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

KMOS EDPS-GUI tutorial

1 rev. 20260516

Issue 1

Date May 16 2026

Prepared:	Burkhard Wolff	May 16 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	15/05/2026	all	First public release for 4.5.3

This page was intentionally left blank

Contents

1 Introduction

1.1 Scope

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

Note: this tutorial refers to:

- KMOS instrument pipeline named `kmos`, version 4.5.3.
- KMOS workflow: `kmos.kmos_wkf`
- `EDPS` version 1.5.7.
- `edps-gui` version 1.0.

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 KMOS demo data. We assume that the `EDPS`, `edps-gui`, the KMOS 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).

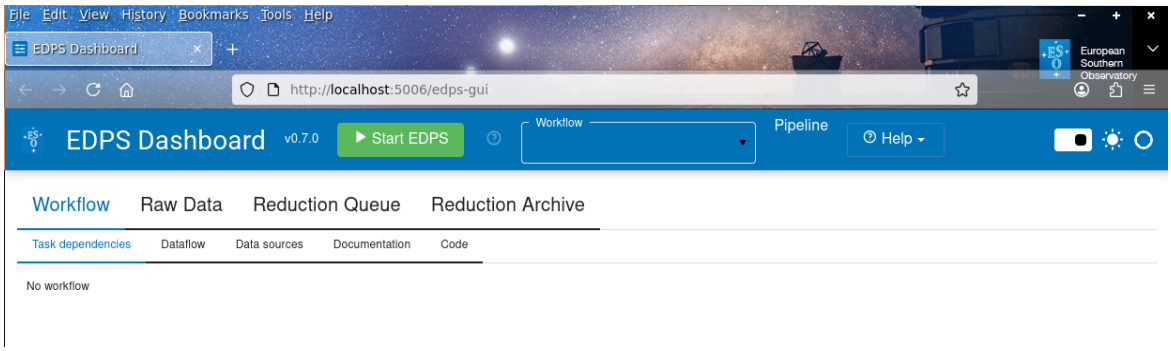

Figure 1: The empty `edps-gui` Dashboard; the underlying `EDPS` engine has not yet been started and no workflow has been loaded.

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

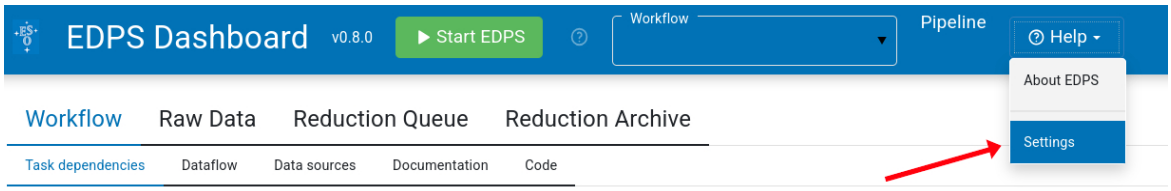

Figure 2: The “Help” –> “Settings” menu.

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

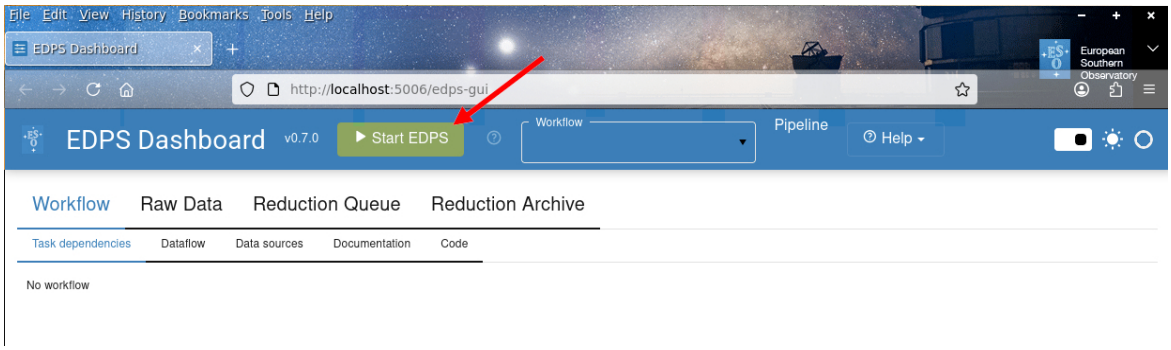

Figure 3: The “Start EDPS” button.

2.2 Selecting the input data

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

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

3. (Optional). Select the reduction target, configure the workflow parameter and specify the association preferences. These steps are optional. For more information see Section 4.

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

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

   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.

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

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.

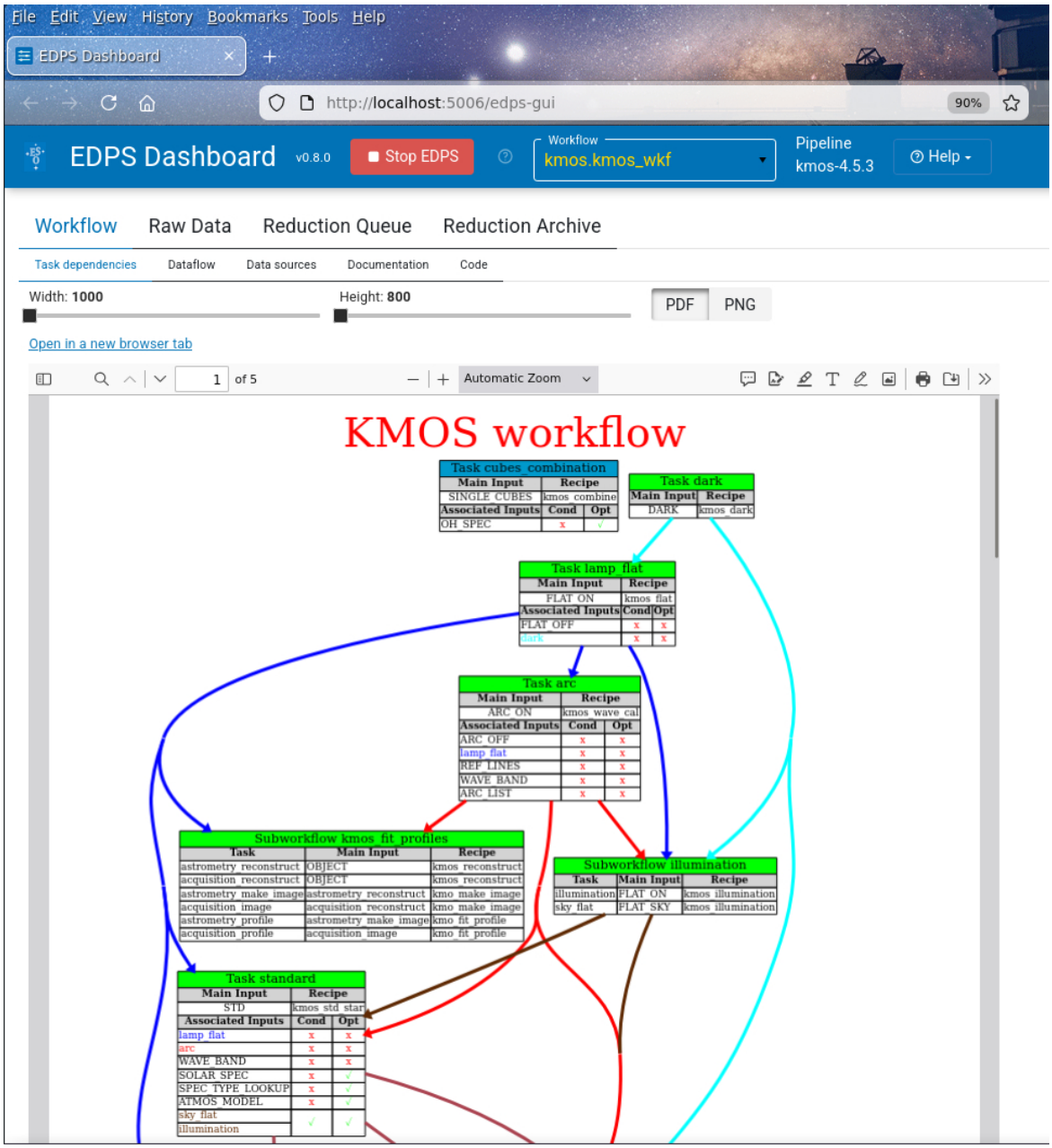

Figure 4: The `edps-gui` with the KMOS workflow loaded.

