European Organisation for Astronomical Research in the Southern Hemisphere

Programme: GEN

Project/WP: Science Operation Software Department

ERIS-SPIFFIER Pipeline User Manual

Document Number:

Document Version: 1.9.6

Document Type: Manual (MAN)

Released on: 2026-04-20

Document Classification: Public

Prepared by: ERIS-SPIFFIER Pipeline Team

Validated by:

Approved by:

Name

Authors

Name	Affiliation
A. Modigliani, M. Neeser	ESO
E. Wiezorrek, Y. Cao	MPE

Change Record from previous Version

Release	Affected Section(s)	Changes/Reason/Remarks
1.1.0	All	First public release
1.1.3	None	
1.2.0	Sec 8, 9, 12	Aligned to pipeline update
1.2.5	Sec 1, 5, 8, 12	Improve document
1.4.1	Sec 1, 5, 7, 8, 9, 11,12, 13, App A	Improve document
1.4.4	Sec 5, 8, 12	Align to pipeline update
1.5.0	Sec 1, 4, 5, 8, 12	Align to pipeline update
1.5.3	None	None
1.6.0	Sec 5	Align to pipeline update
1.8.1	Sec 5	Align to pipeline update
1.9.6	Sec 5, 8, 12	Update how to set offsets for combination of cubes, align to pipeline update

Contents

1 Introduction

1.1 Purpose

The ERIS-SPIFFIER pipeline is a subsystem of the VLT Data Flow System (DFS). It is used in two operational environments, for the ESO Data Flow Operations (DFO), and the Paranal Science Operations (PSO), in the quick-look assesment of data, in the generation of master calibration data, in the reduction of scientific exposures, and in the data quality control. Additionally, the ERIS-SPIFFIER pipeline recipes are made public to the user community, to allow a more personalised processing of the data of the instrument. The purpose of this document is to describe a typical ERIS-SPIFFIER data reduction sequence with the ERIS-SPIFFIER pipeline. This manual is a complete description of the data reduction recipes implemented by the ERIS-SPIFFIER pipeline, reflecting the status of the ERIS-SPIFFIER pipeline as of version 1.9.6.

1.2 Scope

This document describes the ERIS-SPIFFIER pipeline used at ESO to assess instrument and data quality control.

The examples on running individual pipeline recipes in this manual use the esorex command and manually created list of input files. Several interfaces to automatically organise the data, create the list of input files and execute the pipeline recipes in the proper sequence are available, see the `ESO pipeline page` for details.

Please note that the use of Gasgano as a GUI for processing data is deprecated. Its use is no longer recommended and the related section in this manual, as well as support for Gasgano as a data processing GUI application in general will be dropped entirely in a future release.

Updated versions of the present document may be found on [**?**]. For general information about the current instrument pipelines status we remind the user of [**?**] . Additional information on CFITSIO, the Common Pipeline Library (CPL) and EsoRex can be found respectively at [**?**], [**?**], [**?**]. The Reflex front-ends is described in [**?**] and a description of the instrument can be found in [**?**]. The ERIS instrument user manual is in [**?**], the calibration plan in [**?**]. The ERIS pipeline project in a nutshell is also described in [**?**].

1.3 Acknowledgements

The ERIS-SPIFFIER pipeline was designed, implemented and verified by Erich Wiezorrek, Alex Agudo and Yixian Cao of the Max Planck Institute for Extraterrestrial Physics of the ERIS consortium. Kateryna Kravchenko participated in the ERIS commissioning and contributed to the pipeline verification.

Mark Neeser, from the ESO Science Data Quality group, led this pipeline project since its beginning and contributed to its design and overall scientific verification.

Isabelle Percheron, from the ESO Science Data Quality group, is responsible of the integration of the pipeline in the Quality Control environment, and from early commissioning, provided valuable feedback on the data and its quality control.

Andrea Modigliani, from the Science Operations Software Pipeline Systems Group, followed since its beginning

the development of the pipeline to ensure that the algorithms are properly implemented and that the code is robust and easy to maintain. He implemented the ESOReflex workflow, added quality control parameters, contributed to the implementation and verification of several functionalities used to reduce observations like efficiency and response computation, flux calibration and Strehl determination, updated science products to be compliant with the phase3 standard, provided support for the commissioning and Science Verification runs, pipeline operations and contributed to the documentation. Since January 2023 he is the single ESO ERIS pipeline maintainer.

Lars Lundin, from the Science Operations Software Pipeline Systems Group, contributed to improve software maintainability and portability and implementing several recipe regressions tests.

This release and documentation includes improvements to address feedback from scientists and users. We thank Julia Bodensteiner, an ESO fellow during the period of ERIS commissioning and early operations, who provided several useful suggestions to improve the SPIFFIER data reduction.

1.4 Stylistic conventions

Throughout this document the following stylistic conventions are used:

bold	in text sections for commands and other user input which has to be typed as shown
italics	in the text and example sections for parts of the user input which have to be replaced with real contents
`teletype`	in the text for FITS keywords, program names, file paths, and terminal output, and as the general style for examples, commands, code, etc

In example sections expected user input is indicated by a leading shell prompt.

In the text **bold** and *italics* may also be used to highlight words.

1.5 Notational Conventions

Hierarchical FITS keyword names, appearing in the document, are given using the dot–notation to improve readability. This means, that the prefix “HIERARCH ESO” is left out, and the spaces separating the keyword name constituents in the actual FITS header are replaced by a single dot.

2 Related Documents

[1] R. I. Davies. *A method to remove residual OH emission from near-infrared spectra*, March 2007. https://ui.adsabs.harvard.edu/abs/2007MNRAS.375.1099D.

[2] R. I. Davies, A. Agudo Berbel, E. Wiezorrek, M. Cirasuolo, N. M. Förster Schreiber, Y. Jung, B. Muschielok, T. Ott, S. Ramsay, J. Schlichter, R. Sharples, and M. Wegner. The Software Package for Astronomical Reductions with KMOS: SPARK, October 2013.

[3] ERIS Consortium. *ERIS Calibration Plan*. ESO-476498.

[4] ESO, *http://www.eso.org/pipelines*. *ERIS Pipeline Current Status*.

[5] ESO, *http://www.eso.org/pipelines*. *ERIS-SPIFFIER Pipeline Web Page*. .

[6] ESO, *http://www.eso.org/sci/facilities/paranal/instruments/eris/doc.html*. *ERIS User Manual*.

[7] ESO/SDD/DFS, *http://www.eso.org/observing/cpl/download.html*. *Common Pipeline Library User Manual*. VLT-MAN-ESO-19500-2720.

[8] ESO/SDD/DFS, *http://www.eso.org/cpl/esorex.html*. *ESOREX home page*.

[9] ESO/SDD/DFS, *http://www.eso.org/gasgano/* . *Gasgano User's Manual*. VLT-PRO-ESO-19000-1932.

[10] M. Galassi et al. *GNU Scientific Library Reference Manual*. *https://www.gnu.org/software/gsl/*, 3rd edition. IBSN: 0954612078.

[11] NASA, *http://heasarc.gsfc.nasa.gov/docs/software/fitsio/fitsio.html*. *CFITSIO home page*.

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

3 Definitions, Acronyms and Abbreviations

CalibDB	Calibration Database
CPL	Common Pipeline Library
CCD	Charge Coupled Device
DFS	Data Flow System
DRS	Data Reduction System
ESO	European Southern Observatory
EsoRex	ESO Recipe Execution Tool
FITS	Flexible Image Transport System
FOV	Field Of View
GSL	GNU Scientific Library[**?**]
GUI	Graphical User Interface
HDRL	High Level Data Reduction Library
LSF	Line Spread Function
ERIS	Enhanced Resolution Imager and Spectrograph
OB	Observation Block
pixel	picture element (of a raster image)
PSF	Point Spread Function
QC	Quality Control
SDP	Science Data Product
SOF	Set Of Frames
TBD	To be defined
TBC	To be confirmed
VLT	Very Large Telescope
WCS	World Coordinate System

4 Overview

In collaboration with instrument consortia, the Data Flow Systems Department (DFS) of the Data Management and Operation Division is implementing data reduction pipelines for the most commonly used VLT/VLTI instrument modes. These data reduction pipelines have the following three main purposes:

Data quality control: pipelines are used to produce the quantitative information necessary to monitor instrument performance.

Master calibration product creation: pipelines are used to produce master calibration products (*e.g.*, combined bias frames, super-flats, wavelength dispersion solutions).

Science product creation: using pipeline-generated master calibration products, science products are produced for the supported instrument modes. The accuracy of the science products is limited by the quality of the available master calibration products and by the algorithmic implementation of the pipelines themselves. In particular, adopted automatic reduction strategies may not be suitable or optimal for all scientific goals.

Instrument pipelines consist of a set of data processing modules that can be called from the command line, from the automatic data management tools available on Paranal, from EsoReflex, or from Gasgano.

ESO offers three front-end applications for launching pipeline recipes; namely *EsoReflex* [**?**], *Gasgano* [**?**] and *EsoRex* [**?**]. These applications can also be downloaded separately from `www.eso.org/reflex`, `www.eso.org/gasgano`, `www.eso.org/cpl/esorex.html` respectively.

The ERIS instrument and the different types of ERIS raw frames and auxiliary data are described in Sections 6, 9.

A brief introduction to the usage of the available reduction recipes using EsoRex is presented in Section 7.1. In section 8 we summarise known data reduction problems and solutions, if available.

An overview of the data reduction, what are the input data, and the recipes involved in the calibration cascade is provided in section 11.

More details on what are inputs, products, quality control measured quantities, and controlling parameters of each recipe is given in section 12.

More detailed descriptions of the data reduction algorithms used by the individual pipeline recipes can be found in Section 13.

In Appendix A the installation of the eris pipeline recipes is described and in section 3 a list of used abbreviations and acronyms is given.

5 What's new in pipeline release 1.9.6

This pipeline release includes the following improvements, several useful to desktop users:

- Constrained line fit in `eris_ifu_wavecal` and added QC parameters.
- Improved robustness of `eris_ifu_distortion`.
- Improved support for the new ESO Data Reduction System (EDPS) based data reduction.
- Updated documentation.

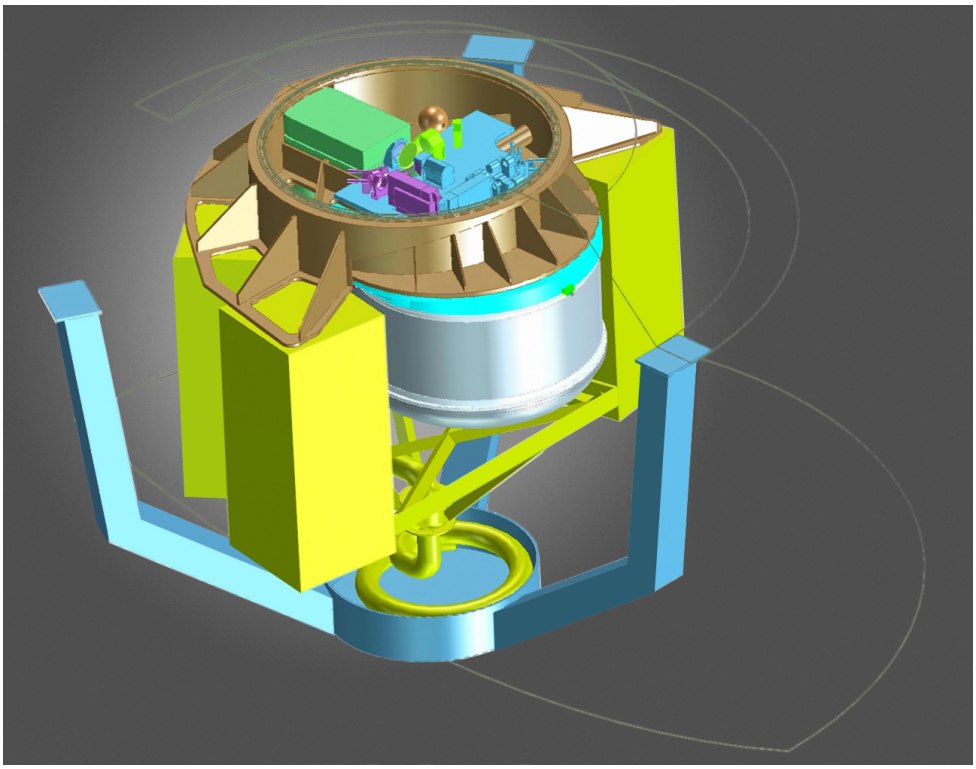


Figure 6.1: Graphical representations of ERIS.

6 The ERIS Instrument

ERIS is part of a new generation of technology for ESO's Very Large Telescope (VLT) located at the Paranal Observatory in Chile. The Enhanced Resolution Imager and Spectrograph (ERIS) instrument combines a general-use infrared imager and integral field spectrograph with the world-class adaptive optics installed on the VLT's Unit Telescope 4 (Yepun).

The versatility of ERIS lends itself to many fields of astronomical research and it aims to take the sharpest images obtained to date using a single 8.2-metre class telescope once operational. ERIS is expected to greatly contribute to probing of distant galaxies, the Galactic Centre, our Solar System and exoplanets.

The new, more sensitive technology of ERIS will succeed the successful NACO and SINFONI instruments, which have been used on the VLT since the early 2000s. ERIS is planned to work for 10 years following first light.

6.1 ERIS-SPIFFER, Spectrograph

The spectrograph of ERIS, named ERIS-SPIFFIER, is a refurbished version of SINFONI's SPIFFI (SPectrometer for Infrared Faint Field Imaging) instrument. The ERIS-SPIFFIER, as an integral field spectrograph, works in much the same way as SPIFFI. Each pixel of the spectrograph is capable of collecting a full spectrum, which

is the intensity of each wavelength measured by the instrument. The data collected is multidimensional and contains a lot of information for astronomers to analyse. This type of spectrograph allows astronomers to determine the relative position in space of each part of the target object. Combined with the infrared observing wavelength, this type of spectroscopy can for example allow astronomers to study the spin, structure, and content of high-redshift galaxies. A higher resolution grating on the spectrograph, used to split incoming light to capture a spectrum, works similarly to how a prism splits light into different colours.

6.2 ERIS-NIX, Infrared Imager

The imager on ERIS is the Near Infrared Camera System (NIX), the next generation of infrared imager for the VLT. Astronomers are able to use two main modes for the imager. One mode is direct imaging of astronomical objects. Another mode is coronagraphy, an observing technique that works by suppressing the direct light of a star. This technique on ERIS-NIX allows astronomers to observe exoplanets and discs of gas and dust around young stars.

ERIS-NIX is capable of collecting data in near-infrared between 3000 and 5000 nanometers, which are particularly difficult wavelengths to observe from the ground due to excess light interference from the sky. As such, ERIS is one of the few ground-based instruments with this wavelength coverage on an 8-metre class telescope and observing at these wavelengths allows for a new and unique view of exoplanets and galaxies.

6.3 ERIS and the Adaptive Optics Facility

ERIS is also equipped with an adaptive optics module that works together with the Adaptive Optics Facility (AOF) installed in 2017 on Yepun. Adaptive optics make real-time corrections for atmospheric turbulence that distorts starlight to allow for much sharper image quality, and are an important part of modern ground-based astronomical observations. The AOF capabilities in conjunction with the module allows ERIS to capture the sharpest data possible with the instrument and cover more of the sky than earlier VLT instruments.

7 Quick Start

7.1 The ERIS Pipeline Recipes

The ERIS-SPIFFIER pipeline reduction of imaging data chain uses the following recipes:

```
eris_ifu_dark          : This recipe perform dark data reduction
eris_ifu_detlin        : Detector's linearity & non linear bad pixels
eris_ifu_distortion    : This recipe performs distortion correction
eris_ifu_flat          : This recipe performs flat field data reduction
eris_ifu_wavecal       : This recipe performs the wavelength calibration
eris_ifu_stdstar       : This recipe reconstruct data cubes from std stars
eris_ifu_jitter        : This recipe reconstruct data cubes from object
eris_ifu_combine_hdrl  : Combine science cubes using hdrl.
eris_ifu_combine       : This recipe combines jittered cubes into a single cube
eris_ifu_pupil         : This recipe monitors the telescope pupil position
```

7.2 Running the ERIS Pipeline Recipes

7.2.1 Getting Started with EsoRex

EsoRex is a command-line tool which can be used to execute the recipes of all standard VLT/VLTI instrument pipelines. With *EsoRex* in your path, the general structure of an *EsoRex* command line is

```
1> esorex [esorex options] [recipe [recipe options] [sof [sof]...]]
```

where options appearing before the recipe name are options for *EsoRex* itself, and options given after the recipe name are options which affect the recipe.

All available *EsoRex* options can be listed with the command

```
1> esorex --help
```

and the full list of available parameters of a specific recipe can be obtained with the command

```
1> esorex --help <recipe name>
```

The output of this command shows as parameter values the current setting, i.e. all modifications from a configuration file or the command line are already applied.

The listing of all recipes known to *EsoRex* can be obtained with the command

```
1> esorex --recipes
```

The last arguments of an *EsoRex* command are the so-called *set-of-frames*. A *set-of-frames* is a simple text file which contains a list of input data files for the recipe. Each input file is followed by an unique identifier (frame classification or frame tag), indicating the contents of this file. The input files have to be given as an absolute path, however *EsoRex* allows the use of environment variables so that a common directory prefix can be abreviated. Individual lines may be commented out by putting the hash character (#) in the first column. An example of a *set-of-frames* is shown in the following:

```
1> cat bias.sof
$RAW_DATA/ERIS.2022-03-29T09:48:53.153.fits DARK
$RAW_DATA/ERIS.2022-03-29T09:50:36.645.fits DARK
$RAW_DATA/ERIS.2022-03-29T09:52:16.513.fits DARK
$RAW_DATA/ERIS.2022-03-29T09:53:47.996.fits DARK
#$RAW_DATA/ERIS.2022-03-29T09:55:04.515.fits DARK
```

These *set-of-frames* files will have to be created by the user using a text editor, for instance. Which classification has to be used with which recipe will be shown in section 7.4

Finally, if more than one *set-of-frames* is given on the command-line *EsoRex* concatenates them into a single *set-of-frames*.

7.3 Useful FITS keywords with IFU data

Keyword name	Description
OBJECT	target name
DPR.TYPE	type of exposure
DPR.CATG	category of exposure (CALIB for calibrations, science for observations)
OBS.NAME	OB name
OBS.ID	OB number
	Instrument and detector configuration (INS1: calib. unit, INS2: NIX, INS3: SPIFFIER
INS3.SPFW.ID	IFS filter wheel
INS3.SPGW.ID	IFS grating wheel
INS3.SPXW.ID	IFS pixel scale
DET.SEQ1.DIT	DIT (Detector Integration Time)
DET.NDIT	number of DITs averaged
OCS.OFFSET.RA	relative RA offset from last position
OCS.OFFSET.DEC	relative DEC offset from last position
OCS.CUMOFFS.RA	RA offset w.r.t initial pointing
OCS.CUMOFFS.DEC	DEC offset w.r.t initial pointing
	Calibration lamps (only present if lamp is ON)
INS1.LAMP1.ID	LDLS: for distortion (NS)
INS1.LAMP2.ID	QTH: for flatfield (NS & FLAT)
INS1.LAMP3.ID	ARGO: argon, for wave-cal (NS & WAVE)
INS1.LAMP4.ID	KRIP: krypton, for wave-cal (NS & WAVE)

continued on next page

continued from previous page	
Keyword name	**Description**
INS1.LAMP5.ID	NEON: neon, for wave-cal (NS)
	Seeing & (AO)
TEL.AMBI.FWHM.START	DIMM seeing at start (zenith, 500nm)
TEL.AMBI.FWHM.END	DIMM seeing at end (zenith, 500nm)
TEL.IA.FWHMLIN	Seeing from tel SH spot (zenith, 500nm)
TEL.IA.FWHMLINOBS	Seeing from tel SH spot (zenith, 500nm)
AOS.LGS.ASM.SEEING	Seeing from SPARTA (los, in band TBC)
AOS.LGS.ASM.WAVELENGTH	wavelength for seeing
AOS.LGS.WFS.FLUX	flux on LGS WFS
AOS.LO.WFS.FLUX	flux on NGS WFS
	Others
ADA.ABSROT.START	rotator angle at start
ADA.ABSROT.END	rotator angle at end
TEL.ALT	telescope altitude

7.4 Data Organization

The ERIS-SPIFFIER pipeline recipes accept the following input tags and generates frames with the following output tags:

- Recipe eris_ifu_dark

```
Input files:

  DO category:                Explanation:                Required:
  --------------------------  ----------                  ---------
  DARK                        Data                        YES

Output files:

  DO category:                Explanation:
  --------------------------  ----------
  MASTER_DARK_IFU             master dark
  BPM_DARK                    hot pixel map
```

- Recipe eris_ifu_detlin

```
Input files:

  DO category:                Explanation:                Required:
  --------------------------  ----------                  ---------
  LINEARITY_LAMP              Data                        YES
```

```
Output files:

  DO category:                Explanation:
  --------------------------  -----------
  GAIN_INFO                   detector gain info table
  BPM_DETLIN                  non-linear bad pixel map
  BPM_DETLIN_FILT             post-filtered BPM_DETLIN
```

- Recipe eris_ifu_distortion

```
Input files:

  DO category:                Explanation:                  Required:
  --------------------------  -----------                   ---------
  DARK_NS                     raw data (dark)               YES
  FIBRE_NS                    raw data (ifu slices)         YES
  FLAT_NS                     raw data (flat)               YES
  WAVE_NS                     raw data (arcs)               YES
  FIRST_WAVE_FIT              static (guess sol, wave, pos) YES
  REF_LINE_ARC                static (ref arc lines)        YES
  WAVE_SETUP                  static (guess sol, range fit) YES
Output files:

  DO category:                Explanation:
  --------------------------  -----------
  DISTORTION                  table with distortion correction coefficient
  BPM_DIST                    bad pixel map
  SLITLET_POS                 slitlets start/end positions
```

- Recipe eris_ifu_flat

```
Input files:

  DO category:                Explanation:                  Required:
  --------------------------  -----------                   ---------
  FLAT_LAMP                   raw data (flat)               YES
  BPM_DARK                    chain    (eris_ifu_dark)      YES
  BPM_DETLIN                  chain    (eris_ifu_detlin)    RECOMMENDED
  BPM_DIST                    chain    (eris_ifu_distortion) RECOMMENDED
Output files:

  DO category:                Explanation:
  --------------------------  -----------
  BPM_FLAT                    bad(dead) pixel map
  MASTER_FLAT                 master flat
```

- Recipe eris_ifu_wavecal

```
Input files:

  DO category:                Explanation:                 Required:
  --------------------------  -----------                  ---------
  WAVE_LAMP                   raw data (arc on)            YES
  WAVE_LAMP                   raw data (arc off)           YES
  DISTORTION                  chain    (eris_ifu_distortion) YES
  MASTER_FLAT                 chain    (eris_ifu_mflat)    RECOMMENDED
  FIRST_WAVE_FIT              static (guess sol, wave, pos) YES
  REF_LINE_ARC                static (ref arc lines)       YES
  WAVE_SETUP                  static (guess sol, range fit) YES
Output files:

  DO category:                Explanation:
  --------------------------  -----------
  WAVE_LAMP_STACKED           stacked arc lamp frame
  WAVE_LAMP_RESAMPLED         resampled arc lamp frame (qc)
  WAVE_MAP                    wavelength map
```

- Recipe eris_ifu_stdstar. This recipe support three cases that are different for DPR.TYPE value and corresponding input frame tag, to easy operations. The standard stars can be photometric standards (DPR.TYPE=STD,FLUX) used to determine the instrument efficiency and response on a limited set of stars observed in photometric nights; PSF standards (DPR.TYPE=PSF-CALIBRATOR), observed to compute the Strehl; other spectrophotometric stars and telluric standards (DPR.TYPE=STD). The corresponsing set of frames differ on the raw frame tag and some static input frames, and are listed below:
  - with photometric std star as input:

```
Input files:

  DO category:                Explanation:                 Required:
  --------------------------  -----------                  ---------
  STD_FLUX                    raw data (flux std)          YES
  SKY_STD_FLUX                raw data (sky for flux std)  YES
  DISTORTION                  chain    (eris_ifu_distortion) YES
  SLITLET_POS                 chain    (eris_ifu_distortion) NO
  MASTER_FLAT                 chain    (eris_ifu_mflat)    YES
  BPM_FLAT                    bad       pixel map          YES
  WAVE_MAP                    wavelength map               YES
  EXTCOEFF_TABLE              static   (atm ext table)      YES
  OH_SPEC                     table with OH lines to get   YES
                              OH based wavelength solution
  RESPONSE_WINDOWS            static windows to compute resp. YES
  RESP_FIT_POINTS_CATALOG     static points to fit response  YES
  TELL_MOD_CATALOG            static telluric model catalog  YES
  FIT_AREAS                   static fit areas               YES
  QUALITY_AREAS               static quality areas           YES
  FLUX_STD_CATALOG            static flux std catalog        YES
  EFFICIENCY_WINDOWS          static efficiency windows      YES
```

```
Output files:

  DO category:                Explanation:
  --------------------------  -----------
  SKY_CUBE                    sky obs with std star data cube
  STD_FLUX_CUBE               flux std star data cube
  STD_FLUX_CUBE_COADD         flux std star data cube resampled
  STD_CUBE_MEAN               mean of STD_FLUX_CUBE_COADD
  STD_FLUX_CUBE_MEDIAN        median of STD_FLUX_CUBE_COADD
  EXTRACTION_MASK             mask used for extraction
  SPECTRUM                    extracted spectrum (from STD_FLUX_CUBE_COADD)
  RESPONSE                    instrument response
  SPECTRUM_FLUXCAL            flux calibrated response
  STD_FLUX_CUBE_COADD_FLUXCAL flux calibrated STD_FLUX_CUBE_COADD
  EFFICIENCY                  instrument efficiency
```

– with PSF std star input:

```
Input files:

  DO category:                Explanation:                  Required:
  --------------------------  -----------                   ---------
  PSF_CALIBRATOR              raw data (PSF std)            YES
  SKY_PSF_CALIBRATOR          raw data (sky for PSF  std)   YES
  DISTORTION                  chain    (eris_ifu_distortion) YES
  SLITLET_POS                 chain    (eris_ifu_distortion) NO
  MASTER_FLAT                 chain    (eris_ifu_mflat)     YES
  BPM_FLAT                    bad       pixel map           YES
  WAVE_MAP                    wavelength map                YES
  EXTCOEFF_TABLE              static   (atm ext table)       YES
  OH_SPEC                     table with OH lines to get    YES
                              OH based wavelength solution
  RESPONSE                    instrument response           OPTIONAL (*)

Output files:

  DO category:                Explanation:
  --------------------------  -----------
  SKY_CUBE                    sky obs with std star data cube
  PSF_CUBE                    PSF std star data cube
  PSF_CUBE_COADD              PSF std star data cube resampled
  PSF_CUBE_MEAN               mean of PSF_CUBE_COADD
  PSF_CUBE_MEDIAN             median of PSF_CUBE_COADD
  EXTRACTION_MASK             mask used for extraction
  SPECTRUM                    extracted spectrum (from PSF_CUBE_COADD)
  SPECTRUM_FLUXCAL            flux calibrated response     (**)
  PSF_CUBE_COADD_FLUXCAL      flux calibrated PSF_CUBE_COADD (**)
```

(*) Due to presence of tellurics, to obtain better flux calibrated results the user should input the RESPONSE obtained in the "_low" resolution setting. If the raw data are of a high resolution setting the pipeline will automatically trim the response to the correct wavelength range of the given high resolution band.

(**) If RESPONSE is provided.

