# EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

# VERY LARGE TELESCOPE

## MATISSE Pipeline User Manual

VLT-MAN-MAT-15860-9310

Issue 2.5.0

Date 06.08.2026

| | Name | Date | Signature |
|---|---|---|---|
| Prepared: | P. Berio | 01.03.2020 | |
| Approved: | S. Lagarde | 01.03.2020 | |
| Released: | B. Lopez | 01.03.2020 | |

This page was intentionally left blank

**Change record**

| Issue/Rev. | Date | Section/Parag. affected | Reason/Initiation/Documents/Remarks |
|---|---|---|---|
| 1.4.1 | 20/05/2019 | All | First release |
| 1.4.2 | 04/06/2019 | 2 | NEWS section added and updated |
| 1.4.3 | 12/06/2019 | 2 | NEWS section updated |
| 1.4.4 | 03/07/2019 | 2, 9.1, 9.21, 9.8, 9.10, 9.7 | |
| 1.4.5 | 09/12/2019 | 9.1, 9.7, 9.19, 11.2.1 | |
| 1.5.0 | 01/04/2020 | 2, 9.1, 9.3, 9.7, 9.9, 9.19, 6, 11.2.1 | |
| 1.5.1 | 06/05/2020 | 2 | NEWS section updated |
| 1.5.2 | 06/06/2020 | 2 | NEWS section updated |
| 1.6.0 | 20/05/2021 | 2 | NEWS section updated |
| 1.7.5 | 20/05/2022 | 2, 11, 9.1 | NEWS updated, GRA4MAT Algo added |
| 1.7.6 | 31/08/2022 | 2, 11.9 | NEWS section updated, BCD calibration added |
| 1.7.7 | 01/04/2023 | 2, E | NEWS section updated, QC parameter added |
| 1.7.8 | 02/04/2023 | 2 | NEWS section updated |
| 2.0.0 | 17/11/2023 | 2 | NEWS section updated |
| 2.0.2 | 15/05/2024 | 2 | NEWS section updated |
| 2.2.0 | 12/03/2025 | 2 | NEWS section updated |
| 2.5.0 | 06/08/2026 | 2 | NEWS section updated |

**List of authors (alphabetical)**

- Fatme Allouche (OCA Nice)
- Philippe Berio (OCA Nice)
- Leonard Burtscher (U Leiden)
- Armin Gabasch (ESO Garching)
- Violeta Gámez-Rosas (U Leiden)
- Matthias Heiniger (MPIfR Bonn)

- Karl-Heinz Hofmann (MPIfR Bonn)
- Jacob Isbell (MPIA Heidelberg)
- Walter Jaffe (U Leiden)
- Stéphane Lagarde (OCA Nice)
- Alexis Matter (OCA Nice)
- Florentin Millour (OCA Nice)
- Dieter Schertl (MPIfR Bonn)
- József Varga (U Leiden)

This page was intentionally left blank

# Contents

**1 Introduction** **11**

1.1 Purpose . . . 11

1.2 Acknowledgements . . . 11

1.3 Scope . . . 11

1.4 Reference and applicable documents . . . 11

**2 What's new** **14**

2.1 What's new in the latest pipeline release . . . 14

2.2 What's new in pipeline release 2.3.0 . . . 14

2.3 What's new in pipeline release 2.2.0 . . . 14

2.4 What's new in pipeline release 2.0.2 . . . 15

2.5 What's new in pipeline release 2.0.0 . . . 15

2.6 What's new in pipeline release 1.7.8 . . . 15

2.7 What's new in pipeline release 1.7.7 . . . 15

2.8 What's new in pipeline release 1.7.6 . . . 15

2.9 What's new in pipeline release 1.7.5 . . . 16

2.10 What's new in pipeline release 1.6.0 . . . 16

2.11 What's new in pipeline release 1.5.2 . . . 16

2.12 What's new in pipeline release 1.5.1 . . . 17

2.13 What's new in pipeline release 1.5.0 . . . 17

2.14 What's new in pipeline release 1.4.9 . . . 17

2.15 What's new in pipeline release 1.4.4 . . . 17

2.16 What's new in pipeline release 1.4.3 . . . 18

2.17 What's new in pipeline release 1.4.2 . . . 18

2.18 What's new in pipeline release 1.4.1 . . . 18

**3 Overview** **19**

**4 MATISSE Instrument Description** **20**

4.1 Instrument overview . . . 20

4.2 Observing Mode . . . 21

**5 Quick start** **25**

5.1 MATISSE pipeline recipes . . . 25

5.1.1 Instrument Calibrations . . . 25

5.2 An introduction to Reflex, Gasgano and EsoRex . . . 25

5.2.1 Using Gasgano . . . 26

5.2.2 Using EsoRex . . . 30

**6 Known Problems** **33**

6.1 Open issues . . . 33

**7 Instrument Data Description** **34**

7.1 RAW science data . . . 35

7.2 RAW calibration data . . . 35

7.3 STATIC calibration . . . 36

7.4 PRODUCT calibration data . . . 37

7.5 PRODUCT science data . . . 38

**8 Data Reduction** **39**

8.1 Graphical overview of the cascade . . . 39

8.2 Data reduction recipes . . . 39

**9 Pipeline Recipe Interfaces** **41**

9.1 mat_raw_estimates . . . 41

9.1.1 Input . . . 41

9.1.2 Output . . . 42

9.1.3 Quality control . . . 42

9.1.4 Options . . . 42

9.2 mat_est_flat . . . 44

9.2.1 Input/Output files and Options . . . 44

9.3 mat_est_shift . . . 45

9.3.1 Input/Output files and Options . . . 45

9.4 mat_est_kappa . . . 46
9.4.1 Input/Output files . . . 46
9.5 mat_cal_oifits . . . 46
9.5.1 Input/Output files and Option . . . 47
9.6 mat_cal_imarec . . . 47
9.6.1 Input and Output files . . . 51
9.6.2 Options . . . 51
9.6.3 Content of the output file . . . 53
9.7 mat_cal_image . . . 56
9.7.1 Input and Output files . . . 57
9.7.2 Options . . . 58
9.7.3 Calibration Maps . . . 59
9.8 mat_ext_beams: Extraction of the Photometry . . . 59
9.8.1 Input and Output files . . . 59
9.8.2 Options . . . 60
9.9 mat_est_corr . . . 60
9.10 mat_est_opd . . . 61
9.11 mat_proc_incoher . . . 61
9.12 mat_proc_coher . . . 61
9.13 mat_merge_results . . . 62
9.14 mat_cal_cphase . . . 62
9.15 mat_cal_dphase . . . 63
9.16 mat_im_basic . . . 63
9.16.1 Input files . . . 64
9.16.2 Output files . . . 64
9.16.3 Options . . . 64
9.17 mat_im_extended . . . 65
9.17.1 QC1 parameters . . . 66
9.17.2 Input files . . . 66
9.17.3 Output files . . . 67

9.17.4 Options . . . 67
9.18 mat_im_rem . . . 67
9.19 mat_cal_det . . . 68
9.19.1 Input files . . . 72
9.19.2 Output files . . . 72
9.19.3 Options . . . 73
9.20 mat_est_aphase . . . 73
9.20.1 Input files . . . 74
9.20.2 Output files . . . 75
9.20.3 Options . . . 75
9.21 mat_est_tf . . . 76
9.22 mat_cal_vis . . . 76

**10 Additional Tools** **78**

**11 Algorithms** **79**
11.1 Input data . . . 79
11.2 General Algorithms . . . 79
11.2.1 Estimating BPM, FFM, nonlinearity map . . . 79
11.2.2 Estimating Shift map, kappa-matrix, shift and zoom coefficients . . . 80
11.2.3 Applying cosmetics . . . 81
11.3 Demodulation . . . 81
11.4 OPD calculation . . . 82
11.5 Incoherent Processing . . . 82
11.6 Coherent Processing . . . 82
11.7 Fringe Jumps Flagging with GRA4MAT . . . 83
11.8 N band coherent processing with GRA4MAT . . . 84
11.9 BCD calibration . . . 84

**A Installation** **86**

**B Abbreviations and acronyms** **87**

**C Troubleshooting Guide** **88**

**D Additional Tools** **89**

D.1 Automatic Pipeline Python Script . . . . . 89

D.2 Image Reconstruction Script . . . . . 89

**E Additional QC paramters** **96**

E.1 Recipe: All recipes . . . . . 96

E.2 Recipe: mat_est_kappa . . . . . 97

E.2.1 Computation of QC.DELTA.SHIFT . . . . . 97

E.3 Recipe: mat_raw_estimates . . . . . 97

# 1 Introduction

## 1.1 Purpose

The MATISSE 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 for the *Paranal Science Operations* (PSO), in the quick-look assessment of data, in the generation of master calibration data, in the reduction of scientific exposures, and in the data quality control. Additionally, the MATISSE pipeline recipes are made public to the user community, to allow a more personalised processing of the data from the instrument. The purpose of this document is to describe a typical MATISSE data reduction sequence with the MATISSE pipeline.

This manual is a complete description of the data reduction recipes implemented by the the MATISSE pipeline, reflecting the status of the MATISSE pipeline as of version 2.5.0

## 1.2 Acknowledgements

## 1.3 Scope

This document describes the MATISSE pipeline used at ESO in Garching and on Paranal for the purpose of data assessment 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 [12]. For general information about the current instrument pipelines status we remind the user of [11]. Quality control information are at [6]. Additional information on CFITSIO, the Common Pipeline Library (CPL) and EsoRex can be found respectively at [22], [4], [8]. The Reflex and Gasgano front-ends are described in [15] and [9] and a description of the instrument can be found in [10]. The MATISSE instrument user manual is in [13], the calibration plan in [20].

Additional information on the DFS and VLT data interfaces are in [16], [5], [7].

## 1.4 Reference and applicable documents

[1] L. Bourgés, S. Lafrasse, G. Mella, O. Chesneau, J. L. Bouquin, G. Duvert, A. Chelli, and X. Delfosse. The JMMC Stellar Diameters Catalog v2 (JSDC): A New Release Based on SearchCal Improvements. In N. Manset and P. Forshay, editors, *Astronomical Data Analysis Software and Systems XXIII*, volume 485 of *Astronomical Society of the Pacific Conference Series*, page 223, May 2014. 36

[2] Philip E. Ciddor. Refractive index of air: new equations for the visible and near infrared. *Appl. Opt.*, 35:1566, Mar 1996. 83

[3] Gilles Duvert, John Young, and Christian A. Hummel. OIFITS 2: the 2nd version of the data exchange standard for optical interferometry. *A&A*, 597:A8, Jan 2017. 34

[4] ESO, *http://www.eso.org/cpl/*. *CPL home page*. 11

[5] ESO. *DFS Pipeline & Quality Control – User Manual*. VLT-MAN-ESO-19500-1619. 11

[6] ESO, *http://www.eso.org/observing/dfo/quality/*. *ESO-Data Flow Operation home page*. 11

[7] ESO. *ESO DICB – Data Interface Control Document*. GEN-SPE-ESO-19400-0794 (3.0). 11

[8] ESO, *http://www.eso.org/cpl/esorex.html*. *ESOREX home page*. 11, 19

[9] ESO, *http://www.eso.org/gasgano/*. *Gasgano User's Manual*. VLT-PRO-ESO-19000-1932. 11, 19

[10] ESO, *http://www.eso.org/sci/facilities/paranal/instruments/matisse/overview.html*. *Matisse instrument home page*. 11

[11] ESO, *http://www.eso.org/observing/dfo/quality/pipeline-status.html*. *Matisse Pipeline Current Status*. 11

[12] ESO, *http://www.eso.org/pipelines*. *Matisse Pipeline Web Page*. . 11

[13] ESO, *http://www.eso.org/sci/facilities/paranal/instruments/matisse/doc.html*. *Matisse User Manual*. 11

[14] ESO. *Reflex MATISSE tutorial*. VLT-MAN-ESO-19500-????? 25

[15] ESO, *http://www.eso.org/sci/software/reflex/*. *Reflex User Manual*. VLT-SPE-XXX-XXX. 11, 19

[16] ESO. *VLT Data Flow System Specifications for Pipeline and Quality Control*. VLT-SPE-ESO-19600-1233. 11

[17] J. A. Gordon and D. F. Buscher. Detection noise bias and variance in the power spectrum and bispectrum in optical interferometry. *A&A*, 541:A46, May 2012. 82

[18] W. J. Jaffe. Coherent fringe tracking and visibility estimation for MIDI. In W. A. Traub, editor, *New Frontiers in Stellar Interferometry*, volume 5491 of *Proc. SPIE*, page 715, October 2004. 83

[19] Richard J. Mathar. Refractive index of humid air in the infrared: model fits. *Journal of Optics A: Pure and Applied Optics*, 9:470–476, May 2007. 83

[20] MATISSE Consortium. *MATISSE Instrument Calibration Plan*. VLT-TRE-MAT-15860-9303. 11

[21] F. Millour, P. Berio, M. Heininger, K.-H. Hofmann, D. Schertl, G. Weigelt, F. Guitton, W. Jaffe, U. Beckmann, R. Petrov, F. Allouche, S. Robbe-Dubois, S. Lagarde, A. Soulain, A. Meilland, A. Matter, P. Cruzalèbes, and B. Lopez. Data reduction for the MATISSE instrument. In *Optical and Infrared Interferometry and Imaging V*, volume 9907 of *Proc. SPIE*, page 990723, August 2016. 79

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

[23] A. Schutz, E. Thiebaut, F. Soulez, M. Tallon, G. Duvert, and I. Tallon-Bosc. Effective a posteriori cophasing of interferometric fringe data. In F. Malbet and J. Creech-Eakma,, editors, *Optical and Infrared Interferometry and Imaging V*, volume 9907 of *SPIE*, page 99073D, August 2016. 82

[24] B. Wirnitzer. Bispectral analysis at low light levels and astronomical speckle masking. *Journal of the Optical Society of America A*, 2:14–21, January 1985. 82

# 2 What's new

## 2.1 What's new in the latest pipeline release

- Group-delay estimation and phase-wrapping correction have been improved (FFT-based algorithm)
- Chromatic OPD correction now uses the Voronin & Zheltikov (2017) refractive-index model instead of the previous Mathar model; the former `ChromaticOpdFit` option has been removed. K4N OPD estimation for N-band GRA4MAT data is controlled by the `k4n` parameter
- Recipe parameters have been renamed: `spectralBinning` to `spectralAverage` and `coherentAlgo` to `cpAlgo`. `spectralAverage` is a sliding average of the dispersed interferograms in the spectral direction. `cpAlgo` selects the closure-phase / incoherent estimation algorithm (1: Bispectrum, default; 2: Coherent). With `cpAlgo=1`, closure phases are computed from the bispectrum rather than from differential phases
- For GRA4MAT observations, new frame-filtering and VIS2-correction options have been added: `filter` (`vf`, `pf`, `jp`; default `vf,pf,jp`) can reject frames affected by visibility loss, photometric flickering or fringe jumps; `vfactor` and `pfactor` (L/M band) correct VIS2 for phase jitter and flux flickering using the GRAVITY fringe-tracker data. Corresponding QC keywords `ESO QC VFACTORi AVG` and `ESO QC PFACTORi AVG` are written to the products
- When `cumulBlock=TRUE`, chopped photometry is averaged across BCD blocks and flux errors are combined in quadrature
- OI_T3 T3AMP/T3AMPERR are now derived from VIS2 (with error propagation) instead of being left at zero
- Phase 3 / OIFITS2 product keywords, provenance and QC parameter propagation have been fixed and improved
- EDPS and Reflex workflows have been updated for the new recipe parameters and QC handling
- Obsolete Gasgano, Data Organiser and RBS configuration files have been removed; the operational system is now EDPS-based
- The HDRL library has been updated to version 1.6.0
- Various recipe robustness fixes have been applied, including failing when no products are created

## 2.2 What's new in pipeline release 2.3.0

- EDPS workflow and QCReport plots have been updated
- Recipes now return a failure if 0 products are created

## 2.3 What's new in pipeline release 2.2.0

- EDPS workflows and adari reports have been added

## 2.4 What's new in pipeline release 2.0.2

- An error in the calculation of the Fourier transform of the fringe patterns has been fixed. Specifically, the algorithm that rolls the fringe pattern in the spatial direction by half the image width was off by 1 pixels for half the image when the image size is odd. This has been corrected.

## 2.5 What's new in pipeline release 2.0.0

- As of Matisse pipeline version 1.5.2, the convention for phases and closure phases for the N band was inconsistent with the convention used for the L and M bands. This caused a 180° rotation of the N-band images and models relative to the North-up and East-to-the-left standard. This has now been corrected beginning with the current pipeline version 2.0.0.

## 2.6 What's new in pipeline release 1.7.8

- A problem/segmentation violation in the computation of the errors has been fixed. If only one frame is left after flagging, a warning is issued and the error is set to FLT_MAX (i.e. a very high number), as the bootstrap routine can not be used in this case to properly give an error estimation.

## 2.7 What's new in pipeline release 1.7.7

- The pipeline and workflows have been updated to work with the new cpl version 7.3 and newer python versions, respectively
- The pipeline code and the workflow plotting routines have been updated to avoid problems with NaN pixels.
- New Quality Control (QC) parameters were added to various recipes. See section E for details.

## 2.8 What's new in pipeline release 1.7.6

- The software to download the latest version of the Earth Orientation Parameter as a function of time (`MJD-OBS`) from IERS has been changed. This was necessary as not only the link but also the download protocol changed from ftp to https
- The BCD calibration has been added to the *mat_cal_oifits* recipe. It is activated if the recipe parameter `cumulBlock` is set to `TRUE` (current default value). See section 11.9 for details.
- The matisse_viscal workflow and the launched mat_cal_oifits recipe has been changed
  - A bug in the file naming scheme of the recipe has been fixed. The bug caused an overwriting of files in some circumstances.
  - In the workflow the maximum associated night–time calibration files have been increased from 10 to 12.

– The recipe parameter `--cumulBlock` has been set to TRUE in the recipe as well as in the workflow. If several BCD combinations are included in the input files, the `--cumulBlock=TRUE` triggers additional outputs where the BCD signature in the files has been removed. These files can be identified by the header keyword `DRS.BCD.COMBINED` set to T (TRUE).

## 2.9 What's new in pipeline release 1.7.5

- Adding QC parameters.
- Adding a new processing dedicated to GRA4MAT Observations (K4N coherent processing).
- Small change for SHITF_MAP in LOW resolution L-band
- Adding a check for the presence of a KAPPA_MATRIX in the mat_ext_beams recipe.

## 2.10 What's new in pipeline release 1.6.0

- Optimization of the Distorsion Map for the MED and HIGH resolution in L band.
- Fixing a header inheritance problem
- Fixing a bug in mat_merge_results affecting the esoreflex workflow
- Fixing a problem related to a wrong keyword PRO.DATANCOM
- Adding parallel section by using OpenMP on linux platforms
- Fixing an out of boundary access

## 2.11 What's new in pipeline release 1.5.2

- A wrong wavelenth range when computing the following QC parameters on the dispersed fringes was fixed:

```
QC.DETi.TFSQRj.MEAN
QC.DETi.VISSQRj
QC.DETi.VISSQRj.SNR
QC.DETi.CPHASEj
QC.DETi.CPHASEj.STDEV
QC.DETi.DPHASEj
QC.DETi.DPHASEj.STDEV
```

## 2.12 What's new in pipeline release 1.5.1

- Default value for recipe parameter `excess_count_n` and `excess_count_lm` (recipe mat_raw_estimates and mat_cal_image) corrected. Moreover the option `taryp` now works for both recipes.
- Estimating a possible TARTYP shift uses now the whole interferometric channel. It is possible to change TARTYP U with S/T even if the estimated or specified shift is 0.
- `tartyp=57` is now the default value whatever the band (L/M or N)
- FLAG column of OIVIS2, OIVIS, OIT3 and IOFLUX is now filled wrt the wavelength (L/M band: If ($2.8\mu m$ < lambda < $4.2\mu m$ or $4.48\mu m$ < lambda < $5.2\mu m$): flag=FALSE else flag=TRUE and N band: If ($8\mu m$ < lambda < $9.18\mu m$ or $9.56\mu m$ < lambda < $13\mu m$): flag=FALSE else flag=TRUE)
- QC parameters are now computed on the same wavelength range as in the previous item

## 2.13 What's new in pipeline release 1.5.0

- In this version, the pipeline could process data recorded in the Very High Resolution in L and M band.
- You can now use the pipeline to calibrate the correlated flux of the science by using the correlated flux of the calibrator. The results (OI_VIS table) are not calibrated from the flux of the calibrator. This step should be done outside the pipeline.

## 2.14 What's new in pipeline release 1.4.9

- New disperion laws in L/M band have been computed. We used observations of Be stars with GRA4MAT to improve the spectrum of plastic foils. Now the laws are modelized with a polynome of order 4. A new parameter (–obsCorrection) has been added to mat_est_shift. You have to select –obsCorrection=TRUE in order to use the new laws.
- Correction of the bad tagging (S,T,U) of the raw frames. The correction would be specified with the parameter –tartyp of mat_raw_estimates. See man-page for a full description.
- We implement spectral binning in LM and N bands. The width in pixel (odd number) of the spectral binning could be specified with the parameter –spectralBinning of mat_raw_estimates. We recommand spectralBinning=5 in L/M band and spectralBinning=7 in N band.
- TARTYP correction method improved, uses up to three methods for determining a possible shift (more flags available for –tartyp) and allows to specify the shift for a subset of exposures in a SOF (–excess_count_list).

## 2.15 What's new in pipeline release 1.4.4

- The following unused recipe parameters were removed:
  - For recipe mat_raw_estimates and mat_est_tf: diamStar, diamErr, Catalog
  - For recipe mat_raw_estimates and mat_ext_beams: useKappaMatrix, useAvgShape, useAvgSky

  - For recipe mat_raw_estimates and mat_est_opd: chromaticOpdFit
  - For recipe mat_raw_estimates and mat_cal_image: binning

- A temporal hampel filter was added to the recipes mat_raw_estimates and mat_ext_beams. The temporal hampel filter was added in order to reduce some extra detector noise (HAWAII-2RG detector) which affects the photometry estimation for low flux stars (< 5JY with ATs). The name of the new parameter (for mat_raw_estimates and mat_ext_beams) is "hampelFilterKernel".

```
--hampelFilterKernel: Only for L/M band. Apply a temporal Hampel
filter to all pixels before deriving the photometry. This filter
improves the photometric estimation in case of faint stars (< 5Jy in
L band with AT). The parameter fixes the size of the kernel of the
filter (kernel=0 is recommended).
```

## 2.16 What's new in pipeline release 1.4.3

- Check for INS.TIMDL.START and INS.TIMDN.START header keywords added. If these keywords were missing, the pipeline aborted. This has been corrected.

## 2.17 What's new in pipeline release 1.4.2

- Some problems in the N band photometry estimation of the mat_ext_beams recipe have been fixed: In N band the pipeline applies a Gaussian fit of the photometry in each spectral channel. In some (rare) cases the fit did not converge. If the fit is not converging, the photometry is equal to zero.
- Estimate and use an average non-linearity response for the LM-Band detector (mat_cal_det recipe) if requested.
- All combinations of cost function and algorithm mode in mat_cal_imarec recipe are now supported. Minor errors fixed for mat_cal_imarec. Flux rescaling for calibrated input is implemented.

## 2.18 What's new in pipeline release 1.4.1

- First release to the public

# 3 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* [15], *Gasgano* [9] and *EsoRex* [8] The last two are included in the pipeline distribution (see Appendix A, page 86). These applications can also be downloaded separately from `www.eso.org/reflex`, `www.eso.org/gasgano`, `www.eso.org/cpl/esorex.html` respectively.

An illustrated introduction to Gasgano is provided in the "Quick Start" Section of this manual (page 25).

The MATISSE instrument and the different types of MATISSE raw frames and auxiliary data are described in Sections 4, 7.

A brief introduction to the usage of the available reduction recipes using Gasgano or EsoRex is presented in Section 5. In section 6 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 8.

More details on what are inputs, products, quality control measured quantities, and controlling parameters of each recipe is given in section 9.

More detailed descriptions of the data reduction algorithms used by the individual pipeline recipes can be found in Section 11.

In Appendix A the installation of the MATISSE pipeline recipes is described and in Appendix B a list of used abbreviations and acronyms is given.

# 4 MATISSE Instrument Description

MATISSE has been developed by the MATISSE consortium which consists of the main partners OCA/LAGRANGE, NOVA, MPIA and MPIfR. The instrument has been made available to the community and started operations on Paranal in period 103. In this chapter a brief description of the MATISSE instrument is given.

## 4.1 Instrument overview

The Multi-Aperture mid-Infrared SpectroScopic Experiment (MATISSE) is a 4-way beam-combiner instrument for the ESO VLTI, designed to be sensitive from the L to the N band. It produces dispersed fringes (Fig. 4.1) on two different detectors simultaneously, the HAWAII-2RG for the L&M bands and the AQUARIUS for the N band.

MATISSE measures the coherent flux, visibilities, closure phases and differential phases. Differential visibilities can also be derived. These quantities are measured as a function of the wavelength in the selected spectral bands and resolutions.

