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 ZIMPOL Polarimetry EDPS-GUI tutorial

1 rev. 20260727

Issue 1

Date July 27 2026

Prepared:	M. Marsset	July 27 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	30/06/2026	all	First public release for 0.59.1

This page was intentionally left blank

Contents

1 Introduction

1.1 Scope

This document describes how to reduce SPHERE ZIMPOL Polarimetry 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 ZIMPOL Polarimetry 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 ZIMPOL Polarimetry pipeline itself, consult the pipeline manual available at: https://www.eso.org/sci/software/pipe_aem_table.html.

Note: this tutorial refers to:

- SPHERE ZIMPOL Polarimetry instrument pipeline named `spher`, version 0.59.1.
- SPHERE ZIMPOL Polarimetry workflow: `sphere.sphere_zimpol_polarimetry_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 ZIMPOL Polarimetry demo data. We assume that the `EDPS`, `edps-gui`, the SPHERE ZIMPOL Polarimetry 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).

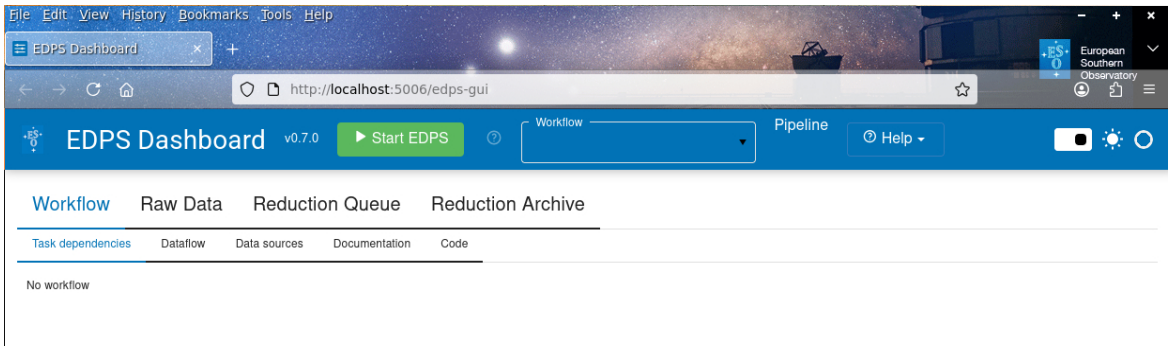

Figure 1: The empty `edps-gui` Dashboard; the underlying `EDPS` engine has not yet been started and no workflow has been loaded.

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

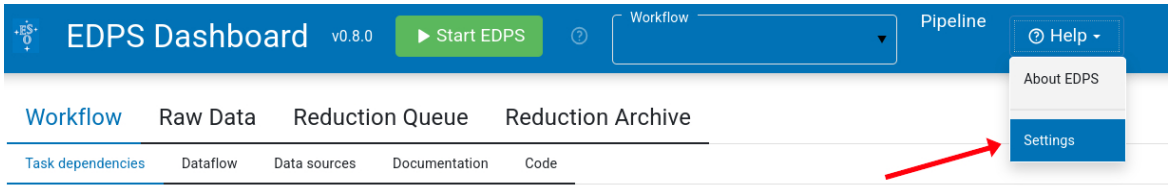

Figure 2: The “Help” –> “Settings” menu.

4. On the browser window with the dashboard, press the button ‘Start EDPS‘.
5. Choose the `sphere.sphere_zimpol_polarimetry_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.

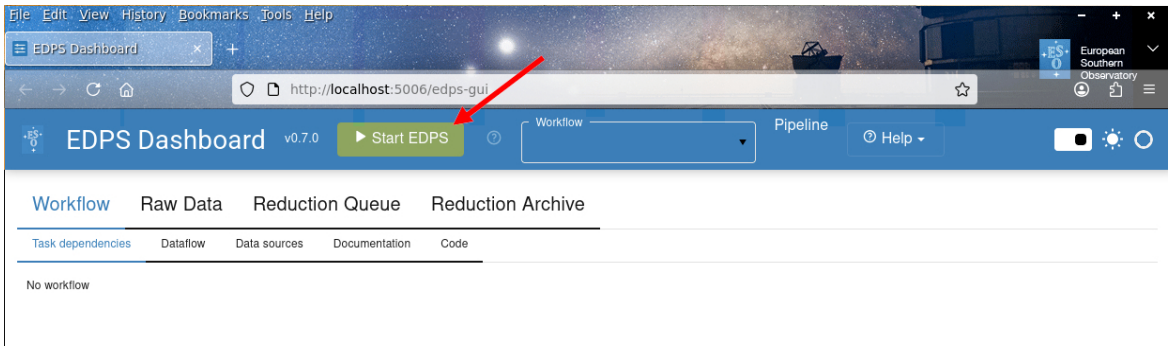

Figure 3: The “Start EDPS” button.

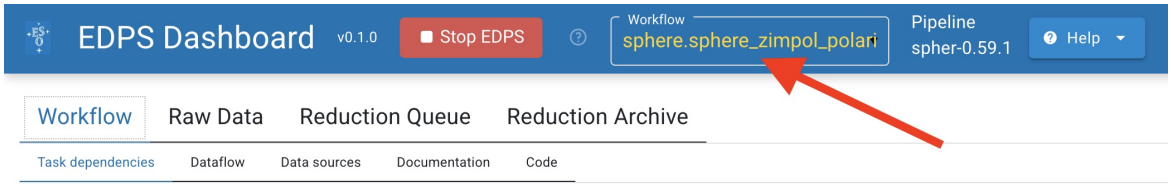

Figure 4: The `edps-gui` with the SPHERE ZIMPOL Polarimetry workflow loaded.

2.2 Selecting the input data

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

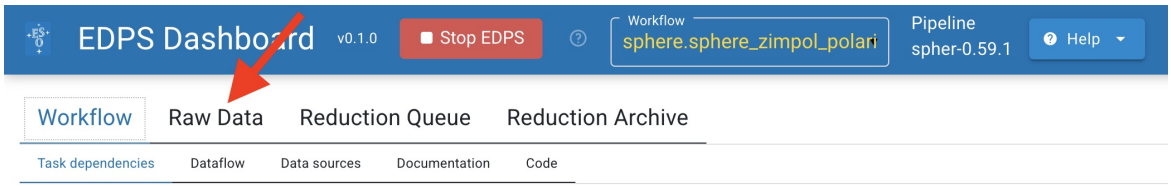

Figure 5: How to select RAW data Tab.

2. Press ‘Select Inputs‘. A selection window will appear that allows to select data that are stored on a local disk (Figure 6).

