EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VERY LARGE TELESCOPE

Reflex MUSE Tutorial

VLT-MAN-ESO-19540-6195

Issue 0.2

Date 2014-July-14

Prepared:	L. Coccato, R. Palsa	2014-July-14	
	Name	Date	Signature
Approved:	W. Freudling		
	Name	Date	Signature
Released:	M. Sterzik		
	Name	Date	Signature

This page was intentionally left blank

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
0.2	16/07/2014	All	Beta release

This page was intentionally left blank

Contents

1 Introduction And Scope **7**

2 Software Installation **8**

3 System requirements **9**

3.1 Hardware 9

3.2 JVM Memory set-up 9

3.3 Execution on machines with less than 64 GB of memory 9

4 Demo Data **10**

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

5.1 Setting the data reduction strategy 13

6 Tips and tricks **18**

6.1 Optimizing the combination of different exposures 18

6.2 Reduction of single files: multiple offset sky exposures within one OB 18

6.3 Combining data from different OBs 19

6.4 Execution of the workflow on computers with limited Memory 19

7 About The Reflex Canvas **20**

7.1 Saving And Loading Workflows 20

7.2 Buttons 20

7.3 Workflow States 20

8 The MUSE Workflow **21**

8.1 Workflow Canvas Parameters 21

8.2 Workflow Actors 22

8.2.1 Simple Actors 22

8.2.2 Lazy Mode 22

9 Test Cases **24**

10 Frequently Asked Questions **25**

1 Introduction And Scope

`Reflex` is the ESO Recipe Flexible Execution Workbench, an environment to run ESO VLT pipelines which employs a workflow engine (Kepler[1]) to provide a real-time visual representation of a data reduction cascade, called a workflow, which can be easily understood by most astronomers. The basic philosophy and concepts of Reflex have been discussed by Freudling et al. (2013, A&A, 559, 96). Please reference this article if you use `Reflex` for your research.

Reflex and the data reduction workflows have been developed at ESO and they are fully supported. If you have any issue, please contact *usd-help@eso.org* for further support.

This document is a tutorial designed to enable the user to employ the MUSE workflow to reduce his/her data in a user-friendly way, concentrating on high-level issues such as data reduction quality and signal-to-noise (S/N) optimisation.

A workflow accepts science and calibration data, as delivered to PIs in the form of PI-Packs (until October 2011) or downloaded from the archive using the CalSelector tool[2] and organises them into DataSets, where each DataSet contains one science object observation (possibly consisting of several science files) and all associated raw and static calibrations required for a successful data reduction. The data organisation process is fully automatic, which is a major time-saving feature provided by the software. The DataSets selected by the user for reduction are fed to the workflow which executes the relevant pipeline recipes (or stages) in the correct order. Full control of the various recipe parameters is available within the workflow, and the workflow deals automatically with optional recipe inputs via built-in conditional branches. Additionally, the workflow stores the reduced final data products in a logically organised directory structure employing user-configurable file names.

The MUSE `Reflex` workflow is designed to process and combine the scientific exposures withing the same Observing Block (OB). The output datacube is therefore the combined contribution of all the target observations within the OB. It also produces as output the reduced pixel tables of the individual target exposures. Therefore, it is possible to recombine the individual reduced frames for alignment optimization, or combine them with those of others OBs for the same target via external call of the `muse_exp_combine` recipe (see the MUSE Data Reduction Cookbook).

The MUSE `Reflex` workflow handles observations that have one or none sky exposure in the same OB of the target. In the latter case, the sky is evaluated in regions in the field of view where the target contribution is negligible. The most relevant parameters for the sky subtraction strategy can be specified directly in the `Reflex` canvas.

[1] *http://kepler-project.org*
[2] *http://www.eso.org/sci/archive/calselectorInfo.html*

2 Software Installation

The software pre-requisites for `Reflex 2.6` may be found at:
http://www.eso.org/sci/software/pipelines/reflex_workflows

To install the `Reflex 2.6` software and demo data, please follow these instructions:

1. From any directory, download the installation script:

   ```
   wget ftp://ftp.eso.org/pub/dfs/reflex/install_reflex
   ```

2. Make the installation script executable:

   ```
   chmod u+x install_reflex
   ```

3. Execute the installation script:

   ```
   ./install_reflex
   ```

   and the script will ask you to specify three directories: the download directory `<download_dir>`, the software installation directory `<install_dir>`, and the directory to be used to store the demo data `<data_dir>`. If you do not specify these directories, then the installation script will create them in the current directory with default names.

4. You will be given a choice of pipelines (with the corresponding workflows) to install. Please specify the numbers for the pipelines you require, separated by a space, or type "A" for all pipelines.

5. To start `Reflex`, issue the command:

   ```
   <install_dir>/bin/reflex
   ```

   It may also be desirable to set up an alias command for starting the `Reflex` software, using the shell command `alias`. Alternatively, the `PATH` variable can be updated to contain the `<install_dir>/bin` directory.

3 System requirements

3.1 Hardware

The processing of MUSE data is very demanding in terms of computing resources. In particular, it requires a machine with sufficient memory installed, and it is available only for 64-bit machines. The recommended platform is a powerful workstation with a recent 64-bit Linux system.

