Observatoire de Genève
51, Ch. des Maillettes, CH-1290 Sauverny
Phone: +41 22 755 26 11 * Fax: +41 22 755 39 83

GIRAFFE

BLDR Software - Reference Manual

Doc. No. VLT-SPE-OGL-13730-0040
Issue 1.09
15 October, 2003

Released
compiled by blecha on October 16, 2003 at 14:55

Prepared	A. Blecha, P. North, F. Royer, G. Simond *Name*	15 October, 2003 *Date*	..... *Signature*
Approved	A. Blecha *Name*	15 October, 2003 *Date*	..... *Signature*
Released	A. Blecha *Name*	15 October, 2003 *Date*	..... *Signature*

Change Record

Issue/Rev.	Date	Section/Page affected	Reason/Remarks
0.9	29 January, 2003	A. Blecha	Set-up of skeleton
1.09	29 Septembre, 2003	All sections	Python SW distribution

Table of Contents

1 Introduction **1**
- 1.1 How to use this document . . . 1
- 1.2 Stylistic conventions . . . 1
- 1.3 Abbreviations, Acronyms and Glossary . . . 1
- 1.4 Overview . . . 3
  - 1.4.1 Inputs . . . 3
  - 1.4.2 Outputs . . . 4
  - 1.4.3 Online mode . . . 4
  - 1.4.4 Off-line mode . . . 4
  - 1.4.5 Software External Components . . . 4
  - 1.4.6 Bad or unused fibers . . . 4
  - 1.4.7 Standards . . . 5

2 Functions **7**
- 2.1 General Description of Functions . . . 7
  - 2.1.1 Introduction . . . 7
  - 2.1.2 General remarks on the presentation of Functions . . . 9
- 2.2 Detailed description of functions . . . 11
  - 2.2.1 **giArithmetics** - Arithmetic operations on Images . . . 11
  - 2.2.2 giArithm.py . . . 12
  - 2.2.3 **giGetRemoveBias** - Get bias value over bias test area . . . 15
  - 2.2.4 giRecBias.py . . . 19
  - 2.2.5 **giSubtractDark** - Subtract Dark . . . 21
  - 2.2.6 giRecDark.py . . . 23
  - 2.2.7 **giDetectCosmicM** - Detect Cosmic ray hits in several frames . . . 25
  - 2.2.8 giRecCosmicMC.py . . . 29
  - 2.2.9 **giAdjustSL** - Adjust the scattered light . . . 31
  - 2.2.10 giRecSLfit.py . . . 33
  - 2.2.11 **giLocalSpectra** - Localization of spectra . . . 36
  - 2.2.12 giRecNLoc2.py . . . 39
  - 2.2.13 **giAdjustLoc** - Adjust localization . . . 43
  - 2.2.14 giRecDLocfit.py . . . 47
  - 2.2.15 **giFitPsfProfile** - Tranverse PSF modeling and accurate Localization . . . 49
  - 2.2.16 giRecPsfLoc.py . . . 54
  - 2.2.17 **giExtractSpectra** - Spectra Extraction . . . 56
  - 2.2.18 giRecNExt.py . . . 61
  - 2.2.19 **giGetWaveSolution** - Wavelength Solution . . . 64
  - 2.2.20 giRecNWcal2.py . . . 69
  - 2.2.21 **giRebinSpectra** - Rebinning in wavelength space . . . 74
  - 2.2.22 giRecBinn2.py . . . 77
  - 2.2.23 **giCrossC** - Computation and fit of the Cross Correlation peak. . . . 80
  - 2.2.24 giCrossC.py . . . 82

3 Data Description **89**
- 3.1 General description of Frames and Tables . . . 89
  - 3.1.1 Description of Frames . . . 89
  - 3.1.2 Description of Tables . . . 90
- 3.2 Detailed Description of Tables . . . 92

3.2.1 General remarks . . . 92
3.2.2 girDLocc - Chebyshev polynomial model of localisation correction obtained from 5 SIMCAL spectra . . . 92
3.2.3 girGrating - Grating Setup Constants . . . 93
3.2.4 girLine - Catalogue of emission lines used for Wavelength Calibration . . . 95
3.2.5 girLocc - Chebyshev polynomial model of localisation . . . 95
3.2.6 girOzpoz - Positioner binary table . . . 97
3.2.7 girRVMask - Binary mask for Correlation . . . 98
3.2.8 girSlitGeo - Geometry of slits - position of fibers . . . 99
3.2.9 girWavCoef - Coefficients of the Polynomial Wavelength Solution . . . 101
3.2.10 girWres - Wavelength spectra shift obtained from 5 SIMCAL spectra . . . 103

4 Processing recipes (pipes) 105
4.1 INTRODUCTION . . . 105
4.2 GENERAL INFORMATIONS ON PIPES . . . 106
4.3 DATA REDUCTION COOK-BOOK . . . 109
4.4 CALIBRATION REDUCTION COOK-BOOK . . . 111
4.5 RDB COOK-BOOK . . . 113
4.6 Calibration Pipes . . . 114
4.6.1 biasMast pipe . . . 114
4.6.2 darkMast pipe . . . 117
4.6.3 locMast pipe . . . 119
4.6.4 wcalMast pipe . . . 123
4.6.5 wcalOnly pipe . . . 128
4.6.6 wcalSlit pipe . . . 132
4.7 Preprocessing Pipes . . . 134
4.7.1 preproc pipe . . . 134
4.8 Extraction Pipes . . . 136
4.8.1 extract pipe . . . 137
4.8.2 rebin pipe . . . 140

Chapter 1

Introduction

1.1 How to use this document

This document is a merge of 2 documents; the GIRAFFE SW reference manual intended to provide all information necessary to fully understand the SW capabilities, data structure and implementation (Chapters 2 and 3 mainly). This part is intended to provide reference documentation to the SW engineer in charge of the integration in the VLT pipeline and *could be skipped at the first reading by the standard user.*

The *pipe* implementation reference manual follows in the chapter 4 which describes the functionalities and use of Python pipes.

The practical user will start with Chapter 4 and go back to Chapter 2 only when he/she wants to understand the processing further in details.

1.2 Stylistic conventions

The following styles are used in **Mathematical description** section for:

functions parameters
: `lowercaseTeletype` with each word without the first capitalized '`functionFirstParam`', for example: `darkThreshold`.
Corresponding LaTeX code:
`\texttt{darkThreshold}` or in math mode: `\mathtt{darkThreshold}`,

table names
: `lowercaseTeletype` with each word without the first capitalized '`exampleTableName`', for example: `lineSkyMast`.
Corresponding LaTeX code:
`\texttt{lineSkyMast}` or in math mode: `\mathtt{lineSkyMast}`,

`table columns`
: *ALL_UPPERCASE_ITALIC* prefixed with `'tableName:'` in the text, '`exTableName` :$FIRST\_COLUMN$' for example: `lineSkyMast` :$WLEN$.
Corresponding LaTeX code:
`$\mathtt{lineSkyMast:}WLEN$` or in math mode: `\mathtt{lineSkyMast:}WLEN`,

`frame names`
: *lowercaseItalic* with each word without the first capitalized '*exampleFrameName*', for example: *dbSubImage*.
Corresponding LaTeX code:
`$dbSubImage$` or in math mode: `dbSubImage`,

`frame keywords`
: `ALL_UPPERCASE_TELETYPE` prefixed with *'frameName.'* in the text, '*exFrameName*.`FITS_KEYWORD`' for example: *dbSubImage*.`EXPTIME`.
Corresponding LaTeX code:
`$dbSubImage.\mathtt{EXPTIME}$` or in math mode: `dbSubImage.\mathtt{EXPTIME}`,

1.3 Abbreviations, Acronyms and Glossary

ASCII	American Standard Code for Information Interchange
ADAS	GIRAFFE Ancillary Data Analysis Software - Data Analysis tools specific to GIRAFFE instrument
ADU	CCD Analog-to-digital Unit - Units used to quantify the CCD signal intensity
ARGUS	GIRAFFE Integral Field Spectroscopy mode
BLDR	GIRAFFE Base Line Data Reduction
BLDRS	GIRAFFE Base Line Data Reduction Software
BOS	Base Operating System
CCD	Charge Coupled Device
CRH	Cosmic Ray Hit (general term)
DFS	Data Flow Software - VLT software controlling the on-line data processing
DICB	Data Interface Control Board - VLT software
DMD	Data Management Division of ESO
DRS	Data Reduction Software (general term)
FF	Flat-Field (general term)
FITS	Flexible Image Transport System - file format for images and tables (general term)
FLAMES	VLT Multi Object Fiber Facility - UT2-VLT Instrument; GIRAFFE is a part of FLAMES
FORS	Focal Reducer Spectrograph - UT1-VLT instrument
FOV	Field of View (general term) used here in sense of Nasmyth FOV
FP	Fiber positionner - part of FLAMES instrument
FSFF	*Full-Slit* Flat Field - FF taken with entrance slit uniformly illuminated
FWHM	Full width at half-maximum
GIRAFFE	Spectrograph - this instrument, part of FLAMES
GUI	Graphical User Interface - set of command windows controlling the DRS (general term)
IFU	Multi-Integral Field Unit mode of GIRAFFE (15 IFUs of 20 image element + sky each)
IOCP	Instrument Operation and Calibration Plan (general term) but referenced here for FLAMES
ISAAC	Infrared Spectrometer And Array Camera - UT1-VLT instrument
KW	Keyword (usually FITS KW) - part of the FITS format
LSQ	Least Square Fit (general term)
MB	Mega-byte
MEDUSA	Multi-Object Spectroscopy mode of GIRAFFE (one fiber per object)
MIDAS	Munich Image Data Analysis System - Standard ESO Image Processing System
MOS	VLT Multi-object slit spectrograph (general term)
NFF	*Narrow* Flat Field - FF taken with slit illuminated through fibers
OGL	Observatoire de Genève et l'Université de Lausanne
OS	Operating system
OP	Observatoire de Paris
PAF	Parameter File Format - VLT DFS standard format
PC	Process Control - part of the BLDRS, set of tools that control the BLDRS execution
PHFF	Photometric Flat Field - FF taken with CCD illuminated directly (without passing through spectrograph)
PSF	Point Spread Function - image resulting from monochromatic point-like illumination of the entrance-slit
QC0	Quality Control Level 0 - results form the comparison of current parameters with nominal values
QC1	Quality Control Level 1 - results form the comparison of trends within the Calibration Database with expected fluctuations
RON	Read-Out Noise - CCD noise measured on 0 time exposure or on the image overscan
RTD	Real-Time Display - VLT tool for display and graphical handling of images
RV	Radial velocity
SEWC	Separate Wavelength Calibration - Calibration made with all fibers illuminated by calibration lamp
SIWC	Simultaneous Wavelength Calibration - Calibration made with 5 dedicated fibers illuminated by calibration lamp
SL	Scattered Light - parasite light with unknown spectral distribution falling on the detector
SLM	Scattered Light Model - continuous model built from discontinuous SL measurements
SNR	Signal to Noise Ratio - (general term)
SPU	Spectrograph unit - GIRAFFE spectrograph
SW	SoftWare
TBC	To Be Confirmed
TBD	To Be Defined
TBU	To Be Updated by ...

UT1-4	VLT Unit Telescope 1 to 4
UVES	UV-Visual Echelle Spectrograph - Third UT2-VLT instrument coupled to Fiber Positionner of FLAMES (8 fibers)
VCS	VLT Control Software
VI(R)MOS	Visual (Infrared) Multi Object Spectrograph - UT3-VLT multi-slit deep low-resolution spectrograph
WLC	or W-calibration Wavelength Calibration - (general term)

(BEGIN:dd.Overview.latex - last update blecha Tue 03/25/03 12:02:29)

1.4 Overview

The Data Reduction Software for GIRAFFE spectrograph relies on the specific GIRAFFE feature — the presence of five simultaneous calibration spectra — and the high instrument stability. The localization is adjusted for all exposures and several options are offered for the extraction.

The number of software modules is kept low through the use of same modules for multi-objet (MEDUSA) and integral-field spectroscopy (ARGUS and IFU) during most of the reduction steps. This is achieved through a careful parameterizing of the reduction functions.

The developed software relies only on UNIX environment and utilities available under UNIX public domain software and standard VLT tools. The link between the DRS and any specific data analysis system, if required, is possible through FITS and ASCII data structures only.

1.4.1 Inputs

On the input, the BLDRS receives the following data:

- The raw images from the VLT acquisition pipeline.

  The acquisition pipeline is a part of the VLT DFS and is running under the control of the FLAMES Base Operating System (BOS). The BLDRS communicates with the BOS through FITS data only. All the necessary *registration data* (Objects ID, positions, magnitudes) are supplied via FITS keywords on raw images, and FITS tables.

- The calibration data.

  The necessary calibration data and standard control parameters are extracted from the instrument database. Two types of calibration data are used:

  - The current calibrations.

    Current calibration files are the data obtained from the calibration pipeline. Most recent or standard calibration may be used. An example of these are the biases, flat-fields, localization or wavelength solutions.

  - The calibration constants.

    These are the data which are not obtained from the normal calibration pipeline. They are not strictly speaking constant, but could only be changed during the maintenance run through a special calibration process or imported by an ad-hoc command. The calibration constants are set before any standard Data Reduction could proceed. An example of such constants are the table of the spectrograph optical parameters, the slit geometry tables, the tables of spectral lines for the wavelength calibration and initial parameters for the processing algorithms.

- The control parameters.

  In order to make the automated processing possible and efficient, all options and controls are fully parameterized. A subset of comprehensible and widely accepted *classical* main parameters is put in evidence separately in order to facilitate the handling by technically inexpert user (with necessary astronomical background). An example is the level of the sigma-clipping, the rebinning method or the accepted level of the scattered light contamination.

1.4.2 Outputs

The BLDRS produces the following data:

- Flux (modulo the lamp spectrum) and wavelength calibrated spectra, associated error and number of pixels in each bin
- Standard set of numerical values that characterize the results and are used for the long term performance monitoring and quality control
- Set of report/history files and graphics (Python version only)

1.4.3 Online mode

All the BLDRS modules will be used into the VLT DFS framework and launched via the Reduction Block Scheduler. This is being done under ESO responsability using the BLDRS C-code only.

1.4.4 Off-line mode

In this mode the BLDRS modules consisting of stand-alone Python programs are chained by Unix C-shell scripts. The Python environment provides mostly the access to data structures, prototyping and testing facilities and some graphics. The actual number-crunching is coded in C-code. Same processing will be implemented through standard reduction recipes available in the VLT DFS.

For the off-line reduction process, all the necessary data products (observation and calibration frames, observation parameters, instrument data configurations, conditions and log files) are extracted from the VLT OLAS (On Line Archive System) or VLT science archive and forms the data-set delivered to the astronomer.
For maintenance or tests at the telescope, the required data are available from disk or from VLT OLAS.

Though the large number of observing modes (high and medium resolutions in all spectral ranges in MEDUSA, IFU and ARGUS modes and for all fiber assemblies) requires a large number of calibration files, the organization of the data-set provides information on how data products are related to each other. This is achieved using a standard Unix directory structure, file naming convention and/or data catalogs.

1.4.5 Software External Components

The following external components are used by the GIRAFFE BLDRS:

CFITSIO Software library to handle FITS formatted files

Python 2 with Numeric-22.0 and biggles-1.6.3

perl Practical Extraction and Report Language, with perlTk for the pipe GUI

Unix C-shell

For more detailed information see installation instruction of girbldrs-1.09 (files INSTALL, INSTALL.REQUIRED and INSTALL_EXAMPLE.Linux/SunOS)

1.4.6 Bad or unused fibers

Fibers and consequently the data associated to them through the slit geometry may have following status:

- Valid fibers (whatever may be the light source)
- Unused fibers, which may be either in parking position (no sky light) or left for sake of time efficiency at the current place on the plate.
- Damaged fibers

The dynamic exposure-related information on the fiber status is partly given by the binary `ozpoz` (present in the frame extension) table through column `OzPoz` :$BUTTON$. Only actually used buttons are listed there. This information is complemented by the static (from archive) `slitGeometry` table which associates the `ozpoz` :$BUTTON$ to `slitGeometry` :RP and provides informations necessary to associate `OzPoz` informations to extracted spectra. Note that while in Medusa mode there is one-to-one correspondance between BUTTON and fiber in IFU/Argus modes it is not the case and therefore the fibers removed from the system must be commented ny hand in `slitGeometry` table.

Note that `FLAMES_FIBRE_Table` given in the frame is not used since in its present version it does not provide information on fiber pattern in IFU mode.

During the preprocessing phase, data associated to all fibers receive the same treatment whatever their actual status.

Starting by localization, only the spectra associated to valid fibers are processed. Extracted spectra of invalid fibers are set to zero so as to make the data structures of the same dimension and compatible with the standard calibration data.

1.4.7 Standards

BLDRS data

All data needed or produced by the BLDRS modules are in one of the two following format:

- FITS images and tables (ESO compliant)
- flat ASCII tabulator delimited tables (internal standard, /rdb or starbase compatible)

BLDRS modules

The BLDRS Software is developed using C programming language following the VLT programming standards guidelines. All necessary scripts use a standard Unix shell (Bourne shell,C shell or Korn shell) or Python.

(END:dd.Overview.latex)

Chapter 2

Functions

2.1 General Description of Functions

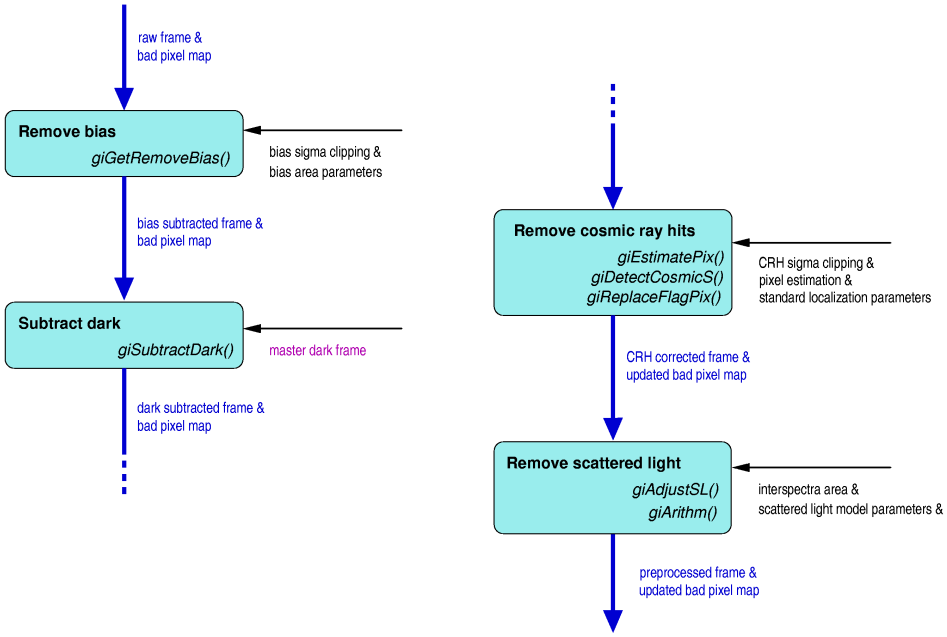
(SOURCE: dd.FunctionsGeneral.latex)

2.1.1 Introduction

The GIRAFFE BLDRS modules/functions are grouped as follow:

Preprocessing modules

- **Remove Bias:** Calls `giGetRemoveBias()` which computes and subtracts the bias value using CCD pre-scan and overscan areas of a raw frame, cuts out these areas and converts the flux from ADU to electrons. This is the first and compulsory step of the Preprocessing.
- **Subtract Dark:** Calls `giSubtractDark()` which subtracts the scaled (with exposure time and CCD gain) Master Dark Frame from the previously obtained pre–processed "bias–removed" frame.
- **Cosmics subtraction (optional):** Calls `giDetectCosmicM()` which removes Cosmic Rays Hits (CRH) on multiple frames.
- **Remove Scattered Light (optional):** Calls `giAdjustSL()` which adjusts a model of scattered light to the inter–spectra regions ready to subtract. Optionally a photometric Flat–Field calibration frame can be used to correct the previously obtained pre–processed frame before the modelization.



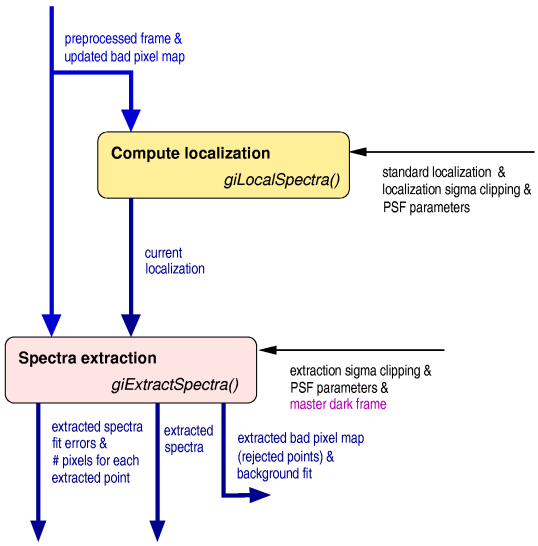
Extraction modules

- **Compute localization (only Flat–Field frame):** Calls `giLocalSpectra()` which determines the centroid position and half-width of each spectral bin for each spectrum from scratch, using a thresholding method, to produce the standard initial localization.

- **Compute transverse PSF (only Flat–Field frame) :** Calls `giFitPsfProfile()` which produces the global model of the transverse PSF (direction perpendicular to the dispersion). This model is necessary to carry on the optimal extraction.

- **Adjust localization:** On any other frame the standard localization could be adjusted through `giAdjustLoc()` function using simultaneous wavelength calibration spectra. This step could only be done if simultaneous wavelength calibration is present on the frame.

- **Spectra extraction:** Calls `giExtractSpectra()` which performs, using the current localization, and, optionnally the model of the transverse PSF (optimal extraction) the extraction of spectra of the pre–processed frame.



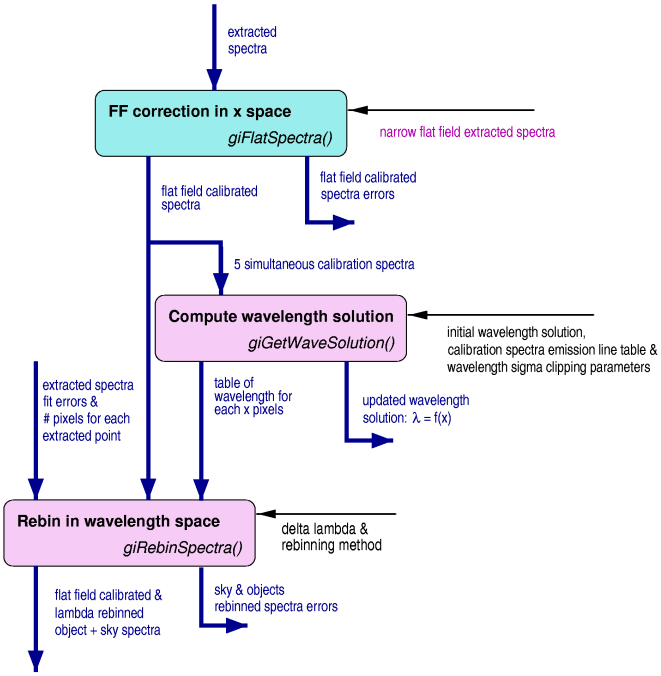
Flat–Field and wavelength calibration modules

- **Flat–Field correction in x space:** Calls `giFlatSpectra()` or `giArithm()` which corrects the extracted spectra for spectrograph and detector signature using a Narrow Flat–Field calibration extracted spectra, but they remain modulated by the spectrum of the Flat–Field calibration lamp.

- **Compute wavelength solution:** Calls `giGetWaveSolution()` which computes the standard wavelength solution using a full calibration ThAr exposure. This solution is used as a first approximation of the wavelength solution for all exposures of the same setup within the observing run.

- **Rebinning in wavelength space:** Calls `giRebinSpectra()` which carries out the rebinning of extracted spectra using the current wavelength solution, so that the resulting rebinned spectra are equally spaced in λ or $\log \lambda$.

- **Adjust wavelength solution:** Calls `giCrossCorrelate()` which computes the cross-correlation between the simultaneous wavelength ThAr calibration rebinned spectra and binary mask and produces interpolated corrections for all spectra which could be used during the subsequent (second) rebinning.



2.1.2 General remarks on the presentation of Functions

The functions operate on data in physical units. It is assumed, that the image elements and henceforth the extracted spectra are in photoelectron/element. The conversion is made during the first operation in the DRS which is normaly the bias determination and subtraction.

- For KWs we favor the type **float** to **real** or **double**. Only if well justified double precision is proposed (unless the KW is taken from existing dictionary). Fitted data are stored as double.
- We keep trace of our description and comments to ESO standard KWs. The following syntax is adopted:
  - The comment (if any) from Alias file comes first prefixed by **AL:**
  - The DIC ESO comment is prefixed by **DIC** and is appended to Alias comment if both are present.

Each BLDRS function is described as follows:

Name , **Section** , **Purpose** and internal running number **nu**

Function parameters list of function control parameters initialized and passed by the calling python recipe to the C-code. Note that the corresponding python parameters, which are accessible to user are sligthly different and are supplement by other parameters necessary to control the python recipe.

Input frames list of all input frames needed by the function with his name, type, format and description. The **Relevant FITS keywords** from the primary header used in the functions are listed:

- **Name** is the GIRAFFE BLDRS alias for the existing ESO HIERACH FITS keyword
- **DIC** is the name of the VLT DICB dictionary where the keyword is defined, '`DRS`' stands for GIRAFFE BLDRS own keyword and '`MISSING`' for keyword supposed to be provided but not found in any VLT dictionary
- **Type** , **Unit** and **Description** have the standard VLT dictionary meaning

Output frames list of all output frames produced by the function. Only new or updated FITS keywords are listed, all other FITS keywords for the output frames comes from the corresponding input frames. All functions parameters with their values used to produce the output frames are stored as FITS keywords in the frame header.

Other inputs/outputs list of all other inputs/outputs needed or produced by the function, this is essentially FITS data tables. These ones are listed with the names of the columns, their type and description needed or updated by the function. The full table description with its FITS header and complete list of columns can be found in the 'Data Description' section.

General description describing the function itself.

Mathematical description describing the mathematical algorithm used.

Description of Python implementation - this is a Python recipe help output obtained with '-h' option.

Python implementation default parameters - this is a Python recipe default values output obtained with '-z' option.

Python implementation test - command and data to execute test and examine results.

(END: dd.FunctionsGeneral.latex)

2.2 Detailed description of functions

2.2.1 giArithmetics - Arithmetic operations on Images

Name: **giArithmetics** Section: **Preprocessing**
Python name: **giArithm.py**

Purpose

Compute basic pixel-to-pixel operations between images.

Function parameters

All input parameters are in one *string* expression

Name	Type	Description
`expression`	`string`	arithmetic expression which operates on input frames

Input frames

Name: *inFrame1* **Type:** `PREPIMG` **Format:** `xyPrep`
Extension: `dbSub.fits`
Description: any 0-3D image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
`NAXIS1`	PR_FITS	`integer`	none	DIC: # of pixels in <axis direction>
`NAXIS2`	PR_FITS	`integer`	none	DIC: # of pixels in <axis direction>

Name: *inFrame2* **Type:** `PREPIMG` **Format:** `xyPrep`
Extension: `dbSub.fits`
Description: any 0-3D image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
`NAXIS1`	PR_FITS	`integer`	none	DIC: # of pixels in <axis direction>
`NAXIS2`	PR_FITS	`integer`	none	DIC: # of pixels in <axis direction>

Name: *badPixMap1* **Type:** `BADPIX` **Format:** `xyPrep`
Extension: `.crmbpm.fits`
Description: Optional current $inFrame1$ bad pixel map
Relevant FITS keywords:

Name	DIC	Type	Unit	Description

Name: *badPixMap2* **Type:** `BADPIX` **Format:** `xyPrep`
Extension: `.crmbpm.fits`
Description: Optional current $inFrame2$ bad pixel map
Relevant FITS keywords:

Name	DIC	Type	Unit	Description

Other inputs

none

Output frames

Name: *outFrame* **Type:** `PREPIMG` **Format:** `xyPrep`
Extension: `dbSub.fits`
Description: any 0-3D image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
`NAXIS1`	PR_FITS	`integer`	none	DIC: # of pixels in <axis direction>
`NAXIS2`	PR_FITS	`integer`	none	DIC: # of pixels in <axis direction>
`DATAMIN`	PR_FITS	`double`	e-	DIC: Minimal pixel value
`DATAMAX`	PR_FITS	`double`	e-	DIC: Maximal pixel value
`DATAMEAN`	PRO	`double`	e-	DIC: Mean of the pixel values in the frame. All pixels are taken into account. AL: Average signal of valid pixels
`DATAMED`	PRO	`double`	e-	DIC: Exact median of the pixel values in the frame. All pixels are taken into account. AL: Median signal of valid pixels
`DATARMS`	PRO	`double`	e-	Sigma of the signal of valid pixels

Name: *badPixMap* **Type:** `BADPIX` **Format:** `xyPrep`
Extension: `.crmbpm.fits`
Description: Optional updated current *outFrame* bad pixel map
Relevant FITS keywords:

Name	DIC	Type	Unit	Description

Other outputs

none

General description

Basic arithmetical operations are computed over 2D images and/or masks. See Python description in this section for currently supported syntax. This function, temporarily replaces the flat-fielding and normalisation of extracted images and also generates static initial mask necessary to remove CCD defects. Besides it is used for test purposes.