The instrumental sensitivity, sampling and throughput are optimized for the L and N bands. The L band is specified from 2.8 to 4.2 $\mu$m and the N band from 8.0 to 13.0 $\mu$m. MATISSE operates also in M band, from 4.5 to 5.0 $\mu$m. The L, M and N bands can be observed simultaneously. The spectral separation of the L&M band from the N band is achieved in the warm optics using a dichroic beam splitter which reflects the L and M band light and transmits the N band light, see Fig. 4.2. The instrument can be used with a number of spectral resolutions. Two spectral resolutions are possible in the N band (R $\approx$ 30, R $\approx$ 220) and four different resolutions are available in the L&M bands (R $\approx$ 30, R $\approx$ 500 for L and M, R $\approx$ 950 for L only, R $\approx$ 3000 at 4.05$\mu$m – Br$\alpha$ – and at 4.65$\mu$m – CO lines). The interferometric pattern and the photometric signals are spectrally dispersed using grisms or prisms (Fig. 4.2) located in the cryostats.

MATISSE is a four-beam multi-axial combiner with two different modes: *SiPhot* mode and the *HighSens* mode. In *SiPhot* mode, the interferometric beam and the photometric beams receive respectively 2/3 and 1/3 of the incoming flux thanks to a beam splitter (Fig. 4.2). For an observation with four telescopes with photometric channels, 5 images are produced on the detector: four photometric channels and the interferometric one (Fig. 4.1). In *HighSens* mode, the beam splitter is replaced by a mirror, no photometry is recorded, and all photons are collected in the interferometric beam.

During observations with 4 telescopes, the interferogram contains a pattern with 6 fringe structures and is dispersed in the spectral direction. The spatial size of this interferometric channel is larger than the photometric ones in order to optimize the sampling of the 6 fringe periods and the flux received per pixel. The superposition of the 4 individual beams that form the interferometric beam is achieved by the camera optics. At this level, the beam configuration is non-redundant (the separation $B$ between beams is $3D$, $9D$, and $6D$ where $D$ is the spatial diameter of the beam) in order to avoid cross-talk between the fringe peaks in Fourier space. The Fourier transform of each spectral column of the interferometric image is thus composed of 6 fringe peaks centred at different frequencies $B_{ij}/\lambda$ (3 D/$\lambda$, 6 D/$\lambda$, 9 D/$\lambda$, 12 D/$\lambda$, 15 D/$\lambda$, 18 D/$\lambda$), and a low frequency peak that contains the object photometry and the thermal background coming from the four telescopes. To measure the coherent fluxes with a good accuracy, the design of MATISSE is based on the use of spatial filters (figure 4.2), including image and pupil stops inside the cryostats. To measure closure and differential phases with good accuracy, a beam commutation (Fig. 4.2) is made at the entrance of the instrument. This allows us to reduce

instrumental effects on the obtained signal. The commutation is made between IP1 and IP3 and/or between IP5 and IP7 (IP means *InPut beam*). Thus, there are four configurations of the two beam commuting devices (BCDs): BCD OUT - OUT, BCD IN - IN, BCD IN - OUT, BCD OUT - IN.

Motorized delay lines and periscopes allow the co-alignment and the cophasing between the LM and the N parts of the instrument. These modules can be used also to correct the chromatic differences (atmospheric refraction or chromatic OPD) between the 2 bands. MATISSE has an imaging mode (2D image observation without dispersion) for the target acquisition. In this case, the instrument uses spectral filters inside the cryostat instead of the prisms or the grisms. In addition, some devices such as artificial sources, the lens for pupil imaging and special material for spectral calibration are implemented in the instrument to perform alignment, test, maintenance, calibration operations.

To measure the coherent fluxes and all the derived interferometric measures such as the differential visibility, differential phase and the closure phase, the key problem is to eliminate cross-talk between the low frequency peak and all the other peaks that introduce sensitivity of the fringe peaks to variations of the thermal background. Two methods are used in MATISSE to ensure this result and estimate the coherent flux with high accuracy: spatial OPD modulation (as it is done in AMBER), and temporal OPD modulation (as in MIDI). The rejection of the thermal background from the correlated flux is thus based on two methods in the multi-axial scheme used by MATISSE:

- **Fourier filtering**, as in AMBER: the background low frequency signal is concentrated in the central low frequency peak while the fringe signals are contained in 6 fringe peaks at the frequencies $nD/\lambda$, with $n = 1, 2, ..., 6$. (see figure 4.3)
- **Optical modulation**, as in MIDI, of the input beams made by piezos just after the LM and N bands separation, providing a specific temporal signature of each of the 6 pairs of beams and fringe peaks and thus allowing, thanks to a demodulation process, to reject the contribution of the background continuum (and cross-talk).

To measure the visibility, we need to extract the source photometry by separating the stellar flux from the sky background using sky chopping. However, the optical paths of the sky and the target are not the same for the two positions of the VLT M2 assumed in the chopping sequence. The chop-subtraction therefore leads to residuals which are the most important contribution to the photometric error (and to the resulting visibility error). Fortunately, chopping is unnecessary to measure the coherent flux, as well as the differential and closure phases.

## 4.2 Observing Mode

The different behaviour of the atmosphere, the different strengths of the thermal background, and the different detectors for the LM and the N bands require different observing and background removal strategies for the observations in the two bands.

In the LM band, tip-tilt jitter combined with Strehl ratio variations yield photometric fluctuations that are the dominant cause for photometric error. Thus, it is required to simultaneously record the interferometric and photometric quantities, i.e. use the *SiPhot* mode.

In the N band, the tip-tilt and Strehl variations will yield photometric fluctuations smaller than the uncertainty due to the background fluctuations. In addition, the detector defaults (especially at the stripe border level)

prevent a good accuracy for the photometric measurement if this measurement is not made with the same pixels as for the interferometric channel. Hence the *HighSens* mode is always usable in the N band with sequential observations of photometry.

We call this mode (i.e. *SiPhot* mode for LM band and *HighSens* for N band) *Hybrid* mode. The Observing Sequence is composed of 14 exposures of around 1 minute each detailed in the Tab. 4.1.

| Exposure Number | LM Band Observations | N Band Observations |
|---|---|---|
| A: Sky observations<br>No chopping, Telescope Offset, No modulation | | |
| 1 | BCD IN - IN | BCD IN - IN |
| 2 | BCD OUT - OUT | BCD OUT - OUT |
| B: Interferometric observations<br>No chopping, OPD modulation in LM and N | | |
| 3 | BCD IN - IN | BCD IN - IN |
| 4 | BCD OUT - IN | BCD OUT - IN |
| 5 | BCD IN - OUT | BCD IN - OUT |
| 6 | BCD OUT - OUT | BCD OUT - OUT |
| C: Photometric observations<br>Chopping, LM: Interferometry with modulation, N: Sequential Photometry without modulation | | |
| 7 | BCD IN - IN | BCD IN - IN / Shut1 open (IP3) |
| 8 | BCD IN - IN | BCD IN - IN / Shut2 open (IP1) |
| 9 | BCD IN - IN | BCD IN - IN / Shut3 open (IP7) |
| 10 | BCD IN - IN | BCD IN - IN / Shut4 open (IP5) |
| 11 | BCD OUT - OUT | BCD OUT - OUT / Shut1 open (IP1) |
| 12 | BCD OUT - OUT | BCD OUT - OUT / Shut2 open (IP3) |
| 13 | BCD OUT - OUT | BCD OUT - OUT / Shut3 open (IP5) |
| 14 | BCD OUT - OUT | BCD OUT - OUT / Shut4 open (IP7) |

Table 4.1: MATISSE standard *Hybrid* Observing Sequence

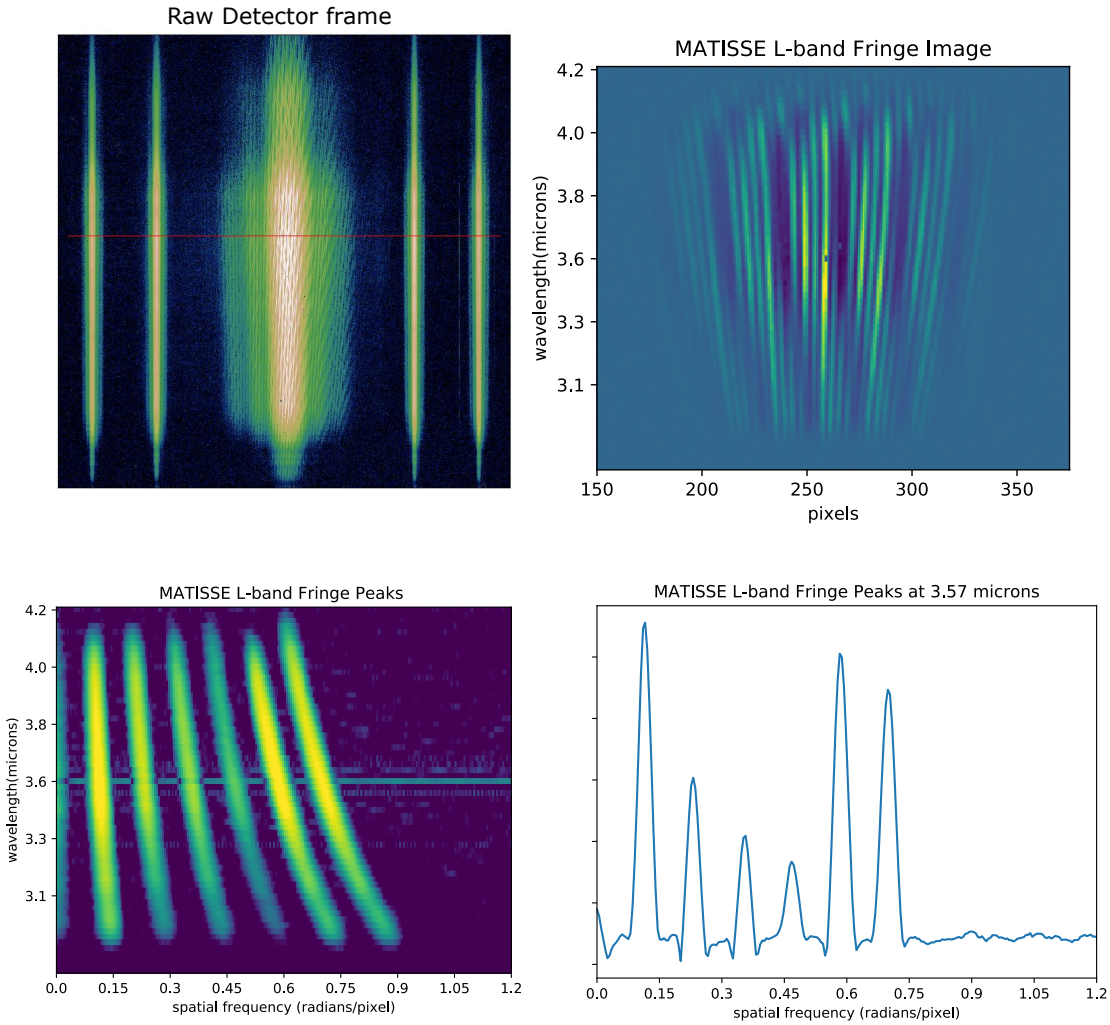


Figure 4.1: Layout of the interferometric pattern on the detector and overview of the fringe processing. *Top left*: raw detector image (central detector blocks, dispersion direction is vertical; rotated 180 deg with respect to the pixel count on the detector) and the four photometric signals on one of the two MATISSE detectors (the HAWAI2RG detector in L band), in *SiPhot* mode. *Top right*: A Single HAWAII-2RG frame of L-band data for a strong calibrator, showing the overlapping fringe patterns of the six interference patterns from all four telescopes. Because of diffraction effects, the spread in the x-direction is proportional to wavelength. The highest spatial frequency fringe is by design one fringe/4 pixels at 3.2 microns. *Lower left*: The two-dimensional Fourier Transform (in the x-direction) of the previous image, showing the clean separation of the six interference patterns by spatial frequency. Note that the spatial frequencies now increase inversely with wavelength. *Lower right*: A cut through the previous figure at 3.57 microns, showing the six distinct peaks for the different telescope combinations.

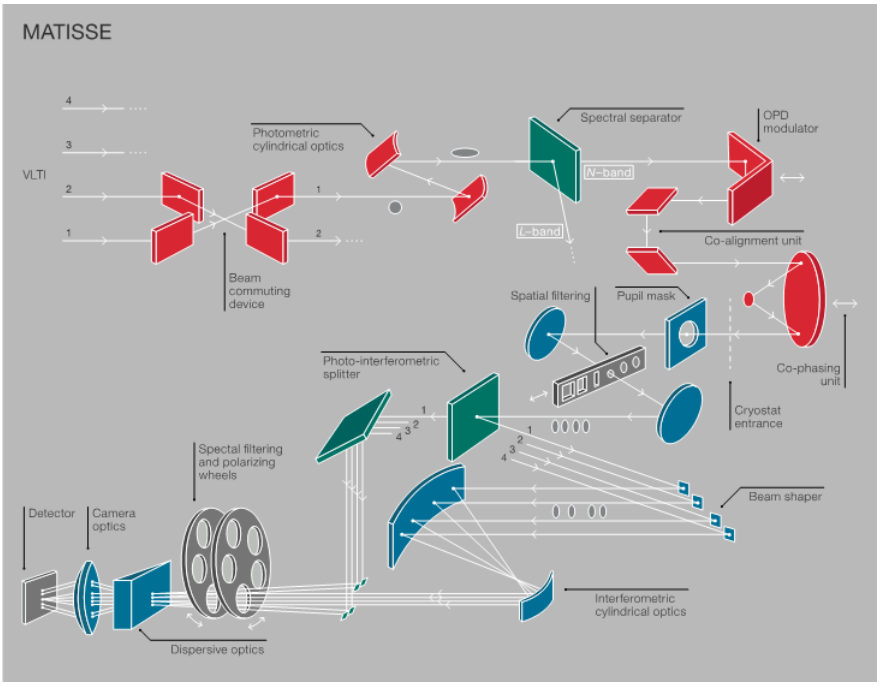


Figure 4.2: Display of some of the MATISSE opto-mechanical functions. Part of these functions, like the OPD (Optical Path Difference) modulators, are located in the so-called Warm Optics while other functions, like the spatial filtering, are located in the Cold Optics. The red components represent optical elements located on the warm optics table at ambient temperature. The blue ones represent optical elements of the Cold Optics located in the cryostats. The green ones correspond to the spectral and photometric beam splitters.

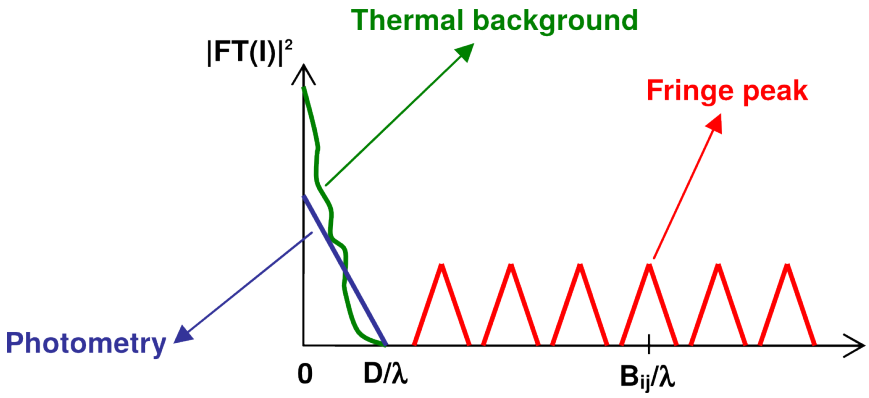


Figure 4.3: Representation of the power spectrum of the interferometric pattern.

# 5 Quick start

This section describes the most immediate usage of the MATISSE pipeline recipes.

## 5.1 MATISSE pipeline recipes

### 5.1.1 Instrument Calibrations

In order to reduce interferometric observations, it is mandatory to calibrate the detectors and the combiners. These calibrations are obtained via dedicated observations in laboratory. The following recipes produce these calibration maps:

- **mat_cal_det** creates the BADPIXEL, FLATFIELD and NONLINEARITY maps.
- **mat_est_flat** creates the bias map, called OBS_FLATFIELD.
- **mat_est_shift** creates the distortion map, called SHIFT_MAP (including the spectral dispersion law).
- **mat_est_kappa** creates the kappa matrix, called KAPPA_MATRIX. It gives the flux ratio between photometric and interferometric channels.

## 5.2 An introduction to Reflex, Gasgano and EsoRex

Before being able to call pipeline recipes to process a set of data, the data must be correctly classified, and associated with the appropriate calibrations. The *Data Classification* consists of tasks such as: "What kind of data am I?", *e.g.*, BIAS, "to which group do I belong?", *e.g.*, to a particular Observation Block or observing template. *Data Association* is the process of selecting appropriate calibration data for the reduction of a set of raw science frames. Typically, a set of frames can be associated if they share a number of properties, such as instrument and detector configuration. Since all the required information is stored in the FITS headers, data association is based on a set of header keywords (called "association keywords") and the process is specific to each type of calibration. The process of data classification and association is known as data organisation.

An instrument pipeline consists of a set of data processing modules that can be called from different host applications, namely:

- *EsoReflex* is a graphical tool that helps the user to execute data reduction workflows which contain several recipes. This dramatically decreases the time the user needs to run a whole reduction chain, from calibration and raw data down to the final products. *EsoReflex* takes care of grouping the different data sets, associating the calibration frames and managing the inter-dependencies between recipes in the calibration cascade. **EsoReflex is the recommended software tool for reducing your data.** The MATISSE EsoReflex workflows are described in [14].
- *Gasgano* is an alternative data management tool that simplifies the data organization process. In addition, *Gasgano* allows the user to execute directly the pipeline recipes on a set of selected files.
- *EsoRex* is a command line tool used to run the pipeline recipes. *EsoRex* commands can be easily scripted.

- The Paranal observatory implements automatic data management tools that trigger the execution of pipeline recipes. This aspect is not covered in this manual.

### 5.2.1 Using Gasgano

*Gasgano* provides a complete graphic interface for data browsing, classification and association, and offers several other utilities such as easy access to recipes documentation and preferred data display tools.

*Gasgano* can be started from the system prompt in the following way:

```
gasgano &
```

The *Gasgano* main window will appear. On Figure 5.1, a view on a set of MATISSE raw data is shown as an example. *Gasgano* can be pointed to the directories where the data to be handled are located using the navigation panels accessible via the *Add/Remove Files* entry of the *File* menu.

The data are hierarchically organised as preferred by the user. After each file name are shown the classification, the template id (which indicates the template the data has been taken with), the original filename, the template exposure number and the number of exposures in the template.

More information about a single frame can be obtained by clicking on its name: the corresponding FITS file header will be displayed on the bottom panel, where specific keywords can be opportunely filtered and searched. Images and tables may be easily displayed using the viewers specified in the appropriate *Preferences* fields.

Frames can be selected from the main window for being processed by the appropriate recipe. On Figure 5.2, the calibrator raw frames (CALIB_RAW), and two sky raw frames(SKY_RAW), together with the already produced Distortion Map (SHIFT_MAP), the Observing Flat Field (OBS_FLATFIELD), the Bad Pixel Map (BADPIX), and the Nonlinearity Map (NONLINEARITY) are all selected and sent to the *mat_cal_image* recipe. This will open a *Gasgano* recipe execution window (see Figure 5.3), having all the specified files listed in its *Input Frames* panel.

Help about the recipe may be obtained from the *Help* menu. Before launching the recipe, its configuration may be opportunely modified on the *Parameters* panel (on top). The window contents might be saved for later use by selecting the *Save Current Settings* entry from the *File* menu.

At this point the recipe can be launched by pressing the *Execute* button. Messages from the running recipe will appear on the *Log Messages* panel at bottom, and in case of successful completion the products will be listed on the *Output Frames* panel, where they can be easily viewed and located back on the Gasgano main window.

Please refer to the *Gasgano User's Manual* [7] for a more complete description of the *Gasgano* interface.

