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

ESPRESSO EDPS-GUI tutorial

1 rev. 20260610

Issue 1

Date June 10 2026

Prepared:	Valentin Ivanov	June 10 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	10/06/2026	all	First public release for 3.3.19

This page was intentionally left blank

Contents

1 Introduction

1.1 Scope

This document describes how to reduce ESPRESSO data with the `edps-gui` (Graphic User Interface), the dashboard of the ESO Data Processing System (`EDPS`), which the recommended interface to reduce data from ESO telescopes. Details on the ESPRESSO 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 ESPRESSO pipeline itself, consult the pipeline manual available at: https://www.eso.org/sci/software/pipe_aem_table.html.

Note: this tutorial refers to:

- ESPRESSO instrument pipeline named `espresso`, version 3.3.19.
- ESPRESSO workflow: `espresso.espresso_wkf`
- `EDPS` version 1.5.7.
- `edps-gui` version 0.9.1.

Note: the ESPRESSO workflow needs the eso-tk pipeline package to be installed in the case one wants to combine individual spectra with the task **combine_science**. This is automatically done if the pipeline is installed via Hombebrew or rpm.

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 consist 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. This workflow can the 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 an 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 processes 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 (definying 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 path 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 ESPRESSO demo data. We assume that the `EDPS`, `edps-gui`, the ESPRESSO 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).

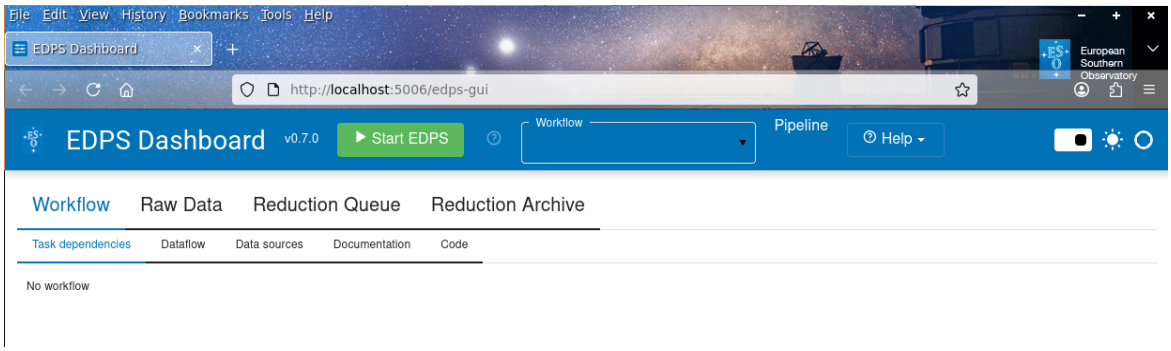

Figure 1: The empty `edps-gui` Dashboard; the underlying `EDPS` engine has not yet been started and no workflow has been loaded.

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

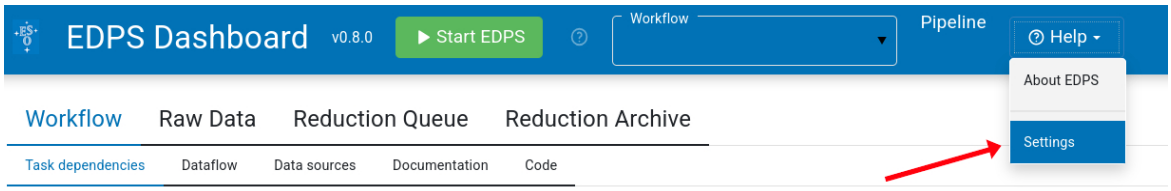

Figure 2: The “Help” –> “Settings” menu.

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

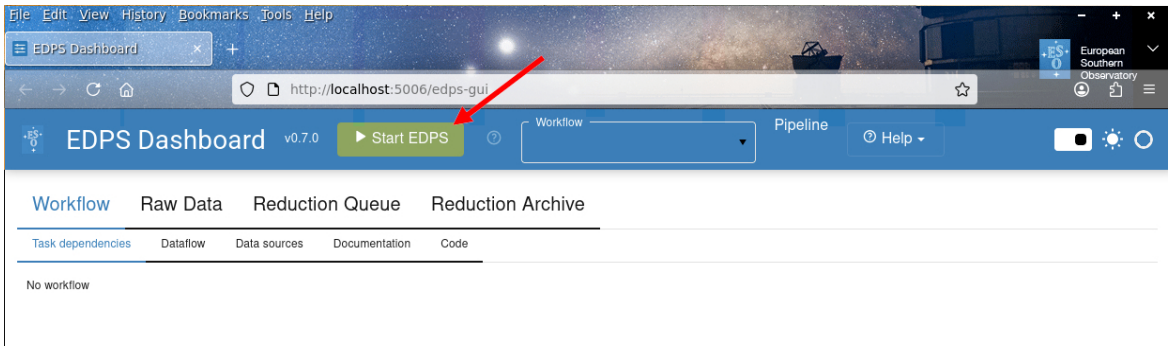

Figure 3: The “Start EDPS” button.

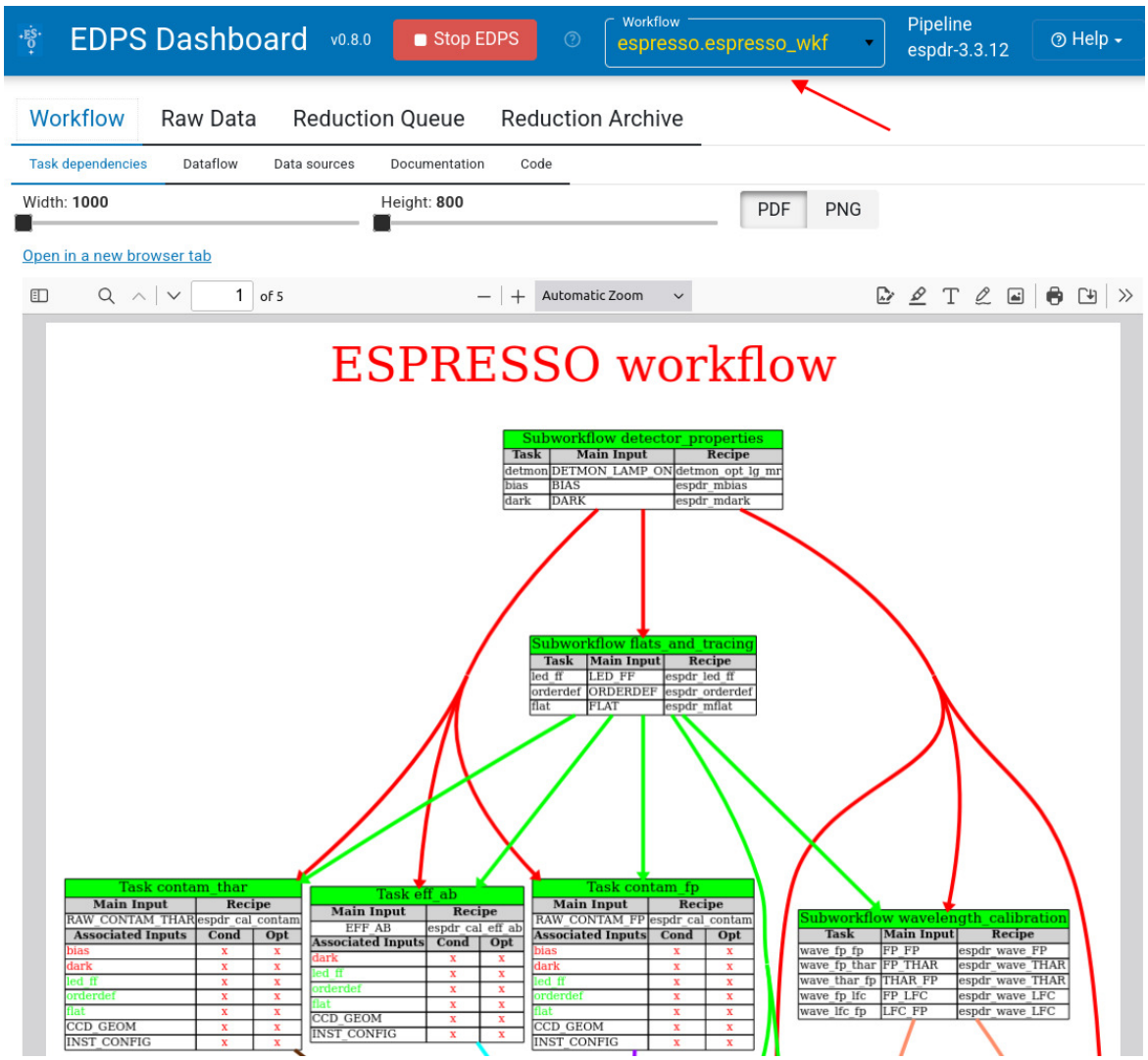

Figure 4: The `edps-gui` with the ESPRESSO workflow loaded.

