

OSLO

Optics Software for Layout and Optimization

Program Reference

Version 6.X

Lambda Research Corporation
25 Porter Road
Littleton, MA 01460
USA

Tel: 978-486-0766
Fax: 978-486-0755

support@lambdares.com

Warranties

Although Lambda Research Corporation has made every effort to ensure that OSLO and the OSLO documentation is technically accurate, Lambda Research Corporation makes no representations or warranties of any kind whatsoever, directly or indirectly, with respect to the contents hereof or the software described herein. In no event shall Lambda Research Corporation be liable for any damages whatsoever, whether direct or consequential, including without limitation, damages for loss of business profits, business interruption, loss of information, or other pecuniary loss, arising out of the use of or inability to use the information or software, even if Lambda Research Corporation has been advised of, or is otherwise aware of the possibility of such damages.

Trademarks and Copyrights

OSLO[®], and Optics Toolbox[®] are registered trademarks of Lambda Research Corporation.

AppMan[™], Autodraw[™], Spreadsheet Buffer[™], Click/Command[™], SmartCell[™], SCL[™], SCP[™], & CCL[™] are trademarks of Lambda Research Corporation.

Other trademarks and registered trademarks are property of their respective owners and are used for identification purposes only.

Some of the math functions (fft, polint, slveqs, and rand) in the CCL library were adapted from routines appearing in "Numerical Recipes: The Art of Scientific Computing", published by Cambridge University Press, and are used by permission.

Copyright © 2004-2006 Lambda Research Corporation. All rights reserved.

Documentation Guide

The Program Reference and Optics Reference manuals constitute the principal hard-copy documentation for OSLO. The general content of these manuals is as follows.

Program Reference

Description of how to use OSLO, organized according to the default menu commands, with summary descriptions of commands and macro programming at the end.

Optics Reference

Background information of the theory and algorithms used in OSLO, together with examples of typical use, organized according to optical tasks to be performed.

The official documentation for all versions of OSLO is the on-line help system, specifically the Commands reference volume, which is, in fact, the source code for the command definitions that are used to build the program. This information is best referenced through the context-sensitive help files coupled to the individual spreadsheets and dialog boxes in the program

Table of Contents

CHAPTER 1	<i>Introduction</i>	<i>1</i>
	Overview	1
	Documentation Guide	2
	<i>Typographic conventions</i>	2
	Installation	3
	Program User Interface	3
	<i>Main Window</i>	3
	<i>Spreadsheet Windows</i>	4
	<i>Graphics Windows</i>	6
	<i>Text Windows</i>	7
CHAPTER 2	<i>The Main OSLO Window</i>	<i>9</i>
	Overview	9
	<i>Title bar</i>	9
	<i>Menu bar</i>	9
	<i>Tool bar</i>	10
	<i>Status Bar</i>	14
CHAPTER 3	<i>The Surface Data Spreadsheet</i>	<i>17</i>
	Window Control	18
	System Data	20
	<i>Lens (Lens Identification)</i>	20
	<i>Zoom (Configuration data)</i>	20
	<i>EFL (Effective Focal Length)</i>	27
	<i>Entrance Beam Radius (System Aperture)</i>	28
	<i>Field angle (Field of View)</i>	28
	<i>Primary Wavln (Wavelength)</i>	28
	Surface data entry	28
	<i>Row Buttons</i>	29
	<i>Radius of curvature</i>	40
	<i>Thickness</i>	46
	<i>Aperture radius</i>	48
	<i>Glass</i>	53
	<i>Special surface data</i>	61
	<i>Coordinates (C)</i>	66
	<i>Perfect Lens (L)</i>	70
	<i>Polynomial Asphere (A)</i>	71
	<i>Spline Surface (S)</i>	82
	<i>Diffraction Surface (D)</i>	85
	<i>Gradient Index (G)</i>	97
	<i>Multilayer Coating (M)</i>	105
	<i>Polarization Element (Z)</i>	107
	<i>Eikonal Surface (E)</i>	108

<i>Non-sequential Data (Q)</i>	108
<i>User Surface (U)</i>	110
<i>Interferometric Deformation (I)</i>	112
Toolbar Icons	116
Spreadsheet Buttons	117
<i>Gen (General Conditions)</i>	117
<i>Setup (Paraxial Setup)</i>	120
<i>Wavelength</i>	124
<i>Field Points</i>	125
<i>Variables</i>	125
<i>Draw On / Draw Off</i>	125
<i>Group / Surfs</i>	125
<i>Notes (System Notes)</i>	126

CHAPTER 4 *The Graphics Window* 127

Overview	127
Window Control	127
<i>Pop-up Menu</i>	128
<i>Zooming</i>	129
<i>Graphic Preferences</i>	130
Toolbars	130
<i>New graphics window</i>	131
<i>Tile windows</i>	131
<i>Set window title</i>	131
<i>Invert background</i>	131
<i>Right-click actions</i>	131
<i>Standard Tools</i>	132
<i>Lens Drawing</i>	132
<i>Ray Analysis</i>	133
<i>Wavefront</i>	133
<i>Spot Diagram</i>	133
<i>PSF</i>	135
<i>MTF</i>	135
<i>Energy Analysis</i>	136
<i>Zoom</i>	136
<i>Polarization</i>	136

CHAPTER 5 *The Text Window* 137

Overview	137
Window Control	137
Window Operation	137
<i>Pop-up Menu</i>	139
Toolbars	141
<i>Switch text windows</i>	141
<i>Tile windows</i>	141
<i>Set window title</i>	142
<i>Right-click actions</i>	142
<i>Set Toolbars/Row</i>	142
<i>Standard Tools</i>	143

<i>Lens Data Tools</i>	146
<i>Aberrations</i>	146
<i>Ray trace</i>	148
<i>Image Analysis</i>	149
<i>Tolerancing Data</i>	149
<i>Polarization Analysis</i>	150

CHAPTER 6 *The File Menu* 151

Overview	151
File organization	152
<i>Public data</i>	153
<i>Private data</i>	155
New Lens	156
<i>Custom lens</i>	157
<i>Catalog lens</i>	157
<i>Perfect Lens</i>	159
Open Lens	160
<i>Opening a lens file</i>	160
Save Lens	161
Save Lens As	161
Lens Database	161
<i>Public / Private</i>	162
<i>Edit Folders List</i>	167
<i>Make Private Databases</i>	167
Import Lens File	167
Export Lens to CAD	168
Open Database	169
Print Text Window	170
Page Setup (Text)	171
Save Text As	171
Print Graphics Window	171
Page Setup (Graphics)	172
Save Graphics As	172
Preferences	173
<i>Preference Groups</i>	173
<i>Set Preference / Show Preference</i>	174
<i>Restore Default Preferences</i>	180
Reopening a previous file	180
Exiting OSLO	180

CHAPTER 7 *The Lens Menu* 183

Overview	183
Glass Catalogs	184
<i>Glass Manager</i>	184
<i>Update Private Catalog</i>	186
<i>Rebuild Private Database</i>	188

<i>Browse Glass Database</i>	189
Coatings	191
<i>Update Materials</i>	192
<i>Update Designs</i>	193
<i>Show Materials</i>	194
<i>Show Designs</i>	195
<i>Thin Film Uniformity</i>	196
<i>Some general comments on multilayer systems</i>	196
<i>Coating material considerations</i>	197
<i>Some sample multilayer systems</i>	198
Show Surface Data	199
<i>Data Types</i>	200
Show Tolerance Data	205
<i>Surface</i>	205
<i>Component</i>	206
<i>Group</i>	207
Show Optimization Data	207
<i>Variables</i>	207
<i>Field Point / Ray Set</i>	208
<i>Plot Field Points/Ray Set Data</i>	210
<i>Spot Diagram Set</i>	211
<i>Error Function Operands</i>	211
Show Operating Conditions	213
<i>General</i>	213
<i>System Notes</i>	216
<i>Lens Drawing</i>	216
<i>Optimization</i>	216
<i>Configuration</i>	216
<i>Partial Coherence</i>	216
<i>Polarization</i>	216
<i>Tolerance</i>	217
Show Auxiliary Data	217
<i>Edge Thickness</i>	217
<i>Edge Thickness Table</i>	218
<i>Surface Sag</i>	219
<i>Sag Table</i>	219
<i>Lens Weight/Volume</i>	220
<i>Coordinate Matrix</i>	220
<i>Diffraction Surface Phase</i>	221
<i>Diffraction Surface Zones</i>	221
<i>Gradient Index Value</i>	222
<i>Axial Gradient Profile</i>	222
<i>Internal Transmittance</i>	223
<i>Test Glass Analysis</i>	223
Lens Drawing	224
<i>System</i>	224
<i>Drawing types</i>	225
<i>Draw rays</i>	228
<i>Element</i>	230
<i>Zoom Layout</i>	236
<i>Element Drawing Conditions</i>	237
<i>Zoom/Conditions Weights</i>	237

Lens Drawing Conditions	238
-----------------------------------	-----

CHAPTER 8 *The Evaluate Menu* 247

Overview	247
Paraxial setup	249
Paraxial Ray Analysis	251
Zoom Lens Parameters	254
Aberration coefficients	256
Zoom Aberrations	261
<i>Chromatic</i>	261
<i>Third-Order</i>	262
<i>Fifth-Order</i>	262
<i>Zoom Lens Sensitivity</i>	263
Other Aberrations	264
<i>Seidel Wavefront</i>	264
<i>Zernike Wavefront</i>	265
<i>Total Surface Contributions</i>	268
<i>Axial GRIN Seidel Aberrations</i>	268
Single Ray trace	269
Ray Fans	272
<i>Single Field Point</i>	272
<i>2-Dimensional Field Variation</i>	280
Other Ray Analysis	280
<i>Report Graphic</i>	280
<i>Coddington Astigmatism</i>	282
<i>F-Theta Analysis</i>	283
<i>Chromatic Focal Shift</i>	284
<i>Lateral Chromatic Shift</i>	284
<i>Distortion Plot (Grid)</i>	285
Spot Diagram	286
<i>Report Graphic</i>	286
<i>SPD vs. Field (2D)</i>	287
<i>Single Spot Diagram</i>	288
<i>Spot Size Analysis</i>	292
<i>Recipolar Spot Diagram</i>	293
<i>Spot Size and OPD vs. Field</i>	295
Wavefront	296
<i>Report Graphic</i>	296
<i>Wavefront vs. Field (2D)</i>	298
<i>Wavefront Analysis</i>	299
<i>Map (pwf map)</i>	302
<i>Contour (pwf con)</i>	302
<i>Interferogram (pwf int)</i>	303
Transfer Function	304
<i>Through-Frequency Report Graphic</i>	305
<i>Through-Focus Report Graphic</i>	305
<i>Print/Plot OTF</i>	306
<i>Print FFT-based OTF</i>	310
<i>Plot FFT-based MTF</i>	311

<i>MTF and PTF</i>	311
<i>Square-Wave Response</i>	311
<i>MTF vs. Field</i>	312
<i>Through Focus MTF/Field Curvature</i>	313
Spread Function	314
<i>Report Graphic</i>	314
<i>PSF vs. Field (2D)</i>	315
<i>Plot PSF Map/Contour</i>	316
<i>Plot PSF Scans</i>	316
<i>Create Contouring Grid</i>	317
<i>Plot Contours</i>	319
<i>Print PSF Value</i>	320
<i>Print PSF Grid</i>	320
Energy Distribution	321
<i>Diffraction</i>	321
<i>Geometrical</i>	324
<i>Line Spread/Knife Edge</i>	325
Polarization	329
<i>Stokes Parameters</i>	329
<i>Reflectance/Transmittance</i>	330
<i>Polarization Transmittance</i>	332
<i>Pupil Polarization State</i>	332
<i>Polarization Conditions</i>	333
Autofocus	335
<i>Paraxial Focus</i>	
<i>Minimum on-axis spot size (monochromatic)</i>	
<i>Minimum on-axis spot size (polychromatic)</i>	
<i>Field-averaged on-axis spot size (monochromatic)</i>	
<i>Field-averaged on-axis spot size (polychromatic)</i>	
<i>Minimum on-axis RMS OPD (monochromatic)</i>	
<i>Minimum on-axis RMS OPD (polychromatic)</i>	
<i>Field-averaged on-axis RMS OPD (monochromatic)</i>	
<i>Field-averaged on-axis RMS OPD (polychromatic)</i>	335

CHAPTER 9

The Optimize Menu 337

Overview	337
Generate Error Function	338
<i>Singlet</i>	338
<i>Cemented Doublet</i>	338
<i>GENII Ray Aberration</i>	339
<i>OSLO Spot Size/Wavefront</i>	343
<i>Aberration Operands</i>	346
<i>Ray Operands</i>	347
<i>Field Point Set</i>	349
Error Function Tables	350
<i>Field Point Set</i>	350
<i>Ray Set</i>	351
<i>Spot diagram set</i>	352
Operands	353
Variables	366
<i>Thickness variable default bounds</i>	367

<i>Default and adaptive derivative increments</i>	367
<i>Adaptive variable damping</i>	367
<i>Choose derivative increments by type</i>	367
<i>Vary-all options buttons</i>	367
<i>Variable number</i>	368
<i>Surface number</i>	368
<i>Configuration number</i>	368
<i>Variable type</i>	368
<i>Minimum value and maximum value</i>	373
<i>Damping</i>	373
<i>Increment</i>	373
<i>Value</i>	374
Slider-Wheel Design	374
<i>Window setup</i>	374
<i>Using the interactive design windows</i>	375
Iterate	376
Advanced Optimization	378
<i>Downhill Simplex</i>	378
<i>View Derivative Matrix</i>	379
<i>Powell's Method</i>	379
<i>Global Explorer</i>	379
<i>ASA: Adaptive Simulated Annealing</i>	383
<i>Background ASA Window</i>	385
<i>Run Background ASA</i>	386
<i>Stop Background ASA</i>	386
<i>Foreground ASA</i>	386
<i>Terminate ASA</i>	386
Optimization Conditions	387
Support Routines	389
<i>Vignetting</i>	389
<i>Ghosts</i>	393
<i>Y-Ybar</i>	394
<i>Narcissus</i>	394
<i>Convert to Double Pass</i>	395

CHAPTER 10 *The Tolerance Menu* 397

Overview	397
Update Tolerance Data	398
<i>Surface Tolerance Data</i>	398
<i>Component (Air-to-Air) Tolerance Data</i>	404
<i>Group (User-Defined) Tolerance Data</i>	407
<i>Grades/Conditions</i>	409
<i>Set Tolerance Value</i>	412
Monte Carlo Analysis	413
<i>Number of random systems to evaluate</i>	414
<i>Perturbation distributions</i>	414
<i>Plot error function distribution</i>	414
MTF/Wvf Tolerancing	416
<i>Preparation of the system model</i>	417
<i>Analysis Options</i>	418

<i>Statistical performance analysis</i>	420
User-Defined Tolerancing	421
<i>Computing tolerance sensitivities</i>	422
<i>Computing inverse sensitivities</i>	423
Compute Change Table	424
<i>Tolerance units</i>	424
<i>Format of the change tables</i>	425
<i>Change table entry definitions</i>	427
Distribute Changes	431

CHAPTER 11 *The Source Menu* 433

Overview	433
Edit Extended Sources	434
<i>Lambertian</i>	434
<i>LED</i>	436
<i>Arc</i>	437
<i>Creating User-Defined Source Types (Advanced)</i>	439
View Extended Source Image	443
<i>Lambertian / LED / Arc</i>	443
<i>Analysis Results</i>	445
Pixelated Object	446
<i>Input Data</i>	447
<i>Analysis Results</i>	447
Paraxial Gaussian Beam (ABCD)	448
<i>Printing the Gaussian beam data</i>	450
<i>Plot the Gaussian Beam</i>	451
Skew Gaussian Beam (Astigmatic Gaussian Beam)	451
Truncated Gaussian Beam	454
Fiber Coupling	455
<i>Output</i>	457
Partial Coherence	457
<i>Print/Plot Image</i>	458
<i>Partial Coherence Conditions</i>	461

CHAPTER 12 *The Tools Menu* 465

Overview	465
Compile CCL	466
Search CCL Library	466
Plotting Toolkit	467
<i>Gplot Kit Sample</i>	467
<i>Gplot Sample</i>	467
<i>Gplot2D Sample</i>	468
<i>Fill_contour Sample</i>	469
<i>Gplot Template</i>	469
<i>Gplot2D Template</i>	470
<i>Colors and Pens</i>	470
Special	471

<i>2D Waveguide</i>	471
<i>Spherical Mirrors</i>	472

CHAPTER 13 *The Window Menu* 475

Overview	475
Text	475
<i>New & Open</i>	476
<i>Close</i>	477
<i>Reset</i>	477
<i>Title</i>	477
<i>Remove Toolbar</i>	477
<i>Restore Toolbar</i>	478
<i>Save As File</i>	478
<i>Copy to Clipboard</i>	478
Graphics	479
<i>New & Open</i>	479
<i>Close</i>	479
<i>Reset</i>	479
<i>Title</i>	479
<i>Update All</i>	480
<i>Remove Toolbar</i>	480
<i>Restore Toolbar</i>	480
<i>Save As File</i>	480
<i>Copy to Clipboard</i>	481
Editor	481
<i>Open</i>	481
<i>Close</i>	482
<i>UltraEdit</i>	482
Choose Fonts	483
Arrange Icons	484
Status Bar	484
Tile Windows	484

CHAPTER 14 *The Help Menu* 487

Overview	487
OSLO Help	487
Tip of the Day	488
About OSLO	489
<i>Edition</i>	489
<i>Revision</i>	489
<i>License Code</i>	489
<i>ID number</i>	490
<i>Maximum Users</i>	490
<i>License Type</i>	490
<i>Expiration Date</i>	490
<i>License Agreement</i>	490

CHAPTER 15 *Commands* **491**

Overview	491
Working with Commands	491
<i>Command Syntax</i>	491
<i>The Command Line</i>	492
<i>OSLO Editor</i>	493
<i>SmartCells</i>	494
OSLO Commands	494
<i>Optical Analysis</i>	500
<i>Optimization</i>	505
<i>Lens editing</i>	506
<i>Lens Surface Data</i>	508
<i>Operating Conditions</i>	514
Command Definitions	518
<i>General</i>	518
<i>Argument definitions</i>	520
<i>Lists</i>	521

CHAPTER 16 *SCP & CCL Programming* **523**

Overview	523
Data Entry Methods	524
<i>Command Line</i>	524
<i>Internal OSLO Text Editor</i>	524
Global variables	524
Lens data (read-only)	524
<i>Data values</i>	524
<i>Data type codes</i>	525
<i>Non-sequential surface group data</i>	530
<i>Lens array data</i>	531
<i>Optimization data</i>	531
<i>Other variables</i>	532
<i>Data functions</i>	532
Spreadsheet buffer elements	533
Star Command Programming (SCP)	534
Compiled Command Language (CCL)	536
<i>Introduction to CCL</i>	536
<i>Programming in CCL</i>	538
<i>Using CCL with OSLO</i>	547
<i>Error handling in CCL and SCP</i>	550
User-defined support routines	551
<i>User-defined surfaces</i>	551
<i>User-defined gradients</i>	554
<i>User-defined eikonals</i>	555
<i>User-defined operands</i>	558
Dynamic-Link Libraries (DLL's)	559
<i>Writing DLL source code</i>	559
<i>Accessing OSLO global variables from DLL's</i>	560
<i>Accessing the Spreadsheet Buffer from DLL's</i>	561

<i>Executing OSLO commands from DLL's</i>	561
<i>Creating DLL's</i>	562
<i>External operands procedures</i>	562
<i>External user-surface raytrace procedures</i>	563
<i>External user-defined GRIN media</i>	564
<i>External eikonal raytrace procedures</i>	564
<i>Load_library</i>	565
<i>Free_library</i>	565
<i>Get_proc_handle</i>	566
<i>Call_external</i>	566

CHAPTER 17 *CCL Library* 567

Overview	567
Window management	567
File & I/O routines	574
Math library	584
Mathematical and trigonometric functions	590
String functions	591
Graphics library	594

Overview

OSLO is an acronym for *Optics Software for Layout and Optimization*. The original OSLO program was developed at the University of Rochester in the 1970's. The first commercial version was produced in 1976. Since then, OSLO has been rewritten several times as computer technology has advanced. In 1993, Sinclair Optics acquired the GENII program for optical design, and many of the features of GENII are now included in OSLO. Lambda Research purchased the OSLO program to compliment TracePro, an illumination and straylight analysis program developed in-house at Lambda.

To provide a tool that is useful for a broad range of applications, OSLO has been divided according to the general requirements of three user disciplines:

OSLO Premium - for advanced research and development.

OSLO Premium is a flexible, comprehensive optical design program that provides forefront computational tools for optical researchers.

OSLO Standard - for professional optical design.

OSLO Standard is a subset of OSLO Premium that allows you to carry out contemporary lens design using a wide variety of optimization and evaluation options.

OSLO Light - for general optical engineering.

OSLO Light makes use of the unique capabilities of OSLO to provide an economical program, that is simultaneously easy to use and unusually powerful.

The division of OSLO into three editions has been done to extend its range of applicability, and to provide economical access to the program by eliminating advanced features from the simpler versions. For internal program development at Lambda Research, there is only one program. The different editions are obtained by means of compiler directives from the same source code - this reduces the possibility of discrepancies between the different editions.

The features found in OSLO Standard are a subset of those found in OSLO Premium and the features found in OSLO Light are a subset of those found in OSLO Standard. Lenses saved in an edition of OSLO can be read by any other OSLO edition as long as the lens file doesn't contain features that are absent from the edition being used to read the file.

The material in this manual describes features contained in all editions of OSLO. Whenever the functionality between the editions differ, the description will be highlighted with appropriate icon(s) describing which edition(s) the particular

EDU **Lt** **Std** **Prm**

feature appears in. The absence of an icon denotes that the feature is available in all editions: OSLO-EDU, Light, Standard and Premium.

All editions of OSLO are supported on Microsoft Windows ('95, '98, NT4, 2000, Me & XP) as well as LINUX operating systems.

Documentation Guide

The purpose of this manual is to provide an item-by-item description of how to use OSLO, organized according to the default layout of the menu items, toolbars and window options within the program. Summary descriptions of commands and an overview of macro programming is also included. The information contained in this manual is also included in the OSLO on-line help system - with the added benefit that the on-line help system has an extensive search and cross referencing capabilities, and can be updated much more frequently than a printed manual. This manual is for the benefit of those OSLO users that prefer to read and learn about the operation of the OSLO program in a book format.

A companion to this manual is the **OSLO Optics Reference** manual. The Optics Reference contains background information of the theory and algorithms used in OSLO, together with the examples of typical use, organized according to the optical tasks to be performed.

This manual is not necessarily the best way for a new OSLO user to learn about the operation of OSLO. There is a limited description about the operation of OSLO in a later section of this chapter, but a more appropriate place for a new user to learn about OSLO might be the first chapter of the OSLO Optics Reference (entitled, "Quick Start"), or the tutorials, examples, and Knowledge Base items on our web site (<http://www.lambdares.com>).

Typographic conventions

Throughout the manuals there are notes like this in the margin. These notes contain extra information that is supplemental to the main discussion.

Bold type is used to indicate commands, file names, and other material that you type. What you type is usually shown in lower case, although upper case can be used for most input if you prefer.

Italic type is used for emphasis and to indicate a value, when placed in brackets, viz. `<file_name>`.

Placeholders or computer names often contain underscores or embedded capital letters to make them easy to read, such as `directory_name`, or `DirectoryName`.

Monospaced font is used to indicate computer output.

Keystrokes are shown in small capital letters (ENTER). Two keys to be pressed at the same time are linked by a + sign (SHIFT+ENTER); two (or more) keys to be pressed in sequence are linked by commas (ALT, F, X)

Menu items are indicated in normal font with a capital letter. Submenu items are indicated with a double *greater-than* sign. For example, File >> Open refers to the Open command on the File menu.

Most of the material in the manual is valid for all three editions of OSLO (Light, Standard and Premium). When necessary, material that is only relevant to particular OSLO editions is associated with an icon, as shown below.

EDU This material is relevant to OSLO-EDU.

Lt This material is relevant to OSLO Light.

Std This material is relevant to OSLO Standard.

Prm This material is relevant to OSLO Premium.

These icons are typically used in combination. When no icon is present, the material being covered in the manual is relevant to all OSLO editions. When all icons are present, the material being covered is in all editions but the free edition of OSLO: OSLO-EDU.

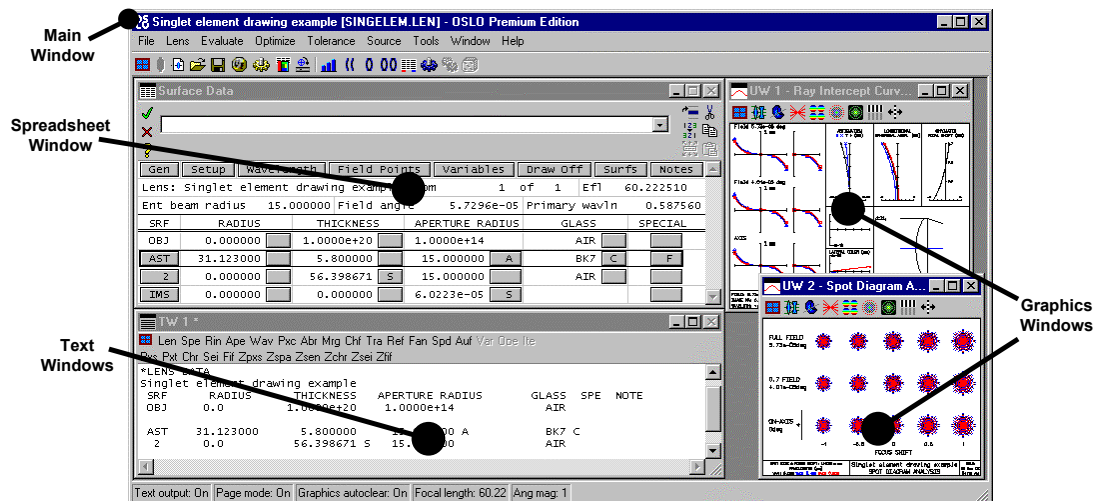
Installation

The installation of OSLO is described in the *OSLO Installation Guide* which is shipped in hardcopy form with the OSLO program. The *OSLO Installation Guide* is also available in Adobe Acrobat format in the "...Docs\" directory on the OSLO installation CD-ROM.

Program User Interface

OSLO works similarly to other windows programs. If you are familiar with other windows software, you will be able to use OSLO without difficulty. However, the OSLO user interface does contain several unique features that make it an efficient and easy-to-use optical design program, and you can enhance your productivity by reading through this chapter.

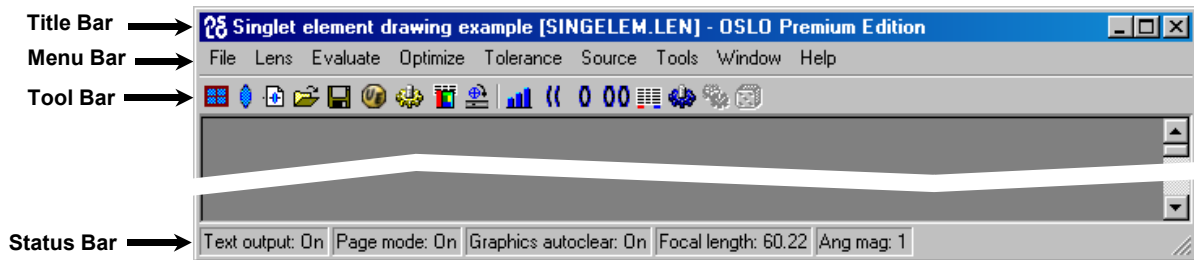
The screen shot below shows a typical configuration of OSLO. It shows the main components of the OSLO user interface: Main Window, Spreadsheet Window, Graphics Windows and Text Windows.



Main Window

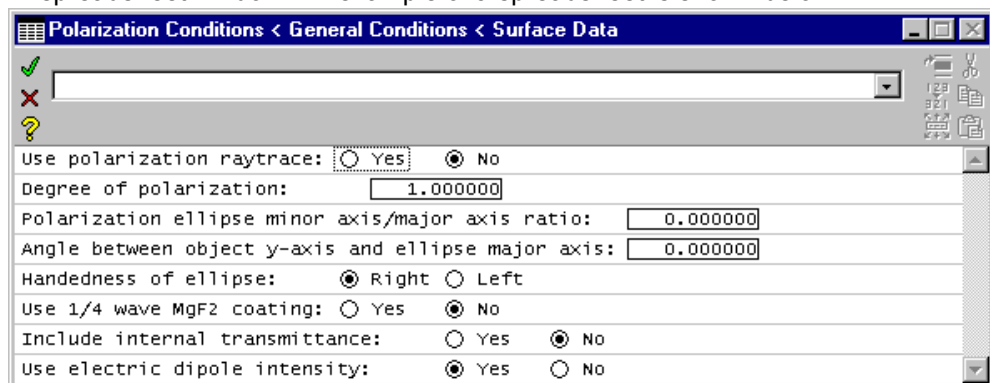
The main window is the central control point for most OSLO tasks. Usually, the main window should be maximized to make use of the full display area, allowing as much room for the other OSLO windows as possible. It contains the title bar,

menu bar, tool bar, status bar, and a large work area that contains the other OSLO windows.



Spreadsheet Windows

OSLO uses spreadsheets for data entry. There can only be one spreadsheet open at a time in OSLO. If you open another spreadsheet while one is still open, the first one will effectively disappear “underneath” the second one. The first spreadsheet will only re-appear when the second is closed. Spreadsheets come with sometimes have drastically different look, but they always open in the same spreadsheet window. An example of a spreadsheet is shown below.



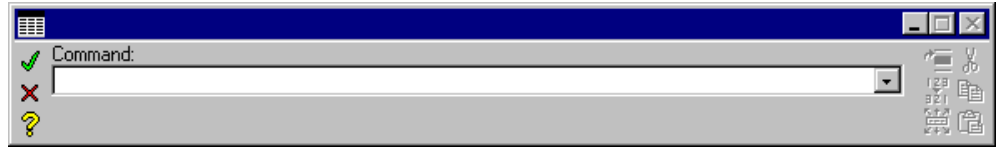
The spreadsheet window title bar shows the names of all the spreadsheets currently in the stack. By looking at the title bar, you can see that there are three spreadsheets “on top” of each other.



This title bar shows that the “Polarization Conditions” spreadsheet is on the top (this is the spreadsheet you are looking at), the “General Conditions” spreadsheet is in the middle and the “Surface Data” spreadsheet is on the bottom of the spreadsheet “stack”. You would have to close the two upper spreadsheets in order to reach the spreadsheet on the bottom: the “Surface Data” spreadsheet, in this case.

OSLO uses the green check (✓), and red X (✗) buttons to control the closing of spreadsheets in OSLO. The green check indicates the user accepts all the recent changes to the data items in the window and the red X indicates that the user wants to ignore all the recent changes to the data items in the window. In both cases, the window is closed after the button is chosen. When the yellow question mark (?) button is chosen, the OSLO on-line help system will start and the user will be taken to the appropriate help page for the current portion of the program.

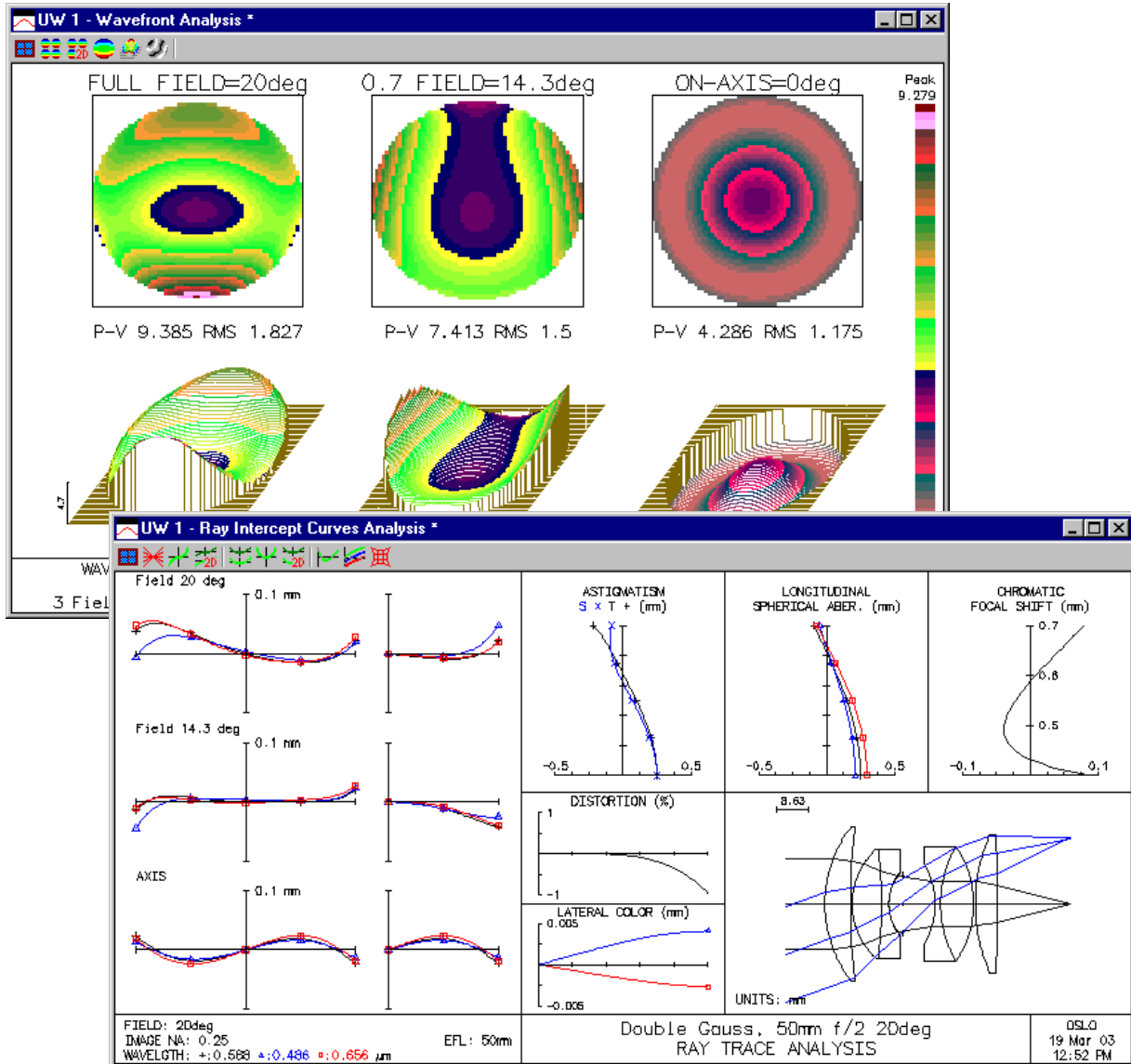
Note that the lowest level OSLO spreadsheet is often the Surface Data Spreadsheet. If the Surface Data Spreadsheet is closed, there will be no spreadsheets showing “underneath” it and the spreadsheet window effectively disappears. The resulting closed spreadsheet window looks like the figure below with just a command line showing.





To re-open the Surface Data Spreadsheet, choose Lens>>Surface Data Spreadsheet from the OSLO menu.

Note that the even when the last spreadsheet is closed, the command line interface is still available. The command line interface is addressed in Chapter 16, ‘SCP & CCL Programming’. The Surface Data Spreadsheet is covered in Chapter 3, ‘The Surface Data Spreadsheet’, and the other spreadsheet windows will be introduced as required throughout this manual.

Graphics Windows



All graphical analysis output will appear in the graphics windows. Up to 32 graphics windows can be open at a time. You can close or minimize any of the graphics windows using the standard MS Windows approach of clicking on the close button in the upper right corner of the window (). Note that the last graphics window in the main OSLO window cannot be closed. When there is only one graphics window left open, the close button in the upper right corner of the graphics window will be disabled (). OSLO always has to have at least one graphics window open at all times (the last graphics window may be minimized or “docked” but it will still be counted as “open”).

Text Windows

***LENS DATA**
Double Gauss, 50mm f/2 20deg

SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPE	NOTE
OBJ	--	1.0000e+20	3.6397e+19	AIR		
1	34.600000	7.000000	21.000000 K	BASF51	C	
2	179.200000	0.500000	21.000000	AIR		
3	20.340000	8.000000	15.000000	LAKN7	C	
4	-137.220000	2.000000	15.000000	SF1	C	
5	13.490000	3.880000	8.500000	AIR		
AST	--	8.770000	7.500000 A	AIR		
7	-21.520000	1.500000	11.000000	BASF2	C	
8	32.170000	9.000000	15.500000	LAF2	C	
9	-27.740000	0.500000	15.500000	AIR		
10	47.670000	6.000000	19.000000	LAF2	C	
11	-416.530000	20.000000	19.000000 K	AIR		
IMS	--	--	18.500000			

***REFRACTIVE INDICES**

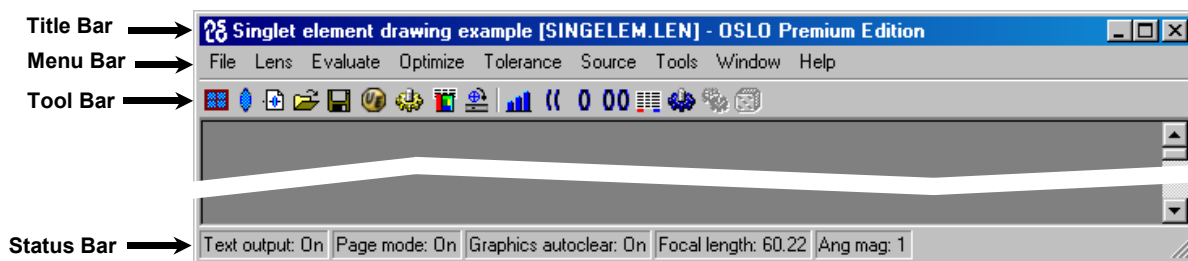
SRF	GLASS	RN1	RN2	RN3	VNBR	TCE
0	AIR	1.000000	1.000000	1.000000	--	--
1	BASF51	1.723728	1.737120	1.718129	38.110473	54.000000
2	AIR	1.000000	1.000000	1.000000	--	236.000000
3	LAKN7	1.651600	1.659345	1.648210	58.516903	71.000000
4	SF1	1.717360	1.734621	1.710314	29.512317	81.000000
5	AIR	1.000000	1.000000	1.000000	--	236.000000
6	AIR	1.000000	1.000000	1.000000	--	236.000000
7	BASF2	1.664460	1.677575	1.659030	35.828763	82.000000

All numerical (text) output appears in the text window. One text window is opened when the program starts. The user can open a second text window. Like graphics windows, a minimum of one text window must be open.

The text window serves as a serial record of a design session, allowing you to scroll backwards through up to 1999 lines of text output. Text output from the program will be directed to the current text window (marked by an asterisk on the title bar). Each text window has its own spreadsheet buffer. The concept of a spreadsheet buffer will be described in Chapter 16, 'SCP & CCL Programming'.

Overview

The main window is the central control point for most OSLO tasks. It contains the title bar, menu bar, tool bar, status bar and a large work area that contains all other OSLO windows. Usually, the main window should be maximized to make use of the full display area, allowing as much room for the other OSLO windows as possible.



Title bar

The title bar includes the Windows system button (displaying the OSLO icon), the ID of the current lens, the name of the current lens file (in brackets), and the name of the edition of OSLO that is running. At the right end of the main window title bar are minimize, maximize, and close buttons, which perform their usual functions for the main window.

Menu bar

The menu bar displays a list of top-level items for the OSLO menu system. Each item contains a pull-down menu with several options. Some options contain pull-right menus with additional choices. The OSLO menu works like other Windows menus, but it can easily be enhanced or modified to include user-supplied commands. The menu that is displayed when OSLO is run is determined by the contents of a user-modifiable file called `a_menu.ccl`. The standard menu file provides the following options - the details of these options are discussed in later chapters of this manual:

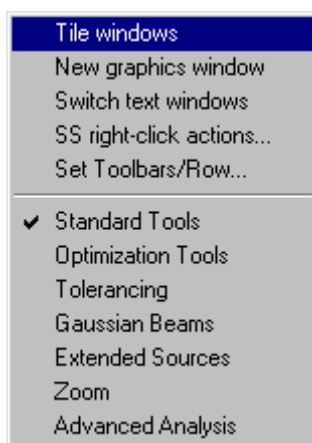
Menu Item	Page
File - Commands that open and save lens files, access lens and other databases, manage printing and plotting, and set program preferences. The last option on the File menu is Exit, which stops the program	151
Lens - Commands that enter or display lens and system data. These include access to OSLO spreadsheets, data output routines, and lens drawings	183
Evaluate - Commands that analyze or compute the performance of the system, such as aberration analysis, ray tracing, and image analysis	247
Optimize - Commands that set up and carry out optimization tasks	337

Tolerance - Commands that set up and carry out tolerancing tasks	397
Source - Commands that define or analyze the current system using special sources, such as extended or Gaussian beam sources.	433
Tools - CCL compiler and special or demo commands supplied with the program	465
Window - Options to open, close, and update windows. Window also includes commands to copy text and graphical output to a file or the Windows clipboard	475
Help - Provides the main entry into the OSLO help system. You can also obtain context-sensitive help by clicking on the help button in a spreadsheet or dialog box	487

Tool bar

Although OSLO has only one menu bar, it has several different tool bars associated with different windows (text windows, graphics windows,...etc.). The main window tool bar has several tool sets that are grouped according to their function. The “Standard Tools” in the main window tool bar include buttons that invoke basic file and data handling commands. Supplemental tools are grouped according to application area (optimization, tolerancing, sources, etc.).

Like the main menu, the tools in the main toolbar are determined by the contents of a user-modifiable file: `a_menu.ccl`.



The main window tool bar begins with the Window Setup button (a blue window with a red frame). This button is found in all OSLO toolbars, and when you click on the Window Setup icon, the resulting pop-up menu is divided in two parts. The top part shows window setup commands, while the bottom part shows various tool sets appropriate for the window. The ***Tile windows*** item, common to all Window setup buttons, is a user-modifiable command that attempts to lay out the various spreadsheet, graphics, and text windows in a convenient pattern. ***New graphics window*** and ***Switch text windows*** create graphics and text windows. The ***SS right-click actions*** item pops up a menu of editing commands that pertain to the selected rows in the current spreadsheet (if any). The SS right-click actions menu also appears when you right-click in a spreadsheet. Finally, the ***Set Toolbars/Row*** item allows you to choose how many tool sets will be concatenated before starting a new row on the toolbar (when the toolbar is shown).

Standard Tools



Open surface data spreadsheet: The lens spreadsheet button opens the main surface data spreadsheet, which in turn contains several buttons that open subsidiary spreadsheets for special and supplemental data. This is equivalent to choosing *Lens>>Surface Data Spreadsheet* from the OSLO menu.



Open a new lens: This button creates a new lens file. It is equivalent to choosing *File>>New Lens* from the OSLO menu.



Open an existing lens: This button opens an existing lens file. It is equivalent to choosing *File>>Open Lens* from the OSLO menu.



Save the current lens: This button saves the current lens. It is equivalent to choosing *File>>Save Lens* from the OSLO menu.

Lt



Open the standard text editor: This tool invokes the internal OSLO editor. This is equivalent to choosing *Window>>Editor>>Open* from the OSLO menu.

Std

Prm



Open UltraEdit: This tool invokes the external editor: Ultra-Edit. Ultra-Edit is a third party program that ships with OSLO Standard and Premium. This is equivalent to choosing *Window>>Editor>>UltraEdit* from the OSLO menu.

Std

Prm



Compile all private CCL: This tool compiles the code in the private CCL directory. This is equivalent to choosing *Tools>>Compile CCL = Private* from the OSLO menu.



Open the database spreadsheet: This button opens the current CDB database spreadsheet, which contains its own menu button and buttons for various column editing functions, as well as user-defined buttons for database callbacks.



Open the slider-wheel spreadsheet: This button opens a spreadsheet used to set up a slider-wheel window. This window allows you to vary lens parameters during design or evaluation by dragging graphic sliders or rotating the mouse wheel. This is equivalent to choosing *Optimize>>Slider-Wheel Design* from the OSLO menu.

Optimization Tools



Autofocus for best average spot size on-axis: This is equivalent to choosing *Evaluate>>Autofocus>>Minimum on-axis spot size (monochromatic)* from the OSLO menu.



Generate error function: This is equivalent to choosing *Optimize>>Generate error function* from the OSLO menu.



Optimize (up to 50 iterations): This is equivalent to choosing *Optimize>>Iterate... Number of iterations=50* from the OSLO menu.



Edit optimization operands. This opens up the Operand Data editor. This is equivalent to choosing *Optimize>>Operands* from the OSLO menu.



Edit optimization conditions: This opens up the Optimization Conditions Editor. This is equivalent to choosing *Optimize>>Optimization Conditions* from the OSLO menu.

Std

Prm



Edit optimization ray set: This opens up the Ray Set Data Editor. This is equivalent to choosing *Optimize>>Error Function Tables>>Ray Set* from the OSLO menu.

Std

Prm











Edit zoom Data: This opens up the Zoom Data Editor. This is equivalent to clicking on the *Zoom* button in the Surface Data Spreadsheet when a system is defined as having more than one configuration.

Prm






Edit optimization spot diagram set: This opens up the Spot Diagram Data Editor. This is equivalent to choosing *Optimize>>Error Function Tables>>Spot Diagram Set* from the OSLO menu.




Tolerancing

- 
Edit tolerancing grades/Conditions: This is equivalent to choosing *Tolerance>>Update Tolerance Data>>Grades/Conditions* from the OSLO menu.
- 
Edit surface tolerances: This is equivalent to choosing *Tolerance>>Update Tolerance Data>>Surface* from the OSLO menu.
- 
Edit component tolerances: This is equivalent to choosing *Tolerance>>Update Tolerance Data>>Component* from the OSLO menu.
- 
Edit group tolerances: This is equivalent to choosing *Tolerance>>Update Tolerance Data>>Group* from the OSLO menu.
- 
Edit tolerancing operands: This is equivalent to choosing *Optimize>>Operands* from the OSLO menu.
- 
MTF/Wavefront tolerancing: This is equivalent to choosing *Tolerance>>MTF/Wvf Tolerancing* from the OSLO menu.
- 
User-defined tolerancing: This is equivalent to choosing *Tolerance>>User-Defined Tolerancing* from the OSLO menu.
- 
Monte-Carlo tolerancing: This is equivalent to choosing *Tolerance>>Monte Carlo Analysis* from the OSLO menu.

Gaussian Beams

- 
Open the Gaussian beam spreadsheet: This is equivalent to choosing *Source>>Paraxial Gaussian Beam (ABCD)* from the OSLO menu.
- 
Astigmatic Gaussian beam analysis: This is equivalent to choosing *Source>>Skew Gaussian Beam* from the OSLO menu.
- 
Single-mode fiber coupling: This is equivalent to choosing *Source>>Fiber Coupling* from the OSLO menu.

Extended Sources

- 
Edit sources: This is equivalent to choosing *Source>>Edit Extended Sources* from the OSLO menu.
- 
Extended source analysis: This is equivalent to choosing *Source>>View Extended Sources* from the OSLO menu.
- 
Pedestaled source analysis: This is equivalent to choosing *Source>>Pixelated Object* from the OSLO menu.

Std **Prm** **Zoom**



Switch to first configuration / Switch to last configuration.



Switch to previous configuration / switch to next configuration: This is equivalent to clicking on the << and >> buttons in the Surface Data Spreadsheet when a system is defined as having more than one configuration.



Automatically update graphic windows when switching configurations.

Prm **Advanced Analysis**



Polarization Conditions: Opens the Polarization Conditions Editor. This is equivalent to choosing *Evaluate>>Polarization>>Polarization Conditions* from the OSLO menu.



Thin Film Materials: Opens the Film Material Definition Editor. This is equivalent to choosing *Lens>>Coatings>>Update Materials* from the OSLO menu.



Multilayer Coatings: Opens the Multilayer Coating Definition Editor. This is equivalent to choosing *Lens>>Coatings>>Update Designs* from the OSLO menu.



Partial coherence conditions: Opens the Partial Coherence Conditions Editor. This is equivalent to choosing *Source>>Partial Coherence>>Partial Coherence Conditions* from the OSLO menu.



Partial coherence imaging: Performs a partial coherence imaging analysis. This is equivalent to choosing *Source>>Partial Coherence>>Print/Plot Image* from the OSLO menu.

Status Bar

OSLO has the option of displaying a status bar at the bottom of its multiple-document-interface (MDI) client area. By default, the status bar is displayed, but the display may be suppressed by turning the **Show_status_bar** preference **off**.

You may select the items to be displayed in the status bar by choosing "Window>>Status Bar" from the OSLO menu, and then choosing the desired items from the resulting "Configure Status Bar" dialog. Up to 16 items may be displayed. Window size, screen resolution, etc., may limit the number of status bar parts that are visible at any given time.

You can also bring up the "Configure Status Bar" dialog by double-clicking on the status bar area.

The possible choices to be displayed in the status bar are given in the table below.

Status Bar Configuration Options

Text output preference	Petzval radius
Page mode preference	Lagrange invariant
Output logging preference	Current wavelength
Output command echo preference	All wavelengths
Graphics autoclear preference	Current configuration
Graphics labels preference	Current object point
Graphics axes preference	Entrance pupil radius
Evaluation mode (focal or afocal)	Entrance pupil position
Ray aiming mode	Exit pupil radius
Lens units	Exit pupil position
Aperture divisions for spot diagrams	CCL error handler
Effective focal length	Current lens directory
Numerical aperture	Current text directory
Working f-number	Test glass file
Lateral (transverse) magnification	ID for files opened from CCL
Angular magnification	Status bar preference 1,2,3,...,16 (See below)
Gaussian image height	

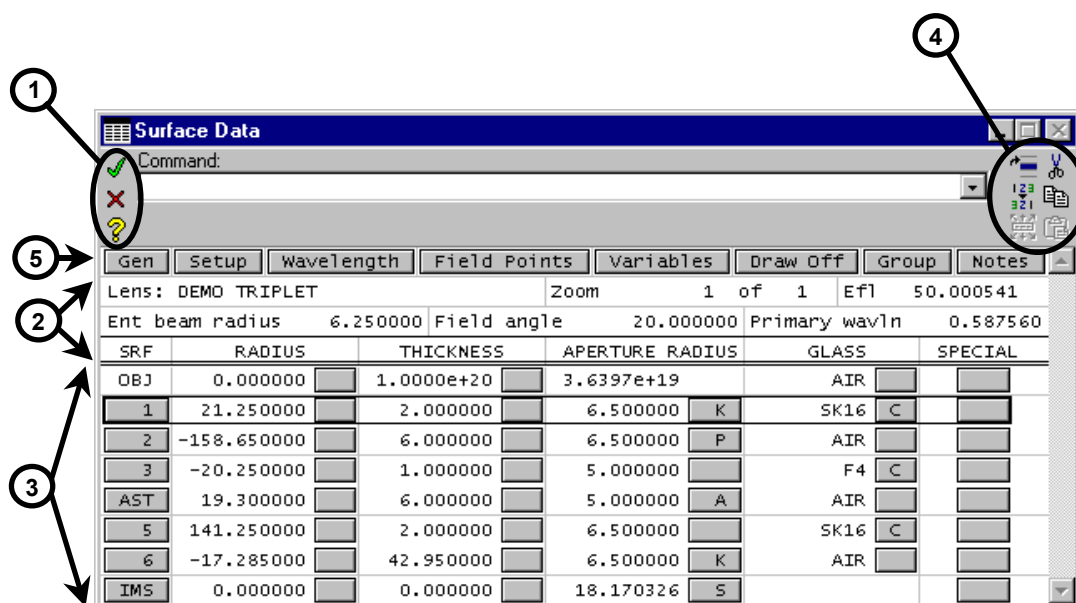
Selecting "Status bar preference" for part *i* of the status bar will display the current value of the Status_bar_*i* preference. The 16 Status_bar_*i* preferences are just string preferences available for the display of arbitrary, user-defined information. See an example in the OSLO on-line help.

Note: string preferences are limited in size by Windows (as are all status bar parts) to a maximum length of 127 characters.

There are three main window types in OSLO:

1. Graphical Analysis Windows (a.k.a. Graphic Windows),
2. Text Analysis Windows (a.k.a. Text Windows), and
3. Spreadsheet Windows.

A short overview describing the different windows in OSLO is outlined in a section of Chapter 1, "Program User Interface". The following section deals with the specifics of one of the most important spreadsheets windows: the Surface Data Spreadsheet.



The layout of the Surface Data Spreadsheet can be broken up into several main regions:

Window Item	Page
1) Window Control buttons allow you to keep or discard changes to the Surface Data Spreadsheet window, and perform context sensitive lookups in the on-line help.	18
2) System Data cells encompass all the optical system parameters that are needed to fully model a lens prescription. System data includes items like wavelengths, magnification, f-number, etc.	20
3) Surface Data rows describe the shape and orientation of the various refracting, reflecting or diffracting surfaces in the lens. This is an area of the spreadsheet that can be scrolled in order to see more of the spreadsheet than can be shown at one time.	28

4) Toolbar Icons are available so that simple mouse clicks can perform functions such as cutting, copying, pasting and inserting blank rows of data into the Surface Data spreadsheet. 116

5) Spreadsheet Buttons are aligned in a row near the top of the Surface Data Spreadsheet window and allow the user to quickly maneuver to other critical spreadsheets that control system parameters. 117

The surface data spreadsheet allows you to change any item of surface data, as well as to access the common system data spreadsheets. To open the surface data spreadsheet, choose “Surface Data Spreadsheet” from the “Lens” menu.

Window Control



Green check button — causes OSLO to accept and terminate the current data entry. It is equivalent to an OK button. If there is no current data entry, the green check button causes OSLO to accept changes and close the spreadsheet. From the keyboard, ENTER accepts the current data entry, but you must press SHIFT+ENTER to close a spreadsheet.

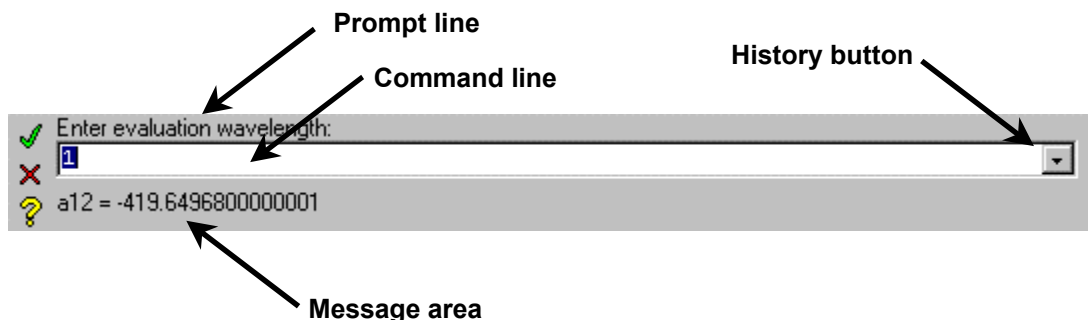


Red X button — equivalent to a Cancel button. The red X is equivalent to pressing ESCAPE, except that when a spreadsheet is open, you must press SHIFT+ESCAPE to close it. The red X button causes OSLO to reject and terminate the current data entry. OSLO has a *Revert_enable* preference that allows you to cancel changes made to a lens in a spreadsheet. If this preference is set, whenever you exit a spreadsheet by clicking the red X button, a confirmation box will appear asking you to confirm that you want to revert to the data that existed when you entered the spreadsheet. Note that the normal response to this query is yes, otherwise you would have clicked the green check button.



Help button — opens the OSLO Help system. It opens different pages, depending on the state of the command line and spreadsheet.

- If the command line is empty and no spreadsheet is open, the Help button goes to the main page of the help system.
- If the command line contains the name of a command, the help button opens that command definition.
- If a spreadsheet is open, the help button navigates to the page of the help system that provides primary help for that spreadsheet.



History button — It is possible to run OSLO using menus, toolbar icons or commands. Whenever menus or toolbar icons are used, the names and definitions of OSLO commands are given in the on-line help system. A convenient way to learn commands is to use the **History buffer** in conjunction with the menus and toolbars. You can execute a desired command using the menu system, then recall it from the History buffer to see the actual command, which can be entered from the command line. Many commands have two forms that are equivalent. The short form is easier to type; the long form is easier to comprehend. If the *Command_history_aliases* preference is on, the short form of commands is shown; otherwise the long form is shown.

The command area itself consists of three sub-areas for text:

- **Prompt line** — The prompt line contains a prompt that either states the current status of the program or requests specific keyboard input. OSLO contains a special feature that generates prompts for all required input data.
- **Command line** — This is where keyboard input to OSLO occurs. When a spreadsheet is active, keystrokes entered into the program are echoed in both the spreadsheet and on the command line. When the command line expects an argument that is an item from a list of allowed values, a pop-up options list appears containing all the allowed values.
- **Message area** — The message area contains informative messages of 1 or 2 lines from the program. Error messages appear in a separate alert box. The message area is also used for program output, such as calculator results or file actions.

Commands in OSLO are self-prompting. To make it easy to use commands, OSLO uses default values for command arguments (also called parameters) where possible. Any remaining arguments are prompted for. In addition, OSLO has a special question-mark parameter. If you want to be prompted for all arguments, enter a command followed by a space and question mark.

Commands can include symbolic input. All lens data can be referenced by name (e.g. CV[2], TH[i], RN[1][1], IMS, AST etc.). In addition OSLO has named storage registers (A-Z, the Spreadsheet buffer, several pre-defined arrays, standard math functions, and various other named global variables). When you enter a numeric expression in the command line, OSLO substitutes the value of the symbols, evaluates the expression, and enters the result. Several commands can be given on a single line, separated by semicolons. Each line can be up to 512 characters, and loops and other control structures allowed in C can be used.

Commands can also be entered while a spreadsheet cell is highlighted. In OSLO, each cell for numeric data is called a SmartCell™. SmartCells automatically detect and execute commands entered in spreadsheet cells. In certain cases, commands may not be legal entries to cells (e.g. you can't enter the command **cv 2 0.001** in the cell for row 3) and are disabled. In addition, where a cell expects a string entry, it may be necessary to enclose it in quotes to prevent it from being interpreted as a command. For example, if you want to use the string "Paraxial_constants" as a lens ID, you must enter it in quotes to prevent Paraxial_constants from being interpreted as a command.

System Data

This section of the spreadsheet summarizes some system parameters. Note that the EFL value is “read-only” (it is calculated from the other system information provided by the user), but most of the other items in this section are user editable.

Lens (Lens Identification)

Underneath the toolbar is a field for a short description of the current lens. This is to help you identify your current lens and is alternately referred to as the Lens ID, Name or Title. The Lens ID can be any combination of words or numbers, up to 32 characters in length and is typically used for labelling plots, output analysis,...etc. In CCL programming, the Lens ID can be set through the “lid” function and accessed through the “lid” global variable. The Lens ID should not be confused with the file name which may be limited by the operating system to 8 characters in length. When a lens is opened, the file name will appear in the title bar of the main window, not the Lens ID.

Prm Zoom (Configuration data)

Std

(OSLO Standard and Premium Only. This cell is blank in OSLO-EDU and Light)

NOTE: Typically, a “zoom” lens system is a multi-element lens system that changes element spacing thereby changing into a new lens system with a different power and focal length. A “multi-configuration” lens system is more general in nature. The multi-configuration lens system might not just change element spacing and positioning (including tilts and decenters), but might also substitute whole lens elements in different configurations. In OSLO, the system for modeling “zoom” lenses and “multi-configuration” lenses is exactly the same. Therefore, for purposes of explanation, the OSLO interface and documentation uses the terms “configuration” and “multi-configuration” interchangeably with “zoom” and “zoom position”.

The lens data that you enter using the surface data spreadsheet and its subsidiary spreadsheets describes an optical system in what may be called the *base* configuration. OSLO has the capability to optimize simultaneously several systems that differ from the base system in only part of their data. Such systems are called *multiconfiguration systems*.

The **Zoom** field in the Surface Data Spreadsheet shows the current zoom position and the total number of zoom positions that are defined for the lens. To create lens with more than one zoom position (configuration), just change the total number of zoom positions in this field.

Typical lens in
Surface Data
Spreadsheet

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes
Lens: DEMO TRIPLET				Zoom	1 of 1	Efl	50.000541
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	3.6397e+19		AIR		
1	21.250000	2.000000	6.500000	K	SK16	C	
2	-158.650000	6.000000	6.500000	P	AIR		
3	-20.250000	1.000000	5.000000		F4	C	
AST	19.300000	6.000000	5.000000	A	AIR		
5	141.250000	2.000000	6.500000		SK16	C	
6	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	18.170326	S			

When the number of configurations is increased to a number greater than 1, these multi-configuration buttons appear

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes
Lens: DEMO TRIPLET							
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE	RADIUS	GLASS	SPECIAL	
OBJ	0.000000	1.0000e+20	3.6397e+19		AIR		
1	21.250000	2.000000	6.500000	K	SK16	C	
2	-158.650000	6.000000	6.500000	P	AIR		
3	-20.250000	1.000000	5.000000		F4	C	
AST	19.300000	6.000000	5.000000	A	AIR		
5	141.250000	2.000000	6.500000		SK16	C	
6	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	18.170326	S			

To change the current zoom position, you can click on the left (<<) and right (>>) arrow buttons, or just type in a new current zoom position number. Once you change the current zoom position, you can edit a surface radius, thickness, aperture, or glass data for that configuration directly in the surface data spreadsheet. For any analysis that is performed on your lens system when it is in an alternate configuration (configuration >1), the generated analysis output will indicate the current configuration number.

When the current configuration is changed to a number greater than 1, you have the ability to make changes directly to certain surface parameters.

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes
Lens: DEMO TRIPLET							
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE	RADIUS	GLASS	SPECIAL	
OBJ	0.000000	1.0000e+20	3.6397e+19		AIR		
1	21.250000	2.000000	6.500000	K	SK16	C	
2	-158.650000	8.000000	6.500000	P	AIR		
3	-20.250000	1.000000	5.000000		F4	C	
AST	19.300000	6.000000	5.000000	A	AIR		
5	141.250000	2.000000	6.500000		SK16	C	
6	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	20.576241	S			

Configuration data is displayed for all alternate system configurations. When more than one configuration is defined for the optical system, the system can be switched to an alternate configuration by using the Calculate >> Setup Cfg/ Wavelength command. Output generated when the system is in an alternate configuration will indicate the current configuration number

When you change to a zoom position greater than 1, note that some of the spreadsheet buttons disappear. You no longer have access to items in the SPECIAL and SRF data columns. This means that you must use a different method to access these items in different zoom positions.

When the current configuration is changed to a number greater than 1, the ability to make some changes directly from the spreadsheet disappear.

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes
Lens: DEMO TRIPLET							
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE	RADIUS	GLASS	SPECIAL	
OBJ	0.000000	1.0000e+20	3.6397e+19		AIR		
1	21.250000	2.000000	6.500000	K	SK16	C	
2	-158.650000	8.000000	6.500000	P	AIR		
3	-20.250000	1.000000	5.000000		F4	C	
AST	19.300000	6.000000	5.000000	A	AIR		
5	141.250000	2.000000	6.500000		SK16	C	
6	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	20.576241	S			

To edit this type of data in different zoom positions (configurations), you must click on the “Zoom” button. This will bring up the Configuration Data spreadsheet, where you can enter any allowable configuration data.

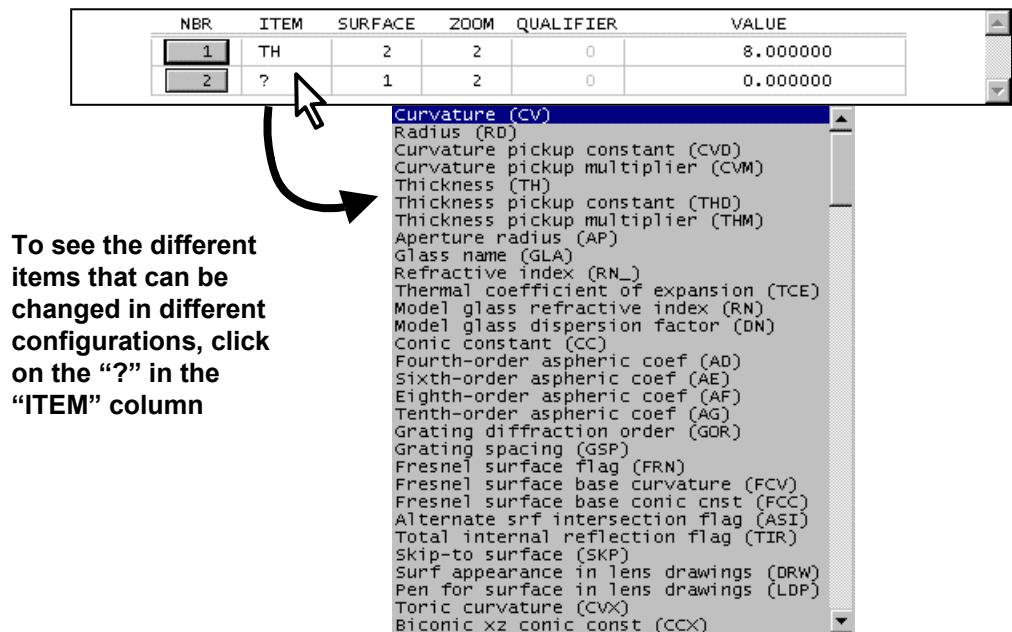
To make changes in different configurations, clicking on the “Zoom” button opens the Configuration Data editing spreadsheet

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes
Lens: DEMO TRIPLET							
Ent beam radius		6.250000	Field angle	9.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	3.6397e+19	AIR			
1	21.250000	2.000000	6.500000	SK16			
2	-158.650000	8.000000	6.500000	AIR			
NBR	ITEM	SURFACE	ZOOM	QUALIFIER	VALUE		
1	TH	2	2	0	8.000000		
IMS	0.000000	0.000000	20.576241	S			

Each row in the spreadsheet corresponds to one item of configuration data. Note that any changes you have previously made in different zoom positions is recorded here in the Configuration Data editor. This spreadsheet is just like other spreadsheets in OSLO, so you have control over adding, deleting, copying and pasting rows of data.

The **first column** just contains the configuration item number, which is on a Row Button in the number (NBR) column. You can select a row for editing by clicking on the Row Button in the first column. Once you have an row selected, a range of rows may be selected by clicking on the item where you want the range to end. The range may extend to either preceding or subsequent (but not both) rows from the initially selected row. Clicking on the CANCEL button will cancel the current selection. After selecting one or more rows, right clicking the mouse on the selected rows will show a pop-up editing menu containing the items “Cut”, “Copy”, “Paste”, “Reverse”, “Insert Before”, and “Insert After”. You can “Cut” or “Copy” the current selection to a clipboard, and then “Paste” it back in the spreadsheet at a different location. You can “Delete” the row selection if you wish, or “Insert” a new row above the selected rows. Also, the current selected rows can be “Reversed”.

The **second column** contains the identifier for the configuration item (curvature, thickness, etc.). The item may be entered directly by typing in the name of the item, or, if you activate the item cell by clicking on it, a pop-up list containing all of the allowed configuration items will be displayed.



To see the different items that can be changed in different configurations, click on the “?” in the “ITEM” column

You can choose the desired item from the list. Note that the data type for configuration data does not have to be present in the base configuration or in other configurations. For example, in a system with 4 configurations, a surface may be aspheric in configuration 3, but not in configurations 1, 2, and 4. The identifiers for allowed configuration data items are given in the following table. In the table, an italicized *i* stands for an integer value (1, 2, 3, etc.).

The refractive index and dispersion factor for model glasses are also configuration data items. Note that these items are valid configuration data items only if the medium defined for the surface is a model glass.

More information on model glasses may be found in the Optics Reference.

Cfg item	Description
cv	curvature
rd	radius of curvature (to avoid possible overlap, stored as <i>curvature</i>)
th	thickness
ap	aperture radius
gla	glass name
rni	refractive index at wavelength <i>i</i> (direct index or model glass only)
tce	thermal coefficient of expansion
cc	conic constant
ad	4 th order aspheric constant
ae	6 th order aspheric constant

af	8 th order aspheric constant
ag	10 th order aspheric constant
gor	grating diffraction order
gsp	grating spacing
frn	Fresnel surface flag
fcv	Fresnel surface substrate curvature
fcc	Fresnel surface substrate conic constant
asi	alternate surface intersection flag
tir	total internal reflection only surface flag
skp	skip to surface
cvx	toric curvature
ccx	x-z conic constant (x-toric and biconic surfaces)
dt	decenter-tilt order flag
dcx	<i>x</i> decentration
dcy	<i>y</i> decentration
dcz	<i>z</i> decentration
tla	tilt around <i>x</i> -axis
tlb	tilt around <i>y</i> -axis
tlc	tilt around <i>z</i> -axis
asp	aspheric surface type
asi	aspheric surface coefficient <i>i</i>
asai	ISO aspheric surface coefficient <i>i</i>
asbi	ISO aspheric surface coefficient <i>i</i>
asci	ISO aspheric surface coefficient <i>i</i>
asdi	ISO aspheric surface coefficient <i>i</i>
cns	cone surface coefficient
cnx	asymmetric cone surface <i>x</i> -coefficient
cny	asymmetric cone surface <i>y</i> -coefficient
spl	spline surface
shi	spline surface zone height <i>i</i>
ssi	spline surface slope <i>i</i>
doe	diffractive surface type
dwv	diffractive surface design wavelength
dfi	diffractive surface phase coefficient <i>i</i>
dor	diffractive surface diffraction order

zoe	Zernike surface type
zri	Zernike surface coefficient i
hv1	hologram source point 1 real/virtual factor
hv2	hologram source point 2 real/virtual factor
hor	hologram diffraction order
hwv	hologram construction wavelength
hx1	hologram source point 1 x -coordinate
hy1	hologram source point 1 y -coordinate
hz1	hologram source point 1 z -coordinate
hx2	hologram source point 2 x -coordinate
hy2	hologram source point 2 y -coordinate
hz2	hologram source point 2 z -coordinate
usr	user ray trace surface
unm	user surface CCL ray trace command name
uti	user ray trace surface coefficient i
pfl	perfect lens focal length
pfm	perfect lens magnification for perfect imagery
jaa	amplitude of polarization element J_A
jpa	phase (in degrees) of polarization element J_A
jab	amplitude of polarization element J_B
jpb	phase (in degrees) of polarization element J_B
jac	amplitude of polarization element J_C
jpc	phase (in degrees) of polarization element J_C
jad	amplitude of polarization element J_D
jpd	phase (in degrees) of polarization element J_D
apn	number of special apertures
ay1	special aperture minimum y -coordinate
ay2	special aperture maximum y -coordinate
ax1	special aperture minimum x -coordinate
ax2	special aperture maximum x -coordinate
aan	special aperture orientation angle
atp	special aperture type
aac	special aperture action
ebr	entrance beam radius

nao	numerical aperture in object space
obh	object height
ang	object angle
tem	temperature (degrees C)
pre	pressure (atmospheres)
ast	aperture stop surface number
rfs	reference surface number
ims	image surface number
afo	evaluation mode
amo	aberration mode
warm	wide-angle ray aiming mode
xarm	extended aperture ray aiming mode
tele	telecentric entrance pupil mode
wvi	wavelength i
wwi	wavelength weight i
gdt	GRIN medium type
dth	GRIN medium ray trace step size
nz1	GRIN medium z coefficient
nz2	GRIN medium z^2 coefficient
nz3	GRIN medium z^3 coefficient
nz4	GRIN medium z^4 coefficient
nr1	GRIN medium r^2 or r'^2 coefficient
nr2	GRIN medium r^4 or r'^4 coefficient
nr3	GRIN medium r^6 or r'^6 coefficient
nr4	GRIN medium r^8 or r'^8 coefficient
gww	reference wavelength for Gradium gradient
gmz	Gradium blank thickness
goz	Gradium axial offset into blank
gnzi	Gradium z^i coefficient
grai	Gradium gradient α_i coefficient
grbi	Gradium gradient β_i coefficient
grci	Gradium gradient χ_i coefficient
grdi	Gradium gradient δ_i coefficient

nrx	elliptical gradient x-coefficient
nry	elliptical gradient y-coefficient
sgc	spherical gradient center of symmetry distance
sva	radial gradient sinusoidal variation amplitude
svp	radial gradient sinusoidal variation period
svf	radial gradient sinusoidal variation phase
tas	radial gradient linear taper slope
tao	radial gradient linear taper offset
ugc	user gradient CCL ray trace command name
ugri	user GRIN coefficient <i>i</i>
rn	refractive index
dn	dispersion factor

Note: Configuration items that are only available in OSLO Premium are shown in shaded cells in the table.

The **third column** in the spreadsheet is the surface number for the configuration item. This entry is not needed for system data configuration items (**ebr**, **nao**, **obh**, **ang**, **tem**, **pre**, **ast**, **rfs**, **ims**, **afo**, **amo**, **warm**, **xarm**, **tele**, **wvi**, **wwi**).

The **fourth column** is the configuration number for which the data is to be used. You are defining the *differences* from the base configuration (configuration 1), so the minimum value for configuration number that you can enter is 2.

The **fifth column** is a qualifier that is required for some configuration items. For **asp** (aspheric surface), **doe** (diffractive surface), and **gdt ugr** (user-defined gradient) it is the maximum order of the appropriate power series expansion. For the special aperture data (**ay1**, **ay2**, **ax1**, **ax2**, **aan**, **atp**, and **aac**), the qualifier is the identification for the special aperture, i.e., “A”, “B”, etc. For the gradient index medium coefficients (**nz1**, **nz2**, **nz3**, **nz4**, **nr1**, **nr2**, **nr3**, **nr4**), the qualifier is the wavelength number for the coefficient, i.e., “wv”. The qualifier is not needed for other configuration data items.

The **sixth column** in the spreadsheet is the value of the data item in the specified configuration.

If you want to leave the spreadsheet and define the displayed data to be the configuration data, click OK (green check). The configuration items will be sorted by configuration number and checked for validity. Any invalid configuration data requests will be displayed in the text window.

EFL (Effective Focal Length)

This is a display-only field that shows the effective focal length of the current lens. This is provided so that you can see how the focal length changes as you enter lens data. The field is grayed out when there are special data that prevent the computation of focal length.

Entrance Beam Radius (System Aperture)

For lenses working with an infinitely distant object (OSLO uses a value of 1.0×10^{20} to indicate an infinite distance), the entrance beam radius is displayed below the lens name. The entrance beam radius is defined as the radius of the axial beam of light entering the lens, measured at surface 1. For finite conjugate systems, the numerical aperture in object space is displayed. Numerical aperture is defined as the product of refractive index and the sine of the (half) cone angle of the beam. Thus, for a system operating in air, the numerical aperture has a value between 0 and 1. The value of the appropriate quantity may be changed by selecting the cell and typing in the desired value. The button located at the right-hand side of the second line of the spreadsheet provides access to the paraxial setup spreadsheet, where you will find further options for designating the working aperture of the lens (e.g., f -number).

Field angle (Field of View)

For lenses with an infinitely distant object, the field angle is displayed. This is the half angle, in degrees, subtended by the object at the entrance pupil. For finite conjugate systems, the object height is displayed. This is the distance from the optical axis to the edge of the object. The value of the appropriate quantity may be changed by selecting the cell and typing in the desired value. The button located at the right-hand side of the second line of the spreadsheet provides access to the paraxial setup spreadsheet, where you will find further options for designating the field of view of the lens.

Primary Wavln (Wavelength)


The current value of the primary wavelength (wavelength1) is displayed and editable from this field. Note that wavelength values are always entered and displayed in μm . The value of the wavelength may be changed by selecting the cell and entering the new value. Alternatively, if you activate the wavelength cell by clicking on it, a list of common spectroscopic and laser lines will be presented, from which you can select the desired wavelength, if appropriate. If more than one wavelength is defined, the button at the right-hand end of the field will display an asterisk (*). Clicking this button provides access to the wavelengths spreadsheet editor for more comprehensive wavelength editing features (e.g., inserting and deleting wavelengths, editing wavelength weights).

Surface data entry

Below the fixed area of several spreadsheets is a scrolled area that contains a variable number of rows, each having a button at the left-hand end (whether or not the area needs to be scrolled depends on how many rows are defined, relative to the vertical extent of the spreadsheet window). Although the surface data spreadsheet is the most conspicuous example of a spreadsheet with a scrolled area, there are many others (wavelengths, field points, variables, etc.) and they all work similarly. Each is a form of object editor, where the objects are the entities described by the data in the rows. For example, in the surface data spreadsheet, the objects are lens surfaces. The spreadsheet allows you to manipulate the objects by selecting the rows of interest and then performing the desired action.

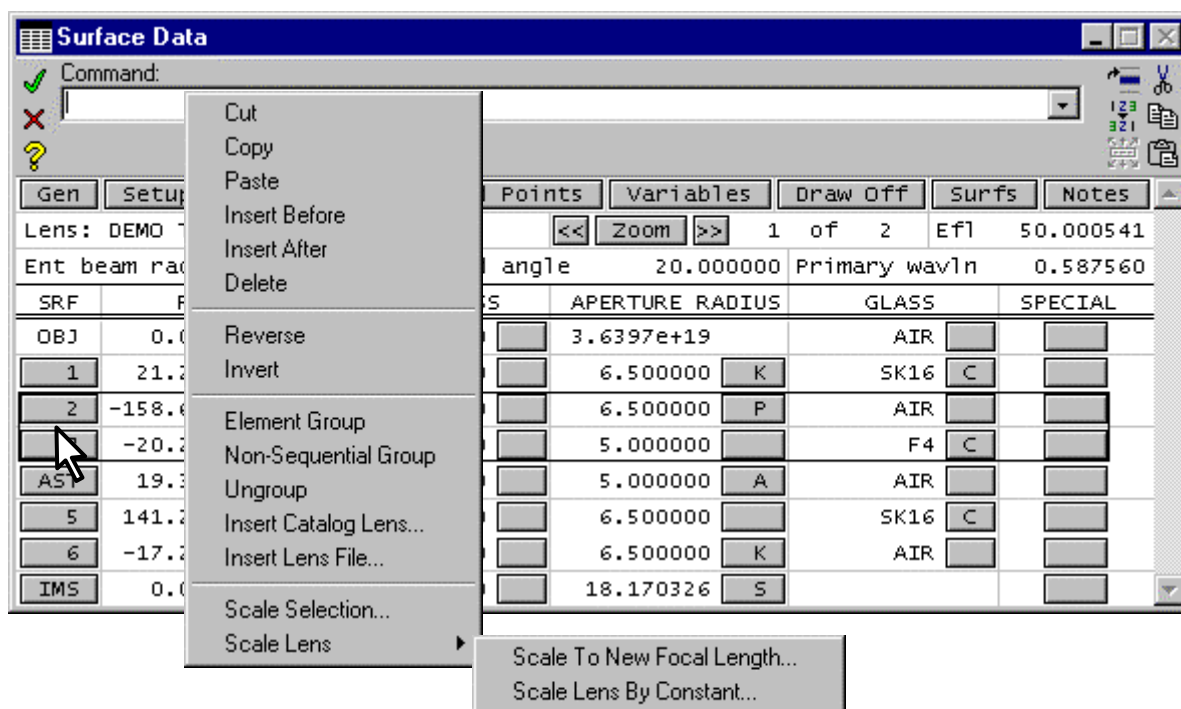
Row Buttons

Each surface in your lens is represented by a row of data in the spreadsheet. The surfaces are numbered sequentially on the **row buttons**, in the order in which light traverses the system. The object is surface number 0 and is always noted as “OBJ” in the spreadsheet. The highest numbered surface is called the image surface and is noted “IMS”. The Aperture Stop surface is noted “AST”.

You can select a surface for editing by clicking on the row button for that surface. Once you have a surface selected, a range of surfaces may be selected by clicking on the surface where you want the range to end. The range may extend to either preceding or subsequent (but not both) surfaces from the initially selected base surface. Clicking the CANCEL button (the red “X” button  in the upper left corner of the spreadsheet) will cancel the current surface selection.

After selecting one or more surfaces, you will note that all of the items on the Edit menu are active, along with the Cut, Copy, Paste, Reverse, Insert Before, and Insert After toolbar buttons. This means that the toolbar editing functions described in section “Toolbar Icons” on page 116 can be used on the current selection. You can Cut or Copy the current selection to a clipboard, and then Paste it back in the lens at a different location. You can Delete the selected surfaces if you wish, or Insert a new surface before or after the selected range. Also, the current selection can be Reversed or Inverted. If you want to search the catalog databases for an appropriate lens, Insert Catalog Lens will invoke the database. Insert Lens File allows you to insert the surface data from another lens file into the current lens. You can “Group” the selected range of surfaces for a desired element, or you can “Ungroup” a previously defined element. See “Toolbar Icons” on page 116 for a complete discussion of these editing functions.

Right clicking with the mouse on the row buttons brings up a pop-up menu that allows you to perform various tasks:



Cut

The Cut menu command is often used to move an element from one location to another in a lens. Cut deletes the current selection and saves it in the *clipboard*. The Cut and Copy commands save data in the clipboard, and the Paste command inserts the contents of the clipboard (if any) before the current selection.

To use the Cut command, first select a surface or a range of surfaces as described above, then cut the selected lines by choosing Cut from the Edit menu (or, alternatively, by clicking the Cut button on the toolbar, or by pressing CTRL+X). The selected surfaces are deleted from the spreadsheet and saved in the clipboard.

Copy

The Copy menu command saves the current selection in the clipboard without deleting it. Copy changes only the contents of the clipboard; the Surface Data are not affected. The Copy command is invoked by first selecting a surface or a range of surfaces and then choosing Copy from the Edit menu (or, alternatively, by clicking the mouse on the Copy button on the toolbar, or by pressing CTRL+C).

Paste

When working with the Surface Data Spreadsheet, the contents of the clipboard are in a special format that can be interpreted only by OSLO and can not be pasted into other programs. However, it can be pasted into other instances of OSLO.

The Paste menu command inserts the current clipboard contents into the spreadsheet before the first line of the current selection. Before Paste can be used, two conditions must be satisfied: the clipboard must contain some data (resulting from use of the Cut or Copy command), and one or more rows must be selected. The Paste command can be invoked from the Edit menu (or, alternatively, by clicking the Paste toolbar button, or by pressing CTRL+V).

Suppose that you wish to move the middle (negative) element of a Cooke triplet to the front of the lens. This can be done in four steps. First, select the middle element (surfaces 3 and 4).

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: DEMO TRIPLET				<< Zoom >>	1 of 2	Efl	50.000541
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	3.6397e+19	AIR			
1	21.250000	2.000000	6.500000	K	SK16	C	
2	-158.650000	6.000000	6.500000	P	AIR		
3	-20.250000	1.000000	5.000000		F4	C	
AST	19.300000	6.000000	5.000000	A	AIR		
5	141.250000	2.000000	6.500000		SK16	C	
6	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	18.170326	S			

Next, use the Cut menu command to and save it in the clipboard.

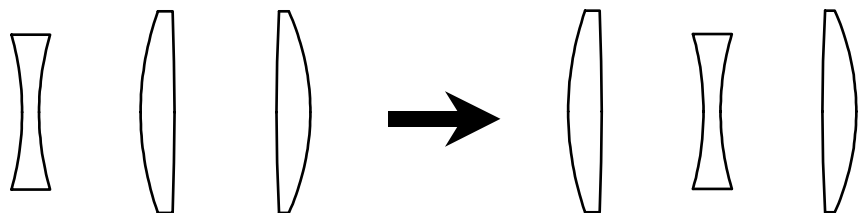
Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: DEMO TRIPLET		<< Zoom >>		1 of 2	Efl	16.071647	
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	3.6397e+19	AIR			
1	21.250000	2.000000	6.500000	K	SK16	C	
2	-158.650000	6.000000	6.500000	P	AIR		
AST	141.250000	2.000000	6.500000	A	SK16	C	
4	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	32.280068	S			

Then select the surface in front of which the negative element is to be inserted (surface 1).

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes
Lens: DEMO TRIPLET		<< Zoom >>		1 of 2	Efl	16.071647	
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	3.6397e+19	AIR			
1	21.250000	2.000000	6.500000	K	SK16	C	
2	-158.650000	6.000000	6.500000	P	AIR		
AST	141.250000	2.000000	6.500000	A	SK16	C	
4	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	32.280068	S			

Finally, use the Paste command to insert the clipboard contents before the selected surface:

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes
Lens: DEMO TRIPLET		<< Zoom >>		1 of 2	Efl	22.127061	
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	3.6397e+19	AIR			
1	-20.250000	1.000000	5.000000		F4	C	
AST	19.300000	6.000000	5.000000	A	AIR		
3	21.250000	2.000000	6.500000	K	SK16	C	
4	-158.650000	6.000000	6.500000	P	AIR		
5	141.250000	2.000000	6.500000		SK16	C	
6	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	11.506673	S			



Insert Before/After

The Insert Before or After commands add an empty line before or after the selected row in a spreadsheet. To insert a line, first select the surface before or after which the new line is to be inserted, then use the Insert Before or Insert

After command according to where you want the inserted line relative to the selected row. Alternately, you can select the Insert Before or Insert After toolbar buttons at the top of the spreadsheet. In the Surface Data spreadsheet, inserted surfaces are air spaces with zero curvature and thickness, and a solved aperture radius.

To insert multiple rows quickly, highlight the button for the row after which you wish to insert the new row(s), then press SHIFT+SPACEBAR. Pressing SHIFT+SPACEBAR allows you to insert multiple surfaces easily, because it is not necessary to select a row or range of rows before the new line is inserted. As an alternate, you can click on a row button while holding down the shift key.

Delete

The Delete menu command acts similarly to the Cut command, except that the selected lines are not saved in the clipboard. The previous contents of the clipboard (if any) are not affected by the Delete command. The Delete command can be invoked by pressing the DELETE key or CTRL+D.

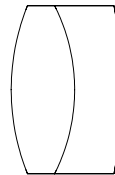
Reverse

The Reverse and Invert menu commands are used to change the orientation of selected surfaces. The Reverse command takes the selected range of surfaces and turns them 180 degrees as a group. This action is best illustrated by an example: suppose that you have designed a cemented doublet with *crown-in-front orientation*, and you wish to test its performance in a different application using the reverse, flint-in-front orientation.

The Reverse command will not work for ranges that include tilted/decentered surfaces, gradient-index elements, or other asymmetric special data.

The Reverse command will not work for ranges that include tilted/decentered surfaces, gradient-index elements, or other asymmetric special data.

The original design has the crown (positive) element on the object side:



To reverse the doublet, select its three surfaces.

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Cemented Doublet				Zoom	1 of 1	Efl	19.991886
Ent beam radius		4.500000	Field angle	5.7296e-05	Primary wavln		0.587560
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	1.0000e+14	AIR			
AST	12.070000	3.390000	5.000000	BALKN3			
2	-9.990000	2.500000	5.000000	SF15			
3	-25.840000	0.000000	5.000000	AIR			
IMS	0.000000	0.000000	3.892150	S			

Caution - Reverse deletes any pickups or solves on the selected surfaces. The pickups and solves are deleted because it is generally necessary to change them on the reversed surfaces; OSLO, however, cannot anticipate the necessary corrections.

then invoke the Reverse command:

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Cemented Doublet				Zoom	1 of 1	Efl	19.991886
Ent beam radius		4.500000	Field angle	5.7296e-05	Primary wavln		0.587560
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	1.0000e+14	AIR			
AST	25.840000	2.500000	5.000000	SF15			
2	9.990000	3.390000	5.000000	BALKN3			
3	-12.070000	0.000000	5.000000	AIR			
IMS	0.000000	0.000000	4.223404	S			

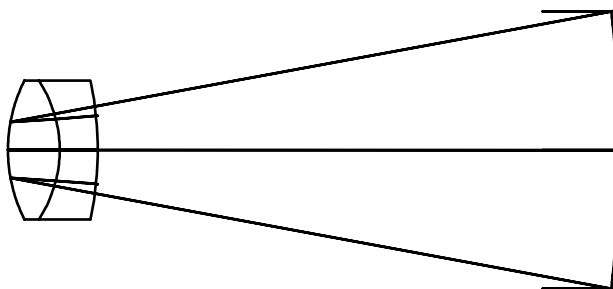
The doublet has been reversed 180 degrees:



Invert

The Invert menu command is typically used to turn a group of surfaces by 180 degrees after a mirror (or an odd number of mirrors) that has been inserted before the surfaces. Although a lens that has been inverted may appear in drawings as if it has simply been reversed, inversion differs significantly from reversal. Whereas reversal changes the order of the selected surfaces as well as their orientations, inversion simply changes the signs of the curvatures, aspheric coefficients, and thicknesses of the selected surfaces.

Again, inversion is best illustrated by an example: suppose that you wish to insert a mirror before the doublet used in the example above. If the surfaces after the mirror are not inverted, rays traced through the system will travel from left to right after the first surface of the doublet:



To get the rays to pass from right to left after reflection, select all the surfaces after the mirror...

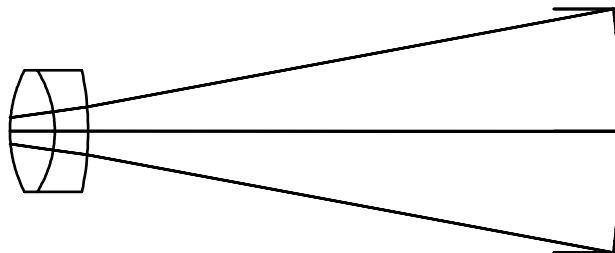
Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Cemented Doublet				Zoom 1 of 1	Efl -137.092083		
Ent beam radius 10.000000		Field angle 5.7296e-05		Primary wavln 0.587560			
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	1.0000e+14	AIR			
1	-100.000000	-40.000000	10.000200	S REFL_HATCH			
AST	25.840000	F 2.500000	5.000000	AF SF15	F		
3	9.990000	F 3.390000	5.000000	F BALKN3	F		
4	-12.070000	F 0.000000	5.000000	F AIR			
IMS	0.000000	0.000000	2.629750	S			

and use the Invert command to invert them:

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Cemented Doublet				Zoom 1 of 1	Efl -36.626704		
Ent beam radius 10.000000		Field angle 5.7296e-05		Primary wavln 0.587560			
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	1.0000e+14	AIR			
1	-100.000000	-40.000000	10.000200	S REFL_HATCH			
AST	-25.840000	F -2.500000	5.000000	AF SF15	F		
3	-9.990000	F -3.390000	5.000000	F BALKN3	F		
4	12.070000	F 0.000000	5.000000	F AIR			
IMS	0.000000	0.000000	1.124421	S			

Invert, like Reverse, deletes any pickups or solves from the selected surfaces.

The rays now travel in the desired direction:



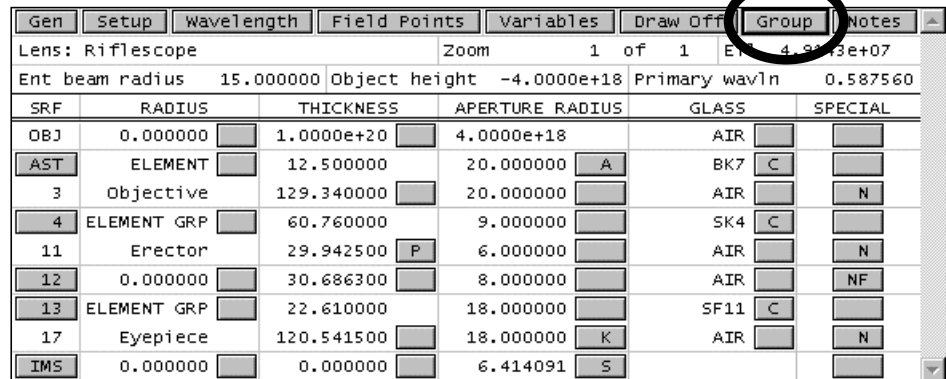
Element Group

The Element Group command collects the selected range of surfaces into a single element group. Although grouping surfaces does not change their optical properties, grouping multiple surfaces into elements can often provide a clearer view of components in a lens system. Lenses from the Catalog Lens Database are automatically entered as element groups. You can also define groups in other lenses.

To define a group, select a range of surfaces and execute the Element Group command on the Edit menu. If a 10-character note is attached to the final surface of the group, it will appear in the RADIUS column of the spreadsheet. If the group contains internal glass-to-air surfaces, it will be labeled an ELEMENT GRP, otherwise it will be labeled an ELEMENT. Element groups are not hierarchical, i.e. a group can not contain another group.

Element grouping primarily affects the view in the surface data spreadsheet. The only other part of the program that uses element groups is the single ray trace, where surface output is shown in groups.

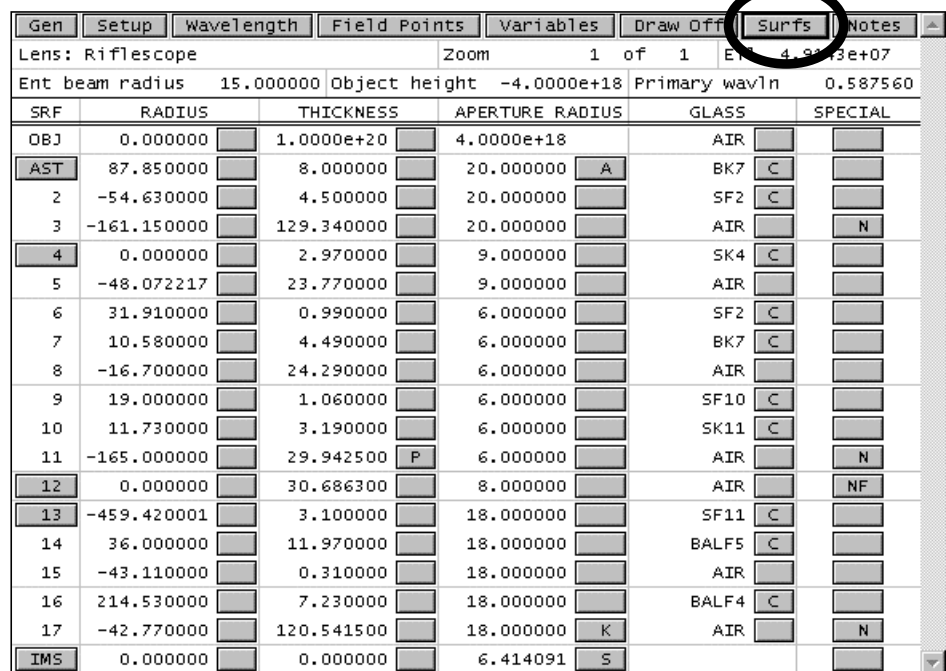
An example of a user-defined group is the **riflscop.len** file in the ".../public/len/demo/light/" directory. The figure below shows the Surface Data Spreadsheet for this lens with the Surfs/Group view button set to Group.



Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes
Lens: Rifle scope		Zoom 1 of 1		Ent 4.943e+07			
Ent beam radius 15.000000		Object height -4.0000e+18		Primary wavln 0.587560			
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	4.0000e+18	AIR			
AST	ELEMENT	12.500000	20.000000	BK7	C		
3	Objective	129.340000	20.000000	AIR		N	
4	ELEMENT GRP	60.760000	9.000000	SK4	C		
11	Erector	29.942500	6.000000	AIR		N	
12	0.000000	30.686300	8.000000	AIR		NF	
13	ELEMENT GRP	22.610000	18.000000	SF11	C		
17	Eye piece	120.541500	18.000000	AIR		N	
IMS	0.000000	0.000000	6.414091	S			

Note that only 8 rows are shown in the spreadsheet, although there are a total of 18 surfaces. Each group has a single row button, and selecting that button selects the entire group.

If the Surfs/Group button is set to Surfs, the spreadsheet appears as follows.



Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Rifle scope		Zoom 1 of 1		Ent 4.943e+07			
Ent beam radius 15.000000		Object height -4.0000e+18		Primary wavln 0.587560			
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	4.0000e+18	AIR			
AST	87.850000	8.000000	20.000000	BK7	C		
2	-54.630000	4.500000	20.000000	SF2	C		
3	-161.150000	129.340000	20.000000	AIR		N	
4	0.000000	2.970000	9.000000	SK4	C		
5	-48.072217	23.770000	9.000000	AIR			
6	31.910000	0.990000	6.000000	SF2	C		
7	10.580000	4.490000	6.000000	BK7	C		
8	-16.700000	24.290000	6.000000	AIR			
9	19.000000	1.060000	6.000000	SF10	C		
10	11.730000	3.190000	6.000000	SK11	C		
11	-165.000000	29.942500	6.000000	AIR		N	
12	0.000000	30.686300	8.000000	AIR		NF	
13	-459.420001	3.100000	18.000000	SF11	C		
14	36.000000	11.970000	18.000000	BALF5	C		
15	-43.110000	0.310000	18.000000	AIR			
16	214.530000	7.230000	18.000000	BALF4	C		
17	-42.770000	120.541500	18.000000	AIR		N	
IMS	0.000000	0.000000	6.414091	S			

Prm Non-sequential group

A non-sequential group is a special type of group that serves as a container for a range of surfaces that rays strike in some order that can not be determined at the outset. The actual ordering of surfaces in a non-sequential group depends on the ray trajectory, and is generally different for different rays. The first surface of a

non-sequential group is called the *entry port*, and the last surface is called the *exit port*. Once a ray passes through the entry port, it strikes anywhere from none to all surfaces in the group, possibly repeatedly and from either side, and eventually (possibly) strikes the exit port. If it strikes the exit port, it emerges from the group and thereafter strikes the remaining surfaces in sequential order.

Non-Sequential groups are different from element groups, but are created in a similar manner. To create a non-sequential group, select a range of surfaces in the Surface data spreadsheet, and execute the Non-Sequential Group command on the Edit menu.

If the Surfs/Group view is set to Group, the group will appear similar to the element group, excepting the title NON SEQ GRP in the RADIUS column.

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Group	Notes	
Lens: Non-Sequential Example			Zoom	1 of 1	Efl	1.0000e+54		
Ent beam radius		1.000000	Field angle	5.7296e-05	Primary wavln		0.587560	
SRF	RADIUS		THICKNESS		APERTURE RADIUS		GLASS	SPECIAL
OBJ	0.000000		1.0000e+20		1.0000e+14		AIR	
1	0.000000		0.000000		1.000000		S	AIR
AST	NON SEQ GRP				1.000000		AS	AIR
3			0.000000		1.000000			AIR
IMS	0.000000		0.000000		1.000000		S	

However, if the view is set to Surfs, the row buttons in the group will have outline boxes as shown in the figure below.

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Non-Sequential Example			Zoom	1 of 1	Efl	1.0000e+54	
Ent beam radius		1.000000	Field angle	5.7296e-05	Primary wavln		0.587560
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	1.0000e+14	AIR			
1	0.000000	0.000000	1.000000	S	AIR		
AST	0.000000		1.000000	AS	AIR		
3	0.000000	0.000000	1.000000		AIR	C	
IMS	0.000000	0.000000	1.000000	S			

Selected Row →

Selected
Row →

The outline boxes indicate that the rows in the group can be selected, so that rows can be inserted, deleted, etc. in a non-sequential group, as shown in the following figure, obtained by selecting the Insert After toolbar button for the spreadsheet shown above.

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Non-Sequential Example			Zoom	1 of 1	Efl	1.0000e+54	
Ent beam radius		1.000000	Field angle	5.7296e-05	Primary wavln		0.587560
SRF	RADIUS		THICKNESS	APERTURE RADIUS		GLASS	SPECIAL
OBJ	0.000000		1.0000e+20		1.0000e+14	AIR	
1	0.000000		0.000000		1.000000 S	AIR	
AST	0.000000				1.000000 AS	AIR	
3	0.000000				0.000000	AIR	
4	0.000000		0.000000		1.000000	AIR	C
IMS	0.000000		0.000000		1.000000 S		

Ungroup

The ungroup command converts a group into a series of surfaces. The group can be the product of a previous Group command (see above) or it can be an element from the Catalog Lens Database. In the case of a non-sequential group, the surfaces become sequentially ordered, which may not make optical sense.

The Ungroup command is often used to convert a Catalog Lens Database element into a series of surfaces so that the surfaces can be edited. To Ungroup an element, first select it by clicking on its row button, then execute the Edit >> Ungroup command. The figure below shows a lens from the Catalog Lens Database before and after ungrouping. Notice that ungrouping a lens also removes any *fixed* designations that have been placed in the group.

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Melles Griot MGLAL005				Zoom	1 of 1	Efl	19.991886
Ent beam radius		5.000000	Field angle	5.7296e-05	Primary wavln		0.587560
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	1.0000e+14	AIR			
AST	12.070000	3.390000	5.000000	BALKN3			
2	-9.990000	2.500000	5.000000	SF15			
3	-25.840000	17.291408	5.000000	AIR			
IMS	0.000000	0.000000	1.9992e-05				

Gen	Setup	Wavelength	Field Points	Variables	Draw Off	Surfs	Notes
Lens: Melles Griot MGLAL005				Zoom	1 of 1	Efl	19.991886
Ent beam radius		5.000000	Field angle	5.7296e-05	Primary wavln		0.587560
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	1.0000e+14	AIR			
AST	12.070000	3.390000	5.000000	BALKN3			
2	-9.990000	2.500000	5.000000	SF15			
3	-25.840000	17.291408	5.000000	AIR			
IMS	0.000000	0.000000	1.9992e-05				

Insert Catalog Lens

The Insert Catalog Lens menu command is used to insert a lens element from the Catalog Lens Database in front of the first selected surface.

To insert a Catalog lens element, select one or more surfaces in the Surface Data spreadsheet, and invoke the Insert Catalog Lens command. The Catalog Lens Database window appears, allowing you to specify the element type, focal length range, and lens diameter range for the list of displayed lens elements. See “Catalog lens” on page 157 for a further description of the Catalog Lens Database.

Suppose that you wish to replace the middle element of the Demo Triplet lens (**demotrip.len** in the “.../public/len/demo/edu/” directory) with a similar element from the Catalog Lens Database. The middle element of the triplet has a diameter

of 10 mm and a focal length of roughly -16 mm. To replace the middle element, select surfaces 3 and 4 in the Surface Data spreadsheet:

Gen	Setup	Wavelength	Field Points	Variables	Draw On	Group	Notes
Lens: Demo Triplet 50mm f/4 20deg			Zoom	1 of 1	Efl	50.000541	
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	3.6397e+19	AIR			
1	21.250000	2.000000	6.500000 K	SK16	C		
2	-158.650000	6.000000	6.500000 P	AIR			
3	-20.250000	1.000000	5.000000	F4	C		
AST	19.300000	6.000000	5.000000 A	AIR			
5	141.250000	2.000000	6.500000	SK16	C		
6	-17.285000	42.950000	6.500000 K	AIR			
IMS	0.000000	0.000000	18.170326 S				

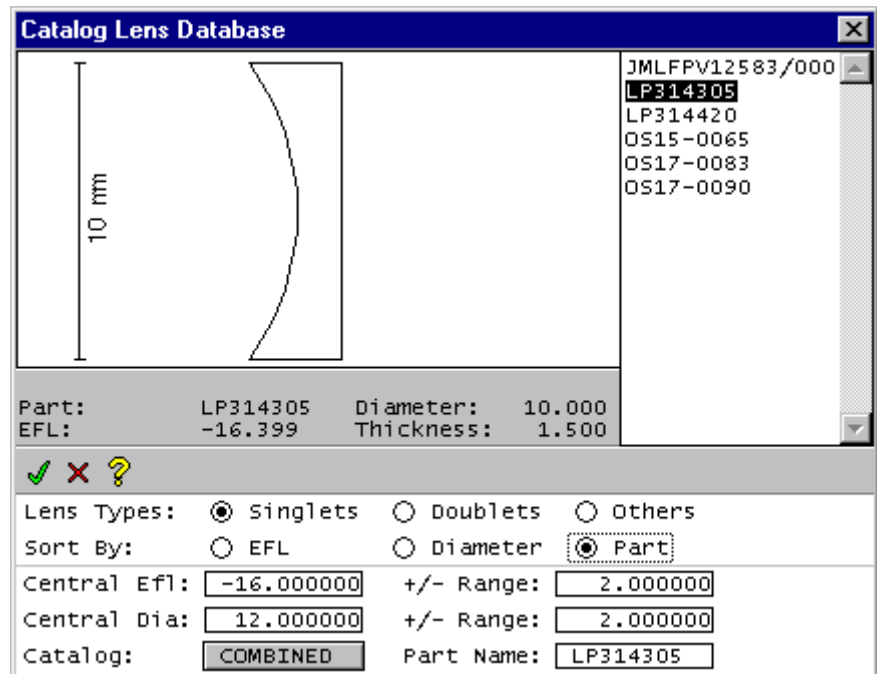
and use the Delete command to remove them:

Gen	Setup	Wavelength	Field Points	Variables	Draw On	Group	Notes	
Lens: Demo Triplet 50mm f/4 20deg				Zoom	1 of 1	Efl	16.071647	
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln		0.587560	
SRF	RADIUS		THICKNESS		APERTURE RADIUS		GLASS	SPECIAL
OBJ	0.000000		1.0000e+20		3.6397e+19		AIR	
1	21.250000		2.000000		6.500000	K	SK16	C
2	-158.650000		6.000000		6.500000	P	AIR	
AST	141.250000		2.000000		6.500000	A	SK16	C
4	-17.285000		42.950000		6.500000	K	AIR	
IMS	0.000000		0.000000		32.280068	S		

Since the new middle element is to be placed between surface 2 and surface 3 (formerly surface 5), select surface 3 in the spreadsheet:

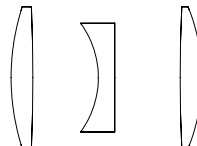
Gen	Setup	Wavelength	Field Points	Variables	Draw On	Group	Notes
Lens: Demo Triplet 50mm f/4 20deg			Zoom	1 of 1	Efl	16.071647	
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS		THICKNESS	APERTURE RADIUS		GLASS	SPECIAL
OBJ	0.000000		1.0000e+20		3.6397e+19	AIR	
1	21.250000		2.000000		6.500000 K	SK16 C	
2	-158.650000		6.000000		6.500000 P	AIR	
AST	141.250000		2.000000		6.500000 A	SK16 C	
4	-17.285000		42.950000		6.500000 K	AIR	
IMS	0.000000		0.000000		32.280068 S		

and invoke the “Insert Catalog Lens” command from the row button pop-up menu. The Catalog Lens Database will be displayed. Since we wish to replace the middle element with a catalog element with similar properties, we shall use the controls in the Catalog Lens Database window to restrict the set of lenses from which the replacement element will be chosen. First, use the Catalog button to select the “Combined” lens catalog so that we can choose from all of the catalog sets. Next, set the Central EFL to -16 , with a range of ± 2 , so that only elements with focal lengths between -18 and -14 will be listed. Finally, set the Central Diameter to 12 with a range of ± 2 , so that only elements with diameters between 10 and 14 will be displayed (we want the diameter of the replacement element to be at least as large as that of the original element).



Note that only 6 components meet the selection criteria. Part LP314305 has a focal length closer to that of the original middle element, so select it in the scrolled list and click on the Green Checkmark (OK) to insert it into your lens. Use the Aperture Radius options list button on surface 4 to place the aperture stop there (the Delete command moved the aperture stop to the last element). Change the thickness of surface 4 to 6.0 so that the position of the inserted element closely resembles that of the deleted element.

Gen	Setup	Wavelength	Field Points	Variables	Draw On	Group	Notes
Lens: Demo Triplet 50mm f/4 20deg		Zoom	1 of 1	Efl	46.675421		
Ent beam radius		6.250000	Field angle	20.000000	Primary wavln	0.587560	
SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPECIAL		
OBJ	0.000000	1.0000e+20	3.6397e+19	AIR			
1	21.250000	2.000000	6.500000	K	SK16	C	
2	-158.650000	6.000000	6.500000	P	AIR		
3	LP314305	1.500000	F	F	FIXED	F	
AST		6.000000	5.000000	AF	AIR		
5	141.250000	2.000000	6.500000		SK16	C	
6	-17.285000	42.950000	6.500000	K	AIR		
IMS	0.000000	0.000000	17.983382	S			



Insert Lens File

The Insert Lens File menu command is used to insert a lens file from disk in front of the first selected surface.

To insert a lens, select one or more surfaces in the Surface Data spreadsheet, and invoke the Insert Lens File command. The File Selection dialog box appears, allowing you to choose a lens file to merge into the current lens.

Only the surface data proper from the chosen lens are inserted into the current lens; the object and image surfaces from the chosen lens are not added to the current lens. In addition, the variables, configuration, and optimization data from the inserted lens are not used. This enables you to save error functions to use with different lenses.

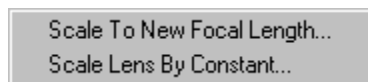
Saving error functions is only possible when the error function does not refer explicitly to any surfaces in the lens.

To save a standard error function for use with another lens, open a lens that has the desired error function, and delete all the surfaces. This leaves you with a null lens that has the standard error function. You can save this lens on the disk with some descriptive name. When you want to use this error function with some other lens, first open the error function file, then insert the lens into the error function.

Scale Surface

The Scale Surface command allows you to scale a range of surfaces by a given factor. All linear dimensions of the selected range are scaled by this factor.

Scale Lens

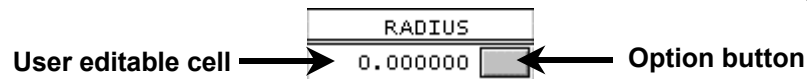


The Scale Lens command allows you to scale the focal length of the entire lens either by or to a given value. It is not necessary to select any surface range for this command. When you execute the command, an options box queries the type of scaling you want. Double click the selected option, then enter the appropriate value in the command line, following the prompt.

Radius of curvature

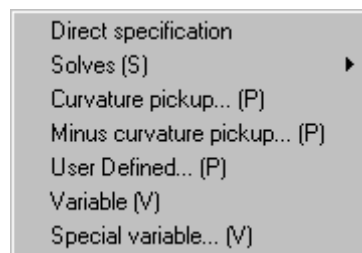
Next to the surface number is the Radius SmartCell. The cell displays the current value of the radius of curvature of the surface. (If you prefer to work with the curvature, i.e., the inverse of the radius of curvature, of surfaces rather than the radius of curvature, set the **Spreadsheet_curvature_mode (scvm)** preference to On before entering the surface data spreadsheet.) Each surface in the lens has a local, right-handed (x, y, z) coordinate system associated with it. The vertex of the surface is located at the origin of the coordinate system. Usually, the z-axis coincides with the optical axis of the lens, so the xy plane is tangent to a spherical surface at the origin. The center of curvature of the optical surface is located on the local z-axis. If the center of curvature lies along the positive z-axis (i.e., to the right in a system with no mirrors or tilted/decentered elements), then the radius of curvature is positive. Conversely, if the center of curvature is located on the negative z-axis, the radius is negative. The value of a radius of curvature is

changed by moving the cursor to the radius cell of the desired surface and typing the new value for the radius. A radius of 0 indicates that the surface is a plane.



In addition to specifying the radius directly, there are several other ways to set its value. The radius **options button**, which lies directly to the right of the radius value cell, serves two purposes. First, clicking the button presents you with a pop-up menu of possible ways to specify the radius. Second, if the radius has been set by some means other than a direct user specification, the button will be labeled with a letter indicating how the radius was set. (Also, when the radius options button is selected, a brief form of the radius-setting command will be displayed in the command line.)

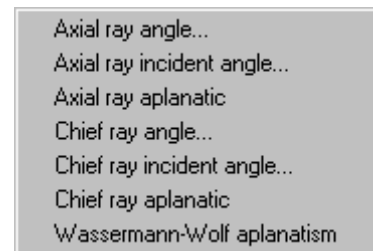
The options presented in the radius pop-up menu for setting radius of curvature are:



Direct specification (rd, cv)

This means that you have entered a prescribed value for the radius. Selecting this item will remove any existing solve or pickup data for the radius.

Solves



Axial ray angle solve (pu)

The radius of curvature will be set to a value such that the paraxial axial ray leaves the surface with a specified slope. This slope corresponds to the value of PU in the paraxial ray trace. This option might be used, for example, on the last surface of a lens to maintain focal length or *f*-number as other lens data items are changed.

Axial ray incident angle solve (pi)

The radius of curvature will be set to a value such that the paraxial axial ray strikes the surface at the specified angle. The angle is the one that corresponds to the value of PI in the paraxial ray trace.

Axial ray aplanatic solve (al)

The radius of curvature will be set to a value such that the paraxial axial ray satisfies the aplanatic condition. Under this condition, the surface introduces zero spherical aberration, and no third-order coma or astigmatism.

Chief ray angle solve (puc)

The radius of curvature will be set to a value such that the paraxial chief ray leaves the surface with a specified slope. This slope corresponds to the value of PUC in the paraxial ray trace. This option might be used, for example, to make the system telecentric, by forcing the paraxial chief ray to exit the last lens surface with a slope of zero, thereby forcing the exit pupil to infinity.

Chief ray incident angle solve (pic)

The radius of curvature will be set to a value such that the paraxial chief ray strikes the surface at the specified angle. The angle is the one that corresponds to the value of PIC in the paraxial ray trace.

Chief ray aplanatic solve (alc)

The radius of curvature will be set to a value such that the paraxial chief ray satisfies the aplanatic condition. Under this condition, the surface introduces zero spherical aberration for the chief ray.

The presence of a solve for a radius is indicated by an S printed on the radius options button. Radius of curvature solves can not be placed on the object or image surfaces. Also, angle and aplanatic solves can not be placed on dummy surfaces (i.e., surfaces that have the same refractive index on both sides), since rays do not refract at a dummy surface.

Prm Wassermann-Wolf aplanatism solve

The Wassermann-Wolf solve (WW) is a technique for designing two adjacent aspheric surfaces such that the optical system containing these two surfaces is aplanatic (i.e., all orders of spherical aberration are corrected and the Abbe sine condition is satisfied for all points in the aperture). Assuming that the construction parameters for all of the surfaces preceding and succeeding the two WW surfaces are all known, rays are traced from the axial object point to the tangent plane of the first WW surface and (backwards) from the desired axial image point to the tangent plane of the second WW surface. Given the ray and differential ray data on these two planes and the condition that the system is to be aplanatic, a system of two simultaneous first-order differential equations is formed. The solution of these equations (which must, in general, be done numerically) forms the definition of the two aspheric surfaces.

There are several assumptions that are made during the derivation of the WW differential equations.

1. The system is rotationally symmetric.
2. The two WW surfaces are adjacent, i.e., there are no intervening optical surfaces between the two WW aspheric surfaces.
3. Obviously, there can be only a single pair of WW solve surfaces in an optical system.
4. The media before the first WW surface, between the two WW surfaces, and after the second WW surface are all homogeneous.

5. The construction parameters of the optics before and after the WW surfaces are known. The practical implication of this is that no curvature or thickness solves are allowed after the WW surfaces.

To enter a WW solve in OSLO, click the "Radius" options button for the desired first WW surface and select "Solves>>Wassermann-Wolf aplanatism" from the pop-up menu. The first data item to specify is the number of points to use in the solution of the differential equations. As mentioned above, the equations are solved numerically and this value is the number of discrete points at which the solution will be found. (OSLO uses a combination of Runge-Kutta and Adams-Bashforth-Moulton algorithms in the solution.) The surfaces are represented as radial spline profile surfaces and the solution points of the equations are converted directly into spline zone heights and slopes.

Number of solution points for surfaces: <input type="text" value="4"/>	
Wassermann-Wolf aperture ray specification	
Object space	
Ray height on tangent plane to surface 1:	<input type="text" value="6.250000"/>
Numerical aperture of W-W aperture ray:	<input type="text" value="6.2500e-20"/>
Image space	
Ray height on tangent plane to surface 6:	<input type="text" value="5.403186"/>
Numerical aperture of W-W aperture ray:	<input type="text" value="0.124818"/>
Delete Wassermann-Wolf Solve	
Delete Solve and Spline Surfaces	

The other solve data item is the specification of the aperture ray for the aplanatic condition. This determines the size of the bundle of rays that will be used in solving the equations. The ray is specified by an object space value and an image space value; these values may either be ray heights on tangent planes (on surface one and the surface preceding the image surface) or numerical apertures. Generally, the ray height specification is recommended for an infinite conjugate and the numerical aperture specification for a finite conjugate. The WW aperture ray specification will be related to, but not necessarily the same as, the entrance beam radius/object numerical aperture and image space numerical aperture of the lens. For example, surfaces removed from the stop are generally larger than the size required by the axial ray bundle. Thus, the WW aperture ray should be outside the axial ray of the system so that the WW surfaces may be defined for points outside the region seen by the axial bundle.

If the two WW surfaces are not specified as spline surfaces when the WW solve is entered, they will be converted into spline surfaces with the number of spline zones equal to the number of WW solution points. However, you may designate either or both of the WW surfaces as spline surfaces with a number of zones larger than the number of WW solution points. The slopes of the zones outside the WW solution range may be set independently and/or designated as optimization variables. Be aware that in addition to the fact that the WW algorithm is specifying the inner range of spline zones, the usual requirements for spline surfaces apply (e.g., the zones must be unique and in ascending order), so care should be exercised when setting the values of the spline heights and zones outside the WW solution region.

There are two buttons at the bottom of the spreadsheet for deleting the WW solve specification. If you click the "Delete Wassermann-Wolf Solve" button on the left,

the solve specification will be deleted but the spline profiles of the two surfaces will be retained. If you click the "Delete Solve and Spline Surfaces" button on the right, both the solve specification and all of the spline data for the two surfaces will be deleted.

The nature of the WW differential equations is such that if a physical solution exists for the equation, it is unique. If the solve fails, it is usually because too few solution points have been specified. Because the equations are solved numerically, the solution is found at a discrete number of points. Between the points, the surfaces are found by interpolation (a cubic spline). This interpolant will not, in general, match the desired exact surface profile (except at the spline zone heights), so the ray-intercept curves (for example) will typically exhibit a zigzag pattern around the "true" value. Increasing the number of solution points will, in general, both decrease the magnitude and increase the frequency of the zigzag pattern.

Curvature pickup (pk cv)

The curvature of the surface is constrained to be the same as a preceding surface (with the optional addition of a constant).

Minus curvature pickup (pk cvm)

The curvature of the surface is constrained to be the negative value as that of a preceding surface (with the optional addition of a constant).

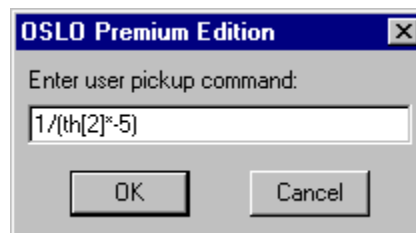
Pickups might be used, for example, to maintain symmetry between two parts of a lens system, or to represent the same surface traversed more than once (e.g., after reflection). The presence of a pickup for a radius is indicated by a P on the radius options button. If the source surface for the pickup has any conic, toric, aspheric, or spline surface data, that information will also be picked up. Thus, both the pickup source and destination surfaces will have the same surface profile. (Conic, toric, aspheric, and spline surfaces are described later in this chapter.)

Prm User Defined (ucp)

This pickup allows arithmetic operations with global data to be used to calculate the curvature. The command text may be any valid operation which returns a numeric value. This includes CCL functions which return a real value.

The pickup value is not checked for valid data so care should be exercised when using global surface data values especially when the value is for a surface defined after the surface with the pickup. The pickups are evaluated in surface order.

An example of a pickup entry follows:



Variable (vb ins)

The radius of curvature will be used as a variable during optimization. Note that a radius of curvature specified by a solve or pickup can not be a variable, since its value is constrained by the solve or pickup requirement.

Special variable

The radius of curvature will be used as a variable during optimization. Also, the variables spreadsheet editor will be opened to allow you to edit the controls (minimum value, maximum value, etc.) for the variable.

A variable radius of curvature is indicated by a V on the radius options button.

If the letter F appears on the options box (or next to the radius value for surfaces in a lens module), this means that the radius of curvature is “fixed” and can not be directly changed by the user. This condition is set for catalog lenses, which are entered in group mode. It is possible to ungroup a lens module by clicking the radius options button for the first surface of a module or by selecting the module and choosing Ungroup from the Edit menu. Ungrouping a module removes the fixed designation for surfaces in that module, allowing them to be changed.

Std **Prm** Test glass fitting

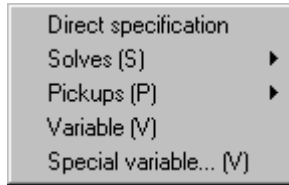
Note that this option is not available on the popup menu for the Surface Radius of Curvature.

If a list of test glass radii is available, the current radius can be changed to that of the closest test glass. This is done by selecting the radius value and then clicking twice with the SHIFT key held down, or pressing CTRL+PAGE UP twice. If you want to change the radius to this suggested test glass radius, click OK or press ENTER. A T will appear on the radius options button, indicating that the radius was chosen from a list of test glass radii. If you want to return to the original value, press ESCAPE.

After finding the closest test glass radius, you may scroll the list of test glass radii. If you want the next larger radius from the list, click in the spreadsheet above the number, or press the up arrow key. If you want the next smaller radius from the list, click in the spreadsheet below the number, or press the down arrow key. Press ENTER to confirm, or ESCAPE to cancel.

Test glass radii are contained in the *.tgl files in the directory **installation_directory\bin\lmo**. The first line of each file is the number of radii in the file. The subsequent lines in the file are the available test glass radii, listed in order of increasing radius of curvature. These files may be edited with a text editor, and the updated lists will be used the next time OSLO is started. The current test glass list is given by the **Test_glass_file (tglf)** preference.

Thickness



The thickness of a surface is the distance, measured along the z-axis, to the next surface. Thus, a positive thickness for surface i means you travel in the positive z direction to get from surface i to surface $i+1$. Conversely, a negative thickness means you travel in the negative z direction to get from surface i to surface $i+1$. The convention in OSLO is that the light originally travels in the positive z direction. For all-refracting systems (i.e., no mirrors) with no tilted or decentered elements, the light is always traveling in the positive z direction, so all thicknesses are positive. Normally, a mirror reverses the direction of travel of the light, so usually thicknesses are negative for surfaces following an odd number of reflections. The value of surface thickness can be set by selecting the thickness cell and typing the desired value.

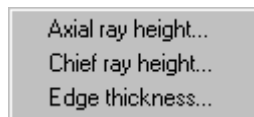
In addition to directly specifying the thickness, there are several other ways to set its value. The thickness options button, which lies directly to the right of the thickness value cell, serves two purposes. First, clicking the button presents you with an options menu of possible ways to specify the thickness. Second, if the thickness has been set by some means other than a direct user specification, the options button will be labeled with a letter indicating how the thickness was set. (Also, when the thickness options button is selected, a brief form of the thickness setting command will be displayed in the command line.)

The options presented in the thickness options menu for setting thickness are:

Direct specification (th)

This means that you have entered a prescribed value for the thickness. Selecting this item will remove any existing solve or pickup data for the thickness.

Solves



Axial ray height solve (py)

The thickness of the surface will be set such that the paraxial axial ray intersects the next surface at a prescribed height. This height corresponds to the value of PY in the paraxial ray trace. This option might be used, for example, to locate a surface at a paraxial image.

Chief ray height solve (pyc)

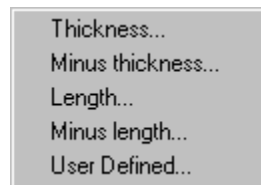
The thickness of the surface will be set such that the paraxial chief ray intersects the next surface at a prescribed height. This height corresponds to the value of PYC in the paraxial ray trace. This option might be used, for example, to locate a surface at the paraxial exit pupil.

Edge contact solve (ec)

The thickness of the surface will be set such that the current surface and the subsequent surface are in contact at a prescribed distance from the axis. The default value of the height used for the solve is the aperture radius of the current surface.

The presence of a solve for a thickness is indicated by an S printed on the thickness options button. Height and edge contact solves can not be placed on the object or image surfaces.

NOTE: Solves should be removed from air spaces when changing the environmental temperature of the system (using the TEM command).

Pickups**Thickness pickup (pk th)**

The thickness of the surface is set to be that of a preceding surface (with the optional addition of a constant).

Minus thickness pickup (pk thm)

The thickness of the surface is set to the negative of that of a preceding surface (with the optional addition of a constant).

Length pickup (pk ln)

The thickness of the surface is set to the sum of the thicknesses between any two preceding surfaces, plus a constant.

Minus length pickup (pk lnm)

The thickness of the surface is set to the negative of the sum of the thicknesses between any two preceding surfaces, plus a constant. This option, might be used, for example, to maintain the overall length of a lens (or a subsection of a lens) at a prescribed value, as is often required in a zoom lens.

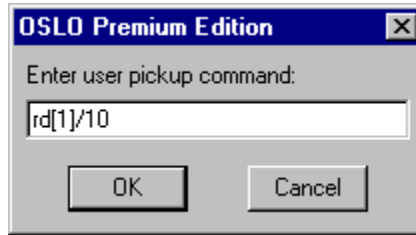
The presence of a pickup for a thickness is indicated by a P on the thickness options button.

Prm User Defined pickup (utp)

This pickup allows arithmetic operations with global data to be used to calculate the thickness and length. The command text may be any valid operation that returns a numeric value. This includes CCL functions which return a real value.

The pickup value is not checked for valid data so care should be exercised when using global surface data values especially when the value is for a surface defined after the surface with the pickup. The pickups are evaluated in surface order.

An example of a pickup entry follows:



Variable

The thickness will be used as a variable during optimization. Thicknesses specified by solves or pickups can not be variable, since their values are constrained by the solve or pickup specification.

Special variable

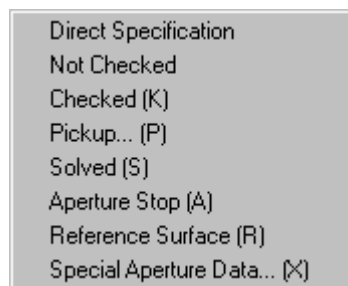
The thickness will be used as a variable during optimization. Also, the variables spreadsheet editor will be opened so that you can edit the controls (minimum value, maximum value, etc.) for the variable.

A variable thickness is denoted by a V on the thickness options button.

If the letter F appears next to the thickness value for surfaces in a lens module, this means that the thickness is fixed and can not be directly changed by the user.

Image surface thickness - Since there is no “next” surface for the image surface, the thickness value has a different meaning. If the thickness of the image surface is non-zero, it is taken as a focus shift of the image surface. Thus, the distance from the next-to-last to image surfaces is the sum of the thickness values of the next-to-last and image surfaces. The Autofocus option sets the thickness of the image surface to a value that minimizes the on-axis spot size.

Aperture radius



The aperture radius is the radius of the (circular) optically effective area of the surface. An aperture radius can be specified by selecting the aperture radius cell and typing the desired value. Normally, the only use of aperture radii by OSLO is to draw the edges on lens elements. So long as rays can be reflected or refracted by the surfaces, they will be transmitted through the system. However, this mode of operation can be changed by setting special options.

It is important to set the entrance beam radius or object numerical aperture to a value sufficiently large to fill the actual aperture stop under all conditions, since the size of the beam is determined by these operating conditions. Individual lens

apertures can then be set to be *checked* so that rays will be blocked if they pass outside the given aperture.

Whether rays are blocked by apertures depends on the particular routine, the type of aperture specification, and the setting of the **aperture_check (apck)** general operating condition (see the General Conditions dialog on page 117). The following shows whether rays will be blocked by apertures under various conditions. For a description of the type specifications, see below.

(apck off/on)	Not checked	Checked	Special
Lens drawing	no/no	no/yes	no/yes
Ray tracing	no/no	no/yes	no/yes
Spot diagram	no/no	yes/yes	yes/yes
Optimization	no/no	no/no	no/no

In addition to directly specifying the aperture radius, there are several other ways of setting its value along with other properties that may be assigned to the surface. The aperture radius options button, which lies directly to the right of the aperture radius value cell, serves two purposes. First, clicking the button presents you with a pop-up menu of possible options for specifying the aperture radius and associated properties. Second, if the aperture radius has been set by some means other than a direct user specification, the options button will be labeled with a letter indicating how the aperture radius was set. Multiple letters can appear on a button to indicate simultaneous conditions. An aperture can be made fixed (F) if it cannot be changed, as for example, with a Catalog Lens. Also, when the aperture radius options button is selected, a brief form of the aperture radius setting command will be displayed in the command line.

The options presented in the aperture radius options menu for setting aperture radius are:

Direct specification

This means that you have entered a prescribed value for the aperture radius. Selecting this item will remove any existing solve or pickup data for the aperture radius.

Not checked (ap)

This is the default condition for an aperture radius. As described above, the value is only used for drawing purposes.

Checked (ap chk)

If aperture checking is used in the ray trace (see general operating conditions), rays are checked and blocked if they fall outside the specified aperture radius. A checked aperture is indicated by a K on the aperture radius options button.

Pickup (pk ap)

The aperture radius is set to the same value as the aperture radius for a preceding surface. An aperture pickup is indicated by a P on the aperture radius options button.

Solved

If no aperture radius value is specified, OSLO will assign an aperture radius value equal to the sum of the magnitudes of the paraxial axial ray height and the paraxial chief ray height on the surface. (This is the paraxial requirement for no vignetting.) Thus, changing other lens data will, in general, result in a change of the aperture radius value. A solved aperture is indicated by an S on the aperture radius button.

Aperture stop (ast)

Selecting this option designates the current surface as the aperture stop for the system. The aperture stop surface is indicated by an A on the aperture radius button.

Reference surface (rfs)

Selecting this option designates the current surface as the reference surface, i.e., the surface at which exact reference rays are aimed. Usually the reference surface is the same as the aperture stop surface. If the reference surface differs from the aperture stop surface, the reference surface is indicated by an R on the aperture radius button.

If the letter F appears on the aperture radius options button, this means that the aperture radius is “fixed” and can not be directly changed by the user.

Prm

Special aperture (apn)

Lt**Std**

In OSLO Light and Standard, a surface may have up to 2 special apertures. In OSLO Premium, any number of special apertures may be placed on a surface.

Aperture shapes other than circular may be defined through the use of a special aperture. Special aperture data is indicated by an X in the aperture radius options button. To define special aperture data for a surface, select Special aperture data

from the pop-up menu and enter the desired number of special apertures. A spreadsheet for editing the special aperture data will be opened.

Surface 3					
Number of special apertures: 2					
Ap Id: A	Pickup Srf: None				
Type: Ellipse	Action: Obstruct	Group: 0			
X min	X max	Y min	Y max	Angle	
-0.500000	0.500000	-0.500000	0.500000	0.000000	
Ap Id: B	Pickup Srf: None				
Type: Quadrangle	Action: Transmit	Group: 0			
AVX1	AVX2	AVX3	AVX4		
-1.000000	-1.000000	1.000000	1.000000		
AVY1	AVY2	AVY3	AVY4		
-1.000000	1.000000	1.000000	-1.000000		
Delete Special Aperture Data					

The special apertures are distinguished by a letter identifier - e.g., aperture A, aperture B. The shape of the aperture may be elliptical, triangular, rectangular, or quadrangular. To select the aperture shape, activate the Type (**atp**) cell by clicking on it, and choose the desired shape from the pop-up menu.

For elliptical or rectangular apertures, four numbers are necessary to define the boundary of the aperture: Y min (**ay1**) and Y max (**ay2**) are the minimum and maximum y values of the aperture and X min (**ax1**) and X max (**ax2**) are the minimum and maximum x values of the aperture. The orientation of the aperture may be specified by an angle **aan** (in degrees) relative to the y-axis. These five quantities may be changed by selecting the appropriate cell and entering the desired value.

Three and four sided special apertures (triangles and quadrangles) are specified by the coordinates of their vertices. For a triangle, the vertices are (**avx1, avy1**), (**avx2, avy2**), and (**avx3, avy3**), and their order is unimportant. For a quadrangle, four vertices, (**avx1, avy1**),..., (**avx4, avy4**), must be specified. Any vertex may be chosen as (**avx1, avy1**). The remaining vertices must then progress either in a clockwise or in a counterclockwise direction:

Correct:

(**avx1, avy1**) = (-1.0, -1.0)

(**avx2, avy2**) = (-1.0, +1.0)

(**avx3, avy3**) = (+1.0, +1.0)

(**avx4, avy4**) = (+1.0, -1.0)

Incorrect:

(**avx1, avy1**) = (-1.0, -1.0)

(**avx2, avy2**) = (+1.0, +1.0)

(**avx3, avy3**) = (-1.0, +1.0)

(**avx4, avy4**) = (+1.0, -1.0)

The shape of a quadrangle must be “convex” - i.e., the interior angle formed by any two adjacent sides must be less than 180 degrees. More complex shapes may be created by assigning triangles and quadrangles to special aperture groups.

The aperture may have one of three properties:

1. it may be a transmitting aperture,
2. it may be an obstruction, or
3. it may allow rays to pass through it undeviated (i.e., a hole in the surface).

To select the action for the aperture, activate the action button cell by clicking on it, and choose the desired action from the pop-up menu. As an alternative to specifying the aperture data, special aperture data may be picked-up from a preceding surface in the lens. To pick-up a special aperture, activate the Pickup Srf button (**pk ap**) for the desired target aperture. The program will prompt you to enter the source surface and aperture identifier. If you want to delete the special aperture data for a surface, click the Delete Special Aperture Data button at the bottom of the spreadsheet.

Multiple special apertures (transmitting apertures, obstructions and holes) may be assigned to a surface. The special apertures assigned to a surface may be subdivided into groups, identified by an integer group number. By default, all special apertures belong to group 0. The group field may be used to assign selected special apertures to groups other than group 0 - i.e., groups 1, 2, etc. Users are free to choose any set of group numbers, but sequential group numbers (1, 2, 3,...) are recommended.

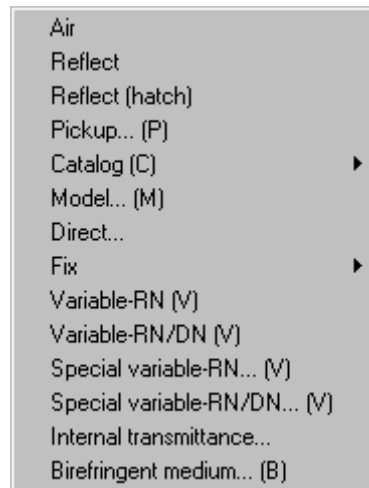
When aperture checking takes place for a ray at a surface having more than one special aperture, built-in “**ANDing**” and “**ORing**” rules are applied in order to determine whether or not the ray passes through the collection of special apertures. It is in this context that group assignments are important: a ray passes through a collection of special apertures if it passes through any single group. That is, the ray is allowed to proceed if it passes through group 0 or group 1 or group 2, etc. (an “**ORing**” rule).

Within any single group, a ray passes through the group if:

- (a) it passes inside every transmitting aperture in the group
(an “**ANDing**” rule),
- (b) it passes outside every obstruction (another “**ANDing**” rule).

A composite aperture which is the “union” of the members of a set of transmitting apertures is formed by assigning each member to a different group. A composite aperture which is the “intersection” of the members of a set of transmitting apertures is formed by assigning each member to the same group.

Glass



The glass associated with a surface is the name of the material between that surface and the next surface. In this context, “glass” can mean any optically transmitting material, not just true glasses. The glass for a surface can be entered by selecting the glass cell and typing the name of the glass. In addition to the glasses in the supplied catalogs, the words **air**, **rfl**, and **rfh** can be used to specify a surface as followed by air, as a reflector, or as a reflector drawn with hatch markings, respectively.

In addition to directly specifying the glass, there are several other ways of setting its value. The glass options button, which lies directly to the right of the glass name cell, serves two purposes. First, clicking the button presents you with a pop-up menu of possible ways to specify the glass. Second, if the glass has been set by some means other than a direct user specification, the options button will be labeled with a letter indicating how the glass was set. (Also, when the glass options button is selected, a brief form of the glass setting command will be displayed in the command line.)

The options presented in the glass options menu for setting glass are:

Air (air)

Specifies that the current surface is followed by air, which has a refractive index of 1.0 for all wavelengths. Selecting this option is equivalent to typing **air** in the glass SmartCell.

Reflect (rfl)

Specifies that the current surface is a reflecting surface (i.e., a mirror). Selecting this option is equivalent to typing **rfl** in the glass SmartCell.

Reflect (hatch) (rfh)

Specifies that the current surface is a reflecting surface (i.e., a mirror) and is to be drawn with “hatch” markings in plan view lens drawings. Selecting this option is equivalent to typing **rfh** in the glass SmartCell.

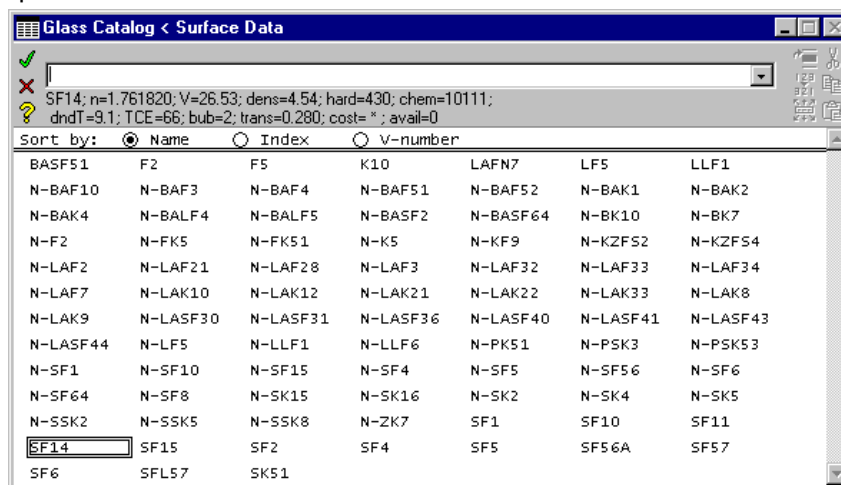
Pickup (pk gla)

The glass is set to be the same as that of a preceding surface. A glass pickup is indicated by a P on the glass options button. Note that only refractive materials are valid for a glass pickup.

Catalog (gla)



Selecting this option allows access to a pull-right menu of the available glass catalogs. After you select the desired catalog, OSLO will open a new spreadsheet, which lists the names of all of the available glasses in that catalog. Selecting a glass (using the arrow keys or by clicking on it with the mouse) will result in the display, in the message area, of the available information in the catalog about that glass. To designate that the selected glass is to be used at the current surface, click OK (). Click CANCEL () if you want to leave the glass catalog spreadsheet and return to the surface data spreadsheet without changing the glass for the current surface. A catalog glass is indicated by a C on the glass options button.



In a catalog glass spreadsheet, glasses can be sorted by name, refractive index (at wavelength 1), or Abbe (V , v) number (at wavelengths 1, 2, and 3). By default, the refractive indices are computed at the d, F, and C wavelengths; if you change the wavelengths or open a lens that uses other wavelengths, a button labeled "Recalculate n and V" will be displayed in the top row of the catalog glass

spreadsheet. Select this button to recalculate n and V for all glasses at the new wavelengths.

Highlighting a glass entry in the catalog glass spreadsheet displays the name of the glass as well as several items of information in the message area next to the command line.

Item	Description
n	Refractive index (at wavelength 1)
V	Abbe number (at wavelengths 1, 2, and 3)
dens	Density (g/cm^3)
hard	Knoop hardness, HK
chem	Chemical properties (first digits of climatic resistance CR, stain resistance FR, acid resistance SR, alkali resistance AR, and phosphate resistance PR)
$dndT$	Derivative of refractive index with respect to temperature ($1 \times 10^{-6} / \text{K}$)
TCE	Thermal coefficient of expansion ($1 \times 10^{-7} / \text{K}$)
bub	Bubble group
trans	Internal transmittance (25 mm thickness, 400 nm wavelength)
cost	Cost relative to BK7 (for Schott catalog) or BSL7 (for Ohara catalog)
avail	Availability code



Note that many of the non-optical properties are available for Schott glasses only; the exception is the availability of cost data for Ohara glasses.

Model (gla mod glassname index1 V_number)

Selecting this option will designate the glass as a Model glass, i.e., a material having a specified index and Abbe ν -number, which do not necessarily correspond to those of a real glass. After selecting Model from the menu, you will be asked to enter a name for the glass, and its refractive index and ν -number. OSLO will use this information to compute refractive indices, as needed, for other wavelengths. A model glass is indicated by an M on the glass options button. If you want to vary the refractive index of a surface in the interactive design window or in optimization, a model glass must be used.

Direct (gla glassname index1 ...)

A direct glass is one for which you have explicitly entered the refractive index for each wavelength. Selecting this option will open a spreadsheet that allows you to enter a glass name and a refractive index value for each wavelength. After you

have entered all the data, click OK () to return to the surface data spreadsheet, or CANCEL () to leave the glass type unchanged.

Prm Internal Transmittance

There is internal transmittance data for the Schott, Ohara and Corning glass catalogs. The data is stored as wavelength-internal transmittance data pairs. The transmittance is for a 5 mm thickness of the material. For other thicknesses, a Beer's law, $\exp(-az)$, dependence is assumed. Local quadratic interpolation is used for wavelengths between the data points.

You can enter transmittance values for non-catalog glasses and override catalog values. Enter values for a 5 mm thick sample for each wavelength.

The internal transmittance data may be displayed and edited using the Glass options button on the Update surface data spreadsheet or the **itn** command. The **internal_trans** CCL command plots the internal transmittance as a function of wavelength for a specified surface.

By default, the internal transmittance is not included during the intensity calculation of the polarization ray trace. However, you can turn on the polarization operating condition to include internal transmittance. (The intensity values will be reduced by the $\exp(-\alpha D)$ factor.) Change this condition using the Update polarization operating conditions spreadsheet (Update >> Operating conditions >> Polarization and select yes for Include Internal Transmittance) or the **pzit** command.

Prm Birefringent Medium

OSLO Premium has the capability of tracing rays through uniaxial birefringent materials such as calcite. Two types of waves (and rays) can propagate in uniaxial media; these waves are called the ordinary wave (or o-ray) and the extraordinary wave (or e-ray). Ordinary waves and rays can be traced using the same techniques as are used for ray tracing in isotropic media. Tracing extraordinary rays, however, is more complicated. The refractive index for extraordinary rays is a function of the angle of incidence. Also, for the extraordinary ray, the wave vector (the normal vector to the wavefront) is generally not in the same direction as the ray vector (the vector in the direction of energy flow, i.e., the Poynting vector). We provide here only a very brief description of the optics of crystals; for a more in-depth treatment, please refer to any optics textbook, such as E. Hecht and A. Zajac, *Optics* (Addison-Wesley, 1974), Chapter 8 or M. Born and E. Wolf, *Principles of Optics*, Sixth Edition (Pergamon Press, 1980), Chapter 14.

The interaction of an electric field with a material is characterized by the permittivity (dielectric constant) ϵ of the material. The permittivity relates the electric field E to the electric displacement D . (See the Optics Reference, Chapter 2). For the nonmagnetic materials used in optical systems, Maxwell's relation states that the refractive index n is equal to the square root of the permittivity, i.e., $n^2 = \epsilon$. For isotropic materials, the permittivity is a scalar quantity (although a function of wavelength). By contrast, the permittivity of an anisotropic medium such as a crystal must be described by a tensor. In other words, ϵ is a 3×3 matrix that relates the components of E to the components of D . (The difference between the wave vector and the ray vector for the extraordinary ray is a consequence of D no longer being collinear with E .)

Since the refractive index is no longer a constant at a particular point in the material, the medium is termed birefringent, since the index of refraction depends on the propagation direction. For the crystal materials under consideration here, a coordinate system can be found in which only the diagonal elements of the dielectric tensor are non-zero. The coordinate axes of this system are called the principal axes and the diagonal elements of the tensor are called the principal values of the permittivity. For uniaxial media, two of the principal values are equal. For biaxial media, all three of the principal values are different. (Note that OSLO does not treat biaxial materials.) For a uniaxial material, the axis along which the permittivity differs is the crystal axis, i.e., the axis of symmetry of the crystal. The principal values and principal axes define the index ellipsoid. In order to trace rays through this uniaxial birefringent medium, then, we must specify the ordinary refractive index, the extraordinary refractive index, and the orientation of the crystal axis.

In OSLO, the data for the ordinary indices is taken from the normal glass specification for the surface. To specify that a medium is birefringent, click the glass options button for the desired surface, and select Birefringent medium from the pop-up menu. In this spreadsheet, you can specify the material that defines the extraordinary refractive indices and the orientation of the crystal axis.

The extraordinary indices may either be calculated from a catalog material, or the indices may be specified explicitly. Click the appropriate radio button to specify your choice. The orientation of the crystal axis is determined by the specification of direction numbers for the axis. The direction numbers (denoted by cak , cal , and cam , which are the direction numbers in x , y , and z , respectively) are the Cartesian components of a vector in the direction of the crystal axis. If the direction numbers are normalized (i.e., the magnitude of the vector is unity), the direction numbers are the direction cosines of the crystal axis. It is not necessary, however, to enter the direction numbers in normalized form. For example, if the crystal axis is parallel to the y -axis of the surface, the direction numbers would be $CAK = 0$, $CAL = 1$, $CAM = 0$. (For this case, where $CAK = CAM = 0$, CAL could be any non-zero value.) For birefringent materials, the crystal axis direction numbers may be made optimization variables and operand components.

In general, a ray incident upon a birefringent material will be split into two rays (o and e), with orthogonal linear polarizations. Since OSLO does not split rays, you must specify which ray is to be traced through the material. This designation is also made in the birefringent medium spreadsheet. By default, the ordinary ray is traced. The easiest way to see the results of tracing the other ray is to make the system a multiconfiguration system, where the configuration item is the ray that is traced through the medium (the name of the configuration item is BRY).

As mentioned above, for the extraordinary ray the wave vector and the ray vector are generally not in the same direction. Thus, we need two sets of direction cosines to characterize the propagation of the ray through the medium. In OSLO, the direction cosines (K , L , and M) reported in the `trace_ray` command (`Evaluate>>Single Ray Trace`) correspond to the ray vector. Similarly, the `rvk`, `rvl`, and `rvm` operands refer to the data for the ray vector. If an extraordinary ray is being traced, the output of the `trace_ray` command will contain three more columns of numbers (columns 7, 8, and 9 of the spreadsheet buffer). These columns contain the values of the direction cosines for the wave vector. These columns are labeled LWV , KWV , and MWV , keeping with the OSLO convention of outputting the y value before the x value. If it is desired to use the wave vector direction cosines in ray operands, the components KWV , LWV , and MWV are available. For ordinary rays in birefringent media and for rays in isotropic media,

the KVV, LWV, and MWV components have the same value as the RVK, RVL, and RVM components.

Five common uniaxial materials have been added to the miscellaneous glass catalog: calcite (CaCO_3), ADP, KDP, MgF_2 , and sapphire (Al_2O_3). With the exception of the o-indices for calcite, the dispersion equations for these materials were taken from the Handbook of Optics, Volume II, Chapter 33, "Properties of Crystals and Glasses", by W. J. Tropf, M. E. Thomas, and T. J. Harris, Table 22 (McGraw-Hill, New York, 1995). The calcite o-index dispersion equation data was calculated by performing a least-squares Sellmeier fit to the data in Table 24 of the reference. The RMS error of this fit is 0.000232, as compared to the tabulated values, over the wavelength range from 0.200 μm to 2.172 μm represented by the values in Table 24. (Also, several minor typographical errors in Table 22 have been corrected. The equations for calcite should be in terms of n_2 not n . The absorption wavelength in the third term for the e-index of MgF_2 should be 23.771995, not 12.771995.) Note that each material corresponds to two entries in the glass catalog: one for the o-indices (CALCITE_O, ADP_O, KDP_O, MGF_2 _O, SAPPHIRE_O) and one for the e-indices (CALCITE_E, ADP_E, KDP_E, MGF_2 _E, SAPPHIRE_E).

As an example of the use of birefringent materials, consider the following system, which is a Wollaston prism. This model can be found in the "...\\public\\len\\demo\\premium" sub-folder of your OSLO installed directory. This prism consists of two wedges of a birefringent material, in this case calcite. In the first wedge, the crystal axis is in the y-direction, while in the second wedge, the crystal axis is in the x-direction. With this orientation of crystal axes, an o-ray in the first wedge becomes an e-ray in the second wedge, and vice-versa. If a beam of circularly polarized light is normally incident on the prism, two beams exit the prism: one deflected upwards, with horizontal linear polarization, and the other downwards, with vertical linear polarization.

*LENS DATA

Wollaston Prism

SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPE	NOTE
OBJ	--	1.0000e+20	1.0000e+14	AIR		
AST	--	--	10.000000 A	AIR		
2	--	5.773503	10.000000	CALCITE_O	CB	
3	--	5.773503	11.547005	CALCITE_O	CB*	
4	--	3.000000	10.000000	AIR		
IMS	--	--	8.000010 S			

*TILT/DECENTER DATA

3	RCO	3					
	DT	1	DCX	--	DCY	--	DCZ
			TLA	-30.000000	TLB	--	TLC

*SURFACE TAG DATA

3	DRW	ON
---	-----	----

*WAVELENGTHS

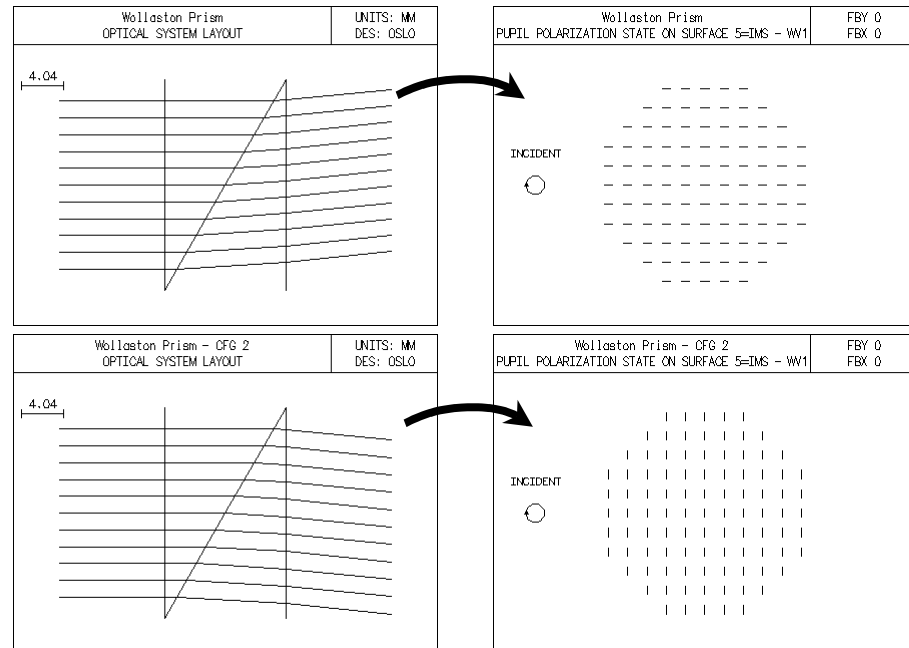
CURRENT	WV1/WW1
1	0.589290
	1.000000

*REFRACTIVE INDICES

SRF	GLASS	RN1	TCE			
0	AIR	1.000000	--			
1	AIR	1.000000	236.000000			
2	CALCITE_O	1.658342	--			
EXTRA	CALCITE_E	1.486440				
BRY	Ordinary	Crystal Axis (K,L,M)	--	1.000000	--	
3	CALCITE_O	1.658342	--			
EXTRA	CALCITE_E	1.486440				
BRY	Extraordinary	Crystal Axis (K,L,M)	1.000000	--	--	
4	AIR	1.000000	236.000000			
5	IMAGE SURFACE					

*CONFIGURATION DATA

TYPE	SN	CFG	QUALF	VALUE
BRY	2	2	0	EXT
BRY	3	2	0	ORD



Fix (gla fix)

If the current glass type is either Model or Direct, there is another option on the menu - Fix. If you select this option, OSLO will look in the glass catalogs for the closest match to the optical properties of your model or direct index glass. You can select the catalog(s) you wish to search from the pull-right menu. Click OK () if you want to change the glass of the current surface to the suggested catalog glass or CANCEL () to leave the glass unchanged.

Note that the Fix command has no connection to the Fixed designation that indicates that a glass cannot be changed, as with a catalog lens.

Variable

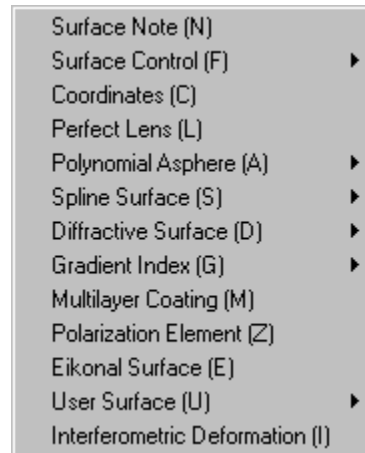
If the current glass type is Model, choose this option to indicate that the refractive index or chromatic dispersion is to be used as a variable during optimization.

Special variable

If the current glass type is Model, choose this option to indicate that the refractive index or chromatic dispersion is to be used as a variable during optimization. Also, the variables spreadsheet will be opened so that you can edit the controls (minimum value, maximum value, etc.) for the variable.

A variable refractive index is indicated by a V on the glass options button. If the letter F appears next to the glass name for surfaces in a lens module, this means that the glass is “fixed” and can not be directly changed by the user.

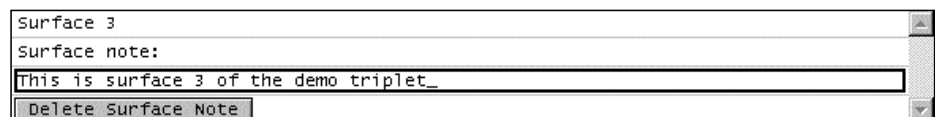
Special surface data



In addition to the normal surface description data of radius, thickness, and glass, there are several types of special data that can be defined for a surface. Clicking on the special data options button will pop-up a menu of the available special data types. To update any of this data, choose the appropriate item from the menu. If any of this data is already defined for a surface, the special options button will be marked with a letter code for that data type, as shown in the menu to the left. Also, the elements in the menu will have a check mark, if that type of special data is present. If a special data type is not allowed on a certain surface, that type will not be shown on the menu.

Surface note (N)

If a surface note is present, the special options button will be marked with an N. The surface note is simply an optional line of text information that can be attached to each surface. It is available so that you can provide any annotation or explanation that is relevant to that surface. If you wish to enter a note for a surface, click the special options button in the surface data spreadsheet and select Surface Note to open the surface note spreadsheet. To enter the note, type the desired text into the line beneath the label Surface note in the spreadsheet. The note can be up to 80 characters in length. The current note can be deleted by clicking the Delete Surface Note button at the bottom of the spreadsheet.



Surface Control (F)

General
Regular Lens Array
Tabular Lens Array
Element Drawing Data

General

The general surface control type is used to indicate general items that do not fall in a specific category. This spreadsheet allows you to designate the surface has having one or more of these special characteristics.

Surface 1	
Fresnel surface: <input checked="" type="checkbox"/> Yes	
Base curvature: <input type="text" value="0.000000"/>	Base conic constant: <input type="text" value="0.000000"/>
Skip to surface: <input type="text" value="0"/>	
Alternate surface intersection: <input type="checkbox"/> No	
Total internal reflection only: <input type="checkbox"/> No	
Extrude aperture to form rod: <input type="checkbox"/> No	
Surface appearance in lens drawings: <input type="text" value="Automatic"/>	
Pen number for surface in lens drawings: <input type="text" value="0"/>	
Thermal coefficient of expansion: <input type="text" value="63.000000"/>	$\times 10^{** -7}$

Fresnel surface (frn) - A Fresnel surface is a hypothetical surface that has the power of a curved surface, but which is actually the shape of a different (base) curve. In OSLO, a Fresnel surface is implemented by treating the surface as defined by the base curvature (**fcv**) and base conic constant (**fcc**) for the purposes of ray transfer, but defined by the “normal” definition of the surface (**cv**, **cc**, aspheric coefficients, etc.) for the purpose of refraction or reflection. If the base curvature is zero (the default), the Fresnel surface substrate is a plane. To designate a surface as a Fresnel surface or to clear the Fresnel designation, activate the Fresnel surface (**frn**) cell by clicking on it; clicking the button toggles the Fresnel surface designation on and off. The substrate shape of the Fresnel surface is

$$z = \frac{fcvr^2}{(1 + \sqrt{1 - fcv^2(fcc + 1)r^2})} \quad (3.1)$$

where $r^2 = x^2 + y^2$.

Std

Prm

Skip to surface (skp) - If a skip surface has been entered, the ray trace will stop at the surface with the skip data and then skip to the designated surface, thereby omitting the intervening surfaces. To clear a skip surface designation, enter 0 as the skip surface.

A typical use of skip surfaces is in multiconfiguration systems where the different configurations represent different beam paths (possibly through different optical elements) through the system. In this case, it is likely that each configuration will skip different parts of the overall system and that the complete system (i.e., surfaces 0 to image surface) is never traversed in any of the configurations.

Alternate surface intersection (asi) - Ray transfer in a homogeneous medium between spherical surfaces can be thought of as finding the intersection point between a line and a sphere. Since a sphere is a closed surface, a line will, in general, intersect the sphere in two places. The OSLO ray trace is set up so that

the transfer of a ray from one surface to the next is carried out with the user's having to specify which intersection point represents the one that corresponds to the actual physical surface. However, there are a few optical systems for which the selection algorithm breaks down. (Note that this does not mean there is something wrong with the algorithm. The alternate surface intersection may lead to a perfectly valid system, just not the one that is wanted.) For example, after an odd number of reflections, a ray is usually traveling in the minus z-direction. If a ray is propagating in the positive z-direction (this is often termed a *strange* ray), it may be transferred to the wrong branch of a conic section unless the ray trace equations are set up to find the alternate intersection between the ray and the surface. To designate the use of the alternate surface intersection or to clear its use, activate the alternate surface intersection (**asi**) cell by clicking on it to toggle between the "normal" and alternate intersection points.

Total internal reflection only (tir) - A total internal reflection (TIR) only surface behaves like a reflecting surface if the condition for total internal reflection is satisfied, i.e., the ray is incident at an angle greater than the critical angle. If the TIR condition is not satisfied, it is treated as a ray failure, and the tracing of the ray is terminated. Since the TIR condition is a function of the refractive indices on both sides of the interface, the glass of the TIR surface should be designated as a refractive material, *not* as a reflector. In other words, make the glass of the TIR surface the material into which the ray would refract if it were incident at less than the critical angle. The refractive index of the glass for the TIR surface will be used to determine if rays satisfy the TIR condition.

Extrude aperture to form rod (rod) - This is the simplest way to specify a straight "light pipe." An extruded surface consists of two parts: (a) the surface defined by the usual specifications (limited by the defined aperture boundary), and (b) the surface generated by "pulling" the aperture boundary in the z-direction by the distance specified by the thickness variable (**th**). The effect is to generate a rod shaped object whose cross section conforms to the aperture. If an elliptical special aperture is specified, this is used to define the cross-sectional shape of the rod. Otherwise, the circular aperture specified by the aperture radius (**ap**) is used.

The glass specification for the surface defines the medium of the rod. Normally, rays are confined to the interior of the rod by total internal reflection (which may take place several times before the ray encounters the next surface). If a ray exits the rod by refraction, it refracts into the original incident medium. The rod is terminated by the next surface in sequence.

Surface appearance in lens drawings (drw) - OSLO has display options for the image surface when the ray trajectories are drawn to the image surface. This option controls the appearance of the current surface in OSLO lens drawings. There are four possible settings: automatic (aut), drawn (on), not drawn (off), or aperture only drawn (ap). The default appearance is "automatic"; the surface is drawn unless it is a dummy surface, i.e., the same refractive index on both sides of the surface. If the "drawn" option is selected, the surface is always drawn, even if it is a dummy surface. If the "not drawn" option is selected, the surface will not be drawn when the lens drawing is created. This does not affect the drawing of faces specified by boundary drawing information, however. Thus, if it is desired to represent surfaces in a lens drawing by one or more boundary drawing faces, the "not drawn" option may be used to suppress the normal drawing of such surfaces (i.e., the drawing derived from lens data). Finally, if the "aperture only drawn" option is selected, the surface drawing from lens data is only suppressed in solid-model drawings (as with the not drawn option). However, the aperture outline of

the surface is drawn as a non-opaque object. This is useful when it is desired to see the boundary of the actual optical aperture imposed on a boundary drawing face.

Pen number for surface in lens drawings (ldp) - If the pen number is defined as "0" (default), the shaded solid color defined in the Lens Drawing Conditions spreadsheet (see Chapter 7, page 238) is used for Shaded Solid drawings. Any other color (1-209) will be taken from the colors map index (**colors** command). If two different colors are applied on two surfaces of the same element, a color gradient is applied in Shaded Solid drawings. You might have to insert (a) dummy surface(s) to display cemented elements with split colors.

Thermal coefficient of expansion (tce) - This property is used to change the dimensional properties of the surface if the temperature is changed. The **tce** parameter defined here overrides any definition of Thermal Coefficient of Expansion that was defined in the glass catalogs.

Note that for an air surface, the Thermal Coefficient of Expansion is defined to be that of aluminum (236×10^{-7} mm/mm/ $^{\circ}$ C) in order to mimic an aluminum spacer. Note also that the spacer on an air surface is defined to be a ring spacer. A ring spacer means that it is supported by the aperture edges of the current surface and the next surface. An expansion of a ring spacer means that the expansion of the air space between the current surface and the next surface takes place relative to the aperture edges of these surfaces, not relative to the vertices of these surfaces. OSLO users can easily change the material of the spacer by changing the value of the **tce** parameter. However, to change the type of spacer from a ring spacer to an "optical axis" spacer, users will have to create a new material in the private glass catalog that has the index of refraction of "air", but the thermal coefficient of the desired material.

Prm Lens Arrays

Lens arrays are aggregations of identical optical assemblies that differ only in their position. The assemblies may be regarded as occupying *channels* defined by array parameters that specify the location and orientation of each channel. In OSLO, an array is defined with respect to a specified surface within the lens system that is called the *channel surface*. This surface encompasses all of the channels in the array. Channels are centered at specified (x, y) positions on this surface and may be shifted by varying amounts in the z direction.

The optical assembly to be placed at each channel center is defined by the sequence of surfaces following the channel surface up to and including the "end-of-array" surface. When you create an array (by selecting Regular Lens Array or Tabular Lens Array from the Surface Control pull-right menu) you will be prompted for the end-of-array surface number. The thickness entered at the end-of-array surface specifies the distance from the array's channel surface to the surface following the end of the array, i.e., an automatic coordinate return to the channel surface is created at the end-of-array surface. This avoids the need for a special end-of-array surface encompassing all channels. If the surface following the end-of-array surface has a global coordinate specification, the thickness at the end-of-array surface is, of course, unnecessary.

There are no special restrictions on the surfaces that comprise the optical assembly replicated at each channel location. The assembly may contain non-sequential groups, tilted and decentered surfaces, etc.

Regular Lens Array - A regular array has its channel centers distributed in a uniform grid over the channel surface. For a regular array, the channel centers are located at the (x, y) coordinates

$$x = i(\mathbf{x_spacing}), \quad \text{where } i = 0, \pm 1, \pm 2, \dots$$

$$y = j(\mathbf{y_spacing}) + \mathbf{offset}, \quad \text{where } j = 0, \pm 1, \pm 2, \dots$$

and $\mathbf{offset} = \mathbf{y_offset}$ if i is odd, or 0 if i is even

The z coordinate of a channel center is the sag of the channel surface at the (x, y) coordinates of the channel center.

If a zero value is specified for the $\mathbf{x_spacing}$, all of the channel centers will lie along the y -axis and the $\mathbf{y_offset}$ value will be ignored. Similarly, if the $\mathbf{y_spacing}$ value is zero, all of the channel centers will lie along the x -axis and $\mathbf{y_offset}$ will again be ignored.

The channels are oriented so that the z -axis of the channel coincides with the normal to the channel surface at the location of the channel center. The y -axis of the channel is oriented so that it is parallel to the yz plane of the local coordinate system of the channel surface.

Surface 1		Delete Lens Array
Array type:	Regular	Draw all channels: <input type="radio"/> Yes <input checked="" type="radio"/> No
End surface:	2	
X spacing:	0.000000	
Y spacing:	0.500000	
Y offset:	0.000000	

Tabular Lens Array - For a tabular lens array, the location and orientation of the channel centers are individually specified. X CTR, Y CTR, and Z CTR specify the coordinates of the channel center. The Z CTR value is added to the sag of the channel surface at the point (X CTR, Y CTR). Thus, if Z CTR = 0, the channel center will lie on the channel surface. **tlA**, **tlB**, and **tlC** specify the tilt angles that will be used to orient the channel relative to the local coordinate system of the channel surface. Note that for tabular lens arrays, the normal to the channel surface plays no part in orienting channels. The channel number (CH NBR) field may be activated to toggle channels on and off. Channel center coordinates and channel tilt angles are preserved when this is done so that this data need not be reentered when an "off" channel is turned back "on".

Surface 1		Delete Lens Array				
Array type:	Tabular	Number of channels: 3 Draw all channels: <input type="radio"/> Yes <input checked="" type="radio"/> No				
End surface:	2					
CH NBR	X CTR	Y CTR	Z CTR	TLA	TLB	TLC
1	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
2	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Array Channel Selection - When a ray has been traced to the channel surface of an array (and refracted or otherwise bent as determined by the properties of the channel surface), a channel is selected prior to continuing the ray trace. Channel selection is based solely on the (x, y) coordinates. The channel whose center is closest to the point of intersection of the ray with the channel surface is the one selected. The z coordinate of the selected channel is obtained by finding the sag of the channel surface at the channel center and, if the array is a tabular array, adding the specified Z CTR value for the selected channel.

Once the coordinates of the channel center have been determined, a new coordinate system is established. This new coordinate system has its origin at the channel center and has its axes oriented as described above, depending upon the type of the array. The ray intersection point is then transformed into this new coordinate system and the ray tracing continues. When that ray has been traced to the end-of-array surface and refracted (or otherwise bent), a return is made to the original coordinates of the channel surface before the thickness on the end-of-array surface is applied. Ray tracing then continues normally.

Array Channel Clipping - The aperture (or any special apertures) at the channel surface is used to clip the array channels. If the selected channel center is not obstructed and falls within the aperture (or any special apertures), ray tracing continues through the array. Otherwise, a ray failure occurs. Thus, the extent of the array is determined by the channel surface aperture specification. This channel clipping occurs whether or not aperture checking has been requested at the channel surface. Note that it is possible for a ray to fall outside the channel surface aperture but still continue through the array if the selected channel center falls within the aperture. If aperture checking has been requested at the channel surface, normal aperture checking is performed prior to channel selection and rays may be rejected prior to channel selection.

Element Drawing Data

For those surfaces that are the first surface of an element, there is also an entry for Element Drawing data in the pop-up menu (if the surface media is "air" then the Element Drawing Data option will not be shown on the pop-up menu). See the discussion of element drawings in the section "Element" on page 230 for a discussion of the ISO element drawing features of OSLO.

Coordinates (C)

The default condition for surface data is that all of the surfaces share a common mechanical axis, which coincides with the optical axis of the system. It is possible to position elements arbitrarily, if desired. A tilted and/or decentered surface is described by tilting and/or decentering the coordinate system in which the surface is described. Since each surface has a local (x, y, z) coordinate system associated with it, you need to specify the position and orientation of this coordinate system. The surface itself is then defined in the normal way, just with respect to this new coordinate system. All tilt and decenter data is entered from the Coordinates spreadsheet, which is accessed by clicking the special options

button in the surface data spreadsheet, and choosing Coordinates from the pop-up menu.

Surface 1					
Decenter - Tilt Order: Decenter, then Tilt					
Decentrations					
DCX	DCY	DCZ			
0.000000	0.000000	0.000000			
Rotation Angles (degrees)					
TLA	TLB	TLC	Offset of Tilt Vertex		
0.000000	0.000000	0.000000	TOX	TOY	TOZ
			0.000000	0.000000	0.000000
Pickup type: None					
Coordinates: Local		Tilt and bend: No			
Coordinate return: No					
Use base coordinate system for coordinate returns to this surface: No					
Delete Coordinates Data					

Since decentration and tilting are non-commutative operations (i.e., tilting, then decentering results in a different coordinate system from decentering, then tilting), you must specify the order in which the tilt and decentration are performed on the coordinate system. The default condition is to decenter the coordinate system, then tilt it. To change this order, click on the item labeled **Decenter - Tilt Order (dt)** in the spreadsheet. Clicking this item just toggles back and forth between the two options.

