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 VISIR Tutorial

VLT-MAN-ESO-19500-....

Issue 1.6

Date 2023-05-17

	Name	Date	Signature
Prepared:	L. Luvaul	2023-05-17	
Approved:	W. Freudling		
Released:	M. Sterzik		

This page was intentionally left blank

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
1.0	2016-02-25	All	First public release
1.1	2016-04-18	All	Public release
1.4	2022-05-19	All	Public release
1.5	2022-05-30	All	Pipeline minor version bump
1.6	2023-05-17	Figures 5.0.2, 5.0.3	Public release

This page was intentionally left blank

Contents

1 Introduction to `Esoreflex` **7**

2 Status of the VISIR Reflex Workflows **8**

3 Software Installation **9**

3.1 Installing `Esoreflex` workflows via `macports` 9

3.2 Installing `Esoreflex` workflows via `rpm/yum/dnf` 9

3.3 Installing `Esoreflex` workflows via `install_esoreflex` 9

4 Demo Data **11**

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

6 Reducing Your Own Science Data **19**

7 Reduced Data Description **20**

7.1 Reduced Data Description for the Imaging Workflow 20

7.2 Reduced Data Description for the Spectroscopic Workflow 21

8 About the main `esoreflex` canvas **23**

8.1 Saving And Loading Workflows 23

8.2 Buttons 23

8.3 Workflow States 23

9 The VISIR Workflow **24**

9.1 Workflow Canvas Parameters 24

9.2 Workflow Actors 25

9.2.1 Simple Actors: Imaging Workflow Steps 25

9.2.2 Composite actors: Imaging Workflow Steps 26

9.2.3 Simple Actors: Spectroscopy Workflow Steps 27

9.2.4 Composite actors: Spectroscopy Workflow Steps 28

9.2.5 Lazy Mode 29

10 Workflow Steps **31**

10.1 Imaging Workflow . . . 31

10.1.1 Workflow parameters . . . 31

10.1.2 Step 1: Data Organization And Selection . . . 32

10.1.3 Step 2: Image Registration and Coaddition . . . 34

10.1.4 Step 3: Optional Photometry . . . 35

10.1.5 Step 5: Calibration and Output Organization . . . 35

10.2 Long-Slit and Cross-Dispersed Spectroscopy Workflows . . . 36

10.2.1 Workflow parameters . . . 36

10.2.2 Step 1: Data Organization And Selection . . . 37

10.2.3 Step 2: Data Decoding . . . 37

10.2.4 Step 3: Distortion Correction . . . 37

10.2.5 Step 4: Wavelength Calibration and Spectral Extraction . . . 37

10.2.5.1 Interactive GUI (“Aperture Editor”) description . . . 37

10.2.5.2 Interactive GUI (“Aperture Editor”) operation . . . 40

10.2.5.3 Two Different Optimal Extraction Methods . . . 41

10.2.6 Step 5: Photometric Calibration and Output Organization . . . 42

11 Optimizing Results Through Workflow Interaction **43**

12 Frequently Asked Questions **44**

13 Troubleshooting **47**

A Abbreviations and acronyms **51**

1 Introduction to `Esoreflex`

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

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

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

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

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

2 Status of the VISIR Reflex Workflows

The VISIR Reflex Workflows were first released immediately after a major instrument upgrade at the end of 2015. Improved workflows were released in 2022: the single spectroscopic workflow that used to process all spectroscopic data was split into separate long-slit and echelle (HRX) workflows, and interactive GUIs were added to allow interactive definition of the apertures used to extract the 1-dimensional spectra. Coronography and burst mode data are reduced by the imaging workflow, but it does not exploit these types of data to their full capacity. Further improvements may be implemented in the future, subject to available resources. The SAM mode is not supported.

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 pipeline is bundled with demo data sets for the imaging and spectroscopic VISIR modes, to run the Reflex workflow and to verify the installation and to experiment with before you start to work on your own data. They contain a few imaging and spectroscopic data sets - each one is an observation of a different target.

The test sets are complete (e.g., they contain the necessary calibrations; some calibrations may be included in the static database that comes with the reflex workflow). They can be processed successfully with the default parameters, but the users are encouraged to experiment changing the parameters before they start to work on their own data.

A minimum of ∼2.5 Gb of free space is needed on the disk where the directories `<download_dir>`, `<install_dir>`, and `<data_dir>` are located. The raw demo files take ∼1.5 Gb. A single reduction of the complete VISIR demo data set will expand the total used space to ∼2.5 Gb, and one must ensure to have more space if further reductions, perhaps with different parameters, are attempted. If the user has limited space, before experimenting make sure to set the global parameter `EraseDirs` to "true" and process not the entire demo data set, but one data set at a time.

The demo data sets are relatively small, and it take a matter of minutes to process them on an average desktop. This may not be the case with a large multi-night set of real science data!

5 Quick Start: Reducing The Demo Data

The application resides in sub-dir install/bin of the installation directory. The full path to this sub-dir should be in the PATH variable (Section 3).

1. Start the `Reflex` application typing at the prompt:

   `esoreflex &`

   or

   `./esoreflex &`

   from the `.../install/bin` sub-directory, if the PATH has not been set to include this sub-directory.

   The empty `Reflex` canvas will appear (Figure 5.0.1).

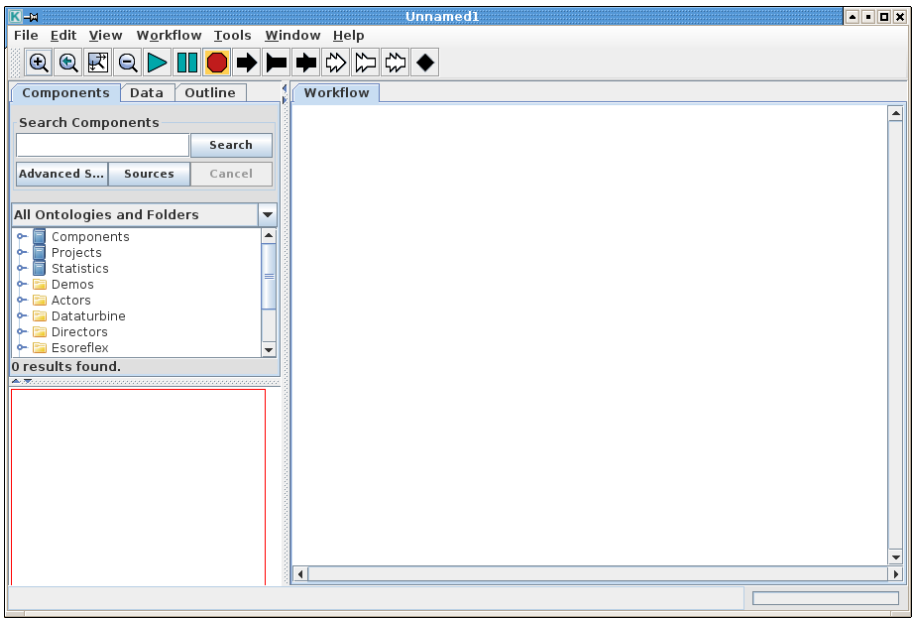


Figure 5.0.1: *The empty* `Reflex` *canvas.*

2. Load the xml file containing the workflow by clicking on `File => Open File`; a pop-up browser window will appear; select sub-dir `visir-4.4.5` (here and later the version numbers may differ), then one of the VISIR workflows (e.g. `visir_img.xml`, `visir_spc.xml`, or `visir_ech.xml`) and press `Open`. The corresponding workflow will appear (see Figs. 5.0.2 and 5.0.3 for examples) after a few seconds. If the window is re-sized, it is advisable to press the `Zoom Fit` button (or press simultaneously Ctrl/Shift/– on the keyboard), so the entire workflow is visible (use Ctrl/+ or Ctrl/– to zoom in or out).

   The canvas provides instructions on how to run/monitor the workflow ("Workflow Instruction", upper-left), how to set the working directories ("Setup Directories", upper-center) and how to set other global parameters ("Global Parameters", upper right). In the lower part of the canvas, the reduction cascade is shown.

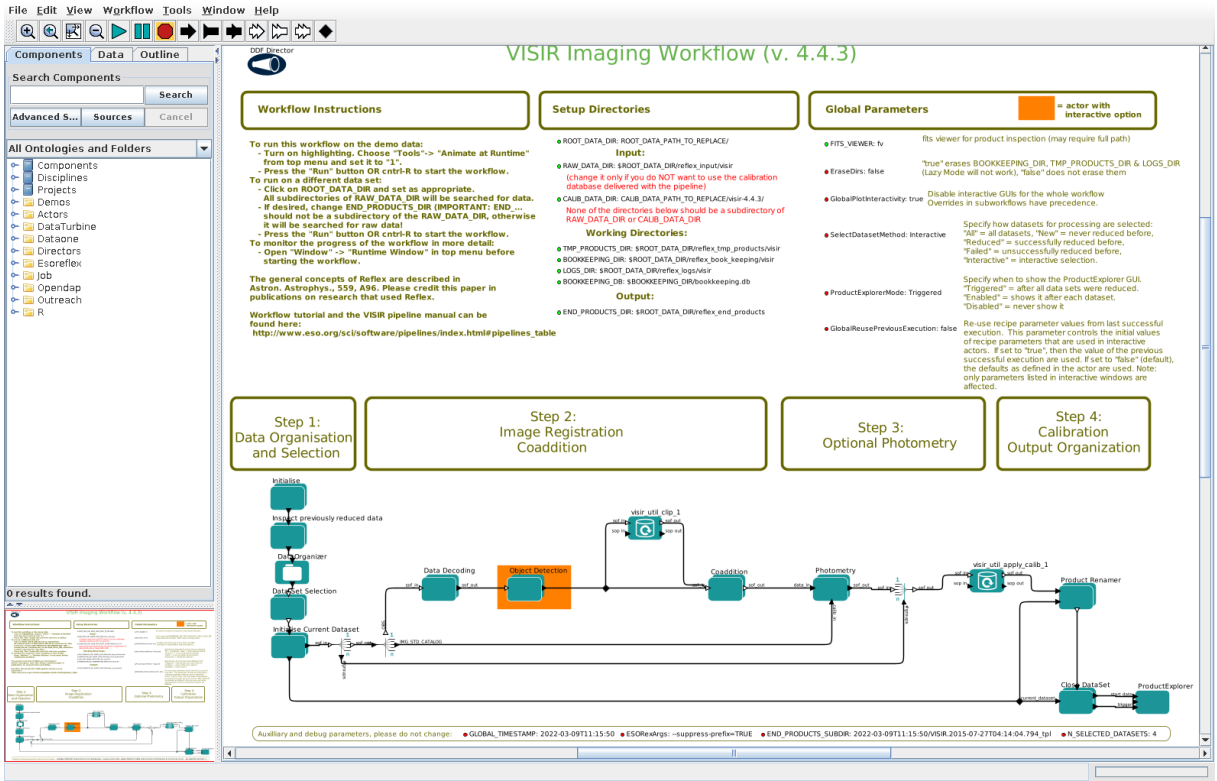


Figure 5.0.2: *VISIR imaging workflow. Newer versions may differ slightly.*

3. Enable the optional highlighting of the currently executed actor. This feature is useful for visual tracking of the reduction cascade progress: in the Workflow window click on `Tools => Animate at Runtime`, and enter the number of milliseconds representing the animation delay (100 ms is recommended), then click `OK`.