2.2 Selecting the input data

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

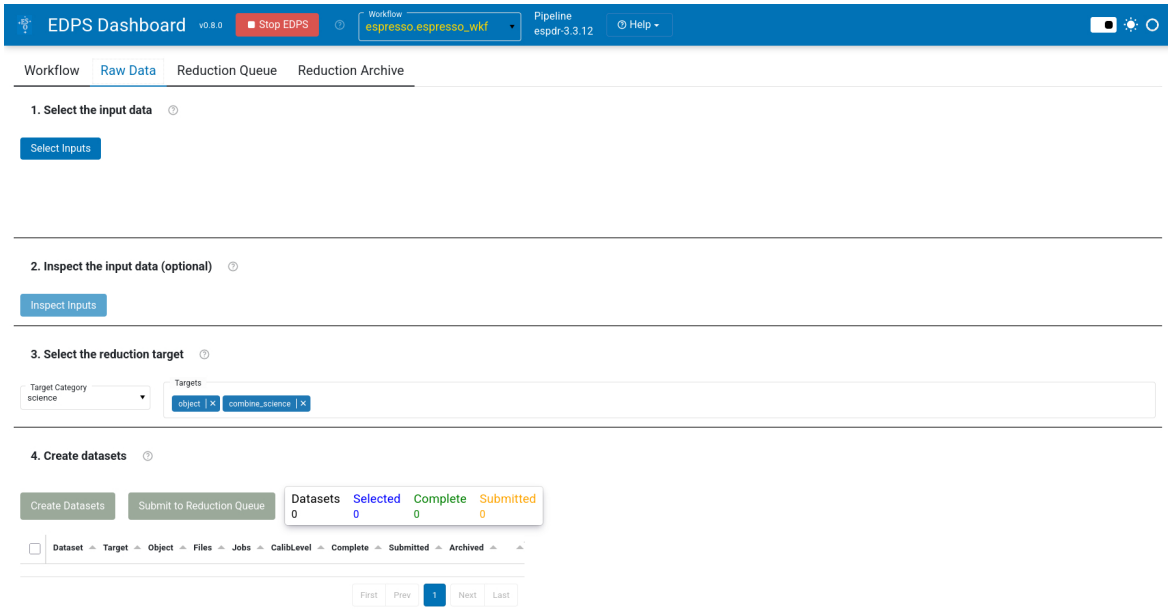

Figure 5: How to select RAW data Tab.

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

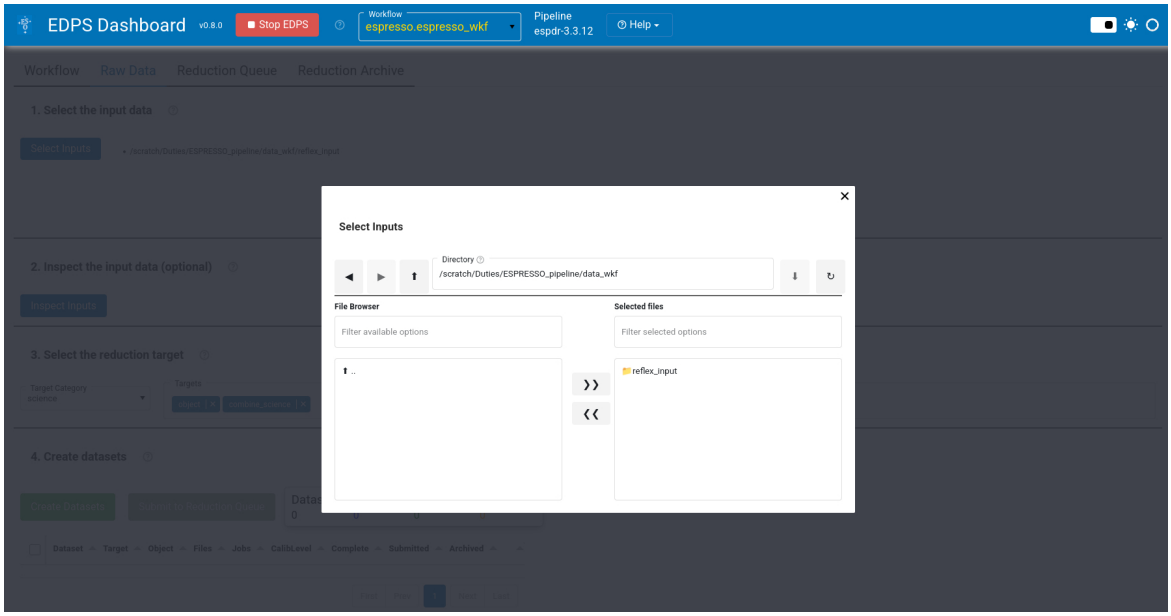

Figure 6: How to select input data.

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.

4. Press ‘Create Datasets‘. A list of datasets appears, one line for each set of science data (Figure 7).

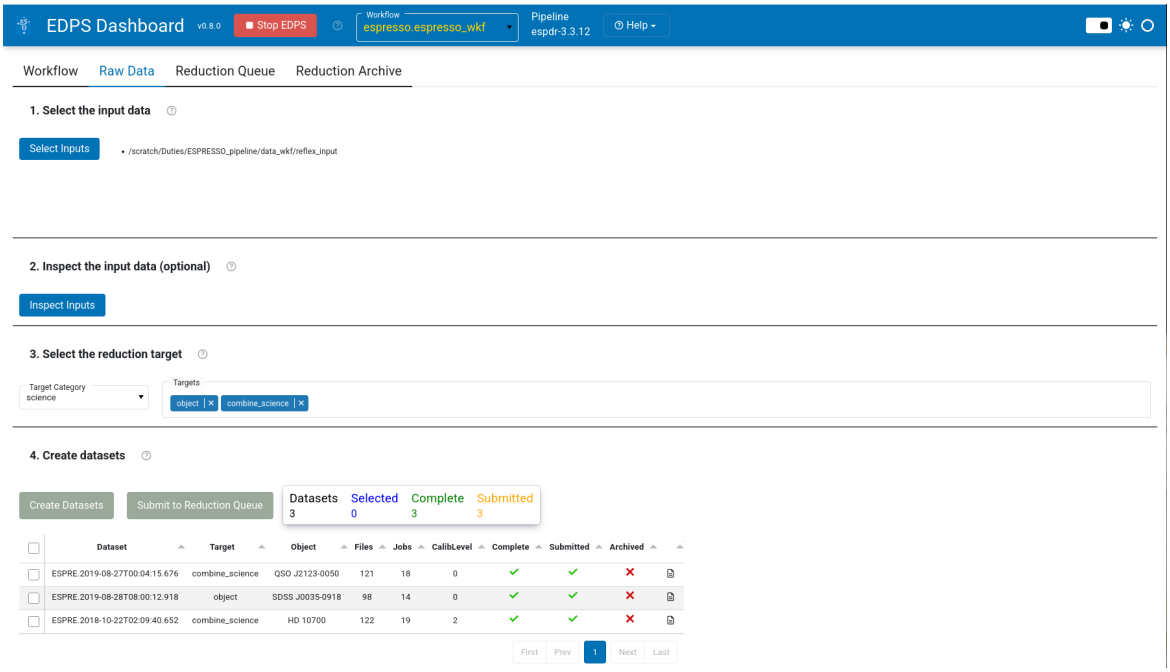

Figure 7: How to inspect the input data directory to create datasets.

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

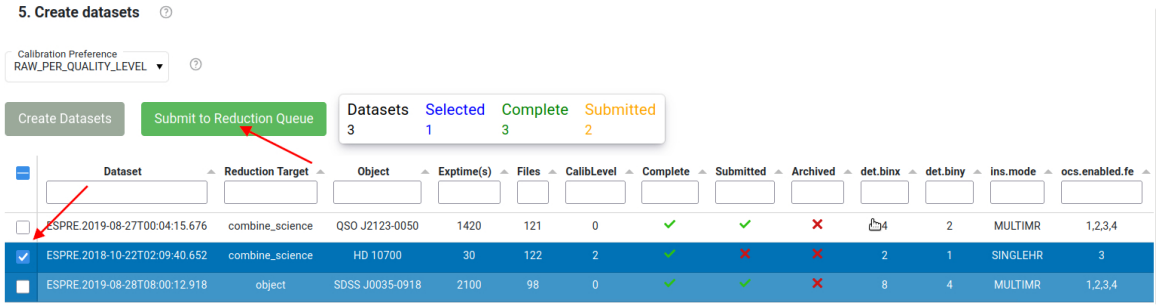

Figure 8: How to send the selected datasets to the Reduction Queue for processing.

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).

