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-19500-....

Issue 1.1

Date 2012-04-23

	Name	Date	Signature
Prepared:	Sabine Moehler, Daniel Bramich	2012-04-23	
Approved:	W.Freudling		
Released:	F. Comeron		

This page was intentionally left blank

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
1.0	28/02/2012	All	Beta release
1.1	20/03/2012	All	Comments from Beta release

This page was intentionally left blank

Contents

1 Introduction And Scope

`Reflex` is the ESO Recipe Flexible Execution Workbench, an environment to run ESO VLT pipelines which employs a workflow engine (Kepler[1]) to provide a real-time visual representation of a data reduction cascade, called a workflow, which can be easily understood by most astronomers. This document is a tutorial designed to enable the user to employ the X-shooter workflow to reduce his/her data in a user-friendly way, concentrating on high-level issues such as data reduction quality and signal-to-noise (S/N) optimisation.

A workflow accepts science and calibration data, as delivered to PIs in the form of PI-Packs (until October 2011) or downloaded from the archive using the CalSelector tool[2] 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 through the workflow which executes the relevant pipeline recipes (or stages) in the correct order, providing optional user interactivity at key data reduction points with the aim of enabling the iteration of certain recipes in order to obtain better results. 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 and employing user-configurable file names.

This tutorial deals with the reduction of X-shooter SLIT observations only via the X-shooter `Reflex` workflow, using the Physical Model mode of the X-shooter pipeline. The user is referred to the X-shooter user manual (Martayan et al. 2011[3]) for more information on the instrument itself, and the X-shooter pipeline user manual (Modigliani 2011[4]) for the details of X-shooter pipeline recipes. The quick start section (see Section 4) describes the minimum effort to get started, and it makes up only two pages of text in this tutorial.

User support for this software is available by sending enquiries to `usd-help@eso.org`.

[1] https://kepler-project.org

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

[3] available at
http://www.eso.org/sci/facilities/paranal/instruments/xshooter/doc

[4] available at ftp://ftp.eso.org/pub/dfs/pipelines/xsh/xsh-pipeline-manual-7.0.pdf

2 Software Installation

The software pre-requisites for `Reflex 2.0` may be found at:

```
http://www.eso.org/sci/software/pipelines/reflex_workflows
```

To install the `Reflex 2.0` software and demo data, please follow these instructions:

1. From any directory, download the installation script:

   ```
   wget ftp://ftp.eso.org/pub/dfs/reflex/install_reflex
   ```

2. Make the installation script executable:

   ```
   chmod u+x install_reflex
   ```

3. Execute the installation script:

   ```
   ./install_reflex
   ```

   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. You will be given a choice of pipelines to install. Please specify the numbers for the pipelines you require, separated by a space, or type “A” for all pipelines.

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

   ```
   <install_dir>/bin/reflex
   ```

   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.

3 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 ∼1.6 GB, ∼0.9 GB and ∼3.1 GB of free disk space for the directories `<download_dir>`, `<install_dir>` and `<data_dir>`, respectively. The X-shooter demo data have been retrieved with the CalSelector tool[5], with AFC data for the telluric standard stars added manually. The demo data sets contains only `SLIT` observations, as the workflow does currently not support `IFU` observations. There is one data set each for `UVB` data taken in `OFFSET` mode, `VIS` data taken in `STARE` mode, and `NIR` data taken in `NODDING` 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. 4.0.3).

Before March 2011, the AFC frames were not used for the processing of SLIT observations. Therefore they are not delivered via CalSelector for data from that time. The AFC frames of standard stars are also never part of any PIPack. In order to retrieve AFC frames please go to
`http://archive.eso.org/wdb/wdb/eso/xshooter/form` and select

`DPR CATG`	`CALIB`
`DPR TYPE`	`LAMP_AFC`
`OBS ID`	`OBS ID` of standard star/science data
`OBS TARG NAME`	`OBS TARG NAME` of standard star/science data
`Night`	date at start of the night (changes at 12:00 UT)
`INS.OPTI2.NAME`	`SLOT`
`INS.OPTI<num>.NAME`	`Pin_0.5` (with `<num>` = `3/4/5` for UVB/VIS/NIR)

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

4 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 `Reflex 2.0` release. By following these steps, the user should have enough information to attempt a reduction of his/her own data without any further reading:

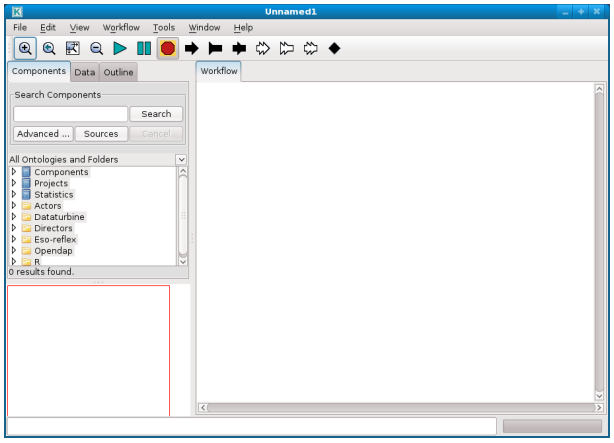


Figure 4.0.1: *The empty* `Reflex` *canvas.*

1. Start the `Reflex` application:

   ```
   reflex &
   ```

   The empty `Reflex` canvas as shown in Figure 4.0.1 will appear.

2. Now open the X-shooter workflow by clicking on `File -> Open File`, selecting the file `xsh-1.4.6/Xshooter.xml` in the file browser. You will be presented with the workflow canvas shown in Figure 4.0.2. Note that the workflow will appear as a canvas in a new window.

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 (1 ms is recommended), and click `OK`.

4. Under "Setup Directories" in the workflow canvas there are seven parameters that specify important directories (green dots). Setting the value of `ROOT_DATA_DIR` is the only necessary modification if you want to process data other than the demo data[6], since the value of this parameter specifies the working directory within which the other directories are organised. Double-click on the parameter `ROOT_DATA_DIR` 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.

[6] If you used the install script `install_reflex`, then the value of the parameter `ROOT_DATA_DIR` will already be set correctly to the directory where the demo data was downloaded.