There are six data items that specify the tilt and decenter of a surface. They can be changed by moving to the appropriate cell and typing in the desired value. The items labeled **dcx**, **dcy**, and **dcz** are the x, y, and z decentrations, respectively. The items labeled **tla**, **tlb**, and **tlc** are the tilts, in degrees, around the -x, -y and +z axes, respectively¹. If tilting is done after decentering, the tilting proceeds in the order **tla**, **tlb**, **tlc**. If tilting is done first, the tilting order is **tlc**, **tlb**, **tla**. For a surface described in local coordinates, the z-decentration (**dcz**) is normally not used.

It is important to note that the tilt and decenter data describe the orientation of the coordinate system in which the surface is described, not the angles of rays that are reflected or refracted at the surface. In general, reflected or refracted rays will “walk away” from the coordinate axes in the new system, making it difficult to set up the elements that follow a tilted and/or decentered surface. (As a simple example, recall that tilting a mirror by an angle θ will deviate the incident beam by an angle 2θ .) To make analysis simpler, it is generally advisable to add an additional dummy surface in contact with the tilted surface, to restore the orientation of the z-axis so that it coincides with the direction of the reflected or refracted reference ray.

Coordinates Pickup - The coordinate data from a surface can be set by a pickup from a preceding surface. To select a coordinate data pickup, click on the item following the Pickup type label. From the pop-up menu, choose whether you want no pickup, a tilt/decenter pickup, or a minus tilt/decenter pickup. If you choose either pickup type, you will be prompted for the surface number from which the data is to be picked up. A minus tilt/decenter pickup copies the data from the source surface, but with a change in sign. This is useful for restoring the coordinate system of a system that has tilted and/or decentered elements.

1. Note that “-x” and “-y” notation signifies that the rotation about the x and y axes are opposite of the right hand rule.

Tilt and Bend - Tilted mirrors are often used to redirect (or fold) the optical axis of a system. A ray along the z-axis in the space preceding a mirror defines, after reflection, a new z-axis for the following surface(s). In order to specify this situation, it is common to introduce an extra tilted “dummy” surface following a tilted mirror. The tilt-and-bend specification (**ben**) eliminates the need for such an artifice.

When the tilt-and-bend specification is entered for a locally tilted mirror, the tilt angles, **tla** and/or **tlb**, which were applied prior to reflection are applied again after reflection before proceeding to the next surface. If surface decentering coordinates are specified, they are applied before the surface is tilted and are not reapplied after reflection.

The tilt angle **tlc** at a tilt-and-bend surface is calculated by OSLO and is chosen in such a way as to cause a meridional ray to remain meridional after reflection. The **tlc** tilt angle will be non-zero only if both **tla** and **tlb** are non-zero.

The tilt-and-bend specification is incompatible with the global coordinates and coordinate return specifications - i.e., the tilt angles are local.

When a tilt-and-bend designation is deleted, the surface tilt specification remains in effect, however, so that the surface is tilted prior to (but not after) reflection.

Coordinate Return - Ordinarily, the base coordinate system of the surface following a given surface, *S*, is obtained by translating the local coordinates of surface *S* along its z-axis by a distance equal to the thickness assigned to surface *S*. Tilts and decenters may then be applied to further position the following surface. Coordinate returns are often used to return to the “main axis” of a system after traversing several locally tilted and/or decentered surfaces.

Sometimes, however, it is convenient to position the following surface with respect to a coordinate system other than the local system of the current surface. Suppose, for example, that it is desired to move a group of surfaces freely in the space separating a surface *A* (preceding the group) and a surface *B* (following the group) without disturbing the position of surface *B* with respect to surface *A*. This could be accomplished by returning to the local coordinate system of surface *A* before using the thickness at the last surface of the group to move to surface *B*. The coordinate return (**rco**) specification may be used for this purpose.

A coordinate return for a given surface, *S*, specifies the surface number of a prior surface, *S'*, which will be returned to before using the thickness at surface *S* to establish the base coordinate system of the next surface (following *S*).

The ray tracing sequence at a surface for which a coordinate return has been specified is as follows: The ray is traced to the surface and refracted (or reflected, diffracted, etc.). Then the base coordinate system of the next surface is established by translating the local coordinate system of surface *S'* along its z-axis by a distance equal to the thickness assigned to surface *S*.

A special case of the coordinate return occurs when the current surface is specified as the surface to return to. (This is, in fact, the default when no surface number is specified with the coordinate return specification.) In this case the base coordinate system of the current surface is “returned to” before applying the thickness to the next surface. The effect is to reverse any surface tilts and/or decenters prior to moving to the next surface. (If the current surface has a global tilt/decenter specification, the base coordinate system is taken to be the system with respect to which the global tilts/decenters were specified). This option is

useful if it is desired to tilt and/or decenter a surface without disturbing the spatial arrangement of any other surfaces in a system.

Prm

In a non-sequential surface group, a coordinate return may only be used on the exit surface of the group.

Global Coordinates - Ordinarily, the local coordinate system of a given surface, *S*, is obtained by translating the local coordinate system of the previous surface along its *z*-axis by a distance equal to the thickness assigned to the previous surface. We may call this translated system the base coordinate system of surface *S*. If tilt angles and/or decentering coordinates are specified at surface *S*, they are applied to reorient and/or reposition the local coordinate system of surface *S* with respect to this base coordinate system. Tilt angles and decentering coordinates applied in this way are called "local."

In situations involving a number of surfaces that are tilted and decentered with respect to one another, it is sometimes more convenient to specify the position and orientation of each surface with respect to some fixed coordinate system. The global coordinates (**gc**) specification may be used for this purpose.

The global coordinates specification designates the surface number of a surface within the system whose local coordinate system will be used in place of the base system of a given surface, *S*, as the reference coordinate system when the surface is tilted and/or decentered. Tilt angles and decentering coordinates applied in this way are called "global."

Note that the global specification in and of itself specifies no tilt angles or decentering coordinates. It simply defines the coordinate system with respect to which tilt angles and decentering coordinates (specified by **tla**, **dcx**, etc.) are applied.

When tilt angles and decentering coordinates at a surface, *S*, are interpreted as "global" with respect to the coordinate system of a prior surface, the thickness assigned to the surface preceding surface *S* is not used. The *z*-position of the surface is controlled solely by the *z* decentration specified by the **dcz** command on surface *S*. Because of this, the thickness preceding a globally tilted or decentered surface is displayed as a blank field in the lens data spreadsheet and ceases to be editable.

Prm

For non-sequential surface groups, a global coordinate specification may only be used on the entry surface, since the remaining surfaces in the group are always positioned and oriented with respect to the entry surface.

Automatic Conversion of Tilts and Decentrations - Whenever the Coordinates item is changed in the Local/Global Coordinates spreadsheet, a new set of tilt angles and decentrations is calculated automatically so that the position and orientation of the current surface remains unaltered. Thus, for example, tilt angles and decentrations for one or more surfaces may be set up initially in global coordinates and then changed to a local coordinate representation simply by activating the Coordinates item. The automatic conversion of tilt angles and decentrations also takes place when a **gc** or **gcd** command is executed in global editing mode.

When switching to local coordinates, the *z*-decentration of the current surface is set to zero and a thickness is calculated and assigned to the preceding surface. When switching from local to global coordinates, the thickness assigned to the preceding surface becomes undefined and an appropriate *z*-decentration is assigned to the current surface.

Prm

The position and orientation of each surface following the entry surface in a non-sequential group is required to be specified globally with respect to the entry surface. When a non-sequential group is created, either by executing the **group** command or by selecting the Non-sequential Group item from the Edit menu in the Update Surface Data spreadsheet, tilt angles and decentrations with respect to the entry surface are also calculated automatically for each surface following the entry surface. Similarly, surface positions and orientations are converted back to a local representation when a non-sequential group is ungrouped. Note, however, that when a non-sequential group is ungrouped, the resulting lens will not necessarily make optical sense.

The set of tilt angles defining the orientation of a surface is not unique. For example, a **tla** of 180 degrees followed by a **tlc** of 180 degrees is equivalent to a **tlb** of 180 degrees. When tilt angles are calculated, the following limits are imposed in order to eliminate ambiguity:

(3.2)

$$-180^{\circ} \leq \mathbf{tla} \leq +180^{\circ}$$

$$-90^{\circ} \leq \mathbf{tlb} \leq +90^{\circ}$$

$$-180^{\circ} \leq \mathbf{tlc} \leq +180^{\circ}$$

No automatic calculation of tilt angles and decentrations takes place while the surface data spreadsheet is open since the final lens set up is not performed until all lens editing is completed and the spreadsheet is closed.

Perfect Lens (L)

The perfect lens surface simulates a rotationally-symmetric lens that is perfectly corrected for a specified magnification. In addition, it is assumed that the principal points are imaged without spherical aberration. The perfect lens surface transforms the ray data (ray intersection, direction, and path length) in the front principal plane into the ray data in the back principal plane. The results are exact, i.e., not based on a series development.

The surface on which the perfect lens is defined corresponds to the location of both principal planes of the lens; hence the perfect lens surface should be planar. The thickness for the surface preceding the perfect lens surface is the distance to the front principal plane of the perfect lens. The thickness for the perfect lens surface is the distance from the real principal plane to the next surface. To define a perfect lens, select Perfect Lens from the special data popup menu. Two items of data are needed to define the perfect lens: the focal length (**pfl**) of the lens (entered in the current lens units) and the magnification (**pfm**) at which the lens is perfectly corrected. The magnification can have any value except values in a small interval surrounding +1. OSLO will set any magnification larger than 1.0×10^7 equal to 1.0×10^7 . A common use of the perfect lens surface is to convert an afocal system to a focal system. In this case, the lens should be perfect for an

infinite object side conjugate so the magnification for which the lens is corrected should be zero.

Surface 1	
Focal length:	100.0
Magnification:	0.000000
Delete Perfect Lens	

Polynomial Asphere (A)

Conic/Toric
Standard Asphere
Symmetric General Asphere
Asymmetric General Asphere
Symmetric Asphere (all orders)
Asymmetric Asphere (abs. value)
ISO Symmetric Asphere
ISO X-toric
ISO Y-toric
Biconic
Symmetric Cone
Asymmetric Cone
Symmetric Zernike Sag
Asymmetric Zernike Sag

Of the various types of aspheric surfaces, those having a surface sag given by a polynomial expression are the most common. Within this subclass, there are several different polynomials that are commonly used to characterize surfaces in optical design.

OSLO uses the terms *symmetric* and *asymmetric* to differentiate aspheric surfaces. A symmetric asphere has rotational symmetry about the z-axis, while an asymmetric asphere does not. Of course, symmetric surfaces are easier to fabricate than asymmetric surfaces, but the design of rotationally symmetric systems is also much easier. Among the asymmetric aspheres, most have symmetry about the yz plane, but the Asymmetric General Asphere surface can be used to represent a surface with no symmetry whatever.

The so-called ISO surfaces are ones proposed in the ISO 10110 drawing standard. Most of these are similar to other surfaces used in optical design software in the past, and are included to provide compatibility with the new specification.

Many of the aspheric surface types allow you to choose the maximum order of the expansion polynomial that you want to use. A new spreadsheet will be opened, in which you can edit the values of the various aspheric coefficients. In each spreadsheet there is a button labeled Delete Aspheric Data that can be used to remove all the aspheric specifications from the surface. The types of surfaces and the descriptions are:

Conic/Toric

A surface described only by a radius of curvature or curvature value has a spherical shape. The items in the conic and toric spreadsheet can change the shape of the surface.

A conic surface is defined by the equation

(3.3)

$$z = \frac{\mathbf{cv}r^2}{1 + \sqrt{1 - \mathbf{cv}^2(\mathbf{cc} + 1)r^2}}$$

where $r^2 = x^2 + y^2$, \mathbf{cv} is the curvature of the surface ($\mathbf{cv} = 1/\mathbf{rd}$, where \mathbf{rd} is the radius of curvature) and \mathbf{cc} is the conic constant. (The conic constant is equal to minus the square of the eccentricity.)

The type of conic depends on the value of \mathbf{cc} as follows:

$\mathbf{cc} = 0$	sphere
$\mathbf{cc} = -1$	paraboloid
$\mathbf{cc} < -1$	hyperboloid
$-1 < \mathbf{cc} < 0$	ellipsoid

If \mathbf{cc} is greater than 0, the surface is no longer a conic of revolution, but rather a type of surface known as an *oblate spheroid*.

To enter a value for the conic constant for a surface, click the special options button in the surface data spreadsheet and select Conic/Toric from the Polynomial Asphere pull-right menu to open the conic and toric data spreadsheet. Select the conic constant cell, and type in the desired value. On the line below the conic constant value, the type of conic label will change to reflect the type of surface you have defined.

Surface 1	
Conic constant:	1.100000
Conic type:	Oblate Spheroid
Surface type:	<input checked="" type="radio"/> Rotationally symmetric <input type="radio"/> Cylinder <input type="radio"/> Toroid
Toric curvature:	0.000000

A toroidal surface is formed by rotating a spherical, conic, or aspheric surface (defined in the yz plane) about an axis that is parallel to the y-axis, but displaced from it by a distance

(3.4)

$$\mathbf{rdx} = \frac{1}{\mathbf{cvx}}$$

where \mathbf{cvx} is called the *toric curvature*. The surface has a circular profile (of radius of curvature \mathbf{rdx}) in the xz plane, and a profile in the yz plane given by

(3.5)

$$z = \frac{\mathbf{cv}y^2}{1 + \sqrt{1 - \mathbf{cv}^2(\mathbf{cc} + 1)y^2}}$$

If $\mathbf{cvx} = 0$, the surface is a cylinder, with its axis parallel to the x-axis. If \mathbf{cvx} is not zero, the surface is a true toric. Torics are designated by the direction of their rotation axis; the surface described above is a “y-toric.” For most applications in practical optics, this surface must be rotated 90 degrees about the z-axis, making an “x-toric”. It is important to distinguish the two types of torics, since one cannot be converted into the other by interchanging \mathbf{cv} and \mathbf{cvx} .

The type of surface is selected by clicking the appropriate radio button. For a rotationally symmetric surface, click the Rotationally symmetric button. Click the Cylinder button if you want a cylindrical surface. Recall, as described above, that the cylinder axis is parallel to the x-axis. Cylindrical surfaces at other azimuth angles are defined by rotating about the z-axis. See the discussion of local/global coordinates earlier in this chapter for a discussion of how to rotate the coordinate system for a surface. If you want to define a toroidal surface, click the Toroid radio button. To enter a value for the toric curvature of a surface, select the toric curvature cell in the conic/toric spreadsheet, and enter the desired value.

Standard Asphere (ADO)

The standard type of aspheric surface that is most commonly used in optical design is a rotationally symmetric surface described by power series terms (in r) of up to 10th order. This surface has the general form

(3.6)

$$z = \frac{\mathbf{cv}r^2}{1 + \sqrt{1 - \mathbf{cv}^2(\mathbf{cc} + 1)r^2}} + \mathbf{ad}r^4 + \mathbf{ae}r^6 + \mathbf{af}r^8 + \mathbf{ag}r^{10}$$

where

(3.7)

$$r = \sqrt{x^2 + y^2}$$

\mathbf{cv} and \mathbf{cc} are the curvature and conic constant as described in the preceding section. \mathbf{ad} , \mathbf{ae} , \mathbf{af} , and \mathbf{ag} are aspheric constants. The sign convention for the aspheric coefficients is the same as for curvatures, i.e., the coefficient is positive if it leads to a surface deformation in the positive z-direction. These constants are entered in the Standard Asphere spreadsheet, which is accessed by clicking the

special options button in the surface data spreadsheet, and choosing Standard Asphere from the Polynomial Asphere pull-right menu.

Surface 1	
4th order deformation coefficient:	0.000000
6th order deformation coefficient:	0.000000
8th order deformation coefficient:	0.000000
10th order deformation coefficient:	0.000000
Delete Standard Asphere	

To change the value of an aspheric coefficient, select the cell corresponding to the desired order coefficient, and enter the new value. All of the standard aspheric data for a surface may be deleted by clicking the Delete Standard Asphere button at the bottom of the spreadsheet.

Note that if you have defined the surface to be a toroid (see the preceding section), the surface profile in the yz plane is given by

(3.8)

$$z = \frac{c y^2}{1 + \sqrt{1 - c v^2 (c c + 1) y^2}} + a d y^4 + a e y^6 + a f y^8 + a g y^{10}$$

Lt Std Prm Symmetric General Asphere (ASR)

This surface is an extension of the rotationally symmetric aspheric surface described in the preceding section, with the exception that this surface allows the use of polynomials of any order. When you define the surface, you also define the maximum order that you want to use in the polynomial. The coefficients in the polynomial are denoted by **asi**, where **asi** is the coefficient of the term of order $2i$. The profile of the **ASR** surface is given by

(3.9)

$$z = \frac{c v r^2}{1 + \sqrt{1 - c v^2 (c c + 1) r^2}} + a s 1 r^2 + a s 2 r^4 + a s 3 r^6 + a s 4 r^8 + a s 5 r^{10} + a s 6 r^{12} + \dots$$

In the above equation, **cv** and **cc** are the curvature and conic constant, as described in the discussion of conic surfaces earlier in this chapter.

Surface 1				
ASP ASR 10				
AS0: r^0	AS1: r^2	AS2: r^4	AS3: r^6	AS4: r^8
0.000000	0.000000	0.000000	0.000000	0.000000
AS5: r^10				
0.000000				
Delete General Asphere Data				

Pm Asymmetric General Asphere (ASX)

This surface is a general “potato-chip” surface that is specified by a polynomial in x and y . The coefficients in the polynomial are denoted by **asi**. Any order polynomial may be used. The profile of the **ASX** surface is given by

(3.10)

$$z = \sum_{i=1}^n z_i$$

where

(3.11)

$$z_i = (\mathbf{asi})x^j y^k$$

(3.12)

$$i = \frac{1}{2}[(j+k)^2 + j + 3k]$$

Given the coefficient number i , it is possible to calculate the order o of the term in the series, as well as the exponents j and k , according to

(3.13)

$$o = \text{floor}\left(\frac{\sqrt{(1+8i)}-1}{2}\right)$$

(3.14)

$$k = i - o(o+1)/2$$

(3.15)

$$j = o - k$$

Note that the curvature and conic constant are not used for **ASX** surfaces.

Surface 1				
ASP ARA 10 H = fabs(sqrt(x^2 + y^2))				
AS0: H^0	AS1: H^1	AS2: H^2	AS3: H^3	AS4: H^4
0.000000	0.000000	0.000000	0.000000	0.000000
AS5: H^5	AS6: H^6	AS7: H^7	AS8: H^8	AS9: H^9
0.000000	0.000000	0.000000	0.000000	0.000000
AS10: H^10				
0.000000				
Delete General Asphere Data				

Lt **Std** **Prm** **Symmetric Asphere (all orders)(ARA)****Prm** **Asymmetric Asphere (abs. value)(AXA)**

Both of these aspheric types are expansions in the absolute value of the surface coordinates and may be useful in the design of axicon-like surfaces. The rotationally symmetric version is denoted as the **ARA** type. This surface is a base conic, plus a series in all orders of H , where

$$H = \left| \sqrt{x^2 + y^2} \right| \quad (3.16)$$

The surface profile is defined by

$$z = \frac{cvr^2}{1 + \sqrt{1 - cv^2(cc+1)r^2}} + as0 + as1H + as2H^2 + as3H^3 + as4H^4 + \dots \quad (3.17)$$

The asymmetric asphere – absolute value surface (type **AXA**) is similar to the asymmetric general asphere (type **ASX**), except that the absolute values $|x|$ and $|y|$ are used in the expansion. Thus, the profile of the **AXA** surface is given by

$$z = \sum_{i=0}^n z_i \quad (3.18)$$

where

$$z_i = (asi)|x|^j|y|^k \quad \text{and} \quad i = \frac{1}{2}[(j+k)^2 + j + 3k] \quad (3.19)$$

Lt **Std** **Prm** **ISO Symmetric Asphere (ISR)**

This is one of the surfaces defined in the ISO standard 10110, Part 12. The difference between this surface and the **ASR** surface defined above is the form of the expansion polynomial. The polynomial coefficients are denoted by **asa i** , where i is the order of the coefficient. The profile of the **ISR** surface is given by

$$z = \frac{cvr^2}{1 + \sqrt{1 - cv^2(cc+1)r^2}} + asa3H^3 + asa4H^4 + asa5H^5 + asa6H^6 + \dots \quad (3.20)$$

where

$$H = \left| \sqrt{x^2 + y^2} \right| \quad (3.21)$$

In the above equation, **cv** and **cc** are the curvature and conic constant, as described in the discussion of conic surfaces earlier in this chapter.

Surface 1				
ASP ISR 10 H = fabs(sqrt(x^2 + y^2))				
ASA3: H^3	ASA4: H^4	ASA5: H^5	ASA6: H^6	ASA7: H^7
0.000000	0.000000	0.000000	0.000000	0.000000
ASA8: H^8	ASA9: H^9	ASA10: H^10		
0.000000	0.000000	0.000000		
Delete General Asphere Data				

Prm ISO X-Toric (TRX)

A toric surface is generated by the rotation of a defining curve contained in a plane about an axis that lies in the same plane. An x-toric has its defining curve [denoted by $z = g(x)$] in the xz plane and its axis of rotation parallel to the x -axis. The z -coordinate at which the axis of rotation intersects the z -axis is given by **rd** ($= 1/\mathbf{cv}$). The equation of an x-toric surface is

(3.22)

$$z = \frac{cv[y^2 - g^2(x)] + 2g(x)}{1 + \sqrt{1 - cv\{cv[y^2 - g^2(x)] + 2g(x)\}}}$$

The defining curve $g(x)$ is given by

(3.23)

$$g(x) = \frac{cvxx^2}{1 + \sqrt{1 - cvx^2(ccx + 1)x^2}} + asc3|x|^3 + asa4x^4 + asc5|x|^5 + asa6x^6 + \dots$$

The defining curve xz plane curvature (**cvx**) and conic constant (**ccx**) may be edited in the Conic/Toric Spreadsheet.

Surface 1				
ASP TRX 10 X = fabs(x) Y = fabs(y)				
ASC3: X^3	ASA4: X^4	ASC5: X^5	ASA6: X^6	ASC7: X^7
0.000000	0.000000	0.000000	0.000000	0.000000
ASA8: X^8	ASC9: X^9	ASA10: X^10		
0.000000	0.000000	0.000000		
Delete General Asphere Data				

Prm ISO Y-Toric (TRY)

A y-toric has its defining curve [denoted by $z = g(y)$] in the yz plane and its axis of rotation parallel to the y-axis. The z-coordinate at which the axis of rotation intersects the z-axis is given by **rdx** ($= 1/\mathbf{cvx}$). The equation of a y-toric surface is

(3.24)

$$z = \frac{\mathbf{cvx}[x^2 - g^2(y)] + 2g(y)}{1 + \sqrt{1 - \mathbf{cvx}\{\mathbf{cvx}[x^2 - g^2(y)] + 2g(y)\}}}$$

The defining curve $g(y)$ is given by

(3.25)

$$g(y) = \frac{\mathbf{cv}y^2}{1 + \sqrt{1 - \mathbf{cv}^2(\mathbf{cc} + 1)y^2}} + \mathbf{asd3}|y|^3 + \mathbf{asb4}y^4 + \mathbf{asd5}|y|^5 + \mathbf{asb6}y^6 + \dots$$

The defining curve conic constant (**cc**) and the toric curvature (**cvx**) may be edited in the Conic/Toric spreadsheet.

Surface 1				
ASP TRY 10 X = fabs(x) Y = fabs(y)				
ASD3: Y^3	ASB4: Y^4	ASD5: Y^5	ASB6: Y^6	ASD7: Y^7
0.000000	0.000000	0.000000	0.000000	0.000000
ASB8: Y^8	ASD9: Y^9	ASB10: Y^10		
0.000000	0.000000	0.000000		
Delete General Asphere Data				

Prm Biconic (BIC)

This surface has the form of a conic section in both x and y meridians, but with (generally) different curvatures and conic constants. In addition, a power series asphericity may be added to the base biconic. The power series terms are denoted by **asai**, **asbi**, **asci**, and **asdi**. The equation of the biconic surface is

(3.26)

$$z = \frac{\mathbf{cvxx}^2 + \mathbf{cv}y^2}{1 + \sqrt{1 - \mathbf{cvx}^2(\mathbf{ccx} + 1)x^2 - \mathbf{cv}^2(\mathbf{cc} + 1)y^2}} + \mathbf{asa4}x^4 + \mathbf{asb4}y^4 + \mathbf{asa6}x^6 + \mathbf{asb6}y^6 + \dots + \mathbf{asc3}|x|^3 + \dots + \mathbf{asd3}|y|^3 + \dots$$

The curvature (**cvx**) and conic constant (**ccx**) in the xz azimuth can be edited in the Conic/Toric data spreadsheet.

Surface 1				
ASP BIC 10 X = fabs(x) Y = fabs(y)				
ASC3: X^3	ASD3: Y^3	ASA4: X^4	ASB4: Y^4	ASC5: X^5
0.000000	0.000000	0.000000	0.000000	0.000000
ASD5: Y^5	ASA6: X^6	ASB6: Y^6	ASC7: X^7	ASD7: Y^7
0.000000	0.000000	0.000000	0.000000	0.000000
ASA8: X^8	ASB8: Y^8	ASC9: X^9	ASD9: Y^9	ASA10: X^10
0.000000	0.000000	0.000000	0.000000	0.000000
ASB10: Y^10				
0.000000				
Delete General Asphere Data				

Lt Std Prm Symmetric Cone (CNR)

The cone surface used in OSLO is also defined in ISO standard 10110. The data item **cns** defines the “cone angle” of the cone. If the cone has a half-angle of θ , **cns** has a magnitude of $1/\tan\theta$. The sign of **cns** determines the orientation of the cone. If any power series terms are used to describe the surface, they are denoted by **asa i** . The profile of the **cnr** surface is given by

(3.27)

$$z = \text{cns}\sqrt{x^2 + y^2} + \text{asa3}H^3 + \text{asa4}H^4 + \text{asa5}H^5 + \dots$$

where

(3.28)

$$H = \left| \sqrt{x^2 + y^2} \right|$$

Surface 1				
ASP CNR 10 H = fabs(sqrt(x^2 + y^2))				
CNS				
0.000000				
ASA3: H^3	ASA4: H^4	ASA5: H^5	ASA6: H^6	ASA7: H^7
0.000000	0.000000	0.000000	0.000000	0.000000
ASA8: H^8	ASA9: H^9	ASA10: H^10		
0.000000	0.000000	0.000000		
Delete General Asphere Data				

Prm Asymmetric Cone (CNX)

An asymmetric cone has different cone angles in the x and y azimuths. The data items **cns**, **cnx**, and **cny** are used to define the base cone surface. In addition, a power series asphericity may be added to the cone. The power series terms are denoted by **asa_i**, **asb_i**, **asc_i**, and **asd_i**. The profile of the **cnx** surface is given by

(3.29)

$$z = \text{cns} \sqrt{\frac{x^2}{\text{cnx}^2} + \frac{y^2}{\text{cny}^2}} + \text{asa}4x^4 + \text{asb}4y^4 + \text{asa}6x^6 + \text{asb}6y^6 + \dots + \text{asc}3|x|^3 + \dots + \text{asd}3|y|^3 + \dots$$

Surface 1				
ASP CNX 10 X = fabs(x) Y = fabs(y)				
CNS	CNY	CNX		
0.000000	0.000000	0.000000		
ASC3: X^3	ASD3: Y^3	ASA4: X^4	ASB4: Y^4	ASC5: X^5
0.000000	0.000000	0.000000	0.000000	0.000000
ASD5: Y^5	ASA6: X^6	ASB6: Y^6	ASC7: X^7	ASD7: Y^7
0.000000	0.000000	0.000000	0.000000	0.000000
ASA8: X^8	ASB8: Y^8	ASC9: X^9	ASD9: Y^9	ASA10: X^10
0.000000	0.000000	0.000000	0.000000	0.000000
ASB10: Y^10				
0.000000				
Delete General Asphere Data				

Lt Std Prm Symmetric Zernike Sag (ZSR)

The symmetric Zernike sag surface is an aspheric surface type for which the sag departure from a base conic is described by a series of rotationally-symmetric Zernike polynomials. The shape of the surface is defined by the equation

(3.30)

$$z = \frac{c r^2}{1 + \sqrt{1 - c v^2 (c c + 1) r^2}} + \sum_{i=0}^n (a s i) Z_i$$

In the above equation **cv** and **cc** are, as usual, the curvature and conic constant, $r^2 = x^2 + y^2$, n is the number of Zernike terms used in the series, **asi** is the coefficient of Zernike term i , and Z_i is the Zernike polynomial i . The numbering of the Zernike polynomials is the same as for the Zernike phase diffractive surfaces. Referring to the table on p. 92, the numbering for the rotationally symmetric Zernike sag surface is the same as for the rotationally symmetric Zernike phase surface (**zrr**). The radial coordinate ρ in the Zernike polynomials is normalized such that $\rho = 1$ at the aperture radius specified in the lens data. Note that this surface type modifies the shape of the surface, in contrast to the Zernike phase surfaces, which do not. Care should be exercised when using Zernike data obtained, for example, from an interferometric test. If the data refers to an actual surface profile, then a Zernike sag surface should probably be used; if the data refers to an overall wavefront, then a Zernike phase surface should probably be used.

Surface 1				
See Help page (click "?" button) for definition of Zernike coefficients.				
ASP	ZSR	16		
AS0	AS1	AS2	AS3	AS4
0.000000	0.000000	0.000000	0.000000	0.000000
AS5	AS6	AS7	AS8	AS9
0.000000	0.000000	0.000000	0.000000	0.000000
AS10	AS11	AS12	AS13	AS14
0.000000	0.000000	0.000000	0.000000	0.000000
AS15	AS16			
0.000000	0.000000			
Delete General Asphere Data				

Prm Asymmetric Zernike Sag (ZSX)

The asymmetric Zernike sag surface is an aspheric surface type for which the sag departure from a base conic is described by a series of Zernike polynomials. The shape of the surface is defined by the equation

(3.31)

$$z = \frac{c v r^2}{1 + \sqrt{1 - c v^2 (c c + 1) r^2}} + \sum_{i=0}^n (a s i) Z_i$$

In the above equation **cv** and **cc** are, as usual, the curvature and conic constant, $r^2 = x^2 + y^2$, n is the number of Zernike terms used in the series, **asi** is the coefficient of Zernike term i , and Z_i is the Zernike polynomial i . The numbering of the Zernike polynomials is the same as for the Zernike phase diffractive surfaces. Referring to the table on p. 92, the numbering for the asymmetric Zernike sag surface is the same as for the asymmetric Zernike phase surface (**zrx**). The radial

coordinate ρ in the Zernike polynomials is normalized such that $\rho = 1$ at the aperture radius specified in the lens data, and the angular coordinate θ is measured from the local y-axis of the surface. Note that this surface type modifies the shape of the surface, in contrast to the Zernike phase surfaces, which do not. Care should be exercised when using Zernike data obtained, for example, from an interferometric test. If the data refers to an actual surface profile, then a Zernike sag surface should probably be used; if the data refers to an overall wavefront, then a Zernike phase surface should probably be used.

Surface 1				
See Help page (click "?" button) for definition of Zernike coefficients.				
ASP ZSX 8				
A50	A51	A52	A53	A54
0.000000	0.000000	0.000000	0.000000	0.000000
A55	A56	A57	A58	
0.000000	0.000000	0.000000	0.000000	
Note: Positive angle theta is a rotation from the +y axis toward the +x axis.				
Delete General Asphere Data				

Spline Surface (S)

Radial Spline Profile
Asymmetric Bi-Cubic Spline

Lt Std Prm Radial Spline Profile

In contrast to the aspheric surfaces described in the previous sections, which are directly defined in terms of surface deformation, the spline aspheric surface is a rotationally symmetric surface that is described by the slope of the surface at prescribed radial distances from the surface vertex. There is no limit on the number of spline surface radial zone heights, but the zone heights must be unique and in ascending order. Since the surface profile can not be computed for radial distances larger than the largest defined spline radial zone height, rays incident on the spline surface at heights larger than the largest zone height will be blocked and the ray trace for that ray will be terminated. Thus, it is advisable to place the largest spline zone height at a value that is larger than the expected incident height of any rays. There is no restriction on minimum or maximum zone widths, as long as the width is non-zero. To define a spline surface, activate the Special options button for the desired surface and choose Spline Profile from the pop-up

menu. You will be prompted for the number of radial spline zones you wish to place on the surface.

Surface 1			
Number of spline surface zones: 4			
Aperture Zone 0		Aperture Zone 1	
Radial Height	Surface Slope	Radial Height	Surface Slope
0.000000	0.000000	0.000000	0.000000
Aperture Zone 2		Aperture Zone 3	
Radial Height	Surface Slope	Radial Height	Surface Slope
0.000000	0.000000	0.000000	0.000000
Aperture Zone 4			
Radial Height	Surface Slope		
0.000000	0.000000		
Delete Spline Surface			

The definition of the spline aspheric surface is

(3.32)

$$z = \frac{1}{6}c_{(i-1)}\frac{(sh_i - r)^3}{sh_i - sh_{(i-1)}} + \frac{1}{6}c_i\frac{(r - sh_{(i-1)})^3}{sh_i - sh_{(i-1)}} + \left[ss_i - \frac{1}{2}c_i(sh_i - sh_{(i-1)})\right](sh_i - r) + Z_{(i-1)} - \frac{1}{6}c_{(i-1)}(sh_i - sh_{(i-1)})^2$$

the above equation, r ($r^2 = x^2 + y^2$) is the radial position on the surface, which lies in radial zone i of the spline surface. Radial zone i lies between the defined zone heights sh_{i-1} and sh_i . ss_i is the slope of the surface (dz/dr) at the zone height sh_i . The curvatures c_i and surface sags Z_i at the spline zones are computed from the initial values

(3.33)

$$c_0 = \mathbf{cv}$$

(3.34)

$$Z_0 = 0$$

and the recursion formulae

(3.35)

$$c_i = \frac{2(ss_i - ss_{(i-1)})}{sh_i - sh_{(i-1)}} - c_{(i-1)}$$

(3.36)

$$Z_i = \left[ss_i - \frac{1}{6}(sh_i - sh_{(i-1)})(c_{(i-1)} + 2c_i) \right] (sh_i - sh_{(i-1)}) + Z_{(i-1)}$$

Prm Asymmetric Bi-Cubic Spline

The Asymmetric spline aspheric surface is a non-rotationally symmetric surface that is described by the sag for a specific X,Y position. The surface is fit to the set of defined points by a Bi-Cubic interpolation. OSLO interprets the data as Y points for a given X value. For each row of X, OSLO computes an interpolating spline curve of the YZ values. For a given x,y point, each a z value is computed from the YZ curves to define data for an XZ curve at the value y. The XZ curve is used to evaluate the surface point at x. for each value of X. An orthogonal spline curve is computed in X at the slope of the surface at prescribed radial distances from the surface vertex. For the OSLO algorithm, at least 3 values of X are required and for each X, three values of Y and Z are required. Therefore, to define an asymmetric Bi-Cubic spline surface, a minimum of 9 X,Y,Z points is required.

Surface 1				
Number of asymmetric spline points: <input type="text" value="9"/>				
Enter increasing values of X and Y. Enter all Y values for each X.				
Spline Point 0	X =	0.000000	Y =	0.000000
Spline Point 1	X =	0.000000	Y =	0.000000
Spline Point 2	X =	0.000000	Y =	0.000000
Spline Point 3	X =	0.000000	Y =	0.000000
Spline Point 4	X =	0.000000	Y =	0.000000
Spline Point 5	X =	0.000000	Y =	0.000000
Spline Point 6	X =	0.000000	Y =	0.000000
Spline Point 7	X =	0.000000	Y =	0.000000
Spline Point 8	X =	0.000000	Y =	0.000000
<input type="button" value="Delete Spline Surface"/>				

Since the surface profile cannot be computed for radial heights greater than the largest defined spline radial zone height, rays incident on the spline surface at heights greater than the largest zone height will be blocked and the ray trace terminated for that ray.

Diffractive Surface (D)

Linear Grating
Optical Hologram
Symmetric CGH (even orders)
Asymmetric CGH (power series)
Symmetric CGH (all orders)
Asymmetric CGH (abs. value)
Symmetric Zernike Phase
Asymmetric Zernike Phase
User-defined Diffractive Surface

Diffractive surfaces have a periodic structure that determines the direction of rays passing through the surface. OSLO has several types of diffractive surfaces, including ordinary gratings such as are used in spectroscopic applications, optical holograms formed by interfering two beams of light, and several types of computer-generated holograms, with a user-specified phase distribution. All diffractive surfaces are represented as phase distributions that are equivalent to grating rulings. For most surfaces, you will be prompted for the maximum order you wish to use in the polynomial expansion of the phase distribution.

Linear Grating

This type of surface represents an ordinary (ruled) diffraction grating. The substrate of the grating is determined by the description of the current surface profile. The grating lines are parallel to the local x-axis. An azimuthal tilt of the surface (**tlc** - see the description of local/global coordinates earlier in this chapter) can be used to provide an arbitrary orientation of the grating lines. Two quantities are used to describe the grating. They are changed by selecting the appropriate cell and entering the desired value. The grating order (**gor**) is the order of diffraction that is to be used for evaluation. The grating spacing (**gsp**) is the separation of the grating lines, given in the current lens units. In the case where the grating is ruled on a non-planar substrate, the grating lines are equally spaced along the chord of the substrate (corresponding to normal ruled grating fabrication technology). To remove a grating from a surface, set the order and grating spacing to zero.

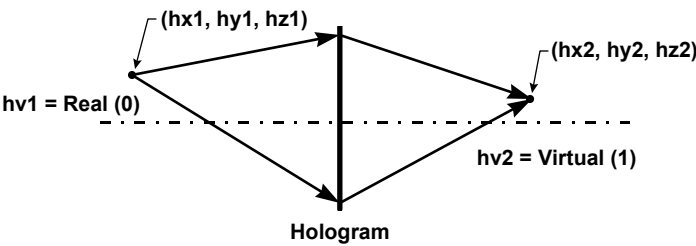
Surface 1	
Grating order:	1
Grating spacing:	0.010000
Blaze order:	0
Groove depth:	0.000000

The blaze order (**gbo**) and groove depth (**gdp**) are used only for the extended scalar theory diffraction efficiency calculations described in Chapter 10 of the Optics Reference manual.

Lt **Std** **Prm** Optical Hologram (HOE)

A holographic optical element is considered, for the purposes of ray tracing, to be the interference pattern between two point sources (the “object” and “reference” beams), referred to as source 1 and source 2. The hologram can be placed on an aspheric surface, but the source beams always remain spherical. (See the discussion of CGH diffractive surfaces later in this chapter for a discussion of how

to use a more general diffractive phase function.) The hologram itself is assumed to infinitely thin.



The location of point source 1, as measured from the vertex of the current surface, is specified by the data items **hx1**, **hy1**, and **hz1**. These are the x, y, and z coordinates of the point source. The location of point source 2, as measured from the vertex of the current surface, is specified by **hx2**, **hy2**, and **hz2**. **hor** is the hologram reconstruction diffraction order. The wavelength (in μm) of the construction beams used to make the hologram is **hwv**. All of these items are changed by selecting the desired item and entering the new value. **hv1** and **hv2** are the *virtual* factors for the point sources. They indicate whether the respective point source is real or virtual. If the construction beam ray vector points from a source toward the hologram, the factor is real. Conversely, if the construction beam ray vector points from the hologram toward the source point, the factor is virtual. Activating the cell for **hv1** or **hv2** by clicking on it toggles the virtual factor between its two options. The HOE may be removed from a surface by clicking the Delete HOE button at the bottom of the spreadsheet.

Surface 1				
HOR	1	HWV	0.587560	
Point Source 1				
HV1	Real			
	HX1	HY1	HZ1	
	0.000000	0.000000	-100.000000	
Point Source 2				
HV2	Real			
	HX2	HY2	HZ2	
	0.000000	0.000000	100.000000	
Delete HOE				

Lt **Std** **Prm** **Symmetric CGH (even orders) - (DFR)**

The phase function of this surface is given by a power series in the radial coordinate r . The coefficient of order $2i$ in the polynomial is denoted by **dfi**. The phase added to a ray at a **DFR** surface is given by

(3.37)

(3.38)

$$r^2 = x^2 + y^2$$

$$\phi(r) = \text{dor} \frac{2\pi}{\lambda_0} (df0 + df1r^2 + df2r^4 + df3r^6 + df4r^8 + \dots)$$

In the above equation, **dor** is the diffraction order to be used in the ray trace, and λ_0 is a constant called the *design wavelength*. The design wavelength is converted into the units of the current lens in order to evaluate the above equation, but like all wavelengths used in OSLO, the wavelength (**dwv**) is entered in μm .

Surface 1				
DOE DFR	10	Diffraction order:	1	Design wavelength:
				0.587560
Kinoform construction order:		1	Kinoform zone depth:	
			0.000000	
DF0: r^0	DF1: r^2	DF2: r^4	DF3: r^6	DF4: r^8
0.000000	0.000000	0.000000	0.000000	0.000000
DF5: r^10				
0.000000				
Delete Diffractive Surface				

The kinoform construction order (**kco**) and kinoform zone depth (**kdp**) are used only for the extended scalar theory diffraction efficiency calculations described in Chapter 10 of the Optics Reference manual.

Prm Asymmetric CGH (power series) - (DFX)

The phase function of this surface is given by a power series in x and y . The coefficients of the polynomial are denoted by **dfi**. The phase added to a ray at a **DFX** surface is given by

(3.39)

$$\phi(x, y) = \sum_{i=1}^n \phi_i$$

where

(3.40)

$$\phi_i = \text{dor} \frac{2\pi}{\lambda_0} (\text{dfi}) x^j y^k$$

(3.41)

$$i = \frac{1}{2}[(j+k)^2 + j + 3k]$$

In the above equation, **dor** is the diffraction order to be used in the ray trace, and λ_0 is a constant called the *design wavelength*. The design wavelength is converted into the units of the current lens in order to evaluate the above equation, but like all wavelengths used in OSLO, the wavelength (**dwv**) is always entered in μm . Given the coefficient number i , it is possible to calculate the order o as well as the exponents j and k , according to

(3.42)

$$o = \text{floor}\left(\frac{\sqrt{(1+8i)}-1}{2}\right)$$

(3.43)

$$k = i - o(o+1)/2$$

(3.44)

$$j = o - k$$

Surface 1				
DOE DFx	5	Diffraction order:	1	Design wavelength:
				0.587560
Kinoform construction order:		1	Kinoform zone depth:	
				0.000000
DF0:x^0 y^0	DF1:x^1 y^0	DF2:x^0 y^1	DF3:x^2 y^0	DF4:x^1 y^1
0.000000	0.000000	0.000000	0.000000	0.000000
DF5:x^0 y^2	DF6:x^3 y^0	DF7:x^2 y^1	DF8:x^1 y^2	DF9:x^0 y^3
0.000000	0.000000	0.000000	0.000000	0.000000
DF10:x^4 y^0	DF11:x^3 y^1	DF12:x^2 y^2	DF13:x^1 y^3	DF14:x^0 y^4
0.000000	0.000000	0.000000	0.000000	0.000000
DF15:x^5 y^0	DF16:x^4 y^1	DF17:x^3 y^2	DF18:x^2 y^3	DF19:x^1 y^4
0.000000	0.000000	0.000000	0.000000	0.000000
DF20:x^0 y^5				
0.000000				
Delete Diffractive Surface				

The kinoform construction order (**kco**) and kinoform zone depth (**kdp**) are used only for the extended scalar theory diffraction efficiency calculations described in Chapter 10 of the Optics Reference manual.

Lt **Std** **Prm** **Symmetric CGH (all orders) - (DRA)**

The phase function for this surface is determined by taking the absolute value of a power series expansion in the radial coordinate r . The coefficient of order i is denoted by **dfi**. The phase function for the **DRA** surface is

(3.45)

$$\phi(H) = d\text{or} \frac{2\pi}{\lambda_o} (df0 + df1H + df2H^2 + df3H^3 + df4H^4 + \dots)$$

where

(3.46)

$$H = \sqrt{x^2 + y^2}$$

In the above equation, **dor** is the diffraction order to be used in the ray trace, and λ_o is a constant called the *design wavelength*. The design wavelength is converted into the units of the current lens in order to evaluate the above equation, but like all wavelengths used in OSLO, the wavelength (**dww**) is always entered in μm .

Surface 1				
DOE DRA 10 Diffraction order: 1 Design wavelength: 0.587560				
H = fabs(sqrt(x^2 + y^2))				
Kinoform construction order: 1 Kinoform zone depth: 0.000000				
DF0: H^0	DF1: H^1	DF2: H^2	DF3: H^3	DF4: H^4
0.000000	0.000000	0.000000	0.000000	0.000000
DF5: H^5	DF6: H^6	DF7: H^7	DF8: H^8	DF9: H^9
0.000000	0.000000	0.000000	0.000000	0.000000
DF10: H^10				
0.000000				
Delete Diffractive Surface				

The kinoform construction order (**kco**) and kinoform zone depth (**kdp**) are used only for the extended scalar theory diffraction efficiency calculations described in Chapter 10 of the Optics Reference manual.

Prm**Asymmetric CGH (absolute value) - (DXA)**

The phase function for this surface is determined by taking the absolute value of a power series expansion in the x and y coordinates. The coefficients of the expansion are denoted **dfi**. The phase added to a ray at a **DXA** surface is given by

(3.47)

$$\phi(x, y) = \sum_{i=0}^n \phi_i$$

where

(3.48)

$$\phi_i = \text{dor} \frac{2\pi}{\lambda_0} (\text{dfi}) |x|^j |y|^k$$

(3.49)

$$i = \frac{1}{2}[(j+k)^2 + j + 3k]$$

In the above equation, **dor** is the diffraction order to be used in the ray trace, and λ_0 is a constant called the *design wavelength*. The design wavelength is converted into the units of the current lens in order to evaluate the above equation, but like all wavelengths used in OSLO, the wavelength (**dwv**) is always entered in μm .

Surface 1				
DOE DXA	5	Diffraction order:	1	Design wavelength:
X = fabs(x) Y = fabs(y)				
Kinoform construction order:		1	Kinoform zone depth:	
DF0:X^0 Y^0	DF1:X^1 Y^0	DF2:X^0 Y^1	DF3:X^2 Y^0	DF4:X^1 Y^1
0.000000	0.000000	0.000000	0.000000	0.000000
DF5:X^0 Y^2	DF6:X^3 Y^0	DF7:X^2 Y^1	DF8:X^1 Y^2	DF9:X^0 Y^3
0.000000	0.000000	0.000000	0.000000	0.000000
DF10:X^4 Y^0	DF11:X^3 Y^1	DF12:X^2 Y^2	DF13:X^1 Y^3	DF14:X^0 Y^4
0.000000	0.000000	0.000000	0.000000	0.000000
DF15:X^5 Y^0	DF16:X^4 Y^1	DF17:X^3 Y^2	DF18:X^2 Y^3	DF19:X^1 Y^4
0.000000	0.000000	0.000000	0.000000	0.000000
DF20:X^0 Y^5				
0.000000				
Delete Diffractive Surface				

The kinoform construction order (**kco**) and kinoform zone depth (**kdp**) are used only for the extended scalar theory diffraction efficiency calculations described in Chapter 10 of the Optics Reference manual.

Lt

Std

Prm

Symmetric Zernike Phase (ZRR)

Prm

Asymmetric Zernike Phase (ZRX)

In OSLO Light and Standard, only the Symmetric Zernike Phase surface is available, while in Premium both Zernike Phase surfaces are available.

A Zernike surface is a diffractive surface in which the phase function of the surface is described as a series of Zernike polynomials. You will be prompted for the number of Zernike coefficients you wish to use in the expansion. There are two types of Zernike surfaces: a symmetric surface (**zrr**) and an asymmetric surface (**zrx**).

Surface 1				
See Help page (click "?" button) for definition of Zernike coefficients.				
ZOE ZRX	15	Diffraction order:	1	Design wavelength:
		Kinoform construction order:	1	Kinoform zone depth:
Trace reference rays through Zernike phase surface: <input type="checkbox"/> Off				
ZR0	ZR1	ZR2	ZR3	ZR4
0.000000	0.000000	0.000000	0.000000	0.000000
ZR5	ZR6	ZR7	ZR8	ZR9
0.000000	0.000000	0.000000	0.000000	0.000000
ZR10	ZR11	ZR12	ZR13	ZR14
0.000000	0.000000	0.000000	0.000000	0.000000
ZR15				
0.000000				
Note: Positive angle theta is a rotation from the +y axis toward the +x axis.				
<input type="button" value="Delete Zernike Phase Surface"/>				

Zernike polynomials are functions of polar coordinates ρ and θ . The radial coordinate ρ is normalized such that $\rho = 1$ at the aperture radius specified in the lens data, and the angular coordinate θ is measured from the local y-axis of the surface. The normalization of the Zernike polynomials themselves is consistent with the description given in the ISO standard 10110 Part 5 and with that used by a number of commercial interferometer analysis programs. However, this is not the only possible normalization, and care must be used to ensure that a consistent normalization is maintained. The coefficients of the various Zernike terms are in units of wavelengths, at a prescribed design wavelength (**dwv**).

For symmetric **zrr** surfaces, the phase introduced by the surface is independent of the azimuth angle θ . The phase is specified in terms of the rotationally symmetric Zernike polynomials. As indicated in the table below, **zri** is the coefficient of the rotationally symmetric Zernike term of leading order $2i$.

For asymmetric **zrx** surfaces, the phase is specified by the coefficients **zri** of the allowed 37 Zernike polynomials used by OSLO. Note that the azimuth angle θ is measured from the y-axis and is positive toward the positive x-axis, in order to be consistent with the conventions of the **dfx** diffractive surfaces and with the convention of taking the y-axis as the meridional axis. Some interferometer analysis programs and optical design programs may reference the azimuthal angle to their x-axis. In such cases, the 90 degree rotation in the reference azimuth must be taken into account in relating the coefficients of the Zernike polynomials.

Term	zrr type	zrx type	Zernike polynomial
0	zr0	zr0	1
1		zr1	$\rho \cos \theta$
2		zr2	$\rho \sin \theta$
3	zr1	zr3	$2\rho^2 - 1$
4		zr4	$\rho^2 \cos 2\theta$
5		zr5	$\rho^2 \sin 2\theta$
6		zr6	$(3\rho^2 - 2)\rho \cos \theta$
7		zr7	$(3\rho^2 - 2)\rho \sin \theta$
8	zr2	zr8	$6\rho^4 - 6\rho^2 + 1$
9		zr9	$\rho^3 \cos 3\theta$
10		zr10	$\rho^3 \sin 3\theta$
11		zr11	$(4\rho^2 - 3)\rho^2 \cos 2\theta$
12		zr12	$(4\rho^2 - 3)\rho^2 \sin 2\theta$
13		zr13	$(10\rho^4 - 12\rho^2 + 3)\rho \cos \theta$
14		zr14	$(10\rho^4 - 12\rho^2 + 3)\rho \sin \theta$

15	zr3	zr15	$20\rho^6 - 30\rho^4 + 12\rho^2 - 1$
16		zr16	$\rho^4 \cos 4\theta$
17		zr17	$\rho^4 \sin 4\theta$
18		zr18	$(5\rho^2 - 4)\rho^3 \cos 3\theta$
19		zr19	$(5\rho^2 - 4)\rho^3 \sin 3\theta$
20		zr20	$(15\rho^4 - 20\rho^2 + 6)\rho^2 \cos 2\theta$
21		zr21	$(15\rho^4 - 20\rho^2 + 6)\rho^2 \sin 2\theta$
22		zr22	$(35\rho^6 - 60\rho^4 + 30\rho^2 - 4)\rho \cos \theta$
23		zr23	$(35\rho^6 - 60\rho^4 + 30\rho^2 - 4)\rho \sin \theta$
24	zr4	zr24	$70\rho^8 - 140\rho^6 + 90\rho^4 - 20\rho^2 + 1$
25		zr25	$\rho^5 \cos 5\theta$
26		zr26	$\rho^5 \sin 5\theta$
27		zr27	$(6\rho^2 - 5)\rho^4 \cos 4\theta$
28		zr28	$(6\rho^2 - 5)\rho^4 \sin 4\theta$
29		zr29	$(21\rho^4 - 30\rho^2 + 10)\rho^3 \cos 3\theta$
30		zr30	$(21\rho^4 - 30\rho^2 + 10)\rho^3 \sin 3\theta$
31		zr31	$(56\rho^6 - 105\rho^4 + 60\rho^2 - 10)\rho^2 \cos 2\theta$

32		zr32	$(56\rho^6 - 105\rho^4 + 60\rho^2 - 10)\rho^2 \sin 2\theta$
33		zr33	$(126\rho^8 - 280\rho^6 + 210\rho^4 - 60\rho^2 + 5)\rho \cos \theta$
34		zr34	$(126\rho^8 - 280\rho^6 + 210\rho^4 - 60\rho^2 + 5)\rho \sin \theta$
35	zr5	zr35	$252\rho^{10} - 630\rho^8 + 560\rho^6 - 210\rho^4 + 30\rho^2 - 1$
36	zr6	zr36	$924\rho^{12} - 2772\rho^{10} + 3150\rho^8 - 1680\rho^6 + 420\rho^4 - 42\rho^2 + 1$

FRINGE was a program for interferometric analysis that used the x-axis as a reference for Zernike polynomials.

Using the numbering of the Zernike polynomials as in the above table, a set of OSLO coefficients using the y-axis as the $\theta = 0$ azimuth (denoted Zi_y) can be related to a set of FRINGE coefficients using the x-axis as the $\theta = 0$ azimuth (denoted Zi_x) via the following table:

OSLO	FRINGE	OSLO	FRINGE	OSLO	FRINGE
Z1 _y	Z2 _x	Z13 _y	Z14 _x	Z25 _y	Z26 _x
Z2 _y	Z1 _x	Z14 _y	Z13 _x	Z26 _y	Z25 _x
Z3 _y	Z3 _x	Z15 _y	Z15 _x	Z27 _y	Z27 _x
Z4 _y	-Z4 _x	Z16 _y	Z16 _x	Z28 _y	-Z28 _x
Z5 _y	Z5 _x	Z17 _y	-Z17 _x	Z29 _y	-Z30 _x
Z6 _y	Z7 _x	Z18 _y	-Z19 _x	Z30 _y	-Z29 _x
Z7 _y	Z6 _x	Z19 _y	-Z18 _x	Z31 _y	-Z31 _x
Z8 _y	Z8 _x	Z20 _y	-Z20 _x	Z32 _y	Z32 _x
Z9 _y	-Z10 _x	Z21 _y	Z21 _x	Z33 _y	Z34 _x
Z10 _y	-Z9 _x	Z22 _y	Z23 _x	Z34 _y	Z33 _x
Z11 _y	-Z11 _x	Z23 _y	Z22 _x	Z35 _y	Z35 _x
Z12 _y	Z12 _x	Z24 _y	Z24 _x	Z36 _y	Z36 _x

The kinoform construction order (**kco**) and kinoform zone depth (**kdp**) are used only for the extended scalar theory diffraction efficiency calculations described in Chapter 10 of the Optics Reference manual.

Trace reference rays through Zernike phase surface (zrt) - With this option set to "off" (the default), reference rays ignored the additional phase introduced by Zernike phase surfaces. Thus, Zernike phase surfaces representing interferometric data would reproduce the input wavefront, including any tilt terms. There may be situations however, when users would like reference rays to be traced through the Zernike phase. For example, the Zernike phase surface may be being used as just an arbitrary diffractive surface, in place of a polynomial expansion of the phase. This option in the Zernike phase surface spreadsheet allows the user to specify whether reference rays are to be traced through the Zernike phase.

Prm User Defined Diffractive Surface (UDF)

The user-defined diffractive surface allows for the implementation of diffractive surfaces with phase functions that can not be described by one of the polynomial-based CGH surfaces. Similarly to the user-defined sag surface, you do *not* have to trace the ray to the surface and refract or reflect it. With the user-defined diffractive surface, the user-supplied routine only needs to compute the value of the phase function that describes the surface and its derivatives with respect to *x* and *y*; OSLO will find the intersection of the ray with this surface and change the ray direction and optical path length in accordance with the grating equation and the phase function that you specify.

To designate a surface as a user-defined diffractive surface, select the Special button for the desired surface in the surface data spreadsheet, and then select

Diffraction Surface>>User-defined Diffraction Surface from the pop-up menu. You may define an arbitrary number of UDF*i* coefficients. These coefficients are used to pass parameter values to your CCL/DLL command (in the Udf_udfcf[] array) and can also be made optimization variables and operand components. In the spreadsheet, enter the name of the command that computes the value of the phase function and its x and y derivatives, and the values of the UDF*i* coefficients.

For the user-defined diffraction surface, the CCL global variables set by OSLO are the following.

Variable	Interpretation
Udf_sfc	surface number (integer)
Udf_x	x coordinate of ray, relative to surface vertex
Udf_y	y coordinate of ray, relative to surface vertex
Udf_z	z coordinate of ray, relative to surface vertex
Udf_dwv	design wavelength (dwv), in μm , of diffraction surface
Udf_nudfcf	number of user-defined diffraction surface coefficients (integer)
Udf_udfcf [Udf_nudfcf]	array [0, ..., Udf_nudfcf – 1] of user-defined diffraction surface coefficients

When the user-defined diffraction surface is encountered in the ray trace, the values of the variables in the above table are set, and control is passed to the specified CCL or DLL command. This command computes the value of the phase function and its x and y derivatives at the supplied point of intersection with the surface. These values are transmitted to OSLO in the CCL global variables listed in the table below.

Variable	Interpretation
Udf_err	error flag (integer); 0 when control passed to CCL
Udf_phfntyp	type of phase function values computed by command (integer; 0, 1, or 2)
Udf_phasefn	phase function value at (x, y, z)
Udf_dphfndx	derivative of the phase function with respect to x
Udf_dphfndy	derivative of the phase function with respect to y

Upon completion of the execution of the CCL/DLL command, OSLO diffracts the ray and continues the ray trace through the system. If the ray cannot be traced through the diffraction surface, the CCL/DLL command should set the value of the variable Udf_err to a non-zero integer value. If Udf_err is non-zero when control of the ray trace is returned to OSLO, the ray trace (for that ray) is terminated.

The integer variable Udf_phfntyp should be set to either 0, 1, or 2 depending upon the units used for the returned value of Udf_phasefn.

Udf_phfntyp	Units of Udf_phasefn
0.	lens units
1.	design wavelengths (dvw)
2.	radians

For example, consider the simple quadratic phase function

$$\phi(x, y) = (2\pi/\lambda_0)a(x^2 + y^2) .$$

If the CCL command computes $a(x^2 + y^2)$, Udf_phfntyp should be set equal to 0.

If $(a/\lambda_0)(x^2 + y^2)$ is computed, Udf_phfntyp should be set equal to 1.

Finally, if $(2\pi/\lambda_0)a(x^2 + y^2)$ is computed, the proper value of Udf_phfntyp is 2.

Gradient Index (G)



A gradient index medium is one in which the refractive index is a function of position, rather than a constant. To designate a surface to be followed by a gradient index medium, activate the special options button in the surface data spreadsheet and select Gradient Index from the pop-up menu. You can select the type of index distribution to use from the pull-right menu.

The paraxial ray trace in OSLO is correct for gradients that are only radial *or* axial. This affects the computation of paraxial constants, the paraxial ray trace, paraxial setup, **pwr** operand components, and the Gaussian beam spreadsheet. Solves in gradient index systems are supported. Note that the computation of aberration coefficients does *not* include the effects of the gradient index.

Gradient step size (dth) - For all of the gradient index types, there is a step size data item. This is the step length (in the current lens units) used in tracing the ray through the medium. A smaller step size will result in a more accurate ray trace, but at the expense of increased computation time.

Gradient index of refraction - In the following equations defining the gradient types, n_0 is the base index of the material, as determined by the glass entered for that surface. In general, n_0 and the **nr*i*** and **nz*i*** coefficients are functions of wavelength. The **nr*i*** and **nz*i*** coefficients can be changed by selecting the appropriate cell and entering the desired value.

Gradient index coordinate system - The coordinate system in which the gradient index profile is defined may be tilted and/or decentered from the local coordinate system of the surface for which the gradient data is defined. The gradient coordinate data only affects the index distribution, it does not affect the position of any surfaces. The gradient coordinate system is defined relative to the local coordinate system of the gradient surface. The functional form of the index distribution is unchanged; the change is in the (x, y, z) coordinate system in which the distribution is defined. The gradient system may be tilted (**gta, gtb, gtc**) and/or decentered (**gdx, gdy, gdz**). Just as when specifying surface coordinates, you need to designate whether the tilt or the decentration is performed first. Separation of the gradient coordinate system from the surface coordinate system is useful both for tolerancing and for the specification of unusual gradient index orientations.

Deleting a gradient index material - A gradient index medium may be deleted by clicking the Delete GRIN Medium button at the bottom of the spreadsheet.

The available gradient types and their definitions are:

Lt **Std** **Prm** **Axial-Radial Gradient (ARG)**

The index distribution is given by

$$n(r, z) = n_0 + \mathbf{nz1}z + \mathbf{nz2}z^2 + \mathbf{nz3}z^3 + \mathbf{nz4}z^4 + \mathbf{nr1}r^2 + \mathbf{nr2}r^4 + \mathbf{nr3}r^6 + \mathbf{nr4}r^8 \quad (3.50)$$

where

$$r^2 = x^2 + y^2 \quad (3.51)$$

Surface 1							
GDT ARG		Step Size:		0.100000			
WVN	Wavelength	NZ1	NZ2	NZ3	NZ4		
1	0.587560	0.000000	0.000000	0.000000	0.000000		
2	0.486130	0.000000	0.000000	0.000000	0.000000		
WVN	Wavelength	NR1	NR2	NR3	NR4		
1	0.587560	0.000000	0.000000	0.000000	0.000000		
2	0.486130	0.000000	0.000000	0.000000	0.000000		
GRIN coordinate system decentration/tilt: Decenter, then Tilt							
	GDX	GDY	GDZ		GTA	GTB	GTC
	0.000000	0.000000	0.000000		0.000000	0.000000	0.000000
Delete GRIN Medium							

Wood Lens (WOD)

The index distribution is given by

(3.52)

$$n(r) = n_0 + \text{nr1}r^2$$

where

(3.53)

$$r^2 = x^2 + y^2$$

Surface 1							
GDT WOD		Step Size:		0.100000			
WVN	Wavelength	NR1					
1	0.587560	0.000000					
2	0.486130	0.000000					
GRIN coordinate system decentration/tilt: Decenter, then Tilt							
	GDX	GDY	GDZ		GTA	GTB	GTC
	0.000000	0.000000	0.000000		0.000000	0.000000	0.000000
Delete GRIN Medium							

Obviously, the Wood lens gradient is just a special case of the axial and radial gradient; it is included only for efficiency.

Lt Std Prm SELFOC™ Gradient (SEL)

This gradient is a radial gradient in a form used by Nippon Sheet Glass. The index distribution is given by

(3.54)

$$n^2(r) = n_0^2[1 - (\text{nr1}r)^2 + \text{nr2}(\text{nr1}r)^4 + \text{nr3}(\text{nr1}r)^6 + \text{nr4}(\text{nr1}r)^8]$$

where

(3.55)

$$r = \sqrt{x^2 + y^2}$$

Note that, unlike the other gradient types, the SELFOC gradient is a power series in n^2 , rather than n .

Surface 1						
GDT SEL		Step Size:		0.100000		
WVN	Wavelength	NR1	NR2	NR3	NR4	
1	0.587560	0.000000	0.000000	0.000000	0.000000	
2	0.486130	0.000000	0.000000	0.000000	0.000000	
GRIN coordinate system decentration/tilt: Decenter, then Tilt						
	GDY	GDZ		GTA	GTB	GTC
	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Delete GRIN Medium						

Lt Std Prm GRADIUM™ Gradient (GRD)

The Gradium gradient index material is an axial gradient, with chromatic dispersion properties that are determined according to a model developed by LightPath Technologies, Inc. (www.lightpath.com). A primary feature of this type of gradient is that, in addition to the refractive index being a function of position in the medium, the dispersion is also a function of position.

At a base wavelength λ_0 , the gradient index distribution n_{λ_0} is given by

(3.56)

$$\begin{aligned}
 n_{\lambda_0} z' = & \text{gnz0} + \text{gnz1} \left(\frac{z'}{\text{gmz}} \right) + \text{gnz2} \left(\frac{z'}{\text{gmz}} \right)^2 + \text{gnz3} \left(\frac{z'}{\text{gmz}} \right)^3 \\
 & + \text{gnz4} \left(\frac{z'}{\text{gmz}} \right)^4 + \text{gnz5} \left(\frac{z'}{\text{gmz}} \right)^5 + \text{gnz6} \left(\frac{z'}{\text{gmz}} \right)^6 \\
 & + \text{gnz7} \left(\frac{z'}{\text{gmz}} \right)^7 + \text{gnz8} \left(\frac{z'}{\text{gmz}} \right)^8 + \text{gnz9} \left(\frac{z'}{\text{gmz}} \right)^9 \\
 & + \text{gnz10} \left(\frac{z'}{\text{gmz}} \right)^{10} + \text{gnz11} \left(\frac{z'}{\text{gmz}} \right)^{11}
 \end{aligned}$$

In Eq. (4.52), **gmz** is the thickness of the Gradium blank and z' is the distance into the blank and is given by

(3.57)

$$z' = \text{goz} + z$$

where z is, as usual, the axial distance from the vertex of the surface and **goz** is the offset into the Gradium blank for the vertex of the surface.

Typically, λ_0 is 0.58756 μm ; in OSLO, λ_0 is denoted by **gwv**. At wavelengths other than λ_0 , the refractive index is computed from a fourth-order expansion in Buchdahl's chromatic coordinate ω . Defining a constant κ by

(3.58)

$$\kappa = \frac{1}{\lambda_0 - 0.187}$$

the chromatic coordinate ω for a wavelength λ is given by

(3.59)

$$\omega = \frac{\delta\lambda}{1 + \kappa (\delta\lambda)}$$

where

(3.60)

$$\delta\lambda = \lambda - \lambda_0$$

The index n_λ at wavelength λ is then given as

(3.61)

$$n_\lambda(z) = n_{\lambda_0} + V_1\omega + V_2\omega^2 + V_3\omega^3 + V_4\omega^4$$

where

(3.62)

$$V_1 = \alpha_1 n_{\lambda_0}^3 + \beta_1 n_{\lambda_0}^2 + \gamma_1 n_{\lambda_0} + \delta_1$$

(3.63)

$$V_2 = \alpha_2 n_{\lambda_0}^3 + \beta_2 n_{\lambda_0}^2 + \gamma_2 n_{\lambda_0} + \delta_2$$

(3.64)

$$V_3 = \alpha_3 n_{\lambda_0}^3 + \beta_3 n_{\lambda_0}^2 + \gamma_3 n_{\lambda_0} + \delta_3$$

(3.65)

$$V_4 = \alpha_4 n_{\lambda_0}^3 + \beta_4 n_{\lambda_0}^2 + \gamma_4 n_{\lambda_0} + \delta_4$$

In OSLO, the α coefficients in the above equations are denoted by **gra1**, **gra2**, **gra3**, and **gra4**; the β coefficients by **grb1**, **grb2**, **grb3**, and **grb4**; the γ coefficients by **grc1**, **grc2**, **grc3**, and **grc4**; and the δ coefficients by **grd1**, **grd2**, **grd3**, and **grd4**. Unlike the other gradient types, the **gnz**, **gra**, **grb**, **grc**, and **grd** coefficients are fixed properties of the material and may not be made optimization variables. The design is controlled by the choice of which section of the blank is used for a Gradium element. Thus the relevant variables are the thickness (**th**) of the element and the offset of the front vertex (**goz**) of the element into the blank. When designing with a Gradium material, it is generally necessary to add

boundary conditions to the error function to ensure that the element lies completely within the available volume of the blank. The element thickness, blank thickness, and offset are all available as optimization operands.

Prm Axial and Elliptical Gradient (AEG)

Prm Axial and Sinusoidal Radial Gradient (SNS)

Prm Axial and Tapered Radial Gradient (TAP)

These three gradient types are combinations of an axial gradient with another gradient that improves the accuracy of real-world modeling. The data entry spreadsheet is similar for all, but the row for input of the elliptical, sinusoidal, or tapered factors is tailored to the particular type.

Axial and elliptical gradient - This gradient is a generalization of the axial and radial gradient, with the modification that, in the transverse direction, the loci of constant refractive index are, in general, elliptical. The index distribution is given by

$$n(r', z) = n_0 + \mathbf{nz1}z + \mathbf{nz2}z^2 + \mathbf{nz3}z^3 + \mathbf{nz4}z^4 + \mathbf{nr1}r'^2 + \mathbf{nr2}r'^4 + \mathbf{nr3}r'^6 + \mathbf{nr4}r'^8 \quad (3.66)$$

where

$$r'^2 = \mathbf{nrxx}^2 + \mathbf{nryy}^2 \quad (3.67)$$

For a cylindrical index distribution, either **nrx** or **nry** can be made zero.

Surface 1						
GDT AEG		Step Size:		0.100000		
NRX	1.000000	NR1	1.000000			
WVN	Wavelength	NZ1	NZ2	NZ3	NZ4	
1	0.587560	0.000000	0.000000	0.000000	0.000000	
2	0.486130	0.000000	0.000000	0.000000	0.000000	
WVN	Wavelength	NR1	NR2	NR3	NR4	
1	0.587560	0.000000	0.000000	0.000000	0.000000	
2	0.486130	0.000000	0.000000	0.000000	0.000000	
GRIN coordinate system decentration/tilt: Decenter, then Tilt						
	GDY	GDZ	GTA	GTB	GTC	
	0.000000	0.000000	0.000000	0.000000	0.000000	
Delete GRIN Medium						

Axial and sinusoidal radial - This gradient is similar to the axial and radial gradient, but the transverse coordinate r' is a sinusoidal function of the distance into the medium z . This distribution may be used, for example, to model and tolerance aspects of the gradient formation process. The index distribution is given by

(3.68)

$$n(r', z) = n_0 + \mathbf{nz1}z + \mathbf{nz2}z^2 + \mathbf{nz3}z^3 + \mathbf{nz4}z^4 \\ + \mathbf{nr1}r'^2 + \mathbf{nr2}r'^4 + \mathbf{nr3}r'^6 + \mathbf{nr4}r'^8$$

where

(3.69)

$$r'^2 = \left[1 + \mathbf{sva} \sin\left(\frac{z}{\mathbf{svp}} + \mathbf{svf}\right) \right] (x^2 + y^2)$$

Axial and tapered radial - This gradient is similar to the axial and radial gradient, but the transverse coordinate r' is a linear function of the distance into the medium z . This distribution may be used, for example, to model and tolerance aspects of the gradient formation process. The index distribution is given by

(3.70)

$$n(r', z) = n_0 + \mathbf{nz1}z + \mathbf{nz2}z^2 + \mathbf{nz3}z^3 + \mathbf{nz4}z^4 \\ + \mathbf{nr1}r'^2 + \mathbf{nr2}r'^4 + \mathbf{nr3}r'^6 + \mathbf{nr4}r'^8$$

where

(3.71)

$$r'^2 = \left[1 + (\mathbf{tas}z + \mathbf{tao}) \right] (x^2 + y^2)$$

Prm Spherical Gradient (SPH)

This index distribution is spherically symmetric about a specified point. The form of the index distribution is

(3.72)

$$n(r') = n_0 + \mathbf{nr1}(\mathbf{sgc} - r') + \mathbf{nr2}(\mathbf{sgc} - r')^2 + \mathbf{nr3}(\mathbf{sgc} - r')^3 \\ + \mathbf{nr4}(\mathbf{sgc} - r')^4$$

where

(3.73)

$$r' = \sqrt{x^2 + y^2 + (z - \mathbf{sgc})^2}$$

sgc is the distance from the surface vertex to the center of refractive index symmetry. Note that like the other forms of gradient index distribution, n_0 is the refractive index at the surface vertex (*not* at the center of symmetry).

Surface 1						
GDT SPH		Step Size:		0.100000		
SGC		0.000000				
WVN	Wavelength	NR1	NR2	NR3	NR4	
1	0.587560	0.000000	0.000000	0.000000	0.000000	
2	0.486130	0.000000	0.000000	0.000000	0.000000	
GRIN coordinate system decentration/tilt:				Decenter, then Tilt		
	GDX	GDY	GDZ	GTA	GTB	GTC
	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
Delete GRIN Medium						

Prm Maxwell's Fisheye Lens (MXF)

This is a form of spherical gradient first described by J. C. Maxwell (1854) as a nontrivial example of an absolute imaging instrument. The index distribution is

(3.74)

$$n(r') = \frac{n_0}{1 + (r'/nr1)^2}$$

where

(3.75)

$$r' = \sqrt{x^2 + y^2 + (z - \mathbf{sgc})^2}$$

Surface 1							
GDT MXF		Step Size:		0.100000			
SGC	0.000000						
WVN	Wavelength	NR1					
1	0.587560	0.000000					
2	0.486130	0.000000					
GRIN coordinate system decentration/tilt:				Decenter, then Tilt			
	GDX	GDY	GDZ		GTA	GTB	GTC
	0.000000	0.000000	0.000000		0.000000	0.000000	0.000000
Delete GRIN Medium							

Prm Luneburg Lens (LBG)

This is a form of spherical gradient proposed by R. K. Luneburg (1964). This refractive index distribution focuses every parallel bundle of rays to a perfect point image. The index distribution is

(3.76)

$$n(r') = n_0 \sqrt{2 - (r'/nr1)^2}$$

where

(3.77)

$$r' = \sqrt{x^2 + y^2 + (z - \text{sgc})^2}$$

Surface 1							
GDT LBG		Step Size:		0.100000			
SGC		0.000000					
WVN	Wavelength	NR1					
1	0.587560	0.000000					
2	0.486130	0.000000					
GRIN coordinate system decentration/tilt:				Decenter, then Tilt			
	GDx	GDY	GDZ		GTA	GTB	GTC
	0.000000	0.000000	0.000000		0.000000	0.000000	0.000000
Delete GRIN Medium							

Prm User GRIN (arbitrary coefficients) (UGR)

This gradient type allows you to define your own refractive index distribution. You are free to define an arbitrary number of coefficients for use in computing the refractive index, and must supply a CCL command or dynamic link library that carries out the computation. For more information, please see the section “User-defined gradients” on page 554.

Prm Multilayer Coating (M)

OSLO has features to define and model multilayer coatings on lens surfaces. A single coating design may be applied to multiple surfaces on multiple lenses. Defining multilayer coatings is performed on the “Multilayer Coating Definition” spreadsheet displayed using “Lens>>Coatings>>Update Designs” (**mse** command). Please see the section “Coatings” on page 191.

Attach a multilayer to a surface by using the “Special” pop-up menu of the “Surface Data Spreadsheet”, and then selecting the “Multilayer Coating (M)” item, or use the **mco** command directly from the command line. (The **mcd** command deletes the coating from the surface.) The “Multilayer coating name” cell in the

“Multilayer Coating Data” Update surface data spreadsheet shows the active coating. Clicking the name cell displays a pop-up menu of available coatings.

The coating layers may have uniform or non-uniform thicknesses described by one of three models. The OSLO default assumes the layers are uniform, with thicknesses as entered. Each model for non-uniform thickness introduces a rotationally-symmetric thickness variation of the form

(3.78)

$$t(r) = t_0 T(r)$$

where r is the radial coordinate on the surface, $t(r)$ is the layer thickness at that point, t_0 is the nominal thickness as entered in the coating definition, and $T(r)$ is a dimensionless function of r . Note that $T(r)$ is the same for all layers in the stack, so at any given position, the thickness of each layer is changed by the same fractional amount.

The most general model, the polynomial model, is just a power series in r^2 , that is,

(3.79)

$$T(r) = LT0 + LT1r^2 + LT2r^4 + \dots + LT15r^{30}$$

By default, $LT0 = 1.0$ and $LTi = 0.0$ for $i = 1, \dots, 15$.

The other two models are based on analyses of coating uniformity on rotating planar substrates.² You must enter the distance from the evaporation source to the substrate (h) and the distance from the source to the substrate rotation axis (R). The source may be either a point source or a directed surface source (that is, the emission distribution falls off as $\cos \theta$).

For a directed surface source:

(3.80)

$$T(r) = \frac{(1 + R^2/h^2)^2 (1 + r^2/h^2 + R^2/h^2)}{\left[1 + r^2/h^2 + R^2/h^2 - 2(r/h)(R/h)\right]^{3/2} \left[1 + r^2/h^2 + R^2/h^2 + 2(r/h)(R/h)\right]^{3/2}}$$

2. H. A. Macleod, *Thin-Film Optical Filters*, Second Edition, McGraw-Hill, 1989, Chapter 10.

For a point source:

$$T(r) = \frac{2(1 + R^2/h^2)^{3/2}}{\pi[1 + (R+r)^2/h^2]^{3/2}} \Pi(\pi/2, -k^2, k) \quad (3.81)$$

Here, $k^2 = 4rR/[h^2 + (R+r)^2]$ and Π is the Legendre elliptic integral of the 3rd kind:

$$\Pi(\phi, n, k) \equiv \int_0^\phi \frac{d\theta}{(1 + n \sin^2 \theta) \sqrt{1 - k^2 \sin^2 \theta}} \quad (3.82)$$

The **thunf** CCL command will plot $T(r)$ for a specified surface.

Prm Polarization Element (Z)

The polarization element provides a general facility to modify the polarization state of a beam of light. You may specify the elements of a general 2 x 2 Jones matrix. Given an incident polarized wave, propagating in the z-direction, we write the x and y components of the electric field as the column (Jones) vector

$$\mathbf{E}_i = \begin{bmatrix} E_{ix} \\ E_{iy} \end{bmatrix} \quad (3.83)$$

where E_{ix} and E_{iy} are, in general, complex. The polarization element transforms the incident field \mathbf{E}_i into the transmitted field \mathbf{E}_t according to

$$\mathbf{E}_t = \mathbf{J} \mathbf{E}_i \quad (3.84)$$

where \mathbf{J} is the 2 x 2 Jones matrix

$$\mathbf{J} = \begin{bmatrix} J_A & J_B \\ J_C & J_D \end{bmatrix} \quad (3.85)$$

The z-component of the electric field, if any, is unchanged by the polarization element. The polarization element allows you to specify the amplitude and phase of the four complex elements of the Jones matrix, J_A , J_B , J_C , and J_D . These elements may be specified in the Polarization Element special data spreadsheet, or with the commands **jaa**, **jpa** (amplitude and phase in degrees of J_A), **jab**, **jpb** (amplitude and phase in degrees of J_B), **jac**, **jpc** (amplitude and phase in degrees

of J_C), and **jad**, **jpd** (amplitude and phase in degrees of J_D). The default values for a polarization element are $J_A = J_D = 1$, $J_B = J_C = 0$, i.e., the identity matrix.

Surface 1					
	AMPLITUDE	PHASE		AMPLITUDE	PHASE
JA	1.000000	0.000000	JB	0.000000	0.000000
JC	0.000000	0.000000	JD	1.000000	0.000000
Delete Polarization Element					

A tabulation of Jones matrices for common polarization elements (e.g., linear polarizers, wave plates, etc.) may be found in the section on Jones Calculus in Chapter 2 of the OSLO Optics Reference manual.

Prm Eikonal Surface (E)

The eikonal surface allows you to specify an optical component by its eikonal (characteristic) function, rather than construction data (radii, thicknesses, etc.). Using this type of surface you can, for example, study the behavior of a perfect lens without having to design a perfect lens. For information on preparing a CCL eikonal command, please see “User-defined Eikonals” in Chapter 16, page 555. For general information about optical design using eikonals and descriptions of the eikonal examples that are supplied with OSLO Premium, please see Chapter 10, “Eikonal Design” in the Optics Reference manual.

Prm Non-sequential Data (Q)

The Non-sequential data item on the special data pop-up menu only appears for surfaces that are within a non-sequential group. A non-sequential group can be created using the “Edit>>Non-sequential Group” menu option.

In OSLO, a non-sequential surface group is treated as a sub-group of surfaces that is associated with a selected surface in the sequential portion of the lens. The first surface of a non-sequential group is called the *entry port*. Surfaces in the non-sequential group itself are numbered in the usual way, although rays will not generally strike them in the given order. The last surface in the group is called the *exit port*.

When any traced ray reaches the entry port, a special ray tracing procedure is invoked that attempts to find a path through the non-sequential group to the exit port surface. Surfaces within the group will be traversed in whatever sequence is necessary to produce a real path to the exit port, i.e., the path length from each surface to the next surface through the group is required to be greater than or equal to zero. Total internal reflection is allowed and the same surface may be visited multiple times.

The thickness of the group surfaces is not used for surfaces within a non-sequential group. The position and orientation of each surface is specified in the local coordinate system of the entry port surface, using the three decentration coordinates (**dcx**, **dcy**, and **dcz**) and the three tilt angles (**tla**, **tlb**, and **tlc**), as entered in the local/global coordinates spreadsheet.

Non-sequential surface profile data (radii of curvature, aspheric data, etc.) are entered in the same way as normal surface data.

If an aperture radius is specified for a group surface, it is assumed to apply to that part of the surface for which $(1 + \text{cc}) \text{cv} z < 1$, where **cc** is the conic constant and **cv** is the curvature of the surface. Only that part of the surface for which the preceding condition is true is assumed to be present. Therefore, the surface is open. A spherical surface, for example, will be treated as a hemisphere. If no aperture radius is specified for a spherical or elliptical group surface (or an x-toric surface with a spherical or elliptical xz cross section), the entire area of the closed surface is assumed to be present. Note that since paraxial rays are not traced through non-sequential groups, there are no solved apertures in the group, so non-zero aperture radii must be entered explicitly.

If an aperture is specified for a group surface, a ray that falls outside the aperture is free to pass by the surface if aperture checking is off for that surface. If aperture checking is on, the region exterior to the surface is regarded as an infinite baffle, and a ray that reaches the surface via a real path will be stopped.

A number of special rules apply to the tracing of rays through a non-sequential group. In particular, since a surface may be approached from either side, it is necessary to specify how rays are to be reflected, refracted, or obstructed depending upon the direction from which the surface is approached. If a special action is not specified, the action defaults to whatever is specified by the normal lens data.

In the local coordinate system of a surface in a non-sequential group, the surface may be viewed as dividing space into two regions. Since the surface always passes through the local origin, a point on the local z-axis, slightly displaced from the origin in the negative z-direction, will lie in one of these regions, which we call the negative region. Conversely, a point on the local z-axis that is slightly displaced from the origin in the positive z-direction will lie in the other region, which we call the positive region. A ray is said to be traveling *to positive* if it approaches the surface from a point in the negative region. It is said to be traveling *to negative* if it approaches the surface from a point in the positive region.

The glass specification for a surface specifies the medium into which a ray is refracted at a surface. When rays are known to traverse the surfaces of a lens in sequential order, this information is sufficient to trace the rays. For surfaces in a non-sequential group, however, the sequence is not known in advance. Moreover, the same surface may be visited more than once from either its negative or positive side.

Special actions are specified using the Non-sequential data item in the Special data pop-up menu. You can specify both a special action and a condition under which the action is to be taken. One action may be specified for ordinary rays and one action may be specified for the reference ray traced from a new object point. When no action is specified, or the condition specified for the action does not apply, the normal glass specification for the surface determines the action. If a special action is specified for ordinary rays but not for reference rays, the action specified for ordinary rays will apply to reference rays as well. The action and condition are chosen by clicking the appropriate cell in the spreadsheet and selecting the desired item from the pop-up menu.

Surface 2 (in Non Seq Grp)	
Element Id Number:	<input type="text" value="0"/>
Ordinary Ray:	
Action:	<input type="text"/>
Condition :	<input type="text"/>
Reference Ray:	
Action:	<input type="text" value="Same as ordinary ray"/>
Condition :	<input type="text"/>

Pickup from group surf
 Reflect
 Obstruct
 Pass undeviated
 Trace to exit port only
 No action

Surface 2 (in Non Seq Grp)	
Element Id Number:	<input type="text" value="0"/>
Ordinary Ray:	
Action:	<input type="text" value="Pickup from group surf"/> Surface: <input type="text" value="1"/>
Condition :	<input type="text"/>
Reference Ray:	
Action:	<input type="text" value="Same as ordinary"/>
Condition :	<input type="text"/>

To negative
 To positive
 Before n-th hit
 After n-th hit

The “pickup from group surface” action means that the ray is to be refracted into the medium defined by the glass entry on the specified group surface. For example, suppose that surface 3 has glass BK7 and group surface 4 has an AIR specification. If a ray approaches surface 4 from negative to positive, it refracts from BK7 into air. This is the normal action. If the ray is subsequently reflected and arrives again at group surface 4 traveling from positive to negative, it should refract from air into the glass BK7. Since AIR is specified for group surface 4, it is necessary to define a special action for group surface 4 for this case. The action would be “pickup from group surface 3” and the condition would be “to negative”.

The Optics Reference manual gives additional discussion and an example of a non-sequential group.

Prm User Surface (U)

The user-defined surface allows you to interrupt the internal ray trace algorithms in OSLO and take control of the ray trace. Basically, the internal ray trace computes the ray trajectory and optical path length up to the surface immediately preceding the user surface, interrupts its trace, and calls a user-written CCL or DLL ray trace routine specified for the surface. After the CCL or DLL routine is executed, control is returned to OSLO, which completes the trace through the remaining surfaces. For detailed information on user-defined surfaces, please see the section “User-defined surfaces” on page 551.

User Ray Trace (USR)

OSLO's internal ray trace computes the ray trajectory and optical path length up to the surface immediately preceding the user surface, interrupts its trace, and calls the user-written CCL ray trace routine specified for the surface. Upon completion

of the execution of the user CCL command, OSLO regains control of the ray trace and uses the new values of the above variables to continue the ray trace through the system (or to the next user surface, where the user surface ray trace process would be repeated).

A sample user ray trace is the CCL command **utrace** in the file `usraytr.ccl` in the "...\\public\\ccl" directory.

For a complete explanation of the User Ray Trace surface see Chapter 16, page 551

Use Sag Surface (UDS)

There is a user-written CCL or DLL routine that determines a surface type as a user-defined sag surface. This surface is similar to the user ray trace surface, but differs in functionality. With the user ray trace, the user routine must both trace the ray to the surface and "bend" (for example, refract or reflect) the ray. With a user-defined sag surface, you specify the surface profile; OSLO finds the intersection of the ray with the surface, then bends the ray. Thus, the user-defined sag surface may be refracting, reflecting, diffractive, etc., depending upon the normal specification of the glass of the surface and any other special data defined for the surface. The CCL or DLL command computes the shape of the surface.

The surface is defined by the equation³

$$F(x, y, z) = 0 \quad (3.86)$$

where x , y , and z are the coordinates in the local coordinate system of the surface. The surface is defined in this way, rather than an explicit expression for the sag, for purposes of generality; there are many surfaces for which an explicit expression for the sag is either difficult or impossible to obtain. If the surface you want to model is given in terms of its sag (z) at aperture points (x , y), that is, the equation of the surface is $z = s(x, y)$, the appropriate implementation of this surface as a user-defined sag surface is

$$F(x, y, z) = z - s(x, y) = 0 \quad (3.87)$$

During the ray trace, the command will be supplied with x , y , and z coordinate values; the task for the command is to compute the value of $F(x, y, z)$ and the three derivatives of F (*direction numbers* for the surface normal), $\partial F/\partial x$, $\partial F/\partial y$, and $\partial F/\partial z$, at the point (x , y , z). Note that if $(\partial F/\partial x)^2 + (\partial F/\partial y)^2 + (\partial F/\partial z)^2 = 1$, the direction numbers are the direction cosines for the surface normal vector; it is not necessary, however, for the CCL/DLL command to normalize the derivatives. For the most accurate result, it is preferable that the direction numbers be computed explicitly from the definition of the function F . If this is not possible, you may need to use the finite difference approximation for the derivatives, for example,

$$\frac{\partial F}{\partial x} \approx \frac{F(x + \Delta x, y, z) - F(x, y, z)}{\Delta x} \quad (3.88)$$

or, to reduce the truncation error, but at the expense of more function evaluations,

3. G. H. Spencer and M. V. R. K. Murty, "General ray-tracing procedure," J. Opt. Soc. Am. **52**, 672–678 (1962).

$$\frac{\partial F}{\partial x} \approx \frac{F(x + \Delta x, y, z) - F(x - \Delta x, y, z)}{2 \Delta x} \quad (3.89)$$

Designate a surface as a user-defined sag surface, using the Update surface data spreadsheet Special pop-up menu, then select User Surface >> User-defined sag. You will be prompted for the number of **UTi** coefficients you want to define. These coefficients are used to pass parameter values to your CCL/DLL command (in the *Uds_ucf*] array) and are also made optimization variables and operand components. In the spreadsheet, enter the name of the command that computes the value of *F* and its derivatives, and the values of the **UTi** coefficients.

When the equation defining the surface needs to be evaluated, the surface number and the *x*, *y*, and *z* coordinates are placed in the global variables, *Uds_sfc*, *Uds_x*, *Uds_y*, and *Uds_z*. The command computes the values of *F*(*x*, *y*, *z*) and the direction numbers at (*x*, *y*, *z*), and sets the values of the global variables *Uds_f*, *Uds_dfdx*, *Uds_dfdy*, and *Uds_dfdz*. If the surface equation can not be evaluated, the variable *Uds_err* is set to a non-zero value and the ray trace for that ray is halted.

Prm Interferometric Deformation (I)

The interferogram file provides a mechanism for attaching surface or wavefront deformation or intensity apodization data to a lens surface or to a pupil. This file is called an interferogram file (or INT file) because one common use of the file is to transfer interferometrically measured data from an interferometer to a design program like OSLO. Two of the interferometer vendors (Zygo and Wyko/Veeco) directly support this file format (which was originally developed by Optical Research Associates). Although interferometers are one source of INT files, they are not the sole source; the files are simple ASCII text files, so they may be created with any text editor, from a CCL command, or from a stand-alone program. (Note that these files are often called INT files because of the connection with interferometry, but they have no connection with the C and CCL "int" data type.)

File format

The interferogram file, as mentioned, above is a text file. There are two classes of files:

1. Grid data
2. Zernike polynomial coefficient data

INT files may have comment lines, with the restrictions that all comments must come at the beginning of the file and that comments stand on lines by themselves. A comment line begins with the exclamation point '!' character. A file may have any number (including zero) comment lines. Following the comments (if any), the file may be broken down into three main sections:

1) Title

The file title (not to be confused with the name of the file) is the first line in the file that does not start with the comment character '!'. Each INT file must have a title line, which may be up to 80 characters in length.

2) Parameters

After the title, there is a single line of text that contains information about the type of data represented by the file. The format of this line depends upon the type of data. In the format descriptions below,

- Items in CAPITAL LETTERS represent keywords that are used in the parameter line,
- Items in *italics* represent values that are specific to that file,
- Items in angle brackets < > are optional, and
- Default values will be used if they are not present in the file.

The order of the keywords on the parameter line is not important. The item SUR or WFR or FIL means that one of the three keywords (SUR, WFR, FIL) is present. One of these three keywords is required, as it indicates what type of data is represented by the file.

Keyword	Description
SUR	Surface deformation data
WFR	Wavefront deformation data
FIL	Intensity apodization filter data

Rectangular or linear grid

The parameter line has the format:

```
GRD xsize ysize SUR or WFR or FIL WVL wavelength SSZ scalesize <NDA no_data_value> <XSC
xscale><NNB>
```

Xsize and *ysize* are the number of points in the data grid in x and y respectively. A linear grid (the data value is constant in one direction) is represented by a value of 1 for either *xsize* or *ysize*.

Wavelength is the value of the wavelength, in microns, at which the interferogram was measured. This item is not required if the file represents an intensity apodization filter (FIL).

Scalesize is the grid data value that represents either one wavelength (at WVL *wavelength*) of surface deformation (SUR), one wavelength of wavefront deformation (WFR), or an intensity transmission of one (FIL).

No_data_value is the grid data value that represents "no dat", i.e., a region outside the measurement area. If this item is omitted from the file, the default value is 32767. Rays that are incident upon areas corresponding to the "no data" value are blocked.

Xscale is the ratio of the physical extent of the grid in the x direction to the physical extent of the grid in the y direction. A square grid has an *xscale* value of 1.0, which is the default value that will be used if this item is omitted from the file.

If the keyword NNB is present in the parameter line, nearest-neighbor interpolation, rather than the default of linear interpolation, is used.

Radial grid (rotationally symmetric data)

The parameter line has the format

GRD *rsiz*e R SUR or WFR or FIL WVL *wavelength* SSZ *scalesize* <NDA *no_data_value*> <XSC *xscale*>

Rsize is the number of data points along the radius in the data grid.

The other items have the same meaning as for rectangular grids.

Zernike polynomial data

The parameter line has the format:

ZFR *numofzterms* SUR or WFR or FIL WVL *wavelength* SSZ *scalesize* <XSC *xscale*>

or

ZRN *numofzterms* SUR or WFR or FIL WVL *wavelength* SSZ *scalesize* <XSC *xscale*>

Numofzterms is the number of Zernike polynomial coefficients in the file.

There are two "styles" for the Zernike INT files: fringe (ZFR) and standard (ZRN).

The differences between these two styles will be explained in the section "Zernike INT files" on page 115.

3) Data

The third section of the INT file is the actual data. The data section consists of multiple lines of data, corresponding to the number of required values as specified in the parameters for the file. There may be any number of data values on a single line of the file, separated by one or more blank spaces, subject to the restriction of a maximum of 4096 characters on a line.

For grid data (rectangular, linear, or radial), the data values are integers. OSLO allows the use of long (i.e., 4 byte) integers for the data values. For rectangular grids, there must be a total of *xsize* × *ysize* values; for radial grids there must be *rsiz*e values. (Note that the data in the file is in "free format"; each line of the file does *not* necessarily correspond to a single row of data for the grid.) For rectangular grids, the data is ordered from −*x* to +*x*, starting at +*y*. For radial grids, the data is ordered from the origin outwards.

For Zernike polynomial coefficient data, the values are integers or real numbers, and there must be *numofzterms* values.

Assigning INT files to surfaces or pupils

To assign an INT file to a surface, click the SPECIAL button for the desired surface in the surface data spreadsheet, and select Interferometric Deformation from the pop-up menu. The deformation from the INT file is in addition to the base surface shape defined for the surface and is measured normal to the nominal surface.

To assign an INT file to a pupil, first open the Field Points Set spreadsheet. Set the spreadsheet style to Advanced, if it is not already. Click the button on the far right for the field point at which you wish to assign an INT file. Select the desired location (entrance pupil or exit pupil) from the pop-up menu. Only wavefront deformation or intensity apodization filters may be assigned to pupils. Note that the INT file will only be used for the field point to which it is assigned; you may, however, assign the same INT file to more than one field point, if appropriate.

The first item to enter is the name of the INT file. By default, OSLO will assume that INT files have the file extension *.int and are located in the directory named in the Int_file_directory preference. The default value for this directory is your "...private/int" directory. Clicking the file name cell will pop-up a menu (or list) of all

of the *.int files in this directory, from which you can choose the desired file. If the INT file you want is not in the Int_file_directory directory, you must enter the full path to the file.

Once you have assigned an INT file to a surface or a pupil, you need to set the values of several data items that relate the INT file data to the surface or system.

Scale factor for data

This is an overall scaling factor and is separate from the SSZ item in the file itself. A typical use of this scale factor is to relate the testing conditions to the usage conditions. For example, if the interferogram was the result of a double-pass test and the surface or pupil is used in single pass, the appropriate scale factor is 0.5. A reversal in orientation may require a scale factor of -1.0 .

Radius of data

This is the actual physical length (in lens units) that corresponds to the unit length of the INT file. For grid INT files, the data lies in a square of unit semi-side-length; for Zernike INT files, the data is defined for a circle of unit radius.

Center of data

By default, the INT file is centered at the origin of the local coordinate system of the surface or pupil. You may shift the INT file data by entering the x and y values (in lens units) of the center of the INT file data.

Data coordinates and Rotation Angle

By default, the orientation of the x and y axes of the INT file are the same as the local x and y axes of the surface or pupil. If desired, the data may be mirrored (interchange $+x$ and $-x$ or $+y$ and $-y$ values) and/or rotated. (A positive rotation moves the $+x$ axis toward the $+y$ axis. Any mirroring is performed before rotation.)

Zernike INT files

As mentioned above, two different formats for Zernike polynomials are supported: Fringe and standard. To conform with standard usage in interferometry, the polar angle q is measured from the $+x$ axis.

The Fringe Zernike polynomial set consists of the 37 polynomials listed in the Optics Reference. Thus a Fringe Zernike (ZFR) INT file has a maximum of 37 data values.

The standard Zernike polynomial (ZRN) uses a different numbering than the Fringe Zernike polynomial and may contain an arbitrary number of terms. The ordering of the standard Zernike set is indicated in the table below, which contains the first 15 terms.

Term Number	Standard Zernike Polynomial
0	1
1	$r\cos(q)$
2	$r\sin(q)$

3	$r^2 \cos(2q)$
4	$2r^2 - 1$
5	$r^2 \sin(2q)$
6	$r^3 \cos(3q)$
7	$(3r^3 - 2r) \cos(q)$
8	$(3r^3 - 2r) \sin(q)$
9	$r^3 \sin(3q)$
10	$r^4 \cos(4q)$
11	$(4r^4 - 3r^2) \cos(2q)$
12	$6r^4 - 6r^2 + 1$
13	$(4r^4 - 3r^2) \sin(2q)$
14	$r^4 \sin(4q)$

Toolbar Icons

These icons can be conveniently accessed to perform some common tasks on the data in the spreadsheet window. Many of these tasks can be implemented in other ways.



Insert row above selected row(s) (Ctrl+I): This is the same as choosing “Insert Before” from the row button pop-up menu (page 32).



Reverse selected rows: Changes the orientation of the selected surfaces. This is the same as choosing “Reverse” from the row button pop-up menu (page 32).



Show only selected database rows: This hides all non-selected rows in the database. You can use the “Show All Rows” item on the main database pop-up menu to reverse this behavior (Chapter 6, page 164).



Cut selected rows to the clipboard: This is the same as choosing “Cut” from the row button pop-up menu (page 30).



Copy selected rows to the SS clipboard: This is the same as choosing “Copy” from the row button pop-up menu (page 30).



Paste SS clipboard above first selected row: This is the same as choosing “Paste” from the row button pop-up menu (page 30).

Spreadsheet Buttons



Gen (General Conditions)

The **general operating conditions** spreadsheet contains items that control various aspects of how your lens is to be evaluated. The short forms of the associated commands, which are also the names of the CCL/SCP global variables, are given in parentheses below.

The spreadsheet is divided in two part: **Basic** and **Advanced**.

Basic

Evaluation mode (afo)

The evaluation mode controls the computation and display of paraxial and ray data. To change the evaluation mode, click the options button. The choices for evaluation mode are *Focal* and *Afocal*. *Focal* mode should be used for focal systems (i.e., an image is formed at a finite distance); ray data will be reported as transverse quantities. *Afocal* mode should be used for afocal systems (i.e., the image is infinitely distant); ray data will be reported as angular (direction tangent) quantities.

Units (uni)

Lens units are the unit of measure in which the lens is described. The units may be changed by activating the units cell by clicking on it to display a pop-up menu containing the choices for units: millimeters, centimeters, inches, micrometers, or meters).

Ray aiming mode (raim)

For more information, see the discussion of ray aiming modes.

- **Beam half-angle (xaba)**

If extended-aperture ray aiming mode is used, this operating condition defines the (half) cone angle of the light emitted by the source or object.

Wavefront reference sphere position (wrsp)

This controls the location of the reference sphere used for the calculation of optical path differences. There are three possible choices for reference sphere position:

- **Exit pupil:** the reference sphere is located at the real exit pupil position for each field position
- **Infinity:** the reference sphere is located at infinity
- **Last surface:** the vertex of the reference sphere is located at the intersection of the reference ray with the next-to-last surface in the lens

Aperture checking (apck)

- **On:** apertures that have been designated as checked apertures will be checked for vignetting for all rays traced by OSLO, except for reference rays and optimization rays.

- **Off:** apertures will only be checked for spot diagram rays.
- **Include GRIN ray segments (grck):** all ray segments in gradient index media will be checked for vignetting, using the aperture radius (if a checked aperture) of the surface on which the GRIN medium is defined. Without this option, the ray will only be checked against the aperture upon refraction into the medium.

Advanced

Designer (des)

The Designer is a 10 character field that can be used to identify the designer of the lens, if desired. The default designer name assigned to new lenses is controlled by the **designer** preference (**dsgn**). See the section “Set Preference / Show Preference” on page 174.

Aberration mode (amo)

The aberration mode controls how aberration surface contributions are displayed. To change the aberration mode, click the options button. A pop-up menu will be displayed with the three choices for the aberration mode: Transverse, Angular, or Unconverted. Transverse mode is appropriate for focal systems, since the aberrations will be expressed as lateral ray errors. Angular mode is appropriate for afocal systems, since the aberrations will be expressed as angular ray errors. Unconverted mode, which reports the direct surface contributions, can be used for both focal and afocal systems.

Std **Prm** Solves in alternate configurations (cslv)

- If this option is **On**, solves will be carried out in all configurations.
- If the option is **Off**, solves will be carried out only in the base configuration (i.e., configuration 1).

OPD in waves (opdw)

- If this option is **On**, optical path differences will be measured in wavelengths.
- If this option is **Off**, optical path differences will be measured in the current lens units.

Zernike polynomial reference axis (zrxr)

This operating condition selects the reference axis for the polar angle θ used in the definition of the Zernike polynomials.

- If the reference axis is Y, the angle θ is measured from the +y axis, and a positive angle is a rotation toward the +x axis.
- If the reference axis is X, the angle θ is measured from the +x axis, and a positive angle is a rotation toward the +y axis.

This reference axis is used for Zernike polynomial analyses of wavefronts and for both asymmetric Zernike phase and sag surfaces.

Global reference surface for ray data (gcs)

This is the number of the surface that establishes the global coordinate system for ray output data, when global coordinates have been requested. This surface is used as the global reference for the Evaluate>>Single Ray Trace (**tra**) command

and for global coordinate ray operands. This surface is also used as the global reference surface for the global coordinate transformation matrices.

Prm Symmetry State

OSLO makes use of the symmetry, if any, of the current lens in order to maximize the efficiency of operation. For example, if the lens possesses bilateral (i.e., left-right) symmetry, only the rays for which $FX \geq 0$ need to be traced for a spot diagram; the properties of the rays in the other half of the pupil may be deduced from symmetry considerations. For all of the "built-in" surface data types, OSLO is able to determine the overall symmetry state of the lens. This is not possible, however, for user-defined items. If a lens contains a user-defined item (User-Defined Ray Trace surface, User-Defined Diffractive surface,...etc.), OSLO assumes no symmetry in the lens.

In order to maintain efficiency in the presence of user-defined surfaces, a new Symmetry State general condition has been added to OSLO. (This condition may be found in the Advanced general conditions.) By default, the determination of the symmetry state of the lens is automatic, according to the rules discussed above. If you have a user-defined surface and the overall lens retains bilateral symmetry, you can change this condition to force OSLO to assume that the lens is bilaterally symmetric, and thus not trace unnecessary rays. Note that OSLO does not do any checking to ensure the symmetry of the lens; in order to avoid erroneous results, it is up to you to specify this case only if the lens is indeed bilaterally symmetric.

Conversely, there may be situations where you want OSLO to treat a symmetric lens as asymmetric (in order to override some consequences of symmetry assumed during tolerancing, for example). In this case, you can set the Symmetry State condition to force OSLO to assume no symmetry in the lens.

Evaluation z-axis

By default, OSLO performs the bulk of its image evaluation in the local (x, y, z) coordinate system of the image surface. When the image surface is a plane, you generally only need to worry about the x and y coordinates, since all of the rays have a z coordinate of zero. However, when the image surface is curved, the rays now have x, y, and z coordinates. OSLO has added an option to reorient the coordinate system for image evaluations on non-planar image surfaces. The coordinate system used may be either the default (x, y, z) of the image surface, or the coordinate system may move with each field point so that the z-axis of the evaluation coordinate system points along the normal to the image surface at the intersection point of the reference ray with the image surface. The resulting x-y plane of the evaluation coordinate system is tangent to the image surface at the reference ray intersection. Note that this option means that each image point has its own coordinate system. Generally, though, orienting the evaluation z-axis along the surface normal is preferred, since it removes much of the effect of the obliquity between the image surface and the evaluation x-y coordinates. Regardless of which option is chosen for the evaluation coordinate system, the ray coordinates are evaluated on the actual image surface; this condition only affects the coordinate system in which they are measured.

Ray aiming type (epxy)

This operating condition allows for the choice of **Aplanatic** (the default) or **Paraxial** ray aiming.

Source astigmatic distance (sasd)

A non-zero value for this operating condition defines the source to be astigmatic, i.e., rays appear to emanate from different source points in the XZ and YZ azimuths. The source astigmatic distance is the longitudinal separation of the XZ and YZ apparent source points. If the distance is positive the XZ source point is closer to surface 1 than the YZ source point. An astigmatic source may only be used with on-axis object points. To simulate off-axis performance, tilt the optical system at surface 1.

Temperature (tem) and Pressure (pre)

These operating conditions set the temperature of a lens (in degrees Celsius) and the atmospheric pressure (in atmospheres) of its air spaces. Changing the temperature changes the absolute refractive indices of the glasses in the lens; since the refractive index of the air changes with temperature and pressure, the relative (to air) refractive indices of the glasses also change. Note that OSLO uses relative refractive indices for all computation, so the index of AIR is always given as 1.0 for all temperatures, pressures, and wavelengths. To specify a vacuum as the medium between glass elements, simply set the pressure to zero. Note that data for thermal variation of refractive index are currently available in OSLO only for Schott glasses. Thermal expansion is applied to the radii of curvature, thicknesses, aspheric and diffractive surface coefficients, air spaces, and aperture radii of a lens in the base configuration (configuration 1) when the temperature is changed. Note that thermal changes in the refractive index are also applied when the temperature changes.

Prm Polarization raytrace (pzrt)

This button displays the current status of the polarization raytrace and also allows access to the polarization conditions spreadsheet.

Prm Image space spot diagram (sdis)

The default condition for spot diagrams is that the rays are aimed at the entrance pupil in such a way that the rays have equal direction cosine increments in object space. As a result of pupil aberrations, rays traced in this way will, in general, be unequally spaced (in terms of direction cosines) in the image space. If this option is **On**, the spot diagram rays are iteratively traced so that they are equally spaced in image space. Because of the iterative process, of course, the spot diagram will take a longer period of time to trace. However, the use of this option is recommended when maximum accuracy for spot diagram analysis is desired. This option is particularly useful when performing Zernike polynomial analysis of the wavefront.

Prm Use diffraction efficiency calculation (sdde)

If this option is **On**, the diffraction efficiency of diffractive surfaces is considered when calculating the weight of each ray in the spot diagram.

Setup (Paraxial Setup)

The paraxial setup spreadsheet allows you to change those aspects of system data that affect the paraxial properties of the lens, e.g., image size, f-number, magnification, ...etc. The spreadsheet offers three categories of data that you can specify:

- Aperture
- Field
- Conjugates

Aperture		Field		Conjugates	
Entr beam rad*	6.250000	Field angle *	20.000000	Object dist	1.0000e+20
Object NA	6.2500e-20	Object height	-3.6397e+19	Object to PP1	1.0000e+20
Ax. ray slope	-0.124999	Gaus image ht	18.198709	Gaus img dist	43.080554
Image NA	0.124999			PP2 to image	50.000541
Working f-nbr	4.000043			Magnification	0.000000
Aperture divisions across pupil for spot diagram:					17.030000
Gaussian beam	No	1/e ² radius on srf 1: sdx 1.000000 sdy 1.000000			

Aperture

There are five options available for specifying the lens aperture (the size of the axial beam of light traced through the lens). If you change one of them, this becomes the *defining* aperture value and the other aperture parameters will be recalculated to reflect the new *defining* aperture. Note that the *defining* aperture parameter is marked with an asterisk (*).

- **Entrance beam radius**

The entrance beam radius is the radius of the circular beam that is incident upon surface 1 of the lens. Since the entering beam must always be of a finite size, this option is valid for both infinite and finite conjugate systems.

- **Object numerical aperture**

Numerical aperture (NA) is defined as $n \sin(\theta)$, where n is the refractive index and θ is the (half) angle of the converging or diverging cone of light. Specifying the object NA is valid only for finite conjugate systems.

- **Axial ray slope**

The axial ray slope is the angle of the paraxial axial ray in image space. This option is valid if the system forms a finite image.

- **Image numerical aperture**

Image NA is equal to $n \sin(\theta)$ in image space. Since this is a paraxial calculation (i.e., aberrations are neglected), it is assumed that the system is aplanatic (i.e., the Abbe sine condition is satisfied) for the purposes of computing image NA. This option is valid if the system forms a finite image.

- **Working f-number**

Working f-number is defined as $1/(2 \text{ NA})$, where NA is the image space numerical aperture. Working f-number is equivalent to the use of f-number as (focal length/entrance pupil diameter) for infinite conjugate systems. Working f-number may be specified if the system forms a finite image. As noted above, this is a paraxial calculation and assumes an aplanatic system.

Specifying the paraxial aperture of a lens is done by entering either the entrance beam radius (**ebr**) or the numerical aperture of the object (**nao**). Both of these specifications are object space quantities. As an alternative, three image space aperture specifications have been added to OSLO. You may specify either the image space paraxial axial ray slope (**puk**), the image space numerical aperture (**nap**), or the working f-number (**fno**). Entering a value for any of these three items

in the paraxial setup spreadsheet will force the size of the entering beam to automatically be adjusted so that the image space specification is met.

The calculation of these quantities is based on a paraxial raytrace, so these image space specifications should not be used with a system for which paraxial optics is not valid. Also, axial ray angle and thickness solves (except for thickness solves on the surface preceding the image surface) are not allowed if an image space aperture specification is used. An image space aperture specification just means that the size of the entering beam is changed; the lens construction data is not affected. Thus, using an axial ray slope aperture specification is not equivalent to using an angle solve on the last refracting surface of a lens, as the aperture specification will not maintain the focal length of the lens. Generally, to optimize a lens with an image space aperture specification, you will want to have focal length control in the error function.

Field

There are three options available to specify the field-of-view of the lens. If you change one of them, the others will be recalculated to reflect the new field-of-view:

- **Field angle**

For infinite conjugate systems, the field angle is the half-angle, in degrees, subtended by the object at the entrance pupil of the lens.

- **Object height**

The object height is the size of the object, measured from the axis to the end of the object. This option is only valid for finite conjugate systems.

- **Image height**

The image height is the size of the image, measured from the axis to the end of the image. This option is valid if the system forms a finite image.

With regard to the field of view, that is specified by either the field angle (ang) or the object height (obh), both object space quantities. An image space field specification, the Gaussian image height (**gih**), has also been added to OSLO. If you enter a value for the Gaussian image height in the paraxial setup spreadsheet, the field of view will be adjusted to produce the specified image height. (Note that this is the Gaussian image height, which is not necessarily the same as the paraxial chief ray height at the image surface, unless the image surface is at the paraxial focus.) This too is based on a paraxial raytrace, so it should not be used if paraxial optics is not valid. Chief ray angle and thickness solves (except for thickness solves on the surface preceding the image surface) are not allowed with a Gaussian image height specification. Use of the Gaussian image height specification is useful when the image format size is known but the exact focal length is not. Also, for a zoom lens working with a fixed image size, using a Gaussian image height specification eliminates the need for field angle or object height configuration items.

Conjugates

There are five available options for setting the conjugates of the lens. Changing any one item will result in the recalculation of the other items in the column that are affected:

- **Object distance**

This is the distance from the object (surface 0) to surface 1. If you change this value, the thickness of surface 0 will be changed to the specified value, and

the next-to-last thickness will be adjusted so that the nominal location of the image surface (before any defocus) is at the paraxial image plane. For an infinitely distant object, use a value of 1.0e20.

- **Object to first principal point distance**

This is the distance from the object (surface 0) to the first principal point of the lens. If you change this value, the thickness of surface 0 will be changed to the appropriate value, and the next-to-last thickness will be adjusted so that the nominal location of the image surface (before any defocus) is at the paraxial image plane.

- **Image distance**

This is the distance from the next-to-last surface to the paraxial image. If you change this value, the thickness of the next-to-last surface will be changed to the specified value, and the thickness of surface 0 will be adjusted so that surface 0 and the image surface (before defocus) are paraxially conjugate. This option is valid for focal systems.

- **Second principal point to image distance**

This is the distance from the second principal point to the paraxial image. If you change this value, the thickness of the next-to-last surface will be changed to the appropriate value, and the thickness of surface 0 will be adjusted so that surface 0 and the image surface (before defocus) are paraxially conjugate. For an infinitely distant object, the value of the PP2 to image distance is equal to the effective focal length. This option is valid for focal systems.

- **Magnification**

This is the value of the paraxial magnification (lateral magnification for focal systems, angular magnification for afocal systems). If you change this value, the thicknesses of both surface 0 and the next-to-last surfaces will be adjusted so that the object and image surfaces (before defocus) are paraxially conjugate, with the requested magnification. Specifying the magnification is valid only for focal systems (since angular magnification for afocal systems cannot be changed simply by changing the thicknesses of surface 0 and the next-to-last surface).

System aperture settings

Aperture divisions across pupil (sdad)

This is the number of grid intersections across the diameter of the entrance pupil. Increasing the number of aperture divisions (and thereby tracing more rays) should increase the accuracy of spot-diagram-based calculations, but will also increase the computation time needed to perform these calculations. You may type in a number here, or click mouse button 1 on the cell to display a menu containing several recommended values.

Gaussian pupil apodization specification (sdaz)

- **Unapodized:** the pupil is assumed to uniformly illuminated.
- **Gaussian 1/e² spot sizes:** the spot diagram is computed with a Gaussian apodized pupil (i.e., a Gaussian input beam such as is emitted by a laser operating in the TEM00 mode).

1/e² entrance Gaussian irradiance

spot size in x(sdgx) and spot size in y(sdgy):

These are the beam radii in x and y, respectively, at which the irradiance drops to 1/e² of its axial value, measured at surface 1. For example, if the value of

the spot size is equal to the entrance beam radius, the beam irradiance has decreased by $1/e^2$ at the edge of the pupil.

- **Gaussian $1/e^2$ numerical apertures:** the spot diagram is also computed with a Gaussian apodization. The $1/e^2$ divergence data is specified in NA from the point source.

Wavelength

The wavelengths used by OSLO for ray tracing are specified in this spreadsheet. You must always have at least one wavelength defined. The convention used in OSLO is that if three or more wavelengths are defined, wavelength 1 should be the middle wavelength, wavelength 2 should be the short wavelength, and wavelength 3 should be the long wavelength. This is not a requirement, but following this convention will result in plausible values of chromatic aberrations, etc.

The Wavelength numbers are displayed on row buttons. The wavelength number is used to refer to the wavelength in operands and in most commands that require a wavelength to be specified.

The second column contains the Wavelength itself. When a new lens is opened, the values of wavelengths 1, 2, and 3 are set to the current values of the Wavelength_default (**wvld**) preference. The initial values of this preference are 0.58756 μm , 0.48613 μm , and 0.65627 μm . Note that wavelengths are always entered in micrometers. To change the value of a wavelength, there are two possible methods. First, you can simply select the appropriate wavelength cell and type in the desired value for the wavelength. Second, if you activate the wavelength cell by clicking on it, a pop-up list containing many common spectroscopic and laser lines will be displayed. You can change the wavelength by selecting the desired value from the list.

Each wavelength has an associated Weight. When a new lens is opened, the default wavelength weights for wavelengths 1, 2, and 3 are set to the current values of the Weight_default (**wgtd**) preference. The weight for wavelength 1 must always be non-zero. The wavelength weights can be used, for example, to model light sources that do not have a uniform energy distribution across their emission spectrum. The main uses of wavelength weights in OSLO are in spot diagram analysis and in polychromatic optimization error functions.

At any given time, one of the defined wavelengths is designated as the Current wavelength. Usually, the current wavelength is wavelength 1, but you can select any wavelength to be the current one. You might want to change the current wavelength, for example, to compare the aberrations of the lens at different wavelengths. The current wavelength is used for performing paraxial ray traces, computing aberration coefficients, tracing single rays and fans of rays, etc. To designate the current wavelength, click the radio button at the right end of the row for the desired wavelength.

Std **Prm** Generate wavelengths

Rather than enter the wavelengths explicitly, you can ask OSLO to generate a set of wavelengths for you. The wavelengths will be chosen according to the sampling points required for a Gaussian quadrature numerical integration over the specified spectral range. To generate a wavelengths set, click the Generate wavelengths button. You will be prompted for the number of wavelengths you wish to generate,

and the lower and upper bounds of the spectral range of interest (in microns). In OSLO Premium, it is possible to specify a CCL weighting function to be used in generating the wavelengths by specifying the Command for color weighting (**opcw**) optimization condition.

Std **Prm** Field Points

The field points set defines the fractional object-surface coordinates of points from which optimization rays emanate.

A complete discussion of the field point editor can be found in Chapter 9, page 350.

See Vignetting Analysis/Setting Vignetting Factors for automatic settings of vignetting factors.

Variables

Opens the variables spreadsheet, which is used to provide a complete specification of optimization variables, including boundary conditions, derivative increments, etc. Basic surface data (radii, thicknesses) can be specified to be variable in the normal surface data spreadsheet, but special data (tilts, decenters, aspheric coefficients, etc.) must be specified using the variables spreadsheet. In addition to providing detailed specification of variables, the spreadsheet contains buttons for convenient input of multiple variables (all curvatures, etc.). The Variables and the variables spreadsheet it covered more completely in Chapter 9, page 366.

Draw On / Draw Off

The Autodraw window will be open or closed depending on whether Autodraw is turned “off” or “on”. If Autodraw is “on”, the lens and default rays will be drawn in the special Autodraw graphics window each time the lens is changed or a new surface is selected.

The Autodraw window is a special type of graphics window. It can display side views of your system - viewing orthogonal to the YZ, XZ, or XY planes. You can change these views in “Lens>>Lens Drawing Conditions”. The Autodraw window does not allow you to perform more sophisticated graphic analyses within the window, but the Autodraw window is automatically updated whenever the Surface Data Spreadsheet is updated.

Group / Surfs

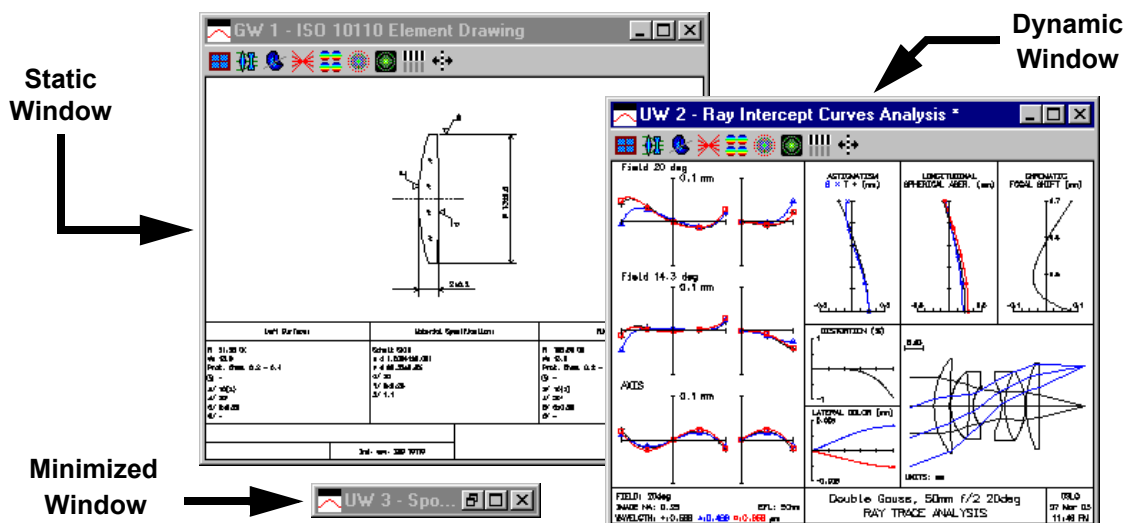
The lens data can be displayed in one of two modes: Group or Surface. The lens data viewing option is selected by clicking the radio button next to the desired choice. The selected view option affects the display of the surface data for lens groups. Examples of lens groups are lens elements selected from one of the catalog databases, user-defined groups, and non-sequential groups. In surface mode, the data are displayed in the same fashion as for surfaces that are not part of a group, i.e., the values for each surface are shown. In group mode, the entire group is represented by the first and last surfaces of the group. The first surface radius cell displays the type of group. For an element group, the thickness cell displays the total axial length of the group.

Notes (System Notes)

Users can add up to 5 lines (80 characters each) of notes that get stored with the lens. The information from the system notes can be displayed in the Text Window using the “Lens>>Special Surface” menu option.

Overview

All graphical analysis output will appear in general-purpose graphics windows. Up to 32 graphics windows can be open at a time.







Depending on how the data is created in the window, each of the graphics windows can be one of two different types. Ordinary graphic windows (denoted by *GW* in the title bar) are *static*, indicating that the contents of the window cannot be changed after it is displayed. Updatable windows (denoted by *UW* in the title bar) are *dynamic*; the contents can be redisplayed using the current data by double clicking anywhere in the content portion of the window.

OSLO graphics are created by a high-speed internal integer-based vector system that can accommodate graphics windows with varying aspect ratios. When you resize a graphics window, the image adapts to both the size and shape of the window.

Note that the AutoDraw window is a special type of graphics window. The AutoDraw window is described in more detail in the section “Draw On / Draw Off” on page 125.

Window Control

The buttons in the upper right corner of the window (on the right side of the title bar) are for minimizing (, maximizing (, and closing () the window. Any of the 32 graphics windows can be open or closed in any order, however the last graphics window cannot be closed. When there is only one graphics window left open, the close button for that window will be disabled (). There must always be at least one graphics window open at all times. The last graphics window may be minimized or “docked” but it will still be counted as “open”.

The current graphics window (the window where graphics will be drawn by default), is noted with an asterisk (*) after the words in the title bar of the window

Pop-up Menu



OSLO graphic windows provide a pop-up menu list that accesses several control features of the windows. To access this menu, simply click the right mouse button anywhere in the content portion of the graphics window.

This pop-up menu can also be accessed from the graphic window toolbar menu. See the section “Right-click actions” on page 131.

Update Window using current data

Recalculates the contents of the window. A window may also be updated by double-clicking in the drawing area of the graphics window. This item will only be available if the graphics window is dynamic (the name in the title bar starts with a *UW*).

Re-calculate using new parameters

Opens a dialog box in which you can change the parameters (scale, etc.) of the window. If the window was originally created using a dialog box, this new dialog box has the same format except that the settings in this dialog box reflect the current status of the contents of the window. This item will only be available if the graphics window is dynamic (the name in the title bar starts with a *UW*).

Zoom In

Increases the current zoom factor by two

Zoom Out

Decreases the current zoom factor by one-half

Zoom Out (Full)

Zooms out to display the full contents of the window

Set Zoom Center

Sets the zoom target point at the current mouse position

Print

Prints the contents of the window

Copy to Clipboard

Copies the contents of the window to the clipboard

Save As

Opens the file dialog box to allow saving the graphics window to a file

Lock (& Unlock)

Locking a window means that the contents of the window can not be changed or cleared (until the window is unlocked). Once a window is locked, the item in this pop-up menu item becomes listed as Unlock so that you reverse the situation. This is useful for ensuring that the window contents that you wish to retain are not inadvertently erased or overwritten

Remove Toolbar (& Restore Toolbar)

“Remove Toolbar” removes the toolbar from the graphics window. This actually gives all the space that the toolbar occupied to the graphics display portion of the window, allowing it to become bigger. Once the toolbar is removed, this menu item becomes listed as “Restore Toolbar” so that you can reverse the situation.

Clear Window

Clears the current graphics window, removes its title (if any), and restores it to a static (GW) window if it was an updatable (UW) window.

Zooming

In addition to the context menu items, there are three additional ways of zooming the contents of a graphics window:

Zoom area selection

A desired zoom area may be selected by using the mouse. Click the left mouse button on one corner of the desired area. While holding the left mouse button down, drag the mouse to the opposite corner of the desired area. A dotted rectangle will indicate the current extent of the selected zoom area. When the left mouse button is released, the contents of the graphics window will be zoomed according to the zoom selection.

Zooming with the mouse wheel

If the active window is a graphics window and your mouse is equipped with a scroll wheel, rotating the wheel will zoom the contents of the window. If the current mouse position is over the vertical scrollbar of the window, wheel rotation will be translated into up and down panning of the window contents. Similarly, if the mouse is over the horizontal scrollbar, rotating the mouse wheel will cause the contents of the window to be panned left and right.

Zooming with the keyboard

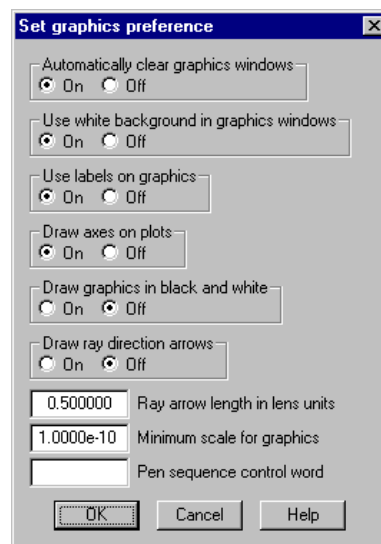
The contents of the current graphics window may be zoomed up or down by pressing CTRL+UPARROW (or CTRL+RIGHTARROW) or CTRL+DNARROW (or CTRL+LEFTARROW). The contents of a zoomed window may be panned by pressing CTRL+SHIFT+UPARROW, CTRL+SHIFT+RIGHTARROW, CTRL+SHIFT+DNARROW, or CTRL+SHIFT+LEFTARROW.

Graphic Preferences

OSLO has over 100 different preferences which can be set that affect the operational parameters of OSLO (see the section “Set Preference / Show Preference” on page 174). Several preferences affect graphics output:


- Automatically clear graphics windows
- Use white background in graphics windows
- Use labels on graphics
- Draw axes on plots
- Draw graphics in black and white
- Draw ray direction arrows
- Set ray arrow length
- Set minimum scale for graphics
- Set pen sequence

Setting all these items is addressed in detail in the section relating to the menu item “File>>Preferences>>Preferences Groups>>Graphics” on page 173.



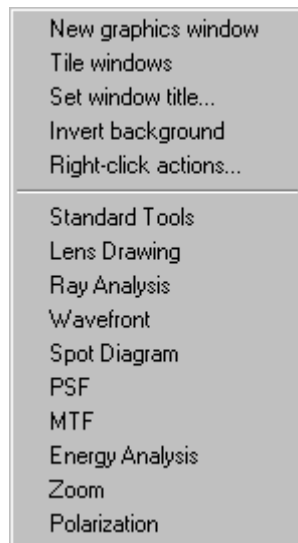
Toolbars

Just below the title bar, the graphics window contains buttons with icons which provide graphics output when selected. These buttons are arranged by associated function into groups. Each group is referred to as an individual toolbar. Because the real estate of a graphics window is very valuable, graphics windows can have only 1 toolbar active at a time, unlike a text window. However, different

graphic windows can have different toolbars. The toolbars that can be used in a graphics window are shown in a menu that drops down from the **Setup Window/Toolbar** button (). This button always appears to the left of any buttons that are on the active toolbar.



The menu that drops down from this button is slightly different, depending on the OSLO edition and whether it is on the main toolbar, a text window toolbar, or a graphics window toolbar. For a graphics window in OSLO Premium, the menu appears as follows:



New graphics window

This menu item creates a graphics window having the next available number and the default size and position (which is the previous size and position of the window when it was last closed).

Tile windows

This menu item automatically arranges OSLO windows in a pattern that you might want. The default layout puts the spreadsheet (if one is open) at the upper left corner of the main OSLO window, the text windows below it, and the graphics windows in the remaining space on the right. The windows are resized to fill as much of the OSLO window as possible, consistent with not allowing the graphics windows to become grossly distorted in aspect ratio.

This item is covered in more detail in association with the menu item “Window>>Tile Windows”. See the section “Tile Windows” on page 484.

Set window title

This item opens a dialog windows that allows you to add a title to the current graphics window.

Invert background

Reverses the background of the current graphics window from white to black or visa versa.

Right-click actions

This item opens the window control pop-up menu that was described previously (see page 128). This option allows you to access the graphics window pop-up menu even if you don't have a right mouse button.

Standard Tools



Draw system (2D plan view)



Draw system (3D view)



Plot report graphics ray analysis



Plot report graphics wavefront analysis



Plot report graphics spot diagram analysis



Plot report graphics PSF analysis



Plot report graphics MTF through-frequency



Plot report graphics through-focus MTF

Lens Drawing



Draw system (2D)



Draw system (3D - Wire model)



Draw system (3D - Solid model)



Draw system (3D - Shaded model)



Draw system (3D w/slider)



View Ray Fans (2D) - Interactive



View Ray Fans (3D) - Interactive

Std **Prm**



Draw system in multiconfiguration layout



Draw an element according to ISO10110



Edit Lens Drawing Conditions

Ray Analysis



RIC Report Graphic



RIC Plot

Std **Prm**



RIC vs. Field Points



OPD Report Graphic



OPD Plot

Std **Prm**



OPD vs. all field points



Chromatic Shift Plot



Lateral Chromatic Shift Plot



Distortion Plot

Wavefront



Plot Wavefront Report Graphic

Std **Prm**



Wavefront vs. All field points



Plot Wavefront Map



Plot Wavefront Contour



Plot Interferogram

Spot Diagram



Plot report graphics spot diagram analysis

Std **Prm**



Spot Diagram vs. All field points



Plot Spot Diagram



Plot Recipolar Spot Diagram



Plot RMS Spot Size/OPD vs. Field



Plot Beam Footprint

Std **Prm**



Plot Beam Footprint Outline



Ghosts analysis

Std **Prm**



Plot Graphic Ray Set and Pupil Sampling

PSF



Display report graphics PSF analysis

Std **Prm**



PSF vs. All field points



Plot FFT-based PSF contour



Plot FFT-based PSF map



Plot PSF contour (Direct Integration)



Plot PSF map (Direct Integration)



Plot PSF scans

MTF



Display report graphics MTF analysis



Plot MTF

Prm



Plot FFT-based MTF



Plot MTF and PTF



Plot MTF vs. Field



Plot square wave response



MTF and Spot Diagram v.s. Defocus (slider)



Plot report graphics through-focus MTF



Plot Through-focus MTF



Plot Through-focus MTF/Field Curvature

Energy Analysis



Diffraction Encircled Energy



Diffraction Encircled Energy



Diffraction Line Spread Function



Plot Encircled Energy (geometrical)



Plot Ensquared Energy (geometrical)



Geometrical Line Spread Function

Prm Zoom

Std



Draw system in multiconfiguration layout



Zoom report graphic

Prm Polarization



Plot polarization state across pupil



Thin Film Uniformity



Plot Intensity Reflectance



Plot Intensity Transmittance



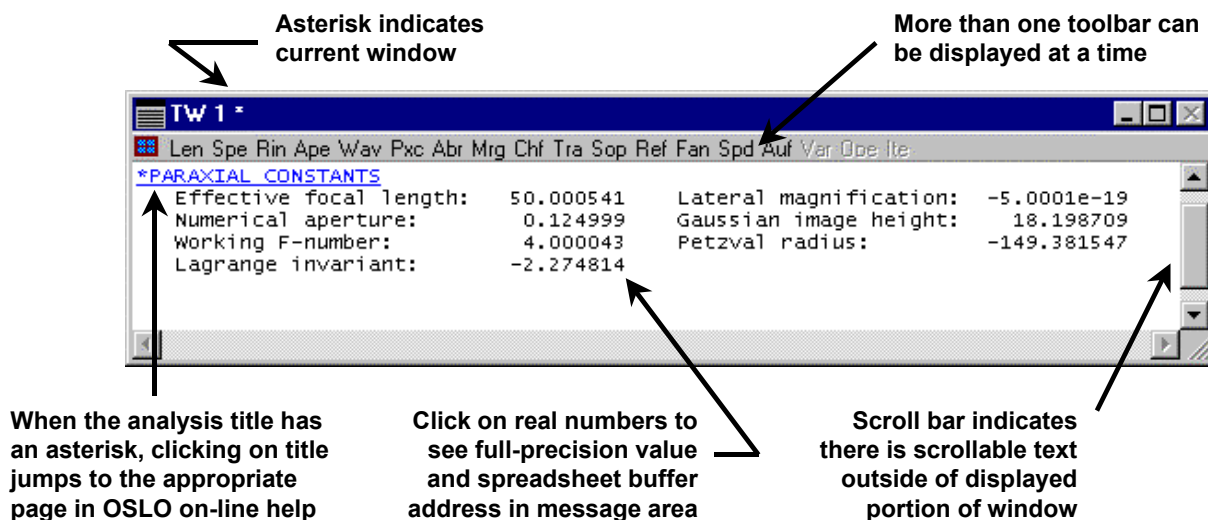
Plot Phase Reflectance







Plot Phase Transmittance

Overview

All alphanumeric program output will appear in the text windows. Up to 2 text windows can be opened at a time. A typical text window is shown below.



Window Control

The buttons in the upper right corner of the window (on the right side of the title bar) are toggles for minimizing (), maximizing (), and closing () the window. The 2 text windows can be open or closed in any order, however the last open text window cannot be closed. When there is only one text window left open, the close button for that window will be disabled (). There must always be at least one text window open at all times. The last text window may be minimized or “docked” but it will still be counted as “open”.

OSLO uses the concept of a current window to determine where to send text output; the current window is the one that has an asterisk in the title bar. The window does not have to be on top of other windows to receive output.

Window Operation

For analysis results that are displayed in the text window, if the analysis output title starts with an asterisk, this is an indication that the title itself is a hotlink to the appropriate page or section of the on-line help where the analysis is further described. This is especially useful for output analyses where header text has been severely abbreviated in order to save space.

In the text window shown above, if you click on the analysis output title ***PARAXIAL CONSTANTS**,

the on-line help window becomes active and automatically displays the appropriate help page shown below.

[OSLO Help Overview](#) - [Command Reference](#)

pxc command

Displays the paraxial constants for the system. The values may be computed in either the y-z (default) or x-z meridian. If the x-z meridian is chosen, the aperture and field may be scaled from the values defined by the entrance beam radius/object numerical aperture and field angle/object height. For focal systems, the quantities displayed are the effective focal length, working f-number, numerical aperture, paraxial (Lagrange) invariant, transverse magnification, Gaussian image height, and Petzval radius. For afocal systems, the quantities displayed are the angular magnification, eye relief, paraxial invariant, and Petzval radius.

Click [here](#) for additional help.

Command Syntax:

```
cmd paraxial_constants
(Paraxial_constants_meridian,
 Fractional_xz_aperture,
 Fractional_xz_field,
 Evaluation_wavelength)
```

Argument Definitions:

Arg #	Type	Name / Definition
0	int	Paraxial_constants_meridian { list = Parax_cons_merid, default (noquery) = "y"}
1	real	Fractional_xz_aperture { lolim = 0.0, default (noquery) = 1.0}
2	real	Fractional_xz_field { lolim = 0.0, default (noquery) = 1.0}
3	int	Evaluation_wavelength { lolim = 0, hilim = numw, default (noquery) = curwav}

List Definitions:

```
list Parax_cons_merid
  "{y}YZ plane" = [3]0,
  "{x}XZ plane" = [1,2,3]1
```

All real numbers (those printed with a decimal point symbol) sent to a text window are backed up in an internal **Spreadsheet Buffer** that can be accessed from the keyboard or by macro programs. You can click on a real number in a text window to see its full value in the message area. For more information on using the spreadsheet buffer see the section "Spreadsheet buffer elements" on page 533.

OSLO Text windows contain 1999 lines. The windows always have scroll bars, but if there is no text to be scrolled, the scrollbars are disabled. The way in which program output is displayed in a text window depends on the window **Page Mode** setting which can be set from the window right-click pop-up menu.

The contents of a text window can be printed, copied to the Windows clipboard, or saved in a text file, using commands from the window right-click pop-up menu.

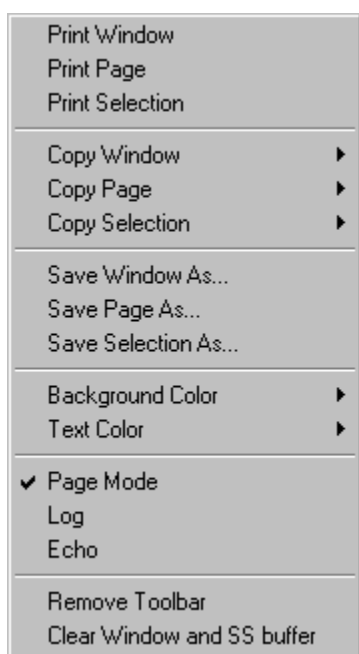
OSLO has a preference called **Output_text (outp)** that controls whether text output is actually displayed in the current window. If **outp** is on, the window displays text in its normal way. However, if **outp** is off, real numbers (in text output that would normally be sent to the window) are still placed in the spreadsheet buffer, but the output is not displayed in the text window itself. This provides a

convenient way for SCP and CCL programs to interact with OSLO. By not displaying the lines, text output is speeded up by a large factor, improving the performance of macro programs, and eliminating visual clutter. The setting of **outp** does not affect **print** or **printf** output sent to the window *by* an SCP or CCL program (**aprint** and **aprintf** output *is* affected).

n.b. the default CCL/SCP error handler turns output on.

Although **outp** is a very useful feature and is widely used, it can have one negative effect. If a macro program that has turned **outp** off fails at some point in its execution, it may not restore **outp** to on before it terminates, leaving OSLO in a state where it seems to be unable to produce any text output. If this happens, you should execute the command **stp outp on** to see if this solves the problem.

Pop-up Menu



OSLO text windows provide a pop-up menu list that accesses several control features of the windows. To access this menu, simply click the right mouse button anywhere in the content portion of the text window.

This pop-up menu can also be accessed from the text window toolbar menu described on page “Right-click actions” on page 142.

...Window

In these commands, “Window” refers to all the text that has been placed into the text window since the text window was last cleared. This includes text that is currently not in view due to the position of the scroll bar.

...Page

In these commands, “Page” refers to all the text that is currently in view in the window.

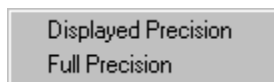
...Selection

In these commands, “Selection” refers to the currently selected text.

Print Window / Page / Selection

A copy of the appropriate text (see “window/page/selection” description above) is sent to the default printer. The user will not be asked to clarify the settings of the default printer before printing occurs.

Copy Window / Page / Selection



A copy of the appropriate text (see “window/page/selection” description above) is sent to the Windows clipboard, from which it can be pasted into another application that supports the clipboard (including the OSLO editor).

Displayed Precision

This option indicates that all real numbers will be sent to the Windows clipboard with the same precision that is shown in the text window.

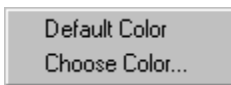
Full Precision

This option indicates that all real numbers will be sent to the Windows clipboard using the full numeric precision that OSLO uses for internal calculations.

Save Window As / Page As / Selection As

A copy of the appropriate text (see “window/page/selection” description above) is sent to a text file. It brings up a standard Windows dialog box which allows you to specify the name of the file and the file location. The default file location is the private directory. The only type of file type allowed is *.txt, which is an ASCII file containing the same spacing used for display in the text window. If you import this file into a word processor, you should be sure to use a fixed-spaced font (such as Courier New) to preserve column alignments. “Save Window As” is identical to the “File>>Save Text As...” from the OSLO menu.

Background Color / Text Color



These options allow you to select the background-color and text-color for the selected text. If no text is selected in the text window, these menu options are inactive.

Default Color

This returns the chosen item to the default color. The default text-color is black. The default background-color is white.

Choose Color

This option allows the user to choose a different color for the text and background of the selected text in the text window.

Page Mode

If “Page Mode” is on (checked), then the first line of output from a command appears at the top of the window; if there are more lines of output than the size of the window, the additional lines are below the bottom, so the lines must be scrolled up to see them. If “Page Mode” is off, the last line of output appears at the bottom of the window, and the beginning lines are above the top, so the lines must be scrolled down to see them. “Page Mode” can also be set using the **Output_page_mode (page)** OSLO preference item.

Log

If “Log” is on (checked), then the text that is printed in either text window is also saved in a log file. The text is saved in the file “oslog.txt” in the private directory. Since this file is rewritten each time OSLO is started, you should copy this file to another location after you exit OSLO if you want to retain its contents. “Log” can also be set using the **Output_log (olog)** OSLO preference item.

Echo

If “Echo” is on (checked), then both text output and commands will be saved in the log file. “Echo” can also be set using the **Output_echo (echo)** OSLO preference item.


Remove Toolbar (& Restore Toolbar)

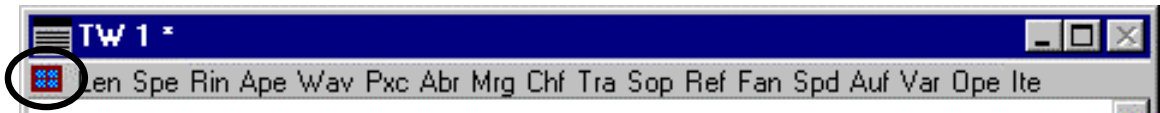
“Remove Toolbar” removes all the toolbars from the text window. This actually gives the space that the toolbar occupied to the text output display portion of the window, allowing it to become bigger. Once the toolbar is removed, this menu item becomes listed as “Restore Toolbar” so that you can reverse the situation.

Clear Window and Spreadsheet Buffer

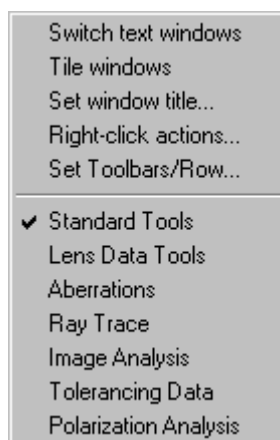
Clears the current text window and resets the Spreadsheet Buffer so that all the contents are zero.

Toolbars

Just below the title bar, the text window contains buttons with icons which provide text output when selected. These buttons are arranged by associated function into groups. Each group is referred to as an individual toolbar. Unlike a graphics window, each button appears as a text sequence of 3 or 4 letters, and a text window can have up to 7 toolbars active (showing) at a time. One or more toolbars can be showing on a single line and you will see how to change this setting below. The two text windows can each have different toolbars active. The toolbars that can be used in a text window are shown in a menu that drops down from the **Setup Window/Toolbar** button (). This button always appears to the left of any buttons that are on the active toolbar.



The menu that drops down from this button is slightly different, depending on the OSLO edition and whether it is on the main toolbar, a text window toolbar, or a graphics window toolbar. For a text window in OSLO Premium, the menu appears as follows:



Switch text windows

This menu item makes the alternate text window active (there are a maximum of two text windows). If the alternate text window is not opened, it is opened before being made active. Since text output goes to the text window that is currently active, this option effectively controls where the next text output will be displayed. Controlling the active text window can also be accomplished using the **textwin_open(two)** command.

Tile windows

This menu item automatically arranges OSLO windows in a pattern that you might want. The default layout puts the spreadsheet (if one is open) at the upper left corner of the main OSLO window, the text windows below it, and the graphics windows in the remaining space on the right. The windows are resized to fill as much of the OSLO window as possible, consistent with not allowing the graphics windows to become grossly distorted in aspect ratio.

This item is covered in more detail in association with the menu item “Window>>Tile Windows”. See the section “Tile Windows” on page 484.

Set window title

This item opens a dialog windows that allows you to change the title of the current text window.

Right-click actions

This item opens the window control pop-up menu that was described previously (see page 139). This option allows you to access the graphics window pop-up menu even if you don't have a right mouse button.

Set Toolbars/Row

This item allows you to specify how many toolbars are shown on each toolbar row in the active text window. The minimum is 1; the default is 2; the maximum is 7. It is recommended that not more than 2 toolbars be placed on a single row as the toolbars to not wrap, and you will have to expand the text window wide enough to see a number of toolbars on a single row. Often it is desirable to set this value to 1, then each toolbar occupies a separate row and you can easily keep them straight.

Standard Tools

- Len** **Lens data:** Displays the regular surface data (radius, thickness, glass, ...etc.) in the text window. This is equivalent to choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type”=“Surface data” in the resulting dialog.
- Spe** **Special data:** Displays the special surface data in the text window. This is equivalent to choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type”= “Special data” and “Special data type”=“All special data” in the resulting dialog.
- Rin** **Refractive index data:** Displays the refractive index data for each surface in the text window. This is equivalent to choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type”=“Refractive indices” in the resulting dialog.
- Ape** **Aperture data:** Displays the regular and special aperture data for each surface in the text window. This is equivalent to choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type”=“Aperture data” in the resulting dialog.
- Wav** **Wavelength data:** Displays the system wavelength data in the text window. This information is included in the analysis that is accessed by choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type”=“All data” in the resulting dialog.
- Pxc** **Paraxial constants:** Displays a summary of the system paraxial constants in the text window. This is equivalent to choosing *Evaluate>>Paraxial Ray Analysis...* from the OSLO menu and then choosing “Command”=“Paraxial constants” in the resulting dialog.
- Abr** **Aberrations:** Displays a summary of the paraxial ray trace and aberration coefficients in the text window. This is equivalent to performing 4 different analysis options together:
- 1) Choosing *Evaluate>>Paraxial Ray Analysis...* from the OSLO menu and then choosing “Command”=“Paraxial raytrace” and “Surface Range:First surface=0” & “Last Surface=0” in the resulting dialog.
 - 2) Choosing *Evaluate>>Aberration Coefficients...* from the OSLO menu and then choosing “Aberration type”=“Paraxial chromatic” and “Surface Range:First surface=0” & “Last Surface=0” in the resulting dialog.
 - 3) Choosing *Evaluate>>Aberration Coefficients...* from the OSLO menu and then choosing “Aberration type”=“Seidel image” and “Surface Range:First surface=0” & “Last Surface=0” in the resulting dialog.
 - 4) Choosing *Evaluate>>Aberration Coefficients...* from the OSLO menu and then choosing “Aberration type”=“Fifth-order image” and “Surface Range:First surface=0” & “Last Surface=0” in the resulting dialog.

- Mrg** **Marginal (axial) ray:** Displays a surface-by-surface summary of a marginal (axial) ray trace in the text window. This is equivalent to tracing a reference ray at the center of the field (*"SOP 0 0"* on the command line) and then choosing *Evaluate>>Single Ray Trace...* from the OSLO menu and then choosing to show all surfaces (*"Surface Selection Option" = "All"*) and choosing to trace a ray through the maximum Y dimension of the system aperture (*FY=1.0*).
- Chf** **Chief (principal) ray:** Displays a full-field chief (principal) ray trace surface-by-surface summary in the text window. This is equivalent to tracing a reference ray at the maximum FBY field (*"SOP 1 0"* on the command line) and then choosing *Evaluate>>Single Ray Trace...* from the OSLO menu and then choosing parameters that trace a ray through the center of the system aperture (*FY=0*).
- Tra** **Trace a specified single ray:** Displays a single ray trace surface-by-surface summary in the text window. The ray parameters are chosen by choosing options in a dialog box. If the *"Set Object Point"* is not chosen, the ray will be traced relative to the current reference ray. This toolbar icon is equivalent to choosing *Evaluate>>Single Ray Trace...* from the OSLO menu.
- Sop** **Set reference ray data (object point):** Opens the Set Object Point dialog box (**sops**). This dialog box allows the user to choose values associated with the a reference ray. Once the values in the dialog box are accepted, OSLO traces the new reference ray using the Set Object Point (**sop**) command. Many analysis options require that the reference ray be set to different values in order for the analysis to be performed at different field points.
- Ref** **Show current reference ray data:** Displays the current reference ray data in the text window. This is equivalent to entering *"SOP"* (no quotes) on the command line.
- Fan** **Trace a specified ray fan:** Displays a summary in the text window, of where a fan of rays hit on the image surface for a single field point. This is equivalent to choosing *Evaluate>>Ray Fans>>Single Field Point...* from the OSLO menu and then choosing the appropriate *"Command" = "Print Y ray-fan"* or *"Command" = "Print X ray-fan"* from the resulting dialog.
- Spd** **Trace a spot diagram:** Displays a summary of spot diagram and wavefront data in the text window. This is equivalent to performing 3 different analysis options together:
- 1) Choosing *Evaluate>>Spot Diagram>>Single Spot Diagram* from the OSLO menu and then choosing *"Spot Diagram Data"* from the resulting dialog.
 - 2) Choosing *Evaluate>>Spot Diagram>>Spot Size Analysis* from the OSLO menu and then making no changes to the default values in the resulting dialog.
 - 3) Choosing *Evaluate>>Wavefront>>Wavefront Analysis* from the OSLO menu and then choosing *"Center of wavefront reference sphere coordinates" = "Focal shifted image surface"* from the resulting dialog.
- Auf** **Autofocus:** Calculates the appropriate focus position of the image plane and adjusts the value of the image thickness appropriately. This is equivalent to choosing *Evaluate>>Autofocus* from the OSLO menu.

Var

Variables: Displays a summary of the active system variables in the text window. This is equivalent to choosing *Lens>>Show Optimization Data>>Variables* from the OSLO menu.

Ope

Operands: Displays a summary of the current optimization error function in the text window. This is equivalent to choosing *Lens>>Show Optimization Data>>Error Function Operands* from the OSLO menu.

lte

Iterate: Performs ten full iteration cycles of optimization using the current variables and error function. The results of each iteration are displayed in the text window. This is equivalent to choosing *Optimize>>Iterate...* from the OSLO menu and then accepting the default values in the resulting dialog.

Lens Data Tools

- Std** **Prm** **Cfg** **Zoom data:** Displays a summary in the text window of the system multiconfiguration (zoom) data. This analysis is equivalent to the first part of the analysis performed when choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type” = “Configuration data” in the resulting dialog.
- Slv** **Solves:** Displays a summary in the text window of all the solves in the system. This analysis is equivalent to choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type” = “Solve data” in the resulting dialog.
- Pkp** **Pickups:** Displays a summary in the text window of all the surface pickups in the system. This analysis is equivalent choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type” = “Pickup data” in the resulting dialog.
- Ape** **Aperture data:** Displays a summary in the text window of all the special aperture data in the system. If there are no special apertures, no data is presented. If there are special apertures in the system, this toolbar item is equivalent to choosing *Lens>>Show Surface Data...* from the OSLO menu and then choosing “Data type” = “Aperture data” in the resulting dialog.

Aberrations

- Pxs** **Paraxial setup:** Displays a summary in the text window of the system paraxial setup parameters. This is equivalent to choosing *Evaluate>>Paraxial Setup...* from the OSLO menu.
- Pxt** **Paraxial trace:** Displays the paraxial chief and axial ray trace for each surface in the text window. This is equivalent to choosing *Evaluate>>Paraxial Ray Analysis...* from the OSLO menu and then choosing “Command” = “Paraxial raytrace” in the resulting dialog.
- Chr** **1st order chromatic aberrations:** Displays the paraxial chromatic aberration contributions for each surface in the text window. This is equivalent to choosing *Evaluate>>Aberration Coefficients...* from the OSLO menu and then choosing “Aberration type” = “Paraxial chromatic” from the resulting dialog.
- Sei** **3rd order Seidel aberrations:** Displays the third order Seidel aberration contributions for each surface in the text window. This is equivalent to choosing *Evaluate>>Aberration Coefficients...* from the OSLO menu and then choosing “Aberration type” = “Seidel image” from the resulting dialog.
- Fif** **5th order Seidel aberrations:** Displays the fifth order aberration contributions for each surface in the text window. This is equivalent to choosing *Evaluate>>Aberration Coefficients...* from the OSLO menu and then choosing “Aberration type” = “Fifth-order image” from the resulting dialog.

- Std** **Prm** **Zpxs** **Zoom paraxial data:** Displays a summary in the text window of the first order analysis for multiconfiguration (zoom) systems. This analysis is equivalent to the second part of the analysis performed when choosing *Evaluate>>Zoom Lens Parameters* from the OSLO menu.
- Std** **Prm** **Zspa** **Zoom spacing analysis:** Displays a summary in the text window of the spacing between moving groups in multiconfiguration (zoom) systems. This analysis is equivalent to the first part of the analysis performed when choosing *Evaluate>>Zoom Lens Parameters* from the OSLO menu.
- Std** **Prm** **Zsen** **Zoom systems sensitivity:** Displays a summary in the text window of the sensitivities for different moving groups in multiconfiguration (zoom) systems. This analysis is equivalent to choosing *Evaluate>>Zoom Aberrations>>Zoom Lens Sensitivity...* from the OSLO menu and then choosing the default values in the resulting dialog.
- Std** **Prm** **Zchr** **Zoom chromatic aberrations:** Displays a summary in the text window of the chromatic aberration coefficients for different moving groups in multiconfiguration (zoom) systems. This analysis is equivalent to choosing *Evaluate>>Zoom Aberrations>>Chromatic...* from the OSLO menu and then choosing the default values in the resulting dialog.
- Std** **Prm** **Zsei** **Zoom 3rd order Seidel coeffs:** Displays a summary in the text window of the third-order Seidel aberration coefficients for different moving groups in multiconfiguration (zoom) systems. This analysis is equivalent to choosing *Evaluate>>Zoom Aberrations>>Third-order...* from the OSLO menu and then choosing the default values in the resulting dialog.
- Std** **Prm** **Zfif** **Zoom 5th order Seidel coeffs:** Displays a summary in the text window of the fifth-order Seidel aberration coefficients for different moving groups in multiconfiguration (zoom) systems. This analysis is equivalent to choosing *Evaluate>>Zoom Aberrations>>Fifth-order...* from the OSLO menu and then choosing the default values in the resulting dialog.

Ray trace

Ref **Print current reference ray data:** Displays the current reference ray data in the text window. This option does not change the value of the current reference ray. This is equivalent to entering “SOP” or “TRR” (no quotes) on the command line.

bot **Trace the lower rim ray:** Displays a surface-by-surface summary of the lower rim-ray in the text window. A lower rim-ray is analogous to tracing a lower marginal ray relative to the current reference ray. This option does not change the value of the current reference ray.

This option is equivalent to choosing *Evaluate>>Single Ray Trace...* from the OSLO menu and then choosing to show all surfaces (“Surface Selection Option” = “All”) and choosing to trace a ray through the minimum Y dimension of the system aperture ($FY = -1.0$).

chf **Trace the chief ray:** Displays a surface-by-surface summary of the current reference ray in the text window. This option does not change the value of the current reference ray.

This option is equivalent to choosing *Evaluate>>Single Ray Trace...* from the OSLO menu and then choosing to show all surfaces (“Surface Selection Option” = “All”) and choosing to trace a ray through the center of the system aperture ($FY=0$).

top **Trace the upper rim ray:** Displays a surface-by-surface summary of the upper rim-ray in the text window. An upper rim-ray is analogous to tracing an upper marginal ray relative to the current reference ray. This option does not change the value of the current reference ray.

This option is equivalent to choosing *Evaluate>>Single Ray Trace...* from the OSLO menu and then choosing to show all surfaces (“Surface Selection Option” = “All”) and choosing to trace a ray through the maximum Y dimension of the system aperture ($FY = 1.0$).

skw **Trace the skew rim ray:** Displays a surface-by-surface summary of a “skew” ray in the text window. In this case, the skew ray passes through the +X edge of the system aperture and is relative to the current reference ray. This option does not change the value of the current reference ray.

This option is equivalent to choosing *Evaluate>>Single Ray Trace...* from the OSLO menu and then choosing to show all surfaces (“Surface Selection Option” = “All”) and choosing to trace a ray through the maximum X dimension of the system aperture ($FX = 1.0$).

Image Analysis

- Sp_s** **Spot size analysis:** Displays a spot size analysis in the text window. This icon opens a dialog box where analysis options are chosen. This icon is equivalent to choosing *Evaluate>>Spot Diagram>>Spot Size Analysis* from the OSLO menu
- W_{wf}** **Wavefront analysis:** Displays a wavefront analysis in the text window. This icon opens a dialog box where analysis options are chosen. This icon is equivalent to choosing *Evaluate>>Wavefront>>Wavefront Analysis* from the OSLO menu
- M_{tf}** **MTF analysis:** Displays a Modulation Transfer Function (MTF) analysis in either a graphic window or a text window. This icon opens a dialog box where analysis options are chosen. This icon is equivalent to choosing *Evaluate>>Transfer Function>>Print/Plot OTF* from the OSLO menu.
- P_{sf}** **PSF analysis:** Displays various Point Spread Function (PSF) analyses in either the text window or graphics window. This icon opens a dialog box where analysis options are chosen. The “Command” option in the dialog box actually allows the user to pick from 5 different PSF analyses:
- 1) Points Spread Value: Reports a specific PSF value for individual x, y & focus points on the image plane in the text window. This option is similar to choosing *Evaluate>>Spread Function>>Print PSF Value* from the OSLO menu.
 - 2) Plot Point Spread: Plots a Fast Fourier Transform (FFT) or Direct Integration PSF in the current graphic window. This option is similar to choosing *Evaluate>>Spread Function>>Plot PSF Map/Contour* from the OSLO menu.
 - 3) Plot X and Y Scans: Plots an X and Y cross-section of the PSF in the current graphics window. This option is similar to choosing *Evaluate>>Spread Function>>Plot PSF Scans* from the OSLO menu.
 - 4) Print Irradiance Grid: Displays the complete irradiance grid of the PSF in the text window. This option is similar to choosing *Evaluate>>Spread Function>>Print PSF Grid* from the OSLO menu. Choose “Output Type” = “Irradiance” in the resulting dialog.
 - 5) Print Amplitude/Phase Grid: Displays the complete amplitude and phase grid of the PSF to the text window. This option is similar to choosing *Evaluate>>Spread Function>>Print PSF Grid* from the OSLO menu. Choose “Output Type” = “Amplitude Phase” in the resulting dialog.

Tolerancing Data

- S_{rf}** **Print surface tolerances:** Displays a summary in the text window of the current surface tolerances. The current surface tolerances can be accessed for editing from the OSLO menu item *Tolerance>>Update Tolerance Data>>Surface*.

Cmp **Print component tolerances:** Displays a summary in the text window of the current component tolerances. The current component tolerances can be accessed for editing from the OSLO menu item *Tolerance>>Update Tolerance Data>>Component*.

Prm **Grp** **Print group tolerances:** Displays a summary in the text window of the current user-defined group tolerances. The current user-defined group tolerances can be accessed for editing from the OSLO menu item *Tolerance>>Update Tolerance Data>>Group*.

Prm Polarization Analysis

Mat **Thin film materials:** Displays multilayer coating material data in the text window. This icon opens a dialog box where analysis options are chosen. This icon is equivalent to choosing *Lens>>Coatings>>Show Materials* from the OSLO menu.

Lay **Defined coatings:** Displays multilayer coating design data in the text window. This icon opens a dialog box where analysis options are chosen. This icon is equivalent to choosing *Lens>>Coatings>>Show Designs* from the OSLO menu.

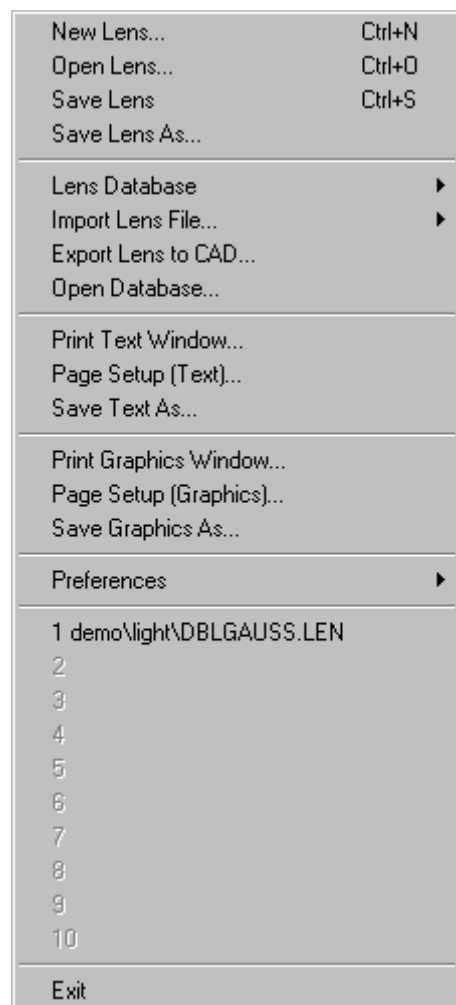
Stk **Compute Stokes Parameters:** Displays a surface-by-surface summary of Stokes parameters in the text window. This icon opens a dialog box where analysis options are chosen. This icon is equivalent to choosing *Evaluate>>Polarization>>Stoke Parameters* from the OSLO menu.

Srt **Surface reflectance and transmittance:** Displays surface reflectance and transmittance data for a given surface in the text window. This icon opens a dialog box where analysis options are chosen. This icon is equivalent to choosing *Evaluate>>Polarization>>Reflectance/Transmittance>>Surface Data* from the OSLO menu.

Xmt **Transmission through system - Options:** Displays a surface-by-surface system transmittance summary in the text window. This icon opens a dialog box where analysis options are chosen. This icon is equivalent to choosing *Evaluate>>Polarization>>Polarization Transmittance* from the OSLO menu.

Gho **Ghosts analysis options:** Displays ghost analysis results in text window. A “movie” showing the different paths can be shown in the current graphic window. This icon opens a dialog box where analysis options are chosen. *NOTE: The spreadsheet window has to be closed for this icon to be active.* This icon is equivalent to choosing *Optimize>>Support Routines>>Ghosts* from the OSLO menu.

Overview



The File menu in the main window deals with lens data, which are ASCII files that contain a series of OSLO commands. The File menu that appears in the text editor window (opened under the Window menu) deals with other file types, such as CCL macros and various other text files. This chapter is only about the main File menu.

File menu commands are used for opening and saving lens data in disk files, printing text and graphics, and exiting the OSLO application. To simplify operation, OSLO uses the standard file dialog boxes prescribed for the windowing system used.

Special commands are included on the File menu to handle output from text and graphics windows to files, as well as to printers. Copying text and graphics output to the Windows clipboard is supported on the Windows menu. Operations that involve inserting a lens or a catalog lens into another lens are included in the Edit menu.

The commands in the File menu are:

Menu Item	Page
New Lens... erases the current lens and enters a new custom, catalog, or perfect lens.	156
Open Lens... replaces the current lens with an existing lens file.	160
Save Lens overwrites the current lens disk file with the current lens data.	161
Save Lens As... prompts for a file name and saves the current lens in a file.	161
Lens Database... opens the private and public lens libraries in an easy to browse database format.	161
Import Lens File... imports a lens file from other popular lens design software formats.	167

Export Lens to CAD... saves a drawing of the current lens on the disk in 3D DXF or IGES format.	168
Open Database... opens files in OSLO's database format. These files include various types of lens data, lists of files, glass catalogs, and user-defined special data.	169
Print Text Window... sends the contents of the current text window to a printer.	170
Page Setup (Text)... configures the printer for text output.	171
Save Text As... sends the contents of the current text window to a file.	171
Print Graphics Window... sends the contents of the current graphics window to a printer.	171
Page Setup (Graphics)... configures the printer for graphics output.	172
Save Graphics As... sends the contents of the current graphics window to a file.	172
Preferences... displays and manages many attributes of the OSLO application.	173
Recent Files is a list of the last 20 most recently opened lens files. Clicking on the name of the file you want will open that file.	180
Exit stops the program.	180

File organization

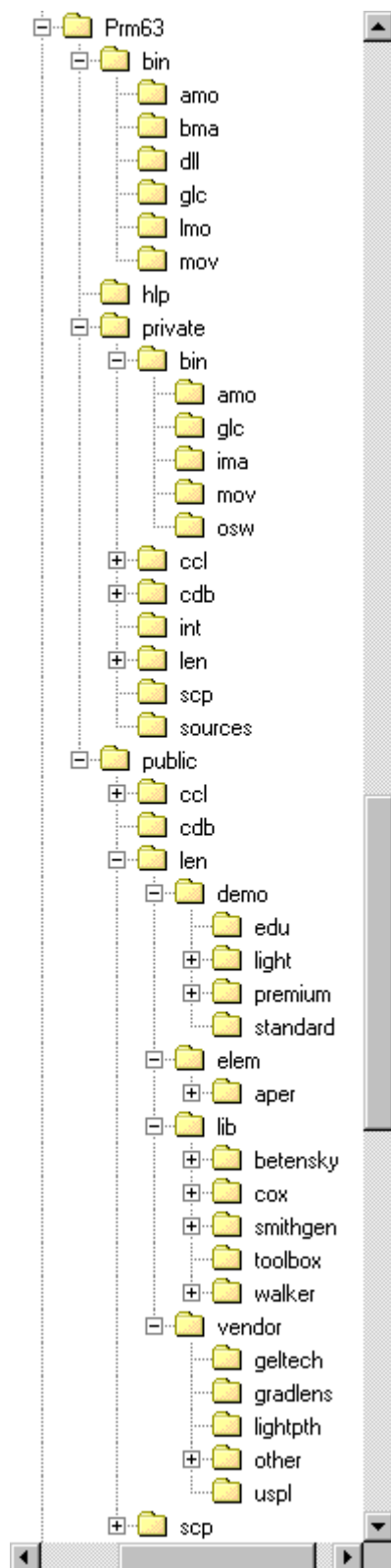
OSLO uses a large number of files of various types, including binary programs (*.exe, *.amo, and *.dll), binary data (*.chm, *.bmp, and *.wmf), and text data (*.len, *.scp, and *.ccl), to name a few. As installed, all OSLO files are located under a single master directory specified during program installation. OSLO does not install any files in the Windows directory or elsewhere in the file system of the host computer. Because of this, all OSLO files can be specified relative to the installation directory, which you establish when you install the program. For OSLO Premium, the default installation directory is Prmxx, where xx depends on the revision number (i.e. 62, 63, ...etc.). For OSLO Standard the default directory is Stdxx, for OSLO Light the default directory is Lightxx, and for OSLO-EDU the default directory is EDUxx.

The figure at the left side of the next page shows the general organization of the file system.

The OSLO file organization consists of a directory structure under the installation directory. There are top level subdirectories: **bin**, which contains program libraries and some support data (e.g. glass and catalog lens databases), **hlp**, which contains the OSLO help system, and two directories called **private** and **public**, which contain both lens and macro data. It is intended that data supplied with the program, as well as data that is to be made available to all users in a multi-user installation, should reside in the **public** directory. Data that you create should normally be saved in your **private** directory. OSLO contains a preference called **Osdata_path (osdp)** that points to the location of your private data. **Osdata_path**

is set to **installation_directory\private** by default, although you can set it to another directory by making an entry in the **win.ini** file (see Appendix).

To simplify moving around the file system, OSLO includes special buttons in the File >> Open and File >> Save As dialog boxes. These buttons are labeled Private and Public, and switch the file listing to your current directories when you select them.



Public data

The public directory contains four subdirectories: **ccl**, **scp**, **cdb**, and **len**. The first two contain source code for macros; the files in these directories are managed with a text editor, not with the commands on the main File menu. The **cdb** directory contains delimited data files for the database structures that OSLO supports. The files in this directory are also managed with a text editor, not with the commands on the main File menu. The **len** directory has several subdirectories that contain hundreds of lens files that are shipped with OSLO, as follows.

len\demo

Contains lens files that serve as samples or examples for a variety of applications of OSLO to practical problems.

edu - contains demonstration lens files that can be opened with the capabilities in OSLO-EDU (similar capabilities of OSLO Light but limited to no more than 10 surfaces).

