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-IMG Tutorial

VLT-MAN-ESO-19500-XXXX

Issue 4.1.14

Date March 24th 2026

Prepared: ESO VIMOS Pipeline Team March 24th 2026
Name Date Signature

Approved: W.Freudling
Name Date Signature

Released: M. Sterzik
Name Date Signature

This page was intentionally left blank

Authors

Name	Affiliation
M. N	ESO
G. M.	CASU

This page was intentionally left blank

Change Record from previous Version

Affected Section(s)	Changes/Reason/Remarks
None	Version number aligned to pipeline

This page was intentionally left blank

Contents

1 Introduction to Esoreflex

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

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

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

A workflow accepts science and calibration data, as downloaded from the archive using the CalSelector tool[1] (with associated raw calibrations) and organises them into DataSets, where each DataSet contains one science object observation (possibly consisting of several science files) and all associated raw and static calibrations required for a successful data reduction. The data organisation process is fully automatic, which is a major time-saving feature provided by the software. The DataSets selected by the user for reduction are fed to the workflow which executes the relevant pipeline recipes (or stages) in the correct order. Full control of the various recipe parameters is available within the workflow, and the workflow deals automatically with optional recipe inputs via built-in conditional branches. Additionally, the workflow stores the reduced final data products in a logically organised directory structure employing user-configurable file names.

The VIMOS-IMG workflow provides optional user interactivity for most steps of the data reduction process. This interactivity enables a user to assess the pipeline products and/or re-run a recipe with different parameters before moving to the next step. Full control of the all 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 filenames.

The VIMOS-IMG Reflex workflow described in this tutorial supports the reduction of VIMOS data taken **only** in imaging mode (IMG), including pre-imaging. The user is referred to the VIMOS User Manual[2] for more more information about the instrument and the pipeline.

By default, the workflow will group data together by the start time of the **template** in which it was taken (TPL.START), **not** by Observation Block. This means that the workflow cannot be used to combine or stack images that were taken as part of templates with different start times. If a user wishes to run pipeline recipes on data that have different template start times (e.g. all files taken as part of one OB), they may either use the command line interface "esorex", modify the workflow, or modify the TPL.START keywords such that they are the same for all images that are to processed together. Note that as discussed in the VIMOS-IMG Pipeline manual [2], the pipeline is not designed to process science data or standard star fields that span more than one OB.

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

[2] *http://www.eso.org/sci/facilities/paranal/instruments/vimos/doc.html*

The VIMOS-IMG pipeline can provide high quality data products using the default parameters for most recipes. These include calibrated sky and science images, mosaics, and catalogues of objects in the images.

The quick start section (§5) describes the minimum effort to get started and reduce the demonstration data.

2 System Requirements

The data reduction for VIMOS images may require substantial memory resources. However, the amount of memory required to run a recipe depends on a number of factors, e.g. number of files, stacking method, sky coverage, maximum jitter offset, etc. Below is a table with the minimum resident memory needed to process data as a function of recipe and number of input files. Note that the number of input files here is the number of raw files, i.e. divide this by 4 to get the number of coherent sets of VIMOS detectors. This table assumes that all recipe parameters are set to their default and a small jitter offset between frames. In general, the execution time of a recipe can be shortened by choosing 'fast' stacking, but this will increase the memory use (see description of `stk_fast` and `stk_nfst` parameters in [2])

Table 2.1: Minimum memory requirements for selected VIMOS-IMG recipes

Recipe	N science frames	Mininum RAM
vimos_standard_process	4	1.8 GB
vimos_science_process	12	6.2 GB
vimos_science_process	32	9.2 GB

3 Software Installation

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

The recommended way is to use the package repositories if your operating system is supported. The pipelines and Reflex can be installed from the ESO macports repositories that support macOS platforms, the and the rpm/yum repositories that support Fedora and CentOS platforms. For any other operating system it is recommended to use the install_esoreflex script.

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

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

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

3.1 Installing Esoreflex workflows via macports

This method is supported for the macOS operating system. It is assumed that macports (*https://www.macports.org*) is installed. Please read the full documentation at *https://www.eso.org/sci/software/pipelines/installation/macports.html*, which also describes the versions of macOS that are currently supported.

3.2 Installing Esoreflex workflows via rpm/yum/dnf

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

3.3 Installing Esoreflex workflows via install_esoreflex

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

The software pre-requisites for Reflex 2.11.5 may be found at: *https://www.eso.org/sci/software/pipelines/reflex_workflows*

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

1. From any directory, download the installation script:

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

2. Make the installation script executable:

   ```
   chmod u+x install_esoreflex
   ```

3. Execute the installation script:

   ```
   ./install_esoreflex
   ```

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

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

5. To start Reflex, issue the command:

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

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

4 Demo Data

The VIMOS-IMG pipeline kit comes with a set of demonstration data. This data is intended to be used as a means to become familiar with the pipeline and workflow. They were selected to show the typical steps to reduce most VIMOS-IMG data sets. Data taken with an uncommon or complex observing strategy may not be suitable for a straight-forward reduction with the Reflex workflow. In this case, a deeper understanding of the pipeline may be required.

The full collection of demo data comprise raw data, static calibration data, reference data, and photometric and astrometric catalogues. A single, complete data set can be formed from the demo data: one Stetson standard star field (‘SA92’) and one set of science observations toward the a rich cluster of galaxies (‘SPT0546’) at a redshift of z=1.067 ([1]).

This data set can be used to create a master bias, a master dark, a master twilight flat, (optionally) a table of the detector readnoise and gain, a standard star field, and finally, a co-added science exposure.

The VIMOS-IMG pipeline can use a number of optional files to calibrate the astrometry and photometry of processed images. **These files are only necessary if a user wishes to process data without an internet connection.** The default assumes that the user has an internet connection. In this case, the VIMOS-IMG recipes can retrieve the required data automatically through the Strasbourg astronomical Data Center[3] (CDS). It is recommended that VIMOS-IMG data use the AAVSO Photometric All-Sky Survey (APASS) catalogue [3] for photometric calibration of the standard star field, and the PPXML [5] catalogue for astrometric calibration. Due to the very large size of the astrometric and photometric catalogues, the preferred method is via the CDS and an internet connection. Here, only the catalogue sources in the area of sky covered by the data set will be downloaded. A description of how to download the full catalogue data, when processing without an internet connection cannot be avoided, is given in the FAQ’s of §10.

A detailed description of the various VIMOS imager data types is given in the Appendix (§A)

[3] *http://cdsweb.u-strasbg.fr/*

5 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 `esoreflex 2.11.5` release. By following these steps, the user should have enough information to perform a reduction of his/her own data without any further reading:

1. First, type:

   ```
   esoreflex -l
   ```

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

2. Open the VIMOS Imaging by typing:

   ```
   esoreflex vimos_ima&
   ```

   Alternatively, you can type only the command `esoreflex` the empty canvas will appear (Figure 5.1) and you can select the workflow to open by clicking on `File -> Open File`. Note that the loaded workflow will appear in a new window. The VIMOS Imaging workflow is shown in Figure 5.2.

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

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

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

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

5. Click the ▶ button to start the workflow

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

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

7. The Data Set Chooser actor will be highlighted next and will display a "Select Datasets" window (see Figure 5.3) that lists the datasets along with the values of a selection of useful header keywords[4]. The first column consists of a set of tick boxes which allow the user to select the datasets to be processed. By default all complete datasets which have not yet been reduced will be selected. A full description of the options offered by the Data Set Chooser will be presented in Section 7.3.2.

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

9. Once the reduction of all datasets has finished, a pop-up window called *Product Explorer* will appear, showing the datasets which have been reduced together with the list of final products. This actor allows the user to inspect the final data products, as well as to search and inspect the input data used to create any of the products of the workflow. Figure 5.4 shows the Product Explorer window. A full description of the *Product Explorer* will be presented in Section 7.3.3.

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

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

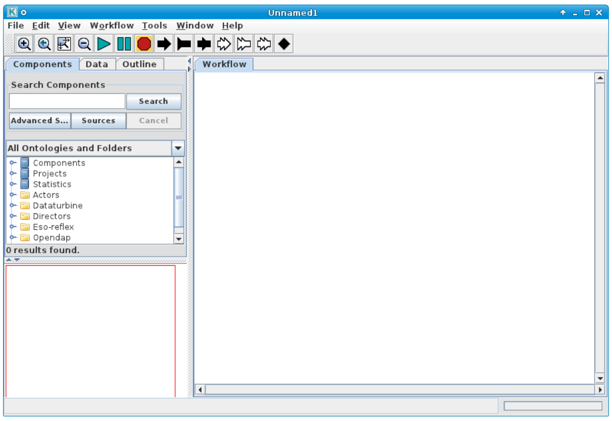
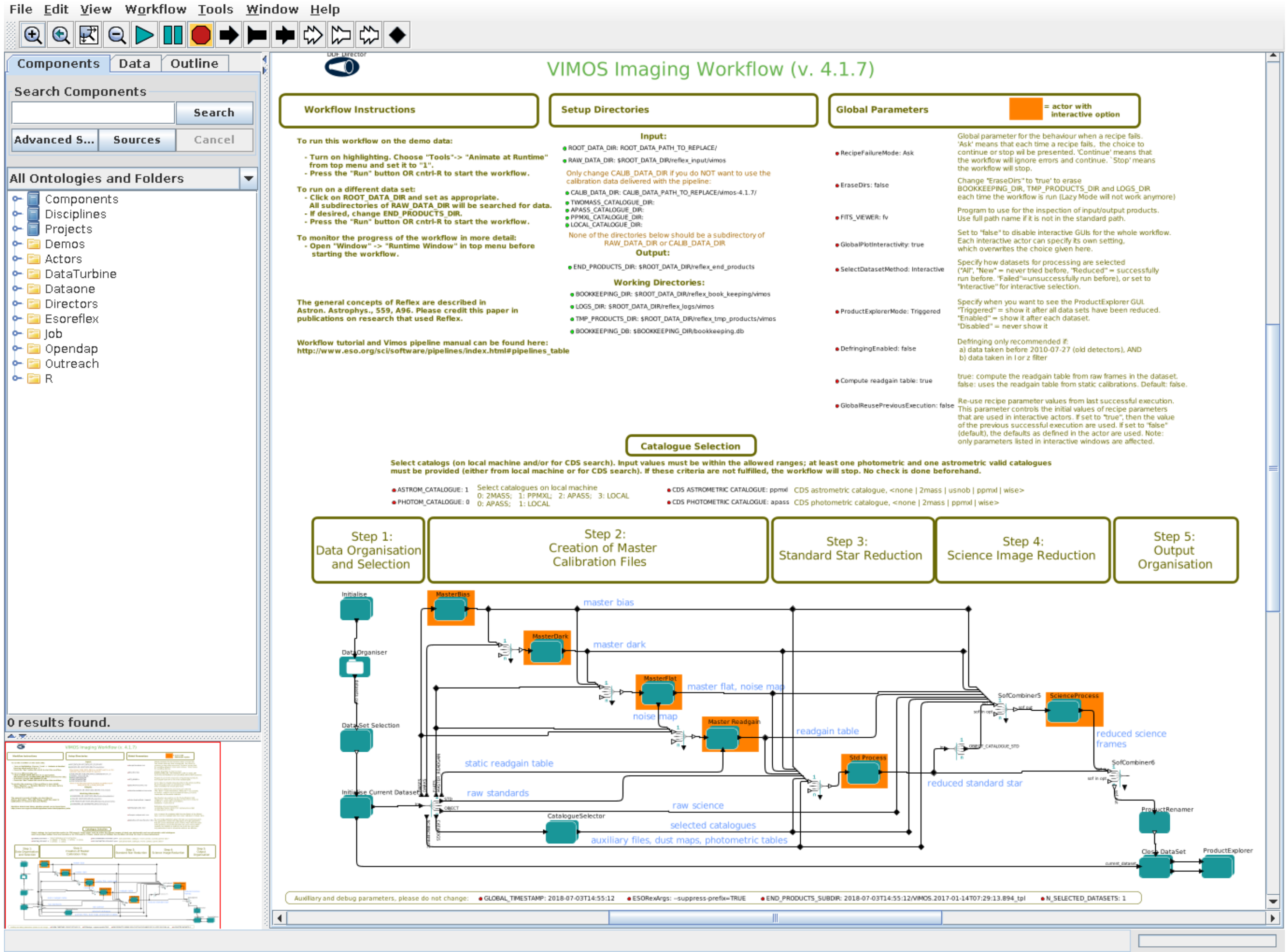


Figure 5.1: *The empty* Reflex *canvas.*

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



VIMOS Imaging Workflow (v. 4.1.7)

Workflow Instructions

To run this workflow on the demo data:

- Turn on highlighting. Choose "Tools"-> "Animate at Runtime" from top menu and set it to "1".
- Press the "Run" button OR cntrl-R to start the workflow.

To run on a different data set:

- Click on ROOT_DATA_DIR and set as appropriate. All subdirectories of RAW_DATA_DIR will be searched for data.
- If desired, change END_PRODUCTS_DIR.
- Press the "Run" button OR cntrl-R to start the workflow.

To monitor the progress of the workflow in more detail:

- Open "Window" -> "Runtime Window" in top menu before starting the workflow.

The general concepts of Reflex are described in Astron. Astrophys., 559, A96. Please credit this paper in publications on research that used Reflex.

Workflow tutorial and Vimos pipeline manual can be found here: http://www.eso.org/sci/software/pipelines/index.html#pipelines_table

Setup Directories

Input:

- ROOT_DATA_DIR: ROOT_DATA_PATH_TO_REPLACE/
- RAW_DATA_DIR: $ROOT_DATA_DIR/reflex_input/vimos

Only change CALIB_DATA_DIR if you do NOT want to use the calibration data delivered with the pipeline:

- CALIB_DATA_DIR: CALIB_DATA_PATH_TO_REPLACE/vimos-4.1.7/
- TWOMASS_CATALOGUE_DIR:
- APASS_CATALOGUE_DIR:
- PPMXL_CATALOGUE_DIR:
- LOCAL_CATALOGUE_DIR:

None of the directories below should be a subdirectory of RAW_DATA_DIR or CALIB_DATA_DIR

Output:

- END_PRODUCTS_DIR: $ROOT_DATA_DIR/reflex_end_products

Working Directories:

- BOOKKEEPING_DIR: $ROOT_DATA_DIR/reflex_book_keeping/vimos
- LOGS_DIR: $ROOT_DATA_DIR/reflex_logs/vimos
- TMP_PRODUCTS_DIR: $ROOT_DATA_DIR/reflex_tmp_products/vimos
- BOOKKEEPING_DB: $BOOKKEEPING_DIR/bookkeeping.db

Global Parameters

🟧 = actor with interactive option

- RecipeFailureMode: Ask
  Global parameter for the behaviour when a recipe fails. 'Ask' means that each time a recipe fails, the choice to continue or stop wil be presented. 'Continue' means that the workflow will ignore errors and continue. `Stop' means the workflow will stop.
- EraseDirs: false
  Change "EraseDirs" to 'true' to erase BOOKKEEPING_DIR, TMP_PRODUCTS_DIR and LOGS_DIR each time the workflow is run (Lazy Mode will not work anymore)
- FITS_VIEWER: fv
  Program to use for the inspection of input/output products. Use full path name if it is not in the standard path.
- GlobalPlotInteractivity: true
  Set to "false" to disable interactive GUIs for the whole workflow. Each interactive actor can specify its own setting, which overwrites the choice given here.
- SelectDatasetMethod: Interactive
  Specify how datasets for processing are selected ("All", "New" = never tried before, "Reduced" = successfully run before. "Failed"=unsuccessfully run before), or set to "Interactive" for interactive selection.
- ProductExplorerMode: Triggered
  Specify when you want to see the ProductExplorer GUI. "Triggered" = show it after all data sets have been reduced. "Enabled" = show it after each dataset. "Disabled" = never show it
- DefringingEnabled: false
  Defringing only recommended if:
  a) data taken before 2010-07-27 (old detectors), AND
  b) data taken in I or z filter
- Compute readgain table: true
  true: compute the readgain table from raw frames in the dataset. false: uses the readgain table from static calibrations. Default: false.
- GlobalReusePreviousExecution: false
  Re-use recipe parameter values from last successful execution. This parameter controls the initial values of recipe parameters that are used in interactive actors. If set to "true", then the value of the previous successful execution are used. If set to "false" (default), the defaults as defined in the actor are used. Note: only parameters listed in interactive windows are affected.

Catalogue Selection

Select catalogs (on local machine and/or for CDS search). Input values must be within the allowed ranges; at least one photometric and one astrometric valid catalogues must be provided (either from local machine or for CDS search). If these criteria are not fulfilled, the workflow will stop. No check is done beforehand.

