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 X-shooter Tutorial

VLT-MAN-ESO-19540-5899

Issue 3.9.1

Date 2026-10-19

	Name	Date	Signature
Prepared:	Sabine Moehler, Daniel Bramich	2026-10-19	
Approved:	W.Freudling		
Released:	M. Sterzik		

This page was intentionally left blank

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
1.0	05/04/2012	All	Public release
1.1	23/04/2012	All	Comments from previous release
1.2	15/05/2012	All	Comments from previous release
1.3	22/05/2012	All	Coordinated wih other pipelines releases
1.5	21/12/2012	All	Science grade release
2.0	29/04/2013	All	Coordinated wih other pipelines release
2.1	21/10/2013	All	Coordinated wih other pipelines release
2.2	10/04/2014	All	Upgrade to Reflex 2.6.
2.3	25/06/2014	All	Support for IFU data reduction, creation of SDP.
2.4	02/03/2015	All	Upgrade to xshoo-2.6.0
2.5	17/07/2015	All	Upgrade to xshoo-2.6.5
2.6	14/09/2015	All	Upgrade to xshoo-2.6.8
2.8	31/03/2016	All	Upgrade to xshoo-2.7.1
2.9	25/07/2016	All	Upgrade to xshoo-2.8.0
2.10	16/09/2016	All	Upgrade to xshoo-2.8.3
2.11	27/09/2016	All	Upgrade to xshoo-2.8.4
2.13	19/04/2017	All	Upgrade to xshoo-2.9.0, restructuring of tutorial
2.14	31/05/2017	All	Upgrade to xshoo-2.9.1
2.16	19/10/2017	All	Upgrade to xshoo-2.9.3
2.17	01/04/2018	All	Upgrade to xshoo-3.1.0
3.50	22/05/2020	All	Upgrade to xshoo-3.5.0, redundant info. removed

This page was intentionally left blank

Contents

1 Introduction to `EsoReflex` **9**

2 Software Installation **10**

2.1 Installing `Esoreflex` workflows via `macports` . . . 10

2.2 Installing `Esoreflex` workflows via `rpm/yum/dnf` . . . 10

2.3 Installing `Esoreflex` workflows via `install_esoreflex` . . . 10

2.4 Demo Data . . . 11

3 Quick Start: Reducing The Demo Data **12**

4 About the main `esoreflex` canvas **16**

4.1 Saving And Loading Workflows . . . 16

4.2 Buttons . . . 16

4.3 Workflow States . . . 16

5 The X-shooter Workflow **17**

5.1 Workflow Canvas Parameters . . . 17

5.2 Workflow Actors . . . 18

5.2.1 Simple Actors . . . 18

5.2.2 Composite Actors . . . 19

5.2.3 Recipe Execution within Composite Actors . . . 20

5.2.4 Lazy Mode . . . 26

6 Reducing your own data **27**

6.1 The `esoreflex` command . . . 27

6.2 Launching the workflow . . . 27

6.3 Workflow Steps . . . 29

6.3.1 Data Organisation And Selection . . . 29

6.3.2 DataSetChooser . . . 31

6.3.3 The ProductExplorer . . . 31

6.3.4 Creation Of Master Calibration Files . . . 33

6.3.5 2D Distortion Mapping . . . 34
6.3.6 Response Determination, Science Reduction . . . 34
6.3.7 Output Organisation . . . 36

7 Optimising Your Results Through Workflow Interaction 37
7.1 Interactive Windows . . . 37
7.1.1 Buttons . . . 37
7.1.2 Plots . . . 38
7.1.3 Recipe Parameters . . . 38
7.2 Reducing Stare Mode Observations . . . 38
7.2.1 Interactive Window . . . 40
7.2.2 Improving the result . . . 40
7.3 Reducing Nodding Mode Observations . . . 44
7.3.1 Interactive Window . . . 45
7.3.2 Improving the result . . . 45
7.4 Reducing Offset Mode Observations . . . 48
7.4.1 Interactive Window . . . 49
7.4.2 Improving the Result . . . 49
7.5 Instrument Response . . . 53
7.6 Known Problems . . . 57
7.6.1 Workflow Problems . . . 57
7.6.2 Pipeline Problems . . . 57

8 Frequently Asked Questions 60
8.1 X-shooter specific questions . . . 62

9 Troubleshooting 69

A From Spectral Format Check Through to Wavelength Calibration 71
A.1 Spectral Format . . . 72
A.1.1 Interactive Window . . . 72
A.1.2 Improving the Results . . . 73

A.2 Order Position 73
A.2.1 Interactive Window 73
A.3 Master Flat Creation 75
A.3.1 Interactive Window 75
A.4 2D Geometry 76
A.4.1 Interactive Window 76

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 environment, 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 X-shooter observations only via the X-shooter `Reflex` workflow, using the Physical Model mode of the X-shooter pipeline **and raw calibration files**. The user is referred to the X-shooter user manual (Martayan et al. 2011[2]) for more information on the instrument itself, and the X-shooter pipeline user manual (Modigliani 2012[3]) for the details of X-shooter pipeline recipes. The quick start section (see Section 3) describes the minimum effort to get started, and it makes up only two pages of text in this tutorial.

In case of problems we recommend the user to carefully read sections 7.6 and 8.1 and check the X-shooter and Reflex F.A.Q. pages. [4]

[1] *https://www.eso.org/sci/archive/calselectorInfo.html*
[2] available at *http://www.eso.org/sci/facilities/paranal/instruments/xshooter/doc*
[3] available at *ftp://ftp.eso.org/pub/dfs/pipelines/xsh/xsh-pipeline-manual-3.9.1.pdf*
[4] available at *http://www.eso.org/sci/data-processing/faq.html#xshooter*

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.

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 (*https://www.macports.org*) is installed. Please read the full documentation at *https://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 *https://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 those 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 3.12` may be found at:
https://www.eso.org/sci/software/pipelines/reflex_workflows

To install the `Reflex 3.12` 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 you will also recieve a demo data set, that allows you to run the `Reflex X-shooter` workflow without any changes in parameters. This way you have a data set to experiment with before you start to work on your own data.

Note that you will need a minimum of ∼2.1 GB, ∼1.5 GB and ∼2.8 GB of free disk space for the directories `<download_dir>`, `<install_dir>` and `<data_dir>`, respectively, at least ∼3.0 GB of total memory (physical + swap), and a screen with a resolution better than 1024x800 pixels . The X-shooter demo data have been retrieved with the CalSelector tool[5] The demo data sets contains only SLIT observations. There are two data sets for UVB data one taken in offset mode, and one taken in stare mode, two data sets for VIS data taken in stare mode, and four data sets for NIR data of which three are taken in nodding mode and one in stare mode. In addition we provide one incomplete NIR data set taken in stare mode to illustrate how an incomplete DataSet looks in the "Select DataSets" window (see Fig. 3.3). The first six valid data sets include standard star observations taken in offset mode, as used to be observed for early X-shooter operations, while the last two valid data sets include standard star observations taken in nodding mode, as is currently done.

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

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 X-shooter demo data set supplied with the `esoreflex 3.12` 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 X-shooter by typing:

   ```
   esoreflex xshooter&
   ```

   Alternatively, you can type only the command `esoreflex` the empty canvas will appear (Figure 3.1) 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 X-shooter workflow is shown in Figure 3.2.

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 3.3) that lists the datasets along with the values of a selection of useful header keywords[6]. 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 3.4 shows the Product Explorer window. A full description of the *Product Explorer* will be presented in Section 6.3.3.

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

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

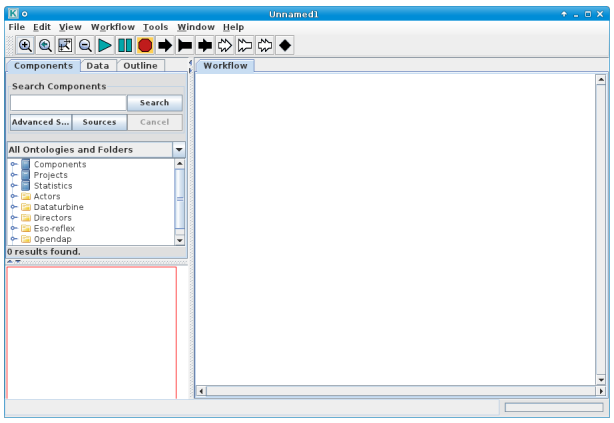


Figure 3.1: *The empty* `Reflex` *canvas.*

[6]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.

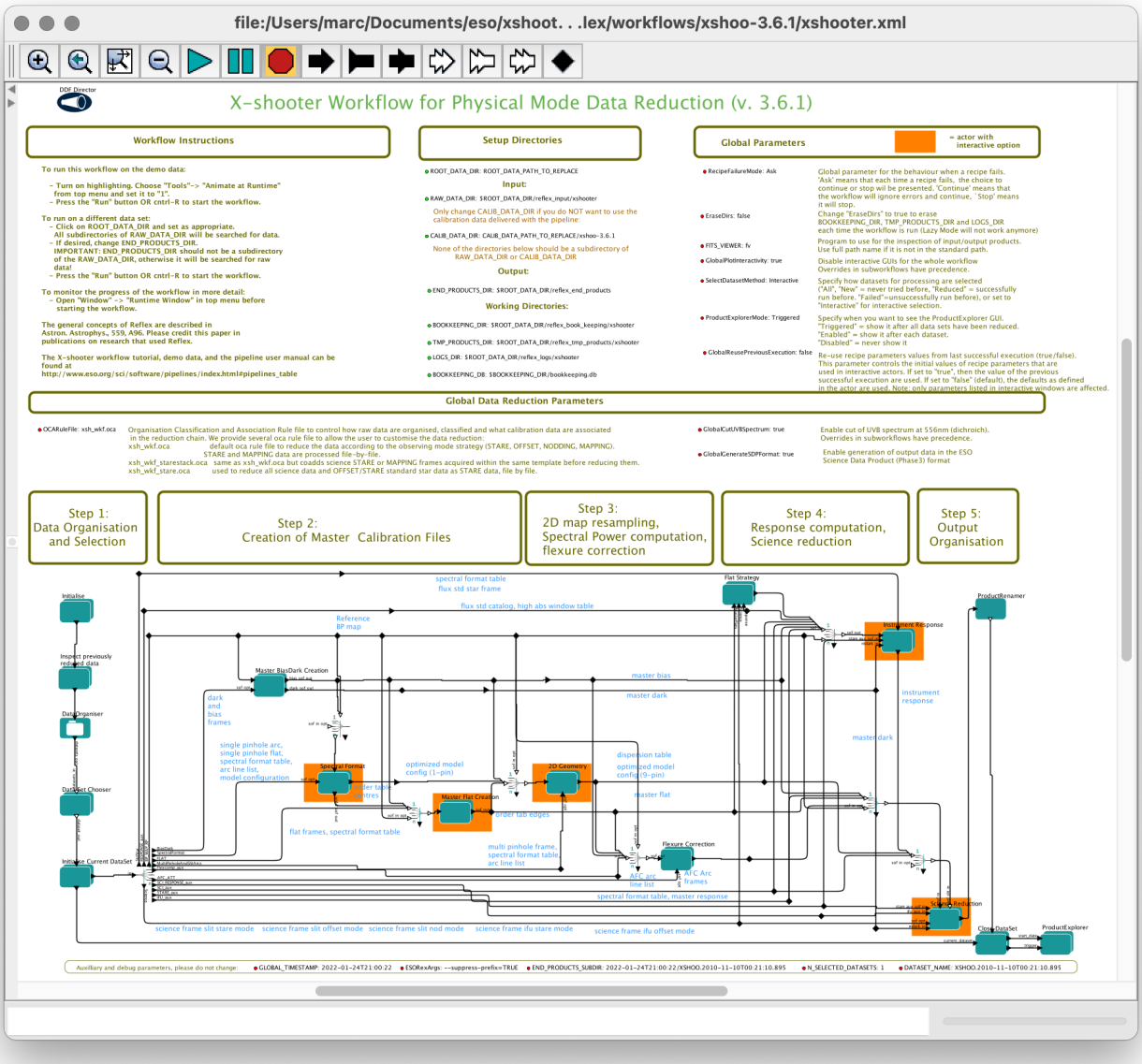


Figure 3.2: *X-shooter workflow general layout.*

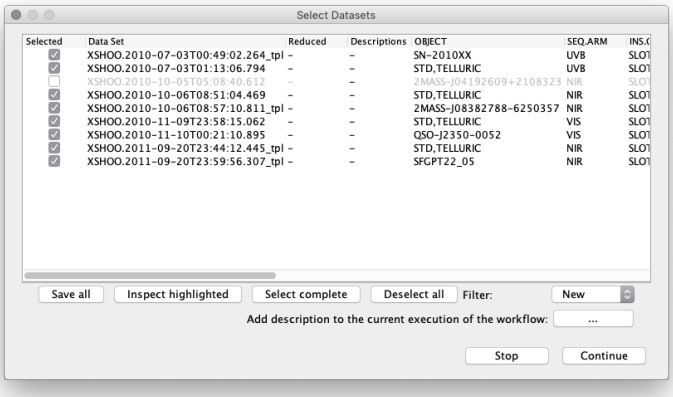


Figure 3.3: *The "Select Datasets" pop-up window. On purpose the demo includes an incomplete (grey) DataSet.*

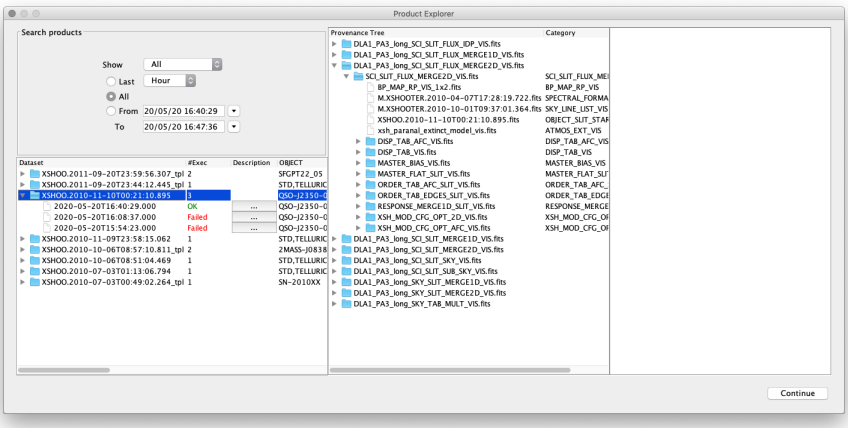


Figure 3.4: *The Product Explorer shows all datasets reduced in previous executions together with the full reduction chain for all the pipeline products.*

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 X-shooter Workflow

The X-shooter 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 (2012)).

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

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

By default the `EraseDirs` parameter is set to `false`, which means that no directories are cleaned before executing the workflow, and the recipe actors will work in Lazy Mode (see Section 5.2.4), reusing the previous pipeline recipe outputs if input files and parameters are the same as for the previous execution, which saves considerable processing time. Sometimes it is desirable to set the `EraseDirs` parameter to `true`, which forces the workflow to recursively delete the contents of the directories specified by `BOOKKEEPING_DIR`, `LOGS_DIR`, and `TMP_PRODUCTS_DIR`. This is useful for keeping disk space usage to a minimum and will force the workflow to fully re-reduce the data each time the workflow is run.

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

The parameter `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 `GlobalReusePreviousExecution` controls the initial values of recipe parameters that are used in interactive actors. The possible values are `true`, `false`: `false` (the default) will use the parameter defaults defined in the actor, `true` will instead cause the values of the previous successful execution to be used. Note that this only applies to parameters listed in the actor: some recipe parameters may not be listed, and for those the value will always be as defined by the recipe itself, regardless of the value of `GlobalReusePreviousExecution`.

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.

The X-shooter workflow has two more global parameters:

`GlobalCutUVBSPectrum` defines if the UVB spectra are cut at 556 nm to avoid the huge pseudo-absorption from the dichroic (`true` by default).

`GlobalGenerateSDPFormat` defines if the extracted 1-dimensional spectra are also saved as tables (`true` by default).

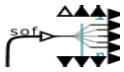
5.2 Workflow Actors

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).

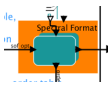
5.2.2 Composite Actors

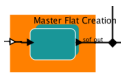
Composite actors have workflow symbols that consist of multiply-layered green-blue rectangles. They generally do not have a logo within the rectangle. A composite actor represents a combination of more Simple or composite actors which hides over-complexity from the user in the top-level workflow. In the X-shooter workflow, the following actors are composite actors (actors with interactive workflows are indicated by an orange rectangle: `Spectral Format`, `Master Flat Creation`, `2D Geometry`, `Instrument Response` and `Science Reduction`).):

-

- The `Initialise` actor.
-

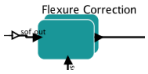
- The `Display Current DataSet` actor.
-

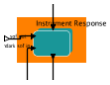
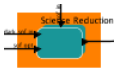
- The `Master BiasDark Creation` actor.
- 

- The `Spectral Format` actor.
- 

- The `Master Flat Creation` actor.
-

- The `Flat Strategy` actor.
-

- The `2D Geometry and Wavecal` actor.
- 

- The `Flexure Correction` actor.
- 
- The `Instrument Response` actor (with an interactive window).
- 
- The `Spectrum Reduction` actor (with an interactive window).
-

- The `Close DataSet` actor.

Access to the parameters for an interactive composite actor is achieved by right-clicking on the actor and selecting `Configure Actor`. This will open an "Edit parameters" window (see Figure 5.1).

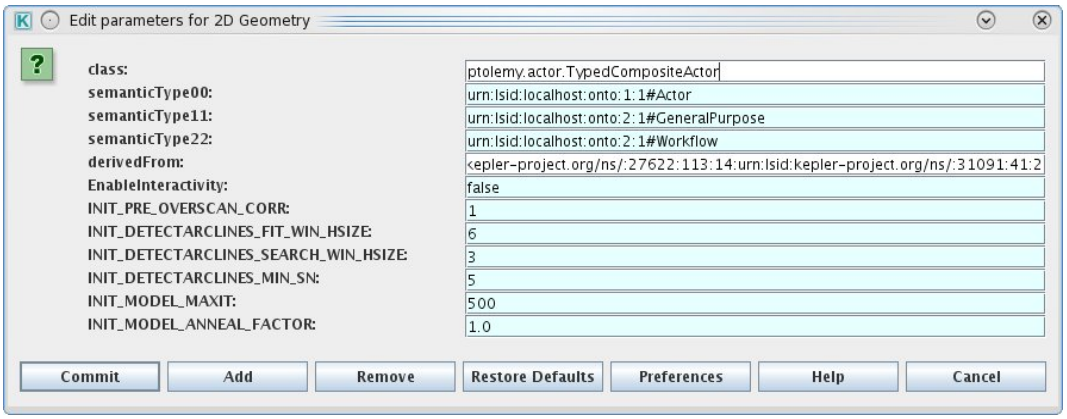


Figure 5.1: *The "Edit parameters" window for a typical composite actor..*

Composite actors may also be expanded for inspection. To do this, right-click on the actor and select `Open Actor`, which will expand the composite actor components in a new `Reflex` canvas window.
For an X-shooter specific actor this window provides a brief summary of the processing performed by the actor. If the composite actor corresponds to a pipeline recipe, then the corresponding `RecipeExecuter` actor will be present as a Simple Actor, and its parameters are accessible as for any other Simple Actor. Alternatively you may still find composite actors, on which you need to repeat the first step to access the `RecipeExecuter`.

5.2.3 Recipe Execution within Composite Actors

The X-shooter workflow contains composite actors to run pipeline recipes. This is in the most simple case due to the `SoF Splitter`/`SoF Accumulator`[7], which allow to process calibration data from different setting within one given DataSet (e.g. bias frames with different binnings, dark frames with different DIT). More complex composite actors contain several actors (e.g. `RecipeExecuter`).

The central elements of any Reflex workflow are the `RecipeExecuter` actors that actually run the recipes. One basic way to embed a `RecipeExecuter` in a workflow is shown in Fig 5.2, which is one of the most simple version of a composite actor within the X-shooter workflow. The reader for the moment should pay attention only to the core functionality where a `RecipeExecuter` is preceded by an `SofSplitter`, and followed by an `SofAccumulator`; the rest of the workflow implements extra functionality to decide whether the dark recipe should be triggered or not, depending on the existence of raw dark frames. The function of the `SofSplitter` is to investigate the incoming SoFs, sort them by "purpose", and create separate SoFs for each purpose. The `RecipeExecuter` then processes each of the SoFs independently. Finally, the `SofAccumulator` packs all the results into a single output SoF. The direct relation between the `SofSplitter` and `SofAccumulator` is used to communicate the number of different SoFs created by the `SofSplitter`. A workflow will only work as intended if the purpose of all the files a recipe needs as input is

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

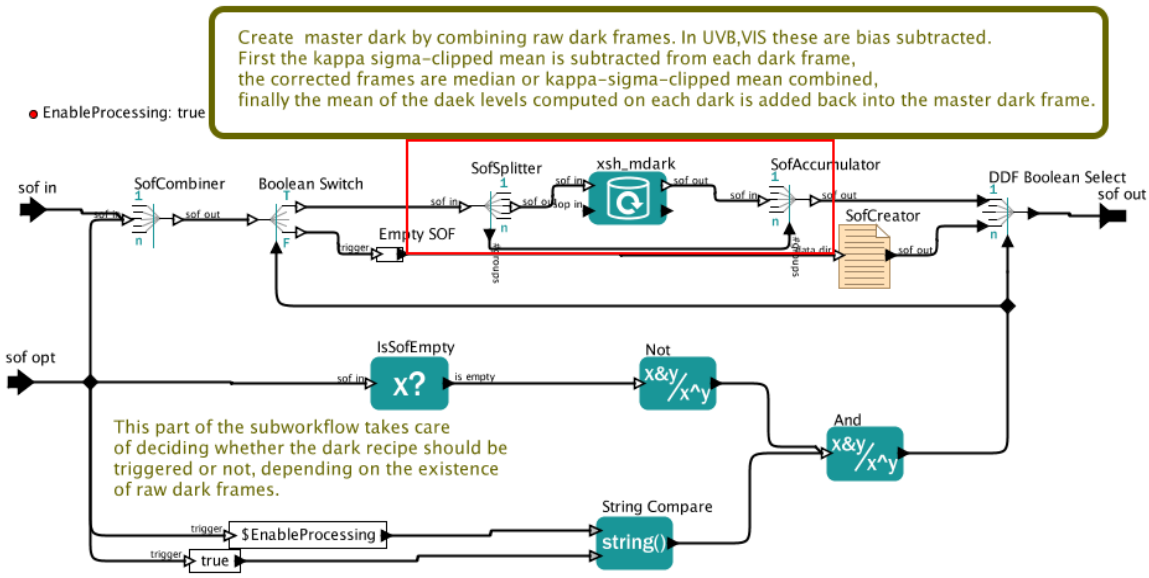


Figure 5.2: *This is the window you get when you choose* `Open Actor` *for the composite actor* `Master Dark Creation`*. This is the most simple case for a composite actor in the X-shooter workflow and the red rectangle marks the most simple composite actor that is possible..*

identical. The only exception to this rule is that a purpose can also be "default". In this case, the file is included in any output SoF created by the `SoFsplitter` and `SofAccumulator`.

