Rfringe Installation and User Guide

Michael Peck <mpeck1@ix.netcom.com>

October 21, 2004

Acknowledgements

Thanks to the following individuals: Vladimir Galogaza for diligent alpha testing. Steve Koehler for valuable programming ideas, especially his approach to object instantiation in R. James Lerch for permission to use the results of his interferometry.

The Rfringe GUI is based in part and uses code fragments from the R package `Rcmdr` by John Fox (http://socserv.mcmaster.ca/jfox/). Permission to use his work is greatly appreciated.

1 Introduction

Rfringe is an R [3] package for the analysis of interferograms. Its features include:

- User assisted semi-automatic fringe tracing.
- Wavefront estimation by least squares fits to Zernike polynomials, as described in Malacara, ch. 13 [2]. Instead of the numerically unstable algorithm described by Malacara Rfringe uses the modern, proven linear model solver provided as part of the base R system.
- Although it might be a bad idea to do so, in principle arbitrary sets of Zernike polynomials can be used in the fit.
- In addition to the usual diagnostic plots, there are a few – possibly even useful – surprises.
- Printable reports are prepared in Adobe Acrobat format.
- A simple project manager is provided to analyze multiple interferograms as a group.
- Images that are rotated relative to an arbitray 0 point are handled correctly, with the appropriate user input of rotation angle.
- A simple but effective GUI is provided using the R Tcl/Tk library. Power users can also access all data and functions from the R command line.
- Rfringe is open source, free, and licensed under the GPL.
- Rfringe is portable. It's known to run on Linux and current versions of Microsoft Windows, and *should* work on most Unix/Linux platforms and possibly on Mac OSX.
- Rfringe is completely customizable. All components of the package, including menu and report templates, help files, and the Rfringe code are in ordinary text files. Users may modify these in any way they please, and the Gnu Public License gives you explicit right to do so.

Rfringe does have some limitations. They include:

- Some critical functions, especially fringe tracing, are slow.
- It's memory hungry.
- The package relies on a suite of tools. Most of them are standard in a Unix/Linux environment, but are probably unheard of to most Windows users. All of the tools *do* have free Windows ports available.
- Only circular apertures can be analyzed, although the input image need not have a 1:1 aspect ratio. Also, the use of Zernike polynomials assumes that the entire aperture is used. The results of analyses could be misleading for perforated or obstructed optics.
- There is no *direct* support for any kind of image capture hardware. You must be able to acquire images and store them in some suitable graphics format yourself. On the other hand, Rfringe makes use of ImageMagick's `convert` tool to enable the use of a wide variety of graphics formats.

1.1 Operating system and hardware requirements

Rfringe is known to work on the following operating systems:

- Linux (tested on Red Hat 7.3 and Fedora Core 2),
- Microsoft Windows® XP and 2000.

Most Unix/Linux systems should be able to run Rfringe, although see the detailed software requirements in the next section. The R system also runs on Mac OSX, however Rfringe has not been tested in that environment (reports of successful [or unsuccessful!] attempts to run it are welcome). In the rest of this manual I will refer to Linux only, since that is the environment in which Rfringe has been tested.

For hardware, the author recommends a P4 class processor running at 1.5 GHz or better, and at least 512 MB of memory for **comfortable** use. A full color graphics display with resolution 1280 x 1024 or better is recommended.

This document is intended to provide a chatty, "user-friendly" guide to installation and use of the Rfringe GUI. The formal R style documentation in Rfringe.pdf documents the functions used in the package.

1.2 Software components

A number of software components are required for full functionality of the Rfringe package. They include:

- The R [3] language and environment for statistical computing. The main R site on the web is http://www.r-project.org/. The master download site for R is CRAN, at http://cran.r-project.org/. There are CRAN mirrors worldwide. The simplest way to locate a nearby one is to click on the CRAN link from the main R site.

  Please note that R is a complete environment, part of which is the R language interpreter. When you work in Rfringe you are working in R. R itself is normally used at the command line, but Rfringe provides a simple GUI to spare you the pain of learning the underlying language. Some minimal command line interaction is required to start Rfringe and exit R.

- Rfringe (this package), is supplied in three forms: as a gzipped source tarball suitable for Linux installation, as a zipped set of source files, and as a zipped 'binary' package suitable for installation on Windows. Except for the two Adobe Acrobat files included for documentation all files in the 'binary' package are ordinary text files.

- Additional R packages available from CRAN. At this time the only package required by Rfringe that is **not** part of the standard R distribution is the `pixmap` library, which is used to import graphics files in "Portable Anymap" format. This package is **required**, and should be downloaded and installed prior to using Rfringe.

- The graphics library rgl is required for display of interactive 3d wavefront maps. `RGL` is a collection of OpenGL graphics routines for the R language. The package is optional, but fun and informative.

- Tcl/Tk provides the GUI toolkit for Rfringe. The required library files are included with Windows binary distributions of R. Just be sure to include them when you install the R system. Most Linux systems should have the required libraries installed, but if not Tcl/Tk is available at http://www.tcl.tk/.

- ImageMagick's `convert` tool is used to convert from a wide range of graphics formats to the `pnm` format required by the `pixmap` library. Strictly speaking this is optional, but Rfringe expects to find it if graphics conversions are necessary when loading interferograms. ImageMagick is likely to be installed on your Linux system, but if not it can be downloaded at http://www.imagemagick.org/.

- Finally, Rfringe uses `pdflatex` to prepare printer ready reports in Adobe Acrobat format. Again, your Linux system is likely to have the program and all necessary supporting packages already installed. If not, CTAN is the official download site for all TEX related software.

  For Windows installations the best option appears to be MiKTeX, which is available at http://www.miktex.org/.

  `pdflatex` is assumed to be present and is required for conversion of documents from LATEX format to Adobe Acrobat. If you prefer another final format for printer ready reports you should edit the Rfringe source to your preferences. If you are a Windows user you probably want to leave the source alone and acquire the appropriate document processing software.

2 Installation

Installation of the tools discussed above in section 1.2 is probably going to be simpler in a Linux environment, if only because some of them are likely already to be installed on your system. I'm going to assume you have the necessary support tools (ImageMagick, LATEX, gcc, etc.) installed or know how to do it, and just discuss R related installation.

2.1 Linux installation

Proceeding in order:

1. Install R. There are two practical ways to do this: either build from sources or install from an RPM. CRAN provides RPM's for several popular Linux distributions. If that option is too easy building from sources is straightforward provided you have the correct development tools and support libraries installed. The definitive resource for R installation is the "R Installation and Administration" manual, which is available in PDF format from the main R site – just click on the Manuals link to locate it. You should also read the general and system specific FAQs.

2. Install the `pixmap` and Rfringe libraries. `pixmap` is available from CRAN. At present Rfringe is available at my home page in the directory http://home.netcom.com/~mpeck1/astro/rfringe/. The file you want is 'Rfringe_ver-no.tar.gz'. Both packages can be installed with the command `R CMD INSTALL` *`filename`* where *filename* is the name of the downloaded package file.

3. Install the `rgl` library. Download the source files from http://wsopuppenkiste.wiso.uni-goettingen.de/~dadler/rgl/ or from CRAN. I recommend getting the most recent available version (0.64-13 at this writing). After unpacking them consult the README for installation instructions. On the author's machine a vanilla installation worked.

2.2 Windows installation

All of the software components discussed in section 1.2 are open source, and given the right development tools it is certainly possible to build from source in Windows. I will discuss installation of binaries only – if you know how to build software from sources you know more than I do.

1. Installing R and R packages. The latest version of the R binary installer can be downloaded from CRAN by following the links to the Windows binary. Unlike some software R does **not** tend to get less stable at major new releases, so get the latest version (currently 1.8.1). Accepting default installation options should work for most users – be sure to install the Tcl/Tk support libraries.

   Once R is installed and running (see section 3) there is a menu item to install packages either from CRAN or local zip files. You may use either option to install the `pixmap` library. If you download it separately be sure to get the "precompiled" binary distribution from CRAN.

   Rfringe has not yet been submitted to CRAN. The "precompiled" ready to install version is the file Rfringe.zip.

2. Installing RGL. Download the latest available precompiled binary version. Versions prior to 0.64-6 failed to install properly using the R installation routines, but this appears to have been corrected in the most recent versions.

