EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VERY LARGE TELESCOPE

FORS2 Spectropolarimetry Cookbook and Reflex Tutorial

VLT-MAN-ESO-19500-....

Issue 1.9

Date 17th May 2023

Prepared:	C.A. Hummel, S. Moehler, C.E. García Dabo	17th May 2023	
	Name	Date	Signature
Approved:	W.Freudling		
	Name	Date	Signature
Released:	M. Sterzik		
	Name	Date	Signature

This page was intentionally left blank

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
1.1	July 2017	All	First public release
1.2	October 2017	All	Changes for release 5.3.32
1.3	April 2019	All	Changes for release 5.4.3
1.5	June 2020	All	Changes for release 5.5.4
1.6	May 2021	All	Changes for release 5.5.7
1.7	May 2022	Sections 6 and 9 (interactive windows)	Changes for release 5.6.2
1.8	August 2022	Section 9	Changes for release 5.6.3
1.9	17/05/2023	Sections 9, 10	Changes for release 5.6.5

This page was intentionally left blank

Contents

1 Spectropolarimetry with the FORS2 instrument: an introduction

FORS is the visual and near-UV FOcal Reducer and low dispersion Spectrograph for the Very Large Telescope (VLT) of the European Southern Observatory (ESO). Two versions of FORS have been built and mounted at the VLT, but in April 2009, FORS1 was removed to make room for X-shooter, so only FORS2 is now in operation. FORS is designed as an all-dioptric instrument for the wavelength range from 330 nm to 1100 nm and provides an image scale in the standard mode of $0''.25$/pixel with the readout mode (2×2 binning).

The spectropolarimetric mode of FORS2 is enabled by a retarder waveplate (half-wave or quarter-wave retardance between the fast and slow axes) followed by a Wollaston prism (the "analyzer") which splits the light beam into two orthogonal polarizations. In order for the two beams not to overlap on the detector, a strip mask is formed by every second MOS slit jaw carrier arm across the field of view of the instrument as shown in Fig. 1.1. MOS slits can be positioned in the free strips along the dispersion axis to desired target locations.

Frames are recorded at different retarder plate angles in order to allow the computation of the Stokes parameters for the spectra (Bagnulo et al., 2009).

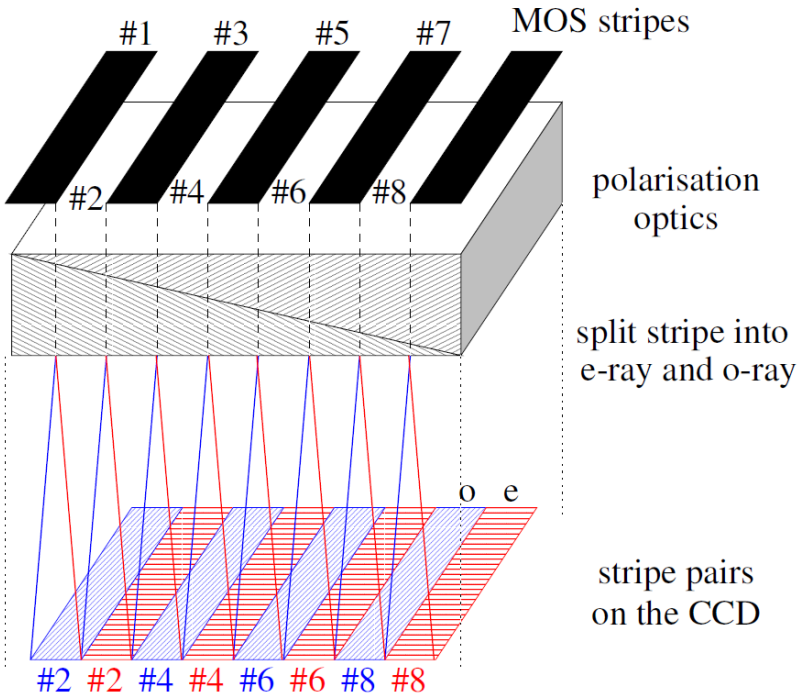


Figure 1.1: *Strip mask formed in the focal plane to allow recording of the ordinary o-ray and extraordinary e-ray beams on the detector. In the even numbered strips, MOS slits can be moved to a particular target position. For a single point source one strip is sufficient.*

This document explains how to reduce data taken in the spectropolarimetry mode of `FORS1` and `FORS2`. It will not cover the other modes, which are described in different cookbook and tutorial documents.

Information on FORS2 is given at `eso.org/sci/facilities/paranal/instruments/fors.html`, which also provides access to manuals. In particular, we recommend that you read the User Manual.

2 Data reduction of FORS2 spectropolarimetry

The reduction of FORS2 spectropolarimetry consists of essentially 4 steps:

1. Creating a master bias frame
2. Determining the dispersion relation and spatial distortion, together with the slit limits and creating a normalized flat field
3. Calibrating and extracting the spectra, including sky subtraction
4. Computing the Stokes parameters by combining the calibrated spectra of the ordinary and extraordinary beams, taken at different retarder angles

Observations of polarization standards (polarized and/or unpolarized) may also be used to remove systematic errors from the Stokes parameters. The calibration frames can be obtained at the same time when downloading the science data from the ESO Science Archive with the `CalSelector` option.

Reducing and calibrating FORS2 spectropolarimetric observations requires the following frames:

- Science frames obtained during the night
- Calibration frames
  - Bias frames
  - Internal screen flats
  - Arc lamp exposures
  - Polarization standard star observations (optional)
  - Static calibrations, e.g. standard star data

2.1 Bias frames

Bias frames are taken with an exposure time of 0 seconds and a closed shutter. They thus record only the signal that is added during the read-out of the CCD to avoid negative numbers. A sequence of 5 or 20 bias frames[1] are taken the day following the observations as part of the FORS calibration plan.

2.2 Flat field

The purpose of the flat-field is to remove the pixel-to-pixel sensitivity variations across the detector. As these variations act as a noise source, the precision of the flat-field correction will have direct consequences on the polarimetric precision that can be achieved, and on the signal-to-noise ratio of the reduced observations.

For the spectroscopic modes one will use internal screen flats in most cases. These flats are taken during daytime with the telescope pointing to zenith and the instrument in calibration position. The flats are taken at zero retarder plate angle.

[1] The calibration plan was changed in Nov. 2014, and 20 bias frames are now taken per setting instead of 5.

2.3 Wavelength calibration

For the wavelength calibration one may use the He, HgCd, and Ar lamps (at the lowest spectral resolution with grism 150I) and in addition the Ne lamp (at higher resolution). Wavelength calibration exposures are done during the day only and at all retarder plate angles used for the science observations.

2.4 Static Calibration Data

In addition to calibration frames taken regularly, the FORS pipeline also uses static calibration tables related to spectropolarimetry. In the following list, the data organizer category is given in parenthesis.

1. Arc lamp line wavelengths (`MASTER_LINECAT`). This table contains the reference wavelengths (in Å) for the arc lamp used.
2. Grism table (`GRISM_TABLE`). This table contains grism-specific parameters, like the dispersion in Å/pixel, the start and end wavelength, the polynomial degree to be used for the wavelength calibration, etc.
3. Distortion table (`MASTER_DISTORTION_TABLE`). This table is necessary for the identification of the ordinary and extraordinary spectral beams.
4. Waveplate chromatism (`RETARDER_WAVEPLATE_CHROMATISM`). This table provides information on the color dependency of the direction of the linear polarization vector.
5. Polarimetric standard stars catalogue (`STD_PMOS_TABLE`). A table named `fors2_pol_sta.fits`, listing the measured linear polarisation from a number of standard stars, is available in the calibration directory.

A description of these tables is given in the FORS Pipeline User Manual. We refer to this manual for detailed information on the recipes and on what parameters can be configured. These tables are provided together with the FORS pipeline and located in the directory <INSTALL-DIR>/calib/.

3 Running the FORS2 pipeline

There are three ways to run the ESO pipelines, in all cases the executed recipes are, however, the same. The differences are in the user interfaces.

1. Reflex is the **recommended** environment to reduce ESO data. It automatically organizes input files according to their category and runs the entire reduction chain at the push of a button. It supports break points in the reduction sequence in order to inspect and interact with intermediate and final products and rerun the corresponding step if necessary. A more detailed description on how to use Reflex to reduce FORS2 spectropolarimetric data is provided in sections 6 to 9.3.
2. Gasgano is a Java-based data file organizer developed and maintained by ESO. It can be used to manage and organize in a systematic way the astronomical data observed and produced by all VLT compliant telescopes. Gasgano offers functionalities for data viewing, grouping, sorting, classification, searching, and filtering of data. And, of course, Gasgano will run any requested CPL recipe on the selected data. Gasgano is automatically installed when installing the stand-alone FORS pipeline kit available from `http://www.eso.org/sci/software/pipelines/`, but not as part of the Reflex installation.
3. Esorex, a command-line utility for running pipeline recipes is also available (and is used also by Reflex to run the pipeline recipes). Esorex may be embedded by users into data reduction scripts for the automation of processing tasks. See `http://www.eso.org/sci/software/cpl/esorex.html` for more information.

The underlying algorithms and recipes are the same for a given instrument pipeline, irrespective if Reflex, Gasgano or Esorex is used.

The pipeline itself can be accessed from the web at `http://www.eso.org/sci/software/pipelines/`.

To reduce FORS2 spectropolarimetry data, one can just execute the following 3 recipes in succession:

1. `fors_bias`: process all bias files
2. `fors_pmos_calib`: processes all lamp calibration files (flat fields and arcs).
3. `fors_pmos_science`: processes the science or standard star files, applying the calibrations processed in the previous step. The science files are typically taken at these retarder angles, depending on the desired level of reduction of systematic error:
   (a) Circular polarimetry:
      i. 2 angles: -45.0, 45.0
      ii. 4 angles: -45.0, 45.0, 135.0, 225.0
   (b) linear polarimetry:
      i. 4 angles: 0.0, 22.5, 45.0, 67.5,
      ii. 8 angles: 0.0, 22.5, 45.0, 67.5, 90.0, 112.5, 135.0, 157.5
      iii. 16 angles:0.0, 22.5, 45.0, 67.5, 90.0, 112.5, 135.0, 157.5, 180.0, 202.5, 225.0, 247.5, 270.0, 292.5, 315.0, 337.5

More information of the FORS2 pipeline is available in the FORS Pipeline User Manual.

The association of calibration files with the science data is based on the so-called OCA rules (see Sect. 9.3.1), also used by the CalSelector to retrieve the correct calibrations from the archive with your science data. The OCA rules associate files by header keywords and we use Gasgano here to show this for the demo data set delivered with Reflex. The following GUI can be seen after having specified the data directory in Gasgano[2].

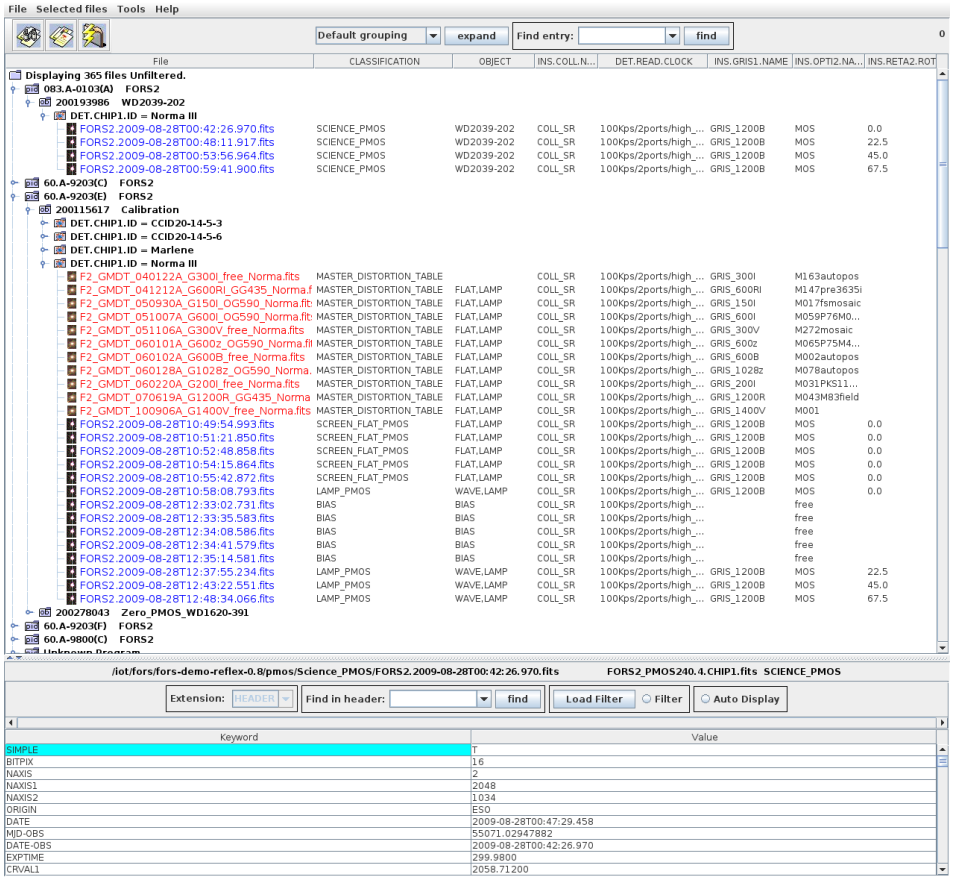


Figure 3.1: *The Gasgano window, showing the 4 science files at the top (recorded on the E2V mosaic called Norma) and the associated BIAS, FLAT, and WAVE calibration files.*

[2]The corresponding windows for Reflex are shown in Figs. 9.1

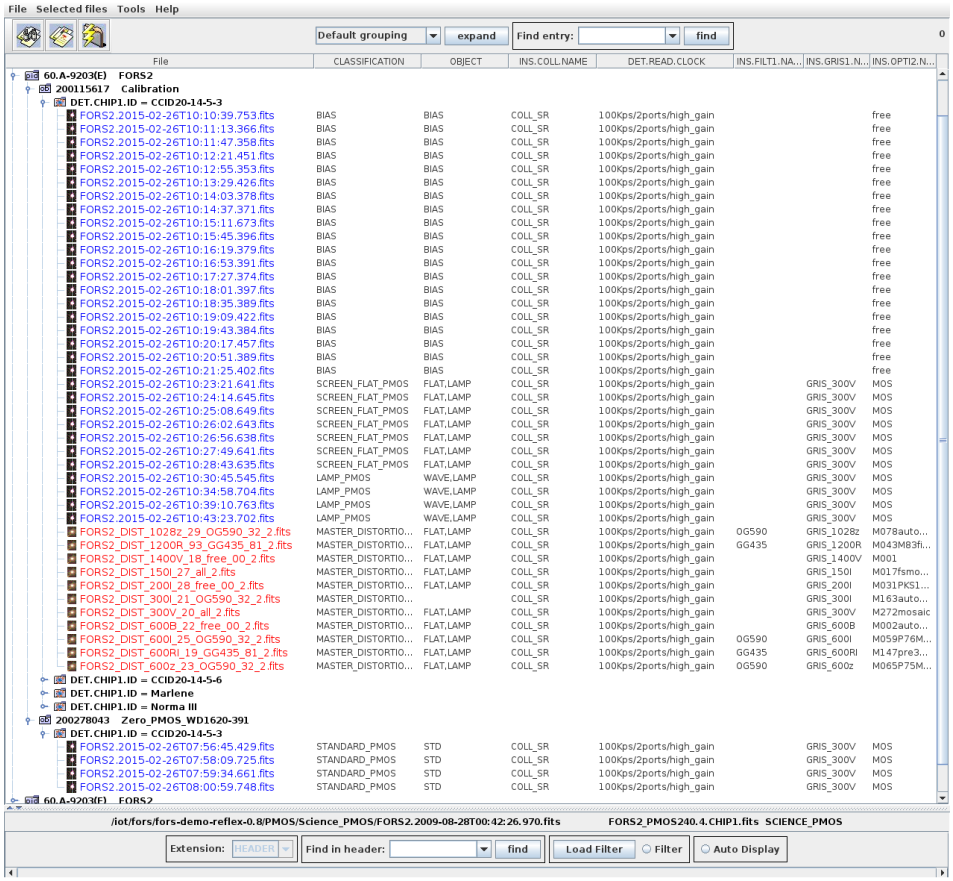


Figure 3.2: *The Gasgano window, showing the 4 files of the polarization standard at the top (recorded on the E2V mosaic called Norma) and the associated BIAS, FLAT, and WAVE calibration files.*

Gasgano (and also Reflex, see Sect. 9.3.1) allows to display files. For example, to check a wavelength calibration file, select one of category “LAMP_PMOS” and choose the “Display...” option under the “Selected files” menu. Below, we show an example.

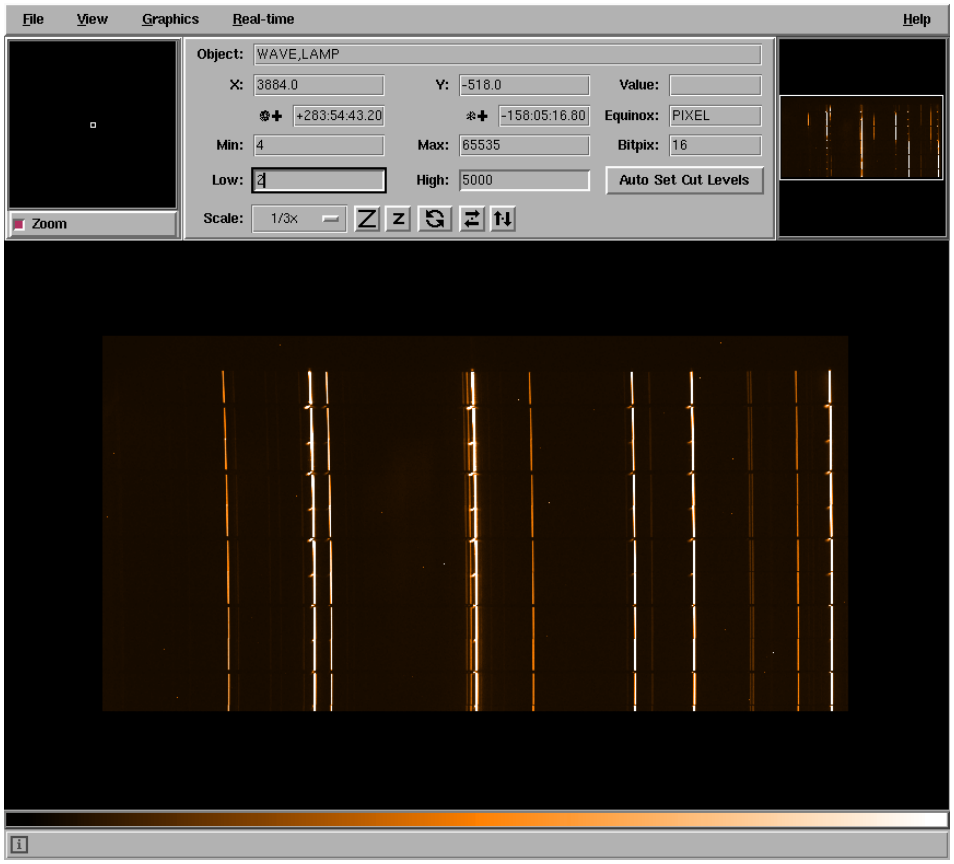


Figure 3.3: *Visualization of one of the wavelength calibration files. There is one file for each retarder angle setting. Note the slight curvature of the lines due to distortion.*

The following sections describe the data reduction using Reflex.