- ASTROM_CATALOGUE: 1 — Select catalogues on local machine 0: 2MASS; 1: PPMXL; 2: APASS; 3: LOCAL
- PHOTOM_CATALOGUE: 0 — 0: APASS; 1: LOCAL
- CDS ASTROMETRIC CATALOGUE: ppmxl — CDS astrometric catalogue, <none | 2mass | usnob | ppmxl | wise>
- CDS PHOTOMETRIC CATALOGUE: apass — CDS photometric catalogue, <none | 2mass | ppmxl | wise>

Step 1: Data Organisation and Selection

Step 2: Creation of Master Calibration Files

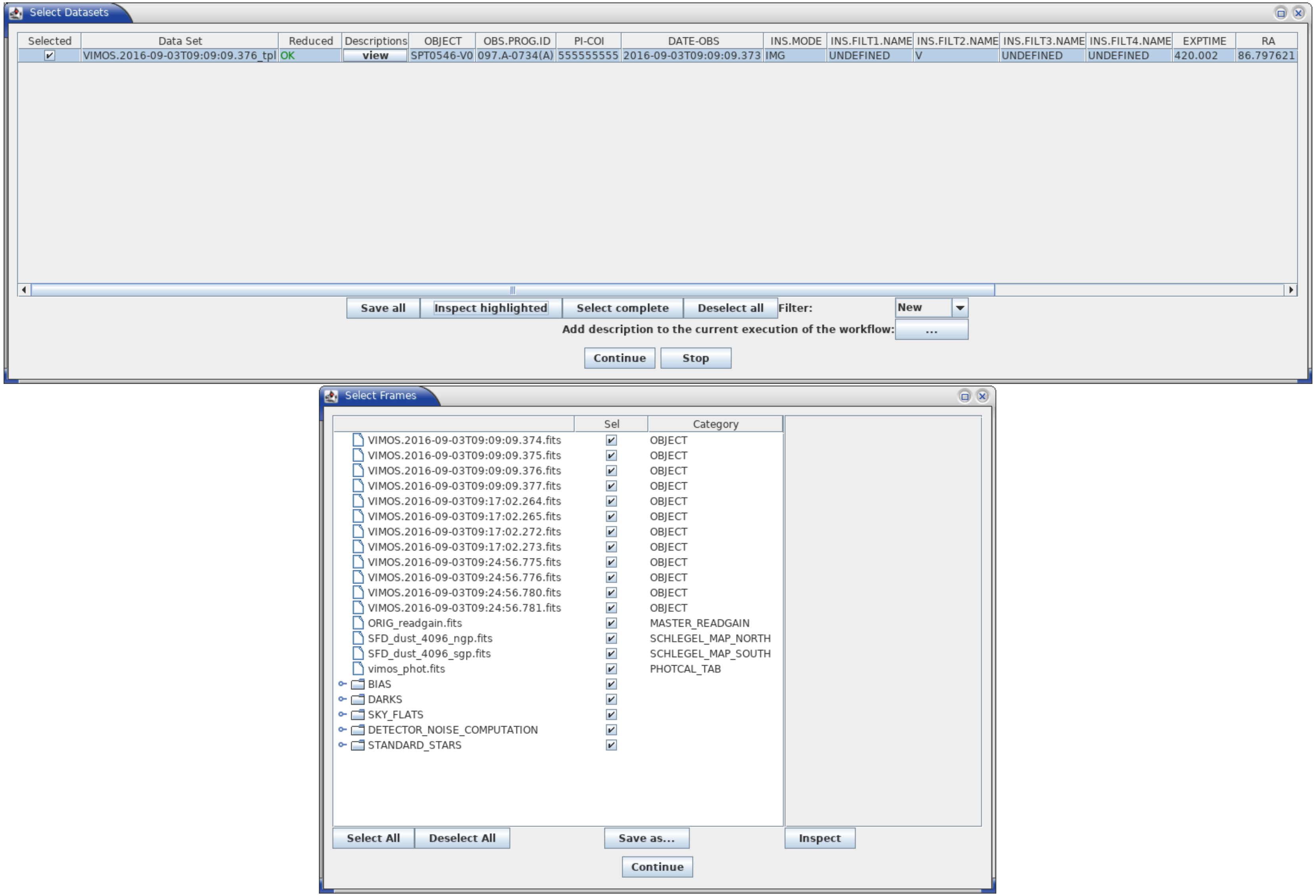
Step 3: Standard Star Reduction

Step 4: Science Image Reduction

Step 5: Output Organisation

Auxiliary and debug parameters, please do not change:
- GLOBAL_TIMESTAMP: 2018-07-03T14:55:12
- ESORexArgs: --suppress-prefix=TRUE
- END_PRODUCTS_SUBDIR: 2018-07-03T14:55:12/VIMOS.2017-01-14T07:29:13.894_tpl
- N_SELECTED_DATASETS: 1

Figure 5.2: The VIMOS-IMG workflow canvas.



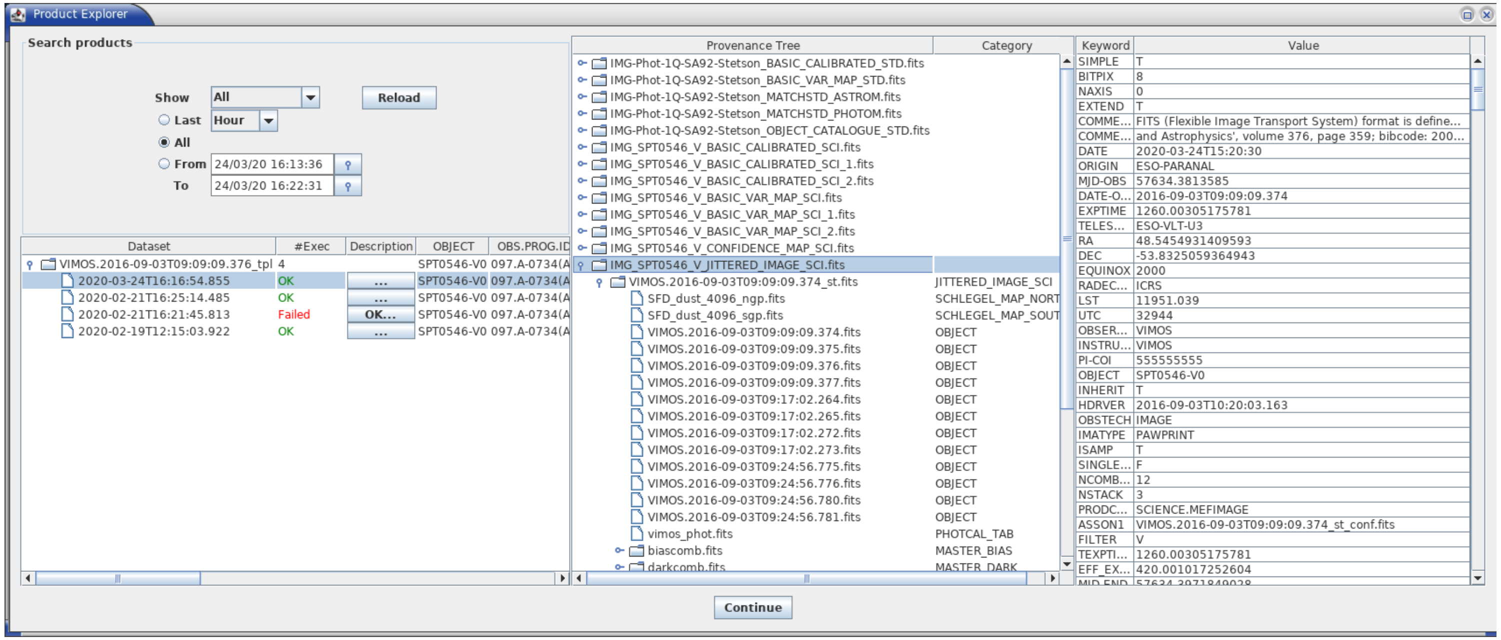
Selected	Data Set	Reduced	Descriptions	OBJECT	OBS.PROG.ID	PI-COI	DATE-OBS	INS.MODE	INS.FILT1.NAME	INS.FILT2.NAME	INS.FILT3.NAME	INS.FILT4.NAME	EXPTIME	RA
☑	VIMOS.2016-09-03T09:09:09.376_tpl	OK	view	SPT0546-V0	097.A-0734(A)	555555555	2016-09-03T09:09:09.373	IMG	UNDEFINED	V	UNDEFINED	UNDEFINED	420.002	86.797621

Filter: New

Add description to the current execution of the workflow: ...

	Sel	Category
VIMOS.2016-09-03T09:09:09.374.fits	☑	OBJECT
VIMOS.2016-09-03T09:09:09.375.fits	☑	OBJECT
VIMOS.2016-09-03T09:09:09.376.fits	☑	OBJECT
VIMOS.2016-09-03T09:09:09.377.fits	☑	OBJECT
VIMOS.2016-09-03T09:17:02.264.fits	☑	OBJECT
VIMOS.2016-09-03T09:17:02.265.fits	☑	OBJECT
VIMOS.2016-09-03T09:17:02.272.fits	☑	OBJECT
VIMOS.2016-09-03T09:17:02.273.fits	☑	OBJECT
VIMOS.2016-09-03T09:24:56.775.fits	☑	OBJECT
VIMOS.2016-09-03T09:24:56.776.fits	☑	OBJECT
VIMOS.2016-09-03T09:24:56.780.fits	☑	OBJECT
VIMOS.2016-09-03T09:24:56.781.fits	☑	OBJECT
ORIG_readgain.fits	☑	MASTER_READGAIN
SFD_dust_4096_ngp.fits	☑	SCHLEGEL_MAP_NORTH
SFD_dust_4096_sgp.fits	☑	SCHLEGEL_MAP_SOUTH
vimos_phot.fits	☑	PHOTCAL_TAB
BIAS	☑	
DARKS	☑	
SKY_FLATS	☑	
DETECTOR_NOISE_COMPUTATION	☑	
STANDARD_STARS	☑	

Figure 5.3: Top: "Select DataSets" window. **Bottom:** "Inspect highlighted" window showing details of the VIMOS demo data set.



Search products

Show: All ▾ Reload

○ Last: Hour ▾

● All

○ From: 24/03/20 16:13:36

To: 24/03/20 16:22:31

Dataset	#Exec	Description	OBJECT	OBS.PROG.ID
VIMOS.2016-09-03T09:09:09.376_tpl	4		SPT0546-V0	097.A-0734(A
2020-03-24T16:16:54.855	OK	...	SPT0546-V0	097.A-0734(A
2020-02-21T16:25:14.485	OK	...	SPT0546-V0	097.A-0734(A
2020-02-21T16:21:45.813	Failed	OK...	SPT0546-V0	097.A-0734(A
2020-02-19T12:15:03.922	OK	...	SPT0546-V0	097.A-0734(A

Provenance Tree	Category
IMG-Phot-1Q-SA92-Stetson_BASIC_CALIBRATED_STD.fits	
IMG-Phot-1Q-SA92-Stetson_BASIC_VAR_MAP_STD.fits	
IMG-Phot-1Q-SA92-Stetson_MATCHSTD_ASTROM.fits	
IMG-Phot-1Q-SA92-Stetson_MATCHSTD_PHOTOM.fits	
IMG-Phot-1Q-SA92-Stetson_OBJECT_CATALOGUE_STD.fits	
IMG_SPT0546_V_BASIC_CALIBRATED_SCI.fits	
IMG_SPT0546_V_BASIC_CALIBRATED_SCI_1.fits	
IMG_SPT0546_V_BASIC_CALIBRATED_SCI_2.fits	
IMG_SPT0546_V_BASIC_VAR_MAP_SCI.fits	
IMG_SPT0546_V_BASIC_VAR_MAP_SCI_1.fits	
IMG_SPT0546_V_BASIC_VAR_MAP_SCI_2.fits	
IMG_SPT0546_V_CONFIDENCE_MAP_SCI.fits	
IMG_SPT0546_V_JITTERED_IMAGE_SCI.fits	
VIMOS.2016-09-03T09:09:09.374_st.fits	JITTERED_IMAGE_SCI
SFD_dust_4096_ngp.fits	SCHLEGEL_MAP_NORT
SFD_dust_4096_sgp.fits	SCHLEGEL_MAP_SOUT
VIMOS.2016-09-03T09:09:09.374.fits	OBJECT
VIMOS.2016-09-03T09:09:09.375.fits	OBJECT
VIMOS.2016-09-03T09:09:09.376.fits	OBJECT
VIMOS.2016-09-03T09:09:09.377.fits	OBJECT
VIMOS.2016-09-03T09:17:02.264.fits	OBJECT
VIMOS.2016-09-03T09:17:02.265.fits	OBJECT
VIMOS.2016-09-03T09:17:02.272.fits	OBJECT
VIMOS.2016-09-03T09:17:02.273.fits	OBJECT
VIMOS.2016-09-03T09:24:56.775.fits	OBJECT
VIMOS.2016-09-03T09:24:56.776.fits	OBJECT
VIMOS.2016-09-03T09:24:56.780.fits	OBJECT
VIMOS.2016-09-03T09:24:56.781.fits	OBJECT
vimos_phot.fits	PHOTCAL_TAB
biascomb.fits	MASTER_BIAS
darkcomb.fits	MASTER_DARK

Keyword	Value
SIMPLE	T
BITPIX	8
NAXIS	0
EXTEND	T
COMME...	FITS (Flexible Image Transport System) format is define...
COMME...	and Astrophysics', volume 376, page 359; bibcode: 200...
DATE	2020-03-24T15:20:30
ORIGIN	ESO-PARANAL
MJD-OBS	57634.3813585
DATE-O...	2016-09-03T09:09:09.374
EXPTIME	1260.00305175781
TELES...	ESO-VLT-U3
RA	48.5454931409593
DEC	-53.8325059364943
EQUINOX	2000
RADEC...	ICRS
LST	11951.039
UTC	32944
OBSER...	VIMOS
INSTRU...	VIMOS
PI-COI	555555555
OBJECT	SPT0546-V0
INHERIT	T
HDRVER	2016-09-03T10:20:03.163
OBSTECH	IMAGE
IMATYPE	PAWPRINT
ISAMP	T
SINGLE...	F
NCOMB...	12
NSTACK	3
PRODC...	SCIENCE.MEFIMAGE
ASSON1	VIMOS.2016-09-03T09:09:09.374_st_conf.fits
FILTER	V
TEXPTI...	1260.00305175781
EFF_EX...	420.001017252604
MJD-END	57634.3971849028

Figure 5.4: The VIMOS-IMG workflow Product Explorer window. It shows all datasets reduced in previous executions; for each dataset the full reduction chain for all the pipeline products is also shown.

6 About the main esoreflex canvas

6.1 Saving And Loading Workflows

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

6.2 Buttons

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

- ⊕ - Zoom in.
- 🔍 - Reset the zoom to 100%.
- ⊡ - Zoom the workflow to fit the current window size (Recommended).
- ⊖ - Zoom out.
- ▶ - Run (or resume) the workflow.
- ⏸ - Pause the workflow execution.
- ⬣ - Stop the workflow execution.

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

6.3 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.

7 Reducing your own data

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

First, we suggest the reader to familiarize with the workflow by reducing the demo dataset first (Section 5), but it is not a requirement.

7.1 The esoreflex command

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

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

```
esoreflex -h
```

The output is the following.

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

7.2 Launching the workflow

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

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

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

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

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

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

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

2. Type:

   ```
   esoreflex -e vimos_ima
   ```

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

   ```
   esoreflex -e BOOKKEEPING_DB <database_path> vimos_ima
   ```

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

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

   A full description of the Product Explorer will be given in Section 7.3.3

3. Type:

   ```
   esoreflex  vimos_ima &
   ```

   to launch the VIMOS Imaging `esoreflex` workflow. The VIMOS Imaging workflow window will appear (Fig. 5.2). Please configure the set-up directories `ROOT_DATA_DIR`, `RAW_DATA_DIR`, and other workflow parameters as needed. Just double-click on them, edit the content, and press `OK`. Remember to specify the same `<database_path>` as for the Product Explorer, if it has been opened at step #2, to synchronize the two processes.

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

5. Press the button ▶ to start the workflow. First, the workflow will highlight and execute the `Initialise` actor, which among other things will clear any previous reductions if required by the user (see Section 9.1).

Secondly, if set, the workflow will open the Product Explorer, allowing the user to inspect previously reduced datasets (see Section 7.3.3 for how to configure this option).

7.3 Workflow Steps

7.3.1 Data Organisation And Selection

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

A DataSet lists three different pieces of information for each of its files, namely 1) the file name (including the path), 2) the file category, and 3) a string that is called the "purpose" of the file. The DO uses the OCA[5] rules to find the files to include in a DataSet, as well as their categories and purposes. The file category identifies different types of files, and it is derived by information in the header of the file itself. A category could for example be RAW_CALIBRATION_1, RAW_CALIBRATION_2 or RAW_SCIENCE, depending on the instrument. The purpose of a file identifies the reason why a file is included in a DataSet. The syntax is action_1/action_2/action_3/ ... /action_n, where each action_i describes an intended processing step for this file (for example, creation of a MASTER_CALIBRATION_1 or a MASTER_CALIBRATION_2). The actions are defined in the OCA rules and contain the recipe together with all file categories required to execute it (and predicted products in case of calibration data). For example, a workflow might include two actions action_1 and action_2. The former creates MASTER_ CALIBRATION_1 from RAW_CALIBRATION_1, and the later creates a MASTER_CALIBRATION_2 from RAW_CALIBRATION_2. The action_2 action needs RAW_CALIBRATION_2 frames and the MASTER_ CALIBRATION_1 as input. In this case, these RAW_CALIBRATION_1 files will have the purpose action_ 1/action_2. The same DataSet might also include RAW_CALIBRATION_1 with a different purpose; irrespective of their purpose the file category for all these biases will be RAW_CALIBRATION_1.

The Datasets created via the DataOrganiser will be displayed in the DataSet Chooser. Here the users have the possibility to inspect the various datasets and decide which one to reduce. By default, DataSets that have not been reduced before are highlighted for reduction. Click either Continue in order to continue with the workflow reduction, or Stop in order to stop the workflow. A full description of the DataSet Chooser is presented in Section 7.3.2.

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

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

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

7.3.2 DataSetChooser

The DataSetChooser displays the DataSets available in the “Select Data Sets” window, activating vertical and horizontal scroll bars if necessary (Fig. 5.3).