The recommended configuration of the target machine for creating the final data cube from *a single* MUSE observation and the required set of calibrations is:

- 64 GB of memory
- 24 CPU cores (physical cores)
- 4 TB of free disk space
- GCC 4.8.2 (or newer)

Scientific programs usually foreseen the creation of a datacube by merging multiple exposures taken at the same position. On average, the memory consumption grows linearly with the number of observations.

In the case of creation of mosaic, the size of the data cube may become really huge, and the required memory grows accordingly.

For more information, please refer to the MUSE Data Reduction Cookbook available at *http://www.eso.org/sci/software/pipelines/*.

3.2 JVM Memory set-up

The MUSE workflow need a sufficient amount of memory. The best way to set the memory allocation of `Reflex` is to run the `reflex_set_memory` script that is distributed with `Reflex` *before* starting `Reflex`. The recommended setting for MUSE is to leave the “Minimum amount of memory” unchanged, and set the “Maximum amount of memory” to 2000. Alternatively, the memory setting can be done after starting `Reflex` by clicking on "Tools – JVM Memory Settings" in the menu bar. `Reflex` needs to be restarted for this change to be applied.

3.3 Execution on machines with less than 64 GB of memory

The MUSE pipeline and the `Reflex` workflow can be still executed in less powerful machines, such as laptops with 8GB of RAM, provided that the user restricts the wavelength range to short interval (e.g. 100 Å). This set-up, although still demanding in terms of computational time, allows the user to test the data reduction strategy before having access to a more powerful machine and reduce the data on the full wavelength range. For example, it can be used to create sky masks, to find the best method and parameters for the sky subtraction in critical wavelength ranges, to calculate the coordinate offsets between different exposures, and much more. More information are in Section 6.4.

4 Demo Data

A demo dataset is distributed together with the MUSE `Reflex` workflow. It consists in two target exposures, one off-set sky exposure, on-sky calibration frames (sky flats, standard star), and instrument calibration frames (biases, flats, arcs). Offset-sky and target exposures are taken within the same Observing Block. In addition, the set of static calibrations included in the pipeline distribution is needed for the reduction of the demo dataset.

Warning: The files that are downloaded through the ESO archive, are compressed with fpack (they have the .fz extension). The files need to be uncompressed before running the workflow.

- **Target exposures.** Frame Tags: `PRO.CATG=OBJECT`; `DPR.TYPE=OBJECT`
  - `MUSE_WFM-NOAO_OBS173_0069.fits`
  - `MUSE_WFM-NOAO_OBS173_0071.fits`
- **Offset-sky exposures.** Frame Tags: `PRO.CATG=SKY`; `DPR.TYPE=SKY`
  - `MUSE_WFM-NOAO_OBS173_0070.fits`
- **On-Sky calibration exposures.**
  - **Sky flats.** Frame Tags: `PRO.CATG=SKYFLAT`; `DPR.TYPE=FLAT,SKY`
    - `MUSE_WFM_SKYFLAT172_0001.fits`
    - `MUSE_WFM_SKYFLAT172_0002.fits`
    - `MUSE_WFM_SKYFLAT172_0003.fits`
    - `MUSE_WFM_SKYFLAT172_0004.fits`
    - `MUSE_WFM_SKYFLAT172_0005.fits`
  - **Standard star.** Frame Tags: `PRO.CATG=STD`; `DPR.TYPE=STD`
    - `MUSE_WFM_STD172_0002.fits`
- **Instrument calibration exposures.**
  - **Biases.** Frame Tags:`PRO.CATG=BIAS`; `DPR.TYPE=BIAS`
    - `MUSE_CAL_BIAS173_0004.fits`
    - `MUSE_CAL_BIAS173_0005.fits`
    - `MUSE_CAL_BIAS173_0006.fits`
    - `MUSE_CAL_BIAS173_0007.fits`
    - `MUSE_CAL_BIAS173_0008.fits`
  - **Flat fields.** Frame Tags:`PRO.CATG:FLAT`; `DPR.CATG:FLAT,SKY`
    - `MUSE_WFM_FLAT172_0049.fits`
    - `MUSE_WFM_FLAT172_0050.fits`
    - `MUSE_WFM_FLAT172_0051.fits`
    - `MUSE_WFM_FLAT172_0052.fits`
    - `MUSE_WFM_FLAT172_0053.fits`

  - **Wavelength calibrations.** Frame Tags:`PRO.CATG=ARC`; `DPR.TYPE=WAVE`
    * `MUSE_WFM_WAVE173_0001.fits`
    * `MUSE_WFM_WAVE173_0002.fits`
    * `MUSE_WFM_WAVE173_0003.fits`
    * `MUSE_WFM_WAVE173_0004.fits`
    * `MUSE_WFM_WAVE173_0005.fits`
    * `MUSE_WFM_WAVE173_0006.fits`
    * `MUSE_WFM_WAVE173_0007.fits`
    * `MUSE_WFM_WAVE173_0008.fits`
    * `MUSE_WFM_WAVE173_0009.fits`