4. `Set up directories` in the workflow window (upper center, marked with green dots). To modify the paths double-click on their names (the green dot is highlighted in yellow, and a browsing pop-up window comes up). To set the new directory press the `Select` and `OK` buttons to confirm the new choice.

   For processing the demo data there is no need to change anything – the installation script sets the paths to point at the correct sub-dir where the data is. For any other data sets the only critical sub-dir to update is `ROOT_DATA_DIR`, and may be `RAWDATA_DIR`. Make sure that the `END_PRODUCTS_DIR` is different from `RAWDATA_DIR`.

5. Set up interactivity (only relevant for workflows with interactive actors): to reduce several data sets in a row, one should disable the interactive GUIs by double-clicking on the parameter `EnableInteractivity` in the "Global Parameters" (top right in the workflow canvas); a pop-up window will appear; enter `false` for non-interactive regimen (recommended for repeated processing), or `true` for interactive regimen (recommended for the first processing).

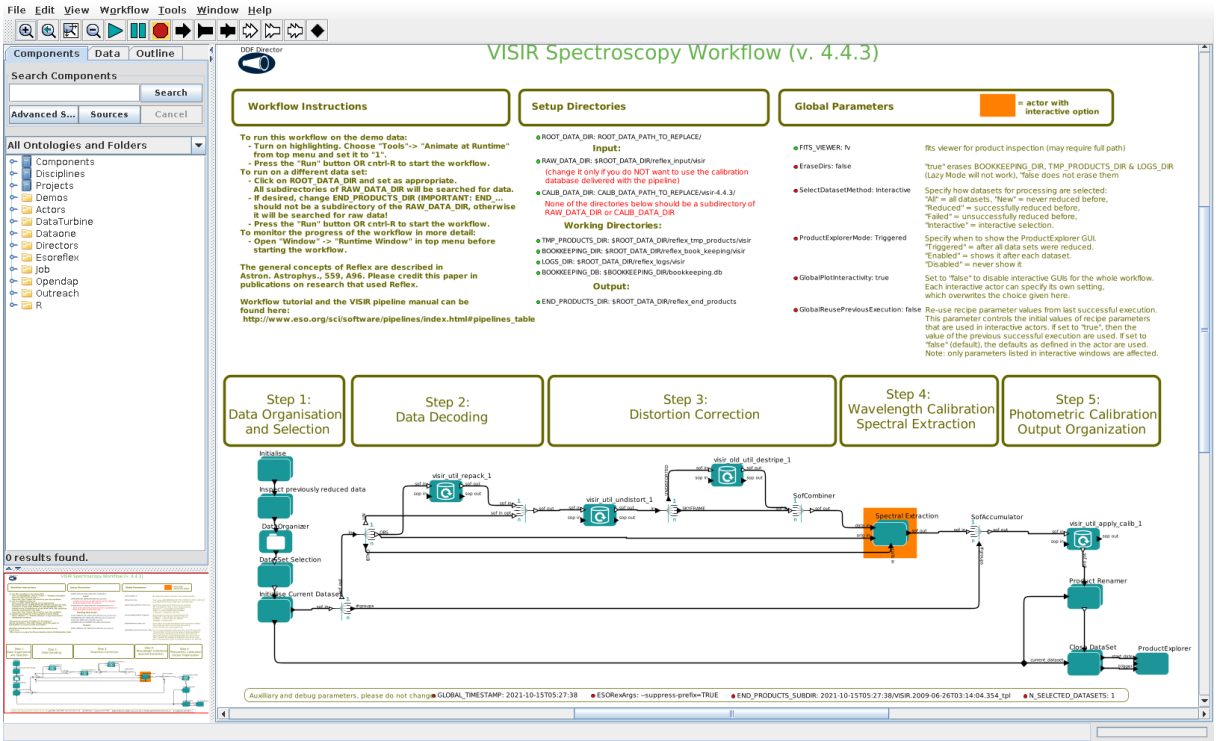


Figure 5.0.3: *VISIR spectroscopic workflow. Newer versions may differ slightly.*

6. Launch the workflow data pressing cascade by clicking on the `Run/Resume` button in the workflow window (or press simultaneously Ctrl/R on the keyboard).

   The `Data Organiser` actor will be highlighted. It scans the raw data directory (specified by the parameter `RAWDATA_DIR`, see step 4) and constructs DataSet(s). The raw and static calibration data must be present either in `RAWDATA_DIR` or in `CALIB_DATA_DIR`, otherwise the DataSet(s) may be incomplete and cannot be processed. A potential problem will arise if the same reference file was downloaded twice in different sub-dirs, and Reflex will not be able decide which one to use.

7. Data selection step: `Data Set Selection` actor is highlighted next, and it opens a pop-up window titled `Select Datasets`, listing the data set(s) identified in the previous step (Figure 5.0.4). Each set is presented on a separate line, that starts with a tick box for user selection (by default the completed, not-reduced sets are selected). Next on the line is the number of files in each data set, followed by the reduction status (e.g. "–" for not processed, "OK" for reduced) and a `View` Nightlog button that allows the user to inspect the night log. Header keywords can be displayed, the list is modified by right-clicking on the `Data Organizer` actor, then selecting the `Configure Actor` option, and editing the keyword list (which is empty by default, so no header keywords are displayed) in the second line of the pop-up window. The data sets can be inspected (with the selected `FITS_VIEWER` in the Global Parameters space), one at a time, bu highlighting a DataSet (click with the left mouse button) and pressing `Inspect highlighted` button. Files can be excluded from a data set by deselecing the corresponding tick mark). Once satisfied, the user must click on the *Continue* button to proceed with the data reduction. The workflow progress is shown in the canvas by red highlighting of the current actor. A message window

titled `Current DataSet` will show which data set is being processed.

The data sets are not selected if the they have already already been processed before (Figure 5.0.5). If the user chooses to re-reduce the data, the data set selection button on the left must be ticket before pressing the *Continue* button. The new products will be stored in a different sub-sub-dir, and will not overwrite the older products.

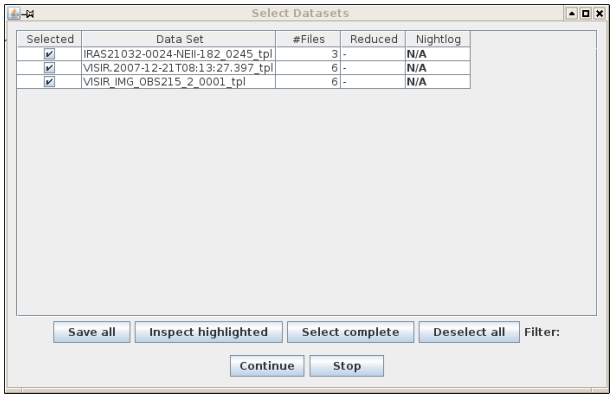


Figure 5.0.4: *The* `Select Datasets` *pop-up window during the first processing of a data set.*

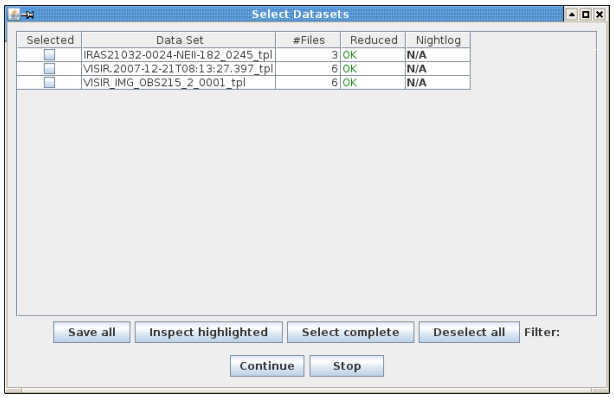


Figure 5.0.5: *The* `Select Datasets` *pop-up window for a re-run (e.g. when the data have already been reduced once and the previous data products are in the same output sub-dir.*

8. Data processing. This step differs for the different VISIR modes. Furthermore, if the data set contains calibration observations, some steps may be repeated to process the calibrations. Once the first data set is complete, a `Finished Datasets` reporting window will pop-up, and it will be updated after a new data set is completed (Figure 5.0.6).

   – imaging: The workflow will perform bad pixel detection and cleaning, source detection, image shifts and coaddition, photometry and flux calibration (if a flux standard is available) of the images. First, the `Object detection` actor will be highlighted; it will open a window shown in Figure 5.0.7. A number of diagnostic measurements and reduction parameters are available to optimize the data reduction

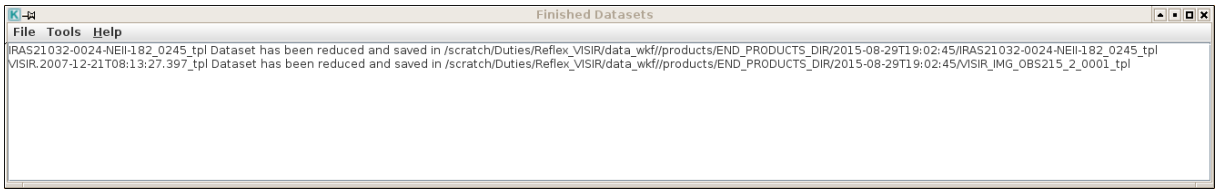


Figure 5.0.6: *Finished Datasets reporting window.*

(see Section 11 for details). IMPORTANT: If the user wants to modify the default parameter values, the actor must be re-executed by pressing the button *Re-run Recipe*. Once the user is satisfied, press the *Continue Wkf* button.

To switch to non-interactive (a.k.a. "lazy") mode, useful for large uniform data sets, on this and other interactive windows tick the *Disable this window in subsequent interactions* button. Note that the following executions will use the default parameter values, discarding any user made modifications.

– long-slit and cross-dispersed spectroscopy: These workflows include an interactive extraction actor with a GUI that lets the user to define up to ten apertures per order/object. The reduction includes bad pixel detection and cleaning, distortion correction of the 2-dimensional images, shifting–and–adding them, 1-dimensional spectra extraction and wavelength calibration. If the DataSet contains a standard, it is reduced the same way. The buttons *Re-run Recipe*, *Continue Wkf* and *Disable this window in subsequent interactions* work the same as for the imaging workflow.

This step is repeated until all data sets are processed. The data products are stored in `END_PRODUCTS_DIR` sub-dir defined in the workflow. The full path can also be found in the `Finished Datasets` reporting window mentioned earlier in this item (Figure 5.0.6).

9. `Product Explorer` (a.k.a. `Provenance Explorer`) window (Figure 5.0.8) opens up when all data sets (listed in the left sub-window) are processed. The actor `Product Explorer` is highlighted. The final products are listed, together with the entire data reduction cascade – all raw and calibration files used in the reduction (central sub-window). The user can see their headers (right sub-window), and open the files in a viewer (buttons `Inspect` and `Inspect with...`). The products can also be visualized with outside tools.