Some properties of the DataSets are displayed: the name, the number of files, a flag indicating if it has been successfully reduced (a green OK), if the reduction attempts have failed or were aborted (a red FAILED), or if it is a new dataset (a black "-"). The column "Descriptions" lists user-provided descriptions (see below), other columns indicate the instrument set-up and a link to the night log.

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

You may also highlight a single DataSet in blue by clicking on the relevant line. If you subsequently click on Inspect Highlighted, then a “Select Frames” window will appear that lists the set of files that make up the highlighted DataSet including the full filename[6], the file category (derived from the FITS header), and a selection tick box in the right column. The tick boxes allow you to edit the set of files in the DataSet which is useful if it is known that a certain calibration frame is of poor quality (e.g: a poor raw flat-field frame). The list of files in the DataSet may also be saved to disk as an ASCII file by clicking on Save As and using the file browser that appears.

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

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

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

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

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

7.3.3 The ProductExplorer

The ProductExplorer is an interactive component in the `esoreflex` workflow whose main purpose is to list the final products with the associated reduction tree for each dataset and for each reduction attempt (see Fig. 5.4).

Configuring the ProductExplorer

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

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

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

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

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

Exploring the data reduction products

The left window of the ProductExplorer GUI shows the executions for all the datasets (see Fig. 5.4). Once you click on a dataset, you get the list of reduction attemps. Green and red flags identify successfull or unsucessfull reductions. Each reduction is linked to the “Description” tag assigned in the “Select Dataset” window.

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

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

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

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

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

8 VIMOS Imaging Pipeline Processing

This section provides a brief summary of the processing steps of the VIMOS imaging pipeline that are executed by running the `esoreflex` workflow. A more detailed description of the recipes can be found in the VIMOS imaging pipeline manual ([2]).

For all observations the following data reduction steps are performed:

- Bias correction using a master bias frame.
- Dark correction using a master dark frame that has been scaled to the exposure time of the science/standard observation.
- Flat fielding using a master twilight flat field for the matching filter.
- A gain correction is applied to each detector. This is done to compensate for slight gain variations between the detectors.
- Images science OB's are trimmed to remove parts that are badly affected by shadowing. This is not done in standard star images, as these are not, ultimately, stacked.
- A source catalogue is extracted for each exposure and this is used to fit a world coordinate system (WCS). The ZPN projection is used in conjunction with known projection coefficients. In general the WCS is done relative to the PPMXL point-source catalogue.
- The individual exposures for a single OB and a given detector are stacked using the WCS solutions defined above. The stacks are formed using a bi-linear interpolation algorithm to resample the input pixels onto the output grid. This leads to an OB stack for each detector. The stack is averaged by the number of input images. Therefore, all magnitude computations should be done using the effective exposure time header `EFF_EXPT`.
- A source catalogue is extracted from the stacked images (one for each detector).
- The source catalogue is used to redefine the WCS for the stack.
- **The photometric calibration of the science frames is done using VIMOS standard star exposures.** Since the limiting magnitude of the APASS catalogue is too bright, almost all its standard stars are saturated in all but the shortest exposure time VIMOS science images. Therefore, an *in situ* photometric calibration of the science fields is not possible with the APASS catalogue. A more detailed description of this is given in Appendix B. In the event that no suitable standard star exposure exists (i.e. if no STD image exists within the same night as the associated science image), the zeropoint of the science exposure is defined using the zeropoint median (as measured over several years of standard star data) for that VIMOS filter. If such an estimate of the zeropoint has been made, it will be reflected in the data product headers:

  `ZPFUDGED` = True

  `PHOTZPER` = 0.30 (a large zeropoint error equivalent to a conservative σ of the filter average)

  `HIERARCH ESO PRO REC1 CAL8 CATG` $\neq$ `OBJECT_CATALOGUE_STD`

9 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.

9.1 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; [7]).

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

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

By default the `EraseDirs` parameter is set to `false`, which means that no directories are cleaned before executing the workflow, and the recipe actors will work in Lazy Mode (see Section 9.2.4), reusing the previous pipeline recipe outputs if input files and parameters are the same as for the previous execution, which saves considerable processing time. Sometimes it is desirable to set the `EraseDirs` parameter to `true`, which forces the workflow to recursively delete the contents of the directories specified by `BOOKKEEPING_DIR`, `LOGS_DIR`, and `TMP_PRODUCTS_DIR`. This is useful for keeping disk space usage to a minimum and will force the workflow to fully re-reduce the data each time the workflow is run.

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

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

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

`DefringingEnabled`: If set to `true`, the workflow will create a master fringe frame from a sequence of science images. That master fringe frame will then be used to de-fringe those images. This is only recommended for data taken with the "old" set of VIMOS chips (prior to Aug 2010) and with the two reddest filters (I or z). If set to `false` no fringe frames are created and the science images will not be de-fringed.

`Compute readgain table`: If set to `false`, the default values for the VIMOS-IMG detector readnoise and gain (provided by a static calibration file) will be used. If set to `true`, the workflow will attempt to identify the appropriate raw data files and will process them in the `Master Readgain` actor. If the recipe is successful, the values derived from the input raw data will be used in subsequent actors.

There a number of additional parameters that can be set under the “Catalogue Selection” area of the workflow canvas:

`ASTROM_CATALOGUE`: The default behaviour is to use the CDS to retrieve the astrometric catalogue online. If a user does not use CDS to retrieve an astrometric catalogue online, this parameter is used to identify the catalogue that should be used. Accepted values are 0 for 2MASS, 1 for PPMXL (default), and 2 for APASS, and 3 for a LOCAL catalogue. Note that index files and the catalogues themselves must be present in one of the Setup Directories.

`PHOTOM_CATALOGUE`: The default behaviour is to use the CDS to retrieve the photometric catalogue online. If a user does not use CDS to retrieve a photometric catalogue online, this parameter is used to identify the catalogue that should be used. Accepted values are 0 for APASS (default), and 1 for a LOCAL catalogue. Note that index files and the catalogues themselves must be present in one of the Setup Directories.

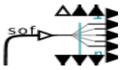
`CDS ASTROMETRIC CATALOGUE`: If a user wishes to use CDS to retrieve an astrometric catalogue online, this parameter is used to identify the catalogue that should be used. Accepted values are 2mass, usnob, ppmxl (default), wise, and none. If ’none’ is selected, the data will be calibrated using the catalogue specified in `ASTROM_CATALOGUE`.

9.2 Workflow Actors

This section briefly describes the different actors present on the main canvas. These are divided into two categories: simple actors and composite actors. A simple actor is a fundamental unit of ‘action’ within a workflow; it is represented by a blue/green rectangle or other shape. A composite actor is comprised of one or more simple actors and is represented as a ’layered’ blue/green rectangle. A number of parameters that affect how an actor behaves can be edited by right-clicking on an actor and selecting ‘Configure Actor’. For composite actors, these parameters may be edited by double-clicking on the actor.

9.2.1 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:

-
DataOrganizer
input data
datasets out
 - The DataOrganiser actor.
-
DataSetChooser
datasets in
dataset out
#selected
 - The DataSetChooser actor (inside a composite actor).
- 

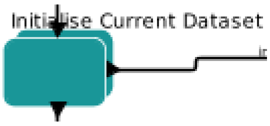
 - The FitsRouter actor Redirects files according to their categories.
-
Product Renamer
 - The ProductRenamer actor.
-
ProductExplorer
end date
trigger
 - The ProductExplorer actor (inside a composite actor).

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

9.2.2 Composite Actors

The following is a list of composite actors, with a brief description, on the main canvas of the VIMOS Imaging workflow:

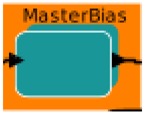
-
Initialise
 - The Initialise actor will execute some preliminary steps: erase directories if requested, and assign the timestamp and the location of Setup Directories.
-
DataSet Selection
 - The DataSet Selection actor constructs data sets and selects which one(s) to process.

- 

 - The Initialise Current Dataset executes some preliminary steps required to process a particular dataset, e.g. create directories, show a window with the processing status, keep track of which data set this is, etc.

-

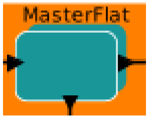
 - The CatalogueSelector actor selects which photometric and astrometric catalogue to use when processing a data set; this is determined by the values of ASTROM_CATALOGUE and PHOTOM_CATALOGUE parameters on the main canvas.

- 

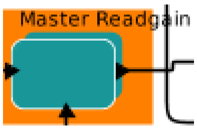
 - The MasterBias actor executes the vimos_ima_bias recipe and may launch an interactive window displaying the results.

-

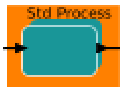
 - The MasterDark actor executes the vimos_ima_dark recipe and may launch an interactive window displaying the results.

- 

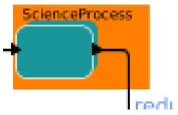
 - The MasterFlat actor executes the vimos_ima_twilight_flat recipe and may launch an interactive window displaying the results.

- 

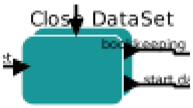
 - The MasterReadgain actor executes the vimos_ima_det_noise recipe (if requested) and may launch an interactive window displaying the results.

- 

 - The StdProcess actor executes the vimos_ima_standard recipe and may launch an interactive window displaying the results.

- 

 - The ScienceProcess actor executes the vimos_ima_science recipe and may launch an interactive window displaying the results.

- 

 - The Close Dataset actor creates a README file with the list of final product files, displays a window with the processed data set, and organises data for the ProductExplorer actor.

9.2.3 Recipe Execution within Composite Actors

The composite actors that run the VIMOS-IMG recipes contain a number of components. With the exception of MasterReadgain, the actors generally have the same basic structure. Only a brief description of the structure is described here. If a user would like more information or would like to alter the functionality of these actors, please contact *https://support.eso.org*.

The actors begin by splitting the input files into those that share the same purpose; this is done by the `SofSplitter` actor. The appropriate files and recipe parameters are converted into a SoF file and recipe configuration file, respectively. These are then fed into a `RecipeLooper`. The looper starts by feeding the two files into the recipe executer actor (the actor named after the recipe). The output of the recipe and an interactivity flag are sent to a `PythonActor`. The `PythonActor` runs a Python script that inspects the output, and if requested, launches an interactive window. This window enables a user to examine the recipe products and to alter any recipe parameters. If a user selects "Re-run Recipe" in the interactive window, then the whole process starts again with the new recipe parameters. If a user selects "Continue" in the interactive window, or if interactivity is disabled, then some diagnostic data files might be thrown away, and the resultant data products are accumulated by the `SofAccumulator`. A new SoF file containing the recipe products is then created and sent to an output port. The interactivity is set or restored according to the user's preference. Note that for some actors, there are additional `Fits Router` actors, boolean switches, etc. that are used to handle missing or incomplete data. Figure 9.1 shows the inside of the MasterFlat actor.

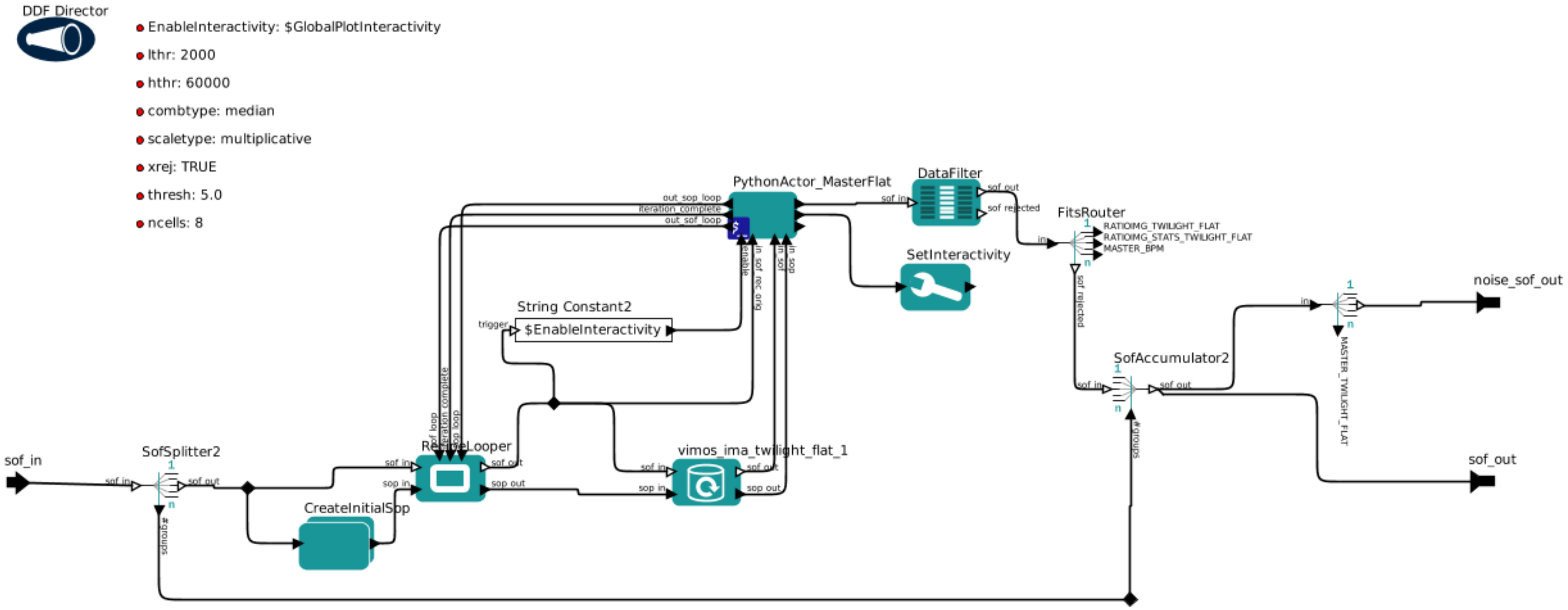
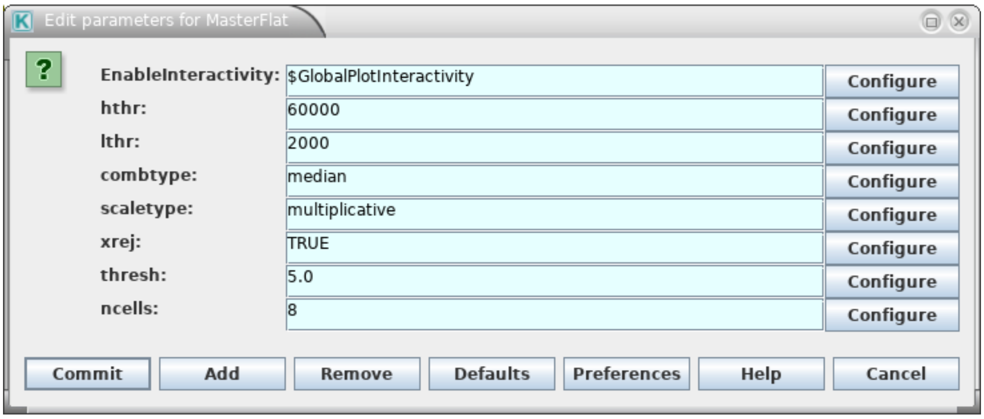


Figure 9.1: The internal actors within the MasterFlat actor.



EnableInteractivity: $GlobalPlotInteractivity
hthr: 60000
lthr: 2000
combtype: median
scaletype: multiplicative
xrej: TRUE
thresh: 5.0
ncells: 8

Figure 9.2: The Configure Actor window for the `vimos_ima_twilight_flat` Recipe Executer.

9.2.4 Lazy Mode

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

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

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

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

9.3 Workflow Details

This section describes the inner workings of some actors in more detail. It is intended for those users who wish to know what is going on behind the scenes, to diagnose errors or unexpected behaviour, or for those that wish to make modifications to the workflow.

9.3.1 Data Organisation and Selection Actors

The `Data Organiser` actor uses a special set of 'rules' to organise, classify, associate, and define purposes for data files. These rules are defined in a file called `vimos_ima.oca`. The default location for this file is `<install_dir>/share/esopipes/<pipeline-version>/reflex/`. Users may edit this file to suit their data reduction needs. Note, however, that the syntax and implementation of rules in an .oca file are somewhat arcane. Please refer to the OCA User Manual [8] for details.

By default, the workflow will group data together by the start time of the **template** in which it was taken (`TPL.START`), **not** by Observation Block. This means that the workflow cannot be used to combine or stack images that were taken as part of templates with different start times. If a user wishes to run pipeline recipes on data that have different template start times (e.g. all files taken as part of one OB), they may either use the command line interface "`esorex`", modify the OCA rules, or modify the `TPL.START` keywords such that they are the same for all images to be processed together. Note, however, that the VIMOS-IMG pipeline is **not** designed to process science data or standard star fields that span more than one OB.

A key role of the `Data Organiser` is to assign one or more 'purposes' to each file. The purpose of a file identifies the reason why a file is included in a DataSet. It is used, in part, as a filtering and accounting mechanism to ensure the correct files are sent to the correct recipes in the correct order. A `DataFilter` actor may be inserted into any relationship (i.e. solid black line) to see which files are being broadcast and to see the purposes of each file. The syntax for a single purpose is `ACTION_1/ACTION_2/ACTION_3/ ... ACTION_n`, where each `ACTION_i` describes each processing step for this file. These actions are defined in the OCA rules. In this context, an ACTION comprises a recipe name, a list of required and optional inputs for that recipe, and the classification of output files from the recipe. Note that a single file may have more than one purpose. For example, the same raw `BIAS` file may be used to create a master bias image and to create a master readgain table.