- **Static calibrations.**
  - Files included in the pipeline distribution:
    * `astrometry_wcs_wfm_comm2a.fits` Frame Tags:`PRO.CATG=ASTROMETRY_WCS`
    * `filter_list.fits` Frame Tags:`PRO.CATG=FILTER_LIST`
    * `lsf_table_comm2a.fits` Frame Tags:`PRO.CATG=LSF_TABLE`
    * `badpix_table-21.fits` Frame Tags:`PRO.CATG=BADPIX_TABLE`
    * `geometry_table_wfm_comm2a.fits` Frame Tags:`PRO.CATG=GEOMETRY_TABLE`
    * `sky_lines.fits` Frame Tags:`PRO.CATG=SKY_LINES`
    * `extinct_table.fits` Frame Tags:`PRO.CATG=EXTINCT_TABLE`
    * `line_catalog.fits` Frame Tags:`PRO.CATG=LINE_CATALOG`
    * `std_flux_table.fits` Frame Tags:`PRO.CATG=STD_FLUX_TABLE`

5 Quick Start: Reducing The Demo Data

In this Section we describe how to reduce the demo data supplied with the `Reflex` 2.6 release. The steps to follow are listed below:

1. Start the `Reflex` application by typing

   ```
   reflex &
   ```

   at the terminal command line. The empty `Reflex` canvas will appear (Figure 5.1). We refer to Section 7 for a description of the main elements of the canvas.

2. If you have not set the Memory alocation before (see section 3.2), then set the JVM Memory settings, and restart `Reflex`.

3. Open the MUSE workflow by clicking on `File -> Open...`, and then selecting the file `muse-0.18.5/muse.xml` in the file browser. A new `Reflex` window will appear, containing the MUSE workflow, as shown in Figure 5.2.

4. To aid visual tracking of the reduction cascade, it is advisable to use component (or actor) highlighting. Click on `Tools -> Animate at Runtime`, enter the number of milliseconds representing the animation interval (1 ms is recommended), and click OK.

5. Specify the location of the directories containing the data, static calibration, and the desired output directories. This can be done by double clicking with the mouse on the appropriate fields in the workflow canvas, under the "Set-up directories", and selecting the correct path on the file browser that will pop-up. In the current workflow installation, all the set-up directories are configured with your system by default, and they point to the location where the demo data and the required static calibrations are located. Further information on the set-up directories are given in Section 8.1.

6. If the files in the input directory are compressed, uncompress them. Demo data are already uncompressed.

7. Specify the desired data reduction strategy, by setting the relevant parameters in the workflow canvas, under the "Data reduction strategy parameter". The description of these parameters will be done in Section 5.1.

8. Press  to start the workflow. The Data Organizer probes the input directories and group the files on the basis of header information and a set of OCA [3] rules to create the datasets to be reduced.

9. The reduction of the demo data has started. The Data Organizer probes the input directories and group the files on the basis of header information and a set of `OCA` rules to create the datasets to be reduced. The `DataChooser` window pops up (Figure 5.3), allowing the user to inspect the files that compose the datasets that have been created by the Data Organizer. Press "Continue" to continue the data reduction.

[3] `OCA` stands for Organization, Classification, Association and refers to rules, which allow to classify the raw data according to the contents of the header keywords, organize them in appropriate groups for processing, and associate the required calibration data for processing. They can be found in the directory `< install_dir > /share/esopipes/ < pipeline-version > /reflex/` , carrying the extension `.oca`

10. The workflow processes and combines the exposures within the same observing block. To reduce the single exposures separately, please consult Section 6. The workflow executes the following pipeline recipes:

    - **muse_bias**.
    - **muse_dark** (if dark frames are available).
    - **muse_flat**.
    - **muse_wavecal**.
    - **muse_skyflat** (if sky flat observations are available).
    - **muse_scipost**.
    - **muse_scibasic**.
    - **muse_standard**.
    - **muse_create_sky** (if offset-sky observations are available).
    - **muse_astrometry** (on demand, and if astrometric observations are available).
    - **muse_scipost**.

    We refer the reader to the MUSE Cookbook for further information on these recipes and their parameters.

    At the end of the data reduction, the `Reflex` Provenance Explorer pups up and the products association tree can be explored. Press “Continue” on the Provenance Explorer window to conclude the data reduction.

    Warning: Due to the large amount of files produced by the MUSE data reduction pipeline, the current version of the `Reflex` Provenance Explorer is not able to handle the entire file association tree. To avoid unexpected crash of the workflow, or memory allocation errors, it is recommended to click on “Continue” without exploring the files. This bug will be fixed in the next pipeline release.

The following final products `DATACUBE_FINAL`, `IMAGE_FOV`, and the `PIXELTABLE_REDUCED`s are saved in the `reflex_end_product` directory. The exact name of the file depends on the header keywords of the input dataset.

5.1 Setting the data reduction strategy

All the recipe parameters can be changed by configuring the associated `RecipeExecuter` actor This can be done by opening the various composite actors until the desired `RecipeExecuter` is visible. To open a composite actor, click on it with the mouse right button, and select "Open Actor". To configure the desired `RecipeExecuter` click on it with the mouse right button, and select “Configure Actor”.

The main `Reflex` canvas offers to the user a quick selection of some key parameters and options, which are relevant to select the appropriate strategy for the data reduction. They are located in the section “Data reduction strategy parameters” (Figure 5.4).

