clap/c/c 1 .225
  if 1 = 1 then
c make a binary surface
  binary/lens/surface 2 radius radius level 2^count
  plotno = plotno+1
  plot/watch ex89c_@plotno.plt
  title surface profile
  plot/l 2 ns=64 h=.2
  int2phas/two 1 2 [2.*pi*(rindex-1)/color]
  endif
  if 1 = 0 then
c implement binary optic as a phase screen
  binary/lens/phase 1 rindex radius radius level 2^count
  endif
  if 1 = 0 then
c implement binary optic as a residual phase
  binary/lens/residual 1 rindex radius radius level 2^count
  endif
  dist [radius/(rindex-1)] # propagate to focal length of lens
  plotno = [plotno+1]
  plot/watch ex89c_@plotno.plt
  title far-field irradiance
  plot/x/i 1
  variab/set peak 1 peak
  udata/set count count peak
macro/end
count = 0
mac binary/4
udata/xlabel level order
udata/ylabel far-field peak
```

Jump to: [Commands](#), [Theory](#)

```

title far-field peak vs. level order
plotno = plotno+1
plot/watch ex89c_@plotno.plt
set/grid 3 1 5 1
plot/udata min=0
pause 5
end

```

Ex89d: Binary lens, negative element

Example 89d illustrates a negative lens similar to the positive lens of Ex89c and the far-field is found 100 cm behind the lens at the virtual focus.

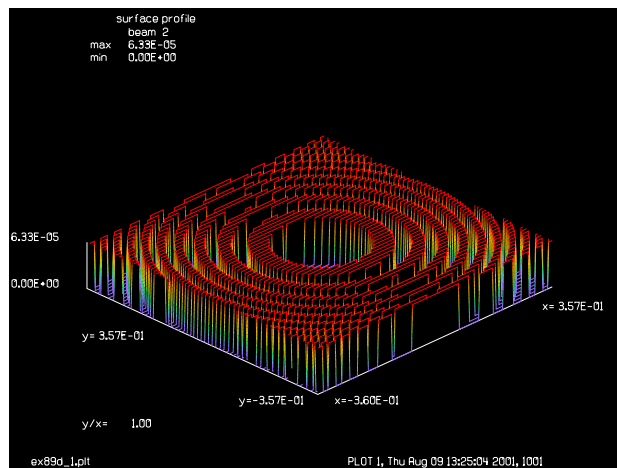


Fig. 89d.1. Negative lens with 2 levels.

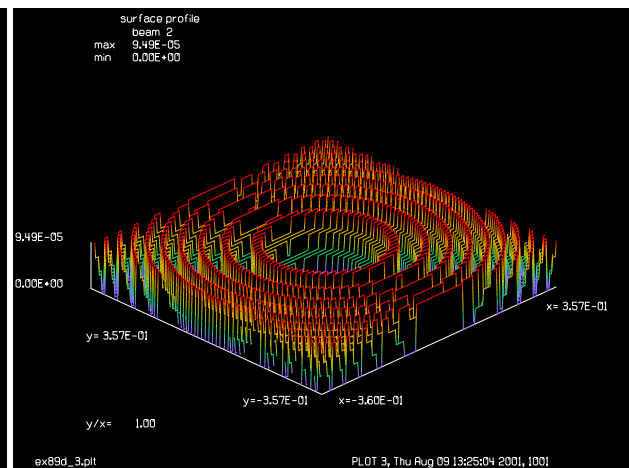


Fig. 89d.2. Negative lens with 4 levels.

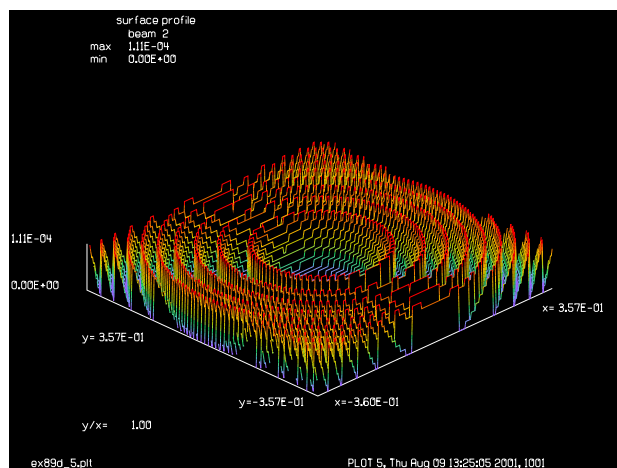


Fig. 89d.3. Negative lens with 8 levels.

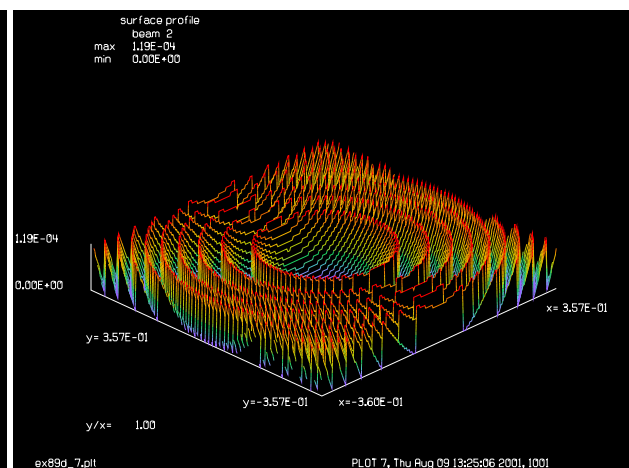


Fig. 89d.4. Negative lens with 16 levels.

Input: `ex89d.inp`

```

c## ex89d
c

```

Jump to: [Commands](#), [Theory](#)

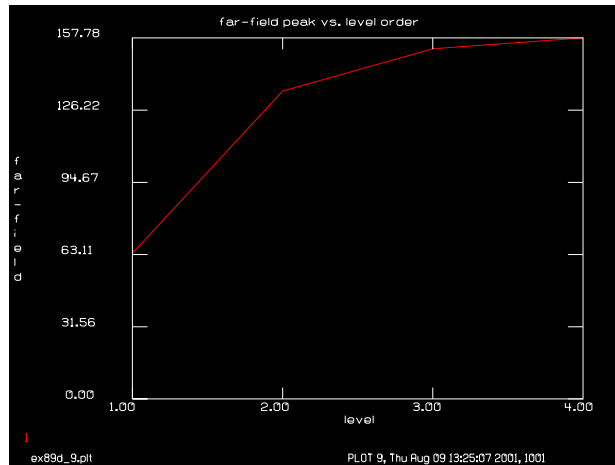


Fig. 89d.5. Far-field peak as a function of the number of sublevels expressed as a power of 2 for Ex89d.

c Example 89d: Binary lens, negative element

c

poptext/compose ex89c.txt

Examples of some negative binary lenses with
2, 4, 8 and 16 binary levels.

Note decrease in sidelobes with more binary levels
showing improved diffraction efficiency.

poptext/end

poptext ex89c.txt 8

variab/dec/int count plotno

color = .6328e-4

radius = -100

mem/set/b 1

array/s 1 256

nbeam 2 256 data

wavelength/set 0 color*1e4

units/s 0 .00281

rindex = 1.5

set index of refraction to 1.5

level = color/(rindex-1.) list

macro/def binary/o

count = count+1

array/reset 1

units/s 1 .00281

clap/c/c 1 .225

if 1 = 1 then

binary/lens/surface 2 radius radius level 2^count

plotno = plotno+1

plot/watch ex89d_@plotno.plt

title surface profile

plot/l 2 ns=64 h=.2

int2phas/two 1 2 [2.*pi*(rindex-1)/color]

endif

if 1 = 0 then

binary/lens/phase 1 rindex radius radius level 2^count

endif

dist [radius/(rindex-1)]

Jump to: [Commands](#), [Theory](#)

```

plotno = plotno+1
plot/watch ex89d_@plotno.plt
title far-field irradiance
plot/x/i 1
variab/set peak 1 peak
udata/set count count peak
macro/end
count = 0
mac binary/4
udata/xlabel level order
udata/ylabel far-field peak
title far-field peak vs. level order
plotno = plotno+1
plot/watch ex89d_@plotno.plt
set/grid 3 1 5 1
plot/udata min=0

```

Ex89e: Binary lens, positive-negative element

Example 89e is a positive-negative element with equal cylindrical focal lengths of plus and minus 200 cm. Separable propagation and a transpose step are used to obtain the far-field distribution.

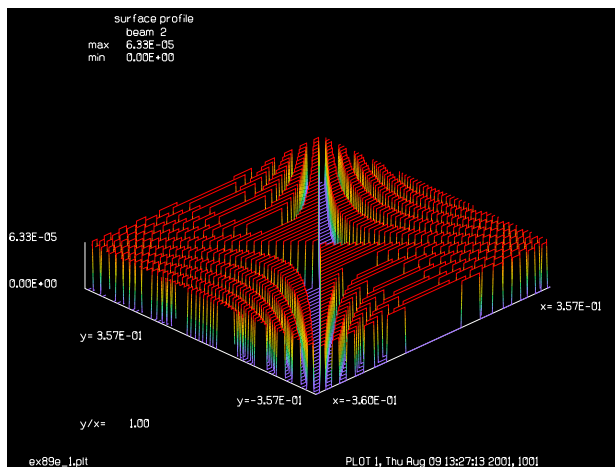


Fig. 89e.1. Positive-negative lens with 2 levels.

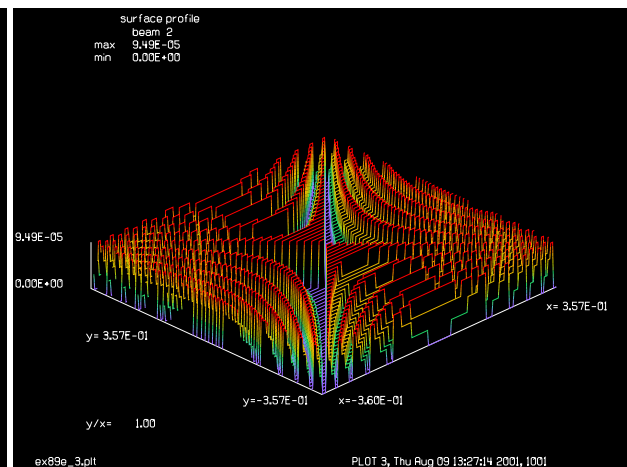


Fig. 89e.2. Positive-negative lens with 4 levels.

Input: ex89e.inp

```

c## ex89e!580254113244719
c
c Example 89e: Binary lens, positive-negative element
c
variab/dec/int count plotno
color = .6328e-4
mem/set/b 1
array/s 1 256
nbeam 2 256 data
wavelength/set 0 color*1e4

```

Jump to: [Commands](#), [Theory](#)

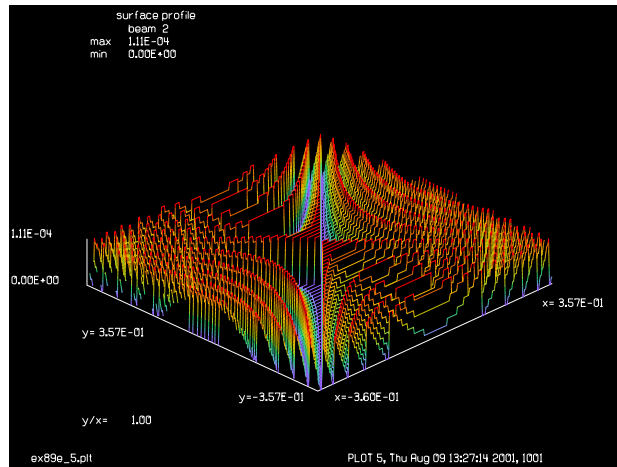


Fig. 89e.3. Positive-negative lens with 8 levels.

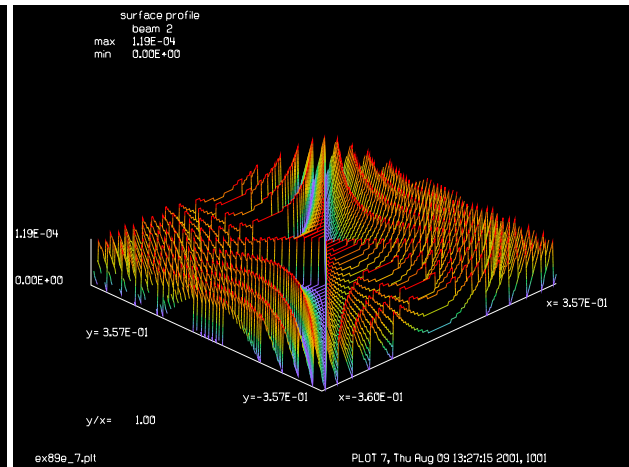


Fig. 89e.4. Positive-negative lens with 16 levels.

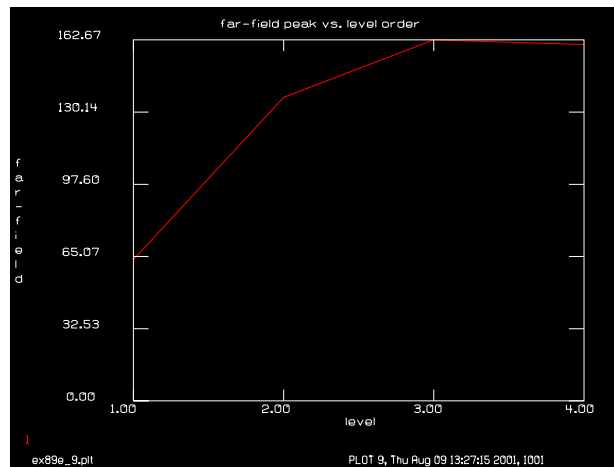


Fig. 89e.5. Far-field peak as a function of the number of sublevels expressed as a power of 2 for Ex89e.

```

units/s 0 .00281
rindex = 1.5
level = color/(rindex-1.) list
macro/def binary/o
  count = count+1
  array/reset 1
  units/s 1 .00281
  clap/c/c 1 .225
  binary/lens/surface 2 -100 100 level 2^count
  plotno = plotno+1
  plot/watch ex89e_@plotno.plt
  title surface profile
  plot/l 2 ns=64 h=.2
  int2phas/two 1 2 [2.*pi*(rindex-1)/color]
  dist -200 xonly
  transpose 1
  dist 200 xonly
  plotno = plotno+1
  plot/watch ex89e_@plotno.plt
  title far-field irradiance

```

set index of refraction to 1.5

propagate two directions

separately

Jump to: [Commands](#), [Theory](#)


```

plot/x/i 1
variab/set peak 1 peak
udata/set count count peak
macro/end
count = 0
mac binary/4
udata/xlabel level order
udata/ylabel far-field peak
title far-field peak vs. level order
plotno = plotno+1
plot/watch ex89e_@plotno.plt
set/grid 3 1 5 1
plot/udata min=0
end

```

Ex89f: Binary lens, dispersion

Example 89f calculates the change in focal length, considering the dispersion due to the binary level characteristics—but ignoring dispersion of the index of refraction.

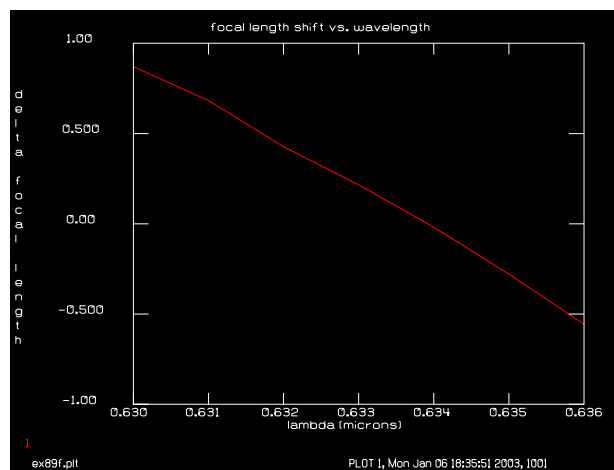


Fig. 89f.1. Focal length shift as a function of wavelength (microns).

Input: ex89f.inp

```

c## ex89f!796102561824305
c
c Example 89f: Binary lens, dispersion
c
variab/dec/int count
color = .6328e-4
rindex = 1.5 # set index of refraction to 1.5
focallength = 100/(rindex-1)
level = color/(rindex-1.) list
macro/def binary/o
  count = count+1
  array/reset 1
  units 1 .00281
  clear 1 1

```

Jump to: [Commands](#), [Theory](#)

```

clap/c/c 1 .225
color = .6328e-4+(count-4)*1e-7
wavelength/set 1 color*1e4
binary/lens/phase 1 1.5 100 100 level 16
fitphase/focus/list 1 list
variab/set focus fitxfocus list
radius = 1./focus/2./color list
udata/set count [count-4] [radius-focallength]
macro/end
array/set 1 256
mac binary/7
udata/list
udata/xlabel delta lambda (nanometers)
udata/ylabel delta focal length
title focal length shift vs. wavelength
plot/watch ex89f.plt
set/grid 6 0 4 1
plot/udata min=-1 max=1
end

```

Ex89g: Binary fractioning of an arbitrary surface

This example illustrated fractioning of an arbitrary surface, in this case a gaussian function, into a binary surface. See Fig. 89.2.

Input: ex89g.inp

```

c## ex89g
c
c Example 89g: Binary fractioning of an arbitrary surface
c
c Any distribution may be tuned into a binary surface with the
c command BINARY/SURFACE
c
c The binary surface may then be implemented with the SFOCUS command.
color = .6328e-4
rindex = 1.5
array/s 1 256
nbeam 2
wavelength/set 0 color*1e4
units/s 0 .00281
level = color/(rindex-1.) list
c make a gaussian shaped surface
c
c choose the peak just under 3 levels (avoid the center point
c exactly at 3 levels -- creating a singular condition for that point
c
gauss/c/c 1 2.999*level .225 1          # make a gaussian surface
copy 1 2
binary/surface 2 level 4                # turn into binary surface
set/grid 4 1 3 1
plot/watch ex89g.plt

```

Jump to: [Commands](#), [Theory](#)

```
title continuous surface and binary equivalent  
plot/x/i fi=1 la=2 fmax=3*level
```


Ex90: High numerical aperture lens with vector diffraction calculation

Table. 90.1. Table of Ex90 examples

Ex90a: High numerical aperture objective lens.	1
Ex90b: High numerical aperture spatial filter, includes recollimation step.	4

This example illustrates the command HIGHNA. Fresnel diffraction theory is not adequate to treat high numerical aperture objective lenses. A vector diffraction treatment will show an elliptical shape to the far-field image for a linearly polarized incident field. Vector diffraction calculations may be implemented with varying degrees of sophistication—enabling the accurate treatment of larger numerical apertures for the more sophisticated models. The method employed by the command HIGHNA is based on dipole projection. The incident field is assumed to have minimum angular spread and to be well represented by the two transverse components of the electric field. (Also see Ex8d.inp for vector effects with a high NA telescope).

For a discussion of the theoretical basis, see Section 6.4 of the GLAD Theoretical Description.

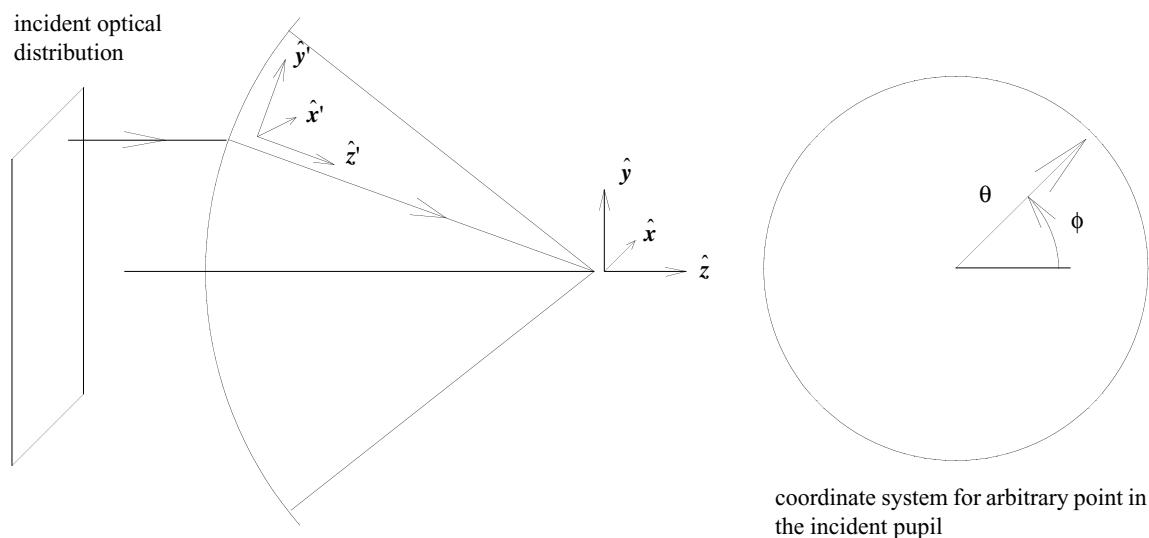


Fig. 90.1. Sketch of dipole projection theory of vector diffraction imaging. A distribution is incident from the left to a high numerical aperture lens. Consider dipoles in the x -, y -, and z -directions for the incident radiation. After refraction, the coordinate system is rotated, taking the form, x' , y' , and z' . The primed system projects onto the original system causing mixing of polarization states, including projection of the transverse components onto the z -axis. Each point in the pupil is designated by the radial and azimuthal angles, θ and ϕ respectively.

Ex90a: High numerical aperture objective lens

Input: `ex90a.inp`

c## ex90a

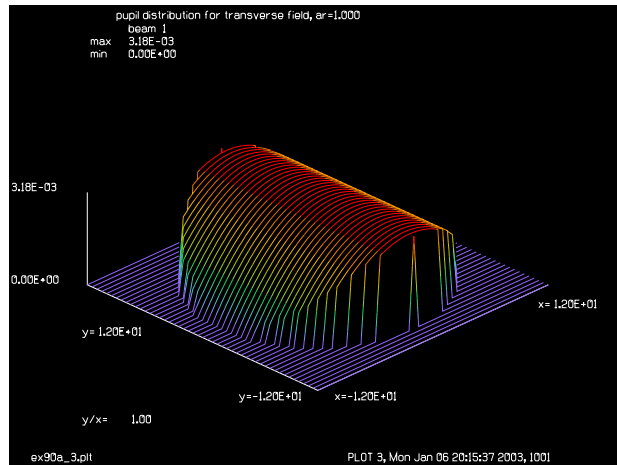


Fig. 90a.1. Pupil distribution for irradiance projected into transverse mode, aperture ratio (RA, radius/focal length) = 1.

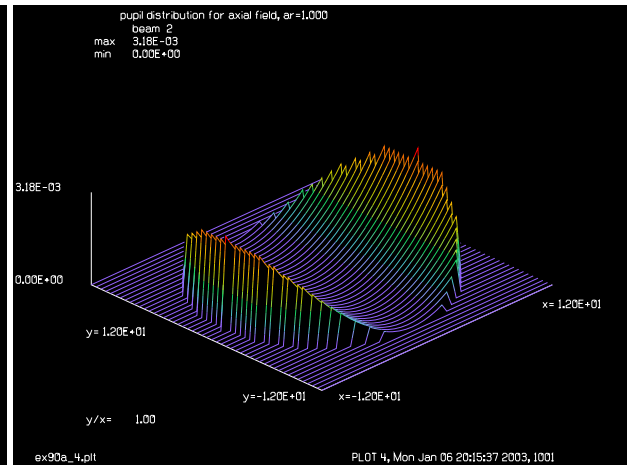


Fig. 90a.2. Pupil distribution for irradiance projected into axial mode, aperture ratio = 1.

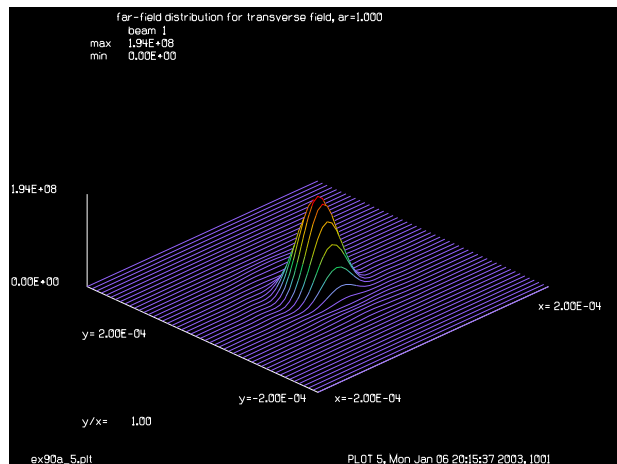


Fig. 90a.3. Far-field distribution for transverse mode, aperture ratio = 1.

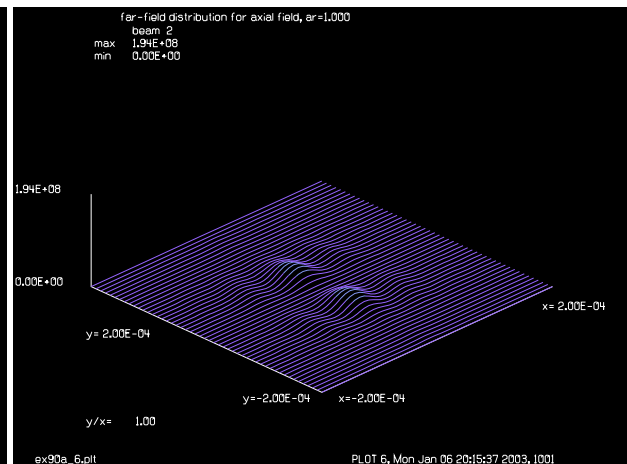


Fig. 90a.4. Far-field distribution for axial mode, aperture ratio = 1, plotted to same scale as Fig. 90a.4.

```

c
c Example 90a: High numerical aperture objective lens
c
c This example illustrates the use of command HIGHNA to apply
c vector diffraction to a fast objective lens. The algorithm
c uses the dipole projection method to separate the incident
c optical field into the three orthogonal components corresponding
c to the x-, y-, and z-components of the focused distribution.
c
alias Name ex90a
variab/dec/int pass switch Nline
Nline = 256
array/s 1 Nline Nline 1          # array for incident field, polarized
                                # this array will contain x- and y-components
                                # of focused field

```

Jump to: [Commands](#), [Theory](#)

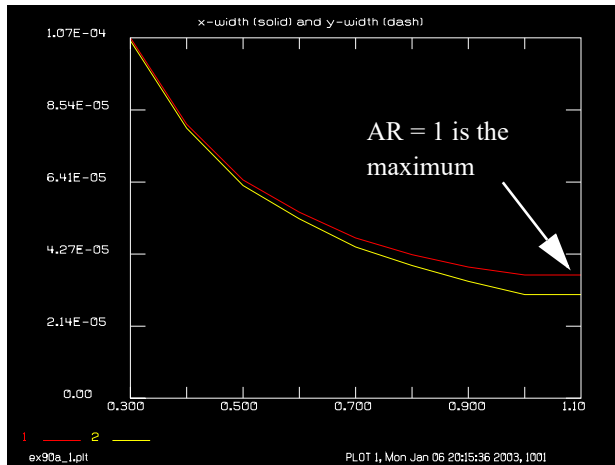


Fig. 90a.5. Average width of x-direction (solid) and y-direction (dash) of far-field transverse mode for the incident field polarized in the x-direction. aperture ratio = 1 is the maximum, the spot size does not decrease for aperture ratio larger than 1.

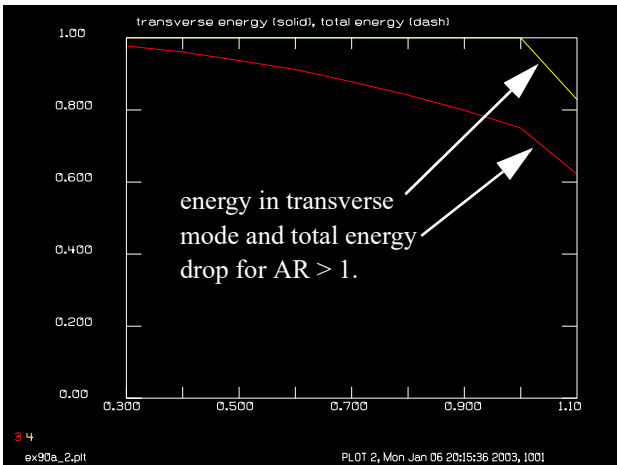


Fig. 90a.6. Energy in the transverse mode (solid) and total energy (dash). Up to $NA = 1$ the energy lost from the transverse mode goes into the axial mode. The aperture is effectively limited to $NA = 1$, so total energy is lost for aperture ratio > 1.0 .

```
nbeam 2 Nline Nline 0          # array for z-component of focused field
focal_length = 10.
wavelength = 1.06
wavelength/set 0 wavelength
macro/def step/o
    pass = pass + 1
    zreff/se 1 0
    zreff/se 2 0
    clear 1 1
    units/s 1 .5
    aperture_ratio = aperture_ratio + .1
    radius = aperture_ratio*focal_length
    clap/c/c 1 radius
    energy/norm 1
    highna 1 2 10
    if switch = 1 then
        variab/set energy1 1 energy
        variab/set energy2 2 energy
        energy_tot = energy1 + energy2
        fitgeo 1 thres=.05
        clap/c/c 1 1e-3
        variab/set xwidth 1 fitxrad
        variab/set ywidth 1 fityrad
        xwidth = 2*xwidth
        ywidth = 2*ywidth
        udata/set pass aperture_ratio xwidth ywidth energy1 energy_tot
    endif
macro/end
aperture_ratio = .2
switch = 1
mac step/9
set/grid 8 0 5 0
```

Jump to: [Commands](#), [Theory](#)

```

plot/watch @Name_1.plt
title x-width (solid) and y-width (dash)
plot/udata 1 2 min=0 dash
plot/watch @Name_2.plt
title transverse energy (solid), total energy (dash)
plot/udata/set y03 y04
plot/udata/seq max=1. min=0.
c
c  reset aperture ratio to look at pupils
c
switch = 0
aperture_ratio = .9 # was .9
mac step
dist -10
plot/watch @Name_3.plt
title pupil distribution for transverse field, ar=@aperture_ratio
plot/l 1 1 xrad=12 ns=64
plot/watch @Name_4.plt
title pupil distribution for axial field, ar=@aperture_ratio
plot/l 2 2 xrad=12 ns=64
dist 10
plot/watch @Name_5.plt
title far-field distribution for transverse field, ar=@aperture_ratio
variab/set peak 1 peak
plot/l 1 1 xrad=2e-4 ns=128 max=peak
plot/watch @Name_6.plt
title far-field distribution for axial field, ar=@aperture_ratio
plot/l 2 2 xrad=2e-4 ns=128 max=peak
set/win/abs -1e-4 1e-4 -1e-4 1e-4
plot/w @Name_7.plt
plot/c 1 ilab=0

```

Ex90b: High numerical aperture spatial filter, includes recollimation step

This example illustrates the use of command `highna` to apply vector diffraction to spatial filter consisting of a high NA objective lens and a high NA recollimating lens. The algorithm uses the dipole projection method to separate the incident optical field into the three orthogonal components corresponding to the x-, y-, and z-components of the focused distribution.

Input: ex90b.inp

```

c## ex90b
c
c  Example 90b: High numerical aperture spatial filter
c
c  This example illustrates the use of command HIGHNA to apply
c  vector diffraction to a spatial filter consisting of both
c  a high NA objective lens and a high NA refocusing lens.
c
c  The algorithm uses the dipole projection method to separate
c  the incident optical field into the three orthogonal components
c  corresponding to the x-, y-, and z-components of the focused

```

Jump to: [Commands](#), [Theory](#)

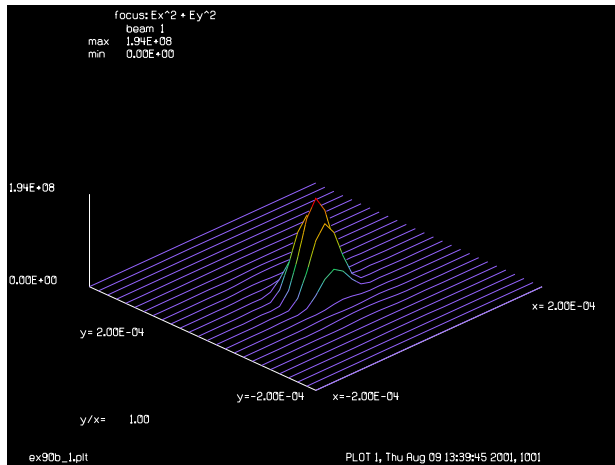


Fig. 90b.1. Intensity of orthogonal files at focus.

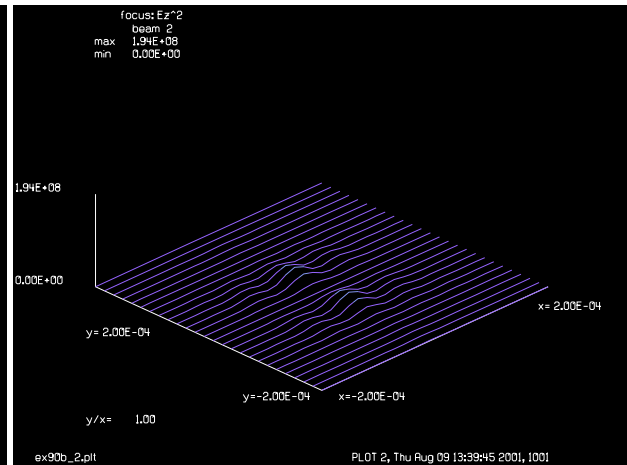


Fig. 90b.2. Intensity of axial filed at focus.

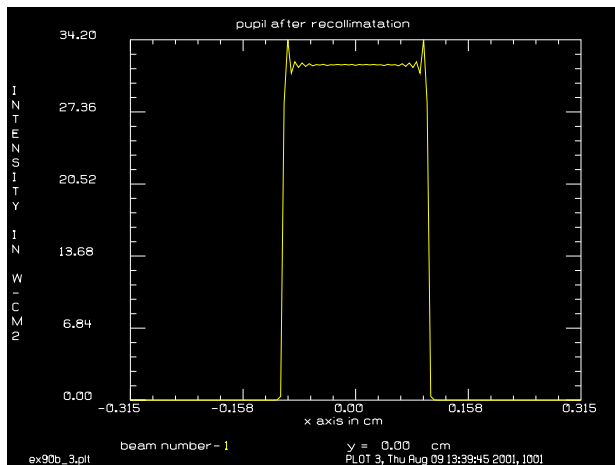


Fig. 90b.3. Beam after recollimation, showing expected diffraction ringing.

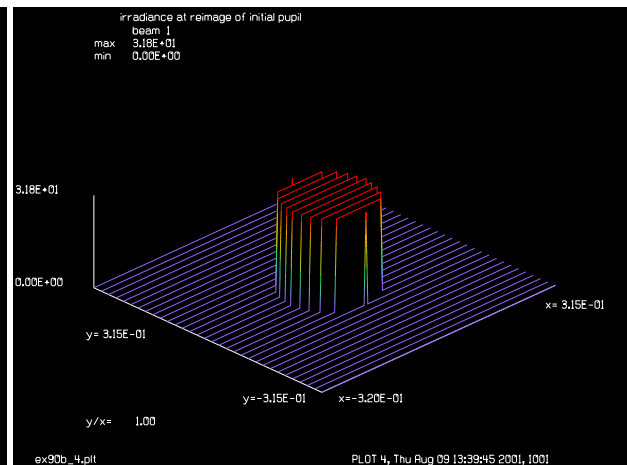


Fig. 90b.4. Beam after recollimation at the image of the original pupil, two focal lengths beyond last lens.

c distribution.

c

alias Name ex90b

variab/dec/int pass switch

```
array/s 1 128 128 1      # array for incident field, polarized
                          # this array will contain x- and y-components
                          # of focused field
nbeam 2 128 128 0        # array for z-component of focused field
```

focal_length = .1

wavelength = 1.06

wavelength/set 0 wavelength

zreff/se 1 0

zreff/se 2 0

clear 1 .1

units/s 1 .005

aperture_ratio = 1

radius = aperture_ratio*focal_length

clap/c/c 1 radius

Jump to: [Commands](#), [Theory](#)

```
energy/norm 1
highna/focus 1 2 focal_length      # make focus
variab/set Peak 1 peak
plot/w @Name_1.plt
title  focus:  $E_x^2 + E_y^2$ 
plot/l 1 1 xrad=.0002 max=Peak ns=128
plot/w @Name_2.plt
title focus:  $E_z^2$ 
plot/l 2 2 xrad=.0002 max=Peak ns=128
energy
c
c  Put spatial filter here
c
highna/collimate 1 2 focal_length    # recollimate
plot/w @Name_3.plt
title pupil after recollimation
plot/x/i 1
c
c  propagate to image of original image
c
prop 2*focal_length
plot/w @Name_4.plt
title irradiance at reimage of initial pupil
plot/l 1 1
```

Ex91: Measuring beam width and M-squared

Table. 91.1. Table of Ex91 examples

Ex91a: Measuring spot width by mode fitting.	1
Ex91b: Measuring spot width using <code>fitgeo</code> with noise	2
Ex91c: Power-in-the-bucket	4
Ex91d: Fitting Hermite-Gaussian	5
Ex91e: Finding and displaying the region containing specified energy	7
Ex91f: Fitting embedded gaussian to data set	10
Ex91g: Fitting embedded gaussian, noise and aberration	12
Ex91h: Fitting embedded gaussian, noise and aberration	15
Ex91i: Calculation of best-fit gaussian as a function of lens spacing	18

In this example, several means of measuring beam size are demonstrated. In particular, M^2 has been argued to be the best method of characterizing all laser beams. Actually, M^2 is defined and measured in different ways and these various methods, as will be shown, do not produce consistent results.

Ex91a: Measuring spot width by mode fitting

Example 91a illustrates mode fitting of a gaussian beam of variable width to an ideal gaussian beam and a gaussian beam with 5% noise added. With no noise the waist is fit to the value 9.99 in excellent agreement with the actual waist radius of 10.00. With 5% noise the fit is still fairly good at a radius of 9.10.

Input: `ex91a.inp`

```

c## ex91a
c
c Example 91a: Measuring spot width by mode fitting
c
c One of the principle problems in measuring the width of a beam
c is errors due to small amounts of noise. Mode matching is relatively
c insensitive to noise but assumes that the general form of the beam
c distribution is known.
c
c In this example, the width of a gaussian beam is fit to the test
c distribution. The correlation of the two beams is used a merit
c function. Without noise the fit is nearly perfect. Even with
c 5% noise the fit is still fairly good.
c
nbeam 2
gaus/c/c 1 1 10
macro/def step/o
    gaus/c/c 2 1 width
    mult/mode/corr 1 2
    variab/set abscorr abscorr
    tar = 1 - abscorr
macro/end
opt/var/add width .1
opt/tar/add tar
opt/name step

```

```

width = 8
write/off
opt/run 8
write/on
width=
pause
gaus/c/c 1 1 10
noise 1 .05
width = 8
write/off
opt/run 8
write/on
width=
end

```

Ex91b: Measuring spot width using `fitgeo` with noise

Example 91b illustrates measurement of beam width in the presence of noise. `fitgeo` calculates the variance and standard deviation of the irradiance. The standard deviation is extremely sensitive to the presence of noise in the distribution. Noise of 5% is added to the distribution and then thresholding is used to reduce the noise. Fig. 91.1 shows a decentered gaussian beam with no noise, pass = 1. Figs. 91.2 to 91.6 (passes 2 to 6) show the appearance of the distribution with threshold values of 5% to 20% in 5% steps. Fig 91.7 shows the variation in the calculated value of gaussian 1/e amplitude radius for the different threshold values. Since standard deviation varies so much with the choice of threshold, thresholding is not a reliable method of removing the effects of noise. A more reliable method of removing the effects of noise is to fit a set of Hermite gaussian functions as illustrated in Ex 91d.

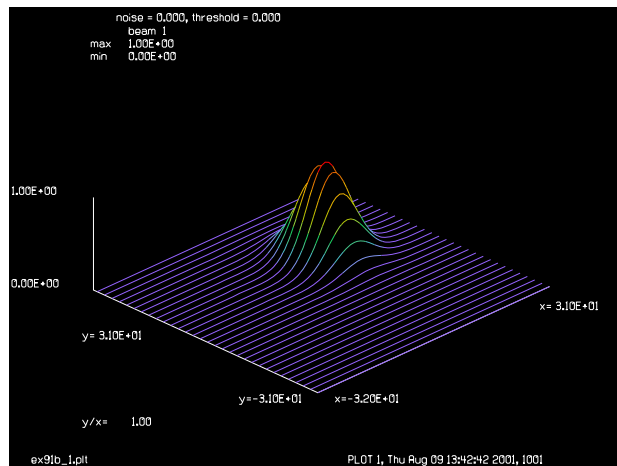


Fig. 91.1. Decentered gaussian with no noise.

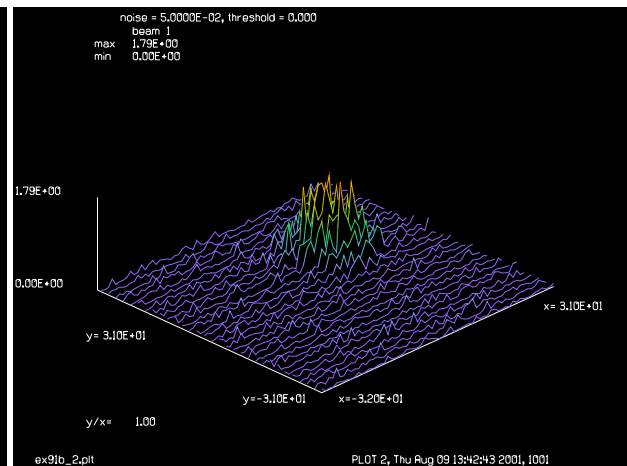


Fig. 91.2. Decentered gaussian with 5% noise added coherently, threshold is zero.

Input: `ex91b.inp`

```

c## ex91b
c

```

Jump to: [Commands](#), [Theory](#)

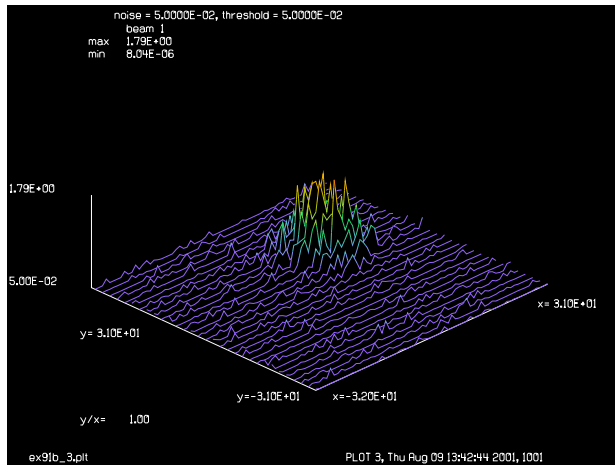


Fig. 91.3. Decentered gaussian with 5% noise added coherently, threshold is 5%.

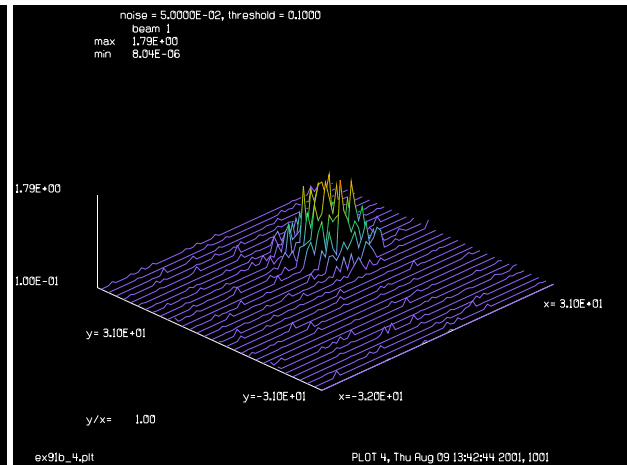


Fig. 91.4. Decentered gaussian with 5% noise added coherently, threshold is 10%.

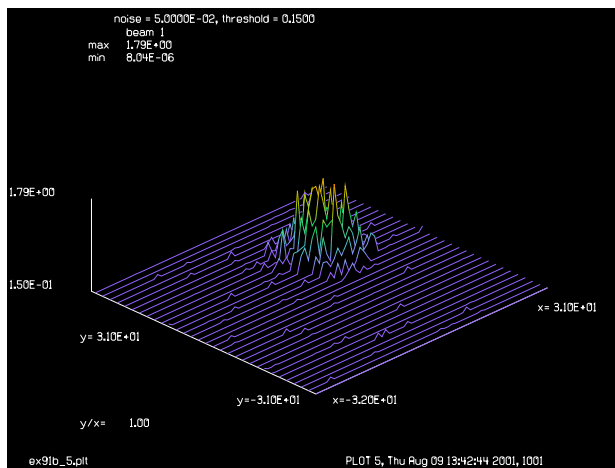


Fig. 91.5. Decentered gaussian with 5% noise added coherently, threshold is 15%.

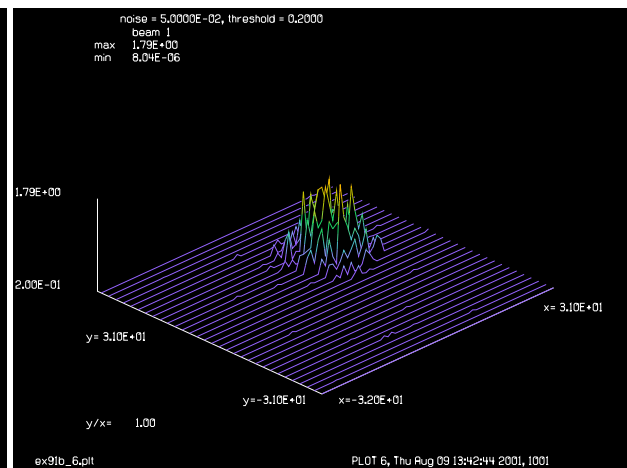


Fig. 91.6. Decentered gaussian with 5% noise added coherently, threshold is 20%.

c Example 91b: Measuring spot width using FITGEO with noise

c

c The principle problem in measuring the width of a beam

c is the sensitivity of most measures to noise.

c

```
variab/dec/int pass
```

```
gaus/c/c 1 1 10 decx=10 decy=10
```

```
pass = 0
```

```
threshold=0.
```

```
noise=0.
```

```
macro/def ex91b/o
```

```
    pass = pass + 1
```

```
    if pass = 2 then
```

```
        noise = .05
```

```
        noise 1 noise
```

```
    endif
```

```
    fitgeo 1 threshold=threshold
```

Jump to: [Commands](#), [Theory](#)

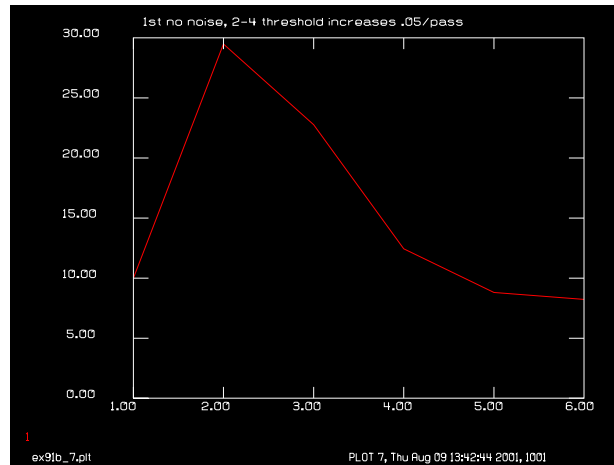


Fig. 91.7. Pass 1 shows the correct gaussian radius of 10. Pass 2 gives the radius at about 30 when 5% noise is added. Passes 3 to 6 have, 5%, 10%, 15%, and 20% thresholding. It is difficult to get a reliable measurement of beam size by using standard deviation if there is noise—even with thresholding—since it is difficult to know which threshold to use.

```

variab/set xc fitxcen list      # x- and y-centers
variab/set xs fitxsig list      # x and y standard deviation
plot/watch ex91b_@pass.plt
title noise = @noise, threshold = @threshold
plot/l 1 min=threshold
udata/set pass pass 2*xs xc
if pass >= 2 threshold = threshold + .05
macro/end
macro ex91b/6
udata/list
title 1st no noise, 2-4 threshold increases .05/pass
plot/watch ex91b_7.plt
set/grid 5 0 6 0
plot/udata min=0 max=30
end

```

Ex91c: Power-in-the-bucket

The diameter of a circular aperture may be fit to a distribution to achieve specified energy, using the optimization feature.

Input: ex91c.inp

```

c## ex91c
c
c Example 91c: Power-in-the-bucket, circular aperture with optimization
c
nbeam 2
gaus/c/c 1 1 10
energy/norm
copy 1 2
mac/def step/o

```

Jump to: [Commands](#), [Theory](#)

```

copy 2 1
clap/c/c 1 width
variab/set Energy 1 energy
mac/end
variab/set Units 1 units
opt/name step
opt/var/add width .5*Units
opt/tar/add Energy [1.-exp(-2.)]
width = 6
write/off
opt/run 6
write/on
echo/on
c
c Should be 10
c
width=
opt/tar
end

```

Ex91d: Fitting Hermite-Gaussian

As illustrated in Ex91d, thresholding is an unreliable means of removing the effects of noise. A method which does work consists of fitting a set of Hermite gaussians to the distribution and then calculating M^2 from the noise-free data based on the set of polynomials. Hermite gaussian polynomials of order (1,0) and (0,1) of 50% magnitude gives an M^2 value of 1.026 in the absence of noise. With only 0.1% noise, a direct fit of M^2 gives a value of 1.4—a huge error. Using optimization to fit the magnitude of the three lowest order Hermite gaussians to the noisy data, gives the values as shown in Table 91.2.

Table. 91.2. Result of optimization to fit the three lowest order Hermite modes.

input parameters	Fit data with no noise	Fit data with noise
peak for HG order (0,0), ideal 1.0	1.014	1.009
peak for HG order (1,0), ideal 0.5	0.489	0.493
peak for HG order (0,1), ideal 0.5	0.489	0.493
M^2 , ideal 1.026, with noise 1.4	1.024	1.025

Input: ex91d.inp

```

c## ex91d
c
c Example 91d: Fitting Hermite-Gaussian
c
c The principle problem in measuring the width of a beam
c is the sensitivity of most measures to noise.
c
nbeam 3
clear 1 0
hermite/c 1 1 10 10 0 0
hermite/c 2 .5 10 10 1 0
add/inc/con 1 2

```

Jump to: [Commands](#), [Theory](#)

```
hermite/c 2 .5 10 10 0 1
add/inc/con 1 2
title Hermite gaussian set without noise
plot/watch ex91d_1.plt
plot/l 1
fitgeo/msquared 1
peak 1
variab/set peak peak
alpha = 2
beta = 1
gamma = 1
point/list/xy/i 1 0 0
variab/set f1 point/i
point/list/xy/i 1 0 7
variab/set f2 point/i
point/list/xy/i 1 7 0
variab/set f3 point/i
point/list/xy/i 1 0 -7
variab/set f4 point/i
point/list/xy/i 1 -7 0
variab/set f5 point/i
first = 1
mac/def step/o
  clear 2 0
  hermite/c 2 alpha 10 10 0 0
  hermite/c 3 beta 10 10 1 0
  add/inc/con 2 3
  hermite/c 3 gamma 10 10 0 1
  add/inc/con 2 3
  point/list/xy/i 2 0 0
  variab/set s1 point/i
  point/list/xy/i 2 0 7
  variab/set s2 point/i
  point/list/xy/i 2 7 0
  variab/set s3 point/i
  point/list/xy/i 2 0 -7
  variab/set s4 point/i
  point/list/xy/i 2 -7 0
  variab/set s5 point/i
mac/end
mac/def optim/o
  pass = pass + 1
  opt/run 1
  variab/set xmerit merit
  udata/set pass pass log10(xmerit)
macro/end
opt/name step
opt/var/add alpha .001
opt/var/add beta .001
opt/var/add gamma .001
opt/tar/add s1 f1
opt/tar/add s2 f2
opt/tar/add s3 f3
opt/tar/add s4 f4
```

Jump to: [Commands](#), [Theory](#)


```

opt/tar/add s5 f5
write/off
mac optim/5
write/on
mult/mode/corr 1 2
fitgeo/msquared 2
variab/set msquared msqx
echo/on
alpha=      # should be      1.000
beta=       # should be      .500
gamma=      # should be      .500
msquared=   # should be      1.026
pause
clear 2 0
noise 2 peak*.001                # add 0.1% noise and recalculate
add/inc/con 1 2
fitgeo/msquared 1
title Hermite gaussian set with noise, m**2 = 1.401
plot/watch ex91d_2.plt
plot/l 1 1
alpha = 2
beta = 1
gamma = 1
write/off
mac optim/5
write/on
mult/mode/corr 1 2
fitgeo/msquared 2
variab/set msquared msqx
echo/on
alpha=      # should be      1.000
beta=       # should be      .500
gamma=      # should be      .500
msquared=   # should be      1.026   was 1.401 after 0.1% noise added
end

```

Ex91e: Finding and displaying the region containing specified energy

There are various ways of plotting the region defined by the intensity level defined by the `fitlevel` command, as illustrated in EX91e.inp.

Input: ex91e.inp

```

c## ex91e
c
c Example 91e: Finding the region containing specified energy
c
c In this example, the command FITLEVEL is used to find the intensity
c such that all points with this or higher intensity values have
c a specified relative energy
c
array/set 1 256

```

Jump to: [Commands](#), [Theory](#)

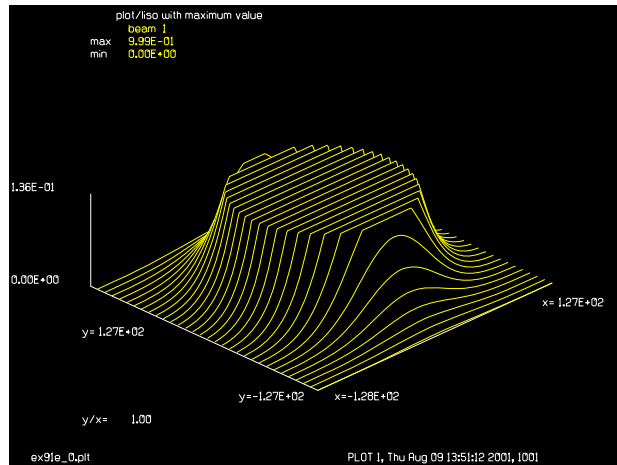


Fig. 91.8. Display of truncated distribution with plot/1 and maximum value: EX91E_0.PLT.

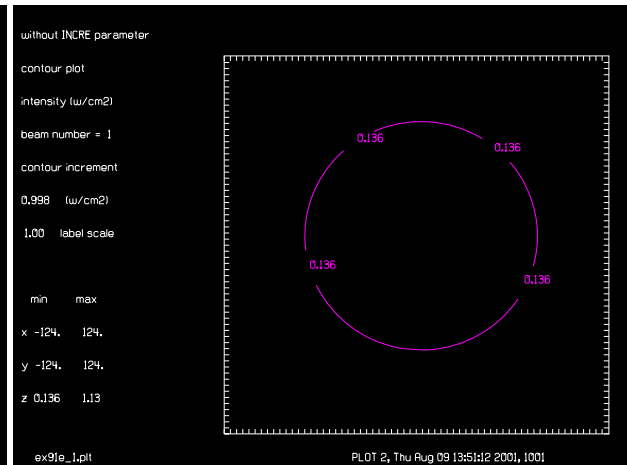


Fig. 91.9. Contour plot: EX91E_1.PLT, without parameter increment.

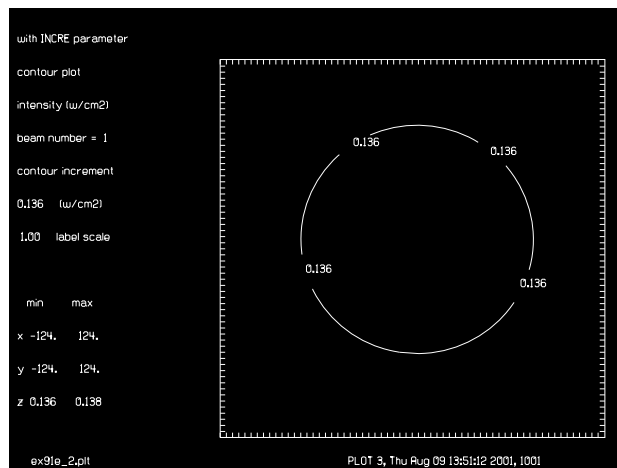


Fig. 91.10. Contour plot: EX91E_2.PLT, with parameter increment.

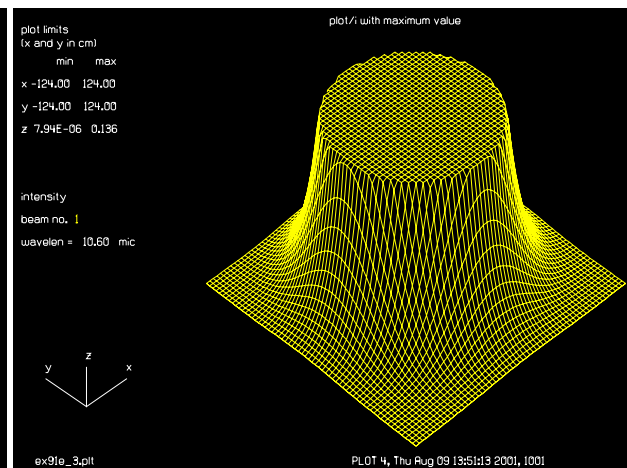


Fig. 91.11. Isometric plot: EX91E_3.PLT with maximum value specified.

```

gaus/c/c 1 1 75 1 3. 6.
time/i
fitlevel 1

```

```

time
variab/set fl fitlevel list
variab/set xw fitxfwhm list
variab/set yw fityfwhm list
variab/set xc fitxcen list
variab/set yc fitycen list
variab/set xr fitxrad list
variab/set yr fityrad list
peak 1
variab/set peak 1 peak list
set/density 64
plot/watch ex91e_0.plt

```

```

# find level for default relative energy
# of .865 corresponding to 1/e^2 intensity
# for a gaussian beam

```

```

# display intensity level, fl
# full-width-half-maximum of region (x)
# full-width-half-maximum of region (y)
# x- and y-centers

```

```

# x- and y-average radii

```

```

# set peak value

```

Jump to: [Commands](#), [Theory](#)

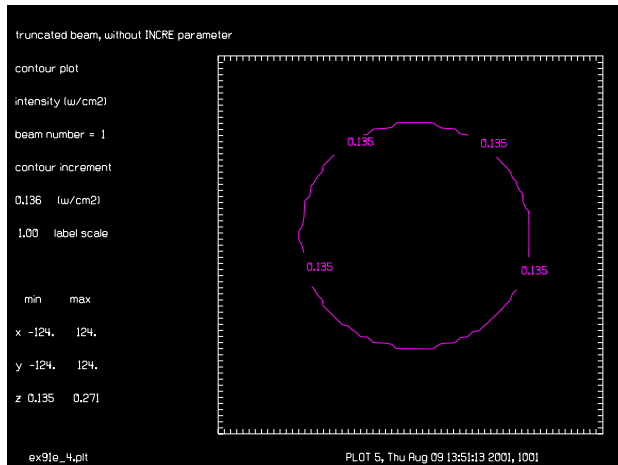


Fig. 91.12. Contour plot: EX91E_4.PLT, with intensity truncation, without increment.

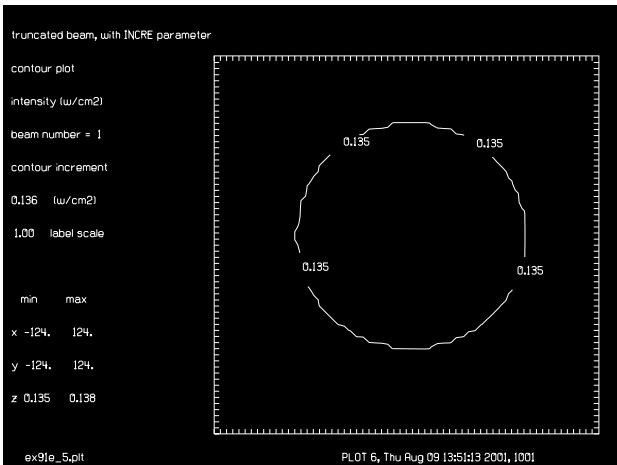


Fig. 91.13. Contour plot: EX91E_5.PLT, with intensity truncation, with increment.

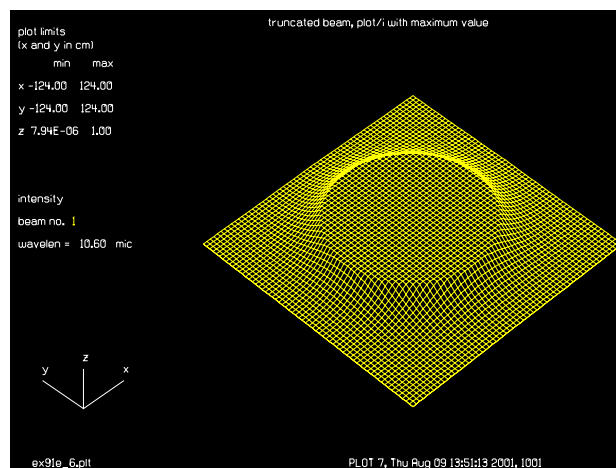


Fig. 91.14. Isometric with plot/i iso after intensity truncation: EX91E_6.PLT.

```

title plot/liso with maximum value
plot/liso 1 max=f1
c
c Make contour plot two different ways.
c
plot/watch ex91e_1.plt
title without INCRE parameter
plot/c 1 thres=f1/peak*100 con=1
plot/watch ex91e_2.plt
title with INCRE parameter
plot/c 1 min=f1 max=1.01*f1 con=f1 incre
plot/watch ex91e_3.plt
title plot/i with maximum value
plot/i 1 max=f1
fitlevel/truncate # truncate distribution to specified level
c
c Make contour plot two different ways.
c

```

```

plot/watch ex91e_4.plt
title truncated beam, without INCRE parameter
plot/c 1 thres=99 peak=100 con=1
plot/watch ex91e_5.plt
title truncated beam, with INCRE parameter
plot/c 1 min=.99*f1 max=1.01*f1 con=f1 incre
plot/watch ex91e_6.plt
title truncated beam, plot/i with maximum value
plot/i 1 max=peak
end

```

Ex91f: Fitting embedded gaussian to data set

GLAD can calculate a gaussian beam which best fits a set of data consisting of transverse radii at various axial positions. GLAD performs a damped least squares fit to the data using, the axial position of the waist, waist radius, and effective wavelength as optimizing variables. The ratio of the effective wavelength (from the data fit) to the true wavelength gives a measure of M^2 . The data set may be specified by the user, as is done in Ex. 91f, or by fitting the transverse radius to a propagating beam using one of the fitting routines: `fitfwhm`, `fitgeo`, `fitknife`, `fitlevel`, or `encircled/level`, as is shown in Exs. 91g and 91h.

Input: ex91f.inp

```

c## ex91f
c
c Example 91f: Fitting embedded gaussian to data set
c
c GLAD can calculate a gaussian (sometimes called the embedded gaussian)
c which best fits a data set consisting of axial positions and transverse
c radii at these positions. The waist size, axial position of the waist,
c and effective wavelength are solved to best fit the data.
c
c Case 1: automatic fit to ideal data
c Case 2: automatic fit to aberrated data and setting geodata
c Case 3: fit from calculated data
c Case 4: fit data set calculated with fitmsquared
c
c
c Case 1: Using the automatic fit
c
Lambda = 2.e-4
wavelength/set 1 Lambda*1e4
w0 = 1.e-2
units/s 1 w0/10
gaus/c 1 1 1e-2
fitegaus/automatic/list 1
pause
fitegaus/reset
c
c Case 2: Automatic fit to aberrated data and setting geodata
c

```

Jump to: [Commands](#), [Theory](#)

```

Lambda = 2.e-4
wavelength/set 1 Lambda*1e4
w0 = 1.e-2
units/s 1 w0/10
gaus/c 1 1 1e-2
phase/random 1 .2 .003          # add random phase
fitegaus/automatic/setgeodata 1  # set surrogate gaussian beam
geodata 1
c
c Case 3: Make a data set consisting of four ideal data points
c
fitegaus/reset
w0 = 1.e-2
Lambda = 2.e-4
zr = pi*w0^2/Lambda
wavelength/set 1 Lambda*1e4
c
c Calculate ideal data set
c
z = -.4; w = w0*sqrt(1+(z/zr)^2); fitegaus/data 1 -.4 w
z = .0; w = w0*sqrt(1+(z/zr)^2); fitegaus/data 2 .0 w
z = .4; w = w0*sqrt(1+(z/zr)^2); fitegaus/data 3 .4 w
z = .8; w = w0*sqrt(1+(z/zr)^2); fitegaus/data 4 .8 w
fitegaus/calc
variab/set z fitegaus/z list
variab/set w fitegaus/waist list
variab/set l fitegaus/l list
pause
fitegauss/reset
c
c Case 4: Fit data set calculated with fitmsquared
c
gaus/c 1 1 w0
prop -.4; z1 = -.4
fitmsquared/spatial 1
variable/set w1 fitxomega
fitegaus/data 1 z1 w1          # data set 1
prop .4; z1 = z1 + .4
fitmsquared/spatial 1
variable/set w1 fitxomega
fitegaus/data 2 z1 w1          # data set 2
prop .4; z1 = z1 + .4
fitmsquared/spatial 1
variable/set w1 fitxomega
fitegaus/data 3 z1 w1          # data set 3
prop .4; z1 = z1 + .4
fitmsquared/spatial 1
variable/set w1 fitxomega
fitegaus/data 4 z1 w1          # data set 4
fitegaus/calc                  # calculate embedded gaussian
variab/set z fitegaus/z list
variab/set w fitegaus/waist list
variab/set l fitegaus/l list

```

Jump to: [Commands](#), [Theory](#)

Ex91g: Fitting embedded gaussian, noise and aberration

GLAD can calculate the embedded gaussian—the gaussian which best fits a set of transverse radii values versus axial position. In Ex91g, a simple gaussian beam is propagated and the effective 1/e width is computed using, in order, `fitgeo` (1), `encircled/level` (2), `fitlevel` (3), `fitknife` (4), and `fitfwhm` (5). In the absence of noise and aberration, all methods show good agreement as shown in Fig. 15. With 0.01% noise `fitgeo`, which uses standard deviation, is very inaccurate but the other measures, based on measuring central lobe width are in good agreement. Adding 0.05 wave rms of aberration, which has a Strehl ratio of 0.9, is diffraction-limited also causes the calculation from `fitgeo` to be very inaccurate. The central lobe measurements are better but still show considerable variation. The zone commands are needed to improve the `fitgeo` calculation, by avoiding units change. The central lobe measures are not significantly affected by units change.

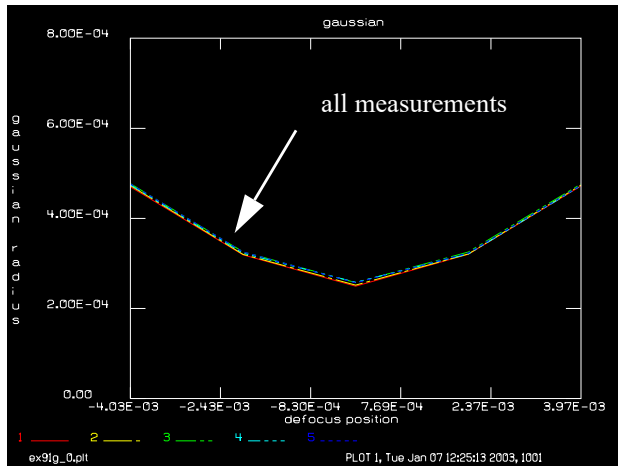


Fig. 91.15. Through-focus beam size with five different methods for simple gaussian beam: `fitgeo` (red), `pib/level` (yellow), `fitlevel` (green), `fitknife` (cyan), `fitfwhm` (blue).

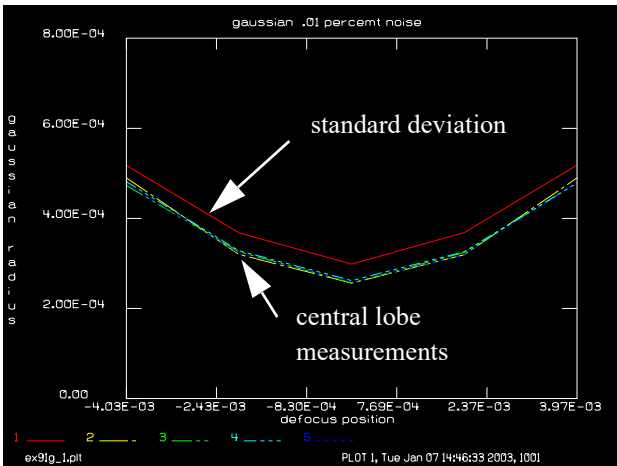


Fig. 91.16. Through-focus beam size with five different methods for gaussian beam with 0.01% noise added. The measurement by `fitgeo`, using standard deviation is not very accurate.

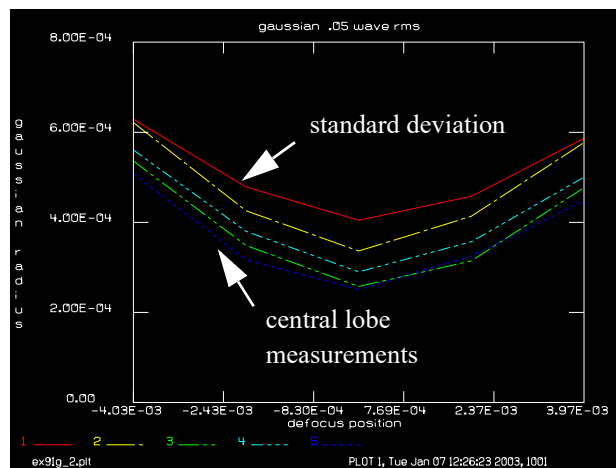


Fig. 91.17. Through-focus beam size with 0.05 waves rms of random aberration. The measurement by `fitgeo`, using standard deviation is not very accurate.

Input: ex91g.inp

```

c## ex91g!164688299188894
c
c Example 91g: Fitting embedded gaussian, noise and aberration
c
c GLAD can calculate a gaussian (sometimes called the embedded gaussian)
c which best fits a data set consisting of axial positions and transverse
c radii at these positions. The waist size, axial position of the waist,
c and effective wavelength are solved to best fit the data.
c
c Make a data set consisting of four data points
c
variab/dec/int switch icol point ntimes
c
c switch = 0, simple gaussian, TEM00
c switch = 1, gaussian with noise
c switch = 2, gaussian with aberration
c
switch = 1
macro/def pass/o
  prop dz 1
  @command
  variab/set fitxomega fitxomega
  variab/set z 1 zreff
  udata/set [point=point+1] [z-100] y0@icol=[fitxomega]
  fitegauss/data point z
macro/end
dz = .002
ntimes = 5
array/s 1 128
nbeam 1
clear 1 0
wavelength/set 0 .8
hermite/c 1 1 10 10 0 0
if switch = 0 then
  alias tag
endif
if switch = 1 then
  peak 1
  variab/set peak 1 peak
  noise 1 peak*.0001
  alias tag , .01 percent noise
endif
if switch = 2 then
  phase/random 1 .05 1.5
  alias tag , .05 wave rms
endif
lens 1 100
zstart = -3*dz
zone/fix 100 3.1*dz
zone/in 1
prop 100+zstart 1
c

```

Jump to: [Commands](#), [Theory](#)

```
c  Fit with FITGEO
c
alias command fitgeo 1
icol = 1
fitegauss/reset
point = 0
mac/run pass/ntimes
c fitgeo
fitegauss/calc
prop -ntimes*dz 1
c
c  Fit with PIB/LEVEL
c
alias command pib/level 1
icol = 2
fitegauss/reset
point = 0
mac/run pass/ntimes
c pib/level
fitegauss/calc
prop -ntimes*dz 1
c
c  Fit with FITLEVEL
c
alias command fitlevel 1
icol = 3
fitegauss/reset
point = 0
mac/run pass/ntimes
c fitlevel
fitegauss/calc
prop -ntimes*dz 1
c
c  Fit with FITKNIFE
c
alias command fitknife 1
icol = 4
fitegauss/reset
point = 0
mac/run pass/ntimes
c fitknife
fitegauss/calc
prop -ntimes*dz 1
c
c  Fit with FITFWHM
c
alias command fitfwhm 1
icol = 5
fitegauss/reset
point = 0
mac/run pass/ntimes
c fitfwhm
fitegauss/calc
plot/watch ex91g_@switch.plt
```

Jump to: [Commands](#), [Theory](#)


```

title gaussian@tag
set/grid 5 0 4 0
plot/udata/xlabel defocus position
plot/udata/ylabel gaussian radius
plot/udata/set y01 y02 y03 y04 y05
plot/udata/seq min=0 max=8e-4 dash
end

```

Ex91h: Fitting embedded gaussian, noise and aberration

The calculations of Ex. 91g are repeated for an ensemble of three Hermite gaussians. It is assumed that these represent three independent transverse modes of a laser, so that we would observe the time-integrated irradiance. To simulate steady-state performance of these modes, each of which is at a slightly different optical frequency, the modes are propagated separately and the incoherent sum is formed after propagation. Fig. 18 shows the through-focus data using the five measures of transverse radius, with no noise and no aberration. `fitgeo`, using standard deviation does a very accurate job. The central lobe measurement methods predict radii that are too high and they do not agree well among them selves. When 0.1% noise is added the data derived from `fitgeo` is not very accurate, but the central lobe measurements are relatively unaffected. When 0.05 wave rms aberration (Strehl ratio = 0.9) is added, the standard deviation data is very inaccurate, similarly to the case of 0.01% noise.

A preliminary conclusion to be drawn from Ex 91h is that none of the M^2 methods do very well for an ensemble of Hermite gaussians if there is even negligible noise or very small amounts of aberration. The best approach seems to be polynomial fitting (as illustrated in Ex. 91d) combined with measurement of aberration by wavefront variance or Strehl ratio.

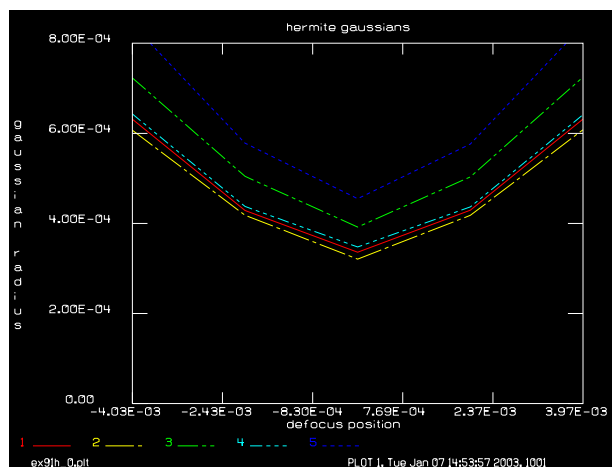


Fig. 91.18. Through-focus beam size with five different methods for simple gaussian beam: `fitgeo` (red), `pib/level` (yellow), `fitlevel` (green), `fitknife` (cyan), `fitfwhm` (blue).

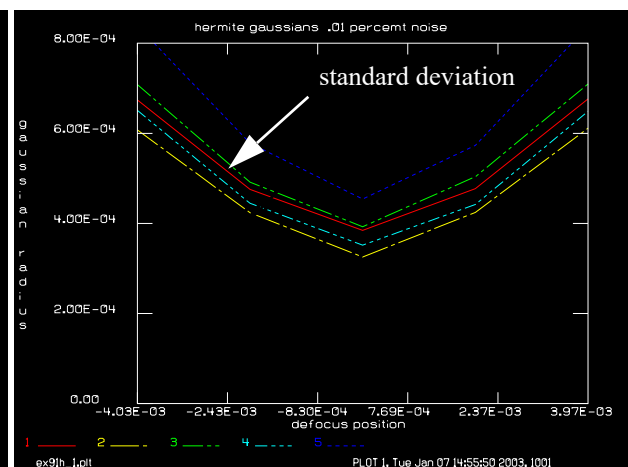


Fig. 91.19. Through-focus beam size with five different methods for gaussian beam with 0.01% noise added. The measurement by `fitgeo`, using standard deviation is not very accurate.

Input: `ex91h.inp`

```

c## ex91h
c

```

Jump to: [Commands](#), [Theory](#)

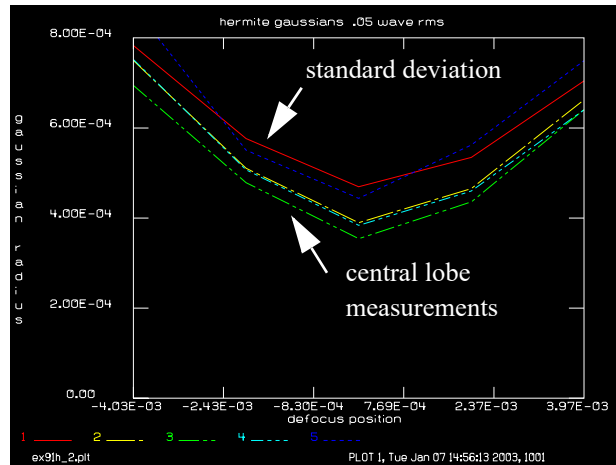


Fig. 91.20. Through-focus beam size with 0.05 waves rms of random aberration. The measurement by fitgeo, using standard deviation is not very accurate.

```

c Example 91h: Fitting embedded gaussian to a set of Hermite gaussians
c
c GLAD can calculate a gaussian (sometimes called the embedded gaussian)
c which best fits a data set consisting of axial positions and transverse
c radii at these positions. The waist size, axial position of the waist,
c and effective wavelength are solved to best fit the data.
c
c Make a data set consisting of four data points
c
variab/dec/int switch icol point ntimes
c
c switch = 0, ensemble of hermite gaussians
c switch = 1, hermite gaussians with noise
c switch = 2, hermite gaussians with aberration
c
switch = 2
macro/def pass/o
  prop dz 1 2 3
  variab/set units 1 units
  units/set 4 units
  clear 4 0
  add/inc 4 1 2 3
  @command
  variab/set fitxomega fitxomega
  variab/set z 1 zreff
  udata/set [point=point+1] z-100 y0@icol=fitxomega
  fitegauss/data point z
macro/end
dz = .002
ntimes = 5
array/s 1 128
nbeam 4
clear 1 0
wavelength/set 0 .8
hermite/c 1 1 10 10 0 0

```

Jump to: [Commands](#), [Theory](#)

```

hermite/c 2 1 10 10 1 0
hermite/c 3 1 10 10 0 1
lens 1 100
lens 2 100
lens 3 100
if switch = 0 then
    alias tag
endif
if switch = 1 then
    peak 1
    variab/set peak 1 peak
    noise 1 peak*.0001 seed=1
    noise 2 peak*.0001 seed=1
    noise 3 peak*.0001 seed=1
c
c zone calculations are needed for noise calculations because
c m-squared by variance is so sensitive to even small noise effects
c
    zone/fix 100 3.1*dz
    zone/in 1
    zone/in 2
    zone/in 3
    alias tag , .01 percent noise
endif
if switch = 2 then
    phase/random 1 .05 1.5
    phase/random 2 .05 1.5
    phase/random 3 .05 1.5
    alias tag , .05 wave rms
endif
zstart = -3*dz
prop 100+zstart 1 2 3
c
c Fit with FITGEO
c
alias command fitgeo 4
icol = 1
fitegauss/reset
point = 0
mac/run pass/ntimes
c fitgeo
fitegauss/calc
prop -ntimes*dz 1 2 3
c
c Fit with PIB/LEVEL
c
alias command pib/level 4
icol = 2
fitegauss/reset
point = 0
mac/run pass/ntimes
c pib/level
fitegauss/calc
prop -ntimes*dz 1 2 3

```

Jump to: [Commands](#), [Theory](#)

```

c
c  Fit with FITLEVEL
c
alias command fitlevel 4
icol = 3
fitegauss/reset
point = 0
mac/run pass/ntimes
c fitlevel
fitegauss/calc
prop -ntimes*dz 1 2 3
c
c  Fit with FITKNIFE
c
alias command fitknife 4
icol = 4
fitegauss/reset
point = 0
mac/run pass/ntimes
c fitknife
fitegauss/calc
prop -ntimes*dz 1 2 3
c
c  Fit with FITFWHM
c
alias command fitfwhm 4
icol = 5
fitegauss/reset
point = 0
mac/run pass/ntimes
c fitfwhm
fitegauss/calc
plot/watch ex91h_@switch.plt
title hermite gaussians@tag
set/grid 5 0 4 0
plot/udata/xlabel defocus position
plot/udata/ylabel gaussian radius
plot/udata/set y01 y02 y03 y04 y05
plot/udata/seq min=0 max=8e-4 dash
end

```

Ex91i: Calculation of best-fit gaussian as a function of lens spacing

Calculation of best-fit gaussian as a function of lens spacing using embedded gaussian fit.

Input: ex91i.inp

```

c## ex91i
#
# Calculation of best-fit gaussian as a function of lens spacing
#
alias Name ex91i

```

Jump to: [Commands](#), [Theory](#)

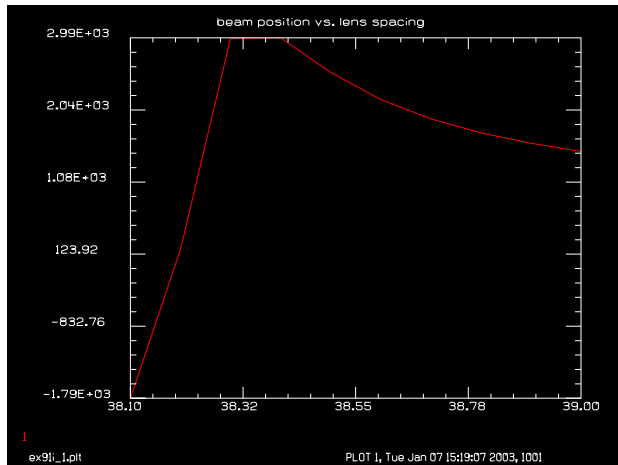


Fig. 91.21. Beam position vs. lens spacing.

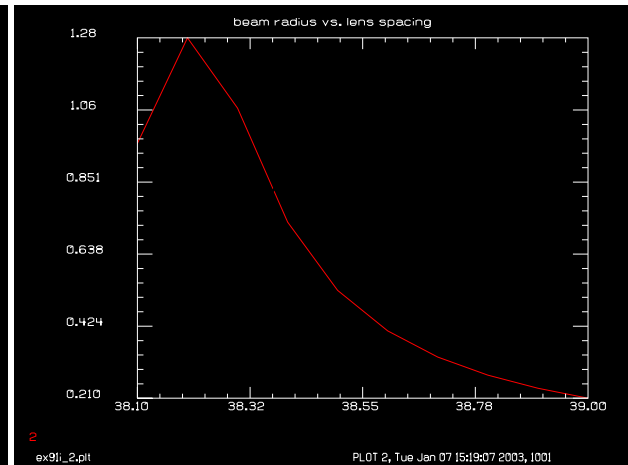


Fig. 91.22. Beam radius vs. lens spacing.

```

wavelength/set 1 10.6
array/s 1 128
gaus/c/r 1 10. .4872
f1=12.7      # first lens
f2=25.4      # 2nd lens
s1=626       # distance to 1st lens
ds=.1        # increment lens distance
dz=20        # axial step length
variab/dec/int ntimes
ntimes=0
macro/def system/o
c calculate system
  ntimes=ntimes+1
  s2=f1+f2+(ntimes-1)*ds
  zreff/se 1 0.                                # reset z postition
  gaus/c/r 1 10. .4872                          # reinitialize gaussian beam
  prop s1
  lens 1 f1
  clap/c/n 1 2.54
  prop s2
  lens 1 f2
  clap/c/n 1 2.54
  mac pass1
macro/end
macro/def pass1/o
c calculate embedded gaussian for each configuration
  focus/list
  focus/apply/waist
  prop -3*dz
  point = 0
  fitegaus/reset
  mac pass2/5
  fitegaus/calc
# store embedded gaussian position and waist radius
  variab/set bm_position 1 fitegauss/zwaist
  variab/set bm_size 1 fitegauss/waist

```

Jump to: [Commands](#), [Theory](#)

```
    udata/set ntimes s2 bm_position bm_size
macro/end
macro/def pass2/o
c scan through focus to find best-fit gaussian beam
    prop dz
    fitgeo 1
    variab/set fitxomega fitxomega
    variab/set z 1 zreff
    fitegauss/data [point=point+1] z fitxomega
macro/end
mac system/10
plot/w @Name_1.plt
title beam position vs. lens spacing
plot/udata 1
plot/w @Name_2.plt
title beam radius vs. lens spacing
plot/udata 2
```

Ex92: Thermal changes in refractive elements

Table. 92.1. Table of Ex92 examples

Ex92a: Two-dimensional heat flow, window, metal mount, air contact, internal heat source. . .	2
Ex92b: Two-dimensional heat flow, window, air contact, internal heat source.	5
Ex92c: Three-dimensional heat flow, point source of heat	7
Ex92d: Thermally induced aberrations in a window	9
Ex92e: Thermally induced change in optical power in a lens	10
Ex92f: Three-dimensional heat flow, YAG material	11
Ex92g: Thermally-induced stress birefringence	14
Ex92h: Simple model of thermal array and optical aberrations	18
Ex92i: Thermal distortion due to end pumping	19

This example illustrates thermal effects in refractive components. GLAD can calculate the optical aberrations due to variation of refractive index and material thickness with temperature. GLAD can also calculate heat flow due to conduction and convection. Chapter 19 of the GLAD Theory Manual explains the theory of heat flow and the concepts used in defining the material configuration. In summary, materials may be arranged in transverse sections with up to four different materials used in each section. The material distribution within a section is completely general as is the initial temperature distribution. Up to nine material sections may be defined to form an axial distribution. Heat conduction occurs in all three directions.

Materials may be of three types: optical materials, solid non-optical materials, and fluids. Optical materials may contain a distribution of internal heat generation to allow modeling of heat due to flash lamp pumping and similar effects. A library of materials is maintained in the file MATLIB. This library may be modified or extended by the user. Materials in the library are accessible by name, for example BK7 (glass), aluminum, forced air, etc. Optical materials are characterized by the thermal properties of heat conductivity and the product of density and specific heat as well as index of refraction and first and second derivatives of index of refraction and coefficient of expansion with respect to temperature. Solid non-optical materials are characterized only by thermal properties. Fluids are characterized by a coefficient of convection, which may be a variable distribution, and a temperature distribution. In GLAD, fluids are assumed to maintain their temperature—giving up or accepting heat as needed to preserve their temperature distribution. Thermal energy is assumed to not be conserved in fluid in this model. This assumption implies that flow of the fluid is adequate to avoid heat build-up.

The usual GLAD arrays are used to represent the material sections. However, the data represented by the complex words of the array is different. In regular beam arrays, the real and imaginary components of the array represent the complex amplitude of an optical field. For thermal arrays, the real word represents the local temperature. Wherever the temperature is zero, GLAD assumes a perfect insulator exists. The user may work in Celsius or Kelvin as convenient. If temperatures are in Celsius, care must be taken that zero or negative temperatures do not occur. Kelvin temperatures, can, of course, never have non-positive temperatures. The imaginary word of the array is used to identify the material by the order of its definition with the `thermal/material/add` command. Up to four materials may be defined per thermal section. For optical materials the imaginary word also contains the internal heat source distribution. For fluids, the imaginary word contains the distribution of changes of the convection constant with respect to the mean value as defined in the `thermal/material/add` command. This allows cooling or heating by convection to be variable. The least significant two bits of the imaginary word are used to define the material

number. The rest of the mantissa is used for the internal heat source for optical materials or convection constant distribution for fluids.

The commands `plot/liso/real`, `plot/liso/aimag`, and `plot/liso/material` may be used to display the real word, imaginary word, and material number distributions. The field and point commands may be used to display real and imaginary words.

Ex92a: Two-dimensional heat flow, window, metal mount, air contact, internal heat source

Example 92a illustrates a glass window surrounded by an aluminum ring cooled by forced air. The `thermal/material/add` command is used to combine the distributions for the three materials into one thermal array. GLAD looks up the material properties from MATLAB based on the name of the material specified. Figure 92.1 shows the distribution of materials using the `plot/liso/material` command. The first material defined is displayed with the value 4, the second with 3, etc.

Figure 92.2 shows that both the glass window and aluminum ring are initially at 30 and the air is at 25. The effects of thermal conduction and convection are implemented with the `thermal/settle` command. The forced air flow begins cooling the material lowering the temperature from the outside inward. An internal heat source generates heat in the center of the window. The concept of time constants are discussed in Chap. 19, GLAD Theory Manual. For 0.1 cm spacing between array points, the thermal time constants for the materials are: glass, 2 sec; aluminum, 0.0043 sec; forced air, 12.2 sec. It is not necessary to fully resolve the time constant of the aluminum. Because the aluminum has such high thermal conductivity compared to glass and air convection, the slope of the temperature distribution is nearly zero in the aluminum. We can take time steps longer than the 0.0043 time constant for aluminum provided the slope of the temperature distribution over the aluminum is still essentially zero. By setting the variable `ntimes = 10`, the incremental time is 0.1 seconds. We can see that in all of the temperature distributions that the slope is flat across the aluminum ring. Figures 92.3-92.6 show increasing diffusion of heat into the air. The effect of the internal heat source is barely observable at 5 seconds in Fig. 92.3 but is more evident at 20 seconds. In Fig. 92.6, at 150 seconds, the internal heat source and the air cooling are in a steady-state condition. The aluminum ring does not significantly alter the steady-state thermal distribution except in the immediate vicinity of the edge.

Input: ex92a.inp

```
c## ex92a!441374482733151
c
c Examples 92a: Two-dimensional heat flow, window, metal mount,
c             air contact, internal heat source
c
c inner 4.4 cm diameter is BK7 glass
c a .4 cm thickness annular ring of aluminum outside the glass
c surrounding the aluminum is forced air
c
c the thermal constants between matrix points are
c glass          2.0 seconds
c aluminum       0.0043 seconds
c forced air     12.2 seconds
c
```

Jump to: [Commands](#), [Theory](#)

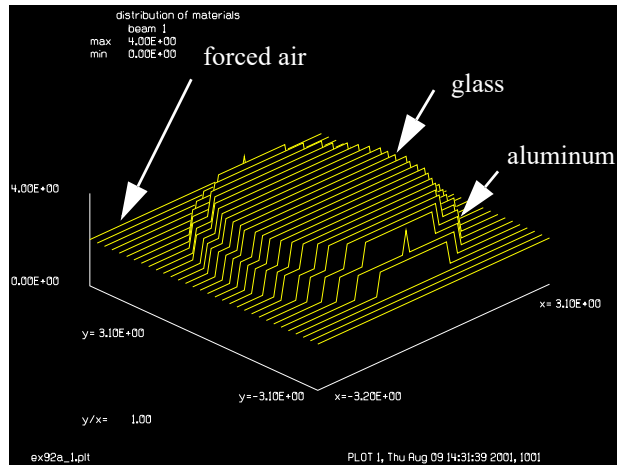


Fig. 92.1. Display of material numbers with plot/1/mat. First material is 4. The second is 3.

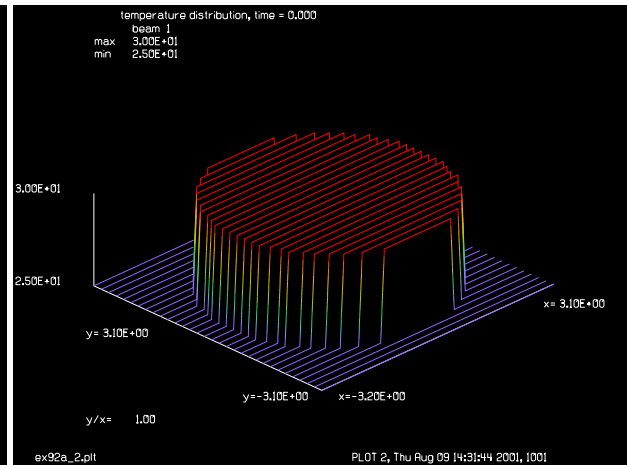


Fig. 92.2. Temperature distribution displayed by plot/1/r. 30° in glass and metal, 25° in air.

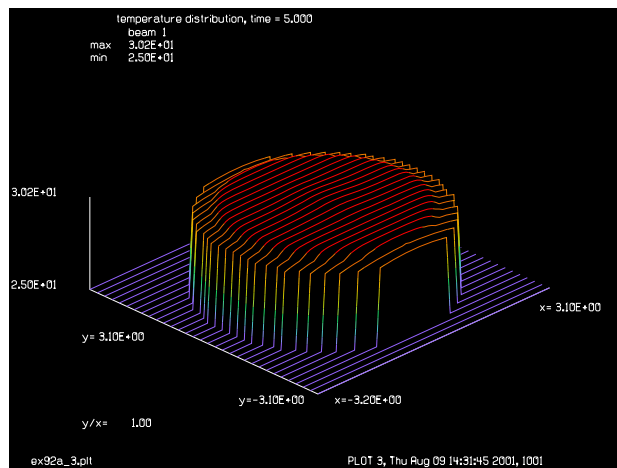


Fig. 92.3. Temperature after 5 seconds. Glass is starting to cool near the metal ring.

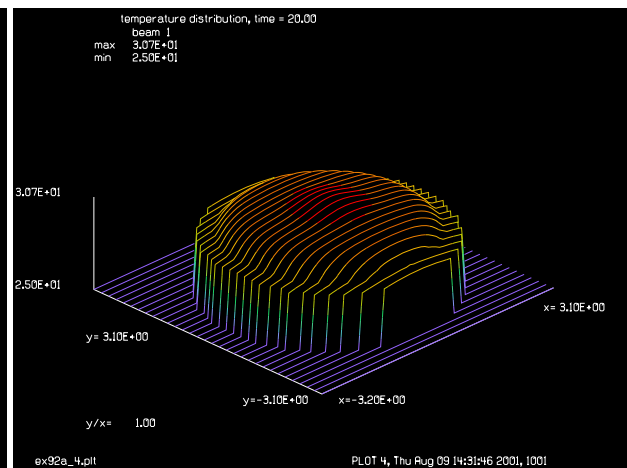


Fig. 92.4. Temperature at 20 seconds. Edge cooling is clear. Note internal heat source in the center.

```

c The thermal conductivity of aluminum is so high that it will exhibit
c negligible temperature gradients. Since the time constant for glass
c is about 6 times faster than aluminum. The temperature of the
c aluminum will be proportionally closer to the glass than the forced air
c
variab/dec/int plot ntimes
nbeam 2                                # establish two beams
units/s 0 .1                          # units are 0.1 cm
c
c set up material 1 for thermal surface as glass, BK7
c
clear 1 30                             # set beam 1 to 30 deg.
clap/c/c 1 2.2                         # circular aperture of 2.2 cm
thermal/mat/add 1 1 BK7                # establish a thermal array for beam 1
c
c set up material 2 for thermal surface as aluminum
c

```

Jump to: [Commands](#), [Theory](#)

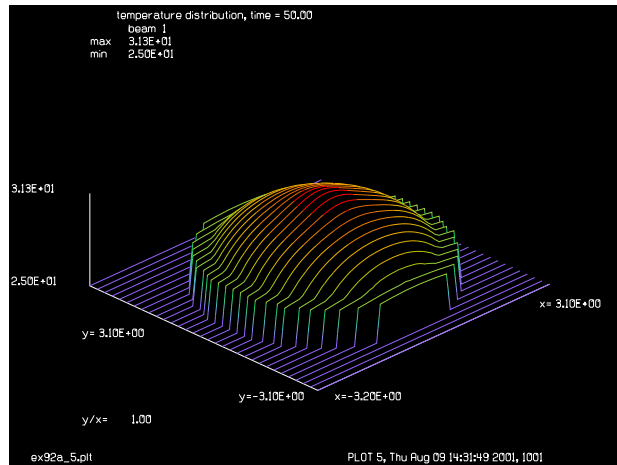


Fig. 92.5. Temperature after 50 seconds, approaching steady-state.

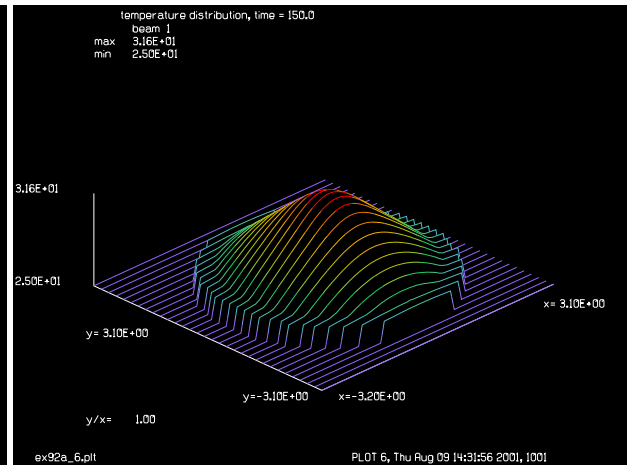


Fig. 92.6. Temperature at 150 seconds at steady-state.

```

clear 2 30
clap/c/c 2 2.6
thermal/mat/add 1 2 aluminum
c
c set up internal source
c
gaus/c/c 2 .1 1
thermal/mat/source/square 1 2
clear 2 25
thermal/mat/add 1 2 forced_air
plot/watch ex92a_1.plt
title distribution of materials
plot/l/mat
thermal/mat/list
pause 5
c
c Use a macro to settle for 1 second. Integration increment is .2 second.
c
total_time = 0
plot = 2
title temperature distribution, time = @total_time
plot/watch ex92a_@plot.plt
plot/l/r
mac/def therm/o
    therm/settle 1 time nstep=0 # Use automatic number of steps
    total_time = total_time + time
    title temperature distribution, time = @total_time
    plot = plot + 1
    plot/watch ex92a_@plot.plt
    plot/l/r
mac/end
time = 5
mac therm
time = 15
mac therm

```

Jump to: [Commands](#), [Theory](#)

```

time = 30
mac therm
time = 100
mac therm
end

```

Ex92b: Two-dimensional heat flow, window, air contact, internal heat source

Since the aluminum ring complicates the problem in Ex92a, we rerun the example without the ring. Figure 92.12 shows that the steady-state solution is essentially identical to Fig. 92.6 except in the immediate vicinity of the edge. We used a time step of 0.5 seconds—cutting the calculation time by five without the metal ring. The transient temperature distributions in Figs. 92.8-92.11 show some differences. The steady-state distribution at 150 seconds in Fig. 92.12 is, except near the edge, very close to the steady-state solution with the metal ring as shown in Fig. 92.6.

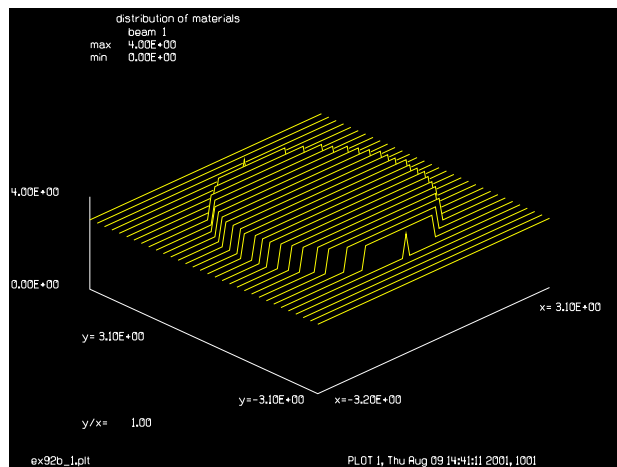


Fig. 92.7. Display of material numbers with `plot/1/mat`. First material is 4. The second is 3.

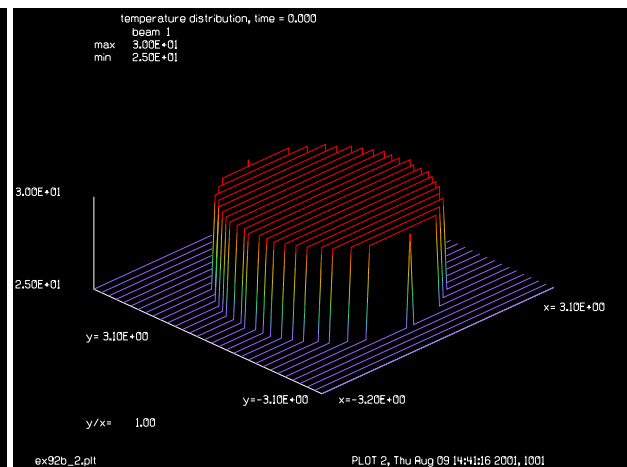


Fig. 92.8. Temperature distribution displayed by `plot/1/r`. 30° in glass and metal, 25° in air.

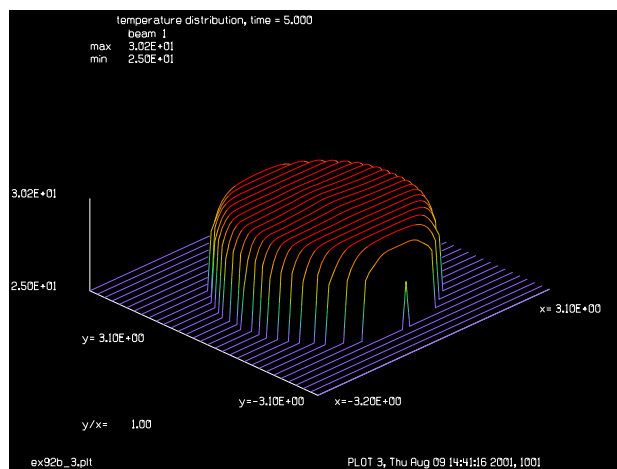


Fig. 92.9. Temperature after 5 seconds. Glass is starting to cool at the edge.

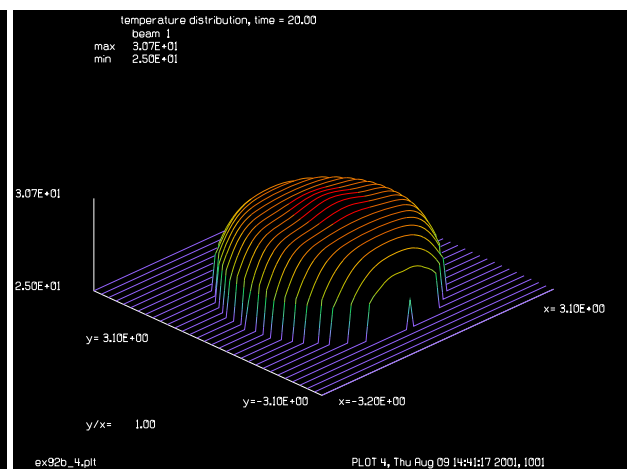


Fig. 92.10. Temperature at 20 seconds. Edge cooling is clear. Note internal heat source in center.

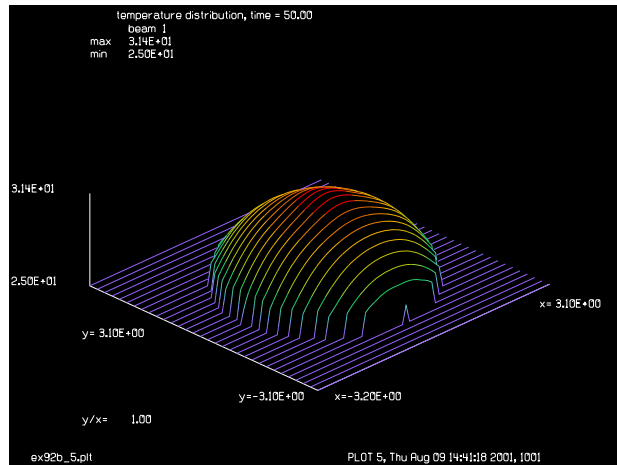


Fig. 92.11. Temperature after 50 seconds, approaching steady-state.

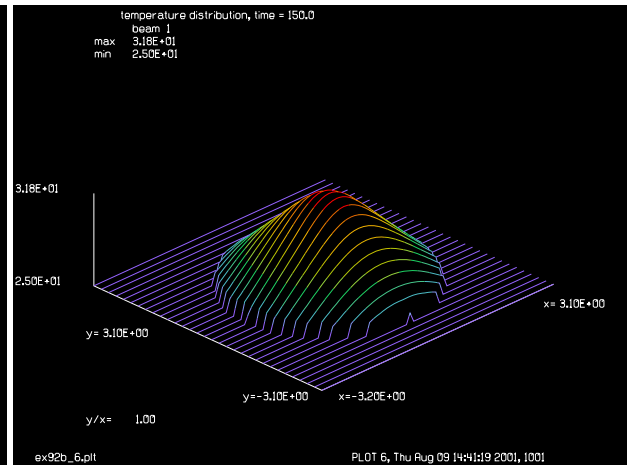


Fig. 92.12. Temperature at 150 seconds at steady-state. Compares with Fig. 92.6.

Input: ex92b.inp

```

c## ex92b
c
c Examples 92b: Two-dimensional heat flow, window, air contact,
c               internal heat source
c
c This example is the same as Ex92a except that the aluminum is deleted,
c because it is such a good thermal conductor, the system acts as if
c the forced air where in direct contact with the glass.
c
c inner 4.4 cm diameter is BK7 glass
c a .4 cm thickness annular ring of aluminum outside the glass
c surrounding the aluminum is forced air
c
c the thermal constants between matrix points are
c glass          2.0 seconds
c aluminum       0.0043 seconds
c forced air     12.2 seconds
c
variab/dec/int plot ntimes
nbeam 2                      # establish two beams
units/s 0 .1                 # units are 0.1 cm
c
c set up material 1 for thermal surface as glass, BK7
c
clear 1 30                    # set beam 1 to 30 deg.
clap/c/c 1 2.2                # circular aperture of 2.2 cm
thermal/mat/add 1 BK7         # establish a thermal array for beam 1
c
c set up internal source
c
gaus/c/c 2 .1 1
thermal/mat/source/square 1 2
clear 2 25

```

Jump to: [Commands](#), [Theory](#)

```

thermal/mat/add 1 2 forced_air
plot/watch ex92b_1.plt
title distribution of materials
plot/l/mat
thermal/mat/list
pause 5
c
c Use a macro to settle for 1 second. Integration increment is .2 second.
c
total_time = 0
plot = 2
title temperature distribution, time = @total_time
plot/watch ex92b_@plot.plt
plot/l/r
mac/def therm/o
    therm/set 1 time nstep=0 # use automatic calculation of steps
    total_time = total_time + time
    title temperature distribution, time = @total_time
    plot = plot + 1
    plot/watch ex92b_@plot.plt
    plot/l/r
mac/end
time = 5
mac therm
time = 15
mac therm
time = 30
mac therm
time = 100
mac therm
end

```

Ex92c: Three-dimensional heat flow, point source of heat

By using multiple thermal arrays arranged in a group, heat conduction in all three directions may be modeled. This is of importance for a thick window or rod (such as a solid state laser) where the primary cooling occurs at the optical faces. Up to nine thermal arrays may be defined and all defined thermal arrays may be combined into a group during thermal settling. In Ex 92c, a block of glass is defined. We show that a point source of heat is dissipated in all directions at an equal rate. Figures 92.13 and 92.14 show the transverse distribution (XY) and horizontal distribution (XZ) at $t = 0$. Figures 92.15 and 92.16 show the XY and XZ distributions at $t = 10$ seconds. The transverse and horizontal distributions are essentially identical. Ultimately there will be differences because there are only 9 axial samples and 16 transverse samples. The boundary of the array for the XY plane is assumed to be an insulator. Fluids on the front and back surfaces may be defined by the `thermal/ends` command or by making the beginning and ending thermal sections contain only fluids.

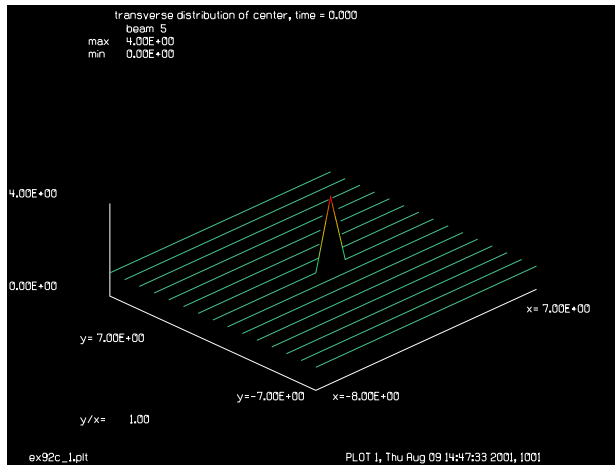
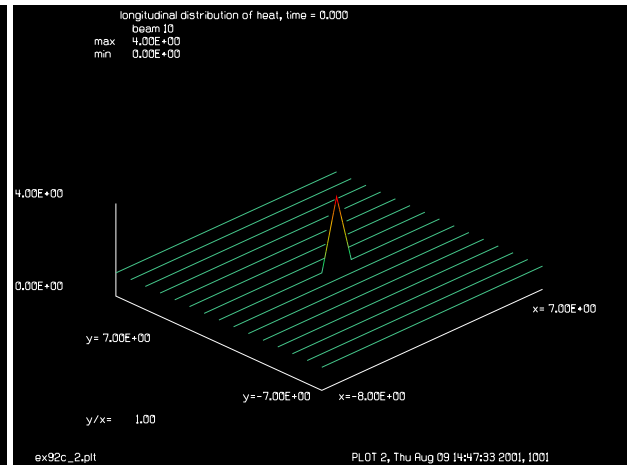
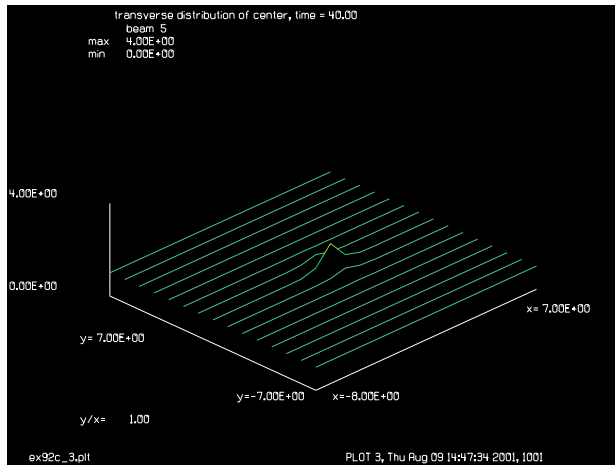
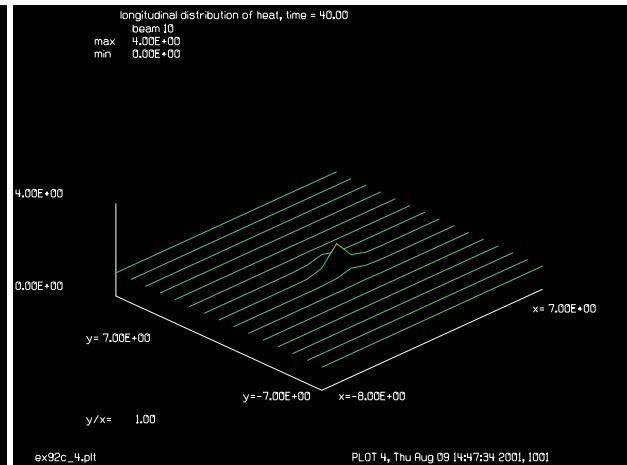
Input: ex92c.inp

```

c## ex92c
c

```

Jump to: [Commands](#), [Theory](#)

Fig. 92.13. XY temperature distribution at $t = 0$.Fig. 92.14. XZ temperature distribution at $t = 0$.Fig. 92.15. XY temperature distribution at $t = 10$ seconds.Fig. 92.16. XZ temperature distribution at $t = 10$ seconds.

c Example 92c: Three-dimensional heat flow, point source of heat

c

c This example illustrates three-dimensional heat flow. Nine thermal
c arrays are used. A point source of heat is injected into array 5
c and heat flows outward in the transverse and axial directions.

c

```
variab/dec/int row plot Nline
```

```
Nline = 16
```

```
Nc = Nline/2 + 1
```

```
array/s 1 Nline
```

```
nbeam 9
```

```
nbeam 10 Nline 16 # x-z scan
```

```
units/s 0 1
```

```
units/s 10 1 1
```

```
point/set 5 Nc Nc sqrt(4)
```

```
thermal/mat/add 1 1 1 BK7
```

```
thermal/mat/add 2 2 1 BK7
```

```
thermal/mat/add 3 3 1 BK7
```

Jump to: [Commands](#), [Theory](#)

```

thermal/mat/add 4 4 1 BK7
thermal/mat/add 5 5 1 BK7
thermal/mat/add 6 6 1 BK7
thermal/mat/add 7 7 1 BK7
thermal/mat/add 8 8 1 BK7
thermal/mat/add 9 9 1 BK7
mac/def ex92c/o
    therm/set/group time 1 1 2 3 4 5 6 7 8 9
    total_time = total_time + time
    row = 0
    mac step/9
mac/end
macro/def step/o
c   copy center row of all thermal arrays into beam 10
    row = row + 1 list
    copy/row row 10 9 row+4
macro/end
macro/def plot/o
    title transverse distribution of center, time = @total_time
    plot = plot + 1
    plot/watch ex92c_@plot.plt
    plot/l/r 5 min=0 max=4
    title longitudinal distribution of heat, time = @total_time
    plot = plot + 1
    plot/watch ex92c_@plot.plt
    plot/l/r 10 min=0 max=4
macro/end
time = 10
total_time = 0
plot = 0
c
c   Calculate initial appearance of longitudinal array
c
row = 0
mac step/9
macro plot
c
c   Allows 40 seconds to pass
c
macro ex92c/4
mac plot

```

Ex92d: Thermally induced aberrations in a window

The optical aberrations of a window depend on the temperature distribution and the optical constants define in the material library. See Chapt. 19, GLAD Theory Manual for a discussion of the aberrations due to nonuniform heating. In addition, thermal/window implements an absorption coefficient, if defined, with the absorbed light going into the window and raising the temperature.

$$\Delta OPD = \frac{2\pi}{\lambda_0} \left[\frac{dN}{dT} + (N-1)\alpha \right] L \Delta T \quad (92.1)$$

Jump to: [Commands](#), [Theory](#)

In this example, a gaussian temperature distribution, induces a gaussian shaped phase aberration into a window.

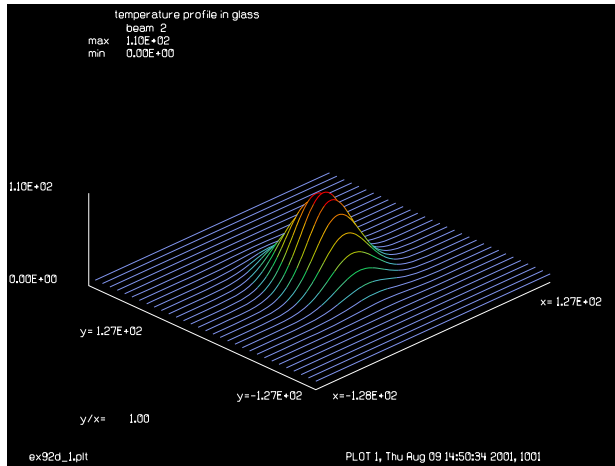


Fig. 92.17. Gaussian shaped temperature profile, min=10°, max=110°.

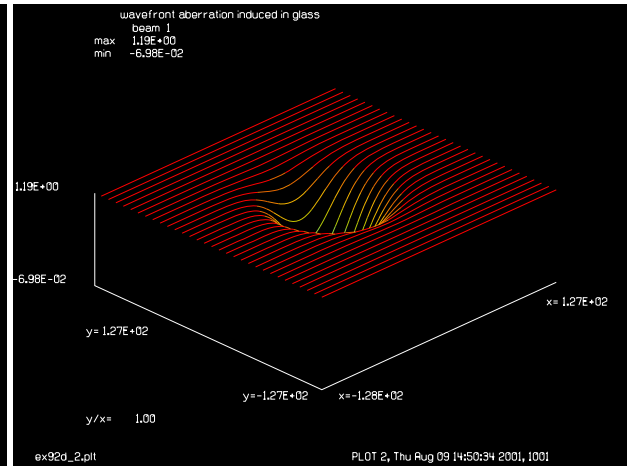


Fig. 92.18. Wavefront aberration due to temperature profile. The effect is dispersive.

Input: `ex92d.inp`

```
c## ex92d
c
c Example Ex92d: Thermally induced aberrations in a window
c
array/s 1 256
nbeam 3
c
c Form temperature distribution with gaussian profile
c
clear 2 10
gaus/c/c 3 100 50
add/inc/con 2 3
field 2
thermal/mat/add 2 2 2 BK7
title temperature profile in glass
plot/watch ex92d_1.plt
plot/l/r 2 min=0.
thermal/win 1 0. 2
title wavefront aberration induced in glass
plot/watch ex92d_2.plt
plot/l/w 1
end
```

Ex92e: Thermally induced change in optical power in a lens

The command `thermal/lens` includes thermal window effects with radii defined at both end faces. The optical power of the radii on the end faces is calculated using the mean temperature and the optical coefficients.

Jump to: [Commands](#), [Theory](#)

Input: ex92e.inp

```

c## ex92e
c
c  Example ex92e: Thermally induced change in optical power in a lens
c
echo/on
nbeam 3
clear 2 25
thermal/mat/add 2 2 1e-6 BK7
thermal/lens 1 0. 100. -100. 2
c
c  focal length at 25 degree should be 96.7118
c
geodata 1
pause
array/s 1 64
clear 2 30
thermal/mat/add 2 2 1e-6 BK7
pause
thermal/lens 1 0. 100. -100. 2
c
c  focal length at 30 degree should be 96.71245
c
c          dN  1      dL          3e-6
c f * (  --  ---  -  --- ) dT = f * (  ----  - 7.1e-6 ) 5 = -.00063
c          dT N-1    LdL          .517
c
c 97.71182 - -.00063 = 96.71245
c
geodata 1
pause
clear 2 25
array/set 1 64
gaus/c/c 3 10 20
add/inc/con 2 3
thermal/mat/add 2 2 1e-6 BK7
thermal/lens 1 0. 100. -100. 2
geodata 1
end

```

Ex92f: Three-dimensional heat flow, YAG material

A perfectly insulated Yb:YAG thin disc is being pumped with 10 Watts. Dimensions: $r = 0.25$ cm, $t = 0.1$ cm. The YAG windows is modeled with the `thermal/window` and `thermal/settle/group` commands. The volume of the disc is represented by five sheets. Sheet 1 is the first sheet exposed to the illumination and Sheet 5 is the last. Sheet 1 heats up more than Sheet 5 because of the absorption of light passing through the disk as illustrated in Fig. 92.20, which shows temperature rise over 12 time steps.

Input: ex92f.inp

```

c## ex92f

```

Jump to: [Commands](#), [Theory](#)

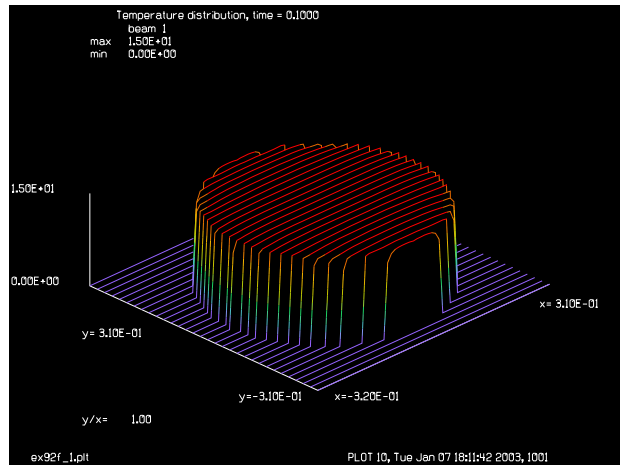


Fig. 92.19. Temperature after 12 time steps at front face.

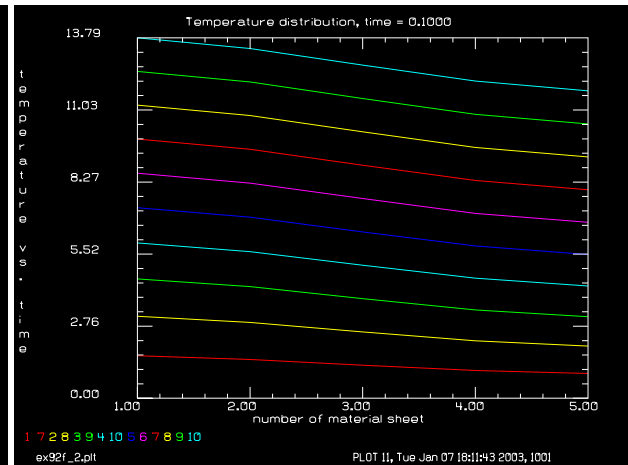


Fig. 92.20. Temperature for 12 time steps from Sheet 1 at front to Sheet 5 at the end. Note that a gradient appears from front to back as temperature increases.

```
#
# Example 92f: Three-dimensional heat flow, YAG material
#
# A perfectly insulated Yb:YAG thin disc is being pumped with 10 Watts.
# Dimensions : r = 0.25 cm., t = 0.1 cm.
# Volume = 0.0785 cm^3

# Material properties of Yb:YAG :

# Conductivity = 0.14 w/cm C
# Density*specific heat = 2.69 J/cm^3 C
# Index of refraction = 1.82
# Change in index with temp. = 7.3*10^-6
# Second order change = 0
# Expansion coefficient = 7.8*10^-6
# Coefficient of convection = 0
# Energy absorption per cm. = 10.8 /cm

# add the next line to matlab
# YbYAG      1 .14      2.69  1.82      25  7.3e-6  0.  7.8e-6  0.  10.8

# approximate solution
# -----
# Irradiance reduction in 0.1 cm, dI = I(0)*(1-exp(-alpha*length))
# I(0) = 52 W/cm^2
# alpha = 10.8 1/cm
# length = .1
# dI = 34.34 W/cm^2
# dT per second = dI/(density*specific heat)/length = 34.34/2.69/.1 = 127.7
# dT/sec = 127.7 deg/sec
# approximate increase in 0.1 sec is 12.8 deg.
# using detailed calculation below we see about 12.7 deg for .1 sec
# in the real value of Beam 3 which is in the middle of the disk
```

```

# material time constant = (density*specific heat)*Units^2/kappa
# density*specific heat = 2.69 J cm^3 C
# Units = .01
# kappa = .14 W/cm C

# material time constant = 1.9e-3 sec

# Beam 1    thermal array
# Beam 2    optical beam
# Beam 3    work array

variab/dec/int nstep time_step Array
array/s 0 64
nbeam 6 data
Units = .01
units/s 0 Units
time_constant = 2.69*Units^2/.14
wavelength/set 0 0.941 1.82 1.82

c this is supposed to be a tophat with peak 52 watts, and total energy 10 watts
c It will be used as thermal source.

