# 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

## Reflex ESPRESSO Tutorial

ESO-331896

Issue 3.3.22

Date 2025-11-04

| | Name | Date | Signature |
|---|---|---|---|
| Prepared: | Andrea Modigliani | 2025-11-04 | |
| Approved: | | | |
| Released: | | | |

This page was intentionally left blank

**Change record**

| Issue/Rev. | Date | Section/Parag. affected | Reason/Initiation/Documents/Remarks |
|---|---|---|---|
| 1.0.0 | 10/12/2018 | all | First public release |
| 1.2.2 | 25/03/2019 | all | Update public release |
| 2.3.0 | 03/03/2021 | all | cosmic ray detection |
| 2.4.0 | 11/07/2022 | all | Public releases group update |
| 3.3.22 | 04/11/2025 | — | Public releases update |

This page was intentionally left blank

# Contents

**1 Introduction to `Esoreflex`** **7**

**2 Software Installation** **8**

2.1 Installing `Esoreflex` workflows via `macports` . . . . . 8

2.2 Installing `Esoreflex` workflows via `rpm/yum/dnf` . . . . . 8

2.3 Installing `Esoreflex` workflows via `install_esoreflex` . . . . . 8

2.4 Demo Data . . . . . 9

**3 Quick Start: Reducing The Demo Data** **10**

**4 About the main `esoreflex` canvas** **14**

4.1 Saving And Loading Workflows . . . . . 14

4.2 Buttons . . . . . 14

4.3 Workflow States . . . . . 14

**5 The ESPRESSO Workflow** **15**

5.1 Workflow Canvas Parameters . . . . . 15

5.2 Workflow Actors . . . . . 16

5.2.1 Simple Actors . . . . . 16

5.2.2 Lazy Mode . . . . . 16

**6 Reducing your own data** **18**

6.1 The `esoreflex` command . . . . . 18

6.2 Launching the workflow . . . . . 18

6.3 Workflow Steps . . . . . 20

6.3.1 Data Organisation And Selection . . . . . 20

6.3.2 DataSetChooser . . . . . 21

6.3.3 Recipe execution . . . . . 22

6.3.4 Final products . . . . . 23

6.3.5 The ProductExplorer . . . . . 24

**7 Optimizing results through workflow Interactivity** **27**

7.1 Master Bias . . . 27

7.2 Master Dark . . . 28

7.3 Led FF . . . 29

7.4 Orderdef . . . 30

7.5 Master Flat . . . 30

7.6 Wavelength Calibration: processing of frames where both fibres are illuminated by a Fabry Perot 31

7.7 Wavelength Calibration: processing of FP,THAR or THAR,FP data . . . 32

7.8 Cross fibre contamination determination . . . 34

7.9 Relative Fibre efficiency determination . . . 36

7.10 Flux calibration . . . 37

7.11 Science Data Reduction . . . 38

7.11.1 Cosmic rays cleaning . . . 40

7.11.2 Radial Velocity . . . 41

**8 Frequently Asked Questions** **42**

**9 Troubleshooting** **45**

**A Useful information** **49**

A.1 Reflex directory naming conventions . . . 49

A.2 Available Organisation Classification and Association (OCA) rules . . . 49

A.3 Glossary . . . 50

A.3.1 espresso data reduction naming convention . . . 50

A.3.2 reflex workflow acronims . . . 51

# 1 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[1] (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.

This tutorial deals with the reduction of ESPRESSO observations only via the ESPRESSO workflow. The user is referred to the `ESPRESSO web page` for more information on the instrument itself, and the ESPRESSO pipeline user manual distributed on the `ESO-VLT Instrument pipelines home page` for the details of the pipeline recipes.

To run the ESPRESSO pipeline please take into account that:

- Since the size of the raw data and individual data products can be up to 700 MB and the full reduction of a SINGLEHR11 mode data set generates around 25 GB of data, we recommend the user to have enough disk space, at least a few TB to properly handle recipe I/O. In case of installation done with the `install_esoreflex` script, if the user decides to install also the demo data, we recommend the user to install the reflex demo on a large disk area.

  This release parallelises some basic data reduction step with OpenMP. Thus the user may benefit of execution on computers with several cores.

- The user needs at least 16 GB RAM, but more RAM (e.g. 32 MB) is recommendable. Reduction is faster if the directory where reflex is executed and the input data are on a Solid State Disk (SSD) disk.

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

# 2 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 pipelines and Reflex can be installed from the ESO `macports` repositories that support macOS platforms, the and the `rpm/yum` repositories that support Fedora and CentOS platforms. 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.

## 2.1 Installing `Esoreflex` 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*, which also describes the versions of macOS that are currently supported.

## 2.2 Installing `Esoreflex` workflows via `rpm/yum/dnf`

This method is supported for Fedora and CentOS platforms and requires sudo rights. Please read the full documentation at *http://www.eso.org/sci/software/pipelines/installation/rpm.html*, which also describes the versions of Fedora and CentOS that are currently supported.

## 2.3 Installing `Esoreflex` 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 2.11.5` may be found at:
*http://www.eso.org/sci/software/pipelines/reflex_workflows*

To install the `Reflex 2.11.5` software and demo data, please follow these instructions:

1. From any directory, download the installation script:

```
wget https://eso.org/sci/software/pipelines/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.

## 2.4 Demo Data

Together with the pipeline the user will also receive a demo data set, that allows you to run the `Reflex ESPRESSO` workflow without any changes in parameters. This way you have a data set to perform a run-time verification after the installation and to experiment with before you start to work on your own data.

Note that you will need a minimum of ∼9.5 GB, ∼0.887GB and ∼25 GB of free disk space for the directories `<download_dir>`, `<install_dir>` and `<data_dir>`, respectively.

# 3 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 ESPRESSO demo data set supplied with the `esoreflex 2.11.5` 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 `espresso` by typing:

   ```
   esoreflex espresso&
   ```

   Alternatively, you can type only the command `esoreflex` the empty canvas will appear (Figure 3.0.1) 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 `espresso` workflow is shown in Figure 3.0.2.

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 3.0.3) that lists the datasets along with the values of a selection of useful header keywords[2]. 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 6.3.2.

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 3.0.4 shows the Product Explorer window. A full description of the *Product Explorer* will be presented in Section 6.3.5.

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 ESPRESSO workflow that merit a look at the rest of this tutorial.

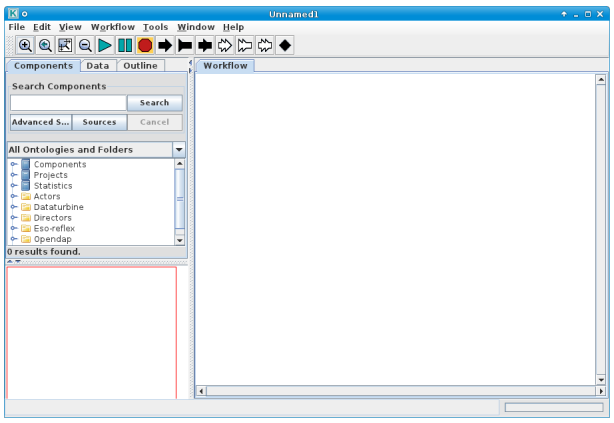



Figure 3.0.1: *The empty* `Reflex` *canvas.*

[2]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.

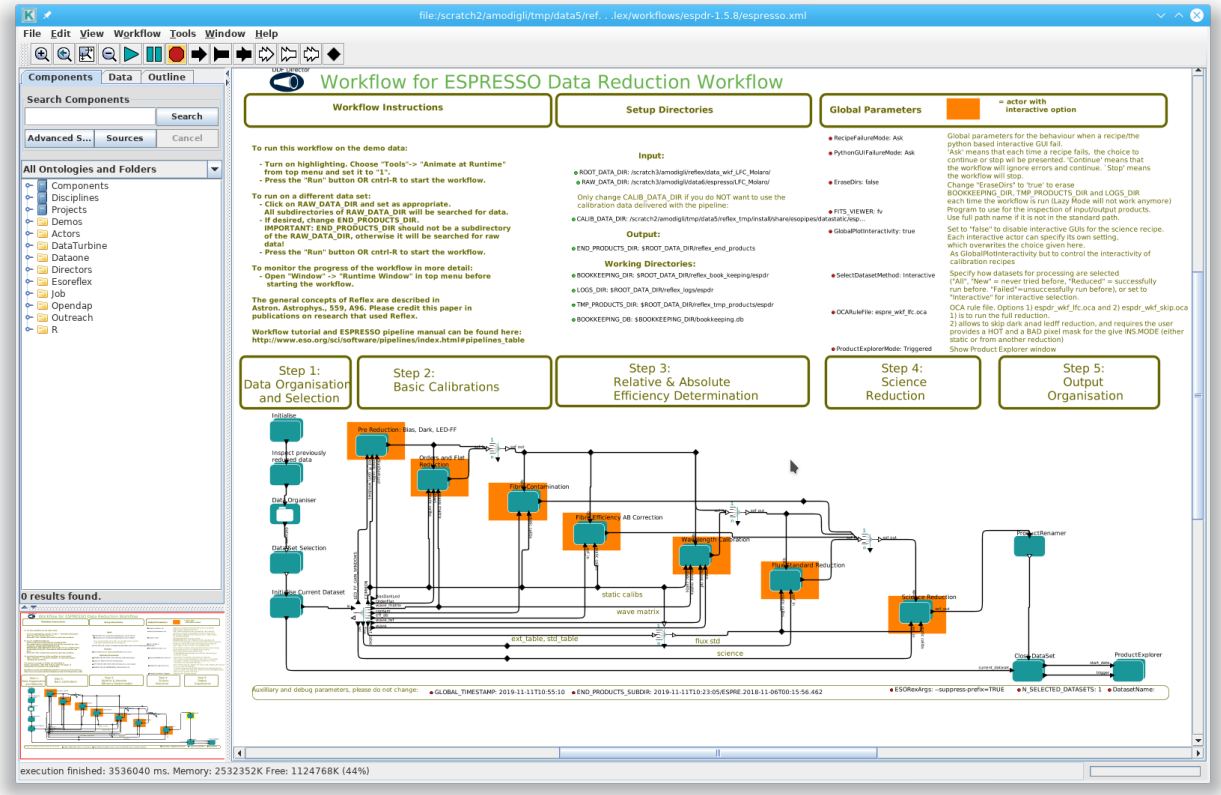



Figure 3.0.2: *ESPRESSO workflow general layout.*

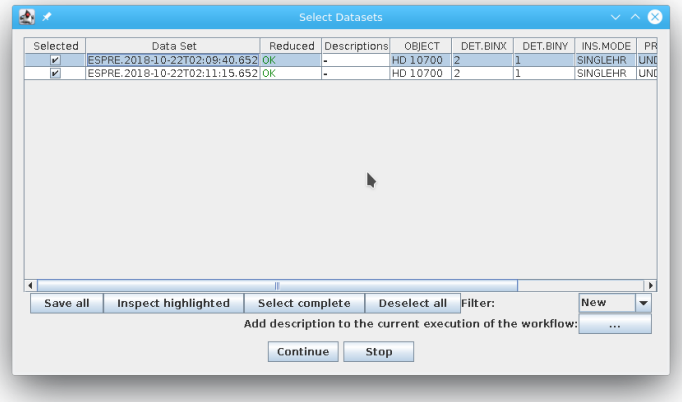



Figure 3.0.3: *The "Select Datasets" pop-up window.*

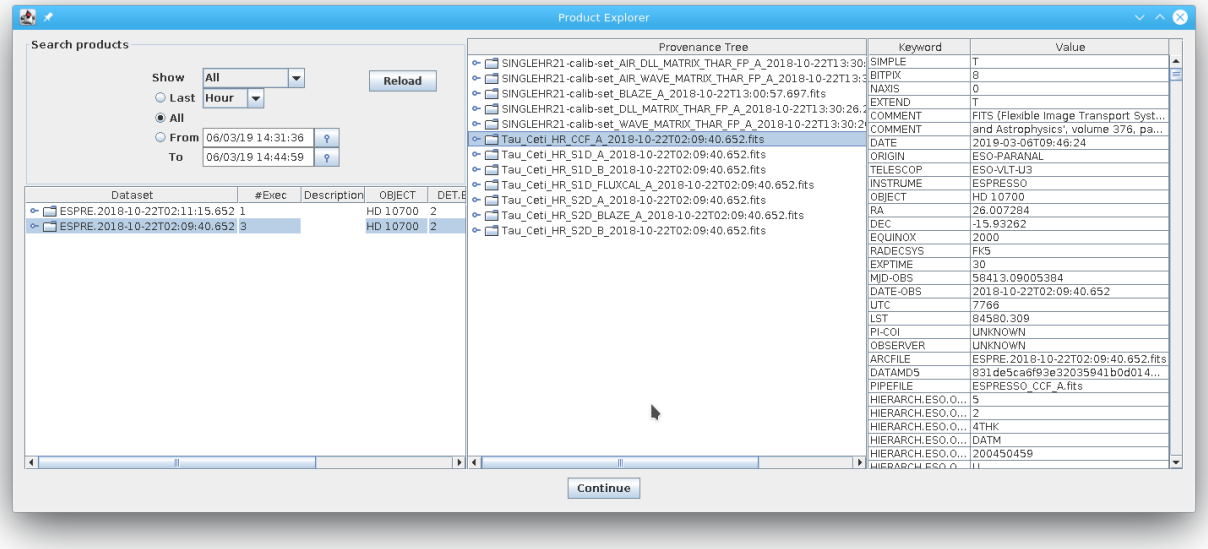



Figure 3.0.4: *The Provenance Explorer shows all datasets reduced in previous executions together with the full reduction chain for all the pipeline products.*

# 4 About the main `esoreflex` canvas

## 4.1 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.

## 4.2 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.

## 4.3 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.

# 5 The ESPRESSO Workflow

The ESPRESSO 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.

## 5.1 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 (2018)).

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 5.2.2), 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 `GlobalPlotInteractivity` controls whether the interactive windows will appear for those windows which are *enabled* by default. The possible values are `true, false`. Take into account that some

windows are disabled in the default configuration and therefore are not affected by this parameter.

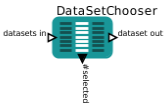
The parameter `ProductExplorerMode` controls whether the `ProductExplorer` actor will show its window or not. The possible values are `Enabled`, `Triggered`, and `Disabled`. `Enabled` opens the ProductExplorer GUI at the end of the reduction of each individual dataset. `Triggered` (default and recommended) opens the ProductExplorer GUI when all the selected datasets have been reduced. `Disabled` does not display the ProductExplorer GUI.

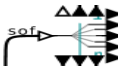
## 5.2 Workflow Actors

### 5.2.1 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).

### 5.2.2 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.*

# 6 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 3), but it is not a requirement.

## 6.1 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.
```