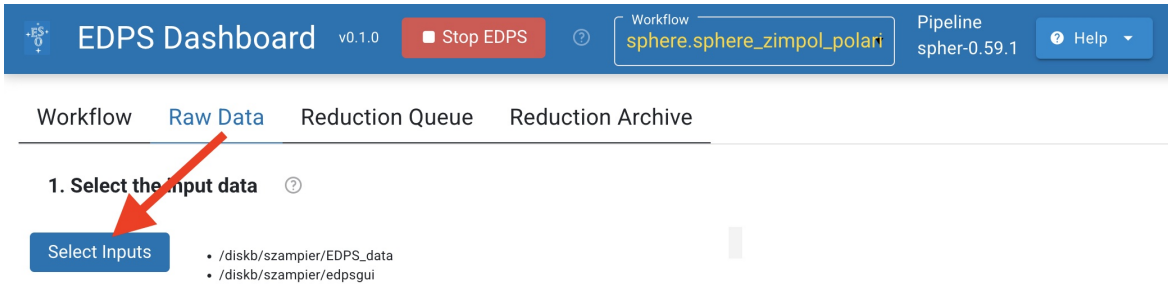

Figure 6: How to select input data.

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).

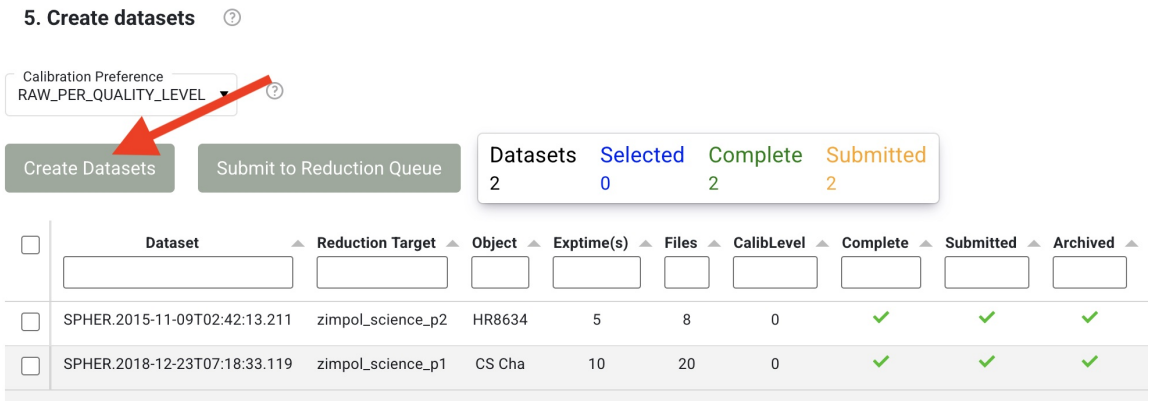

Figure 7: How to inspect the input data directory to create datasets.

5. Choose the datasets that should be processed (Figure 8)

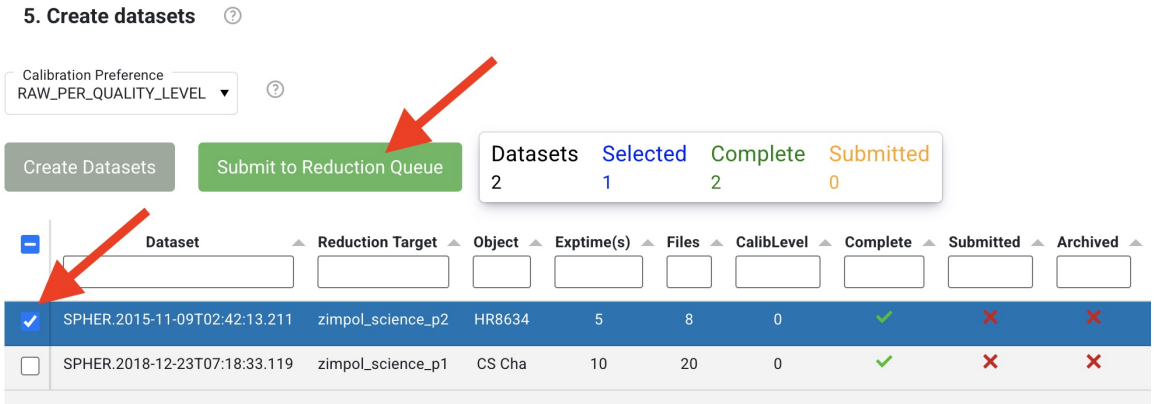

Figure 8: How to send the selected datasets to the Reduction Queue for processing.

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).
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.

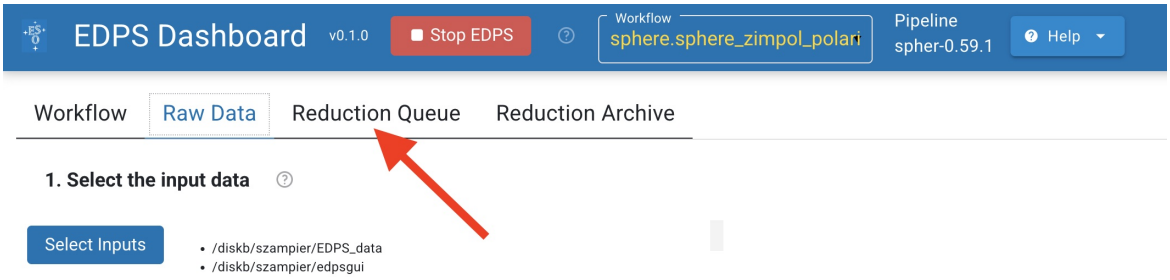

Figure 9: How to select Reduction Queue tab.

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

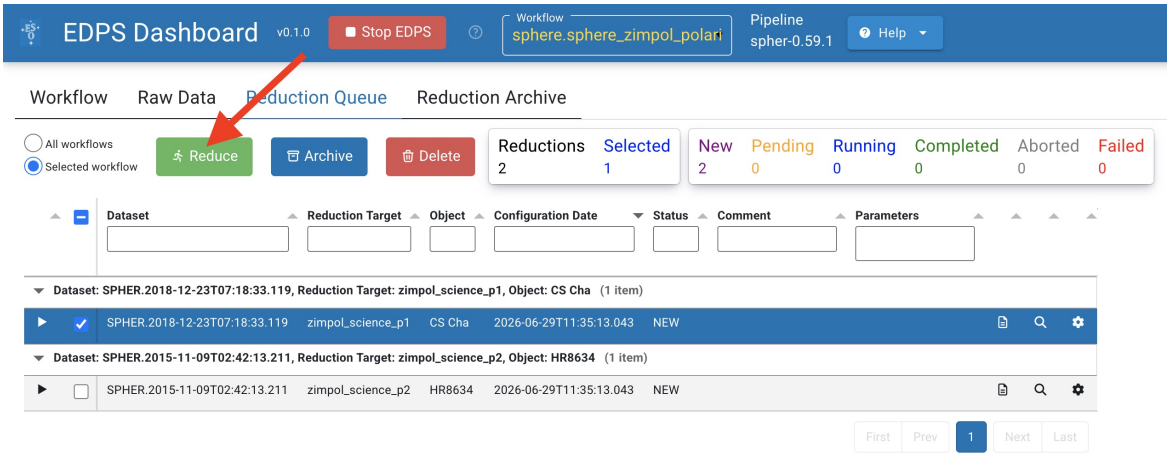

Figure 10: Reduce.