The OCA rules are also used to assess whether or not a data set is complete. In order for a data set to be considered complete, a number of criteria must be met: a) there are a sufficient number of static calibration files of the right classification (this includes a check that at least one complete set of four files is present, one for each chip), b) there are enough files to create the appropriate master dark and master flat frames, and c) at least one raw standard star or science image is present. An incomplete data set will appear in the interactive DataSet Selection window in grey text. The 'missing' components of the data set are listed if the mouse is hovered over the row in which the missing data set appears, e.g. "MISSING MASTER_DARK". Inspecting incomplete data sets can be used to reveal why the organiser considers the data set to be incomplete. A user may still select incomplete data sets for processing, but the workflow will not be completed successfully.

9.3.2 Editing Recipe Parameters

Most parameters are explicitly set to their default values. In the `Std Process` and `Science Process` actors, the `savemstd` and `savecat` parameters are set to 'true' rather than the default value of 'false' and `cacheloc` is set to `$TMP_PRODUCTS_DIR` instead of '.'.

There are several ways to view or edit the value of recipe parameters. The choice of method depends on how a user prefers to interact with the workflow.

1. If `GlobalPlotInteractivity` is set to `true`, an interactive window will be launched when a recipe finishes. This window shows the values of the recipe parameters that were used to create the products for interactive inspection. Users may change the value of these parameters as needed and then click Re-run Recipe to run the recipe with the new parameters. Note that some recipes have a large number of parameters; some parameters may appear under another tab on the upper right-hand side of the interactive window.
2. From the main workflow canvas, a user may double-click on a composite actor that contains an actor that executes recipes (i.e. those with orange boxes around them). A list of recipe parameters, and perhaps other actor parameters, will appear. To edit a parameter, change the value in the box and press Commit.
3. If a user opens a composite actor that executes a recipe (right-click, then choose 'Open Actor'), the parameters can be seen on the canvas of that actor. The parameters are represented as "StringParameters" on the canvas and have a small red dot next to them. They can be changed by clicking on them in the canvas.
4. A user may double-click on the actor that runs a recipe; these actors share the name of the recipe they execute and appear to have a cylinder with a thick circular arrow on them. A window will appear that enables a user to change that actor's behaviour, including parameter values (see Figure 9.2). Note that most recipe parameters have a value set to `PORT`; this tells the actor to use specially crafted values from an input port. This method of changing recipe parameters is *not* recommended. If a non-`PORT` value is specified, the actor will ignore any changes to parameters made using the three methods listed above. However, there are two recipe parameters that can only be changed using this method: `prettynames` and `preview_only`. They are both set to `false`; `true` values have no use in a workflow environment.

9.3.3 Master Calibration Actors

The layout of the `MasterBias`, `MasterDark` and `MasterFlat` composite actors is fairly straightforward. However, the `Master Readgain` actor is somewhat complicated. The reason for the complexity is that a) a user can choose whether to run the recipe, and b) the master readgain table in the static calibration directory may be used instead of the recipe product. If the recipe is run, the interactive window compares the values in the static calibration file with the pipeline product.

9.3.4 Standard Star and Science Fields Actors

In addition to the main recipe loop, these actors check if any raw standard star or science fields are present before attempting to process them. A DataSet may only have one of these categories of files. If the required category of files is missing, the boolean logic in the actor ensures that the workflow will skip the recipe and create an empty product before moving to the next actor on the main canvas.

The `ScienceProcess` actor has an additional composite actor (`Master Fringe`) that is run just before the main recipe loop. If requested by the user, this actor creates a master fringe frame and passes it along to the main loop.

9.3.5 Output Organisation

After processing the input data for a particular DataSet, the workflow executes the `Product Renamer` actor. This actor copies the final products of the `Std Process` and `ScienceProcess` actors into the `END_PRODUCTS_DIR` directory and renames them with a name derived from values of certain FITS header keyword values. By default, the final products are renamed to a file of the form
`<HIERARCH.ESO.OBS.NAME>_<HIERARCH.ESO.PRO.CATG>.fits`,
with `<HIERARCH.ESO.OBS.NAME>` and `<HIERARCH.ESO.PRO.CATG>` representing the values of the corresponding FITS header keywords. If a file of that name already exists, an underscore followed by an incremental integer is appended to the filename. A user may customise this format (or other actor behaviour) by right-clicking on the `Product Renamer` actor, selecting "Configure Actor", and then editing `RenameKeywords` as appropriate. Users are referred to the VIMOS-IMG Pipeline Manual [2] for a description of the pipeline products and associated `HIERARCH.ESO.PRO.CATG` values.

9.4 Interactive Windows

The VIMOS-IMG workflow contains six interactive windows that allow the user to iterate on the processing of their data. The windows are launched by a Python actor that is part of the main recipe execution loop. A user may inspect or change the Python script that runs these windows. The name and location of the source `.py` files can be seen by double-clicking on the Python actor.

Every interactive window shares the same eight buttons on the top left side of the window, e.g. a Home icon. These buttons control zooming in on images and plots, the colour scale of images, etc. Users are referred to the Reflex Users Manual [7] for details on the functionality of these buttons.

Each window also shares a similar layout on the right hand side. One or more tabs appear in the top right; these show the values of the recipe parameters used to generate the data shown on the left. A short description of each parameter, the default value, and accepted values are displayed if the mouse is hovered over the white box. Three buttons appear below the list of parameters: 1) Continue Wkf will close the window and the workflow will continue with the pipeline products, 2) Re-run Recipe will re-run the recipe with some new parameters, and 3) Help opens a small window describing how the interactivity works. The "Disable this window in subsequent runs" is a tick-box. If clicked, the window will change the value of `EnableInteractivity` to `false` after closing the window. Alternatively, all interactive windows can be disabled by setting the `GlobalPlotInteractivity` parameter, on the main workflow canvas, to `False`.

If a recipe produces insufficient or invalid data for a particular plot or image, an informational message will be displayed in the appropriate panel, e.g. "No extension found" or "No objects in the catalogue".

9.4.1 Master Bias

This window (Figure 9.3) opens by showing a master bias image, one chip per image panel. The title of each image shows extension (chip) name of the image. The radio buttons on the upper left allow a user to change what is shown in each image panel. Clicking anywhere on the button or explanatory text will change the selection; note that there may be a short delay of 1-2 seconds between clicking and seeing the change.

If a reference bias is not provided to the recipe, the button options are a) the master bias image (default), and b) a histogram of the values in each image. The histograms show the pixel value distribution over ten bins within a range of the median value ± 7.4 times the median absolute deviation (equivalent to 5σ for a Gaussian distribution). The panels also show the useful values for how the pixel values are distributed (median, mean, median absolute deviation, and root-mean-square deviation from mean).

If a reference *is* provided to the recipe, two additional radio button options are available: c) an image of the reference bias subtracted from the master bias, and d) a scatter plot showing how the difference image varies over the chip in different "cells". The error bars on the scatter plot points are 1.48 times the median absolute deviation of the pixel values ($\approx 1\sigma$) within a cell on the difference image.

9.4.2 Master Dark

Same as for `Master Bias`, but for dark images (see Figure 9.4).

9.4.3 Master Flat

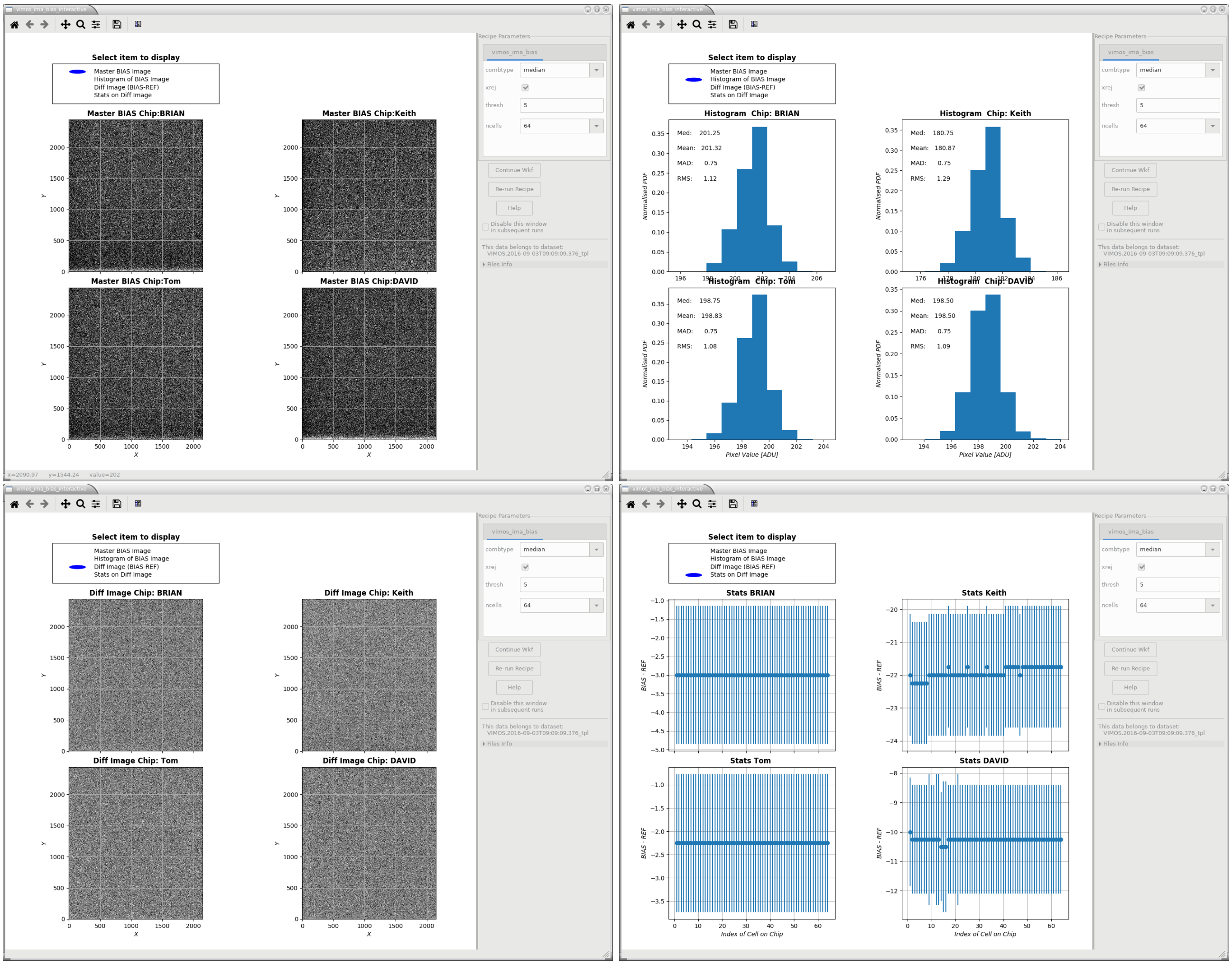
This window (see Figure 9.5 top left) opens by showing a master flat image, one chip per image panel. The title of each image shows filter and extension (chip) name of the image. The radio buttons on the upper left allow a user to change what is shown in each image panel. Clicking anywhere on the button or explanatory text will change the selection; note that there may be a short delay of 1-2 seconds between clicking and seeing the change.

If a reference flat is not provided to the recipe, the button options are a) the master flat image (default), and b) a histogram of the values in each image. The histograms (Figure 9.5 top right) show the pixel value distribution over ten bins within a range of the median value ± 7.4 times the median absolute deviation (equivalent to 5σ for a Gaussian distribution). The panels also show the useful values for how the pixel values are distributed (median, mean, median absolute deviation, and root-mean-square deviation from mean).

If a reference flat *is* provided to the recipe, two additional radio button options are available: c) an image of the master flat divided by the reference flat (Figure 9.5 bottom left), and d) a scatter plot showing how the ratio image varies over the chip in different "cells" (Figure 9.5 bottom right). The error bars on the scatter plot points are 1.48 times the median absolute deviation of the pixel values ($\approx 1\sigma$) within a cell on the difference image.

9.4.4 Master Readgain

This window (Figure 9.6) shows two scatterplots; there are no radio buttons. The lefthand plot compares the readnoise values in the static calibration file ("reference readnoise") to the master readnoise values derived by the



combtype median

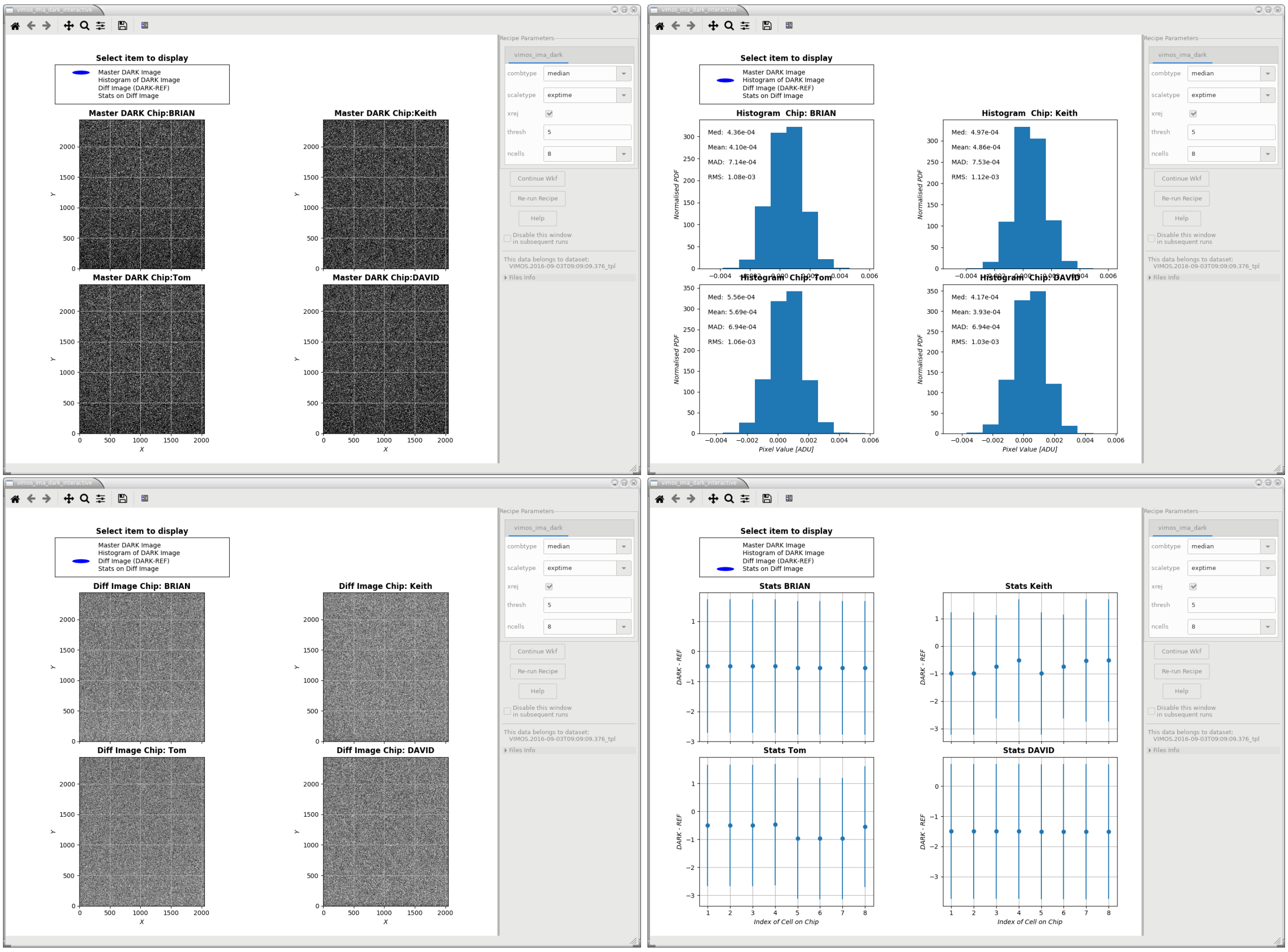
xrej ☑

thresh 5

ncells 64

This data belongs to dataset: VIMOS.2016-09-03T09:09:09.376_tpl

Figure 9.3: Example interactive windows for evaluating the master bias frames. Each layer of the interactive window can be accessed in the top left box "Select item to display".
Top Left: The master bias images of the four VIMOS detectors.
Top Right: The histogram statistics of each master bias frame.
Bottom Left: The difference images (master bias - reference bias) for each VIMOS detector.
Bottom Right: The difference image statistics (master bias - reference bias) for each VIMOS detector.