– with normal std star input:

```
Input files:

  DO category:                Explanation:                  Required:
  --------------------------  ----------                    ---------
  STD                         raw data (normal std)         YES
  SKY_STD                     raw data (sky for normal std) YES
  DISTORTION                  chain    (eris_ifu_distortion) YES
  SLITLET_POS                 chain    (eris_ifu_distortion) NO
  MASTER_FLAT                 chain    (eris_ifu_mflat)     YES
  BPM_FLAT                    bad       pixel map           YES
  WAVE_MAP                    wavelength map                YES
  EXTCOEFF_TABLE              static   (atm ext table)      YES
  OH_SPEC                     table with OH lines to get    YES
                              OH based wavelength solution
  RESPONSE                    instrument response           OPTIONAL (*)

Output files:

  DO category:                Explanation:
  --------------------------  ----------
  SKY_CUBE                    sky obs with std star data cube
  STD_CUBE                    std star data cube
  STD_CUBE_COADD              std star data cube resampled
  STD_CUBE_MEAN               mean of STD_CUBE_COADD
  STD_FLUX_CUBE_MEDIAN        median of STD_CUBE_COADD
  EXTRACTION_MASK             mask used for extraction
  SPECTRUM                    extracted spectrum (from STD_CUBE_COADD)
  SPECTRUM_FLUXCAL            flux calibrated response (**)
  STD_CUBE_COADD_FLUXCAL      flux calibrated STD_CUBE_COADD (**)
```

(*) Due to presence of tellurics, to obtain better flux calibrated results the user should input the RESPONSE obtained in the "_low" resolution setting. If the raw data are of a high resolution setting the pipeline will automatically trim the response to the correct wavelength range of the given high resolution band.

(**) If RESPONSE is provided

- Recipe eris_ifu_jitter (science object input)

```
Input files:

  DO category:                Explanation:                  Required:
  --------------------------  ----------                    ---------
  OBJ                         raw data (object)             YES
  SKY_OBJ                     raw data (sky for object)     YES
  DISTORTION                  chain    (eris_ifu_distortion) YES
  SLITLET_POS                 chain    (eris_ifu_distortion) NO
  MASTER_FLAT                 chain    (eris_ifu_mflat)     YES
  BPM_FLAT                    bad       pixel map           YES
  WAVE_MAP                    wavelength map                YES
  EXTCOEFF_TABLE              static   (atm ext table)      YES
  OH_SPEC                     table with OH lines to get    YES
```

```
                                OH based wavelength solution
  RESPONSE                      instrument response            OPTIONAL (*)
Output files:


  DO category:                  Explanation:
  --------------------------    -----------
  SKY_CUBE                      sky obs with object data cube
  OBJ_CUBE                      std star data cube
  OBJ_CUBE_COADD                std star data cube resampled
  OBJ_CUBE_MEAN                 mean of OBJ_CUBE_COADD
  OBJ_CUBE_MEDIAN               median of OBJ_CUBE_COADD
  EXTRACTION_MASK               mask used for extraction
  SPECTRUM                      extracted spectrum (from OBJ_CUBE_COADD)
  SPECTRUM_FLUXCAL              flux calibrated response (**)
  OBJ_CUBE_COADD_FLUXCAL        flux calibrated OBJ_CUBE_COADD (**)
```

(*) Due to presence of tellurics, to obtain better flux calibrated results the user should input the RESPONSE obtained in the "_low" resolution setting. If the raw data are of a high resolution setting the pipeline will automatically trim the response to the correct wavelength range of the given high resolution band.

(**) If RESPONSE is provided

7.5 Example of data reduction using EsoRex

A simple, typical data reduction procedure is described here.

We suggest the user to organize his data per type, observed band and camera setting. Dark frames may be grouped per detector DIT, frames to compute distortion and frames to compute detector non linearities may be organized per observed band. The detector DIT is given by the value of the FITS keyword DET SEQ1 DIT[1]. The observed band is indicated by the value of the FITS keyword INS3 SPGW ID. The pre-optic setting is indicated by the value of INS3 SPXW ID. In the examples below we suppose the user has data acquired in band K_low and with the 100 mas pre-optic setting, and DIT=600. In the following examples `/path_raw/` indicates the full path to the source tree directory containing raw data, `/path_ref/` indicates the full path to the source tree directory containing reference static calibration data, `/path_cdb/` indicates the full path to the source tree directory containing master calibration data. All calibration frames have DPR.CATG='CALIB'. Science observations have DPR.CATG='SCIENCE'.

Dark Frames: those calibration frames are characterized by DPR.TYPE='DARK',

```
/path_raw/DARK/600/ERIS.2022-08-23T09:36:12.316.fits DARK
/path_raw/DARK/600/ERIS.2022-08-23T09:51:59.824.fits DARK
/path_raw/DARK/600/ERIS.2022-08-23T10:07:40.760.fits DARK
```

[1] We omit here the prefix HIERARCH ESO

Detector linearity flat field frames: those calibration frames are caracterized by DPR.TYPE='LINEARITY,LAMP'

```
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:09:50.882.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:10:07.455.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:10:23.047.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:10:38.240.fits LINEARITY_LAMP
...
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:10:56.403.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:11:31.128.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:11:59.183.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:12:27.247.fits LINEARITY_LAMP
```

Fibre frames, flat frames and arc lamp frames to compute distortions: those calibration frames have DPR.TYPE respectively equal to 'NS,DARK', 'NS,SLIT', 'NS,FLAT,DARK' 'NS,FLAT,FLAT' (usually 2 frames, one with calibration flat lamp switched off, one with the lamp switched on), 'NS,WAVE,DARK', 'NS,WAVE,LAMP' .

```
/path_raw/DISTORTION/K_low/ERIS.2020-03-14T11:46:25.168.fits DARK_NS
/path_raw/DISTORTION/K_low/ERIS.2020-03-14T11:46:53.132.fits FIBRE_NS
/path_raw/DISTORTION/K_low/ERIS.2020-03-14T11:53:22.863.fits FLAT_NS
/path_raw/DISTORTION/K_low/ERIS.2020-03-14T11:53:45.397.fits FLAT_NS
/path_raw/DISTORTION/K_low/ERIS.2020-03-14T11:54:08.861.fits WAVE_NS
/path_raw/DISTORTION/K_low/ERIS.2020-03-14T11:55:10.220.fits WAVE_NS
```

Standard flat field frames: those calibration frames are caracterized by DPR.TYPE='FLAT,DARK' and DPR.TYPE='FLAT,LAMP'

```
/path_raw/FLAT/K_low/100/ERIS.2020-02-28T16:27:43.232.fits FLAT_LAMP
/path_raw/FLAT/K_low/100/ERIS.2020-02-28T16:28:05.846.fits FLAT_LAMP
...
/path_raw/FLAT/K_low/100/ERIS.2020-02-28T16:28:45.566.fits FLAT_LAMP
/path_raw/FLAT/K_low/100/ERIS.2020-02-28T16:29:08.165.fits FLAT_LAMP
```

Arc lamp frames: those calibration frames are caracterized by DPR.TYPE='WAVE,DARK' and DPR.TYPE='WAVE,LAMP'

```
/path_raw/WAVE/K_low/100/ERIS.2022-02-28T17:55:44.753.fits WAVE_LAMP
/path_raw/WAVE/K_low/100/ERIS.2022-02-28T18:00:15.875.fits WAVE_LAMP
```

To determine the instrument efficiency and response the user may want to reduce also flux standard star frames: those calibration frames are caracterized by DPR.TYPE='STD,FLUX' or DPR.TYPE='SKY,STD,FLUX'

```
/path_raw/STD/K_low/100/ERIS.2022-02-27T09:17:05.775.fits STD_FLUX
/path_raw/STD/K_low/100/ERIS.2022-02-27T09:18:10.566.fits SKY_STD_FLUX
```

Strehl is computed on PSF standard star frames: those frames are caracterized by `DPR.TYPE='PSF-CALIBRATOR'` or `DPR.TYPE='SKY,PSF-CALIBRATOR'`

```
/path_raw/PSF/K_low/100/ERIS.2022-08-23T07:35:16.597.fits PSF_CALIBRATOR
/path_raw/PSF/K_low/100/ERIS.2022-08-23T07:36:55.413.fits SKY_PSF_CALIBRATOR
```

Additional standard frames are observed. Those frames are caracterized by DPR.TYPE='STD' or DPR.TYPE='SKY,STD'

```
/path_raw/STD/K_low/100/ERIS.2022-02-27T09:17:05.775.fits STD
/path_raw/STD/K_low/100/ERIS.2022-02-27T09:18:10.566.fits SKY_STD
```

science frames: those frames are caracterized by DPR.CATG='SCIENCE', DPR.TYPE='OBJECT' or DPR.TYPE='SKY'

```
/path_raw/SCI/K_low/100/ERIS.2022-02-27T04:30:22.175.fits OBJ
/path_raw/SCI/K_low/100/ERIS.2022-02-27T04:30:54.620.fits SKY_OBJ
```

We describe below a typical data reduction sequence using EsoRex. In this section we assume that the user sets in the EsoRex configuration file ($HOME/.esorex/esorex.rc) the flag *suppress-prefix* to TRUE, so that the pipeline product file names have standard names, with extention .fits for images and tables. We suggest to verify to have the flag *readonly* set to FALSE, if the user would like to run the same recipe several times with EsoRex having standard names for product files. This setting allows the pipeline to overwrite previously generated products [2]. In the following we indicate only those frames involved in the data reduction cascade, suggesting the user to rename them according to their PRO.CATG, preoptic and band, and to remove the other products after each recipe execution.

1. The user may start to generate a master dark. Raw dark frames may be put in an ASCII file, dark_sof. This file will look like as follows:

   ```
   /path_raw/DARK/600/ERIS.2022-08-23T09:36:12.316.fits DARK
   /path_raw/DARK/600/ERIS.2022-08-23T09:51:59.824.fits DARK
   /path_raw/DARK/600/ERIS.2022-08-23T10:07:40.760.fits DARK
   ```

   Then the user can generate the master dark with the command

   esorex eris_ifu_dark dark_sof

   This command will generate two files: eris_ifu_dark_bpm.fits (PRO.CATG=BPM_DARK), a hot pixel map, and eris_ifu_dark_master_dark.fits (PRO.CATG=MASTER_DARK_IFU), a master dark. For convenience we indicate with `/path_cdb` the full path to a directory containing relevant data reduction products.

   mv eris_ifu_dark_bpm.fits /path_pro/BPM_DARK.fits

   mv eris_ifu_dark_master_dark.fits /path_pro/MASTER_DARK_IFU_600.fits

   rm -rf out*fits *.log

[2] By default installation in the EsoRex configuration file ($HOME/.esorex/esorex.rc) the flag *suppress-prefix* to FALSE and the flag *readonly* is set to FALSE, a possible combination, in which case pipeline product filenames will have a prefix out_, an increasing a four digit number, and extention .fits for images and tables.

2. Then the user may generate a map of non linear pixels. A set of linearity raw flat field frames may be put in the ASCII file linearity_sof.

```
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:09:50.882.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:10:07.455.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:10:23.047.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:10:38.240.fits LINEARITY_LAMP
...
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:10:56.403.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:11:31.128.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:11:59.183.fits LINEARITY_LAMP
/path_raw/LINEARITY/K_low/ERIS.2022-02-26T20:12:27.247.fits LINEARITY_LAMP
```

   The user can generate a non linearity bad pixel map (PRO.CATG=BPM_DETLIN) with the command:

   esorex eris_ifu_detlin linearity_sof

   This command will generate several files, including the non linearity bad pixel map, stored in the file “eris_ifu_detlin_bpm.fits ”.

   mv eris_ifu_detlin_bpm.fits /path_pro/BPM_DETLIN_K_low.fits

   rm -rf out*fits *.log

3. Then the user may determine the optical distortions (PRO.CATG=DISTORTION) and the slitlets positions (PRO.CATG=SLITLETS_POS).

   The user will select all the files containing the string DISTORTION in their DPR.TYPE. Using those files and the line reference table of the corresponding band (K) and a drs setup table will generate an ASCII file distortion_sof:

```
/path_raw/ERIS.2022-03-14T11:28:19.007.fits DARK_NS
/path_raw/ERIS.2022-03-14T11:28:43.781.fits FIBRE_NS
/path_raw/ERIS.2022-03-14T11:53:22.863.fits FLAT_NS
/path_raw/ERIS.2022-03-14T11:53:45.397.fits FLAT_NS
/path_raw/ERIS.2022-03-14T11:54:08.861.fits WAVE_NS
/path_raw/ERIS.2022-03-14T11:55:10.220.fits WAVE_NS
/path_cdb/eris_ifu_first_fit.fits FIRST_WAVE_FIT
/path_cdb/eris_ifu_ref_lines.fits REF_LINE_ARC
/path_cdb/eris_ifu_wave_setup.fits    WAVE_SETUP
```

   To successfully run the distortion recipe are needed: a NS,DARK (DARK_NS) and a NS,SLIT (FIBRE_NS) frame; two FLAT_NS frames, one with calibration flat lamp switched on, one with the lamp switched off; two WAVE_LAMP frames, one with calibration flat lamp switched on, one with the lamp switched off; and the three static frames, the reference line list of the appropriate band, with tag REF_LINE_ARC, a static table, with tag FIRST_WAVE_FIT, that contains foreach instrument setting, first guess solutions for wavelengths and their pixel positions of a few reference lines, and the static table, with tag WAVE_SETUP, that contains the range value to be used in the first fit of the wavelength calibration solution, where with

`/path_cdb` we have indicated the full path to the directory containing calibration data (see also Sec. 10).

The command

esorex eris_ifu_distortion distortion_sof

Will generate several products. Between those also eris_ifu_distortion_distortion.fits, a table containing information on the polynomial distortion coefficients, and eris_ifu_distortion_slitlet_pos.fits, a table containing information on the slitlet edges positions.

mv eris_ifu_distortion_distortion.fits /path_pro/DISTORTION_K_low.fits

rm -rf out*fits *.log

4. Then one selects the raw flat fields and list them in an ASCII file flat_sof together with some static calibrations and previously obtained products:

```
/path_raw/FLAT/ERIS.2022-02-28T16:27:43.232.fits FLAT_LAMP
/path_raw/FLAT/ERIS.2022-02-28T16:28:05.846.fits FLAT_LAMP
/path_raw/FLAT/ERIS.2022-02-28T16:28:18.820.fits FLAT_LAMP
/path_raw/FLAT/ERIS.2022-02-28T16:28:32.593.fits FLAT_LAMP
/path_raw/FLAT/ERIS.2022-02-28T16:28:45.566.fits FLAT_LAMP
/path_raw/FLAT/ERIS.2022-02-28T16:29:08.165.fits FLAT_LAMP
/path_pro/BPM_DARK.fits BPM_DARK
/path_pro/BPM_DIST_K_low.fits BPM_DIST
/path_pro/BPM_DETLIN_K_low.fits BPM_DETLIN
```

The command:

esorex eris_ifu_flat flat_sof

generates several frames, including the master flat field, stored in the file “eris_ifu_flat_master_flat.fits”, and the master bad pixel map, stored in the file “eris_ifu_flat_bpm.fits”.

mv eris_ifu_flat_master_flat.fits /path_pro/MASTER_FLAT_K_100.fits

mv eris_ifu_flat_bpm.fits /path_pro/BPM_FLAT_K_100.fits

rm -rf out*fits *.log

5. Then one lists raw arc lamp frames and additional static calibration frames and previously obtained master calibrations in an ASCII file wcal_sof:

```
/path_raw/WAVE/ERIS.2022-02-28T17:55:44.753.fits WAVE_LAMP
/path_raw/WAVE/ERIS.2022-02-28T18:00:15.875.fits WAVE_LAMP
/path_pro/MASTER_FLAT_K_100.fits MASTER_FLAT
/path_pro/DISTORTION_K_100.fits DISTORTION
/path_cdb/eris_ifu_ref_lines.fits REF_LINE_ARC
/path_cdb/eris_ifu_wave_setup.fits WAVE_SETUP
/path_cdb/eris_ifu_first_fit.fits FIRST_WAVE_FIT
```

The command:

esorex eris_ifu_wavecal wcal_sof

generates several frames, including the master wavelength map, stored in the file “eris_ifu_wave_map.fits”.

mv eris_ifu_wave_map.fits /path_pro/WAVE_MAP_K_100.fits

rm -rf out*fits *.log

6. To create the instrument response the user will reduce the (“_low”) resolution grating) flux standard star data:

```
/path_raw/OBJNOD/ERIS.2022-02-26T06:09:19.358.fits STD_FLUX
/path_raw/OBJNOD/ERIS.2022-02-26T06:10:10.996.fits SKY_STD_FLUX
/path_pro/DISTORTION_K_100.fits DISTORTION
/path_pro/BPM_FLAT_K_100.fits BPM_FLAT
/path_pro/MASTER_FLAT_K_100.fits MASTER_FLAT
/path_pro/WAVE_MAP_K_100.fits WAVE_MAP
/path_ref/OH_SPEC.fits OH_SPEC
/path_ref/EXTCOEFF_TABLE.fits EXTCOEFF_TABLE
/path_ref/EFFICIENCY_WINDOWS.fits EFFICIENCY_WINDOWS
/path_ref/FIT_AREAS.fits FIT_AREAS
/path_ref/QUALITY_AREAS_K_low.fits QUALITY_AREAS
/path_ref/RESPONSE_WINDOWS_K_low.fits RESPONSE_WINDOWS
/path_ref/RESP_FIT_POINTS_CATALOG.fits RESP_FIT_POINTS_CATALOG
/path_ref/TELL_MOD_CATALOG.fits TELL_MOD_CATALOG
```

The command:

esorex eris_ifu_stdstar stdstar_sof

generates several frames, including the instrument RESPONSE frame.

mv eris_ifu_stdstar_response.fits /path_pro/RESPONSE_K_100.fits

rm -rf out*fits *.log

7. Finally the user will reduce the science data:

```
/path_raw/OBJNOD/ERIS.2022-02-27T06:09:19.358.fits OBJ
/path_raw/OBJNOD/ERIS.2022-02-27T06:10:10.996.fits SKY_OBJ
/path_raw/OBJNOD/ERIS.2022-02-27T06:11:33.949.fits OBJ
/path_raw/OBJNOD/ERIS.2022-02-27T06:13:08.204.fits SKY_OBJ
/path_pro/DISTORTION_K_100.fits DISTORTION
/path_pro/BPM_FLAT_K_100.fits BPM_FLAT
/path_pro/MASTER_FLAT_K_100.fits MASTER_FLAT
/path_pro/WAVE_MAP_K_100.fits WAVE_MAP
/path_pro/RESPONSE_K_100.fits RESPONSE
/path_ref/OH_SPEC.fits OH_SPEC
/path_ref/EXTCOEFF_TABLE.fits EXTCOEFF_TABLE
```

The command:

esorex eris_ifu_jitter sci_sof

generates several frames, including a reconstructed cube of each target observation, stored in the file “eris_ifu_jitter_obj_cube_00#.fits” (#=0,1,..N), sky frames, stored in the file “eris_ifu_jitter_sky_cube_00#.fits” (#=0,1,..N), a mosaic cube coadding each single cube observation, stored in the file “eris_ifu_jitter_obj_cube_coadd.fits” its median and mean images, stored in the file “eris_ifu_jitter_obj_cube_median.fits” and “eris_ifu_jitter_obj_cube_mean.fits”.

8 Known Issues

8.1 Use ESO Reflex

We recommend to reduce the data with ESOReflex. This release provides a workflow for ERIS-SPIFFIER data reduction (`eris_spiffier.xml`). The reconstructed cube resampling is RAM intensive, and requires some minutes. We could reduce most of commissioning and Science Verification data with 20 GB of RAM but we recommend users to have at least 32 GB of RAM. In a few cases, if observations imply large offsets over the FOV, the user may require more RAM.

8.2 Combining Multiple Cubes (possibly observed with different OBs)

Users may have observations of the same object acquired by different OBs or template start/end sequences. They may want to combine them. Alternatively users may want not to combine some cubes but only the others into the final 3D cube mosaic. This release provides a recipe, `eris_ifu_combine_hdrl`, that allows to do this. The user has to do the following steps:

1. Run the `eris_ifu_jitter` recipe to reconstruct the individual cubes. The user may set the parameter cube.combine=FALSE to skip the combine step.
2. Prepare the comb.sof, that may look like the following:

   ```
   eris_ifu_jitter_obj_cube_000.fits OBJECT_CUBE
   eris_ifu_jitter_obj_cube_001.fits OBJECT_CUBE
   eris_ifu_jitter_obj_cube_002.fits OBJECT_CUBE
   eris_ifu_jitter_obj_cube_003.fits OBJECT_CUBE
   response.fits RESPONSE
   atm_extinction.fits EXTCOEFF_TABLE
   ```

   Run the `eris_ifu_combine_hdrl` recipe:
   `esorex eris_ifu_combine_hdrl comb.sof`
3. By default, the WCS keywords in each object cube will be used to resample the data onto the same grid. In case the WCS keywords do not align the different cubes well, users can obtain a proper alignment by specifying additional information via an ASCII file (default name offset.list) as follows:

   (a) If `offset_unit` is set to “PIXEL”: The user should provide the offsets (usually a fraction of a pixel) in pixel units to be applied to the values of CRPIX1,2 to correct each cube alignment. The pipeline implements the correction to each cube i as follows:

   ```
   crpix1_corr[i] = crpix1_orig[i]+offsetx[i]
   crpix2_corr[i] = crpix2_orig[i]+offsety[i]
   ```

   This means that if the CRPIX1,2 of the cubes are overestimated the user should apply a correction of opposite sign via the input offsets.
   For example a possible input in pixel unit may look for four data cubes as:

```
-0.2061   0.2620
 0.2536 -0.2629
-0.1783   0.3088
-0.3206   0.4228
```

and run the `eris_ifu_combine_hdrl` recipe as:

```
esorex eris_ifu_combine_hdrl --offset_mode=FALSE--offset_unit=PIXEL  --name_i=offset.list comb.sof
```

(b) If `offset_unit` is set to "DEGREE": The user should provide the offsets (usually equivalent to a fraction of a pixel) in units of decimal degrees, to be applied to the values of CRVAL1,2 to correct each cube alignment. The pipeline implements the correction to each cube i as follows:

```
crval1_corr[i] = crval1_orig[i]+offset_alpha[i]
crval2_corr[i] = crval2_orig[i]+offset_delta[i]
```

This means that if the CRVAL1,2 of the cubes are overestimated the user should apply a correction of opposite sign via the input offsets.

For example a possible input in unit of degres may look for four data cubes as:

```
-0.0000000   0.0000000
 0.0000316 -0.0000373
-0.0000316 +0.0000341
-0.0000290 +0.0000229
```

and run the `eris_ifu_combine_hdrl` recipe as:

```
esorex eris_ifu_combine_hdrl --offset_mode=FALSE--offset_unit=DEGREE  --name_i=offset.list comb.sof
```

(c) If `offset_unit` is set to "ARCSEC": The correction is applied similarly to previous "DEGREE" case, but the user will provide correction in arcsec units and thus the pipeline will apply a factor 1./3600. to get the correct shift value in degree units.

(d) If `offset_unit` is set to "PIXDEG": This method should lead to the most accurate result. In this case the user can measure the centroid position (in pixel unit) of the same reference object contained in each data cube to be aligned using a tool, e.g. sextractor, ds9, gaia. Then define a RA, DEC position for the object (e.g. from CDS, or GAIA or take the RA, DEC from the first frame) and assign the same value to all the frames. The recipe will resample the data to obtain a combined cube where the reference pixel (CRPIX1,2) and its RA, DEC values are set at the resampled position of the reconstructed data cube.

For example a possible input for four data cubes where the centroid position and the corresponding sky coordinates have been determined may look as:

```
30.7939 29.2620     227.6427899 -38.0137346
21.2536 28.7371     227.6427899 -38.0137346
30.8217 29.3088     227.6427899 -38.0137346
30.6794 38.4228     227.6427899 -38.0137346
```

and run the `eris_ifu_combine_hdrl` recipe as:

```
esorex eris_ifu_combine_hdrl --offset_mode=FALSE--offset_unit=PIXDEG  --name_i=offset.list comb.sof
```

In order to improve results we also recommend the user to apply the Differential Atmospheric Refraction correction before measuring the position of the object in each data cube.

As information we also note that in current release, if one combines data cubes obtained with different templates or observation blocks, and possibly acquired with different integration times, the recipe `eris_ifu_combine_hdrl` does not yet wiight intensity by exposure times. Users willing to take into account of that may use the recipe `eris_ifu_combine` instead.

This release includes also the recipe `eris_ifu_combine`. This combines cubes using an algorithm similar to what was implemented for the SINFONI pipeline. Differently from `eris_ifu_combine_hdrl`, this recipe does not resample each data cube before combining them, which results in a much faster cube combination even if the comination methods are less than the other. The `eris_ifu_combine` recipe provides the ability to apply an initial percentile clipping and a consecutive kappa-sigma-clipping (see parameters pclip, **ks_clip, kappa**). This may result in better 'quality on the combined data cube. For more details see Section 12.10). However the recipe `eris_ifu_combine` is know to generate sometimes a combined data cube with inaccurate WCS. We have not yet fixed that problem. In such cases we kindly ask the user to correct the WCS for example correcting the values of CRPIX1,2.

8.3 Correcting possible residual minor shifts of the observed object

The observed object raw frame may have a small shift along the cross-dispersion direction, typically within the range [-2,2] pixels, depending on telescope pointing position and the rotation angle of the instrument. This may affect the quality of the reconstructed frame, where the spectrum wraps around from one side of the cube to the other. To correct this effect, the user may use a dedicated parameter, `derot_corr`, and set it to the small shift. A positive number will shift the science cube to the right hand side, and the column(s) on the right edge will wrap to the left edge; a negative number will shift to the left hand side, and the left edge will be wrap to the right. With current release this correction is automatically done by the pipeline as the parameter `derot_corr` default is 'auto', that means the pipeline automatically estimates (from a model based on sampled data) the correct value of this parameter. If the user is not happy changing `derot_corr` parameter value overwrites the automatic setting.

8.4 Combining Multiple Cubes

During instrument commissioning we found rare cases where objects with very different WCS were observed in the same template sequence. This may lead to a combined cube of large size that may require more RAM of what the users has. In order to prevent the process is killed by the Linux Kernel we decided to define a parameter `-max-cubes-centres-dist` that sets the maximum allowed distance between the reconstructed cubes, for which a combined cube is created. Current default is set to 240 pixels. This corresponds to a maximm RAM usage of approximatively 20 GB. If the (estimated) computed maximum distance between reconstracted cubes is greater than that value, the pipeline does not generate the `OBJECT_CUBE_COADD` data product, nor the extracted spectrum (that is computed on the `OBJECT_CUBE_COADD`). If the user has enough RAM and would like to obtain the missing products, she/he may change the `max-cubes-centres-dist` parameter value.

To (slightly) reduce the cube resampling computation time the user may reduce the value of the parameter `method.loop-distance` to 1. Often this does not worsen the reconstructed cube quality (see Figure 8.1). The default `method.loop-distance=3` was set to optimise quality over a large data set even if this value implies a longer computation.