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.

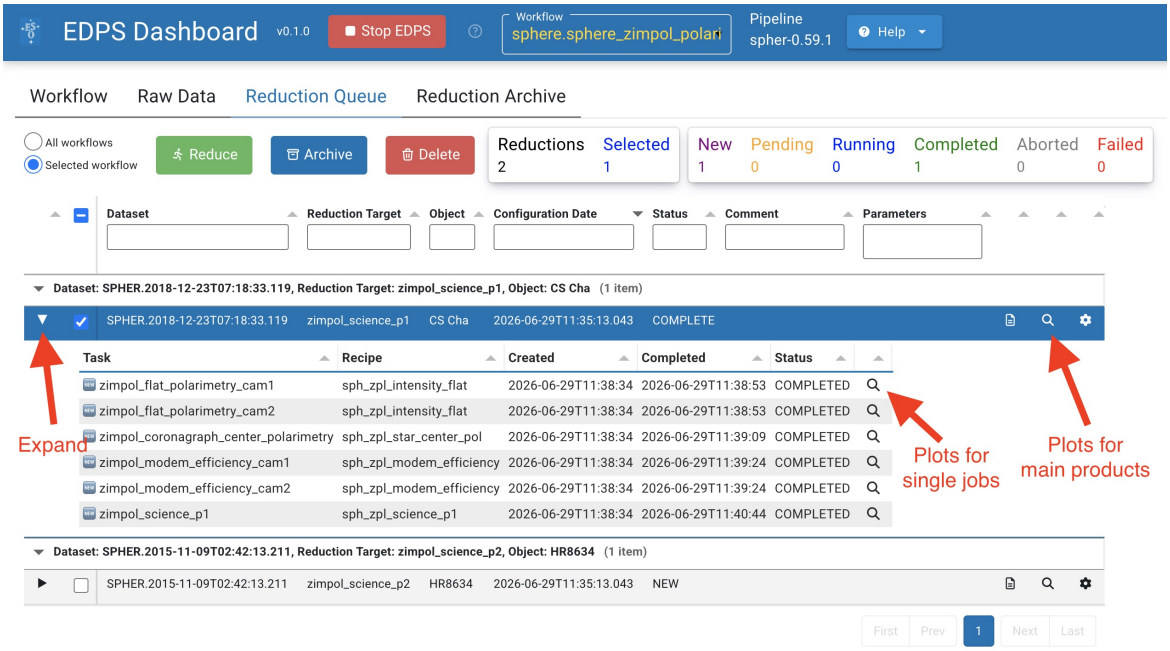

Figure 11: How to look for job information from the Reduction Queue tab.

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.

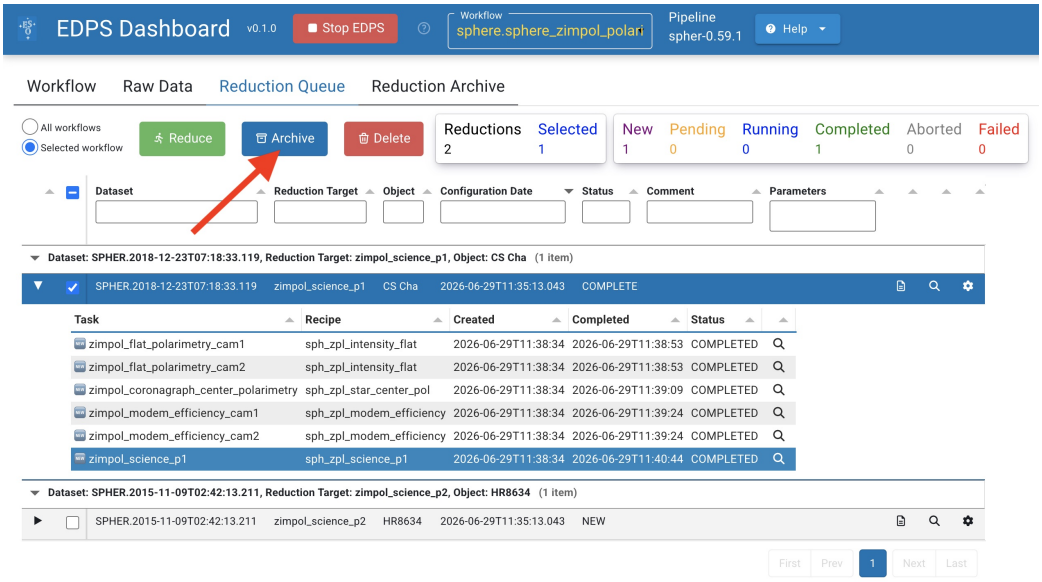

Figure 12: How to archive a completed reduction from the Reduction Queue tab.

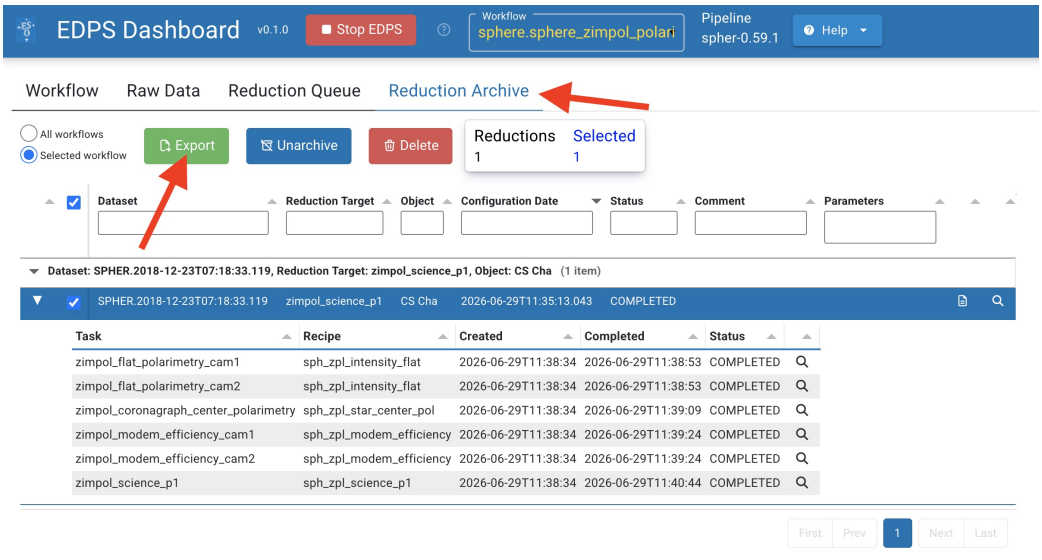

Figure 13: The reduction archive tab. This table contains all the different configurations of datasets that are declared "finished" and removed from the Reduction Queue. From this page, the user can export the most important files into a desired local directory.