   The end product are stored as extensions in a single FITS file. For imaging it contains the following extensions (all are 2-dimensional): a combined image, an error map, and a weight map. For the spectroscopy the extensions are: a combined 2-dimensional spectrum, a relative 2-dimensional weight map, and a table containing the 1-dimensional spectrum extracted from the 2-dimensional image.

Note that unless the workflow is saved before exiting, any change that the user has made will not be kept if `Reflex` is restarted from scratch. The workflows are stored in sub-dir of the directory where the Reflex is installed:

```
.../install/share/reflex/workflows/visir-4.4.5/
```

The File => Save As option will save the workflow as a Java file with an extention "kar". If you want to save the workflow in a executable xml format, go to File => Export => XML option (Sec. 8).

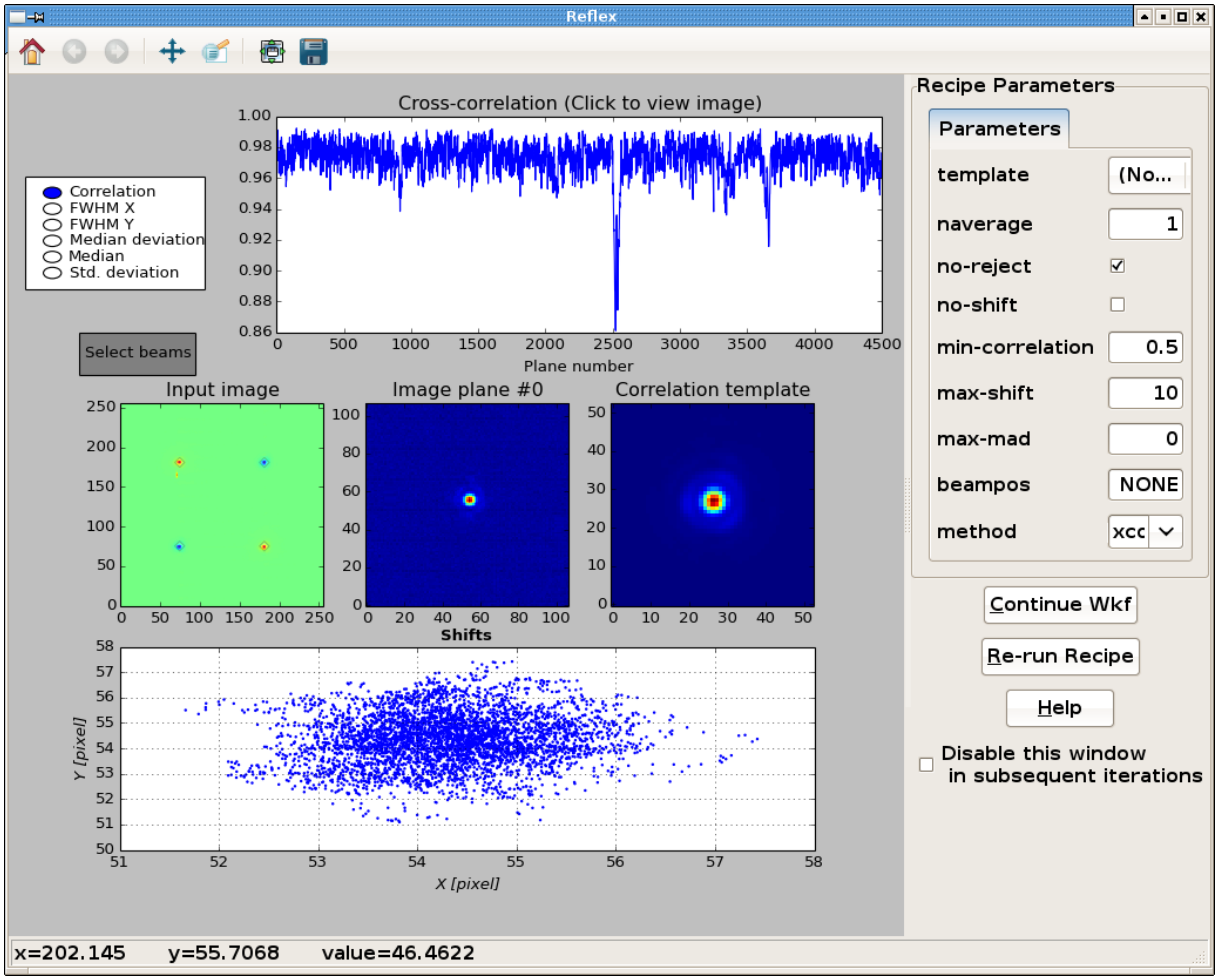


Figure 5.0.7: *The interactive window for the VISIR imaging tt Object Detection actor. On this example a standard star is being processed.*

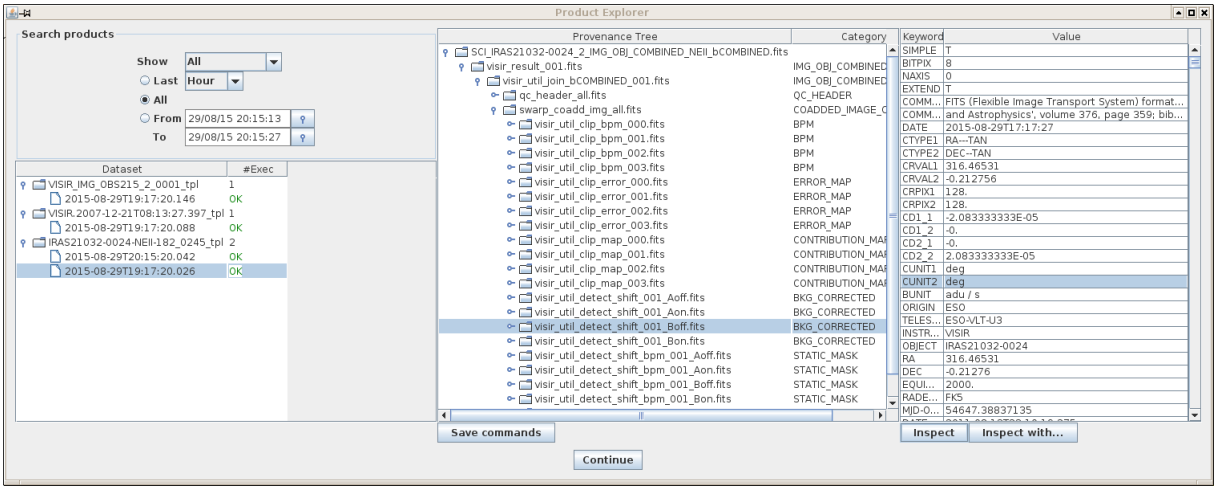


Figure 5.0.8: *The* `Provenance Explorer` *shows all DataSets reduced during the current workflow execution, with the full reduction chain for all products, and the header keywords for a selected (by clicking) FITS file.*

6 Reducing Your Own Science Data

To reduce your own science data, simply change the path to the raw data directory. This is defined at the top of the workflow window, near the center, in the area labeled `Setup Directories`: double click on the `RAW_DATA_DIR`, enter the path to your raw science directory and run the workflow in the same way as it was done for the tutorial demo data.

Note: to avoid conflicts make sure that the end product directory `END_PRODUCTS_DIR` is different from `RAW_DATA_DIR`.

By default the product directory points at the home-dir of the user. This may pose a problem, if the processed data set is large and the home-dir has a volume limit.

7 Reduced Data Description

The VISIR workflows adopt uniform data format across all instrument modes. The reduced data are stored in standard FITS[2] format, in a separate file for each science object. If the DataSet contains a calibration object (e.g., telluric or photometric star), it is stored in a separate FITS file. The various data products (e.g. combined image, bad pixel map, 1-dimensional extracted spectra, etc.) are stored in separate FITS extensions.

7.1 Reduced Data Description for the Imaging Workflow

The reduced science data products are stored in a file named:

`<OBS NAME>_IMG_OBJ_COMBINED_<INS FILT1 NAME>_bCOMBINED.fits`

where the triangular brackets contain the FITS header parameters for the object name and filter, for example:

MWC-300_IMG_OBJ_COMBINED_Q3_bCOMBINED.fits

The reduced photometric standards are named as follows:

`STD_ima_<INS OBS TARG NAME>_<flux>_IMG_PHOT_COMBINED_<INS FILT1 NAME>_bCOMBINED.fits`

where in addition to the entries described above, <flux> is the standard star's approximate flux in the *B10_7* filter (similar to *N* band), in units of *Jy*. For example:

STD_ima_HD178345_9.6Jy_IMG_PHOT_COMBINED_Q3_bCOMBINED.fits

Deviations from these names patterns may occur when processing data from the pre-upgrade VISIR (e.g. before 2014). Some users include for their own convenience in the target name: SCI, CAL, ima, IMG, etc. These will be translated with no changes into the names of the data products.

Each imaging data product file contains four extensions:

- science (or photometric standard) combined and aligned image; if the DataSet contains a spectrophotometric standard, the image is calibrated in $mJy\,px^{-1}$, if not – it is in $ADU\,sec^{-1}\,px^{-1}$ (see the header keyword `BUNIT` for the used units).
- error map; the units are the same as for the combined image; undefined pixels have a value of infinity and the corresponding pixel in the bad pixel map (see below the description of the last extension) has a value of 1)
- weight map, computed from the errors as ω=1/σ^2; strictly speaking this is a duplication, as the weights can easily be calculated from the errors, but errors are convenient to have for further analysis of the images, and the weights are convenient to have for further analysis of the spectra (e.g. if the users want to carry out their own 1-dimensional spectral extraction); for the sake of uniformity both pieces of data were retained for all workflows.
- bad pixel map with value of 0 for good pixels and 1 for bad pixels. Do not confuse this bad pixel mask, which is derived for a given data product, with the bad pixel mask for the detector, which a input

[2]Flexible Image Transport System; `http://fits.gsfc.nasa.gov/`

file, normally taken from the static calibration database that comes with the pipeline. This mask may be entirely empty (e.g. full of zeros), even though the detector contains bad pixels, because the chopping/jittering/nodding could ensure that all parts of the sky has been imaged by at least one good pixel during the entire observing sequence – this will occur if the chopping/jittering/nodding is large enough. If the chopping/jittering/nodding is not large enough, there may be bad pixels near the edges. Finally, if there is no chopping/jittering/nodding at all, e.g. in stare mode observations or if there is a single image, all bad pixels on the detector will translate directly into bad pixels in the data product – this is a hypothetical case, in observations like this there will be severe problems with the sky subtraction, that will probably render the data useless.

7.2 Reduced Data Description for the Spectroscopic Workflow

The reduced science data products are stored in a file named:

`<OBS NAME>_<PRO CATG>_<INS RESOL>.fits`

where the triangular brackets contain the FITS header parameters that are used to form the name, for example:

NGC1068-SPC-V1_SPC_OBS_LMR_TAB_LR.fits

Here `<OBS NAME>` is equal to `NGC1068-SPC-V1`, explicitly listing the instrument mode; `<PRO CATG>` is equal to `SPC_OBS_LMR_TAB`; and `<INS RESOL>` is equal to `LR`.

