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

SPHERE IFS EDPS-GUI tutorial

1 rev. 20260723

Issue 1

Date July 23 2026

Prepared:	M. Marsset	July 23 2026	
	Name	Date	Signature
Approved:			
	Name	Date	Signature
Released:			
	Name	Date	Signature

This page was intentionally left blank

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
1	23/07/2026	all	First public release for 0.59.1

Contents

1 Introduction **7**

1.1 Scope . . . 7

1.2 What is `EDPS`? . . . 7

1.3 Main concepts . . . 7

1.4 Installation . . . 8

1.4.1 Prerequisites . . . 8

1.4.2 Installation steps . . . 9

2 Reducing demo data **10**

2.1 Setting the workflow . . . 10

2.2 Selecting the input data . . . 11

2.3 Start the reduction . . . 12

2.3.1 Quality plots . . . 13

2.4 Exporting the final products . . . 14

3 The SPHERE IFS data reduction flow. **17**

3.1 Subworkflow: Dark Background . . . 17

3.2 Spectra Positions . . . 19

3.3 Wavelength Calibration . . . 20

3.4 Subworkflow: Detector Flat . . . 20

3.5 IFU Flat . . . 21

3.6 Distortion Map . . . 21

3.7 Subworkflow: Science Data Reduction . . . 22

3.8 Subworkflow: Standard Data Reduction . . . 23

4 Overview of all the data reduction configuration options **24**

4.1 Selection of most appropriate calibrations . . . 24

4.2 Configuration of parameters: the configuration editor . . . 24

4.3 Troubleshooting . . . 28

4.3.1 Master darks . . . 28

4.3.2 Distortion map 28
4.3.3 Wavelength calibration 28
4.3.4 Coronagraph centre 28
4.3.5 Spectral deconvolution 29
4.3.6 Science reduction 29

5 List of workflow tasks **30**

6 Frequently Asked Questions **31**

1 Introduction

1.1 Scope

This document describes how to reduce SPHERE IFS data with the edps-gui (Graphic User Interface), the dashboard of the ESO Data Processing System (EDPS), which is the recommended interface to reduce data from ESO telescopes. Details on the SPHERE IFS data reduction stream and how to configure the reduction to meet specific scientific needs are also given.

For a more extensive documentation on the edps-gui itself, consult the dedicated manual here.

For a description of the SPHERE IFS pipeline itself, consult the pipeline manual available at: https://www.eso.org/sci/software/pipe_aem_table.html.

Note: this tutorial refers to:

- SPHERE IFS instrument pipeline named spher, version 0.59.1.
- SPHERE IFS workflow: sphere.sphere_ifs_wkf
- EDPS version 1.7.1.
- edps-gui version 0.91.

1.2 What is EDPS?

The ESO Data Processing System (EDPS) is a framework to run ESO's data processing pipelines and it is meant to eventually replace the previous ESOReflex environment. The general principles of EDPS have been described by Freudling, Zampieri, Coccato et al. [2024, A&A, 681, A93]. Please refer to that paper if you have used EDPS for research resulting in a scientific publication.

Each of ESO's data processing pipeline consists of a series of standalone programs called *recipes*. Each recipe is designed to process certain type(s) of input data. The processing of these input data typically requires a range of auxiliary files such as calibration files. EDPS is designed to select appropriate input data for the different recipes of a pipeline, and execute them in sequence. This is done by specifying for each pipeline the workflow for organizing data and executing the recipes. Then, the workflow can be used to process a set of data fully automatically.

1.3 Main concepts

EDPS is an environment designed to execute the recipes of an instrument pipeline according to a series of instructions. The main concepts in EDPS are:

- **Workflow** and **reduction cascades**. A workflow is a series of instructions designed to reduce data with an instrument pipeline in potentially multiple ways, by carrying on a sequence of tasks. Each workflow can define multiple reduction cascades, depending on the scientific needs. For example, the same workflow can be used to process data following different strategies that trigger different reduction steps (e.g. in one strategy flux calibration can be omitted) or different end-points (e.g., combine different science exposures, or stop after the reduction of individual exposures without combining them). Each of these "strategies" defines a "reduction cascade".

- **Task**, **jobs**, and **recipes**. A task is an element in the workflow that performs a given step of the data reduction cascade. Tasks are often associated to a recipe of the underlying instrument pipeline. A jobs is a work unit in a processing environment, that runs a recipe on a set of input data with a set of recipe parameters. A single task can generate several jobs: for example, a "bias" task, can generate multiple jobs, each of the running the bias recipe on a different set of input files.
- **Dataset**. A dataset is a collection of files, that are needed to perform the data reduction as specified by the workflow. It consists, for example, of one or more science files plus the calibrations needed to process them. In `EDPS`, datasets have a hierarchical structure, which highlights the connections between the various files and tasks (e.g., task A is an input to task B).
- **Target** and **Target category**. The "target", or the "target task" is the end point of the reduction cascade. When specifying a target, `EDPS` will process all and only the files needed to execute it. For example, if my target is "science", and the science files need the bias files, `EDPS` will process only the biases that have been selected to process those science files; then it processes the science using the product of the bias reduction. However, if my target is bias, then `EDPS` will process all and only the bias files, regardless they are not used by any science. In this case, `EDPS` does not process the science, as it has already reached the end reduction point (e.g., process all biases). The "Target category" is a group of targets that have similar purposes. For example, the target category "science", includes all the tasks that deliver final scientific products, the target category "qc1calib" includes all and only the tasks that processes calibrations (e.g., bias, flat fields, standard stars).

1.4 Installation

1.4.1 Prerequisites

Prerequisites for a well functioning installation of `EDPS` and `EDPS`-gui are:

- Recent Firefox or Chrome browser, Python 3.11 or higher (but there are issues with Python 3.14).
- At least one ESO pipeline with EDPS workflow should be in your system. To install the desired ESO pipelines, follow the instructions in the ESO pipelines pages. NOTE: the `apptainer` installation method is currently not supported. After the installation, the esorex command must be in the path. To test whether the installation was successful, type

  ```
  esorex --recipes
  ```

  A list of available recipes should appear.
- Install `graphviz`, `fv`, and `ds9`, which have to be included in the system path (defining aliases not enough). On linux, `Graphviz` can be easily installed via:

```
sudo apt install graphviz (Debian, Ubuntu)
sudo dnf install graphviz (Fedora)
```

Check the `Graphviz` webpage for installation instructions for other OS.

`fv` and `ds9`, are optional. To install them, follow the instructions in corresponding webpages. You can test whether these three packages are installed and their paths are correctly set by typing on a terminal:

```
dot -V
fv  -version
ds9 -version
```

1.4.2 Installation steps

To install EDPS follow these steps:

- Create a new Python virtual environment and activate it:

  ```
  python3 -m venv edpsgui
  . edpsgui/bin/activate
  ```

  Make sure the python3 version is 3.11 or higher, but not 3.14.

- Install the required packages:

  ```
  pip install --extra-index-url \
      https://ftp.eso.org/pub/dfs/pipelines/repositories/stable/src \
      edps edpsgui edpsplot adari_core
  ```

To run the `edps-gui` type from a terminal (with the active environment):

```
edps-gui
```

Important note. The first time `edps-gui` is executed, you will be asked to specify the directory where the reduction products (fits files and quality plots) will be stored. The default location is `$HOME/EDPS_data`. During the first execution, a configuration file named `application.properties` will also be saved in the directory (newly created) `$HOME/.edps`.

2 Reducing demo data

Follow this procedure to quickly reduce SPHERE IFS demo data. We assume that the EDPS, edps-gui, the SPHERE IFS pipeline and its associated demo data are installed in your system. For general instructions on how to install EDPS and the pipeline, see Section 1.4 or please visit: https://www.eso.org/sci/software/pipe_aem_main.html.

2.1 Setting the workflow

Proceed as follows:

1. If not done already, activate the EDPS virtual environment, defined during installation (Sect. 1.4).
2. Start the edps-gui dashboard by typing:

```
edps-gui
```

   The edps-gui dashboard will start in a browser window (Figure 1).