The reason for this scheme is best explained by an example. For a complex DataSet, the `Data Organiser` might have selected a large number of individual raw bias frames. The different bias frames are to be used to de-bias different frames, e.g. the science frames, the various arc lamp and flat field frames and the standard star frames. The `Data Organiser` determines and records this "purpose" of each bias, and this information is included in the DataSet and each SoF created from this DataSet. The `FitsRouter` directs all raw biases to the `Master BiasDark Creation` composite actor. The `SofSplitter` then creates SoFs, one for the biases to be used for the science images, and (possibly) separate ones for the biases to be used for the calibration frames. The `xsh_mbias` recipe creates one master bias for each SoF, and the `SofAccumulator` then creates a SoF that contains all the master biases.

Additional complexity within the X-shooter workflow is created by the combination of three different arms (e.g. `MasterBiasDark Creation`, which uses `xsh_mbias` to create master bias frames for UVB and VIS data and `xsh_mdark` to create master dark frames for NIR data) and different observing modes (e.g. `Instrument Response`, which uses `xsh_respon_slit_stare`, `xsh_respon_slit_nod`, or `xsh_respon_slit_offset` to create a response curve, depending on the observing mode used for the flux standard star). Table 5.1 shows which actors contain which recipes and what they do.

A `RecipeExecuter` actor is used in the workflow to run a single X-shooter pipeline recipe (e.g: in the `Master Bias Creation` actor the recipes `xsh_mbias` and `xsh_mdark` are executed). In order to configure the `RecipeExecuters`, one has to first use `Open Actor` (possibly repeatedly) to get to the level of the recipe executors (see Figs. 5.3 and 5.4).

Table 5.1: The X-shooter pipeline actors and their contents

actor	recipes	description
Master BiasDark Creation	`xsh_mbias` `xsh_mdark`	create master bias (UVB/VIS) create master dark (NIR)
Spectral Format	`xsh_predict` `xsh_orderpos`	spectral format check determine centers of orders
Master Flat Creation	`xsh_mflat`	create master flat and table for order edges
2D Geometry	`xsh_2dmap`	calibration of wavelength and spatial distortion

Instrument Response	`xsh_respon_slit_stare`
	`xsh_respon_slit_nod`
	`xsh_respon_slit_offset`
		determine response function

Flexure Correction	`xsh_flexcomp`	determine line shift due to instrument flexure

Science Reduction	`xsh_scired_slit_stare`
	`xsh_scired_slit_nod`
	`xsh_scired_slit_offset`
	`xsh_scired_ifu_stare`
	`xsh_scired_ifu_offset`
		reduce science data

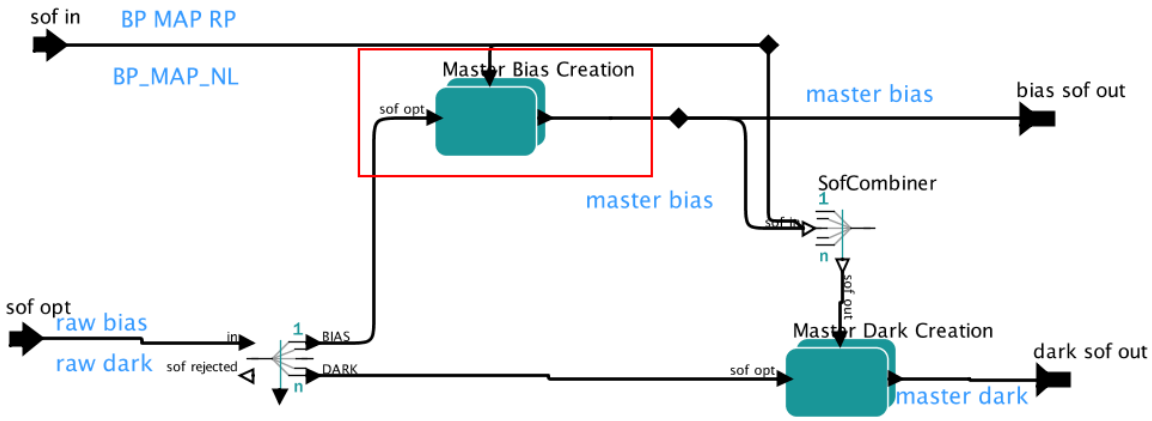


Figure 5.3: *This is the window you get when you choose* `Open Actor` *for the composite actor* `Master BiasDark Creation`. *Using* `Open Actor` *on* `Master Bias Creation` *(marked by red rectangle) gives you Fig. 5.4.*

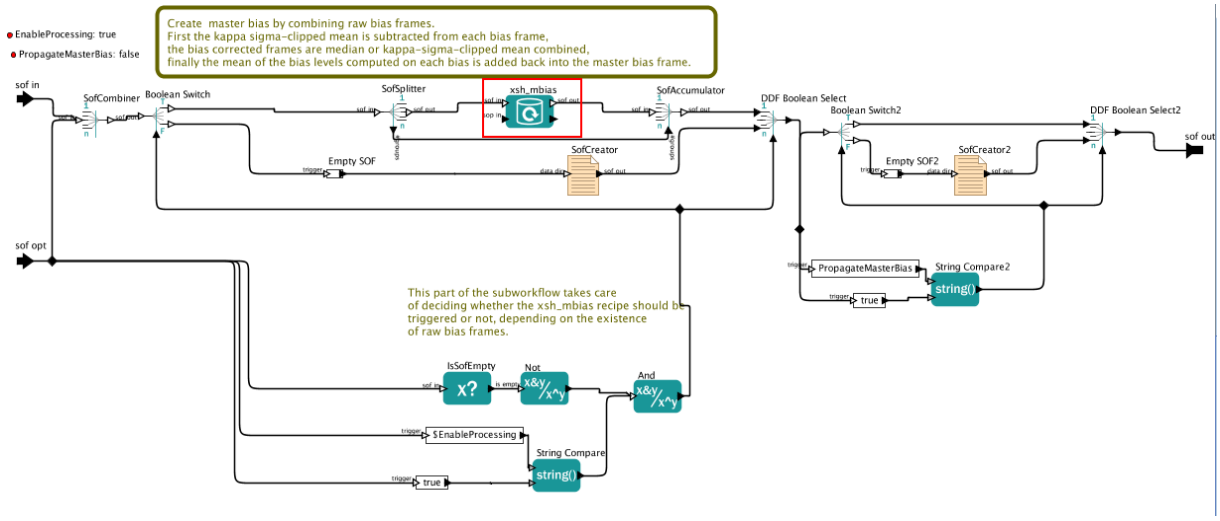


Figure 5.4: *This is the window you get when you choose* `Open Actor` *for the composite actor* `Master Bias Creation` *within* `Master BiasDark Creation`. *Using* `Configure Actor` *on* `xsh_mbias_1` *(marked by red rectangle) gives you Fig. 5.5.*

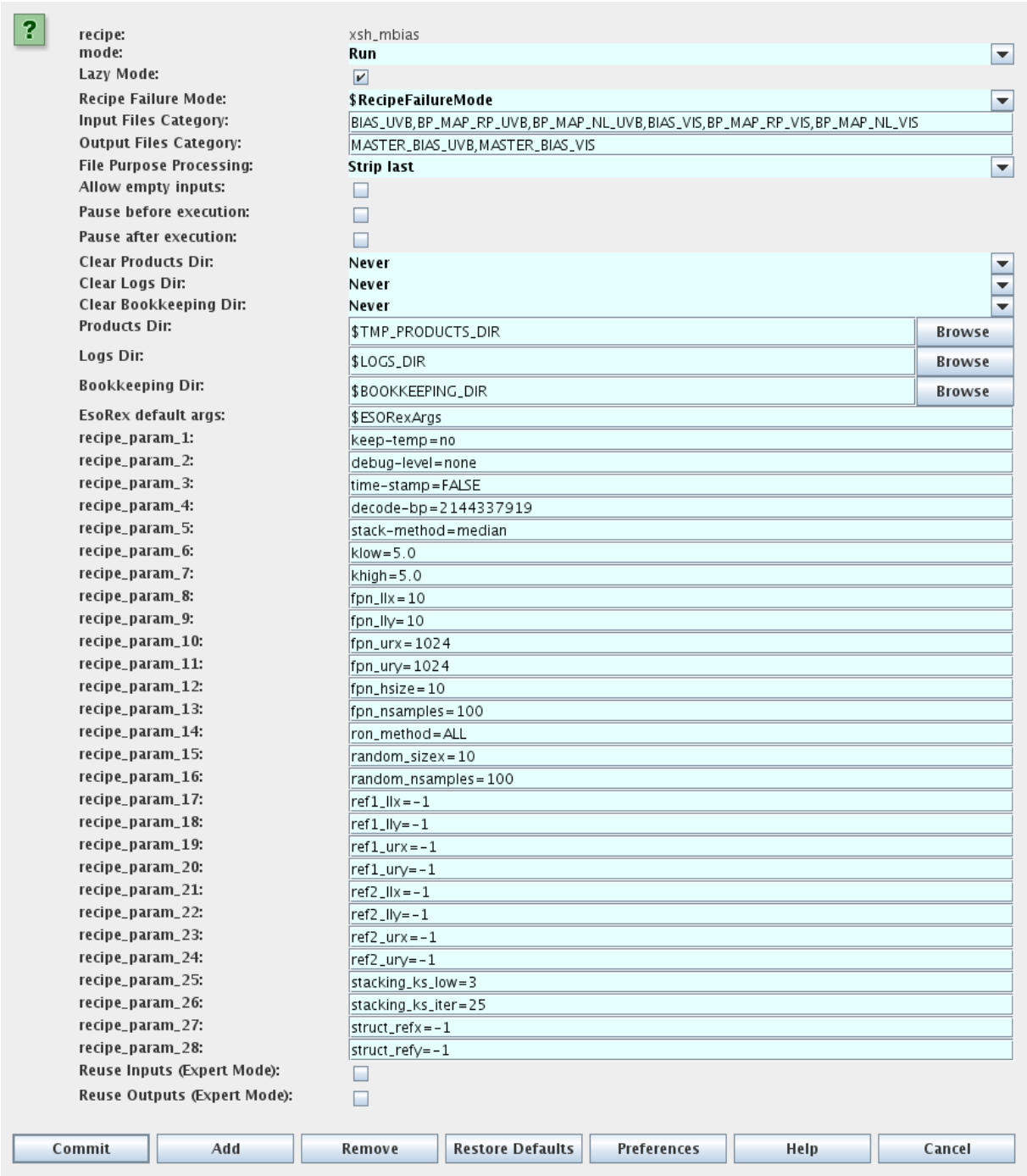


Figure 5.5: *The "Edit parameters" window for a typical* `RecipeExecuter` *actor, the* `xsh_mbias_1` *actor which runs the* `xsh_mbias` *pipeline recipe.*

In Figure 5.5 we show the "Edit parameters" window for a typical `RecipeExecuter` actor, which can be displayed by right-clicking on the actor and selecting `Configure Actor`. In the following we describe in more detail the function of some of the parameters for a `RecipeExecuter` actor:

- The "recipe" parameter states the X-shooter pipeline recipe which will be executed.
- The "mode" parameter has a pull-down menu allowing the user to specify the execution mode of the actor. The available options are:
  - `Run`: The pipeline recipe will be executed, possibly in Lazy mode (see Section 5.2.4). This option is the default option.
  - `Skip`: The pipeline recipe is not executed, and the actor inputs are passed to the actor outputs.
  - `Disabled`: The pipeline recipe is not executed, and the actor inputs are not passed to the actor outputs.
- The "Lazy Mode" parameter has a tick-box (selected by default) which indicates whether the `RecipeExecuter` actor will run in Lazy mode or not. A full description of Lazy mode is provided in the next section.
- The "Recipe Failure Mode" parameter has a pull-down menu allowing the user to specify the behaviour of the actor if the pipeline recipe fails. The available options are:
  - `Stop`: The actor issues an error message and the workflow stops.
  - `Continue`: The actor creates an empty output and the workflow continues.
  - `Ask`: The actor displays a pop-up window and asks the user whether he/she wants to continue or stop the workflow. This option is the default option.

  This parameter should be set to `$RecipeFailureMode` to use the global parameter `RecipeFailureMode`.
- The set of parameters, which start with "recipe_param_" and end with a number, corresponds to the parameters of the relevant X-shooter pipeline recipe. By default in the `RecipeExecuter` actor, the pipeline recipe parameters are set to their pipeline default values. If you need to change the default parameter value for any pipeline recipe, then this is where you should edit the value. `PORT` as parameter value marks parameters that can be changed in interactive windows. For more information on the X-shooter pipeline recipe parameters, the user should refer to the X-shooter pipeline user manual (Modigliani 2012[8]).

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

[8] Available from ftp://ftp.eso.org/pub/dfs/pipelines/xsh/xsh-pipeline-manual-7.0.pdf

5.2.4 Lazy Mode

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

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

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

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

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` X-shooter. 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 xshooter
   ```

   to launch the Product Explorer. The Product Explorer allows you to inspect the data products already reduced by the X-shooter `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> xshooter
   ```

   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 6.3.3

3. Type:

   ```
   esoreflex xshooter &
   ```

   to launch the X-shooter `esoreflex` workflow. The X-shooter workflow window will appear (Fig. 3.2). 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 6.3.3 for how to configure this option).

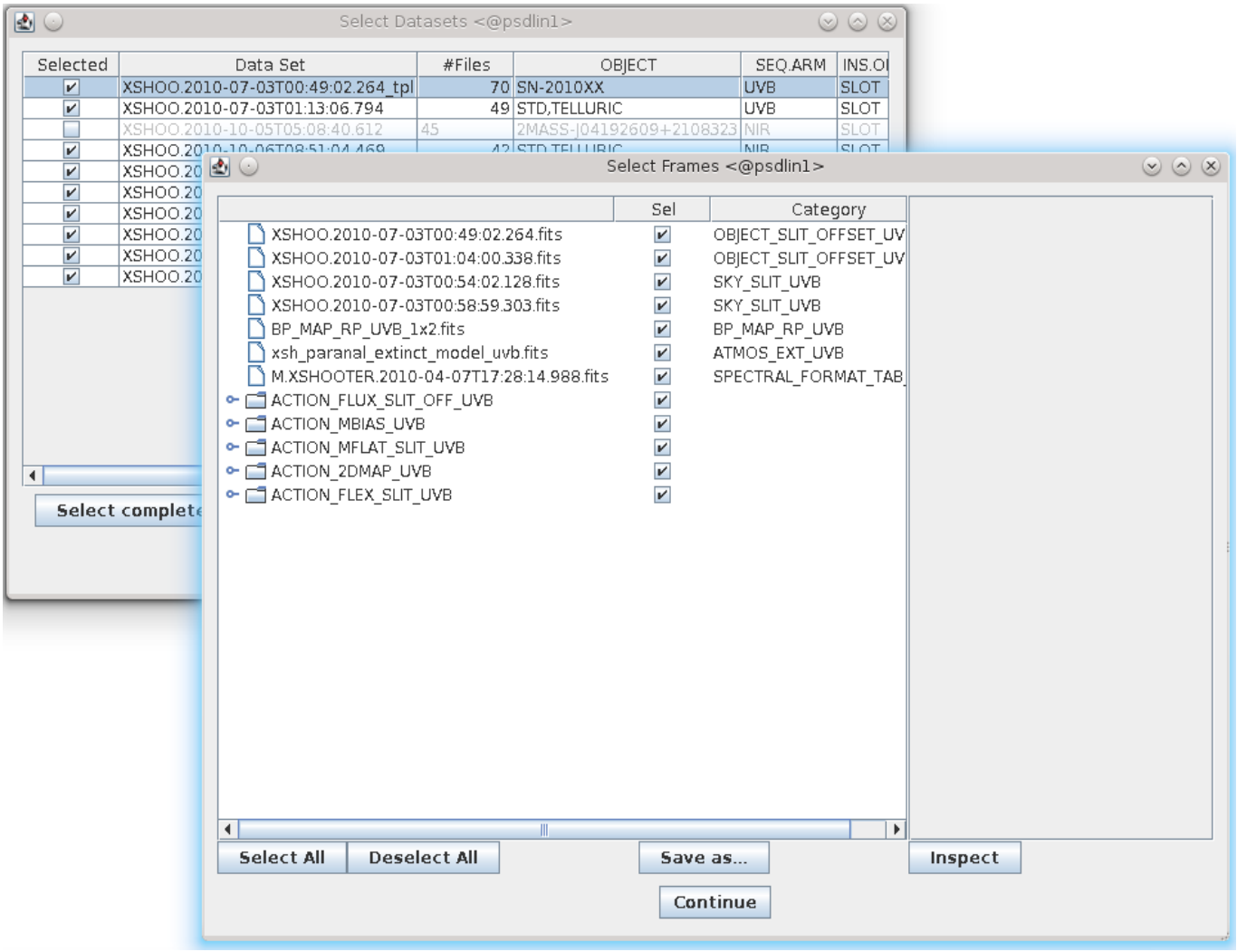


Figure 6.1: *The "Select Frames" window with a single file from the current Data Set highlighted in blue, and the corresponding FITS header displayed in the text box on the right. Hidden partially behind the "Select Frames" window is the "Select DataSets" window with the currently selected DataSet highlighted in blue.*

6.3 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[9] 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.

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

[9] 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`

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. 3.3).

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[10], the file category (derived from the FITS header), and a selection tick box in the right column. The tick boxes allow you to edit the set of files in the DataSet which is useful if it is known that a certain calibration frame is of poor quality (e.g: a poor raw flat-field frame). The list of files in the DataSet may also be saved to disk as an `ASCII` file by clicking on `Save As` and using the file browser that appears.

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

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

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

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

6.3.3 The ProductExplorer

The ProductExplorer is an interactive component in the `esoreflex` workflow whose main purpose is to list the final products with the associated reduction tree for each dataset and for each reduction attempt (see Fig. 3.4).

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

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.

Exploring the data reduction products

The left window of the ProductExplorer GUI shows the executions for all the datasets (see Fig. 3.4). 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.

6.3.4 Creation Of Master Calibration Files

In this step of the workflow, the following X-shooter recipes are executed in the order listed below. Please refer to the X-shooter pipeline user manual (Modigliani 2012: Sections 9, 10 & 11) for the details of each recipe and the algorithms employed:

1. The `Master BiasDark Creation` actor will execute the X-shooter pipeline recipes `xsh_mbias` (or `xsh_mdark`) in order to create a combined master bias (or dark) frame from the set of raw bias (or dark) frames from the UVB/VIS (or NIR) arms. By default the master bias is produced by the workflow for UVB/VIS arms but it is not fed to other recipes because we verified that to use instead the overscan correction provides usually more accurate results. If a user would like instead to use the master bias, the user has to edit the `Master Bias Creation` actor configuration within the `Master BiasDark Creation` composite actor, and change the actor parameter `PropagateMasterBias` from false to true.

2. The `Spectral Format` actor will execute the X-shooter pipeline recipe `xsh_predict` in order to compute initial guesses for the wavelength solution and order positions from a 1-pinhole arc lamp frame. This is done by employing a physical model of the X-shooter instrument combined with the information on the atmospheric pressure, temperature and corresponding instrument setting stored in the FITS header of the input 1-pinhole arc-lamp frame. It then executes the X-shooter pipeline recipe `xsh_orderpos` to determine the central positions of the orders from a 1-pinhole flat field.

3. The `Master Flat Creation` actor will execute the X-shooter pipeline recipe `xsh_mflat` in order to create from the set of raw flat frames a combined master flat frame and a table describing the edges of each order as measured on the master flat field. For IFU flat fields the edges of the slices are traced.

6.3.5 2D Distortion Mapping

In this step of the workflow, the `2D Geometry` actor will execute the X-shooter pipeline recipe `xsh_2dmap` on a 9-pinhole arc lamp frame to create a wavelength calibration and spatial distortion solution. This recipe updates the configuration file for the X-shooter physical model received from the `Spectral Format` actor (results of `xsh_predict`), which is then used in the `Flexure Correction` actor (recipe `xsh_flexcomp`) and the `Instrument Response` actor. Please refer to the X-shooter pipeline user manual (Modigliani 2012: Sections 9, 10 & 11) for the details and the algorithms employed.

6.3.6 Response Determination, Science Reduction

In this step of the workflow, the `Instrument Response` and the `Science Reduction` actors will determine the response function (if a flux standard star is provided) and perform the science reduction. Depending on the value of the parameter `UseResponseFlat` in the `FlatStrategy` actor either the flat field of the science data (`false`, default) or the standard star's own flat field (`true`) will be used to process the flux standard star observation. See Sect. 7.5 for the background of this procedure. The science recipes are also used to process any telluric standard stars if they are available. Please refer to the X-shooter pipeline user manual (Modigliani 2012: Sections 9, 10 & 11) for the details of these recipes and the extraction algorithms employed. The flux-calibration of the science spectrum will be carried out if an appropriate instrument response curve (master or otherwise) has been provided as an input to the `Science Reduction` actor.