4 Introduction to `EsoReflex`

This document is a tutorial designed to enable the user to to reduce his/her data with the ESO pipeline run under an user-friendly environmet, called `EsoReflex`, concentrating on high-level issues such as data reduction quality and signal-to-noise (S/N) optimisation.

`EsoReflex` is the ESO Recipe Flexible Execution Workbench, an environment to run ESO VLT pipelines which employs a workflow engine to provide a real-time visual representation of a data reduction cascade, called a workflow, which can be easily understood by most astronomers. The basic philosophy and concepts of Reflex have been discussed by Freudling et al. (2013A&A...559A..96F). Please reference this article if you use Reflex in a scientific publication.

Reflex and the data reduction workflows have been developed by ESO and instrument consortia and they are fully supported. If you have any issue, please have a look to https://support.eso.org to see if this has been reported before or open a ticket for further support.

A workflow accepts science and calibration data, as downloaded from the archive using the CalSelector tool[3] (with associated raw calibrations) and organises them into DataSets, where each DataSet contains one science object observation (possibly consisting of several science files) and all associated raw and static calibrations required for a successful data reduction. The data organisation process is fully automatic, which is a major time-saving feature provided by the software. The DataSets selected by the user for reduction are fed to the workflow which executes the relevant pipeline recipes (or stages) in the correct order. Full control of the various recipe parameters is available within the workflow, and the workflow deals automatically with optional recipe inputs via built-in conditional branches. Additionally, the workflow stores the reduced final data products in a logically organised directory structure employing user-configurable file names.

The FORS2 `Reflex` workflow described in this tutorial supports the reduction of FORS2 spectropolarimetric observations (PMOS). The user is referred to the FORS User Manual [4] for more information on the instrument itself, and the FORS Pipeline User Manual[5] for the details of the spectroscopic FORS2 pipeline recipes. There are also other tutorials that guide you through the MOS/MXU/LSS workflow and the IMG workflow respectively. Check the ESO pipeline main webpage.

The quick start section (see Section 6) describes the minimum effort to get started, and it makes up only two pages of text in this tutorial.

[3] *http://www.eso.org/sci/archive/calselectorInfo.html*
[4] available at *http://www.eso.org/sci/facilities/paranal/instruments/fors/doc*
[5] available at *https://www.eso.org/sci/software/pipelines/index.html* (Documentation)

5 Software Installation

`Esoreflex` and the workflows can be installed in different ways: via package repositories, via the `install_esoreflex` script or manually installing the software tar files.

The recommended way is to use the package repositories if your operating system is supported. The `macports` repositories support macOS 10.14 to 11, while the `rpm/yum` repositories support Fedora 28 to 32, CentOS 7, Scientific Linux 7. For any other operating system it is recommended to use the `install_esoreflex` script.

The installation from package repository requires administrative privileges (typically granted via sudo), as it installs files in system-wide directories under the control of the package manager. If you want a local installation, or you do not have sudo privileges, or if you want to manage different installations on different directories, then use the `install_esoreflex` script. Note that the script installation requires that your system fulfill several software prerequisites, which might also need sudo privileges.

Reflex 2.11.x needs java JDK 11 to be installed.

Please note that in case of major or minor (affecting the first two digit numbers) Reflex upgrades, the user should erase the `$HOME/KeplerData`, `$HOME/.kepler` directories if present, to prevent possible aborts (i.e. a hard crash) of the `esoreflex` process.

5.1 Installing Reflex workflows via `macports`

This method is supported for the macOS operating system. It is assumed that macports (*http://www.macports.org*) is installed. Please read the full documentation at *http://www.eso.org/sci/software/pipelines/installation/macports.html*.

5.2 Installing Reflex workflows via `rpm/yum/dnf`

This method is supported for Fedora 28 to 32, CentOS 7, Scientific Linux 7 operating systems, and requires sudo rights. To install, please follow these steps

1. Configure the ESO repository (This step is only necessary if the ESO repository has not already been previously configured).
   - If you are running Fedora, run the following commands:

     ```
     sudo dnf install dnf-plugins-core
     sudo dnf config-manager --add-repo=ftp://ftp.eso.org/pub/dfs/
                 pipelines/repositories/stable/fedora/esorepo.repo
     ```

   - If you are running CentOS 7, run the following commands:

     ```
     sudo yum install yum-utils ca-certificates yum-conf-repos
     sudo yum install epel-release
     sudo yum-config-manager --add-repo=ftp://ftp.eso.org/pub/dfs/
               pipelines/repositories/stable/centos/esorepo.repo
     ```

- If you are running SL 7, run the following commands:

```
sudo yum install yum-utils ca-certificates yum-conf-repos
sudo yum install yum-conf-epel
sudo yum-config-manager --add-repo=ftp://ftp.eso.org/pub/dfs/
    pipelines/repositories/stable/sl/esorepo.repo
```

2. Install the pipelines

- The list of available top level packages for different instruments is given by:

```
sudo dnf list esopipe-\*-all # (Fedora)
sudo yum list esopipe-\*-all # (CentOS 7, SL 7)
```

- To install an individual pipeline use the following (This example is for X-Shooter. Adjust the package name to the instrument you require.):

```
sudo dnf install esopipe-xshoo-all # (Fedora)
sudo yum install esopipe-xshoo-all # (CentOS 7, SL 7)
```

- To install all pipelines use:

```
sudo dnf install esopipe-\*-all # (Fedora)
sudo yum install esopipe-\*-all # (CentOS 7, SL 7)
```

For further information, please read the full documentation at
http://www.eso.org/sci/software/pipelines/installation/rpm.html.

5.3 Installing Reflex workflows via `install_esoreflex`

This method is recommended for operating systems other than what indicated above, or if the user has no sudo rights. Software dependencies are not fulfilled by the installation script, therefore the user has to install all the prerequisites before running the installation script.

The software pre-requisites for `Reflex 2.11` may be found at:
http://www.eso.org/sci/software/pipelines/reflex_workflows

To install the `Reflex 2.11` software and demo data, please follow these instructions:

1. From any directory, download the installation script:

```
wget https://ftp.eso.org/pub/dfs/reflex/install_esoreflex
```

2. Make the installation script executable:

```
chmod u+x install_esoreflex
```

3. Execute the installation script:

```
./install_esoreflex
```

and the script will ask you to specify three directories: the download directory `<download_dir>`, the software installation directory `<install_dir>`, and the directory to be used to store the demo data `<data_dir>`. If you do not specify these directories, then the installation script will create them in the current directory with default names.

4. Follow all the script instructions; you will be asked whether to use your Internet connection (recommended: yes), the pipelines and demo-datasets to install (note that the installation will remove all previously installed pipelines that are found in the same installation directory).

5. To start `Reflex`, issue the command:

```
<install_dir>/bin/esoreflex
```

It may also be desirable to set up an alias command for starting the `Reflex` software, using the shell command `alias`. Alternatively, the `PATH` variable can be updated to contain the `<install_dir>/bin` directory.

5.4 Demo Data

Together with the pipeline you will also receive a demo data set, that allows you to run the `Reflex FORS2` workflow without any changes in parameters. This way you have a data set to experiment with before you start to work on your own data.

Note that you will need a minimum of ∼0.5 GB, ∼0.6 GB and ∼1 GB of free disk space for the directories `<download_dir>`, `<install_dir>` and `<data_dir>`, respectively. The FORS2 demo data have been retrieved with the CalSelector tool[6].

[6] *http://www.eso.org/sci/archive/calselectorInfo.html*

6 Quick Start: Reducing The Demo Data

For the user who is keen on starting reductions without being distracted by detailed documentation, we describe the steps to be performed to reduce the science data provided in the FORS2 demo data set supplied with the `esoreflex 2.11` release. By following these steps, the user should have enough information to perform a reduction of his/her own data without any further reading:

1. First, type:

   ```
   esoreflex -l
   ```

   If the `esoreflex` executable is not in your path, then you have to provide the command with the executable full path `<install_dir>/bin/esoreflex -l` . For convenience, we will drop the reference to `<install_dir>`. A list with the available `esoreflex` workflows will appear, showing the workflow names and their full path.

2. Open the FORS Spectropolarimetry by typing:

   ```
   esoreflex fors_pmos&
   ```

   Alternatively, you can type only the command `esoreflex` the empty canvas will appear (Figure 6.1) and you can select the workflow to open by clicking on `File -> Open File`. Note that the loaded workflow will appear in a new window. The FORS Spectropolarimetry workflow is shown in Figure 6.2.

3. To aid in the visual tracking of the reduction cascade, it is advisable to use component (or actor) highlighting. Click on `Tools -> Animate at Runtime`, enter the number of milliseconds representing the animation interval (100 ms is recommended), and click `OK`.

4. Change directories set-up. Under "Setup Directories" in the workflow canvas there are seven parameters that specify important directories (green dots).

   By default, the `ROOT_DATA_DIR`, which specifies the working directory within which the other directories are organised. is set to your `$HOME/reflex_data` directory. All the temporary and final products of the reduction will be organized under sub-directories of `ROOT_DATA_DIR`, therefore make sure this parameter points to a location where there is enough disk space. To change `ROOT_DATA_DIR`, double click on it and a pop-up window will appear allowing you to modify the directory string, which you may either edit directly, or use the `Browse` button to select the directory from a file browser. When you have finished, click `OK` to save your changes.

   Changing the value of `RAW_DATA_DIR` is the only necessary modification if you want to process data other than the demo data

5. Click the  button to start the workflow

6. The workflow will highlight the `Data Organiser` actor which recursively scans the raw data directory (specified by the parameter `RAW_DATA_DIR` under "Setup Directories" in the workflow canvas) and constructs the datasets. Note that the raw and static calibration data must be present either

in `RAW_DATA_DIR` or in `CALIB_DATA_DIR`, otherwise datasets may be incomplete and cannot be processed. However, if the same reference file was downloaded twice to different places this creates a problem as `esoreflex` cannot decide which one to use.

7. The `Data Set Chooser` actor will be highlighted next and will display a "Select Datasets" window (see Figure 6.3) that lists the datasets along with the values of a selection of useful header keywords[7]. The first column consists of a set of tick boxes which allow the user to select the datasets to be processed. By default all complete datasets which have not yet been reduced will be selected. A full description of the options offered by the `Data Set Chooser` will be presented in Section 9.3.2.

8. Click the `Continue` button and watch the progress of the workflow by following the red highlighting of the actors. A window will show which dataset is currently being processed.

9. Once the reduction of all datasets has finished, a pop-up window called *Product Explorer* will appear, showing the datasets which have been reduced together with the list of final products. This actor allows the user to inspect the final data products, as well as to search and inspect the input data used to create any of the products of the workflow. Figure 6.4 shows the Product Explorer window. A full description of the *Product Explorer* will be presented in Section 9.3.3.

10. After the workflow has finished, all the products from all the datasets can be found in a directory under `END_PRODUCTS_DIR` named after the workflow start timestamp. Further subdirectories will be found with the name of each dataset.

Well done! You have successfully completed the quick start section and you should be able to use this knowledge to reduce your own data. However, there are many interesting features of `Reflex` and the FORS2 workflow that merit a look at the rest of this tutorial.

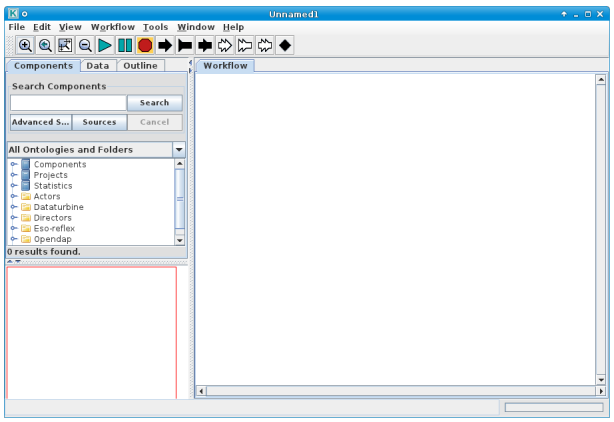


Figure 6.1: *The empty* `Reflex` *canvas.*

[7]The keywords listed can be changed by double clicking on the `DataOrganiser` Actor and editing the list of keywords in the second line of the pop-up window. Alternatively, instead of double-clicking, you can press the right mouse button on the `DataOrganiser` Actor and select `Configure Actor` to visualize the pop-up window.

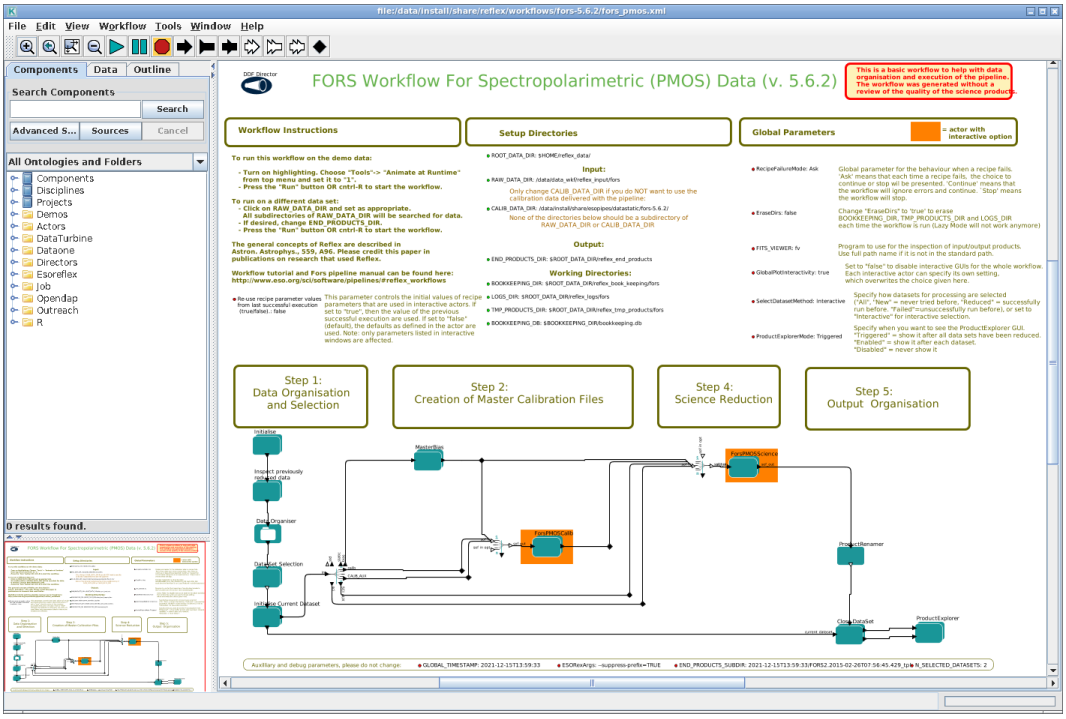


Figure 6.2: *FORS2 workflow general layout.*

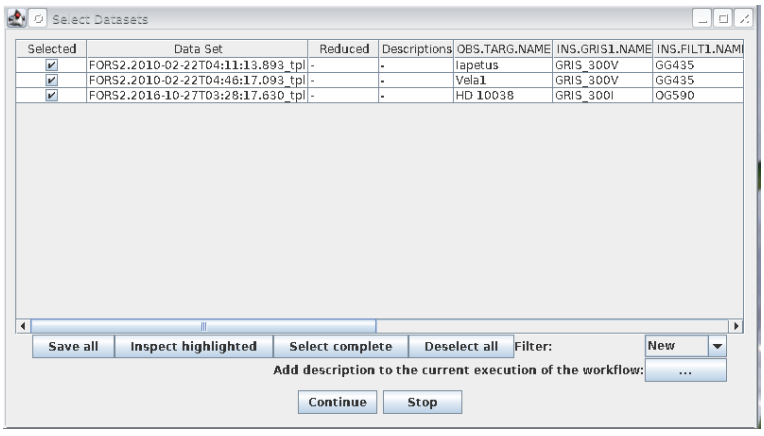


Figure 6.3: *The "Select Datasets" pop-up window.*

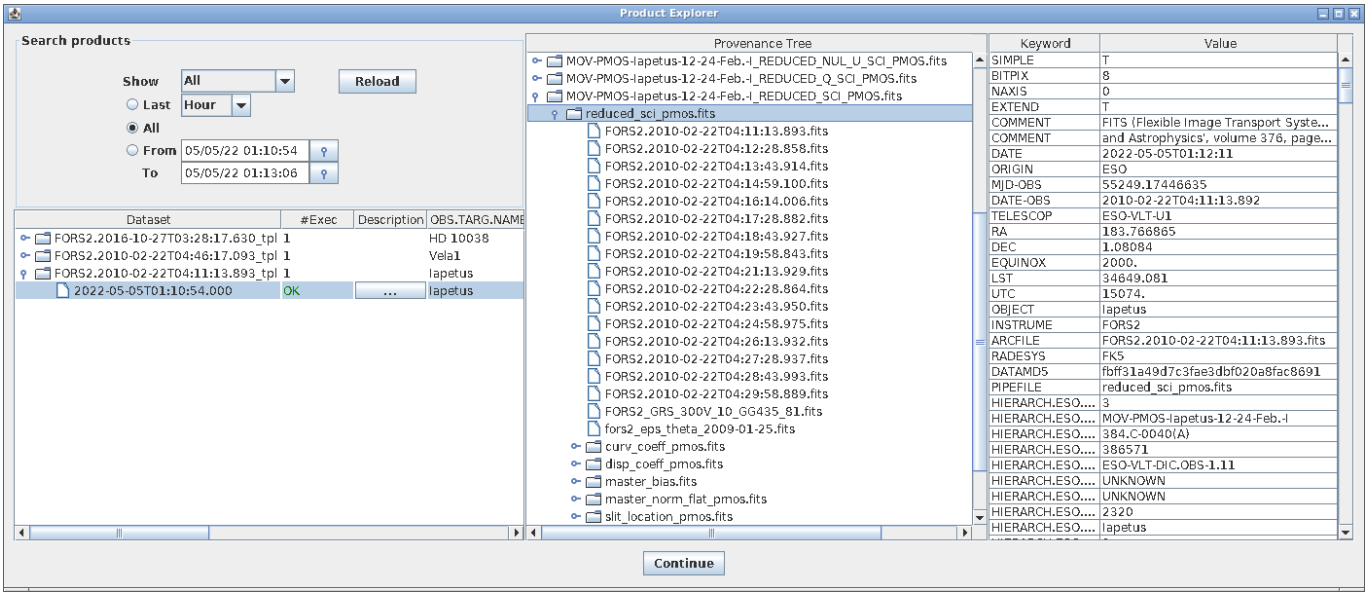


Figure 6.4: *The Product Explorer shows all datasets reduced in previous executions together with the full reduction chain for all the pipeline products.*

7 About the main `esoreflex` canvas

7.1 Saving And Loading Workflows

In the course of your data reductions, it is likely that you will customise the workflow for various data sets, even if this simply consists of editing the `ROOT_DATA_DIR` to a different value for each data set. Whenever you modify a workflow in any way, you have the option of saving the modified version to an `XML` file using `File -> Export As` (which will also open a new workflow canvas corresponding to the saved file). The saved workflow may be opened in subsequent `esoreflex` sessions using `File -> Open`. Saving the workflow in the default Kepler format (.kar) is only advised if you do not plan to use the workflow with another computer.