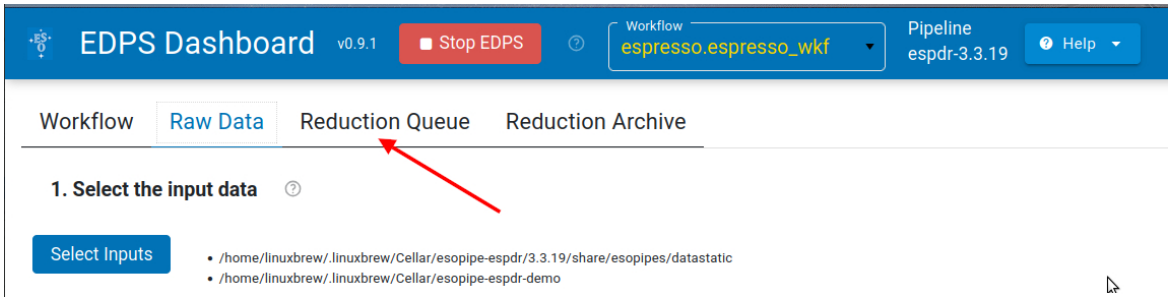

Figure 9: How to select Reduction Queue tab.

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.

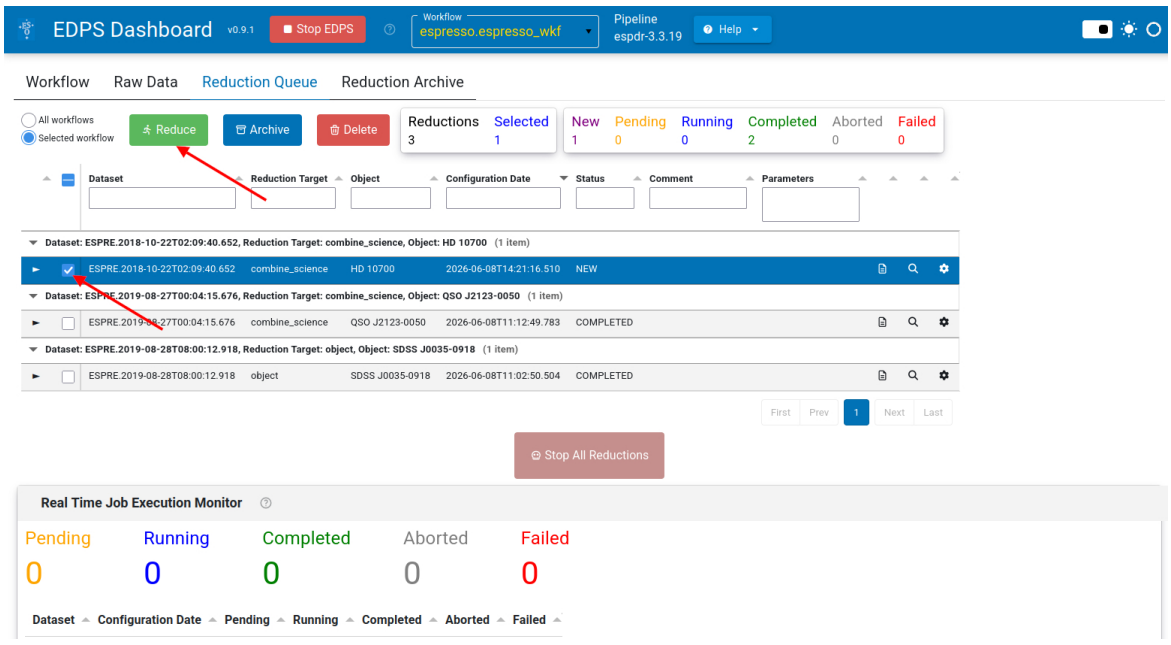

Figure 10: Reduce.

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.

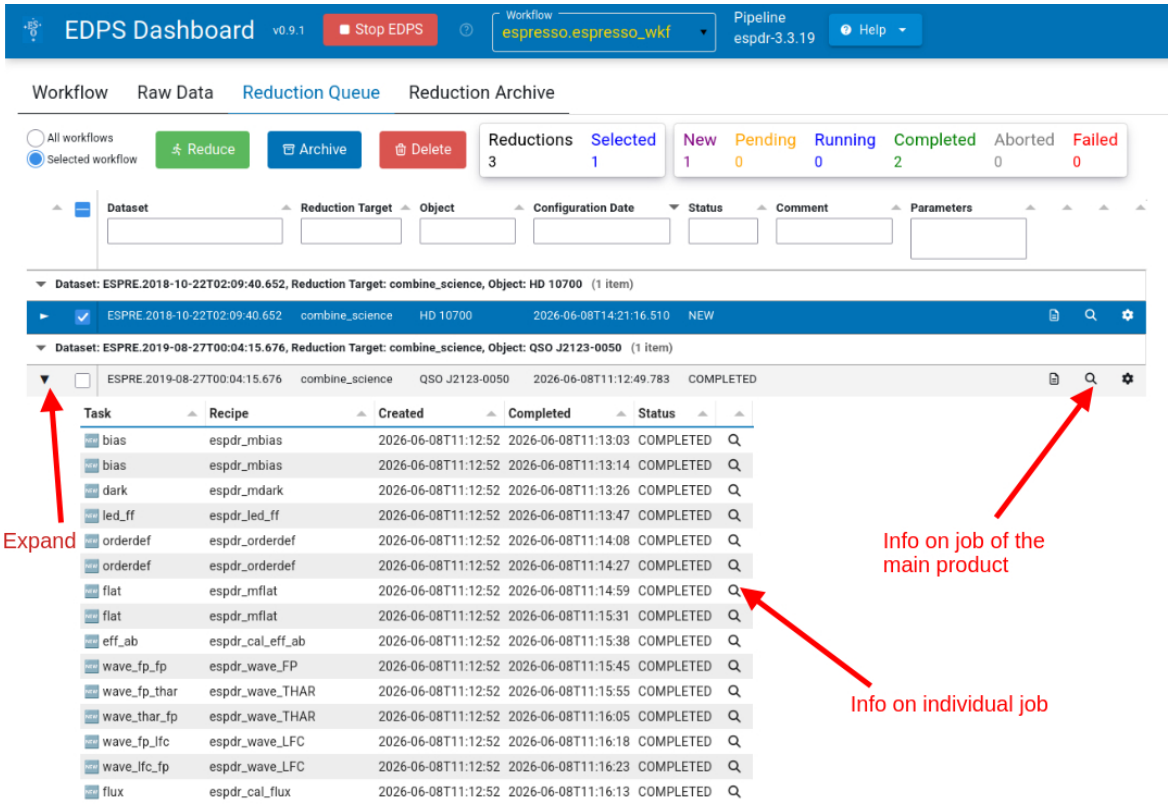

Figure 11: How to look for job information from the Reduction Queue tab.

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. Additionaly, even if all products for all tasks are saved in the EDPS_data

directory, the most important producs 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.

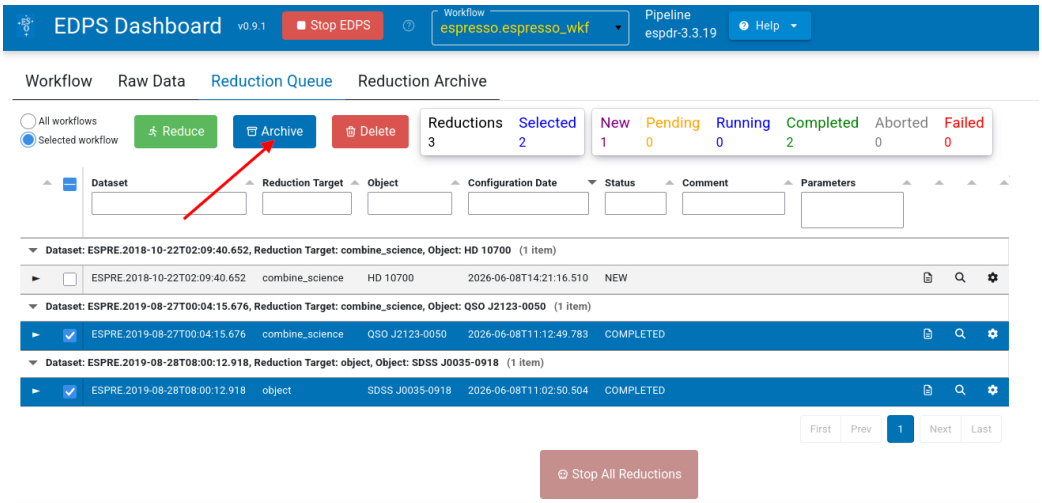

Figure 12: How to archive a completed reduction from the Reduction Queue tab.

2. Go in the Reduction Archive tab and click on the ’Export’ button. A new tab window appear where you can indicate the directory you want to copy your final products; finally press "Export" to copy the data.