## 6.2 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> espresso` 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
  espresso
```

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 espresso`. 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 espresso
   ```

   to launch the Product Explorer. The Product Explorer allows you to inspect the data products already reduced by the `espresso 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> espresso
   ```

   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 6.3.5

3. Type:

   ```
   esoreflex espresso &
   ```

   to launch the `espresso esoreflex` workflow. The `espresso` workflow window will appear (Fig. 3.0.2). 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 5.1).

Secondly, if set, the workflow will open the Product Explorer, allowing the user to inspect previously reduced datasets (see Section 6.3.5 for how to configure this option).

## 6.3 Workflow Steps

### 6.3.1 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[3] 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 6.3.2.

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.

[3] 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`

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.

### 6.3.2 DataSetChooser

The `DataSetChooser` displays the DataSets available in the “Select Data Sets” window, activating vertical and horizontal scroll bars if necessary (Fig. 3.0.3).

Some properties of the DataSets are displayed: the name, the number of files, a flag indicating if it has been 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[4], 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.

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

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.

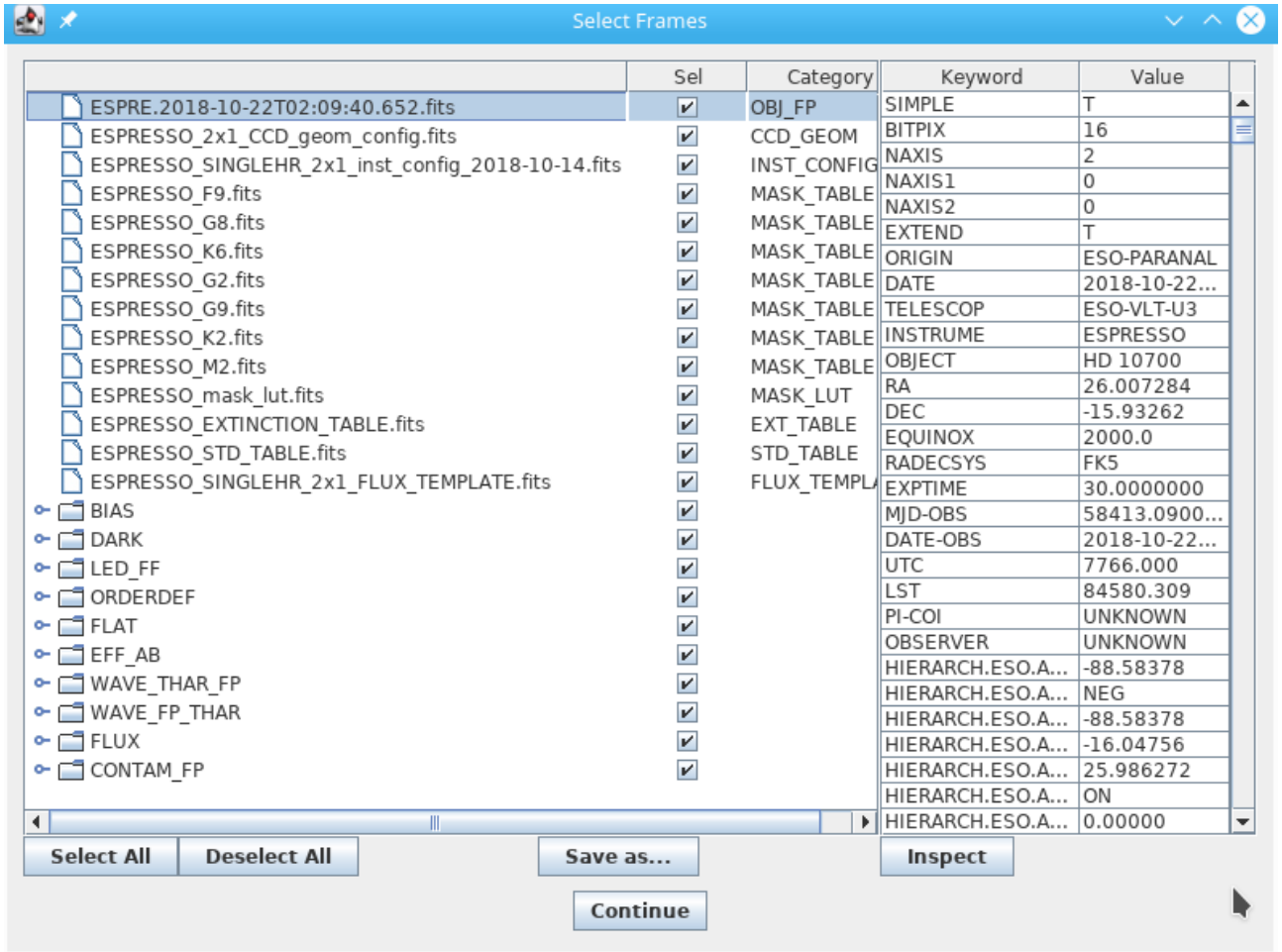



Figure 6.3.1: *The "Select Frames" window with a single file from the current Data Set highlighted in blue, and the corresponding FITS header displayed in the text box on the right. Hidden partially behind the "Select Frames" window is the "Select DataSets" window with the currently selected DataSet highlighted in blue.*

### 6.3.3 Recipe execution

Once the datasets to be reduced are selected, press the `Continue` button on the dataset organised window to proceed with the data reduction. The workflow will automatically execute the pipeline recipes and construct the `.sof` files to feed the pipeline recipes with. Each sof file will be saved in the `BOOKKEPING_DIR` directory (and subdirectory within it), depending on the recipe it is associated to and the execution time. The pipeline parameters can be changed as shown in figure 6.3.2.

### 6.3.4 Final products

Once a dataset is reduced (i.e. when the `espdr_sci_red` recipe is terminated), a window containing the list of science product pops up. Each file can be inspected with the selected fits viewer. Final science products will be stored in the `END_PRODUCTS_DIR`, and sorted by execution time and dataset identifier (i.e. the ARCHIVE file name of the science frame the dataset corresponds to). Default name for the science products is: `<TARG name>_S1D_FINAL_A<DATE-OBS>.fits` Its category is _S1D_FINAL_A. It is a fits binary table, that contain the following columns:

| | | |
|---|---|---|
| WAVE | [ Å] | Wavelength vector, in vacuum reference system. |
| FLUX | [erg cm$^{-2}$ s$^{-1}$ Å$^{-1}$] or [e–] | Flux of the final extracted spectrum in Fibre A, fully reduced till the end of the data reduction cascade. It is flux calibrated (if flux calibration was performed) and sky subtracted (if sky subtraction was performed). |
| ERR | [erg cm$^{-2}$ s$^{-1}$ Å$^{-1}$] or [e–] | Error associated to FLUX |
| QUAL | | Quality flag associated to FLUX |
| SNR | | Signal to noise ratio, obtained from FLUX/ERR. |
| WAVE_AIR | [ Å] | Wavelength vector, in the air reference system. |
| FLUX_EL | [e–] | Extracted flux from fibre A, not sky subtracted nor flux calibrated. |
| ERR_EL | [e–] | Error associated to FLUX_EL. |
| QUAL_EL | | Quality flag associated to FLUX_EL. |
| FLUX_CAL | [erg cm$^{-2}$ s$^{-1}$ Å$^{-1}$] | Extracted flux from fibre A, flux calibrated but not sky subtracted. |
| ERR_CAL | [erg cm$^{-2}$ s$^{-1}$ Å$^{-1}$] | Error associated to FLUX_CAL. |
| QUAL_CAL | | Quality flag associated to FLUX_CAL. |
| FLUX_CAL_SKYSUB | [erg cm$^{-2}$ s$^{-1}$ Å$^{-1}$] | Extracted flux from fibre A, flux calibrated and sky subtracted. |
| ERR_CAL_SKYSUB | [erg cm$^{-2}$ s$^{-1}$ Å$^{-1}$] | Error associated to FLUX_CAL_SKYSUB. |
| QUAL_CAL_SKYSUB | | Quality flag associated to FLUX_CAL_SKYSUB. |
| FLUX_EL_SKYSUB | [e–] | Extracted flux from fibre A, sky subtracted by not flux calibrated. |
| ERR_EL_SKYSUB | [e–] | Error associated to FLUX_EL_SKYSUB. |
| QUAL_EL_SKYSUB | | Quality flag associated to FLUX_EL_SKYSUB. |

Note: Some columns are present only if the corresponding step in the reduction cascade took place.

Other useful products are:
`<TARG name>_S1D_FINAL_B.fits`,
`<TARG name>_CCF_A_<DATE-OBS>.fits`, `<TARG name>_CCF_B_<DATE-OBS>.fits`,
`<TARG name>_S1D_A<DATE-OBS>.fits`, `<TARG name>_S1D_B<DATE-OBS>.fits`,