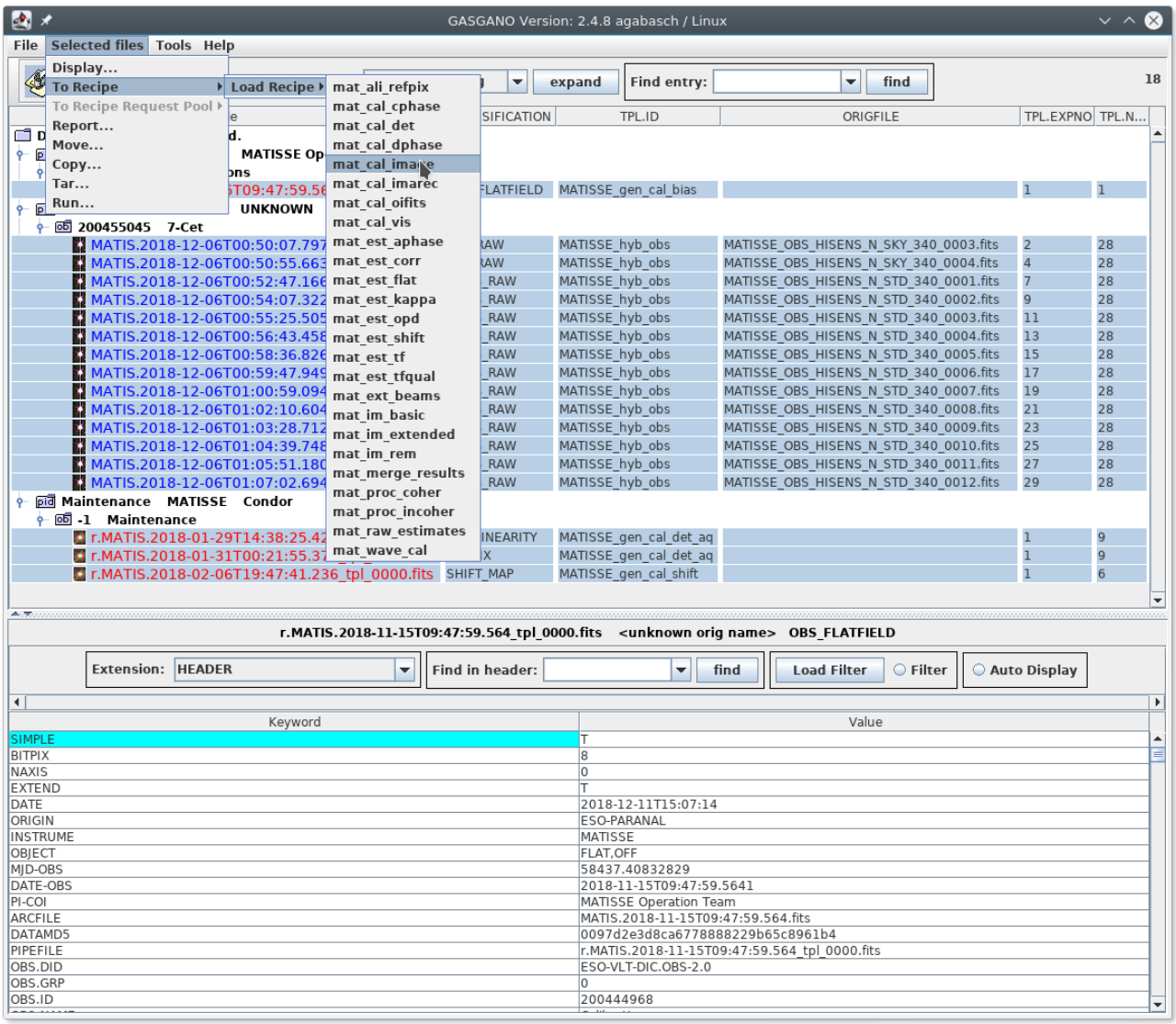


Figure 5.1: *The Gasgano main window.*

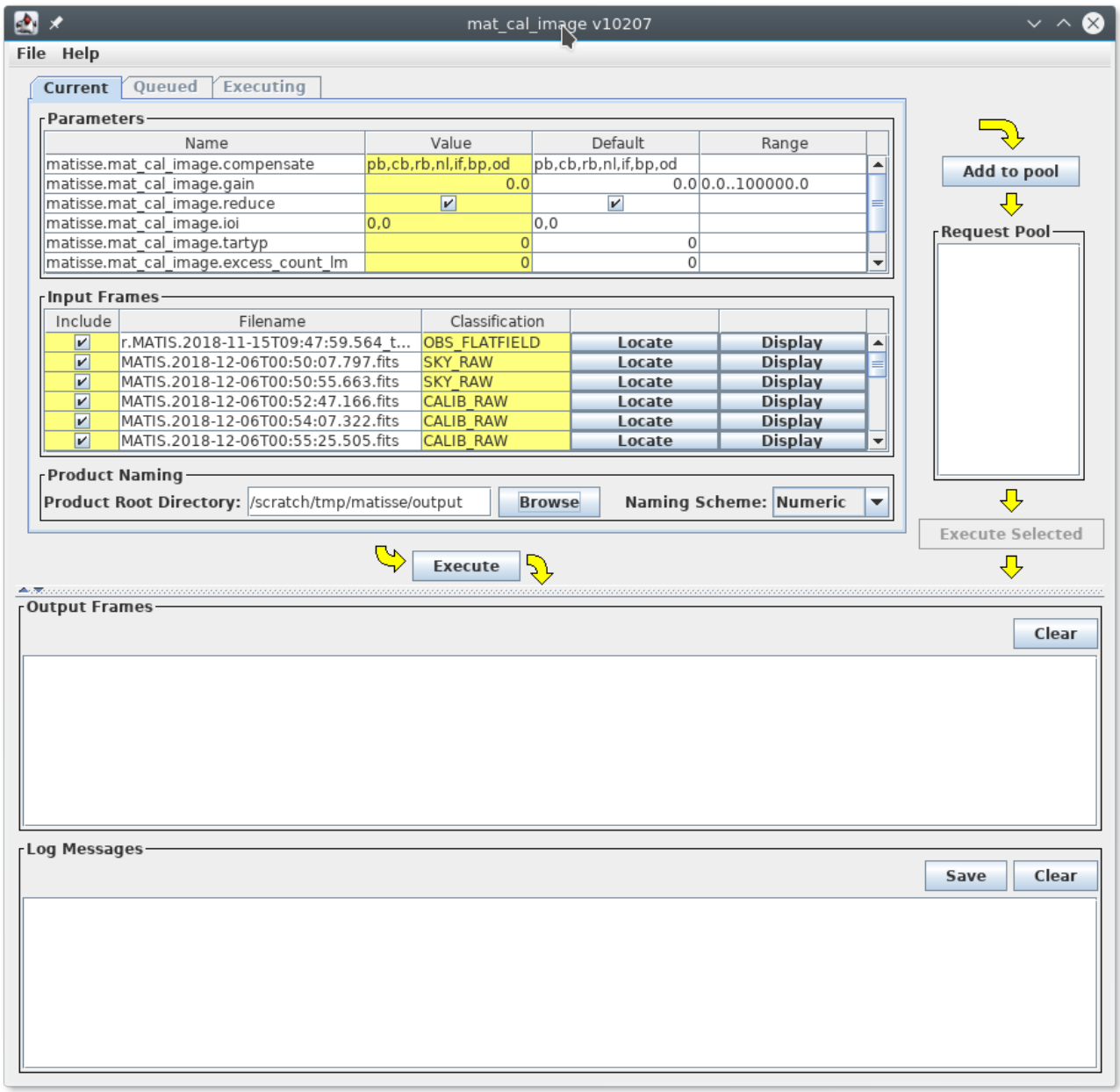


Figure 5.2: *Selecting files to be processed by a MATISSE pipeline recipe.*

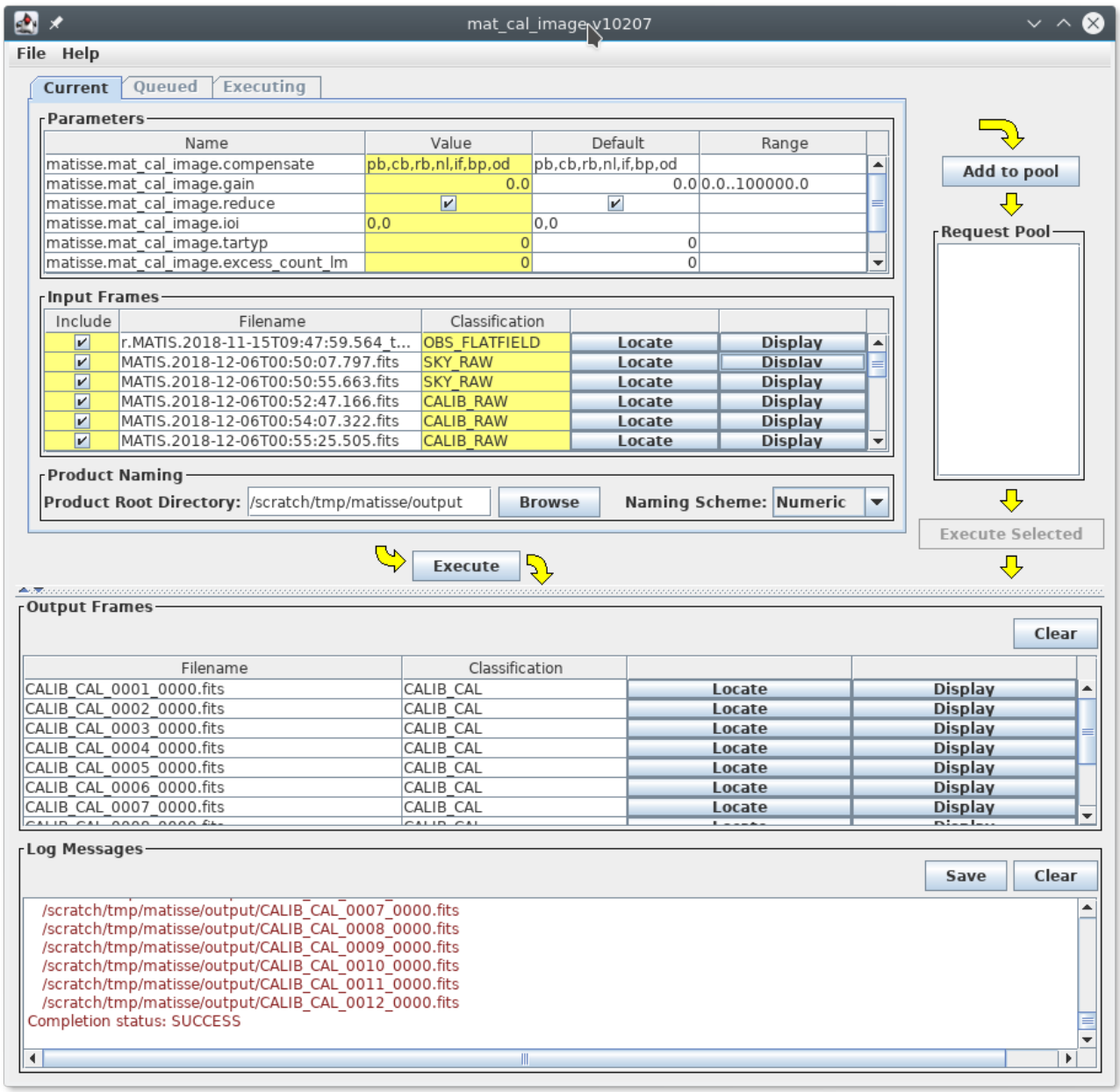


Figure 5.3: *The Gasgano recipe execution window.*

### 5.2.2 Using EsoRex

*EsoRex* is a command line utility for running pipeline recipes. It may be embedded by users into data reduction scripts for the automation of processing tasks. On the other side, *EsoRex* doesn't offer all the facilities available with *Gasgano*, and the user must classify and associate the data using the information contained in the FITS header keywords. The user should also take care of defining the input set-of-frames and the appropriate configuration parameters for each recipe run:

**The set-of-frames:** Each pipeline recipe is run on a set of input FITS data files. When using *EsoRex* the filenames must be listed together with their Data Organizer (DO) category in an ASCII file, the *set-of-frames* (SOF), that is required when launching a recipe. [1]

Here is an example of SOF, valid for the *mat_cal_image* recipe

```
/path/raw/MATIS.2018-12-06T00:50:07.797.fits            SKY_RAW
/path/raw/MATIS.2018-12-06T00:50:55.663.fits            SKY_RAW
/path/raw/MATIS.2018-12-06T00:52:47.166.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T00:54:07.322.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T00:55:25.505.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T00:56:43.458.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T00:58:36.826.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T00:59:47.949.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T01:00:59.094.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T01:02:10.604.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T01:03:28.712.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T01:04:39.748.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T01:05:51.180.fits            CALIB_RAW
/path/raw/MATIS.2018-12-06T01:07:02.694.fits            CALIB_RAW
/path/cal/r.MATIS.2018-01-29T14:38:25.429_tpl_0002.fits NONLINEARITY
/path/cal/r.MATIS.2018-01-31T00:21:55.372_tpl_0000.fits BADPIX
/path/cal/r.MATIS.2018-02-06T19:47:41.236_tpl_0000.fits SHIFT_MAP
/path/cal/r.MATIS.2018-11-15T09:47:59.564_tpl_0000.fits OBS_FLATFIELD
```

It contains for each input frame the full path file name and its DO category. The pipeline recipe will access the listed files when required by the reduction algorithm.

Note that the MATISSE pipeline recipes do not verify in any way the correctness of the *DO Category* specified by the user in the SOF. The reason of this lack of control is that the MATISSE recipes are just the DRS (data reduction system) component of the complete pipeline running on Paranal, where the task of data classification and association is carried out by separate applications. Moreover, using *Gasgano* as an interface to the pipeline recipes will always ensure a correct classification of all the data frames, assigning the appropriate DO category to each one of them (see Section 5.2.1, page 26).

A recipe handling an incorrect SOF may stop or display unclear error messages at best. In the worst cases, the recipe would apparently run without any problem, producing results that may look reasonable, but are actually flawed.

[1] The set-of-frames corresponds to the *Input Frames* panel of the *Gasgano* recipe execution window (see Figure 5.3, page 29).

**EsoRex syntax:** The basic syntax to use ESOREX is the following:

**esorex [esorex_options] recipe_name [recipe_options] set_of_frames**

To get more information on how to customise ESOREX (see also [7]) run the command:

**esorex - -help**

To generate a configuration file esorex.rc in the directory $HOME/.esorex run the command:

**esorex - -create-config**

A list of all available recipes, each with a one-line description, can be obtained using the command:

**esorex - -recipes**

All recipe parameters (aliases) and their default values can be displayed by the command

**esorex - -params recipe_name**

To get a brief description of each parameter meaning execute the command:

**esorex - -help recipe_name**

To get more details about the given recipe give the command at the shell prompt:

**esorex - -man-page recipe_name**

**Recipe configuration:** Each pipeline recipe may be assigned an *EsoRex* configuration file, containing the default values of the parameters related to that recipe.[2] The configuration files are normally generated in the directory `$HOME/.esorex`, and have the same name as the recipe to which they are related, with the filename extension `.rc`. For instance, the recipe *mat_cal_image* has its *EsoRex* generated configuration file named `mat_cal_image.rc`, and is generated with the command:

**esorex - -create-config mat_cal_image**

The definition of one parameter of a recipe may look like this:

```
# --tartyp
# TARTYP estimation (0 = none, 1 = N*S+U+N*T+U, 2 = show intensity, 4 = show
# correlation, 8 = estimate TARTYP, 16 = change TIME, TARTYP, LOCALOPD and
# STEPPING_PHASE, 32 = exchange U with S or T).
matisse.mat_cal_image.tartyp=0
```

In this example, the parameter `matisse.mat_cal_image.tartyp` is set to the value `0`. In the configuration file generated by *EsoRex*, one or more comment lines are added containing information about the possible values of the parameter, and an alias that could be used as a command line option.

The recipes provided by the MATISSE pipeline are designed to implement a cascade of macro data reduction steps, each controlled by its own parameters. For this reason and to prevent parameter name clashes we specify as parameter prefix not only the instrument name but also the name of the step they refer to. Shorter parameter aliases are made available for use on the command line.

The command

**esorex - -create-config recipe_name**

generates a default configuration file **recipe_name.rc** in the directory **$HOME/.esorex**[3].

[2] The *EsoRex* recipe configuration file corresponds to the *Parameters* panel of the *Gasgano* recipe execution window (see Figure 5.3, page 29).

[3] If a number of recipe parameters are specified on the command line, the given values will be used in the created configuration file.

A recipe configuration file different from the default one can be specified on the command line:

**esorex - -recipe-config=my_alternative_recipe_config**

Recipe parameters are provided in section 9 and their role is described in Section 11.

More than one configuration file may be maintained for the same recipe but, in order to be used, a configuration file not located under `$HOME/.esorex`, or having a name different from the recipe name, should be explicitly specified when launching a recipe.

**Recipe execution:** A recipe can be run by specifying its name to *EsoRex*, together with the name of a set-of-frames. For instance, the following command line would be used to run the recipe *mat_cal_image* for processing the files specified in the set-of-frames `mat_cal_image.sof`:

**esorex mat_cal_image mat_cal_image.sof**

The recipe parameters can be modified either by editing directly the used configuration file, or by specifying new parameter values on the command line using the command line options defined for this purpose. Such command line options should be inserted after the recipe name and before the SOF name, and they will supersede the system defaults and/or the configuration file settings. For instance, to set the *mat_cal_image* recipe *tartyp* parameter to `1`, the following should be typed:

**esorex mat_cal_image - -tartyp=1 mat_cal_image.sof**

For more information on *EsoRex*, see `http://www.eso.org/cpl/esorex.html`.

# 6 Known Problems

- As with all other ESO data, the use of the colon sign (:) in file names causes problems on macOS and on Windows operating systems. This is not specific to the MATISSE pipeline, but it should be noted. It can be circumvented by renaming files locally.

## 6.1 Open issues

Besides known problems there are also a few open issues or limitations, where the functionality and applicability of the pipeline is not yet fully established.

- ...

# 7 Instrument Data Description

MATISSE data can be separated into *raw* frames and *product* frames. Raw frames are the unprocessed output of the MATISSE instrument observations, while product frames are either the result of the MATISSE pipeline processing (as reduced frames, master calibration frames, etc.), or are outsourced (as standard stars catalogues, lists of grism characteristics, etc.).

Any raw or product frame can be classified on the basis of a set of keywords read from its header. Data classification is typically carried out by the DO or by *Gasgano* [7], that apply the same set of classification rules. The association of a raw frame with calibration data ($e.g.$, of a science frame with a master bias frame) can be obtained by matching the values of a different set of header keywords.

Each kind of raw frame is typically associated to a single MATISSE pipeline recipe, *i.e.*, the recipe assigned to the reduction of that specific frame type. In the pipeline environment this recipe would be launched automatically.

A product frame may be input to more than one MATISSE pipeline recipe, but it may be created by just one pipeline recipe (with the same exceptions mentioned above). In the automatic pipeline environment a product data frame alone wouldn't trigger the launch of any recipe.

A number of MATISSE data product frames are in the $2^{\mathrm{nd}}$ version of the OIFITS standard. The OIFITS (Optical Interferometry FITS) is the standard for exchanging data from optical (visible or infrared) interferometers [3]. A valid OIFITS file contains an OI_TARGET table, one or more OI_ARRAY tables, one or more OI_WAVELENGTH tables, and at least one data table (OI_VIS, OI_VIS2, OI_T3, or the optional OI_FLUX). The OI_TARGET table contains information on the target, the OI_ARRAY table specifies the station configuration, and the OI_WAVELENGTH table gives the wavelengths of the spectrally resolved data. The VISAMP column in the OI_VIS table contains the differential visibility by default, but by setting the -corrFlux reduction parameter it will store the correlated flux instead. The header keyword AMPTYPE indicates the data type in VISAMP (AMPTYPE = 'differential' or 'correlated flux'). The OI_OPDWVPO product frame and the TF2 binary table contain pipeline-specific data, and these are not part of the original OIFITS standard. The VIS2DATA in the OI_VIS2 table may contain squared correlated fluxes (by setting the -corrFlux option), instead of the standard squared visibilities. The following table gives an overview of the OIFITS data tables present in the MATISSE product frames.

| Product frames | Table | Column | Unit | Description |
|---|---|---|---|---|
| TARGET_CAL_INT,<br>CALIB_RAW_INT,<br>TARGET_RAW_INT,<br>RAW_VIS2,<br>CAL_VIS2 | OI_VIS2 | VIS2DATA<br><br><br>VIS2ERR | | squared visibility or squared correlated flux<br>(-corrFlux option)<br>VIS2DATA error |
| TARGET_CAL_INT,<br>CALIB_RAW_INT,<br>TARGET_RAW_INT,<br>RAW_CPHASE,<br>CAL_CPHASE | OI_T3 | T3AMP<br>T3AMPERR<br>T3PHI<br>T3PHIERR | <br><br>deg<br>deg | not used<br>not used<br>closure phase<br>closure phase error |
| TARGET_CAL_INT,<br>CALIB_RAW_INT,<br>TARGET_RAW_INT,<br>RAW_DPHASE,<br>CAL_DPHASE | OI_VIS | VISAMP<br><br><br>VISAMPERR | | differential visibility or coherently processed correlated flux (-corrFlux and –coherentAlgo options)<br>VISAMP error |

| | | | | |
|---|---|---|---|---|
| | | VISPHI<br>VISPHIERR | deg<br>deg | differential phase<br>differential phase error |
| CALIB_RAW_INT,<br>RAW_TF2 | TF2 | TF2<br>TF2ERR<br>TF<br>TFERR | | transfer function (squared)<br>transfer function error (squared)<br>transfer function error<br>transfer function error |
| TARGET_CAL_INT,<br>CALIB_RAW_INT,<br>TARGET_RAW_INT,<br>RAW_SPECTRUM | OI_FLUX | FLUXDATA<br>FLUXERR | ADU<br>ADU | total spectrum<br>total spectrum error |
| OI_OPDWVPO | OPD | OPD<br>OPDERR<br>TEMPOFF<br>TEMPOFFERR<br>HUMOFF<br>HUMOFFERR | $\mu$m<br>$\mu$m | OPD estimation<br>OPD error<br>not used<br>not used<br>achromatic phase<br>achromatic phase error |

In the following all raw and product MATISSE 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* (see Section 5.2.2, page 30).

## 7.1 RAW science data

The RAW science frames are created with the following DPR keywords (CATG/TECH/TYPE):

- **SCIENCE/OBJECT/INTERFEROMETRY**: are observations of a scientific object. Such frames could contain fringe data only (in HighSens mode), photometric data only (in HighSens mode) or fringes + photometric data (in SiPhot mode). In HighSens mode, shutter keywords (INS.BSNi.ST) have to be checked to determine if the frame corresponds to fringes or photometry. (DO.CATG = TARGET_RAW)

## 7.2 RAW calibration data

The RAW frames used to calibrate the instrument have the following DPR keywords (CATG/TECH/TYPE):

- **CALIB/STD/INTERFEROMETRY**: are observations of a calibrator object. Such frames could contain fringe data only (in HighSens mode), photometric data only (in HighSens mode) or fringes+photometric data (in SiPhot mode). In HighSens mode, shutter keywords (INS.BSNi.ST) have to be checked to determine if the frame corresponds to fringes or photometry. (DO.CATG = CALIB_RAW)
- **CALIB/SKY/INTERFEROMETRY**: are observations of sky (all shutters open). (DO.CATG = SKY_RAW)
- **CALIB/DARK/IMAGE,DETCHAR**: are observations with all shutters closed. (DO.CATG = DARK)
- **CALIB/FLAT/IMAGE,DETCHAR**: are observations of the internal source without pinhole with all shutters open. (DO.CATG = FLAT)

- **CALIB/FLAT,OFF/IMAGE,DETCHAR**: are observations with all shutters closed and in the same detector configuration as in the scientific observations (DIT, Windowing). (DO.CATG = OBSDARK)
- **CALIB/DARK/IMAGE,BASIC**: are observations with all shutters closed for basic detector monitoring. (DO.CATG = IM_COLD)
- **CALIB/DARK/IMAGE,EXTENDED**: are observations with all shutters closed for extended detector monitoring. (DO.CATG = IM_DARK)
- **CALIB/FLAT/IMAGE,EXTENDED**: are observations of the internal source without pinhole with all shutters open for extended detector monitoring. (DO.CATG = IM_FLAT)
- **CALIB/FLAT,LAMP/REMANENCE**: are observations for remanence detector monitoring. (DO.CATG = IM_PERIODIC)
- **CALIB/DARK/IMAGE**: are observations of laboratory background with pinholes spatial filter. They are used to determine the distortion map. (DO.CATG = DISTOR_HOTDARK)
- **CALIB/WAVE,LAMP,PINHOLE/SPECTRUM**: are observations with internal source and pinholes spatial filter. They are used to determine the distortion map. (DO.CATG = DISTOR_IMAGES)
- **CALIB/WAVE,LAMP,SLIT/SPECTRUM**: are observations with internal source and slit spatial filter. They are used to determine the distortion map. (DO.CATG = SPECTRA_HOTDARK)
- **CALIB/WAVE,LAMP,FOIL/SPECTRUM**: are observations with internal source, slit spatial filter and plastic foils. They are used to determine the distortion map. (DO.CATG = SPECTRA_IMAGES)
- **CALIB/KAPPA,BACKGROUND/SPECTRUM**: are observations of background with one shutter open only. They are used to determine the kappa matrix. (DO.CATG = KAPPA_HOTDARK)
- **CALIB/KAPPA,LAMP/SPECTRUM**: are observations with internal source and one shutter open only. They are used to determine the kappa matrix. (DO.CATG = KAPPA_SRC)
- **CALIB/BACKGROUND/INTERFEROMETRY**: are observations of laboratory background. (DO.CATG = HOT_DARK)
- **CALIB/LAMP/INTERFEROMETRY**: are fringe observations with internal source. (DO.CATG = CALIB_SRC_RAW)

## 7.3 STATIC calibration

The STATIC calibration frames have the following PRO.CATG keyword:

- **JSDC_CAT**: is the JMMC Stellar Diameters Catalog[4] (JSDC) [1]. It contains angular diameter estimations for nearly all the stars of the Hipparcos and Tycho catalogue that have an associated spectral type in Simbad/CDS.

[4]The catalog is online available at https://vizier.u-strasbg.fr/viz-bin/VizieR?-source=II/346&-to=3.

## 7.4 PRODUCT calibration data

The PRODUCT of the calibration recipes has the following PRO.CATG keywords:

- **BADPIXEL**: is the bad pixels map.
- **FLATFIELD**: is the flat field map.
- **NONLINEARITY**: is the detector nonlinearity map.
- **OBS_FLATFIELD**: is the observing flat field. In fact, it contains the bias map of the detector in the same configuration as in the scientific observations (DIT, windowing, readout mode) only.
- **SHIFT_MAP**: is the distortion map with the spectral dispersion law.
- **KAPPA_MATRIX**: is the kappa matrix.
- **IM_BASIC**: is the basic detector monitoring map.
- **IM_EXTENDED**: is the extended detector monitoring map.
- **IM_REMANENCE**: is the detector remanence.
- **CALIB_RAW_INT**: is the OIFITS of calibrator stars or internal source. It contains uncalibrated interferometric measurements.
- **OI_OPDWVPO**: is the OPD estimation.

The following are produced as INTERMEDIATE products during the execution of the pipeline recipe **mat_raw_estimates** which utilizes several other recipes:

- **mat_proc_coher**
  - RAW_DPHASE: is the OIFITS of the differential phase, as well as the coherently processed correlated flux and visibilities.
- **mat_proc_incoher**
  - BSimag/BSreal: are the imaginary and real parts of the bispectrum, respectively.
  - fringePeaks: is the sky-subtracted *squared* 2-D Fourier transform of the interferogram.
  - DSPtarget/DSPsky: are the spectral densities of the target and sky respectively, used in estimating the correlated flux.
  - OBJ_CORR_FLUX: is the sky-subtracted 2-D Fourier transform of the interferogram.
  - RAW_CPHASE: is the OIFITS of the uncalibrated closure phase.
  - RAW_SPECTRUM: is the OIFITS of the uncalibrated spectrum.
  - RAW_VIS2: is the OIFITS of the uncalibrated squared visibilities.
- **mat_est_tf**

  - RAW_TF2: is the OIFITS of the transfer function.

- **mat_est_opd**
  - nrjImag/nrjReal: are the imaginary and real parts of the fringe peaks used for calculating the water vapor OPD.

- **mat_ext_beams**
  - PHOT_BEAMS: is the OIFITS of the extracted photometric beams.

## 7.5 PRODUCT science data

The PRODUCT of the science reduction are identified by the following PRO.CATG keyword:

- **TARGET_RAW_INT**: is the OIFITS of science object. It contains uncalibrated interferometric measurements.
- **TARGET_CAL_INT**: is the OIFITS of science object. It contains calibrated interferometric measurements.
- **TARGET_REC**: is the reconstructed image.

# 8 Data Reduction

## 8.1 Graphical overview of the cascade

Fig. 8.1 shows the main MATISSE data reduction cascade.

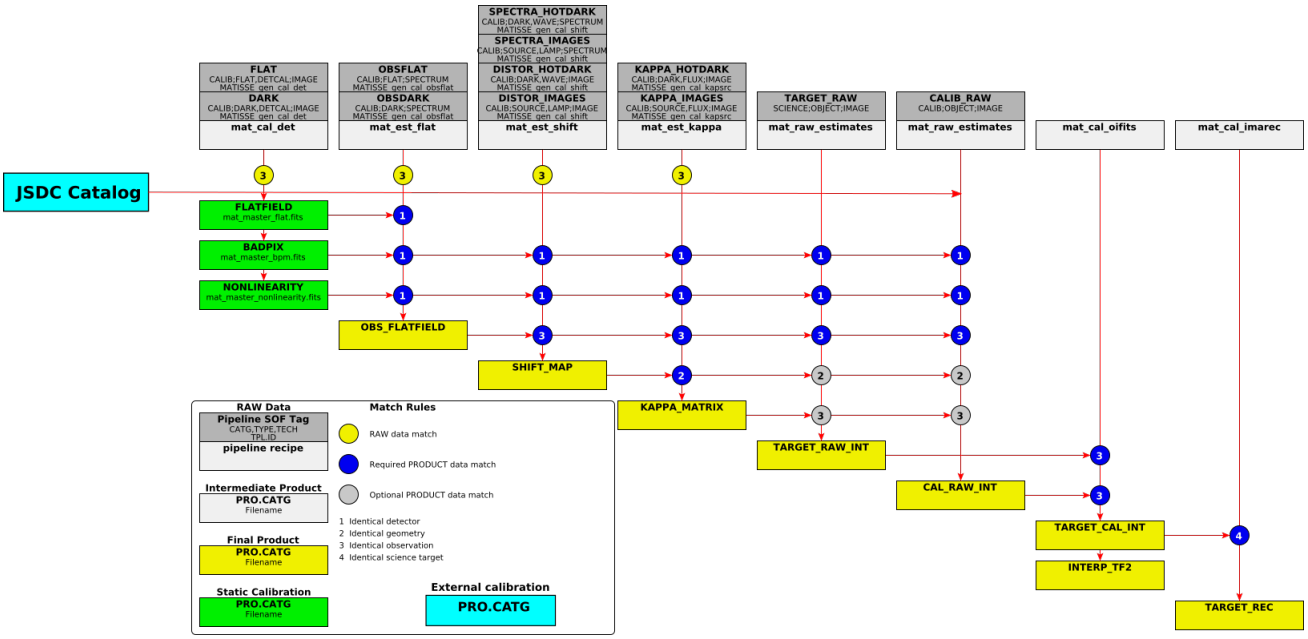


Figure 8.1: Main MATISSE data reduction cascade

## 8.2 Data reduction recipes

We distinguish between **master** and **atomic** recipes. The processing of science data can be done using just these three master recipes which call a number of atomic recipes:

- `mat_raw_estimates` It computes raw interferometric observables (visibility, closure phase, differential phase) from raw images. It is applied on all FITS files recorded during an OB.
- `mat_cal_oifits` It calibrates raw interferometric observables of the scientific target from the raw interferometric observables of calibrator stars. It produces OIFITS files.
- `mat_cal_imarec` It reconstructs an image from a series of OIFITS files. It uses the IRBIS algorithm described in Section 9.6.