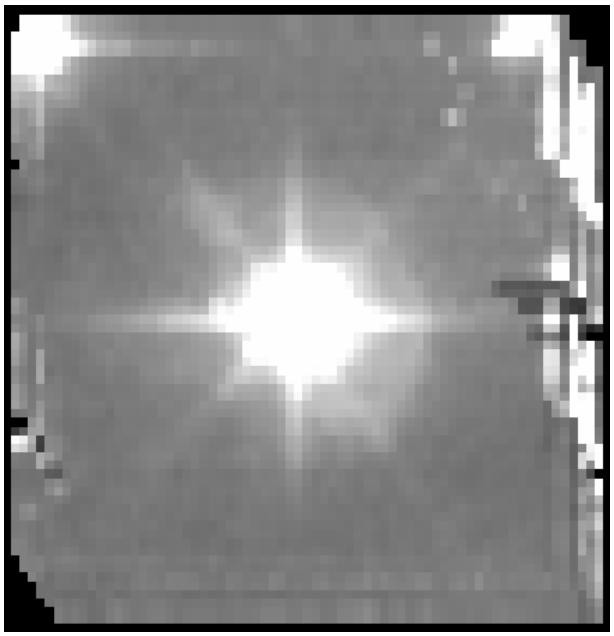


Figure 8.1: This figure displays results obtained for the same object (V V2423 Ori) observation. The cube quality is almost identical thus we show only one plane of a datacube obtained with parameter defaults. The quality is good. With and Intel(R) Core(TM) i7-8665U CPU @ 1.90GHz, the computations times are respectively of 104.22 s, 109.18 s, and 120.15 s. This means that the user may slightly improve computation time by reducing the value of the method.loop-distance. As the speed-up improvement is only of 15 %, we recommend to keep parameters defaults.

As the eris_ifu_jitter recipe spend a significant amount of data reduction time during the combination of the data cubes, that is an HDRL based parallelised data reduction step, it is obviously important to be able to run the recipe with a multi-core CPU. For example the reduction of an observation of RMC on a 100mas setting, that generates a combined cube of 94x94x2147 pixels, passed from 2:02.04 (mm:ss) on a simple Intel(R) Core(TM) i7-8665U CPU @ 1.90GHz with 4 physical cores (8 threads), to 1:22.65 (mm:ss) using a 12th Gen Intel(R) Core(TM) i9-12900K with 16 cores (8+8) and 24 threads (16+8).

If the user does not want to create the combined cube, one may set the parameter **cube.combine** to false. In this case also the extracted spectrum will not be created.

8.5 Handling cosmics in cube combination

As reported from some user the recipe `eris_ifu_jitter` may have problems handling cosmic ray hits. In those cases the user is recommended to obtain the contributing cubes with the recipe `eris_ifu_jitter`, and then combining them with the recipe `eris_ifu_combine`. A clear example is shown in the Figure 8.2 kindly provided by a user. The attached image shows on the top row left-to-right the combined cube from the recipe `eris_ifu_jitter`, the same results using the recipe `eris_ifu_combine`, and then two contributing cubes, and on the bottom row the other four contributing cubes. Clearly at the given wavelength, 1.2584 μm, we see that each cube has a cosmic near the left edge, in different positions for the different cubes. As the jitter recipe cannot handle it well, it results in the combined cube left edge like three spots of different intensity, wile the result of the recipe `eris_ifu_combine` is much cleaner.

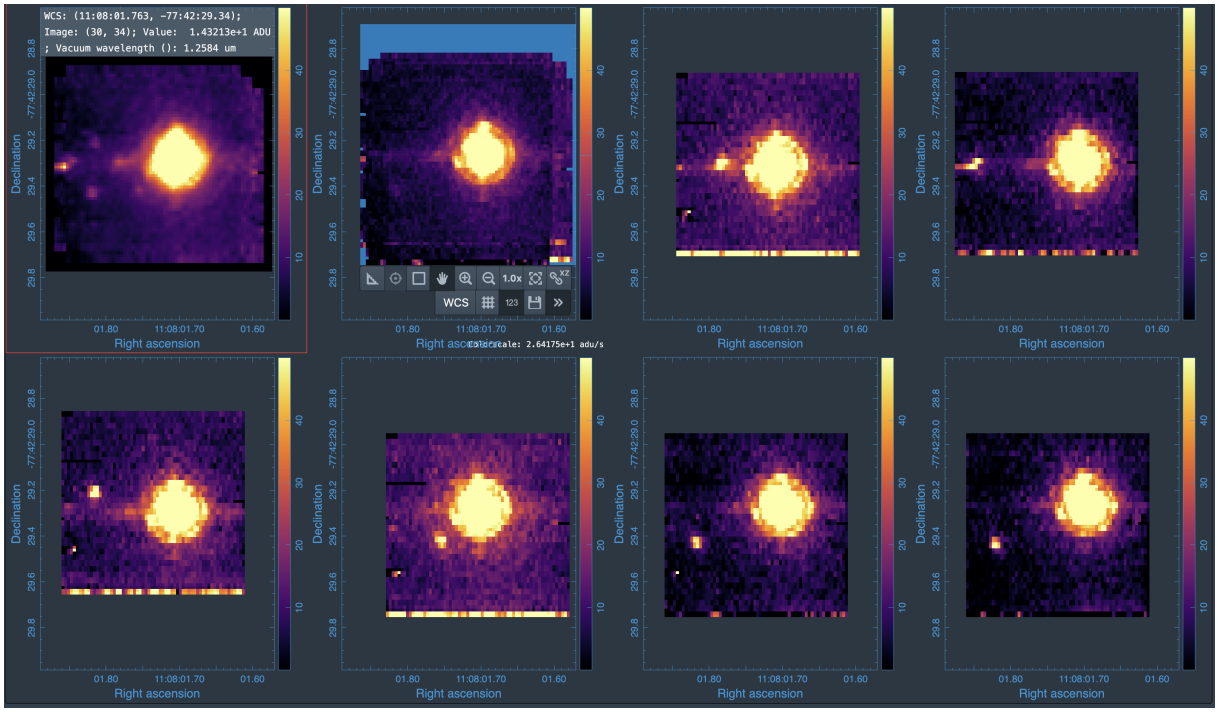


Figure 8.2: This figure shows on the top row left-to-right the combined cube from the recipe eris_ifu_jitter, the same results using the recipe eris_ifu_combine, and then two contributing cubes, and on the bottom row the other four contributing cubes. Clearly at the given wavelength, 1.2584 μm, we see that each cube has a cosmic near the left edge, in different positions for the different cubes. As the jitter recipe cannot handle it well, it results in the combined cube left edge like three spots of different intensity, wile the result of the recipe eris_ifu_combine is much cleaner.

8.6 Refining the Wavelength Calibration Using Sky Lines

The static frame with PRO.CATG OH_SPEC is now indicated as required input for the recipes `eris_ifu_stdstar` and `eris_ifu_jitter`. In the current pipeline release it is actually a recommended input. If the user does not input that frame the `eris_ifu_stdstar` and `eris_ifu_jitter` may still successfully reduce data, but their wavelength calibration may be less accurate, as it will be based only on the solution obtained from the

daily calibrations provided by the lamp wavelength flat. This calibration can be affected by instrumental flexures (SPIFFIER is mounted on the Cassegrain focus of UT4). By providing the OH_SPEC fits table the wavelength solution initially computed from the lamp is refined using [OH] lines in the spectrum.

8.7 Missing BPM_DARK in Reflex DataOrganiser

To simplify ERIS-SPIFFIER operations we associate as input BPM_DARK of the flat recipe one with a DIT of 10s and NDIT=1. This is a dark acquired every day for QC. The OCA rules of CalSelector have been updated accordingly. Should the user have data acquired before, it might be that the ESOReflex data set chooser is greyed-out, indicating *Missing* `BPM_DARK` data. To solve this problem the user has to add DARKS of DIT 10s and NDIT=1 to the input data.

8.8 Wiggles Visible in Under-Sampled Data Cubes

If IFU data is spatially under-sampled, ripples can appear in the extracted spectrum from the resampled cube. This is an unavoidable interpolation effect. A example of this is apparent in a standard star exposure taken in the 100mas scale with AO and in the K-middle passband (see figure 8.3). The standard star is spatially under-sampled and the majority of its flux falls in a single rectangular pixel. If the spectrum is extracted over a single pixel aperture the spectrum will have very apparent undulations. This is a known phenomena and has been described for KMOS in [**?**] (in particular, see their figures 12 and 13). However, if one integrates the flux over a reasonable aperture the wiggles go away, as the total flux is preserved (see figure 8.4).

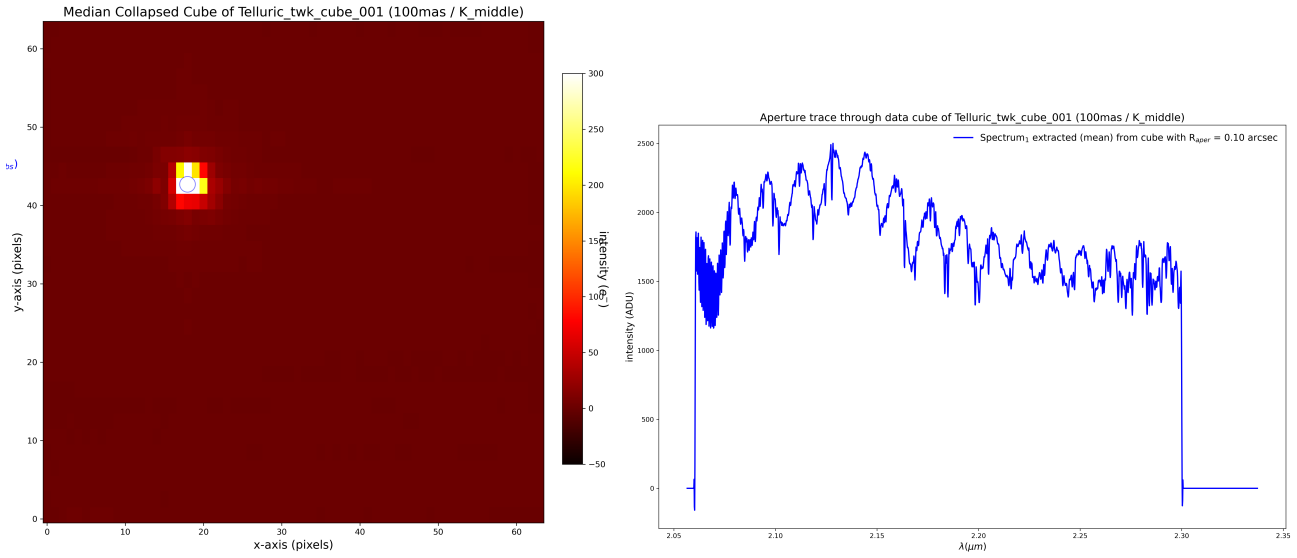


Figure 8.3: An example of extracted spectrum 'wiggles' in an under-sampled standard star. To emphasise the under-sampling, the median collapsed cube has been scaled to the image minimum and maximum flux levels (left panel). Clearly, the majority of the standard star's flux falls into a single rectangular pixel. When extracting through the data cube over an aperture close to this single pixel, the undulating spectrum becomes apparent (right panel).

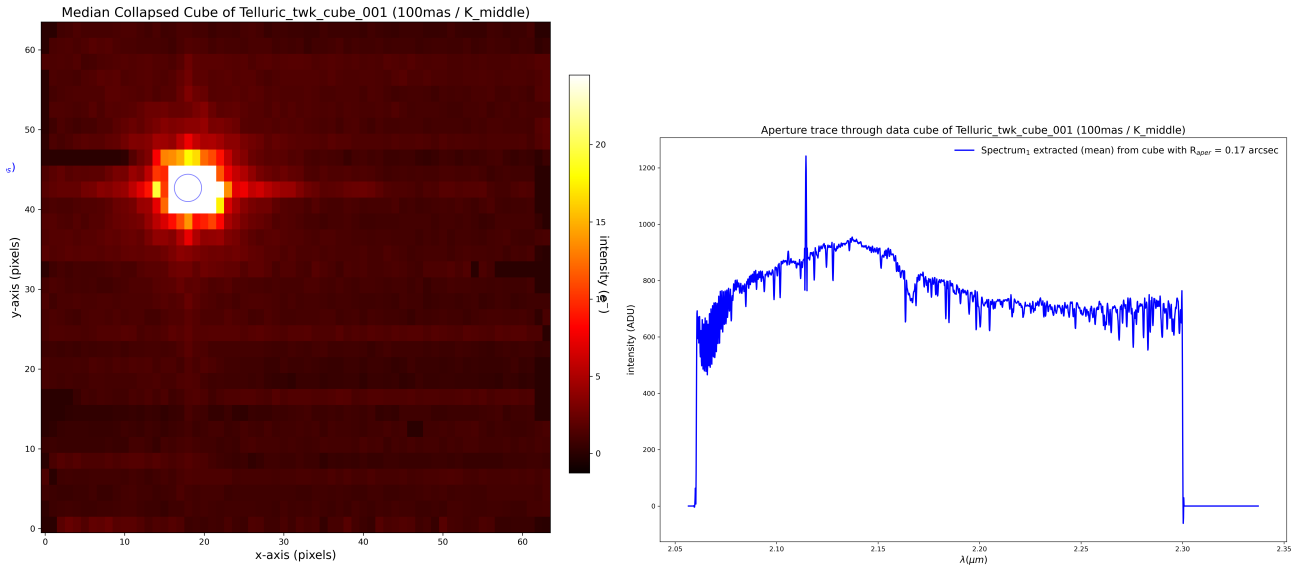


Figure 8.4: The identical under-sampled standard star of figure 8.3, but differently scaled (left panel). Clearly, the majority of the standard star's flux falls into a single rectangular pixel. When extracting through the data cube over a reasonable aperture the wiggles disappear (right panel).

8.9 Possible failure of recipe eris_ifu_jitter in case –crea-phase3=true and –sky_tweak=1

Occasionally the recipe `eris_ifu_jitter` may fail if –crea-phase3=true and –sky_tweak=1. This because the sky_tweak modifies the sky background level and this affects the computation of a FITS keyword added as part of the phase3 data format. We are looking into a possible fix. In the meantime the user has two options:

- Deactivate creation of phase3 data format products (which is used internally by ESO to add FITS keyword information required for the injestion in the phase3 database to add metadata used later for archive catalog frame classification/search/association). This solves the problem.
- If the user does not require to apply the Davies method for sky modeling and residual correction, an option is also to set sky_tweak to 0. This usually solves the problem.

8.10 Obtaining proper flux calibrated results

To obtain proper flux calibrated results the response must be accurate. As descibed in section 13.2.11 the response is proportional to the ratio between the flux of the observed standard star provided by a reference catalog and the total flux of the standard star obtained extracting the SPIFFIER data. It is important to be sure to have integrated all flux of the observed standard star. Thus the extraction **radius** must be large enough to include all flux of the star (usually four times the start PSF sigma is a good value), but not too large to introduce noise from the adjacent SKY. The flux std stars are all observed in 250 mas setting without AO. Usually for most of them a value of 10 to 12 pixels is appropriate, depending on the seeing observing conditions.

8.11 Determination of response and spectrum flux calibration based on photometric standard stars not part of the calibration plan

The ERIS-SPIFFIER pipeline allows to flux calibrate combined data cubes and the corresponding extracted spectra, if the user determines first the instrument response by processing a spectro-photometric standard. The spectro photometric STD stars observed by ERIS are listed in the static input catalog with PRO.CATG `FLUX_STD_CATALOG`. Those are a few, but the ERIS calibration plan ensures they are observed in photometric conditions and with proper frequency. We recommend users use those flux standards to flux calibrate their data.

Users willing to flux calibrate the data with another STD star have to determine the response with their own data reduction, in the same format as the one created by the pipeline (at least for the first exention with the smoothed response) and then use it as input of the jitter recipe. We warn that the user may have to request to observe those stars and those maybe observed not in photometric conditions.

It maybe possible to eventually determined the response automatically on another standard star as follows:

- The user needs to have the tabulated flux of the standard as observed by Space. This has to be in the same format as the fluxes contained in extensions 2-6 of the `FLUX_STD_CATALOG` table.
- Add an entry in the 1st table extention listing:

  - The 'ext_id' integer number where the tabulated flux standard spectrum will be stored
  - The 'name' of the standard star, as a string.
  - Its 'ra' (right ascension), as a double precision number.
  - Its 'dec' (declination), as a double precision number.

- Add the corresponding spectrum in the table extention corresponding to 'ext_id'. The wavelength information has to go in a column named 'LAMBDA', the flux in a column named 'FLUX, the bin width (usually 1.0 for unbinned data) in a column named 'BIN_WIDTH'. The reference standard star spectrum should be defined possibly by many sampling data points.

- Upgrade the table RESP_FIT_POINTS_CATALOG in a similar manner. In the extention corresponding to the new std star are indicated the wavelengths at which the "raw" response determined by the pipeline should be fit. Thise have to be stored in a double precision column named 'LAMBDA'. The values should be the wavelength values at which the sampling is done. Their position is refined by the standard star continuum, chosen no to fall in a region of telluric lines. As first approximation one may take the same values chosen for the other flux standard stars.

8.12 Possible failure of recipe eris_ifu_jitter in case –mask_method is set to 'fit'

The recipe eris_ifu_jitter may fail when –mask_method is set to 'fit' in case the image of a collapsed 3D data cube does not show an object with a good contrast with respect to the sky background. In this case the extraction with –mask_method set to 'fit' may fail as the 2D Gaussian fit does not converge. The user may eventually choose to set –mask_method to 'max' to get a spectrum, even if in such cases probably the extracted spectrum is not very meaningful. An example of such a case is shown in Figure 8.5.

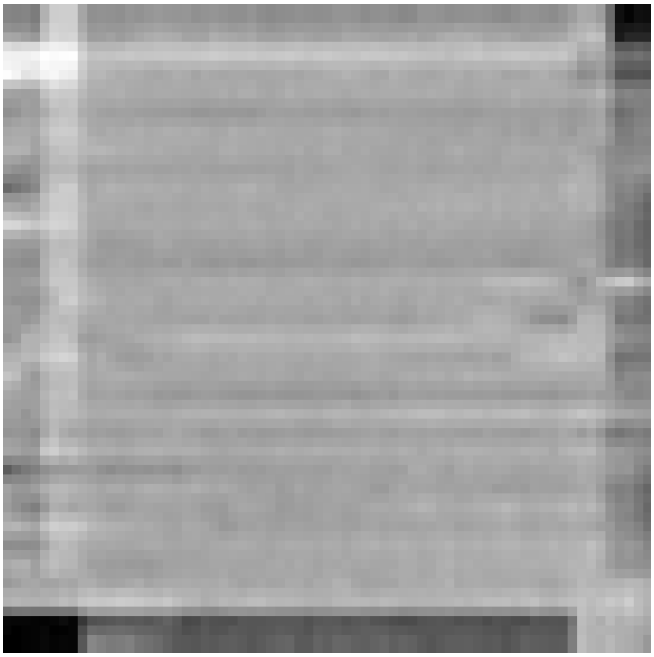
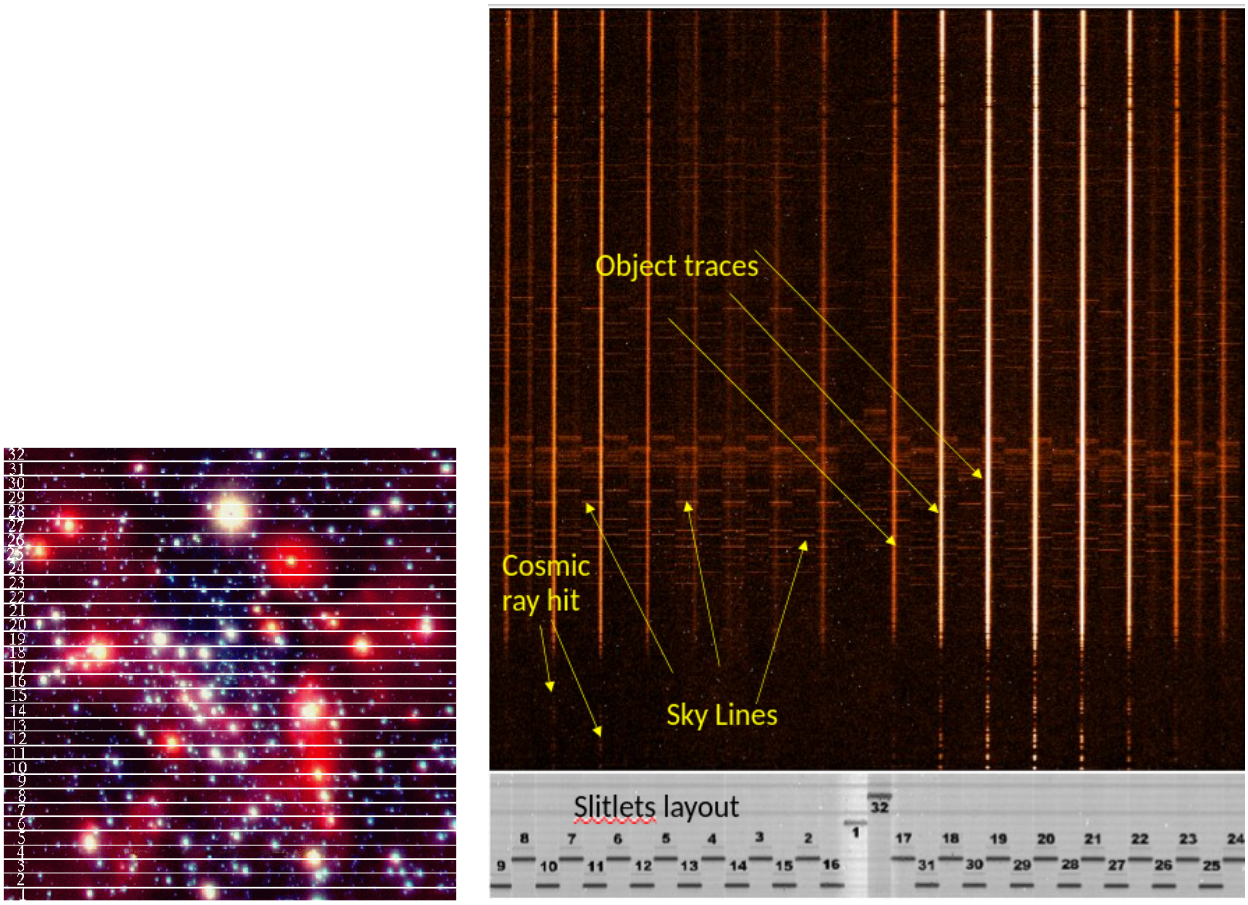


Figure 8.5: Here we show the image of a 3D data cube mean collapsed along the wavelength axis. In this case the object cannot be distuinguished from the background this leading to a failure of the extraction in case –mask_method is set to 'fit'. Using –mask_method=max one can still obtain a spectrum.

9 ERIS Data Description

In figure 9.1 we have represented how a possible observation target, (*a*), is imaged on the detector, (*b*). The 32 input source's slices generated by the two image slicers are imaged on the detector. These are also called slitlets and appear as vertical strips. Each slitlet has two edges, which can be seen by looking at the horizontal traces of sky emission lines or atmospheric absorption from the hydroxyl molecule OH. Those lines also show that the slitlets are arranged in a brick-wall pattern, to ease the detection of their edges, and that the first slitlet has a left edge not coincident with the left border of the detector. This information, together with the actual position of each slitlet edge is necessary to reconstruct the input FOV image. The spatial information (both X and Y directions on the sky) is distributed along the horizontal direction alone (the X sky coordinate in each slitlet and the Y sky coordinate in the different slitlets) while the wavelength information is along the vertical direction, increasing downwards. The signal from a point-like object in the FOV with a finite PSF appears as a vertical stripe in each slitlet where its emission is relevant and it is superimposed on the sky background, more uniform along the spatial direction except in coincidence of emission lines visible as horizontal bright stripes. During an exposure the detector can be hit by cosmic rays.



(*a*) (*b*)

Figure 9.1: A possible target for observation (*a*) and a typical raw science frame layout (*b*).

ERIS-SPIFFIER data reduction often involves pair of frames: calibrations in which an “on” frame is acquired with a calibration lamp switched on and an “off” frame recorded with the same lamp switched off, or observations of science objects and of the sky background. Additionally, dark frames are taken as well as special fibre flat frames used to compute optical distortions. Fibre flats are taken with the calibration fibre moved along the y-axis perpendicular to the image slices.

Each kind of raw frame is typically associated to a single ERIS-SPIFFIER pipeline recipe, *i.e.*, the recipe assigned to the reduction of that specific frame type. In the pipeline environment this recipe would be executed automatically. The recipe to compute optical distortions takes as input several kinds of raw frames, including slit-fibre flats, and standard pairs of flat field and arc lamp frames.

In the following sections after a brief description of how most of the raw data look like, all raw ERIS-SPIFFIER data frames are listed, together with the keywords used for their classification and correct association. The indicated *DO category* is a label assigned to any data type after it has been classified, which is then used to identify the frames listed in the *Set of Frames*. To classify the SPIFFIER raw data normally are sufficient the values of three dedicated FITS keywords, “DPR TYPE”, “DPR CATG”, “DPR TECH” (and the values of INSTRUME and “SEQ ARM”). Other FITS keyords, used to specify the preoptics and grating settings and the DIT are used to associate data. Those are indicated in the following subsections.

9.1 Darks

These are frames obtained with the shutter closed, acquired to evaluate the detector dark current level, build a master frame, used for quality control and instrument health diagnostic.

Frame tag: `DARK`
Processed by: `eris_ifu_mdark`

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB DPR.TYPE = DARK DPR.TECH = IMAGE INSTRUME = ERIS SEQ.ARM = SPIFFIER	DET.DIT TPL.START	

An example is shown in Table 9.1.

9.2 Linearity frames

A ramp of lamp-on frames and lamp-off frames is acquired at increasing detector DITs to evaluate the detector pixel linearity, determine the gain, and compute a map of non linear pixels.

Frame tag: `LINEARITY_LAMP`
Processed by: `eris_ifu_detlin`

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB DPR.TYPE = LINEARITY,LAMP,DETCHAR DPR.TYPE = LINEARITY,DARK,DETCHAR DPR.TECH = TECHNICAL INSTRUME = ERIS SEQ.ARM = SPIFFIER	TPL.START	

An example is shown in Table 9.2.

9.3 North-South test data

A set of frames, one obtained illuminating all slitlet edges with a slit, a dark, a flat and an arc lamp are acquired to determine the detector distortions.

Frame tags: DARK_NS, FIBRE_NS, FLAT_NS, WAVE_NS
Processed by: eris_ifu_distortion

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB DPR.TYPE = NS,DARK DPR.TYPE = NS,FLAT,DARK DPR.TYPE = NS,FLAT,LAMP DPR.TYPE = NS,WAVE,DARK DPR.TYPE = NS,WAVE,LAMP DPR.TECH = IFU INSTRUME = ERIS SEQ.ARM = SPIFFIER	TPL.START INS3.SPXW.ID INS3.SPXW.NAME INS3.SPGW.ID INS3.SPGW.NAME	

An example is shown in Table 9.3.

9.4 Flats

A set of usually five flat on frames and five flat off frames are acquired to determine a master flat to correct for pixel to pixel detector non uniformities and larger scale variations.

Frame tags: FLAT_LAMP
Processed by: eris_ifu_mflat

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB DPR.TYPE = FLAT,DARK DPR.TYPE = FLAT,LAMP DPR.TECH = IFU INSTRUME = ERIS SEQ.ARM = SPIFFIER	TPL.START INS3.SPXW.ID INS3.SPXW.NAME INS3.SPXW.ID INS3.SPXW.NAME	

An example is shown in Table 9.4.

9.5 Arc lamp frames

A pair of arc lamp on and off frames are acquired to perform the wavelength calibration.

Frame tags: `WAVE_LAMP`
Processed by: `eris_ifu_wavecal`

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB DPR.TYPE = WAVE,DARK DPR.TYPE = WAVE,LAMP DPR.TECH = IFU INSTRUME = ERIS SEQ.ARM = SPIFFIER	 TPL.START INS3.SPXW.ID INS3.SPXW.NAME INS3.SPGW.ID INS3.SPGW.NAME	

An example is shown in Table 9.5.