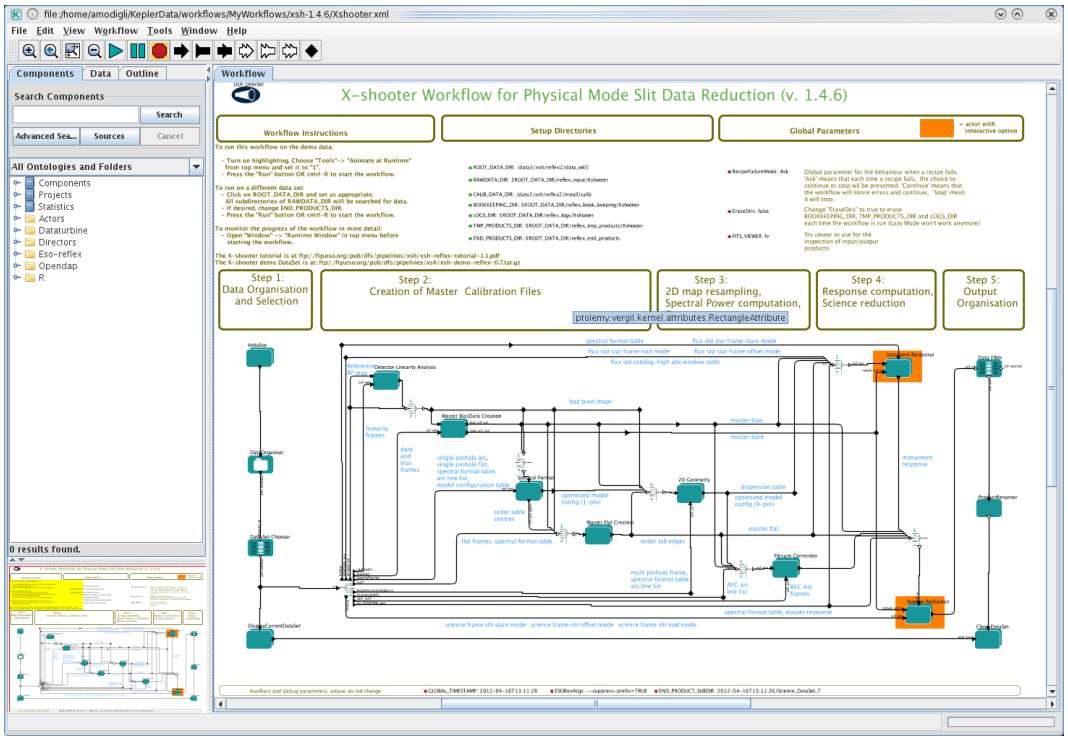


Figure 4.0.2: *X-shooter workflow general layout.*

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 `RAWDATA_DIR` under "Setup Directories" in the workflow canvas) and constructs the DataSets. Note that the calibration and reference data must be present either in `RAWDATA_DIR` or in `CALIB_DATA_DIR`, otherwise DataSets may be incomplete and cannot be processed.

7. The `Data Set Chooser` actor will be highlighted next and will display a "Select Datasets" window (see Figure 4.0.3) that lists the DataSets along with the values of a selection of useful header keywords, namely the object name, X-shooter arm, observing mode (SLOT/IFU), observing technique (STARE/NODDING/OFFSET), detector read mode (for UVB/VIS detectors only), slit widths (`INS.OPTI.<num>`, for `num=3/4/5` corresponding to UVB/VIS/NIR, respectively), exposure time, DIT (defined for NIR data only), and OB number. The first column consists of a set of tick boxes which allow the user to select the DataSets to be processed, and by default all DataSets are selected.

   You will find that one DataSet is grey and has no tick box. This is an incomplete DataSet provided to demonstrate how such DataSets look in the "Select Datasets" window. Moving the mouse over this grey DataSet will give you the information which kind of files are missing (see Fig. 9.0.2 for illustration).

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. When the workflow has finished executing the pipeline recipe `xsh_respon_slit_offset` in the

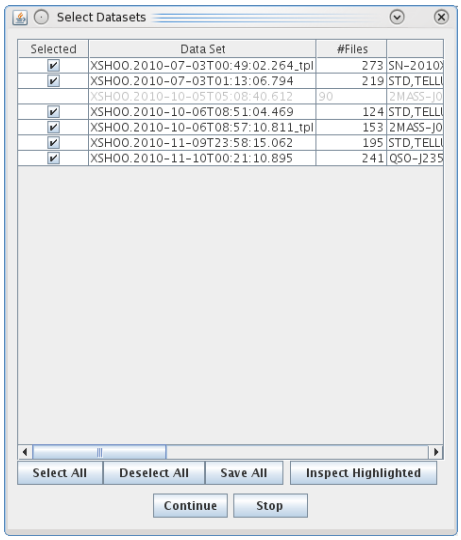


Figure 4.0.3: *The "Select Datasets" pop-up window. On purpose the demo includes an incomplete (grey) DataSet.*

`Instrument Response` actor for the first DataSet, an interactive window will appear (see Figure 7.0.2) which shows a plot of the extracted and merged standard star spectrum in the top panel (and the response curve, if the standard star is found in the catalogue). Using the buttons at the top of this window, one may pan and zoom in on the spectrum in order to inspect absorption/emissions lines and other interesting spectral features. Below the extracted spectra the 2-dimensional rectified data are displayed - sky-corrected and sky (the latter only for STARE and OFFSET data).

10. When the workflow has finished executing the final pipeline recipe `xsh_scired_slit_stare/nod/offset` in the `Spectrum Reduction` actor for the first DataSet, an interactive window will appear (see Figure 4.0.4) which shows a plot of the extracted and merged spectrum in the top panel (and the flux-calibrated version, if a suitable instrument response curve exists, in the panel below). Below the extracted spectra the 2-dimensional rectified data are displayed - sky-corrected and sky (the latter only for STARE and OFFSET data).

11. Click on the `Continue wkf` button and the workflow will write out the important products of the reduction cascade to the end products directory (specified by the parameter `END_PRODUCTS_DIR` under "Setup Directories" in the workflow canvas), which includes the extracted and merged spectrum.

12. The workflow will automatically move on to the next DataSet repeating the reduction cascade and displaying the interactive windows with the response curves and extracted and merged spectra, respectively. For each DataSet the procedure is the same; simply inspect the response curve and the science spectrum and then continue until all data sets are processed.

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.