3. Installing ImageMagick. Follow the links to the download site and get the appropriate binary. The installer will do the rest of the work for you.

   One caution about ImageMagick. The only tool used directly by Rfringe is the `convert` program, which happens to be the name of a Windows system program. To work properly ImageMagick must be on the search path before the Windows system directories. The ImageMagick installer will do this for you – just remember not to edit it out of your PATH environment variable.

If you have an alternative way to get files into "Portable Anymap" format you need not use ImageMagick, but you are responsible for performing conversions outside Rfringe.

4. Installing MiKTeX. I've saved the best for last. As usual, follow the links to the download site. The first thing you will download is a relatively small setup "wizard." When you run that you will, after clicking through a few steps, be directed to a download mirror to complete the installation. Select one nearby, if possible. When I installed MiKTeX this step took several tries until I finally found a mirror in Canada that would complete the download without timing out. A small download set is almost sufficient for use by Rfringe.

   Once MiKTeX is installed on your machine you need to retrieve one more package named 'fancyvrb'. To do that run the MiKTeX package manager, which you should now find on your Windows Start|MiKTeX menu. This program will download a rather large database of available TEX packages. Once it's loaded find fancyvrb and have the package manager install it.

   This is still not quite sufficient. You need one more file from your R distribution named 'Sweave.sty'. On a default R installation that will be located in the directory C:\Program Files\R\rw1081\share\texmf. You need to copy that file and then paste it into a location that MiKTeX can find. On my machine a convenient location was the directory C:\texmf\tex\latex\fancyvrb.

   Finally, MiKTeX needs to know where to find the file you just copied. To update its database run a program labelled MiKTeX options from the Windows Start menu. A tabbed window will open. Select the first tab labelled General and click on the button Refresh Now in the item File name database.

   That's it! You can now produce beautifully typeset reports in officially sanctioned R document style format. If you don't like the looks of that you can learn LATEX and write your own style files.

3 Using Rfringe

To use Rfringe you must first start R. In Linux the easiest way to start R is to first open up an X terminal window, cd to the directory containing your data, and type R at the shell prompt. A short introduction will scroll by and you will be greeted with the R prompt, which by default is >.

In Windows I like to keep desktop shortcuts pointing to the directories containing any projects I happen to be working on at the time. You should let the R installer create a desktop shortcut for you, then copy and edit its properties as needed.

The R for Windows "GUI" can run as either an sdi or mdi application. For Rfringe it needs to run as sdi, which is not the default. To make R run as an sdi application add the command line argument `--sdi` **after** the final `"` in the target field. Once you start R its behavior is basically similar to a Linux environment, except that you interact with it in a console window that also contains a limited set of menu choices and taskbar items.

Once R is up and running type in the following commands at the prompt:

```
> library(Rfringe)
> Rfringe()

<Tcl>
```

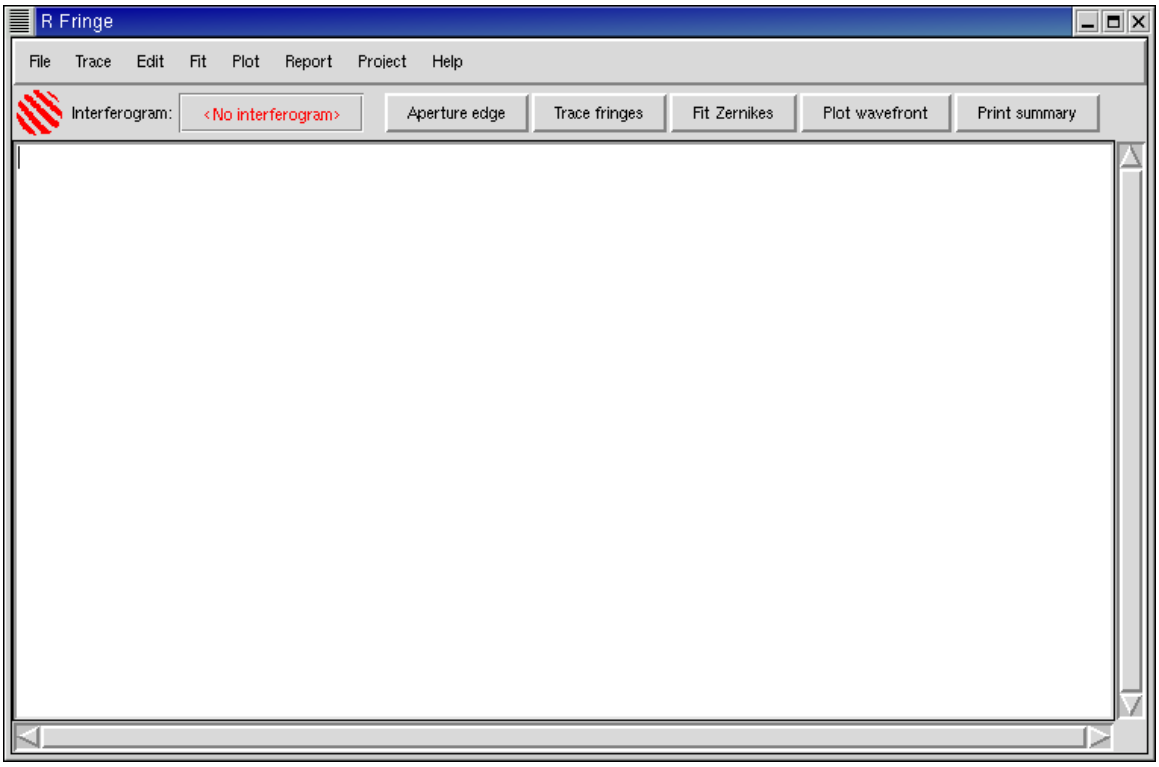

Figure 1: The Rfringe Main Window

Some additional messages may scroll by and then, if everything was installed properly, the main Rfringe window will pop up, which at start up looks like figure 1. If you wish you may now minimize the R console window (don't close it!) since all interaction and output other than R errors will be sent to the Rfringe GUI. At the moment the only other R command you really must know is how to get out, which is `q()`. When prompted it's a good idea to type `y` if you want to save your work.

The Rfringe main window presents a simple, no-frills GUI. There is a menu bar at the top, a row of buttons just below it for single click access to common tasks, and a text window below that which is used both for prompts to the user and typed output. The ordering of menu items is intended to reflect typical work flow, at least for people who normally read from left to right. You first get some data into your workspace, you process it in various ways, you tell Rfringe things it needs to know to perform analyses, you do the least squares fit to the data, you examine the resulting analysis, then you prepare a report for your client. If you want additional precision the best way to get it is to combine data from multiple interferograms, and Rfringe offers project management functions for that purpose.

4 Rfringe menu choices

The remainder of this document will go through the Rfringe menu an item at a time. I will say little about interferometry and the analysis of interferograms. Again, Malacara, ch. 13 [2]

is the standard reference on fringe analysis, although some of the algorithmic details presented there are obsolete.

4.1 File

Rfringe works on two kinds of objects: interferograms and interferometry projects. The File menu items are used to create interferogram and project objects and for basic workspace management. R stores its objects – which include functions and data – in a workspace in memory during an R session, optionally saving it to a file when you exit R. The objects Rfringe creates can be rather large, especially interferogram objects. If you work with a large number of interferograms in a session you can easily fill up hundreds of megabytes of your workspace, and you may need to do some manual memory management to keep R running smoothly. The File menu is the place to look for memory management commands.

The companion manual Rfringe.pdf contains more details on the contents of interferogram and project objects. The R functions `interferogram` and `project` provide the data structures and functions for manipulating these objects.

4.1.1 Load interferogram from image...

This command loads an image from disk and creates a new interferogram object. The dialog box contains three entries: the first prompts you for the name for the interferogram and is mandatory. The other two fields labelled Tester and Test date are optional. The test date entry will be filled with the current system date and time. These last two fields are just used for labelling reports, and you may put anything you like in them.