# predefine thermal arrays with some small non-zero value
# (zero is an insulator, we set region outside .25 to be
# exactly zero)

clear 1 .0001
clap/c/c 1 .25
clear 2 .0001
clap/c/c 2 .25
clear 3 .0001
clap/c/c 3 .25
clear 4 .0001
clap/c/c 4 .25
clear 5 .0001
clap/c/c 5 .25
thermal/mat/add 1 1 .02 YbYAG
thermal/mat/add 2 2 .02 YbYAG
thermal/mat/add 3 3 .02 YbYAG
thermal/mat/add 4 4 .02 YbYAG
thermal/mat/add 5 5 .02 YbYAG
field 1

time = .01
nstep = time/time_constant

# set up x-axis
udata/set/add 1 0 1
udata/set/add 2 0 2
udata/set/add 3 0 3
udata/set/add 4 0 4
udata/set/add 5 0 5

```

Jump to: [Commands](#), [Theory](#)

```

macro/def set_curves/o
  Array = Array + 1
  point/list Array 33 33
  variab/set Point point/sr
  if [time_step<=12] then
    udata/set/address Array time_step Point    # set column count1
  endif
mac/end
mac/def therm/o
  time_step = time_step + 1
  gaus/c/c 6 52 0.25 100          # supergaussian heat source applied to Beam 6
  therm/window 6 time 1 2 3 4 5
  total_time = total_time + time
  title Temperature distribution, time = @total_time
  plot/watch ex92f_1.plt
  plot/l/r 1 max=15
  thermal/settle/group time nstep 1 2 3 4 5
  Array = 0; mac set_curves/5
mac/end
mac therm/10
total_time=
field 3
plot/udata/set y01 y02 y03 y04 y05 y06 y07 y08 y09 y10
plot/watch ex92f_2.plt
plot/udata/xlabel number of material sheet
plot/udata/ylabel temperature vs. time
plot/udata/seq min=0

```

Ex92g: Thermally-induced stress birefringence

The thermally-induced stress birefringence may be calculated by first forming a z-integrated index array and then using this index array to calculate the optical effects. Theoretical discussion of stress birefringence is found in Chap. 19, GLAD Theory Manual. The index of refraction effects include Young's Modulus, thermal expansion coefficient, and photoelastic tensor coefficients. In this treatment both isotropic and cubic crystals may be treated.

In Ex. 92g, we use the same three-dimensional problem as Ex. 92c and add `thermal/indexarray` and to form the z-integrated index array and `thermal/birefringence` to calculate the optical effects.

The temperature distribution calculations are identical to Ex92c. The command `thermal/indexarray` performs the z-integration of temperature and calculates both the stress and photoelastic calculations. Figs. 92.19 and 92.20 show the thermal distribution in the x-y and x-z planes after 40 seconds of settling. Figs. 92.21-92.22 show the resulting z-integrated N_x , N_y , and N_{xy} respectively. Fig. 92.24 shows a starting linear polarization state at 45 inclination. Fig. 92.25 shows the disruption in the polarization plot created by the hot spot in the center, see Fig. 92.15 for the same temperature data from Ex92c.

Input: ex92g.inp

```

c## ex92g
c

```

Jump to: [Commands](#), [Theory](#)

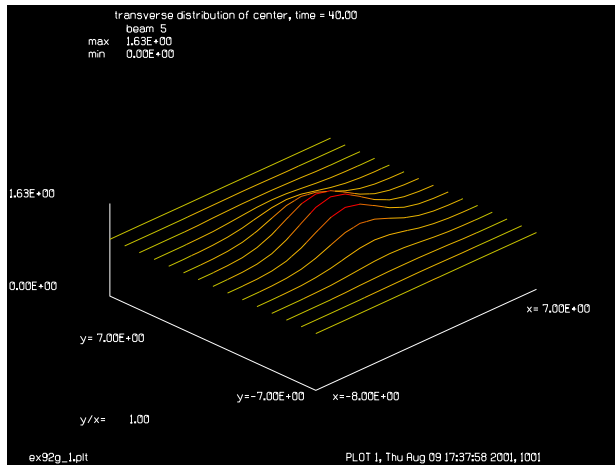


Fig. 92.21. Thermal distribution in center (x-y plane) after 40 seconds.

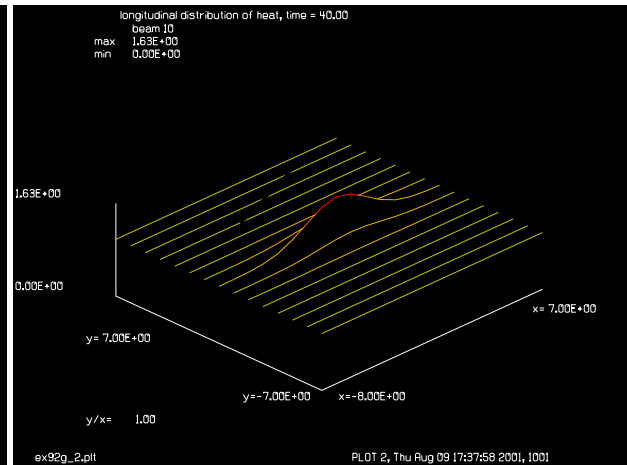


Fig. 92.22. Longitudinal temperature distribution (x-z plane) after 40 seconds.

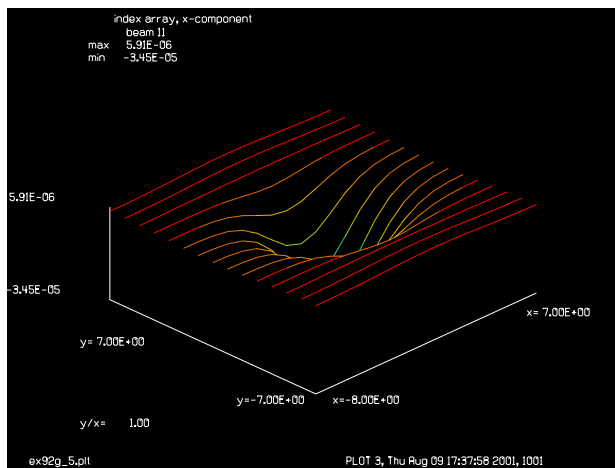


Fig. 92.23. Z-integrated N_x for ex92g. The thermal response is identical to Ex92c.

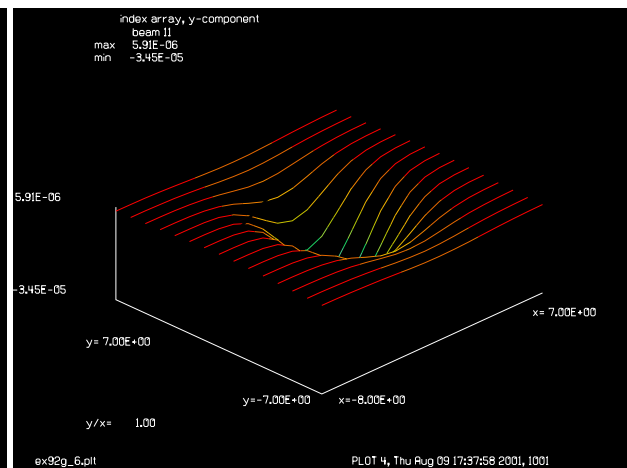


Fig. 92.24. Z-integrated N_y for ex92g.

c Example 92g: Three-dimensional heat flow with stress birefringence

c

c This example illustrates three-dimensional heat flow. Nine thermal
c arrays are used. A point source of heat is injected into array 5
c and heat flows outward in the transverse and axial directions.

c

c Beams Description

c -----

c 1-9 thermal material arrays

c 11 birefringence array (nx*L,transmission), (ny*L,nxy*L)

c 12 optical

variab/dec/int row plot Nline

Nline = 16

Nc = Nline/2 + 1

array/s 1 Nline

nbeam 9

nbeam 10 Nline 16 # x-z scan

Jump to: [Commands](#), [Theory](#)

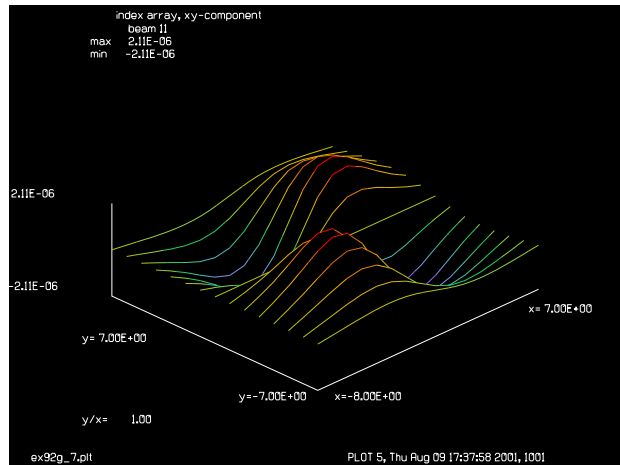


Fig. 92.25. Z-integrated Ny for ex92g.

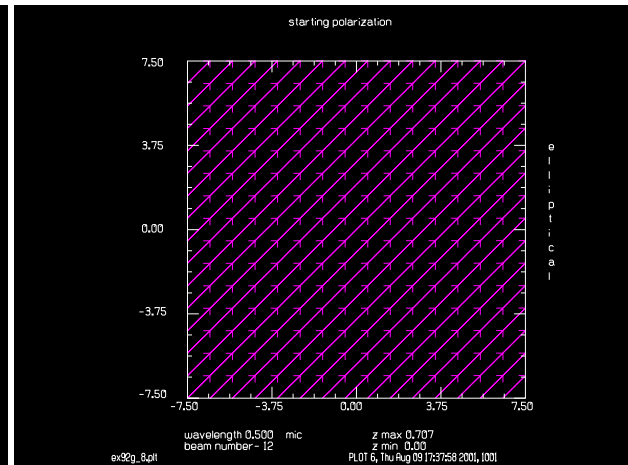


Fig. 92.26. Starting polarization (before birefringent material).

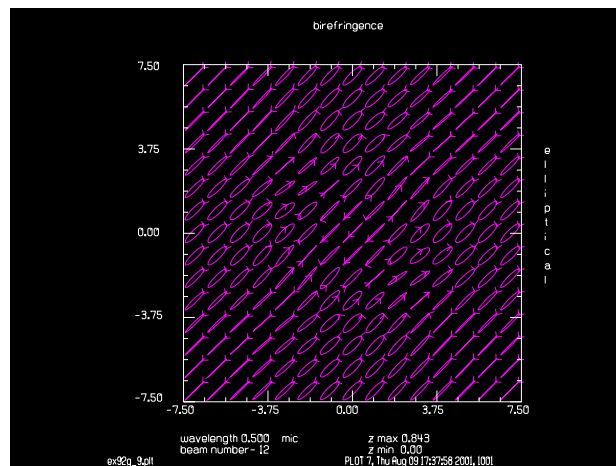


Fig. 92.27. Polarization after applying birefringent array. Note polarization effects in the center.

```

units/s 0 1
units/s 10 1 1
#
# Construct gaussian distribution with 1 degree peak
# above baseline of 1 degree temperature base
#
gaus/c/c 5 1 4
c# clear 5 0
c# point/set 5 Nc Nc 1
nbeam 11 Nline Nline
add/inc/c 5 11
nbeam 10
thermal/mat/add 1 1 1 BK7
thermal/mat/add 2 2 1 BK7
thermal/mat/add 3 3 1 BK7
thermal/mat/add 4 4 1 BK7
thermal/mat/add 5 5 1 BK7
thermal/mat/add 6 6 1 BK7

```

Jump to: [Commands](#), [Theory](#)

```

thermal/mat/add 7 7 1 BK7
thermal/mat/add 8 8 1 BK7
thermal/mat/add 9 9 1 BK7
mac/def ex92g/o
    therm/set/group time 1 1 2 3 4 5 6 7 8 9
    total_time = total_time + time
    row = 0
    mac step/9
mac/end
macro/def step/o
c   copy center row of all thermal arrays into beam 10
    row = row + 1
    copy/row row 10 9 row+4
macro/end
macro/def plot/o
    title transverse distribution of center, time = @total_time
    plot = plot + 1
    plot/watch ex92g_@plot.plt
    plot/l/r 5 min=0
    title longitudinal distribution of heat, time = @total_time
    plot = plot + 1
    plot/watch ex92g_@plot.plt
    plot/l/r 10 min=0
macro/end
time = 10
total_time = 0
plot = 0
c
c   Allows 40 seconds to pass
c
macro ex92g/4
row = 0
mac step/9
mac plot
nbeam 11 Nline ipol=1
youngalpha = 1e7*2e-4
index = 1.5
c1 = 1e-8
c2 = .5e-8
c
c   Order of magnitude of effect is
c
c    $.5 * \text{Young} * \alpha * C \cdot dT * L / \lambda / \text{index}^3$  (waves)
c    $.5 * e8 \cdot e-5 \cdot e-8 * 1 * 10 / .5e-4 / 1.5^3 = .2963$  (waves)
c
therm/indexarray/isotropic 11 youngalpha index c1 c2 1 2 3 4 5 6 7 8 9
c
c   Array 11 now contains (nx,transmission), (ny,nxy)
c
plot/w ex92g_5.plt
title index array, x-component
plot/l/r 11
plot/w ex92g_6.plt
title index array, y-component

```

Jump to: [Commands](#), [Theory](#)

```

plot/l/pr 11
plot/w ex92g_7.plt
title index array, xy-component
plot/l/pa 11
nbeam 12 Nline ipol=1
Lambda = .5e-4
wavel/set 12 Lambda*1e4
jones/set cr=1
c jones/set cr=-1
jones/mult 12
peak/norm 12 1
plot/w ex92g_8.plt
title starting polarization
plot/e 12
c
c apply index array, (nx*L,transmission),(ny*L,nxy*L)
c
therm/birefringence 12 11
plot/w ex92g_9.plt
title birefringence
plot/e 12
plot/w ex92g_10.plt
title wavefront
plot/l/w 12

```

Ex92h: Simple model of thermal array and optical aberrations

Example illustrating how thermal distortion of an internal window would affect a laser. The end pumping is assumed to be a gaussian beam and its bulk absorption is treated as an internal thermal source.

Input: ex92h.inp

```

c## ex92h
c
c Example 92h: Simple model of thermal array and optical aberrations
c
c Beams      Description
c -----
c 1          optical array
c 2          thermal array

alias name ex92h
array/s 1 128
nbeam 3 data
units 0 .004
wavelength 0 1.064

                                # units of all beams .04mm
                                # NdYAG wavelength

gaus/c/c 2 30 .8
                                # define initial 30 temp in BK7
                                # with gaussian roll off
clap/sqr/c 2 .15
                                # square section of glass, .25 thick
thermal/mat/add 2 2 thick=.25 BK7

```

Jump to: [Commands](#), [Theory](#)


```

clear 3 15                                # set cooling water to 15 deg
thermal/mat/add 2 3 forced_water          # add 15 degree coolant water
plot/w @name_1.plt
title Show material distribution
plot/l/mat 2                              # make a plot just to show material distributions
plot/w @name_2.plt
title Show initial temperature relative to coolant
plot/l/r 2
c
c Calculate initial thermal aberrations
c
clear 1 1                                # start with plane wave
c calculate thermal aberrations
thermal/win kbeam=1 time=0. kb1=2
strehl 1
plot/l/real 2
plot/watch @name_3.plt
title Optical aberrations at starting conditions
plot/l/w 1 min=-.4 max=.4
time = 5                                # time interval
total_time = 300                        # total time
therm/settle 2 time total_time          # settling time = total_time
c
c Calculate thermal aberrations again
c
clear 1 1                                # start with plane wave again
thermal/win kbeam=1 time=0. kb1=2
strehl 1
plot/w @name_4.plt
title Temperature distribution at time = @total_time
plot/l/real 2
plot/w @name_5.plt
title Optical aberrations at time= @total_time
plot/l/w 1 min=-.4 max=.4

```

Ex92i: Thermal distortion due to end pumping

Example illustrating how thermal distortion of an internal window would affect a laser. The end pumping is assumed to be a gaussian beam and its bulk absorption is treated as an internal thermal source.

Input: ex92i.inp

```

c## ex92i
c
c Examples 92i: Two-dimensional heat flow, window, air contact,
c              internal heat source
c
c
c inner 4.4 cm diameter is BK7 glass
c a .4 cm thickness annular ring of aluminum outside the glass
c surrounding the aluminum is forced air
c

```

Jump to: [Commands](#), [Theory](#)

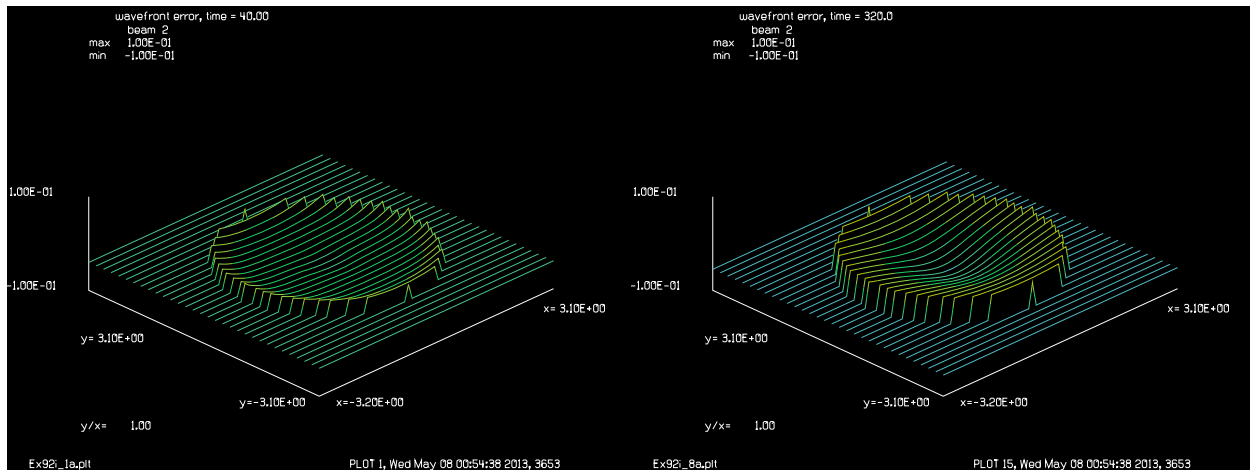


Fig. 92.28. Wavefront distortion after 40 seconds of end pumping for Ex92i.

Fig. 92.29. Wavefront distortion after 320 second of end pumping. Distribution is approaching steady state for Ex92i

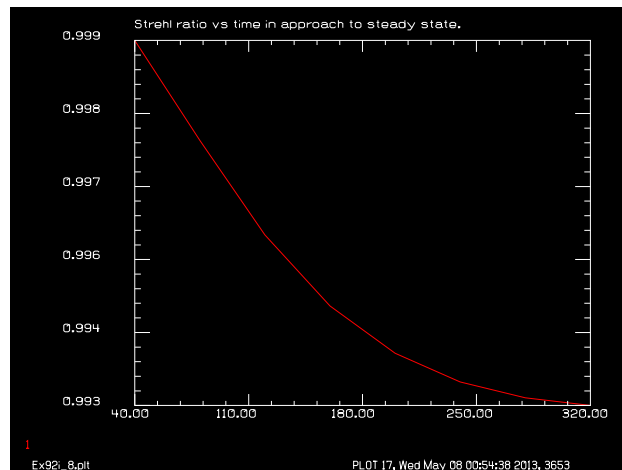


Fig. 92.30. Strehl Ratio over 320 seconds of end pumping. The Strehl Ratio is approaching a steady state condition as the thermal distortion stabilizes.

```

c the thermal constants between matrix points are
c glass          2.0      seconds
c aluminum       0.0043  seconds
c forced air     12.2     seconds
c
alias Name Ex92i
variab/dec/int plot ntimes
nbeam 5 data                                # establish beams
c
c Beams
c 1                                           # optical beam
c 2                                           # phase screen
c 3                                           # thermal array
c 4                                           # pump distribution
c 5                                           # temporary array
c

```

Jump to: [Commands](#), [Theory](#)

```

units/s 0 .1                # units are 0.1 cm
c
c  set up material 1 for thermal surface as glass, BK7
c
c  set up initial temperature
c
clear 3 30                   # set beam 3 to 30 deg.
clap/c/c 3 2.2               # circular aperture of 2.2 cm
thermal/material/add 3 3 1. BK7  # establish a thermal array for beam 3
c
c  Set up pump distribution as internal source
c  in array 4
c
gaus/c/c 4 .1 1
thermal/mat/source/square 3 4
clear 5 25
thermal/mat/add 3 5 1. forced_air
thermal/mat/list
total_time = 0
plot = 0
echo/on
macro/def StartBeam
  plot = plot + 1
  c
  c  Set up optical beam as gaussian.
  c  This could be a beam generated in a resonator.
  c
  gaus/c/c 1 1 1.1
  time = 40                # time advances 40 seconds
  mac therm                 # compute cumulative thermal effects
  plot/w @Name_@plota.plt
  title wavefront error, time = @total_time
  plot/l/w 2 min=-.1 max=.1
  mult/beam 1 2             # apply phase plate
  plot/w @Name_@plotb.plt
  title Optical beam: intensity and phase, time = @total_time
  plot/x 1
  variab/set Strehl 1 strehl
  udata/set plot total_time Strehl
mac/end
macro/def therm
  therm/set 3 time nstep=0 # use automatic calculation of steps
  total_time = total_time + time
  c
  c  Make phase plate containing aberrations
  c
  clear 2 1
  therm/window 2 0. 3
mac/end
macro/run StartBeam/8
udata/list
plot/w @Name_@plot.plt
title Strehl ratio vs time in approach to steady state.
plot/udata

```

Jump to: [Commands](#), [Theory](#)

Ex93: Design of far-field distribution by phase-retrieval methods

Table. 93.1. Table of Ex93 examples

Ex93a: Design of far-field distribution by phase-retrieval methods	7
Ex93b: Recovery of pupil aberrations from spherically aberrated image	11

Phase-retrieval methods provide a method finding a phase distribution which produces a specified far-field irradiance. Phase-retrieval methods have been used successfully to solve the inverse problem of finding the pupil phase which produces a known far-field irradiance. This technique was used to determine the wavefront error of the Hubble Space Telescope from the far-field image of stars.

Phase-retrieval may methods may be used to find a wavefront which generates an arbitrary far-field distribution. In this example, the desired far-field shape is a supergaussian. The near-field irradiance distribution is a flat-top function. The objective of the phase-retrieval method is to find the near-field phase distribution which best generates the far-field supergaussian. The common phase-retrieval methods are based on the Gerchberg-Saxton theory. In its basic form Gerchberg-Saxton consists of starting with a noise distribution and propagating back and forth between the near-field and far-field and normalizing the irradiance distribution to the desired form at each plane, while leaving the phase unchanged. See Fig. 93.1.

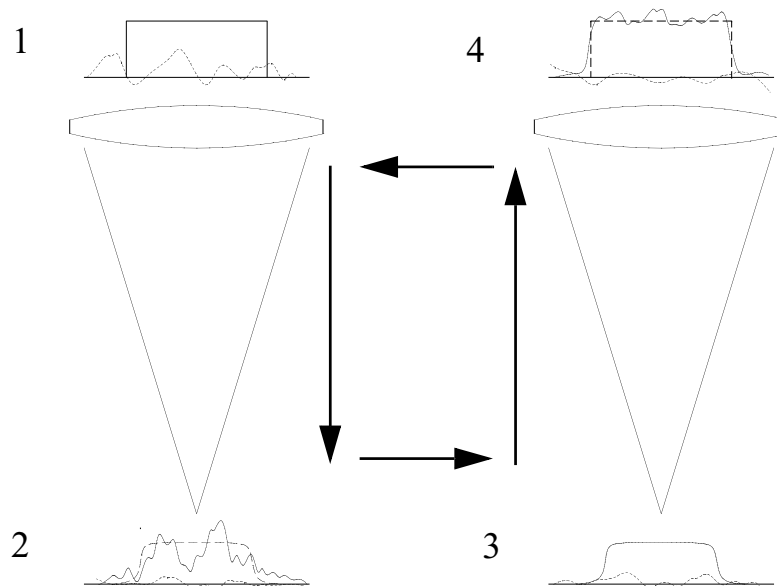


Fig. 93.1. The goal of phase retrieval in this application is to find the phase at a flat-top pupil intensity distribution (solid line) in the near-field (Point 1) to create a supergaussian distribution in the far-field (Point 2). We assume random phase (dotted line) at Point 1 to start and compute the intensity in the far-field at Point 2. We then normalize the far-field intensity to the desired shape—supergaussian in this case—leaving the phase unchanged. This is indicated at Point 3. The beam is then back-propagated to the near-field at Point 4. The intensity does not perfectly match the initial flat-top distribution. We normalize the near-field to the flat-top envelope, as indicated in Step 1, leaving the phase unchanged, and repeat the process to convergence. At convergence, the phase at Point 1 is the required answer to create the desired far-field irradiance.

The Gerchberg-Saxton method creates a speckle distribution whose envelope matches the desired distribution well. The speckled nature of the solution seems to be an inevitable result of the method.

Similarly the near-field phase has a speckled form and there are intrinsic phase discontinuities. Well behaved wavefronts may range over many wavelengths. In order to display the wavefront it is necessary to determine the phase by taking the arctangent of the complex amplitude representation. The arctangent function determines phase in the range of $-\pi$ to π . GLAD invokes an algorithm to remove the 2π discontinuities, which arise in continuous wavefronts with a large range of aberration.

In phase-retrieval, there are the usual 2π discontinuities, which may be removed; but there are also discontinuities which are fundamental. The fundamental discontinuities are similar to poles in the complex plain. These poles are readily seen using `plot/vector` which shows complex amplitude in the real-imaginary plane. Poles are generally located at intensity minima, but these minima need not be exactly zero intensity. If we observe `plot/vector` displays, the poles can be identified abrupt changes in the complex amplitude. If we try to resolve the 2π jumps by observing the complex amplitude along a continuous path, we get different answers depending on which side the path passes the pole. The wavefront jump can be 2π or -2π depending on whether the pole is clockwise or counterclockwise and the side of the pole taken by the path. Fig. 93.13 illustrates a pole pair with a branch cut which is a tear of the wavefront. [Fig. 93.14 shows the loci of real zeros and imaginary zeros, where the loci cross the intensity is precisely zero and a phase pole is formed. Fig. 93.15. shows a kinoform surface formed by this complex amplitude distribution with 2 jumps. Fig. 93.16 shows the result of a simple phase unwrapping algorithm to remove phase discontinuities. The branch cut causes a ramp of raised phase. The length of the phase ramp determines the amount of scattering. If the algorithm recognizes branch cuts (as shown in Fig. 93.17), the length of the discontinuity may be minimized to just the length of the branch cut—greatly shortening the length of the discontinuity and reducing scattering.

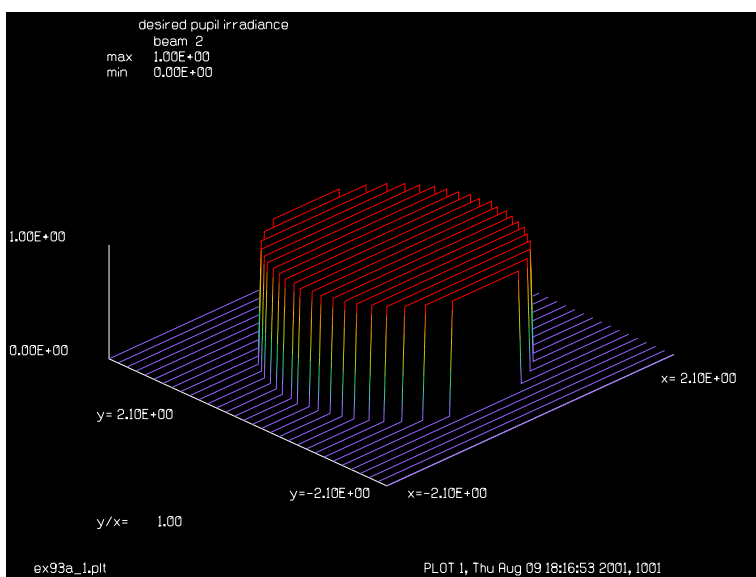


Fig. 93.2. Flat-top pupil distribution. The near-field is renormalized to this shape at the start of each pass.

References

1. W. Gerchberg and W. O. Saxton, "A practical algorithm for the determination of phase from image and diffraction plane pictures," *Optik* 35, 237-246 (1972).

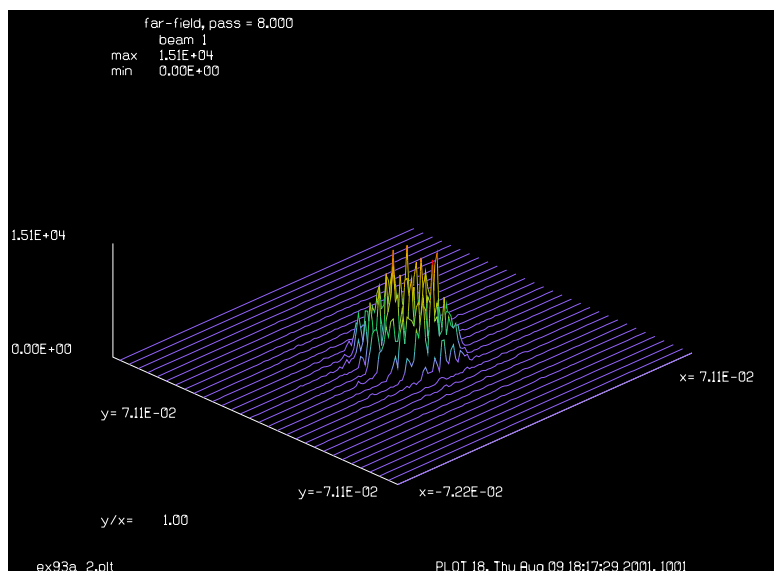


Fig. 93.3. Desired far-field distribution. The far-field is normalized to this shape during each pass.

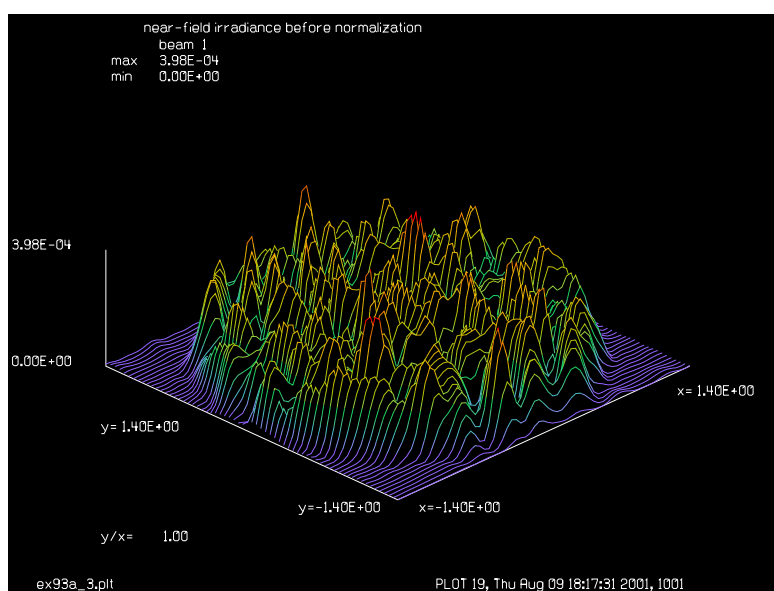


Fig. 93.4. Near-field irradiance after return from far-field and before renormalization to the flat-top function shown in Fig. 93.1. Note, speckled structure.

2. O. Saxton, Computer Techniques for Image Processing in Electron Microscopy (Academic, New York, 1978).
3. J. R. Feinup, "Phase-retrieval algorithms: a comparison," Appl. Opt., Vol. 21, No. 15, 2758-2769 (1982).
4. B. Ya Zel'dovich, N. F. Pilipetsky, and V. V. Shkunov, "Principles of Phase Conjugation," pp 79-84, Springer-Verlag, New York (1985).

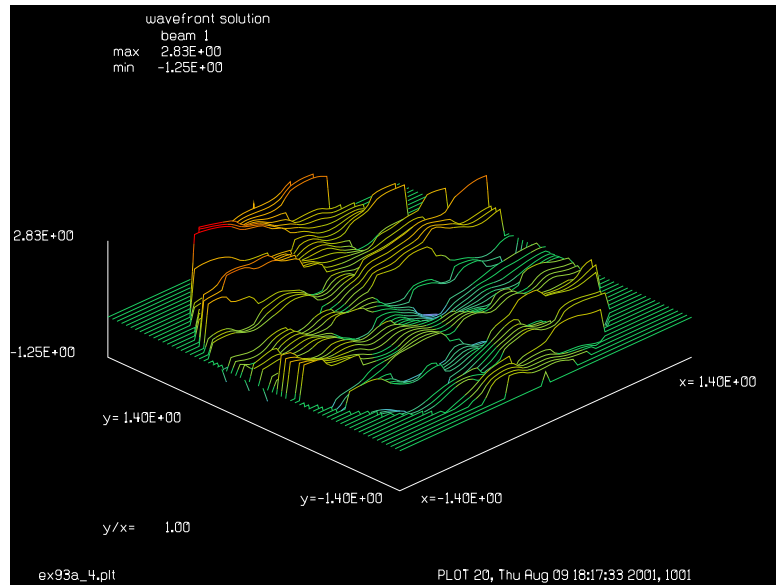


Fig. 93.5. Near-field irradiance after return from far-field and before renormalization to the flat-top function shown in Fig. 93.1. Note, speckled structure.

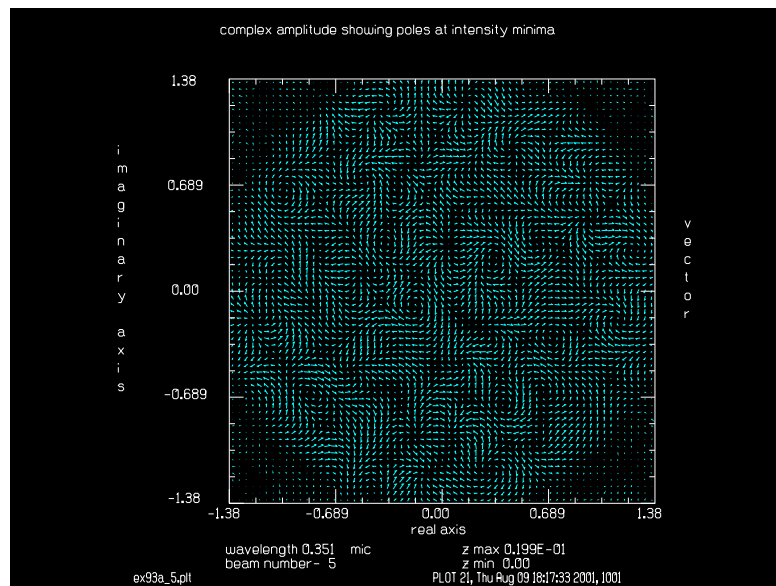


Fig. 93.6. Plot of complex amplitude in real-imaginary plane. Note the regions of constant amplitude—corresponding to speckles—and discontinuities.

5. Richard Barakat and Barbara H. Sandler, "Determination of the wave-front aberration function from measured values of the point-spread function: a two-dimensional phase-retrieval problem," JOSA A, Vol. 9, No. 10, 1715-1723 (1992).
6. J. R. Fienup, "Phase-retrieval algorithms for a complicated optical system," Appl. Opt., Vol. 32, No. 10, 1737-1745 (1993).
7. J. R. Fienup, J. C. Marron, T. J. Schulz, and J. H. Seldin, "Hubble Space Telescope characterized by using phase-retrieval algorithms," Appl. Opt., Vol. 32, No. 10, 1747-1767 (1993).

Jump to: [Commands](#), [Theory](#)

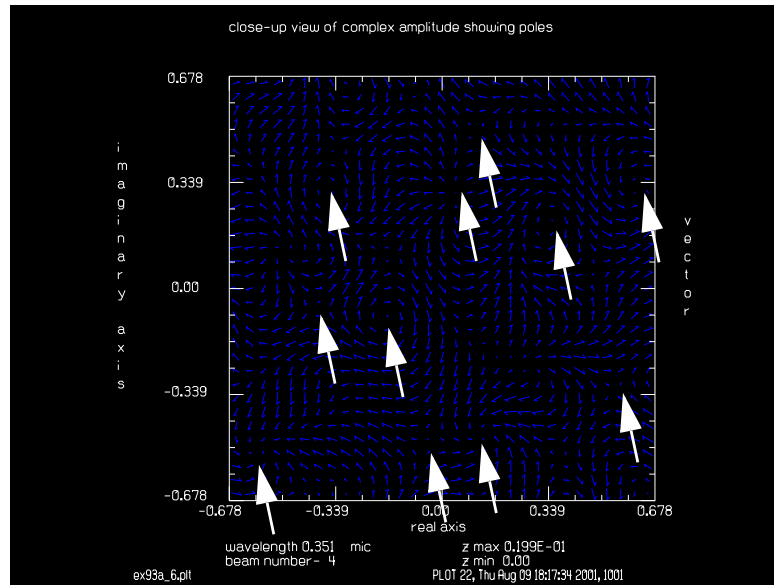


Fig. 93.7. Blow-up of complex amplitude, showing the inner one half of the near-field distribution. Note the poles (some of which are indicated by arrows). These poles cause irreducible phase jumps.

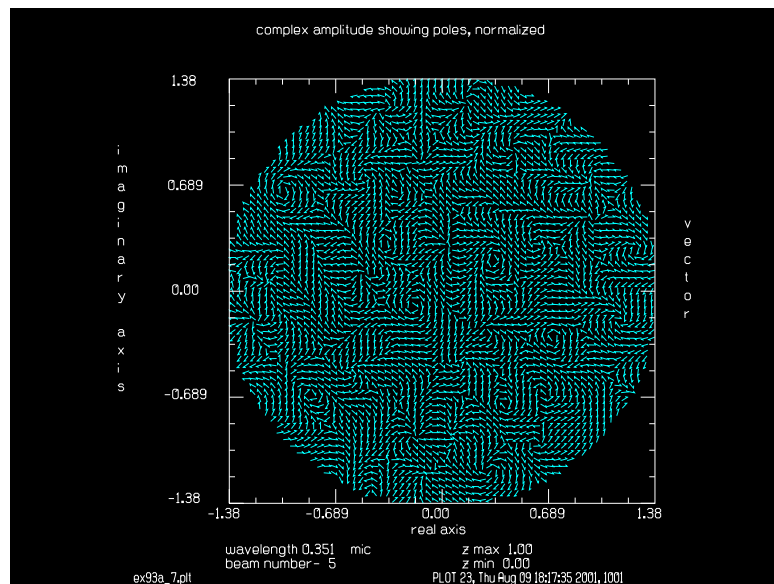


Fig. 93.8. Plot of complex amplitude in real-imaginary plane after normalization. Note the regions of constant amplitude—corresponding to speckles—and discontinuities.

8. Mohammad H. Maleki and Anthony J. Devaney, "Phase-retrieval and intensity-only reconstruction algorithms for optical diffraction tomography," JOSA A, Vol. 10, No. 5, 1086-1092 (1993).
9. M. S. Scivier and M. A. Fiddy, "Phase ambiguities and the zeros of multidimensional band-limited functions," JOSA A, Vol. 2, No. 5, 693-697 (1985).
10. J. R. Fienup, "Iterative method applied to image reconstruction and to computer-generated holograms," Opt. Eng. 19 297-306 (1980).

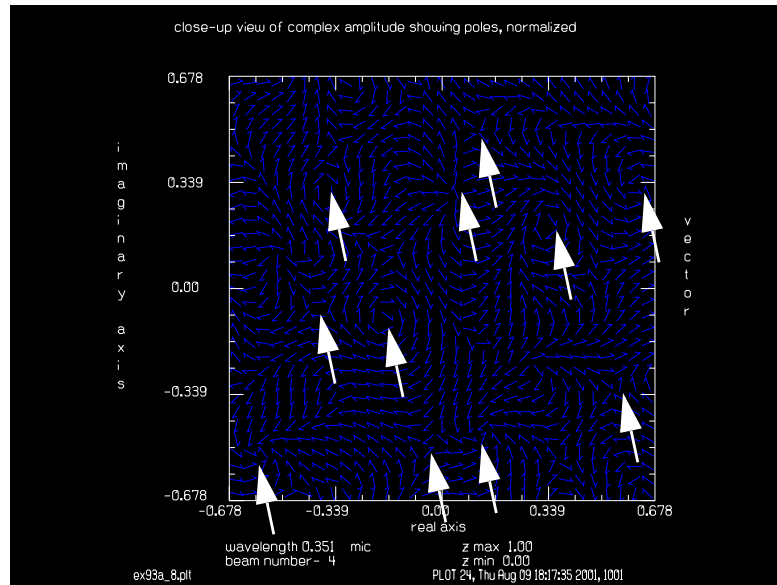


Fig. 93.9. Blow-up of complex amplitude after normalization, similar to Fig. 93.6. Note the poles (some of which are indicated by arrows). These poles cause irreducible phase jumps.

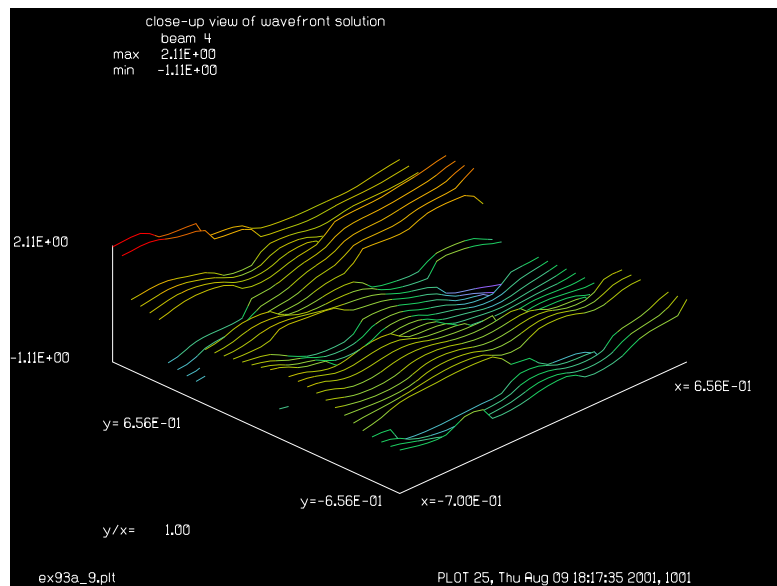


Fig. 93.10. Close-up of phase with the same scale as Figs. 93.6 and 93.8, showing phase jumps which have not been removed by the GLAD algorithm. These phase jumps are at the site of poles.

11. F. Wyrowski, "Iterative Fourier-transform algorithm applied to computer holography," J. Opt. Soc. Am. A 5, 1058-1065 (1988).
12. Stephen D. Mellin and Gregory P. Nordin, "Limits of scalar diffraction theory and an iterative angular spectrum algorithm for finite aperture diffractive optical element design," Optics Express, Vol. 8, No. 13, 705-722 (2001).
13. Keith A. Nugent, David Paganin, and Tim E. Gureyev, "A Phase Odyssey," Physics Today, Vol. 54, No. 8, 27-32, (August, 2001).

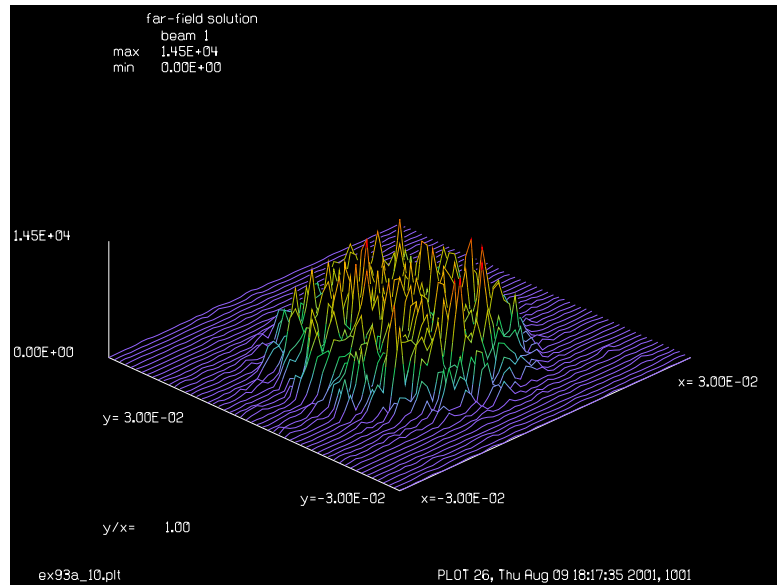


Fig. 93.11. Far-field speckle pattern. Compare with desired shape shown in Fig. 93.2.

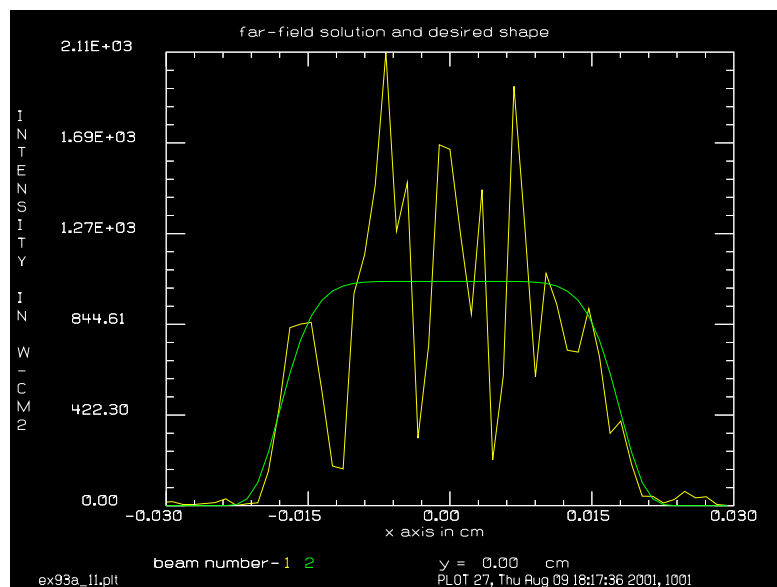


Fig. 93.12. Comparison of far-field solution, showing speckles, and the desired supergaussian mode shape.

Ex93a: Design of far-field distribution by phase-retrieval methods

Input: `ex93a.inp`

```
c## ex93a!42778589598529
c
c Example 93a: Design of far-field distribution by phase retrieval methods
c
c An iterative method called phase retrieval may be used to design the
c envelope of the irradiance in the far-field, using phase-only control
```

Jump to: [Commands](#), [Theory](#)

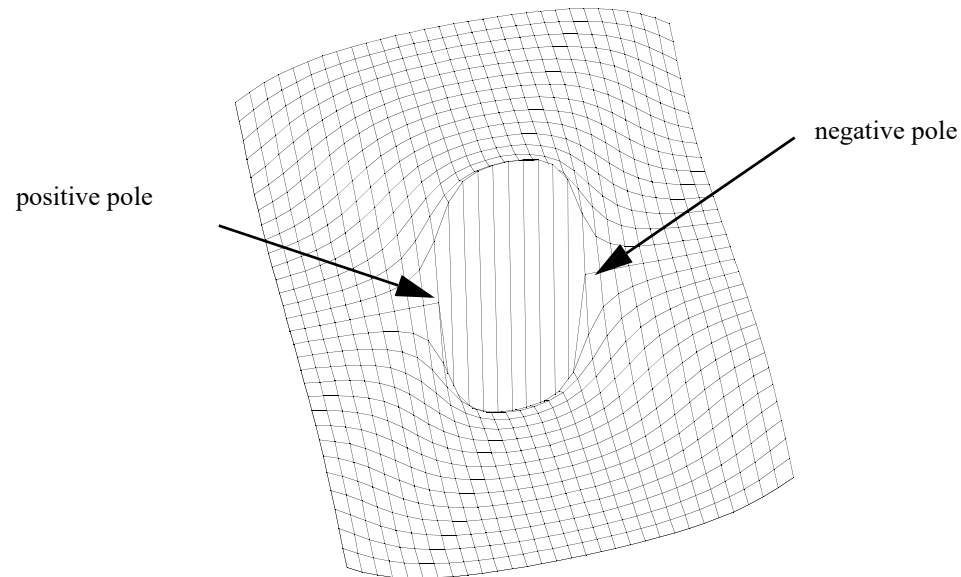


Fig. 93.13. A pair of positive and negative wavefront poles which form a branch cut. The wavefront is torn along the branch cut. Wavefront slopes approach infinite values near the poles. In the vicinity of a pole, the wavefront slope is approximately $\lambda/2\pi r$ where r is the distance from the pole.

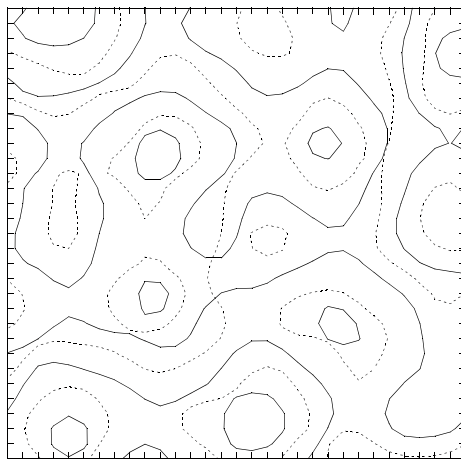


Fig. 93.14. Loci of zeros for real (solid) and imaginary function (dashed). Poles exist at loci crossings.

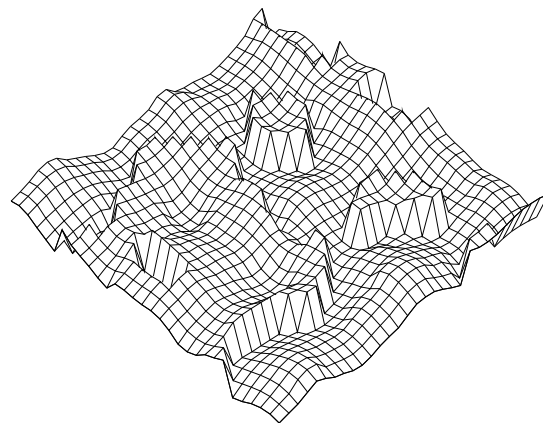


Fig. 93.15. Kinoform made from the wavefront. Long boundary lines cause large scattering.

c of the near-field.

c

c At both the near- and far-fields, the intensity is normalized to the
c desired shape and the phase is left unchanged. The resulting far-field
c distribution is speckled but its envelope closely matches the desired
c supergaussian shape.

c

c The phase, generated in this way, tends to have inherent discontinuities
c of phase. These manifest as poles in the complex amplitude distribution
c The poles may be clockwise or counter-clockwise. These
c discontinuities are fundamental and the GLAD two pi removal algorithm,
c used for phase plots, can not remove them.

Jump to: [Commands](#), [Theory](#)

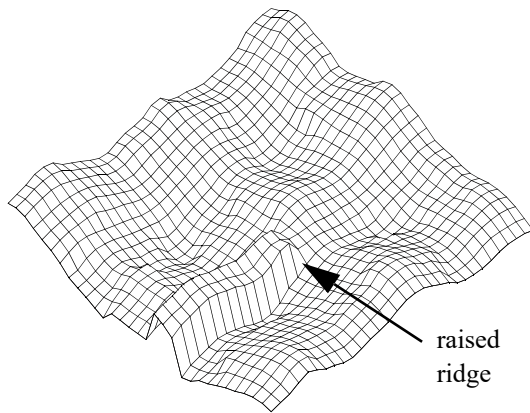


Fig. 93.16. Simple phase unwrapping leaves a ridge of raised wavefront because of the branch cut.

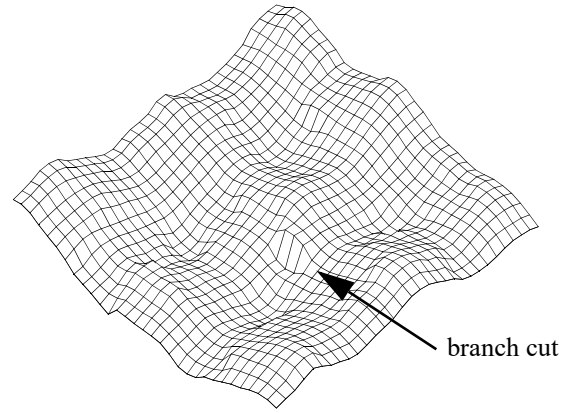


Fig. 93.17. A more sophisticated algorithm recognizes the branch cut and minimizes discontinuity length.

```

c
c  beam arrays used
c  -----
c  beam 1      propagating beam
c  beam 2      desired near-field shape
c  beam 3      desired far-field shape
c  beam 4      32 x 32 array to show phase jumps in inner region of pupil
c  beam 5      64 x 64 array to show phase jumps
c
variab/dec/int nline
nline = 128                      # array size
clap1 = 1.4                      # radius of flat top function for pupil
clap2 = .02                      # supergaussian radius for far-field
array/s 1 nline                  # set array size
nbeam 3 data
nbeam 4 32 data
nbeam 5 64 data
wavelength/set 0 .3511
units2 = clap1/32
units/s 1 units2
units/s 4 units2
units/s 5 units2
clap/c/c 1 clap1
copy 1 2
title desired pupil irradiance
plot/watch ex93a_1.plt
plot/l 2 xrad=1.5*clap1
units3 = .3511e-4*180/(nline*units2) # calculate far-field units
units/s 3 units3
gaus/c/c 3 1 clap2 4
title desired far-field irradiance
plot/watch ex93a_2.plt
plot/l 3 ns=64 xrad=1.5*clap2
clear 1 0
noise 1 1                      # start distribution with noise

```

```

lens 1 180                                # set up convergent beam
set/density 32 32
plot/tty/pause 2
plot/tty
macro/def phaser/o
    pass = pass + 1
    norm/two 1 2                          # normalize pupil to flat-top
    title near-field, pass = @pass
    plot/l/w 1
    dist 180 1                            # move to far-field
    title far-field, pass = @pass
    plot/l 1
    norm/two 1 3                          # normalize far-field to supergaussian
    dist -180 1                           # propagate back to pupil
macro/end
mac phaser/8
title near-field irradiance before normalization
plot/watch ex93a_3.plt
plot/l 1 ns=128 xrad=1.4
copy 1 3                                  # make a copy before normalization
norm/two 1 2                             # normalize pupil to flat-top
clap/c/c 1 clap1
plot/tty/pause 0
title wavefront solution
plot/watch ex93a_4.plt
plot/l/w 1 ns=128 xrad=1.4
title complex amplitude showing poles at intensity minima
set/density 64 64
plot/watch ex93a_5.plt
copy/con 3 5
plot/vec 5
copy/con 3 4
plot/watch ex93a_6.plt
title close-up view of complex amplitude showing poles
set/density 32 32
plot/vec 4
copy/con 1 5
title complex amplitude showing poles, normalized
set/density 64 64
plot/watch ex93a_7.plt
plot/vec 5
copy/con 1 4
plot/watch ex93a_8.plt
title close-up view of complex amplitude showing poles, normalized
set/density 32 32
plot/vec 4
plot/watch ex93a_9.plt
title close-up view of wavefront solution
plot/l/w 4
dist 180 1
title far-field solution
plot/watch ex93a_10.plt
plot/l 1 ns=64 xrad=1.5*clap2
energy/norm 1 1

```

Jump to: [Commands](#), [Theory](#)

```

units/s 2 units3
gaus/c/c 2 1 clap2 4
energy/norm 2 1
plot/watch ex93a_11.plt
title far-field solution and desired shape
plot/x/i fi=1 la=2 le=[-1.5*clap2] ri=[1.5*clap2]
end

```

Ex93b: Recovery of pupil aberrations from spherically aberrated image

A typical example of phase retrieval consists of trying to determine pupil aberrations from the far-field irradiance, as demonstrated in EX93B.INP. In this example, 0.25 waves of spherical aberration are added to a flat-top distribution. The far-field irradiance is saved to form the target irradiance. The phase retrieval process is then started with flat phase in the pupil. The logarithm of one minus the correlation is calculated and displayed by plot/udata.

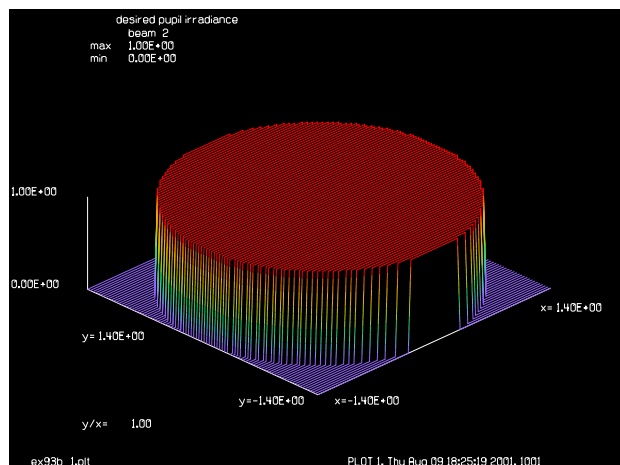


Fig. 93.18. Flat-top pupil irradiance.

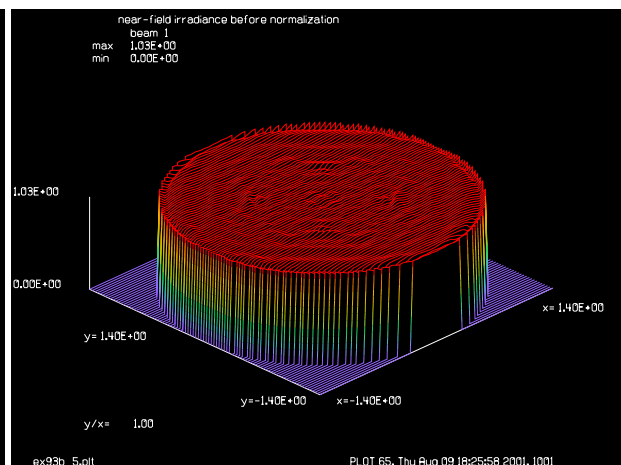


Fig. 93.19. Recreated pupil after 60 passes.

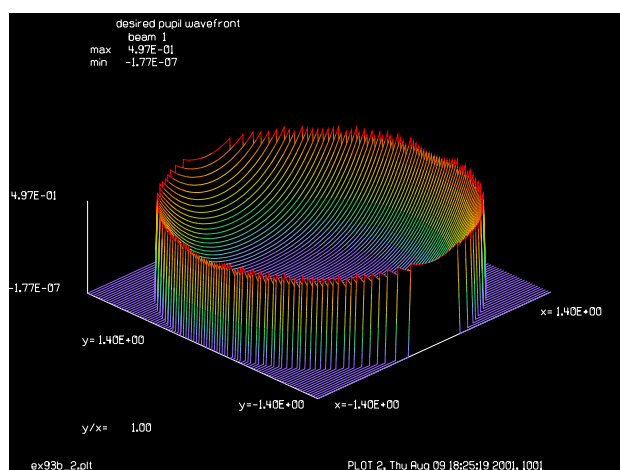


Fig. 93.20. Actual pupil phase is 0.5 waves of spherical aberration.

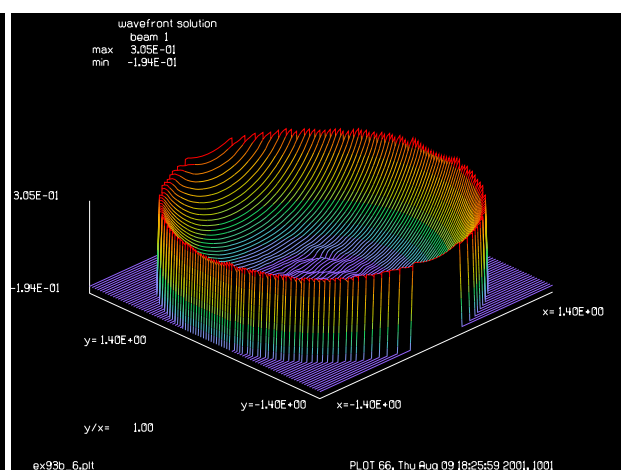


Fig. 93.21. Recreated pupil phase.

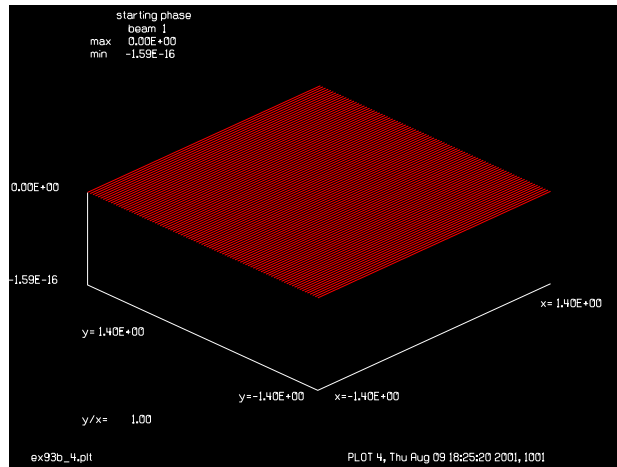


Fig. 93.22. Starting phase for phase retrieval is flat.

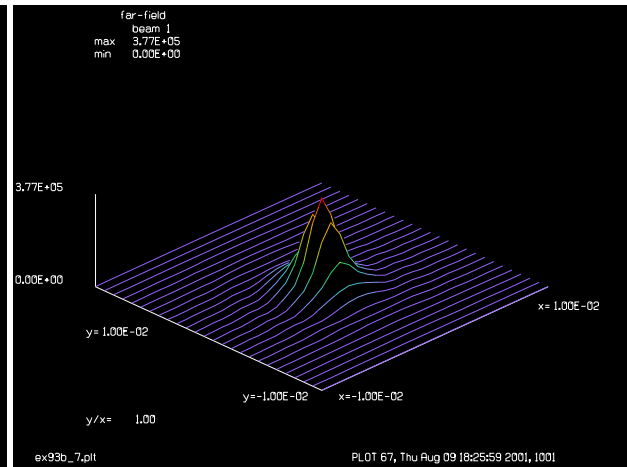


Fig. 93.23. Desired far-field irradiance.

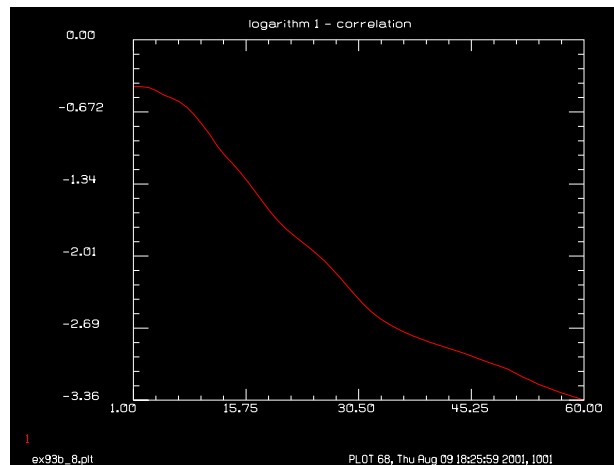


Fig. 93.24. Logarithm of (1 - correlation) for 60 steps of the phase retrieval algorithm. Correlation is 0.99956 after 60 steps. The algorithm is expected to continue to improve the fit for additional steps.

Input: `ex93b.inp`

```

c## ex93b
c
c Example 93b: Design of far-field distribution by phase retrieval methods
c
c An iterative method called phase retrieval may be used to design the
c envelope of the irradiance in the far-field, using phase-only control
c of the near-field.
c
c At both the near- and far-fields, the intensity is normalized to the
c desired shape and the phase is left unchanged. The resulting far-field
c distribution is speckled but its envelope closely matches the desired
c supergaussian shape.
c
c The phase, generated in this way, tends to have inherent discontinuities
c of phase. These manifest as poles in the complex amplitude distribution
c The poles may be clockwise or counter-clockwise. These

```

Jump to: [Commands](#), [Theory](#)


```

c discontinuities are fundamental and the GLAD two pi removal algorithm,
c used for phase plots, can not remove them.
c
c beam arrays used
c -----
c beam 1      propagating beam
c beam 2      desired near-field shape
c beam 3      desired far-field shape
c beam 4      32 x 32 array to show phase jumps in inner region of pupil
c beam 5      64 x 64 array to show phase jumps
c
mem/set/b 2
variab/dec/int nline
nline = 256                      # array size
clap1 = 1.4                      # radius of flat top function for pupil
array/s 1 nline                 # set array size
nbeam 4 data
wavelength/set 0 .3511
units2 = clap1/45
units/s 1 units2
units/s 4 units2
clap/c/c 1 clap1
copy 1 2
title desired pupil irradiance
plot/watch ex93b_1.plt
plot/l 2 ns=128 xrad=clap1
units3 = .3511e-4*180/(nline*units2) # calculate far-field units
lens 1 180                      # set up convergent beam
aberr = .5
abr/sph 1 aberr rn=clap1
copy 1 4
title desired pupil wavefront
plot/watch ex93b_2.plt
plot/l/w 1 ns=128 xrad=clap1
dist 180 1
copy 1 3
title desired far-field irradiance
plot/watch ex93b_3.plt
plot/l 3 ns=256 xrad=.01
dist -180 1
copy/con 2 1
title starting phase
plot/watch ex93b_4.plt
plot/l/w 1 ns=128 xrad=clap1
plot/watch plot1.plt
macro/def phaser/o
    pass = pass + 1
    norm/two 1 2                # normalize pupil to flat-top
    plot/l/w 1 xrad=clap1
    mult/mode/corr 1 4
    variab/set/para abscorr abscorr
    udata/set pass pass [log10(1-abscorr)]
    dist 180 1                  # move to far-field
    title far-field, pass = @pass

```

Jump to: [Commands](#), [Theory](#)

```
    norm/two 1 3                # normalize far-field to aberrated image
    dist -180 1                 # propagate back to pupil
macro/end
mac phaser/60
title near-field irradiance before normalization
plot/watch ex93b_5.plt
plot/l 1 ns=128 xrad=clap1
copy 1 3                       # make a copy before normalization
norm/two 1 2                   # normalize pupil to flat-top
clap/c/c 1 clap1
plot/screen/pause 0
title wavefront solution
plot/watch ex93b_6.plt
plot/l/w 1 ns=128 xrad=clap1
dist 180 1
plot/watch ex93b_7.plt
title far-field
plot/l 1 ns=128 xrad=.01
title logarithm 1 - correlation
plot/watch ex93b_8.plt
plot/udata max=0.
```

Ex94: Fiber optic laser

Table. 94.1. Table of Ex94 examples

Ex94a: Single centered core	1
Ex94b: Single decentered core	5
Ex94c: Four cores	9

This example illustrates a fiber laser, where the pump is injected into a multimode fiber and one or more higher index regions constitute single-mode core regions for lasing. Figure 94.1 shows the index profile. The multimode fiber is a circular region of index 1.5 (against a background of air at index 1.0) and diameter of 8 microns. Four core regions are a visible with a slightly higher index of 1.52. It is assumed that a plane wave pump is injected into the multimode core. There is doping in the laser cores that absorbs some of the pump light which passes through them. The absorption coefficient was set to 500 cm^{-1} . The absorbed energy causes pumping into N_2 , thus creating a population inversion. Four lasers develop from spontaneous emission in the laser cores and build up fairly quickly to saturation. When the lasers are not saturated, the transfer of pump light to laser light is inefficient and much of the N_2 excitation simply decays through fluorescence. When the laser is saturated it sweeps out almost all of the available excitation.

Ex94a: Single centered core

Input: `ex94a.inp`

```

c## ex94a
#
# Example: fiber laser, single centered core
#
# This example illustrates a laser consisting of a single core
# embedded in a larger multimode optical fiber
#
# The fiber consists of a cladding of index 1.5 and a core of index
# 1.52, giving a difference in index of 0.02. The modeling directly
# treats the phase build-up versus distance. Beam 2 contains the
# pattern of index change associated with the core. The INT2PHAS/TWO
# command is used to implement the incremental index difference defined
# on Beam 2 as a phase effect on Beam 1.
#
#
# Beam          purpose
# ----          -
# 1              propagating pump beam
# 2              refractive index distribution
# 3              history array for pump mode above center
# 4              history array for pump mode at center
# 5              history array for pump mode below center
# 6              loss function per step
# 7              deposited pump energy
# 8              propagating laser array
# 9              laser medium
# 10             history of laser, middle two cores

```

```

#
set/cpu 2 # valid only for dual CPU machines
mem/set/b 8 # request 3 megabytes memory
# declare names of integer variables
variab/dec/int pass pass1 count nliney count1 nskip
c set/label/off # take off labels of plots
Dist = 1.8e-5 # step length of propagation
nliney = 128 # propagating array width
nsteps = 2048
nskip = nsteps/64 # display 64 strips
nliney = nsteps/nskip # length of history arrays
shift = 1.5e-4 # decentering of cores
array/s 1 nliney # beam 1, propagating beam,128x128
nbeam 2 data # beam 2, index distribution
nbeam 5 nliney nliney data # beam 3, history, above center
# beam 4, history, middle
# beam 5, history, below center
nbeam 6 nliney nliney data # loss function per step
nbeam 7 nliney nliney data # array for deposited pump
nbeam 8 nliney nliney beam # laser array
nbeam 9 nliney nliney 1 data # medium array
nbeam 10 nliney nliney data # beam 10, history of laser

mem/cont
units/field 0 8e-4 # set same units 1.25e-5 for all 5 beams
variab/set units 3 units # make variable name units=units of beam 3
units/set 3 units Dist*nskip # reset units of history arrays (beam 3-5)
units/set 4 units Dist*nskip
units/set 5 units Dist*nskip
units/set 10 units Dist*nskip
lambda_l = .6328e-4 # laser wavelength
lambda_p = .53e-4 # laser wavelength
wavelength/set 0 lambda_l*1e4 1.5 # set wavelength and cladding index
wavelength/set 7 lambda_p*1e4 1.5
wavelength/set 8 lambda_p*1e4 1.5
Clap = .6e-4 # radius of a individual core
Index = .02 # index difference
clear 1 Index # intensity of beam 1 = Index
clear 2 Index # intensity of beam 2 = Index
clap/c/c 2 Clap # make core
clear 1 .5 # reinitialize beam 1 and beams 3-5
clap/c/c 1 4e-4 # radius of cladding is 5e-4
add/inc/con 2 1 # add cladding index
clear 1 1 # vacuum index
add/inc/con 2 1
clear 3 0
clear 4 0
clear 5 0
obs/c 6 Clap # specify pump absorption of doping
loss_coef = 500 # cm-1
loss_per_step = loss_coef*Dist
peak/norm 6 loss_per_step
clear 1 [1-loss_per_step]
add/inc/con 6 1

```

Jump to: [Commands](#), [Theory](#)

```

alpha1 = 2.*pi*Dist/lambda_p      # phase coefficient per step = k*n*Dist
alpha2 = 2.*pi*Dist/lambda_l      # phase coefficient per step = k*n*Dist

width = 1.45e13                    # Set line width, hz
center = 4.739e14                  # Set line center, hz
tspont = 3e-9                     # Set spontaneous emission rate, sec
t20 = 3e-1                         # Set decay rate for level 2, sec
t10 = 1e-11                       # Set decay rate for level 1, sec
mode_sep = 5e11                   # Set longitudinal mode separation, hz
                                   # (not used)
gaincv = 2./pi/width               # peak of gain response curve
plank = 6.626e-34
hnu = plank*center
index = 1.5
Beinstein = (lambda_l/index)^2/8./pi/tspont*gaincv list
ratexy = 1e7*loss_coef*lambda_p/lambda_l list
xr2 = ratexy/hnu list
t2 = 1./(1./t20+1./tspont) list
dpopss = xr2*t2 list
gain0 = Beinstein*dpopss list
Dist=
amplification = exp(gain0*Dist) list
xisat = hnu/Beinstein/t2 list

gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
pack/set 8 9                      # set beams to be packed
set/grid 4 2 4 2
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0                   # reinitialize propagating distance
    zreff/se 8 0                   # reinitialize propagating distance
    geodata/set/waist 1 waistx=1e-4 waisty=1e-4
    geodata/set/waist 8 waistx=1e-4 waisty=1e-4
c
c take two steps together
c
    int2phas/two 1 2 alpha1        # implement index in beam 2 on beam 1
    int2phas/two 8 2 alpha2        # implement index in beam 2 on beam 1
    split/beam 1 6 7
    variab/set Eng2 1 energy
    mult 7 [1./Dist]
    gain/rate/n2pump 7 9 .6328     # update pump rate
    pack/in                        # pack beams
    gain/rate/steady Dist 1        # implement rate eq. gain
    pack/out                       # unpack beams
    variab/set Eng4 8 energy
    dist Dist 1 8
    clap/s/n 1 7e-4               # rectangular absorbing boundary
    clap/s/n 8 7e-4              # rectangular absorbing boundary
    if count = nskip then
        pass1 = pass1 + 1

```

Jump to: [Commands](#), [Theory](#)

```

copy/row 1 3 [nlinex/2+1-12] pass1 # copy row 53 of beam 1 to
                                     # beam 3, above center
copy/row 1 4 [nlinex/2+1] pass1    # copy central row of beam 1 to
                                     # beam 4, middle
copy/row 1 5 [nlinex/2+1+12] pass1 # copy row 77 of beam 1 to
                                     # beam 5, below center
copy/row 8 10 [nlinex/2+1] pass1   # copy central row of beam 1 to
                                     # beam 4

title laser mode
plot/watch ex94a_3.plt
plot/l 8 ns=64 h=.2

title history of pump, middle cores
plot/watch ex94a_5.plt
plot/l 4 ns=64

title history of laser, middle cores
plot/watch ex94a_6.plt
plot/l 10 ns=64

count1 = count1 + 1
udata/set count1 pass Eng2 Eng4

title pump and laser energy
plot/watch ex94a_8.plt
plot/udata/set y01 y02
plot/udata/seq dash
count = 0
endif
macro/end
clear 1 1e7                                     # pump beam starts at 1e7
clap/c/c 1 4e-4                                # limit pump to large core
clear 8 0.                                     # zero laser array
clear 9 0.                                     # zerp medium array

pass = 0
macro step/nsteps                               # propagate nliney times

title index difference
plot/watch ex94a_1.plt
plot/l 2 ns=64 h=.4

plot/watch ex94a_2.plt
title final mode shape, pump
plot/l 1 ns=64 h=.2

plot/watch ex94a_3.plt
title final mode shape, laser
plot/l 8 ns=64 h=.2

plot/watch ex94a_4.plt
title history pump, above center
plot/l 3 ns=64

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch ex94a_5.plt
title history of pump, middle
plot/1 4 ns=64

plot/watch ex94a_6.plt
title history of laser, middle
plot/1 10 ns=64

plot/watch ex94a_7.plt
title history pump, below center
plot/1 5 ns=64

title power of pump (solid) and laser (dashed)
plot/watch ex94a_8.plt
plot/udata 1 2 dash

```

Ex94b: Single decentered core

Input: `ex94b.inp`

```

c## ex94b
#
# Example: fiber laser, single decentered core
#
# This example illustrates a laser consisting of a single decentered
# embedded in a larger multimode optical fiber
#
# The fiber consists of a cladding of index 1.5 and a core of index
# 1.52, giving a difference in index of 0.02. The modeling directly
# treats the phase build-up versus distance. Beam 2 contains the
# pattern of index change associated with the core. The INT2PHAS/TWO
# command is used to implement the incremental index difference defined
# on Beam 2 as a phase effect on Beam 1.
#
#
# Beam          purpose
# ----          -
# 1             propagating pump beam
# 2             refractive index distribution
# 3             history array for pump mode above center
# 4             history array for pump mode at center
# 5             history array for pump mode below center
# 6             loss function per step
# 7             deposited pump energy
# 8             propagating laser array
# 9             laser medium
# 10            history of laser, middle two cores
#
set/cpu 2                # valid only for dual CPU machines
mem/set/b 8              # request 3 megabytes memory
# declare names of integer variables

```

Jump to: [Commands](#), [Theory](#)

```

variab/dec/int pass pass1 count nlinex nliney count1 nskip
c set/label/off # take off labels of plots
Dist = 1.8e-5 # step length of propagation
nlinex = 128 # propagating array width
nsteps = 2048
nskip = nsteps/64 # display 64 strips
nliney = nsteps/nskip # length of history arrays
shift = 1.5e-4 # decentering of cores
array/s 1 nlinex # beam 1, propagating beam,128x128
nbeam 2 data # beam 2, index distribution
nbeam 5 nlinex nliney data # beam 3, history, above center
# beam 4, history, middle
# beam 5, history, below center
nbeam 6 nlinex nliney data # loss function per step
nbeam 7 nlinex nliney data # array for deposited pump
nbeam 8 nlinex nliney beam # laser array
nbeam 9 nlinex nliney 1 data # medium array
nbeam 10 nlinex nliney data # beam 10, history of laser

mem/cont
units/field 0 8e-4 # set same units 1.25e-5 for all 5 beams
variab/set units 3 units # make variable name units=units of beam 3
units/set 3 units Dist*nskip # reset units of history arrays (beam 3-5)
units/set 4 units Dist*nskip
units/set 5 units Dist*nskip
units/set 10 units Dist*nskip
lambda_l = .6328e-4 # laser wavelength
lambda_p = .53e-4 # laser wavelength
wavelength/set 0 lambda_l*1e4 1.5 # set wavelength and cladding index
wavelength/set 7 lambda_p*1e4 1.5
wavelength/set 8 lambda_p*1e4 1.5
Clap = .6e-4 # radius of a individual core
Index = .02 # index difference
clear 1 Index # intensity of beam 1 = Index
clear 2 Index # intensity of beam 2 = Index
clap/c/c 2 Clap xdec=shift # make core shifted to the right
clear 1 .5 # reinitialize beam 1 and beams 3-5
clap/c/c 1 4e-4 # radius of cladding is 5e-4
add/inc/con 2 1 # add cladding index
clear 1 1 # vacuum index
add/inc/con 2 1
clear 3 0
clear 4 0
clear 5 0
obs/c 6 Clap xdec=shift # specify pump absorption of doping
loss_coef = 500 # cm-1
loss_per_step = loss_coef*Dist
peak/norm 6 loss_per_step
clear 1 [1-loss_per_step]
add/inc/con 6 1
alpha1 = 2.*pi*Dist/lambda_p # phase coefficient per step = k*n*Dist
alpha2 = 2.*pi*Dist/lambda_l # phase coefficient per step = k*n*Dist

width = 1.45e13 # Set line width, hz

```

Jump to: [Commands](#), [Theory](#)


```

center = 4.739e14          # Set line center, hz
tspont = 3e-9             # Set spontaneous emission rate, sec
t20 = 3e-1               # Set decay rate for level 2, sec
t10 = 1e-11              # Set decay rate for level 1, sec
mode_sep = 5e11           # Set longitudinal mode separation, hz
                           # (not used)
                           # peak of gain response curve

gaincv = 2./pi/width
plank = 6.626e-34
hnu = plank*center
index = 1.5
Beinstein = (lambda_1/index)^2/8./pi/tspont*gaincv list
ratexy = 1e7*loss_coef*lambda_p/lambda_1 list
xr2 = ratexy/hnu list
t2 = 1./(1./t20+1./tspont) list
dpopss = xr2*t2 list
gain0 = Beinstein*dpopss list
Dist=
amplification = exp(gain0*Dist) list
xisat = hnu/Beinstein/t2 list

gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
pack/set 8 9              # set beams to be packed
set/grid 4 2 4 2
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0           # reinitialize propagating distance
    zreff/se 8 0           # reinitialize propagating distance
    geodata/set 1 0 0 1e-4 1e-4 1 1 # control diffraction algorithms
    geodata/set 8 0 0 1e-4 1e-4 1 1 # control diffraction algorithms
c
c take two steps together
c
    int2phas/two 1 2 alpha1 # implement index in beam 2 on beam 1
    int2phas/two 8 2 alpha2 # implement index in beam 2 on beam 1
    split/beam 1 6 7
    variab/set Eng2 1 energy
    mult 7 [1./Dist]
    gain/rate/n2pump 7 9 .6328 # update pump rate
    pack/in                   # pack beams
    gain/rate/steady Dist 1    # implement rate eq. gain
    pack/out                  # unpack beams
    variab/set Eng4 8 energy
    dist Dist 1 8
    clap/s/c 1 7e-4           # rectangular absorbing boundary
    clap/s/c 8 7e-4           # rectangular absorbing boundary
    if count = nskip then
        pass1 = pass1 + 1
        copy/row 1 3 [nlinex/2+1-12] pass1 # copy row 53 of beam 1 to
                                           # beam 3, above center
        copy/row 1 4 [nlinex/2+1] pass1    # copy central row of beam 1 to
                                           # beam 4, middle

```

Jump to: [Commands](#), [Theory](#)

```

copy/row 1 5 [nlinex/2+1+12] pass1 # copy row 77 of beam 1 to
                                     # beam 5, below center
copy/row 8 10 [nlinex/2+1] pass1   # copy central row of beam 1 to
                                     # beam 4

title laser mode
plot/watch ex94b_3.plt
plot/1 8 ns=64 h=.2

title history of pump, middle cores
plot/watch ex94b_5.plt
plot/1 4 ns=64

title history of laser, middle cores
plot/watch ex94b_6.plt
plot/1 10 ns=64

count1 = count1 + 1
udata/set count1 pass Eng2 Eng4

title pump and laser energy
plot/watch ex94b_8.plt
plot/udata/set y01 y02
plot/udata/seq dash
count = 0
endif
macro/end
clear 1 1e7                                     # pump beam starts at 1e7
clap/c/c 1 4e-4                                 # limit pump to large core
clear 8 0.                                       # zero laser array
clear 9 0.                                       # zerp medium array

pass = 0
macro step/nsteps                               # propagate nliney times

title index difference
plot/watch ex94b_1.plt
plot/1 2 ns=64 h=.4

plot/watch ex94b_2.plt
title final mode shape, pump
plot/1 1 ns=64 h=.2

plot/watch ex94b_3.plt
title final mode shape, laser
plot/1 8 ns=64 h=.2

plot/watch ex94b_4.plt
title history pump, above center
plot/1 3 ns=64

plot/watch ex94b_5.plt
title history of pump, middle
plot/1 4 ns=64

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch ex94b_6.plt
title history of laser, middle
plot/l 10 ns=64

plot/watch ex94b_7.plt
title history pump, below center
plot/l 5 ns=64

title power of pump (solid) and laser (dashed)
plot/watch ex94b_8.plt
plot/udata 1 2 dash

```

Ex94c: Four cores

This example assumes the pump is at 0.53 microns and that lasing occurs at 0.6328 so the maximum energy conversion, based on wavelength ratio, is 84%. This is evident in a plot of energy in the pump and the combined energy of all lasers, as shown in Fig. 94.2. We see that the energy of the lasers is converging to about 84% of the pump energy.

The pump will decay because of the absorption due to doping, whether or not lasing takes place. Pump decay is not exactly exponential, because while some modes cross the fiber modes at relatively sharp angles, other modes are nearly parallel and essentially “hide” from the absorption regions at the fiber cores. Of course, all modes will be eventually absorbed but some will be relatively long-lived. Fig. 94.3 shows the history of the pump irradiance along a horizontal diagonal. We can see that the long-lived modes exist in the outer regions of the multimode core (outside the region where the laser fibers are) and in the center (inside the four fibers). Fig. 94.4 shows the pump mode at the arbitrary axial position where the calculation was stopped. Had the calculation been stopped a little earlier or later we might have seen different proportions of light in the center versus the outer zones.

The four cores are relatively strongly coupled because the core index was only 0.02 above the multimode index. The intensity is not perfectly correlated among the cores because of spontaneous emission and finite coupling, so we see slow variations in relative height of the cores. This is particularly easy to see in Fig. 94.5 which is a history of the irradiance along a horizontal diagonal slice through the two middle cores. The slow variations of power between the laser cores may be due to weak coupling between the four cores, similar to the effects shown in Example 86. The laser irradiance at the end of the calculation is shown in Fig. 94.6.

We also see relatively rapid fluctuations in the power in each laser core due to beating of the laser modes. Although high angle laser light (from spontaneous emission and imprinting of high spatial frequency pump modulation) is scattered out of the laser cores, it is nevertheless trapped in the multimode core because of its high index difference of 0.5. Unlike a conventional laser which scrapes off high angle light at the apertures, there is no mechanism in this device for deleting high angle light and the relatively rapid fluctuations in the laser modes versus distance appear due to this the continued presence of this high angle laser light.

The rate equations are [1]

$$\Delta N_2 = \left[R_2 - \frac{N_2}{t_2} - (N_2 - N_1)W_i(v) \right] \Delta t, \quad (94.1)$$

$$\Delta N_1 = \left[R_1 - \frac{N_1}{t_{10}} + \frac{N_2}{t_{spont}} + (N_2 - N_1)W_i(v) \right] \Delta t, \quad (94.2)$$

where

ΔN_1	change in population of lower level, atoms/cm ³ ,
ΔN_2	change in population of upper level, atoms/cm ³ ,
R_2	pump rate for upper level, excitations/sec/cm ³ ,
R_1	pump rate for lower level, excitations/sec/cm ³ ,
t_{spont}	spontaneous decay lifetime, sec,
t_{20}	decay time from upper level to ground, sec,
t_2	total decay time from upper level to ground, sec, $1/t_2 = 1/t_{20} + 1/t_{spont}$,
t_{10}	decay time from lower level to ground, sec,
$W_i(v)$	transition probability density, probability/sec/cm ³ ,
Δt	elapsed time.

The transition probability density is,

$$W_i(v) = \frac{\lambda^2 g(v)}{8\pi n^2 h v t_{spont}} I, \quad (94.3)$$

where

λ	wavelength,
v	normalized lineshape,
n	index of refraction,
h	Planck's constant,
v	frequency of the radiation,
I	irradiance of the radiation.

The transition probability of Eq. (94.3) may be written in terms of the Einstein B-coefficient:

$$W_i(v) = B(v) \frac{I}{h v}, \text{ where } B(v) = \frac{\lambda^2 f(v)}{8\pi n^2 h v t_{spont}} I. \quad (94.4)$$

The small signal amplification takes the form

$$I(z) = I(0) e^{B \Delta N z}. \quad (94.5)$$

Jump to: [Commands](#), [Theory](#)

Under steady-state conditions, the irradiance of the optical field is constant and Eqs. (94.1) and (94.2) lead to the steady-state solution for the population inversion:

$$\Delta N^0 = R_2 t_2 - \left(R_1 + \frac{t_2}{t_{spont}} \right) R_2 t_{10}. \quad (94.6)$$

The small signal gain coefficient is

$$g_0(\nu) = B(\nu) \Delta N^0. \quad (94.7)$$

The gain coefficient for homogeneous broadening and for arbitrary irradiance magnitude is,

$$g(\nu) = \frac{g_0(\nu)}{1 + \frac{I}{I_s}}, \quad (94.8)$$

where,

$$I_s = \frac{8\pi n^2 h\nu}{\left(\frac{t_2}{t_{spont}} \right) \lambda^2 g(\nu)} = \frac{h\nu}{B(\nu) t_2}. \quad (94.9)$$

In the case of strong saturation, Eq. (94.8) is well approximated by

$$\frac{dI}{dz} \approx g_0(\nu) I_s = (B \Delta N^0) \left(\frac{h\nu}{B(\nu) t_2} \right) = \frac{\Delta N^0 h\nu}{t_2}. \quad (94.10)$$

Where the pumping rate into the upper level R_2 dominates the process, Eqs. (94.6) and (94.10) give the saturated gain coefficient as

$$\frac{dI}{dz} = R_2 h\nu, \quad g(\nu) = R_2 h\nu_l, \quad (94.11)$$

showing, in the case of saturated steady-state gain, a linear growth of irradiance with distance based on the pumping flux density. The subscript l is now used to indicate the laser frequency.

In the case of the fiber laser with coaxial pumping, the pumping rate is determined by the rate of decay of the pump. At the vicinity of the laser cores the pump experiences Beer's law decay. Assuming the decay constant for the pump light P is α , the decay rate in the laser core regions is $P(z) = P(0)e^{-\alpha z}$. The pump light drives the pumping rate for the laser according to

$$R_2 = \frac{\alpha P}{h\nu_p} \quad (94.12)$$

Jump to: [Commands](#), [Theory](#)

$$\frac{dI(z)}{dz} = \alpha P \frac{v_l}{v_p} = \alpha P \frac{\lambda_p}{\lambda_l} \quad (94.13)$$

showing the linear rate of growth of the laser in the saturated regime for colinear pumping. Assuming the all of the pump light is absorbed by the doping in the laser cores, the maximum conversion pump power to laser power is λ_p/λ_l . This ideal efficiency will be approached when the laser small signal gain and saturation intensity are such that the laser moves quickly into saturated condition. Although the total laser power is necessarily less than the total pump power, the laser irradiance will be considerably higher according to the ratio of the laser core area to the area of the pump region consisting of the multimode region. The laser irradiance may, therefore, be two or three orders of magnitude higher than that of the pump and (for a single mode fiber) of nearly ideal beam quality.

In simplest form spontaneous emission may be injected for a distance Δz as

$$\Delta I_{\text{noise}} = \frac{(N_2 - N_1)h\Delta z}{2t_{\text{spont}}} \frac{\lambda^2}{4\pi\Delta x\Delta y}, \quad (94.14)$$

where the solid angle subtended by the computer array is $\Delta\Omega = \lambda^2/4\pi\Delta x\Delta y$ when using sampling intervals of Δx and Δy . This noise is introduced as a delta-correlated, normally distributed random phasor.

References

1. A. E. Siegman, *Lasers*, University Science Books (1986).
2. H. Zellmer, U. Willamowski, A. Tunnermn, H. Welling, S. Unger, V. Reichel, H.-R. Muller, J. Kirchof, P. Albers, "High-power cw neodymium-doped fiber laser operating at 9.2 W with high beam quality," Opt. Lett., Vol. 20, No.6, pp 578-580 (March 15, 1995).
3. T. Weber, W. Luthy, H. Weber, V. Neuman, H. Berthou, G. Kotrotsios, "A longitudinal and side-pumped single transverse mode double-clad fiber laser with a special silicone coating," Opt. Comm. 115, pp 199-104 (1995).
4. Weber, W. Luthy, H. Weber, V. Neuman, H. Berthou, G. Kotrotsios, J. Dan, and H. Hintermann, "Cladding-Pimped Fiber Laser," IEE J. of Q. Elec., Vol. 31, No. 2, pp 328-329 (February, 1995).
5. P. Glas, M. Naumann, I. Reng, A. Schirrmacher, J. Townsend, "A novel design for high brightness fibre lasers pumped by high-power diodes," J. of Mod. Opt., Vol. 43, No. 5, pp 879-892 (1996).

Input: ex94c.inp

```
c## ex94c
#
# Example: fiber laser, four close fibers
#
# This example illustrates a laser consiting of four cores
# embedded in a larger multimode optical fiber
#
# The fiber consists of a cladding of index 1.5 and a core of index
```

Jump to: [Commands](#), [Theory](#)

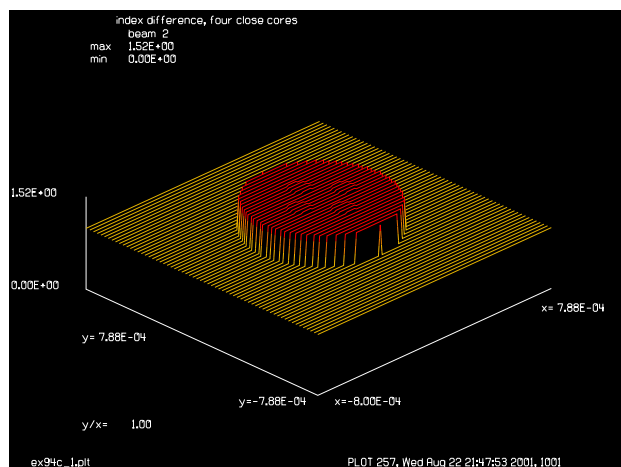


Fig. 94.1. Flat-top pupil distribution. The near-field is renormalized to this shape at the start of each pass.

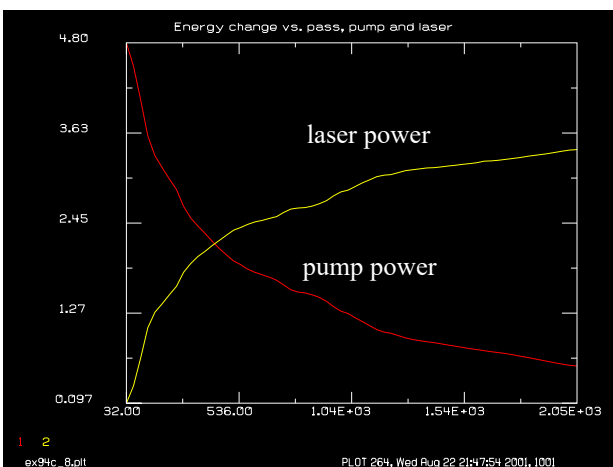


Fig. 94.2. Power conversion per pass. The pump power is initially at about 5 watts. Initially the pump loses some wide angle light from the boundary of the index 1.5 region. Most of the loss, however, is due to absorption by the laser dopant. Pump modes which frequently cross the laser cores. Some pump modes are slow to be absorbed because they operate in the cracks between the laser cores. Ultimately all pump modes will spread sufficiently because of diffraction to be absorbed. The pump light is a 0.53 micron and the laser light is at 0.633 micron so the maximum power conversion is 84%.

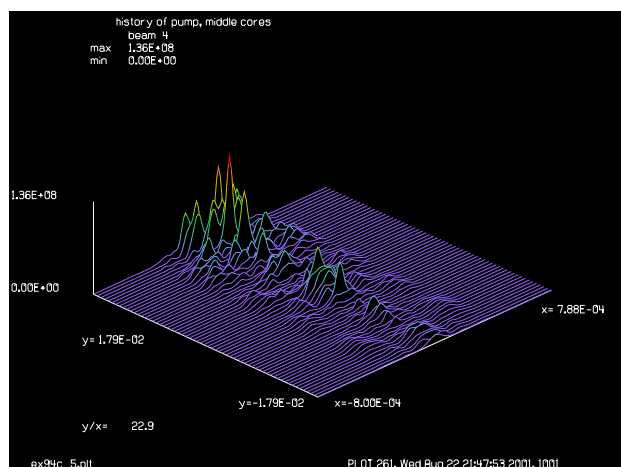


Fig. 94.3. History of pump irradiance from collimated start. The calculation is done every 0.18 micron but displayed in 64 scans of at intervals of 32 steps (11.52 microns). Toward the end of the propagation, the modes which cross the laser cores have largely been absorbed leaving modes in the outer diameter and in the center, in between the cores. Ultimately these long-lived modes will be absorbed as well.

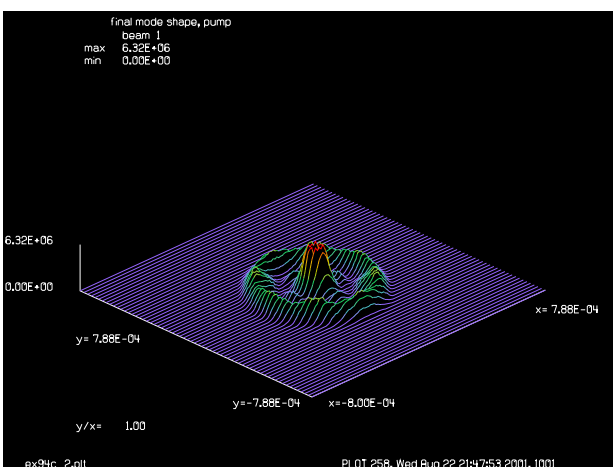


Fig. 94.4. Mode shape of the pump at the point where the calculation was stopped. Part of the long-lived modes are in the zones outside the cores and part at the center inside the region of the cores.

1.52, giving a difference in index of 0.02. The modeling directly

Jump to: [Commands](#), [Theory](#)

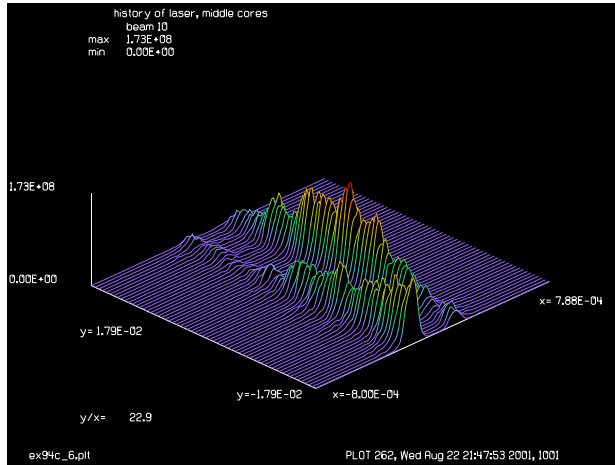


Fig. 94.5. History of the laser cores. The plot shows a series of diagonal scans through the center two cores. The fluctuation in the peaks is due to laser mode beating from trapped light. The weak coupling of the laser cores also results in a relatively slow wobble of the power between the four cores.

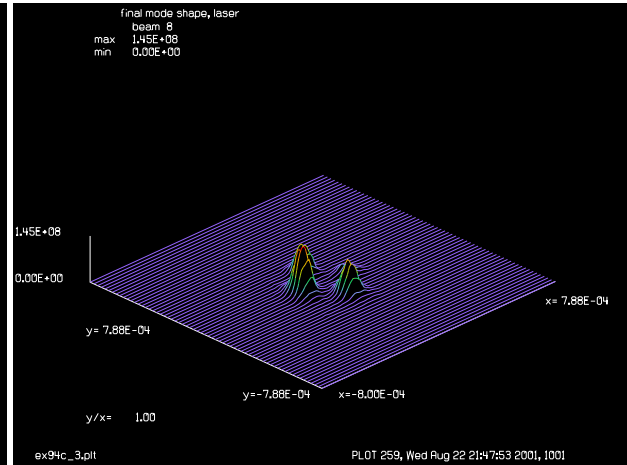


Fig. 94.6. Laser distribution at the end of the calculation. The four laser cores are weakly coupled causing a relatively slow fluctuation in power among the four cores.

```
# treats the phase build-up versus distance. Beam 2 contains the
# pattern of index change associated with the core. The INT2PHAS/TWO
# command is used to implement the incremental index difference defined
# on Beam 2 as a phase effect on Beam 1.
#
#
# Beam      purpose
# ----      -
#   1      propagating pump beam
#   2      refractive index distribution
#   3      history array for pump mode of upper core
#   4      history array for pump mode of middle cores
#   5      history array for pump mode of lower core
#   6      loss function per step
#   7      deposited pump energy
#   8      propagating laser array
#   9      laser medium
#  10      history of laser, middle two cores
#
set/cpu 2                                # valid only for dual CPU machines
mem/set/b 8                             # request 3 megabytes memory
# declare names of integer variables
variab/dec/int pass pass1 count nliney nliney count1 nskip
c set/label/off                          # take off labels of plots
Dist = 1.8e-5                            # step length of propagation
nliney = 128                             # propagating array width
nsteps = 2048
nskip = nsteps/64                        # display 64 strips
nliney = nsteps/nskip                   # length of history arrays
shift = 1.5e-4                           # decentering of cores
array/s 1 nliney                        # beam 1, propagating beam,128x128
```

Jump to: [Commands](#), [Theory](#)


```

nbeam 2 data
nbeam 5 nlinear nlinear data

nbeam 6 nlinear nlinear data
nbeam 7 nlinear nlinear data
nbeam 8 nlinear nlinear beam
nbeam 9 nlinear nlinear 1 data
nbeam 10 nlinear nlinear data

mem/cont
units/field 0 8e-4
variab/set units 3 units
units/set 3 units Dist*nskip
units/set 4 units Dist*nskip
units/set 5 units Dist*nskip
units/set 10 units Dist*nskip
lambda_l = .6328e-4
lambda_p = .53e-4
wavelength/set 0 lambda_l*1e4 1.5
wavelength/set 7 lambda_p*1e4 1.5
wavelength/set 8 lambda_p*1e4 1.5
Clap = .6e-4
Index = .02
clear 1 Index
clear 2 Index
clap/c/c 1 Clap xdec=-shift
clap/c/c 2 Clap xdec=shift
add/inc/con 2 1
clear 1 Index
clap/c/c 1 Clap ydec=shift
add/inc/con 2 1
clear 1 Index
clap/c/c 1 Clap ydec=-shift
add/inc/con 2 1
clear 1 .5
clap/c/c 1 4e-4
add/inc/con 2 1
clear 1 1
add/inc/con 2 1
clear 3 0
clear 4 0
clear 5 0
obs/c 6 Clap xdec=shift
obs/c 6 Clap xdec=-shift
obs/c 6 Clap ydec=shift
obs/c 6 Clap ydec=-shift
loss_coef = 500 # cm-1
loss_per_step = loss_coef*Dist
peak/norm 6 loss_per_step
clear 1 [1-loss_per_step]
add/inc/con 6 1
alpha1 = 2.*pi*Dist/lambda_p
alpha2 = 2.*pi*Dist/lambda_l

# beam 2, index distribution
# beam 3, history of top core
# beam 4, history of middle two cores
# beam 5, history of bottom core
# loss function per step
# array for deposited pump
# laser array
# medium array
# beam 10, history of laser

# set same units 1.25e-5 for all 5 beams
# make variable name units=units of beam 3
# reset units of history arrays (beam 3-5)

# laser wavelength
# laser wavelength
# set wavelength and cladding index

# radius of a individual core
# index difference
# intensity of beam 1 = Index
# intensity of beam 2 = Index
# make left core
# make right core
# sum the two beams, two cores in beam 2
# reinitialize beam 1
# make top core
# sum the two beams, three cores in beam2

# make bottom core
# sum the two beams, four cores in beam 2
# reinitialize beam 1 and beams 3-5
# radius of cladding is 5e-4
# add cladding index
# vacuum index

# phase coefficient per step = k*n*Dist
# phase coefficient per step = k*n*Dist

```

Jump to: [Commands](#), [Theory](#)

```

width = 1.45e13          # Set line width, hz
center = 4.739e14        # Set line center, hz
tspont = 3e-9            # Set spontaneous emission rate, sec
t20 = 3e-1              # Set decay rate for level 2, sec
t10 = 1e-11             # Set decay rate for level 1, sec
mode_sep = 5e11          # Set longitudinal mode separation, hz
                        # (not used)
gaincv = 2./pi/width     # peak of gain response curve
plank = 6.626e-34
hnu = plank*center
index = 1.5
Beinstein = (lambda_1/index)^2/8./pi/tspont*gaincv list
ratexy = 1e7*loss_coef*lambda_p/lambda_1 list
xr2 = ratexy/hnu list
t2 = 1./(1./t20+1./tspont) list
dpopss = xr2*t2 list
gain0 = Beinstein*dpopss list
Dist=
amplification = exp(gain0*Dist) list
xisat = hnu/Beinstein/t2 list

gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/noise
gain/rate/list
pack/set 8 9             # set beams to be packed
set/grid 4 2 4 2
macro/def step/o
    pass = pass+1
    count = count+1
    zreff/se 1 0          # reinitialize propagating distance
    zreff/se 8 0          # reinitialize propagating distance
    geodata/set 1 0 0 1e-4 1e-4 1 1 # control diffraction algorithms
    geodata/set 8 0 0 1e-4 1e-4 1 1 # control diffraction algorithms
c
c take two steps together
c
    int2phas/two 1 2 alpha1 # implement index in beam 2 on beam 1
    int2phas/two 8 2 alpha2 # implement index in beam 2 on beam 1
    split/beam 1 6 7
    variab/set Eng2 1 energy
    mult 7 [1./Dist]
    gain/rate/n2pump 7 9 .6328 # update pump rate
    pack/in                  # pack beams
        gain/rate/steady Dist 1 # implement rate eq. gain
    pack/out                 # unpack beams
    variab/set Eng4 8 energy
    dist Dist 1 8
    clap/s/c 1 7e-4         # rectangular absorbing boundary
    clap/s/c 8 7e-4         # rectangular absorbing boundary
    if count = nskip then
        pass1 = pass1 + 1
        copy/row 1 3 [nlinex/2+1-12] pass1 # copy row 53 of beam 1 to row pass of
                                           # beam 3, central row of top core

```

Jump to: [Commands](#), [Theory](#)

```

copy/row 1 4 [nlinex/2+1] pass1    # copy central row of beam 1 to row pas
                                   # of beam 4, central row of middle core
copy/row 1 5 [nlinex/2+1+12] pass1 # copy row 77 of beam 1 to row pass of
                                   # beam 5, central row of bottom core
copy/row 8 10 [nlinex/2+1] pass1    # copy central row of beam 1 to row pas
                                   # of beam 4, central row of middle core

title laser mode
plot/watch ex94c_3.plt
plot/1 8 ns=64 h=.2

title history of pump, middle cores
plot/watch ex94c_5.plt
plot/1 4 ns=64

title history of laser, middle cores
plot/watch ex94c_6.plt
plot/1 10 ns=64

count1 = count1 + 1
udata/set count1 pass Eng2 Eng4

title pump and laser energy
plot/watch ex94c_8.plt
plot/udata/set y01 y02
plot/udata/seq
count = 0
endif
macro/end
clear 1 1e7                      # pump beam starts at 1e7
clap/c/c 1 4e-4                  # limit pump to large core
clear 8 0.                       # zero laser array
clear 9 0.                       # zerp medium array

pass = 0
macro step/nsteps                # propagate nliney times

title index difference, four close cores
plot/watch ex94c_1.plt
plot/1 2 ns=64 h=.4

plot/watch ex94c_2.plt
title final mode shape, pump
plot/1 1 ns=64 h=.2

plot/watch ex94c_3.plt
title final mode shape, laser
plot/1 8 ns=64 h=.2

plot/watch ex94c_4.plt
title history pump, top core
plot/1 3 ns=64

plot/watch ex94c_5.plt

```

Jump to: [Commands](#), [Theory](#)

```
title history of pump, middle cores  
plot/1 4 ns=64
```

```
plot/watch ex94c_6.plt  
title history of laser, middle cores  
plot/1 10 ns=64
```

```
plot/watch ex94c_7.plt  
title history pump, lower core  
plot/1 5 ns=64
```

```
title Energy change vs. pass, pump and laser  
plot/watch ex94c_8.plt  
plot/udata 1 2
```

Ex95: Optical parametric oscillator

Table. 95.1. Table of Ex95 examples

Ex95a: Interference between a straight and tilted beam	4
Ex95b: Propagation in a uniaxial crystal	7
Ex95c: Optical parametric amplifier: tuned and detuned beams	10
Ex95d: Optical parametric amplifier: aligned and misaligned beams	13
Ex95e: Optical parametric amplifier, misaligned, various crystal lengths	19
Ex95f: Use of <code>mult/tensor</code> for three-wave interactions	22
Ex95g: Use of <code>mult/tensor</code> for four-wave interactions	23
Ex95h: Interference between a straight and tilted beam, in glass	25
Ex95i: Resonator with OPA included	26

This example illustrates several aspects of the optical parametric oscillator (OPL) including the three-wave amplification process, birefringent propagation in uniaxial crystals, and the interaction of k-vector and dispersion effects. Using the common terminology, OPO consists of the interaction of three beams: E_p pump, E_s signal, and E_i idler. The OPO interactions are described in detail in Chap. 9, GLAD Theory Manual.

The basic equations are:

$$\frac{\partial}{\partial z} E_p(\mathbf{k}_p, z) = \frac{i \omega_p}{2 n_p} \chi^{(2)} \left(\frac{\mu_0}{\epsilon_0} \right)^{1/2} E_s(\mathbf{k}_s, z) E_i(\mathbf{k}_i, z) e^{-i \Delta \mathbf{k} \cdot \mathbf{z}}, \quad (95.1)$$

$$\frac{\partial}{\partial z} E_s(\mathbf{k}_s, z) = \frac{i \omega_s}{2 n_s} \chi^{(2)*} \left(\frac{\mu_0}{\epsilon_0} \right)^{1/2} E_p(\mathbf{k}_p, z) E_i^*(\mathbf{k}_i, z) \Delta z e^{i \Delta \mathbf{k} \cdot \mathbf{z}}, \quad (95.2)$$

$$\frac{\partial}{\partial z} E_i(\mathbf{k}_i, z) = \frac{i \omega_i}{2 n_i} \chi^{(2)*} \left(\frac{\mu_0}{\epsilon_0} \right)^{1/2} E_p(\mathbf{k}_p, z) E_s^*(\mathbf{k}_s, z) \Delta z e^{i \Delta \mathbf{k} \cdot \mathbf{z}}, \quad (95.3)$$

$$\Delta \mathbf{k} = \mathbf{k}_p - \mathbf{k}_s - \mathbf{k}_i. \quad (95.4)$$

with plane wave components expressed in terms of k-vectors. The above equations are written in time-independent form, where it has been assumed that the frequencies exactly satisfy the expression

$$\omega_p - \omega_s - \omega_i = 0 \quad (95.5)$$

The kinetic equations given above must be solved simultaneously with the diffraction propagation equations using the split-step method. Chapter 9, GLAD Theory Manual outlines the theory.

Strong interactions can occur only when the k-vector error $\Delta \mathbf{k} \cdot \mathbf{z} \rightarrow 0$. This can occur without having the k-vectors aligned if the dispersion compensates for the mismatch due to alignment. However, misaligned k-vectors will lead to relative shear between the beams which will also cause a reduction in efficiency.

Strongest coupling occurs when all k-vectors are parallel and the scalar value Δk is zero. For most materials ordinary dispersion results in higher index for the shorter wavelength pump resulting in finite Δk and coupling is poor. The dispersion problem may be solved by using a birefringent material so that the different index of refraction of the extraordinary ray (e-ray) may be used to correct for the dispersion of the ordinary rays. By using a combination of ordinary rays (o-rays) and e-rays, it is possible to reduce the k-vector error to zero for copropagating beams, giving optimal interaction strength.

For a uniaxial crystal the index of refraction ellipse may be written:

$$\frac{x^2}{n_o^2} + \frac{y^2}{n_o^2} + \frac{z^2}{n_e^2} = 1. \quad (95.6)$$

In terms of the angle of the crystal axis with respect to the propagation direction:

$$\frac{1}{n_e^2(\theta)} = \frac{\cos^2 \theta}{n_o^2} + \frac{\sin^2 \theta}{n_e^2}. \quad (95.7)$$

See Yariv, p88 [2].

$$\frac{dn_e(\theta)}{d\theta} = \frac{1}{2}n_e^3(\theta) \sin \left\{ 2\theta \left(\frac{1}{n_o^2} - \frac{1}{n_e^2} \right) \right\}. \quad (95.8)$$

See Shen, p128 [3]. The index of refraction in a small neighborhood of the extraordinary ray is[1]:

$$n_e(\theta_e + \Delta\theta) = n_e(\theta) + \left. \frac{dn_e(\theta)}{d\theta} \right|_{\theta_e} \Delta\theta. \quad (95.9)$$

The term $\Delta\theta$ is the incremental angle in the local angular neighborhood of the beam where the spread of k-vectors due to diffraction and aberration are significant. For a finite value of θ , an incremental angle $\Delta\theta$ may be written in terms of the paraxial angles α and β

$$\Delta\theta = \alpha \sin \psi + \beta \sin \psi, \quad (95.10)$$

where ψ is the azimuthal angle of the axis of the uniaxial crystal as it projects onto the α - β plane, similar to the x-y plane, and measured clockwise from the y-axis. Eq. (95.10) is not appropriate when the crystal axis is closely aligned to the optical axis.

Once ψ is defined, the polarization state may be decomposed into the e-ray which has the electric vector parallel to ψ and the o-ray which has the electric vector orthogonal to ψ . Since in GLAD we already have a decomposition into x- and y-states, we may work with this x-y decomposition if the azimuthal angle of the crystal is constrained to be either 0° or 90° . For $\psi = 0^\circ$, x-polarization corresponds to the o-ray and y-

polarization corresponds to e-ray. For $\psi = 90^\circ$, x-polarization corresponds to the e-ray and y-polarization corresponds to o-ray. The phase factor for diffraction propagation from Eq. (3.36) is

$$e^{jk_z z} = e^{jz\sqrt{k^2(1-\alpha^2-\beta^2)}} \approx e^{jk_z z} e^{-\frac{jk_z}{2}(\alpha^2 + \beta^2)} \quad (95.11)$$

The incremental angle $\Delta\theta$ is equivalent to α and β . Including the variation of index with angle from Eq. (95.9), but neglecting third order terms of the form

$$\frac{\partial}{\partial \alpha} n_e(\theta) \alpha^3 \text{ and } \frac{\partial}{\partial \beta} n_e(\theta) \beta^3,$$

we have

$$e^{jk_z z} = e^{j\frac{2}{\pi} \left[n_e(\theta) + \frac{dn_e(\theta)}{d\theta} (\alpha \sin \psi + \beta \cos \psi) \right] z} e^{-j\frac{2\pi}{\lambda} n_e(\theta) \frac{\alpha^2 + \beta^2}{2} z}, \quad (95.12)$$

$$e^{jk_z z} = e^{jk_e z} e^{-jk_e \left[\frac{\alpha^2 + \beta^2}{2} - \frac{1}{n_e(\theta)} \frac{dn_e(\theta)}{d\theta} (\alpha \sin \psi + \beta \cos \psi) \right] z}, \quad (95.13)$$

where $k_e = 2\pi n_e(\theta_e)/\lambda$. The linear terms in Eq. (95.13) causes the well-known walk-off effect. The birefringent walk-off causes the e-component of the beam to acquire a shear with respect to the e-ray direction due to diffraction propagation. The phase shift as a function of spatial frequency is

$$e^{j\frac{2\pi}{\lambda} \frac{dn_e(\theta)}{d\theta} (\xi \sin \psi + \eta \cos \psi) z}, \quad (95.14)$$

where ξ and η are the spatial frequency components. The walk-off as a function of propagation distance is

$$\text{walk-off} = \frac{1}{n_e(\theta)} \frac{dn_e(\theta)}{d\theta} (\hat{x} \sin \psi + \hat{y} \cos \psi) z. \quad (95.15)$$

In the case of KDP, $n_o = 1.51$ and $n_e = 1.47$ for $\theta = 20^\circ$ and $\frac{1}{n_e(\theta)} \frac{dn_e(\theta)}{d\theta} = -.0176$, giving a walk-off angle of about -17 milliradians.

If $\psi = 0$, the y-state of polarization is the e-ray and if $\psi = 90^\circ$ the x-state is the e-ray. It is necessary to set the index of refraction so n_o and n_e correspond to the x- or y-state of polarization appropriately. It is also necessary to include Eq. (95.7) for whichever state constitutes the e-ray.

The walk-off results in a shearing of the e-ray such that the beam moves at a slight angle with respect to the normal to the wavefront. This walk-off significantly reduces the effective coupling because the beams can be made to overlap exactly over the length of the nonlinear crystal.

Phase aberrations cause a beam to have a mix of k-vectors resulting in some degree of detuning of the OPO effect depending on the value of the local slope of the wavefront. The aberrations may be characterized

by the k-vector components in the x- and y-direction and the detuning may be calculated. Evaluating the phase along the z-axis, we have

$$e^{j\mathbf{k} \cdot \mathbf{z}} = e^{jk_z z} \quad (95.16)$$

where k_x , k_y , and k_z are the components of the wavenumber vector.

$$k_x^2 + k_y^2 + k_z^2 = |\mathbf{k}|^2 = k^2 \quad (95.17)$$

We can make the approximation

$$e^{jk_z z} = e^{jz\sqrt{k^2(1-\alpha^2-\beta^2)}} \approx e^{jk_z z} e^{-\frac{jk_z}{2}(\alpha^2 + \beta^2)} \quad (95.18)$$

where $\alpha = k_x/k$ and $\beta = k_y/k$ are the direction cosines in the transverse direction. Equation (13) is the transfer function for a plane wave in homogeneous, isotropic media. For single beam propagation, the term $\exp(jk_z z)$ may be dropped, however it is necessary to include this term for OPO's. The solution of the OPO problem is facilitated by separating Eq. (95.18) into

$$e^{jk_z z} \text{ and } e^{jk_z z} e^{-\frac{jk_z}{2}(\alpha^2 + \beta^2)} \quad (95.19)$$

The effect of $\exp(jk_z z)$ will be included in the solution of the differential equations in the spatial domain and $\exp[jk_z(\alpha^2 + \beta^2)/2]$ is the usual quadratic phase factor and will be included by the standard diffraction propagation routines in GLAD.

Ex95a: Interference between a straight and tilted beam

We may visualize the effect of angular detuning by examining the interference pattern created between an on-axis beam and a beam at a small angle. Fig 95.1 illustrates an on-axis beam and a beam at an angle by the two arrows. Fringes of constant phase and intensity form bisecting the angle for propagation in an ordinary (non-birefringent) material. Fig. 95.2 shows the intensity of the interference pattern. The function `add/op1` has been used to include the absolute phase. However the interference pattern is the same for the two beam having the same index of refraction, whether or not absolute phase is included. Fig. 95.3 shows the irradiance along a z-axis cut. Both the linear phase term (dashed line) and the oscillating irradiance are due to rotating phasor behavior due to the $\exp[-jk_z(\alpha^2 + \beta^2)/2]$ term. The resulting finite Δk term will inhibit OPO interaction.

If the tilted beam can be made to have a different index of refraction, then the k-vector mismatch may be corrected. Fig. 95.4 shows the interference pattern between the two beams still having the same angles but with the off-axis beam having an index of 1.0002 as compared with the on-axis beam at an index of 1.0000. For the off-axis beam, the slightly higher index causes the absolute phase term $\exp(jk_z z)$ to compensate for the angular $\exp[-jk_z(\alpha^2 + \beta^2)/2]$ term so the total phase evolution matches the on-axis beam. The result is lines of constant phase and irradiance parallel to the z-axis. In Fig. 95.5, we see that the

irradiance (solid line) is constant because the total phase of the sum of the two beams is constant even though the angular phase term $\exp[-jkz(\alpha^2 + \beta^2)/2]$ is varying as shown by the dashed curve. The non-varying behavior along the z-axis is necessary for optimum OPO coupling.

GLAD can model the birefringent behavior of uniaxial crystals. Any polar angle—the angle between the crystal axis and the z-axis—may be specified. The polar angle may be varied to achieve the desired index of refraction for the e-ray according to Eq. (10). The azimuthal angle is constrained to 0° , 90° , 180° , and 270° . This restriction causes the crystal axis to align with either the x- or y-axis, so that either the x- or y-axis becomes the o- or e-ray.

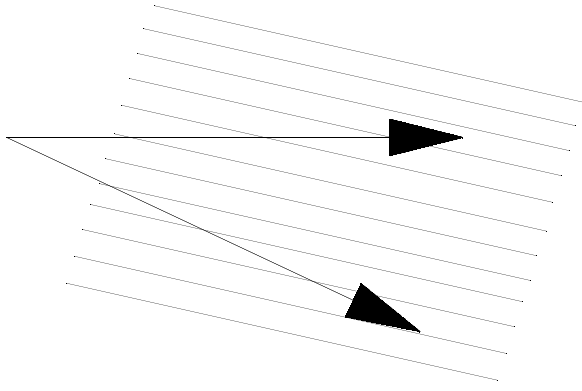


Fig. 95.1. A beam propagating along the z-axis will interfere with a beam at an angle to form a series of interference fringes. The interference fringes are aligned to bisect the angles between the beams. The interference fringes cut the axial ray at a finite angle producing a cyclical behavior or rotating phaser of the form $\exp(j\Delta kz)$ where Δk is the detuning factor from Eq. (95.10). This rotating phaser will reduce the efficiency of OPO coupling.

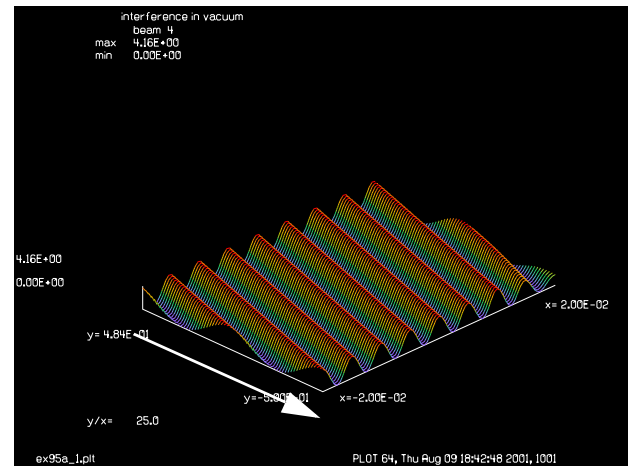


Fig. 95.2. The irradiance between an on-axis beam and a beam propagating angle is shown. The loci of constant intensity bisect the angle between the two beams. This distribution was calculated in Ex95a and illustrates the interference case illustrated in Fig. 95.1. A cut along the z-axis (shown by the arrow) will show rotating phaser behavior, as shown in Fig. 95.3.

Input: `ex95a.inp`

```
c## ex95a
c
c Example 95a: Interference between a straight and tilted beam
c
c This example demonstrates the interference pattern between a beam
c headed along the z-axis with one propagating at an angle.
c
c If the crossing angle is theta, we will have an interference
c pattern with transverse period
c
c p = lambda/theta
c
c and the Talbot distance gives us the axial period
c
c Z-talbot = 2p^2/lambda = 2lambda/theta^2
```

Jump to: [Commands](#), [Theory](#)

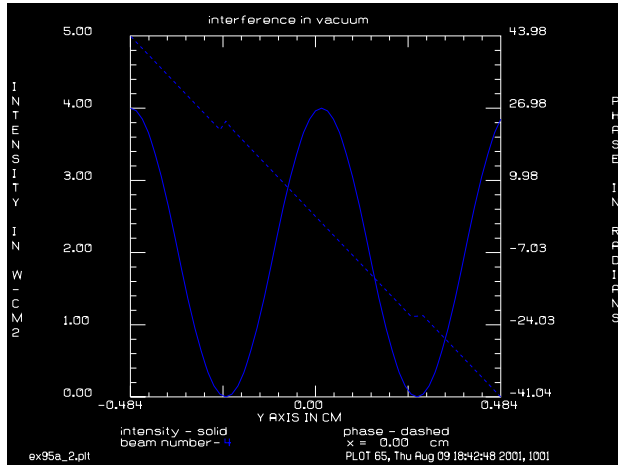


Fig. 95.3. The interference of the beams shown in Fig. 95.2, exhibit a rotating phase behavior when cut along the z-axis. The intensity is shown here as a solid line and the phase as dashed. The rotating phaser will inhibit coupling efficiency.

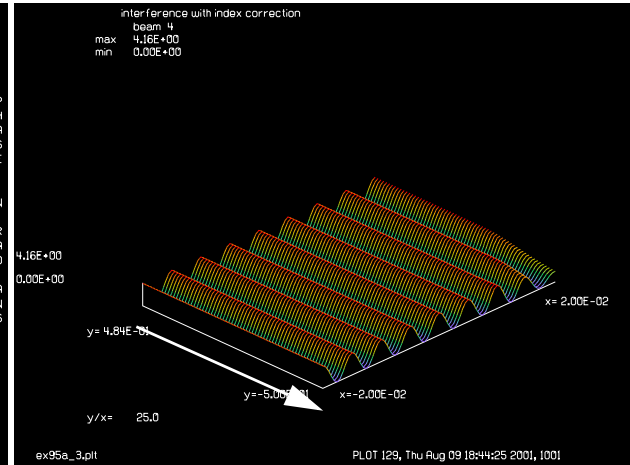


Fig. 95.4. In this case the two beams have exactly the same angles as for Fig. 95.1 and 95.2, but the beam at an angle is propagating with an index of 1.0002 as compared with the index of 1.0000 of the on-axis beam. The slightly higher index causes the phase of the tilted beam to rotate faster such that it matches the on-axis beam giving near-perfect phase matching as shown by a cut along the z-axis (see Fig. 95.4 below).

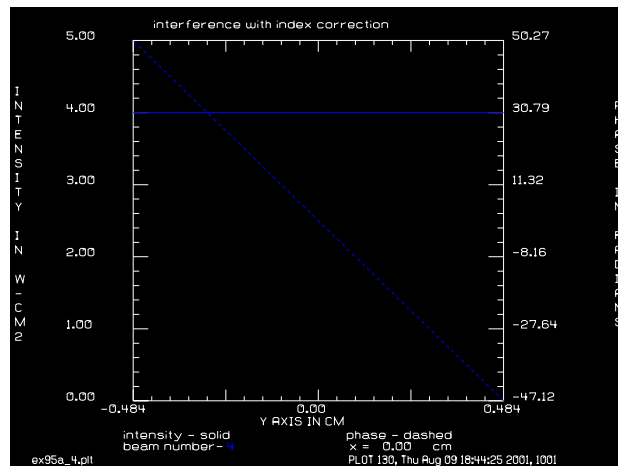


Fig. 95.5. A cut along the z-axis for the case shown in Fig. 95.3 shows illustrates that the there is no rotating phaser behavior, resulting in constant intensity along the z-axis. The dashed line shows that the there is still a relative phase change along the z-axis because of the angle of the beam due to the angular term $\exp[-jkz(\alpha^2 + \beta^2)/2]$ but this is compensated by the higher index in the absolute phase term $\exp(jkz)$.

```

c
c Choosing lambda = 1 micron and theta = 20 milliradian, we have
c
c Z-talbot = .5 cm
c p = .005 cm
c
c The sampling should be 5 to 10 times the period to be sampled.
c dx = .0005

```

Jump to: [Commands](#), [Theory](#)

```

c
variab/dec/int row
mem/set/b 8
array/set 1 512
nbeam 2
units/s 1 .0005           # Set units.
units/s 2 .0005           # Set units.
nbeam 4 data
array/set 4 512 64 data
clear 4 0.
units/s 4 .0005 .015625
wavelength/set 0 1        # Set wavelength.
gaus/c/c 1 1 .04 8
gaus/c/c 2 1 .04 8
abr/tilt 2 200 rn=1 az=90
plot/watch ex95a_1.plt
title interference in vacuum
macro/def ex95a/o
  row = row + 1 list
  prop .015625
  clear 3 0
  add/opl 3 1 2
  copy/row 3 4 257 row
  plot/1 4 h=.1 ns=64 xrad=.02 yrad=.5
macro/end
macro ex95a/64
plot/watch ex95a_2.plt
plot/y 4 fmax=5.
wavelength/set 2 1 1.0002
zreff 0 0.
gaus/c/c 1 1 .04 8
gaus/c/c 2 1 .04 8
abr/tilt 2 200 rn=1 az=90
row = 0
clear 4 0
plot/watch ex95a_3.plt
title interference with index correction
macro ex95a/64
plot/watch ex95a_4.plt
plot/y 4 fmax=5.

```

Ex95b: Propagation in a uniaxial crystal

Example 95b illustrates walk-off as a function of polar and azimuthal angle. Fig. 95.6 shows the o-ray which is centered and Fig. 95.7 shows the e-ray decentered by about -.136 cm. Fig. 95.8 shows the separation of o-ray with x-polarization from the e-ray with y-polarization.

Input: ex95b.inp

```

c## ex95b
c
c Example 95b: Propagation in a uniaxial crystal

```

Jump to: [Commands](#), [Theory](#)

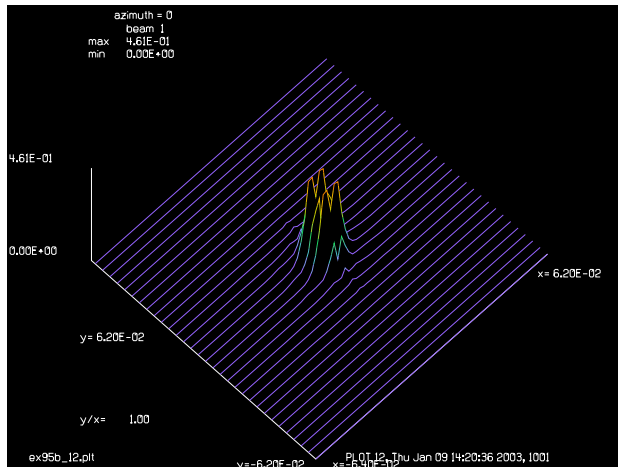


Fig. 95.6. O-ray is still centered after 0.8 cm propagation through uniaxial crystal.

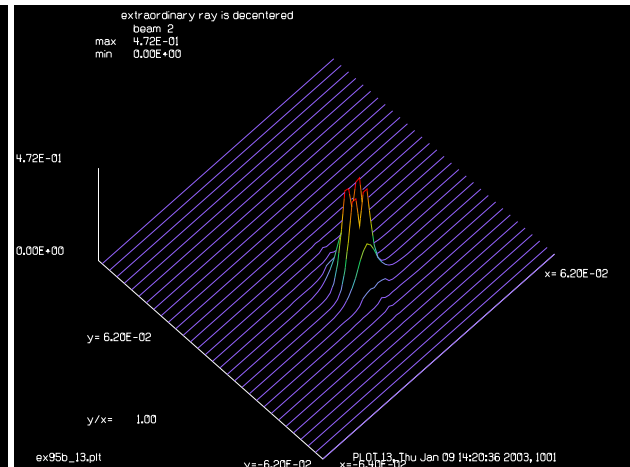


Fig. 95.7. E-ray shows walk-off because of crystal axis at 0° azimuthal angle (up). The walk-off is downward because the index of the e-ray is lower.

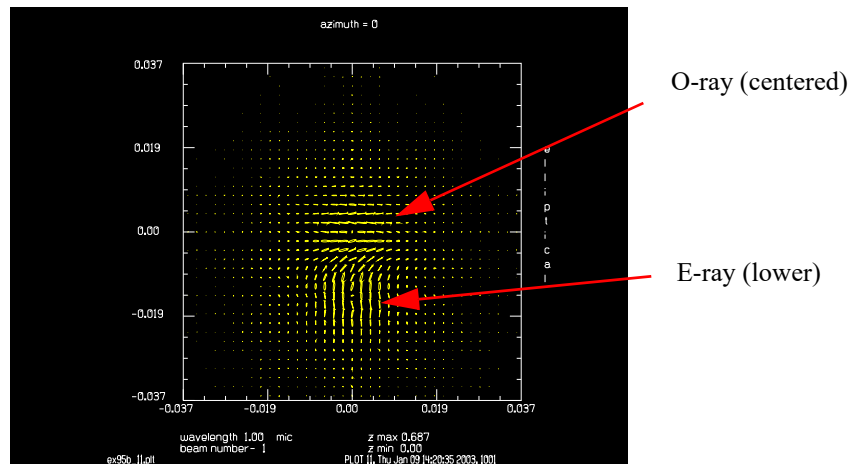


Fig. 95.8. O-ray has x linear polarization and e-ray has y-linear polarization if the crystal axis is 0° azimuth angle (up) for index of e-ray lower.