Recipe Parameters

vimos_ima_dark

combtype: median

scaletype: exptime

xrej: ☑

thresh: 5

ncells: 8

☐ Disable this window in subsequent runs

This data belongs to dataset: VIMOS.2016-09-03T09:09:09.376_tpl

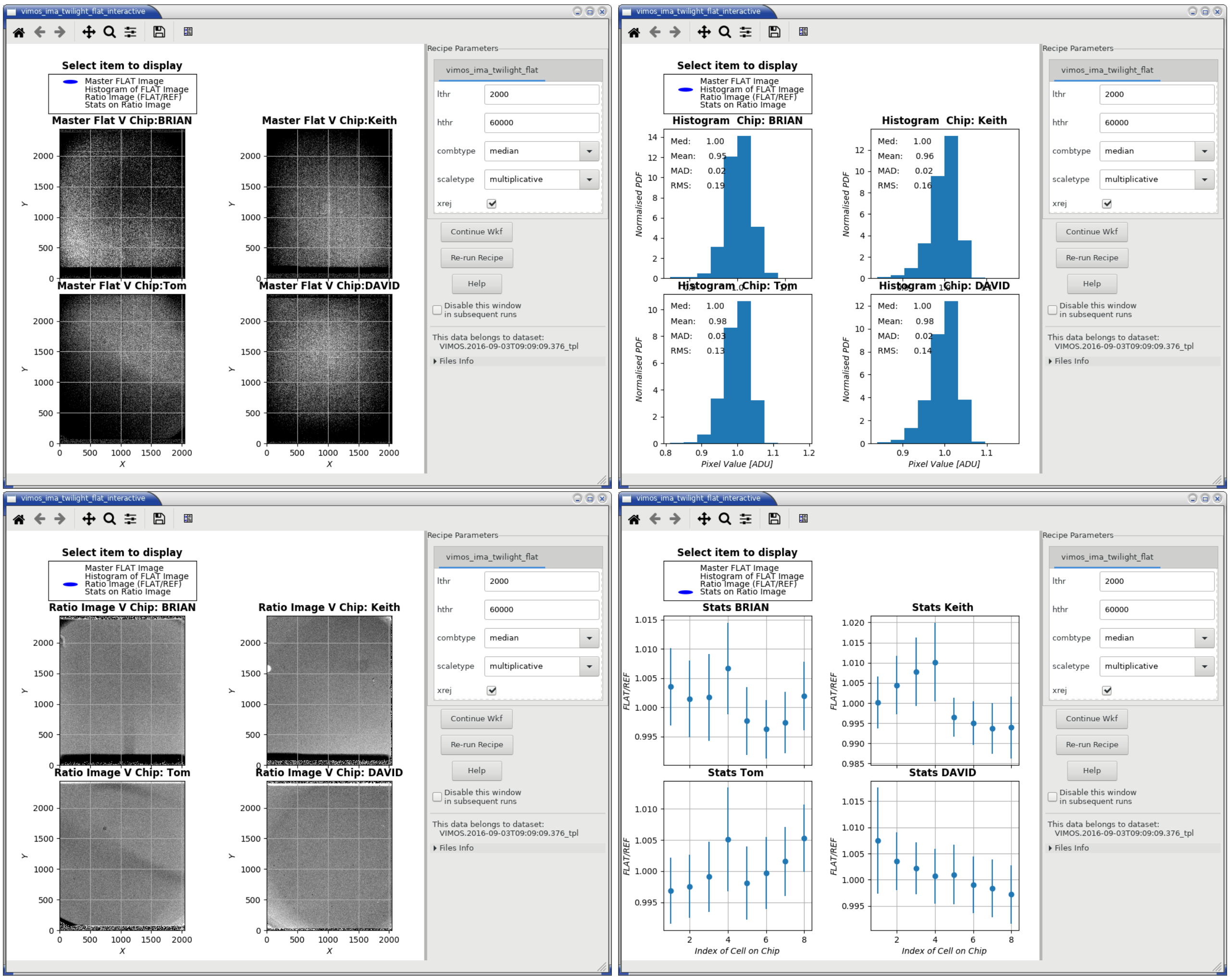
Histogram Chip: BRIAN
Med: 4.36e-04
Mean: 4.10e-04
MAD: 7.14e-04
RMS: 1.08e-03

Histogram Chip: Keith
Med: 4.97e-04
Mean: 4.86e-04
MAD: 7.53e-04
RMS: 1.12e-03

Histogram Chip: Tom
Med: 5.56e-04
Mean: 5.69e-04
MAD: 6.94e-04
RMS: 1.06e-03

Histogram Chip: DAVID
Med: 4.17e-04
Mean: 3.93e-04
MAD: 6.94e-04
RMS: 1.03e-03

Figure 9.4: Example interactive windows for evaluating the master dark frames. Each layer of the interactive window can be accessed in the top left box "Select item to display".
Top Left: The master dark images of the four VIMOS detectors.
Top Right: The histogram statistics of each master dark frame.
Bottom Left: The difference images (master dark - reference dark) for each VIMOS detector.
Bottom Right: The difference image statistics (master dark - reference dark) for each VIMOS detector.



Histogram Chip: BRIAN — Med: 1.00, Mean: 0.95, MAD: 0.02, RMS: 0.19

Histogram Chip: Keith — Med: 1.00, Mean: 0.96, MAD: 0.02, RMS: 0.16

Histogram Chip: Tom — Med: 1.00, Mean: 0.98, MAD: 0.03, RMS: 0.13

Histogram Chip: DAVID — Med: 1.00, Mean: 0.98, MAD: 0.02, RMS: 0.14

Recipe Parameters

vimos_ima_twilight_flat

lthr 2000

hthr 60000

combtype median

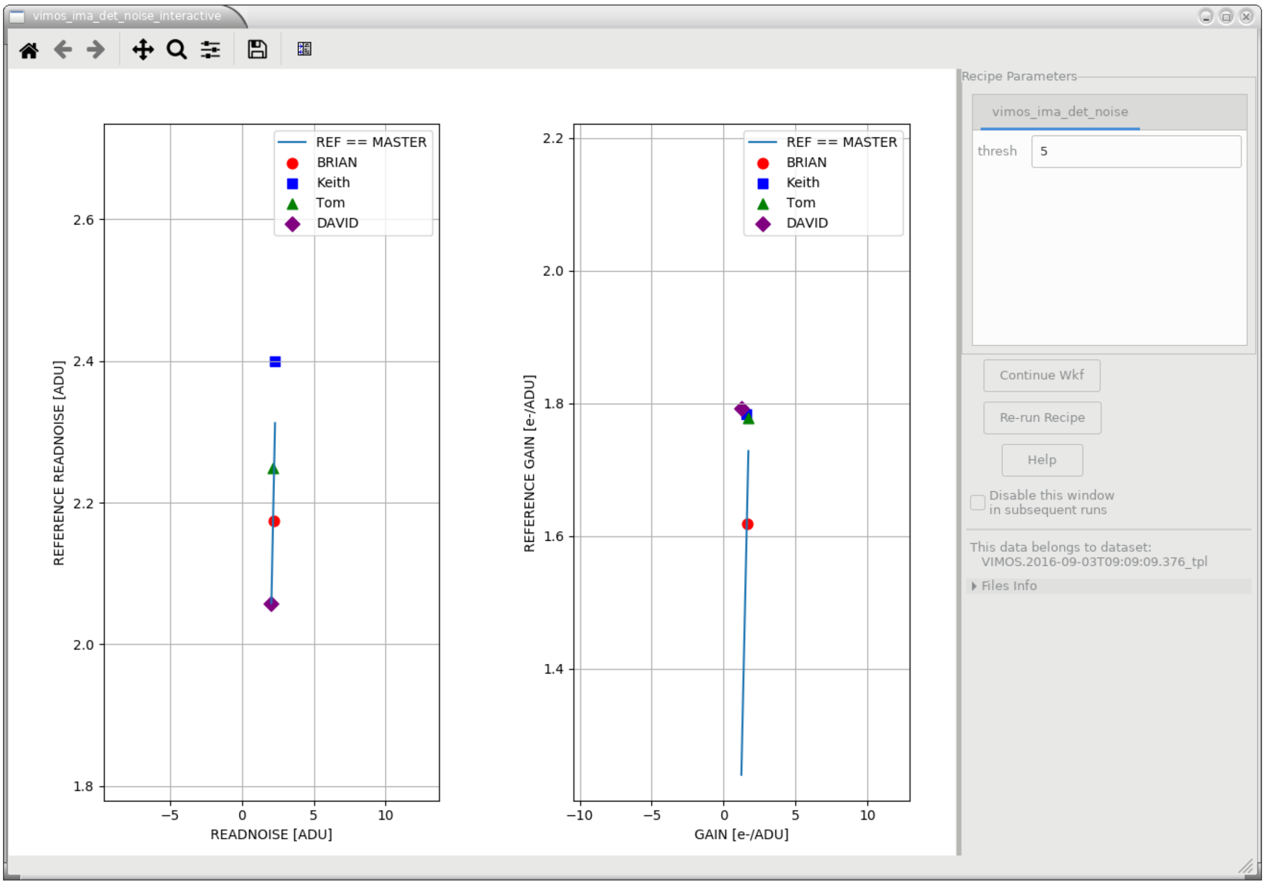
scaletype multiplicative

xrej ☑

□ Disable this window in subsequent runs

This data belongs to dataset: VIMOS.2016-09-03T09:09:09.376_tpl

Figure 9.5: Example interactive windows for evaluating the master flat frames. Each layer of the interactive window can be accessed in the top left box "Select item to display".
Top Left: The master flat images of the four VIMOS detectors.
Top Right: The histogram statistics of each master flat frame.
Bottom Left: The ratio images (master flat / reference flat) for each VIMOS detector.
Bottom Right: The ratio image statistics (master flat / reference flat) for each VIMOS detector.



thresh 5

□ Disable this window in subsequent runs

This data belongs to dataset:
VIMOS.2016-09-03T09:09:09.376_tpl

Figure 9.6: Example interactive window for evaluating master read-noise and gain values. The reference read-noise and gain relations are shown as a solid blue line.

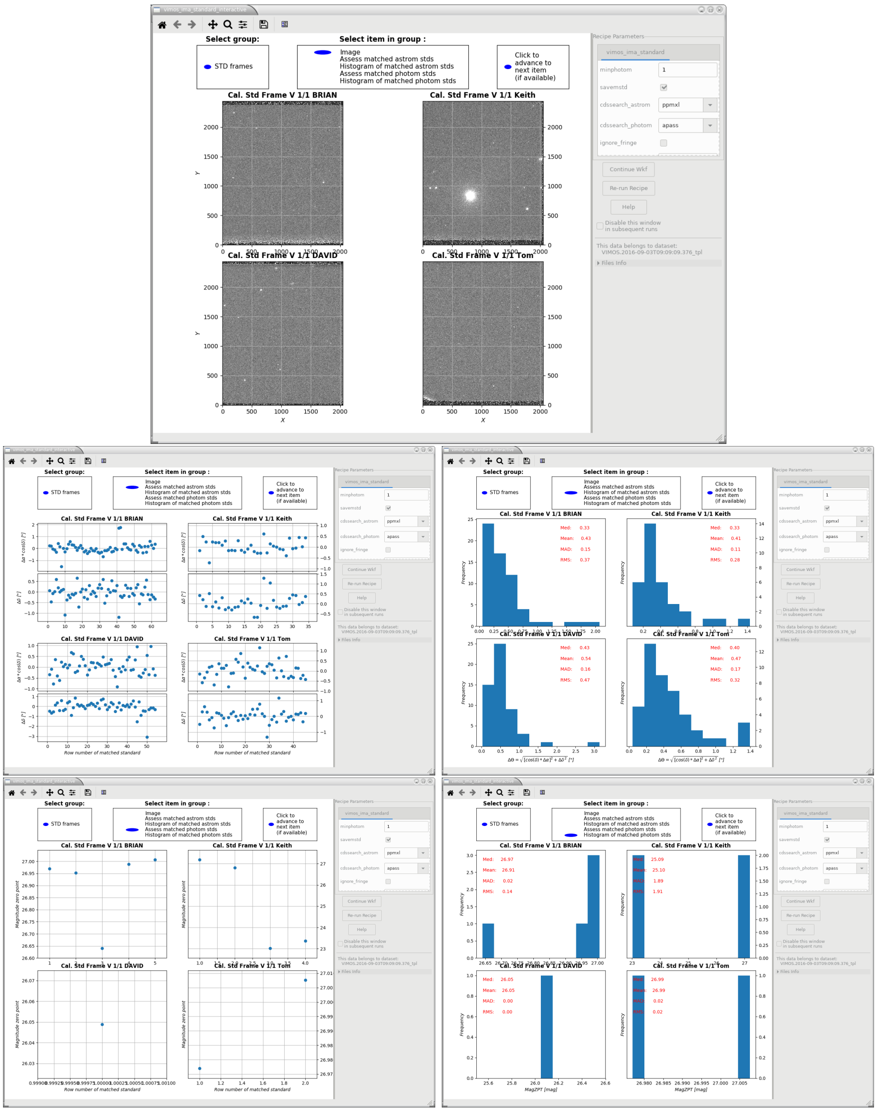
recipe; the units are ADUs. The righthand plot compares the gain values in the static calibration file ("reference gain") with the gain derived by the recipe; the axes are in units of electrons per ADU.

Each data point represents a different chip as indicated in the legend. In both panels, the solid blue line shows where the data points would be if the reference and master values are identical.

If any of the values in the recipe product are NULL or less than zero, the interactive window will not show any data. Instead, a message indicating that at least one value is invalid is displayed.

9.4.5 Standard Stars

This window (Figure 9.7) opens by showing a processed image of a standard star field, one chip per panel. The title of each panel indicates which image this is in the sequence of standard star fields. For example, if the recipe is provided with four `STD` files, the window will open with an image with `1/4` in the title to indicate that the first image (out of four) is shown. The full pathname of the file being displayed is shown if the mouse hovers over the image. The title also shows the extension name of the image.



Recipe Parameters

vimos_ima_standard

minphotom: 1

savemstd: ☑

cdssearch_astrom: ppmxl

cdssearch_photom: apass

ignore_fringe: □

□ Disable this window in subsequent runs

This data belongs to dataset:
VIMOS.2016-09-03T09:09:09.376_tpl

Select group:
STD frames
Select item in group :
Image
Assess matched astrom stds
Histogram of matched astrom stds
Assess matched photom stds
Histogram of matched photom stds
Click to advance to next item (if available)
Cal. Std Frame V 1/1 BRIAN
Cal. Std Frame V 1/1 Keith
Cal. Std Frame V 1/1 DAVID
Cal. Std Frame V 1/1 Tom
Continue Wkf
Re-run Recipe
Help
Files Info
Row number of matched standard
Magnitude zero point
Frequency
MagZPT [mag]
Med:
Mean:
MAD:
RMS:

Figure 9.7: Example interactive windows for evaluating the results of the `vimos_ima_standard` recipe. Each layer of the interactive window can be accessed in the top centre box "Select item in group".
Top: The standard star images of the four VIMOS detectors.
Centre Left: The $\Delta\alpha * \cos(\delta)$ and $\Delta\delta$ astrometric matches between the catalogue sources and the sources in the VIMOS standard star field.
Centre Right: Histograms and statistics of the astrometric matches between the catalogue sources and the sources in the VIMOS standard star field.
Bottom Left: The zeropoint magnitudes of the photometric matches between the `APASS` catalogue sources and the sources in the VIMOS standard star field.
Bottom Right: Histograms and statistics of the photometric matches between the `APASS` catalogue sources and the sources in the VIMOS standard star field.

There are three boxes with radio buttons across the top of the window. The button in the leftmost "Select group:" box does not respond to any clicks; it only indicates that the window is showing processed `STD` images.

The middle box ("Select item in group:") shows five options:

1. Image (default): show processed image (tag `BASIC_CALIBRATED_STD`) (see Figure 9.7 top panel).

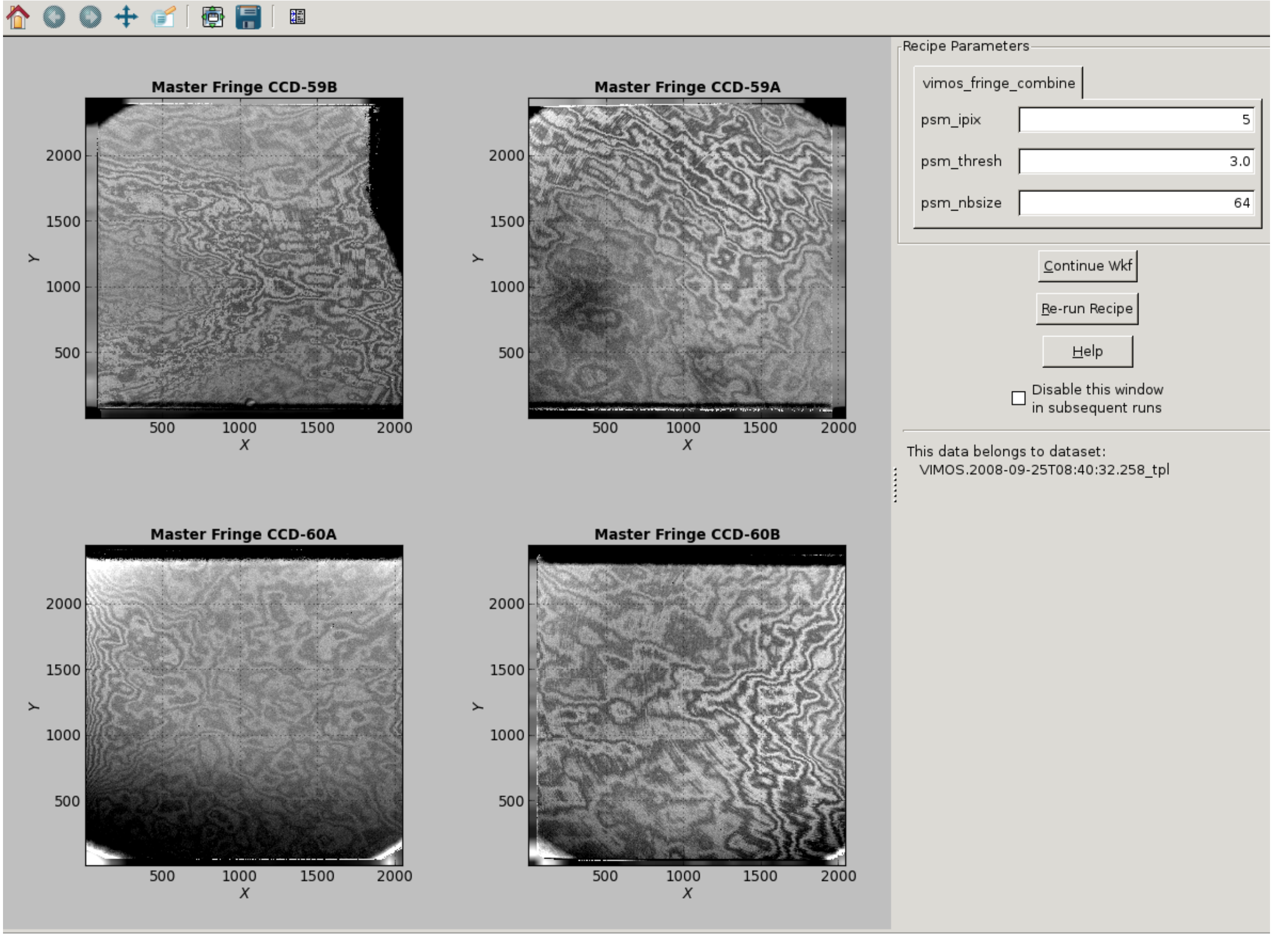
2. Assess matched astrom stds: if the recipe parameter `savemstd` is set to `true` (default value), clicking this button will show a comparison between the location of the astrometric standards in the reference catalogue (e.g. PPXML) and the derived location of the same objects on the calibrated image. Each image panel is now split into two subpanels. The top subpanel shows the difference between the right ascension (weighted by cos(DEC)) for each catalogued standard in units of arcseconds. The bottom subpanel shows the difference between the declination for each catalogued standard in units of arcseconds. In all subpanels, the x-axis refers to the row number of that object in the matched standard catalogue table. The full pathname to the table is shown if the mouse hovers over any panel. These plots may be useful to identify any outliers or systematic error in the astrometric calibration (See Figure 9.7 centre left).