The `Instrument Response` actor will execute one of the X-shooter pipeline recipes `xsh_response_stare`, `xsh_response_nod`, `xsh_response_offset` (depending on the observing mode used for the flux standard star)[11] in order to create an instrument response curve from the observation of a standard star (if it is listed in the standard star catalogue), which will subsequently be used to flux-calibrate the science observation.

After the spectrum extraction, the actor will display an interactive window to allow the user to inspect the extracted spectrum and the response curve (see Figure 7.7, p. 50).

Note that this actor will be skipped if there are no observations of a standard star in the current DataSet. A standard star observation is only included in the DataSet for a science observation by calSelector if it was taken within ±30 nights of the science observation. Generally, X-shooter PI-Packs as well as calSelector data sets for

[11] The pipeline does not determine response curves for IFU flux standard star observations.

data taken after 2011-07-15 are supplied with master instrument response curves which will also be included in the DataSet for a science observation. Due to variations of the flat fields with time, which influence the response, we strongly discourage the use of master response curves delivered by calSelector for UVB and VIS data. The `Science Reduction` actor will execute the X-shooter pipeline recipes `xsh_scired_slit_stare`, `xsh_scired_slit_nod`, `xsh_scired_slit_offset`, `xsh_scired_ifu_stare`, or `xsh_scired_ifu_offset` (depending on the observing mode used for the science data) to extract and rectify the orders and apply sky subtraction. The X-shooter workflow will flux-calibrate the slit mode science observations using the instrument response curve derived from the standard star observation if it exists in the current DataSet. If both a master response and an individual response curve are provided the pipeline will use the individual response.

If there is no standard star observation in the current DataSet, then the science observation will not be flux-calibrated. As mentioned before the pipeline does not create response curves for IFU data and they are therefore not flux-calibrated.

After the spectrum extraction, the actor will display an interactive window to allow the user to inspect the extracted spectrum and assess the extraction quality (see Figure 6.2).

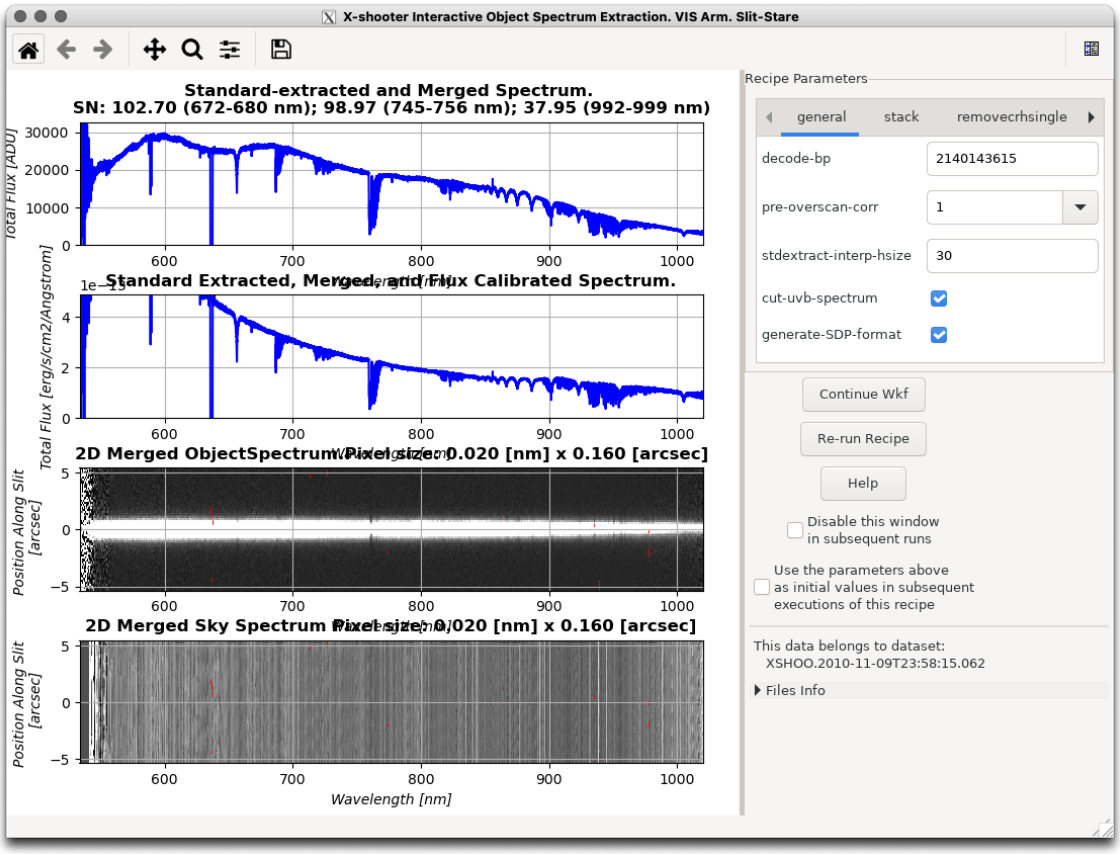


Figure 6.2: *The interactive pop-up window for the* `Science Reduction` *actor and X-shooter pipeline recipe* `xsh_scired_slit_stare`*. The extracted and merged spectrum for the telluric standard star in the VIS DataSet is displayed in the top panel..*

6.3.7 Output Organisation

After having processed the input data for a DataSet, the workflow highlights and executes the `Product Renamer` actor, which, by default, will copy the defined final products of the X-shooter `Science Reduction` actor to the directory specified by `END_PRODUCTS_DIR` and rename them with names derived from the values of certain FITS header keywords. Specifically, final products are renamed by default with names of the form `<HIERARCH.ESO.OBS.NAME>_<HIERARCH.ESO.PRO.CATG>.fits`, where `<HIERARCH.ESO.OBS.NAME>` and `<HIERARCH.ESO.PRO.CATG>` represent the values of the corresponding FITS header keywords. These names are fully configurable by right-clicking on the `Product Renamer` actor, selecting `Configure Actor`, and then editing the string as appropriate.

The final products that are copied and renamed are:

- `<HIERARCH.ESO.OBS.NAME>_SCI_SLIT_FLUX_MERGE1D_<ARM>.fits` - The flux-calibrated, extracted and merged science spectrum. This product is only generated if an appropriate instrument response curve (master or otherwise) was used as an input to the `Science Reduction` actor.
- `<HIERARCH.ESO.OBS.NAME>_SCI_SLIT_FLUX_MERGE2D_<ARM>.fits` - The flux-calibrated, rectified and merged 2-dimensional science spectrum. This product is only generated if an appropriate instrument response curve (master or otherwise) was used as an input to the `Science Reduction` actor.
- `<HIERARCH.ESO.OBS.NAME>_SCI_SLIT_MERGE1D_<ARM>.fits` - The extracted and merged science spectrum.
- `<HIERARCH.ESO.OBS.NAME>_SCI_SLIT_MERGE2D_<ARM>.fits` - The merged 2-dimensional science spectrum.
- `<HIERARCH.ESO.OBS.NAME>_SKY_SLIT_MERGE1D_<ARM>.fits` - The merged 2-dimensional sky spectrum (not for `nodding` mode data).
- `<HIERARCH.ESO.OBS.NAME>_SCI_SLIT_IDP_<ARM>.fits` - The corresponding final 1D extracted spectra in a format compatible with the Science Data Product Standard needed to submit files to ESO's Science Archive Facility. The standard itself is described in http://www.eso.org/sci/observing/phase3/p3sdpstd.pdf and tips on how to handle the data with common tools are here http://archive.eso.org/cms/eso-data/help/1dspectra.html.

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

7 Optimising Your Results Through Workflow Interaction

In this section, we use the information from previous sections along with the X-shooter demo data supplied with `Reflex 3.12` to illustrate how to optimise the scientific products in terms of quality and S/N. This is achieved by interaction with the workflow actors via interactive windows displayed at key data reduction points in the data flow, which enable iteration of certain recipes in order to obtain better results.

We recommend that the user has already carried out the reductions for all the data sets as described in Section 3, although this is not a pre-requisite to following this section. By doing this, the user will be taking advantage of the workflow Lazy Mode, with minimal waiting time between various pipeline recipe executions.

Please follow the steps in the next subsections in order to optimise the reductions for the X-shooter demo data supplied with `Reflex 3.12`.

The reduction steps from format check to wavelength calibration are generally very robust and need no optimization. Readers interested in the details of the corresponding actors are refered to Appendix A (p. 71).

7.1 Interactive Windows

All interactive windows in the X-shooter workflow have the same basic structure. The left part is occupied by displays, while the right part provides access to selected pipeline recipe parameters and information about files used to create the currently displayed results.

Help for each of these features may be accessed by leaving the mouse pointer for about one second over the plot/parameter/button for which information is required. Note that the recipe parameters are grouped via labels at the top of the "Recipe Parameters" box that may be left-clicked to reveal the associated parameters.

7.1.1 Buttons

The panel of buttons at the top-left of the interactive window may be used to manipulate the displayed plots. The buttons have the following functions:

- - Reset all the plot ranges to their original values.
- - Undo the last modification of the plot ranges (if possible).
- - Redo the next modification of the plot ranges (if possible).
- - Selecting this button allows the user to use the mouse to shift the plot ranges by left-clicking on the target plot canvas and then dragging the mouse around while keeping the left mouse button held down, and releasing when ready.
- - Selecting this button allows the user to zoom in on each plot by left-clicking on the target plot canvas to mark the top-left corner of a rectangle and then dragging the mouse to the bottom-right corner of the rectangle and releasing. The plot ranges will then be modified to match the rectangle that was defined.

The following constraints can be made by holding simultaneously a key while pressing the left mouse button:

Constrain pan/zoom to x axis	hold X
Constrain pan/zoom to y axis	hold Y
Preserve aspect ratio	hold Ctrl

- - Clicking this button opens a "Configure subplots" window that allows the user to adjust the spacing and positioning of the individual plots.
- - Clicking this button opens a "Save to file" window which allows the user to save a screenshot of the current interactive window.
- - Clicking this button allows the user to change the display levels of an image by clicking on pixels within the image (similar to `ds9`).

Use these buttons to inspect the plots in the interactive window in more detail.

7.1.2 Plots

The plots should be checked carefully if they show any unexpected features, e.g. much lower or higher noise than expected, gaps, remaining sky lines. In such a case please check Sect. 8.1, p. 62 to see if you find the problem described.

Note that in general for the images displayed in the workflow interactive windows, pixels that are flagged as bad (according to the value of `decode-bp`) in the third extension of the corresponding FITS file are overlaid in the images using a red hue. In case of reduction of stare data using the Kelson (sky-method=BSPLINE) method, as the number of pixels where the sky model is inaccurate or model outliers are detected is high, the pixels corresponding to these quality codes are filtered out from the display of the 2D sky subtracted resampled frame.

7.1.3 Recipe Parameters

Left-clicking on the tabs on the right provides a list of configurable recipe parameters. To change a parameter left-click on the field with the value besides the parameters name. Then click the `Re-run recipe` button to re-run the recipe with the new parameter setting, which will close the interactive window and process the data again with the new parameter settings. The new results will appear ina new version of the interactive window.

7.2 Reducing Stare Mode Observations

The `Science Reduction` actor executes the X-shooter pipeline recipes `xsh_scired_slit_stare`, `xsh_scired_slit_offset`, or `xsh_scired_slit_nod` as appropriate. For the second DataSet, the science object observation is in stare mode, and so the `Science Reduction` actor executes the `xsh_scired_slit_stare` recipe and displays the corresponding interactive window (see Figure 7.1). For details of the processing steps that are performed by this recipe, please see Section 11 of the pipeline manual (Modigliani 2012).

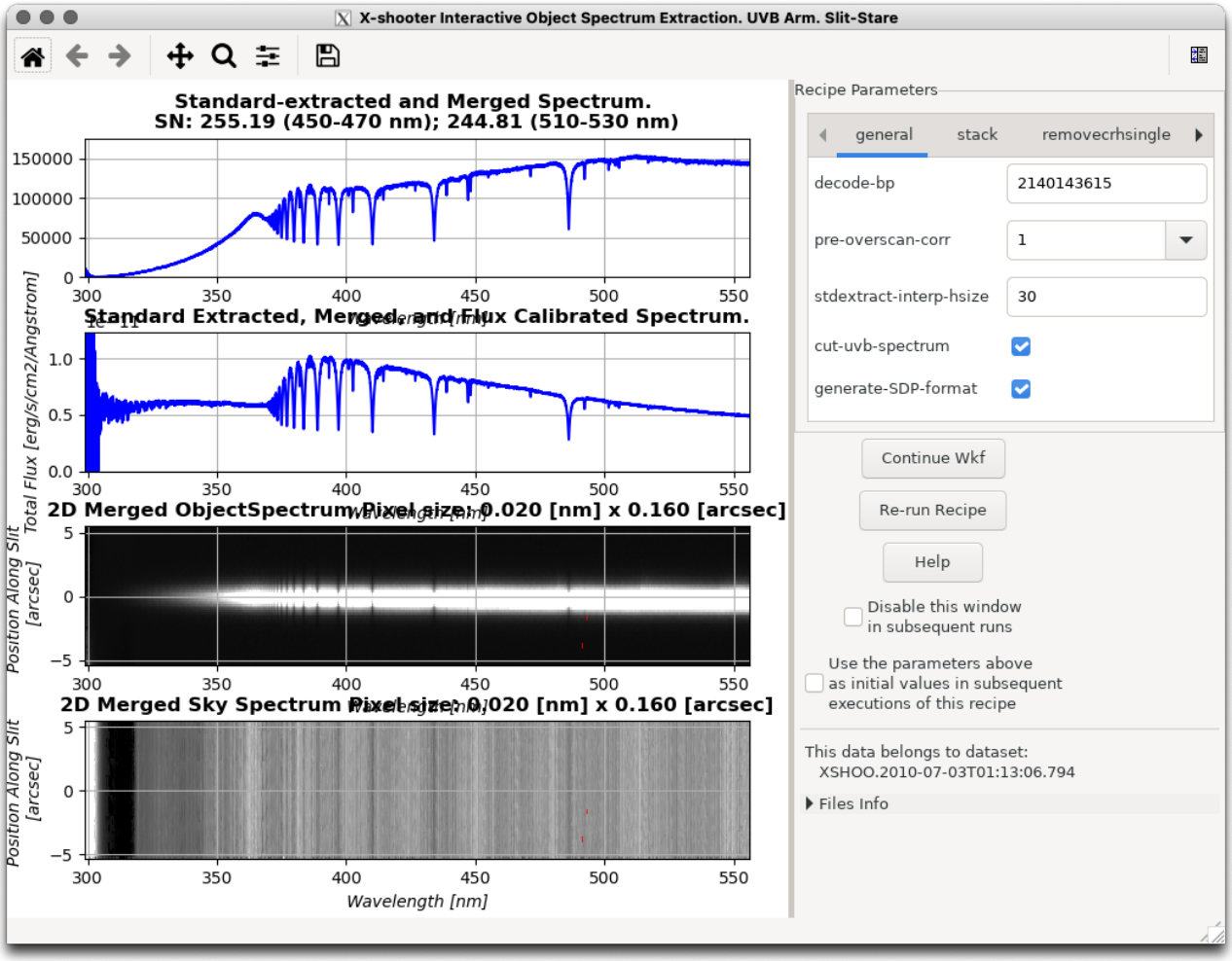


Figure 7.1: *The interactive window for the* `Science Reduction` *actor for the recipe* `xsh_scired_slit_stare`.

To prepare for the reduction of stare data only proceed as follows:

- Since we will discuss the optimisation of the flux calibration in Section 7.5, it is more efficient for now to disable the interactive mode for this actor. Do this by double-clicking on the composite actor, setting the "EnableInteractivity" parameter to `false`, and clicking `Commit` to save the changes to the workflow.

- Click the button.

- In the "Select Datasets" window that pops up, click the button `Deselect all` and then select the DataSet `XSHOO.2010-07-03T01:13:06.794` by ticking the corresponding box in the first column. This DataSet is a UVB arm observation taken in stare mode. Click the `Continue` button.

7.2.1 Interactive Window

The **top plot** in the interactive window show the extracted (using standard extraction) and merged one-dimensional spectrum of the science object (blue) with ±1-sigma uncertainties plotted as the light blue region encompassing the object spectrum (and bounded by the light grey spectra). The **second plot** shows the flux-calibrated spectrum in the same way. The **third plot** plot shows the corresponding two-dimensional rectified (and merged) object spectrum as an image with flagged pixels overlaid using a red hue. The **bottom plot** shows the two-dimensional rectified (and merged) sky background spectrum model that was subtracted from the object spectrum before order rectification.

The plots should be checked carefully for any unexpected features, e.g. much lower or higher noise than expected, gaps, remaining sky lines. In such a case please check Sect. 8.1, p. 62 .

The one-dimensional merged spectra that are shown in the interactive windows of the X-shooter workflow also indicate which wavelengths have been flagged for some property in the third extension of the corresponding FITS file. Wavelengths that have any positive flag value apart from zero (i.e. any property, good or bad) are marked with a dark-grey vertical bar stretching between the ±1-sigma limits for those wavelengths. Wavelengths that have an associated flag value that is decoded as including a bad property are marked with a red vertical bar stretching between the ±1-sigma limits. By "decoded as a bad property", we mean that a bitwise-AND comparison is made between the pixel flag value and the value of the pipeline recipe parameter `decode-bp`, and the result is greater than zero (i.e. at least one of the "bad" properties encoded in the value of `decode-bp` is also present in the pixel flag value).

7.2.2 Improving the result

The interactive window for the `Science Reduction` for data observed in stare mode provides access to all of the parameters of the `xsh_scired_slit_stare` recipe. Rather than describing each of these parameters in detail (see the pipeline manual Modigliani 2012), we provide below some hints on how to improve results, which are ordered by the tabs shown in the right part of the interactive window.

general The parameters in this tab are common to all slit science recipes.

`decode-bp` Zoom in on the one-dimensional and two-dimensional spectra between 491-499 nm for the current DataSet (see Figure 7.2). You will find that there are four regions of flagged pixels in the two-dimensional spectrum. The pipeline performs by default standard extraction between ±2.0 arcsec (controlled by `localize-slit-hheight`) around the centre of the slit (controlled by `localize-slit-position`). Since two of these flagged pixel regions are outside of the extraction region, they do not affect the standard extraction algorithm. The remaining flagged pixel regions contain bad pixels that are interpolated during the standard extraction algorithm by constructing a local spatial profile determined from the wavelengths within `stdextract-interp-hsize` grid points of the wavelength for which interpolation is required. The output fluxes for the wavelengths where interpolation has occurred during standard extraction are flagged with the code `4194304` which is not considered "bad" for the default value of `decode-bp` and hence the one-dimensional spectrum displays dark-grey bars of length $\pm 1\sigma$ at these wavelengths. Due to the high signal-to-noise ratio you have to zoom in to see the bars. Add `4194304` to the value of `decode-bp` in the "Recipe Parameters" box and re-run the science reduction by clicking the `Re-run Recipe`

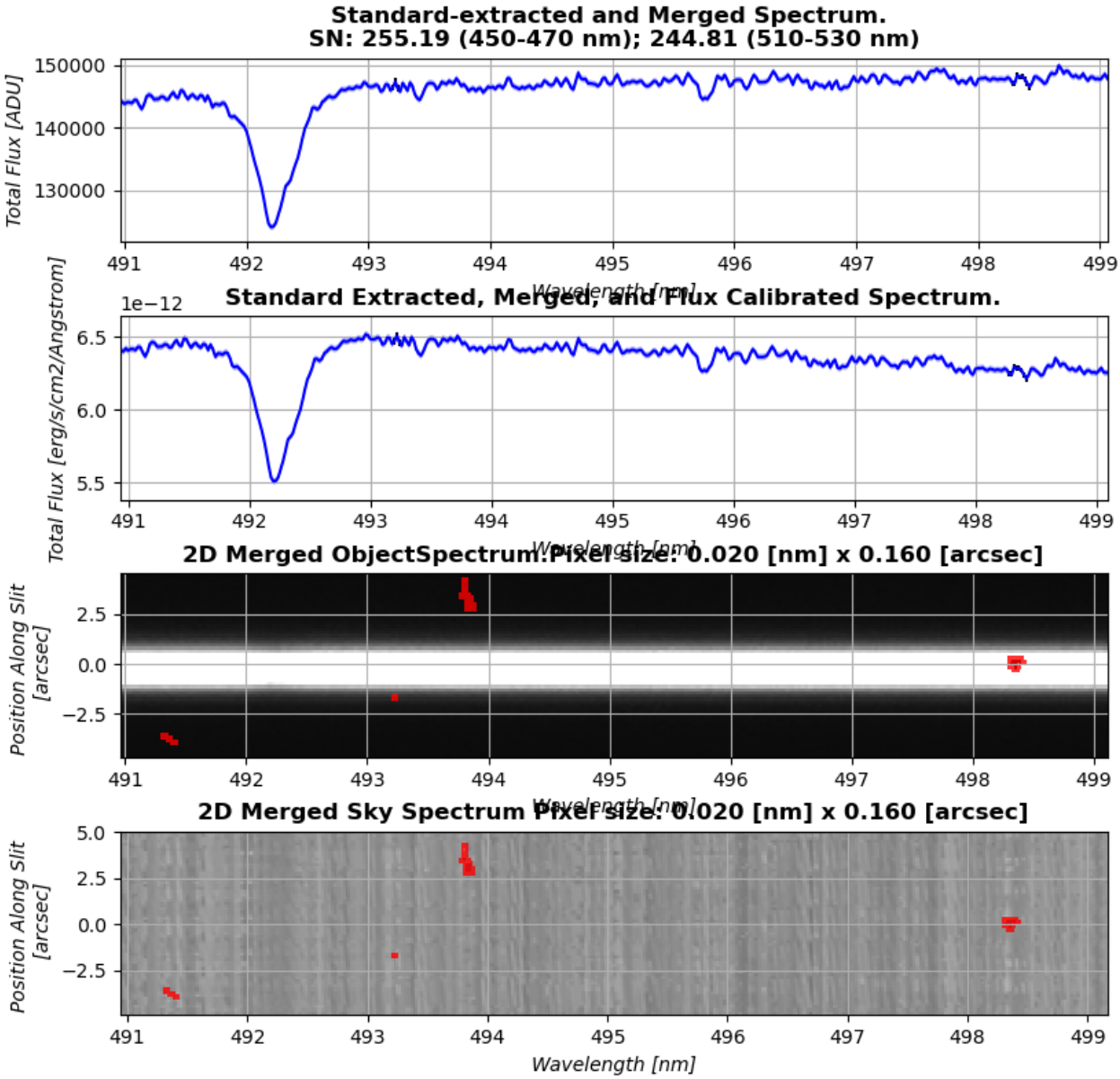


Figure 7.2: *Zoom of the one-dimensional and two-dimensional spectra between 491-499 nm after setting* `decode-bp = 2140143615.`.