(OK blecha Mon 02/03/03 18:16:33)

Mathematical description

This function computes operations (unary and binary) over 2D images. Polish notation is used. See Python description in this section for currently supported syntax.

(OK blecha Mon 02/03/03 18:16:33)

Statistical keywords are computed over the resulting frame:

- `DATAMIN` : the minimum intensity value
- `DATAMAX` : the maximum intensity value
- `DATAMEAN` : the mean intensity value
- `DATAMED` : the median intensity value
- `DATARMS` : the standard deviation of intensity values

2.2.2 giArithm.py

Python implementation description

```
(Following description could be obtained on-line using: giArithm.py -h )
GIRAFFE BLDRS plots

   Functions
```

```
=========

giArithm     - make arithmetics on matching files (same NAXIS1/2)

SYNOPSIS
  giArithm [-h|--help]
or
  giArithm [options] expression

  only simple expressions, enclosed by '' are supported:

    operators:        *,/DIV,+,-
                      note that 0 results when dividing by 0
    operands:         numbers or images (format: xxxx.fits)
                      if any operand contains '?' wildcards, the expression is repeated
                      with all instances of the operand and the output file is numbered
                      with 2-digits
    functions:        average()

    EX: 'clean.fits=dbSub.fits-SL.fits'
        'clean.fits=dbSub.fits*5-SL.fits+25'
        'average.fits=average(image?.fits)'     # wild-card '?' supported
        ' .... --STAT dbSub.fits  qq.fits ....' # Only statistics

  No parenthesis are implemented, expression is executed left -> right:

    qq=10+20*300 is executed as qq=(10+20)*300

  To  check  the  syntax  you  may  execute purely numerical
  expression with output to a symbolic variable as:

    a=10+20*3

  Note that if several images are used as arguments, resulting image inherit the header
  of the first image.

DESCRIPTION

Do simple arithmetics on images

General options:
   -h|--help               - display help and exit
   -d|--debug              - debug messages
   -v|--verbose            - verbose processing
   -q|--quiet              - no messages
   -z|--defaults           - show default values

Processing limits:
   -X <X1:X2>              - only part of image X1<=x<X2,Y1:Y2 to output
   -Y <Y1:Y2>
   -N <NSP0>               - update KW of number of spectra on image
                             (for test purpose only if truncated in Y)
Special options:

   -Z <zmin:zmax:'in/out'> - result will be a mask with 1 at pixels
                             if 'in':  zmin<=pix_value<zmax
                             if 'out': zmin>pix_value | pix_value<=zmax
                             if "-" is given default zmin:zmax are used
                             if 's' is appended to zmin or/and zmax we take
                             average+zmin*sigma and or average+zmax*sigma instead of zmin;zmax
                             EX: -Z 100:200:in
                                 -Z -10s:10s:out
   -S <x1:x2:y1:y2,...>    - used with '-Z' option: set value to 1 on this area
                             EX: -S 0:4096:414:420 'mask.fits=biasMast.fits'
                                 # bad columns of EEV on image with overscans
```

```
                                -S 0:4096:364:370 'mask.fits=biasMast.fits'
                                # bad columns of EEV on image without overscans
                              You may specify -S def
    --STAT                  - print statistics of arguments: average sigma minimum
                              maximum
                              if  the  expression is mere name (no '=' present)
                              only statistics is produced
    --norm                  - normalise output so as to have output_average=1
    --nan <nanval>          - replace 'not-a-numbers' by 'nanval'
                              default is: --nan 0
    --nonan                 - do not replace 'nan's
    --average               - take average of argument images
    --gain <KW>             - multiply result by value in the KW;
                              if KW='-' we use default 'CONAD' (ESO DET OUT CONAD)

  Outputs
    --on                    - forced numbering of the result file
    --outdir <dir>          - output directory for products
    -w|--ovwrite            - overwrite product files
```

Python defaults

```
 Source: 29278 oct 16 10:10 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giArithm.py
(Following default values could be obtained on-line using: giArithm.py -z )
    #$Id: giArithm.py,v 1.4 2003/06/20 13:01:53 anonymous Exp $
    v(erbose) : 0
    d(ebug)   : 0
    expression:
    args      : ['']
    ops       : ['']
    norm      : 0
    nan (val) : 1 ( 0 )
    ovwrite   : 0
    X         : 0:-0
    Y         : 0:-0
    N         : 0
    STAT      : 0
```

2.2.3 giGetRemoveBias - Get bias value over bias test area

Name: **giGetRemoveBias** Section: **Preprocessing**
Python name: **giRecBias.py**

Purpose

The bias and the read-out noise are obtained as the average value and sigma over the overscan areas. The bias value is subtracted and the overscan lanes are trimmed out. The flux is also converted from ADU to e^-.

Function parameters

Name	Type	Description
bSigma	float	multiple of σ (σ-clipping)
bNiter	integer	number of iterations (σ-clipping)
bMfrac	float	min fraction of points accepted/total (σ-clipping)
bMethod	string	Method used, one of: "UNIFORM", "PLANE" ,"CURVE" + optionally one of: "MASTER", "ZMASTER"
bRemove	integer	1/0 we remove or not the bias
blimits	integers	area where we take the bias: X11:X21:Y11:Y21,X12:X22:Y12:Y22,...
xorder	integer	degree of the polynomial fit in X
yorder	integer	degree of the polynomial fit in Y
xstep	integer	X binning for fit
ystep	integer	Y binning for fit

Input frames

Name: *rawFrame* **Type:** RAWIMG **Format:** xy
Extension:
Description: raw frame with bias prescans and overscans included
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PRESCX	CCDDCS	integer	none	DIC: Number of pixels in X read before real data arrives.
PRESCY	CCDDCS	integer	none	DIC: Number of pixels in Y read before real data arrives.
OVRSCX	CCDDCS	integer	none	DIC: Number of pixels in X read as overscan.
OVRSCY	CCDDCS	integer	none	DIC: Number of pixels in Y read as overscan.
CONAD	CCDDCS	double	e-/ADU	DIC: The conversion factor which translates ADU to photonic electrons.

Name: *biasMast* **Type:** BIASIMG **Format:** xy
Extension: biasMast.fits
Description: master bias
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BIASVALUE	DRS	float	ADU	Computed average bias level in pre/over-scan

Name: *badPixMast* **Type:** BADPIX **Format:** xyPrep
Extension: .crmbpm.fits
Description: Optional bias bad pixel map with active flag 'bBadPixFlag'
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
CCDID	CCDDCS	string	none	DIC: Detector system identification Format: <CCD Id> - <ACE Id>

Other inputs

none

Output frames

Name: *bSubImage* **Type:** `BRMIMG` **Format:** `xyPrep`
Extension: `bSub.fits`
Description: bias subtracted image converted in electrons with overscan lane trimmed out
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
`BITPIX`	PR_FITS	`integer`	none	DIC: # of bits per pix value
`BZERO`	PR_FITS	`double`	none	DIC: real=pixel_value*BSCALE+BZERO
`BSCALE`	PR_FITS	`double`	none	DIC: real=pixel_value*BSCALE+BZERO
`CREATOR`	DRS	`string`	none	Source of data (responsible or processing recipe)
`DATAMIN`	PR_FITS	`double`	e-	DIC: Minimal pixel value
`DATAMAX`	PR_FITS	`double`	e-	DIC: Maximal pixel value
`DATAMODE`	DRS	`double`	e-	Mode of the signal of valid pixels
`DATAMEAN`	PRO	`double`	e-	DIC: Mean of the pixel values in the frame. All pixels are taken into account. AL: Average signal of valid pixels
`BIASMETHOD`	DRS	`string`	none	Method used to subtract bias; one of UNIFORM, PLANE, CURVE + MASTER, ZMASTER
`BCLIPSIGMA`	DRS	`float`	none	multiple of sigma (sigma-clipping)
`BCLIPNITER`	DRS	`integer`	none	number of iterations (sigma-clipping)
`BCLIPMFRAC`	DRS	`float`	none	min fraction of points accepted/total (sigma-clipping)
`BIASSIGMA`	DRS	`float`	ADU	Computed bias sigma in pre/over-scan
`BIASVALUE`	DRS	`float`	ADU	Computed average bias level in pre/over-scan
`BIASPLANE`	DRS	`float`	ADU	n coefficients of the x,y surface describing the fitted bias=B0+Bx*X+By*Y...; X,Y origine is at the first active pixel.
`BIASAREAS`	DRS	`string`	none	Areas used to compute bias; x11:x21:y11:y11,x12:x22:y12:y22
`BIASFILE`	DRS	`string`	none	Bias Master used

Other outputs

none

General description

This is the first processing of any GIRAFFE raw image. It is the only function operating on images larger than the CCD active area (pre/overscans present) and using ADU units.

The Bias Areas in the prescan and overscan lanes in both, x and y, directions are defined on the command line or through (not yet implemented) the `biasLimitsMast` table. A 2D polynomial is fitted on the data in these Bias Areas and the linear coefficients describing the plane are saved as quality indicators in *bSubImage*.`BIASPLANE`.

The control parameter <`bMethod`> is used to differentiate various levels of the processing: the single valued average subtraction, the plane or the curve subtraction and the *biasMast* subtraction with or without zero adjustment.

In any case the pre/overscans are removed prior the bias subtraction (the *biasMast* not having necessarily the same size of pre/overscans as *rawFrame*) and the *rawFrame* is converted from ADU to e^- using the keyword *rawFrame*.`CONAD`.

Mathematical description

At least 2 areas (line prescan and overscan) are necessary, but it is desirable to use also the image (column) pre/overscans. Area are given in as string <`blimits`> =Xstart1:Xend1:Ystart1:Yend1, ... (Fig. 2.1).

The control parameter <`bMethod`> is used as follows:

- <`bMethod`> = "UNIFORM"

  In the constant model, *bSubImage*.`BIASVALUE` is set to the mean of the pixel values within the defined bias areas (<`blimits`>).

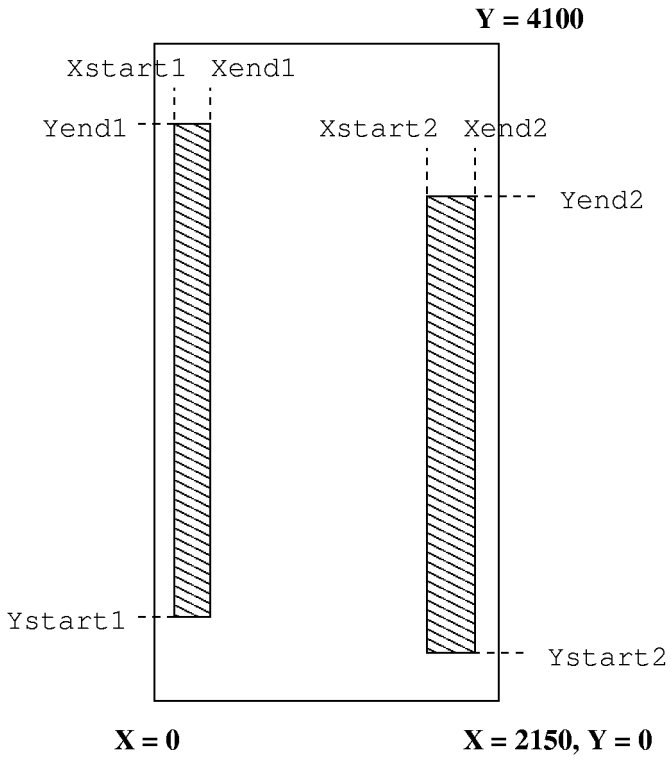


Figure 2.1: Sketch of the CCD frame with the Bias Areas and the corresponding positions of their coordinates.

The average bias $bSubImage$.BIASVALUE is subtracted.
$bSubImage(x, y) = rawFrame(x, y) - bSubImage$.BIASVALUE

- <bMethod> = "PLANE"
  When the chosen model is a plane, a bi-linear form $Bmodel = B_0 + B_x.X + B_y.Y$ is fitted with sigma-clipping. A coordinate system with [0,0] on the first active pixel [PRESCX + 1, PRESCY + 1] is used. In each iteration *iter* controlled by input parameters <bSigma> , <bNiter> , and <bMfrac> , the plane or curve is fitted and the biasSigma over valid points is computed as follows:

$$biasSigma_{iter} = \left(1/bNpAccept \sum_{i=1}^{bNpAccept} (rawFrame(x_{j_i}, y_{j_i}) - Bmodel(x_{j_i}, y_{j_i}))^2 \right)^{0.5}$$

  Only valid pixels $[x_{j_i}, y_{j_i}]$ satisfying condition

$$\mid rawFrame(x_{j_i}, y_{j_i}) - Bmodel(x_{j_i}, y_{j_i}) \mid < \texttt{bSigma} \times biasSigma_{iter-1}$$

  are retained. The sigma-clipping is repeated till no new points are rejected or the <bNiter> iteration is reached or when too many points are rejected. ($\frac{bNpAccept}{bNpTot} >$ bMfrac). The average bias of the $rawFrame$ (used and saved further as the keyword $bSubImage$.BIASVALUE) is computed as

$$BIASVALUE = Bmodel(X_{center}, Y_{center}),$$

  where X_{center}, Y_{center} are coordinates of the center of the CCD active image as given in 2.1. Note that the formula 2.1 refers to keywords of the $rawFrame$.

$$\begin{aligned} X_{center} &= (NAXIS1 - (PRESCX + OVERSCX + 1)) \times 0.5 \\ Y_{center} &= (NAXIS2 - (PRESCY + OVERSCY + 1)) \times 0.5 \end{aligned} \tag{2.1}$$

  The fitted plane rather than a single value is subtracted pixel-by-pixel.
  $bSubImage(x, y) = rawFrame(x, y) - Bmodel(x, y)$

- <bMethod> = "CURVE"

  This case is very similar to the previous one (<bMethod> ="PLANE"), but the bi-linear form $Bmodel(x, y)$ is replaced by 2D polynomial defined by the degrees <xorder> and <yorder> .

  The keyword $bSubImage$.BIASVALUE is set to the median value of the polynomial $Bmodel(x, y)$. And this polynomial curve is subtracted pixel-by-pixel, as in the previous case:

$$bSubImage(x, y) = rawFrame(x, y) - Bmodel(x, y)$$

- <bMethod> = "MASTER"

  The $biasMast$ image is subtracted without modification. The actual processing depends on the presence/absence of the $badPixMast$.

  – "MASTER" alone:
  $bSubImage(x, y) = rawFrame(x, y) - biasMast(x, y)$
  – "MASTER" + ("PLANE" or "CURVE"):
  In this case, the bad pixel mask $badPixMast$ is to be provided. The bit 'bBadPixFlag' is extracted from $badPixMast$ to form logical mask $bBadPixMask$ which is applied as follows:

$$\begin{aligned} bSubImage(x, y) \quad = \quad & (rawFrame(x, y) - biasMast(x, y)) * bBadPixMask(x, y) \cup \\ & (rawFrame(x, y) - Bmodel(x, y)) * (1 - bBadPixMask(x, y)) \end{aligned}$$

  The method to obtain the bias model $Bmodel(x, y)$ is one of "PLANE" or "CURVE".

- <bMethod> = "ZMASTER"

  A possible drift of average bias is compensated using the pre/overscans reference value. We consider the $biasMast$ as an additive correction and therefore the adjustment is made by the shift of the $biasMast$ and not by the scaling (multiplication).

  Prior to any processing, the $biasMast$ frame is corrected for the drift:

$$biasMast(x, y) = biasMast(x, y) - biasMast.\texttt{BIASVALUE} + bSubImage.\texttt{BIASVALUE}$$

  As for method "MASTER", the subsequent processing depends on the presence/absence of the $badPixMast$

  – "ZMASTER" alone:

$$bSubImage(x, y) = rawFrame(x, y) - biasMast(x, y)$$

  – "ZMASTER" + ("PLANE" or "CURVE"): In this case, the bad pixel mask $badPixMast$ is to be provided. The bit 'bBadPixFlag' is extracted from $badPixMast$ to form logical mask $bBadPixMask$ which is applied as follows:

$$\begin{aligned} bSubImage(x, y) \quad = \quad & (rawFrame(x, y) - biasMast(x, y)) * bBadPixMask(x, y) \cup \\ & (rawFrame(x, y) - Bmodel(x, y)) * (1 - bBadPixMask(x, y)) \end{aligned}$$

  The method to obtain the bias model $Bmodel(x, y)$ is one of "PLANE" or "CURVE".

Pre/overscans lanes are removed from $rawFrame$.
The $rawFrame$ is converted from ADU to e^-.

$$bSubImage(x, y) = bSubImage(x, y) \times rawFrame.\texttt{CONAD}$$

Note that the bias is not subtracted if <bRemove> =0.
The subtraction of the $biasMast$ is made on the already trimmed frame (the $biasMast$ not having necessarily same size of pre/overscans as $rawFrame$).

Note that keywords (PRE/OVR)SC(X/Y) are left unchanged on the output image though the prescan/overscan lanes are trimmed out.

The model of the bias built from (PRE/OVR)SCAN slightly differs from the model built on the full biases. This difference seems to be well conserved during the commissionning period and could be taken into account.

2.2.4 giRecBias.py

Python implementation description

```
(Following description could be obtained on-line using: giRecBias.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]

   NAME
   giRecBias.py - CCD bias correction recipe.

   SYNOPSIS
           giRecBias.py [-h|--help]
   or
           giRecBias.py [options] [rawFrameList [mbiasFrame]]

           rawFrameList - coma-separated list or
                          quote-enclosed wild card with '?' for each character

   DESCRIPTION
   Run giGetRemoveBias() on the specified raw frames <rawFrameList>
   where:
         <mbiasFrame>     - master bias frame filename, '-' for none
                            default none

   valid options are:
         -h|--help                display help and exit
         -d|--debug               set debug mode
         -v|--verbose             set verbosity level,
                                  repeated occurence increase verbosity
         -q|--quiet               no messages
         -t|--time                show processing time
         -z|--defaults            show default values
         -b|--bpmap <bpixFile>    specify bad pixel map filename
                                  '-' for none
         -o|--output <outFrameLList> coma-sepaarted output frame filename list
                                  must have same number of elements as rawFrameList
                                  or single-element; in last case numbered names
                                  are generated
                                  '-' for none
         --on <output_filename>  - force output file numbering
         --params                 dump INI config parameters
         -w|--ovwrite             overwrite product files
         --novwrite               do not overwrite product files
         --[no]compress           [do not] compress product files
         --outdir <dir>           output directory for products
         --datadir <dir>          specify input data directory
         --bmethod <method>       specify bias correction method, one of:
                                  UNIFORM,PLANE,MASTER,ZMASTER,MASTER+PLANE,
                                  ZMASTER+PLANE,CURVE,MASTER+CURVE,ZMASTER+CURVE
         --blimits <limits>       specify bias areas limits as a FITS table
                                  filename or as a string:
                                      'Xl0:Xr0:Yl0:Yr0,...,Xln:Xrn:Yln:Yrn'
         --bremove                do subtract bias
         --nobremove              do not subtract bias
         --noblimits              do not use default bias areas limits
         --bsigma <sigma>         bias-clipping: multiple of sigma
         --bniter <niter>         bias-clipping: number of iterations
         --bmfrac <mfrac>         bias-clipping: min fraction of points
                                   accepted/total. Should be in range [0.0-1.0]
```

```
      BIAS_CURVE method specific options
       --xorder <xdeg>         order of X polynomial fit
       --yorder <ydeg>         order of Y polynomial fit
                               xdeg = ydeg = 1 will fit a plane
       --xstep <step>          sampling step along X (every 'xstep' pixel taken)
       --ystep <step>          sampling step along Y (every 'ystep' pixel taken)

  OUTPUTS
  unless specified with -o or --output resulting frame goes to
  <rawFrame>.bsub.fits

  ENVIRONMENT VARIABLES
    PYTHONSTARTUP  - python startup script
    GIR_PYTHON     - GIRAFFE BLDRS python modules directory
    GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
    GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 17374 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecBias.py
(Following default values could be obtained on-line using: giRecBias.py -z )
   Reading BLDRS parameters from girBLDRS.ini:[General]
   #$Id: giRecBias.py,v 1.8 2003/07/20 09:52:26 anonymous Exp $
    rawFile = []
    bpixFile= -
    biasFile= -
    bpmap   : -
    bmethod : UNIFORM
    bremove : 1
    bsigma  : 2.5
    bniter  : 5
    bmfrac  : 0.8
    blimits : None
    output  = []
    verbose : 1
    debug   : 0
    time    : 1
    ovwrite : 0
    datadir : None
    outdir  : None
    cfgdir  : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
```

2.2.5 giSubtractDark - Subtract Dark

Name: **giSubtractDark** Section: **Preprocessing**
Python name: **giRecDark.py**

Purpose

The scaled (with exposure time) current standard mean dark is subtracted from the current image.

Function parameters

Name	Type	Description
dThresh	float	dark threshold value in e/pixel
dMethod	string	one of "UNIFORM" or "MASTER"

Input frames

Name: *bSubImage* **Type:** BRMIMG **Format:** xyPrep
Extension: bSub.fits
Description: any bias subtracted GIRAFFE frame
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NAXIS1	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
NAXIS2	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
EXPTIME	PR_FITS	double	s	DIC: Integration time

Name: *darkMast* **Type:** DARKIMG **Format:** xyPrep
Extension: darkMast.fits
Description: dark frame in electron/second units (Master Dark Frame)
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NAXIS1	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
NAXIS2	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
EXPTIME	PR_FITS	double	s	DIC: Integration time

Name: *badPixMast* **Type:** BADPIX **Format:** xyPrep
Extension: .crmbpm.fits
Description: Current *inFrame*1 bad pixel map
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NAXIS1	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
NAXIS2	PR_FITS	integer	none	DIC: # of pixels in <axis direction>

Other inputs

none

Output frames

bad pixel map = badPixMaster unchanged

Name: *dbSubImage* **Type:** BDRMIMG **Format:** xy
Extension: dbSub.fits
Description: dark and bias subtracted frame
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BITPIX	PR_FITS	integer	none	DIC: # of bits per pix value
BZERO	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
BSCALE	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
CREATOR	DRS	string	none	Source of data (responsible or processing recipe)
DATAMIN	PR_FITS	double	e-	DIC: Minimal pixel value
DATAMAX	PR_FITS	double	e-	DIC: Maximal pixel value
DATAMODE	DRS	double	e-	Mode of the signal of valid pixels
DATAMEAN	PRO	double	e-	DIC: Mean of the pixel values in the frame. All pixels are taken into account. AL: Average signal of valid pixels
DARKEXPECT	DRS	float	e-	The value of the expected dark current as computed by subtractDark from masterDark and exposure time
DARKVALUE	DRS	float	e-	The value of the actualy subtracted dark current. It is set to 0 if no subtraction is made (treshold not reached).
DARKMETHOD	DRS	string	none	One of 'UNIFORM' (we subtract a single number) and 'MASTER' (we subtract a scaled masterDark).
DARKTHRESH	DRS	float	e-	If the average expected dark is below this value, no dark subtraction is made.

Other outputs

none

General description

The bias subtracted frame is corrected for dark current. Based on the maximum value of expected dark current and the threshold given as input parameter <dThresh> , the decision whether the dark current is subtracted is taken. The <dMethod> indicates whether a constant value (this method does not introduce additional random noise) or a masterDark should be used. The first "UNIFORM" method is intended typically for short exposures (to prevent any average bias due to the unsubtracted dark) while "MASTER" is appropriate for long exposures. The *badPixMast* is associated to *darkMast* and flag 'dBadPixFlag' is used to identify pixels where the dark current is far above the average. The *badPixMast* is used together with the <dThresh> input parameter to enable the local use of *darkMast*.

Mathematical description

The dark current subtraction proceeds as follows:

1. Get expected values

   The maximum, *darkMaxExpected*, and the mode, *darkModExpected*, of the expected dark are computed using the keywords *darkMast*.DATAMAX and *darkMast*.DATAMOD and the exposure times *bSubImage*.EXPTIME and *darkMast*.EXPTIME of the processed frame and of the master dark:

   $darkMaxExpected = darkMast.\texttt{DATAMAX}\frac{bSubImage.\texttt{EXPTIME}}{darkMast.\texttt{EXPTIME}}$

   $darkModExpected = darkMast.\texttt{DATAMOD}\frac{bSubImage.\texttt{EXPTIME}}{darkMast.\texttt{EXPTIME}}$

2. Subtract dark - <dMethod> = "UNIFORM"

   We subtract the constant, uniform dark equal to *darkModExpected*, for each pixel. If the darkThreshold > *darkMaxExpected* no dark subtraction is made.

3. Subtract dark - <dMethod> = "MASTER"

   We subtract *darkMast* scaled by the exposure time of the processed frame (*bSubImage*.EXPTIME) on selected pixels and *darkModExpected* constant values on unselected pixels. The pixel selection is governed

by the presence/absence of the $badPixMast$ and the value of <dThresh> . A selection mask is either extracted from $badPixMast$ or built using the <dThresh> value.

Two preliminary steps are necessary:

- Scaling of $darkMast$ according to the exposure times:
  $scaledDark(x,y) = darkMast(x,y)\frac{bSubImage.\texttt{EXPTIME}}{darkMast.\texttt{EXPTIME}}$
- Building of the internal selection mask dMastPixMask
  (a) $badPixMast$ is absent and <dThresh> =0
  All points are selected for the pixel-to-pixel subtraction.
  $dMastPixMask = 1$
  Note that the implementation do not actually build (unnecessarily) the mask in this case since the general formula 2.2 simplify.
  (b) $badPixMast$ is absent and <dThresh> >0
  Pixels with expected dark above the <darkThreshold> are selected.
  $dMastPixMask(x,y) = scaledDark(x,y) >$ dThresh
  (c) $badPixMast$ is present
  The bit 'dBadPixFlag' is extracted from $badPixMast$ to form logical mask and inverted to form dMastPixMask so as the bits flagged in the $badPixMast$ are set to 1 in the dMastPixMask.

Eventually, the $scaledDark(x,y)$ image and $darkModExpect$ constant are merged according the formula:

$$\begin{aligned} scaledDark(x,y) &= scaledDark(x,y) * dMastPixMask(x,y) \cup \\ & \quad darkModExpected * (1 - dMastPixMask(x,y)) \end{aligned} \tag{2.2}$$

and subtracted from input frame

$dbSubImage = bSubImage(x,y) - scaledDark(x,y)$

4. update $dbSubImage$ keywords:

   The $dbSubImage$.DATAMODE is set to the $darkModExpected$, $dbSubImage$.DARKMETHOD is set to <dMethod> with 'BADMASK' appended if $badPixMast$ was used and all dark-associated KW ($dbSubImage$.DARKMETHOD, $dbSubImage$.DARKTHRESH) are written to the output image $dbSubImage$.

Note that in both methods and all cases, all pixels of $bSubImage$ are affected and $badPixMast$ remains unchanged and therefore there is no output associate to the $badPixMast$ input.