The first field is actually used for three different purposes, and it will help to know what it does. First, and most trivially, it is used as an image id label in plots and reports. Second, Rfringe uses this as the name of the R object that this command is creating. The way it does that is to create a syntactically correct R name from the string you enter, which becomes the name of the interferogram object. You don't need to use a syntactically correct name for the image id, but it may help not to stray too far from R naming conventions. Third, if you decide to save your interferogram object to a data file, Rfringe will use the object name as the base name of the file, append the extension '.rdat' to it, and save it in the current directory (see subsection 4.1.3). As an example, suppose you name your interferogram `foo bar`. Rfringe will create an object having the name `"foo.bar"`. If you save it from within the Rfringe GUI a file named "foo.bar.rdat" will be saved with the contents of the object. If you have a number of images to analyze you might assign them id's like `foo bar 01`, `foo bar 02`, etc., which will produce objects named `"foo.bar.01"`, `"foo.bar.02"`, etc. R requires object names to start with a letter, and underscores (among other symbols) cannot be used in variable names; `01_foo bar` becomes the object `"X01.foo.bar"`.

Once you press OK in the dialog a file selection dialog will pop up, from which you select the interferogram image you want to work on. As hinted previously, Rfringe can only work directly on images in "Portable Anymap" format. This is a limitation of R, or more correctly the `pixmap` library for importing images. To get around this Rfringe uses ImageMagick's `convert` program to convert files, if necessary. The file selection dialog box will look for jpeg and TIFF file extensions, but those are not by any means the only graphics formats supported. See the ImageMagick documentation for the full list of supported graphics types.

The other limitation of Rfringe image processing is that it only works on grayscale images. If it is given an RGB file as input it will convert it to grayscale using the R channel only. This is done because most interferometers use a He-Ne laser source with a wavelength of 632.8 nm, so

there's likely to be useful data in the R channel only. If that is not the case with your equipment you should do the conversion outside Rfringe.

4.1.2 Open interferogram

In the Rfringe GUI you can work on at most one interferogram at a time, and at most one project at a time. The two Open interferogram dialogs are used to select a previously saved interferogram object to work on, either from the R workspace or a file. You don't need to do anything special to save your current interferogram object. It will remain in your R workspace until you either remove it or exit R without saving.

4.1.3 Save current

Saves the current interferogram object to an R data file. See subsection 4.1.1 for details of how file names are created. A warning box will pop up if a file of the same name as the one about to be saved already exists in the current directory. You can choose to overwrite it or cancel.

4.1.4 Clean workspace...

The dialog box presents a list of interferogram objects in your workspace. Select one or more to be removed from memory. You will be warned if *any* selected object has not yet been saved.

4.1.5 Project

The Project menu commands perform the same functions on project objects as the higher level menu items described in the preceding subsections do for interferogram objects. There are four submenu choices under the Project menu.

1. Create...

   Creates a new project object. There are four entries in this dialog box, of which only the first is mandatory. As with interferogram objects (see subsection 4.1.1) the project name entry is used to create a name for the project object stored in the R workspace.

   When you create a new project it becomes the current project and any project management operations you perform will act on it.

2. Open.

   Opens a previously saved project, either from the workspace or an R data file. The selected object will become your current project. Again, you need not do anything to save your current project if you replace it. It will remain in your R workspace until you either remove it or exit R without saving.

3. Save.

   Saves the current project object to an R data file. A warning box will pop up if a file of the same name as the one about to be saved already exists in the current directory. You can choose to overwrite it or cancel.

4. Clean workspace...

   The dialog box presents a list of project objects in your workspace. Select one or more to be removed from memory. You will be warned if *any* selected object has not yet been saved.

Project objects store a tiny fraction of the data generated in an interferogram analysis, so in contrast to interferogram objects they require relatively little memory.

4.1.6 Exit

There are two choices under Exit. From Rfringe will close the Rfringe window and return you to R. From Rfringe and R will save your workspace and close R as well as Rfringe.

4.2 Trace

There are three items to be traced in an interferometric image: the edge of the aperture, the edge of the obstruction (or perforation) if any, and the fringes themselves. All of these require some degree of user interaction using the mouse or other pointing device.

In all of these routines a single press of the left mouse button will select a point. To exit a routine press the right mouse button. In Windows a small box will pop up with a prompt to either stop or continue. In Linux you'll just exit the routine without an opportunity to change your mind.

For all of these routines you want the interferogram image to be the top window, and if at all possible it should not overlap the Rfringe program window. Overlapping active windows force the operating system to do frequent redraws, which can be time consuming and annoying.

4.2.1 Trace aperture edge

As with other similar programs, Rfringe makes no use of any scale or other reference markers to define a coordinate system. Instead the edge of the aperture is used to define a polar coordinate system with origin at the estimated center. Since Zernike polynomials are defined on the unit circle the edge is defined to have a radius of 1, and all pixel values are scaled appropriately.

To outline the edge, just left click on a number of edge points. Rfringe requires that you select at least 5. Six to 12 points, distributed more or less evenly around the aperture, should be sufficient. They do not have to be picked in any particular order.

Rfringe does a least squares fit to the selected points to find the aperture center and radius (in pixels). It will test for an image with non-square aspect ratio, and adjust the x and y scales if they are significantly different. A line showing the estimated edge will be drawn as soon as you finish. If it doesn't look right you should start over now.

This command is also accessed with the taskbar button Aperture edge.

4.2.2 Trace obstruction

If the center of the aperture is obstructed (usually because the test was done in autocollimation with a perforated flat) you also need to outline the edge of the obstruction. Again, just left click on several points, and right click to get out and see the estimated edge overlaid on the image.

This version of Rfringe assumes the pupil is completely filled, so estimated wavefronts will be extrapolated to the center even if the image is obstructed.

4.2.3 Autotrace options...

This dialog sets parameters used by the fringe tracing function discussed in the next subsection. The default parameters work reasonably well in most situations, so you should adjust them only if you need to.

The fringe tracing routine is a three step algorithm. Starting from an initial selection it uses something similar to a photo manipulation program's "magic wand" selection tool to select candidate fringe points. Once a fringe has been fully filled out, a small neighborhood of each candidate point is searched for a local minimum of grayscale value. At this stage any duplicates are removed from the list of candidates. Finally, all but a fraction of previously selected points are discarded. The main reason for this final step is that the algorithm will naturally select many adjacent pixels. Since real images don't have 1 pixel resolution that means that adjacent points aren't independent, which violates a key assumption of standard least squares analysis. Experimentation with real images suggested to me that keeping every 3rd to 4th point provides an adequate approximation to independent measurements.

The user settable fringe tracing parameters are, in the order they're presented in the dialog box:

- A measure of gray scale tolerance. This sets a tolerance as a percentile of the range of grayscale values in localized regions of the image. Its value must be between 0 and 1, and should probably never exceed 0.5. If the fringe contrast is highly variable you may find that increasing it to – say – 0.35 will help fill out fringes with less user interaction.
- Search window size for local min. This controls the second stage of the fringe selection process. A larger value will make selected points cluster closer to fringe centers, but in regions with crowded fringes there is some risk of points "jumping" fringes.
- Trace to rho. This controls how close to the aperture edge a fringe will be traced. In general this should be close to 1, but it's often difficult to trace fringes all the way to the edge because of diffraction effects, actual poor edge quality, etc.
- Keep every. The entry means, keep every n'th selected point. A value of 3 or 4 is almost always reasonable. There is very little speed advantage to limiting the sample size in the least squares fit, so don't set this to a large value for that reason.

4.2.4 Autotrace fringes

Fringe tracing is reasonably straightforward, and with a bit of practice the user interaction required isn't too onerous, but there are some issues to keep in mind.

First, fringes must be traced in order. Rfringe has no independent way to tell how fringes are ordered, so it relies on you to do the ordering for it. In my experience the commonest user failure is simply failing to "tell" the routine that you're done tracing a fringe, and then tracing 2 or more fringes without incrementing the fringe order. Rfringe uses a palette of rainbow colors to mark selected fringe points (see figure 10), which will give you a visual cue that you are in fact tracing different fringes.