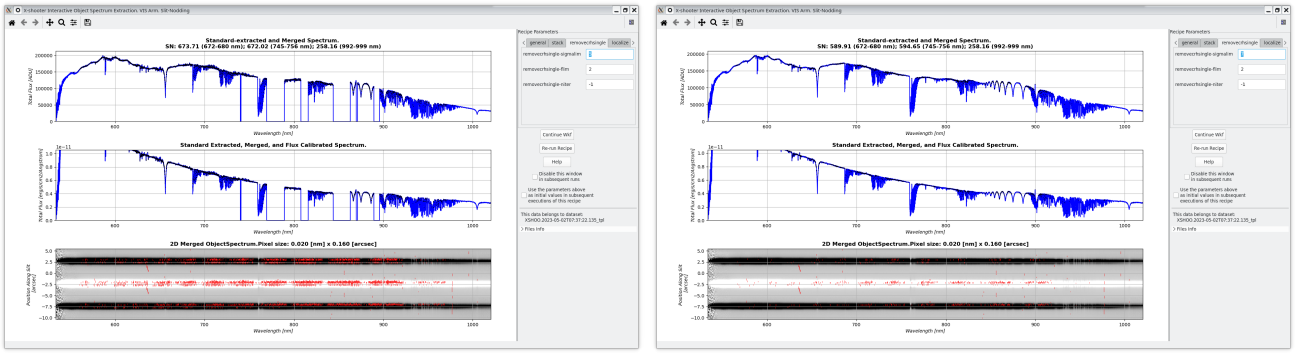


Figure 7.3: Here we show the `VIS` spectrum of the telluric standard star Hip089684 observed on May 1, 2023 (`XSHOO.2023-05-02T07:37:22.135`) with very good seeing, where the default value for `removecrhsingle-sigmalim` has to be increased to avoid spurious detections of cosmic ray hits. **Left**: `removecrhsingle-sigmalim = 5.0` **Right**: `removecrhsingle-sigmalim = 7.0`

button. When the interactive window appears again, the fluxes at the wavelength of ∼498.35 nm in the one-dimensional spectrum will now display dark-grey bars, which is useful for alerting the user to the fact that the fluxes were calculated using some interpolation.

`stdextract-interp-hsize` In some cases it may be necessary to increase the value of `stdextract-interp-hsize` when a region of bad pixels is too large to allow the interpolation within the standard extraction (see Section 7.4 for an example). However, there are only a few small regions of bad pixels in the science object frame for the current DataSet and therefore the default value of `stdextract-interp-hsize` is robust.

Since there is only one input science object frame for the current DataSet, the parameters for image combination under the tab **stack** are not relevant here.

removecrhsingle The parameters in this tab control the application of the van Dokkum algorithm (2001, PASP, 113, 1420) for the detection of cosmic ray contaminated pixels to the individual input bias-corrected science object frames.

`removecrhsingle-sigmalim` This parameter is set by default to the value of `5.0` to detected cosmic ray hits for `UVB` and `VIS` spectra. For `NIR` the cosmic ray detection is by default switched off by setting `removecrhsingle-niter` to `0`.

Figure 7.3 shows the potentially negative effect of using `removecrhsingle-sigmalim = 5.0` for spectra observed with very good seeing and how to address the problem.

The parameters under the tab **background** generally do not need changing since the default values for these parameters are already optimised for the vast majority of input data.

localize The parameters under this tab control the integration limits along the slit for the science object spectrum and the sky background.

`localize-method` This parameter defines the method used to determine the extraction window: `MANUAL`, `MAXIMUM`, or `GAUSSIAN`. For more details on the `MAXIMUM` and `GAUSSIAN` methods, please see the pipeline manual. Be aware that these two methods do not work well for low S/N data because of the difficulty with automatic localisation of such spectra within the slit, and sometimes they do not work well (or they cause a crash) for higher S/N data. The `MANUAL` method for spectrum localization within the slit is controlled by two parameters

- `localize-slit-position` defines the central position of the extraction slit along the y-axis (in arcsec), which is by default 0 (= slit centre).
- `localize-slit-hheight` defines the half size of the extraction slit window around `localize-slit-position` (in arcsec) and is by default 2.

Set `localize-slit-hheight` to `1.9` and `localize-slit-position` to `-0.2` to change the extracted spectrum S/N in the wavelength regions 450-470 nm and 510-530 nm. Feel free to continue experimenting with these parameters to see their influence on the S/N.

rectify

`rectify-bin-lambda` This parameter may be used to control the output wavelength step of the one-dimensional extracted spectrum. However, be aware that increasing the wavelength step will not serve to bin the output spectrum in order to obtain higher S/N since the rectification algorithm in the X-shooter pipeline only *samples* from the pixel values in the non-rectified orders. Binning of output spectra should be done by the user independently of the X-shooter pipeline as a post-processing step. Note that these comments also apply to the same parameters in the nodding and offset mode pipeline recipes.

optextract An implementation of optimal extraction (Horne K., 1986, PASP, 98, 609; Mukai K., 1990, PASP, 102, 183) is available in the X-shooter pipeline (for the stare recipes only). The code for the optimal extraction is written by Goldoni et al. (ADASS XXI, 2012, ASP Conf. Ser. Vol. 461, Page 741) and does not include any treatment of bad pixels. The corresponding algorithm and its implementation have not been investigated in detail, although some testing by ESO (for UVB) indicates that a ∼10-20% decrease in the noise compared to standard extraction may be obtained for point sources in selected wavelength regions, but between these clear regions significant artifacts are introduced. Furthermore, the response curve is not applied to the optimally extracted spectrum. Optimal extraction may be switched on by setting `do-optextract` to `TRUE`[12]. See the pipeline manual Section 11 for more details.

sky This final set of parameters corresponds to the sky background subtraction.

`sky-method` This parameter allows the user to choose between two methods for fitting the sky background as a function of wavelength: `MEDIAN` or `BSPLINE`. The `BSPLINE` option instructs the pipeline to use an implementation of the Kelson (2003, PASP, 115, 688) method to determine the sky background. However, this method is very difficult to tune in order to get good sky subtraction results, and for this reason it is not the default sky subtraction algorithm for the stare recipes in the X-shooter pipeline. In this tutorial we concentrate on optimising the results for the `MEDIAN` method.

`sky-median-hsize` For `sky-method` set to `MEDIAN`, the pipeline constructs a vector of pixel values from the sky background regions of the two-dimensional (non-rectified) object spectrum as a

[12]The optimally extracted spectrum is only available in the `TMP_PRODUCTS_DIR/xsh_scired_slit_stare/<time_stamp>` and will not be displayed in the interactive window.

function of wavelength. This vector is median-smoothed using a smoothing window of half-width `sky-median-hsize` points (*not* pixels). Hence, to obtain a smoothing window of half-width $\sim N$ pixels in the two-dimensional (non-rectified) object spectrum along the wavelength direction, you need to set `sky-median-hsize` to the value $M \cdot N$, where M is the approximate number of sky pixels available at each wavelength. Generally, the default value of `sky-median-hsize` delivers decent sky subtraction results.

`sky-slit-edges-mask` This parameters defines the size of the masked regions at the slit edges (in arcsec). Increasing this number reduces the risk of artefacts but also reduces the number of pixels available to determine the sky background.

`sky-position1/2, sky-hheight1/2` These parameters can be used to define up to two ranges along the slit in the two-dimensional spectrum to determine the sky background[13] `sky-position1/2` define the centers of the sky regions and `sky-hheight1/2` define the half-sizes of the windows around those centers. These ranges are used instead of the default ranges whenever `sky-hheight1` and/or `sky-hheight2` are set to positive values. This flexibility is most useful when multiple objects or objects offset from the center of the slit are observed in a single exposure.

For the current DataSet, try increasing `sky-median-hsize` to `100` and re-running the recipe. This creates a smoother sky background (appropriate for UVB) while improving the extracted spectrum S/N.

Click the `Continue Wkf` button in the interactive window and the workflow will finish. Remember that the final extracted spectra for the DataSet that was processed are saved in the `END_PRODUCTS_DIR/<time_stamp>` directory.

7.3 Reducing Nodding Mode Observations

To prepare for the reduction of nodding data only proceed as follows:

- Click the button.
- In the "Select Datasets" window that pops up, click the button `Deselect all` and then select the DataSet `XSHOO.2010-10-06T08:57:10.811_tpl` by ticking the corresponding box in the first column. This DataSet is a NIR arm observation taken in nodding mode. Click the `Continue` button.

In brief, the recipe `xsh_scired_slit_nod` takes a sequence of science object frames AAA BBB CCC DDD at nodding positions A, B, C, and D, and combines them into the image sequence A B C D (where the combination method is controlled by the parameters under the tab "stack" in the "Recipe Parameters" box). Pairs of combined frames are subtracted to form a sequence A-B, C-D, etc. which are then flat fielded and the orders are rectified. For each subtracted science frame pair, the recipe forms [(A-B) - shifted(B-A)], and if there is more than one subtracted science object frame pair, then the frames of the form [(A-B) - shifted(B-A)] are combined using the mean (`combine-method = MEAN`) or the median (`combine-method = MEDIAN`).

In case of incomplete `nodding` data (only 1 nodding position) the pipeline recipe `xsh_scired_slit_nod` will fail. Such data can be reduced as `stare` data and the detailed description how to do so can be found in Sect. 7.6.2 at the question "**Incomplete nodding data**"

[13] By default the pipeline uses all pixels not covered by the extraction slit or the masking of the slit edges.

7.3.1 Interactive Window

The final combined frame of the form [(A-B) - shifted(B-A)] is displayed in the **bottom plot** of the interactive window in Figure 7.4. The one-dimensional object spectrum extracted from this frame using standard extraction is displayed in the **top plot** of the interactive window, with the flux calibrated version plotted in the **middle plot**. errors and bad pixels are displayed in the same way as described in Sect. 7.2.1.

The plots should be checked carefully for any unexpected features, e.g. much lower or higher noise than expected, gaps, remaining sky lines. In such a case please check Sect. 8.1, p. 62 .

7.3.2 Improving the result

Zooming in on the two-dimensional object spectrum image in the interactive window in the wavelength range 1600-1800 nm (see Figure 7.5) you see that there are many pixels flagged due to cosmic ray hits in the two "negative" spectra while the "positive" spectrum only suffers from a few such pixels. This happens because the cosmic ray contaminated pixels are correctly ignored during the combination of the (A-B) and shifted (B-A) spectrum images, and so the positive spectrum is cleaned of cosmic ray hits.

The interactive window for the `Science Reduction` for data observed in nodding mode provides access to all of the parameters of the `xsh_scired_slit_nod` recipe. The panels **general**, **removecrsingle**, **localize**, and **rectify** contain mostly the same parameters as for `xsh_scired_slit_stare`. The most important ones are discussed in Sect. 7.2.2, p. 40. Rather than describing each of these parameters in detail (see the pipeline manual Modigliani 2012), we provide below some hints on how to improve results, which are ordered by the tabs shown in the right part of the interactive window.

general **Most parameters in this tab are identical to those in the stare interactive window, so we discuss only most important different ones below:**

`correct-sky-by-median` By default this parameters is set to `TRUE` and the recipe calculates and subtracts the median pixel value for each column (wavelength) in the rectified frame from the the column pixel values to correct for any residual sky background changes between the individual nodding positions. It is generally recommended to leave this parameter set to its default value of TRUE unless the user is sure that the changes in the sky background between each pair of nodding positions are negligible.
Try changing `correct-sky-by-median` to `FALSE` and re-running the recipe. The S/N in the wavelength region 1514-1548 nm reported in the title of the top plot will decrease very slightly, indicating that the extra sky correction performed by default by the pipeline recipe `xsh_scired_slit_nod` has little impact for the current DataSet. Put `correct-sky-by-median` back to `TRUE`.

`extract-method` The default value `NOD` integrates the positive flux taking into account the nodding offsets. In the very rare cases where this method may not give good results one can instead set the parameter to `LOCALIZATION`, and set the parameters in the **localize** tab accordingly (see below).

In Figure 7.5, one can see that there is a sharp edge to the object spectrum image at the slit position 0.0 arcsec due to the fact that the upper nodding position loses some of the object flux off the edge of the slit. In general, the recipe `xsh_scired_slit_nod` automatically chooses the best slit integration

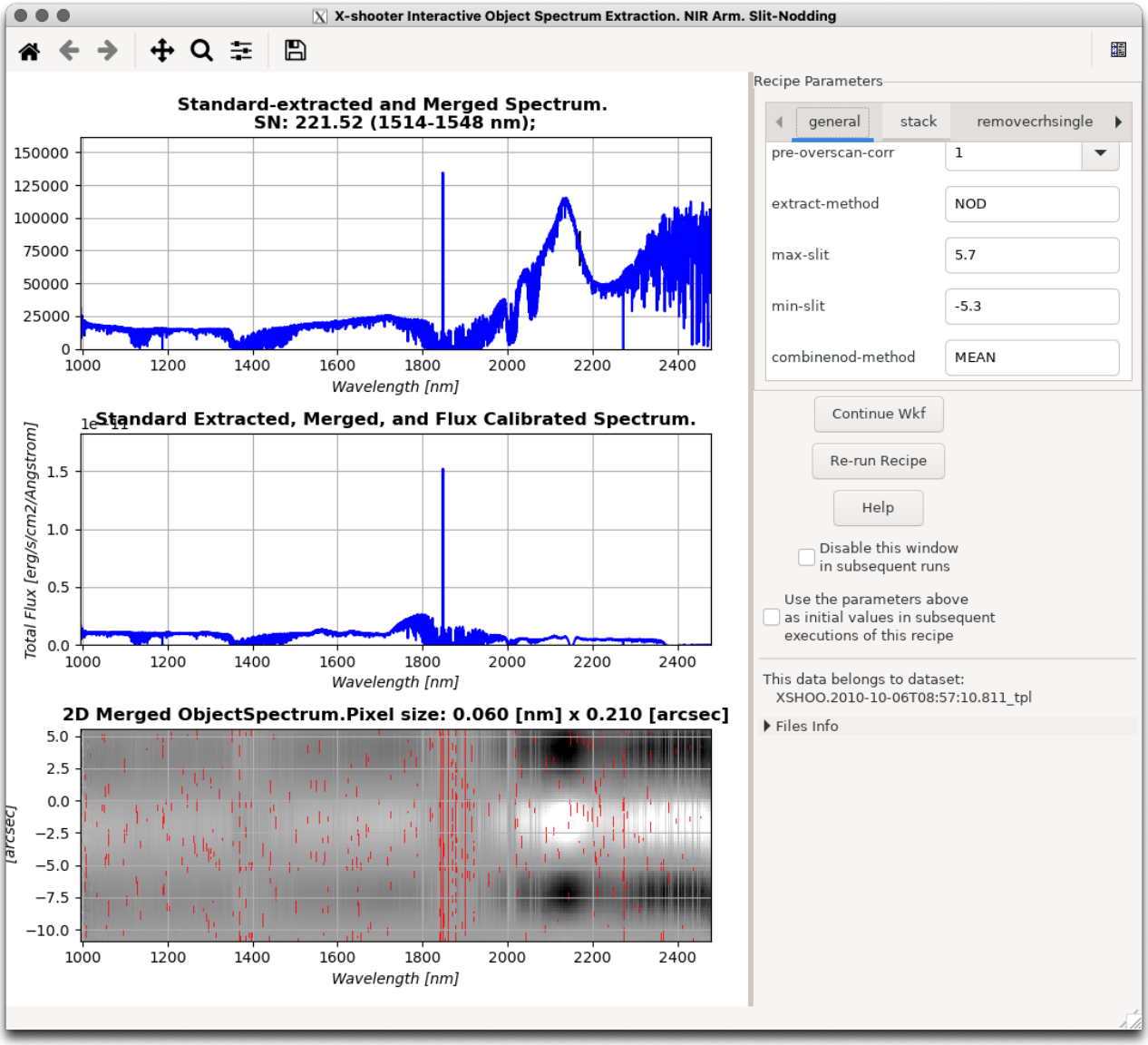


Figure 7.4: *The interactive window for the* `Science Reduction` *actor for the recipe* `xsh_scired_slit_nod`*. The weird shape of the flux-calibrated spectrum in the middle panel is due to problems with offset mode flux standard star data (see Sect. 7.5 for more details)..*

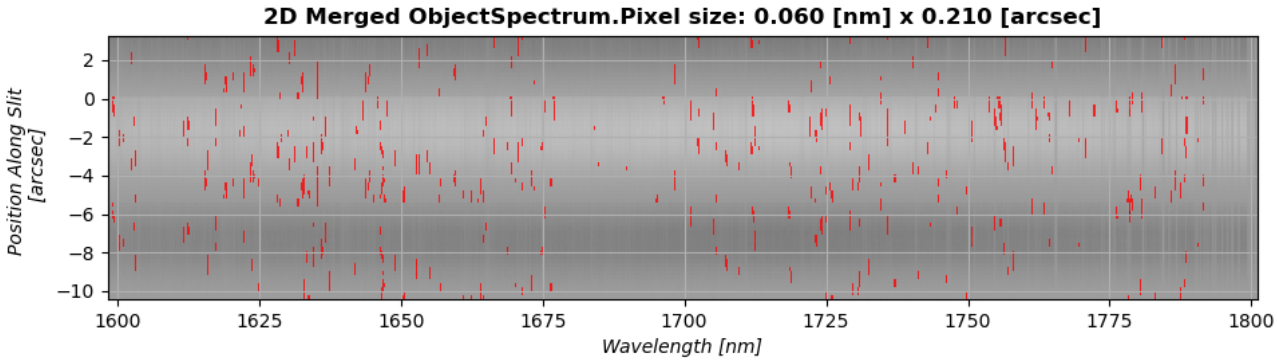


Figure 7.5: *Zoom of the two-dimensional spectrum between 1600-1800 nm in the interactive window for the* `Science Reduction` *actor for the recipe* `xsh_scired_slit_nod`. .

limits for the standard extraction on the two-dimensional object spectrum. However, in some cases it is worth optimising these limits further, which is the case for our current DataSet. To enable control over the standard extraction slit integration limits, set the parameter `extract-method` to `LOCALIZATION`.

localize For a first iteration, the upper slit limit should be set to `0.0` arcsec, while the lower slit limit should be set to `-5.4` arcsec, which defines the exact overlap range between the (A-B) and shifted (B-A) spectra. To set these limits for the recipe, set `localize-slit-position = -2.7` arcsec and `localize-slit-hheight = 2.7` arcsec (these parameters have the same meaning as for the stare recipes), and re-run the recipe. Unfortunately, the reported S/N drops. However, with a slightly shorter slit (`localize-slit-position = -2.5` and `localize-slit-hheight = 2.5`) the S/N improves.

Note that similar to the stare pipeline recipes, we find that the `GAUSSIAN` and `MAXIMUM` methods for the spectrum localisation do not always produce good results. For the current DataSet, these methods work poorly.

The extracted spectrum cannot be improved further by adjusting the recipe parameters in the interactive window, which is due to the fact that the default parameter values are generally pretty robust for the nodding recipes. Click the `Continue Wkf` button in the interactive window and the workflow will finish. Remember that the final extracted spectra for the DataSet that was processed are saved in the `END_PRODUCTS_DIR/<time_stamp>` directory.

7.4 Reducing Offset Mode Observations

Offset mode observations are performed as separate exposures on the science object and on the sky, and the two-dimensional sky spectrum image is subtracted from the two-dimensional object spectrum image in order to remove the sky background.

If instead offset mode observations contain only object data (for instance because the offset mode template was used to map a target or the template was aborted), the DataOrganiser will mark the DataSets as incomplete (grey) and the pipeline will fail. The question **Incomplete offset data** in Sect. 7.6.2 describes how to reduce such data as `stare` data.

A typical problem with offset mode observations is that the sky background varies significantly between the acquisition of the science object and sky background spectra. Offset mode observations with significant sky background are available via the flux standard star observations of the VIS and NIR data sets from 2010. Therefore please double-click on the `Instrument Response` composite actor, and set `EnableInteractivity` to `true`. To disable the interactive window for the `Science Reduction` set its `EnableInteractivity` to `false`.

Click the button.

In the "Select Datasets" window that pops up, click the button `Deselect all` and then select the DataSets `XSHOO.2010-10-06T08:57:10.811_tpl` and `XSHOO.2010-11-10T00:21:10.895` by ticking the corresponding box in the first column. Click the `Continue` button.

After the workflow has run through all of the actors before the `Instrument Response` actor for the current DataSet, the interactive window for the `Instrument Response` actor and the pipeline recipe

`xsh_respon_slit_offset` will appear (see Figure 7.9, p. 7.9).

7.4.1 Interactive Window

The **top plot** in the interactive window show the extracted (using standard extraction) and merged one-dimensional spectrum of the science object (blue) with ±1-sigma uncertainties plotted as the light blue region encompassing the object spectrum (and bounded by the light grey spectra). The **second plot** shows the flux-calibrated spectrum in the same way. The **central plot** is specific to the `Instrument Response` actor and shows the fitted response. The **fourth plot** plot shows the corresponding two-dimensional rectified (and merged) object spectrum as an image with flagged pixels overlaid using a red hue. The **bottom plot** shows the two-dimensional rectified (and merged) sky background spectrum model that was subtracted from the object spectrum before order rectification. Sect. 7.2.1 provides additional details.