The reduced telluric standards are named as follows:

`Telluric_Standard_<PRO CATG>_<INS RESOL>.fits`

where the triangular brackets contain the same FITS header parameters as above. For example:

`Telluric_Standard_SPC_PHOT_TAB_LR.fits`

Here `<PRO CATG>` is equal to `SPC_OBS_LMR_TAB`, and `<INS RESOL>` is equal to `LR`.

Deviations from these names patterns may occur when processing data from the pre-upgrade VISIR (e.g. before 2014).

Each spectroscopic data product file contains four extensions:

- science (or photometric standard) combined and aligned image; if the DataSet contains a spectrophotometric standard, the image is calibrated in $Jy\,px^{-1}$, if not – it is in $ADU\,sec^{-1}\,px^{-1}$.
- error map; the units are the same is for the combined image; undefined pixels have a value of infinity and the corresponding pixel in the bad pixel map (see below the description of the last extension) has a value of 1)
- weight map, computed from the errors as ω=1/σ^2; strictly speaking this is a duplication, as the weights can easily be calculated from the errors, but errors are convenient to have for further analysis of the images, and the weights are convenient to have for further analysis of the spectra (e.g. if the users want to carry out their own 1-dimensional spectral extraction); for the sake of uniformity both pieces of data were retained for all workflows.

- bad pixel map with value of 0 for good pixels and 1 for bad pixels; it may be entirely empty, even though the detector contains bad pixels, because the chopping/jittering/nodding could make it that all parts of the sky has been observed by at least one good pixel during the entire observing sequence. Do not confuse this bad pixel mask, which is derived for a given data product, with the bad pixel mask for the detector - see for details the explanations at the end of Sec. 7.1 above.

8 About the main `esoreflex` canvas

8.1 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.

8.2 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.

8.3 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.

9 The VISIR Workflow

The VISIR 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; [4]).

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 9.2.5), reusing the previous pipeline recipe outputs if input files and parameters are the same as for the previous execution, which saves considerable processing time. Sometimes it is desirable to set the `EraseDirs` parameter to `true`, which forces the workflow to recursively delete the contents of the directories specified by `BOOKKEEPING_DIR`, `LOGS_DIR`, and `TMP_PRODUCTS_DIR`. This is useful for keeping disk space usage to a minimum and will force the workflow to fully re-reduce the data each time the workflow is run.

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

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

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

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

9.2 Workflow Actors

Help on the individual actors is available from within the workflow canvas. Perhaps, the easiest way to get it, is to type the actor's name in the search window, then right-click on the actor's name, and select from the pop-up menu the *View Documentation* option. This will open a new window with short description of the actor's function, parameters and ports. The java source can also be inspected clicking on the *Java Source* button.

The yellow background on some of the images below means that the actor was selected on the workflow (by left-button clicking on it) when the screen shot was taken, it is not a special functional or other designation. The orange background, on the other hand, indicates that the actor is interactive.

9.2.1 Simple Actors: Imaging Workflow Steps

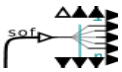
Simple actors have workflow symbols that consist of a single (as opposed to multiple) green-blue rectangle. They may also have a logo within the rectangle to aid in their identification.

Access to the parameters for a simple actor is achieved by right-clicking on an actor and selecting `Configure Actor` from the pop-up menu. This will open an `Edit parameters` window.

The following actors in the imaging workflow are simple actors:

-

- The `Data Organiser` actor classifies and organizes all raw files and products in a given directories and its sub-directories and produces a set of datasets (List of Science Observations) that can be processed by other actors. The logic used to generate the datasets is described using OCA[3] rules: OCA is a SQL-like language developed at ESO to define associations through files based on certain FITS keywords.

- 

- The `Fits Router` actor routes files to different outputs, base on the files' categories.

[3]Rules which classify the raw data according to the contents of the header keywords, organise them into appropriate groups for processing, and asscociate the required calibration data for processing. They can be found in the directory `<install_dir>/share/esopipes/<pipeline-version>/reflex/`, carrying the extension *.oca* – they are editable, but it is not recommended, except for expert users.

-

- The `visir_util_clip_1` actor calls the `visir_util_clip` recipe that performs clipping according to the kappa-sigma clipping algorithm on the chop-nod corrected data and determines the background noise.

-

- Actor `visir_util_apply_calib_1`. If the data set contains a science observation and an associated calibration standard star observation then this actor will call the recipe `visir_util_apply_calib` that will apply the determined conversion factor to the science data. If no calibration observation can be associated with the science data, this step is skipped.

-

- The `Product Explore` (a.k.a. `Provenance Explorer`) actor allows the user to browse all the input and output FITS files, to visualize the calibration cascade and the intermediate products.

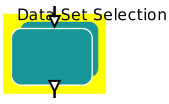
Access to the parameters for a simple actor is achieved by right-clicking on the actor and selecting `Configure Actor` from the pop-up menu. This will open an `Edit parameters` window. Note that the `Product Renamer` actor is a *jython* script (Java implementation of the Python interpreter) meant to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

9.2.2 Composite actors: Imaging Workflow Steps

The composite actors actors are in effect sub-workflows, allowing for a hierarchical workflow structure. The users can see their content by right clicking on them, and selecting the *Open actor* option from the pop-up menu (alternatively, click on the actor and press Ctrl+L). This will open a new Reflex canvas with the content of the composite actor.

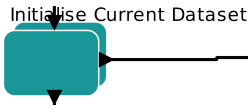
-

- The `Initialise` actor deletes old directories, if requested, and provides the value of the raw data directory to the rest of the workflow. This actor must be the first in a workflow.

- 

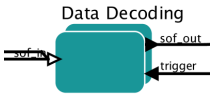
- The `Data Set Selection` actor is a wrapper for the simple actor `Data Set Chooser`

which allows the user to view and select which DataSets created by the DataOrganizer to process.

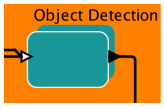
- 

- The `Initialise Current Dataset` actor contains a sub-workflow that displays in a window the name of the currently processed DataSet, sets timestampts, and makes a sub-

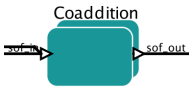
directory where the files for this DataSet will be stored. It also conveys the DataSet further along the workflow.

- 

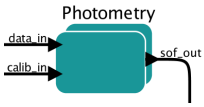
- The `Data Decoding` actor contains the `visir_util_repack` recipe which decodes the raw data into into separate image files. This is needed because the original raw input files may be in a format, that is not suitable for the various data processing steps, so the format is modified. The repacking includes the extraction of single exposures into separate files and optionally applying of chop/nod background subtraction where the chop/nod pairs of files are subtracted from each other. The resulting files in the imaging workflow contain these chop-nod corrected images for each exposure in a multi extension FITS file.

- 

- The `Object Detection` actor contains the `visir_util_detect_shift` recipe which handles object detection. The parameters can be configured with a GUI. Further information on the output of the recipe can be found in the VISIR pipeline manual ([2]).

- 

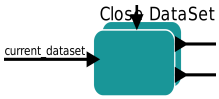
- The `Coaddition` actor contains the `visir_util_run_swarp` recipe which shifts-and-adds individual images via the Astromatic SWARP program [1] using the previously determined object positions. In the VISIR imaging workflow it also executes a recipe for quality control.

- 

- The `Photometry` actor contains the `visir_img_phot` recipe which calculates the sensitivity and ADU–to–mJy conversion factor from standard star calibration observations. It is skipped for science object observations.

-

- The `Product Renamer` actor moves (or copies or links) its input sof to a final product directory based on some FITS keywords. This actor is a *jython* script (Java implementation of the Python interpreter) meant to be customized by the user (by double-clicking on it).

- 

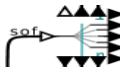
- The `Close Dataset` actor contains a sub-workflow that takes care of the final tasks in the data reduction chain: creating a README file with the list of files, displaying a window with the finished DataSets, and creating an input for the `Product Explorer`.

9.2.3 Simple Actors: Spectroscopy Workflow Steps

The following actors in the spectroscopic workflow are simple actors:

-

- The `Data Organiser` actor – identical to the corresponding actor in the imaging workflow (see Section 9.2.1), except for the different OCA rules file, which contains rules for the spectroscopic mode.

- 

- The `Fits Router` actor routes files to different outputs, base on the files' categories.

-

- The `visir_util_repack_1` actor executes the `visir_util_repack` recipe. It has similar function to the corresponding repacking actor in the imaging workflow (see Section 9.2.2) – to modify the format of the input files, copying them into single files, containing each the single chop and nod cycles. This format is better suited for the next data reduction steps. The images, however, are not chop-nod corrected yet, as one requires a raw sky background image to do the wavelength calibration.

-

- The `visir_util_undistort_1` actor calls the `visir_util_undistort` recipe which corrects the 2-dimensional spectra for distortion caused by the optical components before the detector. The distortion map is part of the static calibration data base. It also extracts and undistorts a skyframe for the wavelength calibration and chop-nod corrects and co-adds the images.

-

- The `visir_old_util_destripe_1` actor removes the stripes from the 2-dimensional spectra running the recipe. Destriping is only applied to old DRS detector data, a destriping method for the new Aquarius detector has not yet been developed. `visir_old_util_destripe`.

-

- Actor `visir_util_apply_calib_1`. If the data set contains a science observation and an associated calibration standard star this actor will call the `visir_util_apply_calib` recipe that will apply the determined conversion factor to the science data. If no calibration observation can be associated with the science data, this step is skipped.

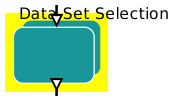
-

- The `Product Explore` (a.k.a. `Provenance Explorer`) actor allows the user to browse all the input and output FITS files, similar to the corresponding actor in the imaging workflow (Section 9.2.1.

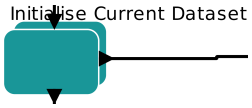
9.2.4 Composite actors: Spectroscopy Workflow Steps

-

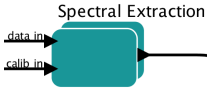
- The `Initialise` actor is simiar to the corresponding actor in the imaging workflow (Sec. 9.2.2. It deletes old directories, if requested, and provides the value of the raw data directory to the rest of the workflow. This actor must be the first in a workflow.

- 

- The `Data Set Selection` actor is simiar to the corresponding actor in the imaging workflow (Sec. 9.2.2. It allows the user to view and select which DataSets created by the DataOrganizer to process.

- 

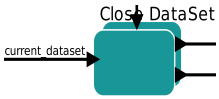
- The `Initialise Current Dataset` actor is simiar to the corresponding actor in the imaging workflow (Sec. 9.2.2.

- 

- The `Spectral Extraction` actor contains the `visir_old_spc_obs` recipe which executes the optimal extraction and wavelength calibration on the coadded image. If a standard star observation is provided, then it will also calculate the sensitivity.

-

- The `Product Renamer` actor is simiar to the corresponding actor in the imaging workflow (Sec. 9.2.2.

- 

- The `Close Dataset` actor is simiar to the corresponding actor in the imaging workflow (Sec. 9.2.2.