```

c
alias Name ex95b
set/den 32 32
array/s 1 64 64 1
unit 1 .002
nbeam 2 data
jones/set ar=.5 cr=.5
jones/mult 1
wavelength/set 1 1. 1.51 1.47
clap/c/c 1 .01
crystal/uniaxial/set 20 0
prop .8
title azimuth = 0
plot/watch @Name_1.plt
plot/e 1
plot/watch @Name_2.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/1
zreff/set 1 0
global/def 1 0
array/set 1 64 64 1
unit 1 .002
jones/set ar=.5 cr=.5
jones/mult 1
wavelength/set 1 1. 1.51 1.47
clap/c/c 1 .01
fitgeo 1
crystal/uniaxial/set 20 90
prop .8
title azimuth = 90
plot/watch @Name_3.plt
plot/e 1
plot/watch @Name_4.plt
plot/1
zreff/set 1 0
global/def 1 0
array/set 1 64 64 1
unit 1 .002
jones/set ar=.5 cr=.5
jones/mult 1
wavelength/set 1 1. 1.51 1.47
clap/c/c 1 .01
crystal/uniaxial/set 20 180
prop .8
title azimuth = 180
plot/watch @Name_5.plt
plot/e 1
plot/watch @Name_6.plt
plot/1
zreff/set 1 0
global/def 1 0
array/set 1 64 64 1
unit 1 .002
jones/set ar=.5 cr=.5
jones/mult 1
wavelength/set 1 1. 1.51 1.47
clap/c/c 1 .01
crystal/uniaxial/set 20 270
prop .8
title azimuth = 180
plot/watch @Name_7.plt
plot/e 1
plot/watch @Name_8.plt
plot/1
zreff/set 1 0
global/def 1 0
array/set 1 64 64 1
unit 1 .002
jones/set ar=.5 cr=.5
jones/mult 1
wavelength/set 1 1. 1.51 1.47

```

Jump to: [Commands](#), [Theory](#)

```

clap/c/c 1 .01
crystal/uniaxial/set 20 0
prop .8
title azimuth = 0
plot/watch @Name_9.plt
plot/e 1
plot/watch @Name_10.plt
plot/l
set/win/rel 20 80 20 80
plot/w ex95b_11.plt
plot/e 1
copy/con 1 2
jones/set ar=1 cr=0 dr=0 br=0
jones/mult 1
jones/set ar=0 cr=0 dr=1 br=0
jones/mult 2
plot/w ex95b_12.plt
plot/l 1 theta=42
title ordinary ray is centered
plot/w ex95b_13.plt
title extraordinary ray is decentered
plot/l 2 theta=42

```

Ex95c: Optical parametric amplifier: tuned and detuned beams

Example 95c illustrates OPO interaction for perfect tuning (Fig. 95.9) and for significant detuning (Fig. 95.10).

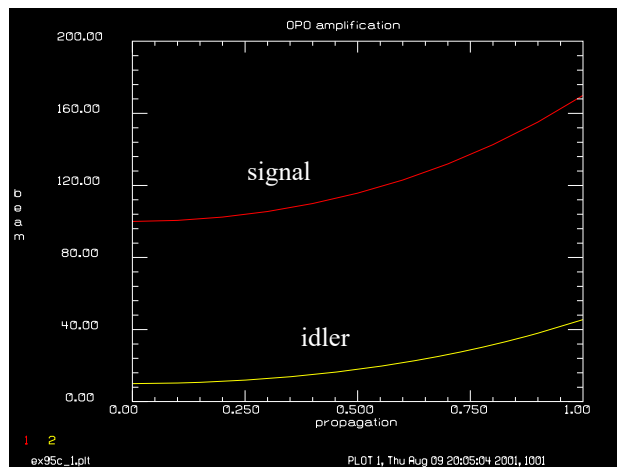


Fig. 95.9. Amplification of signal and idler beams for perfect tuning.

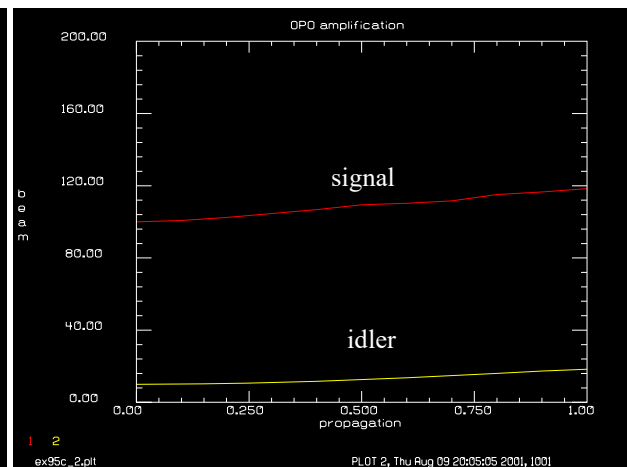


Fig. 95.10. Amplification of signal and idler beams for poor tuning.

Input: ex95c.inp

```

c## ex95c
c
c Example 95c: Optical parametric amplifier: tuned and detuned beams
c -----

```

Jump to: [Commands](#), [Theory](#)

```

c
nbeam 3                                # Define three beams
lambda_p = .5                          # pump, vacuum wavelength
lambda_s = .75                          # signal, vacuum wavelength
#
# vacuum wavelength for idler may be solved by the condition that
# the sum of frequencies is zero -- identical to the sum of
# inverse vacuum wavelengths
#
array/set 1 64 64                       # Set arrays to 64 x 64
array/set 2 64 64 1                     # Set arrays to 64 x 64
array/set 3 64 64                       # Set arrays to 64 x 64
lambda_i = 1./(1./lambda_p - 1./lambda_s)
lambda_i=                               # idler, vacuum wavelength
n_p = 1.52
n_i = 1.50
n_o = 1.51
n_e = 1.54
wavelength/set 1 lambda_p n_p           # Define wavelength of pump beam
wavelength/set 2 lambda_s n_o n_e       # Define wavelength of signal
wavelength/set 3 lambda_i n_i           # Define wavelength of idler
n_s = lambda_s*(n_p/lambda_p - n_i/lambda_i)
n_s=                                     # desired index for signal
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
n_s=                                     # calculated index for signal
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list
crystal/uniaxial/set polar 0
units/field 0 .1                        # Set units
gaus/c/c 1 1e6 .05
clear 2 1e2                             # Define signal intensity
clear 3 1e1                             # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
udata/clear                             # Clear user data variables
wave4/set 0.                            # Set zstart for four-wave mixing
pack/set 1 2 3                          # Define arrays for memory processing
variab/set Peak1 1 peak
variab/set Peak2 2 peak
variab/set Peak3 3 peak
udata/set 1 0. Peak1 Peak2 Peak3        # First call to user data
zstep = .1                              # Set step length
c
c Define macro for kinetic step
c
macro/def kinstep/over
    pack/in
        opo/constant zstep=zstep kappa=1e-3 nstep=50 xyx
    pack/out
    pass = pass + 1
    z = z + zstep
    variab/set Peak1 1 peak

```

Jump to: [Commands](#), [Theory](#)

```

    variab/set Peak2 2 peak
    variab/set Peak3 3 peak
    udata/set pass z Peak1 Peak2 Peak3
udata
macro/end
c
c  Run calculations
c
variab/set energy1 1 energy      # Define Param 1 = Energy of Beam 1
variab/set energy2 2 energy      # Define Param 2 = Energy of Beam 2
variab/set energy3 3 energy      # Define Param 3 = Energy of Beam 3
Engtot1 = energy1 + energy2 + energy3
z = 0.                          # Initialize z-axis
pass = 1                        # Initialize udata index
macro/run kinstep/10            # call propagation macro
variab/set energy1 1 energy
variab/set energy2 2 energy
variab/set energy3 3 energy
Engtot2 = energy1 + energy2 + energy3
Engtot1=
Engtot2=
x=Engtot2-Engtot1 list
c
c  Set up plot titles
c
udata/xlab propagation distance (cm)
udata/ylab beam power
title OPO amplification
udata/list
plot/watch ex95c_1.plt/o
plot/udata 2 3 min=0 max=200
zreff/se 1 0
zreff/se 2 0
zreff/se 3 0
global/opl
global/opl/set 1 0. 0.
global/opl/set 2 0. 0.
global/opl/set 3 0. 0.
gaus/c/c 1 1e6 .05
clear 2 1e2                     # Define signal intensity
clear 3 1e1                     # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
polar = 55.1
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list
crystal/uniaxial/set polar 0
z = 0.                          # Initialize z-axis
pass = 1                        # Initialize udata index
udata/maxrows 0
macro/run kinstep/10            # call propagation macro
plot/watch ex95c_2.plt/o

```

Jump to: [Commands](#), [Theory](#)


```
plot/udata 2 3 min=0 max=200
```

Ex95d: Optical parametric amplifier: aligned and misaligned beams

Example 95d shows amplification for an idler beam that is initially zero, with the signal and pump beams aligned (Fig. 95.11) and for the signal beam at an angle to the pump (Fig. 95.12). The OPO gain is significantly lower when the signal is at an angle because the crystal polar angle was set to compensate for an on-axis signal beam and is, therefore, detuned for the off-axis signal beam. In addition the off-axis signal beam causes the idler beam to be generated at an angle. Figs. 95.13 and 95.16 show the far-field distributions for the signal and idler beams. The angle of the idler beam is about twice the angle of the signal beam because the wavelength of the idler beam at $1.5\ \mu$ is about twice the wavelength of the signal beam at $0.75\ \mu$ so the angle of the idler beam must be twice to cancel the signal k-vector and of opposite sign. See Fig. 95.17. The width of the idler beam is significantly larger than the signal beam.

An important aspect of the method used in GLAD is that the angular detuning is not treated explicitly in the kinetic equations, but rather by the fact that the diffraction propagation causes the complex amplitude of the various spatial frequencies to rotate according to $\exp[-jk(\alpha^2 + \beta^2)/2]$. This may be definitively demonstrated by making a series of calculations all of which have the same gain-length but which differ in the total length. The gain-length is $kz|A_p|^2$, where z is the length of the interaction region. For negligible pump-depletion, perfect dispersion correction, and no detuning, interactions which have the same gain-length product will have the same amplification. The characteristic length for diffraction propagation is based on the wavefront slopes. For an angle θ between pump and signal beam, the detuning will be π for a distance of θ/λ^2 , where λ is the wavelength in the medium.

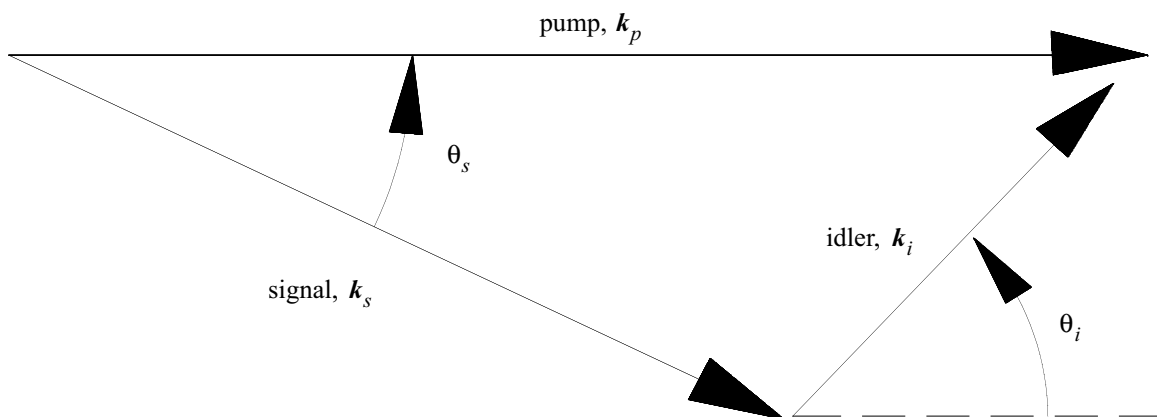


Fig. 95.11. A k-vector diagram showing the pump, signal, and idler beams. The length is inversely proportional to the wavelength so k_p is the longest vector and is in the direction of the z-axis for Ex. 95d. The signal k-vector k_s is twice the length of the idler k_i , so the angle of the idler beam θ_i must be twice θ_s to get the minimum value of $\Delta k \cdot z$. The idler beam develops to the proper angle when started from zero simply by calculating OPO and diffraction equations.

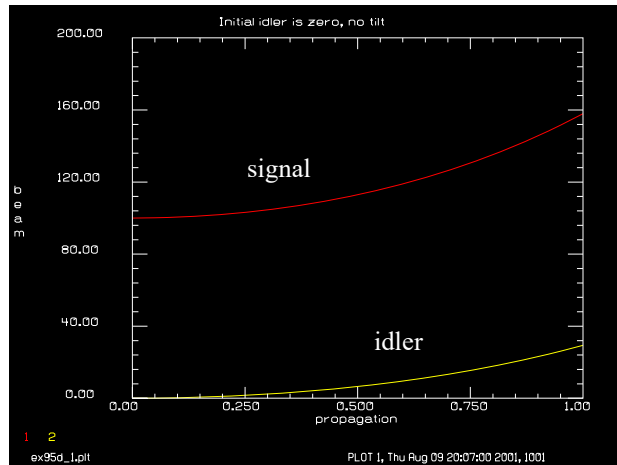


Fig. 95.12. Amplification of signal and idler beams for aligned signal and zero initial idler power. Compare with Fig. 95.9.

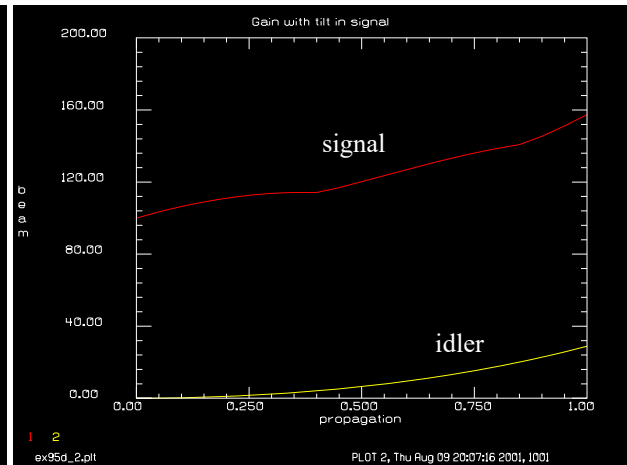


Fig. 95.13. Amplification of signal and idler beams for misaligned signal and zero initial idler power. Compare with Fig. 95.10

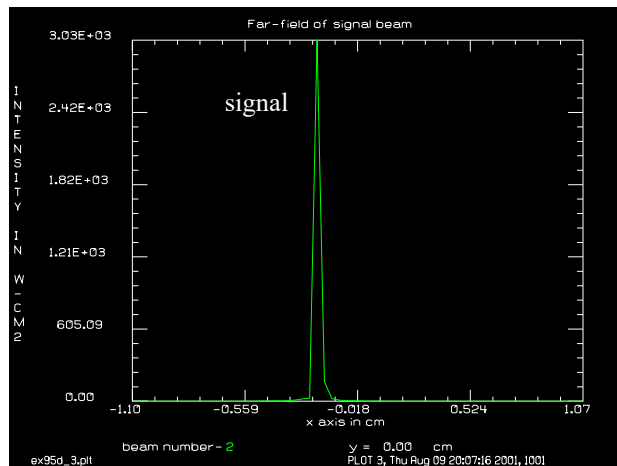


Fig. 95.14. Far-field of signal beam indicating off-axis angle by the decentered peak.

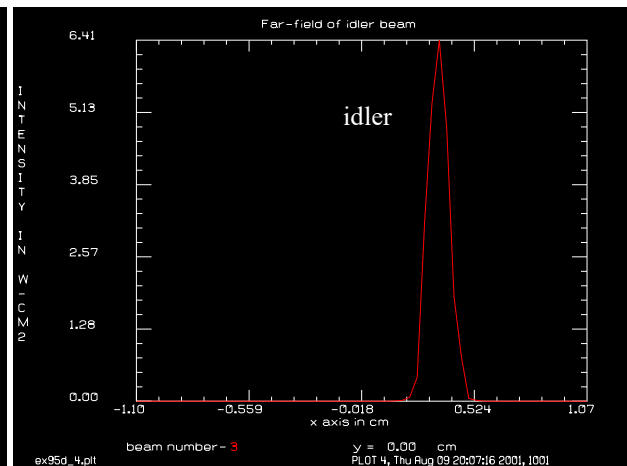


Fig. 95.15. Far-field of the idler beam. The angle is twice the magnitude and of opposite sign to compensate the k-vectors with a difference of two in wavelength. Note the idler is much wider as it is limited only by angular detuning.

Input: ex95d.inp

```
c## ex95d
c
c Example 95d: Optical parametric amplifier: aligned and misaligned beams
c -----
c
c nbeam 3                                     # Define three beams
c lambda_p = .5                             # pump, vacuum wavelength
c lambda_s = .75                             # signal, vacuum wavelength
c #
c # vacuum wavelength for idler may be solved by the condition that
```

Jump to: [Commands](#), [Theory](#)

```

# the sum of frequencies is zero -- identical to the sum of
# inverse vacuum wavelengths
#
array/set 1 64 64                                # Set arrays to 4x4
array/set 2 64 64 1                              # Set arrays to 4x4
array/set 3 64 64                                # Set arrays to 4x4
lambda_i = 1./(1./lambda_p - 1./lambda_s)
lambda_i=                                         # idler, vacuum wavelength
n_p = 1.52
n_i = 1.50
n_o = 1.51
n_e = 1.54
wavelength/set 1 lambda_p n_p                    # Define wavelength of pump beam
wavelength/set 2 lambda_s n_o n_e                # Define wavelength of signal
wavelength/set 3 lambda_i n_i                    # Define wavelength of idler
n_s = lambda_s*(n_p/lambda_p - n_i/lambda_i)
n_s=                                              # desired index for signal
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
n_s=                                              # calculated index for signal
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list
crystal/uniaxial/set polar 0
units/field 0 .1                                # Set units
gaus/c/c 1 1e6 .05
clear 2 1e2                                     # Define signal intensity
clear 3 0.                                       # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
udata/clear                                     # Clear user data variables
wave4/set 0.                                    # Set zstart for four-wave mixing
pack/set 1 2 3                                  # Define arrays for memory processing
variab/set Peak1 1 peak
variab/set Peak2 2 peak
variab/set Peak3 3 peak
udata/set 1 0. Peak1 Peak2 Peak3                # First call to user data
zstep = .05                                     # Set step length
c
c Define macro for kinetic step
c
macro/def kinstep/over
  pack/in
  opo/constant zstep=zstep kappa=1e-3 nstep=50 xyx
  pack/out
  pass = pass + 1
  z = z + zstep
  variab/set Peak1 1 peak
  variab/set Peak2 2 peak
  variab/set Peak3 3 peak
  udata/set pass z Peak1 Peak2 Peak3
  udata
macro/end
c

```

Jump to: [Commands](#), [Theory](#)

```

c  Run calculations
c
variab/set energy1 1 energy      # Define Param 1 = Energy of Beam 1
variab/set energy2 2 energy      # Define Param 2 = Energy of Beam 2
variab/set energy3 3 energy      # Define Param 3 = Energy of Beam 3
Engtot1 = energy1 + energy2 + energy3
z = 0.                            # Initialize z-axis
pass = 1                          # Initialize udata index
macro/run kinstep/20             # call propagation macro
variab/set energy1 1 energy
variab/set energy2 2 energy
variab/set energy3 3 energy
Engtot2 = energy1 + energy2 + energy3
Engtot1=
Engtot2=
x=Engtot2-Engtot1 list
c
c  Set up plot titles
c
udata/xlab propagation distance (cm)
udata/ylab beam power
title Initial idler is zero, no tilt
udata/list
plot/watch ex95d_1.plt/o
plot/udata 2 3 min=0 max=200
zreff/se 1 0
zreff/se 2 0
zreff/se 3 0
global/def 1 0 0 0
global/def 2 0 0 0
global/def 3 0 0 0
global/op1
global/op1/set 1 0. 0.
global/op1/set 2 0. 0.
global/op1/set 3 0. 0.
gaus/c/c 1 1e6 .05
clear 2 1e2                      # Define signal intensity
deltak_s = .2 list
lambda_s=
period_s = sqrt(pi*lambda_s*1e-4/deltak_s)
period_s=
abr/tilt 2 [1./period_s] rn=1 az=90
clear 3 0                        # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list
crystal/uniaxial/set polar 0
zstep = .05                      # Set step length
z = 0.                            # Initialize z-axis
pass = 1                          # Initialize udata index

```

Jump to: [Commands](#), [Theory](#)

```

udata/maxrows 0
macro/run kinstep/20                                # call propagation macro
plot/watch ex95d_2.plt
title Gain with tilt in signal
plot/udata 2 3 min=0 max=200
wavelength/set 2 lambda_s 1. 1.                    # Define wavelength of signal
wavelength/set 3 lambda_i 1. 1.                    # Define wavelength of signal
global 2
zreff/se 2
lens 2 100
prop 100 2
lens 3 100
prop 100 3
plot/w ex95d_3.plt
title Far-field of signal beam
plot/x/i 2
variab/set Units 2 units
rescale/units 3 Units
plot/w ex95d_4.plt
title Far-field of idler beam
plot/x/i 3
c## ex95c
c
c Example 95c: Optical parametric amplifier: tuned and detuned beams
c -----
c
nbeam 3                                              # Define three beams
lambda_p = .5                                       # pump, vacuum wavelength
lambda_s = .75                                       # signal, vacuum wavelength
#
# vacuum wavelength for idler may be solved by the condition that
# the sum of frequencies is zero -- identical to the sum of
# inverse vacuum wavelengths
#
array/set 1 64 64                                    # Set arrays to 64 x 64
array/set 2 64 64 1                                  # Set arrays to 64 x 64
array/set 3 64 64                                    # Set arrays to 64 x 64
lambda_i = 1./(1./lambda_p - 1./lambda_s)
lambda_i=                                           # idler, vacuum wavelength
n_p = 1.52
n_i = 1.50
n_o = 1.51
n_e = 1.54
wavelength 1 lambda_p n_p                            # Define wavelength of pump beam
wavelength 2 lambda_s n_o n_e                        # Define wavelength of signal
wavelength 3 lambda_i n_i                            # Define wavelength of idler
n_s = lambda_s*(n_p/lambda_p - n_i/lambda_i)
n_s=                                                # desired index for signal
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
n_s=                                                # calculated index for signal
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list

```

Jump to: [Commands](#), [Theory](#)

```

crystal/uniaxial/set polar 0
units/field 0 .1                                # Set units
gaus/c/c 1 1e6 .05
clear 2 1e2                                       # Define signal intensity
clear 3 1e1                                       # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
udata/clear                                       # Clear user data variables
wave4/set 0.                                     # Set zstart for four-wave mixing
pack/set 1 2 3                                   # Define arrays for memory processing
variab/set Peak1 1 peak
variab/set Peak2 2 peak
variab/set Peak3 3 peak
udata/set 1 0. Peak1 Peak2 Peak3                # First call to user data
zstep = .1                                       # Set step length
c
c Define macro for kinetic step
c
macro/def kinstep/over
  pack/in
  opo/constant zstep=zstep kappa=1e-3 nstep=50 xyx
  pack/out
  pass = pass + 1
  z = z + zstep
  variab/set Peak1 1 peak
  variab/set Peak2 2 peak
  variab/set Peak3 3 peak
  udata/set pass z Peak1 Peak2 Peak3
udata
macro/end
c
c Run calculations
c
variab/set energy1 1 energy                      # Define Param 1 = Energy of Beam 1
variab/set energy2 2 energy                      # Define Param 2 = Energy of Beam 2
variab/set energy3 3 energy                      # Define Param 3 = Energy of Beam 3
Engtot1 = energy1 + energy2 + energy3
z = 0.                                           # Initialize z-axis
pass = 1                                         # Initialize udata index
macro/run kinstep/10                             # call propagation macro
variab/set energy1 1 energy
variab/set energy2 2 energy
variab/set energy3 3 energy
Engtot2 = energy1 + energy2 + energy3
Engtot1=
Engtot2=
x=Engtot2-Engtot1 list
c
c Set up plot titles
c
udata/xlab propagation distance (cm)
udata/ylab beam power
title OPO amplification
udata/list

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch ex95c_1.plt/o
plot/udata 2 3 min=0 max=200
zreff 1 0
zreff 2 0
zreff 3 0
global/op1
global/op1/set 1 0. 0.
global/op1/set 2 0. 0.
global/op1/set 3 0. 0.
gaus/c/c 1 1e6 .05
clear 2 1e2                                # Define signal intensity
clear 3 1e1                                # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
polar = 55.1
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list
crystal/uniaxial/set polar 0
z = 0.                                     # Initialize z-axis
pass = 1                                  # Initialize udata index
udata/maxrows 0
macro/run kinstep/10                       # call propagation macro
plot/watch ex95c_2.plt/o
plot/udata 2 3 min=0 max=200

```

Ex95e: Optical parametric amplifier, misaligned, various crystal lengths

In example Ex95e, $\theta = 0.005713$, the period of the interference pattern is 0.008581, the index of the signal beam is 1.53 and the length for π detuning is 1.5 cm from the expression λ/θ^2 . Fig. 95.18 shows the peak power of the idler beam for a gain-length held constant at 1.0 for a range of distances from 0 to 1.5 cm for perfect tuning, as calculated in Ex95e. For each choice of crystal length the coupling coefficient was reset to maintain the gain-length at 1.0. For pump and signal beams perfectly aligned, there is no fall off of idler power for larger distances. In Fig. 95.18 peak idler power is calculated for a range of crystal lengths from 0 to 1.5. The peak idler power drops of to a minimum value at a crystal length of about 1.0, somewhat less than the π tuning length of 1.5.

Input: ex95e.inp

```

c## ex95e
c
c Example 95e: Optical parametric amplifier, misaligned,
c various crystal lengths
c -----
c
variable/dec/int pass pass1 nline
nbeam 3                                # Define three beams
lambda_p = .5                          # pump, vacuum wavelength
lambda_s = .75                         # signal, vacuum wavelength
#
# vacuum wavelength for idler may be solved by the condition that

```

Jump to: [Commands](#), [Theory](#)

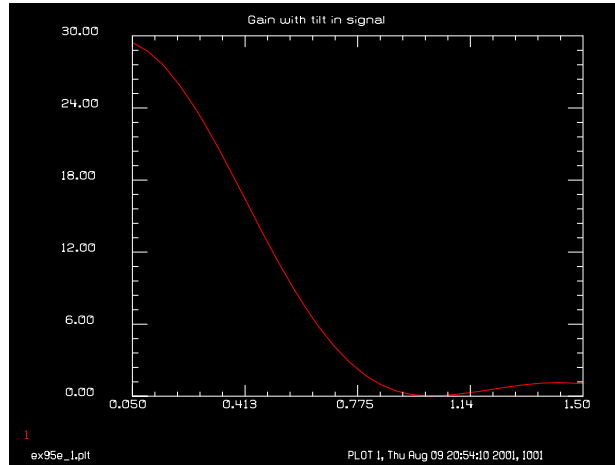


Fig. 95.16. Peak power of the idler beam over a range of crystal lengths where the coupling coefficient is reset to maintain constant gain-length. For Ex. 95e, the signal beam is misaligned by 0.00 causing minimum coupling efficiency for a crystal length equal to the π detuning distance of 1.5 cm. Because of the combined effect with the signal and idler beams, the minimum occurs ahead of the π detuning distance. The drop off in gain arises from the diffraction propagation and the relative angle of the beams—not the kinetic equations.

```
# the sum of frequencies is zero -- identical to the sum of
# inverse vacuum wavelengths
#
mem/set/b 2
nline=256
array/s 1 nline nline
array/s 2 nline nline 1
array/s 3 nline nline
lambda_i = 1./(1./lambda_p - 1./lambda_s)
lambda_i=                                # idler, vacuum wavelength
n_p = 1.52
n_i = 1.50
n_o = 1.51
n_e = 1.54
wavelength/set 1 lambda_p n_p             # Define wavelength of pump beam
wavelength/set 2 lambda_s n_o n_e         # Define wavelength of signal
wavelength/set 3 lambda_i n_i             # Define wavelength of idler
n_s = lambda_s*(n_p/lambda_p - n_i/lambda_i)
n_s=                                     # desired index for signal
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
n_s=                                     # calculated index for signal
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list
crystal/uniaxial/set polar 0
units/field 0 .1                          # Set units
gaus/c/c 1 1e6 .05
clear 2 1e2                               # Define signal intensity
clear 3 0.                                # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
```

Jump to: [Commands](#), [Theory](#)


```

udata/clear                                # Clear user data variables
wave4/set 0.                               # Set zstart for four-wave mixing
pack/set 1 2 3                             # Define arrays for memory processing
variab/set Peak3 3 peak
udata/set 1 0. Peak3                       # First call to user data
zstep = .05                                # Set step length
deltak_s = .2 list
lambda_s=
period_s = .25*sqrt(pi*lambda_s*1e-4/deltak_s)
period_s=
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
Talbot = n_s*period_s^2/lambda_s/1.e-4
Talbot=
c
c Define macro for kinetic step
c
macro/def kinstep/over
    pack/in
        opo/constant zstep=zstep kappa=Kappa nstep=25 xyx
    pack/out
        z = z + zstep
        pass = pass + 1
macro/end
macro/def length/over
    pass1 = pass1 + 1
    z = 0.                                # Initialize z-axis
    pass = 1                              # Initialize udata index
    zreff/se 1 0
    zreff/se 2 0
    zreff/se 3 0
    global/opl
    global/opl/set 1 0. 0.
    global/opl/set 2 0. 0.
    global/opl/set 3 0. 0.
    array/s 1 nline nline
    array/s 2 nline nline 1
    array/s 3 nline nline
    units/field 0 .1                      # Set units
    gaus/c/c 1 1e6 .05                   # Define signal intensity
    clear 2 1e2                           # Define idler intensity
    abr/tilt 2 [1./period_s] rn=1 az=90
    clear 3 0
    jones/set ar=0 cr=1
    jones/mult 2
    Kappa = 1e-3/zstep/pass1
    macro/run kinstep/pass1               # call propagation macro
    variab/set Peak3 3 peak
    udata/set pass1 pass1*zstep Peak3
    udata/list
macro/end
c

```

Jump to: [Commands](#), [Theory](#)

```

c Run calculations
c
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list
crystal/uniaxial/set polar 0
z = 0. # Initialize z-axis
udata/maxrows 0
pass1 = 0
macro/run length/30 # call propagation macro
plot/watch ex95e_1.plt
title Gain with tilt in signal
plot/udata 1 min=0 max=30
wavelength/set 2 lambda_s 1. 1. # Define wavelength of signal
wavelength/set 3 lambda_i 1. 1. # Define wavelength of signal
lens 2 100
dist 100 2
lens 3 100
dist 100 3
set/label/off
plot/w ex95e_2.plt
title Far-field of signal beam
plot/x/i 2
variab/set Units 2 units
rescale/units 3 Units
plot/w ex95e_3.plt
title Far-field of idler beam
plot/x/i 3
plot/w ex95e_4.plt
title Far-field of idler beam
plot/x/i 1

```

Ex95f: Use of `mult/tensor` for three-wave interactions

Input: `ex95f.inp`

```

c## ex95f
c
c Example 95f: Use of mult/tensor for three-wave interactions
c
echo/on
array/s 1 64 64
nbeam 3
clear 3 0
mult/tensor/3 3 1 2 xxx
title all real
field 3
pause
clear 3 0
phase/piston 1 90
mult/tensor/3 3 1 2 xxx
title no conjugation, beam 1 imaginary
field 1
field 3

```

Jump to: [Commands](#), [Theory](#)

```

pause
clear 3 0
mult/tensor/3 3 1 2 Xxx
title conjugate Beam 3
field 3
pause
clear 3 0
mult/tensor/3 3 1 2 xXx
title conjugate Beam 1
field 3
clear 1 1
phase/piston 2 90
clear 3 0
mult/tensor/3 3 1 2 xxX
title conjugate Beam 2
field 3
pause
array/s 3 64 64 1
clear 3 0
mult/tensor/3 3 1 2 yxX
title conjugate Beam 2, store in y-state, beam 3
field 3
pause
array/s 1 64 64 1
jones/set ar=0 br=0 cr=1 dr=0
jones/mult 1
clear 3 0
mult/tensor/3 3 1 2 xyX
title conjugate Beam 2, y-state Beam 1
field 1
field 3
pause
clear 3 0
phase/piston 1 90
field 1
pause
mult/tensor/3 3 1 2 xYX
title conjugate Beams 1 and 2, y-state Beam 1
field 3

```

Ex95g: Use of `mult/tensor` for four-wave interactions

Input: `ex95g.inp`

```

c## ex95g
c
c Example 95g: Use of mult/tensor for four-wave interactions
c
echo/on
array/s 1 64 64
nbeam 4
clear 4 0

```

Jump to: [Commands](#), [Theory](#)

```
mult/tensor/4 4 1 2 3 xxxx
title all real
field 4
pause
clear 4 0
mult/tensor/4 4 1 2 3 xxxx
title no conjugation, beam 1 imaginary
field 1
field 4
pause
clear 4 0
mult/tensor/4 4 1 2 3 Xxxx
title conjugate Beam 4
field 4
pause
clear 4 0
mult/tensor/4 4 1 2 3 xXxx
title conjugate Beam 1
field 4
clear 1 1
phase/piston 2 90
clear 4 0
mult/tensor/4 4 1 2 3 xxXx
title conjugate Beam 2
field 4
clear 2 1
phase/piston 3 90
clear 4 0
mult/tensor/4 4 1 2 3 xxxX
title conjugate Beam 3
field 4
array/s 4 64 64 1
clear 4 0
mult/tensor/4 4 1 2 3 yxxX
title conjugate Beam 3, store in y-state, beam 4
field 4
pause
array/s 1 64 64 1
jones/set ar=0 br=0 cr=1 dr=0
jones/mult 1
clear 4 0
mult/tensor/4 4 1 2 3 xyxX
title conjugate Beam 3, y-state Beam 1
field 1
field 4
pause
clear 4 0
phase/piston 1 90
field 1
pause
mult/tensor/4 4 1 2 3 xYxX
title conjugate Beams 1 and 3, y-state Beam 1
field 4
```

Jump to: [Commands](#), [Theory](#)

Ex95h: Interference between a straight and tilted beam, in glass**Input: ex95h.inp**

```

c## ex95h
c
c Example 95h: Interference between a straight and tilted beam, in glass
c
c This example demonstrates the interference pattern between a beam
c headed along the z-axis with one propagating at an angle.
c
c If the crossing angle is theta, we will have an interference
c pattern with transverse period
c
c  $p = \lambda/\theta$ 
c
c and the Talbot distance gives us the axial period
c
c  $Z\text{-talbot} = 2p^2/\lambda = 2\lambda/\theta^2$ 
c
c Choosing  $\lambda = 1$  micron and  $\theta = 20$  milliradian, we have
c
c  $Z\text{-talbot} = .5$  cm
c  $p = .005$  cm
c
c The sampling should be 5 to 10 times the period to be sampled.
c  $dx = .0005$ 
c
echo/on
variab/dec/int row
mem/set/b 8
array/set 1 512
nbeam 2
units/s 1 .0005 # Set units.
units/s 2 .0005 # Set units.
nbeam 4 data
array/set 4 512 64 data
clear 4 0.
units/s 4 .0005 .015625
wavelength/set 0 1 2 # Set wavelength.
gaus/c/c 1 1 .04 8
gaus/c/c 2 1 .04 8
abr/tilt 2 200 rn=1 az=90
plot/watch ex95h_1.plt
title interference in vacuum
macro/def ex95a/o
  row = row + 1 list
  prop .015625
  clear 3 0
  add/opl 3 1 2
  copy/row 3 4 257 row
  plot/l 4 h=.1 ns=64 xrad=.02 yrad=.5
macro/end

```

Jump to: [Commands](#), [Theory](#)

```

macro ex95a/64
plot/watch ex95h_2.plt
plot/y 4 fmax=5.
wavelength/set 2 1 1.0002
zreff/se 0 0.
gaus/c/c 1 1 .04 8
gaus/c/c 2 1 .04 8
abr/tilt 2 200 rn=1 az=90
row = 0
clear 4 0
plot/watch ex95h_3.plt
title interference with index correction
macro ex95a/64
plot/watch ex95h_4.plt
plot/y 4 fmax=5.

```

Ex95i: Resonator with OPA included

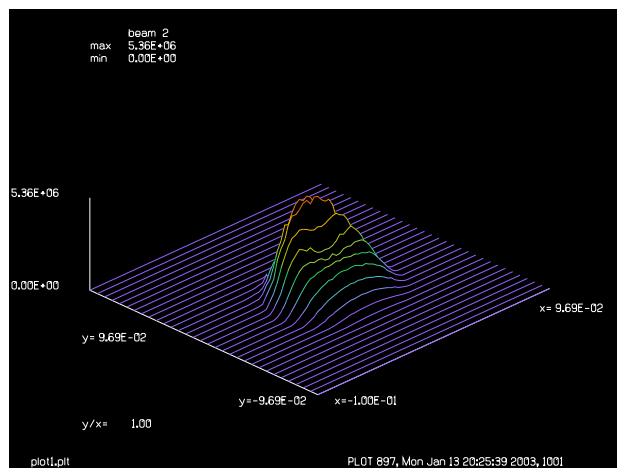


Fig. 95.17. Mode shape for OPO example.

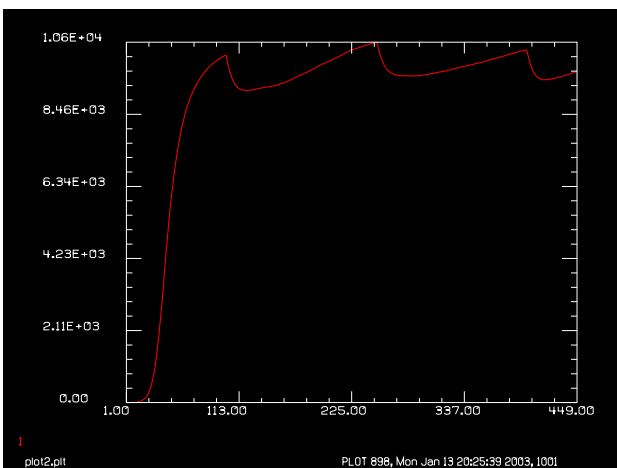


Fig. 95.18. Power vs. cycle number for OPO example.

Input: ex95i.inp

```

c## ex95i
#
# Example: resonator with OPA included
#
variab/dec/int Ntimes count
nbeam 3
lambda_p = .5
lambda_s = .75
#
# vacuum wavelength for idler may be solved by the condition that
# the sum of frequencies is zero -- identical to the sum of
# inverse vacuum wavelengths
#
# Define three beams
# pump, vacuum wavelength
# signal, vacuum wavelength

```

Jump to: [Commands](#), [Theory](#)

```

array/set 1 64 64          # Set arrays to 64 x 64
array/set 2 64 64 1        # Set arrays to 64 x 64
array/set 3 64 64          # Set arrays to 64 x 64
lambda_i = 1./(1./lambda_p - 1./lambda_s)
lambda_i=                  # idler, vacuum wavelength
n_p = 1.52
n_i = 1.50
n_o = 1.51
n_e = 1.54
wavelength/set 1 lambda_p n_p # Define wavelength of pump bea
wavelength/set 2 lambda_s n_o n_e # Define wavelength of signal
wavelength/set 3 lambda_i n_i    # Define wavelength of idler
n_s = lambda_s*(n_p/lambda_p - n_i/lambda_i)
n_s=                          # desired index for sig
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_s = 1./sqrt(x^2 + y^2)
n_s=                          # calculated ind
delta_k = n_p/lambda_p - n_s/lambda_s - n_i/lambda_i list
crystal/uniaxial/set polar 0
units/field 0 .1             # Set units
clear 2 1e2                  # Define signal
jones/set ar=0 cr=1
jones/mult 2
udata/clear                  # Clear user dat
wave4/set 0.                 # Set zstart for
pack/set 1 2 3               # Define arrays for memo
variab/set Peak1 1 peak
variab/set Peak2 2 peak
variab/set Peak3 3 peak
udata/set 1 0. Peak1 Peak2 Peak3 # First call to user data
zstep = .1                   # Set step length
Ntimes = 1./zstep
c
c Define macro for kinetic step
c
macro/def kinstep/over
    pack/in
    opo/constant zstep=zstep kappa=1e-3 nstep=Nsteps xyx
    pack/out
macro/end

macro/def opores/o
    count = count + 1
    gaus/c/c 1 1e6 .05
    clear 3 1e1
    zreff/se 1 0
    zreff/se 2 0
    zreff/se 3 0
    geodata/set 1 zwastx=0 zwasty=0 waistx=.05 waisty=.05
    geodata/set 2 zwastx=0 zwasty=0 waistx=.05 waisty=.05
    geodata/set 3 zwastx=0 zwasty=0 waistx=.05 waisty=.05

```

```
macro/run kinstep/Ntimes
mult/scalar 2 0.9
mirror/flat 2
clap/c/n 2 .05
rescale/shift 2 0 -3.5e-2
prop 1 2
mirror/flat 2
clap/c/n 2 .05
plot/w plot1.plt
plot/l 2
variab/set Energy 2 energy
udata/set count count Energy
plot/w plot2.plt
plot/udata
macro/end

Nsteps = 495
macro opores/1000
```

References

1. A. Yariv, *Introduction to Optical Electronics*, Holt, Rinehart, and Winston, New York, (1976).
2. A. Yariv, *Quantum Electronics*, 2nd Ed., John Wiley & Sons, (1975).
3. Y. R. Shen, *The Principles of Nonlinear Optics*, John Wiley & Sons, New York (1984).

Ex96: Circular propagator

Table. 96.1. Table of Ex96 examples

Ex96a: Example of one-dimensional, circular array	1
Ex96b: Compare diffraction propagation of square and circular beams	3
Ex96c: General propagation of circular beams	6

This example illustrates the circularly symmetrical propagator (see GLAD Theory Manual, Section 5.5). Arrays of form $N \times 1$ may be propagated many times faster than the conventional two-dimensional propagators, however the accuracy is somewhat less, so care must be exercised in its use. Example 96a illustrates definition of an aperture and Zernike form of spherical in a circular array, transfer of the circular data to the center row of the square array, and the use of the SPIN command to form a rotationally symmetric distribution in the square array. For a 1024×1024 array, this procedure fills the square array with the high order Zernike polynomial many times faster than direct computation in the square array, since the Zernike coefficients must be calculated for only about 1,000 points in the circular array versus about 1,000,000 points in the square array. The interpolation procedure in the SPIN command does require some time, but is still much faster than filling the full square array with Zernike terms.

Ex96a: Example of one-dimensional, circular array

Figure 96.1 shows the wavefront for Zernike spherical aberration of order 14. The wavefronts for both the circular array and the square array are plotted on the same graph in Fig. 96.1. The plots overlap precisely indicating that the spin command has rendered the aberration accurately in the square array with no observable error from the interpolations process.

The most accurate circular propagation case consists of propagation to the far-field from a pupil with well-defined boundaries. This is more accurate than the procedure which is usual in GLAD for small defocus, consisting of propagation to the exact far-field followed by an additional propagation to make the small focus adjustment.

Input: ex96a.inp

```
c## ex96a
c
c Example: Example of one-dimensional, circular array
c
c N x 1 arrays may be defined with attribute "circular"
c
c A square array of size N x N may be generated from
c the N x 1 array with the SPIN command. The SPIN command
c fills the square array by rotating the diagonal and
c using linear interpolation.
c
c We can go from square to circular with the COPY/ROW
c command and from circular to square with a combination
c of COPY/ROW and SPIN
c
mem/set/b 10
variab/dec/int nline ncxs Nrad
```

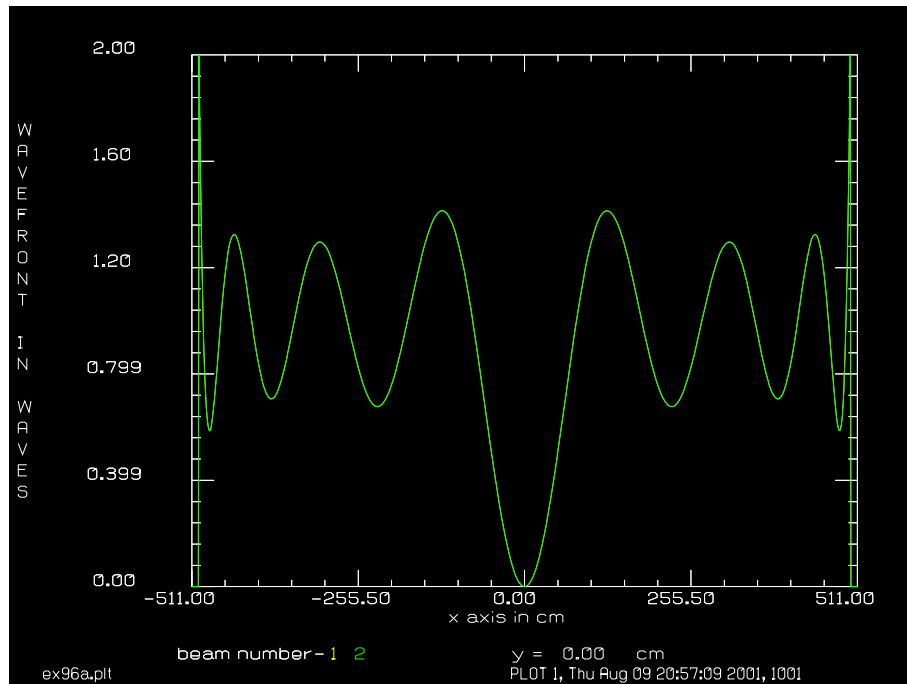


Fig. 96.1. Wavefront for Zernike spherical aberration of order 14. The wavefronts for both the circular array and the square array are plotted. These plots overlap precisely indicating that the spin command has rendered the aberration accurately in the square array with no observable error from the interpolations process.

```

nline = 1024
ncxs = nline/2 + 1
array/s 1 nline                # square array of N x N
nbeam 2 nline 1 circular      # circular array of N x 1

Nrad = 14
Waves = 1.
c
c Check time to put Zernike polynomials in square array
c

time/i
Clap = 500
clap/c/c 1 Clap
abr/zern/rad 1 Nrad Waves      # Zernike polynomials
time

c
c Check time to generate Zernike polynomials in N x 1 array
c
time/i
clap/c/c 2 Clap
time/i
abr/zern/rad 2 Nrad Waves      # Zernike polynomials
copy/row 2 1 1 ncxs           # copy to diagonal of N x N array
time
c

```

Jump to: [Commands](#), [Theory](#)

```

c Check time to fill square array from diagonal by SPIN command
c
time/i
spin 1
time
plot/w ex96a.plt
plot/x/w fi=1 la=2

```

Ex96b: Compare diffraction propagation of square and circular beams

Example 96b shows comparison of a diffraction pattern of Fresnel number 12 calculated by full square array of 1024 x 1024 and by a circular array of 1024 x 1. Figure 96.2 superimposes the plots for the square array and the circular array. The outline of the geometric aperture, with no diffraction effects is also superimposed on this figure. The geometric distribution is calculated from the magnification at the particular defocus position. We can see that both methods of calculation conform well to the correct mean value as determined by the geometric profile. Figure 96.2 also displays the geometric outline at 25% of the geometric intensity. The diffraction distribution should be exactly 25% of the geometric profile at the edge of the geometric profile. Figure 96.2 shows that both calculations conform well to the 25% point. For even Fresnel numbers the center of the distribution should be exactly zero. In fact, both methods of calculation result a center point which is slightly higher than zero. This is due to the fact that both methods result in the aperture having an effective radius slightly less than the value defined by the CLAP command. The slight undersizing results from the superposition of a circular aperture onto a square array of points. The numerical aperture shape is actually defined by the shape of the points in the square array of points which have radius less than the specified aperture value. Figure 96.3 illustrates this numerical effect for a small array. The actual numerical boundary has an effective radius slightly less than the clipping radius. The effect is essentially the same for the circular array as for the square array because the circular propagation algorithm has, in a virtual sense, the same mapping into the square array of points when calculating the sum along the y-directions. The center point may be driven to a nearly perfect point by slightly increasing the radius of the clipping aperture. Figure 96.2 shows that the circular propagation algorithm calculates the diffraction peaks slightly below the value for the square array.

Input: ex96b.inp

```

c## ex96b
c
c Example: Compare diffraction propagation of square and circular
c arrays
c
c The circular array is of form N x 1 and allows rapid calculation
c of diffraction effects of circular arrays. However the circular
c diffraction algorithms are not as accurate as the algorithms
c for square arrays. For circular beams propagated from the pupil
c to the far-field, the accuracy of the circular propagation
c algorithms is almost as good as for the full square array.
c
c In this example, we calculate the diffraction pattern for
c Fresnel Number = 12. It is more accurate with the circular
c beam to do this by adding defocus than it is to propagate

```

Jump to: [Commands](#), [Theory](#)

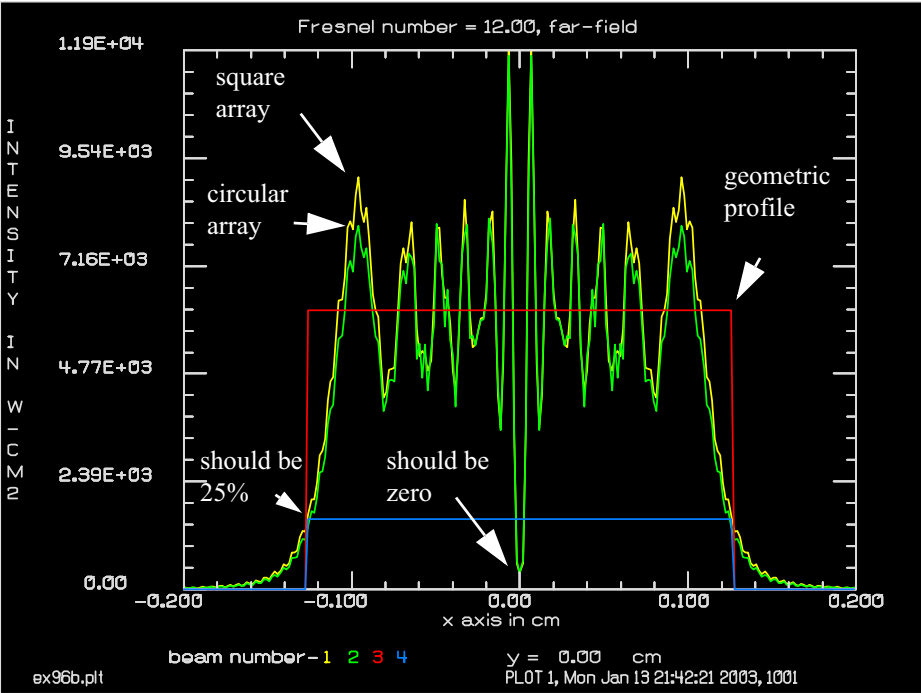


Fig. 96.2. Comparison of diffraction patterns for full square array and circular array propagation. The geometric profile is also shown as well as a 25% profile. The diffraction pattern should pass through the geometric profile at the 25% points. For the even Fresnel number of 12 the center point should be zero. The small error is due to the numerical aperture radius being slightly less than the desired value.

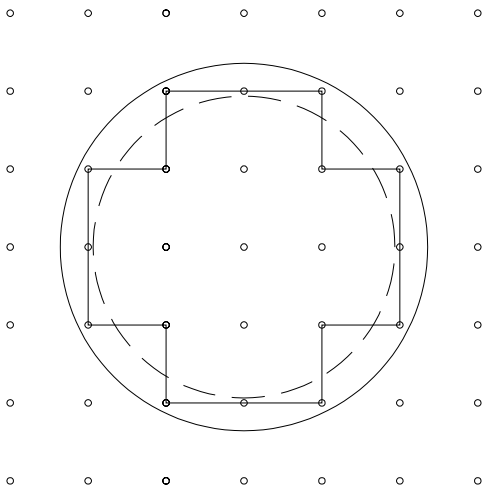


Fig. 96.3. A circular boundary (solid) in numerical implementation results in an irregular ragged boundary with smaller effective size (dashed).

```
c by the defocus amount from the center of the far-field.
c
c Beams
c -----
c 1          Square array of 1024 x 1024
```

```

c 2          circular array of 1024 x 1
c 3          circular array of 1024 x 1 to show geometric
c           outline
c 3          circular array of 1024 x 1 to show geometric
c           outline at 25% level
mem/set/b 10
variab/dec/int nline ncxs
nline = 1024
ncxs = nline/2 + 1
array/s 1 nline          # make N x N square array
units/s 1 64/nline 64/nline
nbeam 2 nline 1 circular # make N x 1 circular array
units/s 2 64/nline 64/nline

FocalLength = 100          # focal length of the lens
Lambda = 10.6E-4          # wavelength
Rnormal = 10              # radius of the aperture
FresnelNumber = 12        # desired Fresnel number
Waves = FresnelNumber/2.  # waves of defocus aberration
                          # at the edge of the aperture
                          # DeltaF is the defocus from
                          # the center of the far-field
DeltaF = -2.*Waves*Lambda*FocalLength^2/Rnormal^2
DeltaF=

clap/c/c 2 Rnormal        # clear aperture for circular beam
abr/foc 2 Waves           # defocus aberration

time/i
copy/row 2 1 1 ncxs      # copy to center row of square array
spin 1                   # spin to expand into square array

time                      # Spin time
lens 1 100
lens 2 100
time/i
dist 100 1
time                      # time for propagation of square array
time/i
dist 100 2
time                      # time for propagation of circular array
#
# Make beams 3 and 4 to show geometric outline aperture at
# 100% and 25% levels
#
nbeam 4 4*nline 1 circular
variab/set Units 1 units
units/s 3 Units
clear 3 [(FocalLength/DeltaF)^2] # set peak to magnification squared
clap/c/c 3 [Rnormal*abs(DeltaF)/FocalLength]
copy 3 4
#
# Set beam 4 to 25% of geometric aperture to show that the
# diffraction patterns are at 0.25 at the geometric boundary

```

Jump to: [Commands](#), [Theory](#)

```
#
mult 4 .25
title Fresnel number = @FresnelNumber, far-field
plot/w ex96b.plt
plot/x/i fi=1 la=4 le=-.2 ri=.2
```

Ex96c: General propagation of circular beams

Example 96c illustrates propagation through the geometry illustrated in Fig. 96.4. A circular pupil is defined for both a square array and circular $N \times 1$ array just in front of the focusing lens. The distribution at the focal point of the lens is calculated by a Fourier transform of the pupil distribution. Figure 96.5 shows the superposition of the profiles from the square and circular arrays. The agreement is excellent. A field lens, as illustrated in Fig. 96.4, causes the pupil to be imaged at a distance two focal lengths away. This is effectively a condition of zero effective propagation distance, as is always true of propagation to an image of the initial plane. For the square array the zero propagation length case is accurate to the level of machine precision. The zero propagation length case is especially difficult for circular propagation because of the greatly extended Airy-like pattern at the intermediate focal point (the frequency domain). To minimize the artifacts produced the slight smoothing of the original distribution was implemented to limit the extent of the distribution in the far-field (frequency domain). The zero propagation case shows a slight rounding of the overall profile as well as some higher frequency artifacts in the center. Fig. 96.5 shows propagation of length equal to the focal length beyond the image pupil. Both methods of calculation agree well but not perfectly in this case.

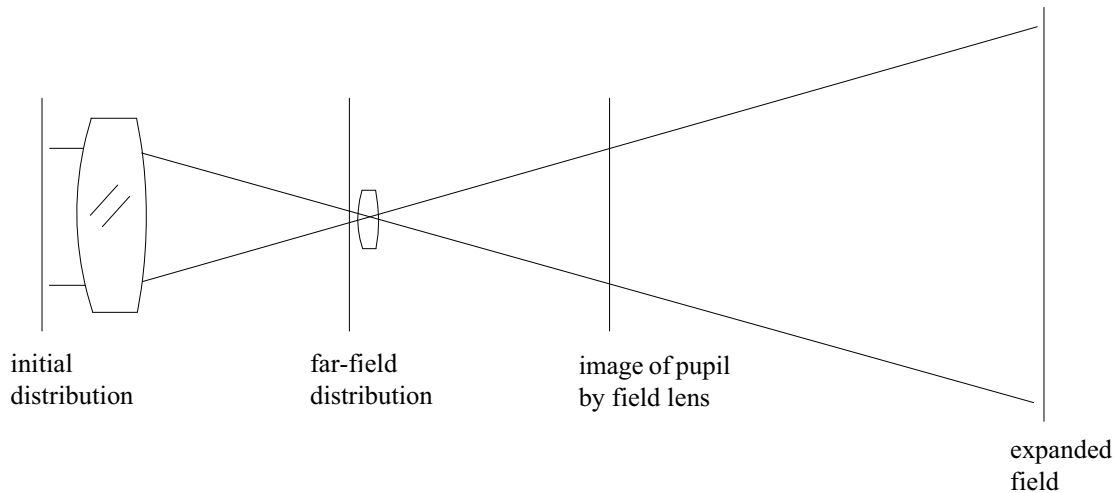


Fig. 96.4. In Example 96c, an initial distribution (Fig. 96.5) is focused to the far-field (Fig. 96.6) and then relayed to an image of the pupil formed by the field lens (Fig. 96.7). The beam subsequently expands displaying near-field diffraction (Fig. 96.8).

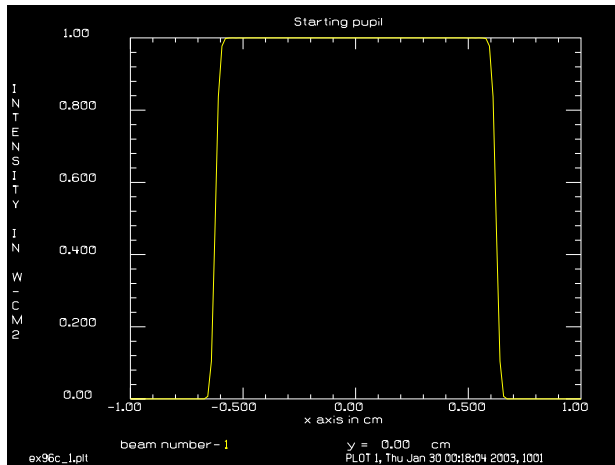


Fig. 96.5. Initial distribution with slight smoothing.

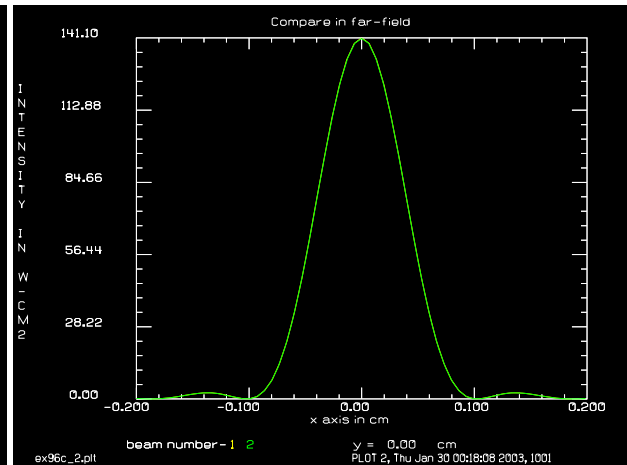


Fig. 96.6. Far-field showing square and circular beams superimposed.

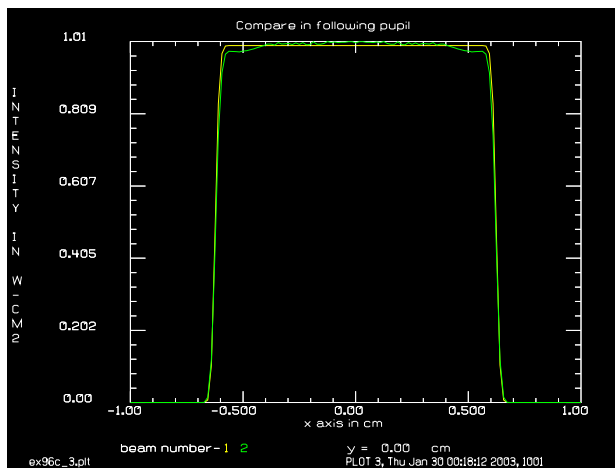


Fig. 96.7. Relayed pupil (zero propagation distance). Circular propagator (green) shows slight droop.

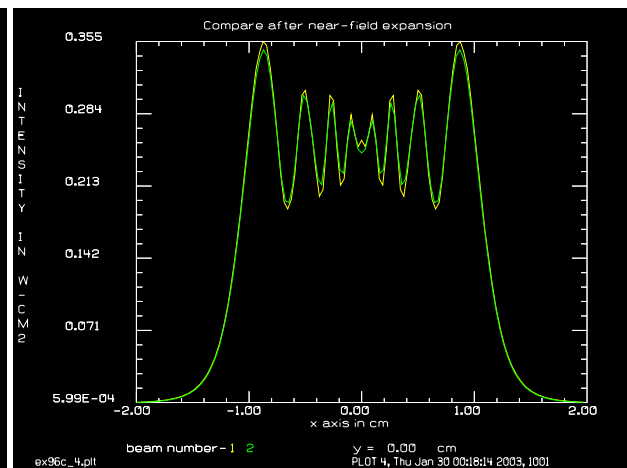


Fig. 96.8. Expanded beam showing near-field diffraction. Square beam (yellow) has higher peaks.

Input: `ex96c.inp`

```
c## ex96c
#
# Example: General propagation of circular beams
#
# This example illustrates propagation under various conditions
# of near-field and far-field.
#
# Propagation of a square N x N array and an equivalent circular
# N x 1 array are compared.
#
mem/set/b 10
variab/dec/int nline ncxs
nline = 1024
ncxs = nline/2 + 1
```

Jump to: [Commands](#), [Theory](#)

```

array/s 1 nline
Units = 16/1024
units/s 1 Units
nbeam 2 nline 1 circular
units/s 2 Units
Clap = .625

#
# It is fastest to set the aperture with the circular array
#

time/i
clap/c/c 2 Clap           # set clear aperture
conv/gaus 2 .025          # smooth pupil to remove
                           # high spatial frequencies
copy/row 2 1 1 ncxs       # copy to square array
time

time/i
spin 1                    # expand the diagonal to
time                       # make a circular distribution
                           # in the square array

title Starting pupil
plot/w ex96c_1.plt
plot/x/i le=-1 ri=1
nbeam 4 nline 1 circular
lens 1 100                 # implement lenses of f = 100 cm
lens 2 100
#
# Compare times for square and circular arrays
#
time/i
dist 100 1
time
time/i
dist 100 2
time
plot/w ex96c_2.plt
title Compare in far-field
plot/x/i fi=1 la=2 le=-.2 ri=.2
lens 1 50                   # field lenses to image initial
lens 2 50                   # pupil to next plane
dist 100 1
dist 100 2
plot/w ex96c_3.plt
title Compare in following pupil
plot/x/i fi=1 la=2 le=-1 ri=1
time/i
dist 100 1
time
time/i
dist 100 2
time
plot/w ex96c_4.plt

```

Jump to: [Commands](#), [Theory](#)


```
title Compare after near-field expansion
plot/x/i fi=1 la=2 le=-2 ri=2
end
```


Ex97: Volume holograms and GRIN lens arrays

Table. 97.1. Table of Ex97 examples

Ex97a: Volume hologram showing mode conversion versus propagation length	1
Ex97b: Propagation through three-beam interference pattern	5
Ex97c: Propagation through four-beam interference pattern	8

GLAD form the interference patterns due to two or more beams over a three-dimensional region giving the (x,y,z) interference pattern. This interference pattern may then be converted to phase screens allowing modeling of volume hologram. Example 97a illustrates a volume hologram created by two-beam interference of two collimated beams with a relative angle between them to form a three-dimensional interference pattern. This intensity distribution is converted to phase modulation simulating a modulated gradient index distribution similar to that which would be created in photoresist of other holographic recording material. Once formed, the volume hologram will convert light incident from one of the original construction angles into light exiting along the other of the original construction angles. The mode conversion takes place gradually as the beam propagates through the volume hologram.

Example 97a shows the gradual conversion and also illustrated the relatively high efficiency of the conversion process. Examples 97b and 97c illustrate three- and four-beam interference respectively. These beams create an approximation to an array of gradient index (GRIN) lenses and illustrate some focusing effects similar to that which would be achieved with an array of true GRIN elements.

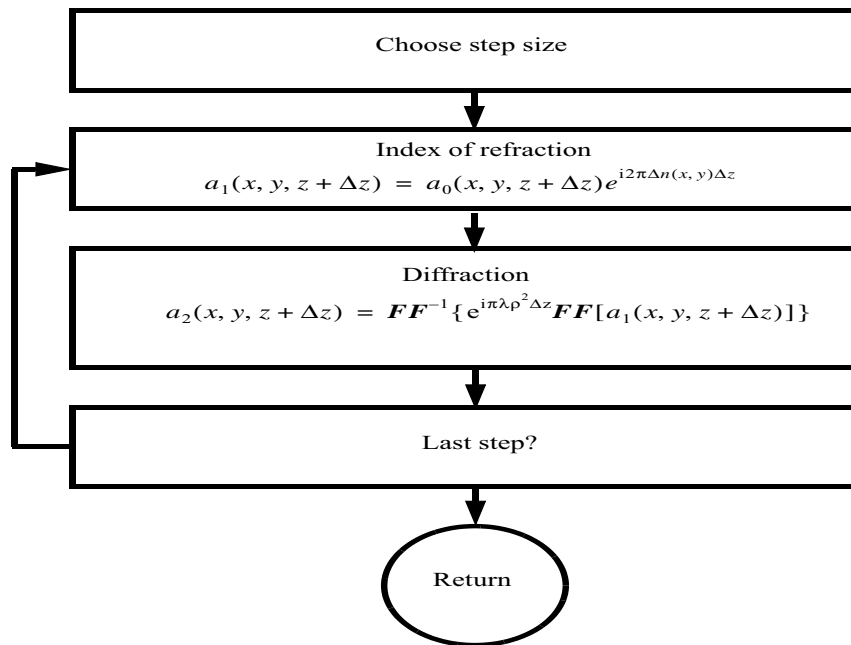


Fig. 97.1. Flow chart for split-step method of treating diffraction and the refractive index function $\Delta n(x, y)$. For a small step Δz , the effect of the refractive index is implemented as a phase screen of the form $e^{i2\pi\Delta n(x, y)\Delta z}$ to change the initial complex amplitude $a_0(x, y, z + \Delta z)$ into the intermediate result $a_1(x, y, z + \Delta z)$. A diffraction step is applied to the intermediate result, implemented by FFT methods, to create the full split-step procedure.

Ex97a: Volume hologram showing mode conversion versus propagation length

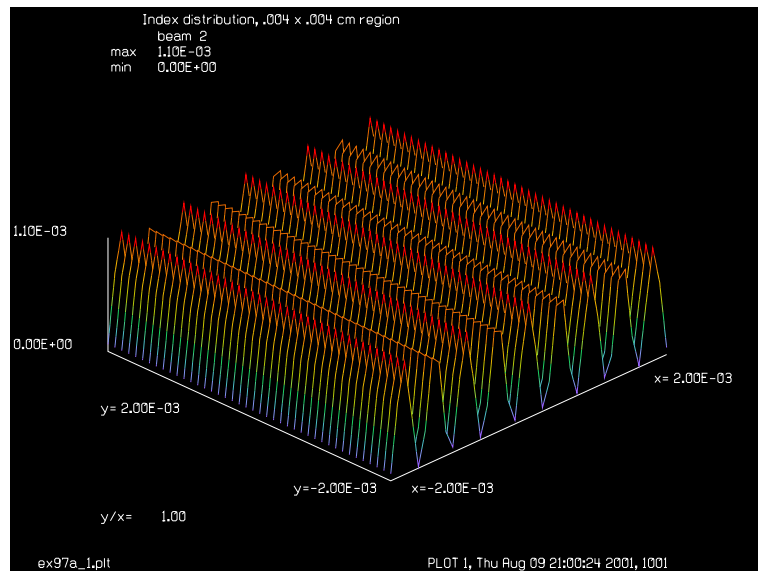


Fig. 97.2. Close-up view of interference pattern created by the interference of two beams. These particular beams are collimated and free of aberration but noncollimated and aberrated beams may be used to form imperfect holograms. This interference pattern extends along the z-axis in identical form and forms the basis of gradient index structure, i.e., a volume hologram.

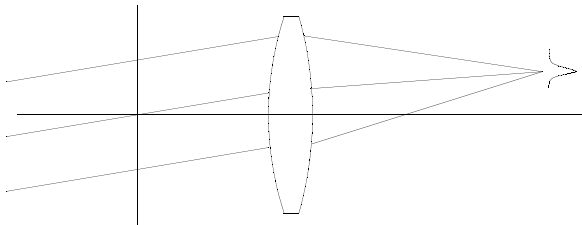


Fig. 97.3a. Configuration with no hologram.

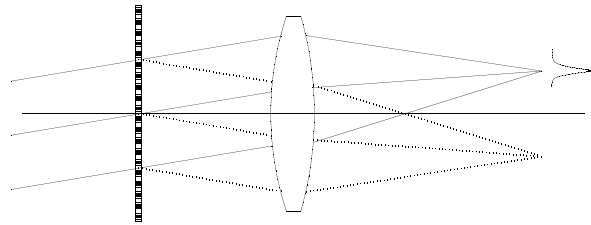


Fig. 97.3b. Thin section of volume hologram. Negligible conversion.

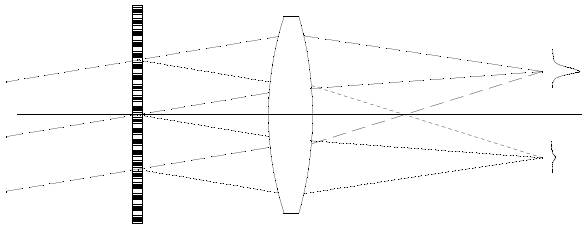


Fig. 97.3c. Thicker volume hologram. Far-field shows weak side lobe.

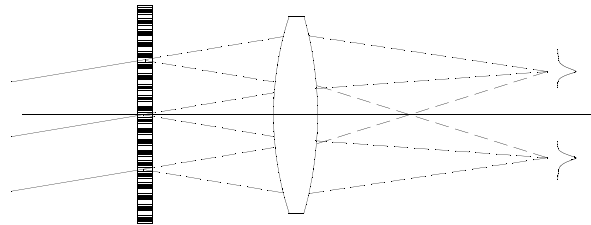


Fig. 97.3d. Further thickness leads to equal power in both orders.

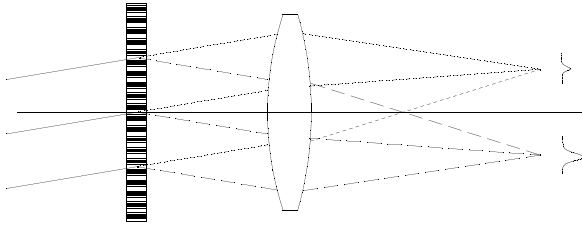


Fig. 97.3e. Sidelobe dominates.

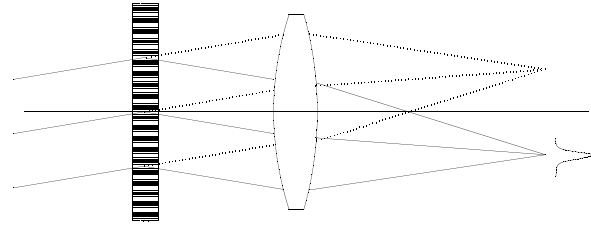


Fig. 97.3f. Complete conversion to sidelobe.

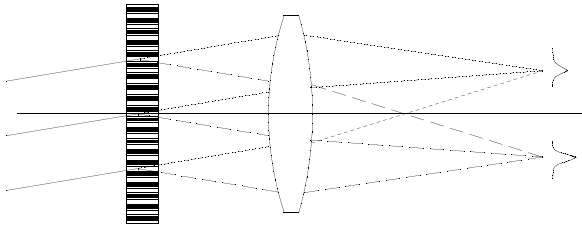


Fig. 97.3g. Further thickness increase couples back into zero order mode.

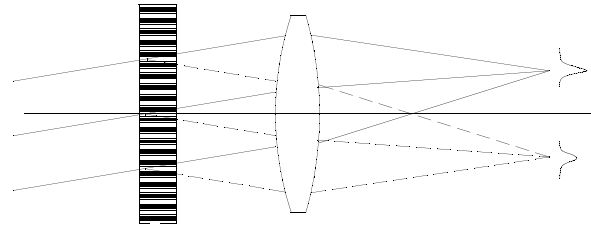


Fig. 97.3h. With more thickness energy continues to return to zero order mode.

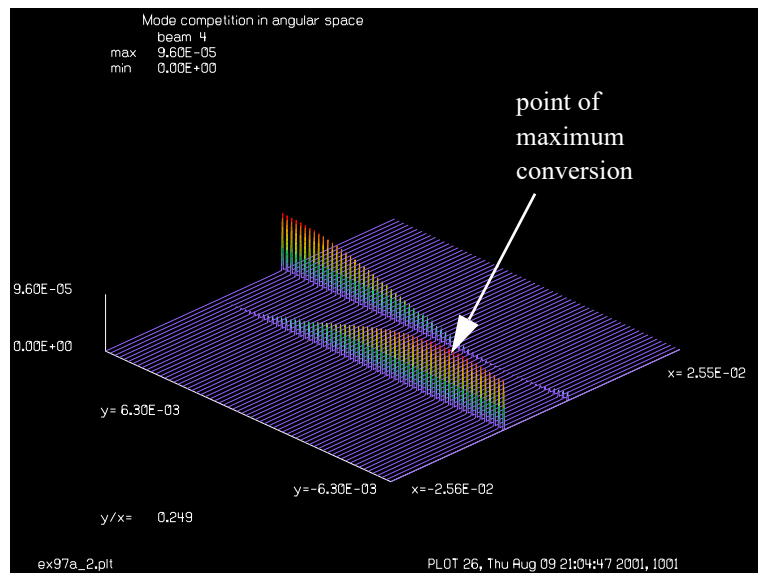


Fig. 97.4. Conversion of one optical mode to another by volume hologram. This plot shows the mode structure in angular space—the image as indicated in Fig. 97.3a. The incident beam is collimated and is shown in angular space by an intensity spike to the right of center. This beam has been aligned to the same angle as one of the original construction beams. As the light propagates through the volume hologram, the light is gradually converted to a mode aligned with the other construction beam as indicated by the intensity spike to the left of center. Propagating beyond the point of maximum conversion causes light to go back into the original mode.

Figs. 97.3a-97.3h correspond to different thickness of the hologram.

Input: `ex97a.inp`

c## `ex97a`

Jump to: [Commands](#), [Theory](#)

```

c
c Example 97a: Volume hologram and two-beam interference
c
c Command deck to calculate two-beam interference representing
c a volume hologram. The example illustrates exchange of energy
c from light injected along one of the original construction
c angles into the mode corresponding to the other of the original
c construction angles.
c
variab/dec/int iplt count pass row Nline Nsteps
Nline = 512
Nsteps = 128
Units = .0001
Clap=2*.0256/sqrt(2.)
Tilt=40.
Dist = 5e-4
Lambda0 = .85e-4
Index = 1.5
wavelength/set 1 Lambda0*1e4 Index
array/set 0 Nline          # set array size for both beams
units/s 0 Units           # set units/s for both beams
nbeam 3
array/attribute 2 data
array/set 4 Nline Nsteps data # beam 4, history of index profile
units/set 4 Units Dist
c
c Two beam interference
c
abr/tilt 1 Tilt rn=Clap az=-90
abr/tilt 2 Tilt rn=Clap az= 90
add/coh/con 2 1
peak/norm 2 .0011
title Index distribution, .004 x .004 cm region
plot/watch ex97a_1.plt
plot/1 2 ns=512 xrad=.002
plot/watch ex97a_2.plt
set/density 64
title Two beam interference
Alpha = -2.*pi*Index/Lambda0*Dist
int2phase 2 Alpha
gaus/c/c 1 1 .02 10
abr/tilt 1 Tilt rn=Clap az=-90
beams/off 3
beams/off 4
clear 4 0.
macro/def step/o
    pass = pass + 1
    mult/beam 1 2
    prop Dist
    z = z + Dist
    title pass = @pass, z = @z
    beams/off 1
    array/reset 3
    units/beam 3 1      # copy units from Beam 1

```

Jump to: [Commands](#), [Theory](#)

```

beams/on 3
copy/con 1 3
zreff/se 3 0.
global/def 3 0 0 0
lens 3 100
prop 100
beams/off 3
beams/on 1
row = row + 1
copy/row 3 4 [Nline/2 + 1] row      # store mode in history array
count = count + 1
if [count==10] then
    bell
    plot/watch ex97a_2.plt
    plot/l 4 ns=128 h=.2
    plot/watch ex97a_3.plt
    plot/x/i 3
    count=0
endif
macro/end
macro step/Nsteps
title Mode competition in angular space
plot/watch ex97a_2.plt
plot/l 4 ns=128 h=.2

```

Ex97b: Propagation through three-beam interference pattern

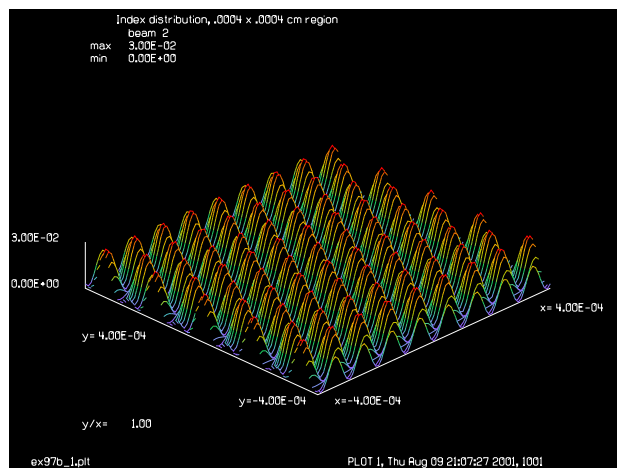


Fig. 97.5. Index distribution due to interference pattern from three beams.

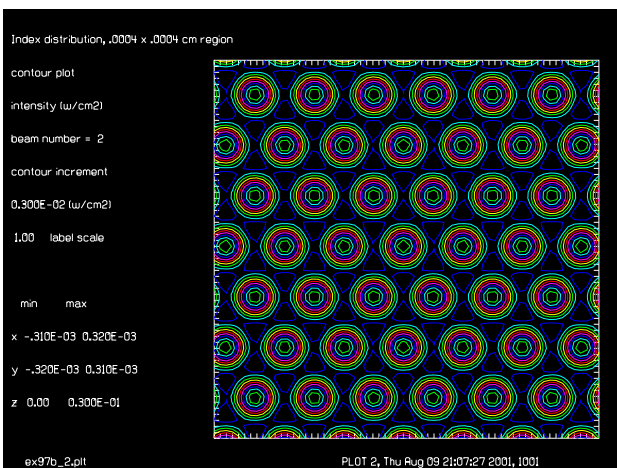


Fig. 97.6. Contour plot showing lens-like structure (three beams).

Input: ex97b.inp

```

c## ex97b
c
c Example 97b: Three-beam interference

```

Jump to: [Commands](#), [Theory](#)

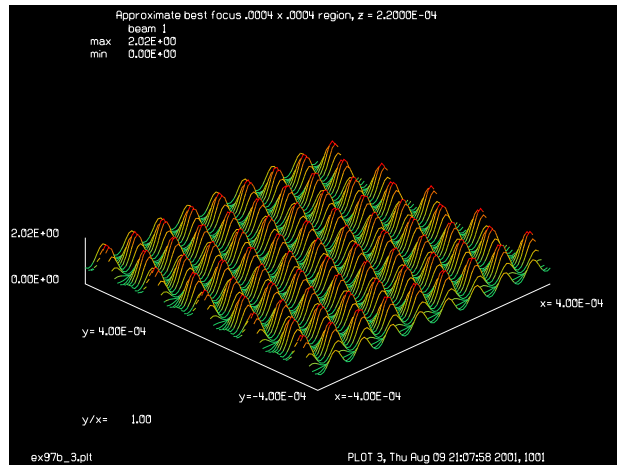


Fig. 97.7. Irradiance at best focus (three beams).

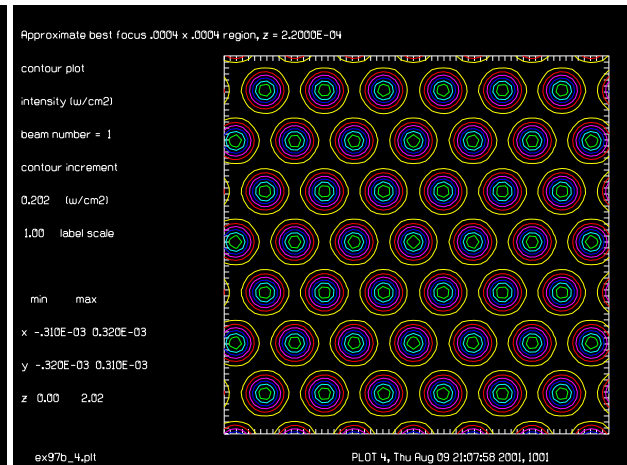


Fig. 97.8. Contour plot at best focus (three beams).

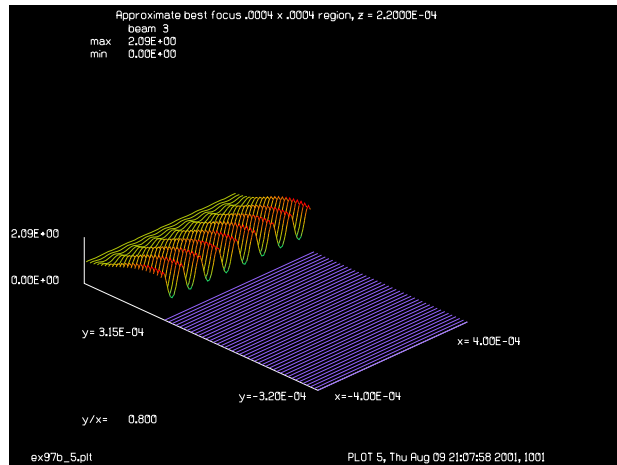


Fig. 97.9. Through-focus display at best focus (three beams).

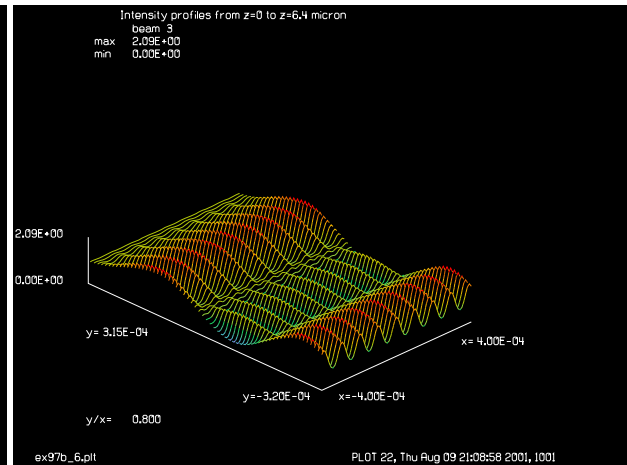


Fig. 97.10. Through-focus display beyond best focus, showing Talbot imaging (three beams).

```

c
c  Command deck to calculate three-beam interference
c
variab/dec/int iplt Nline pass count
Nline = 512
Lambda0 = .55e-4
Index = 1.5
wavelength/set 1 Lambda0/1e-4 Index          # select wavelength
mem/set/b 8
c
c Beams
c -----
c Beam 1      propagating beam
c Beam 2      phase screen
c Beam 3      history array
c
nbeam 3
array/set 1 Nline          # set array size

```

Jump to: [Commands](#), [Theory](#)


```

array/set 2 Nline data      # set array size
array/set 3 Nline 128 data  # set array size
units/s 0 0.00001          # set units for both beams
c
c  Three beam interference
c
Rn = .00025
Ewav = 1.5*8./7.
abr/tilt 1 Ewav rn=Rn az=-60.
abr/tilt 2 Ewav rn=Rn az=60.
add/coh/con 2 1
clear 1 1
abr/tilt 1 Ewav rn=Rn az=180
add/coh/con 2 1
transpose 2
gaus/c/c 1 1 .0020 10
peak/norm 2 .03
title Index distribution, .0004 x .0004 cm region
plot/watch ex97b_1.plt
plot/1 2 ns=512 h=.2 xrad=.0004
set/window/abs -.0004 .0004 -.0004 .0004
set/density 64
plot/watch ex97b_2.plt
plot/c 2 ilab=0
Dist = .05e-4
Alpha = 2.*pi*Index*Dist/Lambda0
int2phase 2 Alpha
set/density 64
title Three beam interference
plot/watch ex97b_3.plt
clear 3 0.
units/s 3 .00001 Dist
macro/def step1/o
    pass = pass + 1
    mult/beam 1 2
    prop Dist
    row = row + 1
    copy/row 1 3 [Nline/2+1] row      # store mode in history array
    z = z + Dist
    title pass = @pass, z = @z
macro/end
macro/def step2/o
    pass = pass + 1
    prop Dist
    z = z + Dist
    title pass = @pass, z = @z
    row = row + 1
    copy/row 1 3 [Nline/2+1] row      # store mode in history array
    count = count + 1
    if [count==10] then
        bell
        plot/watch ex97b_6.plt
        plot/1 3 ns=64 h=.2 xrad=.0004 yrad=64*Dist
        plot/watch ex97b_7.plt

```

Jump to: [Commands](#), [Theory](#)

```

        plot/x/i 1
        count=0
    endif
macro/end
c
c Propagation in the HOE
c
macro step1/40
c
c Best focus is about 2.2 micron from front of the HOE
c
str 1
c
c Propagation to approximate best focus
c
macro step2/4
title Approximate best focus .0004 x .0004 region, z = @z
plot/watch ex97b_3.plt
plot/l 1 ns=64 h=.2 xrad=.0004
plot/watch ex97b_4.plt
plot/c 1 ilab=0
plot/watch ex97b_5.plt
plot/l 3 ns=64 h=.2 xrad=.0004 yrad=64*Dist
c
c Some more propagation
c
macro step2/84
title Intensity profiles from z=0 to z=6.4 micron
plot/watch ex97b_6.plt
plot/l 3 ns=64 h=.2 xrad=.0004 yrad=64*Dist

```

Ex97c: Propagation through four-beam interference pattern

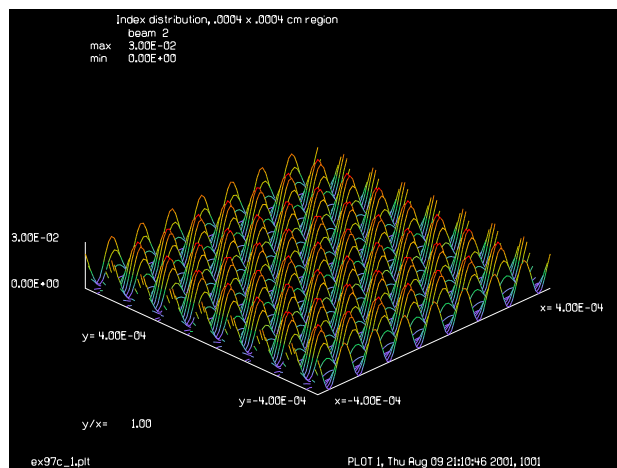


Fig. 97.11. Index distribution due to interference pattern from four beams.

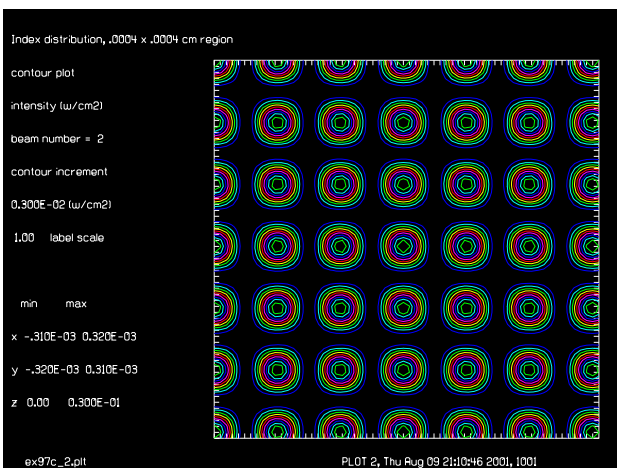


Fig. 97.12. Contour plot showing lenslike structure (four beams).

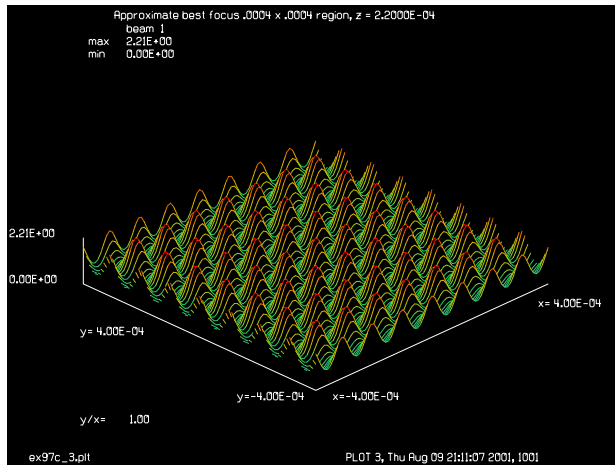


Fig. 97.13. Irradiance at best focus (four beams).

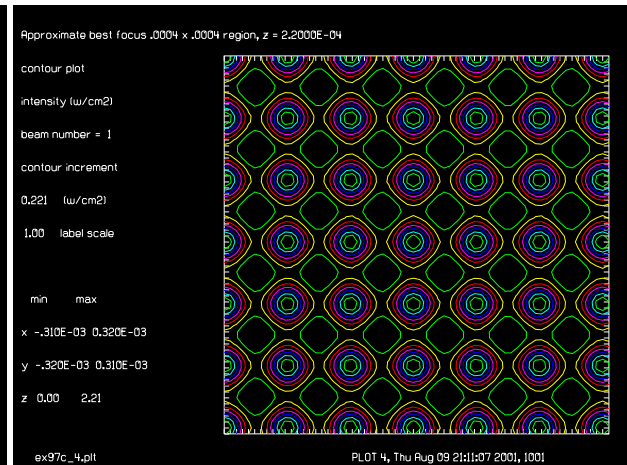


Fig. 97.14. Contour plot at best focus (four beams).

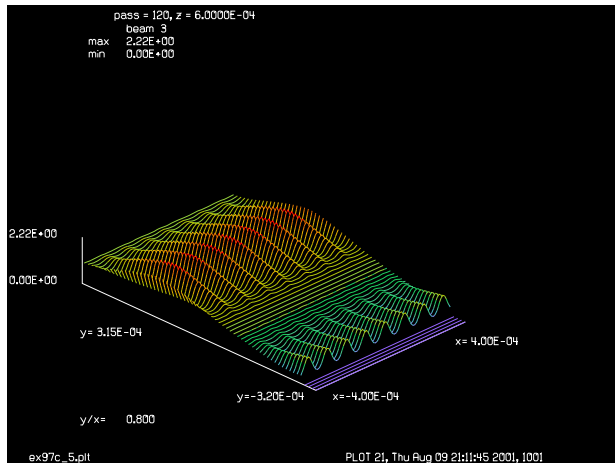


Fig. 97.15. Through-focus display at best focus (four beams).

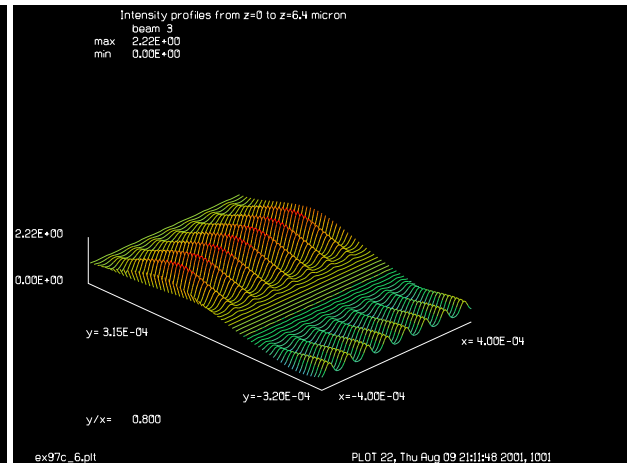


Fig. 97.16. Through-focus display beyond best focus, showing Talbot imaging (four beams).

Input: `ex97c.inp`

```

c## ex97c
c
c Example 97c: Four-beam interference
c
c Command deck to calculate four-beam interference
c
variab/dec/int iplt Nline pass count
Nline = 512
Lambda0 = .55e-4
Index = 1.5
wavelength/set 1 Lambda0/1e-4 Index      # select wavelength
mem/set/b 8
c
c Beams
c -----
c Beam 1      propagating beam

```

Jump to: [Commands](#), [Theory](#)

```

c Beam 2      phase screen
c Beam 3      history array
c
nbeam 3
array/set 1 Nline          # set array size
array/set 2 Nline data     # set array size
array/set 3 Nline 128 data # set array size
units/s 0 0.00001         # set units for both beams
c
c Four beam interference
c
Rn = .00025
Ewav = 1.5*8./7.
abr/tilt 1 Ewav rn=Rn az=45.
abr/tilt 2 Ewav rn=Rn az=135.
add/coh/con 2 1
clear 1 1
abr/tilt 1 Ewav rn=Rn az=225
add/coh/con 2 1
clear 1 1
abr/tilt 1 Ewav rn=Rn az=315
add/coh/con 2 1
gaus/c/c 1 1 .0020 10
peak/norm 2 .03
title Index distribution, .0004 x .0004 cm region
plot/watch ex97c_1.plt
plot/1 2 ns=512 h=.2 xrad=.0004
set/window/abs -.0004 .0004 -.0004 .0004
set/density 64
plot/watch ex97c_2.plt
plot/c 2 ilab=0
Dist = .05e-4
Alpha = 2.*pi*Index*Dist/Lambda0
int2phase 2 Alpha
set/density 64
title Three beam interference
plot/watch ex97c_3.plt
clear 3 0.
units/s 3 .00001 Dist
macro/def step1/o
    pass = pass + 1
    mult/beam 1 2
    prop Dist
    row = row + 1
    copy/row 1 3 [Nline/2+1] row    # store mode in history array
    z = z + Dist
    title pass = @pass, z = @z
macro/end
macro/def step2/o
    pass = pass + 1
    prop Dist
    z = z + Dist
    title pass = @pass, z = @z
    row = row + 1

```

Jump to: [Commands](#), [Theory](#)

```

copy/row 1 3 [Nline/2+1] row      # store mode in history array
count = count + 1
if [count==10] then
    bell
    plot/watch ex97c_7.plt
    plot/x/i 1
    plot/watch ex97c_6.plt
    plot/1 3 ns=64 h=.2 xrad=.0004 yrad=64*Dist
    count=0
endif
macro/end
c
c Propagation in the HOE
c
macro step1/40
c
c Best focus is about at 2.2 microns from the front of the HOE
c
str 1
c
c Propagation to approximate best focus
c
macro step2/4
title Approximate best focus .0004 x .0004 region, z = @z
plot/watch ex97c_3.plt
plot/1 1 ns=64 h=.2 xrad=.0004
plot/watch ex97c_4.plt
plot/c 1 ilab=0
plot/watch ex97c_5.plt
plot/1 3 ns=64 h=.2 xrad=.0004 yrad=64*Dist
c
c Some more propagation
c
macro step2/84
title Intensity profiles from z=0 to z=6.4 micron
plot/watch ex97c_6.plt
plot/1 3 ns=64 h=.2 xrad=.0004 yrad=64*Dist

```


Ex98: Design of far-field distribution by simulated annealing

Table. 98.1. Table of Ex98 examples

Ex98a: Initialize array	3
Ex98b: Run this example to convergence, about 100,000 times	7
Ex98c: Final run to make phase plot	11

This example describes a surface-based form of simulated annealing which is extraordinarily powerful but also extraordinarily slow. Simulated annealing is, in principle, capable of achieving the optimum solution for problems with many thousands of degrees of freedom. The method used here achieves low scatter by guaranteeing strictly continuous- surface phase plates. The number of possible continuous-surface solutions is much smaller than the number of discontinuous solutions based on pixels, so the dimensionality of the problem is greatly reduced leading to more practical calculation times because of the smaller size of the possible solution space.

In surface-based simulated annealing, we started from an initially aberration-free pupil distribution. A smoothed random wavefront of weak magnitude was constructed and added to the pupil, the far-field irradiance was calculated by diffraction propagation using fast Fourier transform algorithms, and the far-field performance was compared with the desired far-field irradiance envelope. Smoothed random surface perturbations were constructed by taking a two-dimensional field of real random numbers and smoothing and scaling this distribution to obtain the specified variance and autocorrelation width. The result is the perturbation distribution for the surface of the phase plate. These perturbation aberration surfaces are made unique by choosing a different random seed for each cycle.

We could have used any irradiance envelope and we could have added other constraints such as control of spatial frequencies or other design parameters. For the particular calculations presented in this paper, the target shape was taken to be

$$\text{target shape} = t(r) = e^{-2\left(\frac{r}{r_0}\right)^M} \quad (98.1)$$

where M is the supergaussian order, r is the transverse radius in the far-field, and r_0 is the characteristic radius of the far-field distribution. An advantage of simulated annealing is that any cost function may be used. To test the feasibility of the simulated annealing we adopted a simple cost function of the form,

$$\text{cost function } S = \int_0^{\infty} (f(r) - t(r))^2 2\pi r dr \quad (98.2)$$

where $f(r)$ is the azimuthal average profile of the far-field with a particular phase plate. We used the azimuthal average profile to provide noise averaging of the quasi-speckle pattern in the far-field. Various perturbations of the phase plate are tried. If the cost function is lower, the perturbation is accepted and the modified phase plate becomes the basis of further trials. If the cost function is higher than the previous cycle, the trial perturbation will be accepted if a random number X , which is uniformly distributed between 0 and 1, is less than the simulated annealing probability value p , as defined,

$$P = e^{-\frac{\Delta S}{T}} \quad (98.3)$$

where ΔS is the change in the cost function due to the perturbation and T is the parameter known as the simulated annealing temperature. We may consider ΔS to be the temperature of the process and T to be the controlling temperature, similar to the temperature of an annealing oven. Since the initial, aberration-free pupil matches the target profile quite poorly, the initial cost function is high and the process temperature is set to be quite high also. T is lowered over a number of propagation steps and serves to control the cooling rate of the process. A premise of the theory of simulated annealing is that if the process is cooled sufficiently slowly, local minima will be overcome resulting in convergence to the global optimum. The phase retrieval design with so many optical modes may take on the order of 10,000 cycles to converge.

For an aperture diameter of 20 cm, a supergaussian far-field irradiance distribution of eighth order and width of 800 m, a wavelength of 0.351 m, there are about 100 speckles across the far-field distribution diameter. Counting one optical mode per speckle, we estimate the number of optical modes at about 10,000. Since the phase plate must be specified in the pupil of the system and measured in the far-field, we must perform a diffraction propagation between pupil and far-field for each optimization step.

The literature contains various prescriptions for cooling schedules to achieve optimum or near optimum performance in minimum time. In most cases, the cooling schedules reported were, at least in part, determined heuristically by numerical experiments with the particular type of problem of interest to the respective authors.

Example 98a.inp is run once to initialize the calculation. Ex98b.inp is then run as many times as necessary to achieve convergence. Run ex98c.inp to display any final graphics. The variable `count` should be reset to keep track of the total number of prior trials. To make the calculation times practical on PC computers, the aperture has been reduced to 10 cm diameter, allowing an array of only 512 x 512. Essentially complete convergence was achieved after 100,000 cycles.

References

1. Nobukasu Yoshikawa and Toyohiko Yatagai, "Phase optimization of a kinoform by simulated annealing," *Appl. Opt.* 33,863-868 (1994).
2. S. Kirkpatrick, C. D. Gelatt, Jr., M. P. Vecchi, "Optimization by simulated annealing," *Science* 220, 671-680 (1983).
3. Scott Kirkpatrick, "Optimization by simulated annealing: Quantitative studies," *J. Stat. Phys.* 34, 975-986 (1984).
4. Shizhou Yin, Mingzhe Lu, Chulung Chen, and Francis T. S. Yu, and Tracy D. Hudson and Deanna K. McMillen, "Design of a bipolar composite filter by a simulated annealing algorithm," *Opt. Lett.* 20, 1409-1411 (1995).
5. Harold Szu and Ralph Hartley, "Fast simulated annealing," *Phys. Lett. A* 122, 157-162 (1987).
6. K. Ergenzinger, K. H. Hoffman and P. Salamon, "Optimal simulated annealing schedules for self similar systems," *J. Appl. Phys.* 77, 5501-5508 (1995).

Ex98a: Initialize array

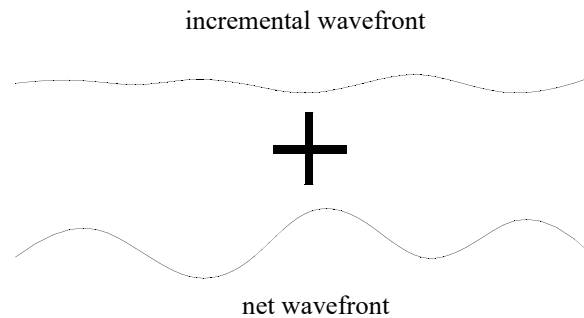


Fig. 98.1. An incremental wavefront is added to the net wavefront, if the system is improved, the incremental change is kept.

Input: ex98a.inp

```
c## ex98a.inp
c
c  Design of far-field distribution by simulated annealing: start
c
c  Beam 1: propagating beam
c  Beam 2: temporary storage for Beam 1
c  Beam 3: phase plate to be added
c  Beam 4: ideal beam, far-field
c
c  Run this file first one time.
c  Then run ex98b.inp one or more times until merit function
c  appears to stabilize at the best value.
c
variab/dec/int nline count rejected recopy pass continue tstep passmin
variab/dec/int n_times iplot reverse accept
iplot = 0
c
c  temp          - temperature
c  sum1          - cumulative sum
c
nline = 512                # array size
auto = .4
n_points = min(1.41*nline/2,512) # number of points to calculate
clap1 = 5.0                # radius of flat top function for pupil
clap2 = .035               # supergaussian radius for far-field
mem/set/b 48
array/s 1 nline            # set array size
x=srand(11)
c
c  Create target supergaussian shape
c
units2 = 1/30
```

Jump to: [Commands](#), [Theory](#)

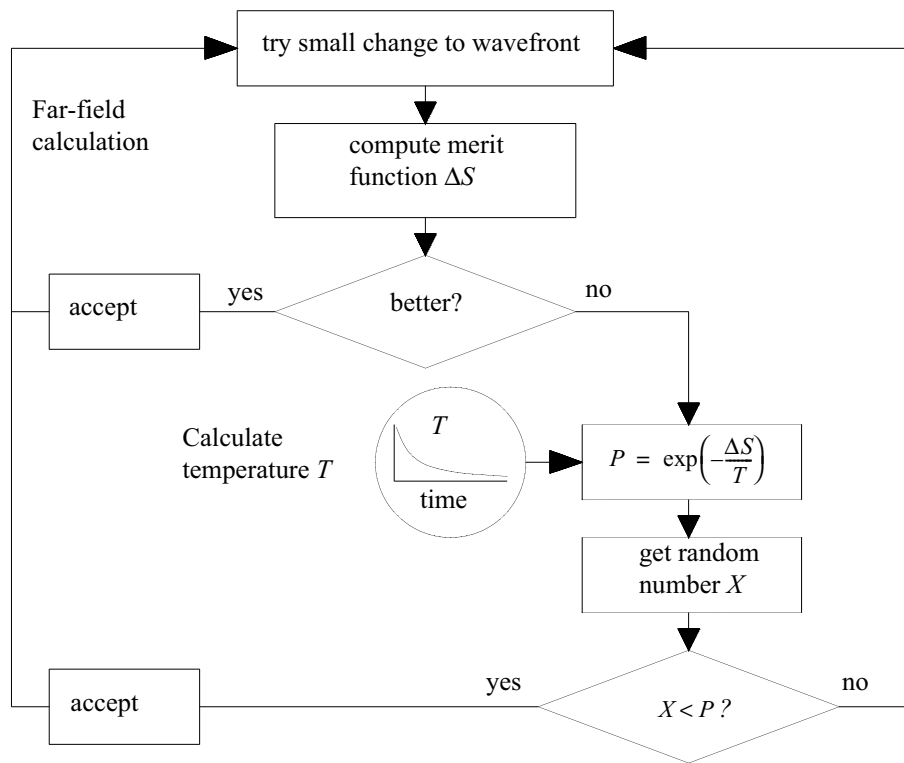


Fig. 98.2. The simulated annealing method. An incremental wavefront with small magnitude aberration is added to the pupil of the lens. The distribution is propagated to the far-field and the merit function is computed. If the merit function is improved, the incremental change is accepted and a new incremental change is computed. If the merit function is worse, a random number is computed with probability based on the merit function ΔS and a decaying function referred to as temperature T . The temperature is made to decay according to some schedule of the number of cycles to simulate cooling of an oven. If $X < P$, then the change is accepted even though the merit function gets worse, otherwise the incremental change is rejected and the wavefront is restored to its prior state and a new incremental wavefront is computed.

```

units1 = .3511e-4*180/(nline*units2)  # calculate far-field units
units/s 1 units1
c
c calculate ideal shape for target and store in udata column 2
c
gaus/c/c 1 1 clap2 4
energy/norm 1 1
encir/cal/pro 1
encir/udata 2
c
c Make three beams
c
nbeam 3 data
wavelength/set 0 .3511
units/s 1 units2          # propagating beam
units/s 2 units2          # save near-field pupil
units/s 3 units2          # phase screen

```

Jump to: [Commands](#), [Theory](#)

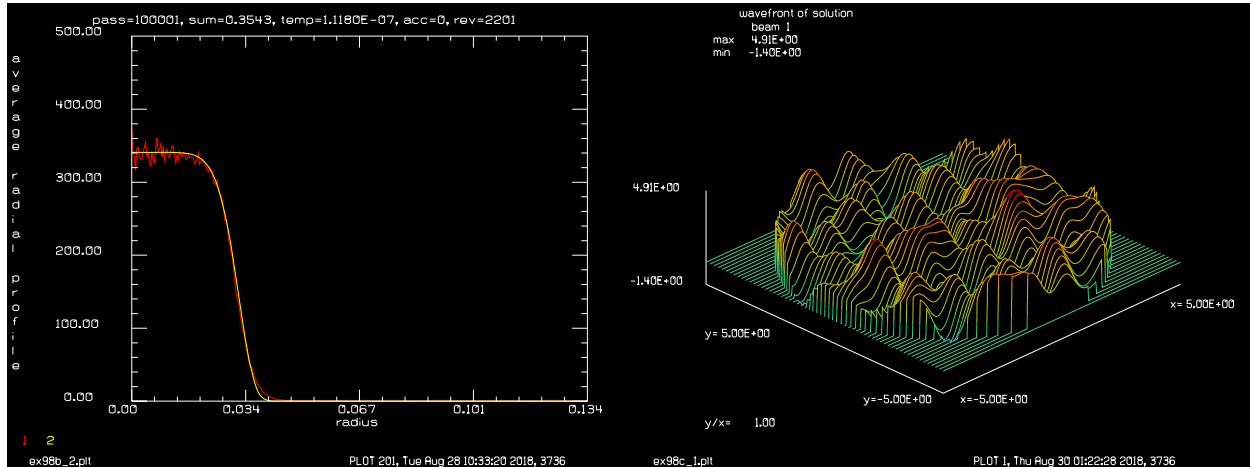


Fig. 98.2a. Average radial profile (red) fit to target supergaussian (yellow), after 100,000 iterations.

Fig. 98.2b. Wavefront after 100,000 iterations.

```
clear 1 1
clap/c/c 1 clap1
energy/norm 1 1
phase/random/sqr/clap 1 1 auto          # build random phase
lens 1 180
copy 1 2                                # save starting function
plot/w ex98a_1.plt
t0 = .001
tstep=0
temp = t0
t_set = 100
tempmin=1
sum1 = 100000
sum2 = 100000
summin = 100000
passmin = 0
recopy = 0
n_times = 1
mac anneal/n_times
outfile sa_data.dat/no/binary 2
c Save parameters
write/disk sa_data.txt/o
pass=
sum=
sum1=
sum2=
summin=
write/disk/off
plot/w ex98a_2.plt
title starting phase plate
plot/x 1 pmin=-20 pmax=20

mac/def xxx/o
    count = count + 1
    variab/set/param v1 count 1 udata/point
```

Jump to: [Commands](#), [Theory](#)

```

    variab/set/param v2 count 2 udata/point
    sum = sum + 2.*pi*(count-.5)*units1^2*(v1-v2)^2
macro/end
macro/def anneal/o
    pass = pass + 1
    clear 3 1                                # clear Beam 3
    phase/screen/clap 3 .03 auto             # build random phase
    mult/beam 1 3                            # apply phase random
    dist 180 1                              # move to far-field
    encir/calc/profile 1
    encir/udata 1
#    sum = 0; count = 0; macro/run xxx/n_points
    udata/radrms 1 2
    variab/set udata_radrms udata/radrms
    sum = udata_radrms*units1^2
    title pass = @pass , sum = @sum , summin = @summin, accept = @accept
    plot/udata 1 2
    sumdif = sum-sum1
    relsumdif = sumdif/sum list
    recopy = 0
    if sum > sum1 then
        p = exp(-(sum-sum1)/temp)
        x = rand()
c Force recopy for sa_start
        if 1 then
            rejected = rejected + 1
            recopy = 1
        else
            reverse=reverse+1
        endif
        x=
        p=
    endif
    if iplot then
        plot/l 1
    endif
    dist -180 1                             # propagate back to pupil
    if recopy then
c reject
        copy 2 1
        sum1 = sum2
        sum = sum1
    else
c accept
        copy 1 2
        sum2 = sum
        sum1 = sum
        accept = accept+1
    endif
    clap/c/c 1 clap1
    if sum<summin then
        summin = sum
        passmin = pass
        plot/watch ex98a_2.plt

```

```

        plot/udata 1 2
        plot/watch ex98a_1.plt
    endif
    sum=
    pass=
    summin=
    passmin=
    rejected=
    temp=
    recopy=
    relsumdif=
    temp = temp*(1. - 1./t_set) list
macro/end

```

Ex98b: Run this example to convergence, about 100,000 times

Input: ex98b.inp

```

c## ex98b.inp
c
c  Design of far-field distribution by simulated annealing: next
c
c  Beam 1: propagating beam
c  Beam 2: temporary storage for Beam 1
c  Beam 3: phase plate to be added
c
c  Run ex98a.inp to setup the calculation.
c  Run ex98b.inp until the calculation stabalizes.
c
c  t_set sets the settling time constant. Longer settling times
c  will generall improve the solution a bit although
c  the number of iterations must be must higher.
c
c  n_times determines how many times ex98b will loop.
c  For these conditions, about 100,000 loops resulted
c  in the merit functikon summin < 0.3.
c
c  This example may benefit from better sampling, for
c  example going to 1024 x 1024 array for the same aperture sizes.
c
variab/dec/int nline count rejected recopy pass continue passmin
variab/dec/int n_times reverse accept pnum switch arg
c
c  temp          - temperature
c  sum1          - cumulative sum
c
nline = 512                # array size
auto = .4
n_points = min(1.41*nline/2,512) # maximum number of points
clap1 = 5.0                # radius of flat top function for pupil
clap2 = .035               # supergaussian radius for far-field
array/s 1 nline            # set array size

```

Jump to: [Commands](#), [Theory](#)

```

udata/numdatasets 2
udata/dataset 1
x=srand(time())           # reset random seed to time variable
x=0;
p=0;
c
c  Create target supergaussian shape
c
units1 = 1/30
units2 = .3511e-4*180/(nline*units1)  # calculate far-field units
units/s 1 units2
gaus/c/c 1 1 clap2 4
energy/norm 1 1
encir/cal/pro 1
encir/udata 2
c
c  Make three beams
c
nbeam 3 data
wavelength/set 0 .3511
units/s 1 units1          # propagating beam
units/s 2 units1          # save near-field pupil
units/s 3 units1          # phase screen
read/disk sa_data.txt     # read in variables pass and the suns
pass=
c connect/keep sa_data 1
infile sa_data.dat/no/binary 1
normalize 1               # trim up aperture shape
clap/c/c 1 clap1
energy/norm 1 1
plot/w ex98b_1.plt
plot/x 1 pmin=-20 pmax=20
plot/w ex98b_2.plt
lens 1 180
copy 1 2                  # save starting function

t0 = .03                  # starting temperature, set very low, .02
t_set = 8000              # 1/e cycles for settling
passmin = 0
recopy = 0
read/disk sa_data.txt
pass=
temp = t0*exp(-pass/t_set) list # set temperature
n_times = 20000
write/off
time/i
mac anneal/n_times
write/on
time
c disconn 2 sa_data       # write out current phase screen
outfile sa_data.dat/no/binary 2
write/on
temp=
write/disk sa_data.txt/o

```

Jump to: [Commands](#), [Theory](#)

```

pass=
sum=
sum1=
sum2=
summin=
reverse=
write/disk/off

end
mac/def xxx/o
  count = count + 1
  variab/set/param v1 count 1 udata/point
  variab/set/param v2 count 2 udata/point
  sum = sum + 2.*pi*(count-.5)*units2^2*(v1-v2)^2
macro/end
macro/def anneal/o
  pass = pass + 1
  normalize 1                                # trim up aperture shape
  clap/c/c 1 clap1
  energy/norm 1 1
  clear 3 1                                  # clear Beam 3
  phase/screen/clap 3 .02 auto                # build random phase
  mult/beam 1 3                               # apply phase random
  dist 180 1                                  # move to far-field
  encir/calc/profile 1
  encir/udata 1
#  sum = 0; count = 0; macro/run xxx/n_points
  udata/radrms 1 2
  variab/set udata_radrms udata/radrms
  sum = udata_radrms*units2^2
  if [mod(pass-1,100)==0] then
    title pass=@pass, sum=@sum, temp=@temp, acc=@accept, rev=@reverse
    plot/udata 1 2 max=500
  endif
  switch = mod(pass-1,500)
  if switch=0 then
    pnum = (pass-1)/500 + 1
    sys/copyfile ex98b_1.plt plt@pnum.plt
  endif
  sumdif = sum-sum1
  relsumdif = sumdif/sum list
  recopy = 0
  if sum>sum1 then
    arg = -(sum-sum1)/temp
    if [arg>-100] then
      p = exp(arg)
    else
      p = 0.
    endif
    x = rand()
    if x>p then
      rejected = rejected + 1
      recopy = 1
    else

```

```

        reverse=reverse+1
    endif
    x=
    p=
endif
dist -180 1                                # propagate back to pupil
if recopy then
c reject
    copy 2 1
    sum1 = sum2
    sum = sum1
else
c accept
    copy 1 2
    sum2 = sum
    sum1 = sum
    accept = accept+1
endif
if sum<summin then
    summin = sum
    udata/dataset 2
    write/on
    Count = Count + 1
    udata/set Count pass summin
C C Count=@Count, pass=@pass, summin=@summin, reverse=@reverse
    write/off
    plot/w ex98b_4.plt
    title Merit function progress
    plot/udata min=0
    udata/dataset 1
    passmin = pass
    outfile sa_data.dat/no/binary 2
    sys/copyfile sa_data.dat best.dat
    plot/watch ex98b_3.plt
    title Best pass=@pass, summin=@summin
    plot/udata 1 2 max=500
    plot/watch ex98b_2.plt
    title pass=@pass, sum=@sum, temp=@temp, acc=@accept, rev=@reverse
endif
sum=
pass=
summin=
passmin=
rejected=
temp=
recopy=
relsumdif=
temp = t0*exp(-pass/t_set) list
macro/end

```


Ex98c: Final run to make phase plot

Input: ex98c.inp

```

c## ex98c!402168444358407
c# write/off
c
c  Design of far-field distribution by simulated annealing: last
c
c  Beam 1: propagating beam
c  Beam 2: temporary storage for Beam 1
c  Beam 3: phase plate to be added
c
variab/dec/int nline count rejected recopy pass continue passmin
variab/dec/int n_times reverse accept pnum switch n_points
c
c  temp          - temperature
c  sum1          - cumulative sum
c
nline = 512                      # array size
n_points = min(1.41*nline/2,512) # maximum number of points
clap1 = 5.0                      # radius of flat top function for pupil
clap2 = .035                     # supergaussian radius for far-field
mem/set/b 48
array/s 1 nline                  # set array size
x=srand(time())                  # reset random seed to time variable
x=0;
p=0;
c
c  Create target supergaussian shape
c
units1 = clap1/150
units2 = .3511e-4*180/(nline*units1) # calculate far-field units
units/s 1 units2
gaus/c/c 1 1 clap2 4
energy/norm 1 1
encir/cal/pro 1
encir/udata 2
c
c  Make three beams
c
nbeam 3 data
wavelength/set 0 .3511
units/s 1 units1                  # propagating beam
units/s 2 units1                  # save near-field pupil
units/s 3 units1                  # phase screen
auto = .4
c connect/keep 1 sa_data
infile sa_data.dat/no/binary 1
normalize 1                        # trim up aperture shape
clap/c/c 1 clap1
energy/norm 1 1
plot/w ex98c_1.plt

```

Jump to: [Commands](#), [Theory](#)

```
title wavefront of solution
plot/1/w xrad=5 ns=64
end
```

Ex99: Michelson interferometer and point diffracting interferometer

Table. 99.1. Table of Ex99 examples

Ex99a: Initialize array	2
Ex99b: Michelson interferometer, relative tilt.	5
Ex99c: Michelson interferometer, finite spectral width.	8
Ex99d: The point diffracting interferometer	11

This example illustrates a Michelson interferometer. This interesting device consists of a finite sized diffuser plate (such as ground glass) illuminated by broad band illumination. This interferometer was used in Michelson's classical experiments showing that there was no ether. The interferometer is capable of showing small displacements in the length of the two arms without requiring laser illumination. As will be discussed, it exhibits problems of fringe localization. The Tyman-Green interferometer may be considered to derive from the Michelson interferometer, but because it uses laser illumination there are no fringe localization problems and the Tyman-Green has almost completely replaced the Michelson interferometer as the method of choice. Born and Wolf, *Principles of Optics*, Sect. 7.5.4, Pergammon Press, have an excellent discussion of the Michelson interferometer.

Figure 99.1 shows a typical configuration. The finite sized source is shown on the left. The source is a diffuser—a rough surface such that the phase at any point is essentially uncorrelated with the phase of neighboring points. Consequently, each point interferes only with the image of itself. Although in principle the device can work with white light, use of narrow band illumination makes it much easier to set up the instrument. The beam divider element is a second surface mirror. To color-compensate the device a tilted plate of identical glass and thickness is placed in one arm, so both arms have an identical glass path. When the light from the two arms are recombined, a telescope is used to observe the fringes, effectively by viewing from infinity. To understand the device, we can consider the “unfolded” system as shown in Fig. 99.2. When the arms are of unequal length, the two images of the source appear to be of different distances. Because the source is a rough surface (delta-correlated) a point interferes only images of the same point interfere. In Fig. 99.2, we see that the bulls eye fringes created by the image pair P1 and P1' are centered on the axis while the image pair P2 and P2' create an offset bulls eye pattern as observed at the plane of the lens. Since the various image pairs all have different centers of their bulls eye patterns, the fringe pattern at the plane of the lens will be washed out if the source has any significant extent. However, at the focal plane of the lens, where the far-field is found, the bulls eye fringes for all image pairs of points are coincident and clear fringes are observed for light of sufficiently narrow spectrum. For this reason we say the fringes due to unequal arms are localized at infinity. Example 99a illustrates the fringe localization at the lens rear focal plane.

The source may be represented as a rough surface. A perfectly rough surface (delta-correlated) can be modeled using the noise command to generate the complex amplitude that would be created by the rough surface. Delta-correlated noise will scatter the light too much resulting in aliasing. To keep aliasing from being a problem, the noise is smoothed slightly so that the distribution over fills the lens but does not over fill the array.

In Fig. 99a.1 shows the model of the source distribution from Ex99a. It consists of a fine speckled pattern corresponding to a rough surface. Smoothing with radius 18 micron radius has been applied so the energy overfills the 0.5 cm radius lens but does not overfill the array which would cause aliasing. Fig. 99a.2

shows the distribution at the plane of the lens. Since the fringes are localized at infinity, no fringes are observed in this plane. Fig. 99a.3 shows bulls eye fringes at the rear focal planes of the lens.

Example 99b illustrates the effect of relative tilt between the mirrors. The images of the source appear to have been rotated about their respective mirrors. Linear fringes are observed at the plane of the mirrors (Fig. 99b.1) and of course at the image of this plane which is formed behind the lens (Fig. 99b.3). At other planes Fig. 99b.2, at the plane of the lens) the fringe patterns of the various pairs of image points are sheared so that the fringe pattern is washed out if the source has any significant size.

Example 99a, discussed above, showed bulls eye fringes due to a large displacement of 5.12 cm. This would only be observed if the coherence length of the light

$$\text{coherence length} = \frac{\lambda^2}{\Delta\lambda} \quad (99.1)$$

The coherence length would have to be significantly larger than the longitudinal displacement of the images (twice the displacement of the mirrors) for fringes of good contrast to be observed.

Given a spectral distribution centered at λ_0 , the relative phase of slightly different wavelengths is

$$\text{relative phase} = 2\pi\Delta L \frac{\lambda - \lambda_0}{\lambda^2} \quad (99.2)$$

where ΔL is the difference in length between the two arms of the interferometer. The interference pattern produced by each wavelength sample includes a slightly different relative phase difference. The interference patterns will be incoherently summed with weighting by the intensity of the light at that wavelength sample. In Ex 99c., equal power-weighting of all wavelength samples is assumed with the coherence length set exactly to the source separation of 5.12 cm. In this particular case the relative phase will scan over the range of 0 to 360 degrees with equal intensity weighting. Because the wavelength range is quite slight, the scale of the bulls eye pattern does not change over the range. The center fringe, however, varies from being a bright fringe to a dark fringe and back again when the phase is scanned over 360 degrees and the outer fringes vary in similar fashion so that the bulls eye fringes are washed out. Figure 99c.1 shows bulls eye fringes formed by strictly monochromatic light. Figure 99c.2 shows the result of a range of wavelengths corresponding to uniform weighting where the wavelength range corresponds to a coherence length equal to twice the difference in the arm length. The fringe visibility is significantly degraded. Fig. 99c.3 shows the interference pattern made by equal weighting of wavelengths having a coherence length equal to the difference in the optical lengths of the arms.

Ex99a: Initialize array

Input: ex99a.inp

```
c## ex99a
#
# The most elementary rendering of a Michelson interferometer
```

Jump to: [Commands](#), [Theory](#)

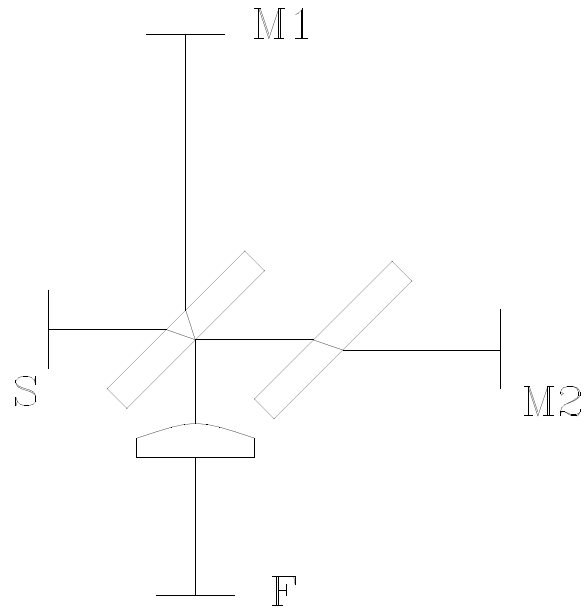


Fig. 99.1. Michelson interferometer. The finite sized source is shown on the left. The source is a diffuser—a rough surface such that the phase at any point is essentially uncorrelated with the phase of neighboring points. Consequently, each point interferes only with the image of itself. Although in principle the device can work with white light, use of narrow band illumination makes it much easier to set up the instrument. The beam divider element is a second surface mirror. To color-compensate the device a tilted plate of identical glass and thickness is placed in one arm, so both arms have an identical glass path. When the light from the two arms are recombined, a telescope is used to observe the fringes, effectively by viewing from infinity.

```
# as the coherent sum of two nearly identical images of a rough
# surface and its image. The images may differ by longitudinal
# displacement, creating bulls eye fringes, or by small tilt,
# creating linear fringes.
#
# We may, to a good approximation, neglect the fold mirrors and
# optical system which forms the two images of the source.
#
# In this example, the mirrors are separated by 2.06 cm, so the
# images of the sources differ by 5.12 cm. Bull eye fringes are
# produced. These fringes are localized at infinity so it is
# necessary to observe at the focal plane of a lens.
#
mem/set/b 16                # use 8 if there is not enough memory for 16
array/s 1 1024
nbeam 2
wavelength/set 0 .5         # select .5 micron wavelength
units/field 0 1.
clear 1 0
noise 1 1
convol/gaus 1 1.8e-3        # smooth speckle to avoid aliasing at lens
clap/c/c 1 .5
title source with rough surface
plot/w ex99a_1.plt
```

Jump to: [Commands](#), [Theory](#)

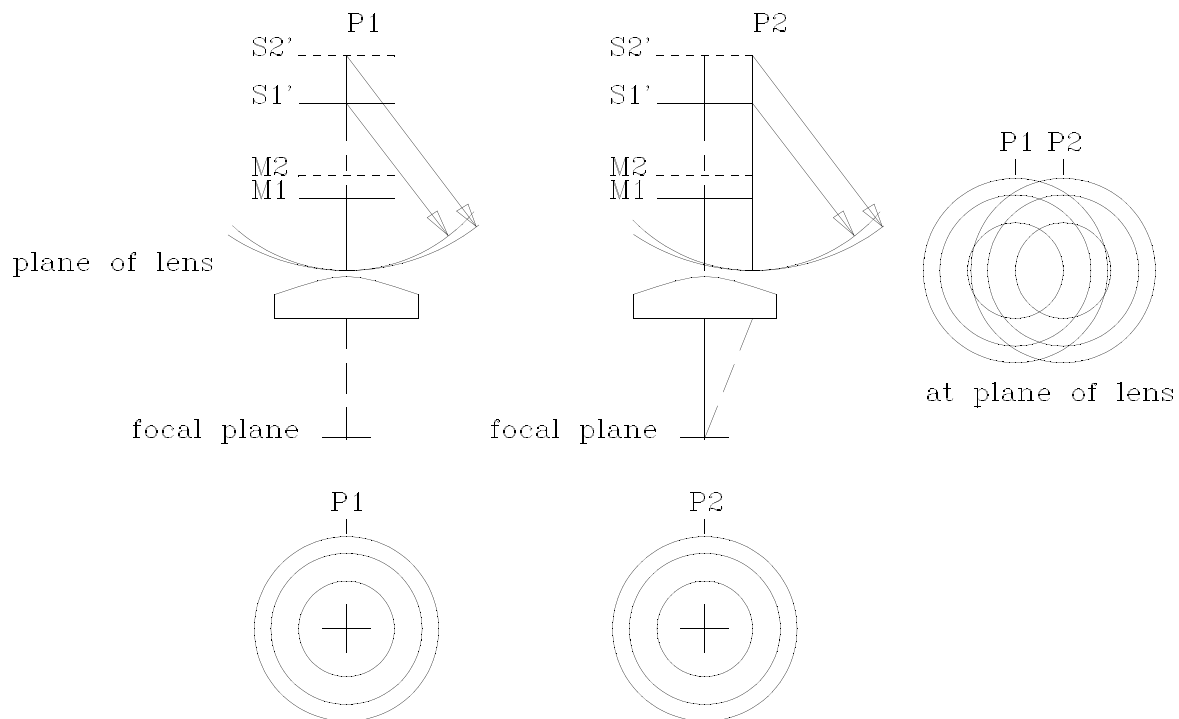


Fig. 99.2. The “unfolded” Michelson interferometer. Mirrors M1 and M2 create images of the source S1' and S2'. Bulls eye fringes created by the image pair P1 and P1' are centered on the axis while the fringes from image pair P2 and P2' create an offset bulls eye pattern as observed at the plane of the lens. Since the various image pairs all have different centers of their bulls eye patterns, the fringe pattern at the plane of the lens will be washed out if the source has any significant extent. However, at the focal plane of the lens, where the far-field is found, the bulls eye fringes for all image pairs of points are coincident and fringe visibility is observed.