- **Calibrations parameters.**

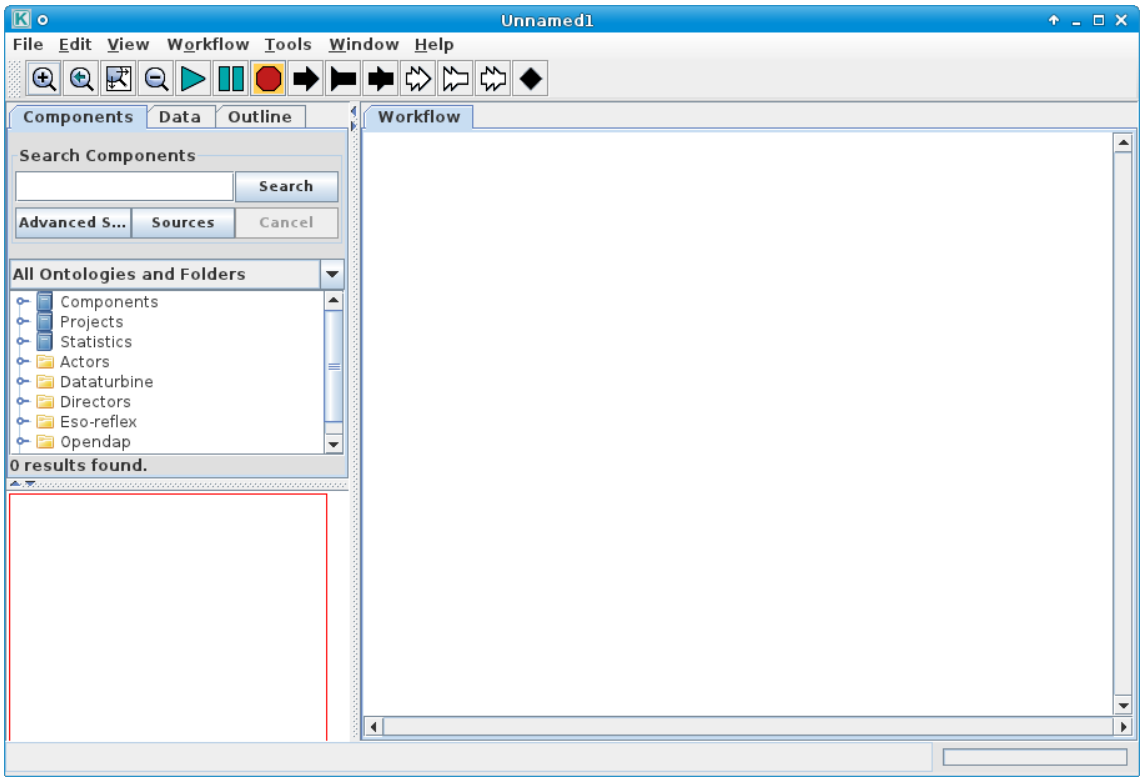


Figure 5.1: The empty `Reflex` canvas.

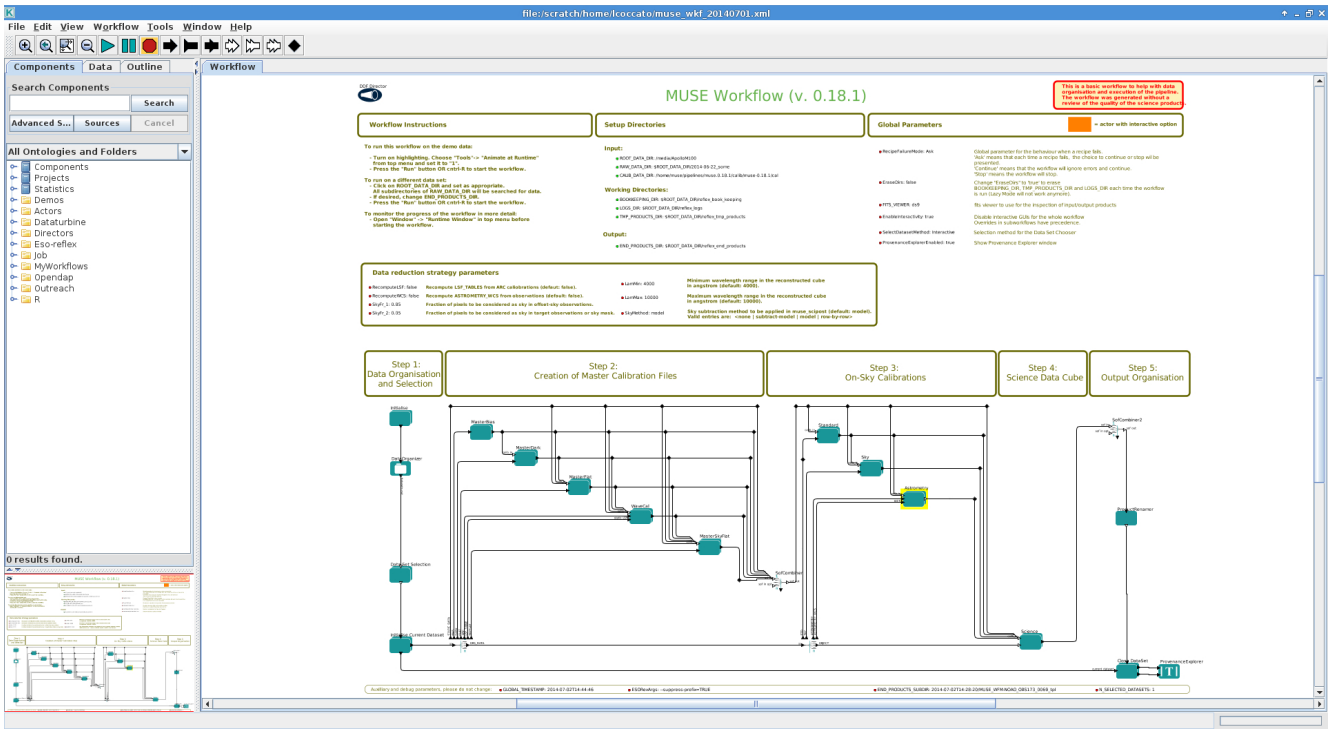


Figure 5.2: The MUSE `Reflex` workflow.