2.2.6 giRecDark.py

Python implementation description

```
(Following description could be obtained on-line using: giRecDark.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]

   NAME
   giRecDark.py - CCD dark correction recipe.

   SYNOPSIS
           giRecDark.py [-h|--help]
   or
           giRecDark.py [options] [bsubFrame [mdarkFrame]]

   DESCRIPTION
   Run giSubtractDark() on the specified bias corrected frame <bsubFrame>
   where:
         <mdarkFrame>     - master dark frame filename, '-' for none

   valid options are:
         -h|--help               display help and exit
         -d|--debug              set debug mode
         -v|--verbose            set verbosity level,
```

```
                          repeated occurence increase verbosity
    -q|--quiet            no messages
    -t|--time             show processing time
    -z|--defaults         show default values
    -o|--output <outFrame>  specify output frame filename
                          '-' for none
    -b|--bpmap <bpixmap>  specify bad pixel map filename
    --params              dump INI config parameters
    -w|--ovwrite          overwrite product files
    --novwrite            do not overwrite product files
    --[no]compress        [do not] compress product files
    --outdir <dir>        output directory for products
    --datadir <dir>       specify input data directory
    --dmethod <method>    specify bias correction method, one of:
                          UNIFORM,MASTER
    --dthresh <thresh>    dark threshold value

OUTPUTS
unless specified with -o or --output resulting frame goes to
<bsubFile>.dbsub.fits

ENVIRONMENT VARIABLES
  PYTHONSTARTUP  - python startup script
  GIR_PYTHON     - GIRAFFE BLDRS python modules directory
  GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
  GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 12313 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecDark.py
(Following default values could be obtained on-line using: giRecDark.py -z )
   Reading BLDRS parameters from girBLDRS.ini:[General]
   $Id: giRecDark.py,v 1.2 2003/03/25 15:18:19 anonymous Exp $
    bsubFile= /giraffe/blecha/BLDRS/girBLDRS1.09/python/girData/ThaMedFullM10.bsub.fits
    bpixFile= -
    darkFile= /giraffe/blecha/BLDRS/girBLDRS1.09/python/girRecipes/darkFullMast.fits
    bpmap   : -
    dmethod : UNIFORM
    dthresh : 2.5
    output  = ThaMedFullM10.bsub.dbsub.fits
    verbose : 1
    debug   : 0
    time    : 1
    ovwrite : 0
    datadir : None
    outdir  : None
    cfgdir  : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
```

2.2.7 giDetectCosmicM - Detect Cosmic ray hits in several frames

Name: **giDetectCosmicM** Section: **Preprocessing**
Python name: **giRecCosmicMC.py**

Purpose

The detection of CRH is obtained via the sigma-clipped average of all input images.

Function parameters

Control parameters for σ-clipping average

Name	Type	Description
crSigma	float	multiple of σ (σ-clipping)
crNiter	integer	number of iterations (σ-clipping)
crMfrac	float	min fraction of points accepted/total (σ-clipping)
crMethod	string	"DATA": noise from data, "MODEL": noise from model
avgMethod	string	averaging method: "MEDIAN" or "MEAN"
scaling	logical	flag indicating whether data are to be scaled or not
alertVal	float	number of cosmic ray hits used as threshold for an alert

Input frames

CRH detection for a set of multiple frames taken in similar conditions and with the same exposure time.

Name: *dbSubImage* **Type:** BDRMIMG **Format:** xy
Extension: dbSub.fits
Description: *Nimage* similar bias-dark-subtracted GIRAFFE frames
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
CONAD	CCDDCS	double	e-/ADU	DIC: The conversion factor which translates ADU to photonic electrons.
EXPTIME	PR_FITS	double	s	DIC: Integration time
BIASSIGMA	DRS	float	ADU	Computed bias sigma in pre/over-scan
DARKVALUE	DRS	float	e-	The value of the actualy subtracted dark current. It is set to 0 if no subtraction is made (treshold not reached).

Name: *curBadPixMap* **Type:** BADPIX **Format:** xyPrep
Extension: .crmbpm.fits
Description: Optional one or *Nimage* current mask(s) of bad pixels corresponding to the *Nimage* similar input frames; if only one mask is given (default) it will be used for all images.
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PRESCX	CCDDCS	integer	none	DIC: Number of pixels in X read before real data arrives.
PRESCY	CCDDCS	integer	none	DIC: Number of pixels in Y read before real data arrives.
OVRSCX	CCDDCS	integer	none	DIC: Number of pixels in X read as overscan.
OVRSCY	CCDDCS	integer	none	DIC: Number of pixels in Y read as overscan.

Other inputs

none

Output frames

Name: *averageImage* **Type:** ESTIMG **Format:** xyPrep
Extension: .crmavg.fits
Description: sigma-clipped mean or median average image of the *Nimage* input frames
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BITPIX	PR_FITS	integer	none	DIC: # of bits per pix value
AVGMETHOD	DRS	string	none	Method of average
CRMETHOD	DRS	string	none	Method of cosmic detection
SCALING	DRS	integer	none	1 - Frame scaled to common average
CRHMSIGMA	DRS	float	none	multiple of sigma (sigma-clipping)
CRHMNITER	DRS	integer	none	number of iterations (sigma-clipping)
CRHMMFRAC	DRS	float	none	min fraction of points accepted/total (sigma-clipping)
ALERTVAL	DRS	float	none	number of cosmic ray hits used as threshold for an alert

Name: *averageCrhBadPixMap* **Type:** BADPIX **Format:** xyPrep
Extension: .crmbpm.fits
Description: bad pixel map of the average image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BITPIX	PR_FITS	integer	none	DIC: # of bits per pix value
BZERO	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
BSCALE	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
DATAMIN	PR_FITS	double	e-	DIC: Minimal pixel value
DATAMAX	PR_FITS	double	e-	DIC: Maximal pixel value

Name: *sigmaImage* **Type:** ESTSIGM **Format:** xyPrep
Extension: .crmsig.fits
Description: standard error image of the *Nimage* input frames
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
DATAMIN	PR_FITS	double	e-	DIC: Minimal pixel value
DATAMAX	PR_FITS	double	e-	DIC: Maximal pixel value

Name: *crhmCount* **Type:** CRHIMG **Format:** xyPrep
Extension: .crmavg.cosmics.fits
Description: *Nimage* individual CRH count frame
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
DATAMIN	PR_FITS	double	e-	DIC: Minimal pixel value
DATAMAX	PR_FITS	double	e-	DIC: Maximal pixel value

Name: *crhBadPixMap* **Type:** BADPIX **Format:** xyPrep
Extension: .crmbpm.fits
Description: *Nimage* updated bad pixel map for each of the input frames
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
DATAMIN	PR_FITS	double	e-	DIC: Minimal pixel value
DATAMAX	PR_FITS	double	e-	DIC: Maximal pixel value

Other outputs

none

General description

This function performs the CRH detection for a set of multiple frames taken in similar conditions (same `SLIT`, `GRATING` and `WLEN0`). The detection of CRH is made simultaneously with the computation of the combined final image *averageImage* which is the output image and represents the most complete cleaned image available under the specified conditions of the computation.

First an averaged image is computed pixel by pixel from the complete set and with pixel value rejection. Two options are offered: the first one (<`crMethod`> = "DATA") is a classical sigma clipping method and applies to frames with the same exposure time only because no scaling factor is used, while the second one (<`crMethod`> = "MODEL") operates with a variance computation using the averaged value at each iteration of the sigma clipping process. In this second option scaling by the exposure time can be adopted or not. The averaging method is either the median or the mean. Depending on the number of frames (smaller than 10 or not) and the validity of the variance modelization, the median or the mean and one of the two options have to be chosen. The method which could apply with a fairly good result in a lot of cases (particularly with a small number of frames) is the use of median and the second option (ie.<`crMethod`> = "MODEL" and <`avgMethod`> = "MEDIAN").

Then each individual frames of *dbSubImage* is scaled to the averaged image *averageImage* using the valid points only. The excess count is determined as $individualCrhmCount(x,y) = scaledDbSubImage(x,y) - averageImage(x,y)$. If the excess count exceeds a threshold, it is considered as a CRH.

Mathematical description

The CRHs are detected using the data of $Nimage$ frames *dbSubImage* and their corresponding current masks of bad pixels *curBadPixMap* (a bad pixel mask 'MASTER' or only one bad pixel mask can be used for all the frames). The extraction of the $Nimage$ $badPixMask$ is done with the same list of flags <`badPixFlags`> for all the frames. Then for a frame j:

$$badPixMask_j(x,y) = curBadPixMap_j(x,y) \cap (\texttt{badPixFlag1} \cup \texttt{badPixFlag2} \cup ...)$$

In the following discussion the selection of pixel values are done only if the pixel is valid in the bad pixel mask ($badPixMask(x,y) = 0$).

For each pixel (x,y) an iterative process is applied. First the preliminary estimated value $averageImage(x,y)$ is computed from the total number $Nimage$ of frames as well as the square root of the variance $sigmaImage(x,y)$ associated to it, taking only valid pixels.

- if <`avgMethod`> = "MEDIAN", the estimate value is the mean of the two central values when the number of selected frames is even, and the central value when this number is odd (the values being sorted by increasing order).
- if <`avgMethod`> = "MEAN", the value computed is the arithmetic mean of the selected points.

Then a rejection criterion is used iteratively and the pixel value for a frame j are be kept if:
$badPixMask_j(x,y) = 0$ and
$| dbSubImage_j(x,y) - averageImage(x,y) |< \texttt{crSigma} \times sigmaImage(x,y)$
At each step new values for $averageImage(x,y)$ and $sigmaImage(x,y)$ are calculated. The sigma-clipping is repeated till no pixel value is rejected or the <`crNiter`> iterations have been reached or when two many frames have been rejected:

$$\frac{crNpAccept}{Nimage} < \texttt{crMfrac}$$

The number of retained values should never be smaller than 2. If after rejection the number of retained values is below this limit, the value with the smallest residual in absolute value is added back (in case of two values have the same residuals, they are kept). This process is repeated till the number of retained values reaches or exceeds the limit.
The detailed computations depend on the chosen method:

1. if <`crMethod`> = "DATA", then the estimate image *averageImage* is computed by attributing to each pixel the estimate value corresponding to selected frames and $sigmaImage(x,y)$ is the standard error computed for this sample:

$$sigmaImage(x,y)^2 = \left(\sum_{select.frames} (dbSubImage_j(x,y) - averageImage(x,y))^2 \right) / (select.frames.Number - 1)$$

2. if <crMethod> = "MODEL", the same calculation is done for the estimate image but the variance is computed with a noise modelization.

   - if <scaling> = FALSE, the data are not scaled and the variance is given by :

     $$var(x,y) = averageImage(x,y) + dbSubImage.\texttt{RON}^2 + dbSubImage.\texttt{DARKVALUE} \times dbSubImage.\texttt{EXPTIME}$$

   - if <scaling> = TRUE, the data are scaled by the exposure time:

     $$cmSubImage_j(x,y) = dbSubImage_j(x,y) \,/\, dbSubImage_j.\texttt{EXPTIME}$$

     and then $averageImage(x,y)$ is computed from the selected values of $cmSubImage_j(x,y)$. Finally a variance is computed for each frame j:

     $$var(x,y)_j = \frac{averageImage(x,y)}{dbSubImage_j.\texttt{EXPTIME}} + \left(\frac{dbSubImage_j.\texttt{RON}}{dbSubImage_j.\texttt{EXPTIME}}\right)^2 + \frac{dbSubImage_j.\texttt{DARKVALUE}}{dbSubImage_j.\texttt{EXPTIME}}$$

     The rejection criterion is now: $badPixMask_j(x,y) = 0$ and
     $\mid cmSubImage_j(x,y) - averageImage(x,y) \mid < \texttt{cmClipSigma} \times sigmaImage_j(x,y)$, with $sigmaImage_j(x,y) = \sqrt{var(x,y)_j}$.

At the end an estimate image $averageImage$ and a standard error image $sigmaImage$ are computed depending if a scaling has been performed:

1. if <scaling> = FALSE, this is done from the $dbSubImage$ with only the retained pixels and the averaging method "MEAN" or "MEDIAN". The standard error $sigmaImage(x,y)$ is the standard deviation of $dbSubImage_j(x,y)$ from its estimate value $averageImage(x,y)$ also for the retained pixels.
   In this case the exposure time of all the frames is assumed to be the same so the keyword EXPTIME of the output frame $averageImage$ are set equal to the one of the first frame in the set ($dbSubImage_1.\texttt{EXPTIME}$).

2. if <scaling> = TRUE, the estimate image is computed from the $cmSubImage$ set with only the retained pixels and the averaging method "MEAN" or "MEDIAN". The standard error $sigmaImage(x,y)$ is also the standard deviation of $cmSubImage_j(x,y)$ from its estimate value $averageImage(x,y)$ with the retained pixels.
   But those two images are scaled to the mean exposure time for the set calculated as:

   $$averageImage.\texttt{EXPTIME} = \sum_{frames} dbSubImage_j.\texttt{EXPTIME} \,/\, Nimage$$

   And we have:

   $$averageImage(x,y) = averageImage(x,y) \times averageImage.\texttt{EXPTIME}$$
   $$sigmaImage(x,y) = sigmaImage(x,y) \times averageImage.\texttt{EXPTIME}$$

   The keyword EXPTIME of the output frame $averageImage$ are set equal to $averageImage.\texttt{EXPTIME}$.

Then an intensity scaling is applied to each individual image $dbSubImage_j$ in order to normalize each frame to the equivalent exposure time calculated for $averageImage$. For that a mean pixel intensity is computed with the valid pixels of each image; and the scaling factor is obtained by dividing the mean pixel intensity of $averageImage$ by the mean pixel intensity of image j.
If <scaling> = FALSE, the sigma-clipping process is repeated after substituting $scaledDbSubImage_j$ to $dbSubImage_j$. At the end a new intensity scaling is done for each image $dbSubImage_j$ but with the final set of valid pixels.

The scaled excess count linked with CRH represents the excess count which should occur for the equivalent exposure time of $averageImage$ and is determined as:

$$individualCrhmCount_j(x,y) = scaledDbSubImage_j(x,y) - averageImage(x,y)$$

Quality Assessment:
The $individualCrhmCount(x,y)$ frame is used to compute a frame with the total number of electrons/s due to CRH for each $dbSubImage_j$:

$$crhmPhot_j = \left(\sum_{x,y} individualCrhmCount(x,y)_j\right) \,/\, averageImage.\texttt{EXPTIME}$$

If the total number $\texttt{crhmPhot}_j$ for a given $dbSubImage_j$ is larger than the CRH alert value <alertVal> , a warning message is printed out.

2.2.8 giRecCosmicMC.py

Python implementation description

```
(Following description could be obtained on-line using: giRecCosmicMC.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]

   NAME
   giRecCosmicMC.py - CRH detection using several frames

   SYNOPSIS
           giRecCosmicMC.py [-h|--help]
   or
           giRecCosmicMC.py [options] dbsubFrameList [badPixmapList]

   DESCRIPTION
   detect CRH in multiple frames taken in the same conditions
   where:
          <dbsubFrameList> - comma separated list of N frame(s) to process with N >= 3
          <badPixmapList>  - comma separated list of corresponding bad pixel map(s) to process
                             one for each frame or one for all

   valid options are:
         -h|--help                display help and exit
         -d|--debug               set debug messages
         -v|--verbose             set verbose processing
         -q|--quiet               no messages
         -t|--time                show processing time
         -z|--defaults            show default values
         --params                 dump INI config parameters
         -w|--ovwrite             overwrite product files
         --novwrite               do not overwrite product files
         --[no]compress           [do not] compress product files
         --outdir <dir>           output directory for products
         --datadir <dir>          specify input data directory
         --crmethod <crmethod>    noise estimation: 'DATA' or 'MODEL'
         --avgmethod <avgmethod>  averaging estimation: 'MEAN' or 'MEDIAN'
         --[no]scaling            [do not] scale data
         --[no]bpmap              [do not] output individual bad pixel map(s)
         --crhron <ron>           new bias sigma (RON) value for dbsubframe
         --[no]crhcount           [do not] output individual CRH count frame(s)
         --crsigma <sigma>        crh-clipping: multiple of sigma
         --crniter <niter>        crh-clipping: number of iterations
         --crmfrac <mfrac>        crh-clipping: min fraction of points
                                  accepted/total. Should be in range [0.0-1.0]
         --alertval <value>       CRH alert value

   OUTPUTS
      - frame1.crmavg.fits:      average CRH cleaned frame
      - frame1.crmbpm.fits:      average CRH cleaned bap pixel map
      - frame1.crmsig.fits:      average CRH cleaned sigma frame
      - frame<n>.crmicnt.fits:   individual CRH count frame (n in [0-N[)
      - frame<n>.crmibpm.fits:   individual CRH bap pixel map (n in [0-N[)

   EXAMPLES
   Display all defaults values and exits

      giRecCosmicMC.py -z

   Verbose run with a median average method and 'bpmMaster.fits'
   on frames 'dark0.fits', 'dark1.fits' and 'dark2.fits'

      giRecCosmicMC.py -v --avgmethod=MEDIAN  dark0.fits,dark1.fits,dark2.fits bpmMaster.fits

   ENVIRONMENT VARIABLES
     PYTHONSTARTUP  - python startup script
```

```
    GIR_PYTHON     - GIRAFFE BLDRS python modules directory
    GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
    GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 22480 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecCosmicMC.py
(Following default values could be obtained on-line using: giRecCosmicMC.py -z )
    Reading BLDRS parameters from girBLDRS.ini:[General]
    bSubFiles : dark0.fits
                dark1.fits
                dark2.fits
    avgmethod : MEDIAN
    crmethod  : DATA
    scaling   : 0
    crsigma   : 8.0
    crniter   : 4
    crmfrac   : 0.7
    bpmap     : 0
    crhcount  : 0
    alertval  : 10.0
    oavgFile  = dark0.crmavg.fits
    obpmFile  = dark0.crmbpm.fits
    osigFile  = dark0.crmsig.fits
    verbose   : 1
    debug     : 0
    showtime  : 0
     ovwrite  : 0
     datadir  : None
     outdir   : None
     cfgdir   : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
```

2.2.9 giAdjustSL - Adjust the scattered light

Name: **giAdjustSL** Section: **Preprocessing**
Python name: **giRecSLfit.py**

Purpose

Adjust a model of scattered light to the inter-spectra region.

Function parameters

Modelization control parameters

Name	Type	Description
slPhffCor	logical	option for correcting the input frame from the photometric flat field
slModel	string	option indicating if the model is a polynomial ("POLYNOM") or a polynomial fraction ("POLYFRAC")
slPolyDeg1X	integer	degree of the numerator polynomial of the model
slPolyDeg2X	integer	degree of the denominator polynomial of the model
slPolyDeg1Y	integer	degree of the numerator polynomial of the model
slPolyDeg2Y	integer	degree of the denominator polynomial of the model
contamW	integer	extra width parameter (in pixels) to cut out the part of inter-spectra region used for SL determination

Input frames

Name: *prepImage* **Type:** PREPIMG **Format:** xyPrep
Extension: dbSub.fits
Description: any cleaned, bias-and-dark subtracted image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NX	DRS	integer	none	Size of image in X direction (lambda)
NY	DRS	integer	none	Size of image in Y direction (slit)
STARTX	CCDDCS	integer	none	DIC: First window pixel in X direction within the detector physical system.
STARTY	CCDDCS	integer	none	DIC: First window pixel in Y direction within the detector physical system.
EXPTIME	PR_FITS	double	s	DIC: Integration time
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *locYMast* **Type:** LOCY **Format:** nxExt
Extension: .clocy.fits
Description: standard localization giving the spectrum center along y at each x for the current setup
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
EXTNX	DRS	integer	none	Number of pixel per spectrum
EXTNS	DRS	integer	none	Number of extracted spectra on the rebinned S2dFrame
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *locWyMast* **Type:** `LOCWY` **Format:** `nxExt`
Extension: `.clocw.fits`
Description: standard width of the NFF spectra (measured along the y axis) for the current setup
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
EXTNX	DRS	integer	none	Number of pixel per spectrum
EXTNS	DRS	integer	none	Number of extracted spectra on the rebinned S2dFrame
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *phffImageMast* **Type:** `PHFFIMG` **Format:** `undef`
Extension:
Description: Optional master photometric flat-field calibration image for the current setup. Its use is optional
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NX	DRS	integer	none	Size of image in X direction (lambda)
NY	DRS	integer	none	Size of image in Y direction (slit)
STARTX	CCDDCS	integer	none	DIC: First window pixel in X direction within the detector physical system.
STARTY	CCDDCS	integer	none	DIC: First window pixel in Y direction within the detector physical system.
EXPTIME	PR_FITS	double	s	DIC: Integration time
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *badPixMap* **Type:** `BADPIX` **Format:** `xyPrep`
Extension: `.crmbpm.fits`
Description: Optional current *repPixImage* bad pixel map
Relevant FITS keywords:

Name	DIC	Type	Unit	Description

Other inputs

none

Output frames

Name: *slImage*	**Type:** SLIMG	**Format:** xyPrep
	Extension: .slimg.fits	

Description: image of scattered light

Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLPHFFCOR	DRS	logical	none	option for correcting input frame frome photometric flat-field
SLMODEL	DRS	string	none	option indicating the model of scattered light: 'POLYNOM' or 'POLYFRAC'
SLPOLYDEG1	DRS	integer	none	degree of the numerator polynom of the SL model
SLPOLYDEG2	DRS	integer	none	degree of the denominator polynom of the SL model

Other outputs

none

General description

The inter-spectra pixels of the bias and dark subtracted image are used to model the contribution of scattered light on the whole image.

The model of scattered light (defined by <slModel>) is fitted on the inter-spectra regions of *prepImage*. The inter-spectra regions are determined by the localization mask using $locYMast$ and $locWyMast$ which are chosen according to the keywords *prepImage*.SLIT, *prepImage*.GRATING and *prepImage*.WLENO of the input frame. A width parameter <contamW> (additive to $locWyMast$) allows to shrink the interspectra region used in the estimation of the scattered light and then avoid the contamination by object light. This parameter eliminates almost all the interspectra regions in an Argus/IFU frame, except the broader interslitlets regions.

As an option (depending on <slPhffCor>), the bias and dark subtracted image can be corrected by a photometric flat-field frame ($phffImageMast$) before modeling the scattered light.

Mathematical description

The model of scattered light, depending on <slModel> , is either a polynomial $\sum_{i,j=0}^{\texttt{slPolyDeg1X,slPolyDeg1Y}} c_{ij} T_i(x') T_j(y)$ or a polynomial fraction $\frac{\sum_{i,j=0}^{\texttt{slPolyDeg1X,slPolyDeg1Y}} a_{ij} T_i(x') T_j(y)}{\sum_{m,n=0}^{\texttt{slPolyDeg2X,slPolyDeg2Y}} b_{mn} T_m(x) T_n(y)}$.

Degrees of numerator and denominator are respectively defined by <slPolyDeg1X> , <slPolyDeg1Y> and <slPolyDeg2X> , <slPolyDeg2Y> (<slPolyDeg2X> and <slPolyDeg2Y> are zero in case of a polynomial model). Models are both represented using Chebyshev polynomials T_i defined on the normalized coordinates x' and y'.

A LSQ fit of the model is carried out on the inter-spectra region $prepImage(x_{\rm is}, y_{\rm is})$ (optionally divided by $phffImageMast(x_{\rm is}, y_{\rm is})$), where $x_{\rm is}$ and $y_{\rm is}$ stand for coordinates of inter-spectra pixels ($locYMast(x_{\rm is}, n) + locWyMast(x_{\rm is}, n) + contamW < y_{\rm is} < locYMast(x_{\rm is}, n+1) - locWyMast(x_{\rm is}, n+1) - contamW$).

Then the full image of the scattered light *slImage* is constructed from the coefficients of the model with, optionally, the transform back to the unflatfielded image. There is no bad pixel mask for the scattered light image which is supposed defined (positive and lower than the linearity limit) for every pixels.

2.2.10 giRecSLfit.py

Python implementation description

```
(Following description could be obtained on-line using: giRecSLfit.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]

   NAME
```

```
giRecSLfit.py - scattered light correction recipe.

SYNOPSIS
  giRecSLfit.py [-h|--help]
or
  giRecSLfit.py [options] [dbsubFrame [locyFrame [locwyFrame [phffMFrame]]]]

DESCRIPTION
Run giAdjustSL() on the
specified dark and bias corrected frame <dbsubFrame> :
where
   <locyFrame>      - locy frame filename,
                          default 'dbsubFrame.locy.fits'
   <locwyFrame>     - locwy frame filename
                          default 'dbsubFrame.locw.fits'
   <phffMFrame>     - master photometric flat-field frame
                          default none

valid options are:
  -h|--help                display help and exit
  -d|--debug               set debug mode
  -v|--verbose             set verbosity level,
                           repeated occurence increase verbosity
  -t|--time                show processing time
  -q|--quiet               quiet mode
  -z|--defaults            show current assignement of parameters values
                           (default + specified on command line0
  -b|--bpmap <bpixmap>     specify bad pixel map filename
  --params                 display INITIAL config parameters
  --ovwrite                overwrite product files
  --novwrite               do not overwrite product files
  --remove                 remove fitted scattered light from input frame
  --outdir <dir>           specify product directory
  --datadir <dir>          specify input data directory
  --slmodel <model>        Fitted model: 'POLYNOM' or 'POLYFRAC'
  --xorder <order>         Fitted polynom order along X (dispersion)
  --yorder <order>         Fitted polynom order along Y
  --xstep <step>           sampling step along X (every 'xstep' pixel taken)
  --ystep <step>           sampling step along Y (every 'ystep' pixel in
                           inter-spectra region is taken)
                           use rather --ewidth option to reduce the
   number of used pixels
  --xslice <slices>        sampling along X; overrides the --xstep option in a
                           specific region.
                           example: --xstep 50 --xslice 3000:4000:10 will sample by
                             step of 10 between 3000 and 4000 while elsewhere
                             the step 50 is used.
  --ewidth <pix>           extra width added to both sides of loc. width [pixels]
  --iswidth <pix>          minimum width required for IS region [pixels]
  --[no]trim               do [not] use first and last IS region
  --slphcor                with photometric FF correction
  --noslphcor              no photometric FF correction

OUTPUTS
resulting extraction goes to slFile = '<rawFrame>.slimg.fits'

EXAMPLES
Display all defaults values and exits:

  giRecSLfit.py -z

Verbose run with a POLYNOM model on frame 'flat137_1.bsub.fits':

  giRecSLfit.py -v --slmodel='POLYNOM'  flat137_1.bsub.fits
```

```
  will create 'flat137_1.bsub.slimg.fits'

  ENVIRONMENT VARIABLES
    PYTHONSTARTUP  - python startup script
    GIR_PYTHON     - GIRAFFE BLDRS python modules directory
    GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
    GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 19184 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecSLfit.py
(Following default values could be obtained on-line using: giRecSLfit.py -z )
    Reading BLDRS parameters from girBLDRS.ini:[General]
    $Id: giRecSLfit.py,v 1.7 2003/07/20 09:52:26 anonymous Exp $
     dbsubFile= /giraffe/blecha/BLDRS/girBLDRS1.09/data/dummy.fits
     locyFile = /giraffe/blecha/BLDRS/girBLDRS1.09/data/dummy.clocy.fits
     locwFile = /giraffe/blecha/BLDRS/girBLDRS1.09/data/dummy.clocw.fits
     phffFile = -
     bpmap    : -
     slmodel  : POLYNOM
     xorder   : 3
     yorder   : 3
     xstep    : 10
     ystep    : 1
     ewidth   : 0.5
     iswidth  : 2
     trim     : 1
     slphcor  : 0
     remove   : 0
     slFile    = /giraffe/blecha/BLDRS/girBLDRS1.09/data/dummy.slimg.fits
     slchebFile= /giraffe/blecha/BLDRS/girBLDRS1.09/data/dummy.slcheb.tfits
     verbose  : 1
     debug    : 0
     time     : 1
     ovwrite  : 0
     datadir  : None
     outdir   : None
     cfgdir   : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
```

2.2.11 giLocalSpectra - Localization of spectra

Name: **giLocalSpectra** Section: **Extraction**
Python name: **giRecNLoc2.py**

Purpose

Determine the centroid and the width of each spectral bin of each spectrum using (or not) the preliminary localization. A new current localization is defined through the polynomial fit of the widths and centroids found.

Function parameters

Name	Type	Description
lClipSigma	float	multiple of σ (σ-clipping), locSigClip[1]
lClipNiter	integer	number of iterations (σ-clipping), locSigClip[2]
lClipMfrac	float	min fraction of points accepted/total (σ-clipping), locSigClip[3]
lNoiseMult	double	noise detector multiple
locYDeg	integer	degree of 1-D (X) polynomial fit of the localization ridge
locWDeg	integer	degree of 2-D (X,Y) polynomial fit of the width of the localization mask
lExtraWid	double	additional width to add on both side of the fitted localization mask
lbstart	integer	x position where we start the localisation (-1 for center)
lbtries	integer	localization detection xbin retries
lywidth	integer	full width of the equalizing filter in the direction perpendicular to the dispersion
norm	logical	normalize spectra along dispersion axis
lcentroid	logical	use barycenter for mask center
lhwidth	logical	use half-sum for mask center

Input frames

Name: *dbSubFrame* **Type:** BDRMIMG **Format:** xy
Extension: dbSub.fits
Description: any bias-dark-subtracted image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BIASSIGMA	DRS	float	ADU	Computed bias sigma in pre/over-scan
NFIBRES	DRS	integer	none	Number of fibres for the slit
CONAD	CCDDCS	double	e-/ADU	DIC: The conversion factor which translates ADU to photonic electrons.