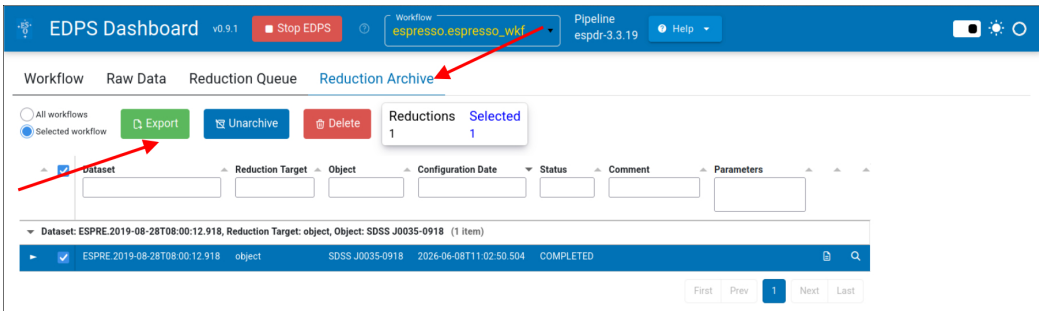

Figure 13: The reduction archive tab. This table contains all the different configurations of datasets that are declared "finished" and removed from the Reduction Queue. From this page, the user can export the most important files into a desired local directory.

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

The ESPRESSO workflow processes individual science observations, generating a separate product for each raw input file, but also combines together spectra from the same observing template (task `combine_science`).

The final products saved in the specified directory are:

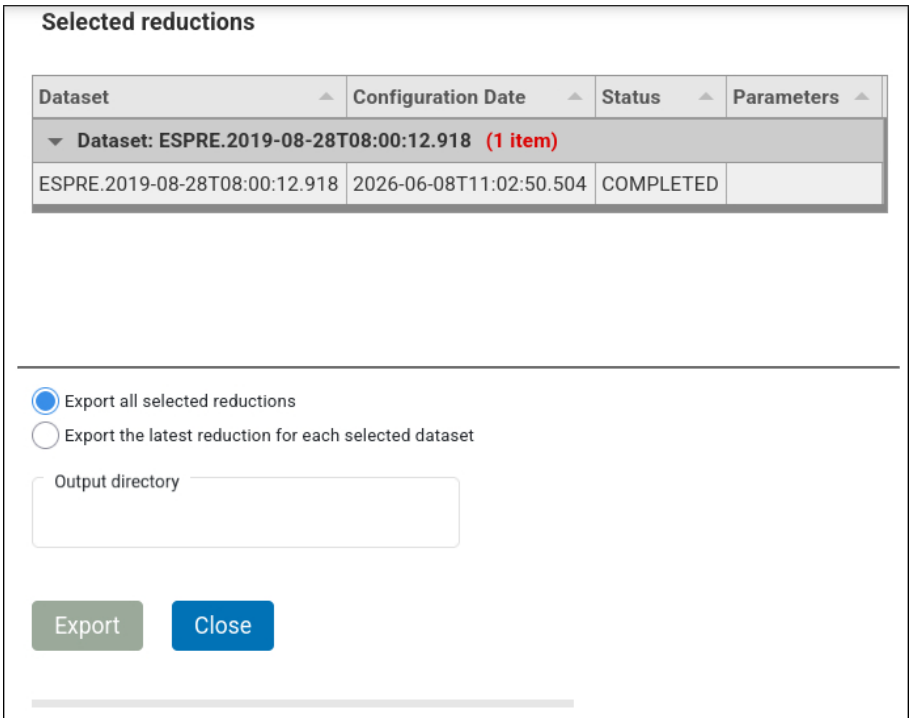

Figure 14: The EXPORT dialogue window, where the user can decide which reduced configuration to save and where.

- SPECTRUM_COMBINED followed by the target name
- SPECTRUM_ followed by the exposure identifier (header keyword 'arcfile'). They are located in a sub-directory of the combined spectrum.

3 The ESPRESSO data reduction flow.

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

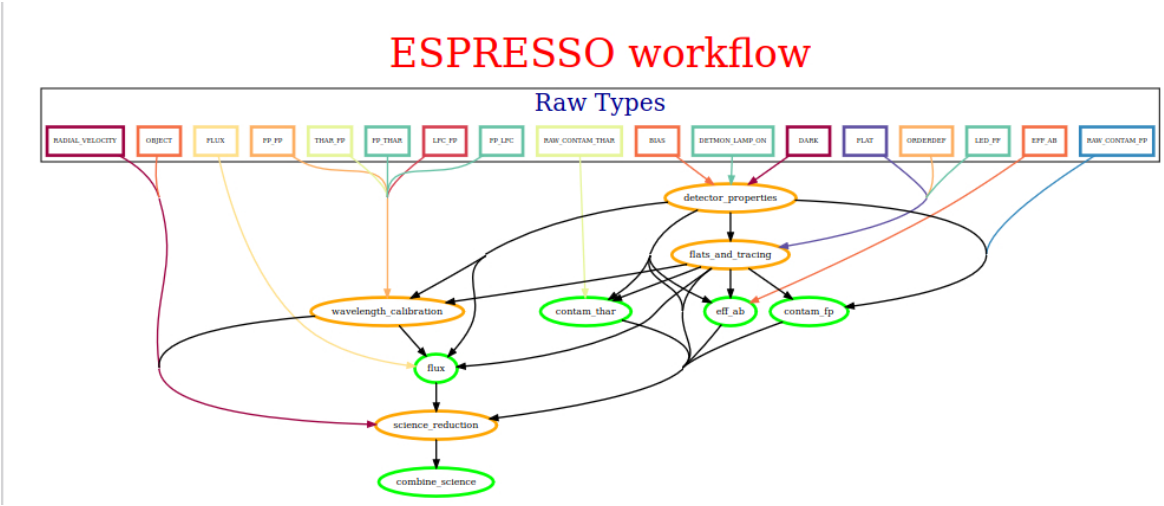

Figure 15: The data reduction cascade of the ESPRESSO workflow.

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 `espresso.espresso_wkf EDPS` workflow is designed to execute the tasks that deliver the final reduced data cube 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 the 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 standar 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 `espresso.espresso_wkf` 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 for more info on the configuration editor 4.2

The EDPS data reduction workflows process data in batches, called datasets, consisting of related science observations and the necessary calibrations. Therefore, the first steps in executing an EDPS data reduction workflow are to classify the input data and organize it into datasets. ESPRESSO follows the pattern adopted for many instruments designed for RV monitoring: each dataset includes only a single science observation (and all necessary calibrations for its processing) and each dataset is processed by the pipeline recipes separately, generating a product for each raw input file.

Next comes the processing of calibrations. Only those that are needed by the selected scientific exposures are processed.

Finally, the science data are processed and, as explained, a product is generated for each individual dataset, which is equivalent to a product for each individual raw science file. However, ESO provides a set of additional recipes grouped in ESO Tool Kit (ESOTK) that includes a recipe `esotk_spectrum1d_combine` designed to combine 1-dimensional spectra (in the ESO internal data product format; the ESPRESSO pipeline products are already in this format, so there is no need for conversion). The `espresso.espresso_wkf` workflow uses this recipe to generate a combined spectrum of multiple exposures. By default, exposures of the same template are selected for combination. To skip combination, deselect the **combine_science** task from the Reduction Targets field in item 3 (Select the reduction target) of the Raw Data tab.

The reduction steps of the `espresso.espresso_wkf` workflow are:

3.1 Detector properties

The processing begins with the usual detector-related manipulations (bias, dark, detector monitoring), marked with the rectangle in the top row and performed in a separate subworkflow, named Detector properties (see Figure 16).

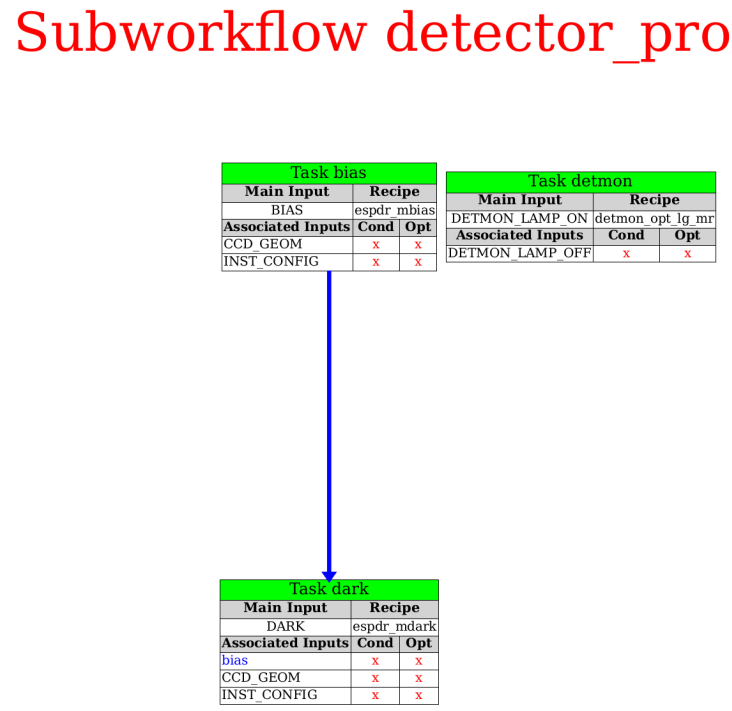

Figure 16: The list of tasks and their dependencies in the detector properties subworkflow.