9.6 Flux standard star frames

A frame pair obtained observing a flux standard star and a sky frame are acquired to determine the instrument efficiency and the instrument response.

Frame tags: `STD_FLUX`
Processed by: `eris_ifu_stdstar`

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB DPR.TYPE = STD,FLUX DPR.TYPE = SKY,STD,FLUX DPR.TECH = IFU INSTRUME = ERIS SEQ.ARM = SPIFFIER	 TPL.START INS3.SPXW.ID INS3.SPXW.NAME INS3.SPGW.ID INS3.SPGW.NAME	

An example is shown in Table 9.6.

9.7 PSF standard star frames

A frame obtained observing a PSF standard star are acquired to determine the Strehl.

Frame tags: `PSF_CALIBRATOR`
Processed by: `eris_ifu_stdstar`

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB DPR.TYPE = PSF-CALIBRATOR DPR.TYPE = SKY,PSF-CALIBRATOR DPR.TECH = IFU INSTRUME = ERIS SEQ.ARM = SPIFFIER	 TPL.START INS3.SPXW.ID INS3.SPXW.NAME INS3.SPGW.ID INS3.SPGW.NAME	

An example is shown in Table 9.7.

9.8 Other standard star frames

Some observations of other standard stars can be performed.

Frame tags: `STD`
Processed by: `eris_ifu_stdstar`

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB		
DPR.TYPE = STD	TPL.START	
DPR.TYPE = SKY,STD	INS3.SPXW.ID	
DPR.TECH = IFU	INS3.SPXW.NAME	
INSTRUME = ERIS	INS3.SPGW.ID	
SEQ.ARM = SPIFFIER	INS3.SPGW.NAME	

An example is shown in Table 9.8.

9.9 Pupil monitoring frames

Some observations of other standard stars can be performed.

Frame tags: `PUPIL_LAMP`
Processed by: `eris_ifu_pupil`

Classification Keywords	Association Keywords	Remarks
DPR.CATG = CALIB		
DPR.TYPE = PUPIL,LAMP	TPL.START	
DPR.TYPE = SKY,PUPIL,LAMP	INS3.SPXW.ID	
DPR.TECH = IFU	INS3.SPXW.NAME	
INSTRUME = ERIS	INS3.SPGW.ID	
SEQ.ARM = SPIFFIER	INS3.SPGW.NAME	

9.10 Science observations

Science (and usually sky) frames are acquired to do science.

Frame tags: `OBJ`
Processed by: `eris_ifu_jitter`

Classification Keywords	Association Keywords	Remarks
DPR.CATG = SCIENCE		
DPR.TYPE = OBJECT	TPL.START	
DPR.TYPE = SKY	INS3.SPXW.ID	
DPR.TECH = IFU	INS3.SPXW.NAME	
INSTRUME = ERIS	INS3.SPGW.ID	
SEQ.ARM = SPIFFIER	INS3.SPGW.NAME	

An example is shown in Table 9.10.

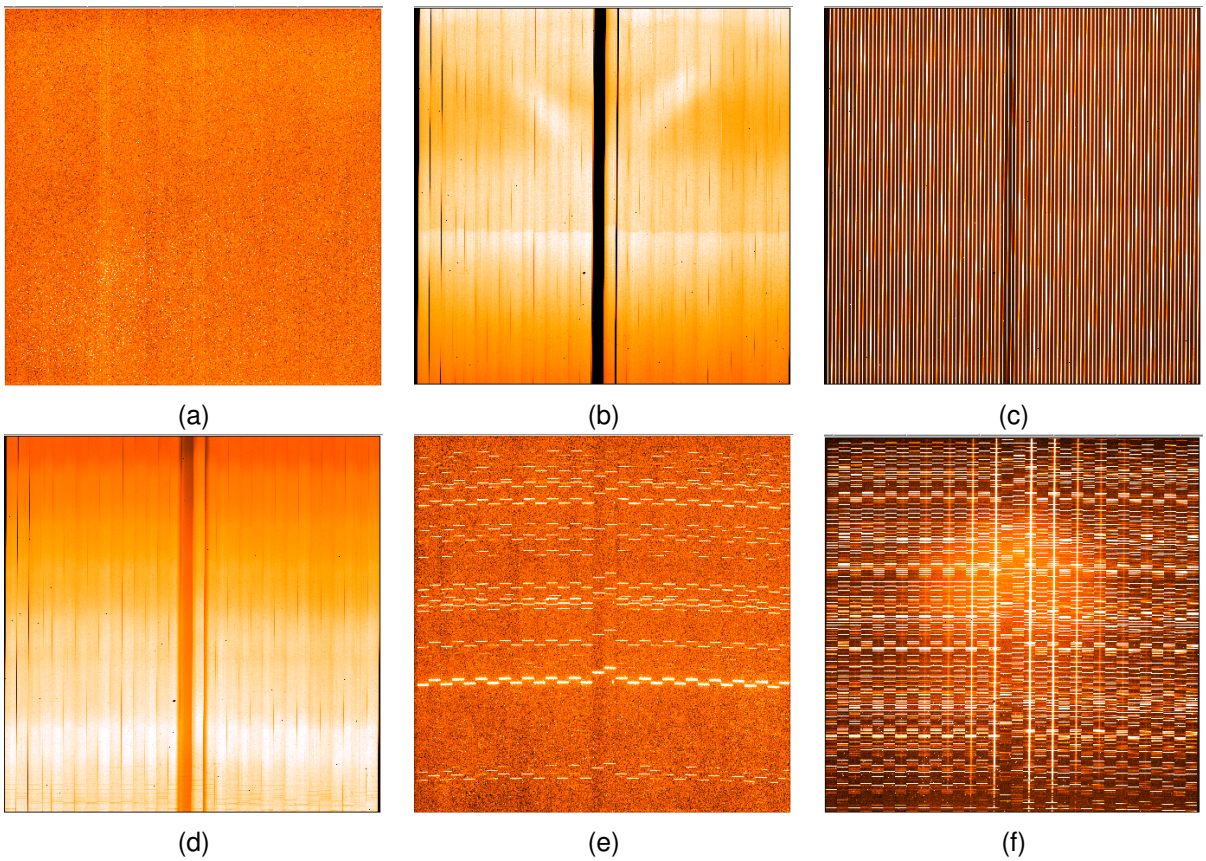


Figure 9.2: (a) a raw dark frame; (b) a raw linearity frame; (c) a raw slit fibre flat frame to compute optical distortions; (d) a raw flat on frame; (e) a raw arc lamp on frame; (f) a raw std star frame.

10 Static Calibration Data

10.1 First Wave Fit

This static table contains foreach instrument setting a list of a few lines, their wavelength and approximate position on the detector, that are used to perform the first step of the wavelength calibration. In this way one is sure to have a robust initial wavelength calibration solution that is later refined using the several more weaker lines that can be found on the detector.

Frame Tag: `FIRST_WAVE_FIT`
Classification keywords: `PRO.CATG = FIRST_WAVE_FIT`

Column Name	Column Type	Description
`instrument`	string	value of INSTRUME
`band`	string	value of observation band
`lamps`	string	calibration lamp id
`block`	int	block id
`position`	int	position id
`wavelength`	double	Line wavelength um
`Flux`	float	Relative line flux
`Ion`	string	Ion from which the line originates
`Quality`	int	Quality flag (0: undetected line, 1: etc.)
`Comment`	string	Optional column for further remarks

An example is shown in Table 10.1.

10.2 Reference Line Catalogue

This is a static table containing all lines that are used in the final fit of the wavelength calibration.

Frame Tag: `REF_LINE_ARC`
Classification keywords: `PRO.CATG = REF_LINE_ARC`

Column Name	Column Type	Description
`wavelength`	double	Line wavelength um
`intensity`	double	value of line intensity
`ignored`	int	if 0 the line is ignored
`element`	string	chemical line source
`state`	int	state of ionisation

An example is shown in Table 10.2.

10.3 Wavelength Setup Table

This is a static table and indicate foreach instrument setting the range to be used in the fit of the few reference lines used during the first step of the wavelength calibration.

Frame Tag: WAVE_SETUP
Classification keywords: PRO.CATG = WAVE_SETUP, INS3.SPXW.ID= J_low, or H_Low, or K_low

Column Name	Column Type	Description
band	string	observing band id
fwhm	double	Line FWHM pix
s_range	int	
c_range	int	
ri_dispersion	double	dispersion
ri_centrallambda	double	central wavelength

An example is shown in Table 10.3.

10.4 Catalog of Flux Standards Stars

Frame Tag: FLUX_STD_CATALOG
Classification keywords: PRO.CATG = FLUX_STD_CATALOG

This is a multi extension table containing in the first extension information on the extension number (ext_id) where is located the spectrum of a star, its name (name), and values of right ascension (ra) and declination (dec). The other extensions contain the spectra of wavelength (LAMBDA), flux (FLUX) in phisical units (erg/ph/cm2/s/A) and sampling bin width.

10.5 Atmospheric extinction table

Frame Tag: EXTCOEFF_TABLE
Classification keywords: PRO.CATG = EXTCOEFF_TABLE

This is a table containing information on the extinction (EXTINCTION) as a function of wavelength (LAMBDA). It is used to determine the instrument response and flux calibrate the data.

10.6 Efficiency Windows

Frame Tag: EFFICIENCY_WINDOWS
Classification keywords: PRO.CATG = EFFICIENCY_WINDOWS

This is a table indicating the start and end wavelength of each region where the efficiency is evaluated.

10.7 Fit Areas

Frame Tag: `FIT_AREAS`
Classification keywords: `PRO.CATG = FIT_AREAS`

This is a table indicating the start and end wavelength of each region where the a fit of the raw response spectrum is performed.

10.8 Quality Areas

Frame Tag: `QUALITY_AREAS`
Classification keywords: `PRO.CATG = QUALITY_AREAS`

This is a table indicating the start and end wavelength of each region where the quality of the match of a telluric model to the actual data is evaluated.

10.9 Response Windows

Frame Tag: `RESPONSE_WINDOWS`
Classification keywords: `PRO.CATG = RESPONSE_WINDOWS`

This is a table indicating the start and end wavelength of each region where the response fit is performed.

10.10 Catalog of fit points to fit the Response

Frame Tag: `RESP_FIT_POINTS_CATALOG`
Classification keywords: `PRO.CATG = RESP_FIT_POINTS_CATALOG`

This is a multi extension table with format similar to the FLUX_STD_CATALOG listing for each of the flux standards stars in the FLUX_STD_CATALOG, the wavelength of the points to be used to perform a fit.

10.11 Catalogue of Telluric models

Frame Tag: `TELL_MOD_CATALOG`
Classification keywords: `PRO.CATG = TELL_MOD_CATALOG`

This is a multi extension table with 73 spectra of telluric standards corresponding to different parameters of an atmospheric model. These are used during determination of the instrument response to identify the model that best matches to the data and then divide the observed spectrum by the best matching model to reduce the effect of the tellurics when evaluating the instrument response curve.

11 Data Reduction

11.1 The ERIS Data Reduction Pipeline

In this section, after an overview of the main problems the data reduction needs to solve, we list the required data and the recipes which allow to solve them, giving the data reduction sequence necessary to reduce calibration and science data.

11.2 Data reduction overview

The principle of integral field spectroscopy is described by figure 11.1. A two-dimensional image of the sky is separated by an image slicer into several components. Those are then aligned on a slit and dispersed to separate its spectral information and then imaged on a detector. The main ERIS-SPIFFIER data reduction problems to solve are the following.

- Correct for the detector signature: bad pixels, detector contribution to the measured signal, flat fielding (correct pixel to pixel gain variations and relative slitlet throughput differences), correct geometric distortions.
- Perform the wavelength calibration.
- Reconstruct the image FOV from the 32 image slices in a format which contains both the spatial and the spectral information.
- Devise proper calibrations and observations to be able to properly correct the emission from the sky, from the instrument and from the telescope which are very strong in the NIR. This requires to take sky frames together with the object frames in the night observations, daily calibrations with the flat lamp switched on and off, and possibly dark frames.

In the following description we also indicate in parenthesis for each frame the corresponding PRO.CATG. To locate the detector bad pixels one uses a bad pixel map. A master bad pixel map resulting from the combination of a set of (different) bad pixel maps is generated by the master flat recipe. This master pixel map describes the main cosmetics of the ERIS IFU detector, and its effects in the reconstructed cube is shown in figure 11.2. Hot pixels will be determined on dark frames (BPM_DARK). Non-linear response pixels are instead indicated by a bad pixel map (BPM_DETLIN) obtained by evaluating the pixel response of a set of flat exposures of increasing intensity. Other bad pixels (BPM_DIST) are determined processing North-South data.

A master flat field (MASTER_FLAT) generated from a set of raw flat fields, is used to correct the different detector pixel sensitivities. It is known that the image sliced and projected on the detector is affected by distortions. The eris_ifu_distortion recipe computes the distortions (DISTORTION) and the slitlet positions (SLITLET_POS). It uses a set of raw frames where only the first column of each slitlet is illuminated through a slit. In addition flats and arcs are taken within the north south template as required to reduce the data. A set of “on” and “off” arc lamp frames, a reference line table, a master flat field, the optical distortions map and a good guess of the

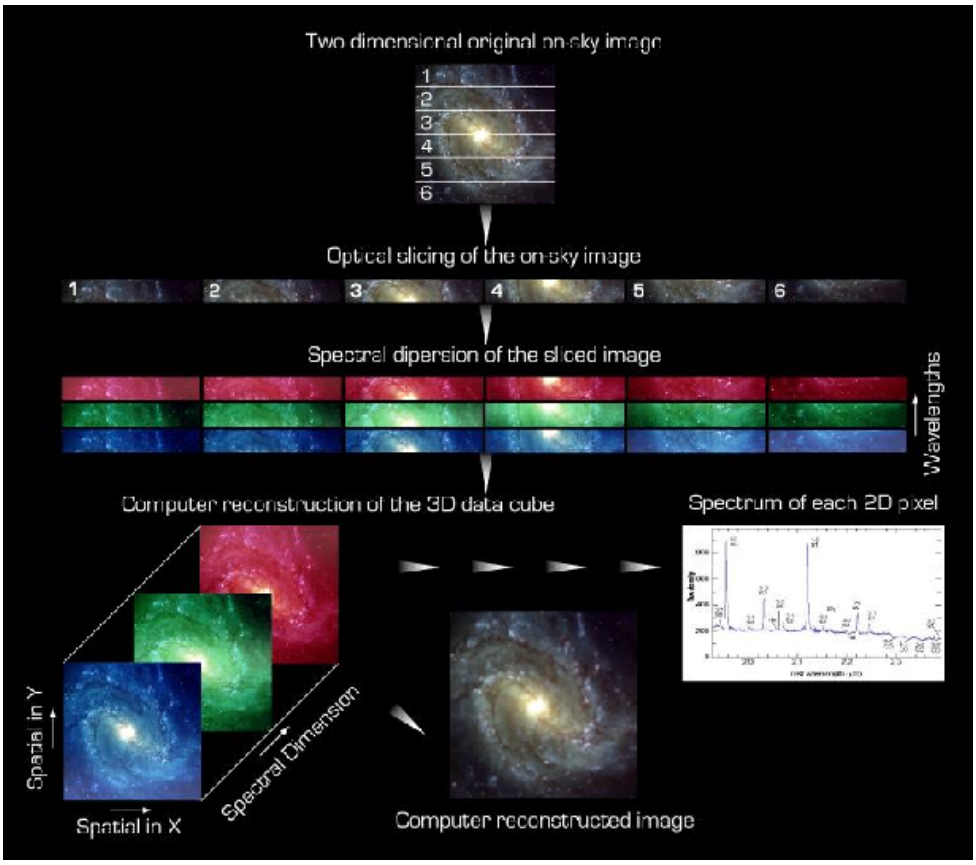


Figure 11.1: Integral Field Spectroscopy data reduction principle.

slitlet positions are input of the wavecal recipe. This recipe determines a wavelength map and obtains a better computation of the left edge position of each slitlet.

Science (or PSF or telluric standard) frames are corrected for sky background, flat field, distortions, and calibrated in wavelength. In order to correct the effect of instrument flexures during night observations, the wavelength calibration is corrected using the OH sky lines.

The sorted slitlets are then stacked in a cube, taking into account the relative distances of the position of the edge of each slitlet to the one of the first slitlet, with a final image realignment to get sub pixel accuracy. The final product is thus a 3D data cube where the full spatial information is stored along the X and Y directions, and the wavelength information is stored along the Z direction. Each plane of the cube is a monochromatic reconstruction of the ERIS-SPIFFIER FOV.

The north south test template traces each of the 32 slitlets by only one fibre exposure; therefore non linearities of the image scale within the 64 pixel of a single slitlets are currently not corrected and could cause minor slice to slit ripples in the reconstructed cube.

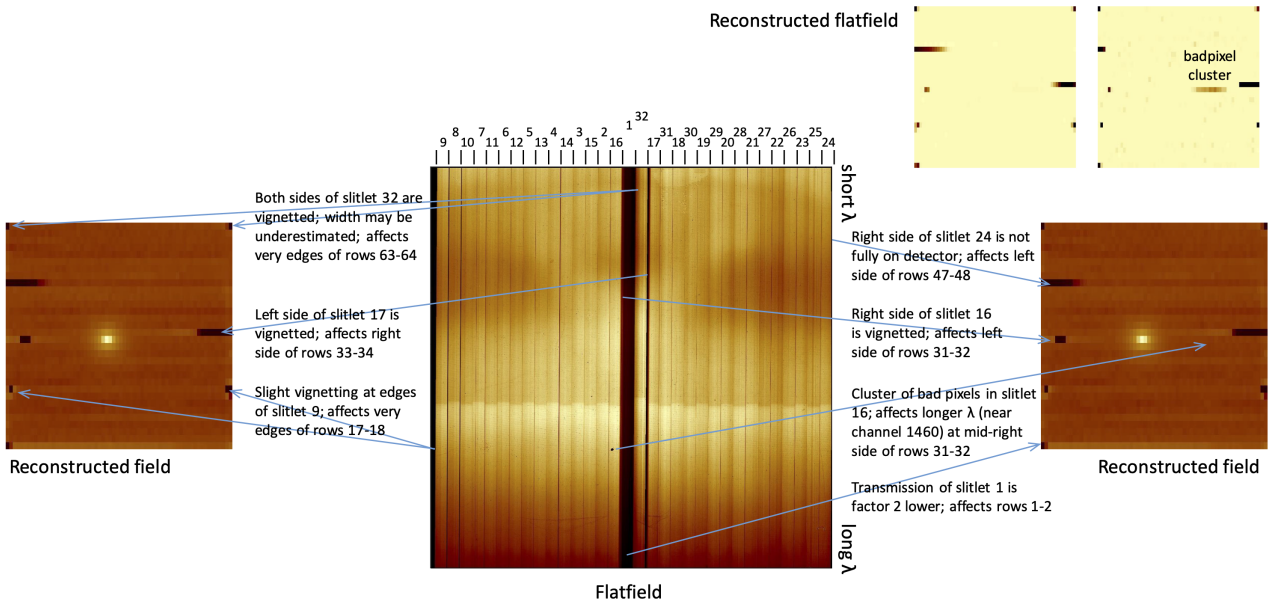


Figure 11.2: Main cosmetic effects in the reconstructed cube.

11.3 Required input data

To be able to reduce science data one needs to use raw, product data and pipeline recipes in a given sequence which provides all the input necessary to each pipeline recipe. We call this sequence a data reduction cascade. The ERIS-SPIFFIER data reduction cascade involves the following input data:

- Reference files:
  - A list of arc lamp emission lines containing vacuum wavelengths and predicted intensities for wavelength calibration.
  - The WAVE_SETUP table as input of eris_ifu_distortion and eris_ifu_wavecal to specify parameters peculiar of the wavelength calibration algorithm.
- Raw frames:
  - Linearity flat frames, to determine a map of non linear bad pixels.
  - Darks, to determine master darks.
  - Flat fields, to determine master flats.
  - Slit-Fibre frames, to trace the first column of each slitlet and, using also on/off lamp flats and arc lamp frames, to compute the optical distortions and slitlet distances.
  - Arc lamps, to perform the wavelength calibration.
  - Sky frames, to evaluate and subtract the strong and time-variable NIR sky emission.
  - Flux STD star frames, to compute the instrument response and efficiency.

  - PSF standards, to evaluate the Strehl.
  - Science frames, to finally do science.

- Calibration data products.
  - Bad pixel maps, to correct for the detector defects.
  - Distortion coefficients, to correct for the optical distortions.
  - Master flats, to correct for different detector pixel efficiencies.
  - Master darks, to correct for the instrument bias if no off or sky frames are available.
  - Slitlet positions, to be able to properly reconstruct a cube.
  - Wavelength maps, to obtain a cube calibrated in wavelength.

Calibration data products can be generated from raw data using the pipeline recipes. Alternatively the user may use calibration products obtained from the ESO archive or from the ESO Data Flow Operation department. Master bad pixel maps, the bad pixel maps coming from the standard flats, the master flats, the slitlets position table, the wavelength map depend from the observed band and instrument's pre-optic. Bad pixel maps coming from the non linearity test, distortion tables, slitlet distances, reference line tables depend only on the observed band. The reference bad pixel map, and master darks depend neither from the band nor the pre-optic. Science data requiring master dark need to have matching values of the FITS keyword DET SEQ1 DIT.

11.4 Reduction cascade

The ERIS-SPIFFIER data reduction follows the following sequence. A short description of the available recipes is given in section 7.1. In parenthesis we provide the value of the DO category corresponding to each frame.

- Run **eris_ifu_detlin** on a set of flats with increasing intensity (LINEARITY_LAMP) to determine the non linearity pixels bad pixel map (BPM_DETLIN).
- Run **eris_ifu_dark** on a set of raw darks (DARK) to determine the master dark (MASTER_DARK) and the hot pixels bad pixel map (BPM_DARK). This map depends on DET SEQ1 DIT.
- Run **eris_ifu_flat** on a set of standard flat fields (FLAT_LAMP), the BPM_DETLIN, BPM_DIST and the BPM_DARK, to determine the master bad pixel map (BPM_FLAT) and the master lamp flat (MASTER_FLAT).
- Run **eris_ifu_distortion** on a set of fibre flats (FIBRE_NS), on/off arc lamps (WAVE_NS) and on/off lamp flats (FLAT_NS), using a reference line table (REF_LINE_ARC) to determine the optical distortions (DISTORTION) and the slitlet distances (SLITLET_POS). To set a few data reduction parameters which depends from the observed band and used instrument pre-optics the user has also to provide in input a WAVE_SETUP table frame.
- Run **eris_ifu_wavecal** on a set of arc lamp frames (WAVE_LAMP), a MASTER_FLAT, recommended to remove long and short effects on the position of the arc lines before they are used to determine the wavelength solution, a DISTORTION, to determine the wavelength map (WAVE_MAP). To use a first guess solution, a list of reference arc lines, and to set a few data reduction parameters which depends from the

observed band and used instrument pre-optics the user has also to provide in input FIRST_WAVE_FIT, REF_LINE_ARC, a WAVE_SETUP.

- Run **eris_ifu_stdstar** on a reference flux standard (STD_FLUX) and a BPM_FLAT, a MASTER_FLAT, a DISTORTION, and a WAVE_MAP, and additional static calibrations, the EXTCOEFF_TABLE, the RESP_FIT_POINTS the TELL_MOD_CATALOG, the FIT_AREAS, the QUALITY_AREAS, the EFFICIENCY_WINDOWS, the FLUX_STD_CATALOG, to reduce the flux standards and get information on the instrument response and efficiency.
- Run **eris_ifu_stdstar** on PSF (or other STD) standards and a BPM_FLAT, a MASTER_FLAT, a DISTORTION, and a WAVE_MAP, a RESPONSE, and a EXTCOEFF_TABLE, to reduce the PSF (STD) standard and get information on the instrument's Strehl.
- Run **eris_ifu_jitter** on your scientific object (OBJ) and (if available) sky (SKY_OBJ) data and a BPM_FLAT, a MASTER_FLAT, a DISTORTION, and a WAVE_MAP the EXTCOEFF_TABLE, RESPONSE, to reduce science observations.

Examples of set of frames input for each recipe are provided in section 12.

12 Recipe Reference

12.1 Common Recipe Parameters

The following parameters can be applied to all recipes.

Parameter	Type	Default	Description
product_depth	integer	0	set what kind of products are created (0 minimal, 1 more, ...)
line_corr	boolean	FALSE	If TRUE raw exposure image line correction will be applied.
col_corr	boolean	FALSE	If TRUE raw exposure image column correction will be applied.
crh_corr	boolean	FALSE	If TRUE raw exposure image cosmic ray hit correction will be applied.
crh_detection	boolean	FALSE	If TRUE raw exposure image cosmic ray hit detection will be applied.
crh.sigma_lim	double	5.0	Poisson fluctuation threshold to flag cosmics(see van Dokkum, PASP,113,2001,p1420-27).
crh.f_lim	double	2.0	Minimum contrast between the Laplacian image and the fine structure image that a point must have to be flagged as cosmics.
crh.max_iter	int	3	Maximum number of algorithm iterations.

product_depth Defines which output products will be generated.

- **0 = PD_SCI** Only the main science or calibration products are created,
- **1 = PD_AUX** Additional products will be create which either help to evaluate the final results or which are saving important intermediate steps.
- **2 = PD_ALL** All products which might help the pipeline user are created.
- **3 = PD_DBG** In addition products which might help the pipeline developer are created.

line_corr For SINFONI/SPIFFIER the read out electronics subtracts from all pixels in a certain row the mean of the 4 referenc pixels to the left and the right of the frame. This is done on purpose in order to get rid of a row-number dependant bias. However, whenever a hot pixel appears among the reference pixels, this creates too dark lines in the image. To correct for this the mean of the reference pixels is added to the whole row and then the median of the reference pixels is subtracted.

col_corr For SINFONI/SPIFFIER there can be artefacts in the vertical direction of the raw exposure images. One of the pattern is that every second column is too bright and every other too dark. The problem can be correct by subtracting the value of reference pixels which are on the detector for each column: Calculate the median of the first 4 and the last 4 pixels of each column and subtract the result from all pixels in this column.

crh_corr Enable cosmic ray hit correction (not yet implemented)

crh_detection Enable cosmic ray hit detection (not yet implemented)

crh.sigma_lim Refers to σ_{lim} in the paper van Dokkum, PASP,113,2001,p1420-27

crh.f_lim Refers to f_{lim} in the paper van Dokkum, PASP,113,2001,p1420-27

crh.max_iter Maximum number of iteration for CRH detection

12.2 eris_ifu_dark

12.2.1 Description

This recipe performs ERIS/SPIFFIER dark frames data reduction. It calculates the master dark frame and a bad pixel mask.

It is required to provide three or more dark exposures to produce a reasonable master dark with associated noise.

Two different bad pixel mask are generated selected by the *–bpm.method* recipe parameter:

2D For the 2D bad pixel mask the master dark image is searched for outliers.

3D For the 3D bad pixel mask the provided dark exposures are stacked. Then outliers in the variance of each pixel are determined.