```

plot/l xrad=.5 ns=64 theta=42
geodata/set/all 1 waistx=.5 waisty=.5
copy/con 1 2
dist 40 1 # distance from image of source to lens
dist [40+8*.64] 2 # distance from second image of source to lens
add/coh/c 1 2 # coherent sum to make interference pattern
title coherent sum of both beams at lens, no fringes
plot/w ex99a_2.plt
plot/l ns=64 theta=42
clap/c/c 1 .5
lens 1 10 # lens of f=10 cm
dist 10 1 # propagate to far-field
title Interference pattern at far-field, fringes observed
plot/w ex99a_3.plt
plot/l xrad=.08 ns=64 theta=42
set/density 256 256 # set plot density to max value
set/window/abs -.08 .08 -.08 .08 # set window limits
peak/norm 1 1 # normalize peak for convenience
title Interference pattern at far-field, fringes observed
plot/w ex99a_4.plt
plot/bitmap/int/burn 1 max=.1 # saturate peaks

```

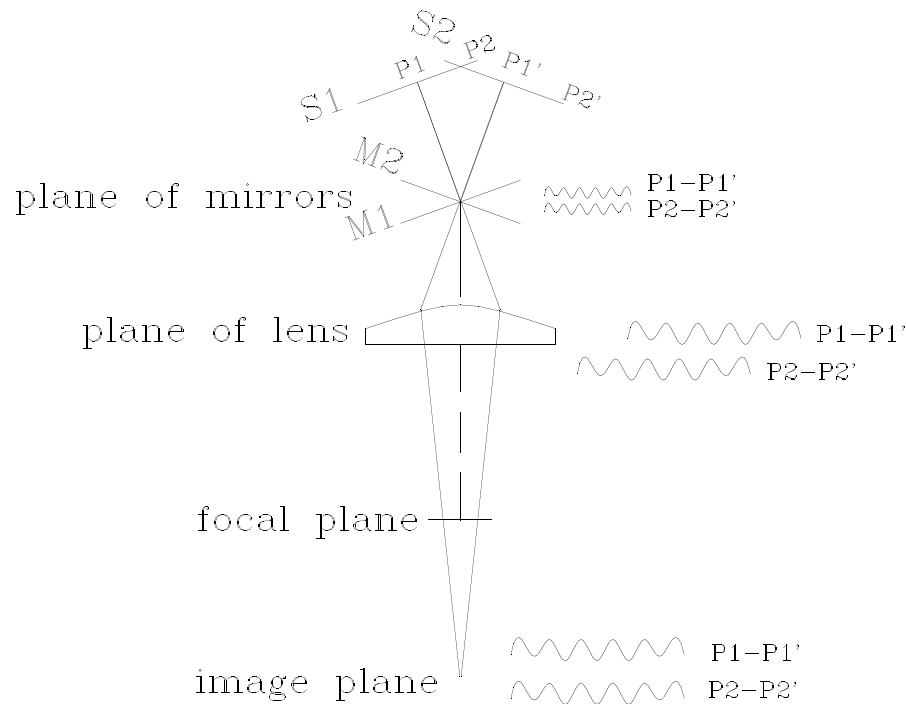


Fig. 99.3. The effect of relative tilt between the mirrors. The images of the source appear to have been rotated about their respective mirrors. Linear fringes are observed at the plane of the mirrors and, of course, at the image of this plane which is formed behind the lens. At other planes the fringe patterns of the various pairs of image points are sheared so that the fringe pattern is washed out if the source has any significant size.

end

Ex99b: Michelson interferometer, relative tilt

Input: `ex99b.inp`

```
c## ex99b
#
# The most elementary rendering of a Michelson interferometer
# as the coherent sum of two nearly identical images of a rough
# surface and its image. The images may differ by longitudinal
# displacement, creating bulls eye fringes, or by small tilt,
# creating linear fringes.
#
# We may, to a good approximation, neglect the fold mirrors and
# optical system which forms the two images of the source.
#
# This example simulates the effects of relative tilt between the
# mirrors. Linear fringes are created that are localized to the
# plane of the mirrors and, of course, the image of the mirror plane
# formed by the lens after the rear focal plane.
#
```

Jump to: [Commands](#), [Theory](#)

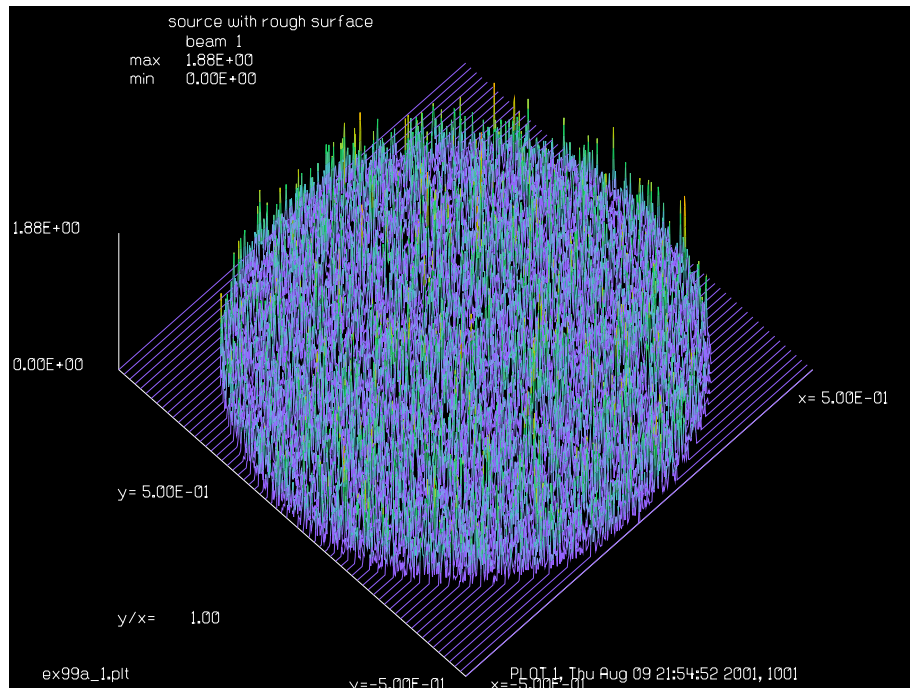


Fig. 99a.1. The rough surface of the source is simulated by a random complex amplitude. A delta-correlated noise function is smoothed slightly with a smoothing radius of 18 microns to limit the angular subtense of the scattered radiation so that the beam does not overfill the array at the plane of the lens (see Fig. 99a.3).

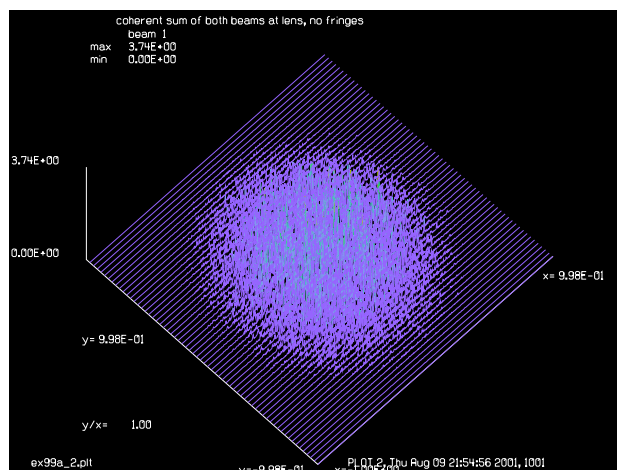


Fig. 99a.2. Interference pattern at the plane of the lens. Note there are no fringes here. The smoothing of the source is enough to keep the energy inside the array but it still overfills the 0.5 cm radius aperture (half the size of the full field).

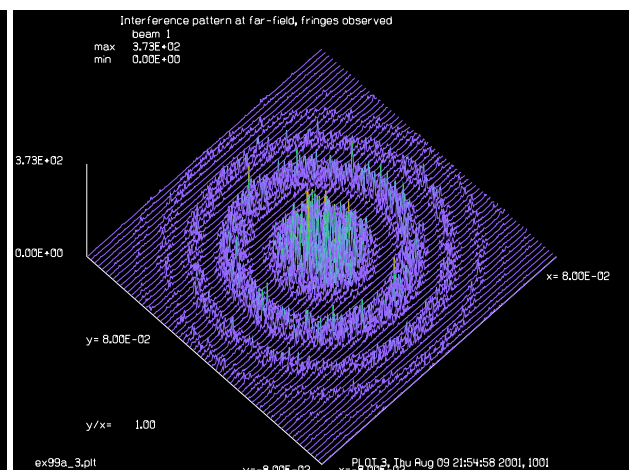


Fig. 99a.3. Interference pattern at the rear focal plane of the lens. The fringes are clearly visible here because they are localized at infinity. The bulls eye pattern is due to the difference in radius because the arms have an optical path difference of 5.12 cm.

```
mem/set/b 16 # use 8 if there is not enough memory for 16
array/s 1 1024
nbeam 2
wavelength/set 0 .5
units/field 0 1.
clear 1 0
```

Jump to: [Commands](#), [Theory](#)

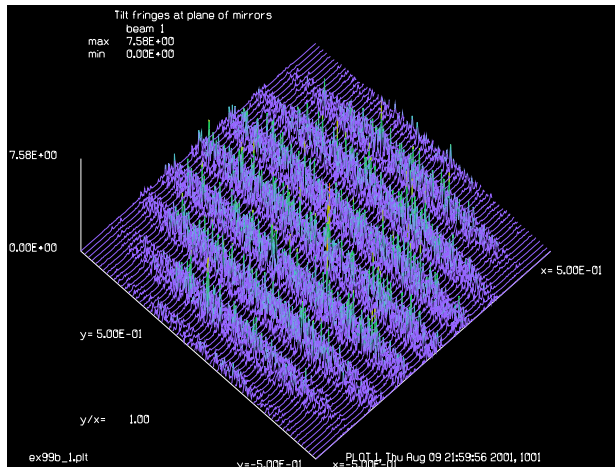


Fig. 99b.1. Tilt fringes at the plane of the mirrors.

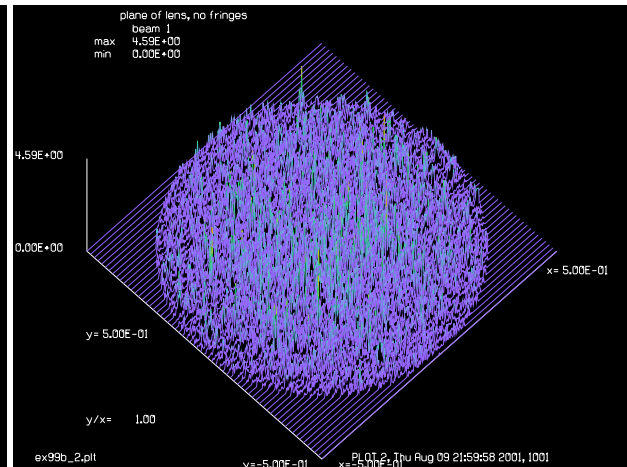


Fig. 99b.2. Interference pattern due to mirror tilt at the plane of the lens, after the 0.5 cm radius aperture. No fringes are observed in this plane, because the linear fringes for each pair of image points are sheared with respect to other pairs of image points.

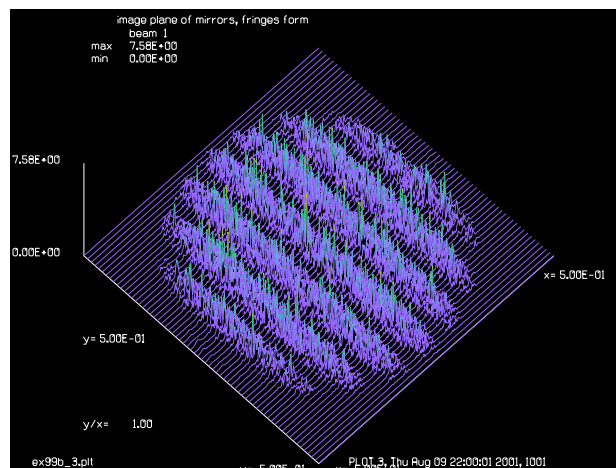


Fig. 99b.3. Linear fringes observed at the image of the plane of the mirrors created by the lens.

```

noise 1 1
convol/gaus 1 1.8e-3      # smooth speckle to avoid aliasing at lens
clap/c/c 1 .5
geodata/set/all 1 waistx=.5 waisty=.5
copy/con 1 2
dist 20 1
dist 20 2
abr/tilt 1 4 rn=1. az=90
abr/tilt 2 -4 rn=1. az=90
add/coh/con 1 2
plot/w ex99b_1.plt
title Tilt fringes at plane of mirrors
plot/l 1 ns=64 xrad=.5 theta=42
dist 20 1
clap/c/c 1 .5

```

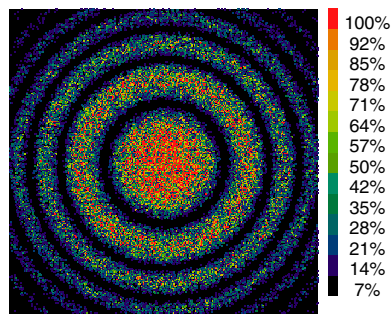
Jump to: [Commands](#), [Theory](#)

```

plot/w ex99b_2.plt
title plane of lens, no fringes
plot/l 1 ns=64 xrad=.5 theta=42
lens 0 10
dist 20 1
plot/w ex99b_3.plt
title image plane of mirrors, fringes form
plot/l xrad=.5 ns=64 theta=42
set/density 256 256          # set plot density to max value
set/window/abs -0.5 0.5 -0.5 0.5 # set window limits
peak/norm 1 1                # normalize peak for convenience
plot/w ex99b_4.plt
plot/bitmap/int/burn 1 max=.1 # saturate peaks
end

```

Ex99c: Michelson interferometer, finite spectral width



Fringe pattern, pass = 1
intensity, Beam 4
range: 0.000E+00, 3.725E+01,
x-limits: -8.000E-02, 8.000E-02,
y-limits: -8.000E-02, 8.000E-02,
PLOT 3, Thu Aug 09 22:21:41 2001, 1001

Fig. 99c.1. Bulls eye fringes for perfectly monochromatic light.

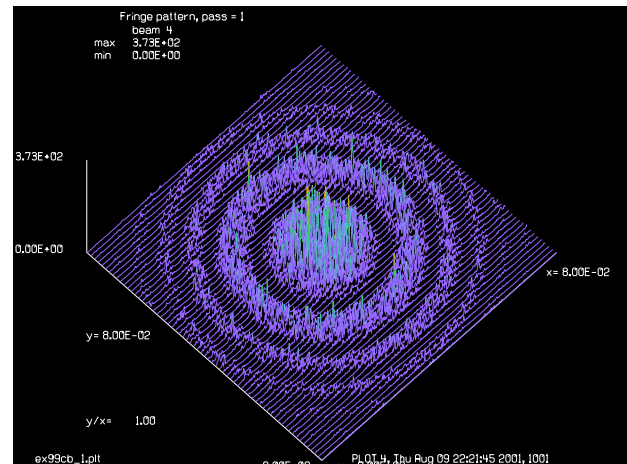


Fig. 99c.2. Bulls eye fringes for perfectly monochromatic light.

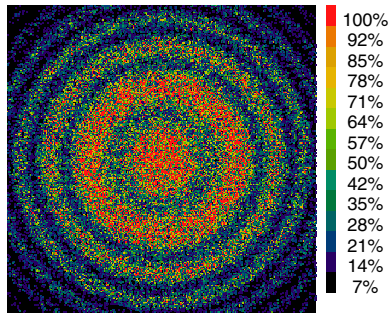
Input: ex99c.inp

```

c## ex99c
#
# The most elementary rendering of a Michelson interferometer
# as the coherent sum of two nearly identical images of a rough
# surface and its image. The images may differ by longitudinal
# displacement, creating bulls eye fringes, or by small tilt,
# creating linear fringes.
#
# We may, to a good approximation, neglect the fold mirrors and
# optical system which forms the two images of the source.
#
# In this example, finite spectral width is simulated by varying
# the phase of one of the beam from 0 to 360 degrees. This

```

Jump to: [Commands](#), [Theory](#)



Fringe pattern, pass = 5
intensity, Beam 4
range: 0.000E+00, 1.032E+02,
x-limits: -8.000E-02, 8.000E-02,
y-limits: -8.000E-02, 8.000E-02,
PLOT 13, Thu Aug 09 22:22:04 2001, 1001

Fig. 99c.3. Bulls eye fringes averaged over a range of wavelengths yielding a coherence length equal to twice the optical path length difference of the arms. Note the loss of fringe visibility.

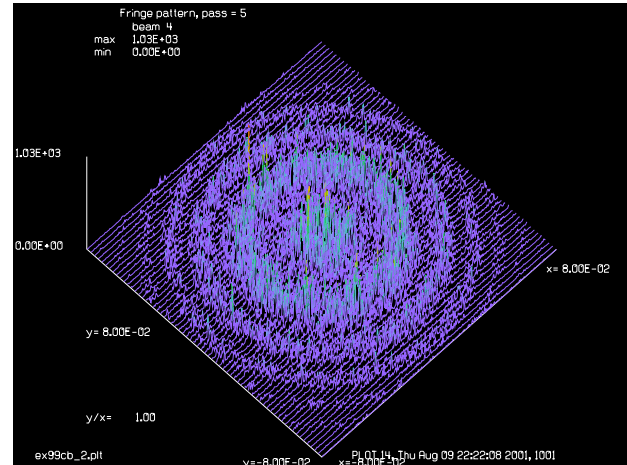
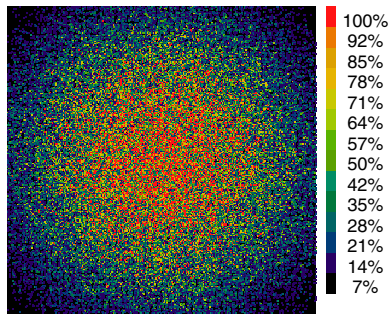


Fig. 99c.4. Bulls eye fringes averaged over a range of wavelengths yielding a coherence length equal to twice the optical path length difference of the arms. Note the loss of fringe visibility.



Fringe pattern, pass = 10
intensity, Beam 4
range: 0.000E+00, 2.111E+02,
x-limits: -8.000E-02, 8.000E-02,
y-limits: -8.000E-02, 8.000E-02,
PLOT 23, Thu Aug 09 22:22:29 2001, 1001

Fig. 99c.5. Bulls eye fringes averaged over a range of wavelengths having a coherence length equal to the optical path difference of the arms. Fringe visibility is completely lost.

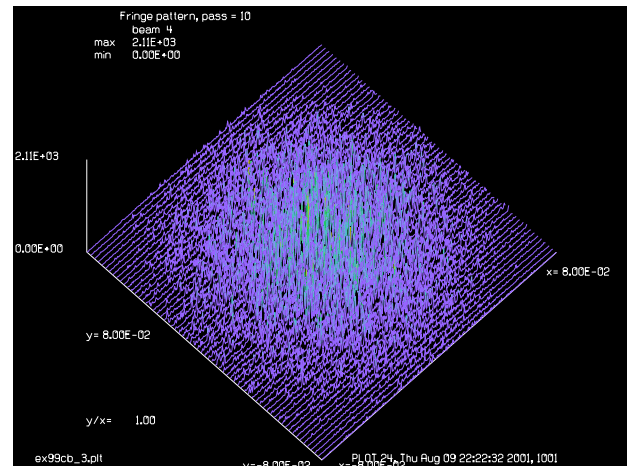


Fig. 99c.6. Bulls eye fringes averaged over a range of wavelengths having a coherence length equal to the optical path difference of the arms. Fringe visibility is completely lost.

```
# corresponds to a coherence length equal to the mirror separation
# of 5.12 cm with uniform spectral weighting.
#
mem/set/b 16      # use 8 if there is not enough memory for 16
variable/dec/int Nlast count
Nlast = 10
array/s 1 1024
nbeam 2
wavelength/set 0 .5
units/field 0 1.
clear 1 0
noise 1 1
convol/gaus 1 1.8e-3      # smooth speckle to avoid aliasing at lens
```

Jump to: [Commands](#), [Theory](#)

```

clap/c/c 1 .5
geodata/set/all 1 waistx=.5 waisty=.5
copy/con 1 2
prop 40 1
prop 45.12 2
clap/c/c 0 .5
lens 0 10
prop 10
nbeam 4
variab/set Units 1 units
units/set 3 Units
units/set 4 Units
wavelength/set 3 .5
wavelength/set 4 .5
set/density 256 256 # set plot density to max value
set/window/abs -.08 .08 -.08 .08 # set window limits
macro/def spectral/o
    count = count + 1
    clear 3 0
    add/coh/con 3 1 2
    add/inc/con 4 3
    phase/piston 1 36
    variab/set Peak 4 peak
    title Fringe pattern, pass = @count
    plot/w ex99ca_3.plt
    plot/bitmap/int/burn 4 max=.1*Peak # saturate peak
    pause 3
    plot/w ex99cb_3.plt
    plot/l 4 xrad=.08 ns=64 theta=42
    if count=1 then
        title Fringe pattern, pass = @count
        plot/w ex99ca_1.plt
        plot/bitmap/int/burn 4 max=.1*Peak # saturate peaks
        pause 3
        plot/w ex99cb_1.plt
        plot/l 4 xrad=.08 ns=64 theta=42
    endif
    if count=5 then
        title Fringe pattern, pass = @count
        plot/w ex99ca_2.plt
        plot/bitmap/int/burn 4 max=.1*Peak # saturate peaks
        pause 3
        plot/w ex99cb_2.plt
        plot/l 4 xrad=.08 ns=64 theta=42
    endif
macro/end
clear 4 0
macro spectral/Nlast
end

```

Ex99d: The point diffracting interferometer

A simple common path interferometer may be made by placing a partially transmitting filter with a small, clear pinhole at the image plane. The pinhole generates a good spherically expanding beam which provides an aberration-free reference when it interferes with the primary beam passing through the partially transmitting region. If the energy passing through the pinhole is nearly equal to the energy passing through the partially transmitting region maximum fringe contrast will be observed.

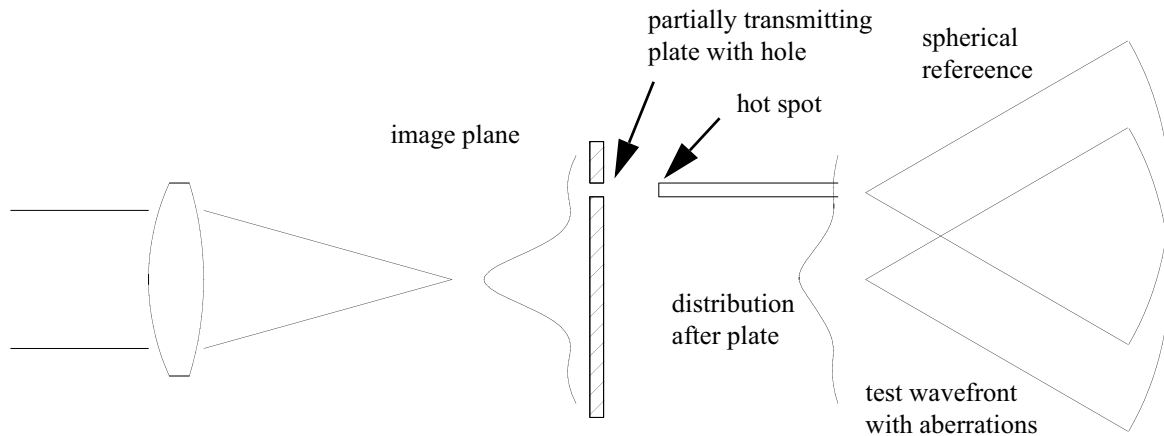


Fig. 99c.7. Point diffracting interferometer. A partially transmitting plate is placed and the image plane. The plate has a small hole which acts as a near-perfect point source (hot spot) to produce a near-perfect expanding spherical wavefront. The spherical wavefront interferes with the original aberrated wavefront which passes through the partially transmitting part of the plate to create an interference pattern. If the transmitted energy is roughly evenly distributed between the pinhole and the rest of the partially transmitting plate, then fringe contrast will be good.

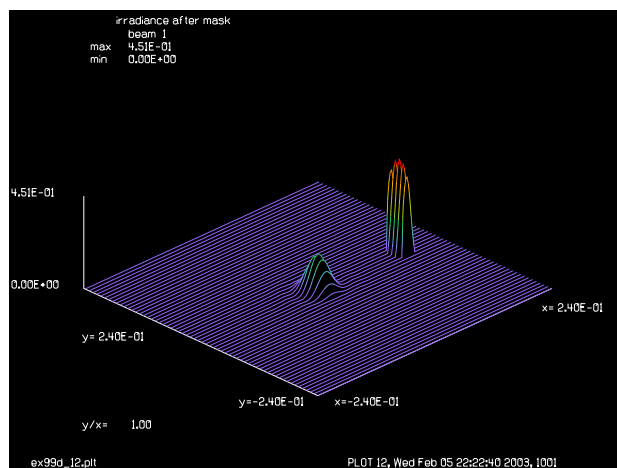


Fig. 99d.1. Main beam passes through the center with significant attenuation. Side beam passes through small hole with 100% transmission. Energy is roughly equal.

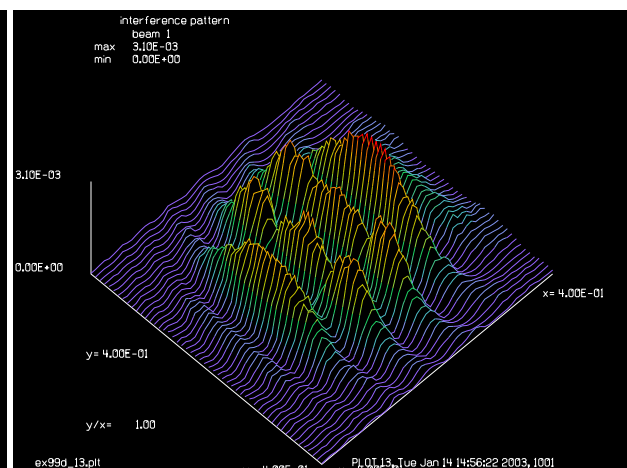


Fig. 99d.2. Interference fringes produced by off-axis pinhole reference with on-axis aberrated beam.

Input: `ex99d.inp`

```
c## ex99d
#
```

Jump to: [Commands](#), [Theory](#)

```

# Point Diffracting Interferometer (Smartt Interferometer)
#
# A simple common path interferometer may be made by placing a
# partially transmitting filter with a small, clear pinhole at
# the image plane. The pinhole generates a good spherically expanding
# beam which provides an aberration-free reference when it interferes
# with the primary beam passing through the partially transmitting
# region. If the energy passing through the pinhole is nearly equal
# to the energy passing through the partially transmitting region
# maximum fringe contrast will be observed.
#
set/alias_stop/off
mem/set/b 6                # memory big enough for 3 arrays
array/s 1 512
units/field 1 4            # set units
clap/c/c 1 .3              # set initial aperture
energy/norm 1 1            # normalize energy
plot/watch ex99d_1.plt 100 100
title interference pattern
plot/1 xrad=.4 ns=128 theta=40
prop 20                    # start one focal length in front of lens
lens 1 20                  # lens
prop 20                    # propagate to focal plane
variab/set Peak 1 peak
plot/watch ex99d_2.plt
title image intensity
plot/x/i le=0 ri=.24
plot/watch ex99d_3.plt
title image wavefront
plot/x/w le=0 ri=.24
plot/watch ex99d_4.plt
title image real part
plot/x/r le=0 ri=.24 fmax=sqrt(Peak)
plot/watch ex99d_5.plt
title image imaginary part
plot/x/ai le=0 ri=.24 fmax=sqrt(Peak)
nbeam 3 data                # make some extra beams (data)
variab/set Units 1 units    # get units from Beam 1
units/set 2 Units           # set units for Beams 2 and 3
units/set 3 Units
r = 1/2.8e-4                # approximate energy through pin hole
b = 1/(1.+sqrt(r))
a = sqrt(r)*b
x = a + b list
mult 2 b^2
mult 3 a^2
clap/c/c 3 .02 xdec=.17     # make pin hole
abr/focus 3 .5 rnorm=.02 xdec=.17 # add some phase to mask
plot/watch ex99d_6.plt
title Beam 3: wavefront
plot/x/w 3 le=0. ri=.24
plot/watch ex99d_7.plt
title Beam 3: transmission
plot/x/i 3 le=0 ri=.24

```

Jump to: [Commands](#), [Theory](#)

```

plot/watch ex99d_8.plt
title Beam 3: real part,2
plot/x/r 3 le=0 ri=.24
plot/watch ex99d_9.plt
title Beam 3: imaginary part,2
plot/x/ai 3 le=0 ri=.24
add/coh/con 2 3          # coherent addition
plot/watch ex99d_10.plt
title transmission of mask
plot/x/i 2 le=0 ri=.24 fmax=1.25
plot/watch ex99d_11.plt
title wavefront of mask
plot/x/w 2 le=0 ri=.24 pmax=.5
mult/beam 1 2
plot/watch ex99d_12.plt
title irradiance after mask
plot/l 1 xrad=.24 ns=64
prop 20                  # prop to expanded pupil
plot/watch ex99d_13.plt
title interference pattern
plot/l xrad=.4 ns=128 theta=40
plot/watch ex99d_14.plt
plot
plot/watch ex99d_15.plt
set/density 64 64        # bitmap burn pattern simulates interferogram
plot/bit/i/burn

```


Ex100: Cavity mode and power spectrum for flat-flat resonator

This is an example of the cavity mode power spectrum of the cavity mode for a flat-flat resonator. Multiple modes beat together to change the pupil distribution but the power spectrum is slowly varying.

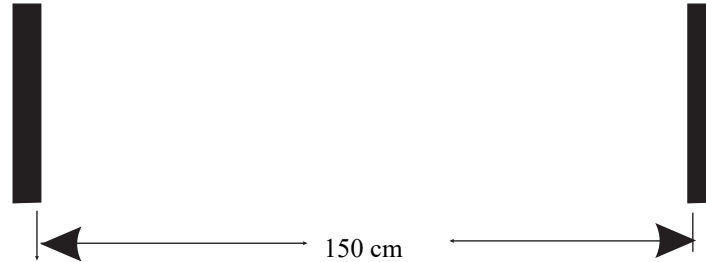


Fig. 100.1. Configuration for flat-flat resonator.

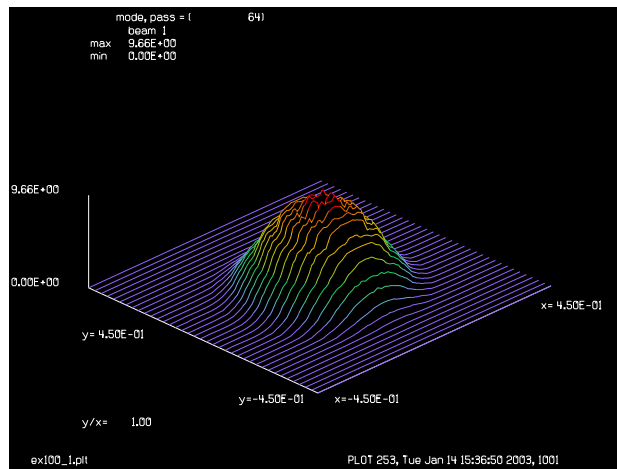


Fig. 100.2. Cavity mode shape at 64th pass.

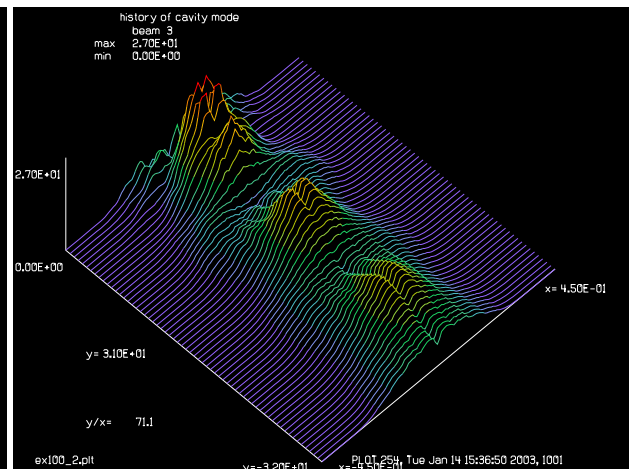


Fig. 100.3. History of cavity mode form over 64 passes.

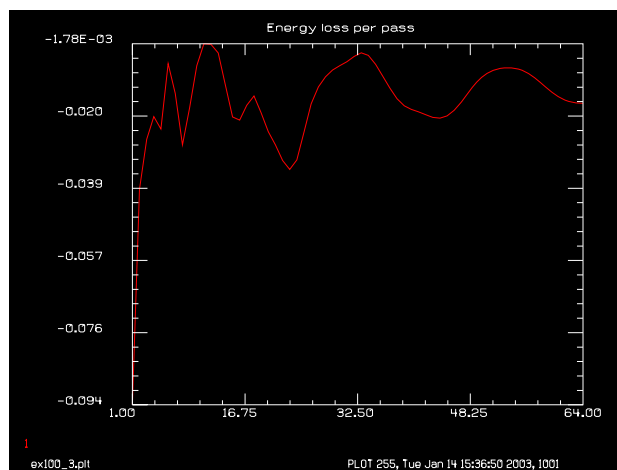


Fig. 100.4. Loss per pass over 64th passes.

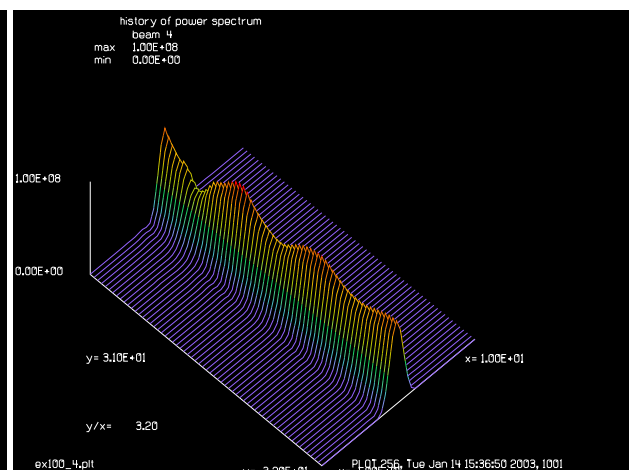


Fig. 100.5. History of power spectrum over 64 passes.

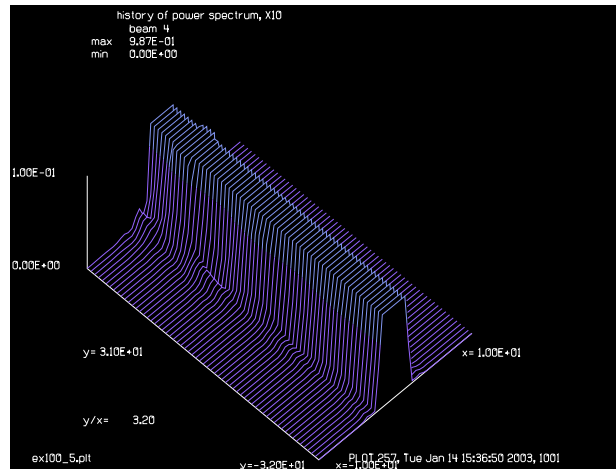


Fig. 100.6. History of power spectrum over 64 passes, x10 scale.

Input: ex100.inp

```

c## ex100
c
c  Example 100: Flat-flat mirror resonator
c
variab/dec/int pass Nline Nline_half Nline3 TEST
Nline = 256
Nline_half = Nline/2 + 1
Nline3 = 64
array/s 1 Nline
nbeam 2
nbeam 3 Nline Nline3 data    # history of cavity mode
nbeam 4 Nline Nline3 data
Lambda = .5e-4
wavelength/set 0 Lambda*1e4    # set wavelengths
Units = .008
units/s 1 Units
units/s 3 Units 1
units/s 4 1./Nline/Units 1
clear 3 0
clear 4 0
Apt = .3
clear 1 1
Length = 150
energy/norm 1 1                # normalize energy
c
c  Force surrogate gaussian beam properties
c
geodata/set/all 1 waistx=2 waisty=2
macro/def step/o
  prop Length
  mirror/flat 1
  clap/c/n 1 Apt                # aperture radius
  prop Length
  if [!TEST] then

```

Jump to: [Commands](#), [Theory](#)

```

    pass = pass + 1                # increment pass counter
    step = step + 1                # increment step number
    title mode, pass = @pass
    plot/w ex100_1.plt
    plot/l xrad=Apt*1.5
    copy/row 1 3 Nline_half pass
    title history of cavity mode
    plot/w ex100_2.plt
    plot/l 3 ns=64 xrad=Apt*1.5 yrad=Nline3/2 theta=40
endif
mirror/flat 1
clap/c/c 1 Apt                    # 1.0 cm. radius aperture
if [!TEST] then
    energy                        # calculate energy in the beams
    variab/set/param Energy 1 energy
    Energy = Energy - 1           # calculate energy difference
    udata/set pass step Energy    # store energy differences
    plot/w ex100_3.plt
    title Energy loss per pass
    plot/udata
    energy/norm 1 1               # renormalize energy
    copy 1 2
    beams/off 1
    beams/on 2
    zreff/se 2 0.
    lens 2 1.
    dist 1 2
    units/s 2 1./Nline/Units 1./Nline/Units # just set units
    copy/row 2 4 Nline_half pass
    title history of power spectrum
    plot/w ex100_4.plt
    plot/l 4 ns=64 xrad=10 yrad=Nline3/2 theta=40
    beams/off 2
    beams/on 1
endif
macro/end
pass = 0 ; step = 0                # initialize variables
TEST = 1
reson/name step
reson/eigen/test 1
reson/eigen/set 1
clear 1 1                          # start with uniform intensity
clap/c/n 1 Apt                     # initial aperture
energy/norm 1 1                    # normalize energy
TEST = 0
resonator/run Nline3
peak/norm 4 1
plot/w ex100_5.plt
title history of power spectrum, X10
plot/l 4 ns=64 xrad=10 yrad=Nline3/2 theta=40 max=.1

```


Ex101: Measurement of excimer laser with Moire fringes

Table. 101.1. Table of Ex101 examples

Ex101a: Crossed Ronci rullings with slight rotation, coherent input	1
Ex101b: Crossed Ronci rullings with slight rotation, excimer with 200 speckle	3

Moire interferometry is often used to measure the spatial coherence of multimode lasers such as the excimer laser. Figure 1 illustrates an excimer laser incident on two gratings with a small mutual rotation. The mutual rotation yielded Moire fringes. The contrast in the Moire fringes is a measure of the spatial coherence at the spatial width associated with the period of the Moire fringes. It is easy to change the period of the Moire fringes by changing the relative rotation angle. The gratings are separated by Talbot distance associated with the fine period of the gratings (not the Moire fringes) so that the first gratings is imaged by diffraction into the plane of the second grating. Let the period of the gratings be p . The Talbot distance is $z_{talbot} = 2p^2/\lambda$. If one grating is rotated $+\theta$ and the other $-\theta$, then the Moire fringes have the period $p_{moire} = p/2\tan\theta$. Figure 2 illustrates schematically, Moire fringes created by overlapping gratings with a small mutual rotation. The Moire interferometer is a form of linear shearing interferometer. The shear for grating separated by one Talbot distance is

$$\text{shear separation } \frac{2p^2\lambda}{\lambda p} = 2p \quad (101.1)$$

so the shear separation is $2p$.

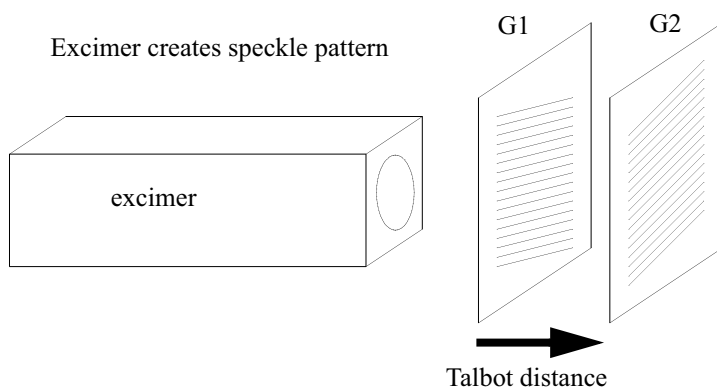


Fig. 101.1. An excimer laser illuminates two gratings with a slight mutual rotation, separated by the Talbot distance.

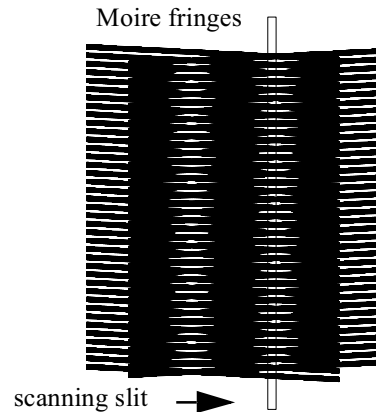


Fig. 101.2. Crossed gratings with a small relative tilt show Moire fringes. A scanning averages the data along the fringes.

Ex101a: Crossed Ronci rullings with slight rotation, coherent input

Ex101.1 illustrates a perfectly coherent beam with two grating with a mutual rotation of 5.725. Rounded edges minimize edge diffraction. The period is 1/120 cm giving 60 cycles over the diameter. Fig. 101a.2 shows a blown up view covering 0.5 cm and showing 6 fringes. Fig. 101a.3 shows a profile of the Moire fringes. The fringes are not simple sinewaves as they contain higher order terms. Also, even in the ideal case,

the Moire fringes are not 100% modulated because of the multiple order interference. The Moire interferometer may also be viewed as a form of lateral shearing interferometer where +1, 0, and -1 orders interfere.

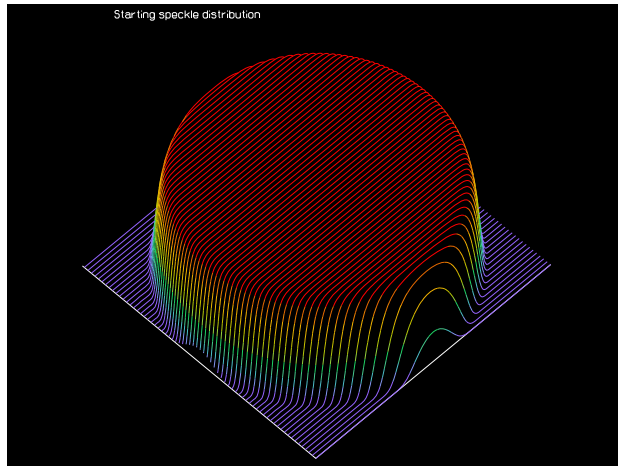


Fig. 101a.1. Aperture of diameter 0.5 cm with grating modulation of $p = 1/120$, so there are 120 cycles.

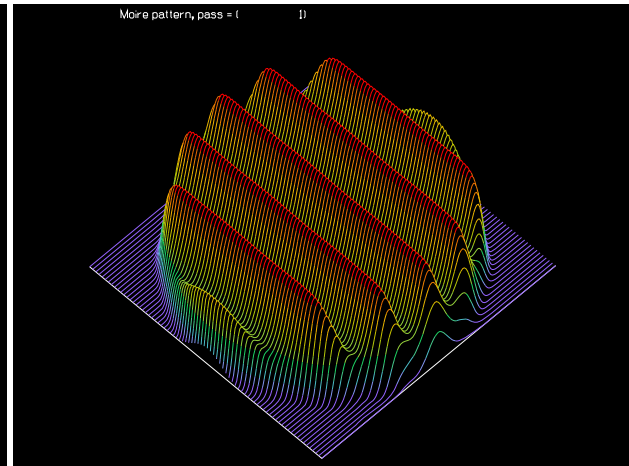


Fig. 101a.2. Close-up view of 6 Ronchi cycles over 0.5 cm diameter.

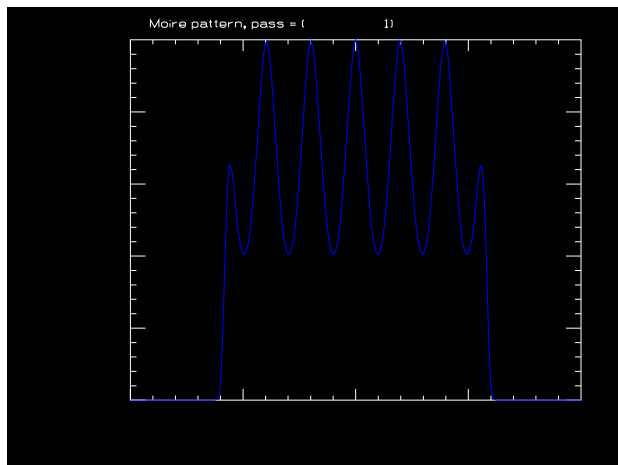
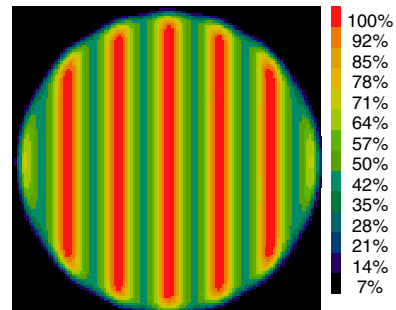


Fig. 101a.3. Close-up view of Moire pattern showing 6 fine fringes and one cycle of Moire fringe.



Moire pattern, pass = (1)
 intensity, Beam 3
 range: 0.000E+00, 1.350E+00,
 x-limits: -2.557E-01, 2.523E-01,
 y-limits: -2.523E-01, 2.557E-01,
 PLOT 4, Tue Jan 14 16:08:49 2003, 1001

Fig. 101a.4. Bitmap view of the Moire pattern.

Input: ex101a.inp

```
c## ex101a
#
# Example: Crossed Ronchi rullings with small relative rotation
#
set/cpu 2
mem/set/b 18
set/label/off
variab/dec/int pass Nline
Nline = 1024
array/s 1 Nline
```

Jump to: [Commands](#), [Theory](#)

```

array/s 2 256 256 data
array/s 3 256 256 data
array/s 4 256 1 data
clear 3 0
period = 1/120
Field = .85/2
Apt = .25
units/field 1 Field
units/field 2 Field
units/field 3 Field
units/field 4 Field
wavelength/set 0 .5106
echo/on
macro/def step/o
    pass = pass + 1 list
    zreff/se 1 0
    array/s 1 Nline
    units/field 1 Field
c  noise/smooth 1 1 .01
    split/c/in 1 Apt 20
    energy/norm 1 1
    title Starting speckle distribution
    plot/w ex101a_1.plt
    plot/l xrad=Apt ns=64 theta=40
    grat/c 1 pe=period az=90+2.864
    prop 2.720
    grat/c 1 pe=period az=90-2.864
    lens 1 100
    dist 100 1
    clap/c/c 1 .3
    dist -100
    copy/border 1 2 0 0 Nline/256 Nline/256
    add/inc/c 3 2
    plot/w ex101a_2.plt
    title Moire pattern, pass = @pass
    plot/l 3 xrad=Apt ns=64 theta=40
    copy/row 3 4 129 1
    plot/w ex101a_3.plt
    title Moire pattern, pass = @pass
    plot/x/i 4
macro/end
macro/run step/1
plot/w ex101a_4.plt
title Moire pattern, pass = @pass
set/den 128 128
set/win/rel 20 80 20 80
plot/b/i/b 3

```

Ex101b: Crossed Ronchi rulings with slight rotation, excimer with 200 speckle

In Ex101b an excimer beam is simulated by using gaussian speckle. The autocorrelation diameter was set to 200 microns for the speckle. Fig 101.7 shows an instantaneous view of a typical excimer beam. One

Jump to: [Commands](#), [Theory](#)

would have to observe on the order of picoseconds exposure time to view the instantaneous pattern. The period was set to 83.3 microns and the shear for one Talbot distance is 166.7 microns (Talbot distance = 2.720 cm). In this case the autocorrelation distance is only slightly larger than the shear width and the modulation of the fringes is substantially reduced. Fig 101.11 shows the Moire fringes after integration over 1,000 time constants. Integration over additional time constants would improve the smoothness of the Moire fringes. The level of contrast can be used to determine the spatial coherence of the beam.

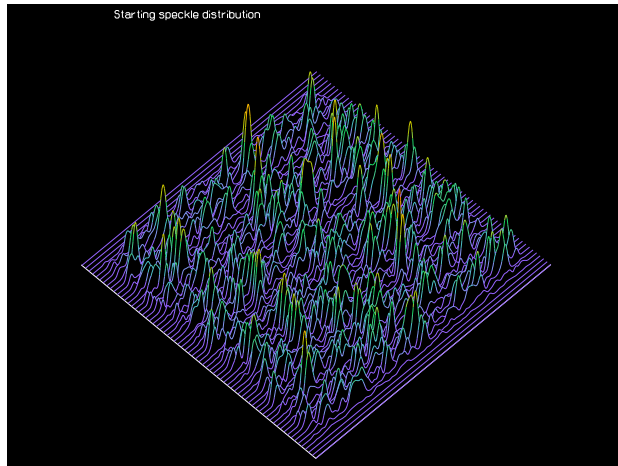


Fig. 101b.1. Instantaneous irradiance distribution for excimer beam with $M^2 = 25$.

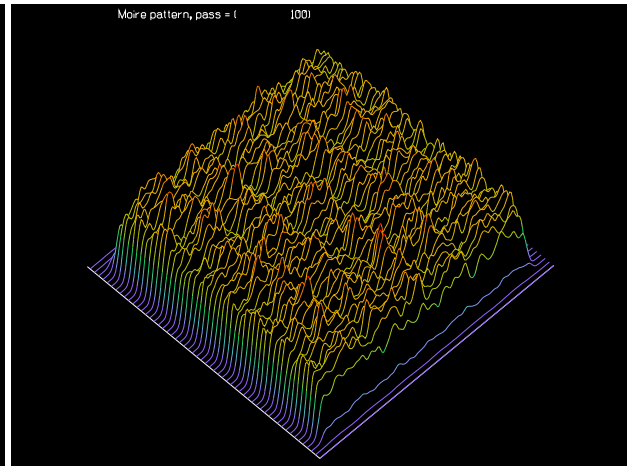


Fig. 101b.2. Moire pattern for excimer beam after integration over 1,00 time constants. Compare with Fig. 101b.1.

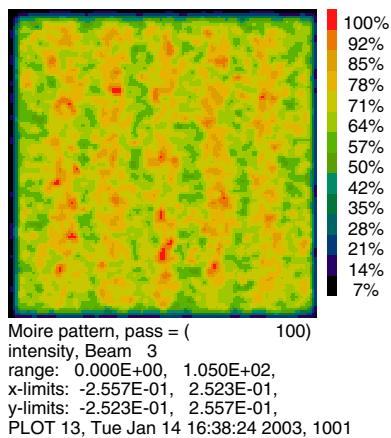


Fig. 101b.3. Profile of Moire fringes shown after integration over 100 time constants. The modulation is reduced showing partial coherence.

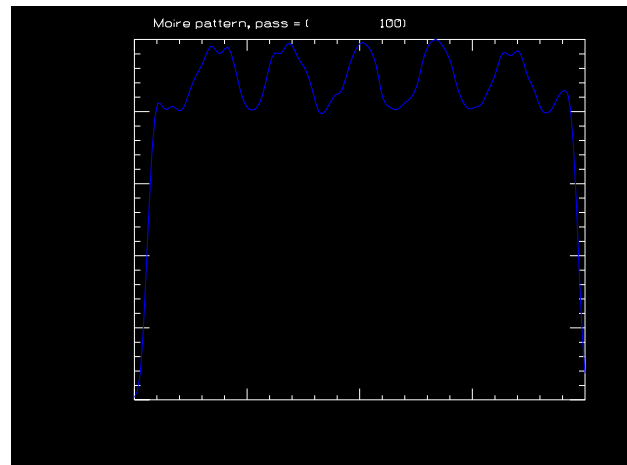


Fig. 101b.4. Results of slit scan. The pattern is integrated along the y-direction with the project command.

Input: ex101b.inp

```
c## ex101b
#
# Example: Crossed Ronchi rullings with small relative rotation
#
```

Jump to: [Commands](#), [Theory](#)

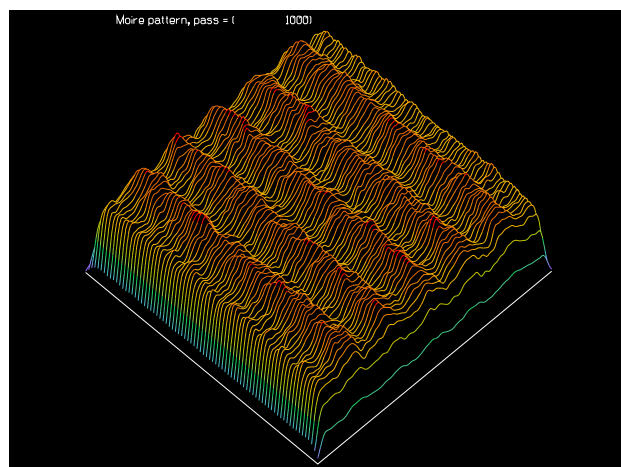


Fig. 101b.5. Integration over 1,000 cycles, showing relatively smooth fringes.

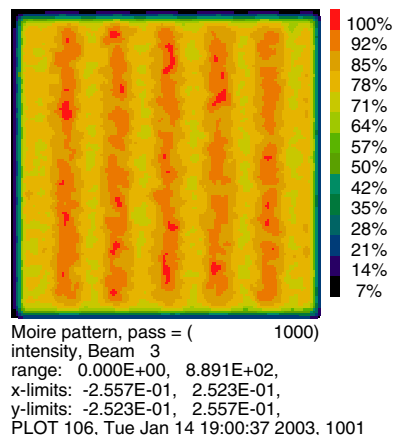


Fig. 101b.6.

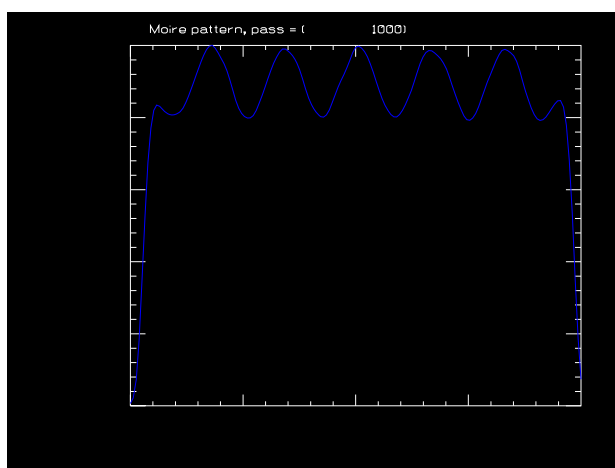


Fig. 101.3. Results of slit scan. The pattern is integrated along the y-direction. The depth of modulation indicates the coherence width.

```
set/cpu 2
mem/set/b 18
set/label/off
variab/dec/int pass Nline Count
Nline = 1024
array/s 1 Nline
array/s 2 256 256 data
array/s 3 256 256 data
array/s 4 256 1 data
clear 3 0
period = 1/120      #
Field = .85/2
Apt = .25
units/field 1 Field
units/field 2 Field
units/field 3 Field
units/field 4 Field
```

Jump to: [Commands](#), [Theory](#)

```

wavelength/set 0 .5106
c echo/on
macro/def step/o
    pass = pass + 1 list
    Count = Count + 1 list
    zreff/se 1 0
    array/s 1 Nline
    units/field 1 Field
    noise/smooth 1 1 .01
    split/s/in 1 Apt 20

    energy/norm 1 1
    if [pass==1] then
        title Starting speckle distribution
        plot/w ex101b_1.plt
        plot/l xrad=1.1*Apt ns=64 theta=40
    endif
    grat/c 1 pe=period az=90+2.864
    prop 2.720
    grat/c 1 pe=period az=90-2.864
    lens 1 100
    dist 100 1
    clap/s/c 1 .3
    dist -100
    copy/border 1 2 0 0 Nline/256 Nline/256
    add/inc/c 3 2
    if [Count==20] then
        plot/w ex101b_2.plt
        title Moire pattern, pass = @pass
        plot/l 3 xrad=1.1*Apt ns=64 theta=40
    endif
    if [Count==20] then
        project 3 4
        plot/w ex101b_3.plt
        title Moire pattern, pass = @pass
        plot/x/i 4
        Count = 0
    endif
macro/end
write/off
macro/run step/100
write/on
plot/w ex101b_4.plt
title Moire pattern, pass = @pass
plot/l 3 xrad=1.1*Apt ns=64 theta=40

plot/w ex101b_5.plt
title Moire pattern, pass = @pass
set/den 128 128
set/win/rel 20 80 20 80
plot/b/i/b 3

plot/w ex101b_6.plt
title Moire pattern, pass = @pass

```

Jump to: [Commands](#), [Theory](#)

```
plot/x/i 4

write/off
macro/run step/900
write/on
plot/w ex101b_7.plt
title Moire pattern, pass = @pass
plot/l 3 xrad=Apt ns=64 theta=40
plot/w ex101b_8.plt
title Moire pattern, pass = @pass
set/den 128 128
set/win/rel 20 80 20 80
plot/b/i/b 3

plot/w ex101b_9.plt
title Moire pattern, pass = @pass
plot/x/i 4
```


Ex102: Vector addition of beams to make lenslet array

Beams with plane reference surfaces may be summed with consideration of vector properties. This is useful for calculating interference patterns created by beams at large incidence angles. The beams are defined by the choice of polarization state $\pm E_x$, $\pm E_y$, and the polar angle over the range 15, 30, and 45 degrees.

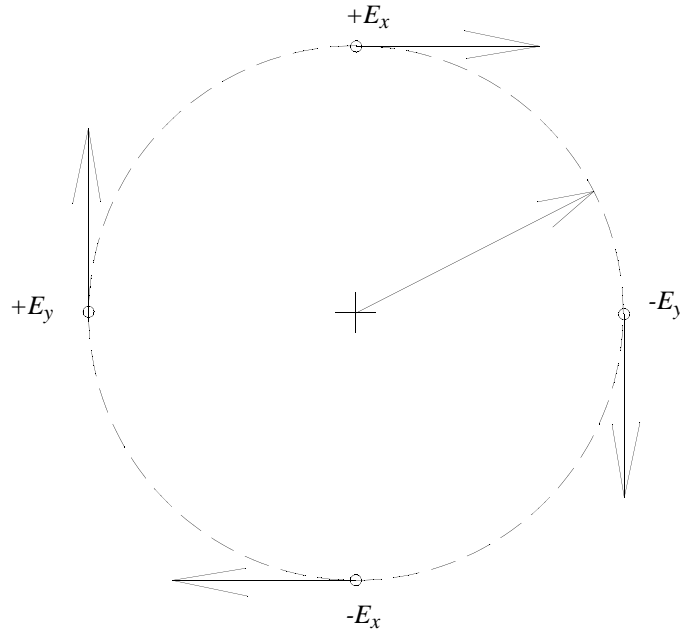


Fig. 102.1. Top view of the angular orientation of the four beams to be summed and the polar angle θ .

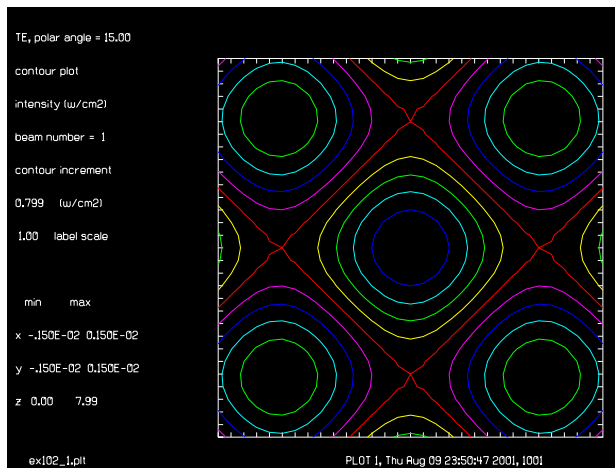


Fig. 102.2. Interference with polar angle 15 degrees.

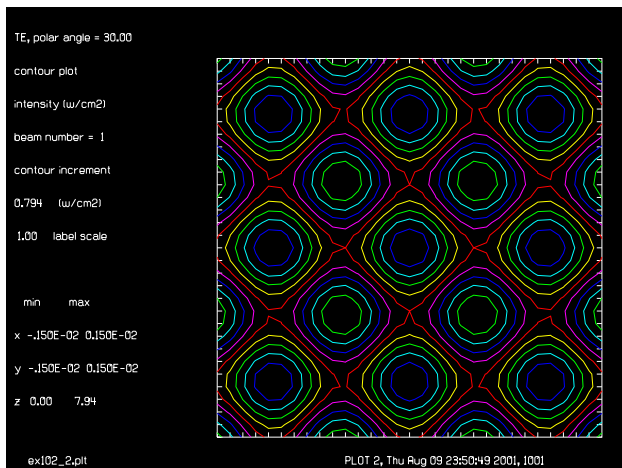


Fig. 102.3. Interference with polar angle 30 degrees.

Input: `ex102.inp`

```
c## ex102
#
```

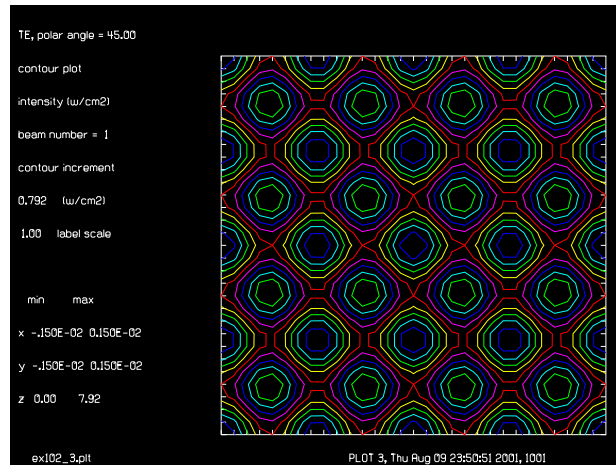


Fig. 102.4. Interference with polar angle 45 degrees.

```
# add four beams considering vector diffraction
# to form a lens-like array
#
# Polarization is in a box-like configuration with the polarization
# tangent to form a Swastika-like a pattern
#
# Unlike TM polarisation which works well only for 45 degree polar
# angle, TE polarization gives a good interference pattern for
# all polar angles.
#
# Beams      Description
# -----
# Beam 1     Ex and Ey of sum
# Beam 2     Ez of sum
# Beam 3     +Ex, 0 azimuth
# Beam 4     -Ey, 90 azimuth
# Beam 5     -Ex, 180 azimuth
# Beam 6     Ey, 270 azimuth
# Beam 7     temporary copy of 1
nbeam 6
Ang=0
variab/dec/int step
macro/def system/o
  step = step + 1
  Ang = Ang + 15
  array/reset 1
  array/reset 2
  array/reset 3
  array/reset 4
  array/reset 5
  array/reset 6
  units/s 0 .00005

# Beam 3      +Ex, 0 azimuth

global/oper 3 rx=Ang
```

Jump to: [Commands](#), [Theory](#)

```

# Beam 4      -Ey, 90 azimuth
jones/set ar=0 br=0 cr=-1 dr=0
jones/mult 4
global/oper 4 ry=-Ang

# Beam 5      -Ex, 180 azimuth
phase/piston 5 180
global/oper 5 rx=-Ang

# Beam 6      Ey, 270 azimuth
jones/set ar=0 br=0 cr=1 dr=0
jones/mult 6
global/oper 6 ry=Ang

clear 1 0
clear 2 0
add/vector 1 2 3 4 5 6
add/inc/c 1 2
plot/w ex102_@step.plt
title TE, polar angle = @Ang
set/den 32 32
plot/c 1 ilab=0
macro/end
array/s 1 64 64 1
array/s 2 64 64 0
array/s 3 64 64 1
array/s 4 64 64 1
array/s 5 64 64 1
array/s 6 64 64 1

macro system/3

```


Ex103: Circular and pentagonal reflecting wall waveguide

Table. 103.1. Table of Ex103 examples

Ex103a: Example of cylindrical rod analyzed by the method of images.	1
Ex103b: Circular rod, two reflecting walls	4
Ex103c: Circular rod, small memory model	7
Ex103d: Circular rod, large memory model	8
Ex103e: Pentagonal rod	9

The example illustrates finite sized, perfectly reflecting wall, waveguides of circular and pentagonal cross section. Metal wall waveguides and dielectric waveguides operating by TIR closely approximate perfectly walls. The method of images is used. We take advantage that the propagation of a beam in a perfectly reflecting wall waveguide is identical to propagation of a composite beam consisting of the beam in the waveguide and the infinite array of images produced by successive multiple reflections from the walls.

Various types of waveguides may be modeled with GLAD. Ex77 illustrates a rectangular reflecting wall waveguide. The technique used for analysis of reflecting wall waveguides takes advantage of the principle of aliasing. It is, in fact, a perfect solution, even more accurate than free space propagation but can only be applied to rectangular waveguides. Ex86 illustrates single and multiple mode dielectric waveguides using the beam propagation method (BPM). Waveguides of any cross section may be analyzed with the BPM method, however propagation must be done in small steps—generally less than a micron. The BPM method is ill suited to both large widths and long propagation distances.

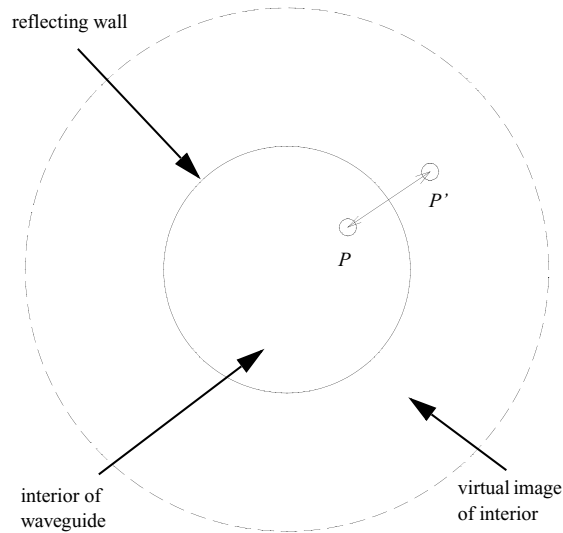


Fig. 103.1. A virtual image of the interior of the waveguide region is formed into an annular region. The point P is imaged into P' equidistant from the wall.

Ex103a: Example of cylindrical rod analyzed by the method of images

Figure 103.1 shows a circular waveguide in cross section. Any point P within the interior is imaged by the reflecting wall to a point P' . The intensity is reduced by the ratio R_p/R_p' .

The waveguide to be modeled has a radius of 0.4 cm. We assume a starting distribution to be a gaussian beam with radius of 0.5 cm. For short propagation distances, the Fresnel number is very large. With a 1024 x 1024 array, the center structure can be resolved at distances of about 20 cm. At that point the first center peak is observed and center peaks are observed there after at distances corresponding to odd Fresnel numbers.

In Ex103a only the first reflection is considered. Fig. 103a.1 shows the gaussian beam injected into the interior of the waveguide. The point of the reflecting wall is indicated in Figs. 103a.1 and 103a.2. The image of the interior appears in an annular ring. The image intensity is reduced by the ratio of the radius of the image to the radius of the source point. We assume a π phase change due to TIR reflection from a high index interior region surrounded by lower index material. The distribution is displayed at $z = 98$ cm in Figs. 103a.4 and 103a.5.

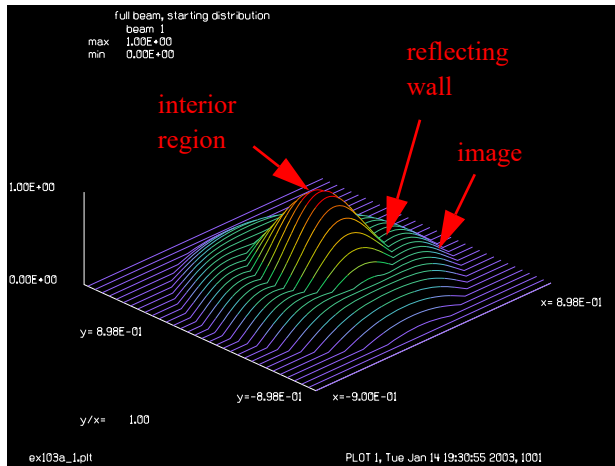


Fig. 103a.1. A gaussian distribution and its image formed by the reflecting wall (with reduced intensity).

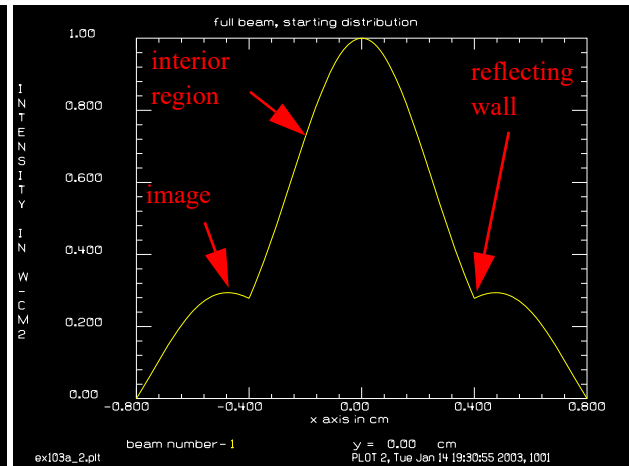


Fig. 103a.2. A gaussian distribution and its image formed by the reflecting wall (with reduced intensity).

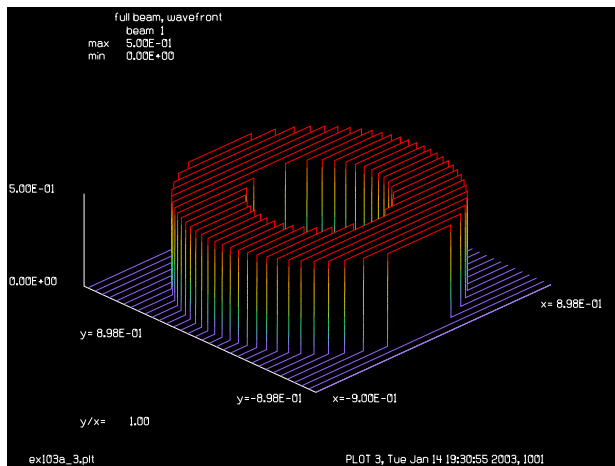


Fig. 103a.3. Phase of starting distribution. Grazing light reflected by TIR has a π phase change.

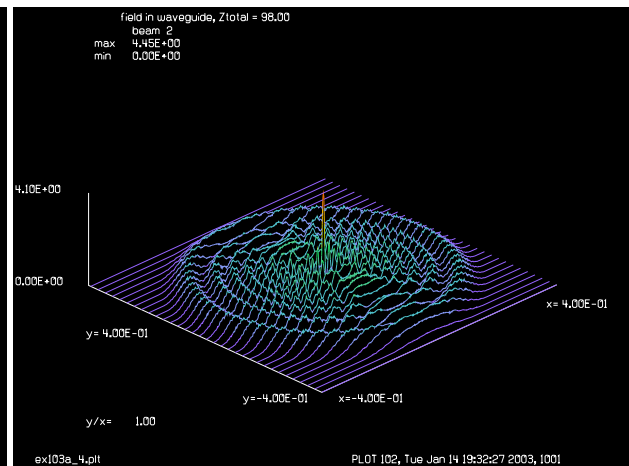


Fig. 103a.4. The cavity mode after propagating 98 cm. The center point appears at every odd Fresnel number.

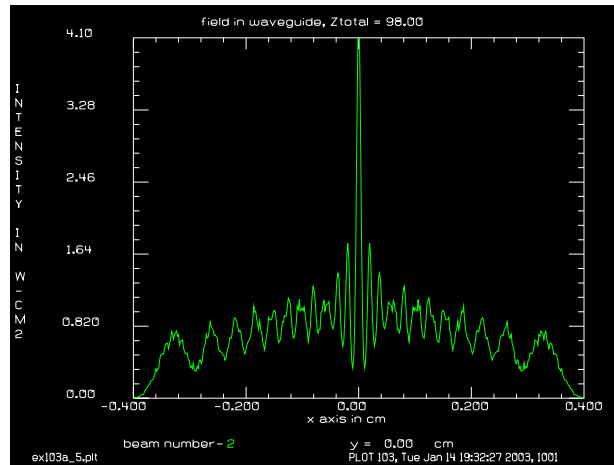


Fig. 103a.5. Profile of distribution at $z = 98$ cm. The beam peaks at odd Fresnel numbers and has a zero at even Fresnel numbers.

Input: `ex103a.inp`

```
c## ex103a
#
# Example of cylindrical rod analyzed by the method of images
#
# The command rod creates the image formed by a cylinder with perfectly
# reflecting walls. A single call to rod produces a single nearest-
# neighbor image corresponding to a single reflection from the walls.
# This limits the range of validity to a distance for which double
# reflections are not important.
#
# Multiple reflections may be demonstrated by using rod additional
# times, doubling the radius each time.
#
# This example illustrates construction of the first nearest
# neighbor image.
#
# For grazing incidence reflection there is a Pi phase change
# upon reflection from a high index material surrounded by
# a lower index material.
#
macro/def prop/o
# Propagate beam 1 and call displays
  prop Z
  Ztotal = Ztotal + Z      # calculate total position
macro display
macro/end
macro/def display/o
c
c Copy result to beam 2 to clip out inner circle representing
c waveguide mode
c
  copy/con 1 2              # copy to temporary beam
  clap/c/c 2 Radius        # apply aperture to see guide mode
```

Jump to: [Commands](#), [Theory](#)

```

plot/w ex103a_4.plt 10 310 400 300
title field in waveguide, Ztotal = @Ztotal
plot/l 2 xrad=Radius max=4.1          # plot guide mode
plot/w ex103a_5.plt 410 310 400 300
plot/x/i 2 le=-.4 ri=.4 fmax=4.1
macro/end
mem/set/b 18
array/s 1 1024
nbeam 2 512 data
units/field 1 .9                      # set units
variab/set Units 1 units
units/set 2 Units
wavelength/set 0 1.064                # set wavelength
Waist = .5                           # radius of starting gaussian
Radius = .4                           # radius of cylindrical wave guide
gaus/c/c 1 1 Waist
rod 1 Radius
plot/w ex103a_1.plt 10 10 400 300
title full beam, starting distribution
plot/l 1
plot/w ex103a_2.plt 410 10 400 300
plot/x/i le=-.8 ri=.8
plot/w ex103a_3.plt 810 10 400 300
title full beam, wavefront
plot/l/w 1
Index = 1.5
wavelength/set 1 1.064 Index
Z = 2
Ztotal = 0
macro display
macro prop/49

```

Ex103b: Circular rod, two reflecting walls

In Ex103b the same configuration is modeled with two reflections. The span of the second reflection goes out to a radius of 1.6, as shown in Figs. 103b.1 and 103b.2. To fit this distribution into a 1024 x 1024 array it is necessary to double the sample spacing. The courser sample spacing means that we must go out to about 44 cm before the center peak structure is sufficiently resolved to appear. Fig. 103b.3 shows the alternation of 0 and π phase change for the interior and 1st image and the two images of those regions, making four regions for the two-image case. For the propagation of $z = 98$ cm virtually no light scatters out of the 2nd image into the interior region as shown in Figs. 103b.4 and 103b.5.

Input: ex103b.inp

```

c## ex103b
#
# Example of cylindrical rod analyzed by the method of images
#
# The command rod creates the image formed by a cylinder with perfectly
# reflecting walls. A single call to rod produces a single nearest-
# neighbor image corresponding to a single reflection from the walls.

```

Jump to: [Commands](#), [Theory](#)

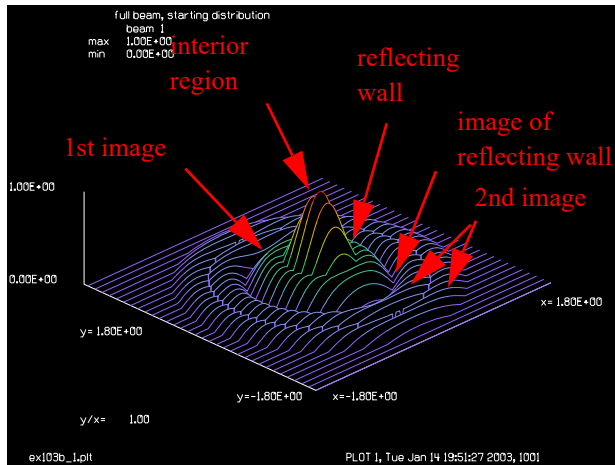


Fig. 103b.1. A gaussian distribution and 1st and 2nd images.

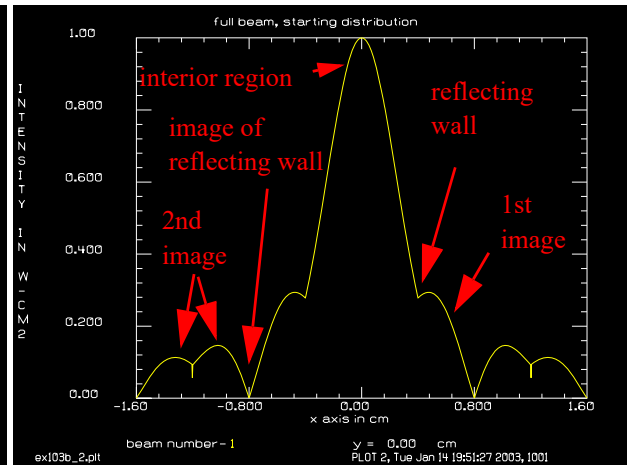


Fig. 103b.2. 1st and 2nd images by considering the reflecting wall and the image of the reflecting wall.

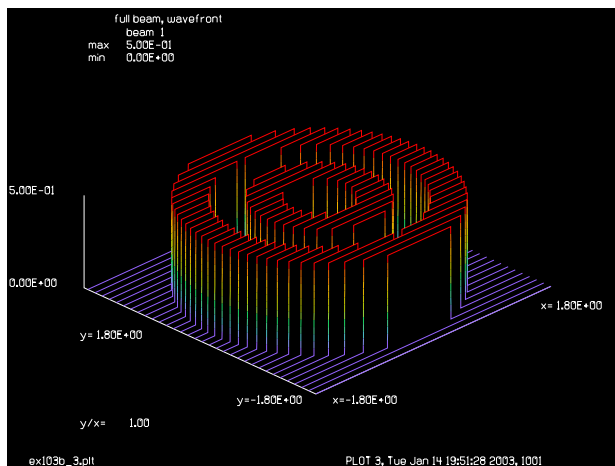


Fig. 103b.3. Phase of starting distribution. Phase are 0, interior of waveguide, π for first image, 0 and π for images of interior and 1st image, making four zones in all.

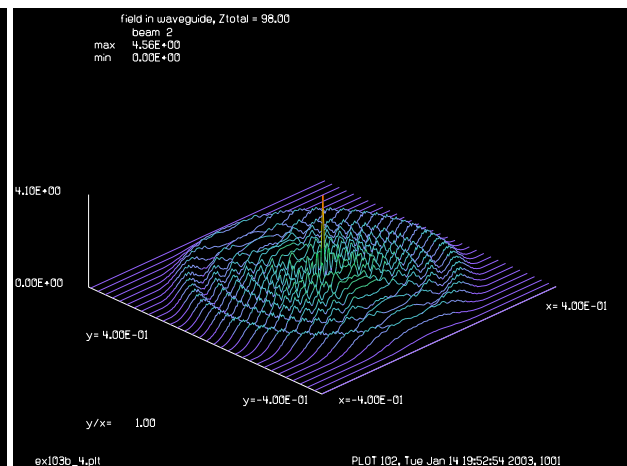


Fig. 103b.4. Distribution at $z=98$. The same as Fig. 103a.4. The use of the 2nd image does not change distribution.

```
# This limits the range of validity to a distance for which double
# reflections are not important.
#
# Multiple reflections may be demonstrated by using rod additional
# times, doubling the radius each time.
#
# This example illustrates construction of the first and second
# nearest neighbor image.
#
# For grazing incidence reflection there is a Pi phase change
# upon reflection from a high index material surrounded by
# a lower index material.
#
macro/def prop/o
c Propagate beam 1 and call display macro
```

Jump to: [Commands](#), [Theory](#)

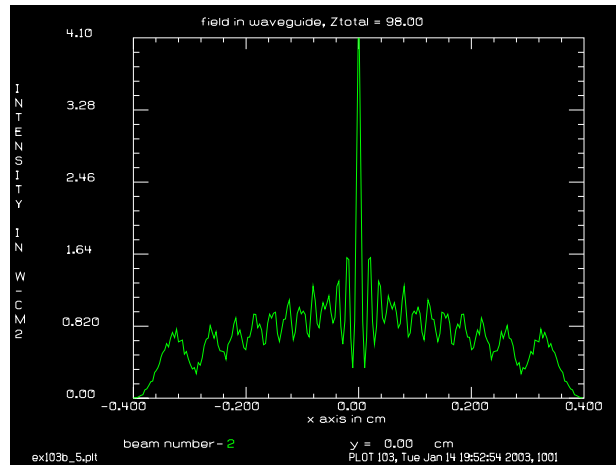


Fig. 103b.5. Distribution at $z=98$. The same as Fig. 103a.5. The use of the 2nd image does not change distribution.

```

prop Z
  Ztotal = Ztotal + Z          # calculate total position
  macro display
macro/end
macro/def display/o
c
c Copy result to beam 2 to clip out inner circle representing
c waveguide mode
c
  copy/con 1 2                  # copy to temporary beam
  clap/c/c 2 Radius            # apply aperture to see guide mode
  plot/w ex103b_4.plt 10 310 400 300
  title field in waveguide, Ztotal = @Ztotal
  plot/l 2 xrad=Radius max=4.1  # plot guide mode
  plot/w ex103b_5.plt 410 310 400 300
  plot/x/i 2 le=-.4 ri=.4 fmax=4.1
macro/end
mem/set/b 18
array/s 1 1024
nbeam 2 512 data
units/field 1 1.8              # set units
variab/set Units 1 units
units/set 2 Units
wavelength/set 0 1.064         # set wavelength
Waist = .5                     # radius of starting gaussian
Radius = .4                     # radius of cylindrical wave guide
gaus/c/c 1 1 Waist
rod 1 Radius
rod 1 Radius*2
plot/w ex103b_1.plt 10 10 400 300
title full beam, starting distribution
plot/l 1
plot/w ex103b_2.plt 410 10 400 300
plot/x/i le=-1.6 ri=1.6
plot/w ex103b_3.plt 810 10 400 300
title full beam, wavefront

```

```

plot/l/w 1
Index = 1.5
wavelength/set 1 1.064 Index
Z = 2
Ztotal = 0
macro display
macro prop/49

```

Ex103c: Circular rod, small memory model

The command rod creates the image formed by a cylinder with perfectly reflecting walls. A single call to rod produces a single nearest-neighbor image corresponding to a single reflection from the walls. This limits the range of validity to a distance for which double reflections are not important. Multiple reflections may be demonstrated by using rod additional times, doubling the radius each time. The command rod takes considerable time when there is not room for the full array plus scratch array totalling 64 MBytes. If this much memory is not available it is faster to use small memory to keep the disk IO down.

Input: ex103c.inp

```

c## ex103c
#
# Ex103c: cylindrical rod, memory utilization
#
# Example of cylindrical rod analyzed by the method of images
#
# The command rod creates the image formed by a cylinder with perfectly
# reflecting walls. A single call to rod produces a single nearest-
# neighbor image corresponding to a single reflection from the walls.
# This limits the range of validity to a distance for which double
# reflections are not important.
#
# Multiple reflections may be demonstrated by using rod additional
# times, doubling the radius each time
#
# The command rod takes considerable time when there is not room
# for the full array plus scratch array totalling 64 MBytes
# If this much memory is not available it is faster to use small
# memory to keep the disk IO down.
#
array/s 1 2048
nbeam 2 1024 data
units/field 1 1.5          # set units
variab/set Units 1 units
units/set 2 Units
wavelength/set 0 1.064     # set wavelength
Waist = .5                # radius of starting gaussian
Radius = .4                # radius of cylindrical wave guide
gaus/c/c 1 1 Waist
rod 1 Radius
plot/w ex103c_1.plt
title full beam, starting distribution

```

Jump to: [Commands](#), [Theory](#)

```

plot/l 1
plot/w ex103c_2.plt
title full beam, wavefront
plot/l/w 1
Index = 1.5
wavelength/set 1 1.064 Index
macro/def prop/o
    prop Z
    Ztotal = Ztotal + Z          # calculate total position
    copy/con 1 2                # copy to temporary beam
    clap/c/c 2 Radius          # apply aperture to see guide mode
    plot/w ex103c_3.plt
    title full field, Ztotal = @Ztotal
    plot/l 1                    # plot full beam
    plot/w ex103c_4.plt
    title field in waveguide, Ztotal = @Ztotal
    plot/l 2 xrad=Radius        # plot guide mode
    plot/w ex103c_5.plt
    plot/x/i 2 le=-.4 ri=.4
macro/end
Z = 5
macro prop/9

```

Ex103d: Circular rod, large memory model

The command rod creates the image formed by a cylinder with perfectly reflecting walls. A single call to rod produces a single nearest-neighbor image corresponding to a single reflection from the walls. This limits the range of validity to a distance for which double reflections are not important. Multiple reflections may be demonstrated by using rod additional times, doubling the radius each time. The command rod takes considerable time when there is not room for the full array plus scratch array totalling 64 MBytes. If this much memory is not available it is faster to use small memory to keep the disk IO down.

Input: ex103d.inp

```

c## ex103d
#
#
# Ex103d: Cylindrical rod, large memory model
#
# Example of cylindrical rod analyzed by the method of images
#
# The command rod creates the image formed by a cylinder with perfectly
# reflecting walls. A single call to rod produces a single nearest-
# neighbor image corresponding to a single reflection from the walls.
# This limits the range of validity to a distance for which double
# reflections are not important.
#
# Multiple reflections may be demonstrated by using rod additional
# times, doubling the radius each time
#
# Large memory version of Ex103c

```

Jump to: [Commands](#), [Theory](#)


```

#
mem/set/b 72
array/s 1 2048
nbeam 2 1024 data
units/field 1 1.5          # set units
variab/set Units 1 units
units/set 2 Units
wavelength/set 0 1.064     # set wavelength
Waist = .5                 # radius of starting gaussian
Radius = .4                # radius of cylindrical wave guide
gaus/c/c 1 1 Waist
rod 1 Radius
c mem/set/b 32
plot/w Ex103c_1.plt
title full beam, starting distribution
plot/l 1
plot/w Ex103c_2.plt
title full beam, wavefront
plot/l/w 1
Index = 1.5
wavelength/set 1 1.064 Index
macro/def prop/o
  prop Z
  Ztotal = Ztotal + Z      # calculate total position
  copy/con 1 2             # copy to temporary beam
  clap/c/c 2 Radius       # apply aperture to see guide mode
  plot/w Ex103c_3.plt
  title full field, Ztotal = @Ztotal
  plot/l 1                 # plot full beam
  plot/w Ex103c_4.plt
  title field in waveguide, Ztotal = @Ztotal
  plot/l 2 xrad=Radius     # plot guide mode
  plot/w plot5.plt
  plot/x/i 2 le=-.4 ri=.4
macro/end
Z = 5
macro prop/9

```

Ex103e: Pentagonal rod

The method of images may be applied to a rod with pentagonal cross section. We may use macros to construct the five nearest neighbor images.

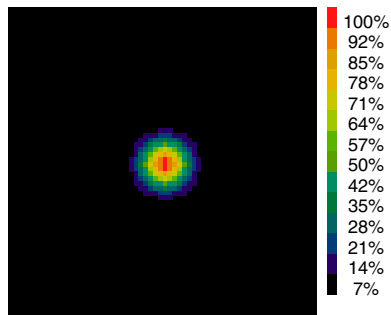
Input: ex103e.inp

```

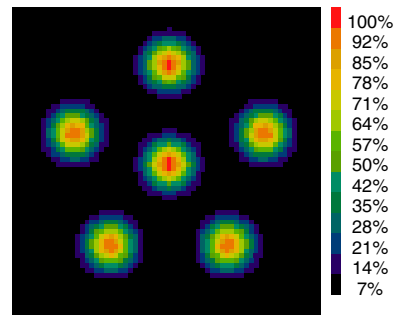
c## ex103e
#
# Examples: Pentagonal reflecting wall waveguide by the method of images
#
# Simulate the effects of reflections by constructing the nearest
# neighbors that would result
#

```

Jump to: [Commands](#), [Theory](#)



intensity, Beam 2
 range: 0.000E+00, 1.000E+00,
 x-limits: -1.000E+00, 1.000E+00,
 y-limits: -1.000E+00, 1.000E+00,
 PLOT 1, Tue Jan 14 21:15:54 2003, 1001



intensity, Beam 1
 range: 0.000E+00, 1.000E+00,
 x-limits: -1.000E+00, 1.000E+00,
 y-limits: -1.000E+00, 1.000E+00,
 PLOT 6, Tue Jan 14 21:16:13 2003, 1001

Fig. 103.3. Starting gaussian beam.

Fig. 103.4. After propagation through pentagonal mirror.
 The result in the center is similar to the result from
 propagating with four images of the center.

```
# It is assumed that the images are pi out of phase with respect
# to the waveguide mode.
#
# Beam      Description
# 1          propagating beam, center plus nearest neighbor images
# 2          temporary beam
# 3          temporary beam
#
mem/set/b 32
array/s 1 1024
set/den 64 64
set/win/abs -1 1 -1 1
nbeam 3 data
Radius = .4      # center to corner radius of pentagon
Waist = .5       # waist is .5 cm
Piston = 180     # phase of images relative to center
units/field 0 1.5
Waist = .2
gaus/c/c 2 1 Waist
abr/tilt 2 4 az=90
c obs/c 2 .1 xdec=.25
clap/pentagon/c 2 Radius # pentagonal aperture
plot/w ex103e_1.plt
plot/b/i/b 2
copy/con 2 1
phase/piston 2 Piston # set phase associated with reflection
Deltar = 2.*Radius*cos(36*pi/180) # radius to shift
Azdeg = 0
macro/def shift/o
    copy/con 2 3          # copy beam 2 to beam 3
    rotate 3 -Azdeg       # rotate to proper azimuthal angle
    flip/y 3              # flip vertical
    xdec= 0
    ydec = Deltar
```

```

    rescale/shift 3 xdec ydec rect # shift beam to decentered position
    rotate 3 Azdeg                # rotate to proper azimuthal angle
    add/coh/con 1 3               # add shifted, rotated image
    plot/w ex103e_2.plt
    plot/b/i/b 1                  # make a plot for entertainment
    Azdeg = Azdeg + 72            # increment azimuthal angle
macro/end
macro shift/5                    # run image construction macro
read/screen
wavelength/set 0 1.064
Ztot = 0.
#
# propagation is put in a macro so we can copy out
# the center hexagon to beam 2 yet leave beam 1 ready for
# more propagation steps
macro/def prop/o
    prop Z
    Ztotal = Ztotal + Z          # calculate total position
    copy/con 1 2                 # copy to temporary beam
    clap/pentagon/c 2 Radius     # apply aperture to see guide mode
    plot/w ex103e_3.plt
    title full field, Ztotal = @Ztotal
    plot/l 1                     # plot full beam
    plot/w ex103e_4.plt
    title field in waveguide, Ztotal = @Ztotal
    plot/l 2 xrad=Radius         # plot guide mode
macro/end
Z = 5
#
# You can propagate the distance more than once
#
macro prop/19

```


Ex104: Phase gratings: resolved and nonresolved

Table. 104.1. Table of Ex104 examples

Ex104a: Phase grating with <code>abr/lrip</code> , square wave and blazed, resolved model.	1
Ex104b: Phase grating with <code>grating/*/phase</code> , square wave and blazed, resolved model .	5
Ex104c: Cosine phase grating, comparison of resolved and nonresolved models.	7
Ex104d: Global grating with vertex tilt.	8
Ex104e: Global/ <code>grating</code> used with a global spherical mirror.	9
Ex104f: Including aberrations of the grating rulings.	10
Ex104g: Comparison of side lobe intensity for <code>global/grating</code> and resolved models. . .	11

GLAD can model a variety of amplitude (absorption) and phase gratings. The gratings may be resolved, so that there at least 6 to 8 sample points per grating line, or a nonresolved model may be used with the `grating/global` commands. The nonresolved model treats only a single order and calculates the efficiency including line shape and polarization effects. See Chap 11, GLAD Theory Manual for a discussion of the grating model.

Ex104a: Phase grating with `abr/lrip`, square wave and blazed, resolved model

A blazed grating consists of flat, sloping grating lines as indicated in Fig. 104.1. If the blazing angle of the wavefront matches the diffracted angle of a particular order most of the light will be scattered into that order. For a period of p and normal incidence, the diffracted angle is

$$\sin \theta = \frac{m\lambda}{p} \quad (104.1)$$

where m is the diffracted order.

The shape of the land creates relatively wide far-field pattern that is the Fourier transform of the land width and in the direction perpendicular to the wavefront. .

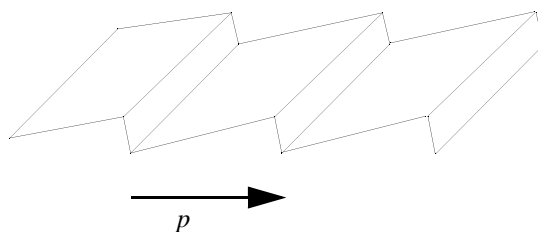


Fig. 104.1. A blazed grating consists of sloping lands to direct the diffracted energy into a particular order. The grating period p and slope angle should be matched for maximum effect.

Input: `ex104a.inp`

```
c## ex104a
#
# Example: Test of blazed phase grating using abr/lrip
```

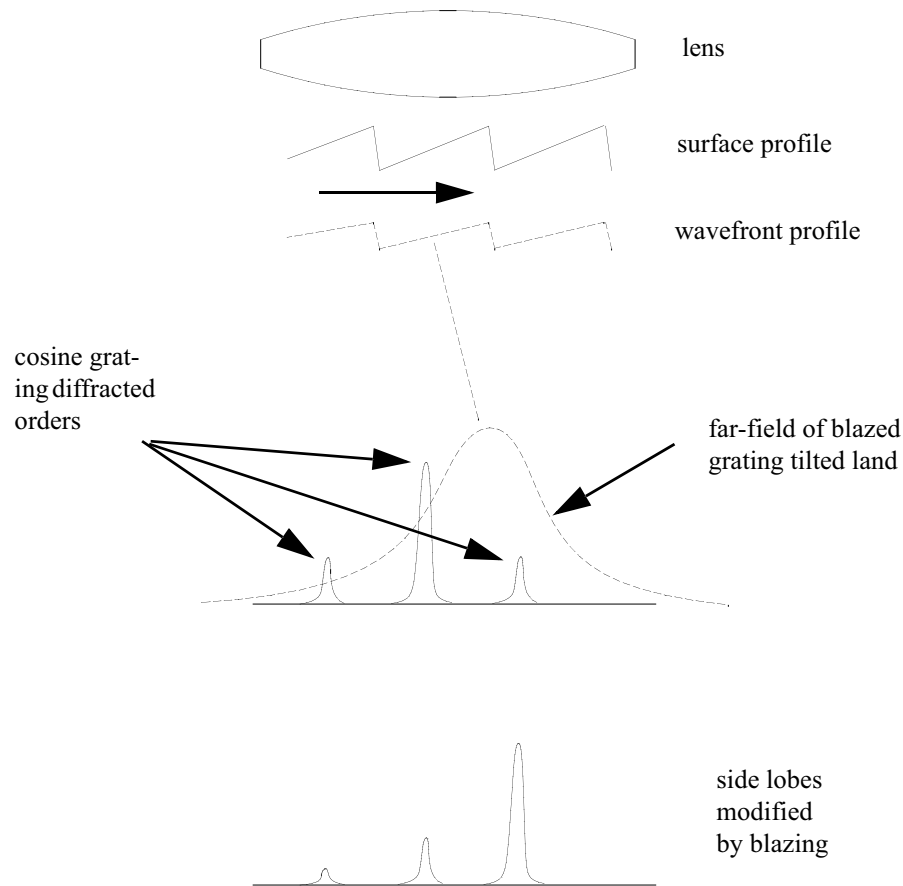


Fig. 104.2. A surface relief blazed grating creates a wavefront profile with $(n-1)$ times the surface relief. The far-field of the sloped land modifies the side lobe structure. The energy can be shifted to enhance a particular order.

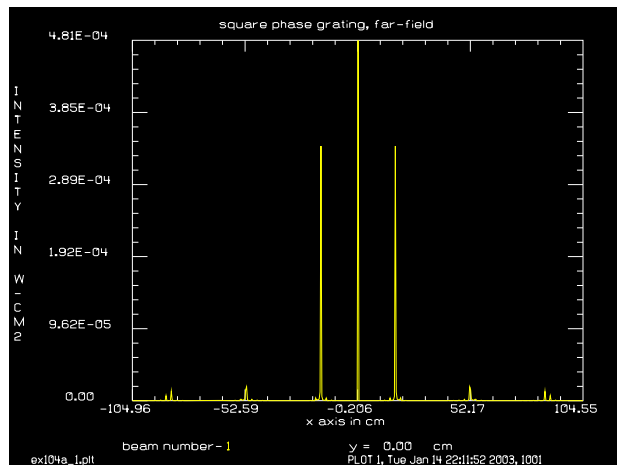


Fig. 104a.1. Far-field of cosine phase grating showing 0.1 wave of wavefront modulation. Note the two side lobes at $\pm 20^\circ$.

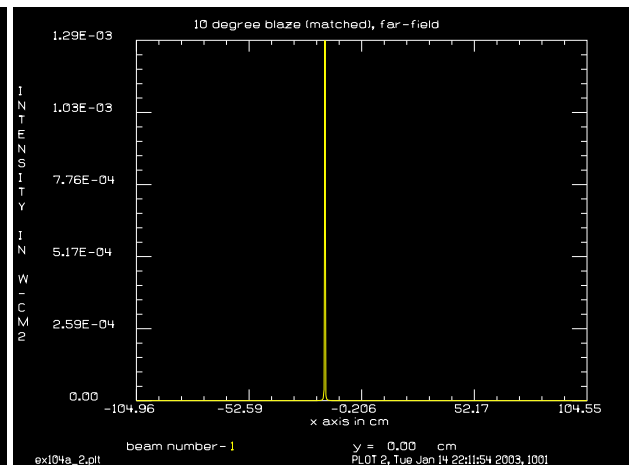


Fig. 104a.2. Far-field of blazed grating with angle of 10° —optimum for the first side lobe. The zero order is effectively completely suppressed.

#

Jump to: [Commands](#), [Theory](#)

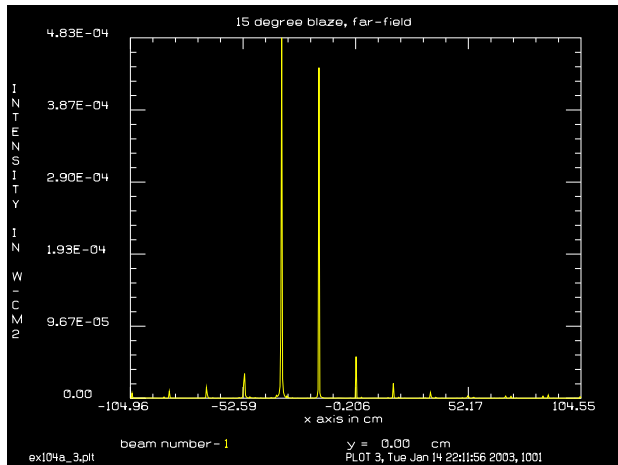


Fig. 104a.1. Far-field of blazed grating with angle of 15° . The blazing is 5° less than the optimum value of 20° . The result is the side lobe is less than in Fig. 104a.2 and some light is left in the zero order.

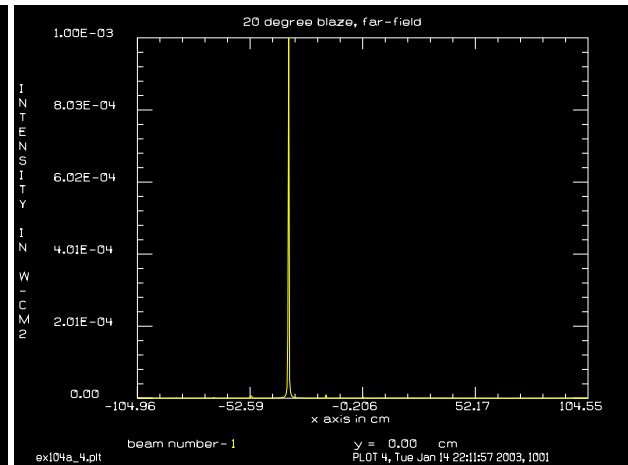


Fig. 104a.2. Far-field of blazed grating with angle of 20° —optimum for the second side lobes. The zero order is effectively completely suppressed.

```
echo/on
mem/set/b 2
array/s 1 512
Units = 3e-5
units/set 1 Units
Lambda = .6328e-4
Theta = 10 # diffraction angle in degrees
#
# Calculate period for desired angle of Theta
#
period = Lambda/sin(Theta*pi/180) # period for 20 deg. angle
wavelength/set 1 Lambda*10000
#
# square wave phase grating
#
abr/lrip/square 1 .3 1./period rnorm=1 azdeg=90
lens 1 100 #
prop 100
title square phase grating, far-field
plot/w ex104a_1.plt
plot/x/i 1
prop -100
#
# blazed grating with blazing set to match 10 diffraction angle
#
clear 1 1
prop 100
title 10 degree blaze (matched), far-field
plot/w ex104a_2.plt
plot/x/i 1
prop -100
#
# mismatched blazing angle, set to 15 degrees should be 10 degrees
```

```

#
clear 1 1
abr/lrip/blazed 1 15 1./period rnorm=1 azdeg=90
prop 100
title 15 degree blaze, far-field
plot/w ex104a_3.plt
plot/x/i 1
prop -100
#
# blazing set to second order at 20 degrees
#
clear 1 1
abr/lrip/blazed 1 20 1./period rnorm=1 azdeg=90
prop 100
title 20 degree blaze, far-field
plot/w ex104a_4.plt
plot/x/i 1
c## ex104a
#
# Example: Test of blazed phase grating using abr/lrip
#
mem/set/b 2
array/s 1 512
Units = 3e-5
units/set 1 Units
Lambda = .6328e-4
Theta = 10                                # diffraction angle in degrees
#
# Calculate period for desired angle of Theta
#
period = Lambda/sin(Theta*pi/180) # period for 20 deg. angle
wavelength 1 [Lambda*10000]
#
# square wave phase grating
#
abr/lrip/square 1 .3 [1./period] rnorm=1 azdeg=90
lens 1 100                                #
prop 100
title square phase grating, far-field
plot/w ex104a_1.plt
plot/x/i 1
prop -100
#
# blazed grating with blazing set to match 10 diffraction angle
#
clear 1 1
abr/lrip/blazed 1 10 [1./period] rnorm=1 azdeg=90
prop 100
title 10 degree blaze (matched), far-field
plot/w ex104a_2.plt
plot/x/i 1
prop -100
#
# mismatched blazing angle, set to 15 degrees should be 10 degrees

```

Jump to: [Commands](#), [Theory](#)


```
#
clear 1 1
abr/lrip/blazed 1 15 [1./period] rnorm=1 azdeg=90
prop 100
title 15 degree blaze, far-field
plot/w ex104a_3.plt
plot/x/i 1
prop -100
#
# blazing set to second order at 20 degrees
#
clear 1 1
abr/lrip/blazed 1 20 [1./period] rnorm=1 azdeg=90
prop 100
title 20 degree blaze, far-field
plot/w ex104a_4.plt
plot/x/i 1
```

Ex104b: Phase grating with `grating/* /phase`, square wave and blazed, resolved model

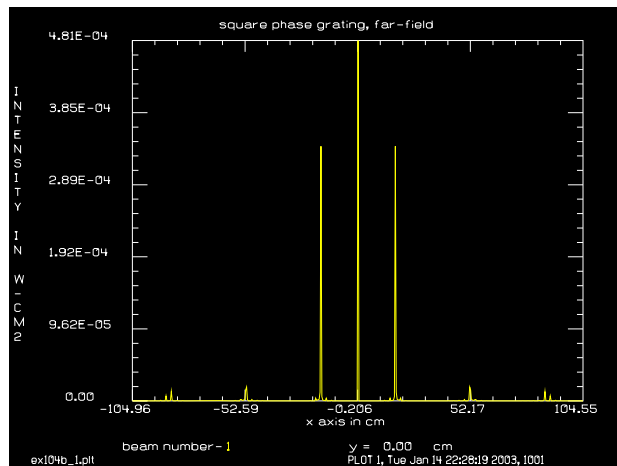


Fig. 104a.1. Far-field of cosine phase grating showing 0.1 wave of wavefront modulation. Note the two side lobes at $\pm 20^\circ$. Ex104b.

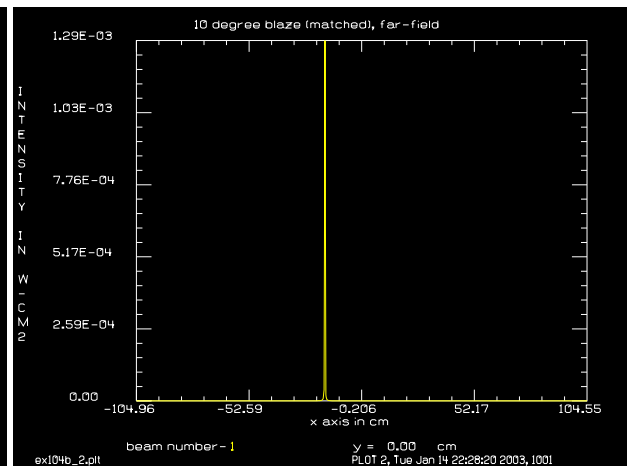


Fig. 104a.2. Far-field of blazed grating with angle of 10° —optimum for the first side lobe. The zero order is effectively completely suppressed. Ex104b.

Input: `ex104b.inp`

```
c## ex104b
#
# Example: Test of blazed phase grating using grating/ /phase commands
# (using grating/* /phase commands)
#
mem/set/b 2
array/s 1 512
Units = 3e-5
```

Jump to: [Commands](#), [Theory](#)

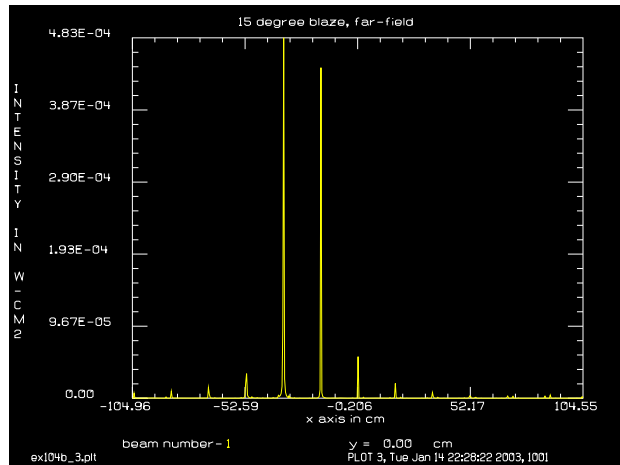


Fig. 104a.1. Far-field of blazed grating with angle of 15° . The blazing is 5° less than the optimum value of 20° . The result is the side lobe is less than in Fig. 104a.2 and some light is left in the zero order. Ex104b.

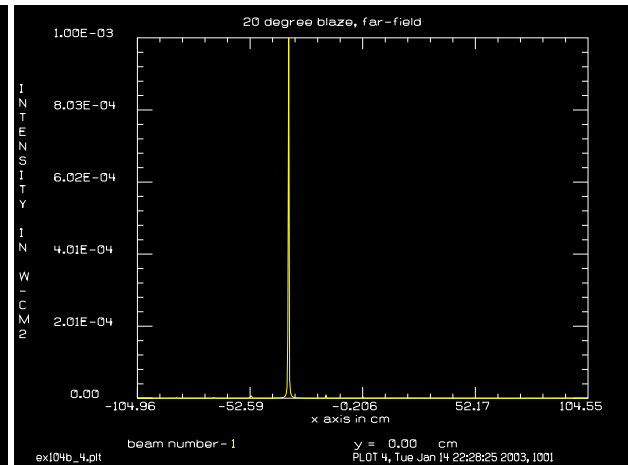


Fig. 104a.2. Far-field of blazed grating with angle of 20° —optimum for the second side lobes. The zero order is effectively completely suppressed. Ex104b.

```

units/set 1 Units
Lambda = .6328e-4
Theta = 10                                # diffraction angle in degrees
#
# Calculate period for desired angle of Theta
#
period = Lambda/sin(Theta*pi/180) # period for 20 deg. angle
wavelength/set 1 Lambda*10000
lens 1 100                                # lens to create far-field image
#
# square wave phase grating
#
grating/square/phase 1 .3 period azdeg=90
prop 100
title square phase grating, far-field
plot/w ex104b_1.plt
plot/x/i 1
prop -100
#
# blazed grating with blazing set to match 10 degree diffraction angle
#
clear 1 1
grating/blazed/phase 1 10 period azdeg=90
prop 100
title 10 degree blaze (matched), far-field
plot/w ex104b_2.plt
plot/x/i
prop -100
#
# mismatched blazing angle, set to 15 degrees should be 10 degrees
#
clear 1 1
grating/blazed/phase 1 15 period azdeg=90

```

Jump to: [Commands](#), [Theory](#)

```

prop 100
title 15 degree blaze, far-field
plot/w ex104b_3.plt
plot/x/i
prop -100
#
# blazing set to second order at 20 degrees
#
clear 1 1
grating/blazed/phase 1 20 period azdeg=90
prop 100
title 20 degree blaze, far-field
plot/w ex104b_4.plt
plot/x/i

```

Ex104c: Cosine phase grating, comparison of resolved and nonresolved models.

Input: ex104c.inp

```

c## ex104c
#
# Example: Test of blazed phase grating:
#           comparison of resolved and nonresolved models
#
# The grating/global series of command can model a grating at relatively
# large angles. The model calculates the efficiency on a single diffracted
# order.
#
variab/dec/int order Npoints
array/s 1 256 256 1
jones/set cr=1
jone/mult                                # put equal amplitude in both polarization
                                         # states
peak/norm 1 1                           # normalize peak intensity
Units = 5e-5
units/set 1 Units
Lambda = .6328e-4
wavelength/set 1 Lambda*10000
#
# global model of grating
#
vertex/rotate/add 0 0 0
AngleDeg = 10                           # diffraction angle desired for grating (deg)
order = 1                                # desired grating order
#
# Calculate period needed for diffraction angle
#
period = order*Lambda/sin(pi*abs(AngleDeg)/180.) list
ewav = .1                                # phase modulation in grating
#
# Call global grating model
#

```

Jump to: [Commands](#), [Theory](#)

```

grating/global/cosine/phase 1 ewav period order
global                # display ray direction after grating
# peak is about .088 by grating/global
peak 1
variab/set GlobalPeak 1 peak list # store peak value
pause 2
#
# Perform the same calculation with the resolved model
#
global/def 0 0 0        # reinitialize beam coordinate system
Npoints = 15           # number of sample points per grating line
units/set 1 period/Npoints
Apt = 100*period/Npoints # set aperture to 100 sample points
#
# Set up lens and propagation at distance L
# In the first step, calculate the peak of the far field (without the
# grating )
#
L = 1
lens 1 L
clear 1 1
clap/c/c 1 Apt
dist L
variab/set Peak 1 peak    # save pak of far field
dist -L
clear 1 1
clap/c/c 1 Apt           # trim beam to same aperture size
#
# Resolved model of a grating.
#
grating/cosine/phase 1 ewav period azdeg=90
dist L
mult 1 1./Peak          # normalize to peak (without grating)
plot/w ex104c.plt
title side lobes for cosine wave grating
plot/x/i fmax=.1
echo/on
# peak is about .088 by grating/global
GlobalPeak=

```

Ex104d: Global grating with vertex tilt

The output angle is shown to match analytical calculation.

Input: ex104d.inp

```

c## ex104d
#
# Example: Global grating with vertex tilt
#
variab/dec/int Beam order
array/s 1 128 128 1

```

Jump to: [Commands](#), [Theory](#)

```

jones/set cr=1
jone/mult
peak/norm 1 1
field
Units = 3e-5
units/set 1 Units
Lambda = .6328e-4
wavelength/set 1 Lambda*10000
vertex/rotate/add 30 0 0          # rotate the grating 30 deg about x-axis
variab/dec/int order
order = 1
AngleDeg = 10.
period = order*Lambda/sin(pi*abs(AngleDeg)/180.)
Beam = 1
#
# amplitude grating with ramp "sawtooth" form
#
grating/global/ramp Beam period order
ThetaOut = 180/pi*asin(order*Lambda/period + sin(30.*pi/180.)) -30 list
global
echo/on
# CosineOut agrees with cosine of k-vector z-cosine
CosineOut = cos(ThetaOut*pi/180) list

```

Ex104e: Global/grating used with a global spherical mirror

Input: ex104e.inp

```

c## ex104e
#
# Example: Global/grating used with a global spherical mirror
#
variab/dec/int Beam order
array/s 1 256
units/field 1 4
Lambda = .6328e-4
wavelength/set 1 Lambda*10000
clap/c/c 1 .5
#
# First, model the global mirror by itself
#
vertex/rotate/add 0 2 0          # rotate the grating 30 deg about x-axis
mirror/global -10 exact/rays
str
prop 5
plot/w ex104e_1.plt
title Far-field without grating showing coma
plot/l 1 1 xrad=.002
zreff/se 1 0
global/def
array/s 1 256
units/field 1 4

```

Jump to: [Commands](#), [Theory](#)

```

clap/c/c 1 .5
#
# The mirror and grating share the same vertex
#
vertex/rotate 0 2 0          # rotate the grating 30 deg about x-axis
mirror/global -10 exact/rays
order = 1
AngleDeg = 10.
period = order*Lambda/sin(pi*abs(AngleDeg)/180.)
Beam = 1
#
# amplitude grating with ramp "sawtooth" form
#
grating/global/ramp Beam period order
str
prop 5
plot/w ex104e_2.plt
title Far-field with grating showing coma
plot/l 1 1 xrad=.002
echo/on
#
# Note change in direction cosines due to grating diffraction
#
global

```

Ex104f: Including aberrations of the grating rulings

The ruling lines may have imperfections. These imperfections are multiplied by the grating order to create the aberration imposed on the beam.

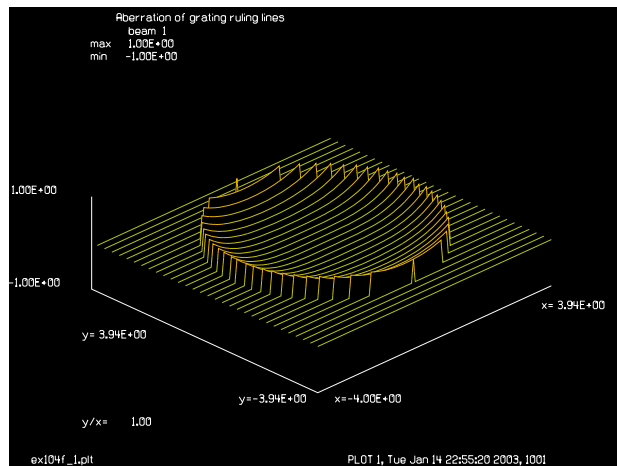


Fig. 104a.1. Aberration of grating lines.

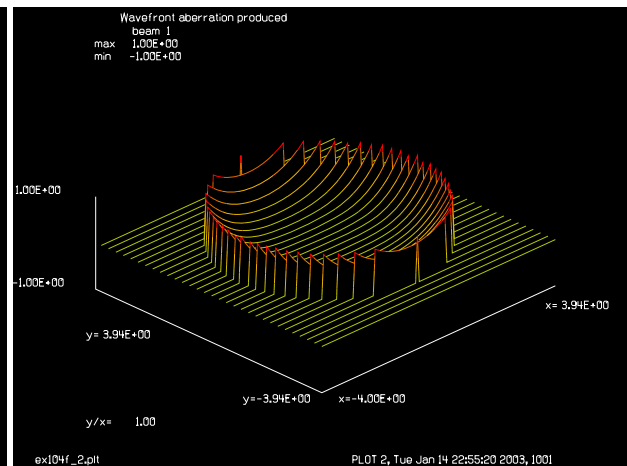


Fig. 104a.2. Wavefront aberration produced for order = 2.

Input: ex104f.inp

```

c## ex104f
#
# Example: Including aberrations of the grating rulings

```

Jump to: [Commands](#), [Theory](#)

```
#
# The ruling lines may have imperfections. These imperfections
# are multiplied by the grating order to create the aberration
# imposed on the beam.
#
variab/dec/int Beam order
array/s 1 128
units/field 1 4
Lambda = .6328e-4
wavelength/set 1 Lambda*10000
clap/c/c 1 3
abr/sph 1 .5
plot/w ex104f_1.plt
title Aberration of grating ruling lines
plot/l/w 1 min=-1 max=1
order = 2
mult/phase 1 order
plot/w ex104f_2.plt
title Wavefront aberration produced
plot/l/w 1 min=-1 max=1
```

Ex104g: Comparison of side lobe intensity for global/grating and resolved models

A comparison of the side lobes formed with the global/grating model and the regular resolved model. Linear polarization at 45 degrees is used for the global/grating model. TE polarization gives higher side lobe intensity than TM.

Input: ex104g.inp

```
c## ex104g
#
# Example: Comparison of intensity of diffracted order using global/grating
# model versus regular (resolved) model
#
array/s 1 128 128 1
jones/set cr=1 # set up for 45 deg. linear polarization
jone/mult
peak/norm 1 1 # normalize peak intensity
Units = 3e-5
units/set 1 Units
clap/c 1 Units*40
Lambda = .6328e-4
wavelength/set 1 Lambda*10000
vertex/rotate/add 0 0 0
variab/dec/int order
AngleDeg = 20
Order = 1
Period = Lambda/sin(pi*abs(AngleDeg)/180.) list
energy/norm 1 1
variab/set Peak1 1 peak
```

Jump to: [Commands](#), [Theory](#)

```

grating/global/cosine 1 period=Period order=Order 1
global
variab/set Peak2 1 peak
RelPeak1 = Peak2/Peak1 list
echo/on
C
C For grating/global, RelPeak1 = @RelPeak1
C
echo/off
clear 1 1
zreff 1 1
global/def 0 0 0
units/s 1 1
clap/c/c 1 40
c
c Find far-field peak and peak-normalize
c
lens 1 100
dist 100
peak/norm 1 1
dist -100
Period = 10
grating/cosine 1 Period 1 1
dist 100
plot/w ex104g_1.plt
title side lobes for cosine wave grating
plot/x/i fmax=.05
X = 100*Lambda/Period      # peak location
plot/w ex104g_2.plt
clap/c 1 2E-4 xdec=X
plot/x/i fmax=.05
variable/set RelPeak2 1 peak list
C
C grating/global: RelPeak1 = @RelPeak1, grating/cosine RelPeak2 = @RelPeak2
C

```


Ex105: Three-dimensional arrays

Table. 105.1. Table of Ex105 examples

Ex105a: Conversion between $N \times M \times 2$ and $N \times M$ polarized arrays	1
Ex105b: Transpose /xyz. Swaps y- and z-directions	3
Ex105c: /zxy (right circular) transpose followed by /yzx (left circular)	5
Ex105d: /yzx (left circular) transpose followed by /zxy (right circular)	7
Ex105e: /xzy transpose of non-cubic 3D array	10
Ex105f: Transpose of 3D array, left circular permutation	13
Ex105g: Transpose of 3D array, right circular permutation	17

This example illustrates some of the features of 3D arrays in GLAD. Originally GLAD was built to perform propagation with 2D arrays. The 2D arrays rectangular in the form of $N \times M$ pixels. 3D arrays have been added to better represent material volumes. The 3D arrays are organized into sheets of 2D arrays organized into axial sections in the form $N \times M \times L$. Each 2D section describes an x-y section. The ordinary 2D operations such as beam initialization, apertures, etc. may be applied to any of the 2D sections. The `set/section` command is provided to allow the user to set the control to the section of interest.

The two-sections arrays of $N \times M \times 2$, constitute a special case. These arrays are essentially identical in terms of memory structure with the $M \times N$ polarized arrays. The commands `array/convert/` `polarize` will change an $M \times N$ polarized into an $M \times N \times 2$ array.

The command `array/convert/unpolarize` will transform an $N \times M$ polarized array into an $N \times M \times 2$ array. Since the ordinary 2D tools may be applied to individual sections of the $N \times M \times 2$ array it may be easier to set up a beam with a complex polarization state as a two-section 3D and then convert to 2D polarized form. It will be necessary to set the 2D polarized array attribute to “beam” after transformation from the 3D form which always has the nonpropagating “data” attribute.

Ex105a: Conversion between $N \times M \times 2$ and $N \times M$ polarized arrays

We can use a $64 \times 64 \times 2$ array to form a complex state of polarization more readily than using the Jones calculus operations. Here we form a circular distribution shifted to the right for Section 1 and then shifted to the left for Section 2. We then convert the array to 64×64 polarized with the polarization state shown in Fig 105.1. Next we convert back to the original two-section form and display Section 1 (Fig. 105.2) and Section 2 (Fig. 105.3) for the s- and p-polarization states respectively.

Input: ex105a.inp

```
c## ex105a
#
# Example: demonstrate how to convert between
#          N x M X 2 and N x M polarized beams
#
array/s/3d 1 64 64 2      # make a 3D beam, dimensions: 64 x 64 x 2
#
# a 3D array can be addressed in terms of 2D sections
# The usual 2D operations can be applied to any section.
# Specify the section to be addressed by changing the section number.
#
```

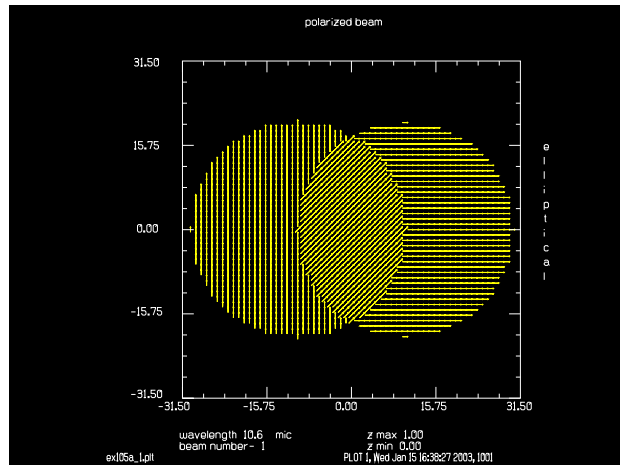


Fig. 105.1. Separated s-polarization (right) and p-polarization (left). Created with two sections and then converting to polarization array.

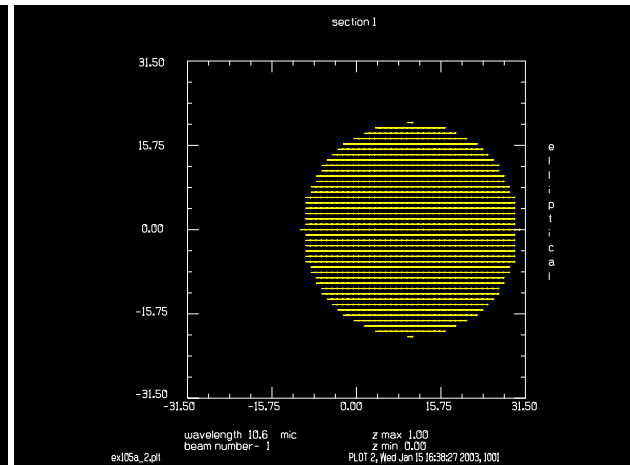


Fig. 105.2. Section 1 reconstructed from s-polarization.

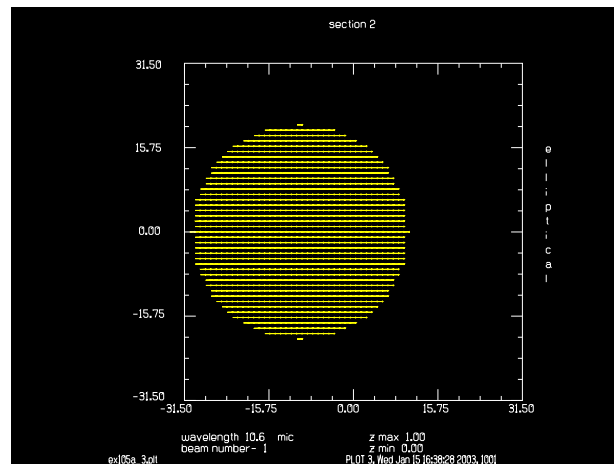


Fig. 105.3. Section 2 reconstructed from p-polarization.

