



# 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 VIMOS/IFU Tutorial

VLT-MAN-ESO-19540-5898

Issue 2.4

Date April 2018

Prepared:	Lodovico Coccato, C. E. García Dabó	April 2018	
	Name	Date	Signature
Approved:	W. Freudling		
	Name	Date	Signature
Released:	M. Sterzik		
	Name	Date	Signature

This page was intentionally left blank

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	3 of 30

### Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
1.0	28/06/2012	All	Initial version
1.1	04/01/2013	All	Put into the standard tutorial format
2.0	04/01/2013	All	New version for archiving purposes
2.3	01/04/2016	All	Updates for version 3.1.3

This page was intentionally left blank

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	5 of 30

## Contents

<b>1</b>	<b>Introduction And Scope</b>	<b>6</b>
<b>2</b>	<b>Software Installation</b>	<b>7</b>
2.1	Installing Reflex workflows via <code>macports</code> . . . . .	7
2.2	Installing Reflex workflows via <code>rpm/yum</code> . . . . .	7
2.3	Installing Reflex workflows via <code>install_esoreflex</code> . . . . .	8
<b>3</b>	<b>Demo Data</b>	<b>10</b>
<b>4</b>	<b>Quick Start: Reducing The Demo Data</b>	<b>11</b>
<b>5</b>	<b>About The Reflex Canvas</b>	<b>15</b>
5.1	Saving And Loading Workflows . . . . .	15
5.2	Buttons . . . . .	15
5.3	Workflow States . . . . .	15
<b>6</b>	<b>The VIMOS Workflow</b>	<b>16</b>
6.1	Workflow Canvas Parameters . . . . .	16
6.2	Workflow Actors . . . . .	17
6.2.1	Simple Actors . . . . .	17
6.2.2	Composite Actors . . . . .	17
6.2.3	Recipe Execution within Composite Actors . . . . .	18
6.2.4	Lazy Mode . . . . .	20
6.3	Workflow Steps . . . . .	21
6.3.1	Step 1: Data Organisation And Selection . . . . .	21
6.3.2	Step 2: Recipe execution . . . . .	22
6.3.3	Step 3: Final products . . . . .	24
<b>7</b>	<b>Frequently Asked Questions</b>	<b>25</b>
<b>8</b>	<b>Troubleshooting</b>	<b>27</b>

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	6 of 30

## 1 Introduction And Scope

Reflex 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 contact [usd-help@eso.org](mailto:usd-help@eso.org) for further support.

This document is a tutorial designed to enable the user to employ the VIMOS 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 downloaded from the archive using the CalSelector tool<sup>1</sup> (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 VIMOS Integral Field Unit observations only via the VIMOS/IFU workflow. The user is referred to the VIMOS web page (<http://www.eso.org/sci/facilities/paranal/instruments/vimos/>) for more information on the instrument itself, and the VIMOS pipeline user manual for the details of the pipeline recipes (<http://www.eso.org/sci/software/pipelines/>).

The workflow uses association rules known to work with files downloaded from the ESO archive with the CalSelector tool (from year 2009 onwards). For older datasets where the data were directly delivered to the PI (e.g. in DVDs) see 8.

---

<sup>1</sup><http://www.eso.org/sci/archive/calselectorInfo.html>

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	7 of 30

## 2 Software Installation

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

The recommended way is to use the package repositories if your operating system is supported. The `macports` repositories support OS X, while the `rpm/yum` repositories support Fedora 20 to 25. For any other operating system it is recommended to use the `install_esoreflex` script.

### 2.1 Installing Reflex workflows via `macports`

This method is supported for the OS X operating system. It is assumed that `macports` (<http://www.macports.org>) and `java` are installed. If you have any problem with this installation method, please read the full documentation at <http://www.eso.org/sci/software/pipelines/installation/macports.html>.

For a quick installation, the following steps will install the ESO pipeline `macports` repository, the VIMOS pipeline, including the Reflex workflow support and Reflex itself:

- Set up the repository:

```
# curl ftp://ftp.eso.org/pub/dfs/pipelines/repositories/macports/setup/Portfile -o Portfile
# sudo port install
# sudo port sync
```

- Install the VIMOS pipeline:

```
# sudo port install esopipe-vimos-all
```

### 2.2 Installing Reflex workflows via `rpm/yum`

This method is supported for Fedora 20/21/22/23/24/25 operating systems. If you have any problem with this installation method, please read the full documentation at <http://www.eso.org/sci/software/pipelines/installation/rpm.html>.