2.3.1 Quality plots

Almost all processing tasks can display the input raw frames and the products in the so called "quality plots", which can be inspected from the ‘Reduction Queue‘ window. Those associated for the main product can be inspected by pressing the magnifying glass symbol at the right side of each dataset. To inspect those associated to each individual job (if created),

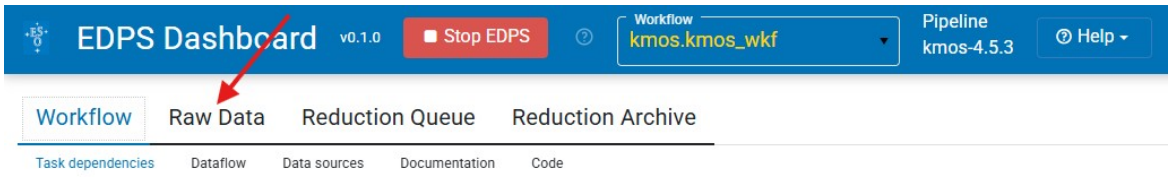

Figure 5: How to select RAW data Tab.

Figure 6: How to select input data.

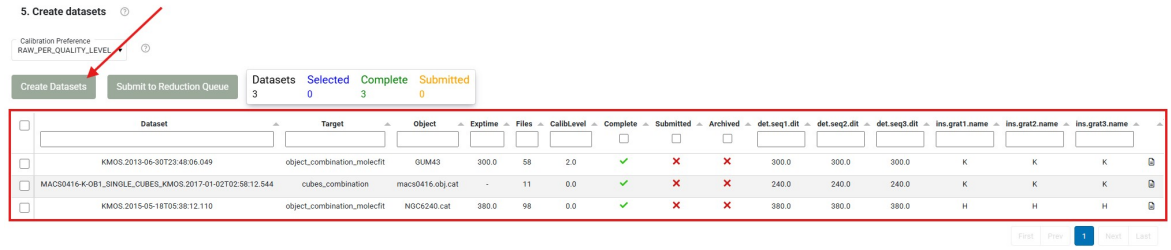

Figure 7: How to inspect the input data directory to create datasets.

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

- 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. Only plots for completed jobs can be inspected.

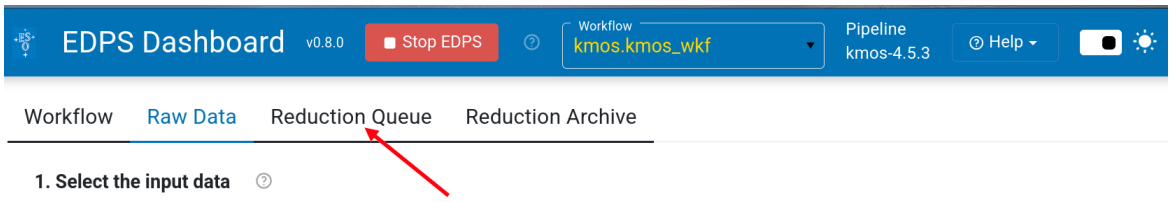

Figure 9: How to select Reduction Queue tab.

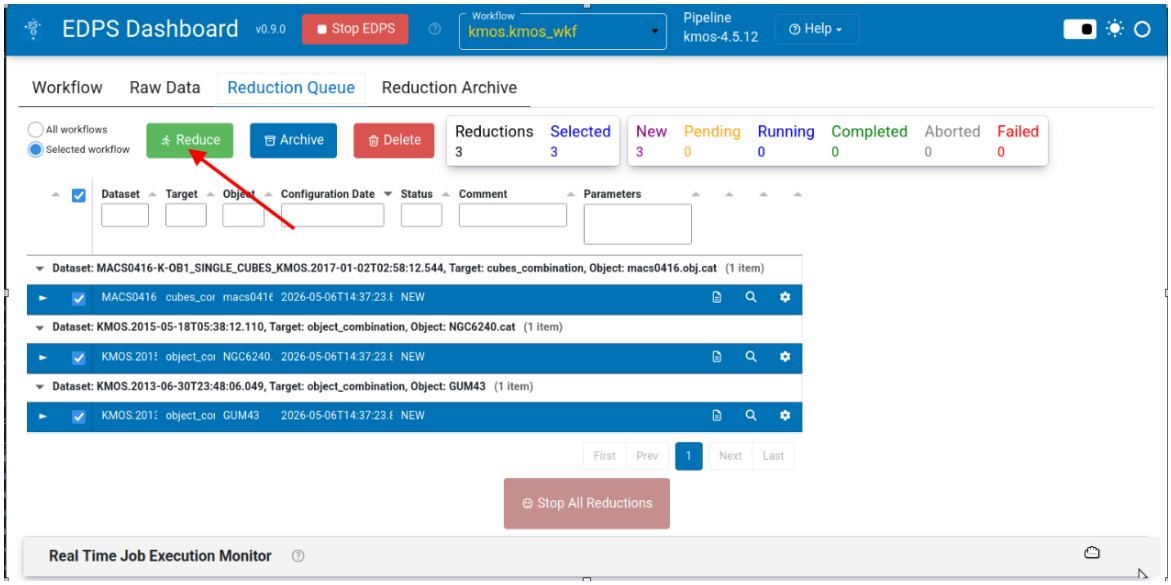

Figure 10: Reduce.

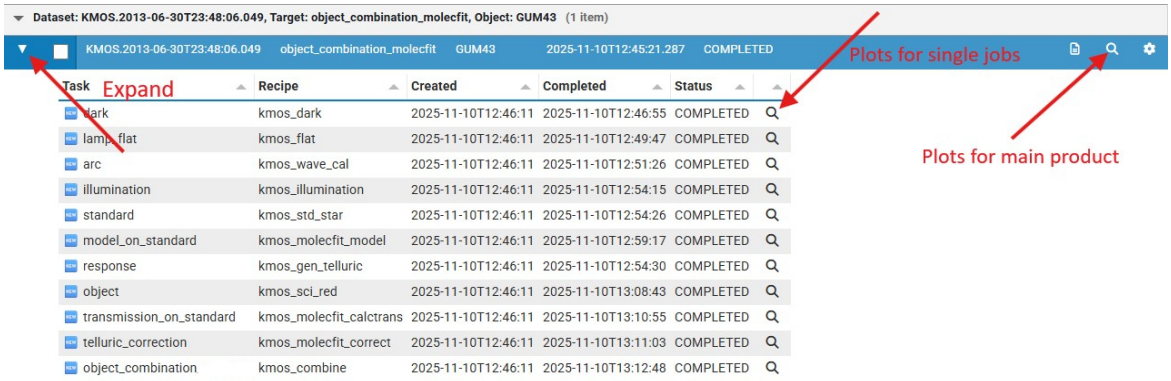

Figure 11: How to look for quality plots from the Reduction Queue tab.

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.