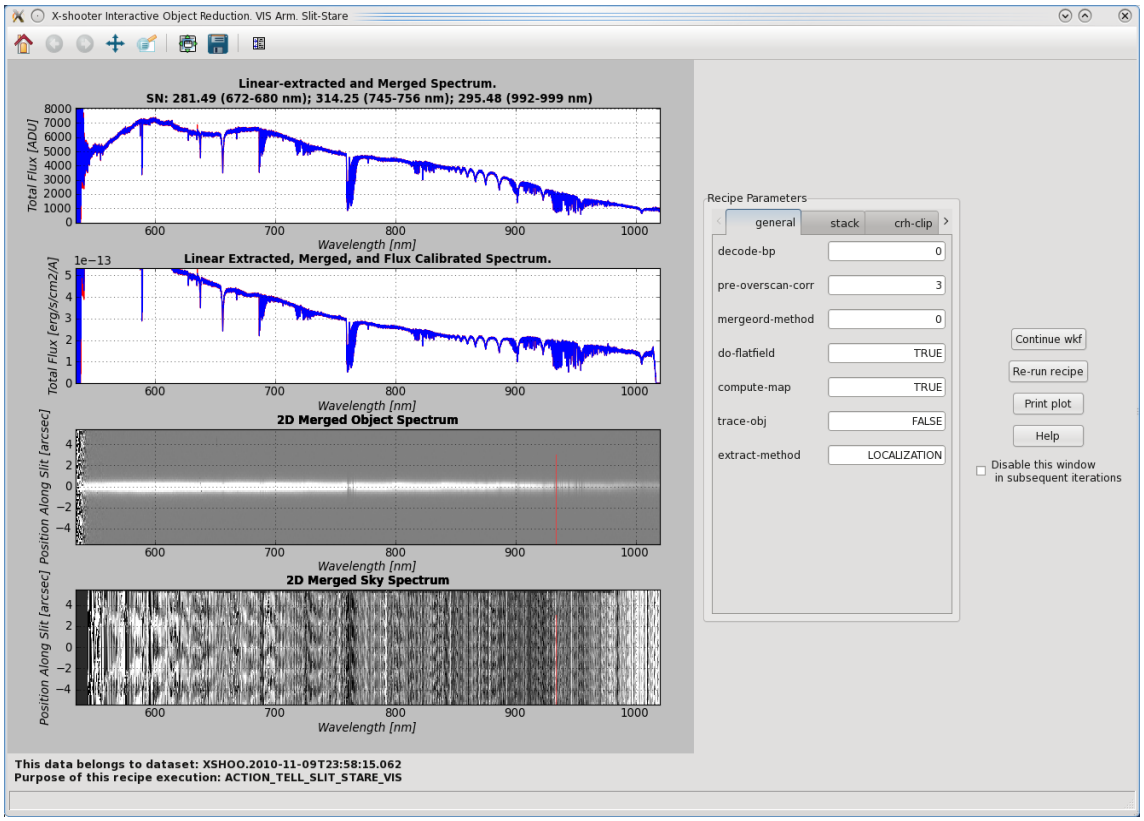


Figure 4.0.4: *The interactive pop-up window for the* `Science Reduction` *actor and X-shooter pipeline recipe* `xsh_scired_slit_stare`*. The extracted and merged spectrum for the telluric standard star in the VIS DataSet is displayed in the top panel. The next panel show the flux-calibrated spectrum.*

5 About The Reflex Canvas

5.1 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 `Reflex` sessions using `File -> Open`.

5.2 Buttons

At the top of the `Reflex` canvas are a set of buttons which have the following useful 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.

5.3 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. Note that after clicking the  button, it is possible that more than one actor is executed, since this behaviour depends on the workflow scheduling. For instance, if there are two actors in parallel, and you pause the workflow while one is being executed, then both of them will be executed before the workflow is actually paused. After pausing, the workflow may be resumed by clicking the  button again.

5.4 The Runtime Window

You may find the runtime window a useful aid in monitoring the reduction progress of your data. This window may be started by clicking `Workflow -> Runtime Window` from the `Reflex` canvas menu. You will notice that on the left-hand side the runtime window has buttons allowing the control of the workflow (`Go`, `Pause`, `Resume`, `Stop`) and text boxes for controlling workflow parameters such as the working data directory etc.

On the right-hand side of the runtime window is a text box with the title "Recipe Status" which lists the current status of each pipeline recipe and the reduction status of each DataSet. A recipe may have the following status values:

- `Not Running` - The recipe has not yet been run for any DataSet so far.
- `Executing` - The recipe is currently executing for a DataSet.
- `Done` - The last execution of the pipeline recipe was successful.
- `Failed` - The recipe failed on the last DataSet.
- `Skip` - The recipe was skipped for the last DataSet.
- `Disabled` - The recipe was disabled for the last DataSet.
- `Stopped` - The workflow was stopped during the reduction of a DataSet.

Below the list of recipe status values is a detailed list of input and output files used for each DataSet within each recipe execution. This information is sometimes very useful for the user who wants to know exactly which files were used as input for a particular DataSet for a given pipeline recipe, and where the relevant output files were written.

6 The X-shooter Workflow

The X-shooter workflow canvas is organised into a number of areas (see Figure 4.0.2). 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.

6.1 Workflow Canvas Parameters

The workflow canvas displays a number of parameters that may be set by the user (see Figure 4.0.2). Under "Setup Directories" the user is only required to set the `ROOT_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. Raw data should be stored in a subdirectory of `ROOT_DATA_DIR`, defined by the parameter `RAWDATA_DIR`, which is recursively scanned by the `Data Organiser` actor for input raw data. The directory `CALIB_DATA_DIR`, which is 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).

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.

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 6.2.4), reusing the previous pipeline recipe outputs where 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 rereduce the data each time the workflow is run.

The remaining two global parameters are set automatically by the workflow itself and should not be modified.

6.2 Workflow Actors

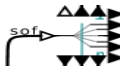
6.2.1 Simple Actors

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

- 

- The `Data Organiser` actor.

- 

- The `Data Set Chooser` actor.
- 

- The `Fits Router` actor
- 

- The `Product Renamer` actor.
- 

- The `Data Filter` 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 (see Figure 6.2.4). Note that the `Product Renamer` actor is a jython script (Java implementation of the Python interpreter) meant to be customized by the user (by double-clicking on it).

6.2.2 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. `ActorName` marks an actor with an associated interactive window (`Instrument Response` and `Science Reduction`). In the X-shooter workflow, the following actors are composite actors:

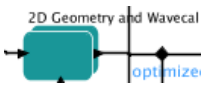
- 

- The `Initialise` actor.
- 

- The `Display Current DataSet` actor.
- 

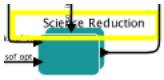
- The `Master BiasDark Creation` actor.
- 

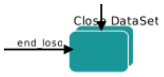
- The `Spectral Format` actor.
- 

- The `Master Flat Creation` actor.
- 

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

- The `Instrument Response` actor (with an interactive window).

- 

- The `Spectrum Reduction` actor (with an interactive window).
- 

- The `Close DataSet` actor.

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

6.2.3 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. `Recipe Executer`).

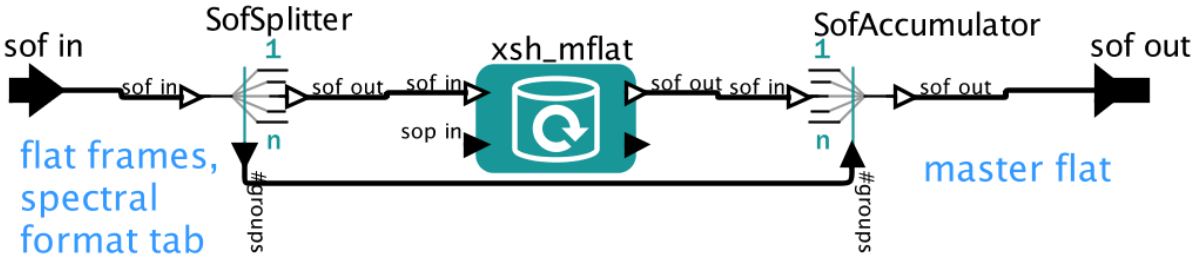


Figure 6.2.1: *This is the window you get when you choose* `Open Actor` *for the Composite Actor* `Master Flat Creation`. *This is the most simple case for a Composite Actor.*

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 6.2.1, which is the most simple version of a Composite Actor within the X-shooter workflow. The `RecipeExecuter` is preceded by an `SofSplitter`, and followed by an `SofAccumulator`. The function of the `SofSplitter` is to investigate the incoming SoFs, sort them by "purpose", and create separate SoFs for each purpose. The `RecipeExecuter` then processes each of the SoFs independently. Finally, the `SofAccumulator` packs all the results into a single output SoF. The direct relation between the `SofSplitter` and `SofAccumulator` is used to communicate the number of different SoFs created by the `SofSplitter`. A workflow will only work as intended if the purpose of all the files a recipe needs as input is identical. The only exception to this

[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`).