For a quick installation, the following steps will install the ESO pipeline `rpm` repository, the VIMOS pipeline, including the Reflex workflow support and Reflex itself:

- Set up the repository for Fedora 20/21:

```
# sudo yum install yum-utils
# sudo yum-config-manager \
  --add-repo=ftp://ftp.eso.org/pub/dfs/pipelines/repositories/fedora/esorepo.repo
```

- Set up the repository for Fedora 22/23/24/25:

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

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	8 of 30

- Install the VIMOS pipeline (Fedora 20/21):  
# `sudo yum install esopipe-vimos-all`
- Install the VIMOS pipeline (Fedora 22/23/24/25):  
# `sudo dnf install esopipe-vimos-all`

### 2.3 Installing Reflex workflows via `install_esoreflex`

The software pre-requisites for Reflex 2.9.0 may be found at:  
[http://www.eso.org/sci/software/pipelines/reflex\\_workflows](http://www.eso.org/sci/software/pipelines/reflex_workflows)

To install the Reflex 2.9.0 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_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. You will be asked whether you want to use your Internet connection. Unless you want to reuse already downloaded packages (only advanced users), use the default Yes.
5. You will be given a choice of pipelines (with the corresponding workflows) to install. Please specify the numbers for the pipelines you require, separated by a space, or type “A” for all pipelines.
6. For the pipelines to be installed you will be prompted for the demo data sets to be installed. Type “A” for all demo datasets. Take into account that if you are installing in a directory that already contains data, it won’t be removed.
7. The script will also detect whether previous versions of the workflows or Reflex were installed and in this case you have the option to update links or remove obsolete cache directories. It is advised to use the defaults.
8. If some of the prerequisite binaries for Reflex are not under one of the paths indicated by the command,



<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	9 of 30

```
getconf PATH
```

then you will need to add the appropriate paths as a colon separated list to the `esoreflex.path` parameter in the configuration file `<install_dir>/etc/esoreflex.rc`. This will usually be necessary when the FITS viewer (`fv`) is installed outside of `/usr/bin`. As an example, assume `fv` is installed into the directory `/usr/local/fv5.4`, the file `esoreflex.rc` should then have the line setting `esoreflex.path` look similar to the following:

```
esoreflex.path=/usr/local/fv5.4
```

In the case of OS X `/Applications/fv.app/Contents/MacOS/` is the typically installation directory. Thus, this should be similar to the following line instead:

```
esoreflex.path=/opt/local/bin:/Applications/fv.app/Contents/MacOS
```

9. 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.

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	10 of 30

### 3 Demo Data

Together with the pipeline you will also receive a demo data set, that allows you to run the `Reflex VIMOS` 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. The demo data for VIMOS includes both data for the IFU and MOS workflows, but a given workflow will only use the data for that mode.

Note that you will need a minimum of  $\sim 1.3$  GB,  $\sim 0.6$  GB and  $\sim 7.0$  GB of free disk space for the directories `<download_dir>`, `<install_dir>` and `<data_dir>`, respectively. The VIMOS demo data have been retrieved with the CalSelector tool<sup>2</sup>.

---

<sup>2</sup><http://www.eso.org/sci/archive/calselectorInfo.html>

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	11 of 30

## 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 VIMOS demo data set supplied with the Reflex 2.9.0 release. By following these steps, the user should have enough information to perform a reduction of his/her own data without any further reading:

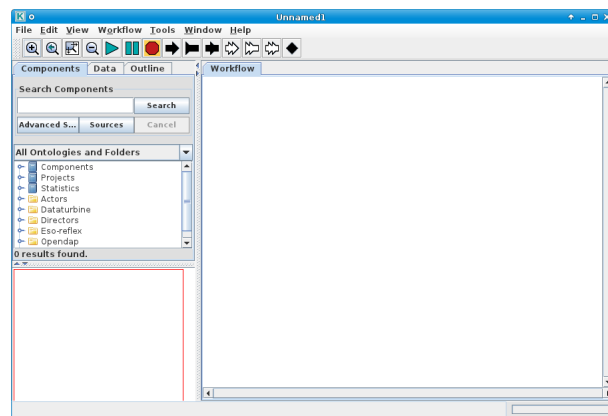


Figure 4.1: *The empty Reflex canvas.*

1. Start the Reflex application:

```
esoreflex &
```

If `install_esoreflex` was used or manual installation was performed then the start command is:


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

The empty Reflex canvas as shown in Figure 4.1 will appear.

2. Now open the VIMOS workflow by clicking on File -> Open File, selecting first `vimos-3.2.3` and then the file `vimos_ifu.xml` in the file browser. You will be presented with the workflow canvas shown in Figure 4.2. 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 (100 ms is recommended), and click .
4. Under "Setup Directories" in the workflow canvas there are seven parameters that specify important directories (green dots). Changing the value of `ROOT_DATA_DIR` and/or `RAW_DATA_DIR` is the only

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	12 of 30

necessary modification if you want to process data other than the demo data<sup>3</sup>, 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.

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 `Reflex` 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 4.3) that lists the `DataSets` along with the values of a selection of useful header keywords<sup>4</sup>. 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.
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 4.4 shows the *Product Explorer* window.
10. After the workflow has finished, all the products from all the `DataSets` can be found in a directory under `END_PRODUCTS_DIR` with the named with 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 `VIMOS` workflow that merit a look at the rest of this tutorial.

<sup>3</sup>If you used the install script `install_esoreflex`, then the value of the parameter `ROOT_DATA_DIR` will already be set correctly to the directory where the demo data was downloaded.

<sup>4</sup>The keywords listed can be changed by right-clicking on the `DataOrganiser` Actor, selecting `Configure Actor`, and then changing the list of keywords in the second line of the pop-up window.