`<TARG name>_S1D_SKYSUB_A.fits`, `<TARG name>_S2D_BLAZE_A.fits`,
`<TARG name>_S1D_FLUXCAL_A<DATE-OBS>.fits`, `<TARG name>_S2D_A<DATE-OBS>.fits`,
`<TARG name>_S2D_B<DATE-OBS>.fits`, `<TARG name>_S2D_BLAZE_A<DATE-OBS>.fits`,
`<TARG name>_S2D_SKYSUB_A<DATE-OBS>.fits`,
`<TARG name>_AIR_DLL_MATRIX_THAR_FP_A<DATE-OBS>.fits`,

`<TARG name>_AIR_WAVE_MATRIX_THAR_FP_A<DATE-OBS>.fits`,
`<TARG name>_BLAZE_A<DATE-OBS>.fits`,
`<TARG name>_DLL_MATRIX_THAR_FP_A<DATE-OBS>.fits` and
`<TARG name>_WAVE_MATRIX_THAR_FP_A<DATE-OBS>.fits`.

The user may find more details on the science recipe products in the ESPRESSO pipeline manual.

Note that we have added in the product filename the value of the FITS keyword <DATE-OBS>. Moreover, the last five FITS producs in the previous list come from calibration recipes, and they have been added to the `END_PRODUCTS_DIR` to ease the user willing to analise the pipeline products with the ESPRESSO Data Analysis Software.

### 6.3.5 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. 3.0.4).

*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:

- 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.

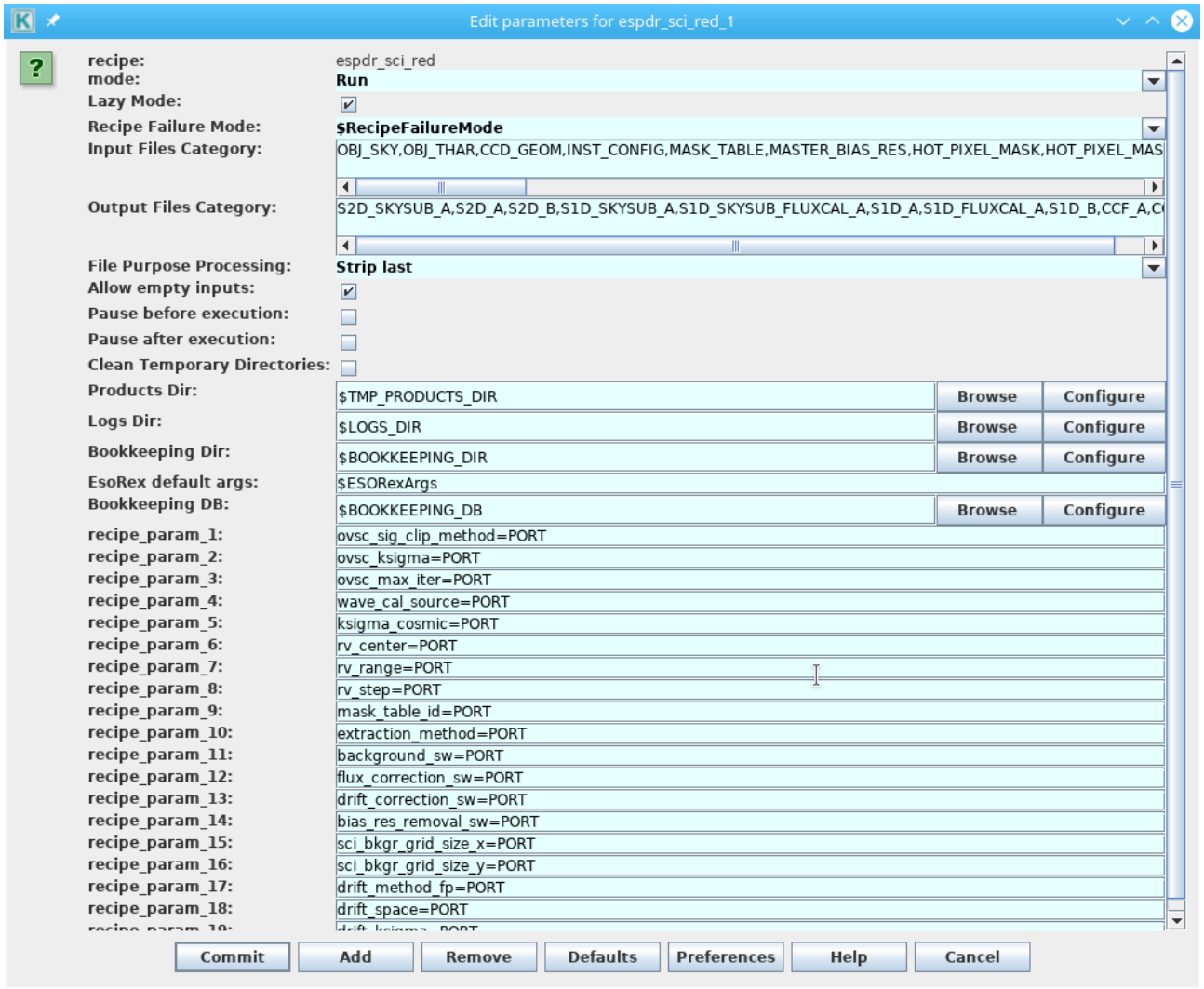



Figure 6.3.2: *The configuration parameters of a the science data reduction actor. The user may edit the configuration to change parameter values.*

*Exploring the data reduction products*

The left window of the ProductExplorer GUI shows the executions for all the datasets (see Fig. 3.0.4). 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.

# 7 Optimizing results through workflow Interactivity

This section describes the interactive GUIs provided in the ESPRESSO workflow. All recipes provide interactive workflows. By default reflex interactivity is set to false for all calibration recipes and to true for the science data reduction. To activate calibration recipes interactivity the "GloalPlotInteractivityCalibs" reflex global parameter (you will see on the top left hand side of the reflex top level workflow layout) has to be set to true.

In general the reflex interactive GUI layout presents one image at the top, and one graph at the bottom. Specific functionality provided by buttons or sliders may appear in some interactive workflow as appropriate. The image at the top is used to display 2D images either in pixel-pixel or pixel-order (S2D format) scales. The graphs at the bottom are used to display scatter plots of quality control parameters. On the top left the user may change a few recipe parameters. To optimise image update time the images in pixel-pixel coordinates are resampled of a factor that depends on the detector binning, the larger the smaller the bin size. The following sections provide more details on each interactive workflow. Each section name refer to the main product or goal of the corresponding data reduction step.

## 7.1 Master Bias

By default the master bias GUI displays at the top an image of the master bias of the BLUE detector (see Figure 7.1.1). The user may also display the master bias residuals or results for the RED detector.

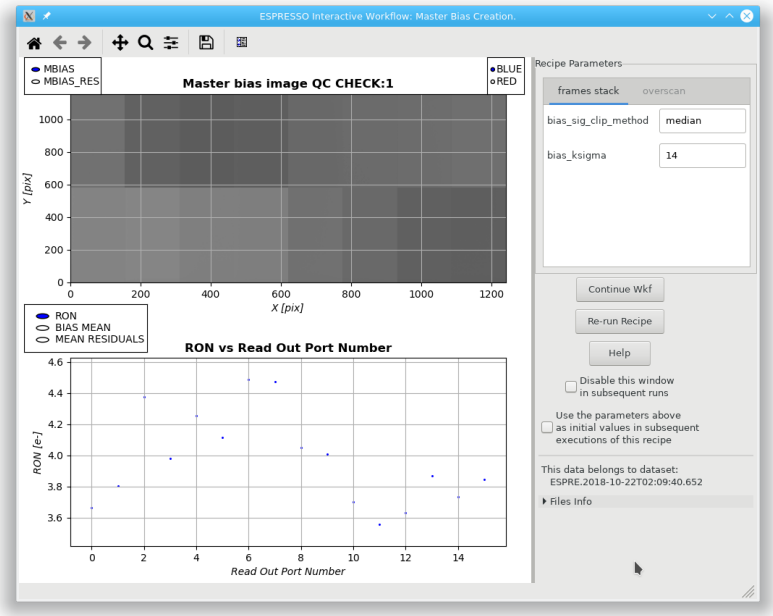



Figure 7.1.1: *Master Bias Interactive workflow.*

At the bottom the GUI shows a scatter plot with the distribution of the read out noise as a function of the read out port number. Alternatively the user may select to plot the bias mean or the mean of the computed residuals

as a function of the read out ports. The workflow allow to change parameters controlling the frame stacking or the computation of the overscan.

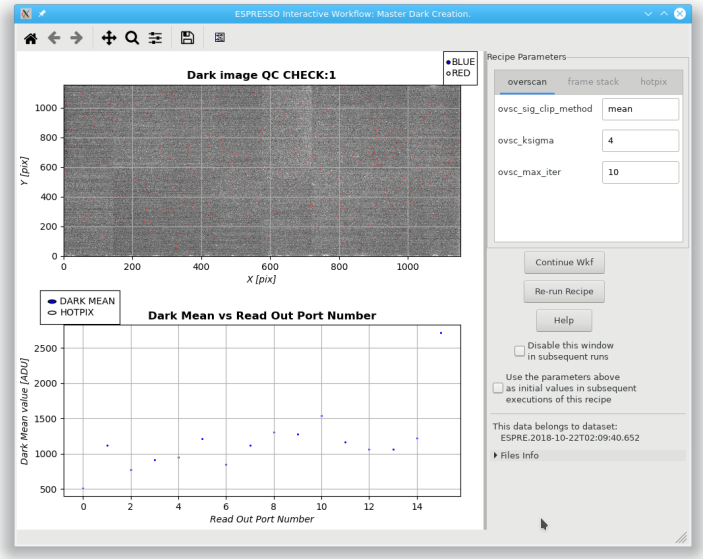



Figure 7.1.2: *Master Dark Interactive workflow.*

## 7.2 Master Dark

By default the master dark GUI displays at the top an image of the master dark of the BLUE detector (see Figure 7.1.2). The user may also display the master dark for the RED detector. At the bottom the GUI shows a scatter plot with the distribution of the mean of the DARK as a function of the read out port number. Alternatively the user may select to plot the number of hot pixels as a function of the read out ports. The workflow allows to change parameters controlling the frame stacking or the determination of the overscan, the frame stacking and of the hot pixels.

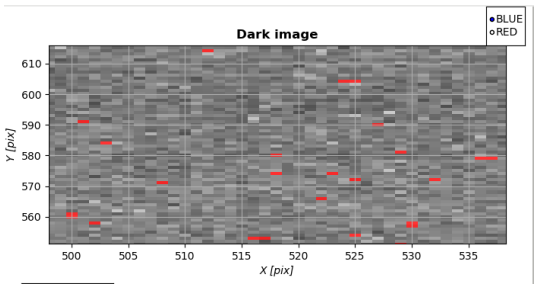



Figure 7.2.1: *Zoom in around a few hot pixels. These are flagged with a red hue.*

Note that the master dark image shows some pixels flagged with a red hue (see Figure 7.2.1). These corresponds

to the pixels detected as hot pixels.

## 7.3 Led FF

By default the led FF GUI displays at the top an image of the bad pixels (see Figure 7.3.1). At the bottom the GUI displays a plot of the computed conversion factor (CONAD[$\frac{e^-}{ADU}$]) as a function of the read-out port.

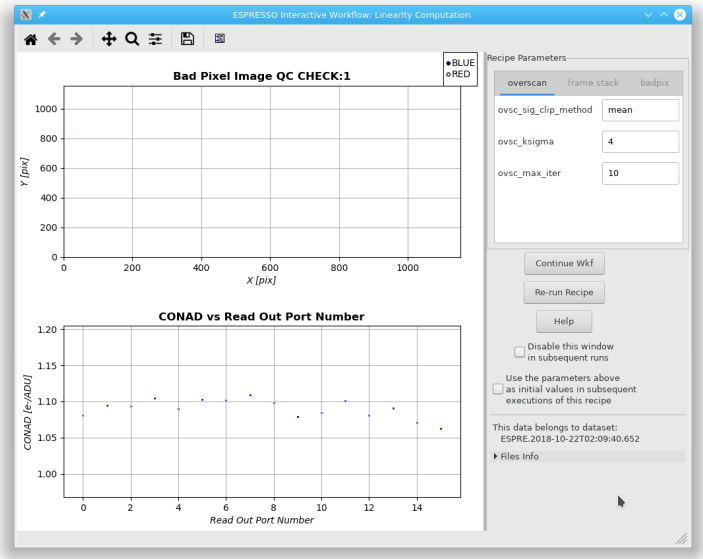



Figure 7.3.1: *Led-FF Interactive workflow.*

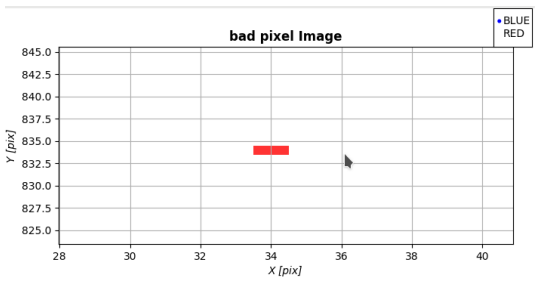



Figure 7.3.2: *Zoom on a detector area where there is a bad pixel.*

The bad pixels are flagged as a red hue (see Figure 7.3.2).