It includes the following tasks **bias** (recipe `espd_mbias`, for the master bias and bad pixel mask) **detmon** (recipe `detmon_opt_lg_mr`, for the linearity correction if the detmon package is installed) and **dark** (recipe `espdr_dark`, for the master dark and the hot pixel mask).

Graphical reports for the products of tasks **bias** and **dark** can be inspected. Recipe parameters that control the stacking method or the sigma clipping can be used if needed, but typically default parameters provide the best results.

3.2 Flats and tracing

This subworkflow is dedicated to generate the flat fielding, the blaze function, and to trace and identify the various dispersion orders of the fibre spectra on the detector (Figure 17).

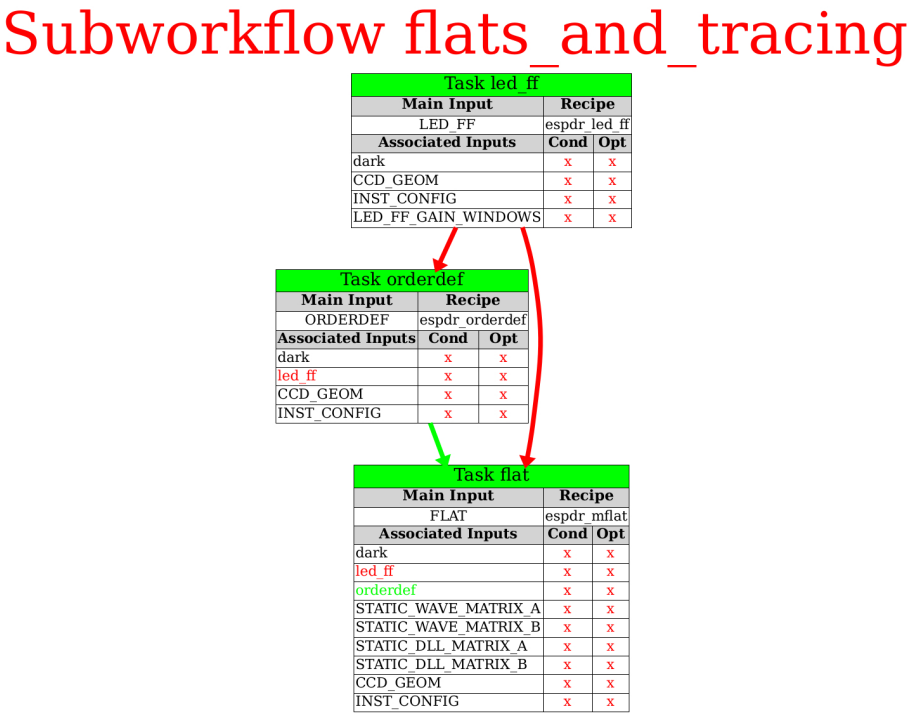

Figure 17: The list of tasks and their dependencies in the flats and tracing subworkflow.

It contains the tasks **led_ff** (recipe `espdr_led_ff`), **orderdef** (recipe `espdr_orderdef`), and task **flat** (recipe `espdr_mflat`).

The products from these and the previous recipes are used across the rest of the ESPRESSO workflow, as can be seen from the arrows that connect them with the following recipes and sub-workflows.

Graphical reports for the products of the tasks **orderdef** and **flat** can be inspected. Tipicallt, the default recipe parameters provide already the best results.

3.3 Wavelength calibration

This subworkflows executes all the steps and the tasks to compute the wavelength calibration of ESPRESSO data (Figure 18).

The wavelenght solution is computed wavelength calibration using various calibration sources, and illuminating both fibres with them, on a combination that depends on the observing strategy.

The task **wave_fp_fp** runs the recipe `espdr_wave_FP` to process calibrations that have both fibres illuminated by the the Fabry-Pérot etalon lamp.

The tasks **wave_fp_thar** and **wave_thar_fp** run the recipe **espd_wave_THAR**, to process calibrations that have one fibre illuminated by the Fabry-Pérot and the other illuminated by the Thorium-Argon lamp.

There is also an option of laser frequency comb-based wavelength calibration via the tasks **wave_fp_lfc**, **wave_lfc_fp** (recipe `espdr_wave_LFC`). The LFC is a preferable wavelength calibration method for programs aiming to obtain accurate radial velocities (RV), because ThAr lamps degrade with time, leading to larger RV scatter. However, the ThAr provides a first guess for deriving the LFC solution, which explains the sequence of tasks in the workflow.

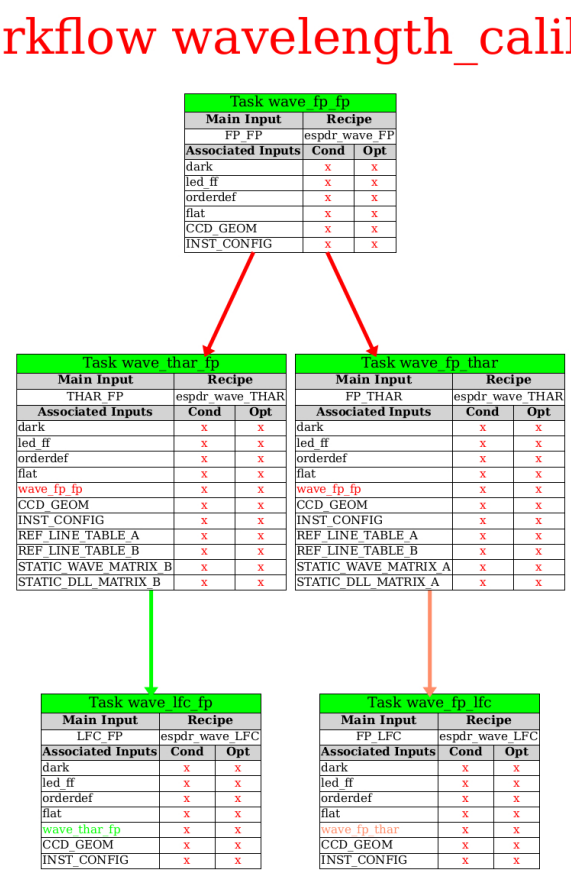

Figure 18: The list of tasks and their dependencies in the wavelength calibration subworkflow.

Graphical reports for the products of all the tasks can be inspected; typically the recipe default parameters already provide the best results.

3.4 comtam_thar, contam_fb, eff_ab

These tasks computes the fibre contamination when illuminated by the Thorium-Argon and Fabry-Perot lamps (recipe `espdr_cal_contam`), and the mutual contamination between the two fibres (recipe `espdr_cal_eff_ab`). Tipically, default values give the best results.

The user may change parameters controlling the overscan determination, the inter-order background computa-

tion, and the extraction. The user may change the parameters that control the overscan, the inter-order background determination, the extraction and the polynomial fit degree used to fit the efficiency

Occasionally, cosmic rays might cause a bad result. In this case, it is recommended to clean the raw contaminated frames via external tools before processing them with edps.

3.5 Flux

The task **flux** runs the recipe recipe **espdr_cal_flux**, and generates the response curve that converts the flux conversion into to ergs s^{-1} cm^{-2} $Å^{-1}$, based on a flux standard observation; it can also correct for the extinction.

Default recipe parameters generally provide the best results, but the user may change the parameters that control the overscan, the inter-order background determination, the extraction and the polynomial fit degree used to fit the efficiency.

3.6 Science Reduction

The subworkflow science reductions processes science data and radial velocity standard using the recipe `espdr_sci_red`
in the tasks **object** and **rv_stars**

The two tasks performs the same steps on different types of data, therefore the folloing applies to both of them.

3.6.1 Sky subtraction

For very low S/N regime, readout limited or close to it, it is recommended to use the option smooth for sky subtraction, thus reducing the readout noise contribution from the step of sky subtraction. This can be done by setting the recipe parameter `sky_corr_method` to `smoothed` in the configuration editor.

3.6.2 Background intra-order contamination

The recipe parameter `background_sw` removes the intra-order diffuse background contamination (it does not have to be mis-interpreted by sky background, which is regulated by `sky_sub_method` and active only for OBJECT,SKY observations). It is evaluated on boxes of size `sci_bkgr_grid_size_x` × `sci_bkgr_grid_size_y`
and then interpolated all over the frame. The diffuse background is proportional to the dispersed light inside the spectrograph (and thus, to a good approximation, to the total flux recorded in the detector). If the diffuse background is not measured correctly (as happens when is negligible), correcting for it will only add an additional source of noise to the flux. Fabry-Perot light contributes significantly to the inter-order background and as such exposures with FP on fiber B have the inter-order background correction flag by default at ON. In most cases, stellar light does not contribute significantly to inter-order background and as such exposures with SKY on fiber B have the inter-order background correction flag by default at OFF. If any exposure points to the SKY on fiber B and yet has a very large target flux in fiber A, the user should change the inter-order correction flag to ON.