3. Histogram of matched astrom stds: if the recipe parameter `savemstd` is set to `true`, clicking this button will show a histogram of the angular distance between the celestial coordinates of astrometric standards in the reference catalogue and the same objects on the calibrated image (in units of arcseconds). All data points are split into 10 bins in the histogram. The x-axis of each panel is angular distance; the y-axis is the frequency of occurrence. The red text in the upper right of each panel shows some statistical values describing the distribution of all the data points (median, mean, median absolute deviation, and rms deviation from the mean) (See Figure 9.7 centre right),

4. Assess matched photom stds: if the recipe parameter `savemstd` is set to `true`, clicking this button will show a comparison between the flux of the photometric standards in the reference catalogue (e.g. APASS) and the flux of the same objects on the calibrated image. This is represented as a magnitude zero point (using the measured fluxes from the "aper5" column of the object catalogue). The x-axis refers to the row number of each object in the matched standard catalogue table. The full pathname to that table is shown if the mouse hovers over any panel. Note that not all objects in the matched standard catalogue are shown; only those used for calibration because they have an error less than `magerrcut` are displayed on this plot (See Figure 9.7 bottom left).

5. Histogram of matched photom stds: if the recipe parameter `savemstd` is set to `true`, clicking this button will show a histogram of the magnitude zero points from the item above. All data points are split into 10 bins in the histogram. The x-axis of each panel is magnitude zero point in units of (Vega) magnitudes; the y-axis is the frequency of occurrence. The red text in the upper left of each panel shows some statistical values describing the distribution of all the data points (median, mean, median absolute deviation, and rms deviation from the mean) (See Figure 9.7 bottom right).

The rightmost box contains one radio button "Click to advance to next item (if available)". Clicking this button will advance each panel to show the same information for the next processed image. For example, if a user is looking at a histogram of matched photometric standards for the fourth image (out of four), clicking the button will show the same information, but for the first image in the sequence.



psm_ipix 5

psm_thresh 3.0

psm_nbsize 64

□ Disable this window in subsequent runs

This data belongs to dataset:
VIMOS.2008-09-25T08:40:32.258_tpl

Figure 9.8: Example interactive window for evaluating master fringe frames.

9.4.6 Master Fringe

If `DefringedEnabled` is set to true on the main canvas, there is an actor inside `Science Process` that will generate a master fringe frame (from the science frames themselves) using `vimos_ima_fringe`. That master fringe frame will then be used to de-fringe those images. This is only recommended for data taken with the "old" set of VIMOS chips (prior to Aug 2010) and with the two reddest filters (I or z). Since the tutorial data set included with the `esoreflex` workflow has been taken with the "new" VIMOS detectors, a fringe correction is not necessary. When use, the `vimos_ima_fringe` recipe will trigger a simple interactive window that will display an image of the master fringe frame for each chip (Figure 9.8). The chip name is shown in the title of each panel.

9.4.7 Science Fields

This window (Figure 9.9) opens by showing a processed, stacked image of a science field (one chip per panel). The full pathname of the file being displayed is shown if the mouse hovers over the image. The title shows the extension name of the image.

There are three boxes with radio buttons across the top of the window. The buttons in the leftmost "Select group:" box are used to select either the individual processed science images (SCI tag) or the single stacked image. If SCI is selected, the title of each panel will indicate which image in the sequence is being shown, e.g. "1/3" for the first image in a sequence of three. As above, the full pathname of the file being displayed is shown if the mouse hovers over an image.

The middle box ("Select item in group:") show five options:

1. Image: show processed image. If SCI is selected in the left hand box, the image is from a file with a BASIC_CALIBRATED_SCI tag. Otherwise, it is the JITTERED_IMAGE_SCI (stacked frame) file by default. (Figure 9.9 top and bottom, respectively).

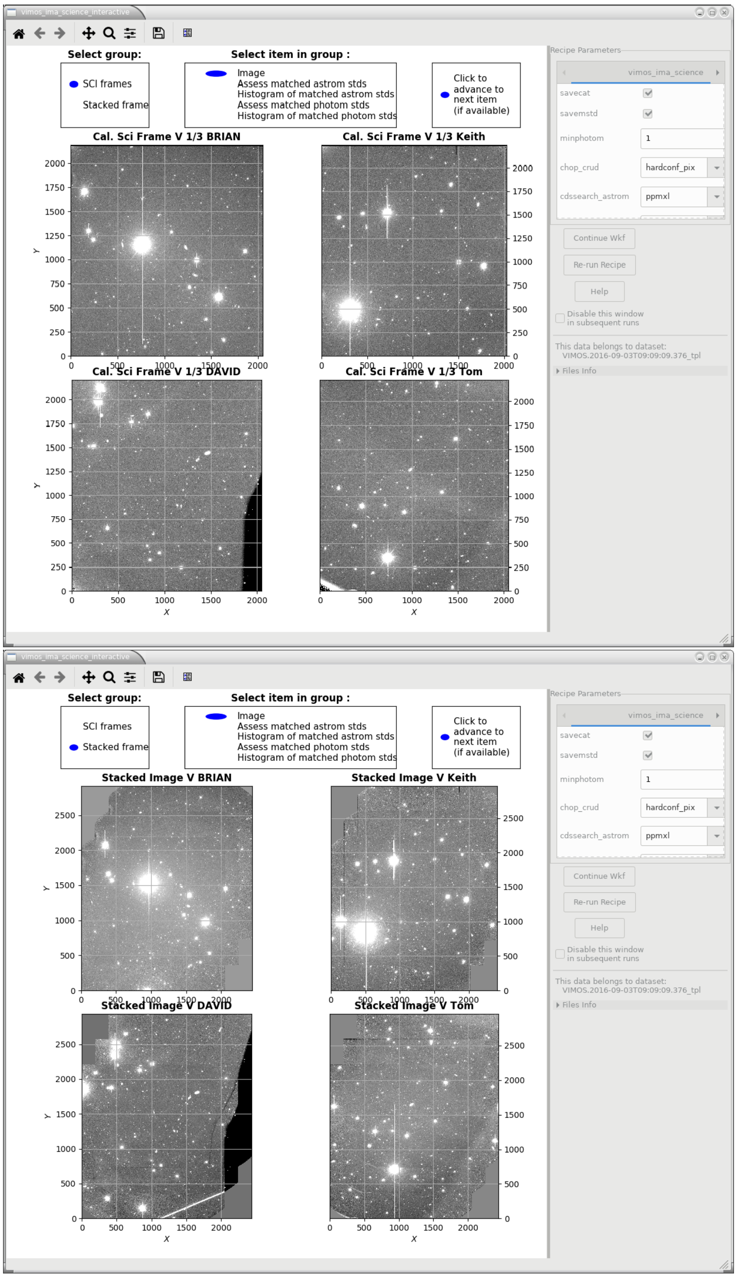
2. Assess matched astrom stds (Figure 9.10): if the recipe parameter savemstd is set to true (default value), this button shows a comparison between the location of the astrometric standards in the reference catalogue (e.g. PPXML) and the derived location of the same objects on the selected calibrated image. Each image panel is now split into two subpanels. The top subpanel shows the difference between the right ascension (weighted by cos(dec)) for each catalogued standard in units of arcseconds. The bottom subpanel shows the difference between the declination for each catalogued standard in units of arcseconds. In all subpanels, the x-axis refers to the row number of that object in the matched standard catalogue table. The full pathname to the table is shown if the mouse hovers over any panel. These plots may be useful to identify any outliers or systematic error in the astrometric calibration.

3. Histogram of matched astrom stds (Figure 9.10): if the recipe parameter savemstd is set to true, this button shows a histogram of the angular distance (in arcseconds) between the celestial coordinates of astrometric standards in the reference catalogue and the same objects on the selected calibrated image. All data points are split into 10 bins in the histogram. The x-axis of each panel is angular distance; the y-axis is the frequency of occurrence. The red text in the upper right of each panel shows some statistical values describing the distribution of all the data points (median, mean, median absolute deviation, and rms deviation from the mean).

4. Assess matched photom stds (Figure 9.10). Since the photometric zeropoint for the science images are adapted from the associated standard star field, no photometic map can be displayed for the science frames.

5. Histogram of matched photom stds (Figure 9.10). Since the photometric zeropoint for the science images are adapted from the associated standard star field, no photometic assessment histogram can be displayed for the science frames.

The rightmost box contains one radio button "Click to advance to next item (if available)". Clicking this button will advance each panel to show the similar information for the next image in the sequence. For example, if a user is looking at a histogram of matched astrometric standards for the fourth image (out of four), clicking the button will show similar plots, but for the first image in the sequence.


vimos_ima_science_interactive
Select group:
● SCI frames
Stacked frame
Select item in group :
Image
Assess matched astrom stds
Histogram of matched astrom stds
Assess matched photom stds
Histogram of matched photom stds
Click to advance to next item (if available)
Cal. Sci Frame V 1/3 BRIAN
Cal. Sci Frame V 1/3 Keith
Cal. Sci Frame V 1/3 DAVID
Cal. Sci Frame V 1/3 Tom
X
Y
vimos_ima_science_interactive
Select group:
SCI frames
● Stacked frame
Select item in group :
Image
Assess matched astrom stds
Histogram of matched astrom stds
Assess matched photom stds
Histogram of matched photom stds
Click to advance to next item (if available)
Stacked Image V BRIAN
Stacked Image V Keith
Stacked Image V DAVID
Stacked Image V Tom
X
Y
Continue Wkf
Re-run Recipe
Help

Recipe Parameters

vimos_ima_science

savecat ☑

savemstd ☑

minphotom 1

chop_crud hardconf_pix

cdssearch_astrom ppmxl

□ Disable this window in subsequent runs

This data belongs to dataset:
VIMOS.2016-09-03T09:09:09.376_tpl

Files Info

Recipe Parameters

vimos_ima_science

savecat ☑

savemstd ☑

minphotom 1

chop_crud hardconf_pix

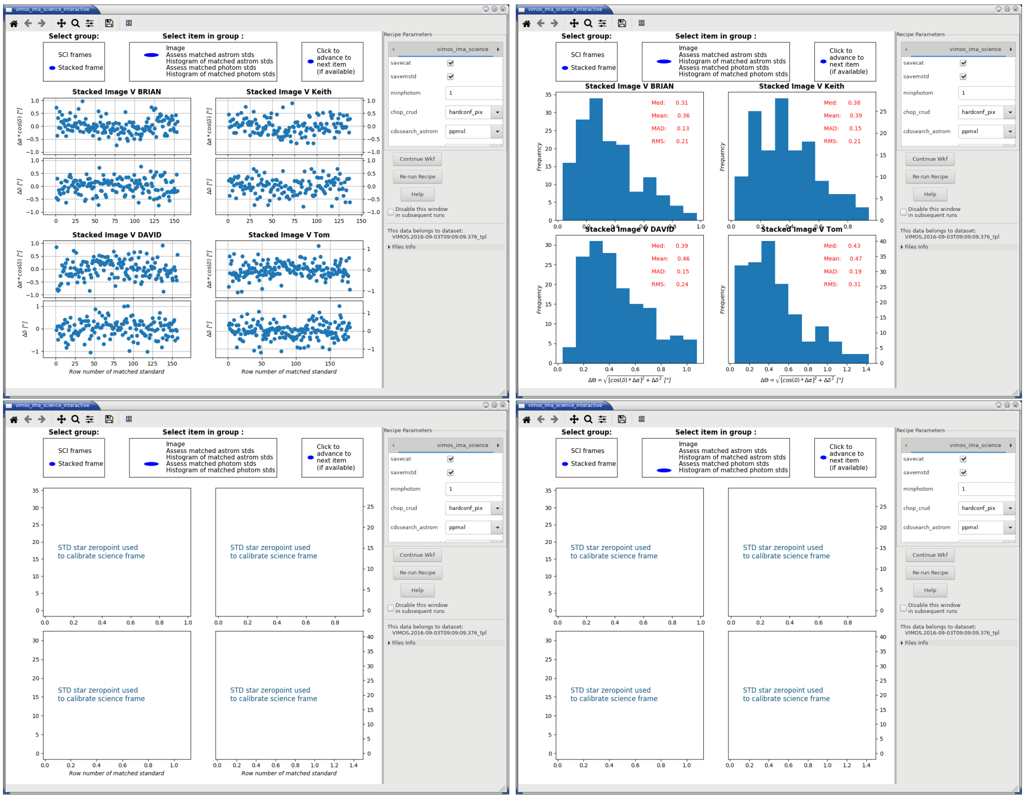
cdssearch_astrom ppmxl

□ Disable this window in subsequent runs

This data belongs to dataset:
VIMOS.2016-09-03T09:09:09.376_tpl

Files Info

Figure 9.9: Example interactive windows for evaluating the results of the `vimos_ima_science` recipe. The User can alternate between images of the individual science frames and the stacked (co-added) science frames via the top left box "Select group". Further interactive windows for the science actor are shown in Figure 9.10.
Top: The images of the processed individual science frames. Clicking on the top right hand box "Click to advance to next item" will display the next set of individual images.
Bottom: The images of the processed co-added (stacked) science frames.
In both figures, the frame vignetting (common to VIMOS images and due to the guide-probe) is apparent in the bottom right-hand corner of the bottom-left detector ('DAVID').


vimos_ima_science_interactive
Select group:
SCI frames
Stacked frame
Select item in group :
Image
Assess matched astrom stds
Histogram of matched astrom stds
Assess matched photom stds
Histogram of matched photom stds
Click to advance to next item (if available)
Stacked Image V BRIAN
Stacked Image V Keith
Stacked Image V DAVID
Stacked Image V Tom
Row number of matched standard
Frequency
Med:
Mean:
MAD:
RMS:
STD star zeropoint used to calibrate science frame
Recipe Parameters
vimos_ima_science
Continue Wkf
Re-run Recipe
Help
Disable this window in subsequent runs
Files Info

savecat ☑
savemstd ☑
minphotom 1
chop_crud hardconf_pix
cdssearch_astrom ppmxl

This data belongs to dataset: VIMOS.2016-09-03T09:09:09.376_tpl

Figure 9.10: Example interactive windows for evaluating the results of the `vimos_ima_science` recipe (continued from Figure 9.9). Each layer of the interactive window can be accessed in the top centre box "Select item in group".
Top Left: The $\Delta\alpha * \cos(\delta)$ and $\Delta\delta$ astrometric matches between the catalogue sources and the sources in the VIMOS science field.
Top Right: Histograms and statistics of the astrometric matches between the catalogue sources and the sources in the VIMOS science field.
Bottom Left: The zeropoint magnitudes of the are not determined from sources in the science frame, but are adapted from the associated standard star field.
Bottom Right: The zeropoint magnitudes of the are not determined from sources in the science frame, but are adapted from the associated standard star field.

9.5 Improving Results through Workflow Interaction

The workflow is a convenient way to process data without much effort needed by the user. The default values for the recipes are selected to produce high quality results in most circumstances. However, the automatic nature of the workflow means that a user may not know if the pipeline products are valid. This section describes a few tips to assess if the processed data matches a user's expectations for accuracy and usefulness.

Check the log files. Each recipe will create a log file with information, warnings, and errors that occurred during processing. In several instances, such warning or errors may not cause the workflow to stop. Depending on a users's workflow settings, there may be no messages that appear to tell a user that a warning or error occurred. Users are therefore strongly encouraged to check the contents of every logfile for each recipe. The log files can be found in `$LOGS_DIR/<recipe_name>/esorex.log`.

Check the astrometric calibration. The recipes will attempt to calibrate the astrometry by fitting the location of point sources to a model of the World Coordinate System (WCS). One way to check this is to examine the matched astrometric standards catalogue in the interactive windows. An alternative is to look at the value of the Quality Control FITS header keyword `QC.WCS_RMS`. Values greater than $\approx 0.5''$ may indicate problems with the fitting. Common sources of problems with WCS fitting are a) non-photometric conditions, b) very few standards in the field, c) extremely inaccurate values of the approximate location of the field (RA/DEC keywords from the telescope control system). Large errors in the WCS fit may also lead to strange results for the stacked or tiled images. Another recommendation is to check the value of the mean ellipticity of point sources in the object catalogues `QC.ELLIPTICITY`. The ellipticity of objects in individual images should be very similar to the ellipticity of objects in the stacked images.