```

set/section 1 1          # set the section number to 1
clap/c/c 1 20 x = 10     # make an aperture in section 1, decenter to +10
set/section 1 2          # set the section number to 2
clear 1 1                # set the beam to 1's
clap/c/c 1 20 x = -10    # make an aperture in section 2, decenter to -10
array/convert/polarize 1 # convert N x N x 2 3D array to polarized beam
plot/w ex105a_1.plt
title polarized beam
plot/e 1                 # plot elliptical polarization
#
# Observe decentered s- and p-polarizations
array/convert/unpolarize 1 # convert back N x N x 2, 3D array
set/section 1 1          # set section number to 1
plot/w ex105a_2.plt
title section 1
plot/e 1                 # plot polarization of section 1
set/section 1 2

```

Jump to: [Commands](#), [Theory](#)

```

plot/w ex105a_3.plt
title section 2
plot/e 1                                # plot polarization of section 2

```

Ex105b: Transpose /xyz. Swaps y- and z-directions

We can define an ellipsoidal volume using a macro to define an aperture which scales according to the z-position. In this case the ellipsoidal volume has the radial values (11.25, 3.75, 7.5). So we have the following sections: x-y section (11.25, 3.75), x-z section (11.25, 7.5), y-z section (3.75, 7.5).

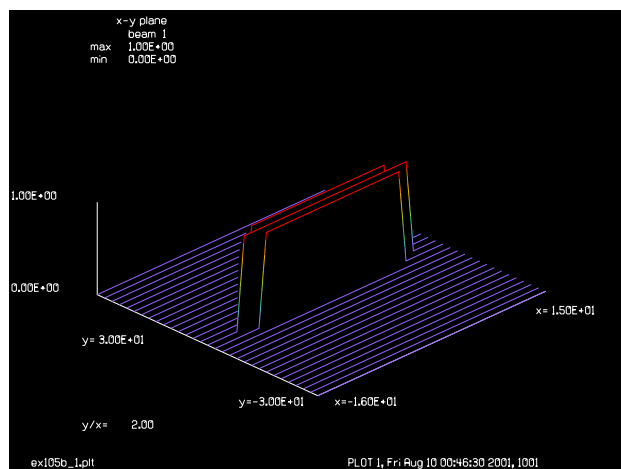


Fig. 105.4. A display of the X-Y section (11.25, 3.75).

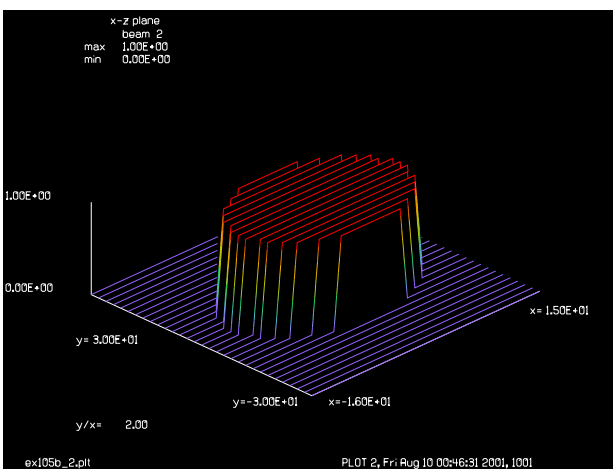


Fig. 105.5. X-Z section (11.25, 7.5) created with the DISPLAY macro that copies rows from the sections into Beam 2 to create a 2D array corresponding to the x-z section.

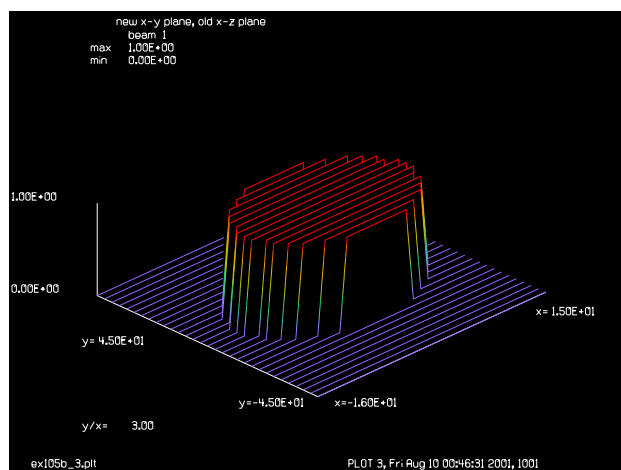


Fig. 105.6. A display of the X-Y section after the XZY transpose. The X-Y section is now identical to the original X-Z section. The x-axis is unchanged but the y-direction now has the values of the original z-axis. Section is (11.25, 7.5).

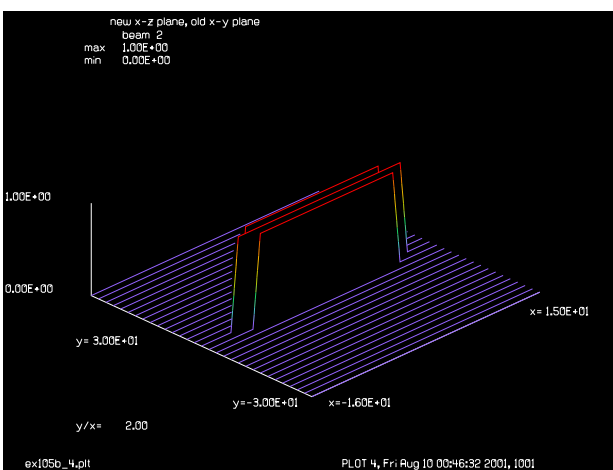


Fig. 105.7. X-Z section after XZY transpose. The X-Z section is now identical to the original X-Y section. The x-axis is unchanged but the z-direction now has the values of the original y-axis. Section is (11.25, 3.75).

Input: ex105b.inp

```

c## ex105b
#
# Examples: Illustrate XZY transpose, flips Y- and Z-axes
#
mem/set/b 1
array/s/3d 1 32 32 32 0
units/set 1 1 2 3
clear -1 0
nbeam 2 32 32
set/section 1 1
variab/dec/int i j k section row
radius = 7.5
section = 0
macro/def sections/o
# macro to scan through all sections and add aperture
# in x-y plane of variable size as a function of z position
    section = section + 1
    set/section 1 section
    z = section - 17
    if [abs(z)<radius] then
        apt = sqrt(radius^2 - z^2) list
        clap/e/c 1 1.5*apt .5*apt
    else
        clear 1 0.
    endif
macro/end
macro sections/32
macro/def display1/o
# copy center array to temporary array for later display
    section = section + 1
    row = section
    set/section 1 section
    copy/row 1 2 17 row
macro/end
macro/def display/o
    section = 0
    clear 2 0
    macro display1/32
    plot/l 2
macro/end
set/sect 1 17
plot/w ex105b_1.plt 10 10 400 300
title x-y plane
plot/l 1
#
# display the aperture in the x-z plane
#
title aperture defined in x-z plane
plot/w ex105b_2.plt 410 10 400 300
title x-z plane
macro display
echo/on

```

Jump to: [Commands](#), [Theory](#)

```

tran/xzy 1
plot/w ex105b_3.plt 10 310 400 300
title new x-y plane, old x-z plane
set/sect 1 17
plot/l 1
plot/w ex105b_4.plt 410 310 400 300
title new x-z plane, old x-y plane
macro display

```

Ex105c: /xzy (right circular) transpose followed by /yzx (left circular)

We can perform a full 3D transpose. The ZXY transpose exchanges the y- and z- axes and then the x- and y-axes. This is a right circular permutation of the XYZ system. The (11.25, 3.75, 7.5) ellipsoid is transformed into an (7.5, 11.25, 3.75) ellipsoid. The ZXY transpose may be reversed by a YXZ transposes, left circular from the XYZ system.

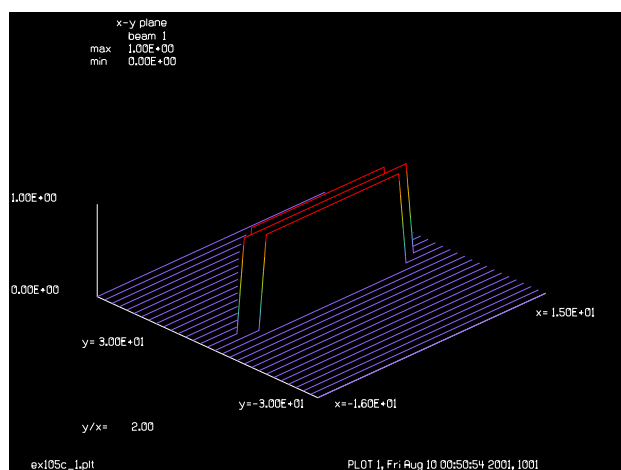


Fig. 105.8. X-Y section of the start (11.25, 3.75).

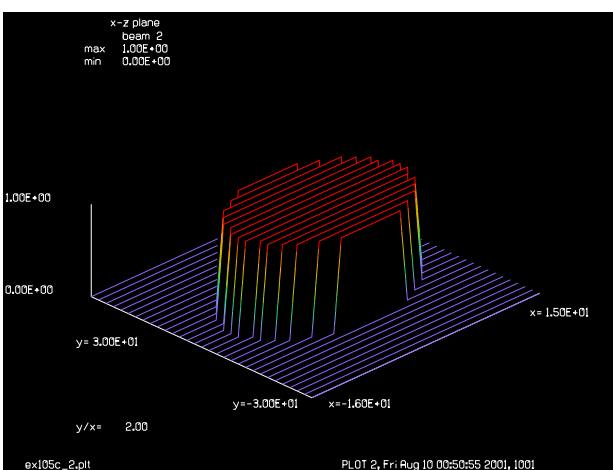


Fig. 105.9. X-Z section (11.25, 7.5) created with the DISPLAY macro.

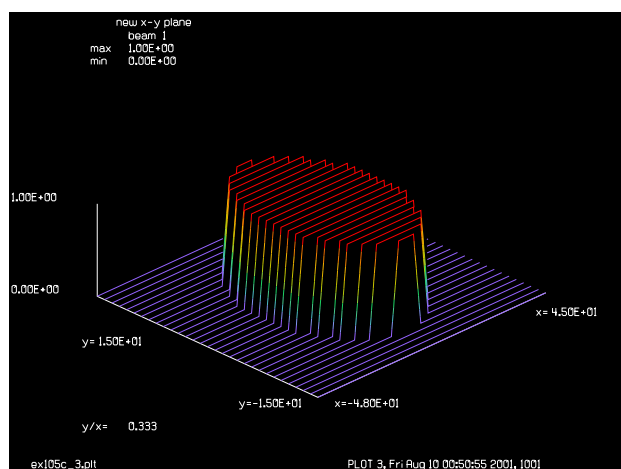


Fig. 105.10. X-Y section after ZXY transpose. Section is (7.5, 11.25).

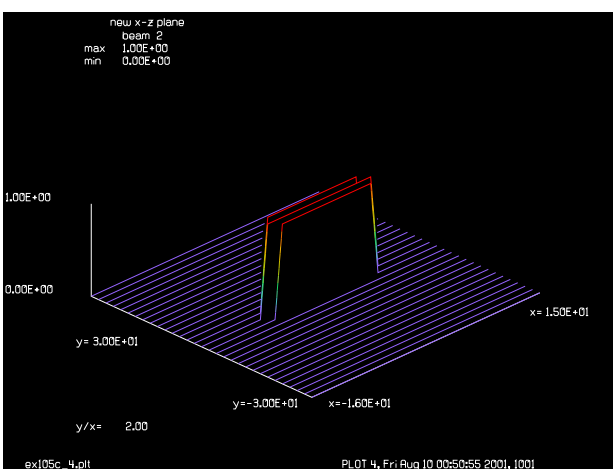


Fig. 105.11. X-Z section after ZXY transpose. Section is (11.25, 3.75).

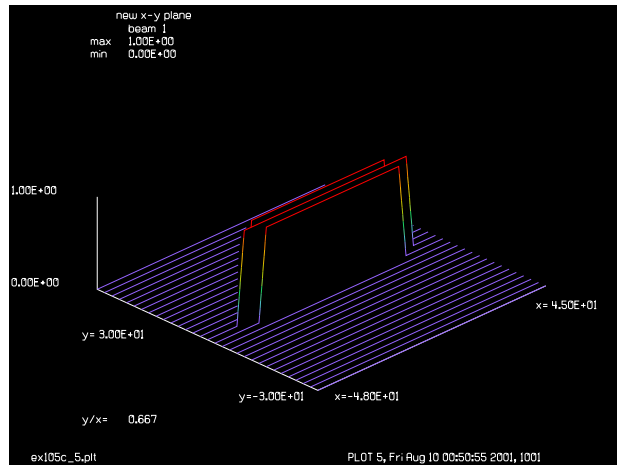


Fig. 105.12. X-Y section after the a ZXY followed by YZX transpose. Restores original distribution.

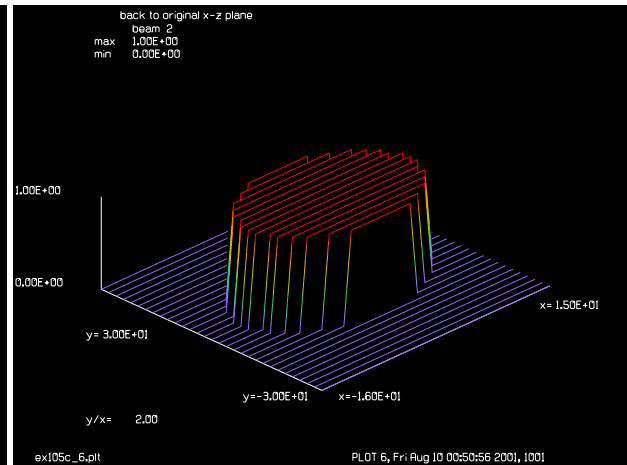


Fig. 105.13. X-Z section after the a ZXY followed by YZX transpose. Restores original distribution.

Input: ex105c.inp

```
c## ex105c
#
# Example: ZXY (right circular) transpose followed by YZX (left circular)
#
# The ZXY transpose changes the Y- and Z-axes (XYZ => XZY)
# and then the X- and Y-axes (XZY => ZXY) to do a full
# 3D transpose. The letters XYZ are right circularly permuted to ZXY
#
# The YZX transpose is a left circularly permutation. It will reverse
# the right circular transpose ZXY.
#
mem/set/b 1
array/s/3d 1 32 32 32 0
units/s 1 1 2 3
clear -1 0
nbeam 2 32 32
set/section 1 1
variab/dec/int i j k section row
radius = 7.5
section = 0
macro/def sections/o
# macro to scan through all sections and add aperture
# in x-y plane of variable size as a function of z position
    section = section + 1
    set/section 1 section
    z = section - 17
    if [abs(z)<radius] then
        apt = sqrt(radius^2 - z^2) list
        clap/e/c 1 1.5*apt .5*apt
    else
        clear 1 0.
    endif
macro/end
```

Jump to: [Commands](#), [Theory](#)

```

macro sections/32
macro/def display1/o
# copy center array to temporary array for later display
  section = section + 1
  row = section
  set/section 1 section
  copy/row 1 2 17 row
macro/end
macro/def display/o
  section = 0
  clear 2 0
  macro display1/32
  plot/l 2
macro/end
#
# display initial x-y plane
#
set/sect 1 17
plot/w ex105c_1.plt
title x-y plane
plot/l 1
#
# display the aperture in the x-z plane
#
title aperture defined in x-z plane
plot/w ex105c_2.plt
title x-z plane
macro display
tran/zxy 1          # ZXY transpose permutes all axes
plot/w ex105c_3.plt
title new x-y plane
set/sect 1 17
plot/l 1
plot/w ex105c_4.plt
title new x-z plane
macro display
tran/yzx 1          # YZX transpose reverses ZXY transpose
set/sect 1 17
plot/w ex105c_5.plt
title new x-y plane
plot/l 1
plot/w ex105c_6.plt
title back to original x-z plane
macro display

```

Ex105d: /yzx (left circular) transpose followed by /zxy (right circular)

We can perform a full 3D transpose. The YZX transpose exchanges the x- and y- axes and then the x- and z-axes. This is a left circular permutation of the XYZ system. The (11.25, 3.75, 7.5) ellipsoid is transformed into an (3.75, 7.5, 11.25) ellipsoid. The YZX transpose may be reversed by a ZXY transpose, right circular from the XYZ system.

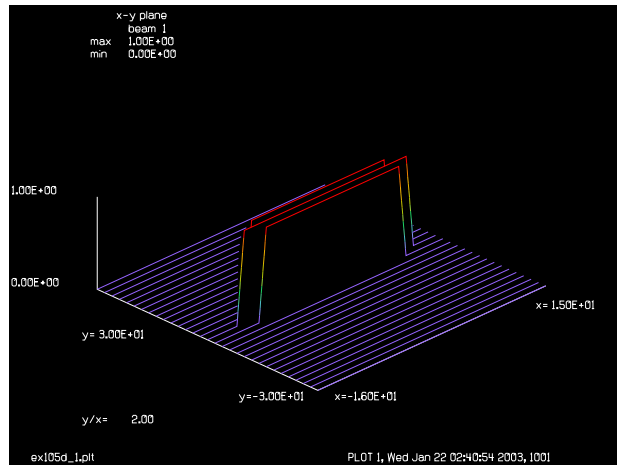


Fig. 105.14. X-Y section of the start (11.25, 3.75).

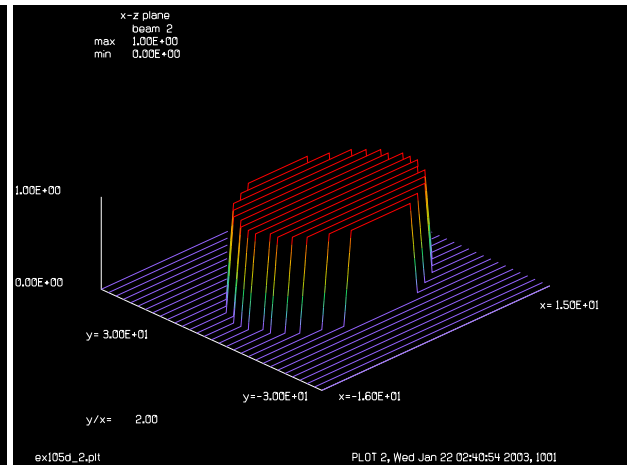


Fig. 105.15. X-Z section (11.25, 7.5) created with the DISPLAY macro.

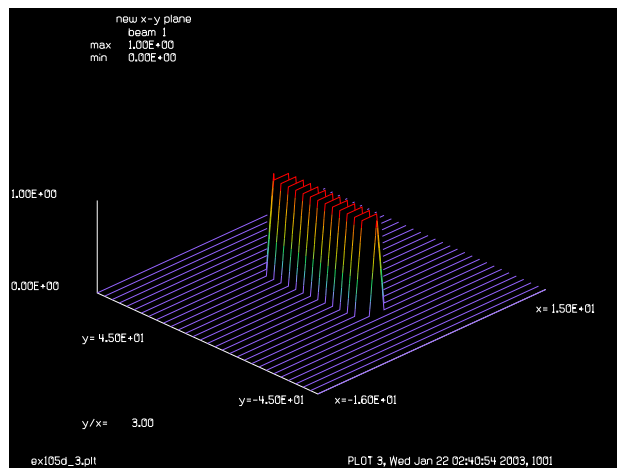


Fig. 105.16. X-Y section after YZX transpose. Section is (3.75, 7.5).

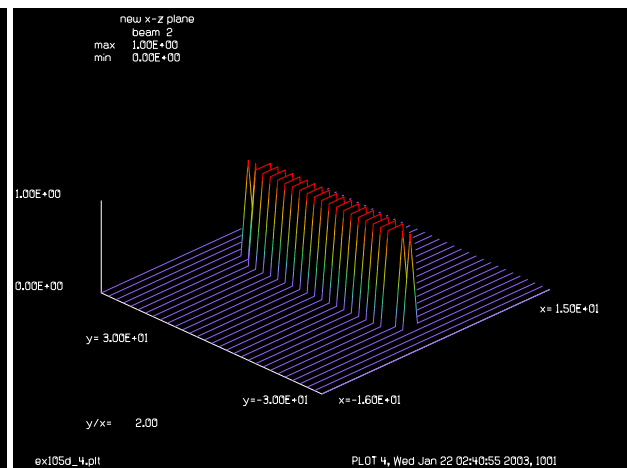


Fig. 105.17. X-Z section after ZXY transpose. Section is (3.75, 11.25).

Input: ex105d.inp

```

c## ex105d
#
# Example: YZX (left circular) transpose followed by ZXY (right circular)
#
# The YZX transpose changes the X- and Y-axes (XYZ => YXZ)
# and then the Y- and Z-axes (YXZ => YZX) to do a full
# 3D transpose. The letters XYZ are left circularly permuted to YZX.
#
# The ZXY transpose is a right circularly permutation. It will reverse
# the left circular transpose YZX.
#
mem/set/b 1
array/s/3d 1 32 32 32 0
units/s 1 1 2 3

```

Jump to: [Commands](#), [Theory](#)

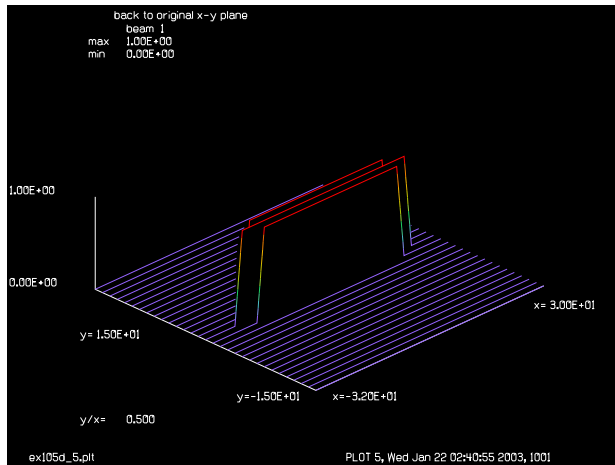


Fig. 105.18. X-Y section after the a YZX followed by ZXY transpose. Restores original distribution.

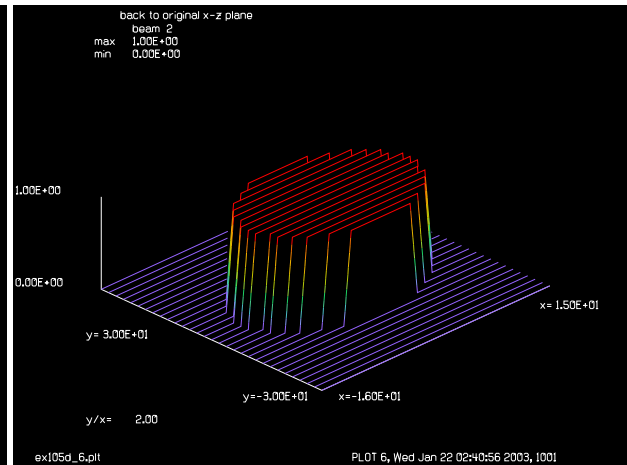


Fig. 105.19. X-Z section after the a YZX followed by ZXY transpose. Restores original distribution.

```
clear -1 0
nbeam 2 32 32
set/section 1 1
variab/dec/int i j k section row
radius = 7.5
section = 0
macro/def sections/o
# macro to scan through all sections and add aperture
# in x-y plane of variable size as a function of z position
    section = section + 1
    set/section 1 section
    z = section - 17
    if [abs(z)<radius] then
        apt = sqrt(radius^2 - z^2) list
        clap/e/c 1 1.5*apt .5*apt
    else
        clear 1 0.
    endif
macro/end
macro sections/32
macro/def display1/o
# copy center array to temporary array for later display
    section = section + 1
    row = section
    set/section 1 section
    copy/row 1 2 17 row
macro/end
macro/def display/o
    section = 0
    clear 2 0
    macro display1/32
    plot/1 2
macro/end
#
# display initial x-y plane
```

Jump to: [Commands](#), [Theory](#)

```

#
set/sect 1 17
plot/w ex105d_1.plt
title x-y plane
plot/l 1
#
# display the aperture in the x-z plane
#
title aperture defined in x-z plane
plot/w ex105d_2.plt
title x-z plane
macro display
tran/yzx 1          # YZX transpose permutes all axes
plot/w ex105d_3.plt
title new x-y plane
set/sect 1 17
plot/l 1
plot/w ex105d_4.plt
title new x-z plane
macro display
tran/zxy 1          # ZXY transpose reverses ZXY transpose
set/sect 1 17
plot/w ex105d_5.plt
title back to original x-y plane
plot/l 1
plot/w ex105d_6.plt
title back to original x-z plane
macro display

```

Ex105e: /xzy transpose of non-cubic 3D array

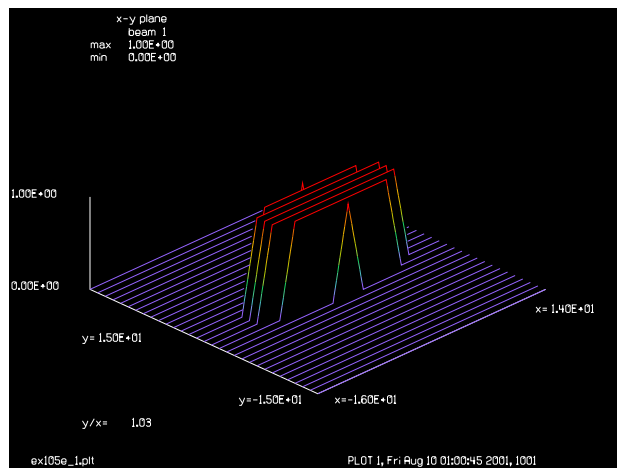


Fig. 105.20. A display of the X-Y section (11.25, 3.75). Non-cubic array.

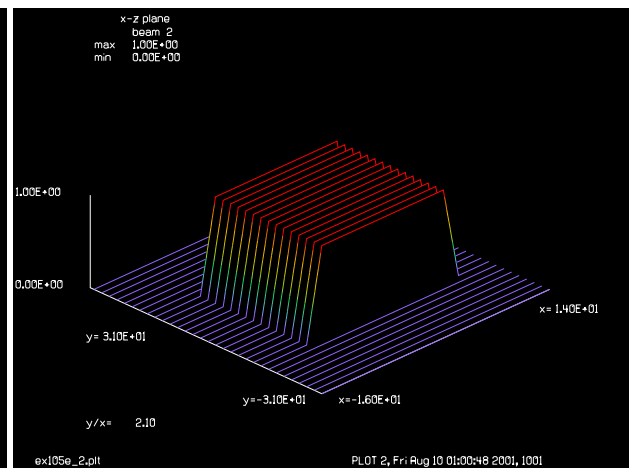


Fig. 105.21. X-Z section (11.25, 7.5) created with the DISPLAY macro that copys rows from the sections into Beam 2 to create a 2D array corresponding to the x-z section. Non-cubic array.

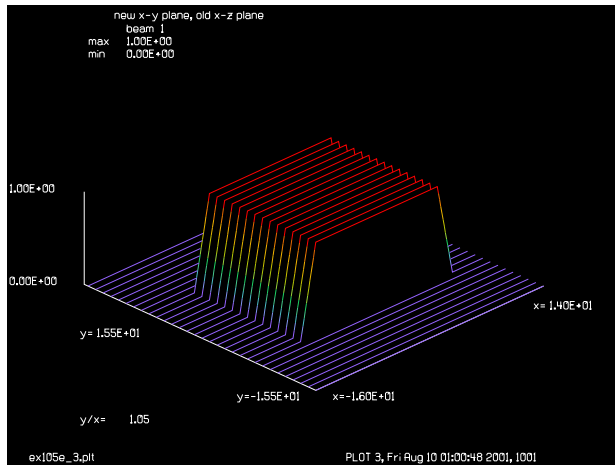


Fig. 105.22. A display of the X-Y section after the XZY transpose. The X-Y section is now identical to the original X-Z section. The x-axis is unchanged but the y-direction now has the values of the original z-axis. Section is (11.25, 7.5). Non-cubic array.

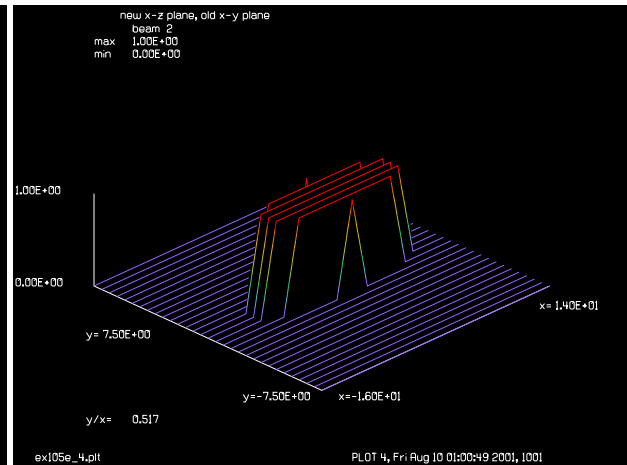


Fig. 105.23. X-Z section after XZY transpose. The X-Z section is now identical to the original X-Y section. The x-axis is unchanged but the z-direction now has the values of the original y-axis. Section is (11.25, 3.75). Non-cubic array.

Input: ex105e.inp

```
c## ex105e
#
# Example: Illustrate XZY transpose for non-cubic 3D array,
#           Flips Y- and Z-axes
#
variab/dec/int Nlinx Nliny Nlinz
variab/dec/int section row
mem/set/b 1
Nlinx = 16                # x-width
Nliny = 32                # y-width
Nlinz = 64                # z-width
array/s/3d 1 Nlinx Nliny Nlinz
units/s 1 2 1 .5
clear -1 0
nbeam 2 Nlinx Nliny
set/section 1 1
radius = 15 # 7.5
apt = 6
section = 0
macro/def sections/o
# macro to scan through all sections and add aperture
# in x-y plane of variable size as a function of z position
    section = section + 1
    set/section 1 section
    z = section - (Nlin/2 + 1)
    if [abs(z)<radius] then
c        apt = sqrt(radius^2 - z^2) list
        clap/e/c 1 1.5*apt .5*apt
    else
        clear 1 0.
```

Jump to: [Commands](#), [Theory](#)

```

endif
macro/end
Nlin = Nlinz
macro sections/Nlinz
macro/def display1/o
# copy center array to temporary array for later display
section = section + 1
row = section
set/section 1 section
copy/row 1 2 Nliny1/2+1 row
macro/end
macro/def display/o
array/s 2 Nlinx Nlinz1
section = 0
clear 2 0
macro display1/Nlinz1
plot/l 2
macro/end
set/sect 1 Nlinz/2+1
plot/w ex105e_1.plt 10 10 400 300
title x-y plane
plot/l 1
#
# display the aperture in the x-z plane
#
title aperture defined in x-z plane
plot/w ex105e_2.plt 410 10 400 300
title x-z plane
Nliny1 = Nliny
Nlinz1 = Nlinz
macro display
tran/xzy 1
plot/w ex105e_3.plt 10 310 400 300
title new x-y plane, old x-z plane
set/sect 1 Nliny/2+1
plot/l 1
plot/w ex105e_4.plt 410 310 400 300
title new x-z plane, old x-y plane
Nliny1 = Nlinz
Nlinz1 = Nliny
macro display
tran/xzy 1
plot/w ex105e_5.plt 10 620 400 300
title new x-y plane, old x-z plane
set/sect 1 Nlinz/2+1
plot/l 1
plot/w ex105e_6.plt 410 620 400 300
title new x-z plane, old x-y plane
Nliny1 = Nliny
Nlinz1 = Nlinz
macro display

```

Ex105f: Transpose of 3D array, left circular permutation

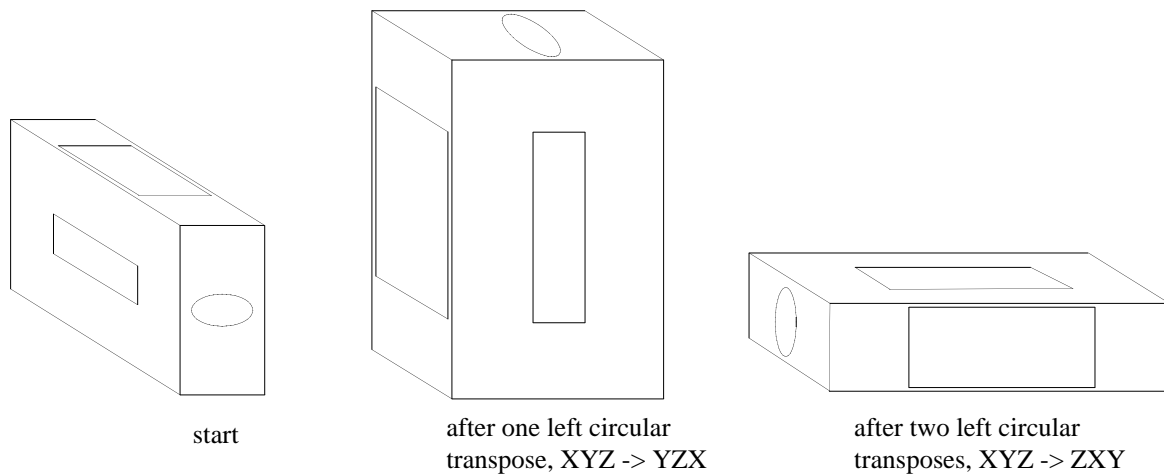


Fig. 105.24. A 3D volume transposed by a left circular permutation (YZX) followed by a second left circular permutation. Example 105f illustrates this effect with an array of dimensions 16 x 32 x 64.

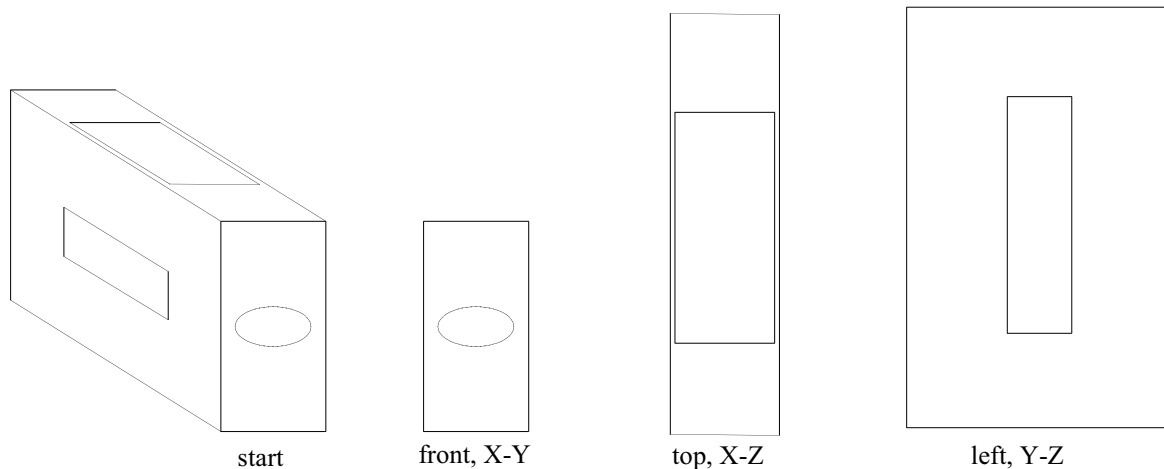


Fig. 105.25. Projections of the initial 3D volume onto the front (X-Y plane), top (X-Z plane), and left side (Y-Z) plane. These three projections are illustrated in Ex105f.inp in Figs. 105.28-10530.

Input: ex105f.inp

```
c## ex105f
#
# Illustrate transposes for 3D array, Permute XYZ axes with three
# left circular permutations of the axes. The third left shift
# restores the original distribution. The macros displayxz and
# displayyz shows the x-z and y-z axes respectively.
#
echo/on
variab/dec/int Nlinx Nliny Nlinz
```

Jump to: [Commands](#), [Theory](#)

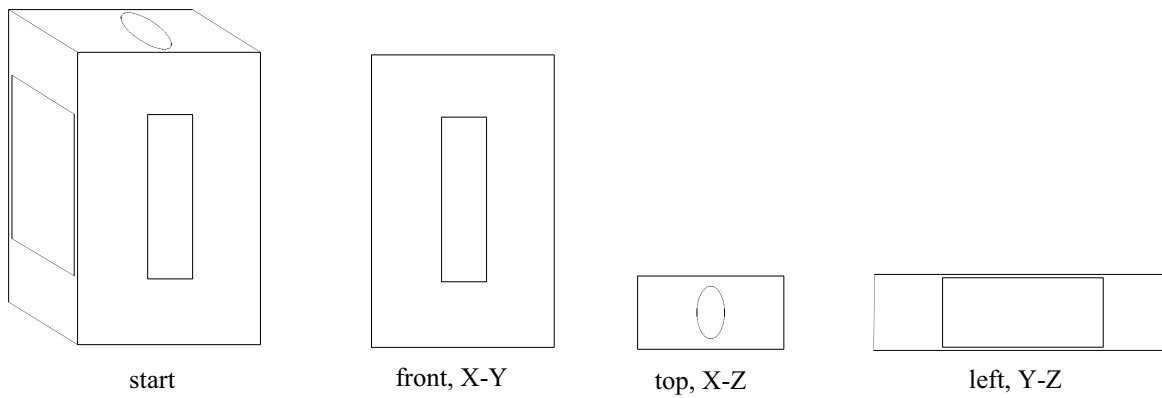


Fig. 105.26. After one left transpose, projections of the 3D volume onto the front (X-Y plane), top (X-Z plane), and left side (Y-Z) plane. These are illustrated in Ex105f.inp in Figs. 105.31-105.33.

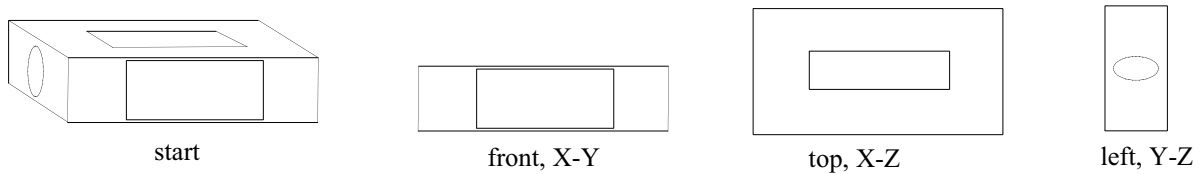


Fig. 105.27. After two left transposes, projections of the 3D volume onto the front (X-Y plane), top (X-Z plane), and left side (Y-Z) plane. These are illustrated in Ex105f.inp in Figs. 105.34-105.35.

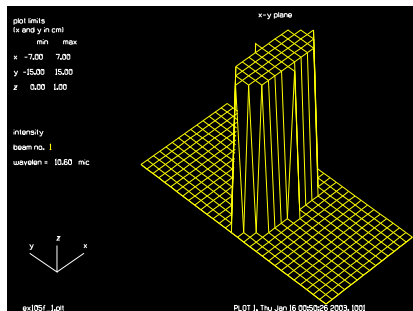


Fig. 105.28. Start, X-Y plane.

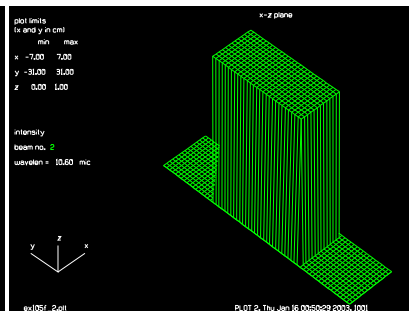


Fig. 105.29. Start, X-Z plane.

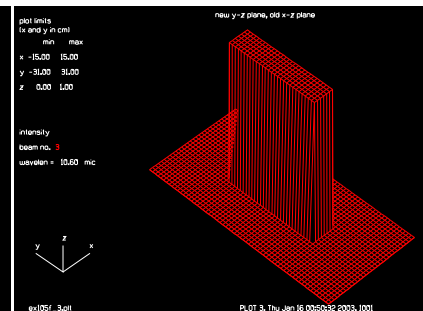


Fig. 105.30. Start Y-Z plane.

```

variab/dec/int section row
mem/set/b .0625 # 1
Nlinx = 16
Nliny = 32
Nlinz = 64
mem/set/b 1 # .125
array/s/3d 1 Nlinx Nliny Nlinz
mem/cont
pause
nbeam 2 Nlinx Nliny
nbeam 3 Nlinx Nliny
set/section 1 1

```

```

# x-width
# y-width
# z-width

```

Jump to: [Commands](#), [Theory](#)

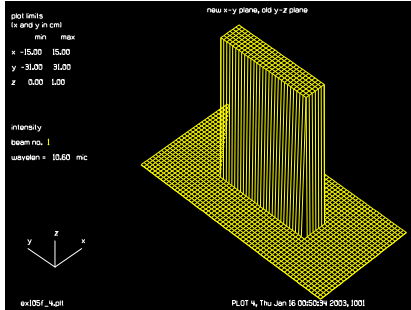


Fig. 105.31. Left (1), X-Y plane.

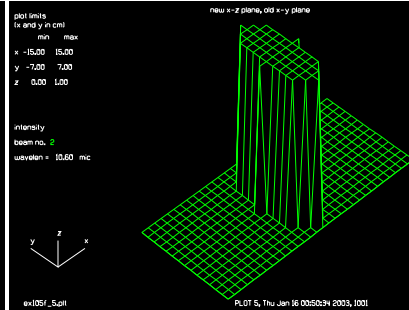


Fig. 105.32. Left (1), X-Z plane.

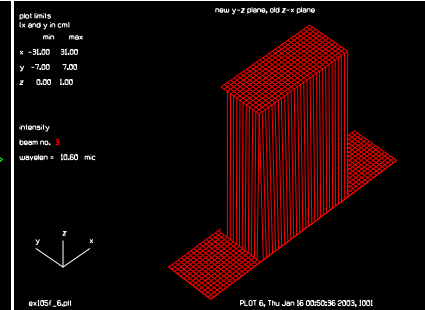


Fig. 105.33. Left (1), Y-Z plane.

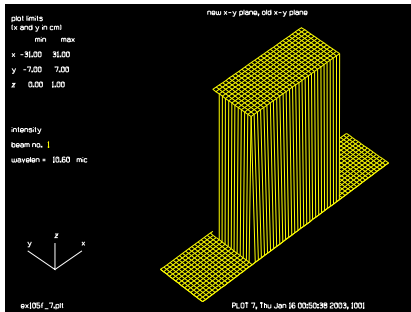


Fig. 105.34. Left (2), X-Y plane.

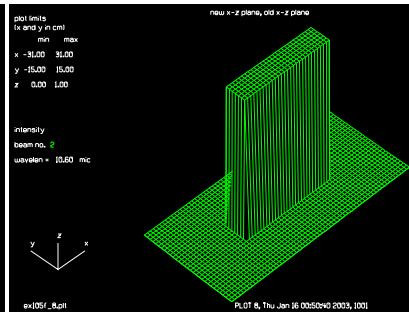


Fig. 105.35. Left (2), X-Z plane.

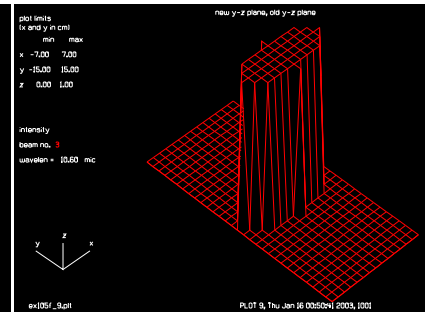


Fig. 105.36. Left (2), Y-Z plane.

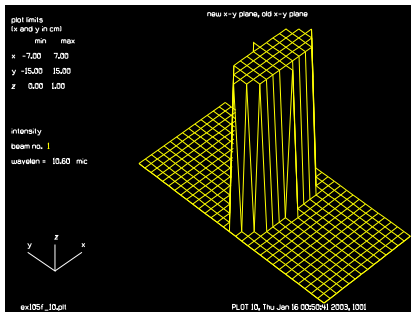


Fig. 105.37. Left (3), X-Y plane.

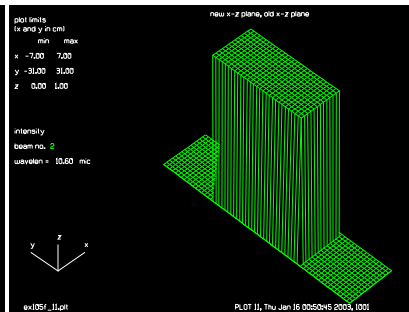


Fig. 105.38. Left (3), X-Z plane.

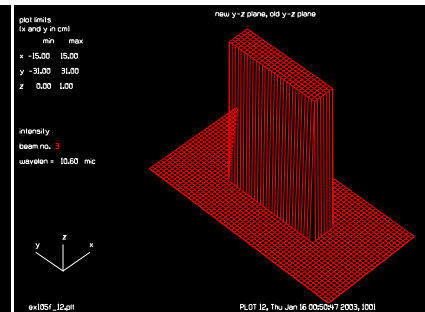


Fig. 105.39. Left (3), Y-Z plane.

```

radius = 15 # 7.5
apt = 6
section = 0
macro/def sections/o
# macro to scan through all sections and add aperture
# in x-y plane of variable size as a function of z position
    section = section + 1
    set/section 1 section
    z = section - (Nlin/2 + 1)
    if [abs(z)<radius] then
        clap/e/c 1 apt .5*apt
    else
        section=
        clear 1 0.
    endif
macro/end
Nlin = Nlinz

```

Jump to: [Commands](#), [Theory](#)

```

macro sections/Nlin
macro/def displayxz1/o
# copy center array to temporary array for later display
  section = section + 1
  row = section
  set/section 1 section
  copy/row 1 2 Nliny1/2+1 row
macro/end
macro/def displayxz/o
  array/s 2 Nlinx1 Nlinz1
  section = 0
  clear 2 0
  macro displayxz1/Nlinz1
  plot 2
macro/end
macro/def displayyz1/o
# copy center array to temporary array for later display
  section = section + 1
  row = section
  set/section 1 section
  copy/row 1 3 Nlinx1/2+1 row
macro/end
macro/def displayyz/o
  tran/xyz 1
  array/s 3 Nliny1 Nlinz1
  section = 0
  clear 3 0
  macro displayyz1/Nlinz1
  tran/xyz 1
  plot 3
macro/end
Nlinx1 = Nlinx
Nliny1 = Nliny
Nlinz1 = Nlinz
set/sect 1 Nlinz1/2+1
plot/w ex105f_1.plt 10 10 400 300
title x-y plane
plot 1
#
# display the aperture in the x-z plane
#
title aperture defined in x-z plane
plot/w ex105f_2.plt 410 10 400 300
title x-z plane
Nlinx1 = Nlinx
Nliny1 = Nliny
Nlinz1 = Nlinz
macro displayxz
plot/w ex105f_3.plt 810 10 400 300
title new y-z plane, old x-z plane
macro displayyz

tran/left 1 # left circular
Nlinx1 = Nliny

```

Jump to: [Commands](#), [Theory](#)


```

Nliny1 = Nlinz
Nlinz1 = Nlinx
set/sect 1 Nlinz1/2+1
plot/w ex105f_4.plt 10 310 400 300
title new x-y plane, old y-z plane
plot 1
plot/w ex105f_5.plt 410 310 400 300
title new x-z plane, old x-y plane
macro displayxz
plot/w ex105f_6.plt 810 310 400 300
title new y-z plane, old z-x plane
macro displayyz

tran/left 1 # left circular
Nlinx1 = Nlinz
Nliny1 = Nlinx
Nlinz1 = Nliny
set/sect 1 Nlinz1/2+1
plot/w ex105f_7.plt 10 610 400 300
title new x-y plane, old x-y plane
plot 1
plot/w ex105f_8.plt 410 610 400 300
title new x-z plane, old x-z plane
macro displayxz
plot/w ex105f_9.plt 810 610 400 300
title new y-z plane, old y-z plane
macro displayyz

tran/left 1 # left circular
Nlinx1 = Nlinx
Nliny1 = Nliny
Nlinz1 = Nlinz
set/sect 1 Nlinz1/2+1
plot/w ex105f_10.plt 10 910 400 300
title new x-y plane, old x-y plane
plot 1
plot/w ex105f_11.plt 410 910 400 300
title new x-z plane, old x-z plane
macro displayxz
plot/w ex105f_12.plt 810 910 400 300
title new y-z plane, old y-z plane
macro displayyz

```

Ex105g: Transpose of 3D array, right circular permutation

Input: ex105g.inp

```

c## ex105g
#
# Illustrate transposes for 3D array, Permute XYZ axes with three
# right circular permutations of the axes. The third right shift
# restores the original distribution. The macros displayxz and
# displayyz shows the x-z and y-z axes respectively.
#

```

Jump to: [Commands](#), [Theory](#)

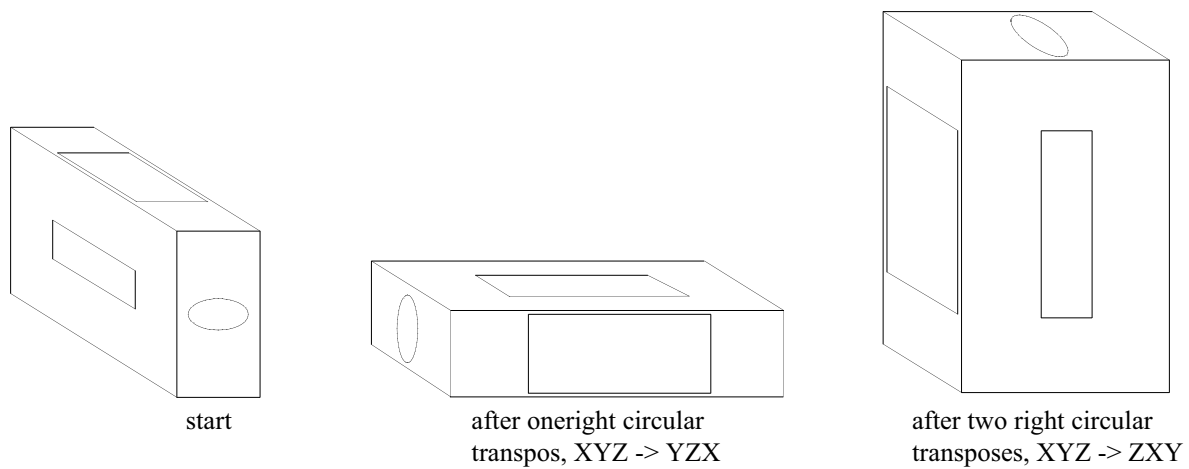


Fig. 105.40. A 3D volume transposed by a right circular permutation (YXZ) followed by a second right circular permutation. Example 105g illustrates this effect with an array of dimensions 16 x 32 x 64.

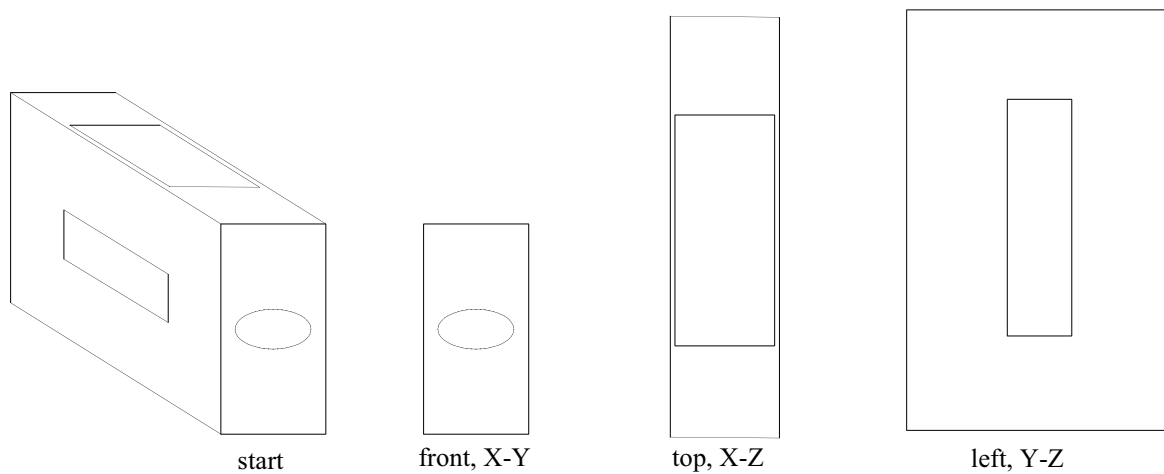


Fig. 105.41. Projections of the initial 3D volume onto the front (X-Y plane), top (X-Z plane), and left side (Y-Z) plane. These three projections are illustrated in Ex105g.inp in Figs. 105.44-105.46.

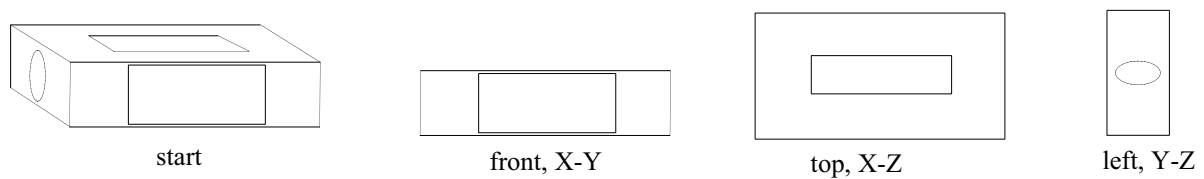


Fig. 105.42. After one right transpose, projections of the 3D volume onto the front (X-Y plane), top (X-Z plane), and left side (Y-Z) plane. These are illustrated in Ex105g.inp in Figs. 105.47-105.49.

```
echo/on
variab/dec/int Nlinx Nliny Nlinz
```

Jump to: [Commands](#), [Theory](#)

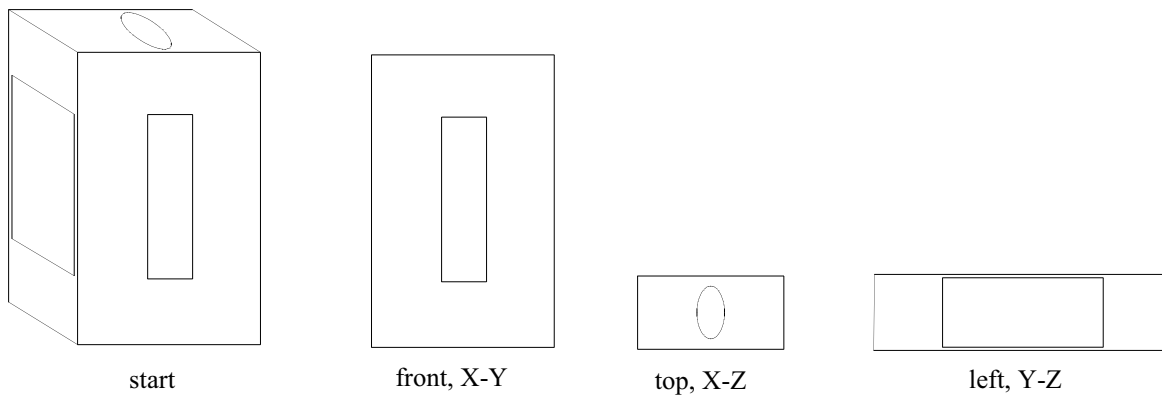


Fig. 105.43. After two right transposes, projections of the 3D volume onto the front (X-Y plane), top (X-Z plane), and left side (Y-Z) plane. These are illustrated in Ex105g.inp in Figs. 105.50-105.53.

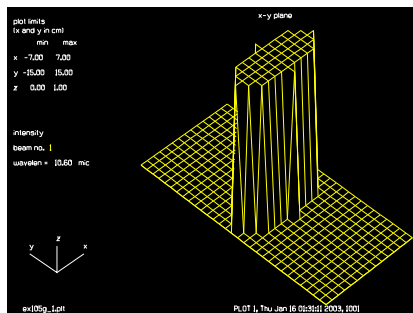


Fig. 105.44. Start, X-Y plane.

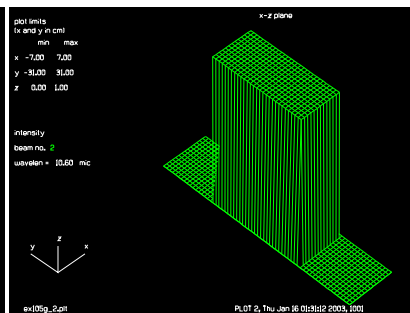


Fig. 105.45. Start, X-Z plane.

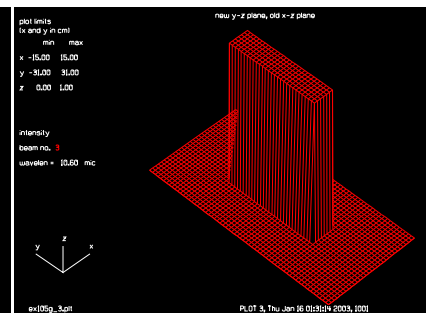


Fig. 105.46. Start Y-Z plane.

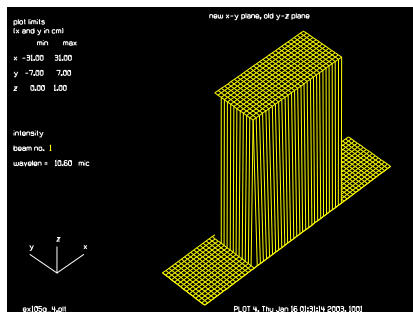


Fig. 105.47. Right (1), X-Y plane.

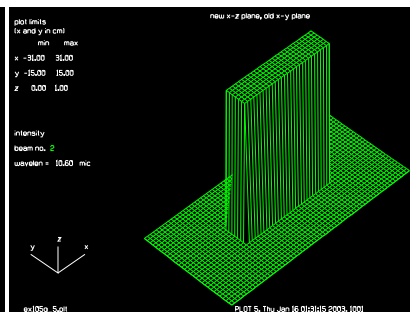


Fig. 105.48. Right (1), X-Z plane.

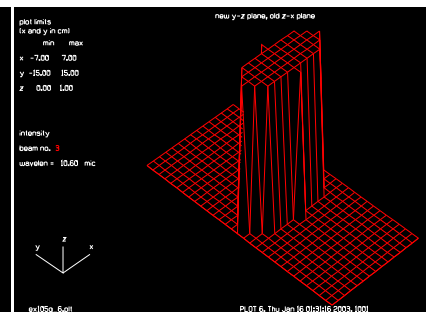


Fig. 105.49. Right (1), Y-Z plane.

```

variab/dec/int section row
mem/set/b 1
Nlinx = 16                # x-width
Nliny = 32                # y-width
Nlinz = 64                # z-width
array/s/3d 1 Nlinx Nliny Nlinz
nbeam 2 Nlinx Nliny
nbeam 3 Nlinx Nliny
mem/cont
set/section 1 1

```

Jump to: [Commands](#), [Theory](#)

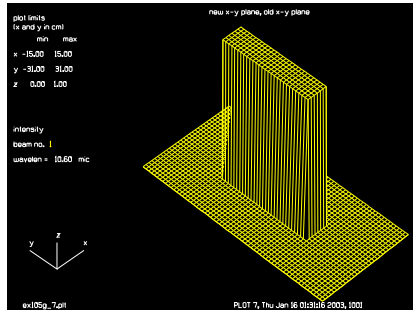


Fig. 105.50. Right (2), X-Y plane.

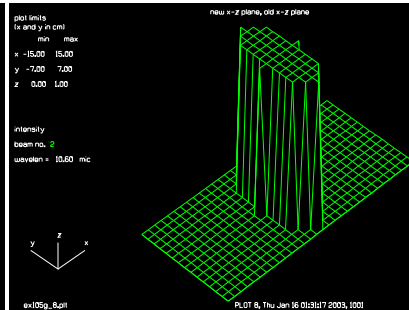


Fig. 105.51. Right (2), X-Z plane.

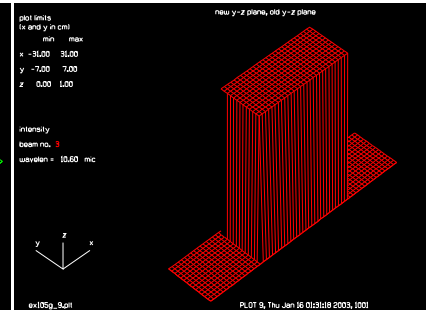


Fig. 105.52. Right (2), Y-Z plane.

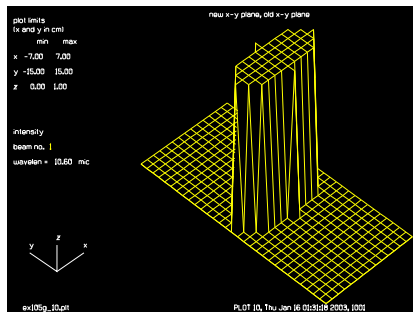


Fig. 105.53. Right (3), X-Y plane.

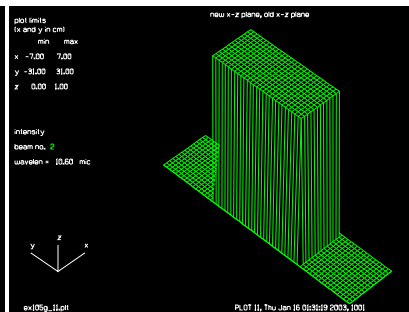


Fig. 105.54. Right (3), X-Z plane.

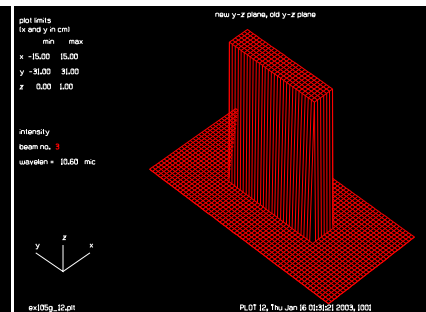


Fig. 105.55. Right (3), Y-Z plane.

```

radius = 15
apt = 6
section = 0
macro/def sections/o
# macro to scan through all sections and add aperture
# in x-y plane of variable size as a function of z position
    section = section + 1
    set/section 1 section
    z = section - (Nlin/2 + 1)
    if [abs(z)<radius] then
        clap/e/c 1 apt .5*apt
    else
        section=
        clear 1 0.
    endif
macro/end
Nlin = Nlinz
macro sections/Nlin
macro/def displayxz1/o
# copy center array to temporary array for later display
    section = section + 1
    row = section
    set/section 1 section
    copy/row 1 2 Nliny1/2+1 row
macro/end
macro/def displayxz/o
    array/s 2 Nlinx1 Nlinz1
    section = 0
    clear 2 0

```

Jump to: [Commands](#), [Theory](#)

```

macro displayxz1/Nlinz1
plot 2
macro/end
macro/def displayyz1/o
# copy center array to temporary array for later display
section = section + 1
row = section
set/section 1 section
copy/row 1 3 Nlinx1/2+1 row
macro/end
macro/def displayyz/o
tran/xyz 1
array/s 3 Nliny1 Nlinz1
section = 0
clear 3 0
macro displayxz1/Nlinz1
tran/xyz 1
plot 3
macro/end
Nlinx1 = Nlinx
Nliny1 = Nliny
Nlinz1 = Nlinz
set/sect 1 Nlinz1/2+1
plot/w ex105g_1.plt 10 10 400 300
title x-y plane
plot 1
#
# display the aperture in the x-z plane
#
title aperture defined in x-z plane
plot/w ex105g_2.plt 410 10 400 300
title x-z plane
Nlinx1 = Nlinx
Nliny1 = Nliny
Nlinz1 = Nlinz
macro displayxz
plot/w ex105g_3.plt 810 10 400 300
title new y-z plane, old x-z plane
macro displayyz
tran/right 1 # right circular
Nlinx1 = Nlinz
Nliny1 = Nlinx
Nlinz1 = Nliny
set/sect 1 Nlinz1/2+1
plot/w ex105g_4.plt 10 310 400 300
title new x-y plane, old y-z plane
plot 1
plot/w ex105g_5.plt 410 310 400 300
title new x-z plane, old x-y plane
macro displayxz
plot/w ex105g_6.plt 810 310 400 300
title new y-z plane, old z-x plane
macro displayyz
tran/right 1 # right circular

```

Jump to: [Commands](#), [Theory](#)

```
Nlinx1 = Nliny
Nliny1 = Nlinz
Nlinz1 = Nlinx
set/sect 1 Nlinz1/2+1
plot/w ex105g_7.plt 10 610 400 300
title new x-y plane, old x-y plane
plot 1
plot/w ex105g_8.plt 410 610 400 300
title new x-z plane, old x-z plane
macro displayxz
plot/w ex105g_9.plt 810 610 400 300
title new y-z plane, old y-z plane
macro displayyz
tran/right 1 # right circular
Nlinx1 = Nlinx
Nliny1 = Nliny
Nlinz1 = Nlinz
set/sect 1 Nlinz1/2+1
plot/w ex105g_10.plt 10 910 400 300
title new x-y plane, old x-y plane
plot 1
plot/w ex105g_11.plt 410 910 400 300
title new x-z plane, old x-z plane
macro displayxz
plot/w ex105g_12.plt 810 910 400 300
title new y-z plane, old y-z plane
macro displayyz
```

Ex106: Fiber-to-fiber coupling

Table. 106.1. Table of Ex106 examples

Ex106a: Fiber-to-fiber coupling with a single ideal lens.	1
Ex106b: Fiber-to-fiber coupling with a single aspheric lens, using lensgroup.....	3
Ex106c: Fiber-to-fiber coupling with tilt at input	5
Ex106d: Fiber-to-fiber coupling with decenter at input	6
Ex106e: More complex example of fiber-to-fiber coupling	8
Ex106f: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), no apertures, through-focus	12
Ex106g: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), no apertures, best focus	16
Ex106h: Fiber-to-focus GRIN, aberration ($\alpha = 1.8$), no apertures, through-focus	18
Ex106i: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), apertures implemented, through-focus.....	22
Ex106j: Fiber-to-focus GRIN, aberration ($\alpha = 1.8$), apertures implemented	25
Ex106k: Optimization of fiber-to-fiber GRIN lens system (paraxial).....	29
Ex106l: Optimization of fiber-to-fiber GRIN lens system (phase aberration model).....	30
Ex106m: Optimization of fiber-to-fiber GRIN lens system	33
Ex106n: Example of multi-mode diode laser	36

This example illustrates several cases of fiber-to-fiber coupling as indicated in Table 106.1. Fig. 106.1 illustrates a simple 1:1 fiber-to-fiber coupler using an ideal lens. Ex106a.inp illustrates a simple fiber-to-fiber coupler using a single, ideal lens. The in-coupling is modeled by using the `mult/mode/parallel` function.

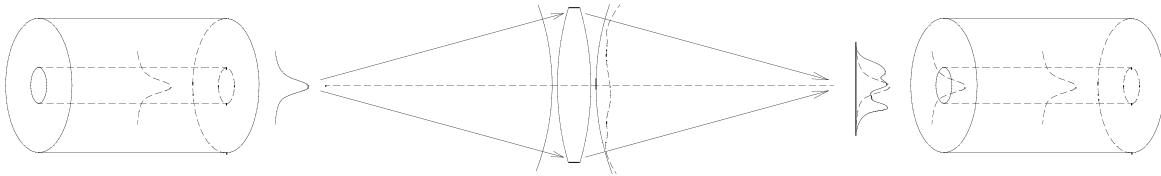


Fig. 106.1. Output of a fiber on the left is reimaged by a lens (or lens system) onto a single mode fiber on the right. The receiving fiber extracts its mode shape from the input, scattering out the remainder.

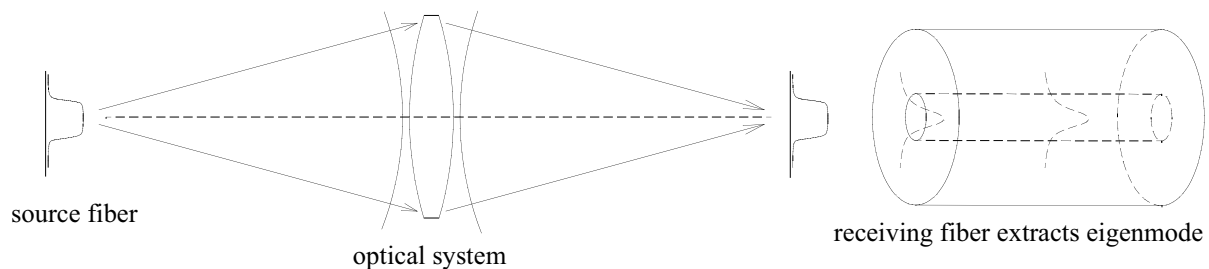


Fig. 106.2. Hypothetical case: couple a supergaussian output into a fiber with a gaussian mode shape.

Ex106a: Fiber-to-fiber coupling with a single ideal lens.

Example 106a idealized lens treatment of a simple fiber-to-fiber coupler.

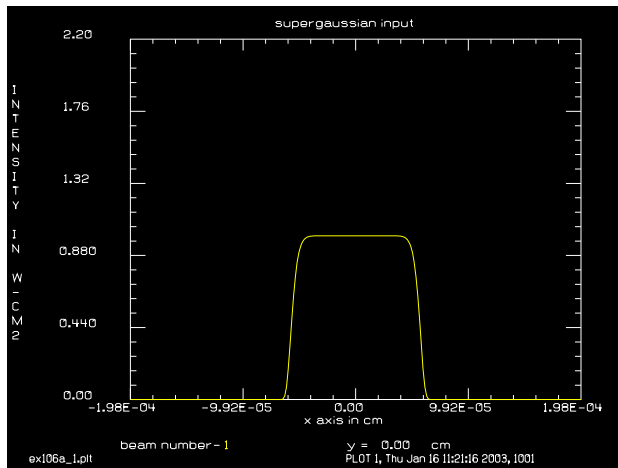


Fig. 106.3. Hypothetical starting distribution from initial fiber for ex106a.

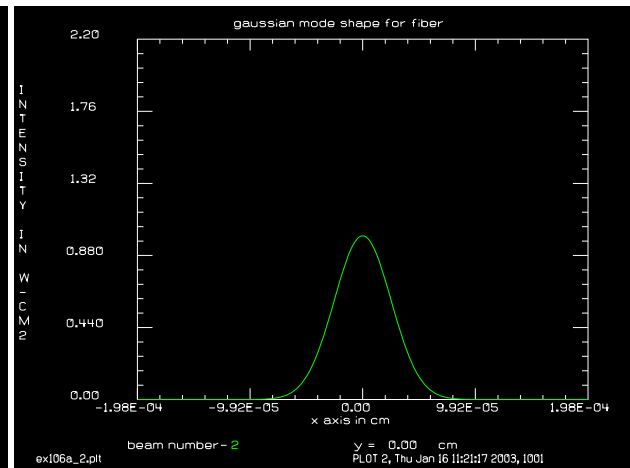


Fig. 106.4. Assumed gaussian mode shape for receiving fiber for ex106a.

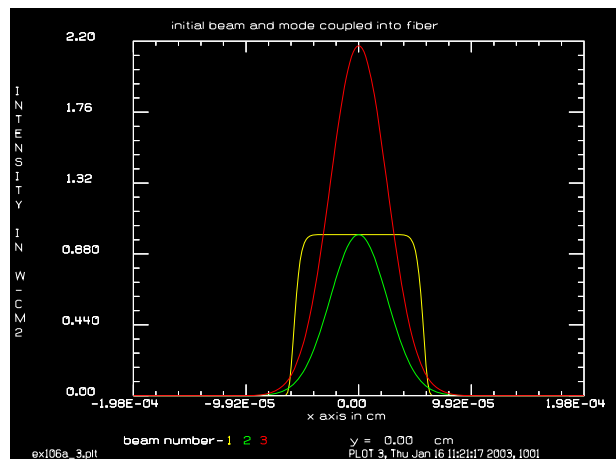


Fig. 106.5. Extracted mode (red): 87.3% energy conversion from start (yellow) for ex106a.

Input: `ex106a.inp`

```
c## ex106a
```

```
# Example: Simple, single mode fiber coupling
```

```
alias Name ex106a
```

```
array/s 1 256
```

```
nbeam 3 data
```

```
units/field 0 .0002 # field half-width of array
```

```
gaus/c/c 1 1 .00006 8
```

```
c
```

```
c A simple system to reimage the beam
```

Jump to: [Commands](#), [Theory](#)


```

c
prop 10    # distance to lens
lens 1 5    # lens
prop 10    # lens to image distance
c
c Make a copy of beam 1 in beam 3
c
copy 1 3
title supergaussian input
plot/w @Name_1.plt
plot/x/i 1 fmax=2.2
c
c make gaussian beam for single mode fiber
c
gaus/c/c 2 1 .00005
plot/w @Name_2.plt
title gaussian mode shape for fiber
plot/x/i 2 fmax=2.2
c
c Mult beam 3 by beam 2
c
energy 3
mult/mode/parallel 3 2
energy 3
plot/w @Name_3.plt
title initial beam and mode coupled into fiber
plot/x/i fi=1 la=3 fmax=2.2

```

Ex106b: Fiber-to-fiber coupling with a single aspheric lens, using lensgroup.

This is an example of a single lens, fiber-to-fiber coupler using lensgroup to represent an aspheric lens.

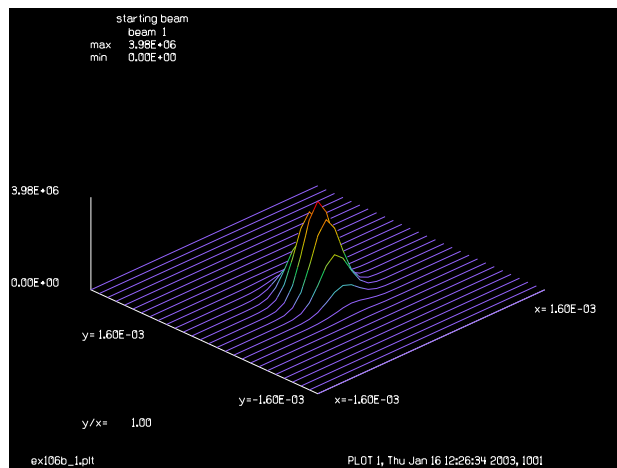


Fig. 106.6. Starting distribution for ex106b.

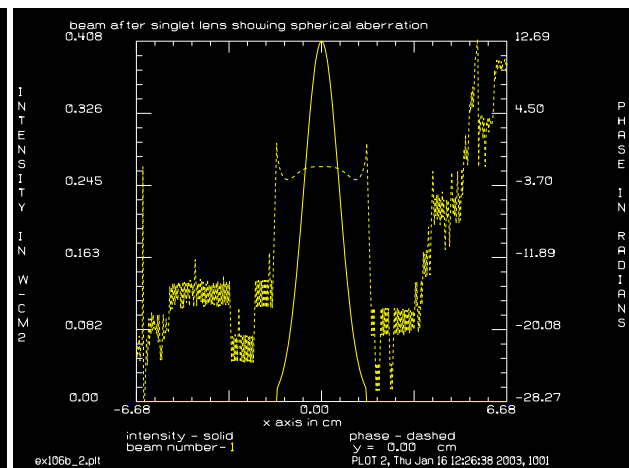


Fig. 106.7. Intensity and phase after conic singlet for ex106b.

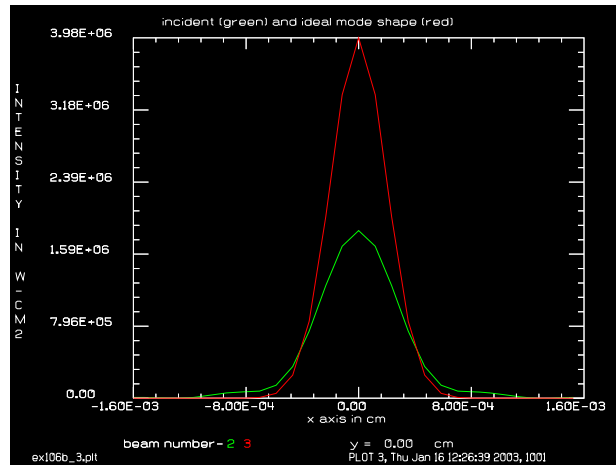


Fig. 106.8. Incident beam after lens (green) and ideal mode shape (red) for ex106b.

Input: `ex106b.inp`

Ex106c: c## ex106b

```
#
# Example: Fiber-to-fiber coupling with a single aspheric lens, using lensgroup
#

# A gaussian beam representing the output of an optical fiber
# is imaged by a plano-convex lens onto an identical fiber.
# The lens has a conic surface to remove most of the spherical
# aberration
alias Name ex106b
mem/set/b 8      # set memory to more than enough
array/s 1 512    # set array size
Waist=4e-4       # waist of 4 microns
nbeam 3 data     # make two extra arrays for data
units/field 0 .03 # field half-width of array
wavelength/set 0 1.55 # set wavelength to 1.55 micron
gaus/c/c 1 1 Waist # gaussian start
energy/norm 1 1   # set energy to unity
plot/w @Name_1.plt
title starting beam
plot/l 1 xrad=4*Waist
c
c Define a lensgroup to reimage the beam
c
lensgroup/def singlet/overwrite
#      radius thickness glassname
```

Jump to: [Commands](#), [Theory](#)

```

surface_lensgroup 1e10 .2    bk7
#    radius thickness conic-constant [aspheric terms]
surface_lensgroup -2.537 .0    cc=-1.132          air
image focus
lensgroup/end
vertex/locate/abs 0 0 10      # locate lensgroup 10 cm from start
prop/vertex
clap/c 1 1.6                  # insert clear aperture
zreff
lensgroup/run/radial singlet 1    # call lensgroup
plot/w @Name_2.plt
title beam after singlet lens showing spherical aberration
plot/x 1
focus/apply/waist            # propagate to gaussian waist
c
c Make a copy of beam 1 in beam 2
c
copy 1 2
c
c make gaussian beam to represent single mode fiber
c
units/beam 3 1
gaus/c/c 3 1 Waist
energy/norm 3 1
plot/w @Name_3.plt
title incident (green) and ideal mode shape (red)
plot/x/i fi=2 la=3 le=-4*Waist ri=4*Waist
c
c Mult beam 1 by beam 3
c
mult/mode/parallel 2 3
energy 2                      # How much energy got into fiber mode?
Fiber-to-fiber coupling with tilt at input
A tilted input may be represented by a phase tilt—in this case 10°.

```

Input: ex106c.inp

```

c## ex106c

# Example: Simple, single mode fiber coupling with tilt

alias Name ex106c
array/s 1 256
nbeam 3 data
Lambda = 1.3e-4

```

Jump to: [Commands](#), [Theory](#)

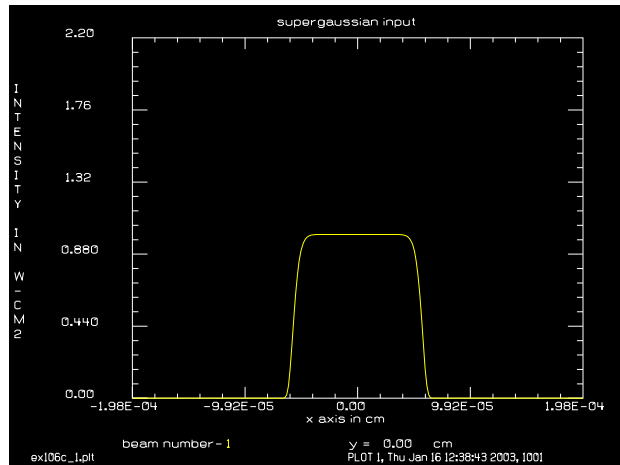


Fig. 106.9. Starting distribution for ex106c.

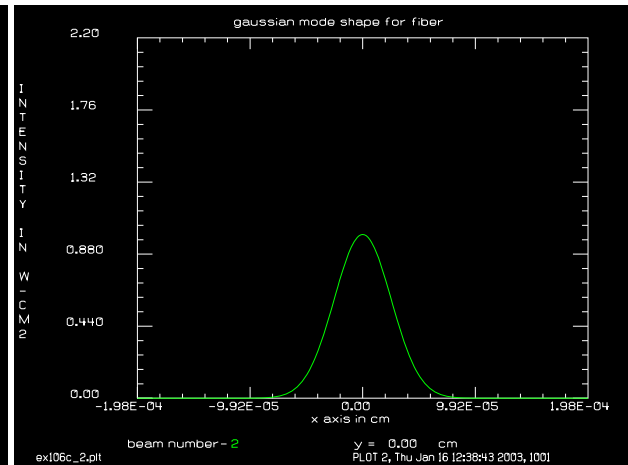


Fig. 106.10. Assumed gaussian mode shape for ex106c.

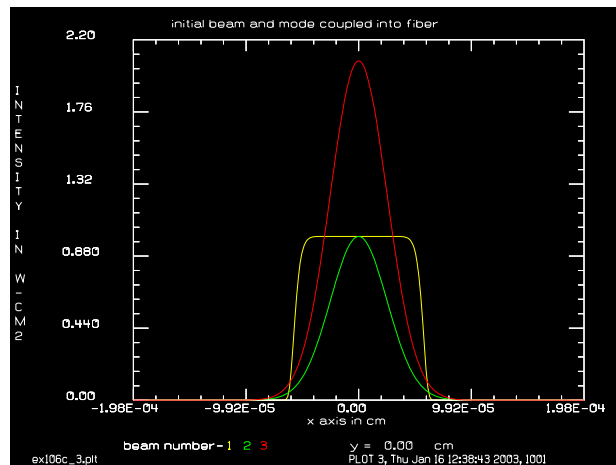


Fig. 106.11. Extracted mode (red) for ex106c. Note the peak has dropped relative to Fig. 106.8.

```
wavelength/set 0 Lambda*1e4
units/field 0 .0002 # field half-width of array
gaus/c/c 1 1 .00006 8
c
c A simple system to reimagethe beam
c
prop 10 # distance to lens
lens 1 5 # lens
prop 10 # lens to image distance
c
c Make a copy of beam 1 in beam 3
c
copy 1 3
title supergaussian input
plot/w @Name_1.plt
plot/x/i 1 fmax=2.2
c
c make gaussian beam for single mode fiber
c
```

Jump to: [Commands](#), [Theory](#)

```

gaus/c/c 2 1 .00005
plot/w @Name_2.plt
title gaussian mode shape for fiber
plot/x/i 2 fmax=2.2
c
c Mult beam 3 by beam 2
c
Angle_Degrees = 10
Rnorm = 1
Waves = pi*Angle_Degrees/180.*Rnorm/Lambda list
abr/tilt 3 Waves rnorm=Rnorm
variab/set Energy1 3 energy

mult/mode/parallel 3 2
variab/set Energy2 3 energy
Efficiency = Energy2/Energy1 list

plot/w @Name_3.plt
title initial beam and mode coupled into fiber
plot/x/i fi=1 la=3 fmax=2.2

```

Ex106d: Fiber-to-fiber coupling with decenter at input

A decentered input may be represented rescale/shift.

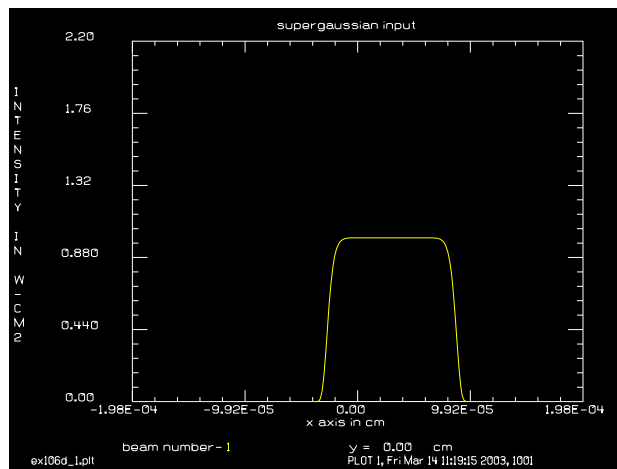


Fig. 106.12. Starting distribution for ex106d after decenter.

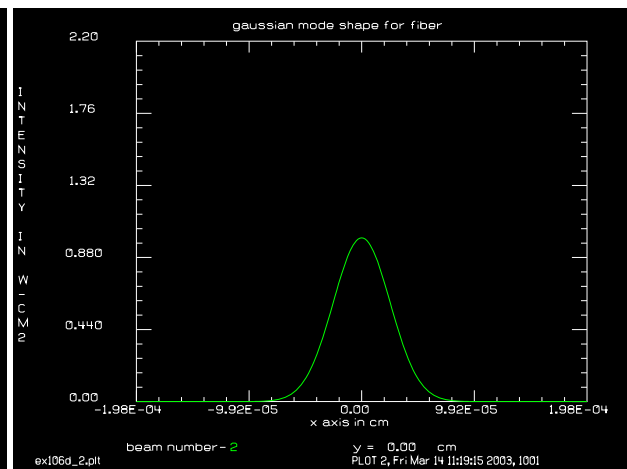


Fig. 106.13. Assumed gaussian mode shape for ex106d.

Input: ex106d.inp

```

c## ex106d

# Example: Simple, single mode fiber coupling, decenter

alias Name ex106d
array/s 1 256
nbeam 3 data
Lambda = 1.3e-4

```

Jump to: [Commands](#), [Theory](#)

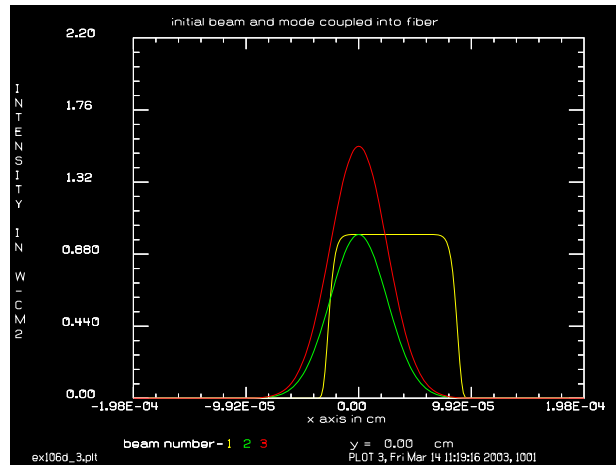


Fig. 106.14. Extracted mode (red) for ex106k. Note the peak has dropped relative to Fig. 106.8.

```
wavelength/set 0 Lambda*1e4
units/field 0 .0002 # field half-width of array
gaus/c/c 1 1 .00006 8
c
c A simple system to reimage the beam
c
prop 10 # distance to lens
lens 1 5 # lens
prop 10 # lens to image distance
c
c Make a copy of beam 1 in beam 3
c
rescale/shift 1 .3e-4
copy 1 3
title supergaussian input
plot/w @Name_1.plt
plot/x/i 1 fmax=2.2
c
c make gaussian beam for single mode fiber
c
gaus/c/c 2 1 .00005
plot/w @Name_2.plt
title gaussian mode shape for fiber
plot/x/i 2 fmax=2.2
c
c Mult beam 3 by beam 2
c
variab/set Energy1 3 energy

mult/mode/parallel 3 2
variab/set Energy2 3 energy
Efficiency = Energy2/Energy1 list

plot/w @Name_3.plt
title initial beam and mode coupled into fiber
plot/x/i fi=1 la=3 fmax=2.2
```

Jump to: [Commands](#), [Theory](#)

Ex106e: More complex example of fiber-to-fiber coupling

This example takes the output of a fiber, approximated by a gaussian beam and collimates the beam by a plan-convex lens. The beam is then propagated 1 cm to a block of silica and then 3 cm through the block. The radius of the lens was selected to put the gaussian waist near the center of the silica block. An identical lens (reversed) is placed after the silica block. The in-coupling efficiency is calculated with `mult/mode/parallel` to determine the light entering the fiber. A macro is constructed so the configuration is can be changed just by changing the appropriate variables.

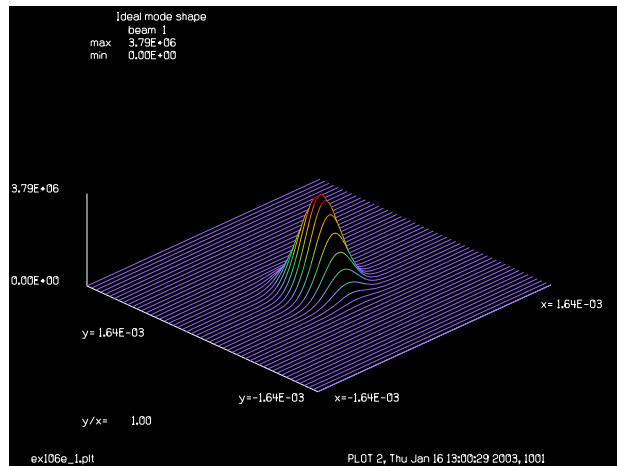


Fig. 106.15. Starting distribution for ex106e after decenter.

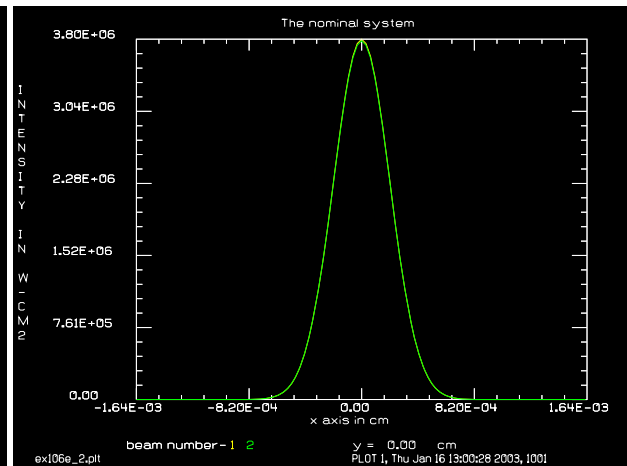


Fig. 106.16. The nominal system for ex106e.

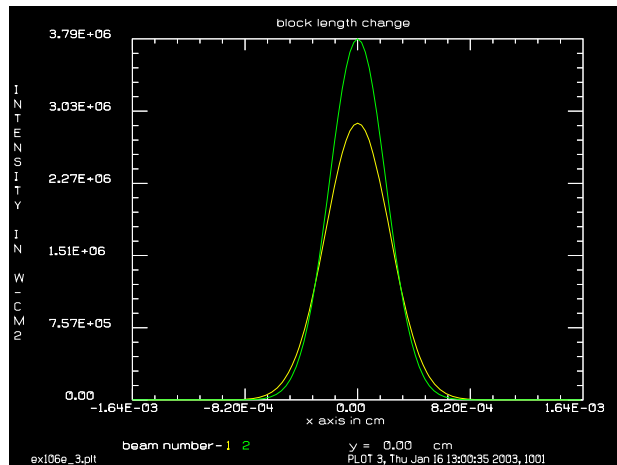


Fig. 106.17. Effect of block length change for ex106e.

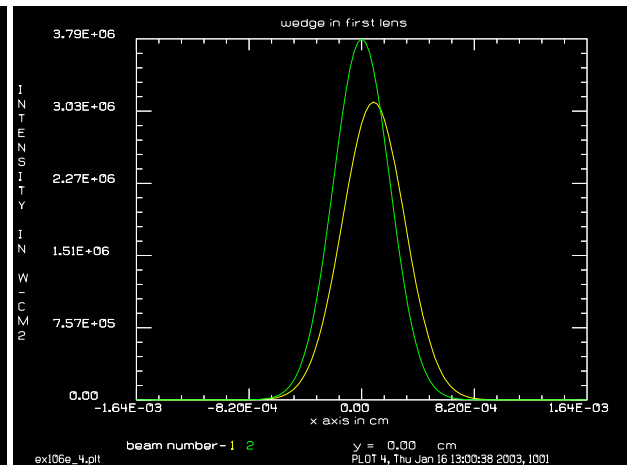


Fig. 106.18. Effect of wedge of 0.003 milliradians for ex106e.

Input: ex106e.inp

```
c## ex106e
#
# Example: coupling of one optical fiber into another
#
# This example takes the output of a fiber, approximated by
```

Jump to: [Commands](#), [Theory](#)

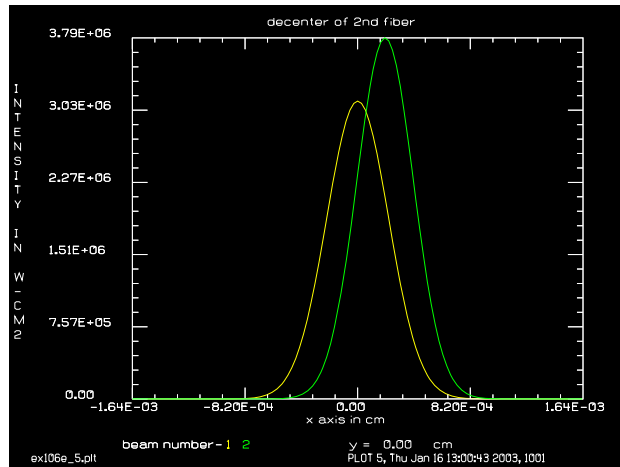


Fig. 106.19. Decenter of 2nd fiber ex106e.

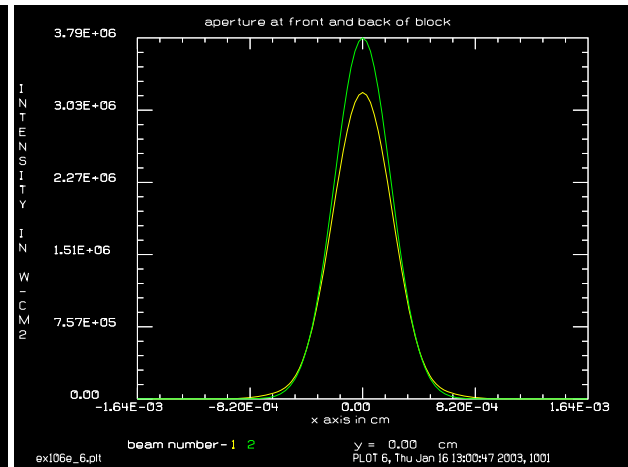


Fig. 106.20. With aperture at front and back of block for ex106e.

```
# a gaussian beam and collimates the beam by a plan-convex
# lens. The beam is then propagated 1 cm to a block of silica
# and then 3 cm through the block. The radius of the lens
# was selected to put the gaussian waist near the center of
# the silica block.
#
# an identical lens (reversed) is placed after the silica block.
# The in-coupling efficiency is calculated with mult/mode/parallel
# to determine the light entering the fiber.
#
# A macro is constructed so the configuration is can be changed
# just by changing the appropriate variables.
#
# Cases
# 1. Nominal system, near-perfect efficiency
# 2. Change in the length of the block
# 3. Wedge in the first lens
# 4. Decenter of the 2nd fiber
# 5. Add apertures at front and back of block
#
# output of optical fiber          0.41 micron radius
# path in BK7                     0.4438 cm
# radius of plano-convex lens     .13 (adjusted for 1.3 micron light)
# glass of plano-convex lens      BK7
# lens-to-block                   1 cm
# glass of block                   silica
# matching system on the other side of the block

alias Name ex106e
variable/dec/integer Nline
Nline = 512                                # define size of the array
array/set 1 Nline                          # set array size
Lambda = .00013                            # wavelength 1.3 micron (assumed)
wavelength/set 1 Lambda*1e4                # set wavelength
w0 = 0.000410                             # set gaussian waist radius
```



```

lens_radius = .13          # was .13688
lens_thickness = .4438
lens_to_block = 1
block_thickness = 3
Wedge_lens_radians = 0    # wedge angle of lens 1 in radians
Xdecenter = 0             # decenter of 2nd fiber
Clear_aperture = .1       # start with large aperture

nbeam 2 data               # make a new beam

macro/def system/o
c  write/off
  array/set 1 Nline
  units/field 1 20*w0      # set width of field to be 30*w0
  zreff/se 1 0
  gaus/c/c 1 1.0 w0       # set up gaussian beam
  energy/norm 1 1         # normalize to 1 w total power

  lens/flat 1 bk7          # start BK7 plan0-convex lens
  prop lens_thickness      # propagate through lens

  Waves = Wedge_lens_radians/Lambda
  abr/tilt 1 Waves az=90 rnorm=1
  lens/sph/surface 1 -lens_radius air # convex surface

  prop lens_to_block      # propagate to block

  clap/cir/noadjust 1 Clear_aperture
  lens/flat/surface 1 silica # start block
  prop block_thickness
  lens/flat/surface 1 air
  clap/cir/noadjust 1 Clear_aperture

  prop lens_to_block      # propage to next lens

  lens/sph/surface 1 lens_radius bk7
  prop lens_thickness

# make 2nd gaussian for input mode shape
units/beam 2 1
gaus/c/c 2 1.0 w0 decx=Xdecenter # make a second gaussian beam
energy/norm 2 1
plot/x/i fi=1 la=2 le=-4*w0 ri=4*w0

  mult/mode/parallel 1 2    # calculate in-coupling by overlap
                           # integral
  energy 1
c  write/on
macro/end

C Nominal design

title The nominal system
plot/watch @Name_2.plt

```

Jump to: [Commands](#), [Theory](#)

```

macro system
variable/set Energy_nominal 1 energy list
title Ideal mode shape
plot/w @Name_1.plt
plot/l 1 ns=64 xrad=4*w0

C length change of block

block_thickness = 3.02
title block length change
plot/watch @Name_3.plt
macro system
variable/set Energy_block_change 1 energy list

C wedge in first lens

block_thickness = 3
Wedge_lens_radians = .0003

title wedge in first lens
plot/watch @Name_4.plt
macro system
variable/set Energy_wedge_change 1 energy list

C decenter 2nd fiber by Xdecenter

Wedge_lens_radians = 0.
Xdecenter = .0002

title decenter of 2nd fiber
plot/watch @Name_5.plt
macro system
variable/set Energy_decenter 1 energy list

C aperture at front and back of block

Xdecenter = .0000
Clear_aperture = .025

title aperture at front and back of block
plot/watch @Name_6.plt
macro system
variable/set Energy_aperture 1 energy list

variables                                # list variables

```

Ex106f: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), no apertures, through-focus

Example 106.24 illustrates a gradient index lens and fiber-to-fiber coupling. Light is output from a single mode fiber at $\lambda = 1.32$ micron, with an approximately gaussian output of radius $\omega_0 = 4.5$ micron, into

Jump to: [Commands](#), [Theory](#)

a silica rod ($n = 1.4467$). A GRIN lens is placed at the end of the silica rod. The GRIN lens has an index of the form:

$$n(r) = n_0 \left(1 - \frac{\gamma^2}{2} r^2 \right), \quad n_0 = 1.49 \text{ and } \gamma^2 = 1378.68 \text{ cm}, \quad a = 2.0 \text{ or } 1.8 \quad (106.1)$$

An exact representation uses the `abr/grin` command and propagation in a series of split-steps, to model the distributed effect of GRIN material. The exponent α will be 2.0 for an ideal GRIN lens. In the through-focus examples Ex106h and Ex106j, a value of $\alpha = 1.8$ is used to demonstrate an imperfect GRIN lens. The index of 1.8 will result in overcorrected spherical aberration forming a caustic beyond the waist. A paraxial lens may be created with the `lens/grin` command (see Ex106g).

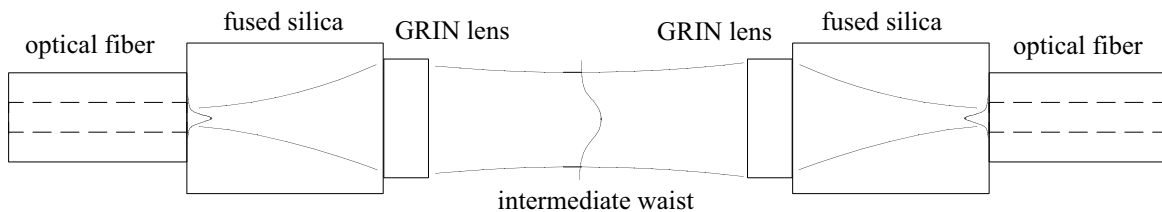


Fig. 106.21. Light is emitted from a single mode fiber into a fused silica rod. At the end of the rod a GRIN lens creates a large diameter waist. The beam is collected by the mirror-image system.

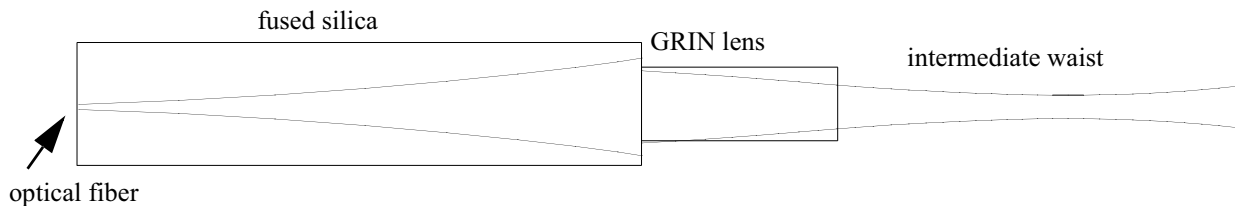


Fig. 106.22. Light is emitted from a single mode fiber into a fused silica rod. At the end of the rod a GRIN lens creates a large diameter waist. Examples Ex106a-d show the through-focus behavior past the waist.

Input: ex106f.inp

```
c## ex106f
#
# Example 106f: Fiber-to-focus GRIN, no aberration, no aperture
#
# The objective is to scan the beam profile at all points along the
# path. The system consists of the output of an optical fiber,
# a path through fused silica, a GRIN lens, and an air path.
# The GRIN lens is implemented as abr/grin and a series of
# diffraction steps.
#
# In this case, the GRIN coefficient is alpha=2 with no apertures
# implemented.
#
# We search for the best focus position as defined by minimum
# waist size.
```

Jump to: [Commands](#), [Theory](#)

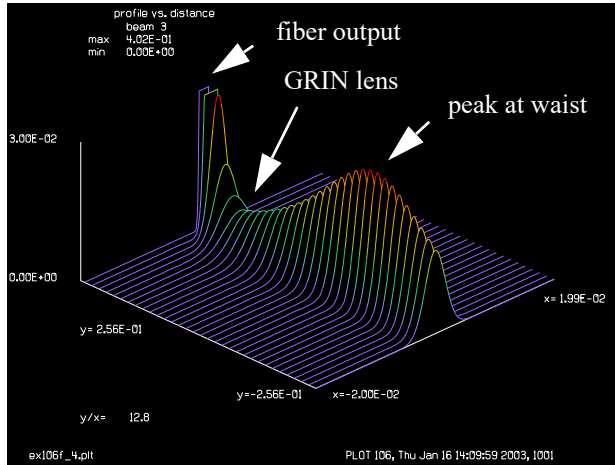


Fig. 106.23. Through-focus display of beam profile as a function of distance from the optical fiber. The display goes to $Z_{total} = 0.512$ cm—beyond the best focus position at about $Z_{total} = 0.375$ cm.

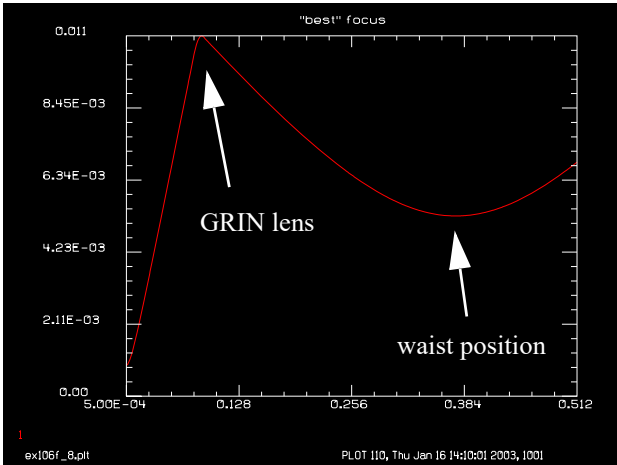


Fig. 106.24. Beam width vs. axial position. The minimum waist is at about 0.375 cm.

```
#
alias Name ex106f
n0 = 1.49                                # index of GRIN
Dz = .0005                               # axial step length
A = .038                                 # GRIN coefficient
a = .00525                               # normalizing radius for GRIN polynomial
mem/set/b 10                             # set memory to 10 megabytes
Nline = 512                              # set number of pixels across array
Nc = Nline/2 + 1                         # center point of array
w0 = .00045
Apt1 = .00625
Apt2 = .00525
array/s 1 Nline                          # set size of Array 1
nbeam 2 Nline data                       # GRIN aberration per step
nbeam 3 Nline 1024 data                  # history of slices vs. length
nbeam 4 Nline 1 data                    # N x 1 array for temporary copy
nbeam 5 Nline data                      # copy of optical beam
nbeam 6 Nline data                      # "best" fit gaussian
Field = .0200                           # half-width covered by array
units/field 1 Field                     # set pixel-to-pixel spacing
units/field 2 Field
units/field 3 Field 1024*Dz/2
units/field 4 Field 1
units/field 5 Field
units/field 6 Field
variab/dec/int count count Ntimes SPLIT PlotCount # define variables
Max = .03
SPLIT = 0
Lambda = 1.32e-4                        # set wavelength variable
wavelength/set 0 Lambda*1e4 1.4467     # set wavelength of all beams
macro/def silica_step/o
# propagate in pure silica
zreff/se 1 0                            # reset initial position
```

Jump to: [Commands](#), [Theory](#)

```

geodata/set/all 1 zwastx=0 zwasty=0      # reset surrogate beam size
count = count + 1
if SPLIT clap/c 1 Apt1                    # absorb at outside of array
prop dz                                  # propagate
Ztotal = Ztotal + dz                      # calculate total distance
fitfwhm 1 1/e^2
variab/set Diameter fitxfwhm
variab/set Peak 1 peak
udata/set count Ztotal Diameter Peak
copy/row 1 4 Nc 1                          # copy center row to beam 4
c peak/norm 4 1                          # normalize beam 4
copy/row 4 3 1 count                      # copy beam 4 to history array 3
if [mod(count,10)==0] then                # plot every 10th time
    plot/w plot1.plt
    plot/l 3 max=Max
endif
macro/end
macro/def grin_step/o
# propagate in GRIN media
zreff/se 1 0
geodata/set/all 1 zwastx=0 zwasty=0
count = count + 1
if SPLIT clap/c 1 Apt2
mult/beam 1 2                            # multiply by GRIN phase per step
prop dz
Ztotal = Ztotal + dz
fitfwhm 1 1/e^2
variab/set Diameter fitxfwhm
variab/set Peak 1 peak
udata/set count Ztotal Diameter Peak
copy/row 1 4 Nc 1
c peak/norm 4 1
copy/row 4 3 1 count
if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
endif
macro/end
macro/def air_step/o
# propagate in air
zreff/se 1 0
geodata/set/all 1 zwastx=0 zwasty=0
count = count + 1
prop dz
Ztotal = Ztotal + dz
fitfwhm 1 1/e^2
variab/set Diameter fitxfwhm
variab/set Peak 1 peak
udata/set count Ztotal Diameter Peak
if [Diameter<BestDiameter] then
    BestZtotal = Ztotal
    BestDiameter = Diameter
endif
copy/row 1 4 Nc 1

```

```

c peak/norm 4 1
copy/row 4 3 1 count
if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
endif
c# if [count==412] macro/exit/all
macro/end
macro/def compare/o
# find correlation with best fit gaussian
copy 1 5
fitfwhm 5 1/e^2
variab/set Width fitxomega
variab/set Peak5 5 peak
gaus/c/c 6 Peak5 Width
mult/mode/corr 5 6
PlotCount=PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/x/i fi=5 la=6
macro/end
#
# Initialize variables and arrays
#
PlotCount = 0
clear 3 0
gaus/c/c 1 1 w0
geodata/set/all 1 waistx=1 waisty=1
c energy/norm 1 1
title SMF-28 to pure silica
mac compare
Ztotal = 0
Ntimes = .0775/Dz list
dz = .0775/Ntimes list
mac silica_step/Ntimes # propagate through pure silica
title pure silica to GRIN
mac compare
Ntimes = .0100/Dz list
dz = .0100/Ntimes list
abr/grin 2 -A*n0/2 2 dz rn=a # calculate GRIN phase per step
lens/flat/sur 1 1.49
c clap/c 1 Apt2
mac grin_step/Ntimes # propage through GRIN media
title GRIN to air
mac compare
lens/flat/sur 1 1
dz = Dz
Ntimes = 1024 - count
wavelength/set 1 Lambda*1e4
BestDiameter = 1e10 # set initial value of best diameter
mac air_step/Ntimes # propagate through air
PlotCount = PlotCount + 1
title profile vs. distance
plot/w @Name_@PlotCount.plt
plot/l 3 max=Max h=.6

```

Jump to: [Commands](#), [Theory](#)

```

PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/c 3 ilab=0
title "best" focus
mac compare
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/x/i fi=5 la=6 le=-.003 ri=.003
energy 1
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/udata min=0
udata/list/disk ex106f.out

```

Ex106g: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), no apertures, best focus

This example illustrates the through-focus beam size of a beam emitted from an optical fiber into a fused silica pipe, terminated by a GRIN lens that focuses the beam into a liquid of index $n = 1.33$. The calculation is ended at $Z_{\text{total}} = 0.375$ which was determined from Ex106f to be the position of the waist. The calculation does not include the multiple steps which were only needed for the through-focus display.

Input: ex106g.inp

```

c## ex106g
#
# Example 106g: Through-focus calculation for fiber-to-fiber system
#
# The objective is to scan the beam profile at all points along the
# path. The system consists of the output of an optical fiber,
# a path through fused silica, a GRIN lens, and an air path.
# The GRIN lens is implemented as abr/grin and a series of
# diffraction steps.
#
# In this case, the GRIN coefficient is alpha=2 with no apertures
# implemented.
#
# We stop at the best focus position (Ztotal = .375) as defined
# by the minimum waist size calculated from ex106a.
#
alias Name ex106g
n0 = 1.49          # index of GRIN
Dz = .0005         # axial step length
A = .038           # GRIN coefficient
a = .00525         # normalizing radius for GRIN polynomial
mem/set/b 10       # set memory to 10 megabytes
Nline = 512        # set number of pixels across array
Nc = Nline/2 + 1   # center point of array
w0 = .00045
Apt1 = .00625
Apt2 = .00525
array/s 1 Nline    # set size of Array 1
nbeam 2 Nline data # GRIN aberration per step
Field = .0200      # half-width covered by array

```

Jump to: [Commands](#), [Theory](#)

```

units/field 1 Field          # set pixel-to-pixel spacing
units/field 2 Field
variab/dec/int count count Ntimes SPLIT PlotCount  # define variables
Max = .03
SPLIT = 0
Lambda = 1.32e-4             # set wavelength variable
wavelength/set 0 Lambda*1e4 1.4467 # set wavelength of all beams
macro/def silica_step/o
# propagate in pure silica
  zreff/set 1 0              # reset initial position
  geodata/set/all 1 zwastx=0 zwasty=0      # reset surrogate beam size
  if SPLIT clap/c 1 Apt1     # absorb at outside of array
  prop dz                    # propagate
  Ztotal = Ztotal + dz      # calculate total distance
macro/end
macro/def grin_step/o
# propagate in GRIN media
  zreff/set 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  if SPLIT clap/c 1 Apt2
  mult/beam 1 2              # multiply by GRIN phase per step
  prop dz
  Ztotal = Ztotal + dz
macro/end
macro/def air_step/o
# propagate in air
  zreff/set 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
  prop dz
  Ztotal = Ztotal + dz
macro/end
#
# Initialize variables and arrays
#
PlotCount = 0
gaus/c/c 1 1 w0
geodata/set/all 1 waistx=1 waisty=1
Ztotal = 0
dz = .0775
mac silica_step              # propagate through pure silica

Ntimes = .0100/Dz list
dz = .0100/Ntimes list
abr/grin 2 -A*n0/2 2 dz rn=a # calculate GRIN phase per step
lens/flat/sur 1 1.49        # change to index 1.49
c clap/c 1 Apt2
mac grin_step/Ntimes        # propagate through GRIN media
lens/flat/sur 1 1          # change to index 1.00

dz = .375 - Ztotal          # best focus distance from ex106f
wavelength/set 1 Lambda*1e4
mac air_step                # propagate through air

```

Jump to: [Commands](#), [Theory](#)


```

title "best" focus
plot/w @Name_1.plt
plot/x/i 1

```

Ex106h: Fiber-to-focus GRIN, aberration ($\alpha = 1.8$), no apertures, through-focus

Ex106h repeats the through-focus calculation with an imperfect GRIN lens having $\alpha = 1.8$, see Figs. 106.26-106.26. By making the index exponent lower than 2, overcorrected spherical aberration is created resulting in a caustic that extends the focus region to the far side of the GRIN lens. Note also that Ex106a, with $\alpha = 2.0$, achieves a higher peak (see Fig. 106.26).

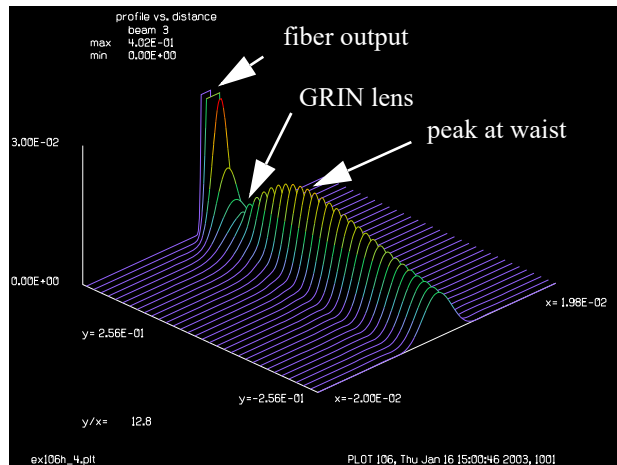


Fig. 106.25. Through-focus display of beam profile as a function of distance from the optical fiber, with aberration included.

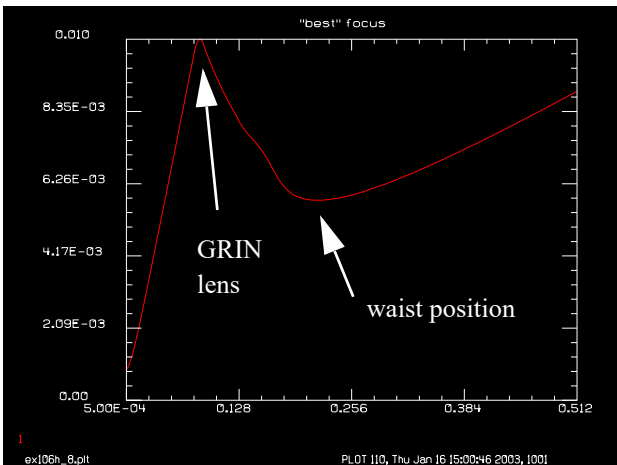


Fig. 106.26. Beam width vs. axial position, with aberration included.

Input: ex106h.inp

```

c## ex106h
#
# Example 106h: Fiber-to-focus, aberration (alpha = 1.8), no apertures
#
# The objective is to scan the beam profile at all points along the
# path. The system consists of the output of an optical fiber,
# a path through fused silica, a GRIN lens, and an air path.
# The GRIN lens is implemented as abr/grin and a series of
# diffraction steps. The GRIN lens uses alpha = 1.8 index as opposed
# to the ideal of alpha = 2.0. No apertures implemented.
#
# We search for the best focus position as defined by minimum
# waist size.
#
alias Name ex106h
n0 = 1.49                # index of GRIN
Dz = .0005               # axial step length
A = .038                 # GRIN coefficient
a = .00525               # normalizing radius for GRIN polynomial
mem/set/b 10             # set memory to 10 megabytes

```

Jump to: [Commands](#), [Theory](#)

```

Nline = 256                # set number of pixels across array
Nc = Nline/2 + 1          # center point of array
w0 = .00045
Apt1 = .00625
Apt2 = .00525
array/s 1 Nline            # set size of Array 1
nbeam 2 Nline data         # GRIN aberration per step
nbeam 3 Nline 1024 data    # history of slices vs. length
nbeam 4 Nline 1 data       # N x 1 array for temporary copy
nbeam 5 Nline data         # copy of optical beam
nbeam 6 Nline data         # "best" fit gaussian
Field = .0200             # half-width covered by array
units/field 1 Field        # set pixel-to-pixel spacing
units/field 2 Field
units/field 3 Field 1024*Dz/2
units/field 4 Field 1
units/field 5 Field
units/field 6 Field
variab/dec/int count count Ntimes SPLIT PlotCount # define variables
Max = .03
SPLIT = 0
Lambda = 1.32e-4           # set wavelength variable
wavelength/set 0 Lambda*1e4 1.4467 # set wavelength of all beams
macro/def silica_step/o
# propagate in pure silica
  zreff/se 1 0              # reset initial position
  geodata/set/all 1 zwastx=0 zwasty=0 # reset surrogate beam size
  count = count + 1
c if SPLIT clap/c 1 Apt1    # absorb at outside of array
  prop dz                  # propagate
  Ztotal = Ztotal + dz     # calculate total distance
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak
  udata/set count Ztotal Diameter Peak
  copy/row 1 4 Nc 1        # copy center row to beam 4
c peak/norm 4 1            # normalize beam 4
  copy/row 4 3 1 count     # copy beam 4 to history array 3
  if [mod(count,10)==0] then # plot every 10th time
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
  endif
macro/end
macro/def grin_step/o
# propagate in GRIN media
  zreff/se 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
c if SPLIT clap/c 1 Apt2
  mult/beam 1 2            # multiply by GRIN phase per step
  prop dz
  Ztotal = Ztotal + dz
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm

```

Jump to: [Commands](#), [Theory](#)

```

variab/set Peak 1 peak
udata/set count Ztotal Diameter Peak
copy/row 1 4 Nc 1
c peak/norm 4 1
copy/row 4 3 1 count
if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
endif
macro/end
macro/def air_step/o
# propagate in air
zreff/se 1 0
geodata/set/all 1 zwastx=0 zwasty=0
count = count + 1
prop dz
Ztotal = Ztotal + dz
fitfwhm 1 1/e^2
variab/set Diameter fitxfwhm
variab/set Peak 1 peak
udata/set count Ztotal Diameter Peak
copy/row 1 4 Nc 1
c peak/norm 4 1
copy/row 4 3 1 count
if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
endif
c# if [count==412] macro/exit/all
macro/end
macro/def compare/o
# find correlation with best fit gaussian
copy 1 5
fitfwhm 5 1/e^2
variab/set Width fitxomega
variab/set Peak5 5 peak
gaus/c/c 6 Peak5 Width
mult/mode/corr 5 6
PlotCount=PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/x/i fi=5 la=6
macro/end
#
# Initialize variables and arrays
#
PlotCount = 0
clear 3 0
gaus/c/c 1 1 w0
geodata/set/all 1 waistx=1 waisty=1
c energy/norm 1 1
title SMF-28 to pure silica
mac compare
Ztotal = 0
Ntimes = .0775/Dz list

```

Jump to: [Commands](#), [Theory](#)

```

dz = .0775/Ntimes list
mac silica_step/Ntimes           # propagate through pure silica
title pure silica to GRIN
mac compare
Ntimes = .0100/Dz list
dz = .0100/Ntimes list
abr/grin 2 -A*n0/2 1.8 dz rn=a   # calculate GRIN phase per step
lens/flat/sur 1 1.49
c clap/c/c 1 Apt2
mac grin_step/Ntimes             # propage through GRIN media
title GRIN to air
mac compare
lens/flat/sur 1 1
dz = Dz
Ntimes = 1024 - count
wavelength/set 1 Lambda*1e4
mac air_step/Ntimes              # propagate through air
PlotCount = PlotCount + 1
title profile vs. distance
plot/w @Name_@PlotCount.plt
plot/l 3 max=Max h=.6
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/c 3 ilab=0
title "best" focus
mac compare
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/x/i fi=5 la=6 le=-.003 ri=.003
energy 1
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/udata min=0
udata/list/disk ex106b.out

```

Ex106i: Fiber-to-focus GRIN, no aberration ($\alpha = 2.0$), apertures implemented, through-focus.

Examples 106i demonstrates the aberration-free case with apertures present. Figs. 106.27 and 106.28 show the results of the through-focus calculations.

Input: ex106i.inp

```

c## ex106i
#
# Example 106i: Fiber-to-fiber, no aberration, apertures
#
# The objective is to scan the beam profile at all points along the
# path. The system consists of the output of an optical fiber,
# a path through fused silica, a GRIN lens, and an air path.
# The GRIN lens is implemented as abr/grin and a series of
# diffraction steps.
#

```

Jump to: [Commands](#), [Theory](#)

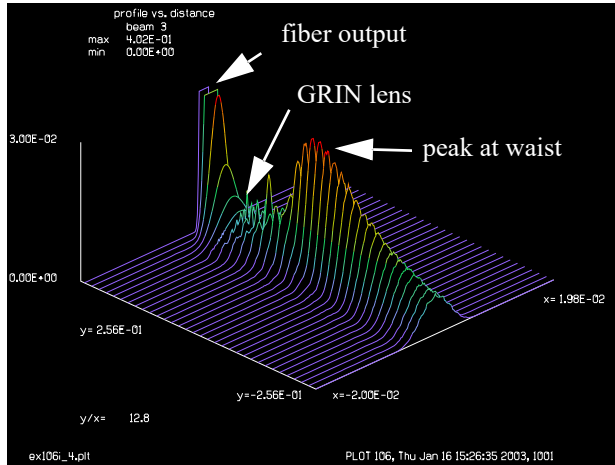


Fig. 106.27. Through-focus display of beam profile as a function of distance from the optical fiber, with apertures included.

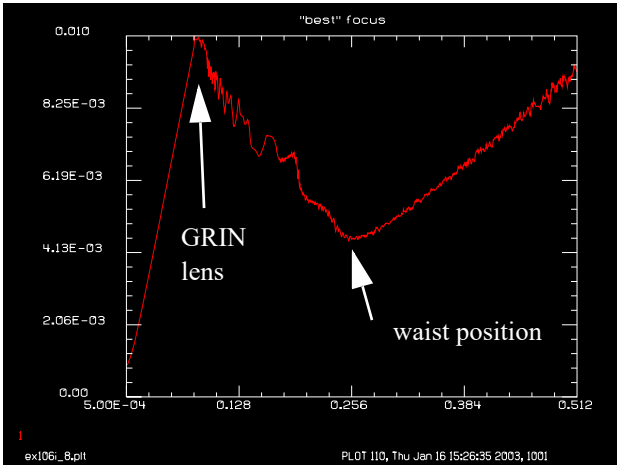


Fig. 106.28. Beam width vs. axial position with apertures included.