7.2 Buttons

At the top of the `esoreflex` canvas are a set of buttons which have the following functions:

-  - Zoom in.
-  - Reset the zoom to 100%.
-  - Zoom the workflow to fit the current window size (Recommended).
-  - Zoom out.
-  - Run (or resume) the workflow.
-  - Pause the workflow execution.
-  - Stop the workflow execution.

The remainder of the buttons (not shown here) are not relevant to the workflow execution.

7.3 Workflow States

A workflow may only be in one of three states: executing, paused, or stopped. These states are indicated by the yellow highlighting of the  ,  , and  buttons, respectively. A workflow is executed by clicking the  button. Subsequently the workflow and any running pipeline recipe may be stopped immediately by clicking the  button, or the workflow may be paused by clicking the  button which will allow the current actor/recipe to finish execution before the workflow is actually paused. After pausing, the workflow may be resumed by clicking the  button again.

8 The FORS2 Workflow

The FORS2 workflow canvas is organised into a number of areas. From top-left to top-right you will find general workflow instructions, directory parameters, and global parameters. In the middle row you will find five boxes describing the workflow general processing steps in order from left to right, and below this the workflow actors themselves are organised following the workflow general steps.

8.1 Workflow Canvas Parameters

The workflow canvas displays a number of parameters that may be set by the user. Under "Setup Directories" the user is only required to set the `RAW_DATA_DIR` to the working directory for the dataset(s) to be reduced, which, by default, is set to the directory containing the demo data. The `RAW_DATA_DIR` is recursively scanned by the `Data Organiser` actor for input raw data. The directory `CALIB_DATA_DIR`, which is by default within the pipeline installation directory, is also scanned by the `Data Organiser` actor to find any static calibrations that may be missing in your dataset(s). If required, the user may edit the directories `BOOKKEEPING_DIR`, `LOGS_DIR`, `TMP_PRODUCTS_DIR`, and `END_PRODUCTS_DIR`, which correspond to the directories where book-keeping files, logs, temporary products and end products are stored, respectively (see the Reflex User Manual for further details; Forchì (2012)).

There is a mode of the `Data Organiser` that skips the built-in data organisation and uses instead the data organisation provided by the CalSelector tool. To use this mode, click on `Use CalSelector associations` in the `Data Organiser` properties and make sure that the input data directory contains the XML file downloaded with the CalSelector archive request (note that this does not work for all instrument workflows).

Under the "Global Parameters" area of the workflow canvas, the user may set the `FITS_VIEWER` parameter to the command used for running his/her favourite application for inspecting FITS files. Currently this is set by default to `fv`, but other applications, such as `ds9`, `skycat` and `gaia` for example, may be useful for inspecting image data. Note that it is recommended to specify the full path to the visualization application (an alias will not work).

By default the `EraseDirs` parameter is set to `false`, which means that no directories are cleaned before executing the workflow, and the recipe actors will work in Lazy Mode (see Section 8.2.4), reusing the previous pipeline recipe outputs if input files and parameters are the same as for the previous execution, which saves considerable processing time. Sometimes it is desirable to set the `EraseDirs` parameter to `true`, which forces the workflow to recursively delete the contents of the directories specified by `BOOKKEEPING_DIR`, `LOGS_DIR`, and `TMP_PRODUCTS_DIR`. This is useful for keeping disk space usage to a minimum and will force the workflow to fully re-reduce the data each time the workflow is run.

The parameter `RecipeFailureMode` controls the behaviour in case that a recipe fails. If set to `Continue`, the workflow will trigger the next recipes as usual, but without the output of the failing recipe, which in most of the cases will lead to further failures of other recipes without the user actually being aware of it. This mode might be useful for unattended processing of large number of datasets. If set to `Ask`, a pop-up window will ask whether the workflow should stop or continue. This is the default. Alternatively, the `Stop` mode will stop the workflow execution immediately.

The parameter `ProductExplorerMode` controls whether the `ProductExplorer` actor will show its window or not. The possible values are `Enabled`, `Triggered`, and `Disabled`. `Enabled` opens the Product-

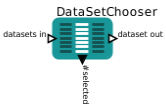
Explorer GUI at the end of the reduction of each individual dataset. `Triggered` (default and recommended) opens the ProductExplorer GUI when all the selected datasets have been reduced. `Disabled` does not display the ProductExplorer GUI.

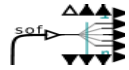
8.2 Workflow Actors

8.2.1 Simple Actors

Simple actors have workflow symbols that consist of a single (rather than multiple) green-blue rectangle. They may also have an icon within the rectangle to aid in their identification. The following actors are simple actors:

- 

- The `DataOrganiser` actor.
- 

- The `DataSetChooser` actor (inside a composite actor).
- 

- The `FitsRouter` actor Redirects files according to their categories.
- 

- The `ProductRenamer` actor.
- 

- The `ProductExplorer` actor (inside a composite actor).

Access to the parameters for a simple actor is achieved by right-clicking on the actor and selecting `Configure Actor`. This will open an "Edit parameters" window. Note that the `Product Renamer` actor is a jython script (Java implementation of the Python interpreter) meant to be customised by the user (by double-clicking on it).

8.2.2 Composite Actors

Composite Actors have workflow symbols that consist of multiple-layered green-blue rectangles. They generally do not have a logo within the rectangle. A Composite Actor represents a combination of more Simple or Composite Actors which hides over-complexity from the user in the top-level workflow. In the FORS2 workflow, the following actors are composite actors:

- 

- The `Initialise` actor.
- 

- The `Initialise Current DataSet` actor.

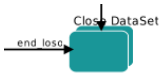
- 

- The `Product Explorer` actor (contains the simple actor).
- 

- The `Initialise Current DataSet` actor.
- 

- The `MasterBias` actor.
- 

- The `ForsPmosCalib` actor.
- 

- The `ForsPmosScience` actor.
- 

- The `Close DataSet` actor.

Composite Actors may also be expanded for inspection. To do this, right-click on the actor and select `Open Actor`, which will expand the Composite Actor components in a new `Reflex` canvas window. If the Composite Actor corresponds to a pipeline recipe, then the corresponding `RecipeExecuter` actor will be present as a Simple Actor, and its parameters are accessible as for any other Simple Actor. Alternatively you may still find Composite Actors, on which you need to repeat the first step to access the `Recipe Executer`.

8.2.3 Recipe Execution within Composite Actors

The FORS2 workflow contains Composite Actors to run pipeline recipes. This is in the most simple case due to the `SoF Splitter`/`SoF Accumulator`[8], which allow to process calibration data from different settings within one given DataSet (e.g. lamp frames taken with different slits/masks). More complex Composite Actors contain several actors (e.g. `Recipe Executer`).

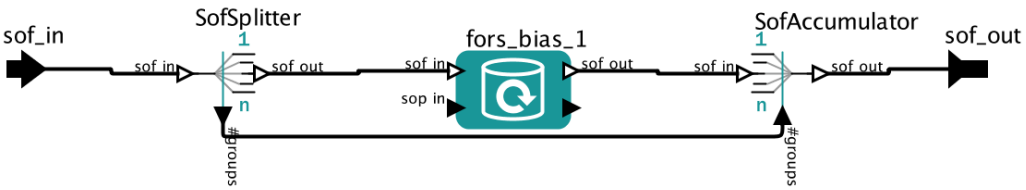


Figure 8.1: *This is the window you get when you choose* `Open Actor` *for the Composite Actor* `MasterBias`. *This is the most simple case for a Composite Actor. Using* `Configure Actor` *on* `fors_bias_1` *gives you Fig. 8.2.*

[8] SoF stands for Set of Files, which is an ASCII file containing the name (and path) of each input file and its category (e.g. `BIAS`).

Table 8.1: The FORS2 pipeline actors and their contents

actor	recipes	description
MasterBias	`fors_bias`	create master bias
ForsPmosCalib	`fors_pmos_calib`	create master flat, determine coefficients for wavelength calibration and correction of spatial distortion
ForsPmosScience	`fors_pmos_science`	reduce science data and standard star data

The central elements of any Reflex workflow are the `RecipeExecuter` actors that actually run the recipes. One basic way to embed a `RecipeExecuter` in a workflow is shown in Fig 8.1, which is the most simple version of a Composite Actor. The `RecipeExecuter` is preceded by an `SofSplitter`, and followed by an `SofAccumulator`. The function of the `SofSplitter` is to investigate the incoming SoFs, sort them by "purpose", and create separate SoFs for each purpose. The `RecipeExecuter` then processes each of the SoFs independently (unless they are actually the same files). Finally, the `SofAccumulator` packs all the results into a single output SoF. The direct relation between the `SofSplitter` and `SofAccumulator` is used to communicate the number of different SoFs created by the `SofSplitter`. A workflow will only work as intended if the purpose of all the files a recipe needs as input is identical. The only exception to this rule is that a purpose can also be "default". In this case, the file is included in any output SoF created by the `SoFsplitter` and `SofAccumulator`.

The reason for this scheme is best explained by an example. For a complex DataSet, the `Data Organiser` might have selected a large number of individual raw lamp frames (arc and flat field). The different lamp frames are to be used to calibrate different frames, e.g. the science frames and the standard star frames. The `Data Organiser` determines and records this "purpose" of each lamp frame, and this information is included in the DataSet and each SoF created from this DataSet. The `FitsRouter` directs all raw lamp frames to the `ForsCalib` Composite Actor. The `SofSplitter` then creates SoFs, one for the lamp frames to be used for the science frames, and (probably) separate ones for the lamp frames to be used for the standard star observations. The `fors_calib` recipe creates one master flat field (and other products) for each SoF, and the `SofAccumulator` then creates a SoF that contains all the products.

A `RecipeExecuter` actor is used in the workflow to run a single FORS2 pipeline recipe (e.g: in the `MasterBias` actor the recipe `fors_bias` is executed). In order to configure the `RecipeExecuters`, one has to first use `Open Actor` to get to the level of the recipe executors (see Fig. 8.1).

In Figure 8.2 we show the "Edit parameters" window for a typical `RecipeExecuter` actor, which can be displayed by right-clicking on the actor and selecting `Configure Actor`. In the following we describe in more detail the function of some of the parameters for a `RecipeExecuter` actor:

- The "recipe" parameter states the FORS2 pipeline recipe which will be executed.
- The "mode" parameter has a pull-down menu allowing the user to specify the execution mode of the actor. The available options are:
  - `Run`: The pipeline recipe will be executed, possibly in Lazy mode (see Section 8.2.4). This option is the default option.
  - `Skip`: The pipeline recipe is not executed, and the actor inputs are passed to the actor outputs.

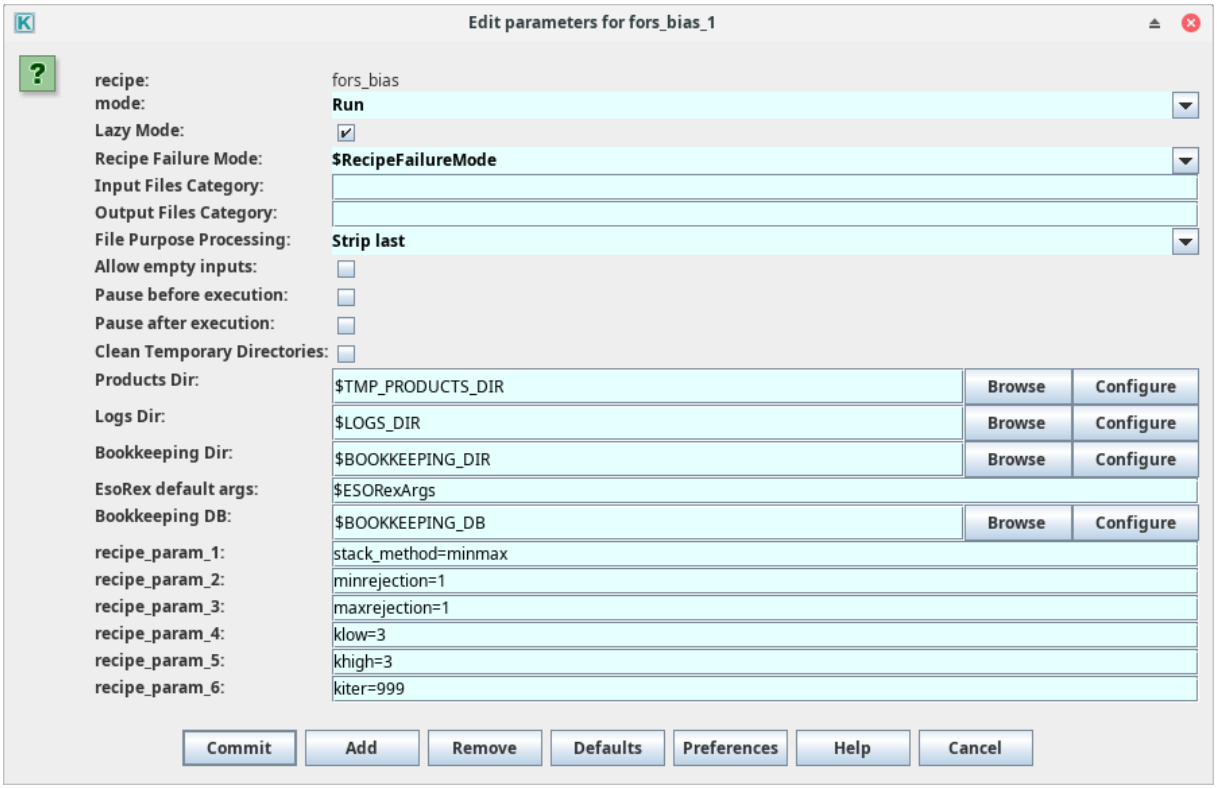


Figure 8.2: *The "Edit parameters" window for a typical* `RecipeExecuter` *actor, the* `fors_bias_1` *actor which runs the* `fors_bias` *pipeline recipe.*

  - `Disabled`: The pipeline recipe is not executed, and the actor inputs are not passed to the actor outputs.

- The "Lazy Mode" parameter has a tick-box (selected by default) which indicates whether the `RecipeExecuter` actor will run in Lazy mode or not. A full description of Lazy mode is provided in Sect. 8.2.4.

- The "Recipe Failure Mode" parameter has a pull-down menu allowing the user to specify the behaviour of the actor if the pipeline recipe fails. The available options are:

  - `Stop`: The actor issues an error message and the workflow stops.
  - `Continue`: The actor creates an empty output and the workflow continues.
  - `Ask`: The actor displays a pop-up window and asks the user whether he/she wants to continue or stop the workflow. This option is the default option.

- The set of parameters which start with "recipe param" and end with a number or a string correspond to the parameters of the relevant FORS2 pipeline recipe. By default in the `RecipeExecuter` actor, the pipeline recipe parameters are set to their pipeline default values. If you need to change the default parameter value for any pipeline recipe, then this is where you should edit the value[9]. For more information on

[9]Some of the pipeline parameters are read from the `GRISM_TABLE`s, which contain grism-specific parameters. These cannot be changed here.

the FORS2 pipeline recipe parameters, the user should refer to the FORS2 pipeline user manual (FORS Pipeline User Manual).

The description of the remainder of the `RecipeExecuter` actor parameters are outside the scope of this tutorial, and the interested user is referred to the Reflex User Manual for further details (Forchì 2012). Any changes that you make in the "Edit parameters" window may be saved in the workflow by clicking the `Commit` button when you have finished.

8.2.4 Lazy Mode

By default, all `RecipeExecuter` actors in a pipeline workflow are "Lazy Mode" enabled. This means that when the workflow attempts to execute such an actor, the actor will check whether the relevant pipeline recipe has already been executed with the same input files and with the same recipe parameters. If this is the case, then the actor will not execute the pipeline recipe, and instead it will simply broadcast the previously generated products to the output port. The purpose of the Lazy Mode is therefore to minimise any reprocessing of data by avoiding data re-reduction where it is not necessary.

One should note that the actor's Lazy Mode depends on the contents of the directory specified by the parameter `BOOKKEEPING_DIR` and the relevant FITS file checksums. Any modification to the directory contents and/or the file checksums will cause the corresponding actor to run the pipeline recipe again when executed, thereby re-reducing the input data.

The re-reduction of data at each execution may sometimes be desirable. To force a re-reduction of data for any single `RecipeExecuter` actor in the workflow, right-click the actor, select `Configure Actor`, and uncheck the Lazy mode parameter tick-box in the "Edit parameters" window that is displayed. For many workflows the `RecipeExecuter` actors are actually found inside the composite actors in the top level workflow. To access such embedded `RecipeExecuter` actors you will first need to open the sub-workflow by right-clicking on the composite actor and then selecting `Open Actor`.