12.2.2 Input Frames

Frame Tag	Type	Count	Description
DARK	raw	>= 3	Raw dark exposure

12.2.3 Product Frames

Default File Name	Frame Tag	Product Depth	Description
eris_ifu_dark_master_dark.fits	MASTER_DARK_IFU	0	Master dark frame (data, noise, DQI)
eris_ifu_dark_bpm.fits	BPM_DARK	0	Bad pixel mask
eris_ifu_dark_bpm2d.fits	DARK_BPM2D	1	2D bad pixel mask
eris_ifu_dark_bpm3d.fits	DARK_BPM3D	1	3D bad pixel mask
eris_ifu_dark_contribMap.fits	DARK_CONTRIBMAP	2	contribution map from image list collapse
eris_ifu_dark_dbg_image.fits		3	master dark image (single FITS image)
eris_ifu_dark_dbg_cube.fits		3	input dark images as stack (FITS image list)
eris_ifu_dark_dbg_3d.fits		3	Detailed output of 3D bad pixel detection

eris_ifu_dark_master_dark.fits (MASTER_DARK_IFU) This is the master dark frame in the format with three FITS extensions: data, noise and data quality. The data quality extension is an integer image with following bit encodings:

bit number	bit value	description
4	8	saturated pixel
5	16	general bad pixel (the first/last 4 rows/columns)
7	64	bad pixel from BPM 2D
8	128	bad pixel from BPM 3D

eris_ifu_dark_bpm.fits (BPM_DARK) Bad pixel mask holding all types of bad pixels

eris_ifu_dark_bpm2d.fits (DARK_BPM2D) 2D bad pixel mask only

eris_ifu_dark_bpm3d.fits (DARK_BPM3D) 3D bad pixel mask only

eris_ifu_dark_contribMap.fits (DARK_CONTRIBMAP) Contribution map from image list collapse

eris_ifu_dark_dbg_image.fits Master dark image as a single FITS image

eris_ifu_dark_dbg_cube.fits A cube (image stack) of the input raw dark exposures is the primary FITS data section, the first extension stores a cube of the bad pixel masks created by reading the images (edge and saturated pixels) and the second extension contains an image of the summed bad pixel masks.

eris_ifu_dark_dbg_3d.fits The primary FITS data is an image where each pixel holds a count how many input frames are found too be bad for this pixel. The first extension hold a bit pattern image where the a set bit indicates the frame number where this pixel was detected as bad.

12.2.4 Recipe Parameters

Please see also the description for common recipe parameters in section 12.1 (page 53)

Parameter	Type	Default	Description
bpm.method	string enum ['2d', '3d', '2d3d', 'none']	2d3d	Specifies the VLT instrument 2d,3d,2d3d
collapse.method	string enum ['MEAN', 'WEIGHTED_MEAN', 'MEDIAN', 'SIGCLIP', 'MINMAX', 'MODE']	MEDIAN	Method used for collapsing the data
collapse.sigclip.kappa-low	double	3	Low kappa factor for kappa-sigma clipping algorithm
collapse.sigclip.kappa-high	double	3	High kappa factor for kappa-sigma clipping algorithm
collapse.sigclip.niter	int	5	Maximum number of clipping iterations for kappa-sigma clipping
collapse.minmax.nlow	double	1	Low number of pixels to reject for the minmax clipping algorithm
collapse.minmax.nhigh	double	1	High number of pixels to reject for the minmax clipping algorithm

continued on next page

continued from previous page

Parameter	Type	Default	Description
collapse.mode.histo-min	double	10	Minimum pixel value to accept for mode computation
collapse.mode.histo-max	double	1	Maximum pixel value to accept for mode computation
collapse.mode.bin-size	double	0	Binsize of the histogram
collapse.mode.method	string enum ['MEDIAN', 'WEIGHTED', 'FIT']	MEDIAN	Mode method (algorithm) to use
collapse.mode.error-niter	int	0	Iterations to compute the mode error
2dBadPix.method	string enum ['FILTER', 'LEGENDRE']	FILTER	Method used
2dBadPix.legendre.kappa-low	double	10	Low RMS scaling factor for image thresholding
2dBadPix.legendre.kappa-high	double	10	High RMS scaling factor for image thresholding
2dBadPix.legendre.maxiter	int	3	Maximum number of algorithm iterations
2dBadPix.legendre.steps-x	int	20	Number of image sampling points in x-dir for fitting
2dBadPix.legendre.steps-y	int	20	Number of image sampling points in y-dir for fitting
2dBadPix.legendre.filter-size-x	int	10	X size of the median box around sampling points
2dBadPix.legendre.filter-size-y	int	10	Y size of the median box around sampling points
2dBadPix.legendre.order-x	int	2	Order of x polynomial for the fit
2dBadPix.legendre.order-y	int	2	Order of y polynomial for the fit
2dBadPix.filter.kappa-low	double	10	Low RMS scaling factor for image thresholding
2dBadPix.filter.kappa-high	double	10	High RMS scaling factor for image thresholding
2dBadPix.filter.maxiter	int	3	Maximum number of algorithm iterations
2dBadPix.filter.filter	string enum ['AVERAGE', 'AVERAGE_FAST', 'MEDIAN']	MEDIAN	Filter mode for image smooting
2dBadPix.filter.border	string enum ['FILTER', 'CROP', 'NOP', 'COPY']	NOP	Border mode to use for the image smooting filter (only for MEDIAN filter)
2dBadPix.filter.smooth-x	int	5	Kernel y size of the smoothing filter
2dBadPix.filter.smooth-y	int	5	Kernel y size of the image smoothing filter
3dBadPix.kappa-low	double	12	Low RMS scaling factor for image thresholding.
3dBadPix.kappa-high	double	12	High RMS scaling factor for image thresholding.
3dBadPix.method	string enum ['absolute', 'relative', 'error']	relative	Thresholdig method to use for bpm detection
qc_ron_xmin	int range [1 ... 2044]	9	qc_ron_xmin
qc_ron_xmax	int range [1 ... 2048]	2040	qc_ron_xmax
qc_ron_ymin	int range [1 ... 2048]	9	qc_ron_ymin
qc_ron_ymax	int range [1 ... 2048]	2040	qc_ron_ymax

continued on next page

continued from previous page

Parameter	Type	Default	Description
qc_ron_hsize	int	4	qc_ron_hsize
qc_ron_nsamp	int	100	qc_ron_nsamp
qc_fpn_xmin	int range [1 ... 2048]	7	qc_fpn_xmin
qc_fpn_xmax	int range [1 ... 2048]	2042	qc_fpn_xmax
qc_fpn_ymin	int range [1 ... 2048]	7	qc_fpn_ymin
qc_fpn_ymax	int range [1 ... 2048]	2042	qc_fpn_ymax
qc_fpn_hsize	int	2	qc_fpn_hsize
qc_fpn_nsamp	int	1000	qc_fpn_nsamp

12.2.5 Quality Control Parameters

QC.DARK.NBADPIX	Total number of bad pixels but border pixels
QC.DARK.BPIXFRAC	Fraction of bad pixels to total number of pixels
QC.DARK.NBADPIXSAT	Saturated pixels count
QC.DARK.NBADPIX2D	Dark 2D bad pixels count
QC.DARK.NBADPIX3D	Dark 3D bad pixels count
QC.DARK.BPIXFRAC2D	Fraction of 2D bad pixel to total number of pixels
QC.DARK.BPIXFRAC3D	Fraction of 3D bad pixel to total number of pixels
QC.MASTERDARK.MEAN	[ADU] Clean mean of master dark image
QC.MASTERDARK.STDEV	[ADU] Clean stdev of master dark image
QC.DARKMED.AVE	[ADU] Average of raw darks medians
QC.DARKMED.STDEV	[ADU] Read Out Noise
QC.DARKFPN	Fixed Pattern Noise of combined frames
QC.RON	[ADU] Read Out Noise
QC.RONRMS	[ADU] Error on Read Out Noise
QC.RONi	[ADU] Read Out Noise
QC.NRONS	Number of RON frames

QC.DARKMED.AVE Calculate median of all raw dark exposures and return the average of the sequence

QC.DARKMED.STDEV Calculate median of all raw dark exposures and return the STDEV of the sequence

QC.DARKFPN Read out noise (see Readout Noise Computation, page 85) using the master dark frame

QC.RON Read out noise (see Readout Noise Computation, page 85) using the master dark frame

QC.RONRMS STDEV of the read out noise (see Readout Noise Computation, page 85) using the master dark frame

QC.RONi Read out noise (see Readout Noise Computation, page 85) using the difference of input frame *i* - input frame *i+1*

12.3 eris_ifu_detlin

12.3.1 Description

This recipe is used to determine a map of detector pixels that do not have a linear response to input light. The recipe input a set of (Lamp-on and Lamp-off) input frames obtained with increasing DET SEQ1 DIT.

12.3.2 Input Frames

Frame Tag	Type	Count	Description
LINEARITY_LAMP	raw	24 (min)	Raw linearity on and off frames

12.3.3 Product Frames

Default File Name	Frame Tag	Description
eris_ifu_detlin_gain_info.fits	GAIN_INFO	Table with information on gain comutation
eris_ifu_detlin_bpm.fits	BPM_DETLIN	Bad pixel mask frame
eris_ifu_detlin_bpm_filt.fits	BPM_DETLIN_FILT	Filtered bad pixel mask frame

12.3.4 Quality Control Parameters

QC.LIN.COEFFi	Linearity coefficient value
QC.LIN.COEFFi.ERR	Linearity coefficient error value
QC.LINEARITY.NBADPIX	Number of bad pixels
QC.LINEARITY.BPIXFRAC	Fraction of bad pixels to total number of pixels

12.3.5 Recipe Parameters

Parameter	Type	Values	Description
instrument	string	**ERIS**	Specifies the VLT instrument ERIS,SINFONI,NONE. <ERIS | SINFONI | NONE>
product_depth	int	**0**	Specifies the product output depth instrument (>0 for auxillary products).
degree	int	**2**	Degree of polynomial to fit.

continued on next page

continued from previous page

Parameter	Type	Values	Description
pval	double	**-1.0**	p-value threshold (in percent). Fits with a p-value below this threshold are considered bad pixels.
rel-chi-low	double	**8.0**	Relative chi threshold. Pixels with with a chi value smaller than mean - rel-threshold * stdev-of-chi are considered bad pixels.
rel-chi-high	double	**8.0**	Relative chi threshold. Pixels with with a chi value larger than mean + rel-threshold * stdev-of-chi are considered bad pixels.
rel-coef-low	double	**-1.0**	Relative fit coefficient threshold. Pixels with with a coefficient value smaller than mean +- rel-threshold * stdev-of-coeff are considered bad pixels.
rel-coef-high	double	**-1.0**	Relative fit coefficient threshold. Pixels with with a coefficient value larger than mean +- rel-threshold * stdev-of-coeff are considered bad pixels.
pfx	int	**3**	X Size of the post filtering kernel.
pfy	int	**3**	Y Size of the post filtering kernel.
pfm	string	**closing**	Post filtering mode. <closing | dilation>
stack-method	string	average, median, **ksigma**	Stacking method
clip-low	double	**3.**	Low sigma threshold for *ksigma* pixel rejection method
clip-high	double	**3.**	High sigma threshold for *ksigma* pixel rejection method

12.4 eris_ifu_distortion

12.4.1 Description

This recipe allows to correct for raw detector image distortions in X and Y directions. For more details, please see Section 13.2.3.

12.4.2 Input Frames

Frame Tag	Type	Count	Description
DARK_NS	raw	1 (min)	Raw dark frame
FIBRE_NS	raw	1 (min)	Raw slit fibre NS frame
FLAT_NS	raw	2 (min, on and off)	Raw flat NS frame
WAVE_NS	raw	2 (min, on and off)	Raw wave NS frame
FIRST_WAVE_FIT	static	1	Table with first fit solution
REF_LINE_ARC	static	1	Table reference arc line values
WAVE_SETUP	static	1	Table with guess wavelength calib solution values

12.4.3 Product Frames

Default File Name	Frame Tag	Description
eris_ifu_distortion_bpm.fits	BPM_DIST	Bad pixel mask frame
eris_ifu_distortion_slitlet_pos.fits	SLITLET_POS	Table frame with slitlet positions
eris_ifu_distortion_distortion.fits	DISTORTION	Table frame with distortions

12.4.4 Quality Control Parameters

QC.FLAT.SAT.NCOUNTS	nr. saturated pixels of master flat
QC.SPECFLAT.OFFFLUX	[ADU] average flux off frames
QC.SC.SPECFLAT.NCNTSAVG	[ADU] average counts
QC.SC.SPECFLAT.NCNTSSTD	[ADU] stdev counts
QC.LFLAT.FPNi	[ADU] Fixed Pattern Noise of combined frames
QC.FLAT.NBADPIX	Number of bad pixels
QC.FLAT.BPIXFRAC	Fraction of bad pixels to total
QC.FLAT.STDDEV	[ADU] Std deviation on flat image
QC.FLAT.MEAN	[ADU] Mean value on flat image
QC.FLAT.MEDIAN	[ADU] Median value on flat image
QC.DISTORTION.NBADPIX	Number of bad pixels
QC.DISTORTION.BPIXFRAC	Fraction of bad pixels to total

12.4.5 Recipe Parameters

```
clip-low & double & \textbf{\mbox{3.}} &
Low sigma threshold for \textit{ksigma} pixel rejection method \\

--instrument       : Specifies the VLT instrument {ERIS,SINFONI,NONE}. <ERIS | SINFONI | NONE> [NONE]
--product_depth       : Specifies the product output depth instrument (>0 for auxillary products). [0]
--bpm.method          : Specifies the VLT instrument {2d,3d,2d3d}. <2d | 3d | 2d3d | none> [2d]
--collapse.method     : Method used for collapsing the data. <MEAN | WEIGHTED_MEAN | MEDIAN | SIGCLIP | MINMA
--collapse.sigclip.kappa-low  : Low kappa factor for kappa-sigma clipping algorithm. [3.0]
```

```
--collapse.sigclip.kappa-high  : High kappa factor for kappa-sigma clipping algorithm. [3.0]
--collapse.sigclip.niter  : Maximum number of clipping iterations for kappa-sigma clipping. [5]
--collapse.minmax.nlow: Low number of pixels to reject for the minmax clipping algorithm. [1.0]
--collapse.minmax.nhigh  : High number of pixels to reject for the minmax clipping algorithm. [1.0]
--collapse.mode.histo-min  : Minimum pixel value to accept for mode computation. [10.0]
--collapse.mode.histo-max  : Maximum pixel value to accept for mode computation. [1.0]
--collapse.mode.bin-size  : Binsize of the histogram. [0.0]
--collapse.mode.method: Mode method (algorithm) to use. <MEDIAN | WEIGHTED | FIT> [MEDIAN]
--collapse.mode.error-niter  : Iterations to compute the mode error. [0]
--2dBadPix.method     : Method used. <FILTER | LEGENDRE> [LEGENDRE]
--2dBadPix.legendre.kappa-low  : Low RMS scaling factor for image thresholding. [10.0]
--2dBadPix.legendre.kappa-high  : High RMS scaling factor for image thresholding. [10.0]
--2dBadPix.legendre.maxiter  : Maximum number of algorithm iterations. [3]
--2dBadPix.legendre.steps-x  : Number of image sampling points in x-dir for fitting. [20]
--2dBadPix.legendre.steps-y  : Number of image sampling points in y-dir for fitting. [20]
--2dBadPix.legendre.filter-size-x  : X size of the median box around sampling points. [10]
--2dBadPix.legendre.filter-size-y  : Y size of the median box around sampling points. [10]
--2dBadPix.legendre.order-x  : Order of x polynomial for the fit. [2]
--2dBadPix.legendre.order-y  : Order of y polynomial for the fit. [2]
--2dBadPix.filter.kappa-low  : Low RMS scaling factor for image thresholding. [10.0]
--2dBadPix.filter.kappa-high  : High RMS scaling factor for image thresholding. [10.0]
--2dBadPix.filter.maxiter  : Maximum number of algorithm iterations. [3]
--2dBadPix.filter.filter  : Filter mode for image smooting. <AVERAGE | AVERAGE_FAST | MEDIAN> [MEDIAN]
--2dBadPix.filter.border  : Border mode to use for the image smooting filter (only for MEDIAN filter). <FILTE
                           [NOP]
--2dBadPix.filter.smooth-x  : Kernel y size of the smoothing filter. [5]
--2dBadPix.filter.smooth-y  : Kernel y size of the image smoothing filter. [5]
--3dBadPix.kappa-low  : Low RMS scaling factor for image thresholding. [12.0]
--3dBadPix.kappa-high : High RMS scaling factor for image thresholding. [12.0]
--3dBadPix.method     : Thresholdig method to use for bpm detection. <absolute | relative | error> [relative]
--flat.mode           : Mode of flat-calculation. <segment | hdrl | fast> [segment]
--flat_lo.filter-size-x  : Smoothing filter size in x-direction. [5]
--flat_lo.filter-size-y  : Smoothing filter size in y-direction. [5]
--flat_lo.method      : Method to use for the master flatfield calculation. <low | high> [low]
--flat_hi.filter-size-x  : Smoothing filter size in x-direction. [7]
--flat_hi.filter-size-y  : Smoothing filter size in y-direction. [7]
--flat_hi.method      : Method to use for the master flatfield calculation. <low | high> [high]
--qc_fpn_xmin1        : Fixed Pattern Noise: qc_fpn_xmin1. [512]
--qc_fpn_xmax1        : Fixed Pattern Noise: qc_fpn_xmax1. [1536]
--qc_fpn_ymin1        : Fixed Pattern Noise: qc_fpn_ymin1. [512]
--qc_fpn_ymax1        : Fixed Pattern Noise: qc_fpn_ymax1. [1536]
--qc_fpn_xmin2        : Fixed Pattern Noise: qc_fpn_xmin2. [1350]
--qc_fpn_xmax2        : Fixed Pattern Noise: qc_fpn_xmax2. [1390]
--qc_fpn_ymin2        : Fixed Pattern Noise: qc_fpn_ymin2. [1000]
--qc_fpn_ymax2        : Fixed Pattern Noise: qc_fpn_ymax2. [1200]
--distortion.arcs.thresh_factor  : median_value(image)+ kappa*sigma is the minimum intensity threshold of acc
                                   [0.33333]
--distortion.arcs.min_arclen_factor  : factor which sets minimum arc length (1.0.3). [1.19]
--distortion.arcs.window_size  : Size of window for low pass filter used in an horizontal low pass filter to
                                 (5-64). [14]
--distortion.smooth_rad  : Size of smoothing factor (1-11) used to prevent for possible intensity drops from
                           fibre illuminated slitlets (1-11). [3]
--wave.div            : Divide masterflat from subtracted wave frame. [FALSE]
--pixel_saturation    : Pixel saturation level. [1.8e+04]
```

12.5 eris_ifu_flat

12.5.1 Description

This recipe allows to determine a master flat to correct for the pixel-to-pixel and larger spatial scale detector responsiveness, determine a map of dead pixels, and combine this map with the other input bad pixel maps.

12.5.2 Input Frames

Frame Tag	Type	Count	Description
FLAT_LAMP	raw	10 (min)	Raw arc frame (5-on and 5-off)
BPM_DARK	cdb	1	hot pixel from dark frames
BPM_DETLIN	cdb	1	not linear pixels
BPM_DIST	cdb	1	bad pixels from distortion frames processing

12.5.3 Product Frames

Default File Name	Frame Tag	Description
eris_ifu_flat_bpm.fits	BPM_FLAT	dead bad pixels map
eris_ifu_flat_master_flat.fits	MASTER_FLAT	Master flat

12.5.4 Quality Control Parameters

QC.FLAT.SAT.NCOUNTS	nr. saturated pixels of master flat
QC.SPECFLAT.OFFFLUX	[ADU] average flux off frames
QC.SPECFLAT.NCNTSAVG	[ADU] average counts
QC.SPECFLAT.NCNTSSTD	[ADU] stdev counts
QC.LFLAT.FPNi	[ADU] Fixed Pattern Noise
QC.FLAT.NBADPIX	Number of bad pixels
QC.FLAT.BPIXFRAC	Fraction of bad pixels to total pixel number
QC.FLAT.STDDEV	[ADU] Std deviation on flat image
QC.FLAT.MEAN	[ADU] Mean value on flat image
QC.FLAT.MEDIAN	[ADU] Median value on flat image

12.5.5 Recipe Parameters

```
--instrument          : Specifies the VLT instrument {ERIS,SINFONI,NONE}. <ERIS | SINFONI | NONE> [ERIS]
--product_depth       : Specifies the product output depth instrument (>0 for auxillary products). [0]
--bpm.method          : Specifies the VLT instrument {2d,3d,2d3d}. <2d | 3d | 2d3d | none> [2d3d]
--collapse.method     : Method used for collapsing the data. <MEAN | WEIGHTED_MEAN | MEDIAN | SIGCLIP | MINMA
```

```
--collapse.sigclip.kappa-low  : Low kappa factor for kappa-sigma clipping algorithm. [3.0]
--collapse.sigclip.kappa-high  : High kappa factor for kappa-sigma clipping algorithm. [3.0]
--collapse.sigclip.niter  : Maximum number of clipping iterations for kappa-sigma clipping. [5]
--collapse.minmax.nlow: Low number of pixels to reject for the minmax clipping algorithm. [1.0]
--collapse.minmax.nhigh  : High number of pixels to reject for the minmax clipping algorithm. [1.0]
--collapse.mode.histo-min  : Minimum pixel value to accept for mode computation. [10.0]
--collapse.mode.histo-max  : Maximum pixel value to accept for mode computation. [1.0]
--collapse.mode.bin-size  : Binsize of the histogram. [0.0]
--collapse.mode.method: Mode method (algorithm) to use. <MEDIAN | WEIGHTED | FIT> [MEDIAN]
--collapse.mode.error-niter  : Iterations to compute the mode error. [0]
--2dBadPix.method     : Method used. <FILTER | LEGENDRE> [FILTER]
--2dBadPix.legendre.kappa-low  : Low RMS scaling factor for image thresholding. [10.0]
--2dBadPix.legendre.kappa-high  : High RMS scaling factor for image thresholding. [10.0]
--2dBadPix.legendre.maxiter  : Maximum number of algorithm iterations. [3]
--2dBadPix.legendre.steps-x  : Number of image sampling points in x-dir for fitting. [20]
--2dBadPix.legendre.steps-y  : Number of image sampling points in y-dir for fitting. [20]
--2dBadPix.legendre.filter-size-x  : X size of the median box around sampling points. [10]
--2dBadPix.legendre.filter-size-y  : Y size of the median box around sampling points. [10]
--2dBadPix.legendre.order-x  : Order of x polynomial for the fit. [2]
--2dBadPix.legendre.order-y  : Order of y polynomial for the fit. [2]
--2dBadPix.filter.kappa-low  : Low RMS scaling factor for image thresholding. [10.0]
--2dBadPix.filter.kappa-high  : High RMS scaling factor for image thresholding. [10.0]
--2dBadPix.filter.maxiter  : Maximum number of algorithm iterations. [3]
--2dBadPix.filter.filter  : Filter mode for image smooting. <AVERAGE | AVERAGE_FAST | MEDIAN> [MEDIAN]
--2dBadPix.filter.border  : Border mode to use for the image smooting filter (only for MEDIAN filter). <FILTE
--2dBadPix.filter.smooth-x  : Kernel y size of the smoothing filter. [5]
--2dBadPix.filter.smooth-y  : Kernel y size of the image smoothing filter. [5]
--3dBadPix.kappa-low  : Low RMS scaling factor for image thresholding. [12.0]
--3dBadPix.kappa-high : High RMS scaling factor for image thresholding. [12.0]
--3dBadPix.method     : Thresholdig method to use for bpm detection. <absolute | relative | error> [relative]
--flat.mode           : Mode of flat-calculation. <segment | hdrl | fast> [segment]
--flat_lo.filter-size-x  : Smoothing filter size in x-direction. [5]
--flat_lo.filter-size-y  : Smoothing filter size in y-direction. [5]
--flat_lo.method      : Method to use for the master flatfield calculation. <low | high> [low]
--flat_hi.filter-size-x  : Smoothing filter size in x-direction. [7]
--flat_hi.filter-size-y  : Smoothing filter size in y-direction. [7]
--flat_hi.method      : Method to use for the master flatfield calculation. <low | high> [high]
--qc_fpn_xmin1        : Fixed Pattern Noise: qc_fpn_xmin1. [512]
--qc_fpn_xmax1        : Fixed Pattern Noise: qc_fpn_xmax1. [1536]
--qc_fpn_ymin1        : Fixed Pattern Noise: qc_fpn_ymin1. [512]
--qc_fpn_ymax1        : Fixed Pattern Noise: qc_fpn_ymax1. [1536]
--qc_fpn_xmin2        : Fixed Pattern Noise: qc_fpn_xmin2. [1350]
--qc_fpn_xmax2        : Fixed Pattern Noise: qc_fpn_xmax2. [1390]
--qc_fpn_ymin2        : Fixed Pattern Noise: qc_fpn_ymin2. [1000]
--qc_fpn_ymax2        : Fixed Pattern Noise: qc_fpn_ymax2. [1200]
```

12.6 eris_ifu_wavecal

12.6.1 Description

This recipe allows to perform the wavelength calibration. For more details, please see Section 13.2.5.