3.6.3 Radial velocity

The computation of radial velocity is a key-feature in the `espresso` pipeline. One might decide to fine-tune the recipe parameters to improve the calculation or to run a dedicated esoreflex workflow and recipe to execute only this step: they are available as part of the ESPRESSO-DAS pipeline, which is capable of dealing also with ESPRESSO data. We refer the user to the ESPRESSO-DAS documentation for this option.

The recipe parameters that optimize the computation of radial velocity are: `rv_range` and `rv_step`. The general recommendation is to start with a large range `rv_range` and a course grid `rv_step` to locate the minimum. Then a fine-tuning of the range and grid are necessary to increase the precision of the measurement. However, M-dwarfs are particularly prone to bad convergence in the CCV computation if the RV range is large. It is not recommended to derive RVs using an RV range larger than 20 km/s for M dwarfs. When computing the radial velocity of the same target over a certain period of time, it is important to use the same template (mask_table_id) to remove systematic effects between measurements at different epochs.

The quality can be checked by inspecting the CCF product (see 3.6.5) and check whether a clear minimum is visible over the individual orders.

Figure 19 show an example of bad and good CCF computations.

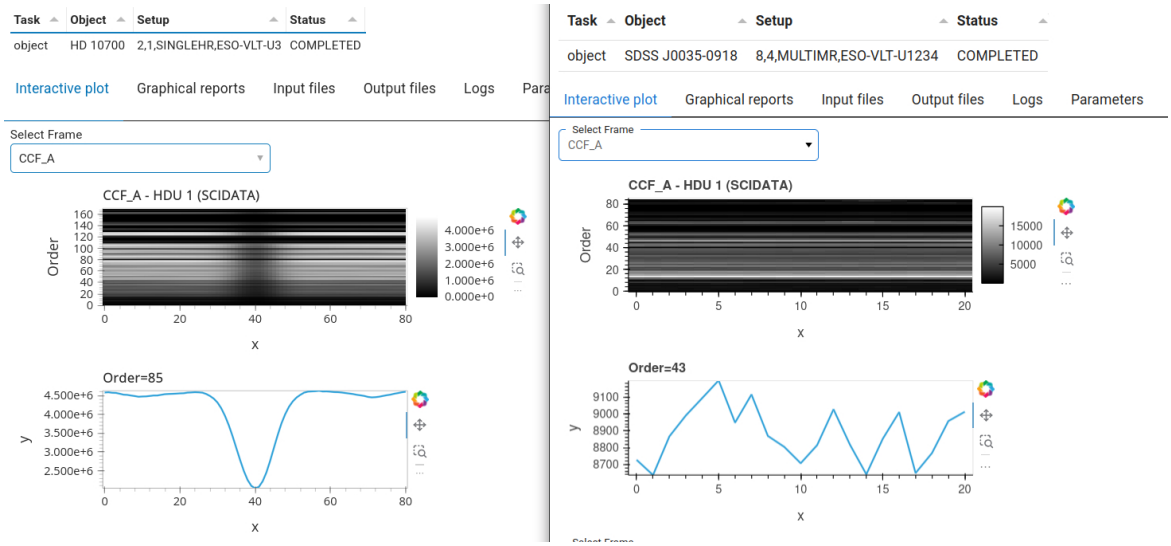

Figure 19: Examples of CCF. The left panel shows a CCF with a well defined minimum over the orders; the right panel shows a CCF with undefined minimum over the orders. In this latter case, the initial guess of the velocity was way off the analized range specified by `rv_range`.

3.6.4 Cosmic rays cleaning

The science recipe has two algorithms for cosmic ray cleaning. They user can decide to use none, one or both by setting the corresponding recipe parameter in the configuration editor. The parameters are:

- `ksigma_cosmic`. ksigma for removing cosmics on fiber A or SKY; Set it to -1 to deactivate it. Default: 3.5. The algorithm operates on the extracted spectrum and flags pixels that deviate more than a certain

factor.

- `cosmic_detection_sw`. Use the LA Cosmic detection (van Dokkum 2001, PASP 113, 1420). Set it to 0 to disable, 1 to enable. `cosmic_detection_sw` to 1 or 0, respectively. Pixels that are flagged are ignored during the extraction of the one-dimensional spectrum. The tab "LACOSMIC" in the interactive window allows the user to specify the recipe parameters that control the algorithm. By default, the workflow switches on/off the algorithm and loads the optimal recipe parameters for a given observing mode. We refer the user to the pipeline manual for more information on the recipe parameters. The user might want to inspect the products with category CRH_MAP and CCD_CORR_SCIENCE to assess the quality of the flagging process. These products are available by pressing the magnifying glass of the corresponding job.

The general recommendation is to switch on both algorithms, but pay attention whether very intense targets or very intense exposures of fibre B (in particular with high binning modes), do generate too many false positive detections. In this case, it is recommended either to switch the LA Cosmic algorithm off, or to increase the `f_lim` `sigma_lim` thresholds, regulated by the corresponding recipe paramters.

If a cosmic ray is only partially detected, one might dilate the cosmic-ray mask by setting recipe parameters `post-filter-x` and `post-filter-y` parameters to 1 or 2 (higher values are not recommended).

The user has the option to provide a mask with cosmic rays detection (category CRH_MAP). If provided, it will switch off the execution of the LA Cosmic algorithm. In order to be used by the workflow and the recipes, the user-provided mask has to:

- be present amon the inputs.
- have the category `CRH_MAP` (header keyword HIERARCH ESO PRO CATG).
- have the same format of `CRH_MAP` produced by switching on the LA Cosmic algorithm. The convention is: pixels with value of 1 are cosmic rays, other pixels have value 0 (integers).
- have the header keyword ARCFILE of the same value of the science frame it is supposed to be associated to.

It is therefore recommended to first run the reduction with the LA Cosmic algorithm turned on, with the option `extra_products_sw`, and then edit the CRH_MAP produced by the workflow according to the needs. It is also possible to run a third-party cosmic ray detection algorithm on CCD_CORR_SCIENCE and record the flagged pixel into CRH_MAP. Note that the inclusion of an user-provided CRH_MAP replaces the LA Cosmic algorithm in the science recipe, but the "k-sigma clipping" algorithm can still be executed.

3.6.5 Products

In general, the products of the science reduction subworkflow follow two formats:

- One dimensional spectra or S1D. They are FITS tables that consist of a header with no data and a regular FITS table stored in the first extension (and can be inspected with any tool that reads FITS tables, such

as topcat or fv). The combine the information of several orders. The column names in the table vary depending on the product type. Information about some columns that can be found in the S1D tables (e.g., in the S1D_FINAL_A files) is given in Table 2, and a plot of the S1D spectrum, generated with EDPS, is shown in Figure 20 and in the bottom panel of Figure 21.

- Two dimensional spectra, or S2D. They show in each row the spectrum of each order. It has multiple extensions, the first extension contain the spectra, other extensions contain the wavelength information (vacuum and air in separate extensions), errors, and quality flags. The wavelength runs horizontally across the order, increasing from left to right, but it is stored in one of the extensions, so normally the files are displayed in pixels. The bluest orders are located at the bottom, and they are shorter than the reddest orders which are located on the top – this is due to the specifics of the ESPRESSO optical design. These files are 2-dimensional images, and they can be inspected with any image display tool such as fv or ds9. An example of S2D quality plot is shown in Figure 21 (top panel).
- Other products, such as
  - CCF_A. Cross correlation function for fibre A with the object in S2D format. It contains the CCF for each order, with the corresponding errors and the data quality (in separate FITS extensions).
  - CCF_RESIDUALS_A Residuals from CCF computation of the RV for the spectrum – a FITS table with RVs, their errors and the residuals from the mean RV for each individual order of the spectrum.
  - DRIFT_MATRIX_B. Drift matrix in S2D format with a single extension, storing the relative shift of the wavelength calibration solution position of the spectrum in fibre A (with the science target) relative to fibre B (with the calibration source) between the time of (night) observation and the time when the calibration was acquired (during the day).
  - S2D_BLAZE_A/B Blaze functions on order-by-order basis for fibre A/B.

A full list of products and their description is given in the pipeline manual. It is worth showing here the columns present in the extracted spectra (Table 3.6.0).

3.7 combine science

This tasks combines the extracted science spectra observed within the same template using the recipe `esotk_spectrum1`
For studies that implies the study of spectral variation less than 1 hours (the typical OB lenght), it is recommended not to execute it.

Table 3.6.0: An example of the ESPRESSO S1D product spectra.