The raw data can be reduced by using these master recipes as well as by reducing them step by step with the *EsoReflex* workflows. In this case the user applies *atomic* recipes. An *atomic recipe* applies a specific processing,

whereas a *master* recipe applies several processings sequentially. For example to compute raw interferometric observables from raw images, the user should apply the following sequence:

1. `mat_cal_image` raw image calibration
2. `mat_ext_beams` photometry estimation
3. `mat_est_corr` correlated flux estimation
4. `mat_est_odp` OPD estimation
5. `mat_proc_incoherent` incoherent processing
6. `mat_proc_coherent` coherent processing
7. `mat_est_tf` transfer function estimation
8. `mat_merge_results` merge intermediate products into OIFITS files

This sequence of recipes is equivalent to the *master* recipe **mat_raw_estimates**.

The other atomic recipes are:

- `mat_cal_cphase` Calibration of the phase closure
- `mat_cal_det` Calculates a bad pixel map, a flatfield map and a non-linearity map
- `mat_cal_dphase` Calibration of the differential phase
- `mat_cal_vis` Calibration of the squared visibility
- `mat_est_aphase` Estimates the absolute phase from reference images and differential phases
- `mat_est_flat` Estimation of the ObsFlatField Map
- `mat_est_kappa` Estimation of the Kappa Matrix
- `mat_est_shift` Estimation of the Shift Map
- `mat_im_basic` Calculates a basic detector monitoring map
- `mat_im_extended` Calculates a extended detector monitoring map
- `mat_im_rem` Calculates a detector remanence map

In addition, there are three more recipes included in the current pipeline code base: `mat_est_tfqual`, `mat_wave_cal`, `mat_ali_refpix`. These atomic recipes are currently not used and are also not further described here.

# 9 Pipeline Recipe Interfaces

In this section we list the different recipes used for the data reduction along with the required input data, the output provided by the recipe, as well as their tags.

We also provide a list of the pipeline products for each recipe, indicating their default recipe name, the value of the FITS keyword HIERARCH ESO PRO CATG (in short PRO.CATG), and a short description. For each recipe we also list in a table the input parameters (as they appear in the recipe configuration file), the corresponding aliases (the corresponding names to be eventually set on command line), and their default values. Quality control parameters are also listed. Those are stored in relevant pipeline products. More information on instrument quality control can be found on http://www.eso.org/qc

## 9.1 mat_raw_estimates

This master recipe calls all data reduction steps which transform the measured raw frames into averaged **raw** interferometric products like the raw squared visibilities, the raw closure phases, and the raw differential phases. This recipe could be applied to both target and calibrator frames.

It supports two different methods to estimate the correlated flux, the raw observable in MATISSE data processing. The default method (`corrFlux=False`) estimates the correlated flux in an incoherent way. In order to obtain correlated fluxes for a faint target, we recommend to use the coherent processing which can be activated by appending `-corrFlux=TRUE` and `-coherentAlgo=2` to the call of this recipe. The results will be in `OIVIS.VISAMP`.
In the case of GRA4MAT observations, the fringe tracker data are used to:

- Flag the MATISSE frames corresponding to fringe jumps
- Apply the N band a posteriori co-phasing method with the fringe tracker data in K band (see section Algorithms)

These two methods are used automatically as soon as the GRA4MAT data are in the *sof* file. In this case, the *coherentAlgo* parameter is no more relevant. If one does not want these two methods applied, the GRA4MAT files should be removed from the *sof* file.

### 9.1.1 Input

In the following table we gather the inputs to run the mat_raw_estimates recipe. Depending on the DO category (Data Organizer category) mandatory inputs are indicated in the table.

| DO category | Explanation | Is Required? |
|---|---|---|
| CALIB_SRC_RAW | Lab Lamp data | Yes (if lab data) |
| HOT_DARK | Lab Background data | Yes (if lab data) |
| SKY_RAW | Sky data | No |
| CALIB_RAW | Calibrator data | Yes (if calibrator) |
| TARGET_RAW | Target data | Yes (if target) |

| | | |
|---|---|---|
| OBS_FLATFIELD | Observing Flat Field | Yes |
| BADPIX | Bad Pixel Map | Yes |
| NONLINEARITY | Nonlinearity Map | Yes |
| SHIFT_MAP | Distortion Map | Yes |
| KAPPA_MATRIX | Kappa Matrix | Yes (if in SciPhot Mode) |

### 9.1.2 Output

| DO category | explanation |
|---|---|
| TARGET_RAW_INT | OIFITS file with raw data on target |
| CALIB_RAW_INT | OIFITS file with raw data on calibrator |
| OI_OPDWVPO | OPD estimation |

### 9.1.3 Quality control

The QC parameters, or Quality control parameters are computed by the recipe and stored in the header of the output files. If the option –corrFlux in used during the data reduction, or if it is automatically set when photometric data are missing, some keywords can be irrelevant. The following table summarizes the keywords, their significance and there relevance depending on the use of the corrflux options.

| QC keyword | Significance | Corrflux usage |
|---|---|---|
| QC DET$_{1,2}$TFSQR$_{1...6}$ MEAN | Corrected visibility from the angular diameter of the target | irrelevant |
| QC DET$_{1,2}$ VISSQR$_{1...6}$ | The Visibility of the corresponding baseline | relevant |
| QC DET$_{1,2}$ VISSQR$_{1...6}$SNR | and its signal to noise ratio | relevant |
| QC DET$_{1,2}$ FLUX IP$_{13,5,7}$ | The flux in ADU | irrelevant |
| QC DET$_{1,2}$ RATIO IP$_{1,3,5,7}$ | Ratio of IP$_x$ normalized to the sum of all four beams | irrelevant |
| QC DET$_{1,2}$ CPHAS$_{1...4}$ | The closure phase of the corresponding triplet | relevant |
| QC DET$_{1,2}$ CPHASE$_{1...4}$ STDEV | and its standard deviation | relevant |
| QC DET$_{1,2}$ DPHASE$_{1...6}$ | Differential phase for the corresponding baseline | relevant |
| QC DET$_{1,2}$ DPHASE$_{1...6}$ STDEV | and its standard deviation | relevant |

### 9.1.4 Options

| Option | Explanation | Default |
|---|---|---|
| --compensate | Defines which kind of compensation should be applied (**none**: no compensation at all, **all**: all compensations possible, **dd**: detector specific defaults, **pb**: subtract pixel bias, **gb**: subtract global bias, **cb**: subtract detector channel bias, **rb**: subtract row bias, **ct**: subtract crosstalk, **nl**: nonlinearity compensation, **if**: divide by instrument flat, **df**: divide by detector flat, **bp**: bad pixel interpolation, **el**: convert to electrons, **od**: remove optical distortion). | [pb,cb,rb,nl,if,bp,od] |

| | | |
|---|---|---|
| --gain | Default conversion gain in [e-/DU]. | [0.0] |
| --reduce | Flag if the reference sub-windows should be removed from the result. | [TRUE] |
| --ioi | images of interest: <first>,<count>. | [0,0] |
| --tartyp | TARTYP[5] estimation (**flag**, please add the following values for the desired effect: **0** = none, **1** = N*S+U+N*T+U, **2** = show intensity, **4** = show correlation, **8** = estimate TARTYP, **16** = change TARTYP, **32** = exchange U with S or T, **64** = change TIME, **128** = change LOCALOPD and STEPPING_PHASE, **256** = show the extracted exposure setup, **512** = show the first chopping cycle, **1024** = show the extracted timetable, **2048** = show the modified frame info, **4096** = use shift calculated from square wave fit, **8192** = use shift calculated from timing). | [57] |
| --excess_count_lm | Excess frames mistakenly produced before first TIM-Board trigger (LM-Band). | [-1] |
| --excess_count_n | Excess frames mistakenly produced before first TIM-Board trigger (N-Band). | [-1] |
| --excess_count_list | Specification of an excess count for each exposure number (ESO TPL EXPNO keyword, excess count pair) as a list of numbers separate by comma. | [none] |
| --replaceTel | Replace Photometry of one telescope by the mean of the 3 others. An integer with 2 digits is expected. The first digit is used for UTs and the second for ATs. 0: none, 1: AT1/UT1, 2: AT2/UT2, 3: AT3/UT3, 4: AT4/UT4. For example: replaceTel=01 means replace AT1, replaceTel=32 means replace UT3 and AT2. | [3] |
| --hampelFilterKernel | Only for L/M band. Apply a temporal Hampel filter to all pixels before deriving the photometry. This filter improves the photometric estimation in case of faint stars (< 5Jy in L band with AT). The parameter fixes the size of the kernel of the filter (kernel=0 is recommended). | [0] |
| --useOpdMod | useOpdMod option. | [FALSE] |
| --coherentIntegTime | Specify a coherent integration time (in s). | [0.0] |
| --corrFlux | corrFlux option. | [FALSE] |
| --cumulBlock | cumul all blocks of an OB. | [FALSE] |
| --coherentAlgo | Estimation Algorithm (**1**: AMBER like Method, **2**: CRAL Cohrent Integration Method). | [1] |
| --spectralBinning | Spectral Binning in pixels. Should be an odd integer number. If an even integer is given, the pipeline will select the next odd integer. | [1] |

[5] where S means Sky, T means Target, U means Undefined, and N ist the number of frames

## 9.2 mat_est_flat

This plugin uses a raw file containing cold dark frames (DO classification OBSDARK) produced by the MATISSE_gen_cal_bias template. The plugin calculates for a specific observation a setup specific bias map with the same setup (geometry and exposure time) as the science observation. The static bad pixel map, the static nonlinearity map and the static flatfield map are needed for running this plugin.

The static bad pixel map (PRO.CATG = BADPIX) is needed to ignore the bad pixels during normalization of the optical flat. The static nonlinearity map is needed to separate the optical (large scale) variations from the fix pattern noise and the nonlinearity (small scale) variations. The static flatfield map (PRO.CATG = FLATFIELD) is used to get the conversion factor for converting ADU into electrons and copy it into the optical flatfield.

This plugin produces a FITS file which contains the following data:

- An empty primary HDU which contains the keywords from the first raw data file and the QC1 parameters.
- A binary table extension (OBS_FLAT_BIAS) containing the optical flat and the bias map (median cold dark).

The following QC1 parameters will be calculated (or copied) and stored in a cpl_propertylist and the primary HDU of the result FITS file:

- QC DET<i> GAIN<p> : The global detector gain (in electrons per ADU, specific for a readout mode) will be used for scientific purposes. This is a copy from the static flatfield map.
- QC DET<i> CHANNEL<j> GAIN<p> : The detector gain (in electrons per ADU) for each channel is also a copy from the static flatfield map
- QC DET<i> CHANNEL<j> OFFSET<p> : The offset for the detector channels and each readout mode.

where i is the detector number (1 = L/M-Band, 2 = N-Band), j is the detector channel number (1 .. 32 for the L/M-Band, 1 .. 64 for the N-Band), and p is the readout mode (HAWAII-2RG: 1 = slow readout, 2 = fast readout, Aquarius: 1 = low gain mode, 2 = high gain mode).
The detector number (used for DET<i>) will be always 1 for the L/M-Band detector and 2 for the N-Band detector. In addition, the DET<i> CHIP1 ID keyword allows to distinguish between different detector chips. This is necessary, to deal with a detector change after, for example, an instrument repair.
This recipes was originally developed to compute also an instrument flat field. This option is no more used for MATISSE. This recipes is only used to determine the BIAS map.
We recommend to keep –obsflat_type=det and –recalc_flat=0.

### 9.2.1 Input/Output files and Options

The input and output files of this recipe are summarized in the following table along with the applicable options.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| OBSDARK | dark data | YES |
| FLATFIELD | Flat field | YES |
| BADPIX | Bad Pixel Map | YES |
| NONLINEARATIRY | Non-linearity Map | YES |
| OUTPUT FILES | | |
| OBS_FLATFIELD | Observing Flat Field | |
| OPTIONS | | |
| –obsflat_type | Defines which kind of OBS_FLATFIELD is created (const = 1.0 for all pixels, det = detector flat, instr = instrument flat) | [det] |
| –recalc_flat | Flag if the detector flatfield should be recalculated | [0] |
| –gain | Default conversion gain in [e-/DU] | [0.0] |

## 9.3 mat_est_shift

This plugin calculates the distortion coefficients of the photometric and interferometric channels, and it determines the relative shifts between each photometric channel and the interferometric one. The input data are: a series of images containing the laboratory background for each pinhole and for the slit (DISTOR_HOTDARK and SPECTRA_HOTDARK), a series of images of the three pinholes (DISTOR_IMAGES), a series of spectral images (SPECTRA_IMAGES), the static bad pixel map (BADPIX), the static nonlinearity map (NONLINEARITY) and the observation specific flatfield map (OBS_FLATFIELD). This plugin produces the SHIFT map and computes the QC DET<i> PICOV<l> (the relative coverage of the interferometric channel by the photometric channels).

### 9.3.1 Input/Output files and Options

The input and output files of this recipe are summarized in the following table along with the applicable option.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| DISTOR_HOTDARK | data Hole/Background | Yes |
| DISTOR_IMAGES | data Hole/Lamp | Yes |
| SPECTRA_HOTDARK | data Slit/Foil/Background | Yes |
| SPECTRA_IMAGES | data Slit/Foil/Lamp | Yes |
| OBS_FLATFIELD | Observing Flat Field | Yes |
| BADPIX | Bad Pixel Map | Yes |
| NONLINEARITY | Nonlinearity Map | Yes |
| OUTPUT FILES | | |
| SHIFT_MAP | Distortion Map with Spectral Dispersion Law | |
| OPTIONS | | |

| | | |
|---|---|---|
| –debug | This parameter allows to produce debugging informations | [FALSE] |
| –obsCorrection | This parameter allows to correct the spectral dispersion law and the shifts between photometric and interferometric channels from observations of Be stars and sky. | [TRUE] |

## 9.4 mat_est_kappa

This plugin calculates from calibration measurements the intensity ratio, the relative shift and zoom between each photometric channel and the interferometric one for each spectral channel. The input data are: a series of images containing the laboratory background or the sky (KAPPA_HOTDARK or KAPPA_SKY), a series of images for each shutter with the internal source or the target (KAPPA_SRC or KAPPA_OBJ), the static shift map (SHIFT_MAP), the static bad pixel map (BADPIX), the static nonlinearity map (NONLINEARITY) and the observation specific flatfield map (OBS_FLATFIELD). This plugin produces the KAPPA_MATRIX and computes the QC DET<i> SHIFT<l> MEAN, QC DET<i> SHIFT<l> STDEV, QC DET<i> ZOOM<l> MEAN, QC DET<i> ZOOM<l> STDEV, QC DET<i> RATIO<l> MEAN, QC DET<i> RATIO<l> STDEV and QC DET<i> FLUX<l>.

### 9.4.1 Input/Output files

The input and output files of this recipe are summarized in the following table. Note that no options are provided for this recipe.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| KAPPA_HOTDARK | Lab Background data | Yes |
| KAPPA_SRC | Lab Lamp data | Yes |
| SHIFT_MAP | Distortion Map | Yes |
| OBS_FLATFIELD | Observing Flat Field | Yes |
| BADPIX | Bad Pixel Map | Yes |
| NONLINEARITY | Nonlinearity Map | Yes |
| OUTPUT FILES | | |
| KAPPA_MATRIX | Kappa Matrix Map | |

## 9.5 mat_cal_oifits

This master recipe calls all data reduction steps which calibrate visibility, closure phase and differential phase. Correlated flux calibration is currently not supported by the pipeline. It produces OIFITS files containing all the **calibrated** interferometric measurements. It is essentially a wrapper calling mat_cal_cphase, mat_cal_dphase, and mat_cal_vis.

### 9.5.1 Input/Output files and Option

The input files of this recipe are summarized in the following table.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| TARGET_RAW_INT | Science OIFITS | Yes |
| CALIB_RAW_INT | Calibrator OIFITS | Yes |
| OUTPUT FILES | | |
| TARGET_CAL_INT | Science OIFITS | |
| OPTIONS | | |
| –tfKeep | store interpolate function | [0] |
| –tfInterp | transfer function interpolation method (**0**: average, **2**:linear function) | [0] |
| –cumulBlock | cumul all blocks of an OB - Apply the BCD calibration | [TRUE] |

## 9.6 mat_cal_imarec

This master recipe reconstructs an image based on calibrated interferometric measurements.

It uses one or several OI-FITS files with calibrated interferometric measurements to reconstruct the image of an object.

Each OI-FITS file (DO classification TARGET_CAL_INT) must comply to the OI-FITS standard. This means they contain exactly one OI_TARGET table plus one or more of the data tables OI_VIS2, OI_T3 or OI_VIS. Each data table must refer to an OI_WAVELENGTH table present in the file. All other tables defined in the standard are not used by this plugin.

The image reconstruction method is able to use two different kinds of data:

1. The bispectrum data is used for the image reconstruction. This method part is described in details in an A&A paper (Hofmann, K.-H., Weigelt, G., & Schertl, D. 2014, A&A, 565, A48). It uses the measured calibrated squared visibilities (OI_VIS2 table) and closure phases (OI_T3 table) at several positions and wavelengths in the uv plane. From these observations the bispectrum elements and their errors are computed.

2. The complex visibilities are used for the image reconstruction. This method part uses the measured complex visibilities (OI_VIS table) at several positions and wavelengths in the uv plane.

The iterative algorithm searches for the two-dimensional image whose bispectrum and/or complex visibilities agrees best with the observed bispectrum and/or complex visibilities.

The –algo_mode option is used to select the data used by the image reconstruction algorithm:

–algo_mode=1 Bispectrum method (default)
–algo_mode=2 Complex visibilities method
–algo_mode=3 Bispectrum and complex visibilities are used

Minimization of cost function is performed with two different algorithms:

1. By the large-scale, bound-constrained nonlinear optimization algorithm ASA_CG:
   (a) Hager, W. W., & Zhang, H. 2005, SIAM J. Optim., 16, 170
   (b) Hager, W. W., & Zhang, H. 2006, SIAM J. Optim., 17, 526
2. By a Limited Memory Algorithm for Bound Constrained Optimization:
   (a) R. H. Byrd, P. Lu and J. Nocedal. A Limited Memory Algorithm for Bound Constrained Optimization, (1995), SIAM Journal on Scientific and Statistical Computing , 16, 5, pp. 1190-1208.
   (b) C. Zhu, R. H. Byrd and J. Nocedal. L-BFGS-B: Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization (1997), ACM Transactions on Mathematical Software, Vol 23, Num. 4, pp. 550 - 560.
   (c) J.L. Morales and J. Nocedal. L-BFGS-B: Remark on Algorithm 778: L-BFGS-B, FORTRAN routines for large scale bound constrained optimization (2011), to appear in ACM Transactions on Mathematical Software.

Selecting an algorithm is done with option –engine:

| | |
|---|---|
| –engine=1 | use ASA_CG for cost function minimization (default) |
| –engine=2 | use L-BFGS for cost function minimization |

Before each reconstruction the start and prior image is set to the loaded images or to a previous reconstruction. This can imply a feedback from previous reconstruction runs and therefore the whole reconstruction chain is needed and no shortcut does exist.

The result is defined by the following parameters:

| | |
|---|---|
| –fov | Field of view for the reconstructed image in [mas]. |
| –npix | Size of the reconstructed image in pixels. |
| –nbresult | Number of reconstructions written to the result file. |

The calibrated interferometric data input is filtered using a specific wavelength intervall. It is possible to specify several different wavelength intervalls:

| | |
|---|---|
| –lambda_from | Shortest wavelength for the input data in [um]. |
| –lambda_to | Longest wavelength for the input data in [um]. |
| –lambda_list | A list of lambda ranges (pairs of lower and upper wavelength [um]). |

Each reconstruction is made for a specific combination of object mask radius and regularization parameter. Both are controlled with the following parameters:

| | |
|---|---|
| –om_start | Start radius of the object mask [mas]. |

| | |
|---|---|
| –om_step | Step size for the object mask radius scan [mas]. |
| –om_count | Number of object mask radius scans. |
| –mu_start | Start value for the regularization parameter mu. |
| –mu_factor | Factor between two consecutive regularization parameter values. |
| –mu_count | Number of regularization parameter scans. |

Instead of using circular object masks with different radii, it is possible to load one or several objects masks from FITS images (DO classification OBJECT_MASK). In that case a scaling factor must be given with the –om_scale parameter. All FITS images in the primary header unit of all given OBJECT_MASK files are used in the same order as in the SOF.

The start image for the iterative algorithm and the prior (target estimate, needed for the regularization) can be provided using two parameters for each image:

| | |
|---|---|
| –start_mode | The mode for reading/creating the start image. |
| –start_param | Additional parameter for the start image creation. |
| –prior_mode | The mode for reading/creating the prior image. |
| –prior_param | Additional parameter for the prior image creation. |

The mode parameter specifies which kind of start image should be used and the param parameter gives an additional value:

| | |
|---|---|
| mode=0 param=<scale> | An image is read from a FITS file (DO classification START_IMAGE or PRIOR_IMAGE). The parameter specifies the image scale in [mas/px]. |
| mode=1 | A point source is put in the image center, no parameter is needed. |
| mode=2 param=<fwhm> | A disc with a gaussian profile is put into the image center. The parameter gives the FWHM in [mas]. |
| mode=3 param=<diameter> | A uniform disc is put into the image center. The parameter gives the diameter in [mas]. |
| mode=4 param=<diameter> | A disc with limb darkening: $I(r) = I_0 * sqrt(r_0^2 - r^2)/r_0$ is put into the center of the image. The parameter gives the diameter $(2 * r_0)$ in [mas]. |
| mode=5 param=<fwhm> | A disc with a Lorentz profile: $I(r) = 1/(1 + a^2 * r^2)^{(3/2)}$ is put into the image center. The parameter gives the FWHM in [mas] $(a = 2/FWHM * sqrt(2^{(2/3)} - 1))$. |

Internally the recipe performs two nested loops:

```
FOR EACH oradius = $oradiusStart+i*$stepSize (i = 0..$oradiusNumber-1) DO
    FOR EACH mu = $muStart*$muFactor^j (j = 0..$muNumber-1) DO
        - Reconstruct one Image with the current parameters
          oradius, mu and regularization function
        - Calculate its qrec ("quality" parameter of the reconstruction)
```

```
        - Manage a internal list of all reconstructions sorted by qrec
    ENDFOR (mu)
ENDFOR (oradius)
RETURN reconstruction with the best (i.e. smallest) qrec
```

For each internal reconstruction run, the start image and the prior (estimated target) can be selected due to two given criteria (–start_select and –prior_select):

| | |
|---|---|
| select=0 | The default image is selected. |
| select=1 | Use the previous reconstruction. |
| select=2 | The best reconstruction with the same or the previous object mask radius is selected. |
| select=3 | The best reconstruction up to now. |
| select=4 | The previous reconstruction or the best reconstruction with the previous object mask radius is selected. |
| select=5 | The default image or the previous reconstructed image with the same object mask radius is selected. |

The IRBis algorithm supports three different cost functions (see A&A paper, section 2.2.1). They can be selected using the –cost_func parameter:

| | |
|---|---|
| cost_func=1 | chi squared of the bispectrum and/or the fourier spectrum (equation 4) |
| cost_func=2 | chi squared of the bispectrum and/or the fourier phasors (equation 5) |
| cost_func=3 | chi squared of the bispectrum and/or the fourier phasors (different from equation 5) |

Because of the sparse uv coverage, different regularization terms can be incorporated into the cost function. For the regularization term one of the following methods (–reg-func) can be selected. If a negative number (for example -4) is used, the prior image is set to a constant image (no estimate of the target).

| | |
|---|---|
| reg_func=0 | no regularization |
| reg_func=1 | pixel intensity quadratic |
| reg_func=2 | maximum entropy |
| reg_func=3 | pixel difference quadratic |
| reg_func=4 | edge preserving |
| reg_func=5 | smoothness |
| reg_func=6 | quadratic Tikhonov |

During the image reconstruction run the parameter –info_flags controls the verbosity of the output of the plugin. The value of –info_flags can be a list of the comma separated keywords:

| | |
|---|---|
| none | No additional output of the plugin. |
| debug | Very specific debugging messages. |
| param | Some parameters and the results of the fit process. |
| images | Start, prior and model image. |

| | |
|---|---|
| vis2 | Used squared visibilities. |
| t3 | Used closure phases. |
| cvis | Used complex visibilities. |
| prepare | Some additional information during preparing the image reconstruction. |
| istd | For each iteration the chi2, residuals, quality of the reconstruction and cost value. |
| idetails | Detailed values for calculating chi2 and residuals. |
| fits | Some images (double and complex) are stored in FITS files. |
| result | Reconstruction results for each reconstruction run (one combination of om and mu) |
| all | Enable all messages. |

This recipe is performing a fit of some simple shapes (gaussian disc, fully darkened disc, uniform disk and Lorentz disc) to the squared visibilities and complex visibilities (after converting them into squared visibilities). It is possible to terminate the recipe execution after this step using the option –guess=1. Sometimes the initial value used for the shape fit is not ok, a better value can be provided using the option –fit_fwhm (value specified in [mas]).