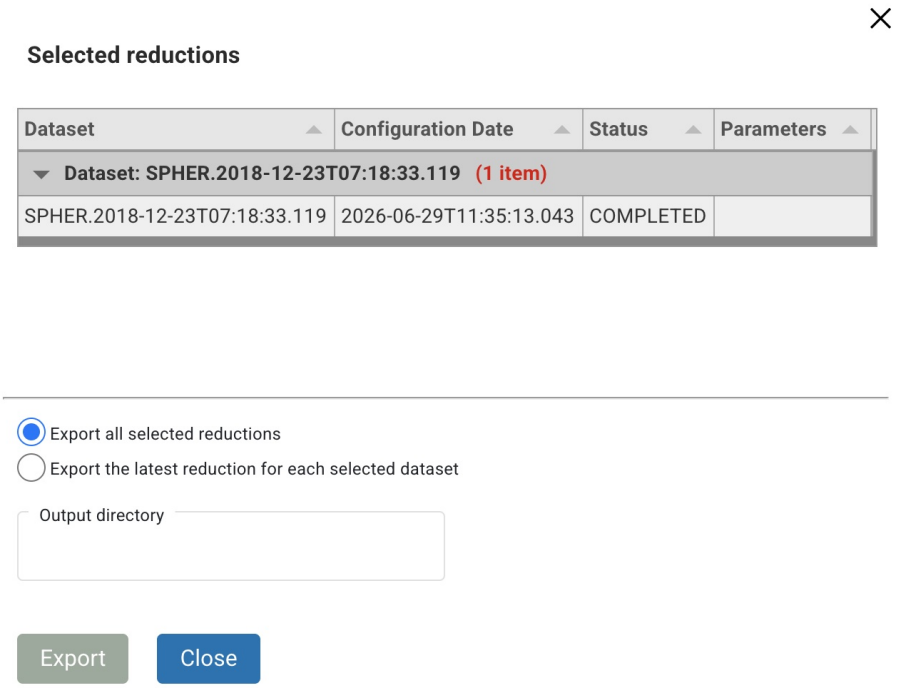

Figure 14: The EXPORT dialogue window, where the user can decide which reduced configuration to save and where.

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

The final products saved in the specified directory are the reduced and combined Stokes images from the two ZIMPOL cameras, with name format `ZIMPOL_` followed by the polarimetric mode (`POL1_` or `POL23_`), Stokes image (`U_` or `Q_`), camera number (`CAM1_` or `CAM2_`), and archive file name (corresponding to header keyword `ARCFILE`).

Each file contains eight FITS extensions. The first four correspond to the reduced science intensity image of the Q or U measurement and include the science image itself, a bad-pixel map, a number-of-combined-frames (ncomb) map, and an RMS map. The remaining four extensions contain the reduced polarization image of the Q or U measurement together with its associated bad-pixel map, ncomb map, and RMS map.

3 The SPHERE ZIMPOL Polarimetry data reduction flow.

The overall data flow of the SPHERE ZIMPOL Polarimetry 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 Internal Calibrations Polarimetry

Recipes:

`sph_zpl_master_bias`

`sph_zpl_master_dark`

`sph_zpl_intensity_flat`

`sph_zpl_modem_efficiency`

This subworkflow produces the internal calibration products required for ZIMPOL polarimetric data reduction. It generates the master bias, master dark, intensity flat fields, and the modulation–demodulation (modem) efficiency calibration for the two ZIMPOL cameras.

The processing proceeds in four steps:

- **zimpol_bias_polarimetry**: uses the pipeline recipe `sph_zpl_master_bias` to create the master bias. This calibration is associated with downstream tasks only in FastPolarimetry readout mode.
- **zimpol_dark_polarimetry**: uses `sph_zpl_master_dark` to generate the master dark. This calibration is used only for SlowPolarimetry exposures longer than or equal to 30 s, and may use the master bias when applicable.
- **zimpol_flat_polarimetry_cam1** and **zimpol_flat_polarimetry_cam2** use `sph_zpl_intensity_flat` to produce the intensity flat fields (pixel-to-pixel sensitivity calibrations) for each ZIMPOL camera.