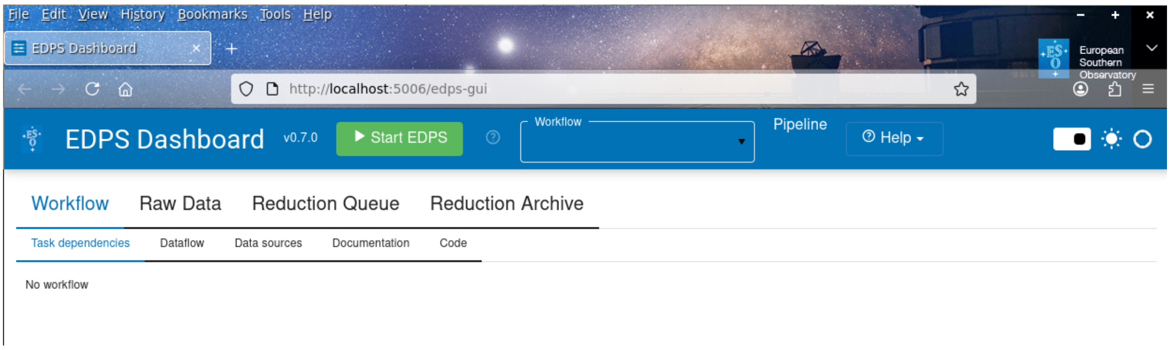

Figure 1: The empty edps-gui Dashboard; the underlying EDPS engine has not yet been started and no workflow has been loaded.

3. Optionally, before starting EDPS, one can specify new settings by pressing Help –> Settings (Figure 2).

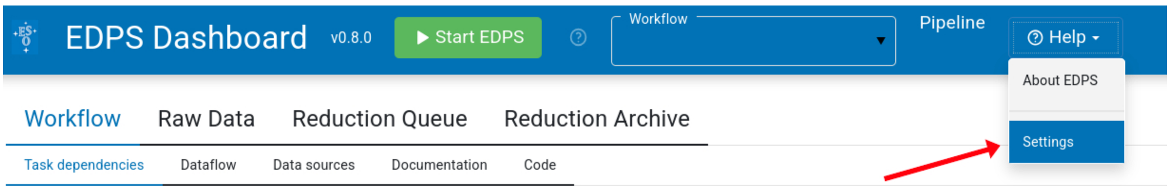

Figure 2: The "Help" –> "Settings" menu.

4. On the browser window with the dashboard, press the button 'Start EDPS'.
5. Choose the sphere.sphere_ifs_wkf workflow from the list in the 'Workflow' field. The workflows offered in this selector depend on the installed pipelines. The graphic workflow representation will appear as in Figure 4.

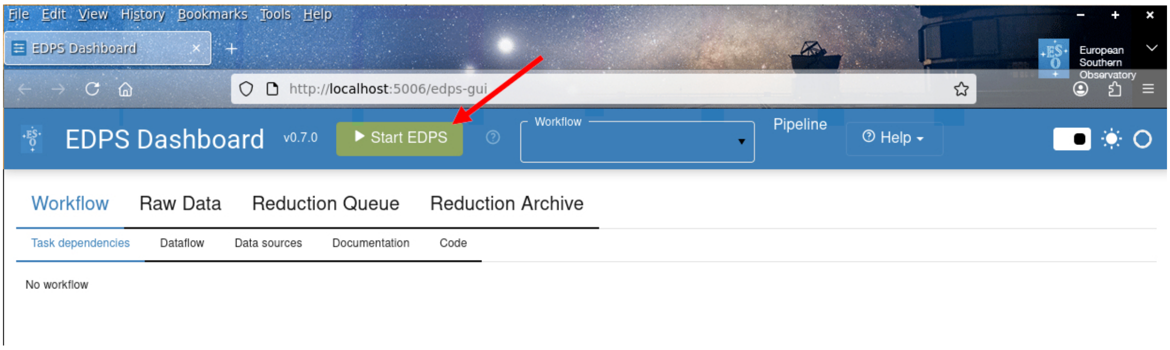

Figure 3: The “Start EDPS” button.

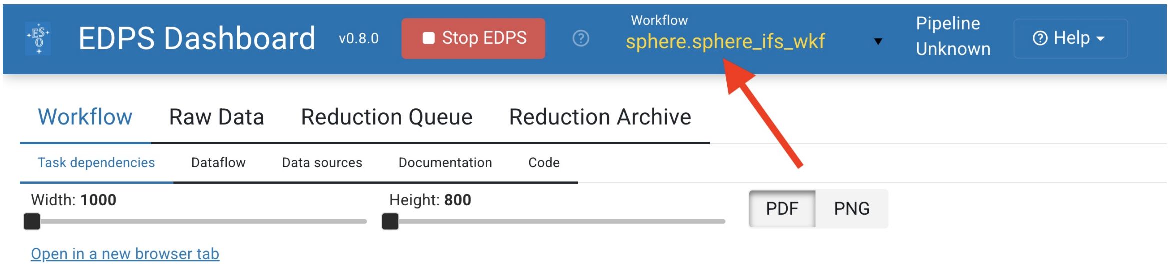

Figure 4: The `edps-gui` with the SPHERE IFS workflow loaded.

2.2 Selecting the input data

1. Press ‘Raw Data‘ to enter the corresponding tab, as in Figure 5.

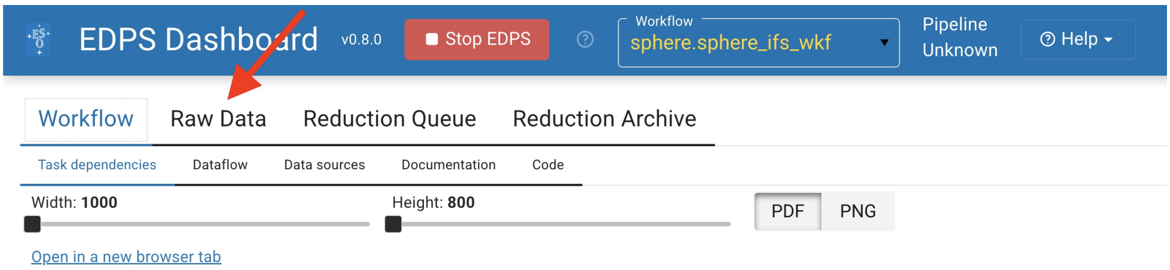

Figure 5: How to select RAW data Tab.

2. Press ‘Select Inputs‘. A selection window will appear that allows to select data that are stored on a local disk (Figure 6).
3. (Optional). Select the reduction target, configure the workflow parameter and specify the association preferences. These steps are optional. For more information see Section **??**.
4. Press ‘Create Datasets‘. A list of datasets appears, one line for each set of science data (Figure 7).
5. Choose the datasets that should be processed (Figure 8)

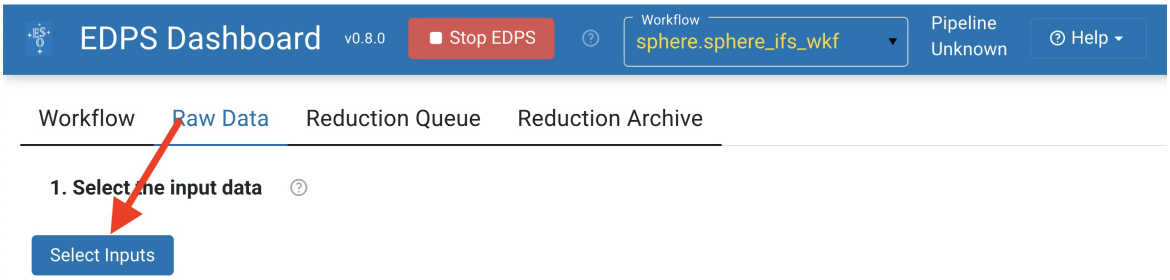

Figure 6: How to select input data.

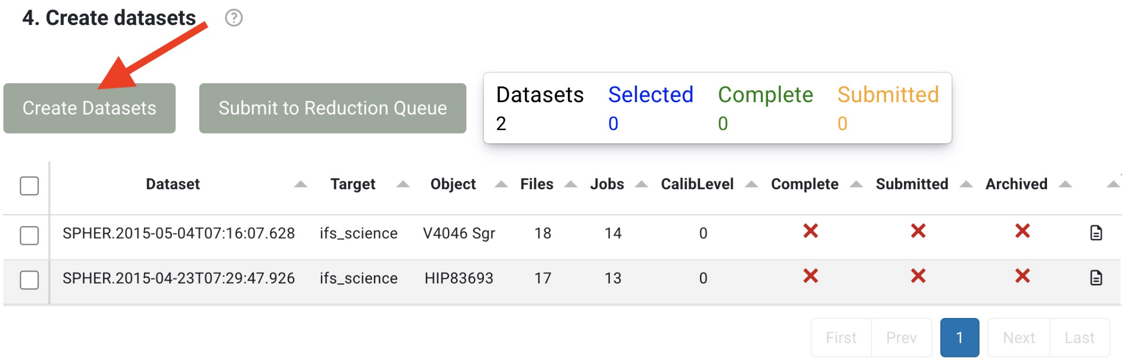

Figure 7: How to inspect the input data directory to create datasets.