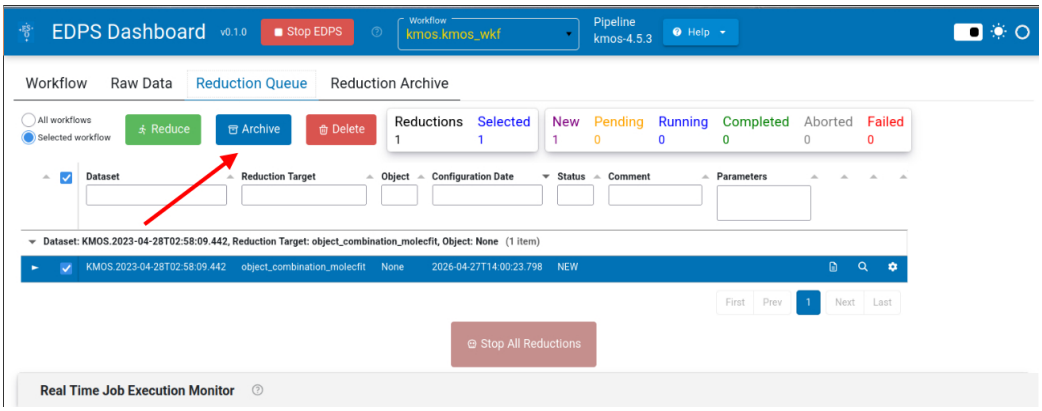

Figure 12: How to archive a completed reduction from the Reduction Queue tab.

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.

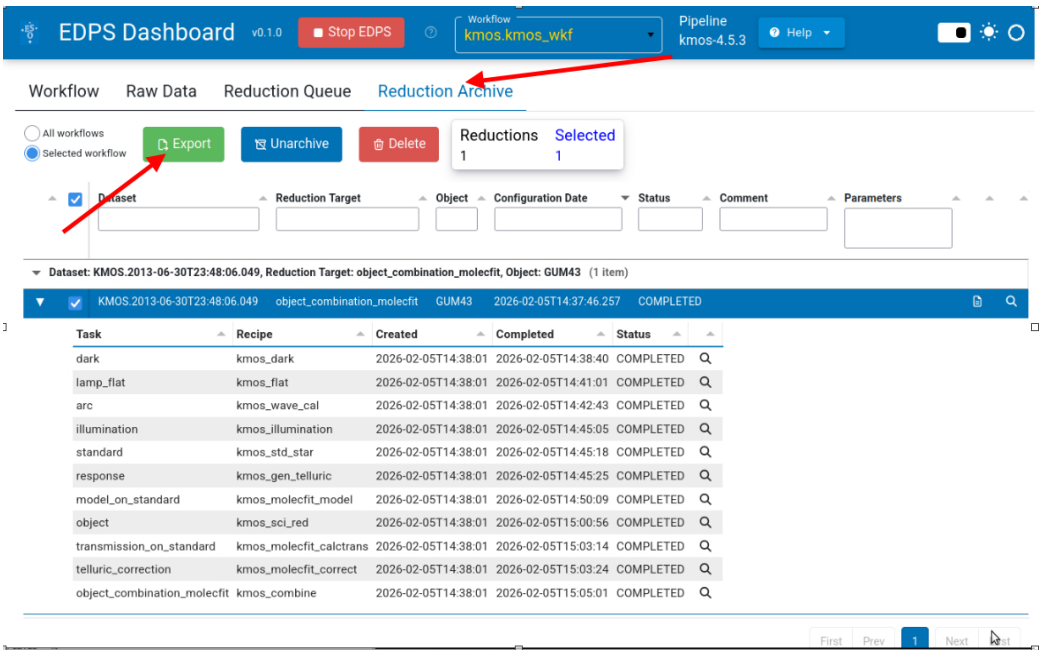

Figure 13: The reduction archive tab. This table contains all the different configurations of datasets that are declared "finished" and removed from the Reduction Queue. From this page, the user can export the most important files into a desired local directory.

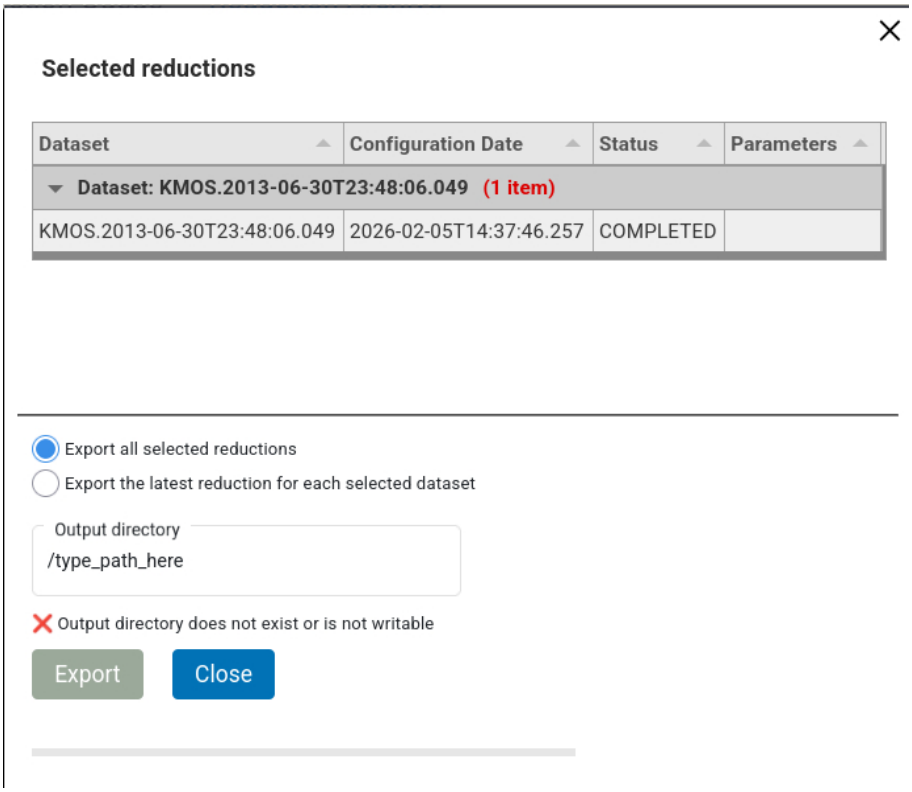

Figure 14: The EXPORT dialogue window, where the user can decide which reduced configuration to save and where.

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:

- One combined data cube for each object in the dataset, coming in two different formats as ‘COMBINED_CUBE‘ and ‘IDP_COMBINED_CUBE‘. Both products have the same data content. The latter file has the needed metadata information for ingestion into the ESO archive.
- A field-of-view image ‘COMBINED_IMAGE‘ and an exposure mask image ‘EXP_MASK‘ for each object.

Individual reconstructed cubes that contributed to each combined cube (telluric corrected or not) are saved in sub-directories.

3 The KMOS data reduction flow.

The overall data flow of the KMOS 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 `kmos.kmos_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 `kmos.kmos_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

3.1 Process dark exposure

This step is carried out by the task **dark** using the pipeline recipe `kmos_dark`. It produces a MASTER_DARK and a BADPIXEL_DARK frame. Per default, only the bad pixel map from this step is used in the reduction chain.

3.2 Create Master Flat

Task: **lamp_flat**. Recipe: `kmos_flat`.

This step uses exposures with the flat-field lamp to create a MASTER_FLAT, files that describe the geometry of the IFU slitlets on the detector (XCAL, YCAL, and SLIT_EDGE), and a BADPIXEL_FLAT frame.

3.3 Wavelength Calibration

Task: **arc**. Recipe: `kmos_wave_cal`.

This step uses exposures with the Argon and Neon arc lamps to create the wavelength calibration LCAL. A useful product for verifying the wavelength calibration is DET_IMG_WAVE which contains the wavelength-calibrated images of the input frames.

3.4 Illumination Correction

Task: **illumination**. Recipe: `kmos_illumination`.