Table 6.1: The X-shooter pipeline actors and their contents

actor	recipes	description
Detector Linearity Analysis	`xsh_lingain`	determine detector linearity
Master BiasDark Creation	`xsh_mbias`	create master bias (UVB/VIS)
	`xsh_mdark`	create master dark (NIR)
Spectral Format	`xsh_predict`	spectral format check
	`xsh_orderpos`	determine centers of orders
Master Flat Creation	`xsh_mflat`	create master flat and table for order edges
2D Geometry and Wavecal	`xsh_2dmap`	calibration of wavelength and spatial distortion
	`xsh_wavecal`	resolution, line tilt (not needed for further processing)
Instrument Response	`xsh_respon_slit_stare`	determine response function
	`xsh_respon_slit_nod`	
	`xsh_respon_slit_offset`	
Science Reduction	`xsh_scired_slit_stare`	reduce science data
	`xsh_scired_slit_nod`	
	`xsh_scired_slit_offset`	

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 `SoAccumulator` 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 6.2.3 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. 6.2.2 and 6.2.3).

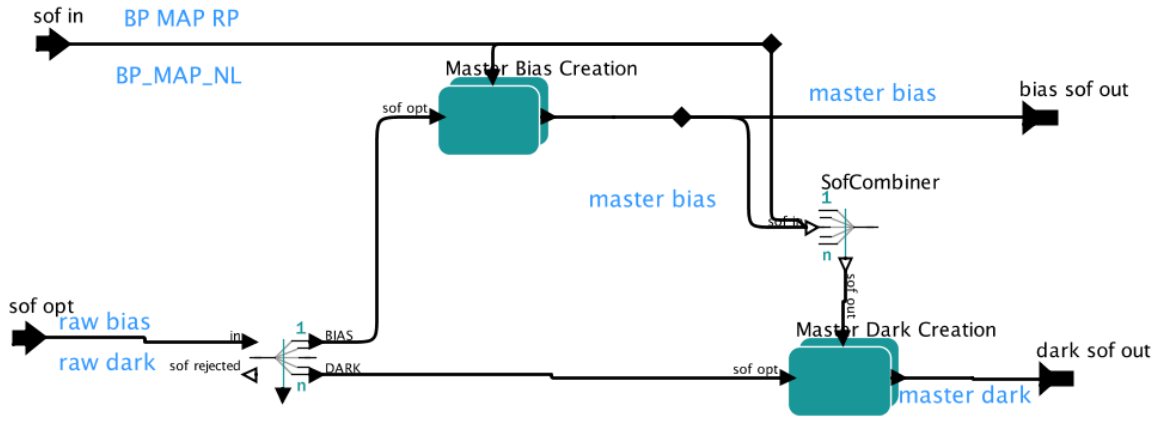


Figure 6.2.2: *This is the window you get when you choose* `Open Actor` *for the Composite Actor* `Master BiasDark Creation`. *Using* `Open Actor` *on* `Master Bias Creation` *gives you Fig. 6.2.3.*.

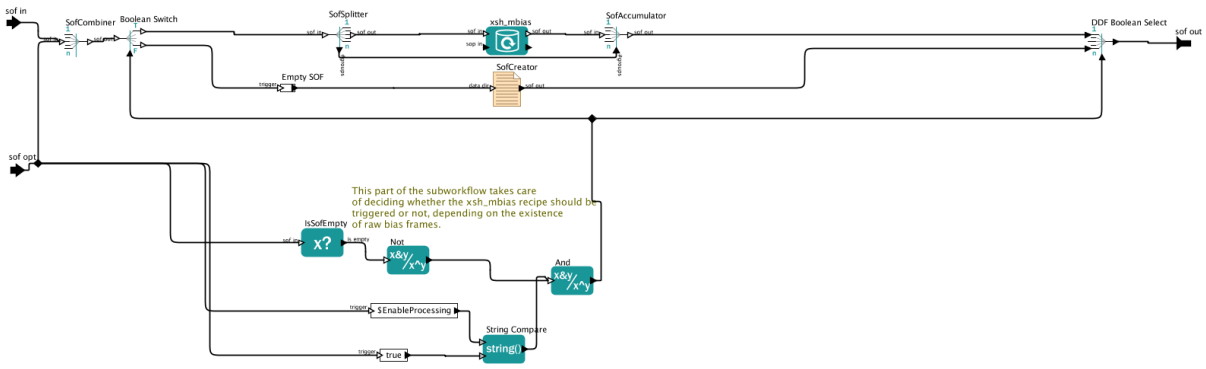


Figure 6.2.3: *This is the window you get when you choose* `Open Actor` *for the Composite Actor* `Master Bias Creation` *within* `Master BiasDark Creation`. *Using* `Configure Actor` *on* `xsh_mbias_1` *gives you Fig. 6.2.4.*

In Figure 6.2.4 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 6.2.4). 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.

- 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. For more information on the X-shooter pipeline recipe parameters, the user should refer to the X-shooter pipeline user manual (Modigliani 2011[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.

6.2.4 Lazy Mode

By default, all recipe executer 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 rereduction where it is not necessary.

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

The forced rereduction of data at each execution may of course be desirable. To force a rereduction of all data for all `RecipeExecuter` actors in the workflow (i.e. to disable Lazy mode for the whole workflow), set the

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

`EraseDirs` parameter under the "Global Parameters" area of the workflow canvas to `true`. This will then remove all previous results as well. To force a rereduction of data for any single `RecipeExecuter` actor in the workflow (which will be inside the relevant composite actor), right-click the `RecipeExecuter` actor, select `Configure Actor`, and uncheck the Lazy mode parameter tick-box in the "Edit parameters" window that is displayed.

6.3 Workflow Steps

6.3.1 Step 1: Data Organisation And Selection

On clicking the  button on the `Reflex` canvas, 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 6.1).

1. The `DataOrganiser` (DO) is the first crucial component of a Reflex workflow. The DO takes as input `RAWDATA_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.

   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 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. A category could for example be `FLAT_SLIT_VIS`, `DARK_NIR` or `SCI_SLIT_OFFSET_UVB`. 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. 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_MBIAS_VIS` and `ACTION_MFLAT_SLIT_VIS`. The former creates a master bias from raw biases, and the later creates a master flat from raw flats. The `ACTION_MFLAT_SLIT_VIS` action needs raw flats and the master bias as input. In this case, these biases will have the purpose `ACTION_MBIAS_VIS/ACTION_MFLAT_SLIT_VIS`. The same DataSet might also include biases with a different purpose, e.g. `ACTION_MBIAS_VIS/ACTION_MDARK_VIS`. Irrespective of their purpose the file category for all these biases will be `BIAS_VIS`.

2. Next the `DataSet Chooser` displays the DataSets available in the "Select Data Sets" window[10], activating a vertical scroll bar on the right if necessary (see Figure 4.0.3). 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)

[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 asscociate the required calibration data for processing. They can be found in the directory `<install_dir>/share/esopipes/<pipeline-version>/reflex/`, carrying the extension `.oca`

[10] If you run the `Data Organiser` in Lazy Mode, changes in the `Keywords to be displayed` list will have no effect on the output shown in the `DataSet Chooser`.

DataSets for processing using the tick boxes in the first column, and the buttons `Select All` and `Deselect All` at the bottom left.