Lt **Std** **Prm**

light - contains demonstration lens files that can be opened with the capabilities in OSLO Light.

Std **Prm**

standard - contains lenses that require OSLO Standard.

Prm

premium - contains lenses that can only be opened by OSLO Premium. This directory is further divided into

subdirectories containing specialized examples.

Note that the higher editions of OSLO can open the all lens files in the folders of the lower editions.

len\lib

This subdirectory contains complete lens prescriptions that can serve as starting designs for optimization.

- Std** **Prm** **betensky** - contains the zoom designs from the Betensky Zoom Lens Design course.
- Std** **Prm** **cox** - contains the Arthur Cox lens library, consisting of about 300 starting designs derived from lens patents, as published in the book "A System of Optical Design".
- Std** **Prm** **smithgen** - contains about 300 lenses from the book "Modern Lens Design", by Warren Smith and Genesee Optics Software, published by McGraw-Hill.
- Std** **Prm** **toolbox** - contains about 100 miscellaneous lenses chosen as typical prototypes, formerly available as a separate library called the Optics Toolbox.
- Lt** **Std** **Prm** **walker** - contains several examples from Bruce Walker's book, "Optical Engineering Fundamentals" (McGraw-Hill), supplied with OSLO Light, Standard and Premium.

Std **Prm** len\vendor

This subdirectory contains lenses that are commercially available. These lenses are in addition to the lenses from Edmund Industrial Optics, Geltech, JML Optical Industries, Linos Photonics (Spindler Hoyer), Melles Griot, Newport, Oriel (Spectra-Physics) and OptoSigma, which are to be found in the Catalog lens database (see page 157).

- geltech** - contains prescriptions for several aspheric singlets originally available from Geltech, Inc. Geltech has since been purchased by LightPath technologies: www.lightpath.com.
- gradlens** - contains lens files for a series of gradient-index lenses called EndoGRINS™, available from Gradient Lens Corporation: www.endogrins.com.
- lightpth** - contains blanks for lenses made using the LightPath Technologies Gradium™ material. Contact LightPath Technologies: www.lightpath.com.
- other** - contains designs from other vendors and lenses that are not included in the OSLO Catalog database because they do not meet the restricted requirements for database lenses.
- other/edmund** - Edmund Industrial Optics - www.edmundoptics.com
- other/mellesg** - Melles Griot - www.mellesgriot.com
- other/optosig** - now available from Linos Photonics - www.linos.com
- other/spinhoyr** - Spindler & Hoyer: now available from Linos Photonics - www.linos.com
- uspl** - contains lens files from US Precision Lens, now 3M Precision Lens: www.3m.com.

Std Prm len\elem

aper - contains several single-surface systems that illustrate the construction of special apertures. These can serve either as examples or as surfaces that can be merged into other lenses when needed. They are not complete lenses! To see what an aperture looks like, open the file and execute Calculate >> Display Spot Diagram, accepting the defaults.

Private data

The **private** directory is intended to be the repository for data that you create while using OSLO. It contains 7 subdirectories, **bin**, **ccl**, **scp**, **int**, **cdb**, **sources** and **len**.

The **bin** directory contains files with font and license information, as well as subdirectories

amo - which contains object code for your compiled CCL files (AMO: Application Manager Object)

glc - which contains your private glass catalog

ima - which contains pixelated object files (these are in ascii format)

mov - which contains private movies that you construct

osw - contains files that store OSLO slider window settings.

The **ccl** and **scp** directories contain private CCL and SCP source code. As mentioned above, the files in these directories are managed with a text editor, not with the commands on the main File menu.

Prm The **int** directory contains interferogram files that contain data specifying interferometric deformation surfaces.

The **cdb** directory contains delimited data files for the database structures that OSLO supports.

The **len** directory is intended to be used for lens data that you create. Initially, the directory contains a single lens for demonstration. Although you can save your own lenses directly in the **len** directory, it is suggested that you create subdirectories under **len** to organize lens files according to a scheme that meets your needs.

It is very easy to accumulate hundreds (or more) lens files, and even a modest organizational scheme will repay welcome dividends in your operating efficiency in the future.

Some files are saved at the top level of the **private** directory:

- The private directory contains a file called **oslo.ini**, which saves preferences. This is a text file and can be examined with a text editor, but you should not make changes to it, since it is rewritten automatically by OSLO when the program terminates normally.
- If the **Output_log** (**olog**) preference is on, text that is printed in either text output window is also saved in a log file. If the **Output_echo** (**echo**) preference mode is also on, both commands and text output will be saved in this file. This

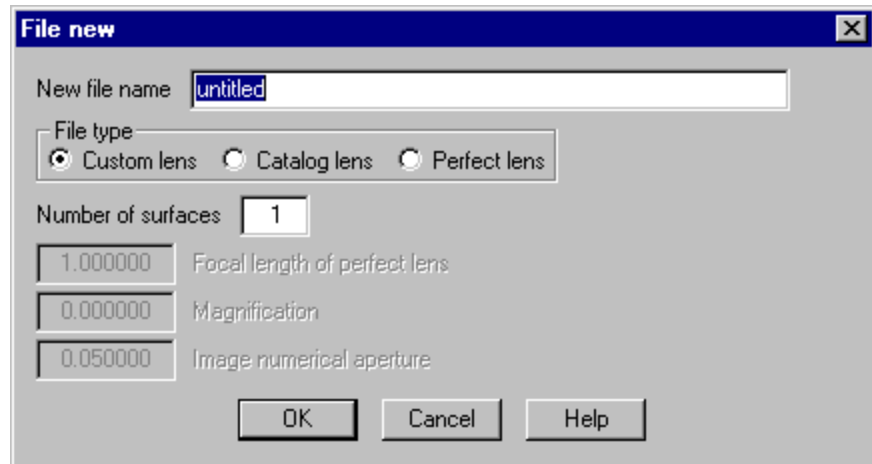
permits you to create a permanent record to trace the development of a design. The text is saved in the file **oslog.txt** in the **private** directory. Since this file is rewritten each time OSLO is started, you should copy this file to another location on disk after you exit OSLO if you wish to retain its contents.

- When you select File >> Save Text As or File >> Save Graphics As, the default file location is the **private** directory, although you can rename and relocate these files, in which case subsequent files will be placed in the same location as you previously used.
- The default location for files written using the Export to CAD command is the **private** directory, although these files can be renamed and relocated, like the text and graphics output files.

New Lens

The New Lens command creates a new lens file that replaces the lens currently in memory. The command provides a dialog box that allows you to enter the file name for saving the new lens to disk, and gives you the choice of three lens types: custom, catalog, or perfect.

Caution - Use the “Save Lens” command or the “Save Lens As” command to save the current lens file to disk if you wish to recover it later, before executing the “New Lens” command.



The file name is the name under which the lens file will be saved if the Save Lens command is issued. OSLO automatically appends the *.len extension to the file name if it is not given.

Default field point set - When a new lens is created, three entries in the field points set will also be created by default. These field points have fractional object height (FBY) values of 0, 0.7, and 1.0, and are equally weighted.

Operating conditions callback - When a new lens is created, the values of the operating conditions are set to default values. It is now possible to specify these values, rather than always using the OSLO defaults. These “personal” defaults are defined by entering the appropriate commands in a CCL command named *opc_init*. When a new lens file is opened, this command will be executed, and any operating conditions commands will thus override the values set by OSLO. Note that the *opc_init* command should consist *only* of operating conditions commands. An easy way to obtain the correct syntax for the operating conditions commands is to look at the “Related Topics” in the Help screen for each operating condition spreadsheet.

As a simple example, consider the lens drawing operating conditions. By default, the lens is drawn with an amount of space determined by “Final distance” reserved for the drawing of rays in image space. (The “Image space rays” item is “Final dist”.) Another option for “Image space rays” is to draw the rays to the image surface. If you always want to use this option, you would enter the *opc_init* command shown below.

```
cmd
opc_init(void)
{
drl_raystosrf(img);
/* other operating conditions commands */
}
```

Custom lens

OSLO prompts for the number of surfaces in the lens, which determines the number of surfaces (in addition to the object and image surfaces) initially available in the Surface Data spreadsheet. Both the file name and the number of surfaces may be changed later. When this option is selected, an empty lens having default values for all data is entered, then the lens spreadsheet is opened automatically.

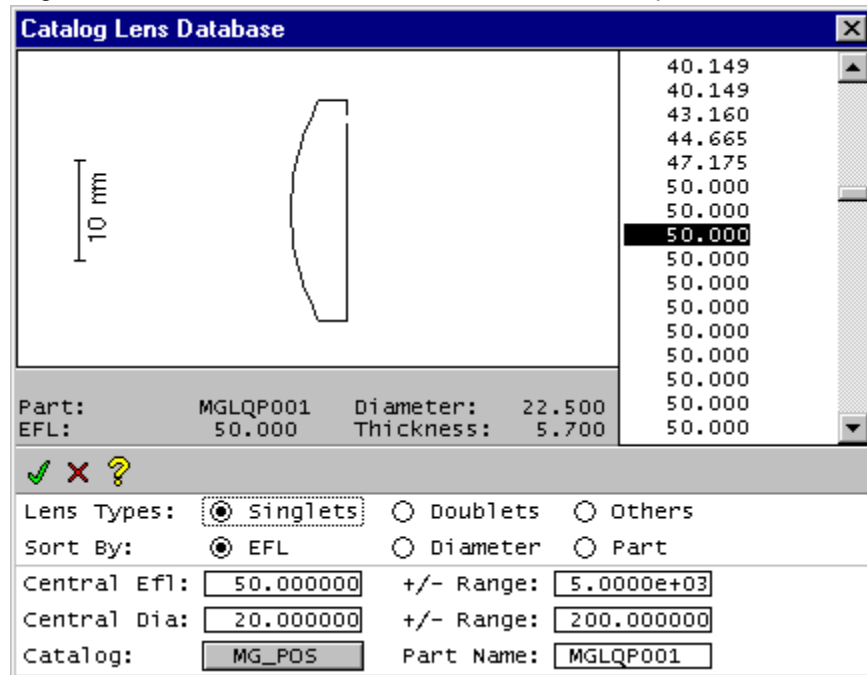
The New Lens command opens the Surface Data spreadsheet so that the surface data can be immediately edited. The number of lines corresponding to surfaces in the spreadsheet is equal to the number of surfaces entered in the New Lens command, plus one line each for the object surface (surface 0) and the image surface.

Catalog lens

Besides allowing the design of custom lens systems, OSLO can use catalog lens elements that are supplied by several vendors. For convenience, the OSLO Surface Data spreadsheet allows you to view either the part numbers or the actual surface data of catalog lenses. Many catalog lens elements have been collected into a Catalog Lens Database. Lens elements of a specified type (singlet, doublet, or other) are displayed in a list that can be sorted according to effective focal length, diameter, or part name. The database makes it easy to select a lens element that is appropriate for a given application.

The Catalog lens option creates a new lens file and, like the Custom lens command, first prompts for a file name. In addition, the dialog box prompts for the magnification; this field is used to set the object and image distances for the lens. After this information is entered and the dialog box is dismissed, the Surface Data spreadsheet is opened and the Catalog Lens Database window appears. A lens from the database can then be selected as described below. After a catalog lens element is selected from the database, the database window is closed and control passes to the Surface Data spreadsheet.

Catalog lenses can also be inserted into other lenses. This is done using the Surface Data spreadsheet, by selecting a row button and then using the Paste Catalog Lens command on the Edit menu. Please see Chapter 3 for details.



The Catalog Lens Database window contains display and control areas. The display area contains:

- a Scrolled List whose entries correspond to lens elements in the database and whose contents are determined by the other controls in the Database window,
- a Drawing Area containing a drawing of the lens corresponding to the currently highlighted item in the Scrolled List,
- a Message Area containing information about the currently highlighted Scrolled List item.
- The control area contains:
 - three radio buttons labeled Lens Types that determine the types of lenses (singlets, doublets, or others) listed in the Scrolled List,
 - three radio buttons labeled Sort By that determine the manner in which the Scrolled List entries are displayed and sorted (effective focal length, lens diameter, or part name),
 - two cells that can be used to specify the allowed range of focal lengths for the entries in the Scrolled List,
 - two cells that can be used to specify the allowed range of diameters for the entries in the Scrolled List,
 - a button that displays a list of the currently available catalogs:
 - Edmund** (Edmund Industrial Optics - www.edmundoptics.com)
 - Geltech** (now LightPath Technologies - www.lightpath.com)
 - JML** (JML Optical Industries - www.jmloptical.com/)
 - LP** (Linos Photonics - www.linos.com/en/index.php)
 - MG** (Melles Griot - www.mellesgriot.com/)
 - Newport** (Newport Corporation - www.newport.com/)

Oriel (now Spectra-Physics - www.spectra-physics.com)
OptoSigma (OptoSigma Corporation - www.optosigma.com),

- a cell that can be used to specify the part name.

Using the catalog lens database

1. Choose the desired lens catalog by selecting the button labeled Catalog. A menu of the available lens catalogs will appear.
2. Choose the desired lens element type (singlet, doublet, or other) by selecting the corresponding *Lens Types* radio button.
3. Choose acceptable ranges of focal lengths and lens diameters by entering the center of the desired EFL range into the *Central Efl* cell and the extent of the range into the *+/- Range* cell. For example, if you wish to select a lens element with a focal length between 20 and 40 millimeters, enter 30 in the *Central Efl* cell and 10 in the *+/- Range* cell. (Note that negative lenses are considered to have negative focal lengths.) A range of lens diameters can be chosen similarly: enter the center of the range in the *Central Dia* cell and the extent of the range in the *+/- Range* cell.
4. Select the desired lens attribute (*EFL*, *Diameter*, or *Part* name) to be shown in the Scrolled List display by selecting the corresponding *Sort By* radio button. If *Efl* is selected, the scrolled list displays in ascending order the focal lengths of all lenses in the database whose focal length and diameter fall within the specified ranges. If *Diameter* is selected, the diameters of the lenses are listed in ascending order, and if *Part* is selected, the part names are listed in alphabetical order.
5. The Drawing Area displays a drawing of the lens corresponding to the highlighted entry in the Scrolled List, and the Message Area displays information about the lens. To highlight the various list entries, use the up- and down-arrow keys on the keyboard, or click the mouse button once on an entry.
6. When an appropriate lens is found, click the OK button (green check mark) to add the catalog lens element to the current lens. Click the CANCEL button (red "X" mark) to cancel the insertion of the catalog lens into the current lens.

Catalog lens elements cannot be saved except as part of an ordinary lens file; that is, it is not possible to add lens elements to the Database. Catalog lens elements that have been inserted into a lens can be modified only by first using the Ungroup feature in the Edit menu (see Chapter 3) to convert the elements to ordinary surface data that can be edited.

Perfect Lens

The third type of lens that can be entered by the File >> New command is a perfect lens. A perfect lens is defined to be a lens of a given focal length and aperture that forms a perfect image of a planar object at a given magnification. Every point in the object plane is focused to a corresponding point in the image plane, and the shape of the image is the same as the shape of the object.

At the given magnification, a perfect lens has no aberrations, but the image is still limited by diffraction effects. Indeed, one of the main applications of perfect lenses in optical design is to study the image of a diffraction-limited system. *If a perfect lens is used at other than the given magnification, it is no longer free from*

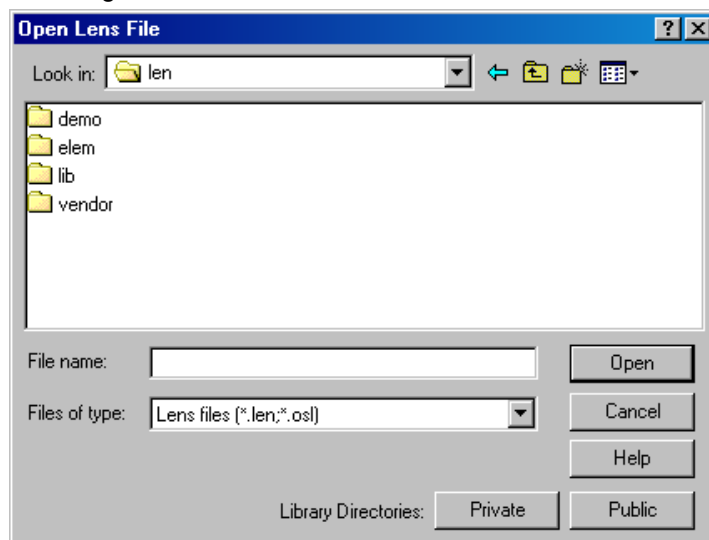
aberrations. Another application of perfect lenses is consequently to see the image degradation that may be expected under such operating conditions.

In OSLO, a perfect lens is implemented as a single plane surface. This surface is to be interpreted as a virtual surface that divides image space from object space, not as an actual surface. Because of the fundamental laws of optics, rays that strike this surface at a point on the object side do not emerge from the same point on the image side. To accommodate this fact in the lens drawing routines, it is necessary to use two surfaces to represent a perfect lens. The first surface is a dummy surface that shows the point of incidence of rays on the surface, and the second surface is the actual perfect lens, which shows how rays leave the surface.

Open Lens

Caution - The file opened by the “Open” command replaces the current file. Use the “Save” command or the “Save As” command to save the current file to disk if you wish to recover it later.

The Open command is used files that are on disk. OSLO displays the standard Windows file dialog box, from which the file name is selected.



Note - The “File Selection” dialog box in OSLO is similar to the one that appears in other applications. Note that it contains the extra Library Directory buttons: **Private** and **Public**.

By default, the drive and directory correspond your private\len directory. You can switch back and forth between the **Private** lens directory (.../private/len/) and **Public** lens directory (.../public/len/) by pressing the appropriate button. It is necessary to use the Drive Selector and Directory List controls only when opening files that are not in the default directory. If you open a lens from a third directory, that directory will become the current lens directory, and the next time you open a lens, the dialog box will show that directory.

Opening a lens file

1. Select File >> Open Lens, click the File open toolbar icon, or press the F2 key.
2. Use the Public and Private buttons, or the Drive selector and Directory List control to select the directory that contains the file.
3. Use the File Name List control to select the file to open, and click the OK button. Alternatively, a file can be opened by typing the file name in the File Name box and clicking the OK button. To cancel the file selection, click the CANCEL button.

The Files of Type field in the File Open dialog box shows the default value *.len, but also allows you to display all files (*.*), in order to view the entire contents of the current directory.

If the selected file is opened successfully, the file that has been opened becomes the current lens file, and its name and identification appear in the title bar of the main window. Note that, unlike the File >> New command, the Open command does not open the Surface Data spreadsheet, although lens files may be opened while the Surface Data spreadsheet is open.

In addition to opening OSLO lens files, OSLO can import lens files saved by other optical design programs. To import a lens from another program, use the File>>Import Lens File command, not File >> Open. (Or type **import filename** in the command line.) Note the distinction between *opening* an OSLO lens file and *importing* a lens file from another program. OSLO can interpret files that were created with CODE V[®], ZEMAX[®], SIGMA[®] and GENII[™] optical design programs. The extent of support for lens files in alternate file formats varies from program to program, but ordinary lenses consisting of centered spherical surfaces and standard catalog glasses can generally be imported with only minor modifications. Lens input commands that are not understood by the OSLO input filter will produce error messages in the current text window.

Files from earlier versions of OSLO (OSLO Series 2 & 3) must be translated using the Export (XPO) or Export file (XPF) commands in OSLO Series 2 & 3 (Revision 2.30 & 3.30). These files can then be opened directly in OSLO Standard or Premium; there is no need to import them.

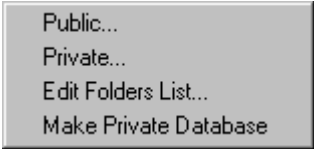
Save Lens

The Save command saves the current file to disk under the name that was given in the New or Open command. The Save command can be invoked with the menu, or simply by clicking the Save toolbar button. Saving the current lens frequently makes it easier to recover from mistakes.

Save Lens As

The current lens can be saved under a different file name, on a different disk, or in a different directory by using the Save As menu command. The Save As command displays the File Selection dialog box (see p. 160), which allows you to choose the name under which the current file will be saved. The Drive Selector and Directory List controls can be used to select the disk and directory for the file in the same manner as the Open command. Type the file name in the File Name box (it is not necessary to append the extension *.len to the file name) and click the OK button to save the file. The rules for file naming (long file names, etc.) are determined by the operating system that you are using.

Lens Database



Public...
Private...
Edit Folders List...
Make Private Database

In addition to the traditional *Open Lens* dialog box, it is possible to easily browse through the Public Lens Library shipped with OSLO, and even through your personal lens files (*.LEN and *.OSL).

This tool is based on OSLO's special implementation of databases; it is highly recommended to read the on-line help item "Contents>>User Interface>>Windows>>Databases" to make the best use of it.

Public / Private

Clicking *Public* or *Private* from the menu will bring up a list (a database) of systems, as well as a collection of description data.

Prescription		Spot Diagram				
Menu	LENS ID	DESIGNER	IMS	NAPER	FIELD	CLASS
1	Anamorphic Bravais objective 2x	OSLO	5	0.05	-2.9	OBJECTIVE
2	Anamorphic prism assembly	MGR	7	0.00	0.6	PRISM
3	Demo Triplet 50mm f/4 20deg	OSLO	7	0.12	20.0	COOKE TRIPL
4	Digital Triplet 10mm f/2.8 20deg	OSLO	7	0.18	20.0	COOKE TRIPL
5	Laser diode collimator 8mm .5 NA	OSLO	7	0.00	0.0	OBJECTIVE
6	Ebert Grating Monochromator	OSLO	4	0.14	0.0	DIFFRACTIV*
7	Dove Prism Example	OSLO	10	0.10	5.0	PRISM
8	Penta Prism	OSLO	10	0.00	0.0	PRISM

The fields are (column by column):

Column Name	Description
Menu	Numerical order of each design given the current sorting (which the user can change)
Lens ID	Name of the Lens
File path (Hidden field)	Location of the system on your computer. For example: <code>"%public%/len/lensfile.len", or</code> <code>"C:/My Documents/Archives"</code>
Designer	Designer Name (see des command)
IMS	Image surface number which gives the number of surfaces in the lens. Gives an indication of the complexity of a system.
NAPER	Numerical Aperture of the system in the image space $= n' \times \sin U'$
FIELD	Value of the Field Angle in Degrees
Class	Category the system belongs to: APERTURE, DOUBLE GAUSS, COOKE TRIplet, ...etc. This field is defined in the system as Surface Note #10 (command SNO10). Use this command to assign a category to the systems in your private directory.

I (IR)	'y' if the system is defined for use in Infrared Wavelength range
A (AFOCAL)	'y' if the system is evaluated in afocal mode (Image at infinity)
A (ASPHERIC)	'y' if the system includes one or more aspheric surfaces
D (DIFFRACTIVE)	'y' if system has one or more diffractive surfaces
G (GRADIENT)	'y' if the system includes a material defined with a gradient index
R (REFLECTIVE)	'y' if the system includes reflective surfaces
C (CATADIOPTRIC)	'y' if the system has one or more reflective surfaces
W (WIDE-ANGLE)	'y' if the system is defined in wide-angle ray aiming mode
N (NON-SEQUENTIAL)	'y' if the system has at least one non-sequential group
Z (ZOOM)	'y' if the system is defined with several configurations

You can sort the library by any item. For example, to find a non-sequential system, grab the horizontal slider to the right until you see the one-character wide fields. The next-to-last of these "short fields" is labeled *N*. Click on the *N*. A pop-up menu will inform you that the real name of this field is *NON-SEQUENTIAL*. Now go down the pop-up menu and select *Sort Down*. All the non-sequential systems in the database are now in the first rows.

As you probably already noticed, clicking on any row will plot a report graphic, giving you quick visual information about the selected system. If you are looking for more detailed information, you can click on one of the 2 top buttons labeled *Prescription* and *Spot Diagram*. The first one will print the lens data to the text window, while the latter will plot a spot diagram in graphic window #2. Those automated actions are called callbacks. It is a CCL routine that you can tailor to your need (See the code for DBCB_SHOWLENS in "PUBLIC/CCL/ASYST_DBCMDS.CCL"). If you wish, you can copy the function into your private CCL directory and customize it

Prescription		Spot Diagram			
Menu	LENS ID	DESIGNER	IMS	NAPER	F
1	Anamorphic Bravais objective 2x	OSLO	5	0.05	

Callback Buttons

A callback is a command or function that exists in memory until some action triggers the command or function to start. OSLO allows a special callback command to be defined within the database mode. The **Prescription** and **Spot Diagram** callback buttons are the only callback options currently defined in the default OSLO database implementation. A user can launch the appropriate callback response by clicking on the respective buttons. Users have the option of

deleting, adding or changing these callback buttons and the underlying callback command by accessing the Menu button described below.

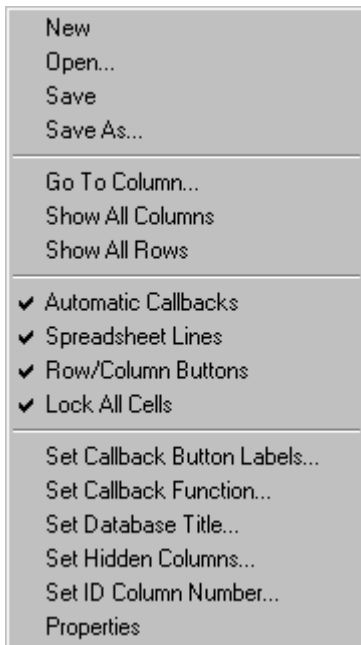
Prescription

The Prescription button will print a listing of the optical prescription (of the currently selected lens in the database) to the active text window.

Spot Diagram

The Spot Diagram button will plot a Through-Focus Spot Diagram to a new graphics window.

Menu



The Menu button brings up the following popup-menu:

New, Open, Save, Save As

These menu options allow you to control the creation and saving of new databases. Before using these options, you should explore the Database Tutorial which is available in the OSLO on-line help under "Contents>>Programming>>CCL Databases>>Database Tutorial".

WARNING: Use the "Save" option with extreme caution as it will overwrite your current database and it may be extremely difficult to recover from this situation if you do this unintentionally.

Go To Column

Allows you to move to (select) a particular column using the keyboard to navigate instead of the mouse

Show All Columns, Show All Rows

Controls the showing of previously hidden columns and rows.

Automatic Callbacks

This option controls the analysis that is performed (controlled by a callback) whenever a lens is selected. Disabling this option disables the automatic analysis that is performed.

Spreadsheet Lines

This option controls the display of the vertical and horizontal lines between individual cells in the spreadsheet.

Row/Column Buttons

This option controls the display of the row and column buttons

Lock All Cells

This option locks all cells from being edited.

Set Callback Button Labels

Allows the user to change the text shown on the face of the callback buttons. Two callback buttons are displayed ("Prescription" and "Spot Diagram") in the default database mode of OSLO.

Set Callback Function

Allows the user to change the name of the database callback function. One callback function controls all the callback actions in the database mode. The various callback actions are controlled by different parameters that are passed to the callback function. The default database callback function is *dbcb_showlens*.

Set Database Title

Allows the user to change the name of the database title. The default database mode has no title. Any title that is defined will appear in the white space to the right of the callback buttons.

Set Hidden Columns

Allows certain database columns to be hidden from the regular user. This allows data that the user would not ordinarily want to see to be stored with individual entries in the database.

Set ID Column Number

Allows the user to change the column that determines the database row ID.

Properties

A summary of the above menu items is shown in the message area underneath the command line. Before using any of these options, you should explore the Database Tutorial which is available in the OSLO on-line help under "Contents>>Programming>>CCL Databases>>Database Tutorial".

Lens ID, Designer, ...etc.

LENS ID
Cut
Copy
Paste
Hide
Delete
Insert Left
Insert Right
Sort Up
Sort Down
Use Case-Sensitive Sort
Set Column Name...
Set Column Width...
Set Data Type...
Make List From File...
Make List From Column
Show List
Remove List
✓ Lock Cells
Properties

The first item in the pop-up menu when the column header is chosen refers to the data contained in the column. In some cases, the wording on the header button is abbreviated (i.e. "Z" for "Zoom"). The first item in the pop-up menu will provide a more understandable name.

Cut

Stores the all the current column data in the Windows clipboard and deletes the column entry in the database.

Copy

Stores a copy of the current column data in the Windows clipboard. The column data in the database is unaffected.

Paste

Inserts a new column to the left of the currently selected column in the database spreadsheet. This new column is populated with data from the Windows clipboard.

Hide

Removes the current column from display in the database spreadsheet. The data from the column is still accessible through the CCL programming interface.

Delete

Removes the current column from the database.

Insert Left / Insert Right

Inserts a new column to the left/right of the current column. At the time the column is created, the user will be asked if the column should contain integer, real number or string data.

Sort Up / Sort Down

Sorts the data in the column alpha-numeric order. *Sort Up* means that the data will be sorted in increasing (i.e. alphabetical) order; *Sort Down* means that the data will be sorted in decreasing (i.e. inverse alphabetical) order.

Use Case-Sensitive Sort

This sets in on/off flag which governs whether the following sorts will be performed in upper/lower “Case-Sensitive” mode.

Set Column Name

Allows the user to change the column name.

Set Column Width

Allows the user to change the column width

Set Data Type

Allows the user to change the column data type to integer, real number or string. Note that data in the column may be lost when performing this action.

Make List from File

Makes a pop-up list from the data in the cells of the current column. Future users can access this list by clicking on a cell in the column. See “Database Tutorial” in the on-line help for more information.

Make List from Column

Makes a pop-up list from the data in a CDB file. Future users can access this list by clicking on a cell in the column. See “Database Tutorial” in the on-line help for more information.

Show List

Displays the contents of the column list in text window (see “Make List” from above).

Remove List

Removes the column list from the current column (see “Make List” from above).

Lock Cells

This sets an on/off flag which governs whether cells in the current column can be edited.

Properties

A summary of the above menu items is shown in the message area underneath the command line. Before using any of these options, you should explore the Database Tutorial which is available in the OSLO on-line help under "Contents>>Programming>>CCL Databases>>Database Tutorial".

Edit Folders List

By default, the Private Lens Library will only include the systems defined in the "...Private/len/" folder. Using this menu item, you can add as many folders as you want by editing the Folder List.

Each line corresponds to one folder. The path separators should be marked as '/' (slash), not as '\' (backslash). For example:

```
%private%/len/  
  
%private%/len/old/  
  
C:/My Documents/Oslo/Archived Systems/
```

Note: "%private%" refers to the path of your private directory, as defined in the string preference "Private_directory".

You will have to rebuild the Private Lens Library to see the new folders appear.

Make Private Databases

This menu item rebuilds the private lens database.

The private lens database files you browse are generated by a routine that opens all the lens files in your private directory and gathers all the information needed for the database. As a consequence, if you work on a new system, and save it under "...private/len/my_new_system.len", it will not automatically appear in the private lens database. All you have to do is choose this menu item to rebuild the private lens database before attempting to browse through it.

Import Lens File

Lt**Std****Prm**

OSLO provides the possibility to import files from other software. As optical designs can be very complex and the filter is limited, it is strongly recommended to check if all the data has been imported, and if the imported data has been imported accurately.

Those import filters are provided as a utility that reduces manual input time. Lambda Research does NOT support those file formats, nor makes any claim that the data will be imported accurately.

The following formats can be imported:

- GENII
- Sigma (*.len;*.lens;*.dat)
- CodeV (*.seq)
- Zemax (*.zmx)

You can append measured wavefront data in the form of interferogram files (see the section “Interferometric Deformation (I)” on page 112).

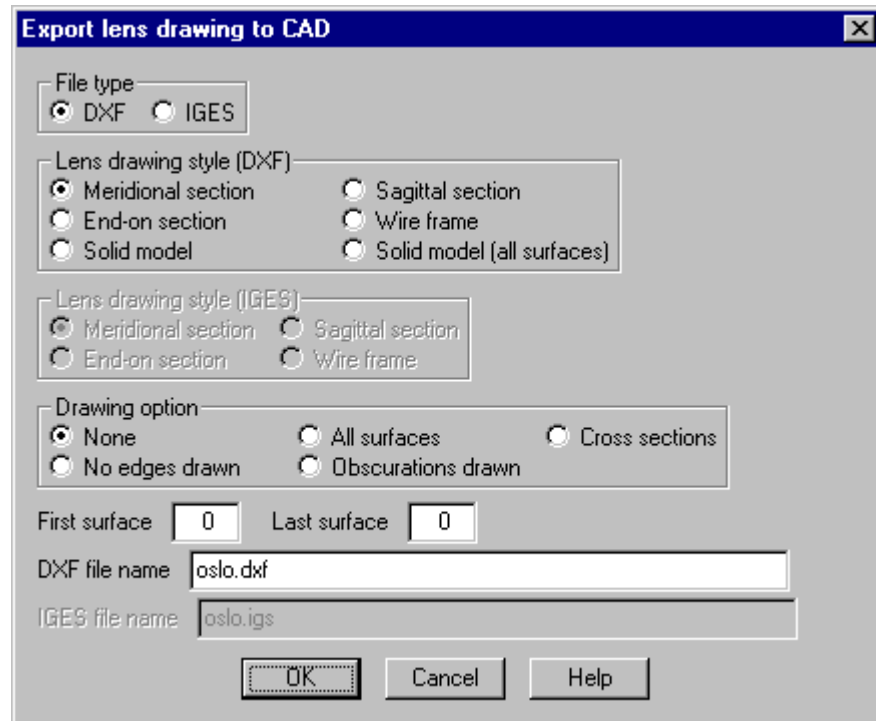
Export Lens to CAD

Lt**Std****Prm**

The Export to CAD command provides output of the current lens in either DXF or IGES format, for import into other CAD software. Both options produce text output files that can be imported into AutoCAD™ and similar programs. The output obtained from the Export to CAD command is prepared directly from the lens data in memory. It is not a print file that corresponds to the current graphics window.

It should be noted that the DXF output format is 3D. Some programs (such as Microsoft Word™) only accommodate 2D DXF, and will not import OSLO DXF output.

The meridional section and sagittal section drawings are inherently 2D, but the viewing conventions in AutoCAD show y vs. x. This means that these images must be rotated 90 degrees after they are imported before they can be viewed properly. The wire frame view is inherently 3D and should display properly (in end view) as imported.



The **File type** argument determines the type of file produced: **DXF** (Drawing Exchange Format) or **IGES** (Initial Graphics Exchange Standard).

For DXF drawings, six styles can be produced:

Meridional section produces a drawing in the Y-Z plane.

Sagittal section produces a drawing in the X-Z plane.

End-on section produces a drawing in the X-Y plane.

Wire frame produces a three-dimensional wire-frame drawing.

Solid model produces a three-dimensional solid-model drawing with hidden surfaces removed. "Dummy" surfaces are not drawn.

Solid model (all surfaces) produces a three-dimensional solid-model drawing that includes "dummy" surfaces.

For IGES files, only meridional, sagittal, and end-on sections and wire frame drawing styles are available.

For non-solid-model DXF and IGES files, five additional **Drawing options** exist:

None produces a normal drawing; that is, "dummy" surfaces are not drawn, edges are drawn, obscurations are not drawn, and surfaces are not drawn as cross-sections.

All surfaces specifies that "dummy" surfaces are to be drawn.

Cross sections specifies that each element is to be drawn as a cross-section.

No edges drawn specifies that only surfaces are to be drawn, not the edges between the surfaces.

Obscurations drawn specifies that obscurations are to be shown in drawings.

The range of surfaces to be drawn is determined by the **First surface** and **Last surface** arguments. The defaults of 0 and 0 specify that surfaces 1 through (image surface – 1) are to be drawn.

The name of the DXF or IGES drawing file to be produced is given by the **DXF file name** or **IGES file name** argument, respectively. If the file name does not specify a directory path, the file is placed in the user's Private directory.

For each drawing type, default drawing rays are drawn as specified by the "Lens>>Lens Drawing Conditions" items: Number of field points for ray fans, Fractional Y Object, Fractional X Object, Rays, Minimum Pupil, Maximum Pupil, Offset, FY or FX fan and Wavelength number.

Note that OSLO's coordinate system differs from the default coordinate system of AutoCAD (and other CAD programs); in OSLO, the Z axis is usually coincident with the optical axis, whereas in AutoCAD the Z axis (by default) is normal to the display. Therefore, it is usually necessary to rotate lens drawings after they have been opened in a CAD program. Two possible sequences of AutoCAD commands to perform this rotation are:

```
ucs y -90  
plan current
```

or

```
rotate3d y 0,0,0 90
```

Open Database

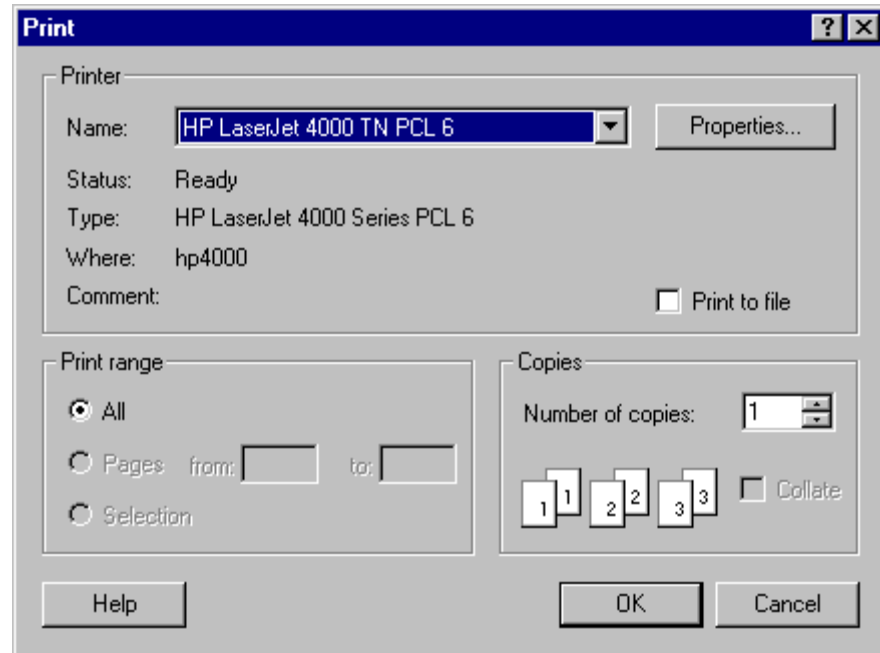
Other databases that have been setup to work within OSLO can be accessed using this menu option. As an example, you can browse the Schott Glass catalog by choosing the "Library Directories" = "Public" button and then opening the "Glass_schott.cdb" file. Of course, an easier way to access this same file is to choose the "Lens>>Glass Catalogs>>Browse Glass Database>>Schott" from the OSLO menu.

Print Text Window

Note - The contents of a text window can also be copied to the Windows clipboard using the “Text >> Copy to Clipboard” command on the “Window” menu. The text on the clipboard can be pasted into most Windows-compatible text editors or word processors.

The Print Text Window menu command is used for printing the contents of the text output window or for saving the contents of the text output window to a print file. To save the contents of the text output window in a text file, use the Save Text As command. To establish the proper configuration for printing, use the Page Setup command (see p. 171).

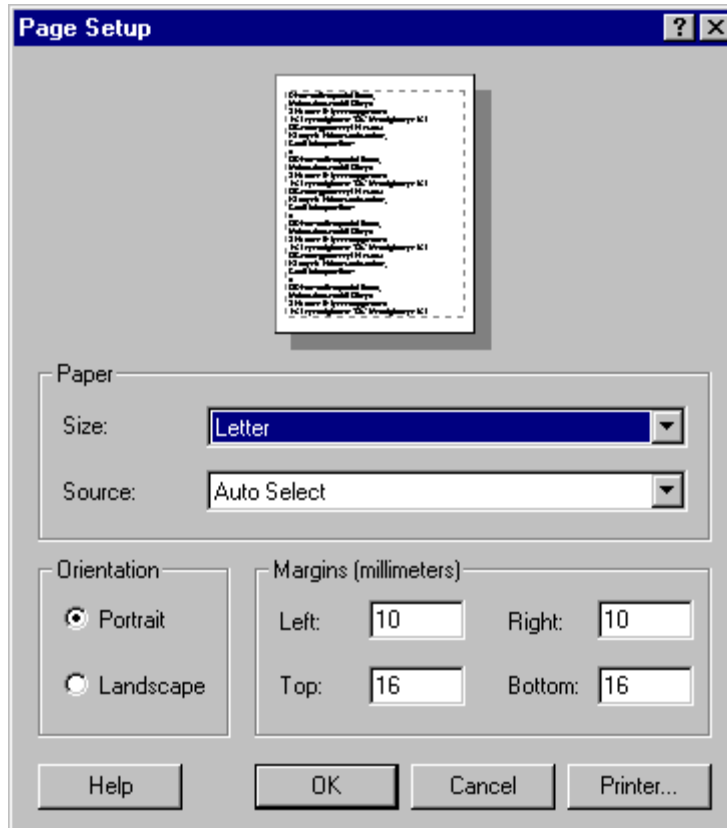
The Print Text Window command displays the standard Print dialog box. The appearance of this box will vary somewhat according to the operating system that you are using.



The output from the Print Text Window command consists of ASCII text that includes line breaks. OSLO uses the standard interface routines for Windows supplied by your printer manufacturer, so the appearance of the output will depend the vendor-supplied print driver and on how you have set up your printer to work with your version of Windows.

Since OSLO does not insert page breaks into its text output window and does not allow ranges of text to be selected, the Selection and Pages buttons are disabled. The amount of text that is printed is determined by the setting of the **output_page_mode (page)** preference in OSLO. If **page** mode is on, the contents of the text window will be printed, starting with the first visible line; if **page** mode is off, the entire contents of the text window are printed (i.e., including previous output). To cancel printing, press CANCEL.

Page Setup (Text)



The Page Setup command displays the standard Windows Page Setup dialog box. The same box is used for both text and graphics windows. The appearance of this dialog box will depend upon which version of Windows is running and which printer has been selected as the default. The controls in the box labeled Printer are used to determine which printer is used to print the text. The orientation of printed text is determined by the Orientation controls. Text is usually printed in Portrait mode. OSLO sets the default orientation to portrait mode at startup.

Save Text As

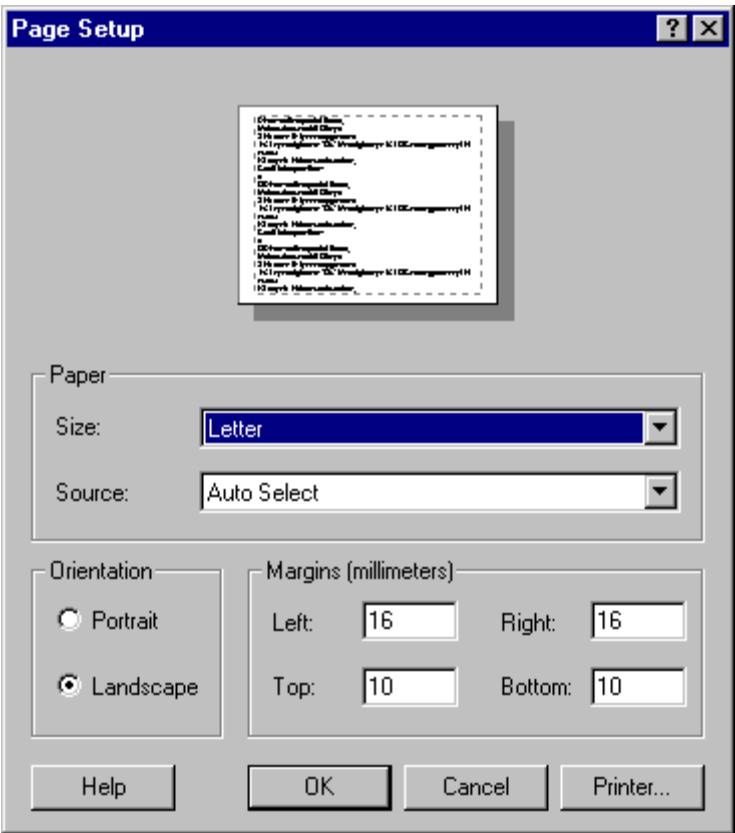
To save the contents of the text output window in a text file, use the File >> Save Text As command. A file selection dialog box will appear; use the controls to give the name of a text file in which the text will be saved. The text file can be edited with a text editor or word processor or incorporated into another document.

Print Graphics Window

Under Windows, OSLO's graphics windows can be printed on nearly any printer or plotter that works with Windows. The Print Graphics Window menu command displays the same Print dialog box as the Print Text Window command.

Page Setup (Graphics)

The Page Setup command displays the standard Windows Page Setup dialog box. The same box is used for both text and graphics windows.



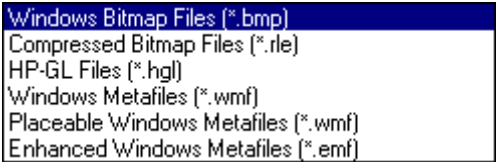
The appearance of this dialog box will depend upon which version of Windows is running and which printer has been selected as the default. The controls in the box labeled Printer are used to determine which printer is used to print the graphics. The orientation of printed graphics is determined by the Orientation controls. OSLO arranges most graphics to appear best when printed in Land-scape mode (the major exceptions to this rule are ISO element drawings, which appear best in Por-trait mode). OSLO sets the default orientation to land-scape mode at startup.

Save Graphics As

Note - The “Graphics >> Copy to Clipboard” command on the “Window” menu copies the currently selected graphics window to the Windows clipboard (in the Windows metafile format). The graphics on the clipboard can be pasted into most Windows-compatible word processors or graphics programs.

The contents of a graphics window can also be saved in any of several standard graphics file formats, by using the File >> Save Graphics As command. This opens the standard file dialog box.

OSLO can save graphics in six formats. The format of the saved graphics file is determined by the Save as type menu in the dialog box:



- Windows bitmap (BMP) files are binary uncompressed bitmap files in a standard Microsoft format and can be opened by bitmap-oriented graphics programs such as Paint or Paintbrush. BMP files created by OSLO are in the 8-bits-per-pixel (256-color) format.
- Compressed bitmap (RLE) files are a compressed variant of BMP files that can be read by some Windows bitmap editors, e.g., Paint.

- Hewlett-Packard graphics language (HP-GL) files are text files that contain instructions that control Hewlett-Packard (or compatible) pen plotters, as well as some laser printers.
- Windows metafiles (WMF) are binary files containing drawing commands. Unlike bitmaps, metafiles can be scaled without distorting the image. Note that these files can not be read by some applications, such as Microsoft Word; placeable metafiles must be used instead.
- Placeable Windows metafiles (also denoted by WMF) are variants of Windows metafiles with extra information that is used to determine the size of the image. Some applications, such as Microsoft Word, require placeable metafiles.
- Enhanced Windows metafiles (EMF) are variants of Windows metafiles with extra information that is used by some 32-bit applications. This option is only available under Windows 95 and Windows NT.

Preferences



Unlike operating conditions, which are associated with (and saved with) each lens file, preferences determine the attributes of the OSLO application. For example, the **Graphics_white_bkgnd** preference can be used to make the background color of the graphics windows black or white.

Preference Groups

Some preferences can be placed in logical groups and accessed through a more convenient dialog interface.

Graphics

Many of the graphics related preferences are grouped together under this menu item so that they can be addressed in a single dialog box. The only preference that might require more explanation is the “Pen Sequence Control Word”.

You can visualize the color palette in OSLO by issuing the command, **colors**, on the command line.



You will note that the color palette contains 210 colors numbered from 0 through 209. Using the mouse to zoom in on the graphic, you can see that the default pen sequence is listed as 0 through 7. This is the order that OSLO uses the pen colors in most graphics (you will notice this sequence used when you trace rays at multiple wavelengths in drawings). You could see that it might be a little confusing to assign a color number to a pen sequence number. For simplicity, the first 26 colors are assigned the letters of the alphabet and you just list these letters in the pen sequence you desire. If you want the first 4 colors of the pen sequence to be black-red-blue-green, the pen sequence would be ADCB.

Std **Prm** Program Size

This command sets several preference related to program size. OSLO uses these number to allocate the appropriate resources when the program starts. If these numbers are exceeded during program operation, OSLO will automatically allocate the memory it needs.

Pen Sequence

In addition to the alphabetic method of defining a pen sequence (see “Graphics” preferences above), this menu option identifies a method of creating a pen sequence that evenly defines a given number of colors in a given color range. This is useful if you want a pen sequence to mimic the a sampling of the spectrum.

Set Preference / Show Preference

All preferences can be accessed through these menu options even if they are also available in Preference Groups above. The values of the various preferences are changed by using the “Set Preference” command (**stp**). When doing this, the current value of the preference is shown before the user has the opportunity to change it. If you don’t want to inadvertently change the preferences, it is recommended to use the “Show Preference” command (**shp**) which only displays the preference values. Since some preferences are “read-only” and cannot be changed, these preferences are only found under “Show Preference” and not under the “Set Preference” option.

There are four types of preferences: Boolean, Integer, Real, and String. Boolean preferences can have one of two values: “On” (1) or “Off” (0). Real preferences are floating-point numbers. String preferences consist of an array of characters, e.g., “ABC Widget Co.” (the quotes are not considered to be part of the string). The value of preferences can be retrieved using the commands **real_pref**(*preference_name*), **int_pref**(*preference_name*), **char_pref**(*preference_name*) [used for Boolean preferences], or **str_pref**(*preference_name*). The last can only be used as an argument in another command, e.g., **print str_pref**(*adr1*).

The available preferences are described in the table below. Each preference has a short and a long-form name. The long name is shown in the description. In the table, the type is shown with a tag indicating whether it is a read-write (**rw**)

preference, or read-only (r). Read-only preferences can be displayed using **shp**, but can not be set using **stp**.

Name	Type	Description
adr1 adr2 adr3	str.rw	Address1 (2,3) - These preferences are used for the address field in element drawings. Each string may be up to 36 characters long.
ccac	bool.r w	CCL_auto_compile - If this preference is On, the CCL compiler will be automatically run whenever a CCL file is saved from the OSLO text editor.
ccop	str.rw	CCL_compiler_options - This string is passed to the CCL compiler when invoked. It allows for overriding the default storage allotment sizes, etc.
chal	bool.r w	Cmd_history_aliases - If this preference is On, command history buffer entries created by OSLO will use the short form (i.e., the alias) for OSLO commands.
cdir	str.r	Current_directory - The full pathname of the current working lens directory.
tdir	str.r	Current_text_directory - The full pathname of the current working text file directory.
debug	bool.r w	Debug - If this preference is On, a traceback list will be generated for commands that fail to execute successfully. This preference is designed to be used during the debugging phase of CCL development.
dsgn	str.rw	Designer - This preference is the default for the “Designer” General Operating Condition. It may be up to ten characters long.
efil	str.r	Edit_file - The full pathname of the current editor file.
edcm	bool.r w	Element_drawing_commas - If this preference is On, a comma will be used as the separator between the integral and fractional parts of real numbers in element drawings. If this preference is Off, a decimal point will be used.
epwc	bool.r w	Entrance_pup_wvfrnt_coords - If this preference is On, wavefront maps and contour plots will be plotted verse fractional entrance (rather than exit) pupil coordinates.
expr	int.r	Expiration_date - The expiration date, if any, for the current license.

fpno	bool.r w	Five_place_numeric_output - If this preference is On, the program will output numeric data with five places to the right of the decimal, which allows it to print numbers between 1000 and 9999 with normal rather than exponential notation.
gfam	bool.r w	Graphics_alternate_mode - Determines the aspect ratio of graphics that are printed, saved in an HP-GL file, or copied to the clipboard. OSLO uses the aspect ratio of a graphics window to determine the aspect ratio of exported graphics. By default, if a window's width is greater than its height, exported graphics are scaled to appear best in landscape mode; otherwise graphics are scaled to appear best in portrait mode. If this preference is On, the aspect of exported graphics is opposite to the aspect of the graphics window.
gac1	bool.r w	Graphics_autoclear - Determines whether graphics windows are automatically cleared when graphics commands are issued. Setting this preference to Off allows the output from several graphics commands to be superimposed.
grax	bool.r w	Graphics_axes - Determines whether axes are drawn in plots.
gfbw	bool.r w	Graphics_black_and_white - If this preference is On, only black and white are used in graphics windows.
scal	real.r	Graphics_current_scale - Current value of the graphics scale. If the Graphics_scale preference is nonzero, Graphics_current_scale is equal to Graphics_scale; otherwise, Graphics_current_scale is the scale factor chosen by OSLO in the most recently issued graphics command.
hgl2	bool.r w	Graphics_HPGL2 - If this preference is On, HP-GL graphics files will be written using the HP-GL/2 command set. Turn this preference Off if you have an older graphics output device that does not support HP-GL/2.
glab	bool.r w	Graphics_labels - Determines whether graphics output contains annotation.
gpcl	int.r	Graphics_PCL - If this preference is On, a PCL prolog and epilog will be added to graphics output sent to a printer or plot file (UNIX only).

pens	array of int.rw	Graphics_pen_sequence - This array defines the sequence of pen colors used by commands.
penw	int.rw	Graphics_pen_width - This preference controls the width of lines in printed output and in graphics files produced by OSLO. The preference is expressed as a percentage of the pen width relative to the “normal width” (which is 0.001 times the diagonal length of the paper). Setting the preference to zero produces the narrowest lines (i.e. one printer pixel) possible on the output device. For example, to set the pen width to five times its normal width, set the preference value to 500.
gsca	real.rw	Graphics_scale - Sets the scale to be used for the next graphics command. 0 implies automatic scaling. Reset to 0 after each OSLO command.
gfwb	bool.r w	Graphics_white_bkgnd - Determines whether the background color for the graphics windows is black or white. If this preference is Off, the background is black. White background should usually be specified when printing or when using the “Copy to clipboard” feature to paste OSLO graphics into other Windows applications.
lauo	bool.r w	Lens_auto_open - If this preference is On, when OSLO is restarted, the lens file that was open during the previous session is reopened. Note that the file can be reopened only if it was saved before exiting OSLO in the previous session.
lfil	str.r	Lens_file - The full pathname of the current lens file.
mcfg	int.rw	Max_config_items - The maximum number of configuration data items that may be defined for a lens.
mspd	int.rw	Max_spd_rays - The maximum number of rays contained in a single spot diagram.
msrf	int.rw	Max_surfaces - The maximum number of surfaces that a lens may contain.
nbru	int.r	Max_users - The maximum number of simultaneous users allowed for this license of OSLO.
mwvl	int.rw	Max_wavelengths - The maximum number of wavelengths that may be defined for a lens.

mrfl mrf2 mrf3 mrf4	str.r	MRU_file_1 (2,3,4) - These preferences contain the full pathnames of the last four most recently opened lens files and are displayed on the File menu.
noeb	bool.r w	No_error_boxes - If this preference is On, the program will display error messages in the main window, rather than in modal alert boxes.
nzsd	bool.r w	Non_zero_special_data - If this preference is Off, all coefficients of special data described by an expansion polynomial (aspheric surfaces, diffractive surfaces, etc.) are displayed. If this preference is On, only those coefficients that are non-zero will be displayed.
opco	int.rw	Oprd_maxconops - The maximum number of constraint operands contained in a single error function.
opec	int.rw	Oprd_maxextcmp - The maximum number of external operand components (i.e., computed by a DLL) contained in a single error function.
opfp	int.rw	Oprd_maxfpts - The maximum number of field points in a single optimization field points set.
opop	int.rw	Oprd_maxoperands - The maximum number of operands contained in a single error function.
opra	int.rw	Oprd_maxrays - The maximum number of rays in a single optimization ray set.
opsd	int.rw	Oprd_maxspds - The maximum number of spot diagrams in a single optimization spot diagram set.
opvb	int.rw	Oprd_maxvariables - The maximum number of optimization variables for a single lens.
osdp	str.r	Osdata_path - This is the full pathname of the private subdirectory for the OSLO file system.
echo	bool.r w	Output_echo - If this preference is On, commands are echoed in the current text output window when they are issued.
olog	bool.r w	Output_log - If this preference is On, text output is sent to a file named oslog.txt in your private directory.

page	bool.r w	Output_page_mode - If this preference is On, the first line of text output from each command appears at the top of the text output window; otherwise, text output is scrolled as it is produced. Also, if this preference is On, print text window prints from the first displayed line to the end of the window, otherwise it prints the entire window.
outp	bool.r w	Output_text - If this preference is On, text output from OSLO commands appears in the text output window; otherwise, no text output is produced. Numeric output is saved in the spreadsheet buffer regardless of the setting of this preference.
plod	int.r	Plot_destination - The destination (plotter (0) or file (1)) when a graphics window is printed. (UNIX only)
plof	str.r	Plot_file - The full pathname for graphics printed to a file. (UNIX only)
prtd	int.r	Print_destination - The destination (printer (0) or file (1)) when a text window is printed. (UNIX only)
prtf	str.r	Print_file - The full pathname for text printed to a file. (UNIX only)
prtz	bool.r w	Print_zero - If this preference is Off, real numbers that are identically zero are printed as "--". If this preference is On, zero-valued real numbers are printed as "0.0". You may wish to turn this preference On if you want to export numerical data from OSLO to another application, such as a spreadsheet.
prlv	int.r	Program_level - The level of the OSLO program (Light = 1, Standard = 2, Premium = 3)
pubd	str.r	Public_directory - The full pathname of the public subdirectory of the directory in which OSLO is installed.
rvrt	bool.r w	Revert_enable - If this preference is On, OSLO will prompt for whether you wish to revert the current lens to its previous state if CANCEL is pressed after changes have been made in a spreadsheet.
rgdt	bool.r w	Rpt_gfx_date_and_time - If this preference is On, the current date and time will be printed in the lower right-hand corner title block area of the report graphics output.
shmb	bool.r w	Show_menubar - If this preference is Off, the main menu bar will not be displayed when OSLO is started.

shtb	bool.r w	Show_toolbar - If this preference is Off, the main tool bar will not be displayed.
scvm	bool.r w	Spreadsheet_curvature_mode - If this preference is On, the Surface Data spreadsheet displays surface curvatures; otherwise, it displays radii of curvature.
tmpd	str.r	Temporary_directory - The full pathname of the directory used by OSLO for storing temporary files.
tglf	str.rw	Test_glass_file - The name of file containing the radii of test glasses used when requesting a test plate fit in the surface data spreadsheet.
trno	bool.r w	Trim_numeric_output - If this preference is On, trailing zeros in text output will be suppressed.
vers	str.r	Version_number - The revision number of OSLO.
wvld	array of real.rw	Wavelength_default - These are the defaults for the first three entries in the wavelengths set. When a new lens is created, the wavelengths will be set equal to the values of this preference. Note that element 0 in the preferences array corresponds to wavelength 1, element 1 corresponds to wavelength 2, etc.
wgtd	array of real.rw	Weight_default - These are the default wavelength weights corresponding to the Wavelength_default preference array.

Restore Default Preferences

This option (**prefs**) restores default preferences and operating conditions.

Reopening a previous file

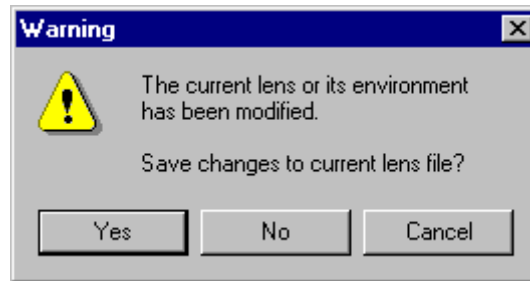
OSLO displays the name of the last ten saved files in the File menu above the Exit command. Selecting one of these items opens the file, which replaces the current lens file. The number of previous files that are referenced can be increased to 20 by editing the "a_menu.ccl" file in your "...private/ccl/" directory.

Note that you cannot revert to a previous version of a file that has not been saved to disk. Although the New menu command sets the name of the current file and places it on the title bar, the file does not exist on disk until it has been explicitly saved.

Exiting OSLO

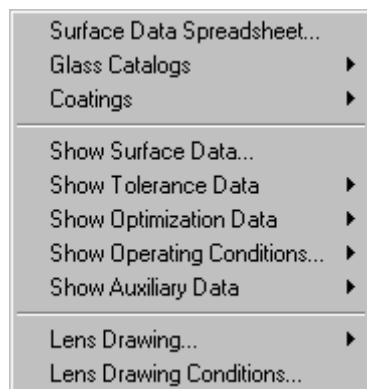
The Exit menu command is used to exit the OSLO application. If OSLO determines that you have not saved the current file to disk since it was created or last modified, a warning box appears:

Note - The warning box appears if there have been *any* changes to the lens. Some changes may be made automatically by the program, for example, when updating a lens originally saved from a previous version of OSLO.



Click "Yes" to save the current file to disk; if the **Lens_auto_open (lauo)** preference has been set, the current file will be reopened the next time OSLO is run. Click "No" to exit OSLO without saving the file. Click "Cancel" to continue using OSLO.

Overview



In OSLO, a lens is defined as a series of surfaces, plus system data such as wavelengths and optimization data such as the operands set. Lens data that are represented by variable-length sets of entries are edited in spreadsheets. These data sets are

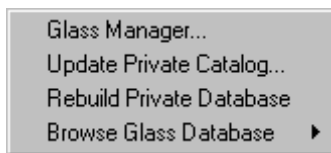
- Surface Data
- Wavelengths
- Configuration Data
- Tolerance Data
- Optimization Data: Variables, Ray Set, Field Points Set, and Operands (*In OSLO Premium, there is also a Spot Diagram Set*).

The commands in the Edit menu are used to add, delete, or rearrange the entries in each of these sets of data. The details on actually filling in the contents of the spreadsheets are presented in the Update and Optimize chapters.

Menu Item	Page
Surface Data Spreadsheet... Enter the construction data for the lens.	184
Glass Catalogs Browse public and update private glass catalogs.	184
Coatings Enter and modify multilayer coating material data, coating designs and visualize coating thickness uniformity.	191
Show Surface Data... Lists lens data such as surface data, aperture data, special data, refractive indices, solves, and pick-ups in the current text window.	199
Show Tolerance Data... Lists the tolerances for surface form, thickness, refractive index, tilt, and decentrations in the current text window.	205
Show Optimization Data... Lists the optimization variables, ray set data, and error function operand data in the current text window.	207
Show Operating Conditions... Lists the operating conditions in the current text window.	213

- Show Auxiliary Data** Contains a submenu that provides utility routines for computing support data, such as edge thickness, surface sag, diffractive surface profiles, and gradient index values. 217
- Lens Drawing...** Draws a plan view, wire frame view, or solid model picture of the lens system, showing ray trajectories in the current graphics window. 224
- Lens Drawing Conditions...** Edit parameters which control how rays and lenses are shown in graphics and drawings. 238

Glass Catalogs



Materials for lens elements in OSLO are selected from any of eight *glass catalogs*: **Schott** (regular, 2000, and Radiation Hard), **Ohara**, **Hoya**, **Corning**, **Sumita**, **Hikari**, **Pilkington**, **Miscellaneous (Misc)**, **Obsolete**, **Shared**, and **Private**. The first nine catalogs contain data for glasses available from the respective manufacturers.

The **Miscellaneous** catalog contains data for a few commonly used optical materials, such as silica and zinc selenide.

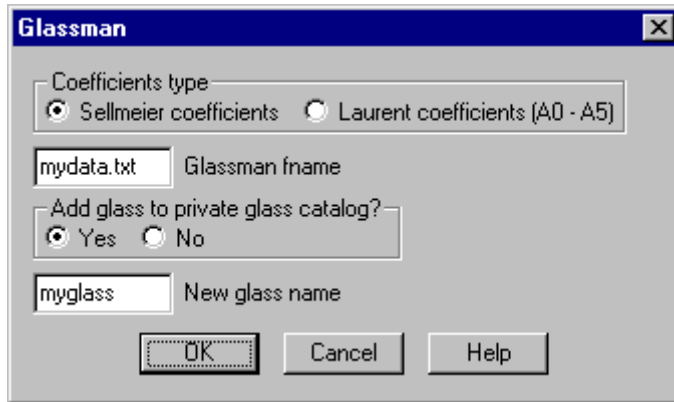
The **Obsolete** catalog contains data for discontinued catalog glasses. Some of the lens files supplied with OSLO are old designs and use these obsolete glasses. If OSLO cannot find a glass in one of the other catalogs, OSLO will search for the glass in the Obsolete catalog. A warning message will be displayed if OSLO substitutes an obsolete glass in a lens.

The **Shared** glass catalog is a “public” catalog that is located in the “.../bin/glc” subdirectory of the OSLO installation directory. This file can be replaced by a user-provided file (typically a copy of a private glass catalog) that can be accessed by all users of OSLO. This makes it convenient for multi-user sites to share user-defined glasses.

The **Private** catalog initially contains no data; unlike the other glass catalogs. However, unlike the other catalogs, the private catalog may be edited by the user using the “Update Private Catalog” menu item (described on page 188).

Glass Manager

The Glass Manager (**glassman**) command computes glass dispersion coefficients for either Laurent (“old” Schott) or Sellmeier (“new” Schott) models. This command makes it convenient for you to add a glass to your private catalog from the data you have, which is usually a series of Wavelength/index terms.



The Glass Manager command will read in the values from a plain ASCII text file located in your current private directory (usually the ".../private/" folder). Alternatively, you can specify the full path name of the file when prompted. The file should have the following format:

```
<wavelength sample 1> <index sample 1>
<wavelength sample 2> <index sample 2>
.
.
.
<wavelength sample n> <index sample n>
```

where n is the number of samples.

For example, the contents of a typical file could be listed as

```
0.4 1.5
0.5 1.6
0.6 1.7
0.7 1.65
```

OSLO then performs a least-squares optimization on the file data to determine the values of coefficients that give the best fit to the supplied data. There is also an option to add the glass directly to your Private Glass catalog without you having to do it manually.

Since the optimization requires that the current lens in memory be overwritten, OSLO requires that the Surface Data Spreadsheet window be closed before performing the optimization. It is recommended that you save the current lens before running the **glassman** command.

The form of the **Laurent**-series dispersion model is

(7.1)

$$n^2(\lambda) = A_0 + A_1\lambda^2 + \frac{A_2}{\lambda^2} + \frac{A_3}{\lambda^4} + \frac{A_4}{\lambda^6} + \frac{A_5}{\lambda^8}$$

where n is the refractive index, λ is the wavelength in μm , and A_0 , A_1 , A_2 , A_3 , A_4 , and A_5 are the dispersion equation coefficients.

The form of the **Sellmeier**-series model is

(7.2)

$$n^2(\lambda) = 1.0 + \frac{b_1\lambda^2}{\lambda^2 - c_1} + \frac{b_2\lambda^2}{\lambda^2 - c_2} + \frac{b_3\lambda^2}{\lambda^2 - c_3}$$

where n is the refractive index, λ is the wavelength in μm , and b_1 , b_2 , b_3 , c_1 , c_2 , and c_3 are the dispersion equation coefficients.

The Sellmeier model is generally found to provide a more accurate fit to experimental data, especially outside the visible spectrum. Note that the Schott glass catalog glasses now use the Sellmeier model.

Point #	Wavelength	Refractive index
1	0.400000	1.500000
2	0.500000	1.600000
3	0.600000	1.700000
4	0.700000	1.650000

RMS error of fit = 0.032408

Coefficients:

```

B1      0.923922
B2      1.439083
B3     -4.1590e+04
C1      -0.129618
C2      -0.128967
C3     -9.7021e+05

```

STORED GLASS MYGLASS

After OSLO computes the coefficients for the chosen model, it reports the following data in the current text window:

- A summary of the data read from the ASCII data file,
- An RMS error for the fit (a good fit should have an error of less than 1.0×10^{-5}),
- The coefficients **glassman** calculated, and
- An indication if the glass was added to the Private Catalog.

Update Private Catalog

The Options >> Update Glass Catalog command is used to add or delete entries in the private glass catalog (".../private/bin/glc/private.glc").

To add a new material to the private glass catalog, first select the radio button labeled Add glass to catalog. Then enter the name of the new material in the Glass name cell. (If you enter the name of a material that already exists in the catalog, the new data will overwrite the old data.) The refractive index data for the new material can be specified in one of three ways: by entering the six Schott dispersion equation coefficients of either a Laurent series (“old” Schott formula) or a Sellmeier series (“new” Schott formula), or by entering three wavelengths and three corresponding refractive indices, in which case refractive indices are interpolated according to the Conrady dispersion equation. The Laurent series dispersion equation specifies refractive index according to the formula

(7.3)

$$n^2(\lambda) = A_0 + A_1\lambda^2 + \frac{A_2}{\lambda^2} + \frac{A_3}{\lambda^4} + \frac{A_4}{\lambda^6} + \frac{A_5}{\lambda^8}$$

where the wavelength λ is in microns. The form of the Sellmeier series dispersion equation is

(7.4)

$$n^2(\lambda) = 1.0 + \frac{b_1\lambda^2}{\lambda^2 - c_1} + \frac{b_2\lambda^2}{\lambda^2 - c_2} + \frac{b_3\lambda^2}{\lambda^2 - c_3}$$

The Sellmeier series model is generally found to provide a more accurate fit to experimental data, especially outside the visible spectrum, as compared to the Laurent series model. The Conrady dispersion equation is given by

(7.5)

$$n(\lambda) = n_0 + \frac{A}{\lambda} + \frac{B}{\lambda^{3.5}}$$

If you enter three wavelength-refractive index pairs, the coefficients n_0 , A , and B in Eq. (8.17) are obtained by solving the set of three simultaneous equations produced by entering the three values of λ and n into the dispersion equation. These values are then stored in the catalog.

The thermal coefficient of expansion (α ; TCE) may be entered in the cell labeled Thermal coefficient of expansion. This value should be specified in units of $1 \times 10^{-7}/K$. For example, if the TCE value is $7.7 \times 10^{-6}/K$, specify a value of 77.0 when entering a value in this cell.

The variation of the refractive index with respect to temperature can be specified by entering values in the Sellmeier index vs. temperature coefficients cells. As in the Schott glass catalog, the variation of refractive index (n) with respect to temperature (T) is modeled by the Sellmeier-type formula

(7.6)

$$\frac{dn_{abs}(\lambda, T)}{dT} = \frac{n^2(\lambda, T_0) - 1}{2n(\lambda, T_0)} \left(D_0 + 2D_1\Delta T + 3D_2\Delta T^2 + \frac{E_0 + 2E_1\Delta T}{\lambda^2 - \lambda_{TK}^2} \right)$$

where T_0 is 20° C, T is the glass temperature (in degrees C), $\Delta T = T - T_0$, λ is the wavelength in μm , and D_0 , D_1 , D_2 , E_0 , E_1 , and λ_{TK} are model parameters. These model parameters are specified as d0, d1, d2, e0, e1, and lTK in the spreadsheet. Note that the ordinary dispersion data need not be entered as Sellmeier coefficients to use the Sellmeier model for index vs. temperature.

The Glass Manager menu item (see page 184) may be used to compute Sellmeier or Laurent series coefficients from wavelength and index data.

To delete a material from the private glass catalog, first select the radio button labeled Delete glass from catalog, then type the name of the glass to be deleted in the Glass name cell.

Rebuild Private Database

OSLO allows you to browse glass catalogs using a database functionality that is embedded in OSLO. In order to browse the different glass catalogs (see the "Browse Glass Database" menu item described on page 189), the glass catalog data must first be transformed into the proper database format for viewing by OSLO. Since many of the glass catalogs are static (supplied by Lambda Research and no need to be edited by the user), the only glass catalog that needs to be transformed into a database format on a regular basis is the Private glass catalog.

Note that if you update your Private glass catalog (see page 186), the changes you make will not appear when viewing the Private glass database (see the "Browse Glass Database" menu item on page 189), until you rebuild the Private glass database using **Rebuild Private Database**.

Browse Glass Database

Schott
Schott 2000
Schott RadHard
Ohara
Hoya
Corning
Sumita
Hikari
Pilkington
Misc
Obsolete
Shared
Private

Clicking on any of the catalog names in the Browse Glass Database submenu item will bring up a list of all the glasses in the named catalog, along with their technical specifications in the OSLO database format.

For a description of the different glass catalogs, see “Glass Catalogs” on page 184.

For a general description of the how to work with the database functionality in OSLO, see the section “Lens Database” on page 161.

When browsing the glass databases, the technical data for the glasses are listed column by column, with the different glasses differentiated by rows.

schott_2000.glc Glass Catalog

Menu	GlassName	Index(d)	Vnbr(dFC)	Density	dn/dT	TCE	Xmittan	Cost	Hardne	Chem
1	BASF51	1.723728	38.11	4.31	8.4	54	0.000	0.0	540	24513
2	F2	1.620040	36.37	3.61	2.8	82	0.000	0.0	420	10121
3	F4	1.616592	36.63	3.58	2.7	83	0.000	0.0	420	10122
4	F5	1.603420	38.03	3.47	3.0	80	0.000	0.0	450	10123
5	K10	1.501371	56.41	2.52	2.7	65	0.000	0.0	470	10111
6	K7	1.511121	60.41	2.53	0.1	84	0.000	0.0	520	30212
7	KZFSN4	1.613401	44.29	3.20	3.9	45	0.000	0.0	450	32544
8	KZFSN5	1.654117	39.63	3.46	4.2	45	0.000	0.0	460	32544

The following table explains the contents of the glass database columns. Each datum is a floating-point number, except where otherwise noted):

Column	Description
GlassName	Glass name (string; padded with spaces to 10 characters)
Index (d)	Index at d wavelength
Vnbr(dFC)	Abbé number (d, F, and C wavelengths)
Density	Density in gm./cm. ³ (zero if not available)
dn/dT	dn/dT at 20° C, e wavelength, in units of 10 ⁻⁶ /°K (zero if not available)
TCE	Thermal expansion coefficient in units of 10 ⁻⁷ /°K (zero if not available)
Xmittan	Bulk transmittance (25 mm. thickness, 400 nm. wavelength) (listed as 1.0 if not available)
Cost	Cost (relative to BK7 for Schott, relative to BSL7 for Ohara, relative to BSC7 for Hoya, zero if not available)

Hardne	Knoop hardness (zero if not available)
ChemCode	Chemical code (integer) - first digits of CR, FR, SR, AR, and PR (zero if not available)
Bub	Bubble group (string) (“?” if not available)
Avail	Availability code (one character) (‘?’ if not available)
DispM	Dispersion model code (integer): 1 = Laurent series (old Schott) 2 = Sellmeier series (new Schott) 3 = Conrady
NbrDspc	# coefficients in dispersion model (integer) Laurent = 6 Sellmeier = 6 Conrady = 3
Dspcoef0	Laurent = A_0 Sellmeier = b_0 Conrady = a
Dspcoef1	Laurent = A_1 Sellmeier = b_1 Conrady = b
Dspcoef2	Laurent = A_2 Sellmeier = b_2 Conrady = c
Dspcoef3	Laurent = A_3 Sellmeier = c_0
Dspcoef4	Laurent = A_4 Sellmeier = c_1
Dspcoef5	Laurent = A_5 Sellmeier = c_2
NvsT	index vs. temperature model code (integer) 0 = none 1 = Sellmeier 2 = single dn/dT value (not currently supported by OSLO)
NbrTmpe	# coefficients in n vs. T model (integer) none = 0 Sellmeier = 6 single dn/dT = 1
Tmpecoef0	Sellmeier = d_0 single dn/dT = dn/dT value
Tmpecoef1	Sellmeier = d_1
Tmpecoef2	Sellmeier = d_2
Tmpecoef3	Sellmeier = e_0
Tmpecoef4	Sellmeier = e_1
Tmpecoef5	Sellmeier = λ_{TK}
Xm	transmittance model code (integer) 0 = none 1 = 5 mm transmittance
Nbr	# coefficients in transmittance model (integer) model 1 = number of wavelength transmittance value pairs

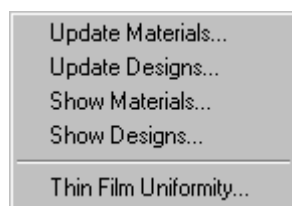
Wv0...Wv49	Wavelength: 5 mm transmittance value data pairs
Xmt0... Xmt49	Transmittance: 5 mm transmittance value data pairs

You can sort the database by any item. To sort glasses by index, click on the “Index” column label. Select “Sort Down” on the Pop-up menu: all the glasses now appear by increasing index value. You can sort on any other parameters in a similar way.

As you probably already noticed, clicking on any row will plot a glass map where the current glass is highlighted in red.

Note: if you find this automation annoying, you can disable it by clicking on the “DB” button, and click on “Disable Callbacks”.

Coatings Prm



OSLO Premium has thin film commands for creating and using thin-film multilayer coatings. There are three steps to using multilayer coatings on lens surfaces:

1. Define the dispersive properties of the coating materials you want to use. The materials can be updated using **Update Materials** and displayed in the current text window using **Show Materials**. This information is stored in a material database that is kept with your OSLO installation.
2. Construct the design of the multilayer stacks by providing the thickness and material for each layer of the coatings. The coating designs can be updated using **Update Designs** and displayed in the current text window using **Show Designs**. This information is stored in a coating design database that is kept with your OSLO installation.
3. Apply coating designs to individual surfaces. Once added, the polarization ray trace includes the effects of the multilayer coatings. This is done in the individual surfaces using the options button in the SPECIAL column of the Surface Data Spreadsheet (see the section “Multilayer Coating (M)” on page 105). This information is kept with the optical prescription of model and is stored in the appropriate lens file.

Note that if non-dispersive materials are used (where the complex index of refraction is constant for all wavelengths), it is possible to skip the material definition step because there is an option in the definition of the coating designs where you can define the properties of the coating layers directly.

Update Materials

Defining a thin film material is performed by choosing this menu item to open the Multilayer Material Spreadsheet editor (**mms**).

Save material		Retrieve material		Clear data		Delete material	
Material name: INCONEL		Note: Complex index = $n - ik$					
Nbr	Wavelength	Index (n)	Extinct Coef (k)				
1	0.200000	1.560000	2.120000				
2	0.280000	1.569300	2.134600				
3	0.290000	1.573700	2.161500				
4	0.300000	1.573300	2.199000				
5	0.310000	1.570500	2.244700				
6	0.320000	1.570700	2.295300				

Each material is defined by a series of wavelength-index pairs. Each row in the spreadsheet corresponds to a wavelength. The standard spreadsheet row operations (insert, cut, copy, paste, etc.) are available. The wavelength (in microns), and the real and imaginary parts of the index are entered in the cells. Note that the index is given by $n - ik$, so the extinction coefficient values should be entered as positive numbers. It is suggested that the index data be entered from short-to-long wavelengths, since it will be stored this way. (The order is not mandatory, as the program will sort the data before saving it.)

If a calculation at a wavelength other than the entered values must be performed, OSLO uses quadratic interpolation using the closest three data values. For wavelengths outside the range of the data points, the nearer end point value is used.

Save the current material and close the spreadsheet by clicking OK (green check mark). OSLO will then prompt you to save the current material.

After defining materials, you can edit them or view them by going to the Multilayer Material Spreadsheet editor and clicking the "Retrieve material" button which pops-up a list of previously defined materials. You can also display a list of materials or details about a particular material using the **Show Materials** menu item (see page 194).

Update Designs

Constructing the design of the multilayer stack is performed by choosing this menu item to open the Multilayer Spreadsheet editor (**mse**).

Layer	Thickness	Catalog	Material	Direct	Index (n)	Extinct Coef (k)
1	0.239250	<input type="radio"/>		<input checked="" type="radio"/>	2.350000	0.000000
2	0.248530	<input type="radio"/>		<input checked="" type="radio"/>	1.450000	0.000000
3	0.223520	<input type="radio"/>		<input checked="" type="radio"/>	2.350000	0.000000
4	0.196410	<input type="radio"/>		<input checked="" type="radio"/>	1.450000	0.000000
5	0.121860	<input type="radio"/>		<input checked="" type="radio"/>	2.350000	0.000000
6	0.147060	<input type="radio"/>		<input checked="" type="radio"/>	1.450000	0.000000
7	0.120780	<input type="radio"/>		<input checked="" type="radio"/>	2.350000	0.000000

Each layer of the stack is represented by a row in the spreadsheet. The standard spreadsheet row operations (insert, cut, copy, paste, etc.) are available. The layer thickness may be specified in μm of either optical thickness or physical thickness, in quarter-wave optical thickness (default), or in full-wave optical thickness.

If we denote the physical thickness in μm of a layer by t_{phys} , the optical thickness t_{opt} is given by

$$t_{opt} = n_0 t_{phys} \quad (7.7)$$

where n_0 is the real part of the refractive index for the material at the reference wavelength λ_0 . The full-wave optical thickness is just the optical thickness measured as a multiple of the reference wavelength λ_0

$$t_{FWOT} = \frac{t_{opt}}{\lambda_0} = \frac{n_0 t_{phys}}{\lambda_0} \quad (7.8)$$

and the quarter-wave optical thickness t_{QWOT} is given by

$$t_{QWOT} = \frac{t_{opt}}{\lambda_0/4} = \frac{t_{FWOT}}{1/4} = \frac{4}{\lambda_0} t_{opt} = \frac{4n_0 t_{phys}}{\lambda_0} \quad (7.9)$$

Thus, if a layer is a quarter-wave-optical-thickness layer, the thickness should be entered as “1.0” if quarter-wave optical thickness is being used and as “0.25” if full-wave optical thickness specification is being used. If the material is dispersive, click the “Catalog” button and enter the name of the material. Click the “material” cell to display a pop-up list of available materials. If the material is non-dispersive, click the “Direct” button and enter the values of n and k to be used for all wavelengths.

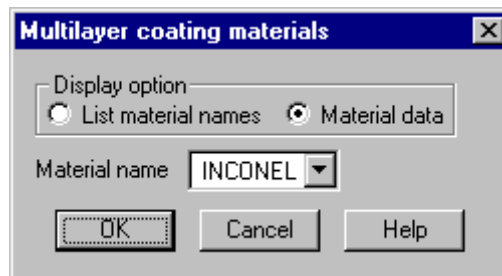
For most coatings, except mirrors (as described in the next paragraph), the ordering of the layers may either be from “incident” medium to “substrate” (default) or from “substrate” to “incident” medium. In either case, the incident and substrate media are not entered directly in the multilayer. When the coating is used in a lens, the incident and substrate media properties will be determined by the glasses on either side of the surface. If the coating is attached to an air-glass or glass-glass interface in a lens, the order of the layers is as entered. If the coating is attached to a glass-air interface, the order will be reversed. In this way, a coating used on both sides of a lens can be defined once.

A reflective metal film should be entered as the last layer for incident-to-substrate ordering or first layer for substrate-to-incident ordering of the stack. The thickness of the layer should be made sufficient to completely absorb the “transmitted” wave. (An optical thickness of 10 to 20 is generally sufficient for metals like aluminum.) With this thickness, the index of the “real” substrate is irrelevant, since no light gets to it. Alternatively, if the thickness of the last layer for incident-to-substrate ordering or first layer for substrate-to-incident ordering is greater than or equal to 1.0×10^8 , then the properties of that material will be used as the substrate for the reflectance/transmittance calculations. Thus, to use a multilayer stack on a dielectric substrate as a mirror, designate the “glass” for the surface, as usual, as REFLECT, but enter an “infinite” thickness of the substrate in the definition of the coating.

After defining multilayer designs, you can edit them or view them by going to the Multilayer Spreadsheet editor and clicking the “Retrieve multilayer” button which pops-up a list of previously defined multilayer designs. You can also display a list of multilayer designs or details about a particular multilayer design using the **Show Designs** menu item (see page 195).

Show Materials

This menu item issues the Multilayer Material Data (**mmd**) command which displays multilayer material data in the current text window.



You have the option of

1. Listing the multilayer materials that have already been defined, or

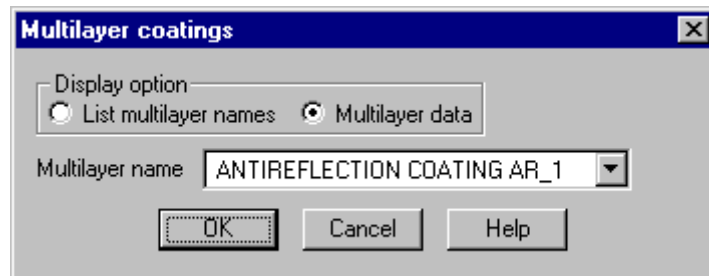


2. Listing detailed material data about a particular material.

*FILM MATERIAL DATA - INCONEL			
NBR	WAVELENGTH	INDEX (n)	EXTINCT COEF (k)
1	0.200000	1.560000	2.120000
2	0.280000	1.569300	2.134600
3	0.290000	1.573700	2.161500
4	0.300000	1.573300	2.199000
5	0.310000	1.570500	2.244700
6	0.320000	1.570700	2.295300
7	0.330000	1.569300	2.353800
8	0.340000	1.572900	2.409100
9	0.350000	1.577000	2.497000
10	0.360000	1.583600	2.562000
11	0.370000	1.592300	2.630200

Show Designs

This menu item issues the Multilayer Data (**mld**) command which displays multilayer design data in the current text window.



You have the option of

1. Listing the multilayer designs that have already been defined, or

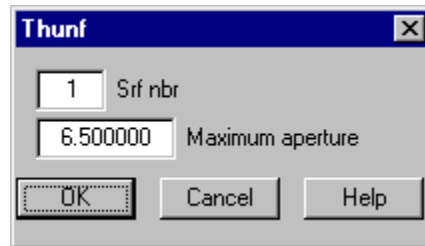
*MULTILAYER COATING CATALOG	
ANTIREFLECTION COATING AR_1	ANTIREFLECTION COATING AR_2
ANTIREFLECTION COATING AR_3	ANTIREFLECTION COATING AR_4
INCONEL BEAM SPLITTER BS_1	LONG WAVE PASS FILTER LP_1
MULTILAYER BEAM SPLITTER BS_2	NARROW BAND FILTER NB_1
NARROW BAND FILTER NB_2	REFLECTOR R_1
REFLECTOR R_2	SHORT WAVE PASS FILTER SP_1

2. Listing detailed material data about a particular multilayer design.

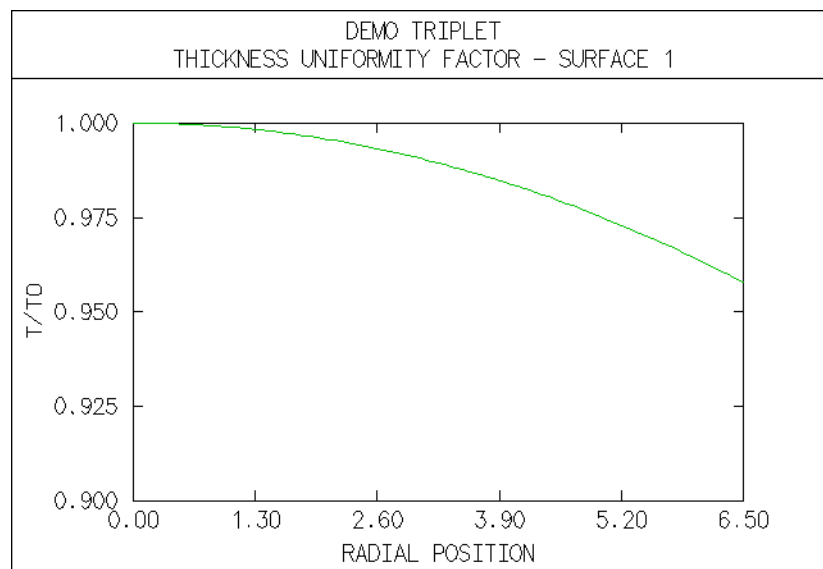
*MULTILAYER COATING DATA - MULTILAYER BEAM SPLITTER BS_2			
REFERENCE WAVELENGTH IN AIR: 0.520000 MICROMETERS			
LAYER THICKNESSES ARE MICROMETERS (OPTICAL THICKNESS)			
LAYER ORDERING: SUBSTRATE TO INCIDENT MEDIUM			
LAYER	THICKNESS	MATERIAL/INDEX	EXTINCT COEF
1	0.239250	2.350000	--
2	0.248530	1.450000	--
3	0.223520	2.350000	--
4	0.196410	1.450000	--
5	0.121860	2.350000	--
6	0.147060	1.450000	--
7	0.120780	2.350000	--

Thin Film Uniformity

Displays the thickness of a coating layer as a function of radius (**thunf**).



The result of this analysis depends on how the coating uniformity is defined when the coating is applied to a particular surface (see the section “Multilayer Coating (M)” on page 105).



Some general comments on multilayer systems¹

Optical thin films have numerous scientific, technological and commercial applications over a wavelength range that extends from the x-ray to the sub-millimeter regions. They can be used to shape the spectral response of the light transmitted and reflected by the surfaces to which they are applied. Some of the generic spectral filters that are made of multilayer coatings include antireflection coatings, neutral beam splitters, reflectors, short- and long wavelength cut-off filters, narrow band transmittance filters and polarizers. To achieve these effects, from one to many tens of layers may be required. From the point of view of a lens designer the most important filter types are antireflection coatings and reflecting coatings. However, there may be instances where it may be of interest for lens designers to perform preliminary order of magnitude calculations with some of the other types of multilayer coatings mentioned above. For this purpose a number of filter designs have been appended to this program. In order to facilitate their use it

1. The information in this and the next two sections was contributed by J. A. Dobrowolski, National Research Council of Canada.

was deemed desirable to include a very brief discussion of some theoretical and practical aspects of optical thin films.

The multilayers can be constructed of dielectric layers only, or they can consist of a combination of both metallic and dielectric layers. With all-dielectric layer systems the incident light is either transmitted or reflected—there are no absorption losses. Furthermore, because the dispersion of the refractive indices of dielectric materials is relatively small, the spectral features can be tuned over a wide range of wave-lengths simply by scaling the thicknesses of all the layers by the same amount. This is true only if the new position of the spectral feature of interest lies within the transparency range of the coating materials. Furthermore, for a given angle of incidence and plane of polarization, the transmittance T and the reflectance R of an all-dielectric layer system are independent of the direction of the incident light and they obey the relation $T + R = 1.0$.

If there are absorbing films in the multilayer system, some of the light incident on the multilayer will be absorbed. The condition $T + R + A = 1.0$ holds, where A represents the absorptance. The scaling of the layer thicknesses in order to shift the features in the spectrum is no longer so simple. First, the dispersion of the optical constants of metals are much more pronounced than that of dielectric coating materials. Second, because the extinction coefficients of metals are large, the changes in the thicknesses of the metal layers must be much smaller. Lastly, whilst the transmittance of a multilayer coating with absorbing layers is independent of the direction of the incident light, in general this is not true for the reflectance and absorptance.

The properties of both all-dielectric and metal-dielectric layer systems depend on the angle of incidence and on the state of polarization of the incident light. In general, the spectral features of multilayer coatings shift towards shorter wavelengths with an increasing angle of incidence. In addition, for angles of incidence greater than 10 or 15 degrees, a marked polarization splitting can usually be observed. For unpolarized light this results in a general broadening of the spectral features. This has important implications for lens designers. For best results, any filter in which there are sharp transitions from high to low transmittance or reflectance should be located within that part of a lens system in which the angles of incidence and the convergence angles are as small as possible.

Coating material considerations

The optical constants of thin films can depend on the actual process used for their deposition. Films produced by thermal evaporation and by conventional electron beam gun evaporation can be quite porous. Although heating of the substrate can result in a reduction of the porosity, it rarely results in completely dense films. The spectral features of coatings produced in this way frequently shift towards longer wavelengths as water vapor is adsorbed by the pores. For some applications it is possible to predict sufficiently accurately the changes that will occur on exposure of the multilayer to the atmosphere. For other, more stringent applications, it is necessary to produce very stable multilayer systems that do not age at all with time and exposure to moisture. Higher energy deposition processes, such as ion assisted electron beam gun evaporation, ion plating, ion sputtering or magnetron sputtering, yield dense coatings that meet these requirements. However, either the equipment is more expensive, or the deposition process is slower and so, as a rule, a premium has to be paid for such coatings.

Dielectrics can be classified into soft and hard coating materials. The former can be deposited by thermal evaporation at relatively low temperatures and are frequently protected by a cemented cover glass from damage due to abrasion. Hard coating materials are deposited by electron beam gun evaporation, or by sputtering. They are much harder and are quite suitable for front surface mirrors.

The optical properties of metals are even more sensitive to the deposition process than those of dielectric layers. This is especially true for partially transparent metal layers used in beam splitters and in certain advanced multilayer coatings.

The conclusion from the above is that, until a deposition process is decided upon, it is difficult to predict what optical constants should be used for the design of the multilayer. It is customary to use approximate optical constants for preliminary designs. Although it is frequently sufficient for this purpose to use non-dispersive refractive indices for the dielectric layers, the dispersion of the optical constants of metals must be taken into account. A good source of information on the optical constants of metals is the "Handbook of Optical Materials" vols. I, II edited by Palik.^{2,30} Any systems designed in this way will require only slight modifications of the thicknesses of the layers to allow for the discrepancy between calculated and experimental optical constants.

Some sample multilayer systems

Some of the multilayer systems presented below are based on multiples of quarter wave layers which are easy to monitor by optical means. These systems are best represented by a notation in which H and L correspond to high and low refractive index layers of quarter wave optical thickness, respectively. Thus, for example, $(HL)^2H$ is the same as HLHLH which represents a five layer quarter wave stack. $(HL)^3(LH)^3$ is the same as HLHLH2LHLHLH which represents an eleven layer narrow band filter in which the central layer is a half wave layer of low refractive index, etc.

Other systems consist of layers with thicknesses that depart from quarter wave thicknesses significantly. The additional degrees of freedom available in such refined systems can be used to optimize the performance of the multilayer. The thicknesses of such layers are frequently monitored using quartz crystal monitors.

The all-dielectric multilayer systems listed in the Table below, with the exception of two systems, are constructed out of two coating materials only with non-dispersive refractive indices 1.45 and 2.35. These values are not too far removed from the refractive indices of the soft coating material pair MgF_2 and ZnS or from the hard coating material pair SiO_2 and Nb_2O_5 . The optical constants of aluminum were taken from Palik. Inconel alloy constants were measured at NRCC. The substrate and incident medium materials in all systems are BK7 glass or air. M in system 2 stands for a quarter wave layer of medium refractive index of 1.7.

2. E.D. Palik, *Handbook of Optical Constants of Solids I* (Academic Press Inc., Orlando, 1985).

3. ⁰ E.D. Palik, *Handbook of Optical Constants of Solids II* (Academic Press Inc., Boston, 1991).

No.	Name	Type	Layers	Figs.	Description
1	AR_1	single layer MgF ₂ AR coating	1	1A	glass/L/air
2	AR_2	quarter-half-quarter AR coating	3	1B	glass/M2HL/air
3	AR_3	narrow band AR coating	2	1C	glass/optimized/air
4	AR_4	wide band AR coating	7	1D	glass/optimized/air
5	R_1	quarter wave stack reflector	13	1E	glass/(HL) ⁶ H/air
6	R_2	opaque Al layer reflector	1	1F	opaque Al /air
7	SP_1	short wavelength pass filter	17	2A	optimized, between glass
8	LP_1	long wavelength pass filter	17	2B	optimized, between glass
9	BS_1	Inconel layer beam splitter	1	2C	0.0125 μ m of Ag cemented between two 45° prisms
10	BS_2	multilayer beam splitter	7	2D	optimized, between two 45° prisms of BK7 glass
11	NB_1	narrow band filter (1 cavity)	15	2E	glass/(HL) ⁴ (LH) ⁴ /glass
12	NB_2	narrow band filter (2 cavities)	31	2F	glass/optimized/glass

The calculated performances of these filters are shown in Figures 1 and 2. A more detailed explanation of the theory of optical thin films will be found in the excellent book by Macleod⁴¹. For more information on classical and less usual applications of optical thin film coatings the interested reader is referred to references^{52,63}.

Show Surface Data

Entering and editing your lens model construction data (i.e. curvature, thickness, aperture radius, ...etc.) is performed using the Surface Data Spreadsheet (see Chapter 3, "The Surface Data Spreadsheet"). Sometimes, it is useful to see a complete listing of construction data summarized into different data groups.

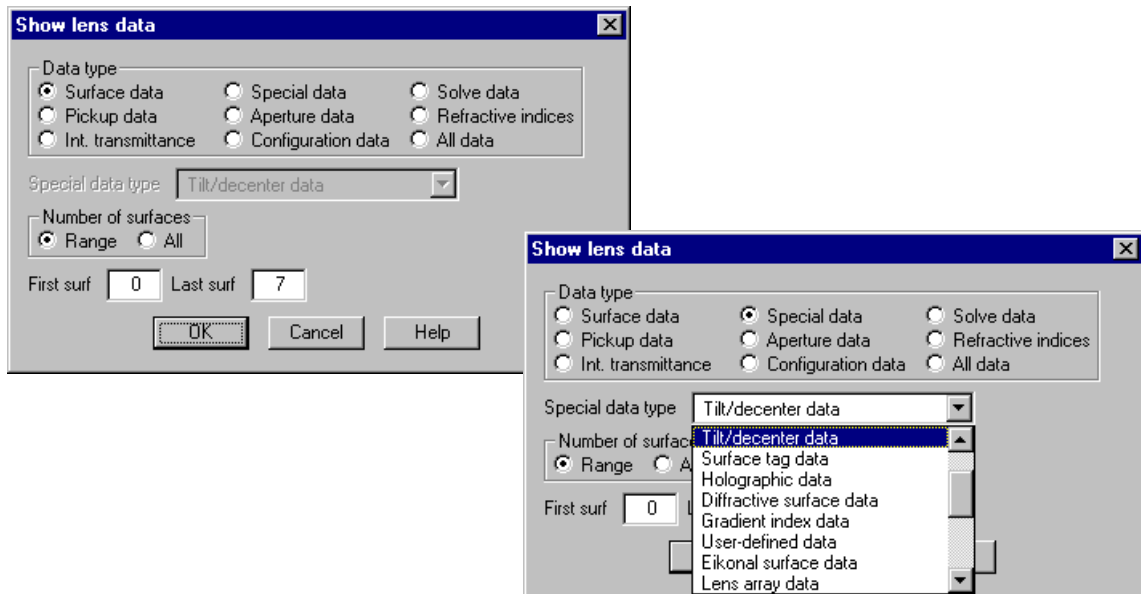
This menu item opens the View Surface Data (**view_surface_data**) dialog which allows you to choose various construction data to show in the current text window.

4. ¹ H.A. Macleod, *Thin Film Optical Filters* (McGraw Hill, New York, 1986).

5. ² J.A. Dobrowolski, "Optical Properties of Films and Coatings," in *Handbook of Optics* (eds. E.i.C. M. Bass) (McGraw-Hill, New York, 1995), pp. 42.1-42-130

6. ³ J.A. Dobrowolski, "Usual and Unusual Applications of Optical Thin Films-An Introduction," in *Thin Films for Optical Coatings* (eds. R.F. Hummel and K.H. Guenther) (CRC Press, Inc., Boca Raton, Florida, 1995), pp. 5-35.

The dialog data is accepted by clicking the OK button or dismissed by clicking the CANCEL button.



Data for the entire optical system will be included in the output if the **Number of surfaces = All** radio button is selected. Output for a subsection of the system is displayed by selecting the **Range** radio button. When the Range button is selected, the first and last surfaces to included in the output are specified by entering data into the **First Surf** and **Last Surf** fields

Data Types

Surface data - Surface data output includes the radius of curvature, thickness, aperture radius, glass or material name, a special data indicator, and a note about the surface. To select surface data from the lens data spreadsheet, click the Surface data button.

DEMO TRIPLET

SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPE	NOTE
0	--	1.0000e+20	3.6397e+19	AIR		
1	21.250000	2.000000	6.500000 K	SK16	C	
2	-158.650000	6.000000	6.500000 P	AIR		
3	-20.250000	1.000000	5.000000	F4	C	
4	19.300000	6.000000	5.000000 A	AIR		
5	141.250000	2.000000	6.500000	SK16	C	
6	-17.285000	42.950000	6.500000 K	AIR		
7	--	--	18.170326 S			

The output displays the lens title and seven columns of data. The column headings and displayed data are:

SRF - displays the surface numbers.

RADIUS - displays the radius of curvature.

THICKNESS - displays the thickness from one surface to the next surface.

APERTURE RADIUS - displays the aperture radius.

GLASS - displays glass or material names.

SPE - displays an asterisk if special data is defined on the surface.

NOTE - displays a surface note. The note will be displayed on the next line, if necessary.

The radius, thickness, glass, and aperture radius columns may have informational tags that follow the data. The informational tags displayed with the surface output have the following meanings:

A - appears only in the aperture radius column indicating the aperture stop surface,

C - appears only in the glass column indicating a catalog glass,

F - indicates that the value is fixed,

K - appears only in the aperture radius column indicating aperture checking,

M - appears only in the glass column indicating a model glass,

P - indicates that the value is picked up from a preceding surface,

R - appears only in the aperture radius column indicating the reference surface, if the reference surface differs from the aperture stop surface,

S - indicates that the value is the result of a solve,

T - appears only in the radius column indicating that the radius has been matched to an available test plate,

V - indicates that the item is free to vary during optimization, and

X - appears only in the aperture radius column indicating that special aperture data is defined for the surface.

Detailed aperture, glass refractive index, pickup, and solve data is output by selecting the radio button for the data type from the spreadsheet.

Special data - Special data on a surface may include any of the special data types available in OSLO. To select special data from the lens data spreadsheet, click the Special Data button to activate the special data type selections and then click the radio button for the desired special data type of output. All of the special data type information is automatically included in the output if the All Data radio button is selected. The output displays the surface number followed by the special data. The meanings of the special data items are detailed in the Special Data section of the Update chapter.

Solve data - Solve data on a surface may include a curvature solve, a thickness solve, and a solve for the aperture radius. To select solve data from the lens data spreadsheet, click the Solve Data button.

The output displays the surface number followed by the solve type and data value. The solve types are:

AL is a curvature solve so that the paraxial axial ray satisfies the aplanatic condition.

ALC is a curvature solve so that the paraxial chief ray satisfies the aplanatic condition.

EC is an edge contact solve for the specified aperture radius.

PI is a curvature solve for the specified paraxial axial ray angle of incidence.

PIC is a curvature solve for the specified paraxial chief ray angle of incidence.

PU is a curvature solve for the specified paraxial axial ray angle.

PUC is a curvature solve for the specified paraxial chief ray angle.

PY is a thickness solve for the specified paraxial axial ray height.

PYC is a thickness solve for the specified paraxial chief ray height.

Aperture radii determined by solves are not included in the solves output because the solve value at a surface is fixed at the sum of the magnitudes of the paraxial axial ray height and the paraxial chief ray height at the surface. Aperture radius solves are displayed with the aperture data.

Pickup data - Pickup data on a surface may include a curvature, thickness or the tilt and decenter data from a preceding surface, an axial length between a range of surfaces, or the negative value of these items. The glass or material from a preceding surface can also be picked up at a surface. To select pickup data from the lens data spreadsheet, click the Pickup Data button.

The output displays the surface number followed by the pickup type, the number of the surface(s) on which the pickup data is specified, and an increment value. The increment value is the difference between the value of the datum on the surface and its value on the "picked up" surface. The available types of pickups are indicated by:

AP - pickup aperture radius.

CV - pickup curvature. This also picks up the conic constant, toric radius, and any aspheric or spline surface coefficients, if they are defined on the picked up surface.

CVM - pickup the inverse value of the curvature. This also picks up the conic constant, inverse toric radius, and inverse aspheric or spline surface coefficients, if they are defined on the picked up surface.

DFC - pickup of CGH diffractive surface coefficients.

LN - pickup the length between surfaces displayed.

LNM - pickup the inverse value of the length between surfaces displayed.

GLA - pickup glass (material) data.

TH - pickup thickness.

THM - pickup the inverse value of the thickness.

TD - pickup of tilt and decenter data. All the special data associated with the tilt and decenter are picked up.

TDM - pickup minus of tilt and decenter data. All the special data associated with the tilt and decenter are picked up.

Aperture data - The aperture radius on a surface can be a direct specification, the result of a pickup, or a solve. Ray tracing may or may not check for vignetting at the aperture. To select aperture data from the lens data spreadsheet, click the Aperture Data button.

The output displays the surface number followed the type of aperture specification, the value of the radius and the status of aperture checking. The data is displayed under the following headings:

SRF - displays the surface number.

TYPE - displays the type of aperture specification.

APERTURE RADIUS - displays the aperture radius.

The aperture types that can be specified in OSLO are:

CMP - aperture radius computed from a solve.

PKP - an aperture value that is picked up from a preceding surface.

SPC - a direct specification of the aperture radius.

The value of an aperture radius is followed by **CHK**, if aperture checking has been designated at the surface.

If special aperture data is included in the lens data, it is also displayed with the output. Special aperture data consists of:

APN - the number of special apertures defined.

A, B - the name of the special aperture. Additional apertures (**C, D**, etc.) are available in OSLO Premium.

AX1 - the minimum x coordinate value (ellipse, rectangle).

AX2 - the maximum x coordinate value (ellipse, rectangle).

AY1 - the minimum y coordinate value (ellipse, rectangle).

AY2 - the maximum y coordinate value (ellipse, rectangle).

AVX1, AVY1 - the x and y coordinates of vertex 1 (triangle, quadrangle).

AVX2, AVY2 - the x and y coordinates of vertex 2 (triangle, quadrangle).

AVX3, AVY3 - the x and y coordinates of vertex 3 (triangle, quadrangle).

AVX4, AVY4 - the x and y coordinates of vertex 4 (quadrangle).

AAN - angle of rotation of an elliptical or rectangular special aperture relative to the y-axis.

ATP - aperture type: elliptical, rectangular, triangular, quadrangular.

AAC - an aperture action that the ray trace tests against. The available actions are a transmitting aperture, an obstruction, or a hole through which rays pass undeviated.

Wavelength data - (This item is displayed when the **All data** option is chosen)

The wavelengths of light for which the lens system is designed are fundamental to the proper design and analysis of the system. The relative importance of each wavelength can also be weighted. Select the Show menu item and click Wavelengths to display the system wavelengths and wavelength weights. The wavelengths are output in units of microns (μm). The wavelength numbers (colors) are 1, 2, 3, etc., which refer to a corresponding wavelength value in microns. The output displayed are:

WV1/WW1, WV2/WW2, WV3/WW3, etc. - displays the wavelength on the first row under the column heading. The relative weight of rays traced in the wavelength is displayed on the second row.

CURRENT - displays the number of the wavelength currently in use for ray trace evaluation, and drawing rays on lens drawings.

The refractive indices of each material are computed at wavelengths WV1, WV2, WV3, etc., as specified in the lens data.

Refractive indices - The refractive index of a surface is the index of the medium on the image side of the surface. At a reflective surface, the refractive index is the same as the immediately preceding surface. To select refractive index data from the lens data spreadsheet, click the Refractive Indices button.

The output displays the surface number followed by the glass name, the value of the refractive index at each wavelength, and the value of a dispersion ν -number. The refractive index of the image is the refractive index of the surface immediately preceding it. The output is displayed under the following column headings:

SRF - displays the surface number.

GLASS - displays the glass name.

RN1, RN2, RN3, etc. - displays the refractive index at each wavelength, WV1, WV2, WV3, etc., respectively.

VNBR - displays the dispersion ν -number. The ν -number is computed from the refractive indices of the material in wavelengths WV1, WV2, WV3, as specified in the lens data.

TCE - the thermal coefficient of expansion (α), in units of $1 \times 10^{-7} / \text{K}$

Internal transmittance - Lists the internal transmission through the media of every surface as a function of the wavelength. Do not confuse this with the actual transmission through each surface of the lens model. For the **internal transmittance** calculation, the same thickness for each material is used (5mm).

SRF - displays the surface number.

GLASS - displays the glass name.

ITN1, ITN2, ITN3, etc. - displays the internal transmittance (for a material thickness of 5mm) at each wavelength, WV1, WV2, WV3, etc., respectively.

Configuration data - In a multi-configuration system, one configuration is designated as the base system, and each of the other systems (configurations) is defined by data that differ from the base system. The different configurations are then able to be described by listing their differences from the base system. A summary of data that are different in different configurations is **Configuration data**.

The configuration data displayed are:

TYPE - is the data type, such as curvature, thickness, object height, etc.

SN - is the surface number for which the item is defined. A surface number of zero indicates the object surface if the data type is surface data, or it indicates the system if the data is an operating condition.

CFG - is the configuration number for which the item is defined.

QUALF - is a data qualifier, such as wavelength number for refractive index data, if required for the configuration item; otherwise, a zero is displayed.

VALUE - displays the value for the item.

A complete list of lens data items that can be specified as configuration data is given in the section "Zoom (Configuration data)" on page 20.

Wavelength data - The wavelengths of light for which the lens system is designed are fundamental to the proper design and analysis of the system. The relative importance of each wavelength can also be weighted. Select the Show menu item and click Wavelengths to display the system wavelengths and wavelength weights. The wavelengths are output in units of microns (μm). The wavelength numbers (colors) are 1, 2, 3, etc., which refer to a corresponding wavelength value in microns. The output displayed are:

WV1/WW1, WV2/WW2, WV3/WW3, etc. - displays the wavelength on the first row under the column heading. The relative weight of rays traced in the wavelength is displayed on the second row.

CURRENT - displays the number of the wavelength currently in use for ray trace evaluation, and drawing rays on lens drawings.

The refractive indices of each material are computed at wavelengths WV1, WV2, WV3, etc., as specified in the lens data.

All data - Surface data for all the previous categories will be displayed in a single command if the All data radio button is selected. In addition, the wavelength data is displayed in the output.

Show Tolerance Data



The tolerance data of a lens system describes the lens manufacturing requirements. In OSLO, tolerance data is separated into three classes - each having its own data editing spreadsheet:

- Surface Tolerance
- Component (Air-to-Air) Tolerances
- Group (User-Defined) Tolerances

The Show Tolerance Data menu option simply lists the tolerance data from the three classes. To find out more details about tolerancing and tolerancing data, see Chapter 10, 'The Tolerance Menu'.

Surface

This menu option lists the system Surface Tolerance Data to the current text window. The tolerance output includes the design values for the radius of curvature, thickness, glass name, and the following tolerance items:

RADIUS TOL - is the radius of curvature tolerance, specified in lens system units. Note that the radius of curvature tolerance is, according to the ISO 10110, Part 1 Subclause 4.6.1, a tolerance that indicates "the range within which the actual surface must be contained."

FRINGES SPH - is the surface form tolerance on sagitta error, specified in units of fringe spacing, where one fringe spacing is equal to one-half of the specified light wavelength.

FRINGES IRR - is the surface form tolerance on irregularity, specified in units of fringe spacing, where one fringe spacing is equal to one-half of the specified light wavelength.

THICKNESS TOL - is the thickness tolerance, specified in lens system units.

INDEX TOL - is the refractive index tolerance.

DECEN TOL - is the decentering tolerance, specified in lens units.

TILT TOL - is the tilt tolerance, specified in decimal degrees.

FRINGE WAVELENGTH - is the wavelength used for the specification of the fringe tolerance items. The default wavelength value is 0.54607 μm .

The tolerance data is also displayed on element drawings.

**SURFACE TOLERANCES

DEMO TRIPLET

RADIUS		RD TOL	FRINGES		THICKNESS	TH TOL	RN TOL		DECEN	TILT				
SRF	CON	CNST	CC	TOL	PWR	IRR	TLC	TOL	DZ	TOL	GLASS	V TOL	Y/X	A/B
1	21.2500	--	10.00	2.00	2.0000	0.2000	SK16	0.0010	--	0.3333				
	--	--			--	--		0.8000	--	0.3333				
2	-158.6500	--	10.00	2.00	6.0000	0.2000	AIR	--	--	0.3333				
	--	--			--	--		--	--	0.3333				
3	-20.2500	--	5.00	1.00	1.0000	0.1000	F4	0.0010	--	0.5000				
	--	--			--	--		0.8000	--	0.5000				
4	19.3000	--	5.00	1.00	6.0000	0.1000	AIR	--	--	0.5000				
	--	--			--	--		--	--	0.5000				
5	141.2500	--	10.00	2.00	2.0000	0.2000	SK16	0.0010	--	0.3333				
	--	--			--	--		0.8000	--	0.3333				
6	-17.2850	--	10.00	2.00	42.9500	0.2000	AIR	--	--	0.3333				
	--	--			--	--		--	--	0.3333				
7	--	--	--	--	--	--			--	--				
	--	--			--	--			--	--				

FRINGE WAVELENGTH: 0.546070

Fringes measured over clear aperture of surface unless indicated.

Tilt tolerances are specified in degrees.

Component

This menu option lists the system Component Tolerance Data to the current text window.

For the tolerancing calculations, OSLO determines that a “*component*” (or “*air-to-air component*”) is defined to be two or more real surfaces and the intervening media (glasses), bounded on both sides by air. For example, a single element lens, a cemented doublet, and a cemented triplet are all air-to-air components, but an air-spaced doublet is considered to be two components. See the section “Component (Air-to-Air) Tolerance Data” on page 404 for more information about Component tolerance data.

COMPONENT TOLERANCES*DEMO TRIPLET**

SRF	DECENTRATION		CLEAR APERTURE TILT		CENTER OF CURV TILT	
	DCY	DCX	ALPHA	BETA	ALPHA	BETA
1	0.110000	0.110000	--	--	0.290000	0.290000
2			--	--	--	--
3	0.090000	0.090000	--	--	0.240000	0.240000
4			--	--	--	--
5	0.090000	0.090000	--	--	0.040000	0.040000
6			--	--	--	--

Tilt tolerances are specified in degrees.

Prm Group

This menu option lists the system Group Tolerance Data to the current text window.

OSLO allows you to create user defined groups of surfaces that you can treat as being related during the tolerancing process. To find out more about Group Tolerance Data, see the section "Group (User-Defined) Tolerance Data" on page 407.

Show Optimization Data

Variables
Field Point/Ray Set
Plot Field Points/Ray Set Data
Spot Diagram Set
Error Function Operands

The principal contributions of modern computer technology to optical design have been related to optimization. Lens performance requirements are defined in terms of a computable error function. In OSLO, the error function is built from "operands." The operands depend on the constructional items, operating conditions of the lens and, if necessary, an associated set of rays and spot diagrams. A subset of the lens construction items, such as curvatures, thicknesses, etc., can be designated as variables and will then be permitted to vary during optimization.

Note that the items in the Show Optimization Data submenu are only available if the appropriate data items already exist. The ray set, operands, and variables are entered and updated separately using items in the "Optimize" menu (see Chapter 9, 'The Optimize Menu').

Variables

Variables are the items that are adjusted independently to establish the performance of the system. In order to optimize a system, you must have both operands and variables, and the operands must be computable functions of the variables. To display the variables data, select the Show menu item, select Optimization Data and then, from the pull-right menu, click Variables.

***VARIABLES**

VB	SN	CF	TYP	MIN	MAX	DAMPING	INCR	VALUE
V 1	2	-	TH	0.100000	1.0000e+04	1.000000	0.000625	6.000000
V 2	4	-	TH	0.100000	1.0000e+04	1.000000	0.000625	6.000000
V 3	6	-	TH	0.100000	1.0000e+04	1.000000	0.000625	42.950000
V 4	1	-	CV	--	--	1.000000	1.6000e-05	0.047059
V 5	2	-	CV	--	--	1.000000	1.6000e-05	-0.006303
V 6	3	-	CV	--	--	1.000000	1.6000e-05	-0.049383
V 7	4	-	CV	--	--	1.000000	1.6000e-05	0.051813
V 8	5	-	CV	--	--	1.000000	1.6000e-05	0.007080
V 9	6	-	CV	--	--	1.000000	1.6000e-05	-0.057854

The variables are described by:

VB - displays the variable number.

SN - displays the surface number.

CF - displays the configuration number. This is the configuration in which this variable takes on the specified value. A configuration number of zero (-) indicates that the variable takes on the specified value in all configurations, i.e., the variable is global.

TYP - displays the variable type, such as curvature, thickness, etc. A list of all possible variable types is given in the Variables section of the Optimize chapter.

MIN, MAX - display the minimum value and maximum value, respectively, that the variable is allowed to assume during optimization.

DAMPING - displays the damping for the variable. Increasing the damping tends to restrict the change in the variable during optimization.

INCR - displays the change applied to the current value of the variable during optimization to estimate the derivatives of the operands with respect to that variable.

VALUE - displays the current value of the variable.

If there is only one configuration of the system being optimized, then all variables are automatically global. If there are multiple configurations being optimized, then a variable that is tagged with a configuration number will be varied only in that configuration. Configuration data must be specified for each such variable.

If the variable exceeds one of its minimum or maximum bounds, a penalty is added to the merit function. The optimization operating condition **opt_boundarywgt** (**opbw**) controls the weight given to boundary value violations in the error function. Increasing the value of this operating condition places more emphasis on correcting boundary condition violations. If both variable bounds are zero, the variable is allowed to assume any value during optimization without penalty.

Std Prm Field Point / Ray Set

The ray set defines the fractional pupil coordinates of the rays that will be traced as part of the optimization process. A ray set consists of two parts, a set of field points and a set of rays. To display ray set data, select the Show menu item, select Optimization Data and then, from the pull-right menu, click Ray Set.

***RAYSET**

FPT	FBY/FY1	FBX/FY2	FBZ/FX1	YRF/FX2	XRF/WGT
F 1	--	--	--	--	--
	-1.000000	1.000000	-1.000000	1.000000	0.250000
F 2	0.707107	--	--	--	--
	-1.000000	1.000000	-1.000000	1.000000	1.000000
F 3	1.000000	--	--	--	--
	-1.000000	1.000000	-1.000000	1.000000	0.250000
RAY	TYPE	FY	FX	WGT	
R 1	Ordinary	--	--	0.016667	
R 2	Ordinary	0.342742	--	0.094619	
R 3	Ordinary	0.597816	--	0.138715	
R 4	Ordinary	0.801633	--	0.138715	
R 5	Ordinary	0.939429	--	0.094619	
R 6	Ordinary	1.000000	--	0.016667	

Field points - The field points set defines the fractional coordinates of points on the object surface from which the rays that will be traced as part of the optimization process emanate. The ten pieces of data that describe a field point are:

FPT - displays the field point number.

FBY, FBX, FBZ - are the fractional coordinates (relative to the object height) of the object point in the y, x, and z directions, respectively. The data is output in the first row for a field point.

YRF, XRF - are the fractional y and x intercepts (relative to the aperture radius of the reference surface), respectively, of the reference ray on the reference surface. The data is output in the first row for a field point.

FY1, FY2 - display the minimum and maximum y-axis pupil bounds, respectively. The data is output in the second row for a field point.

FX1, FX2 - display the minimum and maximum x-axis pupil bounds, respectively. The data is output in the second row for a field point.

WGT - displays the weight given to the field point when constructing the error function. The data is output in the second row for a field point.

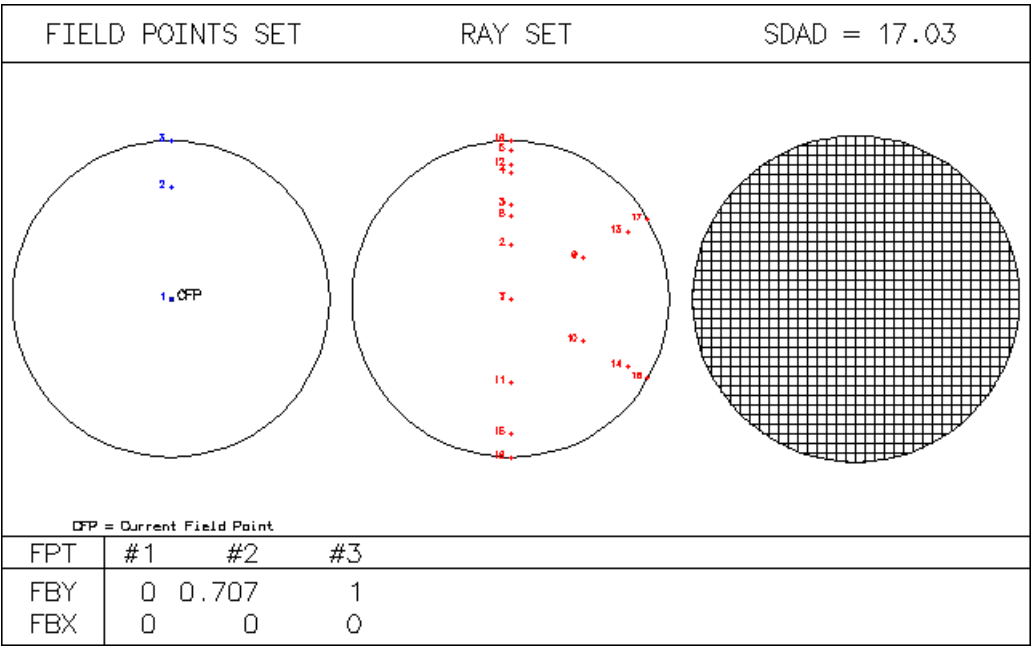
The four quantities, FY1, FY2, FX1, and FX2, are fractional coordinates, relative to the pupil radius. Together, these numbers define an effective elliptical pupil for the field point. Thus, the default values of FY1 = -1, FY2 = +1, FX1 = -1, FX2 = +1 define an unvignetted circular pupil.

Rays - The coordinates of the rays defined in the ray set are displayed starting from the RAY heading. The ray can be one of two types. A *reference ray* is a ray that emanates from a specified field point and passes through a specified point on the reference surface. An *ordinary ray* is a ray that emanates from a specified field point and travels in a prescribed direction. The direction is indicated by the difference between the fractional pupil coordinates of the ray and the fractional coordinates of the ray on the object surface. The operand components that are available for use from a ray depend on the type of ray. (See the discussion of operands in Chapter 5.) The data items describing a ray are:

- RAY** - displays the ray number.
- TYPE** - displays the type of ray, indicating the ray is either an ordinary ray or a reference ray.
- FY, FX** - are the fractional y and x coordinates of the ray, respectively. For reference rays, these are the fractional coordinates of the ray on the reference surface, relative to the aperture radius of the reference surface. For ordinary rays, these are the fractional pupil coordinates.
- WGT** - is the weight assigned to the ray when constructing the merit function.

Std Prm Plot Field Points/Ray Set Data

The Plot Field Points/Ray Set Data (**grst**) command plots a summary of the following items in the current graphics window:



- Field Point Set** which are used in the error function as well as in the report graphics (see the section). The location of each field point and labeled with its number in the field point set. The points are shown relative to the full field represented as a circle with a relative radius of 1. The command also lists the current field points at the bottom of the plot. Note that the field point set is not to be confused with the field point fans that are defined for drawing purposes in the Lens Drawing Conditions dialog (see the section “Lens Drawing Conditions” on page 238).
- Ray Set** which are also used in the error function (see the section “Ray Set” on page 351). The pupil location of each ray is shown and labeled with its number in the ray set. The points are shown relative to the full system aperture represented as a circle with a relative radius of 1.
- Spot Diagram Conditions** where the where the following parameters are plotted:
 - Current grid size (**sdad** - used by most function under the Evaluation menu).

- Gaussian Apodization if relevant, with a gray scale relative illumination plot. The red ellipse represents the i/e^2 equi-level line.

Prm Spot Diagram Set

The spot diagram set defines the properties of the spot diagrams that are traced for spot diagram operands (i.e., MTX, MTY, and WVF). The data items describing spot diagram sets are:

SD - displays the spot diagram number.

FPT - displays the index of an entry into the field points set. This number must be between 1 and the number of entries **fptmax** in the field points set. (The default value of 0 indicates that the spot diagram is undefined, and any such spot diagrams remaining when the spreadsheet is closed are deleted.)

APDIV - is number of aperture divisions across the pupil (grid size) for the spot diagram. A larger number here generally provides more accurate results but causes the spot diagram to take longer to calculate.

FIRST WVL - NBR WVLS - these determine the wavelengths used in computing the spot diagram. If the number of wavelengths (NBR WVLS) is 1, the spot diagram is monochromatic and is computed at the wavelength number given by FIRST WVL. If the number of wavelengths is greater than 1, the spot diagram is polychromatic and is traced at wavelengths FIRST WVL, FIRST WVL + 1, ..., FIRST WVL + NBR WVLS - 1.

Error Function Operands

The operands set is a user-constructed measure of the performance of a lens. The operands set consists of a number of operand definitions which measure various physical and optical properties of a lens system; these quantities are combined into a single figure of merit called the *error function* (or *merit function*). The operand syntax and error function description are detailed in the Operands section of the Optimize chapter. It is the task of the designer and the optimization algorithm to determine the values of the variables that minimize the error function. To display operand data, select the Show menu item, select Optimization Data and then, from the pull-right menu, click Operands.

*OPERANDS						
OP	DEFINITION	MODE	WGT	NAME	VALUE	%CNTRB
O 8	"RMS"	M	0.250000	Yrms1	0.003557	0.35
O 23	"RMS"	M	0.500000	Xrms2	0.008626	4.07
O 38	"RMS"	M	0.500000	Yrms2	0.036093	71.31
O 53	"RMS"	M	0.125000	Xrms3	0.020610	5.81
O 68	"RMS"	M	0.125000	Yrms3	0.036728	18.46
MIN ERROR:		0.024677				

The data describing the operands that comprise the error function are:

OP - displays the operand number.

DEFINITION - displays the definition of the operand.

MODE - displays the operand mode. The mode is either minimize and indicated with an M, or a constraint and indicated with a C.

WGT - displays the operand weight for minimize-mode operands.

NAME - displays a name for the operand. An operand name is optional.

VALUE - displays the computed value of the operand.

%CNTRB - displays the operand's current per cent contribution to the weighted rms error.

MIN ERROR - is the current value of the minimize-mode operands error function.

Operand definitions consist of one component or two components combined by an operator. Definitions are of the form:

A , $A+B$, $A-B$, $A*B$, A/B , $A**B$, $A>B$, $A<B$

where A and B represent operand *components*. The operators

$+$, $-$, $*$, $/$, $**$, $>$, and $<$

represent addition, subtraction, multiplication, division, exponentiation, greater than, and less than, respectively.

For example, the operand definition "SA3-SA5" represents the difference between the operand components "SA3" (which is third-order spherical aberration) and "SA5" (which is fifth-order spherical aberration). The operators "<" and ">" form *one-sided* operands that have zero value when the expressed relation is true, and have a value equal to the difference between the components when the relation is false. The operand definition "-0.005<SA3" will be inactive if SA3 is greater (algebraically) than -0.005, but it will be active with a value of -0.005-SA3 if SA3 is less than -0.005. Conversely, the operand definition "0.005>SA3" will be inactive if SA3 is less than 0.005, but it will be active with a value of 0.005-SA3 if SA3 is greater than 0.005. The result of using both of these operands in the error function simultaneously is to create an operand that ignores SA3 if it is between the values of -0.005 and 0.005.

The optimization algorithm attempts to find exact solutions for the constraints, using the technique of Lagrange multipliers to find values of the variables such that all of the constraints are equal to their target value of zero. Note that there must be at least as many variables as operands for constraint-mode optimization to be possible. Constraints are often used to maintain certain conditions of the lens (e.g., f -number, focal length) exactly. The minimize-mode operands are used to form the merit function. The optimization algorithm attempts to minimize the weighted root-mean-square value of the minimize-mode operands. Minimize-mode operands are used to measure conditions (e.g., transverse ray aberrations, optical path difference) that can not be corrected fully.

If an operand name begins with an underscore, "_", the operand is a "hidden" operand; that is, it is not listed by the **operands** command unless the All argument is specified. Note that the statistical average (**ave**) operands and operands between average operands and the statistical root-mean-square (**rms**) operands are also hidden. See the Optimize chapter for a complete explanation of optimization error functions.

Show Operating Conditions

General
System Notes
Lens Drawing
Optimization
Configuration
Partial Coherence
Polarization
Tolerance

The operating conditions of a lens system are data that describe how the lens is to be used or evaluated. Choosing these submenu items results in a summary being displayed in the current text window.

General

The general operating conditions include items that determine how aberration contributions and ray data will be displayed, the name of the designer, the physical units in which the lens data is entered, etc.

*CONDITIONS: GENERAL

Image surface:	7	Aperture stop:	4
Evaluation mode:	Focal	Reference surface:	4
Aberration mode:	Transverse	Aperture checking:	On
Number of rays in fans:	21	Designer:	OSLO
Units:	mm	OSLO Premium Edition-Rev	6.3 LMR-A
Wavefront ref sph pos:	Exit pupil	OPD reported in wavelengths:	On
Callback level:	0	Print surface group data:	Off
Compute solves in alt. configs:	Off	Zernike polynom. reference axis:	Y
Ray aiming mode:	Central refer. ray	Ext. aper. beam angle:	90.000000
Source astigmatic dist:	--	Ray aiming type:	Aplanatic
Plot ray-intercepts as H-tan U:	Off	Global ref. surf. for ray data:	1
Symmetry state:	Automatic	Evaluation z-axis:	Image srf z-axis
Use equal image space incrmnts.:	Off	Diffraction efficiency calcs.:	Off
Temperature:	20.000000	Pressure:	1.000000

The general operating conditions display the following data items:

Image surface (ims) - The last surface in the system is the image surface, regardless of whether or not an optical image is actually formed there.

Aperture stop (ast) - The surface number of the aperture stop is displayed.

Evaluation mode (afo) - The evaluation mode controls the computation and display of paraxial and ray data. The choices for evaluation mode are 1) Focal and 2) Afocal. Focal mode should be used for focal systems, i.e. an image is formed at a finite distance. Ray data for focal mode will be reported as transverse quantities. Afocal mode should be used for afocal systems, i.e., the image is infinitely distant. Ray data for afocal mode will be reported as angular quantities.

Reference surface (rfs) - The output value is the reference surface number for ray tracing, as designated in the lens surface data.

Aberration mode (amo) - The aberration mode controls how aberration surface contributions are displayed. The three choices for the aberration mode: 1) Transverse, 2) Angular, or 3) Unconverted. Transverse mode is appropriate for focal systems, since the aberrations will be expressed as lateral ray errors. Angular mode is appropriate for afocal systems, since the aberrations will be expressed as angular ray errors. Unconverted mode, which reports the direct surface contributions, can be used in either mode.

Aperture checking in raytrace (apck) - If the option is On, apertures that have been designated as checked apertures will be checked for vignetting for all rays traced by OSLO. If the option is Off, apertures will only be checked for spot diagram rays. See the description of Aperture Radius in the surface data spreadsheet for information on how to designate an aperture as a checked aperture.

Number of rays in fans (fanr) - This is the number of rays used by any evaluation routine that traces a one-dimensional fan of rays through the lens, e.g. plotting ray-intercept curves, plotting OPD curves, etc. The larger the number of rays, the more accurate the resulting evaluation since less interpolation is needed to draw smooth curves, but at the expense of a longer time needed to perform the calculation. The default number of rays is 21, which should be appropriate for most lenses.

Designer (des) - The designer is a 10 character field that can be used to identify the designer of the lens, if desired. The default designer name assigned to a new lens is controlled by the **designer** preference.

Units (uni) - Lens units are the unit of measure in which the lens is described. The units may be in millimeters (mm), centimeters (cm), inches (in), microns (μm), or meters (m).

Program - The OSLO version and revision level of the program that generated the output is displayed.

Wavefront ref sph pos (wrsp) - This operating condition controls the location of the reference sphere used for the calculation of optical path differences for focal systems. There are three possible choices for reference sphere position:

1. Exit pupil - the reference sphere is located at the real exit pupil position for each field position.
2. Infinity - the reference sphere is located at infinity.
3. Last surface - the vertex of the reference sphere is located at the intersection of the reference ray with the next-to-last surface in the lens system.

The default location of the reference sphere for a new lens is at the exit pupil.

OPD reported in wavelengths (opdw) - If the option is On, optical path differences will be measured in wavelengths. If the option is Off, optical path differences will be measured in the current lens units.

Callback level (callback) - This displays the value of the callback from the surface data spreadsheet editor. If this value is non-zero, the Autodraw feature in the surface data spreadsheet is active.

Print surface group data (gpri) - If this option is On, surface data is output for all surfaces within a group. If the option is Off, output data is suppressed for surfaces within a group.

Compute solves in configs (cslv) - If the option is On, solves will be carried out in all configurations. If the option is Off, solves will be carried out only in the base configuration, i.e. configuration 1.

Telecentric entrance pupil (tele) - For finite conjugate systems, the lens can be designated as having a telecentric (i.e., infinitely distant) entrance pupil. If this option is On, all chief rays (reference rays for new field points) will have an initial direction in object space that is parallel to the optical axis. The aperture stop and reference surface designations will be ignored, so in general, if no constraints are

applied, the chief ray will not pass through the center of the aperture stop surface. The default condition is that telecentric mode is Off and the actual entrance pupil location is used.

Wide-angle ray aiming mode (warm) - If this mode is On, a special ray trace algorithm, suitable for wide-angle lenses, is used. Instead of aiming rays at the entrance pupil, rays are iteratively traced backward from the reference surface, until they pass through the required object point. This allows lenses which have a large amount of pupil aberration, e.g. wide-angle fisheye lenses, to be evaluated accurately. In this mode, object points for infinite conjugate systems are measured in fractions of the current field angle. In this way, the paraxial properties of a lens can be calculated (paraxial optics requires a field angle less than 90 degrees), but real field angles greater than 90 degrees can be specified. For example, if the paraxial field angle is specified as 45 degrees, a fractional object height of 2 defines a real field point at a 90 degree field angle.

Aperture checking for all GRIN ray segments (grck) - If this option is On, and aperture checking in the ray trace (**apck**) is also On, all ray segments in gradient-index media will be checked for vignetting, using the aperture radius (if a checked aperture) of the surface on which the GRIN medium is defined. If this option is Off, but aperture checking in the ray trace is On, the ray will only be checked against the aperture upon refraction into the medium.

Extended-aperture ray aiming mode (xarm) - Some imaging systems, and many illumination systems, collect light over greater than a hemispherical solid angle. Extended aperture ray aiming mode may be used to handle such systems. Ordinarily, rays to be traced are specified by fractional object coordinates and fractional coordinates at the reference surface. If this option is On, extended aperture ray aiming mode replaces the reference surface coordinate specification with the specification of two object space angles. See Chapter 5 of the Optics Reference manual for a description of extended aperture mode.

Plot ray-intercepts as H -tan U (htnu) - The default operation for ray-intercept curves and OPD curves is to plot the aberration as a function of fractional entrance pupil coordinates FY and FX. If this option is On, for focal systems, the ray aberration will be plotted as a function of the difference in the tangents of the convergence angles between the pupil ray and the reference ray. This type of curve is traditionally called an " H -tan U " curve, since the ray height is denoted by H and the convergence angle by U . In addition to plotting H -tan U curves, OSLO will plot the OPD as a function of the sine of the angle between the pupil ray and the reference ray. The ray's exit pupil position is related to $\sin U$, so this plot is indicative of the relative size of the exit pupil in x and y .

XARM beam angle (xaba) - This is the half-angle, in degrees, of the cone into which rays are launched if extended-aperture ray aiming mode (**xarm**) is being used.

Source astigmatic distance (sasd) - A non-zero value for this operating condition defines the source to be astigmatic, i.e. rays appear to emanate from different source points in the xz and yz azimuths. The source astigmatic distance is the longitudinal separation of the xz and yz apparent source points. If the distance is positive the xz source point is closer to surface 1 than the yz source point. An astigmatic source may only be used with on-axis object points. To simulate off-axis performance, tilt the optical system at surface 1.

Ray aiming mode (epxy) - If the status of this operating condition is aplanatic, rays are aimed at the entrance pupil in fractional pupil coordinates, computed in

direction cosine coordinates. If the status of this operating condition is paraxial, rays are aimed at the entrance pupil in fractional pupil coordinates, computed as linear distance coordinates.

Temperature (tem) - This is the temperature of the lens, in degrees Celsius.

Pressure (pre) - This is the atmospheric pressure of the air spaces in the lens, in atmospheres.

System Notes

System notes are used for any annotation that needs to be kept with the optical system. Each system note is eighty (80) characters in length and is numbered. Entering system notes is described in section “Notes (System Notes)” on page 126.

Lens Drawing

The lens drawing operating conditions include items that determine the default appearance of plan view, wire frame, and solids lens drawings. The definitions of the default rays for computing ray trajectories for a lens drawing are also in the lens drawing operating conditions. The lens drawing operating conditions are described and illustrated in detail in the section “Lens Drawing Conditions” on page 238.

Optimization

The optimization operating conditions control various aspects of the optimization process. The optimization operating conditions are described and illustrated in detail in the section “Optimization Conditions” on page 387.

Std **Prm** Configuration

The zoom conditions allow you to assign a weight to each configuration (**cfwt**) and to designate whether a particular configuration is active or not (**cfac**). The values of the configuration operating conditions are used during

1. Error function generation, and
2. MTF/RMS wavefront tolerancing.

Prm Partial Coherence

The nature of the object and the illumination for a lens are controlled by the partial coherence operating conditions. The partial coherence operating conditions are described and illustrated in detail in the section “Partial Coherence Conditions” on page 461.

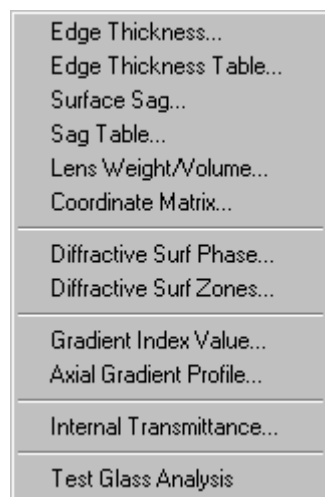
Prm Polarization

The polarization state of the light leaving the object is controlled by the polarization operating conditions. The polarization operating conditions are described and illustrated in detail in the section “Polarization” on page 329.

Prm Tolerance

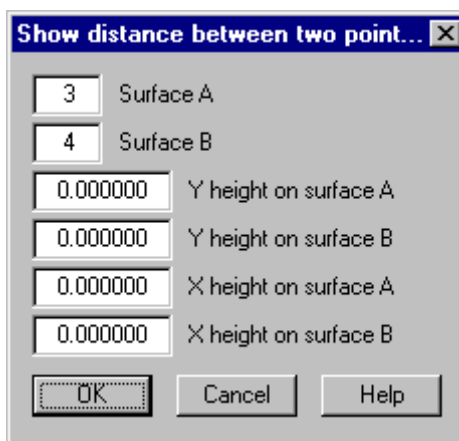
The tolerancing operating conditions control various aspects of the tolerancing process. The tolerancing operating conditions are described and illustrated in detail in the section “Grades/Conditions” on page 409. Although different forms of tolerancing are available in all editions of OSLO, the tolerancing operating conditions are only available in OSLO Premium.

Show Auxiliary Data



Edge Thickness

Edge thickness computes the distance between any two points on any two surfaces in the lens system. Select the Show menu item and then click Edge Thickness to display the distance between points on two surfaces.



The input data for edge thickness are:

Surface A - is the surface number of the surface containing the first point.

Surface B - is the surface number of the surface containing the second point. This surface number may be the same as Surface A.

Y height on surface A - is the y height of the first point, measured from the vertex of the surface without regard to tilt and decentrations.

Y height on surface B - is the y height of the second point, measured from the vertex of the surface without regard to tilt and decentrations.

X height on surface A - is the x height of the first point, measured from the vertex of the surface without regard to tilt and decentrations.

X height on surface B - is the x height of the second point, measured from the vertex of the surface without regard to tilt and decentrations.

To display the output, click the OK button to accept the command. To cancel the command, click the CANCEL button.

The output displays the following data:

SRFA - is the surface number of the Surface A containing the first point.

SRFB - is the surface number of the Surface B containing the second point.

YHTA - is the y height of the point on Surface A.

YHTB - is the y height of the point on Surface B.

XHTA - is the x height of the point on Surface A.

XHTB - is the x height of the point on Surface B.

XSAGA - is the sag of the Surface A at the point YHTA, XHTA.

XSAGB - is the sag of the Surface B at the point YHTB, XHTB.

XDIS - is the x component of the distance between the point on Surface A and the point on Surface B.

YDIS - is the y component of the distance between the point on Surface A and the point on Surface B.

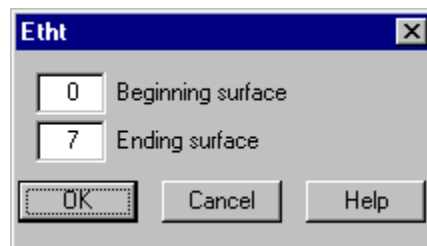
ZDIS - is the z component of the distance between the point on Surface A and the point on Surface B.

GLOBAL DISTANCE - is the distance between the point (XHTA, YHTA, XSAGA) on Surface A and the point (XHTB, YHTB, XSAGB) on Surface B.

The edge thickness command can be used to compute spacer thickness, as well as element thickness. It can also be used to find the distance between points on the same surface.

Edge Thickness Table

Computes edge thicknesses for entered surface range and prints a summary table. Apertures of surfaces are used to specify location of edge. Special aperture information is ignored.



Edge Thickness Summary Table:

SRF	EDGE TH	APER RAD
0	1.0000e+20 *	3.6397e+19
1	0.848262	6.500000
2	5.506220 *	6.500000
3	2.285907	5.000000
4	5.490720 *	5.000000
5	0.581643	6.500000
6	44.218720 *	6.500000
7		18.170326

* Edge Thickness (SRF to SRF+1) calculated with unequal apertures

Surface Sag

Surface sag computes the sag of at any point on any surface in the lens system. Select the Show menu and then click Surface Sag to display the sag and surface normal. The dialog box for surface sag is similar to the one for edge thickness, but refers only to one surface.

The input data for surface sag are:

Surface number - is the surface number of the surface that contains the sag point.

Y height on surface - is the y height of the sag point, measured from the vertex of the surface without regard to tilt and decentrations.

X height on surface - is the x height of the sag point, measured from the vertex of the surface without regard to tilt and decentrations.

To display the output, click the OK button to accept the command. To cancel the command, click the CANCEL button.

The output displays the following sag and surface normal data:

SURFACE - is the surface number of the surface that contains the sag point.

Y - is the y height of the sag point, measured from the vertex of the surface without regard to tilt and decentrations.

X - is the x height of the sag point, measured from the vertex of the surface without regard to tilt and decentrations.

Z (SAG) - is the z component of sag at the sag point, measured from the vertex of the surface without regard to tilt and decentrations.

NVL - is the y direction cosine of the surface normal at the sag point, measured without regard to tilt and decentrations.

NVK - is the x direction cosine of the surface normal at the sag point, measured without regard to tilt and decentrations.

NVM - is the z direction cosine of the surface normal at the sag point, measured without regard to tilt and decentrations.

Sag Table

This command calculates the sag and normal vector of a surface, given a point on the vertex plane.

The Y height on the surface and the X height on the surface specify an (X,Y) point on the vertex plane of the surface.

Six numbers are printed:

Y and **X** are the Y and X coordinates (input values).

Z (SAG) is the surface sag at the (X,Y) position.

NVL, **NVK**, and **NVM** are the Y, X, and Z direction cosines of the surface normal, respectively.

Lens Weight/Volume

The ***weight** command gives an estimate of the weight and volume of a lens. It assumes that the lens surfaces are spherical, and that there are no chamfers, etc., that might affect the volume. The command has the option to estimate the density of a material from its refractive index, which means that it can be used (with varying accuracy) with almost any material.

```
*Lens Volume/Weight Estimate (spherical, centered, units mm)
***WARNING: Using density estimate based on index of F4 = 1.616592
```

```
Surface 3
First surf RD (mm) = -20.250000
Second surf RD (mm) = 19.300000
Lens diameter (mm) = 10.000000
Glass density (gm/cm3) = 3.593219
Lens volume (mm3) = 128.758714
Lens weight (gm) = 0.462658
```

Prm Coordinate Matrix

The Show >> Auxiliary Data >> Coordinate Matrix command allows access to the global coordinate transformation matrices for the current lens. These matrices provide the transformation from the local coordinate system associated with each surface to the global coordinate system established by surface 1. In addition, the 3-D viewing transformation matrix, specified by the **dlva** and **dlha** lens drawing operating conditions (see p. 239), may be accessed.

Evaluation option:	
<input checked="" type="radio"/> Display coordinate matrix	<input type="radio"/> Multiply matrix and vector
Matrix type option:	
<input checked="" type="radio"/> Surface global coordinate matrix	<input type="radio"/> 3-D viewing transformation matrix
<input type="text"/>	x coordinate of vector
<input type="text"/>	y coordinate of vector
<input type="text"/>	z coordinate of vector
<input type="text" value="1"/>	Surface number

The surface global coordinate matrices are displayed in the 3×4 format

(7.10)

$$\begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \end{bmatrix}$$

The transformation from the local coordinates (x_L, y_L, z_L) to the global coordinates (x_G, y_G, z_G) is given by

(7.11)

$$\begin{bmatrix} x_G \\ y_G \\ z_G \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & x_L \\ c_{21} & c_{22} & c_{23} & y_L \\ c_{31} & c_{32} & c_{33} & z_L \end{bmatrix} \begin{bmatrix} c_{14} \\ c_{24} \\ c_{34} \end{bmatrix}$$

Thus, the location of the vertex of the surface, $(x_L, y_L, z_L) = (0, 0, 0)$, is given by the fourth column of the matrix.

The 3-D viewing transformation matrix is a 3×3 general rotation matrix, analogous to the first three columns of Eq. (6.1).

In addition to displaying the matrices, the command may be used to multiply an input local vector by the matrix and display the resulting global vector. This enables arbitrary local surface and ray data to be converted to global coordinates.

Diffractive Surface Phase

The Show >> Auxiliary Data >> Diffractive Surf Phase command computes the additional phase added to a ray by a diffractive surface (linear grating, hologram, power series CGH, or Zernike phase surface) at the point (x, y) on the surface.

1	Surface number
0.000000	Y height on surface
0.000000	X height on surface

The phase is displayed as a multiple of 2π . Also displayed are the effective grating spacings (in the current lens units) in the y and x directions. For a diffractive surface, the effective grating spatial frequency (the inverse of the effective grating spacing) is proportional to the first derivative of the phase.

Std Prm Diffractive Surface Zones

The Show >> Auxiliary Data >> Diffractive Surf Zones command computes and displays the diffractive zone radii for rotationally-symmetric CGH (**dfr**, **dra**, and **zrr**) diffractive surfaces. OSLO solves the equation that defines the phase function for integer multiples of the specified increment of 2π and displays the zone radii in either columnar or tabular format. Minimum and maximum values of the aperture radius range may be specified; the default is from the axis to the aperture radius value of the surface.

Output format for zone radii:	
<input checked="" type="radio"/>	Column of zone radii
<input type="radio"/>	2-D table of zone radii
1	Surface number
1.000000	Multiple of 2 Pi for zone boundaries
0.000000	Minimum aperture radius for zone locations
0.000000	Maximum aperture radius for zone locations

Note that any fraction of 2π may be used as the boundary identification. Thus, the boundaries for the different levels of an N -level binary optic surface may be found by using i/N , with $1 \leq i \leq N$ for the 2π multiple. On the other hand, the zone

boundaries for an M^{th} ($M \geq 1$) order kinoform may be found by using M for the 2π multiple.

Gradient Index Value

The Show >> Auxiliary Data >> Gradient Index Value command computes the value of the index of refraction at the point (x, y, z) relative to the vertex of the specified surface, for the specified wavelength.

1	Surface number
1	Wavelength number
0.000000	Y coordinate of observation point (relative to surface vertex)
0.000000	X coordinate of observation point (relative to surface vertex)
0.000000	Z coordinate of observation point (relative to surface vertex)

The output consists of the input values of the surface number, wavelength, and y, x, and z coordinates, along with the computed value of the refractive index. If the surface is followed by a gradient index medium, the appropriate form of the index distribution equation is used to compute the index; otherwise the reported value is just the normal (homogeneous) index associated with the surface.

Note that the point (x, y, z) is not constrained to lie within the volume defined by the surface's profile, thickness, and aperture.

Axial Gradient Profile

Revision 5.2 has added a new command to plot the index of refraction of an axial gradient index material as a function of axial position.

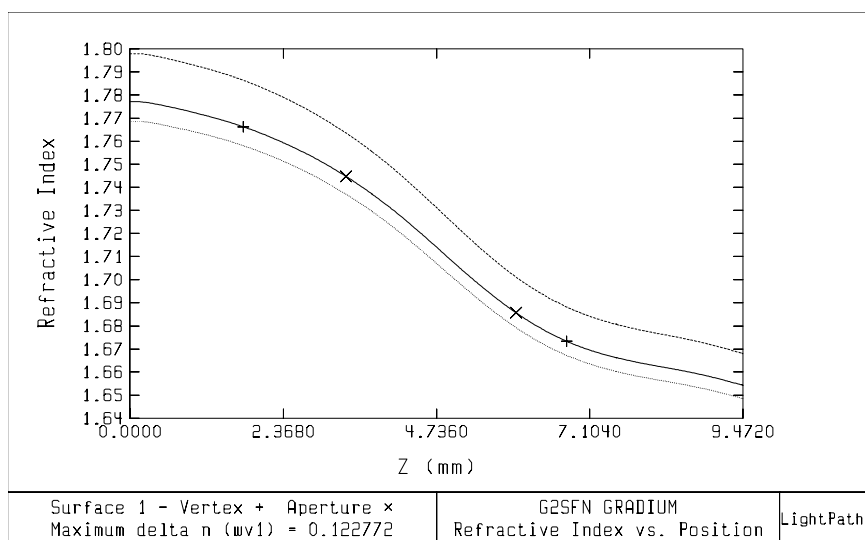
You can access this function using Show >> Auxiliary Data >> Axial Gradient Profile or the **grindex** CCL command with the syntax

grindex(*surface number, wavelength number, number of axial points*)

In addition, if you designate the wavelength for the plot as wavelength 1 and there are three or more currently defined wavelengths, the command will prompt you to select an option for plotting. The two options are plot the index profile for wavelength 1 only (the default), or plot for wavelengths 1 (solid line), 2 (dashed line), and 3 (dotted line).

If the material is a GRADIUM™ axial gradient, the plot is shown for the entire thickness of the blank. The positions of the surface vertices in the blank are marked with “+” symbols and the axial positions of the aperture radius points on the surfaces are marked with “x” symbols. This allows you to see the portion of

the available index profile within the blank that is utilized by the current lens. For other axial gradient types, the plot is shown for the thickness of the element.



Internal Transmittance

This option includes internal transmittance data for the Schott, Ohara, and Corning glass catalogs. Refer to Chapter 4 for additional information on this new feature.

Std Prm Test Glass Analysis

The Test Glass Analysis (**tga**) command performs an analysis of the current radii of curvature using the current test glass list. This command does not change any of the radii in the lens data; it just uses the test glasses to “rank” the surfaces according to five criteria. These rankings may be useful in determining which surfaces to fit to test glasses at various points during the design process. A discussion of the test glass (or test plate) fit procedure is given in Smith.⁷

The output of the command consists of a display of the current radius of curvature, the nearest test glass radius, and the rank of the surface in five different criteria.

Power gap – This is the “power gap” in the test glass list in which the current radius of curvature lies. Assume that the current radius lies between the test glass radii R_a and R_b . The corresponding curvatures of the test glass radii are $c_a = 1/R_a$ and $c_b = 1/R_b$. If n_i and n'_i are the refractive indices on the incident and refracted sides of the surface for wavelength i , and there are M wavelengths currently defined, the power gap for the surface is

$$\text{Power gap} = \left| (c_a - c_b) \sum_{i=1}^M (n'_i - n_i) \right| \quad (7.12)$$

7. W. J. Smith, *Modern Lens Design*, McGraw-Hill, 1992, §2.9, pp. 16–18.

The surface with the largest value of the above quantity has the ranking “1”.

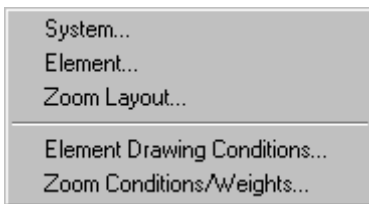
Power difference from test glass radius – The surface with the ranking “1” in this quantity has the largest change in the magnitude of surface power (summed over the defined wavelengths) when the radius is changed to the closest test glass radius.

Curvature difference from test glass radius – The surface with the ranking “1” in this quantity has the largest change in the magnitude of surface curvature when the radius is changed to the closest test glass radius.

Surface power – This is a ranking from largest surface power (in absolute value, summed over the defined wavelengths) to smallest surface power.

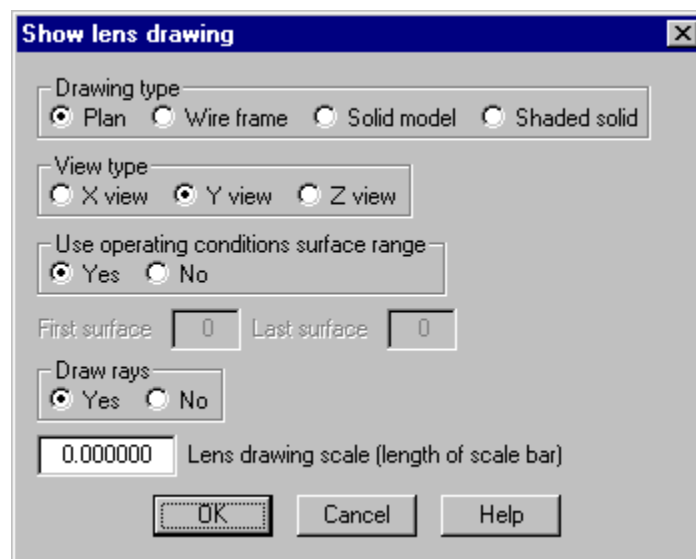
Curvature – This is a ranking from largest curvature (in absolute value), i.e., smallest radius of curvature, to smallest curvature, i.e., largest radius of curvature.

Lens Drawing



System

The lens drawing capabilities of OSLO allow you to draw the entire optical system or any subsection of your system. A lens drawing generated by OSLO accurately depicts the system as interpreted by the program. To generate a lens drawing, click the Show menu and choose Lens Drawing. This will open the lens drawing spreadsheet for complete control of the lens drawing process.



The entire optical system will be included in the drawing if the operating conditions for first surface and last surface have their default values (0). A subsection of the

system is drawn if the Use operating conditions surface range No radio button is selected or if the surface range operating conditions have non-zero values. When the range No button is selected, the first and last surfaces to be drawn are specified by entering data into the First Surface and Last Surface spreadsheet cells.

The scale of the lens drawing is determined automatically by the program. You can choose to override the automatic scaling by setting the value of the Lens drawing scale cell to a specific value prior to drawing the lens system. The value you enter for the scale will be used as the length of the scale bar shown in the upper left of the drawing.

The spreadsheet data is accepted by clicking the OK button or dismissed by clicking the CANCEL button. Accepting the data will generate the drawing.

Unless otherwise specified, the lens system used in the figures is from the Public lens file "demo\usrguide\demotrip.len."

Drawing types

The lens drawing functions in OSLO support plan drawings for views from the x, y, or z axis. The lens drawing functions also support wire frame drawings and solid model drawings with hidden lines removed for views from any orientation. The lens drawing operating conditions control the distances reserved in front and following drawings for ray trajectories, viewing angles for wire frame and solid lens drawings, the default rays for drawing trajectories, and other drawing aspects.

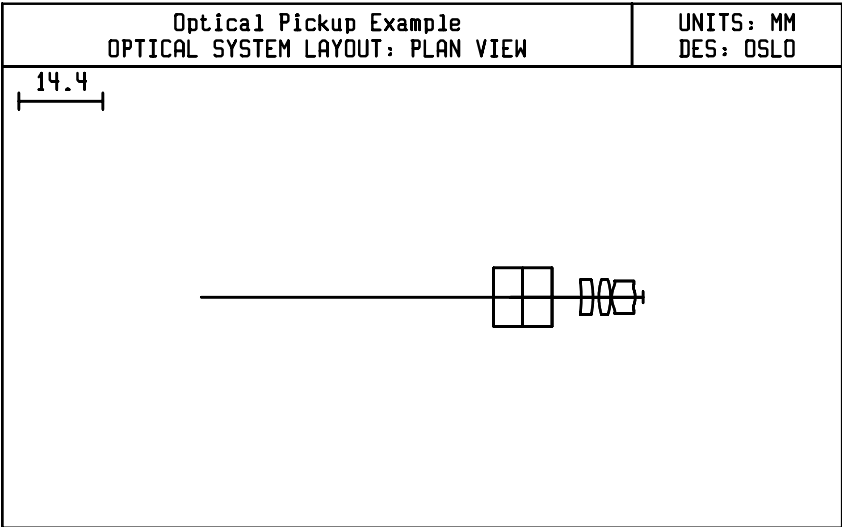
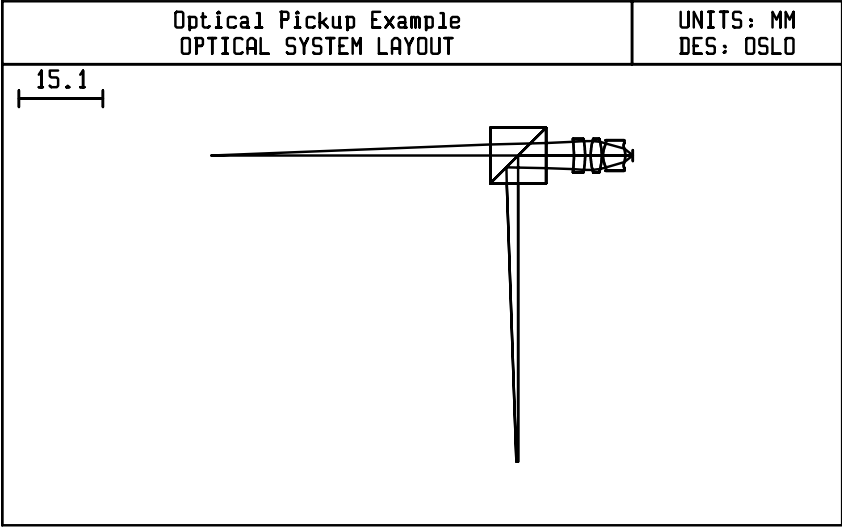
The SHIFT+F1 toolbar button generates a Y plan view lens drawing with default ray trajectories.

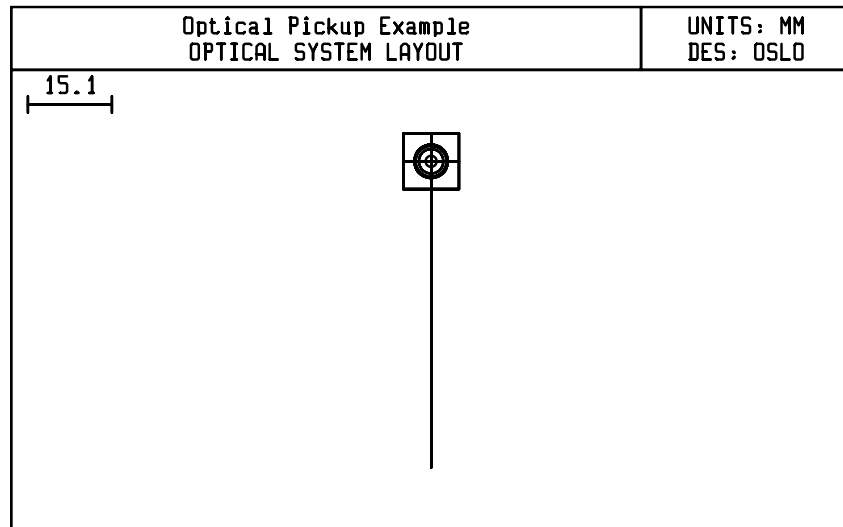
Plan drawing - The plan view is selected by clicking the Plan radio button for the drawing type. Once the plan view is selected, you can select an x, y, or z plan view. The plan view is a 2D drawing, included to obtain the highest speed drawing for general use as well as various interactive display needs in SCP and CCL routines. It is not a section of a 3D drawing. This means, for example, that it does not produce an accurate representation of systems having azimuthal or multiple tilts. You should use the wire frame or solid model drawing for accurate representation of these systems.

The optical pickup example lens in the Public lens file "demo\standard\pickup.len" was used for the drawings on the next page. The default drawing ray set was included in each drawing.

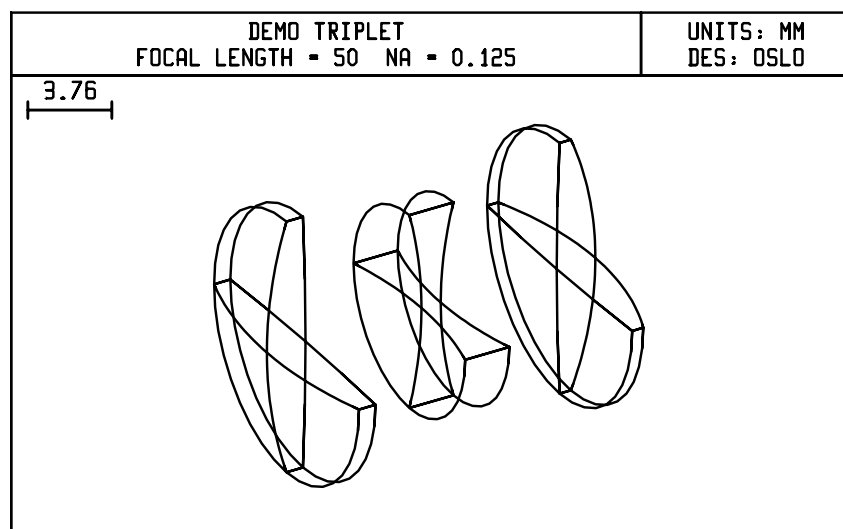
If the focal length of the lens can be computed (i.e. there are no special data that prevent computation of paraxial constants), the title line of the lens drawing will

show the focal length and numerical aperture; otherwise the title will read “OPTICAL SYSTEM LAYOUT,” as shown in the figures in on the next page.





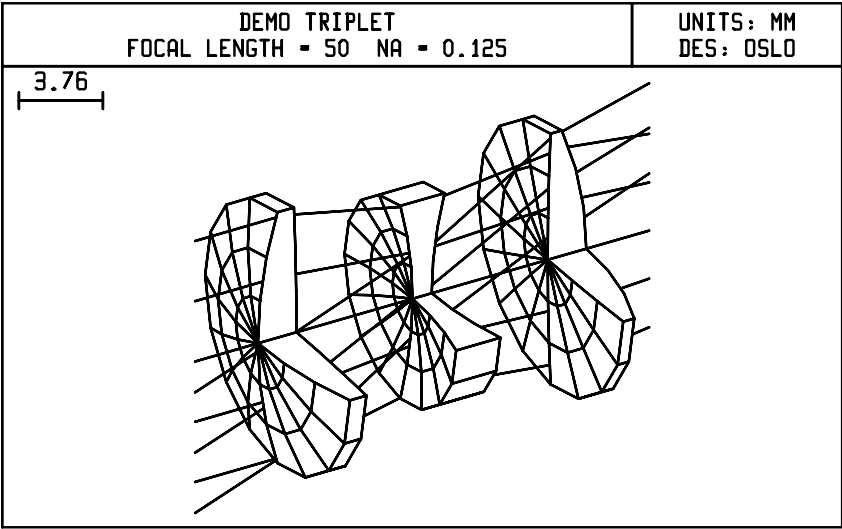
Wire frame drawing - The wire frame view is a 3D drawing, selected by clicking the Wire Frame radio button for the drawing type. The orientation of the view is established by the lens drawing operating conditions. A vertical viewing angle of 30 degrees and a horizontal viewing angle of 240 degrees were used for the wire frame drawing below.



The SHIFT+F2 toolbar button generates a solid model lens drawing with default ray trajectories.

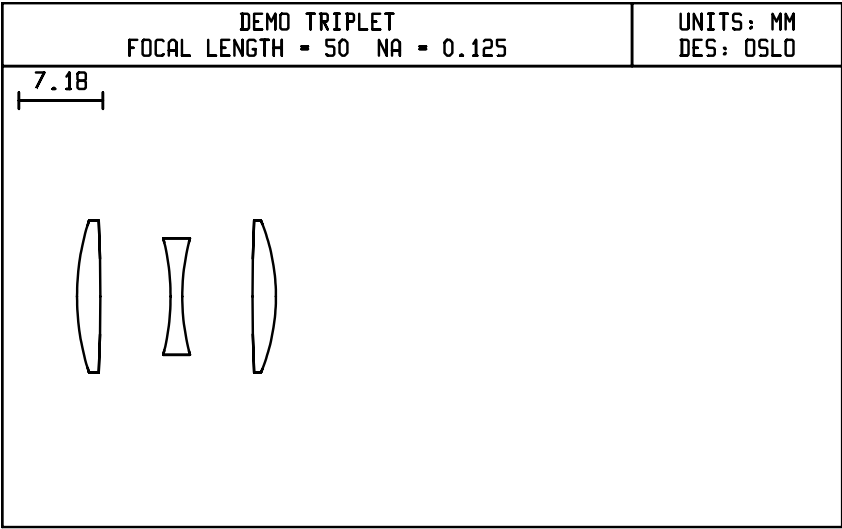
Solid model drawing - The solid model view is selected by clicking the Solid Model radio button for the drawing type. The orientation of the view is established

by the lens drawing operating conditions. The vertical viewing angle and the horizontal viewing angle were the same as those of the wire frame drawing above.



Draw rays

Ray trajectories for a set of default rays, i.e., those rays defined in the lens drawing operating conditions, may be automatically included on a lens drawing generated from the menu. Selecting the Draw Rays Yes radio button will include them in the drawing. Ray trajectories will not be included if the No button is selected. The lens drawing below was drawn with the Draw Rays No button selected. The drawing was forced to the left edge by setting the final distance **dlfd** to the back focus (43 mm).



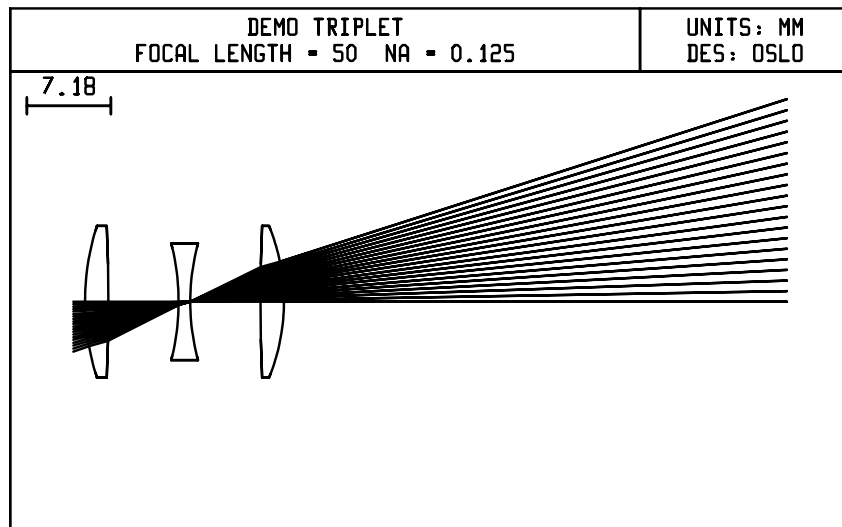
The default rays for a lens drawing can be added any time after the lens is drawn by issuing the **draw_default_rays (ddr)** command. The default ray set is established by defining lens drawing operating conditions for the field point and aperture coordinates for the desired rays.