The fringe tracing routine requires one or more clicks per fringe. Make an initial selection anywhere within the fringe. It seems to make no difference where along a fringe you pick, and you don't have to be too careful about finding the exact center. If all goes well the fringe will, in due course, fill with points. This process can take anywhere from a few to quite a few seconds, depending on the speed of your hardware, the length of the fringe, the size of the image, etc. Don't be alarmed if the entire width of the fringe is filled with points. That's a feature. If the entire length isn't filled, just left click again in the untraced part of the fringe. Fringes that cross obstructions will always need at least two clicks. Once you're done with a fringe, right click. Rfringe does some post-processing on the selected points as described above in subsection 4.2.3, which may take a few more seconds. It will then increment the fringe order and prompt you to

select the next fringe. A second consecutive right click will tell the routine that you are done with fringe tracing.

This version of Rfringe will only trace dark fringes.

There is no way to edit fringes during the initial fringe tracing process, but there is a suite of fringe editing tools (see subsection 4.3.4). If a fringe doesn't look quite right, you inadvertently trace 2 fringes as one, or miss a segment you can fix it later. For larger scale problems it's probably best to break out of the fringe trace, adjust options if necessary (4.2.3), and start over. If you get the sense of the tilt wrong and trace in reverse order you can just change the sign of the fringe scale in the Analysis info dialog box (subsection 4.3.2).

Once you've finished tracing the interferogram you can redisplay it with the selected points by pressing the taskbar button with the interferogram name, or with the Plot|Replot fringe trace menu item (subsection 4.5.8).

4.3 Edit

The Edit menu dialogs are used to enter information about the interferogram that's needed for its analysis, and to edit traced fringes.

4.3.1 Basic image info...

This dialog contains entries for the same three fields you entered when you created the interferogram object, and in addition entries for the test (source) wavelength and image orientation. Changing the Image ID field has no effect on the name of the current interferogram object. The default wavelength is 632.8 nm.

If you analyze a test piece in more than one orientation you specify its orientation in the fourth entry. That specifies the rotation angle as seen in the image in degrees, measured counterclockwise from whatever you've chosen as 0°. If all images in your project were taken with the test piece in the same orientation leave this at 0. Plot and analysis routines will counter-rotate images by the angle specified here so all images in a project should display with the same apparent orientation.

4.3.2 Analysis info...

The entries in this dialog are ones specifically needed for the least squares and summary analyses. There are entry boxes for the evaluation wavelength (in nm) and fringe scale in waves. The default values are 632.8 nm and 0.5 wave for fringe scale, which is appropriate for double pass tests. Since 632.8 nm is well to the red of the peak sensitivity of the human eye it may be appropriate to change the evaluation wavelength to 500 or 550 nm. Summary quality measurements will then be scaled to the evaluation wavelength.

Finally, there are checkboxes for aberrations to be removed from the analysis, by which is meant their corresponding Zernike coefficients are set to 0. Cancel defocus is checked by default. You can also Cancel astigmatism and Cancel coma, both of which are unchecked by default. Rfringe will not allow you to cancel higher order aberrations – at least without learning the command line interface.

4.3.3 Target conic...

In general it really only makes sense to use this dialog if you are performing a single pass test on an asphere at the center of curvature without nulling optics. If you are performing a null test

of any sort you should leave this alone, regardless of the intended shape of the final product. There may be other non-null testing situations where it would be appropriate to enter a target conic, but you are responsible for working out the math.

Entries are provided for the diameter, radius of curvature, f/ratio, and conic constant. You only need to enter a value for one of radius of curvature or f/ratio. If you enter values for both radius of curvature takes precedence. Linear dimensions must be in millimeters.

If you mistakenly enter data for a target conic you can make Rfringe forget about it by entering 0 for the conic constant.

4.3.4 Fringe...

This dialog is for editing previously traced fringes. There is a "spinbox" entry to select a fringe to edit. The choices for editing – which are selected with radio buttons – are:

- Add points. Manually add points to the selected fringe.
- Clear points. Manually clear points from the selected fringe. This will replot the interferogram with only the selected fringe displayed. You do *not* want to make wholesale deletions from a fringe this way. If a large part of a fringe has gone astray it's better to retrace it or clear it and manually add new points.
- Add segment. Adds a segment to a fringe using the auto trace routine.
- Retrace fringe. Clears the currently selected points from a fringe and retraces using the auto trace routine.
- Insert fringe. Inserts a new fringe *before* the selected fringe. This actually adds no points to the fringe trace. All it does is renumber the higher order fringes, starting with the fringe order in the text entry. To actually add new points you want to select add points or add segment. To add a new fringe after the last currently traced fringe just enter a number one higher than the highest numbered fringe in the Fringe to edit entry, and then select one of the add points options. You can also add a fringe 0, or even a fringe -1, -2, etc. simply by typing the number into the Fringe to edit entry. The actual numbers assigned to fringes make no difference, as long as they are separated by 1 and in the proper order.
- Clear fringe. Clears all the points from the selected fringe.

In order to make fringe editing a little less painful than it might otherwise be this dialog behaves a little differently than other Rfringe dialogs. Because of the limitations of R's routines for interacting with graphics you can only do one thing at a time to a fringe. When you click on the Ok button the interferogram will usually redraw with a visual cue to indicate which fringe you are working on. You perform whatever task you selected, then right click to exit. The dialog box will remain open, and you may select another fringe editing task. When you are finished with fringe editing, click on Done. Warning: this program has occasional issues with window focus. This dialog in particular has a tendency to disappear behind other windows, especially in a MS Windows environment. If it gets lost just drag other windows (most likely the main Rfringe window) around until you find it.

A final hint: Autotrace options (subsection 4.2.3) also apply to the fringe editing routines that make use of the auto trace routines. There may be situations where it makes sense to use one set of parameters for the bulk of fringe tracing, and another set for fringe editing. As an example, fringes that lie close to aperture edges are often difficult to trace well using default parameters.

4.4 Fit

These two menu items set the one configurable least squares analysis option and perform the least squares fit.

4.4.1 Max order...

This dialog is used to set the maximum radial polynomial order of the Zernike polynomials used in the least squares fit to the wavefront. In principle you can fit arbitrarily high order polynomials to the data. Zernike polynomials are computed using a recurrence relation for radial Zernikes [7]. The algorithm is numerically accurate up to as high order as has been tested.

In practice it's usually a bad idea to fit too high order polynomials to the data, and in most cases the default "Fringe" [4] set should be sufficient. If you experiment with different values of the maximum polynomial order and decide to go back to the default, enter the value `NA` in the dialog box entry.

4.4.2 Go

Performs the least squares fit. A message will be sent to the text window with a count of the number of Zernike polynomials used in the fit (which will be 37 for the default Fringe set). This command is also accessed with the taskbar button Fit Zernikes.

You may need to redo the least squares fit at times during the course of an interferogram analysis. In particular if you edit the fringe ordering or fringe scale (subsection 4.3.2), the image orientation (subsection 4.3.1), fringe data (subsection 4.3.4), or change the polynomial order of the fit the least squares analysis must be redone.

You do not need to redo the fit if you change the evaluation wavelength (subsection 4.3.2) or add a target conic (subsection 4.3.3), however summary statistics and wavefront evaluations will change so wavefront plots and reports will need to be redone (see subsections 4.5 and 4.6).

4.5 Plot

The Plot routines produce a variety of graphs useful for diagnosing the quality of the fringe analysis and the shape of the wavefront.

4.5.1 Synthetic interferogram

Computes a synthetic interferogram from the raw wavefront estimated by the least squares fit, and displays a gray scale rendering. An example is shown in figure 2, which resulted from a fit to the interferogram fringe trace shown in figure 10.

This is the most basic diagnostic tool for judging how well your fringe trace and least squares fit worked, and is probably the first thing you should look at after doing the fit. If all went well the synthetic interferogram should closely resemble the real thing. If they *don't* closely resemble each other you can reasonably infer that either the fringe trace went wrong or the least squares fit over- or underfit the data.

4.5.2 Wavefront

A wavefront map, with overlaid contours (figure 3). This is the *net* wavefront, with any excluded aberrations (4.3.2) and target conic (4.3.3) removed, calculated at the evaluation wavelength.