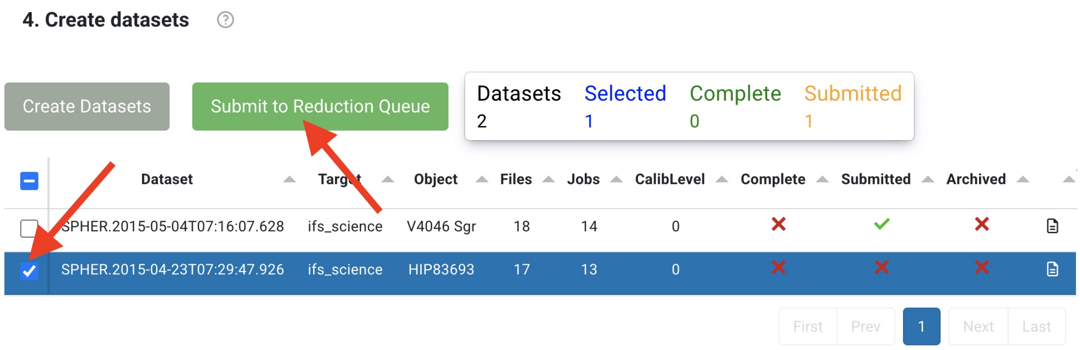

Figure 8: How to send the selected datasets to the Reduction Queue for processing.

and send them to the data reduction queue by pressing ‘Submit to Reduction Queue‘. Note that this action does not start the reduction automatically.

2.3 Start the reduction

1. Press the ‘Reduction Queue‘ tab (Figure 9).

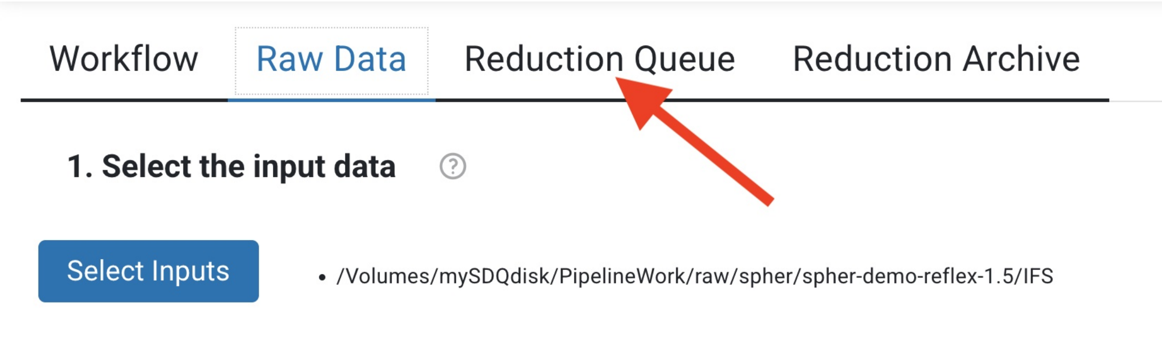

Figure 9: How to select Reduction Queue tab.

2. Select the datasets you'd like to reduce.

3. (Optional). Configure the workflow and recipe parameters by pressing the wheel button to open the configuration editor. See Section 4.2 for more information on the configuration editor.

4. Press the 'Reduce' button (Figure 10). The selected data will now be processed with the configured parameters.

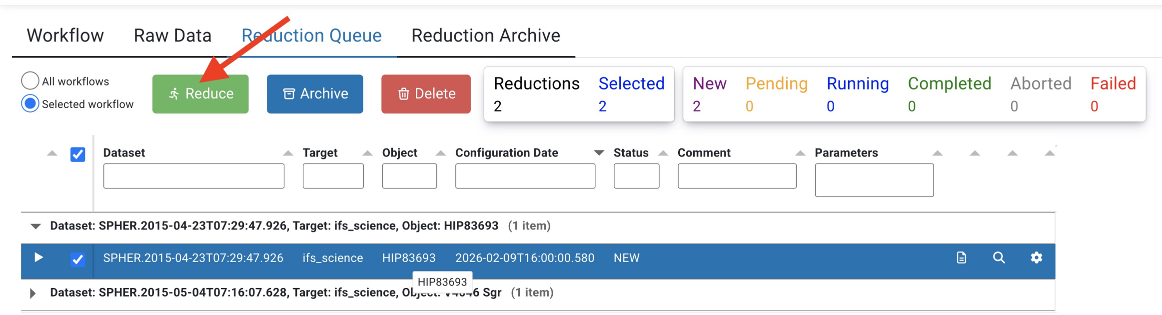

Figure 10: Reduce.

Congratulations! You reduced your first data with the `EDPS` dashboard! All the reduced data are saved in the EDPS_data directory specified when executing `edps-gui` for the first time.

2.3.1 Quality plots

It is possible to inspect the information on each job, such as quality plots showing the products (for the most important jobs and products), the list of inputs and output files, the recipe parameters and logs. All the information on the job processing can be inspected from the 'Reduction Queue' window. While quality plots are produced only for completed jobs, the other relevant information is available also for failed jobs.

The information associated for the main product can be inspected by pressing the magnifying glass symbol at the right side of each dataset. To inspect the information associated to other individual jobs (e.g., calibrations), proceed as follows:

- Expand the desired dataset by pressing the black arrow on its left. The list of jobs will appear with the associated status (COMPLETED, RUNNING, PENDING, MISSING, ABORTED, FAILED)
- Press the magnifying glass symbol at the right side of the job you want to inspect.

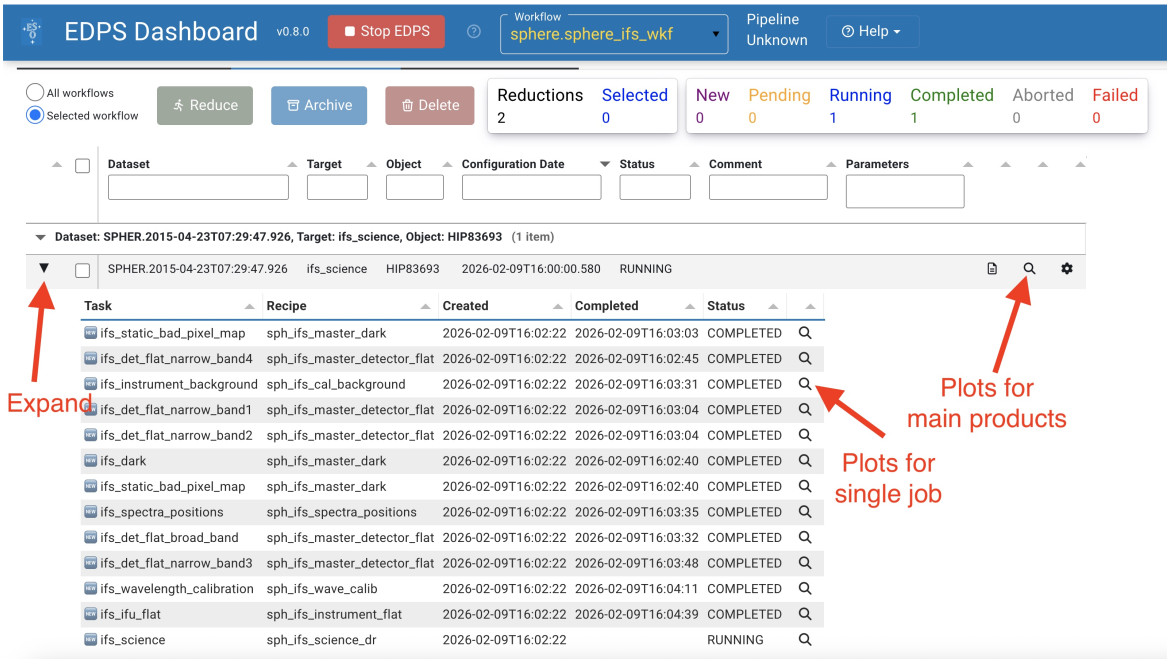

Figure 11: How to look for job information from the Reduction Queue tab.

2.4 Exporting the final products

Completed reductions can be 'Archived' (i.e. declared 'completed' because no more work is needed) and removed from the Reduction Queue. Additionally, even if all products for all tasks are saved in the EDPS_data directory, the most important products can be 'exported' to a desired location.

To do so, proceed as follows:

1. In the 'Reduction Queue' tab, select the dataset and the dataset for which you want to export the final products, and press the 'Archive' button.
2. Go in the Reduction Archive tab and click on the 'Export' button. A new tab window appears where you can indicate the directory you want to copy your final products; finally press "Export" to copy the data.