9.2.5 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.*

10 Workflow Steps

This tutorial describes the VISIR workflow canvas for imaging, echelle, and spectroscopic observations. They have similar structure (also, similar to the structures of other workflows that process the data of the ESO instruments): from top-left to top-right you will find areas with general workflow instructions (left), with setup directory parameters (center), and with some global parameters (right). The middle row contains boxes labeling the workflow general processing steps (four for imaging and five for spectroscopy), in sequential order from left to right. The workflow actors themselves are organized below the labels, following the workflow general steps. Finally, some auxiliary and debug parameters are located in a separate box, at the very bottom of the workflow. They should not be modified.

10.1 Imaging Workflow

This section describes the imaging workflow in `visir_img.xml`.

10.1.1 Workflow parameters

The "Setup Directories" define the sub-directory structure where the various types of data are stored. The default values point at the location of the demo data, supplied with the `Reflex` workflows.

The user can modify all parameters, but the only ones that really need to be set up are `ROOT_DATA_DIR`, pointing at the root of the sub-directory structure (containing the raw/temporary/reduced data, and the reduction logs), and `RAW_DATA_DIR`, pointing at the location of the ray data (and it is a sub-directory of `ROOT_DATA_DIR`). The data can be located in sub-directories of `RAW_DATA_DIR`, because the `Data Organiser` actor scans recursively the content of `RAW_DATA_DIR` and all of its sub-directories for input raw data. Sub-directories located elsewhere, and pointed to by links located in `RAW_DATA_DIR`, are also scanned.

The directory `CALIB_DATA_DIR` points by default at the pipeline installation directory with the static calibrations supplied with the installation. This directory is scanned by the `Data Organiser` actor as well, 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, data reduction logs, temporary products, and end products are stored, respectively (see the Reflex User Manual for further details; [4]).

The "Global Parameters" is the area of the workflow canvas, where the user can set parameters used throughout the entire workflow. The `FITS_VIEWER` specifies the user's favorite application for inspecting FITS files. The default is `fv`, but other applications, such as `ds9`, `skycat`, and `gaia`, among others, may be selected, as long as they are installed on the host system[4].

`EraseDirs` determines whether to delete (if the parameter is set to `true`) or not (if it is set to `false`) the sub-directories `BOOKKEEPING_DIR`, `TMP_PRODUCTS_DIR`, and `LOGS_DIR` every time the workflow is run. The default value is `false`, which means that **no** directories are cleaned before executing the workflow, and the recipe actors will work in Lazy mode (see Section 9.2.5), reusing the previous pipeline recipe outputs

[4] In some cases specifying the full path to the application may be necessary.

in cases where the input files and parameters were the same as for the previous execution, saving considerable processing time. If any parameter or file (including the calibration files) has been changed, the workflow will rereduce the data, and ignore the content of these directories, that remained from previous execution(s). Setting `EraseDirs` to `true` and erasing these directories is useful for lowering the disk space usage, but this will force the workflow to fully rereduce the data each time the workflow is run, so more time will be needed to reprocess the data.

`EnableInteractivity` disables the interactive GUIs for the entire workflow, if it is set to `true` (the default value) – this must be the case if the user wants to run the workflow in Lazy More. Note that local parameters in sub-workflows have precedence over the global parameter, and override it. This parameter is present only in workflows with interactive actors, e.g. it is in the VISIR imaging workflow, but not in the VISIR spectroscopic workflow.

`SelectDatasetMethod` specifies how the DataSets for processing are selected among all DataSets in the `RAW_DATA_DIR` directory. The options are: "All" - to select all DataSets; "Reduced" – to select only the DataSets that have successfully been reduced before (useful if a small subset of all DataSets is being used to experiment with the data reduction parameters); "Failed" – selects only the data sets for whoch the data reduction cascada has failed for some reason (e.g. they may have faint objects that evaded recognition and the alignment failed); "Interactive" – for interactive selection.

`ProductExplorerMode` controls when you want to see the `ProductExplorer` GUI. The options are: "Triggered" – shows it after all data sets have been reduced (default value, suitable for Lazy Mode); "Enabled" – shows it after each DataSet; "Disabled" – never shows it.

10.1.2 Step 1: Data Organization And Selection

As seen from the name, during the first step of the reduction the raw data directory content is scanned to split it into DataSets, and the necessary calibrations are found and associated with the science files that need them.

The first actor `Initialise` clears any previous reductions.

The `Data Organiser` is a crucial component of any Reflex workflow. It takes as input `RAWDATA_DIR` and `CALIB_DATA_DIR` and it detects, classifies, and organizes the files in these directories and any subdirectories.

The output of the `Data Organiser` is a list of "DataSets". A DataSet is a specific Set of Files (SoF), containing one or several science (and/or calibration) files that should be processed together, and all other files (e.g., other calibrations) needed to process these data. The latter include any calibration files, and in turn files that are needed to process these calibrations. Different DataSets might overlap, i.e. some files, typically various calibrations, may be included in more than one DataSet.

A DataSet lists three different pieces of information for each of its files:

(1) the file name (including the full path);

(2) the file category, such as `IM_CAL_FLAT`, `IM_OBS_CHO_NOD_JIT`, `SPEC_OBS_LMR`, etc. These are the same categories that experienced users may be familiar with from previous encounters the traditional ESO pipeline interfaces *Gasgano* and *ESOrex*.

(3) a string that is called the "purpose" of the file, identifying the reason why this file is included in a DataSet. The syntax is: *action_1.action_2.action_3. ... .action_n*, where each *action_i* describes an intended processing

step for this file. The actions are defined in the OCA rules and contain the recipe together with all file categories required to execute it (and predicted products).

The `Data Organiser` uses `OCA` rules to identify which files must be included in a DataSet, and to determine their categories and purposes.

Next, the `Data Set Chooser` displays the available DataSets in the `Select Datasets` window[5] (see Figure 5.0.4), activating a vertical scroll bar on the right if necessary (e.g. if the DataSet list is too long). 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/or the buttons `Select All` and `Deselect All` at the bottom.

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 (see Figure 10.1.1) will appear that lists the set of files that make up the highlighted DataSet including the full filename and path for each file, the file category (from the FITS header), and a selection tick box in the second 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 or a science frame is of poor quality (e.g. a poor raw flat-field frame, a science frame taken while the telescope guiding chopping failed). 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 window that appears.

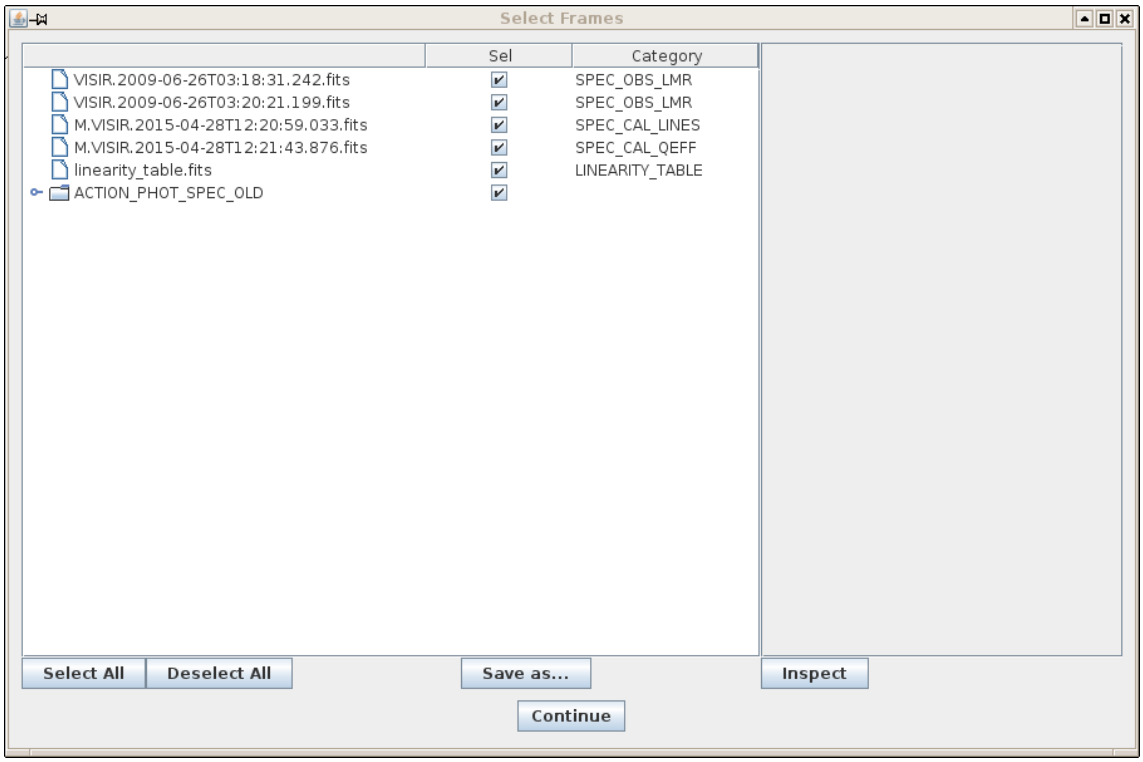


Figure 10.1.1: *The* `Select Frames` *pop-up window during processing of a data set.*

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

Furthermore, clicking on the line corresponding to a particular file in the "Select Frames" window, the file will be highlighted in blue, and the file FITS header will be displayed in the text box on the right (see Figure 10.1.2), allowing a quick inspection of useful header keywords. If you then click on the "Inspect" button at the bottom while the file is highlighted in blue, the workflow will open the file in the selected FITS viewer application defined by the global workflow parameter `FITS_VIEWER`. To exit from the "Select Frames" window, click on the "Continue" button, and to exit from the "Select Datasets" window, click either on the "Continue" in order to continue with the workflow reduction, or on "Stop" in order to stop the workflow.