```
# Apertures are implemented. Alpha = 2.0 for the index.
#
# We search for the best focus position as defined by minimum
# waist size.
#
#
# alpha = 2, apertures
#
alias Name ex106i
n0 = 1.49                                # index of GRIN
Dz = .0005                               # axial step length
A = .038                                 # GRIN coefficient
a = .00525                               # normalizing radius for GRIN polynomial
mem/set/b 10                             # set memory to 10 megabytes
Nline = 256                              # set number of pixels across array
Nc = Nline/2 + 1                         # center point of array
w0 = .00045
Apt1 = .00625
Apt2 = .00525
array/s 1 Nline                          # set size of Array 1
nbeam 2 Nline data                       # GRIN aberration per step
nbeam 3 Nline 1024 data                  # history of slices vs. length
nbeam 4 Nline 1 data                    # N x 1 array for temporary copy
nbeam 5 Nline data                      # copy of optical beam
nbeam 6 Nline data                      # "best" fit gaussian
Field = .0200                           # half-width covered by array
units/field 1 Field                     # set pixel-to-pixel spacing
units/field 2 Field
units/field 3 Field 1024*Dz/2
units/field 4 Field 1
units/field 5 Field
units/field 6 Field
variab/dec/int count count Ntimes SPLIT PlotCount # define variables
Max = .03
```

Jump to: [Commands](#), [Theory](#)

```

SPLIT = 1
Lambda = 1.32e-4 # set wavelength variable
wavelength/set 0 Lambda*1e4 1.4467 # set wavelength of all beams
macro/def silica_step/o
# propagate in pure silica
  zreff/se 1 0 # reset initial position
  geodata/set/all 1 zwastx=0 zwasty=0 # reset surrogate beam size
  count = count + 1
c if SPLIT clap/c 1 Apt1 # absorb at outside of array
  prop dz # propagate
  Ztotal = Ztotal + dz # calculate total distance
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak
  udata/set count Ztotal Diameter Peak
  copy/row 1 4 Nc 1 # copy center row to beam 4
c peak/norm 4 1 # normalize beam 4
  copy/row 4 3 1 count # copy beam 4 to history array 3
  if [mod(count,10)==0] then # plot every 10th time
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
  endif
macro/end
macro/def grin_step/o
# propagate in GRIN media
  zreff/se 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
c if SPLIT clap/c 1 Apt1
  mult/beam 1 2 # multiply by GRIN phase per step
  prop dz
  Ztotal = Ztotal + dz
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak
  udata/set count Ztotal Diameter Peak
  copy/row 1 4 Nc 1
c peak/norm 4 1
  copy/row 4 3 1 count
  if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
  endif
macro/end
macro/def air_step/o
# propagate in air
  zreff/se 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
  prop dz
  Ztotal = Ztotal + dz
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak

```

Jump to: [Commands](#), [Theory](#)

```

    udata/set count Ztotal Diameter Peak
    copy/row 1 4 Nc 1
c   peak/norm 4 1
    copy/row 4 3 1 count
    if [mod(count,10)==0] then
        plot/w plot1.plt
        plot/l 3 max=Max h=.6
    endif
c#   if [count==412] macro/exit/all
macro/end
macro/def compare/o
# find correlation with best fit gaussian
    copy 1 5
    fitfwhm 5 1/e^2
    variab/set Width fitxomega
    variab/set Peak5 5 peak
    gaus/c/c 6 Peak5 Width
    mult/mode/corr 5 6
    PlotCount=PlotCount + 1
    plot/w @Name_@PlotCount.plt
    plot/x/i fi=5 la=6
macro/end
#
# Initialize variables and arrays
#
PlotCount = 0
clear 3 0
gaus/c/c 1 1 w0
geodata/set/all 1 waistx=1 waisty=1
c energy/norm 1 1
title SMF-28 to pure silica
mac compare
Ztotal = 0
Ntimes = .0775/Dz list
dz = .0775/Ntimes list
mac silica_step/Ntimes           # propagate through pure silica
title pure silica to GRIN
mac compare
Ntimes = .0100/Dz list
dz = .0100/Ntimes list
abr/grin 2 -A*n0/2 2 dz rn=a     # calculate GRIN phase per step
lens/flat/sur 1 1.49
clap/c 1 Apt2
mac grin_step/Ntimes           # propage through GRIN media
title GRIN to air
mac compare
lens/flat/sur 1 1
dz = Dz
Ntimes = 1024 - count
wavelength/set 1 Lambda*1e4
mac air_step/Ntimes           # propagate through air
PlotCount = PlotCount + 1
title profile vs. distance
plot/w @Name_@PlotCount.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/1 3 max=Max h=.6
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/c 3 ilab=0
title "best" focus
mac compare
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/x/i fi=5 la=6 le=-.003 ri=.003
energy 1
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/udata min=0
udata/list/disk ex106i.out

```

Ex106j: Fiber-to-focus GRIN, aberration ($\alpha = 1.8$), apertures implemented

Ex106j shows the combined effects of aberration ($\alpha = 1.8$) and apertures. See Figs. 106.29 and 106.30.

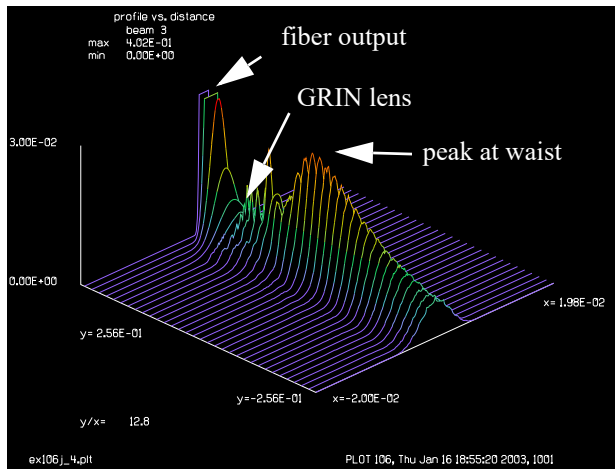


Fig. 106.29. Through-focus display of beam profile as a function of distance from the optical fiber, with apertures included.

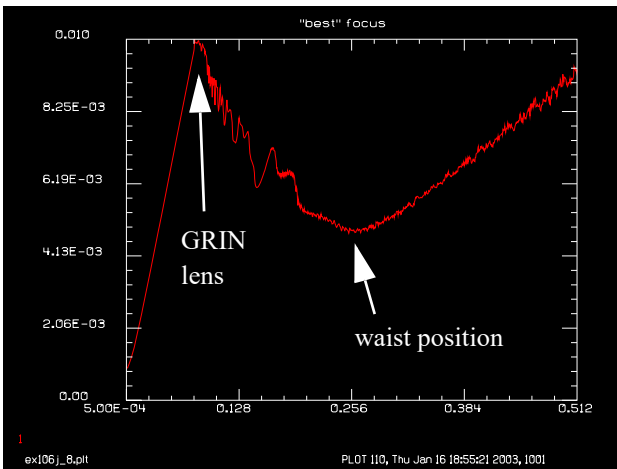


Fig. 106.30. Beam width vs. axial position with apertures included.

Input: ex106j.inp

```

c## ex106j
#
# Example 106j: Fiber-to-Focus, aberration, apertures
#
# The objective is to scan the beam profile at all points along the
# path. The system consists of the output of an optical fiber,
# a path through fused silica, a GRIN lens, and an air path.
# The GRIN lens is implemented as abr/grin and a series of
# diffraction steps.
#
# Apertures are implemented. Alpha = 1.8 for the index
# and apertures are implemented.
#

```

Jump to: [Commands](#), [Theory](#)


```

# We search for the best focus position as defined by minimum
# waist size.
#
#
# alpha = 1.8, apertures
#
alias Name ex106j
n0 = 1.49                # index of GRIN
Dz = .0005              # axial step length
A = .038                # GRIN coefficient
a = .00525              # normalizing radius for GRIN polynomial
mem/set/b 10            # set memory to 10 megabytes
Nline = 256             # set number of pixels across array
Nc = Nline/2 + 1        # center point of array
w0 = .00045
Apt1 = .00625
Apt2 = .00525
array/s 1 Nline         # set size of Array 1
nbeam 2 Nline data      # GRIN aberration per step
nbeam 3 Nline 1024 data # history of slices vs. length
nbeam 4 Nline 1 data    # N x 1 array for temporary copy
nbeam 5 Nline data      # copy of optical beam
nbeam 6 Nline data      # "best" fit gaussian
Field = .0200           # half-width covered by array
units/field 1 Field     # set pixel-to-pixel spacing
units/field 2 Field
units/field 3 Field 1024*Dz/2
units/field 4 Field 1
units/field 5 Field
units/field 6 Field
variab/dec/int count count Ntimes SPLIT PlotCount # define variables
Max = .03
SPLIT = 1
Lambda = 1.32e-4        # set wavelength variable
wavelength/set 0 Lambda*1e4 1.4467 # set wavelength of all beams
macro/def silica_step/o
# propagate in pure silica
  zreff/se 1 0           # reset initial position
  geodata/set/all 1 zwastx=0 zwasty=0 # reset surrogate beam size
  count = count + 1
c if SPLIT clap/c 1 Apt1 # absorb at outside of array
  prop dz                # propagate
  Ztotal = Ztotal + dz   # calculate total distance
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak
  udata/set count Ztotal Diameter Peak
  copy/row 1 4 Nc 1      # copy center row to beam 4
c peak/norm 4 1          # normalize beam 4
  copy/row 4 3 1 count   # copy beam 4 to history array 3
  if [mod(count,10)==0] then # plot every 10th time
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
  endif
endif

```

Jump to: [Commands](#), [Theory](#)

```

macro/end
macro/def grin_step/o
# propagate in GRIN media
  zreff/se 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
c  if SPLIT clap/c 1 Apt2
  mult/beam 1 2                                # multiply by GRIN phase per step
  prop dz
  Ztotal = Ztotal + dz
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak
  udata/set count Ztotal Diameter Peak
  copy/row 1 4 Nc 1
c  peak/norm 4 1
  copy/row 4 3 1 count
  if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
  endif
macro/end
macro/def air_step/o
# propagate in air
  zreff/se 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
  prop dz
  Ztotal = Ztotal + dz
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak
  udata/set count Ztotal Diameter Peak
  copy/row 1 4 Nc 1
c  peak/norm 4 1
  copy/row 4 3 1 count
  if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
  endif
c#  if [count==412] macro/exit/all
macro/end
macro/def compare/o
# find correlation with best fit gaussian
  copy 1 5
  fitfwhm 5 1/e^2
  variab/set Width fitxomega
  variab/set Peak5 5 peak
  gaus/c/c 6 Peak5 Width
  mult/mode/corr 5 6
  PlotCount=PlotCount + 1
  plot/w @Name_@PlotCount.plt
  plot/x/i fi=5 la=6
macro/end

```

Jump to: [Commands](#), [Theory](#)

```

#
# Initialize variables and arrays
#
PlotCount = 0
clear 3 0
geodata/set/all 1 waistx=1 waisty=1
gaus/c/c 1 1 w0
c energy/norm 1 1
title SMF-28 to pure silica
mac compare
Ztotal = 0
Ntimes = .0775/Dz list
dz = .0775/Ntimes list
mac silica_step/Ntimes           # propagate through pure silica
title pure silica to GRIN
mac compare
Ntimes = .0100/Dz list
dz = .0100/Ntimes list
abr/grin 2 -A*n0/2 1.8 dz rn=a   # calculate GRIN phase per step
lens/flat/sur 1 1.49
clap/c 1 Apt2
mac grin_step/Ntimes            # propage through GRIN media
title GRIN to air
mac compare
lens/flat/sur 1 1
dz = Dz
Ntimes = 1024 - count
wavelength/set 1 Lambda*1e4
mac air_step/Ntimes             # propagate through air
PlotCount = PlotCount + 1
title profile vs. distance
plot/w @Name_@PlotCount.plt
plot/l 3 max=Max h=.6
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/c 3 ilab=0
title "best" focus
mac compare
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/x/i fi=5 la=6 le=-.003 ri=.003
energy 1
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/udata min=0
udata/list/disk gamma8.out

```

Ex106k: Optimization of fiber-to-fiber GRIN lens system (paraxial)

Examples Ex106k and Ex106l optimize the length of the silica rod and the GRIN medium. Ex106k does optimization of the paraxial system, while Ex106l performs optimization of the system with aberration and propagation. The light is coupled into the final rod by using the `mult/mode/parallel` command. The intermediate waist is to have a radius of 20 microns. We use the length of the silica rod and the GRIN lens

Jump to: [Commands](#), [Theory](#)

as free variables and solve for the intermediate waist radius and total throughput into the rod at the end. Example 106k illustrates a first order solution using the `lens/grin` command, which can treat $\alpha = 2$ and no apertures. Ex106l uses the `abr/grin` command which allows values of α other than 2 and finite aperture clipping. Ex 106m shows the through-focus beams size. Each step is peak-normalized.

Input: ex106k.inp

```
c## ex106k
#
# Example: Optimization of fiber-to-fiber GRIN lens system (paraxial)
#
alias Name ex106k
variab/dec/int count Nline
Lambda = 1.32e-4
gamma2 = 1.378685e3
w0 = .00045
n0 = 1.49
Nline = 128
array/s 1 Nline
gamma = sqrt(gamma2) list
Field = .0050
Lsilica = .06279274
LGRIN = .01259572
L = .25
nbeam 2 data
wavelength/set 2 Lambda*1e4 1.4467
units/field 2 Field
macro/def system/o
  array/s 1 Nline
  zreff/se 1 0
  wavelength/set 1 Lambda*1e4 1.4467
  units/field 1 Field
  gaus/c/c 1 1 w0
  energy/norm 1 1
  prop Lsilica 1
  plot/w @Name_1.plt
  plot/x/i
  lens/flat/sur 1 1.49
  lens/grin 1 gamma LGRIN
  lens/flat/sur 1 1.332
  prop L
  fitgeo 1
  variab/set Width fitxomega
  prop L
  lens/flat/sur 1 1.49
  lens/grin 1 gamma LGRIN
  lens/flat/sur 1 1.4467
  prop Lsilica
  units/beam 2 1
  gaus/c/c 2 1 w0
  energy/norm 2 1
  plot/w @Name_2.plt
  plot/x/i fi=1 la=2
```

Jump to: [Commands](#), [Theory](#)

```

    mult/mode/parallel 1 2
    variab/set Energy 1 energy
    Energy1 = Energy*Weight
macro/end
Weight=.003
opt/var/add Lsilica .0002
opt/var/add LGRIN .0002
opt/tar/add Width .0020
opt/tar/add Energy1 Weight
opt/name system
c opt/run 1
mac system

```

Ex106l: Optimization of fiber-to-fiber GRIN lens system (phase aberration model)

Example 106l performs optimization of the system with a full treatment of the aberration and propagation. The solutions agree quite closely.

Input: ex106l.inp

```

c## ex106l
#
# Examples: Optimization of fiber-to-fiber GRIN lens system
# (phase aberration model)
#
# Ex106l: Command file to optimize waist width and position
#
alias Name ex106l
n0 = 1.49          # index of GRIN
Dz = .0005         # axial step length
A = .038           # GRIN coefficient
a = .00525         # normalizing radius for GRIN polynomial
mem/set/b 10       # set memory to 10 megabytes
Nline = 512        # set number of pixels across array
Nc = Nline/2 + 1   # center point of array
w0 = .00045
L = .25
Apt1 = .00625
Apt2 = .00625
alpha = 2
Lsilica = .06295965
LGRIN = .01262016
array/s 1 Nline    # set size of Array 1
nbeam 4 Nline data # GRIN aberration per step
Field = w0*25      # half-width covered by array
units/field 1 Field # set pixel-to-pixel spacing
units/field 2 Field
units/field 3 Field
units/field 4 Field
gaus/c 4 1 w0
variab/dec/int count count1 Ntimes APT PlotCount i # define variables
Max = 0

```

Jump to: [Commands](#), [Theory](#)

```

APT = 0
Lambda = 1.32e-4 # set wavelength variable
wavelength/set 0 Lambda*1e4 1.4467 # set wavelength of all beams
macro/def grin_step/o
# propagate in GRIN media
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
  if APT clap 1 Apt2
  mult/beam 1 3 # multiply by GRIN phase per step
  prop dz
  Ztotal = Ztotal + dz
macro/end
#
# Initialize variables and arrays
#
PlotCount = 0
gaus/c/c 2 1 w0
energy/norm 2 1
c energy/norm 1 1
peak/norm 1 1
title SMF-28 to pure silica
c mac compare
mac/def system/o
  count1 = count1 + 1 list
  zreff/se 0 0
  array/s 1 Nline
  wavelength/set 1 Lambda*1e4 1.4467 # set wavelength of all beams
  units/field 1 Field # set pixel-to-pixel spacing
  gaus/c/c 1 1 w0
  energy/norm 1 1
  geodata/set/all 1 waistx=1 waisty=1
  prop Lsilica
  title pure silica to GRIN
  Ntimes = 20
  dz = LGRIN/Ntimes list
  lens/flat/sur 1 1.49
  clear 3 1
  abr/grin 3 -A*n0/2 alpha dz rn=a # calculate GRIN phase per step
  mac grin_step/Ntimes # propagate through GRIN media
  title GRIN to air
  lens/flat/sur 1 1.332
  prop L
  plot/w @Name_2.plt
  plot/x/i 1 le=-.0060 ri=.0060 fmin=0
  fitfwhm 1 1/e^2
  variab/set Width fitxfwhm
  fitgeo 1
  variab/set Width fitxomega
  prop L
  lens/flat/sur 1 1.49
  mac grin_step/Ntimes # propagate through GRIN media
  lens/flat/sur 1. 1.4467
  prop Lsilica
  mult/mode/parallel 1 2

```

Jump to: [Commands](#), [Theory](#)

```

variab/set Energy energy
plot/w @Name_1.plt
plot/x/i fi=1 la=2
Energy1 = Energy*Weight
Lsilica=
LGRIN=
Width=
Energy=
c write/off
  zreff/se 1 0
mac/end
Weight=.003
opt/var/add Lsilica .0001
opt/var/add LGRIN .00002
opt/tar/add Width .0020
opt/tar/add Energy1 Weight
opt/name system
c opt/run 1
# read/screen not recognized in optimization loop
mac system

```

Ex106m: Optimization of fiber-to-fiber GRIN lens system

Example 106m performs optimization of the system with a full treatment of the aberration and propagation but no apertures.

Input: ex106m.inp

```

c## ex106m
#
# Example: Optimization of fiber-to-fiber GRIN lens system
# (phase aberration model)
#
# compute for alpha=2, no apertures
#
alias Name ex106m
n0 = 1.49 # index of GRIN
Lsilica = .06295965
LGRIN = .01262016
Lwater = .5
Dz = (Lwater+2*(Lsilica+LGRIN))/1023 list # axial step length
Dz = .00064
A = .038 # GRIN coefficient
a = .00525 # normalizing radius for GRIN polynomial
mem/set/b 10 # set memory to 10 megabytes
Nline = 512 # set number of pixels across array
Nc = Nline/2 + 1 # center point of array
w0 = .00045
Apt1 = .00625
Apt2 = .00525
array/s 1 Nline # set size of Array 1
nbeam 2 Nline data # GRIN aberration per step
nbeam 3 Nline 1024 data # history of slices vs. length

```

Jump to: [Commands](#), [Theory](#)

```

nbeam 4 Nline 1 data      # N x 1 array for temporary copy
nbeam 5 Nline data        # copy of optical beam
nbeam 6 Nline data        # "best" fit gaussian
Field = .0100             # half-width covered by array
units/field 1 Field       # set pixel-to-pixel spacing
units/field 2 Field
units/field 3 Field 1024*Dz/2
units/field 4 Field 1
units/field 5 Field
units/field 6 Field
variab/dec/int count count Ntimes SPLIT PlotCount  # define variables
Max = .06
SPLIT = 0
Lambda = 1.32e-4          # set wavelength variable
wavelength/set 0 Lambda*1e4 1.4467  # set wavelength of all beams
macro/def silica_step/o
# propagate in pure silica
  zreff/se 1 0             # reset initial position
  geodata/set/all 1 zwastx=0 zwasty=0  # reset surrogate beam size
  count = count + 1
  if [count==1025] macro/exit
  if SPLIT clap/c 1 Apt1   # absorb at outside of array
  prop dz                  # propagate
  Ztotal = Ztotal + dz     # calculate total distance
  copy/row 1 4 Nc 1        # copy center row to beam 4
c  peak/norm 4 1           # normalize beam 4
  copy/row 4 3 1 count     # copy beam 4 to history array 3
  if [mod(count,10)==0] then # plot every 10th time
    plot/w plot1.plt
    plot/l 3 max=Max
  endif
macro/end
macro/def grin_step/o
# propagate in GRIN media
  zreff/se 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
  if [count==1025] macro/exit
  if SPLIT clap/c 1 Apt2
  mult/beam 1 2            # multiply by GRIN phase per step
  prop dz
  Ztotal = Ztotal + dz
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak
  udata/set count Ztotal Diameter Peak
  copy/row 1 4 Nc 1
c  peak/norm 4 1
  copy/row 4 3 1 count
  if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
  endif
macro/end

```

Jump to: [Commands](#), [Theory](#)


```

macro/def water_step/o
# propagate in water
  zreff/se 1 0
  geodata/set/all 1 zwastx=0 zwasty=0
  count = count + 1
  if [count==1025] macro/exit
  prop dz
  Ztotal = Ztotal + dz
  fitfwhm 1 1/e^2
  variab/set Diameter fitxfwhm
  variab/set Peak 1 peak
  udata/set count Ztotal Diameter Peak
  copy/row 1 4 Nc 1
c peak/norm 4 1
  copy/row 4 3 1 count
  if [mod(count,10)==0] then
    plot/w plot1.plt
    plot/l 3 max=Max h=.6
  endif
c# if [count==412] macro/exit/all
macro/end
macro/def compare/o
# find correlation with best fit gaussian
  copy 1 5
  fitfwhm 5 1/e^2
  variab/set Width fitxomega
  variab/set Peak5 5 peak
  gaus/c/c 6 Peak5 Width
  mult/mode/corr 5 6
  PlotCount=PlotCount + 1
  plot/w @Name_@PlotCount.plt
  plot/x/i fi=5 la=6
macro/end
#
# Initialize variables and arrays
#
PlotCount = 0
clear 3 0
gaus/c/c 1 1 w0
geodata/set/all 1 waistx=1 waisty=1
c energy/norm 1 1
title SMF-28 to pure silica
mac compare
Ztotal = 0
Ntimes = Lsilica/Dz list
dz = Lsilica/Ntimes list
mac silica_step/Ntimes # propagate through pure silica
title pure silica to GRIN
mac compare
lens/flat/sur 1 1.49
Ntimes = LGRIN/Dz list
dz = LGRIN/Ntimes list
abr/grin 2 -A*n0/2 2 dz rn=a # calculate GRIN phase per step
mac grin_step/Ntimes # propagate through GRIN media

```

Jump to: [Commands](#), [Theory](#)

```

title GRIN to water
mac compare
lens/flat/sur 1 1.332
dz = Dz
Ntimes = Lwater/Dz
dz = Lwater/Ntimes
mac water_step/Ntimes                # propagate through water

lens/flat/sur 1 1.49
Ntimes = LGRIN/Dz list
dz = LGRIN/Ntimes list
abr/grin 2 -A*n0/2 2 dz rn=a        # calculate GRIN phase per step
mac grin_step/Ntimes                # propagate through GRIN media
Ntimes = Lsilica/Dz list
dz = Lsilica/Ntimes list
mac silica_step/Ntimes              # propagate through pure silica
PlotCount = PlotCount + 1
title profile vs. distance
plot/w @Name_@PlotCount.plt
plot/l 3 max=Max h=.6
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/c 3 peak=1 ilab=0
title "best" focus
mac compare
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/x/i fi=5 la=6 le=-.003 ri=.003
energy 1
PlotCount = PlotCount + 1
plot/w @Name_@PlotCount.plt
plot/udata min=0
udata/list/disk @Name.out

```

Ex106n: Example of multi-mode diode laser

An example of a multi-mode laser diode. For laser diodes that have a long emitting region, perhaps a length of 50 to 100 microns, the radiation tends to break into filaments. The size of the typical filament determines the divergence angle of the diode in the respective direction. These devices usually operate as a single mode in the narrow direction. In GLAD we model the laser diode accurately by treating the radiation as a coherent beam with instantaneous speckled structure to represent the filamentation. The beam is propagated through the optical system—just as the actual beam propagates—and the resulting intensity distribution is integrated to simulate the time-averaged result.

To add a more complex system, continue the propagation of Beam 1 through lenses, mirrors, apertures, etc. and place the integration step at the end. Propagate through the system for each pass.

Input: ex106n.inp

```

c## ex106n
#
# Example: Multi-mode diode.

```

Jump to: [Commands](#), [Theory](#)

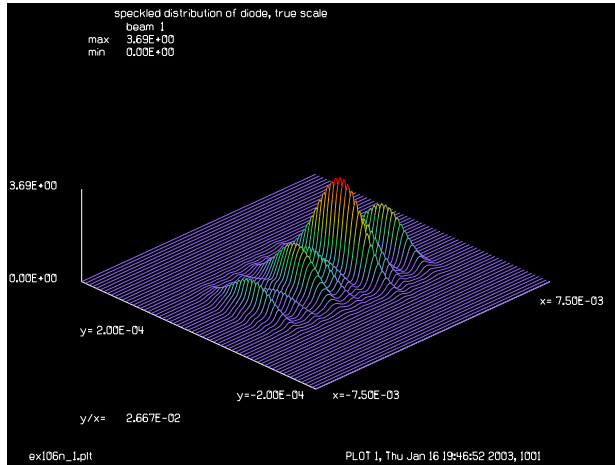


Fig. 106.31. Instantaneous view of laser diode showing speckled distribution in x-direction.

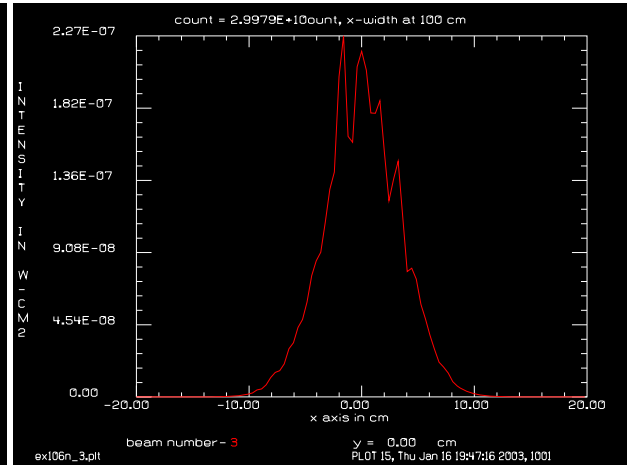


Fig. 106.32. The x-direction, far-field profile after averaging over 100 instances.

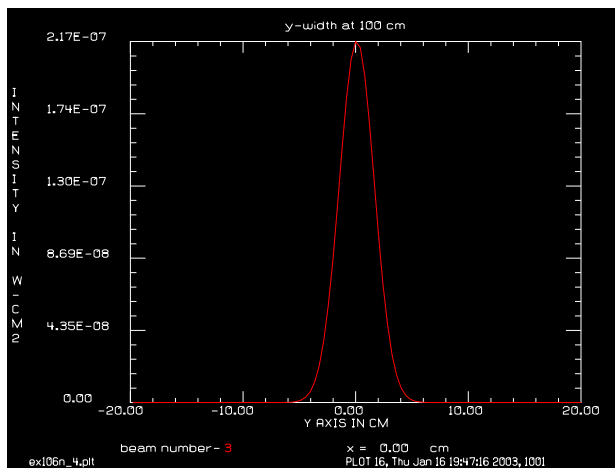


Fig. 106.33. The y-direction far-field profile after averaging.

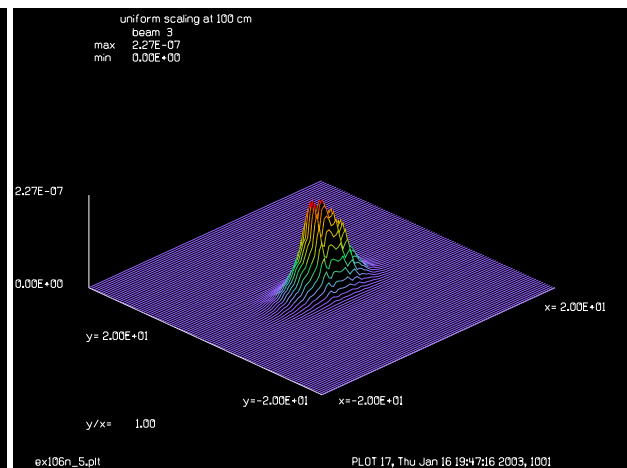


Fig. 106.34. Far-field distribution after averaging showing x-width due to multi-mode operation.

```
#
# For laser diodes that have a long emitting region, perhaps a length
# of 50 to 100 microns, the radiation tends to break into filaments.
# The size of the typical filament determines the divergence angle of
# the diode in the respective direction. These devices usually operate
# as a single mode in the narrow direction.
#
# In GLAD we model the laser diode accurately by treating the radiation
# as a coherent beam with instantaneous speckled structure to represent
# the filamentation. The beam is propagated through the optical system
# -- just as the actual beam propagates -- and the resulting intensity
# distribution is integrated to simulate the time-averaged result.
#
# To add a more complex system, continue the propagation of Beam 1 through
# lenses, mirrors, apertures, etc. and place the integration step at the
# end. Propagate through the system for each pass.
#
```

Jump to: [Commands](#), [Theory](#)

```

# Beams
# -----
# 1      coherent propagating beam
# 2      smoothed, speckled structure
# 3      time-integrated result
#
alias Name ex106n
variab/dec/int Count Nline
Nline = 256
mem/set/b 8
array/s 1 Nline
nbeam 3 data
units/field 0 .0100 .0016
Lambda = .808e-4
speckle_size = .0004 # guess speckle size at 4 microns diameter
                    # for y-direction
X_width = .00500 # x-half width
Y_width = .00005 # y-half width
wavelength/set 1 Lambda*10000
# take a guess at "typical" beam size to set up scaling
clear 3 0
macro/def step/o
  Count = Count+1
  array/s 1 Nline
  zreff/se 1 0
  units/field 1 .0100 .0008
  gaus/r/c 1 1 .0050 .00007 10 1
  # now refill the array with speckled data
c clear 2 0
c noise 2 1
c convol/gaus 2 speckle_size # smooth the random noise to generate speckle
                             # of specified size
noise/smooth 2 1 speckle_size
mult/beam 1 2
if [Count==1] then
  plot/w @Name_1.plt
  title speckled distribution of diode, true scale
  plot/l ns=64 xrad=.0075 yrad=.0002
  title speckled distribution of diode, true scale
  plot/w @Name_2.plt
  plot/i
endif
prop 100
if [Count==1] then
  variab/set Units 1 units
  units/set 3 Units
endif
add/inc/c 3 1
if [mod(Count,20)==0] then
  plot/w @Name_3.plt
  title count = @count, x-width at 100 cm
  plot/x/i 3 le=-20 ri=20 fmin=0
  plot/w @Name_4.plt
  title y-width at 100 cm

```

Jump to: [Commands](#), [Theory](#)

```
plot/y/i 3 le=-20 ri=20 fmin=0
plot/w @Name_5.plt
title uniform scaling at 100 cm
plot/l 3 ns=256 xrad=20 yrad=20 min=0
endif
macro/end
write/off
macro step/100
write/on
```


Ex107: Sum frequency generation (SFG)

Table. 107.1. Table of Ex107 examples

Ex107a: SFG, plane wave case	1
Ex107b: SFG, gaussian beams, distributed propagation steps.....	3
Ex107c: SFG, gaussian beams, aberration, distributed propagation steps.	5

This example illustrates sum frequency generation. A pump beam of wavelength $1.06\ \mu$ is mixed with a second beam of $10.6\ \mu$ to create a beam of $0.963636\ \mu$. In Ex107a.inp, two collimated beams are mixed to create the sum frequency beam using the values presented in Section 9.8, GLAD Theory Manual. Ex107b uses the wavelengths with gaussian beam profiles. The energy exchange from Beam 1 to Beams 2 and 3 is shown in steps through the 1 cm width of the SFG medium. Example 107c shows the same case as Ex107b.inp, with the addition of aberration. The aberration lowers the efficiency of the conversion.

Ex107a: SFG, plane wave case

Input: `ex107a.inp`

Ex107b: c## ex107a

```

c
c Example 107a: Sum-frequency generation (SFG): plane wave beams
c -----
c
nbeam 3                # Define three beams
lambda_1 = 1.06E-4      # pump, vacuum wavelength
lambda_2 = 10.6E-4      # mix beam, vacuum wavelength
lambda_3 = .963636E-4   # up-converted beam
#
# vacuum wavelength for the up-converted beam may be solved by the
# condition that # the sum of frequencies is zero -- identical to
# the sum of inverse vacuum wavelengths
#
hbar = h/2/pi
variab/dec/int Nline Xcent Ycent A1 A2 A3
Nline = 256
Xcent = Nline/2 + 1
Ycent = Nline/2 + 1
A1 = 1
A2 = 3
A3 = 2
array/set A1 Nline Nline      # Set array dimensions
array/set A2 Nline Nline 1     # Set array dimensions

```

```

array/set A3 Nline Nline      # Set array dimensions

lambda_3 = 1./(1./lambda_1 + 1./lambda_2)# up-converted, vacuum wavelength
n_1 = 2.6                      # index beam 1
n_3 = 2.597                    # index beam 3
n_o = 2.5                      # index beam 2, ordinary ray
n_e = 2.653080                 # index beam 2, extraordinary ray
Chi2 = 1.1E-20                 # cross section
DeltaK_set = 0.                # Explicitly set for GLAD calculation
wavelength/set A1 lambda_1*1E4 n_1  # Define wavelength of pump beam
wavelength/set A2 lambda_2*1E4 n_o n_e # Define wavelength of signal
wavelength/set A3 lambda_3*1E4 n_3  # Define wavelength of up-convert
n_2 = lambda_2*(n_3/lambda_3 - n_1/lambda_1)
C C Desired beam 2 index, n_2 = @n_2  # desired index for sum beam
WaveNumber1 = 2.*pi*n_1/lambda_1
WaveNumber2 = 2.*pi*n_2/lambda_2
WaveNumber3 = 2.*pi*n_3/lambda_3
delta_k = WaveNumber1 + WaveNumber2 - WaveNumber3
C C Calculation of delta_k from derivation: delta_k = @delta_k
polar = 42.6894
x = cosd(polar)/n_o
y = sind(polar)/n_e
n_2c = 1./sqrt(x^2 + y^2)
C C Desired index, n_2 = @n_2, for polar angle @polar, n_2c = @n_2c
x1 = 2*pi*n_1/lambda_1
x2 = 2*pi*n_2/lambda_2
x3 = 2*pi*n_3/lambda_3
delta_k = x1 + x2 - x3
delta_k = 2*pi*(n_1/lambda_1 + n_2c/lambda_2 - n_3/lambda_3)
C C Calculation of delta_k from index ellipsoid: delta_k = @delta_k
crystal/uniaxial/set polar 0
units/field 0 .15              # Set units
I1 = 1e4
I2 = 1.
I3 = 0.
clear A1 I1                    # Define pump intensity
clear A2 I2                    # Define signal intensity
clear A3 I3                    # Define idler intensity
echo/on
jones/set ar=0 cr=1
echo/off
jones/mult A2
udata/clear                    # Clear user data variables

```



```

wave4/set 0.                # Set zstart for four-wave mixing
pack/set A1 A2 A3           # Define arrays for memory processing
point/list/ij A1 Xcent Ycent
variab/set I1_1 point/i
point/list/ij A2 Xcent Ycent
variab/set I1_2 point/i
point/list/ij A3 Xcent Ycent
variab/set I1_3 point/i
udata/set 1 0. I1_1 I1_2 I1_3 # First call to user data
zstep = 1                   # Set step length
macro/def kinstep/over
  pack/in
C C call sfg
  sfg zstep=zstep chi2=Chi2 nstep=1 deltak=DeltaK_set xyx list
pack/out
pass = pass + 1
z = z + zstep
variab/set Peak1 A1 peak
variab/set Peak2 A2 peak
variab/set Peak3 A3 peak
udata/set pass z Peak1 Peak2 Peak3
udata
macro/end
c
c Run calculations
c
variab/set energy1 A1 energy # Define Param 1 = Energy of Beam 1
variab/set energy2 A2 energy # Define Param 2 = Energy of Beam 2
variab/set energy3 A3 energy # Define Param 3 = Energy of Beam 3
Engtot1 = energy1 + energy2 + energy3
z = 0.                        # Initialize z-axis
pass = 1                      # Initialize udata index
omega_1 = 2.*pi*c/lambda_1
omega_2 = 2.*pi*c/lambda_2
omega_3 = 2.*pi*c/lambda_3
C C optical frequencies 1, 2, 3: @omega_1, @omega_2, @omega_3
C C call kinstep
macro/run kinstep/1          # call propagation macro
point/list/ij A1 Xcent Ycent
variab/set I2_1 point/i
point/list/ij A2 Xcent Ycent
variab/set I2_2 point/i
point/list/ij A3 Xcent Ycent

```

```

variab/set I2_3 point/i
I1_total = I1_1 + I1_2 + I1_3
I2_total = I2_1 + I2_2 + I2_3
C C Starting intensities: I1_1 = @I1_1, I1_2 = @I1_2, I1_3 = @I1_3, I1_total = @I1_total
C C Ending intensities: I2_1 = @I2_1, I2_2 = @I2_2, I2_3 = @I2_3, I2_total = @I2_total
dP1 = (I2_1 - I1_1)/omega_1/hbar
dP2 = (I2_2 - I1_2)/omega_2/hbar
dP3 = (I2_3 - I1_3)/omega_3/hbar
C C photon changes from GLAD: dP1 = @dP1, dP2 = @dP2, dP3 = @dP3
variab/set energy1 A1 energy
variab/set energy2 A2 energy
variab/set energy3 A3 energy
Engtot2 = energy1 + energy2 + energy3
Engtot_error_GLAD = 100*(Engtot2 - Engtot1)/Engtot1
Engtot_error_GLAD = 100*(I2_1+I2_2+I2_3 - (I1+I2+I3))/(I1+I2+I3)
C C GLAD, ending intensities: Beam 1 @I2_1, Beam 2 @I2_2, Beam 3 @I2_3
C C GLAD, incremental photon changes: Beam 1 @dP1, Beam 2 @dP2, Beam 3 @dP3
C C GLAD, energy conservation error: @Engtot_error_GLAD percent
C
C C Hand calculation of equation
C
Mu0 = 4*pi*1E-7
Epsilon0 = 8.85E-12
Kappa = 1/sqrt(2)*omega_2*(Mu0/Epsilon0)^.75/sqrt(n_1*n_2*n_3)*Chi2
C C Cross section:      Kappa = @Kappa
g0 = 2*sqrt(omega_3/omega_2)*Kappa*sqrt(I1_1)
g = sqrt(g0^2 - DeltaK_set^2)
C C Simple gain:      g0 = @g0
C C Delta-k modified gain:      g = @g
C2 = (-DeltaK_set*sqrt(I1) + sqrt(omega_2/omega_3)*g0*sqrt(I3))/g
C3 = (-DeltaK_set*sqrt(I3) + sqrt(omega_3/omega_2)*g0*sqrt(I2))/g
C C Harmonic coefficients,      C2 = @C2
C C Harmonic coefficient:      C3 = @C3
K1 = 2.*pi*n_1/lambda_1
K2 = 2.*pi*n_2/lambda_2
K3 = 2.*pi*n_3/lambda_3
delta_K_calculated = K1 + K2 - K3
C C Delta-K calculations      delta_K_calculated = @delta_K_calculated
arg = g*zstep/2
cosarg = cos(arg)
sinarg = sin(arg)
C C cos(g*zstep/2):      cosarg = @cosarg
C C sin(g*zstep/2):      sinarg = @sinarg

```

```

A2_z = sqrt(I2)*cosarg
A3_z = C3*sqrt(I2)*sinarg
C C 2nd pump amplitude:      A2_z = @A2_z
C C Sum beam amplitude:     A3_z = @A3_z
I2_z = A2_z^2
I3_z = A3_z^2
C C 2nd pump intensity:      I2_z = @I2_z
C C Sum beam intensity:      I3_z = @I3_z
dP2_z = (I2_z - I2)/omega_2/hbar
dP3_z = (I3_z - I3)/omega_3/hbar
C C 2nd pump, incremental photons: dP2_z = @dP2_z
C C Sum beam, incremental photons: dP3_z = @dP3_z
c
c Calculate 1st pump amplitude change
c
Tmp0 = omega_1/omega_2/g*Kappa
Tmp1 = sqrt(I2)*sqrt(I3) - C2*C3
Tmp2 = sqrt(I2)*sqrt(I3) + C2*C3
Tmp3 = C2*sqrt(I3) - C3*sqrt(I2)
Tmp4 = 2*sin(arg)^2
dA1 = Tmp0*Tmp3*Tmp4
C C Change in 1st pump amplitude: dA1 = @dA1
A1_z = sqrt(I1)+dA1
C C New 1st pump amplitude:    A1_z = @A1_z
I1_z = A1_z^2
C C Starting 1st pump intensity: I1_1 = @I1_1
C C Ending 1st pump intensity:  I1_z = @I1_z
dI1 = I1_z - I1_1
C C 1st pump intensity change:  dI1 = @dI1
P1_1 = I1_1/omega_1/hbar
C C Starting beam 1 photons:   P1_1 = @P1_1
P1_z = I1_z/omega_1/hbar
C C Ending beam 1 photons:     P1_z = @P1_z
dP12_avg = 0.5*(dP2_z - dP3_z)
C C Average photon changes 2,3: dP12_avg = @dP12_avg
dP1_z = P1_z - P1_1
C C Beam 1 photon change:      dP1_z = @dP1_z
Amp_Adjust = sqrt((P1_1 + dP12_avg)/P1_z)
C C Amplitude adjustment factor: Amp_Adjust = @Amp_Adjust
A1_z_adj = A1_z*Amp_Adjust
I1_z_adj = A1_z_adj^2
P1_z_adj = I1_z_adj/omega_1/hbar
dP1_z_adj = P1_z_adj - P1_1

```

```

C C Adjusted beam 1 photon change: dP1_z_adj = @dP1_z_adj
C
Engtot_error_Calc = 100*(I1_z+I2_z+I3_z - (I1+I2+I3))/(I1+I2+I3)
C C GLAD, ending intensities: Beam 1 @I2_1, Beam 2 @I2_2, Beam 3 @I2_3
C C Calc, ending intensities: Beam 1 @I1_z_adj, Beam 2 @I2_z, Beam 3 @I3_z
C C GLAD, incremental photon changes: Beam 1 @dP1, Beam 2 @dP2, Beam 3 @dP3
C C Calc, Incremental photon changes: Beam 1 @dP1_z_adj, Beam 2 @dP2_z, Beam 3 @dP3_z
C C GLAD, energy conservation error: @Engtot_error_GLAD percent
C C Calc, energy conservation error: @Engtot_error_Calc percent

```

SFG, gaussian beams, distributed propagation steps

In this example, gaussian beam are used for Beams 1 and 2 are gaussian profile. We can observe the walk-off of Beam 2 due to the properties of the uniaxial crystal in Fig. 107.1. Both Beams 2 and 3 increase due energy taken from Beam 1, as shown in Fig. 107.2.

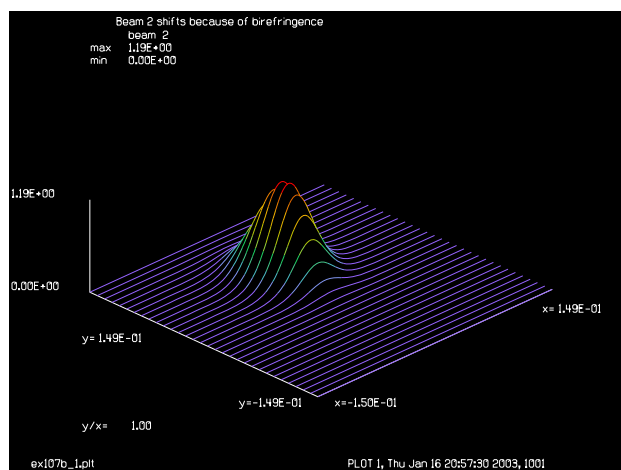


Fig. 107.1. Beam 2 (10.6 μ) after SFG shows upward shift due to uniaxial crystal.

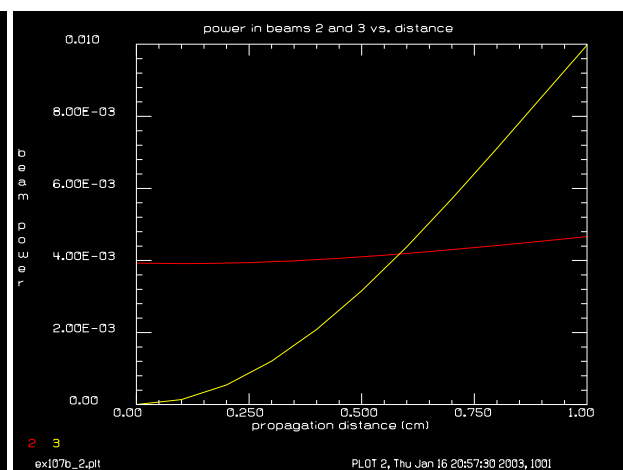


Fig. 107.2. Energy of Beams 2 and 3 (0.963636 μ) through the SFG region with no aberrations.

Input: ex107b.inp

Ex107c: c## ex107b

```

c
c Example 107b: Sum-frequency generation (SFG): gaussian beams, tuned
c -----
c
nbeam 3                # Define three beams
lambda_1 = 1.06         # pump, vacuum wavelength
lambda_2 = 10.6         # mix beam, vacuum wavelength

```

Jump to: [Commands](#), [Theory](#)

```

lambda_3 = .963636          # up-converted beam
#
# vacuum wavelength for the up-converted beam may be solved by the
# condition that # the sum of frequencies is zero -- identical to
# the sum of inverse vacuum wavelengths
#
variab/dec/int Nline Xcent Ycent
Nline = 256
Xcent = Nline/2 + 1
Ycent = Nline/2 + 1
array/set 1 Nline Nline      # Set array dimensions
array/set 2 Nline Nline 1     # Set array dimensions
array/set 3 Nline Nline      # Set array dimensions

lambda_3 = 1./(1./lambda_1 + 1./lambda_2)
lambda_3=                    # up-converted, vacuum wavelength
n_1 = 2.6
n_3 = 2.6
n_o = 2.5
n_e = 2.653080
wavelength/set 1 lambda_1 n_1  # Define wavelength of pump beam
wavelength/set 2 lambda_2 n_o n_e  # Define wavelength of signal
wavelength/set 3 lambda_3 n_3      # Define wavelength of up-convert
n_2 = lambda_2*(n_1/lambda_1 - n_3/lambda_3)
n_2=                            # desired index for signal
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_2 = 1./sqrt(x^2 + y^2)
n_2=                            # calculated index for signal
delta_k = n_1/lambda_1 + n_2/lambda_2 - n_3/lambda_3 list
delta_k=
crystal/uniaxial/set polar 0
units/field 0 .15              # Set units
gaus/c/c 1 1e4 .05
gaus/c/c 2 1. .05
clear 3 0.                     # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
wave4/set 0.                   # Set zstart for four-wave mixing
pack/set 1 2 3                 # Define arrays for memory processing
zstep = .1                     # Set step length
c

```

```

c Define macro for kinetic step
c
macro/def kinstep/over
  pack/in
    sfg zstep=zstep chi2=1.1e-20 nstep=1 xyx
  pack/out
  pass = pass + 1
  z = z + zstep
  variab/set Energy1 1 energy      # Define Param 1 = Energy of Beam 1
  variab/set Energy2 2 energy      # Define Param 2 = Energy of Beam 2
  variab/set Energy3 3 energy      # Define Param 3 = Energy of Beam 3
  udata/set pass z Energy1 Energy2 Energy3
macro/end
c
c Run calculations
c
variab/set Energy1 1 energy      # Define Param 1 = Energy of Beam 1
variab/set Energy2 2 energy      # Define Param 2 = Energy of Beam 2
variab/set Energy3 3 energy      # Define Param 3 = Energy of Beam 3
Engtot1 = Energy1 + Energy2 + Energy3
z = 0.                          # Initialize z-axis
pass = 1                        # Initialize udata index
udata/set pass z Energy1 Energy2 Energy3
omega_1 = 2.*pi*c/lambda_1 list
omega_2 = 2.*pi*c/lambda_2 list
omega_3 = 2.*pi*c/lambda_3 list
macro/run kinstep/10           # call propagation macro
plot/w ex107b_1.plt
title Beam 2 shifts because of birefringence
plot/l 2
variab/set Energy1 1 energy
variab/set Energy2 2 energy
variab/set Energy3 3 energy
Engtot2 = Energy1 + Energy2 + Energy3
Engtot1=
Engtot2=
x=Engtot2-Engtot1 list
c
c Set up plot titles
c
udata/xlab propagation distance (cm)
udata/ylab beam power
title SFG amplification

```

```

udata/list
plot/watch ex107b_2.plt
title power in beams 2 and 3 vs. distance
plot/udata 2 3 min=0 max=.01
end
SFG, gaussian beams, aberration, distributed propagation steps.

```

Ex107c.inp adds aberration to Beam 1 at the start of the SFG region. The aberration inprints into intensity modulation in both Beam 2 (shown in Fig. 107.3) and in Beam 3. The gain is lowered by the aberration as well as illustrated in Fig. 107.4 when compared with Fig. 107.4.

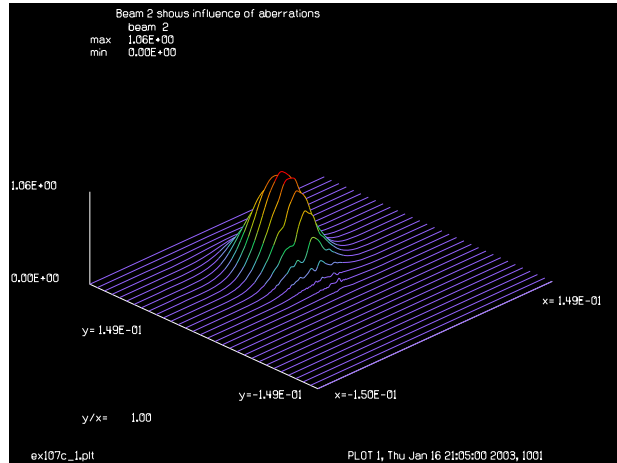


Fig. 107.3. Beam 2 (10.6μ) after SFG shows upward shift due to uniaxia crystal, with aberration added to Beam 1. Compare with Fig. 107.1.

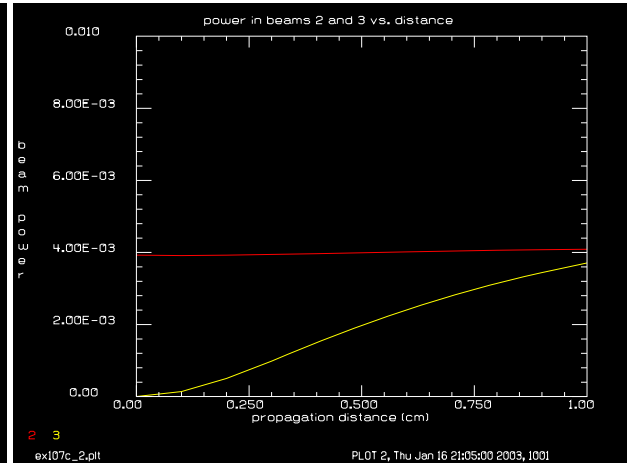


Fig. 107.4. Enegy of Beams 2 and 3 (0.963636μ) through the SFG region, with aberrations added to Beam 1. Compare with Fig. 107.2.

Input: ex107c.inp

```

c## ex107c
c
c Example 107c: Sum-frequency generation (SFG): gaussian beams, aberration
c -----
c
nbeam 3                                # Define three beams
lambda_1 = 1.06                        # pump, vacuum wavelength
lambda_2 = 10.6                        # mix beam, vacuum wavelength
lambda_3 = .963636                    # up-converted beam
#
# vacuum wavelength for the up-converted beam may be solved by the
# condition that # the sum of frequencies is zero -- identical to
# the sum of inverse vacuum wavelengths
#
variab/dec/int Nline Xcent Ycent
Nline = 256
Xcent = Nline/2 + 1
Ycent = Nline/2 + 1
array/set 1 Nline Nline                # Set array dimensions
array/set 2 Nline Nline 1              # Set array dimensions

```

Jump to: [Commands](#), [Theory](#)

```

array/set 3 Nline Nline                                # Set array dimensions

lambda_3 = 1./(1./lambda_1 + 1./lambda_2)
lambda_3=                                              # up-converted, vacuum wavelength
n_1 = 2.6
n_3 = 2.6
n_o = 2.5
n_e = 2.653080
wavelength/set 1 lambda_1 n_1                          # Define wavelength of pump beam
wavelength/set 2 lambda_2 n_o n_e                      # Define wavelength of signal
wavelength/set 3 lambda_3 n_3                          # Define wavelength of up-convert
n_2 = lambda_2*(n_1/lambda_1 - n_3/lambda_3)
n_2=                                                  # desired index for signal
polar = 55.133
x = cos(polar*pi/180.)/n_o
y = sin(polar*pi/180.)/n_e
n_2 = 1./sqrt(x^2 + y^2)
n_2=                                                  # calculated index for signal
delta_k = n_1/lambda_1 + n_2/lambda_2 - n_3/lambda_3 list
delta_k=
crystal/uniaxial/set polar 0
units/field 0 .15                                     # Set units
gaus/c/c 1 1e4 .05
phase/random 1 2 .02
gaus/c/c 2 1. .05
clear 3 0.                                             # Define idler intensity
jones/set ar=0 cr=1
jones/mult 2
wave4/set 0.                                           # Set zstart for four-wave mixing
pack/set 1 2 3                                         # Define arrays for memory processing
zstep = .1                                             # Set step length
c
c Define macro for kinetic step
c
macro/def kinstep/over
    pack/in
        sfg zstep=zstep chi2=1.1e-20 nstep=1 xyx
    pack/out
    pass = pass + 1
    z = z + zstep
    variab/set Energy1 1 energy                        # Define Param 1 = Energy of Beam 1
    variab/set Energy2 2 energy                        # Define Param 2 = Energy of Beam 2
    variab/set Energy3 3 energy                        # Define Param 3 = Energy of Beam 3
    udata/set pass z Energy1 Energy2 Energy3
macro/end
c
c Run calculations
c
variab/set Energy1 1 energy                            # Define Param 1 = Energy of Beam 1
variab/set Energy2 2 energy                            # Define Param 2 = Energy of Beam 2
variab/set Energy3 3 energy                            # Define Param 3 = Energy of Beam 3
Engtot1 = Energy1 + Energy2 + Energy3
z = 0.                                                 # Initialize z-axis
pass = 1                                               # Initialize udata index

```

Jump to: [Commands](#), [Theory](#)


```

udata/set pass z Energy1 Energy2 Energy3
omega_1 = 2.*pi*c/lambda_1 list
omega_2 = 2.*pi*c/lambda_2 list
omega_3 = 2.*pi*c/lambda_3 list
macro/run kinstep/10                                # call propagation macro
plot/w ex107c_1.plt
title Beam 2 shows influence of aberrations
plot/l 2
variab/set Energy1 1 energy
variab/set Energy2 2 energy
variab/set Energy3 3 energy
Engtot2 = Energy1 + Energy2 + Energy3
Engtot1=
Engtot2=
x=Engtot2-Engtot1 list
c
c  Set up plot titles
c
udata/xlab propagation distance (cm)
udata/ylab beam power
title SFG amplification
udata/list
plot/watch ex107c_2.plt
title power in beams 2 and 3 vs. distance
plot/udata 2 3 min=0 max=.01
end

```


Ex108: Fan-out grating

This example illustrates the design of a fan-out grating to create eleven peaks of equal height in the far-field of a lens. Four harmonic orders of phase modulation are combined. Damped least squares optimization is used to find the coefficients that produce the result. A special smoothing step is included to suppress the side lobes beyond the eleven peaks and the tendency of the design to have the extreme peaks too high.

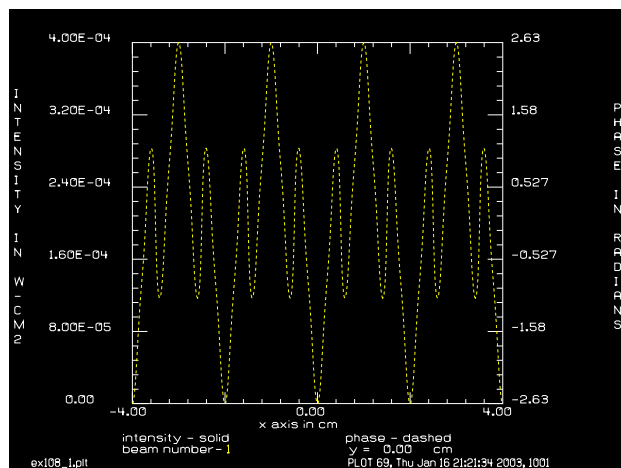


Fig. 108.1. Phase modulation of solution showing the effect of multiple harmonics.

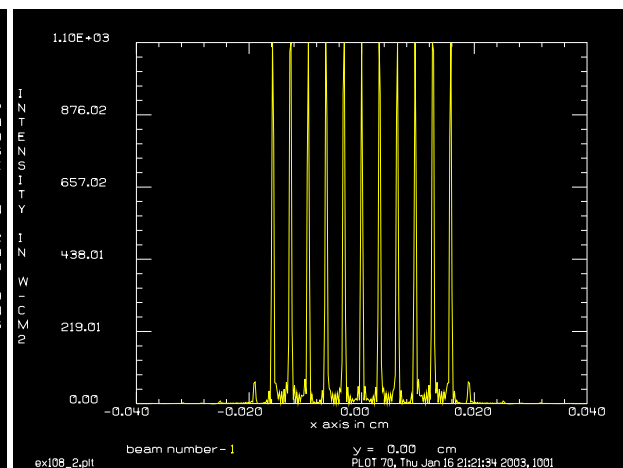


Fig. 108.2. Far-field irradiance showing eleven equal peaks and minimal extraneous scattering.

Input: ex108.inp

```
c## ex108
#
# Example 108: Design of a fan-out grating
#
# This command file illustrates solution of a fan-out grating
# with 11 peaks.
#
# Four harmonics of cosine phase modulation are mixed.
#
# Some smoothing of the phase function by convolution reduces
# both the end peaks and the side lobe values outside the
# region of the eleven peaks

variab/dec/int Pass Nline
Nline = 16384
array/s 1 Nline 1
nbeam 2
wavelength/set 0 0.3511
Wavenum = 2.5
Rnorm = 5
Period = Rnorm/Wavenum
shift = .3511e-4*180/Period
clap=.001

# period determines spacing of
# first side lobe
# spacing between side lobes
# aperture used for sensing side
# lobe peak
```

```

macro/def loop/o
  zreff/se 1 0
  array/reset 1
  units/s 1 .001875
#
# phase modulation harmonics
#
abr/lrip 1 a Wavenum 90 90 Rnorm
abr/lrip 1 b Wavenum*3 90 90 Rnorm
abr/lrip 1 c1 Wavenum*5 90 90 Rnorm
abr/lrip 1 d Wavenum*7 90 90 Rnorm
convol/gaus 1 Smooth # smoothing to limit wide side bands
clap/r/c 1 7. 1. # define aperture
plot/watch ex108_1.plt
plot/x 1 le=-4 ri=4 fmin=0 fmax=4e-4
lens/xcyl 1 180.
dist 180. 1 xonly
#
# find peak values of various side lobes
#
copy 1 2
clap/r/c 2 clap .5 # order 0
variab/set t1 2 peak list
copy 1 2
clap/r/c 2 clap .5 shift # order 1
variab/set t2 2 peak list
copy 1 2
clap/r/c 2 clap .5 shift*2 # order 2
variab/set t3 2 peak list
copy 1 2
clap/r/c 2 clap .5 shift*3 # order 3
variab/set t4 2 peak list
copy 1 2
clap/r/c 2 clap .5 shift*4 # order 4
variab/set t5 2 peak list
copy 1 2
clap/r/c 2 clap .5 shift*5 # order 5
variab/set t6 2 peak list
copy 1 2
clap/r/c 2 clap .5 shift*6 # order 6
variab/set t7 2 peak list
copy 1 2
clap/r/c 2 clap .5 shift*7 # order 7
variab/set t8 2 peak list
#
# compute target values for optimization
#
tt1 = t1 - t2 # 0 - 1st order difference
tt2 = t1 - t3 # 0 - 2nd order difference
tt3 = t1 - t4 # 0 - 3rd order difference
tt4 = t1 - t5 # 0 - 4th order difference
tt5 = t1 - t6 # 0 - 5th order difference
tt6 = t1 - t7 # 0 - 6th order difference, computed but not used
macro/end

```

Jump to: [Commands](#), [Theory](#)

```

mac/def opt1/o
  Pass = Pass+1
  opt/run
  variab/set/par Merit merit
  udata/set Pass Merit a b
  udata/set Pass y06=tt1 y07=tt2 y08=tt3 y09=tt4 y10=tt5
  plot/watch ex108_2.plt
  plot/x/i 1 le=-.04 ri=.04 fmin=0
macro/end
opt/nam loop
#
#  starting values
#
a = .224084821455
b = .258843445949
c1 = -6.761791900362E-02
d = .106502168145
Smooth = .1084
opt/var/add a .001
opt/var/add b .001
opt/var/add c1 .001
opt/var/add d .001
opt/tar/add tt1 0. # order 1
opt/tar/add tt2 0. # order 2
opt/tar/add tt3 0. # order 3
opt/tar/add tt4 0. # order 4
opt/tar/add tt5 0. # order 5
mac opt1/10

```


Ex109: Flat-flat and polygon resonators

Table. 109.1. Table of Ex109 examples

Ex109a: Flat-flat bare cavity resonator	1
Ex109b: Polygon bare cavity resonator	2

These examples illustrate flat-flat resonators and polygon resonators. The flat-flat configuration is on the boundary between stable and unstable devices. The special resonator commands are not helpful. It is necessary to explicitly reset the `Zreff` position and the surrogate gaussian beam each pass. When the global positioning commands are used, it is necessary to reinitialize the global commands as well.

Slight positive power may be treated best by using a phase lens.

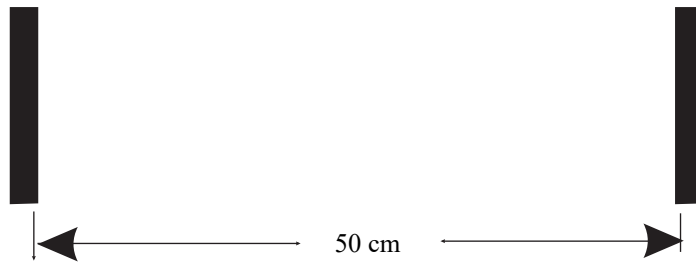


Fig. 109.1. Flat-flat resonator for Ex109a.

Ex109a: Flat-flat bare cavity resonator

A polygon resonator is basically a flat-flat resonator with mirror folds. It is treated in much the same way except the global position is reset at each pass to avoid creep due to slight numerical errors. Also see Example 33c.

Input: `ex109a.inp`

Ex109b: c## ex109a

```
#
# Example 109a: Example of flat-flat resonator
#
# Weak positive power is added with the use of a phase lens.
#
# bare cavity analysis with aperture of 0.2
variab/dec/int Array Pass
Pass= 0
MirrorApertur = 0.2
GainLength    = 18
FocalLength= 5000
ResonLength= 50
```

```

PropLength1= (ResonLength-GainLength)/2
PropLength2= (ResonLength-GainLength)/2
Array = 256
Field = .75
WaveL = 1.064e-4
array/set 1 Array
units/field 0 Field
wavelength/set 0 WaveL*1e4
plot/watch ex109_1.plt
macro/def reson/o
  Pass = Pass + 1
  prop PropLength1
  clap/c/n 1 MirrorApertur
  lens/sph/ele/phase 1 FocalLength # phase lens for weak power
  prop PropLength2
  mirror/flat 1
  prop PropLength2
  clap/c/n 1 MirrorApertur
  lens/sph/ele/phase 1 FocalLength # phase lens for weak power
  prop PropLength1
  mirror/flat 1
  energy/norm
  plot/l 1
macro/end
clear 1 0
noise 1 1
reson/name reson
reson/eigen/test 1
reson/eigen/list
reson/run 1000
Polygon bare cavity resonator

```

Input: ex109b.inp

```

c## ex109b
#
# Example 109b: polygon resonator
#
# This example illustrates a polygon resonator. It is functionally
# the same as a flat-flat resonator, such as Ex109a.
#
# The resonator commands are not helpful for this configuration.
# The zreff position, surrogate gaussian beam parameters, and the
# global coordinates are reset each pass.
#
mem/set/b 2

```

Jump to: [Commands](#), [Theory](#)


```

variab/dec/int Array Pass Nsides SideNo
Pass = 0
MirrorApertur = 0.2
GainLength = 18
FocalLength= 5000
LengthPerSide= 20
Array = 256
Field = .75
WaveL = 1.064e-4
array/set 1 Array
units/field 0 Field
wavelength/set 0 WaveL*1e4
Nsides = 8
Theta = 2.*pi/Nsides
macro/def side/o
#
# Macro for each side
#
    SideNo = SideNo + 1      # increment side number counter
    prop LengthPerSide      # propagate length between sides
    vertex/locate/rel       # relocate vertex to current beam position
    vertex/rotate/set ry=(SideNo-.5)*360/Nsides
    mirror/global/flat 1    # global flat mirror
    clap/c/n 1 MirrorApertur
c phase lens adds effect of slightly curved mirror
    lens/toric/ele/phase 1 FocalLength/cos(Theta) FocalLength
macro/end

macro/def reson/o
#
# Macro for each pass through the resonator
#
    Pass = Pass + 1         # increment pass counter
    zreff/set 1 0          # trim reference position
    global/def 1           # trim global position
c reset surrogate gaussian parameters
    geodata/set/all 1      zwaistx=0    zwaisty=0    waistx=MirrorApertur
    waisty=MirrorApertur
    SideNo = 0             # initialize side number counter
    macro side/Nsides      # call side macro
    plot/l 1
macro/end
clear 1 0
noise 1 1
macro reson/100

```


Ex110: Beam reshaping optics

This example illustrates beam reshaping optics that transform a gaussian beam into a flat-top distribution. Spherical aberration, including higher order terms, is introduced to cause the far-field distribution to have the proper irradiance distribution. A second aspheric lens is used to correct the spherical aberration so that the beam exiting the second lens can be essentially aberration-free.

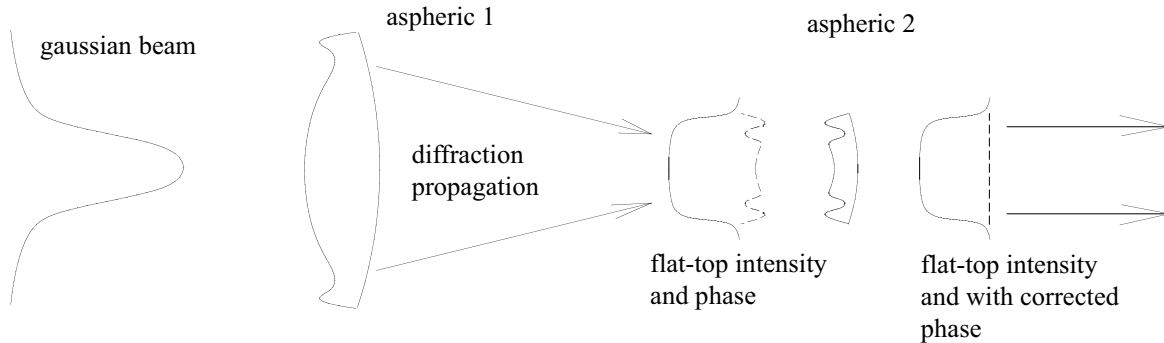


Fig. 110.1. Aspheric 1 adds special spherical aberration of high order such that the beam becomes nearly flat-top when propagated. The resulting beam has strong phase aberrations. Aspheric 2 compensates the aberrations yielding a beam which has both nearly flat intensity and flat phase. This beam will propagate as a flat-top, collimated beam.

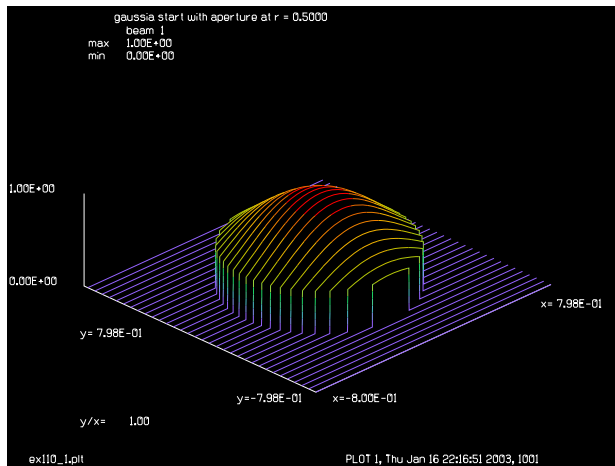


Fig. 110.2. Starting distribution, clipped.

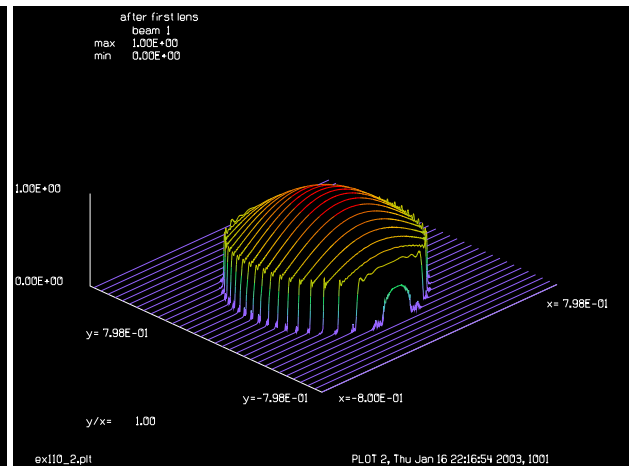


Fig. 110.3. Intensity after passing through the first aspheric lens. Showing some edge diffraction.

Input: `ex110.inp`

```
c## ex110
#
# Example 110: Beam reshaping optics, gaussian to flat-top
#
# This example illustrates beam reshaping optics that transform
# a gaussian beam into a flat-top distribution. Spherical aberration,
# including higher order terms, is introduced to cause the far-field
# distribution to have the proper irradiance distribution. A second
```

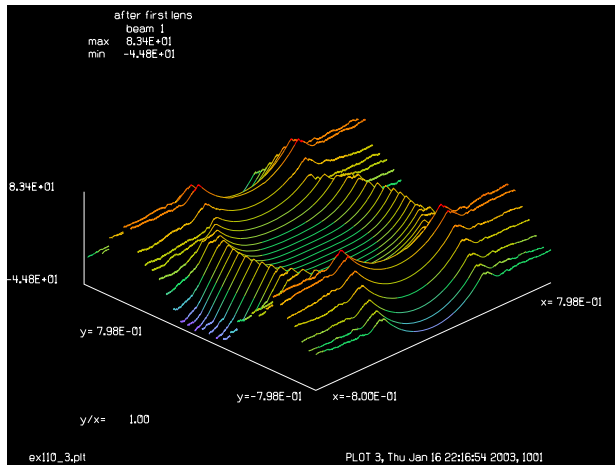


Fig. 110.4. Phase after first aspheric lens. The cusp in the phase is due to aliasing outside the aperture.

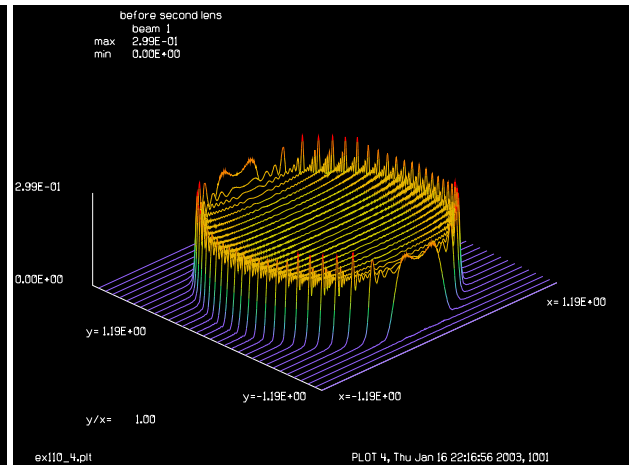


Fig. 110.5. Nearly flat intensity after propagation.

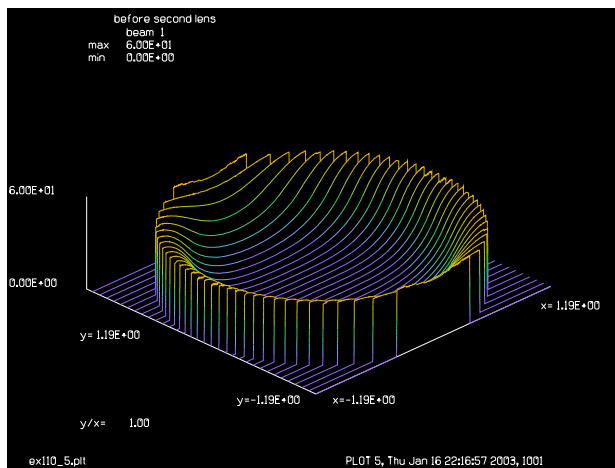


Fig. 110.6. Residual phase after correction by the second aspheric—not perfect.

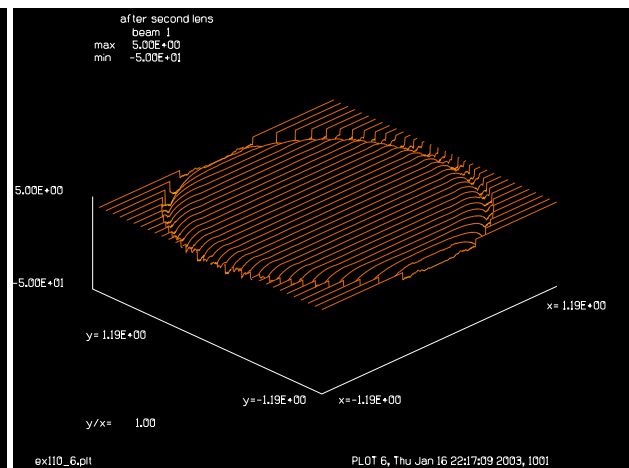


Fig. 110.7. Phase after correction by fitzern shows nearly flat phase.

```
# aspheric lens is used to correct the spherical aberration so that
# the beam exiting the second lens can be essentially aberration-free.
#
mem/set/b 8
array/set 1 1024
nbeam 1
wavelength/set 1 .441
units/field 0 .8      # The radius of the Gaussian is 0.8cm, and
variab/set/par unit1 1 units list
c                      # after it passed throught an afocal system,
c                      # it is expanded into a 1.25 radius beam.
c
c Set variables
gaussian/cir/con 1 1 .8 1.
c
c# the 1. is the supergaussian exponent, see the equation in the
c# manual
```

Jump to: [Commands](#), [Theory](#)

```

radius = .5
clap/c/c 1 radius
title gaussia start with aperture at r = @radius
plot/w ex110_1.plt
plot/l
yo = 0.
obj_dist = .8/.5e-10
pupil_dist = 0.
lensgroup/def asp1/o
  object lambda=.441
  surface_lensgroup 1e10 1. caf2
  surface_lensgroup 4.786175 0. -1.11436 -7.15329e-2 3.37289e-2 -1.49168e-2 &
  5.98365e-3 -1.51665e-3 air
  image focus
lensgroup/end
lensgroup/trace/yfan/beam asp1
lensgroup/run/radial asp1
geodata
title after first lens
plot/w ex110_2.plt
plot/l
plot/w ex110_3.plt
plot/l/w
vertex/locate/rel 1 0 0 15
prop/vertex
variab/set/par unit2 1 units list
clap/c/c 1 radius*unit2/unit1
title before second lens
plot/w ex110_4.plt
plot/l xrad=radius*unit2/unit1
geodata
lensgroup/def asp2/o
  object lambda=.441
  surface_lensgroup 11.364905 1. -1.48771 -2.63246e-3 9.40588e-4 -2.3098e-3 &
  1.58396e-3 -4.84387e-4 caf2
  surface_lensgroup 1e10 1e10 air
  image
lensgroup/end
clap/c/c 1 radius*unit2/unit1
plot/w ex110_5.plt
plot/l/w 1 xrad=radius*unit2/unit1 max=60 min=0
lensgroup/run/radial asp2
clap/c/c 1 radius*unit2/unit1
#
# This design does not remove all aberration after the second lens.
# To show how much aberration can be removed, fitzern is used.
#
fitzern/remove/radial
title after second lens
clap/c/c 1 radius*unit2/unit1
plot/w ex110_6.plt
plot/l/w 1 xrad=radius*unit2/unit1 max=5 min=-50
prop 1
title after 1 cm propagation

```

Jump to: [Commands](#), [Theory](#)

```
plot/w ex110_7.plt  
plot/l
```

Ex111: Guide star, ground-to-space

This illustrates scattering of a beam from the sodium layer at about 90 km above the earth. In this example a gaussian beam is projected upward by a telescope (see Fig. 111.1). The beam is aberrated by the effects of atmospheric turbulence created a spot at the sodium layer larger than the diffraction limit. The spot jitters over time at the sodium layer as well. The sodium layer acts as a delta-correlated scattering source with an overall envelope limited by the intensity of the image. To avoid aliasing difficulties, a finite size of scattering aberration is used, limiting the angle of the light projected back to the ground. Provided the projected scattered light overfills the receiving aperture (in this case the same as the transmitting aperture) the received illumination will be essentially the same as if a true delta-correlated source were used.

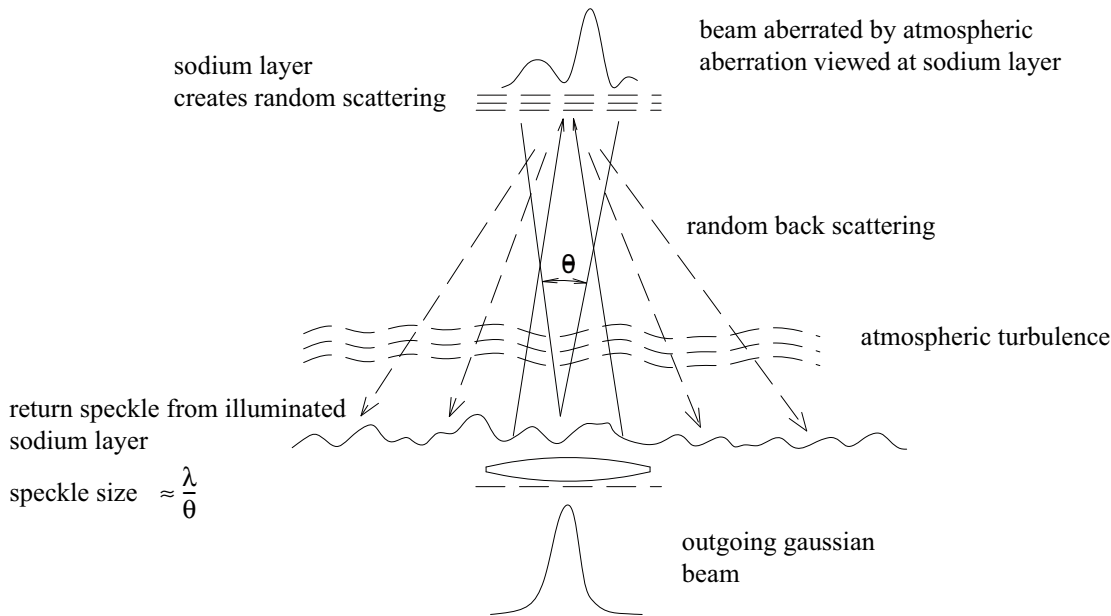


Fig. 111.1. The outgoing beam (Fig. 111.2) is distorted by atmosphere to create aberrated beam (Fig. 111.3) in upper atmosphere at sodium layer at 90 km altitude. Sodium layer is a random scattering source. The back scattered radiation creates a speckled pattern at the ground where the speckle size is $\approx \lambda/\theta$, where θ is the subtended angle of the illuminated region at the sodium layer. The speckle covers essentially an infinite region on the ground, but we consider only an $800\text{cm} \times 800\text{cm}$ region as shown in Fig. 111.4 — overfilling the 50 cm diameter receiving aperture by 16 times and effectively modeling an infinite width of speckle. The 50 cm diameter aperture cuts out a few speckles as shown in Fig. 111.5.

Input: `ex111.inp`

```
c## ex111
#
# Example 111: Reflection off the sodium level in the atmosphere
#
# Includes:
# 1) sodium layer scattering
# 2) atmospheric turbulence that broadens spot at sodium layer
#
```

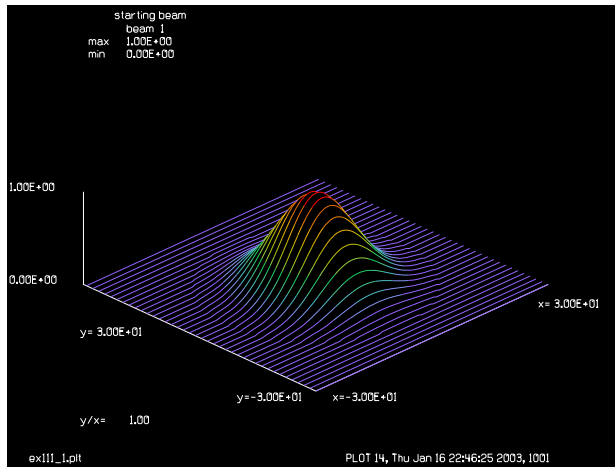


Fig. 111.2. Starting gaussian beam.

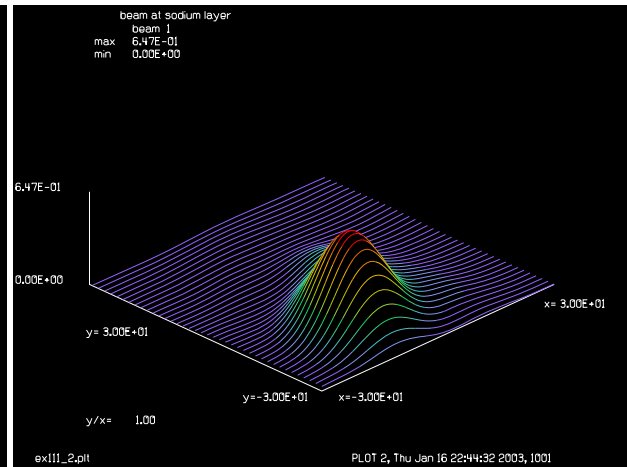


Fig. 111.3. Image on sodium layer showing the effects of atmospheric turbulence.

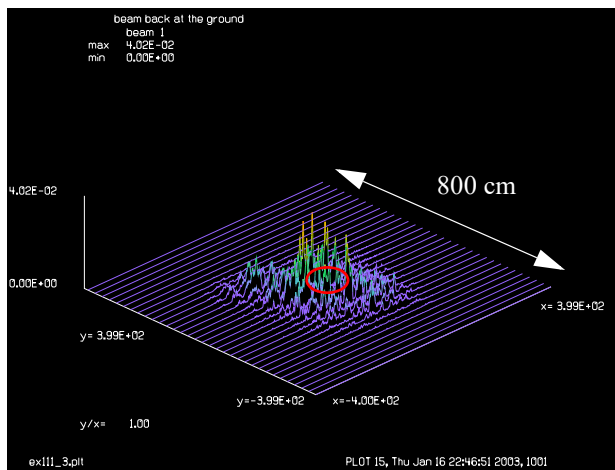


Fig. 111.4. Speckle pattern observed on the ground. Actually extends across an infinite plane. The finite sized scattering sources used in the numerical model limit the angular subtense to several times the width of the telescope diameter.

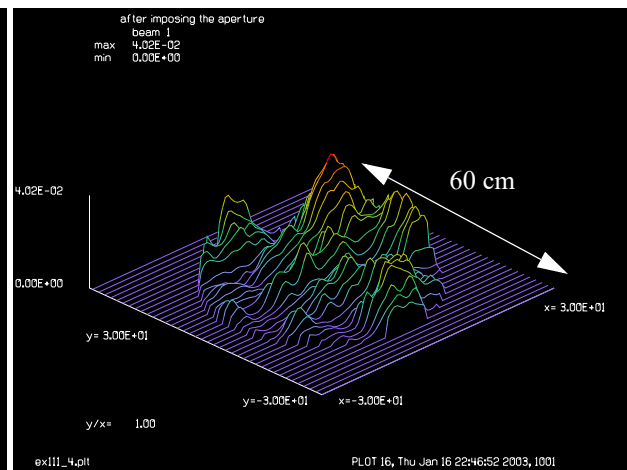


Fig. 111.5. Speckle pattern received by telescope aperture. 13.3X expanded scale from Fig. 111.4, showing the 50 cm aperture receiving aperture.

```
# Observe that there is fine structure in the returned
# speckle pattern due to larger spot at sodium layer
#
set/alias_stop/off
variab/dec/int RemainingIterations count
mem/set/b 8
array/s 1 1024
Lambda = .589e-4 # wavelength in cm
wavel/set 1 Lambda*1e4
Radius = 9e6 # radius of curvature is 2 km
Waves = .5/Radius/Lambda # waves of focus at 1 cm radius
units/field 1 400
macro/def step/o
    count = count + 1
```

Jump to: [Commands](#), [Theory](#)


```

gaus/c/c 1 1 15
clap/c/n 1 22.5
abr/foc 1 Waves rn=1           # lens defined as phase object
plot/w ex111_1.plt
title starting beam
plot/l 1 1 xrad=30
phase/kol 1 12                 # Kolmogorov, 10 cm seeing parameter
prop Radius                    # propagate to center of curvature
if [count==1] then
    plot/w ex111_2.plt
    title beam at sodium layer
    plot/l 1 1 xrad=30
endif
amplitude/abs 1                # take absolute value to remove
                                # all phase information from beam
                                # due to original beam
phase/random 1 1 6             # scatter at modest angles
prop -Radius
plot/w ex111_3.plt
title beam back at the ground
plot/l 1 1
plot/w ex111_4.plt
clap/c/c 1 22.5                # aperture of telescope
title after imposing the aperture
plot/l 1 1 xrad=30
variab/set RemainingIterations macro/remaining
macro/end
macro step/5

```


Ex112: Data reduction of interferograms by Fourier Transform Method

Summary

The Fourier Method is a convenient and fast method of reducing interferograms in digitized form. Three issues are considered in this report:

- The best accuracy may be achieved with proper size, location, and shape of the filtering aperture. An elliptical aperture located at the centroid of the side lobe works well.
- A self-consistency test consisting of simulating an interferogram with a known wavefront, serves to validate the method and to test the accuracy. An accuracy of better than $\lambda/100$ may be achieved.
- We may routinely check the accuracy of the method by numerically forming an interferogram from the recovered wavefront and calculating the correlation with the original interferogram. A correlation of 0.97 is found for the specific data used here as a sample—indicating high accuracy in recovering the wavefront.

Background

An interferogram is formed by a test beam containing information about the system of interest and a reference beam which is nominally aberration-free and of uniform intensity. The Fourier Method provides a convenient means of determining the phase from an intensity map of the interferogram as recorded digitally with a CCD camera. The method is not perfect but, provided there is a substantial angle between test and reference beams, errors due to the numerical method may be kept less than $\lambda/100$ for typical problems.

We can explain the procedure mathematically. Consider the test and reference beams to have complex amplitudes α_1 and α_2 respectively. The irradiance pattern formed by the two beams is

$$I = \alpha_1 \alpha_1^* + \alpha_2 \alpha_2^* + \alpha_1 \alpha_2^* + \alpha_1^* \alpha_2 = I_1 + I_2 + \alpha_1 \alpha_2^* + \alpha_1^* \alpha_2 \quad (112.1)$$

In reconstruction we illuminate the recorded irradiance with the reference beam α_2 to create

$$\alpha_2 I = \alpha_2 (I_1 + I_2) + \alpha_2 \alpha_2 \alpha_1^* + I_2 \alpha_1 \quad (112.2)$$

For an aberration-free reference beam $\alpha_2 \alpha_2 = I_2$ and

$$\alpha_2 I = \alpha_2 (I_1 + I_2) + I_2 \alpha_1^* + I_2 \alpha_1 \quad (112.3)$$

If the reference beam α_2 is aligned with the optical axis, then $\alpha_2 (I_1 + I_2)$ represents the 0 order and $I_2 \alpha_1^*$ and $I_2 \alpha_1$ are the -1 and +1 orders. These orders are most readily distinguished in the far-field. Using tildes to indicate the far-field distributions created by Fourier transforms we have

$$\tilde{\alpha}_2 ** \tilde{I}_1 = \tilde{\alpha}_2 ** (\tilde{I}_1 + \tilde{I}_2) + I_2 ** \tilde{\alpha}_1^* + I_2 ** \tilde{\alpha}_1 \quad (112.4)$$

where $**$ represents two dimensional convolution. For uniform reference beam I_2 , the Fourier transform \tilde{I}_2 is a delta function and apart from uniform scale factors

$$\tilde{I} = (\tilde{I}_1 + \tilde{I}_2) + \tilde{a}_1^* + \tilde{a}_1 \quad (112.5)$$

Provided the angle of separation, as measured in waves, is greater than the wave aberration of the test beam, the side lobes will be well separated from the 0 order (the central lobe of the far-field). We can isolate one of the side lobes by using a decentered aperture. If the initial test beam has an x-tilt of negative sign and we apply a decentered aperture we can separate out either a_1 or a_1^*

$$I \times \text{aperture}(\tilde{x} - \tilde{x}_0) \approx \tilde{a}_1 \quad (112.6)$$

$$I \times \text{aperture}(\tilde{x}_0 - \tilde{x}) \approx \tilde{a}_1^* \quad (112.7)$$

One can not tell from the interferogram alone which order represents the original beam and which is the conjugate side lobe. The phase errors of \tilde{a}_1^* are of opposite sign to that of \tilde{a}_1 , so it is quite important to know which side lobe to use. This uncertainty of sign of the wavefront is the case with all interferograms: we must have knowledge of the construction configuration to know which sign to use. Great care must be taken so as to avoid mistakes.

The extracted side lobe contains the phase tilt associated with the construction geometry. One may numerically remove the phase tilt by shifting the side lobe to be at the center. An inverse Fourier will then reconstruct the original test beam a_1 . Of course the aperture can not completely separate the side lobe from the 0 order. Some of a_1 is invariably clipped by the aperture causing Gibbs phenomena in reconstructed pupil distribution, and some of the central lobe radiation spreads as far as the side lobe and will be passed by the aperture.

Numerical methods

The numerical implementation is fairly straightforward. The major concern is determining the best location and size of the clipping aperture.

Calculation steps

The steps in the numerical solution are

1) Embed the interferogram into a larger array. Embed the 240 x 240 interferogram data into a 1024 x 1024 array. This gives a relatively high number of pixels across the width of the side lobes. An array of 512 x 512 could also have been used with coarser sampling across the side lobes, but the calculation time for 1024 x 1024 was not burdensome so I used this size.

2) Form the far-field distribution by forward Fourier transform of the interferogram.

3) Locate the center of the aperture. Proceed in three steps. First, Clipp the central lobe and the unwanted side lobe. This was done so that the peak of the desired side lobe could be found simply by scanning over the array. Located the coordinates of the peak of the side lobe. Apply a second aperture centered about the peak. This second aperture clipped away the stray light widely separated from the side lobe which would

otherwise bias the measurement of the side lobe centroid. The final step is to find the centroid of the light remaining after applying the two clipping apertures.

4) Shift the side lobe to put the centroid of the side lobe at the center of the coordinate system. This removes the tilt used in the construction of the interferogram.

5) Inverse Fourier transform to recover α_1 .

6) Check the quality of the recovered phase by forming a second interferogram and finding the correlation of the calculated interferogram with the measured interferogram. Typical data sets have yielded calculated-to-measured interferogram correlations of about 0.97.

The command file implements all necessary steps automatically for any digitized interferogram so that it is not necessary to make any manual corrections.

Checking by simulated input (self-consistency test)

The accuracy of the numerical method was checked by simulating the interferogram with a test beam having known aberration. 0.5 waves of astigmatism and 9 waves of tilt was used to create an interferogram that looked quite similar to the interferograms to the experimental data of interest. See Fig. 112.1. The data was transformed to have only 256 intensity levels: the same as the 8 bit accuracy of the test data. The Fourier transform was then applied to form the far-field. Fig. 112.2 shows the central lobe and two side lobes representing far-field of the test beam and its conjugate. Figure 112.2 has been expanded 25 times to show the side lobes more clearly. Note that there is an energy bridge that forms between the central lobe and the side lobes. An elliptical aperture with the long axis aligned with the long image produced by astigmatism. A narrow width of the aperture in the tilt direction was used to separate the side lobe from the light from the central lobe. The apertured side lobe is recentered to remove the tilt aberration, as shown in Fig. 112.3. The wavefront of the recovered beam is shown in Fig. 112.4. The residual aberration was calculated between the original beam and the recovered beam. The residual errors were less than $\lambda/100$ rms.

A typical example

Figure 112.6 shows an example of an actual interferogram. After data reduction using the Fourier method the original test beam is recovered. To check the accuracy an interferogram is calculated from the recovered beam and shown in Fig. 112.7. The correlation of the two interferograms is 0.974 indicating that the fit is quite good.

Input: ex112.inp

```
c## ex112
#
# Examples 112: Data reduction of interferograms by Fourier Transform
# Method
#
# This example illustrates the extraction of the
variab/dec/int Nord
mem/set/b 48
set/cpu 2
array/s 1 1024
nbeam 3 256 256 data
infile/intens ex112.dat/noheader/byte 2
```

Jump to: [Commands](#), [Theory](#)

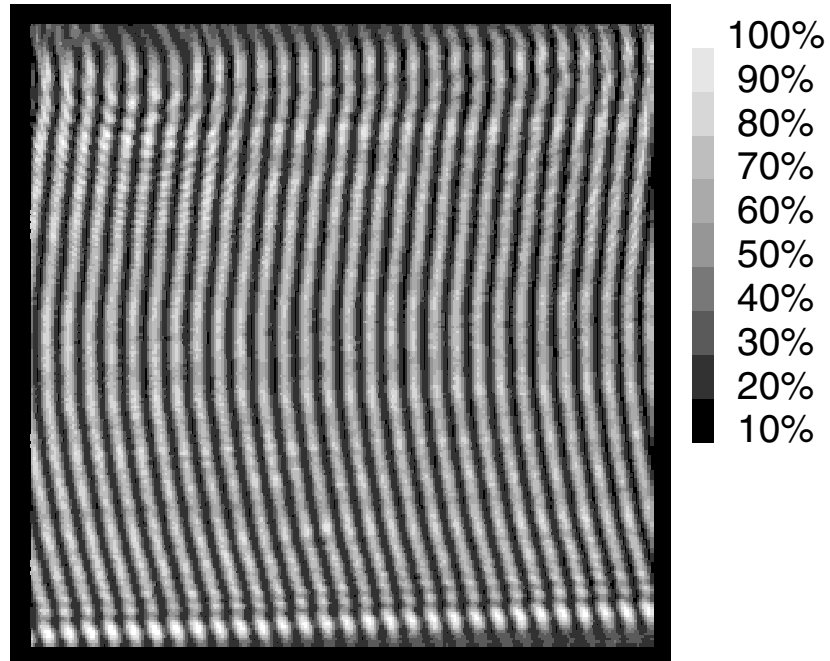


Fig. 112.1. Simulated interferogram with 0.5 waves of astigmatism.

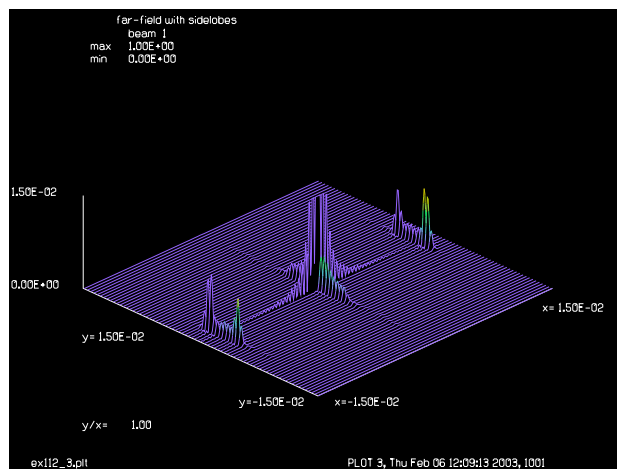


Fig. 112.2. Fourier transform of interferogram showing left and right side lobes. Intensity is multiplied by 25X to show side lobes bridge of energy between the lobes.

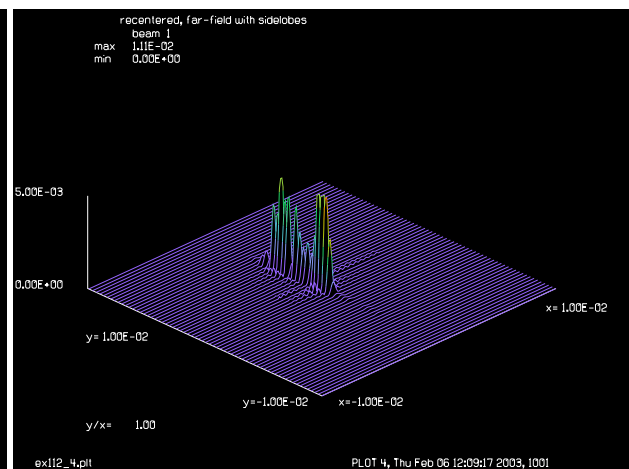


Fig. 112.3. The right side lobe is recentered after the central lobe and left side lobe are clipped. The bridge of energy to the central lobe has been clipped off.

```
transpose 2                # change to x-fringes
outfile/intens ex112a.dat/no/byte 2
plot/w ex112_1.plt
title original interferogram
plot/l 2
plot/w ex112_2.plt
set/den 256
plot/b/i/g 2
clear 1 0
copy/con 2 1
lens 1 100
```

Jump to: [Commands](#), [Theory](#)

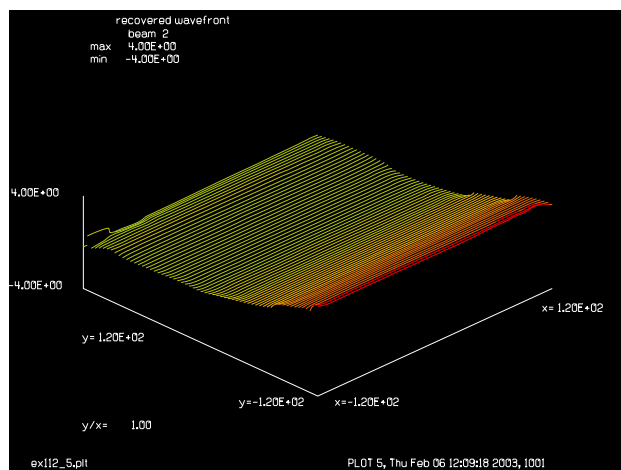


Fig. 112.4. Recovered wavefront matches simulated wavefront to better than 0.01 wave rms. No phase ripple is evident.

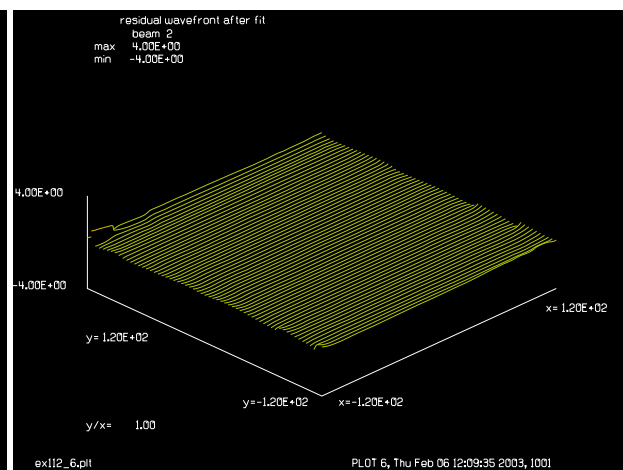


Fig. 112.5. Residual wavefront formed by difference of recovered and original wavefront.

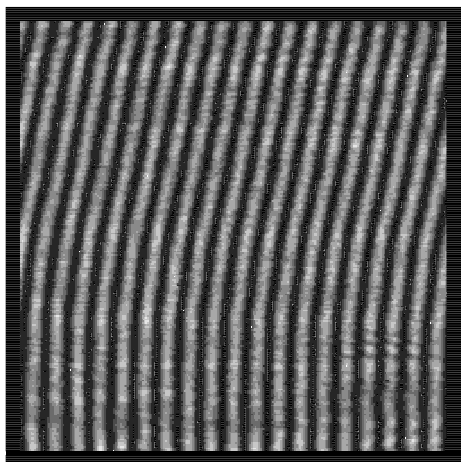


Fig. 112.6. Measured Interferogram. Data has been rotated 90 to put tilt in x-direction.

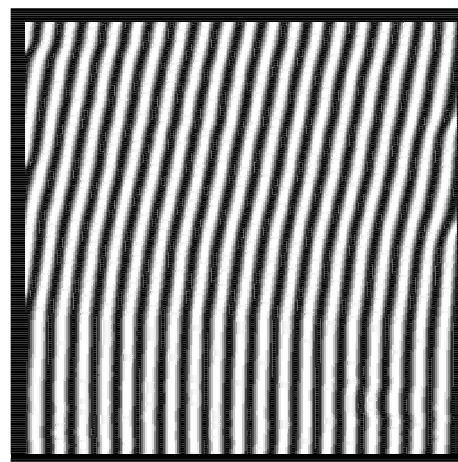


Fig. 112.7. Reconstruction of measured interferogram made from calculated wavefront. Correlation is 0.974 with the original data set shown to the left.

```
prop 100
peak/norm 1 1
plot/w ex112_3.plt
title far-field with sidelobes
plot/l max=.015 xrad=.015 ns=64
c
c first clip off center and left half so
c peak will be found correctly
c
clap/r/c 1 .5 1 .51
peak 1
c
c use coordinates of peak to locate aperture
c need new GLAD 4.5 to have find coordinates of peak
c
```

Jump to: [Commands](#), [Theory](#)

```

variab/set/par X fitxcen
variab/set/par Y fitycen
clap/c/c 1 .006 x=X y=Y # was .003
c
c now use fitgeo since field is already clipped we will get
c a reliable centroid
c fitgeo
c variab/set/par X fitxcen
c variab/set/par Y fitycen
shift/cart 1 x=-X y=-Y
plot/w ex112_4.plt
title recentered, far-field with sidelobes
plot/l max=.005 xrad=.01 ns=64
dist -100
copy/con 1 2 # copy back to small array
clap/s/c 2 120
plot/w ex112_5.plt
title recovered wavefront
Wmax=4.
plot/l/w 2 min=-Wmax max=Wmax ns=64 xrad=120
copy 2 3
transpose 2 # back to original orientation
write/d fitzern.dat/o
Nord=8
fitzern/list 2 nrad=Nord rn=120 rignore=200
write/screen
str 2
fitzern/remove 2 nrad=Nord rn=120 rignore=200
str 2
bell
pause 5
title residual wavefront after fit
transpose 2 # back to flipped form for final display
plot/w ex112_6.plt
plot/l/w 2 min=-Wmax max=Wmax ns=64 xrad=120
conjugate 2 # flip sign of residual phase
peak/norm 2 1
peak/norm 3 1
normalize 3 .001 # make beam 3 into a phase screen
mult/beam 2 3
c
c now add tilt back into beam by shifting
c
clear 1 0
copy/con 2 1
dist 100 1
shift/cart 1 x=X y=Y
dist -100 1
clear 2 0
copy/con 1 2
c
c reconstruct original interferogram
c
clear 3 1

```

Jump to: [Commands](#), [Theory](#)


```
peak/norm 2 1
add/coh/con 2 3
clap/s/c 2 120
peak/norm 2 1
plot/w ex112_7.plt
title recreated from polynomial fit
plot/b/i/g 2 max=.9
transpose 2
c outfile/intens ex112b.dat/noheader/byte 2
c system byte2ps xxx.dat > xxx.ps
```


Ex113: Optical limiting

Optical limiting is a nonlinear increase in absorption as a function of incident irradiance or fluence. consider the general case of

$$\frac{d}{dz}I(x, y) = \alpha I(x, y) + \beta I(x, y)^2 \quad (113.1)$$

$$\frac{d}{dz}W(x, y) = \gamma I(x, y) \quad (113.2)$$

Solution of the α term is simply Beer's Law loss. The β term, taken by itself, has the solution:

$$I(z) = \frac{I(0)}{1 - \beta z I(0)} \quad (113.3)$$

and may be considered as a second order Beer's Law. The above functional form is implemented in the routine `functions/inverse`. For $\alpha = -0.1$, $\beta = -0.1$, $\gamma = 1$; the first two terms reduce the peak and the third term increases the peak after propagation due to self-focusing.

Input: ex113.inp

```
c## ex113
#
# Example: Optical limiting, self-focusing as a function of I(x,y) and I(x,y)^2
#
alias Name ex113
Lambda = .5          # define wavelength, .5 micron
Zstep = .4           # specify step length
alpha = -.1          # Beer's Law loss
beta = -.1           # second order Beer's Law
gamma = 1.           # self-focusing effect
w0 = 2
array/s 1 256
nbeam 2 data
nbeam 3
wavelength/set 0 Lambda*1e4
units/s 0 0.1        # set pixel-to-pixel spacing
gaus/c 1 1 w0
copy 1 2
Coef1 = beta*Zstep
Coef2 = gamma*2.*pi/Lambda*Zstep
beer/set g0=alpha es=1e10
macro/def step/o
  copy 1 3
  beer/noprop 1 Zstep      # implements dI/dz = alpha*I
C peak intensity drops by Beer's Law
peak 1
pause
```

```
functions/inverse 1 Coef1      # implements  $dI/dz = \beta I^2$ 
C peak intensity drops by nonlinear Beer's Law
peak 1
pause
  int2phase/two 1 3 Coef2      # implement  $dP/dz = \gamma I$ 
  prop Zstep
C self-focusing increases the peak after propagation
  plot/w @Name_1.plt
  title intensity profile
  plot/x/i 1
  plot/w @Name_2.plt
  title wavefront
  plot/x/w 1
macro/end
macro step
plot/w @Name_3.plt
title Beam 1 is the output Beam 2 is the input
plot/x/i fi=1 la=2
```

Ex114: Various plot styles

This example illustrates various plot styles.

Input: ex114.inp

```

c## ex114
#
# Illustrate some of the graphic file features
#
# linear isometric with 13 colors
#
gaus 1 1 20
plot/watch ex114a_1.plt
title liso plot, 13 colors
plot/liso 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
#
# linear isometric with 6 colors
#
set/numberofcolors 6
plot/watch ex114a_2.plt
title liso plot, 6 colors
plot/liso 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
#
# linear isometric with 1 colors
#
set/numberofcolors 1
plot/watch ex114a_3.plt
title liso plot, single color determined by beam number
plot/liso 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
#
# linear isometric in monochrome
#
set/monochrome
plot/watch ex114a_4.plt
title liso plot, monochrome
plot/liso 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
set/color          # return to color plot
#
# bitmap, wireframe
#

```

```
plot/watch ex114a_5.plt
title bitmap wireframe
plot/bitmap/intensity/wireiso 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
#
# bitmap, paintiso, default 6 colors
#
plot/watch ex114a_6.plt
title bitmap paintiso
plot/bitmap/intensity/paintiso 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
#
# bitmap, paintiso, default 6 colors, grid
#
plot/watch ex114a_7.plt
title bitmap paintiso, 6 colors, grid
plot/bitmap/intensity/paintiso 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
#
# bitmap, paintiso, style 7
#
plot/watch ex114a_8.plt
title bitmap paintiso, 12 colors, grid
plot/bitmap/intensity/7 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
#
# bitmap, paintiso, style 8
#
plot/watch ex114a_9.plt
title bitmap paintiso, 12 colors, no grid
plot/bitmap/intensity/8 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
#
# bitmap, paintiso, style 9
#
plot/watch ex114a_10.plt
title bitmap paintiso, 12 colors, no grid
plot/bitmap/intensity/9 1
c plot/meta/wmf
c pause 2          # pause to let metafile complete
c plot/meta/cgm
```

Ex115: Pulse compression with grating rhombs

Temporal pulses may be stretched or compressed by the use of grating rhombs as indicated in Fig. 115.1 to take advantage of group velocity dispersion (GVD). The first rhomb adds angular divergence which results in rays for different optical frequencies striking the second rhomb after traveling different optical paths lengths (OPL), yielding the optical path difference functions (OPD). The second rhomb corrects the angular divergence, but the difference in optical path length remains.

We may treat a particular frequency component with propagation constant β ,

$$E(z, f) = E(f) e^{j(\beta(\Delta f) \cdot z)} \quad (115.1)$$

where $\beta(z)$ is the propagation vector and \hat{z} is the unit vectors for the z-axis. Let us allow for either dispersion of the propagation constant or angular divergence in terms of the dot product of the propagation constant and z-axis and use the frequency increment $\Delta f = f - f_0$ with respect to some center frequency f_0 (the carrier frequency).

$$\beta'(\Delta f)z \equiv \beta(\Delta f) \cdot z, \quad (115.2)$$

so that Eq. (115.2) represents the dispersive effect of both index of refraction and angle. This generalized scalar propagation constant may be written as a power series:

$$\beta'(\Delta f) = \sum_{i=0}^{\infty} a_i \Delta f^i \quad (115.3)$$

The zeroth term determines the phase velocity and the first order term determines the group velocity. The 2nd order term determines pulse compression or expansion. The third and higher order terms determine temporal pulse distortion. Let us define the pulse distortion distribution as $e_2(t)$,

$$\begin{aligned} e_2(t) &= \int_{-\infty}^{\infty} E(z, f) \Delta f = \int_{-\infty}^{\infty} E(\Delta f) \exp \left[jz \left(\sum_{i=2}^{\infty} a_i \Delta f^i \right) \right] \exp(-j2\pi t \Delta f) \Delta f \\ &= F^{-1} \left\{ E(\Delta f) \exp \left[jz \left(\sum_{i=2}^{\infty} a_i \Delta f^i \right) \right] \right\} \end{aligned} \quad (115.4)$$

The full solution is then,

$$e(t) = e^{j(\beta_0 z - 2\pi f_0 t)} e_2 \left(t - \frac{1}{2\pi} a_1 z \right) = e^{j2\pi f_0 \left(\frac{z}{v_\phi} - t \right)} e_2 \left(t - \frac{z}{v_g} \right). \quad (115.5)$$

We have taken advantage of the fact that a linear phase shift in frequency is equivalent to a linear shift in time:

$$g(t-t_0) = F^{-1}[\exp(j2\pi t_0 f)G(f)] \quad (115.6)$$

The phase velocity is $v_\phi = 2\pi f_0/\beta_0$ and the group velocity is $v_g = 2\pi/a_1$. In this example, a 30 picosecond pulse is used. For the grating rhomb arrangement the OPL values are sufficient to accommodate a compression of about a factor of 20. A quadratic phase factor is introduced into the temporal pulse, creating a linear change in frequency. This is described as “chirp”. Good sampling of the quadratic phase in temporal and frequency domains requires fine sampling. In this case 2048 sample points were used.

Potentially different path length and different shear of the beam would require separate diffraction propagation of each frequency, so we might need to propagate 2048 different arrays. However the difference in diffraction propagation length is a small fraction of the Rayleigh range so that the different lengths for the different frequencies may be neglected, although the phase due to optical path length must be included. The maximum shear of the extreme frequencies relative to the center frequency is about 0.5 cm, so the shear is also relatively small. If both the shear variation and diffraction propagation distances may be neglected, the problem is separable and the change in temporal waveform may be computed without consideration of the of diffraction propagation differences.

In this example, the dispersion is determined by a sample of eight frequency values. A power series polynomial to the fourth order is fit to the eight-point data. The zeroth a_0 and first order a_1 terms are deleted as they effect only the phase velocity and group velocity. The second order term a_2 causes ideal pulse compression for an initial quadratic phase on the temporal pulse. Ideal compression, i.e., no distortion of the temporal pulse shape other than magnification change, requires that the temporal pulse be of gaussian shape and that the dispersion be purely quadratic. The third order term a_3 , because it is not quadratic, limits the degree of compression that may be achieved without excessive distortion of the temporal waveform.

Input: ex115.inp

```
c## ex115
#
# Grating rhomb used for pulse compression
#
# In this analysis eight beams are traced through a grating
# rhomb to determine the dispersion. Then the dispersion
# data is used to calculate the pulse compression for a
# temporal pulse.
#
# The maximum shear of the beam is also calculated to be 0.5,
# but this is small relative to the diameter of 4 cm, so this
# one-dimensional analysis is appropriate
#
# A trace of eight frequency samples is used to determine the
# divergence. The zeroth and first order terms are dropped as
# they determine only the phase velocity and group velocity
#
set/negativeprop/on
mem/set/b 8          # set memory to more than enough
```

Jump to: [Commands](#), [Theory](#)

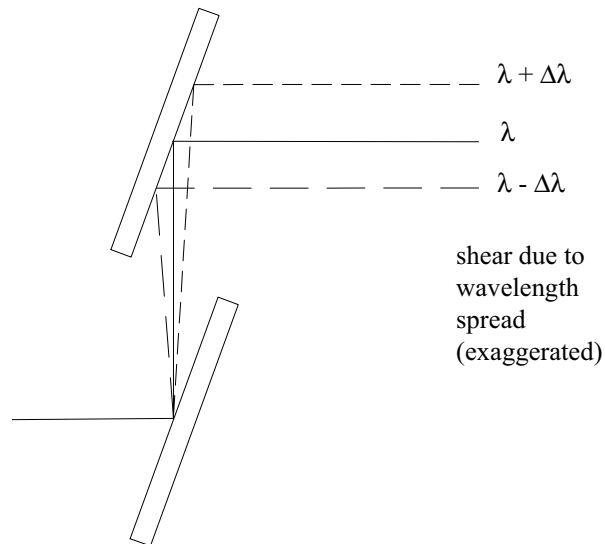


Fig. 115.1. A beam is incident on a grating rhomb. The first grating introduces angular dispersion which results in unequal optical path length for different optical frequencies as well as a small amount of shear. The second grating corrects the angular dispersion but leaves the optical path length variation and shear.

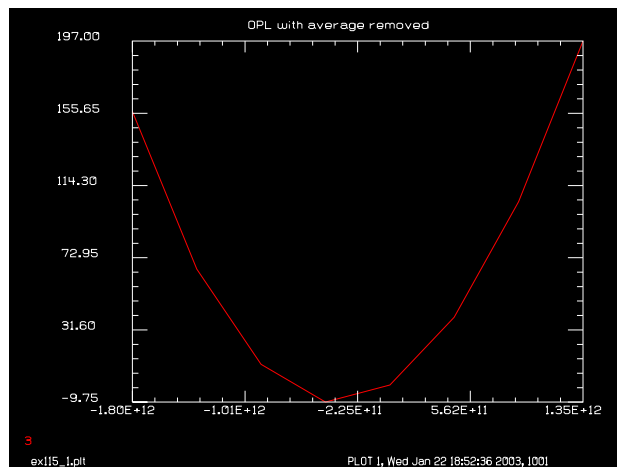


Fig. 115.2. OPD vs. frequency after approximate removal of constant and linear components.

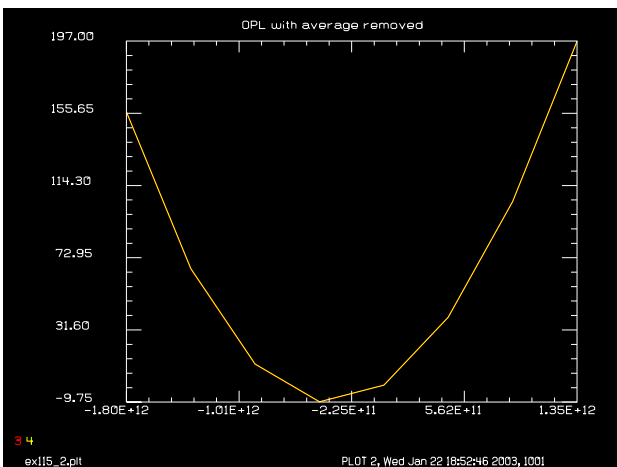


Fig. 115.3. Polynomial fit (yellow) matches OPL data (red) closely.

```
variab/dec/int Beam Order Pass Point Ntimes Nline
Lambda = 1e-4
Frequency = c/Lambda list    # calculate optical frequency

array/s 1 64
VertexDeg = 16.77 # Tip of grating rhomb
DiffractionDeg = 2*VertexDeg - 90 # make beam come out at 90 deg
OutDeg = VertexDeg-DiffractionDeg # output angle
#
# calculate period of grating
Order = 1
period = abs(Order*Lambda/(sin(pi*VertexDeg/180.)-sin(pi*OutDeg/180.)))
Field = 10          # set width of array
```

Jump to: [Commands](#), [Theory](#)

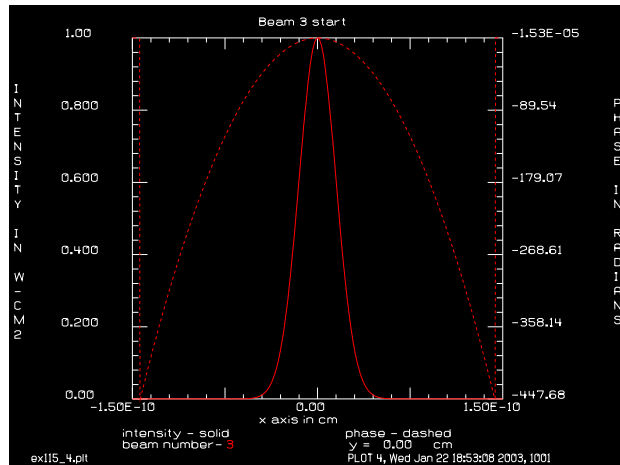


Fig. 115.4. Gaussian temporal pulse of $1/e^2$ radius of 30 picoseconds showing the frequency chirp as quadratic phase error.

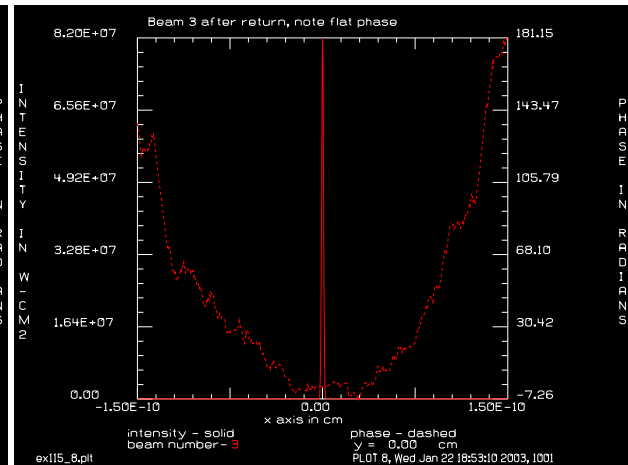


Fig. 115.5. Pulse compressed almost a factor of 20 by applying phase due to OPL dispersion in frequency domain.