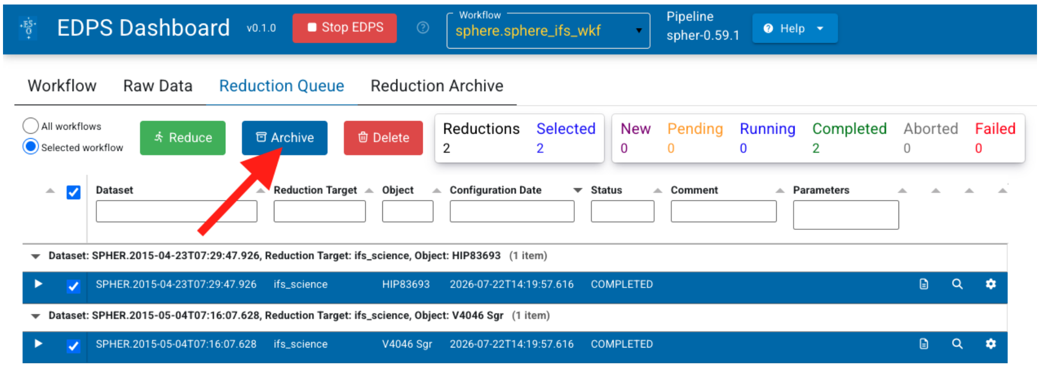

Figure 12: How to archive a completed reduction from the Reduction Queue tab.

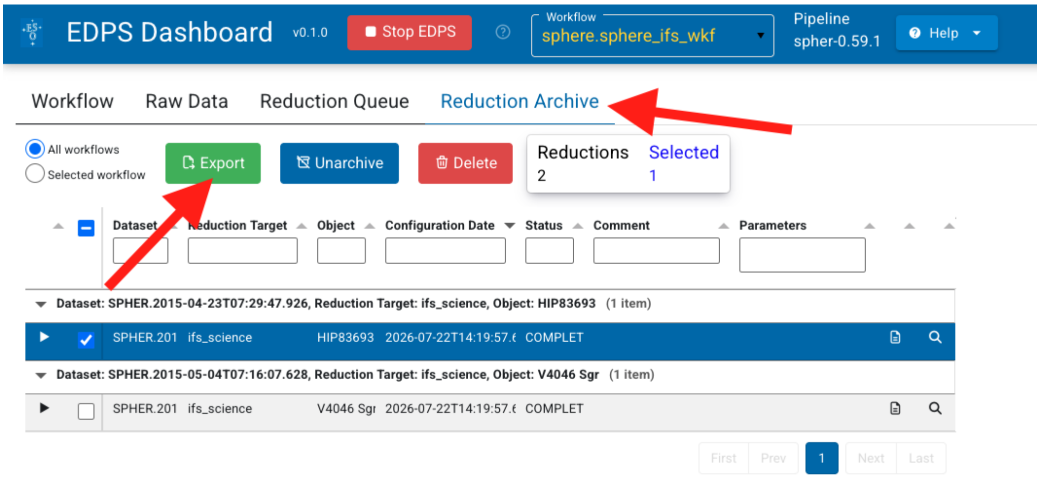

Figure 13: The reduction archive tab. This table contains all the different configurations of datasets that are declared "finished" and removed from the Reduction Queue. From this page, the user can export the most important files into a desired local directory.

Exported products are organized by ‘DATASET‘ (named as the first scientific exposure of the dataset), and ‘TIMESTAMP‘ (time of start of reduction).

The final product saved in the specified directory is the wavelength-calibrated datacube, with name format `DATACUBE_` followed by the archive file name (corresponding to header keyword `ARCFILE`).

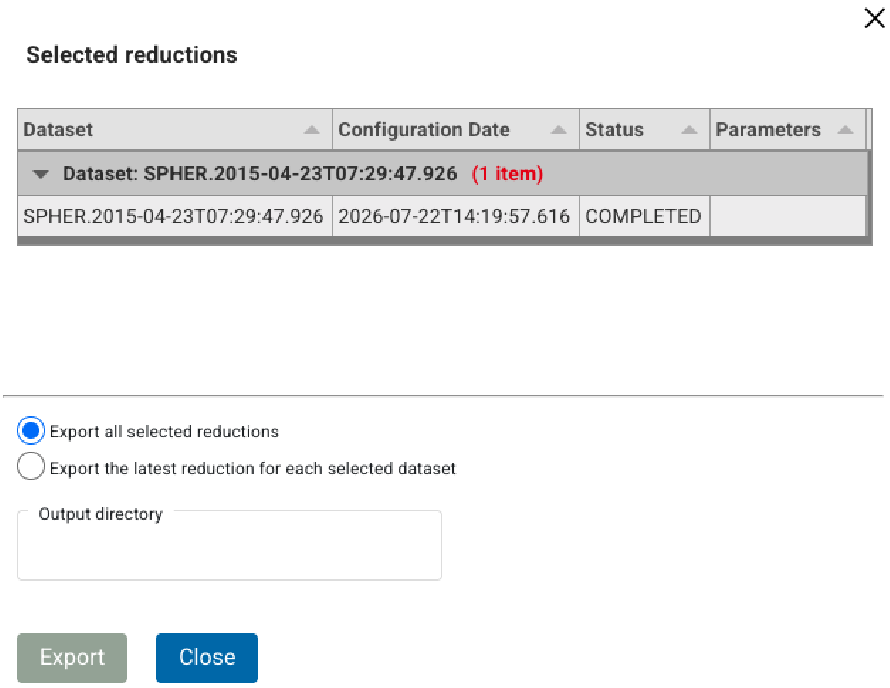

Figure 14: The EXPORT dialogue window, where the user can decide which reduced configuration to save and where.

3 The SPHERE IFS data reduction flow.

The overall data flow of the SPHERE IFS pipeline is displayed in Figure 15.

The reduction cascade is organized in tasks, which represent well-defined steps in the process. Tasks can be grouped inside sub-workflows. Each task runs a recipe; the detailed description of the algorithms, input, outputs and recipe parameters used in each recipe are available in the pipeline manual. Here, we present only the description of most important features.

The EDPS workflow is designed to execute the tasks that deliver the final reduced data for each dataset. It can be either the product of a single exposure, or the combination of multiple exposures. Only calibrations needed by the selected scientific exposures are processed.

It is possible to set EDPS to perform the data reduction until a certain step of the reduction chain (e.g. to reduce only standard stars, or only flat fields). This is done by specifying the desired tasks in the field **Select reduction target** of the **Raw Data** tab.

The reduction steps of the workflow are listed below. Before starting the reduction, the parameters of the recipes associated to each task can be configured by pressing the button close to each dataset configuration. See Section 4 for more information.

The reduction steps are:

3.1 Subworkflow: Dark Background

Recipes: `sph_ifs_cal_background`, `sph_ifs_master_dark`

This subworkflow generates the instrumental-background and dark-current calibration products. It consists of the following tasks:

- **ifs_instrument_background**
- **ifs_sky_background**
- **ifs_dark**
- **ifs_static_bad_pixel_map**

The tasks **ifs_instrument_background** and **ifs_sky_background** use the recipe `sph_ifs_cal_background` to generate instrumental- or sky-background calibration frames expressed in counts s^{-1} pixel^{-1}. The output is a multi-extension FITS file containing the background image, a bad-pixel map, an RMS map, and a weight map.

The task **ifs_dark** uses the recipe `sph_ifs_master_dark` to generate the detector master dark (`IFS_MASTER_DARK`) by combining the input raw dark exposures using the selected collapse algorithm.

The task **ifs_static_bad_pixel_map** runs the same recipe, `sph_ifs_master_dark`, and exports the `IFS_STATIC_BADPIXELMAP` product, which is also stored in the second extension of the master dark.

Bad pixels are identified in three successive steps:

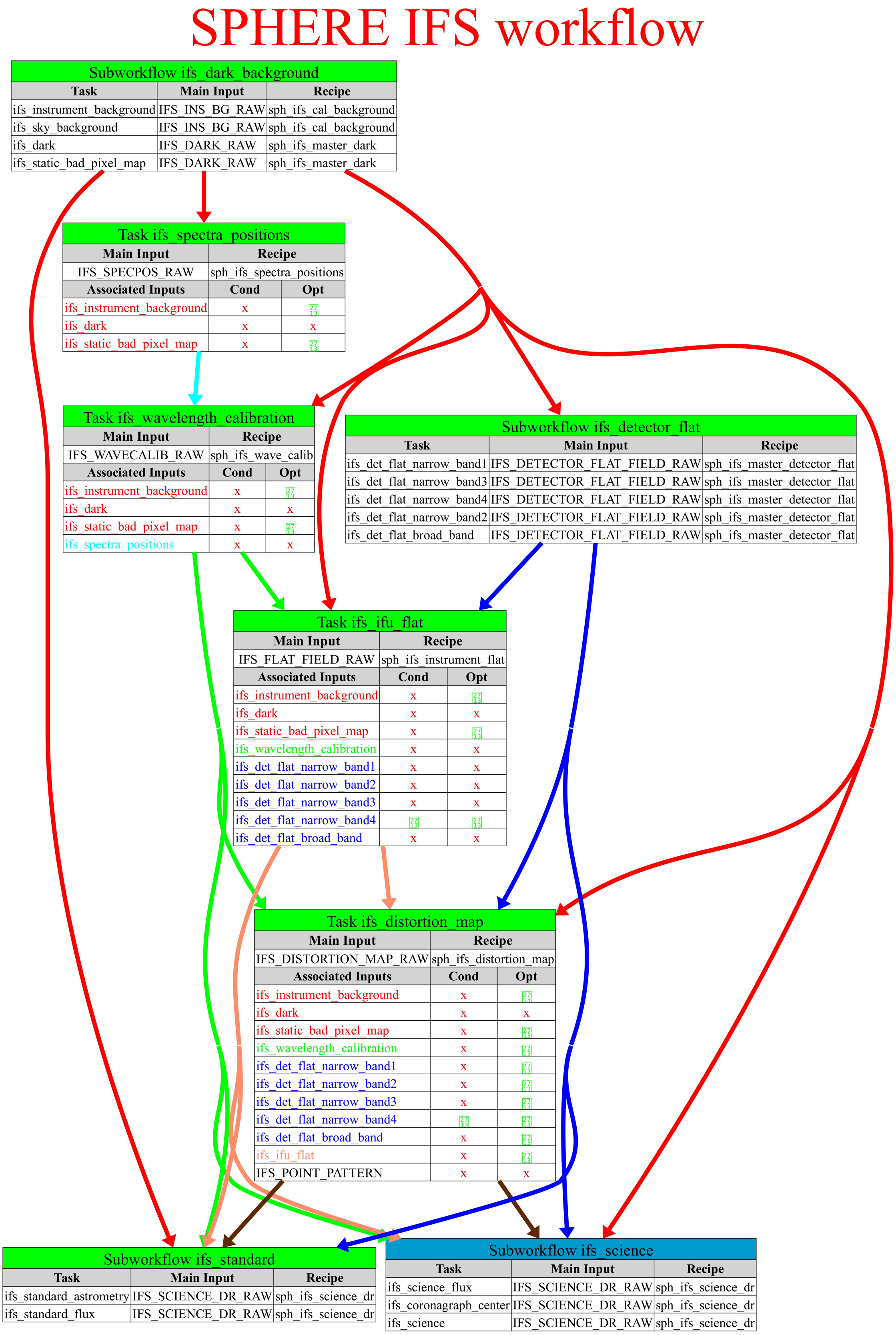

Figure 15: The data reduction cascade of the SPHERE IFS workflow.

1. Initial thresholding using the `min_acceptable` and `max_acceptable` parameters.
2. Smoothing the master dark with a Gaussian kernel and subtracting the smoothed frame to remove large-scale variations.
3. Sigma clipping of the residual image using the `sigma_clip` parameter.

The final master dark is written as a multi-extension FITS file containing the dark image, bad-pixel map, RMS map, and a weight map recording the number of contributing raw frames.

When a background correction is required elsewhere in the workflow, EDPS preferentially associates a sky background, then an instrumental background, and finally a master dark if no dedicated background calibration is available.

Customization Recipe parameters:

- `ifs.master_dark.coll_alg`: Collapse algorithm (`0` = Mean, `1` = Median, `2` = Clean Mean; default).
- `ifs.master_dark.min_acceptable`, `ifs.master_dark.max_acceptable`: Lower and upper pixel-value thresholds used during bad-pixel detection.
- `ifs.master_dark.sigma_clip`: Sigma-clipping threshold used to identify bad pixels (default: `3`; increasing the value to `5` results in fewer rejected pixels).

3.2 Spectra Positions

Recipe: `sph_ifs_spectra_positions`

The task **ifs_spectra_positions** generates the initial mapping between detector pixels and IFS lenslets, and assigns a first-order wavelength solution to each detector pixel.

Spectral traces are detected through thresholding and their centroid positions are measured. These positions are compared with those predicted by a scaled and shifted lenslet model, allowing refinement of the model scale and position. Optionally, a two-dimensional distortion model can be fitted and incorporated into the lenslet model to improve the subsequent wavelength calibration.

The main output product is `IFS_SPECPOS`, a pixel description table (PDT), stored as a multi-extension FITS file containing:

- Wavelength
- Spectral region identifier
- Lenslet identifier
- Wavelength width
- Second derivative of the wavelength
- Illumination fraction

3.3 Wavelength Calibration

Recipe: `sph_ifs_wave_calib`

The task **ifs_wavelength_calibration** performs the wavelength calibration by refining the pixel-to-wavelength associations produced by **ifs_spectra_positions** using wavelength calibration exposures and the existing lenslet model.

One-dimensional spectra are extracted for each spectral region and the positions of known calibration lines are measured using a flux-weighted centroid within a configurable fitting window. The measured line positions are fitted with a polynomial defining the wavelength solution for each spectrum. If the derived solution is inconsistent with the expected dispersion, the original model wavelengths are retained or the affected spectral region is flagged as invalid.

From the final wavelength solutions, the recipe computes quality-control parameters for each spectrum, including the minimum, maximum, and central wavelengths, as well as the spectral resolving power derived from the local wavelength gradient.

The main output product is `IFS_WAVECALIB`, which contains the updated wavelength calibration derived from the initial `IFS_SPECPOS` product.

3.4 Subworkflow: Detector Flat

Recipe: `sph_ifs_master_detector_flat`

This subworkflow generates detector flat-field calibrations from narrow- and broadband lamp exposures. It consists of the following tasks:

- **ifs_det_flat_narrow_band1**
- **ifs_det_flat_narrow_band2**
- **ifs_det_flat_narrow_band3**
- **ifs_det_flat_narrow_band4** (only associated for the `PRI_YJH` disperser.)
- **ifs_det_flat_broad_band**

The recipe produces several detector flat-field components, including a standard detector flat, a large-scale (smoothed) flat, and a preamplifier-correction flat. In most applications, the standard `IFS_MASTER_DFF_LONG*` products are sufficient to correct pixel-to-pixel detector response variations.

Raw frames are first corrected for dark current and preamplifier variations. The detector flat is then derived by fitting, for each pixel, a linear relation between its signal and the mean illumination level of the corresponding exposure. The slope of this fit represents the relative pixel response.

Optional products include a detector non-linearity map and a smoothed large-scale flat-field image.

Customization Recipe parameters:

- `ifs.master_detector_flat.robust_fit`: Controls the fitting method. By default (`TRUE`) a maximum-likelihood fit is used. Setting the parameter to `FALSE` enables a robust linear fit that is more resistant to outliers such as cosmic rays.

3.5 IFU Flat

Recipe: `sph_ifs_instrument_flat`

The task **ifs_ifu_flat** uses the pipeline recipe to generate flat-field calibrations for the IFS. The underlying recipe supports two operating modes. Within the EDPS workflow, the IFU-flat mode is used to generate the calibration products required by subsequent reductions.

In detector-flat (or total-flat) mode, raw calibration frames are combined to generate a flat field that includes the detector response. This product can subsequently be used by the **ifs_spectra_positions** and **ifs_wavelength_calibration** tasks.

In IFU-flat mode, the recipe removes the detector response using the pixel description table (PDT) and wavelength calibration products in order to isolate the transmission of the IFU lenslet array.

A wavelength-dependent "super-flat" is first constructed from the master detector flats and applied to the combined calibration frames. Using the lenslet model provided by the wavelength calibration, spectra are extracted for every lenslet and collapsed along the wavelength direction to derive a single flat-field value for each lenslet.

The main outputs are:

- `IFS_INSTRUMENT_FLAT_FIELD`: detector flat-field product containing the flat-field image, bad-pixel map, RMS map, and weight map.
- `IFS_IFU_FLAT_FIELD`: IFU flat-field product containing the flat-field image, bad-pixel map, RMS map, weight map, together with a binary-table extension listing the lenslet flat-field values.

3.6 Distortion Map

Recipe: `sph_ifs_distortion_map`

The task **ifs_distortion_map** measures the geometric distortion of the SPHERE IFS lenslet grid.

Raw calibration frames are first reduced in a manner similar to science data using the master dark. Instrumental background, wavelength calibration, detector flats and IFU flat may additionally be used when available. The resulting monochromatic images are collapsed along the wavelength axis and point sources are identified using a user-defined detection threshold.

If no reference point pattern is provided, the recipe derives one from the detected sources and writes it as an output product. Otherwise, the supplied reference pattern is used directly.

The distortion is measured by comparing the detected and expected source positions. Vectors larger than a configurable threshold are rejected before the remaining offsets are fitted with a two-dimensional polynomial model.

Optionally, the derived distortion solution can be applied to the data to produce a residual distortion map, providing a diagnostic of the quality of the distortion correction.

The outputs are:

- `IFS_POINT_PATTERN`: reference point pattern that may be reused as input for subsequent executions of the recipe.
- `IFS_DISTORTION_MAP`: multi-extension FITS file containing the distortion maps in both detector coordinates. For each coordinate, the product includes the distortion image, bad-pixel map, RMS map, and weight map.