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 and path for each file, the file category (from the FITS header), and a selection tick box in the right column (see Figure 6.3.1). 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 (see Figure 6.3.1), 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`, and 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.

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` (see Sect. 6.2.3). 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`. Note that while the DO includes files into a DataSet for a reason, and records this reason as the "purpose" of the file, the workflow itself can use these files in a different manner. The `SofSplitter` and `SofAccumulator` accept several SoFs as simultaneous input. The `SofAccumulator` creates a single output SoF from the inputs, whereas the `SofSplitter` creates a separate output SoF for each purpose.

6.3.2 Step 2: 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 2011: 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.

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 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 a1-pinhole flat field.

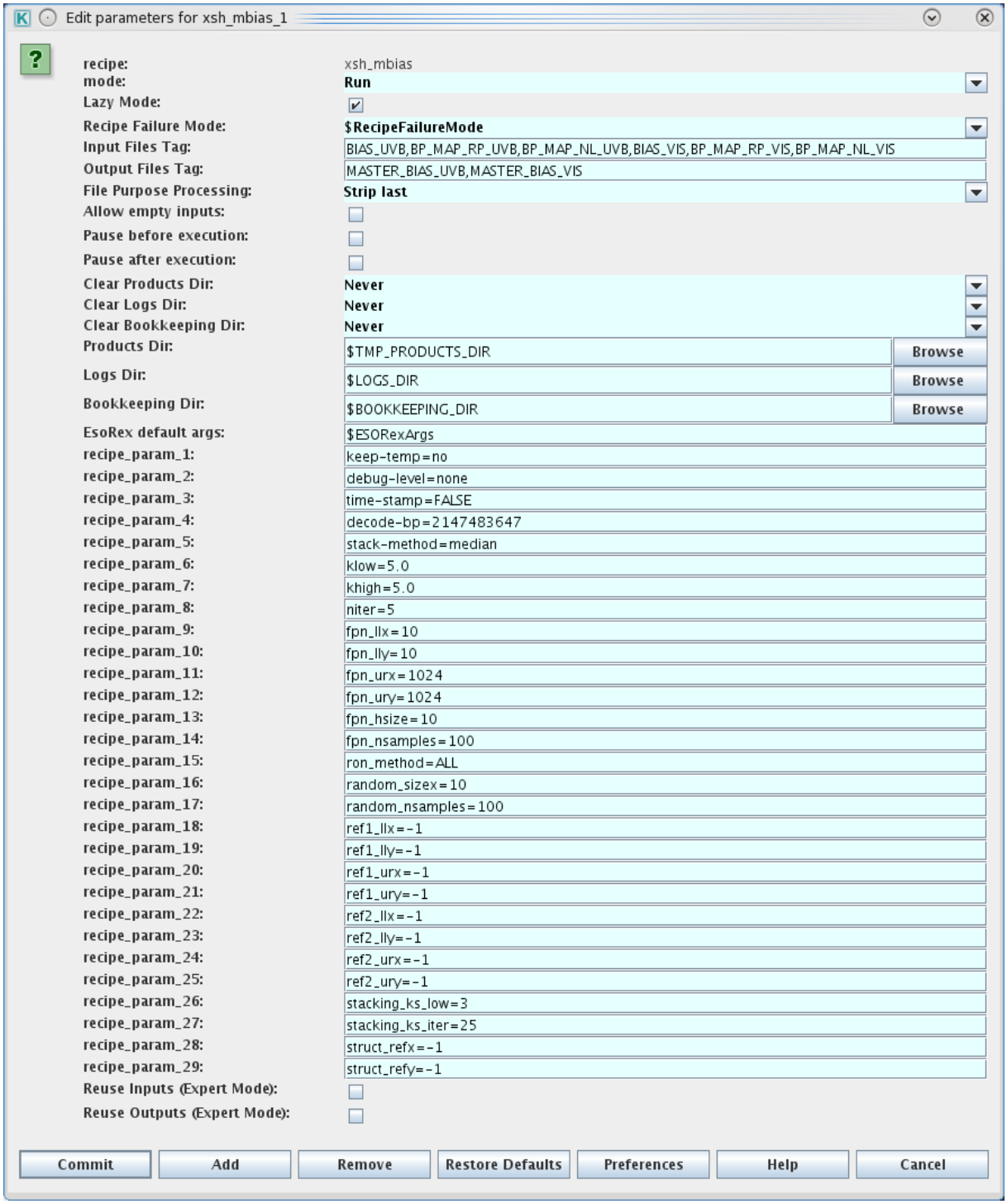


Figure 6.2.4: *The "Edit parameters" window for a typical* `RecipeExecuter` *actor, the* `xsh_mbias_1` *actor which runs the* `xsh_mbias` *pipeline recipe.*