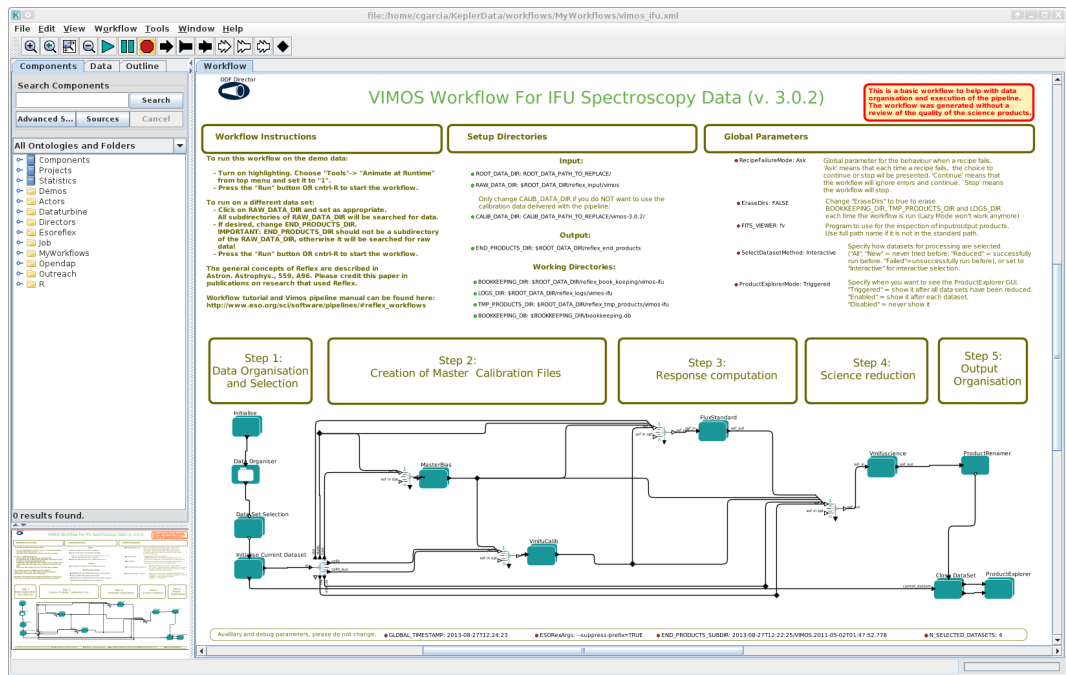


Figure 4.2: *VIMOS/IFU* workflow general layout.

Select Datasets (on gao146999)

Selected	Data Set	#Files	Reduced	Obs. Target Name	Obs. ID	Obs. Name	Obs. Coord. Quad	Obs. Mode
<input checked="" type="checkbox"/>	VIMOS_2011-05-02T01:47:52.776	201	NO	NOX-3593	1540382	IFU_nrg_2593_psm0a_1.2	IFU	
<input checked="" type="checkbox"/>	VIMOS_2011-05-02T01:47:52.776	201	NO	NOX-3593	1540382	IFU_nrg_2593_psm0a_1.3	IFU	
<input checked="" type="checkbox"/>	VIMOS_2011-05-02T01:47:52.776	201	NO	NOX-3593	1540382	IFU_nrg_2593_psm0a_1.1	IFU	
<input checked="" type="checkbox"/>	VIMOS_2011-05-02T01:47:52.776	201	NO	NOX-3593	1540382	IFU_nrg_2593_psm0a_1.4	IFU	

Buttons: Save all, Invert highlighted, Select complete, Deselect all, Filter: New, Continue, Stop

Figure 4.3: The “Select Datasets” pop-up window.

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	14 of 30

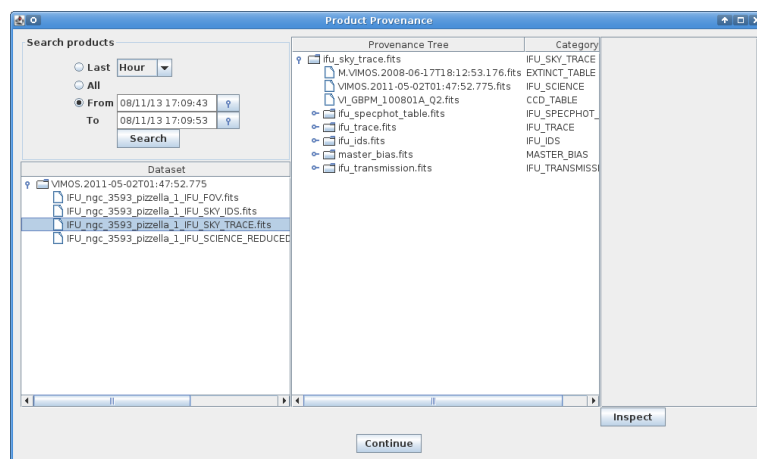


Figure 4.4: The Product Explorer shows all datasets reduced in previous executions together with the full reduction chain for all the pipeline products.

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	15 of 30








## 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`. Saving the workflow in the default Kepler format (.kar) is only advised if you do not plan to use the workflow with another computer.








### 5.2 Buttons

At the top of the Reflex 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.

### 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. After pausing, the workflow may be resumed by clicking the  button again.

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	16 of 30

## 6 The VIMOS Workflow

The VIMOS 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.

### 6.1 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.

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 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 re-reduce the data each time the workflow is run.

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

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





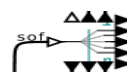
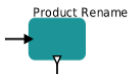

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	17 of 30

that the `ProductExplorer` actor will be shown only at the end of the workflow execution.