SPHERE ZIMPOL polarimetry workflow

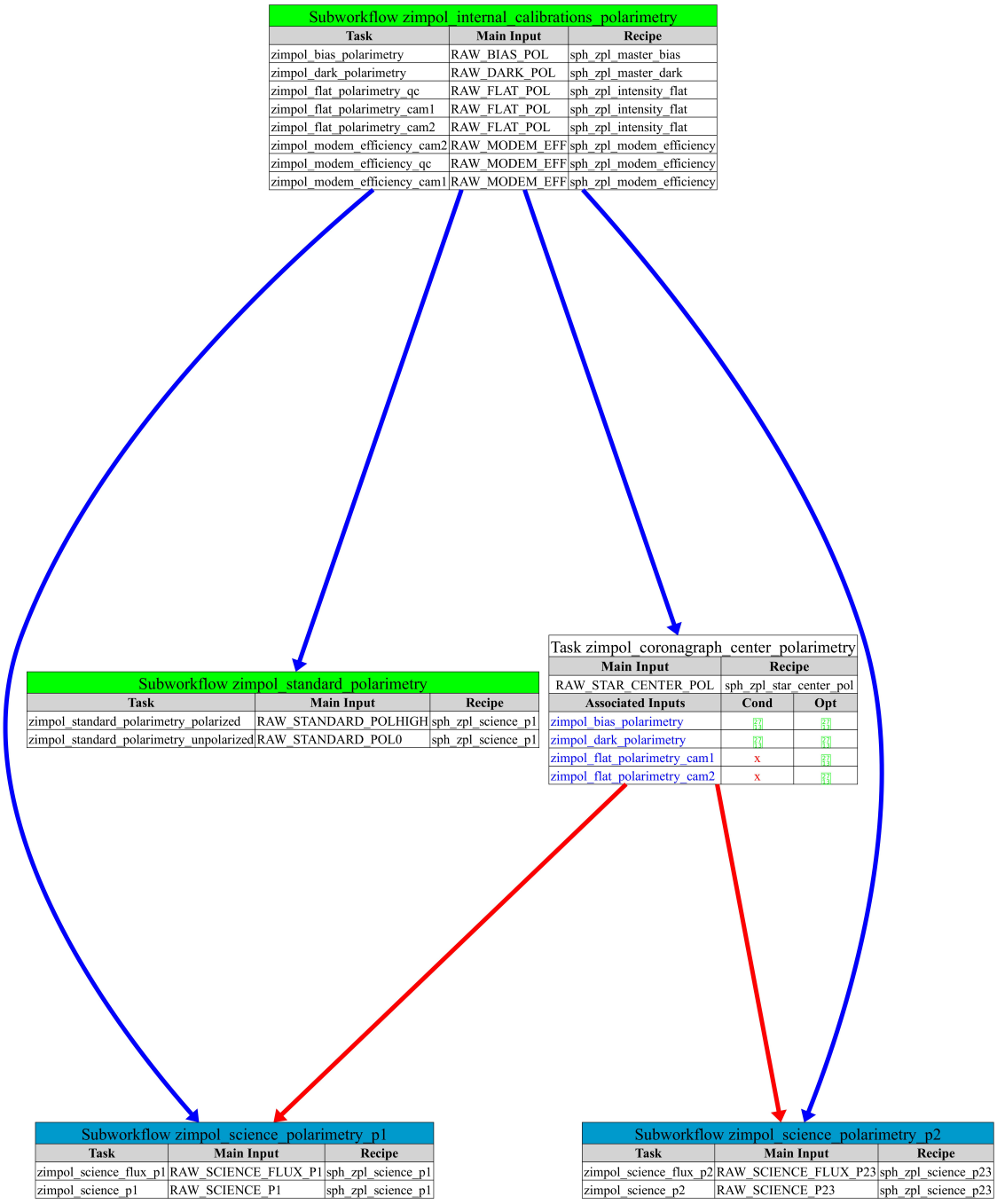

Figure 15: The data reduction cascade of the SPHERE ZIMPOL Polarimetry workflow.

- **zimpol_modem_efficiency_cam1** and **zimpol_modem_efficiency_cam2** use `sph_zpl_modem_efficiency` to produce the modem efficiency calibration. Modem-efficiency calibration frames (FLAT,POL100) are processed to measure the efficiency of the detector modulation–demodulation cycle. The modem efficiency is computed from the measured Stokes signals as the ratio P/I. The resulting modem efficiency products for both cameras are used as calibration inputs by the subsequent ZIMPOL polarimetric reduction recipes.

Customization Recipe parameters:

- `zpl.flat_polarimetry_cam?.coll_alg`: collapse algorithm (0 = Mean, 1 = Median, 2 = Clean Mean; default).
- `zpl.modem_efficiency_cam?.coll_alg`: collapse algorithm (0 = Mean, 1 = Median, 2 = Clean Mean; default).

3.2 Coronagraph Center

Recipe: `sph_zpl_star_center_pol`

The task **zimpol_coronagraph_center_polarimetry** executes the pipeline recipe to determine the stellar center position behind the coronagraph in polarimetric mode and produce calibrated star-center frames.

The task processes raw polarimetric star-center frames (`ZPL_STAR_CENTER_POL_RAW`). Optional calibration inputs include master bias, master dark, and intensity flat-field frames.

The recipe first performs a pre-processing step using `sph_zpl_preproc` to generate pre-processed frames for both ZIMPOL cameras. The calibrated frames are then de-dithered (if required) and saved as intermediate products. All calibrated frames are then combined using a collapse mean algorithm, producing a DOUBLE IMAGE product (8 FITS extensions) containing:

- a combined intensity image (I) with its bad-pixel map, number-of-combined frames map, and RMS map;
- a combined polarimetric image (P) with its bad-pixel map, number-of-combined frames map, and RMS map.

The combined star-center image is analyzed using an aperture detection algorithm that identifies bright regions (e.g., waffle satellite spots or a single star) and computes their geometric center, which defines the frame center.

The detected center coordinates are stored in the FITS headers for each camera:

- `DRS ZPL STAR CENTER IFRAME XCOORD`
- `DRS ZPL STAR CENTER IFRAME YCOORD`
- `DRS ZPL STAR CENTER PFRAME XCOORD`
- `DRS ZPL STAR CENTER PFRAME YCOORD`

The star-center products are used in subsequent science polarimetric reductions to define the rotation center during frame combination.

Customization Recipe parameters:

- `zpl.star_center.coll_alg`: collapse algorithm (0 = Mean; default, 1 = Median, 2 = Clean Mean).
- `zpl.star_center.sigma`: the sigma threshold for source detection (default = 10.0).
- `zpl.star_center.save_interprod`: enable to produce a field center table containing the calculated center positions (default = FALSE).

3.3 Subworkflow Standard Polarimetry

Recipe: `sph_zpl_science_p1`

This subworkflow processes polarimetric standard-star observations used for calibration monitoring. It reduces observations of unpolarized and highly polarized standard stars in order to assess the instrument's polarimetric performance.

Two tasks are executed:

- **zimpol_standard_polarimetry_unpolarized**: Processes observations of unpolarized standard stars to measure the level of instrumental polarization introduced by the instrument.
- **zimpol_standard_polarimetry_polarized**: Processes observations of highly polarized standard stars to determine possible offsets in polarization degree and polarization angle.

Both tasks run the polarimetric science recipe (`sph_zpl_science_p1`) and use the relevant calibration products, including bias, dark, intensity flat, and modulation–demodulation efficiency (modem) calibrations, as well as the polarimetric correction and filter tables. For polarized standards, an additional reference table of polarized stars is used.

The resulting products are used for quality control and instrument monitoring (qc1calib, calchecker) and are not required for the reduction of science observations.

Customization Recipe parameters:

- `zpl.science_p1.coll_alg`: collapse algorithm (0 = Mean; default, 1 = Median, 2 = Clean Mean).
- `zpl.science_p1.save_interprod`: enable to produce a field center table containing the calculated center positions (default = FALSE).

3.4 Subworkflow Science Polarimetry P1

Recipe: `sph_zpl_science_p1`

The subworkflow contains two tasks:

- **zimpol_science_flux_p1**, which reduces flux frames obtained with the star moved off the coronagraph;
- **zimpol_science_p1**, which reduces the science polarimetric observations themselves.

Both tasks use the pipeline recipe `sph_zpl_science_p1` and process polarimetric frames obtained in ZIMPOL P1 mode.

Optional calibration inputs include master bias, master dark, intensity flat-field, and modulation–demodulation efficiency (modem) calibration frames for each camera. A pre-processing step on the raw cubes is performed with `sph_zpl_preproc` to generate pre-processed frames.

The pre-processed science frames are organized into measurement groups corresponding to the Stokes parameters (Q measurements: Qplus and Qminus and U measurements: Uplus and Uminus).