Customization Recipe parameters:

- `ifs.distortion_map.threshold`: Detection threshold used to identify calibration spots (default: `3.0`).

3.7 Subworkflow: Science Data Reduction

Recipe: `sph_ifs_science_dr`

This subworkflow performs the reduction of IFS science observations. It consists of the following tasks:

- **ifs_science_flux**
- **ifs_coronagraph_center**
- **ifs_science**

The workflow reduces science observations with sky-, instrument background-, or dark-subtraction and preamplifier stripe correction.

It also processes **FLUX** frames obtained during coronagraphic observations, in which the telescope is offset to move the target away from the coronagraphic mask in order to measure the stellar flux.

The workflow also processes **CENTER** frames used to determine the stellar position behind the coronagraph. The resulting products are primarily intended for later analysis rather than subsequent EDPS processing.

Large-scale flat-field variations are corrected using a wavelength-dependent “super-flat” derived from multi-colour lamp flats and the wavelength calibration, while small-scale detector response variations are corrected using the broadband detector flat.

The recipe also supports automatic post-processing through spectral deconvolution (Mesa et al. 2015) and Angular Differential Imaging (ADI), which can be enabled or disabled through recipe parameters.

The processing is currently done file-by-file because the pipeline does not stack science data.

Customization Workflow parameters:

- `force_distortion_correction`: Force the association of distortion calibration to IFU scientific task. Possible values are `TRUE`/`FALSE`. QC0 process requires this parameter set to `TRUE`. In the current EDPS workflow, the associated distortion calibration is not propagated to the science recipe.

Recipe parameters:

- `ifs.science_dr.use_adi`: Controls the use of Angular Differential Imaging (`0` = disabled, `1` = always enabled, `2` = automatically enabled when the total field rotation exceeds `ifs.science_dr.min_adi_angle`).
- `ifs.science_dr.min_adi_angle`: Minimum field rotation required for the automatic ADI mode.
- `ifs.science_dr.spec_deconv`: Controls the use of spectral deconvolution (`0` = disabled, `1` = enabled).

3.8 Subworkflow: Standard Data Reduction

Recipe: `sph_ifs_science_dr`

This subworkflow consists of the following tasks:

- **ifs_standard_astrometry**
- **ifs_standard_flux**

At present, astrometric-standard and flux-standard observations are reduced using the same workflow and recipe as science observations. Consequently, the same calibration steps, reduction algorithms, and recipe parameters described in the previous section are applied.

4 Overview of all the data reduction configuration options

4.1 Selection of most appropriate calibrations

By default, EDPS associates raw calibrations to the reduction process. It is also possible to use pre-processed calibrations (a.k.a. master calibrations) if available, in order to speed up the reduction. The preference can be specified in the Raw Data tab, before creating the datasets.

Possible values of the Calibration Preferences are:

- **raw_per_quality_level**: At equal quality of reduction, association of raw calibrations is preferred. This is the default.
- **master_per_quality_level**: At equal quality of reduction, association of master calibrations is preferred.
- **raw**: Association of raw calibration is preferred, despite the quality of results.
- **master**: Association of master calibration is preferred, despite the quality of results.

When master calibrations are used, the reduction step needed to process raw calibrations are not executed. The reduction then moves directly to the process of scientific exposures.

For example, if reduction speed for a quick check is preferred over a high quality reduction, one can select "master". In this case, old master calibrations are associated even if there are raw calibrations closer in time (and therefore more likely to ensure better quality products).

The quality level that the selected calibrations deliver is indicated close to each dataset in the ‘Raw input‘ tab, under the colum ‘CalibLevel‘. CalibLevel=0 indicates that calibrations that follow the rules of the instrument calibration plans have been selected. The higher the number, the poorer the quality of the products.

4.2 Configuration of parameters: the configuration editor

The data reduction of each dataset can be configured according to the scientific needs using an appropriate configuration editor.

The EDPS workflows contain two types of parameters and they both have default values that can be modified to improve the data reduction.

- **Workflow parameters** (for some workflows only) are global and they are applied to the entire workflow. They are accessible both in the ‘Raw Data‘ tab, prior to the creation of a dataset, and in the ‘Reduction Configuration‘ editor, in the ‘Reduction queue‘ tab. Note: some workflow parameters were already configured before creating the dataset and sending it to the reduction queue. Here, they can be changed again. Please, note that the parameters have an effect only on the files that are already in the dataset. If one specifies a parameter that should include extra files in the dataset (e.g., the inclusion of more calibrations), files are not added and the reduction might fail. If you need to change a parameter that modifies the dataset content, please go back to the Raw data tab and create a new dataset.

- **Recipe parameters** are specific to the individual recipes and can be configured per task. They are accessible in the 'Reduction Configuration' editor, in the 'Reduction queue' tab.

This editor allows to configure the data reduction for a given dataset by specifying workflow and recipe parameters.

Note: some workflow parameters were already configured before creating the dataset and sending it to the reduction queue. Here, they can be changed again. Please, note that the parameters have an effect only on the files that are already in the dataset. If one specifies a parameter that should include extra files in the dataset (e.g., the inclusion of more calibrations), files are not added and the reduction might fail. If you need to change a parameter that modifies the dataset content, please go back to the Raw data tab and create a new dataset.

To open the editor, click on the wheel button  next to the dataset you desire to configure the reduction for. A window with the configuration editor appears as shown Figure 16.

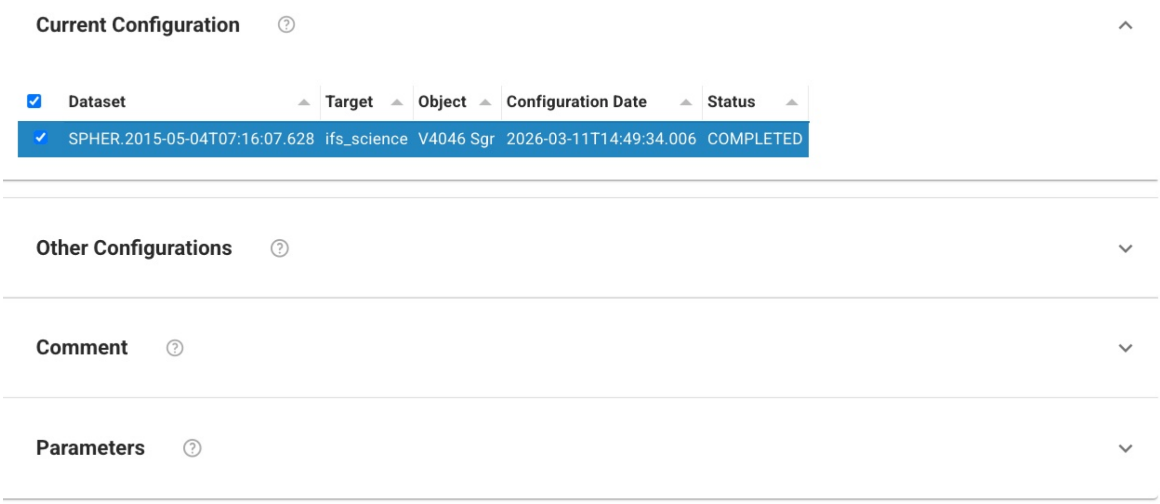

Figure 16: The Reduction Configuration editor. It contains 4 sections, that indicate the current configuration, list of other configurations to set, comments to insert, and the parameters to modify.

The editor is divided into 4 parts, which can be accessed pressing the corresponding expansion arrow.

- **Current configuration.** It indicates the name of the selected configuration for a given dataset (Figure 17).
- **Other configurations.** It allows to specify other configurations, to which the changes shall be copied to (Figure 18).
- **Comment.** It allows to specify a comment to describe the configuration. It is possible to append or replace a comment (Figure 19). Comments can be changed on all configurations. It is possible to save the comment for the current configuration only, or for all the selected configurations.
- **Parameters.** A window as in Figure 20 appears.

  The window allows to:

- Select the parameter set. A pre-determined list of workflow parameters and recipe parameters for a given use case. For the majority of the cases, the "science" parameter set can be used.
- Edit the workflow parameters. These are parameters that regulates the reduction strategy, e.g. whether to use a given calibration or not, or to trigger a certain reduction step. Note that if the changes imply that some files not in the dataset are needed, the reduction might fail. In case, go back to the raw data tab, edit the workflow parameters there, and recreate the datasets.
- Edit the recipe parameters. These are parameters associated to the recipe of a given task. Note: the same recipe parameters can be configured differently for the tasks that run the same recipe. Default parameters are shown (albeit some parameters can be dynamic, e.g. ‘EDPS‘ changes their value depending on the type of input data).