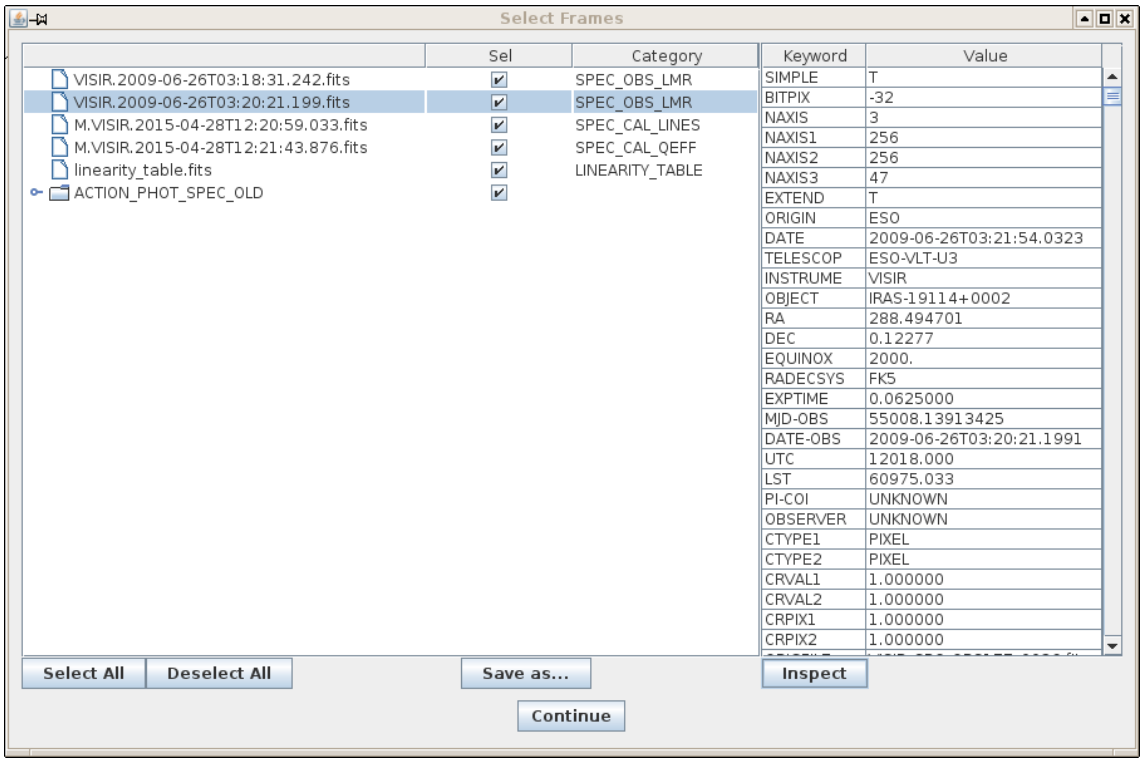


Figure 10.1.2: *The* `Select Frames` *pop-up window during processing of a data set, with a selected fits file, and a display of its header on the right.*

10.1.3 Step 2: Image Registration and Coaddition

The input data is first split – the science images a if present the photometric standards are sent to be decoded, a step that transformed the input FITS files into a format (still FITS) that makes the subsequent processing easier. The relevant entries from the standard star catalog are sent to the actor that will eventually perform the photometry on the processed images.

The object detection is critical for the image alignment. It is the only step that allows interactivity in this workflow, because if the objects are faint, they may be hard to identify automatically. If the workflow is not run in interactive mode, then the user can refine the detection parameters on an interactive GUI (Figure 5.0.7), or to manually mark the position of the objects on individual frames. The left image on the middle row in the gray area of the GUI shows the average of all chop-nod corrected images in the raw data without any correction

applied. The diamonds in this image show the detected pivot beam positions. These positions do not need be exact, they are used just to determine the region in the image where a beam/object is present.

If the beam/object detection fails, then the button "Select beams" just above/left from this window will be colored red, an then the user can attempt to manually define the centers by clicking on the bottom – this will open a new pop-up window (Figure 10.1.3) where the user can select beam/object positions by clicking. If the sign of the beam/object cannot be automatically determined the algorithm will ask in another pop-up window if the last defined beam/object position was positive or negative (the "positive" beams are marked with ticks in the list on the top right). The windows that are used to calculate the object position can be re-sized – to do that hold the mouse button pressed and move the mouse. Selected beams can be removed again by using the right mouse button. It may be useful to practice this exercise before on one of the training sets, before attempting to process your data. More details about this step are available in the VISIR pipeline manual and the "Help" button in the window.

When changes to the parameters have been made one must rerun the recipe in order for them to take effect.

A sky background frame will be create in this step, after applying a kappa-sigma clipping algorithm on the chop-nod corrected data, and finally the sky-subtracted images will be aligned with the appropriate offsets and co-added together into a final image.

10.1.4 Step 3: Optional Photometry

The photometric calibration is optional – it is carried out only if the input DataSet contains a photometric standard. Only the final combined image is calibrated, if you want to study the variability of your source during the observations, split the data into separate DataSets, process them separately, and they will be calibrated individually.

The stellar flux is corrected for the background, and the error budget includes the sky error. Aperture, that maximizes the signal-to-noise ratio is used. The measured standard star flux is compared with the flux from a catalog, in Jy. The final reported sensitivity is computed in units of mJy at 10σ level in 1 hr (the conversion factor is the reciprocal this value). For more details see the VISIR pipeline manual.

10.1.5 Step 5: Calibration and Output Organization

If the previous step yielded a conversion factor, then the final fits are converted from $ADU\ sec^{-1}$ into physical units Jy.

After the processing of the selected DataSet is completed, the `Product Explorer` actor allows the user to inspect the results and select which files to save – according to the value of the global parameter `ProductExplorerMode`.

The `Product Renamer` actor copies the defined final products of the VISIR pipeline recipes to the directory specified by `END_PRODUCTS_DIR` and rename them with names derived from the values of certain FITS header keywords. See Section 7 for description of the file names and their content.

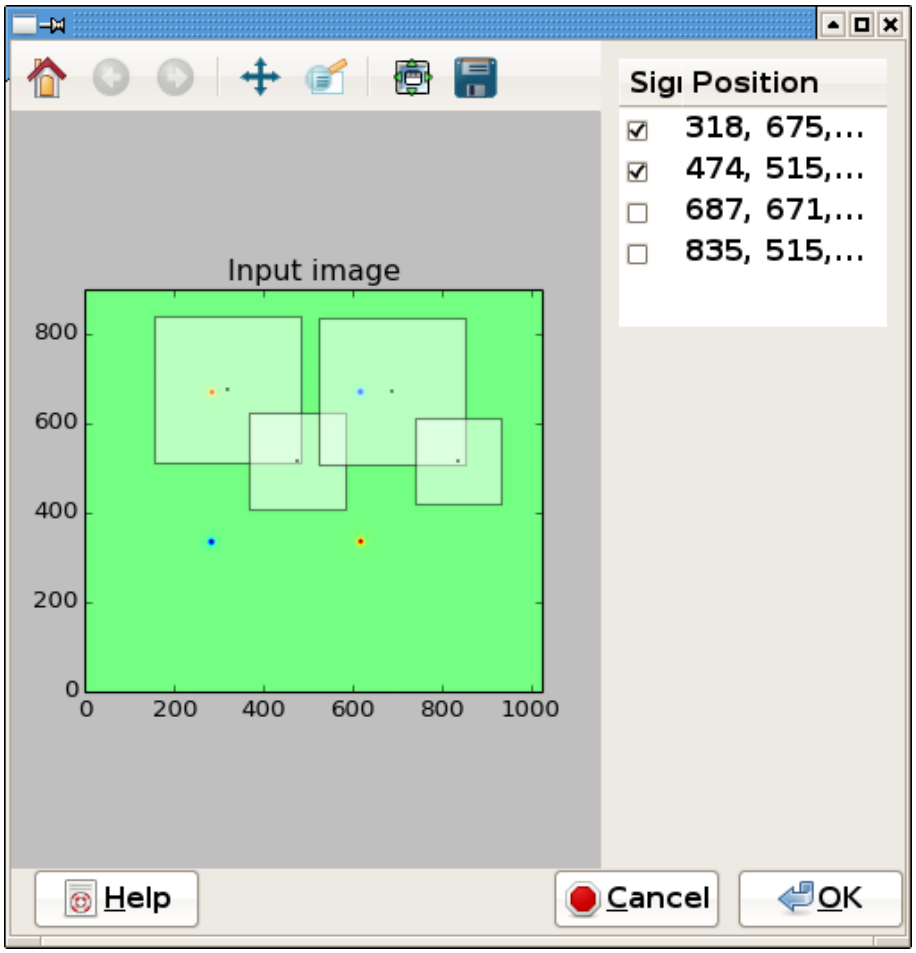


Figure 10.1.3: *The pop-up window for manual object/beam centering. Four beams have been already selected. The "positive" beams are marked with ticks in the list on the top right.*

10.2 Long-Slit and Cross-Dispersed Spectroscopy Workflows

This section describes the long-slit and cross-dispersed spectroscopy workflows in `visir_spc.xml` and `visir_ech.xml`, respectively. These workflows are identical in every way except for the OCA rules that select different data to feed into them.

10.2.1 Workflow parameters

The "Setup Directories" section for the spectroscopic workflow is identical with that for the imaging workflow, and we direct the reader to Sec. 10.1.1.

The available parameters are the same as for the imaging workflow. Again, their description is given in Sec. 10.1.1.

10.2.2 Step 1: Data Organization And Selection

The operations carried out during this step are identical to the ones in the imaging workflow, and they are described in Section 10.1.2.

10.2.3 Step 2: Data Decoding

The decoding is similar to that for the imaging workflow – the input FITS files are transformed into a format (still FITS) that makes the subsequent processing easier. The relevant entries from the standard star catalog are sent to the actor that will eventually perform the photometry on the processed spectra.

10.2.4 Step 3: Distortion Correction

In spectroscopic long slit mode the optical distortion is known analytically. This is used to directly correct the distortion, by interpolating the distortion corrected pixel value from the source pixels. This interpolation ignores source pixels that are marked as bad.

10.2.5 Step 4: Wavelength Calibration and Spectral Extraction

The dispersion relation is approximated well by a polynomial. The VISIR physical model - represented with a first degree polynomial - is used as a first guess. The experience has shown that only a linear shift needs to be adjusted and this is achieved by cross- correlating a sky spectrum from the data and a model of the atmospheric emission. The derived offset is then applied to the model.

By default the pipeline extracts 1D spectrum of the brightest source, applying the optimal extraction method. However, the user has options to define on each image multiple (up to 10) extraction apertures and to use different methods - optimal or aperture extraction - to obtain a 1D spectrum. Therefore, the pipeline allows simultaneous (in a single run) 1D spectra extraction from different objects that fall onto the slit or from different parts of the same object (e.g. core, jet, disk of a galaxy, etc.).

10.2.5.1 Interactive GUI (“Aperture Editor”) description

The Aperture Editor is responsible for creating an aperture definitions file and passing it to the `visir_old_spc_obs` recipe (via its `--apfile`[6] parameter) whenever the user clicks the *Re-run Recipe* button. Figures 10.2.1 and 10.2.2 show the Aperture Editor illustrating an optimal extraction and an aperture sum configuration, respectively. Within the GUI, the left mouse button is used exclusively in most panels. However the combined spectral image panel to the left uses a combination of the left and middle mouse buttons: the left mouse button is used to zoom or pan the image, while the middle mouse button is used to add, modify, or delete apertures.

On the left the GUI displays the combined 2D spectrum. The horizontal bar on the bottom shows the width of the aperture that is to be added or resized. The vertical panel in the center allows the user to perform various

[6] The `visir_spc_reduce` recipe also supports this parameter.

actions, set up some parameters and, in the bottom block, see the status of the current aperture selections (e.g. if there are conflicts like missing sky apertures or overlapping science and sky apertures). The right side is identical with the GUIs for other interactive workflows. The users have two color schemes available, depending on the contrast - they can be alternated by clicking on the "Light" and "Contrast" buttons in the central vertical panel. The display labels the order ID (relevant for cross-dispersed spectra) along the X-axis and the wavelength in microns for the active order along the Y-axis (in case of cross-dispersed spectra, a star next to the order ID indicates which order the wavelength on the Y-axis corresponds to).