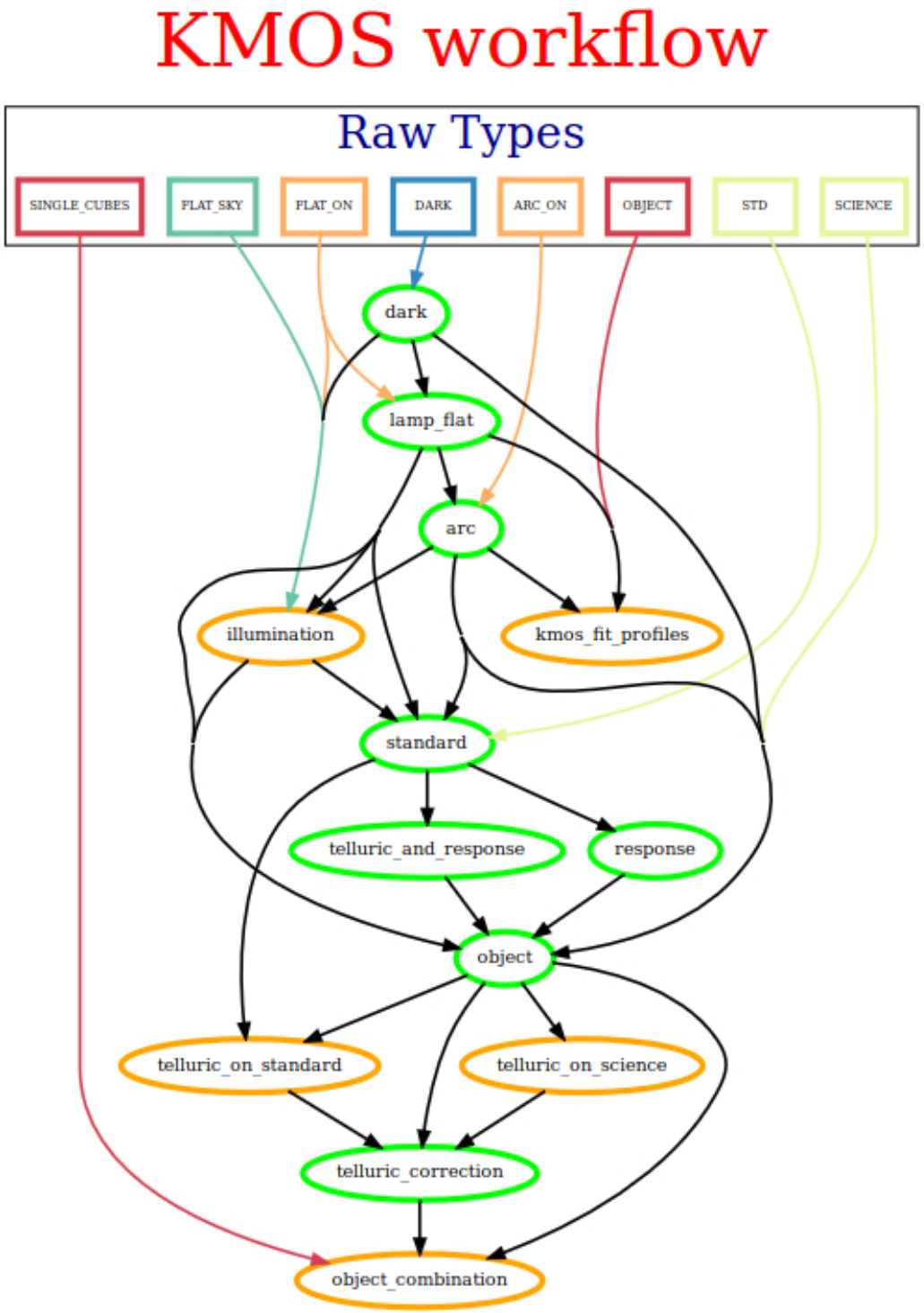

Figure 15: The data reduction cascade of the KMOS workflow.

This step can be run on lamp-flat or sky-flat exposures. It creates for each IFU an image (product: ILLUM_CORR) for correcting the sensitivity variation within the IFU. It is recommended to execute this step on lamp flats (which is the default workflow setup).

This step can be customized by setting the workflow parameter 'use_sky_flats' to 'false' (default, recommended) or 'true'.

3.5 Standard star

Task: **standard**. Recipe: `kmos_std_star`.

This step processes a telluric standard exposure and extracts a spectrum for each IFU that was used. Products:

- STAR_SPEC: extracted spectrum
- TELLURIC: normalized telluric spectrum
- STD_IMAGE: reconstructed IFU field-of-view image
- STD_MASK: mask for extracting the standard star.

The further usage of the products is controlled via the workflow parameter 'molecfit' which can have the values:

- 'standard': Atmospheric features are modeled with the molecfit algorithm using the telluric standard as reference. The results are used to construct the full atmospheric transmission to correct science exposures. A static response curve is used to correct the relative instrumental efficiency with wavelength. The zeropoint computed on the observed telluric standard is used for absolute spectrophotometric calibration.
- 'science': Atmospheric features are modeled with the molecfit algorithm using the scientific exposure itself as reference. To be used only if the science exposure has a bright continuum that allows a proper fit to the atmospheric features. A static response curve is used to correct the relative instrumental efficiency with wavelength. The zeropoint computed on the observed telluric standard is used for absolute spectrophotometric calibration.
- 'false': The observations of the telluric standard stars are compared with a model spectrum for that star. This generates a combined correction that includes response curve and telluric features, which is then used to correct scientific exposures.

3.6 Response

This step involves either the task **response** or the task **telluric_and_response**. Both use the pipeline recipe `kmos_gen_telluric`.

If the workflow parameter 'molecfit' is set to 'false', then the task **telluric_and_response** is used. The telluric spectrum as calculated from the **standard** task and the response are passed to the science reduction. For this other values of the workflow parameter, only the response is used.

3.7 Science reduction

Task: **object**. Recipe: `kmos_sci_red`.

In this step, the basic data reduction of the science data is executed. The data from the IFUs are extracted, calibrated, and reconstructed into data cubes. If the workflow parameter 'molecfit' is set to 'false', then the products are flux-calibrated and corrected for telluric absorption. Otherwise, the products are only flux-calibrated. The telluric correction is then executed in the following workflow steps.

Main products:

- SCI_RECONSTRUCTED: one file for each exposure that contains IFUs on objects (no file for pure sky exposures). Each file has 24 extensions with data cubes.
- SCI_RECONSTRUCTED_COLL: files with 24 extensions containing the reconstructed field-of-view images for each IFU.

3.8 Telluric model

This step either involves the tasks **model_on_standard** and **transmission_on_standard** or the tasks **model_on_science** and **transmission_on_science**. The tasks are encapsulated in the sub workflows telluric_on_standard and telluric_on science, respectively. The pipeline recipes `kmos_molecfit_model`, `kmos_molecfit_calctrans` are used.

If the workflow parameter 'molecfit' is set to 'standard', then the atmospheric model is fit to the extracted spectra from the **reponse** task. With the parameter set to 'science', the fit is executed on the science data themselves. In both cases, the actual transmission curve is calculated using the model and the information from the science data.

3.9 Telluric correction

Task: **telluric_correction**. Recipe: `kmos_molecfit_correct`.

In this step, the science data are corrected for telluric absorption using the transmission from the previous step.

3.10 Object combination

Task: **object_combination**. Recipe: `kmos_combine`.

The different exposures for each object are combined. The final products are:

- One combined data cube for each object in the dataset, coming in two different formats as 'COMBINED_CUBE' and 'IDP_COMBINED_CUBE'. Both products have the same data content. The latter file has the needed metadata information for ingestion into the ESO archive.
- A field-of-view image 'COMBINED_IMAGE' and an exposure mask image 'EXP_MASK' for each object.

3.11 Cubes combination

Task: **cubes_combination**. Recipe: `kmos_combine`.

If data have already been processed with the previous reduction then the resulting cubes can be combined for each object. This is useful for combining observations of the same targets which have been measured using different Observation Blocks.

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 16.