To force the re-reduction of all data in a workflow (i.e. to disable Lazy mode for the whole workflow), you must uncheck the Lazy mode for every single `RecipeExecuter` actor in the entire workflow. It is also possible to change the name of the bookkeeping directory, instead of modifying any of the Lazy mode parameters. This will also force a re-reduction of the given dataset(s). A new reduction will start (with the lazy mode still enabled), but the results of previous reduction will not be reused. Alternatively, if there is no need to keep any of the previously reduced data, one can simply set the `EraseDirs` parameter under the "Global Parameters" area of the workflow canvas to `true`. This will then remove all previous results that are stored in the bookkeeping, temporary, and log directories before processing the input data, in effect, starting a new clean data reduction and re-processing every input dataset. *Note: The option* `EraseDirs = true` *does not work in* `esoreflex` *version 2.9.x and makes the workflow to crash.*

9 Reducing your own data

In this section we describe how to reduce your own data set.

First, we suggest the reader to familiarize with the workflow by reducing the demo dataset first (Section 6), but it is not a requirement.

9.1 The `esoreflex` command

We list here some options associated to the `esoreflex` command. We recommend to try them to familiarize with the system. In the following, we assume the `esoreflex` executable is in your path; if not you have to provide the full path `<install_dir>/bin/esoreflex`

To see the available options of the `esoreflex` command type:

```
esoreflex -h
```

The output is the following.

```
-h | -help              print this help message and exit.
-v | -version           show installed Reflex version and pipelines and exit.
-l | -list-workflows    list available installed workflows and from
                        ~/KeplerData/workflows.
-n | -non-interactive   enable non-interactive features.
-e | -explore           run only the Product Explorer in this workflow
-p <workflow> | -list-parameters <workflow>
                        lists the available parameters for the given
                        workflow.
-config <file>          allows to specify a custom esoreflex.rc configuration
                        file.
-create-config <file>   if <file> is TRUE then a new configuration file is
                        created in ~/.esoreflex/esoreflex.rc. Alternatively
                        a configuration file name can be given to write to.
                        Any existing file is backed up to a file with a '.bak'
                        extension, or '.bakN' where N is an integer.
-debug                  prints the environment and actual Reflex launch
                        command used.
```

9.2 Launching the workflow

We list here the recommended way to reduce your own datasets. Steps 1 and 2 are optional and one can start from step 3.

1. Type: `esoreflex -n <parameters>` FORS Spectropolarimetry to launch the workflow non interactively and reduce all the datasets with default parameters.

<parameters> allows you to specify the workflow parameters, such as the location of your raw data and the final destination of the products.

For example, type (in a single command line):

```
esoreflex -n
  -RAW_DATA_DIR /home/user/my_raw_data
  -ROOT_DATA_DIR /home/user/my_reduction
  -END_PRODUCTS_DIR $ROOT_DATA_DIR/reflex_end_products
  fors_pmos
```

to reduce the complete datasets that are present in the directory /home/user/my_raw_data and that were not reduced before. Final products will be saved in /home/user/my_reduction/reflex_end_products, while book keeping, temporary products, and logs will be saved in sub-directories of /home/user/my_reduction/. If the reduction of a dataset fails, the reduction continues to the next dataset. It can take some time, depending on the number of datasets present in the input directory. For a full list of workflow parameters type esoreflex -p FORS Spectropolarimetry. Note that this command lists only the parameters, but does not launch the workflow.

Once the reduction is completed, one can proceed with optimizing the results with the next steps.

2. Type:

   esoreflex -e fors_pmos

   to launch the Product Explorer. The Product Explorer allows you to inspect the data products already reduced by the FORS Spectropolarimetry esoreflex workflow. Only products associated with the workflow default bookkeeping database are shown. To visualize products associated to given bookkeeping database, pass the full path via the BOOKKEEPING_DB parameter:

   esoreflex -e BOOKKEEPING_DB <database_path> fors_pmos

   to point the product explorer to a given <database_path>, e.g., /home/username/reflex/reflex_bookkeeping/test.db

   The Product Explorer allows you to inspect the products while the reduction is running. Press the button Refresh to update the content of the Product Explorer. This step can be launched in parallel to step 1.

   A full description of the Product Explorer will be given in Section 9.3.3

3. Type:

   esoreflex fors_pmos &

   to launch the FORS Spectropolarimetry esoreflex workflow. The FORS Spectropolarimetry workflow window will appear (Fig. 6.2). Please configure the set-up directories ROOT_DATA_DIR, RAW_DATA_DIR, and other workflow parameters as needed. Just double-click on them, edit the content, and press OK. Remember to specify the same <database_path> as for the Product Explorer, if it has been opened at step #2, to synchronize the two processes.

4. (Recommended, but not mandatory) On the main esoreflex menu set Tools -> Animate at Runtime to 1 in order to highlight in red active actors during execution.

5. Press the button  to start the workflow. First, the workflow will highlight and execute the Initialise actor, which among other things will clear any previous reductions if required by the user (see Section 8.1).

Secondly, if set, the workflow will open the Product Explorer, allowing the user to inspect previously reduced datasets (see Section 9.3.3 for how to configure this option).

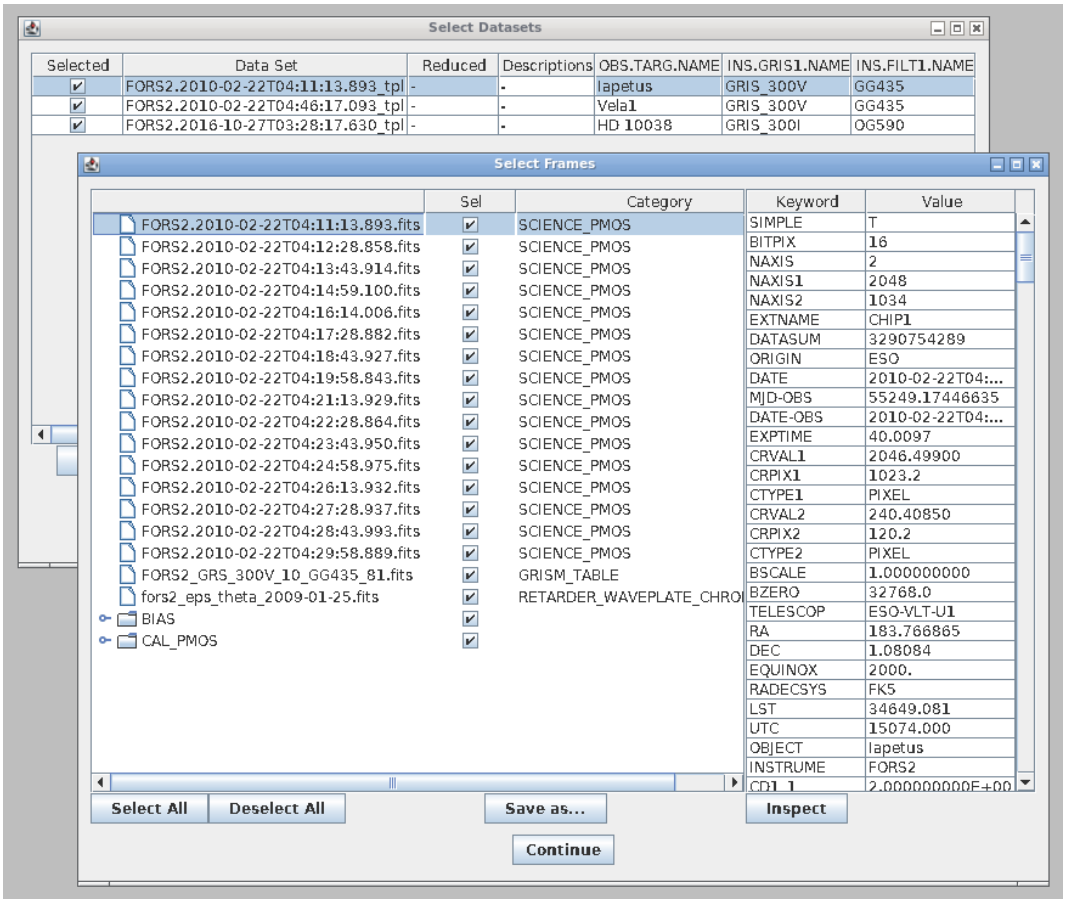


Figure 9.1: *The "Select Frames" window with a single file from the current Data Set highlighted in blue, and the corresponding FITS header displayed in the text box on the right. Hidden partially behind the "Select Frames" window is the "Select DataSets" window with the currently selected DataSet highlighted in blue.*

9.3 Workflow Steps

9.3.1 Data Organisation And Selection

The `DataOrganiser` (DO) is the first crucial component of a Reflex workflow. The DO takes as input `RAW_DATA_DIR` and `CALIB_DATA_DIR` and it detects, classifies, and organises the files in these directories and any subdirectories. The output of the DO is a list of "DataSets". A DataSet is a special Set of Files (SoF). A DataSet contains one or several science (or calibration) files that should be processed together, and all files needed to process these data. This includes any calibration files, and in turn files that are needed to process these calibrations. Note that different DataSets might overlap, i.e. some files might be included in more than one DataSet (e.g., common calibration files).

A DataSet lists three different pieces of information for each of its files, namely 1) the file name (including the path), 2) the file category, and 3) a string that is called the "purpose" of the file. The DO uses the OCA[10] rules to find the files to include in a DataSet, as well as their categories and purposes. The file category identifies different types of files, and it is derived by information in the header of the file itself. A category could for example be `RAW_CALIBRATION_1`, `RAW_CALIBRATION_2` or `RAW_SCIENCE`, depending on the instrument. The purpose of a file identifies the reason why a file is included in a DataSet. The syntax is `action_1/action_2/action_3/ ... /action_n`, where each `action_i` describes an intended processing step for this file (for example, creation of a `MASTER_CALIBRATION_1` or a `MASTER_CALIBRATION_2`). The actions are defined in the OCA rules and contain the recipe together with all file categories required to execute it (and predicted products in case of calibration data). For example, a workflow might include two actions `action_1` and `action_2`. The former creates `MASTER_ CALIBRATION_1` from `RAW_CALIBRATION_1`, and the later creates a `MASTER_CALIBRATION_2` from `RAW_CALIBRATION_2`. The `action_2` action needs `RAW_CALIBRATION_2` frames and the `MASTER_ CALIBRATION_1` as input. In this case, these `RAW_CALIBRATION_1` files will have the purpose `action_ 1/action_2`. The same DataSet might also include `RAW_CALIBRATION_1` with a different purpose; irrespective of their purpose the file category for all these biases will be `RAW_CALIBRATION_1`.

The Datasets created via the `DataOrganiser` will be displayed in the `DataSet Chooser`. Here the users have the possibility to inspect the various datasets and decide which one to reduce. By default, DataSets that have not been reduced before are highlighted for reduction. Click either `Continue` in order to continue with the workflow reduction, or `Stop` in order to stop the workflow. A full description of the `DataSet Chooser` is presented in Section 9.3.2.

Once the `Continue` is pressed, the workflow starts to reduce the first selected DataSet. Files are broadcasted according to their purpose to the relevant actors for processing.

The categories and purposes of raw files are set by the DO, whereas the categories and purpose of products generated by recipes are set by the `RecipeExecuter`. The file categories are used by the `FitsRouter` to send files to particular processing steps or branches of the workflow (see below). The purpose is used by the `SofSplitter` and `SofAccumulator` to generate input SoFs for the `RecipeExecuter`. The `SofSplitter` and `SofAccumulator` accept several SoFs as simultaneous input. The `SofAccumulator` creates a single output SoF from the inputs, whereas the `SofSplitter` creates a separate output SoF for each purpose.

9.3.2 DataSetChooser

The `DataSetChooser` displays the DataSets available in the "Select Data Sets" window, activating vertical and horizontal scroll bars if necessary (Fig. 6.3).

Some properties of the DataSets are displayed: the name, the number of files, a flag indicating if it has been

[10] OCA stands for OrganisationClassificationAssociation and refers to rules, which allow to classify the raw data according to the contents of the header keywords, organise them in appropriate groups for processing, and associate the required calibration data for processing. They can be found in the directory `<install_dir>/share/esopipes/<pipeline-version>/reflex/`, carrying the extension `.oca`. The variable `<install_dir>` depends on the operative system and installation procedure. For installation through `rpm`: `<install_dir>=/usr`; for installation through macport `<install_dir>=/opt/local`; for installation through the installation script install_esoreflex it depends on the path specified during installation, e.g. `<install_dir>=<specified_path>/install`

successfully reduced (a green OK), if the reduction attempts have failed or were aborted (a red FAILED), or if it is a new dataset (a black "-"). The column "Descriptions" lists user-provided descriptions (see below), other columns indicate the instrument set-up and a link to the night log.

Sometimes you will want to reduce a subset of these DataSets rather than all DataSets, and for this you may individually select (or de-select) DataSets for processing using the tick boxes in the first column, and the buttons `Deselect All` and `Select Complete` at the bottom, or configure the "Filter" field at the bottom left. Available filter options are: "New" (datasets not previously reduced will be selected), "Reduced" (datasets previously reduced will be selected), "All" (all datasets will be selected), and "Failed" (dataset with a failed or aborted reduction will be selected).

You may also highlight a single DataSet in blue by clicking on the relevant line. If you subsequently click on `Inspect Highlighted`, then a "Select Frames" window will appear that lists the set of files that make up the highlighted DataSet including the full filename[11], the file category (derived from the FITS header), and a selection tick box in the right column. The tick boxes allow you to edit the set of files in the DataSet which is useful if it is known that a certain calibration frame is of poor quality (e.g: a poor raw flat-field frame). The list of files in the DataSet may also be saved to disk as an `ASCII` file by clicking on `Save As` and using the file browser that appears.

By clicking on the line corresponding to a particular file in the "Select Frames" window, the file will be highlighted in blue, and the file FITS header will be displayed in the text box on the right, allowing a quick inspection of useful header keywords. If you then click on `Inspect`, the workflow will open the file in the selected FITS viewer application defined by the workflow parameter `FITS_VIEWER`.

To exit from the "Select Frames" window, click `Continue`.

To add a description of the reduction, press the button `...` associated with the field "Add description to the current execution of the workflow" at the bottom right of the Select Dataset Window; a pop up window will appear. Enter the desired description (e.g. "My first reduction attempt") and then press `OK`. In this way, all the datasets reduced in this execution, will be flagged with the input description. Description flags can be visualized in the `SelectFrames` window and in the `ProductExplorer`, and they can be used to identify different reduction strategies.

To exit from the "Select DataSets" window, click either `Continue` in order to continue with the workflow reduction, or `Stop` in order to stop the workflow.

9.3.3 The ProductExplorer

The ProductExplorer is an interactive component in the `esoreflex` workflow whose main purpose is to list the final products with the associated reduction tree for each dataset and for each reduction attempt (see Fig. 6.4).

Configuring the ProductExplorer

You can configure the ProductExplorer GUI to appear after or before the data reduction. In the latter case you can inspect products as reduction goes on.

1. To display the ProductExplorer GUI at the end of the datareduction:

[11] keep the mouse pointer on the file name to visualize the full path name.

- Click on the global parameter “ProductExplorerMode” before starting the data reduction. A configuration window will appear allowing you to set the execution mode of the Product Explorer. Valid options are:
  - "Triggered" (default). This option opens the ProductExplorer GUI when all the selected datasets have been reduced.
  - "Enabled". This option opens the ProductExplorer GUI at the end of the reduction of each individual dataset.
  - “Disable”. This option does not display the ProductExplorer GUI.
- Press the  button to start the workflow.

2. To display the ProductExplorer GUI “before” starting the data reduction:

- double click on the composite Actor "Inspect previously reduced data". A configuration window will appear. Set to "Yes" the field "Inspect previously reduced data (Yes/No)". Modify the field "Continue reduction after having inspected the previously reduced data? (Continue/Stop/Ask)". "Continue" will continue the workflow and trigger the DataOrganizer. "Stop" will stop the workflow; "Ask" will prompt another window deferring the decision whether continuing or not the reduction after having closed the Product Explorer.
- Press the  button to start the workflow. Now the ProductExplorer GUI will appear before starting the data organization and reduction.

Exploring the data reduction products

The left window of the ProductExplorer GUI shows the executions for all the datasets (see Fig. 6.4). Once you click on a dataset, you get the list of reduction attemps. Green and red flags identify successfull or unsucessfull reductions. Each reduction is linked to the “Description” tag assigned in the “Select Dataset” window.

1. To identify the desired reduction run via the “Description” tag, proceed as follows:

- Click on the symbol at the left of the dataset name. The full list of reduction attempts for that dataset will be listed. The column Exec indicates if the reduction was succesful (green flag: "OK") or not (red flag: "Failed").
- Click on the entries in the field "Description" to visualize the description you have entered associated to that dataset on the Select Dataset window when reducing the data.
- Identify the desired reduction run. All the products are listed in the central window, and they are organized following the data reduction cascade.

You can narrow down the range of datasets to search by configuring the field "Show" at the top-left side of the ProductExplorer (options are: "All", "Successful", "Unsuccessful"), and specifying the time range (Last, all, From-to).

2. To inspect the desired file, proceed as follows:

- Navigate through the data reduction cascade in the ProductExplorer by clicking on the files.
- Select the file to be inspected and click with the mouse right-hand button. The available options are:
  - Options available always:
    - Copy full path. It copies the full name of the file onto the clipboard. Shift+Ctr+v to past it into a terminal.
    - Inspect Generic. It opens the file with the fits viewer selected in the main workflow canvas.
    - Inspect with. It opens the file with an executable that can be specified (you have to provide the full path to the executable).
  - Options available for files in the `TMP_PRODUCTS_DIR` directory only:
    - command line. Copy of the environment configuration and recipe call used to generate that file.
    - Xterm. It opens an Xterm at the directory containing the file.
  - Options available for products associated to interactive windows only:
    - Display pipeline results. It opens the interactive windows associated to the recipe call that generated the file. Note that this is for visualization purposes only; the recipe parameters cannot be changed and the recipe cannot be re-run from this window.

9.3.4 Creation Of Master Calibration Files

In this step of the workflow, the following FORS2 recipes are executed in the order listed below. Please refer to the FORS2 pipeline user manual (FORS Pipeline User Manual: Sections 9 and 10) for the details of each recipe and the algorithms employed:

1. The `MasterBias` actor will execute the FORS2 pipeline recipe `fors_bias` in order to create a combined master bias frame from the set of raw bias frames
2. The `ForsPmosCalib` actor will execute the FORS2 pipeline recipe `fors_pmos_calib` in order to create from the set of raw flat and arc frames a combined master flat frame as well as coefficients for wavelength calibration and correction of spatial distortions.