Actually the command used to draw the rays was

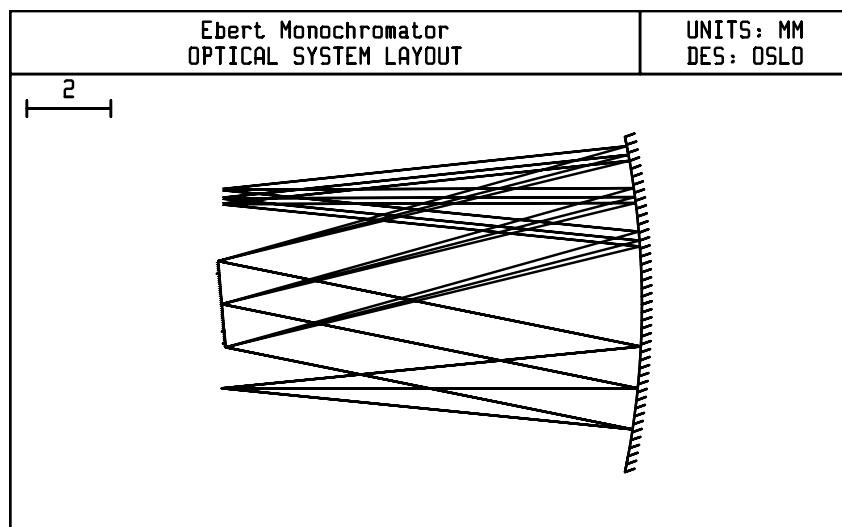
for (f=0; f<=1; f+=.05)

drr(+f, 1, 0, 0)

In addition to the default rays, ray trajectories for any ray from any object point can be added to a lens drawing by using the **draw_rays (drr)** command. The command must be entered from the keyboard after the lens is drawn. If you include a question mark as the argument to the command, i.e. **drr ?**, OSLO will prompt you for the information necessary to compute and display the ray trajectories. The ray trajectories in the figure below were drawn by repeated use of the **draw_rays** command.



The default ray set established in the lens drawing operating conditions allows field points to be defined for different wavelengths. The figure below shows the ray trajectories for three different wavelengths being separated by a grating monochromator system. Light enters from the bottom, is collimated by a mirror, and then strikes a grating.



Element

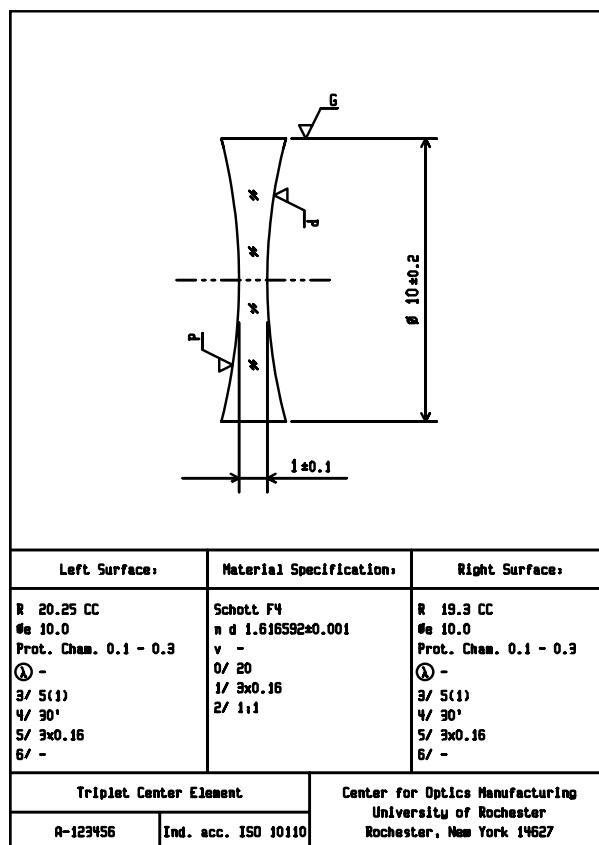
The lens element drawing capabilities of OSLO allow you to generate ISO 10110 style drawings of the individual elements of your system. To generate an element drawing, click the Show menu item and select Element Drawing. You will be asked to select the element by choosing the first surface number of the element from a list of first surface numbers for all the elements in the system. This will open the main element drawing spreadsheet for control of the element drawing process and annotation.

Element Surfaces 3 - 4		Thickness (mm)		1.000000	+/-	0.100000
Material is Schott F4 n d 1.616592 +/- 0.001000				Part		
Drawing Title				Datum Axis		
				No Annotation		
Diameter (mm)		10.000000	+	0.200000	-	0.200000
Rms Surface Roughness for Ground Edges (microns)				-		
Sampling Length for Edge Roughness (mm):				Low Limit	0	High Limit
				0		0
Stress Birefringence (nm per cm of opt. path) (0/):				20		
Bubbles and Inclusions (1/):				Number	3	Grade
						0.160000
Inhomogeneity and Striae (2/):				Inhomogeneity Class	1	Striae Class
						1
Delete drawing data		Edit left surface		Edit right surface		

The left surface spreadsheet and right surface spreadsheet can be opened from the main element drawing spreadsheet. Click the OK button in the main element drawing spreadsheet to accept the data and generate the drawing. The entry fields for the main, left, and right surfaces of the element drawing spreadsheets are described following the sample element drawing shown below.

Note: OSLO element drawings follow the guidelines prescribed by ISO 10110. However, the drawings are not guaranteed to be 100% compliant.

An element drawing consists of several parts: (a) the element drawing itself, (b) a tabular data area divided into sections for the left surface, material, and right surface, and (c) a title area divided into sections for the drawing title and part number and up to three lines of address information (defined by preference items **Address1**, **Address2**, and **Address3**). See ISO 10110, Part 1 for fundamental stipulations, general presentation and dimensioning specifications, and additional indications for optical elements and systems.



Element drawing data is retained in the lens file when a lens is saved. Drawing data for an individual element may be deleted by activating the Delete Drawing Data button in the Element Drawing Specification spreadsheet. Element drawing data for all elements throughout the current lens may be deleted by executing the **clear_all_element_data (cae)** command.

The data entry fields in the main element drawing spreadsheet are:

Drawing title - The drawing can be titled independently for each element.

Part identification - The element can be identified by a part name and/or number.

Datum axis - As stated in ISO 10110, Part 6, "The datum axis is an axis selected by consideration of specific features of an optical system. It serves as a reference for the location of surfaces, elements, and assemblies."

Diameter with tolerance - The outside diameter of the element can be specified with a tolerance value. Independent plus and minus tolerances are permitted for the diameter. The diameter must be greater than or equal to the maximum effective optical aperture, as defined in the lens data.

Rms surface roughness for ground surfaces - See ISO 10110, Part 8 for the definition and specification of surface texture, including rms surface roughness for matte or ground surfaces. The matte surface roughness is indicated in root-mean-square (rms) height variation and depends on the sampling length. Surface texture indications for matte surfaces are illustrated ISO 4287/1, subclause 5.11.

Stress birefringence - See ISO 10110, Part 2 for the definition and specification of stress birefringence. Stress birefringence is specified in terms of optical path difference per unit path length, expressed in nm/cm. The code number for stress birefringence is 0 (zero) and is indicated by

0/A

where **A** is the maximum permissible stress birefringence in nm per cm optical path length.

Bubbles and inclusions - See ISO 10110, Part 3 for the definition and specification of bubbles and other inclusions. The code number for “bubbles and other inclusions” is 1. The specification is indicated by

1/ N x A

The term **N** is the allowed number of bubbles and inclusions, and **A** is a grade number which is equal to the square root of the projected area of the largest permissible bubble and/or inclusion, expressed in millimeters.

Inhomogeneity class - See ISO 10110, Part 4 for the definition and specification of inhomogeneity classes. Six classes of inhomogeneity are defined. Inhomogeneity is characterized by maximum permissible variation in refractive index within a part of 10^{-6} .

Striae class - See ISO 10110 Part 4 for the definition and specification of striae classes. Class 1 to Class 4 are related to a density of striae, defined as the ratio of effective projected area containing striae to the area under test, expressed in per cent. Class 5 is extremely free of striae and expects further information specified in a note.

The code number for inhomogeneity and striae class is 2 and indicated by

2/A;B

where **A** is the inhomogeneity class and **B** is the striae class.

Left (Right) surface drawing specification - Optical specification of the left and right surfaces of an element are entered via the Element Drawing Specification Surface spreadsheet.

Surface 3				Element dimensions are in mm.	
Radius	20.250000	CC +/-	0.000000	Centering Tol. (minutes) (4/)	30.000000
Optically Eff. Diameter	10.000000	Prot. Chamfer	0.100000	-->	0.300000
Surface Polishing Grade	P	Rms Surface Roughness - Rq (nm)	---		
Sampling Length for Rq (microns):		Low Limit	0	High Limit	0
Surface Form Deviations - 3/A(B/C):					
A = Sagitta error, B = Irregularity, C = Rotationally symmetric irregularity					
A (fringes)	5.000000	B (fringes)	1.000000	C (fringes)	0.000000
RMS Residual Surface Deviations (3/):					
RMSt = Total rms deviation, RMSi = Rms irregularity, RMSa = Rms asymmetry					
RMSt <	0.000000	RMSi <	0.000000	RMSa <	0.000000
Surface Imperfection Tolerances (5/):					
Surface Imperfections	Number	3	Grade	0.160000	
Coating Blemishes	Number	0	Grade	0.000000	
Long Scratches	Number	0	Max Width (nm)	0.000000	
Edge Chips	Maximum Extent From Edge (nm)	0.000000			
Surface Treatment/Coating Specification					
Laser Irradiation Damage Threshold (6/)					

Surface form - See ISO 10110, Part 5 for the definition and specification of surface form. Surface form deviation is defined in ISO 10110 as “the distance between the optical surface under test and the nominal theoretical surface, measured perpendicular to the theoretical surface, which shall be nominally parallel to the surface under test.” The code number for surface form is 3 and is indicated in one of the three forms:

- 3/A(B/C)
- 3/A(B/C) RMS x < D (where x is one of the letters t , i , or a)
- 3/- RMS x < D (where x is one of the letters t , i , or a)

where

A is either the maximum permissible sagitta error expressed in fringe spacings, or a dash indicating that the total radius of curvature tolerance is given in the radius of curvature dimension.

B is either the maximum permissible value of irregularity expressed in fringe spacings, or a dash indicating no explicit irregularity tolerance is given.

C is the permissible rotationally symmetric irregularity expressed in fringe spacings or, if no tolerance is specified then /**C** is replaced by the closing parenthesis, and

D is the maximum permissible value for the rms residual deviation of the type indicated by x .

ISO 10110, Part 5 states that, “For this Part of ISO DIS 10110, the unit for surface form deviation is called a ‘fringe spacing,’ and it is equal to one-half the wavelength of light.”

Rms residual surface deviations - See ISO 10110, Part 5 for the definition and specification of root-mean-square (rms) residual surface deviations. The three types of rms residual deviation are: total rms deviation (**RMSt**), rms irregularity (**RMSi**), and rms asymmetry (**RMSa**). Rms residual surface deviations may be included in the code number 3 specification, as described in the **Surface form** explanation above. Note that, as per ISO 10110, “rms residual deviation of a given optical surface cannot be determined visually, and digital techniques are therefore required.”

Centering tolerance - See ISO 10110, Part 6 for the definition and specification of centering tolerances. The code number for centering tolerances is 4 and the tolerance is indicated by one or two tolerance values and, when necessary, a reference to a datum axis. The tolerance has one of three forms:

- 4/ σ
- 4/ σ (L)
- 4/ $\Delta\tau$

where σ is the maximum permissible tilt angle, **L** is the maximum permissible lateral displacement, and τ following the Δ is the maximum permissible cement wedge angle. The values of centering tolerances are in minutes ['] or seconds ["] of arc for angular dimensions and in millimeters for linear dimensions.

Protective Chamfer with tolerance - See ISO 10110, Part 1, Subclause 4.6.4.2 for the definition and indications of protective chamfers on a drawing. The minimum and maximum permissible widths of the chamfers are indicated in a

note. The protective chamfer note on a drawing pertains to all edges and corners that are not individually specified.

Surface polish and polished surface roughness - See ISO 10110, Part 8 for the definition and specification of polishing grades and rms roughness. Surface texture can be specified by one or more methods: rms surface roughness (R_q) with lower and upper limits of sampling length, micro-defects, or power spectral density. The entry here is for specification in terms of rms surface roughness.

Surface imperfections - See ISO 10110, Part 7 for the definition and specification of surface imperfections (scratches, pits, coating blemishes, etc.). Two methods are allowed by ISO 10110 for specifying surface imperfections: method 1 – surface area obscured or affected by the defects, or method 2 – the visibility of the defects. OSLO accepts entry of method 1 data. The code number for surface imperfections is 5. The form of the indication is

5/NxA; C N'xA'; L N"xA"; E A'''

where

NxA indicates the number and size of general surface imperfections where **N** is the number of allowed imperfections of maximal permitted size and the grade number **A** is equal to “the square root of the of the surface area of the maximum allowed defect, expressed in mm.”

C N'xA' indicates the coating blemish specification where **C** is the letter designation, **N'** is the number of allowed blemishes and **A'** indicates the grade number as described for general surface imperfections.

L N"xA" indicates the long scratch specification where **L** is the letter designation, **N"** is the number of allowed long scratches (longer than 2 mm) and **A"** is the maximum width of the scratches, expressed in millimeters.

E A''' indicates the edge chip specification where **E** is the letter designation and **A'''** specifies “the maximum allowable extent of a chip from the physical edge of the surface, measured parallel to the surface, in mm. Any number of edge chips is permissible as long as their extents from the edge do not exceed **A'''**.”

Surface treatment and coatings - See ISO 10110, Part 9 for the definition and specification of surface treatment and coatings. A functional coating is indicated by a circle containing the Greek letter lambda (λ). The nature of coatings and surface treatments is such that separate specification documents are usually necessary to describe them. Therefore, the entry of a note for referencing the specification documents is attached to the indication.

Laser irradiation damage threshold - See ISO 10110, Part 13 for the definition and specification of laser irradiation damage thresholds. The code number for laser damage threshold is 6. The form of the indication is

6/H_{th}; λ ; pdg; f_p ; $n_{TS} \times n_p$ for pulsed laser irradiation, and

6/E_{th}; λ ; n_{TS} for continuous laser irradiation where, from ISO 10110, Part 13,

H_{th} is “the energy density threshold in units of $J\ cm^{-2}$ above which damage occurs.”

E_{th} is “the power density threshold in units of $W\ cm^{-2}$ above which damage occurs.”

λ is “the wavelength of the laser radiation in nm.”

pdg is “the pulse duration group according to ISO/DIS 11254.”

f_p is “the pulse repetition rate, i.e. the number of pulses per second of a repetitively pulsed laser.”

n_{TS} is “the number of test sites.”

n_p is “the number or pulses per test site.”

Entry of the indication should begin with the data following the **6/** code number indication.

Element drawing defaults

When an Element Drawing Spreadsheet is displayed for the first time, various default values from the ISO standard are presented. The ISO standard does not, however, specify defaults for all items appearing in the spreadsheet (and its associated surface data spreadsheets). In order to provide for the specification of defaults for these items, a Default Element Drawing Data spreadsheet is provided. These operating conditions can be viewed and updated by selecting the Element Drawing option under the Update menu Operating Conditions, as described in Chapter 4.

Drawing items taken from current lens data

The radii of curvature, thickness, aperture diameters and material specification used for element drawings are obtained from the current lens data. All linear dimensions are converted to millimeters.

The radii are expressed as positive numbers with the designations CX and CC indicating convex and concave, respectively. The radius of a plane surface is indicated by <infinite> in surface drawing specification spreadsheets, and by the infinity symbol (∞) on drawings.

The optically effective diameter of a surface is indicated in the surface drawing specification spreadsheet and appears in the tabular data area of the drawing. Two diameters may be abstracted from the lens data, however. The first is the ordinary diameter of the surface which is twice the aperture radius of the surface specified by the APERTURE RADIUS item in the lens surface data spreadsheet. The second is the special aperture diameter which is the diameter of the circle which circumscribes a centered special aperture defined for the surface. The optically effective diameter of the surface is taken to the smaller of these two diameters if both are defined.

The outside diameter of an element is an editable item. However, it is required to be at least as great as the largest of all ordinary or special aperture diameters defined for the two surfaces of the element.

When an element is drawn, convex surfaces will extend out to the specified outside diameter. Concave surfaces, however, will only extend out to the largest of the ordinary or special diameters and will then be connected to the edge of the element by a vertical line.

The name of the element material and its refractive index at the principal wavelength appear in the Element Drawing Specification spreadsheet and in the tabular data area of the drawing. The name of the material is the glass name specified in the lens data. If it is a catalog glass, the name of the catalog (e.g., Schott BK7) is included in the material name. The principal wavelength is assumed to be wavelength 1. If it is recognized as a standard spectral line (e.g., d

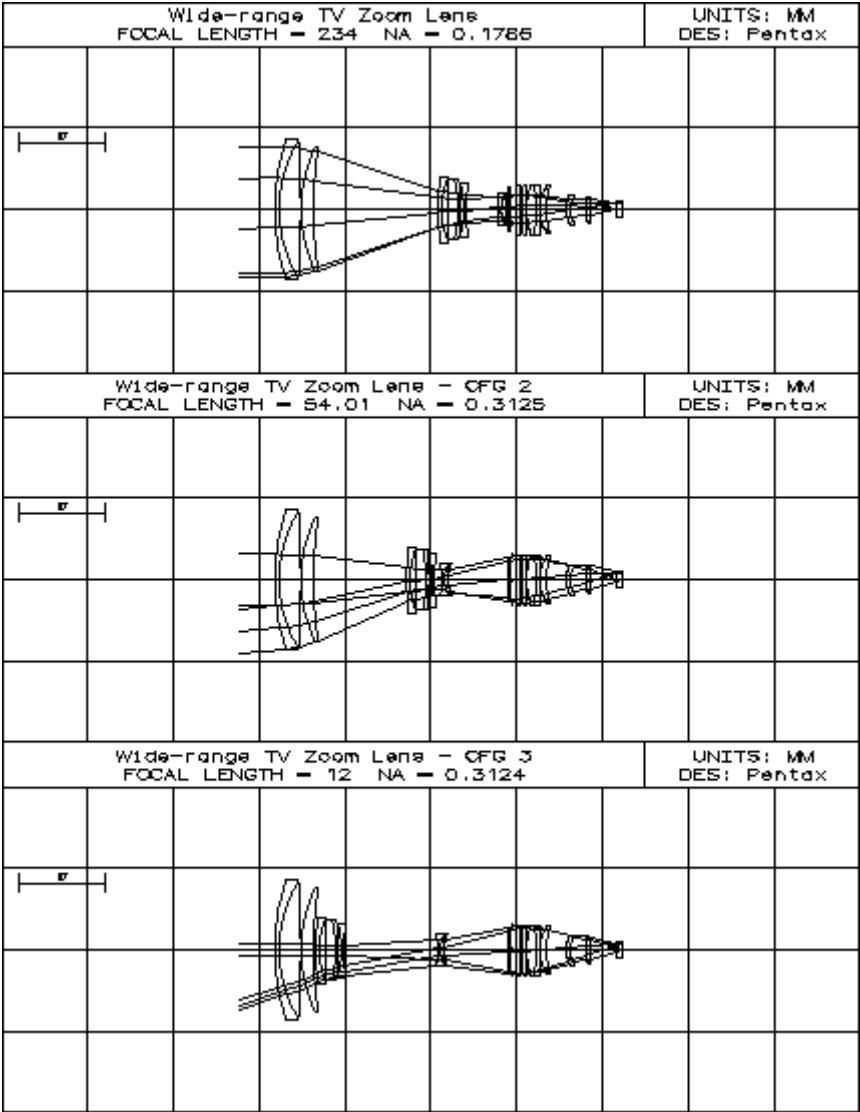
or e) it is indicated by its letter designation. Otherwise, it is indicated by its value in nanometers.

Drawing items taken from current tolerance data

The tolerances for radii or curvature, refractive index, element thickness, surface centering, sagitta error and irregularity used in element drawings are taken from the tolerance data currently defined for the lens.

Tolerances for radii of curvature and thickness are converted to millimeters. The surface centering tolerance is the surface tilt tolerance expressed in minutes of arc. The sagitta error and irregularity are expressed as interferometric fringe spacings over circular test areas.

Std Prm Zoom Layout



This menu option draws all configurations of a multiple zoom position lens in a single drawing window. The different configurations are shown displaced in the vertical direction with their image planes aligned to the same horizontal position.

Vertical and horizontal grid lines are drawn to facilitate identifying the position of the different lens elements in each configuration. EFL and system aperture information are reported in the header of each configuration. The graphic scale and the graphic window aspect ratio must be adjusted until the drawing is satisfactory.

Element Drawing Conditions

When an Element Drawing Spreadsheet for a particular element is opened for the first time, various default values, drawn from the ISO standard, are presented. The ISO standard does not, however, specify defaults for all items appearing in the spreadsheet. In order to provide for the specification of defaults for these items, this spreadsheet allows you to enter default element drawing operating conditions.

The items that can be assigned default values are:

- RMS surface roughness for ground edges
- Surface polishing grade
- RMS surface roughness
- RMS residual surface deviations
- Surface imperfection tolerances
- Surface treatment/coating specification
- Laser irradiation damage threshold

Rms Surface Roughness for Ground Edges (microns)		
Sampling Length for Edge Roughness (mm):	Low Limit <input type="text" value="0"/>	High Limit <input type="text" value="0"/>
Surface Polishing Grade	<input type="text" value="P"/>	Rms Surface Roughness - Rq (nm)
Sampling Length for Rq (microns):	Low Limit <input type="text" value="0"/>	High Limit <input type="text" value="0"/>
RMS Residual Surface Deviations (3/):		
RMSt = Total rms deviation, RMSi = Rms irregularity, RMSa = Rms asymmetry		
RMSt <	<input type="text" value="0.000000"/>	RMSi < <input type="text" value="0.000000"/> RMSa < <input type="text" value="0.000000"/>
Surface Imperfection Tolerances (5/):		
Coating Blemishes	Number <input type="text" value="0"/>	Grade <input type="text" value="0.000000"/>
Long Scratches	Number <input type="text" value="0"/>	Max Width (nm) <input type="text" value="0.000000"/>
Edge Chips	Maximum Extent From Edge (nm) <input type="text" value="0.000000"/>	
Surface Treatment/Coating Specification		
<input type="text"/>		
Laser Irradiation Damage Threshold (6/)		
<input type="text"/>		

Std Prm Zoom/Conditions Weights

The configuration operating conditions allow you to assign a weight to each configuration and to designate whether a particular configuration is active or not. The values of the configuration operating conditions are used during error function generation and MTF/RMS wavefront tolerancing.

Generate error function – If you specify a maximum operand configuration greater than one in the “Optimize>>Generate Error Function” dialog box, the current configuration weights will be used in constructing the operand weights. For example, during a zoom lens design, you may wish to weight some zoom positions more heavily than others. Also, if a configuration is marked as not active,

no operands will be generated for that configuration. Note that the configuration weights and active/inactive status are only used at the time of the error function generation; changing a weight *after* the error function has been generated does *not* change the error function. If you wish to change the weight of a configuration, field point, or wavelength weight in the error function, you must regenerate the error function.

MTF/RMS wavefront tolerancing – If you use the “all field points and configurations” option for “MTF/RMS wavefront tolerancing”, the configuration weights will be used during the computation of compensator values. Also, the tolerancing analysis will only be performed for configurations that are currently active.

Lens Drawing Conditions

The lens drawing operating conditions control the appearance and layout of lens drawings generated by OSLO. These operating conditions can be viewed and updated from the lens drawing spreadsheet by clicking the Operating Conditions button or by selecting the Lens Drawing option under the Update >> Operating Conditions menu. The operating conditions spreadsheet shown below contains the initial values that are established by OSLO whenever a new lens system is opened.

Initial distance:	<input type="text" value="0.000000"/>	Final distance:	<input type="text" value="0.000000"/>
Horizontal view angle:	<input type="text" value="240"/>	Vertical view angle:	<input type="text" value="30"/>
First surface to draw:	<input type="text" value="0"/>	Last surface to draw:	<input type="text" value="0"/>
		Autodraw:	<input type="button" value="YZ"/>
X shift:	<input type="text" value="0.000000"/>	Y shift:	<input type="text" value="0.000000"/>
		DXF/IGES view:	<input type="button" value="Unconverted"/>
Apertures:	<input type="button" value="Quadrant"/>	Rings:	<input type="text" value="3"/>
		Spokes:	<input type="text" value="4"/>
		Image space rays:	<input type="button" value="Final dist"/>
Draw aperture stop:	<input checked="" type="radio"/> Off <input type="radio"/> On	Hatch back of reflectors: <input type="radio"/> Off <input checked="" type="radio"/> On	
Shaded solid color - Red:		<input type="text" value="175"/>	Green: <input type="text" value="185"/>
		Blue:	<input type="text" value="250"/>
Number of field points for ray fans:		<input type="text" value="3"/>	Points for aspheric profiles:
	<input type="text" value="41"/>		
Frac Y Obj	Frac X Obj	Rays	Min Pupil
<input type="text" value="0.000000"/>	<input type="text" value="0.000000"/>	<input type="text" value="5"/>	<input type="text" value="-1.000000"/>
		Max Pupil	Offset
<input type="text" value="0.700000"/>	<input type="text" value="0.000000"/>	<input type="text" value="0.000000"/>	<input type="text" value="0.000000"/>
		FY	FX
<input type="text" value="1.000000"/>	<input type="text" value="0.000000"/>	<input checked="" type="radio"/>	<input type="radio"/>
		Wvn	Cfg
		<input type="text" value="1"/>	<input type="text" value="0"/>

Initial distance (dlid)

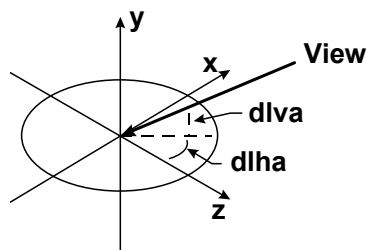
Final distance (dlfd)

These are distances (in lens units) to be reserved on the object and image sides of the first surface in the drawing for the purpose of showing the ray trajectories in object space. The default value for these operating conditions is 0, which means that the program will calculate an appropriate distance to reserve (approximately one-fifth of the lens aperture). The drawing of the triplet lens on p. 228 shows the use of the final distance to leave room on a drawing for rays converging to a focus. The initial distance is useful for drawing rays that emerge from an object point at a finite distance.

The final distance specification can be overridden by the setting of the Image space rays button, as described below. Alternately, you can tailor the drawing of rays in image space by inserting a dummy surface at a location where you want the rays to be drawn to, and then setting the final distance to a very small (but finite) distance, e.g. 1×10^{-6} .

Horizontal view angle (dlha)**Vertical view angle (dlva)**

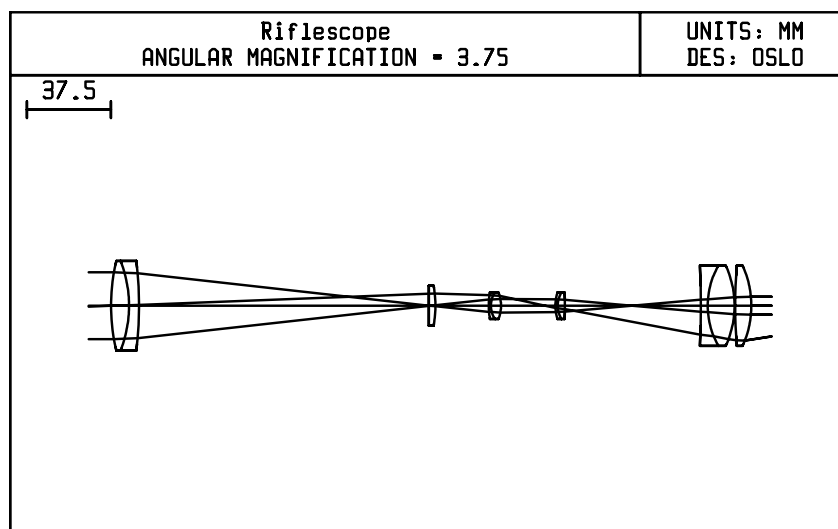
For three dimensional wire frame and solid lens drawings, these set the angles necessary to specify the viewing direction. The angles are specified, relative to the coordinate system of the first surface, in integer degrees. The horizontal view angle is the azimuthal viewing angle, i.e., a rotation angle about the y-axis measured from the z-axis toward the x-axis. The vertical view angle is the polar viewing angle, i.e., the elevation angle from the xz plane toward the y-axis. The default values for new lenses are **dlha** = 240 degrees and **dlva** = 30 degrees.

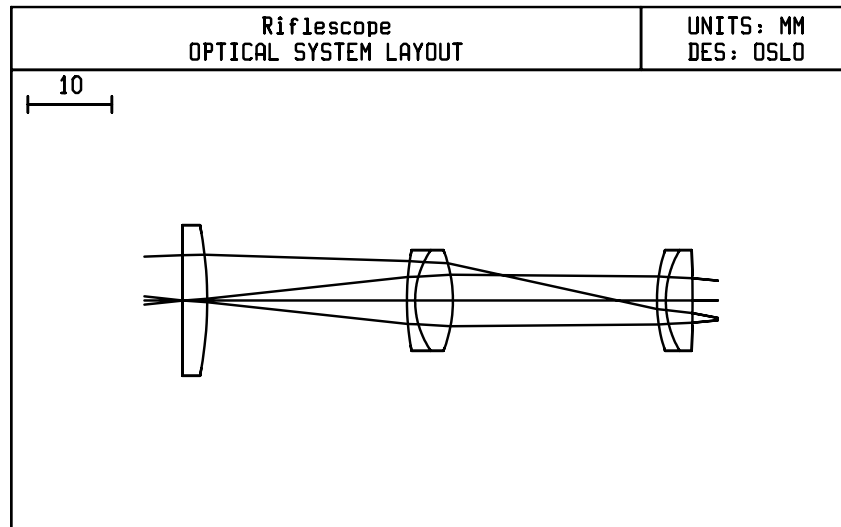


There is a CCL command on the User menu that allows you to set these angles using graphic sliders.

First surface to draw (dlfs)**Last surface to draw (dlls)**

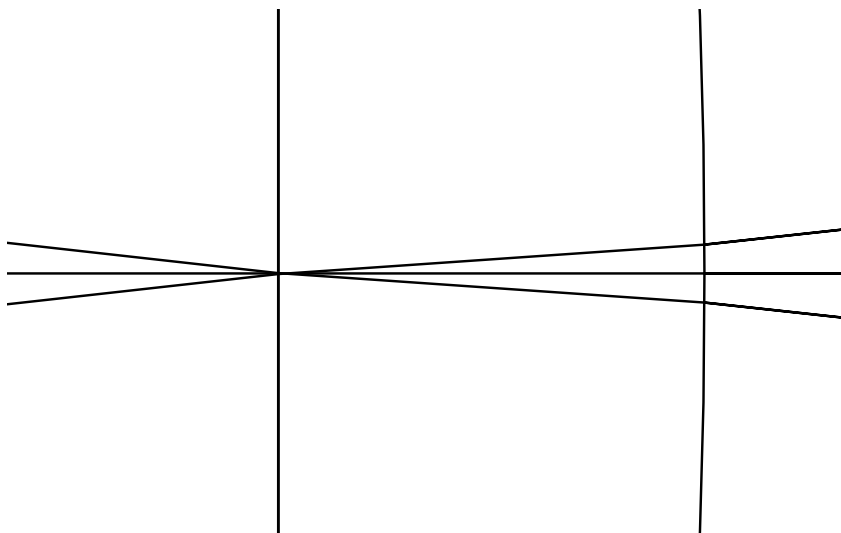
The default surface range for lens drawings is from the first surface to the next-to-last surface. To draw a different surface range, the value of the first and last surfaces to draw can be given explicitly. Note that the field in the lens drawing dialog box overrides, but does not replace, the values established by the operating conditions **dlfs** and **dlls**. The figure below shows a comparison of the default surface range (0, 0), and a drawing made with the surface range set from 4 to 11.





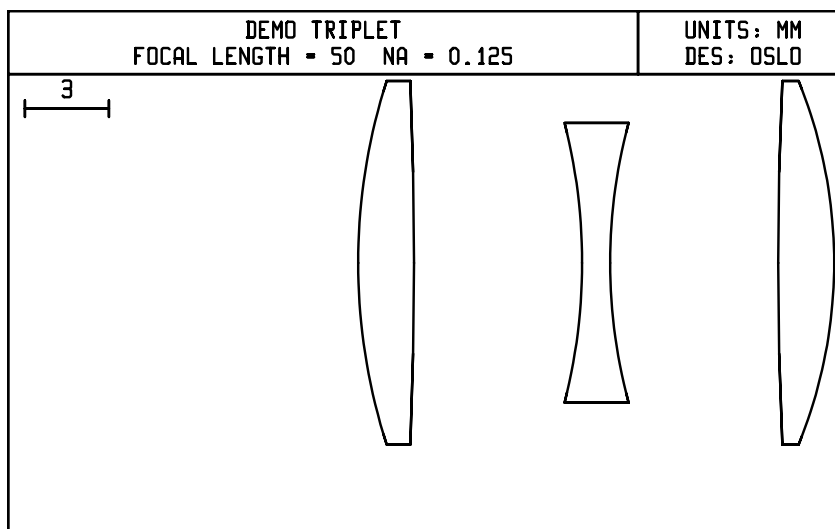
To show more detail in a lens drawing than can be obtained using the first and last surface operating conditions, the window can be zoomed using the controls on the graphics window toolbar. When the window is printed, it will be printed in its zoomed state. The figure

below shows the detail of rays focusing on the front surface (4) of the field lens in the rifle scope shown above. **X shift (dlxs)**



Y shift (dlxs)

These are shifts, in the x and y screen direction, of the position of the lens in the layout drawing, in lens units. They can be used to re-position a lens drawing in the window.



DXF/IGES view (dlcv)

A drawing of the lens can be exported to a file in either DXF or IGES format. The resulting file could be used, for example, as input to a CAD program. The **dlcv** operating condition controls how the data for plan view lens drawings is exported to the file. There are three options:

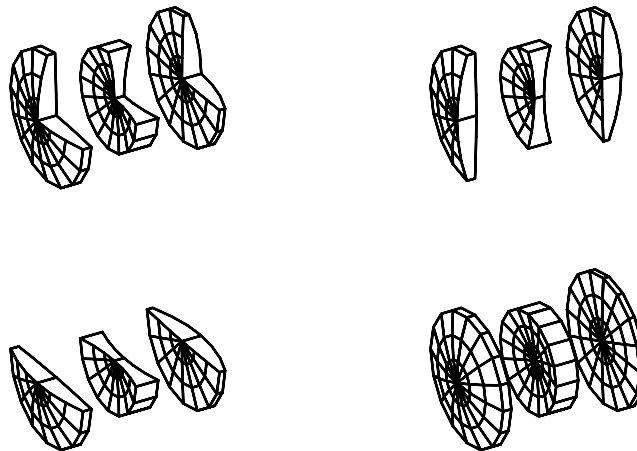
- Unconverted - the data is exported in the global OSLO (x, y, z) coordinate system.
- Use viewing matrix - the lens coordinate data is multiplied by the viewing matrix.
- 2-D transform - the two-dimensional projection of the three-dimensional lens data is exported.

Apertures (dlap)

Controls the drawing of lens apertures in three dimensional wire frame and solid drawings. The four options are:

- Quadrant ($x < 0, y > 0$) cut away,
- Half apertures, cut parallel to the yz plane,
- Half apertures, cut parallel to the xz plane, and
- Full apertures.

The default value for a new lens is quadrant cut away (**dlap** = 0). Different options for the **dlap** operating condition are illustrated below.



Rings (dlri)

Spokes (dlsp)

These are the number of rings and spokes in the aperture used to compute the surface shape in solid model drawings. The default value for rings is 3, and spokes is 4. Increasing these values will increase the resolution in the drawing, but will also increase the computation time necessary to generate the drawing.

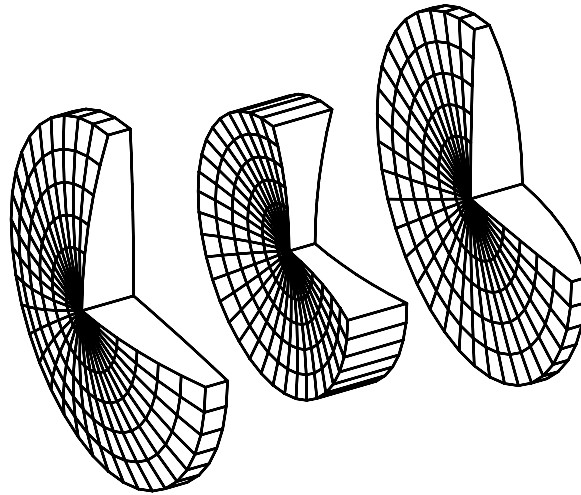
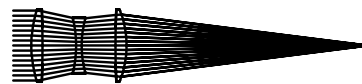
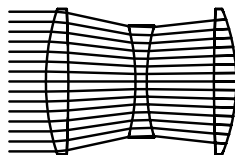
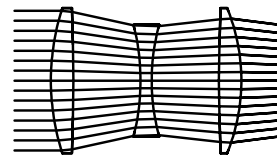
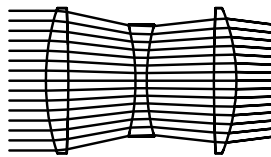


Image space rays (dlrs)

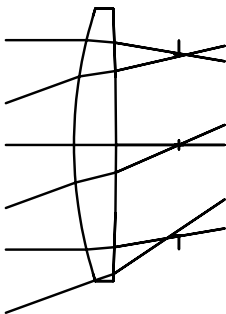
This controls the drawing of ray trajectories in image space. The four options are:

- Use the prescribed final distance,
- Draw the rays to their intersection with the wavefront in image space,
- Draw the rays to their intersection with the last drawn surface, and
- Draw the rays to the image surface.



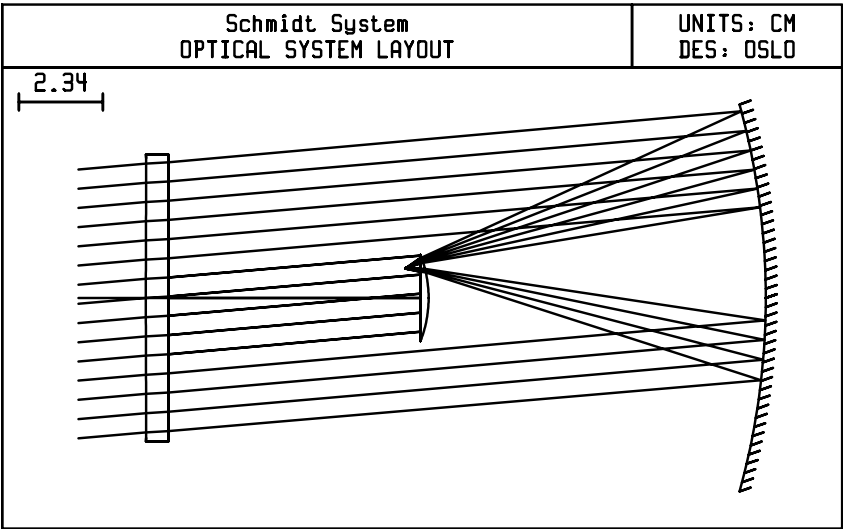
Draw aperture stop (dlas)

If this option is On, the position of the aperture stop will be drawn in x and y plan view lens drawings.



Hatch back of reflectors (dlhr)

If this option is On, the back side of reflecting surfaces can be “hatched.” In order for them to actually be hatched in a lens drawing, the reflecting surface must have a glass REFL_HATCH; both conditions are required. Hatch marks are only drawn in x and y plan view lens drawings. The figure below shows a Schmidt camera system from the OSLO demo library. Note that the aperture stop is on the first surface; rays are blocked by a special aperture (central obstruction) inserted as dummy surface 3, in the same plane as the flat surface of the final lens.



Number of field points for ray fans (dlnf)

This is the number of field points used for the default drawing rays. For new lenses, the default number of field points is two, with 3 axial rays and 1 chief ray being drawn. For each field point, there is an additional line of data that defines the rays from each field point. Each of the following operating conditions is actually an array of operating conditions, one for each field point specified.

Frac Y Obj (dlfp)**Frac X Obj (dlxf)**

This is the fractional object point (in the y and x directions respectively) from which the rays are to be drawn. Thus, this value is 0 for an on-axis object point, and 1 for a full-field point. There is a CCL command, **fpts_copy**, that will copy the specifications of the current field points into the lens drawing operating conditions for the default drawing rays.

Rays (dlnr)

This is the number of rays to be drawn from the field point.

Minimum pupil (dlmn)**Maximum pupil (dlmx)**

These are the range of the pupil that the rays from the object point are to fill. These items are specified in fractional pupil coordinates, so a value of 1 indicates a ray at the edge of the pupil.

Offset (dlos)

This is the offset of the fan of rays from the center of the pupil. The offset is in the direction orthogonal to the ray fan direction. Thus, if a y-fan of rays is traced, the offset is in the x-direction. The offset is specified in fractional pupil coordinates.

FY or FX (dlyf)

This is the orientation of the traced fan of rays. A Y orientation specifies a fan of rays parallel to the y-axis of the entrance pupil. An X orientation specifies a fan of rays parallel to the x-axis of the entrance pupil.

Wvn (dlwn)

This is the wavelength number to be used for drawing the rays.

Any changes to the operating conditions are immediately active. The Update Lens Drawing Operating Conditions changes the lens data. Note that the effects of changes made in the Update Lens Drawing Operating Conditions spreadsheet are identical whether the spreadsheet is accessed from the Show menu or from the Update menu.

Std**Prm****Cfg (dlcf)**

This item specifies which configuration should be used for drawing the rays. If the configuration number is 0, (the default value) that fan of rays is considered to be “global”, i.e., it is shown for lens drawings in all configurations. If the specified configuration number is greater than 0, that fan of rays will be drawn only if the current lens drawing was created in the specified configuration. Configuration-specific default drawing rays may be useful, for example, when a zoom lens has vignetting that varies with zoom position.

Overview



A critical part of any optical design problem is evaluating the imaging performance of the lens. The Evaluate menu provides access to many different types of lens evaluation options, including aberration coefficients, ray analysis, and both geometrical and diffraction image analysis. A special menu is provided for access to the Gaussian beam calculations in OSLO, which use a special ray trace that accounts for shifts in the locations of beam waists in weakly focused systems.

The Analysis routines on the Evaluate menu depend on the current value of four items:

- **Configuration**
- **Wavelength**
- **Object Point**
- **Spot Diagram**

Three setup commands on the menu allow you to set up these items as required. If you do not use these commands, the last value set is used, or if none has been set, an initial default is supplied. The initial default is configuration 1, wavelength 1, an on-axis object point, and a spot diagram with the default operating conditions. You should be aware that some routines, particularly user-written commands, may reset these values without notice.

All monochromatic evaluation in OSLO is carried out for the system in the current configuration, and at the current wavelength. All rays and spot diagrams will be traced from the current object point until a new one is defined.

Image analysis in OSLO is based on extended spot diagrams. OSLO uses the term *spot diagram* (SPD) in a general sense to refer to all the data obtained from tracing a bundle of rays, originating from a single object point, passing through a rectangular grid of points in the pupil. In addition to the ray-intercept information needed to display the actual spot diagram, OSLO spot diagrams maintain data needed to compute the spot diagram on focus-shifted image planes, and also the data needed to compute the image-space wavefront corresponding to the current object point.

Many evaluation options depend on a spot diagram. If a current spot diagram is not available when such an option is executed (as happens, for example, whenever the lens data is changed) one will be traced automatically, using the current object point and the current spot diagram operating conditions, as described above.

The evaluation options on the Evaluate menu are as follows.

Menu Item	Page
Paraxial Setup Enter paraxial properties (f -number, etc.) for the lens.	249
Paraxial Ray Analysis... Compute paraxial constants or the result of a paraxial ray trace.	251
Zoom Lens Parameters Compute thicknesses, paraxial constants and conjugates for surface groups that comprise a zoom/multiconfiguration lens system.	254
Aberration Coefficients... Compute first, third, or fifth order aberration coefficients.	256
Zoom Aberrations... Compute chromatic, third order or fifth order aberrations as well as lens sensitivity for surface groups that comprise a zoom/multiconfiguration lens system.	261
Other Aberrations... Compute Seidel wavefront, Zernike Wavefront, Aldis theorem surface contributions or axial gradient index Seidel aberrations.	264
Single Ray trace... Trace a single ray through the system.	269
Ray Fans... Trace a fan of rays across the system aperture from either a single field point or a series of field points.	272
Other Ray Analysis... Compute ray intercepts curves, optical path difference curves, f -theta analysis, field curves, chromatic analyses, ...etc.	280
Spot Diagram... Compute different spot diagram analyses at single or multiple field points.	286
Wavefront... Compute different wavefront analyses at single or multiple field points.	296
Transfer Function... Calculate through-focus or through-frequency Modulation Transfer Function analyses using different methods	304
Spread Function... Compute the point spread function for the current object point.	314
Energy Distribution... Compute the line spread function and knife edge distributions or encircled and ensquared energy distributions in the image.	321
Polarization... Compute reflectance, transmittance, pupil polarization state or Stokes parameters based on surface properties and polarization conditions	329
Autofocus... Automatically adjust the back focal distance to the best focus position	335

Paraxial setup

The paraxial properties of an optical system include such items as focal length, f -number, pupil location, magnification, Gaussian image size, etc. These and a number of related properties are shown in the output of the paraxial setup. Select the Show menu item and click Paraxial Setup to display the paraxial setup data.

*PARAXIAL SETUP OF LENS

APERTURE

Entrance beam radius:	6.250000	Image axial ray slope:	-0.124999
Object num. aperture:	6.2500e-20	F-number:	4.000043
Image num. aperture:	0.124999	Working F-number:	4.000043

FIELD

Field angle:	20.000000	Object height:	-3.6397e+19
Gaussian image height:	18.198709	Chief ray ims height:	18.154007

CONJUGATES

Object distance:	1.0000e+20	Srf 1 to prin. pt. 1:	13.429779
Gaussian image dist.:	43.080554	Srf 6 to prin. pt. 2:	-6.919987
Overall lens length:	17.000000	Total track length:	1.0000e+20
Paraxial magnification:	-5.0001e-19	Srf 6 to image srf:	42.950000

OTHER DATA

Entrance pupil radius:	6.250000	Srf 1 to entrance pup.:	10.466307
Exit pupil radius:	6.643768	Srf 6 to exit pupil:	-10.070166
Lagrange invariant:	-2.274814	Petzval radius:	-149.381547
Effective focal length:	50.000541		

The paraxial setup data is displayed under four major categories:

- **APERTURE**
- **FIELD**
- **CONJUGATES**
- **OTHER DATA**

A detailed description of each data item is given in the Paraxial Setup section of the Update chapter. The descriptions include an explanation of the actual method of calculation when appropriate for clarity.

APERTURE items are:

Entrance beam radius - is the radius of the axial beam of light that enters the optical system at surface 1, expressed in the units of the lens.

Object num. aperture - is the numerical aperture of the system in object space.

Image num. aperture - is the numerical aperture of the system in image space.

Image axial ray slope - is the paraxial axial ray slope in image space.

F-number - is the infinite conjugate f -number.

Working F-number - is the f -number of the system in image space, computed from the paraxial axial ray slope in image space.

Aperture in the context of paraxial setup refers to the size of the axial beam of light that enters the optical system.

FIELD items are:

Field angle - is the (half) angle that the object subtends from the entrance pupil, expressed in degrees.

Object height - is the height of the object, expressed in the units of the lens.

Gaussian image height - is the conjugate image height, expressed in the units of the lens. Note that the Gaussian image height may not be the height of the paraxial chief ray at the last surface of the lens for a focus shifted system.

Chief ray height - is the height of the paraxial chief ray, expressed in the units of the lens, on the image surface.

CONJUGATES items are

Object distance - is the distance from surface 0 to surface 1.

Srf 1 to prin. pt. 1 - is the distance from surface 1 to the first principal point.

Gaussian image dist. - is the distance from the next-to-last surface to the paraxial image plane. (The image may not be at the last surface in the lens.)

Srf *n* to prin. pt. 2 - is the distance from the next-to-last surface (surface *n*) to the second principal point.

Overall lens length - is the axial length of the optical system from surface 1 to the next-to-last surface.

Total track length - is the axial length of the optical system from surface 0 to the last surface.

Paraxial magnification - is the paraxial magnification computed from the ratio of the paraxial axial ray slope in object space to the ray slope in image space.

Srf *n* to image srf - is the distance from the next-to-last surface to the image surface (the last surface).

OTHER DATA items are

Entrance pupil radius - is the radius of the unaberrated entrance pupil, expressed in the units of the lens.

Srf 1 to entrance pup. - is the distance from the surface 1 to the unaberrated entrance pupil location, expressed in the units of the lens. The sign of the position is the specified relative to the local coordinate system of surface 1, according to the sign convention for thickness.

Exit pupil radius - is the radius of the unaberrated exit pupil, expressed in the units of the lens.

Srf *n* to exit pupil - is the distance from the next-to-last surface to the unaberrated exit pupil location, expressed in the units of the lens. The sign of the position is the specified relative to the local coordinate system of next-to-last surface, according to the sign convention for thickness.

Lagrange invariant - is the value of the Lagrange (paraxial) invariant, expressed in the units of the lens.

Petzval radius - is the radius of curvature of the Petzval surface, expressed in the units of the lens.

Effective focal length - is the effective focal length of the optical system, expressed in the units of the lens.

Paraxial Ray Analysis

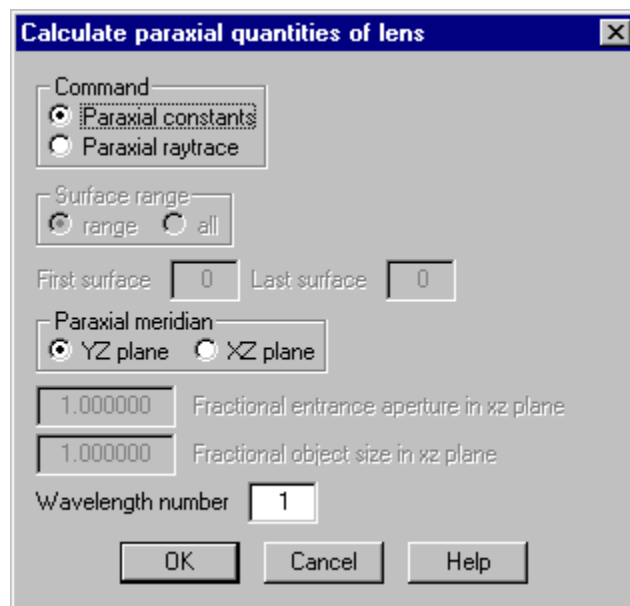
The paraxial region of an optical system is that area where the *small angle approximation* is valid, i.e., $\sin \theta \cong \tan \theta$. This is an area where the rays travel infinitesimally close to the optical axis. Although very few rays in a real optical system will actually be paraxial rays, the calculations of paraxial optics are important for several reasons. First, paraxial imagery is *ideal*, so paraxial calculations yield the properties of the perfect image, even for an imperfect optical system. Second, if a system is to have perfect behavior, the image locations will obey the laws of paraxial optics, so the goal of an optical design is to make the real, aberrated rays go where the paraxial rays go, not the other way around.

Strictly speaking, paraxial optics is only applicable to centered systems (i.e., systems that have an optical axis). If the system contains special surface data (e.g., tilted and/or decentered elements, holograms, etc.), the paraxial analysis will simply ignore the existence of the special data.

All of the paraxial analysis available in OSLO can be accessed by selecting Paraxial Analysis from the Evaluate menu, and choosing the desired option by clicking a radio button.

Paraxial constants

To see the paraxial constants, click the Paraxial constants radio button and then click OK. OSLO will compute and display the paraxial constants.



For focal systems, the values displayed are:

- Effective focal length
- Lateral (transverse) magnification
- Numerical aperture

- Gaussian image height
- Working f -number
- Petzval radius
- Lagrange (paraxial) invariant

***PARAXIAL CONSTANTS**

Effective focal length:	50.000541	Lateral magnification:	-5.0001e-19
Numerical aperture:	0.124999	Gaussian image height:	18.198709
Working F-number:	4.000043	Petzval radius:	-149.381547
Lagrange invariant:	-2.274814		

For afocal systems, the values displayed are:

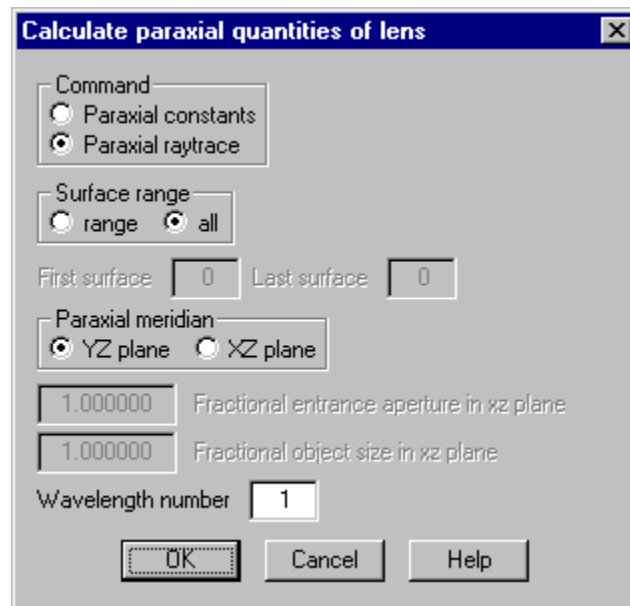
- Angular magnification
- Lagrange (paraxial) invariant
- Eye relief (distance from next-to-last surface to the exit pupil)
- Petzval radius

***PARAXIAL CONSTANTS**

Angular magnification:	1.087559	Lagrange invariant:	-2.274814
Eye relief:	-9.404273	Petzval radius:	-149.381547

Paraxial ray tracing

To see the results of a paraxial ray trace, click the Paraxial raytrace radio button.



If you want to see the data on each surface of the lens, click the “Surface rang” = “all” radio button. To select a range of surfaces to display, click the “Surface range” = “range” radio button, and type in the desired first and last surfaces in the range in the “First surface” and “Last surface” cells. The default is to print the data for all surfaces. After selecting the desired surface option, click OK to perform the paraxial ray trace.

OSLO will trace two paraxial rays through the system, and display the ray trace data computed on the requested surfaces. The two rays traced are the *axial* ray,

also called the *a*-ray or the *marginal* ray, and the *chief* ray, also called the *b*-ray or the *principal* ray. The axial ray is a ray from the center of the object surface through the edge of the paraxial entrance pupil. The chief ray is a ray from the edge of the object surface through the center of the paraxial entrance pupil. For each surface, six pieces of data are displayed:

PY, **PU**, and **PI** - the ray height, ray slope (after refraction or reflection), and angle of incidence, respectively, of the axial ray.

PYC, **PUC**, and **PIC** - the ray height, ray slope (after refraction or reflection), and angle of incidence, respectively, of the chief ray.

The paraxial ray trace in OSLO is correct for gradients that are only radial or axial. This affects the computation of paraxial constants, the paraxial ray trace, paraxial setup, **pwr** operand components, and the Gaussian beam spreadsheet. Solves in gradient index systems are supported. Note that the computation of aberration coefficients does *not* include the effects of the gradient index.

***PARAXIAL TRACE**

SRF	PY	PU	PI	PYC	PUC	PIC
0	0.0	6.2500e-20	6.2500e-20	-3.6397e+19	0.363970	0.363970
1	6.250000	-0.112609	0.294118	-3.809424	0.293253	0.184703
2	6.024781	-0.206034	-0.150585	-3.222919	0.487793	0.313567
3	4.788578	-0.037255	-0.442507	-0.296163	0.296163	0.502418
4	4.751323	0.091568	0.208927	-1.2212e-15	0.478775	0.296163
5	5.300729	0.042141	0.129095	2.872649	0.287679	0.499112
6	5.385011	-0.124999	-0.269402	3.448006	0.342398	0.088199
7	0.016319	-0.124999	-0.124999	18.154007	0.342398	0.342398

Paraxial meridian

The paraxial calculations can be performed in either the yz or the xz meridian. The desired meridian is selected by clicking the radio button for the option. The default is the yz meridian. If you select the xz meridian, you can enter values for the fractional entrance aperture and fractional object size in the xz azimuth. These values do not change the lens data, but are just used as defining parameters for the paraxial ray trace. The fractional entrance aperture is the aperture size in the xz azimuth as a fraction of the current entrance beam radius. The fractional object size is the size of the xz object as a fraction of the current defined object height. The entrance beam radius and the object height can be specified in the paraxial setup spreadsheet (see Chapter 4). If the paraxial ray trace is performed in the xz meridian, the ray heights for the axial and chief rays are denoted by PX and PXC, respectively.

Zoom Lens Parameters

Std **Prm**

This menu option performs two distinct analyses and displays the results in the current text window:

1. It performs a first order analysis for multiconfiguration (zoom) systems, and
2. It summarizes the air spacing between zooming groups in different configurations

First Order Analysis - The first part of this analysis summarizes which surfaces make up the different zooming groups in the lens model. You do not have to previously define the different zooming groups, OSLO will automatically calculate what surfaces are contained in a zooming group by looking at the changing lens spacings. OSLO determines that the surfaces included in each zooming group must lie between surfaces that change from zoom position (configuration) to zoom position (configuration).

The second part of this analysis calculates the following first order parameters and reports them for each group:

POWER - 1/EFL
EFL - Effective Focal Length
FNP - First Nodal Point
SNP - Second Nodal Point

FF - Front Focus

BF - Back Focus

First order parameters are then summarized for each zoom position (configuration) as a whole:

EFL - Effective Focal Length

IMAGE DISTANCE - Distance between the next-to-last surface and the last surface (image surface).

EFFECTIVE F/# - F/# of the lens at the used object and image conjugates

INFINITY F/# - F/# of the lens with the object distance at infinity

IMAGE HEIGHT - Height of the paraxial chief ray on the image surface

FIELD ANGLE - Displayed if all configurations have infinite object distances

OBJECT HEIGHT - Displayed if at least one of the zoom positions has a finite object distance

MAG - Magnification

See *Evaluate>>Paraxial Setup* (page 249) for a further description of each term.

***ZOOM LENS DATA**

Group 1 consists of surf 1 to 5
 Group 2 consists of surf 6 to 7
 Group 3 consists of surf 8 to 10
 Group 4 consists of surf 11 to 18

MAGNIFICATION	CFG1	CFG2	CFG3
GRP1	-1.670e-18	-1.670e-18	-1.670e-18
GRP2	2.4290	1.8426	1.7755
GRP3	-0.3071	-0.6857	-1.0210
GRP4	-0.8026	-0.8017	-0.8002

	POWER	EFL	FNP	SNP	FF	BF
GRP1	0.0073	136.9751	1.4211	-9.1141	-135.5540	127.8610
GRP2	-0.0048	-209.6721	1.3120	-0.5735	210.9841	-210.2456
GRP3	-0.0149	-67.1874	2.8117	-0.8234	69.9991	-68.0108
GRP4	0.0175	57.0488	-9.8033	-31.9002	-66.8521	25.1486

CFG	EFL	IMAGE DISTANCE	EFFECTIVE f/#	INFINITY f/#	IMAGE HEIGHT	FIELD ANGLE	MAG
1	81.9928	70.9332	4.4996	4.4996	21.9699	15.0000	-9.999e-19
2	138.7311	70.8829	4.5071	4.5071	21.9728	9.0000	-1.692e-18
3	198.6883	70.7998	4.5159	4.5159	21.5845	6.2000	-2.423e-18

Zoom Spacing Analysis - The first part of this analysis summarizes which surfaces make up the different zooming groups in the lens model. The thickness of each group is also reported. The reported thickness does include the thickness of the last surface of the group (the spacing to the beginning of the next group).

The second part of this analysis summarizes the spacings between the different zooming groups for each configuration.

***GROUP THICKNESSES AND AIR SPACES FOR ZOOMING SYSTEMS**

Group 1 consists of surf 1 to 5 Thickness = 16.138115

```

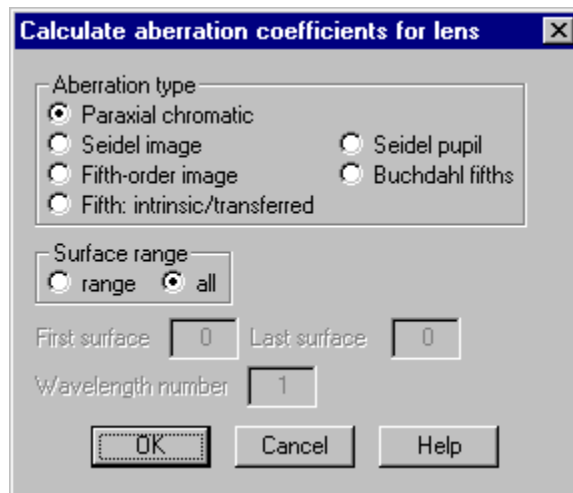
Group 2 consists of surf 6 to 7 Thickness = 2.868998
Group 3 consists of surf 8 to 10 Thickness = 6.455246
Group 4 consists of surf 11 to 18 Thickness = 36.430129

```

CFG	OBJ<->GRP1	GRP1<->GRP2	GRP2<->GRP3	GRP3<->GRP4	GRP4<->IMS
1	8.200e+19	3.1969	10.2464	49.2945	70.9332
2	8.200e+19	30.6675	8.1090	23.9367	70.8829
3	8.200e+19	34.9710	26.2134	1.5370	70.7998

Aberration coefficients

If the real optical system exhibits a deviation from the predictions of paraxial optics, the deviation is called an *aberration*. Aberration theory considers these defects in optical imagery to be described by a polynomial expansion in which the ray displacement from an ideal image point, or the wavefront displacement from an ideal reference sphere, is written in terms of object and pupil coordinates. The expansion coefficients can be computed using only paraxial ray data. The accuracy of predictions made on the basis of aberration theory depends on the order of the expansion polynomial and the particular system under study. OSLO computes five groups of aberrations, called *paraxial chromatic*, *Seidel image*, *Seidel pupil*, *fifth-order image*, and *Buchdahl fifth-order*.



All of the aberration analysis available in OSLO can be accessed by selecting Aberration Analysis from the Evaluate menu, and choosing the desired option by clicking a radio button. The aberration surface contributions will be printed for all surfaces (the default condition) if you click the all radio button. If you wish to print out the data for a range of surfaces, click the range radio button and type in the desired first and last surfaces of the range in the First surface and Last surface cells. The form of the output depends on the setting of the aberration mode (**amo**) general operating condition. (See Chapter 4 for a discussion of how to set operating conditions.) In the case of a focal system, the mode is normally set to *transverse*, in which case the contributions and sums are converted to transverse displacements in the paraxial image plane. For an afocal system, the aberration mode is normally set to *angular*, in which case the contributions and sums are angular aberrations. If the aberration mode is set to *unconverted*, the surface contributions and sums are printed directly, without conversion.

There are also fifth order intrinsic/transferred aberrations available. These are variations of the Buchdahl-format fifth-order image aberrations (MU1 through MU12), with two lines of output displayed for each surface contribution. For each coefficient, the first line displays the intrinsic surface contribution (INT) and the second line displays the transferred (induced) surface contribution (XFR). Select the type of aberration to calculate using Calculate >> Aberrations.

Chromatic aberrations

The chromatic aberrations are paraxial (i.e., first order) aberrations, and are caused by the fact that the refractive index of optical materials changes with wavelength. (Thus, there are no chromatic aberrations in an all-reflective system.) In OSLO, four chromatic aberration terms are computed for each surface:

PAC - primary axial chromatic aberration

SAC - secondary axial chromatic aberration

PLC - primary lateral chromatic aberration

SLC - secondary lateral chromatic aberration

The axial chromatic aberrations are often referred to as primary and secondary axial color. Similarly, the lateral chromatic aberrations are often referred to as primary and secondary lateral color. Axial color is essentially a measure of the change in image position with wavelength. Lateral color is essentially a measure of the change in image size with wavelength. OSLO uses the convention that the middle wavelength is wavelength 1, the short wavelength is wavelength 2, and the long wavelength is wavelength 3. The primary color aberrations are computed using wavelengths 2 and 3, while the secondary color aberrations are computed using wavelengths 2 and 1.

*CHROMATIC ABERRATIONS

SRF	PAC	SAC	PLC	SLC
1	-0.093340	-0.064858	-0.058617	-0.040731
2	-0.074647	-0.051870	0.155440	0.108010
3	0.176527	0.124657	-0.200427	-0.141534
4	0.133689	0.094406	0.189510	0.133824
5	-0.034747	-0.024144	-0.134339	-0.093347
6	-0.119366	-0.082943	0.039079	0.027154
SUM	-0.011884	-0.004752	-0.009354	-0.006623

Seidel image aberrations

Seidel aberrations are third-order aberrations that are present in monochromatic and polychromatic systems. They are called "third-order" aberrations because they are terms of third degree in the power series expansion of the ray displacement error:

SA3 - third-order spherical aberration

CMA3 - third-order coma

AST3 - third-order astigmatism

PTZ3 - third-order Petzval curvature of field

DIS3 - third-order distortion***SEIDEL ABERRATIONS**

SRF	SA3	CMA3	AST3	PTZ3	DIS3
1	-0.150292	-0.094382	-0.059271	-0.372951	-0.271431
2	-0.195919	0.407967	-0.849522	-0.049954	1.873006
3	0.686332	-0.779255	0.884759	0.389878	-1.447211
4	0.248488	0.352242	0.499317	0.409069	1.287675
5	-0.023167	-0.089569	-0.346294	-0.056108	-1.555782
6	-0.619859	0.202934	-0.066438	-0.458502	0.171859
SUM	-0.054416	-6.2549e-05	0.062550	-0.138567	0.058116

In a centered system, we can denote the fractional object height by h ($0 \leq h \leq 1$), and the fractional pupil coordinates (in cylindrical coordinates) by ρ and θ ($0 \leq \rho \leq 1$). For a paraxial ray, the fractional coordinates will be the same in object and image space. Because of aberrations, a real ray will intersect the paraxial image plane at a point displaced from the ideal image point by an amount $(\varepsilon_y, \varepsilon_x)$. The rotational symmetry of the system implies that the ray displacement can be written, using a power series of up to third-order, as

(8.1)

$$\varepsilon_{3y} = (\text{SA3})\rho^3 \cos\theta + (\text{CMA3})h\rho^2(2 + \cos 2\theta) \\ + (3\text{AST3} + \text{PTZ3})h^2\rho \cos\theta + (\text{DIS3})h^3$$

and

(8.2)

$$\varepsilon_{3x} = (\text{SA3})\rho^3 \sin\theta + (\text{CMA3})h\rho^2 \sin 2\theta \\ + (\text{AST3} + \text{PTZ3})h^2\rho \sin\theta$$

Seidel pupil aberrations

Just as the image is an aberrated version of the object, the exit pupil is an aberrated version of the entrance pupil. The Seidel pupil aberrations are reported as

PSA3 - third-order spherical aberration of the pupil**PCM3** - third-order coma of the pupil**PAS3** - third-order astigmatism of the pupil**PDS3** - third-order distortion of the pupil

***SEIDEL PUPIL ABERRATIONS**

SRF	PSA3	PCM3	PAS3	PDS3
1	-0.034729	-0.055301	-0.088061	0.076580
2	0.372804	-0.179032	0.085977	-0.050047
3	-0.033252	0.029287	-0.025794	0.148078
4	1.2083e-16	8.5238e-17	6.0131e-17	-0.105350
5	0.314797	0.081422	0.021060	0.010745
6	-0.016955	0.051790	-0.158191	-0.028081
SUM	0.602665	-0.071835	-0.165009	0.051926

Petzval curvature of the pupil is equivalent to image Petzval curvature. Usually, correction of the pupil aberrations is not as important as the image aberrations, since the pupil aberrations do not have a direct influence on image quality. (They do, however, affect the higher order aberrations.)

Fifth-order image aberrations

The fifth-order image aberrations reported with this option are the five fifth-order analogs of the Seidel image aberrations discussed above, i.e.,

SA5 - fifth-order spherical aberration

CMA5 - fifth-order coma

AST5 - fifth-order astigmatism

PTZ5 - fifth-order Petzval curvature of field

DIS5 - fifth-order distortion

Also reported is

SA7 - seventh-order spherical aberration

***FIFTH-ORDER ABERRATIONS**

SRF	SA5	CMA5	AST5	PTZ5	DIS5	SA7
1	-0.007447	-0.006273	0.003883	0.021429	0.015896	-0.000435
2	-0.023247	0.047504	-0.118851	-0.007709	0.163266	-0.002501
3	0.089959	-0.119939	0.063246	0.005054	-0.070854	0.011654
4	0.054432	0.056651	0.069632	0.042439	0.256930	0.013547
5	-0.009412	-0.022622	-0.076302	-0.014607	-0.259852	-0.003051
6	-0.067039	0.056010	0.027552	0.034110	-0.065633	-0.006597
SUM	0.037247	0.011332	-0.030841	0.080717	0.039752	0.012616

Buchdahl fifth-order aberrations

In general, there are twelve coefficients in the fifth-order aberration polynomials. Using the fractional coordinates (h , ρ , θ) defined in the above section and the notation of H. A. Buchdahl, the fifth-order aberrations are

(8.3)

$$\begin{aligned}\varepsilon_{5y} = & (\text{MU1})\rho^5 \cos\theta + [\text{MU2} + (\text{MU3})\cos 2\theta]h\rho^4 \\ & + [\text{MU4} + (\text{MU6})\cos^2 \theta]h^2\rho^3 \cos\theta \\ & + [\text{MU7} + (\text{MU8})\cos 2\theta]h^3\rho^2 \\ & + (\text{MU10})h^4\rho \cos\theta + (\text{MU12})h^5\end{aligned}$$

and

(8.4)

$$\begin{aligned}\varepsilon_{5x} = & (\text{MU1})\rho^5 \sin\theta + (\text{MU3})h\rho^4 \sin 2\theta \\ & + [\text{MU5} + (\text{MU6})\cos^2 \theta]h^2\rho^3 \sin\theta \\ & + (\text{MU9})h^3\rho^2 \sin 2\theta + (\text{MU11})h^4\rho \sin\theta\end{aligned}$$

The Buchdahl fifth-order option calculates all of the MU*i* coefficients.

*BUCHDAHL ABERRATIONS						
SRF	MU1/2	MU3/4	MU5/6	MU7/8	MU9/10	MU11/12
1	-0.007447	-0.006273	-0.014325	0.008361	0.012365	0.025312
	-0.010950	-0.022204	0.015899	0.017357	0.040842	0.015896
2	-0.023247	0.047504	-0.088785	0.340103	0.138868	-0.126560
	0.080659	-0.220496	-0.109156	0.247087	-0.601965	0.163266
3	0.089959	-0.119939	0.144312	-0.305940	-0.116294	0.068300
	-0.199827	0.364574	0.127712	-0.187016	0.321283	-0.070854
4	0.054432	0.056651	0.075244	0.069791	0.055634	0.112071
	0.105617	0.168161	-0.047422	0.039374	0.390597	0.256930
5	-0.009412	-0.022622	-0.053115	0.012616	-0.026063	-0.090909
	-0.045744	-0.133757	0.020923	-0.017771	-0.396115	-0.259852
6	-0.067039	0.056010	-0.006753	-0.146804	-0.082469	0.061662
	0.086848	-0.024369	0.074933	-0.102880	0.171868	-0.065633
SUM	0.037247	0.011332	0.056578	-0.021873	-0.017960	0.049876
	0.016604	0.131909	0.082889	-0.003848	-0.073489	0.039752

The relationships between the MU*i* coefficients and the fifth-order aberrations as displayed in the previous section are described in the following table.

SA5	MU1
CMA5	MU3
AST5	(MU10 – MU11)/4
PTZ5	(5 MU11 – MU10)/4
DIS5	MU12

The MU_i coefficients that do not have third-order analogs are associated with the aberrations that are usually called *oblique spherical aberration* (MU4, MU5, and MU6) and *elliptical coma* (MU7, MU8, and MU9).

Zoom Aberrations

Std **Prm**

Chromatic...
Third-Order...
Fifth-Order...
Zoom Lens Sensitivity...

In addition to the Evaluate>>Zoom Lens Parameters analysis (see page 254), there are several other analyses that are tailored for zoom or multi-configuration lens systems. Note that many of these analyses have counterparts that work for non-zoom systems.

The output for each of these analyses is displayed in the current text window.

The first part of each of these analyses summarizes which surfaces make up the different zooming groups in the lens model.

Each analysis has the option of being presented in two different layout formats:

- **Standard output (by group)** - The output is primarily grouped by the different zooming groups and then by the individual data items.
- **Alternate output (by data item)** - The output is primarily grouped by the individual data items and then by the different zooming groups.

Chromatic

This menu option calculates the chromatic aberration coefficients for the different zooming groups in a zoom system. The chromatic coefficients that are presented are:

PAC -Paraxial Axial Color

SAC - Secondary Axial Color

PLC - Primary Lateral Color

SLC - Secondary Lateral Color

See the section “Chromatic aberrations” on page 257 for a further discussion of each term.

*VARIATION OF THE 1st ORDER CHROMATIC ABERRATIONS BY ZOOMING

Group 1 consists of surf 1 to 5
Group 2 consists of surf 6 to 7
Group 3 consists of surf 8 to 10
Group 4 consists of surf 11 to 18

	PAC	SAC	PLC	SLC
CFG1				
GRP 1	-0.015589	-0.005157	0.036127	0.011712
GRP 2	0.045491	0.023032	-0.094286	-0.047736
GRP 3	-0.013852	-0.012999	0.046247	0.033565
GRP 4	-0.035360	-0.003937	-0.014978	-0.012473
SUM	-0.019310	0.000939	-0.026890	-0.014932
CFG2				
GRP 1	-0.044555	-0.014739	0.068831	0.022531
GRP 2	0.078940	0.039966	-0.080882	-0.040949
GRP 3	-0.027999	-0.024307	0.043022	0.030269
GRP 4	-0.035252	-0.003920	-0.015000	-0.012486
SUM	-0.028865	-0.002999	0.015971	-0.000635

CFG3

GRP 1	-0.091209	-0.030172	0.090726	0.029777
GRP 2	0.147598	0.074727	-0.103541	-0.052421
GRP 3	-0.043759	-0.036805	0.032841	0.021770
GRP 4	-0.035101	-0.003894	-0.014768	-0.012283
SUM	-0.022471	0.003856	0.005258	-0.013157

Third-Order

This menu item calculates the Seidel (third-order) aberration coefficients for the different zooming groups in a zoom system. The Seidel (third-order) coefficients that are presented are:

SA3 - Spherical aberration

CMA3 - Coma

AST3 - Astigmatism

PTZ3 - Petzval field curvature

DIS3 - Distortion

See the section "Seidel image aberrations" on page 257 for a further discussion of each term.

*VARIATION OF THE 3rd ORDER SEIDEL COEFFICIENTS BY ZOOMING

Group 1 consists of surf 1 to 5
 Group 2 consists of surf 6 to 7
 Group 3 consists of surf 8 to 10
 Group 4 consists of surf 11 to 18

	SA3	CMA3	AST3	PTZ3	DIS3
CFG1					
GRP 1	-0.009799	0.047225	-0.356875	-0.140645	2.253330
GRP 2	0.012778	-0.069940	0.362560	0.084058	-1.632687
GRP 3	0.051827	0.039092	0.190044	0.257101	-1.719034
GRP 4	-0.187607	-0.002544	-0.197986	-0.221023	0.277721
SUM	-0.132801	0.013833	-0.002256	-0.020509	-0.820670
CFG2					
GRP 1	-0.079910	0.193417	-0.597242	-0.140450	1.924527
GRP 2	0.070748	-0.184862	0.430065	0.083941	-0.895437
GRP 3	0.094282	0.003400	0.372848	0.256744	-1.070639
GRP 4	-0.186217	-0.002450	-0.197749	-0.220716	0.278147
SUM	-0.101097	0.009504	0.007921	-0.020480	0.236599
CFG3					
GRP 1	-0.334229	0.472879	-0.793371	-0.135263	1.441106
GRP 2	0.276481	-0.414066	0.565312	0.080841	-0.727163
GRP 3	0.175927	-0.046083	0.426769	0.247263	-0.438773
GRP 4	-0.184372	-0.002260	-0.190507	-0.212565	0.264151
SUM	-0.066193	0.010469	0.008203	-0.019724	0.539322

Fifth-Order

This menu item calculates the fifth-order aberration coefficients for the different zooming groups in a zoom system. The fifth-order coefficients that are presented are:

SA5 - fifth-order spherical aberration

CMA5 - fifth-order coma

AST5 - fifth-order astigmatism

PTZ5 - fifth-order Petzval curvature of field

DIS5 - fifth-order distortion

Also reported is

SA7 - seventh-order spherical aberration

See the section "Fifth-order image aberrations" on page 259 for a further discussion of each term.

***VARIATION OF THE 5th ORDER SEIDEL COEFFICIENTS BY ZOOMING**

Group 1 consists of surf 1 to 5
 Group 2 consists of surf 6 to 7
 Group 3 consists of surf 8 to 10
 Group 4 consists of surf 11 to 18

	SA5	CMA5	AST5	PTZ5	DIS5	SA7
CFG1						
GRP 1	0.000005	0.000204	-0.038895	-0.006842	0.221640	0.000001
GRP 2	0.000092	-0.000657	0.043688	0.005630	-0.214449	0.000000
GRP 3	-0.005154	0.014461	-0.003707	-0.017362	-0.021707	-0.000755
GRP 4	0.149034	-0.021935	0.003500	0.037854	0.024866	0.044117
SUM	0.143977	-0.007927	0.004585	0.019280	0.010350	0.043363
CFG2						
GRP 1	0.000110	0.001503	-0.041579	-0.008854	0.125785	0.000092
GRP 2	0.001351	-0.005224	0.028001	0.006989	-0.055773	0.000004
GRP 3	-0.021921	0.023601	0.013164	0.007048	-0.014470	-0.004470
GRP 4	0.147178	-0.022286	0.006455	0.004464	-0.034751	0.043537
SUM	0.126717	-0.002407	0.006041	0.009647	0.020791	0.039163
CFG3						
GRP 1	0.000936	0.005689	-0.034814	-0.006477	0.061967	0.001610
GRP 2	0.010647	-0.024989	0.025730	0.004928	-0.031739	0.000058
GRP 3	-0.058660	0.010227	0.008022	0.010074	0.014634	-0.015870
GRP 4	0.144935	-0.022597	0.007344	-0.005503	-0.013348	0.042111
SUM	0.097859	-0.031670	0.006282	0.003022	0.031514	0.027909

Zoom Lens Sensitivity

Calculates the sensitivity of zoom groups according to the theory presented by Besenmatter^{1,2}. Axial and Lateral sensitivity factors are reported. Up to 20 zoom groups are allowed.

***ZOOM SENSITIVITY ANALYSIS**

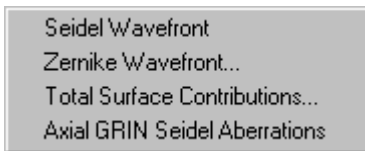
Group 1 consists of surf 1 to 5
 Group 2 consists of surf 6 to 7
 Group 3 consists of surf 8 to 10
 Group 4 consists of surf 11 to 18

1. Besenmatter, *Optik* 51 (1978) 147

2. W. Besenmatter, *Proc. SPIE*, Vol. 237, pp. 242-255 (1980)

	AXIAL	LATERAL
CFG1		
GRP 1	0.358318	0.598596
GRP 2	-0.297587	-0.352160
GRP 3	0.583359	-1.048988
GRP 4	0.355910	1.802552
CFG2		
GRP 1	1.025804	1.012820
GRP 2	-0.723672	-0.463155
GRP 3	0.340542	-1.351334
GRP 4	0.357326	1.801669
CFG3		
GRP 1	2.104076	1.450543
GRP 2	-1.436598	-0.633550
GRP 3	-0.027136	-1.617207
GRP 4	0.359659	1.800213

Other Aberrations



Seidel Wavefront

The method used to compute the Seidel wavefront aberration coefficients for the current wavefront have been improved.

To the level of the third-order (Seidel) aberrations, the wavefront aberration polynomial has the form (see the Optics Reference, p. 69)

$$W(h, \rho, \cos \theta) = W_{040} \rho^4 + W_{131} h \rho^3 \cos \theta + W_{222} h^2 \rho^2 \cos^2 \theta + W_{220} h^2 \rho^2 + W_{311} h^3 \rho \cos \theta \quad (8.5)$$

The subscripts on the W_{ijk} coefficients indicate the powers of h , ρ , and $\cos \theta$, respectively, for that term in the polynomial. To compute and display the five W_{ijk} coefficients for the current wavelength use Options >> General Analysis >> Seidel Wavefront Aberrations or the **seiwvf** CCL command. (The coefficients are computed from the Seidel sums, $S_I - S_V$; see Eq. (3.19) in the Optics Reference.)

```

*SEIDEL WAVEFRONT ABERRATION COEFFICIENTS - WAVELENGTH 1
COEFFICIENTS IN WAVELENGTHS
      W040      W131      W222      W220      W311
      2.894152   0.013307  -13.307079   8.085955  -12.363769

*SEIDEL WAVEFRONT ABERRATION COEFFICIENTS - WAVELENGTH 2
COEFFICIENTS IN WAVELENGTHS
      W040      W131      W222      W220      W311
      3.095667  -0.223414  -15.042924   9.930399  -15.465513

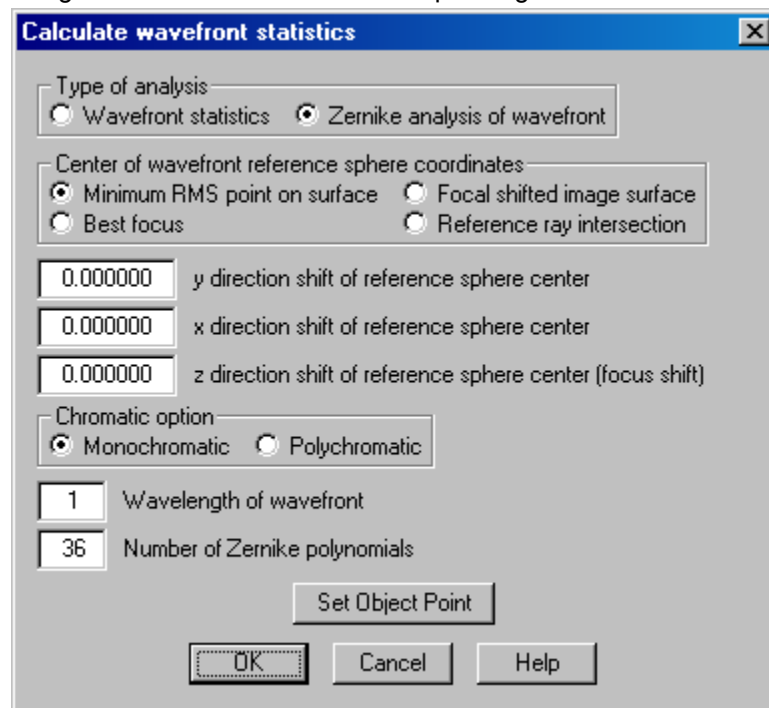
*SEIDEL WAVEFRONT ABERRATION COEFFICIENTS - WAVELENGTH 3
COEFFICIENTS IN WAVELENGTHS
      W040      W131      W222      W220      W311
      2.695815   0.075295  -12.253232   7.169171  -10.857336

```

If the “OPD in waves” general operating condition is on, the coefficients are displayed in wavelengths; otherwise the coefficients are in the current lens units. (Use the **opdw** command or “Update>>Operating Conditions>>General” to change the units.)

Zernike Wavefront

Select Wavefront Analysis from the Evaluate menu to display some data describing the wavefront for the current spot diagram.



Wavefront statistics

The wavelength of the wavefront is selected by typing the desired wavelength number in the Wavelength of wavefront cell. There are six pieces of data displayed for the wavefront. The first three are:

PKVAL OPD - the maximum peak-to-valley optical path difference in the spot diagram data.

RMS OPD - the root-mean-square value of the optical path differences in the spot diagram data.

STREHL RATIO - the value of the point spread function at the requested diffraction focus, relative to an unaberrated system.

There are four options available for locating the center of the reference sphere. These options and the additional data displayed are:

Minimum RMS point on surface

Click this radio button if you want to move the center of the reference sphere by specified y shift and x shift amounts from the point on the focal shifted (by an amount z shift) image surface that minimizes the variance of the wavefront. The specified **Y SHIFT**, **X SHIFT**, and focus shift (**RSZ**) are reported.

*WAVEFRONT

WAVELENGTH 1

PKVAL OPD	RMS OPD	STREHL RATIO	YSHIFT	XSHIFT	RSZ
0.551560	0.123537	0.555861	--	--	--

Focal shifted image surface

Click this radio button if you want to use the point on the focus shifted (by an amount z shift) image surface that minimizes the variance of the wavefront as the center of the reference sphere. **RSY**, **RSX**, and **RSZ** will be reported as the location of this point.

*WAVEFRONT RS

WAVELENGTH 1

PKVAL OPD	RMS OPD	STREHL RATIO	RSY	RSX	RSZ
0.355818	0.135295	0.443876	--	--	0.061000

Best focus

Click this radio button if you want to use the point in three-dimensional space that minimizes the variance of the wavefront as the center of the reference sphere. **RSY**, **RSX**, and **RSZ** will be reported as the location of this point.

*WAVEFRONT BF

WAVELENGTH 1

PKVAL OPD	RMS OPD	STREHL RATIO	RSY	RSX	RSZ
0.175274	0.044911	0.922647	--	--	0.028968

Reference ray intersection

Click this radio button if you want the evaluation performed with the reference sphere centered at a point shifted by prescribed amounts y shift, x shift and z shift from the intersection of the reference ray with the image surface. The specified **Y SHIFT**, **X SHIFT**, and focus shift (**RSZ**) are reported.

*WAVEFRONT REF
WAVELENGTH 1

PKVAL OPD	RMS OPD	STREHL RATIO	YSHIFT	XSHIFT	RSZ
0.551560	0.123537	0.555861	--	--	--

Click the OK button to compute the wavefront statistics after you have specified the desired option.

Zernike analysis

Click the Zernike analysis of wavefront radio button if you wish to perform a Zernike decomposition of the wavefront. You select the wavelength and center of the reference sphere in the same manner as for the wavefront statistics calculation described above. In addition, you can specify the number of Zernike terms to be used in the analysis. The maximum number of Zernike terms is 36.

It is recommended that the equal image space ray increments spot diagram operating condition be active before performing a Zernike polynomial decomposition of the wavefront. The equal increment ray sampling increases the accuracy of the orthogonalization procedure used in the calculations.

The output consists of the coefficients of the various Zernike polynomials, for an expansion of the desired wavefront. The coefficients are in units of wavelengths, defined at the wavelength of the wavefront, if the **opdw** (See Chapter 4) general operating condition is on; otherwise the coefficients are in the current lens units. Note that, as discussed in the definitions of the Zernike sag and phase surfaces in Chapter 4, the azimuthal angle (denoted by **A** in the OSLO output below) is measured from the *y*-axis.

*ZERNIKE ANALYSIS
WAVELENGTH 1

```

-0.319936: [0] 1
--      : [1] RCOSA
--      : [2] RSINA
-0.187489: [3] 2R^2 - 1
--      : [4] R^2COS2A
--      : [5] R^2SIN2A
--      : [6] (3R^2 - 2)RCOSA
--      : [7] (3R^2 - 2)RSINA
0.027369: [8] 6R^4 - 6R^2 + 1
--      : [9] R^3COS3A
--      : [10] R^3SIN3A
--      : [11] (4R^2 - 3)R^2COS2A
--      : [12] (4R^2 - 3)R^2SIN2A
--      : [13] (10R^4 - 12R^2 + 3)RCOSA
--      : [14] (10R^4 - 12R^2 + 3)RSINA
-0.113198: [15] 20R^6 - 30R^4 + 12R^2 - 1
--      : [16] R^4COS4A
--      : [17] R^4SIN4A
--      : [18] (5R^2 - 4)R^3COS3A
--      : [19] (5R^2 - 4)R^3SIN3A
--      : [20] (15R^4 - 20R^2 + 6)R^2COS2A
--      : [21] (15R^4 - 20R^2 + 6)R^2SIN2A
--      : [22] (35R^6 - 60R^4 + 30R^2 - 4)RCOSA
--      : [23] (35R^6 - 60R^4 + 30R^2 - 4)RSINA

-0.008655: [24] 70R^8 - 140R^6 + 90R^4 - 20R^2 + 1
--      : [25] R^5COS5A
--      : [26] R^5SIN5A

```

```

--      : [27] (6R^2 - 5)R^4COS4A
--      : [28] (6R^2 - 5)R^4SIN4A
--      : [29] (21R^4 - 30R^2 + 10)R^3COS3A
--      : [30] (21R^4 - 30R^2 + 10)R^3SIN3A
--      : [31] (56R^6 - 105R^4 + 60R^2 - 10)R^2COS2A
--      : [32] (56R^6 - 105R^4 + 60R^2 - 10)R^2SIN2A
--      : [33] (126R^8 - 280R^6 + 210R^4 - 60R^2 + 5)RCOSA
--      : [34] (126R^8 - 280R^6 + 210R^4 - 60R^2 + 5)RSINA
-0.000567: [35] 252R^10 - 630R^8 + 560R^6 - 210R^4 + 30R^2 - 1
-3.3199e-05: [36] 924R^12 - 2772R^10 + 3150R^8 - 1680R^6 + 420R^4 - 42R^2 + 1

```

Total Surface Contributions

The Aldis theorem allows you to compute the displacement of a real ray as a sum of surface contributions, similar to aberration theory. The Aldis theorem, however, does not have the generality of aberration theory; it only gives results for the actual ray traced. Nevertheless, by comparing results with those from a third-order analysis, you can get a good idea of the contribution of a particular surface to the higher-order aberrations.

The Aldis Theorem command prompts for the coordinates of a ray to be traced, then produces the following output (for trip.len):

```

*ALDIS
TOTAL RAY ABERRATION FOR FBY = 0.500000 FY = 1.000000 FX = --
SRF      DELTA Y      DELTA X      SEIDEL DY      SEIDEL DX
1        -0.470607      --        -0.463486      --
2        -2.6856e-05      --         0.000528      --

3         0.107088      --         0.097587      --
4         1.588636      --         1.414566      --

5        -0.694247      --        -0.625740      --
6        -0.443803      --        -0.458429      --

SUM       0.087041      --        -0.034974      --
DYref     0.100797

```

Additional information on the Aldis theorem can be found in Chapter 3 – Monochromatic Aberrations of the Optics Reference manual.

Axial GRIN Seidel Aberrations

The **grinsei** CCL command is an implementation of Sands' equations from "Inhomogeneous Lenses, IV. Aberrations of Lenses with Axial Index Distributions" (JOSA Vol. 61, No. 8, August 1971). It is particularly wired for the Gradium™ material and is not generalized for other gradient types.

The output format closely duplicates the OSLO **sei** command. For a purely homogeneous optical system, it should return exactly the same aberration coefficients as OSLO's **sei** command. If any Gradium gradients are encountered, the **grinsei** command writes out a line beginning with "IS" for "Inhomogeneous Surface" and prints out the gradient surface contributions. On the second surface of a Gradium lens, the command writes out a line beginning with "IT" for "Inhomogeneous Transfer" and prints out the transfer contributions.

At the bottom of the aberration list, the command writes out the total third-order contributions, then two additional lines containing the sum of the homogeneous contributions and the sum of the gradient contributions.

The only approximation involved is in the transfer contribution. According to Sands, you need to do an integral over the gradient profile. However for axial gradients, the transfer contribution is usually the less significant of the two and so we can obtain a good estimate of the transfer contribution by simply assuming that the profile is linear between the surface vertices and doing the integral in closed form.

Single Ray trace

Once a reference ray has been traced, select Trace Ray from the Evaluate menu to trace a single ray through the system.

There are two forms of output available for the ray data. If you click the Standard radio button, six items of data will be printed for each surface:

Y, X, Z - the y, x, and z coordinates of the intersection of the ray with the surface.

YANG, XANG - the angles, in degrees, between the z-axis and the projection of the ray on the yz and xz planes, respectively. The reported angles refer to the ray after refraction/reflection/diffraction at the surface.

D - the distance along the ray between the previous surface and the current surface.

If you click the Full radio button, in addition to the above data, a second line of data will be printed for each surface:

L, K, M - the direction cosines of the ray with respect to the y, x, and z axes, respectively. The direction cosines refers to the ray after refraction/reflection/diffraction at the surface.

IANG, RANG - the angles, in degrees, of incidence and refraction or reflection of the ray with the surface.

OPL - the optical path length along the ray up to the current surface.

The ray data may be displayed in either local or global coordinates. If local coordinates are used, the surface-by-surface output data is expressed in the local coordinate system of each surface. If global coordinates are used, the output data is expressed in the coordinate system of a single global coordinate surface. The global coordinate surface may be specified by executing the command

gcs *Global_coordinate_surface*

The global coordinate surface is retained as a component of the lens data and in the absence of an explicit specification defaults to surface 1.

Click the All radio button if you want to see the data for all surfaces in the lens. If you click the Srf radio button, you can enter the first and last surfaces of the range for which the data is to be displayed in the cells labeled First surface number and Last surface number. The default is to display the data for the image surface only.

If the "Print surface numbers in spreadsheet buffer" option is used, the surface number for each ray will be printed, in floating point format, in column 7 (i.e., column *g*) of the output and spreadsheet buffer. This is useful for CCL and SCP commands that need to know the surface number for the ray data. For non-sequential groups and lenses with skip surfaces, the row number for the ray output may not correspond to the surface number.

If the "Include diffraction efficiency" option is used, the data in the sixth column (D) is replaced with the local diffraction efficiency (DIFF EFF) of the surface at the point of intersection of the ray. Note that this is the efficiency at that surface, *not* the cumulative efficiency. If the surface is not a diffractive surface, the efficiency is 1.0. The efficiency is computed using the extended scalar theory described in Chapter 10 of the Optics Reference manual.

*TRACE RAY - LOCAL COORDINATES

SRF	Y/L	X/K	Z/M	YANG/IANG	XANG/RANG	D/OPL
1	6.250000	--	0.939904	-6.647017	--	0.939904
	-0.115752	--	0.993278	17.104635	10.457618	0.939904
2	6.140314	--	-0.118870	-12.242819	--	0.947595
	-0.212055	--	0.977258	8.865119	14.460922	2.475397
3	4.945642	--	-0.613220	-1.816353	--	5.633775
	-0.031696	--	0.999498	26.379109	15.952642	8.109172
4	4.874640	--	0.625743	6.379722	--	2.240089
	0.111117	--	0.993807	12.813412	21.009487	11.730481
5	5.487457	--	0.106632	3.072255	--	5.515042
	0.053595	--	0.998563	8.606181	5.298713	17.245523
6	5.540134	--	-0.911908	-7.177534	--	0.982873
	-0.124944	--	0.992164	15.621930	25.871719	18.838180
7	0.016558	--	--	-7.177534	--	44.208335
	-0.124944	--	0.992164	7.177534	7.177534	63.046515
PUPIL	FY	FX				OPD
	1.000000	--				-0.602512

FY and FX are the fractional coordinates of the ray in object space. The default values for FY and FX are 0, i.e., a ray through the center of the pupil. The values for FY and FX may be directly input, or a ray from the optimization ray set may be chosen. If the ray has been traced to the image surface, the optical path difference (OPD) of the ray is also displayed.

If the Use polarization raytrace (**pzrt**) operating condition is On, another line of data will be displayed for each surface. The polarization state of the incident beam is defined by the polarization operating conditions. (See Chapter 4.) For each surface, the following information is displayed. Note that all of the data is calculated after the ray has been refracted/reflected/diffracted at the surface.

INTENSITY - the transmitted intensity of the ray, relative to an incident intensity of 1 for the reference ray.

DEG. POLRZ. - the degree of polarization of the ray. A value of 0 indicates completely unpolarized, while a value of 1 indicates completely polarized.

ELL. RATIO - the ratio of the minor axis of the polarization ellipse to the major axis of the polarization ellipse for the polarized portion of the field. A value of 0 indicates linear polarization, while a value of 1 indicates circular polarization.

ELL. ANGLE - the angle, in degrees, between the y-axis of the polarization ellipse xy plane and the major axis of the polarization ellipse. The polarization ellipse lies in the xy plane of a local, right-handed (x, y, z) coordinate system. The z-axis of this coordinate system lies along the ray. The x-axis is orthogonal to both the ray and the local y-axis of the surface. Given these two axes, the y-axis is defined such that x, y, and z form an orthogonal, right-handed coordinate system.

HANDEDNESS - the polarization handedness (right or left) of the ray.

The field has right-handed polarization (indicated by a value of +1.0 in the spreadsheet buffer) if the field vector rotates in a clockwise direction, as viewed from the direction to which the ray is propagating. Conversely, the polarization is left-handed (indicated by a value of -1.0 in the spreadsheet buffer) if the field vector rotates counter-clockwise.

***TRACE RAY - LOCAL COORDINATES**

SRF	Y	X	Z	YANG	XANG	D
	INTENSITY	DEG. POLRZ.	ELL. RATIO	ELL. ANGLE	HANDEDNESS	
1	6.250000	--	0.939904	-6.647017	--	0.939904
	0.950150	1.000000	--	--	--	
2	6.140314	--	-0.118870	-12.242819	--	0.947595
	0.901097	1.000000	--	--	--	
3	4.945642	--	-0.613220	-1.816353	--	5.633775
	0.864328	1.000000	--	--	--	
4	4.874640	--	0.625743	6.379722	--	2.240089
	0.824387	1.000000	--	--	--	
5	5.487457	--	0.106632	3.072255	--	5.515042
	0.779465	1.000000	--	--	--	
6	5.540134	--	-0.911908	-7.177534	--	0.982873
	0.746889	1.000000	--	-8.5377e-07	--	
7	0.016558	--	--	-7.177534	--	44.208335
	0.746889	1.000000	--	-8.5377e-07	--	
PUPIL	FY	FX				OPD
	1.000000	--				-0.602512

Ray Fans

Single Field Point...
2-Dimensional Field Variation...

Often it is more instructive to look at the results of tracing a prescribed pattern of rays, rather than just a single ray. The simplest pattern is just a series of equally spaced (in the pupil) rays, in a line parallel to either the y- or x-axis of the pupil. This is known as tracing a *fan* of rays through the system.

Single Field Point

A number of ray fan analyses can be performed at single object (field) points. The single object point to be used in the analysis can be chosen in advance by defining a new reference ray, or a new object point can be chosen by using the *Set Object Point* button at the bottom of the "Trace fans of rays through lens" dialog box.

Ray fans

You can trace a y-fan or x-fan through the lens by clicking the appropriate radio button in the spreadsheet.

There are two forms of output available. If you click the Standard radio button, six pieces of data will be reported for each ray.

FY (for a y-fan) or **FX** (for an x-fan) - fractional coordinate of the ray in object space.

DYA, DXA - the differences in the slopes of the ray and the reference ray, measured in the yz and xz planes.

DY, DX, DZ - the differences between the y, x, and z coordinates of the intersection of the ray with the image surface and the y, x, and z coordinates of the intersection of the reference ray with the image surface.

```
*TRACE FAN - FBY 0.00, FBX 0.00, FBZ 0.00
RAY    FY      FRAC RFS    DYA    DXA      OPD      DMD      OSC
  1    1.000000  0.974928  -0.125931  --    -0.602512  0.696079  0.000418
  2    0.900000  0.872573  -0.113379  --    -0.392509  0.667847  0.000245
```

3	0.800000	0.772008	-0.100725	--	-0.342700	0.595247	0.000138
4	0.700000	0.672877	-0.088042	--	-0.336449	0.498192	7.3934e-05
5	0.600000	0.574892	-0.075368	--	-0.319976	0.391446	3.6669e-05
6	0.500000	0.477820	-0.062725	--	-0.276594	0.286036	1.6312e-05
7	0.400000	0.381461	-0.050120	--	-0.210143	0.190195	6.1316e-06
8	0.300000	0.285644	-0.037552	--	-0.134211	0.110005	1.7219e-06
9	0.200000	0.190219	-0.025016	--	-0.065072	0.049827	2.5245e-07
10	0.100000	0.095047	-0.012502	--	-0.017114	0.012595	-1.0415e-08
11	--	--	-1.7347e-17	--	--	--	--
12	-0.100000	-0.095047	0.012502	--	-0.017114	0.012595	-1.0415e-08
13	-0.200000	-0.190219	0.025016	--	-0.065072	0.049827	2.5245e-07
14	-0.300000	-0.285644	0.037552	--	-0.134211	0.110005	1.7219e-06
15	-0.400000	-0.381461	0.050120	--	-0.210143	0.190195	6.1316e-06
16	-0.500000	-0.477820	0.062725	--	-0.276594	0.286036	1.6312e-05
17	-0.600000	-0.574892	0.075368	--	-0.319976	0.391446	3.6669e-05
18	-0.700000	-0.672877	0.088042	--	-0.336449	0.498192	7.3934e-05
19	-0.800000	-0.772008	0.100725	--	-0.342700	0.595247	0.000138
20	-0.900000	-0.872573	0.113379	--	-0.392509	0.667847	0.000245
21	-1.000000	-0.974928	0.125931	--	-0.602512	0.696079	0.000418

If you click the Optical path differences radio button, in place of DY, DX, and DZ, the data reported are:

OPD - the optical path difference between the ray and the reference ray.

DMD - the Conrady "*D minus d*" sum for the ray. (Displayed for fans in wavelength 1 only.)

OSC - if the system is focal, and the current object point is on-axis (i.e., $FBY = FBX = FBZ = 0$), the Offense against the *Sine Condition* (OSC) will also be displayed (see further explanation below).

*TRACE FAN - FBY		0.00, FBX	0.00, FBZ	0.00			
RAY	FY	FRAC RFS	DYA	DXA	OPD	DMD	OSC
1	1.000000	0.974928	-0.125931	--	-0.602512	0.696079	0.000418
2	0.800000	0.772008	-0.100725	--	-0.342700	0.595247	0.000138
3	0.600000	0.574892	-0.075368	--	-0.319976	0.391446	3.6669e-05
4	0.400000	0.381461	-0.050120	--	-0.210143	0.190195	6.1316e-06
5	0.200000	0.190219	-0.025016	--	-0.065072	0.049827	2.5245e-07
6	--	--	-6.9388e-18	--	--	--	--
7	-0.200000	-0.190219	0.025016	--	-0.065072	0.049827	2.5245e-07
8	-0.400000	-0.381461	0.050120	--	-0.210143	0.190195	6.1316e-06
9	-0.600000	-0.574892	0.075368	--	-0.319976	0.391446	3.6669e-05
10	-0.800000	-0.772008	0.100725	--	-0.342700	0.595247	0.000138
11	-1.000000	-0.974928	0.125931	--	-0.602512	0.696079	0.000418

The format of the output for the Print Y ray-fan and Print X ray-fan is as follows: the second column of the output contains the fractional coordinate of the ray on the reference surface (FRAC RFS). This value is the radial coordinate of the ray intersection with the reference surface $[(x^2 + y^2)^{1/2}]$ divided by the aperture radius of the reference surface.

For the fan, you can enter the minimum and maximum pupil points for the fan and an offset from the center of pupil. These values are all entered as fractional pupil coordinates. You can also specify the number of rays to be traced in a fan. All of these values are changed by selecting the appropriate cell and typing the desired value.

OSC is a dimensionless measure of the total amount of linear coma present in a lens. OSC only takes linear coma into account, that is coma that is proportional to the first power of the field coordinate; it is an exact measure in the aperture coordinate. The following definition of OSC is valid for rotationally symmetric systems.

(8.6)

$$\text{OSC} = \frac{\sin U}{u} \cdot \frac{u'}{\sin U'} \cdot \frac{l' - \bar{l}'}{L' - \bar{L}'} - 1$$

In the above equation, u and u' are the initial and final slopes of the paraxial ray, U and U' are the initial and final convergence angles of the real ray, l' and L' are the distances from the last lens surface to the intersections of the paraxial and real rays, respectively, with the optical axis, and \bar{l}' is the distance from the last lens surface to the exit pupil.

In the absence of spherical aberration ($l' = L'$), zero OSC is seen to be equivalent to satisfying the Abbe sine condition. Also, when spherical aberration is present,

OSC is a function of aperture stop position (through the \bar{l}' term), as would be expected from the Seidel aberration stop shift equations.

More details on OSC and derivations of the above equation may be found in Smith³, Kingslake⁴, and Welford⁵.

Ray-intercept curves

Rather than printing text output of ray fan data, you can display the results graphically. Click the Plot ray-intercept curves radio button if you want to plot the ray fans for the current object point. The curves may be plotted in just the current wavelength, in wavelengths 1, 2, and 3 only, or in all defined wavelengths. The default is to plot the curves in wavelengths 1, 2, and 3. This option is useful, for example, to avoid clutter in the ray-intercept curves for systems that are using more than three wavelengths in order to calculate more accurate polychromatic MTFs.

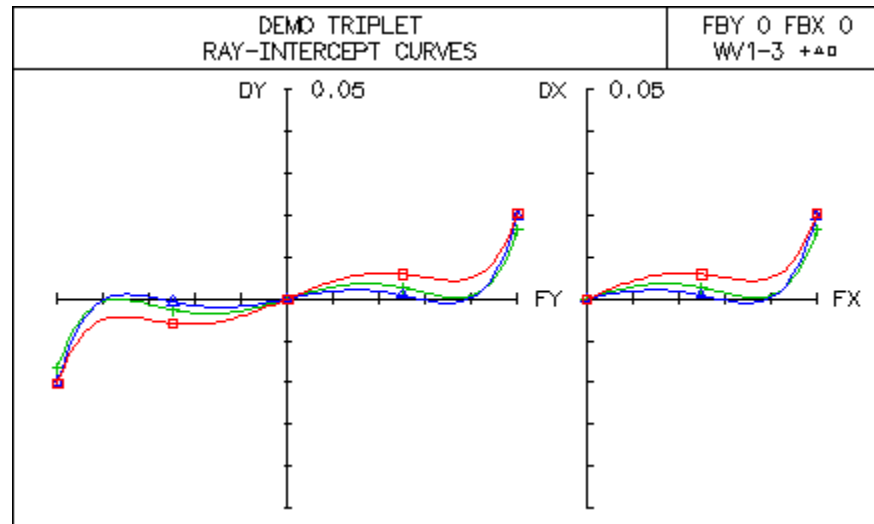
Both x and y-fans are traced from the current object point, and the resulting data is used to make ray displacement error versus fractional pupil coordinate curves. The plotted curves are the result of spline fits to the numerical ray data. For focal systems, the plotted data are ray displacements in linear measure; for afocal systems the plotted data are ray displacements in angular measure. Either the y-component of the ray aberration (DY or DYA) or the x-component (DX or DXA) may be plotted as a function of the y and x-pupil coordinates (FY and FX). The default is to plot DY vs. FY and DX vs. FX. For systems without meridional

3. W. J. Smith, *Modern Optical Engineering*, Second Edition, McGraw-Hill, 1990, pp. 303-304.

4. R. Kingslake, *Lens Design Fundamentals*, Academic Press, 1978, pp. 157-166.

5. W. T. Welford, *Aberrations of Optical Systems*, Adam Hilger, 1986, pp. 172-176.

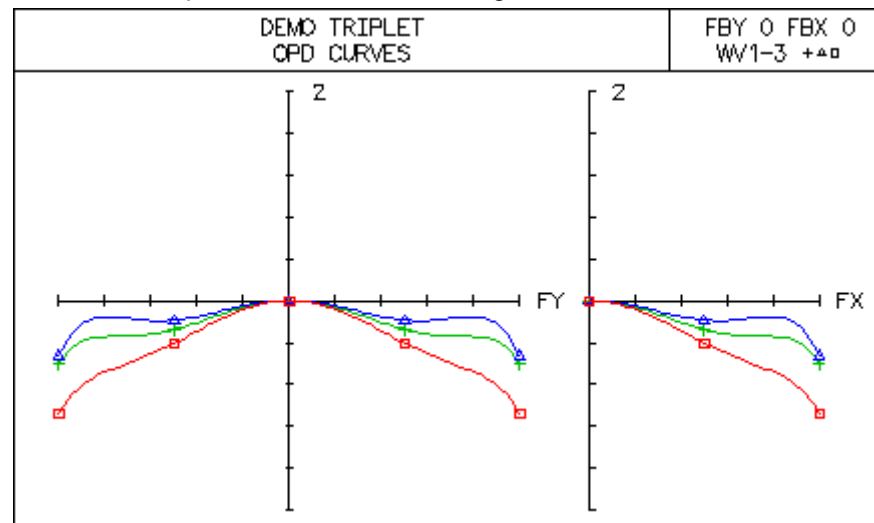
symmetry, the FX curve will be shown for $-1 \leq FX \leq 1$; for symmetric systems only the range $0 \leq FX \leq 1$ is drawn.



A description of the **Plot ray-intercepts as H-tan U** and **OPD-sin U** option is described in the section “Plot ray-intercepts as H-tan U” on page 215.

OPD curves

Click the Plot OPD curves radio button if you want to plot the optical-path-difference curves for the current object point. The curves may be plotted in just the current wavelength, in wavelengths 1, 2, and 3 only, or in all defined wavelengths. The default is to plot the curves in wavelengths 1, 2, and 3.



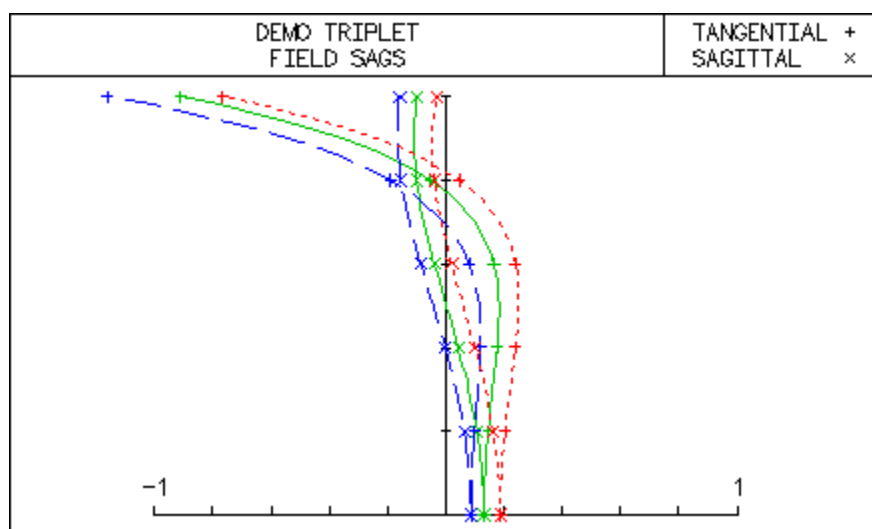
Both x and y-fans are traced from the current object point, and the resulting data is used to make OPD versus fractional object-space FY curves, and OPD versus fractional object-space FX curves. The plotted curves are the result of spline fits to the numerical ray data. For systems without meridional symmetry, the FX curve will be shown for $-1 \leq FX \leq 1$; for symmetric systems only the range $0 \leq FX \leq 1$ is drawn

A description of the **Plot ray-intercepts as H-tan U and OPD-sin U** option is described in the section “Plot ray-intercepts as H-tan U” on page 215.

Field sags

Click the Field sags radio button to generate a plot of the Coddington field sags for the current lens. Real chief rays are traced from a series of object points, and differential displacements are made to determine the x and y-field sags along each ray. The plotted curves are the result of spline fits to the numerical ray data. The field sag data is computed using the current wavelength.

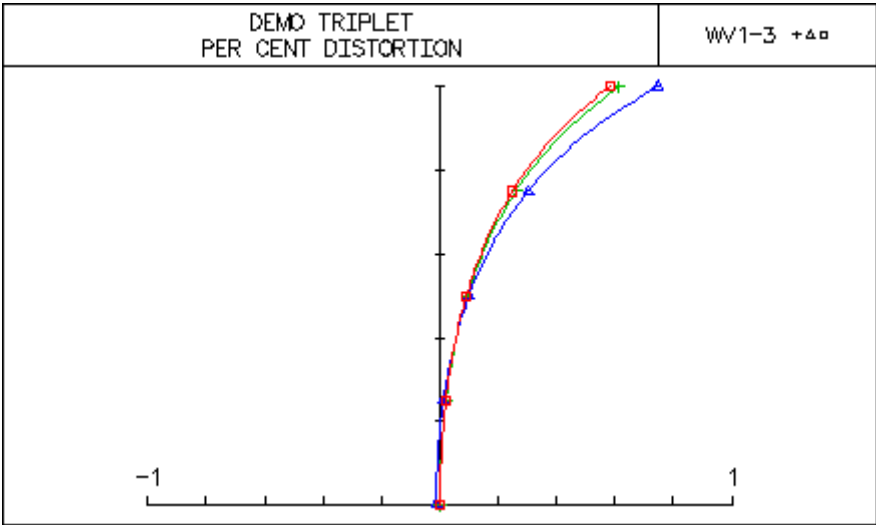
The abscissa of the graph is the field sag, i.e., the distance parallel to the axis from the image surface to the point where the differential rays in the yz and xz planes intersect the chief ray (for focal systems), or diopters of focal shift (for afocal systems). The ordinate of the graph is the fractional object height, ranging from 0 to 1.



Distortion

Click the Distortion radio button to generate a plot of the exact-ray distortion for the current lens. Real chief rays are traced from a series of object points using the current wavelength, and the resultant image surface data are compared to the corresponding paraxial data. The plotted curves are the result of spline fits to the

numerical ray data. The data are only valid for systems having meridional symmetry.

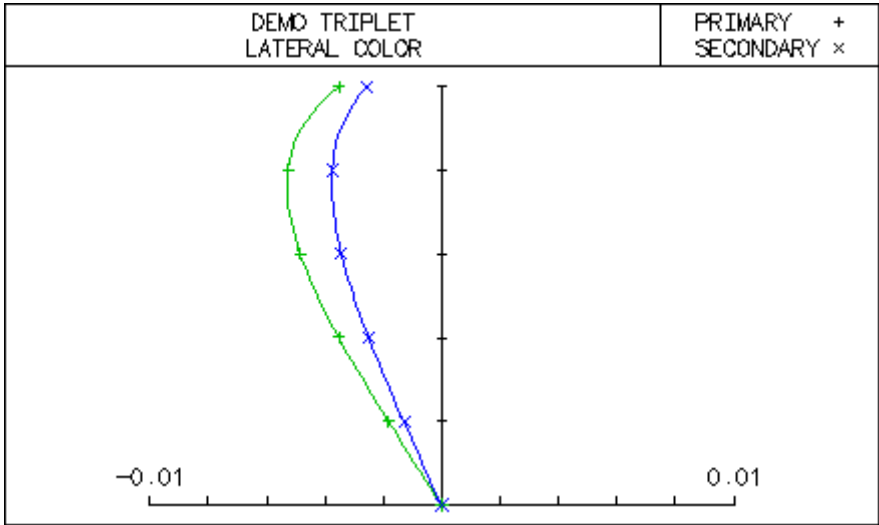


The abscissa of the graph is the distortion, expressed as a percentage. The ordinate of the graph is the fractional object height, ranging from 0 to 1.

Lateral color

Click the Lateral color radio button to generate a plot of the exact-ray lateral color for the current lens. Real chief rays are traced from a series of object points at three wavelengths, and the resultant image surface data are used to construct lateral color plots. The plotted curves are the result of spline fits to the numerical ray data. The data are only meaningful for systems having meridional symmetry. Lateral color can only be displayed if at least three wavelengths are currently defined.

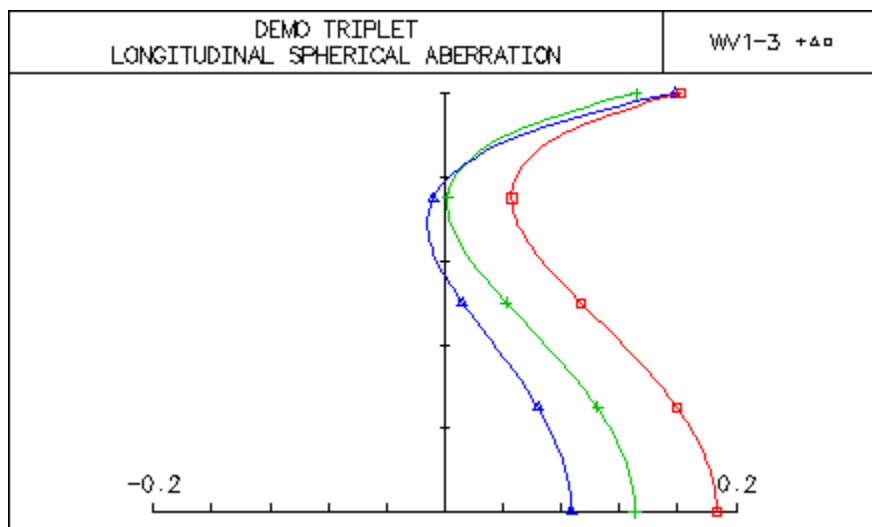
The abscissa of the graph is the lateral color, i.e., the difference in ray-intersections of chief rays in wavelengths 2 and 3 (primary) and in wavelengths 2 and 1 (secondary). The ordinate of the graph is the fractional object height, ranging from 0 to 1.



Longitudinal spherical aberration

Click the Longitudinal spherical aberration radio button if you want to plot the spherical aberration as a longitudinal, rather than transverse, aberration. The curves may be plotted in just the current wavelength, in wavelengths 1, 2, and 3 only, or in all defined wavelengths. The default is to plot the curves in wavelengths 1, 2, and 3. Longitudinal spherical aberration can only be displayed for focal systems.

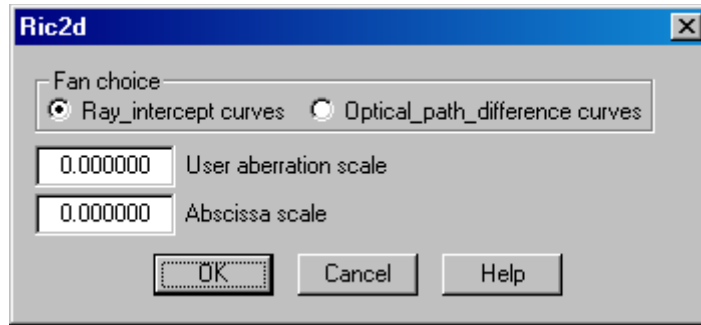
For an on-axis object point, a fan of rays will be traced from the center of the pupil to the edge of the pupil, along the y -axis. (Only this one azimuth is traced because of the assumption of rotational symmetry.) The abscissa of the graph is the longitudinal aberration, i.e., the distance from the image surface to the intersection of the ray with the optical axis. The ordinate of the graph is the fractional pupil coordinate, ranging from 0 to 1.



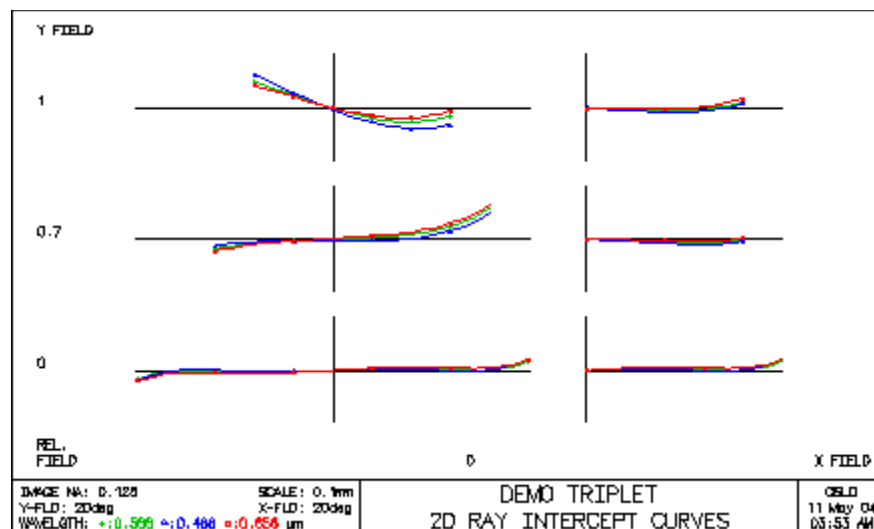
Std

Prm

2-Dimensional Field Variation



The 2D Ray Intercept/OPD Curves Report Graphic plots Ray Intercept/OPD Curves information for all object (field) points defined in the field points set.



Other Ray Analysis

Report Graphic...
Coddington Astigmatism...
F-Theta Analysis...
Chromatic Focal Shift...
Lateral Chromatic Shift...
Distortion Plot (Grid)...

Report Graphic

Ray Intercept Curves / Optical Path Difference Curves

Field Point Input

The ability to plot data for up to 9 field points is common to all report graphics. By default, the plotted field points are taken from the Field Points Set.

The field points are plotted only if they comply with the following rules:

- They must be defined strictly on the Y-axis ($FBX = 0.0$).
- They must be defined for the current configuration. In other words, if the current configuration is 2, all field points defined with a configuration number different from 2 and 0 (0 is equivalent to "all configurations") are ignored.

You are given 3 options as to which field points should be plotted:

- **Take all field points set** - the data is taken from the field point set. If more than 9 field points qualify, the first 9 are selected.
- **Field points from field points set (integers)** - use this option if you need to select a subset from the **Take all field points set** field points. You need to enter the field point number, i.e. an integer.
- **Direct value input** - if you need to plot field points that are not defined in your Field Points Set. The value you enter for each field point is the relative object height (FBY).

Fan Choice

The user can choose to display **Ray Intercept Curves** or **Optical Path Difference Curves** in the first section of the report graphic. These calculations are similar to those detailed earlier except that these analyses can be performed at several field points as described above in **Field Point Input**. The description of **Ray Intercept Curves** is detailed in the section "Ray-intercept curves" on page 275. The description of **Optical Path Difference Curves** is detailed in the section "OPD curves" on page 276.

Astigmatism

The astigmatism graph of the report graphic is detailed in the section "Field sags" on page 277, however here the graph is displayed using only the primary wavelength. The horizontal scale is the displacement along the optical (Z) axis in lens units (default: mm). The vertical scale is the fractional field height, with the top of the scale being $FBY = 1.0$.

Longitudinal spherical aberration

Displays longitudinal spherical aberration vs. aperture in all the defined wavelengths for the on-axis field point. This section of the report graphic is detailed in the section "Longitudinal spherical aberration" on page 279. The horizontal scale is the displacement along the optical (Z) axis in lens units (default: mm). The vertical scale is the fractional system aperture, with the top of the scale being $FY = 1.0$.

Chromatic focal shift plot

Shows the longitudinal displacement of the (paraxial) focus as a function of wavelength. This is the same analysis as the calculation described in section "Chromatic Focal Shift" on page 284, but the graph is turned on its side. The horizontal scale is the displacement along the optical (Z) axis in lens units (default: mm). The vertical scale is the range of wavelengths used in the calculation. This wavelength range usually incorporates the total range of system wavelengths.

Distortion plot

The graphic is essentially the standard distortion plot (**p1d**) turned on its side, and has the advantage that it is easier to visualize positive (pincushion) distortion and negative (barrel) distortion. The horizontal scale is the fractional field height with the right side of the scale being $FBY = 1.0$. The vertical scale is the percent distortion.

Lateral color plot

This graphic is similar to the one detailed in section “Lateral color” on page 278. In this plot, the horizontal scale is the fractional field height with the right side of the scale being $FBY = 1.0$. The vertical scale is the lateral displacement (+/-y) of the rays at different wavelengths in lens units (default: mm). In addition to turning the plot on its side, the curves have a slightly different meaning. Wavelength 1 is defined to be along the x-axis. The meridional displacement of the chief rays in wavelength 2 from wavelength 1 is shown in blue, and the meridional displacement of the chief rays in wavelength 3 from wavelength 1 is shown in red.

Lens Drawing

The lens drawing is the same drawing described in the section “System” on page 224.

Coddington Astigmatism

The CCL command **field** displays a tabular listing of the field aberrations. The syntax of the command is

field [*full_field_output*] *number_of_field_points* *max_fby*

where

full_field_output is an optional qualifier indicating that the analysis is to use the full FBY range of $-max_fby$ to $+max_fby$. The options are “yes” or “no”. The default, “no” is to use 0.0 (on-axis) to $+max_fby$.

number_of_field_points is the number of points at which the aberrations are to be computed (in addition to the on-axis point). The default is 10.

max_fby is the maximum relative value of the field that is used in the analysis.

The data displayed for each fractional field point are:

YC - the intersection height of the chief ray with the image surface.

YFS - the tangential field sag.

XFS - the sagittal field sag.

% DIST - the per cent distortion.

LAT COLOR - the lateral color (difference in chief ray intersection heights between wavelength 2 and wavelength 3).

*FIELD ANALYSIS

WAVELENGTH 1

FIELD	YC	YFS	XFS	% DIST	LAT COLOR
--	--	0.130554	0.130554	--	--
0.100000	1.815460	0.134423	0.124512	0.003254	-0.000925

0.200000	3.631283	0.145305	0.106866	0.013269	-0.001834
0.300000	5.447881	0.160790	0.079042	0.030819	-0.002709
0.400000	7.265761	0.176039	0.043396	0.057268	-0.003528
0.500000	9.085598	0.182404	0.003178	0.094687	-0.004256
0.600000	10.908313	0.165172	-0.037508	0.146057	-0.004845
0.700000	12.735202	0.100028	-0.073756	0.215591	-0.005221
0.800000	14.568119	-0.052499	-0.099923	0.309250	-0.005271
0.900000	16.409782	-0.355792	-0.109723	0.435629	-0.004809
1.000000	18.264293	-0.911383	-0.096417	0.607501	-0.003522

F-Theta Analysis

For typical lenses, the image height (**ImgHt**) relates to the object angle (θ) by the equation

$$ImgHt = F * \tan(\theta) ,$$

where F is the Effective Focal Length. Of course, due to distortion in any system, the effective focal length varies across the field, so F is not constant unless the distortion in the system is zero. Typically users of scan lenses would prefer if the distortion of the lens system followed a different equation, namely

$$ImgHt = F * \theta .$$

This would allow the linear change in the object angle to correspond to a linear change in the image distance. The F-Theta command (**ftheta**) calculates linearity errors for scan lenses based on the desired $F * \theta$ relationship.

*F-THETA ANALYSIS

FRACTIONAL FIELD ANGLE	FIELD ANGLE (DEGREES)	IMAGE HEIGHT	REFERENCE IMG HEIGHT	SCAN ERROR (PER CENT)	LOCAL ERROR (PER CENT)
--	--	--	--	--	--
0.100000	2.000000	1.741821	1.795635	-2.996951	-2.996951
0.200000	4.000000	3.488217	3.591270	-2.869555	-2.742158
0.300000	6.000000	5.243851	5.386905	-2.655590	-2.227660
0.400000	8.000000	7.013572	7.182540	-2.352477	-1.443139
0.500000	10.000000	8.802532	8.978175	-1.956335	-0.371769
0.600000	12.000000	10.616335	10.773810	-1.461647	1.011793
0.700000	14.000000	12.461259	12.569445	-0.860704	2.744960
0.800000	16.000000	14.344585	14.365080	-0.142669	4.883569
0.900000	18.000000	16.275142	16.160715	0.708059	7.513884
1.000000	20.000000	18.264293	17.956350	1.714954	10.777008
CALIBRATED FOCAL LENGTH =		51.441153			
11	-1.000000	-0.974928	0.125931	--	-0.016558
2.922964	0.696079			--	--

Calibrated Focal Length - is the focal length that is the best-fit focal length over the full field of the calculation. Note that for a rotationally symmetric system, the center of the field is much smaller in area than the edge of the field, so the focal length of the center of the field should be weighted less. The equation for the Calibrated Focal Length is a weighted average based on the area of each circular image zone.

$$Calibrated\ Focal\ Length = (ImgHt_1 * \theta_1 + ImgHt_2 * \theta_2 + \dots) / (\theta_1 * \theta_1 + \theta_2 * \theta_2 + \dots)$$

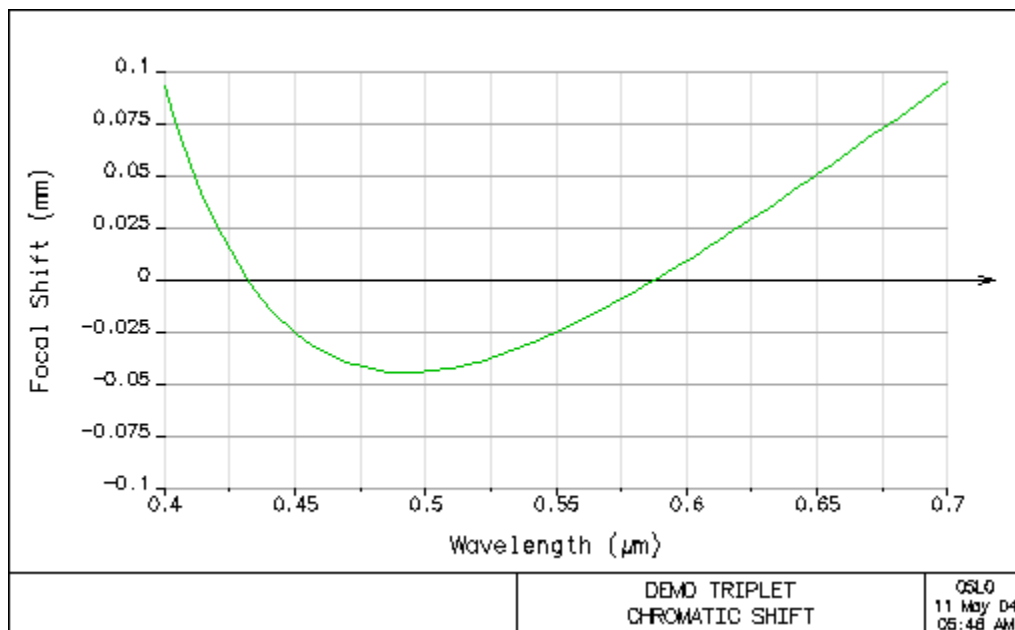
Where:

$ImgHt_i$ = image height of field point $\#i$, and

θ_i = angle of field point $\#i$ in radians

Chromatic Focal Shift

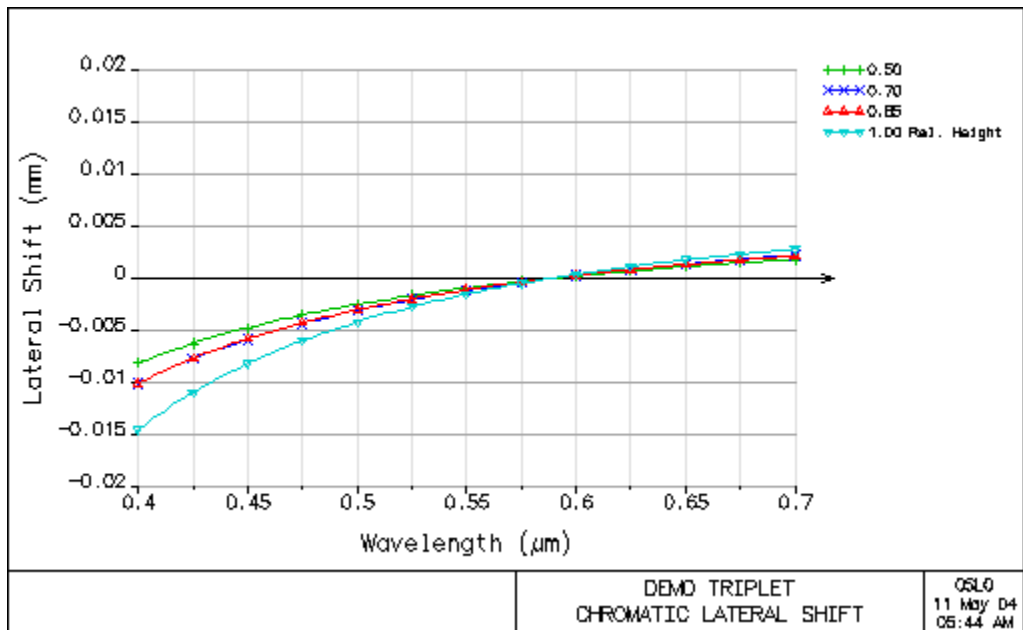
The Chromatic Focal Shift (**chrshft**) command shows the longitudinal paraxial focus shift for an on-axis image point, across a selected part of the spectrum.



Lateral Chromatic Shift

The Lateral Chromatic Shift command (**latshft**) is a complement to the chromatic shift analyses found in the sections “Lateral color” on page 278 and the section “Lateral color plot” on page 282. It shows the lateral chromatic shift in any wavelength range (visible range by default), for 4 field points:

- 0.50 Relative field
- 0.70 Relative field
- 0.85 Relative field
- 1.00 Relative field

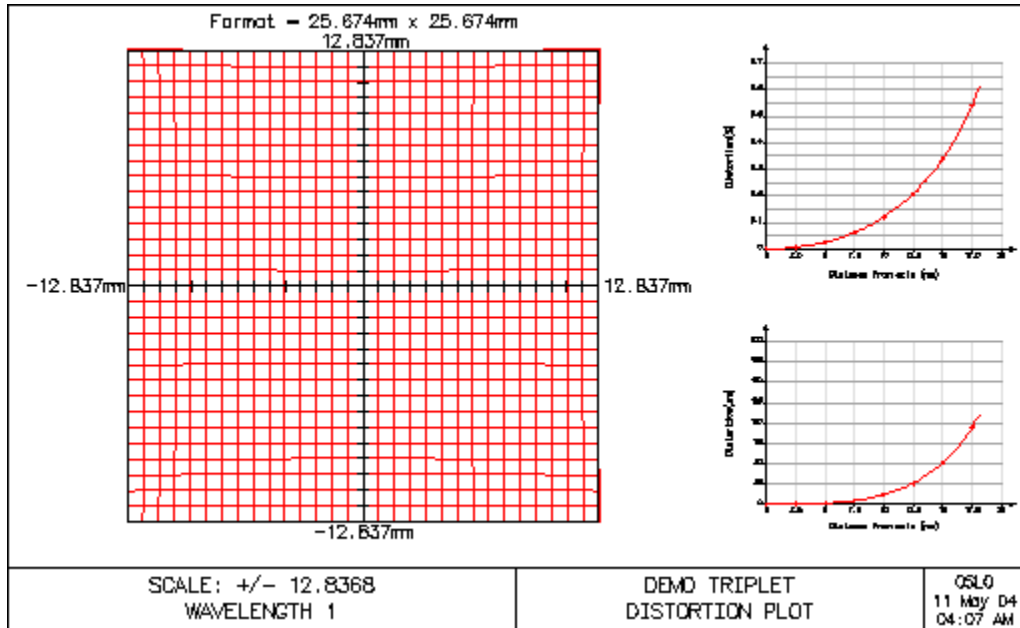


Distortion Plot (Grid)

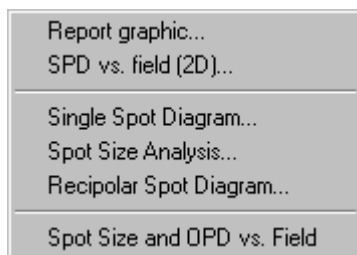
This function plots a distortion analysis of the current system in its current configuration, with 3 graphs:

- The distorted image of a perfect grid. The user is given a choice between many different outputs (plot reference grid, plot symbols only, link distorted grid to reference points, etc.). The grid size can be defined in 3 ways:
 - Automatic:** The grid is given a square shape, whose diagonal matches the ray height at Full Field (i.e. Field Angle or Object height)
 - Field Point Set:** If field points are defined in the field point set with BOTH non zero FBYS AND non zero FBXS, those values are used to set up the image shape of the rectangle.
 - User-Defined:** Enter the values of the X and Y dimension for analysis, for example, 24x36mm.
- The distortion curve in PERCENT, taken along the diagonal of the defined rectangle.
- The distortion curve in MICRONS, taken along the diagonal of the defined rectangle.

NOTE: The results are incorrect for a curved image surface.



Spot Diagram



This group of menu options prints spot diagram ray data, plots a spot diagram, prints information on the current spot diagram, graphs spot size versus field, and graphs spot size versus defocus.

Report Graphic

The Spot Diagram Report Graphic plots spot diagrams information for several object/field points all located along the Y-axis. See the next analysis option, "SPFD vs. field(2D)" for a plot of object/field points in both X and Y.

Defocus - defocus values for the Report Graphic are defined by the input parameters *Number of defocus plots* and *Maximum defocus*.

Scale - specifies the size of the spot diagram plots on the left size of the display (the default of 0.0 means that OSLO will choose an appropriate scale).

The individual plots and labels are based on the graphic and text spot diagram analysis tools.

Field Points Sampling

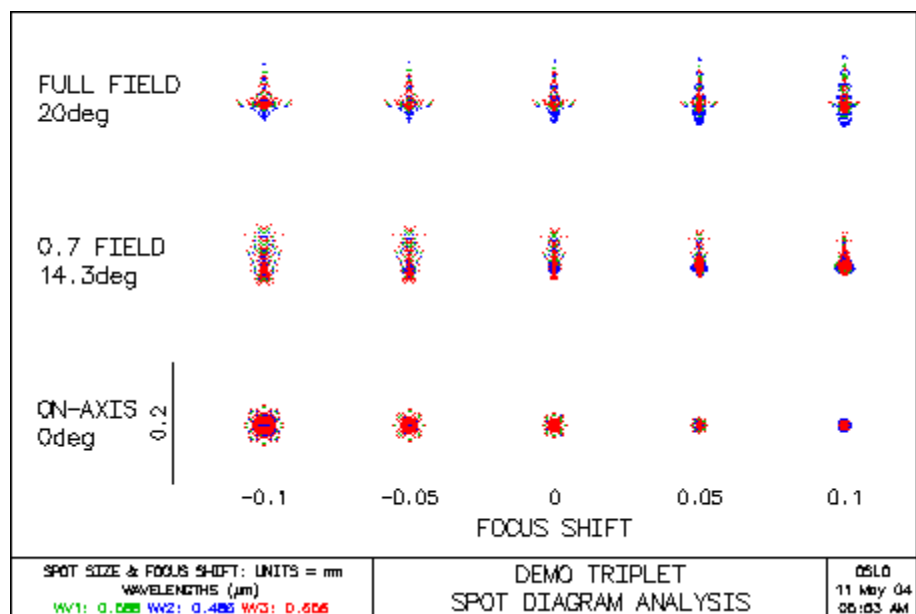
The ability to plot data for up to 9 field points is common to all report graphics. By default, the plotted field points are taken from the Field Points Set.

The field points are plotted only if they comply with the following rules:

- They must be defined strictly on the Y-axis ($FBX = 0.0$).
- They must be defined for the current configuration. In other words, if the current configuration is 2, all field points defined with a configuration number different from 2 and 0 (0 is equivalent to "all configurations") are ignored.

You are given 3 options as to which field points should be plotted:

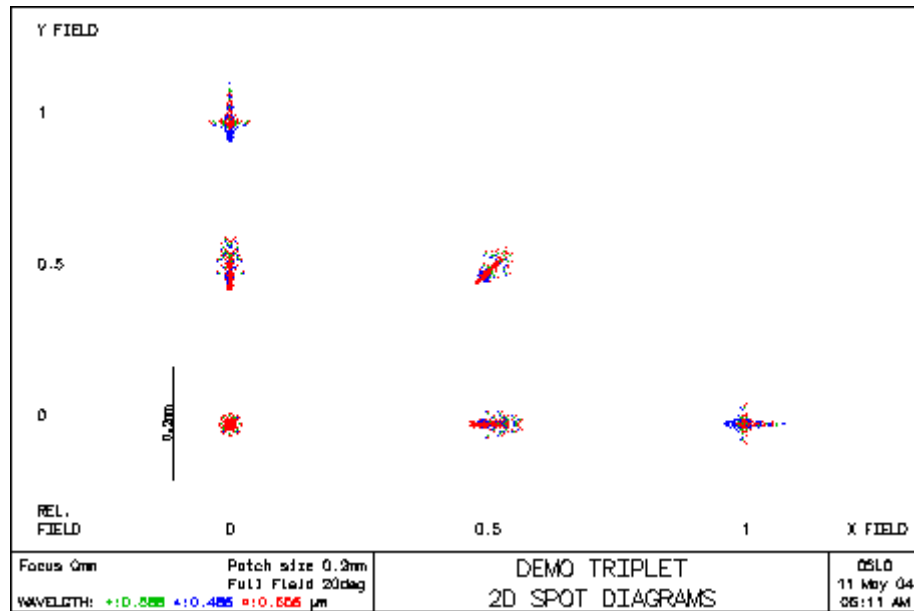
- **Take from field point set** - the data is taken from the field point set. If more than 9 field points qualify, the first 9 are selected.
- **Field points from field points set (integer)** - selects a subset from the **Take from field point set** field points. You need to enter the field point number, i.e. an integer.
- **Direct value input** - plots field points that are not defined in your Field Points Set. The value you enter for each field point is the relative object height (FBY).



Std Prm SPD vs. Field (2D)

The SPD vs. Field graphic (**spd2d**) plots spot diagram information for all object/field points defined in the field points set. The relative orientation of the spot diagrams in the image is maintained, but the inter-spot dimensions are not drawn to scale.

The individual plots and labels are based on the graphic and text spot diagram analysis tools.

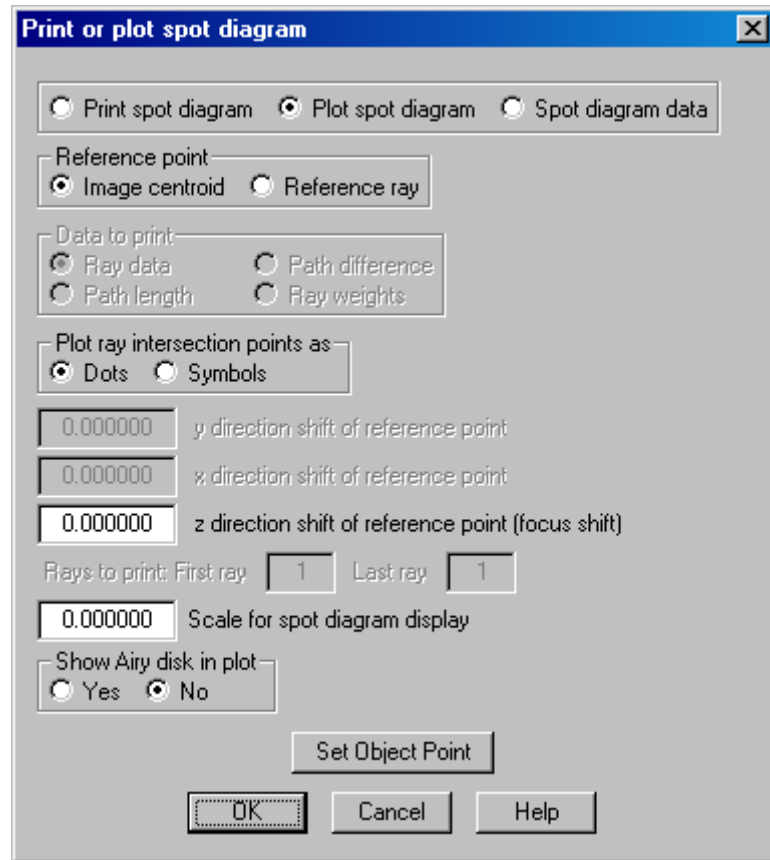


Single Spot Diagram

This command can,

- **Print spot diagram ray data** to the current text window,
- **Plots a spot diagram** to the current graphics window, or
- **Prints information on the current spot diagram** to the current text window.

The input parameters for this analysis includes:



Print spot diagram (prs)

This option prints data for one or more rays in the spot diagram.

Reference point:

- **Image centroid** - ray data are expressed relative to the centroid of the rays on the image surface
- **Reference ray** - ray data are expressed relative to the intersection of the reference ray with the image surface.

Data to print specifies the output to be produced for the range of rays given by the **First ray** and **Last ray** arguments:

- **Ray data** prints
FY and **FX**: fractional pupil coordinates
DY and **DX**: displacement from image centroid or reference ray; printed for focal systems only
DYA and **DXA**: angular displacement from centroid or reference ray for each ray.
 Also printed is **FOCUS**, the input focal shift.
- **Path difference** prints
FY and **FX**: fractional pupil coordinates
YW(IMS-1), **XW(IMS-1)**, and **ZW(IMS-1)**: the coordinates of the ray on a wavefront that passes through the intersection of the reference ray with the next-to-last surface

OPD: optical path difference

Y REF SPH, X REF SPH, Z REF SPH: y, x, and z coordinates of the intersection of the ray with the reference sphere for each ray

Also printed are **YSHIFT**, **XSHIFT**, and **RSZ**, the input reference sphere center shift.

- **Path length** prints

FY and **FX:** fractional pupil coordinates

YS(IMS-1), **XS(IMS-1)**, and **ZS(IMS-1):** coordinates of the ray intersection with the next-to-last surface

OPL(IMS-1): optical path length along the ray from the entrance pupil to the next-to-last surface for each ray

- **Ray weights** prints

FY and **FX:** fractional pupil coordinates

WGT: relative weight of the ray; affected only by Gaussian apodization or user-defined raytrace surfaces

Y(IMS), **X(IMS)** and **Z(IMS):** coordinates of the ray intersection with the image surface for each ray

Also printed is **FOCUS**, the input focal shift.

Plot spot diagram (pls)

This option plots the current spot diagram in the current graphics window. A spot diagram is a plot of ray intercepts on the image surface relative to the reference point.

Reference point:

- **Image centroid** - ray data are expressed relative to the centroid of the rays on the image surface
- **Reference ray** - ray data are expressed relative to the intersection of the reference ray with the image surface.

Plot ray intersection point as:

- **Dots** - each wavelength will be plotted as a single dot in a different color
- **Symbols** - each wavelength will be plotted in a different color, using a different symbol

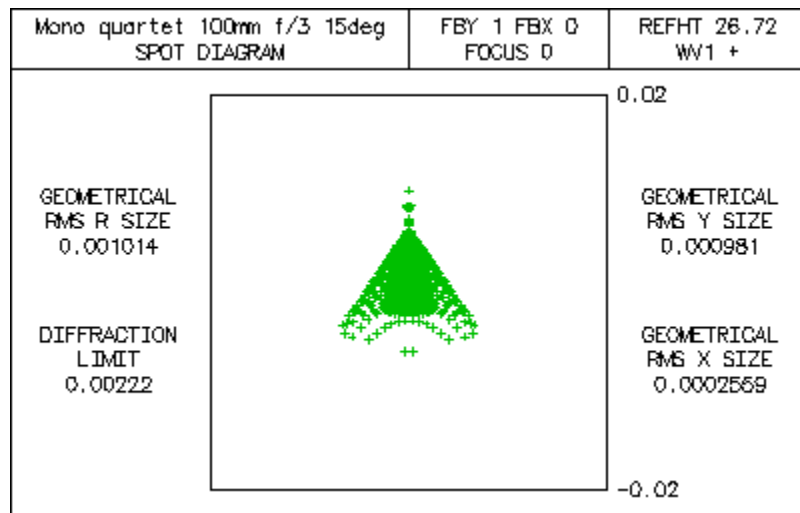
Z direction shift of focal point (focus shift) - the user can enter a focus shift offset (+/-Z) of the image surface (IMS).

Scale for spot diagram display - determines the size of the plot; the default value of 0.0 indicates that OSLO should determine an appropriate scale.

Show Airy disk in plot - If this option is set to *On*, the equivalent Airy disk will be drawn on all spot diagram displays. This provides a qualitative comparison of the geometric image of a point (the spot diagram) to the limiting spot size imposed by diffraction.

The values of the geometric Y, X, and R spot sizes, along with the diffraction limited spot size, are printed in the plot. Spot size data values refer to semi-diameters of the spot. For Focal systems, the ray coordinates are transverse ray

displacements in system units. For Afocal systems, the ray coordinates are angular ray displacements in radians.



Spot diagram data

This option prints information about the current spot diagram:

APDIV - the number of aperture divisions in the entrance pupil grid across one dimension of the pupil.

WAV WEIGHTS - the relative weight applied to each currently defined wavelength.

GAU SSY and **SSX** - the $1/e^2$ irradiance spot sizes, in y and x, respectively, of the entering beam, measured at surface 1. (Only displayed if Gaussian apodization is used.)

NUMBER OF RAYS TRACED - the number of rays traced in each wavelength (note that if **SPOT DIAGRAM SYMMETRY** (see below) is non-zero, the number of rays actually traced is half this number).

PER CENT WEIGHTED RAY TRANSMISSION - For an unapodized system, the per cent transmission is given by $100 \times (\text{the sum of the weights of all rays transmitted through the lens, divided by the sum of the weights of all rays passing through an unvignetted pupil})$. If Gaussian apodization is used, the per cent transmission is given by $100 \times (\text{total transmitted power, as determined by the sum of the weights of all rays transmitted through the lens, divided by the total incident power in the Gaussian beam})$.

SPOT DIAGRAM SYMMETRY - is 1.0 if the spot diagram is symmetric (that is, if OSLO knows that the lens is symmetric and if the current object point is in the Y-Z plane), then only half the number of rays displayed in the **NUMBER OF RAYS TRACED** output have actually been traced.

After you have selected the desired options, click OK to print or plot the spot diagram.

```
*SPOT DIAGRAM - FBY 1.00, FBX 0.00, FBZ 0.00 - MONOCHROMATIC APODIZED
APDIV 41.050000
```

```

WAVELENGTH 1
WAV WEIGHTS:
  WW1
    1.000000
GAU   SSY       SSX
    3.000000    3.000000
NUMBER OF RAYS TRACED:
  WV1
    1.3160e+03
PER CENT WEIGHTED RAY TRANSMISSION: 100.000000
SPOT DIAGRAM SYMMETRY:      1.000000

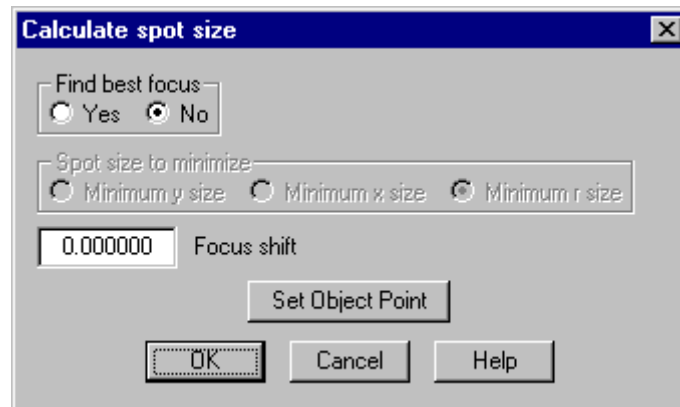
```

Spot Size Analysis

Select Spot Size Analysis from the Evaluate menu to calculate and display some statistical information about the ray distribution in the spot diagram.

- Find Best Focus:**
 Click the Yes radio button if you want OSLO to find the optimum focus shift for minimum spot size.
 Click the No radio button if you want to specify the focus shift, if any, yourself.
- Spot Size to Minimize:**
 If you want OSLO to find the best focus, click the radio button for the spot size to minimize: y spot size (BFY), x spot size (BFX), or r spot size (BFR), where $r^2 = x^2 + y^2$.
- Focus Shift:**
 If you want to specify the focus shift directly, type the value in the Focus shift cell. Focus shift is only applicable to focal systems.
- Set Object Point:**
 Click on this button if you want to change the current reference ray (which will change the field point for the analysis).

Click OK to calculate the spot sizes.



The program will display six pieces of spot size information, along with the focus shift, if any.

GEO RMS Y, GEO RMS X, GEO RMS R - the root-mean-square geometric spot sizes in the y , x , and r directions (displayed for focal systems).

GEO RMS YA, GEO RMS XA, GEO RMS RA - the root-mean-square geometric angular deviations of the rays in the spot diagram (displayed for afocal systems).

DIFFR LIMIT - the radius of the diffraction-limited spot size for the system, including vignetting effects. This value is displayed so that the performance of the actual system can be compared to a diffraction-limited system. This spot size is the radius of the Airy disk that would result from a circular exit pupil of numerical aperture equal to the largest numerical aperture of the real exit pupil as seen from the image point.

CENTY, CENTX - centroids of the spot diagram, referred to the intersection of the reference ray with the (focus-shifted) image surface (displayed for focal systems).

CENTYA, CENTXA - centroids of the angular deviations of rays in the spot diagram, referred to the angle of the reference ray with the image surface (displayed for afocal systems).

*SPOT SIZES BFR

FOCUS: -0.690973

GEO RMS Y	GEO RMS X	GEO RMS R	DIFFR LIMIT	CENTY	CENTX
0.002918	0.002918	0.004126	2.2839e-05	--	--

Recipolar Spot Diagram

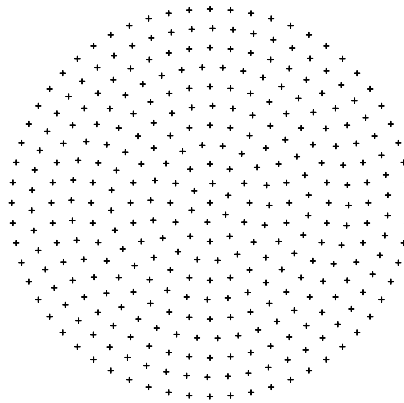
In OSLO there is the ability to calculate and display an additional spot diagram. The default spot diagram calculation in OSLO uses a ray distribution in the entrance pupil based on a square grid pattern. This general pattern is used because many image evaluation calculations (*MTF*, *PSF*, etc.) require that the input wavefront data be arranged on a uniform square grid. For the simple display of a spot diagram, there are, of course, other patterns that may be used, subject to the condition that each ray is associated with an equal and (approximately) equally shaped area of the pupil. The choice of sampling pattern affects symmetries and “sampling artifacts” in the resulting plotted spot diagrams. One pattern that is attractive, particularly for use with rotationally symmetric systems, is the *recipolar* or *hexapolar* array. This array arranges rays on concentric circles, so the boundary of a circular pupil coincides with sampling points, as contrasted to the zigzag boundary resulting from square grid sampling. (This form of the recipolar spot diagram is used by A. Cox in his book *A System of Optical Design*.)

In the recipolar array, the sample points lie on concentric circles, equally spaced in radius. The number of samples around each circle increases by six from one ring to the next larger radius ring. Thus, there are six rays in the first ring, 12 in the second, 18 in the third, etc. The angular separation of adjacent rays in ring n is $\Delta\theta = 60/n$ degrees. To reduce symmetry, the starting azimuths for the circles are staggered according to $\theta_0 = 22\frac{1}{2} + (n - 1) \cdot 7\frac{1}{2}$ degrees. If M rings are traced, along with a ray through the center of the pupil, the total number of rays launched (N_{rays}) is

(8.7)

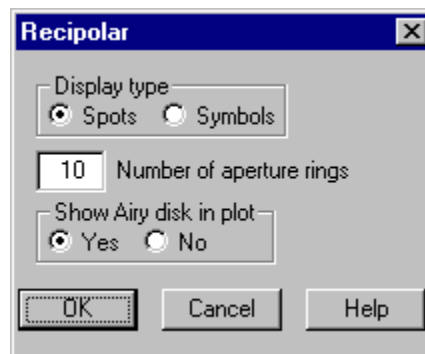
$$\begin{aligned}
 N_{\text{rays}} &= 1 + 1 \times 6 + 2 \times 6 + 3 \times 6 + \dots + M \times 6 \\
 &= 1 + 6 \sum_{n=1}^M n = 1 + 6 \left[\frac{M(M+1)}{2} \right] \\
 &= 1 + 3M(M+1)
 \end{aligned}$$

For example, with 10 rings ($M = 10$), $N_{\text{rays}} = 1 + 3 \times 10 \times 11 = 331$. The pupil sampling pattern is shown below.



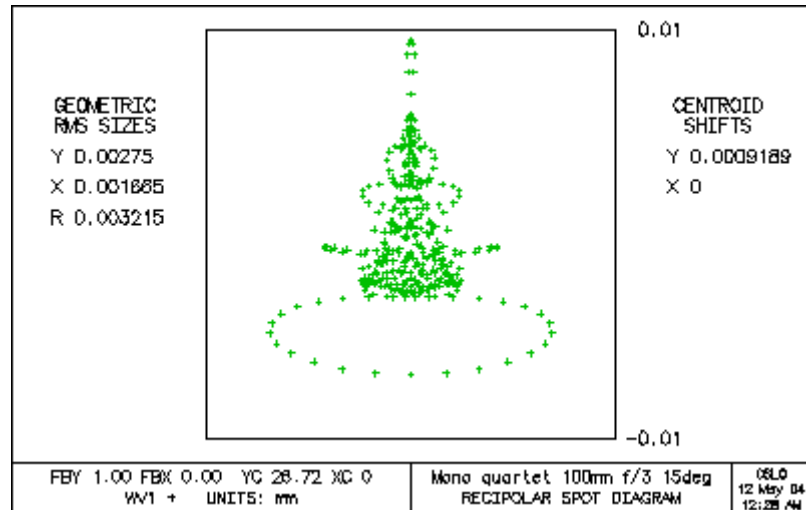
Analysis parameters to choose from for the **recipolar** command include:

- **Display type**
The ray intersection points may be plotted as “spots” or with a symbol corresponding to the wavelength of the ray.
- **Number of aperture rings**
The number of aperture rings (default of 10) may also be specified.
- **Show Airy disk in plot**
If this option is set to *On*, the equivalent Airy disk will be drawn on all spot diagram displays. This provides a qualitative comparison of the geometric image of a point (the spot diagram) to the limiting spot size imposed by diffraction.



Along with the spot diagram itself, the rms spot sizes in y , x , and r , and the shifts of the centroid from the reference ray intersection are displayed in the plot. Note

that because the recipolar array is not point-by-point symmetric about the y -axis, the centroid shift in the x -direction may be reported as a small, non-zero value for rotationally symmetric systems with vignetting.

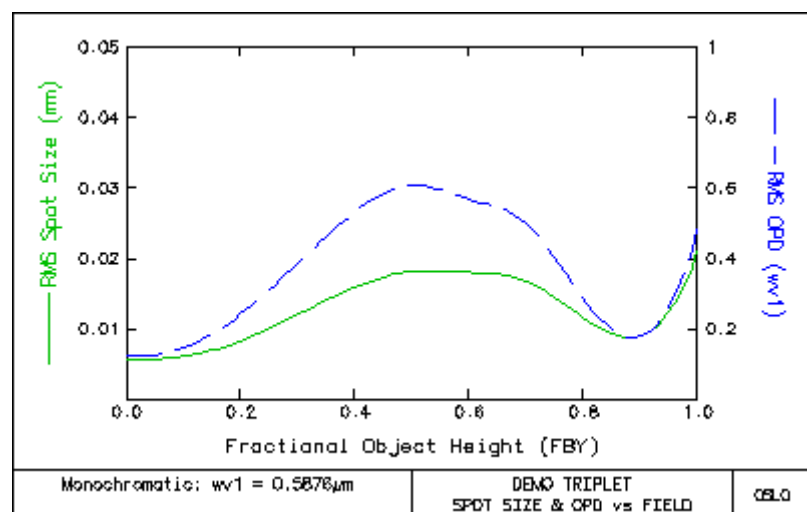


Spot Size and OPD vs. Field

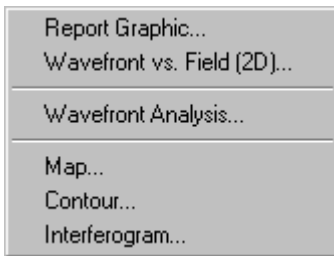
The **spsopd** command displays:

RMS Spot Size - Plotted as a function of fractional object height in the meridional plane (FBY). The RMS spot size output refers to the radius (Semi-Diameter) of the spot in lens units (default: mm).

RMS OPD (Optical Path Difference: wavefront error) - Plotted as a function of fractional object height in the meridional plane (FBY). The RMS OPD is measured in wavelengths of the primary wavelength (WV1).



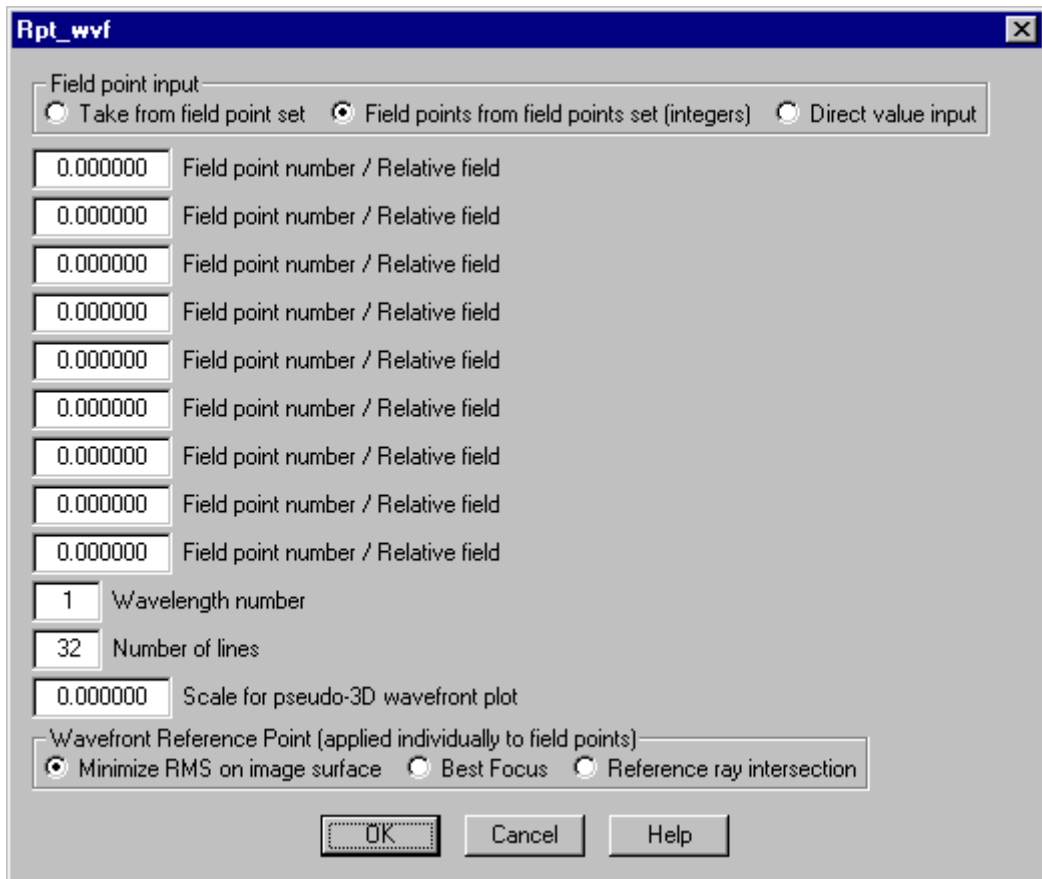
Wavefront



Report Graphic

The wavefront report graphic plots wavefront information for several object/field points all located along the Y-axis. See the section “Wavefront vs. Field (2D)” on page 298 to plot object/field points in both X and Y.

The individual plots and labels are based on the graphic and text wavefront analysis tools.



Field Points Sampling

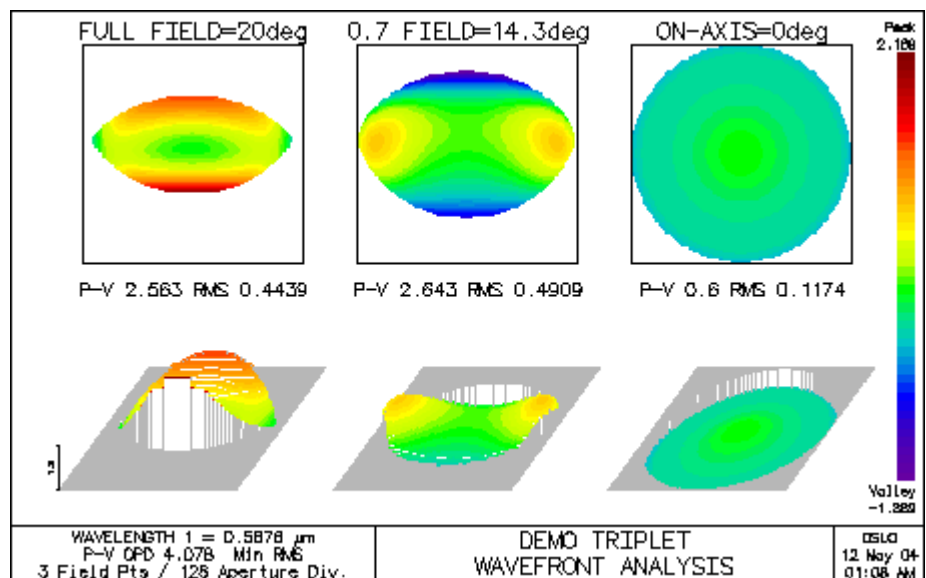
The ability to plot data for up to 9 field points is common to all report graphics. By default, the plotted field points are taken from the Field Points Set.

The field points are plotted only if they comply with the following rules:

- They must be defined strictly on the Y-axis ($FBX = 0.0$).
- They must be defined for the current configuration. In other words, if the current configuration is 2, all field points defined with a configuration number different from 2 and 0 (0 is equivalent to "all configurations") are ignored.

You are given 3 options as to which field points should be plotted:

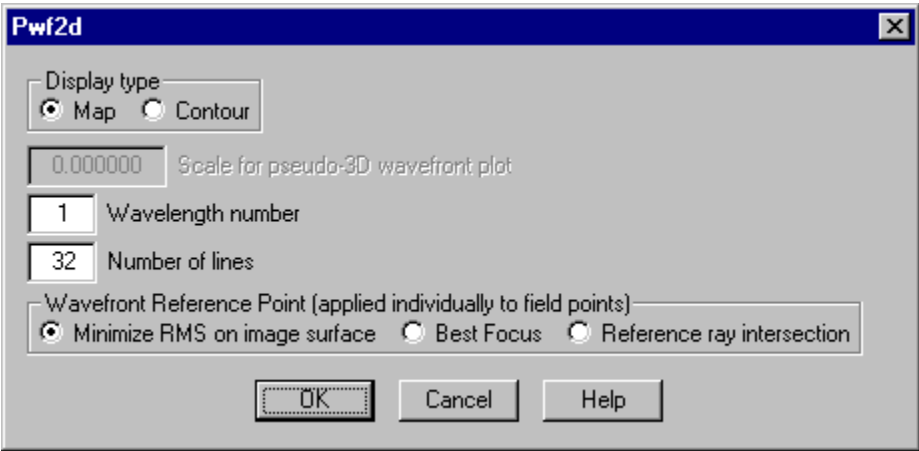
- **Take from field point set** - the data is taken from the field point set. If more than 9 field points qualify, the first 9 are selected.
- **Field points from field points set (integer)** - selects a subset from the **Take from field point set** field points. You need to enter the field point number, i.e. an integer.
- **Direct value input** - plots field points that are not defined in your Field Points Set. The value you enter for each field point is the relative object height (FBY).



Std

Prm

Wavefront vs. Field (2D)

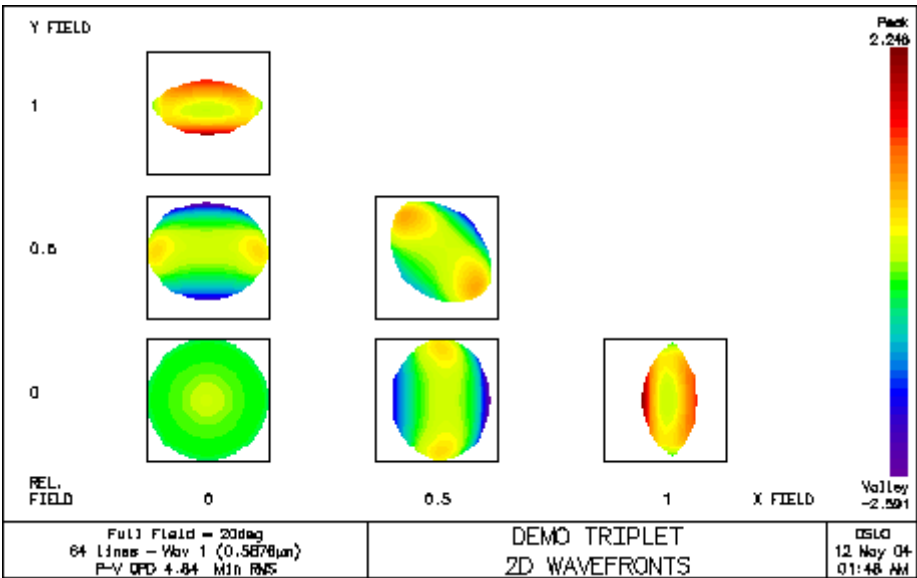


The 2D wavefront report graphic plots wavefront information for all object/field points defined in the field points set. The relative orientation of the wavefront plots in the image is maintained, but the inter-plot dimensions are not drawn to scale.