Change the values according to the needs and then select whether to save it to the current or the selected configurations. Note, complete configurations cannot be modified, new configurations will be automatically created instead.

Figure 17: The first part of the Reduction Configuration Editor, that indicates the selected configuration.

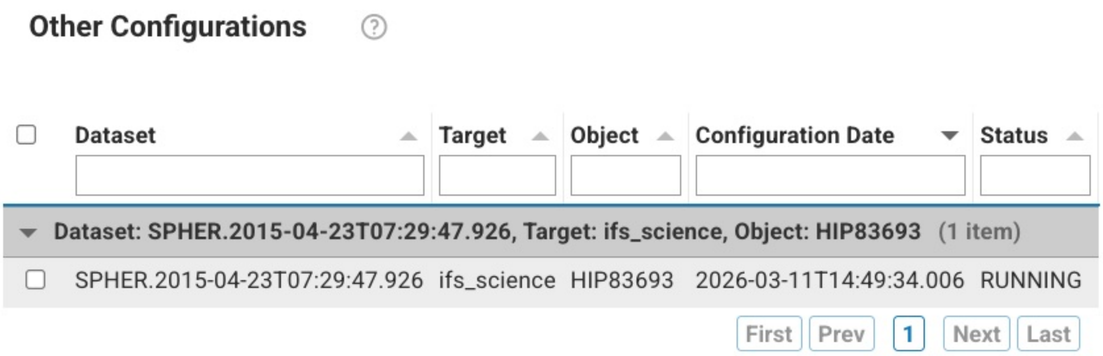

Figure 18: The second part of the Reduction Configuration Editor, that indicates other configurations for which we’d like to apply the changes.

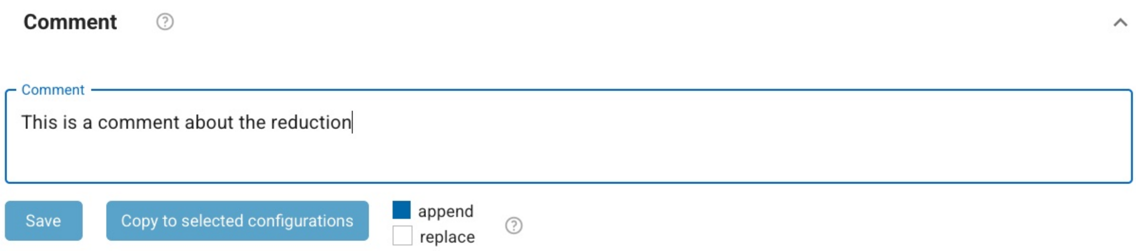

Figure 19: The third part of the Reduction Configuration Editor, that allows to specify a comment to the selected configurations.

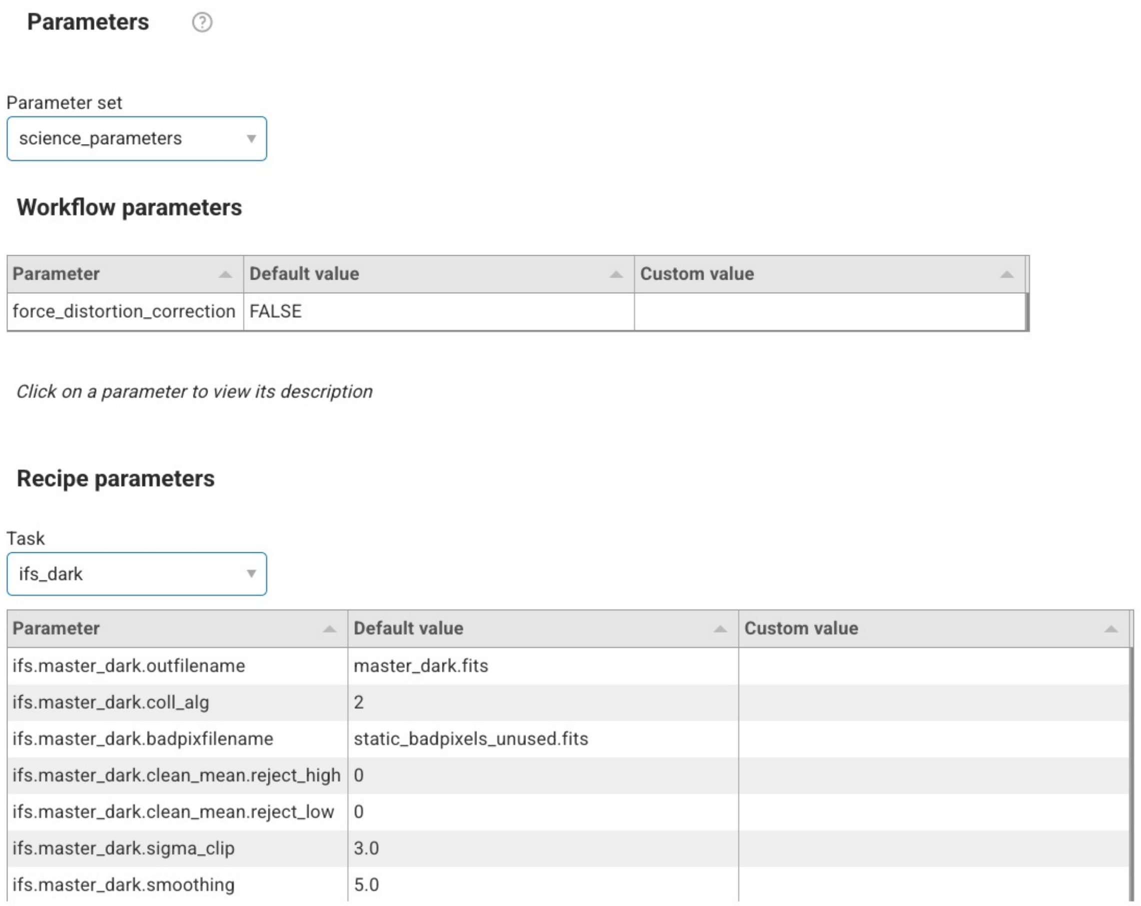

Figure 20: The fourth part of the Reduction Configuration Editor, that allows to specify the parameters sets and the recipe parameter per task. These settings can be applied to the "Selected Configuration" (Fig. 17) or to the "Other Configurations (Fig. 18).

4.3 Troubleshooting

This section provides guidance on configuring the reduction, as well as diagnosing and resolving common issues encountered during the SPHERE IFS data-reduction cascade.

4.3.1 Master darks

The recipe `sph_ifs_master_dark` produces the detector dark-current calibration (`IFS_MASTER_DARK`). The companion task `ifs_static_bad_pixel_map` generates the static bad-pixel map

(`IFS_STATIC_BADPIXELMAP`) from the same calibration frames.

If an excessive number of bad pixels are detected, increasing `ifs.master_dark.sigma_clip` from its default value of 3 to 5 makes the clipping less aggressive and generally reduces the number of flagged pixels.

4.3.2 Distortion map

The recipe `sph_ifs_distortion_map` measures the geometric distortion of the IFS lenslet grid. During the development of the SPHERE pipeline, however, distortion solutions derived from calibration data acquired close in time were found to exhibit significant variations, including changes in the derived central axes. Applying these solutions to science data may therefore introduce artificial distortions rather than correct them.

For this reason, the EDPS workflow sets the workflow parameter `force_distortion_correction` to `FALSE` by default for science and Internal Data Products (IDP) reductions, although the distortion calibration may still be associated. For QC0 reductions, this parameter is set to `TRUE`.

The dominant distortion component is a stable anamorphism of approximately 0.6%. This can be corrected reproducibly by rescaling the detector coordinates, multiplying the X and Y axes by factors of 1.0059 and 1.0011, respectively.

4.3.3 Wavelength calibration

The SPHERE consortium identified systematic wavelength offsets when using the `IFS_SPECPOS` product generated by `sph_ifs_spectra_positions`. To mitigate this issue, an empirical correction was developed and implemented within the consortium's IDL reduction software. The corresponding implementation in the pipeline does not yet reproduce the same level of performance and remains under evaluation.

To facilitate independent verification of the wavelength solution, `sph_ifs_wave_calib` produces a wavelength-calibrated cube of the arc-lamp exposure, allowing users to assess the quality and stability of the derived wavelength calibration.

4.3.4 Coronagraph centre

Coronagraph-centre observations are reduced using `sph_ifs_science_dr`. The centroid measurements obtained from OBJECT,CENTER observations have been reported to exhibit systematic offsets. Consequently, users are encouraged to verify the derived stellar position independently when high astrometric accuracy is required.