This plugin is able to use calibrated interferometric input data scaled with a specific flux. It that case, this flux must be specified with the –flux option. This will rescale the input data to a standard flux of 1.0. The result is not rescaled to the specified flux!.

### 9.6.1 Input and Output files

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| TARGET_CAL_INT | OI-FITS files (arbitrary number of files, at least one) | yes |
| START_IMAGE | Start image | no |
| PRIOR_IMAGE | Prior image | no |
| OUTPUT FILES | | |
| | reconstructed image, see "Content of the output file" below | |

### 9.6.2 Options

| Option | Explanation | [default] |
|---|---|---|
| –fov | Field of view for the reconstructed image in [mas]. | [40.0] |
| –flux | Total flux of the input data (amplitude scale in T3, VIS2 and VIS) in [Jy]. | [1.0] |
| –npix | Size of the reconstructed image in pixels. Powers of 2 should be used (speeds up the FFT), but this is not enforced. | [256] |
| –nbresult | Number of reconstructions written to the result file. The best result is always stored. If 0 is given, all created reconstructions are stored in the result file. | [0] |
| –lambda_from | Shortest wavelength for the input data in [um]. | [1.0] |

| | | |
|---|---|---|
| –lambda_to | Longest wavelength for the input data in [um]. | [15.0] |
| –lambda_list | A list of lambda ranges (pairs of lower and upper wavelength). It is a sequence of comma separated floating point numbers or 'none'. This list overwrites the –lambda_from and –lambda_to parameters. | [none] |
| –engine | Specifies the optimization engine used for the image reconstruction. 1 = ASA-CG, 2 = L-BFGS-B. | [1] |
| –algo_mode | Specifies if bispectrum and/or complex visibilities are used for reconstruction. 1 = use bispectrum, 2 = use complex visibilities, 3 = use bispectrum and complex visibilities. | [1] |
| –calc_t3amp | Flag if the T3 amplitude and error should be calculated. | [1] |
| –calc_vis2f0 | Flag if an artificial squared visibility and error for f=0 should be calculated. | [1] |
| –calc_visamp | Flag if the VIS amplitude and error should be calculated. | [1] |
| –calc_visf0 | Flag if an artificial complex visibility and error for f=0 should be calculated. | [1] |
| –start_mode | The mode for reading/creating the start image. 0 = read from file, 1 = point source, 2 = Gaussian disc, 3 = uniform disc, 4 = fully darkened disc, 5 = Lorentz disc. | [0] |
| –start_param | Additional parameter for the start image creation (mode=0 -> scale [mas/px], mode=2 -> FWHM [mas], mode=3 -> diameter [mas], mode=4 -> diameter [mas], mode=5 -> FWHM [mas]). | [0.0] |
| –start_select | Mode selection for the start image. | [4] |
| –prior_mode | The mode for reading/creating the prior image. 0 = read from file, 1 = point source, 2 = Gaussian disc, 3 = uniform disc, 4 = fully darkened disc, 5 = Lorentz disc. | [0] |
| –prior_param | Additional parameter for the prior image creation (mode=0 -> scale [mas/px], mode=2 -> FWHM [mas], mode=3 -> diameter [mas], mode=4 -> diameter [mas], mode=5 -> FWHM [mas]). | [0.0] |
| –prior_select | Mode selection for the prior image. | [4] |
| –model_scale | Pixel scale of the optional model image [mas/px]. | [0.1] |
| –weight_power | Weight power (uv weight calculation). | [1.0] |
| –om_start | Start radius of the object mask [mas]. | [1.0] |
| –om_step | Step size for the object mask radius scan [mas]. | [1.0] |
| –om_count | Number of object mask radius scans. | [6] |
| –om_scale | Pixel scale of the optional object mask image [mas/px]. | [0.1] |
| –mu_start | Start value for the regularization parameter mu. | [1.0] |
| –mu_factor | Factor between two consecutive regularization parameter values. | [0.1] |
| –mu_count | Number of regularization parameter scans. | [6] |
| –reg_func | Regularisation function (0 = no regularization). | [0] |
| –reg_eps | Epsilon for regularisation function 4 (edge preserving). | [0.0] |
| –grad_tol | Tolerance value for ASA_CG. | [1e-11] |

| | | |
|---|---|---|
| –conv_scale | Scale factor for the convolution (1.0 means max baseline is used, negative value -> Gaussian PSF, positive value -> tent PSF). | [1.0] |
| –cost_func | Cost function (1, 2 or 3). | [1] |
| –cost_weight | Weight for the cost function 2 (weight between closure phase and modulus term). | [0.0] |
| –ncorr | Number of corrections for L-BFGS-B. | [5] |
| –factr | L-BFGS-B tolerance for termination test. | [10.0] |
| –pg_tol | L-BFGS-B projected gradient tolerance for termination test. | [1e-06] |
| –asa_count | Number of ASA-CG iterations per reconstruction. | [1] |
| –cc_threshold | Threshold for cross correlation. | [0.05] |
| –mjd_tol | Maximum allowed MJD difference for finding a VIS2 element for a T3 element [d]. | [0.0001] |
| –bl_tol | Maximum allowed baseline difference for finding a VIS2 element for a T3 element [m]. | [0.05] |
| –wl_tol | Maximum allowed wavelength difference wavelength filter [um]. | [0.0] |
| –precision | Number of digits after decimal point for gradient and cost value (precision < 0 : round relative, precision == 0 : no round, precision > 0 : round absolute). | [0] |
| –wiener_filter | Flag if a Wiener filter is applied to the gradient. | [0] |
| –filter_fwhm | Start value for a Gaussian gradient filter(FWHM) in [mas]. | [0.0] |
| –filter_factor | Factor between two consecutive gradient filter sizes. | [0.99] |
| –guess | Flag if only a model fit is requested. | [0] |
| –fit_fwhm | Start FWHM for the model fit [mas]. | [2.0] |
| –noise_seed | Seed value for the noise random generator. | [42] |
| –noise_factor | Noise factor (noise_sigma = error*factor) for the noise random generator. | [0.0] |
| –info_flags | Flags controlling the information printed during reconstruction. | [param] |
| –vis2_name | ASCII file for measured and reconstructed squared visibilities. | [] |
| –cp_name | ASCII file for measured and reconstructed closure phases. | [] |
| –vis_name | ASCII file for measured and reconstructed complex visibilities. | [] |

### 9.6.3 Content of the output file

The results from an image reconstruction run are stored in a FITS file with the following structure:

- The primary header unit contains the best reconstructed image and some QC values for that image.
- The image extension REC_CONV contains the convolved reconstructed image. The PSF is based on the longest baseline of the input data and may be scaled using –conv_scale (negative value -> Gaussian PSF,

positive value -> tent PSF).

- The binary table extension REC_LIST contains nbresult reconstructions sorted by the reconstruction quality (QREC).
- The image extension UV_COVERAGE contains an image showing the uv coverage (mapped to the FFT array).
- The image extension START_IMAGE contains the scaled and shifted start image.
- The image extension PRIOR_IMAGE contains the scaled and shifted prior image.
- The image extension OBJECT_MASK contains all scaled or created object masks.
- The optional image extension MODEL_IMAGE contains the scaled and shifted model image (DO classification MODEL_IMAGE).
- The optional image extension MODEL_CONV contains the convolved model image (DO classification MODEL_IMAGE).
- The binary table extension REC_VIS2 contains the measured and the reconstructed squared visibilities. This table exists when the bispectrum method is used.
- The binary table extension REC_T3 contains the measured and the reconstructed closure phases. This table exists when the bispectrum method is used.
- The binary table extension REC_VIS contains the measured and the reconstructed complex visibilities. This table exists when the complex visibilities method is used.

The file contains the following QC values (QREC, RRESBIS, RRESVIS2 and RRESCP are defined in the A&A paper):

| | |
|---|---|
| QC REC QREC | Reconstruction quality (0.0 is the ideal reconstruction). |
| QC REC COST | Cost function value from the reconstruction algorithm. |
| Depending on the input data and cost function used, the following QC values are available: | |
| QC REC CHI2BIS | Chi squared value of the bispectrum data. |
| QC REC RRESBIS | Residual ratio of the bispectrum data. |
| QC REC CHI2CP | Chi squared of the closure phases. |
| QC REC RRESCP | Residual ratio of the closure phases. |
| QC REC CHI2VIS2 | Chi squared value of the squared visibilities. |
| QC REC RRESVIS2 | Residual ratio of the squared visibilities. |
| QC REC CHI2PHI | Chi squared of the complex visibility phase. |
| QC REC RRESPHI | Residual ratio of the complex visibility phase. |

The binary table (EXTNAME = REC_LIST) contains the columns:

| | | |
|---|---|---|
| OM | D(1) | Object mask radius in [mas]. |
| MU | D(1) | Regularization parameter value. |
| NBIT | I(1) | Number of iterations. |
| REC | D(N,N) | Reconstructed image. |
| CONV | D(N,N) | Convolved reconstruction. |

Additionally, the QC parameters are given in respective columns (without QC REC name prefix) between NBIT and REC columns.

The rows of this table are sorted with ascending reconstruction quality (QREC). The images from the first row are repeated in the primary header (columns QREC to RRESCP as QC parameters and REC as image) and the second header unit (CONV as image).

The binary table (EXTNAME = REC_VIS2) contains the columns:

| | |
|---|---|
| LAMBDA D(1) | Wavelength of the measurement [m]. |
| UCOORD D(1) | U-coordinate as spatial frequency ([1/mas]). |
| VCOORD D(1) | V-coordinate as spatial frequency ([1/mas]). |
| BL D(1) | Original baseline length ([m]). |
| VIS2DATA D(1) | Measured squared visibility. |
| VIS2ERR D(1) | Error of the measured squared visibility. |
| GDF D(1) | Fitted Gaussian disc (squared visibility). |
| UDF D(1) | Fitted uniform disc (squared visibility). |
| FDDF D(1) | Fitted fully darkened disc (squared visibility). |
| LDF D(1) | Fitted Lorentz disc (squared visibility). |
| VIS2DATAi D(1) | Reconstructed squared visibility for reconstruction number i (row in REC_LIST). |

If a file name is given with parameter –vis2_name, an ASCII-file is written with the same columns including a first column containing a sequence number.

The binary table (EXTNAME = REC_T3) has the following columns:

| | | |
|---|---|---|
| LAMBDA | D(1) | Wavelength of the measurement [m]. |
| U1COORD | D(1) | U-coordinate, first baseline ([1/mas]). |
| V1COORD | D(1) | V-coordinate, first baseline ([1/mas]). |
| U2COORD | D(1) | U-coordinate, second baseline ([1/mas]). |
| V2COORD | D(1) | V-coordinate, second baseline ([1/mas]). |
| T3AMP | D(1) | Measured amplitude. |
| T3AMPERR | D(1) | Error of the measured amplitude. |
| T3PHI | D(1) | Measured closure phase [radian]. |
| T3PHIERR | D(1) | Error of the measured closure phase [radian]. |
| BISERR | D(1) | Measured bispectrum error. |
| T3AMPi | D(1) | Reconstructed amplitude for reconstruction number i (row in REC_LIST). |
| T3PHIi | D(1) | Reconstructed closure phase for reconstruction number i (row in REC_LIST). |

If a file name is given with parameter –cp_name, an ASCII file is written with the same columns including a first column containing a sequence number.

The binary table (EXTNAME = REC_VIS) has the following columns:

| | | |
|---|---|---|
| LAMBDA | D(1) | Wavelength of the measurement [m]. |
| UCOORD | D(1) | U-coordinate as spatial frequency ([1/mas]). |
| VCOORD | D(1) | V-coordinate as spatial frequency ([1/mas]). |
| BL | D(1) | Original baseline length ([m]). |
| VISAMP | D(1) | Amplitude of the measured complex visibility. |
| VISAMPERR | D(1) | Amplitude error of the measured complex visibility. |
| VISPHI | D(1) | Phase of the measured complex visibility [radian]. |
| VISPHIERR | D(1) | Phase error of the measured complex visibility [radian]. |
| GDF | D(1) | Fitted Gaussian disc (visibility). |
| UDF | D(1) | Fitted uniform disc (visibility). |
| FDDF | D(1) | Fitted fully darkened disc (visibility). |
| LDF | D(1) | Fitted Lorentz disc (visibility). |
| VISAMPi | D(1) | Amplitude of the reconstructed complex visibility for reconstruction number i (row in REC_LIST). |
| VISPHIi | D(1) | Phase of the reconstructed complex visibility for reconstruction number i (row in REC_LIST). |

If a file name is given with parameter –vis_name, an ASCII-file is written with the same columns including a column containing a sequence number.

## 9.7 mat_cal_image

Calibration of the raw images. This plugin uses static calibration maps and the observation flatfield to convert raw interferograms into calibrated interferograms. The algorithm for this calibration is able to compensate for detector and instrument effects: individual pixel gain, nonlinearity, defect pixels and optical distortion.

It is possible to check if the TARTYP in the FITS file is correct or not. This is done by analyzing the timing information and intensity of a central part of the frames (center of the interferometric channel). If a shift between the TARTYP in the file and the estimated TARTYP is found, it is possible to change TIME, LOCALOPD, STEPPING_PHASE and TARTYP in the result file. Detection and modification is controlled using the options --tartyp, --excess_count_lm, --excess_count_n and --excess_count_list. The default values do **not** change the output. The following values for --tartyp are proposed:

--tartyp=57 The TARTYP[6] follows the rule N*S+U+N*T+U, which is the currently used schema in MATISSE. The TARTYP is estimated using the intensity and only the TARTYP column is changed. Frames tagged with U are retagged as T or S. It is important to notice that the OPD modulator and detector are in perfect synchronization, only the chopper is slightly shifted (the measured chopper period is not exactly the specified chopper period).

--tartyp=249 Like 57, but in addition the TIME, LOCALOPD and STEPPING_PHASE columns are modified. It is important to notice that in such a case frames were lost (before 2018) which means the relation of the timetable which contains these values and the frames take is shifted which makes it necessary to modify all these columns.

Instead of the estimated TARTYP shift, it is possible to specify the shift with the following options (it is still necessary to use --tartyp=57 or --tartyp=249):

--excess_count_lm This is the shift which should be used when modifying the columns for a LM-Band exposure (detector 1, HAWAII-2RG).

--excess_count_n This is the shift which should be used when modifying the columns for a N-Band exposure (detector 2, Aquarius).

--excess_count_list If several exposures are compensated using one SOF (the standard use case for mat_raw_estimates), for each exposure (identified by ESO TPL EXPNO keyword) the shift can be specified. This will overwrite the estimated TARTYP shift. An example is: --excess_count_list=3,1,6,0,7,-1 it means for exposure 3 a shift of 1 is applied, exposure 6 is not modified and for exposure 7 the estimated shift is applied.

For the excess count, three kinds of values do exist:

-1 Use the estimated TARTYP shift derived from the timing or intensity (default value).

0 The TARTYP and other columns are **not** modified.

X The TARTYP and other columns are modified by applying a shift of X frames, the estimated TARTYP shift is **not** applied.

### 9.7.1 Input and Output files

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| CALIB_SRC_RAW | Lab Lamp data | Yes (if lab data) |
| HOT_DARK | Lab Background data | Yes (if lab data) |
| SKY_RAW | Sky data | No |

[6] where S means Sky, T means Target, U means Undefined, and N ist the number of frames

| | | |
|---|---|---|
| CALIB_RAW | Calibrator data | Yes (if calibrator) |
| TARGET_RAW | Target data | Yes (if target) |
| SHIFT_MAP | Distortion Map | Yes |
| OBS_FLATFIELD | Observing Flat Field | Yes |
| BADPIX | Bad Pixel Map | Yes |
| NONLINEARITY | Nonlinearity Map | Yes |
| OUTPUT FILES | | |
| DO category | Explanation | |
| TARGET_CAL | Calibrated data (if target) | |
| CALIB_CAL | Calibrated data (if calibrator or lab data) | |

### 9.7.2 Options

The table below summarizes the different option parameter that can be used with the mat_cal_image recipe. A more detailed description of the calibration maps and these option parameter can be found in the next section, sect. 9.7.3.

| Option parameter | Explanation | Default |
|---|---|---|
| --compensate | Defines which kind of compensation should be applied (see table in section 9.7.3 for more details | [pb,cb,rb,nl,if,bp,od] |
| --gain | Default conversion gain in [e-/DU] | [0.0] |
| --reduce | Flag if the reference sub-windows should be removed from the result | [TRUE] |
| --ioi | images of interest: <first>,<count> | [0,0] |
| --tartyp | TARTYP[7] estimation (**flag**, please add the following values for the desired effect: **0** = none, **1** = N*S+U+N*T+U, **2** = show intensity, **4** = show correlation, **8** = estimate TARTYP, **16** = change TARTYP, **32** = exchange U with S or T, **64** = change TIME, **128** = change LOCALOPD and STEPPING_PHASE, **256** = show the extracted exposure setup, **512** = show the first chopping cycle, **1024** = show the extracted timetable, **2048** = show the modified frame info, **4096** = use shift calculated from square wave fit, **8192** = use shift calculated from timing) | [57] |
| --excess_count_lm | Excess frames mistakenly produced before first TIM-Board trigger (LM-Band) | [-1] |
| --excess_count_n | Excess frames mistakenly produced before first TIM-Board trigger (N-Band) | [-1] |
| --excess_count_list | Specification of an excess count for each exposure number (ESO TPL EXPNO keyword, excess count pair) as a list of numbers separate by comma. | [none] |

[7] where S means Sky, T means Target, U means Undefined, and N ist the number of frames

### 9.7.3 Calibration Maps

The provided *calibration maps* specify which part of the calibration can be performed. Applying the actual calibration is triggered by the use of the option parameter –compensate.
- If the static bad pixel map (DO classification BADPIX) is provided, the intensity of the defect pixels can be replaced with an interpolated value derived from the neighbouring pixels.
- If the static flatfield map (DO classification FLATFIELD) is provided, the individual pixel gain can be compensated.
- If the static nonlinearity map (DO classification NONLINEARITY) is provided, the individual pixel nonlinearity can be compensated.
- If the observation (setup) specific flatfield map (DO classification OBS_FLATFIELD) is provided, the individual pixel offset and the instrument flatfield is compensated.
- If the optical distortion map (DO classification SHIFT_MAP) is provided, the warping of the optical mapping is compensated.

Each calibration map provides only the available compensation method. The parameter –compensate is used to define which method should be applied. This parameter is a comma separated list of keywords which default value is [pb,gb,nl,if,bp]. The following table details the application of these keywords. For more information regarding the other option parameters please refer to the esorex –man-page of the recipe.

| Parameter | Definition | Comments |
|---|---|---|
| none | no compensation at all | |
| all | all compensations possible | depends on the provided maps |
| pb | subtract pixel bias | OBS_FLATFIELD is needed |
| gb | subtract global bias | OBS_FLATFIELD is needed |
| cb | subtract detector channel bias | OBS_FLATFIELD is needed |
| rb | subtract row bias | OBS_FLATFIELD is needed |
| ct | subtract detector channel crosstalk | OBS_FLATFIELD is needed |
| nl | nonlinearity compensation | NONLINEARITY is needed |
| if | divide by instrument flat | OBS_FLATFIELD is needed |
| df | divide by detector flat | FLATFIELD is needed |
| bp | bad pixel interpolation | BADPIX is needed |
| el | convert to electrons | conversion factor from parameter–gain, FLATFIELD or OBS_FLATFIELD |
| od | remove optical distortion | SHIFT_MAP is needed |

## 9.8 mat_ext_beams: Extraction of the Photometry

In the SiPhot mode, this recipe transforms the photometric beams of each frame from the photometric channel into the interferometric one using the Kappa Matrix. In case of HighSens raw data, the mapping is not necessary. The average sky, measured via chopping, is subtracted from the photometric beam intensity to get the photometric intensity produced by the target alone.

### 9.8.1 Input and Output files

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| TARGET_CAL | Cal data (if target) | Yes (if target) |
| CALIB_CAL | Cal data (if calibrator) | Yes (if calibrator or lab data |
| KAPPA_MATRIX | Kappa Matrix Map | Yes (if SiPhot) |
| OUTPUT FILES | | |
| PHOT_BEAMS | Photometric Beams in interferometric channel | |

### 9.8.2 Options

| DO category | explanation | default |
|---|---|---|
| –hampelFilterKernel | Only for L/M band. Apply a temporal Hampel filter to all pixels before deriving the photometry. This filter improves the photometric estimation in case of faint stars (< 5Jy in L band with AT). The parameter fixes the size of the kernel of the filter (kernel=0 is recommended). | [0] |
| –spectralBinning | Spectral Binning in pixels. Should be an odd integer number. If an even integer is given, the pipeline will select the next odd integer. | [1] |
| –replaceTel | Replace Photometry of one telescope by the mean of the 3 others. An integer with 2 digits is expected. The first digit is used for UTs and the second for ATs. 0: none, 1: AT1/UT1, 2: AT2/UT2, 3: AT3/UT3, 4: AT4/UT4. For example: replaceTel=01 means replace AT1, replaceTel=32 means replace UT3 and AT2. | [3] |

## 9.9 mat_est_corr

This recipe derives the uncontaminated complex correlated flux from each recorded interferogram. The uncontaminated complex correlated flux is the Fourier Transform of the interferogram without its huge low frequency part which contaminates the fringes peaks. This contamination can be removed by applying the OPD modulation algorithm (useOpdMod=TRUE). Alternatively, the user can choose to apply a coherent integration of the frames over a certain duration defined by the parameter coherentIntegTime. In this case, the choice for useOpdMod is ignored.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required/Default |
| TARGET_CAL | Cal data (if target) | Yes (if target) |
| CALIB_CAL | Cal data (if calibrator) | Yes (if calibrator or lab data) |
| OUTPUT FILES | | |
| OBJ_CORR_FLUX | Correlated flux | |
| OPTIONS | | |
| –useOpdMod | useOpdMod option. If TRUE, OPD modulation algorithm applied | [FALSE] |
| –coherentIntegTime | Specify a coherent integration time (in s) | [0.0] |

| | | |
|---|---|---|
| –spectralBinning | Spectral Binning in pixels. Should be an odd integer number. If an even integer is given, the pipeline will select the next odd integer. | [1] |

## 9.10 mat_est_opd

This plugin estimates the OPD for each baseline for each frame. It takes as input the correlated flux.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| OBJ_CORR_FLUX | Correlated flux | Yes |
| OUTPUT FILES | | |
| OI_OPDWVPO | OPD estimation | |

## 9.11 mat_proc_incoher

This recipe calculates the average power and bi-spectrum from all selected frames. It derives raw squared visibility and raw closure phase. As a by-product, this recipe computes the spectrum. It is also possible to compute correlated flux instead of the squared visibility with the option parameter –corrFlux set to TRUE. In this case, PHOT_BEAMS data are not required.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| OBJ_CORR_FLUX<br>PHOT_BEAMS | Correlated Flux<br>Photometric Beam | Yes<br>No (corrFlux parameter) |
| OUTPUT FILES | | |
| RAW_VIS2<br>RAW_CPHASE<br>RAW_SPECTRUM | raw squared visibility<br>raw closure phase<br>spectrum | |
| OPTIONS | | |
| –corrFlux | If TRUE, correlated flux are computed instead of squared visibility | [FALSE] |

## 9.12 mat_proc_coher

This plugin runs the coherent processing. It computes the raw differential phase and visibility. See Section 11.6 for a description of the different algorithms.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| OBJ_CORR_FLUX<br>PHOT_BEAMS | Correlated Flux<br>Photometric Beam | Yes<br>No (corrFlux parameter) |

| OUTPUT FILES | | |
|---|---|---|
| RAW_DPHASE | raw differential phase and visibility | |
| OPTIONS | | |
| –coherentAlgo | Estimation Algorithm (1: AMBER like Method, 2: CRAL Coherent Integration Method) | [1] |

### 9.13 mat_merge_results

This plugin merges the intermediate results (RAW_VIS2, RAW_CPHASE, RAW_CPHASE, RAW_SPECTRUM, RAW_TF2.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| RAW_VIS2 | raw OI_VIS2 | No |
| RAW_CPHASE | raw OI_T3 | No |
| RAW_DPHASE | raw OI_VIS | No |
| RAW_SPECTRUM | OI_FLUX | No |
| RAW_TF2 | Transfer function | No |
| CAL_VIS2 | calibrated OI_VIS2 | No |
| CAL_CPHASE | calibrated OI_T3 | No |
| CAL_DPHASE | calibrated OI_VIS | No |
| TARGET_RAW_INT | raw OIFITS | No |
| OUTPUT FILES | | |
| TARGET_CAL_INT | Calibrated OIFITS | |
| TARGET_RAW_INT | Raw OIFITS (Target) | |
| CALIB_RAW_INT | Raw OIFITS (Calibrator) | |
| OPTIONS | | |
| –cumulBlock | cumul all blocks of an OB | [FALSE] |
| –bcdMode | Select frames with BCD OUT-OUT (0) or frames with BCD IN-IN (1) or frames with BCD IN-OUT (2) or frames with BCD OUT-IN (3) | [0] |

### 9.14 mat_cal_cphase

This recipe is for the Calibration of the phase closure.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |

| | | |
|---|---|---|
| TARGET_RAW_INT<br>CALIB_RAW_INT | Science OIFITS<br>Calibrator OIFITS | Yes<br>Yes |
| OUTPUT FILES | | |
| CAL_CPHASE | OIFITS with OI_T3 binary table | |

## 9.15 mat_cal_dphase

This recipe is for the Calibration of the differential phase.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| TARGET_RAW_INT<br>CALIB_RAW_INT | Science OIFITS<br>Calibrator OIFITS | Yes<br>Yes |
| OUTPUT FILES | | |
| CAL_DPHASE | OIFITS with OI_VIS binary table | |

## 9.16 mat_im_basic

This plugin calculates a small set of detector monitoring QC1 parameters.