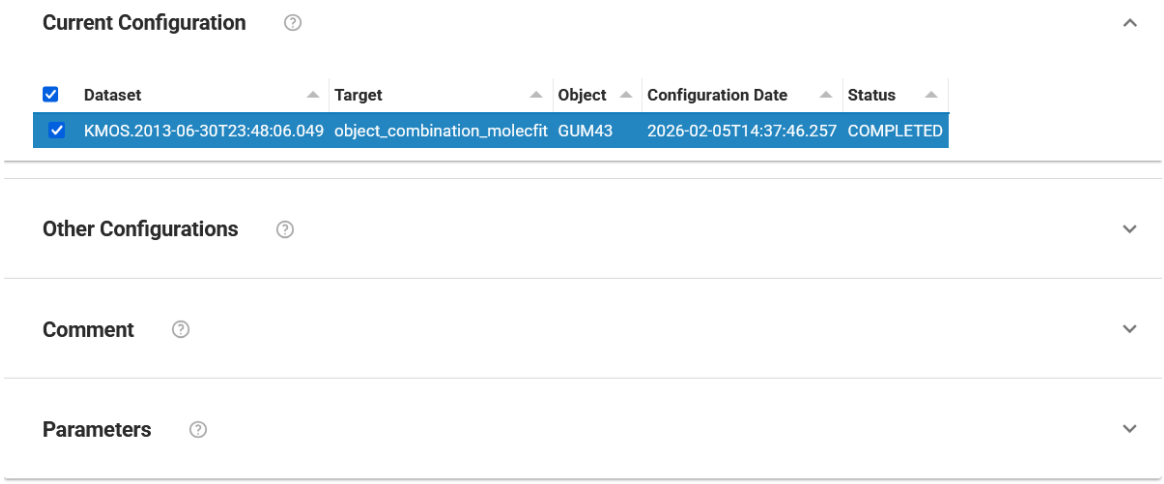

Figure 16: The Reduction Configuration editor. It contains 4 sections, that indicate the current configuration, list of other configurations to set, comments to insert, and the parameters to modify.

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.

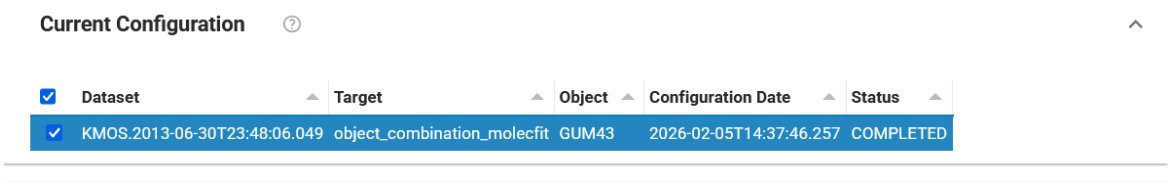

Figure 17: The first part of the Reduction Configuration Editor, that indicates the selected configuration.

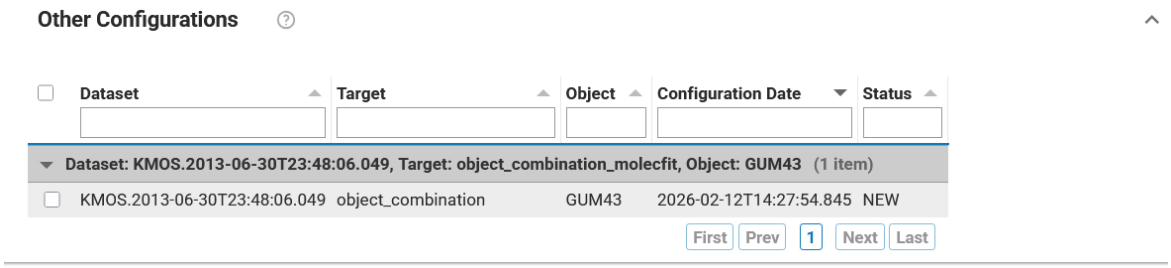

Figure 18: The second part of the Reduction Configuration Editor, that indicates other configurations for which we'd like to apply the changes.

4.3 Wavelength calibration

The interactive window for the wavelength calibration step (task **arc**) shows the reconstructed arc frames for each of the three detectors at one of the six rotator position angles. Other rotator angles (either 0, 60, 120, 180, 240, or 300 degrees) can be chosen with the 'Select Angle' menu. The output file with the reconstructed arc frames has 18 extensions (3 extensions per detector, times 6 angles). Exposures at different angles are taken in order to account for instrument flexures. When applying the wavelength correction to a dataset, the KMOS pipeline automatically selects the calibration with the closest angle.

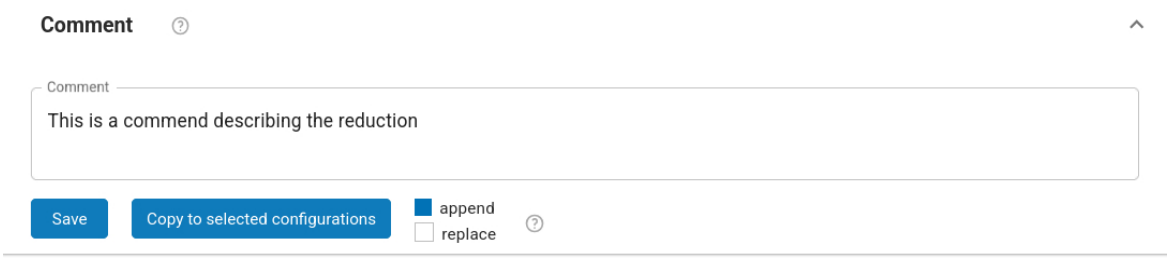

Figure 19: The third part of the Reduction Configuration Editor, that allows to specify a comment to the selected configurations.

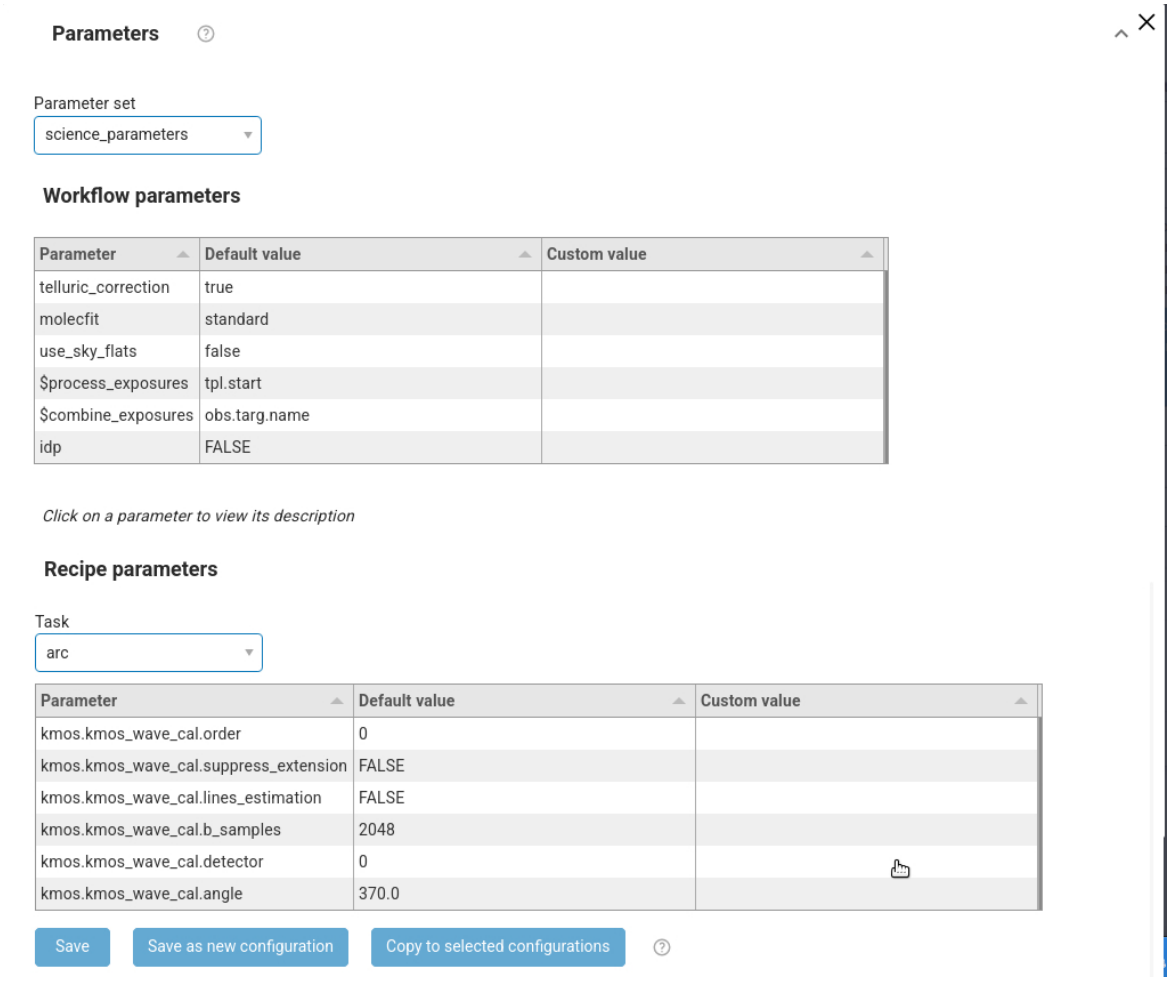

Figure 20: The fourth part of the Reduction Configuration Editor, that allows to specify the parameters sets and the recipe parameter per task. These settings can be applied to the "Selected Configuration" (Fig. 17) or to the "Other Configurations (Fig. 18).