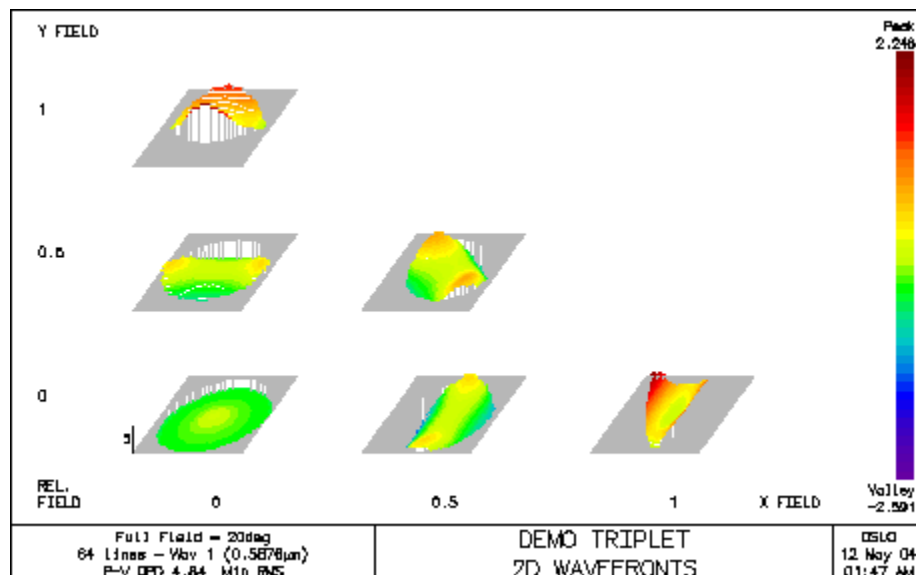
The individual plots and labels are based on the graphic and text wavefront analysis tools.

Display Type

Map



Contour



Wavefront Analysis

Select Wavefront Analysis from the Evaluate menu to display some data describing the wavefront for the current spot diagram.

Calculate wavefront statistics

Type of analysis
☒ Wavefront statistics ☐ Zernike analysis of wavefront

Center of wavefront reference sphere coordinates
☒ Minimum RMS point on surface ☐ Focal shifted image surface
☐ Best focus ☐ Reference ray intersection

0.000000 y direction shift of reference sphere center
 0.000000 x direction shift of reference sphere center
 0.000000 z direction shift of reference sphere center (focus shift)

Chromatic option
☒ Monochromatic ☐ Polychromatic

1 Wavelength of wavefront
 36 Number of Zernike polynomials

Set Object Point

OK Cancel Help

Wavefront statistics

The wavelength of the wavefront is selected by typing the desired wavelength number in the Wavelength of wavefront cell. There are six pieces of data displayed for the wavefront. The first three are:

PKVAL OPD - the maximum peak-to-valley optical path difference in the spot diagram data.

RMS OPD - the root-mean-square value of the optical path differences in the spot diagram data.

STREHL RATIO - the value of the point spread function at the requested diffraction focus, relative to an unaberrated system.

There are four options available for locating the center of the reference sphere. These options and the additional data displayed are:

Center of wavefront reference sphere coordinates

Minimum RMS point on surface

Click this radio button if you want to move the center of the reference sphere by specified y shift and x shift amounts from the point on the focal shifted (by an amount z shift) image surface that minimizes the variance of the wavefront. The specified **Y SHIFT**, **X SHIFT**, and focus shift (**RSZ**) are reported.

*WAVEFRONT

WAVELENGTH 1

PKVAL OPD	RMS OPD	STREHL RATIO	YSHIFT	XSHIFT	RSZ
0.551560	0.123537	0.555861	--	--	--

Focal shifted image surface

Click this radio button if you want to use the point on the focus shifted (by an amount z shift) image surface that minimizes the variance of the wavefront as the center of the reference sphere. **RSY**, **RSX**, and **RSZ** will be reported as the location of this point.

*WAVEFRONT RS

WAVELENGTH 1

PKVAL OPD	RMS OPD	STREHL RATIO	RSY	RSX	RSZ
0.355818	0.135295	0.443876	--	--	0.061000

Best focus

Click this radio button if you want to use the point in three-dimensional space that minimizes the variance of the wavefront as the center of the reference sphere. **RSY**, **RSX**, and **RSZ** will be reported as the location of this point.

*WAVEFRONT BF

WAVELENGTH 1

PKVAL OPD	RMS OPD	STREHL RATIO	RSY	RSX	RSZ
0.175274	0.044911	0.922647	--	--	0.028968

Reference ray intersection

Click this radio button if you want the evaluation performed with the reference sphere centered at a point shifted by prescribed amounts y shift, x shift and z shift from the intersection of the reference ray with the image surface. The specified **Y SHIFT**, **X SHIFT**, and focus shift (**RSZ**) are reported.

*WAVEFRONT REF
WAVELENGTH 1

PKVAL OPD	RMS OPD	STREHL RATIO	YSHIFT	XSHIFT	RSZ
0.551560	0.123537	0.555861	--	--	--

Click the OK button to compute the wavefront statistics after you have specified the desired option.

Zernike analysis

Click the Zernike analysis of wavefront radio button if you wish to perform a Zernike decomposition of the wavefront. You select the wavelength and center of the reference sphere in the same manner as for the wavefront statistics calculation described above. In addition, you can specify the number of Zernike terms to be used in the analysis. The maximum number of Zernike terms is 36.

It is recommended that the equal image space ray increments spot diagram operating condition be active before performing a Zernike polynomial decomposition of the wavefront. The equal increment ray sampling increases the accuracy of the orthogonalization procedure used in the calculations.

The output consists of the coefficients of the various Zernike polynomials, for an expansion of the desired wavefront. The coefficients are in units of wavelengths, defined at the wavelength of the wavefront, if the **opdw** (See Chapter 4) general operating condition is on; otherwise the coefficients are in the current lens units. Note that, as discussed in the definitions of the Zernike sag and phase surfaces in Chapter 4, the azimuthal angle (denoted by **A** in the OSLO output below) is measured from the *y*-axis.

*ZERNIKE ANALYSIS
WAVELENGTH 1

```

-0.319936: [0] 1
--      : [1] RCOSA
--      : [2] RSINA
-0.187489: [3] 2R^2 - 1
--      : [4] R^2COS2A
--      : [5] R^2SIN2A
--      : [6] (3R^2 - 2)RCOSA
--      : [7] (3R^2 - 2)RSINA
0.027369: [8] 6R^4 - 6R^2 + 1
--      : [9] R^3COS3A
--      : [10] R^3SIN3A
--      : [11] (4R^2 - 3)R^2COS2A
--      : [12] (4R^2 - 3)R^2SIN2A
--      : [13] (10R^4 - 12R^2 + 3)RCOSA
--      : [14] (10R^4 - 12R^2 + 3)RSINA
-0.113198: [15] 20R^6 - 30R^4 + 12R^2 - 1
--      : [16] R^4COS4A
--      : [17] R^4SIN4A
--      : [18] (5R^2 - 4)R^3COS3A
--      : [19] (5R^2 - 4)R^3SIN3A
--      : [20] (15R^4 - 20R^2 + 6)R^2COS2A
--      : [21] (15R^4 - 20R^2 + 6)R^2SIN2A
--      : [22] (35R^6 - 60R^4 + 30R^2 - 4)RCOSA
--      : [23] (35R^6 - 60R^4 + 30R^2 - 4)RSINA
-0.008655: [24] 70R^8 - 140R^6 + 90R^4 - 20R^2 + 1
--      : [25] R^5COS5A
--      : [26] R^5SIN5A

```

```

--      : [27] (6R^2 - 5)R^4COS4A
--      : [28] (6R^2 - 5)R^4SIN4A
--      : [29] (21R^4 - 30R^2 + 10)R^3COS3A
--      : [30] (21R^4 - 30R^2 + 10)R^3SIN3A
--      : [31] (56R^6 - 105R^4 + 60R^2 - 10)R^2COS2A
--      : [32] (56R^6 - 105R^4 + 60R^2 - 10)R^2SIN2A
--      : [33] (126R^8 - 280R^6 + 210R^4 - 60R^2 + 5)RCOSA
--      : [34] (126R^8 - 280R^6 + 210R^4 - 60R^2 + 5)RSINA
-0.000567: [35] 252R^10 - 630R^8 + 560R^6 - 210R^4 + 30R^2 - 1
-3.3199e-05: [36] 924R^12 - 2772R^10 + 3150R^8 - 1680R^6 + 420R^4 - 42R^2 + 1
RMS OPD      0.117073      ERROR 3.9530e-07

```

Map (pwf map)

This option creates a 2D false color plot in the current graphics window.

Wavefront Reference Point

Specifies the center of the reference sphere:

- **Minimum RMS wavefront** image surface point determines the optimal reference sphere center on the image surface (after shifting by Focus shift).
- **Best focus** first determines the optimal position of the reference sphere center, then applies an appropriate amount of focus shift.
- **Reference ray** - image surface intersection uses the intersection of the reference ray with the image surface, shifted by Focus shift, as the center of the reference sphere.

Save data as INT file

It is possible to save the wavefront data as an interferogram file for later use.

The file is saved in the INT files directory defined in OSLO preferences. The saved data ignores the following arguments:

- Display type.
- Plot horizontal direction.
- Scale.
- Interferogram related arguments: **OPD per fringe**, **Y tilt**, **X tilt**.

Contour (pwf con)

This option creates a pseudo3D false color plot in the current graphics window.

Scale (contour)

Vertical scale for the pseudo3D plot. Controls the apparent "flatness" of the wavefront.

Wavefront Reference Point

Specifies the center of the reference sphere:

- **Minimum RMS wavefront** image surface point determines the optimal reference sphere center on the image surface (after shifting by Focus shift).
- **Best focus** first determines the optimal position of the reference sphere center, then applies an appropriate amount of focus shift.
- **Reference ray** - image surface intersection uses the intersection of the reference ray with the image surface, shifted by Focus shift, as the center of the reference sphere.

Save data as INT file

It is possible to save the wavefront data as an interferogram file for later use.

The file is saved in the INT files directory defined in OSLO preferences. The saved data ignores the following arguments:

- Display type.
- Plot horizontal direction.
- Scale.
- Interferogram related arguments: **OPD per fringe**, **Y tilt**, **X tilt**.

Interferogram (pwf int)

This option plots wavefront Optical Path Differences in the current graphics window using grey scale intensity fringes.

OPD per fringe

Sets the step in OPD between 2 successive white fringes, relative to the **Wavelength**. The default value is 0.5.

X tilt and Y tilt

Sets the tilt of the test plate in fringes, relative to the **Wavelength**. The default value is 0.5.

Wavefront Reference Point

Specifies the center of the reference sphere:

- **Minimum RMS wavefront** image surface point determines the optimal reference sphere center on the image surface (after shifting by Focus shift).
- **Best focus** first determines the optimal position of the reference sphere center, then applies an appropriate amount of focus shift.
- **Reference ray** - image surface intersection uses the intersection of the reference ray with the image surface, shifted by Focus shift, as the center of the reference sphere.

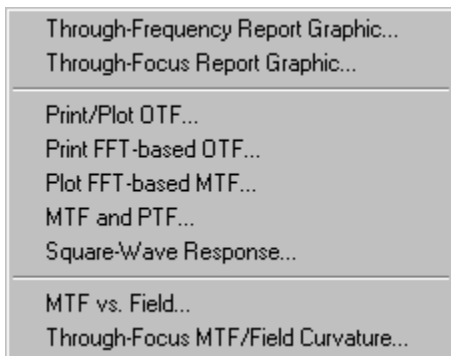
Save data as INT file

It is possible to save the wavefront data as an interferogram file for later use.

The file is saved in the INT files directory defined in OSLO preferences. The saved data ignores the following arguments:

- Display type.
- Plot horizontal direction.
- Scale.
- Interferogram related arguments: **OPD per fringe**, **Y tilt**, **X tilt**.

Transfer Function



A common image evaluation technique involves the use of *transfer functions* to describe the performance of a system in imaging a sinusoidal grating of a particular spatial frequency. The ratio of the image contrast to the object contrast is called the *modulation transfer function* (MTF) of the system, and the shift in phase of the image sinusoidal distribution that occurs because of distortion in the system is called the *phase transfer function* (PTF). Both the MTF and the PTF are, of course, functions of the grating frequency. For more information on the optical transfer function, see Chapter 7 – Image Evaluation in the Optics Reference manual.

Overview: Diffraction versus Geometrical MTF

In terms of calculation, the optical transfer function is the Fourier transform of the point spread function, so obviously both diffraction and aberrations affect the transfer function. In OSLO, the transfer function is computed by convolving the complex pupil function with itself. (Fourier theory shows that this is equivalent to transforming the point spread function.) In the convolution calculation, the OPD of rays from different parts of the pupil is subtracted to find a resultant phase term. In order for the calculation to be accurate, it is important that there not be any 2π uncertainties in the phase of adjacent cells. This means that the aberrations in the wavefront emerging from the system must be sufficiently small for this to occur. If the overall wavefront has aberrations of a few waves, this will generally be the case, but for systems that are nowhere near being diffraction limited, diffraction evaluation calculations will not produce meaningful results.

For systems that have too much aberration to be evaluated using the diffraction-based transfer function calculations, OSLO has an alternative routine to compute the so-called geometrical optical transfer function of the system. The geometrical optical transfer function of a system is defined to be the Fourier transform of the spot diagram. Geometrical optical transfer functions are equivalent to diffraction optical transfer functions for systems that are dominated by aberrations, but when diffraction effects become comparable in magnitude to image degradation introduced by aberrations, geometrical transfer functions do not give meaningful results.

The type of transfer function that is calculated depends on the value of the **spd_mtf_switch (sdms)** system variable. The default value for the peak-to-valley OPD that causes a switch-over from diffraction to geometrical *MTF* has been set at 1000 wavelengths. If the peak-to-valley OPD is less than the value of the operating condition, a diffraction calculation is performed. Otherwise, a

geometrical calculation is done. The diffraction *MTF* calculation algorithm is streamlined so that the time required (particularly for systems with large amounts of aberration) to perform the diffraction calculation is comparable to the geometric calculation. Also, the accuracy of the diffraction *MTF* is maintained, assuming adequate sampling in the pupil. Using the **sdms** default value of 1000 wavelengths means that the majority of systems will produce diffraction-based *MTF* results. Of course, the switch-over OPD can be set to any desired value using the **sdms** command.

Through-Frequency Report Graphic

The MTF Report Graphic plots MTF information for several object/field points all located along the Y-axis. The diffraction limit MTF is also shown and is marked as *IDEAL*.

The plot is interactive: to highlight a specific field point curve, click on its label in the upper left of the window. The curves related to that field point will be plotted in red. To deselect, click on the "FIELD POINTS" label on top. Please allow a few seconds for the graphic to update when selecting field point curves.

The individual plots and labels are based on the standard MTF analysis tools.

Field Points Sampling

The ability to plot data for up to 9 field points is common to all report graphics. By default, the plotted field points are taken from the Field Points Set.

The field points are plotted only if they comply with the following rules:

- They must be defined strictly on the Y-axis (FBX = 0.0).
- They must be defined for the current configuration. In other words, if the current configuration is 2, all field points defined with a configuration number different from 2 and 0 (0 is equivalent to "all configurations") are ignored.

You are given 3 options as to which field points should be plotted:

- **Take from field point set** - the data is taken from the field point set. If more than 9 field points qualify, the first 9 are selected.
- **Field points from field points set (integer)** - selects a subset from the **Take from field point set** field points. You need to enter the field point number, i.e. an integer.
- **Direct value input** - plots field points that are not defined in your Field Points Set. The value you enter for each field point is the relative object height (FBY).

Maximum Frequency

The maximum frequency to be represented in the MTF versus Frequency plot (minimum frequency is = 0.0). Frequency is in *Cycles per mm* for focal systems and *Cycles per radian* for afocal systems.

Through-Focus Report Graphic

The MTF Report Graphic plots MTF information for several object/field points all located along the Y-axis. The diffraction limit MTF is also shown and is marked as *IDEAL*.

The plot is interactive: to highlight a specific field point curve, click on its label in the upper left of the window. The curves related to that field point will be plotted in red. To deselect, click on the "FIELD POINTS" label on top. Please allow a few seconds for the graphic to update when selecting field point curves.

The individual plots and labels are based on the standard MTF analysis tools.

Field Points Sampling

The ability to plot data for up to 9 field points is common to all report graphics. By default, the plotted field points are taken from the Field Points Set.

The field points are plotted only if they comply with the following rules:

- They must be defined strictly on the Y-axis (FBX = 0.0).
- They must be defined for the current configuration. In other words, if the current configuration is 2, all field points defined with a configuration number different from 2 and 0 (0 is equivalent to "all configurations") are ignored.

You are given 3 options as to which field points should be plotted:

- **Take from field point set** - the data is taken from the field point set. If more than 9 field points qualify, the first 9 are selected.
- **Field points from field points set (integers)** - use this option if you need to select a subset from the **Take from field point set** field points. You need to enter the field point number, i.e. an integer.
- **Direct value input** - if you need to plot field points that are not defined in your Field Points Set. The value you enter for each field point is the relative object height (FBY).

Frequency

The plot is calculated for a single frequency. You have the option of entering the desired frequency here. If 0.0 is entered, the value of the **sdtf** global variable will be used. Frequency is in *Cycles per mm* for focal systems and *Cycles per radian* for afocal systems.

Maximum Defocus (relative to IMS)

The maximum (+) and minimum (-) defocus value to be applied to the image surface in the MTF versus Defocus plot. The Defocus is represented in lens units for focal systems. For afocal systems, this analysis will not be performed and an error message will appear below the command line in the spreadsheet window.

Print/Plot OTF

This menu option computes a transfer function using the current spot diagram information.

Calculate optical transfer function

Type of transfer function
☒ Through-frequency ☐ Through-focus

Chromatic option
☐ Monochromatic ☒ Polychromatic

Wavelength number

Calculation type
☐ Print function ☒ Plot function

Scan direction
☒ Meridional ☐ Sagittal

Focus shift
 Maximum frequency in lines/mm
 Frequency increment
 Through-focus spatial frequency
 Maximum focus shift
 Focus shift increment

Points used to compute curve
 Number of focus positions used to compute curve

Plot diffraction limit
☒ Yes ☐ No

Type of transfer function

- **Through-frequency** - to compute the transfer function for a range of spatial frequencies on a fixed image surface.
- **Through-focus** - to compute the transfer function at a fixed spatial frequency for a range of focal shifted image surfaces. The through-focus option is only applicable to focal systems.

Chromatic option

- **Monochromatic** - type in the desired wavelength number in the Wavelength number cell.
- **Polychromatic** - which uses all the currently defined wavelengths according to their wavelengths weights.

Calculation Type

- **Print function** - a table of output values will be printed to the current text window.

- **Plot function** - a graphical display will be plotted in the current graphics window.

Scan direction

If you want text output, click the radio button for either Meridional (y) or Sagittal (x) transfer function data. Plotted output will give you both sets of information automatically.

Through-frequency transfer functions

A focus shift can be applied before computing the transfer function by typing the desired value in the Focus shift cell. The desired values for maximum spatial frequency and spatial frequency increment are entered in the Maximum frequency in lines/mm and Frequency increment cells. Click OK to compute the transfer function.

The output consists of four columns of data:

FREQUENCY - the spatial frequency at which the transfer function is displayed, in cycles/mm (focal systems) or cycles/radian (afocal systems)

MODULUS - magnitude of the optical transfer function

PHASE - phase angle (in degrees) of the optical transfer function

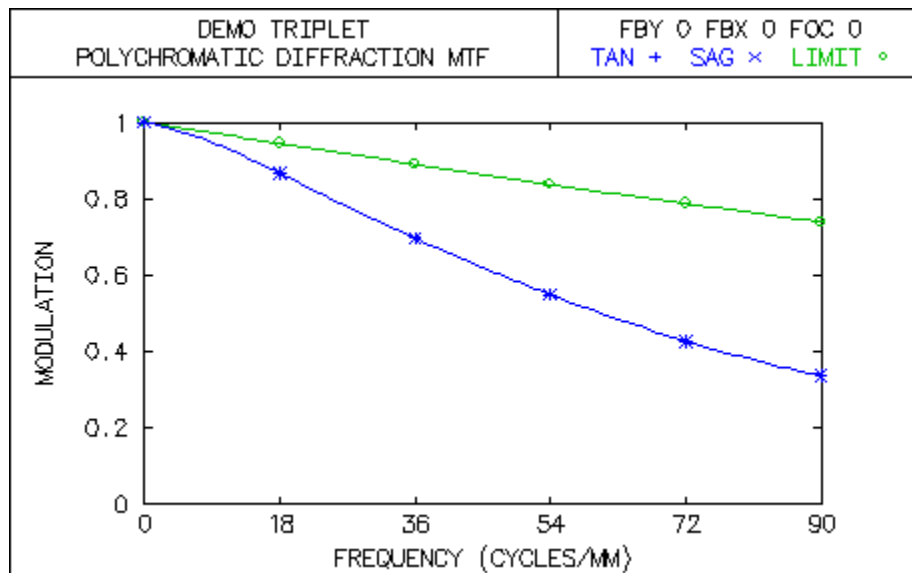
DIFF LIM MTF - magnitude of the optical transfer function for a diffraction-limited system

*POLYCHROMATIC MODULATION TRANSFER FUNCTION Y

NBR	FREQUENCY	MODULUS	PHASE	DIFF LIM MTF
1	--	1.000000	--	1.000000
2	9.000000	0.952934	--	0.974661
3	18.000000	0.874348	--	0.949322
4	27.000000	0.784419	--	0.923983
5	36.000000	0.697927	--	0.898644
6	45.000000	0.619885	--	0.873305
7	54.000000	0.549289	--	0.847966
8	63.000000	0.485071	--	0.822627
9	72.000000	0.427570	--	0.797288
10	81.000000	0.377729	--	0.771949
11	90.000000	0.336271	--	0.746610

If you want a plot of the through-frequency MTF, enter the desired focus shift, if any, and the maximum frequency of the plot. Click OK to display the MTF plot.

Three curves are plotted: the diffraction limit, and the tangential and sagittal MTF.



Through-focus transfer functions

The spatial frequency at which the transfer function is to be calculated is entered in the Through-focus spatial frequency cell. If this value is left at its default of 0, the "Spatial Frequency for through-focus MTF calculations" (**sdtf**) will be used. The desired values for maximum focus shift and focus shift increment are entered in the Maximum focus shift and Focus shift increment cells. Click OK to compute the transfer function.

The output consists of three columns of data:

FOCUS - the defocus from the nominal position of the image surface.

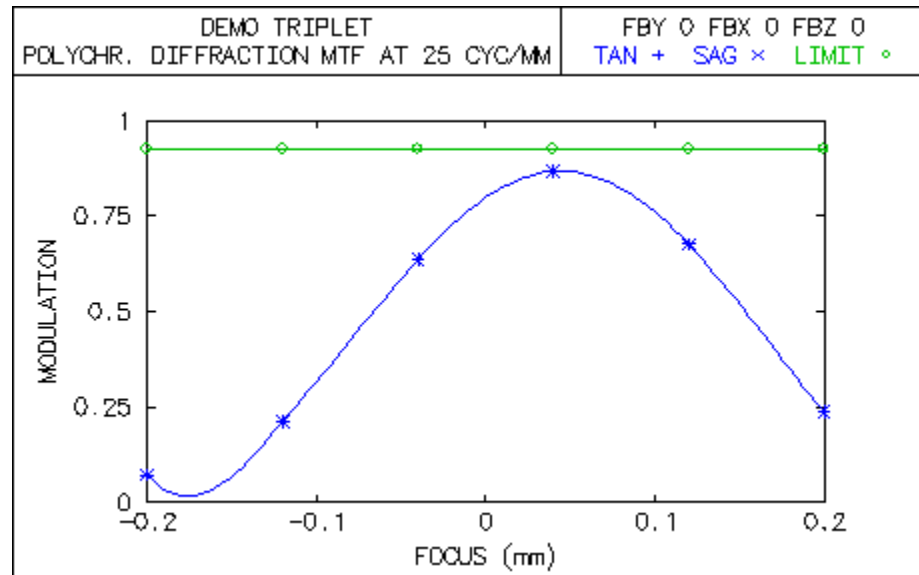
MODULUS - magnitude of the optical transfer function.

PHASE - phase angle (in degrees) of the optical transfer function.

```
*POLYCHROMATIC THROUGH-FOCUS MTF Y
SPATIAL FREQUENCY: 25 CYCLES/MM
NBR      FOCUS      MODULUS    PHASE
1        -0.500000   0.019427  180.000000
2        -0.400000   0.003471  180.000000
3        -0.300000   0.081792  180.000000
4        -0.200000   0.076617  180.000000
5        -0.100000   0.313004   --
6        -2.7756e-17  0.808095   --
7         0.100000   0.765557   --
8         0.200000   0.252751   --
9         0.300000   0.073197 -180.000000
10        0.400000   0.004964 -180.000000
11        0.500000   0.085866   --
```

If you want a plot of the through-focus MTF, enter the desired spatial frequency and maximum focus shift of the plot. Click OK to display the MTF plot.

Three curves are plotted: the diffraction limit, and the tangential and sagittal through-focus MTF.



Prm Print FFT-based OTF

MTF is available as a computation that includes polarization effects. The polarization ray trace can also be utilized by point spread function related calculations (*PSF*, diffraction *LSF* and *KED*, diffraction encircled/ensquared energy, etc.), so any effects of coatings were already included in these computations once the polarization ray trace itself was modified. This computation uses the routines to compute the polarization *PSF* over a grid of points and perform an *FFT* of the *PSF* to obtain the “vector diffraction” *OTF*.

The `fft_mod_trans_func {ftm}` command has the syntax

ftm (*monochromatic or polychromatic,*
wavelength (if monochromatic),
FFT grid points,
number of rays across the pupil,
maximum spatial frequency for output,
spatial frequency increment)

Because the *FFT* gives the entire *OTF*, the output of this command provides *MTF* and *PTF* for tangential, sagittal, and $\pm 45^\circ$ target orientations. The origin used is the centroid location of the *PSF*; this is different from the other *MTF* calculations that use the minimum RMS wavefront point. (Since the *FFT PSF* doesn't use the spot diagram information, RMS wavefront information is not available. However, since the entire *PSF* grid is known, the centroid location is known.) There should not be a large effect since an origin shift adds a linear phase term to the *OTF*, but there will be differences in the *PTF* values for the two calculations. Since *FFT* sampling problems are present, undersampled wavefronts may contain errors for low spatial frequency values. However, use of this routine to include polarization effects infers sampling a highly aberrated wavefront is probably not an issue.

Prm Plot FFT-based MTF

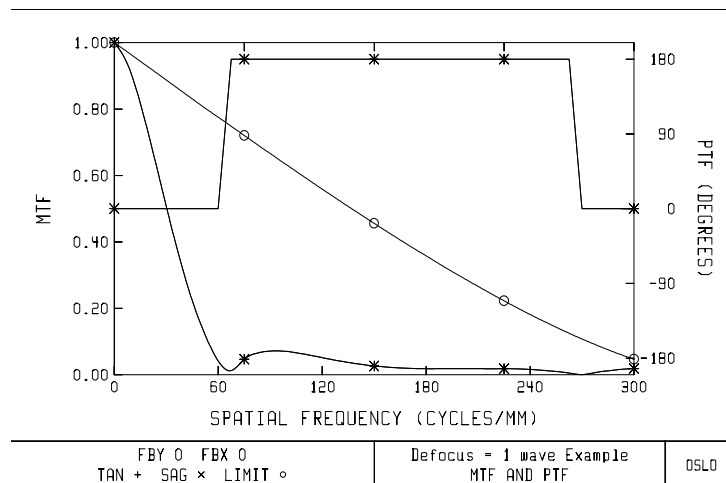
The **fftmftf** CCL command plots the data from the **ftm** output.

MTF and PTF

The **mtfandptf** command allows you to plot the modulation transfer function (*MTF*) and phase transfer function (*PTF*) for the current object point on the same graph. The command has the syntax

mtfandptf (*maximum_frequency*).

The solid lines and left-hand vertical scale are the *MTF*, while the dashed lines and right-hand vertical scale are the *PTF*. The *PTF* lies in the range $-180^\circ \leq PTF \leq +180^\circ$.



Square-Wave Response

You can plot the square-wave response at the current object point using the **squaremtf** CCL command with the syntax:

squaremtf (*maximum_spatial_frequency*, *number_of_frequencies*)

where *maximum_spatial_frequency* is the largest frequency at which the square-wave response will be calculated and *number_of_frequencies* is the number of data points used for the plot.

The response is the resulting modulation of the image of a unit-modulation square wave object as a function of the spatial frequency of the square wave. This can be contrasted to *MTF*, which is the modulation of a sinusoidal intensity object. The square-wave response is computed from the *MTF* by decomposing the square wave into its Fourier (i.e., sine wave) components. The square wave modulation $S(f_0)$ for a square wave of spatial frequency f_0 is given by⁶

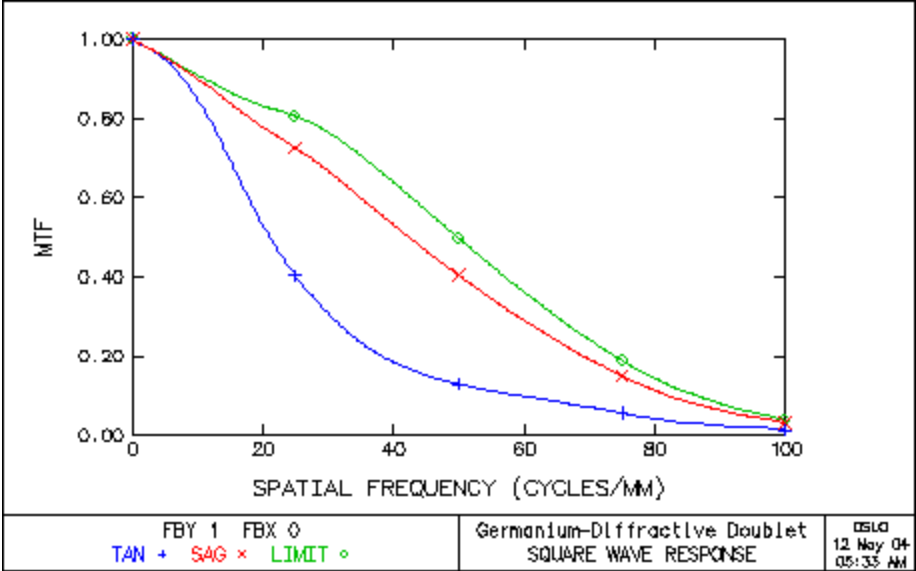
6. W. J. Smith, *Modern Optical Engineering*, Second Edition, McGraw-Hill, 1990, p. 355.

(8.8)

$$S(f_0) = \frac{4}{\pi} \left[MTF(f_0) - \frac{1}{3} MTF(3f_0) + \frac{1}{5} MTF(5f_0) - \frac{1}{7} MTF(7f_0) + \dots \right]$$

Note that the square-wave response at a frequency f_0 is generally higher than the MTF (sine wave response) at the same frequency. This is true for frequencies higher than one-third of the cutoff frequency due to the $4/\pi$ factor and since only the fundamental frequency contributes to $S(f_0)$.

The square-wave response requires more calculation time than the MTF since the summation requires that the MTF be computed at frequencies larger than the maximum frequency used in the plot. The command will prompt you for the data to print in the current text window in addition to the data needed for the plot.

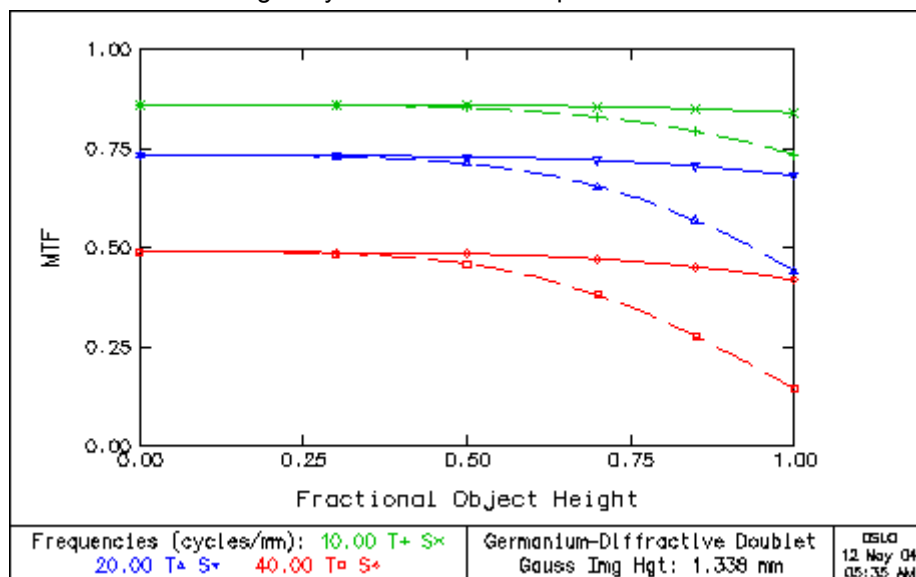


MTF vs. Field

Plots the MTF as a function of fractional object height for (up to) three spatial frequencies, as suggested by ISO. The command prompts for the spatial frequencies to use; enter "0" to omit a frequency. The default frequencies are 10, 20, and 40 cycles/mm.

Note that the MTF is calculated at fractional object heights of 0, 0.3, 0.5, 0.7, 0.85, and 1.0. Intermediate points are the result of a spline fit. The actual data points

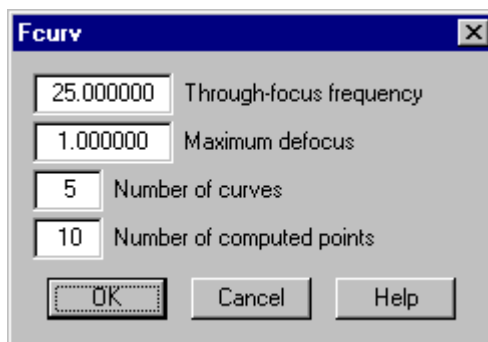
are marked with a symbol. Be aware that the spline fit may give misleading results if the actual MTF varies greatly between the data points.

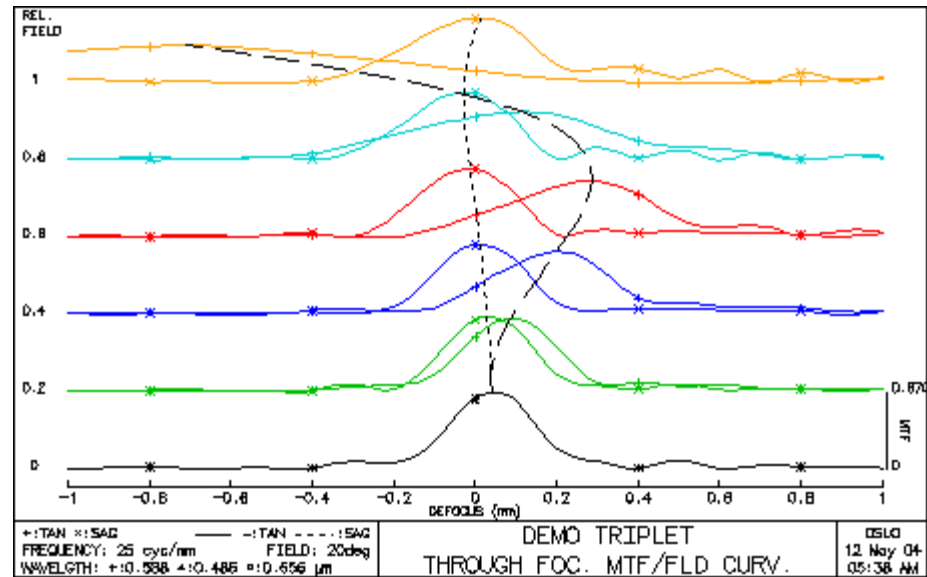


Through Focus MTF/Field Curvature

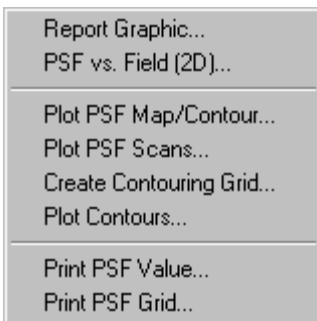
Plots the through-focus MTF curves from equally separated field points.

The maxima of the through-focus MTF curves are linked together by a curve which represents the field curvature.





Spread Function



The peak intensity is normalized to the peak of the un-aberrated PSF (the Strehl ratio).

Report Graphic

The PSF Report Graphic plots PSF information for several object/field points all located along the Y-axis. See the section "PSF vs. Field (2D)" on page 315 to plot object/field points in both X and Y.

The individual plots and labels are based on the graphic and text PSF analysis tools.

Field Points Sampling

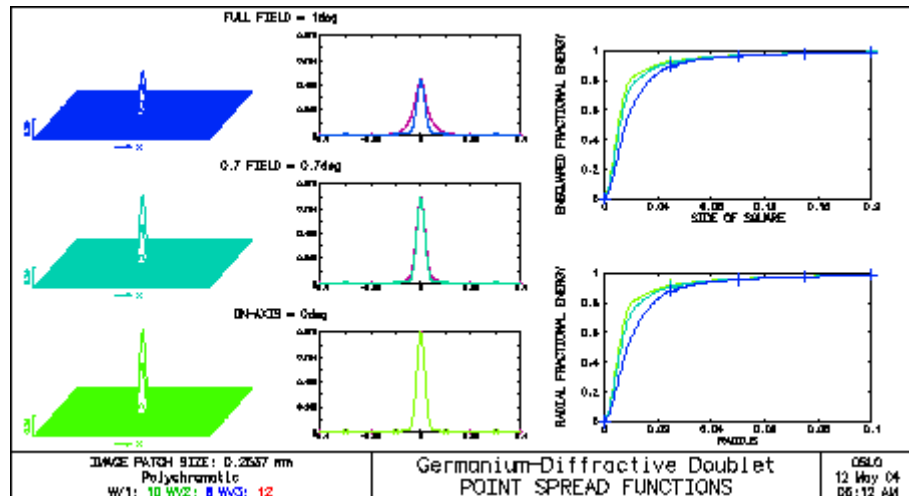
The ability to plot data for up to 9 field points is common to all report graphics. By default, the plotted field points are taken from the Field Points Set.

The field points are plotted only if they comply with the following rules:

- They must be defined strictly on the Y-axis (FBX = 0.0).
- They must be defined for the current configuration. In other words, if the current configuration is 2, all field points defined with a configuration number different from 2 and 0 (0 is equivalent to "all configurations") are ignored.

You are given 3 options as to which field points should be plotted:

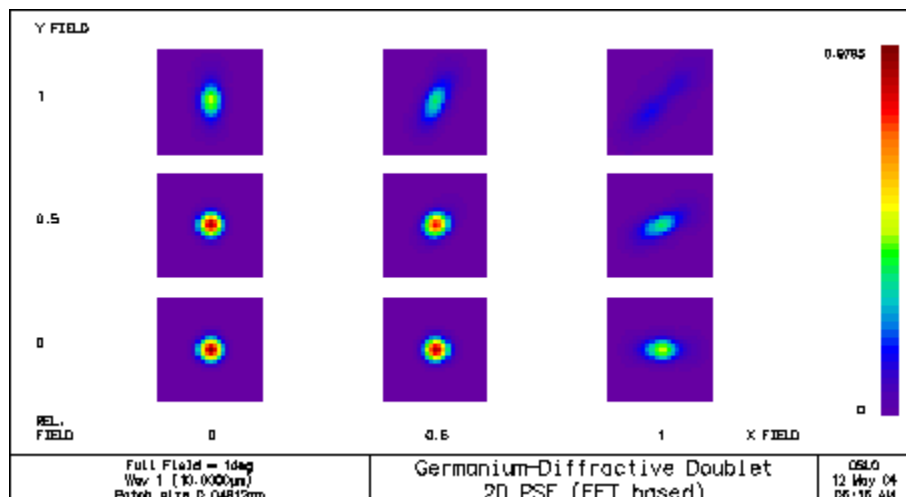
- **Take from field point set** - the data is taken from the field point set. If more than 9 field points qualify, the first 9 are selected.
- **Field points from field points set (integer)** - selects a subset from the **Take from field point set** field points. You need to enter the field point number, i.e. an integer.
- **Direct value input** - plots field points that are not defined in your Field Points Set. The value you enter for each field point is the relative object height (FBY).



Std Prm PSF vs. Field (2D)

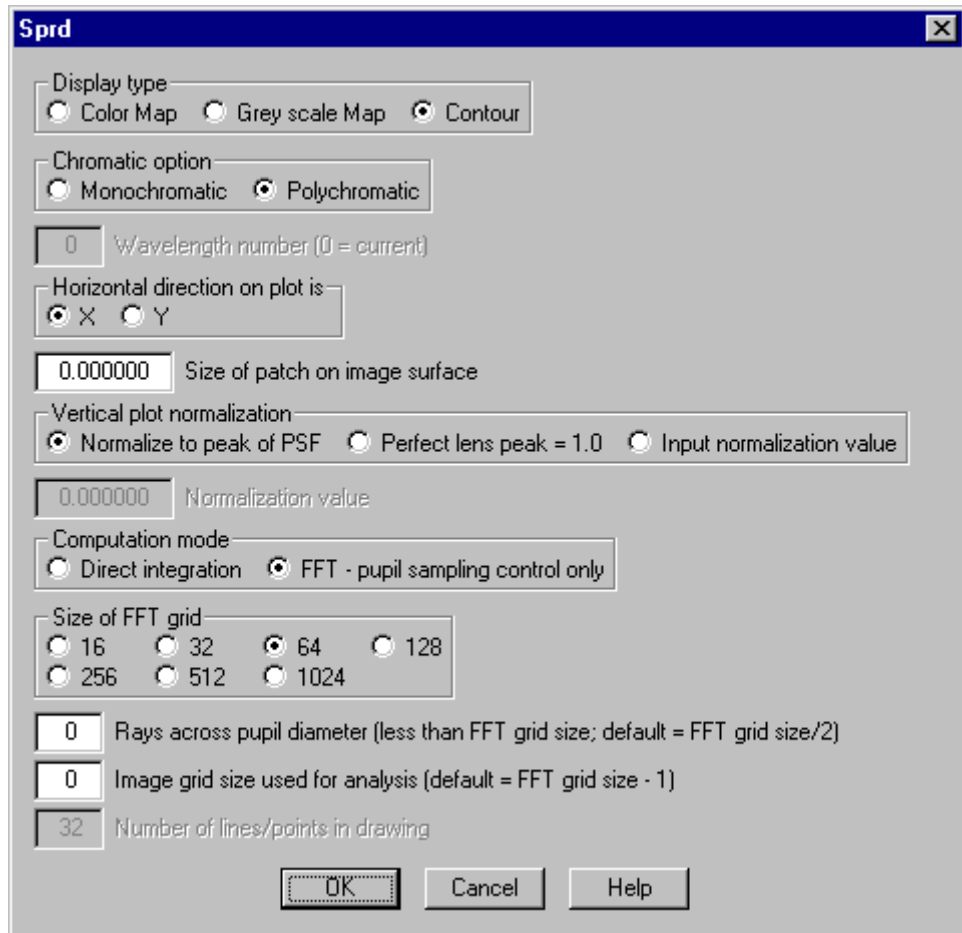
The 2D PSF Report Graphic plots SPF information for all object/field points defined in the field points set.

The individual plots and labels are based on the graphic and text PSF analysis tools.



Plot PSF Map/Contour

The **sprd** command plots a PSF **Color map**, **Grey scale map** or **Countour** plot in the current graphics window.



Computation mode

Direct Integration

While more accurate and easier to scale than the FFT-based calculation, it is also heavily computation intensive.

This computation mode is detailed in the section “Print PSF Value” on page 320.

FFT - pupil sampling control only

The most efficient way to compute a PSF.... if the sampling options are right.

This computation mode is detailed in the section “Print PSF Grid” on page 320.

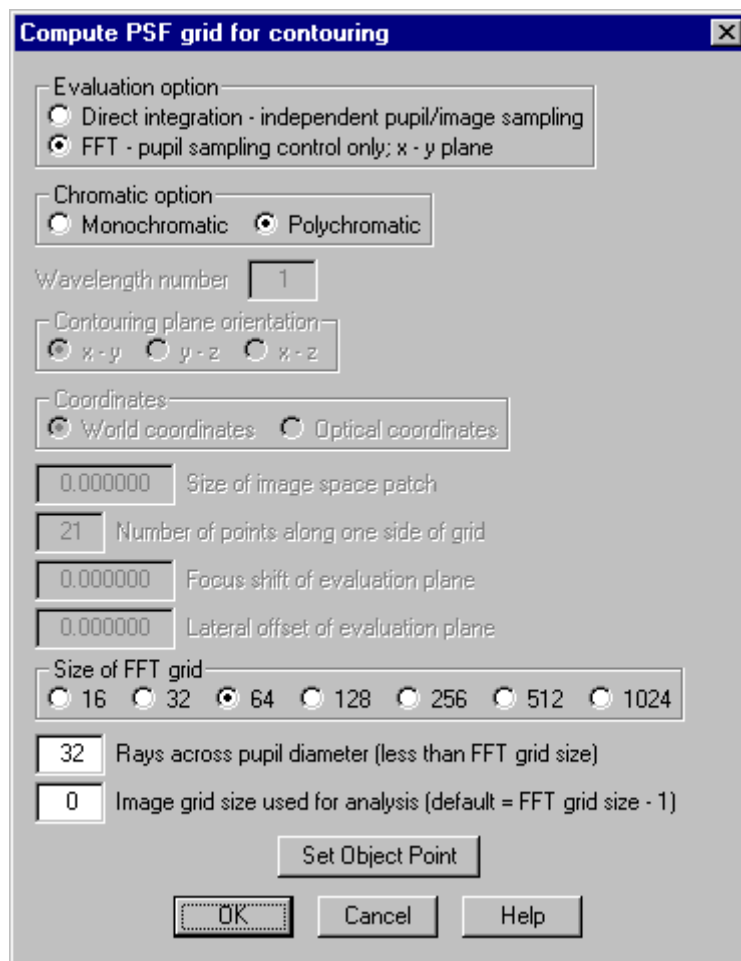
Plot PSF Scans

Plots X and Y cross scans of the PSF (computed by direct integration from the current spot diagram): PSF vs. X at Y = 0 and PSF vs. Y at X = 0 are plotted. Note that for a Gaussian (apodized) beam, the beam properties (spot sizes) must be

entered using the SETUP button in the lens data spreadsheet, or directly using the **sdgx** and **sdgy** commands.

Prm Create Contouring Grid

Point spread function (PSF) irradiance contours are displayed by first constructing a grid of PSF values. This grid is retained in memory, so that multiple contour plots may be drawn without having to recompute the image space PSF distribution. To create this grid, select Create Contouring Grid from the PSF Contouring pull-right menu.



The PSF may be calculated either by Direct (Huygens) integration or by using an FFT. (More details on these methods may be found in Chapter 6 – Image Evaluation of the Optics Reference manual.) The calculation may be either monochromatic or polychromatic. As with all PSF calculations, if the polarization ray trace is being used, the computed PSF will be a vector diffraction calculation.

If you select the Direct integration option, the plane of the grid may be oriented to be parallel to either the xy, yz, or xz planes. The current spot diagram is used to compute the PSF over a grid of points in the specified plane. The value entered in the Size of image space patch cell determines the length of one side of this square grid. The number of points in each dimension for the grid is entered in the

Number of points along one side of grid cell. If the observation plane is the xy plane, this plane may be shifted by the Focus shift of evaluation plane cell amount. The center of the grid is located at the point in the xy plane that minimized the variance of the wavefront. If you select a longitudinal grid (i.e., yz or xz) the plane may be offset by the Lateral offset of evaluation plane amount. Also, you may use the so-called "optical coordinates" to display the data. Using optical coordinates applies different scale factors to the lateral and longitudinal coordinates and generally results in more useful plots. As given in Born and Wolf,⁷ the optical coordinates u and v are

(8.9)

$$u = \frac{2\pi}{\lambda} (NA)^2 z$$

and

(8.10)

$$v = \frac{2\pi}{\lambda} (NA)r = \frac{2\pi}{\lambda} NA \sqrt{x^2 + y^2}$$

In Eqs. (8.9) and (8.10), λ is the wavelength and NA is the numerical aperture. In these coordinates, the boundary of the geometrical shadow lies along the lines $v = \pm u$. Using direct integration is slower than using an FFT, but more flexible. After the grid is constructed, the minimum and maximum PSF irradiance values in the grid are displayed. Also displayed are the location of the point of maximum irradiance, both relative to the center of the grid and relative to the reference ray intersection point.

*PSF CONTOUR GRID

GRID DIVISIONS: 41

MIN PSF	MAX PSF	MAX PSF Y	MAX PSF X	MAX Y (REF)	MAX X (REF)
0.001751	0.247430	-0.002100	--	-0.001563	--

If use an FFT to compute the PSF, the evaluation grid is limited to the xy plane. You may choose the Size of FFT grid and the Rays across pupil diameter. The sampling interval and image grid size is determined by the FFT sampling requirements. (See Chapter 6 of the Optics Reference manual.) The Image grid size used for analysis value determines how much of the resulting image plane grid is retained for the contouring analysis. The grid of PSF values is centered at the point of intersection of the reference ray with the image surface. After the grid is constructed, the minimum and maximum PSF irradiance values and the location of the point of maximum irradiance, relative to the reference ray intersection point, are displayed.

*PSF CONTOUR GRID

FFT GRID SIZE: 128

RAYs ACROSS PUPIL: 32

IMAGE GRID SIZE: 51

MIN PSF	MAX PSF	MAX Y (REF)	MAX X (REF)
0.000126	0.246037	-0.001459	--

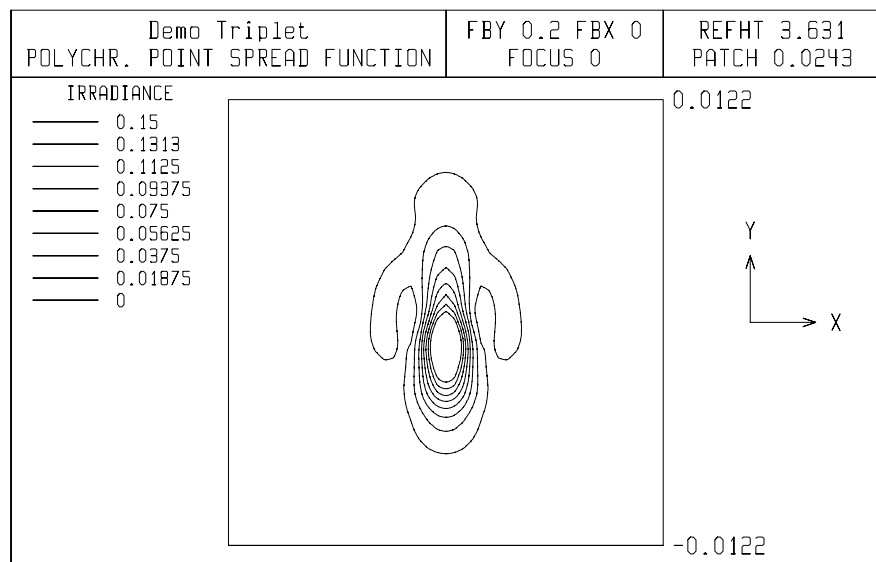
7. M. Born and E. Wolf, *Principles of Optics*, Sixth Edition, Pergamon, Oxford (1980), Section 8.8.

Prm Plot Contours

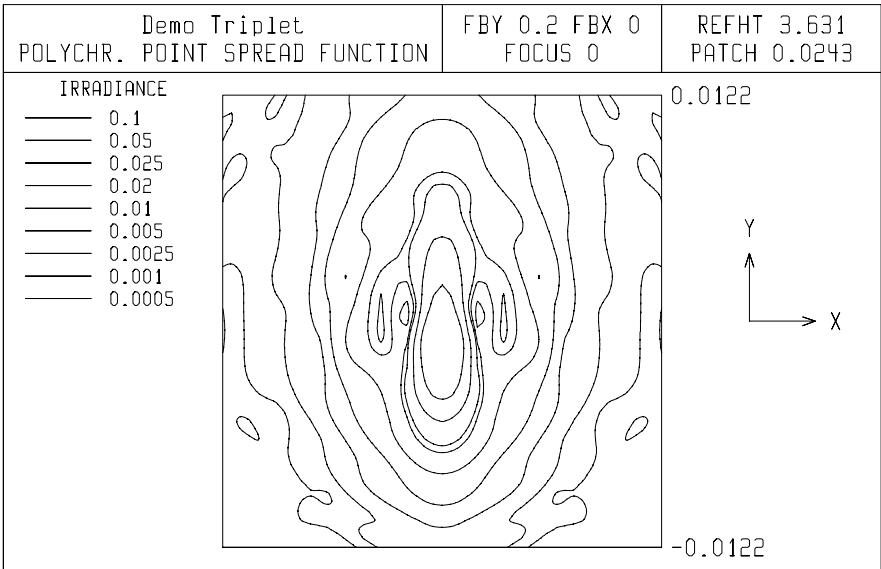
After constructing a contouring grid, select Plot Contours from the PSF Contouring pull-right menu to display a contour plot of the grid data.

The contour lines are the loci of equal point spread function irradiance values. The lines may be drawn as either all solid lines or with different lines styles. Using line styles is recommended if the contour plot will be displayed or printed on a black-and-white output device. The contours may be displayed as either relative irradiance values or in decibels (dB).

If you select the Equal contour increments option, you enter the Number of contour levels and the Minimum contour level value and Maximum contour level value that you wish to display. If you use the default values of 0.0 and 0.0, OSLO will use the minimum and maximum values in the grid of PSF irradiance values.



If you select the Specified contour levels option, the values entered in the Contour values spreadsheet cells will be used. Contour levels starting with the first level and ending with the last nonzero level are used.



Print PSF Value

The **point_spread_func (psf)** command calculates the Point Spread Function (PSF) using the Direct Integration method, and the prints the data to the current text window. This analysis is the "Text Output" equivalent to the graphical analysis in section "Direct Integration" on page 316.

This option Prints the value of the point spread function about the point specified by **Reference point**, **y** and **x coordinates of image point**, and **Focus shift**:

- **Min RMS wavefront** first shifts the image surface by **Focus shift**, then determines the point on the shifted surface about which the RMS OPD is minimized, and finally shifts this point by **y** and **x coordinates of image point**.
- **Reference ray** determines the intersection of the current reference ray with the image surface, then shifts this point by **y** and **x coordinates of image point** and **Focus shift**.

The **Y**, **X**, and **Z** coordinates of the reference point are printed (for afocal systems, the angular coordinates of the reference point, **YA** and **XA**, are printed instead of **Y**, **X**, and **Z**), followed by the **PSF** value (relative to that of an aberration-free system).

If **Chromatic option** is **Monochromatic**, the **AMPLITUDE** and **PHASE** (in degrees) of the PSF are also printed (phase is not defined if **Chromatic option** is **Polychromatic**).

Std Prm

Print PSF Grid

The **fft_psf_print (fpp)** command calculates the Point Spread Function (PSF) using the Fast Fourier Transform (FFT) method, and the prints a grid of data to

the current text window. This analysis is the "Text Output" equivalent to the graphical analysis in section "FFT - pupil sampling control only" on page 316.

Prints the grid of point spread function values computed using an *FFT*. Either the irradiance (monochromatic or polychromatic) or the amplitude and phase (monochromatic) may be displayed. Before the actual *PSF* data, the image surface grid cell size, peak value, and row and column of the peak value are displayed.

The output grids are displayed in "striped" format so that the grid values may be contained in the spreadsheet buffer (which has 10 columns). If the *FFT* was computed using an $N \times N$ grid, the first output stripe of irradiance data consists of rows 0 through $N - 1$, columns 0 through 9; the second stripe is rows 0 through $N - 1$, columns 10 through 19, etc. If you are displaying the amplitude and phase, two columns of data are required for each grid point, so the first stripe contains grid columns 0 through 4, the second stripe contains grid columns 5 through 9, ...etc.

The rows of the grid correspond to the y coordinate on the image surface and the columns of the grid correspond to the x coordinate. Thus, row 0, column 0 is the (x_{min} , y_{max}) corner, row 0, column $N - 1$ is the (x_{max} , y_{max}) corner, row $N - 1$, column 0 is the (x_{min} , y_{min}) corner and row $N - 1$, column $N - 1$ is the (x_{max} , y_{min}) corner. The grid represents a square area on the image surface; the length of a side of the square is $(N - 1) \times$ grid cell size (in lens units). The grid point at row $N/2$, column $N/2$ corresponds to the intersection point of the reference ray with the image surface.

Since the output is formatted in groups of 10 columns, an $N \times N$ *FFT* grid will require $\text{floor}(N/10) + 1$ stripes [or $\text{floor}(N/5) + 1$ stripes for an amplitude and phase grid]. Thus, the largest irradiance grid that will fit in the spreadsheet buffer is 128×128 ; the largest amplitude and phase grid is 64×64 . For larger grids, it is suggested that you turn **output logging** preference on, so that the text output will be sent to a file. You can then perform your analysis using the resulting data file.

Energy Distribution



A common technique for studying an image is to construct an energy distribution. This function approximates the physical process of scanning an expanding pinhole aperture or square aperture across a point image. The objective is to make a plot or table of energy uncovered by the aperture versus its size. OSLO can compute energy distributions using either a diffraction-based or a geometric calculation.

Diffraction

All of the diffraction-based energy distribution calculations are performed using an *FFT* to compute the point spread function. Thus, you may select the size of the *FFT* grid, the number of rays that are traced across the pupil diameter, and the grid size that is used for the energy analysis. See Chapter 6 – Image Evaluation of the Optics Reference manual for a discussion of the use of *FFTs*. All of the

calculations may be either monochromatic or polychromatic. The center of the expanding circle or square is located at the centroid of the point spread function.

If you print either the radial energy distribution or ensquared energy distribution, you can choose to display the data either in terms of equal energy increments or equal position increments. In either case, two columns of data are printed.

FRACTIONAL ENERGY - the fractional energy uncovered either by a circle of a given radius (radial energy) or a square of a given side length (ensquared energy)

RADIUS - the radius of a circle, centered on the centroid of the PSF irradiance (printed for radial energy distribution only)

SIDE OF SQUARE - the length of one side of a square, centered on the centroid of the PSF irradiance (printed for ensquared energy distribution only).

In addition to the energy information, some statistical data computed from the point spread function irradiance grid is displayed. The moment data is displayed for both y and x azimuths. Note that the moments are *central* moments, i.e., variations around the mean (centroid).

CENTROID - the shift of the centroid of the PSF irradiance, relative to the intersection point of the reference ray with the image surface

SECOND MOMENT - the second central moment, i.e., the variance σ^2 , of the PSF irradiance

THIRD MOMENT - the third central moment (M_3) of the PSF irradiance

FOURTH MOMENT - the fourth central moment (M_4) of the PSF irradiance

STD DEV - the standard deviation, σ , of the PSF irradiance

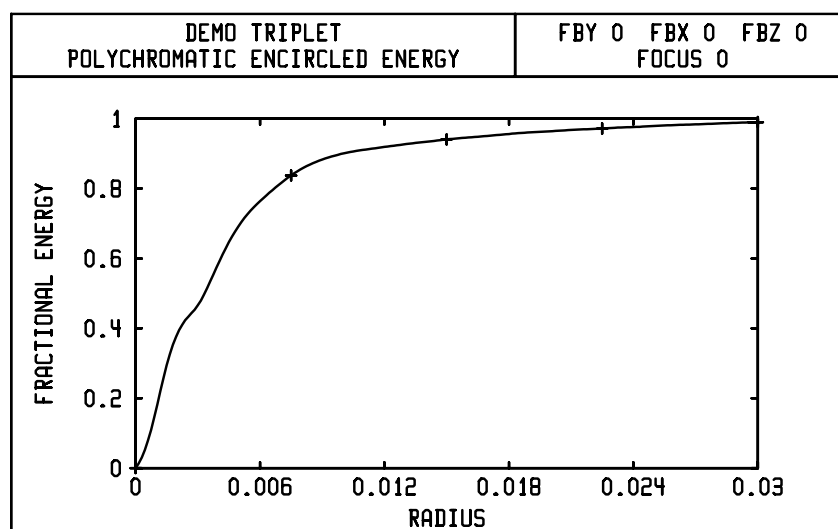
SKEW COEF - the skew coefficient of the PSF irradiance, i.e., M_3/σ^3

*ENCIRCLED ENERGY (DIFFRACTION) - POLYCHROMATIC

FRACTIONAL ENERGY	RADIUS
--	--
0.100000	0.000590
0.200000	0.001133
0.300000	0.001590
0.400000	0.002223
0.500000	0.003274
0.600000	0.004131
0.700000	0.005088
0.800000	0.006697
0.900000	0.010031

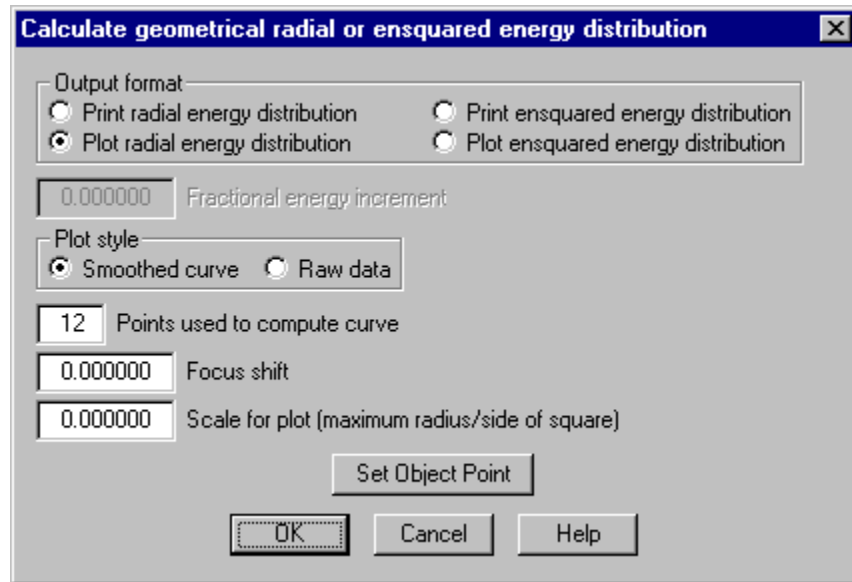
	Y	X		Y	X
CENTROID	4.1564e-15	1.0777e-19			
SECOND MOMENT	2.7194e-05	2.7194e-05	STD DEV	0.005215	0.005215
THIRD MOMENT	1.4043e-18	-2.9678e-23	SKEW COEF	9.9030e-12	-2.0928e-16
FOURTH MOMENT	9.2171e-09	9.2171e-09			

If you plot the radial or ensquared energy distribution, the plotted curve is the result of a spline fit to the numerical raw data. OSLO plots the fractional energy versus either the radius of the circle or the length of the side of the square.



Geometrical

The geometric energy distribution calculations use the spot diagram distribution on the image surface. The center of the expanding circle or square is located at the centroid of the spot diagram.



If you print either the radial energy or ensquared energy distribution, you can enter the fractional energy increment and a focus shift, if desired. The output consists of two columns of data:

ENERGY - the fractional energy uncovered either by a circle of a given radius (radial energy) or a square of a given side length (ensquared energy)

RADIUS - the radius of a circle, centered on the centroid of the (focus-shifted) spot diagram (printed for radial energy distribution only)

SIDE OF SQUARE - the length of one side of a square, centered on the centroid of the (focus-shifted) spot diagram (printed for ensquared energy distribution only).

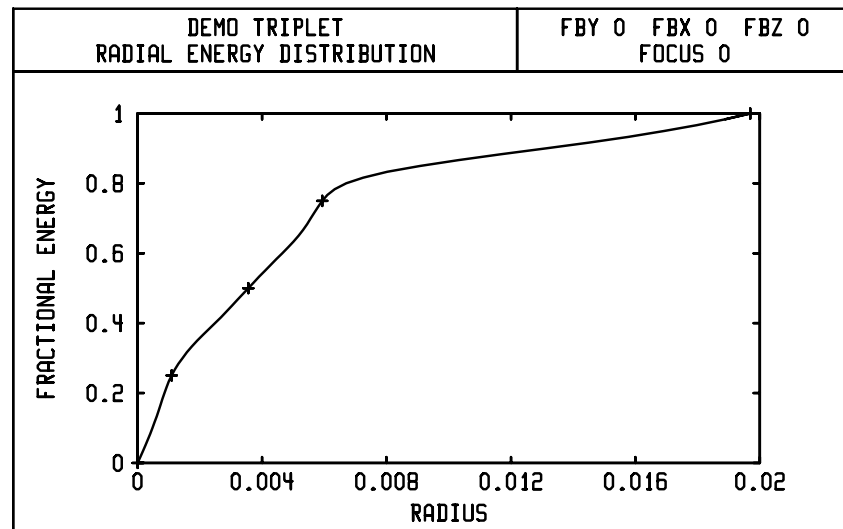
*RADIAL ENERGY DISTRIBUTION

ENERGY	RADIUS
0.100000	0.000550
0.200000	0.001008
0.300000	0.001501
0.400000	0.002394
0.500000	0.003733
0.600000	0.004809
0.700000	0.005719
0.800000	0.008261
0.900000	0.013596
1.000000	0.018624

If you plot the geometric radial or ensquared energy distribution, the plotted curve is the result of a spline fit to the numerical raw data if you select the Smoothed curve option. If you want to eliminate artifacts of the curve fitting calculation, select the Raw data option to just plot the data itself. The number of data points at which the curve is calculated is specified by Points used to compute curve. A focus shift

may be applied by entering a value in the Focus shift cell. Click OK to plot the radial energy or ensquared energy distribution curve.

For the geometrical energy distribution calculations, no account is taken of diffraction. The curve is computed by sorting the coordinates of all of the rays in the spot diagram (i.e., a polychromatic calculation) in order of increasing radius. Each ray is appropriately weighted so that the sum of all the weights is unity, and for each circle radius or square side length, the (weighted) number of rays enclosed by the circle or square indicates the fractional energy uncovered. If you want to include the effects of diffraction, choose a diffraction energy distribution calculation.



Line Spread/Knife Edge

The line spread function is the image formed by the lens of an infinitely long, infinitesimally thin line. It is computed by integrating the point spread function in one dimension. The knife edge distribution is the image of an infinite, one-dimensional edge. Alternatively, the knife edge distribution can be considered as the fractional energy uncovered by scanning a “knife edge” across the line spread function. OSLO can compute line spread functions and knife edge distributions using either a diffraction-based or a geometric calculation. See Chapter 6 – Image Evaluation of the Optics Reference manual for more information on these functions.

Diffraction

The diffraction-based line spread function (LSF) and knife edge distribution (KED) calculations are performed using an FFT to compute the point spread function. Thus, you may select the size of the FFT grid, the number of rays that are traced across the pupil diameter, and the grid size that is used for the analysis. All of the

calculations may be either monochromatic or polychromatic. The center for the calculations is located at the centroid of the point spread function.

Calculate diffraction line spread function and knife edge distribution...

Output format
☐ Print line spread and knife edge ☒ Plot line spread and/or knife edge

Item(s) to plot
☒ Line spread function (LSF) ☐ Knife edge distribution (KED)
☐ LSF and KED

Direction
☒ y ☐ x

Chromatic option
☐ Monochromatic ☒ Polychromatic

Wavelength number

Minimum image position for printed output

Maximum image position for printed output or plot

Number of image points

Size of FFT grid
☐ 16 ☐ 32 ☒ 64 ☐ 128 ☐ 256 ☐ 512 ☐ 1024

Rays across pupil diameter (less than FFT grid size)

Image grid size used for analysis (default = FFT grid size - 1)

If you print the LSF and KED you can choose the azimuth (y or x), the minimum and maximum positions for which you want to compute the functions, and the number of data points. The output consists of three columns of data:

POSITION - the position coordinate, either x or y, relative to the centroid of the point spread function

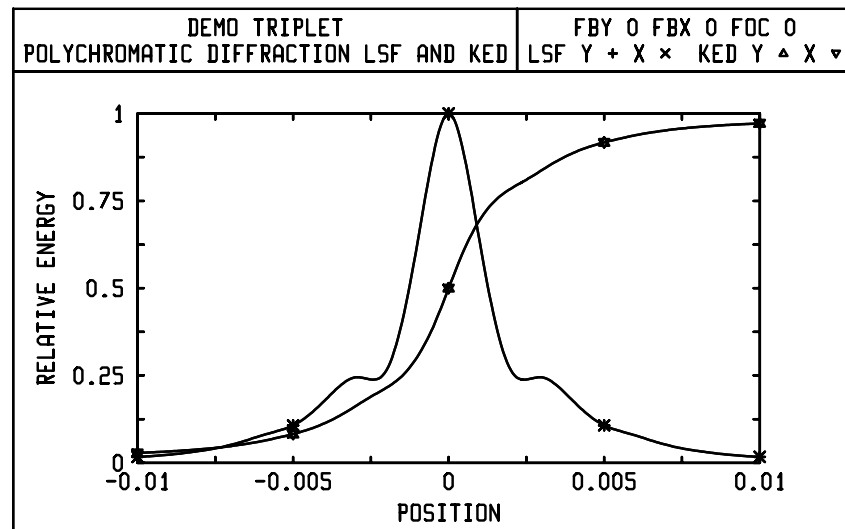
LSF - the value of the diffraction line spread function; the LSF is normalized such that the maximum LSF is 1.0

KED - the value of the diffraction knife edge distribution, i.e., the fraction of the total energy in the image that is uncovered by an infinite edge

*DIFFRACTION LSF AND KNIFE EDGE DISTRIBUTION - Y - POLYCHROMATIC

POSITION	LSF	KED
-0.010000	0.016888	0.028087
-0.008000	0.033900	0.038687
-0.006000	0.078463	0.062205
-0.004000	0.176990	0.112846
-0.002000	0.264708	0.214852
--	1.000000	0.500000
0.002000	0.264708	0.785148
0.004000	0.176990	0.887154
0.006000	0.078463	0.937795
0.008000	0.033900	0.961313
0.010000	0.016888	0.971913

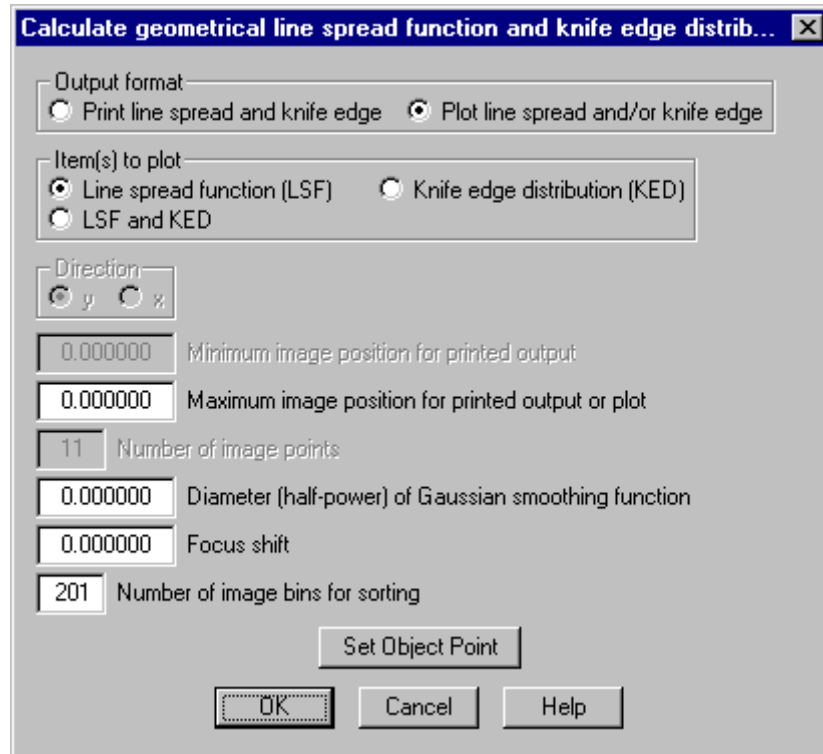
If you choose graphical output, you can plot either the line spread function, or the knife edge distribution, or both. Both x and y azimuths will be shown in the plot. The origin of the position axis is the centroid of the point spread function. The line spread functions are normalized such that the maximum LSF (in either x or y) is 1.0.



Geometrical

The geometric line spread function (LSF) and knife edge distribution (KED) calculations use the spot diagram distribution on the image surface. The center for the calculations is located at the centroid of the spot diagram. A focus shift may be applied by entering a value in the Focus shift cell. The functions are computed by sorting the spot diagram rays into "bins" and convolving the ray intersection densities with a Gaussian smoothing function. You can enter the number of bins used for sorting and the half-power diameter of the Gaussian smoothing function. The default is to use 201 bins. If the maximum total one-dimensional extent of the spot diagram is D , and the number of bins is N , then each bin corresponds to an

image surface increment of $\Delta = D/(N - 1)$. The default Gaussian smoothing diameter is 2Δ , so that each ray contributes to five sorting bins.



If you print the LSF and KED you can choose the azimuth (y or x), the minimum and maximum positions for which you want to compute the functions, and the number of data points. The output consists of three columns of data:

POSITION - the position coordinate, either x or y, relative to the centroid of the spot diagram

LSF - the value of the geometric line spread function; the LSF is normalized such that the maximum LSF is 1.0

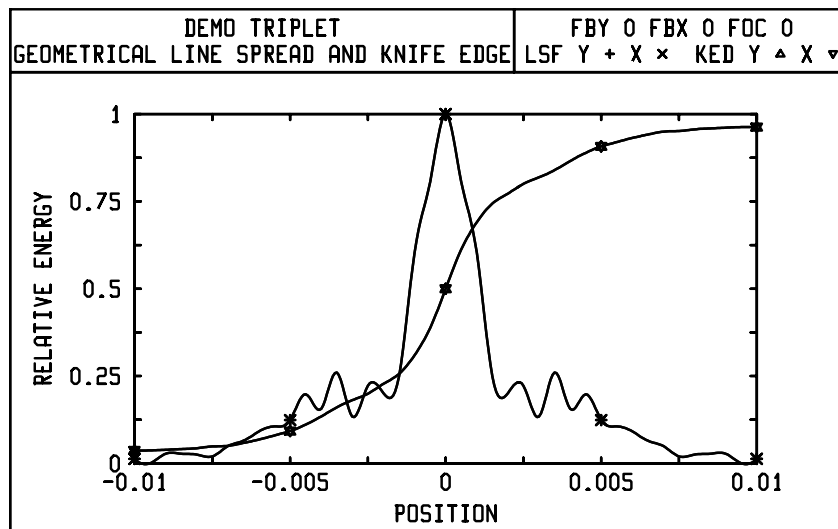
KED - the value of the geometric knife edge distribution, i.e., the fraction of the total energy in the image that is uncovered by an infinite edge

*GEOMETRICAL LSF AND KNIFE EDGE DISTRIBUTION - Y

POSITION	LSF	KED
-0.010000	0.013089	0.037122
-0.008000	0.025186	0.043703
-0.006000	0.091741	0.068206
-0.004000	0.156528	0.134554
-0.002000	0.201495	0.227734
--	1.000000	0.500000
0.002000	0.201495	0.772266
0.004000	0.156528	0.865446
0.006000	0.091741	0.931794
0.008000	0.025186	0.956297
0.010000	0.013089	0.962878

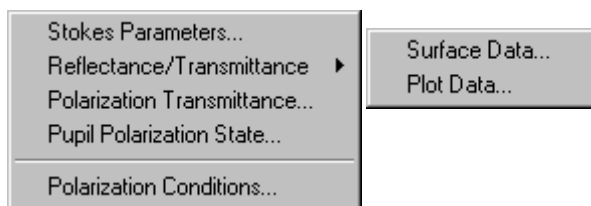
If you choose graphical output, you can plot either the line spread function, or the knife edge distribution, or both. Both x and y azimuths will be shown in the plot.

The origin of the position axis is the centroid of the spot diagram. The line spread functions are normalized such that the maximum LSF (in either x or y) is 1.0.



Polarization

Prm



Stokes Parameters

OSLO calculates the polarization state along a ray described by the Stokes parameters. This is in addition to the OSLO default display of the parameters that directly describe the polarization ellipse.

Display the Stokes output using User >> Special Analysis >> Stokes Parameters or the **stokes** CCL command with the syntax

stokes (*y_pupil*, *x_pupil*)

The Stokes parameters are useful because, among other reasons, they can be determined experimentally as functions only of observable quantities. In terms of the properties of the polarization ellipse, a_x , a_y , and δ (see Optics Reference, pp. 412–414), the total intensity I_t and degree of polarization P , the Stokes parameters (S_0 , S_1 , S_2 , S_3) are given by

(8.11)

$$\begin{aligned}
 S_0 &= I_t \\
 S_1 &= a_x^2 - a_y^2 \\
 S_2 &= 2a_x a_y \cos \delta \\
 S_3 &= 2a_x a_y \sin \delta
 \end{aligned}$$

where the intensity of the polarized portion of the light is $(S_1^2 + S_2^2 + S_3^2)^{1/2}$, so the degree of polarization is

(8.12)

$$P = \frac{\sqrt{S_1^2 + S_2^2 + S_3^2}}{I_t} = \frac{\sqrt{S_1^2 + S_2^2 + S_3^2}}{S_0}$$

Writing the Stokes parameters as the vector (S_0, S_1, S_2, S_3) and assuming unit intensity (i.e., $S_0 = 1$), we see that light that is linearly polarized in the x-direction is given by $(1, 1, 0, 0)$, while y-polarized light is $(1, -1, 0, 0)$. Linearly polarized light at 45° is $(1, 0, 1, 0)$ and linearly polarized light at -45° is $(1, 0, -1, 0)$. Right-handed circular polarization is $(1, 0, 0, 1)$ and left-handed circular polarization is $(1, 0, 0, -1)$. Completely unpolarized (natural) light has the Stokes parameter representation $(1, 0, 0, 0)$.

*STOKES PARAMETERS

	S0	S1	S2	S3
1	1.000000	-1.000000	--	--
2	0.965402	-0.965402	--	--
4	0.965402	0.297431	0.729097	0.558528
5	0.965402	0.135864	-0.822378	0.487070
3	0.965402	0.135864	-0.286563	0.911824
6	0.932001	0.131164	-0.276648	0.880277
7	0.932001	0.131164	-0.276648	0.880277
8	0.932001	0.131857	-0.276319	0.880277
9	0.932001	0.131857	-0.276319	0.880277
PUPIL	FY	FX		
	0.500000	0.500000		

Reflectance/Transmittance

The Polarization Transmittance command is used to compute the fraction of the incident beam that is transmitted by each surface in the current optical system. The computation is based on the rays that propagate from object to image; vignetting is not included. If vignetting was included in this calculation the results would be unclear: you could not separate vignetting losses, which you can get from the percent weighted ray transmission spot diagram output, from

polarization-dependent losses. The beam is traced from the current object point. This command is available using Options >> Polarization Transmittance or **ptn** command.

There are two lines of output for each surface. The first line, labeled **INT**, is the fraction of the beam that is transmitted *internally* by the medium *preceding* that surface (using the same convention with regard to surface number as the *D* ray trace item.) The second line, labeled **SRF**, is the fraction of the incident beam that is transmitted by the surface. (“Transmitted” in this sense could mean reflected; that is, the light continues on to the next surface.)

Note that the values for each surface are *local*; the overall transmission, broken down into internal (**INT TOT**), surface (**SRF TOT**), and total transmission (**TRANS**), is displayed separately.

The incident beam is defined to have a total intensity of 1.0, so the values reported by the Polarization Transmittance command are relative to a total beam intensity of 1.0, incident at the first surface of the lens. You can control the number of rays traced and the wavelength(s) used by setting the spot diagram operating conditions to desired values. The transmittance analysis will trace rays according to the current values of the “Aperture divisions across pupil” and “Use all wavelengths in spot diagram” operating conditions.

This analysis always uses polarization ray tracing, even when the polarization ray trace operating condition is off. If an input polarization state has not been defined, that is, no polarization operating conditions are set, unpolarized light is assumed. In addition to this analysis, polarization ray tracing may be used to compute the state of polarization along the path of a single ray or to compute vector diffraction patterns.

Surface Data

The **surf_refl_trans (srt)** command OSLO provides data from the surface analysis and prints it to the current text window. The calculation analyzes the reflectance and transmittance of a coated or uncoated surface as a function of angle of incidence or wavelength. This analysis is independent of ray tracing and is a characteristic matrix-based calculation. For each data point, the *s*, *p*, and average reflectance and transmittance are displayed in the first output group. The phase changes for *s* and *p* on reflection and transmission along with the retardation ($\phi_s - \phi_p$) and diattenuation $(S - P)/(S + P)$ for reflection and transmission are displayed in the second output group.

By default the phase change angles are in the range $-180^\circ < \phi \leq 180^\circ$. If the *Pos_phase_change_ang {ppca}* preference is *on* (available using Options >> Set Preferences), the phase change angles are always reported as positive values, i.e., $0^\circ \leq \phi < 360^\circ$.

If a lens has a coating with a non-uniform thickness model, or there is gradient index media on one or both sides of the interface, entering explicit *y* and *x* position values on the surface affects the results. If there is no coating or gradient index, entering explicit *y* and *x* position values has no effect on the results.

Plot Data

The **surfprop** command plots the data from the **srt** output table.

Polarization Transmittance

For each surface in the lens, the Polarization Transmittance command displays the intensity of the transmitted beam (relative to an incident intensity of 1.0). If polychromatic spot diagrams are being used, the output is shown for each wavelength, and for the average of all wavelengths.

*OPERATING CONDITIONS: POLARIZATION

Use polarization raytrace:	On	Degree of polarization:	--
Ellipse axes ratio:	--	Y to major axis angle:	--
Handedness of ellipse:	Right	Use 1/4 wave MgF2 coating:	Off

*TRACE REFERENCE RAY

FBY	FBX	FBZ			
--	--	--			
FYRF	FXRF	FY	FX		
--	--	--	--		
YC	XC	YFS	XFS	OPL	REF SPH RAD
--	--	0.130554	0.130554	63.048232	53.020166

*TRANSMITTANCE ANALYSIS: POLYCHROMATIC

APDIV	17.030000			
SRF	WV1	WV2	WV3	SUM(1-3)
1	0.943891	0.942903	0.944323	0.943706
2	0.890965	0.889101	0.891782	0.890616
3	0.841261	0.838038	0.842641	0.840647
4	0.794521	0.790103	0.796397	0.793674
5	0.749997	0.745047	0.752111	0.749052
6	0.707882	0.702480	0.710201	0.706854
7	0.707882	0.702480	0.710201	0.706854

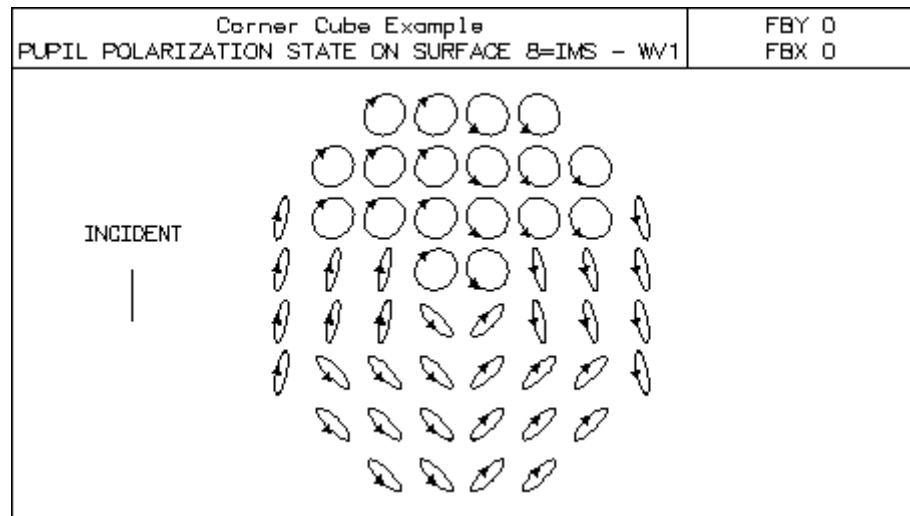
Pupil Polarization State

This **polstate** CCL command provides a graphical representation of the state of polarization across the exiting wavefront for the current object point. The **polstate** CCL command has the syntax

polstate (*rays_across_pupil*)

A grid of rays is traced through the lens and the polarization ellipse is drawn for each successfully traced ray. The incident polarization state (specified by the polarization operating conditions) is also displayed. The “arrowhead” drawn on the ellipse indicates the handedness of the polarization. (Clockwise is right-handed polarization; counter-clockwise is left-handed). The length of the major axis of the ellipse is proportional to the intensity of the polarization ray.

Refer to Chapter 12 of the Optics Reference for information on the polarization ellipse.



Polarization Conditions

The polarization operating conditions determine the state of polarization of the reference ray. For other rays from the same object point, the polarization properties are those of an electric dipole (in the wave or radiation zone) that has an orientation and phase consistent with the specification of the polarization properties of the reference ray. There are two choices for the intensity distribution of rays from the object point. The intensity may be either uniform or that of an electric dipole. In the wave zone of a dipole, the intensity is proportional to $\sin^2(\theta)$, where θ is the angle between the direction of radiation and the dipole axis.⁸

The polarization operating conditions are used to define the input state of polarization.

Use polarization raytrace:	<input checked="" type="radio"/> Yes	<input type="radio"/> No
Degree of polarization:	<input type="text" value="1.000000"/>	
Polarization ellipse minor axis/major axis ratio:	<input type="text" value="0.000000"/>	
Angle between object y-axis and ellipse major axis:	<input type="text" value="0.000000"/>	
Handedness of ellipse:	<input checked="" type="radio"/> Right <input type="radio"/> Left	
Use 1/4 wave MgF2 coating:	<input type="radio"/> Yes <input checked="" type="radio"/> No	
Include internal transmittance:	<input type="radio"/> Yes <input checked="" type="radio"/> No	
Use electric dipole intensity:	<input checked="" type="radio"/> Yes <input type="radio"/> No	

Use polarization raytrace (pzrt)

Selects the use of the polarization ray trace. The polarization operating conditions affect the ray trace only when this option is On.

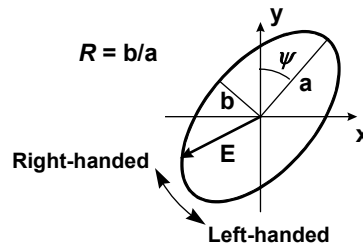
8. M. Born and E. Wolf, *Principles of Optics*, Sixth Edition, Pergamon, 1980, §2.2.3, pp. 81–84.

Degree of polarization (pzdp)

The degree of polarization of the incident beam. A value of 1.0 indicates completely polarized light and a value of 0.0 indicates completely unpolarized light.

Polarization ellipse minor axis/major axis ratio (pzer)

The ratio of the minor axis of the polarization ellipse to the major axis of the polarization ellipse. For each field point, the polarization ellipse is defined to lie in a plane normal to the reference ray, with its x-axis perpendicular to the plane defined by the reference ray and an axis parallel to the y-axis of the entrance pupil. A value of 0.0 indicates linearly polarized light, while a value of 1.0 means circular polarization. In the figure below, this item is shown as R .



Angle from object y axis to ellipse major axis (pzea)

The angle (in degrees) between the y-axis of the plane of the polarization ellipse and the major axis of the ellipse. In the figure above, this angle is shown as ψ .

Handedness of ellipse (pzlh)

The polarization handedness of the beam. When viewed from the direction to which the beam is propagating the electric vector is rotating clockwise for a right-handed beam and counter-clockwise for a left-handed beam, as shown in the figure above.

Use 1/4 wave MgF2 coating (pzmf)

If this option is chosen, refracting surfaces are taken to have a quarter-wave (defined at wavelength 1) optical thickness of magnesium fluoride anti-reflection coating.

Autofocus

Paraxial focus
 Minimum on-axis spot size (monochromatic)
 Minimum on-axis spot size (polychromatic)
 Field-averaged spot size (monochromatic)
 Field-averaged spot size (polychromatic)
 Minimum on-axis RMS OPD (monochromatic)
 Minimum on-axis RMS OPD (polychromatic)
 Field-averaged RMS OPD (monochromatic)
 Field-averaged RMS OPD (polychromatic)

Paraxial Focus

Minimum on-axis spot size (monochromatic)

Minimum on-axis spot size (polychromatic)

Std Prm Field-averaged on-axis spot size (monochromatic)

Std Prm Field-averaged on-axis spot size (polychromatic)

Minimum on-axis RMS OPD (monochromatic)

Minimum on-axis RMS OPD (polychromatic)

Std Prm Field-averaged on-axis RMS OPD (monochromatic)

Std Prm Field-averaged on-axis RMS OPD (polychromatic)

The Autofocus option allows you to adjust the image distance in your lens to optimize its performance based on a real spot size criterion. The spot size is computed in wavelength 1 for an on-axis object point, using the criteria chosen. The pupil is sampled using a Lobatto integration scheme (see the section in Chapter 9, "Generate Error Function"). If necessary, the autofocus calculation will trace rays through the entire pupil, expanding the range of this option to include non-rotationally-symmetric lenses. The thickness of the image surface (IMS) will be adjusted to the defocus value that minimizes this computed spot size. Autofocus is only valid for focal systems, since adjusting the image distance has no effect on the performance of afocal systems.

Overview



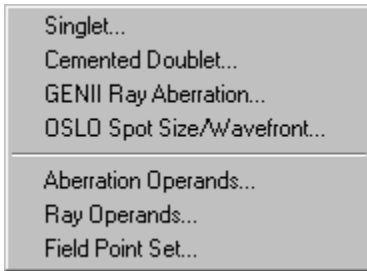
Optimization is the process by which the performance of a lens system is improved by changing the values of some of the lens data (*variables*) such that a measure of lens performance (*error function* or *merit function*) is minimized.

A successful design requires the specification of both variables for the optimization algorithm and an error function to determine the relative merit of various systems. Variables are specified in the variables spreadsheet editor, and the error function is specified in the operands spreadsheet editor and its supporting spreadsheet editors (ray set, field points set, and spot diagram set).

The menu commands relating to optimization are:

Menu Item	Page
Generate Error Function... is used to access different methods for automatically generating an error function	338
Error Function Tables... accesses the Field Point, Ray and Spot Diagram tables which govern how rays are traced for different components of the optimization error function	350
Operands... opens the operands spreadsheet editor; operands are the components that make up the error function	353
Variables... opens the spreadsheet editor; variables are the lens data that are changed by the optimization algorithm	366
Slider-Wheel Design... allows you to interactively adjust the construction data of the lens and immediately evaluate the lens performance	374
Iterate... uses the damped-least-squares (DLS) optimization algorithm to begin optimizing your lens	376
Advanced Optimization... accesses advanced local and global optimization engines and also allows you to view the current state of the derivative matrix	378
Optimization Conditions... opens the optimization operating conditions spreadsheet which allows you to control the optimization process	387
Support Routines... accesses vignetting, aperture setting, ghosting and relative illumination evaluation routines	389

Generate Error Function



Singlet

This menu item performs the **singlet_erf** CCL command which is used to generate an error function for a single-element lens. The error function contains just 2 terms. It tries to put the image plane at the paraxial focus, and correct the 3rd-order spherical aberration, while holding the numerical aperture and the focal length. It is not generally possible to do this (depending on the refractive index), but the error function should determine an optimum.

To use this function, you should:

1. Set up a singlet lens in the surface data spreadsheet, with no "dummy" surfaces (the image surface must be surface 3).
2. Use the Setup spreadsheet to adjust the aperture and field according to your specifications. Make sure that both the aperture and field are specified in object space, not image space (i.e. make sure that the aperture is specified by either Entrance Beam Radius or Object N.A., and the field is specified by Object Height or Field Angle).
3. Make radius of curvature 1 and Thickness 2 variables. The Error Function Generator will automatically insert a solve on surface 2 to hold the aperture and focal length.

Use the "Optimize>>Iterate" command to iterate. After the error function has been minimized, you can use the Autofocus (auf) command to find an optimum image location. This command uses the standard **opcb_abs** callback to evaluate the error function.

Cemented Doublet

The **cemdoublet_erf** CCL command may be used to generate an error function for a doublet with a cemented inner surface. The error function uses data from just one exact ray to correct the marginal spherical aberration and uses the offence against the sine condition (OSC) to correct the coma. To use this error function, you should

1. Enter a cemented (i.e. 3-surface) doublet lens in the surface data spreadsheet. Use crown and flint glasses of your choice.
2. Mark Radii of curvature 1 and 2 variable.
3. Put a $PY = 0$ solve on surface 2. The generator will automatically put an angle solve on this surface to hold the focal length and aperture specification.

Use the **ite** command to iterate. After the error function has been minimized, you can use the Autofocus (auf) command to find an optimum image location. You can

experiment with different glass combinations to get the best performance or chromatic correction.

This command uses the standard **opcb_rays** callback to evaluate the error function. Note that this callback traces two rays, but only 1 is used here.

GENII Ray Aberration

This error function is based on the default error function of the former GENII program. OSLO operands are constructed to mimic the form of the GENII error function, which has a different construction than OSLO. In GENII, the error function M is constructed from *targets* and is defined by

(9.1)

$$M = \sum_{j=1}^N \left(\frac{A_j - D_j}{T_j} \right)^2$$

where A_j is the actual value (the value for the existing lens) of the j^{th} target and D_j is the desired value (the value for the program to work to) for the j^{th} target. T_j is the tolerance of the j^{th} target (for example, the acceptable amount that the j^{th} target is permitted to deviate from its desired value in either direction). N is the number of targets. $A_j - D_j$ is the amount that the j^{th} target is in error and $(A_j - D_j)/T_j$ is the error measured in tolerances, so $(A_j - D_j)/T_j$ is the number of tolerances that the j^{th} target deviates from its desired value. In other words, $1/T_j$ is a weighting factor on the error $A_j - D_j$. M is the weighted sum of the squared errors. In OSLO, cross-reference operand components and operands with zero weight are used to construct the actual operands for computing the error function value.

The default error function from GENII has a consistent set of targets and tolerances to control classical aberrations, i.e., it is assumed that the lens is rotationally symmetric. Color correction is performed using Conrady $D-d$ operands and there is no control on secondary color. Since it is designed to balance aberrations in a focal plane shifted from the paraxial image plane, the image distance should be allowed to vary during optimization. The GENII manual claims that, if the lens is capable of being diffraction limited, this error function can drive it there. If the lens is not capable of being diffraction limited, a reasonable aberration balance can be achieved. If the lens is $f/1.5$ or faster, this error function may fail because too few rays are traced. Also, this error function will not work for doublets.

The rays used in the error function are selected on the assumption that there will be some vignetting. If there is no vignetting, the off-axis rays should be moved further out in the aperture. For some lenses, better correction may be achieved by moving the axial marginal ray in to 0.9 or 0.95, rather than 1.0.

The command creates a field points set with three entries (FBY = 0.0, 0.7, and 1.0) and a ray set with 8 rays, as shown in the following table.

Ray No.	Type	FY	FX
1	Reference	0.0	0.0
2	Ordinary	1.0	0.0
3	Ordinary	0.8	0.0
4	Ordinary	-0.8	0.0
5	Ordinary	0.0	0.7
6	Ordinary	0.7	0.0
7	Ordinary	-0.7	0.0
8	Ordinary	0.0	0.65

Ray 1 (a real chief ray) is used to compute field curvature and distortion operands. Ray 2 is used for the axial marginal ray. Rays 3, 4, and 5 are the aperture rays for the 0.7 field point and rays 6, 7, and 8 are the aperture rays for the 1.0 field point. The aperture coordinates for these rays may need to be adjusted based on the desired vignetting, as discussed above.

Std**Prm**

The GENII error function is generated using Optimize >> GENII Error Function or the **geniierf** CCL command with the syntax

geniierf (*design_f_number*, *design_spatial_frequency*)

where *design_f_number* is the desired working *f*-number in image space and *design_spatial_frequency* is the spatial frequency (in cycles/lens unit) at which good contrast (50% or better) is desired. Note that this command deletes any existing field points, rays, or operands, so it is suggested that you save the current lens under a new name if you want to return to an existing error function. Also note that the *f*-number target is one of the generated operands, so you do not need a curvature solve on the last surface of the lens to maintain the *f*-number. It is strongly suggested, however, that the *f*-number of the starting point be close to the desired *f*-number of the design. By default, the *design_f_number* is 4 and the *design_spatial_frequency* is 30 cycles/lens unit.

The command generates a set with 43 operands, of which only 31 have non-zero weight. These 31 operands comprise the error function. The other operands (with zero weight) are used as intermediate steps in forming GENII-style target definitions. All tolerances are computed from the specified frequency (*design_spatial_frequency*) and the exit angle of the paraxial axial ray (*u'*; computed from *design_f_number*). The basic tolerance, from which the others are computed, is the tolerance on the transverse ray error for the on-axis marginal ray. This tolerance is set at

(9.2)

$$Dy = \frac{1}{6 * (design_spatial_frequency)}$$

Operands 1 through 7 are the various tolerances used in constructing the targets. The default tolerances (operands 6 and 7) for wavefront targets (i.e., *S2T*, *OPD* and *DMD*) assume that wavefront errors are measured in lens units. The values of

these tolerances will be adjusted to perform the conversion from wavelengths to lens units, if OSLO is currently displaying *OPDs* in wavelengths. Operands 8 and 9 are the *f*-number target. The remaining active operands are described in the table below.

Op Nbr	Description of Active Operands	Field	Tolerance
9	Exit angle of paraxial axial ray, u'		0.0001
10	Focus shift penalty	0.0	$3Dy$
11	Marginal transverse ray error on-axis		Dy
12	Marginal <i>OPD</i> on-axis		$u'Dy/3$
13	Marginal <i>DMD</i> for axial color		$u'Dy/3$
16	Percent distortion	0.7	1
17	Tangential field curvature (transverse measure)		$3(0.7)Dy$
18	Sagittal field curvature (transverse measure)		$3(0.7)Dy$
19	Primary aperture coma exact in field		$3.2u'Dy$
20	Transverse ray error in upper aperture		$4(0.7)Dy$
21	<i>OPD</i> in upper aperture		$u'Dy/3$
22	<i>DMD</i> for lateral color in upper aperture	0.7	$u'Dy/3$
23	Transverse ray error in lower aperture		$4(0.7)Dy$
24	<i>OPD</i> in lower aperture		$u'Dy/3$
25	<i>DMD</i> for lateral color in lower aperture		$u'Dy/3$
26	<i>x</i> component of transverse ray error for sagittal ray		$4(0.7)Dy$
27	<i>y</i> component of transverse ray error for sagittal ray (coma)		Dy
28	<i>OPD</i> on sagittal ray		$u'Dy/3$
31	Percent distortion	1.0	1
32	Tangential field curvature (transverse measure)		$3Dy$
33	Sagittal field curvature (transverse measure)		Dy
34	Primary aperture coma exact in field		$3.2u'Dy$
35	Transverse ray error in upper aperture		$4Dy$
36	<i>OPD</i> in upper aperture		$u'Dy/3$
37	<i>DMD</i> for lateral color in upper aperture		$u'Dy/3$

38	Transverse ray error in lower aperture	$4Dy$
39	OPD in lower aperture	$u'Dy/3$
40	DMD for lateral color in lower aperture	$u'Dy/3$
41	x component of transverse ray error for sagittal ray	$4Dy$
42	y component of transverse ray error for sagittal ray (coma)	Dy
43	OPD on sagittal ray	$u'Dy/3$

The distortion tolerance may be changed by changing the values of operands 14 and/or 29. By default, all of the operands in the table are entered with a weight of 1.0. The weights can be adjusted by the user.

EDU Lt

As described earlier, the first few operands of the GENII error function are used to store GENII tolerances, and the remaining operands are exact-ray operands that use the field points and ray set defined by the command. OSLO-EDU and OSLO Light, however, only have access to OCM operands and cannot use the exact-ray operands set up by *geniierf*.

To allow the use of the GENII error function with OSLO Light and EDU, a new command *geniierf_lt* has been written. The results using OCM operands instead of normal compiled operands are the same, but may typically take two times longer to compute. Considering the high speed of current computers, this is often not a point of concern.

The *geniierf_lt* command has a slightly different user interface and default values, compared to the original *geniierf* command. The principal change is that the new commands include arguments for the pupil data for the two off-axis field points and the distortion tolerances. The design f -number is no longer required. The pupil data are shown as operands having zero weight at the end of the operands table, so they can be seen. To change them, you must re-execute *geniierf_lt* with a question mark, so you will be prompted for the required data (or you can include them when you enter the command). The *geniierf_lt* uses system note 6 to save its ray data, so you should not use this note for any other purpose in a lens that uses *geniierf_lt*.

The design f -number is now derived directly from the lens data. It is thus important to have the paraxial properties of the lens set up properly before executing *geniierf* or *geniierf_lt*. The paraxial spreadsheet is helpful for setting up the required data. The error function itself holds the f -number at its starting value, so it is not necessary to use an angle solve on the last surface for this purpose. In addition, the last thickness should be allowed to vary so the error function can find the optimum image location. In spite of its simplicity, the GENII error function can achieve a high degree of image quality when used interactively, often at considerably greater speed than competing approaches. An example of the use of the GENII error function has been added to the OSLO Optics Reference text starting on page 242 (page 250 in the electronic version).

For the sake of consistency, the original *geniierf* command has been updated so that its user interface is the same as *geniierf_lt*, i.e. it maintains the current f -number and allows ray data as arguments.

Std Prm OSLO Spot Size/Wavefront

The generate error function command provides a way for constructing an operands set that measures the RMS spot size or RMS wavefront error (OPD) at the image surface. The spot size or OPD is averaged over the field and, optionally, over the defined wavelengths. The method of computing the spot size or OPD is described in Chapter 7 of the Optics Reference manual. It yields an efficient and accurate estimate of the spot size or OPD *for systems having rotational symmetry*. For systems that lack rotational symmetry, a square grid pattern may be generated for sampling the pupil.

To generate an error function, select Generate Error Function from the Optimize menu. A dialog box will be opened in which you can specify the parameters for the construction of the error function

The mean-square spot size and mean-square wavefront error can be described by integrals of ray aberration or OPD over field, pupil, and, optionally, color. By tracing appropriately selected rays, these integrals can be approximated accurately by sums of the squares of the aberrations of the rays. In this spreadsheet, you can select the number of samples across the field and pupil. The sampling across color, if desired, is done using the currently defined wavelengths. You only need to specify the number of samples (i.e., rays); the program computes the actual coordinates of the rays that need to be traced.

Field sampling

Sampling method - There are four methods available for choosing the field sampling points: Current, Gaussian, Radau, and Lobatto. The Lobatto method, which is the default, fixes two sample points at the ends of the integration interval. The Radau method fixes one sample at the lower endpoint. The Gaussian method fixes none of the sample points. For the field integral, the endpoints are fixed at 0 and 1 (i.e., on-axis and the edge of the field). The integration method is chosen by clicking the appropriate radio button. The number of samples determines both the accuracy of the integral estimate and the speed of its evaluation: a large number of samples yields higher accuracy but requires more time to evaluate (more rays are traced). Selecting Current uses the existing entries in the field points set for generating ray operands.

Number of field samples - This is the number of entries in the field points set that are to be used in calculating the spot size averaged over the field. The default is 3 field points.

Duplicate field samples - During the design of a symmetric lens, only the positive FBY side of the object field needs to be considered. Tolerancing breaks symmetry, so negative FBY field points should be used for tolerancing error functions. If you are generating field samples, there is an option to duplicate the field samples on the negative FBY side. (This is not necessary for the design phase of a symmetric system, only tolerancing).

Pupil sampling

Sampling method - There are four methods available for choosing the pupil sampling points and ray weights: Gaussian, Radau, Lobatto, and Grid. The Lobatto method, which is the default, fixes two sample points at the ends of the

integration interval. The Radau method fixes one sample at the lower endpoint. The Gaussian method fixes none of the sample points. For the radial pupil integral, the endpoints are fixed at 0 and 1 (i.e., the center of pupil and the edge of the pupil, using the vignetting data for the field point). The integration method is chosen by clicking the appropriate radio button. The number of samples determines both the accuracy of the integral estimate and the speed of its evaluation: a large number of samples yields higher accuracy but requires more time to evaluate (more rays are traced). Choosing Grid sampling creates a set of samples that are arranged in a uniformly-spaced grid pattern.

Number of aperture divisions in pupil - If Grid sampling is used, this is the number of samples across the pupil in the sagittal and tangential directions. This number is equivalent to the number of aperture divisions across the pupil in a spot diagram.

Number of rings (on-axis field point) - This is the number of radial samples that should be generated for an on-axis field point (i.e., $FBY = FBX = FBZ = 0$), not counting the on-axis sample (the sample in the center of the pupil). The default is 5 rings. According to the Gaussian-integration scheme, the sum gives an exact value for the radial integral when the order of the integrand is less than or equal to $2N_r - 1$, where N_r is the number of radial sampling points. Thus, the number of sampling rings should be selected according to the highest order aberrations expected to be important for the system being designed.

Number of rings (off-axis field points) - This is the number of radial samples that should be generated, not counting the sample in the center of the pupil, for off-axis field points. The default is 3 rings.

Number of spokes (off-axis field points) - This is the number of angular pupil samples that should be generated. For an on-axis field point, only one spoke is generated. The default is 3 spokes.

Generate meridional spokes for off-axis field points - If Yes is selected (the default), a meridional spoke is generated (i.e., a spoke along the y-axis of the pupil, running from bottom to top). If No is selected, all pupil samples for off-axis field points represent skew rays.

Trace full pupil for symmetric lens - The Gaussian quadrature pupil sampling may also be applied to non-rotationally symmetric systems. The resulting ray pattern is still a “ring and spoke” pattern, but in this case the spokes span the entire pupil, rather than just the $FX > 0$ side. Also, the radial sample points and weights are computed without the assumption of an underlying symmetry in the lens.¹

Although the Gaussian quadrature scheme maintains its efficiency advantage over grid methods for the non-symmetric case, for a given level of accuracy, more rays need to be traced using the non-symmetric ray pattern. In fact, switching to a non-symmetry-based pattern reduces the order of aberration that can be exactly integrated by a factor of $\frac{1}{2}$.

By default, OSLO will use the symmetry of the lens to determine whether to generate rays over the entire pupil, or only over half of it. There are situations, however, when you may wish to generate rays over the entire entrance pupil even

1. The theory of Gaussian quadrature based error functions is described in G. W. Forbes, “Optical system assessment for design: numerical ray tracing in the Gaussian pupil.” J. Opt. Soc. Am. A **5**, 1943–1956 (1988). Note that the tables of Gaussian integration parameters given in this paper assume rotational symmetry of the lens. (See the discussion on p. 1948.)

if the lens is rotationally symmetric. The primary example of this is an error function that is to be used for user-defined or Monte Carlo tolerancing. Even if the nominal lens is rotationally symmetric, during the tolerancing procedure decentrations and tilts will break the symmetry and both sides of the pupil must be ray traced for accurate results. A radio button control in the spreadsheet may be used to force OSLO to ray trace both sides of the pupil, regardless of the symmetry condition of the lens.

Error function type

RMS spot size - The operands set (error function) generated will be an estimate of RMS transverse ray aberration, averaged over field, pupil, and, optionally, color. If the lens system is afocal, the error function will be an estimate of RMS aberration of the direction tangents of the rays in image space.

RMS wavefront error - The operands set generated will be an estimate of RMS wavefront error (optical path difference), averaged over field, pupil, and, optionally, color.

Color correction method

None - The error function will estimate the RMS aberration (transverse or wavefront) of rays in wavelength 1 only.

Use all wavelengths - The error function will estimate the RMS aberration of the rays in all of the currently defined wavelengths. Note that this is the most accurate but slowest color correction method.

Use CHR/DMD operands - The error function will estimate RMS aberration (transverse or wavefront) of rays in wavelength 1, plus RMS Conrady $D-d$ aberration. If RMS wavefront error is the chosen error function type, DMD operands are added to the operands set to estimate chromatic aberration. If RMS spot size is chosen as the error function type, CHR (transverse $D-d$) operands are added to the operands set.

Relative weight of CHR/DMD operands - If "Use CHR/DMD operands" is chosen as the color correction method, the mean-square error is given by (mean-square spot size or wavefront error) + (relative weight of CHR/DMD operands) * (mean-square CHR or DMD).

Error functions for multiconfiguration systems

Maximum operand configuration - The generated operands can be duplicated for multiple configurations by entering the maximum configuration number here.

Distortion correction

Correct distortion at full field - A DIST operand, computed at full field (FBY = 1) will be added to the error function if this option is selected. If other field points are to be used to correct distortion, they must be added manually.

Maximum allowed distortion at full field - This is the magnitude, in per cent, of the maximum allowed distortion at full field. One-sided operands will be used to keep the distortion between plus and minus the entered value. By default, 1% distortion is allowed (i.e., no penalty is added if the relative distortion is less than 0.01 or greater than -0.01).

Edge thickness correction

Generate edge thickness operands - If this option is chosen, operands will be added to the error function that penalize insufficient edge thickness. The operands that are used are simply two sets of path-length (PL) operands between successive pairs of surfaces along the (exact) upper-rim and lower-rim rays. That is, two exact rays are traced from full field, one passing through the top of the pupil ($FY = 1.0$) and the other passing through the bottom of the pupil ($FY = -1.0$). (Note that the vignetting factors in the field points set will affect the actual position of the rays on the reference surface.) The operands are set up to require each ray segment between successive surfaces to have lengths greater than the specified minimum value. If it is not desired to prevent insufficient edge thickness between a given pair of surfaces (e.g., if one of the surfaces is a dummy surface), the corresponding operands should be removed from the operands set (or given zero weight).

Minimum allowed edge thickness - This is the minimum acceptable value for edge thickness values as described above. The default minimum edge thickness is 0.05 current lens units.

Appending a generated error function

Append to existing error function - If this option is chosen, the generated error function operands will be appended to any existing operands. Otherwise, the existing operands will be deleted before the new error function is generated (default).

Aberration Operands

The **opabs_template** CCL command may be used to generate an error function based on the standard aberration coefficients computed by OSLO. The way it works is to generate a template error function containing all the operands that are evaluated by the **opcb_abs** callback. You build your own error function by deleting the operands that you don't want, either manually using the operands spreadsheet, or programatically such as is done in the **cemdoublet_erf** error function generator. The available operands are as follows.

Operand	Name	OCM#
Axial ray height	PY	OCM1
Axial ray slope	PU	OCM2
Chief ray height	PYC	OCM3
Chief ray slope	PUC	OCM4
Primary axial color	PAC	OCM5
Primary lateral color	PLC	OCM6
Secondary axial color	SAC	OCM7
Secondary lateral color	SLC	OCM8

3rd-order spherical	SA3	OCM9
3rd-order coma	CMA3	OCM10
3rd-order astigmatism	AST3	OCM11
3rd-order Petzval blur	PTZ3	OCM12
3rd-order distortion	DIS3	OCM13
5th-order spherical aberration	SA5	OCM14
5th-order linear coma	CMA5	OCM15
5th-order astigmatism	AST5	OCM16
5th-order Petzval blur	PTZ5	OCM17
5th-order distortion	DIS5	OCM18
7th-order spherical aberration	SA7	OCM19
Total (3rd+5th+7th) spherical	TOTAL_SPH	OCM20
Effective focal length	EFL	OCM21

If you want to have additional terms in your error function that cannot be expressed in terms of the OCM's shown in the above table, you will need to make another callback function to evaluate them. You can base this on a modification of the `opcb_abs` command (in `optim_callbacks.ccl`) that evaluates the above operands.

Ray Operands

The **oprays_template** CCL command may be used to generate an error function based on data computed for rays computed from 2 field points. The way it works is to generate a template error function containing all the operands that are evaluated by the **opcb_rays** callback. You build your own error function by deleting the operands that you don't want, either manually using the operands spreadsheet, or programatically such as is done in the **cemdoublet_erf** error function generator. The available operands are as follows.

Operand	Name	OCM#
On-axis fractional beam radius	AXIS_FYMAX	OCM1
Fractional height of off-axis object	OFAX_FOB	OCM2
Lower FY for off-axis object point	OFAX_FYMIN	OCM3
Upper FY for off-axis object point	OFAX_FYMAX	OCM4
FX for off-axis object point skew ray	OFAX_FX	OCM5
Paraxial axial ray height	PY	OCM6

Paraxial axial ray slope	PU	OCM7
Paraxial chief ray height	PYC	OCM8
Paraxial chief ray slope	PUC	OCM9
Effective focal length	EFL	OCM10
axis focus shift	AXIS_FOCUS	OCM11
axis FY zone dy	AXIS_ZDY	OCM12
axis FY zone opd	XIS_ZOPD	OCM13
axis FY zone dmd	AXIS_ZDMD	OCM14
axis FY zone osc	AXIS_ZOSC	OCM15
axis FY edge dy	AXIS_EDY	OCM16
axis FY edge opd	AXIS_EOPD	OCM17
axis FY edge dmd	AXIS_EDMD	OCM18
axis FY edge osc	AXIS_EOSC	OCM19
off-axis chief yc	OFAX_CYC	OCM20
off-axis chief yfs	OFAX_CYFS	OCM21
off-axis chief xfs	OFAX_CXFS	OCM22
off-axis upper FY dy	OFAX_UPDY	OCM23
off-axis upper FY opd	OFAX_UOPD	OCM24
off-axis upper FY dmd	OFAX_UPDMD	OCM25
off-axis lower FY dy	OFAX_LODY	OCM26
off-axis lower FY opd	OFAX_LOOPD	OCM27
off-axis lower FY dmd	OFAX_LODMD	OCM28
off-axis skew dy	OFAX_SKWDY	OCM29
off-axis skew dx	OFAX_SKWDX	OCM30
user-defined coma	OFAX_COMA	OCM31
user-defined distortion	OFAX_DIST	OCM32

In the above table, the first 5 operands are not normally included explicitly in an error function, but are used by the **opcb_rays** callback function to set up the rays traced to compute the operands. These operands are effectively parameters supplied to the **opcb_rays** callback, and in fact are just the arguments of the **oprays_template** command. They are listed in the template error function with

zero weight, so they will be computed, but not part of the error function. In order for the values to be saved in the lens file, they are written into system note 6 by the **oprays_template** command, and recovered from system note 6 by the **opcb_rays** command. If you build another error/callback function, you can use this same technique to handle parameters.

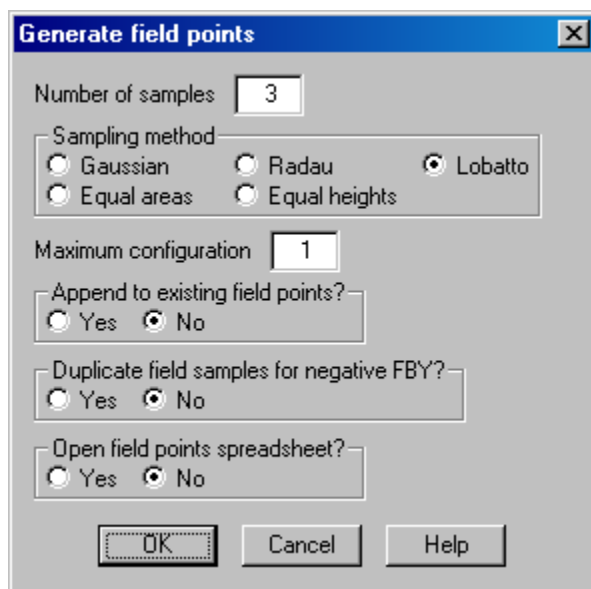
The **cemdoublet_erf** command is an example of an error function that uses the **oprays_template** command programmatically. The business part of this command consists of just 2 lines:

```
oprays_template (1,1,-1,1,1);
```

```
for (ic = 32; ic > 0; ic--) if (ic != 16 && ic != 19) o(ic,del);
```

The first line sets up the entire template, and the second line deletes all the generated operands except OCM16 and OCM19, which evaluate the desired terms. This general technique can be extended to a wide range of design problems. Note that the evaluator (**opcb_rays**) will always compute all the operands for two rays, even though only 2 operands for one ray are all that are used. You could replace the evaluation callback by a more efficient function, but this particular design problem is so simple that it isn't worth the effort. For more complicated design problems where it might make a difference, you should use the internal operands compiler supplied with OSLO Standard or OSLO Premium to achieve greater efficiency.

Std Prm Field Point Set



This command generates an optimization field points set using Gaussian integration.

Number of samples is the number of field points to be generated.

Sampling method determines the pattern of the samples and their weights: Gaussian, Radau, or Lobatto.

The field points are generated between relative field heights $h = 0$ and $h = 1$.

Error Function Tables

Field Point Set...
Ray Set...
Spot Diagram Set...

Std Prm Field Point Set

The field points set defines the fractional coordinates of points on the object surface from which the rays that will be traced as part of the optimization process emanate.

Selecting this menu item will open the field points set spreadsheet editor. An index number of each the field point is displayed on a row button in the first column (FPT) of the spreadsheet. You can select a row for editing by clicking on the row button for that field point. See Chapter 3, "The Surface Data Spreadsheet" for a complete discussion of editing functions in spreadsheets.

FPT	CF	GP	FBY	FBX	FBZ	FYRF	FXRF	FY1	FY2	FX1	FX2	WGT	SP
1	0	0	0.00	0.00	0.00	0.00	0.00	-1.00	1.00	-1.00	1.00	0.250	
2	0	0	0.71	0.00	0.00	0.00	0.00	-1.00	1.00	-1.00	1.00	1.000	
3	0	0	1.00	0.00	0.00	0.00	0.00	-1.00	1.00	-1.00	1.00	0.250	

There are twelve pieces of data that describe a field point. All of them are edited by selecting the desired cell and entering the new value.

The configuration number (CF) is used to denote the configuration(s) in which the field point is to be used. A configuration number of 0 indicates that the field point is global, i.e., to be used in all configurations. A non-zero configuration number means that the field point is only used in the specified configuration. Examples of the use of the field point configuration number are the MTF/RMS wavefront tolerancing and the report graphics display. Also, if the "Current" field sampling option is chosen during error function generation, the configuration number of the field point will be used when constructing the operands. Do not confuse the configuration number of the field point with the configuration number specified for an operand; the configuration number specified in an operand definition is the only configuration specification used in computing the operand value.

The group number (GP) is presently used only as a classification scheme for the field points. For example, if a multiconfiguration error function is generated, all of the field points from the same fractional object coordinate will have the same group number.

The next five data items (FBY, FBX, FBZ, YRF, and XRF) define the reference ray for the field point. FBY, FBX, and FBZ are the fractional coordinates (relative to the object height) of the object point in the y, x, and z directions, respectively. YRF and XRF are the fractional y and x intercepts (relative to the aperture radius of the reference surface), respectively, of the reference ray on the reference surface. If

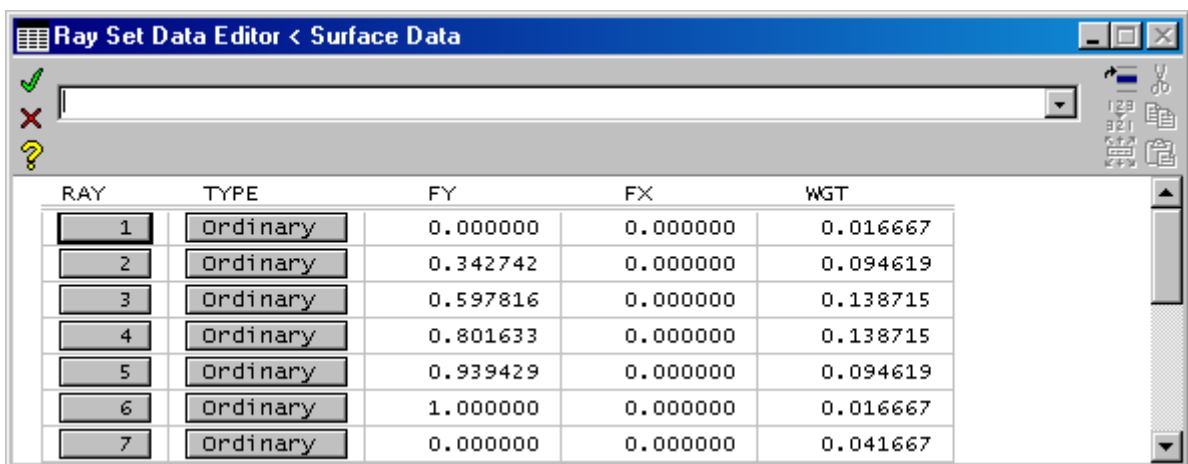
you open the field point set spreadsheet with no currently defined field points, a single field point will be shown, with an FBY value of 0.0 and all other cells grayed out. Entering any value (including 0) in the FBY cell defines the field point and enables the editing of the other cells for that field point.

The second group of five data items (FY1, FY2, FX1, FX2, and WGT) defines the pupil boundary and weight for the field point. FY1 and FY2 are the minimum and maximum, respectively, y-axis pupil bounds, while FX1 and FX2 are the minimum and maximum, respectively, x-axis pupil bounds. These four quantities are fractional coordinates, relative to the pupil radius. Together, these four numbers define an effective elliptical pupil for the field point. Thus, the default values of FY1 = -1, FY2 = +1, FX1 = -1, FX2 = +1 define an unvignetted circular pupil. WGT is the weight given to the field point when constructing the error function.

There is a CCL command, **fpts_copy**, that will copy the specifications of the current field points into the lens drawing operating conditions for the default drawing rays.

Std Prm Ray Set

The ray set defines the fractional pupil coordinates of the rays that will be traced as part of the optimization process. To edit the ray set, select Ray Set from the Optimize menu. The ray set spreadsheet editor will be opened.



RAY	TYPE	FY	FX	WGT
1	Ordinary	0.000000	0.000000	0.016667
2	Ordinary	0.342742	0.000000	0.094619
3	Ordinary	0.597816	0.000000	0.138715
4	Ordinary	0.801633	0.000000	0.138715
5	Ordinary	0.939429	0.000000	0.094619
6	Ordinary	1.000000	0.000000	0.016667
7	Ordinary	0.000000	0.000000	0.041667

The number of the ray is displayed in the first column of the spreadsheet, on a row button. You can select a row for editing by clicking on the row button for that ray. See Edit (Chapter 3) for a complete discussion of editing functions in spreadsheets.

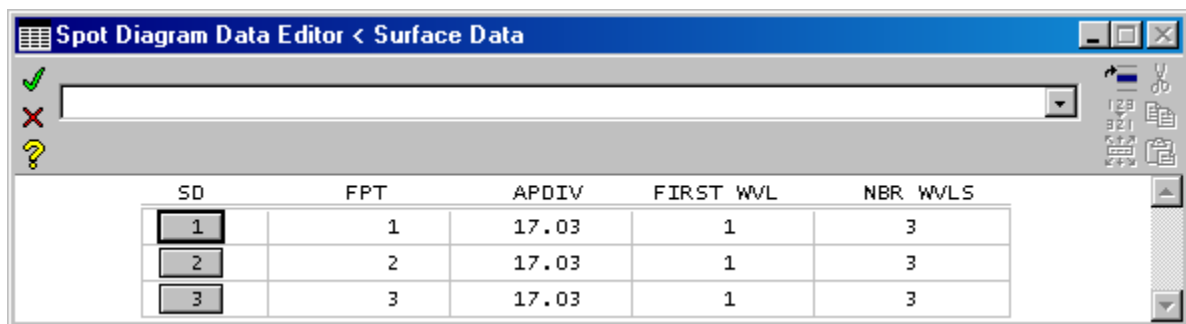
The ray can be one of two types. A *reference ray* is a ray that emanates from a specified field point and passes through a specified point on the reference surface. An *ordinary ray* is a ray that emanates from a specified field point and travels in a prescribed direction. The direction is indicated by the difference between the fractional pupil coordinates of the ray and the fractional coordinates of the ray on the object surface. The operand components that are available for use from a ray depend on the type of ray. (See the discussion of operands below.) To select the type for a ray, click the type cell in the second column of the spreadsheet. Activating the cell in this way just toggles back and forth between the two options.

The third and fourth columns, labeled FY and FX, are the fractional coordinates of the ray. For reference rays, these are the fractional coordinates of the ray on the reference surface, relative to the aperture radius of the reference surface. For ordinary rays, these are the fractional pupil coordinates. FY and FX may be changed by selecting the cell and entering the new value. If you open the ray set spreadsheet with no currently defined rays, a single ray will be shown, with an FY value of 0.0, and the other cells grayed out. This can be considered as a “template” ray entry and will be deleted as an invalid ray if it is not edited. You must enter a value in the FY cell to define the ray.

The fifth column in the spreadsheet, WGT, is the weight assigned to the ray when constructing the error function (Ray weights are not currently used in internal error functions). The weight may be changed by entering the desired weight in the cell.

Prm Spot diagram set

The spot diagram set defines the properties of the spot diagrams that are traced for spot diagram operands (i.e., MTX, MTY, and WVF). To edit the spot diagram set for the current lens, execute the **sds_spreadsheet (sde)** command. The spot diagram spreadsheet editor will be opened.



SD	FPT	APDIV	FIRST WVL	NBR WVLS
1	1	17.03	1	3
2	2	17.03	1	3
3	3	17.03	1	3

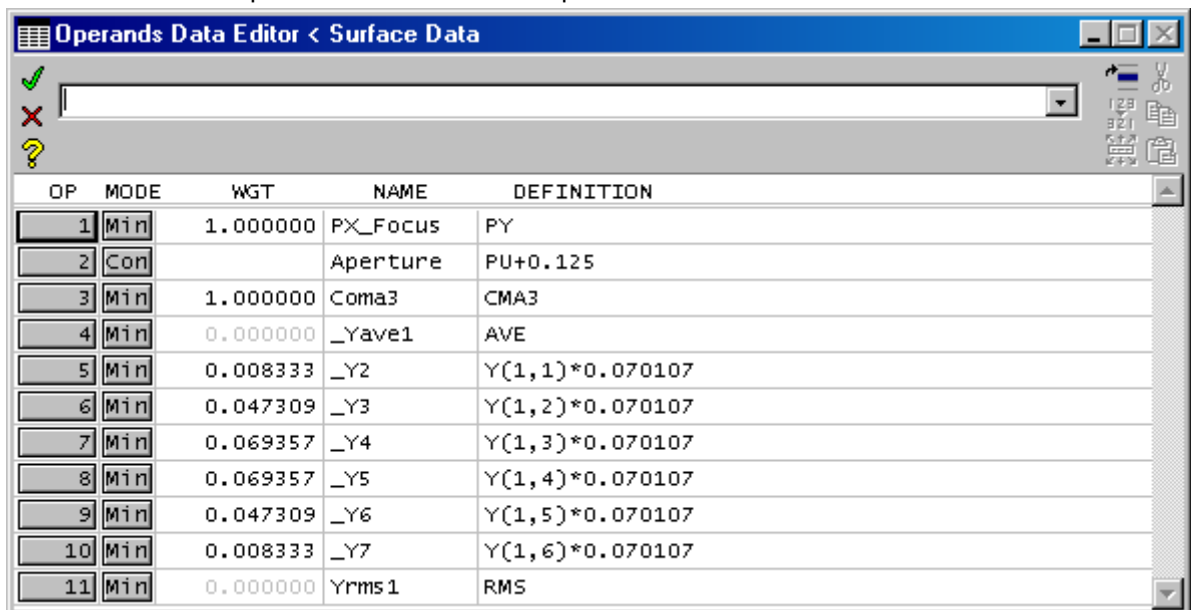
The number of the spot diagram is displayed in the first column of the spreadsheet, on a row button. You can select a row for editing by clicking on the row button for that spot diagram. See Edit (Chapter 3) for a complete discussion of editing functions in spreadsheets.

Each entry in the spot diagram set contains four fields. The first, labeled FPT, is the field point number, which is the index of an entry into the field points set (see above). This number must be between 1 and the number of entries in the field points set. (The default value of 0 indicates that the spot diagram is undefined, and any such spot diagrams remaining when the spreadsheet is closed are deleted.) The second field on each row, labeled APDIV, is the number of aperture divisions across the pupil (grid size) for the spot diagram. A larger number here generally provides more accurate results but causes the spot diagram to take longer to calculate. The third and fourth fields, labeled FIRST WVL and NBR WVLS determine the wavelengths used in computing the spot diagram. If the number of wavelengths (NBR WVLS) is 1, the spot diagram is monochromatic and is computed at the wavelength number given by FIRST WVL. If the number of wavelengths is greater than 1, the spot diagram is polychromatic and is traced at wavelengths FIRST WVL, FIRST WVL + 1, ... , FIRST WVL + NBR WVLS – 1.

Operands

The operands set is a user-constructed measure of the performance of a lens. The operands set consists of a number of operand definitions that measure various physical and optical properties of a lens system; these quantities are combined into a single figure of merit called the *error function* (or *merit function*). It is the task of the designer and the optimization algorithm to determine the values of the variables that minimize the error function.

To edit the operands, select Operands from the Optimize menu. The operands spreadsheet editor will be opened.



The screenshot shows the 'Operands Data Editor < Surface Data' window. It contains a spreadsheet with 5 columns: OP, MODE, WGT, NAME, and DEFINITION. There are 11 rows of operand data. Each row has a row button (a small square with a number) in the first column. The window has a standard Windows-style title bar and a toolbar with icons for saving, undo, redo, and help.

OP	MODE	WGT	NAME	DEFINITION
1	Min	1.000000	PX_Focus	PY
2	Con		Aperture	PU+0.125
3	Min	1.000000	Coma3	CMA3
4	Min	0.000000	_Yave1	AVE
5	Min	0.008333	_Y2	Y(1,1)*0.070107
6	Min	0.047309	_Y3	Y(1,2)*0.070107
7	Min	0.069357	_Y4	Y(1,3)*0.070107
8	Min	0.069357	_Y5	Y(1,4)*0.070107
9	Min	0.047309	_Y6	Y(1,5)*0.070107
10	Min	0.008333	_Y7	Y(1,6)*0.070107
11	Min	0.000000	Yrms1	RMS

The number of the operand is displayed in the first column of the spreadsheet, on a row button. You can select a row for editing by clicking on the row button for that operand. See Chapter 3 for a discussion on editing spreadsheets.

The second column in the spreadsheet contains the definition of the operand. Operand definitions are of the form A , $A+B$, $A-B$, $A*B$, A/B , $A**B$, $A>B$, or $A<B$, where A and B represent *components*. The operators $+$, $-$, $*$, $/$, and $**$ represent addition, subtraction, multiplication, division, and exponentiation, respectively. For example, the operand definition “SA3–SA5” represents the difference between the operand components “SA3” (which is third-order spherical aberration) and “SA5” (which is fifth-order spherical aberration). The operators $<$ and $>$ are used to form *one-sided* operands that have zero value when the expressed relation is true, and have a value equal to the difference between the components when the relation is false. For example, the operand definition “–0.005<SA3” will be inactive if SA3 is greater (algebraically) than –0.005, but it will be active with a value of $-0.005-SA3$ if SA3 is less than –0.005. Conversely, the operand definition “0.005>SA3” will be inactive if SA3 is less than 0.005, but it will be active with a value of $0.005-SA3$ if SA3 is greater than 0.005. The result of using both of these operands in the error function simultaneously is to create an error function that ignores SA3 if it is between –0.005 and 0.005.

The first component of an operand may be preceded by a minus sign, but the second component may not be preceded by a minus sign. For example, the

operand definition " $-0.005 < SA3$ " is legal, but the mathematically equivalent definition " $SA3 > -0.005$ " is not legal.

The names and definitions of valid operand components are given in the following tables. In the tables, *srf* means surface number, *cfg* means configuration number, *srfa* means the first surface in a range, *srfb* means the last surface in a range, *wvn* means wavelength number, *fpt* means field point number, *ray* means ray number, *spd* means spot diagram number, and *freq* means spatial frequency (in cycles/mm for focal systems and cycles/radian for afocal systems). If *wvn* is 0, it refers to color 1. If *cfg* is 0, it refers to configuration 1. If a number is 0, it need not be included in the definition. Unlike the case with variables, there is no such thing as a "global" operand; all operands are defined in one specific configuration. Operand definitions are edited by selecting the definition cell and entering the desired operand definition.

Components defined only in OSLO Premium are shown in shaded rows.

Component type	System operand component
efl(srfa, srfb, cfg)	(paraxial) focal length of the system between surfaces "srfa" and "srfb"
pwr(srfa, srfb, cfg)	(paraxial) power between surfaces "srfa" and "srfb"
tmag(srfa, srfb, cfg)	(paraxial) transverse magnification between surfaces "srfa" and "srfb"
amag(srfa, srfb, cfg)	(paraxial) angular magnification between surfaces "srfa" and "srfb"
ln(srfa, srfb, cfg)	axial length from surface "srfa" to surface "srfb"
eth(srfa, srfb, cfg)	edge thickness from surface "srfa" to surface "srfb" at height of aperture radius
cv(srf, cfg)	curvature
cvd(srf, cfg)	curvature pickup constant
cc(srf, cfg)	conic constant
th(srf, cfg)	thickness
thd(srf, cfg)	thickness pickup constant
ap(srf, cfg)	aperture radius
rn(srf, cfg)	refractive index
dn(srf, cfg)	dispersion factor
cvx(srf, cfg)	toric curvature
ccx(srf, cfg)	xz azimuth conic constant (x-toric; biconic)

fcv(srf, cfg)	Fresnel surface base curvature
fcc(srf, cfg)	Fresnel surface base conic constant
ad(srf, cfg)	4 th order aspheric coefficient
ae(srf, cfg)	6 th order aspheric coefficient
af(srf, cfg)	8 th order aspheric coefficient
ag(srf, cfg)	10 th order aspheric coefficient
dcx(srf, cfg)	<i>x</i> -decentration
dcy(srf, cfg)	<i>y</i> -decentration
dcz(srf, cfg)	<i>z</i> -decentration
tla(srf, cfg)	tilt around <i>x</i> -axis
tlb(srf, cfg)	tilt around <i>y</i> -axis
tlc(srf, cfg)	tilt around <i>z</i> -axis
gsp(srf, cfg)	grating spacing
pfp(srf, cfg)	perfect lens power (1/focal length)
pfm(srf, cfg)	perfect lens magnification
asi(srf, cfg)	aspheric surface coefficient <i>i</i>
asai(srf, cfg)	ISO aspheric surface coefficient <i>i</i>
asbi(srf, cfg)	ISO aspheric surface coefficient <i>i</i>
asci(srf, cfg)	ISO aspheric surface coefficient <i>i</i>
asdi(srf, cfg)	ISO aspheric surface coefficient <i>i</i>
cns(srf, cfg)	ISO cone surface coefficient
cnx(srf, cfg)	ISO asymmetric cone <i>x</i> -coefficient
cny(srf, cfg)	ISO asymmetric cone <i>y</i> -coefficient
ssi(srf, cfg)	spline surface slope at radial zone <i>i</i>
dfi(srf, cfg)	diffractive surface coefficient <i>i</i>
dvw(srf, cfg)	diffractive surface design wavelength
zri(srf, cfg)	Zernike phase surface coefficient <i>i</i>
hx1(srf, cfg)	hologram source point 1 <i>x</i> -coordinate
hy1(srf, cfg)	hologram source point 1 <i>y</i> -coordinate
hz1(srf, cfg)	hologram source point 1 <i>z</i> -coordinate
hx2(srf, cfg)	hologram source point 2 <i>x</i> -coordinate
hy2(srf, cfg)	hologram source point 2 <i>y</i> -coordinate
hz2(srf, cfg)	hologram source point 2 <i>z</i> -coordinate
hwv(srf, cfg)	hologram construction wavelength
nr1(srf, cfg)	GRIN medium r^2 or r'^2 coefficient

nr2(srf, cfg)	GRIN medium r^4 or r'^4 coefficient
nr3(srf, cfg)	GRIN medium r^6 or r'^6 coefficient
nr4(srf, cfg)	GRIN medium r^8 or r'^8 coefficient
nz1(srf, cfg)	GRIN medium z coefficient
nz2(srf, cfg)	GRIN medium z^2 coefficient
nz3(srf, cfg)	GRIN medium z^3 coefficient
nz4(srf, cfg)	GRIN medium z^4 coefficient
vr1(srf, cfg)	GRIN medium r^2 or r'^2 ν -number
vr2(srf, cfg)	GRIN medium r^4 or r'^4 ν -number
vr3(srf, cfg)	GRIN medium r^6 or r'^6 ν -number
vr4(srf, cfg)	GRIN medium r^8 or r'^8 ν -number
vz1(srf, cfg)	GRIN medium z ν -number
vz2(srf, cfg)	GRIN medium z^2 ν -number
vz3(srf, cfg)	GRIN medium z^3 ν -number
vz4(srf, cfg)	GRIN medium z^4 ν -number
pr1(srf, cfg)	GRIN medium r^2 or r'^2 partial dispersion
pr2(srf, cfg)	GRIN medium r^4 or r'^4 partial dispersion
pr3(srf, cfg)	GRIN medium r^6 or r'^6 partial dispersion
pr4(srf, cfg)	GRIN medium r^8 or r'^8 partial dispersion
pz1(srf, cfg)	GRIN medium z partial dispersion
pz2(srf, cfg)	GRIN medium z^2 partial dispersion
pz3(srf, cfg)	GRIN medium z^3 partial dispersion
pz4(srf, cfg)	GRIN medium z^4 partial dispersion
goz(srf, cfg)	GRADIUM axial offset into blank
gmz(srf, cfg)	GRADIUM blank thickness
nrx(srf, cfg)	elliptical GRIN medium x-coefficient
nry(srf, cfg)	elliptical GRIN medium y-coefficient
sgc(srf, cfg)	spherical GRIN center of symmetry
sva(srf, cfg)	sinusoidal GRIN amplitude
svp(srf, cfg)	sinusoidal GRIN period
svf(srf, cfg)	sinusoidal GRIN phase
tas(srf, cfg)	tapered GRIN slope

tao(srf, cfg)	tapered GRIN offset
ugri(srf, cfg)	user GRIN coefficient i
uti(srf, cfg)	user ray trace surface coefficient i
ei(srf, cfg)	eikonal surface coefficient i
jaa(srf, cfg)	amplitude of polarization element J_A
jpa(srf, cfg)	phase of polarization element J_A
jab(srf, cfg)	amplitude of polarization element J_B
jpb(srf, cfg)	phase of polarization element J_B
jac(srf, cfg)	amplitude of polarization element J_C
jpc(srf, cfg)	phase of polarization element J_C
jad(srf, cfg)	amplitude of polarization element J_D
jpd(srf, cfg)	phase of polarization element J_D
sag(srf, cfg)	sag of surface “srf” at height of aperture radius
eth(srfa, srfb, cfg)	edge thickness from surface “srfa” to surface “srfb” at height of aperture radius
pwr(srfa, srfb, cfg)	power between surfaces “srfa” and “srfb”
ln(srfa, srfb, cfg)	axial length from surface “srfa” to surface “srfb”

Component type	Paraxial operand component
py(wvn, srf, cfg)	axial ray height
pu(wvn, srf, cfg)	axial ray slope (after refraction/reflection)
pi(wvn, srf, cfg)	axial ray angle of incidence
pyc(wvn, srf, cfg)	chief ray height
puc(wvn, srf, cfg)	chief ray slope (after refraction/reflection)
pic(wvn, srf, cfg)	chief ray angle of incidence

Component type	Aberration operand component
pac(srf, cfg)	paraxial primary axial chromatic aberration
plc(srf, cfg)	paraxial primary lateral chromatic aberration

sac(srf, cfg)	paraxial secondary axial chromatic aberration
slc(srf, cfg)	paraxial secondary lateral chromatic aberration
sa3(wvn, srf, cfg)	Seidel (third-order) spherical aberration
cma3(wvn, srf, cfg)	Seidel (third-order) coma
ast3(wvn, srf, cfg)	Seidel (third-order) astigmatism
ptz3(wvn, srf, cfg)	Seidel (third-order) Petzval blur
dis3(wvn, srf, cfg)	Seidel (third-order) distortion
psa3(wvn, srf, cfg)	Seidel (third-order) spherical pupil aberration
pcm3(wvn, srf, cfg)	Seidel (third-order) coma of the pupil
pas3(wvn, srf, cfg)	Seidel (third-order) astigmatism of the pupil
pds3(wvn, srf, cfg)	Seidel (third-order) distortion of the pupil
sa5(wvn, srf, cfg)	fifth-order spherical aberration
cma5(wvn, srf, cfg)	fifth-order (linear) coma
ast5(wvn, srf, cfg)	fifth-order astigmatism
ptz5(wvn, srf, cfg)	fifth-order Petzval blur
dis5(wvn, srf, cfg)	fifth-order distortion
sa7(wvn, srf, cfg)	seventh-order spherical aberration
m1(wvn, srf, cfg)	Buchdahl μ_1 coefficient (fifth-order)
m2(wvn, srf, cfg)	Buchdahl μ_2 coefficient (fifth-order)
m3(wvn, srf, cfg)	Buchdahl μ_3 coefficient (fifth-order)
m4(wvn, srf, cfg)	Buchdahl μ_4 coefficient (fifth-order)
m5(wvn, srf, cfg)	Buchdahl μ_5 coefficient (fifth-order)
m6(wvn, srf, cfg)	Buchdahl μ_6 coefficient (fifth-order)
m7(wvn, srf, cfg)	Buchdahl μ_7 coefficient (fifth-order)
m8(wvn, srf, cfg)	Buchdahl μ_8 coefficient (fifth-order)
m9(wvn, srf, cfg)	Buchdahl μ_9 coefficient (fifth-order)
m10(wvn, srf, cfg)	Buchdahl μ_{10} coefficient (fifth-order)
m11(wvn, srf, cfg)	Buchdahl μ_{11} coefficient (fifth-order)
m12(wvn, srf, cfg)	Buchdahl μ_{12} coefficient (fifth-order)

Component type	General ray operand component
x(fpt, ray, wvn, srf, cfg)	x coordinate of ray
y(fpt, ray, wvn, srf, cfg)	y coordinate of ray
z(fpt, ray, wvn, srf, cfg)	z coordinate of ray
rvk(fpt, ray, wvn, srf, cfg)	K direction cosine of ray
rvl(fpt, ray, wvn, srf, cfg)	L direction cosine of ray
rvm(fpt, ray, wvn, srf, cfg)	M direction cosine of ray
nvk(fpt, ray, wvn, srf, cfg)	K direction cosine of surface normal
nvl(fpt, ray, wvn, srf, cfg)	L direction cosine of surface normal
nvm(fpt, ray, wvn, srf, cfg)	M direction cosine of surface normal
xa(fpt, ray, wvn, srf, cfg)	x direction tangent of ray (K/M)
ya(fpt, ray, wvn, srf, cfg)	y direction tangent of ray (L/M)
pl(fpt, ray, wvn, srf, cfg)	path length from previous surface
opl(fpt, ray, wvn, srf, cfg)	optical path length from entrance pupil
xg(ftp, ray, wvn, srf, cfg)	x coordinate of ray in global coordinates
yg(ftp, ray, wvn, srf, cfg)	y coordinate of ray in global coordinates
zg(ftp, ray, wvn, srf, cfg)	z coordinate of ray in global coordinates
rvkg(ftp, ray, wvn, srf, cfg)	K direction cosine of ray in global coordinates
rvlg(ftp, ray, wvn, srf, cfg)	L direction cosine of ray in global coordinates
rvmg(ftp, ray, wvn, srf, cfg)	M direction cosine of ray in global coordinates
xag(ftp, ray, wvn, srf, cfg)	x direction tangent of ray in global coordinates (K/M)
yag(ftp, ray, wvn, srf, cfg)	y direction tangent of ray in global coordinates (L/M)
nvkg(ftp, ray, wvn, srf, cfg)	K direction cosine of surface normal in global coordinates
nvlg(ftp, ray, wvn, srf, cfg)	L direction cosine of surface normal in global coordinates
nvmg(ftp, ray, wvn, srf, cfg)	M direction cosine of surface normal in global coordinates

<code>pin(fpt, ray, wvn, srf, cfg)</code>	Intensity
<code>pdp(fpt, ray, wvn, srf, cfg)</code>	Degree of polarization
<code>per(fpt, ray, wvn, srf, cfg)</code>	Ratio of minor axis to major axis of polarization ellipse
<code>pea(fpt, ray, wvn, srf, cfg)</code>	Angle from “polarization y-axis” to major axis of polarization ellipse
<code>phn(fpt, ray, wvn, srf, cfg)</code>	Handedness of polarization ellipse (Right = +1; Left = -1).

Component type	Reference ray operand component
<code>dxx(fpt, ray, wvn, cfg)</code>	derivative of x -image height with respect to x -pupil shift
<code>dxy(fpt, ray, wvn, cfg)</code>	derivative of x -image height with respect to y -pupil shift
<code>dyy(fpt, ray, wvn, cfg)</code>	derivative of y -image height with respect to y -pupil shift
<code>dyx(fpt, ray, wvn, cfg)</code>	derivative of y -image height with respect to x -pupil shift
<code>xfx(fpt, ray, wvn, cfg)</code>	x -field sag (caustic focus shift in x - z plane)
<code>yfx(fpt, ray, wvn, cfg)</code>	y -field sag (caustic focus shift in y - z plane)
<code>fdxx(fpt, ray, wvn, cfg)</code>	derivative of x -image height with respect to x -object shift
<code>fdxy(fpt, ray, wvn, cfg)</code>	derivative of x -image height with respect to y -object shift
<code>fdyy(fpt, ray, wvn, cfg)</code>	derivative of y -image height with respect to y -object shift
<code>fdyx(fpt, ray, wvn, cfg)</code>	derivative of y -image height with respect to x -object shift
<code>fdxax(fpt, ray, wvn, cfg)</code>	derivative of x -direction tangent with respect to x -object shift
<code>fdxay(fpt, ray, wvn, cfg)</code>	derivative of x -direction tangent with respect to y -object shift
<code>fdyay(fpt, ray, wvn, cfg)</code>	derivative of y -direction tangent with respect to y -object shift
<code>fdyax(fpt, ray, wvn, cfg)</code>	derivative of y -direction tangent with respect to x -object shift

<code>dist(fpt, ray, wvn, cfg)</code>	scalar relative distortion of ray (see below)
<code>s2t(fpt, ray, wvn, srf, cfg)</code>	Curvature of y-direction ray intercept curve about the ray, expressed as wave-front aberration.

Component type	Ordinary ray operand component
<code>dx(fpt, ray, wvn, cfg)</code>	<i>x</i> -displacement with respect to reference ray
<code>dy(fpt, ray, wvn, cfg)</code>	<i>y</i> -displacement with respect to reference ray
<code>dxa(fpt, ray, wvn, cfg)</code>	<i>x</i> -angle displacement with respect to reference ray
<code>dya(fpt, ray, wvn, cfg)</code>	<i>y</i> -angle displacement with respect to reference ray
<code>opd(fpt, ray, wvn, cfg)</code>	optical path difference with respect to reference ray
<code>chr(fpt, ray, wvn, cfg)</code>	Conrady $D-d$ in transverse measure
<code>dmd(fpt, ray, wvn, cfg)</code>	Conrady $D-d$ with respect to reference ray
<code>osc(fpt, ray, wvn, cfg)</code>	Offense against the sine condition (use for meridional rays and on-axis field point for rotationally-symmetric systems only)

Component type	Spot diagram operand component
<code>mtx(spd, freq, cfg)</code>	sagittal MTF
<code>mty(spd, freq, cfg)</code>	tangential MTF
<code>wvf(spd, cfg)</code>	RMS wavefront error

Component type	Statistical operand component
<code>ave</code>	mean value of the first components between this operand and the next <i>rms</i> operand
<code>rms</code>	rms value of all operands between the preceding <i>ave</i> operand and this operand

rms(1)	minimum variance RMS for OPD operands, symmetric system (rays traced only for $FX > 0$)
rms(2)	minimum variance RMS for OPD operands, asymmetric system (rays traced in entire pupil)

Component type	Cross-reference operand component
<i>oi</i>	value of operand number <i>i</i>

Component type	CCL and SCP operand component
ocmi(cfg)	CCL/SCP operand <i>i</i>

Component type	External operand component
ecmi(cfg)	external operand <i>i</i>

The third column in the spreadsheet defines the mode of the operand. The mode is selected by activating the mode cell by clicking on it; this toggles the mode back and forth between its two possible values. An operand can have one of two modes: *minimize* or *constraint*. The optimization algorithm attempts to find exact solutions for the constraint operands, through the technique of Lagrange multipliers. The program attempts to find values of the variables such that all of the constraint operands are equal to their target value of zero. Note that there must be at least as many variables as operands for constraint optimization to be possible. Constraint operands are often used to maintain certain conditions of the lens (e.g., *f*-number) exactly. The minimize-mode operands are used to form the error function. The optimization algorithm attempts to minimize the weighted root-mean-square value of the minimize-mode operands. Minimize-mode operands are used to measure conditions (e.g., lens aberrations) that can not be corrected fully.

For minimize-mode operands, the fourth column in the spreadsheet is the weight of the operand. The weight of an operand determines its relative importance in the error function. The default weight for a minimize-mode operand is 1. For constraint operands, the fourth column is not used. The weight is changed by entering the desired value in the cell.

The fifth column contains a name for the operand. An operand name, which may be up to ten characters long, is optional. The operand name is entered by selecting the name cell for the desired operand, and entering the name. If the name begins with an underscore, “_”, the operand is a “hidden” operand; that is, it is not listed by the **operands** command unless the “all” option is specified. Note that **ave** operands and operands between **ave** operands and **rms** operands are also “hidden.”

The last five general ray operand components are polarization operand components. They correspond directly to the output of the Calculate >> Trace Ray (**tra**) command when polarization ray tracing (**pzrt**) is active. Polarization ray trace information is displayed on the Update Polarization operating conditions spreadsheet (Update >> Operating Conditions >> Polarization). The PIN operand is the intensity as used by the polarization ray trace, so it includes the transmittance effects of all prior surfaces, and internal transmittance, losses. If the surface is used as a reflector, then PIN is computed using reflectance coefficients, otherwise transmittance coefficients are used.

For a discussion of PDP, PER, PEA and PHN refer to the Optics Reference.

The distortion (dist) operand is computed according to the following definition. Let X' and Y' denote the x and y coordinates of the reference ray in the image plane and h_x and h_y denote the x and y fractional object coordinates. Then, the first-order image heights are

$$\begin{aligned}\bar{x}' &= \left. \frac{dX'}{dh_x} \right|_{h_x=0} \\ \bar{y}' &= \left. \frac{dY'}{dh_y} \right|_{h_y=0}\end{aligned}\tag{9.3}$$

If the fractional object coordinates of the reference ray are FBX and FBY, then the ideal first-order radial image position is

$$\bar{r}' = \left[(\text{FBX } \bar{x}')^2 + (\text{FBY } \bar{y}')^2 \right]^{1/2}\tag{9.4}$$

The actual radial image position is

$$R' = (X'^2 + Y'^2)^{1/2}\tag{9.5}$$

Thus, the value of the distortion operand for focal systems is

$$\text{DIST}_{focal} = \frac{R' - \bar{r}'}{\bar{r}'}\tag{9.6}$$

For afocal systems, the relevant quantities are the direction tangents $U' = K'/M'$ and $V' = L'/M'$ (where K' , L' , and M' are the direction cosines of the reference ray in image space) and the first-order angles

(9.7)

$$\bar{u}' = \left. \frac{dU'}{dh_x} \right|_{h_x=0}$$

$$\bar{v}' = \left. \frac{dV'}{dh_y} \right|_{h_y=0}$$

The ideal and actual angles are

(9.8)

$$\bar{t}' = \left[(\text{FBX} \bar{u}')^2 + (\text{FBY} \bar{v}')^2 \right]^{1/2}$$

and

(9.9)

$$T' = (U'^2 + V'^2)^{1/2}$$

The distortion operand for an afocal system is then

(9.10)

$$\text{DIST}_{afocal} = \frac{T' - \bar{t}'}{\bar{t}'}$$

The S2T operand is a measure of coma that was available as a target in the GENII program and is available for use as a reference ray operand in OSLO. The S2T operand is used in the default GENII error function.

The definition of the operand is motivated by the appearance of third-order (Seidel) coma in ray intercept curves. Following the discussion of the wavefront aberration polynomial in the Optics Reference, we denote the Seidel sum for third-order coma as S_{II} .

The wavefront aberration for third order coma is then given by

(9.11)

$$W_{coma,3rd} = \frac{1}{2} S_{II} h \rho^3 \cos \theta$$

The ray aberration in the meridional plane corresponding to this wavefront aberration is

(9.12)

$$\varepsilon_{y,coma,3rd} = \frac{1}{n'u'_a} \frac{\partial W_{coma,3rd}}{\partial \rho_y} = \frac{S_{II}}{2n'u'_a} h \rho^2 (2 + \cos 2\theta)$$

Finally, for the FY vs. DY ray-intercept curve, $\theta = 0$, so

(9.13)

$$\varepsilon_y = \frac{3S_{II}}{2n'u'_a} h \rho^2$$

Thus, at any field point, the transverse ray error of a meridional ray varies as ρ^2 .

By computing the second derivative (for example, the curvature) of the FY vs. DY ray intercept curve at the reference ray, we can construct a measure of coma that is exact in the field, but is third-order in the aperture. S2T is computed from the curvature of the FY vs. DY curve, has the same units as S_{II} , and a magnitude such that, in the absence of all other aberrations, the meridional ray intercept curve would have the form

(9.14)

$$\varepsilon_y = \frac{3 S 2 T}{2 n' u'_a} \rho^2$$

S2T is reported in the same units as OPD, that is, in wavelengths if OPDs are reported in wavelengths, in lens units otherwise.

The offense against the sine condition (OSC) is available as an ordinary ray operand component. The syntax for the component is

```
osc(fpt, ray, wvn, cfg)
```

where *fpt* is the field point number, *ray* is the ray number, *wvn* is the wavelength number, and *cfg* is the configuration number. This component is designed to be used only for meridional rays from an on-axis object point and a rotationally symmetric system. Zernike coefficient operands

The coefficients of a Zernike polynomial decomposition of the wavefront are available as spot diagram operand components. To use these components, you must first define the properties of the spot diagram from which the Zernike coefficients are to be computed. The syntax of the Zernike component is

```
znki(spdi, cfg)
```

where *i* is the number of the Zernike polynomial, *spdi* is the spot diagram number (from the spot diagram set), and *cfg* is the configuration number. The polynomial number *i* is an integer from 0 to 36; the definitions and numbering of the Zernike polynomials used by OSLO are given in the table on p. 118 of the Optics Reference text (p. 127 in the electronic version). If the spot diagram is polychromatic (i.e., the number of wavelengths specified in the spot diagram set is greater than one), the computed Zernike coefficient is the wavelength-weighted average of the coefficients calculated in each wavelength.

The Statistical Operand Components

RMS & AVE - These operands compute statistics of groups of operands, and

should always be present in pairs in the operands set. Note that only one component ("ave" or "rms" itself) is allowed for these operands, and that no weight can be entered. Also note that the value of an ave operand is the weighted mean of the values of only the first components of all subsequent operands, up to (but not including) the next rms operand; the value of an rms operand, however, is the weighted root-mean-square of the values of each of the preceding entire operands.

RMS(1) & RMS(2) - The computation of the variance of the wavefront depends upon the position of the center of the reference sphere to which the OPD is referred. In most cases, it is desirable to let the location of the center of the reference sphere be the point on the image surface that minimizes the variance. This means that the "diffraction focus" (point of maximum irradiance) is not, in general, located at the reference ray position on the image surface. Thus, distortion-like aberrations, which only shift the location of the point spread function, not its shape, do not affect the computed RMS wavefront value. By default, OSLO uses the minimum variance point for the Calculate >> Wavefront Analysis statistics. This point is computed assuming that the additional OPD introduced by a change in reference sphere center position can be expressed as a linear function of the position change² (i.e., the magnitude of the change is small). The resulting expression for the variance can then be solved for the change in position that minimizes the variance.

Because of the additional computation required to compute minimum-variance RMS OPD, "special" RMS operands are required if you wish to perform this calculation. (The AVE-RMS operands, by default, just perform a straight variance calculation of the included operands.) For rotationally symmetric lens error functions, where only rays in the $FX > 0$ side of the pupil are traced, the operand **RMS(1)** should be used to indicate that the referenced OPD operands are to be referred to the minimum variance point. If rays in both sides of the pupil are traced, the operand **RMS(2)** should be used. Note that the argument to the RMS operand should *only* be used for an RMS OPD calculation. If you specify RMS wavefront error when using OSLO's error function generator, the appropriate RMS operand to specify the minimum variance point will automatically be used. Of course, you can change the operand to just "RMS" if you wish to use the reference ray intersection point as the center of the reference sphere.

Variables

Variables are the items that are adjusted independently to establish the performance of the system. In order to optimize a system, you must have both operands and variables, and the operands must be computable functions of the

2. See, for example, W. T. Welford, *Aberrations of Optical Systems*, Adam Hilger, 1986, §7.3, pp. 98–99.

variables. To edit the variables for the current lens, select Variables from the Optimize menu. The variables spreadsheet editor will be opened.

Default air-space thickness bounds:				Minimum	0.100000	Maximum	1.0000e+04	
Default glass thickness bounds:				Minimum	0.500000	Maximum	100.000000	
Vary all curvatures				Vary all thicknesses				Vary all air spaces
V #	Surf	Cfg	Type	Minimum	Maximum	Damping	Increment	Value
1	1	0	Curvature (CV)	0.0000	1.000000	1.000000	1.6000e-05	0.047059
2	2	0	Conic constant (CC)	0.0000	1.000000	1.000000	1.6000e-05	-0.006303
3	3	0	Thickness (TH)	0.0000	1.000000	1.000000	1.6000e-05	-0.049383
4	4	0	Aperture radius (AP)	0.0000	1.000000	1.000000	1.6000e-05	0.051813

Thickness variable default bounds

The first two lines of the spreadsheet contain entries for four operating conditions that specify the default minimum and maximum values for air space and glass thickness variables. Note that these operating conditions specify only the default bounds for the thickness variables; for any particular thickness variable, the defaults can be overridden simply by specifying other minimum and/or maximum values for that variable.

Default and adaptive derivative increments

The damped least squares optimization algorithm estimates the first derivatives of the operands with respect to the variables by changing the value of each variable by a small amount and using the resulting finite difference approximation. The default increment is the value assigned to each new variable. This operating condition value can be overridden for any variable by assigning a specific value to that variable.

If this default increment is equal to zero, OSLO will adaptively adjust the derivative increment during optimization for any variable whose derivative increment is also set to zero.

Adaptive variable damping

If adaptive damping is turned on, OSLO will compute appropriate damping factors for each variable during the optimization process. If adaptive damping is being used, the Damping column in the spreadsheet is disabled.

Choose derivative increments by type

If this option is on and adaptive derivative increments are not used, the default derivative increments will be chosen based upon the type of the variable. Otherwise, the increment for each new variable will be set equal to the default derivative increment, as described above.

Vary-all options buttons

Vary all curvatures - Selecting this button makes the curvature of each surface a variable, except for surfaces that currently have curvature solves or pickups and surfaces for which the index of refraction is identical to that of the previous surface (i.e., dummy surfaces).

Vary all thicknesses - Selecting this button makes the thickness of each surface a variable, except for the object surface and surfaces that currently have thickness solves or pickups. The image surface thickness is only made a variable if the next-to-last surface thickness is not a variable or can not be made a variable.

Vary all air spaces - Selecting this button makes the thickness of each air space a variable, with the same exceptions as the “Vary all thicknesses” button. Note that all variables added by selecting these three buttons have the default values for minimum value, maximum value, damping, and derivative increment. Also, the variables are global variables (i.e., in configuration 0).

Variable number

The number of the variable is displayed in the first column of the spreadsheet, on a row button. You can select a row for editing by clicking on the row button. See Edit (Chapter 3) for a complete discussion of editing functions in spreadsheets.

Surface number

The second column in the spreadsheet is the surface number of the variable. You can change the surface number by selecting the cell and entering the desired value, which must, of course, be less than or equal to the surface number of the image surface.

Configuration number

The third column is the configuration number. This is the configuration in which this variable takes on the specified value. A configuration number of zero indicates that the variable takes on the specified value in all configurations, i.e., the variable is a *global* variable. If there is only one configuration of the system being optimized, then all variables are automatically global. If there are multiple configurations being optimized, then a variable that is tagged with a configuration number will be varied only in that configuration; there must also be an entry in the configuration data for each such variable.

Variable type

The variable type is specified in the fourth column. The type may be specified by entering the type in the cell. Or, if you activate the cell by clicking on it, a pop-up menu containing all of the allowed variable types for the specified surface will be displayed. You can then choose the variable type from the menu. The program will check to make sure that the variable type you have entered is valid for that surface. For example, you will not be allowed to enter HX1 as a variable if you have not defined a hologram to exist on the surface. If a new row is inserted in the variables spreadsheet, the variable type is shown as unknown (“?”); any variables of unknown type that remain when the spreadsheet is closed will be deleted. A list of all possible variable types is given in the following table.

Components defined only in OSLO Premium are shown in shaded rows.

Type	Variable description
cv	curvature
cc	conic constant
th	thickness
ap	aperture radius
rn	refractive index (model glass only)
dn	dispersion factor (model glass only)
cvx	toric curvature
ccx	xz azimuth conic constant
fcv	Fresnel surface base curvature
fcc	Fresnel surface base conic constant
ad	4 th order aspheric coefficient
ae	6 th order aspheric coefficient
af	8 th order aspheric coefficient
ag	10 th order aspheric coefficient
dcx	x-decentration
dcy	y-decentration
dcz	z-decentration
tla	tilt around x-axis
tlb	tilt around y-axis
tlc	tilt around z-axis
gsp	grating spacing
pfp	perfect lens power (1/focal length)
pfm	perfect lens magnification
asi	aspheric surface coefficient i
asai	ISO aspheric surface coefficient i
asbi	ISO aspheric surface coefficient i
asci	ISO aspheric surface coefficient i
asdi	ISO aspheric surface coefficient i
cns	ISO cone surface coefficient
cnx	ISO asymmetric cone x coefficient
cny	ISO asymmetric cone y coefficient
ssi	spline surface slope at radial zone i

dfi	diffractive surface coefficient i
dww	diffractive surface design wavelength
zri	Zernike phase surface coefficient i
hx1	hologram source point 1 x -coordinate
hy1	hologram source point 1 y -coordinate
hz1	hologram source point 1 z -coordinate
hx2	hologram source point 2 x -coordinate
hy2	hologram source point 2 y -coordinate
hz2	hologram source point 2 z -coordinate
hwv	hologram construction wavelength
nr1	GRIN medium r^2 or r'^2 coefficient
nr2	GRIN medium r^4 or r'^4 coefficient
nr3	GRIN medium r^6 or r'^6 coefficient
nr4	GRIN medium r^8 or r'^8 coefficient
nz1	GRIN medium z coefficient
nz2	GRIN medium z^2 coefficient
nz3	GRIN medium z^3 coefficient
nz4	GRIN medium z^4 coefficient
vr1	GRIN r^2 or r'^2 ν -number
vr2	GRIN r^4 or r'^4 ν -number
vr3	GRIN r^6 or r'^6 ν -number
vr4	GRIN r^8 or r'^8 ν -number
vz1	GRIN z ν -number
vz2	GRIN z^2 ν -number
vz3	GRIN z^3 ν -number
vz4	GRIN z^4 ν -number
pr1	GRIN r^2 or r'^2 partial dispersion
pr2	GRIN r^4 or r'^4 partial dispersion
pr3	GRIN r^6 or r'^6 partial dispersion
pr4	GRIN r^8 or r'^8 partial dispersion
pz1	GRIN z partial dispersion
pz2	GRIN z^2 partial dispersion

pz3	GRIN z^3 partial dispersion
pz4	GRIN z^4 partial dispersion
goz	GRADIUM axial offset into blank
nrx	elliptical GRIN x-coefficient
nry	elliptical GRIN y-coefficient
sgc	spherical GRIN center of symmetry
sva	sinusoidal GRIN amplitude
svp	sinusoidal GRIN period
svf	sinusoidal GRIN phase
tas	tapered GRIN slope
tao	tapered GRIN offset
ugri	user GRIN coefficient i
uti	user ray trace surface coefficient i
eii	eikonal surface coefficient i
jaa	amplitude of polarization element J_A
jpa	phase of polarization element J_A
jab	amplitude of polarization element J_B
jpb	phase of polarization element J_B
jac	amplitude of polarization element J_C
jpc	phase of polarization element J_C
jad	amplitude of polarization element J_D
jpd	phase of polarization element J_D
cvd	curvature pickup constant
thd	thickness pickup constant
lithi	thickness of coating layer i
lini	coating (non-dispersive) index
leci	coating (non-dispersive) extinction coefficient

Most of the variable types in the above table are equivalent to the lens data items as described in Chapter 4 – Update. The model glass variables (**rn** and **dn**) are described in Chapter 4 – Optical Materials of the Optics Reference manual. The gradient index ν -number and partial dispersion variables are defined as follows. Let $n_{GRIN}(wvi)$ denote any of the gradient index coefficients **nr1** – **nr4** or **nz1** – **nz4** for wavelength i . Then, the ν -number ν_{GRIN} and partial dispersion P_{GRIN} for that coefficient are given by

(9.15)

$$\nu_{GRIN} = \frac{n_{GRIN}(wv1)}{n_{GRIN}(wv2) - n_{GRIN}(wv3)}$$

(9.16)

$$P_{GRIN} = \frac{n_{GRIN}(wv1) - n_{GRIN}(wv3)}{n_{GRIN}(wv2) - n_{GRIN}(wv3)}$$

If more than three wavelengths are defined, the coefficients $n_{GRIN}(\lambda)$ as a function of the wavelength λ are determined using

(9.17)

$$n_{GRIN}(\lambda) = A + \frac{B}{\lambda^2}$$

where

(9.18)

$$A = \frac{n_{GRIN}(wv1)}{\nu_{GRIN}} \left(1 + \nu_{GRIN} - P_{GRIN} - \frac{\lambda_3^2}{\lambda_3^2 - \lambda_2^2} \right)$$

(9.19)

$$B = \frac{n_{GRIN}(wv1)}{\nu_{GRIN}} \frac{(\lambda_2 \lambda_3)^2}{\lambda_3^2 - \lambda_2^2}$$

In Eqs. (9.18) and (9.19), λ_2 is the value of wv2 in μm and λ_3 is the value of wv3 in μm .

OSLO contains variables for the curvature pickup constant (**cvd**) and the thickness pickup constant (**thd**). The curvature pickup constant variable may be used when the curvature of the current surface is specified by a curvature or minus curvature pickup. Similarly, the thickness pickup constant variable may be used when the thickness of the current surface is specified by a thickness, minus thickness, length, or minus length pickup.

For example, the **cvd** variable can be used to vary the “power” and “bending” of a thin lens, rather than vary the individual surface curvatures. In terms of the surface curvatures (c_1 and c_2) and index of refraction (n), the power (ϕ) of a thin lens is given by

(9.20)

$$\phi = (c_1 - c_2)(n - 1)$$

Given a desired power, index, and first curvature, the second curvature is

(9.21)

$$c_2 = c_1 - \frac{\phi}{n - 1}$$

If the second curvature of the thin lens is specified as a curvature pickup, with a pickup constant of $-\phi/(n-1)$, then the power of the lens will always be ϕ , for any value of c_1 . We can bend the lens by varying the curvature (**cv**) of the first surface and vary the power of the lens by varying the pickup constant (**cvd**) of the second surface.