A specialized template (MATISSE_gen_cal_imbasic) creates one series of cold dark frames (100 raw frames in a FITS file). This file is then used to calculate the detector channel offsets, detector channel noise and power spectrum peaks. Since the template is used before each observing night, the QC1 parameter give a simple but quick status of an instrument detector.

This plugin uses a FITS file with 100 cold dark frames (IM_COLD) for a specific detector.
In addition, a bad pixel (PRO.CATG = BADPIX) and/or a flatfield (PRO.CATG = FLATFIELD) static calibration map can be specified.

This plugin creates an output file (PRO.CATG = IM_BASIC) which contains:

1. an empty HDU containing the QC1 parameters
2. a binary table (IM_STATISTICS) containing the pixel statistics for each detector channel in a row
3. a binary table (IM_FAST_POWERSPECTRUM) containing the fast 1-d power spectrum for each detector channel in a row
4. a binary table (IM_SLOW_POWERSPECTRUM) containing the slow 1-d power spectrum for each detector channel in a row

The following QC1 parameters will be calculated and stored in a cpl_propertylist and the primary HDU of the result FITS file:

| | |
|---|---|
| QC DET<i> CHANNEL<j> OFFSET<p> | The offset for the detector channels. |
| QC DET<i> CHANNEL<j> RON<p> | The read-out noise is given for each detector channel. |
| QC DET<i> PS FAST PEAK<k> FREQ | The frequency for the k-th strongest peak in the fast (horizontal) 1-d power spectrum. |
| QC DET<i> PS FAST PEAK<k> POWER | The power for the k-th strongest peak in the fast (horizontal) 1-d power spectrum. |
| QC DET<i> PS SLOW PEAK<k> FREQ | The frequency for the k-th strongest peak in the slow (vertical) 1-d power spectrum. |
| QC DET<i> PS SLOW PEAK<k> POWER | The power for the k-th strongest peak in the slow (vertical) 1-d power spectrum. |

Where i is the detector number (1 = L/M-Band, 2 = N-Band), j is the detector channel number (1 .. 32 for the L/M-Band, 1 .. 64 for the N-Band), and p is the readout mode (HAWAII-2RG: 1 = slow readout, 2 = fast readout, Aquarius: 1 = low gain mode, 2 = high gain mode).
The detector number (used for DET<i>) will be always 1 for the L/M-Band detector and 2 for the N-Band detector.
In addition, the DET<i> CHIP1 ID keyword allows to distinguish between different detector chips. This is necessary, to deal with a detector change after, for example, an instrument repair. This will also result in a basic detector monitoring map which is only valid for a specific detector chip.

### 9.16.1 Input files

| DO category | Explanation | Required |
|---|---|---|
| IM_COLD | Raw cold frames | Yes |
| BADPIX | Static bad pixel map | No |
| FLATFIELD | Static flatfield map | No |

### 9.16.2 Output files

| DO category | Explanation |
|---|---|
| IM_BASIC | Basic instrument monitoring map |

### 9.16.3 Options

| Option | Explanation | [default] |
|---|---|---|
| –nditskip | number of skipped frames. | [0] |
| –cosmics | flag if cosmics should be detected. | [1] |
| –gain | global detector gain [e-/DU]. | [1.0] |
| –window | window function for 1-d power spectrum. | [3] |
| –hfreq | horizontal base frequency. | [0.0] |

| | | |
|---|---|---|
| –vfreq | vertical base frequency. | [0.0] |
| –expert | expert flag. | [0] |

## 9.17 mat_im_extended

Calculates an extended detector monitoring map

This plugin uses one series of cold dark and one series of flatfield frames to calculate extended detector monitoring information:

- The detector and detector channel specific gain which can be compared with the gain values calculated by the mat_cal_det plugin (comparison of the the QC1 values).
- The detector channel specific offset and noise.
- The Allan Variance give a hint about the long term stability of the detector and the detector related instrument part.
- The autocorrelation specifies the inter-pixel dependencies which influences the spatial and spectral resolution of the raw data.

Since the Allan Variance should cover a time span larger than a science observation, the series of the cold dark and flatfield frames will cover 10 minutes (3000 raw frames for each detector, both detectors will be slowed down to 5 frames per second):

One series of cold darks (IM_DARK) is classified by:

```
DPR.CATG  =  CALIB
DPR.TYPE  =  DARK,IME
DPR.TECH  =  IMAGE
TPL.ID    =  MATISSE_gen_cal_imext_(L_FAST|N_HIGH)
```

One series of flatfield frames (IM_FLAT) is classified by:

```
DPR.CATG  =  CALIB
DPR.TYPE  =  FLAT,IME
DPR.TECH  =  IMAGE
TPL.ID    =  MATISSE_gen_cal_imext_(L_FAST|N_HIGH)
```

In addition, the bad pixel (PRO.CATG = BADPIX) static calibration map is needed.

The pixel statistics for the cold dark frames contains the median (not mean!) intensity and the variance. The values for the flatfield frames are corrected by subtracting the median and variance values for the corresponding cold dark series. This means, that the flatfield values should represent directly the photon statistics. Since

some parts of the detector are not illuminated, the variance values for these parts could be negative because the cold dark and flatfield frames show nearly the same median and variance. This function is equivalent to the corresponding function in the mat_cal_det plugin except that only one series of cold dark and one series of flatfield frames are used.

The offset and noise of a detector channel can indicate the health status of the detector and the electronics. The function is equivalent to the function in the mat_im_basic plugin.

Since the pixel statistics for the flatfield series does not contain the cold dark statistics, the division of the median by the variance directly gives the conversion gain. As many pixels as possible were used to calculate the detector and detector channel specific gain. This function is equivalent to the corresponding function in the mat_cal_det plugin.

This plugin creates an output file (PRO.CATG = IM_EXTENDED) which contains an empty HDU with all QC1 parameters.

### 9.17.1 QC1 parameters

The following QC1 parameters will be calculated and stored in the primary HDU of the result FITS file:

| | |
|---|---|
| QC DET<i> GAIN<p> | The global detector gain (in electrons per ADU) will be used for scientific purposes. |
| QC DET<i> RON<p> | The global read-out noise (in electrons). |
| QC DET<i> CHANNEL<j> GAIN<p> | The detector channel specific gain (in electrons per ADU) will be used to detect detector and read-out electronics problems. |
| QC DET<i> CHANNEL<j> OFFSET<p> | The offset for the detector channels. |
| QC DET<i> CHANNEL<j> RON<p> | The read-out noise is given for each detector channel. |
| GC DET<i> HACORR<p> | The Autocorrelation coefficient between two horizontally adjacent pixels. |
| GC DET<i> VACORR<p> | The Autocorrelation coefficient between two vertically adjacent pixels. |
| GC DET<i> AVAR MIN | Position of the turning point of the Allan variance plot. |

Where i is the detector number (1 = L/M-Band, 2 = N-Band), j is the detector channel number (1 .. 32 for the L/M-Band, 1 .. 64 for the N-Band), and p is the readout mode (HAWAII-2RG: 1 = slow readout, 2 = fast readout, Aquarius: 1 = low gain mode, 2 = high gain mode).
The detector number (used for DET<i>) will be always 1 for the L/M-Band detector and 2 for the N-Band detector.
In addition, the DET<i> CHIP1 ID keyword allows to distinguish between different detector chips. This is necessary, to deal with a detector change after, for example, an instrument repair.

### 9.17.2 Input files

| DO category | Explanation | Required |
|---|---|---|
| IM_DARK | Raw cold frames | Yes |
| IM_FLAT | Raw exposed frames | Yes |
| BADPIX | Static bad pixel map | No |

#### 9.17.3 Output files

| DO category | Explanation |
|---|---|
| IM_EXTENDED | Extended instrument monitoring map |

#### 9.17.4 Options

| Option | Explanation | [default] |
|---|---|---|
| –nditskip | number of skipped frames. | [0] |
| –cosmics | flag is cosmics should be detected. | [0] |
| –roi | region of interest: <x>,<y>,<nx>,<ny>. | [-1,-1,-1,-1] |
| –expert | expert flag. | [0] |

### 9.18 mat_im_rem

Calculates a detector remanence map

The corresponding MATISSE_gen_cal_imrem template creates the input FITS file. Since it is expected that the remanence will not change during the lifetime of the detector, this functionality is only needed during the detector characterization and later on once a year or after exchanging a detector.
The template will create a FITS file containing interwoven dark and flatfield raw frames (DO classification IM_PERIODIC). The fast cold shutter of the instrument will be used to generate a FITS file where a fixed number of flatfield frames (open shutter) is followed by a fixed number of cold dark frames (closed shutter) and so on. In total at least 5 such pairs are needed. It must be noted, that these raw frames are taken as a continuous sequence and they are stored in one FITS file!

In addition, the bad pixel (PRO.CATG = BADPIX) static calibration map is needed.

Since the global illumination of the MATISSE detectors is non-uniform, all calculations are done on individual pixels which show a sufficient dark-to-bright difference. This means for each usable pixel all possible dark-to-bright and bright-to-dark remanence values (one value for each transition) are calculated and used to calculate a transition specific mean (we will have about five dark-to-bright and five bright-to-dark transitions in the input file). The smallest remanence value for a transition type is used to quantify the detector remanence, because if the shutter does not work perfectly, the remanence values are higher than for an ideal shutter.

The sequence of pixel intensities should show a clear square wave form. These intensities are sorted (increasing value) and the middle value between the 5 and 95 percent of the intensity values (5 and 95 percent quantiles) is used as a threshold. All intensities (in the original order!) are classified as dark (below the threshold) or bright

(above the threshold). For a sequence of at least 10 consecutive darks or brights the average is calculated by using the second half of these values (ignoring the last value, because it may be influenced by a slow shutter). For each transition, the first intensity (the first dark intensity after a sequence of bright intensities or the first bright intensity after a sequence of dark intensities) and the difference of the intensity averages is used to calculate the remanence for this transition.

This plugin creates an output file (PRO.CATG = IM_REMANENCE) which contains an empty HDU with all QC1 parameters.

The following QC1 parameters will be calculated and stored in the primary HDU of the result FITS file:

| | |
|---|---|
| QC DET<i> REM TODARK | The remanence value for switching from a bright to a dark frame. |
| QC DET<i> REM TOBRIGHT | The remanence value for switching from dark to bright frames. |

Where i is the detector number (1 = L/M-Band, 2 = N-Band).
The detector number (used for DET<i>) will be always 1 for the L/M-Band detector and 2 for the N-Band detector.
In addition, the DET<i> CHIP1 ID keyword allows to distinguish between different detector chips. This is necessary, to deal with a detector change after, for example, an instrument repair.

## 9.19 mat_cal_det

Calculates a bad pixel map, a flatfield map and a non-linearity map.

This plugin uses several series of cold dark and flatfield exposures to create the static calibration maps for both MATISSE detectors. Due to the two different detectors (HAWAII-2RG and Aquarius) the methods to derive the non-linearity and flatfield maps are detector dependent. To be more precise, the plugin is able to calculate a flux dependent non-linearity map (used for the Aquarius) and a intensity dependent non-linearity map (used for the HAWAII-2RG). The content of the input files in the SOF determine which kind of non-linearity map is created. OBs do exist for both detectors which take the necessary data automatically.

For the intensity dependent non-linearity, the SOF must contain:

- At least five pairs of cold darks and flatfield exposures with different DIT.
- The DO classification of the cold darks is DARK.
- The DO classification of the flatfield exposures is FLAT.

For the flux dependent non-linearity, the SOF must contain (in this order, all with the same DIT):

- The cold dark exposure, DO classification DARK.
- A flatfield exposure with intensities going up to 1/2 of full well capacity, DO classification FLAT. The template achieves this by using a polarizer.

- A flatfield exposure with shutter 1 open and all other shutters closed, DO classification FLAT.
- A flatfield exposure with shutter 2 open and all other shutters closed, DO classification FLAT.
- A flatfield exposure with shutter 3 open and all other shutters closed, DO classification FLAT.
- A flatfield exposure with shutter 4 open and all other shutters closed, DO classification FLAT.
- A flatfield exposure with shutter 1 and 3 open and shutter 2 and 4 closed, DO classification FLAT.
- A flatfield exposure with shutter 1, 2 and 3 open and shutter 4 closed, DO classification FLAT.
- A flatfield exposure with all shutters open, DO classification FLAT.

The static calibration maps cover the whole detector, but due to the partial illumination of the detector, the flat field map contains the default (1.0) value for all not illuminated pixels. The non-linearity coefficients (intensity dependent non-linearity) are set to neutral values for these pixels. The global gain (conversion factor from electrons to DU) of a detector and the detector channel specific gain is calculated which are used later on for the scientific data reduction.

Since the illumination of the pixels is crucial for the quality of the static calibration maps, the current of the infrared lamp must be selected with care:

- For the HAWAII-2RG detector, the lamp current should be set to 4.5 A and the pinhole removed. The DIT values and instrument setup used by the MATISSE_gen_cal_det_L_SLOW and MATISSE_gen_cal_det_L_FAST OBs are adjusted to this lamp setup.
- For the Aquarius detector, the non-linearity map is valid for a specific DIT only. The infrared lamp current must be set manually following the procedure:
    1. Set the Aquarius readout mode to SCI-HIGH-GAIN or SCI-LOW-GAIN.
    2. Select the DIT.
    3. Setup the instrument to:
       INS SFN = DIAPH
       INS PIN = PHOTO
       INS FIN = OPEN
       INS PON = DARK
       INS DIN = HIGH
    4. Wait until the image in the RTD is stable and take it as reference image (cold dark).
    5. Open the polarizer wheel:
       INS PON = OPEN
    6. Subtract the reference from the image.
    7. Adjust the infrared lamp current so that the brightest pixels have an intensity (dark subtracted!) of about 45000 DU.

The process for creating the bad pixel map, flatfield map and nonlinearity map can be controlled to get different kinds of these static calibration maps. This is controlled using the three options --bptype, --fftype and --nltype:

| | |
|---|---|
| --bptype=0 | This means the standard bad pixel map is created. |
| --bptype=1 | The bad pixel map is empty, no pixel is marked as bad. |
| --bptype=2 | A merged bad pixel map is created by using the standard map and additionally given maps (DO classification BADPIX in the SOF file). A pixel is a bad pixel if it is in about 25 percent of the maps marked as bad. All added bad pixel maps should be created with --bptype=0. |
| --fftype=0 | This means the standard flatfield map is created. |
| --fftype=1 | The flatfield map contains a constant gain of 1.0. |
| --fftype=2 | An average flatfield map is created by using the standard map and additionally given maps (DO classification FLATFIELD in the SOF file). All added flatfield maps should be created with --fftype=0. |
| --nltype=0 | This means an individual nonlinearity response for each pixel is created (default for HAWAII-2RG). |
| --ntlype=1 | The nonlinearity map represents a strict linear behaviour. |
| --nltype=2 | An average nonlinearity behaviour is calculated based in the individual response (default for Aquarius). |

Using --bptype=2 and/or --fftype=2 needs to manually create a SOF in order to add the already existing badpixel and/or flatfield maps.

This plugin will create three static calibration files.

1. The first FITS file (PRO.CATG = BADPIX) contains the bad pixel map for a specific detector. This detector is identified by the keywords in the primary header which are copied from the first used raw data file (the FITS file with the cold dark frames for the shortest exposure time).

   | | |
   |---|---|
   | Primary HDU | It contains the bad pixel map as an 8-bit image covering the whole detector. Good pixels are marked by a 0, bad pixels are identified by a 1 (binary image). |
   | QC1 parameters | In the header the corresponding QC1 parameters are stored. |

2. The second FITS file (PRO.CATG = FLATFIELD) contains the detector flatfield map for a specific detector. This flatfield is different from the observation specific flatfield, since the inhomogeneous illumination is compensated. The detector is identified by the keywords in the primary header which are copied from the first used raw data file (the FITS file with the cold dark frames for the shortest exposure time).

   | | |
   |---|---|
   | Primary HDU | It contains the flatfield map as an 32-bit floating point image covering the whole detector. |
   | FFSTDERR | The standard error map (one value for each pixel) is stored in an image extension. |

   For some pixels no valid flatfield value could be calculated due to a lack of sufficient illumination. For them, the flatfield value is set to 1.0 and the standard error to 0.0. The FITS file will contain the QC1 parameters describing the conversion factor between electrons and DU for the whole detector and for each

detector channel. These values will be copied into the observation specific flatfield (OBS_FLATFIELD) by the mat_est_flat plugin. The mat_cal_image plugin will then use the conversion gain to convert the pixel values from DU into electrons.

3. The third FITS file (PRO.CATG = NONLINEARITY) contains the detector specific non-linearity map consisting of the function coefficients for each pixel or a global non-linearity compensation polyline. The detector is identified by the keywords in the primary header which are copied from the first used raw data file (the FITS file with the cold dark frames for the shortest exposure time). In addition the FITS file contains in the header a function and the related coefficients which describe the global behaviour (since pipeline version 1.4.8).

| | |
|---|---|
| Primary HDU | It contains an image with the non-linearity compensation limits for all pixels. They describe the intensity until for a given pixel the non-linearity compensation works well due to the available data. |
| NLSTDERR | The quality of the compensation is specified as a standard error for each pixel. It describes the relative deviation of the function fit/global polyline and the individual pixel data. |
| NLHIST | This is a 2-d histogram showing the relation between measured intensity (x-axis) and calibrated intensity (y-axis) for all pixels. |
| NLCOEFF | This is an image cube containing the non-linearity function coefficients for all pixels. If the illumination of a pixel prohibits the fitting of a function, the coefficients are set to neutral values. This extension does exist in intensity dependent non-linearity maps. |
| NLRCMAP | This is a vector describing the global non-linearity compensation polyline. Each value describes the calibrated intensity for a measured intensity which is 64 times the vector index (starting at 0). This extension does exist in flux dependent non-linearity maps. |

The FITS file will contain the same QC1 parameters describing the conversion factor between electrons and DU for the whole detector and for each detector channel as the flatfield map.

Since the nonlinearity map for the HAWAII-II detector (LM-Band) describes the nonlinearity response for each individual pixel, the usable range depends on the maximum illumination on a pixel. It is possible to estimate an average nonlinearity response from all pixels and use this result for all pixels. This will lead to a nonlinearity compensation valid for all pixels even if they are not illuminated well. The plugin option --nltype allows selecting between the standard nonlinearity estimation for each pixel (default value 0) or estimating an average (global) nonlinearity response (value 1). The use of the global nonlinearity response will also lead to a very high usable intensity range, equal for all pixels.

The following QC1 parameters will be calculated and stored in a cpl_propertylist and the primary HDU of the result FITS file:

| | |
|---|---|
| - QC DET<i> GAIN<p> | The global detector gain (in electrons per ADU, specific for a readout mode) will be used for scientific purposes. |
| - QC DET<i> RON<p> | The global read-out noise (in electrons). |

| | |
|---|---|
| - GC DET<i> HACORR<p> | The Autocorrelation coefficient between two horizontally adjacent pixels. |
| - GC DET<i> VACORR<p> | The Autocorrelation coefficient between two vertically adjacent pixels. |
| - QC DET<i> CHANNEL<j> GAIN<p> | The detector channel specific gain (in electrons per ADU, specific for a readout mode) will be used to detect detector and read-out electronics problems. |
| - QC DET<i> CHANNEL<j> OFFSET<p> | The offset for the detector channels. |
| - QC DET<i> CHANNEL<j> RON<p> | The read-out noise is given for each detector channel. |
| - QC DET<i> FFM STDEV<p> | The standard deviation of the computed flatfield map (the mean is normalized to 1.0). |
| - QC DET<i> GOODPIX | The number of good pixels flagged by a 1 in the bad pixel map. |
| - QC DET<i> BADPIX | The number of bad pixels flagged by a 0 in the bad pixel map. |
| - QC DET<i> GOODPIXELRATIO | The ratio of good over total pixels. |
| - QC DET<i> NL LAW | The function used to describe the nonlinearity. |
| - QC DET<i> NL NCOEFF | The number of nonlinearity function coefficients. |
| - QC DET<i> NL A .. F | The values of the function coefficients (global nonlinearity). |

Where i is the detector number (1 = L/M-Band, 2 = N-Band), j is the detector channel number (1 .. 32 for the L/M-Band, 1 .. 64 for the N-Band), and p is the readout mode (HAWAII-2RG: 1 = slow readout, 2 = fast readout, Aquarius: 1 = low gain mode, 2 = high gain mode).
The detector number (used for DET<i>) will be always 1 for the L/M-Band detector and 2 for the N-Band detector.
In addition, the DET<i> CHIP1 ID keyword allows to distinguish between different detector chips. This is necessary, to deal with a detector change after, for example, an instrument repair. This will also result in a bad pixel and flatfield map which is only valid for a specific detector chip.

### 9.19.1 Input files

| DO category | Explanation: | Required |
|---|---|---|
| DARK | Raw cold frames | Yes |
| FLAT | Raw exposed frames | Yes |

The DO classification tags must contain the substrings DARK or FLAT in order to be recognized properly. This means that the tags DARK, DARK_LM_SLOW_SPEED and DARK_SIDE_OF_THE_MOON are handled as DARK and FLAT, FLAT_LM_SLOW_SPEED and FLAT_EARTH are handled as FLAT.

### 9.19.2 Output files

| DO category | Explanation |
|---|---|
| BADPIX | Static bad pixel map |
| FLATFIELD | Static flatfield map |
| NONLINEARITY | Static non-linearity map |

### 9.19.3 Options

The following options are provided by this recipe.

| | | |
|---|---|---|
| --gain | Default conversion gain in [e-/DU]. | [0.0] |
| --nditskip | number of skipped frames. | [0] |
| --cosmics | flag if cosmics should be detected. | [0] |
| --darklimit | Absolute limit used for darks: good = [-limit ... +limit]. | [100.0] |
| --flatlimit | Relative limit used for flats: good = [ref*(1 - limit) ... ref*(1 + limit)]. | [0.2] |
| --min_linear_range | Threshold for flatfield and non-linearity map calculation: LOW < threshold <= LINEAR. | [100.0] |
| --max_linear_range | Threshold for flatfield and non-linearity map calculation: LINEAR < threshold <= NON_LINEAR. | [1.5e+04] |
| --max_nonlinear_range | Threshold for flatfield and non-linearity map calculation: NON_LINEAR < threshold <= SATURATED. | [3e+04] |
| --max_rel_deviation | Maximum relative deviation for the non-linearity fit. | [0.01] |
| --max_abs_deviation | Maximum absolute deviation for the non-linearity fit [e-]. | [10.0] |
| --compensate | Defines which kind of compensation should be applied. (none = no compensation at all, all = all compensations possible, dd = detector specific defaults, pb = subtract pixel bias, gb = subtract global bias, cb = subtract detector channel bias, rb = subtract row bias, ct = subtract crosstalk). | [pb,ct] |
| --nlstart | first non-linearity mapping sampling point. | [1024] |
| --nlstep | distance between two non-linearity mapping sampling points. | [64] |
| --bptype | type fo the bad pixel map (0=standard, 1=empty, 2=merge) | [0] |
| --fftype | type fo the flatfield map (0=standard, 1=conts, 2=average) | [0] |
| --nltype | type of the nonlinearity in the result file (0=pixel,1=linear, 2=average). | [0] |
| --poi | pixel of interest: <x>,<y>. | [0,0] |
| --expert | expert flag. | [0] |
| --nt | Flag if a new method should be used for detector calibration. | [TRUE] |

## 9.20 mat_est_aphase

Estimates the absolute phase from reference images and differential phases

This plugin uses one or several reference images to estimate the absolute phase from the differential phases given in an OI-FITS file.

It supports the following image reconstruction work flow:

1. Calculate one or more reference images using the mat_cal_imarec plugin and the bispectrum image reconstruction method.
2. Estimate the absolute phase from the differential phase and these reference images using the mat_est_aphase plugin.
3. Reconstruct the final image or images using the mat_cal_imarec plugin and the complex visibility image reconstruction method.

The following pseudocode illustrates the overall scheme of this plugin:

```
BEGIN
   Read all reference images and map them into the internal
   mas to pixel space;
   Read the OI-FITS file with the differential phase;
   FOR EACH differential visibility baseline DO
      Calculate the difference between the differential phase
        and the reconstructed phase from the reference images;
      Fit a straight line to these phase differences;
      The absolute phase is equal to differential phase
        minus the phase difference from the fit;
   ENDFOR
   Store the result (differential phases replaced by the absolute phase)
     in an OI-FITS file;
END
```

### 9.20.1 Input files

| DO category | Explanation | Required |
|---|---|---|
| REFERENCE_IMAGE | Reference image(s) | Yes |
| TARGET_CAL_INT | Calibrated interferometric data (without absolute phases) | Yes |

In the SOF file, the calibrated interferometric data files are classified as TARGET_CAL_INT.