## 7.4 Orderdef

The orderdef GUI at the top displays an image of the frame used to trace the orders with super posed computed order traces in green color for a given detector and a given fiber (see Figure 7.4.1). At the bottom a scatter plot is shown with a distribution of the standard deviation (or the minimum or the maximum) of the computed residuals as a function of the read-out port. ESPRESSO provides data for two detectors, and the orderdef recipe processes two frames, one illuminating fibre A, the other illuminating fibre B. The user may change independently to display information that refers to the BLUE or the RED detector and fiber A or B.

If the user would like to change some recipe reduction parameter, the GUI provides at the top left a tab to modify three parameter values controlling the overscan computation.

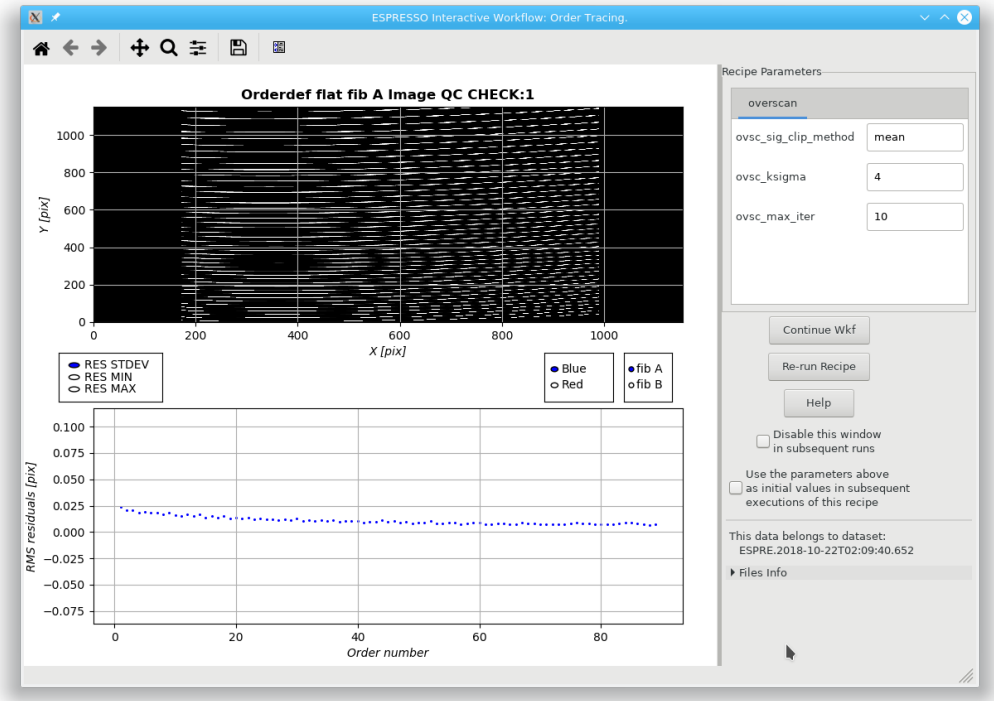



Figure 7.4.1: *Orderdef Interactive workflow.*

## 7.5 Master Flat

By default the master flat interactive workflow shows at the top an image corresponding to the stacking of the extracted flat field spectra from both detectors, end to end (S2D format) for fibre A (see Figure 7.5.1). An S2D image is a two-dimensional extracted spectra image in which each order is represented by an horizontal line. The wavelength runs horizontally across the order, increasing from left to right. The size of the S2D image is chosen to fully contain both the BLUE and the RED detector extracted spectra. BLUE detector extracted spectra appear at the bottom of the S2D image and are shorter, RED detector spectra appear at the top and

use the full image width. Below it is displayed the plot of the spectrum at a given order (by default number 10, thus corresponding to the BLUE detector, and this is the reason because the first 1700 pixels and last 1200 pixels have a uniform intensity equal to zero ADU). The order number ('order no') is indicated and controlled by the horizontal slider at the top. The user may choose to display also results for other products (`FS2D_B`, `FSPECTRUM_A`, `BLAZE_A`, `BLAZE_B`, `ORDER_PROFILE_A`, `ORDER_PROFILE_B`). At the bottom the GUI plots the SNR as a function of the order number. The interactive workflow allows the user to change parameters controlling the determination of the overscan, the frame stacking, the inter-order background determination, and the spectrum extraction. These parameters are grouped and presented in four corresponding tabs.

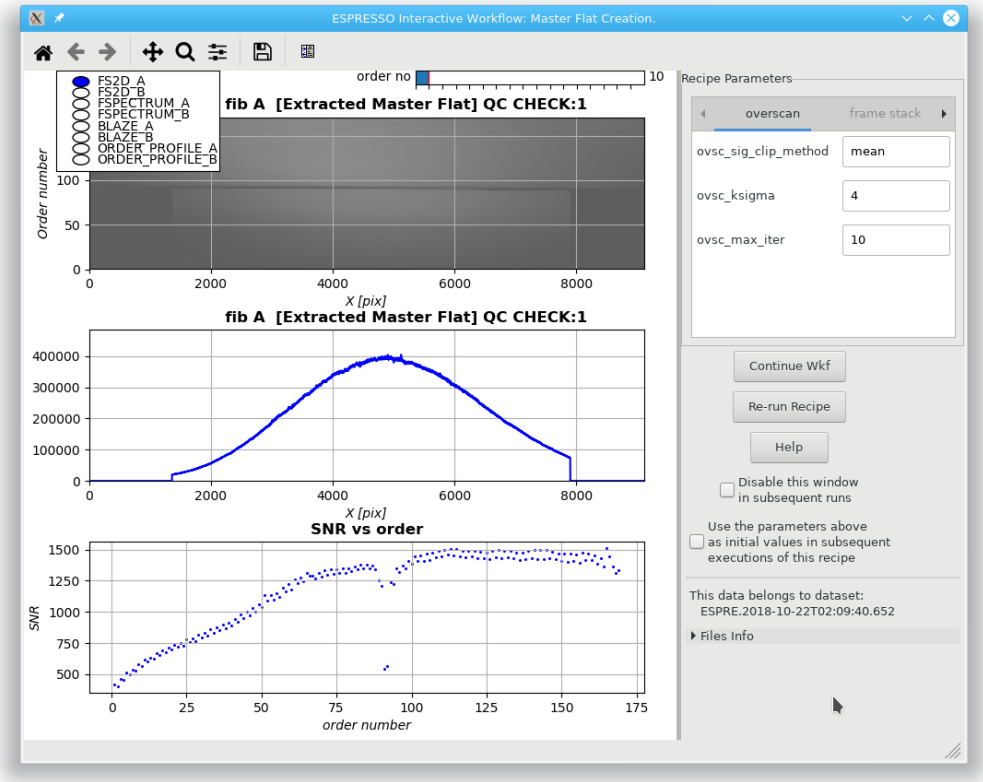



Figure 7.5.1: *Master Flat Interactive workflow.*

## 7.6 Wavelength Calibration: processing of frames where both fibres are illuminated by a Fabry Perot

This interactive workflow shows on the top of the window an image corresponding to the stacking of the extracted Fabry Perot spectra from both detectors, end to end (S2D format, see Figure 7.6.1). The user may display similar images for the fibre B or for the extracted blaze spectrum for fibre A or B. In the middle the GUI shows the extracted spectrum of the order number indicated by the horizontal slider. At the bottom the GUI shows the scatter plot of the number of detected lines as a function of the order number.

The user may change the parameters that control the overscan detection and the extraction.

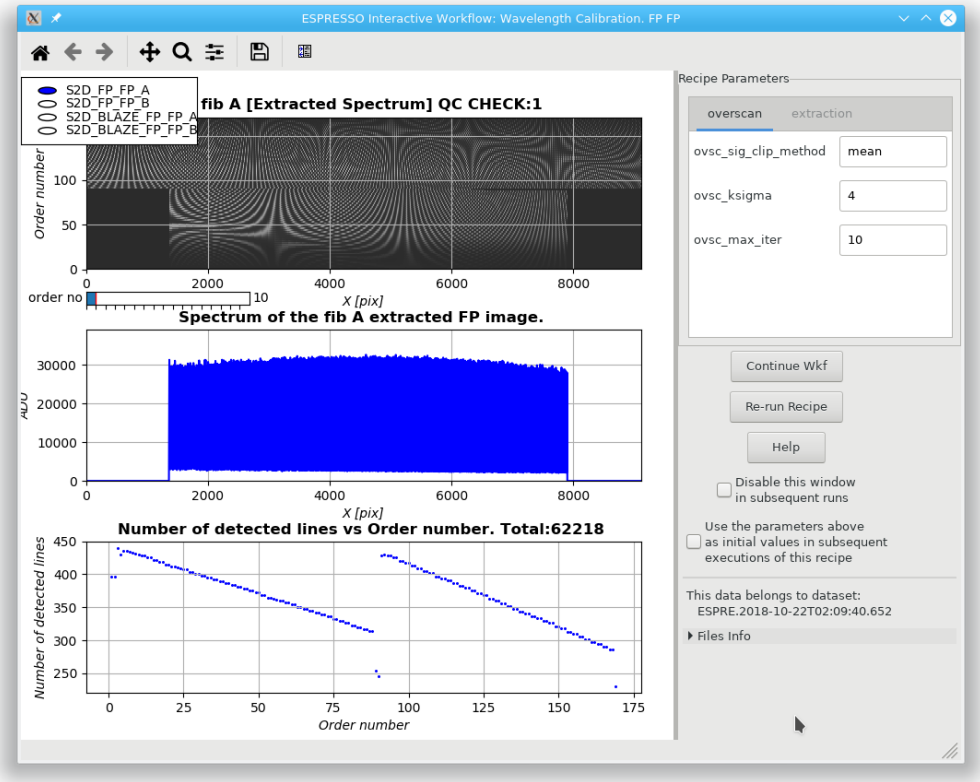



Figure 7.6.1: *Wavelength Calibration (FP,FP) Interactive workflow.*

## 7.7 Wavelength Calibration: processing of FP,THAR or THAR,FP data

This interactive workflows shows at the top an image corresponding to the stacking of the extracted Fabry Perot an ThAr spectra from both detectors, end to end (S2D format, see Figure 7.7.1) for fibre A in S2D format (THAR,FP). The user may display similar images for the fibre B (FP spectrum) or for the extracted blaze spectrum for fibre A or B. In the middle, the GUI shows the extracted spectrum corresponding to the image at the top and to the order number shown in the horizontal slider. At the bottom, the GUI shows the scatter plot of the resolution versus wavelength. Alternatively the user may display an image of the resolution trend in different detector positions (this is shown with a different color scale). Alternatively, the user may display the effective Fabry Perot cavity length (D) measured by the pipeline or its sigma, versus wavelength (LL), or the RMS of the wavelength solution versus the order number, or the computed CHI2 versus the order number.

The user may change the parameters that control the overscan detection, the extraction, the polynomial fit of D, and the polynomial wavelength solution.

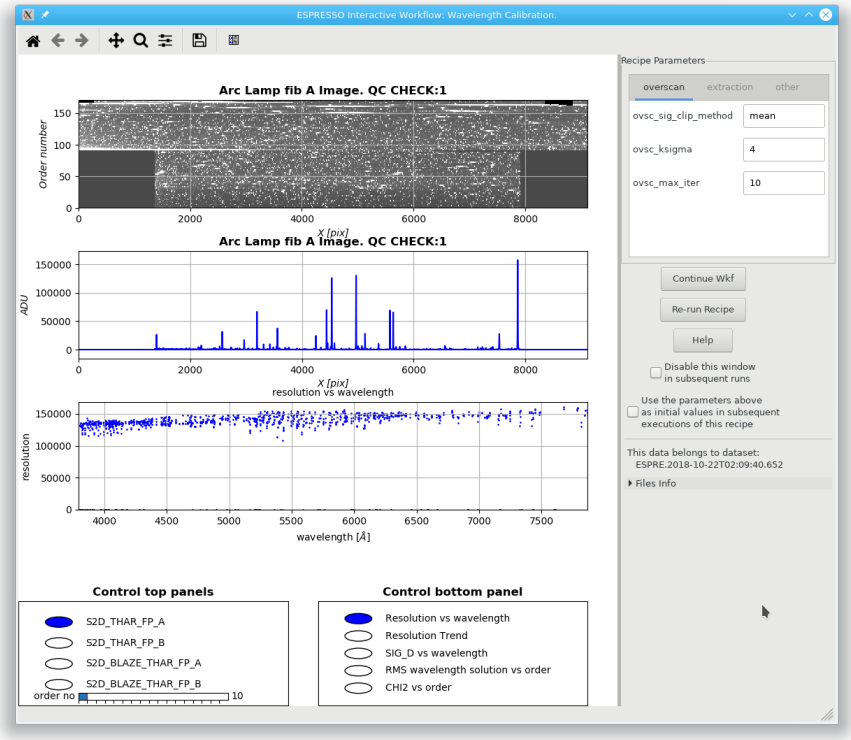



Figure 7.7.1: *Wavelength Calibration (THAR,FP) Interactive workflow.*

The interactive workflow showing results of FP,THAR reduction is similar (see Figure 7.7.2).

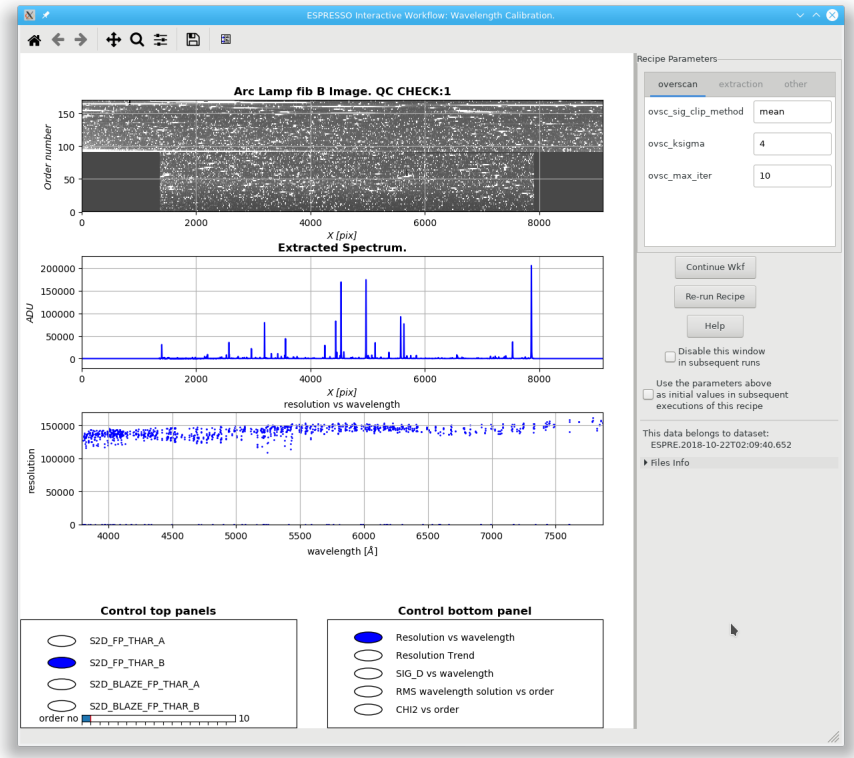



Figure 7.7.2: *Wavelength Calibration (FP,THAR) Interactive workflow.*

Note: If the peaks of the emission lies are not enough intense, the recipe can miss some identifications. In the unfortunate event that none of the peaks that should be identified in both orders are detected, the recipe cannot "connect" the wavelenght solution between orders and it fails. The error message in this case is:

```
[ ERROR ] espdr_wave_THAR: espdr_get_all_FP_ll_whole_image failed: Input data do not match
```

In some cases, this situation can be avoided by changing the recipe parameter `fp_extraction_ksigma` in the interactive window (tab: extraction).

## 7.8 Cross fibre contamination determination

This interactive workflow shows at the top the image of the combined BLUE and RED detector spectra for fibre A, used to compute the contamination of fibre B on fibre A (see Figure 7.8.1). The user may alternatively display the corresponding spectrum for fibre B or the image describing the contamination of the Fabry Perot image. The middle canvas shows the extracted spectrum corresponding to the order given in the horizontal slider. The canvas at the bottom shows the scatter plot of the maximum flux as a function of the order number.

The user may change parameters controlling the overscan determination, the inter-order background computation, and the extraction. The user may change the parameters that control the overscan, the inter-order background determination, the extraction and the polynomial fit degree used to fit the efficiency.

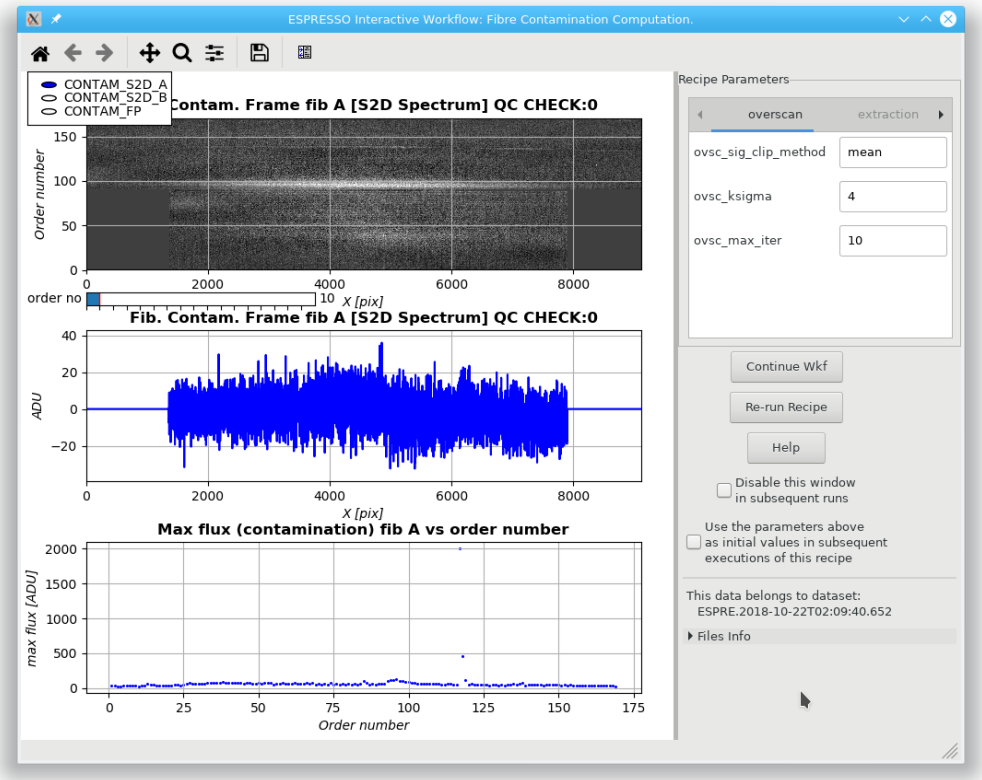



Figure 7.8.1: *Contamination data Interactive workflow.*

## 7.9 Relative Fibre efficiency determination

This interactive workflow shows at the top an image in S2D format of the extracted input sky data for fibre A (see Figure 7.9.1). In the middle shows the extracted spectrum for the order number displayed in the horizontal slider. At the bottom the scatter plot of the SNR versus the order number is shown.

The user may change the parameters that control the overscan, the inter-order background determination, the extraction and the polynomial fit degree used to fit the efficiency.

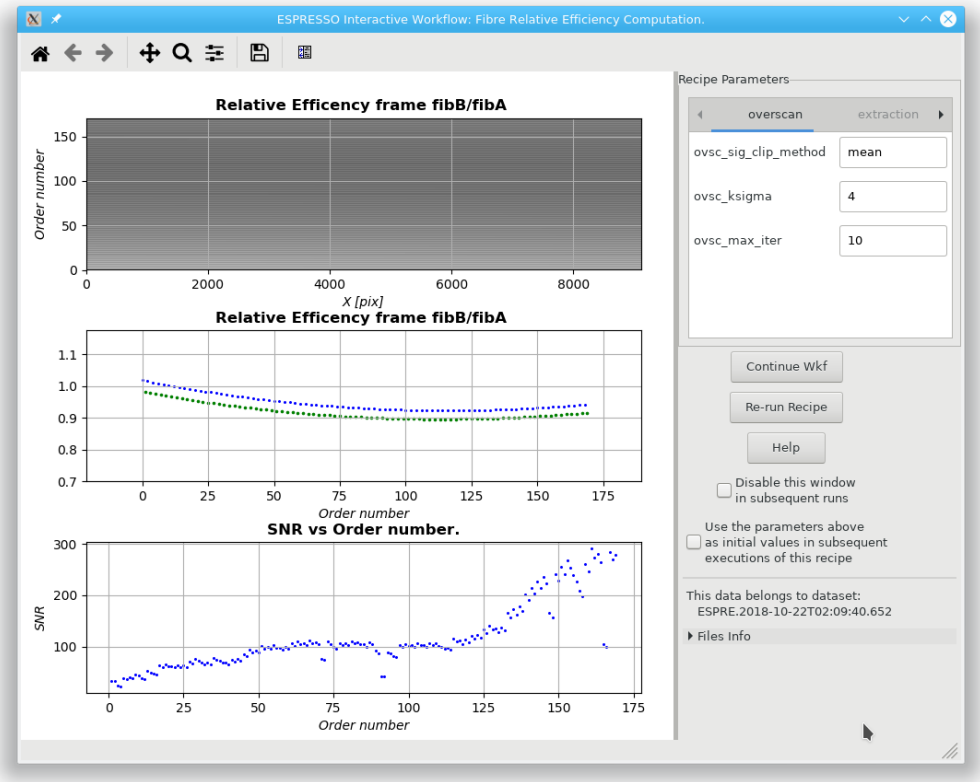



Figure 7.9.1: *Relative fibre efficiency determination Interactive workflow.*

## 7.10 Flux calibration

This interactive workflow shows at the top an image in S2D format of the combined detectors extracted flux STD star spectra for fibre A (see Figure 7.10.1). Alternatively the user may display the image of the blaze function in S2D format.

In the middle the extracted spectrum is shown for the order number corresponding to what the horizontal slider displays. At the bottom the plot of the computed efficiency as a function of wavelength is shown. Alternatively the user may display one of the following spectra:

- the energy spectrum obtained before the polynomial fit,
- the spectrum of the average flux observed for the flux STD star,
- the corresponding S1D extracted spectrum,
- the corresponding S1D measured energy spectrum.

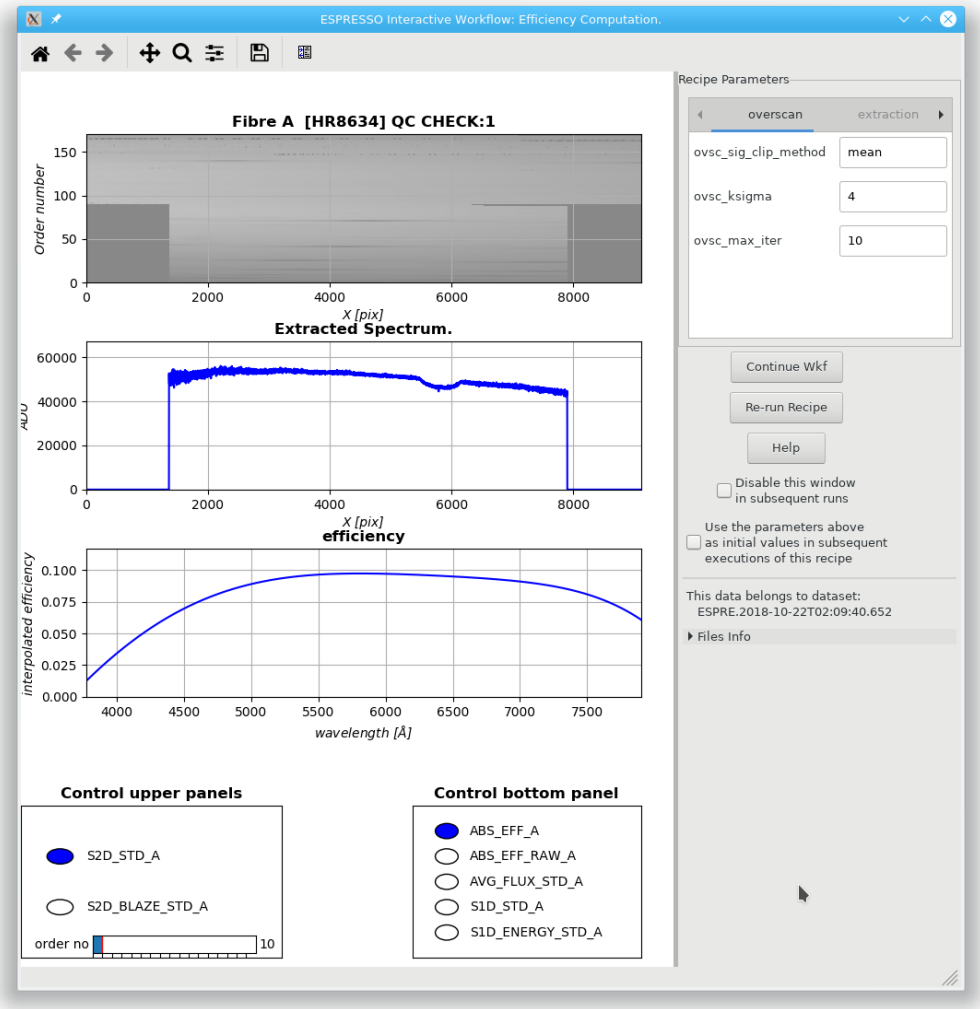



Figure 7.10.1: *Flux calibration determination Interactive workflow.*

The user may change the parameters that control the overscan, the inter-order background determination, the extraction and the polynomial fit degree used to fit the efficiency.

## 7.11 Science Data Reduction

This interactive workflow shows at the top an image in S2D format of the combined detectors extracted object spectra for fibre A (see Figure 7.11.1). Alternatively the user may display the image of the spectrum corresponding to fibre B in S2D format, or and image of the Cross Correlation Function (CCF).

In the middle the extracted spectrum is shown for the order number corresponding to what the horizontal slider displays. Then the GUI may display one of the following

- the spectrum of the object (default),
- the flux calibrated spectrum of the object,
- the spectrum of what is in the fibre B.

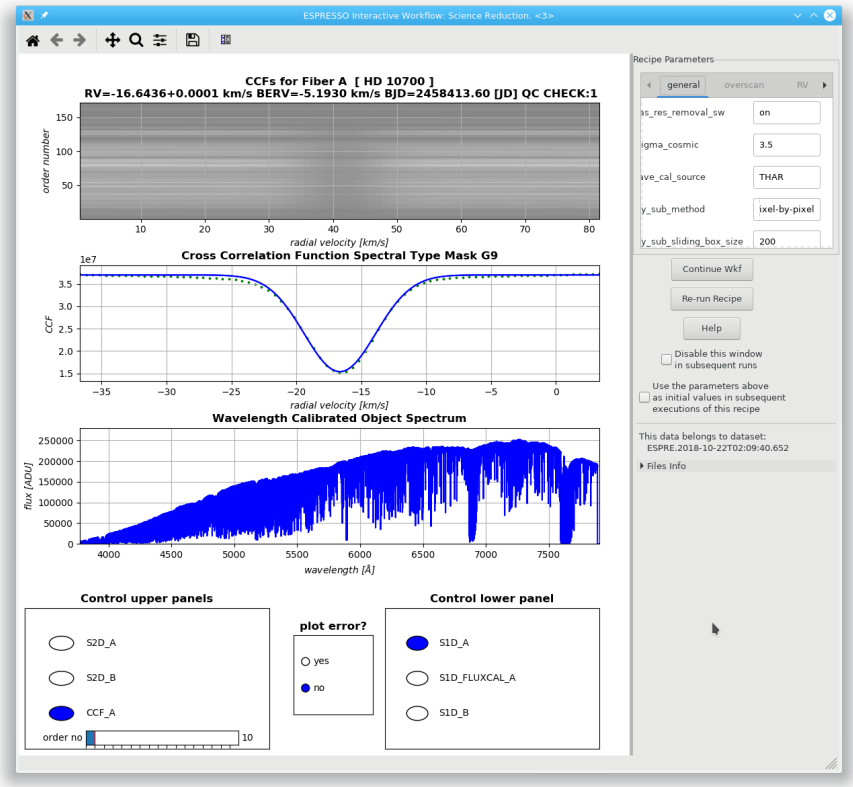



Figure 7.11.1: *Science data reduction Interactive workflow.*

The user may decide to plot also the error-bar (time consuming) on each S1D spectra.

Note: for extra galactic targets, identified by a not null value of the header keyword `HIERARCH ESO OCS OBJ Z EM`, the computation of the cross correlation function is not performed.

*Useful tips:*

The user may change the parameters that control the overscan, the inter-order background determination, the extraction, the radial velocity computation, the rejection of cosmics, the drift correction.

For very low S/N regime, readout limited or close to it, it is recommended to use the option smooth for sky subtraction, thus reducing the readout noise contribution from the step of sky subtraction.

The recipe parameter **background_sw** removes the intra-order diffuse background contamination (it does not have to be mis-interpreted by sky background, which is regulated by **sky_sub_method** and active only for OBJECT,SKY observations). It is evaluated on boxes of size **sci_bkgr_grid_size_x** $\times$ **sci_bkgr_grid_size_y** and

then interpolated all over the frame. The diffuse background is proportional to the dispersed light inside the spectrograph (and thus, to a good approximation, to the total flux recorded in the detector). If the diffuse background is not measured correctly (as happens when is negligible), correcting for it will only add an additional source of noise to the flux. FP light contributes significantly to the inter-order background and as such exposures with FP on fiber B have the inter-order background correction flag by default at ON. In most cases, stellar light does not contribute significantly to inter-order background and as such exposures with SKY on fiber B have the inter-order background correction flag by default at OFF. If any exposure points to the SKY on fiber B and yet has a very large target flux in fiber A, the user should change the inter-order correction flag to ON.

### 7.11.1 Cosmic rays cleaning

The science recipe has two algorithms for cosmic ray cleaning. They user can decide to use none, one or both by setting the corresponding recipe parameter in the interactive window.

The first algorithm uses the LA Cosmic algorithm (van Dokkum 2001, PASP 113, 1420) to flag cosmic rays in the raw frame (subtracted by overscan). This can be enable or disabled by setting the recipe parameter `cosmic_detection_sw` to 1 or 0, respectively. This parameter is located in the tab "General" of the interactive window. Pixels that are flagged are ignored during the extraction of the one-dimensional spectrum. The tab "LACOSMIC" in the interactive window allows the user to specify the recipe parameters that control the algorithm. By default, the workflow switches on/off the algorithm and loads the optimal recipe parameters for a given observing mode. We refer the user to the pipeline manual for more information on the recipe parameters. The user might want to inspect the products with category `CRH_MAP` and `CCD_CORR_SCIENCE` to assess the quality of the flagging process. These products are available in the reflex temporary products directory and produced only if the recipe parameter in the "General" tab `extra_products_sw` is flagged. Future versions of the workflow will allow inspection on the interactive window.