Select variable types by clicking on the Type cell (fourth column) on the Update variables spreadsheet (Optimize >> Variables). The pop-up menu lists variables that are valid for that surface. You can then edit the variable.

OSLO has multilayer coatings and provides several variables to optimize non-dispersive coatings. The dispersion properties of film “catalog” materials may not be varied.

Note that since the film stack data is stored in memory separately from the lens surface data, if you vary a layer in a stack on a surface, you are varying it on *all* surfaces in the lens on which that coating is used. (If the same coating is used on more than one surface in a lens, it is only stored once in memory; this is analogous to catalog glass data.) The current description of the coatings used in the lens can be displayed with the **mult_coat_data** (**mud**) command. If a coating has been changed from its “design” prescription, a message will be printed after the data. You can save an optimized coating in its current form with the **save_optimized_multilayer** (**som**) command.

Minimum value and maximum value

The fifth and sixth columns contain the minimum value and maximum value, respectively, that the variable is allowed to assume during optimization. If the variable exceeds one of these bounds, a penalty is added to the error function. The optimization operating condition “weight of boundary condition violations” (**opbw**) controls the weight given to boundary value violations in the error function. The penalty added to the error function is equal to the **opbw** operating condition value times the square of the difference between the variable’s value and the boundary that has been violated; increasing the value of this operating condition therefore places more emphasis on correcting boundary condition violations. If both bounds are zero, the variable is allowed to assume any value during optimization without penalty.

Damping

The seventh column is the damping for the variable. This is used to restrict the change of a variable during optimization. Increasing the damping tends to restrict its change during optimization. As mentioned above, if adaptive variable damping is being used, this column is disabled, since OSLO will compute the appropriate variable damping value.

Increment

The increment, displayed in the eighth column of the spreadsheet, is the amount added to the current value of the variable during optimization to estimate the derivatives of the operands with respect to that variable. The default derivative increments are controlled by the “default derivative increment” (**opdi**) and “choose derivatives according to variable type” (**opvi**) optimization operating conditions.

Value

The ninth column in the spreadsheet displays the current value of the variable. Note that changing cells in this column changes the lens data.

Slider-Wheel Design

The interactive design option allows you to see how the performance of your lens depends upon selected construction parameters. By coupling two surface data items to graphics “sliders”, you can quickly change your lens and immediately see the impact of the change on the lens performance. To set up the interactive design graphics windows, select Interactive Design from the Optimize menu.

Window setup

The interactive design windows are set up from the interactive design spreadsheet. You can set up the interactive design to vary two items of surface data. The available items are

- curvature
- thickness
- refractive index
- meridional tilt
- meridional decenter

<input type="radio"/> No Internal evaluation			
<input type="radio"/> Draw only <input type="radio"/> Ray-intercept <input type="radio"/> OPD <input type="radio"/> Field sag <input checked="" type="radio"/> Spot diagram <input type="radio"/> Long. SA			
Graphics scale:		<input type="text" value="0.000000"/>	<input checked="" type="radio"/> 1 Field point at FBY <input type="text" value="0.000000"/> <input type="radio"/> All points
Number of sliders:		<input type="text" value="5"/>	<input checked="" type="radio"/> Use drag processing
Surf	Cfg	Item	Value
<input type="text" value="1"/>	<input type="text" value="0"/>	Curvature pickup multiplier (CVM)	1.000000
<input type="text" value="2"/>	<input type="text" value="0"/>	Thickness (TH)	6.000000
<input type="text" value="3"/>	<input type="text" value="0"/>	Thickness (TH)	1.000000
<input type="text" value="0"/>	<input type="text" value="0"/>	?	0.000000
<input type="text" value="0"/>	<input type="text" value="0"/>	?	0.000000
<input type="radio"/> Enable sw_callback CCL function			Level <input type="text" value="0"/>
Setup name:			<input type="text"/> <input type="button" value="Save setup"/>

The tilt and decenter are confined to the local meridional plane, i.e., the local yz plane. The refractive index may only be varied if there is a model glass, so if your lens contains only catalog glasses, the refractive index option will be inactive. Also, fixed curvatures, thicknesses, etc. can not be varied. You select the items to vary by clicking the radio button for the desired item, and entering the surface number in the surface number cell. The current value of the selected item is displayed in the Central value cell. This value is the middle of the range over which the item can be varied. The cell labeled “+/- Range” shows the magnitude of the maximum amount that the item can be varied from the central value. Both the central value and range can be changed by entering the desired value in the appropriate cell. If the range is entered as 0.0, that slider will not be used.

OSLO Premium has a fifth evaluation option: field sags.

Curvature (CV)	Curvature (CV)
Conic constant (CC)	Conic constant (CC)
Thickness (TH)	Thickness (TH)
Aperture radius (AP)	Aperture radius (AP)
X decentration (DCX)	Fourth-order aspheric coef (AD)
Y decentration (DCY)	Sixth-order aspheric coef (AE)
Z decentration (DCZ)	Eighth-order aspheric coef (AF)
Alpha tilt angle (TLA)	Tenth-order aspheric coef (AG)
Beta tilt angle (TLB)	X decentration (DCX)
Gamma tilt angle (TLC)	Y decentration (DCY)
Tilt vertex offset in x (TOX)	Z decentration (DCZ)
Tilt vertex offset in y (TOY)	Alpha tilt angle (TLA)
Tilt vertex offset in z (TOZ)	Beta tilt angle (TLB)
	Gamma tilt angle (TLC)
	Tilt vertex offset in x (TOX)
	Tilt vertex offset in y (TOY)
	Tilt vertex offset in z (TOZ)
	Grating spacing (GSP)

There are four options available for evaluation while using the interactive design windows: ray-intercept curves, OPD curves, spot diagrams, or none. You can select the evaluation type by clicking the radio button next to the option you want. The evaluation can be done just for the one field point (click the “1 object point” radio button and enter the desired fractional object height in the cell labeled “FBY”), or for three field points (click the “Full field” radio button). If the system has no tilted or decentered elements the field points used are on-axis, 0.7 field, and full field; otherwise the 3 field points are on-axis and plus and minus full field.

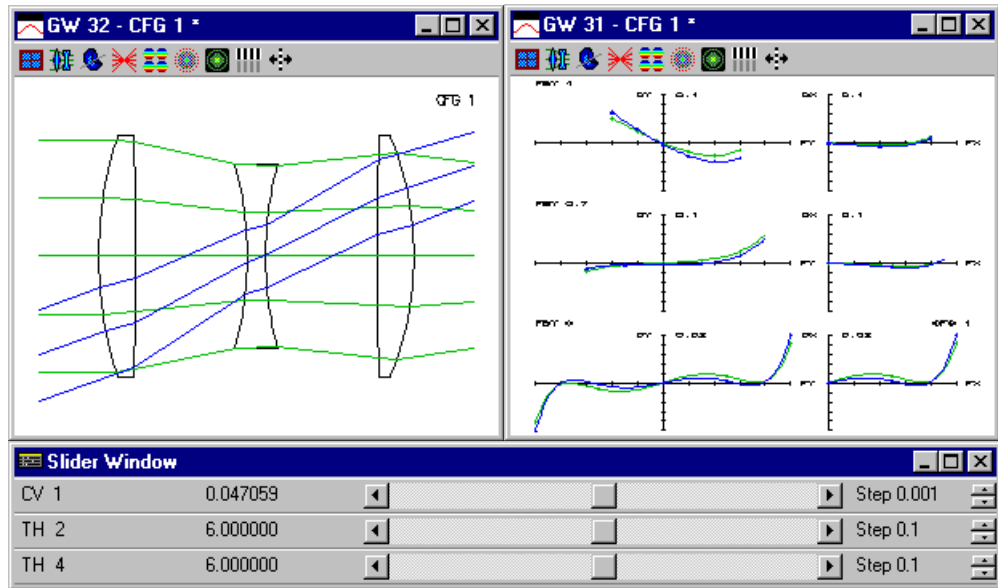
You can select the scale to be used for the display of the evaluation graphics by entering a value in the “Scale for graphics” cell. If you enter a value of 0, OSLO will choose a scale that is appropriate for the initial configuration of the lens. If you want your graphics windows to be updated while you drag the sliders back and forth, click the Yes radio button for the “Use drag processing for sliders” option. If you click the No radio button, the graphics will only be updated when you release the slider.

When you have set up all the options, click OK to close the spreadsheet and open the interactive design graphics windows, or CANCEL to abort the interactive design process. The settings of the controls for interactive design are stored in memory, so the next time you open up the spreadsheet, it will be configured in the same fashion (with the possible exception of the Central value cells, which may have been changed).

Using the interactive design windows

After you have dismissed the spreadsheet, the interactive design windows are opened (if necessary) and set up for your use. In the first interactive design window, a plan view of the lens is drawn, along with the default drawing rays. If you select an evaluation option, it is displayed in a second interactive design window: either ray-intercept curves, OPD curves, spot diagrams, or the field sags.

In the slider control window are one or two “sliders”, corresponding to the items and ranges you selected in the spreadsheet.



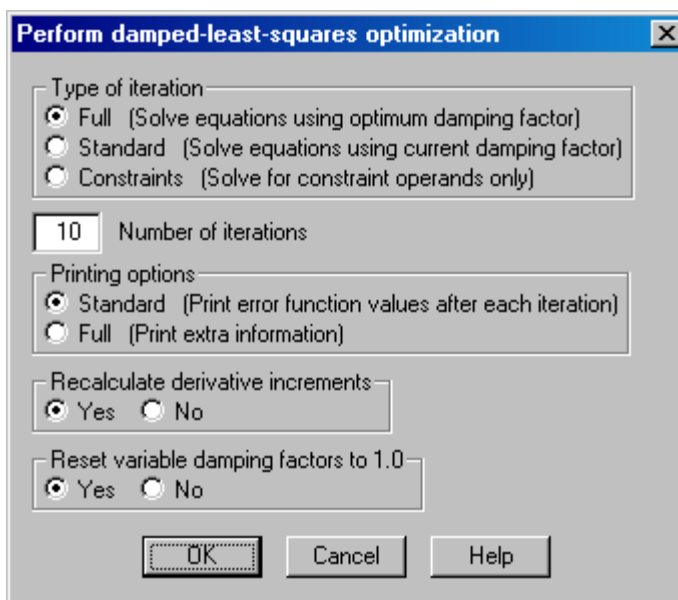
As you adjust the sliders, your lens data is being changed, so you should save a copy of your lens before using the interactive design, if your starting point is valuable to you. At any time you can reopen the interactive design spreadsheet to change the evaluation option, graphics scale, processing option, or varied items.

To use the sliders, hold down mouse button 1 while grabbing the thumb of either slider and move it to the desired value. Alternatively, you can click the arrow at either end of the slider to increment or decrement its value. As you move the slider (if drag processing is on), or when you release the mouse button, the lens picture will be redrawn and the evaluation redone, to reflect the new value of the changed item affected by the slider.

Iterate

After creating a variables set and an error function, the Iterate menu command can be used to optimize the lens. The iterate command has three modes: “standard” mode, “constraint” mode, or “full” mode. Standard-mode optimization consists of computing the derivative matrix for the operands and solving the least-squares normal equations. Constraint-mode optimization consists of optimizing only the constraint operands by the method of Lagrange multipliers. Full-mode optimization consists of repeating a series of standard and constraint-mode iterations with various values of the damping factor in order to determine the optimal value of the damping factor. Also, the Iterate (SHIFT+F9) toolbar button can be used to perform ten full iterations.

The Iterate menu command can be used to select the desired iteration mode and number of iterations:



Iteration is usually performed in full mode. At each full iteration, the derivative matrix is computed and the least-squares normal equations are solved (accounting for any constraint operands) using several values of the damping factor in order to determine its optimal value. After each iteration, the damping factor, root-mean-square value of the minimize-mode operands (if any), root-mean-square value of the constraint operands (if any), and the percentage relative change in the root-mean-square error from the previous iteration are displayed. Under full-mode iteration, the RMS minimize-mode operand error decreases monotonically, but the constraint error may rise and fall. This occurs because a linearized form of the constraints is used to solve for the Lagrange multipliers.

There are three options for printing in the full mode:

Minimal (min) - prints the initial and final error function values in the text output window; intermediate results are printed in the message area to the right of the command line in the main window.

Standard (std) - prints the damping factor, error function value, and percent change after each iteration are printed in the text output window.

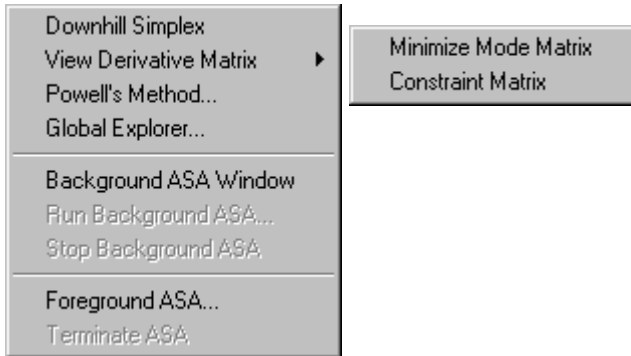
Full (ful) - shows the trial values of the damping factor during each full iteration.

If optimization is performed in constraint mode, only the constraint operands are optimized. It is sometimes useful to optimize the constraint operands separately before optimizing in full mode, since the constraint operands may then tend to remain more “well corrected” during full-mode optimization. Under constraint-mode optimization, the constraint error will decrease monotonically, while the minimize mode error may rise and fall with each iteration, since no account is taken of the minimize-mode operands while correcting the constraint operands.

Iteration is often performed in “standard” mode to perturb the variables when full-mode optimization appears to reach a minimum. Often the damping factor [the “current damping factor” (**opdf**) optimization operating condition] is first set to a small value (e.g., 1.0×10^{-5}). This causes the damped-least-squares algorithm to

change the values of the variables significantly, driving the optimization process far from the current local minimum. Full-mode optimization can then be used to search for other local minima.

Advanced Optimization



The default optimization engine in OSLO is the Damped Least Squares (DLS) algorithm. However, users have the option of choosing a different optimization engine that may enhance your optimization performance. In many cases, the variable settings, optimization error function, and other lens parameters would not need to be altered to take advantage of a different optimization engine.

In the interest of exposing detailed aspects of OSLO's operation to the user, the derivative matrix can be viewed at static points in the optimization process. Viewing the derivative matrix is useful to help investigate confusing or stagnated optimization runs.

Downhill Simplex

The Downhill Simplex optimization method optimizes the lens using the current error function and variables. It is based on the downhill simplex method of Nelder and Mead found in Ref. 3, Section 10.4. This method is provided as an example of CCL programming, not as state-of-the-art optical system optimization technology. However, it may, at times, be a useful or instructional adjunct to damped-least-squares (DLS) optimization. The simplex method only requires that the error function be evaluated; derivative information is not needed. Hence, a typical simplex run requires many more error function evaluations than DLS.

The simplex optimization can be performed when an operand and a variable are defined using Optimize >> Minimization Methods >> Simplex, or the **simplex** CCL command.

Once started, the simplex is initialized using random changes in each of the variables, using the current derivative increment as a scale factor. (No derivatives are calculated; variables are just incremented and the error function recomputed.) Because of this randomness, the algorithm may converge to different solutions from the same starting point, depending on the topology of the error function space. You may want to investigate the effects of the initial scale (the derivative increments) on the path taken by the simplex.

Each time the simplex finds a new "best" system, it is drawn in the current graphics window. When the algorithm terminates, the starting and ending values of the error function are displayed, along with the number of iterations performed. Stopping the optimization before the termination criterion is reached (by pressing the ESCAPE key) restores the system with the lowest error function found up to that point.

View Derivative Matrix

Minimize Mode Matrix / Constraint Matrix

The matrix of partial first derivatives of the operands with respect to the variables may be displayed with the **matrix_dump (mdu)** command. Either the constraint or minimize mode operand matrix may be chosen. This is the matrix used in forming the least squares normal equations. In the display of the matrix, the columns denote variables and the rows denote operands. For display purposes, the matrix is divided into sections, each containing (up to) five variables. For a detailed discussion of the damped least squares algorithm, see Chapter 8 – *Optimization of the Optics Reference* manual.

The capability of printing a sub-matrix of the derivative matrix of partial first derivatives of the operands (rows) with respect to the variables (columns) is available. The matrix can only be displayed when at least one error function operand and one variable are defined. Sections of the derivative matrix can be displayed by using additional parameters of the **matrix_dump** command on the command line.

Prm Powell's Method

Powell's method is an example of a multidimensional direction set function minimization method.³ This type of minimization performs successive line minimizations along "directions" in the variable space. In effect, the problem is reduced to a repeated sequence of one-dimensional minimizations. An interesting feature of Powell's method is that no derivatives are used in the procedure. It is provided in OSLO as a tool for those interested in researching the behavior of various optimization algorithms as applied to the problem of optical design; it is not meant as a replacement for damped-least-squares (DLS) optimization.

Powell's method constructs a set of mutually conjugate directions, starting from a supplied direction set, usually the "basis vectors." In the case of optical design, the basis vectors are changes in each of the variables. Since successive one-dimensional optimizations are done, the initial order in which the basis vectors (for example, the variables) are used may affect the rate of convergence of the algorithm. Five options for the initial ordering of the "directions" are provided. Experimentation is usually the only way to determine which option will result in the most rapid convergence.

Display the Powell optimization using Optimize >> Minimization Methods >> Powell or the **powell (pwl)** command.

Prm Global Explorer

Global Explorer is a CCL command based on the Escape Function method, and is included with OSLO Premium. This User's Guide presents a demo procedure to illustrate the steps in using Global Explorer with OSLO Premium. You should first

3. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes In C*, Second Edition, Cambridge University Press, 1992, Section 10.5.

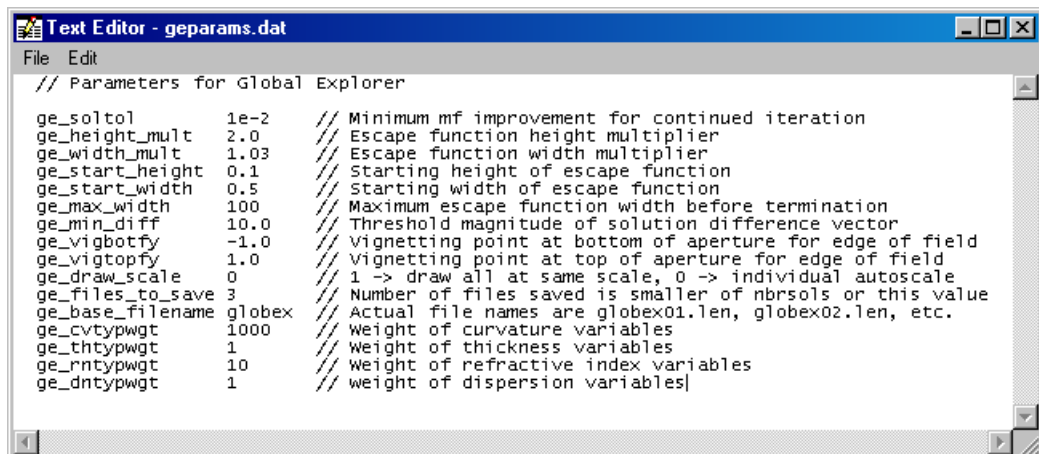
run the procedure as described below, to understand its use. Then you can experiment with different lenses and different values for the parameters.

Before running Global Explorer, you must have a lens in your computer that is ready for optimization, that is, it must have variables and operands that define a suitable error function. The escape function that is added to the error function is generated automatically by Global Explorer, so your error function should not include it. This demo uses a starting design for a wide-angle triplet having the same paraxial specifications as the design example described in the OSLO Optics Reference manual.

Open two graphic windows in addition to the text window, and arrange your display so the windows are similarly sized and located to the figures below.

Click “File>>Open” on the main OSLO menu. Click Public in the dialog box, and open the file demo/Premium/gedemo.len.

First choose the Global Explorer's Set Parameters option by executing it from the “Optimize>>Global Explorer>>Set Parameters” menu, or by typing the command **ge** (If you type the command, it will display an options box). This will display the parameters used by Global Explorer in the OSLO Editor. Don't change these parameters for the demo, just exit from the editor by clicking “File>>Exit” in the text editor (not on the main menu). Later, you can repeat this step to edit different parameters (i.e. change their values) and save the file before exiting.



```

Text Editor - geparams.dat
File Edit
// Parameters for Global Explorer

ge_soltol      1e-2    // Minimum mf improvement for continued iteration
ge_height_mult 2.0     // Escape function height multiplier
ge_width_mult  1.03    // Escape function width multiplier
ge_start_height 0.1    // Starting height of escape function
ge_start_width 0.5     // Starting width of escape function
ge_max_width   100     // Maximum escape function width before termination
ge_min_diff    10.0    // Threshold magnitude of solution difference vector
ge_vigbotfy    -1.0    // Vignetting point at bottom of aperture for edge of field
ge_vigtopfy    1.0     // Vignetting point at top of aperture for edge of field
ge_draw_scale  0       // 1 -> draw all at same scale, 0 -> individual autoscale
ge_files_to_save 3     // Number of files saved is smaller of nbrsols or this value
ge_base_filename globex // Actual file names are globex01.len, globex02.len, etc.
ge_cvtypwgt    1000    // Weight of curvature variables
ge_thtypwgt    1       // Weight of thickness variables
ge_rntypwgt    10      // Weight of refractive index variables
ge_dntypwgt    1       // Weight of dispersion variables
  
```

Most of the parameters for Global Explorer are described in the Beginner's Guide. There are a few additional ones that have been provided in the CCL program. For example, the **ge_vigbotfy** and **ge_vigtopfy** parameters allow you to specify the vignetting that sets the apertures of the solutions. In the demo, the range of fractional aperture vignetting goes from -1.0 to 1.0 . Note that this is not the same as the vignetting defined in the optimization field point set, which has a range from -0.8 to 0.8 . Often it is advantageous to have the optimization vignetting different from the usage vignetting.

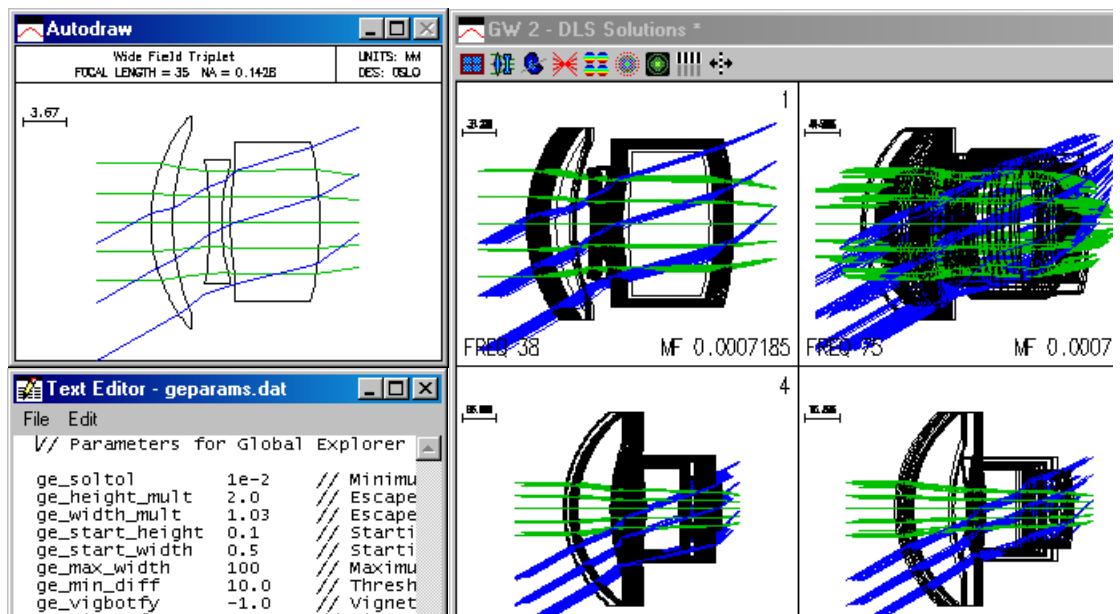
Another of the parameters is called **ge_draw_scale**. If this is set to 1 all the solutions will be drawn to the same scale. If **ge_draw_scale** is set to 0, each solution will be scaled automatically to fit its viewport.

The parameter **ge_soltol** sets the value of the corresponding OSLO operating condition **opst** (optimization solution tolerance). If you have a small display, you may want to use the zoom capability of OSLO graphics windows (provided by the

toolbar buttons) to enlarge a particular viewport of interest so you can see it better.

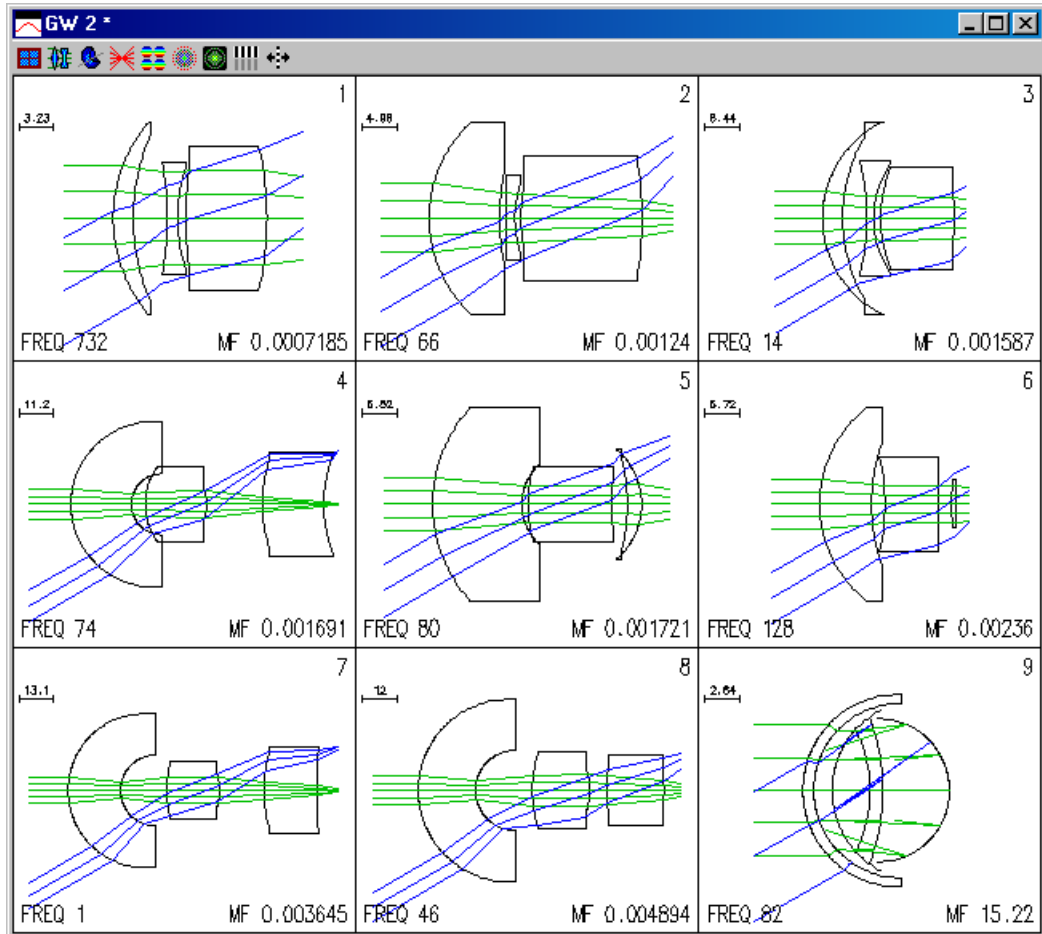
Global Explorer will save as many lens files as are indicated by the **ge_save_files** parameter. Each file will contain the name set by **ge_base_filename**. For the demo, there will be three files, globex01.len, globex02.len, and globex03.len. You can open these and evaluate or modify them as desired. Note that the solutions not saved as files will not be available for evaluation or modification. It is quite usual for most of the Global Explorer solutions to be of only casual interest, and discarding them prevents excessive disk clutter.

Run Global Explorer again, and this time choose the Global Explorer option. The program will display an options box asking for the desired number of solutions. (Note that the values are the squares of integers, which is required by the window formatting that Global Explorer uses.) Double click "9" for this demo. The program will immediately begin to search for solutions. As it progresses, it will display the current starting point in graphics window GW1, and the solutions in graphics window GW2. There will be several solutions that are similar to previous solutions, and the appropriate graphics viewport will be overwritten if the new solution is better (i.e., lower error function) than the old. In each viewport, you will see the solution number in the upper right, the FREQ (i.e., number of times that solution has been obtained) in the lower left, and the minimum error function MF in the lower right. After the search is nearly complete, the window will appear as follows. Note that the Text Window displays the values of the parameters used for the current search.



You will see that it is not possible to tell much about the details of the solutions displayed in each viewport. There are two reasons for this. The first is that the drawing is automatically scaled to fit in the viewport, so different solutions may be drawn at different scales. The second is that each solution drawn in a viewport is compared for similarity only to the solution that currently exists there, not to all previous solutions. This means that after several replacements, the solution may be quite different from the one originally shown in the viewport. However, because solutions are only replaced if the new one is better than the old, the method guarantees that the final one will be the best of a series.

After the search is completed, the solutions will be sorted so the minimum error function appears first, and combined so that all solutions that fall within the minimum difference (as defined by the **ge_min_diff** parameter described in the Beginner's Guide) are merged. The solutions window is then redrawn, with only the best one in each group being displayed. For this reason, the number of final solutions may be smaller than the original number of solutions (the **FREQ** of each final solution is adjusted to indicate the total merged number of overlaps). For the demo, the final display will appear as shown below.



After you have successfully run the demo, you can re-run the program on your own lenses. Global Explorer is a multiple solution generator whose purpose is to provide several solutions that may be interesting for additional design work. You will find that adjusting the parameters used for the program will have major effects on the ability of the algorithm to generate multiple solutions and the quality of the solutions. You should not hesitate to experiment with different values.

One of the interesting aspects of Global Explorer is that it displays chaotic behavior, in contrast to Adaptive Simulated Annealing (ASA), which displays stochastic behavior. That is, if you start Global Explorer with exactly the same input conditions, you will get the same final results in several trials, but if you change the conditions even a tiny amount between trials, you may get results that are grossly different. This is not the case with ASA, which uses random numbers, and hence automatically produces a different execution path for each run.

In setting up a starting point to use with Global Explorer, it is important to use a trusted and stable error function, since the program may run for extended lengths of time without user intervention. It is usually worthwhile to include edge and center thickness controls in the error function, of course in addition to control of paraxial properties.

While you are using Global Explorer, you may wish to abort the optimization if you see that the process is not proceeding according to your expectations. To do this, press the ESCAPE key on your keyboard. It may take a few moments to respond, but the program should terminate gracefully, saving the work previously completed.

For additional information about Global Explorer, please see the "Beginner's Guide to Global Explorer" posted on the web site www.sinopt.com under the Design Note on "Multiple Solution Generators". This document is also supplied as an HTML file in the OSLO help directory. Also, see the paper "Global optimization with escape function" by Masaki Isshiki in the 1998 International Optical Design Conference Proceedings, SPIE Vol. 3482, pp. 104–109 (1998).

Prm ASA: Adaptive Simulated Annealing

Adaptive Simulated Annealing is a Monte Carlo global optimization algorithm that searches for the lens system of lowest error function value over a specified region of interest. Since ASA requires much longer to find a solution than Damped Least Squares (Iterate command), ASA has been designed to run as a background application that is started from OSLO under multitasking operating systems (UNIX and Windows NT), leaving the designer free to use OSLO for other tasks. A version of ASA that is built into OSLO is available under Microsoft Windows; it is not possible to use OSLO for other tasks when this version of ASA is running.

To optimize with ASA, it is necessary first to create a lens file containing lens data, a variables set, and an error function (operands set). This "base" file should then be saved to disk (it is recommended that each base file be saved in a separate directory). Each time ASA is run, it produces one or more solution files by performing global optimization on the base file.

There are a few differences between the procedures for setting up lenses for optimization with ASA and damped least squares:

- Since ASA requires a clearly defined region of interest over which to perform its search, lower and upper bounds (MIN and MAX values) must be specified for all variables. The region of interest should be kept as small as possible.
- Constraint operands are ignored by ASA.
- ASA optimizes most efficiently when a "floating" aperture stop is specified; that is, for the purposes of optimization, surface 1 is specified as the *nominal* aperture stop, and a chief ray height solve (with zero target) is placed on the surface before the *physical* aperture stop. Additionally, the surface containing the physical aperture stop may be specified as the reference surface.

For example, suppose that you wish to set up the Demo Cooke Triplet lens (demotrip.len in the "Public\len\demo\edu" directory) for optimization with ASA. The aperture stop is at surface 4 in the original lens:

*LENS DATA

Demo Triplet

SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPE	NOTE
0	--	1.0000e+20	3.6397e+19	AIR		
1	21.250000	2.000000	6.500000 K	SK16	C	
2	-158.650000	6.000000	6.500000 P	AIR		
3	-20.250000	1.000000	5.000000	F4	C	
4	19.300000	6.000000	5.000000 A	AIR		
5	141.250000	2.000000	6.500000	SK16	C	
6	-17.285000	42.950000	6.500000 K	AIR		
7	--	--	18.170326 S			

To prepare the lens for optimization with ASA, first determine the position of the entrance pupil relative to the first surface of the lens using the Show >> Paraxial Setup command:

*PARAXIAL SETUP OF LENS

APERTURE

Entrance beam radius:	6.250000	Image axial ray slope:	-0.124999
Object num. aperture:	6.2500e-20	F-number:	4.000043
Image num. aperture:	0.124999	Working F-number:	4.000043

FIELD

Field angle:	20.000000	Object height:	-3.6397e+19
Gaussian image height:	18.198709	Chief ray ims height:	18.154007

CONJUGATES

Object distance:	1.0000e+20	Srf 1 to prin. pt. 1:	13.429779
Gaussian image dist.:	43.080554	Srf 6 to prin. pt. 2:	-6.919987
Overall lens length:	17.000000	Total track length:	1.0000e+20
Paraxial magnification:	-5.0001e-19	Srf 6 to image srf:	42.950000

OTHER DATA

Entrance pupil radius:	6.250000	Srf 1 to entrance pup.:	10.466307
Exit pupil radius:	6.643768	Srf 6 to exit pupil:	-10.070166
Lagrange invariant:	-2.274814	Petzval radius:	-149.381547
Effective focal length:	50.000541		

Next, using the Surface Data spreadsheet, add a new surface in front of surface 1 and make the new surface the aperture stop, with an aperture radius equal to the entrance beam radius and with a thickness equal to the negative of the "Srf 1 to entrance pup." value listed under **OTHER DATA** above. The physical aperture stop is to be located at surface 5 (formerly surface 4); make surface 5 the reference surface so that reference rays behave as they did in the original lens. To hold the stop on surface 5, you should target PYC on surface 5 to 0.0 in the error function. The final surface data appears as follows:

LENS DATA*Demo Triplet**

SRF	RADIUS	THICKNESS	APERTURE RADIUS	GLASS	SPE	NOTE
0	--	1.0000e+20	3.6397e+19	AIR		
1	--	-10.466307	6.250000 A	AIR		
2	21.250000	2.000000	6.500000 K	SK16	C	
3	-158.650000	6.000000	6.500000 P	AIR		
4	-20.250000	1.000000	5.000000	F4	C	
5	19.300000	6.000000	5.000000 R	AIR		
6	141.250000	2.000000	6.500000	SK16	C	
7	-17.285000	42.950000	6.500000 K	AIR		
8	--	--	18.170326 S			

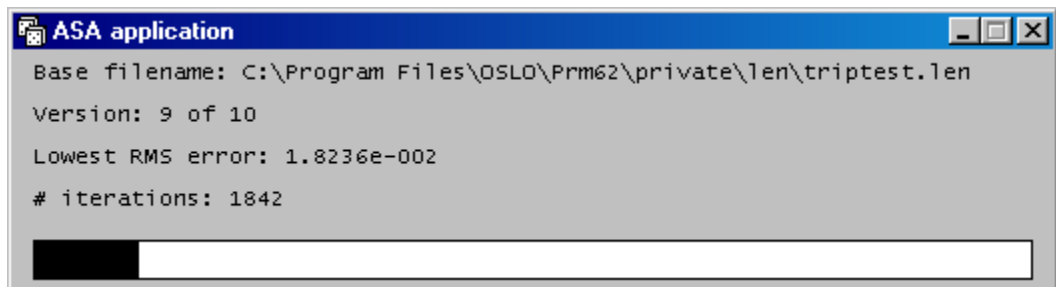
Variables and operands can now be added for optimization, and the lens can be saved to a disk under a new file name. The values of the variables in the base file are unimportant, since ASA does not use them as a starting point for optimization.

ASA creates one or more solution files each time it is run. The names of these files are formed from the name of the “base” file (the lens file that you create, containing the lens data, variables, and error function), plus the solution number. For example, if you create a lens file named abc.len and run ASA with a request to produce twenty solution files, ASA will produce files named abc00.len, abc01.len, abc02.len, and so forth, up to abc19.len. (Users of ASA under Microsoft Windows 3.1 should note that file names may contain only eight characters before the period, so the base file name should be kept as short as possible so that there are enough characters remaining for ASA to append the solution number to the base file name.)

After ASA completes its optimization, the solution files can be opened in OSLO. ASA appends the number of iterations and the error function value to the system notes of each solution file.

Prm Background ASA Window

Under multitasking operating systems (i.e. UNIX and Windows NT), a version of ASA is available that runs as a separate application in the “background.” The Start Background ASA Application command starts the ASA application, which displays a small window:



(See the Run Background ASA command description below for a description of the contents of the window.) The background ASA application can be started only from OSLO and can run only when OSLO is running. Optimization is actually started by using the Run Background ASA menu command.

Prm Run Background ASA

The Run Background ASA command is used to begin optimization with ASA. OSLO prompts for the base file name (specify either the full path name of the base file or, if the base file is in the current working directory, you need give only the file name) and the number of solution files to produce. The background ASA application then begins optimization. The ASA window displays the base file name, the current solution number, the RMS error (error function value) of the best point found up to the current iteration, and the number of iterations performed in the current solution. At the bottom of the window is a “progress indicator” that shows a rough estimate of the progress of optimization on the current solution; at the outset, the box is filled with white, and as optimization progresses, the box gradually turns black.

The ASA window is updated approximately once per second during optimization.

Prm Stop Background ASA

The Stop Background ASA command stops optimization in the background ASA application. The ASA application is still running, however, and optimization can be started with a new base file (or with the same base file) by using the Run Background ASA command.

Prm Foreground ASA

The Run Built-in ASA command invokes a special version of ASA that is built into OSLO; this version of ASA is available only under Windows. The current lens file in OSLO is used as the base file, so OSLO prompts only for the number of solution files to produce. A window is opened to display the progress of ASA optimization. This window is similar to the window that is displayed by the background ASA application (see above):

Note that you cannot use OSLO while the built-in ASA is optimizing; to stop the built-in ASA optimization, select the “Close” menu item from the system menu in the upper left corner of the ASA window.

Prm Terminate ASA

The Terminate ASA Application command halts background optimization with ASA and quits the ASA application. To use ASA again, the Start Background ASA Application and Run Background ASA commands must be used.

Optimization Conditions

The optimization operating conditions control various facets of the optimization process.

Current damping factor:	0.001000
Starting value of damping factor:	1.0000e-08
Scaling factor for damping:	0.615800
Per cent improvement for continuing full iterations:	0.010000
Command for CCL/SCP operands:	oprds
Default derivative increment (0 = adaptive):	1.0000e-07
Weight of boundary condition violations:	1.0000e+03
Use adaptive variable damping:	<input checked="" type="radio"/> Off <input type="radio"/> On
Choose derivative increments according to variable type:	<input type="radio"/> Off <input checked="" type="radio"/> On
Use infinite radius reference sphere during optimization:	<input checked="" type="radio"/> Off <input type="radio"/> On
Error function value:	<input checked="" type="radio"/> Root-mean-square (RMS) <input type="radio"/> Weighted sum of squares
Command for field weighting:	
Command for pupil weighting:	
Command for color weighting:	
ASA Annealing rate:	0.010000
ASA Relative Termination level:	0.010000
Aperture checking in ray operands:	<input checked="" type="radio"/> Off <input type="radio"/> On
Give zero weight to incalculable operands:	<input checked="" type="radio"/> Off <input type="radio"/> On

Current damping factor (opdf)

This is the current value of the damping factor used in the damped-least-squares optimization algorithm.

Starting value of damping factor (opds)

This is the initial value of the damping factor to be used.

Scaling factor for damping (opdm)

This is the change that is made in the damping factor on each pass of the damping factor search done during full iterations. The scaling factor must be less than 1.

Per cent improvement for continuing full iterations (opst)

When the improvement in the error function is less than this value for two consecutive iterations, the iteration process terminates.

Tolerance for one-sided constraint operands (opct)

This is the amount that one-sided constraint operands are allowed to deviate from zero.

Command for CCL/SCP operands (opoc)

Name of the CCL or SCP command that computes CCL or SCP operands.

Default derivative increment (opdi)

This is the default increment value for variables. It is used when calculating derivatives. If this default increment is equal to zero, OSLO will adaptively adjust the derivative increment during optimization for any variable whose derivative increment is also set to zero.

Weight of boundary condition violations (opbw)

This is the weight to be assigned to the contribution of variable boundary condition violations to the error function.

Use adaptive variable damping (opvd)

This option can be set either Off or On by clicking the appropriate radio button. If adaptive damping is turned on, OSLO will compute appropriate damping factors for each variable during the optimization process.

Choose derivative increments according to variable type (opvi)

This option can be set either Off or On by clicking the appropriate radio button. If the option is On, default derivative increments for the variables will be chosen according to the type of variable. If the option is Off, the default derivative increments will all be equal to the default derivative increment (see above).

Command for field weighting (opfw)

Name of the CCL command that computes the weighting function for field points (blank implies uniform weighting).

Command for pupil weighting (oppw)

Name of the CCL command that computes the weighting function for pupil points (blank implies uniform weighting).

Command for color weighting (opcw)

Name of the CCL command that computes the weighting function for wavelengths (blank implies uniform weighting).

ASA annealing rate (opar)

Annealing rate for the adaptive simulated annealing global optimization application.

ASA relative termination level (opat)

Termination level for the adaptive simulated annealing global optimization application.

Aperture checking in ray operands (opac)

Rays will be checked for vignetting by designated checked apertures and special apertures. This operating condition is only used by the adaptive simulated annealing optimization algorithm.

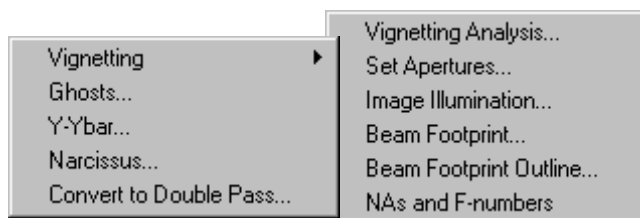
Give zero weight to incalculable operands (opzw)

Removes incalculable operands (e.g. ray failures) from the error function. This operating condition is only used by the adaptive simulated annealing optimization algorithm.

Use infinite radius reference sphere during optimization (opir)

If this option is On, the reference sphere for OPD calculations for optimization rays will be taken to be at infinity, regardless of the setting of the wavefront reference sphere position (**wrsp**) general operating condition. If this option is Off, the reference sphere position is determined by the **wrsp** general operating condition. Use of this option is recommended when the pupil position is not well defined or is varying greatly during optimization, resulting in reference spheres that lie inside the ray caustic.

Support Routines



Vignetting

Vignetting is very important in determining the performance of an optical system. Although the term is often used to describe the reduction of aperture by the edges of lenses that are separated from the aperture stop, OSLO uses the term more generally to refer to either an increase or decrease in aperture from the nominal value determined from the operating conditions. Thus, for example, it is possible to have *on-axis vignetting* when the aperture stop radius is less than the value of the axial ray height on the surface (which is determined by the entrance beam radius). Also, in a wide-angle system, it is common for the pupil to grow as the field angle is increased, which leads to vignetting factors greater than unity.

OSLO uses fractional aperture coordinates as a measure of vignetting. If the extreme upper ray that passes through a surface has a height of 0.8 relative to the ray that passes through the point defined by the edge of the unobscured beam, we say that the beam has a vignetting factor of 0.8.

Vignetting Analysis is performed by the **vig** CCL command, which analyses vignetting for all defined field points in the current field point set. At each field point, **vig** traces several rays at increasing and decreasing apertures until rays

are blocked by a checked aperture (or fail for some other reason). The minimum and maximum aperture are reported. The process is repeated for rays in the sagittal ($y = 0$) plane. In order for the command to work, there must be checked apertures in the system, and aperture checking must be turned on in the general operating conditions spreadsheet.

The **vig** command assumes meridional symmetry. Although it is up to the user, it is usually necessary to have a checked aperture on the aperture stop (this depends on what you are trying to determine).

Std Prm Vignetting Analysis

Aperture checking is not used in optimization, so the effects of vignetting must then accounted for in the initial coordinates of rays that are traced through the system. In addition to determining the vignetting effects of checked apertures and printing the results to the current text window (see previous section), the **vig** command can also:

- Copy the analysis results to the vignetting factors of the current field point set ("Copy vignetting data to field point set" = "Yes"), and
- Copy the analysis results to the pupil extents of the lens drawing conditions (Copy new field point data to drawing fans).

Typical analysis output is shown below.

```
*VIGNETTING FACTORS
FPT   CFG   FBY           FBX           FY1           FY2           FXMAX
  1     0      --           --          -1.024187     1.024187     1.024187
  2     0     0.700000      --          -0.637293     0.810585     0.990811
  3     0     1.000000      --          -0.431105     0.612924     0.901827
Results copied to 'Field Point Set'
'Field Point Set' copied to 'Drawing Operating Conditions'
```

The above output was obtained for the demotrip.len Cooke triplet. The aperture stop was made a checked aperture. A more comprehensive analysis can be accomplished by reporting the complete data of the field point set after the Vignetting Analysis has been performed (Lens>>Show Optimization Data>>Field Point Set/Ray Set).

```
*RAYSET
FPT      FBY/FY1      FBX/FY2      FBZ/FX1      FYRF/FX2      FXRF/WGT      CFG/GRP
F 1      --          --          --          --          --          -
          -1.024187    1.024187    -1.024187    1.024187    1.000000    -
F 2      0.700000      --          --          --          --          -
          -0.637293    0.810585    -0.990811    0.990811    1.000000    -
F 3      1.000000      --          --          --          --          -
          -0.431105    0.612924    -0.901827    0.901827    1.000000    -
NO RAYS DEFINED
```

Note that the on-axis vignetting is greater than 1.0. This is an indication that the aperture radius of the stop surface does not match that established by the paraxial setup.

Set Apertures

The **apset** CCL command sets aperture radii to give specified lower and upper aperture vignetting at full field. The syntax of the command is

apset (*Set_all_apertures, Lower_aperture_vignetting, Upper_aperture_vignetting, Number_of_digits_to_round_to*)

This command assumes that the lens is rotationally symmetric. The default is to set all the apertures in the lens. To set only the apertures that are currently “solved”, select “no” for the *Set_all_apertures* argument.

The vignetting is entered as fractional pupil coordinates, that is, a lower vignetting factor of -1.0 and an upper vignetting factor of +1.0 means there is no vignetting. The apertures that limit the full-field beam will be marked as checked apertures.

Std**Prm**

Image Illumination

The relative image illumination is computed assuming an incoherent, Lambertian object. The illumination is expressed as a percentage of the on-axis illumination.

There are two techniques available for this computation. For symmetric, unobscured systems, an **Elliptical pupil approximation** can be performed, where the size of the exit pupil is determined by tracing three rays. For a more general system, (OSLO Premium only), an **Image space ray grid** may be used: a grid of rays is iteratively traced until the rays are equally incremented in the direction cosine space of the image. This is obviously a much lengthier calculation, but is valid for arbitrary systems.

The illumination may be calculated at the **Current object point** or as a **Scan across object** from **Minimum** to **Maximum fractional object height for scan**. For scans across the object, if **Image space ray grid** is used for the calculations, the scan across the object may be performed along either the Y or the X direction.

The output is the percent Relative Illumination (**REL ILLUM**) at either the Current object point or at the specified sample relative object heights (**FBY**).

Beam Footprint

The **footprt** CCL command plots a pupil footprint at a lens surface. The plot shows how a square grid of rays, evenly spaced in the entrance pupil, intersects the specified surface. The lens aperture is drawn. If the surface has Special Aperture Data applied, the individual aperture components are drawn. Rays that are vignettted before they reach the specified surface are not plotted. Rays that strike the surface but are vignettted by a subsequent lens surface are shown with an red “x”, and rays that pass through the system and reach the image are shown with a green “+”. Note that rays are vignettted only if the surface has been designated as having a checked aperture (or is defined with Special Aperture Data) and aperture checking is currently active. The format of the command is

footprt *fbx fby surface number_of_rays wavelength*

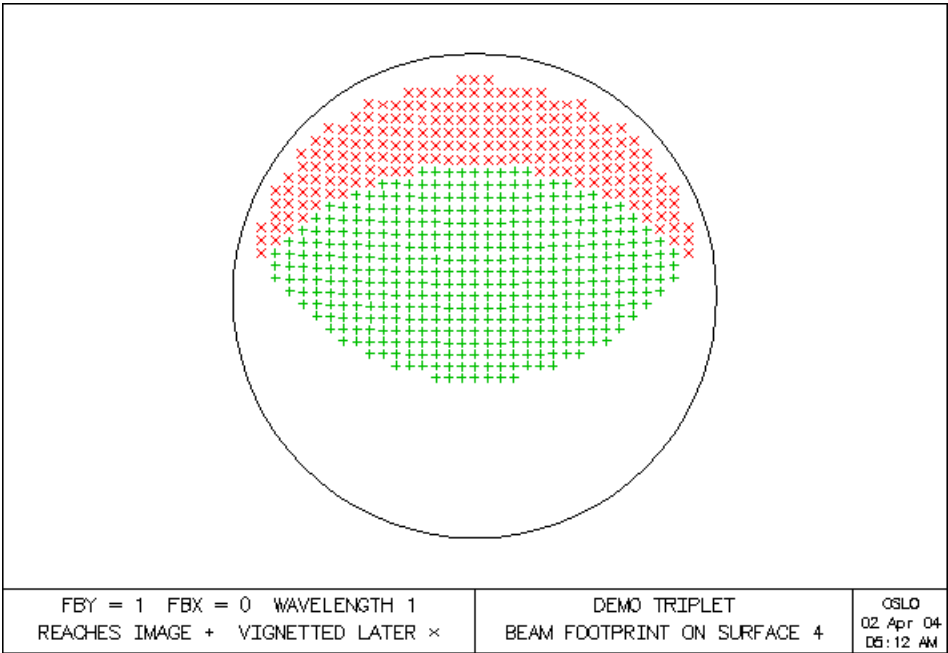
where

fbx and *fby* are fractional object coordinates from which the rays are to be traced.

surface is the number of surface at which the beam footprint is to be plotted.

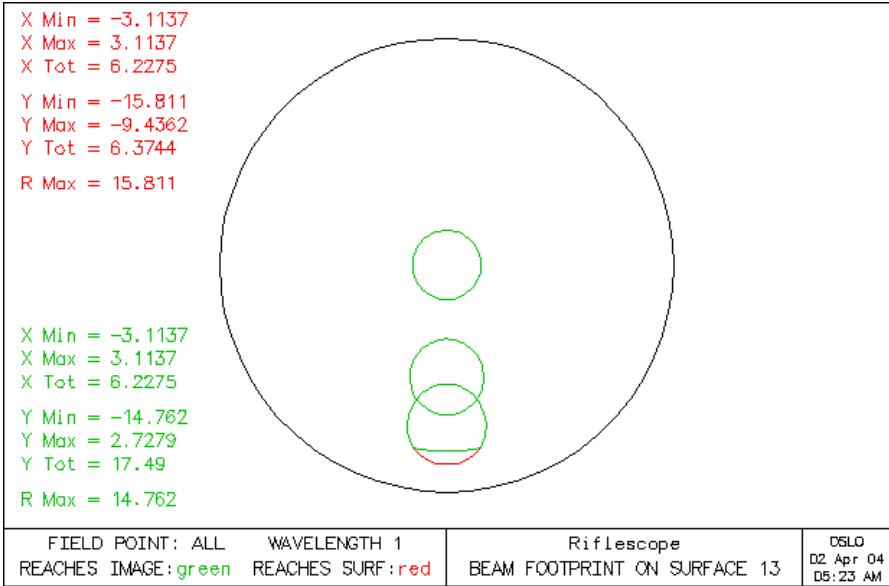
number_of_rays is the number of rays to be traced across the diameter of the entrance pupil. Thus, the starting grid of rays is *number_of_rays* × *number_of_rays*.

wavelength is the number of the wavelength in which the rays are traced.



Std Prm Beam Footprint Outline

The **footpro** CCL command draws the outline of pupil footprints on a surface. This analysis is similar to the aforementioned Beam Footprint analysis except that pupil footprints from several field points can be drawn in one analysis. An accounting of overall dimensions of the footprints is reported as well.



NAs and F-numbers

Compute real-ray based numerical apertures and f-numbers for the current object point. This is a CCL function. See code in **eval_system.ccl** for details.

Ghosts

The CCL command **ghosts** finds the locations of second-order (single reflection) ghost images for the surfaces in the specified range.

The **ghosts** command computes the characteristics of the ghost images formed by pairing each possible combination of surfaces and treating them as a pair of reflectors. The routine stores the original lens in the file "ghostbkp.len." The lens data in memory is modified during the computation but the original lens is restored at the end of the command. Because of the file manipulation and lens modification requirements, a lens to be used with **ghosts** should have no non-sequential surface groups and should contain only catalog glasses.

The **ghosts** routine uses exact ray tracing to locate the ghost images. The exact ray trace is used to approximate paraxial rays in computing the blur circle radius, relative image size, etc., in order to reduce the likelihood of a ray failure. **ghosts** does not check for dummy surfaces or mirrors in the system; therefore, the routine may find ghosts for surfaces that are not of interest.

Reported are

DISTANCE FROM IMS - the ghost image location measured from the primary image surface to the ghost

BLUR CIRCLE RADIUS - the blur circle radius of the out-of-focus ghost at the primary image surface

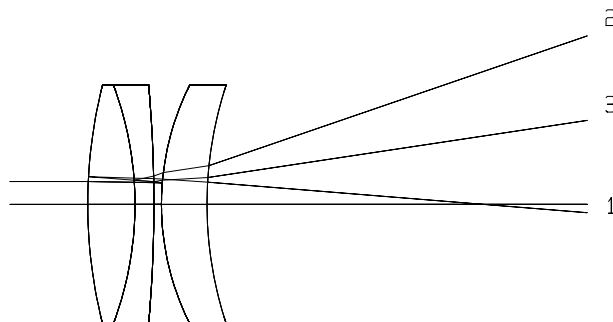
RELATIVE IMAGE SIZE - the size of the out-of-focus ghost relative to the primary image (ratio of the chief ray height on the primary image surface for the ghost system to the chief ray height on the primary image surface for the primary system)

EXIT PUPIL RATIO - the ratio of the exit pupil radius of the primary system to the exit pupil radius of the ghost system

SYSTEM EFL - the effective focal length of the ghost system

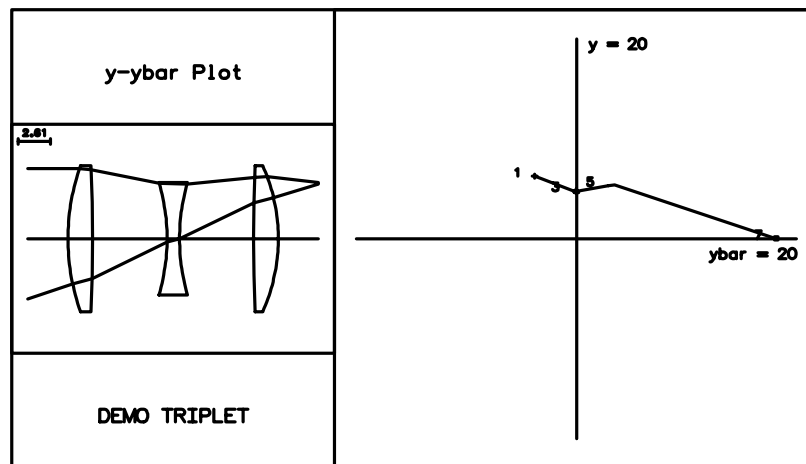
*GHOSTS 4 4

GHOST SRFS	REFLECTED IMAGE			GHOSTS	
	DISTANCE FROM IMS	BLUR CIRCLE RADIUS	RELATIVE IMAGE SIZE	EXIT PUPIL RATIO	SYSTEM EFL
4 1					
.	-13.451525	5.351992	0.745226	0.486763	37.700517
4 2					
.	-65.100262	107.280267	2.050460	2.021266	-9.102363
4 3					
.	-73.255107	55.300991	1.458921	1.462078	-19.869926



Y-Ybar

The CCL command **yybar** can be used to display the $y - \bar{y}$ diagram (sometimes called the Delano diagram) for the current lens. The $y - \bar{y}$ diagram is a plot of the paraxial axial ray height (y) versus the paraxial chief ray height (\bar{y}). Thus, paraxial pupils correspond to points on the vertical axis (where $\bar{y} = 0$) and paraxial images correspond to points on the horizontal axis (where $y = 0$). For more information on the $y - \bar{y}$ diagram see Delano⁴ and Shack.⁵



Std Prm Narcissus

The CCL command **nars** computes narcissus effects for specified cold stop and scanner surface locations. Note that this command modifies the current lens data; you should save your lens before using this command. (A copy of the starting lens will be saved in the file "nars_tmp.len".)

nars calculates the effects of reimaging the cold stop onto a detector (the image surface). Narcissus results when the cold stop is imaged back toward the detector by an unwanted surface reflection. Narcissus analysis is performed with each surface in front of the designated scanner surface individually evaluated as a reflective surface. The resulting out-of-focus image of the cold stop at the detector is described by a blur circle and an emission angle.

Note that **nars** traces an exact ray that approximates a paraxial ray. The results are scaled to determine the clipping surfaces in both the forward (from the cold stop toward the reflecting surface) and reverse directions.

For each surface prior to the scanner surface, the following pieces of narcissus data are presented.

The surface number of the clipping aperture for the narcissus image and a letter indicating whether the beam was clipped on the forward (**F**) or reverse (**R**) path.

4. E. Delano, "First-order design and the y, \bar{y} diagram," Appl. Opt. 2, 1251- 1256 (1963).

5. R. V. Shack, "Analytic system design with pencil and ruler – the advantages of the $y - \bar{y}$ diagram," in *Applications of Geometrical Optics*, Proc. SPIE Vol. 39, pp. 127-140 (1973).

Two paraxial quantities are reported: **RN*PY*PI** is the product of refractive index (on the incident side of the surface), paraxial axial ray height, and paraxial axial ray angle of incidence (i.e., yni). This value can be related to a paraxial prediction of blur circle radius. (Smaller values of this quantity lead to smaller blur circles and hence a greater potential for narcissus problems). **PI/PIC** is the ratio of the paraxial axial ray angle of incidence to the paraxial chief ray angle of incidence (i.e., i/\bar{i}). This quantity is related to the field-of-view fraction over which the center of the detector is within the narcissus blur circle. For more information on the use of these paraxial quantities see Howard and Abel.⁶

The **BLUR CIRCLE RADIUS** is the size of the out-of-focus image of the cold stop at the detector, taking into account the clipping by the denoted aperture.

The **NARCISSUS HALF ANGLE** is the (half) cone angle formed by the narcissus image rays. This angle is reported in degrees.

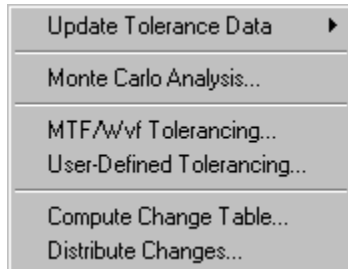
The **NARCISSUS INTENSITY RATIO** is the ratio of intensity of the narcissus image to the intensity of the nominal image. If the narcissus half angle is denoted by n and the axial ray angle at the image by u , the narcissus intensity ratio is given by $\sin^2(n)/\sin^2(u)$.

Std Prm Convert to Double Pass

Converts user specified surface into a mirror and replicates preceding surfaces back to object. Saves modified "filename.len" as "filename.dbl.len". Use once to model a double pass system test, or use repeatedly to model a resonant cavity.

6. J. W. Howard and I. R. Abel, "Narcissus: reflections on retroreflections in thermal imaging systems," Appl. Opt. **21**, 3393-3397 (1982).

Overview



Designing an optical system requires the designer to specify the values of essential construction parameters such as radius of curvature, thickness, irregularity, refractive index, dispersion, centration, and tilt for each surface, component, and group. As one part of the design task, *optimization* defines the best target values for these construction parameters. Another major part of the design task is *tolerancing*, which defines how precisely the components of the design need to match these target values.

The approach to tolerancing in OSLO consists of starting with the current “perfect” system, perturbing certain construction parameters within a certain range and then analyzing the performance of the final “perturbed” system. The choices of what construction parameters to perturb and the ranges of perturbation, are decided by the user and are part of the tolerancing process. OSLO provides excellent tools to be used in this tolerancing process, but a successful tolerance analysis is also heavily dependent on an understanding of the manufacturing process - which is individual to each system.

The initial tolerance data for a new lens in OSLO is based on the default manufacturing tolerances outlined in the ISO 10110 drawing standard. Users are able to change this initial tolerance data by referencing the correct class of data: Surface Tolerance Data, Component Tolerance Data, and User-Defined-Group Tolerance Data. After setting the tolerance data, OSLO allows the user to choose among four different tolerance methods to calculate sensitivity and inverse sensitivity analyses.

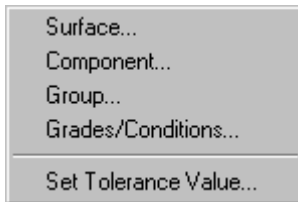
The menu commands relating to tolerancing are:

Menu Item	Page
Update Tolerance Data... Sets the values of construction parameters that will be varied during tolerance analysis. User groups can be defined and tolerance grades and conditions can be set.	398
Monte Carlo Analysis... Creates a user-defined number of virtual optical systems. Each virtual optical system is perturbed according to the defined probability distributions of the tolerance data. Statistical analysis is performed on the virtual systems and reported.	413
MTF/Wvf Tolerancing... Compute the change in the MTF or wavefront as a function of the lens tolerances, using the fast and efficient Hopkins-Tiziani-Rimmer (HTR) method.	416
User-Defined Tolerancing... Compute the effects of the lens tolerances on a user-defined tolerancing (error) function.	421

Compute Change Table... Compute the change in a prescribed set of system performance measures, given a change in a group of system parameters (tolerances). 424

Distribute Changes... Compute the allowed change of a system parameter (tolerance), for the given change in a designated system performance measure. This is the converse analysis of the “Compute Change Table” calculation. 431

Update Tolerance Data



Tolerance data describes the allowable variation in the constructional parameters of the lens. Tolerance data is separated into three classes - each having its own data editing window:

- Surface Tolerance
- Component (Air-to-Air) Tolerances
- Group (User-Defined) Tolerances

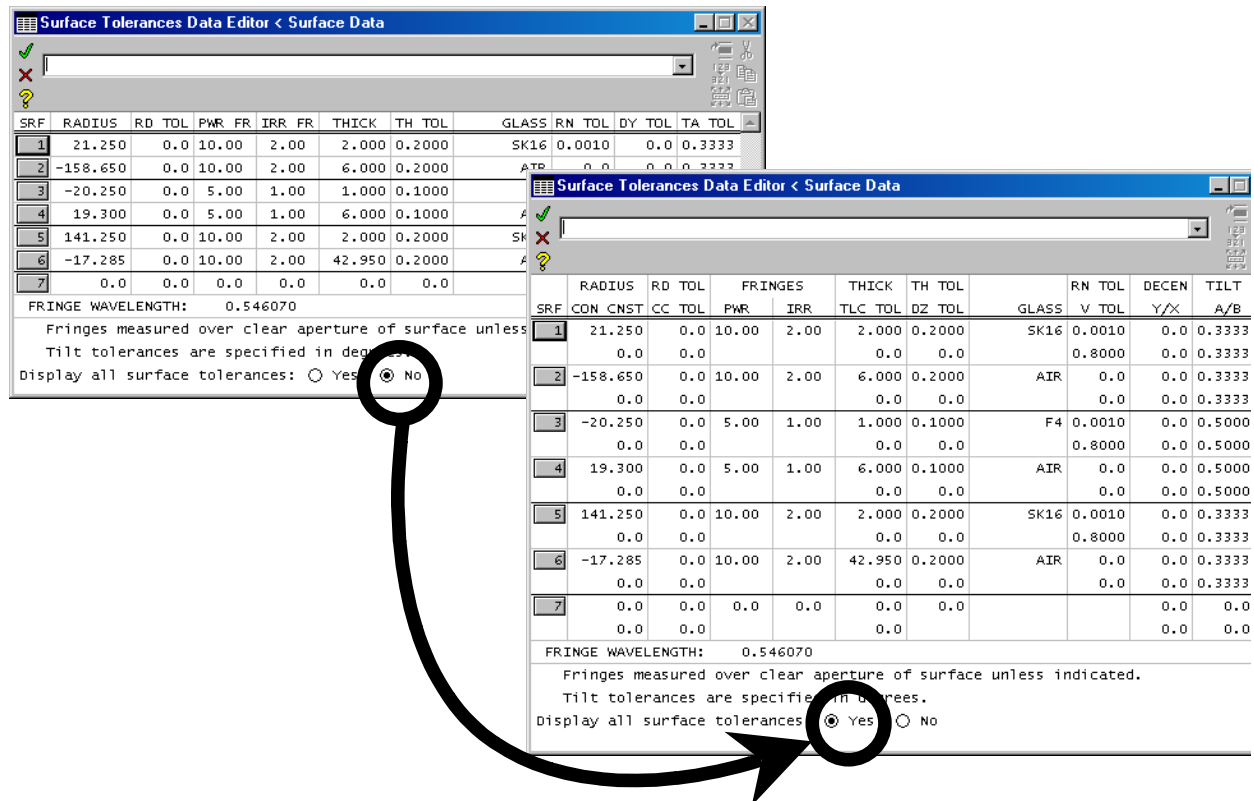
These tolerance classes are calculated independently such that a surface tilt tolerance, for example, affects only the tilt of that surface, and not any component or group tilt.

The tolerance spreadsheet editors (**Surface**, **Component** and **Group**) also display the current value of some of the construction data for the lens, in addition to the tolerance value assigned to that construction data. In the tolerance data spreadsheets, the magnitude of the tolerance data can be changed, but the construction data items they relate to, if shown, cannot be changed in value.

The default tolerance value for each construction item is established according to the ISO 10110 standard. These default tolerance values may be found in Chapter 9, “Tolerancing”, of the OSLO Optics Reference manual. However, the tolerances can easily be changed by the user to be any desired value. The tolerances that differ from the ISO standard will be saved with the lens data.

Surface Tolerance Data

The surface tolerance data editor spreadsheet has both “Expanded” and “Condensed” views. The spreadsheet can be switched between the two views by toggling the **Display all surface tolerances** radio button at the bottom of the spreadsheet. When all surface tolerances are displayed, a second line of tolerance data is shown for each surface.



Note that **Insert Before**, **Insert After**, **Delete**, and **Reverse** are not valid operations in the tolerance data spreadsheets.

The tolerance data for each surface is contained in one or two lines designated by the surface number in the “SRF” column. The surface numbers themselves are displayed on rectangular grey Row Buttons. You can select a surface for editing by clicking on the row button for that surface. Once you have a surface selected, a range of surfaces may be selected by clicking on the surface where you want the range to end. The range may extend to either preceding or subsequent (but not both) surfaces from the initially selected base surface. Clicking the CANCEL button will cancel the current surface selection.

After selecting one or more rows, you will note that the Cut, Copy, and Paste toolbar buttons in the upper right corner of the spreadsheet window is active. The same options are also available from the right-mouse-button-click pop-up menu. This means that the editing features described in Chapter 3, 'Toolbar Icons' can be used on the current selection. You can Cut or Copy the current selection to a clipboard, and then Paste it back in the spreadsheet at a different location. These operations are performed on the tolerance data only, not the lens construction data. See page 116 for a complete discussion of these editing functions.

The value of each item may be entered directly by selecting the cell and typing in the new value. With the exception of the radius of curvature tolerance, each tolerance cell may be activated by clicking on it to produce a pop-up list. From the list you can choose whether the tolerance item is to be user-specified (“Direct Specification”) or to be ISO default value (“ISO 10110 default”). Note that the radius of curvature entry is an exception because the ISO standard has default values for surface form tolerances only when specified in interferometric terms, i.e., fringes.

Prm

The tolerances may also be specified as “Hold at current value” (OSLO Premium only), indicating that the tolerance is not to be adjusted in inverse sensitivity mode.

The wavelength used for the specification of the fringe tolerance items may be changed in the FRINGE WAVELENGTH cell. The wavelength may be entered directly, or selected from the pop-up list that appears when you click on the already selected cell. The default wavelength value is 0.54607 μm .

A complete listing of all parameters included in the surface tolerance data is given by the following table.

Tolerance type	Construction parameter available for tolerancing	Label
<i>(Read Only)</i>	Radius of curvature	RADIUS
<i>(Read Only)</i>	Conic constant	CON CNST
Surface	Radius of curvature	RD TOL
Surface	Conic constant	CC TOL
Surface	Power fringes (error in radius of curvature)	FRINGES PWR
Surface	Irregularity fringes (cylindrical error)	FRINGES IRR
<i>(Read Only)</i>	Surface Thickness	THICK
Surface	Surface Roll - TLC	TLC TOL
Surface	Element thickness	TH TOL
Surface	Air space	TH TOL
Surface	Axial surface shift	DZ TOL
<i>(Read Only)</i>	Glass type	GLASS
Surface	Refractive index	RN TOL
Surface	Abbe V-number	V TOL
Surface	Surface decentration – Y	DECEN Y
Surface	Surface decentration – X	DECEN X
Surface	Surface tilt – TLA	TILT A
Surface	Surface tilt – TLB	TILT B

Radius of curvature/Power fringes

These two items show the effect of a change in the radius of curvature of a surface. The perturbation may be specified directly (radius tolerance) or in terms of interference fringes (power fringes). ISO 10110 specifies a default tolerance in terms of fringes. For a nominal radius of curvature R , a perturbation in the radius of ΔR , and a test area diameter D , the number of interference fringes N at wavelength λ is given by

(10.1)

$$N = \frac{2\Delta R}{\lambda} \left[1 - \sqrt{1 - \left(\frac{D}{2R} \right)^2} \right]$$

If the ratio D/R is small, Eq. (10.1) may be approximated by

(10.2)

$$N \approx \left(\frac{D}{2R} \right)^2 \frac{\Delta R}{\lambda}$$

A positive radius of curvature perturbation (i.e., the radius tolerance or number of spherical error fringes is positive) means that the absolute value of the radius of curvature increases. The change in radius of curvature is implemented by displacing the center of curvature of the surface; the surface vertex does not move.

Radius of curvature tolerances are specified in lens system units. Power fringe tolerances are specified in units of fringe spacing, where one fringe spacing is equal to one-half of the specified light wavelength.

Irregularity fringes

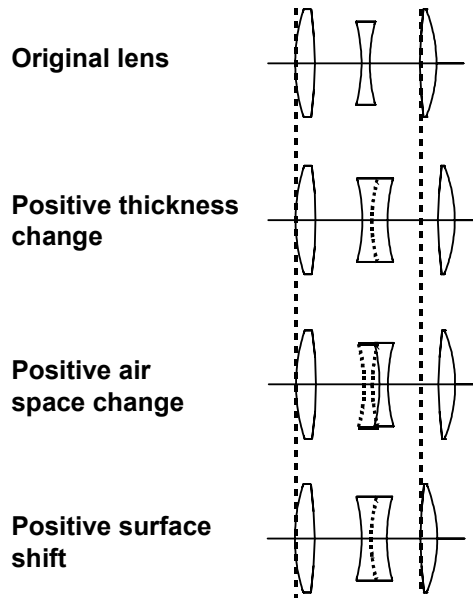
As defined in ISO 10110, Part 5: “The irregularity of a nominally spherical surface is a measure of its departure from sphericity.” In other words, the irregularity is a difference in the radii of curvature between the yz and xz meridians. This difference in radii ΔR is related to the number of irregularity fringes N via Eq. (10.1). When computing tolerance operands, the absolute value of the yz radius is decreased by $\Delta R/2$ and the absolute value of the xz radius is increased by $\Delta R/2$.

Irregularity fringe tolerances are specified in units of fringe spacing, where one fringe spacing is equal to one-half of the specified light wavelength.

Air space/element thickness/axial surface shift

All three of these calculations use the value assigned for the thickness tolerance as the amount of perturbation. The air space calculation shows the effect of a change in the air spaces between lens elements, while the element thickness calculation shows the effect of a change in element (glass) thicknesses. When an axial surface shift is applied to a surface, the sum of the thicknesses adjacent to that surface remains unchanged. These perturbations are illustrated in the figure below.

Spacing, thickness and shift tolerances are specified in lens system units.



Refractive index

When a refractive index tolerance analysis is performed, the index of refraction for each wavelength is changed by the prescribed tolerance amount. By default, only surfaces with a nominal refractive index greater than 1.0 are assigned a non-zero refractive index tolerance.

Abbe V-number tolerance

The Abbe V-number tolerance is given as a percentage (the default value is 0.8%), and the resulting index perturbations are computed using a linear expression for refractive index n as a function of chromatic coordinate ω .

Referring to the OSLO 6.1 Optics Reference, p. 57 (p. 66 in the electronic version), the chromatic coordinate ω as a function of wavelength λ is given by

$$\omega(\lambda) = \frac{\lambda - \lambda_0}{1 + 2.5 (\lambda - \lambda_0)} \quad (10.3)$$

where λ_0 is the central wavelength, i.e., wavelength 1 in OSLO. (The index at this wavelength is denoted by n_0 .) Assuming a linear dependence, then, the refractive index is given by

$$n(\omega) = n_0 + \nu_1 \omega \quad (10.4)$$

If ω_S denotes the chromatic coordinate for the short wavelength (λ_S ; wavelength 2 in OSLO) and ω_L denotes the chromatic coordinate for the long wavelength (λ_L ; wavelength 3 in OSLO), it can be shown that the linear coefficient ν_1 is related to the Abbe V-number [$V = (n_0 - 1)/(n_S - n_L)$] via

$$\nu_1 = \frac{n_0 - 1}{V(\omega_S - \omega_L)} \quad (10.5)$$

(The index of refraction at λ_S is n_S and the index at λ_L is n_L). If the Abbe number changes from V to V' (and, hence, the linear coefficient changes from ν_1 to ν_1' and the index changes from n to n'), then the change in refractive index Δn at chromatic coordinate ω is

$$\begin{aligned} \Delta n &= n'(\omega) - n(\omega) = n_0 + \nu_1' \omega - (n_0 + \nu_1 \omega) = \omega(\nu_1' - \nu_1) \\ &= \frac{(n_0 - 1)\omega}{\omega_S - \omega_L} \left(\frac{1}{V'} - \frac{1}{V} \right) \end{aligned} \quad (10.6)$$

Finally, if V changes by a fractional amount ε , (i.e., the Abbe number changes by 100 ε %), then $V' = (1 + \varepsilon) V$ and the change in index is

$$\Delta n = \frac{(n_0 - 1)\varepsilon \omega}{V(1 + \varepsilon)(\omega_L - \omega_S)} = \frac{(n_S - n_L)\varepsilon \omega}{(1 + \varepsilon)(\omega_L - \omega_S)} \quad (10.7)$$

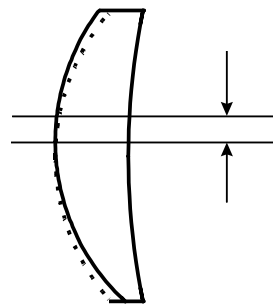
Typically, for the accurate computation of the refractive index itself, one would use at least a quadratic expression for the index as a function of chromatic coordinate. However, the linear expression is sufficiently accurate for the computation of the Δn needed for tolerancing and provides a straightforward way of converting a single tolerance number into an index perturbation for each wavelength. Note that the Abbe number tolerance does not affect the refractive index at the central wavelength, where $\omega = 0$.

Surface decentration

The surface decentration calculation shows the effect of decentering a surface. The decentration is performed in the local yz plane, i.e. it is a DCY decentration.

Decentering tolerances are specified in lens system units.

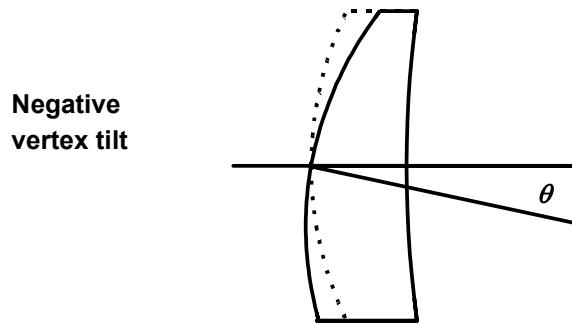
Positive decenter of first surface radius



Surface tilt

The surface tilt tolerance calculation shows the effect of tilting a surface about its vertex by the tilt tolerance amount. The tilt is performed about the local x-axis of the surface, i.e., it is a TLA tilt, as illustrated below.

Surface tilt tolerances are specified in decimal degrees.



Power error tolerance

The conversion from fringes of power or irregularity error to a change in radius of curvature uses the exact expression given by Eq. (10.2). For the majority of surfaces, the difference between these two conversions is negligible; however, the exact expression maintains accuracy for all values of the nominal radius of curvature.

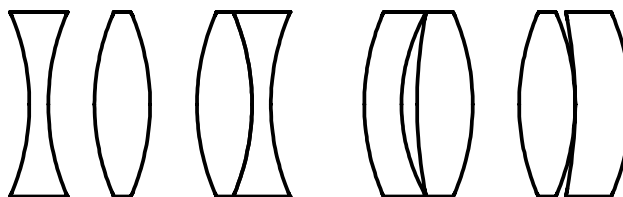
Decentration and tilt tolerances

Usually the x decentration tolerance will be the same as the y decentration, and the β tilt the same as the α tilt. In fact, if the “couple x and y tolerances” operating condition (see section “Grades/Conditions” on page 409) is true, changing the y decentration tolerance will also change the x decentration, and the β tilt will follow the value of the α tilt. If the x and y values are the same, this means that the perturbation has a magnitude equal to the tolerance value, but its azimuthal orientation is uniformly (randomly) distributed between zero and 2π .

Component (Air-to-Air) Tolerance Data

For the tolerancing calculations, OSLO determines that a “*component*” (or “*air-to-air component*”) is defined to be two or more real surfaces and the intervening media (glasses), bounded on both sides by air. A “real” surface, as distinguished from a “dummy” surface, separates media of different refractive indices. For

example, a single element lens, a cemented doublet, and a cemented triplet are all air-to-air components, but an air-spaced doublet is not, as illustrated below.



air-to-air components

not air-to-air components

Component tolerances are entered into their own spreadsheet. Decentration and tilt tolerances may now be applied in both the y and x directions. Default tolerances are assigned for component decentration and tilt about the center of curvature for the first surface of each component. These defaults are not given explicitly in ISO 10110, but are derived from the default surface tilt tolerances and the relationships given in Ref. ¹.

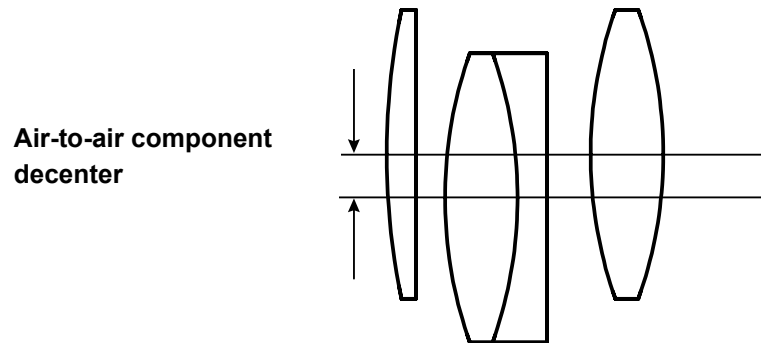
A complete listing of component tolerances is in the table below:)

Tolerance type	Construction parameter available for tolerancing	Label
Component	Component decentration – Y	DECENTRATION DCY
Component	Component decentration – X	DECENTRATION DCX
Component	Component tilt (aperture) – TLA	CLEAR APERTURE TILT ALPHA
Component	Component tilt (aperture) – TLB	CLEAR APERTURE TILT BETA
Component	Component Tilt (center of curvature) – TLA	CENTER OF CURV TILT ALPHA
Component	Component Tilt (center of curvature) – TLB	CENTER OF CURV TILT BETA

1. OSA Standards Committee, *ISO 10110 Optics and Optical Instruments – Preparation of drawings for optical elements and systems: A User's Guide*, R. K. Kimmel and R. E. Parks, Eds. Optical Society of America, 1995, p. 39.

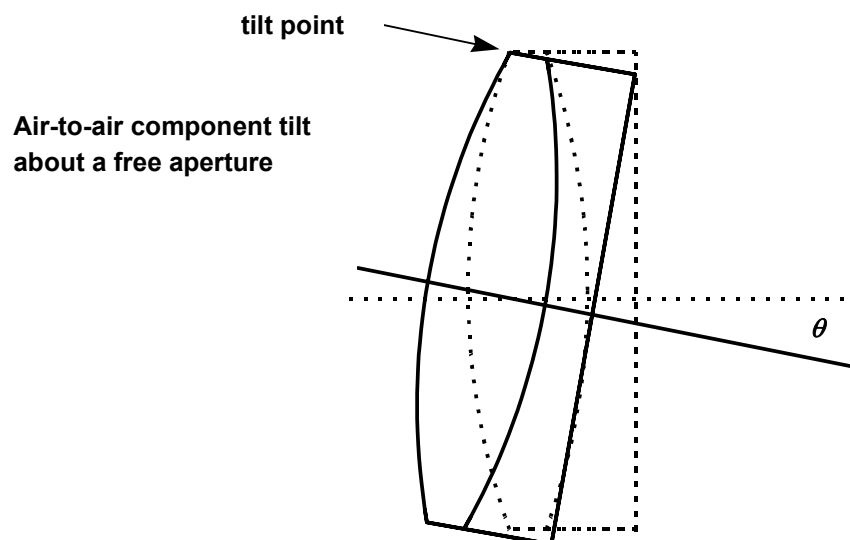
Component decentration

The component decentration calculation uses the value of the surface decentration tolerance for the first surface of the component as the perturbation amount. The decentration is performed in the local yz plane.

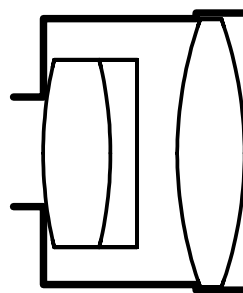


Component tilt about the clear aperture Component tilt about the center of curvature

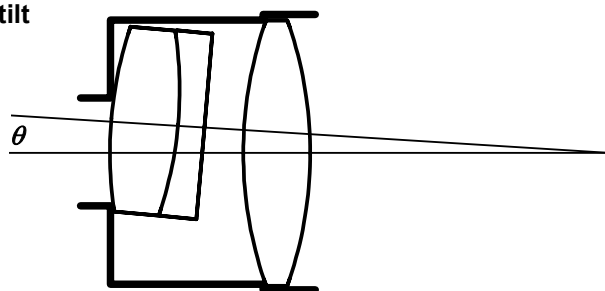
The component tilt calculations use the values of the surface tilt tolerances for the first and last surfaces of the component as the perturbation amounts. The tilt perturbation is performed separately for a tilt applied at the first surface and for a tilt applied at the last surface of the component. The tilts are applied about an axis that is perpendicular to the local yz plane. The meanings of the component tilt perturbations are illustrated below.



Optical elements
with no tilt



Air-to-air component tilt
about the center of
curvature



Prm Group (User-Defined) Tolerance Data

As you saw previously, OSLO considers a “Component” to be a range of surfaces bounded on both sides by air. In addition to this, you can also define your own groups of surfaces and apply tolerances to these groups. Three types of perturbations are available:

- decentration (x and y)
- axial shift (z)
- tilt (α and β)

For each tolerance, you must define the group by indicating the surface numbers of the first (**FSN**) and last (**LSN**) surfaces of the group. OSLO does not define any groups by default. The meanings of the remaining columns in the spreadsheet depend upon which type of tolerance has been selected. A complete listing of possible group tolerances are listed in the table below:

Tolerance type	Construction parameter available for tolerancing	Label
Group	User-Group Decentration – Y	GROUP DECENTRATION (UDC) DCY
Group	User-Group Decentration – X	GROUP DECENTRATION (UDC) DCX

Group	User-Group Axial Shift	GROUP AXIAL SHIFT (USH) DCZ
Group	User-Group Tilt – TLA (tilt about specified point)	GROUP TILT (UTL) TLA
Group	User-Group Tilt – TLB (tilt about specified point)	GROUP TILT (UTL) TLB

Decentration

For group decentration, the active columns are used for the y and x decentration tolerances. If the “couple x and y tolerances” operating condition is active, changing the y tolerance will also change the x tolerance to the same value.

Axial Shift

For group axial shift, the only additional data is the shift (z) tolerance. A group axial shift perturbation is applied in an analogous fashion to the surface axial shift perturbation, i.e., the total distance between surfaces FSN – 1 and LSN + 1 is unchanged.

Tilt

For group tilt, the α and β tilt tolerances must be specified. Again, these two tolerance values are coupled if the “couple x and y tolerances” operating condition is active. In addition, the tilt vertex point (i.e., the center of rotation for the tilt) must be specified. This point is located by specifying a position (**TOY**, **TOX**, **TOZ**) relative to the vertex of a designated surface (**TOSN**). Note that the surface that locates the tilt vertex is not required to be a member of the group. Thus, the group may be tilted about any point in space. For example, to tilt about the vertex of the last surface of the group, **TOSN** would be set equal to **LSN**, and **TOY**, **TOX**, and **TOZ** would be zero. If **TOSN** is left equal to zero, the first surface of the group (**FSN**) will be used to locate the tilt point origin. If you wish to tolerance a surface tilt about a point that is not the vertex of the surface, note that you may define a group consisting of a single surface (if **FSN** = **LSN**).

The current values of the user-defined group tolerances may be accessed via the CCL global variables given in the table below. The number of currently defined user-defined group tolerances is given by the integer variable *Numugt*. In the table, the array index i is the number of the group tolerance, so $1 \leq i \leq \text{Numugt}$.

Variable	Interpretation
Ugt_type[i]	Type of user-defined group tolerance: decentration = 1, axial shift = 2, tilt = 3 (integer)
Ugt_fsn[i]	First surface of group (integer)
Ugt_lsn[i]	Last surface of group (integer)
Ugt_tosn[i]	Tilt vertex origin surface number (integer)
Ugt_dcx[i]	Group x -decentration tolerance (double)

Ugt_dcy[i]	Group y -decentration tolerance (double)
Ugt_dcz[i]	Group axial shift tolerance (double)
Ugt_tla[i]	Group α -tilt tolerance (double)
Ugt_tlb[i]	Group β -tilt tolerance (double)
Ugt_tox[i]	Tilt vertex x -coordinate (double)
Ugt_toy[i]	Tilt vertex y -coordinate (double)
Ugt_toz[i]	Tilt vertex z -coordinate (double)

Prm Grades/Conditions

The tolerancing operating conditions control various option settings of the tolerancing process.

Couple x tolerances to y tolerances (command: **txyc**)

If this option is “on”, changing a y decentration or α tilt tolerance will also change the corresponding x decentration or β tilt. The y or α tolerance is the “controlling” value; changing an x or β tolerance will only change that item.

Tilt tolerance units (command: **ttun**)

By default, tilt tolerances are specified in degrees (as are all coordinate tilt angles in OSLO). You can, however, change the units used for all surface, component, and group tilt tolerances. The available units are degrees, radians, minutes of arc, and seconds of arc. Changing the units does not cause any scaling of previously entered tolerance values. For example, if the tilt tolerance units are degrees and a surface has a tilt tolerance specified as “0.1”, changing the tilt tolerance units to radians means that the surface now has a tilt tolerance of 0.1 radians.

The integer read-only global variable **ttun** indicates the current specification of the tilt tolerance units. The values are:

ttun	Tilt tolerance units
0	degrees
1	radians
2	minutes of arc
3	seconds of arc

Threshold change for display (command: **toth**)

This is the default value of the threshold used in *MTF and RMS Wavefront Tolerancing* for the display of individual tolerance performance changes.

Tolerance limits, increments and grades

These values are used only by the *MTF and RMS Wavefront Tolerancing* routines. The operating conditions determine the minimum and maximum allowed tolerance values, and the increment "unit" for the computed tolerances. The tolerance grade indicates the relative "tightness" or "looseness" of a tolerance. Because the tolerances are specified in many different units (number of fringes, length, percentage, degrees, minutes of arc, etc.), it can be difficult to judge the relative cost of a tolerance based solely on its numeric value. The tolerance grades can provide a kind of "normalization", so that tight tolerances can be more easily distinguished from loose tolerances. Each tolerance type (power fringes, air space, refractive index, etc.) may be assigned four grades: A, B, C, and D. Grade A tolerances are the tightest, and grade D tolerances are the loosest. Since there are four tolerance grades, three transition points must be defined. The data for the transition from grade A to grade B, and the transition from grade B to grade C are clearly marked. The Maximum Value of the tolerance is also used as the transition point from grade C to grade D.

The default values of the tolerancing operation conditions related to tolerance limits, increments and grades are given in the table below.

Tolerance type	Minimum value (A)	Maximum value (C-D)	Increment	A-B transition	B-C transition
Radius of curvature (mm)	0.0200	1.0000	0.0200	0.0500	0.200
Power fringes	1.00	20.00	1.00	4.00	10.00
Irregularity fringes	0.10	3.00	0.50	1.00	2.00
Conic constant	1.0e-08	0.1000	1.0e-06	0.0001	0.0100
Air space (mm)	0.0100	0.5000	0.0100	0.0500	0.2000
Element thickness (mm)	0.0100	0.5000	0.0100	0.0500	0.2000
Index of refraction	0.0001	0.0020	0.0001	0.0003	0.0010
Abbe V-number (%)	0.2000	1.0000	0.1000	0.3000	0.8000
Surface decentration (mm)	0.0100	0.5000	0.0100	0.0300	0.2000
Surface tilt - degrees	0.0100	0.5000	0.0100	0.0500	0.3333
- radians	0.0003	0.0100	0.0003	0.0010	0.0060
- minutes of arc	1.00	30.00	1.00	3.00	20.00
- seconds of arc	60.0	1800.0	60.0	180.0	1200.0
Component/group decentration (mm)	0.0100	0.5000	0.0100	0.0300	0.2000
Component/group tilt - degrees	0.0100	0.5000	0.0100	0.0500	0.3333
- radians	0.0003	0.0100	0.0003	0.0010	0.0060

- minutes of arc	1.00	30.00	1.00	3.00	20.00
- seconds of arc	60.0	1800.0	60.0	180.0	1200.0

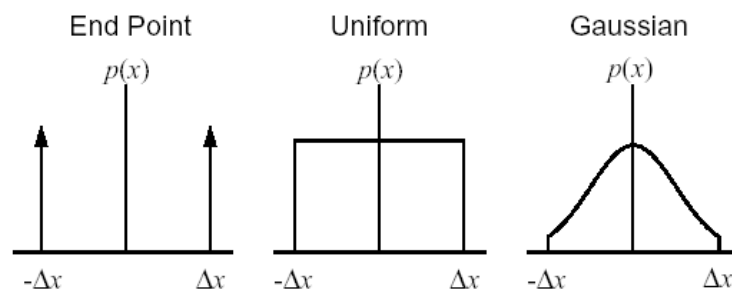
The values in the table are given for lens units in millimeters. If other units are being used, the default linear values will be scaled accordingly (from mm). Depending on the value of the Tilt Tolerance Units, the default tilt tolerances change as described in the table.

The default values for the grade B to grade C transition tolerances for power fringes, irregularity fringes, air space, element thickness, surface tilt, and component tilt are the default values from ISO 10110 for 10 mm to 30 mm aperture diameter parts. (See the Optics Reference.)

For programming access to these Tolerancing Conditions, refer to the following section.

Tolerance distributions

These values are used in computing the statistics of the estimated change in performance for the *MTF and RMS Wavefront Tolerancing* routines and in generating the random systems for Monte Carlo tolerancing. The available distributions are **uniform**, **Gaussian** (normal) truncated at the 2σ point, and **end-point**. See the Optics Reference, pp. 223–224 (hardcopy) or pp. 231-232 (electronic PDF file).



By default, decentration and tilt perturbations have a Gaussian distribution, while the other perturbations are uniformly distributed. The tolerance limits, increments, and distributions may also, of course, be set with commands. The commands are given in the table below. The complete command definitions may be found in the Help system.

Tolerance type	Limits, increments and grades command	Distribution command
Radius of curvature	tlrd	tdrd
Power fringes	tlsp	tdsp
Irregularity fringes	tlir	tdir

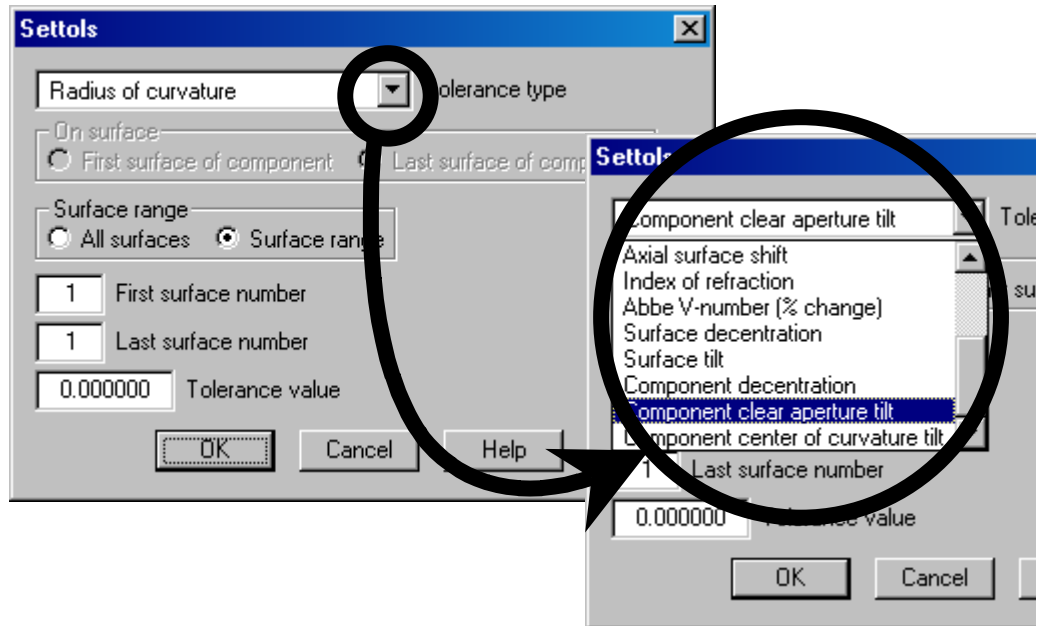
Conic constant	tlcc	tdcc
Air space	tlas	tdas
Element thickness	tlet	tdet
Index of refraction	tlrn	tdrn
Abbe V-number	tlvn	tdvn
Surface decentration	tlsd	tdsd
Surface tilt	tlst	tdst
Component/group decentra- tion	tlcd	tdcd
Component/group tilt	tlct	tdct

The current values of the tolerance limits, increments, and distributions are available as read-only variables with the same name as the above commands. The minimum, maximum, and increments are accessed as the members of a three-element array of double precision variables: element 0 is the minimum, element 1 is the maximum, and element 2 is the increment. For example, `tlrn[0]` is the minimum value for a refractive index tolerance, `tlrn[1]` is the maximum value, and `tlrn[2]` is the refractive index tolerance increment. The tolerance distributions are accessed as integer-valued read-only variables, with the following values:

Distribution variable	Perturbation distribution
0	uniform
1	Gaussian
2	end-point

Set Tolerance Value

The Set Tolerance Value window (**settols**) allows the user to change a variety of tolerance data to be a common value on many different surfaces at once. A common use of this command would be to set the value of a tolerance data type to zero, thereby removing from tolerancing calculations.



Monte Carlo Analysis

Prm

Monte Carlo analysis uses random numbers to generate a sequence of perturbed lenses, where the maximum magnitude of the perturbations is determined by the current values of the tolerances. Each random realization of the lens is constructed by generating random numbers having a prescribed probability density function and then using these random numbers along with the tolerances to perturb the construction parameters of the system. An advantage of Monte Carlo analysis is that all of the construction parameters may be perturbed simultaneously. After all of the perturbations are applied, the compensators (if any) are varied in an attempt to restore the performance of the lens as close as possible to its nominal state of correction. Analysis of the performance of the resulting systems provides a statistical prediction of the distribution of the final fabricated lenses. Because of the stochastic nature of the process, depending upon the lens and its sensitivity to its construction parameters, the Monte Carlo analysis may converge slowly to the true value of the performance statistics. In addition, since all of the parameters are varied simultaneously, it can be difficult to locate which parameters are the most sensitive. However, Monte Carlo analysis can be quite useful when used in conjunction with other tolerancing techniques.

The setup procedure for a Monte Carlo analysis is identical to that for user-defined tolerancing: construction of an error function and designating compensators (variables). The current error function will be used as the performance measure, so it is important that the error function accurately reflects the system performance, even in its perturbed state. For example, if you are tolerancing decentrations and tilts, the error function should contain rays on both sides of the entrance pupil, and both positive and negative field points, even for a nominally rotationally symmetric lens. If the current error function appears to assume rotational symmetry, a warning message to that effect will be issued before the analysis is begun.

Just as for user-defined tolerancing, the error function value will be the prime measure of the change in the lens performance. However, you can also perform a statistical analysis of any selected operands by giving them the name **tolop**. These operands may have zero weight if you don't want them included in the error function.

To begin a Monte Carlo analysis of the current lens, select "Monte Carlo Analysis" from the Options menu. In the dialog box, there are several controls for the computations.

Number of random systems to evaluate

This is the number of different random lenses that will be generated using the current tolerance values. As with all simulations, the more systems that are evaluated, the closer the resulting statistics should be to the "real world." Of course, this accuracy means that the analysis takes a longer time to perform. After beginning the computations, you may interrupt the routine before all of the specified systems have been evaluated by pressing the ESCAPE key.

Perturbation distributions

This option controls the distribution of the random perturbations applied to the lens. The default distributions are obtained from the tolerancing operating conditions (see section "Grades/Conditions" on page 409). The other three options (uniform, Gaussian, end-point) will apply the specified distribution to all perturbations.

Plot error function distribution

If you wish, you can plot both the cumulative probability (continuous curve) and a relative distribution histogram (vertical lines) for the error functions of the resulting ensemble of Monte Carlo systems.

After all of the requested systems have been evaluated, a statistical summary of the error function, compensators, and all "tolop"-named operands is presented. The definitions of the statistical quantities may be found in any statistics reference; we use the notation and terminology of Press, et. al.,² to which the reader is referred for more detail. In the following, N denotes the number of systems evaluated, and x_j denotes either the change in the error function or the change in an operand value.

The mean change \bar{x} is given by

$$\bar{x} = \frac{1}{N} \sum_{j=1}^N x_j \quad (10.8)$$

The standard deviation σ , is the square root of the variance σ^2 , where

2. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, Second Edition, Cambridge University Press, 1992, §14.1, pp. 610–615.

$$\sigma^2 = \frac{1}{N-1} \sum_{j=1}^N (x_j - \bar{x})^2 \quad (10.9)$$

The average deviation (or mean absolute deviation is)

$$\text{AVG DEV} = \frac{1}{N} \sum_{j=1}^N |x_j - \bar{x}| \quad (10.10)$$

The *skewness* is a non-dimensional version of the third central moment, and measures the shape and asymmetry of the distribution. A positive skewness means that the distribution has a “tail” extending toward positive values of x_j from the mean. Conversely, a distribution with a negative skewness has a tail extending toward negative values of x_j .

$$\text{SKEWNESS} = \frac{1}{N} \sum_{j=1}^N \left(\frac{x_j - \bar{x}}{\sigma} \right)^3 \quad (10.11)$$

The *kurtosis* is a non-dimensional version of the fourth central moment, and has a value of zero for a Gaussian (normal) distribution. A distribution with a positive kurtosis is more “peaked” than a Gaussian, while a negative kurtosis distribution is “flatter” than a Gaussian.

$$\text{KURTOSIS} = \left[\frac{1}{N} \sum_{j=1}^N \left(\frac{x_j - \bar{x}}{\sigma} \right)^4 \right] - 3 \quad (10.12)$$

The “+/-” ranges displayed for the mean and standard deviation are,³

respectively, the standard deviation of the mean (σ/\sqrt{N}) and the standard deviation of standard deviation $(\sigma/\sqrt{2(N-1)})$.

Also displayed for the error function and the “tolop” operands is a cumulative probability table. The cumulative probability (or probability distribution) is the probability that the value of the random variable is less than or equal to the specific value. So, the cumulative probability of the minimum change in the quantity is 0 % and the cumulative probability of the maximum change is 100 %. Also, the median change in a quantity is given by the 50 % cumulative probability value.

3. J. B. Kennedy and A. M. Neville, *Basic Statistical Methods for Engineers and Scientists*, Third Edition, Harper and Row, 1986, pp. 338–339.

MTF/Wvf Tolerancing

Prm

The *MTF and RMS Wavefront Tolerancing* routines are based on the analytic wavefront differential method introduced by Hopkins and Tiziani⁴ and expanded upon by Rimmer.⁵ The advantage of this method is that the effects of tolerance perturbations can be calculated from data obtained by ray tracing only the nominal system. This method is many times faster than methods that require that all necessary rays be retraced for each perturbed system in order to compute tolerance operands. This speed and efficiency makes it practical to compute tolerance effects using a numerically intensive computation, such as MTF.

MTF and RMS wavefront tolerancing

Evaluation type
☐ MTF (tangential and sagittal) ☐ RMS wavefront ☒ Tangential MTF only ☐ Sagittal MTF only

Analysis mode
☒ Sensitivity ☐ Inverse sensitivity

Report option
☐ Tolerances only ☒ Full tolerance output

Group output by
☐ Configuration, field point, item ☒ Item, configuration, field point ☐ Item, field point, configuration

Sort by
☒ Tolerance item ☐ Performance change ☐ Tolerance grade

Output option
☐ Compensator variables ☒ Perturbation equation

Tolerance item: Power fringes

Chromatic option
☐ Monochromatic ☒ Polychromatic Wavelength number: 1

20.000000 Spatial frequency (cycles/mm)

17.030000 Aperture divisions across pupil diameter

0.010000 Differential change in RMS wavefront/MTF for calculation

0.000000 Threshold change for display of individual tolerance data

Use cross terms in std. dev. calculation
☐ Yes ☒ No

Assign allowed tolerances to lens
☒ Yes ☐ No

Round allowed tolerances
☒ Yes ☐ No

Decouple x/y decenter/tilt tolerances
☒ Yes ☐ No

OK Cancel Help

4. H. H. Hopkins and H. J. Tiziani, "A theoretical and experimental study of lens centring errors and their influence on optical image quality," *Brit. J. of Appl. Phys.* **17**, 33-54 (1966).

5. M. P. Rimmer, "Analysis of perturbed lens systems," *Appl. Opt.* **9**, 533-537 (1970); "A tolerancing procedure based on modulation transfer function (MTF)," in *Computer-Aided Optical Design*, Proc. SPIE Vol. 147, pp. 66-70 (1978).

Like the user-defined tolerancing and the change table tolerancing, the *MTF*/wavefront tolerancing routines will use the current values of the assigned lens tolerances as the system perturbations. Default values for tolerances are assigned according to the table in ISO 10110, Part 11. The tolerancing may be run in one of two modes. In “sensitivity” mode, the current tolerance values are used to compute the change in system performance. In “inverse sensitivity” mode, the current tolerances are used to establish a functional relationship between the tolerances and performance. Then, using this data, the allowed tolerance for a specified change in performance is computed. Note that the allowed tolerances computed in inverse sensitivity mode are not automatically assigned to the lens tolerance data.

As in the user-defined tolerancing calculations, any defined variables will be used as compensating parameters. Thus, before using the *MTF*/wavefront tolerancing routines, make sure that the current optimization variables correspond to those system parameters that you wish to use as compensators. Even though the lens data will not be changed as part of the tolerance computations, it is always a good idea to save your lens before beginning any tolerance analysis.

The computations are performed at multiple field points as defined by your current field point set. The fundamental division in the performance output can be chosen to be “tolerance item”, “field” or “configuration”. It is important to keep in mind that although tolerancing using the Hopkins–Tiziani algorithm is very fast, it is quite memory intensive. The amount of memory required is roughly proportional to the product of the number of surfaces, number of wavelengths, number of field points, number of configurations, and the square of the number of aperture divisions across the pupil. Depending upon the amount of memory in your computer, you may need to reduce one or more of these quantities for efficient computation, although this is less of an issue now with the availability of computers with large amounts of memory.

Preparation of the system model

There are two setup steps before using the multi-field-point *MTF*/RMS wavefront tolerancing. First, the desired field points at which the analysis is to be performed should be entered in the field points set. Unlike the user-defined and Monte Carlo tolerancing, only one side of the field (usually $0 \leq \text{FBY} \leq 1$) needs to be entered. For the purposes of computing compensator values, the other side of the field will be simulated. If the lens or field of view is not symmetric, all of the desired field points should be entered and no simulation will be performed. The vignetting factors and the weight assigned to the field point will be used by the tolerancing calculations. The weight is used primarily during the calculation of compensator values; a greater emphasis will be placed on minimizing the performance change for field points with a higher weight. If you have a multi-configuration lens, the weights assigned in the configuration operating conditions (see section “Grades/Conditions” on page 409.) will also be used during the calculations. If you wish to exclude a configuration from the tolerancing computations, you can designate it as inactive in the configuration operating conditions spreadsheet.

The second step is the designation of compensators. This is done, as usual, by entering the desired parameters as variables. The use of compensators across configurations is completely analogous to the use of variables in multiconfiguration design. If the configuration number of the compensator (variable) is zero, that compensator is “global” and will have the same value in all configurations. If the configuration number is non-zero, that compensator will be

used only in the designated configurations. As an example, consider a two-position (i.e., two configuration) zoom lens. If the same back focal distance is to be used in both configurations, you would enter a single compensator, with a configuration number of “0”. If each configuration, on the other hand, is allowed to have its own back focal distance, you would enter two compensators; each would be the thickness of the last surface, but one would have a configuration number of “1” and the other would have a configuration number of “2”.

Analysis Options

Analysis mode

Like other tolerancing methods, MTF and RMS Wavefront Tolerancing may be performed in either sensitivity or inverse sensitivity mode. In sensitivity mode, the performance changes produced by all tolerance items with a non-zero tolerance value will be computed and contribute to the predicted total system change in performance. In inverse sensitivity mode, the tolerances will be adjusted so that each perturbation produces the requested change in *MTF* or RMS wavefront error. The presence of a non-zero tolerance indicates that the inverse sensitivity tolerance is to be computed. It is possible to “hold” selected tolerances at their current value during the inverse sensitivity analysis, i.e., their value will not be changed. To designate that a tolerance is to be held at its current value, click on the desired tolerance in the appropriate spreadsheet and select “Hold at current value” from the pop-up menu. Hold tolerances are denoted by an “H” next to the tolerance value in the output.

Report option

There are two options for the display of items related to the individual tolerances. If the “Tolerances only” option is selected, only the assigned tolerances (sensitivity) or allowed tolerances (inverse sensitivity) will be displayed. If the “Full tolerance output” option is selected, for each configuration and field point the change in *MTF* or RMS wavefront for both positive and negative perturbations of each perturbation will be displayed. Additionally, either the value of the compensators (for the positive perturbation) or the coefficients of the quadratic equation describing the change in performance as a function of perturbation will be displayed, depending upon the output option selected. The estimated overall change in *MTF* or RMS wavefront error will be displayed for both report options.

If you select the “Full tolerance output” option, you can display the individual performance changes in one of three groupings. The “Configuration, field point, item” option will display all of the individual tolerance perturbation performance changes together for each field point, at each configuration. The other two options will group together all of the performance changes produced by each tolerance item. For each item, the changes can be grouped first by configuration, and then by field point within each configuration (the “Item, configuration, field point” option) or first by field point, and then for each configuration at that field point (the “Item, field point, configuration” option). If you have a single configuration system, these last two options are equivalent. The same calculations are performed regardless of which output grouping you choose; only the format of the output is affected.

Output option

Your choice of “Output option” is only available if you choose “Full tolerance output” in “Report option” (see previous section).

If you choose “compensator variables,” the change in each compensator will be displayed for each tolerance item. The compensator values correspond to the change necessary for the plus perturbation. No matter what you choose for the “Output option”, the probably compensator change will be summarized at the end of the analysis.

If you choose “perturbation equation,” the coefficients A and B of a quadratic equation that relates perturbation value to performance change will be displayed. For a single perturbation T , the MTF can be expressed as

(10.13)

$$MTF = AT^2 + BT + C$$

and the RMS wavefront can be expressed as

(10.14)

$$RMS = \sqrt{AT^2 + BT + C}$$

(A , B , and C are not the same numerical values in the two above equations.) In both equations, C is a constant equal to the nominal value of the relevant parameter. In these equations, T is a scale factor for the perturbation. Thus, a perturbation equal to plus the tolerance value corresponds to $T = 1.0$; a perturbation of minus the tolerance value corresponds to $T = -1.0$; a perturbation of twice the tolerance value corresponds to $T = 2.0$, etc. With the values of A , B , and C , you can estimate the effect of changing the tolerance value without actually recomputing the tolerance operands. (The overall system performance is modeled as a general quadratic equation in all of the tolerance items, including cross terms. The A and B coefficients are sufficient to calculate the effect of scaling a single tolerance value.)

Threshold change

When the full tolerance output is selected, the default is to display the performance change for each perturbation. If you enter a non-zero value for the threshold, only perturbations that produce a performance change greater than this value (in magnitude) will be displayed. This can be helpful in reducing the amount of output and in determining the most sensitive tolerance items. All computed perturbations will be used in calculating the estimated performance change, regardless of their value relative to the threshold; only the displayed output is affected.

Assign allowed tolerances to lens

If you select “yes” for this option, tolerances computed during an inverse sensitivity analysis will be assigned to the lens, i.e., they become the tolerance values saved with the lens. If you select “no,” the allowed tolerances will be displayed as part of the output, but the lens data itself will not be changed.

Round allowed tolerances

By default, tolerances computed during an inverse sensitivity analysis will be rounded, using the increment defined in the tolerancing operating conditions. Thus, the actual change in performance may be slightly greater or smaller than the requested change, depending upon whether the tolerance is rounded up or down. If you do not want the tolerances to be rounded, select “no” for this option. The minimum and maximum tolerance limits will be used, regardless of whether the tolerances are rounded or not.

Decouple x/y decenter/tilt tolerances

Usually x and y decentration tolerances and α and β tilt tolerances are coupled and identical. In other words, there is a tolerance on the magnitude of the perturbation, but its orientation (i.e., the direction of the decentration or the axis of the tilt) is uniformly and randomly distributed between 0 and 2π (in the local xy plane). If the x and y tolerances are coupled, the effects of perturbations in both directions will be computed, and the tolerance for the two values will be set to whichever result of the inverse sensitivity analysis results in a tighter tolerance. Decoupling the x and y tolerances will allow different values to be assigned to the two tolerances.

Statistical performance analysis

For each surface in the lens with a non-zero tolerance value, the change in *MTF* or RMS wavefront will be reported for both plus and minus perturbations of the tolerance amount. The other information that is reported depends on which analysis option you have chosen.

After the surface-by-surface tolerance data, a statistical estimate of expected overall system performance change is displayed. The range of the performance change is given as a $\mu - 2\sigma$ to $\mu + 2\sigma$ range (μ is the mean change and σ is the standard deviation). Thus, assuming a normal (Gaussian) distribution for the performance change of the overall system (a result of the central limit theorem), approximately 95% of the systems will have a performance change within the displayed range. The mean and standard deviation are also displayed, for completeness. In computing these statistics, it is assumed that all system perturbations, with the exception of decentration and tilt, are uniformly distributed between plus and minus the tolerance value. Decentration and tilt are assumed to have a truncated normal (Gaussian) distribution, with the tolerance value equal to the 2σ truncation point.

If there are any compensation variables, their probable changes are displayed after the system performance summary. The probable change of the compensator is defined to be the 2σ (i.e., twice the standard deviation) value of the compensator change.

The following example output was generated using the OSLO demo triplet (demotrip.len) with back focal distance as a compensator (i.e., variable). The tolerance operand was on-axis polychromatic *MTF* at 20 cycles/mm. Aperture division across the pupil was 32. The default ISO tolerances for spherical error fringes were used. The option for reporting the perturbation equation coefficients “A” and “B” in equations (10.13) and (10.14) was chosen.

```
**MTF SENSITIVITY ANALYSIS - POLYCHROMATIC
TANGENTIAL (Y) MTF - SPATIAL FREQUENCY    20.00 CYCLES/MM
```

```

THRESHOLD CHANGE FOR INDIVIDUAL TOLERANCE DISPLAY:  --
TOLERANCE  SRF/  TOLERANCE              CHANGE IN MTF
  ITEM    GRP   VALUE  GRD  CFG  FPT    PLUS    MINUS      A      B
POWER FR   1   10.0    C   1   1   -0.007    0.006   -0.000452 -0.006553
POWER FR   2   10.0    C   1   1   -0.008    0.007   -0.000575 -0.007709
POWER FR   3    5.0    B   1   1    0.011   -0.013   -0.001148  0.012292
POWER FR   4    5.0    B   1   1    0.009   -0.010   -0.000640  0.009177
POWER FR   5   10.0    C   1   1   -0.010    0.009   -0.000693 -0.009281
POWER FR   6   10.0    C   1   1   -0.021    0.016   -0.002544 -0.018376

```

```

-----
TANGENTIAL (Y) MTF - SPATIAL FREQUENCY  20.00 CYCLES/MM
                                NOMINAL    LOW MTF    MEAN      STD DEV
                                MTF        W/  TOLS   CHANGE    (SIGMA)
  CFG  FPT  FBY      FBX      FBZ      MTF      W/  TOLS   CHANGE    (SIGMA)
  1    1    --      --      --      0.847    0.813   -0.002    0.016
-----

```

```

PROBABLE COMPENSATOR CHANGE (+/-)
NBR  TYPE  SRF    CHANGE
  1    TH    6    0.455308

```

User-Defined Tolerancing

The basic idea behind the user-defined tolerance routines is to tolerance the lens using the current error function. Thus, the tolerance criteria can be anything that can be computed as an operand. With the flexibility provided by the built-in operand types along with CCL/SCP operands, the user-defined tolerancing provides the capability to “tolerance on anything.”

The minimum requirement to do the *User-Defined Tolerancing* calculations is that an operand set must be defined. The operands may be a result of using the error function generator and/or an operands set of your own design. It is common to use the same error function that was used to optimize the lens as part of the tolerance calculations. When computing the performance of the perturbed system, the current surface tolerance values are used. See the section “Update Tolerance Data” on page 398 for information on assigning tolerance values to the lens. For simplicity, the component tolerances (decentration and tilt) are taken to be the same as the surface tolerance for the relevant parameter.

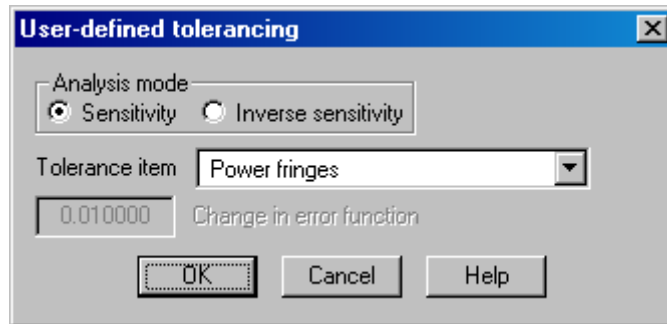
If you would like to see the effects of the perturbations on selected operands, in addition to the change in the overall error function, change the name of the operand you would like to track to **tolop**. **Tolop** must be the full name of the operand - you cannot append characters at the end of the name. You may have as many “tolop”-named operands as you like. This may be useful, for example, when your error function consists of RMS spot size or wavefront error operands for two or more field points. If the “RMS” operands are named “tolop”, the change in the spot size at each field point will be displayed. This allows you to identify the significant contributors to the error function change. Note that “tolop” operands may have zero weight. If this is the case, you can monitor the change in some system measure (e.g., focal length or magnification), without having the change of the system measure affect the error function and compensator calculations.

Often, a system is designed to allow for adjustment of one or more parameters after assembly in an attempt to offset (as much as possible) any fabrication errors. In tolerancing, these adjustable parameters are known as “compensators.” The most common compensator is the back focal distance. The tolerancing routines

will use the currently defined optimization variables as compensators. Thus, before doing any tolerance calculations, make sure that the current variables are the items you wish to use as compensators. During the tolerancing process, the compensators will be varied in an attempt to make the change in the error function caused by the tolerance perturbation as small as possible.

Computing tolerance sensitivities

To compute the tolerance sensitivities using the current error function, select “Sensitivity” from the User-defined tolerancing pull-right menu. (Alternatively, you can use the **tol_sensitivity (tsn)** command.) You will be prompted for the desired tolerance item to be perturbed.



For each item that has a non-zero tolerance value, that item will be changed by the prescribed amount (both plus and minus perturbations) and the error function will be recomputed. If compensators (i.e., variables) are currently defined, several iterations will be performed on the perturbed system to compute the compensator values that minimize the change in the error function. (Note that the compensators are used in an attempt to drive the system back to a state as close to its nominal performance as possible; it is the change in the error function that is minimized. This is different from the optimization procedure where the error function itself is minimized.)

When the computations are complete, a table showing the performance change is displayed. For each tolerance item, the change in the error function for plus and minus perturbations for both uncompensated and compensated (if applicable) cases is displayed. Also, a statistical summary is presented to indicate the overall system performance. (See Chapter 9, “Tolerancing”, of the OSLO Optics Reference manual for a discussion of some statistical aspects of optical system tolerancing.) Displayed are the worst-case system change, and three values (labeled **RSS**, **UNIFORM**, and **GAUSSIAN**) for the standard deviation of the error function change. The three values are calculated for three different distributions of the original system perturbations: end-point (RSS), uniformly distributed, or normally (Gaussian) distributed (truncated at the 2σ point). The input distributions are a function of, for example, the manufacturing process used.

If compensation was used, a statistical summary of the compensators is also presented. The change in the compensators gives you an indication of the range of adjustment that would be required in the system. The RSS value reported is useful as an estimate of the compensator range necessary when overall system performance (i.e., all tolerance perturbations simultaneously) is modeled.

The following example output was generated using the ISO default tolerance for power error fringes with the OSLO demo triplet (demotrip.len). The error function used was a single field point (on-axis) and the result of using the error function

generator with Lobatto pupil sampling and tracing the rays in all three wavelengths. The back focal distance was used as a compensator (i.e., variable).

***TOLERANCE SENSITIVITY ANALYSIS**

ERROR FUNCTION FOR NOMINAL SYSTEM: 0.004514

POWER ERROR TOLERANCE

SRF	TOLERANCE	ERROR FUNCTION CHANGE		COMPENSATED CHANGE	
		PLUS PERT	MINUS PERT	PLUS PERT	MINUS PERT
1	10.0	0.012349	0.005475	0.000115	-9.1541e-05
2	10.0	0.011868	0.004996	0.000134	-0.000105
3	5.0	0.000184	0.006818	-0.000149	0.000202
4	5.0	-0.000157	0.006354	-0.000117	0.000147
5	10.0	0.009305	0.002516	0.000142	-0.000112
6	10.0	0.010228	0.003308	0.000304	-0.000196

STATISTICAL SUMMARY

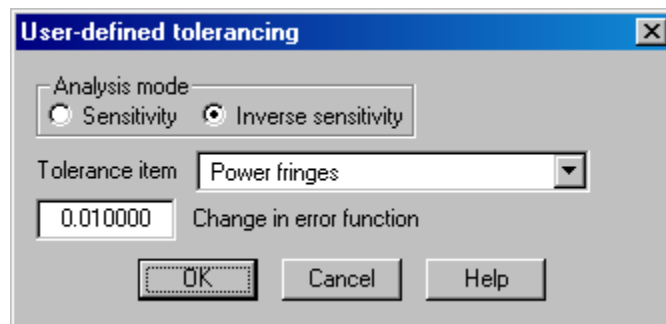
	UNCOMPENSATED	COMPENSATED
WORST CASE CHANGE	0.056922	0.001042
STANDARD DEVIATION		
RSS	0.023904	0.000453
UNIFORM	0.013801	0.000262
GAUSSIAN	0.010513	0.000199

COMPENSATOR STATISTICS

COMP	MEAN	STD DEV	MAX	RSS
TH 6	0.000480	0.165600	0.208868	0.406556

Computing inverse sensitivities

The user-defined tolerancing can also be used in an “inverse” mode. (Choose “Inverse sensitivity” from the User-defined tolerancing menu, or use the `tol_inverse_sens (tis)` command.)

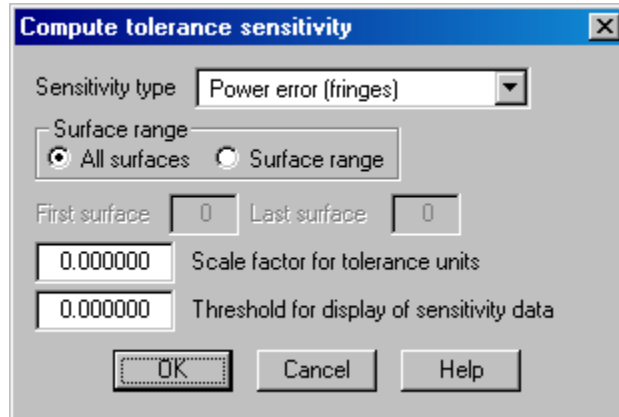


In addition to the perturbation item, you need to enter the desired change in the error function. In this mode, the error function changes are computed as described above. The nominal value and the changes for plus and minus perturbation can be combined to yield a quadratic equation for error function change as a function of the perturbation. This equation is then used to solve for the tolerance value that would result in the input error function change. These computed tolerances are displayed in the output table. Note that this quadratic function is only an approximation, and should be expected to provide accurate inverse sensitivity predictions only for small perturbations. However, it is more generally useful for showing relative sensitivities.

Compute Change Table

Std **Prm**

Change table tolerancing refers to a scheme in which the changes of various performance measures are computed versus changes in the system construction parameters. There are two basic modes in which the analysis is carried out. In the first, known as direct sensitivity, you choose a particular construction parameter grouping (such as “Surface Tilt” or “Power Error”) and the parameter is changed by its defined tolerance. The change in a set of system performance measures is then calculated and reported.



Once a change table tolerance sensitivity is computed, the converse analysis can be carried out and is covered in the section “Distribute Changes” on page 431. This is sometimes called inverse sensitivity, and determines the amount of change in a system parameter that causes a given change in a requested performance measure. In OSLO’s change table tolerancing, the inverse sensitivity analysis (or “Distribute Changes...” option) is based on equal RSS (root-sum-of-squares) contributions, i.e., the amount of change in the system parameter is computed that will cause each to change the performance measure by the same amount.

Tolerance units

The change tables produced by OSLO contain both aberration entries that are specified in terms of *tolerance units* and system quantities that are measured in their respective units. The tolerance units for the aberrations depend on whether the aberration is a transverse measure (T), a longitudinal measure (L), or a wavefront measure (W). Aberrations are measured as a factor times T , L , or W . In the default case, the factor approximately corresponds to the quarter-wave criterion. The default values for T , L , and W are

(10.15)

$$T = \frac{\lambda_0}{|n' u|}$$

(10.16)

$$L = \frac{\lambda_0}{n' u'^2}$$

(10.17)

$$W = 1/4$$

In Eqs. (10.15) and (10.16), λ_0 is the value of wavelength 1 (the reference wavelength), n' is the refractive index in image space for wavelength 1, and u' is the image space slope of the paraxial axial ray in wavelength 1. Also note that Eq. (10.17) is written with the assumption that optical path differences are expressed in wavelengths. For afocal systems, longitudinal aberrations are measured in diopters and the default is

(10.18)

$$L_{afocal} = 1.0$$

Also, aberrations measured transversely for focal systems are expressed in angular measure for afocal systems, as a factor times the angular measure (A), whose default is

(10.19)

$$A = \lambda_0 \left| \frac{\bar{u}'}{H} \right|$$

where \bar{u}' is the image space slope of the paraxial chief ray in wavelength 1 and H is the Lagrange invariant for wavelength 1. The tolerance units may be changed by entering a value in the Scale factor for tolerance units cell in the spreadsheet. The default tolerance units (given by the above equations) are divided by the entered value. This results in the values of the aberrations being multiplied by the entered value. For example, if you want the tolerance units to correspond to one wave of aberration, enter "0.25" for the scale factor.

Format of the change tables

By default, the change tables report all non-zero aberration and system quantity values. If you wish, a threshold for the reporting of performance changes may be set. Only changes that are greater in absolute value than the threshold will be reported. The threshold may be set by entering a value in the Threshold for display of sensitivity data spreadsheet cell.

When a change table is computed, the lens is analyzed in its current configuration. The only solves recognized are thickness solves, which may be used as compensating parameters. Pickups are active. Since curvature solves are deleted prior to the computation of the change table, you may wish to save your lens before performing any change table tolerancing.

The first line of the change table tolerancing output shows the tolerance measures (T , L , and W) upon which the subsequent aberration values are based. The second line shows the threshold value used for the display of the change table entries. The change table contains 18 columns of aberration values divided into three parts of six columns each. The values are listed for the nominal lens (**NOM**)

and the change in the aberration resulting from the specified system parameter change made to each surface individually. Note that the change table entry is the *change* in the aberration from its nominal value. The perturbation value is determined from the currently assigned tolerance data (displayed in the seventh column of the first group of aberrations). The last line of the change tables is a “statistical sum” of the column, defined as

(10.20)

$$RSS = \sqrt{\text{sum of the squares of the change table entries}}$$

The following output shows the change table for spherical error using the OSLO demo triplet (“...\\public\\len\\demo\\edu\\demotrip.len”). The ISO default tolerance values were used. Note that there is a choice of output formats for Change Table Tolerancing. The default output is a three-groups-of-six-quantities format (shown below). If the **compact_chg_tbl_output (ccto)** preference is set to on, the change tables will be displayed in a more compact two-groups-of-nine-quantities format. The computed items and their meanings are identical for both formats.

*CURVATURE SENSITIVITY ANALYSIS

TOLERANCE UNITS

T (Trans.) = 0.004701 L (Long.) = 0.037605 W (Wvfr.) = 0.25
TOLERANCE THRESHOLD = --

SRF	TRANS SPH	AXIAL DMD	MER COMA	FIELD DMD	YFS	XFS	TOL VALUE
NOM	0.00626	1.39	3.58	-2.11	-41.9	-5.61	
1	0.138	-0.0849	0.0306	-0.197	11.6	11.0	10.0
2	0.155	-0.0826	-0.295	-0.235	15.1	11.6	10.0
3	-0.218	0.0505	0.102	0.0252	-5.46	-5.57	5.0
4	-0.16	0.0497	-0.128	0.0201	-5.66	-5.58	5.0
5	0.161	-0.0672	0.0971	0.103	10.1	8.73	10.0
6	0.319	-0.0713	0.0347	0.0903	7.25	7.81	10.0
RSS	0.494	0.169	0.354	0.337	24.1	21.3	

SRF	D BEST FOC	AX RMS OPD	FLD RMS OPD	BACK FOCUS	EFL	TRANS MAG
NOM	-0.479	0.678	6.76	42.95	50.0005	-5.0000e-19
1	0.471	0.013	-0.123	--	0.144	--
2	1.52	0.0171	-0.641	--	0.141	--
3	-0.2	0.0498	-0.0559	--	-0.0893	--
4	-0.416	0.0271	0.00537	--	-0.0898	--
5	0.94	0.0168	-0.248	--	0.163	--
6	-0.154	0.0714	0.146	--	0.167	--
RSS	1.91	0.0952	0.716	--	0.333	--

SRF	% DIST	% TRANS DIST	LAT SHEAR	CENT COMA	YFS FIELD	XFS FIELD
NOM	0.557	--	--	--	--	--
1	-0.0065	--	--	--	--	--
2	-0.018	--	--	--	--	--
3	-0.00547	--	--	--	--	--
4	-0.00628	--	--	--	--	--
5	0.0345	--	--	--	--	--
6	0.0195	--	--	--	--	--
RSS	0.0448	--	--	--	--	--

Change table entry definitions

The definitions and units for the entries in the change table are given in the table below. The ray based aberrations refer to rays that are numbered as follows:

Ray No.	FBY	FBX	Ypupil	Xpupil
1	0.0	0.0	1.0	0.0
2	0.0	0.0	-1.0	0.0
3	0.0	0.0	0.0	0.0
4	1.0	0.0	1.0	0.0
5	1.0	0.0	-1.0	0.0
6	1.0	0.0	0.0	0.0
7	0.0	1.0	0.0	0.0
8	-1.0	0.0	0.0	0.0

In the above table, **Ypupil** and **Xpupil** are fractional coordinates relative to the real, vignetted entrance pupil for each field point. Thus, for example, a value of $Y_{\text{pupil}} = 1$ corresponds to the ray with the largest value of FY that is transmitted through the lens at that field point.

In defining the tolerance aberrations, the following notation is used.

Item	Description
$Y(i)$	Meridional ray height of ray i at the image plane
$OPD(i)$	Optical path difference of ray i at the image plane
$DMD(i)$	Conrady $D-d$ chromatic path difference of ray i at the image plane
$YFS(i)$	Meridional field sag of ray i at the image plane
$XFS(i)$	Sagittal field sag of ray i at the image plane

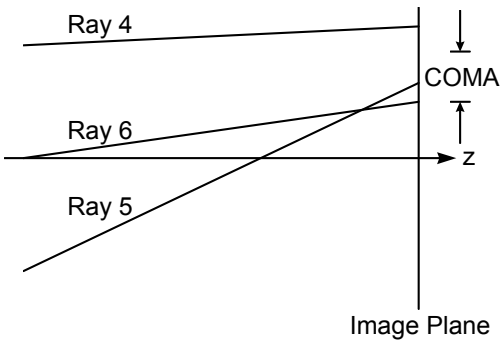
Entries shown in parenthesis (i.e. “TR SPH”) are the headings when the compact form of the output is used.

Entry	Units	Description
<i>TRANS SPH</i> (<i>TR SPH</i>)	4 <i>T</i>	Transverse spherical aberration $TRANS\ SPH = \frac{0.5[Y(1) - Y(2)]}{4T}$ Note: For this calculation, Y(1) and Y(2) are computed in the <i>paraxial</i> image plane, not at the nominal image location.
<i>AXIAL DMD</i> (<i>AX DMD</i>)	2 <i>W</i>	Axial chromatic aberration $AXIAL\ DMD = \frac{0.5[DMD(1) + DMD(2)] - DMD(3)}{2W}$
<i>MER COMA</i> (<i>COMA</i>)	1.8 <i>T</i>	Meridional coma at full field (see figure below) $MER\ COMA = \frac{0.5[Y(4) + Y(5)] - Y(6)}{1.8T}$
<i>FIELD DMD</i> (<i>FLD DMD</i>)	2 <i>W</i>	Lateral chromatic aberration at full field $FIELD\ DMD = \frac{DMD(5) - DMD(4)}{2W}$
<i>YFS</i> (<i>YFS</i>)	0.5 <i>L</i>	Tangential focal shift at full field $YFS = \frac{YFS(6)}{0.5L}$
<i>XFS</i> (<i>XFS</i>)	0.5 <i>L</i>	Sagittal focal shift at full field $XFS = \frac{XFS(6)}{0.5L}$
<i>D BEST FOC</i> (<i>DEL BF</i>)	0.5 <i>L</i>	Difference in best focus for <i>AX RMS OPD</i> and <i>FLD RMS OPD</i> $D\ BEST\ FOC = \frac{\Delta z}{0.5L}$

<i>AX RMS OPD</i> (<i>AX OPD</i>)	4 <i>W</i> /14	Root-mean-square OPD on axis at best focus $AX \text{ RMS OPD} = \frac{\text{axial rms OPD at best focus}}{4W/14}$
<i>FLD RMS OPD</i> (<i>FLD OPD</i>)	4 <i>W</i> /14	Root-mean-square OPD at full field at best focus $FLD \text{ RMS OPD} = \frac{\text{full field rms OPD at best focus}}{4W/14}$
<i>BACK FOCUS</i> (<i>BK FOC</i>)	--	The sum of the final two thicknesses.
<i>EFL</i> (<i>EFL</i>)	--	Effective focal length (determined from a paraxial ray trace).
<i>TRANS MAG</i> (<i>TR MAG</i>)	--	Paraxial transverse magnification $TRANS \text{ MAG} = \frac{nu}{n'u'}$
% <i>DIST</i> (% <i>DIST</i>)	%	Per cent distortion in the meridional plane at full field $\% \text{ DIST} = \frac{100[Y(6) - \bar{y}']}{\bar{y}'}$
% <i>TRANS DIST</i> (% <i>T DIST</i>)	%	Per cent transverse distortion (distortion of a line on the <i>x</i> -axis in the object plane) $\% \text{ TRANS DIST} = \frac{100[Y(7) - Y(3)]}{\bar{y}'}$
<i>LAT SHEAR</i> (<i>LAT SH</i>)	--	Lateral shear $LAT \text{ SHEAR} = Y(3)$
<i>CENT COMA</i> (<i>C COMA</i>)	1.8 <i>T</i>	Central (axial) coma $CENT \text{ COMA} = \frac{0.5[Y(1) + Y(2)] - Y(3)}{1.8T}$

<i>YFS FIELD</i> (<i>YFS FLD</i>)	0.5 <i>L</i>	Tangential focal shift in the field $YFS\ FIELD = \frac{YFS(6) - YFS(8)}{0.5L}$
<i>XFS FIELD</i> (<i>XFS FLD</i>)	0.5 <i>L</i>	Sagittal focal shift in the field $YFS\ FIELD = \frac{YFS(6) - YFS(8)}{0.5L}$

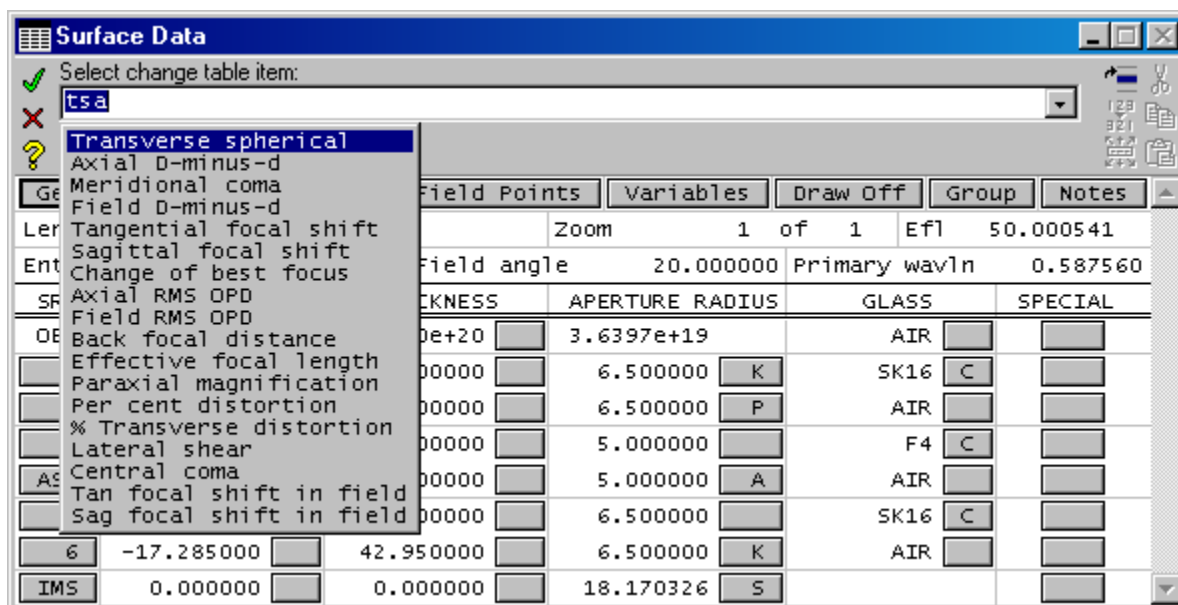
BACK FOCUS, *EFL*, and *LAT SHEAR* are displayed in lens units; *TRANS MAG* is dimensionless; *%DIST* and *%TRANS DIST* are expressed as a percentage. *YFS FIELD* and *XFS FIELD* measure the tip in the tangential and sagittal focus, respectively. Note that for systems and perturbations that maintain axial symmetry (i.e., rotational symmetry about the optical axis), the aberrations *%TRANS DIST*, *LAT SHEAR*, *CENT COMA*, *YFS FIELD*, and *XFS FIELD* are zero. Also note that the change table entries based on a paraxial ray trace may not be valid for systems containing special data that invalidates the paraxial ray trace.



Distribute Changes

Std

Prm



After computing a change table, the tolerance values that result in equal contributions to the RSS for a specified aberration may be computed by selecting the “Distribute Changes...” option. The desired change table entry is selected from the pop-up list. The details of the change table items are described in the previous section, “Change table entry definitions” on page 427. A linear approximation is used to compute the values of the tolerances that would yield equal contributions to the (same) RSS value. For this option, only the computed allowed tolerance values are displayed; the change table itself is not reprinted. Note that the values of the lens tolerance data are not changed when an equal RSS contribution budget is computed. Of course, in general, the tolerance aberrations will not be linear functions of the perturbation, but the equal RSS calculation is useful in showing relative sensitivities.

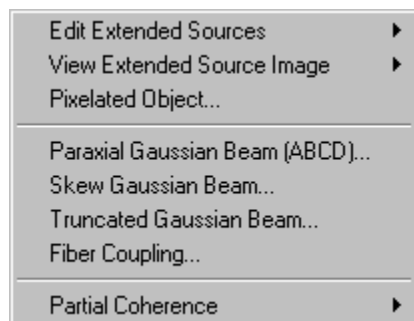
*EQUAL RSS CONTRIBUTION TOLERANCES - TRANSVERSE SPHERICAL

POWER ERROR TOLERANCE (FRINGES)

SRF ALLOWED TOLERANCE

1	14.6
2	13.0
3	-4.62
4	-6.29
5	12.5
6	6.32

Overview



The analysis tools that define or analyze the current system using special sources, such as extended or Gaussian beam sources are gathered under the Source menu.

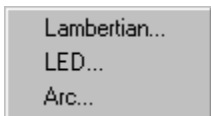
The analyses performed here is based on the current OSLO model which has been defined up to this point. However in general, the source attributes that are defined or applied from this menu do not relate to analyses in other sections of the program. Example: The Fiber Coupling analysis is affected by the Gaussian beam apodization set in the Paraxial Setup Editor (see section “Gaussian pupil apodization specification (sdaz)” on page 123). However the assignment of a particular object shape in the Pixelated Object analysis here, is ignored when performing other analyses such as Point Spread Function and Wavefront analyses, to name a few. In many ways, the source analyses are self-contained: You will choose how sources are implemented in close combination with the analyses you want performed on them.

The menu commands relating to sources are:

Menu Item	Page
Edit Extended Sources... Defines either a Lambertian, Light Emitting Diode (LED) or Arc type source to be placed at the image plane. Used in conjunction with “View Extended Source Image” below.	434
View Extended Source Image... Launches rays from a randomly sampled predefined volume. Rays are aimed randomly (within a predefined pattern) into space. Allows a relative illumination analysis on the image surface.	443
Pixelated Object... Launches rays from a randomly sampled predefined extended object (a finite number of point sources). Rays are aimed randomly within the entrance pupil of the system. Results are plotted in a graphic window	446
Paraxial Gaussian Beam (ABCD)... Performs an ABCD (paraxial) Gaussian Beam propagation through the system using an interactive spreadsheet.	448
Skew Gaussian Beam... Traces a (possibly astigmatic) Gaussian beam through the system using real ray tracing, and prints information about the location of the beam on each surface	451
Truncated Gaussian Beam... Plots X and Y cross scans of the Point Spread Function. The analysis includes the effects of vignetting apertures and Gaussian apodization of the pupil, if defined.	454

- Fiber Coupling...** Traces a grid of rays (partially defined by current spot diagram aperture division and a Gaussian apodization factor) to the system exit pupil, transforms the energy to the image surface and computes the coupling efficiency into a user defined optical fiber. 455
- Partial Coherence...** Computes the partially coherent or incoherent image for the current lens given a one dimensional source frequency pattern. 457

Edit Extended Sources



The purpose of this feature is to analyze illumination patterns given a particular source type placed at the object. In order to achieve this, each source has to be completely defined, in order to take into account not only its shape and position (similar to “Pixelated Object” on page 446), but also its angular ray distribution in space.

The “Edit Extended Sources” menu item allows you to edit the how the different source options are defined and saves the results of the editing process to a source database. The source database for each source type is stored with the installation of OSLO in the “...\private\cdb” directory.

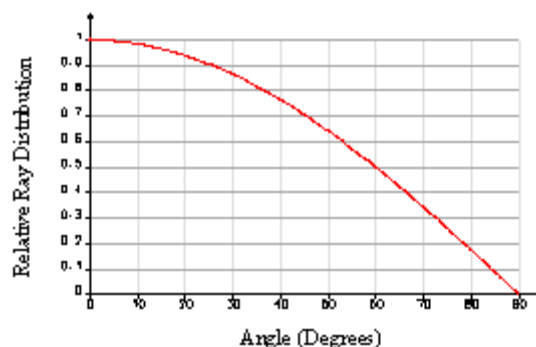
The “Edit Extended Sources” menu item is meant to be used as a precursor to the “View Extended Source Image” menu item described in the following section on page 443.

Currently, three predefined source types are available: Lambertian, LED and Arc sources. The procedure to define your own source types with different shapes and parameters is explained later in this section on page 439.

Lambertian

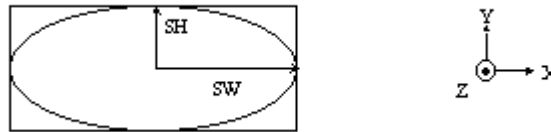
A Lambertian source is characterized by a cosine intensity as a function of the angle of the ray with respect to the Z-axis.

$$I = I_o \times \cos(\theta) \quad (11.1)$$



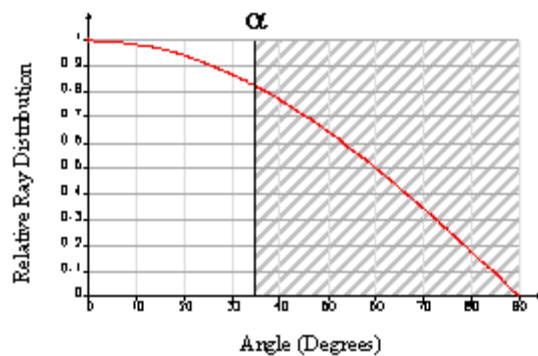
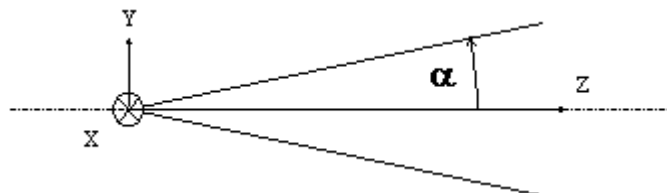
Spatial definition (X, Y, Z definition for rays):

A Lambertian source is considered a flat surface in the XY plane ($Z = 0$). The shape of the source in the XY plane can be defined as an ellipse or rectangle. The size is defined by the Semi Height (SH) along the local Y axis and the Semi Width (SW) along the local X axis. This is described schematically below:



Angular Distribution definition (K, L, M definition for rays):

All the rays are launched within the solid angle defined by the Z-axis and the limiting angle or Divergence Half Angle (α).



The data that needs to be entered to define a Lambertian source is below. The data is presented in a spreadsheet that works like similar spreadsheets in OSLO. You can edit existing cells of the spreadsheet (existing source parameters) and you can add new sources by inserting lines in the spreadsheet. To insert a line, right click on the number in the "Menu" column and select "Insert Before" or "Insert After" from the pop-pup menu. To delete a line, make sure to select the entire line, then right click on the "Menu" number and select "Delete". In order to identify each source, you must give it a unique source name.

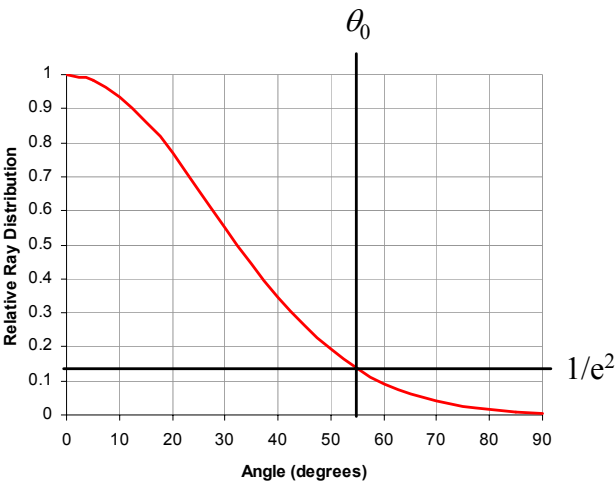
Parameter	Type	Values
Name	String	(up to 16 characters)
Shape	String	ELLIPSE, RECTANGLE
Semi Height (SH)	Double	$0 < SH < \infty$
Semi Width (SW)	Double	$0 < SW < \infty$
Divergence Half Angle (Degrees) = α	Double	$0 < \alpha < +90$

LED

An LED (Light Emitting Diode) is characterized by a Gaussian intensity as a function of the angle of the ray with respect to the Z-axis.

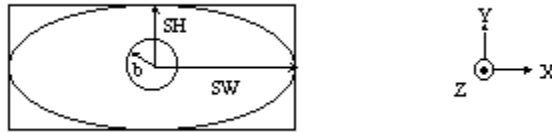
$$I = I_0 \times e^{-2\left(\frac{\theta}{\theta_0}\right)^2}$$

(11.2)



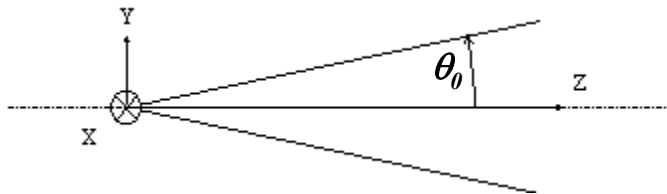
Spatial definition (X, Y, Z definition for rays):

An LED source is considered a flat surface in the XY plane ($Z = 0$). The shape of the source in the XY plane can be defined as an ellipse or rectangle. The size is defined by the Semi Height (SH) along the local Y axis and the Semi Width (SW) along the local X axis. The LED may also have a central obscuration due to a bond wire bonding pad (semi diameter “b”). This is described schematically below:



Angular Distribution definition (K, L, M definition for rays):

All the rays are launched within the solid angle defined by the Z-axis and the $1/e^2$ Divergence Half Angle (θ_0). A limiting angle does not need to be defined.



The data that needs to be entered to define an LED source is below. The data is presented in a spreadsheet that works like similar spreadsheets in OSLO. You can edit existing cells of the spreadsheet (existing source parameters) and you can add new sources by inserting lines in the spreadsheet. To insert a line, right click on the number in the "Menu" column and select "Insert Before" or "Insert After" from the pop-up menu. To delete a line, make sure to select the entire line, then right click on the "Menu" number and select "Delete". In order to identify each source, you must give it a unique source name.

Parameter	Type	Values
Name	String	(up to 16 characters)
Shape	String	ELLIPSE, RECTANGLE
Semi Height (SH)	Double	$0 < SH < \infty$
Semi Width (SW)	Double	$0 < SW < \infty$
Bond Wire Semi diameter (b)	Double	$0 < b < \infty$
$1/e^2$ Half Angle (Degrees) = θ_0	Double	$0 < \theta_0 < +90$

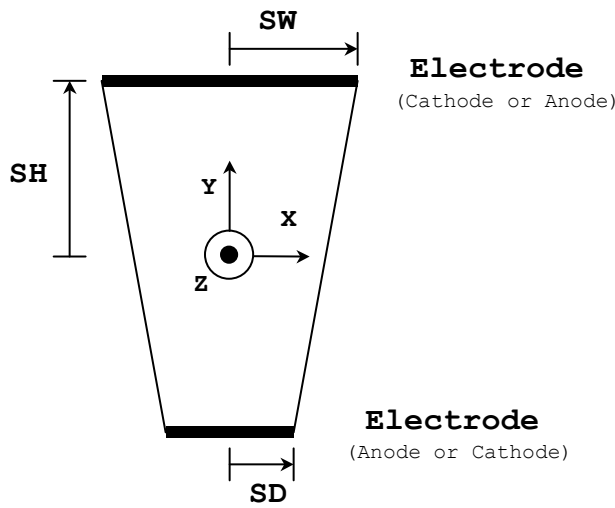
Arc

An arc lamp is made of 2 circular plate electrodes, with a gas between the anode and the cathode.

Shape definition (X, Y, Z definition for rays):

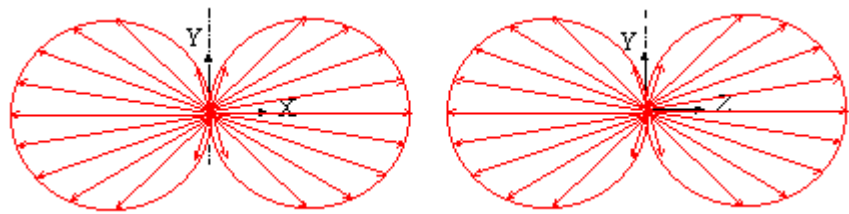
The rays are launched from the tapered cylinder volume defined by the Semi Width (SW - the radius of the electrode on the top), and the Semi Depth (SD - the

radius of the electrode on the bottom). The length of the tapered cylinder is defined by the Semi Height (SH) between the electrodes.



Angular Distribution definition (K, L, M definition for rays):

Angular dispersion is modeled by a Toroidal shape symmetrical about the Y-axis:



The data that needs to be entered to define an Arc source is below. The data is presented in a spreadsheet that works like similar spreadsheets in OSLO. You can edit existing cells of the spreadsheet (existing source parameters) and you can add new sources by inserting lines in the spreadsheet. To insert a line, right click on the number in the “Menu” column and select “Insert Before” or “Insert After” from the pop-up menu. To delete a line, make sure to select the entire line, then right click on the “Menu” number and select “Delete”. In order to identify each source, you must give it a unique source name.

Parameter	Type	Values
Name	String	(up to 16 characters)
Semi Width (SW)	Double	$0 < SW < \infty$
Semi Depth (SD)	Double	$0 < SD < \infty$
Semi Height (SH)	Double	$0 < SH < \infty$

Creating User-Defined Source Types (Advanced)

With OSLO's CCL programming language, you can create a custom source type with custom parameters. Within that source type, you will then be able to define different sources with varying parameter values and store them in a source database just like the existing Lambertian, LED and Arc source types. A step-by-step description of how to create a custom source type follows.

Special Notes (before we begin):

- You will be doing some programming by altering the contents of existing CCL files. It is recommended that you use your text editor's "Find..." feature to locate the described locations easily.
- Some of the files listed below are in the Public CCL folder. All files in the Public CCL folder will be marked as "Read-Only" by default. You will have to change the properties of these files (using Windows Explorer or similar) before you are able to edit these files.
- If you edit commands within files in the Public CCL folder, make sure that you do not already have a copy of those commands in the Private CCL folder. Commands in the Private CCL folder with the same name will take precedence over files in the Public CCL folder.
- The line numbers presented here will not be exact due to (a) slight revisions in files between OSLO releases, and (b) the alterations you will be making in the files themselves. The order of the steps outlined below will instruct you to change the end of the CCL files before the beginning - this should help the line numbers stay consistent as you work through the different files.
- Changes to files in the Public CCL may be overwritten when future OSLO releases are installed. It is advised that you backup your work into files without standard OSLO CCL file names (i.e. files names with *.cclbak delimiters).

Detailed Steps:

First, you need to create a new database file. We will assume that the name of your file is "src_mysource.cdb". This is created in your "...private\cdb" directory. This database will hold all the necessary parameters for the source type you want to describe, such as angles, dimensions, shape flags etc... To make it easier, save an existing sources database (such as "src_LED.cdb") as "src_mysource.cdb", and add new fields or delete irrelevant fields as needed.

Next, you need to make changes to the code. All the source code for the existing sources is available to you. While creating your own type of source is easy, it requires changes in several places. It is highly recommended to make backup copies of the files you modify (by saving them with a *.HX or *.CCX extension, for example).

In some places in the source code where you need to alter the code, the place is marked by the following flag:

```
// CUSTOMIZATION
```

Step #1

In File: ...Public>>CCL>>inc>>listdefs.h

Approximate Line: 7

Action:

Define a new source type by defining SOURCE_TYPE_MYSOURCE.

```
#define LST_SRC_MYSOURCE      4
```

This will attribute an identification number ("4" in this case) to your source type. This number has to be different from the numbers given to existing source types.

Step #2

In File: ...Private>>CCL>>a_menu.ccl

Approximate Line: 525

Action:

Add a new item to the view sources list (view_src_cmds):

```
"My Source..." = "arg_entry=command_line;source_launch(My Source)",
```

This will create a new item on the "Source>>View Extended Source Image" pop up menu.

Step #3

In File: ...Private>>CCL>>a_menu.ccl

Approximate Line: 518

Action:

Add a new item to the edit sources list (edit_src_cmds):

```
"My Source..." = "arg_entry=command_line;source_edit(My Source)",
```

This will create a new item on the "Source>>Edit Extended Sources" pop up menu.

Step #4

In File: ...Public>>CCL>>a_list.ccl

Approximate Line: 17

Action:

Add a new source type to the source_type_list:

```
"My Source" = LST_SRC_MYSOURCE
```

This list is used internally and also for the pop-up list when selecting the appropriate icon from the source toolbar. This utilizes the LST_SRC_MYSOURCE you defined as a unique number in step #1 of this table.

Step #5

In File: ...Public>>CCL>>eval_source.ccl

Approximate Line: 599

Action:

Add a new item to the "switch (source_choice)" statement in the "source_edit()" command:

```
case LST_SRC_MYSOURCE:
    open cdb private "." MY_SOURCE_DB;
    break;
```

This will locate the source database and open it when "Edit Extended Sources" is selected.

Step #6

In File: ...Public>>CCL>>eval_source.ccl

Approximate Line: 596

Action:

Add a new item to the "switch (source_choice)" statement in the "source_launch()" command:

```
case LST_SRC_MYSOURCE:
    db_arg_parameters(MY_SOURCE_DB, 1);
    break;
```

This creates an argument list from the parameters of the previously created database.

Step #7

In File: ...Public>>CCL>>eval_source.ccl

Approximate Line: 427

Action:

This is where the source parameters are plotted as data in the **right** column of the output in the graphic window of the "source()" command.

```
case LST_SRC_MYSOURCE:
    label (.....);
    break;
```

Step #8

In File: ...Public>>CCL>>eval_source.ccl

Approximate Line: 398

Action:

This is where the source parameters are plotted as labels in the **left** column of the output in the graphic window of the "source()" command.

```
case LST_SRC_MYSOURCE:
    label (.....);
    break;
```

Step #9

In File: ...Public>>CCL>>eval_source.ccl

Approximate Line: 249

Action:

This is where the random rays are generated in the "source()" command. Each ray is defined by 7 parameters:

xo, yo, zo: are the coordinates of the ray relative to the object surface. They represent a surface ($z=0$) or a volume from which each ray is launched.

ko, lo, mo: values of the cosines of the ray direction with respect to the X-axis, Y-axis and Z-axis.

pwr: weight given to a specific ray.

There are 2 ways to define an apodization (spatial or angular). You can either:

Modify the ray density, giving a power of 1 to each ray. For spatial apodization, more rays will come from a specific area -volume or surface. For angular apodization, more rays will be launched in a specific direction. This is how the lambertian and LED sources are defined.

Rays are launched equally into space (or from a specific area in space), and a variable power is assigned to each ray. This is how the arc source models the toroidal apodization.

The latter tends to be easier to program but slower. Of course, you can combine both techniques.

```
case LST_SRC_MYSOURCE:
    .....
    break;
```

Step #10

In File: ...Public>>CCL>>eval_source.ccl

Approximate Line: 137

Action:

This is where your source database file is parsed within the "source()" command. The data is stored into your pre-defined variables, which you will define in step #11. The value of each parameter is accessed using the db_getXelement commands, where X stands for i if the element is an integer, s for a string, r for a real.

```
case LST_SRC_MYSOURCE:
    db_get_relement(&.....
    .....
    break;
```


Step #11**In File:** ...Public>>CCL>>eval_source.ccl**Approximate Line:** 75**Action:**

Define variables corresponding with the parameters of your source type in the "source()" command. When your database file is read, the parameters are stored in these variables to be used later.

```
double src_parameter;                // helpful comments
.....
```

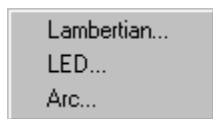
Step #12**In File:** ...Public>>CCL>>eval_source.ccl**Approximate Line:** 21**Action:**

Define the constant MY_SOURCE_DB to store the name of your database:

```
#define MY_SOURCE_DB  "src_mysource.cdb"
```

Once you have completed these steps, remember to recompile both the public and private CCL directories.

View Extended Source Image



Now that you have defined the characteristics of your source in the "Edit Extended Source" menu item (page 434), these menu options allow you to choose some final analysis parameters and then the analysis is performed. Options include choosing how many rays to draw, positioning the source relative to your current system (both tilt and displacement) as well as controlling the size and placement of the image analysis area. All this is performed using this menu item before the final analysis is presented.

Lambertian / LED / Arc

The analysis parameters you enter here and the resulting analysis is actually the same for each of these three menu items. The only difference between the three

menu options is that you will only be allowed to pick from a list of sources of the chosen type.

Input

Source name

You choose from a list of source names specific to the type of source analysis you have chosen (Lambertian, LED, Arc). The individual parameters of these sources can be edited using the “Edit Extended Source” menu item (page 434).

Wavelength number

The number of the wavelength to be used in the analysis. The list of available wavelengths for your system can be chosen in the “Wavelength Data Editor” (page 124).

Number of rays to launch

The number of total rays to be launched for the analysis. Note that if the angular dispersion of the source defined in the “Edit Extended Source” menu item (page 434) is too large, some of your rays may fall outside the entrance pupil of your optical system.

Source x offset / y offset / tla / tlb/ tlc

The entered offsets (x & y) are measured relative to the local vertex of the object surface. The rotation angles TLA, TLB and TLC rotate the source using the same sign convention as for surface tilts (p 66). Labels on the final plot will inform you of the exact position and orientation of the source in system units. Note that the current reference ray definition does not affect the source or image locations in this analysis.

Image x offset / y offset

The entered offsets (x & y) are measured relative to the local vertex of the image surface. They adjust the center point of the resulting analysis. The orientation of the analysis region (image patch) is defined by the image surface itself.

Image patch size

This defines the size or scale of the square image patch to be displayed in the resulting analysis graph. The value entered will be the total width of the square analysis region. A value of zero allows OSLO to choose an appropriate scale.

Smoothing

When smoothing is chosen, the energy produced by a ray landing on a particular analysis pixel is shared by the surrounding pixels according to the algorithm shown. The numbers refer to the energy on each pixel relative to the energy on the center pixel (the pixel where the ray hit).

0.25	0.5	0.25
0.5	1.0	0.5
0.25	0.5	0.25

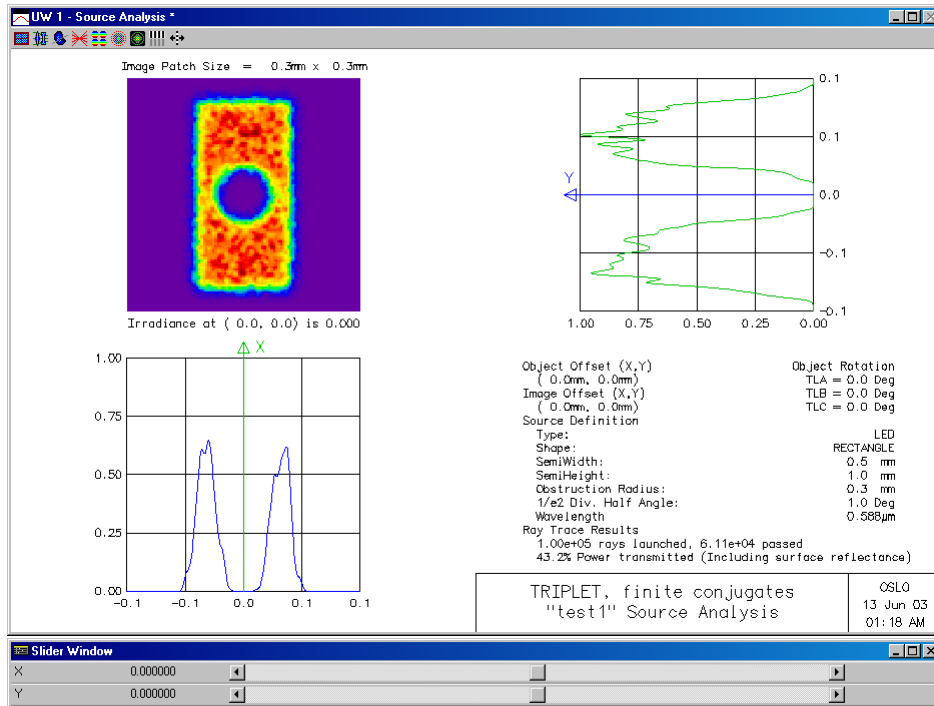
Analysis Results

While the analysis is being calculated, the graphic window shows a basic plot of the image, plotting a square symbol for every 1,000th ray. A running count of how many rays have been traced during the analysis is displayed in the message area under the command line. You can interrupt this analysis at anytime during the computation by hitting the Escape key.

In the top left of the resulting graphic is a 2D view of the irradiance map on the image surface. The map is a projection of the image surface irradiance onto a plane that is in the orientation of the local image surface coordinate system (+y direction is at the top and the +x direction is to the right). The image patch size dimensions presented above the 2D map are full width and full height dimensions. The irradiance value of a particular point of the image patch is presented at the bottom of the image, and the location of this irradiance value is adjusted by using the sliders.

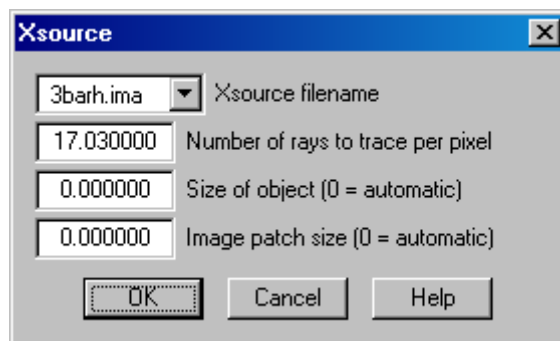
The bottom left and top right corners of the resulting graphic show x and y cross section slices of the irradiance map. The positioning of the cross section slices can be adjusted using the sliders. The sliders are initially located beneath the analysis window, although you can move the slider window and re-size it using the mouse. The position of the x slice is shown in the y cross section graph and the position of the y slice is shown in the x cross section graph.

The bottom right corner of the resulting graphic reports the input parameters as well as the results of ray and power analysis. The total number of rays launched and total number of rays collected on the image patch are reported. The power collected on the image patch is also reported. If the polarization ray trace option is turned on in the General Conditions editor (page 117), then the power reported includes coating losses and polarization effects. *Note that the polarization raytrace option is only available in OSLO Premium.*



Pixelated Object

This analysis option allows you to evaluate the approximate imaging quality of a system. The source option in this analysis is made of a grid of pixels. A pixel is a small square region on the object surface. A bundle of random rays is launched from each of the pixels through the pupil of the system; the number of rays is proportional to the pixel weight. The end result is similar to a spot diagram, but differs in that the object point is different for each of the rays traced. As more rays are traced through the system, the resulting distribution of ray coordinates on the image surface more closely approximates a continuous extended source. Only geometrical effects are accounted for; diffraction is not considered.



Input Data

Xsource filename

You can select which object to image through the system by selecting from a list of existing files in the "XSource filename" field. You can make your own *.ima files using the OSLO text editor to modify one of the supplied files. The *.ima files are simple ASCII text files that can be edited with any text editor including the internal OSLO editor (page 481). If you open one of these files in the "...private\bin\ima" directory, you will see that the first line of each file gives the number of pixels (n vertical and n horizontal) in the object. Each succeeding line represents a sequence of n pixels, 0 representing off, and a number 1 through 9 representing the pixel weight (the included files only use pixel weights of 1). You can edit an existing file or add a new file (just make sure that the file name has an *.ima extension and resides in the "...private\bin\ima" directory).

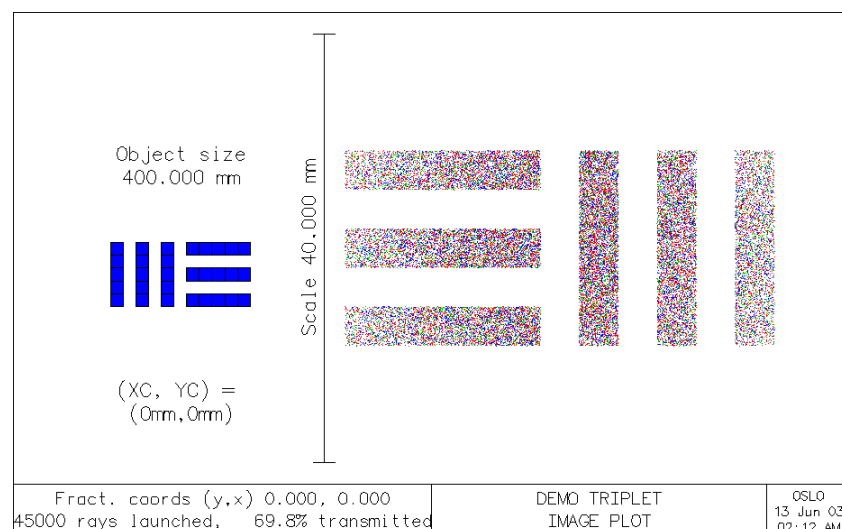
Object scaling

The *.ima object files define a series of pixels, and therefore the object must be scaled to a real dimension using the "Size of Object" field. The entered value is the dimension of the side of the square. The units used are system units if the object is at a finite distance, degrees otherwise. If automatic mode is selected (by entering "0"), the assumed size for the side of the square is twice the Object Height (Field Angle if the object is at infinity) defined for the system.

Image scaling

The image scale (image patch) is defined in degrees in afocal evaluation mode, in system units otherwise. If automatic mode is selected (by entering "0"), the image patch size will appropriately match the object size entered in the previous field.

Analysis Results



A schematic of the object is shown on the left side of the resulting analysis, while the image of the object is shown on the right. Since all the rays are launched towards the system pupil, 100% of the rays should pass to the image if there is no vignetting. However in systems where there is significant vignetting near the edge

of the field, you will notice a reduction in point density at the corners and edges of images. Rays will be traced at each of the system wavelengths with a different color pen. The density of points in each pen color corresponds to the weight of that wavelength in the Wavelength Data Editor (page 124). The pen colors correspond to the order of the wavelengths in the Wavelength Data Editor and do not necessarily correspond to the visual colors of the wavelengths.

Object/Image Positioning

The object is centered on the current Object Point location. This location is reported in the bottom left corner of the analysis graphic ("Fract. coords (y,x) ..."). The values refer to the fraction of the field of view defined for your system. You can change this position by performing a new reference (Chief) ray trace (page 148) prior to running this analysis. The image patch is centered on the location of the Chief ray on the image surface. Its coordinates on the image surface are given by the '(XC,YC)' label.