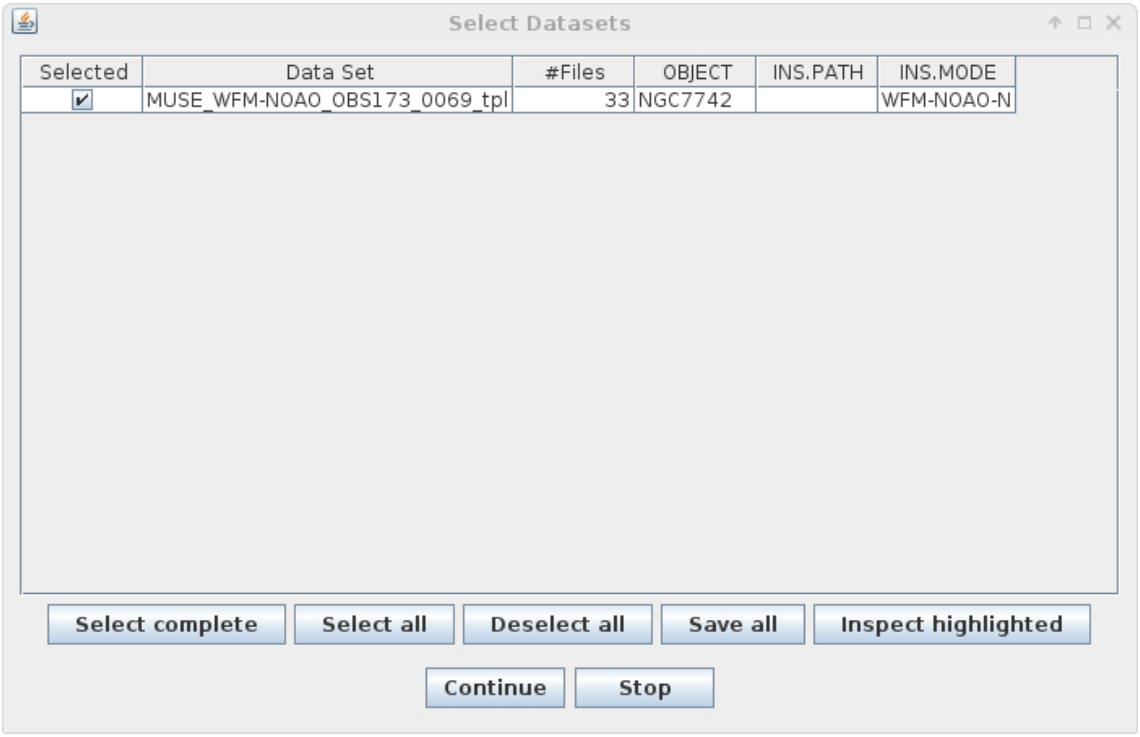


Figure 5.3: The Select Dataset window.

Figure 5.4: The Data reduction strategic parameters section in the `Reflex` canvas.

  - **RecomputeLSF.** If set to *false*, the `LSF_TABLES` are selected from the static calibration directory (recommended option); this is equivalent of setting `-lsf=false` in the **muse_wavecal** recipe. If the parameter is not set to "false", the `LSF_TABLES` will be recomputed from the `ARC` calibrations available in the dataset; this is equivalent to set `-lsf=true` in the **muse_wavecal** recipe. Default: false.
  - **RecomputeWCS.** If set to *false*, the `ASTROMETRIC_WCS` are selected from the static calibration directory (recommended option). Otherwise, they will be recomputed from the `ASTROMETRY` calibrations available in the dataset. Default: false.

- **Wavelength range parameters.** If you are interested in a restricted wavelength range it is possible to create the `DATACUBE_FINAL` accordingly. The following parameters have an affect on the **muse_create_sky**, and **muse_scipost** recipes.

  - **LamMin**. Sets the minimum wavelength (in Å) to consider when reconstructing the datacube in the **muse_create_sky**, **muse_astrometry**, and **muse_scipost** recipes. It corresponds to the recipe parameter `-lambdamin`. Default: 4000.
  - **LamMax.** Sets the maximum wavelength (in Å) to consider when reconstructing the datacube in the **muse_create_sky**, **muse_astrometry**, and **muse_scipost** recipes. It corresponds to the recipe parameter `-lambdamin`. Default: 10000.

  The recipe **muse_standard** is not affected by **LamMin** and **LamMax**, because the change of the corresponding `-lambdamin` and `-lambdamax` parameters will cause the recipe to fail. If you are using on a computer with limited memory capabilities (see Section for hardware specifications 3.1) the **muse_standard** will fail in reconstructing the datacube and the workflow will crash. To avoid that, i) manually set `-lambdabin` and `-lambdamax` to a short wavelength range, and ii) set the "RecipeExecute" mode to "continue". This can be achieved by opening the "Standard" composite actor (mouse right-button – Open Actor), and configuring the appropriate recipe parameters in the corresponding `RecipeExecuter` actor (mouse right-button – Configure Actor).