9.3.5 Science Reduction

The `ForsPmosScience` actor will execute the FORS2 pipeline recipe `fors_pmos_science` to apply sky subtraction and extract the spectra. Please refer to the FORS2 pipeline user manual (FORS Pipeline User Manual: Sections 9 and 10) for the details of this recipe and the extraction algorithms employed.

9.3.6 Output Organisation

After having processed the input data for a DataSet, the workflow highlights and executes the `Product Renamer` actor, which, by default, will copy the defined final products of the `ForsScience` actor to the directory specified by `END_PRODUCTS_DIR` and rename them with names derived from the values of certain FITS

header keywords. Specifically, final products are renamed by default with names of the form `<HIERARCH.ESO.OBS.NAME>_<HIERARCH.ESO.PRO.CATG>.fits`, with `<HIERARCH.ESO.OBS.NAME>` and `<HIERARCH.ESO.PRO.CATG>` representing the values of the corresponding FITS header keywords (`<HIERARCH.ESO.OBS.NAME>` is the name of the OB and `<HIERARCH.ESO.PRO.CATG>` is the category of the product file). These names are fully configurable by right-clicking on the `Product Renamer` actor, selecting `Configure Actor`, and then editing the string as appropriate. In some cases the keyword `<HIERARCH.ESO.OBS.TARG.NAME>` (target name) may be more useful than `<HIERARCH.ESO.OBS.NAME>`.

For PMOS data the final products that are copied and renamed are (for better readbility we replace `<HIERARCH.ESO.OBS.NAME>` by `<OB_NAME>`):

- **1-dimensional extracted spectra** (`<OB_NAME>_REDUCED_*`, created only if spectra are identified and can be extracted).
  The individual spectra are provided as rows in a FITS file with as many extensions as input scientific exposures. The correspondence between these rows and the 2-dimensional frames and/or slit identifications can be obtained from `<OB_NAME>_OBJECT_TABLE_SCI_PMOS.fits`. All extracted spectra have the same format.
  - `<OB_NAME>_REDUCED_SCI_PMOS.fits` spectra
  - `<OB_NAME>_REDUCED_ERROR_SCI_PMOS.fits` 1 σ error of spectra
  - `<OB_NAME>_REDUCED_SKY_SCI_PMOS.fits` fitted sky spectra
- **1-dimensional polarization spectra** (same format as 1-dimensional extracted spectra)
  `X` may be any of `ANGLE, I, L, Q, U, V` according to the convention described in FORS Pipeline User Manual. **Please note that the polarization parameters `U`, `Q` and `ANGLE` are given with respect to the positive y-axis of the raw data. The transformation for non-zero position angles on-sky is described in Sect. 10.1.**
  - `<OB_NAME>_REDUCED_X_SCI_PMOS.fits` extracted polarisation signals from the object-sources.
  - `<OB_NAME>_REDUCED_ERROR_X_SCI_PMOS.fits` 1 σ errors of the extracted polarisation signals
- `<OB_NAME>_OBJECT_TABLE_POL_SCI_PMOS.fits` table with position information for detected spectra, with the information from the extensions included within one table.
- **2-dimensional wavelength calibrated and distortion corrected frames** (`<OB_NAME>_MAPPED_*`, with as many extensions as input scientific exposures)
  - `<OB_NAME>_MAPPED_ALL_SCI_PMOS.fits` frame without sky subtraction
  - `<OB_NAME>_MAPPED_SCI_PMOS.fits` frame, sky-subtracted
  - `<OB_NAME>_MAPPED_SKY_SCI_PMOS.fits` frame with fitted sky background

If **sky alignment** is requested (`skyalign`$\geq$0) the following products are provided in addition to the ones listed above:

- `<OB_NAME>_DISP_COEFF_SCI_PMOS.fits` adjustment of the input `DISP_COEFF_PMOS` table

- `<OB_NAME>_SKY_SHIFTS_SLIT_SCI_PMOS.fits` table with sky line shifts
- `<OB_NAME>_WAVELENGTH_MAP_SCI_PMOS.fits` wavelength map adjusted for sky line shifts

The following actors in this step of the workflow are concerned with the termination of the data flow for the current DataSet and will highlight briefly as they are executed.

Finally, the `Product Explorer` window will appear as shown in Fig. 6.4 with a list of datasets on the left menu. By unfolding the menu under each dataset, all the renamed products appear, and if one is interested in the files, including all intermediate steps, that are used to produce that final product, just click on it and a dependency tree will show the whole reduction chain.

9.4 Interactive Windows

The FORS2 PMOS workflow contains two interactive windows that allow the user to iterate on the processing of their data. They are described below. For troubleshooting and tips how to improve the results see Sect. 11. and 9.6, respectively.

9.4.1 fors_pmos_calib

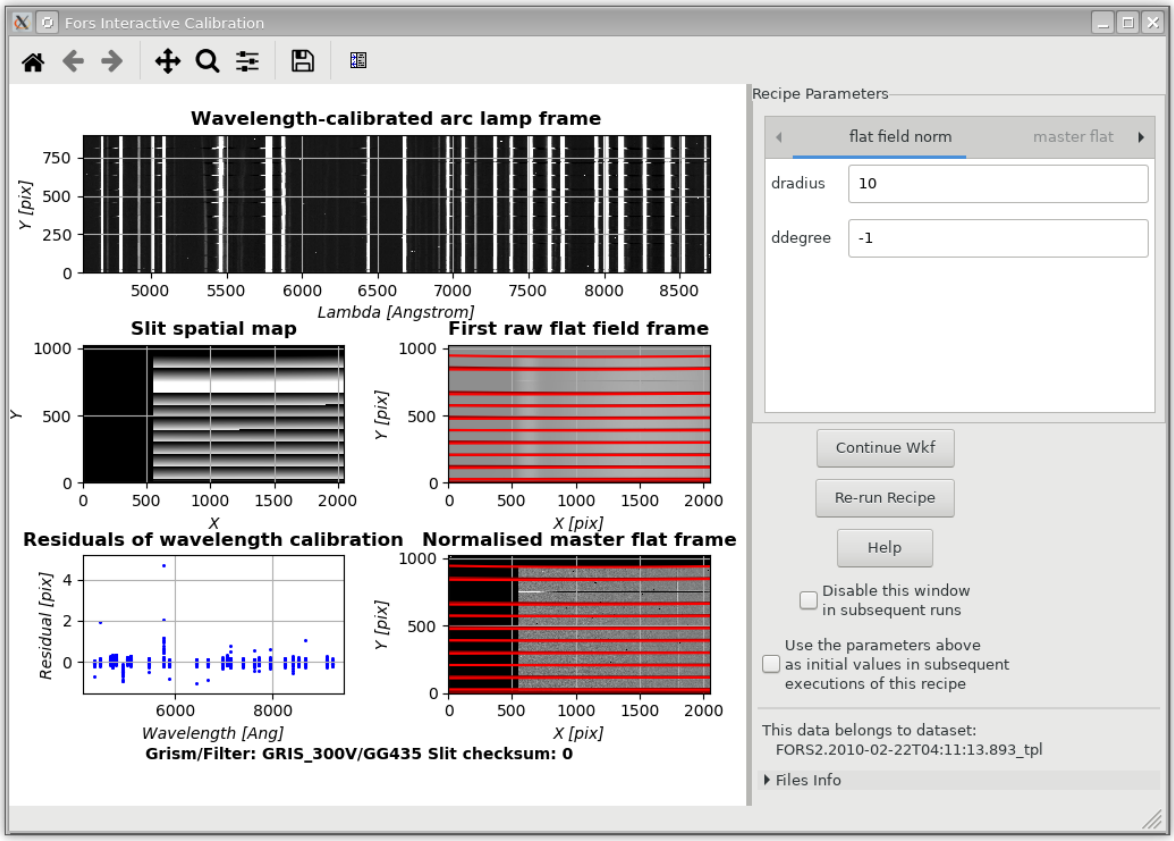


Figure 9.2: *The interactive window of the* `ForsPMOSCalib` *actor for the PMOS calibrations of the first demo DataSet.*.

The interactive window shown in Fig. 9.2 (p. 38) provides information about the quality of the wavelength calibration (top row and bottom left plot), and the flat field combination and normalization (central row, bottom right plot). The plots contain in detail:

Top *Wavelength-calibrated arc lamp frame*: In this plot the arc lamp lines should run straight from top to bottom without any empty rows between them. Particular attention should be given to lines at the blue and red ends of each spectrum, where the polynomial fit is more sensitive to small variations of the signal.

Some arc lines may show gaps due to the placement of the slits, but empty rows without any lines point towards problems with the detection of the arc lamp lines.

Center Left *Spatial map* The spatial map has as pixel values the distance of a pixel from the bottom of the respective slitlet. The regions of the slitlets should not be strongly curved nor should regions of different slitlets overlap with each other. Because all MOS slitlets have similar length the spatial map should show a similar gradient for all, which is not the case for the third slitlets from the top in Fig. 9.2. See Sect. 9.6 on how to fix such a problem.

Center Right *First raw flat* This plot is mostly of interest in comparison to the **Bottom Right** one, as the number of slitlets and the areas covered by them should be identical. The red lines show the traces of the slitlet edges. They should therefore follow the slit edges and not cut across slitlets. All slitlets should be detected and there should be no spurious detections (e.g. one slitlet detected as several) nor should several slitlets be detected only as one (as is the case in Fig. 9.2, see Sect. 9.6 on how to fix such a problem).

Bottom Left *Residuals between predicted and detected arc line position* The residuals should generally be below 0.5 pixel. If the scatter appears very large one should zoom in, because there are often only a few outliers and the majority of the residuals are within ± 0.5 pixels. If the residuals show systematic variations the polynomial degree used to fit the dispersion relation may be too low (or in rare cases too high). The plot shows all the residuals, regardless on whether they are rejected (and thus not used in the fit) or not.

Bottom Right *Normalized master flat* The normalized master flat field should have the same number of slitlets as the first raw flat and their areas should also be identical. If this is not the case see Sect. 9.6 on how to improve the result.

9.4.2 fors_pmos_science

The interactive window shown in Fig. 9.3 provides information about the quality of the sky subtraction and spectrum extraction. It shows the spectra in ADU/sec, i.e. not flux-calibrated, *and only the spectra from the first exposure*:

Top *Mapped sky-subtracted 2-dimensional spectrum*: The wavelength-calibrated, rectified frame (first exposure only) is shown after sky subtraction. The yellow and red lines mark the lower/upper extraction limits of the detected spectra. Right-clicking on such a range will plot the extracted spectrum in the **bottom** plot. Every object creates two spectra due to the splitting of the light by the Wollaston prism.

Bottom *Extracted science spectrum*: The spectrum should not show strong residuals of sky lines. The spectra should have a generally smooth look, and will only appear to be noisier in those regions where bright sky lines were subtracted.

The best way to ensure that the sky was subtracted optimally, at least at the positions of the objects to extract, is to check that the residual noise is compatible with the statistical error associated to the extracted object spectra. The extracted spectra are contained in the `REDUCED_SCI_PMOS` image (one extracted spectrum for each row). Their error spectra (at a 1-σ level) are contained in the `REDUCED_ERROR_SCI_PMOS` image. The regions of the extracted spectra corresponding to a (bright) sky line will include a few noisier points, whose deviation from the spectral continuum should (almost) never pass the 3-σ deviation. See

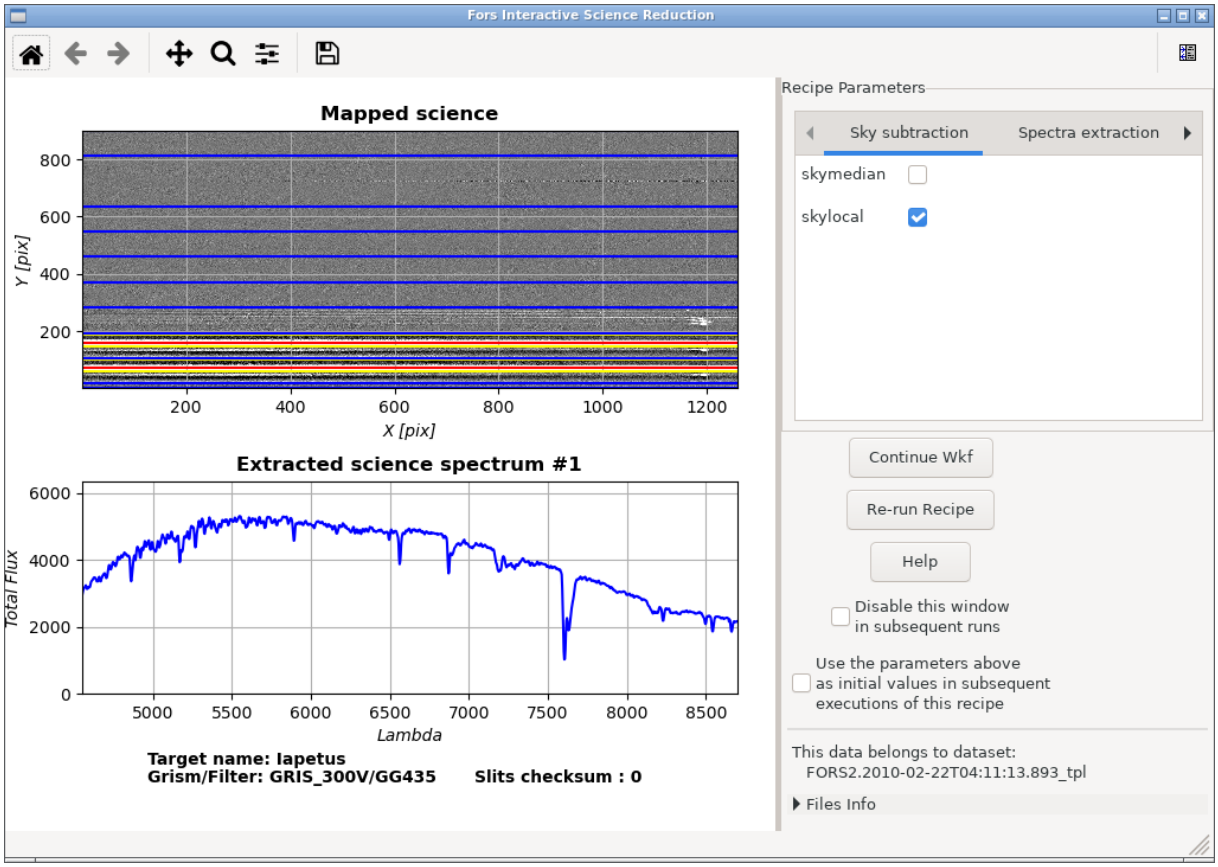


Figure 9.3: *The interactive window of the* `ForsPMOSScience` *actor for the first demo DataSet..*

Sect. 9.5.2 how to display this information with `python` scripts. If this condition is fulfilled, the sky subtraction is probably as good as it can get.

The sky spectra extracted from the modeled sky frames in exactly the same way as the object spectra from the sky-subtracted frames can be found in `REDUCED_SKY_SCI_PMOS`.

Note that if the barycentric correction is applied to the products, the wavelengths are re-computed to match the desired reference system. No interpolation is done on the spectrum itself, only wavelengths are changed. This means that spectra of the same target might be defined at different wavelengths, depending on the velocity correction. This has to be taken into account when combining spectra for further analysis.

9.5 Examining the results and calibration

The FORS PMOS pipeline records the 1-dimensional spectra as rows in an image. For the spectroscopic products these images have as many extensions as observed retarder plate angles (=exposures), and each exposure will create two extracted 1-dimensional spectra per object. The pipeline extracts any signal it finds, so there may be more spectra than intended targets. The extracted polarimetric parameter spectra are stored in single-extension FITS files, with one row per extracted polarimetric spectrum.

For a typical case of four exposures/retarder angles with only 1 bright target and no other objects the spectroscopic products will have four extensions with two rows each for the two spectra created by the Wollaston prism. The polarimetric products will have one extension with 1 row.

Because this format is by now unusual we provide here advice on how to inspect these products with the `ProductExplorer` (Sect. 9.5.1) or with (provided) `python` scripts (Sect. 9.5.2).

9.5.1 Verification with the `Product Explorer`

To help you verifying your results after running the workflow, we provide here a more in-depth description of how to use the `Product Explorer` for quality control.

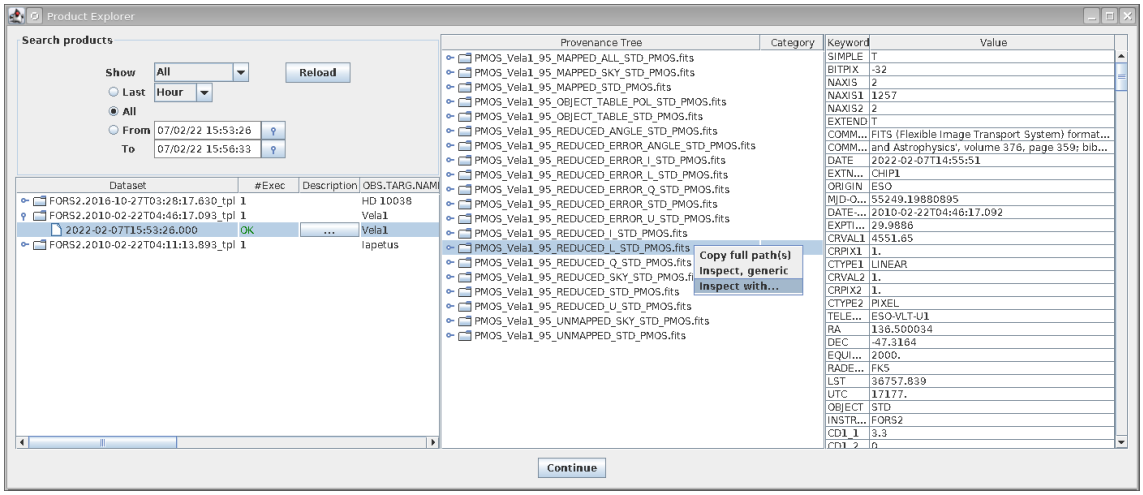


Figure 9.4: *The Product Explorer window. A contextual menu pops up when you right-click on a FITS file in the Provenance Tree.*.

When the `Product Explorer` window pops up (just rerun the workflow to create it again when you had closed it before), select the data set on the left and see the list of products in the `Provenance Tree` window in the middle.