Name: *badPixMap* **Type:** BADPIX **Format:** xyPrep
Extension: .crmbpm.fits
Description: Optional current *dbSubImage* bad pixel map
Relevant FITS keywords:

Name	DIC	Type	Unit	Description

Other inputs

For standard localization the Narrow Flat Field frame is used. none

Output frames

Name: *locY* **Type:** LOCY **Format:** nxExt
Extension: .clocy.fits
Description: current localization giving the spectrum center along y at each x
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NAXIS1	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
NAXIS2	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
LOCNX	DRS	integer	pixel	pixels in dispersion direction
LOCNS	DRS	integer	none	number of spectra (135 Medusa, 316 IFU and ARGUS)
BITPIX	PR_FITS	integer	none	DIC: # of bits per pix value
BZERO	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
BSCALE	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
LMETHOD	DRS	string	none	Method of localisation, one of BARYCENTER or HALFWIDT
LNORMALIZE	DRS	integer	pixel	Dispersion direction normalisation filter
LFULLLOC	DRS	integer	none	1 for localisation using all spectra, 0 on 5 SIMCAL spectra only
LOCYDEG	DRS	integer	none	degree of the x-polynomial fit of localisation
LOCWDEG	DRS	integer	none	degree of the x,y-polynomial fit of localisation half-width
LEXTRAWID	DRS	float	pixel	value added to to preliminary localisation
LNOISEMULT	DRS	float	none	noise detector in preliminary localisation
LNOISETHRH	DRS	float	none	noise threshold for preliminary localisation
LCLIPSIGMA	DRS	float	none	multiple of sigma (sigma-clipping)
LCLIPNITER	DRS	integer	none	number of iterations (sigma-clipping)
LCLIPMFRAC	DRS	float	none	min fraction of points accepted/total (sigma-clipping)
CREATOR	DRS	string	none	Source of data (responsible or processing recipe)

Name: *locWy* **Type:** LOCWY **Format:** nxExt
Extension: .clocw.fits
Description: current width of the NFF spectra (measured along the y axis)
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NAXIS1	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
NAXIS2	PR_FITS	integer	none	DIC: # of pixels in <axis direction>

Other outputs

Name	Type	Description
chebY	girLoccTable	Polynomial models for localisation (1D in x - one model per spectrum) and localisation half-width (2-D one global model)

Extension: .clocc.tfist
Relevant FITS keywords: GIRTTYPE GRATING WLENO LPOLYDEG LOCYDEG LOCWDEG LOCWCOEFF
Relevant table columns:

NSPEC	integer	running number of the spectrum/fibre from 1 to 135(320)
COEFFi	double	Chebyshev polynomial coefficients for localisation Y position

General description

This function performs the localization of the NFF spectra on a **flat-field exposure**, which provides the so-called **master localization**, summarized in the $locYMast$ and $locWyMast$ images. This processing is carried on following two steps:

- spectra are localized from scratch, above a given threshold using local windowing and normalization, from which are derived the limits of the spectra and the corresponding centers and widths,
- the parameters of the derived localization mask (center and half-width) are fitted by a low degree polynomial expression (1D fit per spectrum for centers, 2D fit over all spectra for widths) all accross the CCD frame.

The localization is done from scratch only on a NFF frame. The spectra positions are defined by the barycenters (a method which is independent from the shape of the PSF). The widths are defined by the threshold method (as half width of the "canal" defined by fluxes higher than the threshold). The half widths so defined are used later by the *giExtractSpectra*() function in its simplest option (`eMethod` ="SUMM").

Note that the width can be reliably determined only for NFF spectra, by contrast with calibration ThAr spectra, where the discontinuous nature of emission lines raises problems.

Mathematical description

The localization on calibration NFF frames proceed as follows.

- if the parameter <`norm`> is set to `TRUE`, a rough normalization is carried out prior to the localization. This normalization is aimed at removing the global intensity profile of the spectra in the dispersion direction. This intensity profile is computed by summation of the whole frame along the direction perpendicular to the dispersion:

$$shape(x) = \sum_y dbSubImage(x, y),$$

  and then the localization is hereafter carried out on the normalized frame:

$$dbSubImage'(x, y) = dbSubImage(x, y)/shape(x).$$

- The threshold Trh, above which the spectra is detected, is defined depending on the parameter <`lNoiseMult`> .
  - If `lNoiseMult` > 0, the threshold is a multiple of the bias noise

$$Trh = \texttt{lNoiseMult} \times dbSubImage.\texttt{BIASSIGMA}$$

  - If `lNoiseMult` < 0, the threshold is a fraction of the average value $Zmean$ of the frame.

$$Trh = |\texttt{lNoiseMult}| \times Zmean \times Ns \times \texttt{lmwidth}/NAXIS2.$$

    The factor $Ns \times$ `lmwidth`$/NAXIS2$ is aimed at making the threshold independent from the number of spectra; this correction is mandatory because for different reasons all the fibers are not necessarily used. The value <`lmwidth`> is 9.3 pixels in case of MEDUSA fibers, and 5.8 for IFU and ARGUS.
- Using the threshold Trh, pixels whose intensity is lower than this value are set to 0.
  - If the parameter <`lywidth`> is used, the thresholding method includes a filtering in the direction perpendicular to the dispersion. <`lywidth`> defines the width (in pixels) of a window around each point, perpendicular to the dispersion. Instead of comparing the intensity of the pixel to the threshold, the difference of the intensity and the minimum value in the window is used. This filtering is useful when cross-talk occurs (IFU and ARGUS modes) and when the signal level differs from one slitlet to another.

For each x, each set of consecutive non-zero pixels gives the external limits of the border-pixels. If $y_{\max}$ and $y_{\min}$ are the respective borders of the set for the spectrum n:

$$upperborder(x, n) = y_{\max} + \frac{1}{2}$$

$$lowerborder(x, n) = y_{\min} - \frac{1}{2}$$

The first x for which the localization is done gives the expected number of spectra in the frame. This position is the input parameter <`locXstart`> . Usually it corresponds to the center of the x-axis, and the localization is carried out from the center to the edges.

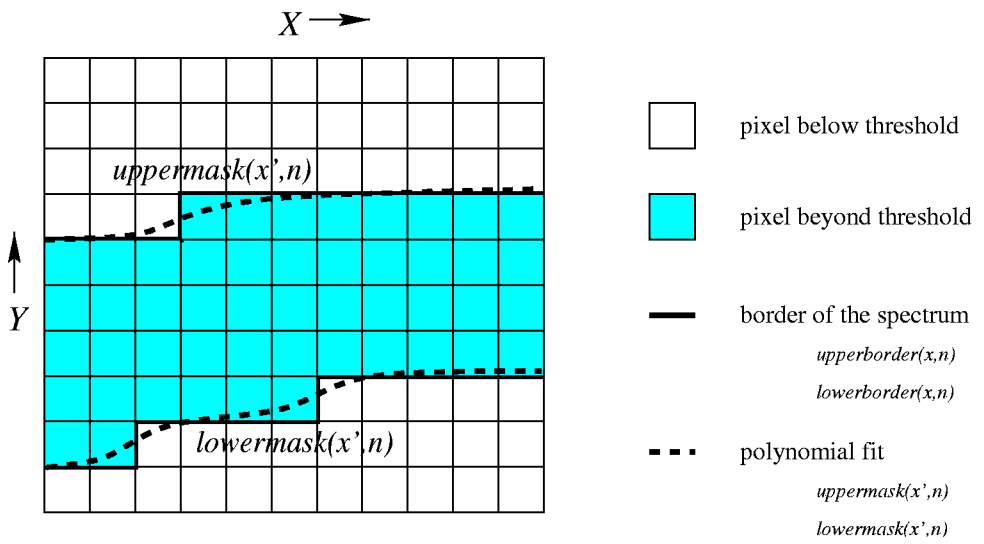


Figure 2.2: Preliminary localization for the determination of $locYMaster$ and $locWyMaster$

For various reasons (e.g. cosmic ray hits), the derived limits at a given x position may lead to a spurious number of spectra. The parameter <`lbtries`> indicates the allowed number of x positions that can be discarded when leading to a different number of spectra.

In order to obtain a smooth localization, *upperborder* and *lowerborder* are fitted with polynomial expressions (resp. $uppermask(x,n)$ and $lowermask(x,n)$), of degree <`locYDeg`> , using the retained points. The LSQ fit is carried out using Chebyshev polynomials so that *uppermask* and *lowermask* are written as :

$$uppermask(x',n) = \sum_{i=0}^{\texttt{locYDeg}} p_i T_i(x')$$

$$lowermask(x',n) = \sum_{i=0}^{\texttt{locYDeg}} q_i T_i(x')$$

where T_i are Chebyshev polynomials and x' is the normalized x-coordinates ($\in [-1,+1]$).

The y positions for each x of the spectra are computed either as centroids (<`lcentroid`> = `TRUE`) or as half sum of limits (<`lhwidth`> = `TRUE`); these positions are fitted with polynomials of degree <`locYDeg`> and stored in the frame $locYMaster(x,n)$.

In order to embrace the maximum flux, an extra-width represented by the parameter <`lExtraWid`> is added to the half-width of the fitted mask and $locWyMaster(x,n)$ is set to:

$$locWyMaster(x,n) = \frac{uppermask(x',n) - lowermask(x',n)}{2} + \texttt{lExtraWid}.$$

The widths are fitted accross the whole frame (over all the spectra) by a 2D polynomial expression, whose degree is given by `locWDeg` × `locWDeg`.

It is worth noticing that all the polynomial fits are carried out using a sigma-clipping algorithm whose parameters <`lClipSigma`> , <`lClipNiter`> and <`lClipMfrac`> are given in input.

2.2.12 giRecNLoc2.py

Python implementation description

```
(Following description could be obtained on-line using: giRecNLoc2.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]
Reading module parameters from girBLDRS.ini:[Extraction Parameters]

   NAME
   giRecNLoc2.py - localization recipe for bias and dark corrected frame.

   SYNOPSIS
```

```
  giRecNLoc2.py [-h|--help]
or
  giRecNLoc2.py [options] [dbsubFrame [mlocyFrame [mlocwFrame
                          [psfprmsTable] ] ] ]

DESCRIPTION
Run giLocalSpectra2() on the specified dark and bias corrected frame
<dbsubFrame> where:
       <mlocyFrame>     - master locy frame filename, '-' for none
       <mlocwFrame>     - master locw frame filename, '-' for none
       <psfprmsTable>   - master PSF params table, '-' for none

valid options are:
      -h|--help                display help and exit
      -d|--debug               debug mode
      -v|--verbose             increase verbosity level
                               repeated occurence increase verbosity
      -q|--quiet               no messages
      -t|--time                show processing time
      -g|--graphs              display graphics
      --nographs               do not build any graphics
      -z|--defaults            show default values
      -b|--bpmap <bpixFile>    specify bad pixel map filename
                               '-' for none
      --params                 dump INI config parameters
      -w|--ovwrite             overwrite product files
      --novwrite               do not overwrite product files
      --outdir <dir>           output directory for products
      --datadir <dir>          specify input data directory
      --splist <list>          get localized spectra number from list:
                                for example: --splist 2,10,30:40:2,55
                                will assign spectra those numbers:
                                   2,10,30,32,34,36,38,40,55
      --lbstart <bstart>       initial localization detection xbin
                               default to -1 (middle of X CCD)
      --lbtries <btries>       localization detection xbin retries
      --lewidth <ewidth>       localization additional width;
                       recommended: Medusa 1.5, IFU 0.5
      --lywidth <ywidth>       full width [px] of the equalizing filter in
                       the direction -| to the dispersion;
recommended: Medusa 14, IFU 4-6
 (distance between 2 adjacent spectra)
      --lcentroid              use barycenter for mask center
      --lhwidth                use half-width for mask center
      --norm                   normalize spectra along dispersion axis
                       before the localisation; this the DEFAULT
      --nonorm                 do not normalize spectra along dispersion axis
      --lnoise <noisemul>      noise factor for threshold
                               noisemul > 0 multiple of BIAS noise
                               noisemul < 0 fraction of the average value of
the frame. Note that since ~ half of CCD is
filled with spectra, noisemult=-1 will use a
treshold of approximately 50% of average value
                               This option recommended with '--norm'
      --lron <ron>             new bias sigma (RON) value for dbsubframe
      --yorder <deg>           order of Chebyshev polnomial fit;
                       recommended: 2-3
      --worder <deg>           order of Chebyshev 2D polnomial fit;
                       recommended: 2
      --lsigma <sigma>         loc-clipping: multiple of sigma
      --lniter <niter>         loc-clipping: number of iterations
      --lmfrac <mfrac>         loc-clipping: min fraction of points
                               accepted/total. Should be in range [0.0-1.0]
      -N                       number of spectra to be localized
```

```
COOK-BOOK

  Main tuning parameters are:      lnoise, lewidth, yorder and worder lywidth
  We recommended for Medusa/IFU        -1    1.5/1     2         2       6/4

  If the spectra curvature is not symetrical ont the localisation plot,
  <yorder> and <worder> should be incrased gently. In COM2 adjustement yorder=4
  and worder=4 should be enough.

  If the 'green' inter-spectra lanes is reduced to single line int the
  localization plots, reduce the value of <lewidth> which governs the extra
  width added to the above-threshold width.

OUTPUTS
resulting localization goes to <rawFrame>.clocy.fits <rawFrame>.clocw.fits
and <rawFrame>.clocc.tfits

EXAMPLES
Display all defaults values and exits
  giRecNLoc2.py -z
Verbose run with a noise factor of 19.0 and a Chebyshev fit order of 15
on frame 'flat137_1.bsub.fits'
  giRecNLoc2.py -v --lnoise=19.0 --yorder=15 flat137_1.bsub.fits

ENVIRONMENT VARIABLES
  PYTHONSTARTUP  - python startup script
  GIR_PYTHON     - GIRAFFE BLDRS python modules directory
  GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
  GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 24201 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecNLoc2.py
(Following default values could be obtained on-line using: giRecNLoc2.py -z )
   Reading BLDRS parameters from girBLDRS.ini:[General]
   Reading module parameters from girBLDRS.ini:[Extraction Parameters]
   $Id: giRecNLoc2.py,v 1.3 2003/04/02 12:52:45 anonymous Exp $
    dbsubFile  = /giraffe/blecha/BLDRS/girBLDRS1.09/data/FfffMedPinH13Seq56dbSub.fits
    mlocyFile  = -
    mlocwFile  = -
    psfPrmFile = -
    bpixFile   : -
    lnoise     : 17.0
    yorder     : 3
    worder     : 3
    lsigma     : 2.5
    lniter     : 2
    lmfrac     : 0.9
    lcentroid  : 1
    lewidth    : 0.5
    lywidth    : 14
    snorm      : 1
    lbstart    : -1
    lbtries    : 10
    clocyFile  = /giraffe/blecha/BLDRS/girBLDRS1.09/data/FfffMedPinH13Seq56dbSub.clocy.fits
    clocwFile  = /giraffe/blecha/BLDRS/girBLDRS1.09/data/FfffMedPinH13Seq56dbSub.clocw.fits
    cloccFile  = /giraffe/blecha/BLDRS/girBLDRS1.09/data/FfffMedPinH13Seq56dbSub.clocc.tfits
    verbose    : 1
    debug      : 0
    showtime   : 1
    graphics   : 0
    ovwrite    : 0
    datadir    : None
    outdir     : None
```

```
cfgdir     : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
```

2.2.13 giAdjustLoc - Adjust localization

Name: **giAdjustLoc** Section: **Extraction**
Python name: **giRecDLocfit.py**

Purpose

Adjust master localization for current frame using the simultaneous ThAr calibration.

Function parameters

Name	Type	Description
XWidth	float	x-size of the search window
YWidth	float	y-size of the search window
dNiter	integer	number of iterations (σ-clipping), locSigClip[2]
dMfrac	float	min fraction of points accepted/total (σ-clipping), locSigClip[3]
dSigma	double	noise detector multiple
XOrder	integer	x-degree of 2-D polynomial fit to localization adjustement
YOrder	integer	y-degree of 2-D polynomial fit to localization adjustement

Input frames

Name: *dbSubFrame* **Type:** BDRMIMG **Format:** xy
Extension: dbSub.fits
Description: any bias-dark-subtracted image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *locY* **Type:** LOCY **Format:** nxExt
Extension: .clocy.fits
Description: master localisation
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BIASSIGMA	DRS	float	ADU	Computed bias sigma in pre/over-scan
NFIBRES	DRS	integer	none	Number of fibres for the slit
CONAD	CCDDCS	double	e-/ADU	DIC: The conversion factor which translates ADU to photonic electrons.

Name: *locWy* **Type:** LOCWY **Format:** nxExt
Extension: .clocw.fits
Description: master localisation half-width
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BIASSIGMA	DRS	float	ADU	Computed bias sigma in pre/over-scan
NFIBRES	DRS	integer	none	Number of fibres for the slit
CONAD	CCDDCS	double	e-/ADU	DIC: The conversion factor which translates ADU to photonic electrons.

Other inputs

Name	Type	Description
girSlitGeo	girSlitGeoTable	Geometrical model of the slit
Extension: girSlitGeoXXX.tfits		
Relevant FITS keywords: GIRTTYPE SLIT GRATING WLEN0		
Relevant table columns:		
NSPEC	integer	running number of the spectrum/fibre from 1 to 135(320)
XF	float	Almost constant fiber position in x (perpendicular to the slit)
YF	float	Almost regularly decreasing fiber position in y (along the slit)
RP	integer	running number of positionner retractor
girWavCoef	girWavCoefTable	Kwords give the optical solution while table columns provide coefficients of the Chebytchev correction
Extension: girWavCoefXXX.tfits		
Relevant FITS keywords: SLIT GRATING WLEN0 WSWSOLUTION WSWMODEL WSWDIRECTION WSWSUBSLITFIT WSWFCOLL WSWGCAM WSWTHETA WSWSLITDX WSWSLITDY WSWSLITPHI WSWLINEFILE WSWLINEMODEL WSWLINETYPE WSWLINE WSWLINETHRESHOLD WSWLINENITER WSWLINENTEST WSWLINEDCHISQ		
Relevant table columns:		
CIJ	double	Coefficients of the wavelength solution (for the residuals relative to the optical model)
linesWlampMast	girLineTable	wavelengths and fluxes of the calibration spectrum
Extension: girLineXXXX.tfits		
Relevant FITS keywords: GIRTTYPE LAMP LAMPID		
Relevant table columns:		
WLEN	float	Wavelength of the line
NAME	string	Name of line
FLUX	float	Expected central flux/[pixel.sec]
COMMENT	string	Setup followed by cotrolled vocabulary comment
girWres	girWresTable	Optional wavelength spectra shift obtained from 5 SIMCAL spectra
Extension: .wlenres.tfits		
Relevant FITS keywords: GIRTTYPE		
Relevant table columns:		
WLRES	float	Wavelength shift of the corresponding spectrum

Output frames

Name: *dLocY* **Type:** DLOCY **Format:** nxExt
Extension: .dlocy.fits
Description: Corrected localization (master+LocD)
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NAXIS1	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
NAXIS2	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
LOCNX	DRS	integer	pixel	pixels in dispersion direction
LOCNS	DRS	integer	none	number of spectra (135 Medusa, 316 IFU and ARGUS)
BITPIX	PR_FITS	integer	none	DIC: # of bits per pix value
BZERO	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
BSCALE	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
LMETHOD	DRS	string	none	Method of localisation, one of BARYCENTER or HALFWIDT
LNORMALIZE	DRS	integer	pixel	Dispersion direction normalisation filter
LFULLLOC	DRS	integer	none	1 for localisation using all spectra, 0 on 5 SIMCAL spectra only
LOCYDEG	DRS	integer	none	degree of the x-polynomial fit of localisation
LOCWDEG	DRS	integer	none	degree of the x,y-polynomial fit of localisation half-width
LEXTRAWID	DRS	float	pixel	value added to to preliminary localisation
LNOISEMULT	DRS	float	none	noise detector in preliminary localisation
LNOISETHRH	DRS	float	none	noise threshold for preliminary localisation
LCLIPSIGMA	DRS	float	none	multiple of sigma (sigma-clipping)
LCLIPNITER	DRS	integer	none	number of iterations (sigma-clipping)
LCLIPMFRAC	DRS	float	none	min fraction of points accepted/total (sigma-clipping)
LCLIPMFRAC	DRS	float	none	min fraction of points accepted/total (sigma-clipping)
CREATOR	DRS	string	none	Source of data (responsible or processing recipe)

Name: *LocD* **Type:** LOCD **Format:** nxExt
Extension: .locd.fits
Description: correction added to the master localization
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NAXIS1	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
NAXIS2	PR_FITS	integer	none	DIC: # of pixels in <axis direction>
LOCNX	DRS	integer	pixel	pixels in dispersion direction
LOCNS	DRS	integer	none	number of spectra (135 Medusa, 316 IFU and ARGUS)
BITPIX	PR_FITS	integer	none	DIC: # of bits per pix value
BZERO	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
BSCALE	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
LMETHOD	DRS	string	none	Method of localisation, one of BARYCENTER or HALFWIDT
LNORMALIZE	DRS	integer	pixel	Dispersion direction normalisation filter
LFULLLOC	DRS	integer	none	1 for localisation using all spectra, 0 on 5 SIMCAL spectra only
LOCYDEG	DRS	integer	none	degree of the x-polynomial fit of localisation
LOCWDEG	DRS	integer	none	degree of the x,y-polynomial fit of localisation halfwidth
LEXTRAWID	DRS	float	pixel	value added to to preliminary localisation
LNOISEMULT	DRS	float	none	noise detector in preliminary localisation
LNOISETHRH	DRS	float	none	noise threshold for preliminary localisation
LCLIPSIGMA	DRS	float	none	multiple of sigma (sigma-clipping)
LCLIPNITER	DRS	integer	none	number of iterations (sigma-clipping)
LCLIPMFRAC	DRS	float	none	min fraction of points accepted/total (sigma-clipping)
LCLIPMFRAC	DRS	float	none	min fraction of points accepted/total (sigma-clipping)
CREATOR	DRS	string	none	Source of data (responsible or processing recipe)

Other outputs

Name	Type	Description
chebDY	girDLoccTable	Polynomial models for localisation correction (1D in x - one model per spectrum)

Extension: .chebd.tfits
Relevant FITS keywords: GIRTTYPE GRATING WLEN0 DLPOLYDEG DLOCXYWID
Relevant table columns:

NSPEC	integer	running number of the spectrum/fibre from 1 to 135(320)
COEFFi	double	Chebyshev polynomial coefficients for the correction of localisation

General description

This function allows the adjustment of the master localization by using the spectral lines of the five SIWC spectra on a **science frame**. The shift in position of these lines (perpendicularly to the dispersion) compared to the master localization provides a correction to the latter, followed by an **adjustment**. This adjustment is needed because of mechanical and optical variations due to temperature variations or other causes; it allows the correct localization of all spectra of the science frame.

The widths of the spectra are not adjusted because SIWC spectra are not very appropriate for this task. The master widths, derived from PSF fitting, are the best approximations.

Mathematical description

Using the five simultaneous wavelength calibration spectra on a science frame, the centroids of the emission lines from the Th-Ar-Ne lamp spectra can be computed.

For each of the five SIWC spectra, the emission lines listed in the input table *girLineThAr* are localized in the frame using the current wavelength solution (tables $girWavCoef$, *girSlitGeo* and optionaly $girWres$). This gives the expected central position of the line in pixels coordinates on the CCD frame. The line is searched inside a box around the expected center (window: `XWidth` × `YWidth`) and the centroid is calculating on the pixels of this box. The centroid of the 2D profile is computed in the box using `iqeFunc` (Image Quality Estimate), a function extracted from the MIDAS package. The resulting position ($X_{\rm line}$, $Y_{\rm line}$) can then be compared to the current localization *locY*. The value of *locY* in $X_{\rm line}$ is calculated and $Y_{\rm line} - locY(X_{\rm line})$ is an estimate of the localization shift for the corresponding line.

This operation is carried out for each valid line center (not rejected by `iqeFunc`), resulting in a frame of difference ($N_{\rm lines} \times 5$). A 2D polynomial fit over all spectra (`XOrder` × `YOrder`) is done on this frame to estimate the localization shift for all science spectra at any position along the dispersion axis. This fit is carried out using a sigma-clipping algorithm.

The result is a `NAXIS2`×`LOCNS` frame which is an offset added to the current localization *locY* for adjustment.

2.2.14 giRecDLocfit.py

Python implementation description

```
(Following description could be obtained on-line using: giRecDLocfit.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]
Reading module parameters from girBLDRS.ini:[Extraction Parameters]
NAME
   giRecDLocfit.py - rebinning recipe.

   SYNOPSIS
           giRecDLocfit.py [-h|--help]
   or
           giRecDLocfit.py [options] [dbsubFrame [locyFrame [locwFrame]] [wlenTable]

   DESCRIPTION
   Run giAdjustLoc() on the specified input frame  <dbsubFrame>
   where:
          <locyFrame>      - locy frame filename,
                             default 'dbsubFrame.locy.fits'
          <locwFrame>      - locw frame filename
                             default 'dbsubFrame.locw.fits'
          <wlenTable>      - line table
                                 default search in $GIR_CALIB

   valid options are:
     -h|--help                display help and exit
     -d|--debug               set debug mode
     -v|--verbose             set verbosity level,
                              repeated occurence increase verbosity
     -q|--quiet               no messages
     -t|--time                show processing time
     -z|--defaults            show default values
     --params                 dump INI config parameters
     -w|--ovwrite             overwrite product files
     --novwrite               do not overwrite product files
     --[no]compress           [do not] compress product files
     --outdir <dir>           output directory for products
     --datadir <dir>          specify input data directory
     -b|--bpmap <bpixFile>    specify bad pixel map filename
                              '-' for none
     --xorder <XPolyOrder>    order of X polynomial fit
     --yorder <YPolyOrder>    order of Y polynomial fit
                              xdeg = ydeg = 1 will fit a plane
     --xwidth <XWinSize>      sampling step along X (every 'xstep' pixel taken)
     --ywidth <YWinSize>      sampling step along Y (every 'ystep' pixel taken)
     --dsigma <sigma>         bias-clipping: multiple of sigma
     --dniter <niter>         bias-clipping: number of iterations
     --dmfrac <mfrac>         bias-clipping: min fraction of points
                              accepted/total. Should be in range [0.0-1.0]
```

```
  --splist <list>          extract specified spectra:
                            for example: --spectra 2,10,30:40:2,55
                            will extract spectra:
                                    2,10,30,32,34,36,38,40,55
  --wres <wlenRes>         Use on input the table 'wlenRes' of wavelength corrections
                           produced by Crosscorrelation. Normally this
                           is used with simultaneous calibration, but could be used
                           with any subset of spectra.
                           if wlenRes='-' we use the name generated from extSpFrame as
                          'first_part_extSpFrame+.wres.tfits' (use -z to see)

OUTPUTS
resulting frames go to <dbsubFrame>.locd.fits and <dbsubFrame>.chebd.tfits

ENVIRONMENT VARIABLES
  PYTHONSTARTUP  - python startup script
  GIR_PYTHON     - GIRAFFE BLDRS python modules directory
  GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
  GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 23482 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecDLocfit.py
(Following default values could be obtained on-line using: giRecDLocfit.py -z )
   Reading BLDRS parameters from girBLDRS.ini:[General]
   Reading module parameters from girBLDRS.ini:[Extraction Parameters]
   $Id: giRecDLocfit.py,v 1.3 2003/03/25 15:15:46 anonymous Exp $
    dbsubFile  = /giraffe/blecha/BLDRS/girBLDRS1.09/data/dummy.fits
    locyFile   = /giraffe/blecha/BLDRS/girBLDRS1.09/data/dummy.clocy.fits
    locwFile   = /giraffe/blecha/BLDRS/girBLDRS1.09/data/dummy.clocw.fits
    wlenFile   = /giraffe/gircalib2.1/girLineSun.tfits
    wresFile   = None
    bpixFile   = -
    xorder     : 4
    yorder     : 2
    xwidth     : 6
    ywidth     : 12
    dsigma     : 2.5
    dniter     : 2
    dmfrac     : 0.9
    verbose    : 1
    debug      : 0
    showtime   : 1
    ovwrite    : 0
    datadir    : None
    outdir     : None
    cfgdir     : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
```

2.2.15 giFitPsfProfile - Tranverse PSF modeling and accurate Localization

Name: **giFitPsfProfile** Section: **Extraction**
Python name: **giRecPsfLoc.py**

Purpose

Fit PSF on a group of stacked spectral bins of each spectrum using the preliminary localization then makes a polynomial X,Y model of these parameters.