Check the photometric calibration. The recipes calculate the fluxes from objects that appear in the photometric calibration catalogue to evaluate a magnitude zero point for each object. These values can be viewed in the interactive windows as described above. An alternative is to check the `QC.MAGZPT` and `QC.MAGZERR` keywords. If a recipe used one or more inappropriate calibration sources, these can be edited out of the matched standard catalogue and fed back into the recipe for re-calibration (but this can only be done outside the `Reflex` environment. Another option using the workflow is to re-run the recipe with a different value of `magerrcut`.

Look at the raw images. On occasion, a detector may misbehave during a readout and produce pathological images that may not be obvious in the pipeline products.

10 Frequently Asked Questions

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

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

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

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

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

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

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

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

- **Can I use different sets of bias frames to calibrate my flat frames and science data?** Yes. In fact this is what is currently implemented in the workflow(s). Each file in a DataSet has a purpose attached to it ([7]). It is this purpose that is used by the workflow to send the correct set of bias frames to the recipes for flat frame combination and science frame reduction, which may or may not be the same set of bias frames in each case.

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

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

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

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

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

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

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

- **How can I add new actors to an existing workflow?** You can drag and drop the actors in the menu on the left of the `Reflex` canvas. Under `Eso-reflex -> Workflow` you may find all the actors relevant for pipeline workflows, with the exception of the recipe executer. This actor must be manually instantiated using `Tools -> Instantiate Component`. Fill in the “Class name” field with `org.eso.RecipeExecuter` and in the pop-up window choose the required recipe from the pull-down menu. To connect the ports of the actor, click on the source port, holding down the left mouse button, and release the mouse button over the destination port. Please consult the Reflex User Manual ([7]) 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 ([7]) for more information.

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

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

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

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

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

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

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

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

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

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

  As indicated in Section 3, in case of major or minor (affecting the first two digit numbers) Reflex upgrades, the user should erase the `$HOME/KeplerData`, `$HOME/.kepler` directories if present, to prevent possible aborts (i.e. a hard crash) of the esoreflex process.

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

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

- **Can I run the VIMOS imaging Reflex workflow without an internet connection?** The VIMOS Imaging pipeline computes an astrometric and photometric solution using external stellar catalogues. Due to the very large size of the 2MASS, PPMXL, and APASS catalogues (∼200 GB total) the preferred method to access these catalogues is via the an internet connection to the CDS. However, if this is not possible there are two solutions. If an internet connection is available a "first" pass through your data can be made in which the stellar catalogues for the areas covered by your images are downloaded. The pipeline will store this data in a local cache. Subsequent runs of the pipeline, even without an internet connection, will then use this cached data. If you have no consistent internet connection, the full catalogue data files can be obtained by contacting ESO. By default, the index files are installed in the same directory as the static calibration data. After installation, *the index files and catalogue data must be placed into another directory (a separate directory for each catalogue.)*. The user is free to chose the name and location of this directory. For example, the `index_casu_2mass_astrom.fits` and `npscXXX.fits` files could be placed in `<installation prefix>/calib/cats/twomass/`.

- **My VIMOS images (both RAW and processed) look odd, with large sections blacked-out. Is this normal?** Yes, and it is generally the result of vignetting due to the VIMOS guide probe entering the field-of-view of one of the detectors. This can be seen in both the raw and processed images (see FIgure 10.1). Since the processed VIMOS Imaging frames (`JITTERED_IMAGE_SCI` are the product of combined, dithered images, the cumulative vignetting can result in some bizarre structures. Note that light can also reflect from the guide probe and cause positive streaks in the VIMOS images.

- **Are external standard star fields the only way to photometrically calibrate VIMOS science images?** At the moment, yes. Both the VIMOS pipeline and the VIMOS `esoreflex` workflow have been designed to use external stellar catalogues to astrometrically and photometrically calibrate its science exposures. However, at the current time there are no visual-band, all-sky, stellar catalogues that go to sufficiently faint limits to calibrate VIMOS science images. A possible candidate may be a future Gaia

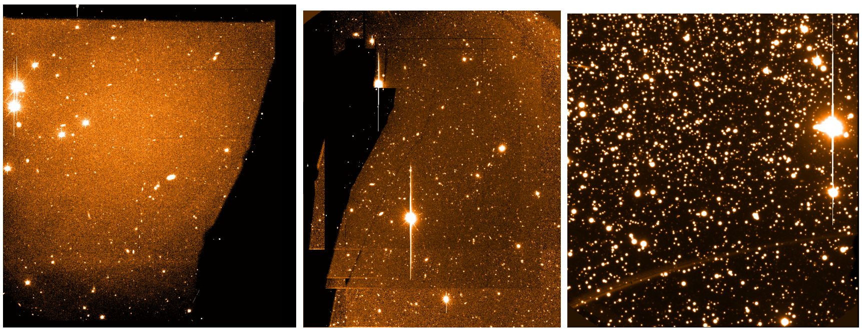


Figure 10.1: Examples of the effect that the VIMOS Imaging guide probe can have on various VIMOS images.
Top Left: A raw VIMOS image (single detector) strongly vignetted by the guide probe.
Top Centre: A processed VIMOS image stacked from multiple frames.
Bottom Right: A processed VIMOS image showing a extended streak caused by light reflected from the guide probe.

data release (see, for example, [4]). Any all-sky photometric catalogue that covers at least several of the VIMOS filters (U, B, V, R, I or z) to a depth of at least 18-19th magnitude would be suitable. There is, however, a significant amount of work necessary to compute the colour transformation terms between the VIMOS filter set and those of the photometric catalogue.

11 Troubleshooting

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

1. **I downloaded data from the ESO archive, put them into a new directory, and tried to run the Reflex workflow. However,**

   (a) **the Reflex execution fails.** This may happen if one of the files was only partially downloaded (check for a file with the extension fits.Z.part. You should remove the partial file and download it again.

   (b) **the Reflex execution fails with error message, "No DataSets have been created, check the data set and the OCA rules."** 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. Alternatively, the directory you have set as your `ROOT_DATA_DIR/RAW` is, in fact, empty.

   (c) **some or all DataSets are greyed-out in the DataSets interactive window.** The greyed-out sets indicate that the data is not complete. By clicking on the affected data set (click on `Inspect highlighted` in the "Select Datasets" window) you can discover which particular data type is missing.

2. **A specific pipeline recipe failed during the Reflex execution.** If this occurs, there are a number of steps you can take to trouble-shoot the cause. In the root directory that you defined on the main Reflex canvas (`ROOT_DATA_DIR`), go to the `reflex_logs` directory. Here you will find a series of directories for each of the VIMOS Imaging components; python interactive actors and pipeline routines. Within each of these directories you will find a timestamp of when the recipe was executed by the Reflex workflow. For the specific pipeline recipe that failed, go to its directory and the directory of the latest timestamp. Here you will find a file called, `esorex.log`. This file is an output of the VIMOS pipeline and contains detailed information about the execution (and possible failure) of this given recipe. For example, if the VIMOS Imaging failed during the flat-fielding process, this file would be located in: `ROOT_DATA_DIR/reflex_logs/vimos/vimos_ima_twilight_flat_1/2020-04-01T13:52:08.445/esorex.log`
   .
   All recipes run through `esorex` will generate this logfile with varying levels of information about what the recipe is doing, warnings, etc. The messages that appear in the logfile are prefixed by a timestamp and one of the following:

   `[ INFO ]` Informational message about what the recipe is doing, e.g. creating output files.

   `[ DEBUG ]` More detailed message about what the recipe is doing, e.g. a statistical description of an intermediate data product. These kind of messages will only appear if a user changes the `esorex` default settings for `msg-level` and/or `log-level` to `debug`. When submitting bug reports, it is recommended that the `debug` option be turned on.

   `[WARNING]` A non-fatal error message that indicates something may have gone wrong during the processing, e.g. not enough photometric standards were found during calibration. A user is advised to check the data products carefully if a `WARNING` is issued.

`[ ERROR ]` A fatal error that indicates a serious problem occurred during the processing and the recipe has stopped, e.g. the recipe configuration file cannot be parsed.

This logfile should be submitted with any bug report.

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

   - Provide a short description of the problem.
   - Provide the error log if available (e.g. from the `reflex_log` directory, or at least the full error message that is reported by the reflex widget.
   - Indicate the recipe affected.
   - Include the full data set list (all ARCFILEs involved in the reduction of your DataSet).
   - If the problem is in the reflex GUI please provide a snapshot.
   - Please specify if the problem concerns data reduction quality or is an unexpected result.
   - Please report your problem to *https://support.eso.org*.

A A Detailed Description of All VIMOS Imaging Data

VIMOS-IMG pipeline data can be separated into four general categories: *raw* files, *static calibration* files, *catalogue* files, and *product* files. Raw files are the unprocessed output of the VIMOS instrument. Static calibration files are a set of mandatory and optional files for various calibration purposes. The catalogue files are optional data used to calibrate photometry and astrometry. Product files are the output of the VIMOS-IMG pipeline processing (as reduced images, master calibration files, object catalogues, etc.). All general categories of files are described in this section.

In preparation for (and during) pipeline processing, all VIMOS-IMG data must be classified into specific categories (e.g. DARK, FLAT, STD, SCIENCE, etc.) These files must then be associated with each other and with one (or more) recipes. This classification and association is done using FITS header keywords. *Reflex* does this classification automatically by using the specially crafted file describing the classification and association rules (provided with the pipeline).

A.1 Raw Data

Each VIMOS-IMG raw FITS file contains a single primary unit with data (image) for one chip. The size of the image depends on the instrument mode; high gain images are 2148 by 4096 pix and low gain images are 2148 by 2440 pix. The EXTNAME keyword identifies the chip name (e.g. `DAVID`). Note that data taken before 7 Aug 2010 (old chips) have chip names `CCD-59A, CCD-60A, CCD-59B,` and `CCD-60B`; the new chips have names `Keith, Tom, DAVID,` and `BRIAN`. Care must be taken to ensure a complete set of four chips are provided to the recipes for processing.

Raw VIMOS-IMG data retrieved from the ESO archive have the standard ESO archive file names, i.e. an instrument identifier followed by a time stamp. The time stamp corresponds to the contents of the FITS header keywords `MJD-OBS` and `DATE-OBS` respectively, i.e. to the date and time when the exposure was taken (a difference of 1 ms between the file name and the contents of the keywords may be present).

The files returned by the archive can be **compressed** using the *fpack* utility; this is indicated by the file name suffix `.fits.fz` instead of the regular `.fits` suffix. These "tile compressed" files may be unpacked using the *funpack* tool distributed as part of the CFITSIO package [7]. The data **must** be uncompressed if the recipes are run within the *Reflex* environment (because the primary header keywords are not otherwise readable). The raw data files included in the *Reflex* tutorial (demonstration data) are *not* compressed.

A.2 Static Calibration Data

The VIMOS-IMG pipeline kit comes with a number of files that are used as part of the calibration of raw data. These files are referred to as "static" calibration files in order to distinguish them from output of recipes that are also used for calibration. Note that the "static" files that come with a particular version of the pipeline may differ from other versions of the pipeline. The default location for the static calibration files automatically set by the *Reflex* installation and is given at the top of the workflow canvas below `Setup Directories` in the `CALIB_DATA_DIR`. A short summary of each of these files is given below.

[7] *https://heasarc.gsfc.nasa.gov/fitsio/*

`vimos_phot.fits`: a binary FITS table containing quantities used to convert the VIMOS-IMG photometric system to/from a reference system (e.g. APASS). There is one extension per system, and one row in each extension for each VIMOS-IMG filter.

`readgain.fits`: a binary FITS table containing reference values for the read noise, gain, and covariance (inter-pixel correlation) of each VIMOS-IMG chip (old and new). These values may be calculated explicitly from any set of data using the `vimos_detector_noise` recipe; those values may then be used in subsequent recipes in the data reduction cascade. If a user chooses not to run `vimos_detector_noise` on their data, the reference values in the static calibration file will be used by other recipes. This reference file was generated in two stages. In the first stage, `vimos_detector_noise` was run separately with raw data from Sep 2008 (old chips) and Mar 2012 (new chips). The noise and gain values in the output files from the recipe are similar to the median values found across all VIMOS-IMG data to date. In the second stage, the two pipeline products were combined using the `merge_readgains.py` script. The provenance of the data for both new and old chips can be found in the `PRO.REC1` and `PRO.REC2` PHU keywords, respectively.

`REF_BIAS_NEW.fits, REF_BIAS_OLD.fits`: multi-extension FITS files containing a *reference* bias image for each VIMOS-IMG detector, one file each for the new and old chips. These files may be used by the `vimos_bias_combine` recipe to compare the recipe product bias image ('master' bias) to a reference bias image. Note that these files are derived from VIMOS-IMG data taken in low-gain (2k by 2k pixels) mode only and should not be used to process 2k by 4k VIMOS-IMG data. They were created using `vimos_bias_combine` with raw data from a 2008 and 2012; the provenance of each file can be seen in the primary headers. After the recipe created these products, the `update_procatg.py` script was used to change the `PRO.CATG` value to `REFERENCE_BIAS` so that these files can be used by Reflex. The statistical distribution of pixel values for the reference files are near the median values among all biases taken with VIMOS.

`REF_DARK_NEW.fits, REF_DARK_OLD.fits`: same as above, except for dark frames created with `vimos_dark_combine`.

`REF_FLAT_XXX.fits`: similar to above, but for flat frames created with `vimos_twilight_flat_combine` and only derived from raw data with the 'new' VIMOS-IMG chips. The value of `XXX` refers to the filter used for the reference image (e.g., `INS.FILT1.NAME`). After the recipe created these products, the `update_ref_flat.py` script was used to place the `INS.FILT?.NAME` in the extension headers into the PHU so they could be used with Reflex. The `update_procatg.py` script was also used to change the `PRO.CATG` value to `REFERENCE_TWILIGHT_FLAT` so that these files can be used by Reflex.

`SFD_dust_4096_ngp.fits`: a FITS image containing the reddening values E_{B-V} in the northern Galactic hemisphere. These values are derived from the dust map described in [6]. The file is used by some VIMOS-IMG recipes for photometric calibration purposes.

`SFD_dust_4096_sgp.fits`: as above, but for the southern Galactic hemisphere.

A.3 Photometric and Astrometric Catalogues

The VIMOS-IMG pipeline can use a number of optional files to calibrate the astrometry and photometry of processed images. **These files are only necessary if a user wishes to process data without an internet connection.** If a user has an internet connection, the VIMOS-IMG recipes can retrieve the required data automatically through the Strasbourg astronomical Data Center[8] (CDS). It is recommended that VIMOS-IMG data use the AAVSO Photometric All-Sky Survey (APASS) catalogue [3] for photometric calibration, and the PPXML [5] catalogue for astrometric calibration.

A short summary of each of these files is given below.

`index_casu_2mass_astrom.fits`: a FITS binary table used by recipes as an index of the 2MASS point sources.

`npscXXX.fits`: a FITS binary table containing information about the 2MASS point sources. There is one file for each degree of right ascension, i.e. npsc000.fits contains objects with $0 < \mathrm{RA} \leq 1.0$.

`index_casu_ppmxl_astrom.fits`: a FITS binary table used by recipes as an index of the PPMXL point sources.

`nppmxlXXX.fits`: a FITS binary table containing information about the PPMXL point sources. There is one file for each degree of right ascension, i.e. nppmxl000.fits contains objects with $0 < \mathrm{RA} \leq 1.0$.

`index_casu_apass_astrom.fits`: a FITS binary table used by recipes as an index of the APASS point sources.

`index_casu_apass_photom.fits`: same file as above; this file is needed to avoid arcane file naming issues in the *Reflex* environment.

`casuapass_XXX.fits`: a FITS binary table containing information about the APASS point sources. There is one file for each degree of right ascension, i.e. casuapass_000.fits contains objects with $0 < \mathrm{RA} \leq 1.0$.

The pipeline kit **only includes the index files** described above. The catalogue data files are rather large and can be obtained elsewhere [TODO: ESO to fill in]. By default, the index files are installed in the same directory as the static calibration data. After installation, *the index files and catalogue data must be placed into another directory (a separate directory for each catalogue.).* The user is free to chose the name and location of this directory. For example, the `index_casu_2mass_astrom.fits` and `npscXXX.fits` files could be placed in `<installation prefix>/calib/cats/twomass/`.

A.4 Pipeline Products

A brief description of the naming and content of pipeline products is given below.

A.4.1 Naming convention for files

The standard method for naming of a data product file for an ESO pipeline is for the product to be given a 'predictable' name. For example, two runs of the same recipe should yield products with exactly the same names. For recipes that generate several products of the same type, each file will have the same root name, but will have a number appended so that each file name is unique. This is the default naming convention for all the recipes in this package.

[8] http://cdsweb.u-strasbg.fr/

For those who would like a more descriptive file name, many of the recipes offer a command line switch called `prettynames`. This system works by creating the output file name from the input file name with an added suffix which denotes the type of product. For example, the recipe `vimos_science_process` produces calibrated exposure files and stacked exposures plus variance images for both. A SoF would include, amongst other things, a list of raw exposures. If the first in the list was called `VIMOS.2011-12-25T02:20:16.502.fits`, then the calibrated exposure and its variance would be called `VIMOS.2011-12-25T02:20:16.502_ex.fits` and `VIMOS.2011-12-25T02:20:16.502_ex_var.fits`, respectively. Stacks are named after the first raw exposure in the SoF, hence this stack and its variance map would be called `VIMOS.2011-12-25T02:20:16.502_st.fits` and `VIMOS.2011-12-25T02:20:16.502_st_var.fits`, respectively.