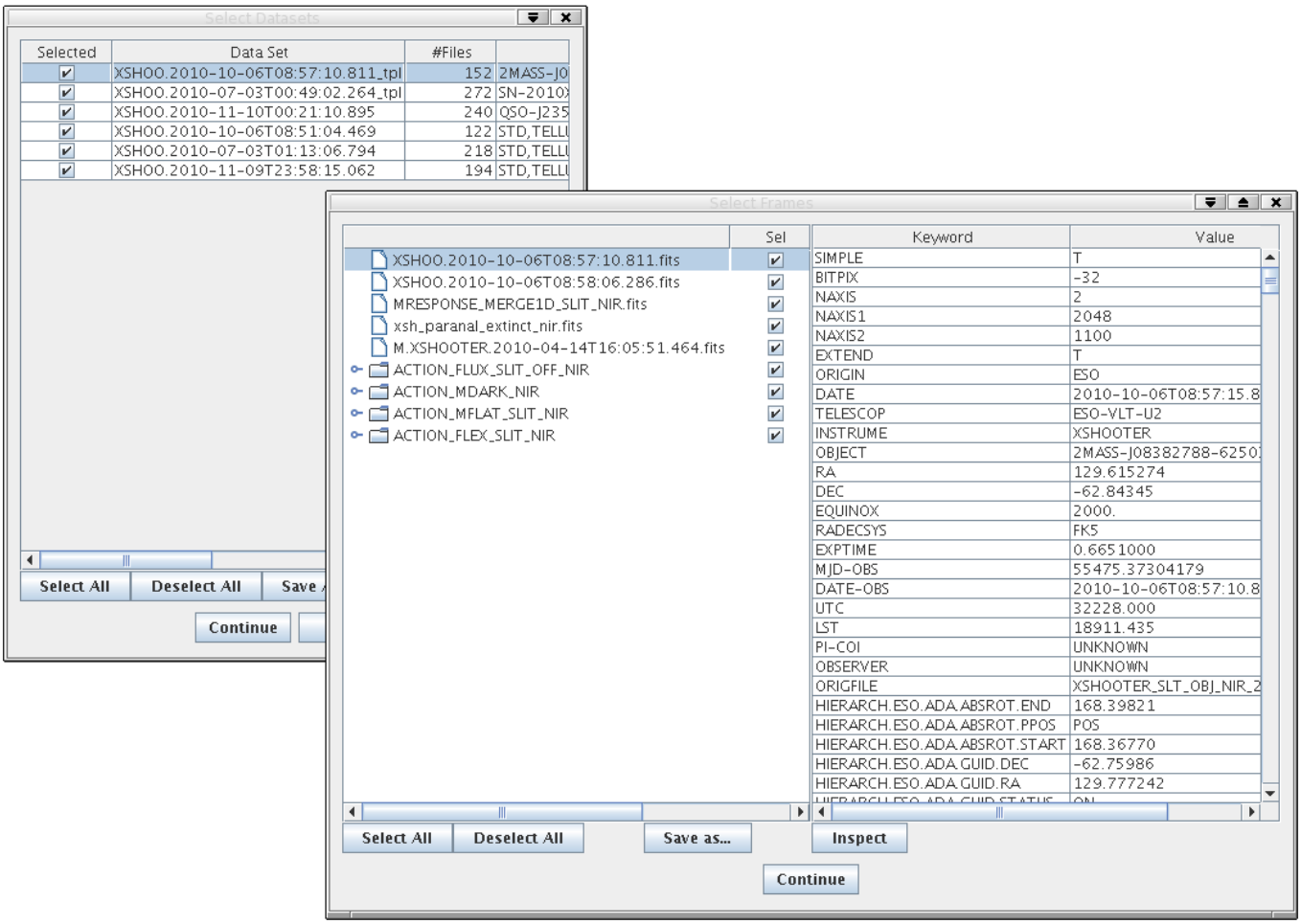


Figure 6.3.1: *The "Select Frames" window with a single file from the current Data Set highlighted in blue, and the corresponding FITS header displayed in the text box on the right. Hidden partially behind the "Select Frames" window is the "Select DataSets" window with the currently selected DataSet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

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.

6.3.3 Step 3: 2D Distortion Mapping, Spectral Resolution

In this step of the workflow, the `2D Geometry and Wavecal` actor will execute the X-shooter pipeline recipes listed below. Please refer to the X-shooter pipeline user manual (Modigliani 2011: Sections 9, 10 & 11) for the details of each recipe and the algorithms employed:

1. `xsh_2dmap` in order to create a wavelength calibration and spatial distortion solution from an input 9-pinhole arc-lamp frame. 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.

2. `xsh_wavecal` in order to determine the spectral resolution, line tilt, and offset with respect to the 9-pinhole positions from an input arc-lamp slit frame. The results of this recipe are not used for further procesing but serve only as a quality check. Since they are not used for processing the raw frames needed for this recipe are usually not part of the `CalSelector` delivery and the recipe is currently not executed.

6.3.4 Step 4: 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. 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 2011: 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 used 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) 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 (if the DataSet contains no master response curve).

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 7.0.2).

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 if it was taken within ± 3 nights of the science observation. Generally, X-shooter PI-Packs as well as CalSelector data sets for data taken after 2011-07-15 are supplied with master instrument response curves which will also be included in the data set for a science observation.

The `Science Reduction` actor will execute the X-shooter pipeline recipes `xsh_scired_slit_stare`, `xsh_scired_slit_nod`, or `xsh_scired_slit_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 science observation using the master instrument response curve if it exists in the current DataSet, and failing this, the instrument response curve derived from the standard star observation will be used if it exists in the current DataSet. There are two possibilities to force the workflow to use the response curve from the standard star: remove the master response curve from both `RAWDATA_DIR` (if it exists there) and from the `CALIB_DATA_DIR` (good for large amounts of data) or deselect the master response curve within the DataSets. If neither a standard star observation nor a master instrument response curve exist in the current DataSet, then the science observation will not be 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 4.0.4).

6.3.5 Step 5: Output Organisation

After having processed the input data for a DataSet, the `Data Filter` actor allows the user to inspect the results and select which files to save. By default this actor is set to skip and all products will be saved. Next 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>.` 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_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 data).

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 Sections 5 & 6 along with the X-shooter demo data supplied with `Reflex2.0` to illustrate how to optimise the scientific products in terms of quality and S/N. This is work in progress and the contents of this section will grow as we gain experience. Optimising the results 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 demo DataSets as described in Section 4, 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 these steps in order to optimise the reductions for the X-shooter demo data supplied with `Reflex2.0`:

1. Carry out the first four steps described in the Quick Start Section 4.
2. In the X-shooter workflow, the two interactive actors `Instrument Response` and `Science Reduction`) are identifiable by a yellow rectangle encompassing the actor name. Their interactive mode is enabled by default. Should you wish to change that first use `Open Actor` to get access to the the components of the interactive actors, then double-click on the composite actors, setting the "EnableInteractivity" parameter to `false`, and clicking `Commit` to save the changes to the workflow.
3. Click the  button
4. Click the `Continue` button in the "Select DataSets" window.
5. `Instrument Response`: The demo data sets supplied with `Reflex2.0` include standard star observations. In Figure 7.0.2 we show what this interactive window looks like. The usage and purpose of the `Instrument Response` interactive window are the same as for the `Science Reduction` interactive window which we describe in detail in the next few steps, and so no more information on this window is given here in the tutorial. Also the response curve is hardly affected by changes of parameters provided here. The masking of regions with strong telluric absorption has a strong influence on the response curve for VIS and NIR data. In order to change that you have to edit (for instance using `fv`) the FITS table `xsh_high_abs_window_<arm>.fits` which can be found in the `<arm>/REF` directory within the demo data set.
6. `Science Reduction`: In Figure 7.0.3, we show the interactive window associated with the `Science Reduction` actor, which executes the X-shooter pipeline recipe `xsh_scired_slit_stare`, `xsh_scired_slit_nod`, or `xsh_scired_slit_offset`. These pipeline recipes perform the extraction of the science object spectrum (by default a linear extraction is used, but optimal extraction is available as well). The top plot in the interactive window shows the extracted and merged spectrum of the science object (blue), and the plot below this shows the flux-calibrated version (if available). Error bars on the spectrum are plotted as 1-sigma limits in red.

   Below the plots displaying the extracted spectra two windows shows the 2-dimensional merged spectrum (sky-corrected) and the two-dimensional rectified sky (for OFFSET and STARE data, in case of NODDING data the current version shows an empty window there as there is no sky frame for such data).

   Use the buttons at the top of the interactive window to inspect these plots in detail, especially the extracted spectra.