The KMOS wavelength calibration uses Argon and Neon arc lines. There is normally no need to fine tune the wavelength calibration result. Since the Argon lines are less numerous and less bright than the Neon ones, the

fit to the line positions is dominated by the Neon lines and the average offset of the Argon lines is larger than the ones of the Neon lines. This can be checked with the QC ARC AR POS MEAN and QC ARC NE POS MEAN header keywords in the product file extensions. Their values are less than the uncertainties. The effect could be mitigated by increasing the recipe parameter 'order' to 7. The parameter determines the order of the fit polynomial. With its default value of 0, the applied order is 6 for the H, K, and YJ grisms, 4 for the IZ grism, and 5 for the HK grism.

4.4 Illumination correction

Task: **illumination**. Recipe: **kmos_illumination**.

This step can be run on lamp-flat or sky-flat exposures. It creates for each IFU an image (product **ILLUM_CORR**) for correcting the sensitivity variation within the IFU. It is recommended to execute this step on lamp flats (which is the default workflow setup).

This step can be customized by setting the workflow parameter 'use_sky_flats' to 'false' (default, recommended) or 'true'.

4.5 Science reduction

4.5.1 Science reduction: grouping of input data

There are two different ways of grouping the input data for the task **object** (recipe **kmos_sci_red**). The workflow parameter '$process_exposures' contains the header keyword that is used for the grouping. Options are:

- 'tpl.start': data are grouped per observing block (template) execution (default).
- 'obs.container.id': data are grouped per container ID. To be used, for example, if target and sky are observed in different OBs but with the same OB container.

Note : this parameter does not affect the final combination of the data cubes with the tasks **object_combination** or **cubes_combination**. It only groups the input data for the task **object** which are then processed with the same set of calibrations (Sections 3.10, 3.11).

4.5.2 Science reduction: level correction

Science data can contain extra counts due to the electronics and other sources. These counts differ from one detector read-out channel to another and can change the intensity across the detector. The KMOS pipeline has two main strategies for removing this extra level ("level correction").

The first strategy (overscan) evaluates the extra counts on pre- and over-scan regions; it evaluates the correction for each of the 32 read-out channels independently and for even and odd pixels.

The second strategy (intra slices), uses the non illuminated portion of the detector located in between the various slices and IFUs to evaluate the extra light and remove it. This strategy is to be used only for very faint sources, otherwise the cross-talk effect contaminates the intra-slices region and the pipeline over corrects the data.

These strategies can be selected by setting the ‘lcmethod‘ parameter in the recipe **kmos_sci_red** (task **object**).

The values for ‘lcmethod‘ are:

- OSCAN It applies the overscan method. This is the default value.
- SLICES_MEAN. It applies the intra-slices method. The input file is divided into a grid of 32×16 windows, each of them 64 × 128 pixels wide. The mean of the non illuminated portions (i.e., intra-slices pixels) of the detector within each window is subtracted from the data. Bad pixels are masked. Intra-slice pixels and bad pixels are identified with the calibrations BADPIXEL_DARK and LCAL produced by the workflow.
- SLICES_MEDIAN. Same as SLICES_MEAN but the median is computed instead of the average.
- NONE. No correction is done.

In general, the default strategy OSCAN is recommended. Intra-slice correction can improve a bit the overall background subtraction (but not the emission sky lines) and it is recommended only for very faint sources. Bright sources generate cross-talk between slices. The intra-slice area is then contaminated by this source of light in the target exposures (which will be over-corrected) and not in the sky exposures. This will result in an over subtraction of the sky continuum for the slices that are more affected by the cross-talk.

4.5.3 Science reduction: optimizing sky removal with recipe parameters

The **kmos_sci_red** recipe triggered by the **object** task automatically assigns a sky IFU to an object IFU. The adopted criteria is to use the same IFU as the object and which is closest in time. To override the automatic assignment, please refer to the next section.

Once the object / sky pair is identified, the recipe proceeds to remove the sky from the corresponding object IFU. To suppress sky subtraction, set ‘no_subtract‘ to true and ‘sky_tweak‘ to false. This setup will create also reconstructed sky cubes.

There are several steps involved in the sky subtraction. First, the object cube is reconstructed and aligned to a reference spectrum with OH emission lines. This includes a 2nd polynomial order fit to compensate non-optimal wavelength calibration. The alignment is triggered if the dataset contains a OH_LINES reference file, which is the default case and the recommended strategy. To remove this step, deselect the OH_LINES reference file from the dataset.

Then the user has the option to re-scale the intensities of the sky emission lines measured on the sky cube to match those observed in the science cube. This option exploits the sky tweaking algorithm described in Davies et al. 2007, MNRAS, 375, 1099, that defines groups of lines with the same scaling factor. This option is triggered by setting ‘sky_tweak‘ to true (default). In some (rare) cases, switching off the ‘skytweak‘ method gives better results than the default configuration. Note that you cannot trigger this option and set ‘no_subtract‘ to true.

The sky lines scaling factors computed via the sky tweaking algorithm can be not correct if there is still a residual mismatch in wavelength between the object spectrum and the science spectrum. This scenario can be identified by the presence of P-Cygni profiles in the residual sky lines in the reconstructed datacubes. In some cases, it is possible to correct for this wavelength mismatch by applying an additional re-alignment of the skylines measured in the sky and object cubes. This can be done by setting the parameter ‘stretch‘ to true. The algorithm

detects bright emission lines from 1D sky spectra extracted from the object and the sky datacubes, identifies the same lines in both spectra, computes the difference in position, and computes a polynomial correction to minimize these differences. This correcting polynomial is then applied to the sky cube only in order to align it better to the object cube.

Note that the alignment of the sky cube to the OH_LINES reference spectrum is still done before applying the stretching algorithm, unless 'skip_sky_oh_align' is set to true. Skipping the sky alignment reduces the number of interpolations to reconstruct the sky datacube. However, in case of large offsets, it might still be useful to have this initial alignment before applying the stretching algorithm. Note that 'skip_sky_oh_align' = true has an effect only if the stretching algorithm is activated.

The outcome of the stretching algorithm needs to be evaluated with care. It is suggested to use high-order polynomials (e.g. between 4 and 14) by setting the 'stretch_degree' parameter accordingly (the default degree is 8). The use of high-order polynomials helps in correcting for small wavelength mismatches in most of the spectral range. It can, however, introduce spurious effects at the edges of the wavelength range.