12.6.2 Input Frames

Frame Tag	Type	Count	Description
WAVE_LAMP	raw	1 (min)	Raw arc frame
MASTER_FLAT	cdb	1 (recommended)	Master flat frame
DISTORTION	cdb	1	Master flat frame
FIRST_WAVE_FIT	static	1	Table with first fit solution
REF_LINE_ARC	static	1	Table reference arc line values
WAVE_SETUP	static	1	Table with guess wavelength calib solution values

12.6.3 Product Frames

Default File Name	Frame Tag	Description
eris_ifu_wave_arcImg_stacked.fits	WAVE_LAMP_STACKED	Processed (stacked) arc lamp frame
eris_ifu_wave_arcImg_resampled.fits	WAVE_LAMP_STACKED_RESAMPLED	As previous but resampled
eris_ifu_wave_map.fits	WAVE_MAP	Wavelength map (pixel intensity equal to wavelength)

12.6.4 Quality Control Parameters

QC.FRMONi.MEANFLUX	[ADU] average of flux of image i
QC.FRMONi.MEDIANFLUX	[ADU] median of flux of image i
QC.FRMONi.MAXFLUX	[ADU] Max of flux of image i
QC.FRMONi.NPIXSAT	Number of saturated pixels of image i
QC.FRMOFFi.MEANFLUX	[ADU] Average of flux of image i
QC.FRMOFFi.MEDIANFLUX	[ADU] Median of flux of image i
QC.FRMOFFi.MAXFLUX	[ADU] Max of flux of image i
QC.FRMOFFi.NPIXSAT	Number of saturated pixels of image i
QC.FRMDIFF.MEANFLUX	[ADU] Average of flux of frame diff
QC.FRMDIFF.MEDIANFLUX	[ADU] Median of flux of frame diff
QC.FRMDIFF.MAXFLUX	[ADU] Max of flux of frame diff
QC.FRMDIFF.NPIXSAT	Number of saturated pixels of frame diff
QC.SLITLETi.PEAK.AVG	[ADU] average line peak flux for slitlet i
QC.SLITLETi.PEAK.MED	[ADU] median line peak flux for slitlet i
QC.SLITLETi.RESOL.AVG	average resolution power for slitlet i
QC.SLITLETi.RESOL.MED	median resolution power for slitlet i
QC.SLITLETi.RESOL.STD	stdev resolution power for slitlet i
QC.RESOL.AVG	average resolution power
QC.RESOL.MED	median resolution power
QC.RESOL.STD	stdev resolution power
QC.SLITLETi.FWHM.AVG	[pix] average FWHM of found lines for slitlet i
QC.SLITLETi.FWHM.MED	[pix] median FWHM of found lines for slitlet i
QC.SLITLETi.FWHM.STD	[pix] stdev FWHM of found lines for slitlet i

QC.FWHM.AVG	[pix] average FWHM of found lines
QC.FWHM.MED	[pix] median FWHM of found lines
QC.FWHM.STD	[pix] stdev FWHM of found lines
QC.SLITLETi.NFITLINES	Number of lines used for the fit on slitlet i
QC.NFITLINES	Number of lines used for the fit
QC.COEFi.AVG	Average wavecal Coef
QC.COEFi.MED	Median wavecal Coef
QC.POSERR.AVG	[um] Average of reference line position errors
QC.POSERR.MED	[um] Median of reference line position errors
QC.POSERR.CLEAN.AVG	[um] Clean average of reference line position errors
QC.POSERR.CLEAN.MED	[um] Clean median of reference line position errors
QC.POSERR.AVG.ABS	[um] Average of reference line position absolute errors
QC.POSERR.MED.ABS	[um] Median of reference line position absolute errors
QC.POSERR.CLEAN.AVG.ABS	[um] Clean average of reference line position absolute errors
QC.POSERR.CLEAN.MED.ABS	[um] Clean median of reference line position absolute errors
QC.WCEN.VALUE	[um] Central wavelength value
QC.WCEN.ERR	[um] Error on central wavelength value
QC.ARCLINE.PEAK.AVG	[ADU] Average peak flux of fit lines on frame
QC.ARCLINE.PEAK.MED	[ADU] Median peak flux of fit lines on frame
QC.ARCLINE.PEAK.STD	[ADU] Stdev peak flux of fit lines on frame
QC.ARCLINE.PEAK.MAX	[ADU] Max peak flux of fit lines on frame

12.6.5 Recipe Parameters

```
--instrument          : Specifies the VLT instrument {ERIS,SINFONI,NONE}. <ERIS | SINFONI | NONE> [ERIS]
 --product_depth       : Specifies the product output depth (>0 for auxiliary products). [0]
 --line_corr           : If TRUE raw exposure image line correction will be applied. [TRUE]
 --col_corr            : If TRUE raw exposure image column correction will be applied. [TRUE]
 --crh_corr            : If TRUE raw exposure image cosmic ray hit correction will be applied. [FALSE]
 --crh_detection       : If TRUE raw exposure image cosmic ray hit detection will be applied. [FALSE]
 --crh.sigma_lim       : Poisson fluctuation threshold to flag cosmics(see van Dokkum, PASP,113,2001,p1420-27)
 --crh.f_lim           : Minimum contrast between the Laplacian image and the fine structure image that a poin
 --crh.max_iter        : Maximum number of alghoritm iterations. [3]
 --pixel_saturation    : Pixel saturation level. [1.8e+04]
```

Parameter	Type	Values	Description
instrument	string	**ERIS**	Specifies the VLT instrument ERIS,SINFONI,NONE. <ERIS | SINFONI | NONE>
product_depth	int	**0**	Specifies the product output depth instrument (>0 for auxillary products).
line_corr	boolean	**TRUE**	If TRUE raw exposure image line correction will be applied. [TRUE]
col_corr	boolean	**TRUE**	If TRUE raw exposure image column correction will be applied. [TRUE]

continued on next page

continued from previous page

Parameter	Type	Values	Description
crh_corr	boolean	**FALSE**	If TRUE raw exposure image cosmic ray hit correction will be applied. [FALSE]
crh_detection	boolean	**FALSE**	If TRUE raw exposure image cosmic ray hit detection will be applied. [FALSE]
crh.sigma_lim	double	**5.0**	Poisson fluctuation threshold to flag cosmics(see van Dokkum, PASP,113,2001,p1420-27). [5.0]
crh.f_lim	double	**2.0**	Minimum contrast between the Laplacian image and the fine structure image that a point must have to be flagged as cosmics. [2.0]
crh.max_iter	int	**3**	Maximum number of alghoritm iterations. [3]

If the user would like to get more (quality control) information on results of the fit, we recommend to set the parameter –product_depth=3 to get the product `eris_ifu_wave_dbg_fit_tables.fits` (PRO.CATG=WAVE_FIT_TABLES) containing all results of the fit, in the appropriate extension. This table contains several extentions: the primary FITS header; the extention slitletFittingTable with data used to fit the IFU slitlets; columnFittingTable with data used to fit the arc lines; slitletCoeffs, with the slitlet id, the number of lines, the polynomial degree and aplot of the coeffifients of the polinomial fit used to fit the slitlets;columnCoeffRaw, with the slitlet id, the number of used lines, the degree of the polynomial used to fit the columns, and the corresponding plot; columnCoeffsSmoother, same as columnCoeffRaw but after smoothing to remove outliers; smoothCoeff a table of the polinomial used to smooth the coefficients. The columnFittingTable contains data reative to the Gaussian fit of the arclines. To make sure the data are good enough to do the fit the following criteria are used:

```
dataRange = dataMax - dataMin
MIN_SIGNAL_RANGE = 5.0
if dataRange < MIN_SIGNAL_RANGE { errorCode = -4}

#define MIN_PEAK_THRESHOLD      10.0
if (
      (fitErr == \verb+CPL_ERROR_NONE+) &&
      (gaussPar.offset > -10. - gaussPar.peak / 300.) &&
      (gaussPar.sigma < waveSetup.sigma) &&
      (gaussPar.sigma > 0.1) &&  /* NEW: minimum sigma */
      (fabs(gaussPar.x0 - firstGuessPos) < 5.01) &&
      (gaussPar.peak > MIN_PEAK_THRESHOLD)
    ) {  /* NEW: minimum peak */
```

The possible fit error codes are defined as follows:

Code	Meaning
0	Successful fit that passed all validation
>0	CPL fitting error code
-1	Search position outside valid spectrum range
-2	Fit succeeded but failed parameter validation
-3	NEW: Peak below `MIN_PEAK_THRESHOLD` (likely noise)
-4	NEW: Pre-fit check failed: data too flat (range $<$ 5 ADU)

12.7 eris_ifu_stdstar

12.7.1 Description

12.7.2 Input Frames

Frame Tag	Type	Count	Description
STD (*)	raw	1 (min)	Raw flux STD frame
SKY_STD (**)	raw	1 (min)	Raw sky frame
MASTER_FLAT	cdb	1	Master flat frame
BPM_FLAT	cdb	1	Master bad pixel map from flat frame
DISTORTION	cdb	1	Distortion frame
WAVE_MAP	cdb	1	Wavelength map frame
OH_SPEC	static	1	Table with OH lines to get OH based wavelength solution
EXTCOEFF_TABLE	static	1	atmospheric extiction table
RESP_FIT_POINTS_CATALOG	static	1	catalog with response fit points
TELL_MOD_CATALOG	static	1	catalogue with model spectra of atmospheric telluric absorbtion lines
FLUX_STD_CATALOG	static	1	reference flux std catalog for efficiency and response computation
FIT_AREAS	static	1	table indicating region to fit response (band = J_low, H_low, K_low)
QUALITY_AREAS	static	1	table with regions to evaluate quality of fit response (band = J_low, H_low, K_low)
EFFICIENCY_WINDOWS	static	1	table with regions used for efficiency computation (band = J_low, H_low, K_low)

Notes:

(*) Alternative input tags are `STD_FLUX` or `PSF_CALIBRATOR` respectively for an input flux standard star or a PSF standard star.

(**) Alternative input tags are SKY_STD_FLUX or SKY_PSF_CALIBRATOR respectively for the sky associated to an input flux standard star or to a PSF standard star.

12.7.3 Product Frames

Default File Name	Frame Tag	Description
eris_ifu_stdstar_dar_cube_001.fits	DAR_CORRECTED_CUBE	Object Data cube DAR corrected (created if -dar-corr=TRUE)
eris_ifu_stdstar_sky_cube_000.fits	SKY_CUBE	Sky Data cube
eris_ifu_stdstar_std_cube_001.fits	STD_CUBE	Object Data cube DAR
eris_ifu_stdstar_std_cube_coadd.fits (*)	STD_CUBE_COADD (*)	Combined data cube
eris_ifu_stdstar_std_cube_mean.fits (**)	STD_CUBE_MEAN (**)	Mean of combined cube
eris_ifu_stdstar_cube_median.fits	STD_CUBE_MEDIAN	Median of combined cube
eris_ifu_stdstar_extraction_mask.fits	EXTRACTION_MASK	Mask used to extract cube
eris_ifu_stdstar_spectrum.fits	SPECTRUM	extracted spectrum
eris_ifu_stdstar_response.fits	RESPONSE	computed response
eris_ifu_stdstar_spectrum_fluxcal.fits	SPECTRUM_FLUXCAL	Flux calibrated STD star spectrum
eris_ifu_stdstar_cube_fluxcal.fits	STD_CUBE_COADD_FLUXCAL	Flux calibrated STD star cube
eris_ifu_stdstar_no_flat_efficiency.fits	EFFICIENCY	computed efficiency

(*) In case of STD_FLUX input the product will be named eris_ifu_stdstar_std_flux_cube_coadd.fits and have PRO.CATG STD_FLUX_CUBE_COADD. In case of PSF_CALIBRATOR input the product will be named eris_ifu_psf_flux_cube_coadd.fits and have PRO.CATG PSF_CUBE_COADD.

(**) In case of STD_FLUX input the product will be named eris_ifu_stdstar_std_flux_cube_mean.fits and have PRO.CATG STD_FLUX_CUBE_MEAN. In case of PSF_CALIBRATOR input the product will be named eris_ifu_psf_flux_cube_mean.fits and have PRO.CATG PSF_CUBE_MEAN.

12.7.4 Quality Control Parameters

QC.OH.LAMBDA.SHIFT.UM	[um] OH based center lambda shift to (each) lamp calib frame
QC.OH.LAMBDA.SHIFT.PIX	[pix] OH based center lambda shift to (each) lamp calib frame
QC.FLAT.SAT.NCOUNTS	[] nr. saturated pixels of master flat
QC.SPECFLAT.OFFFLUX	[] average flux off frames
QC.SPECFLAT.NCNTSAVG	[] average counts
QC.SPECFLAT.NCNTSSTD	[] stdev counts
QC.LFLAT.FPN1	[] Fixed Pattern Noise of combin
QC.LFLAT.FPN2	[] Fixed Pattern Noise of combin
QC.FLAT.NBADPIX	Number of bad pixels
QC.FLAT.BPIXFRAC	Fraction of bad pixels to to
QC.RESP.WINi.WLMIN	[um] Min window wavelength for resp co
QC.RESP.WINi.WLMAX	[um] Max window wavelength for resp co
QC.RESP.WINi.MEAN	Mean response on window
QC.RESP.WINi.MEDIAN	Median response on window
QC.RESP.WINi.MIN	Min response on window

QC.RESP.WINi.MAX	Max response on window
QC.RESP.WINi.RMS	RMS response on window
QC.RESP.WLMIN	[um] Min wavelength for resp comp
QC.RESP.WLMAX	[um] Max wavelength for resp comp
QC.RESP.MEAN	Mean response
QC.RESP.MEDIAN	Median response
QC.RESP.MIN	Min response
QC.RESP.MAX	Max response
QC.RESP.RMS	RMS response
QC.FWHM.LLX	[pix] Lower lext X pos for FWHM detection
QC.FWHM.LLY	[pix] Lower lext Y pos for FWHM detection
QC.FWHM.HBX	[pix] Half Box size for FWHM detection
QC.FWHM.CENTERX	[pix] STD star X centroid position
QC.FWHM.CENTERY	[pix] STD star Y centroid position
QC.FWHM.MAJ	[pix] STD star FWHM on major axis size
QC.FWHM.MIN	[pix] STD star FWHM on minor axis size
QC.CHECK1	Check on evaluation box
QC.FWHMX	[pix] STD star FWHM on X direction
QC.FWHMY	[pix] STD star FWHM on Y direction
QC.STREHL	Computed Strehl
QC.STREHL.ERR	Computed Strehl error
QC.EFF.WINi.WLMIN	[um] Min window wavelength for eff comp
QC.EFF.WINi WLMAX	[um] Max window wavelength for eff comp
QC.EFF.WINi MEAN	Mean efficiency on window
QC.EFF.WINi MEDIAN	Median efficiency on window
QC.EFF.WINi MIN	Min efficiency on window
QC.EFF.WINi MAX	Max efficiency on window
QC.EFF.WINi RMS	RMS efficiency on window
QC.EFF.WIN.WLMIN	[um] Min wavelength for eff comp
QC.EFF.WIN WLMAX	[um] Max wavelength for eff comp
QC.EFF.WIN MEAN	Mean efficiency
QC.EFF.WIN MEDIAN	Median efficiency
QC.EFF.WIN MIN	Min efficiency
QC.EFF.WIN MAX	Max efficiency
QC.EFF.WIN RMS	RMS efficiency

12.7.5 Recipe Parameters

```
--instrument          : Specifies the VLT instrument {ERIS,SINFONI,NONE}. <ERIS | SINFONI | NONE> [ERIS]
--product_depth       : Specifies the product output depth (>0 for auxiliary products). [0]
--line_corr           : If TRUE raw exposure image line correction will be applied. [TRUE]
--col_corr            : If TRUE raw exposure image column correction will be applied. [TRUE]
--crh_corr            : If TRUE raw exposure image cosmic ray hit correction will be applied. [FALSE]
--crh_detection       : If TRUE raw exposure image cosmic ray hit detection will be applied. [FALSE]
--crh.sigma_lim       : Poisson fluctuation threshold to flag cosmics(see van Dokkum, PASP,113,2001,p1420-27)
--crh.f_lim           : Minimum contrast between the Laplacian image and the fine structure image that a poin
```

```
                        as cosmics. [2.0]
--crh.max_iter         : Maximum number of alghoritm iterations. [3]
--pixel_saturation     : Pixel saturation level. [1.8e+04]
--sky_tweak            : Use modified sky cube for sky subtraction.0: don't apply, 1: Davies' method, 2: Austr
--skip_sky_oh_align    : Skip the OH alignment for the SKY. [FALSE]
--oh_align_poly_order : Order of polynomial fit used in the OH alignment for the SKY: [0,3]. 0 is recommended
                        to prevent extrapolation problems at band edges and in band K. A higher value may
                        recover a failed correction in case of large flexures,  but is less accurate at band
                        edges due to polynomial extrapolation. [0]

--discard_subband      : Ignore last sub-band in the sky tweaking. [FALSE]
--stretch              : Stretch sky in the sky tweaking. [FALSE]
--stretch_degree       : Stretch polynomial degree. [8]
--stretch_resampling   : Stretch resampling method (linear/spline). [spline]
--tbsub                : Subtract thermal background from input cube.(TRUE (apply) or FALSE (don't apply). [TR
--velocity_offset      : Specify velocity offset correction in km/s for lambda scale. [0.0]
--bpm_threshold        : Specify a theshold for the interpolated BPM values. [0.5]
--cube.slitlet-detection  : Specifies the slitlet detection: "DIST" slitlet distances as detected by distorti
                            slitlet edges as detected be wavecalrecipe. <DIST | EDGE | GRID> [DIST]
--cube.first-col       : Specifies the first column offset in case the cube.slitlet-detection parameter is set
--cube.fine-tune       : Specifies the row interpolation mode: 0 -> no interpolation, otherwise see eris_ifu_1
--method               : Resampling method. <NEAREST | LINEAR | QUADRATIC | RENKA | DRIZZLE | LANCZOS> [LANCZO
--method.loop-distance: Loop distance used by all (but NEAREST) methods to control the number of surrounding
                        into account. [1]
--method.use-errorweights  : Use additional weights of 1/err^2. [FALSE]
--method.renka.critical-radius  : Critical radius of the Renka method. [1.25]
--method.lanczos.kernel-size  : Kernel size of the Lanczos method. [2]
--method.drizzle.downscale-x  : Drizzle down-scaling factor in x direction. [0.8]
--method.drizzle.downscale-y  : Drizzle down-scaling factor in y direction. [0.8]
--method.drizzle.downscale-z  : Drizzle down-scaling factor in wavelength direction. [0.8]
--subtract-background : Subtract median of the images in 2D only. [FALSE]
--fieldmargin          : Ad this margin/border (in percent) to the resampled image/cube. [5.0]
--edge-trim            : Number or pixels to trim for each plane of the input frames. It should be smaller tha
--extract-source       : If True try to extract a point souce. [TRUE]
--mask_method          : Method to specify extraction mask : mask, position, max or fit. [max]
--center               : The centre of the circular mask (pixel). [32,32]
--radius               : The radius of the circular mask (pixel). [9.0]
--flux-calibrate       : If True flux calibrate the extracted spectrum and data cube. [TRUE]
--dar-corr             : Correct Differential Atmospheric Refraction (DAR). [TRUE]
--dar-shift-method     : The method to shift images for DAR correction. [0]
--dar-shift-length     : Kernel length for DAR shift image interpolation. [0]
--dar-shift-radius     : Kernel radius for DAR shift image interpolation. [0.0]
--fwhm-factor          : Factor to find 2D-Gauss FWHM. The extraction box is: halfbox_x=halfbox_y=fwhm_factor*
--max-cubes-centres-dist  : Maximum distance between cube centers to build a mosaic. Mosaic creation requires
--cube.combine         : With multi-pointing observations combine cubes into a mosaic. [TRUE]
--derot_corr           : The first column offset to be applied on the science data for distortion correction.
                        within +/- 2 pixels. A default correction based on altitude and rotation angle will b
                        [auto]
--bpc_iter             : No. of iterations for bad pixel correction. [1]
--chop-nan             : If true chop cube planes with more than 50% NAN pixels. [FALSE]
--crea-phase3          : If true crea phase3 compliant data products. [FALSE]

--compute-efficiency   : If True try to compute the efficiency from a STD star spectrum. [TRUE]
--compute-response     : If True try to compute the response from a STD star spectrum. [TRUE]
--compute-strehl       : If True try to compute the response from a PSF star spectrum. [FALSE]
--strehl_flux_radius : PSF Flux integration radius [pixels]. If -1 uses 3 times object PSF FWHM. [-1.0]
--strehl_bkg-radius-low : PSF background inner radii [pixels]. If -1 uses 1.5 times strehl_flux_radius. [-1.0
--strehl_bkg-radius-high : PSF background outer radius [pixels]. If -1 uses 2.0 times strehl_flux_radius. [-1
--response-polyfit-deg: Degree of polynomial fit of response data points. [3]
--response-polyfit-ksigma-kappa  : Kappa value used to clip data points in polynomial fit of response data po
--response-polyfit-ksigma-niter  : Degree of polynomial fit of response data points. [3]
```

12.8 eris_ifu_jitter

12.8.1 Description

The *eris_ifu_jitter* recipes reduces scientific targets data by implementing following steps:

- Collect bad pixel masks
- Apply flat field correction (when a master flat image is provided)
- Resample the exposure frames to wavelength calibrated data cubes
- Correct for sky features
- Correct for differential atmospheric refraction (DAR)
- Resample multiple exposure to a single co-added cube
- Extract a source spectrum

Bad pixel masks
The recipe accepts several bad pixel masks (BPM_DARK, BPM_FLAT and BPM_DETLIN) and the data quality extension of the MASTER_FLAT image. All the bad pixel masked are or'ed so a bad pixel in any mask flags this pixel a bad in the final mask.

Flat fielding When a MASTER_FLAT image is provided all exposure images will be divided by it.

Sky correction The recipe checks the FITS header information of the exposure raw frames to find the optimum object-sky pairs. If several skies are available, the one closest in time to the object will be picked. If there is no sky available, there won't be any sky subtraction applied. [3] The order of the files in the SOF is irrelevant.

When the *–sky_tweak* parameter is set to `0` the sky exposure image is simply subtracted from the object exposure. Setting the *–sky_tweak* parameter to `1` activates the sky subtraction using the method of Ric Davies[**?**]. Here the sky cube is tweaked according to pixels in the object cube which represent the background signal. Then this adapted sky cube is subtracted from the object cube. The parameter *–tbsub* is recommended to be set to TRUE (as default) when *–sky_tweak=1* (Davies's method) is used. The thermal background in the object cube will be subtracted.

For short exposures such as used with standard stars, especially when observed in the smallest 25mas scale, set *–sky_tweak = 0* because the OH lines are too faint to apply the scaling corrections reliably.

Concerning the sky tweak we recommend to use the default parameters except *–sky_tweak=1* which turns it on. It is possible to change the wavelength shift from a constant offset to a linear or even quadratic function (these are the *–stretch* and *–stretch_degree* parameters). And this can help, but it is also very dangerous. The reason is that if the OH lines the routine uses are not spread over the full spectrum, one can have a long extrapolation of the function from the last OH line to the end of the spectrum. And this can lead to very large offsets which are probably wrong. We expect sky-tweak to get the OH residual down to the $\simeq$ 1% level. But if one is taking data of really faint objects, this can still appear to be large. And the main improvement then is to clip the residuals away as bad pixels.

[3] In case a dedicated sky frame is missing, the sky can still be modeled and corrected with the methods controlled by positive values of the parameter **aj-method**. We remind the reader to a description of that parameter for more details.

A MASTER_DARK of matching DET.SEQ1.DIT should be used especially for long exposures, since it locates most of the bad pixels that would be later one corrected.

–bpc_iter is introduce to the bad pixel correction. Default iteration is set to 1.

Differential atmospheric refraction (DAR) correction Setting the *dar-corr* is set to `TRUE` the DAR correction will be enabled. With the *–dar-shift-method* parameter several methods can be chosen to shift the image planes of the cube.

The input cube type depends on the preprocessing:

- When DAR correction is enabled the DAR corrected cubes will be used.
- When sky tweaking is enabled the sky tweaked cubes will be used.
- Otherwise the object cubes will be used.

When there is just one object cube the co-added cube is just a copy of it. The resampling method and its parameters can be tuned using the *method* recipe parameter.

Spectrum extraction The pipeline may extract a spectrum if the parameter *extract-source* is set to TRUE. Current version implements different extraction methods, controlled by the parameter *mask_method*, that can be set to values: mask, position, max, fit or optimal. Method `mask` allows the user to specify an (optional) input mask to define how to wait the extracted cube's slice image pixels. Method *position* allows the user to specify, via the parameters `center` and `radius` the center and the radius of a circular mask used for extraction. The method `max` determines the center of the circular extraction mask finding the position of the maximum in the (collapsed cube) image. The method `fit` performs a 2D Gaussian fit of the object on the collapsed cube image. Finally the method `optimal` performs a Horne based optimal extraction (for more details please refer to section 13.2.15 at pag. 95). The most robust method is the default, `fit`.

12.8.2 Input Frames

Frame Tag	Type	Count	Description
OBJ	raw	1 ... n	Object exposure frame
SKY or SKY_OBJ	raw	1 ... n	Sky exposure frame
DISTORTION	cdb	1	Table with distortion correction parameters
WAVE_MAP	cdb	1	Wavelength calibration map
MASTER_DARK	cdb	[0,1]	Optional master dark image
MASTER_FLAT	cdb	[0,1]	Optional master flat image
BPM_DARK	cdb	[0,1]	Optional bad pixel mask
BPM_FLAT	cdb	[0,1]	Optional bad pixel mask
BPM_DETLIN	cdb	[0,1]	Optional bad pixel mask
OH_SPEC	static	1	Vector holding OH lines spectrum
RESPONSE	cdb	[0,1]	Instrument response
EXTCOEFF_TABLE	static	[0,1]	Atmospheric extiction table

Due to presence of tellurics, to obtain better flux calibrated results the user should input the RESPONSE ob-

tained in the "_low" resolution setting. If the raw data are of a high resolution setting the pipeline will automatically trim the response to the correct wavelength range of the given high resolution band.

12.8.3 Product Frames

Default File Name	Frame Tag	Product Depth	Description
eris_ifu_jitter_obj_cube_*nnn*.fits	OBJECT_CUBE	0	Object data cube
eris_ifu_jitter_sky_cube_*nnn*.fits	SKY_CUBE	0	Sky data cube
eris_ifu_jitter_twk_cube_*nnn*.fits	SKY_TWEAKED_CUBE	0	Object data sky tweaked cube
eris_ifu_jitter_dar_cube_*nnn*.fits	DAR_CORRECTED_CUBE	0	Object data cube DAR corrected
eris_ifu_jitter_dar_cube_coadd.fits	DAR_CORRECTED_CUBE_COADD	0	Coadded object cube, DAR corrected
eris_ifu_jitter_dar_cube_mean.fits	DAR_CORRECTED_CUBE_MEAN	0	Collapsed object cube, DAR corrected
eris_ifu_jitter_twk_cube_coadd.fits	SKY_TWEAKED_CUBE_COADD	0	Co-added object cube, "sky tweaked"
eris_ifu_jitter_twk_cube_mean.fits	SKY_TWEAKED_CUBE_MEAN	0	Collapsed object cube, "sky tweaked"
eris_ifu_jitter_obj_cube_coadd.fits	OBJECT_CUBE_COADD	0	Co-added object cube
eris_ifu_jitter_obj_cube_mean.fits	OBJECT_CUBE_MEAN	0	Collapsed object cube
eris_ifu_jitter_bpm_cube_*nnn*.fits	BPM_CUBE	0	Interpolated BPM of object/sky cube
eris_ifu_jitter_cube_median.fits	STD_CUBE_MEDIAN	0	Median of co-added cube
eris_ifu_jitter_extraction_mask.fits	EXTRACTION_MASK	0	Mask used to extract cube
eris_ifu_jitter_spectrum.fits	SPECTRUM	0	extracted spectrum
eris_ifu_jitter_spectrum_fluxcal.fits	SPECTRUM_FLUXCAL	0	Flux calibrated object spectrum
eris_ifu_jitter_cube_fluxcal.fits	OBJECT_CUBE_COADD_FLUXCAL	0	Flux calibrated object cube
eris_ifu_jitter_cube_fluxcal_mean.fits	OBJECT_CUBE_COADD_FLUXCAL_MEAN	0	Flux calibrated object cube mean
eris_ifu_jitter_obj_exposure_map.fits	EXPOSURE_MAP	0	Exposure map

eris_ifu_jitter_obj_cube_*nnn*.fits (OBJECT_CUBE) Reconstructed cube from the n^{th} raw exposure frame specified in the SOF file (tagged as object).

eris_ifu_jitter_sky_cube_*nnn*.fits (SKY_CUBE) Reconstructed cube from the n^{th} raw exposure frame specified in the SOF file (tagged as sky).

eris_ifu_jitter_twk_cube_*nnn*.fits (SKY_TWEAKED_CUBE) Reconstructed cube from the n^{th} raw exposure frame specified in the SOF file with the sky tweak algorithm applied.
This product is only generated when the *–sky_tweak* parameter is set to 1.

eris_ifu_jitter_dar_cube_*nnn*.fits (DAR_CORRECTED_CUBE) Reconstructed cube from the n^{th} raw exposure frame specified in the SOF file with the DAR correction applied. If the *–sky_tweak* parameter is set to one the sky tweak algorithm is also applied.
This product is only generated when the *–dar-corr* parameter is set to TRUE.

eris_ifu_jitter_dar_cube_coadd.fits (DAR_CORRECTED_CUBE_COADD) Resampled cube using all DAR_CORRECTED_CUBE cubes as input.
This product is only generated when both the *–dar-corr* is set to TRUE.

eris_ifu_jitter_dar_cube_mean.fits (DAR_CORRECTED_CUBE_MEAN) Image of the collapsed DAR_CORRECTED_CUBE_COADD cube calculating the average of each spaxel.

eris_ifu_jitter_twk_cube_coadd.fits (SKY_TWEAKED_CUBE_COADD) Resampled cube using all SKY_TWEAKED_CUBE cubes as input.
This product is only generated when the *–sky_tweak* parameter is set to 1 and the *–dar-corr* parameter is set to FALSE.

eris_ifu_jitter_twk_cube_mean.fits (SKY_TWEAKED_CUBE_MEAN) Image of the collapsed SKY_TWEAKED_CUBE_COADD cube calculating the average of each spaxel.

eris_ifu_jitter_obj_cube_coadd.fits (OBJECT_CUBE_COADD) Resampled cube using all OBJECT_CUBE cubes as input.

eris_ifu_jitter_obj_cube_mean.fits (OBJECT_CUBE_MEAN) Image of the collapsed OBJECT_CUBE_COADD cube calculating the average of each spaxel.

eris_ifu_jitter_bpm_cube_*nnn*.fits (BPM_CUBE)

eris_ifu_jitter_cube_median.fits (STD_CUBE_MEDIAN) Image of the collapsed OBJECT_CUBE_COADD cube calculating the average of each spaxel. This file is only created when the source spectrum extraction is requested as this image is the one to locate the source by a 2D Gauss fit.

eris_ifu_jitter_extraction_mask.fits (EXTRACTION_MASK) Mask image indicating pixels used to extract the object.

eris_ifu_jitter_spectrum.fits (SPECTRUM) The extracted spectrum.

eris_ifu_jitter_spectrum_fluxcal.fits (SPECTRUM_FLUXCAL) The flux calibrated extracted spectrum.

eris_ifu_jitter_cube_fluxcal.fits (STD_CUBE_COADD_FLUXCAL) The flux calibrated combined cube.

eris_ifu_jitter_cube_fluxcal_mean.fits (STD_CUBE_COADD_FLUXCAL_MEAN) The flux calibrated combined cube mean.

eris_ifu_jitter_obj_cube_exposure_map.fits (EXPOSURE_MAP) The a map indicating the median exposure time used for each cube plane exposed data point.