Function parameters

Name	Type	Description
`binsize`	`int`	number of pixels to stack for each PSF fit
`YOrder`	`integer`	degree of 1-D (X) polynomial fit of the localization ridge
`Worder`	`integer`	degree of 2-D (X,Y) polynomial fit of the width of the localization mask
`lSpectra`	`integer`	list of spectra (usually from OzPoz table)
`pModel`	`string`	Name of the function used as PSF profile
`pExpWid`	`float`	Initial exponent parameter of the PSF profile
`pMaxWid`	`float`	Initial width parameter of the PSF profile

Input frames

Name: *dbSubFrame* **Type:** `BDRMIMG` **Format:** `xy`
Extension: `dbSub.fits`
Description: any preprocessed image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
`EXPTIME`	PR_FITS	`double`	s	DIC: Integration time
`BIASSIGMA`	DRS	`float`	ADU	Computed bias sigma in pre/over-scan
`DARKVALUE`	DRS	`float`	e-	The value of the actualy subtracted dark current. It is set to 0 if no subtraction is made (treshold not reached).
`CONAD`	CCDDCS	`double`	e-/ADU	DIC: The conversion factor which translates ADU to photonic electrons.
`NFIBRES`	DRS	`integer`	none	Number of fibres for the slit

Name: *locYMaster* **Type:** `LOCY` **Format:** `nxExt`
Extension: `.clocy.fits`
Description: standard localization center
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
`GIRFTYPE`	DRS	`string`	none	type of frame data
`GRATING`	GIR_ICS	`string`	none	DIC: ESO name for grating unit.
`SLIT`	GIR_ICS	`string`	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
`WLEN0`	GIR_ICS	`double`	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *locWyMaster* **Type:** LOCWY **Format:** nxExt
Extension: .clocw.fits
Description: standard localisation half-width
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
GIRFTYPE	DRS	string	none	type of frame data
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Other inputs

none

Output frames

Name: *plocY* **Type:** LOCY **Format:** nxExt
Extension: .clocy.fits
Description: current localization giving the spectrum center along y at each x
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PSFMODEL	DRS	string	none	name of the function used as PSF profile (psfexp2)
PSFXBINS	DRS	integer	none	number of pixels stacked for each PSF fit
PSFYDEG	DRS	integer	none	degree of 1D (X) polynomial of the localization lane
PSFWDEG	DRS	integer	none	degree of 2D (X,Y) polynomial of the width of the localization mask
CREATOR	DRS	string	none	Source of data (responsible or processing recipe)

Name: *plocWy* **Type:** LOCWY **Format:** nxExt
Extension: .clocw.fits
Description: current width of the NFF spectra (measured along the y axis)
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
GIRFTYPE	DRS	string	none	type of frame data

Name: *psfExpo* **Type:** LOCPSFEXPO **Format:** nxExt
Extension: .psfexpo.fits
Description: Exponent of the transverse PSF - used by optimal extraction
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PSFMODEL	DRS	string	none	name of the function used as PSF profile (psfexp2)
PSFPARAM	DRS	string	none	name of the PSF parameter which is represented
PSFXBINS	DRS	integer	none	number of pixels stacked for each PSF fit
PSFCOEFFi	DRS	float	none	2D Cheb. fit coef. to represent the parameter
PSFYDEG	DRS	integer	none	degree of 1D (X) polynomial of the localization lane
PSFWDEG	DRS	integer	none	degree of 2D (X,Y) polynomial of the width of the localization mask

Name: *psfWidt* **Type:** LOCPSFWIDT **Format:** nxExt
Extension: .psfwidt.fits
Description: Width of the transverse PSF - used by optimal extraction
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PSFMODEL	DRS	string	none	name of the function used as PSF profile (psfexp2)
PSFPARAM	DRS	string	none	name of the PSF parameter which is represented
PSFXBINS	DRS	integer	none	number of pixels stacked for each PSF fit
PSFCOEFFi	DRS	float	none	2D Cheb. fit coef. to represent the parameter
PSFYDEG	DRS	integer	none	degree of 1D (X) polynomial of the localization lane
PSFWDEG	DRS	integer	none	degree of 2D (X,Y) polynomial of the width of the localization mask

Name: *psfCent* **Type:** LOCPSFCENT **Format:** nxExt
Extension: .psfcent.fits
Description: Center of the transverse PSF - only for control plot
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PSFMODEL	DRS	string	none	name of the function used as PSF profile (psfexp2)
PSFPARAM	DRS	string	none	name of the PSF parameter which is represented
PSFXBINS	DRS	integer	none	number of pixels stacked for each PSF fit
PSFCOEFFi	DRS	float	none	2D Cheb. fit coef. to represent the parameter
PSFYDEG	DRS	integer	none	degree of 1D (X) polynomial of the localization lane
PSFWDEG	DRS	integer	none	degree of 2D (X,Y) polynomial of the width of the localization mask

Name: *psfBack* **Type:** LOCPSFBACK **Format:** nxExt
Extension: .psfback.fits
Description: Background of the transverse PSF - only for control plot
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PSFMODEL	DRS	string	none	name of the function used as PSF profile (psfexp2)
PSFPARAM	DRS	string	none	name of the PSF parameter which is represented
PSFXBINS	DRS	integer	none	number of pixels stacked for each PSF fit
PSFCOEFFi	DRS	float	none	2D Cheb. fit coef. to represent the parameter
PSFYDEG	DRS	integer	none	degree of 1D (X) polynomial of the localization lane
PSFWDEG	DRS	integer	none	degree of 2D (X,Y) polynomial of the width of the localization mask

Name: *psfAmpl* **Type:** LOCPSFAMPL **Format:** nxExt
Extension: .psfampl.fits
Description: Amplitude of the transverse PSF - only for control plot
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PSFMODEL	DRS	string	none	name of the function used as PSF profile (psfexp2)
PSFPARAM	DRS	string	none	name of the PSF parameter which is represented
PSFXBINS	DRS	integer	none	number of pixels stacked for each PSF fit
PSFCOEFFi	DRS	float	none	2D Cheb. fit coef. to represent the parameter
PSFYDEG	DRS	integer	none	degree of 1D (X) polynomial of the localization lane
PSFWDEG	DRS	integer	none	degree of 2D (X,Y) polynomial of the width of the localization mask

Other outputs

Name	Type	Description
pLocc	girLoccTable	Polynomial models for localisation (1D in x - one model per spectrum) and localisation half-width (2-D one global model)
Extension: .clocc.tfist		
Relevant FITS keywords:	GIRTTYPE GRATING SLIT WLEN0 LPOLYDEG LOCYDEG LOCWDEG LOCWCOEFF	
Relevant table columns:		
NSPEC	integer	running number of the spectrum/fibre from 1 to 135(320)
COEFFi	double	Chebyshev polynomial coefficients for localisation Y position

General description

We use standard or preliminary localization to stack spectral elements (typically per pack of 64 elements) along the center of the localization line of each spectra.

The PSF fit is done on each stack and a global polynomial model of each PSF parameter is built so as to describe the transverse PSF all over the image.

This model is later used for optimal extraction.

Mathematical description

The Point Spread Function perpendicular to the dispersion ($PSF_\perp$) can be fitted on the spectra once the localization mask has been determined.

Each spectrum is divided in <binsize> stacks (which are 4096/binsize pixels wide, typically 64) inside the localization mask. Within each stack, all spectral bins are superimposed using the localization mask as reference center so as to produce a single 1D compressed profile for a given stack.

The points of the resulting 1D profile are fitted by an exponential profile of the PSF[1] (given by the parameter <pModel>), using a non-linear fit (Levenberg-Marquart method):

$$PSF_\perp(Y)_{X,n} = A_{X,n} \exp\left(-\left|\frac{Y - Y_{X,n}}{W'_{X,n}}\right|^{W''_{X,n}}\right) + B_{X,n}, \tag{2.3}$$

where n is the number of the spectrum, and (X,Y) the pixel position (X is the dispersion direction). The fitted parameters $A_{X,n}$, $B_{X,n}$, $W'_{X,n}$, $W''_{X,n}$, $Y_{X,n}$ are respectively the amplitude, the background, the two shape parameters and the localization of the center in the direction perpendicular to the dispersion of the spectrum n for the bin X.

The first guess of the parameters, in the non-linear fit, are given by the threshold localization (from *giLocalSpectra*()) as far as the centroids are concerned. The input parameters <pMaxWid> and <pExpWid> are respectively the first guesses of $W'_{X,n}$ and $W''_{X,n}$.

The parameter $W''_{X,n}$ allows the adjustment of a large range of profiles. $W''_{X,n} = 2$ corresponds to the Gaussian. The larger $W''_{X,n}$ is, the steeper the wings of the PSF and the closer to the box-shaped profile. The lower it is, the sharper the profile is, and the more extended the wings are.

The procedure is illustrated in Fig. 2.3.

The local PSF is therefore described by its center $Y_{X,n}$, two shape parameters $W'_{X,n}$ and $W''_{X,n}$, the amplitude and the background. Only the first three parameters have a "general validity" and are used later for extraction purpose.

The two shape parameters $W'_{X,n}$ and $W''_{X,n}$ are combined to obtain one single width parameter that is analog to the one derived in giLocalSpectra. This parameter $W_{X,n}$ is the half-width at one-hundredth-maximum and represents the whole width inside which the spectra are to be extracted:

$$W_{X,n} = W'_{X,n} \ln(100)^{1/W''_{X,n}}. \tag{2.4}$$

The outputs of the procedure are:

[1]This analytical function can be easily normed. Its integral can be expressed as follows:

$$\int_{-\infty}^{+\infty} e^{\left(\frac{|x|^a}{b}\right)} dx = \frac{\left(-\frac{1}{b}\right)^{\left(-\frac{1}{a}\right)} \Gamma\left(\frac{1}{a}\right)}{a}$$

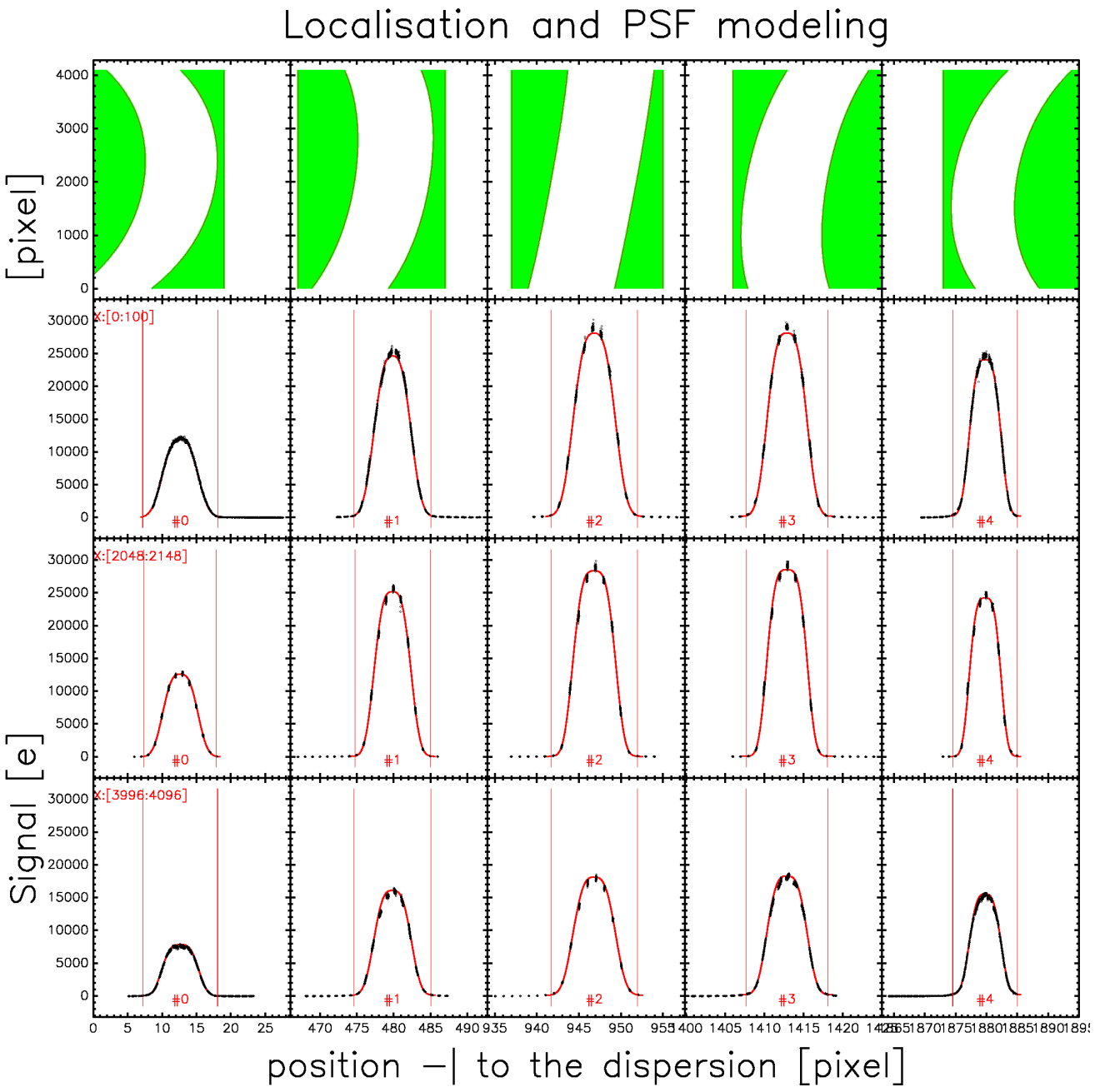


Figure 2.3: Five spectra of the simultaneous calibration flat-field in MEDUSA mode are shown for the setup "high-resolution 9" (λ 535.4–514.3 nm). The upper graphs show the localization lane over the full CCD field (dispersion direction is strongly compressed) while the three bottom panels show the individual spectra cross-sections recentered in the localization lane of each spectral bin. Each dot corresponds to a single pixel. The full line is the fitted exponential model of the $\mathrm{PSF}_{\perp}$ (Eq 2.3). Each box is an over-plot of 100 spectral bins. Spectra are evenly distributed over the detector, the #0 and #4 being located close to the edges of the CCD while the #2 is approximately in the center. The bottom panel corresponds to the red end of the spectra while the upper panel displays spectra located on the "top" of the CCD. Note the continuously populated cross-section, sort of dithering effect, located in the regions where the curvature of the localization lane is strongest.

- *locY* is the update of the localization lane, i.e. the polynomial adjustment of the centers $Y_{X,n}$ of the fitted PSF profiles.
- *chebY* are the coefficients (<`YOrder`> coefficient per spectrum) of the polynomial describing the localization lane.
- *locWy* is the 2D fit (`WOrder` × `WOrder`) of the width $W_{X,n}$ on the whole CCD frame.

If the degrees <`YOrder`> and <`WOrder`> are not specified in the input parameters, the values from the

master localization are used, i.e. *locYMaster*.LOCYDEG and *locYMaster*.LOCWDEG.

2.2.16 giRecPsfLoc.py

Python implementation description

```
(Following description could be obtained on-line using: giRecPsfLoc.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]
Reading module parameters from girBLDRS.ini:[Extraction Parameters]

   NAME
   giRecPsfLoc.py - localization recipe for bias and dark corrected frame.

   SYNOPSIS
     giRecPsfLoc.py [-h|--help]
   or
     giRecPsfLoc.py [options] [dbsubFrame [locyFrame [locwFrame] ] ]

   DESCRIPTION
   Run giFitPsfProfile() on the specified dark and bias corrected frame
   <dbsubFrame> where:
          <locyFrame>     - locy frame filename, default to
                            <dbsubFrame>.clocy.fits
          <locwFrame>     - locw frame filename, default to
                            <locyFrame>.clocw.fits, <dbsubFrame>.clocw.fits

   valid options are:
         -h|--help               display help and exit
         -d|--debug              set debug mode
         -v|--verbose            set verbosity level,
                                 repeated occurence increase verbosity
         -q|--quiet              no messages
         -t|--time               show processing time
         -z|--defaults           show default values
         --params                dump INI config parameters
         -w|--ovwrite            overwrite product files
         --novwrite              do not overwrite product files
         --outdir <dir>          output directory for products
         --datadir <dir>         specify input data directory
         --lfull|--lsewc         localization on all spectra
         --lsiwc                 localization on the 5 SIWC spectra
         --splist <list>         get localized spectra number from list:
                                 for example: --splist 2,10,30:40:2,55
                                 will assign spectra those numbers:
                                     2,10,30,32,34,36,38,40,55
         --yorder <deg>          order of Chebyshev polnomial fit
         --worder <deg>          order of Chebyshev 2D polnomial fit
         --binsize <size>        size of bin along dispersion axis
         --pexpwid <pExpWid>     exponential PSF profile exponent,
                                 will not be fitted if positive
                                 default 3.0
         --pmaxwid <pMaxWid>     exponential PSF profile max width,
                                 default 16
         --pmodel <model>        PSF profile model: 'psfexp', 'psfexp2'
                                 default: 'psfexp'
         -N                      number of spectra to be localized

   OUTPUTS
   resulting localization goes to <rawFrame>.plocy.fits <rawFrame>.plocw.fits
   and <rawFrame>.plocc.tfits

   EXAMPLES
   Display all defaults values and exits

     giRecPsfLoc.py -z
```

```
  Verbose run with a bin size of 64 on frame 'flat137_1.bsub.fits'

    giRecPsfLoc.py -v  --binsize=64 flat137_1.bsub.fits

  ENVIRONMENT VARIABLES
    PYTHONSTARTUP  - python startup script
    GIR_PYTHON     - GIRAFFE BLDRS python modules directory
    GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
    GIR_DEVELOP    - use development version if defined
```

Python defaults

```
  Source: 19022 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecPsfLoc.py
 (Following default values could be obtained on-line using: giRecPsfLoc.py -z )
    Reading BLDRS parameters from girBLDRS.ini:[General]
    Reading module parameters from girBLDRS.ini:[Extraction Parameters]
    $Id: giRecPsfLoc.py,v 1.2 2003/03/25 15:15:47 anonymous Exp $
     dbsubFile  = FfMedFulldummy.fits
     locyFile   = FfMedFulldummy.clocy.fits
     locwFile   = FfMedFulldummy.clocw.fits
     bpixFile   : -
     yorder     : 3
     worder     : 3
     lfull/lsewc: 1
     lsiwc      : 0
     binsize    : 64
     pexpwid    : 3.0
     pmodel     : psfexp
     plocyFile  = FfMedFulldummy.plocy.fits
     plocwFile  = FfMedFulldummy.plocw.fits
     ploccFile  = FfMedFulldummy.plocc.tfits
     psfprmFile = FfMedFulldummy.psf%s.fits
     verbose    : 1
     debug      : 0
     showtime   : 1
     ovwrite    : 0
     datadir    : None
     outdir     : None
     cfgdir     : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
```

2.2.17 giExtractSpectra - Spectra Extraction

Name: **giExtractSpectra** Section: **Extraction**
Python name: **giRecNExt.py**

Purpose

The extraction implements two methods: summation along a virtual slit and classical Horne's method. The local error and the residual background are also computed.

Function parameters

control parameters for extraction PSF and σ-clipping.

Name	Type	Description
eMethod	string	Extraction method: SUMM - simple summation; OPTIMAL
heWid	float	added to half-width of the extraction virtual slit
eSigma	float	multiple of σ (extSigClip[1] σ-clipping)
eNiter	integer	number of iterations (extSigClip[2] σ-clipping)
eMfrac	float	min fraction of points accepted/total (extSigClip[3] σ-clipping)
PfitMinGood	integer	OPTIMAL - required min. number of accepted points (no OPTIMAL for this bin otherwise)
pFlux	integer	OPTIMAL - total flux or amplitude estimate flag
pDwOrd	integer	OPTIMAL preliminar widening fit - order of the polynomial fit to the PSF widening
pFBkg	integer	OPTIMAL - flag, if set PSF fit includes background fitting
pCDis	float	OPTIMAL preliminar widening fit - distance from center rejection threshold
pAFrc	float	OPTIMAL preliminar widening fit - minimum amplitude for constant variance
pMaxDw	float	OPTIMAL preliminar widening fit - maximum value for PSF width correction
pSFrc	float	OPTIMAL preliminar widening fit - maximum admitted value for expected/measured sigma ratio
pNMin	float	OPTIMAL preliminar widening fit - minimum value to compute PSF weight
pAMin	float	OPTIMAL preliminar widening fit - minimum admitted value for PSF amplitude
pModel	string	OPTIMAL - PSF model (psfexp)
hSigma	float	OPTIMAL - Horne sigma. sigma-multiple to sigma-clipping during the PSF fit

Input frames

Name: *dbSubImage* **Type:** BDRMIMG **Format:** xy
Extension: dbSub.fits
Description: any bias-dark-subtracted frame: extMethod=1 needs GIRFTYPE='REPIMG'
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BIASSIGMA	DRS	float	ADU	Computed bias sigma in pre/over-scan
DARKVALUE	DRS	float	e-	The value of the actualy subtracted dark current. It is set to 0 if no subtraction is made (treshold not reached).
EXPTIME	PR_FITS	double	s	DIC: Integration time
CONAD	CCDDCS	double	e-/ADU	DIC: The conversion factor which translates ADU to photonic electrons.

Name: *locY* **Type:** LOCY **Format:** nxExt
Extension: .clocy.fits
Description: current localization giving the spectrum center along y at each x for the current setup.
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *locWy* **Type:** LOCWY **Format:** nxExt
Extension: .clocw.fits
Description: current width of the NFF spectra (measured along the y axis) for the current setup.
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *psfExpo* **Type:** LOCPSFEXPO **Format:** nxExt
Extension: .psfexpo.fits
Description: Exponent of the transverse PSF - used by optimal extraction only
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength
PSFMODEL	DRS	string	none	name of the function used as PSF profile (psfexp2)
PSFPARAM	DRS	string	none	name of the PSF parameter which is represented
PSFXBINS	DRS	integer	none	number of pixels stacked for each PSF fit
PSFCOEFFi	DRS	float	none	2D Cheb. fit coef. to represent the parameter
PSFYDEG	DRS	integer	none	degree of 1D (X) polynomial of the localization lane
PSFWDEG	DRS	integer	none	degree of 2D (X,Y) polynomial of the width of the localization mask

Name: *psfWidt* **Type:** LOCPSFWIDT **Format:** nxExt
Extension: .psfwidt.fits
Description: Width of the transverse PSF - used by optimal extraction only
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLENO	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength
PSFMODEL	DRS	string	none	name of the function used as PSF profile (psfexp2)
PSFPARAM	DRS	string	none	name of the PSF parameter which is represented
PSFXBINS	DRS	integer	none	number of pixels stacked for each PSF fit
PSFCOEFFi	DRS	float	none	2D Cheb. fit coef. to represent the parameter
PSFYDEG	DRS	integer	none	degree of 1D (X) polynomial of the localization lane
PSFWDEG	DRS	integer	none	degree of 2D (X,Y) polynomial of the width of the localization mask

Name: *curBadPixMap* **Type:** BADPIX **Format:** xyPrep
Extension: .crmbpm.fits
Description: Optional current *dbSubImage* bad pixel map
Relevant FITS keywords:

Name	DIC	Type	Unit	Description

Name: *slImage* **Type:** SLIMG **Format:** xyPrep
Extension: .slimg.fits
Description: Optional frame of fitted scattered light for *dbSubImage* for current image
Relevant FITS keywords:

Name	DIC	Type	Unit	Description

Other inputs

none

Output frames

Name: *exSp* **Type:** EXTSP **Format:** nxExt
Extension: .[o/e]xtsp.fits
Description: frame of extracted spectra - file extension according the method used
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BITPIX	PR_FITS	integer	none	DIC: # of bits per pix value
BZERO	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
BSCALE	PR_FITS	double	none	DIC: real=pixel_value*BSCALE+BZERO
CREATOR	DRS	string	none	Source of data (responsible or processing recipe)
DATAMIN	PR_FITS	double	e-	DIC: Minimal pixel value
DATAMAX	PR_FITS	double	e-	DIC: Maximal pixel value
EXTNX	DRS	integer	none	Number of pixel per spectrum
EXTNS	DRS	integer	none	Number of extracted spectra on the rebinned S2dFrame
SLFRAME	DRS	string	none	Frame with scatered light (when used for extraction)
LYFRAME	DRS	string	none	Frame with localisation used for extraction
LWFRAME	DRS	string	none	Frame with localisation width used for extraction
EMETHOD	DRS	string	none	SUMM or OPTIMAL, extraction method
PMODEL	DRS	string	none	PSF profile model used during OPTIMAL extraction
HSIGMA	DRS	float	none	6.0 sigma-multiple during OPTIMAL extraction
HEWIDH	DRS	float	none	Extra width to localisation during OPTIMAL extraction
ECLIPSIGMA	DRS	float	none	multiple of sigma (sigma-clipping of width adjustement during OPTIMAL extraction)
ECLIPNITER	DRS	integer	none	number of iterations (sigma-clipping of width adjustement during OPTIMAL extraction)
ECLIPMFRAC	DRS	float	none	min fraction of points accepted/total (sigma-clipping of width adjustement during OPTIMAL extraction)

Name: *exSpErr* **Type:** EXTERRS **Format:** nxExt
Extension: .[o/e]xtsperr.fits
Description: frame of extracted spectra errors - file extension according the method used
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
GIRFTYPE	DRS	string	none	type of frame data

Name: *exSpNpixels* **Type:** EXTNPIX **Format:** nxExt
Extension: .[o/e]xtspnix.fits
Description: frame of pixel counts in extracted spectra - file extension according the method used
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
GIRFTYPE	DRS	string	none	type of frame data

Name: *extLocY* **Type:** EXTLOCY **Format:** nxExt
Extension: .[e/o]xtspyct.fits
Description: Centroid of the extracted element (SUM) or copy of locY input - file extension according the method used
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
GIRFTYPE	DRS	string	none	type of frame data

Other outputs

none

General description

This function applies to any preprocessed science or calibration frame, obtained either in the MEDUSA or ARGUS/IFU mode. Note, that the scattered light, if not negligible or irrelevant, should be removed previously. Two options are offered:

1. VIRTUAL SLIT - a simple addition of pixel values along a virtual slit. The error is set to the expected shot-noise.

2. OPTIMAL EXTRACTION - This is a variant of Horne's method. The elimination of CRH is carried out in a way inspired by Baranne et al. (1996, A&AS 119, 373). The error is computed from the fit of the PSF and the number of pixels used for each extracted point.

Mathematical description

1. if <`eMethod`> = "SUMM"

   It is assumed that CRH detection and replacement has already been made and scattered light subtracted.

   Summation is made along a virtual slit. The slit is 1 pixel wide (dispersion direction), is centered on $locY(x,n)$ and is $2 \times$ (`heWidth` $+ locWy(x,n)$) long.

   The extracted spectrum is:

$$exSp(x_c,n) = \sum_{i \in slit} prop_i \times dbSubImage(x_c,y_i)$$

   where $prop_i$ is the fraction of the pixel to include, $prop_i = 1$ for all pixel excepted the first and last where $0.0 < prop_i < 1.0$.

   The extracted spectrum error is the sum of variance over included pixesl:

$$exSpErr(x_c,n) = \left(\sum_{i \in slit} prop_i \times vardbSubImage(x_c,y_i) \right)^{1/2}$$

$$\begin{aligned} &vardbSubImage(x_c,y) = \\ &\quad dbSubImage(x_c,y) + dbSubImage.\texttt{BIASSIGMA}^2 \\ &\quad + dbSubImage.\texttt{DARKVALUE} \times dbSubImage.\texttt{EXPTIME} + slImage(x_c,y) \end{aligned}$$

   If $slImage$ is not given it assumes 0.

   In this case the output frame $exSpNpixels$ of pixel counts in extracted spectra is merely a slightly modified replica of $locWy(x,n)$ input value.

2. if <`eMethod`> = "OPTIMAL"

   This method relies heavily on the accuracy of the transverse (perpendicular to the dispersion) PSF. The standard PSF_0 is described, for each spectral bin x_n, by 2 shape parameters - width and exponent - determined from the FF calibration exposure. The position of the center of the PSF_0 $[x_n,n]$ has been previously adjusted using the simultaneous wavelength calibration of the current frame. The actual extraction is made in two steps.