4.3.5 Spectral deconvolution

When `sph_ifs_science_dr` is executed with `ifs.science_dr.spec_deconv = TRUE`, the following recipe parameters are ignored and internally fixed:

- `ifs.science_dr.order = 5`
- `ifs.science_dr.user_cent = FALSE`
- `ifs.science_dr.cx = 0`
- `ifs.science_dr.cy = 0`
- `ifs.science_dr.reflambda = 1.0`
- `ifs.science_dr.fwhm = 2.0`

These values follow the recommendations adopted by the High Contrast Data Center.

4.3.6 Science reduction

Following recommendations from the SPHERE consortium, `ifs.science_dr.spec_deconv` is disabled by default because the current implementation is based on an earlier spectral-deconvolution algorithm. Since the intended science application is not known a priori, `ifs.science_dr.use_adi` is set to 0 by default.

Historically, bad-pixel and crosstalk corrections at the SPHERE Data Center were performed using dedicated IDL software. Equivalent implementations are now available within the pipeline and can be enabled through `ifs.science_dr.badpixco.apply` and `ifs.science_dr.xtalkco.apply`, both of which are disabled by default.

The large-scale crosstalk correction (`ifs.science_dr.xtalkco.largescale.apply`) may improve point-source contrast but can degrade observations of extended sources such as circumstellar discs. It should therefore be enabled only when appropriate for the science case.

5 List of workflow tasks

This table lists all the tasks and associated recipes in the SPHERE IFS workflow. Only some tasks are involved in the scientific reduction. These are marked as "yes" (executed by default) or "optional" (executed only when required by the workflow). Tasks marked as "no" are not part of the default science reduction; they are intended primarily for instrument monitoring and can be executed only when explicitly selected as the reduction target. When a task is selected as the target, all prerequisite tasks required to generate its input calibrations are executed automatically.

TASK	**RECIPE**	**Used in science reduction**	**Notes**
ifs_coronagraph_center	sph_ifs_science_dr	optional	Determines the stellar position behind the coronagraph.
ifs_dark	sph_ifs_master_dark	yes	Creates the master dark (`IFS_MASTER_DARK`).
ifs_det_flat_broad_band	sph_ifs_master_detector_flat	yes	Generates the broadband detector flat used for small-scale pixel-to-pixel response correction.
ifs_det_flat_narrow_band1	sph_ifs_master_detector_flat	yes	Produces the first narrow-band detector flat component.
ifs_det_flat_narrow_band2	sph_ifs_master_detector_flat	yes	Produces the second narrow-band detector flat component.
ifs_det_flat_narrow_band3	sph_ifs_master_detector_flat	yes	Produces the third narrow-band detector flat component.
ifs_det_flat_narrow_band4	sph_ifs_master_detector_flat	yes	Produces the fourth narrow-band detector flat component (associated only for the `PRI_YJH` disperser).
ifs_distortion_map	sph_ifs_distortion_map	no	Measures the geometric distortion of the IFS lenslet grid. Produces `IFS_DISTORTION_MAP` and, when needed, `IFS_POINT_PATTERN`.
ifs_ifu_flat	sph_ifs_instrument_flat	yes	Produces the IFU (lenslet) flat-field calibration used by subsequent reductions.
ifs_instrument_background	sph_ifs_cal_background	yes	Produces the instrumental background calibration.
ifs_science	sph_ifs_science_dr	yes	Main science reduction task applying background subtraction, flat-fielding, wavelength calibration and optional ADI or spectral deconvolution.
ifs_science_flux	sph_ifs_science_dr	optional	Processes FLUX observations obtained with the target offset from the coronagraph to measure the stellar flux.
ifs_sky_background	sph_ifs_cal_background	yes	Produces the sky-background calibration.
ifs_spectra_positions	sph_ifs_spectra_positions	yes	Associates detector pixels with lenslets and generates the initial Pixel Description Table (`IFS_SPECPOS`).
ifs_standard_astrometry	sph_ifs_science_dr	optional	Processes astrometric standard observations.
ifs_standard_flux	sph_ifs_science_dr	optional	Processes spectrophotometric standard observations.
ifs_static_bad_pixel_map	sph_ifs_master_dark	no	Produces the static bad-pixel map (`IFS_STATIC_BADPIXELMAP`).
ifs_wavelength_calibration	sph_ifs_wave_calib	yes	Refines the wavelength calibration and produces `IFS_WAVECALIB`.

Table 1: Tasks and associated recipes of the SPHERE IFS workflow.

Notes

- Background calibrations are associated in the following priority order: `SKY_BG` → `INS_BG` → `MASTER_DARK`.
- The science reduction is performed on a file-by-file basis. The pipeline does not combine multiple science exposures; any stacking or post-processing of multiple reduced frames must be performed afterwards.
- The `force_distortion_correction` workflow parameter is set to `FALSE` by default for science and IDP reductions (see Section 4.3.2). Although a distortion calibration may be associated by EDPS, it is not propagated to the science recipe. For QC0 processing, this parameter is set to `TRUE`.

6 Frequently Asked Questions

- **Q1) Where can I find the final reduced data?**

  Answer: all the products of all the datasets and the reductions are saved into the EDPS_data directory, specified when executing the `edps-gui` for the first time. One can decide to export only the final products for selected datasets and only for the desired reduction attempts into another location for further analysis. See Section 2.4 for further instructions.

- **Q2) How do I stop the application?**

  Answer: Proceed as follows:

  1. Press "Stop EDPS" in the Dashboard.
  2. Type Ctrl-C in the terminal where the application is running. If the application doesn't terminate, type Ctrl-C again.
  3. Alternatively, kill the 'panel serve' process on your system, for example:

     ```
     ps -e | grep panel # get the process ID of the gui (<pid>).
     kill -9 <pid>
     ```

- **Q3) I have closed the browser window where the application is running. How can I reopen the application?**

  Answer: Point your browser to: `http://localhost:5006/edps-gui`

- **Q4) Where can I find some data that I can use to test the application?**

  Answer: Install the 'datademo' package provided with the pipeline installation or download the "Demo Data" package from https://www.eso.org/sci/software/pipe_aem_table.html.

  Please note that the demo data can be large (tens of Gigabytes).

  A convenient script to download demo data for any pipeline is also available and can be used from the command line:

  ```
  curl -O https://eso.org/sci/software/apptainer/eso_download_demodata.sh
  bash ./eso_download_demodata.sh
  ```

- **Q5) How can I start the edps-gui if the following message appears?**

  ```
  Cannot start Bokeh server, port 5006 is already in use
  ```

  Answer: The panel server was not closed properly. Kill it by typing:

  ```
  ps -e | grep panel # get the process ID of the gui (<pid>).
  kill -9 <pid>
  ```

- **Q6) How do I get additional support on EDPS or data reduction in general?**

  Answer: For suggestions, questions, or feedback in general, please open a ticket with the EDPS Support team. This https://support.eso.org/new-ticket?ticket%5Bticket_field_13%5D%5Bdata%5D=227 should take you directly to a webpage for creating and EDPS feedback ticket, but in case you want to navigate there 'manually', go to https://support.eso.org, login, click on "Submit Helpdesk Ticket", and specify the Help topic: "Post Observations", "ESO Data Processing System [EDPS]".

- **Q7) I have a lot of disk space, but when I install `EDPS` with `pip` or an ESO pipeline with `Homebrew` I get the error message: `Cannot mkdir: No space left on device`. How do I fix it?**

  Answer: This depends on how much disk space is allocated to the `/home`, `/var`, and `/tmp` directories. The final solution would be to resize the space allocated to the in the organization of the filesystem. However, we list here few tricks that might do the job.

  - Clearing the pip .cache to make space for new packages. Type the command:

    ```
    pip cache purge
    ```

    before installing `EDPS`.

  - Redirect the cache, `Homebrew` temporary build directories into a partition with enough space. Set some of the following environmental variables in your `.bashrc` file:

    ```
    export HOMEBREW_CACHE=<path_to_new_cache_directory>
    export XDG_CACHE_HOME=<path_to_new_cache_directory>
    export HOMEBREW_TEMP=<path_to_new_temporary_directory>
    export TMPDIR=<path_to_new_temporary_directory>
    ```

    The first moves only the location of `Homebrew` cache, the second the cache of most applications (instead of the default `/home/username/.cache`), the third moves the directory where `Homebrew` builds, extracts, and saves temporary files (instead of the defaults `/tmp` and `/var/tmp`). The last changes the global system temporary directory and affects most of the linux commands.

  - As extreme measure, one can move the `/home/linuxbrew/.linuxbrew` directory somewhere else, and create a symbolic link in `/home/linuxbrew`. For example:

    ```
    cd /home/linuxbrew
    mv -f .linuxbrew <path_to_new_directory>
    ln -s <path_to_new_directory> .linuxbrew
    ```

    Important note: this operation might break some internal links. Recipes requiring external packages such as `telluriccorr` might not work (impacts on KMOS, XSHOOTER, FORS2, and MOLECFIT pipelines).