Paraxial Gaussian Beam (ABCD)

This menu option opens an interactive spreadsheet that allows you to specify the parameters of a Gaussian beam relative to any surface in the lens and then immediately see the beam parameters on any other surface in the lens. All of the Gaussian beam analysis is based on paraxial optics using ABCD matrices, and it is assumed that the beam is relatively "slow", and that diffraction from apertures can be neglected. Its use should be limited to the analysis of on-axis, orthogonal optical systems containing only planar, spherical, or cylindrical refracting or reflecting surfaces.

There are four parameters that specify a Gaussian beam:

- **Spot size (w):** The radius at which the beam irradiance is $1/e^2$ of its axial value
- **Waist spot size (w_0):** The minimum spot radius of the beam
- **Waist distance (z):** The distance from the observation plane to the beam waist
- **Wavefront radius of curvature at the observation plane (R)**

Any two of these quantities may be specified (becoming the independent variables) and the other two will be calculated (becoming the dependent variables) using the normal Gaussian beam equations:

$$w^2(z) = w_0^2 \left[1 + \left(\frac{\lambda z}{\pi w_0^2} \right)^2 \right] \quad (11.3)$$

$$R(z) = z \left[1 + \left(\frac{\pi w_0^2}{\lambda z} \right)^2 \right] \quad (11.4)$$

In the above equations, λ is the wavelength of the beam.

Beam Specification Surface: <input type="text" value="0"/>			Beam Evaluation Surface: <input type="text" value="7"/>		
	Solution I	Solution II		Solution I	Solution II
Spot size (w) *	<input type="text" value="0.050000"/>	0.050000	Spot size (w)	0.933227	0.933483
Waist ss (w0) *	<input type="text" value="0.010000"/>	0.010000	Waist ss (w0)	0.012757	0.014725
Waist dist (z)	<input type="text" value="2.619409"/>	-2.619409	Waist dist (z)	-63.650461	-73.487549
Wvf radius (R)	<input type="text" value="2.728551"/>	-2.728551	Wvf radius (R)	-63.662357	-73.505840
Diverg. (rad)	0.018700	0.018700	Diverg. (rad)	0.014659	0.012700
Rayleigh range	0.534685	0.534685	Rayleigh range	0.870186	1.159378
Wavelength number of beam	<input type="text" value="1"/>		Evaluation surface shift	<input type="text" value="0.000000"/>	
Wavelength	0.587560		Beam meridian:	<input checked="" type="radio"/> y-z <input type="radio"/> x-z	
M-squared	<input type="text" value="1.000000"/>		<input type="button" value="Print beam data in text window"/>		
<input type="button" value="Plot beam spot size"/>			<input checked="" type="radio"/> slider-wheel design <input type="radio"/> Current graphics window		

The beam may be specified by selecting the appropriate cell and typing the desired value. The independent variables are indicated by an asterisk (*) next to the cell in the spreadsheet. The reference plane may be the tangent plane to any surface in the lens. Thus, if the waist is at the beam specification surface, z is 0. The sign convention for z and R is the same as for specifying thickness and radius of curvature (note that this is the opposite of the sign convention generally used in laser theory). Thus, if the beam waist occurs before the beam reaches the beam specification surface, z is negative. For some combinations of independent variables, there are two possible beams that are described. If this is the case, both solutions will be shown, in the columns labeled "Solution I" and "Solution II".

There are two other pieces of data for the beam that are displayed for information purposes in the spreadsheet:

- **Divergence** is the far field (half) angle (in radians) of beam divergence, measured in the far-field of the beam, relative to the position of the waist. The far-field angle θ is given by:

$$\theta = \tan^{-1}\left(\frac{\lambda}{\pi w_0}\right) \quad (11.5)$$

- **Rayleigh range** is the distance from the beam waist to the minimum wavefront radius of curvature of the beam. At a distance from the beam waist equal to the Rayleigh range, the spot size has increased by a factor of $\sqrt{2}$ from the waist size, and the magnitude of the wavefront radius of curvature of the beam is equal to twice the Rayleigh range value. The Rayleigh range is sometimes used as a definition of the collimation of a Gaussian beam. The definition of the Rayleigh range z_R is:

$$z_R = \frac{\pi w_0^2}{\lambda} \quad (11.6)$$

The wavelength of the beam can be any one of the defined wavelengths of the lens. See the section "Wavelength" on page 124 for a discussion of changing the wavelength values. The wavelength is specified by selecting the "Wavelength number of beam" cell and typing in the number of the desired wavelength. The value of the wavelength (in μm) will be displayed in the spreadsheet.

If you want to model a non-ideal beam, a value of "M-squared" (the so-called beam-quality factor) may be entered in the *M*-squared cell. An ideal Gaussian beam has an *M*-squared value of 1.

Each time you change the specification of the beam, the beam parameters on the evaluation surface are recomputed and displayed in the spreadsheet. The reference plane is the tangent plane to the vertex of the beam evaluation surface. The beam parameters at other locations in the space of the evaluation surface may be easily found by typing in a value in the Evaluation surface shift cell. This value is the distance from the evaluation surface to the observation plane.

The Gaussian beam may be traced in either the *y*-*z* or the *x*-*z* meridian. The desired orientation is chosen by clicking the appropriate radio button. You can trace an astigmatic beam by entering different beam parameters for the *y*-*z* and *x*-*z* meridians. The program "remembers" the beam specification in each meridian, so you can switch back and forth without having to reenter your data.

Printing the Gaussian beam data

If you want to print a surface-by-surface listing of the beam parameters into the text window, click the "Print beam data in text window" button. For each surface, seven pieces of data are displayed:

- **SPOT SIZE** – the spot size at the surface
- **DIVERGENCE** – the divergence angle (in radians) after refraction/reflection
- **WAIST SIZE** – the beam waist size after refraction/reflection
- **WAIST DIST** – the distance from the surface to the beam waist after refraction/reflection
- **INC RADIUS** – the incident wavefront radius of curvature
- **RFR RADIUS** – the refracted/reflected wavefront radius of curvature
- **RAYLEIGH RG** – the Rayleigh range of the beam after refraction/reflection

*GAUSSIAN BEAM - YZ PLANE

WAVELENGTH =		0.587560	M-SQUARED =		1.000000		
SRF	SPOT SIZE	DIVERGENCE	WAIST SIZE	WAIST DIST	INC RADIUS	RFR RADIUS	RAYLEIGH RG
INC	1.000000	0.000187	1.000000	--	--	--	5.3468e+03
1	1.000000	0.018016	0.006406	55.499263	--	55.501540	0.355525
2	0.963965	0.032954	0.005673	29.240740	53.501625	29.241753	0.172096
3	0.766174	0.005962	0.019403	128.457784	23.242014	128.540223	3.254220
4	0.760213	0.014652	0.012763	-51.872293	127.540869	-51.886919	0.871014
5	0.848122	0.006744	0.017113	-125.725494	-57.885403	-125.776703	2.537370
6	0.861609	0.019998	0.009351	43.077356	-127.775901	43.082431	0.467554
7	0.009692	0.019998	0.009351	0.127356	1.843855	1.843855	0.467554

Note that for any surface *N*, some of the values reported (such as Waist Size, Waist Distance, ...etc.) refer to the region between surface *N* and surface *N*+1.

The spreadsheet may be used with lenses having an infinite (i.e., greater than 1.0e+8) object conjugate. In this case, using a beam specification of "0" (the default) means that the beam parameters refer to the incident beam at surface 1.

In other words, the reference plane is the tangent plane to surface 1, and the beam parameters specify the beam before refraction/reflection. For infinite object conjugate systems, the z coordinate for the spot size plot refers to the axial distance from surface 1.

The beam may be specified on any surface of the lens (including the image surface), by changing the value of the Beam specification surface cell. The values for waist size, waist distance, and wavefront radius of curvature refer to the values after refraction/reflection at the beam specification surface. (The only exception to this is the case of an infinite object conjugate and a beam specification surface of 0, as described in the preceding paragraph.)

The beam may be evaluated on any surface of the lens (including surfaces before the beam specification surface). The evaluation plane may be adjusted by entering a value in the Evaluation surface shift cell. Note that this is just a shift of the surface in the image space of the evaluation surface; the influence of other surfaces in the lens is not considered.

Plot the Gaussian Beam

The “Plot beam spot size” button produces a schematic illustration of the propagation of the currently defined Gaussian beam. The horizontal axis is the propagation distance from the object surface. Surfaces are drawn as vertical lines, since this is a paraxial analysis. If the “Slider wheel design” radio button is selected, the “Slider Window” is also displayed. By moving the thumb of the slider, you can move a vertical indicator bar along the z axis; the value of the z position of the indicator bar is shown in the Slider Window, and the beam spot size is displayed at the bottom of the Interactive design graphics window. The spot size values shown are the distance to the $1/e^2$ radius of the Gaussian beam in the current system units (mm = default system units). In the drawing, all propagation distances are positive, so mirror systems, for example, are displayed in an “unfolded” view.

If the currently defined beam has two solutions, the spot size for solution I is displayed in green above the horizontal axis, and solution II is displayed in blue below the horizontal axis. If the current beam has only one solution, the spot size is displayed on both sides of the axis. Note that the horizontal and vertical lines are not drawn to the same scale; the drawing is meant to be illustrative, not a scale lens drawing. The default vertical scale is determined by the maximum beam size on any lens surface. Another scale may be used by setting the value of the **Graphics_scale (gsca)** preference (see the section “Set Preference / Show Preference” starting on page 174). The plot will be drawn such that the vertical direction corresponds to a distance of $\pm \text{gsca}$ from the axis.

Skew Gaussian Beam (Astigmatic Gaussian Beam)

The Skew Gaussian Beam trace option allows you to trace a (possibly) astigmatic Gaussian beam along the reference ray from the current object point. This option should be used if your lens contains any special data (tilts, diffractive elements, gradient index media, etc.) or if you wish to propagate a beam along a path that does not coincide with the optical axis of the system.

The propagation will be along the currently defined reference ray. To specify a different reference ray, choose the “Set Object Point” button in the input dialog (see below). Differential rays are traced to determine the first-order properties of the system, using the reference ray as a base ray. Thus, local effects up to second-order in the aperture coordinate are accounted for accurately. It is assumed that all aperture sizes are such that any truncation or aperture diffraction effects can be ignored. The beam profile then remains a Gaussian as it propagates through the system.

The input Gaussian beam is propagated through the lens, beginning at the object surface (surface 0). Since all propagation must be over a finite distance, this analysis can only be performed using a finite object conjugate.

Choosing this menu option results in a spreadsheet in which you can define the input parameters of the beam.

Output format control

There are two forms of output available: Standard and Full. If you select standard, information about the beam size and wavefront radius of curvature will be displayed for each desired surface. In addition, if you select Full, information about the beam waist will be displayed.

Surface selection option

You can display the beam data on all lens surfaces or on a selected range of surfaces. If you select the Range of surfaces option, enter the first and last surfaces in the desired range in the First surface number and Last surface number cells.

Beam sizes at object surface

These are the radii at which the input beam irradiance drops to $1/e^2$ of its central value. The y and x values of the beam size can be different; this defines an elliptical input beam. If the x beam size is entered as 0, it is taken to be equal to the y beam size (i.e., a circular input beam). The y beam size must be non-zero.

Wavefront radii of curvature at object surface

These the radii of curvature of the beam wavefront in the y and x directions. The default value of 0 indicates a planar wavefront, i.e., a beam waist. The sign convention for radius of curvature is the same as for specifying thickness and radii of curvature of lens surfaces. (See Chapter 4.)

Azimuthal orientation angle of beam

If the input beam is not oriented along the x and y axes, this is the angle of azimuthal rotation, in degrees, of the beam, measured from the x -axis. The angle should be a value between -45 degrees and $+45$ degrees.

After you have specified the input beam, click "OK" to trace the Gaussian beam through the lens. For each surface in the specified surface range the following output is displayed. Note that, in general, the coordinate systems used for the reporting of spot size and wavefront curvature information will be different.

- **Y SPT SIZE, X SPT SIZE, and BEAM AZMTH** - The $1/e^2$ beam irradiance spot radii in the y and x directions and the orientation angle of the spot ellipse, measured from the x -axis. The spot sizes and orientation are measured in a local, right-handed (x, y, z) coordinate system. The xy plane is tangent to the surface at the point of intersection of the center of the beam (i.e., the reference ray). In other words, the local z -axis lies along the local surface normal. The local x -axis is orthogonal to the y -axis of the surface. In the familiar case of a beam traveling down the optical axis of a centered optical system, the beam is incident at the vertex of each surface. Thus, the local coordinate system for the spot sizes is the same as the local coordinate system in which the surface is defined.
- **Y RFR RAD, X RFR RAD, and PHASE AZMTH** - The wavefront radii of curvature in the y and x directions and the orientation angle of the phase ellipse, measured from the x -axis. A radius of curvature value of 0 indicates a planar wavefront. The radii of curvature and orientation are measured in a local, right-handed (x, y, z) coordinate system that has its z -axis along the center of the beam (i.e., along the reference ray). The local x -axis is orthogonal to the local y -axis of the surface. Note that the radii of curvature are the values *after* refraction/reflection/diffraction at the surface.

If you have chosen the Full output option, a second line of information about the beam waist is displayed for each surface.

- **Y WST SIZE and X WST SIZE** - The beam waist sizes in the two orthogonal sections of the beam.
- **Y WST DST and X WST DST** - The distance along the beam from the intersection of the beam with the surface to the beam waist in each orthogonal section of the beam.

Note that both the waist sizes and waist distance refer to the beam *after* refraction/reflection/diffraction at the surface. If the beam suffers from general

astigmatism, the waist is not defined and a message to that effect will be displayed in place of the waist size and distance values.

M-squared value - A discussion of M^2 may be found on p. 50 of the Optics Reference (p. 59 of the electronic version).

***TRACE GAUSSIAN BEAM**

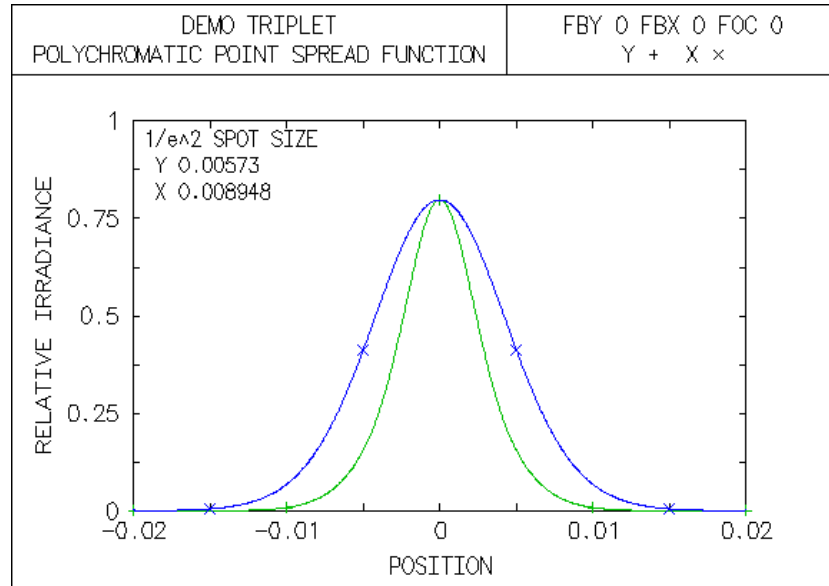
WAVELENGTH = 0.587560

SRF	Y SPT SIZE	X SPT SIZE	BEAM AZMTH	Y RFR RAD	X RFR RAD	PHASE AZMTH
	Y WST SIZE	X WST SIZE		Y WST DST	X WST DST	
0	1.000000	1.000000	--	--	--	--
	1.000000	1.000000		--	--	
1	1.000175	1.000175	--	55.508188	55.508188	--
	0.006405	0.006405		55.505911	55.505911	
2	0.964138	0.964138	--	29.244970	29.244970	--
	0.005673	0.005673		29.243958	29.243958	
3	0.766333	0.766333	--	128.601125	128.601125	--
	0.019408	0.019408		128.518638	128.518638	
4	0.760374	0.760374	--	-51.870637	-51.870637	--
	0.012757	0.012757		-51.856037	-51.856037	
5	0.848329	0.848329	--	-125.729274	-125.729274	--
	0.017103	0.017103		-125.678173	-125.678173	
6	0.861824	0.861824	--	43.091173	43.091173	--
	0.009351	0.009351		43.086100	43.086100	
7	0.009739	0.009739	--	1.742018	1.742018	--
	0.009351	0.009351		0.136100	0.136100	

Truncated Gaussian Beam

Plots X and Y cross scans of the Point Spread Function (PSF), computed by direct integration from the current spot diagram. There are two plots shown: PSF vs. X at Y = 0, and PSF vs. Y at X = 0. The $1/e^2$ spot sizes (semi-diameter in the current system units) are also reported in the upper left corner of the plot for the X and Y cross sections.

Note that for a Gaussian (apodized) beam, the beam properties (spot sizes) must be entered using the SETUP button in the lens data spreadsheet, or directly using the **sdgx** and **sdgy** commands (see the section "Gaussian pupil apodization specification (sdaz)" on page 123). If the Gaussian apodization of the system is not defined, the plot defaults to the regular PSF scan plot described in the section "Plot PSF Scans" on page 316, and the $1/e^2$ spot size dimensions are not reported.



Fiber Coupling

The coupling efficiency (η) between an optical system and an optical fiber is defined as the normalized overlap integral between the amplitude diffraction pattern (U) and the mode pattern of the fiber (ψ).

$$\eta = \frac{\iint U(x, y) \psi^*(x, y) dx dy}{\sqrt{\iint U(x, y) U^*(x, y) dx dy \iint \psi(x, y) \psi^*(x, y) dx dy}} \quad (11.7)$$

where the asterisk denotes the complex conjugate. The power coupling efficiency T is given by

$$T = \eta \eta^* = |\eta|^2 \quad (11.8)$$

OSLO computes the diffraction amplitude distribution (U) over a grid of points in the image plane using FFT diffraction, and then computes the value of η , integrating over this grid. The fiber mode (ψ) is specified as part of the command. The default orientation of the fiber is for the center of the fiber to be coincident with the reference ray intersection with the image plane and the face of fiber to be in

the (x, y) plane. The fiber may, however, be tilted and/or decentered from this position.

The efficiency computation is a monochromatic calculation and is performed at the wavelength specified by the value entered in the Wavelength number cell in the spreadsheet. Since OSLO uses an FFT to compute the amplitude in the diffraction pattern, you can specify the size of the FFT grid by clicking the appropriate radio button and the number of rays that are traced across the pupil of the lens by entering the value in the cell. As mentioned above, by default, OSLO assumes that the face of the receiving fiber is located in the (x, y) plane of the image surface and that the center of the fiber is located at the intersection point of the current reference ray with the image surface. If you wish to decenter the fiber from this location, enter the desired decentration values in the Fiber displacement in y-direction and Fiber displacement in x-direction cells. If you wish to tilt the face of the receiving fiber, the tilt angles (in degrees) are entered in the Fiber tilt around y-axis and Fiber tilt around x-axis cells.

- **To use a Gaussian fiber mode**, select the Gaussian mode radio button. The only value needed to specify this mode is $1/e^2$ radius of the Gaussian, which is entered in the Gaussian mode $1/e^2$ radius cell.
- **To use the fundamental mode of a step-index fiber**, select the Fundamental step-index mode radio button. This mode is defined by the refractive indices of the core and the cladding, and the radius of the fiber core. These values are entered in the Core index, Cladding index, and Core radius spreadsheet cells.
- **To use a user-defined mode**, select the User-defined mode radio button. A user-defined mode is calculated by a CCL command, the name of which is entered in the CCL command name to compute mode cell. For each grid point (x, y), the CCL command will be executed, and the global variables **Ufm_x** and

Ufm_y will be set to the values of *x* and *y* (relative to the center of the fiber) at which the mode is to be computed. The CCL command should compute the mode, and set the values of the global variables **Ufm_re** and **Ufm_im** equal to the real and imaginary parts of the mode.

Output

The output consists of a description of the fiber mode, along with the following information.

- **FIBER DISPLACEMENT Y and X**- the displacement of the center of the fiber from the intersection point of the reference ray with the image surface.
- **FIBER TILT TLB and TLA** - the tilt of the fiber face around the *y* and *x* axes, respectively.
- **POWER COUPLING** - the fraction of the total power in the diffraction irradiance point spread function that is coupled into the fiber mode. The value is given as both an absolute number and in decibels (dB). This value is computed using Eq. (11.8).
- **AMPLITUDE COUPLING REAL and IMAGINARY** - the real and imaginary parts of the coupling efficiency integral, Eq. (11.7).

```
*FIBER COUPLING EFFICIENCY - WAVELENGTH 1
GAUSSIAN MODE - 1/e**2 RADIUS =    0.005000
FIBER DISPLACEMENT  Y      --      X      --
FIBER TILT           TLB     --      TLA     --
POWER COUPLING =    0.654      (  -1.844    dB)
AMPLITUDE COUPLING  REAL =    0.4242    IMAGINARY =    0.6885

*FIBER COUPLING EFFICIENCY - WAVELENGTH 1
FUNDAMENTAL MODE OF STEP-INDEX FIBER
CORE INDEX =    1.500000 CLADDING INDEX =    1.490000 CORE RADIUS =    0.005000
FIBER DISPLACEMENT  Y      --      X      --
FIBER TILT           TLB     --      TLA     --
POWER COUPLING =    0.7071      (  -1.505    dB)
AMPLITUDE COUPLING  REAL =    0.5854    IMAGINARY =    0.6037
```

More information on the computation of fiber coupling efficiency may be found in Chapter 7 – Image Evaluation of the Optics Reference manual.

Partial Coherence

Prm

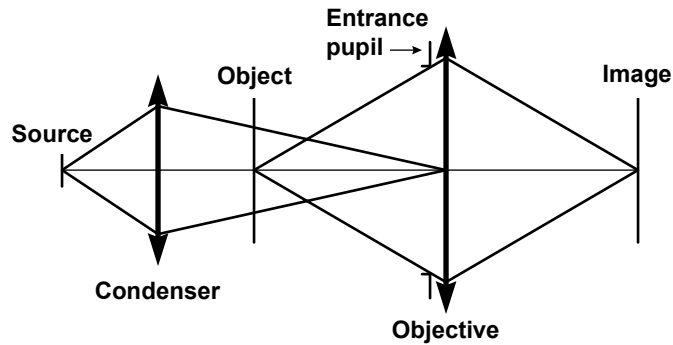
Print/Plot Image...

Partial Coherence Conditions

The structure of an image obviously depends on the object being imaged and the quality of the lens that forms the image, but it also depends on how the object is illuminated. The Partial Coherence analysis options allow you to study these effects on trans-illuminated objects. The theory of optical imaging with partially coherent illumination is given in Chapter 7 – Image Evaluation of the Optics Reference manual.

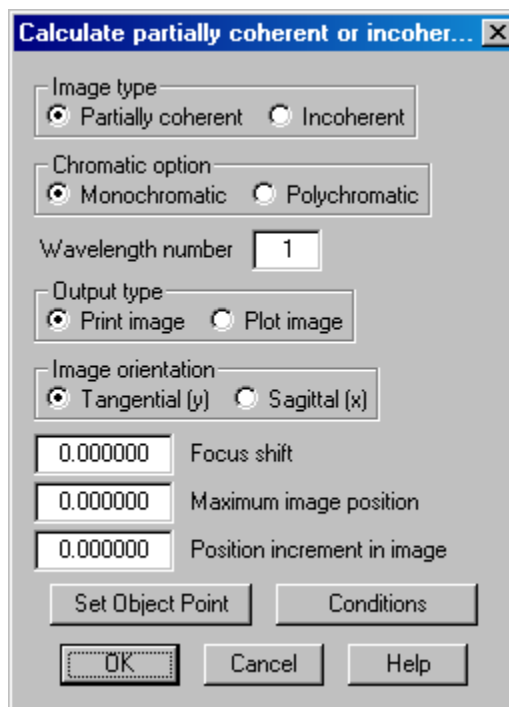
The model used to formulate the image formation equations assumes what is sometimes called “matched illumination.” Matched illumination is required for stationary imaging. Thus, in OSLO, Köhler illumination of the object is assumed to

exist, as shown below. You do not have to enter any extra surface data to account for the illumination. In a Köhler illumination system, an incoherent source is imaged into the entrance pupil of the imaging lens. It is the size and position of this “effective source” that determines the coherence properties of the light that illuminates the object.



Print/Plot Image

Once you have defined the source and image operating conditions, you can compute the resulting image by selecting this menu option.



- **Image type** - You can compute the image using either partially coherent or completely incoherent light. Click the radio button next to the desired option. As mentioned above, values of the relative effective source radius (σ) greater than 2 are essentially incoherent, so it is recommended that you use the Incoherent option for these cases. The length of time required for the computations increases with σ .
- **Chromatic option** - The computed image can be either monochromatic or polychromatic. If you select Monochromatic, enter the number of the wavelength you wish to use in the Wavelength number cell. If you select Polychromatic, all defined wavelengths will be used, along with the current wavelength weights.
- **Output type** - You can request to have the computed image reported in either tabular (Print image) or graphical (Plot image) form.
- **Image Orientation** - You can select whether the image is oriented in the Tangential (y) or Sagittal (x) direction. Available for "Print image" option only. The "Plot image" option will graph both orientations automatically.
- **Focus shift** - Allows you to evaluate the image on an image surface displaced from its nominal location (along the local Z axis).
- **Maximum image position & Position increment in image** - Enter the maximum image extent (image height) and sampling increments for analysis purposes. The analysis will report values from + Max. image position to - Max. image position. Available for "Print image" option only.
- **Set Object Point** - The image will be computed using the current object point unless the "Set Object Point" button is chosen and another reference ray is traced. You can also set the current object point outside of performing this analysis. See the section "Standard Tools" on page 143 for some ways to change the object point using the OSLO text window toolbar.
- **Conditions** - Clicking on this button will take you to the dialog window where you can review and modify the Partial Coherence Conditions.

Print Output

The tabular output consists of two columns of data.

- **Y or X** - The position on the image surface, relative to the intersection of the reference ray with the image surface. The Y or X value depends on what you chose for image orientation.
- **IRRADIANCE** - The relative irradiance at that point.

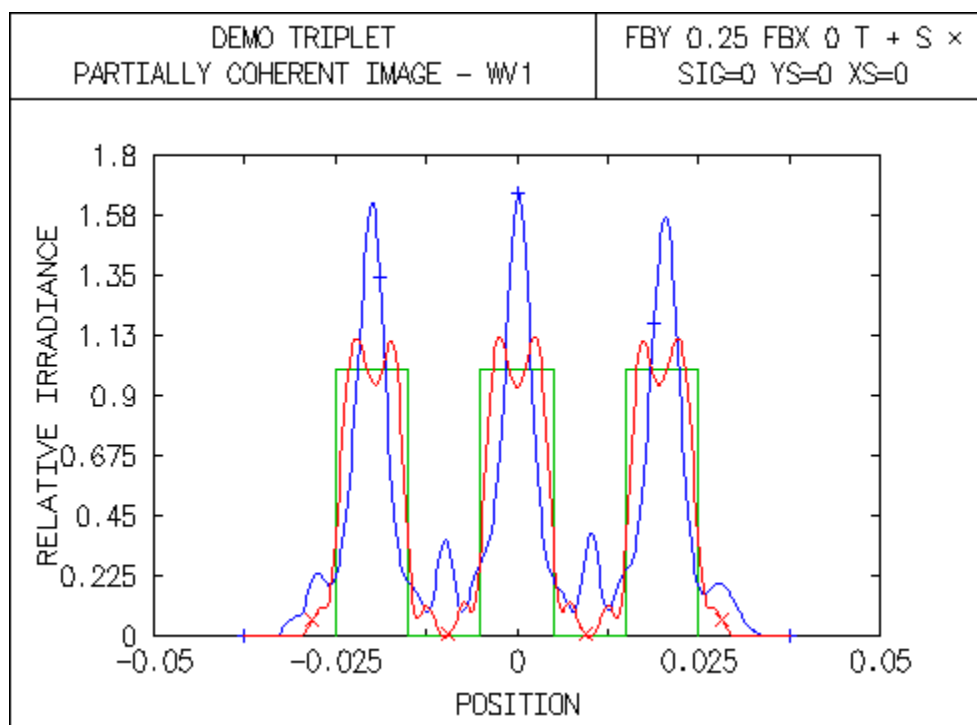
*PARTIALLY COHERENT IMAGE: MONOCHROMATIC

WAVELENGTH 1

NBR	Y	IRRADIANCE
1	-0.037528	0.001553
2	-0.033775	0.001130
3	-0.030023	0.010466
4	-0.026270	0.262668
5	-0.022517	0.846025
6	-0.018764	1.161343
7	-0.015011	0.240165
8	-0.011258	0.078336
9	-0.007506	0.215291
10	-0.003753	0.499012
11	--	1.255662
12	0.003753	0.499012
13	0.007506	0.215291
14	0.011258	0.078336
15	0.015011	0.240165
16	0.018764	1.161343
17	0.022517	0.846025
18	0.026270	0.262668
19	0.030023	0.010466
20	0.033775	0.001130
21	0.037528	0.001553

Plot output

If you chose to plot the resulting image, three curves are displayed: the geometrically perfect image of the bar target and the resulting image for both tangential and sagittal orientations.



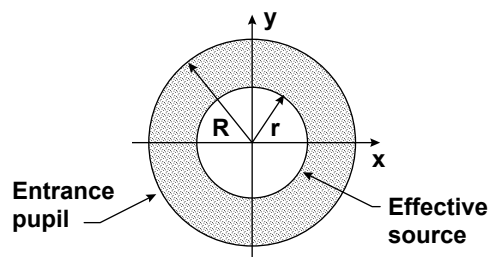
Partial Coherence Conditions

The properties of the effective source and ideal image are determined by the Partial Coherence operating conditions. The ideal image is a bar (amplitude and/or phase) target.

Relative effective source radius (sigma):	<input type="text" value="0.000000"/>
Y source shift: <input type="text" value="0.000000"/>	X source shift: <input type="text" value="0.000000"/>
Relative inner radius of annular source:	<input type="text" value="0.000000"/>
Gaussian source-Y size: <input type="text" value="0.000000"/>	X size: <input type="text" value="0.000000"/>
Number of points in image (power of 2):	<input type="text" value="64"/>
Number of clear bars in image:	<input type="text" value="3"/>
Period of clear bars: <input type="text" value="0.020000"/>	width of clear bar: <input type="text" value="0.010000"/>
Irradiance transmittance between bars:	<input type="text" value="0.000000"/>
Phase transmittance between bars:	<input type="text" value="0.000000"/>
Background irradiance outside bar pattern:	<input type="text" value="0.000000"/>
Normalization:	<input checked="" type="radio"/> Unity object irradiance <input type="radio"/> Object power
Spot diagram type:	<input checked="" type="radio"/> Object space <input type="radio"/> Image space

Relative effective source radius (pcsg)

The ratio of the diameter of the (circular) effective source to the diameter of the entrance pupil. This quantity is commonly denoted as sigma (σ). Referring to the figure below, $\sigma = r/R$. A value of zero is a point source (fully coherent) and a value of one means the pupil is completely filled by the effective source. For most objects (particularly amplitude objects) a value of σ greater than 2 is essentially indistinguishable from the incoherent limit. Since the incoherent image calculation is considerable faster than the partially coherent image calculation for large values of σ , it is recommended that you use the incoherent image option if you want a value of σ greater than 2.



Y source shift (pcys)

The y direction shift of the center of the effective source measured in fractional pupil coordinates from the center of the entrance pupil.

X source shift (pcxs)

The x direction shift of the center of the effective source measured in fractional pupil coordinates from the center of the entrance pupil.

Relative inner radius of annular source (pcia)

The inner radius of an annular effective source, specified as a fraction of the outer radius of the effective source.

Gaussian source Y size (pcgy)

The $1/e^2$ irradiance point in the y direction of a Gaussian effective source, specified relative to the outer radius of the effective source. Thus, for example, a value of 1.0 means the $1/e^2$ point coincides with the edge of the effective source. A value of 0 indicates a uniform effective source irradiance distribution in the y direction.

Gaussian source X size (pcgx)

The $1/e^2$ irradiance point in the x direction of a Gaussian effective source, specified relative to the outer radius of the effective source. Thus, for example, a value of 1.0 means the $1/e^2$ point coincides with the edge of the effective source. A value of 0 indicates a uniform effective source irradiance distribution in the x direction.

Number of points in image (pcip)

The number of discrete points at which the image irradiance is calculated. This number must be a power of 2, because of the computational techniques involved in the calculations (i.e., Fast Fourier Transform).

Number of clear bars in image (pcnb)

The number of clear, i.e., fully transmitting, bars that constitute the object (and ideal image).

Period of clear bars (pcbp)

The center-to-center spacing of the bars, in lens units. The bars have an irradiance transmittance of 1.0 and a phase transmittance of 0.0.

Width of clear bar (pcbw)

The width of each clear bar, in lens units.

Irradiance transmittance between bars (pcmi)

The irradiance transmittance of the spaces between the bars. A value of 0 means the spaces are "black," i.e., fully absorbing.

Phase transmittance between bars (pcsp)

The phase transmittance, in degrees, of the spaces between the bars.

Background irradiance outside bar pattern (pcbg)

The irradiance transmittance outside the region of bars and spaces.

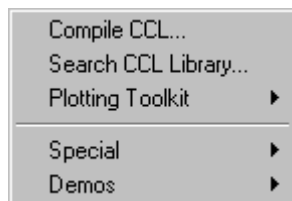
Normalization (pcen)

The image irradiance normalization criterion. If Unity object irradiance is chosen, the object is taken to have a uniform irradiance of unity. If Object power is chosen, the image has the same total power as the object.

Spot diagram type (pcis)

If Object space is chosen, an object space spot diagram will be traced. If Image space is chosen, an image space spot diagram will be traced. See the section “Image space spot diagram (sdis)” on page 120 for a discussion of object and image space spot diagrams.

Overview



This menu contains tools for effectively working with CCL files and efficiently creating plots of data in OSLO. You can also sample demonstrations of a wide range of OSLO capabilities.

Although OSLO has a large number of built-in features, there are always instances where the ordinary capability of the program is not sufficient for a particular design task. To handle these, OSLO contains a macro language that is tightly coupled to the main program. There are actually two macro languages, with slightly different purposes.

The SCP (Star Command Programming) language that is available in all versions of the program is designed to be easy to master, and is suitable for small extensions and modifications to the program that need to be made on a continuing basis. SCP commands are also sometimes called star commands, because the first character of the command must be an asterisk. CCL commands have no such restriction.

CCL (Compiled Command Language) on the other hand, is more formal and suitable for major programming projects. Because CCL is a compiled language, you will find the CCL compiler and search utility useful tools in working with CCL. Both SCP and CCL use standard C syntax.

OSLO provides a plotting toolkit which enables users to conveniently plot data within OSLO. Examples of possible plot types are provided along with a utility that creates properly formatted plotting command syntax from user friendly templates.

The last two menu items access examples of extensions to OSLO written in SCP and CCL. "Special" demonstrates comprehensive analysis capabilities developed by OSLO users. "Demo" demonstrates routines of a more general nature which samples the range of OSLO capabilities.

To understand the details of how a command on the "Special" and "Demo" menu works, you will find it helpful to look at the source code, which is contained in the public CCL and SCP directories (commands that you write yourself should generally be placed in the private CCL and SCP directories). If you are familiar with C, you can possibly understand the commands without additional reference. If you are not familiar with C, you may wish to consult any standard reference, as well as the Optics Reference, the OSLO on-line help and the introductory material in Chapter 16. There are many library functions supplied with CCL, and these are described in Chapter 17.

Menu Item	Page
Compile CCL... Run the CCL compiler on private or public files.	466
Search CCL Library... Scans CCL files in the private and public directories for information about the existing CCL files.	466

Plotting Toolkit... Shows samples and creates templates that enables users to efficiently plot numerical values within OSLO.	467
Special... Execute special CCL routines contributed by OSLO users	471

Compile CCL

As mentioned before, *.ccl files are compiled automatically when they are saved from the OSLO Editor, so the Compile CCL command is needed only for files that are saved from outside of OSLO. The command allows you to choose the type of file to compile (programs, lists, strings, or menus) as well as whether to compile only the private or both the private and public files.

More information on the contents of these files is given in Chapter 17.

Search CCL Library

This menu option scans CCL files for function and argument names. This is a tool for finding information about the existing CCL files, either in the Public directory, Private directory or both. The information retrieved can be:

- **Functions:** names, arguments, returned type, whether they are protected (“static”) or not, file where they are defined and line where they are defined.
- **Arguments:** names, types, the number of times they are used, in which files, in which line.
- **Global variables:** names, type, the names of the files where they are defined, the line where they are defined.

Each item (functions, arguments, global variables) can be sorted either by name or filename. Since CCL commands are typically stored in CCL files with different names, this analysis is especially useful in determining where a particular CCL command is located.

Here is a sample of the beginning of the analysis output, listing the public CCL directory and ordered in alphabetical order by function name.

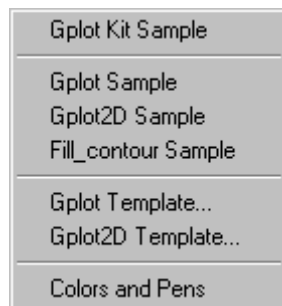

```
** 610 functions are defined in the public directory:**
** C:\Program Files\OSLO\Prm62\public\ccl\
```

TYPE	FCTN NAME	ARGUMENTS	LINE	DEFINITION	FILE
cmd	add_incp1r		643	raytr_wvguide1.ccl	
cmd	add_outcp1r		790	raytr_wvguide1.ccl	
static cmd	ald_asph		74	eval_aldis.ccl	
static cmd	ald_asph		74	eval_aldis.ccl	
static cmd	ald_parax		136	eval_aldis.ccl	
static cmd	ald_parax		136	eval_aldis.ccl	
static cmd	ald_ray		153	eval_aldis.ccl	
static cmd	ald_ray		153	eval_aldis.ccl	
static cmd	ald_sei		191	eval_aldis.ccl	
static cmd	ald_sei		191	eval_aldis.ccl	
cmd	aldis		7	eval_aldis.ccl	
		double aldis_fby			
		double aldis_fy			
		double aldis_fx			
cmd	aldis		7	eval_aldis.ccl	
		double aldis_fby			
		double aldis_fy			
		double aldis_fx			
static cmd	amoeba		58	optim_simplex.ccl	
		double amb_p[][]			
		double amb_y[]			
		int ndim			
		double ftol			
		int nfunck			
		int vsiz			
static double	amotry		171	optim_simplex.ccl	
		double at_p[][]			
		double at_y[]			
		double at_psum[]			
		int at_ndim			
		int at_ihi			
		double fac			
		int ndimp1			

...the analysis output continues...

This is a CCL function. See code in "asyst_scanccl1.ccl" for details.

Plotting Toolkit



The GPLOT plotting toolkit provides a set of routines that allow you to easily plot numerical values. Templates are provided so that you do not have to understand the sometimes complicated syntax of creating the plotting commands directly. Templates are similar to "Wizards" in other Windows programs. Using a template, you answer questions about your data storage variables and plot preferences via a dialog window, and a properly formatted GPLOT command statement is created from your input values.

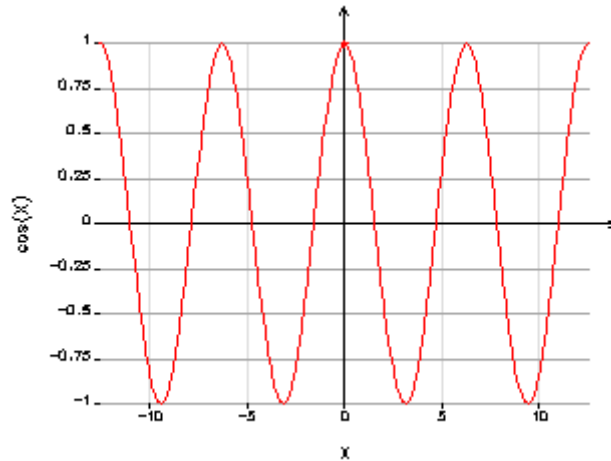
Gplot Kit Sample

This functions demonstrates the possible uses of the functions belonging to the Gplot Toolkit by plotting different flavors of PSF values. Individual samples of the routines that comprise the Gplot toolkit (**Gplot**, **Gplot2D** and **Fill_contour**) are described below.

Gplot Sample

Input Format: Double array_x[], Double array_y[]

Uses: Y vs. X curves

Output Sample:

The picture in the table was generated by the `gplot_sample` routine which is defined in the file **graph_plot_ex.ccl**. The sample routine graphs the equation:

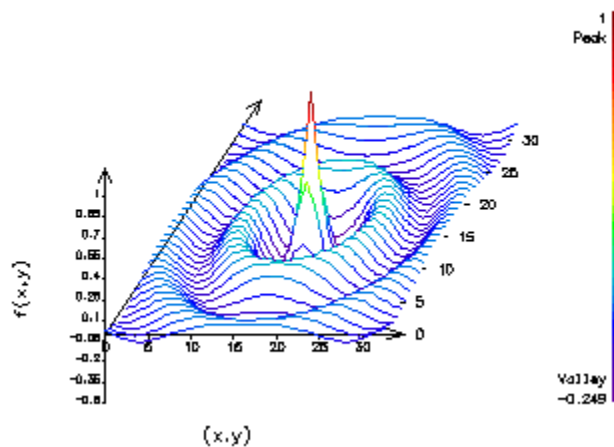
$$Y = f(X) = \cos(X)$$

It is recommended that you investigate the **Gplot Template** described below.

Gplot2D Sample

Input Format: `Double array[][]`

Uses: Pseudo-3D plots, color maps

Output Sample:

The picture in the table was generated by the `gplot2D_sample` routine which is defined in the file **graph_plot_ex.ccl**. The sample routine graphs the set of equations:

$$Z = f(X, Y) = f(R) = \cos(R)/(1 + R)$$

$$R = \sqrt{X^2 + Y^2}$$

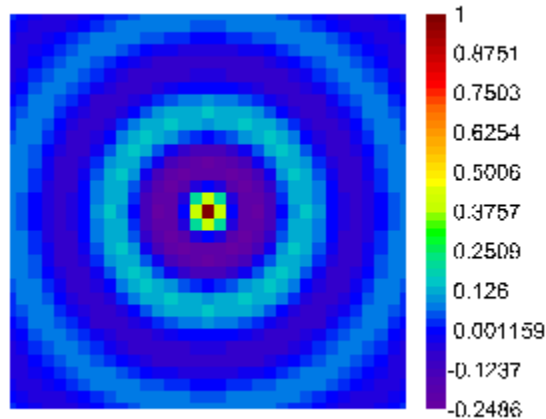
It is recommended that you investigate the **Gplot2D Template** described below.

Fill_contour Sample

Input Format: Double array[][]

Uses: Color maps (OpenGL)

Output Sample:



The picture in the table was generated by the fill_contour_sample routine ("Tools>>Plotting Toolkit>>Fill_Contour Sample" from menu). GPLOT2D_FILL_CONTOUR source code is defined in GR_PLOT_EX.CCL.

Gplot Template

Given the flexibility of *Gplot*, finding the sequence of arguments to use in the proper syntax can be difficult. the *Gplot* template option from the OSLO menu will allow you to generate the correct command through a dialog box. Just fill-in and select the appropriate options you want and click OK. This template will not do anything, except put the arguments in the command line history buffer

Example:

After filling out the *Gplot Template* dialog box with some data and clicking OK, assume the following command is written to the command line history:

```
gplot_template "1" 20 "ArrayX1" "ArrayY1" "Curve #1" "Y axis" "X axis" y y scs 0.0
0.0 0.0 0.0
```

To use the real gplot command in a CCL routine, you can copy it over and modify it as follows:

```
gplot "1" 20 ArrayX1 ArrayY1 "Curve #1" "Y axis" "X axis" y y scs 0.0 0.0 0.0 0.0
```

As you see, only 2 changes had been made:

- *GPLOT_TEMPLATE* is now *GPLOT*: you need to call the real *GPLOT* function, not the one that simulates the input.

- "ARRAYX1" and "ARRAYY1" (with quotes) are now ARRAYX1 and ARRAYY2 (without quotes). This is necessary to use the real arrays declared in your routine, and not a string.

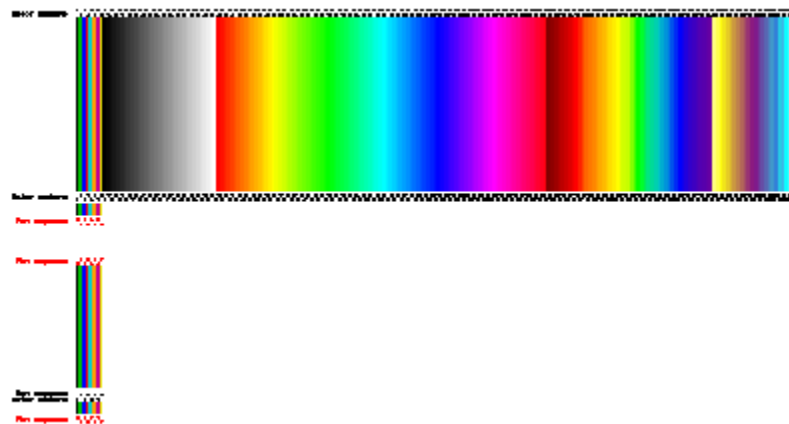
Gplot2D Template

The *Gplot2D Template* works just like the *Gplot Template* (see above) except that the command argument choices in the template dialog box will differ. Once the template of the command syntax is written to the command line history, the same two changes must be made as in the *Gplot Template*:

- Change *GPLOT2D_TEMPLATE* to *GPLOT*: you need to call the real *GPLOT2D* function, not the one that simulates the input.
- Array names need to have quotes removed. This is necessary to use the real arrays declared in your routine, and not a string.

Colors and Pens

The **colors** command displays all the colors available to the pen command. Those colors are defined in "a_list.ccl" in the ".../public/ccl" folder.



The upper part of the graph consists of a column for each defined color, with its color number.

If a pen color sequence is defined, the colors used in the sequence are highlighted on the upper map, and the actual succession of colors will be displayed in the lower part of the window.

To accommodate the improvements in the quality of graphics, the palette in OSLO has been designed to include 209 colors.

Color Range	Description	Use
1	Black if White Background, White if Black Background	It is the default color for labels, axes and grids
2 through 8	Vivid Colors	Those colors are used in the default pen sequence. When several curves are plotted on the same graph, they are plotted using colors in this order. The 1st three colors (Green-Blue-Red) are matching the order in which the wavelengths are defined (central-lower-higher), making the curves on relevant plots easier to identify.
9 through 41	Black to White shading	Mostly used for maps, such as relative illumination or interferograms.
42 through 137	Circular Spectrum	Selection of colors equally spaced in RGB space.
138 through 185	False Color Sheet	Used for false color depth rendering (#185 is low, #138 is high), pseudo 3D or color maps. Ex: wave-front plots, GPLOT toolkit routines
186 through 209	Alternative false color set	

Special

Prm 2D Waveguide



A simple optical model of waveguide lenses is two-dimensional ray-tracing. Since OSLO Premium is basically a 3D design program, there is no theoretical obstacle to the analysis of integrated lens systems. But such lens systems have two important properties which are worth building into OSLO Premium as extensions to facilitate lens entry and evaluation.

One of these special characteristics is optical anisotropy. The necessary routines that we wrote to handle this effect were presented previously (see **iolens**, **anitr2D** and **wvra** CCL routines).

The other problem, that we currently solved, is making the connection between 3D and 2D systems. This involves the excitation of waveguide modes with a 3D laser beam (i.e. incoupling), and getting light energy out of the waveguide in the form of a 3D light beam (i.e. outcoupling). We modeled the case when the above mentioned phenomena are realized by total internal reflection prisms.

In what follows we describe how to work with our **ADD_INCPLR**, **ADD_OUTCPLR**, **INCOUPLER**, **OUTCOUPLER** and **FIND_FOB** routines. For background information about the theory of in- and outcoupling please refer to publications [1, 2, 3] as well as our article entitled "New Tool-Kit for the Optical Design of Waveguide Lenses".

Model of in- and outcoupling

Since OSLO Premium works on the basis of geometrical optics we created a ray-tracing model of in- and outcoupling. The result has been a simple algorithm written in CCL (see INCOUPLER, OUTCOUPLER routines), that can be performed at a User-Defined Surface. Below we briefly overview the elements of the model that we built into OSLO Premium; we chose only those phenomena to be modeled that can be well measured in the laboratory and have an effect on optical design.

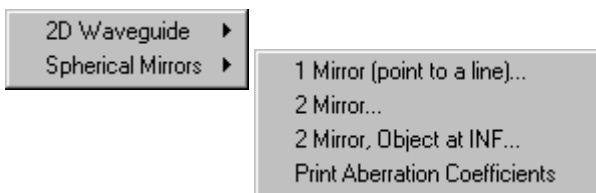
Among different angles of incidence in the prism there exists a so-called synchronous direction (see Figure 1) both at in- and outcoupling. It is the direction in which the incident light transfers power into a definite waveguide mode with maximum efficiency - in the case of incoupling; and in which the outcoupled ray has maximum intensity - in the case of outcoupling (see references [1,2,3]).

This synchronous direction corresponds to the ray direction in the prism which is phase-matched to the desired waveguide mode that propagates under the prism. Phase-matching is essential component of our model.

Further parts of the model:

- In case of incoupling our approximation allows only those incident rays to incouple which are in a given proximity of the synchronous direction. This \pm angle interval is referred to in the messages of our the routines as the "range of coupling angle".
- Since incoupling is possible only in the vicinity of the prism edge, our routines also stops the rays that intersect the prism base at distances measured from the edge more than the so-called "coupling length".
- If the incident angle of rays in the prism exceed the critical angle of total internal reflection with respect to the substrate, these rays cannot be incoupled either.
- At outcoupling the angle range is regarded to be zero, i.e. all the rays in the waveguide couple out under the same angle, which is the synchronous direction.
- At outcoupling the coupling length is also chosen to be zero, i.e. all the rays couple out exactly at the prism edge.
- Diffraction effects at the outcoupler prism edge are not taken into account.

Std Prm Spherical Mirrors



This series of routines is based on research presented in two papers. The first is entitled Imaging a point to a line with a single spherical mirror, and the second Imaging a point with two spherical mirrors. In the first paper, we describe a single spherical mirror system for imaging a point to a line that is aberration free (i.e. no geometric blur). In the

second paper, we investigate unobstructed, plane-symmetric two spherical mirror systems intended for imaging a single point. In this second paper, first and second order imaging constraints are applied to eliminate all but two of the seven parameters necessary to specify the configuration of a two-spherical-mirror system. These CCL's are written as aids in the presentation of results of those two papers.

By employing the graphic sliders interactive design feature of OSLO, the user of these CCLs directly chooses the values of the independent parameters of the system. The remaining (dependent) parameters are then automatically calculated, and the lens data is updated.

Background

In our approach, the imaging properties of the spherical mirror systems are expanded about a ray located at the center of the bundle of rays that pass through the system. This ray is referred to as the base ray, and all mirror tilts and separations are measured with respect to this ray. We consider only plane-symmetric systems, such that the base ray is taken to lie in a plane. The first- and second-order imaging properties are then used to impose constraints on the configuration of these systems, effectively reducing the available number of degrees of freedom to the designer, and thereby simplifying the design task considerably.

It turns out that the single and two-mirror systems found by using the method described above all possess an underlying axis of symmetry. That is, the object point, centers of curvature of the mirrors, and the image point all lie on a line which is the axis of symmetry for the system. To help see this axis, a '+' symbol and an 'x' symbol appears at the centers of curvature of the first and second mirror (when applicable).

Technical Notes

In these routines, a graphics slider is assigned to each of the independent degrees of freedom. In the case where there are multiple solutions (e.g. two-mirror systems), an additional graphics slider is used to select the desired solution family. Also, sliders are assigned so that the user can choose a specific object height (not applicable for the single mirror system) as well as the object space numerical aperture (NA) of the system. When the object is at infinity, the half field of view (HFOV) and the entrance beam radius are used instead of the object height and object space NA. Finally, two graphics sliders are assigned to toggle the displaying of spot diagrams, and for automatic setting of the object and image plane tilts to reduce blur (for the cases where there is some finite object size).

When the object is at infinity, the entrance pupil for each system is located at the first mirror. For finite conjugate systems, the entrance pupil is at infinity (i.e. the systems are telecentric in the object space).

Note that for the two-mirror system, only positive values can be selected for the top slider bar (which is assigned to t_1), whereas for the second slider bar the tilt angle (t_2) can be both positive and negative. Limiting the choices for the tilt of the first mirror does not limit the systems that can be investigated: changing the sign of both tilt angles results in the same system, so that one of the tilt angles can be chosen to be positive without loss of generality. We have chosen that tilt angle to be t_1 .

Also, it is possible for the systems with multiple solutions to result in imaginary values of the constrained system parameters. For example, the finite conjugate case of the two-mirror system involves the solution of a cubic. For this case, at least one of the three solutions of the cubic will be real. Choosing '1' on the "Which solution ?" slider bar will always result in a real solution. However, there are regions of solutions 2 and 3 that result in imaginary values for some choices of mirror tilt angles. The imaginary solutions cannot, of course, be displayed in the

graphics window; so a message stating "Invalid System" is given when this occurs. Note that when an imaginary solution does result, both solutions 2 and 3 fail because they are complex conjugates of one another.

1 Mirror (point to a line)

For the single mirror system imaging a point to a line, five degrees of freedom exist: the mirror curvature (c_1), the object and image distances along the base ray (d_0 and d_1), the mirror tilt angle (t_1), and the image tilt (t_{im}). For this system, the image distance d_1 is automatically calculated from the first order image properties, and the image tilt is determined from the system geometry. These two constraints result in only three remaining degrees of freedom (d_0 , c_1 , and t_1), and one of them (such as c_1) can be used to set the system scale.

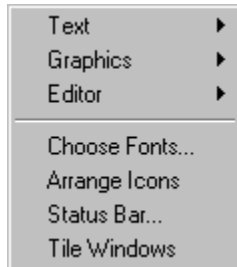
2 Mirror

For the two-mirror system imaging a point, seven degrees of freedom exist: the mirror curvatures (c_1 and c_2), the mirror tilt angles (t_1 and t_2), the object distance (d_0), mirror separation (d_1) and the image distance (d_2). Four degrees of freedom are removed by imposing constraints based on the first- and second-order imaging properties, reducing the original seven parameters to three. However for any set of values of these three parameters, there are potentially three-distinct families of solutions. The three families result from having to solve a cubic equation in determining one of the constraints. In our paper, the tilt angles of the mirrors (t_1 and t_2) are the two independent parameters that are varied for a systematic search, while the mirror separation (d_1) is used to set the system scale.

2 Mirror, Object at INF

If the object point is set at infinity there are two families of solutions, each with two free parameters. A separate CCL command (**two_mir_INF**) is written for this case. Here, the tilt angle of the first mirror (t_1) varies the configuration, while c_1 sets the scale. Non-trivial solutions (i.e., systems in which the mirrors are other than plane) do not exist for the case when both the object and image points are at infinity.

Overview

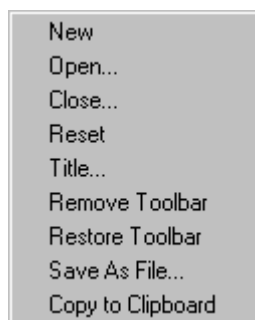


The commands on the Window menu control the various text and graphics windows that OSLO uses for display, provide access to text editors, change fonts that the program uses, arrange positioning of minimized and open windows, and configure the parameters on the status bar.

OSLO has 2 text windows and 32 general-purpose graphics windows which you can have open at the same time. Additional graphics windows for interactive design (see the discussion of the AutoDraw window on page 125) and graphic sliders (page 374) are available, but are not managed by the commands on the Window menu.:

Menu Item	Page
Text... Access commands that manage the 2 possible text output windows.	475
Graphics... Access the commands that manage the 32 possible graphics windows.	479
Editor... Access the text editor commands.	481
Choose Fonts... Select the fonts that are used in windows.	483
Arrange Icons... Arrange the icons of text and graphics windows that have been minimized.	484
Status Bar... Configure the parameters shown in the status bar.	484
Tile Windows... Reposition the open OSLO windows.	484

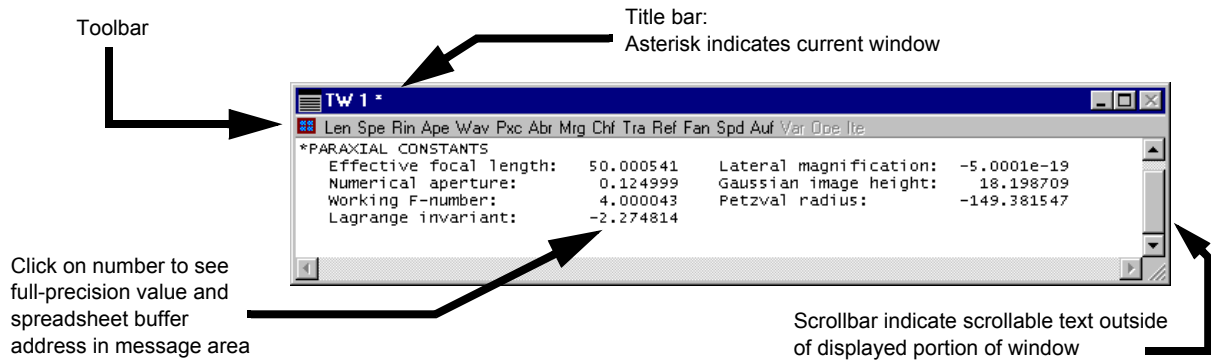
Text



OSLO has two text windows that receive alphanumeric program output. A typical window is shown below.

OSLO uses the concept of a *current* window to determine where to send text output; the current window is the one that has an asterisk in the title bar. The window does not have to be on top of other windows to receive output.

All real numbers sent to a text window are backed up in an internal Spreadsheet Buffer that can be addressed from the keyboard or by macro programs. You can click on a real number in a text window to see its full value in the message area.



OSLO Text windows contain 1999 lines. The windows always have scroll bars, but if there is no text to be scrolled, the scrollbars are disabled. The way in which program output is displayed in a text window depends on the setting of a preference called **Output_page_mode (page)**. If **page** is on, then the first line of output from a command appears at the top of the window; if there are more lines of output than the size of the window, the additional lines are below the bottom, so the lines must be scrolled up to see them. If **page** is off, the last line of output appears at the bottom of the window, and the beginning lines are above the top, so the lines must be scrolled down to see them.

The contents of a text window can be printed, copied to the Windows clipboard, or saved in a text file, using commands on the File or Window menu. What happens when these commands are executed depends on the setting of **page**. If **page** is off, the entire contents of the text window are transmitted. If **page** is on, only the lines below the top line of the display are transmitted.

OSLO has a preference called **Output_text (outp)** that controls whether text output is actually displayed in the current window. If **outp** is on, the window displays text in its normal way. However, if **outp** is off, real numbers in text output sent to the window are placed in the spreadsheet buffer, but the lines are not displayed. This provides a convenient way for SCP and CCL programs to interact with OSLO. By not displaying the lines, text output is speeded up by a large factor, improving the performance of macro programs, and eliminating visual clutter. The setting of **outp** does not affect **print** or **printf** output sent to the window by an SCP or CCL program (**aprint** and **aprintf** output is affected).

n.b. the default CCL/SCP error handler turns output on.

Although **outp** is a very useful feature and is widely used, it can have one negative effect. If a macro program that has turned **outp** off fails at some point in its execution, it may not restore **outp** to on before it terminates, leaving OSLO in a state where it seems to be unable to produce any text output. If this happens, you should execute the command **stp outp on** to see if this solves the problem.

New & Open

The **Text>>New** menu option opens a new text window, if one is available. **Text>>Open** performs a similar function, but prompts for the window number, and allows you to specify the number and length of the lines in the window.

Close

Text>>Close prompts for the number of the window to close and closes the window. When the window is closed, its contents disappear. If you don't want this to happen, minimize the window instead using the controls on the upper right end of the title bar. You can not close the last text window; one needs to be open at all times to receive program output and some error messages (you can send output to a minimized text output window).

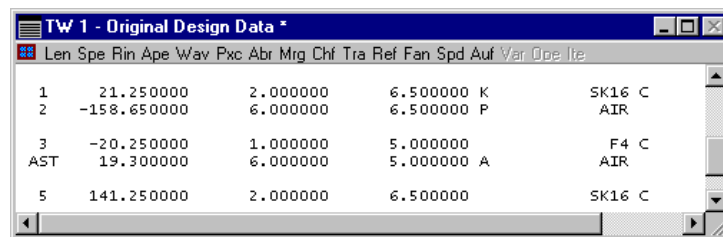
Reset

Text>>Reset carries out three actions:

1. The contents of the window are cleared.
2. The title (if any) is reset to its default value ("Text Window *n*").
3. The spreadsheet buffer associated with the window is set to all zeros.

Title

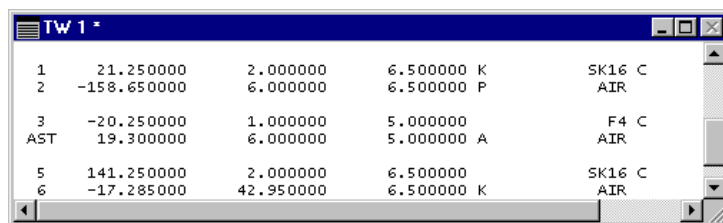
Text>>Title can be used to add a caption to the current text window. OSLO prompts for the title to be added.



	Len	Spe	Rin	Ape	Wav	Pxc	Abr	Mrg	Chf	Tra	Ref	Fan	Spd	Auf	Var	Ope	Ita
1		21.250000					2.000000				6.500000	K				SK16	C
2		-158.650000					6.000000				6.500000	P				AIR	
3		-20.250000					1.000000				5.000000					F4	C
AST		19.300000					6.000000				5.000000	A				AIR	
5		141.250000					2.000000				6.500000					SK16	C

Remove Toolbar

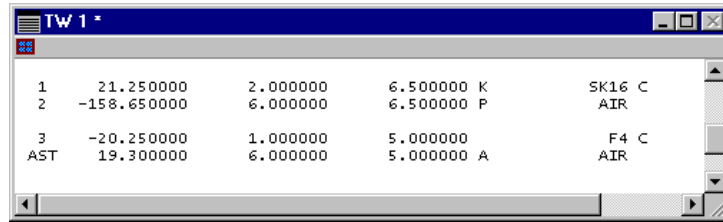
Text>>Remove Toolbar removes the entire section of the window where the toolbars are displayed. When the toolbars are removed, the title bar directly borders the content area of the text window.



	Len	Spe	Rin	Ape	Wav	Pxc	Abr	Mrg	Chf	Tra	Ref	Fan	Spd	Auf	Var	Ope	Ita
1		21.250000					2.000000				6.500000	K				SK16	C
2		-158.650000					6.000000				6.500000	P				AIR	
3		-20.250000					1.000000				5.000000					F4	C
AST		19.300000					6.000000				5.000000	A				AIR	
5		141.250000					2.000000				6.500000					SK16	C
6		-17.285000					42.950000				6.500000	K				AIR	

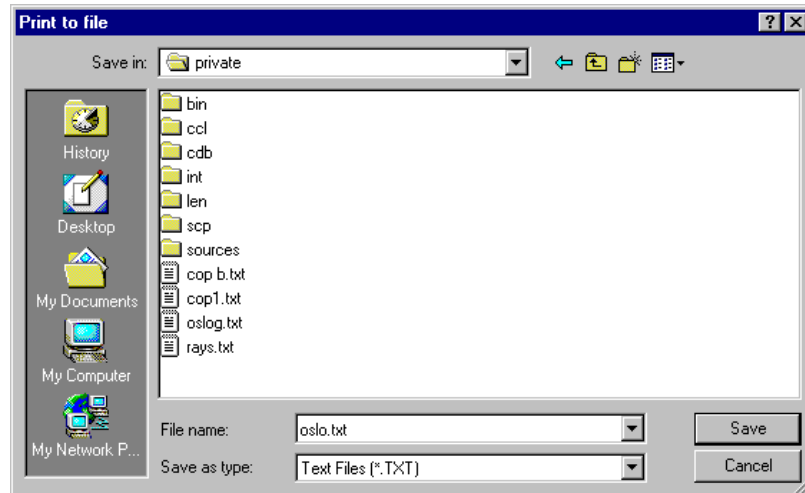
Restore Toolbar

Text>>Restore Toolbar restores the section of the window where the toolbars are displayed. When this section of the window is restored, no toolbars will actually be displayed; you must actively request that toolbars be shown.



Save As File

The output of the text window can be sent to a text file using the **Text>>Save as File** item. This is identical to the **Save Text As** command on the File menu. It brings up a standard Windows dialog box, which allows you to specify the file location, as shown below. The default file location is the private directory.



The only file type allowed is *.txt, which is an ASCII file with the same spaces used for display in the text window. If you import this file into a word processor, you should be sure to use a fixed-spaced font (such as Courier New) to preserve column alignments.

The contents of the file are determined by the setting of the **output_page_mode** (page) preference, as described on page 140.

Copy to Clipboard

Text>>Copy to Clipboard performs a similar function to Save as File, except that the output is sent to the Windows clipboard, from which it can be pasted into another application that supports the clipboard (including the OSLO Editor).

The contents of the clipboard are determined by the setting of the **output_page_mode** (page) preference, as described on page 140.

Graphics

New
Open...
Close...
Reset
Title...
Update All
Remove Toolbar
Restore Toolbar
Save As File...
Copy to Clipboard

All graphical analysis output will appear in general-purpose graphics windows. Up to 32 graphics windows can be open at a time.

Control of graphics windows is predominantly performed through the graphic window interface of each individual window which is described in great detail in Chapter 4, "The Graphics Window". The following sections of this chapter only discuss the items available in the **Window>>Graphics** menu item. These menu items are especially useful if your control of a computer mouse interface is limited or non-existent.

New & Open

The **Graphics>>New** menu item creates a graphics window having the next available number and the default size and position (which is the previous size and position of the window when it was last closed). The **Graphics>>Open** menu item is similar, but prompts for the desired window number and allows you to enter the width and height (in pixels) of the window.

Close

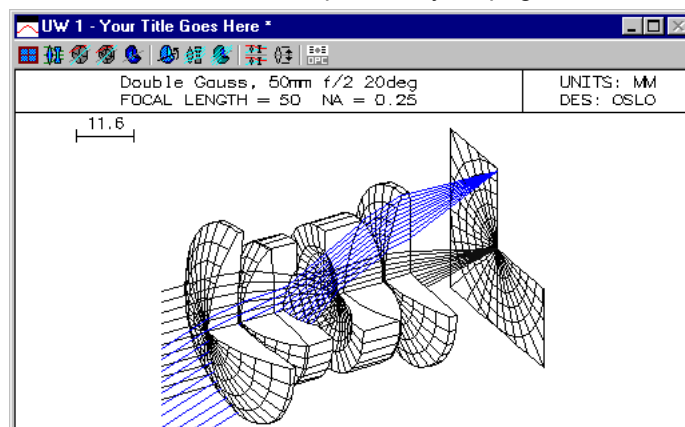
Graphics>>Close prompts for a window number and then closes that window, except that you can not close the last graphics window that is open. When you close a window, its size and position are saved to be restored the next time you open it.

Reset

Graphics>>Reset clears the current graphics window, removes its title (if any), and restores it to a static (GW) window if it was an updatable (UW) window.

Title

Graphics>>Title adds a title to the current graphics window. It is similar to the **Text>>Title** command described previously on page 477.

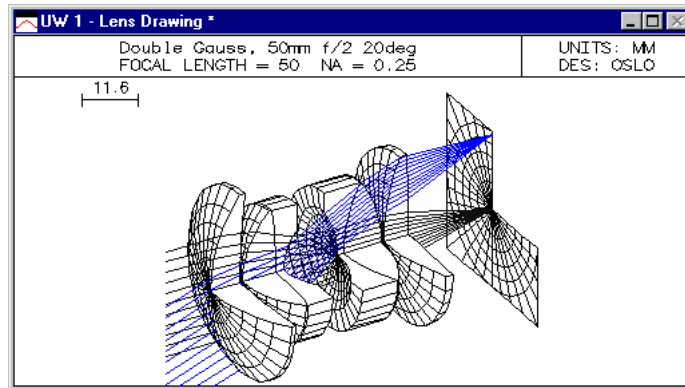


Update All

Graphics>>Update All cycles through all of the graphics windows and updates them using current lens data if they are UW (updatable) windows.

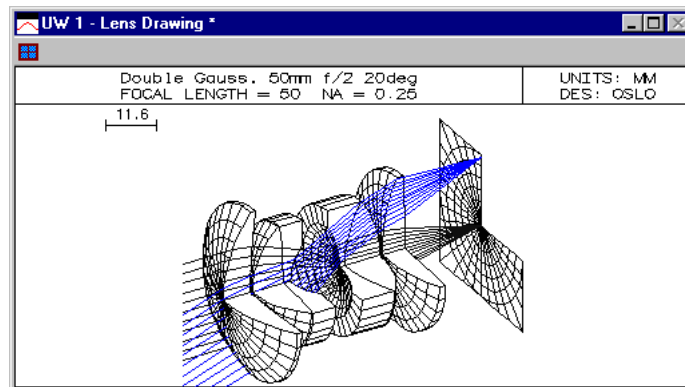
Remove Toolbar

Graphics>>Remove Toolbar removes the entire section of the window where the graphics toolbar is displayed (only one graphics toolbar can be displayed at a time, unlike the text window toolbar). When the toolbar section is removed, the title bar directly borders the content area of the text window.



Restore Toolbar

Graphics>>Restore Toolbar restores the section of the window where the graphics toolbar is displayed. When this section of the window is restored, no toolbar will actually be displayed; you must actively request that a toolbar be shown.



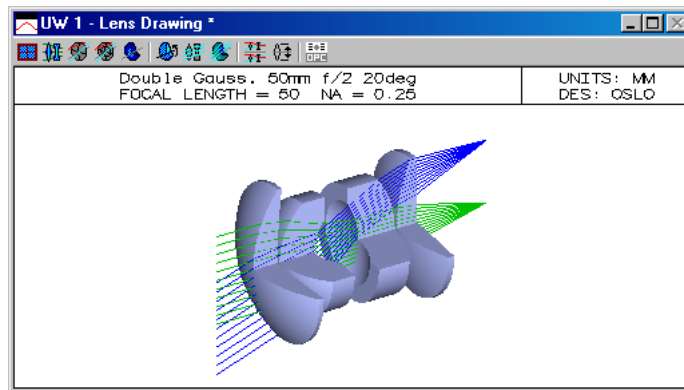
Save As File

The **Graphics>>Save As File** menu item is identical to the **Save Graphics As** item on the **File** menu. The command pops up a dialog box similar to the **Text>>Save as File** box, which allows you to select the file location and type. The default file type is a Placeable Windows Metafile (*.wmf), although 5 other formats are allowed. The various graphics file types are described in the section 'Save Graphics As' on page 172 of Chapter 6.

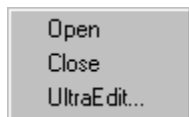
Copy to Clipboard

The **Graphics>>Copy to Clipboard** command is similar to the **Graphics>>Save as File**, except that the output is placed on the Windows clipboard. The graphics are saved to the clipboard using one of two formats

- The default format is a Windows metafile (*.wmf). This is a vector graphics format that can be read by a wide variety of Windows programs.
- When graphics windows contain OpenGL (an Application Programming Interface [API] written by Silicon Graphics), then OSLO uses a bitmap format (*.bmp) for saving the graphic to the clipboard. Graphic windows that utilize OpenGL tend to look gradient or shaded colors such as the Shaded Model shown below.



Editor



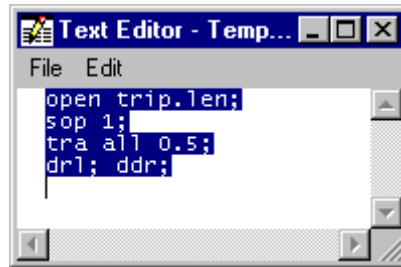
OSLO uses two text editors. The internal OSLO text editor is an integral part of the CCL Application Manager, and has some special features that make it convenient for the development of SCP and small CCL programs, such as the ability to execute selected lines directly and to automatically compile files upon exit. On the other hand, the OSLO text editor is memory-based and is not suitable for working with large files. Because of this, an alternate editor called UltraEdit is supplied for general use and specifically for the development of CCL programs. UltraEdit is a product of IDM Computer Solutions, Inc. (www.ultraedit.com). UltraEdit has been enhanced to work with CCL by the addition of CCL syntax highlighting and automatic tracking of errors generated by the CCL compiler. UltraEdit and a valid authorization code ships with paid versions of OSLO Standard and Premium.

Open

This menu item opens the internal OSLO Editor which is a 32K memory-based text editor especially useful for running a sequence of individual OSLO commands and for writing SCP or CCL routines. SCP and CCL commands are saved as ASCII text files, so any editor could be used for writing commands. The OSLO Editor can also be used to open other files, such as *.len files, as text. In this context, you should note that the editor has a size limit of 32K characters, so some lens files may be too large to handle.

The OSLO Editor contains two unique features that make it especially useful for working with SCP and CCL:

- When you save a *.ccl file from the OSLO editor in a ccl directory, the file is automatically compiled and added to the built-in command list (assuming that the **ccac** preference is on).
- You can select a range of lines in the OSLO editor, and then use the **Execute Selection** command to execute the lines as an SCP command. This gives the editor the capability to handle multi-line command input. You just enter the commands (with semicolons at the end of each line), press CTRL+A to select them, and CTRL+E to execute them. The figure below shows a simple example of this feature. Four commands have been entered and selected; when CTRL+E is pressed, they will be executed as a group. Note that the last command is given with a question mark, which will cause OSLO to prompt for the command arguments.



Note that the editor contains its own menu, with just File and Edit items. This menu is independent of the main menu, and the entries are different tasks from the corresponding ones on the main menu, as described below.

In addition to being activated by the “Window >> Editor >> Open” menu option, the software editor User

>> Editor command, the Editor can also be activated by the fourth toolbar icon in the main window, or by the command.

Close

This menu item closes the OSLO Editor.

Std Prm UltraEdit

This menu item opens UltraEdit if UltraEdit is installed properly. The installation of UltraEdit is separate from your OSLO installation, but an UltraEdit installation program is available in the OSLO installation CD-ROM. When you first run UltraEdit you will be warned that it is a 45 day trial copy. If you purchased the Standard or Premium editions of OSLO, then your original OSLO shipment included a User Name and Authorization Number for unlimited use of a single copy of UltraEdit.

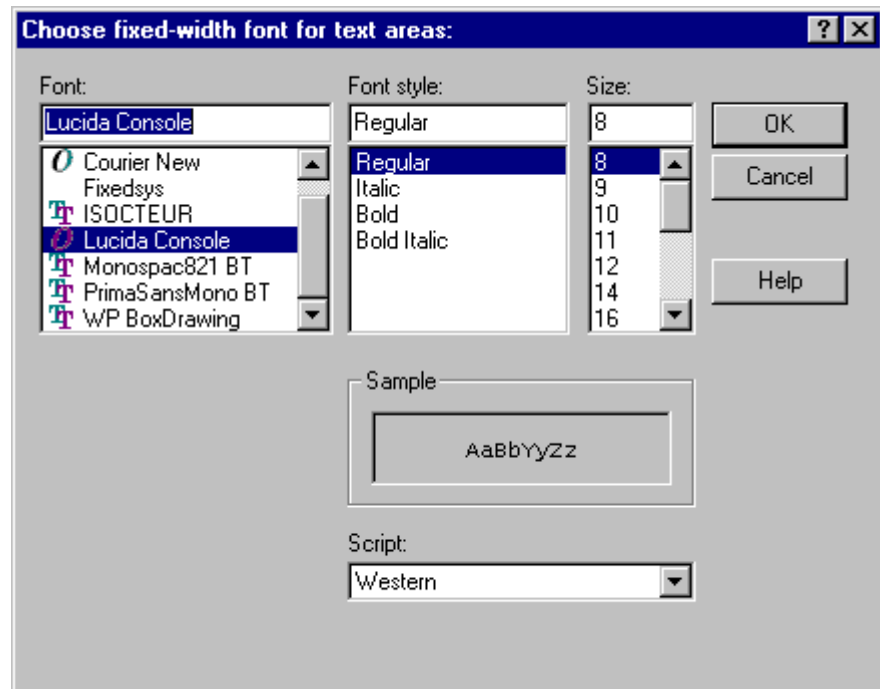
UltraEdit runs as a separate process from OSLO, although you can launch it from the main OSLO toolbar and this menu item. Because it is a separate process, CCL files are not automatically compiled when they are saved from UltraEdit. However, the toolbar icon next to the UltraEdit icon will carry out the required compilation. If you invoke UltraEdit after receiving a CCL compiler error, UltraEdit

will open the file in which the error occurred and place the cursor on the offending line.

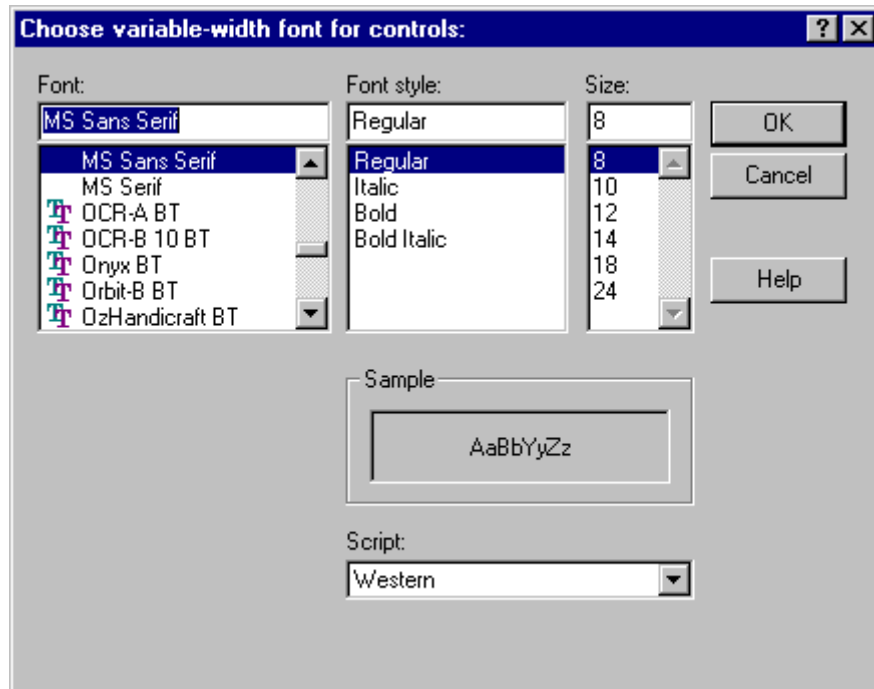
Choose Fonts

OSLO features the ability to customize the appearance of the application, including the ability to choose the fonts that are used in the text windows and in the various controls. There are two fonts used in OSLO: a fixed-width font for all text areas, and a variable-width font for all controls (e.g., push buttons, sliders, etc.). The default fonts are 8-point Courier New Regular for text areas, and 8-point MS Sans Serif Regular for controls.

To choose a new set of fonts for OSLO, select the Options >> Fonts menu command. The standard Windows Font Selection dialog will be displayed twice: once for choosing the text font...



and once for choosing the control font. (Note that a fixed-width font may be chosen for the controls, if desired.)



Changes in OSLO's fonts do not appear until OSLO is restarted.

Arrange Icons

The **Arrange Icons** menu item lines up the icons for any minimized OSLO windows along the bottom of the main OSLO window. It is sometimes helpful in finding a "lost" window that has become buried under other windows.

Status Bar

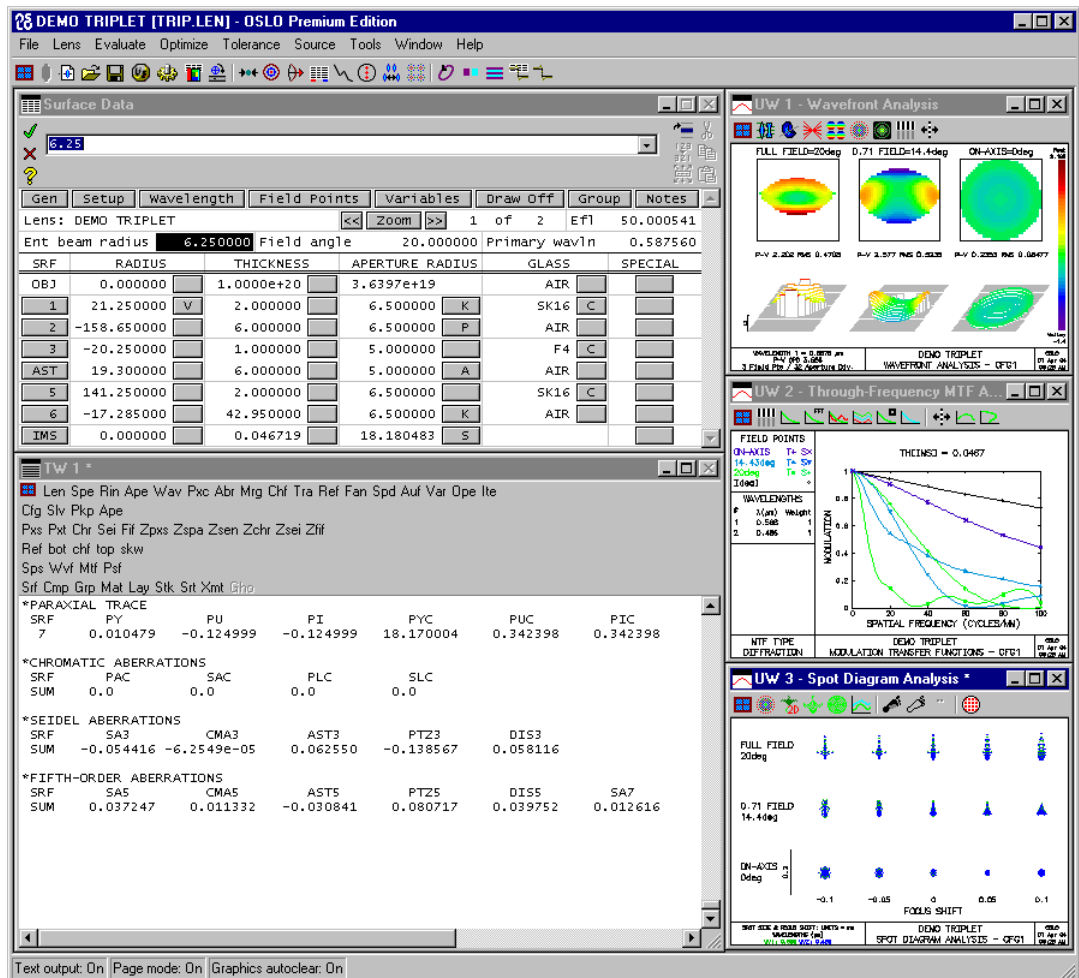
OSLO has the option of displaying a status bar at the bottom of its multiple-document-interface (MDI) client area. By default, the status bar is displayed, but the display may be suppressed by turning the **Show_status_bar** preference **off**.

Please see the section "Status Bar" on page 14 for a more detailed discussion of about the Status Bar.

Tile Windows

This menu item arranges the open OSLO windows in a pattern so that no windows overlap. The default tile command puts the spreadsheet (if any) at the upper left corner, the text windows below it, and the graphics windows in the remaining space on the right. The windows are resized to fill as much of the

OSLO window as possible, consistent with not allowing the graphics windows to become grossly distorted in aspect ratio.



A typical result of using the Tile command.

The tile command only considers open windows in determining the layout. Thus it is possible to achieve a new tile pattern by minimizing a window to an icon and re-tiling. In certain situations, the spreadsheet window may be covered by other windows. To raise the spreadsheet to the top, you can press the spacebar (or another key).

The tile command option has been implemented as a CCL command called "tile". Since the command is written in CCL, you can change it to meet your own requirements. Tiling is complicated in OSLO because there are different types and numbers of Windows that are used in a variety of different applications. The Tile command only deals with spreadsheets, graphics output, and text output windows. Special windows, such as the OSLO editor, the slider window and the catalog database, are not considered.

Overview

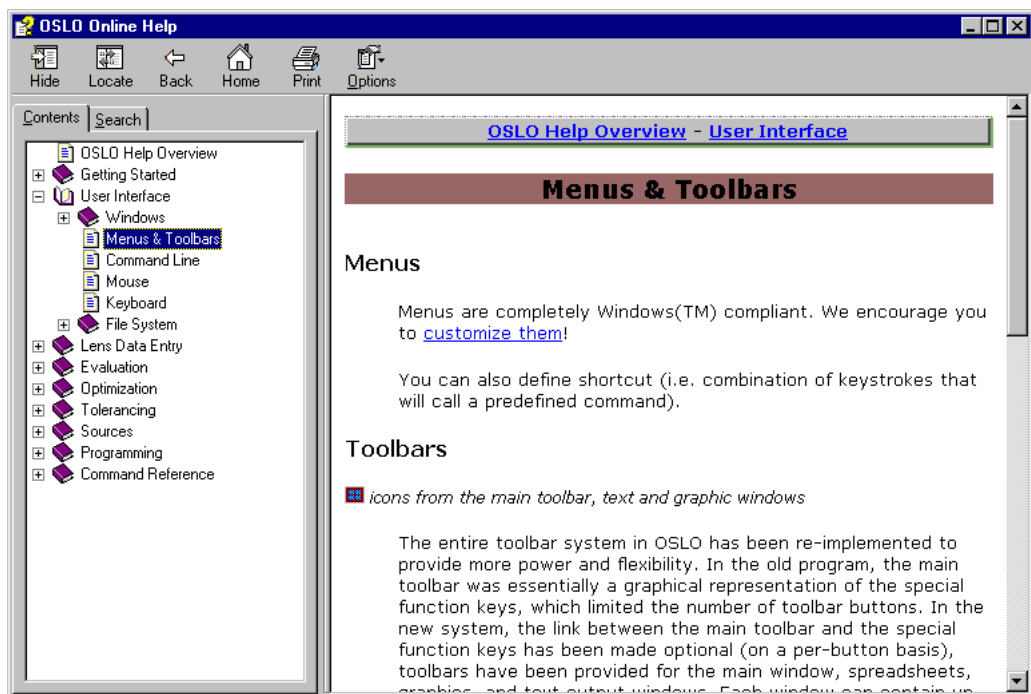
OSLO Help	F1
Tip of the Day	
About OSLO	

The Help menu provides access to the information that will aid you in fully utilizing the capabilities of OSLO.

The menu commands relating to optimization are:

Menu Item	Page
OSLO Help (F1) ... Open the OSLO on-line help system.	487
Tip of the Day... Open the Tip of the Day dialog Window.	488
About OSLO... Open the dialog that describes details about the OSLO version and the license agreement.	489

OSLO Help






OSLO Help is a context-sensitive system designed to be accessed directly from the program. Upon opening OSLO help, you will notice that the left hand viewer pane has two tabs: *Contents* and *Search*. The Contents tab on the viewer pane should be helpful as a general guide to learning about aspects of the program interface. Simply click on the appropriate portion of the outline tree to find out more information. If you want to know about a certain topic, use the Search tab to navigate to relevant parts of the help system. Using the Search tab, you can

locate every occurrence of a word or phrase. To improve the search results, combine multiple words or phrases with AND, OR, NEAR, and NOT.

OSLO Help is written using the popular HyperText Markup Language (HTML), and requires that Microsoft Internet Explorer 4.0 or higher be installed. Note that Internet Explorer does not need to be your default browser, but it does need to be installed on your operating system. Lambda Research is not allowed to offer Internet Explorer on our OSLO installation CD-ROM. You can download Internet Explorer from the Microsoft web site.

In addition to clicking the “Help>>OSLO Help” item on the main menu, there are three other ways to get help:

1. Each dialog box contains a help button that gives you additional information about its features. To find information, you can click a menu item containing an ellipsis (...), then press the help button  in the dialog box.
2. The spreadsheet command area has a help icon  that provides context-sensitive help for the spreadsheet that is currently open.
3. You can type in the name of a command in the command line. If you then click the help icon , you will see the command definition in the help window.

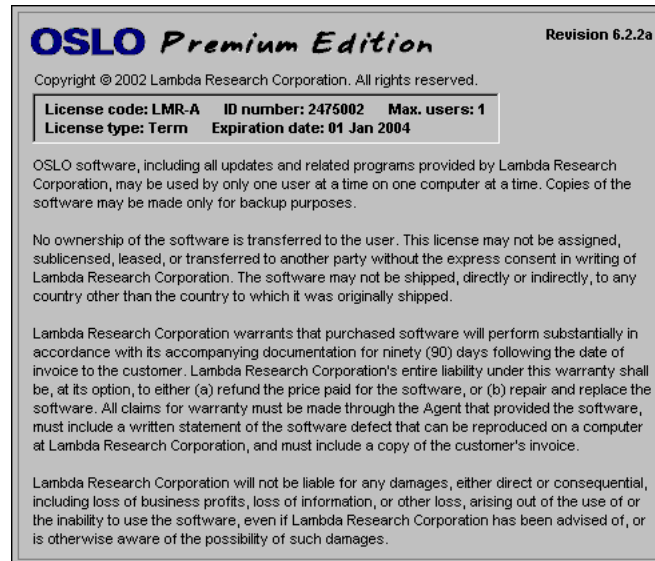
Tip of the Day



The “Tip of the Day” dialog box offers valuable tips for using OSLO. The list of tips are stored in a text file “ostips.txt”, which can be found in the “.../hlp” directory in the top-level OSLO installation directory. Each tip consists of a single line of up to 512 characters, and each is separated from the next tip by a blank line. Formatting of the tip into multiple lines is performed automatically by OSLO. You can add your own tips, messages, etc. by editing the file.

When OSLO is started, this same “Tip of the Day” dialog box is shown. If you do not wish to see these tips on startup, uncheck the “Show tips at startup” option in this dialog box. You can also turn tips on or off using the preference **show_tips_at_startup (stas)**.

About OSLO



The top of the “About OSLO” dialog box describes several important parameters that have to do with the operation and licensing of OSLO.

Edition

This item specifies whether you are using the Premium, Standard, Light or EDU edition of OSLO. For more information about the differences in operation and features available in all the different OSLO editions:

Read this manual. Notations are made next to features that are only found in certain editions.

Read through the OSLO on-line help. The OSLO help in every edition describes the features found in all editions, but notations are made for those features that are not available in the lower editions.

Download OSLO-EDU from the Lambda Research web site. OSLO-EDU is constructed in such a way that users can note what features are found in each edition through the use of edition-specific icons that are placed next to appropriate menu and toolbar items.

Go to the Lambda Research web site or otherwise contact Lambda Research for more information (e-mail preferred: sales@lambdares.com).

Revision

This reports the release number of the OSLO program. Updates that contain different release numbers after the first decimal point (i.e. 6.2, 6.3, 6.4, ...etc.) require that you be under current support contracts in order to install the program.

License Code

This “XXX-X” letter code is unique to each OSLO license and needs to be supplied to Lambda Research when discussing details of technical support and

licensing. The License Code does not appear in this dialog if the OSLO license is a network license.

ID number

The 7 digit ID Number, in part, dictates the Access Number that you enter when starting up a non-network OSLO license for the first time. If the number begins with a “1”, this means that the ID number (and resulting Access Number) are tied to a parallel port or USB port hardware keytag (dongle). If the number begins with a “2”, this means that OSLO is not making any communication with the hardware keytag. The ID Number does not appear in this dialog if the OSLO license is a network license.

Maximum Users

This item refers to the maximum number of users that can use this license. This is really only applicable to Network Licenses where more than one user can operate OSLO at a time.

License Type

Indicates the type of licensing that is being used for this installation of OSLO;

Perm & Permanent Network License

Indicates that the OSLO license is a permanent license that will not expire. You will be free to use the current release of OSLO indefinitely, even if you never purchase an additional support contract.

Term & Temporary Network License

Indicates that the OSLO license will expire and OSLO will be un-usable, after the Expiration Date.

Temporary licenses are typically issued to new users for 30-45 days before invoice payment can be verified. Once the invoice has been paid, a permanent license will be issued.

Expiration Date

If the OSLO license is temporary, this is the date after which OSLO will cease to work. You must obtain a new Access Number from Lambda Research in order to continue using OSLO after this date. The Expiration Date does not appear in this dialog if the OSLO license is permanent.

License Agreement

The text in the lower portion of the “About OSLO” dialog is the OSLO user license agreement. This is the same license agreement that users are asked to agree to during the OSLO installation process. The text of that agreement is also accessible here. When using OSLO, users are bound by this agreement.

In OSLO-EDU this section of the “About OSLO” dialog also contains an explanation of how “edition” icons are used to denote which features are available in different editions of OSLO.

Overview

Most of this manual describes the operation of OSLO using spreadsheets, menus, toolbars, ...etc. You can also run the program by entering commands. In fact, one of the features of the OSLO Click/Command interface is that you can mix the use of commands and spreadsheets to operate the program in whatever way you find the most productive.

OSLO commands are similar to functions in the C programming language, in that they carry out a sequence of actions and possibly return a value. A few extensions have been made to simplify the entry of commands from the keyboard, but OSLO command formats are basically a subset of C.

When you enter a command, you are actually running what is called a command-line procedure in the CCL programming language. CCL (Compiled Command Language) is one feature of the application manager (AppMan) that runs OSLO. It uses a high-speed compiler whose output is pseudo-code (*p-code*) that runs the program.

Working with Commands

Commands are entered by typing into the command line in the main window, into the OSLO Editor, or into a SmartCell in a spreadsheet. A command consists of a command name and zero or more arguments. The arguments are the data, either words or numbers, that the command needs to carry out its action. Some typical commands are

Short (informal)	Long (formal)
<code>rtg</code>	<code>surface_data(srf,0,0)</code>
<code>th 3 5.2</code>	<code>th(3,5.2)</code>
<code>obh -1e19</code>	<code>obh(-1e19)</code>
<code>pxt all</code>	<code>paraxial_trace(all,y,0,0)</code>
<code>drr 0 5 -1 1</code>	<code>draw_rays(0,5,-1,1,0,0)</code>
<code>ite ful 3</code>	<code>iterate(ful,3,std)</code>

Command Syntax

As mentioned above, OSLO commands are similar to C functions. Although formal C syntax would require that arguments be enclosed in parentheses and separated by commas (which is permitted), CCL permits a relaxed syntax for command-line procedures, in which parentheses are not required, arguments can be separated by blanks, and a terminating semicolon is not needed if you are executing a single command. If you use the relaxed syntax, however, you must use it completely, and not mix it with formal syntax.

Format

Most commands in OSLO have both a long form and a short form. Either one can be used to execute the command, but you will find it more convenient to type the short form (which is usually 3 or 4 letters long). The long form is more descriptive of the action of the command and is often used in writing programs. For example, the **ite ful 3** command listed above can also be entered as **iterate(full, 3)**.

Arguments

As mentioned above, the arguments of a command are qualifying words or numbers that define the specific action of a command. In command entry, numbers are usually entered directly (e.g., 6.28), but can also be entered as expressions (e.g., **2*Pi**). Word or string arguments cannot contain embedded blank spaces. If blank space is desired, the argument must be enclosed in quotation marks (e.g., **"Cemented Doublet"**). OSLO makes extensive use of *list* arguments, which are values displayed in an options list box that pops up when a value is required. The desired value can be entered by clicking on the element with the mouse, or by using the arrow keys to select the element and pressing either ENTER or SPACEBAR.

It is not always necessary for you to enter all the arguments of a command. Often, default values are prescribed and do not need to be supplied when you enter a command. This makes it much easier to work with commands on a routine basis. Of course, you can always enter explicit values for the command arguments.

If you fail to enter a value for an argument for which no default value is prescribed, OSLO will issue a prompt for you to enter a value. It is not possible to execute a command without entering a reasonable value for every argument.

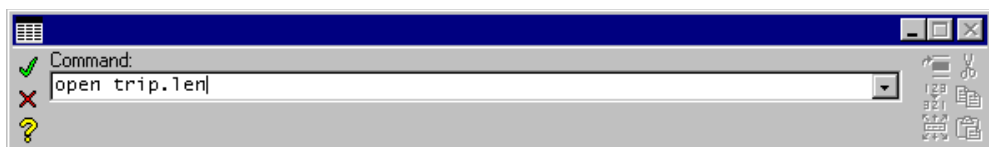
Question Mark Argument

A general problem in using commands is remembering the required arguments. Fortunately, OSLO has a special question mark argument, which makes it not necessary for you to remember the actual arguments. If you enter a command name followed by a question mark (e.g., **rtg ?**), the program will prompt you for all of its arguments.

This feature is useful not only when you can't remember the arguments for a command, but also when a command doesn't do what you expect. This is often caused by default values that are not appropriate to the current situation. By entering a command with a question mark, you can overrule these inappropriate values. Of course, you can achieve the same effect if you enter a command with all of its arguments.

The Command Line

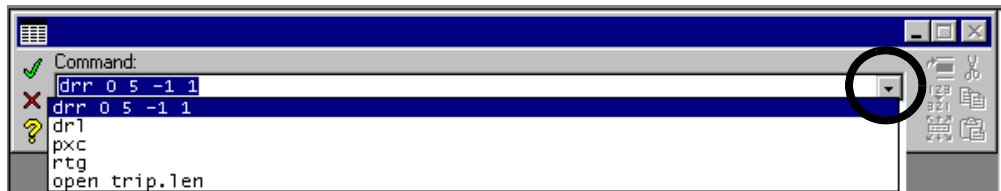
Most commonly, commands are keyed into the command line, a single line, scrollable text entry area that can accommodate commands up to 256 characters in length.



The command line can be edited in the usual way with backspace, delete, and the right and left arrow keys. However, the command line is slightly different from other Windows programs in that it is not coupled to the clipboard. Also, because it is integrated with OSLO spreadsheets (see SmartCells below), the handling of selected text is different. To edit selected text, you should first press the Insert key to remove the selection.

History Buffer

The history buffer holds the last 50 commands. You can scroll through these commands by pressing SHIFT+UP ARROW or SHIFT+DOWN ARROW. Alternately, you can click the combo box arrow at the right-hand end of the command line to drop down a list, as shown below.



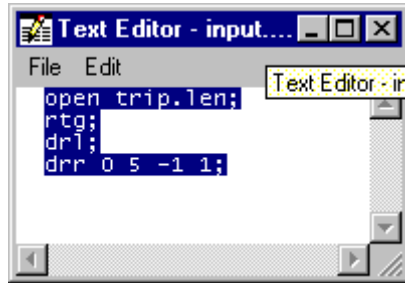
After you have opened the combo box with the mouse, you can scroll through the entries using the arrow keys.

Not everything that you type is entered into the history buffer. Command lines that duplicate the previous history buffer entry are not added to the history buffer. Also, command line entries that go directly into spreadsheet cells (rather than being executed) are not added to the history buffer. Commands selected from a menu and that bring up a dialog box are not added to the history buffer; only the command line generated by the program goes into the history buffer. This generated command line depends on the setting of the **Cmd_history_aliases (chal)** preference. If it is "on" (true), short command aliases will be used instead of long command names in the command strings that are built and put in the history buffer by OSLO. Thus if you choose "Evaluate>>Aberration Coefficients" from the OSLO menu and select "Seidel image" in the dialog box, you will find "seidel_abers ..." in the history if the above preference is "off", or "sei ..." if the preference is "on".

Control over the history buffer is provided by the **history_reset (hbr)** and **print_history (prh)** commands. **History_reset** clears the history buffer. **Print_history** prints the history buffer to the current text window.

OSLO Editor

A useful feature of OSLO is its ability to execute commands directly from its text editor. This is particularly useful when you want to execute a sequence of commands several times. Just enter the command in the Editor, as shown below, press CTRL+A to select them all, and then press CTRL+E to execute them.



When you enter command in the editor, you must append a semicolon to each line, as shown in the figure. Also, you can not have blank lines between commands.

SmartCells

Often, you want to execute a command when a spreadsheet is open, particularly the surface data spreadsheet. In order to accommodate this, OSLO spreadsheets use what are called SmartCells™, which parse the information entered into a data cell to decide whether it is a command or actual data. The portion of a spreadsheet shown below shows a sequence of commands being entered into a numeric data cell.

RADIUS	THICKNESS	APERTURE RADIUS
0.000000	1.0000e+20	3.6397e+19
21.250000	2.000000	6.500000 K
-158.650000	drr; drr 0 5	6.500000 P
-20.250000	1.000000	5.000000
19.300000	6.000000	5.000000 A

The scheme used by OSLO is generally reliable, but is not completely foolproof. For example, if you want to enter the name of a command in a string field, you must enclose it in quotes. Also, if you execute a command that returns a numeric value in a numeric field, the field will get the returned value (this is actually a feature, but it may or may not be what you want). To be safe, you can type the command into a button.

Some commands are not active and will not be executed when a spreadsheet is open. This is not a limitation of the SmartCell feature, but is rather needed to maintain data integrity, or to keep the program from crashing when incompatible actions are requested.

OSLO Commands

To locate a command name, you can use the tables below, or the on-line Help system. It is also possible to find a command name by looking in the history buffer after executing a menu command. There are more than one thousand commands in OSLO Premium, so it is not possible to remember all of them. Experienced users find, however, that it is not only possible but worthwhile to remember the most commonly used commands.

In the tables that follow, OSLO commands are listed together with a short description of the action of the command. In many cases, this description, together with the use of the question mark as mentioned above, will be sufficient

to enable your use of the command. To obtain more detail, you can look for the command in the index of this manual, or consult the on-line help system in the program.

The help system contains a complete listing of commands and their arguments (which are not shown here). The command listing in the OSLO help system is unique in that it is a listing of the OSLO command definitions. If there is a conflict between the description of a command in this manual and the one given in the help system, the help system prevails, since the program is actually compiled from that code. To understand the command definitions in the help system, you should read the ancillary information given there, or the Command Definitions section at the end of this chapter.

It is important to understand that commands in OSLO are associated with environments defined within the program, which keep commands from being executed when they are not valid. For example, you cannot print the graphics window when there is no graphics window. In menu mode, invalid commands are “grayed out.” In command mode, an attempt to run an invalid command will produce an error message.

To help you locate commands, the commands available to users are listed below. The tables have been organized in 6 groups:

Data I/O and Display	Files and printing, text display of lens data, lens drawings, general program control.
Optical Analysis	Aberrations, ray tracing, image analysis routines.
Optimization	Set up optimization data, iteration, run ASA.
Lens Editing	Commands that control the lens editor.
Surface Data	Commands that set values for surface data.
Operating Conditions	Commands that set values for system data.

Cmd	Long form	Data I/O and display commands
ado	aspheric_data_ol ld	Lens data; Displays conic and polynomial asphere (ad, ae, af, ag) data.
ape	aperture_data	Lens data; Displays aperture radius and special aperture data.
bda	boundary_drawing_data	Lens data; Displays boundary drawing information data.
ccl	compile	Invokes CCL compiler.
cfg	configuration_edit	Lens data; Displays configuration data.

chd	change_directory	Changes current directory to a new directory.
chf	choose_fonts	Changes fonts used for text display and output in Windows versions only.
	close_movie	Close “movie” file.
cor	coord_transform	Lens data; Displays coordinate transform matrix of a surface.
ctg	surface_cv_data	Lens data; Displays curvature, thickness, aperture radius, and glass data.
ddr	draw_default_rays	Lens drawing; Draw default rays on current lens drawing.
deh	delete_error_handler	Deletes current error handler for CCL and SCL-Plus commands.
dir	directory	Displays directory listing.
dph	diff_surf_phase	Lens data; Displays phase and effective grating spacing for diffractive surface.
dre	draw_element	Open spreadsheet for editing element drawing data.
drg	draw_graphics	Lens drawing; Add additional graphics and labels to current lens drawing.
drl	draw_lens	Lens drawing; Draw layout of current lens system.
drr	draw_rays	Lens drawing; Draw ray trajectories on current lens drawing.
dsz	diff_surf_zones	Lens data; Displays diffractive zone boundaries for rotationally-symmetric CGH diffractive surfaces.
dxg	dxg_lens_drawing	Export current lens and default ray data to a DXF file.
era	erase	Deletes a file.
eth	edge_thickness	Lens data; Computes the distance between points on two surfaces.
ewc	editwin_close	Close the text editor window.
ewo	editwin_open	Open the text editor window.
exe	execute	Execute the supplied string as a command.
exi	exit	Stops and exits the program.
gcl	gclear	Clears contents of current graphics window.
gcp	graphwin_copy	Copies contents of current graphics window to the clipboard (Windows version only).
gcu	glass_cat_update	Enter glass data in private catalog.

giv	grad_index_value	Lens data; Computes refractive index value at arbitrary (x, y, z) point for a gradient index medium.
glc	glass_cat_list	Displays contents of glass catalog.
gmb	graphwin_modalbtn	Set modal operation of mouse button for selecting graphics window zoom center.
grn	grad_index_data	Lens data; Displays gradient index medium data.
gwc	graphwin_close	Closes a graphics window.
gwe	graphwin_setscale_execute	Sets execution command, with a fixed scale for graphics output, for an updatable graphics window.
gwo	graphwin_open	Opens a graphics window.
gwp	graphwin_print	Prints contents of graphics window.
gwr	graphwin_reset	Clears a graphics window and removes sliders.
gwt	graphwin_title	Adds title to graphics window title bar.
gwu	graphwin_update	Updates selected updatable graphics window.
gwx	graphwin_execute	Sets execution command for updatable graphics window.
gwz	graphwin_zoom	Zoom contents of current graphics window.
hbr	history_reset	Clears contents of command history buffer.
help		Opens OSLO help window.
hoe	holographic_data	Lens data; Displays holographic optical element surface data.
hpw	hpgl_write	Save the contents of the current graphics window as an HP-GL file.
ieh	install_error_handler	Installs error handler for CCL and SCL-Plus commands.
igs	iges_lens_drawing	Export current lens and default ray data to an IGES file.
	import	Import a Genii, Code V, or Zemax lens into OSLO.
lar	lens_array_data	:Lens data; Displays lens array data.
lis	list_file	Displays contents of file in current text window.
	make_movie	Create “movie” of graphics window frames, described by CCL command.
mkd	make_directory	Makes a new directory.
nad	nss_action_data	Lens data; Displays special surface actions for non-sequential group surfaces.
opc	operating_conds	Lens data; Displays operating conditions data.
open		Opens a file.

	open_movie	Opens a “movie” file.
pas	aspheric_data	Lens data; Displays the aspheric surface data (except for standard asphere).
pdf	diffractive_data	Lens data; Displays diffractive surface data.
pei	eikonal_surface _data	Lens data; Displays eikonal surface data.
pkp	pickup_data	Lens data; Displays surface data constrained by pickups.
ppe	polarz_elmnt_data	Lens data; Displays polarization element data.
ppf	perfect_lens_data	Lens data; Displays perfect lens data.
prh	print_history	Displays contents of command line history buffer in current text window.
psp	spline_surface_data	Lens data; Displays radial spline surface data.
pzr	zernike_surface _data	Lens data; Displays Zernike phase surface data.
qui	quit	Stops and exits the program.
rin	refractive_indices	Lens data; Displays refractive index data.
rmd	remove_directory	Removes a directory.
rtg	surface_data	Lens data; Displays radius, thickness, aperture radius, and glass data.
sag	surface_sag	Lens data; Computes the sag of a surface at a specified (x, y) point.
save		Saves current lens data in a file.
	save_frame	Save contents of current graphics window as a “movie” frame.
sbr	ssbuf_reset	Resets spreadsheet buffer row number.
sfn	surface_notes	Lens data; Displays the surfaces notes.
sft	surface_tag_data	Lens data; Displays the toric, grating, Fresnel, and alternate intersection surface data.
	show_movie	Display “movie” created by make_movie command.
shp	show_preference	Displays the value of an OSLO preference.

sle	select_lens_catalog	Sets current lens catalog database.
slv	solve_data	Lens data; Displays surface data specified by paraxial solves.
spe	special_data_types	Lens data; Displays special data on each surface.
ssb		Get value in row i , column j of the spreadsheet buffer, i.e. $ssb(i,j)$.
stp	set_preference	Changes the value of an OSLO preference.
tcp	textwin_copy	Copies contents of text window to the clipboard.
tdc	tilt_decenter_data	Lens data Displays the tilt and decenter (i.e., coordinates) surface data.
tol	tolerances	Lens data; Displays the surface tolerance data.
twc	textwin_close	Closes a text window.
two	textwin_open	Opens a text window.
twp	textwin_print	Prints contents of text window.
twr	textwin_reset	Clears contents of text window.
twt	textwin_title	Adds title to text window title bar.
uda	update_all	Update all updatable graphics windows.
use	user_defined_data	Lens data; Displays user ray trace surface data.
wav	wavelength_data	Lens data; Displays the wavelength data or sets the current wavelength.
xeq	execute_scpfile	Execute SCL-Plus command..

Optical Analysis

The optical analysis commands are ones that are used to evaluate the lens. Most of the commands correspond to items accessed through the Calculate or Options menus.

Cmd	Long form	Optical analysis command
buc	buchdahl_abers	Aberrations; Displays the fifth-order aberration coefficients in H. A. Buchdahl format.
chr	chromatic_abers	Aberrations; Displays the first-order chromatic aberration coefficients.
dfe	diff_rad_energy _dist	Evaluation; Computes the diffraction-based radial energy distribution, using direct integration to compute the PSF.
dlk	diff_lsf_ked	Evaluation; Computes the diffraction-based line spread function and knife edge distribution, using an FFT to compute the PSF.
eed	ensq_energy_dist	Evaluation; Computes the geometric ensquared energy distribution.
eeq	ensquared_energy _quick	Evaluation; Computes the diffraction-based ensquared energy distribution, using an FFT to compute the PSF.
exs	exspd	Evaluation; Computes an image-space spot diagram.
fbc	fiber_coupling	Evaluation; Computes the fiber coupling efficiency.
fif	fifth_order_abers	Aberrations; Displays the fifth-order aberration coefficients in Seidel format.
gbm	gaussian_beam	Gaussian beam; Opens the Gaussian beam spreadsheet.
ini	incoh_image	Partial coherence; Computes incoherent image.
lke	lsf_knife_edge	Evaluation; Computes geometric line spread function and knife edge distribution.
mst	mtf_sagittal _tolerance	Tolerancing; Perform sagittal MTF tolerancing, using analytic wavefront differentials.
mtf	mod_trans_func	Evaluation; Computes the modulation transfer function.
mtt	mtf_tangential _tolerance	Tolerancing; Perform tangential MTF tolerancing, using analytic wavefront differentials.
pci	partcoh_image	Partial coherence; Computes partially coherent image.
pdl	plot_diff_lsf_ked	Evaluation; Plot diffraction-based line spread function and/or knife edge distribution.

pec	plot_equal_contours	Evaluation; Plot equally spaced contour levels using current PSF grid.
ped	plot_ensq_energy_dist	Evaluation; Plot geometric ensquared energy distribution.
pgl	plot_geom_lsf_ked	Evaluation; Plot geometric line spread function and/or knife edge distribution.
pgq	psf_grid_quick	Evaluation; Construct grid of point spread function values for contour plot, using an FFT to compute the PSF.
pii	plot_incoh_image	Partial coherence; Plots incoherent image in x and y.
pla	plot_long_aber	Ray tracing; Plots the longitudinal spherical aberration.
plc	plot_contours	Evaluation; Plot specified contour levels using current PSF grid.
pld	plot_distortion	Ray tracing; Plots the exact ray distortion.
ple	plot_rad_energy_dist	Evaluation; Plots the radial energy distribution.
plf	plot_field_curves	Ray tracing; Plots the Coddington field sags.
pll	plot_lateral_color	Ray tracing; Plots the exact ray lateral color curves.
plm	plot_mod_transfer_func	Evaluation; Plots the modulation transfer function.
plo	plot_opd_fans	Ray tracing; Plots optical path difference versus pupil coordinate curves for the current object point.
plp	plot_point_spread_func	Evaluation; Plots the point spread function, using direct integration to compute the PSF.
plr	plot_ray_intercept_pts	Ray tracing; Plots ray intercept versus pupil coordinate curves for the current object point.
pls	plot_spot_diagram	Evaluation; Plots the current spot diagram.
plw	plot_wavefront	Evaluation; Plots an OPD map or contour plot of the emerging wavefront.
ppi	plot_partcoh_image	Partial coherence; Plots partially coherent image in x and y.
ppq	plot_psf_quick	Evaluation; Plots the point spread function, using an FFT to compute the PSF.

prq	plot_radeng_quick	Evaluation; Plots the diffraction-based radial energy distribution, using an FFT to compute the PSF.
prs	print_spot_diagram	Evaluation; Displays a listing of the coordinates of the rays in the current spot diagram.
pse	paraxial_setup	Paraxial; Opens the paraxial setup spreadsheet.
psf	point_spread_function	Evaluation; Displays the value of the point spread function.
psq	plot_ensqeng_quick	Evaluation; Plots the diffraction-based ensquared energy distribution, using an FFT to compute the PSF.
pss	point_spread_scans	Evaluation; Plot x and y cross sections of point spread function.
ptn	polarization_trans	Polarization; Computes transmittance of beam from current object point.
pup	pupil_abers	Aberrations; Displays Seidel pupil aberration coefficients.
pxc	paraxial_constants	Paraxial; Displays the paraxial constants.
pxs	paraxial_setup_data	Paraxial; Displays the current paraxial setup data.
pxt	paraxial_trace	Paraxial; Displays paraxial ray trace data.
rdg	rs_diffraction_grid	Evaluation; Construct grid of diffraction irradiance values for contour plot, using Rayleigh-Sommerfeld diffraction.
rds	rs_diffraction_scans	Evaluation; Plot x and y cross sections of diffraction irradiance, using Rayleigh-Sommerfeld diffraction.
red	rad_energy_dist	Evaluation; Displays the geometrical radial energy distribution.
req	radial_energy_quick	Evaluation; Displays the diffraction-based radial energy distribution, using an FFT to compute the PSF.
ria	rel_illum_approx	Evaluation; Displays the relative illumination at the current object point, computed using an elliptical pupil approximation.
ril	rel_illumination	Evaluation; Displays the relative illumination at the current object point, computed using an image space ray grid.
ris	rel_illum_scan	Evaluation; Displays the relative illumination for a series of points across the object, computed using an image space ray grid.
	rptg_fans	Evaluation; Displays ray analysis report graphics.

	rptg_mtfs	Evaluation; Displays through-frequency MTF report graphics.
	rptg_psfs	Evaluation; Displays point spread function report graphics.
	rptg_spots	Evaluation; Displays spot diagram report graphics.
	rptg_tf_mtfs	Evaluation; Displays through-focus MTF report graphics.
	rptg_waves	Evaluation; Displays wavefront report graphics.
rsa	rel_scan_approx	Evaluation; Displays the relative illumination for a series of points across the object, computed using an elliptical pupil approximation.
rsd	rs_diffraction	Evaluation; Displays the diffraction irradiance at a point in image space, using Rayleigh-Sommerfeld diffraction.
rsw	rs_wavefront_trace	Evaluation; Constructs wavefront for Rayleigh-Sommerfeld diffraction calculations.
sdd	spot_diagram_data	Evaluation; Displays information about the current spot diagram.
sei	seidel_abers	Aberrations; Displays Seidel image aberration coefficients.
sfg	spread_function_grid	Evaluation; Construct grid of point spread function values for contour plot.
sop	set_object_point	Ray tracing; Establishes a new object point and traces a reference ray from that object point.
spd	spot_diagram	Evaluation; Computes a spot diagram for the current object point.
sps	spot_sizes	Evaluation; Computes geometric spot sizes for the current spot diagram.
tai	airspace_sensitivity	Tolerancing; Computes change table for air space perturbations.
tcca	comp_cctilt_sensitivity	Tolerancing; Computes change table for component tilt about the center of curvature of the outer component surfaces.
tcv	curvature_sensitivity	Tolerancing; Computes change table for surface curvature perturbations.
td	decenter_sensitivity	Tolerancing; Computes change table for surface decentration perturbations.
tda	comp_dec_sensitivity	Tolerancing; Computes change table for component decentration perturbations.
tfaa	comp_fatilt_sensitivity	Tolerancing; Computes change table for component tilt about the edge of outer component surfaces free aperture.

tgb	trace_gaussian_beam	Gaussian beam; Trace astigmatic Gaussian beam along current reference ray.
tig	irreg_sensitivity	Tolerancing; Computes change table for surface irregularity perturbations.
tis	tol_inverse_sens	Tolerancing; Perform user-defined tolerancing in inverse sensitivity mode.
tra	trace_ray	Ray tracing; Traces a ray from the current object point.
trd	trace_ray_derivatives	Ray tracing; Traces a reference ray and displays pupil and object derivative data.
tre	trace_ray_efficiency	Ray tracing; Traces a ray and computes local diffraction efficiency along the ray for diffractive surfaces.
trf	trace_fan	Ray tracing; Displays ray-intercept data for a fan of rays from the current object point.
trg	trace_ray_generic	Ray tracing; Trace a “generic” ray (both field and aperture are specified).
trn	index_sensitivity	Tolerancing; Computes change table for index of refraction perturbations.
trr	trace_ref_ray	Ray tracing; Establishes a new object point and traces a reference ray.
trs	trss	Tolerancing; Compute tolerances for equal contributions to rss value from change table.
tru	radius_sensitivity	Tolerancing; Computes change table for radius of curvature perturbations.
tsn	tol_sensitivity	Tolerancing; Perform user-defined tolerancing in direct sensitivity mode.
tss	ax_shift_sensitivity	Tolerancing; Computes change table for axial surface shift perturbations.
tt	tilt_sensitivity	Tolerancing; Computes change table for surface tilt perturbations.
tth	thickness_sensitivity	Tolerancing; Computes change table for element thickness perturbations.
ttr	tolerance_threshold	Tolerancing; Sets threshold for reporting of change table entries.
tun	tunits	Tolerancing; Scales tolerance units for change tables.
wft	wavefront_tolerance	Tolerancing; Perform rms wavefront tolerancing, using analytic wavefront differentials.

wvf	wavefront	Evaluation; Displays data describing the wavefront for the current spot diagram.
zer	zernike_fit	Evaluation; Compute Zernike polynomial decomposition of wavefront.

Optimization

Optimization commands set up optimization data and carry out the optimization process.

Cmd	Long form	Optimization command
asa		Use ASA global optimization algorithm with current lens and error function.
auf	autofocus	Adjust image surface thickness for best focus.
end		Exit current command-mode optimization data editor (ray set, spot diagram set, operands, or variables)
f		Enter or update a field point.
fpg	fieldpoints_generate	Generate a new set of field points.
fse	fst_spreadsheet	Open field points spreadsheet editor.
ide	interactive_design	Open interactive design setup spreadsheet editor.
ite	iterate	Perform damped-least-squares optimization using current merit function and variables.
	kill_asa	Terminate background ASA process.
mdu	matrix_dump	Display derivative matrix for minimize-mode or constraint operands.
o		Enter or update an operand.
ope	operands	Display current operands or enter command-mode operands editor.
opg	operands_generate	Generate a new error function.
ose	ope_spreadsheet	Open operands spreadsheet editor.
r		Enter or update a ray in ray set.
rse	rst_spreadsheet	Open ray set spreadsheet editor.
rst	rayset	Display current field points and ray set or enter command-mode field points and ray set editor.
	run_asa	Run background ASA process using a base file.

s		Enter or update a spot diagram.
sde	sds_spreadsheet	Open spot diagram set spreadsheet editor.
sds	spdset	Display current spot diagram set or enter command mode spot diagram set editor.
	start_asa	Start background ASA process.
	stop_asa	Halt background ASA process.
v		Enter or update a variable.
var	variables	Display variables or enter cmd-mode variables editor.
vse	var_spreadsheet	Open variables spreadsheet editor.
vya	vary_all	Denote all curvatures, thickness, or air spaces as optimization variables.
wvg	wavelengths_generate	Generate a new set of wavelengths.

Lens editing

There are three or four ways to edit lens data in OSLO, depending on how you count them. The first two are file input and lens editor input, which are variations of the same thing, the only difference being that in the first case, lens input commands are saved in a text file that is then read into the lens editor by the program, while in the second case, the editor is activated, and lens input commands are keyed into the command line. In both cases, the scope of the program is restricted to lens input commands while the lens editor is open. The lens is set up once, when the lens editor is closed.

The third way to enter lens data in OSLO is to use *global* editing. In this scheme, commands are entered one at a time, and the lens is set up after each command. While this is not the most efficient way to update a large amount of data, it has an advantage that other commands (e.g., analysis or display) can be mingled with lens input commands. The following table compares the way in which the third thickness of a lens would be set to the value 6.0 using normal input vs. global editing.

Normal lens input	Global editing
len upd gto 3 th 6.0 end	th 3 6.0

The fourth way to enter lens data is to use a spreadsheet. Spreadsheets are different from the other methods for entering data, because they do not generate commands. Instead, a single command is given to invoke the spreadsheet, which

then manages the data input directly with its memory representation. OSLO spreadsheets are, for the most part, *modeless*, which means that commands can be executed while a spreadsheet is open. Depending on the spreadsheet, some commands are disabled to prevent corruption of data or misleading displays.

The table below shows the commands that relate to the various forms of lens data input.

Cmd	Long form	Lens editing command
c		Command line configuration data global editing command.
cae	clear_all_element_data	Deletes all element drawing data from the lens.
cdb	catalog_database	Opens catalog lens database.
cfg	configuration_edit	Enters configuration data editor, displays current configuration data, or changes configurations.
cse	configuration_spreadsheet	Opens configuration data spreadsheet editor.
del		Delete range of surfaces.
edd	element_drawing_defaults	Opens element drawing defaults spreadsheet editor.
end		Leaves command mode lens editor and resets image surface number.
grp	group	Combines selected surfaces into an element group.
gto		Resets surface pointer in command mode lens editor.
ins		Insert surface.
inv	invert	Inverts range of surfaces
len	lens	Enters command mode lens editor.
lse	lens_spreadsheet	Enters spreadsheet lens surface data editor.
mer	merge	Merges data from lens file into current lens.
	merge_catalog	Opens catalog lens database in order to merge catalog lens into current lens.
nxt	next	Increments surface number in command mode lens editor.
rev		Reverses the data describing a range of surfaces.
scf	switch_configuration	Switches to specified configuration number.

scg	set_configuration	Switches to specified configuration number.
scl	scale_lens	Scales the focal length, f -number, Gaussian image height, or a range of surfaces.
sle	scale_lens_efl	Scales the focal length of the lens.
tse	tolerance_spreadsheet_editor	Open tolerance data spreadsheet editor.
ung	ungroup	Removes group designation from selected group.
uoc	update_operating_conditions	Open operating conditions spreadsheet editor.
wse	wavelengths_spreadsheet	Open wavelengths and weights spreadsheet editor.

Lens Surface Data

Most commands in OSLO merely set the value of a particular item of lens data. For example, the command **th 2 4.5** sets the thickness of the second surface to 4.5. This section enumerates the commands that set up surface data. Lens data that apply to the lens as a whole are called operating conditions, which are shown in the next group.

Lens surface data commands do not have any long forms.

Cmd	Lens surface data command
aac	special aperture action
aan	special aperture azimuthal angle
ach	data for single channel of tabular array
ad	4 th order aspheric coefficient
ae	6 th order aspheric coefficient
af	8 th order aspheric coefficient
ag	10 th order aspheric coefficient
agn	special aperture group number
aif	glass to air and mark as fixed
air	glass to air
al	paraxial axial ray aplanatic solve
alc	paraxial chief ray aplanatic solve
ap	aperture radius

apd	deletes special aperture data
apf	aperture radius; mark as fixed
apk	special aperture pickup
apn	number of special apertures
ard	deletes lens array data
ary	lens array data
asi	aspheric surface coefficient i
asai	ISO aspheric surface A coefficient i
asbi	ISO aspheric surface B coefficient i
asci	ISO aspheric surface C coefficient i
asdi	ISO aspheric surface D coefficient i
asi	alternate surface intersection flag
asp	aspheric surface type
atd	deletes aspheric surface data
atp	special aperture type
avxi	special aperture x vertex i
avyi	special aperture y vertex i
ax1	special aperture minimum x -coordinate
ax2	special aperture maximum x -coordinate
ay1	special aperture minimum y -coordinate
ay2	special aperture maximum y -coordinate
bdd	deletes boundary drawing information
bdi	boundary drawing information
bed	deletes tilt-and-bend
ben	tilt-and-bend surface
cc	conic constant
ccx	xz plane conic constant
cns	ISO cone surface coefficient
cnx	ISO cone surface x coefficient
cny	ISO cone surface y coefficient
csd	deletes curvature solve or pickup
ctf	curvature from test glass radii list
cv	curvature
cvf	curvature; mark as fixed
cvx	toric curvature
cx	deletes toric curvature
dct	decentration tolerance

dcx	x decentration
dcy	y decentration
dez	z decentration
ddd	deletes diffractive surface data
dfi	diffractive surface coefficient i
doe	diffractive surface type
dor	diffractive surface diffraction order
drw	surface drawing control
dt	decenter-tilt order
dth	gradient index medium step size for ray tracing
dwv	diffractive surface design wavelength
ear	end of array surface
ec	edge contact thickness solve
eik	type and CCL command name for eikonal surface
eii	eikonal surface coefficient i
eli	element number for non-sequential surface
esd	deletes eikonal surface data
fcc	Fresnel surface substrate conic constant
fcv	Fresnel surface substrate curvature
fgl	fix model or direct index glass to nearest catalog glass
frn	Fresnel surface flag
gbo	linear grating blaze order
gc	global coordinate reference surface
gcd	deletes global coordinate specification
gdd	deletes gradient index medium data
gdp	linear grating groove depth
gdt	gradient index medium type
gla	glass name (and indices if necessary)
glf	glass name; mark as fixed
gmz	GRADIUM gradient blank thickness
gnzi	GRADIUM gradient z^i coefficient
gor	grating diffraction order
goz	GRADIUM gradient axial offset into blank

grai	GRADIUM gradient chromatic coordinate dispersion data α coefficient i
grbi	GRADIUM gradient chromatic coordinate dispersion data β coefficient i
grci	GRADIUM gradient chromatic coordinate dispersion data γ coefficient i
grdi	GRADIUM gradient chromatic coordinate dispersion data δ coefficient i
gww	GRADIUM gradient dispersion data reference wavelength
gsp	grating spacing
hod	deletes hologram data
hor	hologram diffraction order
hv1	hologram source point 1 real/virtual factor
hv2	hologram source point 2 real/virtual factor
hwv	hologram construction wavelength
hx1	hologram source point 1 x -coordinate
hx2	hologram source point 2 x -coordinate
hy1	hologram source point 1 y -coordinate
hy2	hologram source point 2 y -coordinate
hz1	hologram source point 1 z -coordinate
hz2	hologram source point 2 z -coordinate
idd	deletes Zernike surface data
irt	irregularity surface form tolerance
jaa	polarization element row 1, column 1 amplitude
jab	polarization element row 1, column 2 amplitude
jac	polarization element row 2, column 1 amplitude
jad	polarization element row 2, column 2 amplitude
jpa	polarization element row 1, column 1 phase
jpb	polarization element row 1, column 2 phase
jpc	polarization element row 2, column 1 phase

jpd	polarization element row 2, column 2 phase
jsd	deletes polarization element data
kco	diffractive surface design construction order
kdp	diffractive surface kinoform zone depth
lme	last surface of surface group
lmn	name of surface group
lmo	first surface of surface group
nac	non-sequential surface special action
not	adds surface note
nr1	gradient index medium r^2 or r'^2 coefficient.
nr2	gradient index medium r^4 or r'^4 coefficient.
nr3	gradient index medium r^6 or r'^6 coefficient.
nr4	gradient index medium r^8 or r'^8 coefficient.
nrx	elliptical gradient x-coefficient
nry	elliptical gradient y-coefficient
nz1	gradient index medium z coefficient
nz2	gradient index medium z^2 coefficient
nz3	gradient index medium z^3 coefficient
nz4	gradient index medium z^4 coefficient
pf	boundary drawing plane face
pdf	deletes perfect lens data
pfl	perfect lens focal length
pfm	perfect lens magnification for perfect imagery
pi	paraxial axial ray angle of incidence solve
pic	paraxial chief ray angle of incidence solve
pk	curvature, thickness, aperture, glass, tilt/decenter or diffractive surface pickup
pu	paraxial axial ray angle solve
puc	paraxial chief ray angle solve
py	paraxial axial ray height solve
pyc	paraxial chief ray height solve

rcd	deletes coordinate return specification
rco	coordinate return specification
rd	radius of curvature
rdf	radius of curvature; mark as fixed
rdt	radius of curvature tolerance
rdx	toric radius of curvature
rfh	glass to reflector; hatch markings in lens drawings
rfl	glass to reflector (mirror)
rnt	refractive index tolerance
rod	extruded surface specification
rtf	radius of curvature from test glass radii list
sgc	distance to center of spherical gradient symmetry
shi	radial spline surface zone height i
skd	deletes skip surface designation
skp	skip surface designation
spl	number of radial spline surface zone.
spt	spherical surface form tolerance
ssd	deletes radial spline surface data
ssi	radial spline surface zone slope i
sur	radius of curvature, thickness, glass, and aperture radius
sva	amplitude of sinusoidal GRIN variation
svf	phase of sinusoidal GRIN variation
svp	period of sinusoidal GRIN variation
tao	offset of linear taper GRIN
tas	slope of linear taper GRIN
tce	thermal coefficient of expansion
tdd	deletes tilt and decenter (coordinates) data
th	thickness
tht	thickness tolerance
tir	total internal reflection only specification
tla	tilt around x -axis
tlb	tilt around y -axis
tlc	tilt around z -axis
tlt	tilt tolerance

tsd	deletes thickness solve or pickup
ugc	name of CCL command for user-defined gradient
ugri	user gradient index medium coefficient <i>i</i>
unm	name of CCL command for user ray trace surface
usd	deletes user ray trace surface data
usr	number of items on user ray trace surface
uti	user ray trace surface coefficient <i>i</i>
vxi	boundary drawing vertex
zoe	type of Zernike surface
zri	Zernike surface coefficient <i>i</i>

Operating Conditions

Operating conditions are data that describe the overall optical system, its environment, or a condition of evaluation that is particular for the lens. These data are saved with the lens.

Cmd	Long form	Operating condition
afo		Sets evaluation mode (focal or afocal).
amo		Sets aberration mode.
ang		Sets half-field angle.
apck	aperture_check	Sets aperture checking in ray trace.
ast		Sets aperture stop surface number.
	callback	Invokes autodraw in lens surface data spreadsheet editor.
cslv	config_solves	Sets solve behavior in alternate configurations.
des		Sets designer name.
dlap	drl_apers	Style of apertures used for wire-frame and solid lens drawings.
dlas	drl_aperstop	Controls drawing of aperture stop location in plan view lens drawings.
dlcv	drl_cadview	Style of data exported to DXF and IGES files.
dlfd	drl_finaldist	Distance to reserve in image space for drawing ray trajectories.
dlfp	drl_yfieldpt	Fractional <i>y</i> object height for default drawing rays.
dlfs	drl_firstsrf	First surface of drawn surface range.

dlha	drl_horzang	Horizontal view angle for wire-frame and solid lens drawings.
dlhr	drl_hatchrefl	Controls drawing of “hatch” marking on back side of reflecting surfaces in plan view lens drawings.
dlid	drl_initdist	Distance to reserve in object space for drawing ray trajectories.
dlls	drl_lastsurf	Last surface of drawn surface range.
dlmn	drl_minpup	Minimum fractional pupil coordinate for default drawing rays.
dlmx	drl_maxpup	Maximum fractional pupil coordinate for default drawing rays.
dlnf	drl_nbrfpts	Number of field points for default drawing rays.
dlnr	drl_nbrrays	Number of rays from field point for default drawing rays.
dlos	drl_offset	Offset in pupil for default drawing rays.
dlri	drl_rings	Number of aperture rings for solid lens drawings.
dlrs	drl_raystosrf	Control of drawing of image space ray trajectories.
dlsp	drl_spokes	Number of aperture spokes for solid lens drawings.
dlva	drl_vertang	Vertical view angle for wire-frame and solid lens drawings.
dlwn	drl_wavelnum	Wavelength number for default drawing rays.
dlxf	drl_xfieldpt	Fractional x object height for default drawing rays.
dlxs	drl_xshift	x -direction shift of lens drawing in graphics window.
dlyf	drl_ypupfan	Default drawing rays fan direction: On for y fan, Off for x fan
dlys	drl_yshift	y -direction shift of lens drawing in graphics window.
ebr		Sets entrance beam radius.
epxy	plane_aiming	Sets ray aiming mode: paraxial (on) or aplanatic (off).
fanr	fan_rays	Sets number of rays in fans.
gcs		Sets surface that establishes global coordinate system for ray data displayed in global coordinates.
gprt	group_print	Sets surface group printing mode.
grck	grin_aper_check	Sets use of aperture checking for all ray segments in a gradient index medium.
htnu	htanu_ray_intcpts	Sets use of H -tan U ray intercept curves and OPD - sin U OPD curves

ims		Sets image surface number.
lid		Sets lens identification.
nao		Sets numerical aperture in object space.
nolc	no_latcol_fans	Suppress display of lateral color in OPD fans.
obh		Sets object height.
omna	opt_minairthick	Default minimum value for air thickness variables.
omng	opt_minglassthi ck	Default minimum value for element (glass) thickness variables.
omxa	opt_maxairthick	Default maximum value for air thickness variables.
omxg	opt_maxglassthi ck	Default maximum value for element (glass) thickness variables.
opac	opt_rayopapck	Sets use of aperture checking for optimization rays.
opar	opt_asarate	Annealing rate for ASA global optimization.
opat	opt_asaterm	Termination level for ASA global optimization.
opbw	opt_boundaryw gt	Weight of boundary violation contributions to merit function.
opct	opt_conoptol	Tolerance for one-sided constraint operands.
opcw	opt_colorwgt	Name of CCL command that computes the weighting function for wavelengths.
opdf	opt_damping	Current value of damping factor.
opdi	opt_derivincr	Default derivative increment for variables.
opdm	opt_dampmult	Multiplier for damping factor.
opds	opt_dampstart	Starting value for damping factor.
opdw	opd_in_waves	Sets optical path difference reporting to wavelengths or lens units.
opfw	opt_fieldwgt	Name of CCL command that computes the weighting function for field points.
opoc	opt_oprdcccl	Name of CCL or SCP command that computes the value of CCL operands.
oppw	opt_pupilwgt	Name of CCL command that computes the weighting function for points in the pupil.
opst	opt_solutiontol	Solution tolerance for continuing full iterations
opvd	opt_vardamping	Sets use of adaptive damping for optimization variables.
opvi	opt_vbinccntrl	Sets mode for assigning variable damping increments (according to variable type or adaptive).
opzw	opt_operrzerowt	Give weight of zero to incalculable operands when computing error function.

pcbg	par_background	Background irradiance transmittance outside “bar and space” ideal image.
pcbp	par_barperiod	Period of bars in ideal image.
pcbw	par_barwidth	Width of bars in ideal image.
pcen	par_energynorm	Sets use of unity image energy normalization.
pcgx	par_sourcessx	Relative $1/e^2$ half-width of Gaussian effective source in x direction.
pcgy	par_sourcessy	Relative $1/e^2$ half-width of Gaussian effective source in y direction.
pcia	par_innerannrad	Relative inner radius of annular effective source.
pcip	par_numimgpts	Number of points in ideal image.
pcis	par_imagespd	Sets use of image space spot diagram for partial coherence calculations.
pcmi	par_minirrad	Irradiance transmittance of “spaces” between bars of ideal image.
pcnb	par_numofbars	Number of bars in ideal image.
pcsg	par_sigma	Ratio of effective source radius to entrance pupil radius.
pcsp	par_spacephase	Phase transmittance of “spaces” between bars of ideal image.
pcxs	par_xsource	Relative shift of center of effective source, in x direction, relative to center of entrance pupil.
pcys	par_ysource	Relative shift of center of effective source, in y direction, relative to center of entrance pupil.
pre		Atmospheric pressure in atmospheres.
pzdp	pol_degreepol	Degree of polarization of incident beam.
pzea	pol_ellangle	Angle between y-axis and major axis of polarization ellipse.
pzer	pol_ellratio	Ratio of minor axis to major axis of polarization ellipse.
pzlh	pol_lefthanded	Sets use of left-handed polarization ellipse.
pzmf	pol_mgf2coating	Sets use of quarter-wave MgF_2 coating on refracting surfaces.
pzrt	pol_raytrace	Sets use of polarization ray trace.
rfs		Sets reference surface number.
sasd	src_astig_dist	Sets longitudinal source astigmatism distance.
sdad	spd_apdiv	Number of aperture divisions used in spot diagram.
sdaz	spd_apodized	Sets use of Gaussian pupil apodization.
sdde	spd_diff_eff	Sets use of extended scalar theory diffraction efficiency calculations for spot diagram-related evaluations.

sdgx	spd_gaussxsize	$1/e^2$ irradiance point in x -direction on surface 1 for Gaussian apodized pupil.
sdgy	spd_gaussysize	$1/e^2$ irradiance point in y -direction on surface 1 for Gaussian apodized pupil.
sdis	spd_imagespd	Sets use of image space spot diagrams.
sdms	spd_mtf_switch	Peak-to valley OPD for switch from diffraction to geometrical MTF calculation.
sdpc	spd_polychrom	Sets use of all defined wavelengths in spot diagram.
sdtf	spd_tfr_freq	Default spatial frequency for through-focus MTF calculations.
snoi		Adds system note i .
tele	telecentric_pupil	Sets use of telecentric entrance pupil.
tem		Temperature in degrees Celsius.
twl		Sets wavelength for fringe tolerance items.
uni		Sets lens units (number of mm in current units).
war m	wide_angle_mode	Sets use of wide angle ray aiming mode.
wrsp	wvf_ref_sph_pos	Location of reference sphere for OPD calculations.
wv		Sets wavelengths.
wvi		Sets individual wavelength value for wavelength i .
ww		Sets wavelength weights.
wwi		Sets individual wavelength weight value for wavelength i .
xaba	xarm_beam_angle	Beam angle to be used for extended aperture ray aiming mode.
xarm	extend_aper_mode	Sets use of extended aperture ray aiming mode.

Command Definitions

General

If you want to see the formal definition of a command, along with its arguments and argument lists, you can find it in the on-line help system.

A command in CCL is simply a procedure having a void data type, conventionally specified by the equivalent type “cmd.” As mentioned above, commands are often executed by typing them on the command line, but they may also be invoked from a menu, toolbar, or called from another command. Also as described above, a

command may have zero or more arguments, and the arguments must be specified as to its data type, at the very least. Once a CCL file is compiled, OSLO has knowledge of all command arguments, and if any are missing or incorrect, an error message will be generated, or a prompt for a corrected value will be displayed.

One advantage that CCL has over C as a language is the error checking that it provides for argument values. In the following sections, the facilities available to CCL programmers for argument checking are described in detail. It is important to note, however, that the use of these facilities is optional and is not required by CCL.

To understand command definitions, it is worthwhile to study a typical example. The example chosen here is the definition of the **paraxial_trace** command, which is actually an internal command in OSLO, but the definition has been rephrased as it would appear in CCL. The definition appears in three parts. First is the specification of a list (Srf_Option), second is the declaration of three integers (Surf_Range, First_Surf_Nbr, and Last_Surf_Nbr), and third is the actual definition of the paraxial trace command.

List elements must be declared in a separate file from argument and command definitions, since they are compiled separately.

```
list Srf Option
  {"Srf" = [1,2]0,
   "All" = []1}

int Surf Range{list = Srf Option, default(noquery) = "srf"}
int First Surf Nbr{lolim = 0, hilim = 100, default(noquery) = 0}
int Last Surf Nbr{lolim = 0, hilim = 100, default(noquery) = 0}

cmd paraxial_trace(int Surf Range, int First Surf Nbr, int Last Surf Nbr)
{
  actual code goes here...
}
```

In this example, our interest is only in the format of the command definition, not in the code itself, nor the declaration of any local or global variables that the command might use. The definition states that the **paraxial_trace** command does not return a value, and that it has three integer arguments. If the command is executed from the keyboard, and the argument values are not entered with the command, the program will obtain the needed values according to the argument definitions, as specified above (see discussion in the following sections).

The environment names described here are not related to the environment variables used to describe file locations.

In OSLO, an “environment” is a set of conditions that must be satisfied before a command can be successfully executed. For example, before drawing a lens picture, there must be a valid lens in memory and a graphics window must be open. If these conditions are not currently satisfied, trying to execute the command will cause OSLO to produce an error message stating that the command is currently disabled. Most built-in OSLO commands have an associated environment that is used to control execution and is also used for “graying-out” of menu items and toolbar buttons. CCL commands may also be assigned an environment, by including a statement of the form “env =

environment_name,” in the body of the command, where *environment_name* is the name of an OSLO environment. For specific information on OSLO environments, please see the on-line help system.

Argument definitions

Arguments to procedures are required to be specified as to data type. CCL adds the possibility of extended argument specifications, which allows OSLO to provide error checking that helps to ensure proper functioning of CCL programs.

Among the properties that *may* be associated with scalar CCL procedure arguments are upper and lower bounds for their allowed values. For character string arguments, minimum and maximum character counts may be specified. These bounds, or counts, may be specified as constants or as the current values of specified global variables. If bounds are specified for an argument, the argument will be checked against the specified bounds whenever it is passed to a procedure, either from the command line or from a lower level procedure. When a bound is violated it is reported to the user. If the argument was entered by the user in response to a prompt, the user is prompted to re-enter the argument. Otherwise, the violation is regarded as a fatal error and procedure execution is terminated.

In the above example, the arguments `First_Surf_Nbr` and `Last_Surf_Nbr` are specified to lie in the range between a low limit of 0, and a high limit of 100. If a value is entered for these arguments, it will be accepted if it is within the prescribed range. If a value is out of the prescribed range, an error message will be generated. If values are not supplied for these arguments, according to the definition, default values of 0 should be taken without querying the user, i.e. **default(noquery)**. The alternate specification would be **default(onprompt)**, which would allow the user to override the default value.

Since CCL is a compiled language, the value of **lolim** and **hilim** must be known at the time the command is compiled. In the example, the value of 100 for **hilim** was selected to be large enough to include all possible values.

Within the body of a procedure, the supplied arguments behave in every respect as local variables. In CCL, once an argument is declared its name and type is known globally. Thus, if an argument “int count” is declared in the argument list for a given procedure, it is not permissible to declare an argument “double count” for another procedure. The reason for this is that various properties may be associated with each procedure argument, and these properties are keyed to the argument name.

In order to prevent unwanted interaction between CCL arguments and those used for internal OSLO commands, separate name spaces are used. Thus, if `First_Surf_Nbr` is used by an internal command, the same name can be used in CCL, and an independent variable will be established. If, for some reason, you want to make it the same variable as the one defined internally, you should precede the name with an underscore, i.e., `_First_Surf_Nbr`.

Dimensions are not specified for array arguments declared in an argument list. The size of an array argument passed to a procedure is determined by the caller. Bracket pairs must be specified, however, to indicate the array nature of the argument, i.e., one or two dimensional. The CCL interpreter also maintains a constant check to insure that all array indices remain within the bounds specified in the array declarations, and that all integer and character variables remain within

the limits imposed by their type, e.g., character values must not be assigned values outside the range 0 – 255.

Lists

In the example shown on page 519, list Srf Option {"Srf" = [1,2]0, "All" = []1}, the zeroth argument (CCL arguments are numbered starting from 0) is specified by a list. A CCL list consists of a name, preceded by the word "list," and followed by a set of values enclosed in curly brackets { }. The values are specified in the form of equations relating an expression in quotes (left side) to values (right side).

An argument list has the form:

```
list listname {
  "name1" = [conditional arguments]value1,
  "name2" = [conditional arguments]value2,
  ...
}
or
list listname {
  "{shortname1}longname1" = [conditional arguments]value1,
  "{shortname2}longname2" = [conditional arguments]value2,
  ...
}
```

Depending on the type of argument, the items value1, value2, etc., may be integers, real numbers or quoted strings.

The left side is (usually) broken into two parts: a short string (3 or 4 letters) enclosed in curly brackets, and a longer string that is usually an English word. When the list is displayed, the longer form is shown, but either form can be typed in to select a desired value.

The right side of a list equation specifies two properties of the list arguments. The first, common to all lists, is the value to be used by CCL when the particular list element is selected. This value is commonly specified as a "#define" consisting of a series of capital letters.

The second property of a list is a series of numbers that appear on some lists and specify the arguments to be used when that list element is selected. These numbers appear in square brackets and the arguments are numbered from left to right starting from zero. The numbers in brackets refer to the actual arguments required when that particular list element is selected.

In the above example, the zeroth argument is a list. The zeroth list element shows [1,2] on the right side, and the first element specifies []. Then, if the zeroth list element (srf) is selected, the command will require values for arguments 1 and 2. On the other hand, if the first list element (all) is selected, the command will not require values for arguments 1 and 2. Empty brackets mean that no additional arguments are required, and no brackets mean that all arguments are required.

Lists used in command definitions are compiled by a separate compiler, which must be run prior to compiling any commands that use the results. In OSLO, CCL lists must be stored in a file **a_list.ccl**. This file can be compiled separately or together with CCL commands. It will automatically be compiled before source files because its name begins with **a_**.

Overview

OSLO is a combination of an optical design program and a fully integrated command language, and consequently uses a different paradigm from many other programs. The overall concept in OSLO is that there should be no optical design task that is beyond the capability of the program. Simple things should be pre-programmed and easily accomplished with a few keystrokes, while more complicated tasks should require more user planning and development. However, in all cases, the user should be able to build upon the fundamental capabilities and have access to the results of the built-in routines.

There is a hierarchy of operation levels in OSLO, which increases in complexity of user involvement roughly as follows:

- Clicking toolbar icons
- Selecting menu items
- Executing commands on the command line
- Executing multiple commands in the editor
- Saving commands on disk as SCP commands
- Developing CCL routines that become part of OSLO
- Compiling C routines into dynamic link libraries (DLLs)

Generally, the speed of operation of the program, and to some extent the scope of what can be accomplished, increases according to the same hierarchy.

A key element in OSLO is an extensive support which includes routines to manage an application program such as OSLO. Program management, in this context, refers to putting up windows, handling events such as mouse clicks and keystrokes, comparing user input to command tables, lists, etc., and building complete commands, which are then compiled and executed.

In addition to the standard arithmetic operators for addition (+), subtraction (−), multiplication (*), and division (/), CCL contains an exponentiation operator. Exponentiation is indicated by the FORTRAN convention “**”. For example, $2^{**}3 = 8$. The standard mathematical functions are used in the format of the C programming language math library.

Note that the trigonometric functions require angles to be expressed in radians (as required in C), and that the inverse trigonometric functions report angles in radians. OSLO has two predefined constants to facilitate the use of angles in degrees. **Pi** is the numerical value of the constant π . **Dr** is the conversion factor from degrees to radians. If y is an angle in degrees, $y*\text{Dr}$ is the angle expressed in radians. Similarly, if x is an angle in radians, x/Dr is the angle expressed in degrees. For example, to compute the sine of 45 degrees, type **sin(45*dr)** in the command line.

Several variables are predefined in OSLO, and these are available to command-line procedures, as well as to SCP and CCL programs. The predefined data is of three types:

1. **Global variables**
2. **Lens data (read-only)**
3. **Spreadsheet buffer elements**

These data are supplemented by a series of functions that return values of commonly used data, such as preferences.

Data Entry Methods

Command Line

In OSLO, commands are executed as one-line programs, which gives OSLO some unusual properties not found in conventional command-driven software, one of the most useful being that the command line can serve as a calculator.

If you enter an expression in the command line, it will be evaluated and the result will be displayed in the message area with the prefix "Result = ." If the **Output_echo (echo)** preference is active, the result will also be printed in the text window. If you type a mathematical expression into a SmartCell, it will be evaluated and the result entered as data in the cell. You can also enter a command argument as an expression, but in this case, the type of the result (integer or real) must match the type of the argument.

Internal OSLO Text Editor

Global variables

OSLO predefines the single letters of the alphabet as variables in which you can store numbers. **A** through **H** and **O** through **Z** are predefined as real global variables, and **I** through **N** and **ll** through **NN** are predefined as integer global variables. For example, to store the square root of 2 in the variable **Q**, type **q = sqrt(2)** in the command line. (When using these single letter variables, OSLO makes no distinction between upper case **Q** and lower case **q**.) These variables can be used in subsequent expressions. Thus, continuing from the previous example, type **q/2** to compute and display $\sqrt{2}/2$.

Six real-valued, one-dimensional arrays are also available for use as global variables. These arrays are **Ua[]**, **Va[]**, **Wa[]**, **Xa[]**, **Ya[]**, and **Za[]**. Each array has 1000 elements.

Three string variables, **Astr**, **Bstr**, and **Cstr**, are defined to permit elementary string operations on the command line. Each can hold up to 255 characters (plus the terminating NULL character).

Lens data (read-only)

Data values

Most of the data describing the current lens, including both surface data and operating conditions, are available as read-only variables for use on the command line. Vector-style data (e.g., pertaining to a particular surface) are accessed using

C language style array notation, i.e., square brackets “[]”. In the case where there are two subscripts, e.g., `rn[i][j]`, the surface number is the first subscript.

You should be aware that several of the read-only variable names are the same as either the long or short forms of the command that can be used to change that item. Thus, you might execute a command rather than just display the value of a variable if you use some variables as the first (or only) input on the command line. Using **ebr** as the first item on the command line, for example, will be interpreted as the command **ebr**. If you want to just display the current value of the entrance beam radius, you need to type “**+ebr**”. In this form, **ebr** is interpreted as a numeric value, not a command. An entry such as **t = ebr** is unambiguous; the current value of the entrance beam radius is assigned to the variable *t*.

Generally, the read-only variable for a data item is the same as the command used to set that data item. For example, the radius of curvature of surface 3 is **rd[3]**; the thickness of surface 4 is **th[4]**; the number of aperture divisions for spot diagrams is **sdad**. The commands used to set the values of lens data may be found in Chapter 11 – Commands; in particular, see the Lens Surface Data and Operating Conditions tables. In addition, several additional read-only variables providing lens data are also available, and are described in the following sections.

Data type codes

Curvature type

The variable **cvtyp[i]** indicates the type of curvature (or, equivalently, radius of curvature) on surface *i*. The values are

cvtyp	Type of curvature specification
0	direct user specification
1	curvature pickup
2	minus curvature pickup
3	axial ray angle solve
4	chief ray angle solve
6	fixed curvature
7	test plate fit
8	axial ray angle of incidence solve
9	chief ray angle of incidence solve
10	axial ray aplanatic solve
11	chief ray aplanatic solve

The variable **cvdat[i]** contains the curvature solve value or curvature pickup constant. **cvpsn[i]** is the source surface for a curvature pickup.

Thickness type

The variable **thtyp[i]** indicates the type of thickness on surface *i*. The values are:

thtyp	Type of thickness specification
0	direct user specification
1	thickness pickup
2	minus thickness pickup
3	length pickup
4	minus length pickup
5	axial ray height solve
6	chief ray height solve
8	fixed thickness
9	edge contact solve

The variable **thdat[i]** contains the thickness solve value or thickness pickup constant. **thpksn1[i]** and **thpksn2[i]** are the source surfaces for thickness pickups.

Aperture radius type

The variable **aptyp[i]** indicates the type of aperture radius on surface *i*. The values are

aptyp	Type of aperture radius specification
0	solved aperture radius
1	direct user specification
2	aperture radius pickup
7	fixed aperture radius

The variable **appksn[i]** is the source surface for an aperture radius pickup. The variable **chkap[i]** is equal to 1 (true) if the aperture has been marked as a checked aperture; otherwise **chkap[i]** is 0 (false). **numsap[i]** is the number of special apertures that have been defined for surface *i*.

Glass type

The variable **gltyp[i]** indicates the type of glass following surface *i*. The values are

gltyp	Type of glass specification
0	direct index specification
1	air
2	catalog glass
3	model glass
4	reflector
5	glass pickup
6	direct index specification
7	model glass
9	fixed glass
10	air, fixed (from lens database)

The variable **glpksn[i]** is the source surface for a glass pickup.

Surface type

The variable **srftyp[i]** indicates the type of surface for surface *i*. The values are

srftyp	Type of surface
0	plane
1	sphere
3	cylinder
4	conic
5	asphere
6	toric
8	biconic
9	spline

Toric type

The variable **toric[i]** indicates the type of toric for surface *i*. The values are

toric	Type of toric surface
0	not a toric surface
1	<i>y</i> -toric surface
2	<i>x</i> -toric surface

Aspheric surface type

The variable **asp[i]** indicates the type of polynomial aspheric for surface *i*. The values are

asp	Type of polynomial aspheric surface
0	standard asphere (ado)
1	symmetric general asphere (asr)
2	asymmetric general asphere (asx)
3	ISO symmetric asphere (isr)
4	ISO <i>x</i> toric (trx)
5	ISO <i>y</i> toric (try)
6	biconic (bic)
7	symmetric cone (cnr)
8	asymmetric cone (cnx)
9	symmetric Zernike sag (zsr)
10	asymmetric Zernike sag (zsx)

Gradient index medium type

The variable **gdt[i]** indicates the type of gradient index medium for surface *i*. The values are

gdt	Type of gradient index medium
1	axial and radial gradient (arg)
2	Wood lens (wod)
3	axial and elliptical gradient (aeg)
4	spherical gradient (sph)
5	Maxwell's fisheye lens (mxf)
6	Luneburg lens (lbg)
7	User-defined gradient (usr)
8	axial and sinusoidal radial gradient (sns)
9	axial and tapered radial gradient (tap)
10	SELFOC gradient (sel)
11	GRADIUM gradient (grd)
12	user gradient; arbitrary coefficients (ugr)

Diffractive surface type

The variable **doe[i]** indicates the type of CGH diffractive surface for surface *i*. The values are

doe	Type of CGH diffractive surface
0	symmetric CGH (even orders) (dfr)
1	asymmetric CGH (power series) (dfx)
2	symmetric CGH (all orders) (dra)
3	asymmetric CGH (abs. value) (dx)
4	symmetric Zernike phase (zrr)
5	asymmetric Zernike phase (zrx)

Non-sequential surface group data

These variables are the values, for surface i , of the various components of special actions for surfaces in non-sequential groups.

Variable name	Meaning
<code>nsq_act_ref[i]</code>	special action type for reference rays
<code>nsq_con_ref[i]</code>	special action condition for reference rays
<code>nsq_gsn_ref[i]</code>	pickup from group surface number for reference rays
<code>nsq_hnb_ref[i]</code>	hit number for reference rays
<code>nsq_act_ord[i]</code>	special action type for ordinary rays
<code>nsq_con_ord[i]</code>	special action condition for ordinary rays
<code>nsq_gsn_ord[i]</code>	pickup from group surface number for ordinary rays
<code>nsq_hnb_ord[i]</code>	hit number for ordinary rays

The special action type variables (**`nsq_act_ref`** and **`nsq_act_ord`**) may have the following values:

Value	Special action type
0	no action (ordinary ray; <code>nsq_act_ord</code>) same action as ordinary ray (reference ray; <code>nsq_act_ref</code>)
1	pickup from group surface
2	reflect
3	obstruct
4	pass undeviated

The special action condition variables (**`nsq_con_ref`** and **`nsq_con_ord`**) may have the following values:

Value	Special action condition
1	to negative
2	to positive
3	before n^{th} hit
4	after n^{th} hit

Lens array data

These variables are the values of lens array data on surface i .

Variable name	Meaning
ary_typ[i]	type of array; 1 = regular, 2 = tabular
ary_xspac[i]	x spacing for regular array
ary_yspac[i]	y spacing for regular array
ary_yoff[i]	y offset for regular array
ary_numchn[i]	number of channels for tabular array
ary_xc[i][j]	x center for tabular array channel j
ary_yc[i][j]	y center for tabular array channel j
ary_zc[i][j]	z center for tabular array channel j
ary_tla[i][j]	<i>tla</i> tilt for tabular array channel j
ary_tlb[i][j]	<i>tlb</i> tilt for tabular array channel j
ary_tlc[i][j]	<i>tlc</i> tilt for tabular array channel j

Optimization data

Also available are read-only integer variables denoting the next item number to be added to the various classes of optimization data.

Variable name	Meaning
fptnbr	number of next field point to be added
oprnbr	number of next operand to be added
raynbr	number of next ray to be added
sdsnbr	number of next spot diagram to be added
varnbr	number of next variable to be added

Note that all of these variables indicate the number of the next new item added. Thus, for example, the current variables are numbered from 1 to (**varnbr** – 1).

Other variables

In addition to variables corresponding to lens entry commands and the variables described above these variables are also available.

Variable name	Meaning
beg_selection	first surface in a selected range in the surface data spreadsheet
cfg	current configuration number
current_pen	current graphics pen number
cursnbr	current surface number in lens editing
end_selection	last surface in a selected range in the surface data spreadsheet
gfx_window	current graphics window number
lensym	flag indicating whether lens has meridional symmetry (1) or not (0)
maxcfg	maximum defined configuration number
nbr_pens	number of available graphics pens
numw	number of wavelengths
srfssopen	flag indicating whether surface data spreadsheet is open (1) or closed (0)
txt_window	current text window number
wav	current wavelength number

Data functions

In addition to the of variables described above, OSLO contains functions that return values of commonly used data.

ssb(row,column) returns a spreadsheet buffer element.

get_glass_name(i) copies the name of the glass for surface *i* into the read-only variable **glass_name**.

get_surface_note(i) copies the surface note for surface *i* into the read-only variable **surface_note**.

get_system_note(i) copies system note number *i* into the read-only variable **system_note**.

Also, the current values of OSLO preferences can be obtained from the returned values of the commands

str_pref must be used with **strcpy**, **strcat**, etc. It does not return a value directly to the keyboard.

real_pref(preference_name)

int_pref(preference_name)

char_pref(preference_name)

str_pref(preference_name)

Spreadsheet buffer elements

All text output from OSLO is formatted to contain up to ten columns of real numbers. These columns are understood to be implicitly denoted as *a*, *b*, *c*, *d*, *e*, *f*, *g*, *h*, *i*, and *j*. Real numbers displayed in the text window are automatically stored in, and can be retrieved from, an internal buffer called the *spreadsheet buffer*, using a column-row designation. For example, the designation **c2** indicates the number in the third column, second row of the spreadsheet buffer. Clicking on a real number in the text window will display its spreadsheet buffer location and full-precision numerical value in the message area of the main window. The value of an output quantity can be used in the command line by entering its spreadsheet buffer location. Spreadsheet buffer data can also be obtained using the **ssb(row, column)** function.

Using the spreadsheet buffer

The **ssbuf_reset (sbr)** command provides control for the use of the spreadsheet buffer. The command has the form:

ssbuf_reset(reset_index, rows_to_clear, value_for_clear)

Reset_index can be positive or negative. If it is positive, it gives the spreadsheet buffer row index number that is to be remapped as 1. If is negative, it gives the row index number to be remapped as 1, but indexed negatively relative to the current row 1 (the spreadsheet buffer is a circular buffer). If reset_index is positive, the next row to be written into the spreadsheet will be row 1. If reset_index is negative, the next row to be written into the spreadsheet will be the absolute value of reset_index.

Rows_to_clear is the number of buffer rows to clear, starting with the next row to be written into the spreadsheet. If rows_to_clear is 0, no rows are cleared. If rows_to_clear is omitted, the entire buffer is cleared. Thus if rows_to_clear is omitted, all positive values for reset_index have the same effect, i.e., the buffer is cleared and the next row to be written into is 1.

Value_for_clear is the numerical value that will be written into the cleared rows of the spreadsheet buffer. The default value for Value_for_clear is 0.0. This argument may be used to initialize the spreadsheet buffer to any desired value.

The main purpose of reset_index is to allow screen output to be preserved while a CCL or SCP command uses the spreadsheet buffer for receiving data from OSLO. A typical code sequence might be:

```

k = sbrow; /* see below for definition of sbrow */
ssbuf_reset k 1;
.
/* CCL or SCP program */
.
ssbuf_reset -k 0;

```

Note that values of 1, 0 and -1 for `reset_index` all have the same effect, i.e., no renumbering of the spreadsheet.

Three integer functions, **sbrow**, **sbcount** and **sbmaxrow** are available to obtain row information about the spreadsheet buffer.

Sbrow() returns the row index of the next spreadsheet buffer row to be written into.

Sbcount() returns a count of the number of spreadsheet buffer rows written into since the last **ssbuf_reset** (or **textwin_reset**). This may be greater than the number of rows in the buffer.

Sbmaxrow() returns the number of rows in the spreadsheet buffer.

There is unavoidable interaction between the **ssbuf_reset** command and the display of previously written screen data in the message area (when the data is clicked). **Ssbuf_reset** causes a renumbering of the screen data. If the value of the first argument sets the new pointer to a number that has already been written to the screen, then it is possible to have two displayed numbers referring to the same buffer element. Only the last will be correct, of course. Some experimentation will help you to understand these effects.

The **Output_text (outp)** preference affects text output from OSLO. The **print** and **printf** commands always print to the text window (regardless of whether or not **outp** is off) and do not write anything to the spreadsheet buffer. If you want to write into the spreadsheet buffer from SCP or CCL, use the commands **aprint** and **aprintf**. These work the same as **print** and **printf**, except that real numbers are written into the spreadsheet buffer.

Star Command Programming (SCP)

OSLO commands and predefined data give it two essential elements that are needed to make a macro language. The third element are control statements. OSLO uses the same control statements as the C language, since they form a complete set, are widely used, and well standardized.

SCP is a subset of CCL that has predefined variables that parallel those used in SCL, the macro language used in older versions of OSLO (Series 2 & 3). SCP procedures look like CCL procedures except that they don't have braces around the whole procedure, don't allow variables to be declared, and run slower (since they are not saved in compiled form). All control structures, including switch and case statements, are supported. All OSLO commands, excepting those declared static, can be invoked from SCP. Either the long or short form of a command can be used.

SCP has considerably greater power than SCL, principally because most library functions available to CCL, including string, file, and graphics functions, are available. However, the limitations described above prevent SCP from replacing CCL for serious programming tasks. SCP is intended for easy "macro" extensions to OSLO.

SCP has both integer and real numeric variables. In some parts of SCP, such as variable subscripts, real variables can be used in addition to integers, but in others, such as arguments to commands, it is necessary to use an integer when one is called for in the command definition.

Spreadsheet buffer access in SCP is read-only. The spreadsheet buffer can be cleared using the **textwin_reset** command (**twr**). The spreadsheet buffer row index can be reset using the **ssbuf_reset** (**sbr**) command.

SCP commands are executed using the **xeq** command, which executes a block of commands saved in a file having the delimiter ".scp". The **xeq** command can be invoked by pre-pending a command name with * (hence the terminology "star command"). Anytime "***cmdname**" appears in a context where a command name is expected, it will be interpreted as "**xeq cmdname**." Similarly, "***cmdname:filename**" will be interpreted as "**xeq cmdname:filename**." The use of "*" in place of "**xeq**" applies to the command line and *.scp files. It is not supported in CCL source files.

The public\scp and private\scp directories hold files that contain command sequences to be executed. All the files in these directories should have the extension ".scp." The private directory is searched prior to the public directory.

The format of *.scp files is:

```
*cmdname
ccl command line statement;
ccl command line statement;
- etc. -
```

```
*cmdname
ccl command line statement;
ccl command line statement;
- etc. -
```

Blank lines should only be used between command blocks. Since an SCP procedure is executed as a multiline block, semicolons must be used as statement terminators.

The command line entry "**xeq cmdname**" will look for a block of statements with the heading "***cmdname**" in the file *cmdname.scp* in the private and (if not found) public scp directories. If a statement "**xeq cmdname2**" appears in the block of statements so executed, the statements under the heading "***cmdname2**" in the same file will be searched for and executed (as a subroutine).

When a top level SCP command is executed, the file *default.scp* will be searched if the file *cmdname.scp* does not exist or if the command is not found in *cmdname.scp*.

When a sub-level cmd (***sub_cmd**) is encountered in an SCP procedure, the file where the top level cmd was found is first searched. If the search fails and the top level cmd was not in *default.scp*, then *default.scp* is searched. Finally, the file *sub_cmd.scp* is searched.

The statement "**xeq cmdname:filename**" will only look for the statement block "***cmdname**" in the file *filename.scp*.

Statements following ***cmdname** in a *.scp file are processed as a block (i.e., they are not executed a line at a time). Consequently, you can have for-loops, etc., that span a group of lines. Command blocks are limited to a maximum of 8192 characters.

An SCP command may have up to six real valued and three character string arguments (of string length less than 256 characters). The real argument values are available in the global variables **Arg1**, **Arg2**, **Arg3**, **Arg4**, **Arg5**, and **Arg6**. The string arguments are **Str1**, **Str2**, and **Str3**. The default value of the real arguments is 0.0, and the default string arguments are null (empty) strings. Arguments may be used when executing SCP sub-procedures, but note that the same variables (Arg1, Str1, etc.) will be used, so local copies of the original argument values in the calling procedure should be kept, if necessary. Also note that, unlike built in OSLO commands and CCL commands, the user will not be automatically prompted for arguments for SCP commands.

Compiled Command Language (CCL)

CCL is a command language that allows the user to extend OSLO's repertoire of commands and functions. When OSLO is running, CCL commands and functions are indistinguishable from built-in commands and functions. This section details both the programming language and the CCL development environment.

Introduction to CCL presents an overview of the CCL programming language.

Programming in CCL describes the features of CCL and how they compare to those of C.

Using CCL with OSLO discusses the interaction between CCL and OSLO.

Introduction to CCL

CCL is designed to provide a complete, high-performance macro language that extends the capabilities of a software application such as OSLO. To develop a new CCL routine, you write the required code, save the code as a *.ccl file, and then compile it with the CCL compiler, which automatically generates the pseudo code (*p-code*) used to run the OSLO program.

The CCL language is a subset of the C programming language with a few modifications and extensions. Since CCL is primarily intended to support scientific programming, it does not span the whole range of C, and is consequently easier to learn and use. For example, CCL does not support *pointers*, a mainstay of C. Also, CCL does not support *structs* or *unions*, so the complexity of CCL data structures is limited. It does, however, support extended one and two-dimensional arrays, which are necessary for scientific programming.

CCL does support the essential parts of C, including the basic data types, the full range of operators, and nearly the entire control syntax. Some extensions of C are implemented in CCL, including improved handling of array arguments, extensive checking of limits, built-in list handling, string files, and command environment management. In addition to support of essential components of the standard C library (math, file and string handling routines), CCL provides functions for callback programming, window management, and a compact, high-speed graphics subsystem.

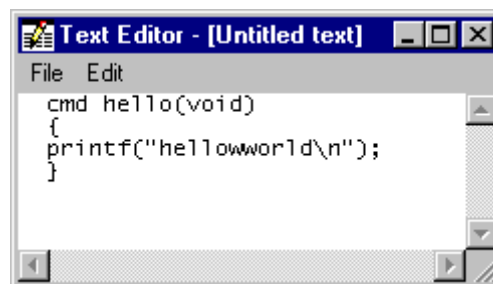
The discussion in this manual assumes that you are familiar with elementary C. It is not necessary to know about pointers, or complicated data structures, since they are not used in CCL. However, if you are not at all familiar with C, you may want to consult a standard text (there are dozens at your local bookstore) before trying to write your own CCL.