The plots should be checked carefully for any unexpected features, e.g. much lower or higher noise than expected, gaps, remaining sky lines. In such a case please check Sect. 8.1, p. 62 .

7.4.2 Improving the Result

For infomation on the most important parameters see Sect. 7.2.2.

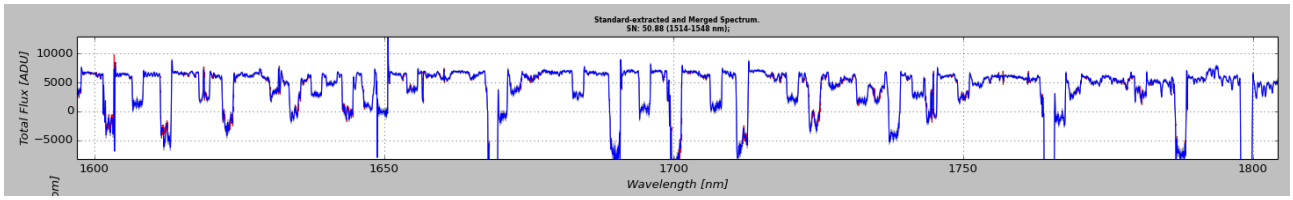


Figure 7.6: *Zoom into the region 1600 nm to 1800 nm in the 1-dimensional merged science spectrum from Fig. 7.9, p. 54.*

The panels **general**, **removecrsingle**, **localize**, and **rectify** contain the same parameters as for `xsh_scired_slit_stare`. The most important ones are discussed in Sect. 7.2.2, p. 40. The interactive window shows clear evidence of over-corrected sky lines in the 1-dimensional extracted spectra. If you zoom in on the wavelength range 1600 nm to 1800 nm and move the spectrum up you can see that there are regions of negative flux at the positions of sky lines (see Fig. 7.6). This can cause trouble when the response curve is fit (zoom in on the corresponding window to have a look). Changing the extraction parameters `localize-slit-position` to `-0.3` and `localize-slit-hheight` to `0.9` and re-running the recipe improves the situation but may cause flux losses. Offset data with such problems can be reduced as stare data by replacing the file `xsh_wkf.oca` in the `DataOrganizer` with the file `xsh_wkf_stare.oca`. Click the `Continue` button.

Once the VIS data have been processed the interactive window pops again, this time showing the results for the VIS flux standard star. Here the sky-subtracted two-dimensional rectified object spectrum still exhibits sky background variations of amplitude ∼1000 ADU as a function of wavelength (move your mouse over the image to inspect the pixel values), indicating again a mismatch in sky background between the `OBJECT` and the `SKY` frame.

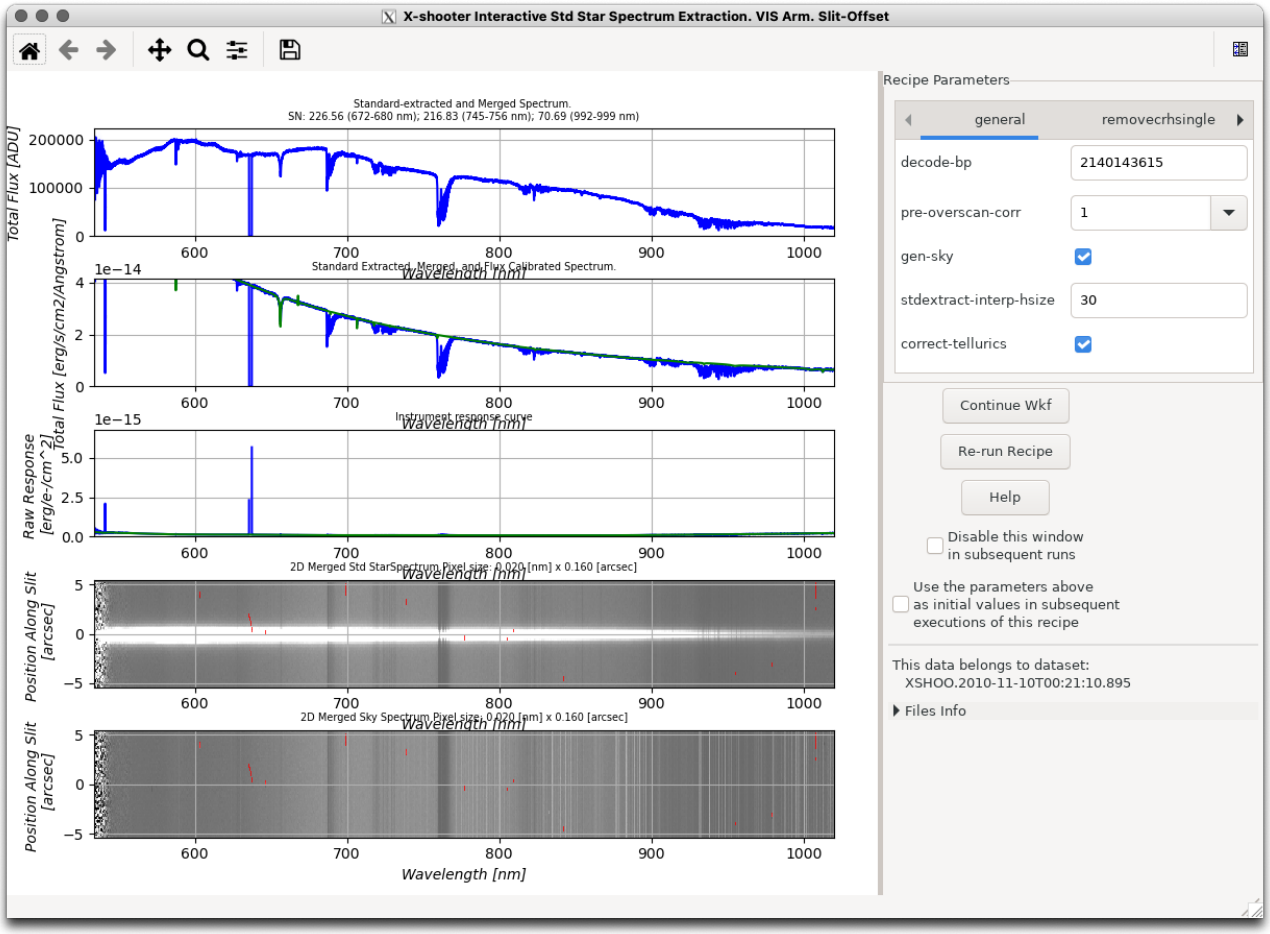


Figure 7.7: *The interactive window for the* `Instrument Response` *actor for the recipe* `xsh_respon_slit_offset`.

Firstly we note that at ∼635-638 nm, the extracted spectrum drops to zero. This happens because the long streak of flagged (bad) pixels in the two-dimensional object spectrum that intersects with the extraction region makes it impossible for the pipeline to construct a local spatial profile for the spectrum given the default value of `stdextract-interp-hsize` = 30. However, if we increase the value of `stdextract-interp-hsize`, we enable the pipeline to construct the local profile over a larger range of wavelengths, and hence properly interpolate over the full set of bad pixels between ∼635-638 nm. Values of `stdextract-interp-hsize` that are greater than or equal to 150 will do the trick (see Figure 7.8).

In all of our examples so far in this tutorial, an improved S/N in the spectrum extraction may be obtained by optimising the slit integration limits for the standard extraction. The current DataSet also benefits from such an optimisation. The standard star spectrum is nicely placed at the centre of the slit in the two-dimensional spectrum image and so we do not need to adjust the parameter `localize-slit-position`. However, integrating the spectrum between ±2 arcsec includes some unnecessary sky background which reduces the S/N obtained. Try setting `localize-slit-hheight = 1.0`, which integrates the spectrum between ±1.0 arcsec, and re-run the recipe. The S/N in the wavelength regions reported in the title of the top plot will increase slightly. You should keep in mind, however, that short slits may cause wavelength dependent slit losses if the spectrum is not perfectly horizontal in the 2-dimensional rectified frame. The results for the final optimised reduction of the standard star for the current DataSet are shown in Figure 7.8.

Click the `Continue Wkf` button in the interactive window and the workflow will move on to the `Science Reduction` actor, which we do not discuss further. The final extracted spectra for the standard star for the current DataSet are available in the `TMP_PRODUCTS_DIR/xsh_respon_slit_offset_1` directory in the subdirectory with the most recent timestamp.

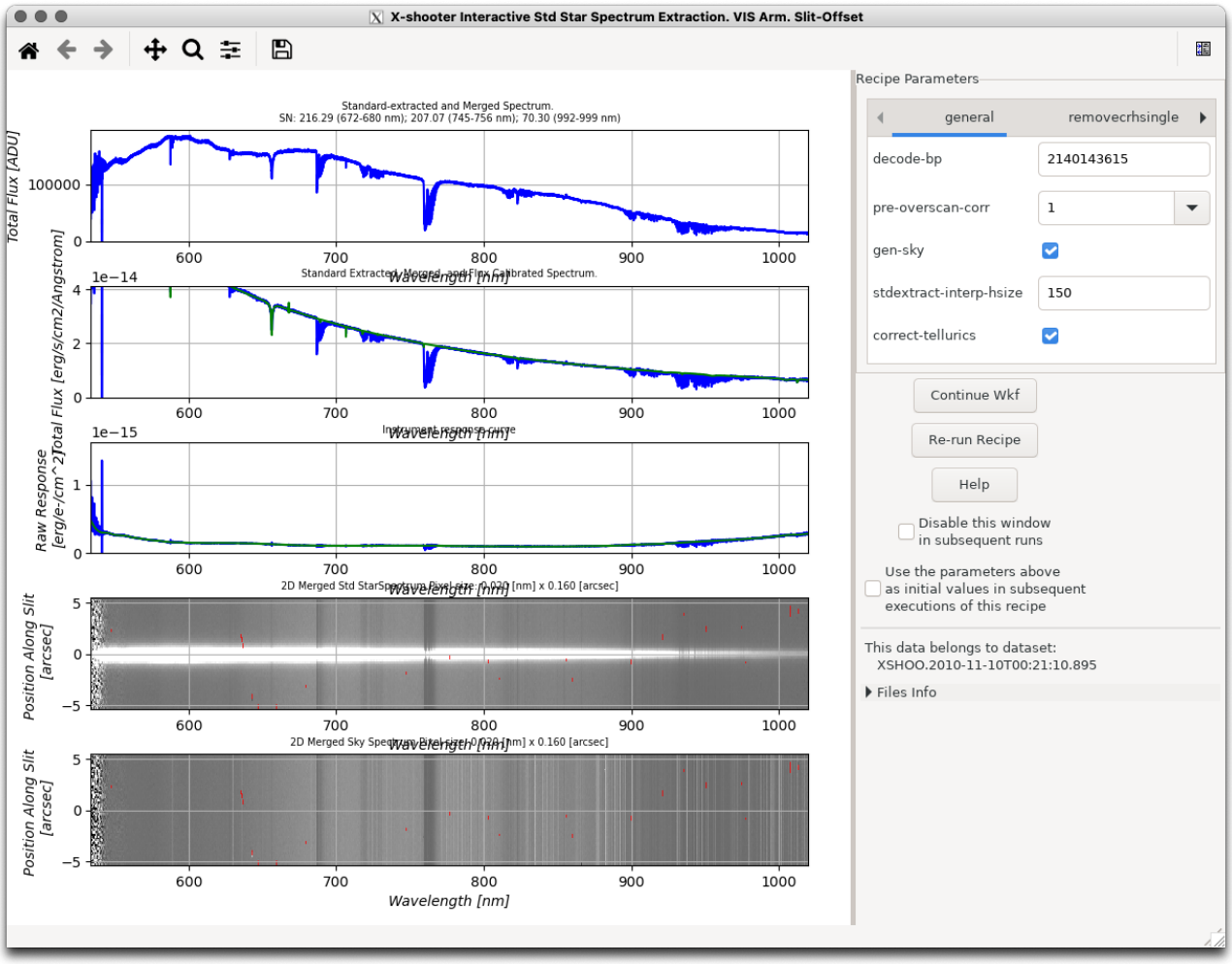


Figure 7.8: *The interactive window for the* `Instrument Response` *actor for the recipe* `xsh_respon_slit_offset` *with* `stdextract-interp-hsize = 150` *and* `localize-slit-hheight = 1.0`.

7.5 Instrument Response

To determine the response function of the X-shooter instrument, flux standard star observations are taken. Until May 2011 these were obtained in offset mode, which often caused problems with the sky subtraction (especially for the NIR arm) because the sky changed too quickly (see Fig. 7.9; Fig. 7.4 shows the effect of such problems on the calibration of the science data). Therefore the flux standard stars have been observed in nodding mode since May 2011 (see Fig. 7.10 for results from such observations). If the sky subtraction for offset mode data causes problems the user should try to use the file `xsh_wkf_stare.oca` for the DataOrganiser, which allows to process all SLIT science data and offset standard star observations as stare and thus determines the sky background from the frame itself instead of from the sky frame. STARE mode is not generally used for flux standard star observations.

The frame processing up to the stage of `FLUX_SLIT_MERGE1D_<arm>` is done exactly as for SCIENCE data of the respective observing mode. The merged 1D spectrum is then corrected for airmass, gain and exposure time. The VIS and NIR spectra are affected by telluric absorption[14]. The pipeline tries to correct for such absorption using a grid of telluric model spectra (`TELL_MOD_CAT_<arm>`[15]). The observed spectrum with the best correction is then used for the response computation. The reference spectrum of the flux standard star is shifted to the the radial velocity of the observed spectrum and then divided by the corrected observed spectrum. The radial velocity correction has an accuracy of 1 pixel (0.02 nm for UVB and VIS and 0.06 nm for NIR), which is sometimes insufficient. Then the ratio of reference vs. observed spectrum show pseudo-P Cygni residuals of varying strength.

In order to avoid such residuals and regions of strong telluric absorption the pipeline uses pre-defined points along the spectrum to fit the response (`RESP_FIT_POINTS_CAT_<arm>`). These points can be changed by editing the multi-extension FITS table (for example with `fv`). They were optimized on nodding data, so for offset mode data some changes may be necessary. The user should be aware that for regions not covered by fitting points (e.g. the wavelength ranges 1300 nm – 1500 nm and 1700 nm – 1980 nm in the NIR arm) the response fit is purely interpolated, which may or may not give a good approximation to the true response.

There are no response-specific recipe parameters, except for `-correct-tellurics`, which is by default set to `TRUE`.

Sometimes instabilities in the flat field lamps cause changes in the spectral energy distribution of the resulting flat fields. In such cases the flux calibration is improved if one uses the same flat field for the flux standard star as is used for the science observations (despite the different slits widths and potentially gains). This is achieved by having `UseResponseFlat` set to `false` in the `FlatStrategy` actor. Fig. 7.11 (left plot) shows an example of such a case.

Unfortunately the transmission/reflection of the dichroic between the UVB and VIS arm sometimes shows slight changes on short timescales. In such cases a good flat field correction may not be possible, but it is worthwhile to try `UseResponseFlat` set to `true` (default: `false`) in the `FlatStrategy` actor. Fig. 7.12 shows a successful example, while Fig. 7.11 (right plot) shows an unsuccessful case (see especially region below 600 nm).

[14] UVB data show only O_3 absorption at the very blue end.

[15] The default telluric catalog for the NIR may provide insufficient correction for data taken during nights with high precipitable water vapour (PWV) content. A catalog extending to higher PWV values is available in the "Data Sets" column for the X-shooter pipeline at the **www.eso.org/pipelines**

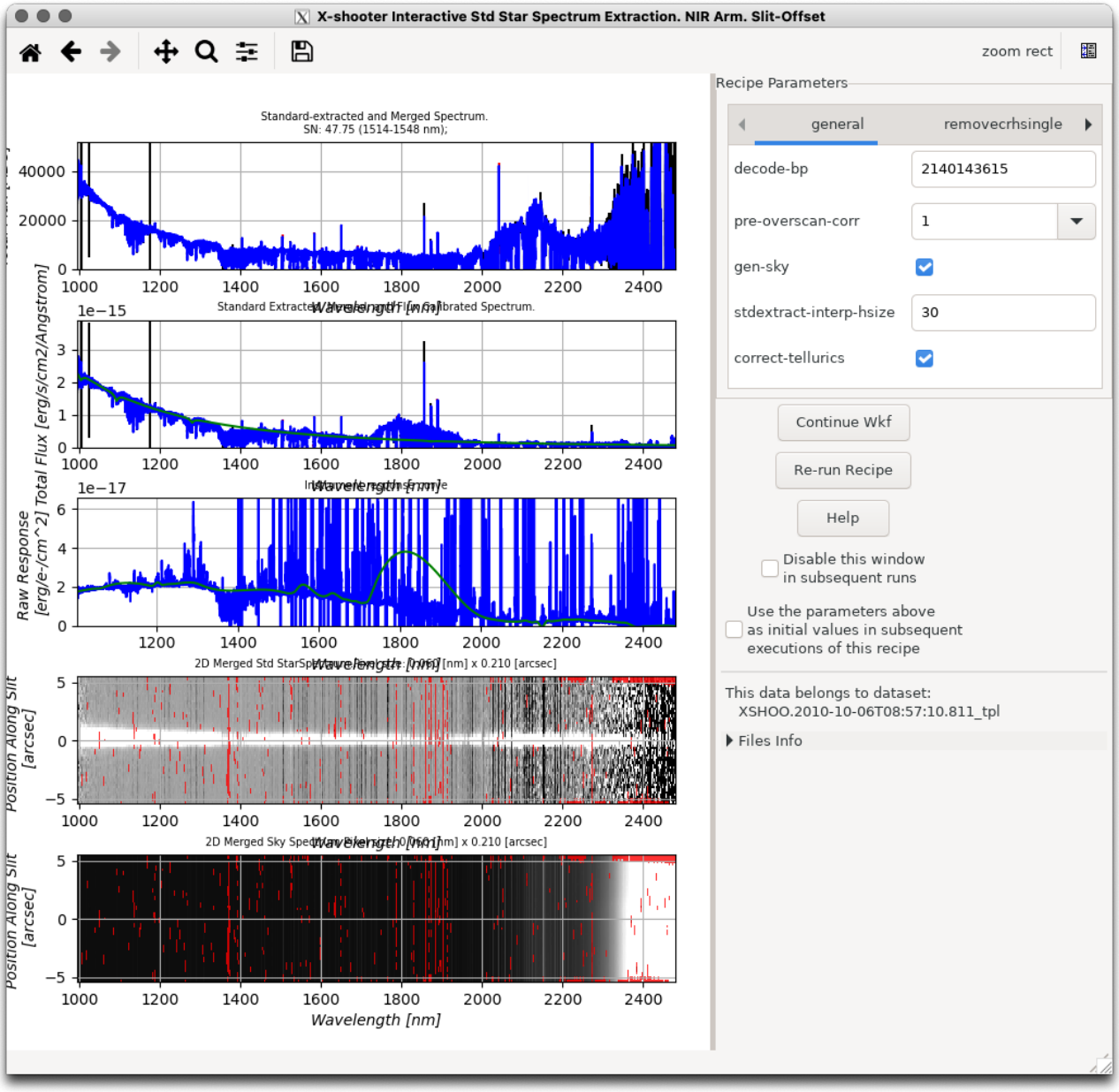


Figure 7.9: *Interactive window of the* `Instrument Response` *actor for the recipe* `xsh_respon_slit_offset`. *The extracted spectrum in the top panel shows clear evidence for sky lines being brighter in the sky frame than in the standard star frame. The weird shape of the response curve is due to these problems with the sky correction of the flux standard star data. Pls note that we have zoomed on the Y axis of the middle panel to make appear this defect.*

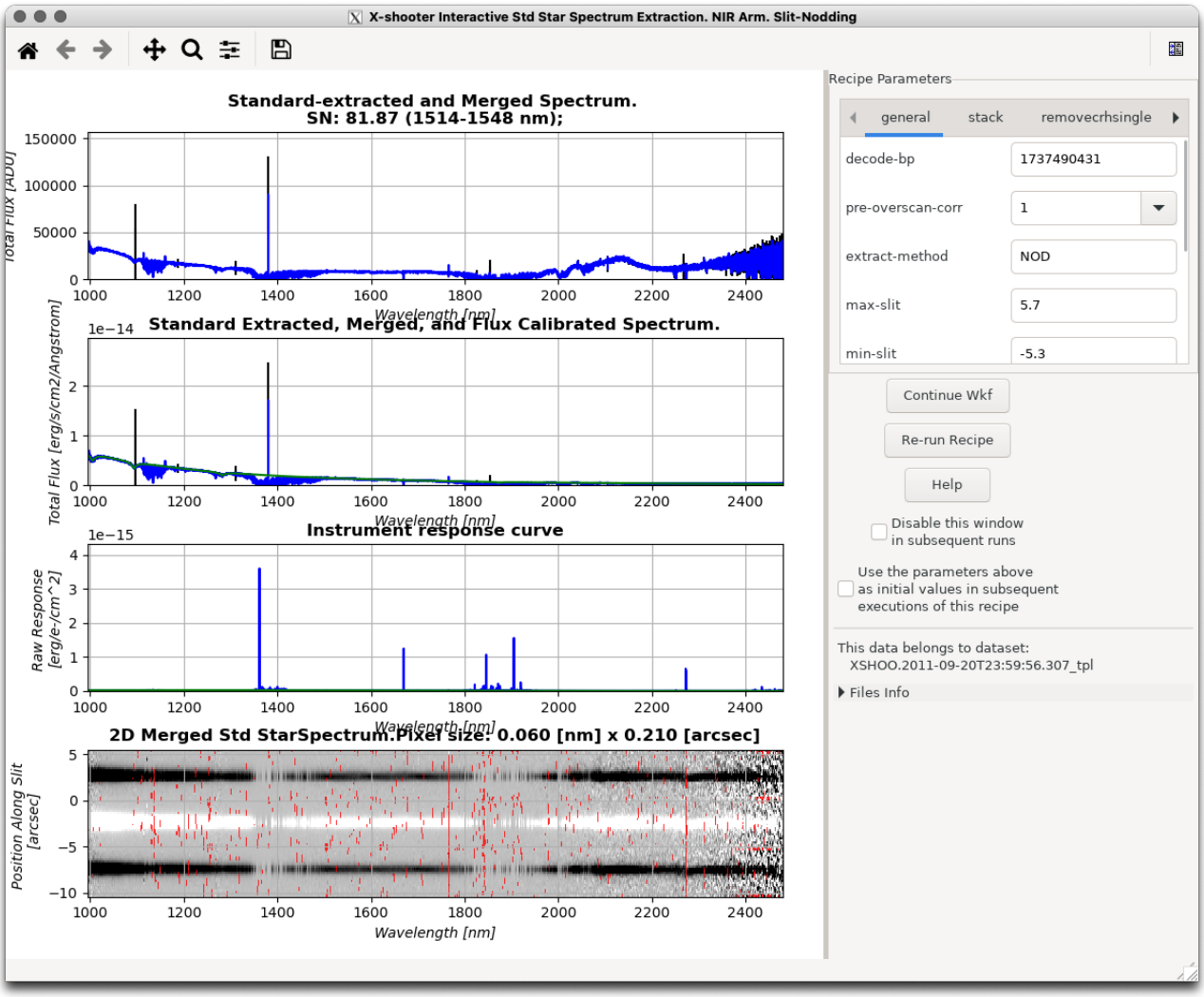


Figure 7.10: *Interactive window of the* `Instrument Response` *actor for the recipe* `xsh_respon_slit_nod`*. The spectrum and the corresponding response show none of the problems seen in Fig. 7.9..*