In case of data acquired in pupil tracking mode (DPR.TECH='IFU,NODDING,PT') to combined cube (OBJECT_CUBE_COADD) product is created and consequently also the products OBJECT_CUBE_MEAN, EXTRACTION_MASK, SPECTRUM, SPECTRUM_FLUXCAL, OBJECT_CUBE_COADD_FLUXCAL are not generated.

12.8.4 Recipe Parameters

We list here only the most important parameters. Please see also the description for common recipe parameters in section 12.1 (page 53)

Parameter	Type	Default	Description
sky_tweak	int	0	Use modified sky cube for sky subtraction.0: don't apply, 1: Davies' method, 2: Austrian method)
skip_sky_oh_align	bool	False	Skip the OH alignment for the SKY (deprecated)[4]
oh_align_poly_order	int	0	Order of polynomial fit used in the OH alignment for the SKY: [0,3]. 0 is recommended to prevent extrapolation problems at band edges and in band K. A higher value may recover a failed correction in case of large flexures, but is less accurate at band edges due to polynomial extrapolation.
discard_subband	bool	False	Ignore last sub-band in the sky tweaking
stretch	bool	False	Stretch sky in the sky tweaking
stretch_degree	int	8	Stretch polynomial degree
stretch_resampling	string	spline	Stretch resampling method (linear/spline)
tbsub	bool	True	Subtract thermal background from input cube.(TRUE (apply) or FALSE (don't apply)
velocity_offset	double	0	Specify velocity offset correction in km/s for lambda scale
bpm_threshold	double	0.5	Specify a theshold for the interpolated BPM values
cube.slitlet-detection	string enum ['DIST', 'EDGE', 'GRID']	DIST	Specifies the slitlet detection: "DIST" slitlet distances as detected by distortion recipe or "EDGE" slitlet edges as detected be wavecal-recipe
cube.first-col	double	1	Specifies the first column offset in case the cube.slitlet-detection parameter is set to "DIST"
cube.fine-tune	int	-1	Specifies the row interpolation mode: 0 -> no interpolation, otherwise see eris_ifu_1d_interpolation
cube.combine	bool	True	With multi-pointing observations combine cubes into a mosaic.
method	string enum ['NEAREST', 'LINEAR', 'QUADRATIC', 'RENKA', 'DRIZZLE', 'LANCZOS']	DRIZZLE	Resampling method
method.loop-distance	int	3	Loop distance used by all (but NEAREST) methods to control the number of surrounding voxels that are taken into account. A small value allow faster re-sampling but may not give good quality
method.use-errorweights	bool	False	Use additional weights of $1/err^2$
method.renka.critical-radius	double	1.25	Critical radius of the Renka method
method.lanczos.kernel-size	int	2	Kernel size of the Lanczos method
method.drizzle.downscale-x	double	0.8	Drizzle down-scaling factor in x direction

continued on next page

[4] In case there are no OH lines detectable, the OH alignment can be skipped. (i.e. very bright object, K_LONG, small plate-scale, short DIT). As a result there will be an overall wavelength shift in the output cube which has to be corrected oneself.

continued from previous page

Parameter	Type	Default	Description
method.drizzle.downscale-y	double	0.8	Drizzle down-scaling factor in y direction
method.drizzle.downscale-z	double	0.8	Drizzle down-scaling factor in wavelength direction
subtract-background	bool	False	Subtract median of the images in 2D only
fieldmargin	double	5	Ad this margin/border (in percent) to the resampled image/cube
edge-trim	int	0	Number or pixels to trim for each plane of the input frames. It should be smaller than half image size
extract-source	bool	False	If True try to extract a point souce
mask_method	string	max	Method to specify extraction mask : mask, position, max, fit or optimal
center	string	32,32	The centre of the circular mask (pixel)
radius	double	4	The radius of the circular mask (pixel)
flux-calibrate	bool	False	If True flux calibrate the extracted spectrum and data cube
dar-corr	bool	False	Correct Differential Atmospheric Refraction (DAR)
dar-shift-method	int range [0 ... 8]	0	The method to shift images for DAR correction
dar-shift-length	int	0	Kernel length for DAR shift image interpolation
dar-shift-radius	double	0	Kernel radius for DAR shift image interpolation
fwhm-factor	double	5	Factor to find 2D-Gauss FWHM. The extraction box is: halfbox_x=halfbox_y=fwhm_factor*(fwhm_x+fwhm_y)*0.5.
max-cubes-centres-dist	int	240	Maximum distance between cube centers to build a mosaic. Mosaic creation requires a lot of RAM. Users may trim this value to fit RAM resources.
cube.combine	bool	TRUE	With multi-pointing observations combine cubes into a mosaic.
derot_corr	string	auto	The first column offset to be applied on the science data for distortion correction. This effect is visible when the spectrum on one side of the cube wraps to the other side. Typical value is within +/- 2 pixels. A default correction based on altitude and rotation angle will be applied if this parameter set to auto; otherwise the user-speficied number of the shift would be used.

continued on next page

continued from previous page

Parameter	Type	Default	Description
aj-method	int	7	Method used to estimate the sky in case of missing sky frames. 0: no sky subtraction. 1: sky is taken from the next in MJD-OBS object frame. 2: sky is taken from the median of all input sky frames. 3: sky is taken from the mean of all input sky frames. 4: sky is taken from the median of user spec. rect. box; 5: sky is taken from the median of 4 regions at FOV edges; 6: sky is taken from the median of 8 regions at FOV edges; 7: sky is taken from each cube slice based on ks-clip based mask computed on collapsed cube.8: sky is taken from each cube slice based on 25% clip based mask computed on collapsed cube.
sky-box-center	int,int	32,32	The centre of the rectangular sky region (pixel). two integer numbers each in the range [1,64].
sky-box-width	int,int	1,1	The rectangular (small) sky region x,y half widths [pixel].Two numbers each in the range [0,10].
sky-box-edges-margin	int, int	1,1	The (small)number of x, y pixels taken as marging to the FOV edges so that sky regions (aj-method=5,6) do not include pixels at the edges of the re-sampled image [pixel]. Two numbers each in the range [0,5].
sky-est-method	int	0	If the user has no input sky frame and the sky need to be estimated from user defined pixels set by aj-method this parameter allow to use the median (0) or the mean (1) [0].
sky-est-kappa	double	3.0	If the user has no input sky frame and the sky need to be estimated from user defined pixels set by aj-method this parameter allow to use the kappa of the kappa sigma clip iteration
sky-est-niter	int	5	If the user has no input sky frame and the sky need to be estimated from user defined pixels set by aj-method this parameter allow to set the number of the kappa sigma clip iterations
bpc_iter	int	1	No. of iterations for bad pixel correction.
chop-nan	boolean	FALSE	If true chop cube planes with more than 50% NAN pixels.
crea-phase3	boolean	FALSE	If true crea phase3 compliant data products.

dar-corr Is set to `TRUE` the object cubes will be corrected for differential atmospheric refraction (DAR).

dar-shift-method Algorithm to shift the image planes of the cubes according the DAR correction.

- **0** cpl_image_warp function using the kernel profile TANH
- **1** cpl_image_warp function using the kernel profile SINC
- **2** cpl_image_warp function using the kernel profile SINC2
- **3** cpl_image_warp function using the kernel profile LANCZOS

4 cpl_image_warp function using the kernel profile HAMMING

5 cpl_image_warp function using the kernel profile HANN

6 cpl_image_warp function using the kernel profile NEAREST

7 GSL[**?**] 2D interpolation function using the interpolation type bilinear

8 GSL[**?**] 2D interpolation function using the interpolation type bicubic

dar-shift-length Kernel length for DAR shift image interpolation, only used if *dar-shift-method* is `0 ... 6`. The default value of `0` selects the CPL default kernel length CPL_KERNEL_DEF_SAMPLES which is `2001`

dar-shift-radius Kernel radius for DAR shift image interpolation, only used if *dar-shift-method* is `0 ... 6`.The default value of `0` selects the CPL default kernel width CPL_KERNEL_DEF_WIDTH which is `2.0`

skip_sky_oh_align The setting skip_sky_oh_align=TRUE applies for a special processing mode for the sky-tweak=1. For the sky-tweak=1 sky subtraction the sky cube can be aligned to the object cube so the wavecal alignment using the OH lines is not needed and may be skipped. But then the "stretch*" parameters have to be set. It is possible to change the wavelength shift from a constant offset to a linear or even quadratic function (these are the -stretch and -stretch_degree parameters).

oh_align_poly_order This parameter control the order of the polynomial fit used in the OH alignment for the SKY. The allowed range is [0,3]. 0 is recommended to prevent extrapolation problems at band edges and in band K (in particular for K_LONG). A higher value may recover a failed correction in case of large flexures, but is less accurate at band edges due to polynomial extrapolation.

cube.combine This parameter allow to de/activate combination of 3D data cubes in case of multi-pointing observations. It maybe used to speed-up data reduction if the user is not interest to the combined cube. If set to false the extracted spectrum will not be created.

aj-method When the target is compact, it can be dithered inside the SPIFFER FoV, without the need for sky frames. In this case, the sky in each frame could be estimated by using the mean or the median between the frames, or (probably better), one of the frames closest in time. A negative image of the target will be present in the final data, but this is not a prioblem if the dithering pattern is accurately designed. This parameter, in case the input data do not contain a dedicated sky observation, controls the creation of a “fake” sky frame from the object frame data. The pipeline currently support the following integer values:

- 0: no sky subtraction.
- 1: sky is taken from the next in MJD-OBS object frame.
- 2: sky is taken from the median of all input sky frames.
- 3: sky is taken from the mean of all input sky frames.
- 4: sky is modeled on each cube plane from the median (or mean) on a user defined rectangular window.
- 5: sky is modeled on each cube plane from the median (or mean) of the medians (or means) computed on four windows taken at the edges of the object FOV.

- 6: sky is modeled on each cube plane from the median (or mean) of the medians (or means) computed on eight windows taken at the border of the object FOV.
- 7: sky is modeled on each cube plane from the median (or mean) computed on the pixels of a region of pixels obtained afte **sky-est-niter** iterations where in each are kappa sigma clipped pixels above the median (or mean).
- 8: sky is modeled on each cube plane from the median (or mean) computed on the pixels of a region of pixels obtained using 25% pixels with minimum intensity.

The current pipeline generates 3D data cubes that have planes filled by NANs at the start and at the end of the Z axis (wavelength direction). This is a consequence of the brick-wall pattern in the data. For this reason we decided, in case of **aj-method** values between 4 and 8, to skip the first and last 75 planes of the 3D data cube during sky modelling and correction. This affects also the extracted spectrum. We considered this as a negligible effect considering that anyway those planes are fully or partially affected by NAN values, and thus of little use.

The default value is 7 is expected to better estimate the sky in all cases. Method 1 allows to obtain best results if the user plans a proper observation sequence. Results on possible observations are shown in Figure 12.1. We emphasize that to benefit from this method and observation choice it is up to the user to choose the proper preoptics (we recommend 100mas or 250 mas) and jitter sequence so that every frame is shifted with respect to the two adjacent in MJD-OBS by at least three times the object FWHMs, so that the next frame in MJD-OBS can be a good sky estimation, without overlapping signal.

flux-calibrate Please note that this release uses information (start and end wavelength) from the extracted spectrum to define the range over the combined cube is flux calibrated. For this reason we recommend the user sets –extract-source=TRUE when flux-calibrate=TRUE.

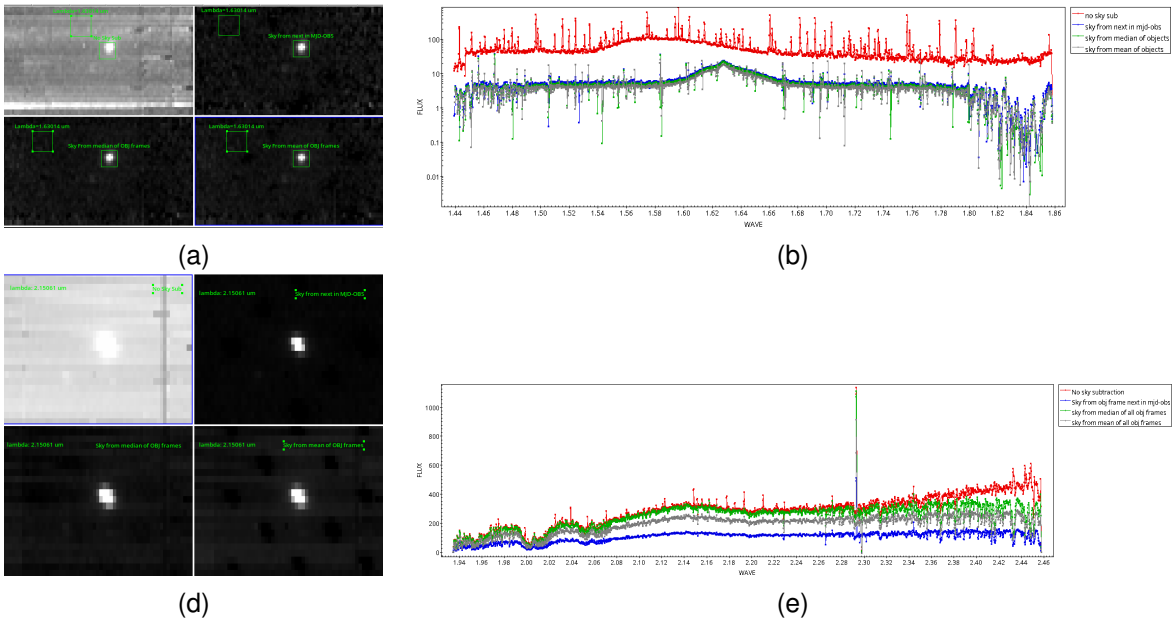


Figure 12.1: Here we show examples of results obtained with different values of the parameter aj-method for observations of the quasar J1502+0257 in H band (top), and the object TWA 25, in K band (bottom), corresponding to a plane at a strong sky line (left) and the corresponding extracted spectra (default extraction, mask_method max, and resampling method DRIZZLE). As we can see for In H band without sky subtraction (upper left of shown image panel) the object is barely visible. In all other cases the object can be well detected, and better results are achievend by aj-method=1. This appears also from the corresponding spectra: in H band (displayed in lin-log scale) in case of no sky subtraction the sky emission lines dominate the continuum, while for other methods are well corrected, better in case of aj-method=1. In case of K band (displayed in lin-lin scale), results are similar with the difference that in this band one notice the strong thermal emission. Again aj-method=1 gives better results.

12.8.5 Quality Control Parameters

`QC.LAMBDA.SHIFT.UM`	[um] OH based center lambda shift
`QC.LAMBDA.SHIFT.PIXEL`	[px] OH based center lambda shift

12.9 eris_ifu_combine_hdrl

12.9.1 Description

The *eris_ifu_combine_hdrl* recipe align and combine SPIFFIER data cubes using the HDRL resampling routine. By default, the routine will use the WCS in each input cube's header to combine the different cubes. Alternatively, users can provide their own offset list in a two-columns (or four-columns) ASCII file and set offset_mode=FALSE for the combination, ignoring the WCS expect for referencing the first OBJECT_CUBE (see Section 8.2 for an example to set offsets).

12.9.2 Input Frames

Frame Tag	Type	Count	Description
OBJECT_CUBE	pro	1 ... n	Science cube(s) to be combined. They should be produced by eris_ifu_jitter recipe; could be OBJECT_CUBE, SKY_TWEAKED_CUBE or DAR_CORRECTED_CUBE
RESPONSE	pro	[0, 1]	option input response, required to flux calibrate the combined data cube.
EXTCOEFF_TABLE	ref	[0, 1]	option input reference atmospheric extinction table, required to flux calibrate the combined data cube.

12.9.3 Product Frames

Default File Name	Frame Tag	Product Depth	Description
eris_ifu_combine_hdrl_obj_cube_coadd.fits	OBJECT_CUBE_COADD	0	Co-added object cube
eris_ifu_combine_hdrl_obj_cube_exposure_map.fits	EXPOSURE_MAP	0	Exposure map
eris_ifu_combine_hdrl_obj_cube_mean.fits	OBJECT_CUBE_MEAN	0	Collapsed object cube

eris_ifu_combine_hdrl_obj_cube_coadd.fits (OBJECT_CUBE_COADD) Combined cube using all OBJECT_CUBE cubes as input.

eris_ifu_combine_hdrl_obj_cube_exposure_map.fits image of the exposure time used on each pixel of the resampled cube.

eris_ifu_combine_hdrl_obj_cube_mean.fits (OBJECT_CUBE_MEAN) Image of the collapsed OBJECT_CUBE_COADD cube calculating the average of each spaxel.

12.9.4 Recipe Parameters

Please see also the description for common recipe parameters in section 12.1 (page 53)

Parameter	Type	Default	Description
offset_mode	bool	False	Offset conventions. If TRUE: the WCS in the input cubes will be used. If FALSE: take user offsets.

continued on next page

continued from previous page

Parameter	Type	Default	Description
name_i	string	offset.list	Input filename. This must be provided if offset_mode is set to FALSE. It allows the user to set X and Y offsets in a two column format; the offsets must be sorted in the same order as the OBJECT_CUBE files in the sof file. The reference point is the cent1er of the first OBJECT_CUBE in the sof file.
offset_unit	string	PIXEL	Offset unit. <PIXEL | ARCSEC | DEGREE| PIXDEG> (see Section 8.2 for an example to set offsets)
method	string enum ['NEAREST', 'LINEAR', 'QUADRATIC', 'RENKA', 'DRIZZLE', 'LANCZOS']	DRIZZLE	Resampling method
method.loop-distance	int	3	Loop distance used by all (but NEAREST) methods to control the number of surrounding voxels that are taken into account. A small value allow faster re-sampling but may not give good quality
method.use-errorweights	bool	False	Use additional weights of $1/err^2$
method.renka.critical-radius	double	1.25	Critical radius of the Renka method
method.lanczos.kernel-size	int	2	Kernel size of the Lanczos method
method.drizzle.downscale-x	double	0.8	Drizzle down-scaling factor in x direction
method.drizzle.downscale-y	double	0.8	Drizzle down-scaling factor in y direction
method.drizzle.downscale-z	double	0.8	Drizzle down-scaling factor in wavelength direction
subtract-background	bool	False	Subtract the background calculated from the median of the 2D image in each channel
subtract-background	bool	FALSE	Subtract median of the images chanel-by-chanel.
extract-source	bool	FALSE	If True try to extract a point souce.
mask_method	string	max	Method to specify extraction mask : mask, position, max, fit or optimal.
center	string	32,32	The centre of the circular mask (pixel).
radius	double	4.0	The radius of the circular mask (pixel).
flux-calibrate	bool	FALSE	If True flux calibrate the extracted spectrum and data cube.
fieldmargin	double	5	Add this margin/border (in percent) to the resampled image/cube
edge-trim	int	0	Number or pixels to trim for each plane of the input frames. It should be smaller than half image size
max-cubes-centres-dist	int	240	Maximum distance between cube centers to build a mosaic. Mosaic creation requires a lot of RAM. Users may trim this value to fit RAM resources.
bpc_iter	int	1	No. of iterations for bad pixel correction.
chop-nan	bool	false	If true chop cube planes with more than 50

12.9.5 Quality Control Parameters

`QC.LAMBDA.SHIFT.UM` [um] OH based center lambda shift
`QC.LAMBDA.SHIFT.PIXEL` [px] OH based center lambda shift

12.10 eris_ifu_combine

12.10.1 Description

The *eris_ifu_combine* recipe align and combine SPIFFIER data cubes using the resampling routine used also in the SINFONI pipeline. By default, the routine will use the WCS in each input cube's header to combine the different cubes. Alternatively, users can provide their own offset list in a two-column ASCII file and set offset_mode=FALSE for the combination, ignoring the WCS expect for referencing the first OBJECT_CUBE. Comparing to `eris_ifu_combine_hdrl` this recipe provides the ability to apply an initial percentile clipping and a consecutive kappa-sigma-clipping (see parameters **pclip, ks_clip, kappa**)

12.10.2 Input Frames

Frame Tag	Type	Count	Description
OBJECT_CUBE	pro	1 ... n	Science cube(s) to be combined. They should be produced by eris_ifu_jitter recipe; could be OBJECT_CUBE, SKY_TWEAKED_CUBE or DAR_CORRECTED_CUBE

12.10.3 Product Frames

Default File Name	Frame Tag	Product Depth	Description
out_cube_coadd.fits	OBJECT_CUBE_COADD	0	Co-added object cube

out_obj_cube_coadd.fits (OBJECT_CUBE_COADD) Combined cube using all OBJECT_CUBE cubes as input.

12.10.4 Recipe Parameters

Please see also the description for common recipe parameters in section 12.1 (page 53)

Parameter	Type	Default	Description
offset_mode	bool	False	Offset conventions. If TRUE: the WCS in the input cubes will be used. If FALSE: take user offsets.

continued on next page

continued from previous page

Parameter	Type	Default	Description
compute_mode	bool	TRUE	Offset conventions. If TRUE: applies reference offset correction. If FALSE: take user offsets. The reference offset is computed as (min_off+max_off)/2.
kernel_type	string	TRUE	Offset conventions. If TRUE: applies reference offset correction. If FALSE: take user offsets. The reference offset is computed as (min_off+max_off)/2.
name_i	string	offset.list	Input filename. This must be provided if offset_mode is set to FALSE. It allows the user to set X and Y offsets in a two column format; the offsets must be sorted in the same order as the OBJECT_CUBE files in the sof file. The reference point is the center of the first OBJECT_CUBE in the sof file.
name_o	string	out_coadd_cube.fits	Output filename.
ks_clip	bool	FALSE	Kappa sigma clipping.
pclip	double	70	Apply an initial percentile clipping based on the absolute deviation from the median if ks_clip=TRUE.
kappa	double	2.0	kappa value for sigma clip.
offset_unit	string	PIXEL	Offset unit. <PIXEL | ARCSEC | DEGREE>
subtract-background	bool	False	Subtract the background calculated from the median of the 2D image in each channel
edge-trim	int	0	Number or pixels to trim for each plane of the input frames. It should be smaller than half image size
weights	string		Optional input filename. The user can specify his own weighting scheme when compute_mode=MEAN A one column format is expected.

12.10.5 Quality Control Parameters

QC.LAMBDA.SHIFT.UM	[um] OH based center lambda shift
QC.LAMBDA.SHIFT.PIXEL	[px] OH based center lambda shift

13 Algorithms

13.1 General Algorithms

13.2 Recipes Algorithms

13.2.1 Dark data reduction

The input dark frames are stacked to generate a master dark frame. Then a map of pixels outliers is determined using the *High Level Data Reduction Library* based algorithms, either on the master dark (method 2D) or using the set of dark frames (method 3D). Then it is determined the master frame read-out noise.

Readout Noise Computation

The CPL function *cpl_flux_get_noise_window* is used to compute the readout noise. This function computes the noise in a frame with MonteCarlo approach.

The provided zone is an array of four integers specifying the zone to take into account for the computation. The integers are specified with the recipe parameters *qc_{ron/fpn}_{x/y}{min/max}* where these coordinates are given in the FITS notation (x from 1 to 2048, y from 1 to 2048).

The algorithm will create *qc_{ron/fpn}_nsamp* windows (with the half size *qc_{ron/fpn}_hsize*) on the frame, scattered optimally using a Poisson law. In each window, the standard deviation of all pixels in the window is computed and this value is stored. The readout noise is the median of all computed standard deviations, and the error is the standard deviation of the standard deviations.

13.2.2 Linearity data reduction

A set of input flat (lamp on and lamp off) frames obtained with increasing DET SEQ1 DIT is analised to determine a map of detector pixels that have a non linear response. The off-frame corrected flat images with intensity below the `pixel_saturation` level are put in a list of images and a polynomial fit of degree `degree` is done to each pixel. Fits with a p-value below the threshold set by the parameter `p-value` are considered bad pixels. Moreover are considered bad pixels also pixels with value smaller than mean - rel-threshold, where the rel-threshold value is set by the parameter `rel-chi-low`, or greater than mean + rel-threshold, where the rel-threshold value is set set by the parameter `rel-chi-high`. Moreover are considered bad pixels also pixels with value smaller than mean - rel-threshold, where the rel-threshold value is set by the parameter `rel-coeff-low`, or greater than mean + rel-threshold, where the rel-threshold value is set set by the parameter `rel-coeff-high`.

13.2.3 North-South (distortion) data reduction

The raw detector image is bent in the x-direction. The North-South recipe generates 32 2D-polynomials (one for each slitlet) to correct for this distortion. Any curvature of the detector in y-direction will be corrected by the wavelength calibration.

To measure distortion, the detector is illuminated through a slit mask (three slits for the 250 and 100 mas scales, one slit for the 25 mas scale), which produces vertical traces on the image. In addition, exposures are taken from the arc lamps. Due to the brick wall pattern of the slits, the arc lines can be used to detect the edges of the slits.

With the distortion correction every pixel will warped to a 64 pixel wide image stripe. The left and right edges of the slitlet (taken from the arc lines) are placed at pixel positions 1 and 64. The vertical traces from the slit mask provide additional reference points in the image. The measured points in the detector image are used to fit a 2D-polynomial to obtain the nominal positions in the corrected image strip.

The final product of the north-south (distortion) data reduction is a FITS file containing the 32 2D-polynomials (in 32 FITS extensions) and an extension specifying the left and right most pixels to extract from the raw detector image.

13.2.4 Flat data reduction

This recipe determines a master flat frame and a map of pixels with very low response (dead pixels).

13.2.5 Wavelength calibration

The wavelength calibration involves the following data reduction steps:

- The input arc line images (WAVE_LAMP, both illuminated and off-lamp frames) are loaded and information on their lamp status (on or off), observed band, preoptic, detector DIT and filename are stored in a table. During this process also a consistency check is performed to make sure frames are of the same band, preoptic and DIT settings. Note that the input frames, usually three taken with the an arc lamp switched on, are obtained with different arc lamps: one with the Argon lamp switched on, one with the Kripton lamp, one with the Neon lamp.
- A reference line list (REF_LINE_ARC) is loaded.
- The optional input master flat (MASTER_FLAT) is loaded.
- If a master flat frame was found the on-off arc frames are divided by the flat.
- The input distortion frame is loaded and using its polynomial distortion coefficients a corrections of the distortion is performed on the arcline frames.
- The slitlets edges position are determined.
- For each IFU slitlet present on the detector a spectrum is determined collapsing the signal along the slitlet extension.
- The arcline positions are determined in a two steps process:
  - first only a few very well isolated arc lines are used to cross correlate them with the ones detected on each of the input arc line frames. Only four or five lines are needed for this first step and the recipes makes sure that both end of the spectra are well covered. The information obtained with the different frames is combined.