Figure 2: Synthetic interferogram

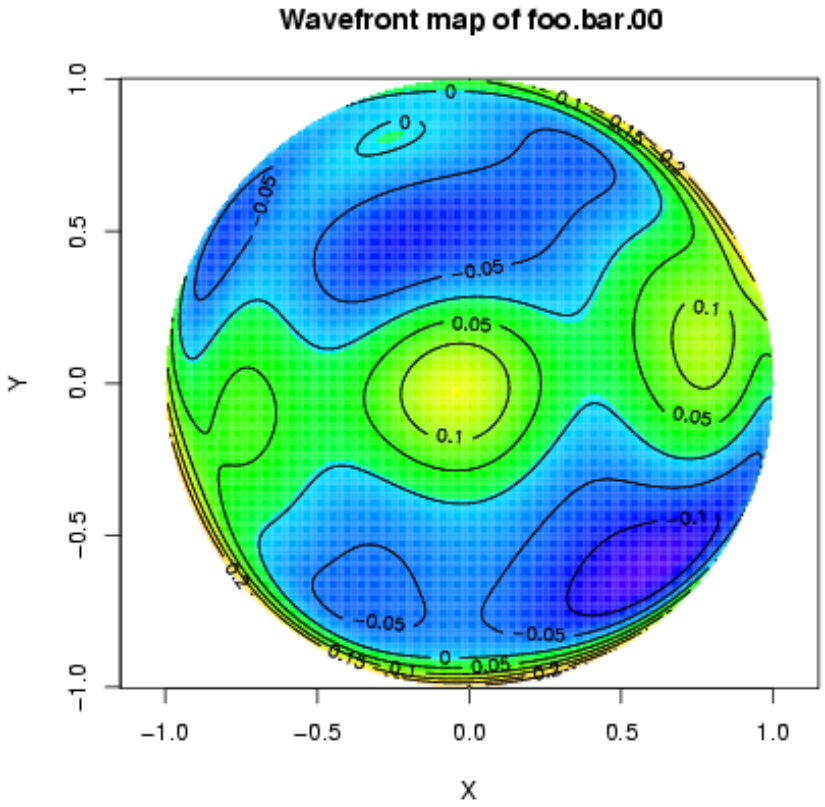

Figure 3: Wavefront

This command is also accessed with the taskbar button Plot wavefront.

The P-V wavefront error is estimated from the calculated wavefront. If you display one of the Report (section 4.6) summaries it will show a P-V of 0 until you've calculated a wavefront (the other summary statistics are calculated from the Zernike coefficients). Also, the two 3d graphs (figures 5 and 6) require that you display the 2d wavefront first.

4.5.3 Cross section...

Plots one or more cross sections along diameters of the pupil. The dialog box has an entry for azimuth angles, and "radio button" check boxes to select display of either wavefront error (in waves) or surface error (in nanometers).

Azimuth angles are in degrees, measured counterclockwise from horizontal. You may enter multiple values separated by commas. Figure 4 shows two cross sections through the wavefront of figure 3. Legends are displayed correctly in plots sent to the screen.

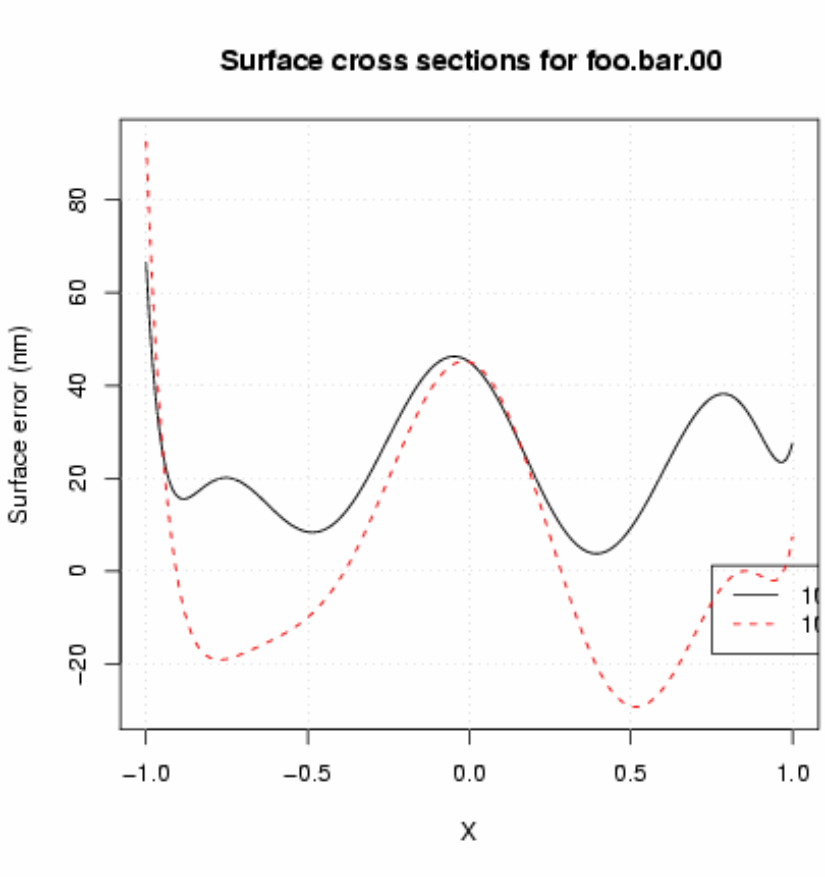

Figure 4: Surface cross sections

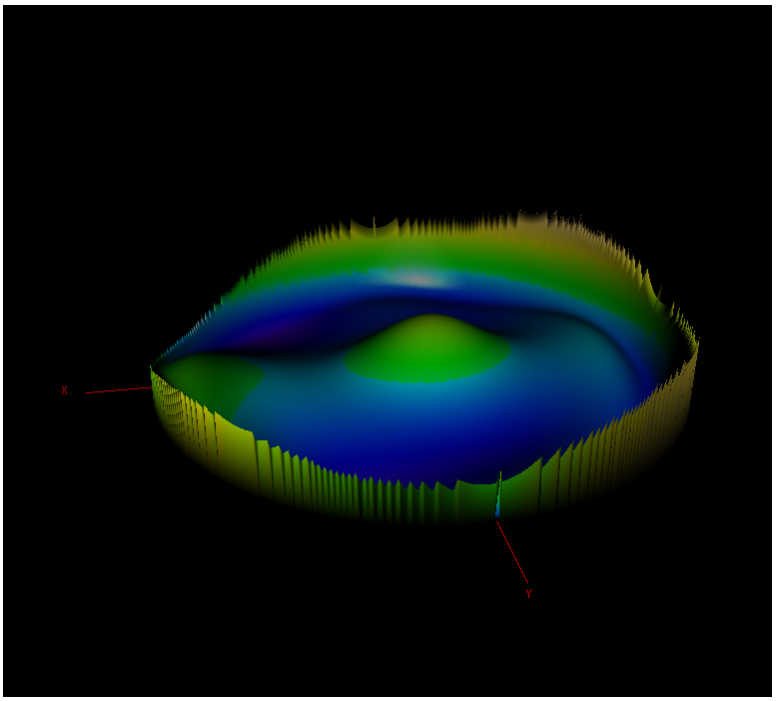

Figure 5: 3d Wavefront from rgl

4.5.4 3d Wavefront (rgl)

An interactive 3d wavefront map using the rgl library for OpenGL graphics. Please consult the `rgl` documentation for details of how to interact with 3d graphs. The static screen capture in figure 5 doesn't do justice to the displayed images.

4.5.5 3d Wavefront (persp)...

This is provided as a rather lame alternative to the rgl based 3d wavefront display. It uses the R base library function `persp` to display the wavefront. The dialog box provides sliders to change the viewpoint angle. Dragging the sliders will update the graph in almost real time, depending on how fast your graphics hardware is. See figure 6 for an example.

4.5.6 Residual analysis

A possibly useful display of some residual diagnostics from the least squares fit, as shown in figure 7. There are 4 panes in the plot. The top two are scatterplots of residuals (in waves) against fringe order and normalized zone radius. The bottom two are a histogram of the residuals and a normal quantile-quantile plot. A discussion of the meaning of the last plot is beyond the scope of this document. See the documentation for the function `qqnorm` in the R base library.