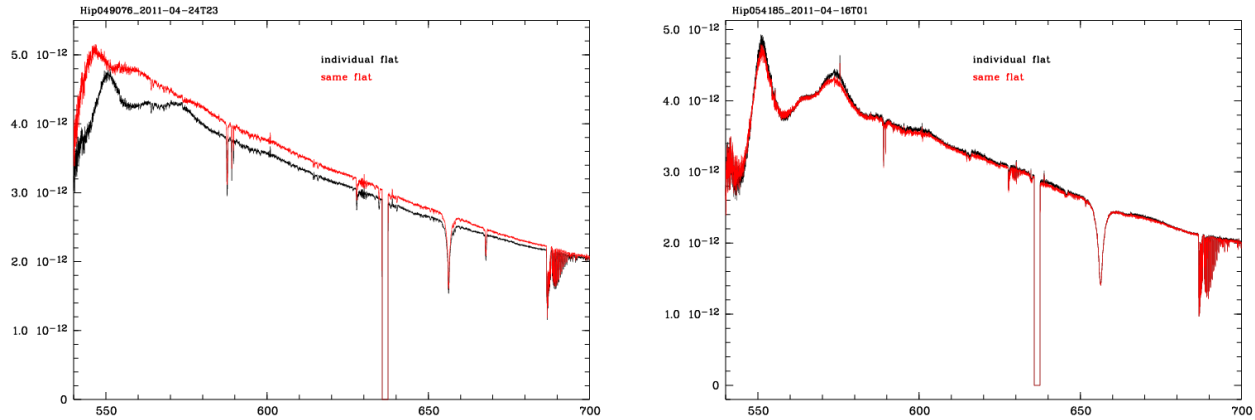


Figure 7.11: **Left**: Here we compare the flux calibrated spectrum of the telluric standard star Hip049076 for both flat field strategies. Using the same flat field for both telluric and flux standard star gives significantly better results. **Right** Same files for Hip054185. Here we see a case where neither flat field strategy provides a good calibration.

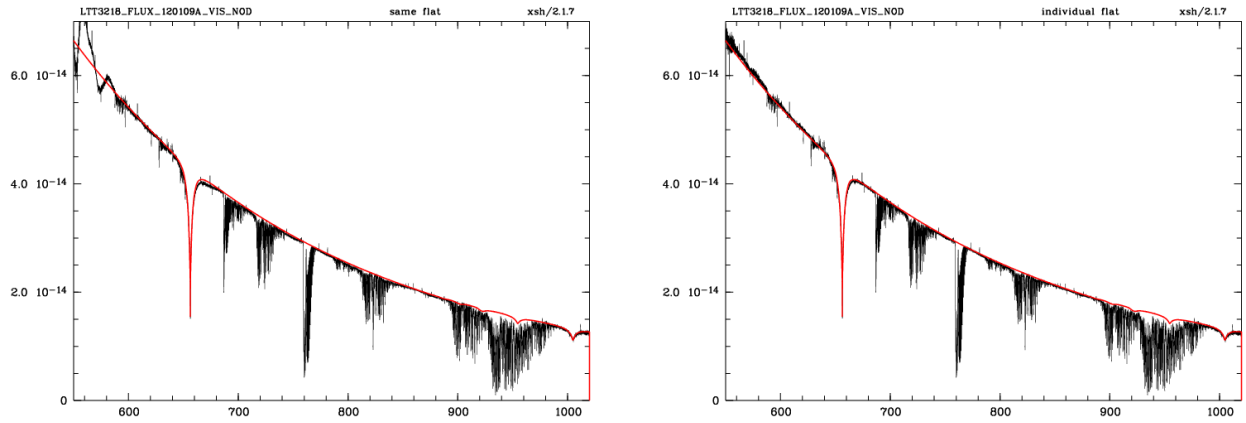


Figure 7.12: Here we compare the flux-calibrated spectra of LTT 3218 (using the response from a different standard star). The black curves shows the flux-calibrated spectrum, while the red curve show the model spectrum (multiplied by a factor to account for the non-photometric conditions under which the standard star for the response was observed). The results shown on the left were obtained with `UseResponseFlat` set to `false`, while for the ones shown on the right `UseResponseFlat` was set to `true`. The oscillations at the blue end of the spectrum on the left are probably caused by shifts in the dichroic curve between the observations.

7.6 Known Problems

Below we provide a list of known problems for X-shooter data, on which we are working.

7.6.1 Workflow Problems

1. **Incorrect parameters in interactive window**
   If the user commits a typo in setting a string or boolean parameter value the recipe fails and the interactive windows will pop-up again asking the user to verify the parameter setting.

7.6.2 Pipeline Problems

1. **Ocasional failures of xsh_mflat recipe** From time to time the recipe xsh_mflat may fail with the following error (shown in the terminal where reflex has been started):

```
[ INFO  ] xsh_mflat:     'PREOVER_COR_XSHOO.2013-10-18T13:23:36.780.fits'
[ INFO  ] xsh_mflat:     'FLAT_PRE_0.fits'
[ INFO  ] xsh_mflat:     'PREOVER_COR_XSHOO.2013-10-18T13:25:25.788.fits'
[ INFO  ] xsh_mflat:     'FLAT_PRE_1.fits'
[ INFO  ] xsh_mflat:     'PREOVER_COR_XSHOO.2013-10-18T13:27:13.487.fits'
[ INFO  ] xsh_mflat:     'FLAT_PRE_2.fits'
[ INFO  ] xsh_mflat:     'PREOVER_COR_XSHOO.2013-10-18T13:29:01.586.fits'
[ INFO  ] xsh_mflat:     'FLAT_PRE_3.fits'
[ INFO  ] xsh_mflat:     'PREOVER_COR_XSHOO.2013-10-18T13:30:49.594.fits'
[ INFO  ] xsh_mflat:     'FLAT_PRE_4.fits'
[ INFO  ] xsh_mflat:     'FLAT_SUB_0_ON_VIS.fits'
[ INFO  ] xsh_mflat:     'FLAT_SUB_1_ON_VIS.fits'
[ INFO  ] xsh_mflat:     'FLAT_SUB_2_ON_VIS.fits'
[ INFO  ] xsh_mflat:     'FLAT_SUB_3_ON_VIS.fits'
[ INFO  ] xsh_mflat:     'FLAT_SUB_4_ON_VIS.fits'
[WARNING] xsh_mflat: Recipe 'xsh_mflat' produced 1 warning  (excluding this one)
[ ERROR ] xsh_mflat: An error occurred, dumping error trace:
[ ERROR ] xsh_mflat:
[ ERROR ] xsh_mflat: Data not found
[ ERROR ] xsh_mflat:  in [6]cpl_image_get_median_window() at cpl_image_stats.c:267
[ ERROR ] xsh_mflat:
[ ERROR ] xsh_mflat: Data not found
[ ERROR ] xsh_mflat:  in [5]xsh_flat_create_normalized_master2() at xsh_create_master.c:2135
[ ERROR ] xsh_mflat:
[ ERROR ] xsh_mflat: Error computing master flat with normalization
[ ERROR ] xsh_mflat:  in [4]xsh_create_master_flat2() at xsh_create_master.c:2329
[ ERROR ] xsh_mflat:
[ ERROR ] xsh_mflat: Data not found
[ ERROR ] xsh_mflat:  in [3]xsh_mflat_combine_flats() at xsh_mflat.c:365
[ ERROR ] xsh_mflat:
[ ERROR ] xsh_mflat: Data not found
[ ERROR ] xsh_mflat:  in [2]xsh_combine_flats_and_detect_edges() at xsh_mflat.c:408
[ ERROR ] xsh_mflat:
[ ERROR ] xsh_mflat: Data not found
[ ERROR ] xsh_mflat:  in [1]xsh_mflat() at xsh_mflat.c:692
[ ERROR ] xsh_mflat:
[ ERROR ] esorex: Execution of recipe 'xsh_mflat' failed, status = 1
```

   This problem is due to too many A/D converter saturated pixels found in the master flat, in some of the orders where the pipeline computes statistics for quality control. To overcome the problem the user may change the value of the parameter **decode-bp** from its default, 2140143615, to 2140139519 (obtained

by subtracting the code 4096, corresponding to A/D converter saturation, from 2140143615). In case of reflex workflow execution this can be done by mouse-select on "Open Actor" on the Master Flat Creation reflex actor, then mouse-select on "Configure Actor" on the xsh_mflat recipe actor, and finally changing the default 2140143615 to 2140139519.

2. **Sky subtraction for NIR stare data**
   The K-band region of the NIR spectra often causes problems for the sky correction. Here using `sky-method = BSPLINE` (or `BSPLINE2`) may be better than `MEDIAN`, but requires some fiddling with the some input parameters (see pipeline manual for more details).

3. **Flat-fielding flux standard stars**
   Changes in the spectral energy distribution between the observations of the flat field for the flux standard star and that for the science data can cause problems in flux calibrations. This can be avoided by flat-fielding the flux standard star spectrum with the same flat field as is used for the science data. This can be done setting `UseResponseFlat` to `false` in the `Flat Strategy` actor.

4. **Incomplete `nodding` data**
   A `nodding` DataSet requires to have at least two nodding positions to be processed successfully. If a template is aborted the user may receive incomplete data, i.e. with only one nodding position. In this case processing will fail. The `DataOrganiser` in the current release will not mark such DataSets as incomplete and the recipe `xsh_scired_slit_nod` will fail. In this case please use for the `DataOrganiser` the OCA file `xsh_wkf_stare.oca` instead of the default one `xsh_wkf.oca`. This is done by right-clicking on the `DataOrganiser`, selecting `Configure Actor`, replacing the name of the OCA file in the first line of the pop-up window and clicking on `Commit`. This change will result in the pipeline processing the nodding data individually with the `xsh_scired_slit_stare` recipe.

5. **Incomplete `offset` data**
   For data taken in offset mode the pipeline expects to find both object and sky data within a given DataSet. If instead your offset mode observations contain only object data (for instance because the offset mode template was used to map a target or the template was aborted), the `DataOrganiser` will mark the DataSets as incomplete (grey) and the pipeline will fail (sky data alone do not create a DataSet with the default OCA rules). In this case please use for the `DataOrganiser` the OCA file `xsh_wkf_stare.oca` instead of the default one `xsh_wkf.oca`. This is done by right-clicking on the `DataOrganiser`, selecting `Configure Actor`, replacing the name of the OCA file in the first line of the pop-up window and clicking on `Commit`. This change will result in the pipeline processing the offset mode data individually with the `xsh_scired_slit_stare` recipe. When reducing offset mode data in stare mode the interactive window of the ScienceReduction shows for the sky frames the interactive window from the previous InstrumentResponse actor.

6. **Processing offset or nodding mode data as stare**
   Some user may prefer to reduce offset mode or nodding mode data as they would be acquired in stare mode. As in the previous cases to do this the user has to use the `xsh_wkf_stare.oca` OCA rule file. When reducing offset mode data in stare mode the interactive window of the ScienceReduction shows for the sky frames the interactive window from the previous InstrumentResponse actor.

7. **Processing of IFU data**
   With this release we provide also a basic workflow to help with the IFU data organisation and execution of the pipeline. The workflow was generated without a review of the quality of the IFU science products.

In particular we are aware that current IFU data requction may provide limited accuracy (up to 1 pix level) in the alignment of the IFU slices (as may be noted in the corresponding IFU object traces quality plots. In case of reduction of low signal to noise ratio (or not point like) science objects the object traces may appear very scattered, this reflecting a not accurate fit of the object position.

8. **Interactive GUI title overlapping plot/image captions**
   The interactive workflow for the response recipe contains a lot of information in a relatively limited size. In some cases this may result in some plot/image title overlapping with the one of the next image/plot caption. This problem may be solved by increasing the size of the interactive window, e.g. to full screen.

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 (2012)). It is this purpose that is used by the workflow to send the correct set of bias frames to the recipes for flat frame combination and science frame reduction, which may or may not be the same set of bias frames in each case.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

- **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`

8.1 X-shooter specific questions

- **Why are telluric standard stars classified on their own?**
  Since there is no pipeline recipe to use the telluric standard stars to correct the telluric absorption in the science spectra, the telluric standard stars are not used for science processing and would therefore not be processed. In order to provide the user with processed telluric standard star spectra (and thus to allow the user to perform the telluric correction outside `Reflex`) they are classified and processed on their own.

- **All my STARE data are processed as individual files - how can I have them stacked by the pipeline?**

  Please use for the `DataOrganiser` the OCA file `xsh_wkf_starestack.oca` instead of the default one `xsh_wkf.oca`. This is done by right-clicking on the `DataOrganiser`, selecting `Configure Actor`, replacing the name of the OCA file in the first line of the pop-up window and clicking on `Commit`. This change will result in the pipeline combining STARE data taken within one template.

- **My OFFSET data are not processed.**

  Offset mode data containing only object or only sky data cannot be processed by the `xsh_scired_slit_offset` recipe. The DataOrganiser in the current release will mark such DataSets as incomplete. If you attempt to process such datasets with `xsh_scired_slit_offset` the workflow will ignore them. Instead please use for the DataOrganiser the OCA file `xsh_wkf_stare.oca` instead of the default one `xsh_wkf.oca`. This is done by right-clicking on the DataOrganiser, selecting

Configure Actor, replacing the name of the OCA file in the first line of the pop-up window and clicking on Commit. This change will result in the pipeline processing the offset mode data individually with the `xsh_scired_slit_stare` recipe.

- **What is the meaning and use of the parameter UseResponseFlat in the FlatStrategy actor?**

  The transmission/reflection behaviour of the dichroics, that separate the wavelength regions for the three X-shooter arms, sometimes varies on short time scales. In consequence the flat fields for the standard star and for the science target may differ in the overlap regions. Therefore the workflow is set to use by default the flat field taken for the science data also for the standard star (UseResponseFlat=false). However, there may also be cases, where this setting results in spurious features in the flux calibrated data (See Sect. 7.5, p. 53 for more details). In such cases one should also check the results of the flux calibration using the standard star flat field to calibrate the standard star observation (UseResponseFlat=true).

- **What is that strong absorption feature in my UVB data at about 570 nm?**
  This feature visible in Fig. 8.1 is seen in both daytime (e.g. flat field) and nighttime (e.g. standard star and science) data, but is variable and can therefore not be calibrated. It is most likely due to the dichroic, which is not specified for this wavelength range in the UVB arm.

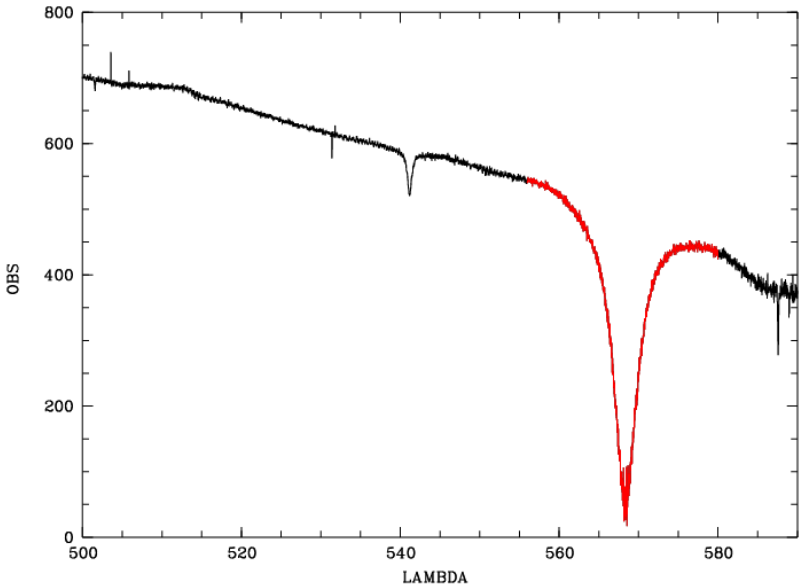


Figure 8.1: *Here you see a spectrum of the flux standard star Feige 110 which clearly show the strong absorption feature from 556 nm to 580 nm..*

- **My MERGE1D spectrum shows a gap although the MERGE2D spectrum does not.**

  During the 1-dimensional extraction the local spatial profile (along the cross-order direction) of the spectrum is determined by collapsing the 2-dimensional spectrum along the dispersion axis. The **stdextract-interp-hsize**

parameter defines the half size of the region across which the spectrum is collapsed. This parameter affects flagged pixels interpolation. In case of spectrum gaps one needs to increase the default value to an optimal value of: (`size_of_gap`[nm]/(2*`size_of_pixel` [nm])+1), where `size_of_pixel` is given by the value of **rectify-bin-lambda**

(by default 0, corresponding to 0.02nm for UVB and VIS data and to 0.06nm for NIR data, recorded in the CDELT1 FITS keyword of the products).

- **Negative pixel values in the NIR 2D spectrum image are not being ignored by the standard extraction procedure. How can I ignore them?**
  The X-shooter pipeline flags but does not ignore those pixels that are negative in its algorithms including standard extraction. This is because a reset anomaly can cause all pixels in a raw image to be negative even though the data is perfectly valid. If a reset anomaly has not occurred in your data, and negative pixels are causing a problem, then simply add 2097152 to the value of the parameter `decode-bp` to start treating negative pixels in the raw data as bad pixels by ignoring them.

- **The MERGE2D spectrum shows lots of bad pixels. Why?**

  The X-shooter pipeline flags bad pixels that are detected during the processing and records them in the QUAL extension. It interpolates across those bad pixels selected by the decode-bp parameter only in the extraction to the MERGE1D spectrum. There is no interpolation across bad pixels in the MERGE2D spectrum.

- **My extracted 1-dimensional `NIR` spectra show many spikes although the raw signal looks very good. What happened?**

  For data observed with very good seeing, steep slopes in the spatial profile of the spectra result in many spurious cosmic ray detections. This happens most often for `NIR` data, because they have a coarser spatial sampling (0.21 arcsec/pixel vs. 0.16 arcsec for `UVB`/`VIS` data), but due to their longer wavelengths better seeing than the bluer wavelength ranges.

  The spurious cosmics may in turn cause spikes as can be seen in the plot (black spectrum). If you process the data with `decode-bp = decode-bp - 32` (effectively treating the cosmic ray detections as good pixels) the spikes caused by spurious comics should vanish (red spectrum, see Fig. 8.2).

- **My science spectra still show telluric absorption - why?**

  The pipeline performs an automatic correction of telluric absorption only for the flux standard stars, as the object's absorption/emission features have to be known for this procedure. For science data either the telluric standard stars or the interactive Molecfit tool may be used (see
  http://www.eso.org/sci/software/pipelines/skytools/molecfit).