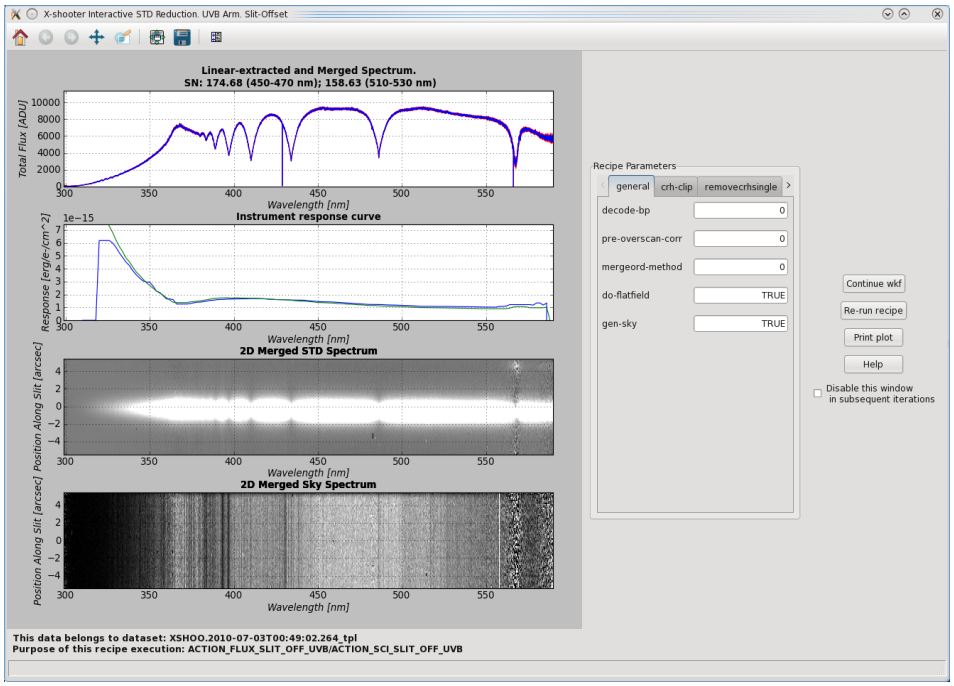


Figure 7.0.2: *The interactive window for the* `Instrument Response` *actor.*

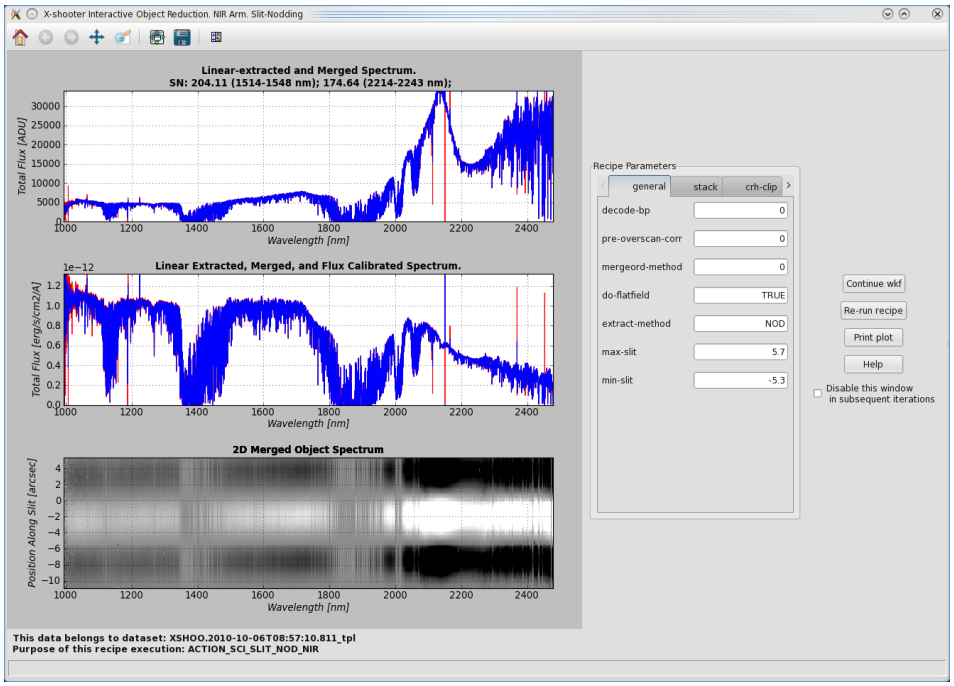


Figure 7.0.3: *The interactive window for the* `Science Reduction` *actor for the NIR DataSet in the data supplied with* `reflex2.0`.

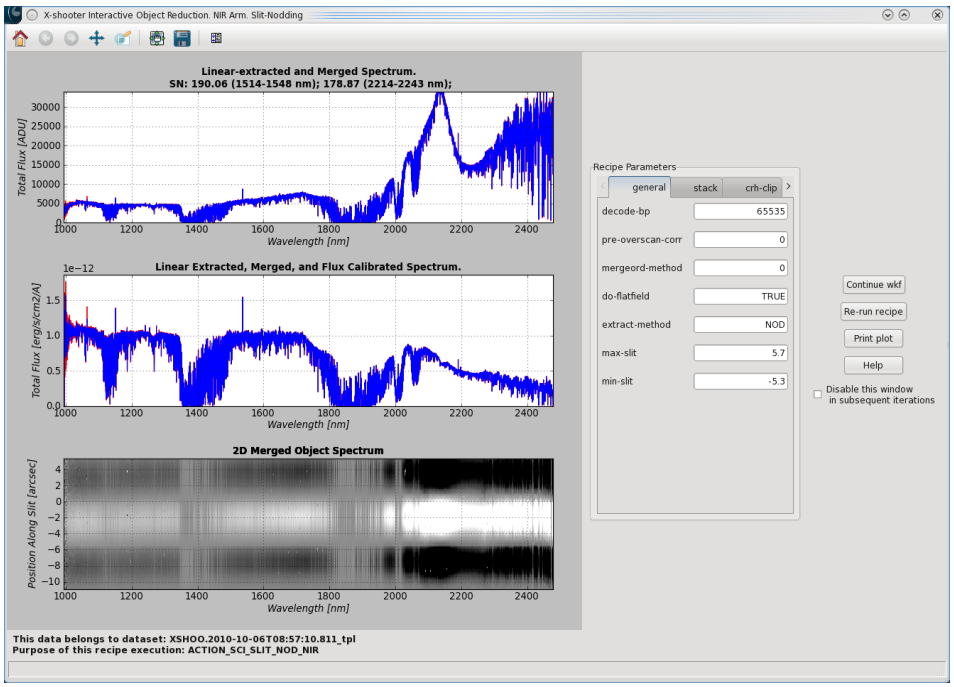


Figure 7.0.4: *The interactive window for the* `Science Reduction` *actor for the NIR DataSet in the data supplied with* `reflex2.0` *after iteration of the corresponding* `xsh_scired_slit_nod` *pipeline recipe with* `decode-bp 65535`.

7. 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 the 2-dimensional merged image by clicking on pixels within the 2-dimensional image (similar to `ds9`).

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

8. The current interactive window provides all pipeline recipe parameters with the aim of allowing the user to optimise the extraction for quality and S/N. For illustrative purposes, set `decode-bp = 65535` for the NIR DataSet[11], and re-run the pipeline recipe. In Figure 7.0.4 we display the interactive window with the bad-pixel corrected spectrum. Notice that the big error bars still visible in Fig. 7.0.3 have vanished.

9. Click the `Continue wkf` button in the interactive window.

10. As noted in the Quick Start Section 4, the workflow will write out the extracted spectrum and other important products to the end products directory (specified by the parameter `END_PRODUCTS_DIR` under “Setup Directories” in the workflow canvas).

11. The workflow will now repeat the above reduction steps for the next DataSet, looping over each DataSet until all DataSets have been processed. The other demo data sets are provided to allow you to experiment with optimising the reductions for other arms and observing modes.

[11] For NIR data it may be helpful in general to change the parameter `-decode-bp` to some high value, to take known bad pixels into account

7.1 Known Problems

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

7.1.1 Workflow Problems

1. **xsk_lingain** We have found a serious conceptual issue with the algorithm implemented in `xsh_lingain`. Hence it is set to "skip" in the workflow.