Column name	**Units**	**Description**
WAVE	[Å]	Wavelength vector, in vacuum reference system.
FLUX	[erg cm^{-2} s^{-1} Å^{-1}] or [e^{-}]	Flux of the final extracted spectrum in Fibre A, fully reduced till the end of the data reduction cascade. It is flux calibrated (if flux calibration was performed) and sky subtracted (if sky subtraction was performed).
ERR	[erg cm^{-2} s^{-1} Å^{-1}] or [e^{-}]	Error associated to FLUX.
QUAL		Quality flag associated to FLUX.
SNR		Signal to noise ratio, obtained from FLUX/ERR.
WAVE_AIR	[Å]	Wavelength vector, in the air reference system.
FLUX_EL	[e^{-}]	Extracted flux from fibre A, not sky subtracted nor flux calibrated.
ERR_EL	[e^{-}]	Error associated to FLUX_EL.
QUAL_EL		Quality flag associated to FLUX_EL.
FLUX_CAL	[erg cm^{-2} s^{-1} Å^{-1}]	Extracted flux from fibre A, flux calibrated but not sky subtracted.
ERR_CAL	[erg cm^{-2} s^{-1} Å^{-1}]	Error associated to FLUX_CAL.
QUAL_CAL		Quality flag associated to FLUX_CAL.
FLUX_CAL_SKYSUB	[erg cm^{-2} s^{-1} Å^{-1}]	Extracted flux from fibre A, flux calibrated and sky subtracted.
ERR_CAL_SKYSUB	[erg cm^{-2} s^{-1} Å^{-1}]	Error associated to FLUX_CAL_SKYSUB.
QUAL_CAL_SKYSUB		Quality flag associated to FLUX_CAL_SKYSUB.
FLUX_EL_SKYSUB	[e^{-}]	Extracted flux from fibre A, sky subtracted but not flux calibrated.
ERR_EL_SKYSUB	[e^{-}]	Error associated to FLUX_EL_SKYSUB.
QUAL_EL_SKYSUB		Quality flag associated to FLUX_EL_SKYSUB.

Figure 20: The quality plot (graphic report) associated to the tasks **object** and **rv_stars**, showing the extracted spectrum (S1D_FINAL_A) and zooms on selected regions. A smoothed version of the spectrum is also shown.

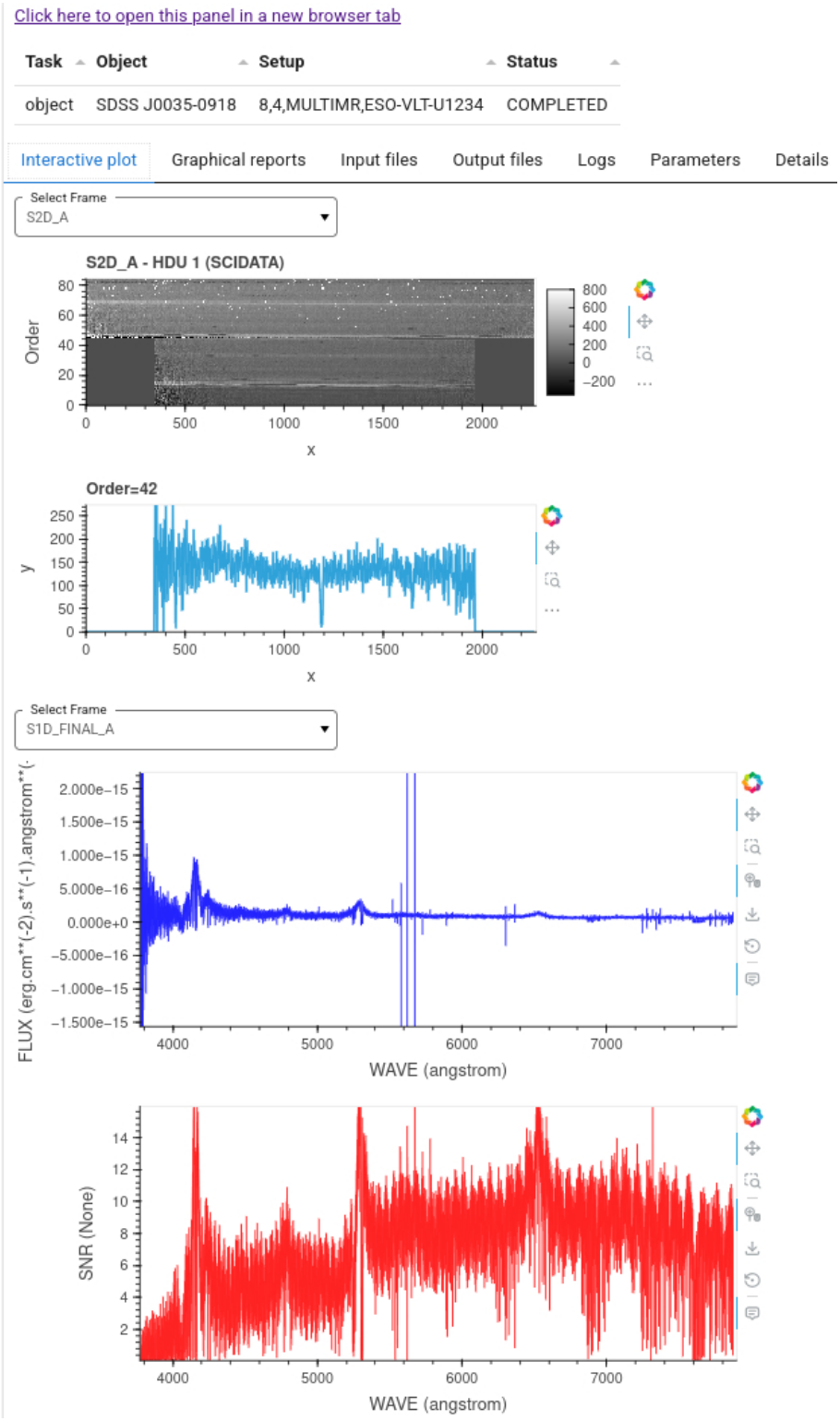

Figure 21: The quality plot (interactive plot) associated to the tasks **object** and **rv_stars**. Various products can be selected for inspection, in the exaple shown in this figure, the S2D_A and S1D_FINAL_A are shown in the upper and bottom panels, respectively.

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 og 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 22.

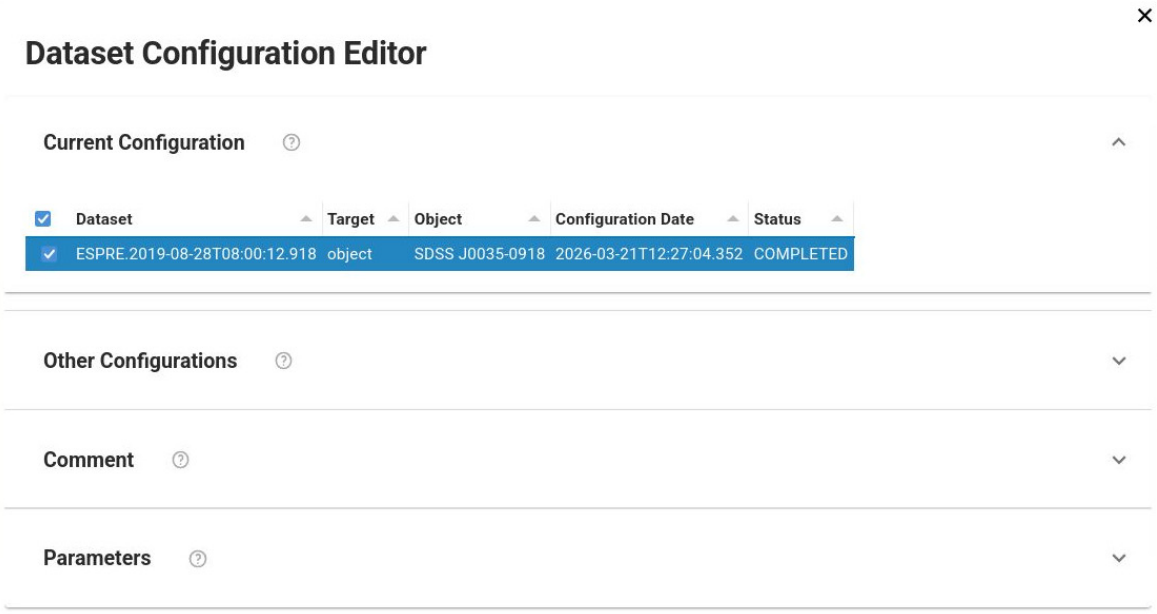

Figure 22: The Reduction Configuration editor. It contains 4 sections, that indicate the current configuration, list of other configurations to set, comments to insert, and the parameters to modify.

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 23).
- **Other configurations.** It allows to specify other configurations, to which the changes shall be copied to (Figure 24.
- **Comment** It allows to specify a comment to describe the configuration. It is possible to append or replace a comment (Figure 25). 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 26 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.

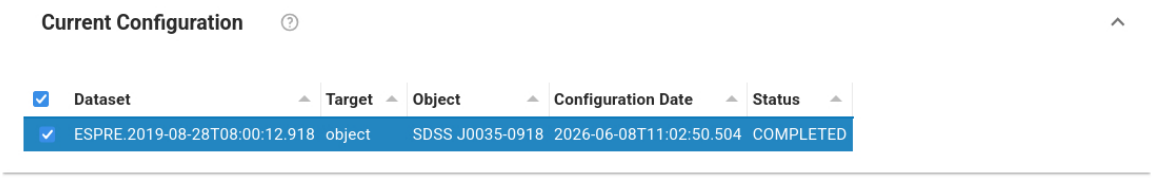

Figure 23: The first part of the Reduction Configuration Editor, that indicates the selected configuration.