## 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 an icon within the rectangle to aid in their identification. The following actors are simple actors:

- 
 - The Data Organiser actor.
- 
 - The Data Set Chooser actor (inside a composite actor).
- 
 - The Fits Router actor
- 
 - The Product Renamer actor.
- 
 - The Product Explorer 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).

### 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.

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 `RecipeExecutor` 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 Executor`.

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	18 of 30

### 6.2.3 Recipe Execution within Composite Actors

The VIMOS workflow contains Composite Actors to run pipeline recipes. This is in the most simple case due to the `SoF Splitter/SoF Accumulator`<sup>5</sup>, which allow to process calibration data from different setting within one given `DataSet` (e.g. lamp frames taken with different slits/masks). More complex Composite Actors contain several actors (e.g. `Recipe Executer`).

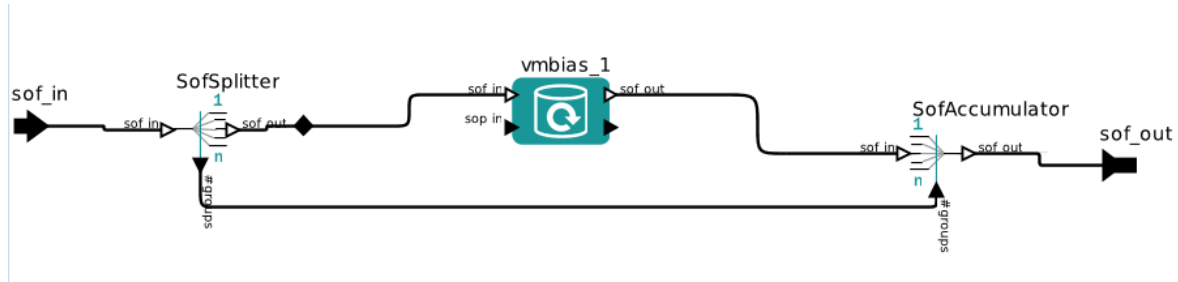


Figure 6.1: This is the window you get when you choose `Open Actor` for the Composite Actor `MasterBias`. This is the most simple case for a Composite Actor. Using `Configure Actor` on `vimos_bias_1` gives you Fig. 6.2.

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 6.1, which is the most simple version of a Composite Actor. The `RecipeExecuter` is preceded by an `SoF Splitter`, and followed by an `SoF Accumulator`. The function of the `SoF Splitter` is to investigate the incoming SoFs, sort them by “purpose”, and create separate SoFs for each purpose. The `RecipeExecuter` then processes each of the SoFs independently (unless they are actually the same files). Finally, the `SoF Accumulator` packs all the results into a single output SoF. The direct relation between the `SoF Splitter` and `SoF Accumulator` is used to communicate the number of different SoFs created by the `SoF Splitter`. A workflow will only work as intended if the purpose of all the files a recipe needs as input is identical. The only exception to this rule is that a purpose can also be “default”. In this case, the file is included in any output SoF created by the `SoF Splitter` and `SoF Accumulator`.

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

A `RecipeExecuter` actor is used in the workflow to run a single VIMOS pipeline recipe (e.g: in the `MasterBias` actor the recipe `vmbias` is executed). In order to configure the `RecipeExecuters`, one has to first use `Open Actor` to get to the level of the recipe executers (see Fig. 6.1).

<sup>5</sup>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).

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	19 of 30

**Edit parameters for vmbias\_1**

recipe:	vmbias
mode:	Run
Lazy Mode:	<input checked="" type="checkbox"/>
Recipe Failure Mode:	\$RecipeFailureMode
Input Files Category:	
Output Files Category:	
File Purpose Processing:	Strip last
Allow empty inputs:	<input type="checkbox"/>
Pause before execution:	<input type="checkbox"/>
Pause after execution:	<input type="checkbox"/>
Clear Products Dir:	Never
Clear Logs Dir:	Never
Clear Bookkeeping Dir:	Never
Products Dir:	\$TMP_PRODUCTS_DIR <span>Browse</span>
Logs Dir:	\$LOGS_DIR <span>Browse</span>
Bookkeeping Dir:	\$BOOKKEEPING_DIR <span>Browse</span>
EsoRex default args:	\$ESORexArgs
recipe_param_1:	AllowSingleFrames=TRUE
recipe_param_2:	StackMethod=Median
recipe_param_3:	KSigmaLow=5.0
recipe_param_4:	KSigmaHigh=5.0
recipe_param_5:	MinRejection=1
recipe_param_6:	MaxRejection=1
recipe_param_7:	RemoveOverscan=TRUE
recipe_param_8:	CleanBadPixel=FALSE
recipe_param_9:	CleanCosmic=FALSE
recipe_param_10:	ComputeQC=TRUE
Reuse Inputs (Expert Mode):	<input type="checkbox"/>
Reuse Outputs (Expert Mode):	<input type="checkbox"/>

Buttons: Commit, Add, Remove, Restore Defaults, Preferences, Help, Cancel

Figure 6.2: The “Edit parameters” window for a typical RecipeExecuter actor, the vmbias\_1 actor which runs the vmbias pipeline recipe.