- **I get inconsistent results from different absorption/emission lines in my NIR spectrum - why?**

  The NIR detector is read in a way that flux is extrapolated once it exceeds 42000 ADU. This is no problem as long as observing conditions stay constant. If for instance the transparency varies a linear extrapolation will no longer give correct results. You can find pixels which have extrapolated flux by looking for the value 1048576 or larger in the QUAL extension. Adding this number to the decode-bp parameter will cause these pixels to be treated as bad. For more details see
  http://www.eso.org/sci/facilities/paranal/instruments/xshooter/doc/reportNDreadoutpublic.pdf

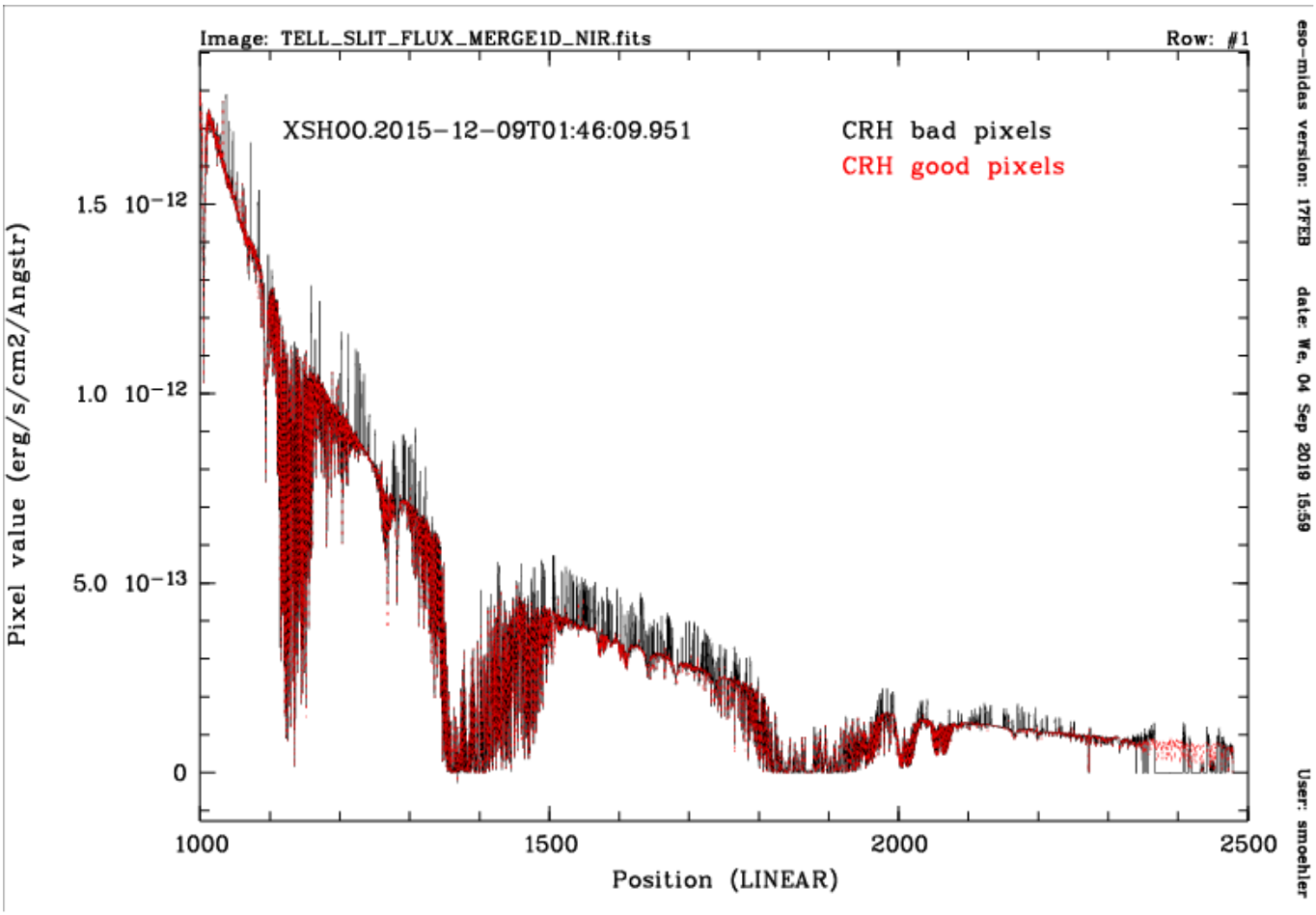


Figure 8.2: *NIR standard star spectrum obtained in very good seeing processed with pipeline default parameters (black) and setting the decode-bp parameter to* `decode-bp = decode-bp - 32` *to accept detected CRHs as good pixels (red).*

- **Some of the sky lines in my STARE data were not corrected at all - why?**

  Bright sky lines have very steep slopes and may be flagged as cosmic ray hits. In that case they will be ignored during the sky fitting. This problem can be solved by switching off the cosmic ray detection, i.e. setting `removecrhsingle-niter` to `0`. Due to the brighter sky lines at longer wavelengths the effect is mostly found for NIR data.

- **Some of my extracted NIR STARE spectra show a jump at about 2.3μm - why and how can I correct it?**

  The fit of the sky background used by the X-shooter pipeline relies on the assumption that there is no gradient in flux along the slit. For the reddest part of the NIR spectra this assumption is not fulfilled and therefore the sky subtraction does not work well. The jump you see is caused by a bad sky correction and shows up most prominently for sky-dominated observations, i.e. sources with little flux compared to the flux of the sky background in that wavelength range.

If this wavelength region is important for your analysis you need to process the data without sky subtraction and then use independent software tools to fit and subtract the sky from the rectified, wavelength calibrated 2-dimensional frame (MERGE2D).

- **Why do my flux-calibrated data not agree with independent photometric measurements?**

  In order to have a true absolute flux calibration several requirements need to be fulfilled:

  1. The fraction of the total flux of an object that is contained in the slit depends on the shape of the object, the width and orientation of the slit, and the seeing. Absolute flux calibration requires that all the flux of both the object and the standard star has been collected.
  2. The flux that arrives at the telescope depends on the transparency of the sky. Absolute flux calibration requires the same transparency for the observations of the target and the standard star.

  A change between the flux standard observation and the science object observations of any of the parameters mentioned above will change the flux scale in the final spectrum. To compare the flux calibrated spectrum to other measurements, differences in slit losses and atmospheric conditions have to be taken into account.

  With respect to the faction of flux contained within the slit one should keep in mind that the flux standard stars are observed with a 5" wide slit, while science data are typically observed with slit widths of 0.8" to 1". For a point source a slit width of 0.8"/1.0" used with a seeing of 0.8" means that some 33%/24%, respectively, of the target flux are lost (see also Fig. 8.3). This results in a too low flux for the flux calibrated spectrum of the target.

  If the standard star or the target or both are observed under non-photometric conditions (e.g. CLR or THN) their observed flux will be lower than it should be. If the standard star is observed under photometric conditions but the science target is not the flux in the flux calibrated target spectrum will be too low. The opposite happens if the target is observed under photometric conditions but the standard star is not. CLR/THN conditions allow for transparency variations of 10%/20%, respectively. Master response curves are derived from the upper envelope of individual response curves and should thus correspond to flux standard stars observed under photometric conditions.

- **The IFU slice object traces show a not negligible shift. How can I improve the corresponding object traces alignment?**

  From time to time the IFU slice traces do not overlap. In such a case the alignment of the IFU slices may be improved by providing (in the sof) the optional static table tagged as `IFU_CFG_TAB_ARM` (ARM=UVB, VIS, NIR). If the offset between slices is constant with wavelength the corresponding ARM coefficients of the `S_LOW_OFF` or `S_UPP_OFF` need to be corrected in this table. The correction is defined by the shift one wants to correct multiplied by the size of the spatial bin one has chosen to reduce the data (by default 0.16/0.16/0.21 arcsec/pixel for UVB/VIS/NIR data, respectively). If the offset between the slices varies with wavelength also the other coefficients provided in that table need to be changed. The coefficients controlling the slope are `W_UPP_COEF1` and `W_LOW_COEF1`. The coefficients controlling the curvature are `W_UPP_COEF2` and `W_LOW_COEF2`. If the user has to change these parameters we recommend to first optimise the offset, then the slopes (judging if it needs to be increased or decreased and, in order to make small variations, starting from the second significant digit of the default value) and similarly eventually changing the curvature coefficients.

This correction can be determined by aligning the traces of the telluric standard observation usually taken in the same night. The table optimized this way should then be used as input of the (usually low signal to noise) object data reduction.

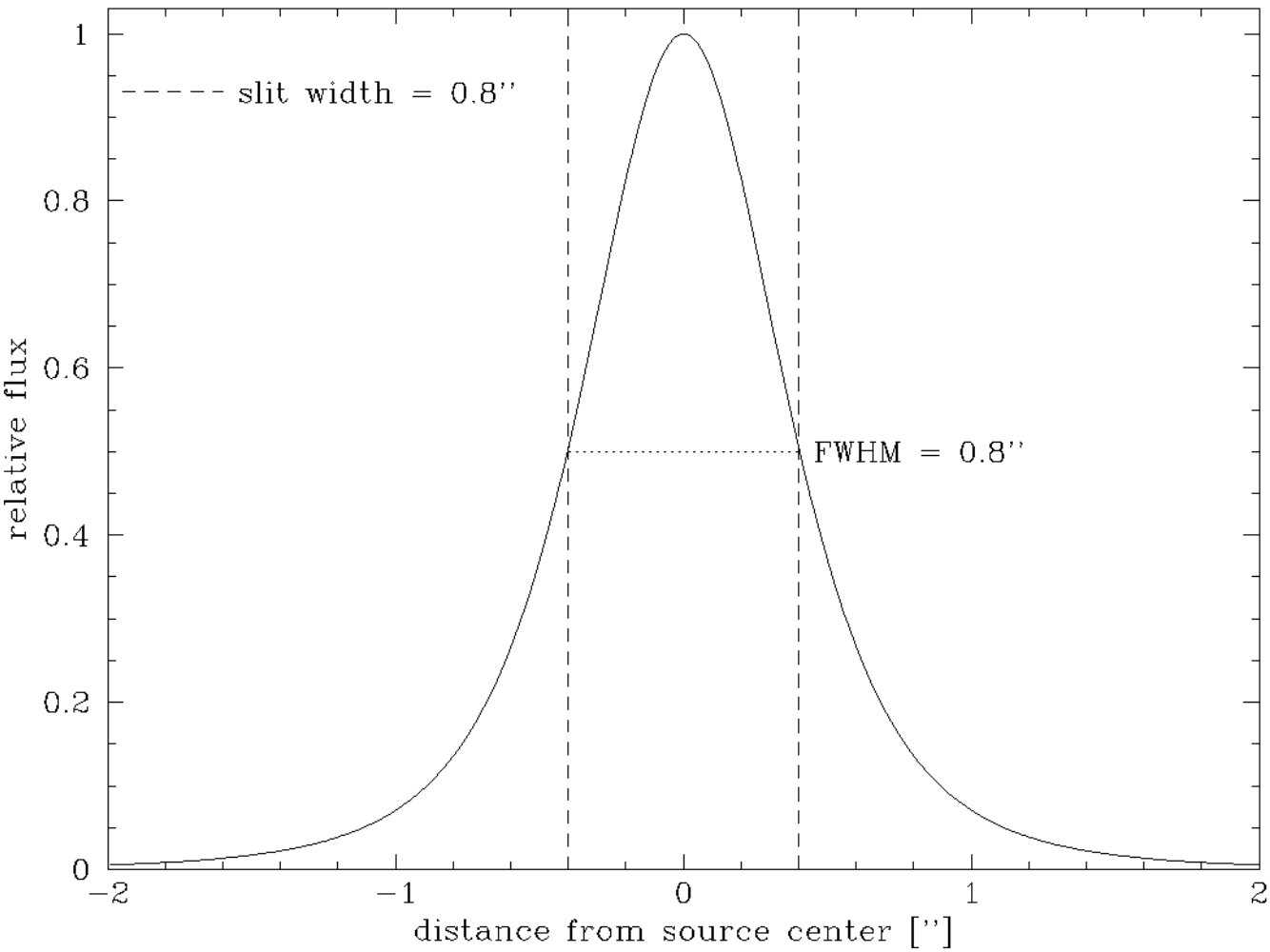


Figure 8.3: *Here a Moffat profile with a FWHM of 0.8" is shown. The dashed lines mark the limits of a 0.8" slit, while the dotted lines mark the FWHM ..*

9 Troubleshooting

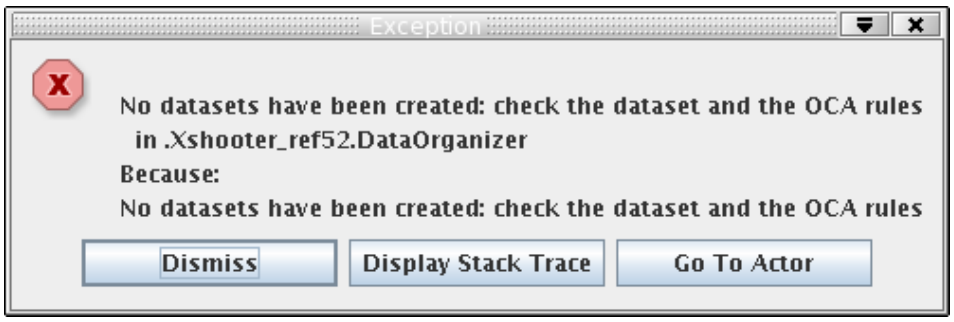


Figure 9.1: *TheDataOrganizer interactive window reports an error ":No DataSets have been created, check the data set and the OCA rules.".*

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

   (a) **it crashes**

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

   (b) **it fails with error message ":No DataSets have been created, check the data set and the OCA rules."(see Figure 9.1.)**

   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?**

   This indicates that the DataSets are incomplete and will in general fail during processing. Moving the mouse over the grey datasets provides information about the missing data (see Fig. 9.2). There are two cases:

   (a) Missing calibrations. Missing raw calibration should already be indicated as such by calSelector (see the `.xml` file). This is unusual but may happen if an OB was classified as "must repeat". All static calibrations should be found by `reflex` in `<install_directory>/calib/<pipeline_version>/cal`.

   (b) Missing science data. In most cases this is due to missing `SKY` frames for `OFFSET` observations and the pipeline will fail. Such data can be processed like `STARE data`, if you use for the `DataOrganiser` the OCA file `xsh_wkf_stare.oca` instead of the default one `xsh_wkf.oca`. This is done by right-clicking on the `DataOrganiser`, selecting `Configure Actor`, replacing the name of the OCA file in the first line of the pop-up window and clicking on `Commit`. This change will result in the pipeline processing the offset mode data individually with the `xsh_scired_slit_stare` recipe.

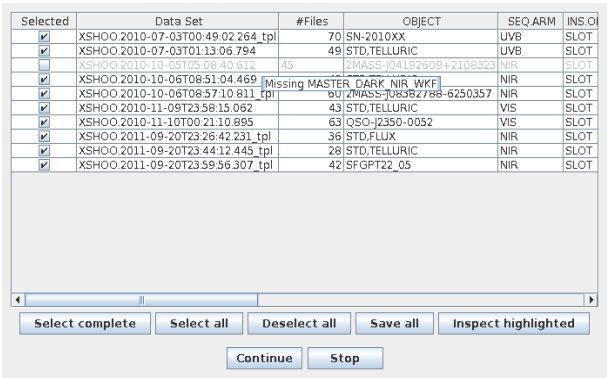


Figure 9.2: *This shows how information about missing files in an incomplete DataSet is displayed when moving the mouse over it..*

3. **the SpectralFormat actor fails**. This may occur occasionally on not well exposed arc lamp frames if not enough lines can be detected. In this case we recommend the user to change the recipe configuration parameter **detectarclines-min-sn** and set it to a small value, for example 2 or 3.

A From Spectral Format Check Through to Wavelength Calibration

Carry out the first four steps described in the Quick Start Section 3.

In the X-shooter workflow, the interactive actors are marked by an orange rectangle encompassing the actor. For `Spectral Format`, `Master Flat Creation`, and `2D Geometry` you will need to enable their interactive mode. Do this by double-clicking on the composite actor, setting the "EnableInteractivity" parameter to `true`, and clicking `Commit` to save the changes to the workflow. Then click the button.

In the "Select Datasets" window that pops up, click the button `Deselect all` and then select the DataSet XSHOO.2010-07-03T01:13:06.794 by ticking the corresponding box in the first column. This DataSet is a UVB arm observation taken in stare mode. Click the `Continue` button.

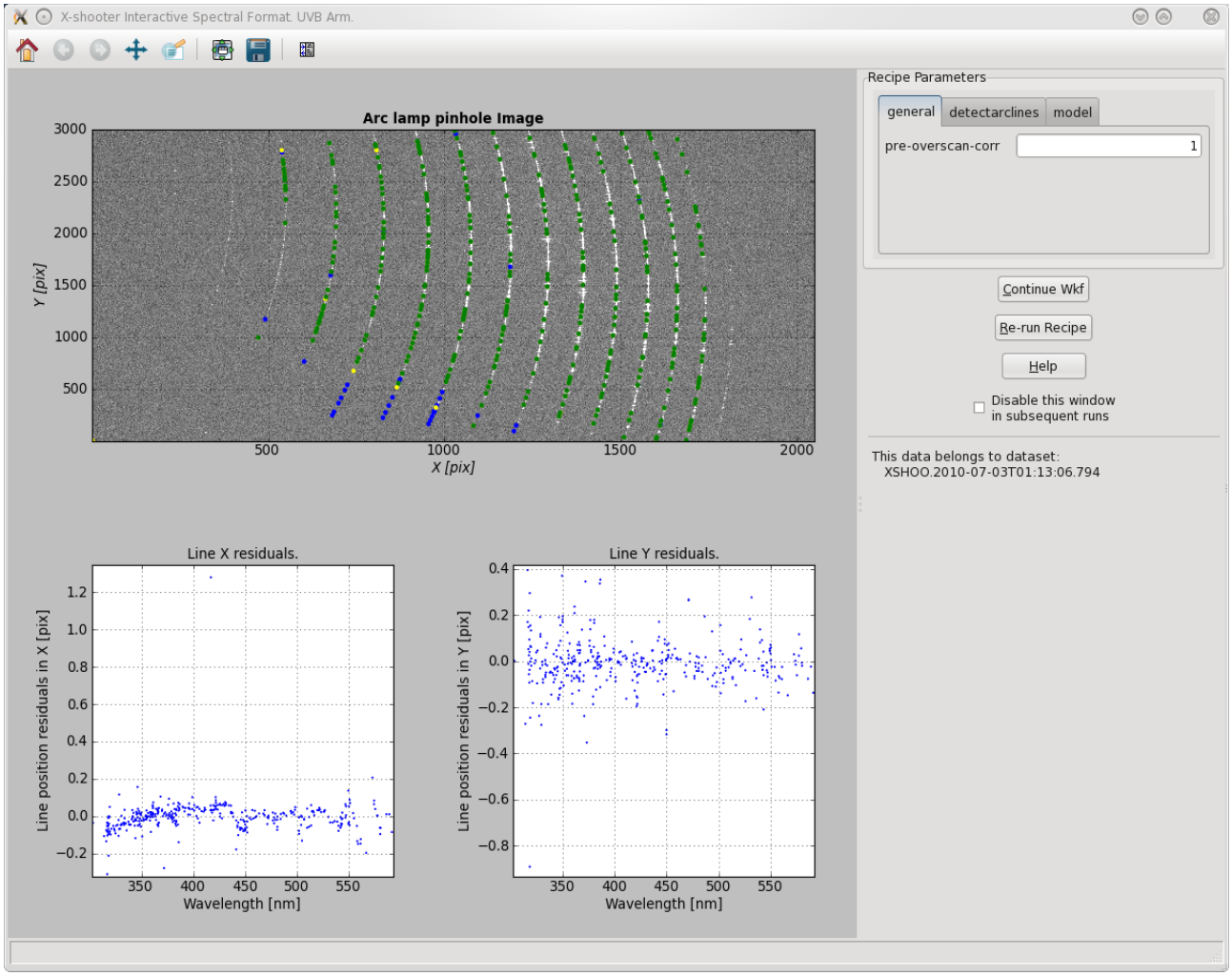


Figure A.1: *The interactive window for the* `Spectral Format` *actor for the recipe* `xsh_predict`.

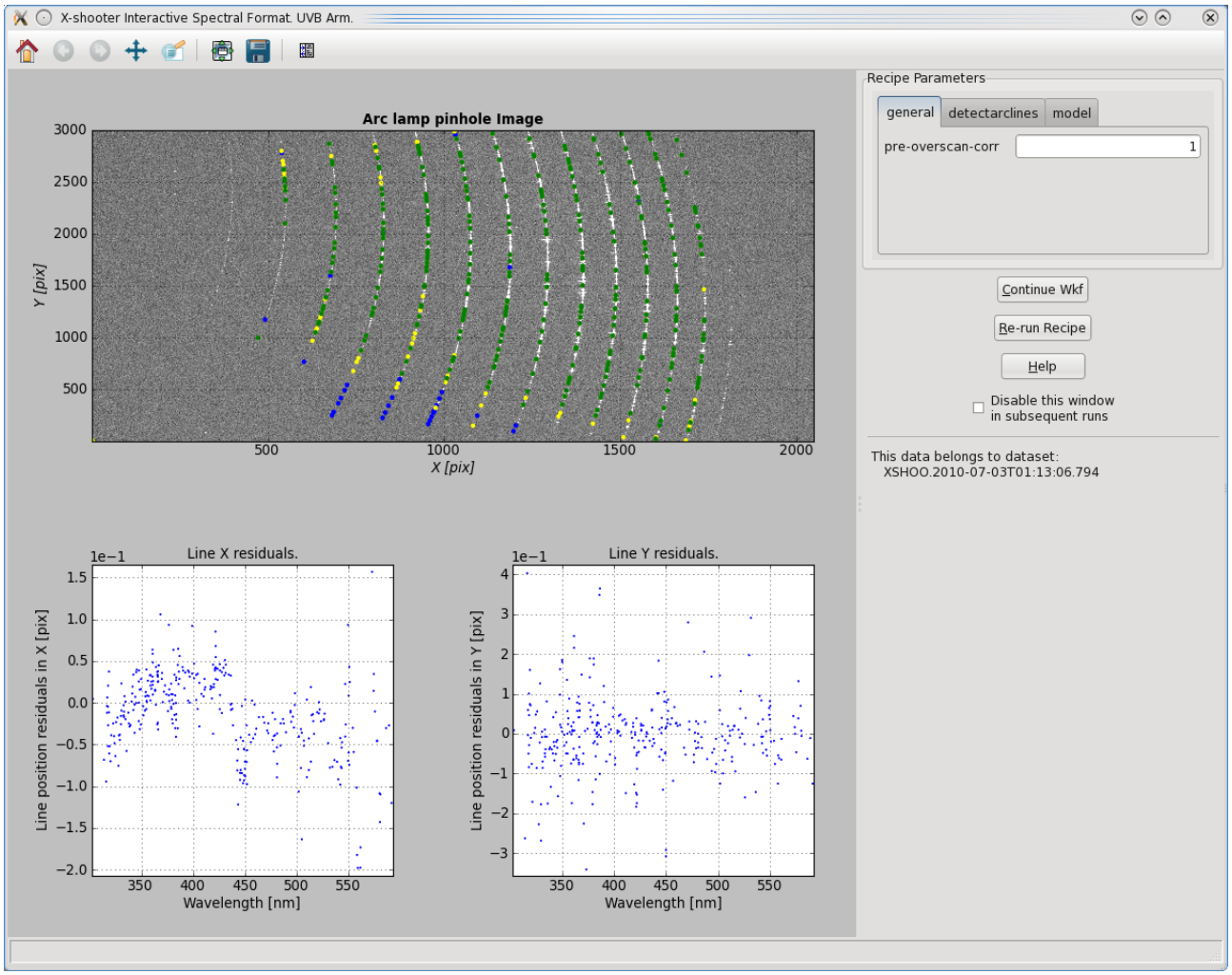


Figure A.2: *The interactive window for the* `Spectral Format` *actor for the recipe* `xsh_predict` *with* `detectarclines-min-sn = 10`.