   (a) Adjustemnt of the PSF

   First of all an attempt is made to adjust the width of the PSF using the current frame.

   For each spectral bin we fit development of the stadard PSF function

   $PSF(A,\Delta w,B) = A \times PSF_0[x_n,n] + \frac{\partial PSF_0}{\partial w}\Delta w + B$

   Only points for which $\mid y_c -$ `pCDis` $\mid > 0$ are used for the fit. Fitted parameters A, Δw and B are accepted if $A >$ `pAMin` and if $\Delta w <$ `pMaxDw`. For spectra with number of accepted bins above (`PfitMinGood`) a Chebyshev polynomial model of degree `pDwOrd` in x is fitted to Δw and standard PSF model for $width$ parameter is updated accordingly: $width = width + \Delta w$.

(b) Extraction

We follow the original Horne's method (1986, PASP 98, 609) which estimates the total flux of the bin as:

$$exSp(x,n) = \frac{\sum_i W_H(x,y_i) \times cleanDbSubImage(x,y_i)/PSF(y_i))}{\sum_i W_H(x,y_i)}$$

with $W_H(x,y_i) = PSF(x,y_i)^2/var(y_i)$, and where *cleanDbSubImage* is identical to *dbSubImage* at the beginning of the first iteration, and differs from it by the detected and flagged CRHs in the course of the iterations.

The *var* and *cleanDbSubImage* are determined by the iterative fit of the PSF to the data. During this step the PSF shape and position are completely freezed. Only the amplitude A of the PSF and, optionally, the background B, are computed. The sigma-clipping is governed in a usual way by the parameters <`eSigma`> , <`eNiter`> and <`eMfrac`> . At each iteration the point with biggest $|\ \frac{(dbSubImage-PSF)}{\sqrt{PSF}}\ |>$ `eSigma` is flagged. All flagged points are replaced by the current fit at each iteration. The iterations are pursued till no point is rejected or <`eNiter`> or <`eMfrac`> is exceeded. The *var* is than computed as *smooth* variance of the fitted PSF rather than directly from the data.

The error of the result is computed as the quadratic sum of weighted measured error of the fit using only un-flagged points.

$$exSpErr(x,n) = \frac{N_{total}}{N_{total}-N_{param}} \sqrt{\frac{\sum_i PSF(y_i) \times mask(y_i) \times W_H(y_i) \times (CleanDbSubImage(y_i) - A \times PSF(y_i))^2}{\sum_i PSF(y_i) \times mask(y_i) \times W_H(y_i)}}$$

The additionnal error takes into account the flux of replaced points

$$exSpErr_{cleaning}(x,n) = \sqrt{\sum_i (1-mask(y_i)) \times A \times PSF(y_i))}$$

2.2.18 giRecNExt.py

Python implementation description

```
(Following description could be obtained on-line using: giRecNExt.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]
Reading module parameters from girBLDRS.ini:[Extraction Parameters]

   NAME
   giRecNExt.py - extraction recipe for bias, dark corrected frame with
   localization.

   SYNOPSIS
     giRecNExt.py [-h|--help]
   or
     giRecNExt.py [options] [dbsubFrame[Range] [locyFrame [locwFrame
                               [slFrame [psfprmsFrame] ] ] ] ]

   DESCRIPTION
   Run giExtractSpectra() on the
   specified dark and bias corrected frame <dbsubFrame>
   where:
          <locyFrame>      - locy frame filename,
                             default 'dbsubFrame.locy.fits'
          <locwFrame>      - locw frame filename
                             default 'dbsubFrame.locw.fits'
          <slFrame>        - frame with model of scattered light, '-' for none
                             default none
          <psfprmsFrame>   - PSF params frame stub, '-' for none
                             (only the common part of name is used EX: 'Medusa1H9Mast')
   valid options are:
```

```
    -h|--help                 display help and exit
    -d|--debug                set debug mode
    -v|--verbose              set verbosity level,
                              repeated occurence increase verbosity
    -q|--quiet                no messages
    -t|--time                 show processing time
    -g|--graphs               build graphics
    --nographs                do not build any graphics
    -z|--defaults             show default values
    -b|--bpmap <bpixmap>      specify bad pixel map filename
    --params                  dump INI config parameters
    -w|--ovwrite              overwrite product files
    --novwrite                do not overwrite product files
    --outdir <dir>            output directory for products
    --datadir <dir>           specify input data directory
    --emethod <method>        extraction method: 'SUMM', 'HORNE' or 'OPTIMAL'
    --splist <list>           extract specified spectra:
                               for example: --spectra 2,10,30:40:2,55
                               will extract spectra:
                                  2,10,30,32,34,36,38,40,55
    --esigma <sigma>          extr-clipping: multiple of sigma
    --eniter <niter>          extr-clipping: number of iterations
    --emfrac <mfrac>          extr-clipping: min fraction of points
                              accepted/total. Should be in range [0.0-1.0]
    --eron <ron>              new bias sigma (RON) value for dbsubframe

  PSF options:
    --pmodel <psf>            PSF profile model
    --hewid <hewid>           Horne additional extract hlf-width to locw
    --hsigma <hsigma>         Horne sigma; points are rejected if:
                                  (fit-data)**2 > hewid*VAR

    --pampl <pTolAmpl>        PSF amplitude tolerance
    --pcent <pTolCent>        PSF position tolerance
    --pbkgd <pTolBkgd>        PSF background tolerance
    --pwid1 <pTolWid1>        PSF width 1 tolerance
    --pwid2 <pTolWid2>        PSF width 2 tolerance

    --pflux or --nopflux      total flux or amplitude estimate flag
    --pmaxdw <PfitMaxDwidth>  max value for PSF width correction
    --pdword <PfitDwOrder>    order of DW polynomial fit
    --pamin <PfitMinAmpl>     min admitted value for PSF amplitude
    --pnmin <PfitMinNorm>     min value to compute PSF weight
    --psfrc <PfitMaxSigfrc>   max admitted value for expected/measure sigma ratio
    --pafrc <PfitMinAmplFrc>  min ampl for constant variance
    --pcdis <PfitCenDist>     distance from center rejection threshold
    --pfbkg                   if set will fit background and amplitude
    --nofbkg                  if set will not fit background

      or alternatively all params

    --psftol <pTolAmpl:pTolCent:pTolBkgd:pTolWid1:pTolWid2>


OUTPUTS
resulting extraction goes to <rawFrame>.extsp.fits

EXAMPLES

  giRecNExt.py -z                          - Display all defaults values and exits
  giRecNExt.py -v --emethod=SUMM Fff.fits  - Verbose run with a summation method
                                             on frame 'Fff.fits'

ENVIRONMENT VARIABLES
  PYTHONSTARTUP  - python startup script
```

```
    GIR_PYTHON     - GIRAFFE BLDRS python modules directory
    GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
    GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 30733 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecNExt.py
(Following default values could be obtained on-line using: giRecNExt.py -z )
    Reading BLDRS parameters from girBLDRS.ini:[General]
    Reading module parameters from girBLDRS.ini:[Extraction Parameters]
    $Id: giRecNExt.py,v 1.3 2003/06/20 13:20:59 anonymous Exp $
     dbsubFiles= None
     locyFile  =
     locwFile  =
     slFile    = -
     psfprmFile=
     bpmap     :
     emethod   : SUMM
     esigma    : 2.5
     eniter    : 2
     emfrac    : 0.9
     eron      : None
     hewid     : 1
     pmodel    : psfexp
     hsigma    : 9.0
    Horne options:
     pampl     : 0.1
     pcent     : 0.3
     pbkgd     : 0.04
     pwid1     : 0.03
     pwid2     : 0.03
    Optimal options:
     pamin     : 30.0
     pnmin     : 0.5
     pflux     : 1
     pamxdw    : 0.5
     psfrc     : 2.0
     pafrc     : 0.1
     pcdis     : 0.1
     pdword    : 3
     pfbkg     : 1
     extspFile    =
     extspErrFile =
     extspNpixFile=
     extspYcenFile=
     verbose      : 1
     debug        : 0
     showtime     : 1
     graphics     : 0
     ovwrite      : 0
     datadir      : None
     outdir       : None
     cfgdir       : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
     caldir       : /giraffe/gircalib2.1
```

2.2.19 giGetWaveSolution - Wavelength Solution

Name: **giGetWaveSolution** Section: **Wavelength calibration**
Python name: **giRecNWcal2.py**

Purpose

The wavelength solution is obtained from a separate (all fibres exposed to calibration lamp - SEWC) or simultaneous (5 fibres exposed to calibration lamp - simultaneous calibration) exposure

Function parameters

control parameters for line detection and for σ-clipping

Name	Type	Description
wsOptSol	logical	flag for fitting the optical parameters (wsOptSol=TRUE) or only the residuals relative to the standard optical model (wsOptSol=FALSE)
lType	string	type of emission lines: "THARNE" or "TELLURIC"
lWidth	double	list of widths in pixels of the window for line detection and fit
leThres	float	multiple of BIAS above which a pixel value is considered to betray the presence of a calibration (emission) line
wsSigma	float	multiple of σ (σ-clipping)
wsNiter	integer	number of iterations (σ-clipping)
wsMfrac	float	min fraction of accepted/total lines (σ-clipping)
xwSigma	float	multiple of σ above which clipping occurs (width in x)
xwNiter	integer	number of σ iterations (width in x)
Mfrac	float	min fraction of accepted/total lines (width in x)
xwOrderX	integer	polynomial degree along the x axis for x-width modeling
xwOrderY	integer	polynomial degree along the y axis for x-width modeling
wsOrderX	integer	polynomial degree along the x axis for wavelength solution
wsOrderY	integer	polynomial degree along the y axis for wavelength solution

Input frames

Any extracted spectra of any calibration source accepted (flat-fielded or not)

Name: *exSp* **Type:** EXTSP **Format:** nxExt
Extension: .[o/e]xtsp.fits
Description: extracted spectra of full W-calibration exposure (with simultaneous calibration)
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
PIXSIZX	CCDDCS	double	um	DIC: Physical extent of a pixel in X direction.
PIXSIZY	CCDDCS	double	um	DIC: Physical extent of a pixel in Y direction.
BIASSIGMA	DRS	float	ADU	Computed bias sigma in pre/over-scan
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
GRATGRV	GIR_ICS	double	gr/nm	DIC: Grating"s number of grooves per nanometer GROOVES is used in the conversion formula. ENC=ZORDER+ ALIGN+ RESOL* asin(WLEN* ORDER* GRV/(2* cos(ROT))) + TEMPRAMP* (TEMP-TEMPREF)
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
FILT	GIR_ICS	string	none	DIC: Filter name. AL: one of LR1-8 HR1-22
LAMP	DRS	string	none	Name of the W Calibration source
LAMPID	TC_ICS	string	none	DIC: Reference light lamp identifier.

Name: *exSpErr* **Type:** EXTERRS **Format:** nxExt
Extension: .[o/e]xtsperr.fits
Description: corresponding extracted spectra errors
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *locY* **Type:** LOCY **Format:** nxExt
Extension: .clocy.fits
Description: current localization giving the spectrum center along y at each x for the current setup.
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Other inputs

Name	Type	Description
girSlitGeo	girSlitGeoTable	Geometrical model of the slit
Extension: girSlitGeoXXX.tfits		
Relevant FITS keywords: SLIT GRATING WLEN0		
Relevant table columns:		
NSPEC	integer	running number of the spectrum/fibre from 1 to 135(320)
XF	float	Almost constant fiber position in x (perpendicular to the slit)
YF	float	Almost regularly decreasing fiber position in y (along the slit)
RP	integer	running number of positionner retractor
girWavCoef	girWavCoefTable	Optional standard solution; kwords give the optical solution while table columns provide coefficients of the Chebytchev correction
Extension: girWavCoefXXX.tfits		
Relevant FITS keywords: GRATING SLIT WLEN0 WSWSOLUTION WSWMODEL WSWDIRECTION WSWSUBSLITFIT WSWFCOLL WSWGCAM WSWTHETA WSWSLITDX WSWSLITDY WSWSLITPHI WSWLINEFILE WSWLINEMODEL WSWLINETYPE WSWLINE WSWLINETHRESHOLD WSWLINENITER WSWLINENTEST WSWLINEDCHISQ		
Relevant table columns:		
CIJ	double	Coefficients of the wavelength solution (for the residuals relative to the optical model)
linesWlampMast	girLineTable	wavelengths and fluxes of the calibration spectrum
Extension: girLineXXXX.tfits		
Relevant FITS keywords: GIRTTYPE LAMP LAMPID		
Relevant table columns:		
WLEN	float	Wavelength of the line
NAME	string	Name of line
FLUX	float	Expected central flux/[pixel.sec]
COMMENT	string	Setup followed by cotrolled vocabulary comment
gratingMast	girGratingTable	parameters of grating and setup used to compute the initial optical model
Extension: girGratingXXX.tfits		
Relevant FITS keywords: GIRTTYPE GRATGRV GRATING		
Relevant table columns:		
SETUP	none	Index
THETA	float	Grating angle
ORDER	integer	Grating order
WLMIN	float	Shortest usable wavelength in the current setup
WLMAX	float	Longest usable wavelength in the current setup
WLEN0	double	Grating central wavelength
BAND	float	WLMAX-WLMIN
RIFA	integer	Resolution in IFU and ARGUS mode
RMED	integer	Resolution in MEDUSA mode

Output frames

Python implementation produces several auxiliary files used for control plots with terminal extension of format '.??res.fits'. These are not included in this list (see description under giRecNWcal2.py)

Other outputs

Name	Type	Description
`girWavCoef`	`girWavCoefTable`	Kwords give the optical solution while table columns provide coefficients of the Chebytchev correction
Extension: `girWavCoefXXX.tfits`		
Relevant FITS keywords:	`GRATING SLIT WLENO WSWSOLUTION WSWMODEL WSWDIRECTION WSWSUBSLITFIT WSWFCOLL WSWGCAM WSWTHETA WSWSLITDX WSWSLITDY WSWSLITPHI WSWLINEFILE WSWLINEMODEL WSWLINETYPE WSWLINE WSWLINETHRESHOLD WSWLINENITER WSWLINENTEST WSWLINEDCHISQ`	
Relevant table columns:		
`CIJ`	double	Coefficients of the wavelength solution (for the residuals relative to the optical model)

General description

The processing is used only on Th-Ar-Ne full calibration frames (SEWC).

1. An initial solution, if no valid current solution exists, is obtained from the optical design.
2. The analytical model of the dispersion relation is used

   The `girWavCoefXXX` table contains set of KWs/parameters describing the analytical model representing the x position (in pixels) as a function of λ and n. If `girWavCoefXXX` does not exists for XXX setup, the initial solution comes from girGratingHR316/LR600 table.

$x_{opt}(\lambda, n) =$

$$\frac{G f_{\text{coll}} \left(\cos\theta \left(-\frac{\lambda m}{a} + \frac{xf(n)\cos\theta}{\mathcal{D}} + \frac{f_{\text{coll}}\sin\theta}{\mathcal{D}} \right) + \sin\theta \sqrt{1 - \frac{yf(n)^2}{\mathcal{D}^2} - \left(-\frac{\lambda m}{a} + \frac{xf(n)\cos\theta}{\mathcal{D}} + \frac{f_{\text{coll}}\sin\theta}{\mathcal{D}} \right)^2} \right)}{- \left(\sin\theta \left(-\frac{\lambda m}{a} + \frac{xf(n)\cos\theta}{\mathcal{D}} + \frac{f_{\text{coll}}\sin\theta}{\mathcal{D}} \right) \right) + \cos\theta \sqrt{1 - \frac{yf(n)^2}{\mathcal{D}^2} - \left(-\frac{\lambda m}{a} + \frac{xf(n)\cos\theta}{\mathcal{D}} + \frac{f_{\text{coll}}\sin\theta}{\mathcal{D}} \right)^2}}$$

   where f_{coll} and G are respectively the focal length of the collimator and the magnification factor of the camera. The $xf(n)$ and $yf(n)$ give the reference position of the fiber n on the entrance slit, while a is the groove spacing, θ is the grating angle and m is the diffraction order; x is the pixel coordinate. All necessary optical parameters are provided via FITS KWs.
3. A list of lines is associated to each source of the calibration spectra (thorium/argon/neon source, sky). that after preliminary set-up, this list was *cleaned* in order to avoid unnecessary systematic rejections during the step 4.
4. A line profile is fitted to each line and the polynomial expansion is adjusted to the differences between the found x positions and those obtained from the current model. The input model is updated. The sigma clipping is implemented to eliminate a few lines which do not fit the model.
5. Optionally (technical set-up) the control of elimination/iteration include access to all optical parameters in order to *tune* the process.

Mathematical description

The Wavelength-calibration proceeds in following steps:
`<lType>` flag; the changes – if any – implied by this possibility are mentioned at the end of each point.

1. **Definition of the initial solution:**

   Read the `girWavCoefXXX` table, which contains the coefficients of the standard wavelength solution and spectrograph constants necessary for the computation of the initial optical model of the setup XXX.

   If such table does not exists or if we want to start from scratch (first calibration after changes in spectrograph) we use spectrograph configuration `girGratingHR316/LR600` . This is completed by slit geometry `girSlitGeoXXX`

In any cases, a first hint of the wavelength solution is computed through the equation of the optical model:

$$x_{opt}(\lambda, n) =$$

$$\frac{G\, f_{\rm coll}\left(\cos\theta\,\left(-\frac{\lambda\, m}{a}+\frac{xf(n)\,\cos\theta}{\mathcal{D}}+\frac{f_{\rm coll}\,\sin\theta}{\mathcal{D}}\right)+\sin\theta\,\sqrt{1-\frac{yf(n)^2}{\mathcal{D}^2}-\left(-\frac{\lambda\, m}{a}+\frac{xf(n)\,\cos\theta}{\mathcal{D}}+\frac{f_{\rm coll}\,\sin\theta}{\mathcal{D}}\right)^2}\right)}{-\left(\sin\theta\,\left(-\frac{\lambda\, m}{a}+\frac{xf(n)\,\cos\theta}{\mathcal{D}}+\frac{f_{\rm coll}\,\sin\theta}{\mathcal{D}}\right)\right)+\cos\theta\,\sqrt{1-\frac{yf(n)^2}{\mathcal{D}^2}-\left(-\frac{\lambda\, m}{a}+\frac{xf(n)\,\cos\theta}{\mathcal{D}}+\frac{f_{\rm coll}\,\sin\theta}{\mathcal{D}}\right)^2}} \tag{2.5}$$

where $\mathcal{D}^2 = xf^2+yf^2+f_{\rm coll}^2$ and where $f_{\rm coll}$ = `girGratingXXXXX` :$FCOLL$, G = `girGratingXXXXX` :$GCAM$, θ = `girGratingXXXXX` :$THETA$, a = `girGratingXXXXX` :$GSPACE$ and m = `girGratingXXXXX` :$GRATORD$ are the same $\forall n$ (fibres). Alternately same parameters are available as KWs in `girWavCoefXXX` . The fibre-dependent $xf(n)$ and $yf(n)$ are given in columns `girSlitGeoXXX:` XF/YF table.

2. **Mask:** For each calibration spectrum, a mask is computed from the `girLineThAr` table. The central x position of each line to be considered (i.e., *a priori*, each line of `girLineThAr`) is computed from the preliminary wavelength solution above, and the pixels to be examined are defined as belonging to a range centered on this computed position and having a width <`lWidth`> .

   If <`lType`> = `FALSE`, the mask is computed from the `girLineSky` .

3. **Line search:** In each range, the pixels are sorted by order of intensities. The x_{max} position of the pixel with the largest intensity I_{max} is retained as a first guess of the position of the PSF to be fitted. The average of the two lowest pixel values $fback$ is retained as a first guess of the background. The inverse contrast r is computed as the ratio of the background error (which is computed in the course of the extraction process) and of the largest intensity, and the line is rejected if $r >$ `leThres`$^{-1}$.

   This process is done iteratively, first taking into account the brightest lines in a large spectral window. Then the windowing in tuned in order to use the more numerous and dimmer spectral lines.

4. **Line fitting:** An exponential PSF function is fitted to each retained line, using x_{max}, $fback$, Δx([2]) and I_{max} as first guesses for the position, background, FWHM width[3] and amplitude respectively. The formal error σ_w on the width is provided by the fit. Likewise, the formal error σ_p on the line position is provided by the fit and is used to compute the weights in the fit of the wavelength solution.

5. **Polynomial fit of the PSF width variation:** The PSF width is modeled through a set of 2-D function describing the PSF in a global way (one function/subslit). Presently only single 2-D function is fully implemented and tested. The widths is LSQ fitted with Chebyshev polynomials, after normalization of the CCD pixel coordinates to the interval [-1,+1]:

$$widthx(x', n_k) = \sum_{i=0}^{I}\sum_{j=0}^{J} cw_{i,j,k} T_i(x') T_j(y'(n_k)), \tag{2.6}$$

   where the subslit number $k = 1, ...13$ in the MEDUSA mode and $k = 1, ...15$ in the IFU/ARGUS modes, $x'_k = \frac{2}{\texttt{NX}-1}(x_k - STARTX - \frac{\texttt{NX}-1}{2})$, $I \equiv$ `xwOrderX`, $J \equiv$ `xwOrderY`, and T_i are Chebyshev polynomials of degree i. Therefore, there is $13 \times npar$ such fits for the MEDUSA mode and $15 \times npar$ for the ARGUS/IFU modes. The information on the y coordinate cannot be replaced by the spectrum number n because even within a given subslit, the function $y(n)$ is not always continuous. The y coordinate has to be normalized separately for each subslit k, so that $[y_{min}(n_k^{min}), y_{max}(n_k^{max})] \rightarrow [-1, +1]$, where n_k^{min} and n_k^{max} are the minimum and maximum spectrum numbers within the subslit k:

$$y'_k = \frac{2}{locY_{max}(n_k^{max}) - locY_{min}(n_k^{min})}\left(locY(n_k) - \frac{locY_{min}(n_k^{min}) + locY_{max}(n_k^{max})}{2}\right) \tag{2.7}$$

   The LSQ fit is done by weighting the measured widths with the inverse of the variance:

$$weightw(x, n_k) = 1/\sigma_w(x, n_k)^2$$

[2] $\Delta x = x_{opt}(\lambda + 0.5 \times$ `gratingMast` $:WLEN0/R) - x_{opt}(\lambda - 0.5 \times$ `gratingMast` $:WLEN0/R)$, where R is the resolving power, i.e. `gratingMast` $:RMED$ or `gratingMast` $:RIFA$, depending on $exSp$.`SLIT`.

[3] for a gaussian, one would rather consider the quantity $\sigma = FWHM/2.355$

where $\sigma_w(x, n_k)$ was estimated for each line through the LSQ fit of the profile.

The lines whose widths differ by more than `xwSigma` $\times \sigma_w$ from the fitted function are discarded (where σ_w was estimated through the fit of the line profile), the list of rejected lines is updated and the fit is re-iterated until no more rejection occurs (a test on only the first width parameter is considered sufficient) or until the number of iterations reaches $<$`xwNiter`$>$ or until the fraction of accepted lines becomes lower than $<$`xwMfrac`$>$. The number $<$`xwNiter`$>$ of iterations and the number of rejected lines $<$`xwMfrac`$>$ are updated.

This process generates an estimate of the PSF width as a function of position on the CCD, but it is expressed in pixels while an estimate in wavelength is needed for sky subtraction. For that, one needs a linear transformation using the wavelength solution.

6. **Fit of the coefficients of the wavelength solution:** The fit is done in two stages, the first of which being carried out only when one wants to fit the wavelength solution.

   (a) If $<$`wsOptSol`$>$ = `TRUE`, an iterative non-linear LSQ fit is done on Equation 2.5 above. The optical parameters are set to their inital values and fit is donne according the specifications given through input parameters. In the LSQ fit, the measured line positions are weighted by the inverse of their variance estimated previously, during the line fitting:

$$weightp = 1/\sigma_p(x, n_k)^2$$

   (b) The residuals $x - x_{opt}(\lambda_{\rm lab}, n)$ are computed and modelized with Chebyshev polynomials: As for optical solution the Chebyshev correction is computed as global 2-D solution:

$$x(\lambda', n) - x_{opt}(\lambda', n) = \sum_{i=0}^{\texttt{wsOrderX}} \sum_{j=0}^{\texttt{wsOrderY}} c_{i,j} T_i(\lambda') T_j(y'(n))$$

   where λ' and y' are normalized quantities as in 2.6. As in the case of the fit to the optical model, this fit involves the same weights $weightp = 1/\sigma_p^2$. The lines farther away than `wsSigma` $\times \sigma_p$ from the fitted solution are discarded (σ_p is the error on the line position estimated from the fit of the profile), the list of discarded lines is updated and the fit iterated until no more outlier is detected, or until the number of iterations or the fraction of rejected lines is reached. The number of iterations $<$`wsNiter`$>$ is updated as well as the number of rejected lines $<$`wsMfrac`$>$.

   Finally, the fitted $c_{i,0}$ coefficients are stored in columns of the `girWavCoefXXX` table (`girWavCoefXXX` :CIJ).