The second algorithm is a "k-sigma' clipping"; it operates on the extracted spectrum and flags pixels that deviate more than a certain factor (specified by the recipe parameter `ksigma_cosmic` times the noise from the model profile. This parameter is located in the tab "General" of the interactive window. Set `ksigma_cosmic` to a negative number to switch off this algorithm.

The general recommendation is to switch on both algorithms, but pay attention whether very intense targets or very intense exposures of the Fabry Perot fibre (which have very sharp edges in the spectral profile, in particular with high binning modes), do generate too many false positive detections. In this case, it is recommended either to switch the LA Cosmic algorithm off, or to increase the `f_lim` `sigma_lim` thresholds. If a cosmic ray is only partially detected, one might dilate the cosmic-ray mask by setting `post-filter-x` and `post-filter-y` parameters to 1 or 2 (higher values are not recommended).

The user has the option to provide a mask with cosmic rays detection (category `CRH_MAP`). If provided, it will switch off the execution of the LA Cosmic algorithm. In order to be used by the workflow and the recipes, the user-provided mask has to:

- be located in the `RAWDATA_DIR` with other input files
- have the category `CRH_MAP` (header keyword `HIERARCH ESO PRO CATG`).
- have the same format of `CRH_MAP` produced by switching on the LA Cosmic algorithm in the science

recipe espr_sci_red (same dimensions and number of extensions) and the same convention: pixels with value of 1 are cosmic rays, other pixels have value 0 (integers).

- have the header keyword ARCFILE of the same value of the science frame it is supposed to be associated to.

It is therefore recommended to first run the reduction with the LA Cosmic algorithm turned on, with the option extra_products_sw, and then edit the CRH_MAP produced by the workflow according to the needs. It is also possible to run a third-party cosmic ray detection algorithm on CCD_CORR_SCIENCE and record the flagged pixel into CRH_MAP.

Note that the inclusion of an user-provided CRH_MAP replaces the LA Cosmic algorithm in the science recipe, but the "k-sigma clipping" algorithm can still be executed.

### 7.11.2 Radial Velocity

The computation of radial velocity is a key-feature in the ESPRESSO pipeline. One might decide to fine-tune the recipe parameters to improve the calculation or to run a dedicated workflow and recipe to execute only this step. A dedicated workflow is available in the so-called ESPRESSO-DAS pipeline, named espda_starIII_wkf we refer the user to the ESPRESSO-DAS documentation for this option.

If one wants to optimize the computation of the radial velocity in the ESPRESSO pipeline, the recipe parameters to optimize are located into the RV tab of the interactive window. The general recommendation is to start with a large range rv_range and a course grid rv_step to locate the minimum. Then a fine-tuning of the range and grid are necessary to increase the precision of the measurement.

However, M-dwarfs are particularly prone to bad convergence in the CCV computation if the RV range is large. It is not recommended to derive RVs using an RV range larger than 20 km/s for M dwarfs.

When computing the radial velocity of the same target over a certain period of time, it is important to use the most updated version of the same template (mask_table_id) to remove systematic effects between measurements at different epochs.

# 8 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 (2018)). 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 (2018)) 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 (2018)) 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.

- **If I enter "-" into an empty integer parameter of an interactive window it is automatically completed to "-1". Why?**

  The parameters are validated for correctness according to their type (e.g. string, integer, float). In the case of an integer or float parameter "-" alone is considered an invalid input and is therefore automatically completed to "-1". This is part of the validation of input done by the WxPython library.

- **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 2, 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`

- **As part of my input raw frames I have several calibrations. Which will be associated to a given raw science?**

  Raw data are Organised Classified and associated following rules coded in an OCA configuration file. There is a default association rule that in case more than a set of frames fullfill certain criteria, the one that has closest in time MJD-OBS to the one of a given science frame, is the one used to reduce the data set associated to that science frame. In case a user would like to reduce data taken before midnight and data taken after midnight with the same set of calibrations, we recommend to choose the raw calibration frames acquired the following morning. These are usually the ones automatically associated by ESO archive tool calSelector.

- **I get different results on different computers. Why?** In some cases, it has been noticed a small variation on the radial velocity depending on the operative system of few cm/sec. This has to do with the intrinsic precision of the compilers. It is therefore recommended to use the same hardware when reducing the data.

# 9 Troubleshooting

In this section we describe some of the problems that may occur when reducing the ESPRESSO data with Reflex. For a more comprehensive description we refer the user to the ESPRESSO pipeline user manual (`http://www.eso.org/sci/software/pipelines/`). The user may also want to consult the Data Reduction F.A.Q. page at *http://www.eso.org/sci/data-processing/faq.html*.

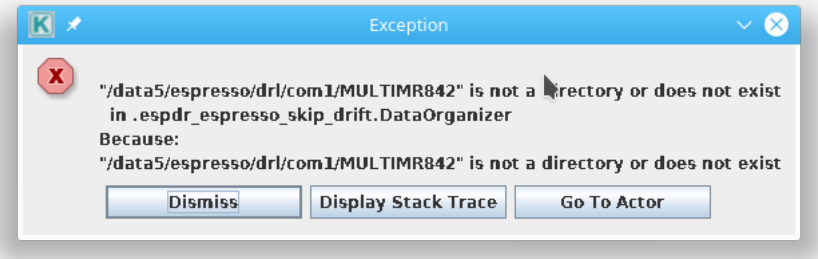



Figure 9.0.1: *TheDataOrganizer interactive window reports an error ":No Datasets have been created, check the DataSet and the OCA rules.".*

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

   (a) **The reflex execution fails at some stage.**
   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 not corrupted file (and remove the partial one).

   (b) **it fails with error message ":No DataSets have been created, check the data set and the OCA rules." (see Figure 9.0.1).**
   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 executing `Reflex`.

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 that 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. `ORDER_PROFILE_A_WKF`, `ORDER_PROFILE_B_WKF`, `FSPECTRUM_A_WKF`, `FSPECTRUM_B_WKF`, `BLAZE_A_WKF`, `BLAZE_B_WKF`). From this the user has then to determine the missing raw data (in this case flat frames for the given instrument mode and detector binning setting). If static calibrations are missing the mechanism unfortunately does not work, but should be found by `reflex` in