A.4.2 Images

All images are created as multi-extension FITS files; synchronous images from all four chips are combined into one file. The primary HDU only contains keywords; there is no data array. The primary HDU keywords are derived from the raw images used to create it. The PHU keywords provide general information about the observation; the extension keywords are relevant to the particular detector. If a detector is flagged as 'dead', a 'dummy' data array will still be created for the appropriate FITS extension, i.e. an image product will always have 4 extensions. Variance maps are also created for each calibrated image.

A.4.3 Catalogues

There are two kinds of catalogues created by VIMOS-IMG recipes: *object* catalogues and *matched standard* catalogues.

Object catalogues are FITS binary tables that contain information about statistically significant sources of emission detected in a recipe product image.

Matched standards catalogues are FITS binary tables that contain information about objects in a photometric or astrometric catalogue (§A.3) that were matched to objects detected in a recipe product image. The objects that appear in a matched standard catalogue are those that *may* be used to calibrate the image. For photometric calibration, the only objects that are actually used are those that satisfy the condition set by the `magerrcut` recipe parameter.

A description of the columns in an *object* catalogue is given in Table A.1.

Table A.1: VIMOS-IMG Source Catalogue Tables

Column	Name	Description
1	Sequence_number	Running number for ease of reference, in strict order of image detections
2	Isophotal_flux	Standard definition of summed flux within detection isophote.
3 4 5	X_coordinate X_coordinate_err Y_coordinate	The x, y intensity-weighted isophotal centre of gravity coordinates and errors with (1, 1) defined to be the centre of the first active pixel in the image array.

Continued on next page

Table A.1 – *Continued from previous page*

Column	Name	Description
6	Y_coordinate_err	
7	Gaussian_sigma	Derived from second moment parameters. Equivalence between parameters and generalised elliptical Gaussian distribution is used to derive sigma = $(\sigma_a^2 + \sigma_b^2)^{1/2}$, ellipticity = $1.0 - \sigma_a/\sigma_b$, and position angle = angle of ellipse major axis wrt x axis in degrees
8	Ellipticity	
9	Position_angle	
10	Areal_1_profile	The number of pixels above a series of threshold levels, relative to local sky. The levels are set at T, 2T, 4T, 8T, 16T, 32T, 64T and 128T where T is the analysis threshold (`stk_cat_thresh`). These can be thought of as a sort of poor man's radial profile. Note that for deblended, i.e. overlapping objects, only the first areal profile is computed and the rest are set to -1 (flagging the difficulty of computing accurate profiles). For blended images, Areal profile 8 is used to flag the start of the sequence of the deblended components by setting the first in the sequence to 0.
11	Areal_2_profile	
12	Areal_3_profile	
13	Areal_4_profile	
14	Areal_5_profile	
15	Areal_6_profile	
16	Areal_7_profile	
17	Areal_8_profile	
18	Peak_height	Peak intensity and its error in ADU relative to local value of sky. Equivalent to zeroth-order aperture flux.
19	Peak_height_err	
20	Aper_flux_1	Flux and error within a specified radial aperture where r_{core} =`stk_cat_rcore` and different rows correspond to (0.5, $1/\sqrt{2}$, 1, $\sqrt{2}$, 2, $2\sqrt{2}$, 4, 5, 6, 7, 8, 10, and 12) times r_{core}.
21	Aper_flux_1_err	
22	Aper_flux_2	
23	Aper_flux_2_err	
24	Aper_flux_3	These are a series of different radii soft-edged apertures designed to adequately sample the curve-of-growth of the majority of objects and to provide fixed-sized aperture fluxes for all objects. For example, in $\approx 0.8''$seeing, an r_{core} = $1''$aperture contains roughly 75% of the total flux of stellar images. The aperture fluxes are sky-corrected integrals (summations) with a soft-edge (i.e. pro-rata flux division for boundary pixels). However, for overlapping objects they are more subtle than this since they are in practice simultaneously fitted top-hat functions, to minimise the effects of crowding. Objects external to the blend are also flagged and not included in the large radius summations. All fluxes are corrected for pixels from overlapping neighbouring objects. The fluxes can be combined with (later-derived) aperture corrections for general purpose photometry and together with parameter 18 (peak flux) give a simple curve-of-growth measurement which forms the
25	Aper_flux_3_err	
26	Aper_flux_4	
27	Aper_flux_4_err	
28	Aper_flux_5	
29	Aper_flux_5_err	
30	Aper_flux_6	
31	Aper_flux_6_err	
32	Aper_flux_7	
33	Aper_flux_7_err	
34	Aper_flux_8	
35	Aper_flux_8_err	
36	Aper_flux_9	
37	Aper_flux_9_err	
38	Aper_flux_10	
39	Aper_flux_10_err	
40	Aper_flux_11	
41	Aper_flux_11_err	
42	Aper_flux_12	basis of the morphological classification scheme. Aperture flux 3 is recommended if a single number is required to represent the flux for ALL images - this aperture has a radius of r_{core}
43	Aper_flux_12_err	
44	Aper_flux_13	
45	Aper_flux_13_err	

Continued on next page

Table A.1 – *Continued from previous page*

Column	Name	Description
46	Petr_radius	Petrosian radius r_p in pixels as defined in Yasuda, et al. 2001, AJ, 112, 1104.
47	Kron_radius	Kron radius r_k in pixels as defined by Bertin and Arnouts 1996, A&A Supp, 117, 393.
48	Half_radius	r_h estimate of object half-light radius.
49 50	Petr_flux Petr_flux_err	Petrosian flux and error to $2r_p$.
51 52	Kron_flux Kron_flux_err	Kron flux and error to $2r_k$.
53 54	Half_flux Half_flux_err	flux and error within half-light radius.
55	Error_bit_flag	Number of bad pixels within aperture of radius r_{core}. Note that this can be fractional due to soft-edged apertures
56	Sky_level	Local interpolated sky level from background tracker
57	Sky_rms	Local estimate of variation in sky level around object
58	Av_conf	Average confidence level within r_{core} aperture. Useful for spotting spurious outliers in various parameter selection spaces
59 60	RA Dec	Single-precision (4 byte) RA and Dec of each object in degrees (only accurate to 50mas!). Astrometry should be derived more precisely from WCS in header and XY in columns 5 & 6.
61	Classification	Simple flag indicating most probable classification for object: -2: Object is compact (maybe stellar), -1: Object is stellar, 0: Object is noise, 1: Object is non-stellar. Saturated objects can be flagged by comparing the peak height + local sky with the SATURATE keyword in the header.
62	Statistic	An equivalent N(0,1) measure of how stellar-like an image is. It is used in deriving the classification (column 61) in a ‘necessary but not sufficient’ sense. This statistic is computed from a discrete curve-of-growth analysis from the peak and aperture fluxes and also factors in ellipticity information. The stellar locus is used to define the ‘mean’ and ‘sigma’ as a function of information such that the ‘statistic’ can be normalised to an approximate N(0,1) distribution. See Irwin et al 1994 (SPIE 5493 411) for more details.
63–80	blank	

Converting the source catalogue fluxes (here, using any of the 13 aperture flux values: Aper_flux_1 to Aper_flux_13) to magnitudes can be done with the following relation:

$$magnitude = \mathtt{PHOTZP} - 2.5 \times \log_{10}(\mathtt{Aper\_flux\_i}) - \mathtt{APCORi} \qquad (for\ i = 1, \ldots, 13) \qquad (1)$$

where uppercase parameters indicate the catalogue header keywords:

PHOTZP = the photometric zeropoint (magnitude)
APCORi = the stellar aperture correction for the ith aperture flux (magnitude)

Note that, since the VIMOS image stacks have been averaged to the number of input exposures, all magnitude computations should be done using the effective exposure time keyword, EFF_EXPT.

B Photometric Calibration of Science Frames with VIMOS Standard Stars

The original intent was to use APASS stars in the science field to photometrically calibrate these images. However, with a magnitude limit of about 16.5 - 17.0, even the faintest APASS stars will be saturated in VIMOS images with exposure times greater than about 130 seconds (in the U- band) and about 20 seconds (in all other filters). Since VIMOS science exposures are typically much longer, most of the APASS field stars will be saturated. The median exposure time of VIMOS standard star fields is ≈14 seconds (in the U-band) and ≈2 seconds (in all other filters), well-below APASS saturation levels. Therefore, we have used the APASS standard star catalogues to calibrate the VIMOS standard stars, which in turn, have been used to define the associated science image zeropoints. If a standard star image existed on the same night and in the same filter as the science exposure, the standard star zeropoint (ZP_{std}) was converted to a science image zeropoint (ZP_{sci}) using:

$$ZP_{sci} = ZP_{std} + ext\_coef * (AM_{sci} - AM_{std}) + (APcor_{sci} - APcor_{std}) \tag{2}$$

where:

ext_coef is the filter-specific extinction coefficient (`HIERARCH ESO DRS EXTCOEF`)
AM_{sci} and AM_{std} are the airmass values of the science and standard images, respectively
$APcor_{sci}$ and $APcor_{std}$ are the aperture correction values (from the `APCOR3` keywords) for the science and standard star images, respectively.

In the event that no suitable standard star exposure existed, the zeropoint of the science exposure was defined using the zeropoint median (as measured over several years of standard star data) for that VIMOS filter. These "default" zeropoints were measured for each of the six filters and for each of the two detector sets, in the following way:

- zeropoints were filtered if they deviated by more than ± 1σ from the median zeropoint.
- a linear regression fit was made to the remaining zeropoints.
- zeropoints were again filtered if outside of ± 1σ of the linear fit.
- a final linear regression fit was made to the remaining zeropoints.
- these values are stored in the `vimos_phot.fits` tables in the `CALIB_DATA_DIR`.

This results in a zeropoint fit of the form: $ZP = m * \texttt{MJD} - \texttt{OBS} + b$ (for each filter and detector). The values of this fit are summarised in Table B.1.

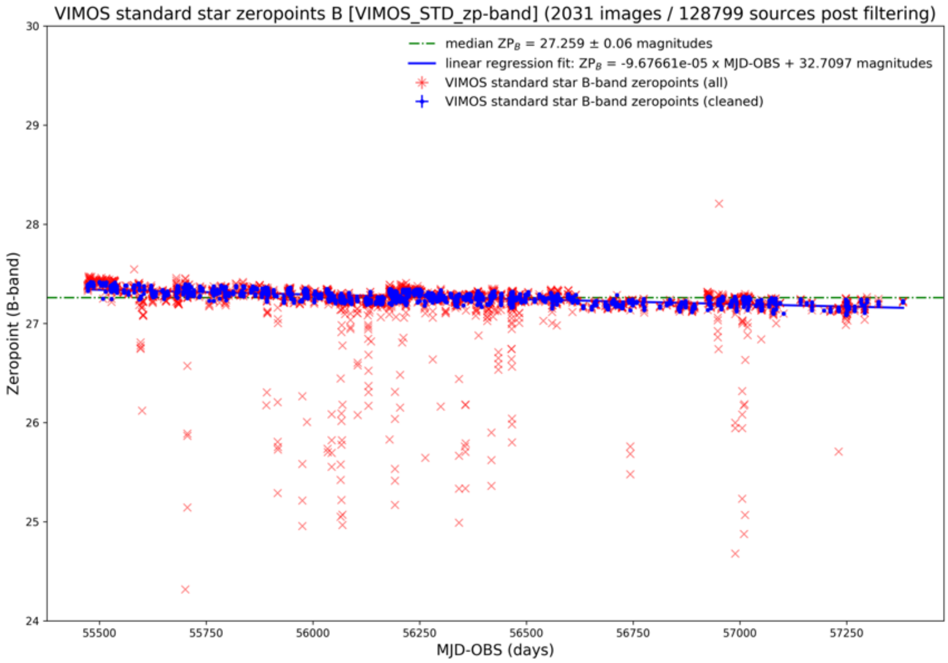


Figure B.1: An example of the full zeropoint history of VIMOS standard stars in the B-band filter (new detectors). All standard star zeropoints are shown as red crosses. The blue points are those zeropoints used to define the zeropoint fit. This fit was repeated for all VIMOS filters and detectors.

Table B.1: A summary of the fits to the cleaned VIMOS standard star zeropoints

VIMOS Standard Star Zeropoint Fits (old detectors)					
Filter	**m** $\times 10^{-5}$	**b**	**median ZP**	**median** AM_{std}	**median** $APcor_{std}$
U	1.2067	25.226	25.870 ± 0.10	1.133	0.301
B	1.9609	26.248	27.310 ± 0.05	1.129	0.264
V	1.2079	26.398	27.049 ± 0.04	1.127	0.228
R	1.2866	26.603	27.300 ± 0.06	1.129	0.210
I	1.2998	26.016	26.722 ± 0.07	1.141	0.186
z	5.3127	22.743	25.622 ± 0.08	1.110	0.212
VIMOS Standard Star Zeropoint Fits (new detectors)					
U	-9.6630	31.054	25.596 ± 0.12	1.126	0.265
B	-9.6766	32.710	27.259 ± 0.06	1.124	0.223
V	-5.1644	29.923	27.013 ± 0.04	1.132	0.205
R	-4.8654	30.123	27.383 ± 0.05	1.128	0.183
I	-3.8051	29.501	27.359 ± 0.06	1.139	0.170
z	-14.330	34.654	26.601 ± 0.10	1.102	0.138

C VIMOS–APASS Colour Transformations

The colour transformations used for the conversion between the VIMOS-IMG photometric system and the system used for photometric calibration (e.g. APASS [3]) can be found in the fits tables (`HIERARCH_ESO_PRO_CATG = PHOTCAL_TAB`) included in the `CALIB_DATA_DIR` distribution of this Reflex workflow. These are divided into two tables defining the transformations for the old VIMOS detector set and the current VIMOS detector set.

For the old VIMOS detectors (data prior to August 1, 2010) the transformations are:

$$U_{vimos} = 3.2 \times g_{apass} - 2.2 \times r_{apass}$$

$$B_{vimos} = 1.0 \times B_{apass} + 0.05 \times g_{apass} - 0.05 \times r_{apass}$$

$$V_{vimos} = 1.0 \times V_{apass}$$

$$R_{vimos} = 1.05 \times r_{apass} - 0.05 \times g_{apass}$$

$$I_{vimos} = 1.16 \times i_{apass} - 0.16 \times r_{apass}$$

$$z_{vimos} = 1.65 \times i_{apass} - 0.65 \times r_{apass}$$

For the current VIMOS detectors (data post August 1, 2010) the transformations are:

$$U_{vimos} = 3.2 \times g_{apass} - 2.2 \times r_{apass}$$

$$B_{vimos} = 1.0 \times B_{apass} + 0.05 \times g_{apass} - 0.05 \times r_{apass}$$

$$V_{vimos} = 1.0 \times V_{apass}$$

$$R_{vimos} = 1.08 \times r_{apass} - 0.08 \times g_{apass}$$

$$I_{vimos} = 0.96 \times i_{apass} - 0.04 \times r_{apass}$$

$$z_{vimos} = 1.65 \times i_{apass} - 0.65 \times r_{apass}$$

[1] Michael L. Balogh, David G. Gilbank, Adam Muzzin, Gregory Rudnick, Michael C. Cooper, Chris Lidman, Andrea Biviano, Ricardo Demarco, Sean L. McGee, Julie B. Nantais, Allison Noble, Lyndsay Old, Gillian Wilson, Howard K. C. Yee, Callum Bellhouse, Pierluigi Cerulo, Jeffrey Chan, Irene Pintos-Castro, Rane Simpson, Remco F. J. van der Burg, Dennis Zaritsky, Felicia Ziparo, María Victoria Alonso, Richard G. Bower, Gabriella De Lucia, Alexis Finoguenov, Diego Garcia Lambas, Hernan Muriel, Laura C. Parker, Alessandro Rettura, Carlos Valotto, and Andrew Wetzel. Gemini Observations of Galaxies in Rich Early Environments (GOGREEN) I: survey description. *Monthly Notices of the Royal Astronomical Society*, 470(4):4168–4185, 06 2017. 15

[2] ESO. *VIMOS-IMG Pipeline Manual*, 4.1.14edition, 2015. VLT-MAN-ESO-XXXXX-XXXX. 11, 12, 28, 38

[3] A. A. Henden, S. Levine, D. Terrell, and D. L. Welch. APASS - The Latest Data Release. In *American Astronomical Society Meeting Abstracts*, volume 225 of *American Astronomical Society Meeting Abstracts*, page #336.16, January 2015. 15, 59, 66

[4] C. Jordi. Photometric relationships between Gaia photometry and existing photometric systems. *University of Barcelona, Technical Note*, 11:1–25, February 2018. 54

[5] S. Roeser, M. Demleitner, and E. Schilbach. The PPMXL Catalog of Positions and Proper Motions on the ICRS. Combining USNO-B1.0 and the Two Micron All Sky Survey (2MASS). *Astronomical Journal*, 139:2440–2447, June 2010. 15, 59

[6] D. J. Schlegel, D. P. Finkbeiner, and M. Davis. Maps of Dust Infrared Emission for Use in Estimation of Reddening and Cosmic Microwave Background Radiation Foregrounds. *Astrophysical Journal*, 500:525–553, June 1998. 58

[7] Forchì V. *Reflex User's Manual*. ESO/SDD/DFS, *http://www.eso.org/gasgano/*, 0.7 edition, 2012. VLT-MAN-ESO-19000-5037. 29, 38, 51, 52

[8] S. Zampieri and V. Forchì. *OCA User Manual*. ESO. VLT-MAN-ESO-19000-4932. 36