In Figure 6.2 we show the “Edit parameters” window for a typical RecipeExecuter actor, which can be displayed by right-clicking on the actor and selecting Configure Actor. In the following we describe in more detail the function of some of the parameters for a RecipeExecuter actor:

- The “recipe” parameter states the VIMOS 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 Sect. 6.2.4.

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	20 of 30

Table 6.1: The VIMOS/IFU pipeline actors and their contents

actor	recipes	description
MasterBias	vmbias	create master bias
VmIfuCalib	vmifucalib	create master flat, determine coefficients for wave-length calibration and correction of spatial distortion
FluxStandard	vmifustandard	determine response function
VmIfuscience	vmifuscience	reduce science data

- The “Recipe Failure Mode” parameter has a pull-down menu allowing the user to specify the behaviour of the actor if the pipeline recipe fails. The available options are:
  - Stop: The actor issues an error message and the workflow stops.
  - Continue: The actor creates an empty output and the workflow continues.
  - Ask: The actor displays a pop-up window and asks the user whether he/she wants to continue or stop the workflow. This option is the default option.
- The set of parameters which start with “recipe param” and end with a number or a string correspond to the parameters of the relevant VIMOS pipeline recipe. By default in the `RecipeExecutor` 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 VIMOS pipeline recipe parameters, the user should refer to the VIMOS pipeline user manual (Izzo et al. 2012<sup>6</sup>).

The description of the remainder of the `RecipeExecutor` 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 must be saved in the workflow by clicking the `Commit` button when you have finished to take effect. If you want to reuse the parameters you have to save the workflow with the saved parameters.

#### 6.2.4 Lazy Mode

By default, all recipe executor 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.

<sup>6</sup> Available at <ftp://ftp.eso.org/pub/dfs/pipelines/vimos/vimos-pipeline-manual-7.2.pdf>


ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	21 of 30

The re-reduction of data at each execution may sometimes be desirable. To force a re-reduction of data for any single `RecipeExecutor` 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 `RecipeExecutor` actors are actually found inside the composite actors in the top level workflow. To access such embedded `RecipeExecutor` 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 `RecipeExecutor` 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.

## 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`<sup>7</sup> 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 `IFU_SCREEN_FLAT`, `IFU_ARC_SPECTRUM` or `IFU_SCIENCE`. 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 `BIAS` and

<sup>7</sup>`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`

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	22 of 30

IFU\_CAL. The former creates a master bias from raw biases, and the later creates (among other products) a master flat from raw flats. The IFU\_CAL action needs raw lamp frames (arc and flat field) and the master bias (or a set of raw biases) as input. In this case, these biases will have the purpose BIAS . IFU\_CAL. The same DataSet might also include biases with a different purpose, e.g. BIAS . IFU\_SCIENCE. Irrespective of their purpose the file category for all these biases will be BIAS.

- Next the DataSet Chooser displays the DataSets available in the “Select Data Sets” window<sup>8</sup>, activating a vertical scroll bar on the right if necessary (see Figure 4.3). 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) 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). 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), 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 RecipeExecutor (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 to generate input SoFs for the RecipeExecutor and the results are collected by the SofAccumulator. 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: Recipe execution

Once the datasets to be reduced are selected, press the Continue button on the dataset organised window to proceed with the data reduction. The workflow will automatically execute the pipeline recipes and construct the .sof files to feed the pipeline recipes with. Each sof file will be saved in the BOOKKEEPING\_DIR directory

<sup>8</sup>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.