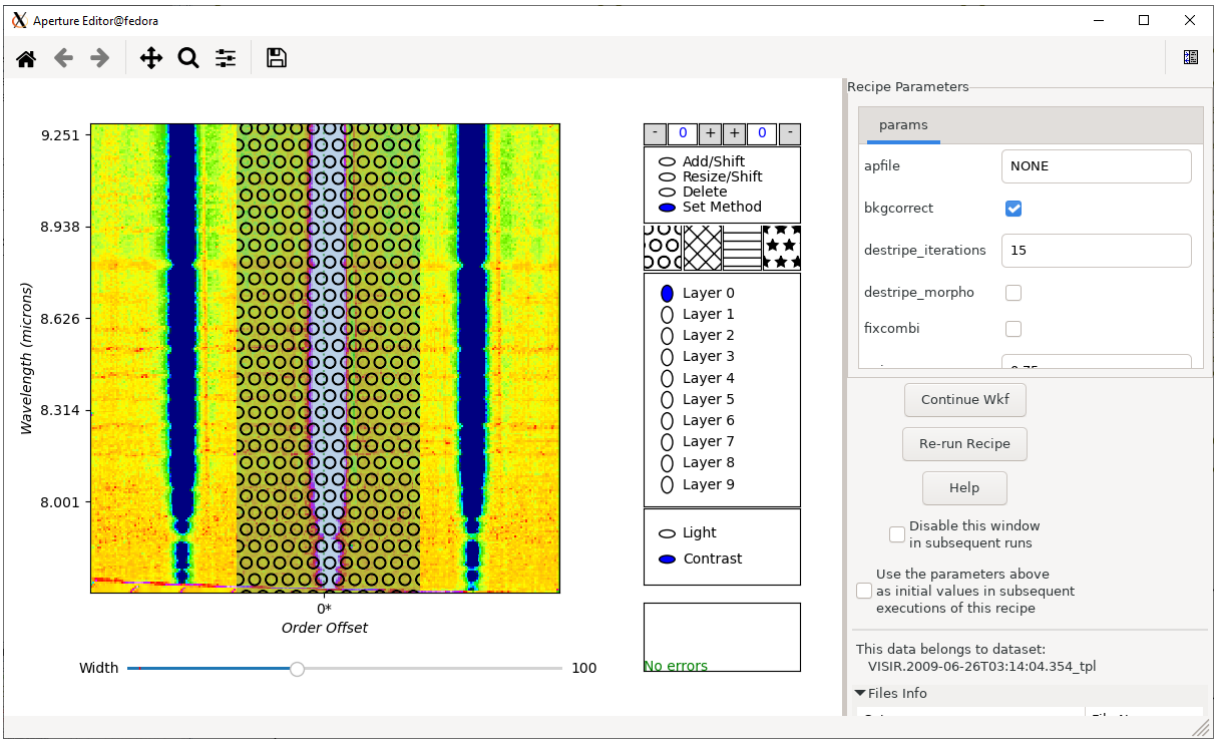


Figure 10.2.1: *Long slit mode: optimal extraction aperture being configured.*

Figure 10.2.3 shows the GUI for a cross-dispersed spectroscopic mode. Green and red order boundaries indicate which orders have been successfully extracted or not, respectively. For the example shown in the figure, the algorithm failed for order -2, hence it is shown in red. The most common problem is with the wavelength calibration (which usually happens for spectral regions with strong atmospheric absorption) although a number of other failure modes are possible.

Textured apertures within the orders indicate the extraction algorithm (in case of optimal extraction, no sky apertures are defined) and the sky calculation algorithm (in case of extraction with a sum within the target aperture). **Circles** are for optimal extraction whereas, in the case of summing, the aperture is textured depending on the way the sky is calculated, with **crossed lines** for averaging of all pixels within all sky apertures, **horizontal lines** for linear fitting of all sky pixels, and **stars** for median averaging of all sky pixels. Note that the sky apertures themselves are not textured, so they can clearly be distinguished from the target aperture (which is important if the sky and target apertures touch).

The central vertical panel is composed of six (6) separate blocks, whose purpose is described below:

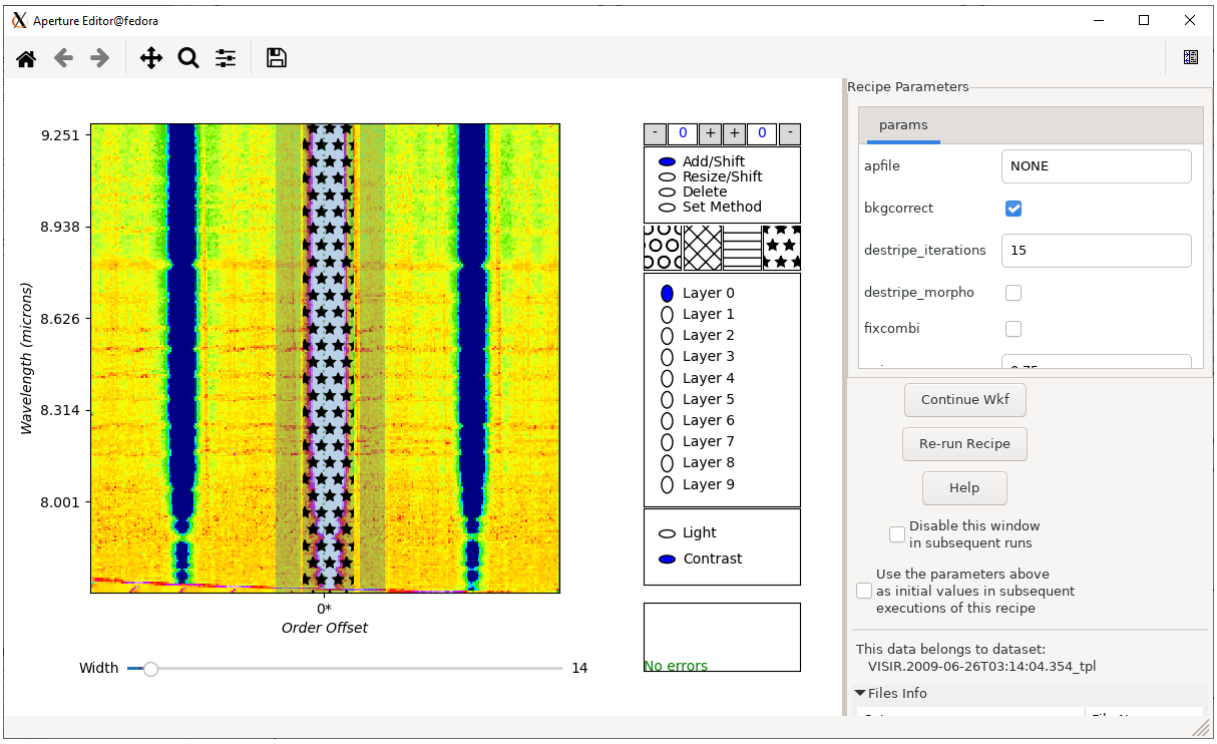


Figure 10.2.2: *Long slit mode: target aperture (to be summed) and sky apertures being configured.*

- **The top block:** This block contains parameters rl and rr - these define how many pixels from the order to skip, counting from the detected left and right edges inwards, respectively. Negative values would make the orders wider, but are not recommended. For long-slit the recommended values of rl and rr, to avoid the negative spectra on the stacked 2D images, are in the range 70-100; for cross-dispersed mode in most cases they should be close to zero.

- **The second block from the top:** This block defines the main actions. They are selected with a left mouse button click and serve respectively to add a new aperture (if there is already an aperture, it will be moved), to re-size and shift an existing aperture, to delete an aperture and finally at the bottom - to define the method of extraction within that aperture. Once the action is selected with the left button click on the oval buttons, then the actual action is performed by clicking with the middle mouse button on the image on the left side of the GUI. Note that optimal extraction is applied across the entire aperture, so the apertures for this method need to be wider, to encompass enough sky.

- **The third block from the top:** This block is just a legend of the selected extraction method. The method alternates along the sequence by (i) selecting "Set Method" and (ii) middle-button clicking on the selected aperture enough times until the desired method comes up, which is recognizable from the changed pattern on the left 2D image.

- **The fourth block from the top:** This block defines the layer: only one aperture can be defined per order per layer; up to ten layers can be defined for each order. Therefore, if the user needs multiple apertures, e.g. with different extraction methods and/or different widths, each of those apertures must be defined in

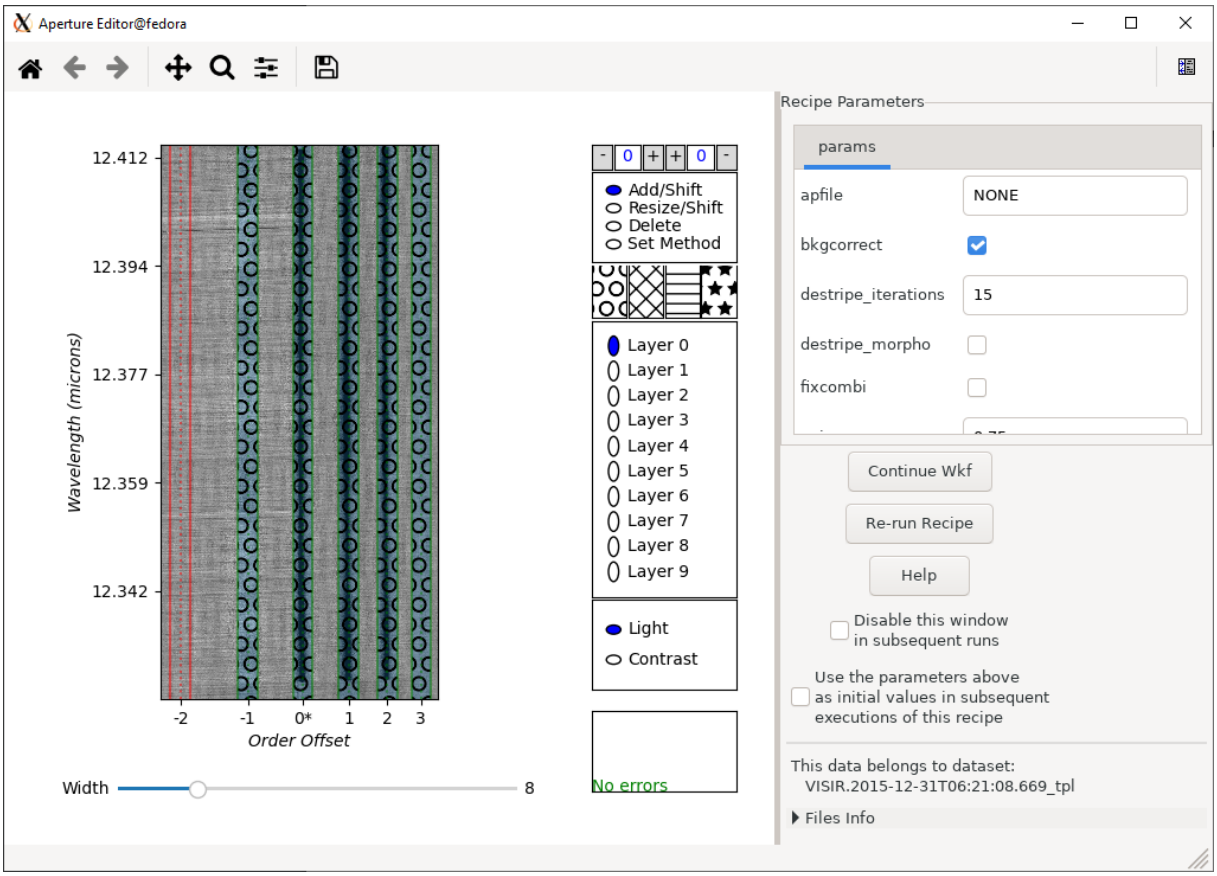


Figure 10.2.3: *Cross-dispersed mode.*

a different layer. The layers are selected by left mouse button clicking on the layer button.