In Fig. 9.6, you see a plot of the Stokes L (fractional degree of polarization) parameter obtained for the second demo data set produced by the `fv`[12] tool:

[12] `fv` is delivered as part of the `scisoft` package maintained by ESO and is also available from NASA

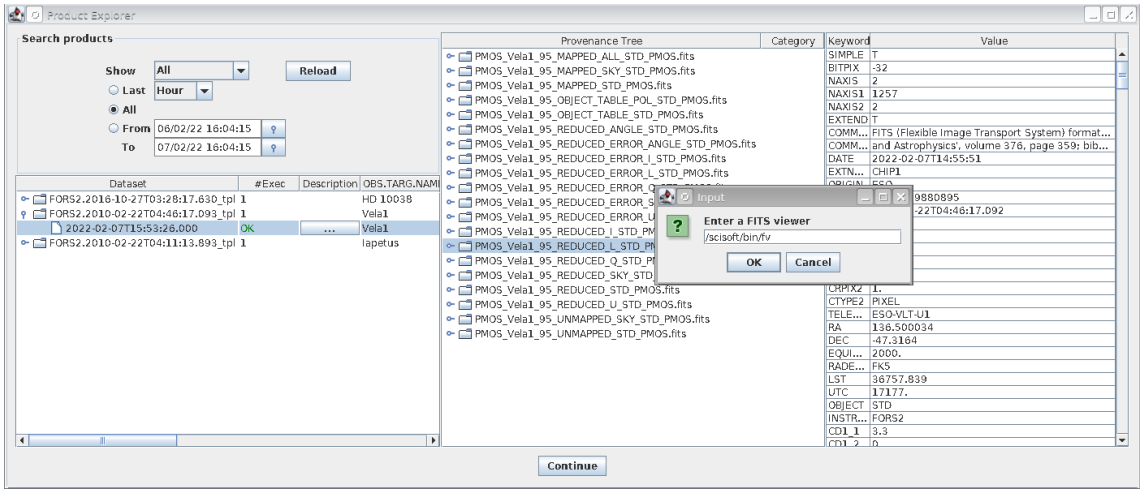


Figure 9.5: *The* `Product Explorer` *window. The small GUI in the right part pops up if the menu item labeled "Inspect with..." is clicked in the pop-up window that opens when you right-click on a selected file. In it, you can specify the path to your favourite FITS file viewer (e.g.,* `/scisoft/bin/fv`*). Alternatively, you can specify a default FITS viewer under the Global Parameters list of the Reflex workflow canvas.*

- open the file `PMOS_Vela1_95_REDUCED_L_STD_PMOS.fits` with `fv`
- in the window titled
  `fv:~Summary of PMOS_Vela1_95_REDUC...FORS2.2010-02-22T04:46:17.093_tpl/`
  click on `Image` to open the image display
- in the image display click on `Tools` → `DrawProfile`
- increase the size of the display window to see both the image and the profile
- left-click in the image display and draw the line across the image to get the profile (see Fig. 9.6).

.

The Stokes parameter `Q`, `U`, `V`, and `ANGLE` can be plotted in the same way.

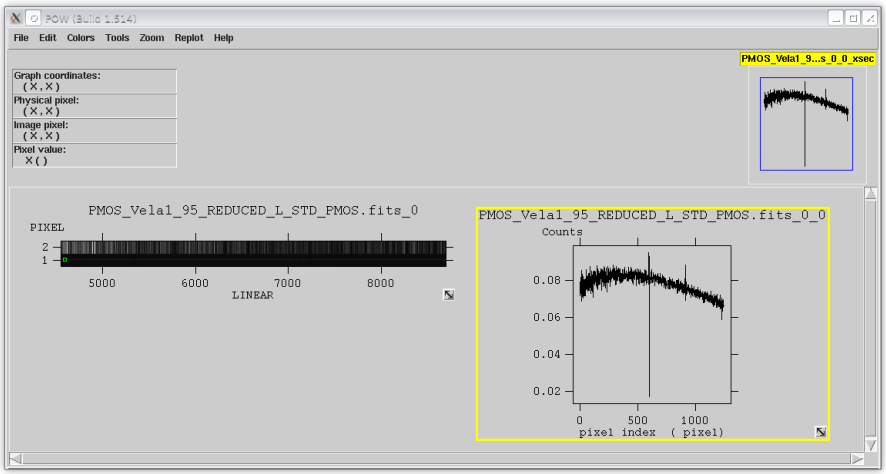


Figure 9.6: *The Stokes L (fractional polarization) spectrum of the polarized standard star Vela1-95.*

`fv` can also be used to inspect truly 2-dimensional images, like for instance `PMOS_Vela1_95_MAPPED_STD_PMOS`, which is a multi-extension FITS (MEF) file.

The number of extensions corresponds to the number of retarder angles recorded. Please note that there is no one-to-one relation between "extracted spectrum" and "object". For each object there will be two extracted spectra (one for each light beam). For instance, if a spectro-polarimetric observation consisted of four exposures at four different angles of the retarder plate, each object would have 4 angles x 2 beams = 8 spectra. Fig. 9.7 shows an example. Fig. 9.9 shows the inspection of a master bias.

- in the window titled
  `fv: Summary of PMOS_Vela1_95_MAPPE...FORS2.2010-02-22T04:46:17.093_tpl/`
  click on `Image` to open the image display for one of the 4 extensions
- in the image display click on `Edit` → `EditGraph` and select the tab `Image` in the pop-up window. There you can select a color table for the image and change the display cuts.

.

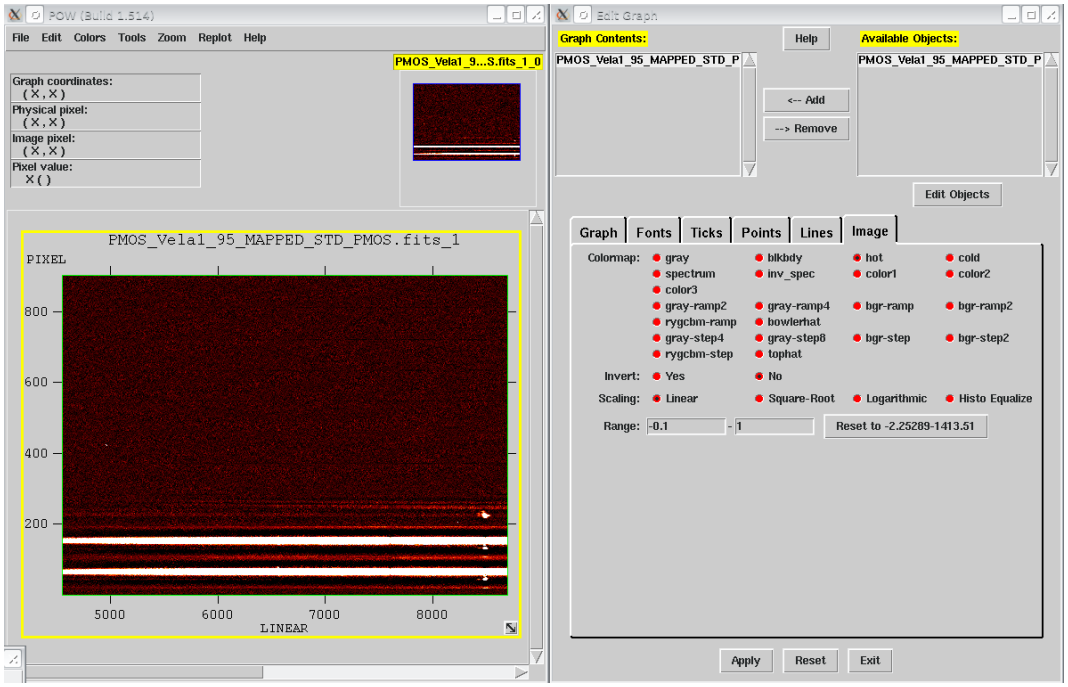


Figure 9.7: *The first extension of the wavelength calibrated, rectified and sky-subtracted image* `PMOS_Vela1_95_MAPPED_STD_PMOS.fits` *displayed with* `fv` *(left) and the* `fv` *window with the display parameters.*

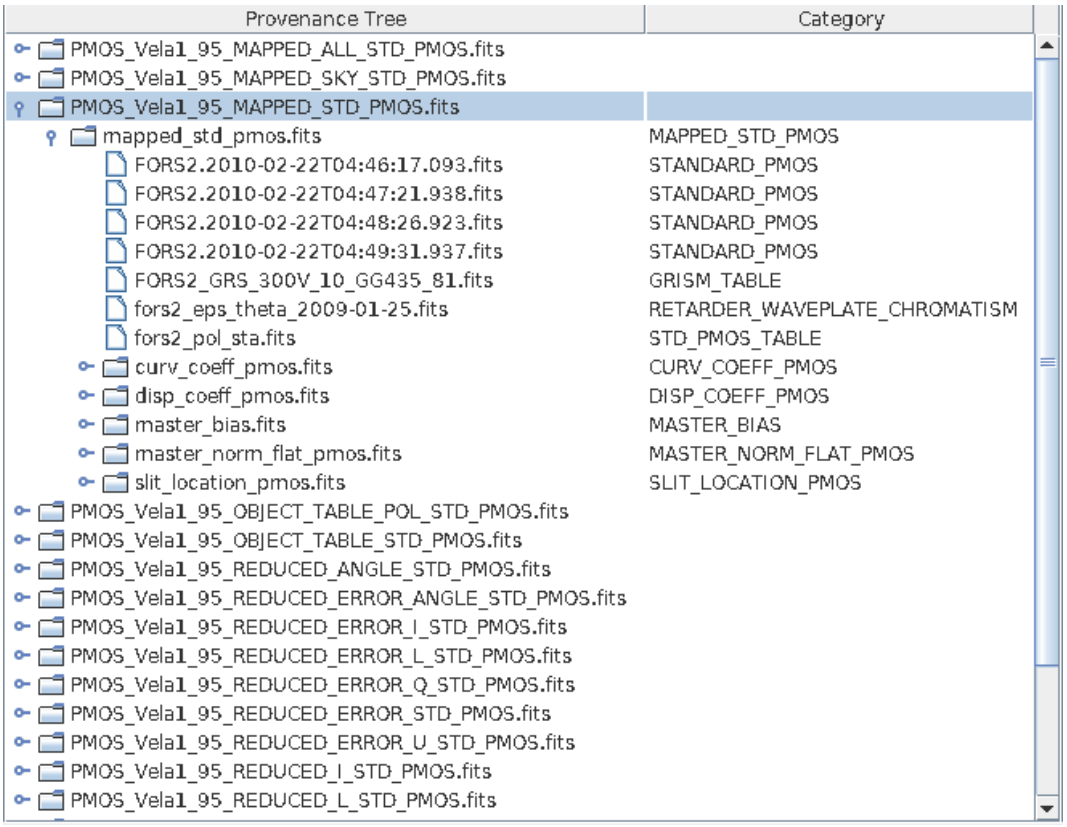


Figure 9.8: *The Product Explorer window showing the provenance tree expanded to show intermediate files created by recipes* `fors_pmos_calib`.

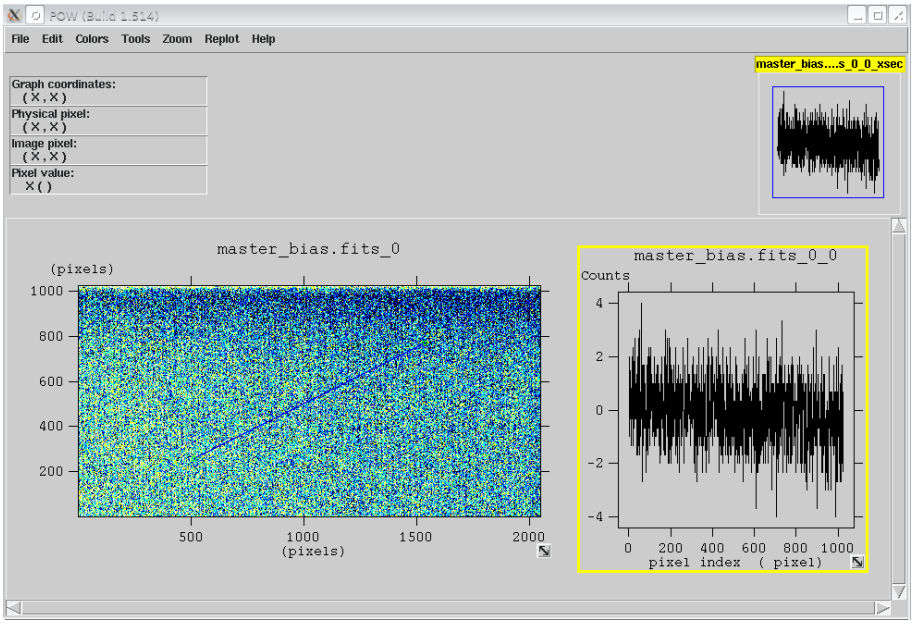


Figure 9.9: *The master bias frame and a profile cut. You can also perform some basic statistics by selecting* `Tools` → `Image Probe`..

9.5.2 Verification with `python` scripts

There are two `python` scripts available to inspect pipeline products, namely `plot_PMOS_pol.py` for polarimetric products (e.g. `REDUCED_L_SCI_PMOS`) and `plot_PMOS_reduced_MEF.py` for spectroscopic MEF products (e.g. `REDUCED_SCI_PMOS`). They are located in the same directory as the OCA rules file, which you can see if you double-click on the `Data Organizer` actor (`OCA File` in the pop-up window.

plot_PMOS_reduced_MEF.py <file1> <file2> <file3> <extension> <row> This script plots three extracted spectra recorded in the same extension and same row of that extension together. It sets the plots limits according to the flux level of the first spectrum. Fig. 9.10 shows the data for the second row in the first extension for the products of the first demo data set `REDUCED_SCI_PMOS` , `REDUCED_SKY_SCI_PMOS` , and `REDUCED_ERROR_SCI_PMOS` . The object has rather low signal, because it was observed by chance in the same slit as the intended target. The plot allows you to judge the effect and quality of the sky subtraction and to compare the error to the signal.

If you wish to plot spectra from fewer than three files put a question mark for <file2> and/or <file3>. By default the first row of the first extension is plotted. If you want to plot a different row you have to specify also the extension.

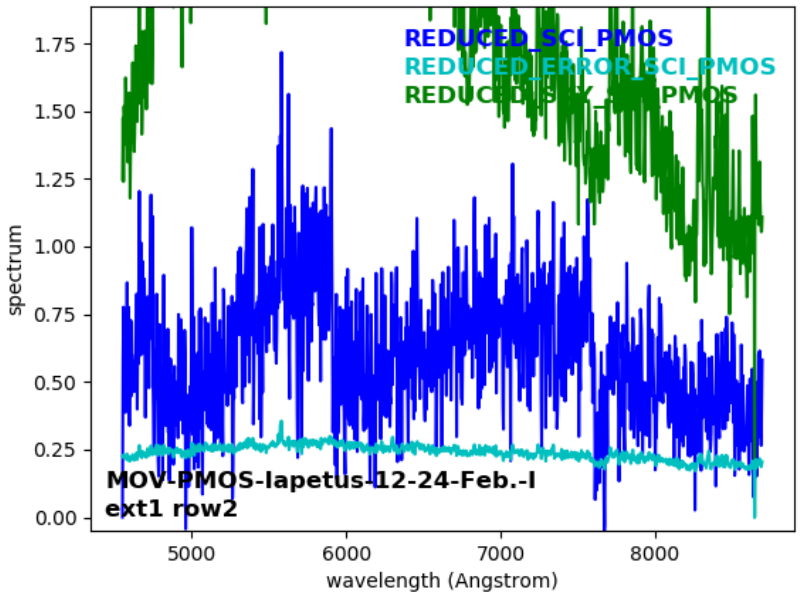


Figure 9.10: *This plots shows the extracted 1-dimensional spectrum (`REDUCED_SCI_PMOS`) of the second object in the first slit (`row 2`) in the first exposure (`ext 1`), its error (`REDUCED_ERROR_SCI_PMOS`), and the subtracted sky spectrum (`REDUCED_SKY_SCI_PMOS`).*

.

plot_PMOS_pol.py <file1> <file2> <row> This scripts plots two polarimetric spectra together, for instance

the linear polarization L and its error. By default the first row is plotted. Examples of such plots are shown in Figs. 9.11 to 9.16

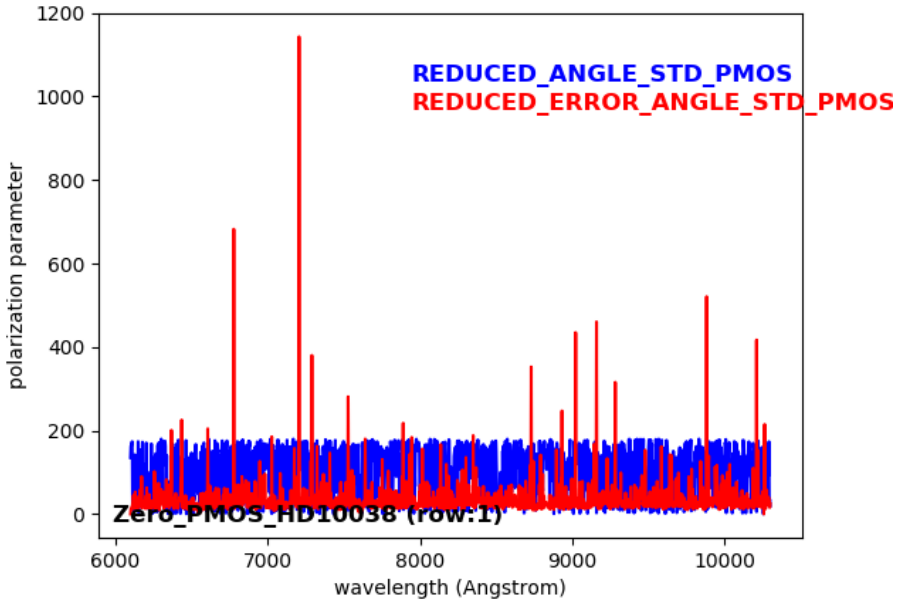


Figure 9.11: *The polarization angle spectrum of the unpolarized standard star HD10038.*

Should you find significant instrumental polarisation (i.e. polarization signal for an unpolarized standard star), subtract the Q/I and U/I values of a standard star observation in the $Q-U$ plane, where "the instrument polarisation is equivalent to a shift of the data points and the position angle zero point is equivalent to a rotation" (H.M. Schmid, "Polarimetry with ESO Instruments", in: "The 2007 ESO Instrument Calibration Workshop", Berlin: Springer, p. 499).