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	23 of 30

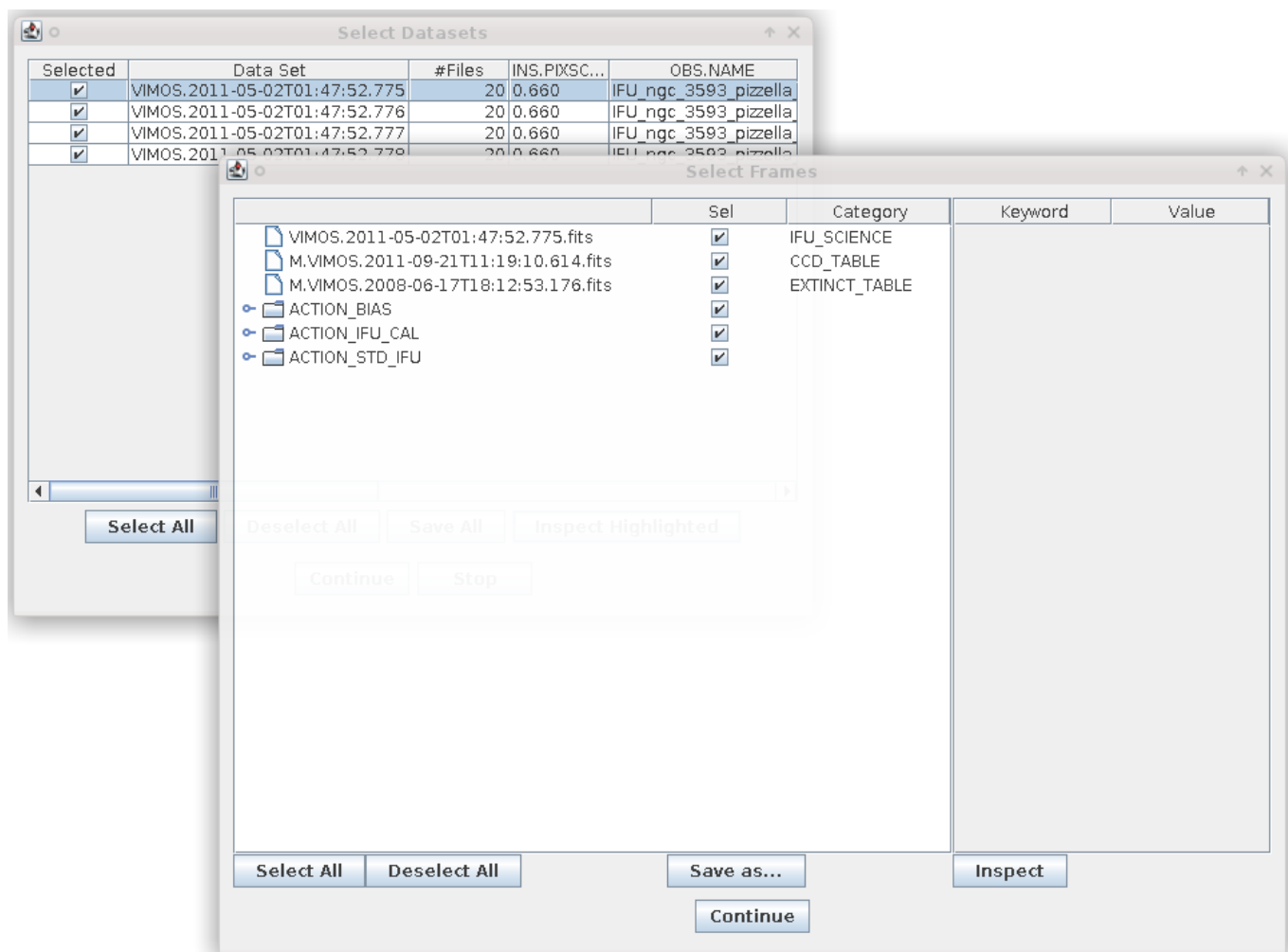


Figure 6.3: The “Select Frames” window with a single file from the current Data Set highlighted in blue, and the corresponding FITS header displayed in the text box on the right. Hidden partially behind the “Select Frames” window is the “Select DataSets” window with the currently selected DataSet highlighted in blue.

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	24 of 30

(and subdirectory within it), depending on the recipe it is associated to and the execution time. The pipeline parameters can be changed as shown in figure [6.2](#).

### 6.3.3 Step 3: Final products

Once a dataset is reduced (i.e. when the `vmifuscience` recipe is terminated), a window containing the list of science product is popped out. Each file can be inspected with the selected fits viewer. Final science products will be stored in the `END_PRODUCTS_DIR`, and sorted by execution time and dataset identifier (i.e. the name of the science frame the dataset is for). Default names for the science products are: `IFU_<OB name>_IFU_FOV.fits`, `IFU_<OB name>_IFU_SCIENCE_FLUX_REDUCED.fits`, `IFU_<OB name>_IFU_SCIENCE_REDUCED.fits`, `IFU_<OB name>_IFU_SKY_IDS.fits`, and `IFU_<OB name>_IFU_SKY_TRACE.fits`. We refer the user to the VIMOS pipeline manual for the description of these files.



ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	25 of 30

## 7 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 (Forchi (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:

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	26 of 30

```
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 executor. This actor must be manually instantiated using `Tools -> Instantiate Component`. Fill in the “Class name” field with `org.eso.RecipeExecutor` 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 (Forchi (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 (Forchi (2012)) for more information.
- **How can I manually run the recipes executed by Reflex?** If a user wants to re-run a recipe on the command line he/she has to go to the appropriate `reflex_book_keeping` directory, which is generally `reflex_book_keeping/<workflow>/<recipe_name>_<number>`. There, subdirectories exist with the time stamp of the recipe execution (e.g. 2013-01-25T12:33:53.926/). If the user wants to re-execute the most recent processing he/she should go to the `latest` directory and then execute the script `cmdline.sh`. Alternatively, to use a customized `esorex` command the user can execute

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

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

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

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

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

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

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	27 of 30

## 8 Troubleshooting

In this section we describe some of the problems that may occur when reducing the VIMOS/IFU with the ESOrex pipeline. For a more comprehensive description we refer the user to the VIMOS user manual (<http://www.eso.org/sci/software/pipelines/>).

### 1. I have data from the old PI packs

For older datasets where the data were directly delivered to the PI (e.g. in DVDs) different set of association rules must be used. These are no longer supported, but we provide a compatible set of rules. To use them, right click on the DataOrganiser actor and select Configure Actor. Then in the OCA file configuration, choose file `vimos_ifu_wkf.dvd.oca`, which is in the same directory as the default `vimos_ifu_wkf.oca` one. Moreover, the data from the DVD has to be cleaned up:

- remove *all* the pipeline products i.e. master bias (whose filename contains the string MBIA), transmission response files (whose filename contains the string PTNF) and directories labelled as `proc` or `reduced`
- remove duplicate files (i.e. files with the same name, but stored in different directories, such as arc lamps).

### 2. Should I change the CALIB\_DATA\_DIR configuration?

This directory is setup automatically to point to the calibration database provided with the pipeline and in principle shouldn't be changed. However, if static calibration data are present in the RAWDATA\_DIR (e.g. calibrations are downloaded from the archive, or copied from ESO-DVD distribution), then you might have to set this directory equal to RAWDATA\_DIR (otherwise an obsolete static calibration file may be selected instead of the most appropriate one).

### 3. Fibre misidentification

The recipe `vmifucalib` uses the IFU\_IDENT calibration file <sup>9</sup> to identify the fibre in the flat field. If a fibre identification file is not specified, the fibre spectra identification is still attempted, but the result is not always correct.

The IFU fibre identification performed by recipe `vmifucalib` appears to be negatively affected by changes in temperature. If in the recipe products more than about 50 fibres appear to be "lost" in one pseudo-slit, it may help to rerun the recipe using the `blind` fibre identification method: this method is always triggered if no fibre identification table is specified in the input set-of-frames.

There are 2 main ways to recognise fibre misidentification problems.

- A defined set of fibres has intentionally null transmission in each quadrant. This help the cross-correlation in the fibre identification process. These dead fibres are 20-21, 60-61, 100-101 and so on, i.e. two fibres each 40. By inspecting the 2D reduced spectrum `SCIENCE_FLUX_REDUCED.fits` it is possible to see whether the "dead fibres" are placed in the correct positions (see Figure 8.2).

---

<sup>9</sup>IFU\_IDENT consists of intensity profiles (one for each IFU pseudo-slit) cut along the cross-dispersion direction of a reference flat field exposure where the fibre spectra have been safely identified. The fibres corresponding to the peak positions of each profile are listed in the IFU\_IDENT file. Such safe identifications would then be transferred to the new input flat fields by cross-correlation. In the calibration directories there is ideally one IFU\_IDENT file for each quadrant/grism combination, named `ifu_ident_grism_q.fits` (where `q` indicates the VIMOS quadrant number, and `grism` the grism name).