- **The fifth block from the top:** This block sets the display mode - respectively optimized for bright or faint sources.
- **The bottom block:** This block reports the status of the current aperture definitions - the GUI automatically checks for errors such as missing sky apertures for aperture (non-optimal) extraction or for overlapping apertures. This is automatic and the user only needs to inspect the reported order/layer combo and correct the problematic aperture.

10.2.5.2 Interactive GUI ("Aperture Editor") operation

The Aperture Editor is only available if the global interactivity parameter is set to true. If it is set to false, the workflow can be run in lazy mode, suitable for processing large data set in batch-like regimen. In that case only the automatically recognized sources will be extracted with the default optimal extraction.

In interactive mode first the workflow will go through the data organizer and depending on the `SelectDatasetMethod` may open a window, asking the user to select which dataset to process. Once

the execution reaches the `Spectra Extraction` actor, it will first automatically identify the sources and place on the wide apertures for optical extraction. If the user is happy, they can press on the *Continue Wkf* button and stay with those apertures. If they want to edit the apertures, they have the following options (it helps to zoom on the display image with the usual Python magnifying glass button before attempting any of the following actions):

- **Add an aperture:** press on the *Add/Shift* button, adjust the Width from the bar below and click with the middle button on the 2D spectrum display at the location where they want to place the new aperture. They can adjust the rl and rr parameters before (and after) doing that. It also helps to minimize the operations, if they adjust the Width before adding the aperture. Subsequent clicks on the same order will shift the aperture, not add a new one, unless the user switches to a new layer that has no aperture on that order. The new aperture will be drawn on the top of the 2D spectrum.
- **Resize an aperture:** press on the *Resize/Shift* button and then click on the aperture you want to edit. Note that for both actions resize and shift will take place, so if one wants to resize only, they have to click at the center of the aperture. The edited aperture will be automatically redrawn with the new parameters.
- **Delete an aperture:** press on the *Delete* button and then click on the aperture you want to remove.
- **Change the extraction and/or sky background algorithms:** press on the *Set Method* button and then click on the selected aperture; each click will alternate in sequence between the different modes. In the case of extracting with simple sum within the target aperture, the mode refers to the type of the sky averaging: linear fit, average and median.

If more than one aperture needs to be defined for a given order, switch to a different layer and proceed the same way. There are ten layers, limiting the maximum number of apertures per order to ten. If more apertures need to be extracted, the user needs to rerun the pipeline.

Once the apertures are set, the user MUST click on the *Re-run Recipe* button on the right. THIS IS IMPORTANT, because otherwise the 1D spectra corresponding to the newly defined apertures will not be extracted! Instead, the apertures generated from the previous execution (or the ones generated automatically at the first run) will be applied.

The Aperture Editor generates an aperture definitions file named `aperty`, saved in a timestamp-named sub-dir of the reflex_tmp_products/visir/ApertureEditor/ directory, which is then passed to the`visir_old_spc_obs` recipe via its `--apfile` parameter when the *Re-run Recipe* button is pressed. The presence or absence of this `--apfile` parameter, and the contents of the file it points to, is what determines which extraction method is used. If you wish to build this file by hand, you can learn more about the required format from the VISIR pipeline manual.

10.2.5.3 Two Different Optimal Extraction Methods

During the first run of the workflow, if the `INIT_APFILE` parameter of the `Spectral Extraction` actor is set to `NONE` (the default), then VISIR's default optimal method for extracting a 1D spectrum from the combined 2D frame will be used. This method computes the flux at given wavelength as a weighted average of the pixels in the spatial direction, where the weights are the same for all wavelengths and are obtained by collapsing the 2D spectrum along the spectral dimension and normalizing the absolute flux of this 1D image to 1. You can revert

back to this default optimal method by resetting `Spectral Extraction`'s `INIT_APFILE` parameter back to `NONE` or, from the Aperture Editor, by deleting all apertures on all orders across all layers and clicking *Re-run Recipe*[7].

An entirely different method for optimal extraction is used if the user supplies an aperture definitions file via `Spectral Extraction`'s `INIT_APFILE` parameter or, from the Aperture Editor, clicks *Re-run Recipe* with at least one optimal extraction aperture defined[8]. The method used in this case is identical to that of the FORS Pipeline, which is based on the classic K. Horne algorithm, [3].

See the VISIR Pipeline manual for more details on the aperture definitions file format.

10.2.6 Step 5: Photometric Calibration and Output Organization

The spectral photometric calibration is carried out comparing a flux model (available in the static calibration database) of the standard star and the observed flux. The final reported sensitivity is computed in units of mJy at 10σ level in 1 hr (the conversion factor is the reciprocal this value). The normalization per unit time is obtained calculating the total integration time as follows: `DIT × NDIT × NFILES × NCHOP × 2`. The last factor of two is due to the half-cycle chopping. For more details see the VISIR pipeline manual.

[7]These are both equivalent to leaving the `--apfile` parameter off of the call to `esorex visir_spc_reduce` or `esorex visir_old_spc_obs`.

[8]These are both equivalent to supplying an aperture definitions file containing at least one optimal extraction line to `visir_old_spc_obs` or `visir_spc_reduce` via the `--apfile` parameter on the `esorex` command line.

11 Optimizing Results Through Workflow Interaction

There are limited possibilities to improve the quality of the final product during the processing, and typically the improvement on one aspect implies a degradation in another. For instance, applying a more stringent requirement for maximum FWHM (see the interactive GUI in the imaging workflow, Section 10.1.3) will improve the resolution by throwing away the images with poorer image quality, but at the price of loosing some sensitivity away from the object, because of the reduced number of images.

12 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 ([4]). 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 ([4]) 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 ([4]) for more information.

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

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

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

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

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

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

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

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

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

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

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

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

  As indicated in Section 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`

- **If the data set contains a spectrophotometric standard, and there are multiple apertures applied to the standard, then which apertures are used to correct the science target?**

  As this opens up too many possibilities, we suggest keeping it simple and always use the aperture from layer 1 of the standard. This approach is robust. The user is free to define more apertures on the standard, e.g. for data quality purposes, but the first one is always used. Also, the default values of rl and rr for this visir_spc workflow should be set to 75 to avoid the negative images.

13 Troubleshooting

- **I downloaded the data from the ESO archive, put them into a new directory, tried to run Reflex on them, but:**
  - **it crashes:**
    This may happen if one of the files was downloaded only partially (check for a file with the extension `fits.Z.part`. You will have to download that file again in order to have an not corrupted file, and remove the partially downloaded file.
  - **it fails with error message "No DataSets have been created, check the data set and the OCA rules." (see Figure 13.0.0):**
    This error may be due to the fact that the data provided by the ESO science archive are compressed (`<filename>.fits.Z`). Please remember to uncompress them before executing `Reflex`.
  - **all DataSets are grayed out in the DataSets interactive window:**
    The ESO science archive used with CalSelector does not always supply all static calibration files. Consequently, some/all DataSets are grayed out because they were missing such required data. Missing static calibration should be found by reflex in
    `<install_directory>/calib/<pipeline_version>/cal`.
  - **Some piece of calibration data is present twice**
    Reflex can not decide which file to use, and crashes. Check the data set (either loading and inspecting the data directory with Gasgano or with the DataSet browser that is available at the start of the workflow to see if any of the calibrations are the same as those in the calibration sub-dir (specified in the workflow, top center, e.g. as `CALIB_DATA_DIR`).

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

  If a DataSet in the "Select DataSets" window is grayed out, then it means that the DataSet that was constructed is missing some key calibration(s) (i.e. the DataSet is incomplete). To find out what calibration (s) are missing from a grayed out DataSet, click on the DataSet in question to highlight it in blue, and then click on the button `Inspect Highlighted`. The "Select Frames" window that appears will report the category of the calibration products that are missing. From this the user has then to determine the missing raw data. If static calibrations are missing the mechanism unfortunately does not not work, but should be found by reflex in

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

  A faster way to identify which kind of data are missing is moving the mouse over the gray DataSet. This will open a pop-up menu that will list the missing calibrations.

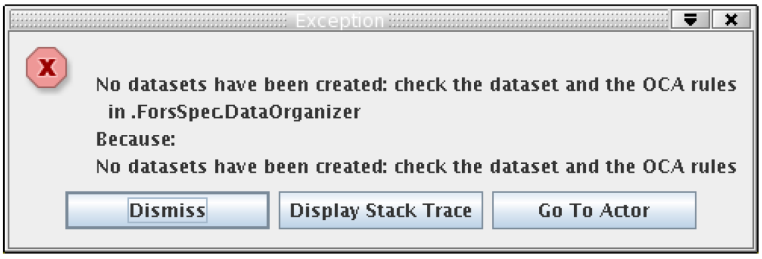


Figure 13.0.0: *The* `DataOrganizer` *interactive window reports an error "No DataSets have been created, check the data set and the OCA rules".*

Acknowledgements

The REFLEX team in alphabetical order consists of Pascal Ballester, Daniel Bramich, Vincenzo Forchí, Wolfram Freudling, César Enrique García, Maurice Klein Gebbinck, Andrea Modigliani, Sabine Moehler, Mark Neeser & Martino Romaniello.

[1] Emmanuel Bertin. *SWARP home page*. Astromatic, *http://www.astromatic.net/software/swarp*. 27

[2] ESO/SDD/DFS, *http://www.eso.org/pipelines*. *VISIR Pipeline User Manual*. 27

[3] Horne K. *An optimal extraction algorithm for CCD spectroscopy*. Publications of the Astronomical Society of the Pacific, 98(604), p.609, March 1986. 42

[4] Forchì V. *Reflex User's Manual*. ESO/SDD/DFS, *http://www.eso.org/gasgano/*, 0.7 edition, 2012. VLT-MAN-ESO-19000-5037. 24, 31, 44, 45

A Abbreviations and acronyms

ANSI	American National Standards Institute
ASCII	American Standard Code for Information Interchange
CalibDB	Calibration Database
CPL	Common Pipeline Library
DFO	Data Flow Operations department
DFS	Data Flow System department
DMD	Data Management and Operations Division
DRS	Data Reduction System
ESO	European Southern Observatory
ESOREX	ESO-Recipe Execution tool
FITS	Flexible Image Transport System
FOV	Field Of View
FPN	Fixed Patter Noise
GUI	Graphical User Interface
OB	Observation Block
OCA	Organisation Classification Association
PSO	Paranal Science Operations
QC	Quality Control
RON	Read Out Noise
SOF	Set Of Frames
SQL	Structured Query Language
UT	Unit Telescope
VLT	Very Large Telescope