2.2.20 giRecNWcal2.py

Python implementation description

```
(Following description could be obtained on-line using: giRecNWcal2.py -h )

   NAME
   giRecNWcal2.py - wavelength solution recipe.

   SYNOPSIS
           giRecNWcal2.py [-h|--help]
   or
           giRecNWcal2.py [options] [extSpFrame [locyFrame]]

   DESCRIPTION
   Run giGetWaveSolution() on the
   specified dark and bias corrected frame <dbsubFrame>:
   where
     <extSpFrame>      - extracted spectra frame filename,
                         will also load extracted spectra error frame:
                         'extSpFrame.extsperr.fits'
     <locyFrame>       - localization centroid frame filename
                         will also load localization half width frame:
                         default 'extSpFrame.locw.fits'

   valid options are:
      -h|--help                  display help and exit
```

```
  -d|--debug        set debug mode
  -v|--verbose      set verbosity level,
                              repeated occurence increase verbosity
  -t|--time                   show processing time
  -q|--quiet                  quiet mode
  --noquiet                   some prints
  -z|--defaults               show default values
  -g|--graphs <-1,0,1>        show graphs (central spectrum+refrence lines)
                              1 = graphs, 0 = nographs, -1 = only graphs then exit
  -L <zmin>                   Use log scale for graph with '0' at zmin
  -X|XARG X1:X2,...           limits of the graphics  in input pixels
  --nographs                  do not build any graphics
  --nolines                   no line detection (use previous run)
  --params                    dump INI config parameters
  -w|--ovwrite                overwrite product files
  --outdir <dir>              output directory for products
  --datadir <dir>             specify input data directory
  --wltable <lineWlenTable>   lines wavelengths table (default: defined by setup)
  --sgtable <slitGeoTable>    slit geometry table (default: defined by setup)
  --wstable <waveSolTable>    wave solution table (default: defined by setup)
  --ltype <lineType>          'THARNE' or 'TELLURIC' lines

line selection before the fit
  --ldetect <lineDetect>      line detection threshold during the preliminary search
                              for lines (multiple of BIAS sigma)
  --lwidth <lineWidth>        list of widths for line detection and fit window
                              (should be decreasing; recommended '60,40,20')
  --lfluxrat                  only lines with neighbours having relative intensity <
                              1./lfluxrat accepted
  --lbright <N or thresh>     integer: only N with highest nominal brightest
                              float: only lines with nominal intensity > threshold
  --lwlen<min,max>            minimal, maximal wavelength in [nm]

lines selection during the fit
  --linmod <lineModel>        line profile model: 'psfexp' 'psfexp2' or 'gaussum'
  --lthresh <lineThresh>      line detection threshold  during  the  line  fitting
                              (multiple of BIAS sigma)
  --loffset <lineOffset>      accepted differenc: position of (raw maximum - fit)
  --lniter <lineNiter>        line detection fit number of iterations
  --lntest <lineNtest>        line detection fit number of test
  --ldchsq <lineDchsq>        line detection chisq diff for test
  --lreswid <lResWid>         line_width/resolution_width factor
  --lexpwid <lExpWid>         exponential line profile exponent,
                              will not be fitted if positive
  --xres/--noxres             use/do not use X residuals for line detection

Otical model fit control
  --optmod <opticalModel>     optical model for dispersion: 'xoptmod' or 'xoptmod2'
  --oniter <optmNiter>        optical model fit number of iterations
  --ontest <optmNtest>        optical model fit number of test
  --odchsq <optmDchsq>        optical model fit chisq diff for test
  --optsol                    fit optical model params
  --optdir                    dispersion direction 1 or -1
  --nooptsol                  fit only residuals
  --subsfit                   fit using subslit geometry
  --nosubsfit                 fit on whole slit

PSF fit control prams
  --xwsigma <xwsigma>         PSFwidth-clipping: multiple of sigma
  --xwniter <xwniter>         PSFwidth-clipping: number of iterations
  --xwmfrac <xwmfrac>         PSFwidth-clipping: min fraction of points accepted/total
                              Should be in range [0.0 - 1.0]
  --xworder <x,y>             X and Y polynomial orders for x-width Chebyshev polyfit

Wave solution Chebyshev correction fit control
```

```
   --wssigma <wssigma>         WaveSol-clipping: multiple of sigma
   --wsniter <wsniter>         WaveSol-clipping: number of iterations
   --wsmfrac <wsmfrac>         WaveSol-clipping: min fraction of points accepted/total
                               Should be in range [0.0 - 1.0]
   --wsorder <x,y>             X and Y polynomial orders for wavelength Chebyshev
                               correction

Test options
   --fcoll                           colimator focal [mm], default 900
   --cfact                           camera scale factor, default 0.4
   --gtheta                          grating angle [rad], default from table
   --gthetad                         grating angle [degree], default from table
   --gwlen0                          grating central wavelngth, default from the table
   --gspace                          grating spacing, default (1/316 or 1/600)
   --stop <0-9>                      0 do everything
                                     1 stop after line detection
   --slit <sdx,sdy,sphi>             slit displacement params (def 0,0,0)
   --fflags (f,G,Th,m,s,dx,dy,dth)   optical parameters fit flags
                                     if set, we fit the param
   --fflags STRING                   where STRING is any combination or F, A, G and S
                                       - F will fit focal length
                                       - G will fit camera factor
                                       - A will fit grating angle
                                       - S will fit slit displacement

OUTPUTS
  <INPUTFRAME>.xxres.fits
             detected lines fitted X center in pixels (along dispersion axis)
  <INPUTFRAME>.yyres.fits
             detected lines fitted Y center in pixels
  <INPUTFRAME>.iires.fits
             detected lines fitted amplitude
  <INPUTFRAME>.bbres.fits
             detected lines fitted background
  <INPUTFRAME>.wdres.fits
             detected lines fitted FWHM
  <INPUTFRAME>.sdres.fits
             detected lines fitted sigma(FWHM)
  <INPUTFRAME>.wsx.fits
             fitted PSF width [ns,nx]
  <INPUTFRAME>.wsx.tfits
             PSF width fit Cheb coefficients
  <INPUTFRAME>.xores.fits
             lines center: computed (using opt model) X in pixels
  <INPUTFRAME>.yores.fits
             lines center: computed (using opt model) Y in pixels
  <INPUTFRAME>.xmres.fits
             lines center: computed (using updated opt model) X in pixels
  <INPUTFRAME>.ymres.fits
             lines center: computed (using updated opt model) Y in pixels
  <INPUTFRAME>.ocres.fits
             2D fit of  X residuals
  <INPUTFRAME>.xcres.fits
             X residuals fit Cheb coefficients
  <INPUTFRAME>.xores.tfits
             wavelength solution

EXAMPLES

  Display all defaults values and exits

        giRecNWcal2.py -z

  Detect lines and fit line-profiles without any optical model
```

```
          giRecNWcal2.py -v --ldetect 50 ThaMedFullM10DARKBIAS.extsp.fits

  Private Function used:
    _wextrem(optParams,xf,yf)  - get minimal maximal wavelength on the CCD

  ENVIRONMENT VARIABLES
    PYTHONSTARTUP  - python startup script
    GIR_PYTHON     - GIRAFFE BLDRS python modules directory
    GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
    GIR_CALIB      - GIRAFFE BLDRS calibration files path
    GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 82109 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecNWcal2.py
(Following default values could be obtained on-line using: giRecNWcal2.py -z )
    Images+tables
     extspFile   = ThaMedFulldummy.extsp.fits
     extsperrFile= ThaMedFulldummy.extsperr.fits
     locyFile    = FfMedFulldummy.clocy.fits
     locwFile    = FfMedFulldummy.clocw.fits
     wltable     = None
     sgtable     = None
     wstable     = None

    Line detection
     ltype   : THARNE
     ldetect : 0.0
     lwidth  : [20.0]
     lfluxrat: 50.0
     lbrightN: 120
     lbrightThresh: 0.0
     lwlen   : 0.0 0.0
     lthresh : 2.5
     lreswid : 0.5
     lexpwid : 3.0
     loffset : 10.0
     lniter  : 50
     linmod  : psfexp
     lntest  : 7
     ldchsq  : 0.0001
     xwsigma : 2.5
     xwniter : 10
     xwmfrac : 0.9

    Optical model
     optsol  : 1
     oniter  : 50
     ontest  : 7
     odchsq  : 0.0001
     optmod  : xoptmod2
     optdir  : 1
     subsfit  : 0
     wssigma : 5.0
     wsniter : 10
     wsmfrac : 0.9
     xworder : 3,3
     wsorder : 8,6
     xres    : 0
     fcoll   : None
     cfact   : None
     gtheta  : None
     gwlen0  : 0.0
     gspace  : 0.0
```

```
 slit    : None None None
 fflags (f,G,Th,m,s,dx,dy,dth): (1, 0, 0, 0, 0, 1, 1, 1)
X graphlim: 0:-0
 stop    : 0
 verbose : 0
 debug   : 0
 time    : 1
 graphs  : 0
 ovwrite : 0
 datadir : None
 outdir  : None
 cfgdir  : /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/config
 caldir  : /giraffe/gircalib2.1
```

2.2.21 giRebinSpectra - Rebinning in wavelength space

Name: **giRebinSpectra** Section: **Wavelength calibration**
Python name: **giRecBinn2.py**

Purpose

Carry out rebinning of extracted spectra so that the result is equally spaced in λ or $\log \lambda$

Function parameters

Name	Type	Description
bnMethod	string	one of "SPLINE", "LINEAR" or "FFT"
bnStep	double	wavelength step used in rebinning (= 0, if rebinning is carried out in logarithmic wavelength space)
bnSize	integer	flag indicating how is derived the spectral range that is rebinned
bnLin	logical	flag indicating whether the rebinning is in lambda
bnLog	logical	flag indicating whether the rebinning is in natural-log lambda[nm]
bnRange	string	flag indicating how is derived the spectral range that is rebinned: "SETUP" or "COMMON"

Input frames

Name: *ffSp* **Type:** EXTSP **Format:** nxExt
Extension: .[o/e]xtsp.fits
Description: extracted and optinaly flat-fielded spectra
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
EXTNS	DRS	integer	none	Number of extracted spectra on the rebinned S2dFrame
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATGRV	GIR_ICS	double	gr/nm	DIC: Grating"s number of grooves per nanometer GROOVES is used in the conversion formula. ENC=ZORDER+ ALIGN+ RESOL* asin(WLEN* ORDER* GRV/(2* cos(ROT))) + TEMPRAMP* (TEMP-TEMPREF)
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Name: *ffSpErr* **Type:** EXTERRS **Format:** nxExt
Extension: .[o/e]xtsperr.fits
Description: associated error frame
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength

Other inputs

Name	Type	Description
girSlitGeo	girSlitGeoTable	Geometrical model of the slit
Extension: girSlitGeoXXX.tfits		
Relevant FITS keywords: SLIT GRATING WLEN0		
Relevant table columns:		
NSPEC	integer	running number of the spectrum/fibre from 1 to 135(320)
XF	float	Almost constant fiber position in x (perpendicular to the slit)
YF	float	Almost regularly decreasing fiber position in y (along the slit)
RP	integer	running number of positionner retractor
girWavCoef	girWavCoefTable	Kwords give the optical solution while table columns provide coefficients of the Chebytchev correction
Extension: girWavCoefXXX.tfits		
Relevant FITS keywords: GRATING SLIT WLEN0 WSWSOLUTION WSWMODEL WSWDIRECTION WSWSUBSLITFIT WSWFCOLL WSWGCAM WSWTHETA WSWSLITDX WSWSLITDY WSWSLITPHI WSWLINEFILE WSWLINEMODEL WSWLINETYPE WSWLINE WSWLINETHRESHOLD WSWLINENITER WSWLINENTEST WSWLINEDCHISQ		
Relevant table columns:		
CIJ	double	Coefficients of the wavelength solution (for the residuals relative to the optical model)
girWres	girWresTable	Optional wavelength spectra shift obtained from 5 SIMCAL spectra
Extension: .wlenres.tfits		
Relevant FITS keywords: GIRTTYPE		
Relevant table columns:		
WLRES	float	Wavelength shift of the corresponding spectrum

Output frames

The output file additional extension is according the binnig used: ...rebilin.fits or ....rebinlog.fits

Name: *wlSp* **Type:** BINSP **Format:** nlWbin
Extension: .rebin[lin/log].fits
Description: frame of rebinned spectra
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
BINWNX	DRS	integer	none	number of spectral elements on the rebinned S2dFrame
BINWNS	DRS	integer	none	number of spectra on the rebinned S2dFrame
CTYPE2	PR_FITS	string	none	DIC: Coordinate system of <axis direction>
CRPIX2	PR_FITS	double	none	DIC: Ref pixel in <axis direction>
CRVAL2	PR_FITS	double	none	DIC: Coordinate at reference pixel in <axis direction>
CDELT2	PR_FITS	double	none	DIC: Increment in <axis direction>
BINWLMIN	DRS	float	nm	wavelength or natural log of the first spectral element
BINWL0	DRS	float	nm	wavelength of the centre of the cetral spectral elements
BINWLMAX	DRS	float	nm	wavelength or natural log last spectral element so as BINWMAX=BINWMIN*(BINWNX-1)*BINSTEP
BINWSTEP	DRS	double	nm	wavelength linear or natural log step of the rebinned spectra
BINRANGE	DRS	string	none	spectral range source (usually 'setup')
BINMETHOD	DRS	string	none	spectral range source (usually 'setup')
CREATOR	DRS	string	none	Source of data (responsible or processing recipe)

Name: *wlSpErr* **Type:** `BINERRS` **Format:** `nlWbin`
Extension: `err.rebin[lin/log].fits`
Description: frame of rebinned spectra errors
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
`BINWNX`	DRS	`integer`	none	number of spectral elements on the rebinned S2dFrame
`BINWNS`	DRS	`integer`	none	number of spectra on the rebinned S2dFrame
`BINWSTEP`	DRS	`double`	nm	wavelength linear or natural log step of the rebinned spectra
`BINWL0`	DRS	`float`	nm	wavelength of the centre of the cetral spectral elements
`BINWSTEPLOG`	DRS	`double`	none	wavelength step of the logarithmic rebinned spectra; the spectral element N has a central wavelength=w0+10**((N-1)*step)

Name: *psflwidth* **Type:** `BINSP` **Format:** `nlWbin`
Extension: `.rebin[lin/log].fits`
Description: Width parameter for 1-D PSF along the dispersion direction (x)
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
`BINWNX`	DRS	`integer`	none	number of spectral elements on the rebinned S2dFrame
`BINWNS`	DRS	`integer`	none	number of spectra on the rebinned S2dFrame
`BINWSTEP`	DRS	`double`	nm	wavelength linear or natural log step of the rebinned spectra
`BINWL0`	DRS	`float`	nm	wavelength of the centre of the cetral spectral elements
`BINWSTEPLOG`	DRS	`double`	none	wavelength step of the logarithmic rebinned spectra; the spectral element N has a central wavelength=w0+10**((N-1)*step)

Other outputs

none

General description

The rebinning is done to obtain the $wlFfSp(Nlambda, Ns)$ spectra. The extracted spectra $ffSp(Nx, Ns)$ and associated information (error, number of points used) are used.

In order to get pixel positions x as a function of λ, the expression of the wavelength solution, via coefficients of `girWavCoefXXX` , `girSlitGeoXXX` and optionally `girWres` tables, is used.

For homogenization's sake, the spectral domain where spectra are rebinned is taken to be the same for each spectrum of the frame, according to <`binWRange`> :

- `binWRange` ="SETUP" : the spectral range is chosen as defined by the wavelength limits `girGratingXXXXX` :$WLMIN$ and `girGratingXXXXX` :$WLMAX$ corresponding to the observing set-up `girGratingXXXXX` :$SETUP$,

- `binWRange` ="COMMON" : the spectral range is taken as the common interval $[\lambda_1, \lambda_2]$ to all spectra of the frame. The wavelength limits $\lambda_{\min}(n)$ $\lambda_{\max}(n)$ are computed by fitting splines on series of points $(\lambda_i, x(\lambda_i, n))$ around the defined limits of the bandpass, for each spectrum.

  Then $\lambda_1 = \max_{n\in[1,Ns]} \left\lceil \frac{\lambda_{\min}(n) - \texttt{girGratingXXXXX:}WLMIN}{\texttt{binWStep}} \right\rceil \times \texttt{binWStep} + \texttt{girGratingXXXXX} :WLMIN$

  and $\lambda_2 = \min_{n\in[1,Ns]} \left\lfloor \frac{\lambda_{\max}(n) - \texttt{girGratingXXXXX:}WLMAX}{\texttt{binWStep}} \right\rfloor \times \texttt{binWStep} + \texttt{girGratingXXXXX} :WLMAX.$

  The limits are shifted to wedge the general wavelength scale (defined by `girGratingXXXXX` :$WLMIN$ and `girGratingXXXXX` :$WLMAX$).

Mathematical description

The new sampling in wavelength space is defined by the chosen extrema and the spacing <bnStep> .

- bnRange ="SETUP" : $ffSp$.BINWL0 = gratingMast :$WLMIN$
  and $ffSp$.BINWNX = $\left\lceil \frac{\text{gratingMast:}WLMAX - \text{gratingMast:}WLMIN}{\text{bnStep}} \right\rceil + 1$
- bnRange ="COMMON": $ffSp$.BINWL0 = λ_1 and $ffSp$.BINWNX = $\left\lceil \frac{\lambda_2 - \lambda_1}{\text{bnStep}} \right\rceil + 1$

The corresponding positions in pixel-space are derived from the analytical model of the wavelength solution stored in girWavCoefXXX , girSlitGeoXXX and optionally girWres tables.

- If <bnMethod> = "LINEAR", the intensity is conserved (but not the flux) and a linear interpolation is performed: the intensity is given by

$$wlFfSp(\lambda_j, n) = ffSp(x_i, n) + [ffSp(x_{i+1}, n) - ffSp(x_i, n)] \times (x(\lambda_j) - x_i) \tag{2.8}$$

  (note that the interval $x_{i+1} - x_i = 1$ since it is defined in the pixel space) and the error is given by interpolation of the variances and appropriate scaling:

$$wlFfSpError(\lambda_j, n) = \left(\frac{1}{x(\lambda_{j+1} - x(\lambda_j)}\right)^{1/2} \times$$
$$\left[ffSpErr(x_i, n)^2 + \left(ffSpErr(x_{i+1}, n)^2 - ffSpErr(x_i, n)^2\right) \times (x(\lambda_j) - x_i)\right]^{1/2} \tag{2.9}$$

- If <bnMethod> = "SPLINE", the intensity is conserved and a spline interpolation is performed.

When rebinning in logarithmic space (<bnLog> = TRUE), the spectra are processed in the same way and the corresponding positions in pixel-space of the rebinned pixels in $\log \lambda$ are also derived from the table girWavCoefXXX .

2.2.22 giRecBinn2.py

Python implementation description

```
(Following description could be obtained on-line using: giRecBinn2.py -h )
Reading BLDRS parameters from girBLDRS.ini:[General]
NAME
   giRecBinn2.py - rebinning recipe.

   SYNOPSIS
           giRecBinn2.py [-h|--help]
   or
           giRecBinn2.py [options] extSpFrame[Range] extSpErrFrame|- locyFrame [locwFrame]

   DESCRIPTION
   Run giRebinSpectra() on the specified extracted spectra <extSpFrame>
   where:
     <extSpFrame>      - extracted spectra frame filename,
                         will also load extracted spectra error frame:
                         'Frame.extsp.fits'
     <extSpErrFrame>   - extracted spectra error frame filename,
                         'Frame.extsperr.fits'
                          if '-' specified name is generated

   valid options are:
     -h|--help               display help and exit
     -d|--debug              set debug mode
     -v|--verbose            set verbosity level,
                             repeated occurence increase verbosity
     -q|--quiet              no messages
     -t|--time               show processing time
     -z|--defaults           show default values
     --params                dump INI config parameters
     -w|--ovwrite            overwrite product files
```

```
  --novwrite              do not overwrite product files
  --[no]compress          [do not] compress product files
  --outdir <dir>          output directory for products
  --datadir <dir>         specify input data directory
  --bnmethod <method>     specify rebinning method, one of:
                           SPLINE,LINEAR or FFT
  --[no]xres              [do not] use X residuals from Wavelength Solution
  --wres <wlenRes>        Use on input the table 'wlenRes' of wavelength corrections
                          uproduced by Crosscorrelation. Normally this
                          is used with simultaneous calibration, but could be used
                          with any subset of spectra.
                          if wlenRes='-' we use the name generated from extSpFrame as
                          'first_part_extSpFrame+.wres.tfits' (use -z to see)
  --bnstep <step>         specify rebinning lambda step [nm]
                          '-' will use resolution-dependent default
                          0.005  [nm] for HR 0.02 [nm] for LR
  --bnsize <nbins>        specify size of rebinned spectra
  --bnlog                 rebinning in natural-log lambda[mm];
                          lambda[nm]=1000000*e**(BINWLMIN+(XPIXEL-1)*BINSTEP)
  --bnlin                 rebinning in lambda
  --bnrange <range>       specify rebinning range, one of:
                          SETUP: use grating lambda minimum and maximum
                          COMMON: use spectra lambda common interval

OUTPUTS
resulting frames go to <extSpFrame>.rebin.fits and <extSpFrame>.rebinerr.fits

ENVIRONMENT VARIABLES
  PYTHONSTARTUP  - python startup script
  GIR_PYTHON     - GIRAFFE BLDRS python modules directory
  GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
  GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 26944 oct 16 10:11 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giRecBinn2.py
(Following default values could be obtained on-line using: giRecBinn2.py -z )
   Reading BLDRS parameters from girBLDRS.ini:[General]
   NAME
      giRecBinn2.py - rebinning recipe.

      SYNOPSIS
              giRecBinn2.py [-h|--help]
      or
              giRecBinn2.py [options] extSpFrame[Range] extSpErrFrame|- locyFrame [locwFrame]

      DESCRIPTION
      Run giRebinSpectra() on the specified extracted spectra <extSpFrame>
      where:
        <extSpFrame>      - extracted spectra frame filename,
                            will also load extracted spectra error frame:
                            'Frame.extsp.fits'
        <extSpErrFrame>   - extracted spectra error frame filename,
                            'Frame.extsperr.fits'
                             if '-' specified name is generated

      valid options are:
        -h|--help              display help and exit
        -d|--debug             set debug mode
        -v|--verbose           set verbosity level,
                               repeated occurence increase verbosity
        -q|--quiet             no messages
        -t|--time              show processing time
        -z|--defaults          show default values
```

```
      --params                dump INI config parameters
      -w|--ovwrite            overwrite product files
      --novwrite              do not overwrite product files
      --[no]compress          [do not] compress product files
      --outdir <dir>          output directory for products
      --datadir <dir>         specify input data directory
      --bnmethod <method>     specify rebinning method, one of:
                               SPLINE,LINEAR or FFT
      --[no]xres              [do not] use X residuals from Wavelength Solution
      --wres <wlenRes>        Use on input the table 'wlenRes' of wavelength corrections
                              uproduced by Crosscorrelation. Normally this
                              is used with simultaneous calibration, but could be used
                              with any subset of spectra.
                              if wlenRes='-' we use the name generated from extSpFrame as
                              'first_part_extSpFrame+.wres.tfits' (use -z to see)
      --bnstep <step>         specify rebinning lambda step [nm]
                              '-' will use resolution-dependent default
                              0.005  [nm] for HR 0.02 [nm] for LR
      --bnsize <nbins>        specify size of rebinned spectra
      --bnlog                 rebinning in natural-log lambda[mm];
                              lambda[nm]=1000000*e**(BINWLMIN+(XPIXEL-1)*BINSTEP)
      --bnlin                 rebinning in lambda
      --bnrange <range>       specify rebinning range, one of:
                              SETUP: use grating lambda minimum and maximum
                              COMMON: use spectra lambda common interval

   OUTPUTS
   resulting frames go to <extSpFrame>.rebin.fits and <extSpFrame>.rebinerr.fits

   ENVIRONMENT VARIABLES
     PYTHONSTARTUP  - python startup script
     GIR_PYTHON     - GIRAFFE BLDRS python modules directory
     GIR_CONFIG     - GIRAFFE BLDRS configuration files directory
     GIR_DEVELOP    - use development version if defined

missing extracted spectra frame
```

2.2.23 giCrossC - Computation and fit of the Cross Correlation peak.

Name: **giCrossC** Section: **CCF**
Python name: **giCrossC.py**

Purpose

Computes the cross-correlation function, with binary masks or with template spectra

Function parameters

Name	Type	Description
corMasktype	Integer	0 - binary masks, 1 - template spectra
spectraList	Integer	Standard list of spectra from 1 to maximum number; default means all spectra taken.
cDomain	Float	corStart,corEnd - Defines the portion of spectra on which we do the correlation in pixels or fraction of domain
cDomain	Float	corStart,corEnd - Defines the portion of spectra on which we do the correlation in lambda [nm]
Vdomain	Float	corVmin,corVmax - Minimum and maximum velocities and velocity [km/sec] where compute the correlation
cStep	Float	Step of correlation in [nm]; default is resolution dependent
templateList	Integer	Standard list of spectra from 1 of template spectra (if any) to which we correlate. If single number is given, we correlate to one spectrum, if list is given the number of items must be equal to number of spectra to which we correlate.

Input frames

Name: *wlSp* **Type:** BINSP **Format:** nlWbin
Extension: .rebin[lin/log].fits
Description: 2P - Ns spectra rebinned (linear or in log λ)
Relevant FITS keywords:

Name	DIC	Type	Unit	Description
NX	DRS	integer	none	Size of image in X direction (lambda)
NY	DRS	integer	none	Size of image in Y direction (slit)
BINWSTEP	DRS	double	nm	wavelength linear or natural log step of the rebinned spectra
BINWLMIN	DRS	float	nm	wavelength or natural log of the first spectral element
BINWLMAX	DRS	float	nm	wavelength or natural log last spectral element so as BINWMAX=BINWMIN*(BINWNX-1)*BINSTEP
BINWL0	DRS	float	nm	wavelength of the centre of the cetral spectral elements
GRATING	GIR_ICS	string	none	DIC: ESO name for grating unit.
WLEN0	GIR_ICS	double	nm	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength
SLIT	GIR_ICS	string	none	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus
BIASSIGMA	DRS	float	ADU	Computed bias sigma in pre/over-scan
ACTMJD	PR_FITS	double	days	DIC: Obs start AL: MJD at which the field was observed
EXPTIME	PR_FITS	double	s	DIC: Integration time
GEOLON	TCS	double	deg.	DIC: Tel geo longitute (+=East) (deg)
GEOLAT	TCS	double	deg.	DIC: Tel geo latitute (+=North) (deg)
GEOELEV	TCS	double	m	DIC: Elevation above sea level (m)
EQUINOX	PR_FITS	double	none	DIC: Standard FK5

Other inputs

Name	Type	Description
girRVMask	girRVMaskTable	Standard binary mask(s) for RV determination
Extension: girXXX.tfits		
Relevant FITS keywords: GRATING WLEN0		
Relevant table columns:		
WLEN1	float	wavelength
WLEN2	float	wavelength
girSlitGeo	girSlitGeoTable	Optional Geometrical model of the slit to be updated (with W-calibration frame only)
Extension: girSlitGeoXXX.tfits		
Relevant FITS keywords: GIRTTYPE SLIT GRATING WLEN0		
Relevant table columns:		
NSPEC	integer	running number of the spectrum/fibre from 1 to 135(320)
XF	float	Almost constant fiber position in x (perpendicular to the slit)
YF	float	Almost regularly decreasing fiber position in y (along the slit)
RP	integer	running number of positionner retractor

Output frames

none

Other outputs

Name	Type	Description
girSlitGeo	girSlitGeoTable	Optional updated geometrical model of the slit (with W-calibration frame only)
Extension: girSlitGeoXXX.tfits		
Relevant FITS keywords: GIRTTYPE SLIT GRATING WLEN0		
Relevant table columns:		
NSPEC	integer	running number of the spectrum/fibre from 1 to 135(320)
XF	float	Almost constant fiber position in x (perpendicular to the slit)
YF	float	Almost regularly decreasing fiber position in y (along the slit)
RP	integer	running number of positionner retractor
girWres	girWresTable	Optional wavelength spectra shift obtained from 5 SIMCAL spectra (any spectra)
Extension: .wlenres.tfits		
Relevant FITS keywords: GIRTTYPE		
Relevant table columns:		
WLRES	float	Wavelength shift of the corresponding spectrum

General description

Computes the cross-correlation function, with binary masks or with template spectra

Mathematical description

This function computes the cross-correlation between the observed spectra and either one single binary mask (i.e. the same for all objects of one frame) or several different masks (i.e. one mask for each type of object). In the first option the mask $RVMask$ has to be specified, while in the second option the masks are selected automatically from the object type, given either by a colour index or by a spectral type (in the case of stars). Otherwise the function assumes that the object is a solar type star and uses the corresponding mask.

An option is also offered to use template spectra ($RVTemplate$) instead of binary masks, using the flag $cfMasktype$: $cfMasktype =' mask'$ for binary masks, and $cfMasktype =' template'$ for template spectra.

The CCF is defined as

$$C(\epsilon) = \frac{R(\epsilon)}{R(\epsilon = \infty)}$$

with

$$R(\epsilon) = \int_{-\infty}^{+\infty} F(v) \cdot M(v - \epsilon)\, dv$$

where F is the spectrum, M the mask and v the velocity. In the case of a binary mask with L "holes" and of an observed spectrum $lwlFlxSp$, this reduces to (for the nth spectrum):

$$R(v, n) = \sum_{l=1}^{L} \sum_{ll_i} p_{l,ll_i}(v) lwlFlxSp(ll_i, n)$$

where n is the spectrum number, ll_i the central $\ln \lambda$ of the ith bin (i varies between two limits $i_{min}(l)$, $i_{max}(l)$ which are set by both edges of the lth mask hole), $lwlFlxSp(ll_i, n)$ the corresponding flux and $p_{l,ll_i}(v)$ the fraction of the ith pixel which is covered by the lth line of the mask at velocity v (let us recall that $v \propto \ln \lambda$). In practice, the CCF is normalized in this way:

$$CCF(\epsilon, n) = \frac{2 \cdot R(\epsilon, n)}{R(\epsilon_{min}, n) + R(\epsilon_{max}, n)}$$

where ϵ_{min} and ϵ_{max} are the borders of the correlation domain.
For template spectra, the expression of R becomes:

$$R(\epsilon, n) = \sum_{i=i_{min}}^{i_{max}} lwlFlxSp(ll_i, n) \cdot T\left(ll_i - \frac{\epsilon}{c}\right)$$

where $lwlFlxSp$ is the observed spectrum, T the interpolated template one, $ll \equiv \ln \lambda$, ϵ is a velocity in $\mathrm{km\,s^{-1}}$ and i_{min}, i_{max} are the bin numbers in the relevant part of the observed spectrum. The interpolation is linear, unless otherwise stated, while the relevant part is imposed by the amplitude of the barycentric correction:

$$\epsilon_{min} \equiv (\ln \lambda)_{min} = ll_1 + \frac{1}{c} barCor_{max}$$

$$\epsilon_{max} \equiv (\ln \lambda)_{max} = ll_{Nll} + \frac{1}{c} barCor_{min}$$

$barCor_{min}$ and $barCor_{max}$ being the minimum (negative) and maximum (positive) values of the barycentric correction, defined as the quantity to be added to the observed radial velocity to obtain the barycentric one. In this way, it is always the *same* part of the observed spectrum which is correlated with the mask or template, whatever the time of the year, provided that the object has no intrinsic RV variability.

Note that in the case of binary masks, the CCF is a dip for absorption line spectra and a peak for emission line spectra, while in the case of template spectra, the CCF is a peak.