2. **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.

3. **Failure on 64 bit platform**
   The X-shooter workflow may have run time problems on 64 bit platform. The recipe `xsh_flexcomp` may fail on demo data with the error message `[ ERROR ] xsh_flexcomp: [tid=000] Singular matrix`

7.1.2 Pipeline Problems

1. **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` may be better than `MEDIAN`, but requires some fiddling with the break points (see pipeline manual for more details).

2. **Combination of D2 and QTH flat fields for UVB**
   The combination of two different flat fields (D2 and QTH) causes a discontinuity in the response curve which results in a small peak in the flux calibrated data at about 360nm.

3. **Airmass correction**
   Currently products have their airmass values copied from the first raw science (or standard star) frame. For more than one input frame this is obviously wrong.

4. **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. To avoid that it would be necessary to flat-field the flux standard star spectrum with the same flat field as is used for the science data. This logic cannot be implemented within the OCA framework used for reflex.

5. **Masking of regions with high telluric absorption**
   In order to determine the instrument response only in regions free of telluric absorption we mask regions of telluric absorption. These masking regions have been chosen very conservatively to ensure that even for observations taken with high telluric absorption the response curve is not affected. Conversely this means that especially for the NIR arm most of the spcetrum is masked and the response curve is dominated by linear interpolation. To change that you need to edit (for instance using `fv`) the FITS table `xsh_high_abs_window_<arm>.fits` which can be found in the `<arm>/REF` directory within the demo data set, where the masking regions are defined.

6. **Incomplete** `NODDING` **data**
   A `NODDING` data set 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.

7. **Incomplete** `OFFSET` **data**
   `OFFSET` data containing only `OBJECT` or only `SKY` data cannot be processed by the pipeline. In such cases the workflow will not attempt to process these data, but will ignore them. The DataOrganiser in the current release will not mark such Datasets as incomplete.

8 Frequently Asked Questions

- **Where are my intermediate pipeline products?**

  Intermediate pipeline products are stored in the directory `<TMP_PRODUCTS_DIR>` (defined on the workflow canvas) and organised further in directories by pipeline recipe.

- **I have many DataSets in my data directory. How can I reduce them interactively without having to wait a long time between interactive windows being displayed?**

  Reduce all the DataSets at once with the interactive windows disabled for all interactive actors. When this reduction has finished, you should re-enable the interactive windows that you require, and run the workflow again. The workflow will run in Lazy mode and no time will be spent on pipeline reductions, unless you specifically change a parameter in one of the interactive windows.

  Note that Lazy mode will not work if the workflow parameter `EraseDirs` is set to `true`.

- **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). In Section 6.3.1 of this document, we have described the concept of purpose which is attached to each file in a DataSet. 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.

- **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 to move a Reflex window if it is bigger than the display?**

  That depends on your window manager/desktop. For some window managers one can use Alt+left mouse button to drag the window around desktop.

- **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.

9 Troubleshooting

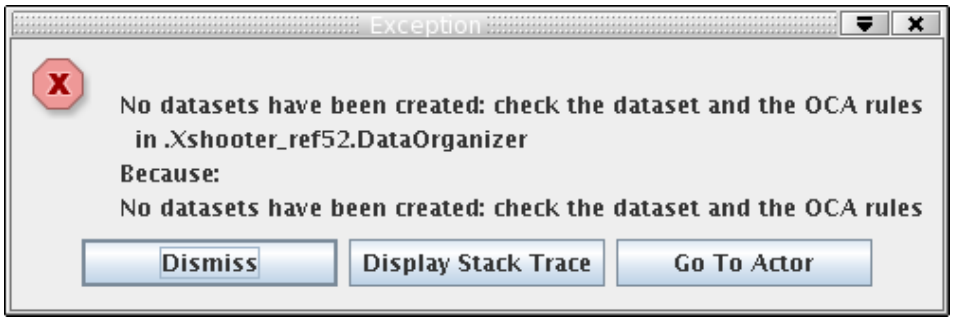


Figure 9.0.1: *TheDataOrganizer interactive window reports an error ":No DataSets have been created, check the data set and the OCA rules.".*

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 9.0.1.)**

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

   Also, please remember that the X-shooter workflow supports data for slit observations only. It is possible that your data consists entirely of IFU observations, in which case the `Data Organiser` actor will not construct any DataSets.

   (c) **all DataSets are greyed out in the DataSets interactive window.**

   The ESO archive used with CalSelector does not always supply all static calibration files (such as for instance the `ARC_LINE_LIST_AFC_UVB`) or flexure compensation data. As a consequence some/all DataSets are greyed out because they were missing such required data.

   Missing static calibration should be found by `reflex` in `<install_directory>/calib/<pipeline_version>/cal`.
   For missing flexure compensation frames see p. 9 on how to retrieve them from the archive.

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

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

the category of the calibration products that are missing (e.g. `MASTER_BIAS_UVB_WKF`). From this the user has then to determine the missing raw data (in this case bias frames for the UVB arm). If static calibrations are missing the mechanism unfortunately does not work, but should be found by `reflex` in `<install_directory>/calib/<pipeline_version>/cal`

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

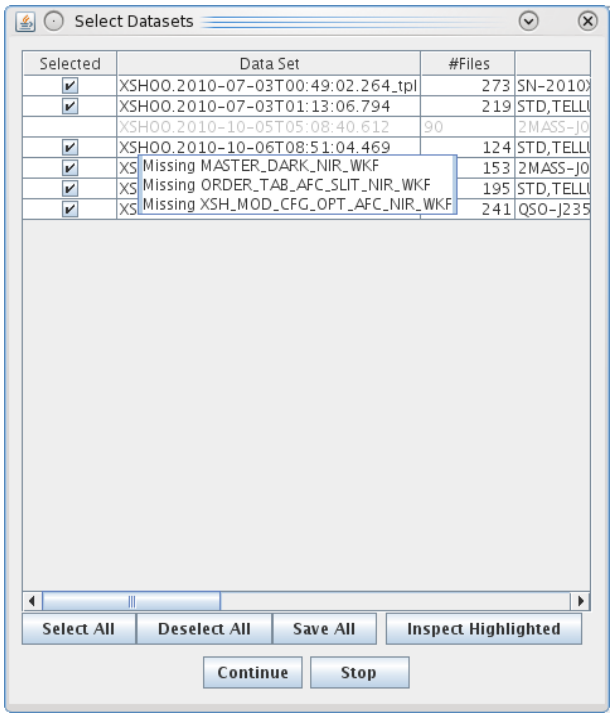


Figure 9.0.2: *This shows how information about missing files in an incomplete DataSet is displayed when moving the mouse over it..*

3. **When I click on the  button in the interactive window (see Section 7, step 6), the window that appears is empty. Can this be corrected?**

   This is a known problem that will addressed in future releases.

4. **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.

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 0.7 16, 21, 34

Martayan C. et al., X-shooter User Manual, VLT-MAN-ESO-14650-4942, Issue 89 7

Modigliani A., X-Shooter Pipeline User Manual, VLT-MAN-ESO-14650-4840, Issue 7.0 7, 21, 23, 26