```
Scale = 1e12          # scale factor used to keep some dimensions
                      # with in good range for numerical calculations
macro/def system/o
c
c Macro to calculate the OPL and shear
c
  Pass = Pass + 1
  zreff/se 1 0
  global/def 1 0 0 0 0 0 0
  dFrequency = dFrequency + ddFrequency # set the frequency increment
  Lambda = c/(Frequency+dFrequency) list # calculate the wavelength
  wavelength/set 1 Lambda*1e4           # set wavelength
  units/field 1 Field                    # field half-width of array
  gaus/c/c 1 1 2.                        # gaussian start
  vertex/locate/abs 0 0 0
  vertex/rotate/set VertexDeg 0 0        # rotate the grating 85 deg about x
  Order = 1
#
# implement reflection grating as a mirror plus a grating
#
  mirror/global/flat
  grating/global/blaze/phase 1 0 period Order
#
# set up next grating at distance of y=+10
#
  vertex/locate/abs 0 10 0
  vertex/rotate/set VertexDeg 0 0
  Order = -1 # make order of 2nd grating of opposite sign
  mirror/global/flat 1
  grating/global/blaze/phase 1 0 period Order
#
# propagate another 5 cm from grating
#
  vertex/locate/abs 0 10 5
```

Jump to: [Commands](#), [Theory](#)

```

    vertex/rotate/set 0 0 0
    prop/vertex
#
# calculate the OPL
#
    variab/set Opl2 1 opl list
    Opl2 = Opl2 - Opl0 list
#
# calculate the shear of the beam
#
    variable/set Gry 1 global/gry
    Gry = Gry - Gry0
    udata/set Pass dFrequency Opl2 Gry
macro/end
Ntimes = 8
#
# change ddFrequency to vary frequency range
#
ddFrequency = 4.5e11    # set up frequency increment in Hz
#
# run system macro once to calculate OPl and shear for
# center wavelength
#
Opl0 = 0.
Gry0 = 0.
dFrequency = -ddFrequency # starting frequency
macro system
Opl0 = Opl2
Gry0 = Gry
#
# offset frequency for Ntimes
#
dFrequency = -(Ntimes+2)/2*ddFrequency list
Pass = 0
macro system/Ntimes
macro/def calc/o
# macro to calculate approximate average
    Point = Point + 1
    variable/set f Point 1 udata
    variable/set dFrequency Point 0 udata
    sum1 = sum1 + f*dFrequency
    sum2 = sum2 + dFrequency^2
macro/end
#
# calculate approximate average
#
sum1 = 0.
sum2 = 0.
Point = 0.
macro calc/Ntimes
a = sum1/sum2
macro/def calc1/o
# macro to remove average
    Point = Point + 1

```

Jump to: [Commands](#), [Theory](#)

```

    variable/set f Point 1 udata
    variable/set Gry Point 2 udata
    variable/set dFrequency Point 0 udata
    f1 = f - a*dFrequency
    udata/set Point dFrequency f Gry f1
macro/end
#
# call macro to remove average
#
Point = 0
macro calc1/Ntimes
plot/watch ex115_1.plt
title OPL with average removed
plot/udata 3

macro/def calc2/o
# macro to fit power series to data
#
# get target data from udata, column 3
#
    variable/set t1 1 3 udata
    variable/set t2 2 3 udata
    variable/set t3 3 3 udata
    variable/set t4 4 3 udata
    variable/set t5 5 3 udata
    variable/set t6 6 3 udata
    variable/set t7 7 3 udata
    variable/set t8 8 3 udata
#
# z is the frequency increment, f is the test polynomial value
#
    z = -4*ddFrequency; f = a0 + a1*z + a2*z^2 + a3*z^3
    x1 = t1 - f
    udata/set 1 y04=f
    z = -3*ddFrequency; f = a0 + a1*z + a2*z^2 + a3*z^3
    x2 = t2 - f
    udata/set 2 y04=f
    z = -2*ddFrequency; f = a0 + a1*z + a2*z^2 + a3*z^3
    x3 = t3 - f
    udata/set 3 y04=f
    z = -ddFrequency; f = a0 + a1*z + a2*z^2 + a3*z^3
    x4 = t4 - f
    udata/set 4 y04=f
    z = 0; f = a0 + a1*z + a2*z^2 + a3*z^3
    x5 = t5 - f
    udata/set 5 y04=f
    z = ddFrequency; f = a0 + a1*z + a2*z^2 + a3*z^3
    x6 = t6 - f
    udata/set 6 y04=f
    z = 2*ddFrequency; f = a0 + a1*z + a2*z^2 + a3*z^3
    x7 = t7 - f
    udata/set 7 y04=f
    z = 3*ddFrequency; f = a0 + a1*z + a2*z^2 + a3*z^3
    x8 = t8 - f

```

Jump to: [Commands](#), [Theory](#)

```

    udata/set 8 y04=f
macro/end
#
# Solve for power series fit
#
# initialize variables
a0=0
a1=0
a2=0
a3=0
#
# set up variable table
opt/var/add a0
opt/var/add a1
opt/var/add a2
opt/var/add a3
#
# set up target table
opt/tar/add x1
opt/tar/add x2
opt/tar/add x3
opt/tar/add x4
opt/tar/add x5
opt/tar/add x6
opt/tar/add x7
opt/tar/add x8
#
# scale frequency increment to avoid overflow
ddFrequency = ddFrequency/Scale # scale frequency
opt/name calc2
opt/run 15      # run optimization

plot/watch ex115_2.plt
plot/udata/set y03 y04
plot/udata/seq
#
# Set up large N x 1 array to represent temporal pulse
#

Nline = 2048
nbeam 2 Nline 1 data      # make a big N x 1 array
#
# calculate frequency increment for N x 1 array
#
ddFrequency = ddFrequency*8/Nline
units/s 2 ddFrequency*Scale

macro/def calc3/o
# macro to put OPL into Beam 2 as real data
# omit zeroth and first order terms as these just determine
# phase velocity and group velocity
    Point = Point + 1
    z = (Point-Nline/2-1)*ddFrequency; f = a2*z^2 + 0*z^3
    point/set/ij/sr 2 Point 1 f

```

Jump to: [Commands](#), [Theory](#)

```
macro/end
# put OPL into real word of Beam 2 with only 2nd and 3rd order terms
Point=0
write/screen/off
macro calc3/Nline
write/screen/on
plot/w ex115_3.plt
title Opl transferred to Beam 2
plot/x/r 2

# restore original frequency increment
ddFrequency = ddFrequency*Scale

nbeam 4 Nline 1 data
#
# create a gaussian temporal pulse in Beam 3
#
dTime = 1/ddFrequency/Nline
units/s 3 dTime*Scale
Width = 30          # radius of temporal pulse is 30 picoseconds
gaus/c 3 1 Width
#
# Add chirp that was found by trial and error to cancel
# quadratic phase generated by divergence
#
Waves = 3.111 # set chirp, tune this value to get best compression
abr/focus 3 Waves rn=Width

# set up proper units
units/s 3 dTime
Width = Width/Scale
plot/w ex115_4.plt
title Beam 3 start
plot/x 3 le=-5*Width ri=5*Width
#
# implement forward FFT
#
convolve/front 3
units/s 3 ddFrequency      # explicitly set frequency units
plot/w ex115_5.plt
title Beam 3 with chirp
plot/x 3
#
# tranfer real part of Beam 2 to waves for Beam 3
#
real2waves/two 3 2 1

#
# Note that wavefront is now flat
#
plot/w ex115_6.plt
title Beam 3 with divergence phase
plot/x 3
#
```

Jump to: [Commands](#), [Theory](#)

```

# Also put divergence phase into Bema 4 to check for aliased phase
#
units/beam 4 3
real2waves/two 4 2 1
plot/watch ex115_7.plt
title Beam 4 wavefront, check for aliased wavefront
plot/x/w 4
#
# Go back to temporal space with a reverse FFT
#
convolve/back 3
units/s 3 dTime
fitgeo 3          # find width of beam now
#
# we now have about the best compression
# although the return phase is not perfectly flat
plot/w ex115_8.plt
title Beam 3 after return, note flat phase
plot/x 3 le=-5*Width ri=5*Width
variable/set Width1 fitxomega
Compression = Width/Width1 list    # compute degree of compression

```


Ex116: CGH test plates for aspheric mirror testing: Burge method

Table. 116.1. Table of Ex116 examples

Ex116a: Scan over one free spectral range	5
Ex116b: Scan over full separation	9

This example considers a computer generated holographic (CGH) test plate used for testing convex aspheric mirrors. Traditionally convex aspheric mirrors were tested using a very large concave sphere, called a Hindle sphere. The configuration consisted of placing the Hindle sphere such that its center of curvature coincided with the center of curvature of the convex test mirror. Generally the Hindle sphere had to be many times the diameter of the test mirror leading to considerable expense.

Burge and Anderson [1-5] devised a method using a CGH in reflection that could replace the expensive Hindle sphere method. The essential part of the configuration is illustrated in Fig. 116.2. A point source and focusing refractive optics (not shown) are used to create a converging beam focusing to the point F. A CGH is placed on the last surface. Optical elements and surfaces upstream (to the left in the drawing) are common path elements and modest amounts of aberration in them does not degrade the measured performance of the test system.

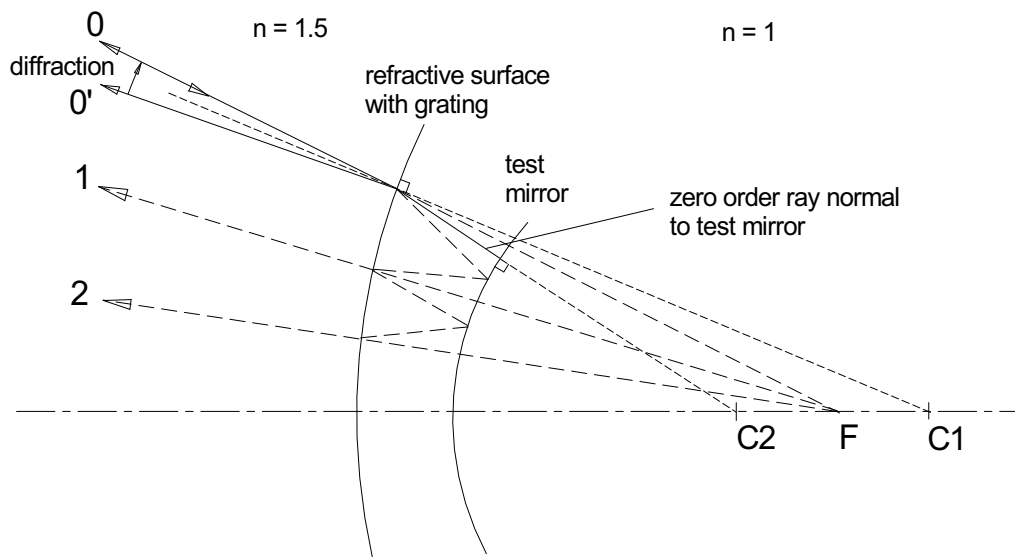


Fig. 116.1. An incident beam (0) is directed from left to right toward point F by optics which are not shown. We view the incident ray in the material of the last element in of the test optics. The transmitted beam passes through the last surface (centered at C1) which contains the CGH. The zero order ray arrives normal to the test mirror (centered at C2) and is retro-reflected to return diverging from point F.

Part of the incident light that reflects from the CGH (0') surface is diffracted into the +1 order such that it also diverges from point F to form the reference beam. The CGH is designed to add errors that compensate the null optics to create a null interference condition when the test optic is made perfectly. Spurious rays arise from the CGH in transmission. These rays will be reflected one or more times from the test mirror and may be further diffracted by the CGH in reflection and transmission. While many spurious rays can be removed by a spatial filter that is placed at an image of F, those spurious rays such as (1) and (2) that diverge from F can not be removed by a simple spatial filter conjugate to point F. Many such combinations of reflection and diffraction that result in spurious rays diverging from F are possible, but these rays are shown to have nominally the same OPD as a single bounce from the test mirror.

Light converging from left to right toward F is partially reflected and transmitted by the last surface containing the CGH. The CGH is a partially transmitting and reflecting zone plate which may be fabricated by as a chromed mask. The reflected beam contains both the zero order (specular reflection) and orders of diffraction. The CGH is designed so that one of the first orders, which may be considered +1, is diffracted to be nominally coincident with the incident light. This diffracted beam forms the reference surface. The CGH may be designed to have OPD errors that almost exactly compensate the ideal figure of the test element, so that a perfect null fringe would be observed for a perfect test optic.

The refractive surface is designed so that the transmitted beam is nominally converging to the center of the test optic. This is identified as the zero order ray and is normal to the surface of the test surface.

The zero order ray in the air space between the last refractive surface and the test optic is reflected from the test optic, picking up the OPD errors of the test piece, and returns along the path of the incident ray to exit diverging from the point F. As both the +1 diffracted order from reflection from the CGH and the zero order ray reflecting from the test optic, diverge from F an interference pattern with modest fringe count will be observed. A spatial filter placed at an image point of F to filter out unwanted orders of diffraction and multiple reflections.

A zone plate CGH creates many orders of diffraction for in both reflection and transmission. The unwanted orders created at the reflection from the CGH of the incident beam will be readily removed by a spatial filter placed conjugate to point F. As other refractive elements in the optics may be antireflection coated, we may disregard any ghost reflections of these refractive surfaces. However the CGH is generally fairly reflective on the order of 30%. If the test optic is reflection coated, multiple reflections and associated diffractions may occur in the air space between the CGH and the test optic. Fig. 116.2 a single bounce (1) and a double bounce (2) ray. The single bounce ray (1) is generated by transmission the +1 diffracted order, the -1 diffracted order diverges from F and will not, therefore, be removed at the spatial filter. The double bounce (2) is started by the +1 diffracted order in transmission but makes an extra bounce by reflection from the CGH. The -2 diffracted order for ray (2) will diverge from F and will pass through the spatial filter as well. Many other combinations of reflections and diffractions off the CGH are possible which can be coupled by some order of CGH diffraction to diverge from F. Additional reflections from the test optic and additional diffractions and/or higher order diffraction adds multiples of the test OPD, potentially degrading the test measurement.

Fig. 116.2 shows a simplified view formed by removing the base curvature of the test mirror and correcting the CGH surface appropriately. The source and return point is F' and the center of the revised CGH surface is C1. The aspheric OPD remains the same. It is clear that CGH surface acting in reflection to form the reference beam has positive optical power that is corrected by the -1 diffraction order, indicated by D^{-1} . The convention used here for the sign of diffraction order is that -1 indicates negative power but the figure introduced is the same as the test mirror. The CGH surface acting in reflection to form a cavity with the test optic has negative optical power which accumulates for with multiple reflections. In order for spurious rays to return to F' the net optical power for reflections and CGH diffraction must be zero.

Fig. 116.3 shows a simplified view with the negative diffraction order D^{-1} of the reference path indicated as a negative lens and a spurious reflection off the convex side of the CGH compensated by the D^{+1} diffraction order shown as a positive lens.

The reflecting surfaces of the CGH and the test mirror may be considered to form a cavity. The complete solution for all possible diffracted and reflected rays in the cavity that will be diverge from F may be written as

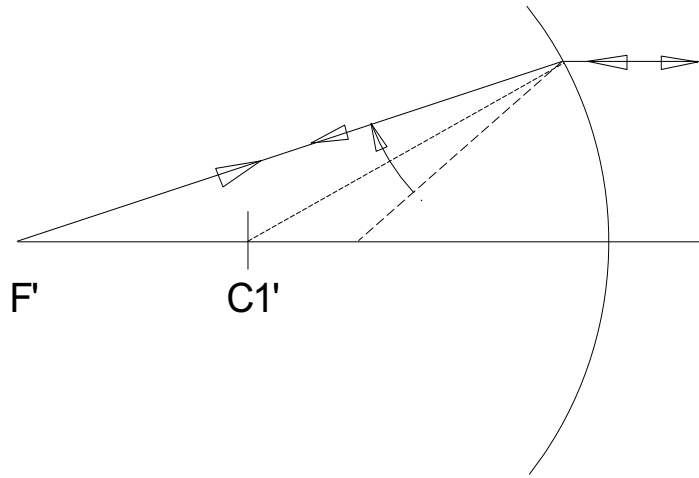


Fig. 116.2. Simplified model with the radius of the test mirror removed from both the test mirror and CGH surface. The equivalent focus point for reference and zero order return from the test optic is indicated by F' and the center of curvature of the modified mirror is C1'. It is clear the CGH surface in reflection for the reference beam has positive optical power and the CGH surface in reflection for the cavity region has negative optical power.

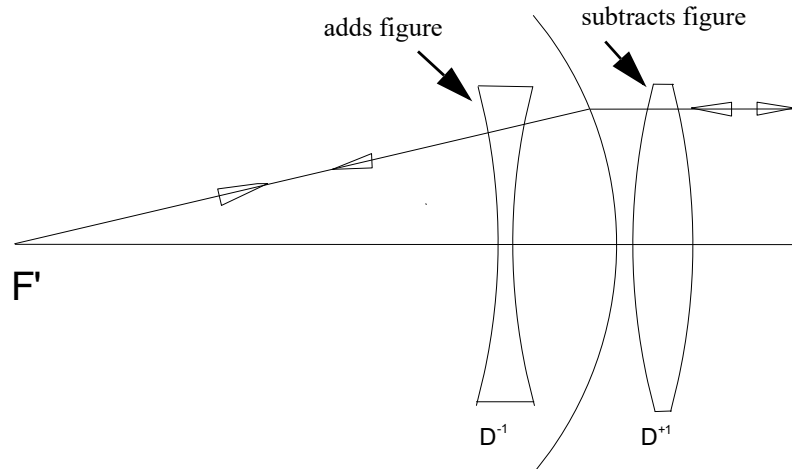


Fig. 116.3. Simplified model showing the reference beam with the -1 diffraction order of the CGH in the reference beam shown as a negative lens (D^{-1}) and the +1 diffraction order of the CGH (D^{+1}) in the cavity shown as a positive lens to compensate for the negative optical power of the convex side of the CGH. Multiple reflections from the CGH may be compensated by higher orders of negative diffraction to create spurious rays that return to F'.

$$T^2 D^L R_{test} (R_{CGH} R_{test})^M D^M D^{-L} \quad (116.1)$$

where

$$T \text{ is transmission through the CGH} \quad (116.2)$$

$$R_{CGH} \text{ is reflection from the CGH} \quad (116.2a)$$

Jump to: [Commands](#), [Theory](#)

R_{test} is reflection from the test mirror (116.2b)

D^L is diffraction from the CGH (116.2c)

L represents the order of diffraction not used to compensate for spurious reflections. May be due to L successive diffractions, L order diffraction, or a combination of the two. L may be any integer and must be matched by an equal number of diffraction of opposite sign. (116.2d)

M is the number of reflections from the cavity side of the CGH. Each such reflection results in an extra reflection from the test optic. M must be a non-negative integer. (116.2e)

Diffraction a ray L times results in the same optical power as diffracting at order L , so we do not need to distinguish these cases. The group $(R_{CGH}R_{test})^M$ represents an extra bounce off the test optic due to reflection from the convex side of the CGH (from Fig. 116.2). Reflection from the CHG is always paired with an extra reflection from the test optic. The number of extra reflections M must be matched by M positive diffraction orders, so that to satisfy the condition of zero net optical power we must have the group $(R_{CGH}R_{test})^M D^M$. Other diffraction orders may interact to create net zero optical power as indicated by the L terms in Eq. 116.1.

Because of the special design of the CGH the optical power of D and R_{CGH} have the same magnitude. For a ray to end up diverging from F, the net optical power of the ray exiting from the cavity must be zero. Eq. (116.1) yields all cases having zero optical power. Extra reflections on the test optic must be canceled by diffractions steps of negative order.

The OPD errors due to diffraction and reflection from the test optic add according to the sign of L and M in Eq. (116.1), so that all spurious additions of OPD must cancel leaving the OPD of the focused spurious rays to be same as the zero order ray. We can express this cancellation as

$$T^2 D^L R_{test} (R_{CGH} R_{test})^M D^M D^{-L} = |T|^2 R_{test} [|D|^{2|L|+M} |R_{CGH}|^M |R_{test}|^M] \quad (116.3)$$

where quantities in absolute values are just efficiency values and we have taken advantage of the relationship between a positive diffraction order combined with the conjugate surface figure errors of the CGH and reflection from the test mirror:

$$D = \frac{|D|}{|R_{test}|} R_{test}^* \text{ and } D R_{test} = |D| |R_{test}| \quad (116.4)$$

indicating that the OPD errors are, to first order, canceled for spurious rays.

$$\text{Reverence beam} \quad \frac{|D|}{|R_{test}|} |R_{CGH}| R_{test} \quad (116.5a)$$

$$\text{zero order test beam} \quad |T|^2 R_{test} \quad (116.5b)$$

Jump to: [Commands](#), [Theory](#)

spurious beams with zero net optical power $|T|^2 R_{test} [|D|^{2|L|} |R_{CGH}|^M |R_{test}|^M]$ (116.5c)

In Eq. 116.5c the spurious contribution is enclosed in brackets.

For best fringe contrast

$$\frac{|D|}{|R_{test}|} |R_{CGH}| \approx |T|^2 \quad (116.6)$$

and the coherent addition of the reference beam to the zero order test beam results in addition of identical beams creating a null fringe.

Ex116a: Scan over one free spectral range

Phase errors arise from both the effects of the standing wave and spurious diffraction orders. We can separate these effects. The standing wave goes through one cycle of change in the relatively short distance of one FSR. The spurious diffraction effects change slowly over the separation distance and are effectively zero when the separation distance is zero.

Ex118a illustrates a scan over on free spectral range (FSR) for a cavity at zero separation and 0.4 reflectivity for the CGH surface and 0.5 reflectivity for the test mirror. In the absence of cavity mode effects, we would expect the phase to vary linearly from 0 to 2π over one FSR and the intensity to be exactly constant. Any departure from liner phase over the FSR represents a systematic error Fig. 116.4 shows the intensity over one FSR. The data reduction algorithm would need to be insensitive to (or have correction for) the intensity variation to avoid having the intensity variation induce an error in measured phase. In the GLAD analysis we measure phase directly from the complex amplitude so the numerical analysis here does not suffer from variation of the intensity over the FSR. Fig. 116.5 shows the phase over one FSR. The slight curvature represents a noticeable phase error. In Fig. 116.6 the linear component that represents ideal behavior is extracted, showing only the phase error. This phase error should be corrected to achieve best results.

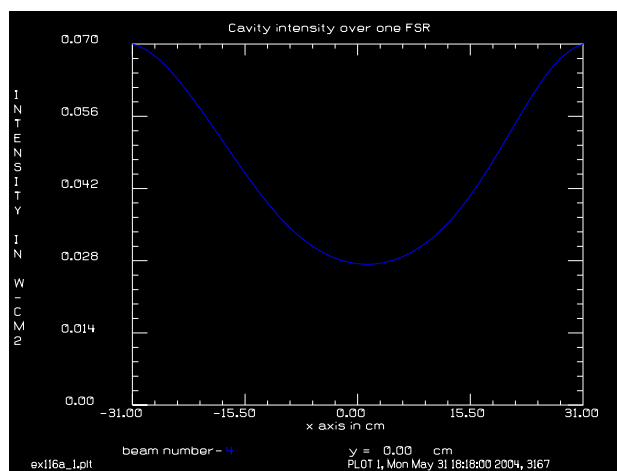


Fig. 116.4. Intensity over one FSR. The variation of intensity may cause problems in the phase extraction algorithm.

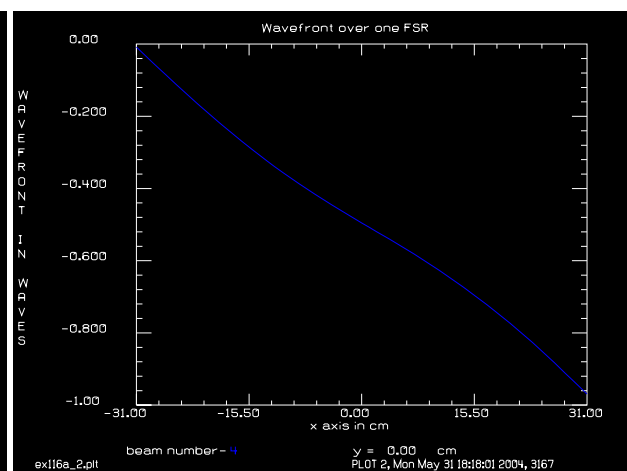


Fig. 116.5. Phase variation over one FSR. Any departure from linear phase variation represents a systematic phase error.

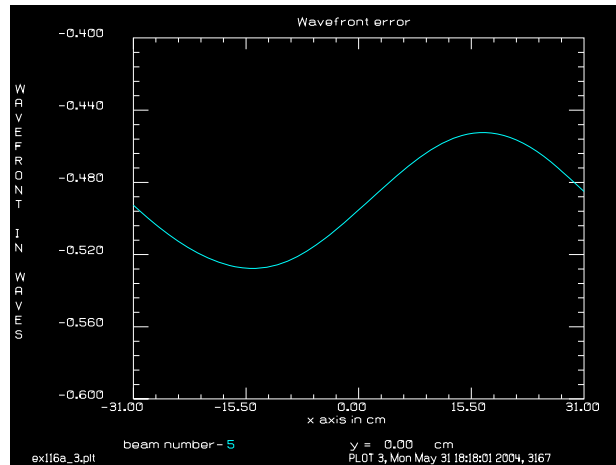


Fig. 116.6. Phase error over one FSR found by extracting the linear phase. This error is due to the standing wave and requires compensation to achieve best results.

Input: ex116a.inp

```

c## ex116a
#
# Example of phase of reflection from etalon vs. length.
#
# T = intensity transmission
# R = intensity reflection

alias name ex116a
set/highlight/off
mem/set/b 8
variable/dec/int count Pass
variable/dec/real dL Waves
NlineX=32768                                # set size of 1D propagating array
NlineY=1
NcX = NlineX/2 + 1
NcY = NlineY/2 + 1
array/s 1 NlineX NlineY data                # incident array
nbeam 2 beam                                # cavity mode
nbeam 3 beam                                # utility array for various purposes
nbeam 4 64 1 data                            # array to record output information
nbeam 5 64 1 data                            # modification of output
nbeam 7 NlineX NlineY data                  # array for aperture

clear 4 0
DiffAngleDeg = 2                            # diffraction angle
MirrorAngleDeg = DiffAngleDeg/2              # mirror angle, not used
Lambda = .6328e-4                           # wavelength
DiffAngleRad = pi*DiffAngleDeg/180           # diffraction angle in radians
Period = Lambda/DiffAngleRad                  # grating period
QuarterTalbot = Period^2/2/Lambda list       # quarter Talbot cycle, for reference
wavelength 0 Lambda*1e4                      # set wavelength
Field = .80
units/field 1 Field

```

Jump to: [Commands](#), [Theory](#)

```

units/field 2 Field
units/field 3 Field
units/field 6 Field
units/field 7 Field
clap/c 7 .6*Field          # set up aperture function
convolve/gaus 7 .1

grating/cosine/absorption 6 Period  # make grating into mask

Focus = 4    # waves at 1 cm radius
L = .21      # nominal cavity length
c
c  Tweak the length for peak response at Lambda0
c
wavelength 1 Lambda*1e4
wavelength 2 Lambda*1e4 1
wavelength 3 Lambda*1e4 1
Nwaves = 2*L/Lambda list
Nwaves = floor(Nwaves)
L0 = Nwaves*Lambda/2 list      # tweak length to start
                                # on an exact integer number of wavelengths
c
c Rest of calculations
c
R = .4          # set reflectivity, affects Q of cavity
T = 1 - R      # transmission
R_back = .5    # reflectivity of test mirror
Npass = 64
dL0 = Lambda/Npass/2      # incremental length for FSR
variables/monitor/add Waves
variables/monitor/add dL
macro/def etalon/o
    count = count + 1 list      # count number of passes
    clear 1 1
    zreff 1 0
    zreff 2 0
    units/field 1 Field
    units/field 2 Field
    mult/beam 1 7              # impose aperture
    copy 1 3                   # reset to initial field
    mult 3 T                    # multiply by transmission
    mult/beam 3 6              # add grating effect
    mult 2 R                    # multiply beam 2 by reflectivity

    abr/tilt/angle 2 DiffAngleRad az=90
    mult/beam 2 6              # add grating effect
    add/coh/con 2 3            # add incident field to cavity field
    prop L1 2                  # propagate length of etalon L
    mirror/flat 2              # flat mirror
    mult 2 R_back              # reflectivity of back mirror
    prop L1 2                  # propagate reverse pass
    mirror/flat 2              # second mirror
    phase/piston 2 Waves*360
macro/end

```

Jump to: [Commands](#), [Theory](#)

```

macro/def scan/o
c  write/off
  Pass = Pass + 1
  clear 2 0 # cavity beam set to 0
  zreff 0 0
  global/def
  count = 0
  macro etalon/10 # run cavity to convergence
  copy 2 3
  mult 3 T # multiply beam 2 transmission
  mult/beam 3 6 # add grating effect
c ----- this section will be used for actual interference pattern
c  mult 1 R # multiply beam 1 by reflectivity
c  mult/beam 1 6 # add grating effect
c  abr/tilt/angle 1 -DiffAngleRad az=90
c  phase/piston 1 180 # phase change on reflection for beam 1
c  add/coh/con 1 3 # light returned from cavity
c ----- end of section will be used for actual interference pattern

c spatial filter
  lens/x 3 100
  prop 100 3
  split/cir/inner 3 1 10
  prop -100 3
c end of spatial filter
  point/list/ij 3 NcX NcY
  variab/set CavityIntensity 3 peak
  variable/set Real 3 point/sr
  variable/set Imaginary 3 point/si
  point/set/ij 4 Pass 1 Real Imaginary
  udata/set Pass Pass Real Imaginary CavityIntensity
  dL = dL0*Pass
  Waves = 2*dL/Lambda list
  L1 = L0 + dL
  write/on
macro/end
Waves = 0
Pass = 0
L1 = L0
macro scan/Npass # scan over FSR
udata
plot/w @name_1.plt
title Cavity intensity over one FSR
plot/x/i 4
plot/w @name_2.plt
title Wavefront over one FSR
plot/x/w 4 pmin=-1 pmax=0
copy 4 5
plot/w @name_3.plt
title Wavefront error
abr/tilt 5 .5 az=90 rn=32
plot/x/w 5 pmin=-.6 pmax=-.4

```


Ex116b: Scan over full separation

Ex118b illustrates a scan over separation distance from 0 to 2.1 mm taken at incremental distance of one wavelength constituting two FSR's of roundtrip distance change per step. Since we are taking data at integer values of FSR, the standing wave effects are identical for each data point and do not effect the data. Instead we see the beating of the diffraction orders consistent with the Talbot distance. Fig. 116.7 shows the intensity over the 2.1 mm range. The approximately 2.5 cycles of intensity oscillation are consistent with 2.5 cycles of Talbot imaging. Both the intensity and phase variation due to spurious effects are relatively slowly varying, however they will show variation across the aperture according to the slope. A curved surface would effective cut across the phase vs. separation distance curve of Fig. 116.8 according to the relative separation at specific points in the aperture. The rapidly varying FSR phase modulation would be superimposed on the slowly varying spurious diffraction effects, such that both would contribute to phase error in the measurement if not compensated in some way.

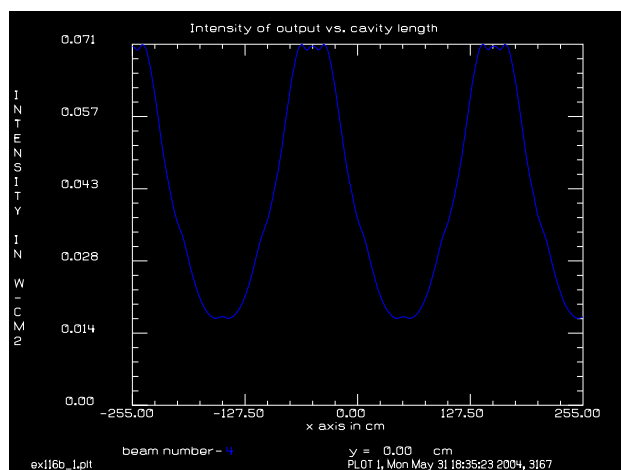


Fig. 116.7. Fluctuation of intensity observed at integer values of FSR the 2.1 mm distance in 512 steps The oscillation is consistent with the Talbot imaging distance.

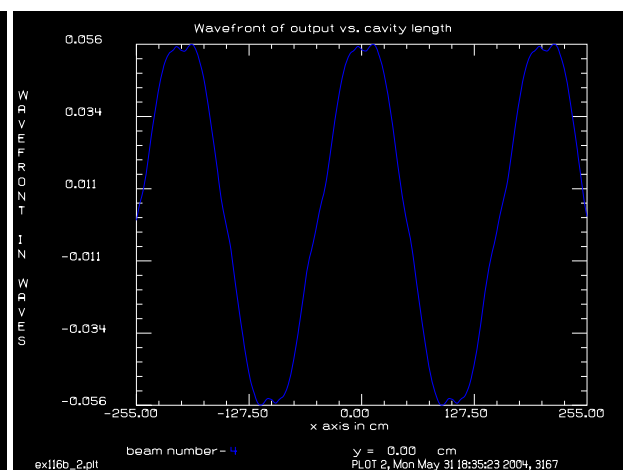


Fig. 116.8. Phase error due to spurious diffraction over the 2.1 mm distance. The magnitude of the error varies with the reflectivity of CGH and test mirror.

Input: `ex116b.inp`

```
c## ex116b
#
# Example of phase of reflection from etalon vs. length.
#
# T = intensity transmission
# R = intensity reflection

alias name ex116b
set/highlight/off
mem/set/b 8
variable/dec/int count Pass
variable/dec/real dL Waves
NlineX=32768                                # size of 1D propagation arrays
NlineY=1                                     # center points
NcX = NlineX/2 + 1
```

Jump to: [Commands](#), [Theory](#)

```

NcY = NlineY/2 + 1
Npass = 512                                # number of passes
array/s 1 NlineX NlineY data               # input beam
nbeam 2 beam                               # cavity mode
nbeam 3 beam                               # utility array for various tasks
nbeam 4 Npass 1 data                       # store data for passes
nbeam 5 Npass 1 data                       # process data for passes
nbeam 7 NlineX NlineY data                 # mask array to replace aperture

clear 4 0
DiffAngleDeg = 4                           # diffraction angle in degrees
MirrorAngleDeg = DiffAngleDeg/2            # mirror angle in degrees, not used
Lambda = .6328e-4                          # wavelength
DiffAngleRad = pi*DiffAngleDeg/180         # diffraction angle in radians
Period = Lambda/DiffAngleRad               # grating period
QuarterTalbot = Period^2/2/Lambda list     # quarter Talbot cycle for reference
wavelength 0 Lambda*1e4                    # set wavelength

R = .4                                     # reflectivity of grating
T = 1 - R                                 # transmission of grating
R_back = .5                               # reflection of test mirror
Field = .80
units/field 1 Field                        # set units
units/field 2 Field
units/field 3 Field
units/field 6 Field
units/field 7 Field
clap/c 7 .6*Field                          # make aperture function
convolve/gaus 7 .1

grating/cosine/absorption 6 Period          # make grating into mask

c
c Tweak the length for peak response at Lambda0
c
wavelength 1 Lambda*1e4 1
wavelength 2 Lambda*1e4 1
wavelength 3 Lambda*1e4 1

L0 = 0                                     # starting cavity length
dL0 = Lambda                               # increment cavity length by 1 wavelength each pass
macro/def etalon/o
c perform one round trip through etalon
  count = count + 1 list                   # count number of passes
  clear 1 1
  zreff 1 0
  zreff 2 0
  units/field 1 Field
  units/field 2 Field
  mult/beam 1 7                             # apply aperture function
  copy 1 3                                  # copy input to make transmitted beam
  mult 3 T                                  # multiply by transmission
  mult/beam 3 6                             # add grating effect
  mult 2 R                                  # multiply beam 2 by reflectivity

```

Jump to: [Commands](#), [Theory](#)

```

abr/tilt/angle 2 DiffAngleRad az=90 # apply mirror tilt to cavity mode
mult/beam 2 6 # add grating effect
add/coh/con 2 3 # add incident field to cavity field
prop L1 2 # propagate length of etalon L1
mirror/flat 2 # flat mirror
mult 2 R_back # reflectivity of back mirror
prop L1 2 # propagate reverse pass
mirror/flat 2 # second mirror
macro/end
macro/def scan/o
c scan over etalon Npass times
  write/off
  Pass = Pass + 1
  clear 2 0 # cavity beam set to 0
  zreff 0 0
  global/def
  count = 0
  macro etalon/10 # run etalon to convergence
c process output beam
  copy 2 3
  mult 3 T # multiply beam 2 transmission
  mult/beam 3 6 # add grating effect
c ----- use this section for true interferometer modeling
c   mult 1 R # multiply beam 1 by reflectivity
c   mult/beam 1 6 # add grating effect
c   abr/tilt/angle 1 -DiffAngleRad az=90
c   phase/piston 1 180 # phase change on reflection for beam 1
c   add/coh/con 1 3 # light returned from cavity
c -----

c Go to far-field to apply spatial filter so phase can be measured
  lens/x 3 100
  prop 100 3
  split/cir/inner 3 1 10
  prop -100 3
c End of spatial filter

  point/list/ij 3 NcX NcY # find center point values
  variab/set CavityIntensity 3 peak # find peak value
  variable/set Real 3 point/sr
  variable/set Imaginary 3 point/si
  point/set/ij 4 Pass 1 Real Imaginary
c store real imaginary and intensity
  udata/set Pass L1 Real Imaginary CavityIntensity
  dL = dL0*Pass
  L1 = L0 + dL
  write/on
macro/end
Waves = 0
Pass = 0
L1 = 0
macro scan/Npass # scan etalon Npass times
plot/w @name_1.plt

```

Jump to: [Commands](#), [Theory](#)

```
title Intensity of output vs. cavity length
plot/x/i 4
plot/w @name_2.plt
title Wavefront of output vs. cavity length
plot/x/w 4
```

References

1. J. H. Burge and D. S. Anderson, "System and method for interferometric measurement of aspheric surfaces utilizing test plate provided with computer generated hologram," U.S. Patent 5,737,070.
2. J. H. Burge and D. S. Anderson, "Full-aperture interferometric test of convex secondary mirrors using holographic test plates," Proc. SPIE **2199**, (1994).
3. J. H. Burge, "Measurement of large convex aspheres," Proc. SPIE **2871**, (1996).
4. B. K. Smith, J. H. Burge, and H. M. Martin, "Fabrication of large secondary mirrors for astronomical telescopes," Proc. SPIE **3134**, (1997).
5. J. H. Burge, "Applications of computer-generated holograms for interferometric measurement of large aspheric optics," Proc. SPIE **2576**, (1994).
6. J. H. Burge, "Fizeau interferometry for large convex surfaces," Proc. SPIE **2536**, (1995).

Ex117: Transverse pumping with an array of laser diodes

Table. 117.1. Table of Ex117 examples

Ex117a: Side pumping with geometric expansion.	1
Ex117b: Side pumping with slab/pump	5
Ex117c: Side pumping with slab/pump and three rotations	6
Ex117d: Resonator with complex side pumping	8

This example illustrates a model of laser diode side-pumping. Non-uniform illumination will result in decreased efficiency and will imprint intensity modulation into the beam which may result in unwanted diffraction effects. It is assumed that an array of fourteen laser diodes illuminates a gain region from the side. This may be modeled in terms of (x,y) gain sheets. A bank of laser diodes, elongated in the z-direction, may be modeled with a single (x,y) gain sheet may be modeled by using a single (x,y) gain sheet multiple times. In Ex117a the expansion of the diode beams is considered to be a simple linear expansion at a specified divergence. In Ex117b the command `slab/pump` is used to model side-pumping with an expanding gaussian beam.

Ex117a: Side pumping with geometric expansion

Fig. 117.1 shows as the beam profile for one laser diode as it propagates through the medium from left to right with the width increasing and the peak intensity decreasing. As the diode beam is well into the far-field when it enters the gain region, the expansion may be treated simply as a geometrical expansion. The beam is attenuated a loss per cm of `alpha` corresponding to the gain medium absorption. Fig. 117.2 shows the diode beam propagation with a contour plot.

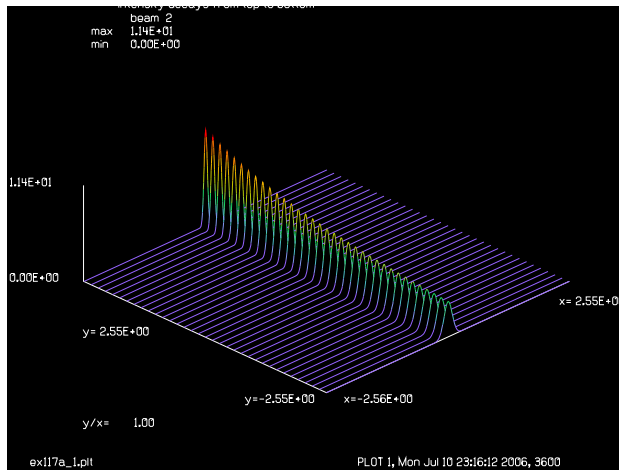


Fig. 117.1. Beam intensity profile of a single diode as it passes through the gain region. The intensity is high at the diode and the width increases and intensity decreases as it crosses the gain region.

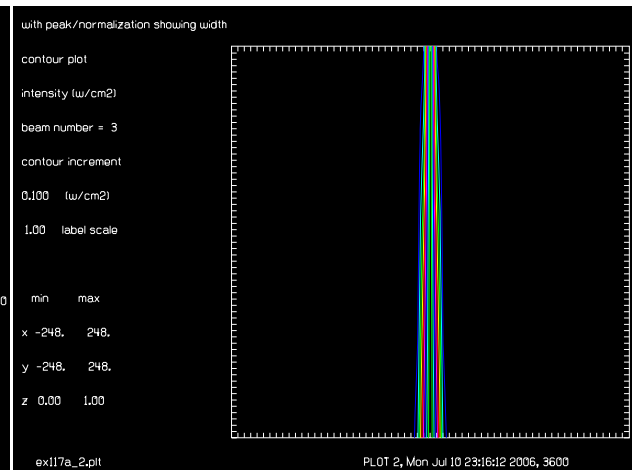


Fig. 117.2. Contour plot of single diode propagating from left to right. The beam expands, the width increases, and the intensity falls because of the beam expansion and because of absorption by the medium.

As the laser diodes are mutually incoherent, the bank of fourteen laser diodes may be modeled simply by summing the single diode beam profile multiple times with shifting, as shown in Fig. 117.3. This case shows relative low divergence diodes to give a somewhat exaggerated case to better illustrate the modeling

features. Similarly, the illumination due to a bank of laser diode arrays on the right side may be modeled simply by creating a flipped copy of the left-to-right distribution, as illustrated in Fig. 117.4.

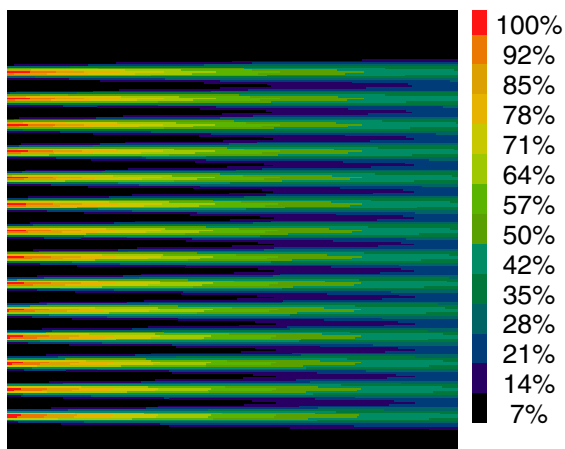


Fig. 117.3. The beam profiles of fourteen laser diodes propagating from left to right.

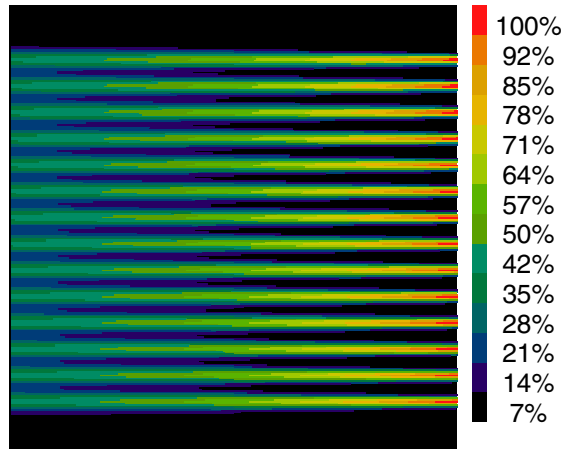


Fig. 117.4. Phase variation over one FSR. Any departure from linear phase variation represents a systematic phase error. The right-to-left distribution is shifted upward by one half the diode spacing for better uniformity.

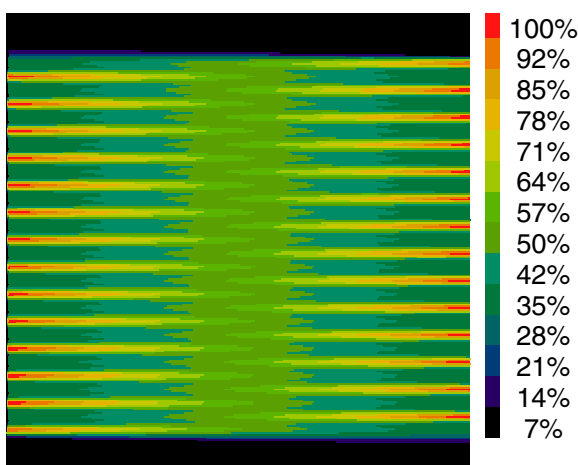


Fig. 117.5. Intensity distribution for both left-to-right and right-to-left side pumping. The right-to-left pumping was given a vertical offset to achieve better uniformity.

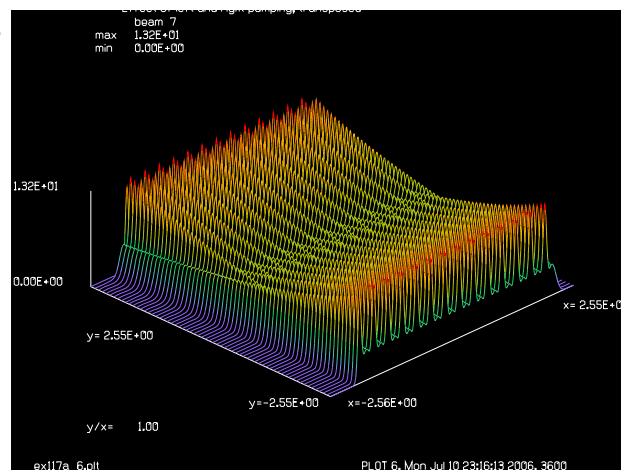


Fig. 117.6. Isometric view of the combined left-to-right and right-to-left pumping as in Fig. 117.5.

Illumination from both the left and right sides may be modeled by summing the distributions for left-to-right and right-to-left illumination. To improve the uniformity of illumination the right-to-left illumination is shifted by one half the diode spacing. A contour plot is shown in Fig. 117.5 and an isometric plot is shown in Fig. 117.6.

Input: ex117a.inp

```
c## ex117a
c
c Simulation of side-pumped laser diode illumination
```

Jump to: [Commands](#), [Theory](#)

```

c
c Shift and add to create 14 line emitters
c
c Beams
c 1      1D array to propagate gaussian (geometric treatment)
c 2      2D array with projected gaussian path, left pumped
c 3      2D array with peak normalization to show width
c 4      temporary array for shifting
c 5      array to hold multiple gaussian paths
c 6      array to hold right pumped distribution with shift
c 7      sum of left and right pumping
c
set/highlight/off
alias name ex117a
variable/dec/int Pass Nline
Nline = 512
array/s 1 Nline 1
nbeam 3 Nline Nline
Units = .01
dZ = .01
Divergence = .02
Width0 = .07          # initial width at gain region
units/s 1 Units 1
units/s 2 Units dZ
Alpha = .02           # loss per cm
macro/def scan/o
    Pass = Pass + 1
    Z = Z + dZ
    Width = Width0 + Z*Divergence
    gaus/c 1 1 Width
    energy/norm 1 1
    Factor = exp(-Alpha*Z)
    mult 1 Factor
    copy/row 1 2 1 Pass
    peak/norm 1 1
    copy/row 1 3 1 Pass
macro/end
clear 2 0
clear 3 0
write/off
macro scan/Nline
write/on
plot/w @name_1.plt
title intensity decays from top to bottom
plot/l 2
plot/w @name_2.plt
title with peak/normalization showing width
plot/c 3 ilab=0
nbeam 4 Nline
nbeam 5 Nline
units/s 0 Units
c
c Make multiple beams
c

```

Jump to: [Commands](#), [Theory](#)

```

Ncycles = 14
Separation = .30 # Separation between lines
macro/def ShiftAndAdd/o
  Pass = Pass + 1
  copy/con 2 4
  rescale/shift 4 x=Pass*Separation
  add/inc/con 5 4
macro/end
Pass = -Ncycles/2
clear 5 0
write/off
macro ShiftAndAdd/Ncycles
write/on
plot/w @name_3.plt
set/density 256
transpose 5          # flip distribution so pumping is horizontal
title Effect of left pumping
plot/bitmap/i/burn 5
transpose 5          # flip distribution so pumping is horizontal
c
c Form an array for right pumping with a small shift
c
nbeam 6
units/s 6 Units
c
c Make a copy of left pumping array
c
copy 5 6
flip/y 6 # flip to make right pumping array
c
c Shift by one half the line source separation
c
rescale/shift 6 x=-Separation/2
plot/w @name_4.plt
title Effect of right pumping with shift
transpose 6
plot/b/i/b 6
transpose 6
nbeam 7
units/s 7 Units
clear 7 0
add/inc/con 7 5 6
transpose 7
plot/w @name_5.plt
title Effect of left and right pumping
plot/b/i/b 7
plot/w @name_6.plt
transpose 7
title Effect of left and right pumping, transposed
plot/l 7 ns=64

```


Ex117b: Side pumping with `slab/pump`

The command `slab/pump` implements pumping from the side with a gaussian beam. An M^2 value may be specified to allow for higher divergence than an ideal gaussian. The incident beam $I(x)$ has units of irradiance [w/cm²]. The absorption factor α results in a deposited energy density of $U(x,y)$ with units [w/cm³]. $U(x,y) = \alpha I(x)$. The command `slab/pump` allows the light to be reflected from the end. This reflected light goes back into the slab contributing to further absorption.

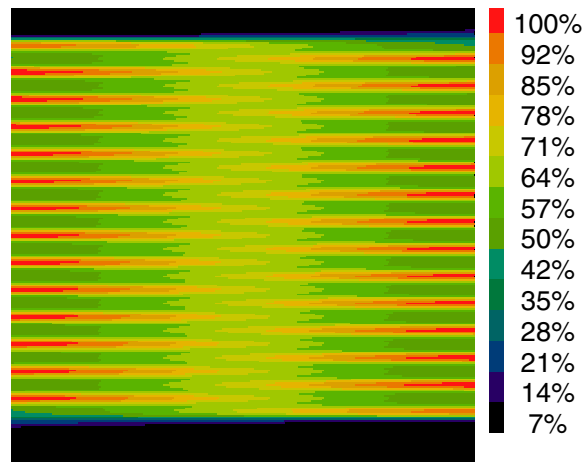


Fig. 117.7. Bitmap image of combined left-to-right and right-to-left pumping by a gaussian beam.

Input: ex117b.inp

```
c## ex117b
c
c Simulation of side-pumped laser diode illumination
c
c Create 14 line emitters
c
c Beams
c 1      sum of left and right pumping
c
alias name ex117b
variable/dec/int Pass Nline

Nline = 512
array/s 1 Nline
Units = .01          # transverse units
dZ = .01             # units in propagation direction
Divergence = .02     # divergence angle
Width0 = .07         # initial width at gain region

Alpha = .02          # loss per cm

M2 = Divergence*pi*Width0/10.6e-4 list  # calculate M^2 value
Zray = pi*Width0^2/10.6e-4 list

units/s 0 Units
c
```

Jump to: [Commands](#), [Theory](#)

```

c Make multiple beams
c
Ncycles = 14
Separation = .30      # Separation between lines
macro/def ShiftAndAdd
    Pass = Pass + 1
    slab/pump/gaus/left kout=1 peak=1. w0=Width0 lambda=10.6 absorption=Alpha
m2=M2 dec=Pass*Separation
    slab/pump/gaus/right kout=1 peak=1. w0=Width0 lambda=10.6 absorption=Alpha
m2=M2 dec=Pass*Separation-Separation/2
macro/end
Pass = -Ncycles/2
clear 1 0
write/off
macro ShiftAndAdd/Ncycles
write/on
plot/w @name_1.plt
set/density 256
title Effect of left and right pumping
plot/bitmap/i/burn 1
C For single pass, peak energy density should be  $\alpha \cdot I(0,0) = 0.02$ .
C Double pass energy density is somewhat higher as defined by
C the peak value below because of the crossing pass.
peak 1

```

Ex117c: Side pumping with `slab/pump` and three rotations

The `slab/pump` command can be implemented with rotations.

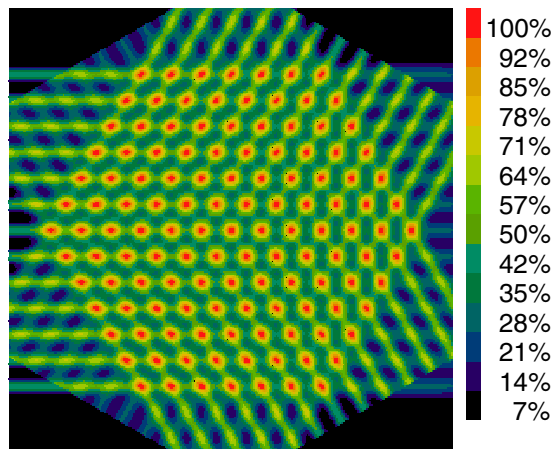


Fig. 117.8. Bitmap image of side pumping from 120 deg. rotations.

Input: ex117c.inp

```

c## ex117c
c
c Simulation of side-pumped laser diode illumination with rotations
c

```

Jump to: [Commands](#), [Theory](#)

```

c Create 14 line emitters with three rotations
c
alias Name ex117c
variable/dec/int Pass Nline Ncycles

Nline = 512
array/s 1 Nline data
array/s 2 Nline data
array/s 3 Nline data
Units = .01          # transverse units
dZ = .01              # units in propagation direction
Divergence = .02      # divergence angle
Width0 = .07          # initial width at gain region

Alpha = .02           # loss per cm

M2 = Divergence*pi*Width0/10.6e-4 list  # calculate M^2 value
Zray = pi*Width0^2/10.6e-4 list

units/s 0 Units
c
c Make multiple beams
c
Ncycles = 13
Separation = .30      # Separation between lines
macro/def ShiftAndAdd
    Pass = Pass + 1
    slab/pump/gaus/left kout=1 peak=1. w0=Width0 lambda=10.6 absorption=Alpha
m2=M2 dec=Pass*Separation
macro/end
Pass = -(Ncycles+1)/2
clear 1 0
write/off
macro ShiftAndAdd/Ncycles
write/on
plot/w @Name_1.plt
set/density 256
title Effect of left pumping, array 1
plot/bitmap/i/burn 1
copy/con 1 2
rotate 2 120
copy/con 1 3
rotate 3 240
plot/w @Name_2.plt
title Effect of 120 degree rotation
plot/bitmap/i/burn 2
plot/w @Name_3.plt
title Effect of 240 degree rotation
plot/bitmap/i/burn 3
add/inc 1 2 3
plot/w @Name_4.plt
title Three beams combined
plot/bitmap/i/burn 1
c Output array

```

Jump to: [Commands](#), [Theory](#)

```
outfile ex117c.dat/no/binary 1
```

Ex117d: Resonator with complex side pumping

The complex side pumping from 117c is used in a stable cavity using the resonator configuration from Ex33. Although the pumping is highly complex, the fine structure does not alter the steady-state mode.

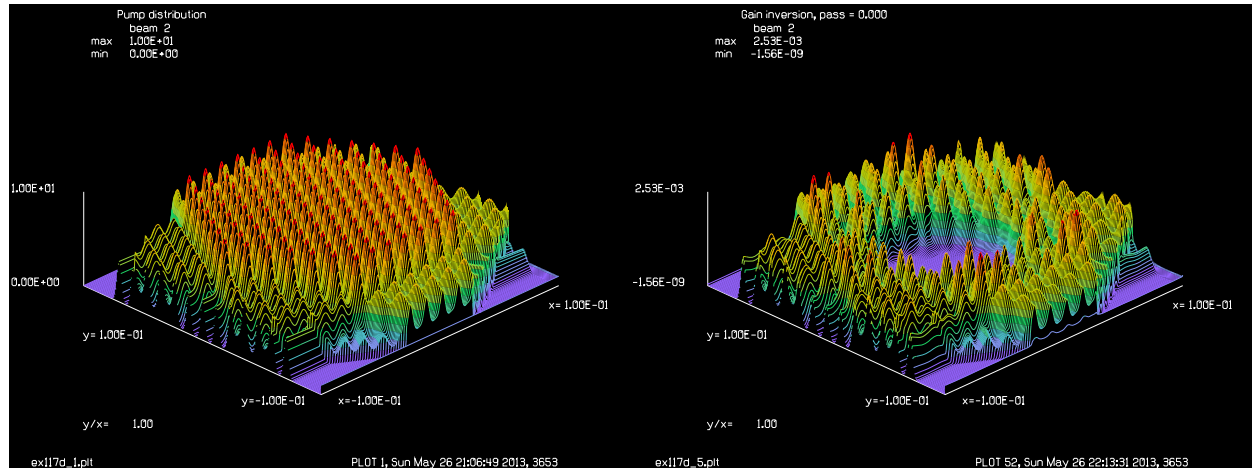


Fig. 117.9. A complex pumping distribution from Ex117c.

Fig. 117.10. Population Inversion after 5,000 cycles showing significant depletion in the center due to the steady-state mode.

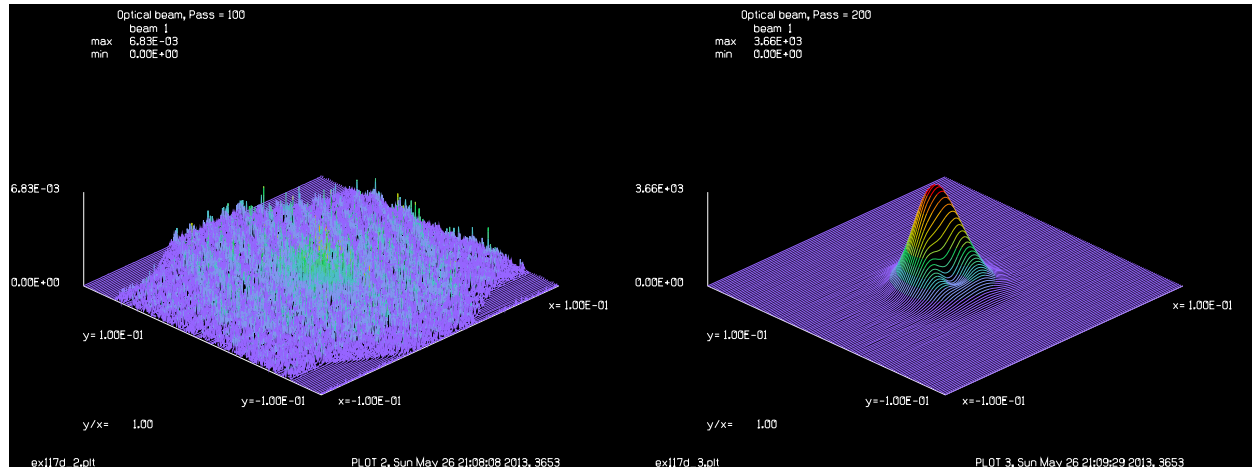


Fig. 117.11. Optical beam distribution after 100 passes starting from spontaneous emission noise showing some residual influence from the complex pumping distribution shown in Fig. 117.9.

Fig. 117.12. Optical distribution after 200 passes. The influence of the complex pumping distribution apparent at 100 passes in Fig. 117.11. is no longer visible. The mode has begun to approach the steady-state condition.

Input: ex117d.inp

```
c## ex117d
c
c Example ex117d: half symmetric YAG laser with complex pump
```

Jump to: [Commands](#), [Theory](#)

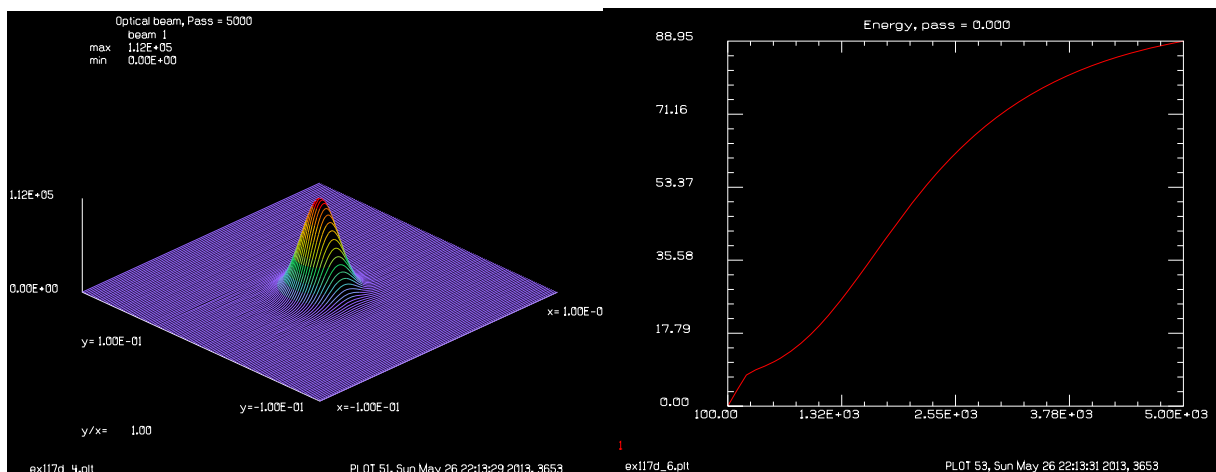


Fig. 117.13. Optical distribution at 5,000 passes showing steady-state form.

Fig. 117.14. Energy in cavity mode after 5,000 passes showing that the energy is approaching the steady-state condition.

```

c
c Run ex117c before ex117d to make ex117c.dat for pump
c
c Beams
c -----
c Beam 1    optical field in the cavity
c Beam 2    medium array
c Beam 3    temporary array
c Beam 4    512 x 512 array for inversion from ex117cx
c
alias Name ex117d
variab/dec/int Pass switch Nline Status Count
Nline = 1024
x = srand(101)
array/s 1 Nline Nline 1
nbeam 2 Nline Nline 1 data          # make gain array
nbeam 3 Nline Nline data            # temporary array
nbeam 4 512 512 data                 # Read in gain
units/field 1:3 .2
units/field 4 .1
wavelength/set 0 1.06               # set wavelength and index
width = 1.2e10                      # set line width, hz
center = 2.83e14                    # set line center, hz
tspont = 3e-4                       # set spontaneous emission time, sec
t20 = 3e-1                          # set decay time for level 2, sec
t10 = 1e-5                          # set decay time for level 1, sec
mode_sep = 5e8                      # set longitudinal mode separation, hz
time = 2e-9
L = 1
c
c Set offset from line center of first mode, 0 hz
c Set fractional pumping into level 1, nlpump, 0
c
gain/rate/set width center tspont t20 t10 mode_sep 0 0

```

Jump to: [Commands](#), [Theory](#)

```

gain/rate/noise
gain/rate/list
pack/set 1 2                # pack both beams
macro/def ex117dx/o
  variables/set Status resonator/status
  Pass = Pass + 1
  prop 45
  mirror/sph 1 -50
  clap/c/n 1 .14
  prop 45
  pack/in
    gain/rate/step L time
  pack/out
  mirror/flat 1
  pack/in
    gain/rate/step L time
  pack/out
  if [mod(Pass,100)==0] then
    Count = Count + 1
    variable/set Energy 1 energy
    CC Pass = @Pass, Energy = @Energy
    udata/set Count Pass Energy
    if Count<3 then
      if Count=1 then
        plot/w @Name_2.plt
        title Optical beam, Pass = @Pass
        plot/l 1 xrad=.1 ns=128
      else
        plot/w @Name_3.plt
        title Optical beam, Pass = @Pass
        plot/l 1 xrad=.1 ns=128
      endif
    else
      plot/w @Name_4.plt
      title Optical beam, Pass = @Pass
      plot/l 1 xrad=.1 ns=128
    endif
  endif
endif
macro/end
time = 2e-7
tot_time = 0
trans = .95
refl = 1. - trans
Pass = 0
c
c Read in N2 pump from ex117c into Array 4
c Run ex117c before ex117d
c
infile ex117c.dat/no/binary 4
peak/norm 4 1e1
clear 2 0.
copy/con 4 3
gain/rate/n2pump 3 2
plot/w @Name_1.plt

```

Jump to: [Commands](#), [Theory](#)

```
title Pump distribution
plot/l/real 2 xrad=.1 ns=256
Pass = 0
resonator/name ex117dx
gaus/c/c 1 1 .1
resonator/eigen/test 1
clear 1 0.                # let noise build up
pass = 0
write/off
reson/run 5000
write/on
plot/w @Name_5.plt
title Gain inversion, pass = @pass
plot/l/ginversion 2 xrad=.1 ns=256
plot/w @Name_6.plt
title Energy, pass = @pass
plot/udata
```


Ex118: Partial coherence for a 3D object

This example illustrates partial coherence effects of a 3D object illuminated with wide band illumination. We consider a simple 3D target consisting of two reflecting disks separated both transversely, to create modulation in the far-field image, and longitudinally. See Fig. 118.1. If the longitudinal separation is comparable to or larger than the coherence length $\lambda^2/\Delta\lambda$, the beam will exhibit partial coherent properties. For this case the modulation in the far-field image will be reduced as the L approaches or exceeds the coherence length.

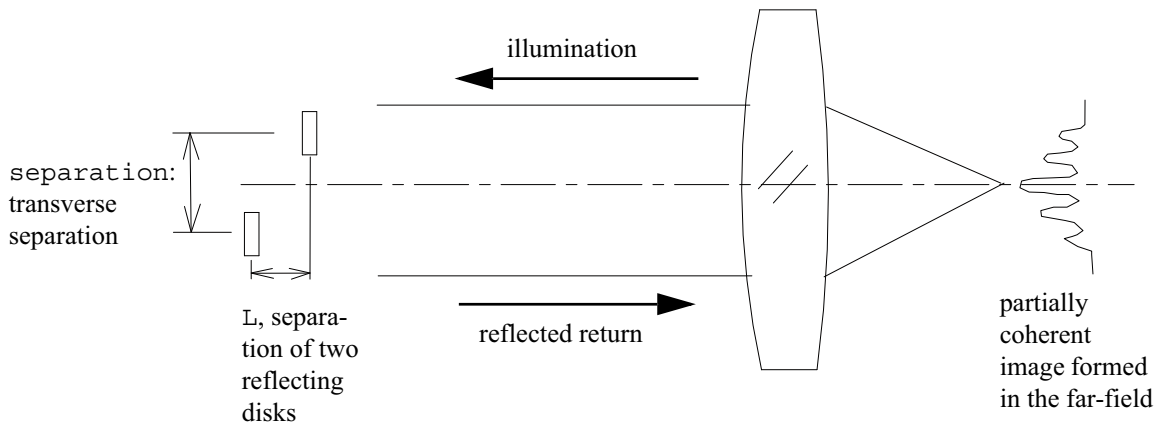


Fig. 118.1. A 3D object consists of two reflecting disks separated transversely by *separation* and longitudinally by L . The object is illuminated by a collimated, wide spectral band beam. The transverse separation leads to a modulated far-field pattern. The longitudinal separation L , if comparable to or greater than the coherence length, will create a partially coherent image. The larger the separation L , the more incoherent the image will be with decreased modulation in the image.

The modeling consists simply of forming the far-field image for a range of wavelength samples over the spectral bandwidth. Figs 118.2 and 118.3 show the far-field distribution for a single choice of wavelength. This is the fully coherent result. For an arbitrary wavelength the pattern may be asymmetrical, as shown in Fig. 118.4.

When choosing a range of wavelengths, the incoherent combination of patterns with varying asymmetries will result in some smoothing out of the coherent pattern to create the partial coherent result. Fig 118.5 shows a yellow line that is the spectral bandwidth for which the coherence length would exactly match the round-trip longitudinal separation of the reflecting disks. For a narrower spectral band the image will exhibit noticeable partial coherence effects. For a significantly wider spectral band, the image will appear to be nearly incoherent. The partially coherent images are formed by incoherently summing all the samples of the bandwidth, taking into account the weighting of according to the spectral curve. Figs. 118.6 and 118.7 show the resulting partially coherent image.

Fig. 118.8 shows a spectral band at about twice that needed to match the coherence length to the round-trip path depth of the object. Figs. 118.9 and 118.10 show the high degree of incoherence in the resulting far-field image.

Input: `ex118.inp`

c## ex118

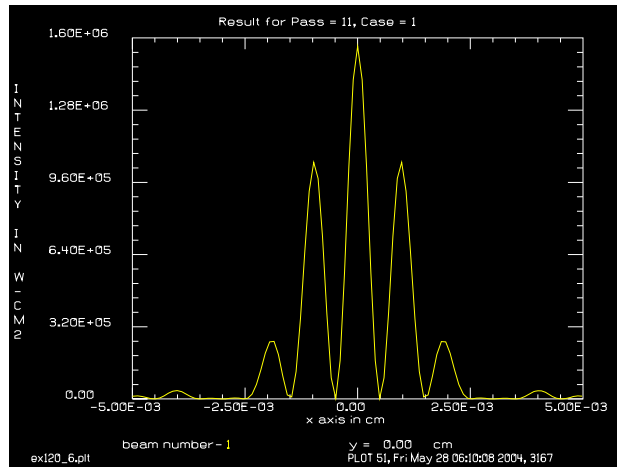


Fig. 118.2. Far-field image from the two reflecting mirrors for a single wavelength. This is the fully coherent image. For this case, the round-trip longitudinal path was an exact multiple of the wavelength.

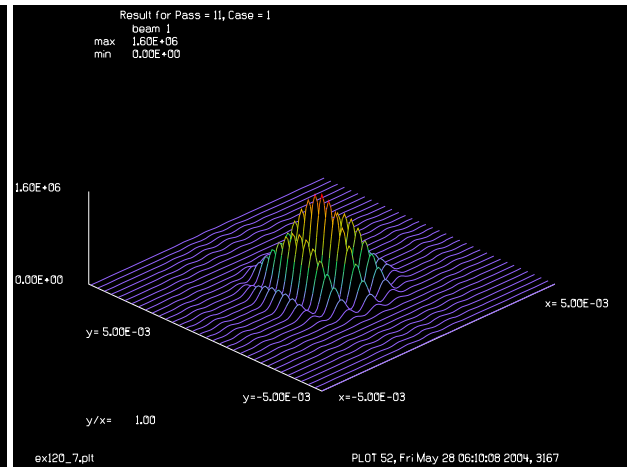


Fig. 118.3. Isometric plot of the coherent case also illustrated in 118.2.

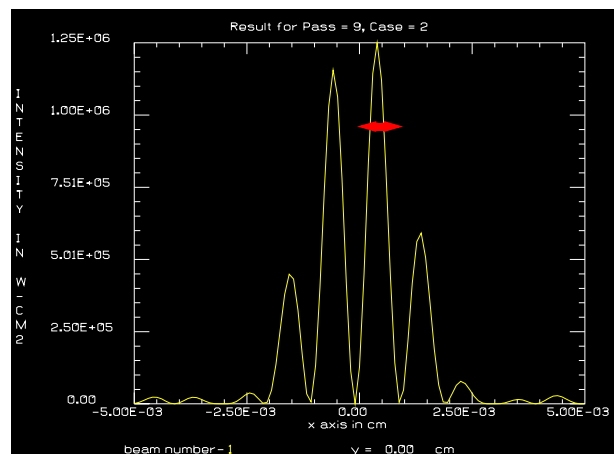


Fig. 118.4. Coherent image with asymmetry because the wavelength for which the round-trip longitudinal path difference between the two reflecting disks is not an even multiple of the wavelength. For various wavelengths the peaks giggle back and forth (represented by the horizontal double red arrow) such that integration over a wide spectral band will result in decreased modulation.

```

c
c Modeling 3D coherence
c
c Finite spectral width defined a gaussian function in terms of
c wavelength about the center point Lambda0 illuminates two reflecting
c disks.
c
c The two disk reflectors if diameter Diameter = 2 are separated
c longitudinally by L = 5 and transversely separated by
c Separation = 5.
c
c First reflection occurs at Object 1 and the transmitted light
c propagates to Object 2, is reflected of Object 2, and then

```

Jump to: [Commands](#), [Theory](#)

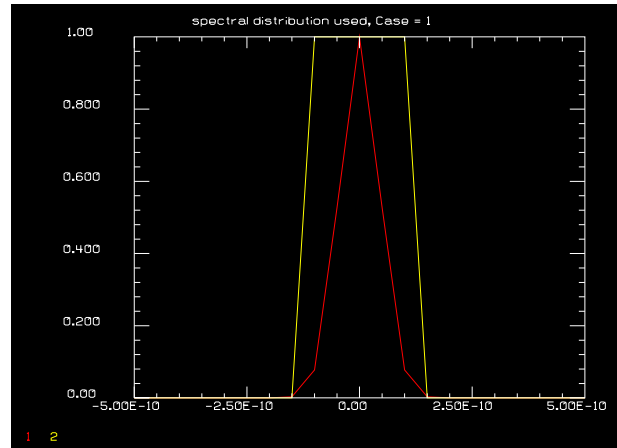


Fig. 118.5. Plot of spectral band (red) showing the case where the band is narrower than the band width (yellow) needed to match the coherence length to the round-trip reflection path difference. This narrower spectral band will result in partial coherence in the far-field image. The more narrow the more band width the more coherent the result.

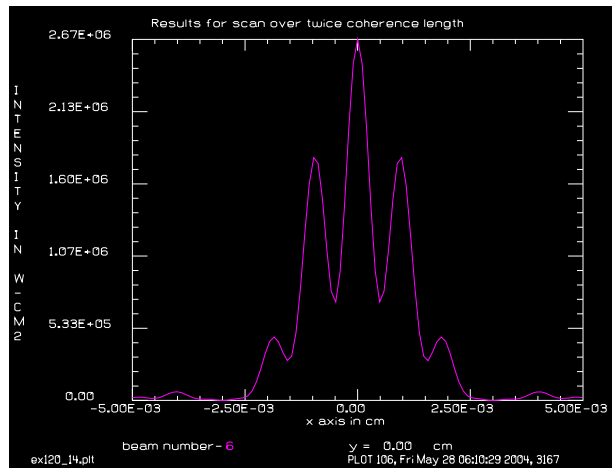


Fig. 118.6. Profile through the far-field image for the narrow spectral band, illustrated in Fig. 118.5, showing the reduced modulation of partial coherence imaging.

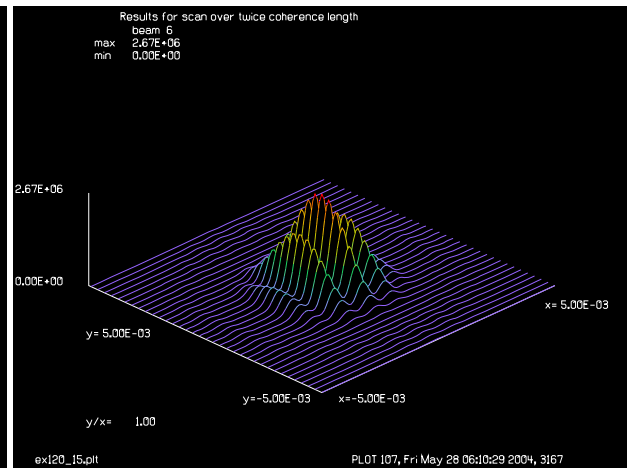


Fig. 118.7. Same case as Fig. 118.6 for the spectrum shown in Fig. 118.5. The image shows partial coherence effect of reduced modulation.

```

c propagates back to Object 1.
c
c The round-trip phase is explicitly calculated for each wavelength
c with respect to the center wavelength.
c
c The interference pattern is observed in the far-field.
c If the spectral bandwidth is sufficient to create a coherence length
c less than  $L = 5$ , the interference function integrated over all
c wavelengths approaches the incoherent sum.
c
c The variable CoherenceLength is set to match the round-trip length of the
c longitudinal separation of the reflecting disks.
c
variable/dec/int Nline Pass Ntimes Pass Case
alias name ex118
Nline = 512

```

Jump to: [Commands](#), [Theory](#)

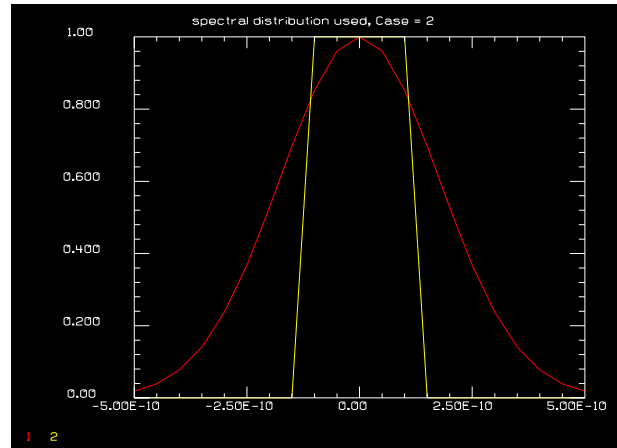


Fig. 118.8. Plot of spectral band (red) showing the case where the band is wider than the band width (yellow) needed to match the coherence length to the round-trip reflection path difference. This wider spectral band will result in near incoherence in the far-field image.

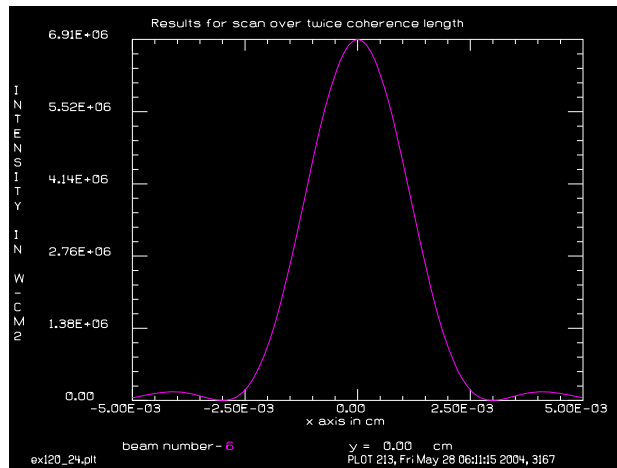


Fig. 118.9. Profile of far-field image for the wide spectral band illustrated in Fig. 118.8 with a high degree of incoherence. This was calculated as Case 2.

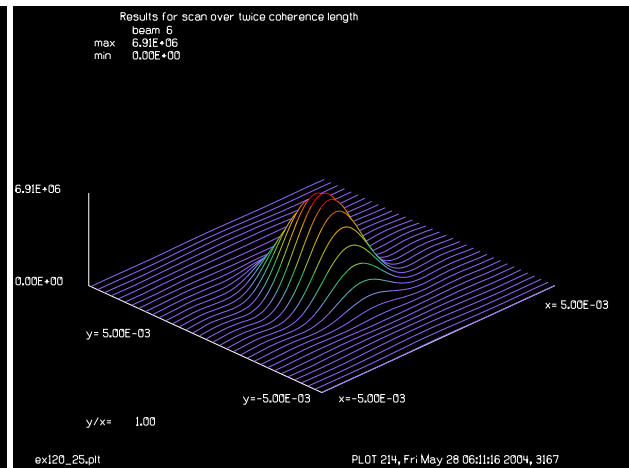


Fig. 118.10. Isometric plot of the incoherent far-field image. This corresponds to the wide spectral band width case of Fig. 118.9 and is the same image as represented in Fig. 118.8.

```
array/s 1 Nline
Lambda0 = .5e-4
nbeam 7 data
c
c Beam 1 propagating beam
c Beam 2 Object 1, transmission
c Beam 3 Object 1, reflection
c Beam 4 Object 2, reflection
c Beam 5 temporary use
c Beam 6 incoherent sum
c
Units = .1
units/set 0 Units
Diameter = 2.0 # diameter of reflecting disks
```

Jump to: [Commands](#), [Theory](#)

```

Separation = 5      # transverse separation of reflecting disks
L = 5              # longitudinal separation of reflecting disks
c
c Compute spectral half-width such that 2*L is one coherence length
c
PlotWidth = 5e-3
c
c make transmission mask (2) and obscuration mask (3)
c for Object 1
c
units 0 Units
wavelength 0 Lambda0*1e4
c
c make transmission mask (2) and reflection mask (3)
c for Object 1
c
obs/c 2 Diameter/2 x=Separation/2
copy/con 2 3
c
c Make reflection mask for Object 1
c
clear 5 1
phase/piston 5 180
add/coh/con 3 5      # make exact complimentary mask
amplitude/abs 3
c
c Make reflection mask for Object 2
c
clap/c 4 Diameter/2 x=-Separation/2

macro/def system/o
c
c Macro to create reflection from Object 1, propagation to Object 2
c and propagation back to Object 1 after reflection at Object 2.
c
c The interference pattern is observed in the far-field.
c
  zreff 1 0
  Lambda = Lambda0 + dLambda
  wavelength 0 Lambda*1e4
  array/s 1 Nline      # starting collimated beam
  peak/norm 1 Peak
  units/set 1 Units
  copy/con 1 5
  mult/beam 1 2        # Object 1, transmission on Beam 1
  mult/beam 5 3        # Object 1, reflection on Beam 5
  prop L
  mult/beam 1 4        # Object 2, reflection on Beam 1
  prop L
  mult/beam 1 2        # Object 1, transmission on Beam 1
c
c Calculate the round-trip phase relative to the center wavelength
c
  Waves = 2*L*dLambda/Lambda0^2 list # differential wavefront error

```

Jump to: [Commands](#), [Theory](#)

```

phase/piston 1 360*Waves    # phase due to extra step to Object 2 and back
add/coh/con 1 5             # add reflection from Object 1 to Object 2
c
c Create the far-field pattern
c
  lens 1 100
  prop 100
  units/beam 6 1
  add/inc/con 6 1
macro/end
Ntimes = 21
Ncenter = (Ntimes+1)/2
Pass = 0
c make window coordinates
Width = 400
Height = 300
x1 = 10; y1 = 10
x2 = x1+Width; y2 = y1
x3 = x2+Width; y3 = y1
x4 = 10; y4 = 10+Height
x5 = x4+Width; y5 = y4
x6 = x5+Width; y6 = y5
x7 = 10; y7 = 10+2*Height
x8 = x7+Width; y8 = y7
x9 = x8+Width; y9 = y8
macro/def spectral/o
  Pass = Pass + 1
  dLambda = ddLambda*(Pass-(Ntimes+1)/2)
  Peak = gauss(dLambda,0,LambdaHalfWidth,1.)
  macro system
    if [Pass==Ncenter && Case==1] then
c Make a plots of the center wavelength to show
c the fully coherent result.
      plot/watch @name_6.plt x6 y6 Width Height
      title Result for Pass = @Pass, Case = @Case
      plot/x/i fmax=1.6e6 le=-PlotWidth ri=PlotWidth

      plot/watch @name_7.plt x7 y7 Width Height
      plot/l 1 max=1.6e6 xrad=PlotWidth
    endif
    if [Pass==9 && Case==2] then
c Make a plots of the center wavelength to show
c the fully coherent result.
      plot/watch @name_8.plt x8 y8 Width Height
      title Result for Pass = @Pass, Case = @Case
      plot/x/i 1 le=-PlotWidth ri=PlotWidth

      plot/watch @name_9.plt x9 y9 Width Height
      plot/l 1 xrad=PlotWidth
    endif
    plot/watch @name_@Case1.plt x1 y1 Width Height
    title Result for Pass = @Pass, Case = @Case
    plot/x/i fmax=1.6e6 le=-PlotWidth ri=PlotWidth
  endmacro

```

```

plot/watch @name_@Case2.plt x2 y2 Width Height
plot/l 1 max=1.6e6 xrad=PlotWidth

udata/set Pass dLambda Peak [abs(dLambda)<=CoherenceHalfWidth]
plot/watch @name_@Case3.plt x3 y3 Width Height
title spectral distribution used, Case = @Case
plot/udata first=1 last=2 left=-0.5e-9 right=0.5e-9 min=0 max=1

plot/watch @name_@Case4.plt x4 y4 Width Height
title Results for scan over twice coherence length
plot/x/i 6 le=-PlotWidth ri=PlotWidth

plot/watch @name_@Case5.plt x5 y5 Width Height
plot/l 6 xrad=PlotWidth
macro/end
Pass = 0
Case = 1
clear 6 0
CoherenceHalfWidth = Lambda0^2/(2*L)/2 # set coherence width to roundtrip
depth of 3D object
LambdaHalfWidth = CoherenceHalfWidth/2 list # 1/2 coherence width
ddLambda = 4*CoherenceHalfWidth/((Ntimes-1)/2) # calculate over 4
CoherenceHalfWidth
macro spectral/Ntimes
Pass = 0
if 0 then
macro/def reset/o
    Pass = Pass + 1
    udata/set Pass y01=0
macro/end
macro/run reset/Ntimes
endif
udata/maxrows 0
Pass = 0
Case = 2
clear 6 0
LambdaHalfWidth = 2*CoherenceHalfWidth list # 2 coherence width
macro spectral/Ntimes

```


Ex119: Sub-round-trip sampling of resonator

It is possible to observe effects in a resonator on a shorter time scale than the round trip time. This may be done by using multiple resonator cavity modes. Figure 119.1 shows a cavity mode sampled by sixteen time samples numbered from 2 to 17. The leading edge of time pulse is 2 and crosses the gain region simultaneously with the opposite side 10. In the model, the beams must alternate as 2, 10, 3, 11, etc. The output is identified by 5'. In Figure 119.2 the samples have advanced 1/16 of the round trip time with samples 2 and 10 having crossed through the gain region. The output is from sample 6 at the partially reflecting mirror on the left.

This approach requires sixteen beams and each of the sixteen would need to be advanced by 1/16 of the round trip distance, so the total time for calculation would be 256 times longer than for a single cavity sample.

The number of propagation steps may be reduced by bunching the samples at either end of the resonator and propagating each sample from one end of the other of the resonator. Figures 119.3 and 119.4 show schematically the bunched model. Initially beams 2 through 9 are stored on the left and beams 10 through 17 are stored on the right. Sample 2 is sent from left to right and stored in the stack on the right. Sample 10 is sent from right to left and stored on the left.

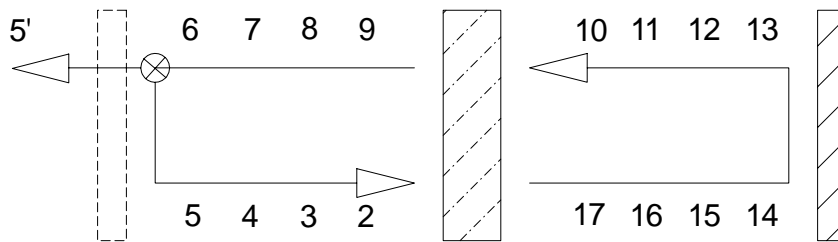


Fig. 119.1. The resonator is assumed to contain sixteen time samples of the cavity mode numbered 2 to 17 and proceeds in a circular fashion. The leading edge of the pulse is at 2 which crosses the gain region first. The opposite side of the pulse is 10 and crosses simultaneously with 2 but is modeled as crossing just after 2.

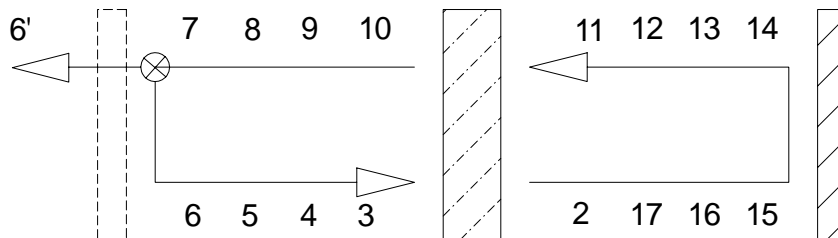


Fig. 119.2. After 1/16 of the round trip time time samples 2 and 10 have advanced across the gain region and the output is sample 6'.

For clarity in construction of the model, each of the sample beams is copied to Beam 1 and all propagation applied to Beam 1 and Beam 1 is used for interaction with the gain region. Beam 1 is then copied back into the proper beam for storage. With this procedure only Beam 1 has attribute “beam” and is

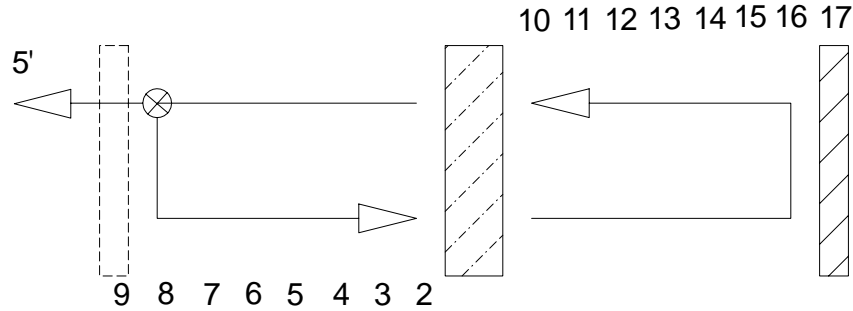


Fig. 119.3. Model with eight of the samples bunched at either end of the resonator. Each sample is sent from one mirror to the other before the next sample from the other side is processed. As shown, sample 2 is about to be sent from the left mirror through the gain region to the right mirror. Then sample 10 is sent from the right mirror to the left.

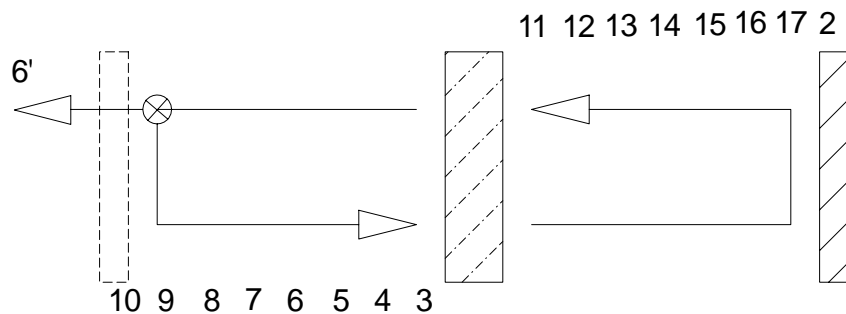


Fig. 119.4. The bunched model after one time step. Samples 2 and 10 have crossed the length of the resonator and have been stored in the stack on the opposite side. The output is taken from sample 6.

propagated. All other beams have attribute “data”. The time spent copying the beams could be saved by rewriting the example to propagate `lbeam1` and `lbeam2`, as required.

Example `ex119` illustrates this procedure. The round trip time is sampled by sixteen arrays. After 512 total passes each sample has passed through the resonator 32 times. Figure 119.5 shows the output of one of the time samples, showing that the resonator has converged relatively well. Figure 119.6 shows the output power after 200 total passes. The leading sub-samples are 2 for the forward pass and 10 for the reverse pass. In the saturated regime, these leading sub-samples lead to greater saturation for the following sub-samples, giving the decaying waveform shown in Fig. 119.6 where sub-samples 2 through 9 and also 10 through 17 form pairs of decaying waveforms. Figure 119.7 shows the waveform after 512 passes—a total of 32 passes for each sub-sample.

In this example, the gain was effectively switched on instantaneously. Other forms of transient behavior could be modeled with this method.

Input: `ex119.inp`

```
c## ex119
c
c Example 119: Sub-round-trip-time sampling of a resonator
c
c In this example, we split the round trip time into 16 sub pulses.
c
alias Name ex119
```

Jump to: [Commands](#), [Theory](#)

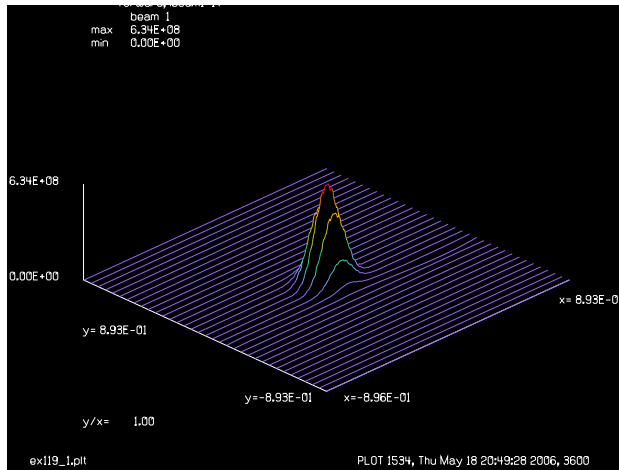


Fig. 119.5. Converged mode after 512 total passes amounting to 32 passes for each time sample through the resonator.

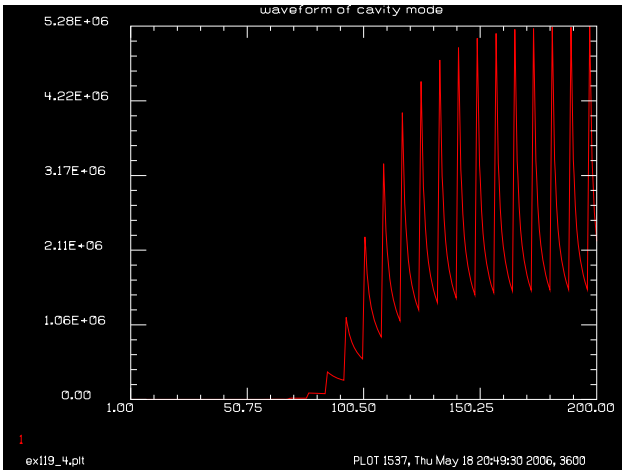


Fig. 119.6. Waveform of output after 200 passes. Note that a sloped waveform has developed when the medium begins to saturate, because the leading edge always has the highest gain.

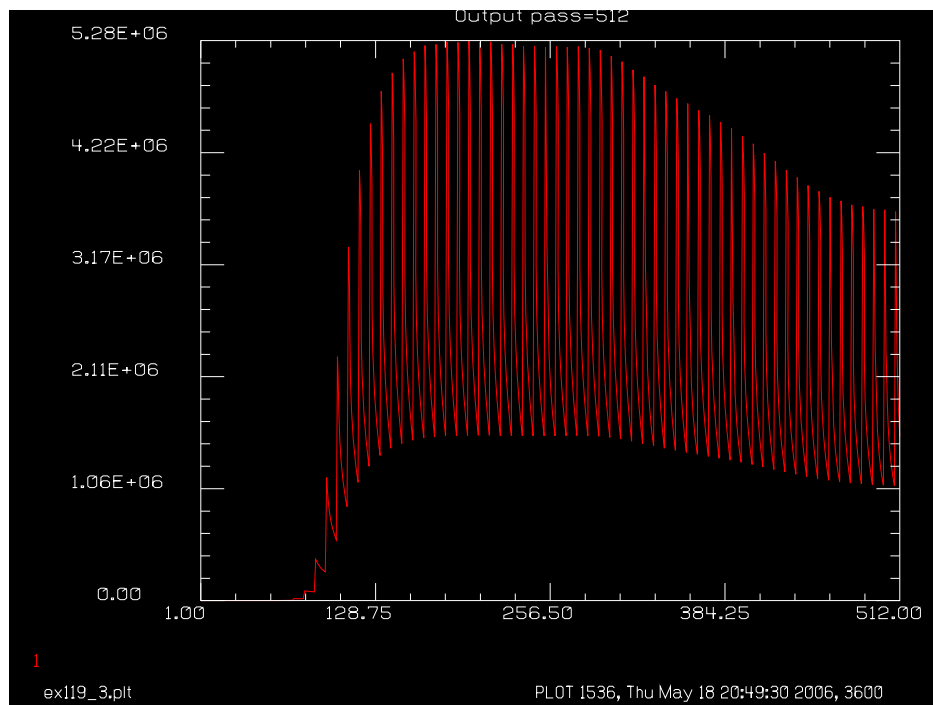


Fig. 119.7. Waveform after 512 passes.

set/highlight/off

c N is the pass counter

c M is the number of time samples

c GainSteps is the number of steps through the gain region

variab/dec/int N M Pixls GainSteps

variab/dec/int MediumBeam SetupBeam OutCoupledBeam CumulativeBeam

variable/dec/int i Pass TEST

Jump to: [Commands](#), [Theory](#)

```

variab/dec/int lbeam1 lbeam2 lbeam3
variab/dec/real Units
c
c lbeam1 is beam from left
c lbeam2 is beam from right
c lbeam3 is output beam
c
variab/monitor/add lbeam1 lbeam2 lbeam3 N

M=16                                # number of sub-pulses in one round trip
if [M>1] then
  MediumBeam = M + 2 list
else
  MediumBeam = M + 1 list
endif
OutCoupledBeam = MediumBeam + 1 list      # Outcoupled beam
CumulativeBeam = OutCoupledBeam + 1 list  # Cumulative beam
SetupBeam = SetupBeam + 1 list           # setup medium
Units=3.5e-3
Pixls=512
array/set 1 Pixls Pixls 0# intracavity beam
c use "data" for non-propagating beams.
nbeam MediumBeam-1 data
nbeam MediumBeam ipol=1 data             # 2-9 storage beams and 10 gain medium,
polarized
nbeam CumulativeBeam ipol=0 data        # 9, integrated irradiance

unit/set 0 Units
wavelength/set 0 1.064

IndSlab=1.818
SlabAperX=0.25
SlabAperY=0.24
Lslab=1.
L1 = 30

OPLcavity = L1 + Lslab*IndSlab + L1 list

dt = 2*OPLcavity/c                   # round trip time

Lwidth = 1.19e11                      # line width, in Hz
Lcenter = 2.82e14                     # line center, in Hz
xsect=2.606e-19                       # cross section
xsect=xsect*.3                        # arbitrary reduction of cross section
                                     # to illustrate effects
# spontaneous emission time, in sec
Tspont = 0.0001064^2/(4*1.818^2*pi^2*Lwidth*xsect)

Ttot=230e-6
T20 = 1/(1/Ttot-1/Tspont)
T10=5e-9
Mode_sep = (3.0e10)/(2*OPLcavity)
GainSteps=1                          # number of gain steps

```

Jump to: [Commands](#), [Theory](#)

```

Ptime=128e-6          # pump pulse duration

PYAG=280              # mJ

clear 1:CumulativeBeam    # clear beams

clear SetupBeam 1.       # used for setup

clap/rec/no SetupBeam SlabAperX SlabAperY
variable/set energySetup SetupBeam energy
mult/scalar SetupBeam PYAG/(1000*LSlab*energySetup)

gain/rate/n2level/set SetupBeam MediumBeam
gain/rate/nonoise
q0=exp(-Ptime/Ttot)
mult/scalar SetupBeam 1/(Ttot*(1-q0))
gain/rate/n2pump/set SetupBeam MediumBeam

gain/rate/set Lwidth Lcenter Tspont T20 T10 Mode_sep 0 0 0
R=0.284                # reflectivity of OC
R = .95
N=0                     # initialize Pass counter

lbeam2=2+mod(N+M/2-1, M) list    # initialize second rotating counter

c beams 10-17 will start from far side backward

TEST=1

macro/def pulse/o
  Pass = Pass + 1 list

  C z = 0

  if TEST=0 then
    N=N+1                list
    lbeam1=2+mod(N-1, M)  list# first rotating counter
    if [M>1] copy/con lbeam1 1
    zreff/list 1

    C Pass = @Pass. Start beam @lbeam1 in forward pass. N = @N.
  endif
  variable/set energy1 1 energy
  title Cavity entry, Power=@energy1 W, N=@N, lbeam1=@lbeam1

  c --- forward pass -----

  prop L1

  C z = L1

  clap/rec/no 1 SlabAperX SlabAperY
  if TEST=0 then

```

```

    pack/in
      gain/rate/step LSlab dt/M GainSteps
    pack/out
  endif

wavelength/set 1 1.064 IndSlab
prop LSlab
wavelength/set 1 1.064 1
clap/rec/no 1 SlabAperX SlabAperY

C z = L1 + LSlab

prop L1

C z = L1 + LSlab + L1

clap/rec/no 1 SlabAperX SlabAperY

if TEST=0 then
  plot/watch @Name_1.plt
  title forward, lbeam1=@lbeam1
  plot/l 1
endif
mirror/flat 1 # mirror 1
variable/set energy1 1 energy

if TEST=0 then
c      Put beam from forward pass into storage
  lbeam1=
  if [M>1] copy/con 1 lbeam1 # store beam

  zreff/list 1
  C Pass = @Pass. Store beam @lbeam1 at end of forward pass. N = @N.
endif

c
c ----reverse pass -----
c
if TEST=0 then
  c      Start reverse pass.
  c      Copy stored array to Beam 1.
  write/screen/on
  lbeam2=2+mod(N+M/2-1, M) list # second rotating counter
  if [M>1] copy/con lbeam2 1
  C Pass = @Pass. Start beam @lbeam2 in reverse pass. N = @N.
endif

C z = L1 + LSlab + L1

prop L1

C z = L1 + LSlab

clap/rec/no 1 SlabAperX SlabAperY

```

```

if TEST=0 then
  pack/in
    gain/rate/step LSlab dt/M GainSteps
  pack/out
endif

wavelength/set 1 1.064 IndSlab
prop LSlab 1
wavelength/set 1 1.064 1
clap/rec/no 1 SlabAperX SlabAperY

C z + L1

prop L1

clap/rec/no 1 SlabAperX SlabAperY
if TEST=0 then
  plot/watch @Name_2.plt
  title reverse, lbeam2=@lbeam2
  plot/l 1
endif

C z = 0

if [TEST==0] then
  c
  c ---- Output coupling
  c
  lbeam3=2+mod(N+M/4-1, M) list      # third rotating counter
  if [M>1] copy lbeam3 OutCoupledBeam
  variable/set power OutCoupledBeam energy
  udata/set N N power*(1-R)
  mult/scalar OutCoupledBeam 1-R
  add/incoh CumulativeBeam OutCoupledBeam
endif

c ---- beam splitter efficiency  mult/scalar 1 R
mirror/flat 1 # mirror 2

variable/set energy1 1 energy
title End backward pass, Power=@energy1 W, N=@N, lbeam2=@lbeam2
if TEST=0 then
  write/screen/on
  if [M>1] copy/con 1 lbeam2          # store beam
  zreff/list 1
  C Pass = @Pass. Store beam @lbeam2 at end of reverse pass. N = @N.
  energy/list 2:17
  pause 2
endif

if TEST=0 then
  title Output pass=@Pass
  plot/w @Name_3.plt
  plot/udata min=0

```

```
endif
macro/end

resonator/name pulse
pack/set 1 MediumBeam
gaussian/cir 1 1 0.25      # gaussian beam used for test
Pass = -1
resonator/eigen/test 1
resonator/eigen/set 1
TEST=0
clear 1:17 1              # clear beams 1 to 17 to 1.0
resonator/run 512          # each beam cycles 32 times
plot/w @Name_4.plt
title waveform of cavity mode
plot/udata left=1 right=200 min=0
```


Ex120: Multi-pass amplifier

Table. 120.1. Table of Ex120 examples

Ex120a: Multipass amplifier, pumping	3
Ex120b: Multipass amplifier, propagation.	6

This example illustrates an amplifier that has five passes of the optical beam through the gain region. Fig. 120.1 show schematically five passes made with four mirrors. It is necessary to interact the optical beam at various positions with the gain region. If the problem is set up so that the pixels of the optical beam always line up exactly with the pixels of the gain array, then no interpolation is required. For the configuration of Fig. 120.1; if we make sure the rise of the beam for the slanted paths is a multiple of six pixels, then the rise of the five gain sheets will be successively multiples of 1, 2, 3, 4, and 5 pixels.

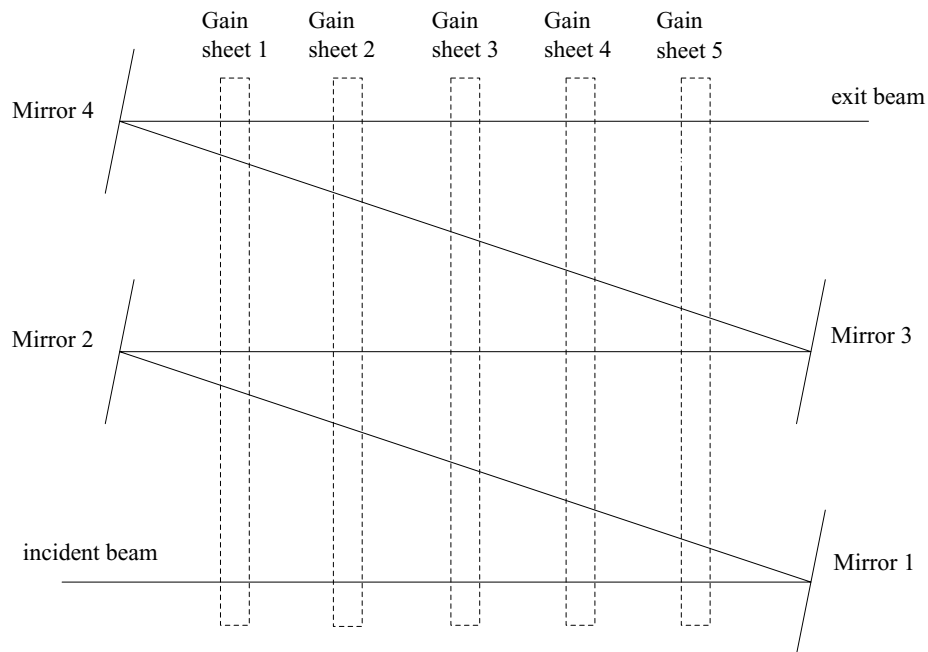


Fig. 120.1. A beam makes five passes through an amplifier region. In the two expanding paths, the optical beam will interact with each of the gain sheets at different points. For this configuration, if the rise in pixels is a multiple of six pixels, then the rise at the gain sheets will be successively multiples of 1, 2, 3, 4, and 5 pixels.

The beam passes through each gain region five times—each at different positions. We can represent the gain region as a rectangular array that is larger than the optical beam as illustrated in Fig. 120.2. The optical beam is incident on the gain region as indicated in the upper left region of Fig. 120.2. The region of the gain array that matches the optical beam is extracted and the extracted section interacts with the optical beam and is finally reinserted into the full gain array.

The same procedure may be used to extract a small section from a large aberration sheet. The the aberration section is then applied to the optical beam. The aberration section is not changed and need not be returned.

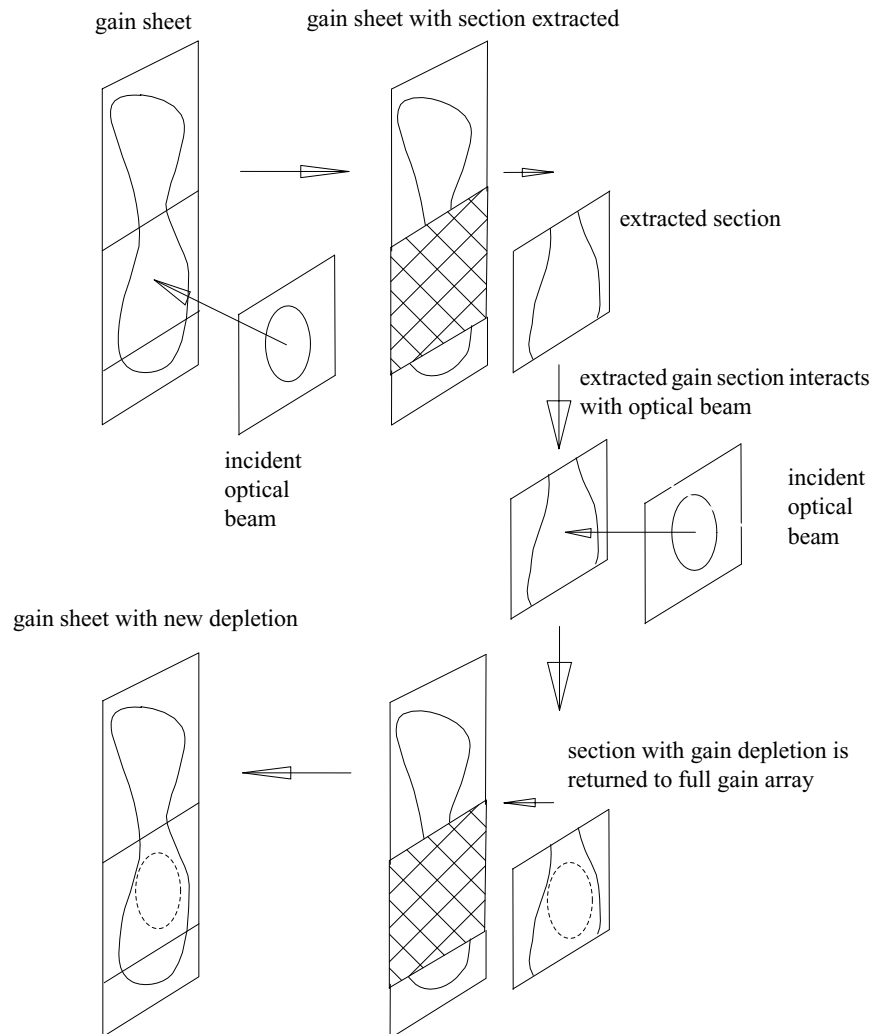


Fig. 120.2. An optical beam in a square array intercepts a rectangular gain sheet (upper left). We extract a square section of identical size to the optical beam (upper right). The extracted square section of the gain interacts with the square optical array (middle right) such the optical beam is amplified and the gain region is somewhat depleted (shown as a circle). The partially depleted square beam is returned to the full gain array (lower right), resulting in the full gain sheet having the new depletion due to the incident optical beam (lower left). The optical beam is correspondingly amplified.

Ex120a: Multipass amplifier, pumping

The calculation begins with the construction of five gain sheets and five corresponding thermal arrays. The full size arrays are set to 2048×512 , but the gains sheets and thermal arrays do not require high spatial resolution and the calculations may be performed with four times coarser sampling at 512×128 . The smaller size arrays lead to much faster thermal equilibration calculation times. The setup command file is ex120a.inp.

Input: ex120a.inp

```
c## ex120a
c
c Multipass amplifier, pumping
c
c This command file sets up the side pumping to establish the
c gain and the thermal distribution.
c
c The output of this command file will be picked up in a second
c command file and used to establish the population inversion
c and the thermal distortions.
c
c It is assumed that a bank of laser diodes pump the gain
c medium from the side.
c
c The full array size is 2048 x 512, but pumping and thermal
c effects can be represented by 512 x 128 arrays.
c
alias Name ex120a
variable/dec/int Nlinex Nliney Icent MediaTemp MediaTemp1 MediaWork
variable/dec/int Media Media1 Media2 Media3 Media4 Media5 Pass
variable/dec/int History Row
variable/dec/int I MediaWork1 Thermal_small
variable/dec/int Scale
variable/dec/int STEADY_STATE

Nlinex = 2048          # horizontal pixels, full size array
Nliney = 512          # vertical pixels, full size array
Lambda = 1.064e-4      # set wavelength
Units = 2/6/30 list    # media array to cover 1.13777 cm
                      # shift is a multiple of six pixels

STEADY_STATE = 0       # if 1, go to steady state solution
Scale = 4              # scale reduction of smaller arrays
Units4 = Scale*Units    # units for course arrays

Width = Units*Nlinex list

c
c Define array numbers
c
Media = 1
MediaWork = 2
Thermal_small = 3
```

Jump to: [Commands](#), [Theory](#)

```

array/s Media Nlinex/Scale Nliney/Scale data

array/s MediaWork Nlinex/Scale Nliney/Scale data

array/s Thermal_small Nlinex/Scale Nliney/Scale data

units/set Media Units          # set units of media and temporary
units/set MediaWork Scale*Units
units/set Thermal_small Scale*Units

c
c Width of gain region
c

Xwide=10; Ywide=2

variable/dec/int ntimes plot

mac/def therm/o
  c
  c macro for thermal settling
  c
  therm/set Thermal_small time nstep=Nstep
  total_time = total_time + time
  title temperature distribution, time = @total_time
  plot = plot + 1
  plot/watch plot2.plt
  plot/l/r Thermal_small ns=128
  point/list Thermal_small
  variab/set Peak point/sr list
  plot/w plot3.plt
  plot/y/r Thermal_small
mac/end

Max = 1.6765e-3          # peak pump energy density
Absorption = .004        # side pump absorption [cm-1]
PeakInt = Max/Absorption # pumping intensity to achieve energy density
Width = .2
Shift = .5

c
c Make a thermal surface with BK7 glass and forced air boundary
c
clear Thermal_small 30          # set temperature of slab
clap/rec/c Thermal_small Xwide Ywide # rectangular aperture
thermal/mat/add Thermal_small BK7    # establish a thermal array for beam
clear MediaWork 30
thermal/mat/add Thermal_small MediaWork forced_air

c
c side-pumping with five gaussian beams
c to put population inversion into gain sheets
c and put a heat source into thermal arrays

```

Jump to: [Commands](#), [Theory](#)

```

c
clear MediaWork 0.
slab/pump/gaus/left   kout=MediaWork   peak=PeakInt   w0=Width   lambda=Lambda
abs=Absorption m2=2. dec=2*Shift refl=.0
slab/pump/gaus/left   kout=MediaWork   peak=PeakInt   w0=Width   lambda=Lambda
abs=Absorption m2=2. dec=1*Shift refl=.0
slab/pump/gaus/left   kout=MediaWork   peak=PeakInt   w0=Width   lambda=Lambda
abs=Absorption m2=2. dec=0*Shift refl=.0
slab/pump/gaus/left   kout=MediaWork   peak=PeakInt   w0=Width   lambda=Lambda
abs=Absorption m2=2. dec=-1*Shift refl=.0
slab/pump/gaus/left   kout=MediaWork   peak=PeakInt   w0=Width   lambda=Lambda
abs=Absorption m2=2. dec=-2*Shift refl=.0

plot/w plot1.plt
title pumping distribution
plot/l MediaWork ns=128

intensity MediaWork

outfile/intensity/rec 'slab_pump.dat'/no/binary MediaWork

thermal/mat/source/square Thermal_small MediaWork 800 # store 10% of inversion
as heat

if STEADY_STATE then
  c
  c Calculate the time constant of the refractive surface.
  c
  RhoC_BK7 = 2.153
  Kappa_BK7 = .011
  T_BK7 = RhoC_BK7*Units1^2/Kappa_BK7 list
  c
  c Set variable to GLAD calculation of time constant of refractive surface.
  c
  variable/set TimeConstant 1 1 thermal/xtime list
  Velocity = Units4/TimeConstant list
  SettleTime = Xwide*TimeConstant/Units4 list
  time = 10*SettleTime
  Nstep = 800
  macro/run therm/10
else
  Nstep = 0
  time = 10
  macro/run therm
endif

c
c Copy temperature from Thermal_small to MediaWork
c using copy/reorder to copy only the real word.
c
clear MediaWork 0.
copy/reorder Thermal_small MediaWork r1000
plot/w plot1.plt

```

Jump to: [Commands](#), [Theory](#)

```

plot/1/r Thermal_small min=30 ns=64
plot/w plot2.plt
plot/1/r MediaWork min=30 ns=64
field MediaWork

amplitude/sqrt MediaWork
plot/w plot3.plt
plot/1/i MediaWork min=30 ns=64

intensity MediaWork

c
c Output temperature distribution
c
outfile/intensity/rec 'temperature.dat'/no/binary MediaWork

```

Ex120b: Multipass amplifier, propagation

Once the gain sheets and thermal arrays are calculated the arrays are imported into the multipass calculation in Ex120b.inp. Small patches are extracted from the full size gain arrays and the full size thermal distortion arrays and interact with the optical beam.

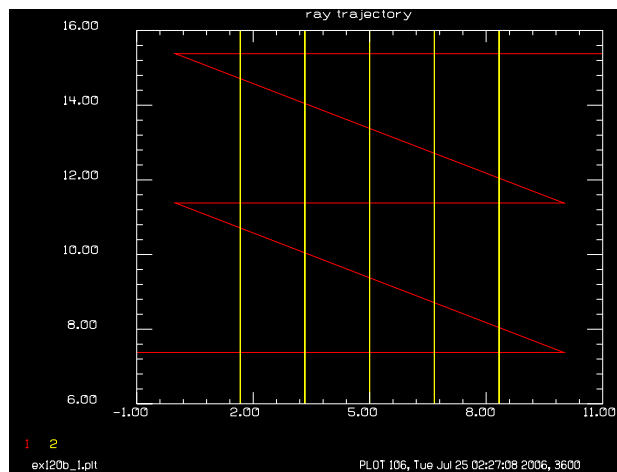


Fig. 120.3a. Path of beam in zig-zag amplifier, similar to Fig. 120.1.

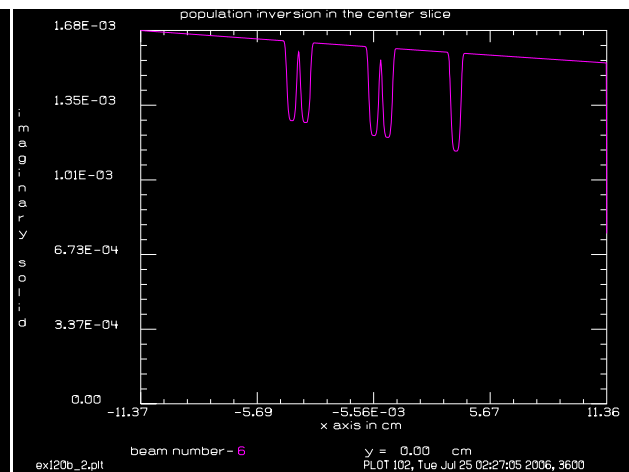


Fig. 120.3b. Holes burned into population inversion at sheet 5 at the end of the zig-zag path. Note that the burned holes match the crossing of the red line with the fifth yellow bar in Fig. 120.3a.

Input: ex120b.inp

```

c## ex120b
c
c Multipass amplifier, ray trajectories
c
c In Example 120b, the beam makes five passes through
c the gain medium. It is assumed that there are four
c mirrors to make the four reflections. The mirrors line along
c the x-axis at z = 0 and z = Length. They are rotated by

```

Jump to: [Commands](#), [Theory](#)

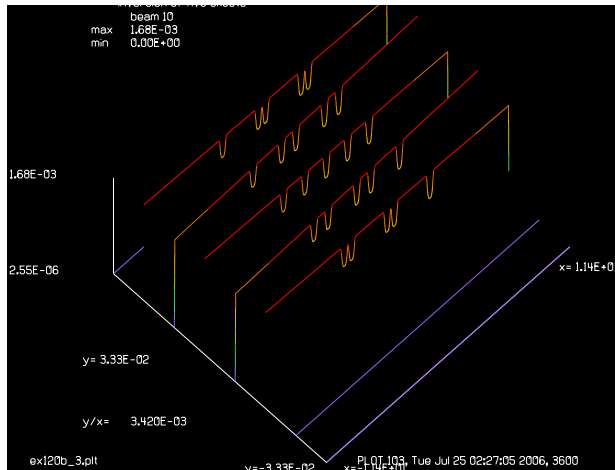


Fig. 120.4a. Five sheets of population inversion showing burns due to depletion by the optical beam. The five sheets match Fig. 120.3a.

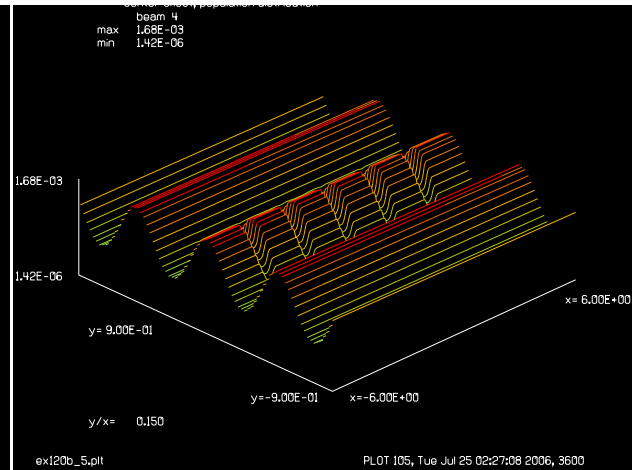


Fig. 120.4b. Burn patterns at Sheet 3, showing equal spacing of burned holes.

```

c "Angle" so that the beam parallel to the z-axis for passes
c 1, 3, and 5 and at angle 2*Angle for passes 2 and 4.
c
c The population inversion and temperature distribution should be
c setup previously
c
c Five gain sheets and corresponding temperature distributions are
c used alternating from left-to-right and right-to-left.
c
c In ex120a, we make sure that beams are in the correct place
c by saving the distribution at the middle point.
c
c Assume 10 cm long, 2 cm separation
c
alias Name ex120b
variable/dec/int Nlinex Nliney Icent MediaTemp MediaTemp1 MediaWork
variable/dec/int Media Media1 Media2 Media3 Media4 Media5 OpticalBeam Pass
variable/dec/int History Row
variable/dec/int Aberration1 Aberration2 Aberration3 Aberration4 Aberration5
variable/dec/int Sheet GainSheetNo AberrationSheetNo Scale
variable/dec/int MediaWork1 Thermal_large

Epsilon = 1e-3          # small number for rounding up
Nlinex = 2048           # horizontal pixels
Nliney = 512            # vertical pixels
Lambda = 1.064e-4       # set wavelength
Shift = 4.0             # shift of the beam at the end of each pass
Units = 2/6/30 list     # media array to cover 1.13777 cm
                        # Shift is a multiple of six pixels
Scale = 4               # scale between setup arrays and full size arrays

Xwide=10; Ywide=2       # width of pumped region

Width = Units*Nlinex list

```

Jump to: [Commands](#), [Theory](#)

```

Length = 10          # length between mirrors

OpticalBeam = 1      # optical beam that propagates through the amplifier

Media1 = 2           # five gain sheets
Media2 = 3
Media3 = 4
Media4 = 5
Media5 = 6
MediaTemp = 7        # full size temporary array, polarized
MediaTemp1 = 8       # full size temporary array
MediaWork = 9        # full size temporary array
History = 10         # show five gain sheets as they are processed

Aberration1 = 11      # five thermal aberration sheets
Aberration2 = 12
Aberration3 = 13
Aberration4 = 14
Aberration5 = 15
Thermal_large = 16   # full size temporary array for thermal effects
MediaWork1 = 17      # small size temporary array for thermal effects

Nsheets = 5          # number of gain sheets

array/s OpticalBeam Nliney Nliney          # set propagating array
units/set OpticalBeam Units                # set units for Beam 1

array/s Media1:Media5 Nlinex Nliney 1 data  # define media array
clear Media1:Media5 0                      # clear media array

array/s History Nlinex 8 1 data             # history array
clear History 0

array/s Aberration1:Aberration5 Nlinex Nliney data # define aberration arrays

array/s MediaTemp Nliney Nliney ipol=1 data # temporary media array

array/s MediaTemp1 Nliney Nliney data       # temporary square array

array/s MediaWork Nlinex Nliney data        # define temporary array

array/s Thermal_large Nlinex Nliney data

array/s MediaWork1 Nlinex/Scale Nliney/Scale data

units/set Media1:Media5 Units               # set units of media and temporary
units/set MediaTemp Units
units/set MediaTemp1 Units
units/set MediaWork Units
units/set Aberration1:Aberration5 Units
units/set Thermal_large Units
units/set MediaWork1 Scale*Units Scale*Units

Xcen = Nlinex*Units/2. list                 # x-center of media array from axis

```

Jump to: [Commands](#), [Theory](#)


```

wavelength/set 1 Lambda*1e4          # set wavelength
Apt = 0.3                             # gaussian radius
gaus/c OpticalBeam 2e7 Apt 2         # was 1e-7      # start for optical beam

Angle = -0.5*atan(Shift/Length)*180/pi list    # angle for crossing paths

Shift1 = tan(2*Angle*pi/180.)*Length list      # check

width = 1.2e10                         # set line width, hz
center = 2.83e14                       # set line center, hz
tspont = 1.9e-2                       # set spontaneous emission time, sec
t20 = 3e-1                             # set decay time for level 2, sec
t10 = 1e-5                             # set decay time for level 1, sec
mode_sep = 5e8                         # set longitudinal mode separation, hz
time = 2e-9

nstep = 4

c
c Set offset from line center of first mode, 0 hz
c Set fractional pumping into level 1, nlpump, 0
c
gain/rate/set width center tspond t20 t10 mode_sep 0 0
gain/rate/nonoise

gain/rate/list
pack/set OpticalBeam MediaTemp        # pack both beams

c clear MediaWork 1.e-5
Max = 1.6765e-3
c gaus/c MediaWork Max 10 6
Absorption = .004                     # side pump absorption [cm-1]
PeakInt = Max/Absorption
Width = .2
c
c Input population inversion from file into small array
c
infile/intensity/rec 'slab_pump.dat'/no/binary MediaWork1

c
c Copy to full size array and rescale
c
clear MediaWork
copy/con MediaWork1 MediaWork         # copy to full size array
plot/watch @Name_1.plt 100 0 400 300
title population inversion copied to full size array
plot/l MediaWork ns=128
units/set MediaWork Scale*Units
rescale/units MediaWork Units         # rescale
plot/w @Name_2.plt 500 0 400 300
title population inversion in full size array after rescale
plot/l MediaWork ns=128

```

Jump to: [Commands](#), [Theory](#)

```

c
c Setup five gain sheets. Alternate left-to-right and right-to-left
c
clear Media1:Media5 0.
gain/rate/n2level MediaWork Media1
gain/rate/n2level MediaWork Media3
gain/rate/n2level MediaWork Media5
flip/x MediaWork
gain/rate/n2level MediaWork Media2
gain/rate/n2level MediaWork Media4

Thick = Length/(Nsheets+1)                # thickness of thermal sheets

clear Thermal_large 30                    # set temperature of slab
thermal/mat/add Thermal_large thick=Thick BK7 # establish a thermal array for
beam
clear MediaWork 30
thermal/mat/add Thermal_large MediaWork forced_air
field Thermal_large
c
c Read in temperature
c
infile/intensity/rec 'temperature.dat'/no/binary MediaWork1
intensity MediaWork1
c
c Form full size temperature distribution
c
clear MediaWork 30 # preset to rest temperature
copy/con MediaWork1 MediaWork
plot/w @Name_3.plt 100 300 400 300
title temperature inversion in small size array
plot/l/i MediaWork min=30 ns=64

units/set MediaWork Scale*Units
rescale/units MediaWork Units
plot/w @Name_4.plt 500 300 400 300
title temperature inversion in full size array
plot/l/i MediaWork min=30 ns=64
intensity MediaWork
irradiance MediaWork
field MediaWork
c
c MediaWork now contains full size temperature distribution.
c Copy temperature into real part of thermal array that
c has already been built.
c
copy/reorder MediaWork Thermal_large r1000
field Thermal_large
c
c Setup aberration sheets 1, 3, and 5 for left to right pumping
c Setup aberration sheets 2 and 4 to be right to left
c
c aberration sheet 1
thermal/window Aberration1 0. Thermal_large

```

Jump to: [Commands](#), [Theory](#)

```

c aberration sheet 2
copy/con Aberration1 Aberration2
flip/x Aberration2                # flip to simulate reverse pumping
c aberration sheet 3
copy/con Aberration1 Aberration3
c aberration sheet 4
copy/con Aberration2 Aberration4
c aberration sheet 5
copy/con Aberration1 Aberration5

C end of setup
pause

macro/def step
  GainSheetNo = Sheet + Media1 - 1 list
  AberrationSheetNo = Sheet + Aberration1 - 1 list
  Position = (GainSheetNo-OpticalBeam)*Length/(Nsheets+1) list
  vertex/locate/abs 0 0 Position; prop/vertex
  variable/set Xray OpticalBeam xray list
  udata/set [Row=Row+1] Position Xray
  if [Pass==1] then
    c Draw vertical lines on udata plot
      udata/set [Row=Row+1] Position Xray 6
      udata/set [Row=Row+1] Position Xray 16
      udata/set [Row=Row+1] Position Xray 6
    endif
    Xcent = (Xcen-Xray)/Units list
    Icent = Xcent+Epsilon*Xcent/abs(Xcent) list # round to integer
    c ----- Extract gain patch
      copy/con GainSheetNo MediaTemp icent=Icent
      pack/in                      # pack beams
      gain/rate/step Length/Nsheets time nstep
      pack/out                     # unpack beams
    c Put MediaTemp back into gain sheet
      copy/con MediaTemp GainSheetNo icent=-Icent
      copy/row GainSheetNo History Nliney/2+1 GainSheetNo
    c ----- done with gain patch

    c ----- Extract aberration patch
      copy/con AberrationSheetNo MediaTemp1 icent=Icent
      mult/beam OpticalBeam MediaTemp1
    c ----- done with aberration patch

  plot/watch @Name_1.plt 100 0 400 300
  title ray trajectory
  plot/udata first=1 last=2 min=6 max=16 left=-1 right=11
  plot/w @Name_2.plt 500 0 400 300
  title population inversion in the center slice
  plot/x/a GainSheetNo fmin=0
  plot/w @Name_3.plt 100 300 400 300
  title inversion of five sheets
  variab/set Max History peak
  plot/l/a History theta=42 min=Max*.9

```

Jump to: [Commands](#), [Theory](#)

```

    gain/rate/inversion GainSheetNo MediaWork Length/Nsheets
    plot/w @Name_4.plt 500 300 400 300
    title Patch population distribution
    plot/l MediaWork ns=64 xrad=6 yrad=.9 min=.5*Max
C Z = @Position, X = @Xray
pause
macro/end

macro/def gain_forward
    Pass = Pass + 1 list
    vertex/rotate/set 0 0 0

    Sheet = 1
    macro step

    Sheet = 2
    macro step

    Sheet = 3
    macro step

    Sheet = 4
    macro step

    Sheet = 5
    macro step

macro/end

macro/def gain_backward
    Pass = Pass + 1 list
    vertex/rotate/set 0 0 0

    Sheet = 5
    macro step

    Sheet = 4
    macro step

    Sheet = 3
    macro step

    Sheet = 2
    macro step

    Sheet = 1
    macro step

macro/end

c
c Propagate optical beam through zig-zag path
c

```

Jump to: [Commands](#), [Theory](#)

```

Xcen=
Shift=
x = Xcen-Shift list

global/def OpticalBeam Xcen-Shift 0 -1
variab/set Zray OpticalBeam zray
variab/set Xray OpticalBeam xray
udata/set OpticalBeam Zray Xray

udata/set [Row=Row+1] Zray Xray

macro gain_forward

C Pass 1, to end

vertex/locate/abs Xcen-Shift 0 Length
vertex/rotate/set 0 Angle 0

Xray=

C Mirror 1

zreff
global OpticalBeam

mirror/global/flat OpticalBeam
variable/set Xray OpticalBeam xray list
variab/set Zray OpticalBeam zray

udata/set [Row=Row+1] Zray Xray

macro gain_backward

C Pass 2, to end

vertex/locate/abs Xcen 0 0
vertex/rotate/set 0 Angle+180 0

C Mirror 2

mirror/global/flat OpticalBeam
variable/set Xray OpticalBeam xray list
variab/set Zray OpticalBeam zray

udata/set [Row=Row+1] Zray Xray

macro gain_forward

C Pass 3, to end

vertex/locate/abs Xcen 0 Length
vertex/rotate/set 0 Angle 0

C Mirror 3

```

Jump to: [Commands](#), [Theory](#)

```
mirror/global/flat OpticalBeam
variable/set Xray OpticalBeam xray list
variab/set Zray OpticalBeam zray
udata/set [Row=Row+1] Zray Xray

macro gain_backward

C Pass 4, to end

vertex/locate/abs Xcen+Shift 0 0
vertex/rotate/set 0 Angle+180 0

C Mirror 4

mirror/global/flat OpticalBeam
variable/set Xray OpticalBeam xray list
variab/set Zray OpticalBeam zray
udata/set [Row=Row+1] Zray Xray

macro gain_forward

C Pass 5, to end

vertex/locate/abs Xcen+Shift 0 Length+1
vertex/rotate/set 0 0 0
prop/vertex

plot/watch @Name_5.plt 100 600 600 450
title Center sheet, population distribution
plot/1/a Media3 ns=64 xrad=6 yrad=.9 min=.5*Max

variab/set Zray OpticalBeam zray
udata/set [Row=Row+1] Zray Xray

plot/watch @Name_1.plt 100 0 400 300
title ray trajectory
plot/udata first=1 last=2 min=6 max=16 left=-1 right=11
```

Ex121: Zigzag amplifier

Table. 121.1. Table of Ex121 examples

Ex121a: Equal length mirror pair	1
Ex121b: Prism configuration	9
Ex121c: Interaction of strongly slanted beams	21

Examples of zigzag amplifiers are illustrated. The zigzag configuration allows the beam to make multiple passes through the amplifier and to average-out some of the effects of nonuniform gain. The optical beam overlaps at each sidewall reflection and sweeps out a complex zigzag path in the amplifier. The beam overlaps itself near the mirrors leading to higher inversion depletion in selected regions. A full 3D model of the gain is required to represent the complex gain region.

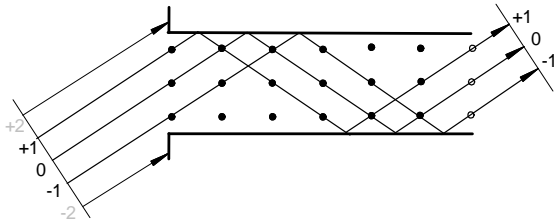


Fig. 121.1. Illustration of a simple two-mirror zigzag amplifier. The gain region is illustrated by a 3×6 grid of points. The three rays +1 to -1 will get through and the 0th ray is the center. Note that some points are not crossed in a single pass, leaving undepleted population inversion.

This configuration is illustrated in example 121a.

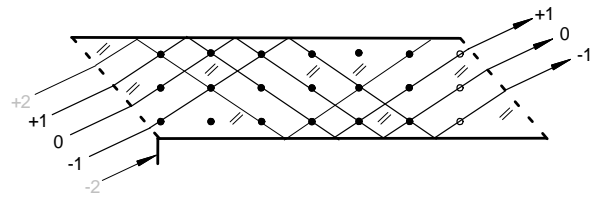


Fig. 121.2. A gain crystal may have prism faces cut on the ends to facilitate beam coupling. Ray -2 is clipped by the bottom edge of the gain region. As drawn, ray +2 can enter the gain region but can not exit without making additional reflections. This ray should be clipped off as it will deplete the gain without contributing to the desired output. Modeling of this configuration is illustrated in example 121b.

Ex121a: Equal length mirror pair

Fig. 121.3 illustrates the configuration for a pair of equal length mirrors.

Input: ex121a.inp

```
c## ex121a
c
c Example of zigzag amplifier
c
c The light will bounce from side-to-side between two
c mirrors and will be amplified by the gain medium
c that is between two mirrors.
c
c As the beam sweeps through the gain medium it depletes the
c medium such that the zigzag pattern of the light
c effectively burns a zigzag pattern of inversion depletion.
```

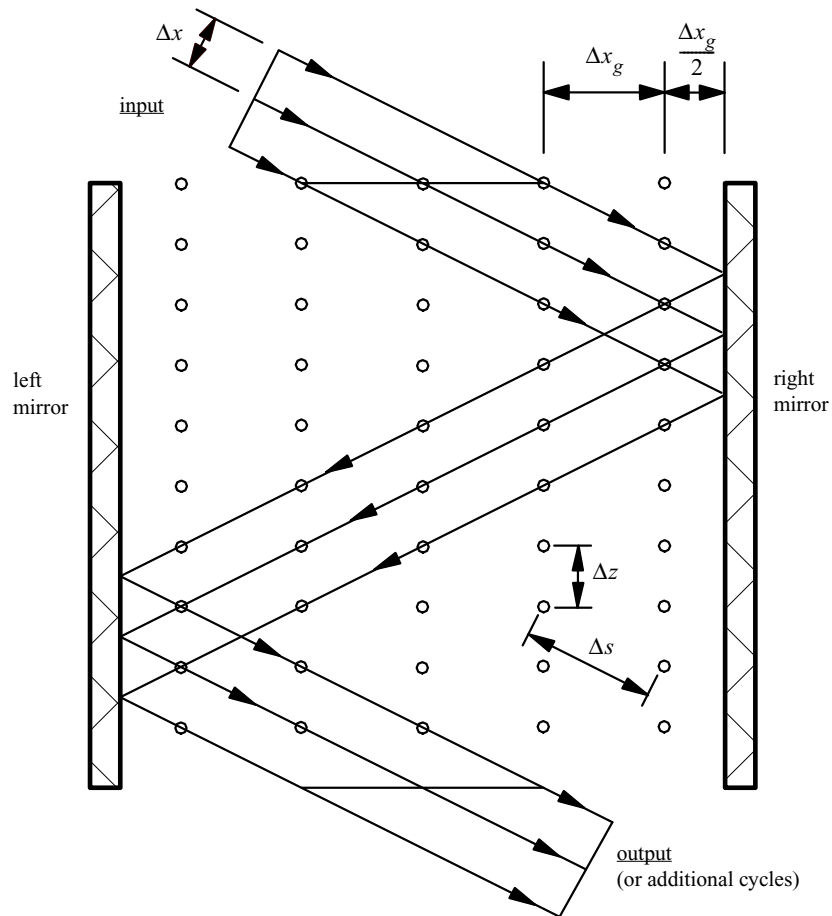


Fig. 121.3. Configuration for a zigzag amplifier showing one a pair of reflections forming one complete cycle. Only one cycle is shown but any number of cycles may be used. The beam overlaps upon reflection at each mirror, causing higher inversion depletion.

The pixels are matched to avoid interpolation and the associated errors. The optical beam, with units Δx , enters on the left at an angle α to the axis of the amplifier region. The transverse units for the amplifier should be $\Delta x_g = \Delta x / \cos(\alpha)$. The units of the amplifier along the axis should be $\Delta z = \Delta x_g / \tan(\alpha)$. The optical propagation along the slant line takes steps of $\Delta s = \Delta z / \cos(\alpha) = \Delta x_g / \sin(\alpha) = \Delta x / \sin(\alpha) \cos(\alpha)$. The gain points are organized into a 3D array.

In this elementary drawing, the aperture is shown with only five pixels. The mirrors are shifted a half pixel. The length for one complete cycle is ten—twice the number of aperture pixels.

c This complex pattern requires detailed treatment of the gain
c region.

c

c As the beam sweeps through the gain medium it depletes the
c gain medium such that the zigzag pattern of the light
c effectively burns a zigzag pattern of inversion depletion.

c This complex pattern requires detailed treatment of the gain
c region.

c

c In this example, the separation of the mirrors will be placed
c at 50 pixels such that there are a total of 101 pixels between
c the two mirror surfaces. For this example, ten bounces will be



Fig. 121.4. Zigzag amplifier consisting of two mirrors, shown in the gain coordinate system. A configuration of ten reflections is shown. The start and end of the center ray is indicated as well as the front and back of the zigzag amplifier. For this case, the ray enters the front.

Fig. 121.5. Zigzag amplifier consisting of two mirrors (same as Fig. 121.4), shown in the beam coordinate system.

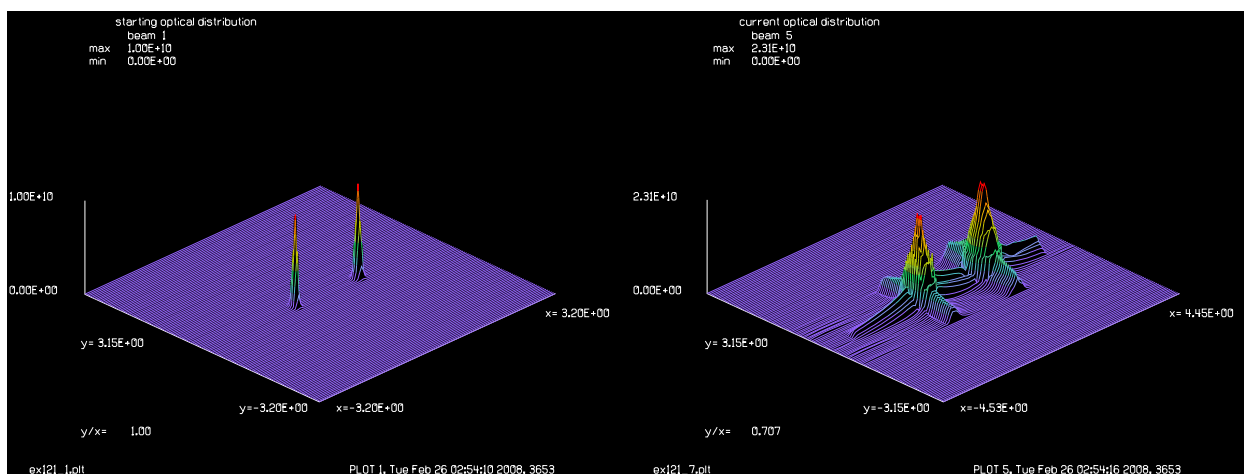


Fig. 121.6. Example of two gaussian beams as input to illustrate clearly the effects of folding and overlap.

Fig. 121.7. Pair of gaussian beams after five complete cycles.

c modeled. To avoid interpolation with the associated loss of
 c accuracy, the zigzag model many gain sheets will be used so that
 c each lateral shift of one pixel in the x-direction results in
 c movement to a new gain sheet. Crossing the 101 points of the
 c aperture requires 101 gain sheets. For ten reflections we
 c cross the aperture ten times and 1010 gain sheets are required.
 c As arrays in GLAD must be powers of 2, the z-pixels of the
 c 3D gain array must be at least 1024.

c
 c The optical field is described by an array of 256 points. The
 c The 3D gain array is 128 x 128 x 1024 points.

c
 alias Name ex121a
 mem/set/b 266 # request 266 MB so 3D array loads quickly

Jump to: [Commands](#), [Theory](#)

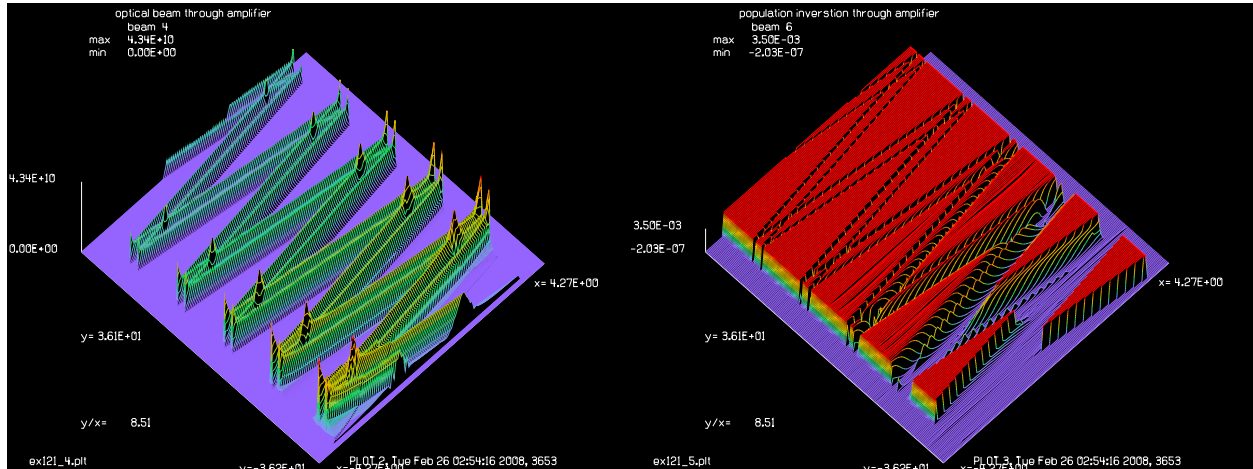


Fig. 121.8. Optical beam passing through the zigzag amplifier by entering from the top and heading first to the right mirror. The beam makes ten reflections and five full cycles. Note the high intensity in the overlap region that leads to higher inversion depletion.

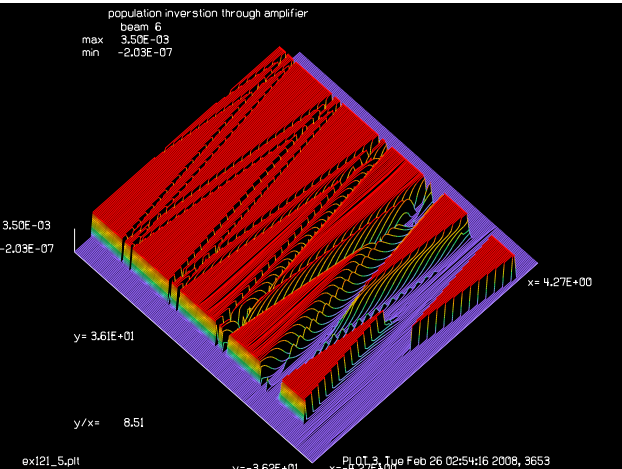


Fig. 121.9. Depletion of the population inversion occurs along the zigzag path and is higher in the overlap regions.

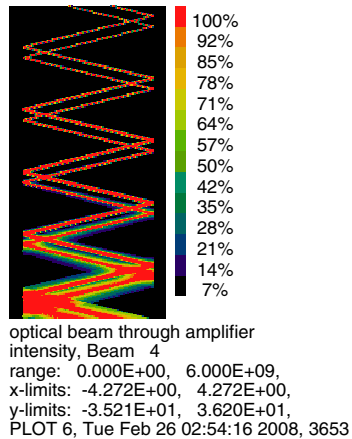


Fig. 121.10. False color map of the optical beam in ten-pass amplifier showing the overlap pattern of the double gaussian input.

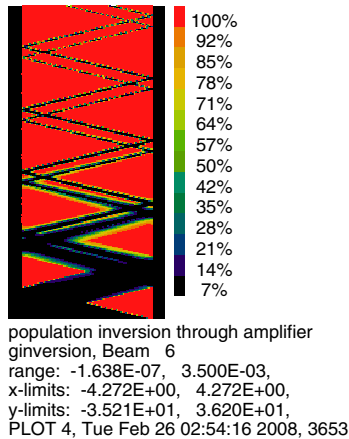


Fig. 121.11. False color map of the population inversion.