4.5.4 Science reduction: object/sky association for sky subtraction

The recipe parameter 'obj_sky_table' available in the task 'object' allows to specify the path to an ASCII file that associates every object exposure with its corresponding sky, overriding the automatic association done by the recipe itself (which is applied with the default parameter value 'none'). The suggested procedure for this is:

1. Run the workflow on only one data set.
2. Locate the file obj_sky_table.txt produced by the **object** task (kmos_sci_red recipe). It is located in the same directory as the output products for the object task. This can be checked by clicking on the "Output files" tab in the "Quality reports" window.
3. Copy it to a safe place and rename it in a way that makes it easy to associate it with the current dataset.
4. Change it according to the needs (an example is provided below). Each time the file is edited, it should be saved with a new name, otherwise a new reduction will not be triggered.
5. For each dataset/configuration the file applies to, open the configuration editor and enter the full path in the 'obj_sky_table' field (task 'object'). Save as new configuration.
6. Save as new configuration (or update existing) and start the reduction.

The ASCII file that associates every object exposure with its corresponding sky looks like the following (the caption lines are not shown here, for sake of clarity).

```
# [caption lines skipped]
Object/sky associations of frames tagged as: SCIENCE
index: filename:
# 0: /scratch/data/edps/kmos/2013-06-30/KMOS.2013-06-30T23:48:06.049.fits
# 1: /scratch/data/edps/kmos/2013-06-30/KMOS.2013-06-30T23:53:23.571.fits
# 2: /scratch/data/edps/kmos/2013-06-30/KMOS.2013-06-30T23:59:09.586.fits
-------------------------------------------------------------------------------
IFU          1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18  19  20 21 22 23 24
```

```
                ------------------------------------------------------------------------
frame #  0:  /scratch/data/edps/kmos/2013-06-30/KMOS.2013-06-30T23:48:06.049.fits
      type:  O  O  O  O  O  O  O  O  O  O  O  O  O  .  .  O  .  O  O   O   O  O  O  O  O
  sky in #:  1  1  1  1  1  1  1  1  1  1  1  1  1  .  .  1  .  1  1   1   1  1  1  1  1
frame # 2:   /scratch/data/edps/kmos/2013-06-30/KMOS.2013-06-30T23:59:09.586.fits
      type:  O  O  O  O  O  O  O  O  O  O  O  O  O  .  .  O  .  O  O   O   O  O  O  O  O
  sky in #:  1  1  1  1  1  1  1  1  1  1  1  1  1  .  .  1  .  1  1  1/20 1  1  1  1  1
------------------------------------------------------------------------------------------
```

The IFUs pointing to Object and Sky are identified with the letters “O” and “S”, respectively. Inactive arms are identified with a dot “.”. For each frame and IFU containing an object, the corresponding frame and IFU containing the sky is displayed in the line below. If the sky location is described with a single number, e.g. N (N is an integer), then the sky to be used is on the exposure with index N, on the same IFU as the object. If the sky location is described in the format N/M (N and M integers, with $1 < M < 24$), then the sky to be used is located in the exposure with index N and IFU M.

In the example shown above, in the object frame KMOS.2013-06-30T23:48:06.049.fits (that has index 0) each n-th IFU has the corresponding sky cube in the same n-th IFU of the frame with index 1 (KMOS.2013-06-30T23:53:23.571.fits). The same is true for the object frame KMOS.2013-06-30T23:59:09.586.fits (that has index 2), except for IFU 19. The corresponding sky has to be found in the frame with index 1, at ifu 20.

4.6 Object combination

Task: **object_combination**. Recipe: `kmos_combine`.

The different exposures for each object are combined. The behavior of this task is controlled by the workflow parameter `$combine_exposures`. Its values can be:

- `obs.targ.name`: one job for each target name is created (default).
- `tpl.start`: one job for each OB template execution is created.

If there are several observations in the data set then one job per target name is created with the first value. If an object was observed with different OB executions then all observations of this object are combined in the final data cube. With the second value, the combination is only done per observation, i.e., OB/template execution.

The behavior of the task can be controlled further with the recipe parameters ‘ifus‘, ‘name‘, ‘filename‘ (task **object_combination**).

- `ifus`: the indices of the IFU arms to combine. E.g.: ’1;2;5’. Default: ’UseNames’.
- `name`: the name of the object to combine (as in the header keyword OCS ARM NAME). Default: ’All’.
- `filename`: the path to the file with the shift vectors for the combination. Only applicable if the recipe parameter ‘method‘ is set to ’user’.

With the recipe parameter ‘method‘, the shifts that are applied in the object combination can be controlled. Its possible values are:

- `none`: no shifting, combined directly
- `header`: shifts are applied according to WCS (default)
- `center`: centering algorithm is used
- `user`: shifts are read from file.

4.7 Combination of pre-reduced cubes

The task **object_combination** works on data cube that share the same target name (OBS TARGET NAME header keyword). In order to create combined data cubes from observations with different target names (but for the same objects), the task **cubes_combination** can be used.

The task **cubes_combination** requires products of type SINGLE_CUBES as input that have been created with previous executions of the task **telluric_correction**. Each file contains the data cube for a single object. The task allows to combine the different data cubes for the same object.

The suggested procedure is:

1. Copy the SINGLE_CUBES files that shall be combined to an (empty) directory.
2. Select this directory as input in the "Raw data" section of the GUI and create a dataset for them. This data set should have the target **cubes_combination**.
3. Start the reduction of this data set in the "Reduction Queue" section.

For each target in the input data set, a product of category COMBINED_CUBE, COMBINED_IMAGE, IDP_COMBINED_ and EXP_MASK, respectively, is then created. The same workflow and recipe parameters as described in Section 3.10 for the task **object_combination** can be applied.

4.8 Telluric correction

The KMOS workflow allows to correct the observations for atmospheric transition. This is controlled by the workflow parameters ‘telluric_correction‘ and ‘molecfit‘.

The parameter ‘telluric_correction‘ can have the values:

- ’true’: telluric correction is performed, using the strategy as defined by the parameter ‘molecfit‘ (default).
- ’false’: telluric correction is not applied.

The workflow allows to select between three strategies which are set by the workflow parameter ‘molecfit‘:

- ’standard’: Atmospheric features are modeled with the molecfit algorithm using the telluric standard as reference. The results are used to construct the full atmospheric transmission to correct science exposures. A static response curve is used to correct the relative instrumental efficiency with wavelength. The zeropoint computed on the observed telluric standard is used for absolute spectrophotometric calibration.

- 'science': Atmospheric features are modeled with the molecfit algorithm using the scientific exposure itself as reference. To be used only if the science exposure has a bright continuum that allows a proper fit to the atmospheric features. A static response curve is used to correct the relative instrumental efficiency with wavelength. The zeropoint computed on the observed telluric standard is used for absolute spectrophotometric calibration.
- 'false': The observations of the telluric standard stars are compared with a model spectrum for that star. This generates a combined correction that includes response curve and telluric features which is then used to correct scientific exposures.

For detailed information on Molecfit, we refer to its user manual which is available here.

The use of molecfit provides in general the best results, as it constructs a model of the atmospheric transmission, and does not transfer to the science issues that can be included in the extraction of the 1D standard star spectrum, such as low S/N, cosmic-rays, and recipe failure in the absorption line fitting. However, it is rather slow, and the user might want to optimize the atmospheric model parameters to improve the execution speed and the fit performance.

Another advantage of using molecfit is that it accounts for the dependency of the instrumental spectral resolution from wavelength, IFU and grating, and rotator angle. This is advisable, for example, if a bright target was not observed with the same IFU and rotator angle as the telluric standard star.

The benefits of using molecfit are in general visible for high S/N targets (S/N >= 20). If the target has low S/N then it could be tried to switch the usage of molecfit off (by setting the workflow parameter to 'false'). This could also be a strategy if the standard star and the target have been observed with the same IFU. In this way, the determination of the response curve is based on the data from that night and does not rely on an average static calibration for the response.

The strategy of running molecfit directly on science has the advantage that the atmospheric model is obtained under the same atmospheric conditions (water vapor and column densities of molecules) as the science spectra. This strategy is recommended if there are science data with high signal-to-noise and few intrinsic features. These data can be used to determine the correction for all other science frames in the dataset.