In the SOF file the reference images are classified as REFERENCE_IMAGE. From each of these files, the images in the primary header unit (if a cube is present all image planes are used) are read in as reference images. The order of the files in the SOF file is important, because it is mapped to the sequence of wavelength windows given with the –lambda_list option.

### 9.20.2 Output files

| DO category | Explanation |
|---|---|
| TARGET_CAL_INT | Calibrated interferometric data. The absolute phases can be found in the OI_VIS table in column VISPHI |

It is possible to write an ASCII file (option –vis_name) which will contain:

- for each wavelength the following line will be created:
  lambda <i> = <wavelength [m]> reference = <reference image or -1> window = <reference window or -1> scale m -> pixel = <m to pixel factor>
- for each baseline the following blocks will be created:
  - for each reference image the following line will be created:
    ref <datemjd> <ucoord [m]> <vcoord [m]> <lambda [m]> <reconstructed phase [rad]> <phase difference [rad]>
  - a line for the fit of the straight line to the phase differences:
    # fit ref <datemjd> <ucoord [m]> <vcoord [m]> offset <offset value [rad]> slope <slope value [rad] chi2 <chi2 value>
  - for each differential phase for the baseline the following line will be created:
    phase <datemjd> <ucoord [m]> <vcoord [m]> <lambda [m]> <measured differential phase [rad]> <phase error [rad]> <fitted phase difference [rad]>

### 9.20.3 Options

| Option | Explanation | [default] |
|---|---|---|
| –fov | Field of view for the internal image in [mas]. | [40.0] |
| –npix | Size of the internal image in pixels. Powers of 2 should be used (speeds up the FFT), but this is not enforced. | [256] |
| –kwd_lambda_min | Keyword name for the lower wavelength stored in the reference image file. | [] |
| –kwd_lambda_max | Keyword name for the upper wavelength stored in the reference image file. | [] |
| –kwd_lambda_center | Keyword name for the central wavelength stored in the reference image file. | [] |
| –kwd_lambda_width | Keyword name for the band width stored in the reference image file. | [] |
| –lambda_unit | Unit used for the wavelength parameter and keyword values [m, mm, um, nm]. | [um] |

| | | |
|---|---|---|
| –lambda_list | A list of lambda ranges (pairs of lower and upper wavelength). It is a sequence of comma separated floating point numbers or 'none'. | [none] |
| –ref_mode | The mode for reading/creating the reference image. 0 = read from file, 1 = point source. | [0] |
| –ref_param | Additional parameter for the reference image creation (mode=0 -> scale [mas/px]). | [0.0] |
| –phi_error | Differential phase error used instead of the measured error (phi_error=0.0 -> use measured error [rad]). | [0.0] |
| –vis_name | ASCII file for differential and absolute phases. | [] |

The option (–phi_error) enables the use of the measured differential phase errors (–phi_error=0.0, default) or an artificial (fixed) phase error (for example 1.0).

## 9.21 mat_est_tf

This plugin estimates the transfer function from one or several RAW_VIS2 FITS files corresponding to observations of calibrator stars. The transfer function is the raw squared visibility of the calibrator after correcting for its spatial extent. Thus, the transfer function gives the response for an unresolved source, including atmospheric and instrumental effects. The angular diameter (and its error) of the calibrators should be known. The diameter and its error are retrieved from the JSDC catalog and RAW_VIS2 files corresponding to different calibrators could be provided in the sof file. The recipes creates a RAW_TF2 FITS file containing the transfer function for each RAW_VIS2 file given as input.

The following QC1 parameters are calculated: - QC DET<i> TFSQR<m>: the average transfer function from the squared raw visibility. where i=detector number (1=L/M band, 2=N band) and m=baseline number (1..6)

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| RAW_VIS2 | VIS2 (Calibrator) | Yes |
| JSDC_CAT | JSDC Catalog | Yes |
| OUTPUT FILES | | |
| RAW_TF2 | transfer function | |

## 9.22 mat_cal_vis

This plugin calibrates the squared visibility. It interpolates the transfer function from several calibrators (CALIB_RAW_INT). Then it divides the raw squared visibility by the interpolated transfer function.

| INPUT FILES | | |
|---|---|---|
| DO category | Explanation | Required |
| TARGET_RAW_INT | Science OIFITS | Yes |
| CALIB_RAW_INT | Calibrator OIFITS | Yes |

| OUTPUT FILES | | |
|---|---|---|
| CAL_VIS2<br>INTERP_TF2 | OIFITS with OI_VIS2 binary table<br>interpolated transfer function | |
| OPTIONS | | |
| –tfKeep<br>–tfInterp | store interpolate function (=1)<br>transfer function interpolation method (0: average, 2:linear function) | [0]<br>[0] |

# 10 Additional Tools

In addition to the core `esorex` pipeline recipes, the MATISSE consortium has also developed several high-level scripts that can help in reducing large amounts of data or call the underlying recipes in a user-friendly way. These are described in appendix D. The tools themselves are not part of the pipeline distribution but can be obtained from the consortium git repository. For access to that, please contact Philippe Berio and Florentin Millour.

# 11 Algorithms

## 11.1 Input data

The basic input data are the raw frames of the MATISSE instrument, stored as FITS files. Let be $i^{\mathrm{raw}}(X, Y, t)$ this raw data intensity at time $t$, given at the $(X, Y)$ pixel detector index. In addition, calibration frames, like flat field map, bad pixels map, or distortion map, are used to correct the basic cosmetic of the detector and optics of the instrument. All relevant parameters are contained in the FITS header and are cumulatively added during the data processing steps. The data are affected by the following effects:

- Detector bias,
- Detector bad pixels,
- Detector + instrument flat field, plus eventually non-linear effects,
- Instrument distortion map,
- Sky background,
- Object spectrum,
- Instrument + atmosphere transfer function.

All these effects must be removed from the data before getting science-grade data. In the following we concisely describe the steps to correct them, and we propose to the reader to refer to [21] for more details.

In this section the data reduction procedures applied by the pipeline recipes currently in use (see Section 5.1) are described in some detail. Common algorithms, as cosmic rays removal or bad pixel cleaning, are described separately.

## 11.2 General Algorithms

### 11.2.1 Estimating BPM, FFM, nonlinearity map

The static calibration maps (bad pixel map, detector flatfield map and nonlinearity map are derived from a set of raw files taken with detector specific observing blocks. From each exposure, containing at least 20 frames, the average intensity and temporal variance is calculated for each pixel. The statistics from the flatfield exposures is compensated by subtracting the related dark statistics.

The nonlinearity map for the Aquarius detector is calculated by estimating a global nonlinearity response from the different measured intensities taken with different shutters open. This flux-dependend nonlinearity law is valid for a specific exposure time (DIT) only.

The nonlinearity map for the HAWAII-2RG detector is estimated for each pixel by fitting a specific function on the pixel intensity to exposure time relation. Then a global nonlinearity response is estimated using all calculated nonlinearity responses. This nonlinearity compensation is based on all well illuminated pixels and

goes near saturation level (the maximum intensity depends on the IR lamp current). It is possible to use the global nonlinearity for all pixels.

The flatfield map is based on a nonlinearity compensated, artificial flat which is separated into an ideal (reference) illumination and a residual. The individual pixel gain is equal to the residual divided by reference illumination. All pixels in the covered areas (borders of both detectors) will have a gain of 1.0.

All pixels where the measured intensity in the dark or the flatfield gain are outside a given interval (absolute range for dark and relative range for the flatfield) are marked as bad pixels.

### 11.2.2 Estimating Shift map, kappa-matrix, shift and zoom coefficients

**Shift map** The shift map is derived from a series of frames containing either a spatial grid (3 separate holes in the slit direction) or the plastic foil (for the wavelength direction). This distortion estimation is done at the same frequency as the flat field map or bad pixel map estimations. The distortion can be expressed as the relation between the pixel coordinates (X, Y) and the spectrograph coordinates $(x, \lambda)$ on the detector:

$$\begin{aligned} X &= a(x,\lambda)x + b(x,\lambda)\lambda \\ Y &= c(x,\lambda)x + d(x,\lambda)\lambda \end{aligned} \tag{1}$$

$(X, Y)$ are the pixel indices, and $(x, \lambda)$ are the position and spectral dispersion, respectively. This can be written as a matrix relation:

$$\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} a(x,\lambda) & b(x,\lambda) \\ c(x,\lambda) & d(x,\lambda) \end{pmatrix} \begin{pmatrix} x \\ \lambda \end{pmatrix} = (A)\,(x,\lambda) \begin{pmatrix} x \\ \lambda \end{pmatrix} \tag{2}$$

The matrix $(A)(x,\lambda) = \begin{pmatrix} a(x,\lambda) & b(x,\lambda) \\ c(x,\lambda) & d(x,\lambda) \end{pmatrix}$

is then inverted to get the distortion relation:

$$\begin{pmatrix} x \\ \lambda \end{pmatrix} = (A)^{-1}(X,Y) \begin{pmatrix} X \\ Y \end{pmatrix} \tag{3}$$

*Spatial Direction:* The spatial grid is used in this case. The $a$ parameter is estimated by fitting a polynomial to the position of the different features detected on the beams (3 features per beam):

$$a(x,\lambda) = a_0(\lambda) + a_1(\lambda) \times x + a_2(\lambda) \times x^2 + ... \tag{4}$$

*Spectral direction:* In that case, we are only interested in determining the $\lambda$-dependent distortion. Therefore, a "normal" $\lambda$-calibration frame (i.e. with the set of absorption plastic foils, or using sky absorption lines) should be available for that. In that case, one can determine the $d$ coefficient:

$$d(x,k) = d_0(x) + a_1(x) \times k + d_2(x) \times k^2 + ... \tag{5}$$

We note here that this first step will give a coarse wavelength-calibration, which will be refined in a further step using telluric lines.

**Shift and zoom, $\kappa$-matrix** The Shift and Zoom coefficients, together with the $\kappa$-matrix are computed using 4 illuminated frames, one for each of the telescope beams, the other beams being closed by shutters. The source is preferably artificial, part of MATISSE instrument, or could be a bright unresolved astronomical target. The photometric contribution $P'_i(x, \lambda)$ of the telescope $i$ to the interferometric beam can be expressed as following:

$$P'_i(x, \lambda) = \kappa_{ij}[x', \lambda, s_j(\lambda), z_j(\lambda)] \times P_{ij}(x', \lambda) \tag{6}$$

where $P_{ij}(x', \lambda)$ is a vector containing the contribution of the telescope $i$ to the photometric beams $j$, and $s_j(\lambda)$, $\kappa_{ij}$ is the linear transformation matrix of the intensities in the photometric channels into the interferometric channel (the so-called "$\kappa$-matrix"), and $z_j(\lambda)$ are the shift offset and zoom coefficient to match the photometric beam $j$ into the interferometric beam. We determine therefore the $\kappa$-matrix and the shift-and-zoom coefficients by fitting the photometric beam shape and intensity to the interferometric beam for each selected wavelength.

### 11.2.3 Applying cosmetics

The first steps of the data reduction are:

1. Compensate for some offset effects (pixel bias, detector channel bias and cross talk),
2. Compensate for the nonlinear intensity response,
3. Compensate for the spatially varying gain in each frame by dividing them by the flat field map,
4. Interpolating bad detector pixels in each frame,
5. Applying the distortion matrix to transform $(X, Y)$ to $(x, \lambda)$, the coordinates of the data,
6. Adjusting the photometric contribution in the interferometric beam by applying the $\kappa$-matrix, shift, and zoom coefficients
7. Removing the thermal background with chopping for the photometric beams estimate.

The results are: a clean fringe pattern $i(x, \lambda, t)$, and clean photometric estimates $P_i(x, \lambda, t)$ (stored in files `TARGET_CAL_XXXX.fits` and `PHOT_BEAMS_XXXX.fits`, respectively).

## 11.3 Demodulation

Correlated fluxes are simply computed by taking the Fourier-transform of the clean interferogram in the $x$ direction.

$$I^{'}(u, \lambda) = \mathrm{FT}_x[i(x, \lambda)] \tag{7}$$

Then, a de-modulation is applied to this Fourier-Transform, baseline-by-baseline, by multiplying it with a phasor containing the counter-modulation:

$$I''_j(\lambda, T) = \sum_{t=1}^{N_t} I^{'}(u_j, \lambda, t) \times e^{-i\Delta_j(t)/\lambda} \tag{8}$$

where $u_j = j \times D/\lambda$ corresponds to the spatial frequency of baseline $j$, and $\Delta_j(t)$ is the introduced OPD on one modulation cycle, which contains $N_t$ steps and which is done on a timescale smaller than the coherence time $\tau$ of the atmosphere. The results is time-tagged on coherence-length units T.

## 11.4 OPD calculation

The OPD estimation implemented in the MATISSE pipeline is based on the proceeding of [23]: the OPD is calculated together with the differential phase and an achromatic phase in an optimal way by minimizing a $\chi^2$.

## 11.5 Incoherent Processing

**Squared visibilities** Squared visibilities are computed as follows:

$$V_j^2(\lambda) = \frac{\sum_u \left\langle |I^{'}(u_j,\lambda,t)|^2 - B \right\rangle_t}{2 \cdot \sum_x \left\langle \overline{P_a}^{'}(x) \cdot \overline{P_b}^{'}(x) \right\rangle_t} \tag{9}$$

where $u_j$ is the frequency of the considered baseline, and B a bias estimated on $I^{'}$ outside the range of frequencies where the fringe peaks are present, and $\overline{P_a}^{'}(x)$ and $\overline{P_b}^{'}(x)$ are the estimates of the photometric fluxes transformed into the interferometric channel (see eq. 6).

In the case where there are no photometries recorded (like in HIGH-SENS mode), the squared visibilities are simply not computed. Instead, the user should use the correlated fluxes.

**Closure phases** The bispectrum is calculated by:

$$O_{ij}^{(3)}(\lambda) = \left\langle I^{'}(u_i,\lambda,t) \times I^{'}(u_j,\lambda,t) \times I^{'*}(u_i+u_j,\lambda,t) \right\rangle_t \tag{10}$$

where $u_i$ and $u_j$ are the frequencies of the two first baselines involved in the closure relation, the third one being automatically set by $u_i + u_j$.

This average bispectrum contains an additive noise bias term and a multiplicative one, because of the all-in-one combination of the interferograms. These biases need to be subtracted in order to get closure phases without systematic errors, following the recipes given in [24] or [17].

## 11.6 Coherent Processing

In the coherent processing, you can select 2 algorithms. The first one is similar to what has been used for AMBER. It allows to estimate differential phases and visibilities. The second one is similar to what has been used for MIDI. It allows to estimate the phase and the coherent flux.

**Differential phases and visibilities (–coherentAlgo=1)** The uncontaminated interferograms, including the variations of the refractive index $n(k)$ of water vapor, can be expressed as:

$$I^{''}(u,\lambda,t) \approx \sum_{j=1}^{N_{\text{bases}}} \left[ F_j^{''}(u - d_j/\lambda, \lambda) \times V_j(\lambda) \times exp(i\Phi_j(\lambda,t)) + \delta B_{rej}(u - d_j/\lambda, \lambda) \right] \tag{11}$$

with the following terms:

- $F_j^{''}(u - d_j/\lambda, \lambda)$ the (complex) shape of the fringe peak
- $V_j(\lambda)$ the instrument + object visibility
- $\delta B_{rej}(u - d_j/\lambda, \lambda) = B_{rej}(u - d_j/\lambda, \lambda) \langle exp(-i\Delta_j/\lambda)\rangle$ the residual sky background after OPD modulation (see eq. 8).
- $\Phi_j(\lambda, t) = 2\pi\Delta_j^{\mathrm{atm}}(t)/\lambda + 2\pi\Delta_j^{\mathrm{atm}}(t)/\lambda \cdot [n(\lambda, t) - 1] + \Phi_j(\lambda)$ the phase containing the OPD, the chromatic OPD and the object phase:
  - $\Delta_j^{\mathrm{atm}}(t)$: atmospheric OPD,
  - $[n(\lambda, t) - 1] \approx (a(\lambda) + b(t))$, chromatic term with $a(\lambda)$ wavelength-dependent but roughly static (see for instance [2, 19]), and $b(t)$ wavelength independent, but with strong variations in time [18].
  - $\Phi_j(\lambda) = \Phi_j^{(0)} + \Phi_j^{(1)}/\lambda + \Phi_j^{\mathrm{diff}}(\lambda)$: Taylor expansion of the object phase and differential phase $\Phi_j^{\mathrm{diff}}(\lambda)$.

**Coherent flux (–coherentAlgo=2)** The demodulated correlated fluxes (see 11.3) for each frame are corrected from the OPDs and achromatic phases (see 11.4) and then averaged over one exposure. From this averaged correlated flux, we derived the phase and the coherent flux for each spectral channel.
Depending on the parameter –corrFlux, you can estimate the coherent flux (–corrFlux=TRUE) or the visibility (–corrFlux=FALSE).

## 11.7 Fringe Jumps Flagging with GRA4MAT

The flagging procedure consists in finding the jumps in the observations from the fringe tracker performed simultaneously with the Matisse instrument. The group Delay is computed from the GRA4MAT data for all 6 baselines. Fringe jumps are detected in these group delay. When a jump is detected , i.e. when the group delay jumps by more than $0.6\lambda$, frames recorded with MATISSE during the jump are flagged (TARTYP=J), and therefore dismissed from the remaining computations of the pipeline.
The flagging depends on the MATISSE spectral bands:

- For the N band data, 10 frames are actually flagged. The main frame which corresponds to the timestamp of the detected jump and the other 9 frames of the modulation cycle to which the main frame belongs.
- For the L band data, the frame with the timestamp corresponding to that of the detected jump is flagged.

Let's note that even if the group delay is computed per baseline, all baselines are affected by the flag. This jump flagging is automatically applied when in the *sof* file GRA4MAT data appear. Removing the latter switches off the automation of the flagging.

## 11.8 N band coherent processing with GRA4MAT

Fringe trackers are used to stabilize the fringes by controlling the Optical Path Differences (OPDs). Once the fringes are stabilized, it is possible to record data with long integration times (from a few seconds up to a few minutes) on the scientific instrument. In the L band, MATISSE can record fringes with an integration time of 10s when the fringes are stabilized with GRA4MAT. It is therefore possible to observe objects fainter than the achievable limiting magnitude with MATISSE when the latter is used in standalone mode. However in N band, this procedure cannot be applied :

- Between $8\mu m$ and $13\mu m$, the radiating thermal background is very bright. It is much brighter than most of the observed astrophysical sources. So the maximal integration time is, as required, set to avoid any saturation of the detector due to the thermal background radiation. For example with MATISSE at low spectral resolution (R=30), the integration time of the AQUARIUS detector is therefore set to 25ms.
- The water vapor content of air affects strongly the OPDs in the N band. So even if the fringes are stabilized in the K band through the GRA4MAT subsystem, the OPDs in N band could fluctuate at a level of a few microns. Any integration time longer than the coherence time - typically 300ms in N band - impacts negatively and even destructs the fringe contrast.

Instead of applying long integration time, it is then possible to co-phase short exposures of the fringes during the data reduction process. This method is called a posteriori co-phasing. In N band, it is optimal in terms of Signal to Noise Ratio since the data are limited by the background photon noise and not by the read out noise of the detector. This method consists in first predicting the OPDs in N band from the OPDs residuals of the fringe tracker (working in another and shorter spectral band, H or K for example), second correcting the fringes from the predicted OPDs and finally in integrating the frames over several seconds or minutes.
To estimate the N band PD and GD from K band data, we implemented the method of Koresko et al (2006). They estimated the differential column density of water vapor from the PD and GD measured in K band. Then, they deduced N band PD and GD through linear relationships. It is then possible to predict the phase at any wavelength in the N spectral band.
The Pipeline automatically applies the a posteriori co-phasing technique when GRA4MAT data are in the *sof* file. Removing the GRA4MAT data from the *sof* file disables it.

## 11.9 BCD calibration

The pipeline produces OIFITS files for each exposure of an OB whatever the status of the BCD. Combining carefully the different BCD exposures allows to remove instrumental signatures in the differential and closure phases. In the pipeline, the BCD calibration is done in the recipes *mat_cal_oifits* when the parameter *cumulBlock* is set to TRUE. It is applied just after the calibration of the scientific measurements by the calibrator measurements. It produces a combined OIFITS file.
The algorithm is the following:

- Visibility: For each baseline, the visibilities are averaged whatever the BCD configuration is. In L/M band, only chopped frames are used.
- Closure Phase: The 4 closure phases could be corrected from the instrumental signatures thanks to the BCD. The pipeline computes linear combinations of the individual closure phases (for example we apply

the following equation $\Delta\phi_{comb} = (\Delta\phi_{OUT-OUT} + \Delta\phi_{OUT-IN} - \Delta\phi_{IN-OUT} - \Delta\phi_{IN-IN})/4$ for the first triplet corresponding to IP1-IP3-IP5).

- Differential Phase: Only 2 differential phases could be corrected from the instrumental signatures thanks to the BCD ($\phi_{comb} = (\phi_{OUT-OUT} + \phi_{OUT-IN} - \phi_{IN-OUT} - \phi_{IN-IN})/4$). They correspond to the pairs of beams: IP1-IP3 and IP5-IP7. The other 4 differential phases are just averaged whatever the BCD configuration is.
- Flux: For each telescope, the flux is averaged whatever the BCD configuration is. In L/M band, only chopped frames are used.

# 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.

# B Abbreviations and acronyms

| | |
|---|---|
| ANSI | American National Standards Institute |
| ASCII | American Standard Code for Information Interchange |
| CalibDB | Calibration Database |
| CPL | Common Pipeline Library |
| DFO | Data Flow Operations department |
| DFS | Data Flow System department |
| DMO | Data Management and Operations division |
| DPR | Data PRoduct |
| DRS | Data Reduction System |
| ESO | European Southern Observatory |
| ESOREX | ESO-Recipe Execution tool |
| FITS | Flexible Image Transport System |
| FOV | Field Of View |
| FPN | Fixed Pattern Noise |
| GUI | Graphical User Interface |
| IP | InPut Channel |
| OB | Observation Block |
| PSO | Paranal Science Operations |
| QC | Quality Control |
| RON | Read Out Noise |
| SOF | Set Of Frames |
| UT | Unit Telescope |
| AT | Auxiliary Telescope |
| VLT | Very Large Telescope |

# C Troubleshooting Guide

This section will be filled in as troubles arise. In case you encounter problems with the MATISSE data reduction software, please contact Philippe Berio and Florentin Millour.

# D Additional Tools

## D.1 Automatic Pipeline Python Script

The Python scripts described here are an esorex “wrapper" aimed at simplifying the process of data reduction. They can create the sof for each reduction, execute the esorex recipes to reduce the data, and create a directory for each reduced observing block. Moreover, several GUI interfaces are provided to select and/or view the data products.

The requirements to use these scripts are:

- Python 3.6
- These Python packages: astropy, matplotlib, numpy, ObjectListView, and wxpython
- The ESO pipeline for MATISSE

Specifically, the script automaticPipeline.py prepares the data by making sofs and reduces them using **mat_raw_estimates**. It introduces several new parameters in addition to those in mat_raw_estimates, e.g.,

- tplSTART – allows you to specify a single observing block to reduce within a larger data directory.
- nbCore – allows you to specify the number of parallel cores utilized in data reduction.
- skipN/skipL – allows you to skip either the N-band or LM-band data, reducing only one band.
- paramL/paramN – allows you to specify the mat_raw_estimates parameters for each band.

An example of using the automaticPipeline.py script is below:

```
python automaticPipeline.py -dirRaw=/data/2018-05-19
-nbCore=2 -paramN=/useOpdMod=TRUE/corrFlux=TRUE -paramL=/cumulBlock=TRUE
```

## D.2 Image Reconstruction Script

The C-Shell scripts described are for easy handling of the MATISSE image reconstruction software.

1. Both C-shell scripts "mat_cal_imarec_all.csh" and "mat_cal_imarec_all.2.csh" are easy-to-use callers of the MATISSE image reconstruction recipe mat_cal_imarec.
2. All necessary scripts and files are stored in one folder, named mat_cal_imarec.scripts00/.
   The environment variable $SCRIPTS must point to this folder, i.e.
   - % setenv SCRIPTS /MYPATH/mat_cal_imarec.scripts00, or
   - % set SCRIPTS /MYPATH/mat_cal_imarec.scripts00
     % export SCRIPTS

3. The following software is required:
   - The ESO pipeline for MATISSE must be installed.
   - The HEASARC ftools package, which can be downloaded from https://heasarc.gsfc.nasa.gov/ftools/ftools_menu.html (for installation instructions see https://heasarc.gsfc.nasa.gov/lheasoft/install.html) As noted there, you must initialize the package by sourcing $HEADAS/headas-init.sh, i.e.
     % source $HEADAS/headas-init.csh or % . $HEADAS/headas-init.sh
   - The scripts need the following LINUX programs, too:
     gnuplot, awk, latex, convert, sort, psmerge.
     You can check if all this software is available with: $SCRIPTS/swtest.csh
4. Overall reconstruction procedure. During the run, this software automatically uses different image reconstruction parameters to find the "best" reconstruction. These parameters are:
   - a number of regularisation functions $regFuncs
   - different regularisation hyperparameters $mu, which control the strength of the regularisation
   - a number of radii of a circular binary object mask, which restricts the image space, where the reconstruction can have non-zero values

   Their overall reconstruction scheme is illustrated by the the following pseudocode: ($name denotes the name of the corresponding parameter in the parameter file)

```
FOR EACH "regularization function" regFuncs DO
  FOR EACH "start value of the hyperparameter" muStarts DO
   # Call IRBis with the actual regularization function and hyperparameter
   # Internally IRBis then performs two loops:
     FOR EACH oradius = oradiusStart+i*stepSize  DO
       FOR EACH mu = muStart*muFactor^j   DO
          Reconstruct one Image with the current parameters oradius,
          mu and regularization function; calculate its qrec ("quality"
          measure of the reconstruction, see below); manage an
          internal list of all reconstructions sorted by qrec
       ENDFOR (mu)
     ENDFOR (oradius)
     RETURN reconstruction with the best (=smallest) qrec
   # end of IRBis call
   Maintain an external list of all best reconstructions returned
   from IRBis, and continue to loop over regFuncs and muStarts
  ENDFOR (muStarts)
ENDFOR (regFuncs)
Look for the reconstruction with the smallest qrec in the external
list and return this reconstruction as the best one.
```

   As a criterion to select the best reconstruction, the minimum of a value named qrec is used. It is calculated from the measured $\chi^2$ and the residual ratio values (for more details see Hofmann, Weigelt, & Schertl, D.