The number of extensions in a file corresponds to the number of retarder angles recorded. Please note that there is no one-to-one relation between "extracted spectrum" and "object". For each object there will be several extracted spectra, i.e., at least two for each exposure (one for each light beam). For instance, if a spectropolarimetric observation consisted of four exposures at four different angles of the retarder plate, each object would have 4 angles x 2 beams = 8 spectra.

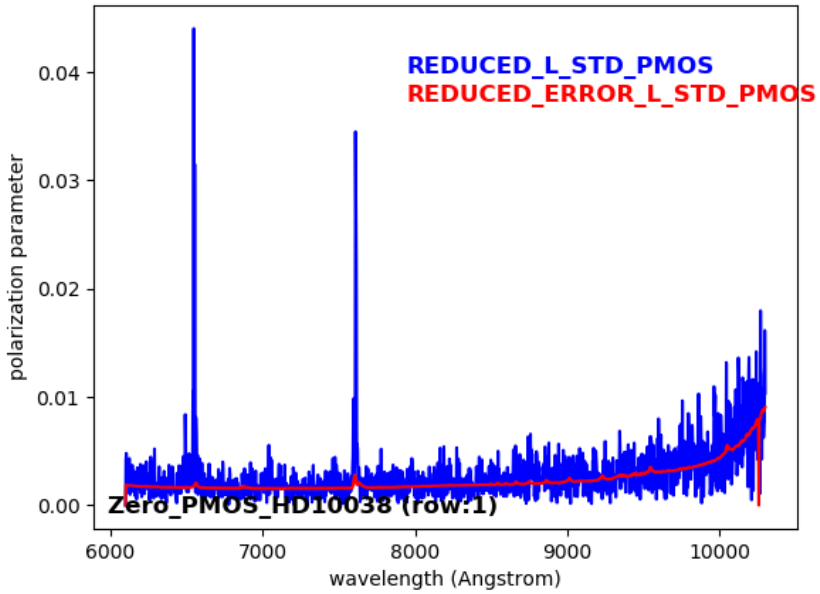


Figure 9.12: *The Stokes L (fractional polarization) spectrum of the unpolarized standard star HD10038.*

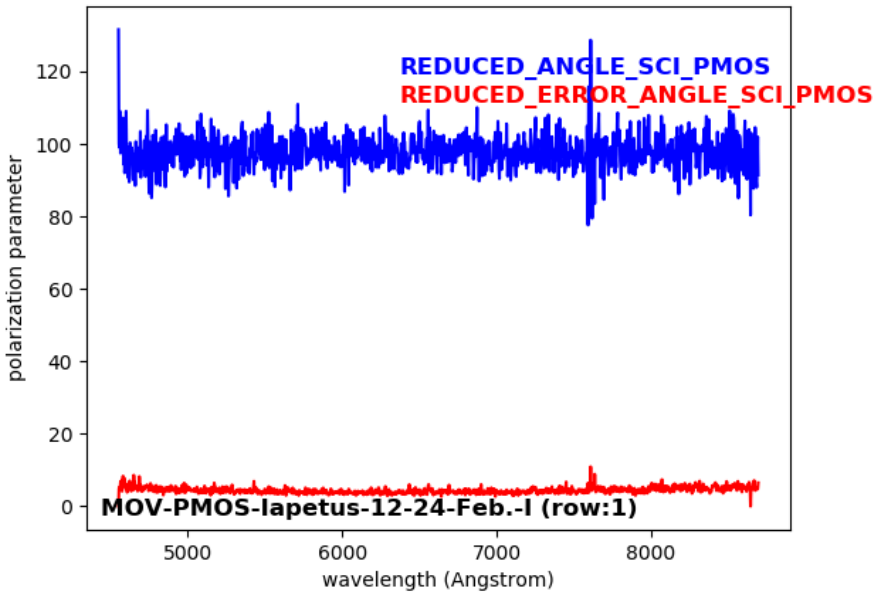


Figure 9.13: *The polarization angle spectrum of the science object Iapetus.*

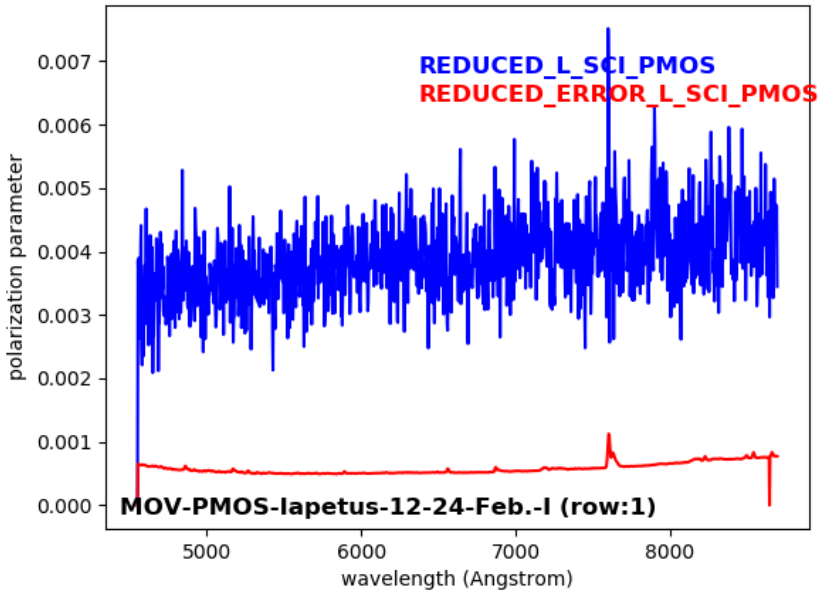


Figure 9.14: *The Stokes L (fractional polarization) spectrum of the science object Iapetus.*

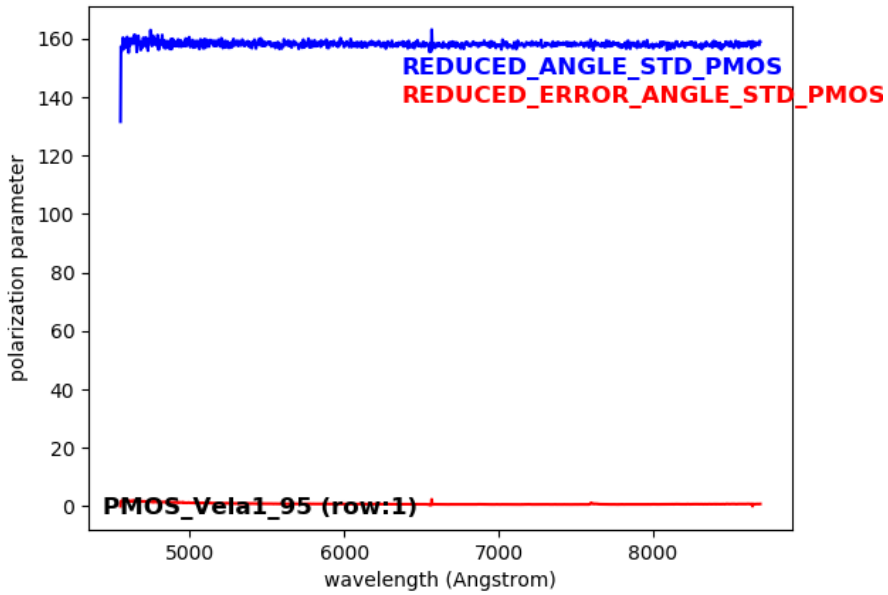


Figure 9.15: *The polarization angle spectrum of the polarized standard star Vela1-95.*

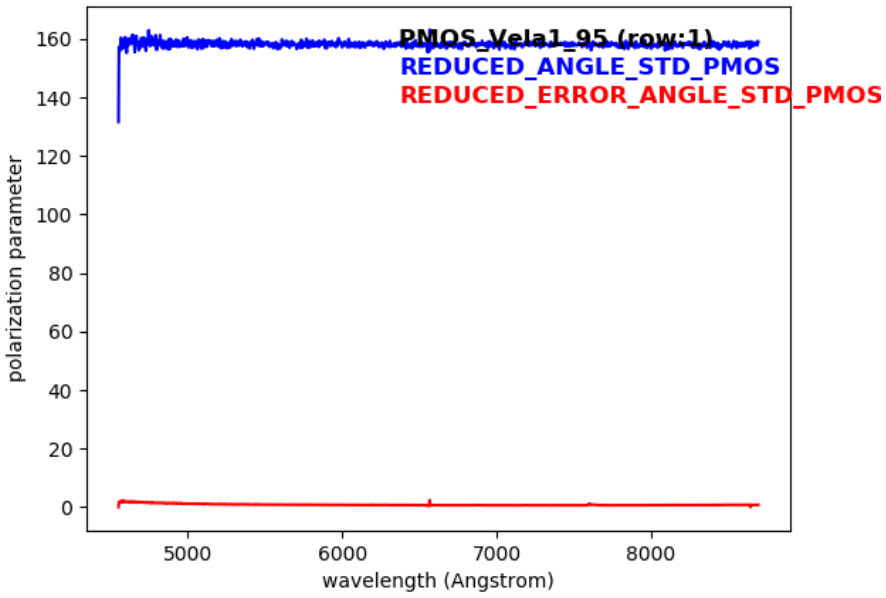


Figure 9.16: *The Stokes L (fractional polarization) spectrum of the polarized standard star Vela1-95.*

9.6 Improving Your Results

In this section we provide information on how to improve your results by changing the parameters of the `fors_pmos_calib` recipe.

`GRIS_300V` **demo data from 2010** The interactive window of the `ForsPMOSCalib` actor shows that two slits were not properly resolved. Increasing the `peakdetection` to `450` solves this problem (see Fig. 9.17).

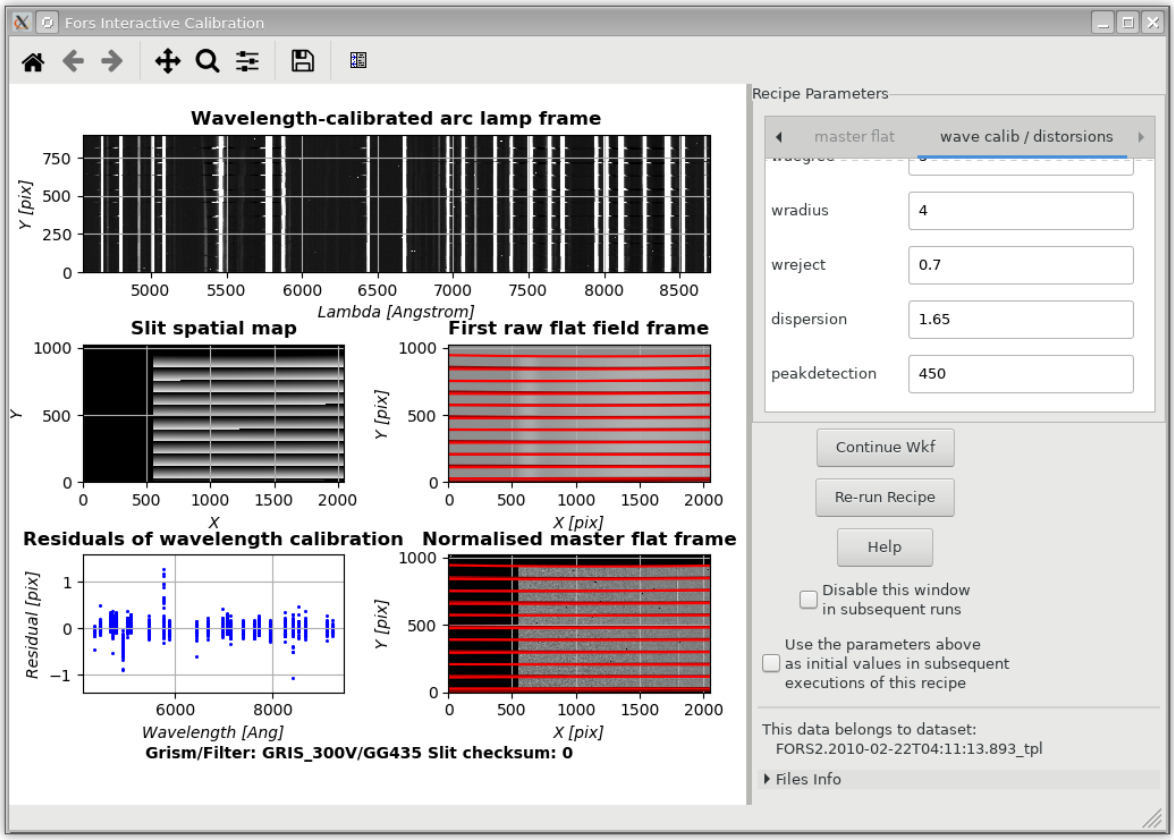


Figure 9.17: *The interactive window of the* `ForsPMOSCalib` *actor for the PMOS calibrations of the first demo DataSet after rerunning the* `fors_pmos_calib` *recipe with* `peakdetection` *changed to* `450`..

`GRIS_300I` **demo data from 2016** The interactive window for the `ForsPMOSCalib` actor shows that several slits were not properly traced (Fig. 9.18), including the ones with the object spectra.

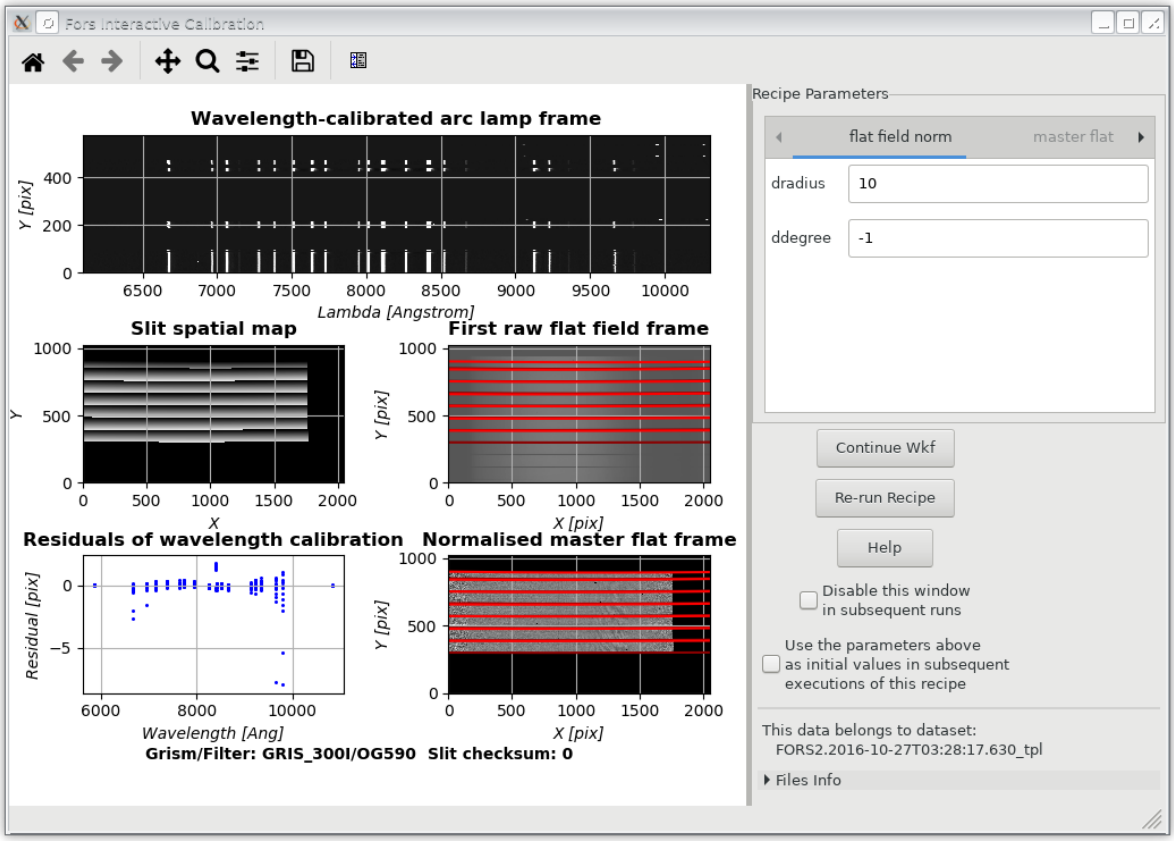


Figure 9.18: *The interactive window of the* `ForsPMOSCalib` *actor for the PMOS calibrations of the last demo DataSet.*

Increasing the tolerance for outliers `wreject` to `2` recoverse all slitlets, but leave some instabilities in the wavelength calibration (Fig. 9.19).

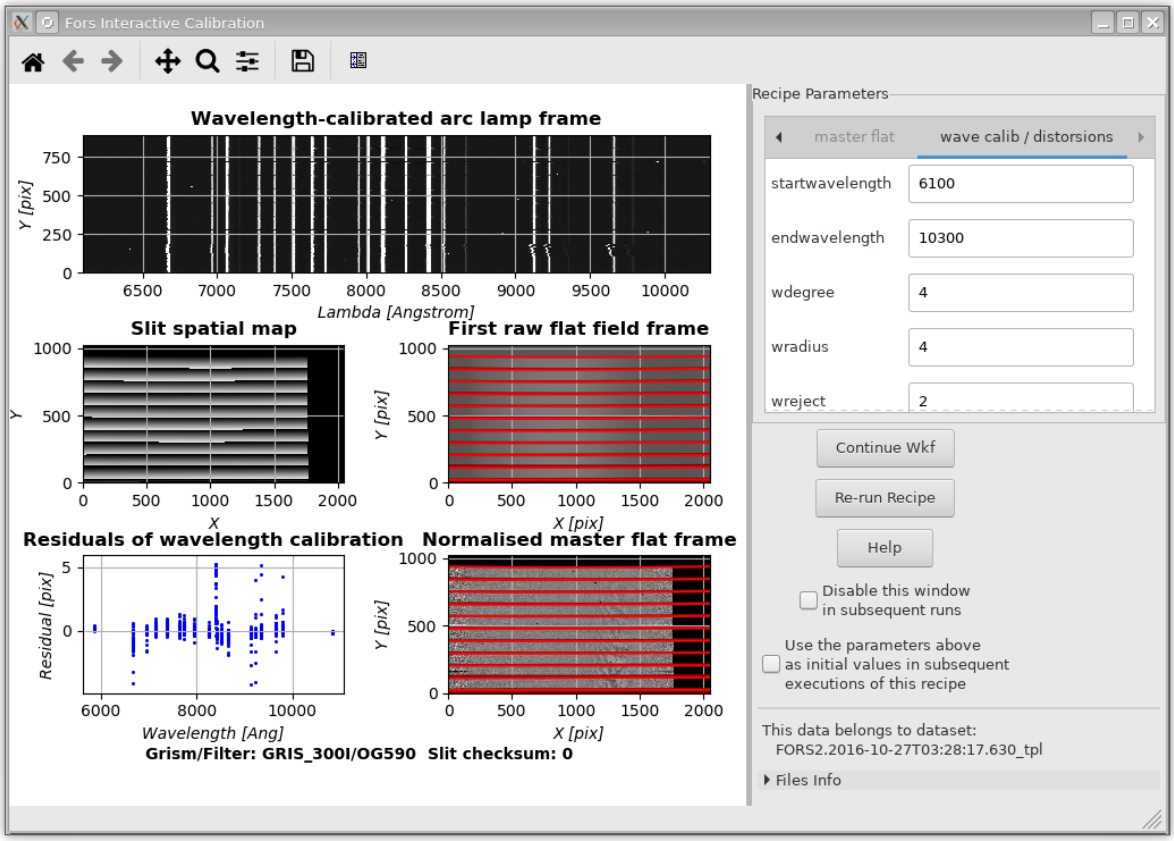


Figure 9.19: *The interactive window of the* `ForsPMOSCalib` *actor for the PMOS calibrations of the last demo DataSet with* `wreject` *set to* `2`..