What you would like to see are (a) no systematic trends in the residual scatterplots, (b) a symmetrical histogram that looks approximately Gaussian in shape, and (c) points falling on a

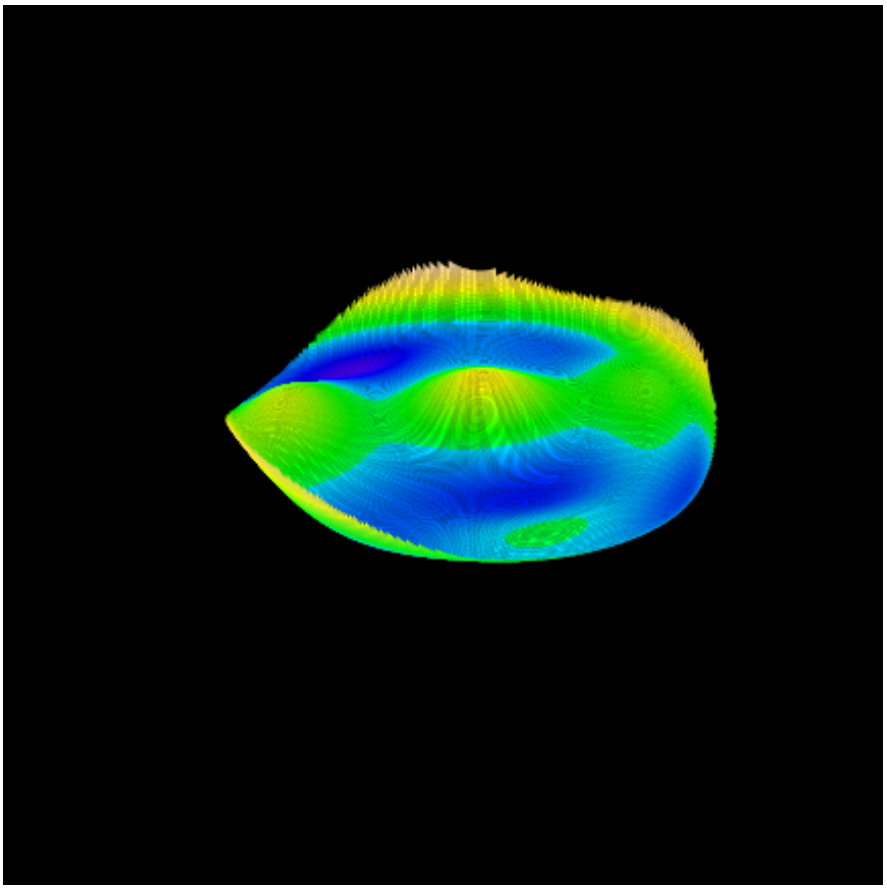

Figure 6: 3d Wavefront from persp()

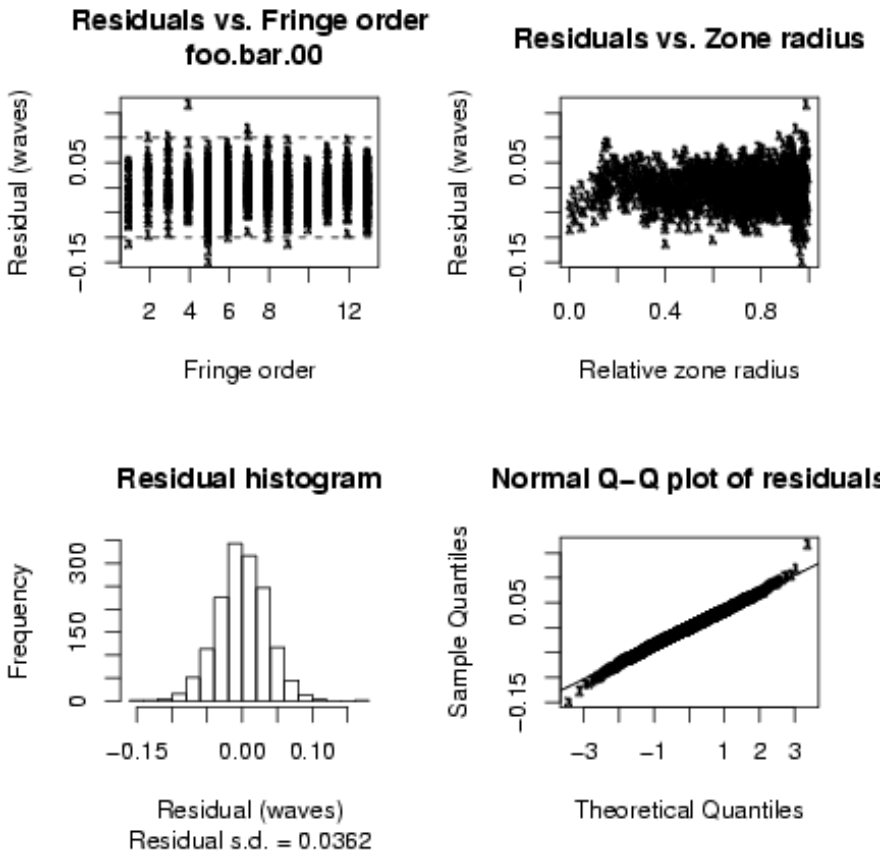

Figure 7: Residual diagnostics

straight line in the q-q plot. The sample plot 7 is fairly typical of a successful fit to real data. The edge of the aperture is not fit quite as well as the interior, and there are slightly more outliers than would be expected for a Gaussian error distribution. Using higher order polynomials in the fit might improve the residual diagnostics slightly, but it's questionable whether the real accuracy of the analysis would be improved.

4.5.7 Star test & mtf...

A simple star test simulator, with optional Modulation Transfer function calculation. The routine calculates the image of a monochrome point source filtered by the estimated wavefront in the approximation of Fraunhofer diffraction theory. See Born and Wolf [1], ch. 9 for details of the theory, and Suiter [5] for the application of the theory to testing small astronomical telescopes. The style of the display is modelled after Suiter [5].

The dialog box contains an entry for the obstruction size, a slider for the amount of defocus (in waves P-V), and a checkbox to select an MTF plot. The resulting displays look like figures 8 and 9, only much better on a large screen monitor. This example, by the way, is from the same wavefront displayed in figure 3, which is diffraction limited by standard criteria.