2014, A&A, 565, A48). There are two versions how qrec is calculated:

- $qrecmode=1: it is calculated using the $\chi^2$ and the residual ratio values of both the visibilities and the phases (Closure or Fourier phase)
- $qrecmode=2: it is calculated using the $\chi^2$ and the residual ratio values of only phases (Closure or Fourier phase)

Experience shows, usually the qrec from $qrecmode=2 yields better results than qrec from $qrecmode=1!

5. The Difference between "mat_cal_imarec_all.csh" and "mat_cal_imarec_all.2.csh" is the different way, the $muStarts are defined and used.

   - "mat_cal_imarec_all.csh" loops over all $regFuncs using a fixed list of different start values $muStart. It will call IRBis with each of this $regFuncs/$muStart pairs.
   - "mat_cal_imarec_all.2.csh" loops over all $regFuncs, too, but using start values $muStart calculated as:
     ..... $muStart(next) = $muStart(actual) * $muFactor0, starting with $muStart0,
     and calls IRBis with this $regFuncs/$muStart pairs. This loop stops, if the returned qrec starts to increase, i.e. the actual qrec is larger than the one of the run before.

6. An image reconstruction run is started with:
   % $SCRIPTS/mat_cal_imarec_all.csh parfile (template file in $SCRIPTS/mat_cal_imarec_all.par)
   or
   % $SCRIPTS/mat_cal_imarec_all.2.csh parfile (template file in $SCRIPTS/mat_cal_imarec_all.2.par).

7. Tutorial — Usage of the scripts "mat_cal_imarec_all.csh" and "mat_cal_imarec_all.2.csh":
   **Step 0:** Define the enviroment variable $SCRIPTS with the absolute path of the script folder:
   % setenv SCRIPTS /MYPATH/mat_cal_imarec.scripts00
   or
   % set SCRIPTS /MYPATH/mat_cal_imarec.scripts00
   % export SCRIPTS

   **Step 1:** Create a work folder for the image reconstruction session (usually named as the target to be processed) and go to this folder.

   **Step 2:** Copy the parameter file "$SCRIPTS/mat_cal_imarec_all.par" ("$SCRIPTS/mat_cal_imarec_all.2.par") of the C-shell script "mat_cal_imarec_all.csh" ("mat_cal_imarec_all.2.csh") into your work folder
   % cp $SCRIPTS/mat_cal_imarec_all.par .

   **Step 3:** Estimation of the size of the target and FOV to be used for image reconstruction:
   a) The size of the target is estimated by fitting a circular Gaussian, uniform disk, fully darkened disk or a modified Lorentz intensity distribution into the observed squared visibilities of the target and their corresponding reduced $\chi^2$.
   b) Calculation of the proposed size for the FOV of the reconstruction (in pixel and milliarcseconds).

- Edit the parameter file "mat_cal_imarec_all.par":

| | | |
|---|---|---|
| set data | = (file1 file2 ...) | list of oifits data files for imaging |
| set lambdaList | = (0.1 20) | wavelength interval (micron); use (0.1 20) to cover all data |
| set guess | = 1 | 1 to switch on first action: estimating data quality & image reconstruction parameters |

- Run the script:
  % $SCRIPTS/mat_cal_imarec_all.csh mat_cal_imarec_all.par

- Have a look to the uv coverage, the wavelengths, and to the fits of the geometrical models to the measured visibilities:
  % gv Parameter.Estimation/uv.ps
  % gv Parameter.Estimation/wavelengths.ps
  % gv Parameter.Estimation/gaussudfdda.ps

- Select one or more wavelength intervals for image reconstruction and change the parfile:
  set lambdaList = (3.1 3.2 3.5 3.7)

- Run the script again:
  % $SCRIPTS/mat_cal_imarec_all.csh mat_cal_imarec_all.par

- Check the selected wavelengths and the resulting uv coverage of the data:
  The text file "Parameter.Estimation/data.parameter" contains information about the size of the target and the recommended FOV. The FOV for image reconstruction should be roughly $\sim$2 to 4$\times$ the target size.

Be careful with the proposed sizes of the target. If the target is composed of many distinct small objects far apart (i.e. binary), then the estimation of the size is too small usually. In this case, you should set the FOV to $\sim$2-4$\times$ the expected binary separation.

**Step 4:** First reconstruction run
a) Carefully read the hints for FOV and number of pixels in the Parameter.Estimation/data.parameter
b) Start with the smallest pixel number possible for the target, i.e. 64x64 pixels ($npix)
c) The Image mask start radius should be $\sim$2$\times$ larger than the estimated target size ($oradiusStart)

- Edit the parfiles "mat_cal_imarec_all.par" or "mat_cal_imarec_all.2.par", respectively:

| | | |
|---|---|---|
| set guess | = 0 | Set to 0 to switch to the image reconstruction run |
| set engine | = 1 | Choose optimization engine; 1: ASA-CG, 2: L-BFGS-B. |
| set algoMode | = 1 | Specify interferometric data to be used for imaging.<br>1 = bispectrum, 2 = complex visibilities.<br>3 = use bispectrum and complex visibilities. |
| set fov | = 50 | FOV (mas) of the reconstruction with respect to the recommendations in "Parameter.Estimation/data.parameter": choose a FOV out of the list of different $2^n$ array sizes & corresponding FOVs which best fit to the target size (FOV $\sim$2-4$\times$ target size). |
| set npix | = 64 | Number of pixels corresponding to the FOV chosen. (because of speed, powers of 2 should be used only) |
| set oradiusStart | = 20 | First radius (mas) of image mask: should be larger than the target (could even be larger than the FOV) |
| set stepSize | = 2 | Step size in mas (usually $> 0$ to increase mask radius) to calculate the next image mask radius:<br>$oradius_next = $oradius + $stepSize |
| set oradiusNumber | = 6 | Number of mask radii to be tested (6 is mostly ok) |
| ——————— | ——— | for "mat_cal_imarec_all.csh" only ————- |
| set muStarts | = (1.0 ..) | Define the list of start regularization parameters muStart. |
| ——————— | ——— | for "mat_cal_imarec_all.2.csh" only ——— |
| set muStart0 | = 10 | First value of muStart to be used. The later muStart values are calculated as:<br>muStart(next) = muStart(actual) * muFactor0 |
| set muFactor0 | = 0.5 | Factor to calculate the next muStart value |
| ——————— | ——— | ——————————————————— |
| set muFactor | = 0.5 | Factor to calculate next regularization parameter $mu in IRBis: $mu_next = $mu * $muFactor |
| set muNumber | = 12 | Number of regularization parameter $mu to be tested |
| set qrecmode | = 2 | Selection of reconstruction quality criterion qrec:<br>qrecmode=1 : qrec is calculated using the $\chi^2$ & residual ratios of both, the phases (Closure Fourier phases) and the squared visibilities.<br>qrecmode=2 : qrec is calculated using the $\chi^2$ & residual ratios of phases (Closure or Fourier) only. |
| set regFuncs | = (-3 -4) | selecting regularization functions |
| set weightPower | = 0.0 | Power for the uv density weight (0.0 for bispectrum and 0.5 for complex visibilities are usually OK) |
| set startmode | = 2 | select start image: 0=read from file, 1=point source, 2=Gaussian disk, 3=uniform disk, 4=fully darkened disk, 5=modified Lorentz function |
| set startparam | = 3.7 | set size (mas) of start image chosen. Use the value of the fit listed in "Parameter.Estimation/data.parameter":<br>startmode= 2 -> FWHM [mas], 3 -> diameter [mas], 4 -> diameter [mas], 5 -> FWHM [mas] |

- Run the script:
  % $SCRIPTS/mat_cal_imarec_all.csh mat_cal_imarec_all.par
  or
  % $SCRIPTS/mat_cal_imarec_all.2.csh mat_cal_imarec_all.2.par

8. Short summary of the results of the scripts "mat_cal_imarec_all.csh" and "mat_cal_imarec_all.2.csh":

   (a) The results are stored in result folders named:
   - $algoMode=1 $\longrightarrow$ BIS.*.Script*.E.1/ (bispectrum data),
   - $algoMode=2 $\longrightarrow$ FT.*.Script*.E.1/ (complex visibilities),
   - $algoMode=3 $\longrightarrow$ FTBIS.*.Script*.E.1/ (bispectrum & complex visibilities).

   If later the script is started again, the extension of the folder will automatically increase, i.e. *.E.2 *.E.3 .. (if the folder name could be identical to another one).
   If the folder has the extension *.Script1.E.1/, then it was produced by the script "mat_cal_imarec_all.csh", and if the folder has the extension *.Script2.E.1/, then it was produced by script "mat_cal_imarec_all.2.csh".

   (b) In the result folder, each subfolder E.1, E.2, .. contains the best reconstruction of one run of IRBis:
   Contents of the subfolders E.1, E.2 ...:
   - rec_*.fits : contains the "best" reconstruction of each run. This is the direct outcome of the recipe mat_cal_imarec (IRBis). It not only contains the unconvolved and convolved reconstruction, but also other data. See the manual of mat_cal_imarec for more. Man page call: % esorex –man-page mat_cal_imarec
   - bestrec.fits, bestrecconv.fits : the "best" reconstruction without this other data, unconvolved and convolved, respectively.
   - model.fits, modelconv.fits : the model image (unconvolved/convolved) if specified.

   (c) In the result folder, the textfile E.liste contains the $\chi^2$/ResidualRatio values and the image reconstruction parameters for all IRBis runs, i.e. in E.1, E.2, ...:
   Each run is listed in one line; the runs are sorted with decreasing image quality, i.e. with increasing value of the image quality parameter qrec:
   - phase-visibility-qrec (qrecmode=1), derived from $\chi^2$ and residual ratios of the squared visibilities and CP's or Fourier phases),
   - phase-qrec (qrecmode=2), derived from $\chi^2$ and residual ratios of the phases (CPs or Fourier phases only).

   (d) In the result folder the best reconstruction out of all reconstructions stored in E.1, E.2, etc.. is named:
   - bestrec.qrec1.fits (best according to the measure phase+visibility-qrec, i.e. qrecmode=1),
   - bestrec.qrec2.fits (best according to the measure phase-qrec, i.e. qrecmode=2).

   (both are fits files and the direct outcome of the recipe mat_cal_imarec (i.e. the fits file rec_*.fits in the subdirectories E.*/), and they contain, for example, the unconvolved and convolved reconstruction)

   (e) In the result folder are postscript files which contain all reconstructions (from E.1, E.2, ...) sorted according to the increasing qrec measure as listed in E.liste:
   - in folder qrecmode=1/ reconstructions sorted according to the increasing value of phase+visibility-qrec,

- in folder qrecmode=2/ reconstructions sorted according to the increasing value of phase-qrec.

The postscript files in these two folders are: *.lin.ps (linear display), *.sqrt.ps (sqrt display), and *.log.ps (log display).

(f) Furthermore, in the result folder there are postscript files for the case of image reconstruction simulations (if the known theoretical target is set in the parfiles), which display the correlation between the qrec parameters and the direct distance between the reconstructed and theoretical object (for the definition of the distance see dist1.0 in the Appendix below):

- *.distqrec.1.plot.ps (for the image quality parameter qrec including phases and visibilties, i.e. qrecmode=1),
- *.distqrec.2.plot.ps (for the image quality parameter qrec including phases only, i.e. qrecmode=2).

9. List of parameters

| | |
|---|---|
| - qrec | image quality parameter derived from the $\chi^2$ and residual ratio values of $V^2$ and $CP$ or absolute phase |
| - cost | value of the cost function |
| - chi2bis, rresbis | $\chi^2$ and residual ratio of the bispectrum |
| - chi2vis2, rresvis2 | $\chi^2$ and residual ratio of squared visibility |
| - chi2cp, rrescp | $\chi^2$ and residual ratio of closure phase |
| - dist1.0 | distance between theoretical $\mathrm{m(x)}$ and reconstructed object $\mathrm{o_k(x)}$; $\mathrm{dist} = \sqrt{\int[\mathrm{m(x)} - \mathrm{o_k(x)}]^2\,\mathrm{dx}/\int \mathrm{m(x)}^2\,\mathrm{dx}}$ |
| - FOV | FOV (mas) of the reconstruction |
| - Reg-Fct. | number of the regularization function used |
| - oradius step number | list of radii of the binary image masks; oradius(next) = oradius(actual) + step*i with i = 0, number |
| - mu factor number | list of regularization parameters mu; mu(next) = mu(actual) * factor$^j$, with j = 0, number |
| - calcVis2f0 | =1: an artificial squared visibility and error for spatial frequency 0 should be calculated; =0: is not calculated |
| - weightPower | power for the uv density weight |
| - npix | size of the reconstructed image in pixels |
| - startmode | start image: 0 = read from file, 1 = point source, 2 = Gaussian disc, 3 = uniform disc, 4 = fully darkened disc, 5 = modified Lorentz function |
| - startparam | startmode=0 $\rightarrow$ scale [mas/px], mode=2 $\rightarrow$ FWHM [mas], mode=3 $\rightarrow$ diameter [mas], mode=4 $\rightarrow$ diameter [mas], mode=5 $\rightarrow$ FWHM [mas] |
| - directory | reconstruction folder E.1, E.2, ... |
| - cpqrec | image quality parameter derived from the $\chi^2$ and residual ratio values of the CP or absolute phase only |

# E Additional QC paramters

These QC parameters have been implemented to easier assess the quality of the pipeline products.

## E.1 Recipe: All recipes

Written into the primary HDU of every output files which contain QC parameters.

| Name of new header key | extension | Computation |
|---|---|---|
| QC.INS.DISP | Primary HDU | If DET.NAME = MATISSE-N then QC.INS.DISP = INS.DIN.ID<br>If DET.NAME = MATISSE-LM then QC.INS.DISP = INS.DIL.ID<br>Dispersive element |
| QC.INS.FILT | Primary HDU | If DET.NAME = MATISSE-N then QC.INS.FILT = INS.FIN.ID<br>If DET.NAME = MATISSE-LM then QC.INS.FILT = INS.FIL.ID<br>Filter |
| QC.INS.POLARIZER | Primary HDU | If DET.NAME = MATISSE-N then QC.INS.POLARIZER = INS.PON.ID<br>If DET.NAME = MATISSE-LM then QC.INS.POLARIZER = INS.POL.ID<br>Polarizer |
| QC.INS.PHOT.SLIDER | Primary HDU | If DET.NAME = MATISSE-N then QC.INS.PHOT.SLIDER = INS.PIN.ID<br>If DET.NAME = MATISSE-LM then QC.INS.PHOT.SLIDER = INS.PIL.ID<br>Photometric slider |
| QC.INS.SPAT.FILTER | Primary HDU | If DET.NAME = MATISSE-N then QC.INS.SPAT.FILTER = INS.SFN.ID<br>If DET.NAME = MATISSE-LM then QC.INS.SPAT.FILTER = INS.SFL.ID<br>Spatial filter |
| QC.INS.FILT.ENC | Primary HDU | If DET.NAME = MATISSE-N then QC.INS.FILT.ENC = INS.FIN.ENC<br>If DET.NAME = MATISSE-LM then QC.INS.FILT.ENC = INS.FIL.ENC<br>Filter encoder position (only used to check the position) |
| QC.INS.DISP.ENC | Primary HDU | If DET.NAME = MATISSE-N then QC.INS.DISP.ENC = INS.DIN.ENC<br>If DET.NAME = MATISSE-LM then QC.INS.DISP.ENC = INS.DIL.ENC<br>Dispersive element encoder position (only used to check the position) |

| | | |
|---|---|---|
| QC.DET1.NAME | Primary HDU | If DET.NAME=MATISSE-LM, QC.DET.NAME = DET.NAME and QC.DET2.NAME=”” <br> If DET.NAME=MATISSE-N, QC.DET1.NAME = “” and QC.DET2.NAME=DET.NAME |
| QC.DET2.NAME | Primary HDU | If DET.NAME=MATISSE-LM, QC.DET1.NAME = DET.NAME and QC.DET2.NAME=”” <br> If DET.NAME=MATISSE-N, QC.DET1.NAME = “” and QC.DET2.NAME=DET.NAME |

## E.2 Recipe: mat_est_kappa

Output product(s) with added keywords: KAPPA_MATRIX

Keywords are either written into the primary HDU or in the extension to which they logically belong to.

| Name of new header key | extension | Computation |
|---|---|---|
| QC.DELTA.SHIFT | Primary HDU | See sec. E.2.1 <br> Tilt of the spectrum (in px) |

### E.2.1 Computation of QC.DELTA.SHIFT

This QC parameter is only calculated when det.name = MATISSE-LM

If INS.DIL.ID=LOW a = 100
If INS.DIL.ID is not LOW a = DET.WIN.MTRH2 -10
Top: from a-15 to a
Bottom: from 5 to 20
From the first KAPPA_SRC raw file, read column DATA9 of the IMAGING_DATA extension -> datafile
From the first KAPPA_HOTDARK raw file, read column DATA9 of the IMAGING_DATA extension –> darkfile
Clean the data : cleandata= median(datafile,axis=0)-np.median(darkfile,axis=0)
Get the cuts : Top and bottom
top= median(cleandata[a-15:a,:],axis=0)
median calculated between pixel a and a-15 (a = det.win.mtrh2 -10 depends of the resolution)
botttom= median(cleandata[5:5+15,:],axis=0)
median calculated between pixel 5 and 20
Fit a guassian for top and bottom and calculate the shift between the top of both gaussians

## E.3 Recipe: mat_raw_estimates

Output product(s) with added keywords: CALIB_RAW_INT, TARGET_RAW_INT, OI_OPDWVPO.

In some cases, some keywords are not in the header (for ex. QC.DETi.TFSQRj.MEAN), the value should be set to “” in the database.

| Name of new header key | extension | Computation |
|---|---|---|
| QC.TOT.FLUX | Primary HDU | From CALIB_RAW_INT or TARGET_RAW_INT: Sum(QC.DETi.FLUX.IPj) with i=1 when det.name = MATISSE-LM and i=2 when det.name = MATISSE-N and j=[1,3,5,7], with j referring to the input pupil<br>Total flux: sum of the fluxes detected by the pipeline for each beam |
| QC.MAX.OPD | Primary HDU | Maximum(ABS(column(OPD))<br>Maximum OPD from every baselines |
| QC.MAX.OPD1 | Primary HDU | Maximum(ABS(column(OPD,1))<br>Maximum OPD for baseline 1 |
| QC.MAX.OPD2 | Primary HDU | Maximum(ABS(column(OPD,2))<br>Maximum OPD for baseline 2 |
| QC.MAX.OPD3 | Primary HDU | Maximum(ABS(column(OPD,3))<br>Maximum OPD for baseline 3 |
| QC.MAX.OPD4 | Primary HDU | Maximum(ABS(column(OPD,4))<br>Maximum OPD for baseline 4 |
| QC.MAX.OPD5 | Primary HDU | Maximum(ABS(column(OPD,5))<br>Maximum OPD for baseline 5 |
| QC.MAX.OPD6 | Primary HDU | Maximum(ABS(column(OPD,6))<br>Maximum OPD for baseline 6 |
| QC.FLUX.AVG | Primary HDU | Average(QC.DETi.FLUX.IPj) with i=1 when det.name = MATISSE-LM and i=2 when det.name = MATISSE-N and j=[1,3,5,7], with j referring to the input pupil<br>Flux average over the 4 beams |
| QC.FLUX.STDEV | Primary HDU | Stdev(QC.DETi.FLUX.IPj) with i=1 when det.name = MATISSE-LM and i=2 when det.name = MATISSE-N and j=[1,3,5,7], with j referring to the input pupil<br>Standard deviation of the flux over the 4 beams |
| QC.FLUX.SNR | Primary HDU | QC.FLUX.AVG/QC.FLUX.STDEV<br>Flux SNR over the 4 beams |
| QC.VIS.AVG | Primary HDU | Average(QC.DETi.VISSQRj.MEAN) with i=1 when det.name = MATISSE-LM and i=2 when det.name = MATISSE-N and j=[1,2,3,4,5,6], with j referring to the baseline<br>Averaged Visibility (6 baselines) |
| QC.VIS.STDEV | Primary HDU | Stdev(QC.DETi..VISSQRj.MEAN) with i=1 when det.name = MATISSE-LM and i=2 when det.name = MATISSE-N and j=[1,2,3,4,5,6], with j referring to the baseline<br>Standard deviation of the visibility (6 baselines) |
| QC.TF.AVG | Primary HDU | Average(QC.DETi.TFSQRj.MEAN) with i=1 when det.name = MATISSE-LM and i=2 when det.name = MATISSE-N and j=[1,2,3,4,5,6], with j referring to the baseline<br>Averaged Transfer function when available (6 baselines) |

| | | |
|---|---|---|
| QC.TF.STDEV | Primary HDU | Stdev(QC.DETi..TFSQRj.MEAN) with i=1 when det.name = MATISSE-LM and i=2 when det.name = MATISSE-N and j=[1,2,3,4,5,6], with j referring to the baseline<br>Standard deviation of the transfer function when available (6 baselines) |
| QC.TF.SNR | Primary HDU | QC.TF.AVG/QC.TF.STDEV<br>SNR on the Transfer Function (when available) |
| QC.AMBI.TRANS | Primary HDU | if OBS.AMBI.TRANS == '1PHO': QC.AMBI.TRANS = 1<br>if OBS.AMBI.TRANS == '2CLR': QC.AMBI.TRANS = 2<br>if OBS.AMBI.TRANS == '3THN': QC.AMBI.TRANS = 3<br>if OBS.AMBI.TRANS == '4THK': QC.AMBI.TRANS = 4<br>Recoding as integer of the transparency |
| QC.CORR.FLUX | Primary HDU | Value of the pipeline parameter 'corrFlux'<br>The pipeline product is a correlated flux or a Visibility |
| QC.FLAG.FLUX | Primary HDU | If there are NO keywords QC DETi FLUX IPj in the header, QC.FLAG.FLUX=2<br>If QC.CORR.FLUX='true' QC.FLAG.FLUX=3<br>If QC.CORR.FLUX='false' and all the QC.DETi FLUX IPj keywords > 0 QC.FLAG.FLUX=0<br>If QC.CORR.FLUX='false' and one of the QC.DETi FLUX IPj keywords <= 0 QC.FLAG.FLUX=10<br>j=[1,3,5,7], with j referring to the input pupil<br>Coding as an integer of the quality of the flux |
| QC.FLAG.FLUXSNR | Primary HDU | If there are NO keywords QC DETi FLUX IPj in the header, QC.FLAG.FLUXSNR=2<br>If QC.CORR.FLUX='true'<br>And QC.FLUX.SNR >= 2 QC.FLAG.FLUXSNR=1<br>If QC.CORR.FLUX='true'<br>And QC.FLUX.SNR < 2 QC.FLAG.FLUXSNR=11<br>If QC.CORR.FLUX='false'<br>And QC.FLUX.SNR >= 2 QC.FLAG.FLUXSNR=0<br>If QC.CORR.FLUX='false'<br>And QC.FLUX.SNR < 2 QC.FLAG.FLUXSNR=10<br>j=[1,3,5,7], with j referring to the input pupil<br>Coding as an integer of the quality of the SNR of the flux |
| QC.MAG | TPD | If DET.NAME=MATISSE-N then QC.MAG= PRO.JSDC.NMAG,<br>If DET.NAME=MATISSE-LM then QC.MAG= PRO.JSDC.LMAG<br>Magnitude at the observing band (from catalog) |

| | | |
|---|---|---|
| QC.OBJ.FLUX | TPD | If DET.NAME=MATISSE-N then QC.OBJ.FLUX= PRO.JSDC.NFLUX,<br>If DET.NAME=MATISSE-LM then QC.OBS.FLUX= PRO.JSDC.LFLUX<br>Flux at the observing band (from catalog) |
| QC.FLAG.VISSNR | Primary HDU | If the keyword QC DETi TFSQR1 MEAN exists<br>And QC.TF.SNR >=2 QC.FLAG.VISSNR=0<br>And QC.TF.SNR <QC.FLAG.VISSNR=10<br>If there is no keyword QC DETi TFSQR1 MEAN<br>And QC.VIS.SNR >=2 QC.FLAG.VISSNR=1<br>And QC.VIS.SNR <2 QC.FLAG.VISSNR=11<br>Coding as an integer of the quality of the SNR of the Visibilities |