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	28 of 30

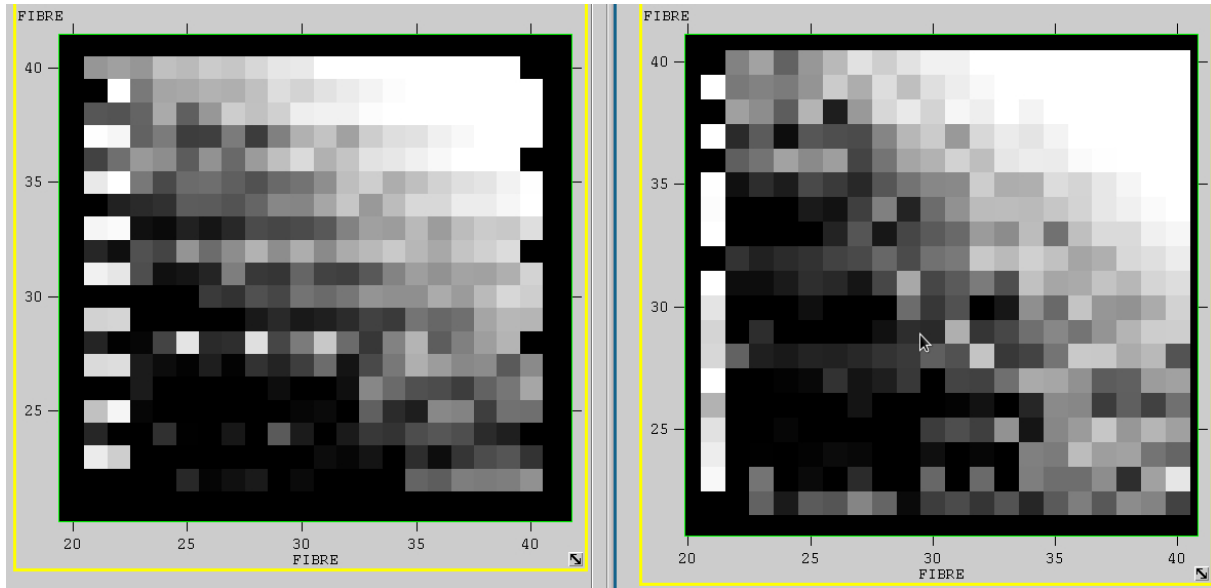


Figure 8.1: Example of field of view where fibre misidentification shuffled horizontally the fibre position (left panel), compared to one where fibre identification worked properly (right panel). The field of view refers only to quadrant 3.

- A fibre misidentification would appear later on the reconstructed image of the field-of view (generated by the `vmifuscience` recipe) as zig-zagging patterns breaking the generally smooth look of the intensity distribution. The field of view file `FOV` can be inspected each time the workflow has reduced a dataset setting mode to `INSPECT` to the `DataFilter2` actor (right-click with the mouse on the actor, and select `configure` actor).

An example of the effect that fibre misidentification has on the field-of-view is shown in Figure 8.1.

If a fibre misidentification occurs, and the `blind` method in `vmifucal` does not work (see point n° 2 above), one solution could be to manually shuffle the fibre of 1 position, according to the fibre coordinates specified in the `IFU_TABLES`<sup>10</sup>. For example with reference to quadrant number 3, if the fibre n° 290 at pixel coordinates (30,35) on the field of view should be placed in at the pixel coordinates (31,35) instead, because of a fibre mismatch problem, it need to be re-identified as fibre number 291, and so on. The inspection of the re-shuffled FOV helps in understanding if the correction was done in a proper way.

Unfortunately, this shuffle correction is not supported in the VIMOS/IFU reflex workflow, and must be operated by the user on the reduced files on a case-by-case basis.

#### 4. Optimising the number of recovered fibres.

The fibre identification process operated in `vmifucal` select useful fib-res on the basis of a trace rejection algorithm. The parameter that regulates this procedure is `MaxTraceRejection`. `MaxTraceRejection` sets the maximum percentage of rejected positions in fibre spectra tracing (default = 50). In the fibre tracing operation, a number of pixel positions may be rejected because the detected position outlays the

<sup>10</sup>The `IFU_TABLES` are static calibration files that contain the correspondence between fibre number in the 2D spectrum and the pixel coordinate on the field of view. See Section 6.21 of the VIMOS manual (version 6.5) for further details.

ESO	Reflex VIMOS/IFU Tutorial	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	29 of 30

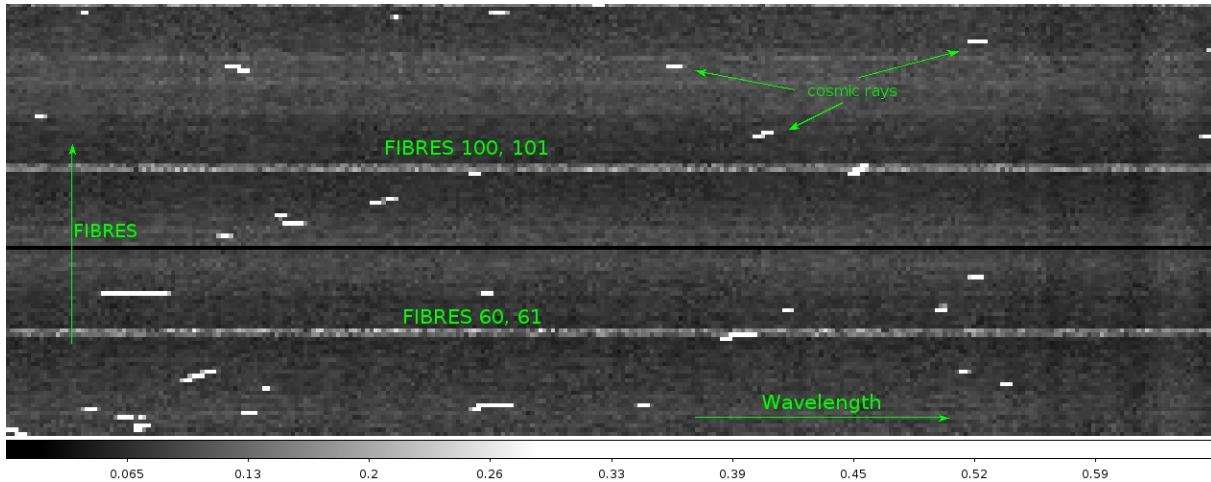


Figure 8.2: Example of 2D spectra where the marked fibres (60-61, and 100-101) are properly identified.

general trend, or because the signal level is too low. When the percentage of rejected positions is more than what is specified here, then the corresponding fibre is flagged as *dead* and excluded from further processing. This parameter can be modified to recover the majority of useful fibres. For example, for the HR grism the range 10– 80 gives good results. The expected number of good fibres is about 380 (quadrants 1 and 3), 310 (quadrant 2) and 360 (quadrant 4).

The `MaxTraceRejection` can be optimised in the VIMOS IFU Reflex workflow by entering the `VmIfuCalib` subworkflow (right click and *Open Actor*) and double clicking on the recipe actor `vmifucalib_1`, similar to the figure 6.2.

## 5. Bug: MODE mismatch between BIAS frames and pixel tables

It can happen that the BIAS frames associated to the reduction flow of one IFU observations was taken in the imaging mode. The bias can be used, as there is no difference between bias taken in imaging or IFU modes. Nevertheless, the `vmbias` recipe requires the bias frame and the bad pixel table<sup>11</sup> to have the same observing mode. This bug can be overcome by changing the `OBSMODE` in the bias frames from `IMG` to `IFU`. This bug is in the `vmbias` recipe, independent from the reflex workflow, and it may be solved in future VIMOS pipeline releases.

<sup>11</sup>`CCD_TABLES` are static calibration files, one per quadrant, one set per observing mode, containing the list of bad pixels. See Section 6.2 in the VIMOS pipeline user manual (version 6.5).

<b>ESO</b>	<b>Reflex VIMOS/IFU Tutorial</b>	Doc:	VLT-MAN-ESO-19540-5898
		Issue:	Issue 2.4
		Date:	Date April 2018
		Page:	30 of 30

Forchì V., 2012, Reflex User Manual, VLT-MAN-ESO-19000-5037, Issue 0.7,  
<ftp://ftp.eso.org/pub/dfs/reflex/ReflexUserManual-3.1.pdf> 16, 20, 25, 26

ESO VIMOS Pipeline Team, VIMOS Pipeline User Manual, VLT-MAN-ESO-19500-3355, Issue 6.6 20