Go to top

4.8.1 Using a telluric model from a standard star observation

The tasks for determining an atmospheric model from a standard star observation are encapsulated in the sub workflow **telluric_on_standard**. It uses the tasks **model_on_standard** (pipeline recipe `kmos_molecfit_model`) and **transmission_on_standard** (recipe `kmos_molecfit_calctrans`).

Go to top

4.8.2 Using a telluric model from a science observation

The tasks for determining an atmospheric model from a science observation are encapsulated in the sub workflow **telluric_on_science**. It uses the tasks **extract_spectra**, **select_reference**, **model_on_science** (pipeline

recipe `kmos_molecfit_model`) and **transmission_on_science** (recipe `kmos_molecfit_calctrans`).

4.8.3 Improving the telluric model

There are several recipe parameters for **model_on_standard** or **model_on_science** which can be used to improve the fit of the model.

Limiting the fit to certain IFUs. 'process_ifus' accepts a list of comma-separated integer numbers. The processing will then only be executed on those IFUs. Using only one IFU can be useful to increase the processing speed when trying different parameters. With the default value of '-1', all IFUs that have data are processed.

Wavelength ranges. The fit is performed on a subset of wavelength ranges and not to the entire spectrum; this is found to be more efficient and less time-consuming. The advice is to select few wavelength ranges that include the expected molecules that are observable in the spectrum. The recipe parameter to be used is 'wave_range'. For example, the entry '0.815,0.830,0.972,0.986' will perform the fit in the two ranges $0.815 < \lambda[\mu m] < 0.830$ and $0.972 < \lambda[\mu m] < 0.986$. The quotation marks are mandatory, spaces are not allowed.

The default value "-1" uses the ranges (values in μm):

- IZ: '0.815,0.830,0.894,0.899,0.914,0.919,0.929,0.940,0.972,0.986'
- YJ: '1.106,1.116,1.075,1.083,1.131,1.137,1.139,1.149,1.155,1.166,1.177,1.189,1.201,1.209,1.263,1.276,1.294,1.30
- H: '1.482,1.491,1.500,1.512,1.559,1.566,1.598,1.605,1.575,1.583,1.622,1.629,1.646,1.671,1.699,1.711,1.721,1.727
- K: '1.975,1.987,1.993,2.010,2.041,2.060,2.269,2.291,2.308,2.335,2.360,2.379,2.416,2.440,2.445,2.475'
- HK: '1.575,1.584,1.594,1.606,1.646,1.671,1.756,1.771,1.781,1.811,1.945,1.969,1.975,1.987,1.993,2.030,2.043,2.08

It is good advice to limit the number of wavelength regions to 4 or 5. The default ranges are defined to provide a good fit for all the possible cases at the cost of high execution time, but in general, they are redundant.

Molecules. Only the molecules that appear in the wavelength region of the grism should be fitted. The recipe parameters 'relcol', 'fit_molec', and 'list_molec' should then be defined accordingly.

The defaults of 'list_molec' (enabled with parameter value "-1") are:

- IZ: 'H2O'
- YJ: 'H2O,CO2,CH4,O2'
- H: 'H2O,CO2,CO,CH4'
- K: 'H2O,CO2,CH4'
- HK: 'H2O,CO2,CH4'

Fit of continuum shape and wavelength solution. It is advisable to fit both the continuum shape and the wavelength solution.

Parameters:

- ‘fit_cont‘: Flag to enable/disable the polynomial fit of the continuum. Default: true.
- ‘cont_n‘: Degree of polynomial continuum fit. Default: 1.
- ‘fit_wlc‘: Flag to enable/disable the refinement of the wavelength solution. Default: true.
- ‘wlc_n‘: Polynomial degree of the refined wavelength solution. Default: 2.
- ‘wlc_const‘: Initial constant term for wavelength adjustment. Default: 0.

Instrumental spectral resolution (kernel). The fit to the observations accounts for the instrumental spectral resolution.

These are available in the `kernel_<GRATING>.fits` files distributed with the pipeline.

Parameters:

- ‘use_input_kernel‘: If true (default), the provided input kernel is used.
- ‘fit_res_box‘: Fit resolution by Boxcar LSF. Default: false.
- ‘relres_box‘: Initial value for FWHM of Boxcar relative to slit width.
- ‘fit_res_gauss‘: Fit resolution by Gaussian. Default: true.
- ‘res_gauss‘: Initial value for FWHM of the Gaussian in pixels.
- ‘fit_res_lorentz‘: Fit resolution by Lorentz. Default: false.
- ‘res_lorentz‘: Initial value for FWHM of the Lorentz in pixels.
- ‘kernmode‘: Voigt profile approximation or independent Gauss/Lorentz fit.
- ‘kernfac‘: Size of Gaussian/Lorentz/Voigt kernel in FWHM.
- ‘varkern‘: Does the kernel size increase linearly with wavelength?

Fit precision. The fit precision is regulated by ‘ftol‘ and ‘xtol‘.

Parameters:

- ‘ftol‘: Relative chi-square convergence criterion. Default: 0.01.
- ‘xtol‘: Relative parameter convergence criterion. Default: 0.001.

4.8.4 Computation of atmospheric transmission: running calctrans

Once the parameters of the atmosphere are computed, the workflow combines them with the reconstructed cube to determine the full telluric correction.

This is executed by the tasks **transmission_on_standard** or **transmission_on_science** (recipe `kmos_molecfit_calct`

The process accounts for the airmass difference between the reference spectrum that was used to model the atmosphere and the science observations to be corrected.

For example, setting the recipe parameter 'IFU_1' to 3 will use the solution determined by the molecfit model obtained on IFU 3 to calibrate the scientific data on IFU 1.

A value of -1 triggers the following automatic matching:

1. If IFU Y in ATMOS_PARM and BEST_FIT_PARM contains data, then 'IFU_Y' = Y.
2. Otherwise, the first IFU X with valid data among those that belong to the same detector as IFU Y will be used.
3. If there are no available IFU for that detector, the first IFU Y among the input data that contains data will be used.

5 List of workflow tasks

This is the list of all the tasks and associated recipes in the KMOS 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
acquisition_image	kmo_make_image	no	Creates white-light images for acquisition data
acquisition_profile	kmo_fit_profile	no	Fits profiles to white-light images for acquisition
acquisition_reconstruct	kmos_reconstruct	no	Creates data cubes from acquisition data
arc	kmos_wave_cal	yes	Computes the wavelength calibration
astrometry_make_image	kmo_make_image	no	Creates white-light images for astrometry
astrometry_profile	kmo_fit_profile	no	Fits profiles to white-light images for astrometry
astrometry_reconstruct	kmos_reconstruct	no	Creates data cubes from astrometry observations
cubes_combination	kmos_combine	optional	Combines already reduced data cubes
dark	kmos_dark	yes	Processes dark exposures
extract_spectra	kmos_extract_spec	optional	Extracts spectra from science data as input for molecfit
illumination	kmos_illumination	optional	Creates the IFU illumination correction from lamp flats
lamp_flat	kmos_flat	yes	Processes lamp flat-field exposures
model_on_science	kmos_molecfit_model	optional	Fits an atmospheric model to science data
model_on_standard	kmos_molecfit_model	optional	Fits an atmospheric model to a telluric standard
object	kmos_sci_red	yes	Extracts science data and creates data cubes
object_combination	kmos_combine	optional	Combines data cubes per object
response	kmos_gen_telluric	optional	Creates the response from a standard star if molecfit is used
select_reference	none	optional	Selects the data for the atmospheric model fit if fit is on science data
sky_flat	kmos_illumination	optional	Creates the IFU illumination correction from sky flats
standard	kmos_std_star	yes	Processes telluric/flux standard stars
telluric_and_response	kmos_gen_telluric	optional	Creates response and telluric correction if molecfit is not used
telluric_correction	kmos_molecfit_correct	optional	Applies the telluric correction if molecfit is used
transmission_on_science	kmos_molecfit_calctrans	optional	Computes the telluric correction if calculated from science data
transmission_on_standard	kmos_molecfit_calctrans	optional	Computes the telluric correction if calculated from standard star

Table 5.0.0: KMOS 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).