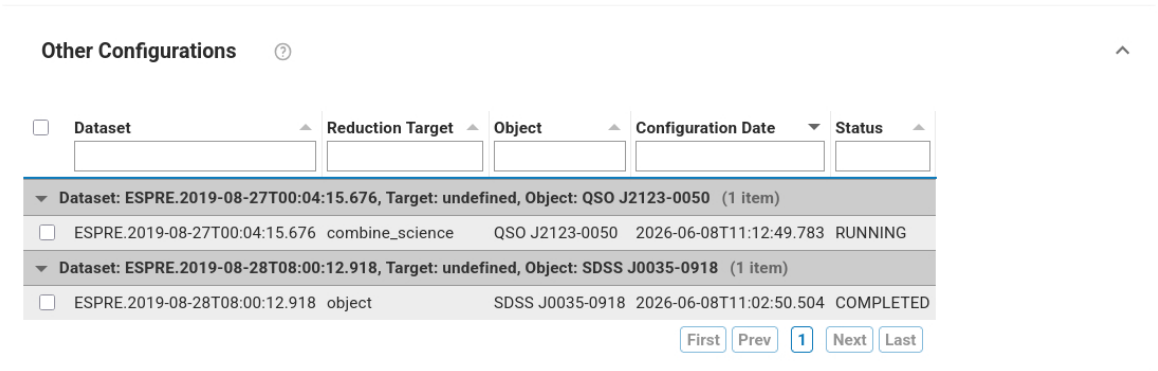

Figure 24: The second part of the Reduction Configuration Editor, that indicates other configurations for which we'd like to apply the changes.

4.3 WORKFLOW PARAMETERS

The current `espresso.espresso_wkf` has only one workflow parameter `is_idp` that is used only for the automatic reduction for the ESO science archive. If set to true it does not use Laser Frequency Combes

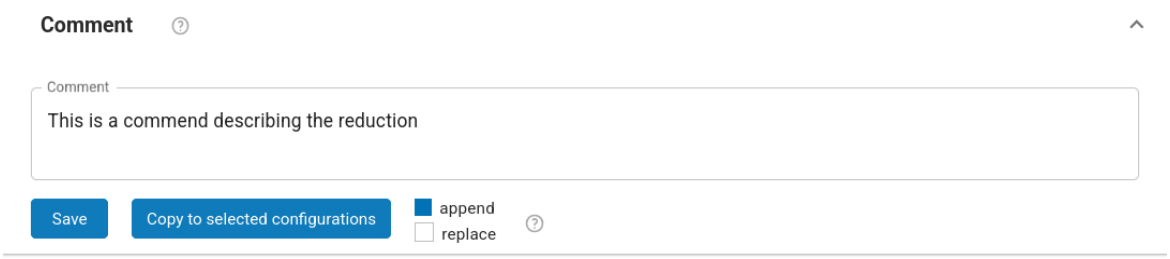

Figure 25: The third part of the Reduction Configuration Editor, that allows to specify a comment to the selected configurations.

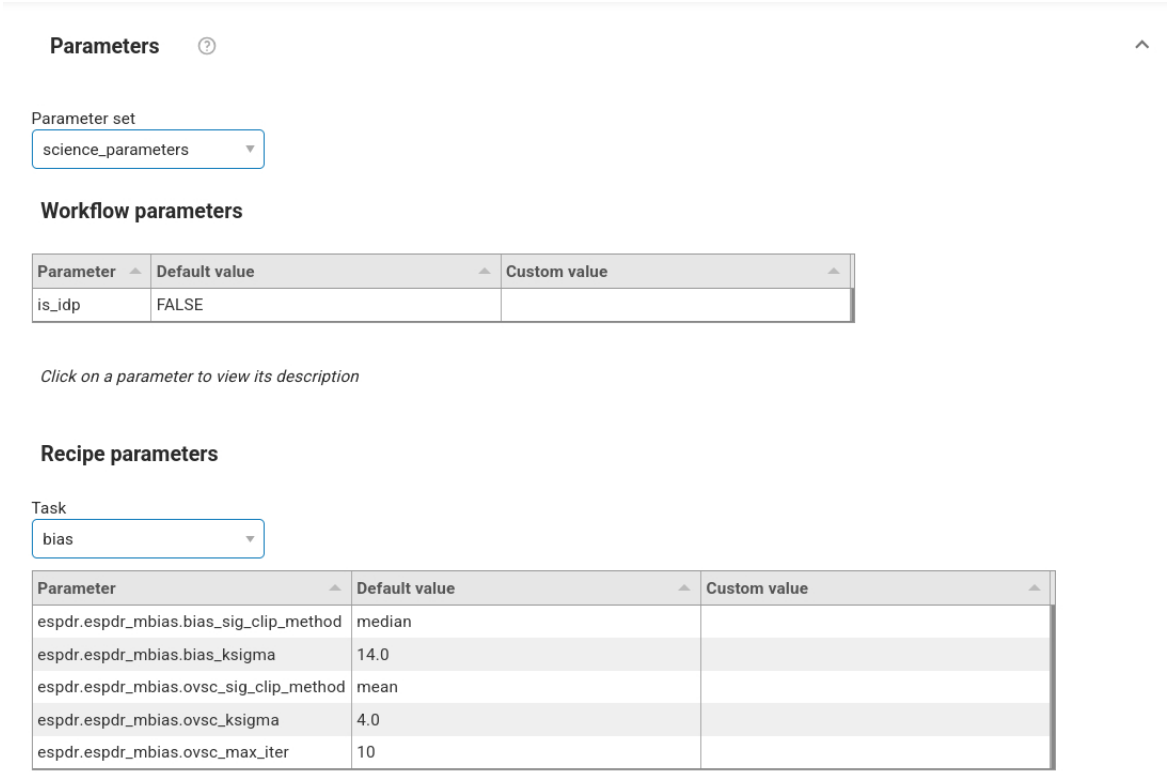

Figure 26: The fourth part of the Reduction Configuration Editor, that allows to specify the parameters sets and the recipe parameter per task. These settings can be applied to the "Selected Configuration" (Fig. 23) or to the "Other Configurations (Fig. 24).

calibrations (not supported yet in the ESO archive products), and it consider a flux standard mandatory (current is optional).

4.3.1 Combination of spectra

By default, the `espresso.espresso_wkf` combines the extracted spectra obtained within the same observing template. If this is not the desired behavior, deselect the task **combine_science** from the list of reduction

tasks in the Raw Data tab, before pressing the button to create the datasets.

5 List of workflow tasks

This is the list of all the tasks and associated recipes in the `espresso.espresso_wkf` workflow. Only some of them are needed for scientific reduction, they are indicated by the flag "yes" (triggered by default) or "optional" (triggered only if requested by a workflow parameter). Other tasks are not used for scientific reduction (they are indicated by the flag "no"), they are mainly used for instrument monitoring and they can be executed only by specifying them as target. Note that, when a task is specified as target, all the tasks that generate the calibrations needed for it are automatically executed.

TASK	**RECIPE**	**Used in science reduction**	**Notes**
bias	espdr_mbias	Y	Generates a masterbias and residuals.
combine_science	esotk_spectrum1d_combine	Y	Combines extracted spectra.
contam_fp	espdr_cal_contam	Y	Computes the contamination from Fabry Perot exposures.
contam_thar	espdr_cal_contam	Y	Computes the contamination from Thorium-Argon exposures.
dark	espdr_mbias	Y	Generates a masterdark and hot pixel mask.
detmon	detmon_opt_lg_mr	N	Computes linearity correction.
eff_ab	espdr_cal_eff_ab	Y	Computes relative efficiency between fibres.
flat	espdr_mflat	Y	Computes the profiles and flat field correction of orders.
flux	espdr_cal_flux	Y	Computes flux calibration.
led_ff	espdr_led_ff	Y	Computes bad pixel mask.
object	espdr_sci_red	Y	Reduces science observations and computes radial velocity.
orderdef	espdr_orderdef	Y	Traces the various orders on the detector.
rv_stars	espdr_sci_red	N	Reduces science observations of radial velocity standards.
wave_fp_fp	espdr_wave_FP	Y	Creates an extracted spectrum table and wavelength solution matrix from Fabry-Perrot calibrations.
wave_fp_lfc	espdr_wave_LFC	Y	Creates an extracted spectrum table and wavelength solution matrix from LFC calibrations.
wave_fp_thar	espdr_wave_THAR	Y	Creates an extracted spectrum table and wavelength solution matrix from Thorium-Argon lamp.
wave_lfc_fp	espdr_wave_LFC	Y	Creates an extracted spectrum table and wavelength solution matrix from LFC calibrations.
wave_thar_fp	espdr_wave_THAR	Y	Creates an extracted spectrum table and wavelength solution matrix from Thorium-Argon lamp.

Table 5.0.0: ESPRESSO pipeline tasks overview

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).