  In this way the workflow does not crash, but the final datacube will not be corrected for instrumental response or telluric absorption.

- **Strategy for sky subtraction.** The following parameters are relevant for the sky subtraction. Each dataset might require different values.

  - **SkyFr_1.** It corresponds to the recipe parameter `-fraction` in the **muse_ create_sky** recipe.
  - **SkyFr_2.** It corresponds to the recipe parameter `-skymodel_fraction` in the **muse_scipost** recipe.
  - **SkyMethod.** Method for sky subtraction. It corresponds to the `-skymethod` parameter in the **muse_scipost** recipe.

  If dedicated sky observations are present, good values could be **SkyFr_1** = 0.85, **SkyFr_2** = 0., and **SkyMethod** = subtract-model. If offset sky observations are not present, the sky will be evaluated from a specified fraction of pixels in the target field of view; good values could be **SkyFr_2** = 1.0 and **SkyMethod** = model; The parameter **SkyFr_1** has no effect.

  Warning: if **SkyMethod** = model, **SkyFr_2** cannot be 0, otherwise the workflow crashes.

6 Tips and tricks

6.1 Optimizing the combination of different exposures

It is possible that be that the final combined datacube cube (DATACUBE_FINAL) suffers for inaccurate coordinate offsets between different exposures. In this case, it is needed to correct the exposures by executing pipeline recipe with the esorex command line, outside the Reflex environment. The steps to follow are listed below. We refer the user to the MUSE Data Reduction Cookbook for additional information.

- Create images of the the individual field of views from the PIXELTABLE_REDUCED using **muse_scipost_make_cube** (type esorex -man muse_scipost_make_cube on the terminal to display the recipe description).
- Compare the individual field of views and calculate the desired offsets in RA and DEC. The offsets are expressed in degrees on the sky plane (i.e. do not apply the $\cos\delta$ correction) and defined as follow:

  ```
  RA_OFFSET = RA_MEASURED - RA_REFERENCE
  DEC_OFFSET = DEC_MEASURED - DEC_REFERENCE
  ```

  It is important that the offsets are given in the order of increasing DATE-OBS of the exposures involved. If one of the exposures has been chosen as the reference the offsets of 0 for this exposure have to be explicitly given when setting the environment variables.
- Merge the PIXELTABLE_REDUCED into a final datacube by specifying the offsets using **muse_exp_combine**. An example command to type on the terminal is the following (single command line):

  ```
  MUSE_XCOMBINE_RA_OFFSETS="0.0000000, -5.4920943e-05, -0.00031381379" \
  MUSE_XSCOMBINE_DEC_OFFSETS="0.0000000, 8.3338105e-05, 7.2223397e-05" \
  esorex muse_exp_combine --skymethod=none pixtab_reduced.sof
  ```

  where pixtab_reduced.sof is the file containing the names of the PIXELTABLE_REDUCED to be combined. Note that -skymethod is set to "none", because the sky has been already subtracted from the input pixel tables during the execution of the reflex workflow (unless **SkyMethod** was explicitly set to none).

6.2 Reduction of single files: multiple offset sky exposures within one OB

The workflow processes and combines the exposures which are taken within the same OB. Only one sky exposure within each OB is selected in the data reduction cascade. In the case that many offset-sky exposures are present in a single OB, it is advisable to process the object exposures individually. This strategy has the advantage that each OBJECT exposure will be reduced with the SKY exposure closest in time. On the other hand, it has the disadvantage that the merged datacube must be combined separately, via esorex command line following the procedure explained in Section 6.1. Future releases of the Reflex workflow will overcome this issue.

To reduce the single exposures with the `Reflex` workflow, it is necessary to edit the OCA rules. The location of the OCA rule files can be seen by clicking with the right button of the mouse on the `DataSelector` Actor and click on "Configure Actor". It is recommended to back-up the original OCA rule file before editing.

Open the OCA rule file with a text editor and change the line If the user prefers to process each object exposure individually, the OCA rule file needs to be edited; change the line:

```
where REFLEX.CATG == "OBJECT" group by TPL.START as (TPL\_A, tpl);
```

into:

```
where REFLEX.CATG == "OBJECT" group by ARCFILE as (TPL\_A, tpl);
```

The `PIXTABLE_REDUCED` can be now merged into the final `DATACUBE_FINAL` following the prescriptions listed in Section 6.1.

6.3 Combining data from different OBs

The combination of exposures of the same target, but that are executed in different Observing Blocks is at the moment not supported in `Reflex`. The best strategy is to execute `Reflex` on each OB and merge all the `PIXTABLES_REDUCED` in a single datacube. Because the different OBs might have been executed distant in time, it is reccommended to correct the exposures for coordinate offsets, as explained in Section 6.1.

6.4 Execution of the workflow on computers with limited Memory

As discussed in Section 3.2 it is possible to analize a short wavelength range and reduce a dataset on a computer with limited ram (e.g. 8 Gb of memory).