```
variable/dec/int Count Iteration Max TotalSteps Position
variable/dec/int X GainNline NcOptical NlineSteps NcGain
variable/dec/int OpticalBeam GainArray GainTemp
variable/dec/int OpticalHistory TempArray I
variable/dec/int Nz Naperture Mode
```

```
c Starting geometric parameters
c -----
```

```
Rad = pi/180 # convert degrees to radians
Reflections = 10 # beam will hit each wall five times
AxisToRayDeg = 45 # beam is 45 degrees from the gain axis
AxisToRayRad = Rad*AxisToRayDeg # angle in radians
```

Jump to: [Commands](#), [Theory](#)

```

Magnification = 1./cos(AxisToRayRad) list # Magnification of gain units
                                         # from optical units
UnitsGainX = 0.05*Magnification          # set gain units
UnitsX = UnitsGainX/Magnification # X-units of the gain region
UnitsY = UnitsX list

Mode = 1                                # mode 1, 2, 3, or 4
HalfWidth = 3.56                        # Half-width of mirror separation
GaussWidth = .1

UnitsZ = UnitsGainX/tan(AxisToRayRad)    # axial spacing along gain axis
c
c Find number of across the aperture
c
Naperture = 2*floor(HalfWidth/UnitsGainX)+1 # Number of points across aperture
c
c Get the number of z-pixels for even number of reflections
c
Nz = Reflections*Naperture list
Length = UnitsZ*Nz list
PlotWidth = 1.2*HalfWidth
HalfHeight = 1

c
c Array numbers
c -----
OpticalBeam = 1          # propagating optical beam
GainArray = 2            # 3D gain array
GainTemp = 3             # temporary data array
OpticalHistory = 4       # history of optical beam formed by zigzag routine
TempArray = 5            # temporary array for optical folded image
GainHistory = 6          # history of population inversion formed by zigzag
c
c Array sizes
c -----
OpticalNline = 256
GainNline = 128
NlineY = 128
NlineSteps = 1024
c
c Array centers
c
NcOptical = OpticalNline/2 + 1
NcGain = GainNline/2 + 1
NcHistory = NlineSteps/2 + 1
X = 20
c
c Initialize arrays
c -----
array/s OpticalBeam OpticalNline NlineY ipol=0
array/set/3d GainArray GainNline NlineY NlineSteps ipol=1 data
nbeam GainTemp GainNline NlineY data
nbeam OpticalHistory GainNline NlineSteps ipol=0 data
nbeam TempArray GainNline NlineY data

```

Jump to: [Commands](#), [Theory](#)

```

nbeam GainHistory GainNline NlineSteps ipol=1 data

c
c Constants
c -----
Lambda = 1e-4
PumpRate = 1.93e6           # pump rate in watts per cm**3
N2=3.5e-3
Time = 1e-11

wavelength/set 0 Lambda*1e4 1      # set wavelength and index

units/set OpticalBeam UnitsX
units/set GainArray UnitsGainX UnitsY UnitsZ
units/set GainTemp UnitsGainX UnitsY
units/set OpticalHistory UnitsGainX UnitsZ
units/set GainHistory UnitsGainX UnitsZ

c
c Form optical beam
c
units/set TempArray UnitsX
gaus/cir OpticalBeam 1 GaussWidth decx=X*UnitsX
gaus/cir TempArray 1 GaussWidth decx=-X*UnitsX
add/inc/con OpticalBeam TempArray
energy/norm OpticalBeam pi*1e8
geodata/set waistx=.1      # set waistx specifically to control Rayleigh range
plot/w @Name_1.plt
title starting optical distribution
plot/1 OpticalBeam xrad=GainNline/2*UnitsX ns=256
units/set TempArray UnitsGainX UnitsX
UnitsX=
UnitsGainX=

c Parameters for zigzag
c -----
c HalfWidth,           Half separation of reflecting walls
c HalfHeight,          Half height of zigzag walls, beam will be clipped
c Length,              Length of zigzag amplifier
c Reflection,           Number of reflections
c Time,                Time between passes through the gain region

zigzag/set OpticalBeam GainArray HalfWidth HalfHeight &
AxisToRayDeg Reflections &
Time mode=Mode OpticalHistory GainHistory

zigzag/noise/off      # no spontaneous emission
zigzag/list
zigzag/list/global
plot/w @Name_2.plt
plot/zigzag/gain
plot/w @Name_3.plt
plot/zigzag/beam

```

Jump to: [Commands](#), [Theory](#)

```

variable/set TotalSteps zigzag/totalsteps list
variable/set StepLength zigzag/step_straight list

C compare precalculated angle with value from zigzag
variable/set AngleCalc zigzag/angle list
AxisToRayDeg=

units
set/density GainNline min(NlineSteps,256)
PlotTop = NlineSteps/2*StepLength list
PlotBottom = PlotTop -TotalSteps*StepLength list
set/window/abs -PlotWidth PlotWidth PlotBottom PlotTop

width = 1.45e13          # Set line width, hz
center = c/Lambda list   # Set line center, hz
tspont = 1e-6            # Set spontaneous emission time, sec
t20 = 3e-1               # Set decay time for level 2, sec
t10 = 1e-11             # Set decay time for level 1, sec

gain/rate/set width center tspond t20 t10

c
c Set initial N2 population in GainTemp
c
clear GainTemp N2
set/sect 2 1
clear GainArray 0
gain/rate/n2level GainTemp GainArray
c
c copy inversion in section 1 to all sections
c
copy/section GainArray GainArray 1 0

clear OpticalHistory 0
clear GainHistory 0
Count = 0

time/i
if 0 then
  c take this path to display step-by-step
  macro step/1 # TotalSteps
else
  c take this path to display only end results
  zigzag/step TotalSteps
  write/on
endif
time

plot/w @Name_6.plt
title optical beam through amplifier
plot/1 OpticalHistory xrad=PlotWidth yrad=(NcHistory+1)*StepLength ns=512 h=.3
th=42

```

Jump to: [Commands](#), [Theory](#)

```

plot/w @Name_7.plt
title population inversion through amplifier
plot/l/ginversion GainHistory xrad=PlotWidth yrad=(NcHistory+1)*StepLength
ns=256 h=.1 th=42
variable/set Position zigzag/position list

plot/w @Name_8.plt
title population inversion through amplifier
plot/b/g GainHistory

plot/w @Name_9.plt
title optical beam through amplifier
plot/b/i OpticalHistory max=.6e10

copy/con OpticalBeam TempArray
plot/w @Name_10.plt
title current optical distribution
plot/l TempArray ns=GainNline

C Name = @Name
energy/list OpticalBeam
end

c =====Macros=====

macro/def step
  Count = Count + 1
  zigzag/step 1
  macro/run Diagnostics
macro/end

macro/def Diagnostics
  plot/w @Name_4.plt
  title medium field N2, Count = @Count
  plot/x/a GainArray left=-PlotWidth right=PlotWidth fmin=0

  c
  c find folded optical pattern by zigzag/image
  c

  zigzag/image TempArray # form optical image at current step
  variable/set Iteration 1 macro/iteration
  variable/set Max 1 macro/maxiteration
  plot/w @Name_5.plt
  title Iteration = @Iteration out of @Max iterations
  plot/x/i TempArray left=-PlotWidth right=PlotWidth # fmax=PeakIntensity*3
  c
  c Plot current optical history array
  c
  plot/w @Name_6.plt
  title optical beam through amplifier
  plot/l OpticalHistory xrad=PlotWidth yrad=(NcHistory+1)*StepLength ns=512
  h=.3 th=42

```

Jump to: [Commands](#), [Theory](#)

```

variable/set Position zigzag/position
variable/set GainSection GainArray section
c
c Plot current gain inversion array
c
plot/w @Name_7.plt
title population inverstion through amplifier
plot/l/ginversion GainHistory xrad=PlotWidth yrad=(NcHistory+1)*StepLength
ns=256 h=.1 th=42
c
c Plot bitmap of gain inversion array
c
plot/w @Name_8.plt
title population inversion through amplifier
plot/b/g GainHistory

plot/w @Name_9.plt
title current optical distribution
plot/l TempArray ns=GainNline
macro/end

```

Ex121b: Prism configuration

A more efficient system may be created by shifting the mirrors to form the configuration shown in Fig. 121.2. Often a prism will be used such that glass or crystal fills the space between the mirrors as indicated in Fig. 121.12.

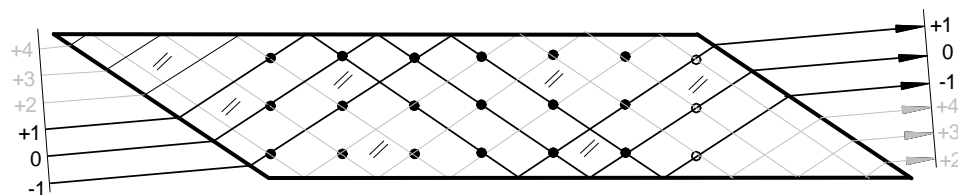


Fig. 121.12. A prism configuration for the gain region with refraction of the incident light. It is assumed that the gain occurs only in the straight wall region as indicated by the solid dots. For this elementary case, there are six gain sheets—three transverse gain points times the two reflections. The three rays -1 to $+1$ make two bounces as required. Rays $+2$ to $+4$ take four bounces and will experience double the propagation length in the glass and will, in general, result in phase mismatch with respect to rays -1 to $+1$ as well as mispositioning. Generally, rays $+2$ to $+4$ should be blocked.

The external angle and sample spacing is defined by the Snell's Law relationship illustrated in Fig. 121.13. The angle of the internal ray to both the amplifier axis and the parallel walls is face angle is α . The angle of the incident, external ray to the gain axis is α_0 . The angle of the prism face to the axis is β . The incident ray angle to the normal of the prism face is I_0 :

$$I_0 = 90^\circ - \alpha_0 - \beta \quad (121.1)$$

after refraction the internal ray normal angle is

Jump to: [Commands](#), [Theory](#)

$$I = 90^\circ - \alpha - \beta \quad (121.2)$$

The incident and refracted angles are related by Snell's Law:

$$N_0 \sin I_0 = N_g \sin I \quad (121.3)$$

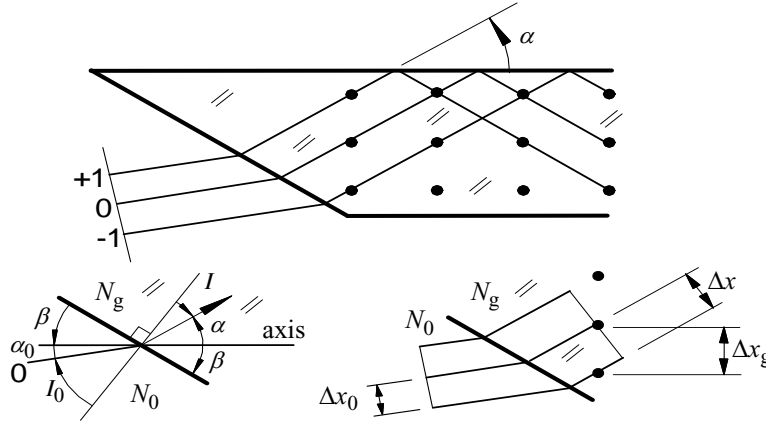


Fig. 121.13. The ray to gain axis angle is α_0 externally and α internally. The internal incident angle I with respect to the entrance face normal is $I = 90^\circ - \alpha - \beta$. The entrance angle to the entrance face is found from Snell's Law $I_0 \sin N_0 = I \sin N_g$ and with the incident angle $I_0 = 90^\circ - \alpha_0 - \beta$ and the exiting angle $I = 90^\circ - \alpha - \beta$, we have the relation $\Delta x_g = \Delta x / \cos(\alpha)$, where Δx_g is the sample spacing in the gain region and Δx is the sample spacing normal to the ray direction. $\Delta x = (\cos I) / (\cos I_0) \Delta x_0$ and $\Delta x_g = (\cos I) / (\cos I_0 \cos \alpha) \Delta x_0$. The units of the gain region Δx_g are calculated from the initial beam units Δx_0 and the internal angle α .

Given the desired incident wall angle α , the necessary orientation angle α_0 of the gain axis to the incident ray is:

$$\alpha_0 = \alpha + (I - I_0) \quad (121.4)$$

The zigzag amplifier may have either an even or odd number of reflections. Fig. 121.14 illustrates the single-reflection case. Similar to the double reflection case, there is a path for spurious three-reflection rays to pass through. Higher odd-number reflections add N gain sheets for each additional reflection, where N is the number of transverse gain sample points..

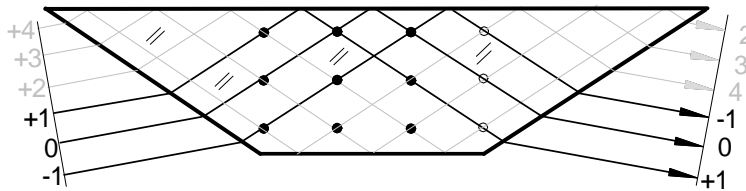


Fig. 121.14. Zigzag amplifier prism using one reflection. Rays -1 to $+1$ pass through with one bounce. The aperture is inverted. Rays $+2$ to $+4$ pass through the prism but have three reflections, so these rays are not, in general, spatially coherent with the single reflection rays. Since these spurious rays deplete the gain region, they should be clipped.

The distance L_1 from the prism face to the active region of gain is shown in 121.15. Consider the ray angle with respect to the prism axis to be α , the angle of the prism face to the axis to be β , and the distance from the pivot point P to the ray intercept to be h . The length L_1 is found from the equation.

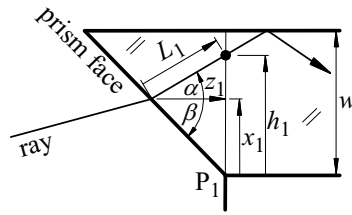


Fig. 121.15. The slant length L_1 from the prism face intercept to the start of the active gain region depends on the pivot point P_1 of the face, the height h_1 from the pivot point, the ray angle α , and the angle of the prism face β . The distance from the intercept point along the gain axis to the start of the active gain region is z_1 and the perpendicular distance from the intercept point to P_1 is x_1 . The width of the gain region is w .

$$L_1 = \frac{h_1}{\cos \alpha (\tan \alpha + \tan \beta)} \quad (121.5)$$

The net length through the prism for the center ray determines the global coordinate transformation. Consider Fig. 121.15 showing a two-reflection (even) system ($N_r = 2$) and Fig. 121.17 a one-reflection (odd) system ($N_r = 1$).

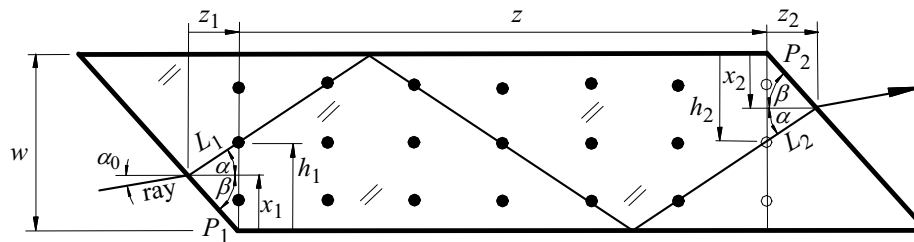


Fig. 121.16. A two-reflection system with $N = 3$ width and incident ray to axis angle of α_0 . Measuring from pivot point P_1 the height of the chief ray at the start of the gain region is $h_1 = N\Delta g/2$. Measuring from the second pivot point P_2 downward, the height of the exiting chief ray is $h_2 = h_1$.

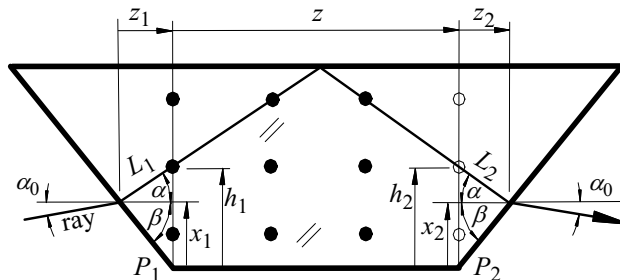


Fig. 121.17. A one-bounce system with $N = 3$ width. Measuring from the pivot point P_1 , the height of the chief ray at the start of the gain region is $h_1 = N\Delta g/2$. Measuring from the second pivot point P_2 , the height of the exiting chief ray in pixels is $h_2 = h_1$.

For the center ray, the distance traveled by the chief ray in both prism ends is

$$L_1 = L_2 = \frac{h_1}{\cos \alpha (\tan \alpha + \tan \beta)} = \frac{N\Delta g/2}{\cos \alpha (\tan \alpha + \tan \beta)} \quad (121.6)$$

The length of all rays through the rectangular gain region exclusive of the angled prism faces is L :

$$L = N_r N \Delta s \quad (121.7)$$

where N_r is the number of reflections. The total length along the ray including the first prism face, the slant path through the gain region, and the second prism face yields

$$L_{\text{tot}} = \frac{N\Delta g}{\cos \alpha (\tan \alpha + \tan \beta)} + (N_r - 1)N\Delta s \quad (121.8)$$

The distance between pivot points P_1 and P_2 is z .

$$z = N_r N \Delta z, \quad (121.9)$$

The height of the ray intercept at the start of the gain region is h_1 :

$$h_1 = \frac{N}{2} \Delta x_g. \quad (121.10)$$

The projections of the prism path onto the gain axis are z_1 and z_2 :

$$z_1 = z_2 = \frac{h_1}{\tan \alpha + \tan \beta} = \frac{N\Delta x_g/2}{\tan \alpha + \tan \beta}, \quad x_1 = \frac{h_1 \tan \beta}{\tan \alpha + \tan \beta} \quad (121.11)$$

$$z_{\text{tot}} = z_1 + z + z_2. \quad (121.12)$$

The net change in height (in the gain coordinate system) is

$$x_{\text{tot}} = \begin{cases} w - 2x_1 & N_r \text{ even} \\ 0 & N_r \text{ odd} \end{cases}. \quad (121.13)$$

The net change in rotational angle is

$$\text{net angle} = \begin{cases} 0 & N_r \text{ even} \\ 2\alpha_0 & N_r \text{ odd} \end{cases}. \quad (121.14)$$

In Ex121b prism ends have been added to the same case as Ex121a. See Fig. 121.18..

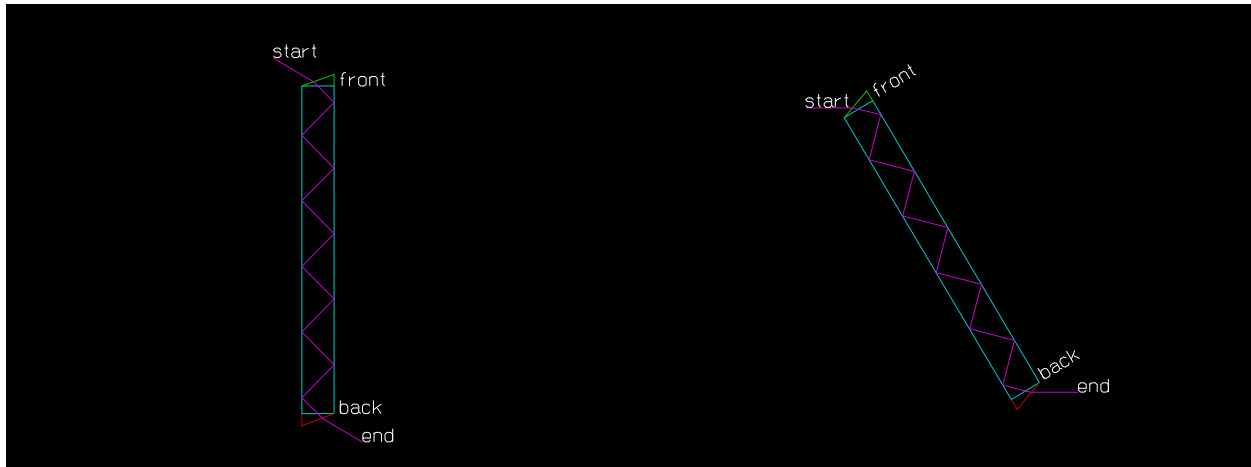


Fig. 121.18. Zigzag amplifier consisting of a prism with two tilted ends, shown in the gain coordinate system. the ray is refracted at both faces. The entrance face is drawn in green and the exit face is drawn in red. A configuration of ten reflections is shown. The start and end of the center ray is indicated as well as the front and back of the zigzag amplifier. For this case, the ray enters the front.

Fig. 121.19. Zigzag amplifier consisting of a prism with two tilted ends (same as Fig. 121.18), shown in the beam coordinate system.

Input: ex121b.inp

```
c## ex121b
c
c Example of zigzag amplifier in a prism
c
c The light will bounce from side-to-side between the two
c walls of the prism and will be amplified by the gain medium
c that is between the prism walls.
c
c The overlap of intensity that occurs at the wall reflection
c is modeled. It is assumed that the angle of incidence on the
c walls is sufficient that self-interference fringes are so
c fine that the effect may be reasonably modeled as incoherent
c addition with the interference fringes ignored.
c
c As the beam sweeps through the gain region it depletes the
c medium such that the zigzag pattern of the light
c effectively burns a zigzag pattern of inversion depletion.
c This complex pattern requires detailed treatment of the gain
c region.
c
c In this example, the separation of the prism walls will be placed
c at 50 pixels such that there are a total of 101 pixels between
c the two walls. For this example, ten bounces will be
c modeled. To avoid interpolation with the associated loss of
c accuracy, the zigzag model many gain sheets will be used so that
c each lateral shift of one pixel in the x-direction results in
```

Jump to: [Commands](#), [Theory](#)

```

c movement to a new gain sheet. Crossing the 101 points of the
c aperture requires 101 gain sheets. For ten reflections we
c cross the aperture ten times, 1010 gain sheets are required.
c As arrays in GLAD must be powers of 2, the z-pixels of the
c 3D gain array must be at least 1024.
c
c The optical field is described by an array of 256 points. The
c The 3D gain array is 128 x 128 x 1024 points.
c
alias Name ex121b
mem/set/b 266 # request 266 MB so 3D array loads quickly

variable/dec/int Count Iteration Max TotalSteps Position
variable/dec/int X GainNline NcOptical NlineSteps NcGain
variable/dec/int OpticalBeam GainArray GainTemp
variable/dec/int OpticalHistory GainHistory TempArray I
variable/dec/int Nz Naperture Prism GainSection HalfWidthPixels Even Mode
variable/dec/int CHECK

c To match the units of Ex121a, the optical units in the prism are set
c the same at 0.05 and the starting units in air are calculated as are
c the units in the gain region

c Starting geometric parameters
c -----
UnitsPrismX = .05 # set x-units in prism to match ex121a
UnitsY = UnitsPrismX
Rad = pi/180 # convert degrees to radians
Reflections = 10 # number of times beam will hit the walls
AxisToRayDeg = 45 # beam angle from the gain axis
AxisToFaceDeg = 70 # face angle from the gain axis

AxisToRayRad = Rad*AxisToRayDeg

Prism = 1
Mode = 4 # mode 1, 2, 3, or 4
GainIndex = 1.

HalfWidth = 3.56 # Half-width of mirror separation
GaussWidth = .1

c calculate UnitsPrismX and UnitsGainX in macro "Calc"
macro/run Calc

HalfHeight = 1

c The gain axis is identified as V
UnitsV = UnitsGainX/tan(AxisToRayRad) # axial spacing along gain axis
c
c Find number of across the aperture
c
Naperture = 2*floor(HalfWidth/UnitsGainX)+1 # Number of points across aperture
c
Nz = Reflections*Naperture list

```

Jump to: [Commands](#), [Theory](#)

```

c Length along gain axis
GainLengthV = UnitsV*Nz list

PlotWidth = 1.5*HalfWidth

c
c Array numbers
c -----
OpticalBeam = 1      # propagating optical beam
GainArray = 2      # 3D gain array
GainTemp = 3        # temporary data array
OpticalHistory = 4   # history of optical beam formed by zigzag routine
TempArray = 5        # temporary array for optical folded image
GainHistory = 6      # history of population inversion formed by zigzag
c
c Array sizes
c -----
OpticalNline = 256
GainNline = 128
NlineY = 128
NlineSteps = 1024
c
c Array centers
c
NcOptical = OpticalNline/2 + 1
NcGain = GainNline/2 + 1
NcHistory = NlineSteps/2 + 1
X = 20
c
c Initialize arrays
c -----
array/s OpticalBeam OpticalNline NlineY ipol=0
array/set/3d GainArray GainNline NlineY NlineSteps ipol=1 data
nbeam GainTemp GainNline NlineY data
nbeam OpticalHistory GainNline NlineSteps ipol=0 data
nbeam TempArray GainNline NlineY data
nbeam GainHistory GainNline NlineSteps ipol=1 data

global/def 1 0 0 0 0 0 0

c
c Constants
c -----
Lambda = 1e-4
PumpRate = 1.93e6          # pump rate in watts per cm**3
N2=3.5e-3
Time = 1e-11
PeakIntensity = 1e10       # saturates fully

wavelength/set 0 Lambda*1e4 1      # set wavelength and index
c clear OpticalBeam 1e-4          # set initial irradiance

units/set OpticalBeam UnitsStartX UnitsY
units/beam TempArray OpticalBeam    # set TempArray initially to optical beam

```

Jump to: [Commands](#), [Theory](#)

```

units/set GainArray UnitsGainX UnitsY UnitsV
units/set GainTemp UnitsGainX UnitsY
units/set OpticalHistory UnitsGainX UnitsV
units/set GainHistory UnitsGainX UnitsV

c
c Form optical beam
c
gaus/cir OpticalBeam PeakIntensity GaussWidth decx=X*UnitsStartX
gaus/cir TempArray PeakIntensity GaussWidth decx=-X*UnitsStartX
add/inc/con OpticalBeam TempArray
energy/norm OpticalBeam pi*1e8
energy/list OpticalBeam
geodata/set waistx=.1 # set waistx specifically to control Rayleigh range
plot/w @Name_1.plt
title starting optical distribution
Xrad = GainNline/2*UnitsPrismX
plot/l OpticalBeam xrad=Xrad ns=256
units/set TempArray UnitsGainX UnitsY
UnitsStartX=
UnitsGainX=

c Parameters for zigzag
c -----
c HalfWidth,      Half separation of reflecting walls
c HalfHeight,     Half height of zigzag walls, beam will be clipped
c Length,         Length of zigzag amplifier
c Reflection,     Number of reflections
c Time,           Time between passes through the gain region

HalfWidth=
UnitsGainX=
HalfWidthPixels = HalfWidth/UnitsGainX list
Naperture=
GainWidth=(HalfWidthPixels*2 + 1)*UnitsGainX list
AxisToRayDeg=
InterceptGainHeight = GainWidth/2 list
UnitsSlant = UnitsV/cosa list

macro/run Check

zigzag/set OpticalBeam GainArray HalfWidth HalfHeight &
AxisToRayDeg Reflections &
Time mode=Mode OpticalHistory GainHistory &
prism=Prism index=GainIndex face=AxisToFaceDeg

zigzag/noise/off # no spontaneous emission
c html/start
zigzag/list
zigzag/list/global
plot/w @Name_2.plt
plot/zigzag/gain
plot/w @Name_3.plt

```

Jump to: [Commands](#), [Theory](#)

```
plot/zigzag/beam
```

```
variable/set TotalSteps zigzag/totalsteps list
variable/set StepLength zigzag/step_straight list
```

```
C compare precalculated angle with value from zigzag
variable/set AngleCalc zigzag/angle list
AxisToRayDeg=
```

```
units
set/density GainNline min(NlineSteps,256)
PlotTop = NlineSteps/2*StepLength list
PlotBottom = PlotTop -TotalSteps*StepLength list
set/window/abs -PlotWidth PlotWidth PlotBottom PlotTop
```

```
width = 1.45e13          # Set line width, hz
center = c/Lambda list   # Set line center, hz
tspont = 1e-6            # Set spontaneous emission rate, sec
t20 = 3e-1               # Set decay rate for level 2, sec
t10 = 1e-11             # Set decay rate for level 1, sec
```

```
gain/rate/set width center tspond t20 t10
```

```
c
c Set initial N2 population in GainTemp
c
clear GainTemp N2
set/sect 2 1
clear GainArray 0
gain/rate/n2level GainTemp GainArray
c
c copy inversion in section 1 to all sections
c
copy/section GainArray GainArray 1 0
```

```
clear OpticalHistory 0
clear GainHistory 0
Count = 0
time/i
if 0 then
  c take this path to display step-by-step
  macro step/TotalSteps
else
  c take this path to display only end results
  zigzag/step TotalSteps
  macro/run Diagnostics
endif
time
```

```
plot/w @Name_6.plt
title optical beam through amplifier
plot/1 OpticalHistory xrad=PlotWidth yrad=(NcHistory+1)*StepLength ns=512 h=.3
th=42
```

Jump to: [Commands](#), [Theory](#)

```

plot/w @Name_7.plt
title population inverstion through amplifier
plot/l/ginversion GainHistory xrad=PlotWidth yrad=(NcHistory+1)*StepLength
ns=256 h=.1 th=42
variable/set Position zigzag/position list

plot/w @Name_8.plt
title population inversion through amplifier
plot/b/g GainHistory

plot/w @Name_9.plt
title optical beam through amplifier
plot/b/i OpticalHistory max=.6e10

copy/con OpticalBeam TempArray
plot/w @Name_10.plt
title current optical distribution
plot/l TempArray ns=GainNline

C Name = @Name
energy/list OpticalBeam
end

c =====Macros=====

macro/def step
  Count = Count + 1
  zigzag/step 1
  macro/run Diagnostics
macro/end

macro/def Diagnostics
  plot/w @Name_4.plt
  title medium field N2, Count = @Count
  plot/x/a GainArray left=-PlotWidth right=PlotWidth fmin=0

  c
  c find folded optical pattern by zigzag/image
  c

  zigzag/image TempArray      # form optical image at current step
  variable/set Iteration 1 macro/iteration
  variable/set Max 1 macro/maxiteration
  plot/w @Name_5.plt
  title Iteration = @Iteration out of @Max iterrations
  plot/x/i TempArray left=-PlotWidth right=PlotWidth # fmax=PeakIntensity*3
  c
  c Plot current optical history array
  c
  plot/w @Name_6.plt
  title optical beam through amplifier
  plot/l OpticalHistory xrad=PlotWidth yrad=(NcHistory+1)*StepLength ns=512
  h=.3 th=42

```

Jump to: [Commands](#), [Theory](#)


```

variable/set Position zigzag/position
variable/set GainSection GainArray section
c
c Plot current gain inversion array
c
plot/w @Name_7.plt
title population inverstion through amplifier
plot/l/ginversion GainHistory xrad=PlotWidth yrad=(NcHistory+1)*StepLength
ns=256 h=.1 th=42
c
c Plot bitmap of gain inversion array
c
plot/w @Name_8.plt
title population inversion through amplifier
plot/b/g GainHistory

plot/w @Name_9.plt
title current optical distribution
plot/l TempArray ns=GainNline

macro/end

macro/def Calc
C -----
C macro: Calc
C -----
if [Prism==1] then
tana = tan(AxisToRayRad)
cosa = cos(AxisToRayRad)
AxisToNormalDeg = 90 - AxisToFaceDeg # angle fromm axis to the normal of
the prism face
AxisToNormalRad = Rad*AxisToNormalDeg
c the gain region is in glass
AirIndex = 1.
c Derived geometric parameters
c -----
tanb = tan(Rad*AxisToFaceDeg)
RayExitingDeg = AxisToRayDeg - AxisToNormalDeg list
RayExitingRad = Rad*RayExitingDeg list
if [GainIndex*sin(RayExitingRad)>1] then
MaxAngleRad = asin(1./GainIndex)
MaxAngleDef = MaxAngleRad/Rad
C Error.
C AxisToRayDeg = @AxisToRayDeg is larger than maximum angle of @MaxAngleDef
allowed by asin(1/GainIndex)
C Starting angle would exceed 90 degrees.
C Adjust the axis-to-face angle @AxisToFaceDeg to be closer to the axis-
to-ray angle @AxisToRayDeg.
read/screen
endif
RayIncidentRad = asin(GainIndex*sin(RayExitingRad)/AirIndex) list
RayIncidentDeg = RayIncidentRad/Rad list
AxisToFaceDeg=
AxisToFaceNormalDeg = (90 - AxisToFaceDeg) list

```

Jump to: [Commands](#), [Theory](#)

```

StartingRayAngleDeg = RayIncidentDeg + AxisToFaceNormalDeg list
c form y-axis rotation angle
if [Mode==1||Mode==3] then
    RyDeg = -StartingRayAngleDeg
else
    RyDeg = StartingRayAngleDeg
endif
RyRad = Rad*RyDeg
RyDeg=
MagnificationX = cos(RayIncidentRad)/cos(RayExitingRad)
UnitsStartX = UnitsPrismX*MagnificationX
UnitsGainX = UnitsPrismX/cos(AxisToRayRad) list
else
    c no prism ends, simple reflectors
    GainIndex = 1
    RyRad = AxisToRayRad
    if [Mode==1||Mode==3] then
        RyRad = -RyRad
    endif
    RyDeg = RyRad/Rad list
    c the gain region is a gaseous region with effectively unit index
    UnitsGainX = UnitsX/cos(AxisToRayRad) list
endif
macro/end

macro/def Check
C -----
C macro: Check
C -----
if [Prism==1] then
    AxisToFaceDeg=
    C For prism, height from pivot point of face.
    InterceptHeight = InterceptGainHeight*tanb/(tana + tanb) list
    Even = Reflections==2*(floor(Reflections/2)) list
    if [Even==1] then
        InterceptHeightNet = GainWidth - 2*InterceptHeight list
        RyExitingDeg = 0.
    else
        InterceptHeightNet = 0. list
        RyExitingDeg = 2*RyDeg list
    endif
    C path length through wedge part of prism
    WedgePathLength = InterceptHeightNet/(cosa*(tana + tanb)) list
    C axis projection of path through wedge
    WedgeVLength = InterceptGainHeight/(tana + tanb) list
    C length along gain axis
    TotalV = 2*WedgeVLength + GainLengthV list
else
    InterceptHeight = InterceptGainHeight
    Even = Reflections==2*(floor(Reflections/2)) list
    if [Even==1] then
        InterceptHeightNet = GainWidth - 2*InterceptHeight list
        RyExitingDeg = 0.
    else

```

Jump to: [Commands](#), [Theory](#)

```

    InterceptHeightNet = 0. list
    RyExitingDeg = 2*RyDeg list
endif
TotalV = GainLengthV list
endif
c Beam translation in gain coordinates
c The ending position is a vector (V1,V2,V3) in gain coordinates
V1 = InterceptHeightNet
V2 = 0.
V3 = TotalV
c calculate rotation to global coordinates
sinRy = sin(RyRad)
cosRy = cos(RyRad)
C Exiting beam position in global coordinates
c The ending position is a vector (G1,G2,G3) in gain coordinates
G1 = V1*cosRy + V3*sinRy
G2 = 0.
G3 = -V1*sinRy + V3*cosRy
C Beam translation vector in gain coordinates: @V1, @V2, @V3
C Rotation factors: sinRy = @sinRy, cosRy = @cosRy
C Exiting beam position vector in global coordinates: @G1, @G2, @G3
C Exiting rotational angle, Ry: @RyExitingDeg
C
macro/end

```

Ex121c: Interaction of strongly slanted beams

This examples shows how to use exact pixel matching to model the interaction of strongly slanted beams. In this example, one changes only the the slant angle variable RotY and then the transverse units and axial spacing are computed to accomplish the exact pixel matching.

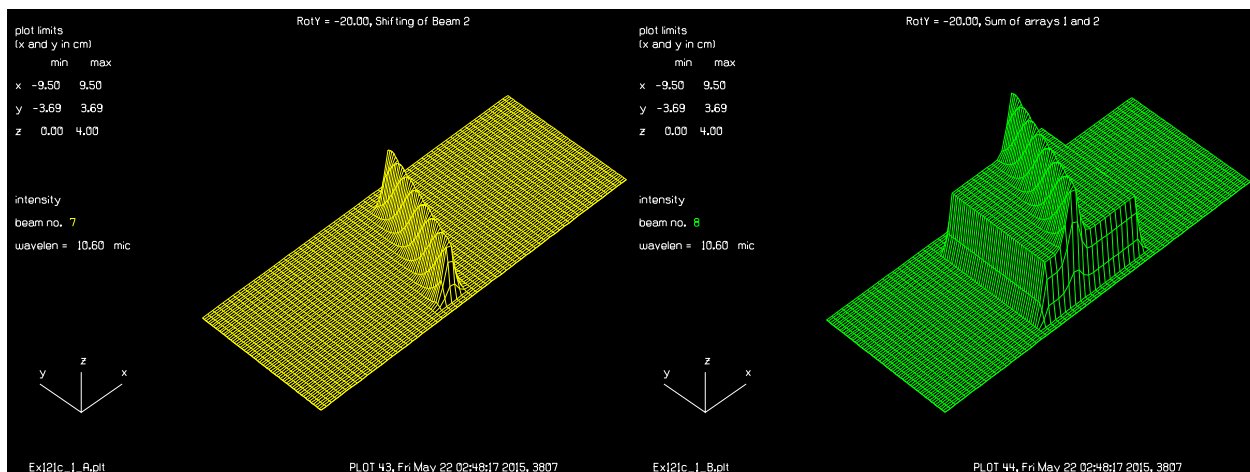


Fig. 121.20. Beam slanted at 10 degrees to be added.

Fig. 121.21. Beam slanted at 10 degrees added to unslanted beam.

Input: `ex121c.inp`

c## Ex121c

Jump to: [Commands](#), [Theory](#)

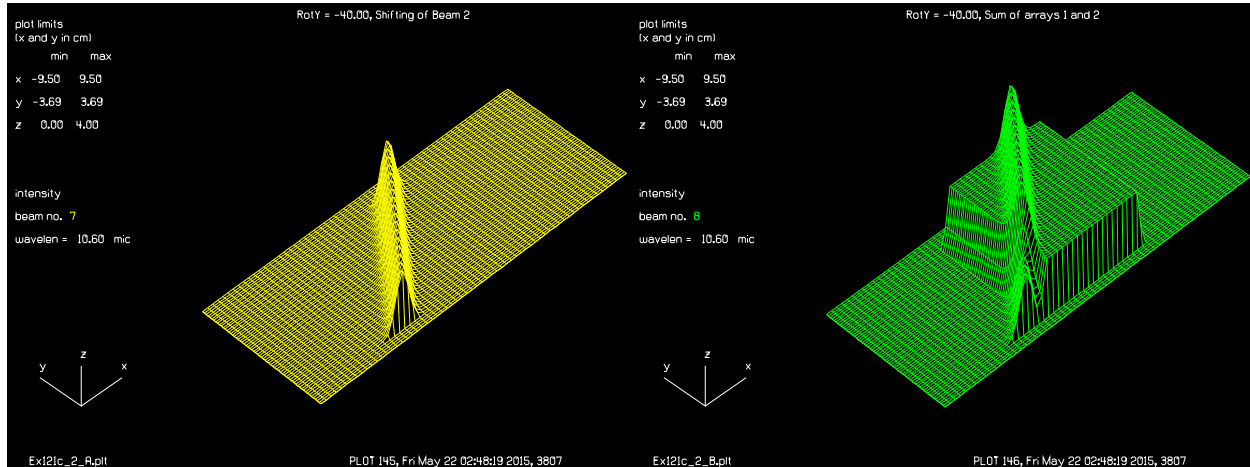


Fig. 121.22. Beam slanted at 40 degrees to be added.

Fig. 121.23. Beam slanted at 40 degrees added to unslanted beam. Note slanted beam passes beyond the unslanted beam at the bottom left.

```

c
c
c Interaction of strongly slanted beams
c
c The example uses the method of exact pixel matching
c to allow a strongly slanted beam to interact with an axial
c beam.
c
c Beam 1 propagates directly along the axis.
c Beam 2 propagates at a slant angle of RotY degrees
c
c Pixels should be matched for axial and slanted beams
c Set axial beam to Units1.
c Slanted units are Units2 = Units1*cos(RotY) .
c Propagation to match pixel shift.
c dZ = Units1/tan(RotY)

alias Name Ex121c
variable/dec/int Icent Pass Row Nsteps Nline1 Nline2 Nc1 Nc2 PlotNum
Nline1 = 256
Nline2 = 64
Nc1 = Nline1/2 + 1
Nc2 = Nline2/2 + 1
Plot_num = 0
Peak = 2
w0 = 2
c
c Set up units to always scale to angle of 40 degrees

Lambda = 10.6e-4
array/set 1 Nline1
nbeam 2                      # create two propagating beams
nbeam 4 data                  # create two more beams for data
nbeam 8 Nline1 Nline2 data    # create history arrays and copies

```

Jump to: [Commands](#), [Theory](#)

```

c Beams
c 1          axial beam
c 2          slant beam
c 3          copy slant beam into axial coordinates
c 4          sum beam in axial coordinates
c 5          history array for decentered beam 2
c 6          history array for summed beam 4
c 7          rescaled from Array 5
c 8          rescaled from Array 6

wavelength/set 1 Lambda*1e4

c
c Calculate properties for 40 degrees which is the last test
c
RotY = -40
Units1_40 = .1
Units2_40 = Units1_40*cosd(RotY)
dZ_40 = Units1_40/tand(abs(RotY))      # propagation distance to shear by Units1
c
c Change the number below to change angle of first run.
RotY = -20
PlotNum = PlotNum + 1
macro/run angle

RotY = -40
PlotNum = PlotNum + 1
macro/run angle
end

macro/def angle
  Units1 = .1
  Units2 = Units1*cosd(RotY)
  dZ = Units1/tand(abs(RotY))      # propagation distance to shear by Units1
  Units1=
  Units2=
  dZ=
  units/set 1,3,4 Units1          # set beams 1,3 to Units1
  units/set 2 Units2 Units1      # Set beam 2 to Units2 and Units1
  units/set 5,6 Units1 dZ        # set units of history arrays
  Icent = 11
  x = Icent*Units1
  y = 0.
  global/def 2 x 0 0 0 RotY 0
  clear 2 1
  Apt = 3
  clap/c 1 3
  gaus/c 2 1 .6
  Nsteps = 2*Apt/dZ + 1
  Row = Nc2- (Nsteps-1)/2
  clear 5 0
  clear 6 0
  write/off
  macro/run step/Nsteps

```

```
write/on
macro/end
macro/def step
  Pass = Pass + 1
  Row = Row + 1
  prop/zproj dZ 2
  vertex/locate/rel 2
  clear 3 0
  variable/set Xdec 2 global/grx # store lateral shift in variable Xdec
  Icent = Icent - 1 list
  copy/con 2 3 icent=Icent
  copy/row 3 5 Nc1 Row
  plot/w @Name_@PlotNum_A.plt
  title RotY = @RotY, Shifting of Beam 2
  copy 5 7
  rescale/units 7 Units2_40 dZ_40
  plot/i 7 max=4
  clear 4 0
  c
  c Instead of interaction physics, form a simple sum of the
  c two beams.
  c
  c Later, add physics and consider cosine effects on the slant array.
  c
  add/inc 4 1 3
  copy/row 4 6 Nc1 Row
  plot/w @Name_@PlotNum_B.plt
  title RotY = @RotY, Sum of arrays 1 and 2
  copy 6 8
  rescale/units 8 Units2_40 dZ_40
  plot/i 8 max=4
macro/end
```

Ex122: Continuous Evolution of Random Processes (movie)

Table. 122.1. Table of Ex122 examples

Ex122a: Continuous evolution of smoothed random distribution	1
Ex122b: Continuous evolution of atmospheric aberration	3

Quasi-continuously varying random processes may be modeled as illustrated in this example.

Ex122a: Continuous evolution of smoothed random distribution

This example illustrates a quasi-continuous randomly varying complex amplitude distribution such as might occur in a multi-mode laser. In this example, it is assumed that from instance to instance about 95 of the distribution is retained and about 5 percent is new. Figure 122.1 illustrates the intensity for one instance. Figure 122.2 shows the average power over the 200 instances..

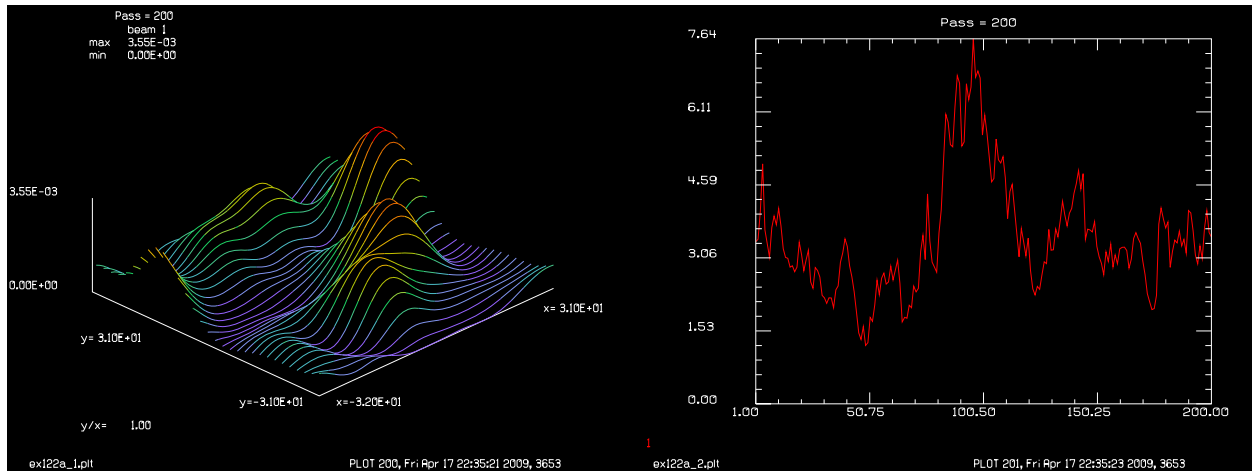


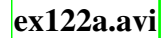
Fig. 122.1. A single instance of a smoothed random distribution.

Fig. 122.2. A plot of the energy in the smoothed random noise distributions over the 200 instances. The process is statistically constant but there is considerable variation from instance to instance in this case.

The process is statistically constant, but there is instance-to-instance variation. A movie of the 200 instances of the smoothed random noise distribution is shown in Fig. 122.3.

Input: ex122a.inp

```
c## ex122a
c
c Generate evolving smoothed random noise pattern
c
c It takes a while for the steady state behavior to develop
c
alias Name ex122a
variable/dec/int CycleSteps Pass RUN
CycleSteps = 20
START = 2*CycleSteps # ignore these early steps
Mult1 = exp(-1/CycleSteps)
```



ex122a.avi

Fig. 122.3. A movie consisting of 200 images showing progression of atmospheric turbulence without a cross wind. The images were displayed in a Watch window and captured using commercial screen capture software. The file ex122a.avi should be present in the installation folder. Click on the phrase “ex122a.avi” in the picture above.

```
Mult2 = 1 - Mult1
nbeam 2
clear 1 0
x = srand(13)    # choose some seed
macro/def step
  if [RUN==1] then
    Pass = Pass + 1 list
  endif
  mult 1 Mult1
  clear 2 0
  noise 2 Mult2
  convolve 2 12
  add/coh/con 1 2
  if [RUN==1] then
    variable/set Energy1 1 energy
    variable/set Energy2 2 energy
    udata/set Pass Pass Energy1 Energy2
    plot/l 1
    title Pass = @Pass
    pause 1
  endif
macro/end
plot/w @Name_1.plt 400 100 480 360
RUN = 0
macro/run step/START
RUN = 1
C pause 5
pause 5
C Go
time/i
macro/run step/200
time
udata/list
plot/w @Name_2.plt 880 100 480 360
```

Jump to: [Commands](#), [Theory](#)


```
plot/udata min=0
```

Ex122b: Continuous evolution of atmospheric aberration

Similarly to the case of evolution of a smoothed random complex amplitude distribution, the continuous evolution of atmospheric aberration may be modeled in a “boiling” condition. Wind shear can be modeled by shifting as indicated in [Ex29](#) in which it is assumed that the structure is essentially constant except the atmosphere moves transversely across the line-of-sight. In a boiling scenario, it is assumed that the distribution changes in-place with about 95 percent remaining the same from instance to instance and 5 percent being new. The aberration process is statistically constant, but there is variation of the Strehl Ratio from instance to instance. To implement aberration addition, the wavefront is converted to and from real values with the commands `waves2real` and `real2waves`. See Figs. 122.4 and 122.5. A movie of the 200 instances modeled is provided [ex122b.avi](#). The movie shows that the distribution changes gradually from one state to another.

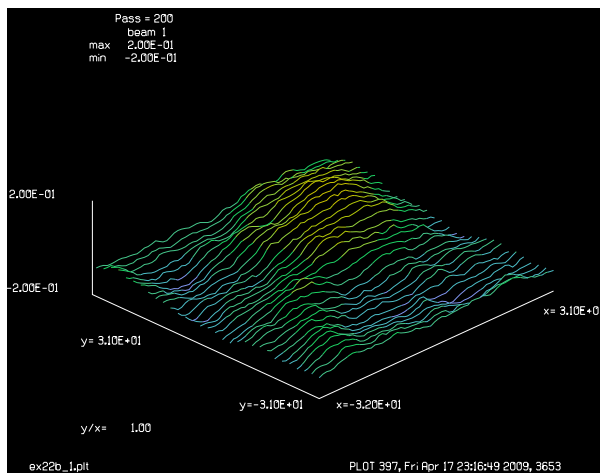


Fig. 122.4. The wavefront built to Kolmogorov statistics in a particular instance. The wavefront changes in a quasi-continuous fashion with statistically constant statistics, although there is variation from instance to instance.

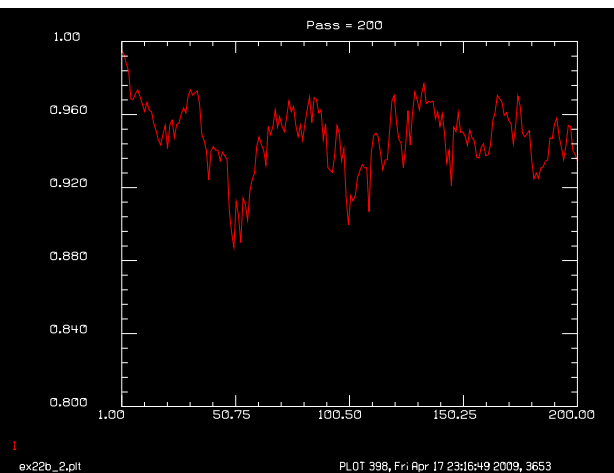


Fig. 122.5. Display of the Strehl Ratio for atmospheric aberration over 200 instances. The average value is constant although there are considerable instance to instance variations.

Input: [ex122b.inp](#)

```
c## ex122b
c
c Generate evolving atmospheric aberration pattern
c
c Set up the coherence temporal length to be 20 times the temporal sample spacing.
c It is assumed that the existing wavefront decreases by Mult1 = exp[-1/CycleSteps] =
.951 per time sample.
c It follows that each new time sample contributes Mult2 = 1 - Mult1 = .049 per time
sample.
c To do the math on the wavefront, the wavefront is converted to "real" values and then
back to wavefront.
c
```

Jump to: [Commands](#), [Theory](#)

```

alias Name ex122b
variable/dec/int CycleSteps Pass RUN Nsteps Max
CycleSteps = 20          # Coherence time length in terms of samples
Nsteps = 200
Mult1 = exp(-1/CycleSteps) list  # will diminish existing wavefront by Mult1
Mult2 = 1 - Mult1 list          # will diminish the new wavefront sample by Mult2
nbeam 3
c
c Beam 1 will contain the evolving atmospheric wavefront
c Beam 2 will contain the evolving atmospheric aberration as "real" data
c Beam 3 will contain an instantaneous atmospheric aberration sheet as "real" data
c
clear 2 0
x = srand(13)              # choose some seed
macro/def step
  Pass = Pass + 1 list
  mult/complex 2 Mult1      # decrease the cumulative "real" wavefront representation
by Mult1
  clear 3 1
  phase/kolmogorov 3 rad0=12 # make a new wavefront sample with Fried's parameter = 12 cm.
  waves2real 3
  mult/complex 3 Mult2      # decrease the "real" wavefront by Mult2
  add/coh/con 2 3
  copy/con 2 1
  real2waves 1 1           # convert "real" wavefront data to a phase sheet.
  variable/set Strehl 1 strehl
  udata/set Pass Pass Strehl
  plot/w @Name_1.plt 400 100 480 360
  title Pass = @Pass
  if [Pass>1] then
    plot/l/w 1 min=-.2 max=.2
    plot/w @Name_2.plt 880 100 480 360
    Max = max(Nsteps,Pass)
    plot/udata min=.8 max=1 right=Max
  endif
  pause 1
macro/end
time/i
C pause 7
plot/w @Name_1.plt 400 100 480 360
pause 7
macro/run step/Nsteps
time

```

Ex123: Encryption and decryption with a hologram^{/Ex123}

Table. 123.1. Table of Ex123 examples

Ex123a: Encryption/decryption, forming sources	2
Ex123b: Encryption/decryption, complex object and point sources	3
Ex123c: Encryption/decryption with hologram, two point objects	6
Ex123d: Encryption/decryption with hologram, noise and point objects	9
Ex123e: Encryption/decryption with hologram, noise and complex objects	11

As is well known, a hologram may be formed by interfering the light from two sources. After development of the hologram, if the hologram is illuminated by one source, the other source is created. A hologram may be used as an encryption/decryption device. In this case a complex random pattern is used as the reference source and the other object source is some pattern to be recovered. In construction, the two sources interfere to create a diffraction pattern which is recorded by film or photoresist to form a hologram. In reconstruction, only the complex random pattern is used. An image of the object is recovered. Unwanted orders are blocked. Figure 123.1 is a schematic of the construction configuration. At the far field, the two sources create an interference pattern. For a complex random reference source, the hologram will be quite complicated for any choice of object source. It would be difficult to determine the object source from the hologram, unless the exact random source is used in reconstruction. Figure 123.2 shows the reconstruction configuration.

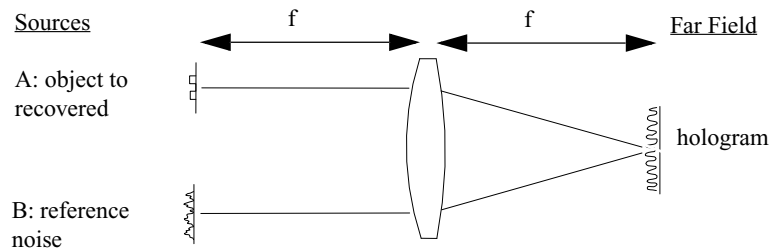


Fig. 123.1. An object source to be encoded and recovered is indicated on the top left. A reference source consisting of complex random noise is shown on the lower left. With the sources located at the front focal plane, the exact far field is located at the rear focal point. An interference pattern is created in the far field and is recorded to make a hologram. The complex random noise creates a complex hologram for any type of source object.

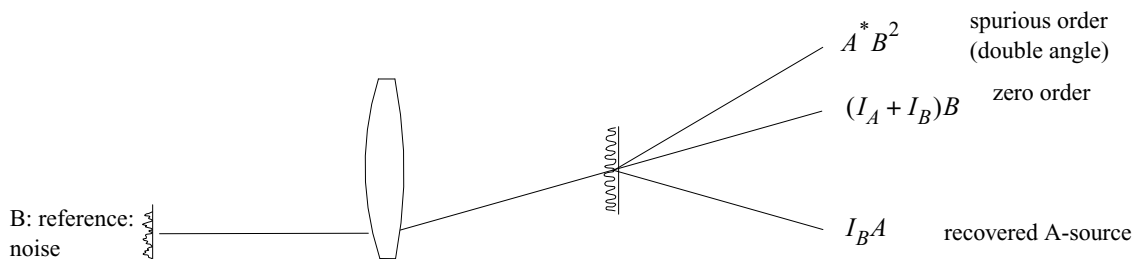


Fig. 123.2. In reconstruction, the complex random source illuminates the hologram. In reconstruction only the B-source is used—the random pattern representing the key. Three orders are created. The reconstructed A-source is $I_A B$ is in the lower left. The zero order $(I_A + I_B)B$ is shown in the left middle. A third, spurious order $A^* B^2$ at the upper left.

The output of the reconstruction process may be calculated from the complex amplitudes, the hologram formation, and the reconstruction process. The irradiance pattern at the far field is indicated in Eq. (123.1)

$$I = (A + B)^* (A + B) = A^* A + B^* A + A^* B + B^* B = I_A + (B^* A + A^* B) + I_B. \quad (123.1)$$

The irradiance pattern is recorded by film or photoresist. If the recording material is linear in response, and only the B-source is used in reconstruction; The result is:

$$[I_A + (B^* A + A^* B) + I_B]B = A^* B^2 + (I_A + I_B)B + I_B A. \quad (123.2)$$

The term $I_B A$ is an intensity scaled version of the original A-object. In Eq. (123.2) terms containing B are diffracted in the opposite direction from A and are removed by blocking the right half-plane as shown in Fig. 123.3.

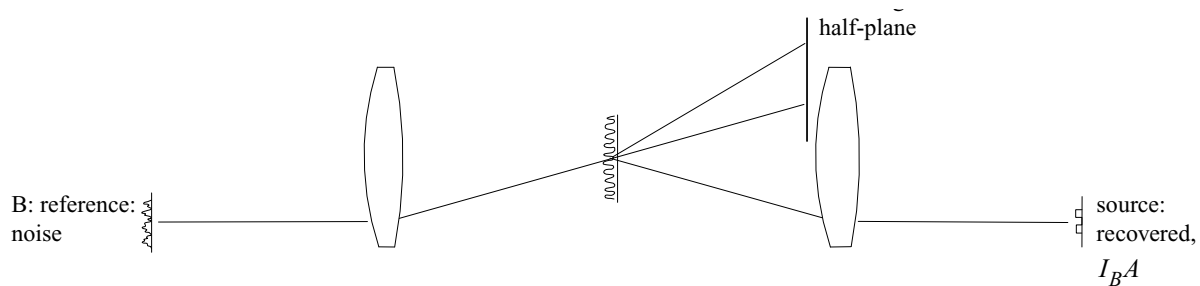


Fig. 123.3. In reconstruction, we can block out the zero and spurious orders leaving only $I_B A$ —the desired object source.

Ex123a: Encryption/decryption, forming sources

In this example, we form a complex source (mask1.dat) and a point source (mask2.dat) and output by outfile for use in Ex123b.inp.

Input: ex123a.inp

```
c## ex123a
#
# Example 123a: Encryption/decryption with hologram, forming sources
#
# Example of encryption and decryption using an object and
# random key mask to make a hologram. When the hologram
# is subsequently illuminated with only the random key mask,
# the object is recovered.
#
# In ex123a, a representative object is created in mask1
# and a point object is created in mask2
#
alias Name Ex123a
variable/dec/int Nobject Nlarge NobjectTemp Size Start
variable/dec/int Input Idestination
Nobject = 32          # number of pixels in object mask
```

Jump to: [Commands](#), [Theory](#)

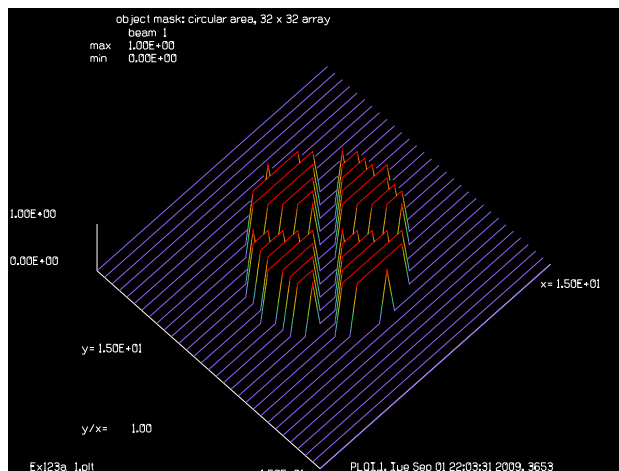


Fig. 123.4. A movie consisting of 200 images showing progression of atmospheric turbulence without a cross wind. The images were displayed in a Watch window and captured using commercial screen capture software. The file ex122a.avi should be present in the installation folder. Click on the phrase “ex122a.avi” in the picture above.

```

NobjectTemp = 32      # number of pixels in oversampled object
Nexpand= NobjectTemp/Nobject
Nlarge = 1024
Size = Nexpand
array/s 1 Nobject data
nbeam 2 data
Units = 1
units/s 0 Units
clap/cir/no 1 8
obs/rec 1 1 1000 azdeg=45
obs/rec 1 1000 1 azdeg=45
plot/w @Name_1.plt
title object mask: circular area, 32 x 32 array
plot/1 1 theta=42 h=.2
c Make a point source for the reference
clear 2 0
point/set 2 NobjectTemp/2+1 17 1
outfile/intens mask1.dat/no/binary 1    # output mask1
outfile/intens mask2.dat/no/binary 2    # output mask2

```

Ex123b: Encryption/decryption, complex object and point sources

Forming a hologram with an imported complex object and a point source. The complex object is the A-source. A point source is used for the reference (B-source). The hologram is formed. Then the the reference source (B-source) is used alone in reconstruction. Three orders are created as indicated in Eq. (123.2). The two orders in the right half-plane are the central order $(I_A + I_B)B$ and a spurious order $A^* B^2$ at twice the angle. Blocking the right half-plane leaves only the desired order $I_B A$ in the left half-plane.

Input: ex123b.inp

```

c## ex123b
#

```

Jump to: [Commands](#), [Theory](#)

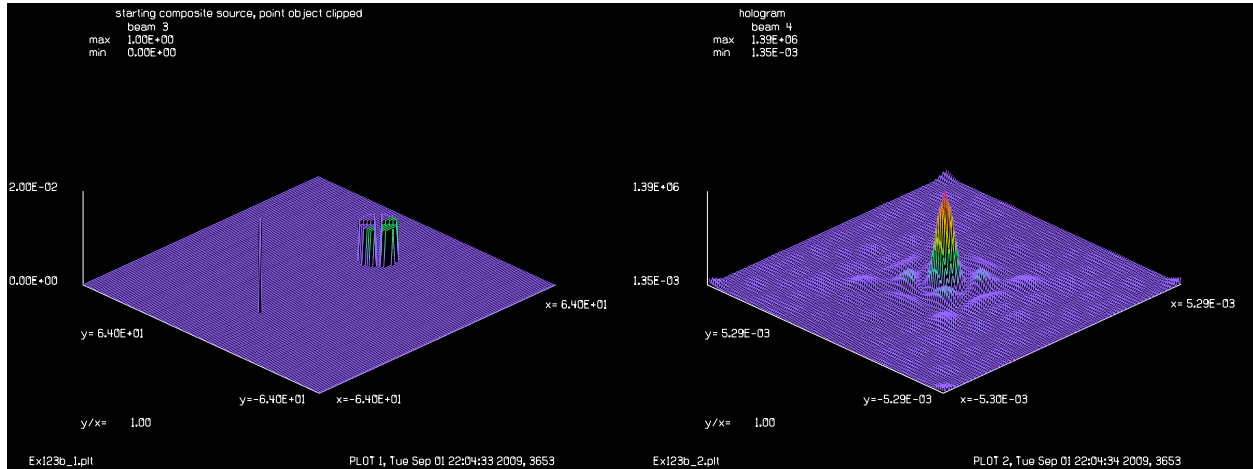


Fig. 123.5. Composite object formed from a complex object (right, the A-object in Fig. 123.1) and a point reference object (right, the B-object in Fig. 123.1).

Fig. 123.6. The hologram formed by the complex object and the point source reference is shown above.

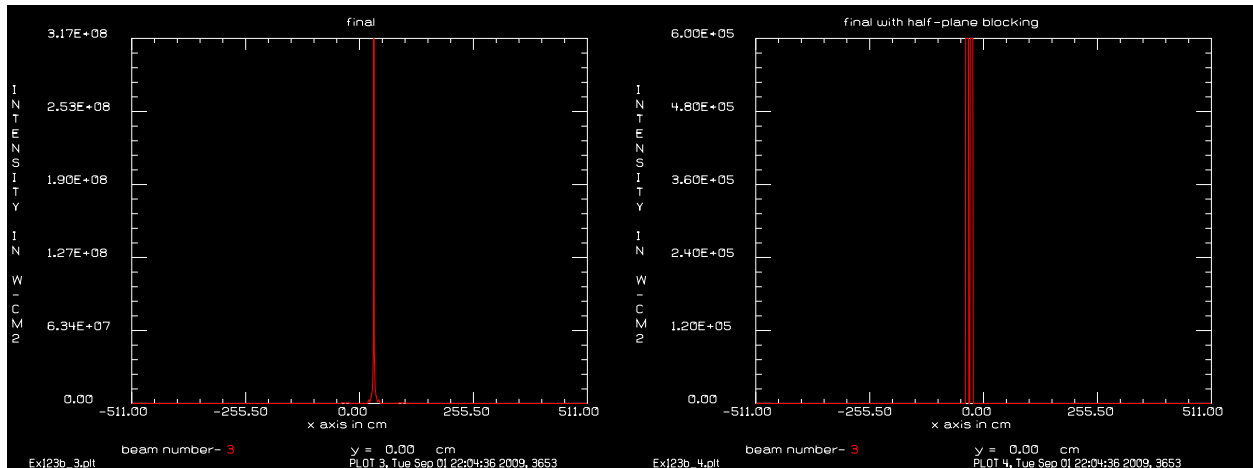


Fig. 123.7. At the second lens before blocking of the right half-plane. Consider Fig. 123.2. The central order $(I_A + I_B)B$ dominates. See Eq. (123.2).

Fig. 123.8. At the second lens after blocking of the right half-plane removes the central order $(I_A + I_B)B$ and the spurious order A^*B^2 , leaving only the recovered complex object $I_B A$ in the left-half plane.

```
# Ex123b: Encryption/decryption with hologram, complex object and point
#
# Read in 32 x 32 bit images, copy with decentering into
# large array, form hologram, reconstruct from point source.
#
alias Name Ex123b
variable/dec/int Nlarge NobjectTemp Shift
NobjectTemp = 32
Nlarge = 1024
Shift = NobjectTemp
array/s 1 NobjectTemp data      # object beam
nbeam 2 NobjectTemp data      # reference beam
nbeam 3 Nlarge                 # propagating beam
nbeam 4 Nlarge data            # hologram
```

Jump to: [Commands](#), [Theory](#)

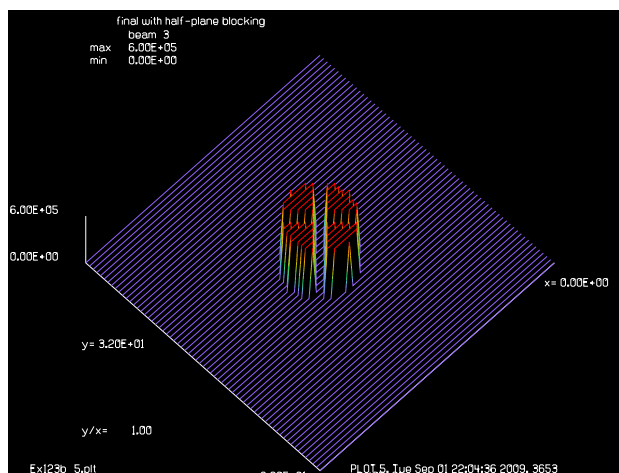


Fig. 123.9. After reconstruction with the point source (B-source) and blocking of the right half-plane, an excellent reproduction of the complex object is achieved. Compare with Fig. 123.4.

```
#
# change this file to either mask1.dat or image.bin
#
infile/intens mask1.dat/no/binary 1    # input object data, complex mask
infile/intens mask2.dat/no/binary 2    # input reference data, point object
energy/norm 1 1                        # balance energy
energy/norm 2 1
clear 3 0
c
c Construct an aperture consisting of mask and point object, both shifted
c
copy/con 1 3 icent=Shift                # copy with shift, A-source, complex object
copy/con 2 3 icent=-Shift               # copy with shift, B-source, point
plot/w @Name_1.plt
title starting composite source, point object clipped
plot/l 3 xrad=64 max=.02 ns=256 # starting field
#
# Form hologram by first going to exact far-field
#
prop 10 3
lens 3 10
prop 10 3
copy 3 4
#
# Form hologram in real word of Array 4
#
irradiance 4                # Form intensity pattern in real word, similar to
                             # chemical development or photoresist

plot/w @Name_2.plt
title hologram
plot/l/r 4 ns=128
#
# Recalculate with just the reference beam and hologram
#
array/s 3 Nlarge
```

Jump to: [Commands](#), [Theory](#)

```

zreff/set 3 0
clear 3 0
units/s 3 1
# recreate one construction beam
copy/con 2 3 icent=-Shift # copy point object B to Array 3 with shift
c
c Go to exact far-field
c
prop 10 3
lens 3 10
prop 10 3
#
# Implement hologram
#
mult/beam 3 4
prop 10 3
lens 3 10
prop 10 3
plot/w @Name_3.plt
title final
plot/x/i 3
#
# Apply decentered aperture to select single sideband
#
clap/sqr 3 1000 xdec=-1000 # mask off positive half plane
plot/w @Name_4.plt
title Final with half-plane blocking
plot/x/i 3
plot/w @Name_5.plt
plot/l 3 ns=256 xrad=32 xdec=-Shift theta=42 h=.2

```

Ex123c: Encryption/decryption with hologram, two point objects

In this example, point objects are used for both the A- and B-sources. A cosine waveform hologram is created and upon reconstruction with the B point source, a near perfect image of the A point is obtained. xx

Input: ex123c.inp

```

c## ex123c
#
# Ex123c: Encryption/decryption with hologram, two point objects
#
# Create a hologram from two point sources. Verify that
# using one point as source creates the other point.
#
# Read in 32 x 32 bit images, copy with decentering into
# large array, form hologram, reconstruct from point source
#
alias Name Ex123c
variable/dec/int Nlarge NobjectTemp Shift
NobjectTemp = 32
Nlarge = 1024

```

Jump to: [Commands](#), [Theory](#)

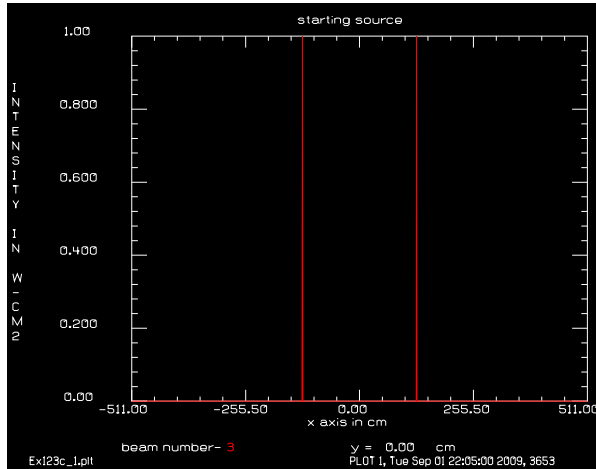


Fig. 123.10. Both A-source and B-source are point objects to create a pure cosinusoidal hologram.

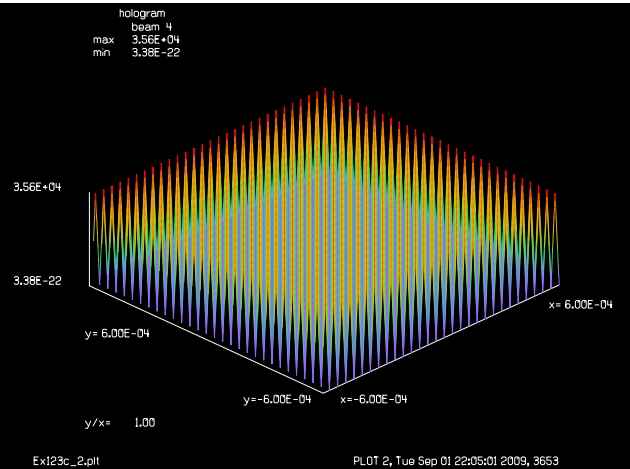


Fig. 123.11. A cosinusoidal hologram (not well resolved in this figure) formed by the two point sources.

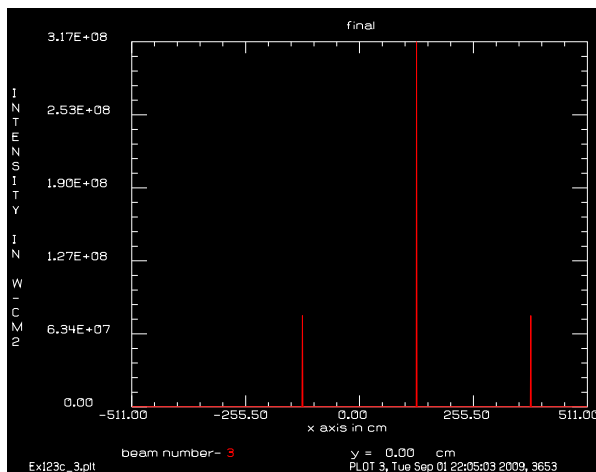


Fig. 123.12. Three orders created by reconstruction with only the B-source. On the left (in the left half-plane) is the recovered A-source $I_B A$. In the right half-plane are the zero order $(I_A + I_B)B$ and a spurious order $A^* B^2$ on the far right.

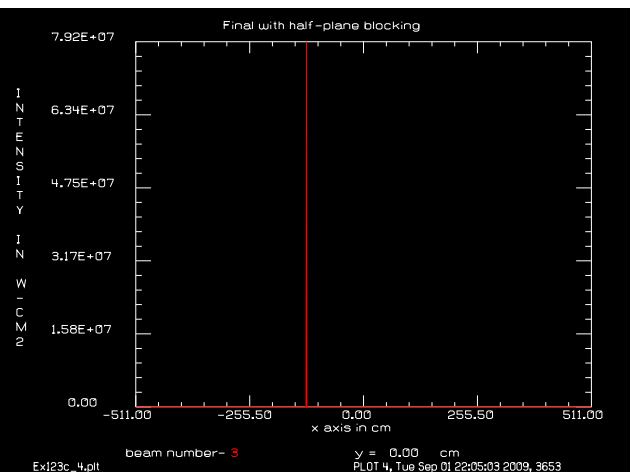


Fig. 123.13. After blocking the right half-plane only the recovered A-source $I_B A$ is left.

```
Shift = 4*NobjectTemp
array/s 1 NobjectTemp data      # object beam
nbeam 2 NobjectTemp data        # reference beam
nbeam 3 Nlarge                   # propagating beam
nbeam 4 Nlarge data             # hologram
clap/c 1 .0001                  # A-source
clap/c 2 .0001                  # B-source
clear 3 0
copy/con 1 3 icent=Shift        # copy with shift
copy/con 2 3 icent=-Shift
plot/w @Name_1.plt
title starting source
plot/x/i 3
#
```

Jump to: [Commands](#), [Theory](#)

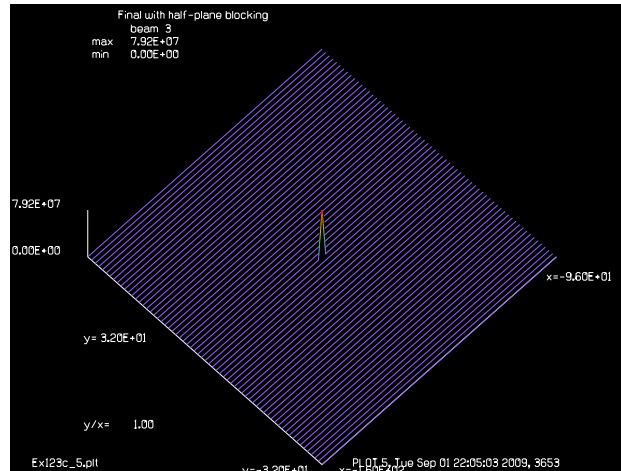


Fig. 123.14. After reconstruction with the point source (B-source) and blocking of the right half-plane, an excellent reproduction of the complex object is achieved. Compare with Fig. 123.4.

```
# form hologram
#
prop 10 3
lens 3 10
prop 10 3
copy 3 4
#
# Form hologram
#
irradiance 4          # Form intensity pattern in real word, similar to
                      # chemical development or photoresist

plot/w @Name_2.plt
title hologram
plot/l/r 4 xrad=.0006
#
# Recalculate with just the reference beam and hologram
#
array/s 3 Nlarge
zreff/se 3 0
clear 3 0
units/s 3 1
# recreate one construction beam
copy/con 2 3 icent=-Shift # reconstruct with only B-source
prop 10 3
lens 3 10
prop 10 3
#
# Implement hologram
#
mult/beam 3 4
prop 10 3
lens 3 10
prop 10 3
plot/w @Name_3.plt
title final
```

Jump to: [Commands](#), [Theory](#)

```

plot/x/i 3
#
# Apply decentered aperture to select single sideband
#
clap/sqr 3 1000 xdec=-1000      # mask off positive half plane
plot/w @Name_4.plt
title Final with half-plane blocking
plot/x/i 3
plot/w @Name_5.plt
plot/l 3 ns=256 xrad=32 xdec=-Shift theta=42 h=.2

```

Ex123d: Encryption/decryption with hologram, noise and point objects

Input: ex123d.inp

```

c## ex123d
#
# Ex123d: Encryption/decryption with hologram, noise and point objects
#
# A random pattern is used as the A-source and a point object is used
# as the B-source. A hologram is formed. After reconstruction with the
# B-source only, the original random pattern (A-source) is recovered.
#
# Read in 32 x 32 bit images, copy with decentering into
# large array, form hologram, reconstruct from point source
#
alias Name Ex123d
variable/dec/int Nlarge NobjectTemp Shift
NobjectTemp = 32
Nlarge = 1024
Shift = 4*NobjectTemp
array/s 1 NobjectTemp data      # object beam
nbeam 2 NobjectTemp data        # reference beam
nbeam 3 Nlarge                   # propagating beam
nbeam 4 Nlarge data              # hologram
nbeam 5 NobjectTemp data        # array to receive recovered A-source
clear 1 0
noise 1 1                        # random noise, A-source
obs/c 1 5 x=9
plot/w @Name_1.plt
title Starting random object with hole, A-source
plot/l 1 theta=42 h=.2
clap/c 2 .0001                   # make a point, B-source
clear 3 0
copy/con 1 3 icent=Shift         # copy with shift, A-source
copy/con 2 3 icent=-Shift        # copy with shift, B-source
plot/w @Name_2.plt
title Starting random object, A-source
plot/l 3 ns=256 xrad=32 xdec=Shift theta=42 h=.2
#

```

Jump to: [Commands](#), [Theory](#)

```

# Form hologram by going to exact far-field
#
prop 10 3
lens 3 10
prop 10 3
copy 3 4
#
# Form hologram as real word in Array 4
#
irradiance 4          # Form intensity pattern in real word, similar to
                      # chemical development or photoresist

plot/w @Name_3.plt
title hologram
plot/l/r 4
#
# Recalculate with just the reference beam and hologram
#
array/s 3 Nlarge
zreff/set 3 0
clear 3 0
units/s 3 1
# Recreate one construction beam, the point, B-source
copy/con 2 3 icent=-Shift
prop 10 3
lens 3 10
prop 10 3
#
# Implement hologram
#
mult/beam 3 4
prop 10 3
lens 3 10
prop 10 3
#
# Apply decentered aperture to select single sideband
#
clap/sqr 3 1000 xdec=-1000      # mask off positive half plane
plot/w @Name_4.plt
title Recreated random noise pattern afer half-plane blocking
plot/x/i 3
plot/w @Name_5.plt
title recovered random A-source
plot/l 3 ns=256 xrad=32 xdec=-Shift theta=42 h=.2
magnify 3 -1
clear 5 0
copy/con 3 5 icent=-Shift      # copy with shift to 5
plot/w @Name_6.plt
title flipped, shifted array 5
plot/l 5 theta=42 h=.2
mult/mode/corr/int 1 5

```

Ex123e: Encryption/decryption with hologram, noise and complex objects

Input: ex123e.inp

```
c## ex123e
#
# Ex123e: Encryption/decryption with hologram, noise and complex objects
#
# Hologram formed and reconstructed with random pattern.
#
# Form the construction beams in two 32 x 32 arrays
# Beam 1 has a square, Beam 2 has the random pattern
#
# Copy Beam 1 and 2 into Beam 3 (1024 x 1024) with shifting
#
# Form hologram. Then reconstruct with the random pattern.
#
alias Name Ex123e
variable/dec/int Nlarge NobjectTemp Shift SWITCH
SWITCH = 1      # 0 use square aperture, 1 use infile
NobjectTemp = 32
Nlarge = 1024
Shift = NobjectTemp # shift the width of the small array
array/s 1 NobjectTemp data      # object beam
nbeam 2 NobjectTemp data        # reference beam
nbeam 3 Nlarge                  # propagating beam
nbeam 4 Nlarge data             # hologram
nbeam 5 NobjectTemp data        # small array to form correlation
if [SWITCH==0] then
  clap/s 1 10      # make a square aperture
else
#
# change this file to either mask1.dat or image.bin
#
  infile/intens mask1.dat/no/binary 1      # input object data
  c infile/intens image.bin/no/binary 1    # input object data
  c infile/intens mask2.dat/no/binary 2    # input reference data
endif
clear 2 0
noise 2 1 seed=11
clear 3 0
copy/con 1 3 icent=Shift      # copy with shift
copy/con 2 3 icent=-Shift
plot/w @Name_1.plt
title starting source
plot/x/i 3
#
# form hologram
#
prop 10 3
lens 3 10
prop 10 3
```

Jump to: [Commands](#), [Theory](#)

```
copy 3 4
#
# Form hologram
#
irradiance 4          # form intensity pattern in real word
plot/w @Name_2.plt
title hologram
plot/l/r 4
#
# Recalculate with just the reference beam and hologram
#
array/s 3 Nlarge
zreff/se 3 0
clear 3 0
units/s 3 1
# recreate one construction beam
copy/con 2 3 icent=-Shift
prop 10 3
lens 3 10
prop 10 3
#
# Implement hologram
#
mult/beam 3 4
prop 10 3
lens 3 10
prop 10 3
plot/w @Name_3.plt
title final
plot/x/i 3
#
# Apply decentered aperture to select single sideband
#
clap/sqr 3 1000 xdec=-1000      # mask off positive half plane
plot/w @Name_4.plt
title final with half-plane blocking
plot/x/i 3
plot/w @Name_5.plt
plot/l 3 ns=256 xrad=16 xdec=-Shift theta=42 h=.2 max=1.5e14
copy/con 3 5 icent=Shift
mult/mode/corr/int 1 5
variable/set Abscorr1 abscorr

# recreate with wrong random construction beam
array/reset 3
zreff/set 3 0
clear 3 0
clear 2 0
noise 2 1 seed=12             # different seed
copy/con 2 3 icent=-Shift
prop 10 3
lens 3 10
prop 10 3
```

Jump to: [Commands](#), [Theory](#)

```
#
# Implement hologram
#
mult/beam 3 4
prop 10 3
lens 3 10
prop 10 3
plot/w @Name_6.plt
title final
plot/x/i 3
#
# Apply decentered aperture to select single sideband
#
clap/sqr 3 1000 xdec=-1000      # mask off positive half plane
plot/w @Name_7.plt
title final with half-plane blocking
plot/x/i 3
plot/w @Name_8.plt
plot/l 3 ns=256 xrad=16 xdec=-Shift theta=42 h=.2 max=1.5e14
copy/con 3 5 icent=Shift
mult/mode/corr/int 1 5
variable/set Abscorr2 abscorr
Abscorr1=
Abscorr2=
```


Ex124: Feedback coupling into a laser from external optics

The frequency of the feedback mode from an external element will be anywhere in the free spectral range depending on the exact round trip path length of the external path. The worst case for interference will occur when the frequency of the external mode lands on one of the regular longitudinal modes. We can know how bad this worst case is simply by calculating the coupling of the mode reflected from a hypothetical dielectric surface with the incident mode.

The command `mult/mode/reflector` calculates the back coupling from a hypothetical (or real) dielectric surface based on the radius and following index of the hypothetical surface.

Ex124: Feedback coupling into a laser from external optics

Simple illustration of feedback coupling from surfaces reflection of a hypothetical surface of specified radius and index of refraction boundary with `mult/mode/reflector`.

Input: ex124.inp

```
c## Ex124.inp
c
c Example 124: Feedback coupling into a laser from external optics
c
c Simple illustration of feedback coupling from surface
c reflection of a hypothetical surface of specified radius
c and index of refraction boundary with mult/mode/reflector.
c
c Reflection from a dielectric boundary can interfere with
c the operation of a laser generating the light. The interference
c will be most significant if the reflected mode matches the
c incident mode closely.
c
c The exact degree of interference depends on the exact frequency
c of the reflected mode which in turn depends on the exact length
c of the reflected path.
c
c The exact length of the extended path that includes the reflecting
c element may either be unknown or may be changing due to temperature
c changes and vibration. So there is considerable uncertainty about
c the exact frequency of the reflected mode.
c
c Knowledge of the coupling coefficient of the reflected mode gives
c a conservative estimate of the worst case interference for any
c partiucarl value of extended path length.
c
c
c Beams
c 1      A reflected mode will be created in beam 1 with the
c        curvature caused by the reflection and the curvature
c        caused by the optical power. The coupling coefficient
c        will be calculated with the original in beam2.
c        The amplitude effect of the dielectric boundary will
c        be explicitly calculated.
```

```

c
c 2          First used for calculating the coupling with the modified
c           beam 1. Later used to calculate using
c           mult/mode/reflector.
c
variable/dec/int Nline
Nline = 256
Lambda = 1e-4
N1 = 1.0
N2 = 1.5
RadiusSurface = -1e7      # make a weak concave curvature
array/set 1 Nline
nbeam 2
wavelength/set 0 Lambda*1e4
units/field 1 2.
copy 1 2                  # make a copy in array 2

c Calculate the reflected mode based a flat reflector and
c the addition of the optical power of the surface radius.
c
OpticalPower = RadiusSurface/-2. list
c
c Form a real reflection and explicitly add the optical power.
c
mirror/flat 1
lens/sph/element/phase 1 OpticalPower
mult/mode/corr 2 1        # Find the correlation of the original beam
                          # in array 2 with the modified beam in array 2

variable/set Corrr1 corrr # real part of correlation
variable/set Corri1 corri # imaginary part of correlation
variable/set Abscorr1 abscorr # magnitude of coupling
c
c Explicitly add the amplitude reflection of the dielectric surface
index_fac = (N1 - N2)/(N1 + N2) list
Corrr1 = Corrr1*index_fac # include amplitude reflection
Corri1 = Corri1*index_fac
Abscorr1 = abs(index_fac)*Abscorr1
PowerCoupling1 = Abscorr1^2

c
c Now use mult/mode/reflector to calculate the effect of a hypothetical
c reflection surface with Array 2.

mult/mode/reflector 2 1.5 xrad=RadiusSurface
variable/set Corrr2 corrr list # Get amplitude correlation from
mult/mode/reflector
variable/set Corri2 corri
variable/set Abscorr2 abscorr
PowerCoupling2 = Abscorr2^2

C
C real: Corrr1 = @Corrr1 should equal Corrr2 = @Corrr2
C imaginary: Corri1 = @Corri1 should equal Corri2 = @Corri2

```

Jump to: [Commands](#), [Theory](#)

```
C
C PowerCoupling1 = @PowerCoupling1 should equal PowerCoupling2 = @PowerCoupling2
C
```


Ex125: Coherent gain model resonator^{Ex125}

Table. 125.1. Table of Ex125 examples

Ex125a: Transient short pulse behavior modeled with coherent gain	1
Ex125b: Coherent gain, short pulse, atomic linewidth	5

The coherent gain model is described in [Sect. 9.15](#). The coherent gain model is built on the theory of Chap. 13, of Sargent, Scully, and Lamb, *Laser Physics*[1]. The rate equation model described in Ex69 and Ex80 and a number of other examples with laser gain is an incoherent model. While the rate equation model is very useful. It has a number of limitations:

- 1) Loss of accuracy with optical effects varying more rapidly than the gain medium response time.
- 2) Atomic line width must be explicitly added.
- 3) Cannot intrinsically explain the formation of longitudinal modes in a laser cavity.
- 4) Cannot describe coherent effects such as π pulses, photon echo, etc.

The coherent gain model remedies all the above problems. It is limited only by 1) the slowly varying envelope (SVE) approximation and 2) the rotating wave approximation, so that optical waves that may be described by an envelope and a carrier frequency may be well represented.

Reference

1. Sargent, Scully, and Lamb, *Laser Physics*, Addison Wesley, 1974.

Ex125a: Transient short pulse behavior modeled with coherent gain

Ex125a compares an analytical expression of the coherent gain model with the GLAD calculations. A pulse of 200ns is amplified by coherent gain with decay time of 10ns.

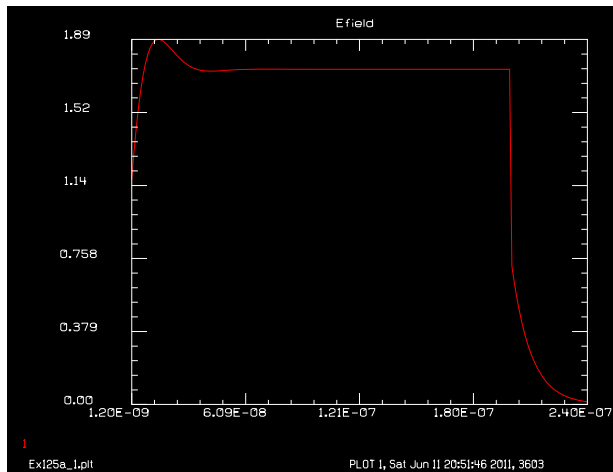


Fig. 125.1. Analytical calculation: E-field response of a 200ns pulse in a 10ns medium. Note initial overshoot and exponential decaying tail at the end of the pulse.

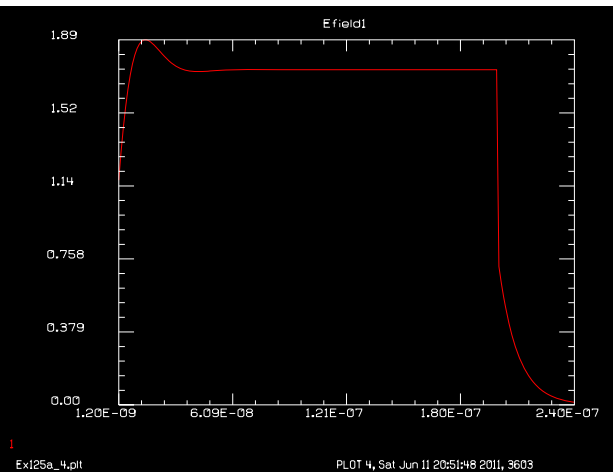


Fig. 125.2. GLAD calculation: E-field response of a 200ns pulse in a 10ns medium. Agrees with analytical calculation in Fig. 125.1

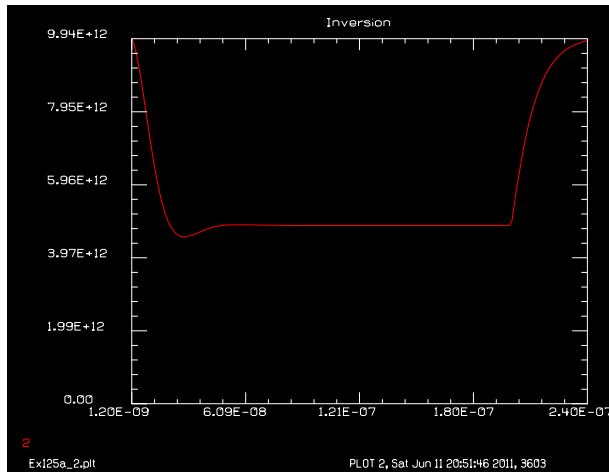


Fig. 125.3. Analytical calculation: population inversion. Note the gradual depletion and restoration to the initial level at the end of the pulse.

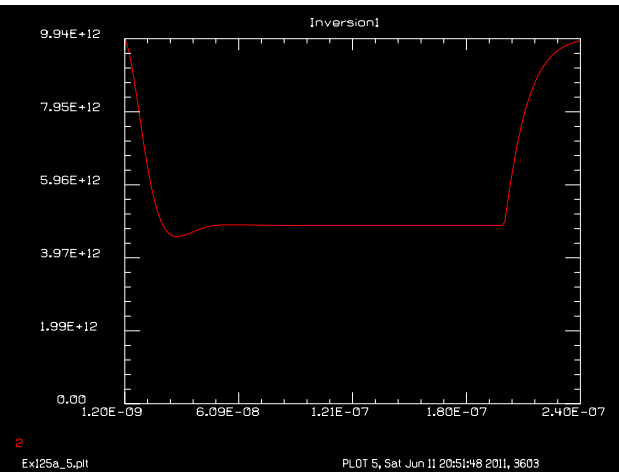


Fig. 125.4. GLAD calculation: population inversion. Agrees with analytical calculation in Fig. 125.3.

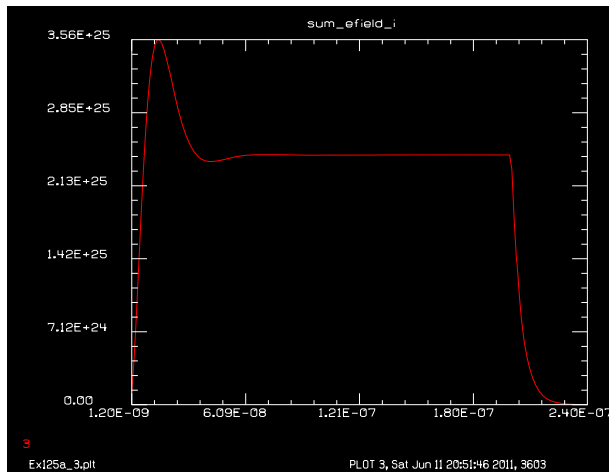


Fig. 125.5. Analytical calculation: medium polarization. Note that the medium polarization builds up in accord with the medium response time.

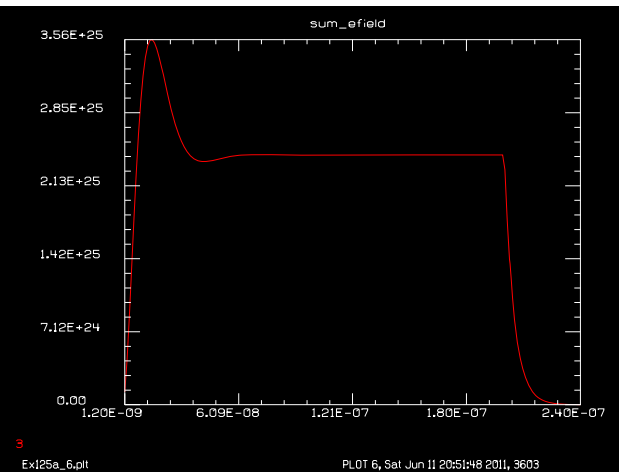


Fig. 125.6. GLAD calculation: medium polarization. Agrees with analytical calculation in Fig. 125.5.

Input: `ex125a.inp`

```
c## ex125a
c
c Example Ex125a: Test of coherent gain
c
c This is an example of the use of coherent gain
c
c Beams Use
c 1 longitudinal mode on line center
c 2 media distribution array
c e work array
c
```

Jump to: [Commands](#), [Theory](#)

```

c write/off
variab/dec/int pass N pass1 Line Ntimes
alias Name Ex125a
Lambda = .248351e-4 list      # wavelength
Index = 1.00                  # index of refraction
v = c/Lambda list             # frequency of radiation
freq = v
Center = v                    # center of atomic line width
hnu = h*v list                 # Joules/photon
tspont = 8.1e-9 list           # spontaneous emission time

Zstep = 0.01 list              # propagation step length
Time = 3e-2 list               # Time between steps
Decay = 1.

Sigma = 3e-11

Efield = 1.

array/s 1 4
array/s 2 4 ipol=1 data
array/s 3 4 data
UnitsX = .1 list
UnitsY = UnitsX list
units/s 0 UnitsX UnitsY
wavelength/set 0 Lambda*1e4 Index # set wavelength and index

Time = 3e-10
TotalTime = 0.
Decay = 1e-8
Rate = 400
D_pump = 2.*Rate/hnu
Inversion = D_pump*Decay list
Inversion_scaled = Inversion*hnu list
CC D_pump = @D_pump, Inversion = @Inversion
sum_efield_d = 0.
macro/def Step
  pass = pass + 1
  Efield0 = 1.
  if [TotalTime>20E-8] Efield0 = 0.
  Efield = Efield0
  decay_factor = exp(-Time/Decay)
  variable/set Line 1 line
  c CC @Line, Inversion = @Inversion
  c Inversion scales with Time
  Inversion = D_pump*Decay + (Inversion - D_pump*Decay)*decay_factor
  Xfac = Inversion*Efield/Decay*Time
  sum_efield_d = (sum_efield_d + Xfac)*decay_factor
  variable/set Line 1 line
  De = .5*Sigma*sum_efield_d*Zstep
  Etotall = Efield*Efield*Time + .5*Inversion*hnu*Zstep
  I1 = Efield*Efield
  Efield = Efield + De
  I2 = Efield*Efield

```

Jump to: [Commands](#), [Theory](#)

```

Dd1 = -2.*(I2-I1)*Time/Zstep/hnu
Inversion = Inversion + Dd1
Inversion_scaled = Inversion*hnu
Etotal2 = Efield*Efield*Time + .5*Inversion*hnu*Zstep
TotalTime = TotalTime + Time
if [mod(pass,N)==0] then
    pass1 = pass1 + 1
    sum_efield_i = sum_efield_d^2
    udata/set pass1 TotalTime Efield Inversion sum_efield_i
endif
macro/end
write/on
N=4
Ntimes = 800
Decay=
macro/run Step/Ntimes

plot/w @Name_1.plt
title Efield
plot/udata 1 min=0
plot/w @Name_2.plt
title Inversion
plot/udata 2 min=0
plot/w @Name_3.plt
title sum_efield_i
plot/udata 3

Inversion = D_pump*Decay list
Inversion_scaled = Inversion*hnu
TotalTime = 0.
sum_efield_d = 0.
Offset = 0          # Frequency offset in Hz
gain/coherent/set Sigma Decay Offset
clear 2 0
clear 3 2.*Rate/hnu
gain/coherent/pump 3 2
clear/complex 3 Inversion
gain/coherent/inversion 3 2
variable/set Inversion1 2 3 3 point/si list
pack/set 1 2
udata/clear
pass = 0
pass1 = 0
macro/def Step1
    pass = pass + 1
    pack/in
    Efield = 1.
    if [TotalTime>20E-8] Efield = 0.
    clear 1 Efield^2
    gain/coherent/step Zstep Time
    TotalTime = TotalTime + Time
    if [mod(pass,N)==0] then
        pass1 = pass1 + 1
        variable/set Efield1 1 3 3 point/sr

```

Jump to: [Commands](#), [Theory](#)


```

variable/set Inversion1 2 3 3 point/si
variable/set sum_efield_r 2 3 3 point/pr
variable/set sum_efield_i 2 3 3 point/pi
sum_efield = sum_efield_r^2 + sum_efield_i^2
  udata/set pass1 TotalTime Efield1 Inversion1 sum_efield sum_efield_r
sum_efield_i
endif
pack/out
macro/end
macro/run Step1/Ntimes
plot/w @Name_4.plt
title Efield1
plot/udata 1 min=0
plot/w @Name_5.plt
title Inversion1
plot/udata 2 min=0
plot/w @Name_6.plt
title sum_efield
plot/udata 3
plot/w @Name_7.plt
title sum_efield_r
plot/udata 4
plot/w @Name_8.plt
title sum_efield_i
plot/udata 5
udata/list

```

Ex125b: Coherent gain, short pulse, atomic linewidth

With the rate equation approximation model, the atomic line width is explicitly imposed. With the coherent gain model, the variation of gain with a range of frequencies about the center frequency of the atomic line is intrinsic to the describing equation. For a pulse length of 240ns the transient atomic linewidth function is a close match to the steady-state function..

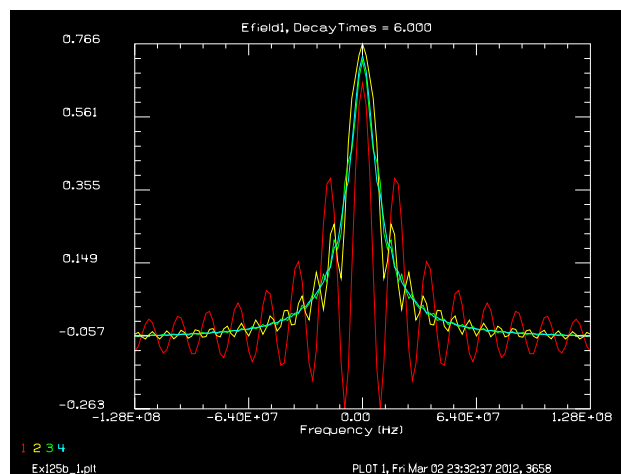


Fig. 125.7. Comparison of intensity for a frequency scan through the atomic line width for various pulse length times of 60ns (red), 120ns (yellow), 180ns (green), and 240ns (cyan).

Jump to: [Commands](#), [Theory](#)

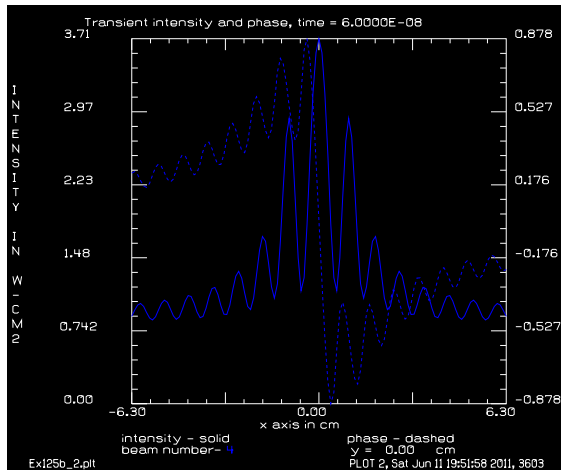


Fig. 125.8. Transient line width vs. frequency after 60ns time.

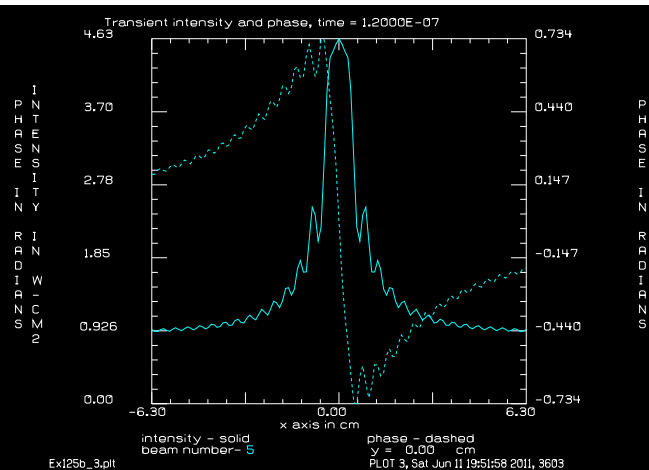


Fig. 125.9. Transient line width vs. frequency after 120ns time.

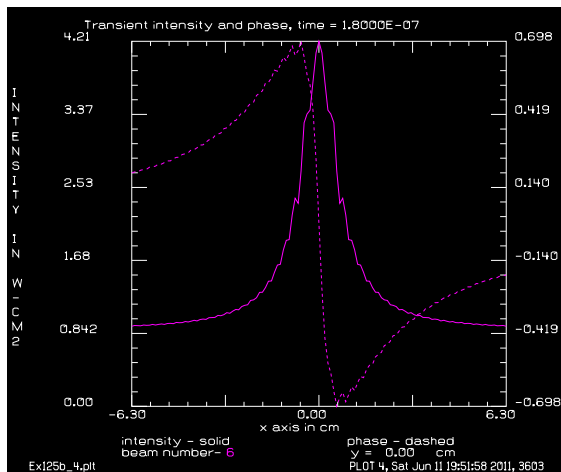


Fig. 125.10. Transient line width vs. frequency after 180ns time.

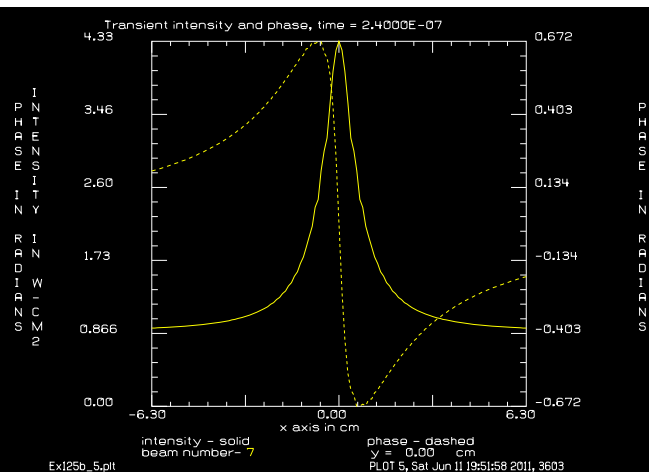


Fig. 125.11. Transient line width vs. frequency after 240ns time.

Input: `ex125b.inp`

```
c## ex125b
c
c Example Ex125b: Test of coherent gain, atomic line width
c
c This is an example of the use of coherent gain
c
c Beams    Use
c 1        longitudinal mode on line center
c 2        media distribution array
c 3        work array
c
c write/off
variab/dec/int pass N N1 N2 pass1 pass2 pass3 Line Ntimes Samples Array I1 I2
alias Name Ex125b
```

Jump to: [Commands](#), [Theory](#)

```

Lambda = .248351e-4 list      # wavelength
Index = 1.00                  # index of refraction
v = c/Lambda list            # frequency of radiation
freq = v
Center = v                    # center of atomic line width
hnu = h*v list                # Joules/photon
tspont = 8.1e-9 list          # spontaneous emission time

Zstep = 0.01 list             # propagation step length
Time = 3e-2 list              # Time between steps

Sigma = 3e-11

Efield = 1.

array/s 1 4
array/s 2 4 ipol=1 data
array/s 3 4 data
nbeam 7 128 1 data
UnitsX = .1 list
UnitsY = UnitsX list
units/s 0 UnitsX UnitsY
wavelength/set 0 Lambda*1e4 Index # set wavelength and index

Rate = 400
Decay = 4e-8
Time = Decay/50 list
MaxTime = Decay*6 list
D_pump = 2.*Rate/hnu
N = 4
Ntimes = MaxTime/Time list
Ntimes4 = Ntimes/4
DecayTimes = Ntimes*Time/Decay list
HalfWidth = 1/Decay list
TotalTime = 0.
macro/def Step2
  pass = pass + 1
  pack/in
  Efield = 1.
  if [TotalTime>MaxTime] Efield = 0
  clear 1 Efield^2
  gain/coherent/step Zstep Time
  TotalTime = TotalTime + Time
  if 0 then
    if [mod(pass,N)==0] then
      pass1 = pass1 + 1
      variable/set Efield1 1 3 3 point/sr
      variable/set Inversion1 2 3 3 point/si
      variable/set sum_efield_r 2 3 3 point/pr
      variable/set sum_efield_i 2 3 3 point/pi
      sum_efield = sum_efield_r^2 + sum_efield_i^2
      udata/set pass1 TotalTime Efield1 Inversion1 sum_efield sum_efield_r
sum_efield_i
    endif
endef

```

Jump to: [Commands](#), [Theory](#)

```

    endif
    pack/out
macro/end
macro/def Step1
    Inversion = D_pump*Decay
    Inversion_scaled = Inversion*hnu
    TotalTime = 0.
    sum_efield_d = 0.
    pass1 = pass1 + 1
    Offset = Offset + dOffset
    gain/coherent/set Sigma Decay Offset
    clear 2 0
    clear 3 2.*Rate/hnu
    gain/coherent/pump 3 2
    clear/complex 3 Inversion
    gain/coherent/inversion 3 2
    variable/set Inversion1 2 3 3 point/si
    pass = 0

    alias Y y01
    Array = 4
    macro/run step3
    alias Y y02
    Array = 5
    macro/run step3
    alias Y y03
    Array = 6
    macro/run step3
    alias Y y04
    Array = 7
    macro/run step3
macro/end
macro/def step3
    macro/run Step2/Ntimes4
    variable/set Efield1 1 3 3 point/sr
    variable/set Efield1_i 1 3 3 point/si
    I1 = pass1
    I2 = 130 - pass1
    point/set Array I1 1 Efield1 Efield1_i
    point/set Array I2 1 Efield1 -Efield1_i
    variable/set Inversion1 2 3 3 point/si
    variable/set sum_efield_r point/pr
    variable/set sum_efield_i point/pi
    sum_efield = sum_efield_r^2 + sum_efield_i^2
    udata/set pass1 Offset @Y=log(Efield1)
    pass2 = N1 + 1 - pass1
    udata/set pass2 -Offset @Y=log(Efield1)
macro/end

pack/set 1 2
dOffset = .2e7
N1 = 129
N2 = (N1+1)/2
pass1 = (N1+1)/2 - 1

```

Jump to: [Commands](#), [Theory](#)

```

Offset = -dOffset
clear 4:7 0
macro/run Step1/N2
plot/w @Name_1.plt
title Efield1, DecayTimes = @DecayTimes
plot/udata/set y01 y02 y03 y04
plot/udata/seq
TempTime = TotalTime/4
plot/w @Name_2.plt
title Transient intensity and phase, time = @TempTime
plot/x 4
TempTime = 2*TotalTime/4
plot/w @Name_3.plt
title Transient intensity and phase, time = @TempTime
plot/x 5
TempTime = 3*TotalTime/4
plot/w @Name_4.plt
title Transient intensity and phase, time = @TempTime
plot/x 6
TempTime = TotalTime
plot/w @Name_5.plt
title Transient intensity and phase, time = @TempTime
plot/x 7
udata/list
Samples = Ntimes list
DecayTimes = TotalTime/Decay list
HalfWidth = 1/Decay list

```


Index

A**ABCD**

equivalent system [72.1](#)

aberration [13.1](#), [30.1](#)

atmosphere [25.1](#)

atmospheric [24.1](#), [26.1](#), [27.1](#), [29.1](#)

axicon [75.26](#)

cone [52.1](#)

grating [104.10](#)

GRIN [106.19](#), [106.26](#), [106.31](#)

plate [88.1](#), [88.6](#)

polarization [75.8](#)

random [17.3](#), [31.1](#), [55.1](#)

scattering [111.1](#)

spherical [7.1](#), [59.1](#), [59.4](#), [93.11](#), [96.1](#), [110.1](#)

thermal blooming [31.1](#)

thermally induced [84.1](#), [92.1](#), [92.9](#), [92.10](#)

tilt [77.1](#), [77.3](#), [77.13](#)

Zernike [13.1](#), [32.1](#)

adaptive mirror [26.1](#), [26.2](#)

adaptive optics [24.1](#), [25.1](#), [27.1](#), [45.1](#)

Airy pattern [40.1](#), [57.1](#), [96.6](#)

array

3D [37.12](#), [105.1](#)

diode [67.1](#)

diodes [67.18](#)

elliptical mirrors [8.6](#)

GRIN lenses [97.1](#)

initialization [1.1](#)

lasers [28.4](#)

lens [64.1](#), [67.1](#)

lens array as optical integrator [81.8](#)

lens array integrator [88.1](#)

lenses with vector addition [102.1](#)

optical fibers [67.15](#), [86.15](#)

phased [28.1](#)

size [2.5](#), [2.11](#)

zone control [81.1](#)

atmospheric

blooming [31.1](#)

jitter [27.1](#), [111.1](#)

turbulence [24.1](#), [25.3](#), [40.1](#), [111.1](#)

wind shear [29.1](#)

B

Babinet's Principle [62.2](#)

bare cavity analysis [11.1](#), [33.1](#), [76.2](#), [109.1](#)

beam fitting [14.1](#)

beam quality [80.4](#)

M-squared [91.1](#)

beam shaping

filter [46.1](#)

optics [110.1](#)

binary optics [89.1](#)

Brewster window [68.1](#)

Burge [116.1](#)

C

CGH [116.1](#)

coherence

partial [83.1](#), [118.1](#)

coherent injection [56.1](#), [76.8](#)

command language [1.1](#)

macros [10.1](#)

udata [10.1](#), [70.1](#)

variables [10.1](#)

coupling

cross [8.4](#)

fiber-to-fiber [106.1](#)

in [42.1](#), [44.1](#)

out [42.1](#), [43.1](#)

TE into TM [8.5](#)

crystal

uniaxial [95.1](#)

D

diode pumping [117.1](#)

dipole projection [90.1](#)

dispersion

group velocity [115.1](#)

E

encryption [123.1](#), [124.1](#)

etalon [119.1](#), [120.1](#)

F

Fabry-Perot [56.1](#), [69.1](#), [76.8](#)
 fiber
 array [86.15](#), [86.17](#)
 multimode [86.12](#)
 fiber optic
 array [67.15](#)
 fiber optics [86.1](#)
 laser [94.1](#)
 four-wave mixing [23.1](#)
 Franz-Nodvik [63.1](#)
 frequency doubling [48.1](#), [49.1](#)
 Fresnel
 diffraction [39.1](#), [86.9](#), [90.1](#)
 number [1.1](#), [11.1](#), [36.1](#), [96.3](#)
 reflection [37.2](#), [37.11](#), [68.3](#)
 transmission [37.4](#), [37.5](#)
 Fresnel number [1.1](#), [96.3](#)
 equivalent [11.1](#)

G

gain
 Beer's Law [63.1](#), [80.17](#)
 coherent [125.1](#)
 coherent, line width [125.5](#)
 Franz-Nodvik [63.1](#)
 rate equation [69.1](#), [80.4](#)
 ruby [69.5](#)
 sheet [47.1](#)
 gaussian
 embedded [91.10](#)
 Hermite [53.1](#), [76.6](#), [91.2](#)
 Laguerre [53.1](#), [54.1](#), [76.6](#)
 geometrical optics [7.1](#), [85.1](#)
 Gerchberg-Saxton [93.1](#)
 global
 commands [6.1](#), [109.1](#)
 grating [104.1](#)
 mirror [7.1](#), [75.35](#)
 optimum [98.2](#)
 resonator [33.13](#)
 Goos-Hanchen [37.10](#), [37.11](#)
 Gouy phase [56.1](#)
 Gouy shift [39.1](#), [76.7](#)
 grating

binary [89.2](#)
 blazed [104.1](#), [104.2](#)
 fan-out [108.1](#)
 global [104.1](#), [104.8](#)
 linear with defect [62.1](#)
 Moire fringes [101.1](#)
 moving [38.1](#)
 nonresolved [104.1](#)
 resolved [104.1](#)
 rhomb [115.1](#)
 Ronchi [101.1](#)
 waveguide [42.1](#), [43.1](#), [44.1](#), [50.1](#), [51.1](#)

GRIN

array [97.1](#)
 lens [106.12](#)
 ground-to-space [25.1](#), [25.2](#), [26.1](#), [27.1](#), [111.1](#)
 group velocity dispersion (GVD) [115.1](#)
 guide star [111.1](#)
 GVD [119.1](#)

H

high numerical aperture [90.1](#)
 hologram
 volume [97.1](#)

I

incoherent
 imaging [65.1](#)
 reflecting-wall waveguide [77.27](#)
 source [83.1](#)
 interferometer
 Mach-Zender [87.19](#)
 Michelson [99.1](#)
 point diffracting [99.1](#), [99.11](#)
 shearing [38.1](#)
 Tyman-Green [99.1](#)
 interferometry
 data reduction [112.1](#)
 Fourier transform method [112.1](#)
 Moire [101.1](#)

J

Jones matrices [37.1](#), [41.1](#)

K

Kohler illumination [83.1](#)
L

laser [25.1](#)

- array [28.4](#)
- Brewster windows [68.1](#)
- characterization with M-squared [91.1](#)
- coherent injection [56.6](#), [79.16](#)
- diode [42.1](#), [43.2](#)
 - multimode [106.36](#)
- diode array [67.1](#)
- excimer [101.1](#)
- fiber optic [94.1](#)
- fiber optics [94.1](#)
- free electron [6.1](#)
- gain [69.1](#)
- ground-to-space [26.1](#), [27.1](#)
- heating [84.1](#), [92.7](#)
- high power [79.1](#)
- Q-switch [80.1](#)
- ruby [69.5](#)
- stable resonator [57.1](#)

lens

- array [64.1](#), [67.1](#)
- array of cylindrical [67.9](#)
- array used for speckle smoothing [88.1](#)
- array with vector addition [102.1](#)
- array, afocal imager [67.13](#)
- array, used as integrator [81.8](#)
- ball [85.13](#)
- beam reshaping [110.1](#)
- binary [89.2](#)
- cylindrical [17.1](#), [18.1](#)
- field [38.1](#)
- GRIN [106.12](#)
- GRIN array [97.1](#)
- high numerical aperture [90.1](#)
- ideal [2.1](#), [8.4](#)
- in resonator [33.4](#)
- lensgroup model [85.1](#)
- relay [83.1](#), [83.2](#)
- simple [5.1](#)
- thermal effects [92.10](#)
- two-focal-length [59.5](#)

- waveguide [87.27](#)

- with steep curve [35.1](#)

Lens Maker's equation [35.2](#)

lensarray command [67.1](#), [88.2](#)

- Schlieren system [71.1](#)

lensgroup command [33.6](#), [67.15](#), [85.1](#), [106.3](#)

lensing medium [45.1](#)

longitudinal distribution [92.15](#)

longitudinal mode competition [69.1](#)

longitudinal modes [56.11](#)
M

mathematical expressions [4.1](#)

memory

- dynamic [73.1](#), [74.1](#)

mirror

- active [45.1](#)
- axicon [75.1](#)
- conic [6.1](#), [8.1](#)
- corner cube [66.1](#)
- dither [88.9](#)
- elliptical [8.4](#)
- flat steering [9.1](#)
- global [7.1](#)
- k-mirror [9.1](#)
- misaligned [12.1](#)
- nutating [88.11](#)
- off-axis [5.1](#)
- phase conjugate [32.1](#)
- relay [25.1](#)
- roof [66.1](#), [80.3](#)
- scraper [11.2](#), [34.1](#), [45.2](#)
- simple [5.1](#)
- walls [77.1](#)

mode fitting [91.1](#)

Moire [101.1](#)

M-squared [2.12](#), [91.1](#)
N

nonlinear

- thermal blooming [31.1](#)

nonlinear mapping

- axicon [75.27](#)

nonlinear optics

- optical limiting [113.1](#)

- phase conjugation [40.1](#)

walk-off [95.3](#)

O

obscuration

- general [15.1](#), [16.1](#)
- in resonator [57.1](#), [76.2](#)
- in Schlieren system [71.1](#)
- through-focus image [59.1](#)

optical limiting [113.1](#)

optical parametric oscillator (OPO) [95.1](#)

optical path length [87.22](#), [99.9](#), [115.1](#), [115.2](#)

optimization

- accelerated mode estimation [61.1](#)
- damped least squares [60.1](#)
- fan-out grating [108.1](#)
- Gerchberg-Saxtion [93.1](#)
- of resonator design [78.1](#)
- phase retrieval [93.1](#)
- simulated annealing [98.2](#)
- size and position of spatial filter [60.1](#)
- to fit Hermite gaussian modes [91.5](#)

OTF [65.1](#)

P

parabola

- off-axis [8.4](#)

phase conjugation [40.1](#)

phase retrieval [93.1](#)

photoelastic tensor [92.14](#)

photolithography [83.1](#)

photoresist [97.1](#)

PIB [91.4](#)

polarization

- axicon [75.2](#)
- birefringence [37.12](#)
- conversion to 3D arrays [105.1](#)
- cross coupling [8.4](#)
- effect of spatial filter [41.1](#)
- in Q-switch laser [80.2](#)
- in resonator with jsurf [68.1](#)
- Jones matrices [37.1](#)
- jsurf [37.5](#)
- thermally induced birefringence [92.16](#)
- waveguide with induced dipole model [51.1](#)
- waveguide, TE and TM [50.1](#)
- with global grating model [104.1](#)

with OPO [95.2](#)

with uniaxial crystal [95.7](#)

with vector addition [102.1](#)

population inversion [69.1](#), [80.3](#), [80.4](#), [80.5](#), [94.1](#)

power-in-the-bucket [91.4](#)

propagation

- circular [96.1](#)
- circular waveguides [103.1](#)
- constant in waveguides [115.1](#)
- distributed [35.1](#)
- in pentagonal waveguides [103.1](#)
- in uniaxial crystal [95.1](#), [95.7](#)
- in volume hologram [97.2](#)
- zone control [81.1](#)

pulse compression [115.1](#)

Q

Q-Switch

- pulse decay [80.5](#)

Q-switch [80.1](#), [80.3](#)

pulsed laser diode pumping [80.23](#), [80.26](#), [80.28](#)

saturable absorber [80.14](#)

slow switching [80.20](#)

R

Raman

- amplifier [17.1](#), [18.1](#), [19.1](#), [20.1](#), [21.1](#), [22.1](#), [23.1](#)
- Stokes, multiple [18.1](#)
- transient [79.1](#)

rand() [27.1](#)

Rayleigh range [2.5](#), [8.9](#), [19.1](#), [31.1](#), [33.1](#), [79.1](#), [115.2](#)

resonator

- Brewster window [68.1](#)
- coherent injection [56.1](#), [76.1](#)
- command [33.1](#)
- flat-flat [100.1](#)
- OPO [95.26](#)
- optimization [61.1](#), [78.1](#)
- polygon [33.8](#), [109.1](#), [109.2](#)
- stable [33.1](#)
- stable in waveguide [77.16](#)
- stable with axicons [75.11](#)

- stable with central obscuration [57.1](#)
- stable-unstable [34.1](#)
- unstable in waveguide [77.19](#)
- unstable ring [45.1](#)
- unstable with axicons [75.13](#)
- unstable, confocal [11.1](#)
- unstable, confocal, apodized [11.5](#)
- unstable, diverging output [11.8](#)
- unstable, misaligned [12.1](#)
- waxicon with inverting optics [75.27](#)
- with waveguide [77.23](#)

rod

- pentagonal [103.9](#)

S

Schlieren [71.1](#)

Schwartzchild [83.1](#)

self-focusing [58.1](#), [58.4](#), [84.1](#), [113.1](#)

Siegman [11.1](#)

simulated annealing [98.1](#)

spatial filter [2.1](#), [17.1](#), [18.1](#)

- high numerical aperture optics [90.4](#)
- optimization of [60.1](#)
- polarization [41.1](#)

speckle

- diode laser, instantaneous [106.36](#)
- excimer laser [101.3](#)
- far-field [98.1](#)
- from rough surface [99.1](#)
- guide star [111.2](#)
- in Q-switch [80.4](#)
- multimode fiber [86.13](#)
- observed in far-field [55.1](#)
- smoothing [88.1](#)
- with Gerchberg-Saxton [93.1](#)

specklon [30.1](#), [58.1](#)

Stokes [18.1](#), [19.1](#), [20.1](#), [79.1](#)

- with four-wave mixing [21.1](#), [22.1](#), [23.1](#)

sum frequency generation (SFG) [107.1](#)

T

table building [82.1](#)

Talbot [62.1](#), [62.2](#), [97.6](#), [101.1](#)

Talbot length [62.2](#)

thermal

- blooming with kinetic cooling [31.5](#)

Jump to: [Commands](#), [Theory](#)

- conduction [92.2](#)
- distortion by finite element [92.1](#)
- optical power change [92.10](#)
- stress birefringence [92.14](#)
- window distortion [92.9](#)

thermal blooming [31.1](#)

thermal distortion

- convolution method [84.1](#)
- finite element method [92.1](#)

through focus

- Raman in converging beam [19.1](#)

through-focus [55.1](#), [59.5](#)

- display of GRIN array [97.6](#)
- embedded gaussian [91.15](#)
- fiber and GRIN system [106.13](#)
- spherical aberration [59.1](#)

U

udata [19.2](#)

unfolding [5.1](#)

units

- automatic control [1.1](#), [2.1](#), [19.1](#)
- control with the zone command [81.1](#)

V

vector diffraction [90.1](#)

W

waveguide

- modeling [86.1](#)
- modeling with extrusion method [87.1](#)
- tapered [77.7](#)
- vector diffraction [50.1](#)

Z

Zel'dovich [30.1](#)

zigzag amplifier [121.1](#)

zone command [81.1](#)