The remaining instabilities in the wavelength calibrations can be fixed by increasing the search window for lines `wradius` to `12` (Fig. 9.20).

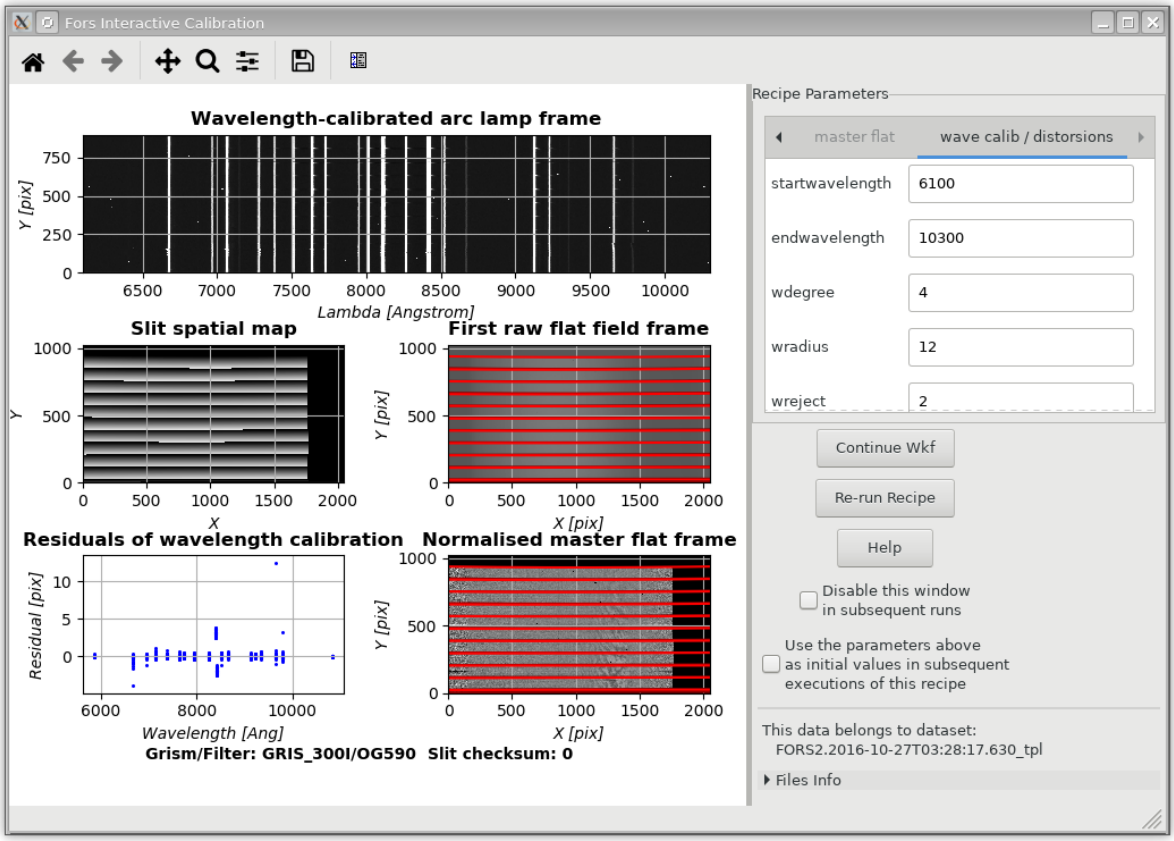


Figure 9.20: *The interactive window of the* `ForsPMOSCalib` *actor for the PMOS calibrations of the last demo DataSet with* `wreject` *set to* `2` *and* `wradius` *set to* `12`..

10 Frequently Asked Questions

- **The error window fills the whole screen - how can I get to the `Continue`/`Stop` buttons?**

  Press the Alt key together with your left mouse button to move the window upwards and to the left. At the bottom the `Continue`/`Stop` buttons will be visible. This bug is known but could not yet be fixed.

- **I tried to `Open` (or `Configure`) an `Actor` while the workflow is running and now it does not react any more. What should I do?**

  This is a limitation of the underlying Kepler engine. The only way out is to kill the workflow externally. If you want to change anything while a workflow is running you first need to pause it.

- **After a successful reduction of a data set, I changed this data set in some way (e.g. modified or removed some files, or changed the rules of the Data Organizer). When I restart Reflex, the Data Set Chooser correctly displays my new data set, but marks it as "reduced ok", even though it was never reduced before. What does this mean?**

  The labels in the column "Reduced" of the Data Set Chooser mark each dataset with "OK", "Failed" or "-". These labels indicate whether a data set has previously successfully been reduced at least once, all previous reductions failed, or a reduction has never been tried respectively. Data sets are identified by their name, which is derived from the first science file within the data set. As long as the data set name is preserved (i.e. the first science file in a data set has not changed), the Data Organizer will consider it to be the same data set. The Data Organizer recognizes any previous reductions of data sets it considers to be the same as the current one, and labels the current data set with "OK" if any of them was successful, even if the previously reduced data set differs from the current one.

  Note that the Product Explorer will list all the previous reductions of a particular data set only at the end of the reduction. This list might include successful and/or unsuccessful reduction runs with different parameters, or in your case with different input files. The important fact is that these are all reductions of data sets with the same first raw science file. By browsing through all reductions of a particular raw science file, the users can choose the one they want to use.

- **Where are my intermediate pipeline products?** Intermediate pipeline products are stored in the directory `<TMP_PRODUCTS_DIR>` (defined on the workflow canvas, under Setup Directories) and organised further in directories by pipeline recipe.

- **Can I use different sets of bias frames to calibrate my flat frames and science data?** Yes. In fact this is what is currently implemented in the workflow(s). Each file in a DataSet has a purpose attached to it (Forchì (2012)). It is this purpose that is used by the workflow to send the correct set of bias frames to the recipes for flat frame combination and science frame reduction, which may or may not be the same set of bias frames in each case.

- **Can I run `Reflex` from the command line?** Yes, use the command:

  ```
  esoreflex -n <workflow_path>/<workflow>.xml
  ```

  The -n option will set all the different options for Kepler and the workflows to avoid opening any GUI elements (including pipeline interactive windows).

  It is possible to specify workflow variables (those that appear in the workflow canvas) in the command line. For instance, the raw data directory can be set with this command:

```
esoreflex -n -RAW_DATA_DIR <raw_data_path> \
             <workflow_path>/<workflow>.xml
```

You can see all the command line options with the command `esoreflex -h`.

Note that this mode is not fully supported, and the user should be aware that the path to the workflow must be absolute and even if no GUI elements are shown, it still requires a connection to the window manager.

- **How can I add new actors to an existing workflow?** You can drag and drop the actors in the menu on the left of the `Reflex` canvas. Under `Eso-reflex -> Workflow` you may find all the actors relevant for pipeline workflows, with the exception of the recipe executer. This actor must be manually instantiated using `Tools -> Instantiate Component`. Fill in the "Class name" field with `org.eso.RecipeExecuter` and in the pop-up window choose the required recipe from the pull-down menu. To connect the ports of the actor, click on the source port, holding down the left mouse button, and release the mouse button over the destination port. Please consult the Reflex User Manual (Forchì (2012)) for more information.

- **How can I broadcast a result to different subsequent actors?** If the output port is a multi-port (filled in white), then you may have several relations from the port. However, if the port is a single port (filled in black), then you may use the black diamond from the toolbar. Make a relation from the output port to the diamond. Then make relations from the input ports to the diamond. Please note that you cannot click to start a relation from the diamond itself. Please consult the Reflex User Manual (Forchì (2012)) for more information.

- **How can I manually run the recipes executed by Reflex?** If a user wants to re-run a recipe on the command line he/she has to go to the appropriate reflex_book_keeping directory, which is generally reflex_book_keeping/<workflow>/<recipe_name>_<number> There, subdirectories exist with the time stamp of the recipe execution (e.g. 2013-01-25T12:33:53.926/). If the user wants to re-execute the most recent processing he/she should go to the `latest` directory and then execute the script `cmdline.sh`. Alternatively, to use a customized `esorex` command the user can execute

```
ESOREX_CONFIG="INSTALL_DIR/etc/esorex.rc"
PATH_TO/esorex --recipe-config=<recipe>.rc <recipe> data.sof
```

where INSTALL_DIR is the directory where Reflex and the pipelines were installed.

If a user wants to re-execute on the command line a recipe that used a specific raw frame, the way to find the proper data.sof in the bookkeeping directory is via `grep <raw_file> */data.sof`. Afterwards the procedure is the same as before.

If a recipe is re-executed with the command explained above, the products will appear in the directory from which the recipe is called, and not in the reflex_tmp_products or reflex_end_products directory, and they will not be renamed. This does not happen if you use the `cmdline.sh` script.

- **Can I reuse the bookkeeping directory created by previous versions of the pipeline?**

  In general no. In principle, it could be reused if no major changes were made to the pipeline. However there are situations in which a previously created bookkeeping directory will cause problems due to pipeline versions incompatibility. This is especially true if the parameters of the pipeline recipes have changed. In that case, please remove the bookkeeping directory completely.

- **How to insert negative values into a textbox?**

  Due to a bug in wxPython, the GUI might appear to freeze when attempting to enter a negative number in a parameter's value textbox. This can be worked around by navigating away to a different control in the GUI with a mouse click, and then navigating back to the original textbox. Once focus is back on the original textbox the contents should be selected and it should be possible to replace it with a valid value, by typing it in and pressing the enter key.

- **I've updated my Reflex installation and when I run esoreflex the process aborts. How can I fix this problem?**

  As indicated in Section 5, in case of major or minor (affecting the first two digit numbers) Reflex upgrades, the user should erase the `$HOME/KeplerData`, `$HOME/.kepler` directories if present, to prevent possible aborts (i.e. a hard crash) of the esoreflex process.

- **How can include my analysis scripts and algorithms into the workflow?**

  EsoReflex is capable of executing any user-provided script, if properly interfaced. The most convenient way to do it is through the Python actor. Please consult the tutorial on how to insert Python scripts into a workflow available here: `www.eso.org/sci/data-processing/Python_and_esoreflex.pdf`

10.1 FORS specific questions

- **There are no extracted spectra - why?**

  While this may happen for observations of extremely faint targets the most common case for this behaviour are PMOS observations from the lower chip, as the main target is usually placed close to the centre of the field-of-view, which is on the upper chip.

  This may for bright sources if `fors_pmos_calib` had problems. In such a case check the products of `fors_pmos_calib` in
  <data_wkf>`/reflex_tmp_products/fors-pmos/fors_pmos_calib_1/`<time-stamp>, especially `master_norm_flat_pmos.fits`, `spatial_map_pmos.fits`, and `reduced_lamp_pmos.fits` to verify if the slits have been properly detected and calibrated.

- **How can I determine polarization angle on sky?**

  The polarization parameters `U`, `Q` and `ANGLE` are given with respect to the positive y-axis of the raw data. If the position angle on sky is zero this corresponds to the standard north-south direction. For non-zero position angles on sky the user should apply the following transformation to obtain the true object polarisation angle:

  polarization angle = pipeline polarization angle − `HIERARCH ESO ADA POSANG` (position angle on sky, `POSANG` for short)

  This also changes `Q` and `U` (see below transformation equations).
  This is seen when one reduces PMOS polarimetric standard star data with the FORS2 pipeline, e.g. in the case of the star Vela X-1:

– For POSANG = 0, the extracted ANGLE = 172.22 (in agreement with tabulated values for Vela X-1), while Q = 0.077 and U = -0.021

– For POSANG = -30, ANGLE = 142.36, while Q = 0.02 and U = -0.077

The following equations define how Q and U change when a position angle not equal to zero is used:

$$new\_\mathtt{Q} = original\_\mathtt{Q} \times \cos(2 \times -\mathtt{POSANG}) + original\_\mathtt{U} \times \sin(2 \times -\mathtt{POSANG}) \quad (1)$$

$$new\_\mathtt{U} = -original\_\mathtt{Q} \times \sin(2 \times -\mathtt{POSANG}) + original\_\mathtt{U} \times \cos(2 \times -\mathtt{POSANG}) \quad (2)$$

- **My reduced spectra do not cover the wavelength range I expected**

  Some grisms (e.g., GRIS_300V) are subject to second order contamination unless they are used with an order separation filter (e.g., GG435). The pipeline will limit the wavelength range of the reduced spectra to the range not affected by the overlap if the filter was not used. The user can change the wavelength range in the interactive window of the `fors_pmos_calib` actor or by editing the startwavelength and/or endwavelength values of the file classified as "GRISM_TABLE". Alternatively these parameters can be specified on the command line.

- **Does the pipeline combine different detectors chips into a common product?**

  No. The spectroscopic pipeline and Reflex workflow works only on files from the same detector chip. Files from different detectors must not be mixed in the same sof.

11 Troubleshooting

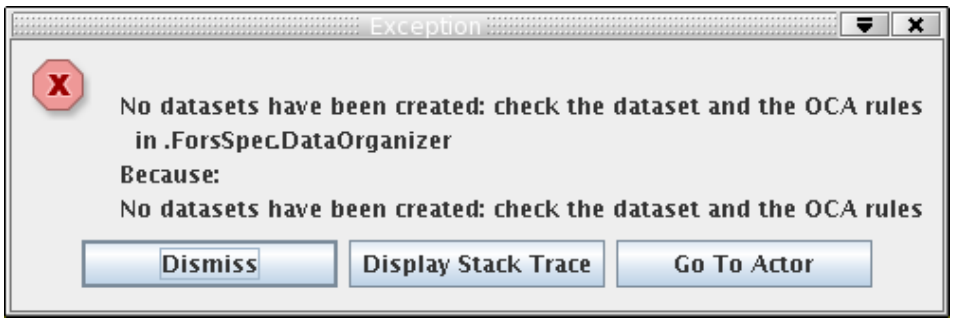


Figure 11.1: *TheDataOrganizer interactive window reports an error ":No DataSets have been created, check the data set and the OCA rules.".*

1. **I downloaded the data from the ESO archive, put them into a new directory, tried to run** `Reflex` **on them, but**

   (a) **it crashes**

   The current release of FORS includes some additional data in the static calibration frames. The recipes would chock if this data is not present. However, the ESO archive with CalSelector will associate calibration data which is old and Reflex will pick the files either from the installed pipeline static data or from the CalSelector in a non-deterministic way. In order to solve the issue, remove the static calibration data downloaded from the archive (all the files starting with M.FORS2).

   This may happen if one of the files was downloaded only partially (check for a file with the extension `fits.Z.part`. You will have to download that file again in order to have an uncorrrupted file (and remove the partial one).

   (b) **The DataOrganiser fails with the error message ":No DataSets have been created, check the data set and the OCA rules."(see Figure 11.1.)**

   This error may be due to the fact that the data provided by the ESO archive are compressed (`<filename>.fits.Z`). Please remember to uncompress the data before running the workflow in `Reflex`.

   Also, please remember that the FORS2 workflow supports only spectro-polarimetric data (PMOS). It is possible that your data consists entirely of IMG/IPOL/LSS/MOS/MXU observations, in which case the `Data Organiser` actor will not construct any DataSets, showing the mentioned error message.

2. **The "Select DataSets" window displays my DataSets, but some/all of them are greyed out. What is going on?**

   If a DataSet in the "Select DataSets" window is greyed out, then it means that the DataSet which was constructed is missing some key calibration(s) (i.e. the DataSet is incomplete). To find out what calibration(s) are missing from a greyed out DataSet, click on the DataSet in question to highlight it in blue, and

then click on the button `Inspect Highlighted`. The "Select Frames" window that appears will report the category of the calibration products that are missing (e.g. `MASTER_BIAS`). From this the user has then to determine the missing raw data (in this case bias frames). If static calibrations are missing the mechanism unfortunately does not work, but such data should be found by `reflex` in `<install_directory>/calib/<pipeline_version>/cal`

3. **The pipeline fails with the error message "The wavelength solution at row <number> does not increase monotonically, which is physically impossible. Try with new parameters"**.

   Non-monotonic dispersion relations are often due to spurious detections. In such cases try to decrease `wdegree` and/or increase `peakdetection`. A further possibility is to decrease `wradius`.

Acknowledgements

The Reflex team in alphabetical order consists of Pascal Ballester, Lodovico Coccato, Vincenzo Forchì, Wolfram Freudling, César Enrique García, Maurice Klein Gebbinck, Martino Romaniello.

Forchì V., 2012, Reflex User Manual, VLT-MAN-ESO-19000-5037, Issue 0.7, *https://ftp.eso.org/pub/dfs/reflex/ReflexUserManual-3.1.pdf* 23, 28, 56, 57

Bagnulo S., et al., 2009, PASP, 121, 993 7

FORS User Manual, VLT-MAN-ESO-13100-1543
https://www.eso.org/sci/facilities/paranal/instruments/fors/doc.html 14

FORS Pipeline User Manual, VLT-MAN-ESO-19500-4106
https://www.eso.org/sci/software/pipelines/index.html 9, 11, 14, 28, 35, 36