This can be done by changing the **LamMin**, **LamMax** parameters, as illustrated in Section 5.1. However, the recipe **muse_standard** is not affected by these parameters, and any change of the `-lambdamin` and `-lambdamax` parameters in the corresponding `RecipeExecuter` actor will cause the recipe to fail in allocating the needed memory and the workflow to crash. This is due to a bug in the **muse_standard** recipe.

To avoid that:

- open the "Standard" composite actor (mouse right-button – Open Actor), and configure the "RecipeExecutor" associated to the **muse_standard** recipe (mouse right-button – Configure Actor).
- manually set `-lambdabin` and `-lambdamax` to a short wavelength range inside the `RecipeExecuter` actor associated to **muse_standard**. This can be done
- set the `Recipe execution mode` to "Continue".

In this way **muse_standard** will still fail, but the workflow will not crash, but the final datacube will not be corrected for instrumental response or telluric absorption.

This problem will be fixed in future MUSE pipeline releases.

7 About The Reflex Canvas

7.1 Saving And Loading Workflows

In the course of your data reductions, it is likely that you will customise the workflow for various data sets, even if this simply consists of editing the `ROOT_DATA_DIR` to a different value for each data set. Whenever you modify a workflow in any way, you have the option of saving the modified version to an `XML` file using `File -> Export As` (which will also open a new workflow canvas corresponding to the saved file). The saved workflow may be opened in subsequent `Reflex` sessions using `File -> Open`. Saving the workflow in the default format (.kar) is only advised if you do not plan to use the workflow in another computer.

7.2 Buttons

At the top of the `Reflex` canvas are a set of buttons which have the following useful functions:

-  - Zoom in.
-  - Reset the zoom to 100%.
-  - Zoom the workflow to fit the current window size (Recommended).
-  - Zoom out.
-  - Run (or resume) the workflow.
-  - Pause the workflow execution.
-  - Stop the workflow execution.

The remainder of the buttons (not shown here) are not relevant to the workflow execution.

7.3 Workflow States

A workflow may only be in one of three states: executing, paused, or stopped. These states are indicated by the yellow highlighting of the  ,  , and  buttons, respectively. A workflow is executed by clicking the  button. Subsequently the workflow and any running pipeline recipe may be stopped immediately by clicking the  button, or the workflow may be paused by clicking the  button which will allow the current actor/recipe to finish execution before the workflow is actually paused. Note that after clicking the  button, it is possible that more than one actor is executed, since this behaviour depends on the workflow scheduling. For instance, if there are two actors in parallel, and you pause the workflow while one is being executed, then both of them will be executed before the workflow is actually paused. After pausing, the workflow may be resumed by clicking the  button again.

8 The MUSE Workflow

The MUSE workflow canvas is organised into a number of areas. From top-left to top-right you will find general workflow instructions, directory parameters, and global parameters. In the middle row you will find five boxes describing the workflow general processing steps in order from left to right, and below this the workflow actors themselves are organised following the workflow general steps.

8.1 Workflow Canvas Parameters

The workflow canvas displays a number of parameters that may be set by the user. Under "Setup Directories" the user is only required to set the `RAWDATA_DIR` to the working directory for the DataSet(s) to be reduced, which, by default, is set to the directory containing the demo data. The `RAWDATA_DIR` is recursively scanned by the `Data Organiser` actor for input raw data. The directory `CALIB_DATA_DIR`, which is by default within the pipeline installation directory, is also scanned by the `Data Organiser` actor to find any static calibrations that may be missing in your DataSet(s). If required, the user may edit the directories `BOOKKEEPING_DIR`, `LOGS_DIR`, `TMP_PRODUCTS_DIR`, and `END_PRODUCTS_DIR`, which correspond to the directories where book-keeping files, logs, temporary products and end products are stored, respectively (see the Reflex User Manual for further details; [1]).

Under the "Global Parameters" area of the workflow canvas, the user may set the `FITS_VIEWER` parameter to the command used for running his/her favourite application for inspecting FITS files. Currently this is set by default to `fv`, but other applications, such as `ds9`, `skycat` and `gaia` for example, may be useful for inspecting image data.

By default the `EraseDirs` parameter is set to `false`, which means that no directories are cleaned before executing the workflow, and the recipe actors will work in Lazy mode (see Section 8.2.2), reusing the previous pipeline recipe outputs where input files and parameters are the same as for the previous execution, which saves considerable processing time. Sometimes it is desirable to set the `EraseDirs` parameter to `true`, which forces the workflow to recursively delete the contents of the directories specified by `BOOKKEEPING_DIR`, `LOGS_DIR`, and `TMP_PRODUCTS_DIR`. This is useful for keeping disk space usage to a minimum and will force the workflow to fully rereduce the data each time the workflow is run.

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

The parameter `GlobalInteractivity` controls whether the interactive windows will appear for those windows which are *enabled* by default. The possible values are `true, false`. Take into account that some windows are disabled in the default configuration and therefore are not affected by this parameter.

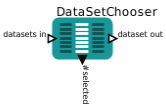
The parameter `ProvenanceExplorerEnabled` controls whether the `ProvenanceExplorer` actor will show its window or not. The possible values are `true, false`.

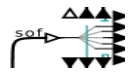
8.2 Workflow Actors

8.2.1 Simple Actors

Simple actors have workflow symbols that consist of a single (rather than multiple) green-blue rectangle. They may also have a logo within the rectangle to aid in their identification. The following actors are simple actors:

- 

- The `Data Organiser` actor.
- 

- The `Data Set Chooser` actor.
- 

- The `Fits Router` actor
- 

- The `Product Renamer` actor.
- 

- The `Provenance Explorer` actor.