A.1 Spectral Format

The `Spectral Format` actor first executes the X-shooter pipeline recipe `xsh_predict`, taking as input a single-pinhole arc lamp frame. The purpose of the recipe is to compute initial parameter values for the physical model wavelength solution.

A.1.1 Interactive Window

During the execution of the `Spectral Format` actor, an interactive window will appear as shown in Figure A.1.

The **top** plot shows the 1-pinhole arc lamp frame with the identified lines marked in colour. The two **bottom**

plots show residuals of the line positions vs. wavelength.

If the physical model prediction is good, then the **bottom** plots will show only slight trends (with peak-to-peak amplitudes of 0.2-0.4 pix) with a few scattered outliers. If the physical model prediction is poor, then these plots will show only scattered points with no trends, although this happens very rarely. Further attempts to remove any trends in these residual plots are not necessary at this early stage in the data reduction cascade, since the physical model fit is an initial guess for use later in the cascade.

A.1.2 Improving the Results

For illustrative purposes you can try changing the parameters for this recipe in the interactive window and re-running the recipe.

detectarclines

`detectarclines-min-sn` This parameter controls the minimum acceptable S/N for an arc line to be used in the physical model fit. Changing it results in a change of the number of arc lines used in the fit.

For a test change `detectarclines-min-sn` to 10.0. Then click the `Re-run recipe` button. The new interactive window shows that the number of arc lines used in the fit has decreased (less green points in the format-check arc-lamp image). The outlier residuals seen in Figure A.1 have disappeared because the corresponding arc lines have S/N<10. However, the scatter in the line x residuals has increased markedly while the scatter in the line y residuals has stayed the same. Clearly the previous parameter value for `detectarclines-min-sn` produced better results. Therefore, change the value of `detectarclines-min-sn` back to 5.0, re-run the recipe (which should be very quick since the lazy mode detects that these parameter values have been run before), and then click the `Continue Wkf` button.

A.2 Order Position

The next pipeline recipe to be executed (still within the `Spectral Format` actor) is `xsh_orderpos`, which accurately traces the order centres on an input single-pinhole continuum-lamp frame.

A.2.1 Interactive Window

The **top** plot in the interactive window shows the continuum-lamp frame with the order traces over-plotted with green points. The **bottom** two plots show the trace residuals of the polynomial fit to the detected order traces as a function of detector x and y coordinates. The results from this recipe are generally very robust with the default parameters.

Click the `Continue Wkf` button in the interactive window.

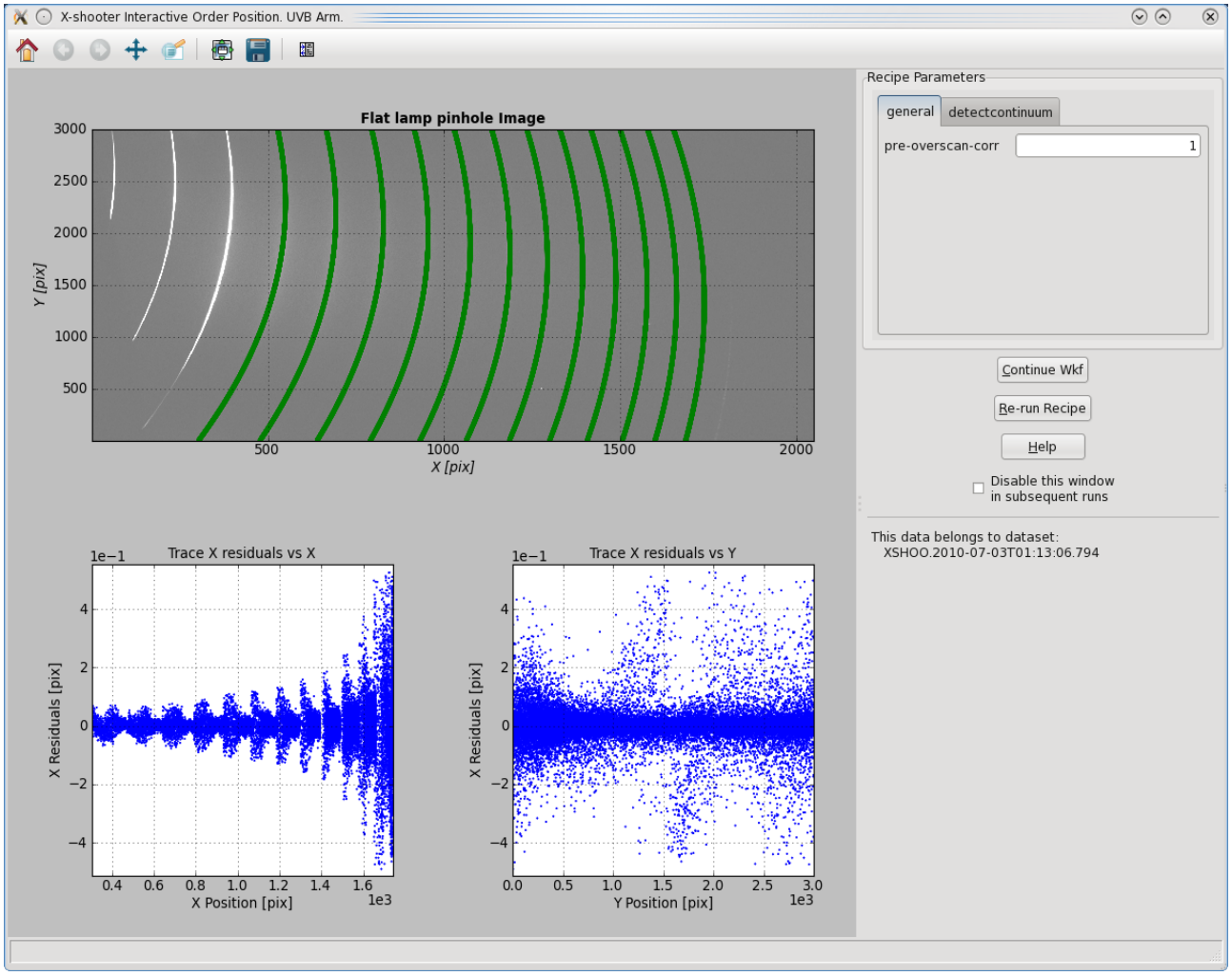


Figure A.3: *The interactive window for the* `Spectral Format` *actor for the recipe* `xsh_orderpos`.

A.3 Master Flat Creation

This recipes creates master flat fields and determines the edges of the slit (or the IFU slices).

A.3.1 Interactive Window

The interactive window shows the master flat frame with the left, right, and centre order traces over-plotted as yellow, green, and blue points, respectively (see Figure A.4). For the current master flat, there is a bad column flagged at $x = 1327$ pix.

A number of parameters are exposed in the "Recipe Parameters" box to allow the user to iteratively improve the master flat frame. The default parameters, however, are known to produce good quality master flat frames and the interactive window does not provide any metrics measuring the quality of the master flat frame.

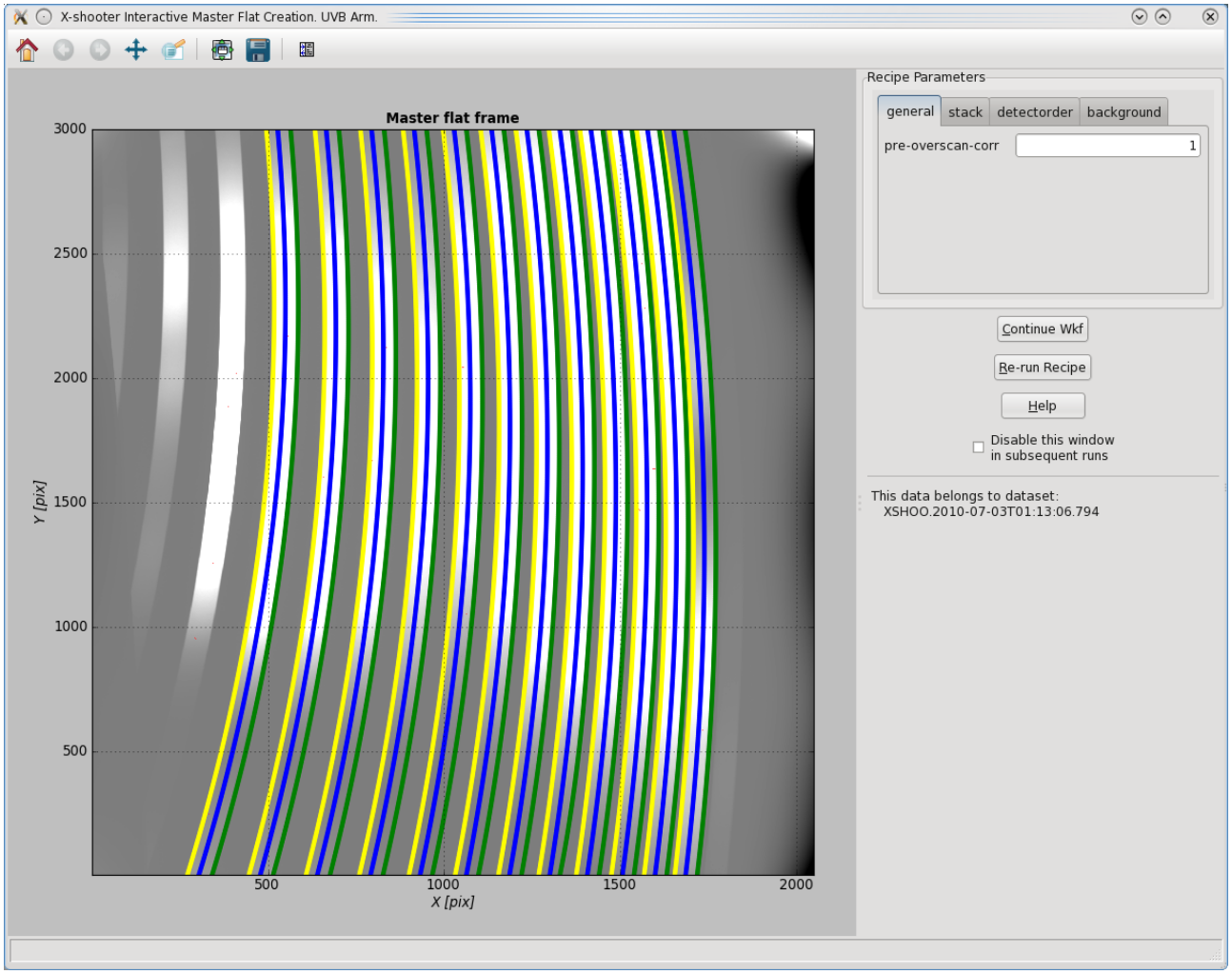


Figure A.4: *The interactive window for the* `Master Flat Creation` *actor for the recipe* `xsh_mflat`.

Click the `Continue Wkf` button in the interactive window. Notice that the `Master Flat Creation` interactive window will appear once more for the second UVB master flat frame that is created. This is because two different master flat frames are created for this DataSet: one is created for performing the flat fielding of the multi-pinhole arc-lamp frame used in determining the wavelength solution, and the other is created for performing the flat fielding of the science object observation. So click on the `Continue Wkf` button again in the interactive window to proceed.

A.4 2D Geometry

The next step in the X-shooter reduction cascade is the determination of the full wavelength and spatial distortion solution using the recipe `xsh_2dmap`. The input to `xsh_2dmap` is a 9-pinhole arc-lamp frame in which the arc lines are detected and their positions measured. These arc line positions and corresponding identifications are then used to fit the parameters of a full physical model of the X-shooter instrument. The best fit parameters of the physical model provide a wavelength and slit position for any position along any of the orders in an X-shooter spectrum image.

A.4.1 Interactive Window

The interactive window (Figure A.5) shows the 9-pinhole arc-lamp frame in the **top** panel with overlaid points marking catalogued arc lines that are not detected (blue), that are successfully detected but have too small S/N (yellow), or that are successfully detected with sufficient S/N (green), respectively. The **bottom** two plots show the position residuals along the x and y image axes as a function of wavelength for the successfully detected arc lines with sufficient S/N.

The wavelength calibration solutions produced by the X-shooter pipeline are generally robust and of good quality, showing only some low level (0.2-0.3 pix peak-to-peak) systematic trends in the residuals and a few outlier points. The main purpose of this interactive window is therefore to allow the user to check that all is well with the full wavelength solution, and to allow some further refinement if so required. Most users will simply accept the solution that is presented by clicking on the `Continue Wkf` button.

The interactive window gives access to six pipeline recipe parameters that may be adjusted in order to improve the wavelength calibration solution. We suggest that you attempt to improve the wavelength calibration solution for the current DataSet, based on the parameter descriptions that are provided when you move the mouse over the parameter window.

We find that for the current DataSet, setting `model-maxit = 1000`
and `model-anneal-factor = 0.5` improves the mean x and y residuals from ∼0.0718 and
∼0.0880 pix for the default parameters to ∼0.0677 and ∼0.0804 pix, respectively. These statistics may be found in the corresponding "esorex.log" files in the `LOGS_DIR` for `xsh_2dmap`.

Click the `Continue Wkf` button in the interactive window.

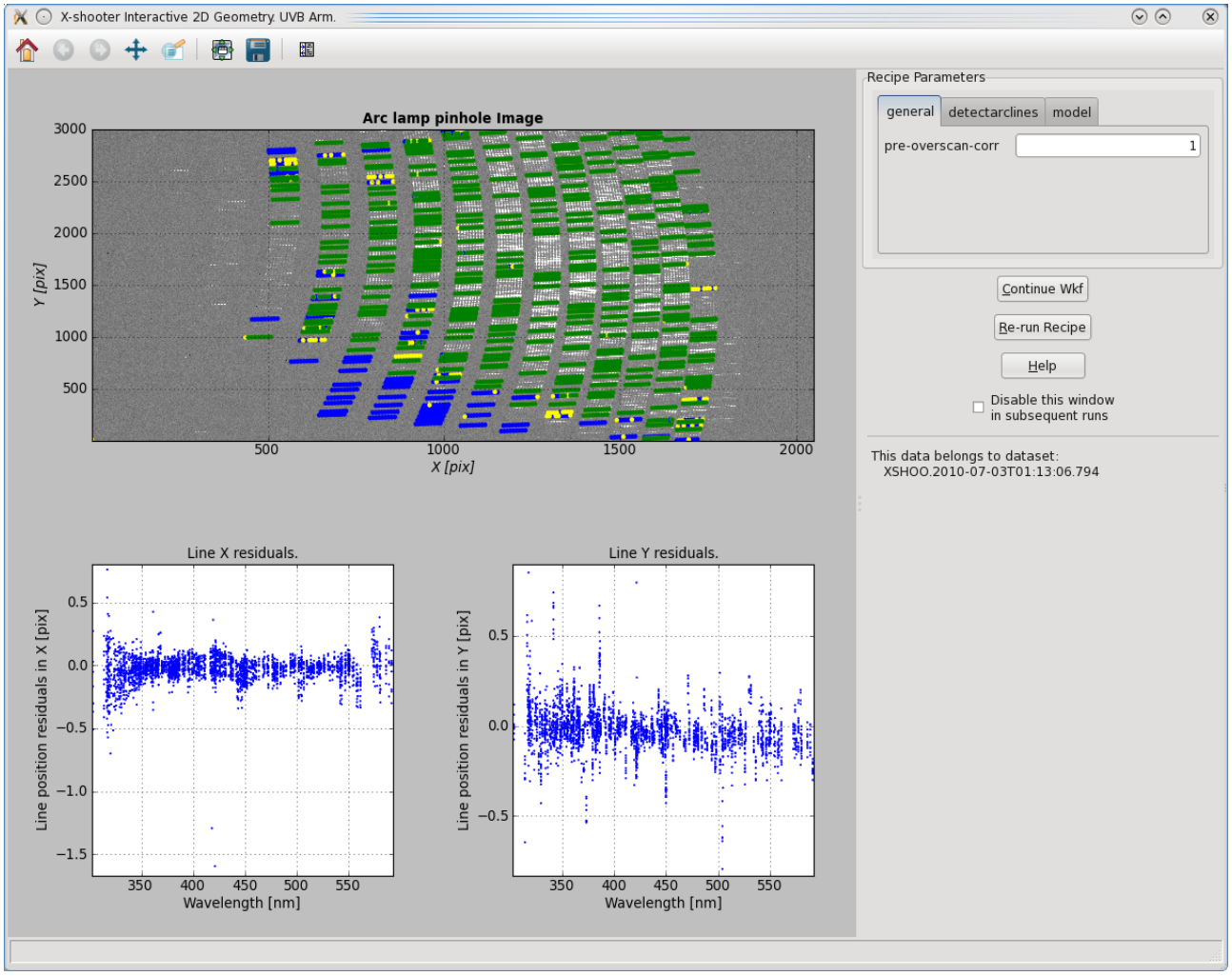


Figure A.5: *The interactive window for the* `2D Geometry` *actor for the recipe* `xsh_2dmap`.

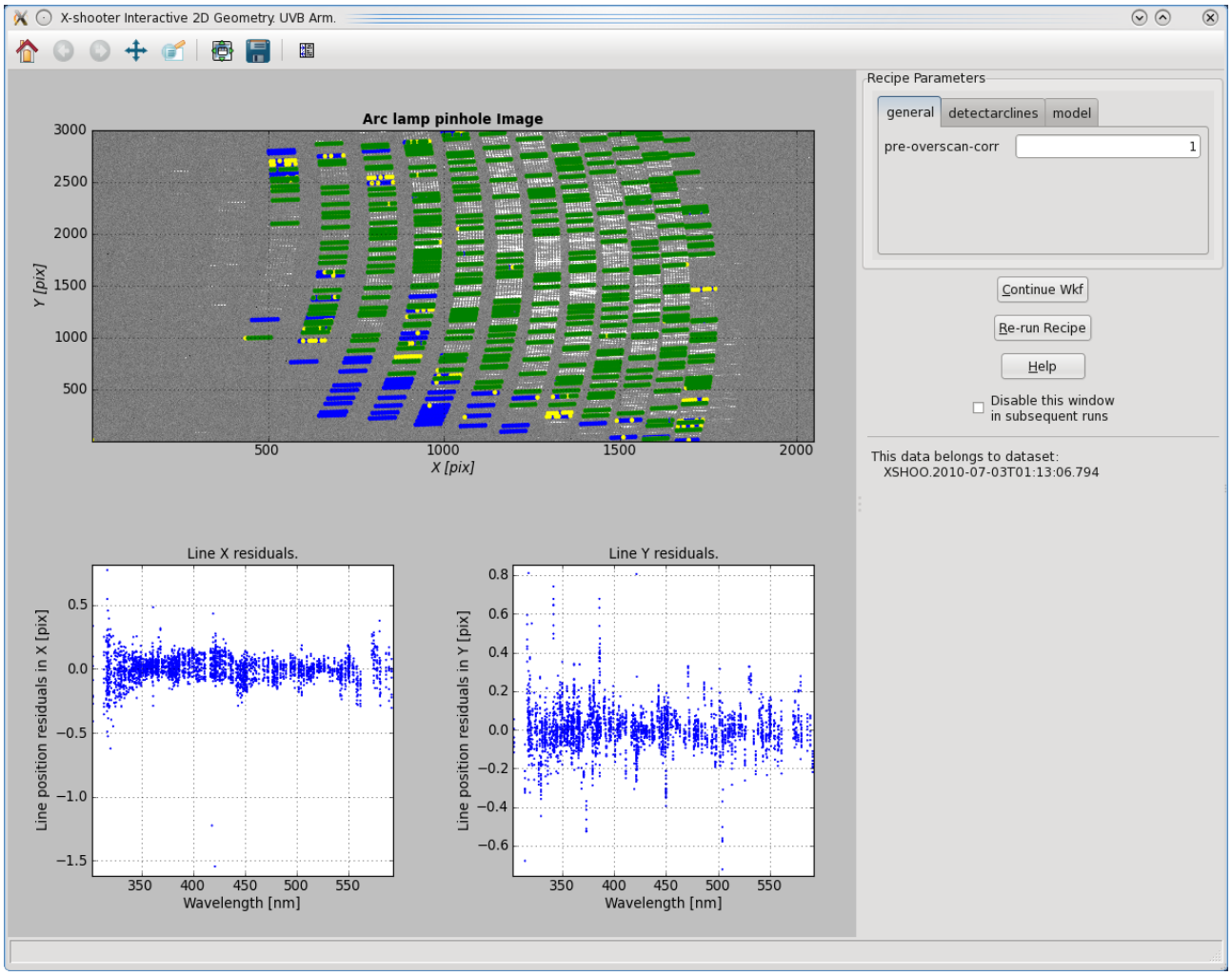


Figure A.6: *The interactive window for the* `2D Geometry` *actor for the recipe* `xsh_2dmap` *with* `model-maxit` = *1000 and* `model-anneal-factor` = *0.5.*

Acknowledgements

The `Reflex` team in alphabetical order consists of Pascal Ballester, Daniel Bramich, Vincenzo Forchì, Wolfram Freudling, César Enrique García, Maurice Klein Gebbinck, Andrea Modigliani, Sabine Moehler & Martino Romaniello. Our thanks go to R. Figuera Jaimes for volunteering to be our first external test user.

Forchì V., 2012, Reflex User Manual, VLT-MAN-ESO-19000-5037, Issue 3.4 17, 25, 60, 61

Martayan C. et al., X-shooter User Manual, VLT-MAN-ESO-14650-4942, Issue 89 9

Modigliani A. & Bramich D.M., X-Shooter Pipeline User Manual, VLT-MAN-ESO-14650-4840, Issue 3.9.1 9, 25, 33, 34, 38, 40, 45