The frames in each group are calibrated by subtracting the master bias and master dark frames and dividing by the appropriate intensity flat field, if available. The demodulation step then computes the intensity (I) and polarization component (P) frames. The polarization flat-field and modem efficiency corrections are subsequently applied.

The calibrated frames are then de-dithered and de-rotated, and saved as intermediate products. The center of rotation is taken from the star-center calibration frames (`ZPL_STAR_CENTER_POL_CAM1` and `ZPL_STAR_CENTER_POL` when available; otherwise, the geometric center of the frame (`xc = yc = 512`) is used.

In the next step, the calibrated frames of each measurement group are combined using the collapse mean algorithm, producing DOUBLE IMAGE products (8 FITS extensions) for each group containing:

- a combined intensity image (I) with its bad-pixel map, number-of-combined (ncomb) frames map, and RMS map;
- a combined polarimetric image (P) with its bad-pixel map, ncomb map, and RMS map.

Finally, the Qplus and Qminus frames (and similarly Uplus and Uminus) are combined polarimetrically:

Q:

$$I = \frac{I(+Q) + I(-Q)}{2}$$

$$P = \frac{P(+Q) - P(-Q)}{2}$$

U:

$$I = \frac{I(+U) + I(-U)}{2}$$

$$P = \frac{P(+U) - P(-U)}{2}$$

The reduced products from **zimpol_science_flux_p1** are used to measure the flux of the target in non-coronagraphic frames and are not themselves the final science products. The task **zimpol_science_p1** produces the final reduced Q and/or U DOUBLE IMAGE products for both cameras.

Customization Recipe parameters:

- `zpl.science_p1.coll_alg`: collapse algorithm (0 = Mean; default, 1 = Median, 2 = Clean Mean).
- `zpl.science_p1.save_interprod`: enable to produce a field center table containing the calculated center positions (default = FALSE).

3.5 Subworkflow Science Polarimetry P2

Recipe: `sph_zpl_science_p23`

The subworkflow contains two tasks:

- **zimpol_science_flux_p2**, which reduces flux frames obtained with the star moved off the coronagraph;
- **zimpol_science_p2**, which reduces the science polarimetric observations themselves.

Both tasks use the pipeline recipe `sph_zpl_science_p23` and process polarimetric frames obtained in ZIMPOL P23 modes.

Optional calibration inputs include master bias, master dark, intensity flat-field, and modulation–demodulation efficiency (modem) calibration frames for each camera. A pre-processing step on the raw cubes is performed with `sph_zpl_preproc` to generate pre-processed frames.

The pre-processed science frames are organized into measurement groups corresponding to the Stokes parameters (Q measurements: Qplus and Qminus and U measurements: Uplus and Uminus).

The frames in each group are calibrated by subtracting the master bias and master dark frames and dividing by the appropriate intensity flat field, if available. The demodulation step then computes the intensity (I) and polarization component (P) frames. The polarization flat-field and modem efficiency corrections are subsequently applied.

The calibrated frames are then de-dithered and saved as intermediate products.

In the next step, the calibrated frames of each measurement group are combined using the collapse mean algorithm, producing DOUBLE IMAGE products (8 FITS extensions) for each group containing:

- a combined intensity image (I) with its bad-pixel map, number-of-combined (ncomb) frames map, and RMS map;

- a combined polarimetric image (P) with its bad-pixel map, ncomb map, and RMS map.

Finally, the Qplus and Qminus frames (and similarly Uplus and Uminus) are combined polarimetrically:

Q:

$$I = \frac{I(+Q) + I(-Q)}{2}$$

$$P = \frac{P(+Q) - P(-Q)}{2}$$

U:

$$I = \frac{I(+U) + I(-U)}{2}$$

$$P = \frac{P(+U) - P(-U)}{2}$$

The reduced products from **zimpol_science_flux_p2** are used to measure the flux of the target in non-coronagraphic frames and are not themselves the final science products. The task **zimpol_science_p2** produces the final reduced Q and/or U DOUBLE IMAGE products for both cameras.

Customization Recipe parameters:

- `zpl.science_p23.coll_alg`: collapse algorithm (0 = Mean; default, 1 = Median, 2 = Clean Mean).
- `zpl.science_p23.save_interprod`: enable to produce a field center table containing the calculated center positions (default = FALSE).

3.6 Others

Additional quality-control tasks (zimpol_flat_polarimetry_qc and zimpol_modem_efficiency_qc) execute the same recipes for calibration monitoring and are not used directly in science reductions.

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.

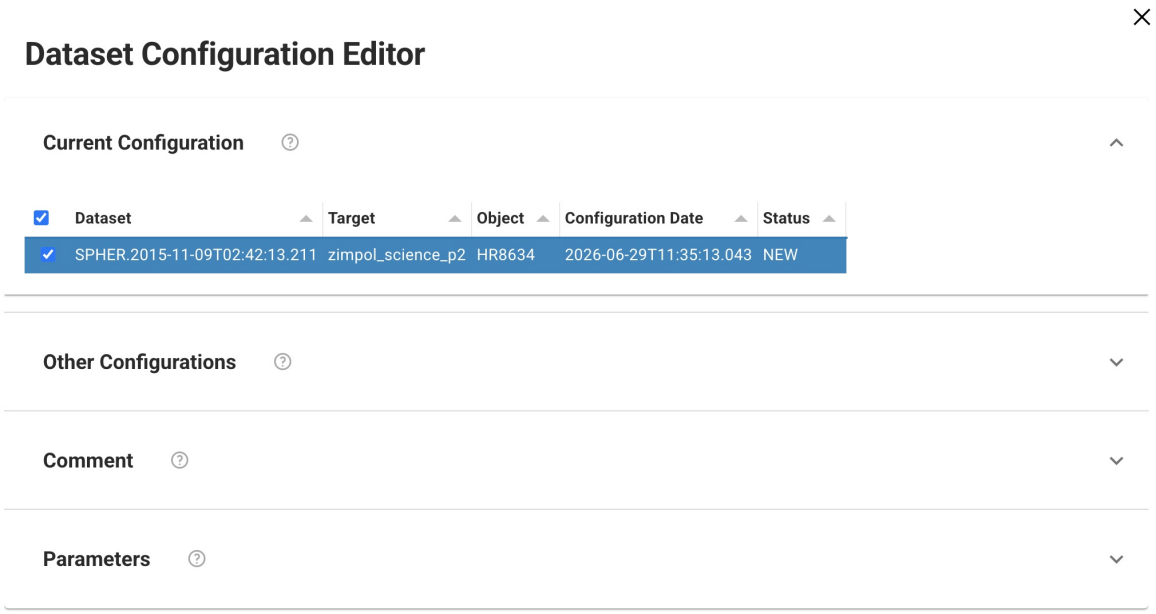

Figure 16: The Reduction Configuration editor. It contains 4 sections, that indicate the current configuration, list of other configurations to set, comments to insert, and the parameters to modify.