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

8.2.2 Lazy Mode

By default, all recipe executer actors in a pipeline workflow are "Lazy Mode" enabled. This means that when the workflow attempts to execute such an actor, the actor will check whether the relevant pipeline recipe has already been executed with the same input files and with the same recipe parameters. If this is the case, then the actor will not execute the pipeline recipe, and instead it will simply broadcast the previously generated products to the output port. The purpose of the Lazy mode is therefore to minimise any reprocessing of data by avoiding data rereduction where it is not necessary.

One should note that the actor Lazy mode depends on the contents of the directory specified by `BOOKKEEPING_DIR` and the relevant FITS file checksums. Any modification to the directory contents and/or the file checksums will cause the corresponding actor when executed to run the pipeline recipe again, thereby rereducing the input data.

The forced rereduction of data at each execution may of course be desirable. To force a rereduction of all data for all `RecipeExecuter` actors in the workflow (i.e. to disable Lazy mode for the whole workflow), set the `EraseDirs` parameter under the "Global Parameters" area of the workflow canvas to `true`. This will then

remove all previous results as well. To force a rereduction of data for any single `RecipeExecuter` actor in the workflow (which will be inside the relevant composite actor), right-click the `RecipeExecuter` actor, select `Configure Actor`, and uncheck the Lazy mode parameter tick-box in the "Edit parameters" window that is displayed.

9 Test Cases

The MUSE `Reflex` workflow 0.18.5 has been tested with the following obserational set-up detailed in Table 9.1:

Table 9.1: Files contained in the datasets on which the workflow has been tested

FRAMES	Dataset 1	Dataset 2	Dataset 3	Dataset 4
Offset sky observations	No	No	Yes	Yes
Sky flat observations	No	No	No	No
Astrometric observations	Yes (used)	Yes (not used)	Yes (used)	Yes (not used)
`LSF_TABLES` computed from raw	No	No	No	No
FRAMES	Dataset 5	Dataset 6	Dataset 7	Dataset 8
Offset sky observations	Yes	Yes	Yes	Yes
Sky flat observations	Yes	Yes	Yes	Yes
Astrometric observations	Yes(used)	Yes (not used)	No	No
`LSF_TABLES` computed from raw	No	No	No	Yes

10 Frequently Asked Questions

- **Where are my intermediate pipeline products?** Intermediate pipeline products are stored in the directory <TMP_PRODUCTS_DIR> (defined on the workflow canvas) and organised further in directories by pipeline recipe.
- **Can I use different sets of bias frames to calibrate my flat frames and science data?** Yes. In fact this is what is currently implemented in the workflow(s). Each file in a DataSet has a purpose attached to it ([1]). It is this purpose that is used by the workflow to send the correct set of bias frames to the recipes for flat frame combination and science frame reduction, which may or may not be the same set of bias frames in each case.
- **Can I launch `Reflex` from the command line?** Yes, use the command:

  ```
  reflex -n <workflow_path>/<workflow>.xml
  ```

  The -n option will set all the different optiions to Kepler and the workflows that avoid any graphical display (including pipeline interactive windows). Note that this mode is not fully supported, and the user should be aware that the path to the workflow must be absolute and even if no GUI elements are shown, it still requires a connection to the window manager.
- **How can I add new actors to an existing workflow?** You can drag and drop the actors in the menu on the left of the `Reflex` canvas. Under `Eso-reflex -> Workflow` you may find all the actors relevant for pipeline workflows, with the exception of the recipe executer. This actor must be manually instantiated using `Tools -> Instantiate Component`. Fill in the "Class name" field with `org.eso.RecipeExecuter` and in the pop-up window choose the required recipe from the pull-down menu. To connect the ports of the actor, click on the source port, holding down the left mouse button, and release the mouse button over the destination port. Please consult the Reflex User Manual ([1]) for more information.
- **How can I broadcast a result to different subsequent actors?** If the output port is a multi-port (filled in white), then you may have several relations from the port. However, if the port is a single port (filled in black), then you may use the black diamond from the toolbar. Make a relation from the output port to the diamond. Then make relations from the input ports to the diamond. Please note that you cannot click to start a relation from the diamond itself. Please consult the Reflex User Manual ([1]) for more information.
- **How can I run manually the recipes executed by Reflex?** If a user wants to re-run a recipe on the command line he/she has to go to the appropriate reflex_book_keeping directory, which is generally reflex_book_keeping/<workflow>/<recipe_name>_<number> There, subdirectories exist with the time stamp of the recipe execution (e.g. 2013-01-25T12:33:53.926/). If the user wants to re-execute the most recent processing he/she should go to the `latest` directory and then execute the script `cmdline.txt`. Alternatively, to ensure that the path to `esorex` is the correct one, the user can execute

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

where INSTALL_DIR is the directory where Reflex and the pipelines were installed. If the user knows the name of the input raw files for the recipe, the correct directory among the many time stamps can be found via `grep <raw_file> */data.sof`. Afterwards the procedure is the same as before. The products will appear in the directory from which the recipe is called, and not in the reflex_tmp_products or reflex_end_products directory, and they will not be renamed.

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