```
<install_directory>/calib/<pipeline_version>/cal
```

A faster way to identify which kind of data are missing is moving the mouse over the grey DataSet (see Fig. 9.0.2).

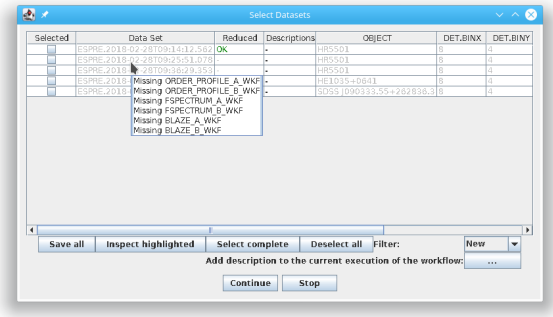



Figure 9.0.2: *This shows how information about missing files in an incomplete DataSet is displayed when moving the mouse over it.*

Similar commands can be given to check for other data types.

3. **Should I change the CALIB_DATA_DIR configuration?**

   This directory is setup automatically to point to the calibration database provided with the pipeline and in principle shouldn't be changed. However, if all the static calibration data are present in the RAWDATA_DIR (e.g. calibrations are downloaded from the archive via CalSelector, or edited by the user to fulfill some specific needs), then you can set CALIB_DATA_DIR to an empty value, in order be sure to use the static calibrations you put in RAWDATA_DIR.

4. **The data reduction fails at the flat reduction step. What I can do?** Please report this problem indicating the full input data set ARCFILE list. You probably have a problematic order determination frames. Usually changing the raw orderdef frames (or removing from the input directory the problematic ones, if you have alternative ones) and restarting the full data reduction solves the problem.

5. **"Why the recipe parameters that I specify to "Use the parameters above as initial values in subsequent executions of this recipe" in the interactive window are not used by the recipe?**

   This can happen if the global parameter "Get initial params from DB" in the main workflow canvas is set to "true" and the dataset has been already reduced in the past. In this way, the parameters used in the last successful run will be used instead of those specified via the interactive windows. Se the "Get initial params from DB" to "false" if you want to use different recipe parameters.

6. **I have a data reduction problem different from the one listed in this tutorial. What I can do?** We have listed here only a limited number of data reduction problems we are aware of. We kindly ask the user to read the error message(s) and recipe log provided by reflex to try to understand the problem. If the origin and solution of the problem not clear and not already described in this tutorial, follow the next steps. You may run into a different problem, in which case:

   - Provide a short description of the problem.

- Provide the error log if available (e.g. from reflex) or at least the full error message that is reported by the reflex widget reporting t hat an error occurred.
- Indicate the recipe affected.
- Include the full data set list (all ARCFILEs involved in the reduction of your DataSet).
- If the problem is in the reflex GUI please provide a snapshot.
- If the problem concerns data reduction quality or is an unexpected results.
- Please collect all problems by category: installation, robustness, reduction quality, reflex, unexpected behaviour/result, improvement.
- Please report data reduction problems to https://support.eso.org.

7. **Should I expect the FWHM of the cross-correlation function (CCF) of ESPRESSO to be smaller than that of HARPS on the same target, due to ESPRESSO's higher resolution?**

   The FWHM of the individual lines measured by ESPRESSO is smaller than that of HARPS, as expected due to the instrument's higher resolution. However, the HARPS and ESPRESSO pipelines use different algorithms with different line weighting schemes for the computation of the CCF's FWHM. These differences make HARPS data reduced with the HARPS pipeline to have smaller FWHM than ESPRESSO data of the same target processed with the ESPRESSO pipeline.

8. **The esprd_sci_red recipe crashes, but all the input files and parameters seem ok. What happened?** There have been reported few cases in which the science pipeline crashes with no apparent reason, providing the following error message:

```
10:00:07 [WARNING] espdr_correct_flux: [tid=000] ESC[31mFlux correction not performed:
                   no flux template available for spectral type F5ESC[0m
10:00:07 [ INFO ] : [tid=000] Computing CCF for fibre A sky_sub for size_y = 170 ...
10:00:47 [ ERROR ] espdr_sci_red: [tid=000] espdr_compute_CCF failed for sky_sub:
                   Access beyond boundaries
10:00:47 [ ERROR ] cpl_errorstate_dump: [tid=000] Lost 11192 CPL error(s)
10:00:47 [ ERROR ] cpl_errorstate_dump_one_level: [tid=000] [11193/11212]
                   'Access beyond boundaries' (11) at cpl_image_get:cpl_image_io.c:747
```

   To our best knowledge, the problem is associated to the presence of cosmic rays on some raw calibrations (category: `WAVE,FP,FP`), which led to a product that caused the failure in the science recipe. Future versions of the pipeline will be able to identify such issues and flag them appropriately.

   To overcome the issue, two solutions are currently available:

   - Remove the cosmic ray from the faulty calibrations with external tools before starting the data reduction.
   - Replace the calibrations with those from a previous or following day. It might be worth replacing all the calibrations (all WAVE, ORDERDEF, and FLAT types), even if not all are affected by cosmic rays, to avoid fibre misalignment between calibrations from different days.

Forchì V., 2018, Reflex User Manual, VLT-MAN-ESO-19000-5037, Issue 3.9, 15, 42, 43

ESPRESSO Pipeline User Manual, VLT-MAN-ESO-19500-XXXX, Issue 16.0

# A Useful information

## A.1 Reflex directory naming conventions

We provide here a short description of the usual directory naming conventions, referring to the ESPRESSO case.

A standard `install_esoreflex` based installation, for example under `install_dir` will generate the following directory structure:

- `data_wkf`, this is the directory to which points the Reflex global variable `ROOT_DATA_DIR`. It contains the following directories:
  - `reflex_input`, containing the demo data, in the sub-directory espdr/raw. The global variable `RAW_DATA_DIR` points to `ROOT_DATA_DIR/reclex_input/espdr`
  - `reflex_book_keeping`, which will be populated by all the reflex bookeeping information, in the subdirectory espdr/. In particular the file bookkeeping.db, that is a mini-sql based database created, updated and used by reflex, the DataOrganizer/ directory, containing information on the current input data organisation classification and association, a set of directories called as `recipe_name_1` (for example `espdr_mbias_1`) which contain a set of subdirectories named as YYYY-MM-DDThh:mm:ss.mmm with the current execution timestamp id (for example 2019-04-16T21:42:54.335) and a link 'latest' pointing to the latest execution. These directories contains information relative to the esorex execution, and may be useful to debug if necessary. There are also other directories called (in case of ESPRESSO) as `InteractiveWorkflow_recipe_name`, `InteractiveWorkflow_espdr_mbias`
  - `reflex_tmp_products`, this directory contains under the subdirectory espdr/ the recipe temporary products (products of calibration recipe executed during the reduction chain) in subdirectories named like `recipe_name_1` and the directories `InteractiveWorkflow_recipe_name_1` (usually empty, but which may contain products of the python scripts used to handle the Reflex interactivity)
  - `reflex_logs` with a structure similar to `reflex_tmp_products` and containing the log of each recipe execution.
  - `reflex_end_products`, that contains a directory nammed with the current timestamp, as YYYY-MM-DDThh:mm:ss.mmm, containing a README file and other directories named as YYYY-MM-DDThh:mm:ss.mmm, by the timestamps of the first raw frame of the product they refer to, and containing all the final products of the corresponding execution.

## A.2 Available Organisation Classification and Association (OCA) rules

The ESPRESSO workflow contains two OCA rules, the `espresso_wkf_full.oca` and the `espresso_wkf_skip.oca`, which allows respectively the execution of the full reduction chain (provided one has all required input data and static calibrations), and the execution of the reduction chain skipping the (time consuming and usually stable) dark and led frame processing, provided one has the products of the recipe

`espdr_mdark` (`HOT_PIXEL_MASK`) and `espdr_led_ff` (`BAD_PIXEL_MASK`) for the given instrument mode and binning, for example obtained by executing the full reduction chain on one data set.

## A.3 Glossary

We define here all relevant achronyms.

### A.3.1 espresso data reduction naming convention

We provide here the definition for the most frequent achronims used in the tutorial. The user is also invited to read the pipeline user manual.

- S1D, extracted and merged spectrum 1D, flux (ADU or physical units) versus wavelength.
- S2D, combined detectors (BLUE and RED camera) extracted spectra order-by order, a 2D image containing at each raw the extracted spectrum of a given order. At the bottom of the image are stored orders corresponding to the BLUE camera, at the top orders corresponding to the red camera.
- CCF: Cross Correlation Function.
- SINGLEHR, single High Resolution observing mode. It may have two bining (XxY): 1x1 or 2x1.
- SINGLEUHR, single Ultra High Resolution observing mode. Only 1x1.
- MULTIMR, Multi-UT, Medium Resolution oserving mode. It may have two binnings: 4x2, 8x4. It combines the light from any of the UTs.
- THAR: Thorium Argon wavelength calibration lamp.
- FP: Fabry Perot wavelength calibration unit.
- LFC: Laser Fiber Comb wavelength calibration unit.
- CCD: Charge Coupled Device, the ESPRESSO detector type.
- RO: (detector) Read Out region/port.
- RON: Read Out Noise.
- EXTi: Extension (index number i)
- FS2D: Order by order exracted spectrum
- BLAZE: Blaze spectrum for a given fibre
- `ORDER_PROFILE`: Order profile for a given fibre
- SNR: Signal to Noise Ratio
- CONAD: Conversion Analog to Digital factor [$e^-$/ADU]

- FSPECTRUM:Order by order exracted spectrum for a given fibre
- DLL: Distance-Wavelength.
- HOT: Hot Pixel (pixel with value significantly above the detector noise)
- BAD: Bad (not-linear) Pixel
- BERV: Barycentric Earth Radial Velocity.
- RV: Radial velocity
- BJD: Barycentric Julian date.
- DRIFT: relative shift of a wavelength calibration solution beween the time of (night) observation and the time when the calibration was acquired (during the day).

#### A.3.2 reflex workflow acronims

- SoF: set of frames
- DataSet: data set