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.

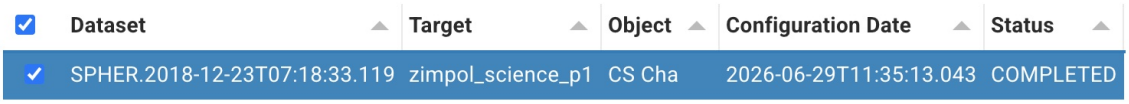

Figure 17: The first part of the Reduction Configuration Editor, that indicates the selected configuration.

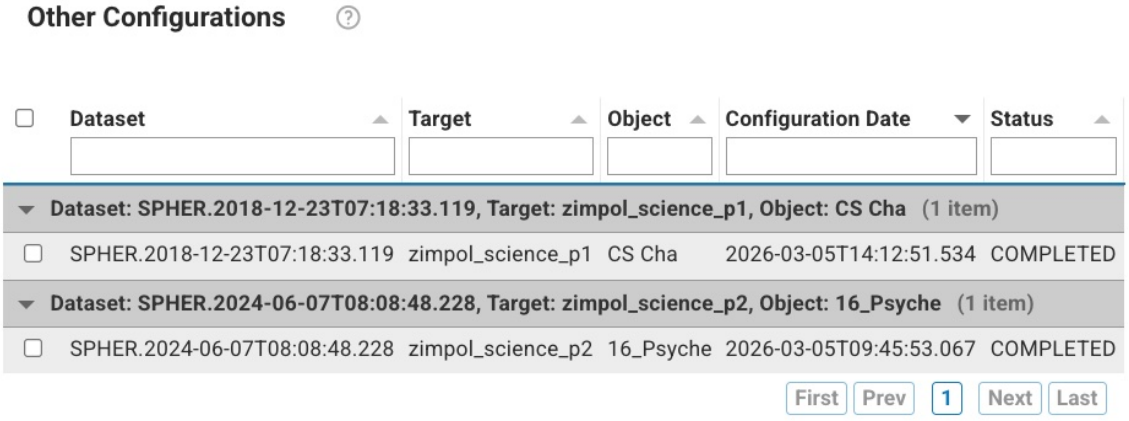

Figure 18: The second part of the Reduction Configuration Editor, that indicates other configurations for which we'd like to apply the changes.

4.3 Troubleshooting

This section provides guidance for diagnosing and resolving common issues encountered during the SPHERE ZIMPOL Polarimetry data-reduction cascade.

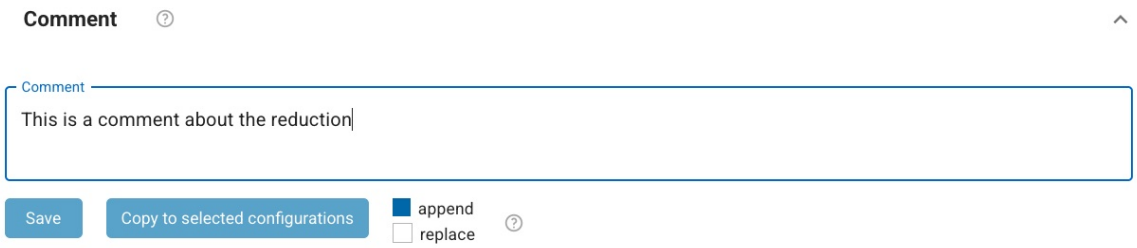

Figure 19: The third part of the Reduction Configuration Editor, that allows to specify a comment to the selected configurations.

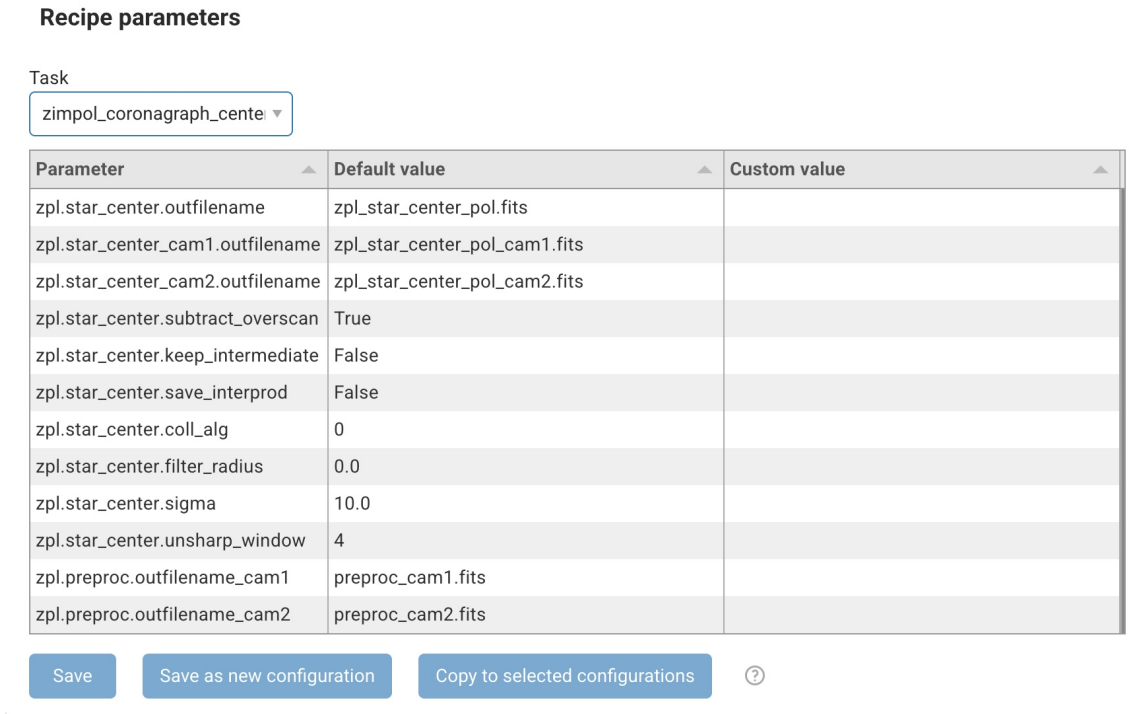

Figure 20: The fourth part of the Reduction Configuration Editor, that allows to specify the parameters sets and the recipe parameter per task. These settings can be applied to the "Selected Configuration" (Fig. 17) or to the "Other Configurations (Fig. 18).

4.3.1 Vertical pairs of bright and dark pixels

For long exposure times (`DET.DIT1` $>$ 50 s), charge traps may appear in the ZIMPOL detector. Those are defects in the CCD lattice that temporarily capture electrons during the charge transfer and release them with a time delay, producing vertical pairs of dark and bright pixels. See Figure 21 as an example.