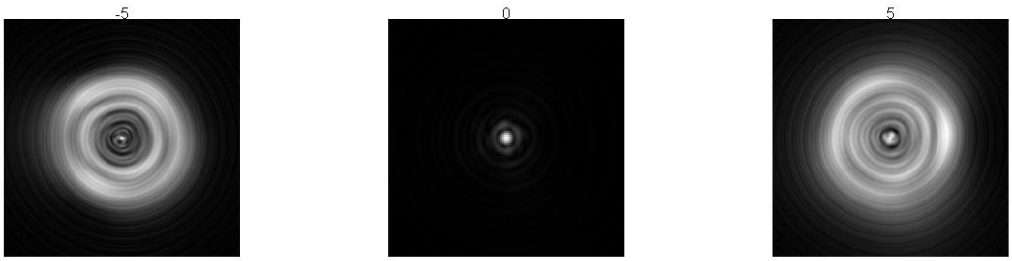

Figure 8: Simulated star test

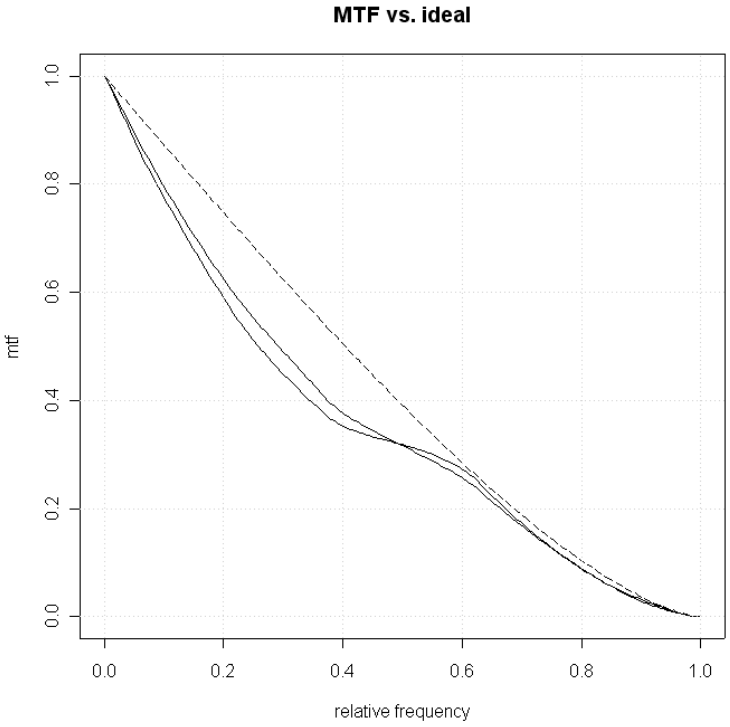

Figure 9: MTF plot

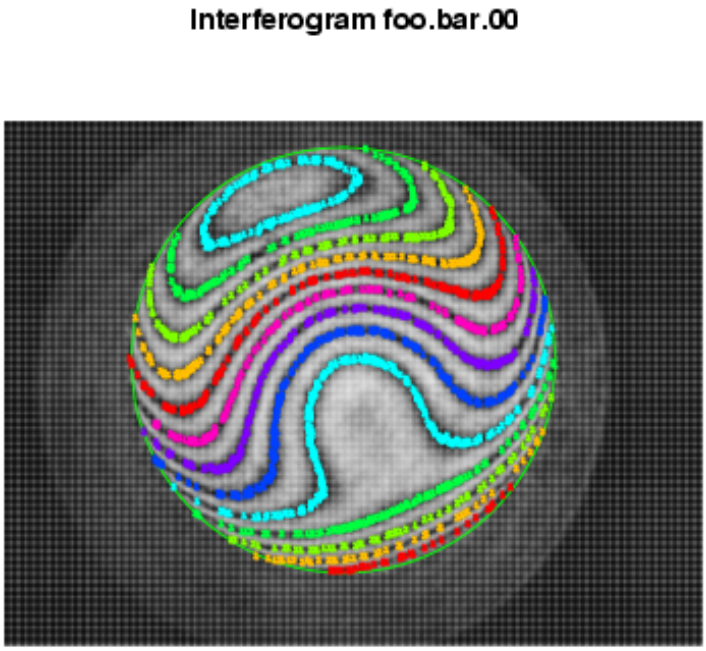

Figure 10: Fringe trace

4.5.8 Replot fringe trace

Redisplay the current interferogram with traced fringes overlaid (figure 10). Hint: clicking on the interferogram name on the task bar will display this plot.

4.6 Report

The Report menu is for display of summary numerical information, either sent to the Rfringe text window or prepared into a printable report in Adobe Acrobat format.

4.6.1 Summary

Prints basic summary statistics of the estimated wavefront to the Rfringe text window, in the format of the following example:

```
Summary results for foo.bar.00
```

```
Tester    A. Non
Test date Wed Oct 29 20:26:39 2003

Test wavelength       632.8
Evaluation wavelength 632.8

RMS          0.0644
P-V          0.429
Strehl ratio 0.849
Astigmatism  0.0314 Axis 27
Coma         0.0752 Axis 131  [removed]
SA           0.0326 P-V 0.109
Adjusted for target conic
Based on 1500 points
```

This command is also accessed with the taskbar button Print summary.

4.6.2 Details

Prints a detailed list of Zernike coefficients from the least squares fit, along with estimated standard errors and t-statistics. If you don't know what the latter two terms mean you should consult a textbook on applied statistics. If you do know what they mean I will comment that estimated uncertainties appear to be good predictors of the variation you might expect in repeated analyses of the same interferogram, but the variability *between* interferograms tends to be considerably larger than these estimates. If you really want to get a feel for the inherent precision of your interferograms you should take multiple images for each test piece.

Nomenclature for classical aberration labels is adopted from Wilson [6]. I use Zernike polynomials in normalized form, that is they are scaled to form an orthonormal basis set for the space of functions defined on the unit circle. This is not the most commonly adopted definition, as given for example in Born and Wolf [1], section 9.2. The reason I use normalized Zernikes is because, since each "axis" has the same scale estimated coefficients are directly comparable, making it possible to tell at a glance which aberrations are contributing to overall wavefront error.

The following sample output is from a single pass test on a small paraboloidal mirror. The coefficient values are from the estimated fit to the actual wavefront, in waves at the test wavelength, unadjusted for any target conic or removed aberrations (see the dialogs in sections 4.3.1, 4.3.2, 4.3.3). In this case the large value of Z8, the spherical aberration coefficient, is due to the fact that this is an asphere being tested at the center of curvature in a non-null test. After adjusting for the target conic (4.3.3) this mirror tests as diffraction limited.

```
Summary results for foo.bar.00

Tester    A. Non
Test date Wed Oct 29 20:26:39 2003

Test wavelength       632.8
Evaluation wavelength 632.8

RMS          0.0644
```

```
P-V          0.429
Strehl ratio 0.849
Astigmatism  0.0314 Axis 27
Coma         0.0752 Axis 131  [removed]
SA           0.0326 P-V 0.109
Adjusted for target conic
Based on 1500 points


Estimated Zernike coefficients and Standard errors
        Term      Coef. s.e.(Coef)  t value Classical aberration
 (Intercept)    7.59253    0.00105  7235.46 Piston
          Z1   -1.14056    0.00095 -1194.83 Tilt
          Z2    2.88193    0.00099  2911.97
          Z3    0.39549    0.00105   375.02 Defocus
          Z4    0.01820    0.00096    18.98 Astigmatism
          Z5    0.02558    0.00097    26.50
          Z6   -0.04977    0.00098   -50.64 Coma
          Z7    0.05633    0.00112    50.14
          Z8   -0.86143    0.00095  -911.24 Spherical 3rd
          Z9    0.00238    0.00097     2.47 Trefoil
         Z10    0.00911    0.00093     9.84
         Z11   -0.02295    0.00099   -23.18 Astigmatism 5th
         Z12    0.00274    0.00103     2.67
         Z13   -0.00238    0.00096    -2.48 Coma 5th
         Z14   -0.01510    0.00113   -13.39
         Z15   -0.00499    0.00101    -4.94 Spherical 5th
         Z16    0.01264    0.00095    13.24 Quadratic 7th
         Z17    0.01750    0.00095    18.33
         Z18   -0.00904    0.00099    -9.15 Triangular 7th
         Z19   -0.00114    0.00091    -1.25
         Z20   -0.00408    0.00103    -3.96 Astigmatism 7th
         Z21    0.00649    0.00105     6.15
         Z22    0.00325    0.00098     3.32 Coma 7th
         Z23   -0.00408    0.00111    -3.69
         Z24    0.01866    0.00100    18.70 Spherical 7th
         Z25    0.00149    0.00097     1.53 5-fold 9th
         Z26   -0.00338    0.00094    -3.61
         Z27   -0.01154    0.00093   -12.36 Quadratic 9th
         Z28   -0.00184    0.00094    -1.95
         Z29   -0.00866    0.00117    -7.41 Triangular 9th
         Z30    0.00252    0.00099     2.55
         Z31    0.00044    0.00105     0.42 Astigmatism 9th
         Z32    0.00108    0.00108     1.00
         Z33   -0.00302    0.00093    -3.24 Coma 9th
         Z34    0.00002    0.00103     0.02
         Z35    0.00087    0.00096     0.90 Spherical 9th
         Z36    0.00128    0.00099     1.30
```

```
Residual standard error     0.0362 on   1463 degrees of freedom
```

4.6.3 Printer friendly

Prepares a printer ready report, in Adobe Acrobat format. If `options(pdfviewer)` is set the report will be displayed in the selected pdf viewer. The general style of the report is, not coincidentally, similar to typical R documentation.

You should calculate a synthetic interferogram, wavefront, and any cross sections you are interested in prior to preparing a final report.

4.7 Project

These menu items are for project management and analysis. Remember you create or change projects under the File menu (subsection 4.1.5).

4.7.1 Add current interferogram

Add the current working interferogram object to the current project. If an interferogram with the same name is already in the project its data will be replaced without warning.

The data saved about each interferogram in a project object include the estimated RMS and P-V wavefront error, estimated Strehl ratio, and estimated Zernike coefficients. The estimated Zernikes are *net* values, adjusted for any removed aberrations or target conic, and calculated at the evaluation wavelength.