Before considering the details of CCL, however, you may wish to see how to set up the ubiquitous “Hello world” program in CCL. The steps to writing a CCL command are simple:

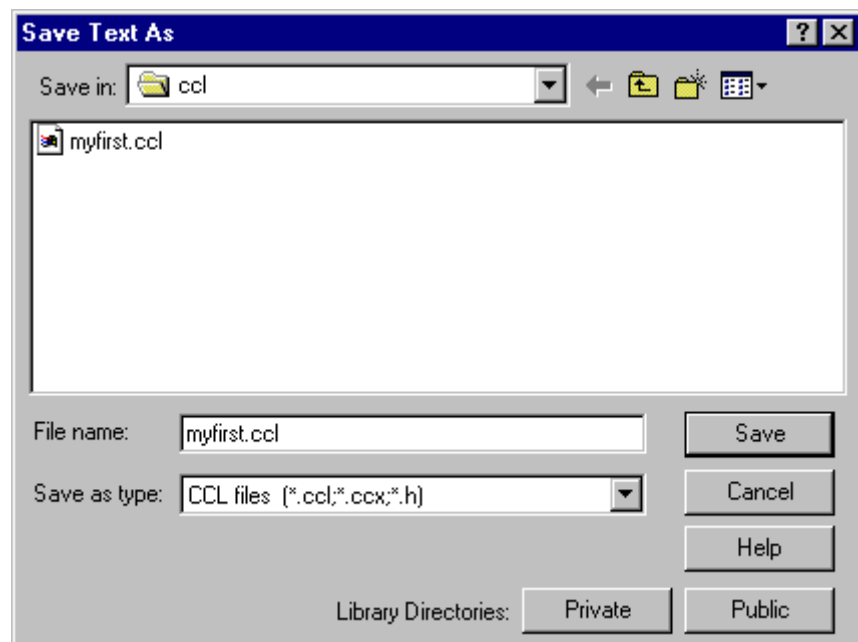
- Open the Text editor window.
- Type the CCL source code into the editor.
- Save the contents of the text editor as a CCL file.
The file will be automatically compiled.
- Execute the command by typing its name on the command line.

The entire process is illustrated with an example to print the greeting “Hello world.”

To open the text editor, click the F4 toolbar icon. If the contents of the text editor are not empty, then in the text editor menu click File, and click New. Next, enter the “hello” command into the text editor as shown below.



Once the command is typed in, click the File menu in the text editor and select Save As. This is step three and will bring up the file dialog box for saving new CCL files. Click the Private button to select your private CCL directory. Then type **myfirst.ccl** in the File Name box. Click the Save button to save the contents of the text editor to the file **myfirst.ccl** in your private CCL directory, as shown below.



The private CCL command files are automatically compiled when the file is saved. The CCL compiler reports the file saved message and displays any compiler errors.

***CCL COMPILATION MESSAGES:**

No errors detected

If any errors were detected, correct the errors in the text editor, and then click the File menu and select Save. If the CCL COMPILATION MESSAGE is not displayed, click User in the main menu, select Compile CCL, and then click Commands on the pull-right menu.

When the CCL file is compiled with no errors detected, the CCL commands are ready to be used. To run the command, enter **hello** in the command line.

Hello world

The result is printed to the current text output window.

This example included just one command “hello” in the CCL file **myfirst.ccl**. Any number of CCL commands can be entered into a single CCL file. In general, any filename with the extension “.ccl” that is acceptable to the operating system, may be used.

Programming in CCL

This section provides a detailed discussion of CCL syntax, including how it differs from C. It is assumed that you are generally familiar with programming in C.

Preprocessor

CCL provides the following compiler directives. Note that the #else directive is not supported:

```
#define
#endif
#ifdef
#ifndef
#undef
#include
```

CCL data

Basic types - CCL supports the void type for commands, and three data types for variables and functions. Objects of CCL type “char” are single byte unsigned integers (0 - 255) and are most commonly used for character strings. Objects of CCL type “int” or “long” are signed integers whose maximum magnitude is at least 2147483647. Objects of CCL type “real,” “float,” or “double” are double precision floating point numbers occupying 8 bytes on most systems. The data type keywords are listed in the following table, according to their C-language counterparts.

CCL type	C type	Typical size
void	void	--
cmd	void	--
char	unsigned char	1 byte
int	long	4 bytes
long	long	4 bytes
real	double	8 bytes
float	double	8 bytes
double	double	8 bytes

Arrays - CCL supports 1 or 2 dimensional array variables for each of the data types. When members of an array variable are used in CCL statements, CCL permits floating point variables or expressions to be used for the array indices. In such cases, the index values are automatically truncated to integer values. In certain situations this feature can be useful. Normally, however, integers or integer expressions should be used for array subscripts.

Scope - Variables may be either global or local in scope, and are declared in accordance with C language conventions. The C “static” storage class is supported for local variables. Variables declared outside of procedures in a source file are global in scope and are available for use in all subsequent procedures, both in the source file where they are declared and in all subsequent files encountered during compilation. Global variables that are needed in more than one CCL file should be placed in a `a_global.ccl`. Global variables are preserved in permanent storage and may be initialized with constants. If an initializer list is specified for an array variable, the first dimension may be omitted and the compiler will determine its value according to the number of initializers.

Examples of variable declarations:

```
int count, incr, elements;
char array1[50];
real matrix[10][20];
```

Examples of initialized global variables:

```
int first = 50, last = 100;
char message[] = "hello";
char list[][10] = {
    "item1",
    "item2",
    "item3"
};
double matrix[2][2] = {
    1.5, 1.9,
    2.8, 3.6
};
```

Variables declared within the body of a procedure are local in scope and are available for use only within that part of the procedure which follows their declaration. Local variables, unless declared as static, are stored in temporary reusable storage and are not preserved after exit from a procedure. They are, of course, preserved around calls to other procedures made from the procedure within which they are declared. Local variables, whether static or not, take precedence over global variables of the same name. Initializer lists are not allowed for local array variables unless they are declared as static. Non-static local scalar variables may be initialized by an expression.

Examples of local variable declarations:

```
static double matrix[10][2];
static double pi = 3.1415926;
int a = 2, b = 3, d = 4;
int g = a*b + c;
double local_array[a * b][d + g];
```

Constants - The form of constants follows C-language conventions. Octal integer constants and special C suffixes, e.g. "L" for long, are not supported. Hexadecimal integer constants and all of the C character constant forms (\n, \t, etc.) are supported, however.

Application Defined Global Variables - A number of global variables are predefined within OSLO and are available for use in CCL procedures or on the command line (see pGlobal variables). Some pre-defined variables are "protected," i.e., they can be used in expressions but their values may not be changed by CCL statements. Examples are the constant **pi** and the conversion factor **dr**, which may be used to convert degrees to radians. The statement "a = 4*pi" is permitted, but "pi = 22/7" is not.

CCL language elements

CCL uses C syntax to form expressions and statements. The following tokens are reserved as delimiters:

{ } ; : ' "

Operators (in order of descending precedence)

Token(s)	Operator(s)
() []	function, subscript
++ --	increment, decrement (postfix)
++ -- ~ ! + - &	increment, decrement (prefix), unary ops
**	exponentiation (not C-compatible)
* / %	multiply, divide, modulus
+ -	add, subtract
<< >>	shift
< <= > >=	relational
== !=	equals, not equals
&	bitwise and
^	bitwise xor
	bitwise or
&&	logical and
	logical or
= *= /= %= += -= <<= >>= &= = ^=	assignment
,	sequential evaluation

CCL reserves several keywords that have the same function as in C:

break	case	char	continue	default	do
double	else	float	for	if	int
long	return	static	switch	void	while

In addition, CCL has other keywords that are special to CCL, most of which relate to CCL's extended argument specifications.

Keyword	Description
cmd	Data type same as C “void”.
env	Command is only valid in specified environment.
hilim	Sets high limit for argument value.
list	Argument specified from a fixed list of values.
lolim	Sets low limit for argument value.
noquery	Causes argument value to be taken as default with no prompt if argument is not supplied.
onprompt	Causes argument to be prompted for with default value suggested.
optional	Argument may be specified but is not required and will not be prompted for.
prompt	Specifies the prompt to use if an argument is required to be entered on the command line.
real	Data type same as C “double”.
setmodule	Next command(s) to be interpreted in specified module.
ulist	Argument specified either from a list or typed-in value.

Comments can be delimited using `/* ... */`, or `//` at the beginning of a line.

CCL reserves the question mark symbol (?) to force an argument prompt. No provision is made in CCL for pointers, structures, or casts, which are not part of the language.

CCL procedures

A CCL procedure has the general format shown below.

```
Datatype Procedure_Name (Argument list)
{
    procedure statement(s)
}
```

The argument list identifies the data items that must be supplied to the procedure. It consists of one or more entries of the form

```
Datatype Argument_Name
```

If more than one argument is specified, the entries in the argument list must be separated by commas. If the procedure has no arguments, the argument list may be omitted entirely or the single keyword “void” may be used for the argument list.

A procedure must be declared prior to invoking it within the body of another procedure. One way of doing this is to place the entire procedure ahead of all

procedures which invoke it. The prototype declaration provides an alternative. It has the form:

```
Datatype Procedure_Name (Argument list);
```

The full procedure may then be placed at any later point within the sequence of source files.

In CCL, a procedure may be declared as static to indicate that it is known only to other procedures. That is, a static procedure may not be invoked from the command line at run-time, but may only be called by other procedures. A procedure is declared to be static by preceding its datatype with the keyword “static,” e.g.

```
static double detval(matrix[][]).
```

“Command line procedures” are procedures that may be directly executed from the OSLO command line. Most often they consist of single command statements or expressions. However, they may consist of any collection of statements that could appear within the body of a CCL procedure, including “while” and “for” loops, although data type declarations may not be made within a command line procedure. Where a terminating semicolon would be required in a formal CCL procedure, it may be omitted in a command line procedure.

Any expression typed into the command line is immediately computed and the result displayed in the message area below the command line. Thus the command line may serve as a calculator as well as a command entry area.

Multiline command procedures may be entered in the CCL text editing window and any block of text within this window may be selected for immediate execution.

Strings

In many programs, certain strings or messages appear repeatedly. Error messages and format statements are common examples. CCL allows you to keep such string data in a file and refer to them using an integer. The various string output functions (**printf**, **sprintf**, **message**, etc.) in CCL can accommodate a string number in place of a string.

The string file has lines of the following format:

```
String_nbr: String_type, String;
```

String_nbr is an integer in the range 1–500, String_type is either **i** (informational), **w** (warning), or **e** (error), and String is a quoted string. A typical example of entries in the string file is

```
...
REAL_OUTFMT:  i, "The result is %f\n";
RANGE_ERR:    e, "Result out of range!";
...
```

It is assumed that REAL_OUTFMT and RANGE_ERR are defined in another file, e.g.

```
...
#define REAL_OUTFMT 57
#define RANGE_ERR   58
```

The **message**, **pause**, and **abort** functions in the CCL library are specially constructed to handle strings of the above type. These functions use the string

type argument to direct its output to either the message area in the main window (**i**, **w**), or to an alert box that it pops up automatically (**e**). Messages of type **w** and **e** produce a beep sound when the message is displayed.

For the **pause** command, strings of type **e** are demoted to type **w**. For the **abort** command, messages of type **i** are promoted to type **w**. The other output functions in CCL (**printf**, **sprintf**, **fprintf**) ignore the string type and only use the value of the string.

The string file must be compiled separately and before any command that uses its contents. OSLO assumes that strings will be found in the file **a_string.ccl**, and compiles this file automatically whenever CCL files are compiled.

CCL differences from C

As mentioned above, CCL is very similar to C. CCL supports all of the C control syntax, except for the conditional expression “(a) ? b : c” and the “goto” statement. Most C operators are supported. Only partial support is provided for C data types (no pointers, structs, or unions), and in a few cases, there are essential differences between CCL and C. In this section, the principal differences between the two languages are described. Some of the items here are also mentioned in other sections of the manual.

Floating point array indices

CCL permits floating point variables or expressions to be used for the array indices. In such cases, the index values are automatically truncated to integer values. Also, in CCL, non-static local array variables may have their dimensions defined by expressions.

Global scope of argument names

In CCL, once an argument is declared its name and type is known globally. Thus, if an argument “int count” is declared in the argument list for a given procedure, it is not permissible to declare an argument “double count” for another procedure.

Static procedures

In CCL, a procedure may be declared as static to indicate that it is known only to other procedures. That is, a static procedure cannot be invoked from the command line at run-time, and can only be called by other procedures.

Optional parentheses in procedure calls

Ordinarily, the arguments of a procedure call are enclosed in parentheses and separated by commas. In a CCL invocation of a command (or void function), the parentheses may be omitted and, if they are, the commas separating the arguments may be omitted as well. Since spaces then serve to separate the arguments, expressions (e.g., $a+b-c$) used as arguments must be packed to contain no white space. If the argument list is enclosed in parentheses and arguments are separated by commas, then expressions used as arguments may contain spaces.

In the case of a call to a function that has no arguments, the parentheses may also be omitted. Thus, if `time()` is a function, the statements “`a = time();`” and “`a = time;`” are equivalent.

Example:

Consider a command whose prototype declaration is:

```
cmd plot (real x1, real y1, real x2, real y2);
```

A call to this command within a procedure might be expressed by any of the following statements:

```
plot(2.4, 3 * x + 4 * sqrt(y), z, z + 1);
plot 2.4, 3*x+4*sqrt(y), z, z+1;
plot 2.4 3*x+4*sqrt(y) z z+1;
```

The last form would most likely be used if the command were invoked from the command line at run-time (in which case the final semi-colon could also be omitted if no additional statements followed).

In situations where C would expect statements to be separated by commas (as in the initialization and end-of-loop statements in a “for” statement) the optional forms are not permitted.

Exponentiation

The power operator (******) is borrowed from FORTRAN. The expression $a^{**}b$ is interpreted as “ a raised to the power b .” a and b may be either integer or double types. If both a and b are integer types and the magnitude of the result is less than 2147483648, the result is of integer type. Otherwise, the result is of double type. If $b = 2$, the result is evaluated by multiplying a by itself. Otherwise, the result is evaluated by the C function call **pow(a, b)**. If $a = 0$ and b is negative, a “divide by zero” error is generated. If a is negative and b is not an integer, an “exponent error” is generated.

The Null Statement

In C, an isolated semicolon (;) is called a null statement and has no effect. In CCL, this null statement may be defined as a built-in OSLO command and therefore may result in an OSLO defined action. At run-time, pressing the Return key when the command line is empty causes a null statement to be executed. In the OSLO command line lens editor, the action is to advance to the next lens surface.

The “Pass by reference” Operator

As in C, CCL normally passes copies of scalar arguments to called procedures and addresses of array arguments to called procedures. Thus, if a called procedure alters a passed scalar variable it does not affect the value of the variable in the calling procedure. Sometimes it is desirable to pass addresses of variables to a procedure so that it can set values of the variables for the caller. In C, the called procedure will declare such arguments as pointer (address) types. CCL does not support explicit pointer types. However, if the name of a scalar variable passed as an argument to a CCL procedure is prefixed by the character “&,” CCL will pass the address of the variable to the called procedure; i.e., the variable will be passed by “reference” rather than by “value.” The called procedure does not know whether its argument is passed by value or by reference.

The ability to pass a scalar variable either by value or by reference is similar to a feature of the C++ language. In CCL it eliminates one of the drawbacks inherent in CCL’s lack of support for explicit pointer types.

Omitted Arguments

When a procedure is invoked either from the command line or from within another procedure, scalar arguments and 1-dimensional character arrays may be omitted.

If any such argument is omitted, the user will either be prompted to enter the argument or a default value will be taken (according to properties that have been associated with the argument). If an integer or double array argument is omitted, a CCL compiler error message will be generated.

The Question Mark argument

If a question mark (?) is used in lieu of an argument when a procedure is invoked, the user is prompted to enter arguments that have default values. If the argument list is terminated immediately following a question mark, any remaining arguments expected by the procedure will be prompted for.

Using the example from Optional parentheses (see p. cmd plot (real x1, real y1, real x2, real y2);), **plot ?** will prompt the user for all of the arguments, while **plot 2.4 ? z z+1** will only prompt the user for the second argument.

Integer Division

In C, the division of one integer by another evaluates to an integer – e.g., 22/7 evaluates to 3. CCL produces a real result in this case – i.e., 22/7 evaluates to 3.142857. Integer truncation can still be performed, however, by equating an integer variable to the result of an integer division. If *j* is an integer variable, the statement “*j* = 22/7” yields a value of 3 for *j*.

Case Label Restrictions

Case labels are restricted to integer constants in the range –32767 to 32767. As is customary in C programs, #define statements may be used to give names to case labels (or any other constants) as illustrated below:

```
#define ALPHA 1
#define BETA 3
switch (value) {
/* Note: CCL provides a library command "printf" that is equivalent to the C library printf function */
    case ALPHA:
        printf("Alpha selected\n");
        break;
    case BETA:
        printf("Beta selected\n");
        break;
    default:
        printf("Nothing selected\n");
        break;
}
```

If the “switch” expression evaluates as a double type it is automatically truncated to an integer.

Two-dimensional arrays

There is an important departure from C in the CCL declaration of a two-dimensional array argument: array[[]]. In C, it is necessary to supply the column dimension – e.g., array[][20]. In CCL, array dimensions are “secretly” passed to procedures. Thus procedures may handle arrays that are variable in both row and column dimensions. CCL also provides library functions (**rowdim**, **coldim**) that may be used to access the integer dimension values for arrays passed as arguments. The availability of array dimensions is not restricted to user defined arrays but extends to OSLO defined arrays as well. This capability, together with

the capability of using expressions instead of constants for the dimensions of local array variables, greatly facilitates operations with matrices.

Examples:

1. A command having a character string (array) and character count as arguments:

```
command prostr(char string[],int count) {
    procedure statement(s)
}
```

2. A function that operates on a 2-dimensional array of floating point values and returns a floating point value (the value of the determinant of a square matrix, for example):

```
double detval(double matrix[][]){
    procedure statement(s)
}
```

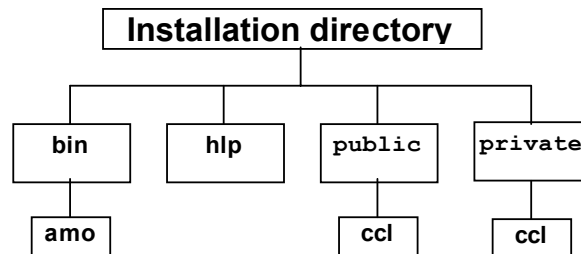
Complete evaluation of logical expressions

In C, logical expressions are evaluated from left to right until the final state of the expression is determined. In some cases, it is not necessary to evaluate the entire expression to determine its final state. In CCL, the entire expression is always evaluated.

Using CCL with OSLO

Files

The file system in OSLO (Windows versions) is anchored to the installation directory, which is set when you install the program. The original installation will place four directories under this directory. In a single user system, it is often satisfactory to maintain this structure. In a multiuser (UNIX) system, it is generally preferable to copy the private directory to each user's \$HOME directory, and to define an environment variable OSDATA that points to this directory.



Source files - CCL source files are given the filetype extension, “.ccl” and are stored in the OSLO public and private directories. Source files may be edited within OSLO and when they are saved, they are automatically compiled and made available for immediate execution. The automatic compilation causes various support files located in the **bin\amo** directory to be restructured and reloaded.

The user must have read-write access to files in the **amo** and **ccl** directories.

It is expected that users will develop CCL programs using the private directory. CCL programs issued by Lambda Research are placed in the public directory. If CCL commands placed in the private directory have the same names and arguments as ones in the public directory, they will replace the public commands. This allows an individual user to customize a standard command without disrupting the system for other users.

Source files may contain one or more procedures. If a procedure calls another procedure, the called procedure must appear earlier in the file or else must have a C-style prototype declaration preceding the calling procedure. A prototype declaration just gives the procedure type, name, and argument list and is terminated by a semicolon. If prototype declarations are used, the actual code for the declared procedure may appear anywhere - even in another **“.ccl”** file.

Several **“.ccl”** file names have special meanings and serve special purposes. All of the following files are duplicated in the public and private directories.

Global file - The file named **a_global.ccl** is always compiled ahead of any other CCL source file. It is a receptacle for any global variable declarations, procedure prototype declarations, and **#defines** that are to be globally known to all CCL procedures.

Since CCL supports the **#include** “filename” statement, you may prefer to place **#define** statements in a separate file - e.g., **a_define.h** - that is included in files that need them by using a **#include** statement. Such an included file should not have the **“.ccl”** filetype.

String file - A number of built-in CCL commands accept character string arguments. Examples are the format strings used in the **printf**, **message**, **pause**, **hpgl**, **label**, **abort** and **halt** commands. While it is permitted to use literal strings (i.e., strings of characters enclosed in double quotes as arguments for these commands, they take up space in the CCL object code. As an alternative to using literal strings, all of the commands that require **printf**-style format strings will also accept an Id Number which identifies a string in the file **a_string.ccl**. When an Id Number is specified in lieu of a literal string, the string is read from the compiled version of the **a_string.ccl** file (which contains a directory of string locations keyed to the string number for quick access).

Argument definitions - Arguments for CCL procedures may be given (optional) extended specifications in the same way as built-in OSLO commands may have extended argument specifications.

Such argument specifications are of the form

```
Datatype Argument_name {condition, condition,...}
```

and appear outside of and ahead of a CCL procedure which uses the argument. It is acceptable to put extended argument definitions in the source files where they are used. However, since the names of CCL arguments have global scope, it is recommended that you place extended argument definitions in the **a_global.ccl** file.

List file - Arguments that are declared as **list** (or **ulist**) type reference a list name where the members of the list are enumerated. The CCL compiler expects that the **a_list.ccl** file will contain all argument lists for user defined arguments.

CCL compiler

CCL files with names ending in “.ccl” may be edited within OSLO and when they are saved, they are automatically compiled and made available for immediate execution, assuming that no errors are encountered during compilation. If errors are found, a message will be generated in the current text window that should prove helpful in locating the source of the error.

To prevent automatic compilation, CCL files should be saved with the suffix “.ccx”. in the **ccl** subdirectory. It is not possible to use CCL while there are errors in a “.ccl” file, so the “.ccx” suffix should be used to tag work in progress. It is also possible to suppress compilation by setting the **ccac** preference to off.

The CCL compiler will act on the public or private directory according to which one receives the saved file. However, if the public CCL is recompiled, all the private directories must also be recompiled, since private CCL is appended to public CCL. If OSLO is used in multiple instances (of the same installed executable program file), it is important not to recompile public CCL while other instances of the program are running, since their CCL files will become obsolete. If the public CCL directory has been recompiled, each user, upon first starting OSLO, will receive notification and will be prompted to recompile the private CCL.

The partitioning of CCL into public and private directories allows users who have a great deal of CCL to move large private CCL programs to the public directory, to prevent recompilation of the large programs whenever a small private CCL program is modified.

Although the CCL compiler is normally run automatically, it is also possible to run it manually using the **ccl** command. Arguments are public, private, followed by the file type: commands, menu, list, string, all. This mode of operation is most useful when CCL programs are developed using an external editor.

The **compile (ccl)** cmd has a fourth optional argument, char Ccl_opts[]. Since this is optional, the user will never be prompted for it. If it is typed in with the command, however, it will be appended to the command string that is executed. This string argument would be used for specifying special compile options (e.g., –g size –l size –c size).

Example: **ccl public all “–g 500000 –l 200000”**

The options which might be specified in the Ccl_opts string are as follows. In each case, “size” is an integer byte count.

g size: specifies the maximum amount of space to be allocated for storage of CCL global variables and constants. The CCL compiler allocates this amount of space for global storage at startup. The space allocated for global storage in OSLO, however, is only the amount actually needed for the global variables and constants encountered during the CCL compilation. The maximum value for size is 8388608. The default size is 1048576.

l size: specifies the amount of stack space allocated in OSLO for temporary storage of local variables and scalar arguments passed to called CCL commands or functions. Upon return from a called CCL command or function, any stack space used becomes free for reuse. The maximum size allowed is 8388608. The default size is 262144.

c size: specifies the amount of storage allocated in OSLO for retaining compiled CCL code in memory for reuse. This must be at least as large as the size of the longest compiled CCL procedure. The CCL compiler will fail if a compiled

procedure exceeds this size. A large size will allow more CCL code to be “cached” in memory and will reduce access to code from disk storage. The maximum value for size is 8388608. The default size is 131072.

-D name [=value] defines the constant “name” with value “value”. This is equivalent to using **#define** in the CCL code.

Setmodule

When constructing a program, there are situations where it is convenient to have several commands that have the same name, but take different arguments and are valid at different times. For example, in the context of optical design, it is desirable to have a single keyword that controls, say, radius of curvature, regardless of whether the context is lens entry, configuration entry, or global editing. OSLO handles situations such as this through the use of “modules.” In particular, in OSLO there is an “L” module that contains the lens entry commands and a “C” module that contains the configuration data commands. Thus, continuing our example, there are actually three “**rd()**” commands in OSLO: one in the L module, one in the C module, and one that is not in a module. The particular command that is executed when “**rd()**” is executed depends upon the current module, if any. The global editing form is the default form if no explicit module has been declared.

In CCL commands, the “setmodule” keyword can be used to distinguish among the various forms of the command. If you want to use a particular module form of a command, you must include the statement “setmodule L;” or “setmodule C;” as part of the function. Setmodule tells the compiler which command is to be used. For example, to edit the configuration data for the radius of surface 3 in configuration 2, the CCL code would be of the form:

```
{
  configuration_edit(update);
  setmodule C;
  rd(ins, 3, 2, 25.0);
  end();
}
```

Error handling in CCL and SCP

Ordinarily, a CCL or SCP procedure terminates with an error message if an error is detected while a built-in command is being processed. It is possible to install a custom procedure to handle errors generated by built-in commands. Such a procedure is specified by the command:

install_error_handler (char Error handler name[])

The use of an error handler allows a CCL or SCP procedure to take special action and continue running when a called built-in command produces an error.

Install_error_handler specifies the name of a CCL or SCP command procedure to be called whenever a built-in command returns an error code. The error handler has no arguments and does not return a value – i.e., it is a command, not a function. If the handler is an SCP procedure, the first character of the specified error handler name must be an asterisk (*).

When the error handler is called, a global CCL integer variable, *Errno*, is set to the error code returned by the built-in command. The error message may be printed from within the error handler by executing “**printf(Errno),**” or may be issued as a

message by executing “**message**(Errno).” If the error handler sets *Errno* = 0 prior to returning, processing of the current command will continue – i.e., the error will be ignored. Otherwise, command processing will terminate immediately after returning from the error handler.

An error handler specified by the **install_error_handler** command remains in use until deactivated by the command:

delete_error_handler(void)

When the **delete_error_handler** command is executed, error handling reverts to the default behavior of terminating the procedure after issuing an error message.

Unless an error handler is intended to be a general replacement for the default behavior, it should be deleted before returning from the procedure within which it was installed. The current error handler name is in the global variable **error_handler**.

The following is a simple error handler in SCP:

```
*errhand
printf("Entering error handler...\n");
printf(Errno);
printf("Continuing...\n");
Errno = 0;
```

User-defined support routines

OSLO Premium allows user-defined surfaces, gradients, and eikonals. All versions of OSLO allow user-defined operands. When the program needs to calculate data of one of the above types, it tests a flag indicating the presence of a user-defined routine, which replaces the normal internal routine for the calculation. User-defined support routines are normally written in CCL, although for highest performance they can be compiled C routines in a dynamic-link library, as described on p. Dynamic-Link Libraries (DLL's). Support routines can be written as SCP procedures if fast computational speed is not required.

User-defined surfaces

To designate a surface as a user-ray trace (**usr**) surface, activate the special options button in the surface data spreadsheet and select User Surface from the pop-up list. You will be prompted for the number of user data items you wish to define for the surface.

In the spreadsheet, you need to enter the name of the CCL or DLL command (**unm**) that performs the ray trace for the surface. The name of the command should be entered in the Raytrace command cell. You can also enter values for

the user data items **uti**. The values of the user data items are available to the CCL ray trace command, and provide a way to pass parameters to the CCL command.

Surface 2					
User ray trace surface		Number of coefficients: 10			
Ray trace command: XXXXXXXXXX					
UT1	UT2	UT3	UT4	UT5	
0.000000	0.000000	0.000000	0.000000	0.000000	
UT6	UT7	UT8	UT9	UT10	
0.000000	0.000000	0.000000	0.000000	0.000000	
Delete User Ray Trace Surface					

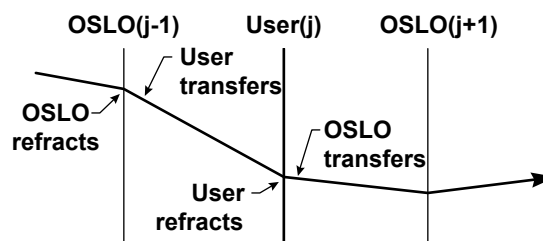
OSLO's internal ray trace computes the ray trajectory and optical path length up to the surface immediately preceding the user surface, interrupts its trace, and calls the user-written CCL ray trace routine specified for the surface.

Variable	Interpretation
Utr_rnl	refractive index on incident side of usr surface at wavelength of current ray
Utr_rnr	refractive index on refracted side of usr surface at wavelength of current ray
Utr_x	x coordinate of ray before transfer
Utr_y	y coordinate of ray before transfer
Utr_z	z coordinate of ray before transfer
Utr_k	x direction cosine of ray before transfer
Utr_l	y direction cosine of ray before transfer
Utr_m	z direction cosine of ray before transfer
Utr_pth	optical path length of ray before transfer
Utr_wgt	weight of ray before transfer
Utr_sfc	surface number of usr surface (integer)
Utr_cv	curvature of usr surface
Utr_cc	conic constant of usr surface
Utr_toric	toric flag for usr surface: no toric data = 0, y-toric surface = 1, x-toric surface = 2 (integer)
Utr_cvtor	value of toric or biconic x-z azimuth curvature of usr surface
Utr_cctor	x-z azimuth conic constant of usr surface
Utr_pftyp	type of aspheric surface profile of usr surface: 10 th order polynomial = 0, asr = 1, asx = 2, isr = 3, trx = 4, try = 5, bic = 6, cnr = 7, cnx = 8, zsr = 9, zsx = 10 (integer)
Utr_npcf	number of aspheric surface polynomial coefficients on usr surface (integer)

Utr_pcf [Utr_npcf]	array [0, ..., Utr_npcf – 1] of aspheric surface polynomial coefficients: Utr_pcf[0] is lens data AS0 , Utr_pcf[1] is lens data AS1 , Utr_pcf[2] is lens data AS2 or ad , etc.
Utr_nucf	number of user defined ut/ items on usr surface (integer)
Utr_ucf [Utr_nucf]	array [0, ..., Utr_nucf – 1] of user defined ut/ items: Utr_ucf[0] is lens data ut1 , Utr_ucf[1] is lens data ut2 , etc.
Utr_wvn	wavelength number (base 0) of current ray (integer)
Utr_wvl	wavelength of current ray
Utr_fbx	fractional x-object height of ray
Utr_fby	fractional y-object height of ray
Utr_fbz	fractional z-object height of ray
Utr_fx	fractional x-pupil coordinate of ray
Utr_fy	fractional y-pupil coordinate of ray

OSLO supplies the ray coordinates, direction cosines after refraction, the optical path, the ray weight, and additional ray, surface, and user-defined surface information to the CCL command via the above predefined CCL global variables. Unless otherwise noted, the variables are of type *double*. In addition to the above variables, all surface and system data items that are available to CCL can, of course, be accessed by the user's ray trace routine. The task and responsibility of the user's CCL ray trace command is two-fold:

1. Transfer the ray to the **usr** surface, i.e., find the intersection of the ray with the **usr** surface and compute the path length along the ray.
2. Refract the ray at the **usr** surface.



After the ray is refracted at the **usr** surface, the user ray trace command should then assign new values for the ray coordinates and direction cosines (with respect to the local coordinate system of the **usr** surface), the path length (not optical path) from the surface immediately preceding the **usr** surface, and the ray weight to the CCL global variables in the following table.

Variable	Interpretation
Utr_err	ray error flag(integer): 0 when control passed to CCL.
Utr_x	x coordinate of ray at usr surface
Utr_y	y coordinate of ray at usr surface
Utr_z	z coordinate of ray at usr surface
Utr_k	x direction cosine of ray after refraction at usr surface
Utr_l	y direction cosine of ray after refraction at usr surface
Utr_m	z direction cosine of ray after refraction at usr surface
Utr_pth	path length of ray during transfer
Utr_wgt	weight (intensity) of ray after transfer and refraction

Upon completion of the execution of the user CCL command, OSLO regains control of the ray trace and uses the new values of the above variables to continue the ray trace through the system (or to the next **usr** surface, where the user surface ray trace process would be repeated). In the event that the user ray trace cannot trace the ray successfully through the **usr** surface, it should set the value of the variable **Utr_err** to a non-zero integer value. If **Utr_err** is non-zero when control of the ray trace is returned to OSLO, the ray trace (for that ray) is terminated.

A sample user ray trace is the CCL command **utrace** in the file `usrraytr.ccl` in the `public/ccl` directory.

User-defined gradients

OSLO Premium supports a user-defined gradient surface, type UGR. The UGR gradient allows you to define an arbitrary number of coefficients for use in computing the refractive index. When a UGR surface is encountered in the ray trace, control is passed to a CCL command that computes the refractive index, and its first derivatives with respect to x , y , and z , at the supplied point (x, y, z) in the medium. These values are transmitted to OSLO in the CCL global variables listed in the table below.

Variable	Interpretation
Ugr_err	user gradient error flag (integer)
Ugr_index	refractive index of medium at (x, y, z)
Ugr_dndx	derivative of the index with respect to x , i.e., dn/dx
Ugr_dndy	derivative of the index with respect to y , i.e., dn/dy
Ugr_dndz	derivative of the index with respect to z , i.e., dn/dz

In the event that the user gradient CCL command cannot compute the refractive index and/or first derivatives of the index, it should set the value of the variable **Ugr_err** to a non-zero integer value. If **Ugr_err** is non-zero when control of the ray trace is returned to OSLO, the ray trace (for that ray) is terminated.

The coefficients that describe the user gradient (**ugri**) are stored in a one-dimensional array. The spreadsheet for the UGR gradient type appears as follows:

Surface 2					
GDT UGR	10	Step Size:	0.100000		
User-defined gradient command: XXXXXXXXXX					
UGR0	UGR1	UGR2	UGR3	UGR4	
0.000000	0.000000	0.000000	0.000000	0.000000	
UGR5	UGR6	UGR7	UGR8	UGR9	
0.000000	0.000000	0.000000	0.000000	0.000000	
Delete GRIN Medium					

For the user-defined gradient, the CCL variables set by OSLO are the following:

Variable	Interpretation
Ugr_first	flag set to 1 on the first call for each ray at a given surface, and 0 thereafter (integer)
Ugr_wvn	wavelength number (base 1) of current ray (integer)
Ugr_sfc	surface number
Ugr_m0	base index (n_0) of medium
Ugr_x	x coordinate of ray, relative to surface vertex
Ugr_y	y coordinate of ray, relative to surface vertex
Ugr_z	z coordinate of ray, relative to surface vertex
Ugr_nugrcf	number of user GRIN coefficients (integer)
Ugr_ugrcf[Ugr_nugrcf]	array [0, ... , Ugr_nugrcf – 1] of user GRIN coefficients

User-defined eikonals

The eikonal surface allows you to specify an optical component by its eikonal (characteristic) function, rather than construction data (radii, thicknesses, etc.). If you are not familiar with optical design using eikonals, please see Chapter 5 in the Optics Reference manual.

To designate a surface as an eikonal (**eik**) surface, activate the special options button in the surface data spreadsheet and select Eikonal Surface from the pop-up menu.

Surface 2					
Eikonal type:		Point	Number of coefficients:		10
Eikonal raytrace command:					
	EI0	EI1	EI2	EI3	EI4
	0.000000	0.000000	0.000000	0.000000	0.000000
	EI5	EI6	EI7	EI8	EI9
	0.000000	0.000000	0.000000	0.000000	0.000000
Delete Eikonal Surface					

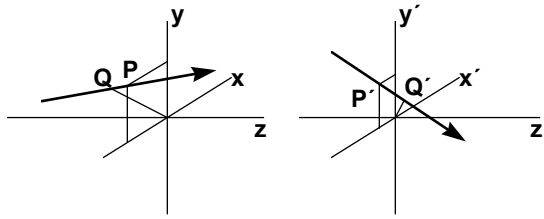
To select the type of eikonal to be used, click the Eikonal type button. A pop-up menu will be displayed, from which you can choose one of the four eikonal types (i.e., Point, Point-Angle, Angle, or Angle-Point). The name of the CCL command or DLL command that computes the eikonal is entered in the spreadsheet cell labeled Eikonal ray trace command. An eikonal surface may be deleted by clicking the Delete Eikonal Surface button.

When an eikonal surface is defined, a number of user-specified data items may be assigned to the surface (similar to the user ray trace surface). These data items are available to the CCL command that computes the eikonal, and may be used to “pass” parameters to the command. If n data items are defined, they are designated **EI0**, **EI1**, ..., **EI($n-1$)**, in the eikonal surface spreadsheet and in the display of eikonal surface data. The items **EI i** ($0 \leq i < n$) may be designated as variables and used as operand components.

In general an eikonal is specified relative to two reference planes, which we denote by (x, y) and (x', y') . The (x, y) plane is in the object space of the eikonal surface, and the (x', y') plane is in the image space of the eikonal surface. In OSLO, for simplicity, both planes are located at the position of the eikonal surface. A ray in the object space of the eikonal surface is specified by four parameters: the coordinates (x, y) of the point of intersection of the ray with the object space reference plane and the direction cosines K and L with respect to the x and y axes. (The object space z coordinate is always 0, since the reference surface is a plane, and the z -direction cosine M is not needed since $K^2 + L^2 + M^2 = 1$.) Similarly, in the image space of the eikonal surface, the ray is specified by its coordinates (x', y') and direction cosines K' and L' .

The optical properties of the component are specified by an eikonal between the reference planes. There are four commonly used eikonals, all of which can be used in OSLO:

- Point eikonal $S(x, y, x', y')$, the optical path from P to P' .
- Point-angle eikonal $V(x, y, K', L')$, the optical path from P to Q' .
- Angle eikonal $W(K, L, K', L')$, the optical path from Q to Q' .
- Angle-point eikonal $V'(K, L, x', y')$, the optical path from Q to P' .



Given an eikonal, the other ray parameters can be computed from derivatives of the eikonal. For example, using the point eikonal,

$$\frac{\partial \mathcal{S}(x, y, x', y')}{\partial x} = -nK, \quad \frac{\partial \mathcal{S}(x, y, x', y')}{\partial y} = -nL \quad (16.1)$$

$$\frac{\partial \mathcal{S}(x, y, x', y')}{\partial x'} = n'K', \quad \frac{\partial \mathcal{S}(x, y, x', y')}{\partial y'} = n'L' \quad (16.2)$$

In the above equations, n and n' are the refractive indices in the object and image spaces of the eikonal surface.

Ray tracing an eikonal surface requires the calculation of the image space ray parameters (x' , y' , K' , L'), given its object space ray parameters (x , y , K , L). Using the point eikonal means, then, solving Eqs. (16.1) for x' and y' , and then using Eqs. (16.2) to calculate K' and L' .

As mentioned earlier, both reference planes for the eikonal are taken to be at the defined location of the surface on which the eikonal data is placed. When the ray trace reaches this surface, the specified eikonal ray trace command is used, along with Newton-Raphson iteration, to solve Eqs. (12.1) (or their equivalent, depending upon the type of eikonal). If the command executes successfully, OSLO continues the ray trace through the remainder of the optical system; otherwise the tracing of that ray will be terminated.

The object space parameters of the ray are available to the CCL command from global CCL variables that are set by OSLO. The names and descriptions of the CCL variables are given in the following table. Unless indicated otherwise, the variable is of type *double*.

Variable	Interpretation
Eik_err	error flag; if Eik_err is non-zero upon returning from the CCL command, the ray trace for that ray will be terminated (integer)
Eik_wvn	wavelength number, base 0, of the ray (integer)
Eik_ncf	number of eikonal surface coefficients (integer)
Eik_cof[Eik_ncf]	array [0, ..., Eik_ncf – 1] of eikonal surface coefficients
Eik_x	x coordinate of the ray, incident on the eikonal surface (i.e., on the object space reference plane)
Eik_y	y coordinate of the ray, incident on the eikonal surface (i.e., on the object space reference plane)
Eik_k	x direction cosine in object space
Eik_l	y direction cosine in object space
Eik_xp	x coordinate of the ray, exiting the eikonal surface (i.e., on the second reference plane)
Eik_yp	y coordinate of the ray, exiting the eikonal surface (i.e., on the second reference plane)
Eik_kp	x direction cosine in image space
Eik_lp	y direction cosine in image space
Eik_path	the eikonal, computed as a function of the appropriate variables, i.e., the optical path between the reference planes

The essential task for the CCL command is to compute the value of Eik_path. Eik_path should be a function only of the ray variables appropriate for the designated type of eikonal. For example, if you have defined a point eikonal, Eik_path should be a function of Eik_x, Eik_y, Eik_xp, and Eik_yp.

Upon completion of the execution of the eikonal CCL command, OSLO regains control of the ray trace and continues the ray trace through the system. If the ray cannot be traced through the eikonal surface, the CCL command should set the value of the variable Eik_err to a non-zero integer value. If Eik_err is non-zero when control of the ray trace is returned to OSLO, the ray trace (for that ray) is terminated.

User-defined operands

User-defined operands are computed using an SCP or CCL command and placed in the **ocm[]** array. User-defined operands are the only type available in OSLO

Light. Procedures and examples of SCP operands are given in the OSLO User's Guide.

The name of the SCP or CCL command that computes user-defined operands is entered using the command **opoc**, or entered in the optimization operating conditions spreadsheet (see Chapter 5). The default value for **opoc** is **oprds**. The following is a CCL command for computing user-defined operands, which will compute **Ocm[0]**, **Ocm[1]**, and **Ocm[2]** when **opoc** is set to "oprds_spot_size".

```

/*****
Sample command for computation of CCL operands.
This command calculates RMS R spot size for wavelength 1
at three field points.
*****/

cmd oprds_spot_size(void)
{
    set_preference(output_text, off);
    trace_ref_ray(0.0);
    ssbuf_reset();
    spot_diagram(mon, 10.0);
    Ocm[0] = c4;
    trace_ref_ray(0.7);
    ssbuf_reset();
    spot_diagram(mon, 10.0);
    Ocm[1] = c4;
    trace_ref_ray(1.0);
    ssbuf_reset();
    spot_diagram(mon, 10.0);
    Ocm[2] = c4;
    set_preference(output_text, on);
}

```

Dynamic-Link Libraries (DLL's)

It is possible for users to write in C (or any suitable compiled language) procedures for external operands calculations, user-surface ray tracing (standard and gradient-index), and eikonal ray tracing. Through the mechanism of dynamic-link libraries (DLL's; shared libraries), it is possible to obtain the high speed of compiled code with relatively little extra effort, as compared to performing these operations in CCL.

The process of using dynamic-link libraries consists of three steps: writing the source code (or translating it from CCL), compiling and linking the source code into a DLL, and specifying the DLL and procedure names in OSLO.

Writing DLL source code

To write the DLL source code it is necessary to understand the mechanisms that OSLO uses to implement user-defined surfaces, GRIN materials, operands, and eikonals in CCL. Typically, OSLO invokes the CCL routines by first setting the values of certain CCL global variables and then calling the user-supplied CCL command; the CCL command then sets the values of these (or other) global variables and returns. For example, OSLO implements the CCL user-surface raytrace by setting the values of the user-raytrace CCL global variables (whose names begin with "Utr_"); OSLO then calls the CCL user-raytrace command

which is responsible for calculating the values of the Utr_x, Utr_y, Utr_z, Utr_k, Utr_l, Utr_m, Utr_pth, and Utr_wgt global variables.

The mechanism by which the user-supplied procedures in DLL's are invoked is somewhat different: OSLO fills the contents of C structures or arrays and passes pointers to the structures or arrays to the user-supplied procedures in the DLL's. For example, the structure whose pointer is passed to the user-GRIN raytrace procedure is called **Ugstruct**. **Ugstruct** contains a number of fields, one of which, x, is a pointer to the Ugr_x variable within OSLO. Whereas a CCL user-GRIN procedure is responsible for setting the value of Ugr_x, the procedure in a DLL is responsible for de-referencing the pointer x and setting the value of the dereferenced pointer. For convenience, a set of macros is supplied with OSLO to ease the transition from CCL to C; for example:

```
#define Ugr_x (*(Ugr->x))
```

where Ugr is the name of the pointer to the Ugstruct that is passed to the DLL procedure.

The source file for any DLL should include the oslodll.h file (supplied with OSLO), that contains:

definitions that are used to access OSLO's global variables, including lens surface and system data

definitions of structures used by DLL's and by OSLO

a set of convenience C preprocessor macros to ease translation of procedures from CCL to C

This #include directive should then be followed by prototypes for the functions in the DLL, DLL global variable declarations, and finally the functions themselves.

Note that functions that are to be called by OSLO should be declared as

```
void OSLOAPI <FunctionName>(<arguments>);
```

After compilation, the DLL source code should be linked with the supplied object file oslodll.o (UNIX) or oslodll.obj (Windows). This object file contains functions for initializing the DLL, accessing OSLO's global variables, executing commands within OSLO, and accessing OSLO's spreadsheet buffer. The source code and a makefile for oslodll.o(bj) are also provided in case it is necessary to recompile them.

Source code and makefiles for examples of the four types of DLL and sample lens files that use them are located in the bin\dll subdirectory of the directory where OSLO is installed. The DLL's themselves are in the bin subdirectory.

Accessing OSLO global variables from DLL's

A set of eight functions is defined for accessing OSLO's global variables:

```
char GetCharGV(int vnum, int arg1, int arg2) - returns the value
of a character global variable
```

```
void SetCharGV(char c, int vnum, int arg1, int arg2) - sets the
value of a character global variable
```

```
long GetLongGV(int vnum, int arg1, int arg2) - returns the value
of long integer global variable
```

```

void SetLongGV(long l, int vnum, int arg1, int arg2) - sets the
    value of a long integer global variable
double GetDoubleGV(int vnum, int arg1, int arg2) - returns the
    value of a double (real) global variable
void SetDoubleGV(double d, int vnum, int arg1, int arg2) -
    sets the value of a double (real) global variable
char *GetStringGV(int vnum, int arg1, int arg2) - returns a
    pointer to a string global variable
void SetStringGV(char *s, int vnum, int arg1, int arg2) - sets
    the value of a string global variable

```

where

c is a character value
l is a long integer value
d is a double (real) value
s is a pointer to a string value
vnum is the variable identifier constant (defined in `oslodll.h`) consisting of the variable name (in upper case), prepended with `GV_`. For example, the identifier for the `lms` (image surface number) global variable is `GV_IMS`.
arg1 and *arg2* are typically array subscripts, used as surface numbers, wavelength numbers, etc. *arg1* and *arg2* are ignored when accessing scalar variables, and *arg2* is ignored when accessing one-dimensional array variables.
 For example, to assign the value of the curvature of surface 5 to the C variable `curv5`:

```
curv5 = GetDoubleGV(GV_CV,5,0);
```

Accessing the Spreadsheet Buffer from DLL's

OSLO's Spreadsheet Buffer can be accessed from DLL's with the `GetSSBValue()` function. This function takes two integer arguments, the row and column of the buffer cell, and returns the value of that cell. It is similar to CCL's `ssb()` function. For example, the following statement gets the value of the cell at row 5, column 6 of the spreadsheet buffer:

```
value = GetSSBValue(5,6);
```

Executing OSLO commands from DLL's

The `ExecOSLOCommand()` function takes a string argument containing an OSLO command to be executed. For example, the following statement traces a reference ray at full field and then traces a spot diagram:

```
ExecOSLOCommand("trr 0; spd");
```

This function is used to execute commands that are in OSLO but not in the C runtime library.

Creating DLL's

After the DLL source code is written, it must be compiled and then linked with `osldll.o(bj)` and any other runtime libraries required by the DLL. The sample makefiles can be used as templates for your own makefiles. The resulting DLL should be placed in the `bin` subdirectory of either `OSROOT` (the OSLO installation directory) or `OSDATA` (the user's private directory).

External operands procedures

CCL operands procedures are responsible for calculating the values the elements of the global array `ocm[]`, which are referred to in the operands set as **OCM<n>**, where **<n>** is the index into the `ocm[]` array. Similarly, external (DLL) operands procedures are responsible for calculating the values the elements of the array `ecm[]` that is passed to the procedure. The elements of `ecm[]` are referred to in the operands set as **ECM<n>**, where **<n>** is the index into the `ecm[]` array.

The external operands procedure is passed two arguments: the maximum number of external operand components available (i.e., the dimension of the external operand component array) and a pointer to the external operand component array. For example, an operands procedure called `spotsizes` could be defined as follows:

```
void OSLOAPI spotsizes(int nexops, double *ecm)
{
    ExecOSLOCommand("set_preference(output_text, off)");
    ExecOSLOCommand("trace_ref_ray(0.0)");
    ExecOSLOCommand("ssbuf_reset()");
    ExecOSLOCommand("spot_diagram(mon, 10.0)");
    ecm[0] = GetSSBValue(4,3); /* c4 */
    ExecOSLOCommand("trace_ref_ray(0.7)");
    ExecOSLOCommand("ssbuf_reset()");
    ExecOSLOCommand("spot_diagram(mon, 10.0)");
    ecm[1] = GetSSBValue(4,3); /* c4 */
    ExecOSLOCommand("trace_ref_ray(1.0)");
    ExecOSLOCommand("ssbuf_reset()");
    ExecOSLOCommand("spot_diagram(mon, 10.0)");
    ecm[2] = GetSSBValue(4,3); /* c4 */
    ExecOSLOCommand("set_preference(output_text, on)");
}
```

This example, which is identical in function to the CCL operands procedure described on pUser-defined operands, uses the **ExecOSLOCommand** function to execute OSLO commands. The outputs of these commands are retrieved from the spreadsheet buffer with the **GetSSBValue** function and are assigned to the first three elements of the supplied `ecm[]` array.

After the DLL is successfully compiled and linked, it is made available to OSLO in much the same manner as CCL operands procedures: the name of the procedure, followed by the name of the DLL in square brackets, is entered as the "Command for CCL/SCP operands" optimization operating condition. In this example, the procedure name is "spotsizes" and the DLL name is "exops.dll," so the operating condition is entered as "spotsizes[exops]."

Note that you should not type the “dll” or “.sl” extension for the DLL file name. If OSLO is unable to locate the DLL or the specified procedure in the DLL, an error message is displayed.

The operands calculated by the DLL are now available as “ECM0,” “ECM1,” etc., in the operand set.

```
*OPERANDS
OP   DEFINITION      MODE  WGT (TGT)  NAME      VALUE    %CNTRB
O 1   "ECM0"          M     1.000000      0.005113    4.02
O 2   "ECM1"          M     1.000000      0.016893   43.88
O 3   "ECM2"          M     1.000000      0.018406   52.10
MIN ERROR:      0.014723
```

External user-surface raytrace procedures

External user-surface raytrace procedures are passed six arguments:

1. a pointer to a UTRSTRUCT structure, which is defined as follows:

```
typedef struct UTRSTRUCT
{
double  fbx;
double  fby;
double  fbz;
double  fx;
double  fy;
double  x;
double  y;
double  z;
double  k;
double  l;
double  m;
double  pth;
double  wgt;
double  *rnr;
double  *rni;
int     sfc;
int     wvn;
int     err;
} UTRSTRUCT;
```

2. the number of user-defined coefficients (integer)
3. a pointer to an array containing the user-defined coefficients (double)
4. the profile type (an integer between 0 and 10 - see Utr_pftyp in OSLO documentation)
5. the number of aspheric polynomial coefficients (integer)
6. a pointer to an array containing the aspheric polynomial coefficients (double)

Most of the members of the UTRSTRUCT structure are identical to the corresponding Utr_* CCL global variables; for example, the fbx member is identical to the CCL global variable Utr_fbx, the k member is identical to Utr_k, and so on. Many of the Utr_* CCL global variables, such as Utr_cv, are not in the UTRSTRUCT structure; they must be obtained by other means, such as using the `GetDoubleGV()` function. Also note that the **rnr** and **rni** members of the UTRSTRUCT structure are pointers to arrays of refractive indices (the number of elements in the arrays is equal to the number of wavelengths); the current wavelength number is given by the **wvn** member.

As with CCL user-surface raytrace procedures, the DLL raytrace procedure is responsible for calculating appropriate values of x , y , z , k , l , m , pth , and wgt before returning.

The name of the raytrace procedure and the DLL are specified in the user-defined surface dialog box (available from the Special button in the Surface Data spreadsheet):

The name of the procedure is given, followed by the name of the DLL (without the “.dll” or “.sl” extension) in square brackets.

External user-defined GRIN media

It is possible to trace rays through gradient-index media other than the types built into OSLO by defining a user-gradient surface. This is accomplished by providing a procedure that computes the refractive index and its spatial derivatives at specified points in the medium. Note that this procedure need not actually trace rays, but simply provide index data for OSLO's built-in gradient-index raytrace. User-gradient procedures in DLLs are passed only one argument, a pointer to a UGRSTRUCT structure:

```
typedef struct UGRSTRUCT
{
    long    *err;
    long    *wvn;
    long    *sfc;
    double  *rn0;
    double  *x;
    double  *y;
    double  *z;
    long    *nugrcf;
    double  *ugrcf;
    double  *index;
    double  *dndx;
    double  *dndy;
    double  *dndz;
} UGRSTRUCT;
```

The members of this structure are **pointers** to the corresponding CCL global variables (e.g., the x member is a pointer to the Ugr_x variable in OSLO). As in the CCL user-defined GRIN procedure, the DLL procedure is responsible for calculating the $index$, $dndx$, $dndy$, and $dndz$ values.

The name of the user-defined GRIN procedure and the DLL are specified in the user-defined gradient dialog box (available from the Special button in the Surface Data spreadsheet). The name of the procedure is given, followed by the name of the DLL (without the “.dll” or “.sl” extension) in square brackets.

External eikonal raytrace procedures

Eikonal surfaces allow the description of lens elements purely in optical terms. The eikonal raytrace procedure is responsible for computing the optical path length of a ray between two reference planes, given four of the following data: coordinates x and y on the first plane, direction cosines K and L incident on the first plane, coordinates x' and y' on the second plane, and direction cosines K' and L' exiting the second plane. The DLL eikonal raytrace procedure is passed a pointer to an EIKSTRUCT structure:

```
typedef struct EIKSTRUCT
{
    long    *err;
    long    *wvn;
    long    *sfc;
    double  *x;
    double  *y;
    double  *k;
    double  *l;
    double  *xp;
    double  *yp;
    double  *kp;
    double  *lp;
    double  *path;
    long    *ncf;
    double  *cof;
} EIKSTRUCT;
```

The members of this structure are **pointers** to the corresponding CCL global variables (e.g., the member `xp` is a pointer to the `Eik_xp` global variable in OSLO). The eikonal raytrace procedure is responsible for computing the `path` member, given four of `x`, `y`, `k`, `l`, `xp`, `yp`, `kp`, and `lp`.

The name of the eikonal raytrace procedure and the DLL are specified in the eikonal surface dialog box (available from the Special button in the Surface Data spreadsheet).

The name of the procedure is given, followed by the name of the DLL (without the “.dll” or “.sl” extension) in square brackets.

Load_library

Attaches to OSLO the dynamic-link (shared) library whose file name is given by `Library_name`. The library file should be in the bin subdirectory of the OSLO installation directory or the user's Private directory. If the library cannot be found, the return value of this function is zero. If the library is loaded successfully, the `InitLibrary()` function in the library will be called. If `InitLibrary()` is called successfully, the return value is a “handle” to the library that can be used in calls to the `Get_proc_handle` function and the `Free_library` command; otherwise, the return value is zero. The `Free_library` command should be called with this handle to detach the library from OSLO when the library is no longer needed.

```
int load_library
    (Library_name)
char Library_name[]
```

Free_library

Detaches from OSLO the dynamic-link (shared) library whose handle is given by `Library_handle`, obtained by a call to the `Load_library` function. This command should be called when the library is no longer needed. The library handle should not be used after this command is called.

```
cmd free_library
    (Library_handle)
int Library_handle
```

Get_proc_handle

Obtains a “handle” to a function within a dynamic-link (shared) library that has been attached to OSLO with the Load_library command. Library_handle is the return value from a call to Load_library, and Procedure_name is the name of the function in the library to be accessed (note that the procedure name is case-sensitive). If the specified procedure is found, the return value is a “handle” to the library routine that can be used in calls to the Call_external command; otherwise, the return value is zero.

```
int get_proc_handle
    (Library_handle,
     Procedure_name)
int  Library_handle
char Procedure_name
```

Call_external

Calls the dynamic-link (shared) library routine whose handle is given by Proc_handle. Proc_handle is a “handle” to a library routine returned by the Get_proc_handle function. The remaining arguments to this command are passed to the library routine, and may be of type char, int, double, or arrays of char (i.e. strings), int, or double. Note that OSLO determines, at run-time, the type of each argument, so care must be taken that integer arguments are passed as integers and double arguments appear as doubles. For example, the integer argument with a value of two should appear as 2, and the double argument with value two should appear as 2.0.

```
cmd call_external
    (Proc_handle,
     Vlist)
int  Proc_handle
```

Overview

OSLO contains over 100 general support routines that are not specifically related to optics. Although some of the routines are restricted to CCL programs, the majority can be executed from SCP also. In the following sections, the routines are described according to their general function: Window management, File & I/O, Math, Strings, Graphics, and Miscellaneous.

Window management

ewc
ewo

editwin_close
editwin_open

The **editwin_open** command causes the text editing window to appear and makes it available for text editing. The **editwin_close** command causes the text editing window to disappear from the screen.

If **editwin_open** is executed from the command line while the command line has the keyboard focus, the keyboard focus passes to the text editing window. When the text editing window is closed, its current contents are retained so that if it is subsequently reopened it will re-display its prior contents for editing.

definitions

```
cmd editwin_open(void)
cmd editwin_close(void)
```

gsa

graphwin_sliderassign(...)

The **graphwin_sliderassign** command assigns a slider to a special slider window. Up to nine sliders may be assigned to the window. Movement of the slider by use of the system mouse causes a user written CCL procedure to be called. The called procedure would normally cause all or a portion of the current graphics window to be redrawn in response to the new value of the slider (which is supplied as an argument to the procedure).

A slider is a graphical object. It acts much like a small horizontal scrollbar that takes on a numeric value according to its position. It is manipulated with the system mouse. The current value is always displayed and whenever its value changes, a user written CCL callback procedure is executed. The label, specified by the Slider Label argument is displayed to the left of the slider.

Each slider is assigned a unique Slider Id Number. Whenever the CCL callback procedure is called it receives two arguments: the first argument is an integer Slider Id Number and the second argument is a real Slider Value. Thus, the user must write a CCL command which has the following form:

```
cmd callback_name(int slider_id, real slider_value)
{
```

```
// procedure code
}
```

In the **graphwin_sliderassign** command, the `callback_name` is specified by the Callback Command argument while the `slider_id` to be returned to the callback procedure is specified by the Slider Id Number argument. The Slider Id Number can be any integer number, but should be different for each slider assigned to the same graphics window. It may be used in the callback procedure to identify the particular slider which was activated when more than one slider uses the same callback.

A slider may display either integer numbers or real numbers (which are displayed with a decimal point). This choice is indicated by the Value Type argument. If the argument is omitted, the value type defaults to a real number. Regardless of whether or not the value is displayed as an integer or as a real number, the callback procedure receives the Slider value as a real number.

Whether or not the callback procedure is invoked while the slider is being “dragged” by the mouse, is controlled by the Slider Action argument. If “d” or “Drag” is specified, the callback procedure will be invoked every time the slider changes value while it is being dragged. If this argument is omitted, a callback will only be made when the mouse button is released.

The Minimum Slider Value and Maximum Slider Value arguments define the range of slider values, and the Initial Slider Value argument specifies the value to which the slider should be set when it is first displayed.

Sliders assigned to a graphics window are not displayed until the **graphwin_slidershow** command is executed.

definition

```
cmd graphwin_sliderassign(Slider_Id_Number, Value_Type, Slider_Action, Slider_Label,
Callback_Command, Minimum_Slider_Value, Maximum_Slider_Value, Initial_Slider_Value)
    int Slider_Id_Number {lolim = 1}
    int Value_Type {list = Value_Types, default(noquery) = r}
    int Slider_Action {list = Drag_Options, default(noquery) = n}
    char Slider_Label[] {lolim = 1, hilim = 12}
    char Callback_Command[] {lolim = 1, hilim = 12}
    real Minimum_Slider_Value {lolim = -1.0e+08, hilim = 1.0e+08}
    real Maximum_Slider_Value {lolim = -1.0e+08, hilim = 1.0e+08}
    real Initial_Slider_Value {lolim = -1.0e+08, hilim = 1.0e+08}
list Value_Types {
    "{r}Real" = 0,
    "{i}Int" = 1
}
list Drag_Options {
    "{n}No_Drag" = 0,
    "{d}Drag" = 1
}
```

graphwin_sliderassign example

In the following CCL code example, the command **fandraw** opens a graphics window, assigns a slider to select object point values, draws the current lens (which will be preserved by the call to **epset**) and draws an initial ray fan. Whenever the slider changes, the command **newfan** is called to redraw the ray fan for the new object point. (The call to **eperase** erases the previous fan drawing). For simplicity, the number of rays is fixed at 10.

```
cmd fandraw()
{
```

```

    set(graphics_autoclear, off);
    graphwin_open(1);
    graphwin_reset;
    GRAPHWIN_SLIDERASSIGN(r,d,1,"Frac. Y Obj.",newfan,0,1,0);
    graphwin_slidershow;
    draw_lens;
    epset;
    draw_rays(0,10,-1,1);
}

cmd newfan(int slider_id, double slider_value)
{
    /* This command is executed whenever the slider is moved.
    * The new fractional object height is given by slider_value.
    * The supplied slider_id is not used, since there is only
    * one slider.
    */
    graphwin_open(1);
    eperase;
    draw_rays(slider_value,10,-1,1);
}

```

gsd

graphwin_sliderdelete

The **graphwin_sliderdelete** command removes all sliders from the current slider window and deletes the window.

definition

```
cmd graphwin_sliderdelete(void)
```

graphwin_sliderdelete example

```
graphwin_sliderdelete;
```

gsr

graphwin_sliderreset(...)

The **graphwin_sliderreset** command permits the value type, range, and current value of an existing graphics window slider to be set.

This command would normally be used within a slider callback procedure.

For information about sliders and the meaning of the arguments used in the **graphwin_sliderreset** command, see **graphwin_sliderassign**.

definition

```
cmd graphwin_sliderreset(Slider_Id_Number, Value_Type, Minimum_Slider_Value,
Maximum_Slider_Value, Initial_Slider_Value)
    int Slider_Id_Number {lolim = 1}
    int Value_Type {list = Value_Types, default(noquery) = r}
    real Minimum_Slider_Value {lolim = -1.0e+08, hilim = 1.0e+08}
    real Maximum_Slider_Value {lolim = -1.0e+08, hilim = 1.0e+08}
    real Initial_Slider_Value {lolim = -1.0e+08, hilim = 1.0e+08}
```

graphwin_sliderreset example

In the example below, The Slider_Id_Number is assumed to have been given the name BENDING_ID by a previous #define statement.

```
graphwin_sliderreset(BENDING_ID,r,-1.5,1.5,0.2);
```

gss

graphwin_slidershow

The **graphwin_slidershow** command causes all sliders previously assigned to be displayed and made available for use.

Sliders defined by **graphwin_sliderassign** are not displayed immediately. When all sliders have been assigned, **graphwin_slidershow** will make them all visible and active at once.

definition

```
cmd graphwin_slidershow(void)
```

graphwin_slidershow example

The following CCL code fragment shows three sliders assigned to the current graphics window which are then made visible and active by the call to **graphwin_slidershow**. (#define statements are assumed to have been previously specified to assign names BENDING_ID, etc., to the Slider Id Numbers).

```
graphwin_sliderassign(r, d, BENDING_ID, "Bending", calc_bend,
-2.5, 2.5, 0);
graphwin_sliderassign(r, d, THICKNESS_ID, "Thickness",
calc_bend, 0, 5.0, 2.5);
graphwin_sliderassign(i, n, ELEMENT_ID, "Element", calc_bend,
1, 3, 1);
GRAPHWIN_SLIDERSHOW();
```

gsv

graphwin_slidervalue(...)

The **graphwin_slidervalue** is a CCL function which returns the current value of the specified slider assigned to the current vector graphics window. This function could be used within a slider callback procedure. When several sliders are attached to a graphics window, the response by a callback procedure to a change in one of the sliders requires a knowledge of the current values of the other sliders. These values might, for example be needed as arguments to various high level drawing commands executed within the callback procedure.

The slider whose value is to be returned by **graphwin_slidervalue** is identified by the Slider_Id_Number which was assigned to the slider by the

graphwin_sliderassign command. For more information about sliders see **graphwin_sliderassign**.

definition

```
real graphwin_slidervalue(Slider_Id_Number)
int Slider_Id_Number {lolim = 1}
```

graphwin_slidervalue example

In the following CCL code example, it is assumed that two sliders, identified by #defined Slider_Id_Numbers OBJ_SLIDER and PUP_SLIDER, have been assigned to graphics window 1. The sliders control parameters used for drawing a fan of rays through a lens which has been previously displayed and preserved by executing the **epset** command. OBJ_SLIDER selects an object point and PUP_SLIDER controls the width (in the pupil) of the fan of rays to be drawn. The callback command is called whenever either slider changes value.

```
cmd redraw_fan(int slider_id, double slider_value)
{
    real object, pupsize;
    if (slider_id == OBJ_SLIDER)
    {
        object = slider_value;
        pupsize = GRAPHWIN_SLIDERVALUE(PUP_SLIDER);
    }
    else
    {
        object = GRAPHWIN_SLIDERVALUE(OBJ_SLIDER);
        pupsize = slider_value;
    }
    graphwin_open(1);
    eperase;
    draw_rays(object,10,-pupsize,pupsize);
}
```

gwc

graphwin_close(...)

gwo

graphwin_open(...)

The **graphwin_open** command causes the specified vector graphics window to become visible on the screen (if it is not already visible) and selects it as the recipient of subsequent drawing commands. Windows are identified by number (1, ..., 32). If no window number is specified (or if 0 is specified), the next available graphics window is opened. Up to 32 graphics windows may be present on the screen simultaneously.

The **graphwin_close** command causes the specified graphics window to be removed from the screen.

The **graphwin_open** command allows an optional width and height (expressed as pixel counts) to be specified for the graphics window. If the specified window is already open, the window is resized to the specified size without losing its current contents. If no size is specified for the window (or 0 values are given) the window will be displayed at the size it had when it was last opened.

When a graphics window is closed, its current contents are destroyed and will not be present when the window is reopened.

A built-in global integer variable, *gfx_window*, is set to the window number of the currently open graphics window (or 0, if no window is open). If no Graphics

Window Number is specified for **graphwin_close** (or 0 is specified), the window identified by *gfx_window* will be closed.

definitions

```
cmd graphwin_open(Graphics_Window_Number, Window_Width, Window_Height)
cmd graphwin_close(Graphics_Window_Number)
    int Graphics_Window_Number
        {lolim = 0, hilim = 6, default(noquery) = 0}
    int Window_Width {lolim = 0, default(noquery) = 0}
    int Window_Height {lolim = 0, default(noquery) = 0}
```

graphwin_open/graphwin_close examples

```
graphwin_open;
graphwin_open 2;
graphwin_open gfx_window;
graphwin_open(1,640,400);
graphwin_close;
graphwin_close 3;
```

gwr

graphwin_reset

The **graphwin_reset** command deletes any sliders which may be attached to the current vector graphics window, erases the contents of the window, and resets the pen to pen 1.

This command is ordinarily used within CCL procedures and requires a vector graphics window to have been previously opened.

Graphwin_reset is normally called at the start of a new CCL graphics drawing procedure (immediately following a call to **graphwin_open**).

definition

```
cmd graphwin_reset(void)
```

graphwin_reset example

In the following CCL code example, the current graphics window (if any) is cleared of all prior contents and any sliders that may be present. The current lens and a fan of rays are then drawn in the window. (If no graphics window is currently open, the **graphwin_open** command will open window 1).

```
cmd drawlens()
{
    graphwin_open(gfx_window);
    GRAPHWIN_RESET;
    draw_lens;
    draw_rays(0,10,-1,1);
}
```

gwt

graphwin_title(...)

The **graphwin_title** command is used to display a title in the title area at the top of the current vector graphics window. The specified title is appended to the standard window title which, for vector graphics windows, is **GW#** (static window) or **UW#** (updatable window), where # is the window number.

definition

```
cmd graphwin_title(Window Title)
    char Window Title[] {default(noquery) = 0}
```

graphwin_title example

```
graphwin_title "Cooke Triplet";
```

tcp**textwin_copy**

The **textwin_copy** command copies the contents of the active text window to the clipboard in text format. If the **Output_page_mode (page)** preference is on, the text window contents are copied starting with the first visible line in the window; otherwise, the entire contents of the window are copied.

definition

```
cmd textwin_copy(void)
```

twc**textwin_close(...)****two****textwin_open(...)**

The **textwin_open** command causes the specified text output window to become visible on the screen (if it is not already visible) and selects it as the recipient of subsequent text output. Two text output windows are available and are identified by the numbers 1 and 2. If no window number is specified (or is 0 is specified), the window not currently in use is opened.

The **textwin_close** command causes the specified text window to be closed and removed from the screen (unless it is the only text output window visible on the screen). When a text output window is closed, its current contents are destroyed.

The **textwin_open** command allows the specification of an optional height and width for the window, expressed as Number of Lines and Number of Characters (per line). If the specified window is already open, the window is resized to the specified size without losing its current contents. If no size is specified for the window (or 0 values are given) the window will be displayed at the size it had when it was last opened.

A built-in global integer variable, *txt_window*, is set to the window number of the currently open text output window. If no Text Window Number is specified for **textwin_close** (or 0 is given), the window identified by *txt_window* will be closed. The **textwin_close** command is ignored if only one text window is present on the screen.

definitions

```
cmd textwin_open(Text_Window_Number, Number_of_Lines, Number_of_Characters)
cmd textwin_close(Text_Window_Number)
    int Text_Window_Number
        {lolim = 0, hilim = 2, default(noquery) = 0}
    int Number of Lines {lolim = 0, default(noquery) = 0}
    int Number of Characters {lolim = 0, default(noquery) = 0}
```

textwin_open/textwin_close example

```
textwin_open;
textwin_open 2;
textwin_open 1,20;
textwin_close;
textwin_close 2;
```

twr

textwin_reset

The **textwin_reset** command deletes the contents of the current text output window and clears the text window storage buffer (the buffer used to store real numbers written to the window).

definition

```
cmd textwin_reset(void)
```

textwin_reset example

```
textwin_reset;
```

tw

textwin_title(...)

The **textwin_title** command is used to display a title in the title area at the top of the current text output window. The specified title is appended to the standard window title which, for text output windows, is "Text Window #," where # is the window number.

definition

```
cmd textwin_title(Window Title)
    char Window Title[] {default(noquery) = 0}
```

textwin_title example

```
textwin_title "Data for Lens A"
```

File & I/O routines

fclose(...)

The **fclose** function closes the file specified by `file_id` (which must identify a file previously opened by the **fopen** command). When a file is closed it is no longer accessible until it is reopened.

If an error occurs (e.g., `file_id` does not identify a currently open file) **fclose** returns a negative value which is the id number of a built-in error message; otherwise, **fclose** returns zero.

definition

```
int fclose(int file_id)
```

fopen(...)

The **fopen** function opens the named file for use in the manner specified by the filemode argument. If the file is opened successfully, **fopen** returns a positive file identification number (file_id). Calls to other CCL file functions which access the file will include the file_id as an argument. CCL allows a maximum of three files to be open simultaneously.

The filemode argument may be one of the following:

Argument	Description
r	open a file for reading - the file must exist
w	create a file for writing - if the file exists, its previous contents will be discarded
a	open a file for writing (appending) at its end - if the file does not exist, it is created
r+	open a file for update (reading and writing)
w+	create a file for update (reading and writing)
a+	open a file for reading and writing (appending) at its end

If a file is opened for update (reading and writing), a write operation may not be directly followed by a read operation without an intervening call to the **fseek** function. Similarly, a read operation may not be directly followed by a write operation without an intervening call to **fseek**.

If the specified file cannot be opened, **fopen** returns a negative value which is the id number of a built-in error message. The error message may be displayed by using this value as the argument in a call to the **message** command.

If the filename does not specify a directory, the directory specified by the OSDATA environment variable is assumed.

It is important to note that an opened file remains open until it is explicitly closed by a call to the **fclose** function. Files are not automatically closed at the end of a CCL procedure. Any open file will be closed, however, when the OSLO program is terminated.

Although the CCL file functions are mainly intended to be used in compiled CCL procedures, the fact that a file remains open until explicitly closed makes it is possible to execute file functions directly from the OSLO command line.

definition

```
int fopen(char filename[], char filemode[])
```

fopen example

For example, the command line sequence:

```
j = fopen(testfile, r)
fseek(j, 0, 2)
ftell(j)
```

will display size (in bytes) of “testfile” in the message area next to the command line.

fprintf(...)

This function operates in the same manner as the **printf** command except that the generated character string is written to the current position in the file identified by `file_id` instead of being printed. `File_id` must be the value returned by a prior call to the CCL **fopen** function.

The “format” argument is either an integer id number identifying a format string stored in the CCL `a_string.ccl` file, or is the format string itself.

The variables are any integers, real numbers or expressions which are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string. If an integer variable is specified for string (%s) conversion, it is taken to be the id number of a string in the CCL `a_string.ccl` file.

Unless an error occurs, **fprintf** returns a count of the number of bytes written to the file. If an error occurs, **fprintf** returns a negative value which is the id number of a built-in error message. This may be used, if desired, in a **message** or **printf** command.

definition

```
int fprintf(int file_id, format, [variables])
```

fread(...)

The **fread** function may be used to read one or more characters, integers or real numbers from the current position in the file identified by `file_id`.

`File_id` identifies the file and is the value returned by a prior call to the CCL **fopen** function.

The “object_address” argument must be the name of a CCL array or the name of a CCL scalar variable preceded by the CCL “pass-by-reference” operator, &. The item (or items) read will be placed in the specified variable or array. The data type of the object_address variable or array must correspond to the data type of the item (or items) to be read.

The `item_count` argument is optional, and specifies a count of the number of items to be read.

If the `item_count` is omitted, as many items are read as will fit in the variable or array specified by the `object_address`. If the `object_address` identifies a scalar variable, a single item will be read.

If the `object_address` is the name of an array and an `item_count` is specified, only the number of items specified will be read into the array. If the array is 2-dimensional, items will fill the array a row at a time. The `item_count` must not exceed the size of the array.

If an error occurs, **fread** returns a negative value which is the id number of a built-in error message. Otherwise, **fread** returns a count of the number of items read.

Except for the use of "&" to indicate the address of a scalar variable into which a single item is to be read, **fread** is identical in form to the CCL **fwrite** function.

definition

```
int fread(int file_id, object_address, <int item_count>)
```

fread examples

1. Write a real number from "v1" and reread it into "v2":

```
int file_id;
real v1 = sqrt(2.5);
real v2;
file_id = fopen(testfile.tmp, w+);
fwrite(file_id, v1);
fseek(file_id, 0, 0);
fread(file_id, &v2);
```

2. Simulation of the C-language **fgetc** function to read a character and return its value as an integer (or -1 if error or end-of-file):

```
int fgetc(int file_id)
{
    char ch;
    if (fread(file_id, &ch) != 1) return -1;
    else return ch;
}
```

fscanf(...)

This function is the complement of the **fprintf** function. It reads a character string from the current position in the file identified by `file_id`, according to the format statement, and converts it into binary data that is saved in the variables. `File_id` must be the value returned by a prior call to the CCL **fopen** function.

The "format" argument is either an integer id number identifying a format string stored in the CCL `a_string.ccl` file, or is the format string itself. The format argument must exactly match the one used to write data to the file, including spaces and ignored characters.

The variables are any characters, integers, real numbers or strings that are needed to satisfy conversion specifications (%c, %d, %f, etc.) appearing in the format string. The variables must be passed by reference using the pass by reference operator defined in CCL.

Unless an error occurs, **fscanf** returns a count of the number of items successfully read and saved in the variables. If an error occurs, **fscanf** returns a

negative value, which is the id number of a built-in error message. This may be used, if desired, in a **message** or **printf** command.

definition

```
int fscanf(int file_id, format, [variables])
```

fscanf example

```
fscanf(1, "%.15e%.15e", &xcoord, &ycoord);
```

fseek(...)

The **fseek** function is used to set the byte position in a file. A subsequent read or write operation will start from this position. The byte position to set is defined by its offset from a specified origin.

File_id identifies the file and is the value returned by a prior call to the CL **fopen** function.

Seek_offset is a count of the number of bytes separating the desired position from the origin defined by seek_origin.

The seek_origin argument may be one of the following:

Argument	Description
0	origin is at the beginning of the file
1	origin is the current position in the file
2	origin is at the end of the file

If seek_origin is omitted, it defaults to 0.

If an error occurs (e.g., file_id does not identify a currently open file or the file position does not exist) **fseek** returns a negative value which is the id number of a built-in error message, otherwise, **fseek** returns zero.

Byte positions in a file are numbered 0, 1, 2, ... , EOF, where EOF denotes end-of-file. If a file contains 100 bytes, EOF = 100 and the position of the last byte in the file is 99.

The “current” position in a file is the position from which the next read operation will get its first byte or the next write operation will put its first byte. Whenever a read or write operation is executed, the current position will be advanced by the number of bytes read or written. The **fseek** function may be used to reset the current position to any point in the file prior to reading or writing.

The most common uses of **fseek** are to go to the beginning or end of a file or to return to a position determined by a previous call to the **ftell** function.

definition

```
int fseek(int file_id, int seek_offset, int_seek_origin)
```

fseek examples

```
fseek(file_id,0,0)  /* sets the current position to beginning of the file. */
fseek(file_id,0,2)  /* sets the current position to the end-of-file (EOF) position. This
is the position immediately following the last byte written to the file. */
fseek(file_id,-2,2) /* set the current position to the next to last byte written to the
file (2 less than the end-of-file position). */
```

ftell(...)

The **ftell** function returns the current byte position in the file identified by `file_id`.

`File_id` identifies the file and is the value returned by a prior call to the CCL **fopen** function.

If an error occurs (e.g., `file_id` does not identify a currently open file) **fseek** returns a negative value which is the id number of a built-in error message.

definition

```
int ftell(int file_id)
```

fwrite(...)

The **fwrite** function may be used to write any CCL variable, constant, or array to the current position of the file identified by `file_id`.

`File_id` identifies the file and is the value returned by a prior call to the CCL **fopen** function.

The “object” argument may be a constant, a literal string, or the name of any CCL scalar variable or array regardless of data type.

The `item_count` argument is optional, and specifies a count of the number of items to be written.

If the `item_count` argument is omitted, the number of items written is determined by the kind of object. If it is an array, all of the elements of the array are written. If the array is 2-dimensional, the array is written in row order - i.e., the elements of the first row are written first, followed by the elements of the second row, and so on. A literal string of N characters (enclosed in quotes) is treated as a character array of $N+1$ elements (the characters of the string followed by a “null” termination character). If the object is a constant or a scalar variable, only a single value is written.

If `item_count` is specified it may not exceed the number of elements contained in the “object.” An `item_count` is superfluous if the object is a constant or scalar variable.

If an error occurs, **fwrite** returns a negative value which is the id number of a built-in error message. Otherwise, **fwrite** returns a count of the number of items written.

Note that integers and real numbers are not converted to character strings when they are written to the file. They are machine representations of the numbers. In

all current versions of OSLO, an integer is written to the file as a string of 4 bytes and a real (floating point) number is written as a string of 8 bytes.

It is important to note that integers and real numbers written to a file on one kind of computer may not be read correctly on another kind of computer. The reason for this is that different machine architectures may order the bytes in integers and real numbers differently.

definition

```
int fwrite(int file_id, object <,int item_count>)
```

gwp

graphwin_print (...)

This command prints the current graphics window. If printer_or_file is "Printer," output is sent to the printer; otherwise, it is sent to a graphics file. If show_print_dialog is "Yes," the standard Print dialog box (for printer output) or File dialog box (for file output) is displayed. If show_print_dialog is "No," the window is printed to the printer or to the graphics file whose name is given by plot_file_name.

definition

```
cmd graphwin_print(int Printer_or_file, int Show_print_dialog,
                  char Plot_file_name)
```

hpw

hpgl_write(...)

The **hpgl_write** command may be used to write a plotter file in HP-GL format representing the contents of the current vector graphics window. The file name is specified explicitly and no special filetype or directory controls are imposed on it.

definition

```
cmd hpgl_write(Pathname)
      char Pathname[] {lolim = 1}
```

hpgl_write example

```
hpgl_write "/myhome/testfile"
```

inp

input(...)

The **input** command that can be used inside a CCL or SCP procedure to get data from the keyboard .

Object_address is either a scalar variable passed by reference or a character array (e.g., &a, &k, Astr). Numeric arrays are not allowed (see example below).

Format is either a literal format string or a message file number for a format string.

Format_parameters are any variables or expressions needed to satisfy %d, %f, %s, etc., in the format string.

Default_value is an optional string to be placed in the command line when the input command is executed.

The format (together with the format_parameters) is used to create the command line prompt.

The user's response is returned in the object_address.

If the user hits CANCEL or ESCAPE, the procedure terminates.

The type of response expected from the user is determined by the data type of the object_address. If the object_address is the name of a character string array, the dimension of the passed array is used as the "hilim" character count for the user's response.

definition

```
cmd input(object_address, format, format_parameters, default_value);
```

input example

The following SCP command illustrates the use of **input**.

```
*prompt
input(astr, "Enter test string:");
printf("\nTest string = \"%s\"\n", astr);
for (j = 0; j < 3; j++)
{
input(&x, "Enter xa[%d]:", j, "0.0");
    xa[j] = x;
}
printf("xa[0] = %f, xa[1] = %f, xa[2] = %f\n", xa[0], xa[1], xa[2]);
```

msg

message(...)

The **message** command causes a message to be printed next to the command line in the OSLO main window. This command is intended to be used in CCL command procedures. The **message** command has the same kind of variable argument list as is used for the **printf** command. The first argument, the format string, may either be a literal string or the id number of a string in the a_string.ccl file. The displayed message may contain up to two lines.

definition

```
cmd message(format, [variables])
```

The "format" argument is either an integer id number identifying a format string stored in the CCL a_string.ccl file, or is the format string itself. The variables are any integers, real numbers or expressions which are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string.

message example

```
message ("The calculation failed after %d iterations.\nSuggest you try adding an element.",
iteration_count);
```

prf**printf(...)****aprf****aprintf(...)**

The **printf** command duplicates the function of the C **printf** command. **Printf** permits wide flexibility in format and allows control of precision with which real numbers are printed. The printing format is determined by the first argument, which is called the “format string.” The **aprintf** command is similar to the **printf** command, except that floating-point output is placed in the spreadsheet buffer.

Output generated by **printf** commands continue to go to the same line until a newline character (**\n**) appears in the format string. A line is not written to the text output window until a newline character is encountered.

The first argument in the **printf** command, the format string, may be either a literal string or the id number of a string in the **a_string.ccl** file.

The format string contains a mixture of characters to be printed and format specifications. Whenever a format specification is encountered, it is used to format the next unprocessed argument in the list of arguments following the format string. The formatted argument replaces the format specification in the printed output.

Format specifications indicate how an item is to be printed in the “field” which replaces the format specification in the output line. The field is just a group of character positions which is at least as long as (and can be longer than) the number of characters required to print the item.

Format specifications begin with a percent sign (%) and end with a conversion character (c, d, e, f, g, i, s, x) which indicates the type of conversion to be performed. The list of allowed conversions is as follows:

Char	Conversion
c	single character
d, i	integer (base 10)
e	real number in exponent notation
f	real number in decimal notation
g	real number in (a) exponent notation if the exponent is less than -4 or greater than the precision, or (b) decimal number notation with no trailing 0's otherwise
s	character string
x	unsigned hexadecimal integer (base 16)

Between the % sign and the conversion character, there may be, in order:

- a minus sign, which indicates left justification in the field.

- a number indicating the minimum field width. If necessary, spaces will be added to fill out the field.
- a period, which separates the field width from the precision.
- a number, the precision, which specifies
- the maximum number of characters to be printed from a string,
- the number of digits after the decimal point of a real number, or
- the minimum number of digits for an integer.

The default precision for e, f, and g conversions is 6 digits following the decimal point.

The default field width is the minimum number of character positions necessary to print the specified item.

A literal percent sign is printed whenever “%%” is encountered in the format string.

A new line is started whenever the newline character, “\n,” is encountered in the format string.

It is possible to use “\” followed by an integer space count within the format string. Thus “\9” produces 9 spaces and “\16” produces 16 spaces.

definition

```
cmd printf(format,[variables])
```

The “format” argument is either an integer id number identifying a format string stored in the CCL a_string.ccl file, or is the format string itself. The variables are any integers, real numbers or expressions that are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string. If an integer variable is specified for string (%s) conversion, it is taken to be the id number of a string in the CCL a_string.ccl file.

printf example

```
printf "The square root of x = %e\n" sqrt(x);
printf ("x = %.9f, y = %.9f, z = %.9f\n", x, y, z);
```

prt

print(...)

aprt

aprint(...)

The **print** command prints a list of character string, real number and integer arguments in the current text output window. The number of arguments may range from 0 to 24. Each argument is formatted according to its type and separated from the previous argument by a single space. This command is usually used in CCL command procedures, but is also useful on the command line for adding comments and annotations to the text output. Integer and real numbers are formatted in the same way as they are in normal OSLO output to the text window. The **aprint** command is similar to the **print** command, except that floating-point output is placed in the spreadsheet buffer.

Spaces may be added between items on a line by enclosing the spaces in a quoted string - e.g., “ ”. To create more than 3 spaces, it is possible to use “\” followed by an integer space count within a quoted string. Thus “\9” produces 9

spaces and “\\16” produces 16 spaces. A **print** command with no arguments simply prints a blank line.

definition

```
cmd print([variable argument list])
```

The arguments may be character strings (enclosed in quotes) or any integer or real numbers or expressions.

print example

```
print "SRF      CV      TH"
for (j = 0; j < imsnbr; j++) print j cv[j] th[j];
```

twp

textwin_print(...)

This command prints the current text output window. If `printer_or_file` is “Printer,” output is sent to the printer; otherwise, it is sent to a text file. If `Show_print_dialog` is “Yes,” the standard Print dialog box (for printer output) or File dialog box (for file output) is displayed. If `Show_print_dialog` is “No,” the window is printed to the printer or to the text file whose name is given by `print_file_name`.

definition

```
cmd textwin_print(int Printer_or_file, int Show_print_dialog,
char Print_file_name)
```

Math library

int coldim(...)

int rowdim(...)

rowdim returns the largest row dimension found in the items listed in the variable list while **coldim** returns the largest column dimension found among the listed items.

Ordinarily, the variable list will consist of a single array name. The variable list may contain items of any data type- i.e., real, integer, or character.

For single dimensional arrays, **rowdim** will always return 1 and **coldim** returns the dimension of the array. For two-dimensional arrays, **rowdim** returns the number of rows in the array and **coldim** returns the number of columns.

definitions

```
int rowdim([variable list])
int coldim([variable list])
```

coldim example

```
real vector_magnitude(real vector[])
{
    int j, dimen = coldim(vector);
    real sum = 0;
    for (j = 0; j < dimen; j++) sum += vector[j]**2;
```

```

    return (sqrt(sum));
}

```

fft(...)

fftnv(...)

These commands compute a Fast Fourier Transform and an Inverse Fast Fourier Transform, respectively. The transforms may be 1 or 2 dimensional. The input[] array and output[] array consist of complex numbers with real and imaginary parts interleaved - i.e., real, imag, real, imag, etc. The output[] array may be the same array as the input[] array (in which case the output replaces the input).

The “rows” and “cols” arguments give, respectively, the row dimension and column dimension of the problem. They must both be powers of 2. A 1 dimensional transform is obtained when rows = 1. Data in the input array is ordered by rows and the total length (dimension) of the input[] and output[] arrays is 2*nrows*ncols.

Use of these commands requires a detailed understanding of what Fast Fourier Transforms mean and how their data arrays are arranged. The implementation of these commands in OSLO is derived from material which may be found in *Numerical Recipes in C* by William H. Press, et al., Cambridge University Press. Please refer to this source for further information.

definitions

```

cmd fft(real input[], real output[], int rows, int cols)
cmd fftnv(real input[], real output[], int rows, int cols)

```

fft example

The following CCL code example calculates the 1 dimensional diffraction based OTF for a rectangular aperture with no aberration:

```

cmd test_fft()
{
    int j, k;
    real v[256];
    /* Set up real square wave as input: */
    for (j = 0; j < 256; j++) v[j] = 0.0;
    for (j = 64; j < 192; j += 2) v[j] = 1.0;
    /* 1st transform - diffraction integral: */
    FFTINV(v,v,1,128);
    /* Multiply transform by its complex conjugate to get */
    /* image intensity distribution: */
    for (j = 0; j < 256; j += 2)
    {
        v[j] = (v[j]*v[j] + v[j+1]*v[j+1]);
        v[j+1] = 0.0;
    }
    /* Transform result to get OTF: */
    FFT(v,v,1,128);
    print "\nRelative Frequency, Re OTF, Im OTF):\n";
    for (j = 0; j <= 128; j += 2)
    {
        k = j/2;
        printf "%3d %10.6f %10.6f\n" k, 2*v[j] v[j+1];
    }
}

```

```

    }
}

```

mmprod(...)

This command multiplies the supplied matrix2[] by the supplied matrix1[] (i.e., matrix1 \times matrix2) and returns the product in the product[] array.

This procedure is intended to be called only from within a CCL procedure. If the dimensions of the supplied arrays are inconsistent with the operation to be performed, an error message will be generated.

definition

```
cmd mmprod(real matrix1[[]], real matrix2[[]], real product[[]])
```

mmprod example

```
cmd rot_matrix(real alpha, real beta, real c[[]])
{
    /* Return "rotation" matrix c[][] representing successive rotations through angles
    alpha and beta: */
    real a[2][2], b[2][2];
    if (rowdim(c) != 2 || coldim(c) != 2)
        abort "Rotation matrix must be 2x2";
    a[0][0] = a[1][1] = cos(alpha);
    a[0][1] = sin(alpha);
    a[1][0] = -a[0][1];
    b[0][0] = b[1][1] = cos(beta);
    b[0][1] = sin(beta);
    b[1][0] = -b[0][1];
    MMPROD(a, b, c);
}

```

mmtprod(...)

This command multiplies the transpose of the supplied matrix[] by the matrix (i.e., matrix \times matrix-transpose) and returns the product in the product[] array.

This procedure is intended to be called only from within a CCL procedure. If the dimensions of the supplied arrays are inconsistent with the operation to be performed, an error message will be generated.

definition

```
cmd mmtprod(real matrix[[]], real product[[]])
```

mmtprod example

```
cmd undercon_eqns(real ma[[]], real vx[], real vy[])
{
    /* Solve an underconstrained set of eqns, ma * vx = vy, and return the result in
    vx[] */
    int nr = rowdim(ma);
    real mmt[nr][nr], vz[nr];
    MMTPROD(ma, mmt);
    slveqs(mmt, vz, vy);
    mtvprod(ma, vz, vx);
}

```

mtmprod(...)

This command multiplies the supplied matrix`[][]` by its transpose (i.e., matrix-transpose \times matrix) and returns the product in the product`[][]` array.

This procedure is intended to be called only from within a CCL procedure. If the dimensions of the supplied arrays are inconsistent with the operation to be performed, an error message will be generated.

definition

```
cmd mtmprod(real matrix[][], real product[][])
```

mtmprod example

```
cmd least_squares(real ma[][], real vx[], real vy[])
{
    /* Solve overconstrained set of eqns, ma * vx = vy, by least squares and return
result in vx[] */
    int nc = coldim(ma);
    real mtm[nc][nc], mty[nc];
    MTMPROD(ma, mtm);
    mtvprod(ma, vy, mty);
    slveqs(mtm, vx, mty);
}
```

mtvprod(...)

This command multiplies the supplied vector`[]` by the transpose of the supplied matrix`[][]` (i.e., matrix-transpose \times vector) and returns the product in the product`[]` array.

This procedure is intended to be called only from within a CCL procedure. If the dimensions of the supplied arrays are inconsistent with the operation to be performed, an error message will be generated.

definition

```
cmd mtvprod(real matrix[][], real vector[], real product[])
```

mtvprod example

```
cmd least_squares(real ma[][], real vx[], real vy[])
{
    /* Solve overconstrained set of eqns, ma * vx = vy, by least squares and return
result in vx[] */
    int nc = coldim(ma);
    real mtm[nc][nc], mty[nc];
    mtmprod(ma, mtm);
    MTVPROD(ma, vy, mty);
    slveqs(mtm, vx, mty);
}
```

mvprod(...)

This command multiplies the supplied vector[] by the supplied matrix[][] (i.e., matrix \times vector) and returns the product in the product[] array.

This procedure is intended to be called only from within a CCL procedure. If the dimensions of the supplied arrays are inconsistent with the operation to be performed, an error message will be generated.

definition

```
cmd mvprod(real matrix[][], real vector[], real product[])
```

mvprod example

```
cmd eqn_check()
{
    /* Note: built-in globals i and j are used for indices */
    real vx[10], vy[10], vc[10], ma[10][10];
    for (i = 0; i < 10; i++) {
        vy[i] = my_vector_calc(i);
        for (j = 0; j < 10; j++)
            ma[i][j] = my_matrix_calc(i,j);
    }
    slveqs(ma, vx, vy); /* Solve ma * vx = vy (for vx) */
    MVPROD(ma, vx, vc); /* Calculate vc = ma * vx */
    max_error = 0;
    for (i = 0; i < 10; i++) {
        if ((error = fabs(vy[i] - vc[i])) > max_error)
            max_error = error;
    }
    print "Max error =" max_error;
}
```

real polint(...)

xp[], yp[], and np define a polynomial, $y = f(x)$, passing through the points xp[j], yp[j], $j = 0, 1, \dots, np - 1$. For the specified value of x, **polint** returns the interpolated polynomial value $y = f(x)$.

This function is only intended for use within CCL procedures.

definition

```
real polint(real xp[], real yp[], int np, real x)
```

polint example

```
cmd polint_test(real x)
{
    static real xp[3] = {0, 1, 2};
    static real yp[3] = {0.1, 0.4, 1.5};
    printf("For x = %f, y = %f\n", x, polint(xp,yp,3,x));
}
```

solveqs(...)

The command solves the set of equations $a[i][j] \times x[j] = y[i]$ for the vector of unknowns, $x[i]$. This command is intended to be called only from within a CCL procedure. If the dimensions of the supplied arrays are inconsistent with the operation to be performed, an error message will be generated.

If the supplied arrays represent a square set of equations, they are solved using LU decomposition. Otherwise they are solved using SV decomposition (Singular Value Decomposition). For a description of these two equation solving methods, see *Numerical Recipes in C* by William H. Press, et al., Cambridge University Press.

definition

```
cmd solveqs (real a[][], real x[], real y[])
```

solveqs example

```
cmd solve_my_eqns()
{
    /* Note: built-in globals i and j are used for indices */
    real vx[10], vy[10], ma[10][10];
    for (i = 0; i < 10; i++){
        vy[i] = my_vector_calc(i);
        for (j = 0; j < 10; j++){
            ma[i][j] = my_matrix_calc(i,j);
        }
    }
    SLVEQS(ma, vx, vy); /* Solve ma * vx = vy (for vx) */
    for (i = 0; i < 10; i++){
        printf("x[%d] = %g.9\n", i, x[i]);
    }
}
```

sort(...)

The **sort** command fills the supplied `index[]` array with `object_array[]` indices which may be used to extract values from the object array in ascending order of value. The elements of `object_array[]` are not moved.

`object_array[]` may be any one-dimensional array (int, real or char) or a two-dimensional char array. For a two-dimensional char array, the rows are treated as character strings and are sorted using **strncmp** (limited by the column dimension if there is no '\0' terminator in a row). In this case, `index[]` selects rows of the array.

`Nbr_of_items` is the count of items in the `object_array` to be sorted. If `nbr_of_items` is not specified, the dimension of `object_array[]` defines `nbr_of_items`.

The `index[]` array must have a dimension greater than or equal to `nbr_of_items`.

definition

```
static void sort (index[], object_array[], nbr_of_items)
```

sort example

Assume real $x[j]$, real $y[j]$, $j = 0, \dots, n$, are the coordinates of n points.

(a) Print out the points in ascending order of x :

```
sort(index, x, n);
for (j = 0; j < n; j++)
printf("x = %12.3f, y = %12.3f\n", x[index[j]], y[index[j]]);
```

(b) Now print out the points in ascending order of y :

```
sort(index, y, n);
for (j = 0; j < n; j++)
printf("y = %12.3f, x = %12.3f\n", y[index[j]], x[index[j]]);
```

Mathematical and trigonometric functions

In addition to the commands documented above, OSLO contains the following mathematical and trigonometric functions.

Function	Description
acos (argument)	Returns the arccosine of <i>argument</i> , in radians.
asin (argument)	Returns the arcsine of <i>argument</i> , in radians.
atan2 (y_argument, x_argument)	Returns the arctangent of <i>y_argument</i> / <i>x_argument</i> , in radians.
ceil (argument)	Returns the smallest integer not less than <i>argument</i> , expressed as a real number.
cos (argument)	Returns the cosine of <i>argument</i> , which is in radians.
exp (argument)	Returns e (2.71828...) raised to the power of <i>argument</i> .
fabs (argument)	Returns the absolute value of <i>argument</i> .
floor (argument)	Returns the largest integer not greater than <i>argument</i> , expressed as a real number.
grand (seed)	Returns a real pseudo-random number taken from a Gaussian (normal) distribution with zero mean and unit variance. The <i>seed</i> argument is used as in the rand function.
j0 (argument)	Returns the Bessel function J_0 of <i>argument</i> .
j1 (argument)	Returns the Bessel function J_1 of <i>argument</i> .
log (argument)	Returns the natural logarithm of <i>argument</i> .
log10 (argument)	Returns the base-10 logarithm of <i>argument</i> .

lrand (seed)	Returns a real pseudo-random number taken from a “Lambertian” distribution: the probability density function is $p(x) = (\pi/2)\cos(\pi x/2)$ for $0 \leq x \leq 1$ and $p(x) = 0$ otherwise. The <i>seed</i> argument is used as in the rand function.
pow (argument, exponent)	Returns <i>argument</i> raised to the power <i>exponent</i> .
rand (seed)	Returns a real pseudo-random number between 0.0 and 1.0, taken from a uniform distribution. If <i>seed</i> (an integer) is positive, it is used to initialize the sequence of pseudo-random numbers (a given <i>seed</i> will produce an identical sequence of random numbers). If <i>seed</i> is negative, the system clock is used to initialize the sequence. If <i>seed</i> is zero, the next number in the sequence is returned.
rint (argument)	Rounds <i>argument</i> to the nearest integer and returns the result as a real number.
round (argument, precision)	Returns <i>argument</i> rounded to <i>precision</i> significant digits.
sin (argument)	Returns the sine of <i>argument</i> , which is in radians.
sqrt (argument)	Returns the square root of <i>argument</i> .
tan (argument)	Returns the tangent of <i>argument</i> , which is in radians.

String functions

real atof(...)

atof(string_name) returns the real value for the string whose array name is string_name. The argument may be a literal string, although this would be unusual since its value would already be known.

definition

```
real atof(char string[])
```

atof example

```
value = atof(realstr);
```

long atol(...)

atol(string_name) returns the integer value for the string whose array name is string_name. The argument may be a literal string, although this would be unusual since its value would already be known.

definition

```
long atol(char string[])
```

atol example

```
daynbr = atol(daystr);
```

int sprintf(...)

This function operates in the same manner as the **printf** command except that the generated character string is copied into the supplied buffer instead of being printed.

The “format” argument is either an integer id number identifying a format string stored in the CCL a_string.ccl file, or is the format string itself.

The variables are any integers, real numbers or expressions that are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string. If an integer variable is specified for string (%s) conversion, it is taken to be the id number of a string in the CCL a_string.ccl file.

Sprintf returns 0 if the copy is successful. If the buffer is not large enough to hold the formatted string, **sprintf** returns the id number of a built-in error message which may be used, if desired, in a **message** or **printf** command. The **sprintf** function provides the principal means of copying a string from the a_string.ccl file into a CCL character array. Note that if no variables are supplied, any conversion specifications appearing in the format string will be copied into the buffer as literal text.

definition

```
int sprintf(char buffer[], format, [variables])
```

sprintf example

```
if (!sprintf(filename, "%s.%s", 132, 143)) file_id = fopen(filename, "r");
```

int strcat(...)

This function concatenates the supplied string to the end of any string already in the supplied buffer and returns 0 if successful. If the buffer is not long enough to contain the resultant string, it is left untouched and **strcat** returns the id number of a built-in error message which may be used, if desired, in a **message** or **printf** command.

The string argument may be a literal string or the name of a character string array.
If `buffer[0] = 0`, this function is equivalent to **strcpy**.

definition

```
int strcat(char buffer[], char string[])
```

strcat example

```
if (error = strcat(mybuffer, filename)) message(error);
```

int strcmp(...)

int stricmp(...)

strcmp(string1, string2) compares string1 to string2, character by character and returns a negative value if string1 is lexicographically less than string2, a positive value if string1 is lexicographically greater than string 2, and zero if the strings are identical. **Stricmp** is a case-insensitive version of the **strcmp** function.

definitions

```
int strcmp(char string1[], char string2[])
int stricmp(char string1[], char string2[])
```

strcmp example

```
if (strcmp(string1,string2)) print "Strings do not match";
```

int strcpy(...)

This function copies a string into a string buffer and returns 0 if the copy is successful. If the buffer is not long enough to hold the string, it is left untouched and **strcpy** returns the id number of a built-in error message which may be used, if desired, in a **message** or **printf** command.

The string argument may be a literal string or the name of a character string array.

definition

```
int strcpy(char buffer[], char string[])
```

strcpy example

```
if (error = strcpy(mybuffer, filename)) message(error);
```

int strlen(...)

strlen(string_name) returns the length, i.e., the character count, for the string whose array name is string_name.

The argument may be a literal string, although this would be unusual since its length would already be known.

Ordinarily this function would be called to determine the length of a string passed as an argument to a procedure.

definition

```
int strlen(char string[])
```

strlen example

```
int check_for_dot(char filename[])
{
    int j;
    for (j = 0; j < strlen(filename); j++)
        if (filename[j] == '.') return 1;
    return 0;
}
```

Graphics library

contour(...)

Contour lines for a two-dimensional array of data can be drawn by using the CCL **contour** command

The first argument, *Contour_level_type*, is given as either “equ” if equally incremented contour levels are to be drawn, or “spc” if specified levels are to be drawn.

The second argument, *Contour_line_style*, is either “sol” if solid lines are to be drawn for the contour levels, or “sty” if line styles are to be used. The program will cycle through the available line styles when drawing the contour lines. The default is to use solid lines.

The third argument, *Data_values*, is either “unc” if the data is to be used in an unconverted form, or “db” if the data is to be plotted in decibels. (OSLO will convert the supplied data to decibels; no transformation needs to be done by the user.)

The fourth argument, *Contour_data*, is a two-dimensional array of real numbers, that contain the data values to be contoured. The data is arranged by analogy to a function $f(x, y)$ in that the first array argument corresponds to the x-direction and the second argument to the y-direction. Thus, *Contour_data*[0][0] is the lower left corner, *Contour_data*[*x_max*][0] is the lower right corner, *Contour_data*[0][*y_max*] is the upper left corner, and *Contour_data*[*x_max*][*y_max*] is the upper right corner. Note that *x_max* and *y_max* may be different. OSLO will assume that all elements of *Contour_data*[][] have values. If the valid data area does not correspond to the entire rectangular area represented by *Contour_data*[], place a value greater than 1.0×10^{20} in those array elements outside the region of interest. OSLO will interpret this as defining the region of data to be contoured.

The fifth argument, *Number_of_contour_levels*, is the number of contour levels to be drawn.

The sixth and seventh arguments, *Minimum_contour_level* and *Maximum_contour_level*, are the minimum and maximum values for equally spaced contours, and only need to be supplied if “equ” was chosen for the *Contour_level_type*. If both minimum and maximum are supplied as 0.0, OSLO will use the minimum and maximum values of the data in *Contour_data*[][].

The eighth argument, *Contour_line_levels*, is a one-dimensional array of real numbers (of size at least equal to *Number_of_contour_levels*) containing the

desired contour levels to be drawn. This argument only needs to be supplied if “spc” was chosen for the `Contour_level_type`.

The ninth argument, `Level_label`, is the integer number for a labeling string defined in `a_string.ccl`. This label is used above the “legend” of contour levels drawn to the left of the plot. If `Level_label` is 0, no label is printed. If the ninth argument, `Level_label` is -1, the contour level legend in the contour plot is suppressed.

definition

```
static void Contour (Contour_level_type, Contour_line_style, Data_values,
    Contour_data[][[]], Number_of_contour_levels, Minimum_contour_level,
    Maximum_contour_level, Contour_line_levels[], Level_label)
```

gcl

gclear(...)

The **gclear** command erases the current vector graphics window and resets the pen to pen 1. The **gclear** command differs from the **graphwin_reset** command in that any sliders attached to the window are retained. **Gclear** is often used to clear the window prior to redrawing its entire contents in response to a slider action. It might also be used in connection with drawing an animated sequence of images.

definition

```
cmd gclear(void)
```

gclear example

```
cmd expanding_box()
{
    real size;

    graphwin_open(1);
    for (size = .02; size < .5; size += 0.02)
    {
        /* Clear the window, draw the frame and refresh
         * the display:
         */
        GCLEAR;
        viewport(.5-size, .5+size, .5-size, .5+size);
        frame;
        gshow;
        /* Let the frame remain visible for 1/10 second: */
        time_reset;
        while (time < 100) ;
    }
}
```

ges

epset

gee

eperase

The **epset** command in conjunction with the **eperase** command allows a fixed background of labels, axes, tickmarks and other objects to be drawn once and preserved while a variable part of the image is redrawn on command (usually in response to a change in position of a slider attached to the window).

Execution of the **epset** command indicates that the present contents of the selected vector graphics window is to be preserved when the **eperase** command is subsequently executed.

Execution of the **eperase** command erases everything that has been written to the selected graphics window since the **epset** command was last executed.

If **epset** has not been executed since the graphics window was last reset (by the **graphwin_reset** command), **eperase** has no effect.

definitions

```
cmd epset(void)
cmd eperase(void)
```

epset/eperase example

In the following CCL code example, the command **fandraw** opens a graphics window, assigns a slider to select object point values, draws the current lens, executes **epset** to preserve the lens drawing and draws an initial ray fan. Whenever the slider changes, the command **newfan** is called to redraw the ray fan for the new object point. **Newfan** executes **eperase** to remove the previous ray fan from the window and redraws a ray fan for the new object point. For simplicity, the number of rays is fixed at 10.

```
cmd fandraw()
{
    set(graphics_autoclear, off);
    graphwin_open(1);
    graphwin_reset;
    graphwin_sliderassign(drag,1,"Frac. Y Obj.",newfan,0,1,0);
    graphwin_slidershow;
    draw_lens;
    EPSET;
    draw_rays(0,10,-1,1);
}

cmd newfan(int slider_id, double slider_value)
{
    /* This command is executed whenever the slider is moved.
     * The new fractional object height is given by slider_value.
     * The supplied slider_id is not used, since there is only
     * one slider.
     */
    graphwin_open(1);
    EPERASE;
    draw_rays(slider_value,10,-1,1);
}
```

gfr

frame

The **frame** command draws a rectangular outline of the current viewport in the current graphics window.

definition

```
cmd frame(void)
```

frame example

In the following CCL code example, the cmd **frame_example** initializes a graphics window, defines a viewport centered in the window and draws a frame around it.

```
cmd frame_example()
{
    graphwin_open(1);
    graphwin_reset;
    viewport(.25, .75, .25, .75);
    FRAME;
}
```

glr**linereel(...)****glt****lineto(...)**

The **lineto** and **linereel** commands draw a line from the current world coordinate point to the specified world coordinate point in the current viewport of the current graphics window. The specified point then becomes the new “current” world coordinate point.

For the **lineto** command, the specified coordinates are relative to the origin of the current world coordinate system.

For the **linereel** command, the specified coordinates are relative to the last world coordinate point set by a previous line drawing or positioning command (e.g., **moveto**, **moverel**, **lineto**, or **linereel**).

The line drawn by **lineto** and **linereel** is clipped at the edges of the current viewport. It is possible for either the current world coordinate point or the specified point or both to be outside the viewport.

definitions

```
cmd lineto(X_Coordinate, Y_Coordinate)
cmd linereel(X_Coordinate, Y_Coordinate)
    real X_Coordinate
    real Y_Coordinate
```

lineto/linereel example

```
glt -131.7 12.65
glt xnext ynext
glr 0 .25
glr delx dely
```

glx**line_style(...)**

The **line_style** command is used to set the style of lines drawn in graphics windows. Five line styles are available:

line_style(sol); solid lines — equivalent to **line_style();**

line_style(dsh);	dashed lines
line_style(dot);	dotted lines
line_style(ddt);	dash-dot lines
line_style(ddd);	dash-dot-dot lines

The default line style is solid (sol). Note that some printers and plotters may not support styles other than solid. The line style remains in effect until the graphics window is reset.

definition

```
cmd line_style(style)
```

line_style example

The following commands draw the lens with a solid pen and then draw the default rays with a dotted pen:

```
drl;
gls dot;
ddr;
gls;
```

gmr

moverel(...)

gmv

moveto(...)

The **moveto** and **moverel** commands set the current world coordinate point in the current viewport of the current graphics window.

For the **moveto** command, the specified coordinates are relative to the origin of the current world coordinate system. For the **moverel** command, the specified coordinates are relative to the last world coordinate point set by a previous line drawing or positioning command (**moveto**, **moverel**, **lineto**, **linerel**, etc.).

definitions

```
cmd moveto(X_Coordinate, Y_Coordinate)
cmd moverel(X_Coordinate, Y_Coordinate)
    real X_Coordinate
    real Y_Coordinate
```

moveto/moverel example

```
gmv -132.7 14.65
gmv xstart ystart
gmr 0 .25
gmr delx dely
```

gpl**polylineto(...)****gpr****polylinerel(...)**

The **polylineto** and **polylinerel** commands draw a sequence of lines, i.e., a path, in the current viewport of the current graphics window. The points to be connected are specified by arrays of x and y coordinate values.

The path drawn by **polylineto** and **polylinerel** is defined by two arrays of real world coordinates; the first array specifies the sequence of x coordinates for points in the path while the second array specifies the corresponding sequence of y coordinates.

For the **polylineto** command, the specified coordinates are all relative to the origin of the current world coordinate system.

For the **polylinerel** command, the coordinates of each point are specified relative to the coordinates of the preceding point. The coordinates of the first point are specified relative to the last world coordinate point set by a previous line drawing or positioning command (e.g., **moveto**, **moverel**, **lineto**, or **linerel**).

The path drawn by **polylineto** and **polylinerel** is clipped at the edges of the current viewport. It is possible for the path to weave in and out of the viewport.

There is no special limit to the size of the specified arrays of coordinates (other than the amount of memory OSLO has available for storage).

definitions

```
cmd polylineto(X_Coordinates, Y_Coordinates, Number_of_Points)
cmd polylinerel(X_Coordinate, Y_Coordinate, Number_of_Points)
    real X_Coordinates[]
    real Y_Coordinates[]
    int Number of Points    {lolim = 2}
```

polylineto/polylinerel example

```
cmd draw_circle(real radius, real center_x, real center_y)
{
    int npts, angle;
    real x[100], y[100];

    for (npts = angle = 0; angle <= 360; angle += 5, npts += 1)
    {
        /* Note: Multiplication by "dr" converts degrees to
         * radians.
         */
        x[npts] = center_x + radius*cos(angle*dr);
        y[npts] = center_y + radius*sin(angle*dr);
    }
    POLYLINE TO(x, y, npts);
}
```

gpp**polypoint(...)**

The **polypoint** command draws a collection of points at locations specified by supplied arrays of x and y world coordinate values. On the screen, points are single pixels. In an HP-GL plot, points are the size of the pen tip.

The points in the current viewport at which points are to be drawn is defined by two arrays of real world coordinates; the first array specifies the sequence of x coordinates for the points while the second array specifies the corresponding sequence of y coordinates. Coordinates are specified relative to the origin of the current world coordinate system.

The points drawn by **polypoint** are clipped at the edges of the current viewport. There is no special limit to the size of the specified arrays of coordinates (other than the amount of memory OSLO has available for storage).

definition

```
cmd polypoint(X_Coordinates, Y_Coordinates, Number_of_Points)
    real X_Coordinates[]
    real Y_Coordinates[]
    int Number_of_Points {lolim = 2}
```

polypoint example

The following CCL code example draws 20,000 points randomly scattered over graphics window 1. The points are obtained by using the **rand** function. The example gives a good visual indication of the behavior of that function. Note that the built-in global variables, i and j , are used as loop counters since i and j are not declared locally.

```
cmd polypoint_example()
{
    real x[100], y[100];
    graphwin_open(1);
    graphwin_reset;
    pen;
    for (i = 0, i < 200; i++)
    {
        for (j = 0; j < 100; j++)
        {
            x[j] = rand();
            y[j] = rand();
        }
        POLYPOINT(x,y,100);
    }
}
```

gps**polysymbol(...)**

The **polysymbol** command draws a group of symbols at locations specified by supplied arrays of x and y world coordinate values. All symbols are identical except for location and are specified by a symbol number and symbol size.

The points in the current viewport at which symbols are to be drawn is defined by two arrays of real world coordinates; the first array specifies the sequence of x

coordinates for the symbols while the second array specifies the corresponding sequence of y coordinates. Coordinates are specified relative to the origin of the current world coordinate system.

The symbols drawn by **polysymbol** are clipped at the edges of the current viewport. There is no special limit to the size of the specified arrays of coordinates (other than the amount of memory OSLO has available for storage).

It is much more efficient to draw a group of symbols using **polysymbol** than to execute a sequence of **symbol** commands.

definition

```
cmd polysymbol(X_Coordinates, Y_Coordinates, Number_of_Points,
Symbol_Number, Symbol_Size)
    real X_Coordinates[]
    real Y_Coordinates[]
    int Number_of_Points {lolim = 2}
    int Symbol_Number {lolim = 0, hilim = 31}
    int Symbol_Size {lolim = 0, hilim = 16, default(noquery) = 0}
```

Note: A zero value for Symbol Size is interpreted as the standard (default) size.

polysymbol example

The following CCL code example draws symbol 1 at a specified number of points lying along a diagonal line which runs from the lower left corner to the upper right corner of graphics window 1. Note that the local arrays $x[]$ and $y[]$ may be dimensioned using the input point count.

```
cmd polysymbol_example(int npts)
{
    int j;
    real delta;
    real x[npts], y[npts];

    graphwin_open(1);
    graphwin_reset;
    delta = 1.0/npts;
    for (j = 0; j < npts; j++)
        x[j] = y[j] = (0.5 + j)*delta;
    POLYSYMBOL(x,y,npts,1,2);
}
```

gsh

gshow

The **gshow** command causes all queued drawing requests for the current vector graphics window to be displayed immediately on the screen.

For efficiency, requests to draw to the current vector graphics window are not always processed immediately. OSLO does not issue a multi-segment line drawing request until the end of a sequence of **lineto** commands has been reached (signaled, for example by the execution of a **moveto** command). In addition to this, the window system will generally not process drawing requests until control has returned to the command line.

The **gshow** command forces all pending drawing requests to be sent immediately to the window system. It is essential that **gshow** be executed at the completion of any intermediate drawing in an animated sequence.

definition

```
cmd gshow (void)
```

gshow example

Gshow is used in the **gclear** example to display the next image.

gsy

symbol(...)

The **symbol** command draws a specified symbol centered on the current world coordinate point in the current viewport of the current graphics window. The size of the symbol may optionally be specified with the command. If not specified, the size defaults to the standard size.

The symbols that may be drawn are identified by numbers 1, 2, ... , 31. The currently defined set, however, contains only the twelve symbols identified by Symbol Numbers 1, 2, ... , 12. Other symbols may be added in the future.

The currently defined symbols are the following:

1: +	7: circle
2: ×	8: filled circle
3: up triangle	9: filled up triangle
4: down triangle	10: filled down triangle
5: square	11: filled square
6: diamond	12: filled diamond

The standard size for a symbol is consistent with the standard size defined for characters in a graphics label. The symbols in the currently defined set all have a height and width equal to half the height of an uppercase label character.

The Symbol Size specified with the symbol command is a decimal fraction of the standard size. Thus a size of 0.5 would produce a symbol of half the standard size. A size of 2.0 would produce a symbol of twice the standard size. Note: A zero value for Symbol Size is interpreted as the standard (default) size.

definition

```
cmd symbol(Symbol_Number, Symbol_Size)
    int Symbol_Number {lolim = 0, hilim = 31}
    int Symbol_Size {lolim = 0, hilim = 16, default(noquery) = 0}
```

symbol example

```
symbol 1
symbol 2 1.5
```


gvp**viewport(...)**

The **viewport** command defines the area of the current vector graphics window that is to be used for subsequent drawing. The viewport area is defined by minimum and maximum window coordinates expressed as fractions of the full window width and height. The viewport definition remains in effect until another **viewport** command is executed.

The `Minimum_Viewport_X` and `Maximum_Viewport_X` are specified as fractions of the full graphics window *width*. The `Minimum_Viewport_Y` and `Maximum_Viewport_Y` are specified as fractions of the full graphics window *height*.

A default Label Size to be used for labels may be specified in the **viewport** command. If this is done, all labels within the viewport will have their aspect ratio (ratio of character height to character width) preserved regardless of how the graphics window is subsequently resized. If the Label Size is not specified in the viewport command, the standard default will be used. Note that it is impossible to maintain label size relative to the viewport and character aspect ratio simultaneously as the graphics window is resized in width and/or height.

When either a **graphwin_reset** or **gclear** command is executed, the viewport is initially set to encompass the entire graphics window (**viewport 0 1 0 1**).

definition

```
cmd viewport(Minimum_Viewport_X, Maximum_Viewport_X,
Minimum_Viewport_Y, Maximum_Viewport_Y, Label_Size)
  real Minimum_Viewport_X {lolim = 0, hilim = 1}
  real Maximum_Viewport_X {lolim = 0, hilim = 1}
  real Minimum_Viewport_Y {lolim = 0, hilim = 1}
  real Maximum_Viewport_Y {lolim = 0, hilim = 1}
  real Label_Size {lolim = 0, hilim = 16, default(noquery) = 0}
```

viewport example

```
viewport 0 .5 .5 1
viewport .25 .75 .25 .75 1
```

hpg**hpgl(...)**

The **hpgl** command inserts a Hewlett Packard Graphics Language (HP-GL) command string at the current point in the graphics command file for the currently open vector graphics window.

As graphics commands (**moveto**, **lineto**, **label**, etc.) are executed, thereby producing output to the current graphics window, a graphics command file is built concurrently within an OSLO memory area. This file is used both to recreate the contents of the graphics window when the window is resized and to create an HP-GL output file.

The **hpgl** command is used to store a literal HP-GL command string in this file at the point representing the current state of the graphics window.

The **hpgl** command has the same kind of variable argument list as is used for the **printf** command. The first argument, the format string, may either be a literal string or the id number of a string in the `a_string.ccl` file.

If the specified command does not terminate with a semicolon (;) a semicolon is automatically provided.

definition

```
cmd hpgl(format,[variables])
```

The “format” argument is either an integer id number identifying a format string stored in the CCL a_string.ccl file, or is the format string itself. The variables are any integers, real numbers or expressions which are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string.

hpgl example

```
hpgl "PG1"
hpgl "CI %d;" radius
```

lab

label(...)

The **label** command is used to draw text labels in the current vector graphics window. Labels may be single or multi-line and have properties defined by the **lsize**, **lspacing**, **langle**, and **lorigin** commands.

Labels are drawn with respect to an origin at the current world coordinate point. This origin point is normally set by a **moveto** or **moverel** command immediately preceding the **label** command. The actual location of the label relative to the origin point is controlled by the **lorigin** command. By default the lower left corner of the first label character is placed at the origin.

A size for the current label may be specified by an **lsize** command preceding the **label** command. If the size is not specified in this way, the size of the label defaults to the Label Size argument specified in the last executed **viewport** command. The default character size is such that 20 lines with 64 characters per line would completely fill the viewport.

A label may be drawn at any angle. The angle at which the current label is drawn may be specified by an **langle** command preceding the **label** command. By default, the label will be drawn horizontally and upright – i.e., a label angle of 0 degrees.

Spacing between characters in the current label may be specified by an **lspacing** command preceding the **label** command. By default, the spacing between characters will be approximately half the character width.

The **label** command has the same kind of variable argument list as is used for the **printf** command. The first argument, the format string, may either be a literal string or the id number of a string in the a_string.ccl file.

Multi-line labels may be drawn by using the newline character (**\n**) in the format string.

The OSLO graphics label font contains a number of special symbols identified by numbers 1,2,3, Symbols may be used in labels by entering an @ sign followed by the symbol number in the label format string – e.g., “Plot Symbols = @1 @2 @3 @4 @5 @6.”

definition

```
cmd label(format,[variables])
```

The “format” argument is either an integer id number identifying a format string stored in the CCL a_string.ccl file, or is the format string itself. The variables are any integers, real numbers or expressions that are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string.

label example

```
label "Cooke Triplet\nOPTICAL SYSTEM LAYOUT"
label "X = %-6.2f, Y = %-6.2f" x y
```

lan**langle(...)**

The **langle** command sets the angle at which the next vector graphics label is to be drawn. The specified angle is in degrees and increases in the clockwise direction. The **langle** command affects only the next label drawn to the current graphics window.

An angle of +90 degrees draws a vertical label with characters running from the top toward the bottom of the window. An angle of -90 degrees draws a vertical label with characters running from the bottom toward the top of the window. If no **langle** command precedes a label command, the label is drawn at an angle of 0 degrees – i.e., horizontally with characters running from left to right.

definition

```
cmd langle(Label_Angle)
  int Label_Angle {default(onprompt) = 0}
```

langle example

```
langle 90
langle -30
langle theta
```

lor**lorigin(...)**

The **lorigin** command defines how a vector graphics label will be drawn with respect to its origin (the current world coordinate point as specified, for example, by a **moveto** command). The Origin Code specified by the **lorigin** command is an integer in the range 1 through 9, 11 through 19, etc. It is equivalent to the number specified for the Hewlett Packard Graphics Language LO command.

The **lorigin** command only affects the next label to be drawn.

The Origin Code specified by the **lorigin** command is an integer of the form *ab* where *a* = 0, 1, ... , and *b* = 1, 2, ... , 9. Whatever the label angle may be, the definition of the Origin Code below applies to the label as viewed horizontally, running from left to right.

The digit *b* indicates the position of the label with respect to the origin, as follows:

Horizontal

- b* = 1,2,3: Left end of the label is aligned with the origin.
- b* = 4,5,6: Label is centered with respect to the origin.
- b* = 7,8,9: Right end of the label is aligned with the origin.

Vertical

$b = 1,4,7$: Vertical extent of the characters is wholly above the origin.
 $b = 2,5,8$: Characters extend equally above and below the origin.
 $b = 3,6,9$: Vertical extent of the characters is wholly below the origin.

A non-zero value of a causes extra space to be added between the origin and the label. A value of 1 adds approximately half a character space, a value of 2 adds approximately a full character space, etc. The space is added in either the horizontal or vertical directions, or both, depending on the positioning selected by the digit b .

The extra space selected by digit a depends on digit b as follows:

Horizontal

$b = 1,2,3$: The label is moved further to the right.
 $b = 7,8,9$: The label is moved further to the left.

Vertical

$b = 1,4,7$: The label is moved further up.
 $b = 3,6,9$: The label is moved further down.

If no **lorigin** command is executed prior to drawing a label, the Origin Code defaults to 1.

definition

```
cmd lorigin(Origin_Code)
    int Origin_Code
```

lorigin example

```
lorigin 3
lorigin 11
```

lsp

lspacing(...)

The **lspacing** command is used to adjust the amount of space between characters in a label and the space between lines of a multi-line label. It applies only to the next label drawn to the current vector graphics window.

By default, the *horizontal* space separating the right edge of an upper case “M” from the left edge of a following uppercase “M” is approximately 60% of the character width. The default *vertical* space separating the bottom of an upper case “M” from the top of an upper case “M” on the next line of a multi-line label is approximately 60% of the character height. Thus the default character-to-character separation is approximately 1.4 times the character width and the default line-to-line separation is approximately 1.4 times the character height.

The **lspacing** command allows adjustment of these spacings for the next label to be drawn. The unit of spacing adjustment for both Line Spacing and Character Spacing is 10% of the default character-to-character separation.

The specified spacing adjustments are algebraically added to the default separations. Thus a value of 0 for either Character Spacing or Line Spacing has no effect. A negative value reduces the separation while a positive value increases the separation. A value of -10 for the Character Spacing adjustment

and -20 for the Line Spacing adjustment would cause all characters in a label to be drawn on top of one another.

Labels are clipped to the edge of the current viewport. That is, a character which falls wholly or partially outside the current viewport will not be drawn.

definition

```
cmd lspacing(Character_Spacing, Line_Spacing)
    Real Character_Spacing
    Real Line_Spacing
```

lspacing example

```
lspacing -2
lspacing 0 2
lspacing 1.5 -1.5
```

lsz

lsize(...)

The **lsize** command sets the size of the next vector graphics label to be drawn in the current graphics window. The Label Size specified by the command is a decimal fraction of the standard (default) size. The standard (default) character size for graphics labels is such that 20 lines with 64 characters per line would completely fill the current viewport.

The **lsize** command specifies a size for the next label only as a decimal fraction of the standard size. Thus a size of 0.5 would produce labels of half the standard size. A size of 2.0 would produce labels of twice the standard size. Note: A zero value for Label_Size is interpreted as the standard (default) size.

definition

```
cmd lsize(Label_Size)
    Real Label_Size {lolim = 0, hilim = 16, default(noquery) = 0}
```

lsize example

```
lsize 2
```

pen

pen(...)

The **pen** command selects a color to be used for drawing to the current vector graphics window. The color is specified by a number called the Pen Number. When an HP-GL plot file is created for the current graphics window, the pen number generates an HP-GL SP instruction in the output file. The colors selected by the pen command are installation dependent. For gray-scale displays, colors are shades of gray.

For color displays, the currently defined color list is (assuming a white background):

```
Pen #1:  Black
Pen #2:  Green
Pen #3:  Cyan
Pen #4:  Red
```

Pen #5: Yellow
 Pen #6: Magenta
 Pen #7: Blue
 Pen #8: Orange

When a graphics window is cleared by executing either the **graphwin_reset** or **gclear** commands, the pen number is initialized to 1.

If a **pen** command is executed without specifying a Pen Number (or by specifying a Pen Number of 0), the next available color is used - e.g., if the current Pen Number is 1, the Pen Number is set to 2. When the end of the current color list is reached, the Pen Number cycles back to 1.

When an HP-GL plot file is created, pen numbers are forced to the range 1, 2, ..., 6. Pen number 7 becomes 1; 8 becomes 2.

definition

```
cmd pen(Pen_Number)
    int Pen_Number {lolim = 0, default(noquery) = 0}
```

pen example

```
pen
pen 3
```

sco

spline_coefs(...)

spi

spline_interp(...)

spp

spline_plot(...)

These commands perform spline fits to supplied data. The data for the spline fits should be placed in the global variable arrays `Xa[]` and `Ya[]`. The fit will be computed for `Ya[]` as a function of `Xa[]`. The `Xa[]` data points do not need to be equally spaced, but they must be unique and arranged in ascending order, i.e., `Xa[0] < Xa[1] < Xa[2] ...`.

After placing the data in the `Xa[]` and `Ya[]` arrays, a spline curve through the data can be plotted in the current graphics window by using the **spline_plot** command. `Number_of_spline_data_points` is the number of (x, y) pairs in the `Xa[]` and `Ya[]` arrays. `Number_of_interpolated_values` is the number of spline interpolations performed between each pair of data points. `Horizontal_axis_of_plot` is either "x" (if the data in `Xa[]` is to be used as the horizontal axis) or "y" (if `Ya[]` is to be used as the horizontal axis). Note that the data is plotted into the graphics window using the coordinates set by the **window** command.

If you want to compute the coefficients of a spline fit, so that you can calculate interpolated values, first issue the command **spline_coefs**. After the coefficients have been computed, an interpolated value of y, given a value of x, can be found with the function **spline_interp**. Argument is the value of x for which y is to be calculated. The value of x should be between the minimum and maximum values of `Xa[]`.

definitions

```
cmd Spline_coefs(Number_of_spline_data_points)
real Spline_interp(Argument)
cmd Spline_plot(Number_of_spline_data_points, Number_of_interpolated_values,
    Horizontal_axis_of_plot)
```

win

window(...)

Most drawing commands (e.g., **moveto**, **lineto**) specify “world coordinates.” The **window** command defines the world coordinates to be used for drawing in the current viewport of the current graphics window and remains in effect until another **window** command is executed. The **window** command also specifies whether or not drawing to the viewport is to be isometric (See below).

The Minimum X and Minimum Y values specified in the window command are the world coordinates of the point at the lower left corner of the current viewport. The Maximum X and Maximum Y values are the world coordinates of the upper right corner of the current viewport.

The Mapping Mode specifies whether or not drawing in the viewport is to be isometric. If drawing is isometric, the 2-dimensional shapes of objects drawn to the window are preserved and world coordinates are adjusted so that 1 unit in the X direction is drawn to the same length as 1 unit in the Y direction - regardless of the viewport shape or how the window is resized.

If mapping is isometric, a drawing in the window may be clipped to the viewport in different ways as the graphics window is resized. In addition, if the ratio of height to width of the viewport expressed in world coordinates is different from the ratio of height to width of the viewport as seen on the screen, either the Minimum/Maximum X or the Minimum/Maximum Y coordinates will be adjusted to correct the situation. The correction is done in such a way as to preserve the world coordinate point at the geometrical center of the viewport.

If the Mapping Mode is “To Window” (the default) rather than “Isometric,” then drawings in the current viewport will stretch or shrink independently in the vertical and horizontal directions as the window is variously resized in height and/or width.

definition

```
cmd window(Mapping_Mode, Minimum_X, Maximum_X, Minimum_Y, Maximum_Y)
    int Mapping_Mode {list = Map_Options, default(noquery) = tow}
    real Minimum_X
    real Maximum_X
    real Minimum_Y
    real Maximum_Y

list Map_Options {
    "{tow}To Window" = 0,
    "{iso}Isometric" = 1
}
```

window example

```
window -.015 8.32 5 50
window iso -.05 .05 0 .1
```

Miscellaneous

abt

abort(...)

The **abort** command immediately terminates a CCL or SCP procedure and returns control to the command line. If a message is supplied with the **abort** command, it is displayed below the command line, accompanied by a “beep” sound. If no message is supplied, the procedure terminates silently.

The **abort** command has the same kind of variable argument list as is used for the **printf** command. The first argument, the format string, may either be a literal string or the id number of a string in the `a_string.ccl` file.

The message specified by the **abort** command may contain up to three lines, but may be omitted.

The **abort** command allows a CCL procedure to be terminated from a point deep within a nest of loops or from within a low level procedure which has been called by the main procedure.

definition

```
cmd abort(format,[variables])
```

The “format” argument is either an integer id number identifying a format string stored in the CCL `a_string.ccl` file, or is the format string itself.

The variables are any integers, real numbers or expressions which are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string.

abort example

```
for (i = 0; i < 10; i++)
{
    y = 0;
    for (j = 0; j < 10; j++)
    {
        if ((x = mycalc(i,j)) < 1000)
            y += x;
        else
            ABORT("Calculation diverged for i = %d", i);
    }
    printf("y[%d] = %g\n", i, y);
}
```

bee

beep

The **beep** command simply generates a “beep” sound (sometimes called the system bell). It has no arguments.

If, for example, it is desired to draw attention to a message that has been printed from within a CCL procedure, a **beep** command could be placed immediately before or after the message command in the procedure.

beep example

The following CCL command sounds a beep at 1 second intervals until the user presses the ESCAPE key. It then terminates.


```

cmd beeper
{
    for (;;)
    {
        time_reset;
        while (time < 1000) ;
        BEEP;
        if (escape) break;
    }
}

```

esc

int escape

Escape is a CCL function which returns a value of 1 (True) if the user has pressed the ESCAPE key since the current CCL procedure was started or since the last call to escape was made. If the user has not pressed the ESCAPE key, **escape** returns a 0 (False) value.

This function may be used within a CCL procedure to give the user a chance to terminate the procedure at an appropriate breaking point.

escape example

```

cmd iterative_calculation()
{
    int iter_count = 0;
    for (;;)
    {
        ...code performing one iteration of a calculation...
        iter_count++;
        if (ESCAPE)
            abort("Stopped after iteration %d", iter_count);
    }
}

```

exe

execute(...)

The **execute** command allows a character string to be executed from within a CCL or SCP procedure. The effect is the same as if the string were executed directly from the command line. The string is executed immediately - i.e., unlike the **execute_scpfile** command, execution is not deferred until control returns to the command line.

Note that the string to be executed may contain several commands separated by semicolons.

definition

```

cmd execute(Command String)
    char Command String[] {lolim = 1}

```

execute example

The following CCL code example defines a command “run” which may be used to execute any of several frequently used command strings by entering **run a**, **run b**, etc., from the command line.

In `a_list.ccl`, define:

```
list run_cmds {
    a = "command string 1",
    b = "command string 2",
    c = "command string 3"
}
```

In a `.ccl` code file, define the “run” command:

```
char run_cmd[] {list = run_cmds}
cmd run (char run_cmd[])
{
    execute run_cmd;
}
```

exi

exit

Terminates the OSLO program. When **exit** or **quit** is executed, various cleanup operations are performed - e.g., the current sizes and locations of OSLO windows are saved so that windows will appear unchanged when OSLO is next started.

A confirmation prompt is posted in the alert box prior to terminating OSLO so that the program is not terminated inadvertently.

hlt

halt(...)

The **halt** command is used as an aid in debugging. It is ignored unless the **Debug** preference is On.

If the **halt** command is encountered in a CCL procedure while **Debug** is On, control is returned to the command line. The procedure is not terminated, however, but is simply suspended. Any message supplied with the **halt** command is printed below the command line when the procedure halts.

While the procedure is halted, values of various global and local CCL variables may be examined and changed, and other procedures may be executed from the command line.

If the command line is empty (blank) when the RETURN key is pressed, execution of the original procedure continues from the statement immediately following the **halt** command.

The **halt** command has the same kind of variable argument list as is used for the **printf** command. The first argument, the format string, may either be a literal string or the id number of a string in the `a_string.ccl` file.

The message specified by the **halt** command may contain up to three lines, but may be omitted. If the message is omitted, the message “Execution halted:” is printed.

While procedure execution is halted, the procedure may be terminated by selecting the CANCEL button or pressing the ESCAPE key.

If it is desired to continue running the procedure without further “halts,” the **Debug** preference may be turned Off.

The value of any CCL variable (except protected built-in variables) may be examined or changed from the command line while the procedure is halted. Local variables in a procedure take precedence over global variables of the same name.

definition

```
cmd halt(format,[variables])
```

The “format” argument is either an integer id number identifying a format string stored in the CCL a_string.ccl file, or is the format string itself.

The variables are any integers, real numbers or expressions which are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string.

halt example

```
for (i = 0; i < 10; i++)
{
    y = 0;
    for (j = 0; j < 10; j++)
    {
        x = mycalc(i,j);
        HALT("Halted at i = %d, j = %d", i, j);
        y += x;
    }
    printf("y[%d] = %g\n", i, y);
}
```

pau

pause(...)

The **pause** command allows the user a choice of allowing a CCL or SCP procedure to continue or to terminate.

This command causes a specified message to be printed next to the command line in the OSLO main window. The message line “Press OK (or Return) to continue...” is appended to the specified message. OSLO then waits for the user to respond.

If the user responds by pressing the RETURN key or selecting the OK button, the procedure continues. If the user presses the ESCAPE key or selects the CANCEL button, the procedure terminates and control is returned to the command line.

The **pause** command has the same kind of variable argument list as is used for the **printf** command. The first argument, the format string, may either be a literal string or the id number of a string in the a_string.ccl file.

The message specified by the **pause** command may contain one or two lines.

definition

```
cmd pause(format,[variables])
```

The “format” argument is either an integer id number identifying a format string stored in the CCL a_string.ccl file, or is the format string itself.

The variables are any integers, real numbers or expressions which are needed to satisfy conversion specifications (%d, %f, etc.) appearing in the format string.

pause example

```
cmd long_calculation()
{
```

```

...initialization code...
PAUSE "The calculation is likely to take 4 or 5 minutes.";
...code performing a long calculation...
}

```

qui

quit

Terminates the OSLO program. When **exit** or **quit** is executed, various cleanup operations are performed - e.g., the current sizes and locations of OSLO windows are saved so that windows will appear unchanged when OSLO is next started.

A confirmation prompt is posted in the alert box prior to terminating OSLO so that the program is not terminated inadvertently.

tbk

traceback

The **traceback** command is used as an aid in debugging and is only effective when the **Debug** preference is set to On. It is normally entered from the command line when a CCL procedure is halted as a result of executing a halt command, but can be placed anywhere in a CCL procedure.

The **traceback** command produces a listing of the names of all CCL procedures which have been descended through at the time the **traceback** command is executed.

A complex CCL procedure may descend through a sequence of sub-procedures - i.e., a command may call a command which calls another command, etc. Depending on actions taken at various branches in various procedures, a number of different paths may be taken through a sequence of sub-procedures. The **traceback** allows this sequence to be viewed. If it is always desired to view a traceback list whenever a **halt** command is executed, a **traceback** command would be placed immediately ahead of the **halt** command.

If preference **Debug** is set to On, the occurrence of a fatal error in the execution of a CCL procedure will automatically generate a traceback list in the current text output window when the error is encountered.

traceback example

If a CCL command, **mycmd**, has called a command, **mycalc**, which has called a function, **myfct**, which has executed a **halt** command, the **traceback** command would produce the following listing in the current text output window:

```

Traceback:
  halt
  myfct
  mycalc
  mycmd

```

tim

int time

Time is an integer function which returns the elapsed time, in milliseconds, since the **time_reset** command was last executed.

If **time_reset** has not been executed prior to calling the **time** function, **time_reset** is automatically executed and the **time** function returns a zero value.

time example

In the following CCL code fragment, **time_reset** and **time** are used to check the time taken to carry out a calculation.

```
time_reset;
...code to carry out calculation...
printf("This procedure took %.3f seconds\n" TIME/1000);
```

tmr

time_reset

The **time_reset** command resets the system timer. It has no arguments. Any subsequent call to the **time** function will return the elapsed time since **time_reset** was last executed.

time_reset example

```
TIME_RESET;
...code to carry out calculation...
printf("This procedure took %.3f seconds\n" time/1000);
```

tst

timestr(...)

The **timestr** command can be used to obtain a formatted string containing the current date and/or time. **Output_string** is a string variable, and **format_string** determines the contents of **output_string**. By including the following format specifiers in **format_string**, the date and time information can be placed in **output_string**.

Specifier	Description
%a	Abbreviated name for the weekday
%A	Full name for the weekday
%b	Abbreviated name for the month
%B	Full name for the month
%c	Date and time
%d	Day of the month (decimal; 01–31)
%H	Hour (decimal; 24-hour clock, 00–23)
%I	Hour (decimal; 12-hour clock, 01–12)
%j	Day of the year (decimal; 001–366)
%m	Month (decimal; 01–12)
%M	Minute (decimal; 00–59)
%p	AM or PM indicator

%S	Second (decimal; 00–59)
%U	Week of the year (decimal; Sunday taken as the first day of a week, 00–51)
%w	Weekday (decimal; 0–6, Sunday is 0)
%W	Week of the year (decimal; Monday taken as the first day of a week, 00–51)
%x	Date
%X	Time
%y	Year without the century (decimal; 00–99)
%Y	Year with century (decimal)
%Z	Time zone

The results of this command are determined by the settings of the Windows International Control Panel.

definition

```
cmd timestr(output_string, format_string)
```

timestr example

To print the current date and time:

```
timestr(astr, "The date is %x.\nThe time is %X\n");
printf(astr);
```

xeq

execute_scfile(...)

The **execute_scfile** command executes the contents of a specified text file. The effect is the same as typing the contents of the file into the command line. The file name is specified explicitly and no special filetype or directory controls are imposed on it.

The **execute_scfile** command is intended to be used on the command line. If it appears in a compiled CCL procedure, its execution will be deferred until control returns to the command line.

Note that this command provides an informal means of saving a frequently used sequence of commands for repeated execution. The specified file is interpreted each time it is executed. It is not compiled as a CCL command procedure.

Index

A

- aberration 256
 - Aldis theorem 268
 - Buchdahl format 259
 - fifth-order 259
 - paraxial chromatic 257
 - pupil 258
 - Seidel wavefront 264
 - third-order (Seidel) 257
- aberration mode 213
- air 53
- air-to-air component 206, 404
- Aldis theorem 268
- alternate surface intersection 62
- angle solve 41, 42
- aperture
 - actions 52
 - ANDing/ORing rules 52
 - array channels 66
 - boundary 51
 - check 48
 - checking 215
 - combining groups 52
 - delete special 52
 - draw options 63
 - examples 155
 - groups 52
 - in non-sequential group 109
 - lens drawing 242
 - number on surface 51
 - pickup 49
 - pickup special 52
 - radius 48
 - shape 51
 - solved 49
 - types 49
- aperture checking 213
- aperture data 202
- aperture radius type 526
- aperture stop 50, 213
 - drawing 244
- aplanatic ray aiming 215
- a-ray
 - paraxial 253
- arc extended source 437, 443
- argument definitions 520
- arguments 492
- array 64
 - channel clipping 66
 - channel selection 65
 - channel surface 64
- ASA 383
 - run in background (UNIX) 386
 - run in foreground (Windows) 386
 - setting up 383
 - solution files 385
- aspheric surface type 528
- astigmatism

- source 215

- AutoCAD™ export 168
- Autodraw 125
- autofocus 335
- axial and elliptical gradient 102
- axial and sinusoidal radial gradient 102
- axial and tapered radial gradient 103
- axial ray
 - paraxial 253
 - slope 249
- axial ray height solve 46

B

- base coordinate system 69
- beam angle 215
- bend surface 68
- biconic surface 78
- binary files 152
- binary surface
 - See diffractive surface 85
- bitmap output files (*.bmp) 172
- boundary conditions 125, 208
- b-ray
 - paraxial 253

C

- Calculate
 - Aberration Analysis 256
 - Energy Distribution 321
 - Line Spread/Knife Edge 325
 - Paraxial Analysis 251
 - Spot Size Analysis 292
 - Trace Ray 269
 - Wavefront Analysis 265, 299
- calculator
 - using command line as 524
- callback level 214
- catalog glasses 54
- catalog lens
 - convert to surface data 36
 - database 37
- catalog lenses
 - database window 158
 - new file 157
- CCL
 - automatic compilation 482
 - compilation 547, 549
 - compiler options 549
 - constants 540
 - data types 538
 - differences from C 536, 544
 - directory 155
 - error handling 550
 - integer division 546
 - introduction 536
 - language elements 541
 - modules 550
 - object code 155

- operators 541
- pass by reference 545
- predefined global variables 540
- preprocessor 538
- procedure format 542
- reserved keywords 541
- set apertures 390
- source code files 547
- strings 543
- using with OSLO 547
- variable arrays 539, 546
- variable scope 539
- central coma 429
- change table tolerancing 424
 - aberration definitions 427
 - output format 425
 - ray table 427
- checked apertures 48
- chief ray
 - paraxial 253
- chief ray height 250
- chief ray height solve 46
- chromatic coordinate 101
- chromatic correction 345, 361
- clipboard 30, 31
- coating 105
- coating specifications 234
- Coddington curves 277
- CODE V™ files 161
- cold stop 394
- coma
 - elliptical 261
- command line 492, 506
- commands
 - arguments 19, 492
 - data I/O and display 495
 - debugging 614
 - drawing lines 597
 - format 492
 - graphics library 594
 - graphics symbols 602
 - graphics viewport 603
 - graphics window 571
 - introduction to using 491
 - labeling graphics 604
 - lens editing 506
 - lens surface data 508
 - mathematical functions 590
 - mathematics library 584
 - operating conditions 514
 - optical analysis 500
 - optimization 505
 - program control 610
 - question mark argument 492
 - solving equations 589
 - sorting 589
 - spline fits 608
 - string functions 591
 - syntax 491
 - tables of 494
 - text editor window 567
 - text output 582
 - text window 573
 - time 614

- trigonometric functions 590
- window management 567
- compensating parameters 417, 422, 425
- configuration data 22
- conic surface 72
- Conrady D minus d 274
- Conrady D-d method 345, 361
- contour plotting 594
- contouring
 - point spread function 319
- conventions
 - symmetry 258, 279
- coordinate matrix 220
- coordinate return 68
- coordinates
 - ray trace data 270
- coordinates spreadsheet 69
- coupling efficiency
 - definition 455, 457
- Cox library 154
- critical angle 24, 63
- current wavelength 203, 205
- curvature 40
- curvature pickup 44
- curvature type 525
- cut/copy/paste 30, 31
- cylindrical surface 73

D

- damping factor 387
- decentered surfaces 67
- decentration tolerance 205
- default drawing rays 228, 244
- degree of polarization 334
- Delano diagram 394
- delete spreadsheet rows 32
- derivative increments 367, 373, 379
- derivative matrix output 379
- designer 214
- diffraction efficiency
 - displaying local 270
- diffractive phase 221
- diffractive surface type 529
- diffractive surface types 85
- diffractive zones 221
- dispersion equation
 - Conrady formula 187
 - Laurent series 187, 188
 - Sellmeier series 187
- distortion
 - correction 345
 - operand value 363
 - plotting 277
- DLL 559
 - accessing global variables 560
 - accessing spreadsheet buffer 561
 - compiling and linking 562
 - eikonal surface 564
 - executing OSLO commands 561
 - external operands procedures 562
 - loading library 565
 - user-defined gradient 564
 - user-defined surface 563
 - writing source code 559

draw ray trajectories 229
draw surface options 63
drawing in a graphics window 597
drawing lines 597
dummy surface 63
DXF
 output 242
DXF files 168
dynamic-link libraries 559

E

echo preference 155
edge contact solve 47
edge thickness 217
edge thickness operands 346
edit
 undo (revert) 18
edit extended sources 434
effective focal length 27, 250, 251
effective source 458
eikonal surface 108, 555
 DLL 564
 global variables 557
element drawing 230
 bubbles and inclusions 232
 centering tolerance 233
 coating specification 234
 current lens data items 235
 current tolerance data 236
 default values 235
 ground surfaces 231
 inhomogeneity 232
 protective chamfer 233
 spreadsheet 232
 stress birefringence 232
 striae 232
 surface form 233
 surface imperfections 234
 surface roughness 231, 234
element drawing defaults 237
Element groups 34
ellipsoidal surface 72
elliptical aperture 50
ensquared energy distribution
 geometric 324
entrance beam radius 28, 249
entrance pupil 250
entry port 35
environment 519
error function 211
 appending 346
 generate 343
 use from another lens 40
evaluation mode 213
exit OSLO 180
exit port 35
exit pupil 250
exponentiation operator 523, 545
export lens drawing 242
extended aperture ray aiming 215
extended sources
 arc 437, 443
 edit 434
 Lambertian 434, 443

 LED 436, 443
 user-defined 439
 view 443
eye relief 252

F

Fast Fourier Transform 585
fiber mode
 Gaussian 456
 step-index 456
 user-defined 456
field
 points 28
field aberrations 282
field angle 28, 250
field points 209
 entering 350
 vignetting 350
field sags
 plotting 279
File
 New lens 156, 157
 Save 161
file commands 574
Files
 DXF, IGES 168
files
 apertures 155
 binary 152
 DXF, IGES 152, 156, 168
 graphics output 172
 importing 161
 installation directory 152
 lens 152
 log 155
 OSLO directory structure 152
 print 170
 Private directory 152
 Public directory 152
 reopening previous lens 180
 text output 170, 171
f-number 28, 249, 252
focal length 27, 250
fonts
 changing 483
footprint plot 391
format of commands 492
fractional coordinates 209
Fresnel surface 24, 62
fringe tolerance 205
fringe wavelength 205

G

Gaussian beam
 ABCD 448
 astigmatic trace 451
 coordinate systems for output 453
 general astigmatism 454
 skew trace 451
Gaussian image distance 250
Gaussian image height 250, 252
Gaussian quadrature 343
GENII-Plus files 161
ghost images 393

- glass
 - air 53
 - catalogs 54
 - data 54, 55
 - direct specification 55
 - entering in lens 53
 - fixing optical properties 60
 - model 55
 - options 53
 - pickup 54
 - sorting 54, 55
 - spreadsheet 54, 55
 - variable 60
- glass catalog
 - adding glass to private 184
 - Corning 184
 - Hoya 184
 - internal transmittance 56
 - obsolete 184
 - Ohara 184
 - Schott 184, 187, 188
 - shared 184
- glass catalog manager 184
- glass name 532
- glass type 527
- global coordinates 69
 - converting to local 69
 - reference surface 270
 - tilt limits 70
- global distance 219
- global editing of lens data 506
- global matrix 220
- global optimization 383
- global variables 208
 - lens data 525
 - predefined 524
- gradient index
 - apertures 215
- gradient index medium type 529
- gradient index types 98
- gradient index value 222
- Gradium™ gradient index 100
- graphic sliders 374
- graphics
 - output 171, 172
- graphics library commands 594
- graphics window
 - close 479
 - copying contents to clipboard 481
 - new 131, 479
 - open 131, 479
 - ordinary 127
 - reset 129, 479
 - saving contents to file 480
 - title 131, 142, 479
 - updatable 127
 - updating all 480
- graphics window commands 571
- graphics window sliders 567
- grating 85
 - blaze order 85
 - groove depth 85
 - order 85
 - spacing 85
- group
 - non-sequential surfaces 69, 108
 - print 214
- groups
 - element 34
 - non-sequential 35
- H**
 - hard copy 171
 - hatched reflector 53
 - help system
 - command reference 18, 19
 - hidden operands 212
 - hidden surface 63
 - hologram 85
 - HP-GL output 172
 - h-tanU curves 215
 - hyperboloidal surface 72
- I**
 - I/O routines 574
 - IGES files 168
 - image distance 250
 - image numerical aperture 249
 - image surface 213
 - importing lenses 161
 - input and output routines 574
 - insert
 - catalog lens 37
 - error function 40
 - lens file 40
 - insert spreadsheet rows 32
 - installation
 - PC 3
 - UNIX 3
 - installation directory 153
 - interactive design window 374
 - interferometric analysis 94
 - irregularity tolerance 205
 - ISO 10110 aspheric types 71
 - ISO 10110 drawings 230
 - ISO asymmetric cone surface 80
 - ISO biconic surface 78
 - ISO symmetric asphere 76
 - ISO symmetric cone surface 79
 - ISO X-toric surface 77
 - ISO Y-toric surface 77, 78
 - iteration 376
- J**
 - Jones matrix 108
- K**
 - kinoform
 - See diffractive surface 85
 - knife edge distribution
 - diffraction 325
 - geometric 327, 328
 - Köhler illumination 457
- L**
 - labels
 - in graphics 604
 - Lagrange invariant 250, 252

- Lagrange multipliers 376
- Lambertian extended source 434, 443
- laser damage specification 234
- lateral color
 - plotting 278
- lateral shear 429, 430
- LED extended source 436, 443
- lens
 - angle solves 42
 - aperture radius 48
 - aperture shapes 51
 - aperture stop 50
 - aperture types 49, 50
 - arrays 64
 - ASI surface 62
 - aspheric 71
 - asymmetric CGH 87, 90
 - asymmetric general asphere 75
 - axial ray slope 28
 - base coordinate system 69
 - biconic surface 78
 - canonical limits on tilt angles 70
 - catalog 157
 - catalog glasses 54
 - combination axial gradients 102
 - configuration data 22
 - configuration items 22
 - conic surface 72
 - coordinate pickup 67
 - coordinates 69
 - curvature pickup 44
 - curvature variables 45
 - custom 157
 - cut/copy/paste surfaces 31
 - cylindrical 73
 - delete surfaces 32
 - diffractive surfaces 85
 - editing 32, 37, 40
 - eikonal surface 108
 - element groups 34
 - files 152
 - fixing glass 60
 - Fresnel 24, 62
 - glass 53
 - glass pickup 54
 - global coordinates 69
 - global/local conversion 69
 - gradient index types 98
 - GRADIUM™ 100
 - holographic surface 85
 - insert catalog lens 37
 - insert lens file 40
 - insert surfaces 32
 - invert surface data 32
 - ISO aspheres 71
 - ISO asymmetric cone surface 80
 - ISO Symmetric Asphere 76
 - ISO symmetric cone surface 79
 - ISO X-toric surface 77
 - ISO Y-toric surface 77, 78
 - Jones matrix 108
 - length 250
 - length pickup 47
 - linear grating 85
 - local coordinates 69
 - Luneburg lens 104
 - material 53
 - Maxwell's fisheye gradient 104
 - minus length pickup 47
 - minus thickness pickup 47
 - model glass 55
 - name 20
 - non-sequential surfaces 35
 - numerical aperture 28
 - perfect 70, 159
 - polarization element 107
 - radius of curvature 40
 - ray angle solves 41
 - rco in non-sequential group 68
 - reference surface 26, 50
 - reflecting surface 53
 - refractive index 55
 - removing fixed attribute 36
 - restoring coordinates 67
 - return-coordinates surface 68
 - reverse surfaces 32
 - Selfoc™ gradient 99
 - skip surface 62
 - special apertures 50
 - special surface data 61
 - special variable 45
 - spherical gradient 103
 - spline surface 82, 83
 - standard asphere 74
 - surface control 62
 - surface data 18, 28
 - surface drawing options 63
 - surface note 61
 - symmetric CGH 86, 89
 - symmetric general asphere 74
 - test glass list 45
 - thickness 46
 - thickness in global coordinates 69
 - thickness pickup 47
 - thickness variables 48
 - tilt and bend surface 68
 - TIR surface 24, 63
 - toroidal surface 72
 - ungrouping surfaces 36
 - updating 18
 - user defined thickness pickup 47
 - user gradient 105
 - user-defined surface 110
 - view special data 201
 - working f-number 28
 - Zernike phase surface 91
 - Zernike sag surface 80, 81
- lens array
 - global variables 531
- lens drawing
 - aperture stop 244
 - apertures 242
 - default rays 244
 - draw rays 228
 - draw reflector hatching 244
 - DXF, IGES output 242
 - element drawings 230
 - operating conditions 225, 238

- plan view 225
- scale 225
- solid model 227
- spreadsheet 224
- surface range 239
- viewing direction 239
- viewing matrix 220
- wire frame 227
- zoom graphics 240
- lens volume 220
- lens weight 220
- Lens_auto_open 180
- lenses
 - demonstration 153
 - library 154
 - vendors 154
- line spread function
 - diffraction 325
 - geometric 327, 328
- linear polarizer 108
- list arguments 521
- Lobatto quadrature 343
- local coordinate system 69
- log file 155
- Luneburg lens 104

M

- magnification 250, 251, 252
- marginal ray
 - paraxial 253
- matched illumination 457
- mathematical functions 590
- mathematics commands library 584
- matrix operations 586, 587, 588
- Maxwell's fisheye gradient 104
- merge lens data 40
- merit function
 - See error function 343
- message area 19
- minimized windows 484
- mirror 53
- Monte Carlo optimization 383
- MTF
 - diffraction vs. geometric 304
 - plotting 308, 309
 - plotting through-focus 309
 - through-focus 307
 - through-frequency 307
- MTF optimization 352
- MTF/Wvf tolerancing
 - output formats 419
 - performance analysis 420
 - range of performance change 420
- multiconfiguration
 - start/stop skip 62
- multiconfiguration data 22
- multiconfiguration variables 208, 368
- multilayer coating 105

N

- narcissus 394
- New lens 156, 157
- non-sequential surface group
 - global variables 530

- number of rays in fan 213
- numerical aperture 28, 251

O

- object coordinates 209
- object distance 250
- object height 28, 250
- object numerical aperture 28, 249
- oblate spheroid 72
- OPD
 - peak-to-valley 266, 300
 - root-mean-square 266, 300
- OPD curves 215, 276
- OPD units 214
- operand components
 - aberration 357
 - CCL & SCP 362
 - external 362
 - general ray 359
 - ordinary ray 361
 - paraxial 357
 - reference ray 360
 - spot diagram 361
 - statistical 361
 - system 354
- operands 211
 - constraint mode 362
 - definitions 353
 - distortion 363
 - generate error function 343
 - hidden 362
 - minimize mode 362
 - name 362
 - user-defined 559
 - weight 362
- operating conditions
 - lens drawing 216
 - optimization 216, 387
 - partial coherence 461
 - spot diagram 216
- optical coordinates 318
- Optics Toolbox 154
- optimization
 - append error function 346
 - autofocus 335
 - chromatic correction 345, 361
 - constraints 376
 - damping factor 387
 - distortion 345
 - field points 350
 - Gaussian quadrature 343
 - generate error function 343
 - global 383
 - interactive design 374
 - iteration modes 376
 - minimization 376
 - operands entry 353
 - operating conditions 387
 - spot diagram set 352
 - variables 367
 - view derivative matrix 379
- optimization data 211
 - global variables 531
- Options

Change Table Tolerancing 424
 Fiber Coupling 455
 Fonts 483
 MTF/Wvf Tolerancing 416
 Partial Coherence 457
 Polarization Transmittance 330
 PSF Contouring 317
 Set Preference 173
 Show Preference 173
 Update Glass Catalog 184
 Update Tolerance Data 398
 User-Defined Tolerancing 421
 ordinary ray 209, 350
 OSDATA 152
 OSLO Light 153
 OSLO PRO
 demo lenses 153
 OSLO Series 2 files 161
 OSLO SIX
 demo lenses 153
 OSLO-EDU 153
 output_log preference 155
 overlap integral 455
P
 page mode 170
 page setup 171, 172
 paraboloidal surface 72
 paraxial approximation 251
 paraxial calculations
 meridian 254
 paraxial constants 251
 paraxial imagery 251
 paraxial ray aiming 215
 paraxial ray trace 252
 paraxial setup data 249
 partial coherence 457, 461
 effective source position in pupil 461
 effective source properties 462
 effective source radius 461
 ideal image properties 462
 partial coherence operating conditions 461
 perfect lens 70
 Petzval radius 250, 252
 phase
 diffractive surface 221
 phase surface
 See diffractive surface 85
 pickup
 coordinates 67
 curvature 44
 length 47
 minus curvature 44
 minus length 47
 minus thickness 47
 thickness 47
 user defined curvature 44
 user defined thickness 47
 pickup data 202
 pixelated object 446
 plan view lens drawing 225
 point spread function
 central moments 322
 polarization element 107

polarization ellipse 271, 334
 polarization ray trace 271, 332, 333
 polarization transmittance 330, 331, 332
 polynomial asphere 71
 preferences
 Address1, Address2, Address3 175
 CCL_auto_compile 175, 482
 CCL_compiler_options 175
 Cmd_history_aliases 175, 493
 Current_directory 175
 Current_text_directory 175
 Designer 175
 echo 155
 Edit_file 175
 Element_drawing_commas 175
 Entrance_pup_wvfnt_coords 175
 Expiration_date 175
 Five_place_numeric_output 176
 Graphics_alternate_mode 176
 Graphics_autoclear 176
 Graphics_axes 176
 Graphics_black_and_white 176
 Graphics_current_scale 176
 Graphics_HPGL2 176
 Graphics_labels 176
 Graphics_PCL 176
 Graphics_pen_sequence 177
 Graphics_pen_width 177
 Graphics_white_bkgnd 173, 177
 lauo 180
 Lens_auto_open 177
 Lens_file 177
 Max_config_items 177
 Max_spd_rays 177
 Max_surfaces 177
 Max_users 177
 Max_wavelengths 177
 No_error_boxes 178
 Non_zero_special_data 178
 obtaining current values of 532
 Oprd_maxconops 178
 Oprd_maxextemp 178
 Oprd_maxfpts 178
 Oprd_maxoperands 178
 Oprd_maxrays 178
 Osddata_path 178
 Output_echo 178, 524
 Output_log 178
 output_log 155
 Output_page_mode 140, 179, 476
 Output_text 138, 179, 476
 page 170
 Plot_destination 179
 Plot_file 179
 Print_destination 179
 Print_file 179
 Program_level 179
 Public_directory 179
 retrieving values of 174
 Revert_enable 179
 Rpt_gfx_date_and_time 179
 Show_menubar 179
 Show_toolbar 180
 Spreadsheet_curvature_mode 180

- spreadsheet_curvature_mode 40
- Temporary_directory 180
- Test_glass_file 180
- Wavelength_default 180
- Weight_default 180
- primary color 257
- principal points 250
- principal ray
 - paraxial 253
- print graphics 171
- Print text window 170
- Private directory 152, 155
- prompt area 19
- PSF contouring 319
 - equal contour increments 319
 - specified contour levels 320
- Public directory 152
- pupil coordinates 209

Q

- quadrangular aperture 50
- question mark argument 19, 492
- quit OSLO 180

R

- Radau quadrature 343
- radial energy distribution
 - diffraction 321
 - geometric 324
- radial gradient 98, 102
- radius of curvature 40
- radius tolerance 205
- ray
 - draw trajectories 229
- ray aiming 215
- ray aiming modes 215
- ray blocking 48
- ray coordinates 209
- ray fan
 - tracing and printing 273
- ray set 209
 - entering 351
 - ordinary ray 351
 - reference ray 351
 - vignetting 351
- ray trace
 - aperture checking 48
 - array channel clipping 66
 - array channel selection 65
 - ASI surface 62
 - sequence with rco surface 68
- ray-intercept curves 275
- rectangular aperture 50
- reference ray 209, 351
- reference sphere 214
- reference surface 26, 50, 213
- reflector 53
 - draw hatch marks 244
 - TIR only 24, 63
- refractive index 55, 204
 - show 204
 - variation with respect to temperature 188
- refractive index tolerance 205
- regions in non-sequential group 109

- regular lens array 64, 65
- return-coordinate surface 68
- reverse lens elements 32
- revision level 214
- rings and spokes 242, 344
- run OSLO 19

S

- sag 219
- sagittal focal shift in the field 430
- Save lens 161
- SCP
 - command arguments 536
 - directory 155
 - error handling 550
 - file format 535
 - introduction 534
- scratch & dig specifications 234
- secondary color 257
- Selfoc™ gradient 99
- set preference 174
- show preference 174
- simulated annealing 383
- sliders
 - assigning and using 567
- SmartCells 494
- solid model lens drawing 227
- solve in alternate configuration 214
- solves
 - aplanatic 41, 42
 - axial ray angle 41
 - axial ray height 46
 - chief ray angle 42
 - chief ray height 46
 - edge thickness 47
 - thickness 46
- solving equations 589
- sorting data 589
- source analysis
 - edit extended 434
 - pixelated 446
 - view extended 443
- source astigmatism 215
- special actions in non-sequential groups 109
- special apertures 50
- special surface data 61
- spherical aberration
 - oblique 261
 - plotting as longitudinal aberration 279
- spherical gradient 103
- spline fits to data 608
- spline surface 82, 83
- spot diagram
 - extended 247
 - set 352
- spot size
 - computing geometric 292
 - diffraction limit 293
 - finding minimum geometric 292
- spreadsheet
 - configuration 22
 - cut/copy/paste 30, 31
 - element drawing 232
 - enter test glass radius 45

- field point set 350
- insert rows 32
- lens drawing 224
- lens drawing operating conditions 225, 238
- operands 353
- ray set 351
- spot diagram set 352
- surface data 28, 32, 37, 40
- update surface data 18
- variables 125, 367
- spreadsheet buffer 138, 475, 533
 - control of 533
- standard asphere 74
- start/stop skip 62
- statistical sum 422, 426
- Strehl ratio 266, 300
- string functions 591
- surface data spreadsheet 32, 37, 40
- surface data tags 201
- surface note 61, 532
- surface number
 - printing in spreadsheet buffer for ray trace 270
- surface sag 110, 111, 219
- surface type 527
- symbols 602
- symmetric asphere 71
- symmetric CGH surface 86, 89
- symmetric general asphere 74
- syntax of commands 491
- system note 532

T

- tabular lens array 64, 65
- tangential focal shift in the field 430
- telecentric pupil mode 215
- test glass list 45
- text editor
 - execute selection command 482
 - executing commands from 493
- text editor window commands 567
- Text output 170
- text output commands 582
- text window 475
 - close 477
 - commands 573
 - copying contents to clipboard 478
 - current 137, 475
 - new 476
 - open 476
 - output control 138, 476
 - output format 140, 476
 - reset 477
 - saving contents to file 478
 - title 477
- thermal coefficient of expansion 204
- thickness 46
- thickness default bounds 367
- thickness solve 46
- thickness tolerance 205
- thickness type 526
- tilt and bend surface 68
- tilt tolerance 205
- time
 - commands related to 614

- tolerance data 205
- tolerance units 424
- tolerances
 - calculation of surface irregularity 401
 - component decentration 406
 - component tilt 406
 - conversion between fringes and radius difference 400
 - definition of component 206, 404
 - reference fringe wavelength 400
 - refractive index 402
 - surface decentration 403
 - surface separation perturbations 401
 - surface tilt 404
- tolerancing
 - change table 424
 - compensators 417, 422, 425
 - MTF/wavefront 416
 - output threshold 425
 - user-defined 421
- toric surface type 528
- toroidal surface 72
- total internal reflection 24, 63
- track length 250
- transverse distortion 429, 430
- triangular aperture 50
- trigonometric functions 590
- typographic conventions 2

U

- units 214
- update glass catalog 184
- update surface data 18
- User
 - Check Vignetting 389
 - Compile CCL 466
 - Set Vignetting Factors 390
- user gradient 105
- user-defined extended source 439
- user-defined gradient 554
 - DLL 564
 - global variables 554
- user-defined operands 559
- user-defined pickup 44
- user-defined surface 110, 551
 - DLL 563
 - global variables 552
- user-defined tolerancing 421
 - computing inverse sensitivities 423
 - computing sensitivities 422
 - output format 422
- using commands 491

V

- variables 207
 - boundary conditions 373, 388
 - curvature 45
 - damping 367, 373, 388
 - derivative increment 373
 - derivative increments 367, 373, 379
 - entering 367
 - fixing glass 60
 - glass 60
 - gradient index 371
 - model glass 55

- multiconfiguration 368
- spreadsheet 125
- table of types 368
- thickness 48
- thickness default bounds 367
- vary-all options 367
- view extended sources 443
- viewing direction 239
- viewing matrix 220
- viewport
 - graphics window 603
- vignetted pupil 351
- vignetting
 - in optimization 390
 - pupil footprint plot 391
- vignetting analysis 389
- vignetting factors 390
- v-number 204
- volume
 - lens 220

W

- Warren Smith library 154
- wave plate 108
- wavefront statistics 265, 300
- wavelength weights 203, 205
- wavelengths 203, 205
- weight
 - lens 220

- wide-angle ray aiming 215
- win.ini file 153
- Window
 - Arrange Icons 484
 - Graphics 127
 - Text 475
- window
 - text 475
- window management commands 567
- windows
 - arranging minimized 484
- Windows metafiles (*.wmf) 172
- wire frame lens drawing 227
- Wood lens 99
- working f-number 249

Y

- y-ybar diagram 394

Z

- ZEMAX™ files 161
- Zernike analysis of wavefront 267, 301
- Zernike phase surface 91
- Zernike polynomials 80, 81, 91, 94
- Zernike sag surface 80
- zones
 - diffractive 221
- zoom data 22
- zoom variables 368