Small errors in the charge transfer accumulate over time. The effect can appear at shorter exposure times in FastPolarimetry mode and may also be present in calibration frames such as `FLAT,LAMP` or `FLAT,POL100`.

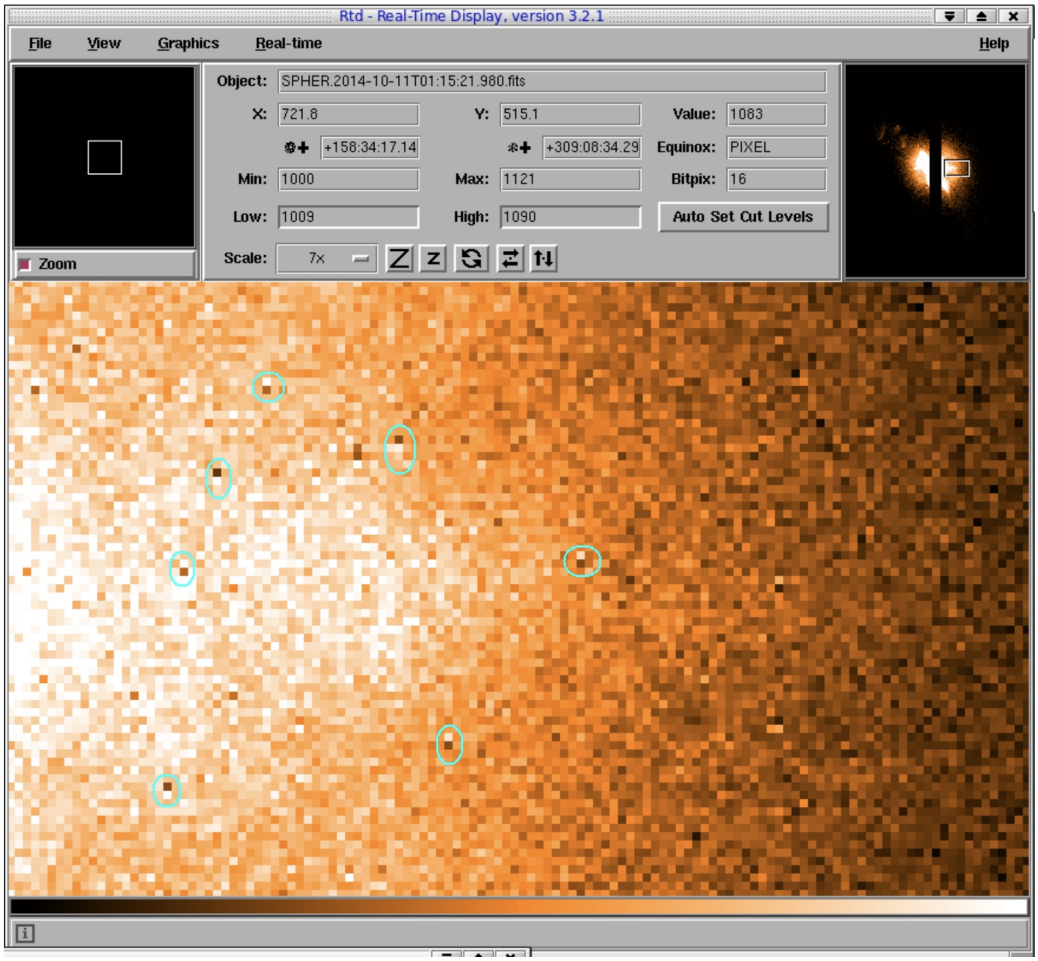

Figure 21: Vertical pairs of bright and dark pixels marked in cyan caused by charge traps in the ZIMPOL detector.

4.3.2 Image Incorrectly Rotated / Signal Smeared out

The task **zimpol_coronagraph_center_imaging** uses the instrument pipeline recipe `sph_zpl_star_center_img` to identify waffle spots and determine the position of the target behind the coronagraph. In some cases the detection threshold of 10σ is too high and the recipe cannot identify the target's position. In such a case the science recipe (`sph_zpl_science_imaging`) will rotate the frames around an incorrect position which will smear out the signal. If you see such behaviour you may try to reduce `star_center_img.sigma` to 5.

5 List of workflow tasks

This table lists all the tasks and associated recipes in the SPHERE ZIMPOL Polarimetry 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
zimpol_bias_polarimetry	sph_zpl_master_bias	no	Creates the polarimetric master bias for both cameras.
zimpol_dark_polarimetry	sph_zpl_master_dark	optional	Produces the polarimetric master dark for both cameras.
zimpol_flat_polarimetry_qc	sph_zpl_intensity_flat	no	Produces intensity flat for QC and monitoring.
zimpol_flat_polarimetry_cam1	sph_zpl_intensity_flat	optional	Creates the polarimetric intensity flat field for camera 1.
zimpol_flat_polarimetry_cam2	sph_zpl_intensity_flat	optional	Creates the polarimetric intensity flat field for camera 2.
zimpol_modem_efficiency_qc	sph_zpl_modem_efficiency	no	Computes modem efficiency for quality control.
zimpol_modem_efficiency_cam1	sph_zpl_modem_efficiency	optional	Creates the modem efficiency for camera 1.
zimpol_modem_efficiency_cam2	sph_zpl_modem_efficiency	optional	Creates the modem efficiency for camera 2.
zimpol_standard_polarimetry_unpolarized	sph_zpl_science_p1	no	Reduces unpolarized standard stars (monitoring).
zimpol_standard_polarimetry_polarized	sph_zpl_science_p1	no	Reduces highly polarized standard stars (monitoring).
zimpol_coronagraph_center_polarimetry	sph_zpl_star_center_pol	optional	Determines stellar position behind the coronagraph.
zimpol_science_flux_p1	sph_zpl_science_p1	optional	Reduces off-axis flux frames for P1 observations.
zimpol_science_p1	sph_zpl_science_p1	yes	Polarimetric science reduction for P1 observations.
zimpol_science_flux_p2	sph_zpl_science_p23	optional	Reduces off-axis flux frames for P23 observations.
zimpol_science_p2	sph_zpl_science_p23	yes	Polarimetric science reduction for P23 observations.

Table 1: SPHERE ZIMPOL Polarimetry polarimetric workflow tasks overview.

Notes

- modem = modulation–demodulation.
- In FastPolarimetry, the master bias is associated when available.
- In SlowPolarimetry, the master dark is associated only for long exposures ($\geq$30 s).
- The coronagraph-center calibration is optional. When unavailable, the science recipes use the geometric frame center (`xc = yc = 512`) as the center of rotation.
- The workflow does not apply distortion-correction to the science.

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).