If you are performing a non-null test on an asphere you must specify the target conic for each individual interferogram. The project management functions do no further adjustment of estimated Zernike coefficients.

Although it's not absolutely mandatory it is a really good idea to be consistent in your analyses of individual interferograms belonging to the same project. In particular you should fit the same set of Zernikes to each one, although the project management function will try to adjust for changing sets of Zernikes. You should also remove the same aberrations from each interferogram.

The only analyses performed on projects are an unweighted average of Zernike coefficients and calculation of other simple summary statistics. These are updated each time you add or remove interferograms, so you can display plots or printed summaries of the current averaged wavefront at any time.

4.7.2 Batch add...

Add multiple interferograms from the workspace to the current project. The dialog will display a list of all interferogram objects found in your workspace. Select any you wish to add to the current project. Any interferograms with the same name as a selected interferogram object will be quietly replaced.

4.7.3 Remove interferograms...

The dialog box presents a list of interferograms in the current project. The data from any selected objects will be removed.

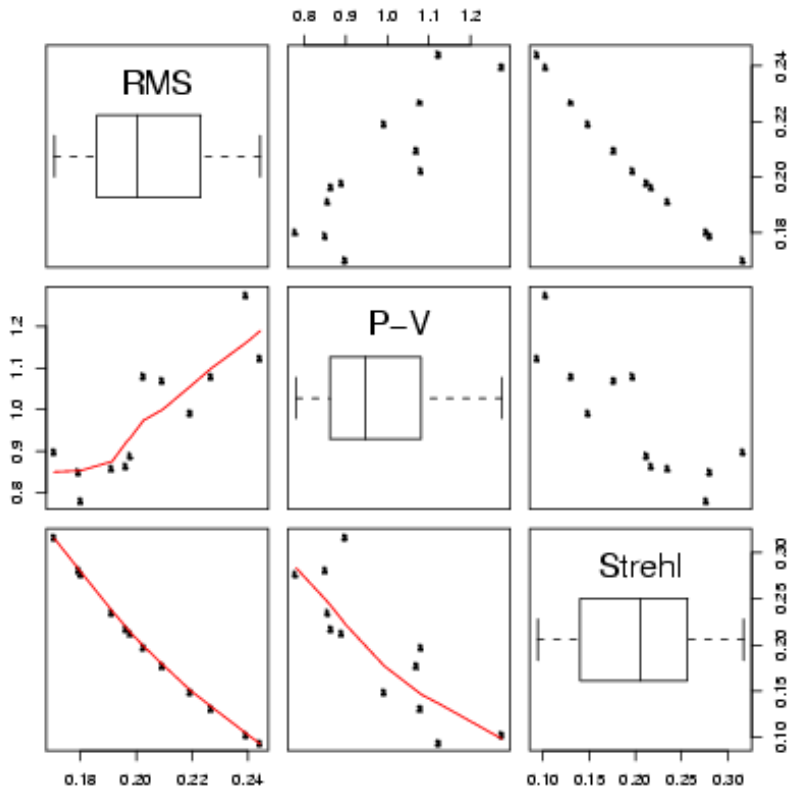

Figure 11: Interferogram comparisons

4.7.4 Plot

The project plot options are identical to the individual interferogram plots, with three deletions (no synthetic interferogram, no real interferogram with fringe trace, and no residual diagnostics) and one addition. All wavefront plots display the averaged wavefront, which is computed from the unweighted average of the Zernike coefficients from the individual interferograms.

The new plot is a comparison of the summary statistics RMS, P-V, and Strehl from the individual interferograms in the project, as shown in figure 11. This particular example compares a dozen interferograms of the same mirror. The plots along the diagonal are "box" plots, which give a useful summary of the distribution of random variables. The off-diagonal plots are pairwise scatter plots. Since there's an exact functional relationship between RMS and Strehl ratio there is no scatter in these plots. As one might expect there is a strong positive correlation in general between estimated RMS and P-V, but with considerable scatter.

4.7.5 Report

Project reports are nearly identical in format to individual interferogram reports, but there are some differences in content to beware of. First, the detailed display of Zernike coefficients gives

net values, with removed aberrations displayed as 0 and the spherical aberration terms for any target conic subtracted (provided you did it consistently in analysing individual interferograms). Second, the standard errors displayed in the detailed display are calculated from actual sample standard deviations of the input data rather than standard errors from least squares fits. If you have a reasonably large sample of interferograms in your project this should give a more realistic estimate of the precision of your data than you get from the individual least squares fits. The summary statistics are for the averaged wavefront, with sample standard deviations displayed alongside. It should be noted that the RMS of the mean wavefront is *not* the same as the average RMS (the same applies to the other displayed summary statistics).

```
Summary results for foo bar 32

Comments  A foobared mirror
Tester    A. Non
Test date Mon Nov 03 15:53:13 2003

Evaluation wavelength 550

RMS          0.188 s.d.  0.0238
P-V          0.8 s.d.  0.146
Strehl ratio 0.247 s.d.  0.0715
Astigmatism  0.0935 Axis -72
Coma         0 Axis 0
SA           0.151 P-V 0.506
Average of 12 interferograms


Estimated Zernike coefficients and Standard errors
        Term      Coef. s.e.(Coef)  t value Classical aberration
          Z3   0.00000    0.00000      NaN Defocus
          Z4  -0.07530    0.00995    -7.57 Astigmatism
          Z5  -0.05540    0.01827    -3.03
          Z6   0.00000    0.00000      NaN Coma
          Z7   0.00000    0.00000      NaN
          Z8   0.15098    0.00644    23.43 Spherical 3rd
          Z9   0.00388    0.00515     0.75 Trefoil
         Z10  -0.00735    0.00517    -1.42
         Z11  -0.00184    0.00548    -0.34 Astigmatism 5th
         Z12   0.00855    0.00286     2.99
         Z13   0.00279    0.00296     0.94 Coma 5th
         Z14   0.00499    0.00195     2.56
         Z15  -0.04252    0.00245   -17.37 Spherical 5th
         Z16   0.03347    0.00178    18.77 Quadratic 7th
         Z17  -0.00967    0.00260    -3.73
         Z18  -0.00065    0.00212    -0.31 Triangular 7th
         Z19  -0.00147    0.00132    -1.11
         Z20  -0.00210    0.00188    -1.12 Astigmatism 7th
         Z21  -0.00574    0.00221    -2.60
         Z22   0.00160    0.00139     1.15 Coma 7th
```

```
Z23    0.00150    0.00158     0.95
Z24   -0.01435    0.00210    -6.84 Spherical 7th
Z25   -0.00329    0.00189    -1.74 5-fold 9th
Z26   -0.00402    0.00241    -1.67
Z27   -0.00399    0.00153    -2.62 Quadratic 9th
Z28   -0.00698    0.00127    -5.50
Z29   -0.00054    0.00178    -0.30 Triangular 9th
Z30   -0.00076    0.00084    -0.91
Z31    0.00015    0.00143     0.11 Astigmatism 9th
Z32    0.00053    0.00078     0.67
Z33    0.00183    0.00135     1.35 Coma 9th
Z34    0.00341    0.00131     2.60
Z35   -0.01775    0.00131   -13.58 Spherical 9th
Z36    0.00162    0.00060     2.69
```

4.8 Help

The two menu items under Help are provided to fill out the menu bar. The R style documentation and this manual are the primary reference sources for Rfringe. Limited context sensitive help is available by pressing the Help button in most dialog boxes.

References

[1] Born, M. and Wolf, E. (1999), *Principles of Optics, 7th Edition*, Cambridge University Press.

[2] Malacara, Daniel, ed. (1992), *Optical Shop Testing, 2nd Edition*, John Wiley & Sons.

[3] R Development Core Team (2003). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-00-3, URL http://www.R-project.org.

[4] Shannon, R.R. (1997), *The Art and Science of Optical Design*, Cambridge University Press, table 3.3.

[5] Suiter, H.R. (1994), *Star Testing Astronomical Telescopes*, Willmann-Bell, Inc.

[6] Wilson, R.N. (1996), *Reflecting Telescope Optics I*, Springer, table 3.23.

[7] http://mathworld.wolfram.com/ZernikePolynomial.html.