  - Once these lines are detected a first approximation of the fitting polynomials (FIRST_WAVE_FIT) can be used to perform a second step where a more accurate fit is performed to all detected arclines from the input line list. A wavelength map (WAVE_MAP) , an image that stores foreach pixel the corresponding wavelength, is determined.

- For quality control the frame is resampled to remove the brick-wall pattern. The corresponding image is an additional quality control product (WAVE_LAMP_RESAMPLED).

13.2.6 OH based Wavelength calibration

To reduce spectral distortions due instrument flexures OH sky lines will be used to recalibrate the wavelength solution.

Since science exposures will typically have integration times of at least a few minutes, the OH sky lines will be bright and clear in individual frames. The processing reconstructs an initial cube from each science frame using the wavelength solution derived from the arc lamp. Then measure the wavelength offset of the frame by comparing the observed wavelengths of the OH lines with respect to their theoretical wavelengths. This offset is folded back into the wavelength solution and the cube reconstructed anew from the raw data and the initial reconstruction is deleted). Thus correcting the spectral flexure does not compromise the quality of the data as does not require additional interpolation steps. As the OH emission comes from high altitude atmosphere layers, the resulting wavelength solution has to be considered in vacuum. The latest version of the pipeline allows to set the polynomial degree of the fit. The default degree has been set to (and should be normally kept at) 0. The reason is that near wavelength band edges, particularly in K band, the correction may diverge, thus the order of the fit should be kept as small as possible. The only case when one might need to increase the order of the fit, for example to 1, is in case of observations of objects near the horizon, that means with strong instrument flexures. In current implementation solution allows now search within a window half width of maximum eigth pixels that should be sufficient for most of the cases. In some previous version (1.5-1.7) of the pipeline the search windows half size was of four pixels and we found a few cases where only increasing the order of the fit one could recover the solution. This is why this release increases to eight the search window size and leaft to the user the possibility to increase the polynomial fit order.

13.2.7 Creation of a 3D data cube

The algorithm described below is executed by recipe eris_ifu_jitter and eris_ifu_stdstar. Given a source image and a corresponding wavelength calibration file, an image is produced in which elements in a given row are associated with a single wavelength. In this way the wavelength shifts between adjacent elements in the rows of the input image are corrected. The output image in the wavelength domain is larger than the input image. Due to the brick wall pattern of the raw frames, some pixels in the first and last few rows have undefined values that are flagged by setting them to NAN. The distribution of these undefined values varies from column to column. The input image is resampled at discrete wavelength intervals using a polynomial interpolation of degree n_coeffs. Different values of the wavelength sampling size and the central wavelength are used for each observed band. Thus, each row has a defined wavelength for each observed band. Since each frame row is now associated with a defined wavelength, each row is used to construct an image which has a defined wavelength. This is done by stacking 32 slitlets on each other, of which each consists of 64 spatial pixels (called also spaxels). Due to the fact that the slitlets length is not exactly 64 pixels and the distance between slitlets is not exactly

64 pixels the edge positions of the slitlets must be known to sub-pixel accuracy. Furthermore, the slitlets must be sorted in the correct sequence to get the correct sequence of the rows in the final images. The centers of each slitlet on the resampled image are determined by averaging the edge positions of the slitlets computed to get the center positions. Then the centers of the slitlets on the raw image are adjusted on the centers of the corresponding raws of the stacked data cube images. Since only integer pixels can be used a sub-pixel error is made at centering, which is stored and used to shift the rows in the reconstructed images of the data cube to the correct sub-pixel position using a user definable method (objnod-fine_tune_mtd). As the edges of the left-most or right-most slitlets may be too near to or fall outside the image margins, then the fitting of the edges may fail. Then the slitlets distances determined must be used to accurately align each slitlet in the final reconstructed image plane.

13.2.8 Correction of the Differential Athmospheric Refraction

Th ERIS-SPIFFIER data reduction pipeline applies a correction of the effects of the atmospheric refraction using *High Level Data Reduction Library* . The remind here the main features of the implemented algorithm.

The effects of atmospheric refraction can be readily seen in any spectrograph or IFU. Atmospheric refraction will displace a source, or its spectrum, by an amount that is dependent on the source wavelength and the angular distance of the source from the zenith. This effect is due to the stratified density structure of our atmosphere, and the displacement will be toward the zenith and will be largest for shorter wavelengths. Because of this latter attribute, differential atmospheric refraction is not generally associated with infrared instruments. However, it can be readily seen in ERIS-SPIFFIER data cubes observed with the largest wavelength coverage (H and K ands) and in the smallest pixel scales (25 mas).

This module uses an analytical approach to compute the expected differential refraction as a function of wavelength, zenith angle, and the refractive index of air which, in turn, depends on temperature, pressure, and water vapour pressure.

The `hdrl_dar` routines require the following inputs (all of which are, generally, available in the input data headers):

- the ambient atmospheric parameters: temperature, pressure, and humidity as contained in the environmental keyword headers: TEL.AMBI.TEMP, TEL.AMBI.PRES.START/END, and TEL.AMBI.RHUM, respectively.
- the instrument rotation angle on the sky
- the parallactic angle of instrument
- and, the world-coordinate system (WCS)

With this input, the differential atmospheric refraction is calculated (optionally, also with an error propagation), and provides an output of the X and Y-axis shifts as a function of the `cpl_vector` with the input wavelengths. The resulting shift corrections can be directly apply to the pixel image in order to correct this effect. The next section describes the algorithms used to make this calculation.

The *High Level Data Reduction Library* functions contains routines to calculate the refractive index of air [5]. The main loop compute the differential refractive offset for the input reference wavelengths, and stores the shift in the coordinates, taking into account the instrument rotation angle on the sky and the parallactic angle at the time of observation.

The differential atmospheric refraction is calculated according to the algorithm from Filippenko (1982, PASP, 94, 715). This algorithm uses the Owens formula which converts relative humidity in water vapour pressure.

Owens saturation pressure This function computes the saturation pressure using the J.C. Owens calibration (1967, Applied Optics, 6, 51-59). The saturation pressure is given by:

$$s_p = -10474 + 116.43\ T - 0.43284\ T^2 + 0.00053840\ T^3 \tag{1}$$

where verb+T+ is the temperature in Kelvins.

Filippenko refractive index At sea level (P=760 `mm Hg` , T = 15 ^oC) the refractive index of dry air is given by (Edlén 1953; Coleman, Bozman, and Meggers 1960):

$$(n(\lambda)_{15,760} - 1)10^6 = 64.328 + \frac{29498.1}{146 - (1/\lambda)^2} + \frac{255.4}{41 - (1/\lambda)^2} \tag{2}$$

where λ is the wavelength of light in vacuum (microns). Since observatories are usually located at high altitudes, the index of refraction must be corrected for the lower ambient temperature and pressure (P) (Barrell 1951):

$$(n(\lambda)_{T,P} - 1) = (n(\lambda)_{15,760} - 1) \cdot \frac{P[1 + (1.049 - 0.0157\ T)10^{-6}\ P]}{720.883(1 + 0.003661\ T)} \tag{3}$$

In addition, the presence of water vapour in the atmosphere reduces $(n - 1)10^6$ by:

$$\frac{0.0624 - 0.000680/\lambda^2}{1 + 0.003661\ T} f \tag{4}$$

where f is the water vapour pressure in mm of Hg, and T is the air temperature in ^oC (Barrell 1951) and is expressed as:

$$f = 0.75006158 \cdot s_p \cdot h \tag{5}$$

where s_p is the saturation pressure of equation 1 and h is the relative humidity in `[%]`.

13.2.9 Sky model

As part of its data processing procedures, the SPIFFIER DRS make use of a sky subtraction technique which is detailed in Davies (2007) [RD3]. The method was developed for SINFONI data but is equally applicable to

[5] See: https://emtoolbox.nist.gov/Wavelength/Documentation.asp#AppendixA, for the formulae used.

any other near infrared data cubes. It is also implemented in the VLT KMOS pipeline (see [RD9] section 7.2.4 and page 97/98). It makes use of the fact that most of the variation on the 1-2.5 μm OH line ratios is due to variations on the vibrational temperature of the OH radical. Fortuitously, to a reasonable approximation, lines from the same vibrational transitions all lie within a well defined wavelength region. Hence one can apply a single scaling across the whole of this region, which is therefore robust against biasing if line emission from the science object occurs at the same wavelength as an OH line. A basic correction can also be made for changes in the rotational temperature, although in this case each spectral segment comprises several short discrete regions. Once the vibrational variations are corrected, any rotational variations can easily be measured and also corrected. While no special mathematical manipulations are required, this procedure is included here because it is not considered a “standard” technique.

1. Identify spaxels with least flux in object cube.
2. Sum spectra from these spaxels in both object and sky cubes separately.
3. Fit a blackbody function to the underlying continuum in the sky spectrum (the thermal background).
4. The fitted function is subtracted from both the original object and sky cubes and from the extracted object and sky spectra.
5. The spectra with removed thermal background are compared with regard to offsets in bright OH lines. The sky cube (with removed thermal background) is shifted accordingly. Note that for SPIFFIER, the default is for spectral flexure to already be corrected. However there may be some situations where this is not so, in which case this step is carried out here.
6. Again the spectrum of the processed object and sky cubes are extracted using the same mask as in step 1.
7. To correct vibrational variations, the spectra are divided into segments along the wavelength axis. For each segment the spectral vectors of bright OH lines are extracted.
8. The sky spectrum is scaled to match the object spectrum in each spectral segment.
9. The scalings of each spectral segment are combined to a single scaling function which is applied to the sky spectrum.
10. To correct rotational variations, steps 7 to 9 are repeated.
11. The two scaling functions are multiplied.
12. The resulting scaling function is multiplied with the compensated sky cube which in turn is subtracted from the compensated object cube.

13.2.10 Cube resampling

The ERIS-SPIFFIER pipeline uses the *High Level Data Reduction Library* based algorithm to resample 3D data cubes. We remind here the main features.

The resampling *High Level Data Reduction Library* project addresses two different scenarios. Assuming that one would like to re-grid and stack 10 dithered images/cubes with pre-determined World Coordinate System (WCS). Then one should be able to:

- Combine all images/cubes in a single cloud of points (we use a table for this step - see below) and when creating the resampled output image/cube all information is used in a single interpolation step, i.e. no image-by-image interpolation is done.
- Resample each individual image/cube on the same output grid (image by image interpolation) and combine the single resampled images in a second step with hdrl stacking methods.

Both scenarios interpolate only once, but the second scenario is important in case not all bad pixels could be properly determined and inserted in the bad pixel mask upfront. Then the implemented HDRL pixel resampling function offers the possibility to use a robust pixel combination method (median or $\kappa\sigma$ clipping) to combine the images/cubes.

Currently there are six different interpolation methods provided by *High Level Data Reduction Library* :

- **Nearest:** Nearest resampling
- **Linear:** Weighted resampling using an inverse distance weighting function
- **Quadratic:** Weighted resampling using a quadratic inverse distance weighting function
- **Renka:** Weighted resampling using a Renka weighting function
- **Drizzle:** Weighted resampling using a drizzle-like weighting scheme
- **Lanczos:** Weighted resampling using a lanczos-like restricted sinc as weighting function

Nearest Neighbour The algorithm does not use any weighting function but simply uses the value of the nearest neighbour inside an output voxel[6] centre as the final output value. If there is no nearest neighbour inside the voxel (but e.g. only outside), the voxel is marked as bad. This speeds up the algorithm considerably. There are no control parameter for this method.

Linear The algorithm uses a linear inverse distance weighting function ($\frac{1}{r}$) for the interpolation. The parameter `loop_distance` controls the number of surrounding pixels that are taken into account on the final grid, e.g. a `loop_distance` of 1 uses 3 pixels (x - 1, x, x + 1) in each dimension, i.e. 9 in total for a 2D image and 27 in total for a 3D cube. Moreover, if the parameter `use_errorweights` is set to TRUE, an additional weight, defined as 1/variance, is taken into account[7].

[6] In 3D computer graphics, a voxel represents a value on a regular grid in three-dimensional space. See *http://https://en.wikipedia.org/wiki/Voxel* for more information.

Quadratic The algorithm uses a quadratic inverse distance weighting function ($\frac{1}{r^2}$) for the interpolation. The parameter `loop_distance` controls the number of surrounding pixels that are taken into account on the final grid, e.g. a `loop_distance` of 1 uses 3 pixels (x - 1, x, x + 1) in each dimension, i.e. 9 in total for a 2D image and 27 in total for a 3D cube. Moreover, if the parameter `use_errorweights` is set to TRUE, an additional weight, defined as 1/variance, is taken into account[7].

Renka The algorithm uses a modified Shepard-like distance weighting function following Renka for the interpolation. The parameter `critical_radius` defines the distance beyond which the weights are set to 0 and the pixels are therefore not taken into account. The parameter `loop_distance` controls the number of surrounding pixels that are taken into account on the final grid, e.g. a `loop_distance` of 1 uses 3 pixels (x - 1, x, x + 1) in each dimension, i.e. 9 in total for a 2D image and 27 in total for a 3D cube. Moreover, if the parameter `use_errorweights` is set to TRUE, an additional weight, defined as 1/variance, is taken into account[7].

Drizzle The algorithm uses a drizzle-like distance weighting function for the interpolation. The down-scaling factors `pix_frac_x`, `pix_frac_y`, and `pix_frac_lambda`, for x, y, and wavelength direction control the percentage of flux of the original pixel/voxel that drizzles into the target pixel/voxel. The parameter `loop_distance` controls the number of surrounding pixels that are taken into account on the final grid, e.g. a `loop_distance` of 1 uses 3 pixels (x - 1, x, x + 1) in each dimension, i.e. 9 in total for a 2D image and 27 in total for a 3D cube. Moreover, if the parameter `use_errorweights` is set to TRUE, an additional weight defined as 1/variance is taken into account[7].

Lanczos The algorithm uses a restricted *sinc* distance weighting function ($\frac{sinc(r)}{sinc(r/kernel\_size)}$) with the kernel size given by the parameter `kernel_size` for the interpolation. The parameter `loop_distance` controls the number of surrounding pixels that are taken into account on the final grid, e.g. a `loop_distance` of 1 uses 3 pixels (x - 1, x, x + 1) in each dimension, i.e. 9 in total for a 2D image and 27 in total for a 3D cube. Moreover, if the parameter `use_errorweights` is set to TRUE, an additional weight defined as 1/variance is taken into account[7].

Note that because *High Level Data Reduction Library* propagates the full data, error and pixel quality information and to perform the resampling step in a general way need to fill an intermediate pixel table with the pixel data, error, pixel quality and corresponding WCS information, this step is demanding in terms of RAM. We could reduce most of the commissioning and early operation data on a computer with 32 GB of RAM. In a few cases, if the user maps a large portion of the FOV, and thus the output cube mosaic may be very large, one may require even more.

Please note that the output is using the Gnomonic projection[8]. The gnomonic projection is from the centre of a sphere to a plane tangential to the sphere. The sphere and the plane touch at the tangent point. This in encoded in the header file by:

```
CTYPE1  = 'RA---TAN'           / Gnomonic projection
CTYPE2  = 'DEC--TAN'           / Gnomonic projection
```

[8]see https://en.wikipedia.org/wiki/Gnomonic_projection

wmin	wmax
1105	1266
1300	1343
1468	1780
1940	1994
2030	2046
2080	2100

Table 13.1: Wavelength [nm] intervals used to evaluate the best telluric model in NIR

- Combine all images/cubes in a single cloud of points (we use a table for this step - see below) and when creating the resampled output image/cube all information is used in a single interpolation step, i.e. no image-by-image interpolation is done.
- Resample each individual image/cube on the same output grid (image by image interpolation) and combine the single resampled images in a second step with hdrl stacking methods.

Both scenarios interpolate only once, but the second scenario is important in case not all bad pixels could be properly determined and inserted in the bad pixel mask upfront. Then the implemented HDRL pixel resampling function offers the possibility to use a robust pixel combination method (median or $\kappa\sigma$ clipping) to combine the images/cubes.

13.2.11 Instrument Response computation

Initially the flux table corresponding to the observed standard star is extracted from an input catalogue. We correct the wavelength scale of the reference spectrum (which is a stellar model spectrum) to the same radial velocity as the observed spectrum and then interpolate the reference spectrum to the same steps as the observed spectrum. We apply a correction of the telluric absorption by looking for the best fitting spectrum within a catalogue of telluric model spectra. The best telluric model is the one that minimises the mean of a “correction” spectrum computed in the following intervals:

This spectrum is obtained via the following steps:

1. adjust telluric model spectrum to wavelength scale of observed spectrum (small shifts may occur due to imperfect wavelength calibration)
2. convolve the telluric spectrum to the same resolution as the observed spectrum
3. divide the observed spectrum by the shifted and convolved telluric spectrum
4. fit the continuum of this ratio at pre-defined wavelength points
5. divide ratio by fit

The response is computed by dividing this telluric-corrected model spectrum of the standard star (in erg $cm^{-2}s^{-1}Å^{-1}$) by the 1D extracted observed spectrum of the standard star corrected for gain, exposure time,

atmospheric extinction (the atmospheric extinction table is interpolated to get the same binning) and for telluric absorption. To reduce the noise of the resulting ratio spectrum we apply a median filter of eleven pixels half width. Then we apply a low order polynomial fit to the median of the signal sampled in a region 4 nm wide around each point defined in the RESP_FIT_POINTS_CATALOG, carefully chosen to sample only the continuum, but also cover the full range of a given spectral band. The fit is repeated to remove by kappa-sigma clip a few sampling points that may be outliers due to noise or presence of lines in the observed to get the final response.

The response is obtained with the following equation:

$$\text{Response}[\text{erg/e}^{-}/\text{cm}^2] = \frac{\text{STD}_{\text{fluxtable}}[\text{erg/s/cm}^2/\text{Å}^{-1}] \times \text{exptime[s]} \times \text{gain}[\text{ADU}/\text{e}-]}{\text{STD}_{\text{observed}}[\text{ADU/pix[Å]}] \times 10^{(0.4 \cdot \text{airmass} \cdot \text{ext})}} \quad (6)$$

Where we have taken into account of the size of a pixel. The response is derived from the order merged flux standard spectrum and applied on the merged science spectrum.

13.2.12 Flux calibration

If the user provides the instrument response and the atmospheric exctinction tables in input of a science recipe, the merged 2D and 1D spectra are then flux calibrated. This operation is performed by first dividing the observed spectra by exposure time and the detector gain and correction for atmospheric extinction, and then multiplying those by the instrument response.

$$\text{I}[\text{erg/s/cm}^2/\text{Å}] = \frac{\text{I}[\text{ADU}/\text{pixel}[\text{Å}]] \times \text{Response}[\text{erg/e}^{-}/\text{cm}^2] \times 10^{(0.4 \cdot \text{airmass} \cdot \text{ext})}}{\text{gain}[\text{ADU}/\text{e}-] \times \text{Exptime[s]} \times \text{bin\_size}} \quad (7)$$

Where we evidence the fact we correct for the size in pixels of the integration bin.

13.2.13 Strehl computation

The central maximum in units of the total flux of a PSF is a measure of the image quality and AO performance. The Strehl ratio is the observed ratio in units of theoretically possible (diffraction limited) ratio. It is a number usually between $\simeq 0.01$ (seeing limited) and 0.6-0.8 depending on the performance and the ambient conditions.

As the ERIS-SPIFFIER possible FOV are either too narrow or too large for most of the calibration stars used to measure the instrument Strehl, it has been defined a special template in which the same PSF standard is observed first in the 0.025 mas scale, to accurately determine the object image Point Spread Function maximum, and the with the 0.100 mas scale to determine the star flux and sky background levels.

Normally the Strehl is determined as ratio:

$$Strehl = \frac{I_{max}/flux}{PSF_{max}/flux_{PSF}}$$

Where I_{max} is the maximum intensity of the object (sky background subtracted), and *flux* the corresponding flux, while PSF_{max} is the maximum intensity of a corresponding theoretical PSF corresponding to a telescope with the same diameters for the primary and secondary mirrors, at a given wavelength and in a given wavelength range, at a given pixel scale, with a given size.

This formula is applied on both images of the PSF standard taken with two camera pixel scales (25 and 100 mas).

The ERIS-SPIFFIER pipeline implements the same algorithm as in the Paranal Multi Strehl Meter GUI.

Initially it is performed a 2D Gaussian fit to the image resulting averaging the STD star cube along the wavelength (z) axis. This fit determines the approximate position of the STD star image, the FWHM along the major and minor axes and the orientation of the elliptic isophotes. Then the sky background is subtracted by taking the median value of the four image plane's corner regions, 8 $\times$ 8 pixels each. Finally it is determined the maximum of the resulting sky corrected image.

The star flux is defined as the flux of the STD star above the sky background.

Then it is built the instrument PSF by taking into account also of the possible anamorphysm and occultation introduced by the secondary mirror and the PSF's maximum and flux are determined.

13.2.14 Standard star position detection

The recipe eris_rec_jitter determines the position of the maximum in the image obtained by collapsing the cube along the z-axis. Then the STD star object (which is assumed to be the only object in the FOV) is more accurately located using a centroid algorithm assuming a 2D Gaussian shape approximation for the object PSF, which also allows to estimate the STD star FWHM along the X and Y directions.

13.2.15 Optimal Extraction

The optimal extraction provided for ERIS is based on the prescription given in Horne (1986; PASP, 95, 609), with some modifications to match it to the pipeline process and adaptive optics data. An outline of the algorithm is given below, and for those interested in further details it is recommended to look over the paper.

A typical use case is to extract a stellar spectrum for telluric or flux calibration. In such cases, the algorithm is effective at correcting bad pixels, and in addition is expected to provide a small increase of 5-20% in the S/N ratio with respect to a simple aperture extraction.

However, there are two things that should be kept in mind. (i) The routine is designed to work on sources for which the spatial distribution changes only gradually with wavelength, e.g. due to atmospheric and adaptive optics effects. So the algorithm will work even if differential atmospheric refraction is not corrected. But, in particular, emission lines with differing distributions to the continuum cannot be handled by this algorithm. In such cases one may wish to try other more complex options such as Schmidt et al. (2019; A&A, 628, A91). (ii) The optimisation uses the noise estimates, which are themselves noisy. Statistically, one can expect 1 in 300 values to be $> 3\sigma$ from its expected value. This means that occasionally a deviant pixel will remain uncorrected. This is natural behaviour, and manual tweaking may be required to correct those instances.

The Horne (1986) paper lists 9 steps in the algorithm. The first three steps are the initial data processing, the initial variance estimates, and fitting the sky background. These steps are performed by the data processing recipes, including subtracting the background. The initial data and variance are therefore taken as fixed input to the optimal extraction routine, and the sky background is assumed to be zero.

The routine used for ERIS begins by picking the brightest pixel in the collapsed image and defining an aperture around it with a radius corresponding to the FWHM of the source so that all pixels containing flux are included. This defines the source values $D_{x\lambda}$ as a function of spatial location x (representing both dimensions) and wavelength λ, and the variance values $V_{x\lambda}$ as the square of the noise. An initial spectrum is created as $f_\lambda^{initial} = \Sigma_x D_{x\lambda}$ with variance $var[f_\lambda] = \Sigma_x V_{x\lambda}$.

A model of the spatial distribution of the source (or PSF) $P_{x\lambda}$ is then constructed by normalising each spectral slice so that $P_{x\lambda} = D_{x\lambda}/f_\lambda$. The resulting model $P_{x\lambda}$ is essentially the probability that a detected photon with wavelength λ falls on pixel x. Because $P_{x\lambda}$ is by definition strictly positive, in a first step any negative values of $P_{x\lambda}$ are set to zero. The second step is to provide some regularisation along the spectral direction, so at each spatial location x the spectral values of $P_{x\lambda}$ are traced. Then, rather than fit these with low order polynomials as was done by Horne (1986), for ERIS AO data it is better to apply a median filter running over a number of pixels defined by the smoothing length. In both cases, the same purpose is achieved: to reject outliers, which is the second core part of the algorithm. Any pixel in $P_{x\lambda}$ for which $(D_{x\lambda} - f_\lambda P_{x\lambda})^2 > \sigma_{clip}^2 V$ is set to zero. Since the atmospheric transmission can vary strongly and rapidly with wavelength (i.e. the source flux can vary somewhat independently of the noise) the threshold σ_{clip} is derived for each spectral slice independently, using a percentile clipping of the values within that slice. The percentile used for this can be set as one of the parameters.

At this point, Horne (1986) updates the variance estimates. However, because the input data are already sky-subtracted we cannot do this. Excluding this step has only a minor impact on the results.

The spectrum estimator is then defined to be a linear combination of unbiased pixel estimates such that

$$f_\lambda^{unbiased} = \frac{\Sigma_x (W_{x\lambda} D_{x\lambda}/P_{x\lambda})}{\Sigma_x (W_{x\lambda})}$$

where the variance of the weighted mean is minimized by choosing weights that are inversely proportional to the variance of the variables, so that

$$1/W_{x\lambda} = var[D_{x\lambda}/P_{x\lambda}] = V_{x\lambda}/P_{x\lambda}^2$$

Substituting these weights into the equation above, one can find the optimal extraction of the spectrum f such that when it is multiplied by the source model P, the result matchs the data D which has variance V:

$$f_\lambda^{optimal} = \frac{\Sigma_x (P_{x\lambda} D_{x\lambda}/V_{x\lambda})}{\Sigma_x (P_{x\lambda}^2 V_{x\lambda})}$$

with variance

$$var[f_\lambda^{optimal}] = \frac{1}{\Sigma_x (P_{x\lambda}^2/V_{x\lambda})}$$

This process is then iterated a second time replacing the initial estimate of f_λ with the first estimate of the optimised spectrum, to yield the final estimate of the optimised spectrum.

A Installation

ESO pipelines can be installed via several methods, depending on your OS, most of which facilitate easy installation, upgrade and removal. Please see the "ESO Data Reduction Pipelines and Workflow Systems" page (https://www.eso.org/pipelines).

A.1 System Requirements

The ERIS-SPIFFIER pipeline implements recipes and algorithms in a very efficient manner. The only system recommendation we would like to give to the user, as the implementation employes OpenMP instructions, is to run the recipe on a system with multi core architecture and enough RAM and disk space. For example the following requirements are recommended:

- 20 (minimum)/(32 or more is better) GB of memory
- 4 (more is better) CPU cores (physical cores)
- 1 TB of free disk space, possibly SSD.
- GCC 8.3.1 (or newer)

In this release we have trimmed the default of the parameter max-cubes-centres-dist of the jitter and stdstar recipes to a value (240) that should allow also users with 20 GB of RAM to run that recipe, eventually not generating some products in case the maximum distance between the centres of the 3D cubes to be combined is greater than that value. However we recommend user to have more RAM.

A.2 Installing the ERIS Pipeline

Installation via RPM or MacPorts is recommended. See Installation instructions for all the details.

In case the user platform is not one for which an RPM or MacPorts are provided the user may use the `install_esoreflex script.`

ESOReflex can be installed as:

```
./install_esoreflex
```

Then follow the instructions on the screen (selecting ERIS). Once the script finishes successfully and the path variables have been set, the installation of the ERIS pipeline ESOReflex workflow and the pipeline are complete.

A.2.1 Build Requirements

There are no particular build requirements for the ERIS pipeline. Installation based on the `install_esoreflex` script, requires the user has installed on her/his desktop certain softwre as describe at:

https://www.eso.org/sci/software/pipelines/installation/software_prerequisites.html.