2.2.24 giCrossC.py

Python implementation description

```
(Following description could be obtained on-line using: giCrossC.py -h )

  NAME
    giCrossC.py  - get crosscorelation with template spectrum

  SYNOPSIS
    giCrossC.py [-h|--help]
    giCrossC.py [options] FrameName

  DESCRIPTION
    FrameName is the name part to the first dot or the full name
    with or without pathof the bias subtracted image. range of name and
    globing '?' or '*' supported

    CC - Crosscorrelate spectra with template spectra:

    Each spectrum is correlated with template spectrum defined by -T option. In default
```

```
   operation the template spectrum is a binary mask selected according to the B-V of the
   spectrum (use -z option to see the mask assignation). The normalised CCf is computed
   in the domain of RV shift window DV1-DV2. Unless required specifically the
   simultaneous calibration spectra are correlated with a ThAr mask and the sky spectrum
   with the sky mask (actually G2 mask) and their RV window is automatically set
   centred on RV=0 according to the spectral resolution of the setup.

   Both Linear and Log binned spectra are accepted on input. In Log binned frame KW
   CTYPE2 must contain the string 'log' and BINWLMIN/MAX must be in Log([nm]) scale.

   Alternately, spectra could be correlated with any set of spectra (including itself)
   specified by -T option, both on the spectrum-spectrum basis or against the average of
   the spectra sub-set (--Tav option).

  Few example are given below to show the best strategy of use:

  EXAMPLES:

    # to see search (+-500 km/sec) + final correlation peak of one spectrum
    giCrossC.py --graph 4 -V 500 -R 1,2,1 SGR_Outer_FieldMedusa2H9dbSub.oxtsp.rebinlin.fits

    # spectra 1-10 correlated with no graphics
    giCrossC.py -N 2:11 -V 500 -R 1,2,1 SGR_Outer_FieldMedusa2H9dbSub.oxtsp.rebinlin.fits

    # all spectra with column 'object' beginning with 'SGRRG'
    giCrossC.py -O SGRRG -N 0:-1 -V 500 -R 1,2,1 SGR_Outer_FieldMedusa2H9dbSub.oxtsp.rebinlin.fits

    # have  a  look  on  object  on which we failed to automatically localise the peak
    giCrossC.py -O SGRRG -N 29 -g 4 -V 500 -R 1,2,1 SGR_Outer_FieldMedusa2H9dbSub.oxtsp.rebinlin.fits
    # specify appropriate search window
    giCrossC.py -O SGRRG -N 29 -g 4 -V -300,0 -R 1,2,1 SGR_Outer_FieldMedusa2H9dbSub.oxtsp.rebinlin.fits

    # The output printout could be converted to rdb and manipulated to produced required output:

    giCrossC.py  -N 1:-0 -V -300,0 SGR_Outer_FieldMedusa2H9dbSub.oxtsp.rebinlin.fits | txt2r | addcol corr \
    | row 'rverr>0.2&&E==0' | column sp object rv rvcorr rverr | tro


OPTIONS
  -h|--help               display help and exit
  -d|--debug              debug messages
  -v|--verbose            verbose processing;
                          repeated occurence of '-v' increase the verbosity
  -q|--quiet              no messages
  -z|--defaults           show default values

  # SIMULATION only (not maintained):
  -S|--simul <DW,SNR,Table>   input file is used as format to simulated spectrum
                              DW delta-lambda [nm]
                              SNR - signal to noise (0 no noise)
                              Table with simulated (solar) spectrum
  --wdomain <wmin,wmax,wst>   spectral domain given in [nm]; wstep must be multiple 0.001
  --wlog                      we simulate logarithmic spectrum

  # Correlation

  -T|--template<template_specifications>
                template_specification may be one of the following:
                - table : we give name with postfix '_RES_.tfits' or no postfix
                  THIS  IS  THE  STANDARD  USE
                  if no path is given, table is taken from $GIR_CALIB
                  EX: -T girG2
                      -T girG2H5.tfits
                      -T -               (Default)
```

```
                To each table, associated specifications must be given as follows
                table_name,BVmin,BVmax,table_class
                Default table+specifications are supplied as:
                 girF0 ,   -9.00 , 0.44 , 0     # class=0 use supplied RV window
                 girG2 ,    0.44 , 0.68 , 0     #        and apply RV correction
                 girK0 ,    0.68 , 9.00 , 0
                 girThAr ,  9.00 , 9.99 , 2     # class=2 use default RV window
                                                #        DO NOT apply RV correction
                completed as follows:
                table_name,-9.0,9.0,0,girThAr,9.00,9.99,2

              - image : we give name with postfix '.fits'
                EX: -T ThArMedPinH5.extsp.rebin.fits
                 spectrum/spectrum are corelated
              - spectra from image: postfix '.fits.[N]'
                if no-name given just the spectrum number '.N'
                EXAMPLES:
                -T ThArMedPinH5.extsp.rebin.fits.0 # first spectrum (number 0)
                -T .1                               # 1-st spectrum from input image)
                -T ThArMedPinH5.extsp.rebin.fits.1:9,10:20,21:-0
                   # if one or several ranges of spectra is given it must correspond
                   # to the cross-correlated spectra list (see -N option)
                -T ThArMedPinH5.extsp.rebin.fits. # use same spectra as on input image
                -T .                              # autocorrelate all spectra
                -T . --Tav                        # correlate with average spectrum
--Tav                   - Correlate with average template spectra (through -T option)
--cstep <cstepValue>    - step of cross-corelation in [nm] - recom. 0.005
                          it must be <= BINNSTEP, BINNSTEP/cstepValue be an integer
                          default is BINNSTEP of the template
                          Note that mask template are made with resolution 0.001 and
                          therefore for EX in  HR  where BINNSTEP=0.005;  possible
                          cstepValues are 0.005,0.0025,0.001,0.0005 ... Limitation: on
                          rebinned spectra only
-V <dvmin,[dvmax],[all_classes]> -
                          Delta RV limits to cross-correlation window [km/sec]. This
                          will normally apply only on the objects of first 'class'. If
                          all_classes  flag is 1 dvmin dvmax apply to all objects
                          (including CAL and SKY class) (use -z to see defined
                          classes).
--ZMAX <zmax,[zval]>      pixels with value >= zmax set to zval;
                          default for zval is zval=zmax (truncation)
--cdomain<X11:X21,X21:x22,...
                        -  if  given correlation is made separately for subsection
                           X11:X21,  X21:X22, ...[pixels]

--fixwidth <val>          fix width of holes in Correlation Mask to 'val[nm]'
                          this option overrides the variable-hole-width in input Mask
--cwdomain<W11:W21...>    as for cdomain but in wavelength units [nm]
-R <Rgraph,Wfactor,nR,lastWfactor> Iterate RV determination at most nR times, each time
                          recentering the window on the current RV and setting the
                          data  window  width to a 'Wfactor'*FWHM. If 'lastWfactor' is
                          specified,  the  last  itteration is followed by a parabolic
                          fit of peak position, other parameters remaining unchanged.
                          If  Rgraph=1  flag  is  set graph of CCF peak is produced at
                          each  iteration and not only at last iteration. Rgraph=2 has
                          the  similar  effect,  but  plot is produced only in case of
                          failure.
                          Any of nR,Wfactor,Rgraph could be defaulted by '-', all of
                          them by single '-'  see EXAMPLES below:
                          -R 1     # we have all correlation peaks
                          -R 0,2   # we fit on data in window 2*width (no plot)
                          -R 1,-,1 # 1 itterations
                          -R 2     # plot in case of failure only
                          -R 1,-,-,0.8   # position by extra parabolic fit
                          -R 1,-,-,-0.8  # an extra fit on reducedset of points
```

```
  --Vcorr <flag>            RV correction; flags possibles:
                             - 'b'  solar system barycentric (default),
                             - 'h'  heliocentric
                             - 'g'  geocentric,
                             - 'n'  no correction
                            Note  that  barycentric and heliocentric corrections INCLUDE
                            the geocentric correction
# Output options

  --txt                   printout in txt format
  --txts                  printout in txt SHORT format
  -s                      suppress header in txt/txts
  -o|--output <namePeak,nameRV>
                          produces image of the correlation peaks (spectra are replaced
                          with CC values) and
                          optionally 1D (Medusa) or 2D (IFU/Arg) image with RV [km/sec]
                          if '-' is given namePeak or nameRV automatic names are
                          generated from input image name
  --xf <affix>            produce new slit geometry table names:
                          girSlitGeo'SLIT''affix'.tfits  in  $GIR_CALIB  TBU  only for
                          primary calibration or after changes on fibre system
  --wres <wlenRes>        Produces table 'wlenRes' of wavelenght correction used for
                          rebinning. Normally this is        used with simultaneous
                          calibration, but could be used with any subset of spectra
                          (sky for exemple).
                          if wlenRes='-' we use the name generated from extSpFrame as
                          'first_part_FrameName+.wres.tfits' (use -z to see)
                          if wlenRes='term' output goes only to the terminal
#  --seq <num/name>        ????

# Input options

  --norm <boxsize>        - norm input spectra to continuum
                            Box_filter of 2*boxsize [px] is used to define the continuum.
                            MASK holes are weighted proportional to the continuum. If graph '3'
                            option is given - the continuum appears in blue
                            EX: --norm 100   uses a box filter +-100 around each point.
  --clean <boxsize,[plot_title]>
                          - remove cosmics using 2*boxsize box filter; default is cleaning
                            with boxsize=10. If plot tilte is given; we show control plot.
  -N N1:[N2[:DN]],N1...   - do CCR on spectra between <N1:N2) (N2 excluded)
           Extended Python notation is used; end: '-0', end-5: '-5'
           EX: '0:-0' or '1:-0'         # all spectra
               '0:30,c:-:20,-30:-0'     # 3 intervals plotted: beginning at 1 ending at 29
                                        # centre to centre+19
                                        # and from end-30 to end
           if N1,N2 fractional <0,1)    # the N/N_total (137 Medusa, 320 IFU/Argus)
           EX: 0.1:0.2,0.4:0.5          # from N_total*0.1-N_total*0.2,...
    or with exclusion
  -N 'N1:N2;excN1:excN2'  - as above but with ranges excN1:excN2 excluded
                     EX: '0:20;12,13'             # from 1 to 19 with 12 and 13 excluded
  -O|--object 'list'      - from  specified  Ns  select  only  object with name matching
                            regular  expression;  if  we  do  not  correlate  with  MASK
                            (spectrum-spectrum  correlation),  this  selection aplies to
                            both input and template spectra.
                            EX: -O CAL     select calibration (ThAr) spectra
                                -O  ST     select all object beginning with 'ST'
                                -O '.*11'  select all object with '11' anywhere
                                -O '^11'   select all object beginning with '11'
                                -O '.*SKY' select all object with 'SKY' anywhere
                            note  that  regexp  using  special  characters  '.*' must be
                            enclosed between simple quotes
  -M|--magnitude <max,[min]>- process only spectra with magnitude in the range
  -g|--graph <list>       - show variuos graphs
```

```
                          1 - correlation template
                          2 - correlation template and spectra (2 graphs)
                          3 - correlation template and spectra overploted
                          4 - correlation peak in RV ([km/sec]) scale
                          4w - correlation peak in wavelength ([nm]) scale
                          5 - VR as function of spectra number
                          5W - same in [nm]
  --titles <'title1;title2;...'>
                        - graphs titles (in order of given graphs)
  -X|XARG X1,...        - graphical limits in input pixels
  --WL <W1, ....        - graphical limits in wavelength
  --DW <DW>             - shift displayed mask (use to check position of C-peak)
  -L <zmin>             - Use log scale for graph with '0' at zmin
  --outdir <dir>        - output directory for products
  --ovwrite             - overwrite product files
  -W/H                  - Graph size in screen points (1280.1024) max
  --OUTPUT 'X11/filename' - printed to filename.eps (def X11);
                          if given 'filename'.eps is created
  -P <printer command>  - as -P lpr

 ENVIRONMENT VARIABLES
      GIR_PYTHON     - GIRAFFE BLDRS python modules directory
      PYTHONSTARTUP  - GIRAFFE BLDRS global initialization script
      GIR_DEVELOP    - use development version if defined
```

Python defaults

```
 Source: 124178 oct 16 10:10 /giraffe/blecha/BLDRS/girBLDRS1.09/python/share/bin/giCrossC.py
(Following default values could be obtained on-line using: giCrossC.py -z )
   I/Os
    frameNames: [None]
    cTableNames: ['girF0H9', 'girG2H9', 'girK0H9', 'girThArH9', 'girSkyMEDH9']
    cTableSpec:
                table BVmin BVmax class                    description object_selection
      --------------- ----- ----- ----- ------------------------------ ----------------
           girF0H9 -9.00  0.00     0                           star               ST
           girG2H9  0.00  0.00     0                           star               ST
           girK0H9  0.00  9.00     0                           star               ST
         girThArH9  9.00  9.00     2      Simultaneous Calibration              CAL
        girSkyMEDH9  9.00  9.00     3                            Sky   .*[sS][kK][yY]
    cImageFile:
    N         : c
    O(bject)  :
   General
    v(erbose) : 0
    d(ebug)   : 0
   outputNames:
   Radvel Options
    Tav       : 0
    TemplateN : 0:-0
    RV window [km/sec] : -100.0 100.0
    cstep [nm]: 0
    Vcorr     : b
    R(epeat) graph_at_each_step_flag,window_width_factor,max_repeat,lastWidthFactor: 0 1.5 3 0.0
    norm (boxsize): 0
    clean(boxsize): 10
         plot     :
    cdomain: 0.1:-0.1
   Simulation: 0
   Output+Graphs
    outputName: ['', '']
    txt,head  : 0 1
    txts,head : 1 1
    xf        : 0
```

```
 wresTable : None
 graph     :
 X graphlim: 0:-0
 titles    :
 HEIGHT       : 300
 WIDTH        : 1200
OUTPUTS      : ['X11']
P(rint)com   :
```

Chapter 3

Data Description

3.1 General description of Frames and Tables

3.1.1 Description of Frames

Frames used and produced during the reduction steps have different formats and types.
Each *format* defines a physical structure which has, for a given observing mode, always the same number of columns and rows. Note that the number of images within a frame is not defined by the format. This is controlled by the standard FITS KWs and is supported only at the level of raw and preprocessed frames.

Frame formats (table giFrameFormats.r)

Format	D	Nz	Nxm	Nym	Mt	Nxa	Nya	At	Nxi	Nyi	It	Description
xy	2	1	4200	2248	37.7	4200	2248	37.7	4200	2248	37.7	image with optionally prescan and overscan
xyPrep	2	1	4096	2048	33.5	4096	2048	33.5	4096	2048	33.5	image without prescan and overscan
nxExt	2	1	4096	137	2.2	4096	320	5.2	4096	320	5.2	image of any variable in [spectra running number,X] space
nlWbin	2	1	10000	137	5.4	10000	320	12.8	10000	320	12.8	image of any variable in [spectra running number,lamda] space

Each *type* correspond to a specific stage of the processing and identify in a unique way the meaning of data pixels." » /obs/ccd3/VLT/GIRAFFE/MAY99/doc/rel1/dd.FrameDescription.latex

Frame format table - Field description

Fieldname	Description
Nfor	
Format	format name
InputKWs	This is a list of all KWs in the header when frame is on input from archive or on the output as data product. It does not include KWs which may be added on during the processing.
SelectionKWs	list of KWs used to select the frame
D	number of dimensions
Nz	number of planes for all modes (always 1 in Medusa mode)
Nxm	number of rows in Medusa mode
Nym	number of columns in Medusa mode
Mt	size [MB] in Medusa mode
Nxa	number of rows in Argus mode
Nya	number of columns in Argus mode
At	size [MB] in Argus mode
Nxi	number of rows in Ifu mode
Nyi	number of columns in Ifu mode
It	size [MB] in IFU mode

Frame format table - Field description

Fieldname	Description
Description	full designation of the format

Frame types (table giFrameTypes.r)

Type	Format	Description	SelectionKWs
BDRMIMG	xy	bias and dark subtracted	
BIASIMG	xy	master bias	CCDID
RAWIMG	xy	raw CCD	
BADPIX	xyPrep	bad pixel mask	CCDID
BRMIMG	xyPrep	bias subtracted	
CRHIMG	xyPrep	Cosmic Ray Hit counts for each pixel	
DARKIMG	xyPrep	Master dark	CCDID
ESTIMG	xyPrep	estimate image (median, average or extraction)	
ESTSIGM	xyPrep	the standard error for each pixel after estimation	
PREPIMG	xyPrep	one of preprocessed image in any stage	
SLIMG	xyPrep	scattered light model	
DLOCY	nxExt	Master localisation + correction to the Y-coordinate of localisation lane	SLIT GRATING WLEN0
EXTERRS	nxExt	extracted spectra errors	
EXTLOCY	nxExt	Y-coordinate centroid of extrated data	
EXTNPIX	nxExt	extracted spectra pixel counts for each x bin	
EXTSP	nxExt	extracted spectra	
LOCD	nxExt	Correction to the Y-coordinate of localisation lane	SLIT GRATING WLEN0
LOCPSFAMPL	nxExt	Parameter of tranverse PSF	SLIT GRATING WLEN0
LOCPSFBACK	nxExt	Parameter of tranverse PSF	SLIT GRATING WLEN0
LOCPSFCENT	nxExt	Parameter of tranverse PSF	SLIT GRATING WLEN0
LOCPSFEXPO	nxExt	Parameter of tranverse PSF	SLIT GRATING WLEN0
LOCPSFWIDT	nxExt	Parameter of tranverse PSF	SLIT GRATING WLEN0
LOCWY	nxExt	Y-width of localisation lane	SLIT GRATING WLEN0
LOCY	nxExt	Y-coordinate of localisation lane	SLIT GRATING WLEN0
BINERRS	nlWbin	λ rebinned extracted spectra errors	
BINSP	nlWbin	λ rebinned extracted spectra	

3.1.2 Description of Tables

Table summary (table TAB/giTables.r+tblprep)

Type	Extension	Nxm	Nym	Mt	Nxa	Nya	At	Title
DLocc	.chebd.tfits	4	135	0	4	135	0	Chebyshev polynomial model of localisation correction obtained from 5 SIMCAL spectra
Grating	girGratingXXX.tfits	22	10	0	22	10	0	Grating Setup Constants
Line	girLineXXXX.tfits	2300	4	0	2300	4	0.1	Catalogue of emission lines used for Wavelength Calibration
Locc	.clocc.tfist	4	135	0.2	4	135	0.3	Chebyshev polynomial model of localisation
Ozpoz	in the image header	135	50	0	15	50	0	Positioner binary table
RVMask	girXXX.tfits	200	5	1.3	200	5	4.1	Binary mask for Correlation
SlitGeo	girSlitGeoXXX.tfits	135	12	0.3	320	12	1.3	Geometry of slits - position of fibers
WavCoef	girWavCoefXXX.tfits	36	1	0	36	1	0	Coefficients of the Polynomial Wavelength Solution

Table summary (table TAB/giTables.r+tblprep)

Type	Extension	Nxm	Nym	Mt	Nxa	Nya	At	Title
Wres	.wlenres.tfits	1	135	0	1	135	0	Wavelength spectra shift obtained from 5 SIMCAL spectra

Table types - Field description

Fieldname	Description
Name	table name
Type	table type (compulsory syntaxe: girXxx..Table)
Nxm	Number of rows in MEDUSA mode
Nym	Number of collumns in MEDUSA mode
Mt	size [MB] in Medusa mode
Nxa	Number of rows in ARGUS mode (default - same as for MEDUSA)
Nya	Number of collumns in ARGUS mode (default - same as for MEDUSA)
At	size [MB] in Argus mode
Title	Title of the table

3.2 Detailed Description of Tables

3.2.1 General remarks

The following section is generated from the development database. It insures that every FITS KW is uniquely defined in and traceable to the Data Dictionary (DIC) and that all data-structures used within the function design are defined in a consistent way. For the time being we reference DIC through an alias look-up table. The FITS KW described in *Table Header* subsections are referenced by their aliases and traced to the existing KW in ESO DICs (column *dicname*). We reference provisionally the DIC named *DRS* whenever we need a new KW created and maintained by the DRS and DIC named *MISSING* or FPMISS for *Fiber Positionner* whenever we failed to find a KW which we expect to receive from FLAMES OS.
The descriptions and comments are as complete as possible and similar formalism as for DIC presentation is used:

- If description/comments are present in both Alias definition and DIC definition files, the texts are prefixed *AL:* and *DIC:* respectively. If specific comment is given in table description it is appended with prefix *TAB:*.
- If description/comments are present only in Alias (usually new KW), there is no prefix.
- If description/comments are present only in DIC (usually ESO KW), there is a prefix *DIC:*
- The description is shortened to 160 characters (followed by + sign).
- If there is a conflict between DIC and Alias table concerning the *type* or *unit* both definition are printed separated by .

Note that in present python implementation, all mentionned KWs are not necessarily present in the header since the selection of the appropriate table instance made by the pipe script is often based on the filename rather then on header KWs. Also most of table exist in ASCII and FITS table format, the later one being generated from ASCII master. For more details see DATA REDUCTION COOK-BOOK in chapter 4

3.2.2 girDLocc - Chebyshev polynomial model of localisation correction obtained from 5 SIMCAL spectra

Purpose: Provide correction to master localisation
Reference Function: giAdjustLoc
Filename format/extention: .chebd.tfits

girDLocc: description

This temporary table provide the Chebyshev polynomial model of the localisation correction.
The localisation center is described by one 1-D model/spectrum. Each row of table gives coefficients for one spectrum.
Same information is available in image format (xxx.dlocy.fits)

girDLocc: working status

table name:	girDLocc
date:	Wed 10/08/03 10:19:56
status:	Unused
table source:	**current directory**
source comment:	Naming convention is XXX_SLIT__RES_dbSub.dlocc.tfits; computed by localisation giRecDLocfit.py
examples:	none

girDLocc: archiving technical informations

Total size is negligible (below 1MB)

girDLocc: Columns

Column	Description	Type	Unit	Comment
NSPEC	running number of the spectrum/fibre from 1 to 135(320)	integer		
COEFFI	Chebyshev polynomial coefficients for the correction of localisation	double		

girDLocc: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	type of table data	string	none	TAB: 'LINETAB'
GRATING	GIR_ICS	DIC: ESO name for grating unit.	string	none	TAB: Selection DIC: Grating common name. AL: HR or LR
WLEN0	GIR_ICS	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength	double	nm	TAB: Selection DIC: Grating central wavelength [nm].
DLPOLYDEG	DRS	degree of the 1-D polynomial fit of localisation correction	integer	none	
DLOCXYWID	DRS	Y:X dimension of the area where we fit the 2-D line-profile during the localisation of line	string	pixel	

3.2.3 girGrating - Grating Setup Constants

Purpose: Provide the setup dependent constants
Reference Function: giGetWaveSolution
Filename format/extention: girGratingXXX.tfits

girGrating: description

This table gives all informations specific to the spectrograph grating and setup (one row/setup) which necessary to the data reduction. Most of data are redundant to those used by the OS for the actual instrument setup and IT IS THEREFORE MANDATORY to updated this table whenever a modification in the available instrument setups occurs.
Two instances corresponding to low and high resolution grating exist.
The KW GRATING is used to select the table while KW WLEN0 is used to select the appropriate line.

girGrating: working status

table name:	girGrating
date:	20011115
status:	Most of setups validated
table source:	**GIR_CALIB**
source comment:	From Spectrograph optical design and wcalMast
examples:	girGratingHR316.tfits,girGratingLR600.tfits

girGrating: archiving technical informations

Total size is negligible (below 1MB)

girGrating: Columns

Column	Description	Type	Unit	Comment
SETUP	Index	none	of the setup	string
ORDER	Grating order	integer	none	
WLMIN	Shortest usable wavelength in the current setup	float	nm	
WLMAX	Longest usable wavelength in the current setup	float	nm	
WLEN0	Grating central wavelength	double	nm	Input - Lookup value of the KW WLEN0
BAND	WLMAX-WLMIN	float	nm	Redundant
RIFA	Resolution in IFU and ARGUS mode	integer	none	
RMED	Resolution in MEDUSA mode	integer	none	
FCOLL	optical model focal lengt	float	mm	
GCAM	optical model camera magnification factor	float	none	
THETA	Grating angle	float	degree	
SDX	optical model slit X (horizontal) shift	float	mm	
SDY	optical model slit Y (vertical) shift	float	mm	
SPHI	optical model slit tilt	float	radian	
LASTUPDATE		string		Date and name of the last update (manual)

girGrating: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	TAB: type of data type of table data	string	none	TAB: 'GRATINGTAB'
GRATING	GIR_ICS	TAB: grating name DIC: ESO name for grating unit.	string	none	TAB: one of HR316 LR600 DIC: Grating common name. AL: HR or LR
GRATGRV	GIR_ICS	DIC: Grating"s number of grooves per nanometer GROOVES is used in the conversion formula. ENC=ZORDER+ ALIGN+ RESOL* asin(WLEN* ORDER* GRV/(2* cos(ROT))) + TEMPRAMP* (TEMP-TEMP+	double	gr/nm	TAB: one of -0.000316 -0.0006 DIC: Grating grooves / nm [gr/nm].

3.2.4 girLine - Catalogue of emission lines used for Wavelength Calibration

Purpose: Provide emission line position and flux for wavelength calibration
Reference Function: giGetWaveSolution
Filename format/extention: girLineXXXX.tfits

girLine: description

This table, unique for all slits and setups, contains the laboratory calibration of a Wavelength Calibration source. The line selection within the table is based on the SLIT, WLEN0 and the $COMMENT$ anf $FLUX$ columns. If the column $COMMENT$ contains word with more then 3 characters (sufficient to design H1-H22 or L1-L8 setups) the line is cosidered as blended nad will be used only to prevent the selection of a close-by neigbour.
Light source could be Calibration lamp or SKY. Presently, only ThArNe table is provided. This is a static table, though some adjustement will have to be made each time the source lamp is changed. Note also that line-selection could probably be improved.
The accurate wavelengths are given while the flux accuracy is very crude. The later is used for line selection only.

girLine: working status

table name:	girLine
date:	20020107
status:	OK - could be improved for ThAr
table source:	**GIR_CALIB**
source comment:	Laboratory calibration provided by GIRAFFE DRS team.
examples:	girLineThAr

girLine: archiving technical informations

Total size is negligible (below 1MB)

girLine: Columns

Column	Description	Type	Unit	Comment
WLEN	Wavelength of the line	float	nm	Central wavelength of the line
NAME	Name of line	string		If any
FLUX	Expected central flux/[pixel.sec]	float	e	Mode and Resolution dependent (See table description)
COMMENT	Setup followed by cotrolled vocabulary comment	string		Must have format: Hn[n]/Ln [comment]; see table description

girLine: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	type of table data	string	none	TAB: 'LINETAB'
LAMP	DRS	Name of the W Calibration source	string	none	One of ThAr,Ne,ThArNe or 'SKY' (telluric lines)
LAMPID	TC_ICS	DIC: Reference light lamp identifier.	string	none	DIC: LAMP unique ID

3.2.5 girLocc - Chebyshev polynomial model of localisation

Purpose: Provide master localisation model

Reference Function: giLocalSpectra
Filename format/extention: .clocc.tfist

girLocc: description

This table, one for each set in each all slit, provide the Chebyshev polynomial model of the localisation.
The localisation center is described by one 1-D model/spectrum. Each row of table gives coefficients for one spectrum.
The localisation half-width is given by unique 2-D model defined through LOCWCOEFF coefficients in the table header.

girLocc: working status

table name:	girLocc
date:	Wed 10/08/03 10:19:56
status:	Unused
table source:	**current directory**
source comment:	Naming convention is XXX_SLIT__RES_dbSub.clocc.tfits; computed by localisation giRecNLoc2.py
examples:	none

girLocc: archiving technical informations

Total size is negligible (below 1MB)

girLocc: Columns

Column	Description	Type	Unit	Comment
NSPEC	running number of the spectrum/fibre from 1 to 135(320)	integer		
COEFFI	Chebyshev polynomial coefficients for localisation Y position	double		

girLocc: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	type of table data	string	none	TAB: 'LINETAB'
GRATING	GIR_ICS	DIC: ESO name for grating unit.	string	none	TAB: Selection DIC: Grating common name. AL: HR or LR
WLEN0	GIR_ICS	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength	double	nm	TAB: Selection DIC: Grating central wavelength [nm].
LPOLYDEG	DRS	degree of the polynomial fit	integer	none	
LOCYDEG	DRS	degree of the x-polynomial fit of localisation	integer	none	
LOCWDEG	DRS	degree of the x,y-polynomial fit of localisation half-width	integer	none	
LOCWCOEFF	DRS	x,y-polynomial fit coefficients of localisation half-width	double	none	TAB: LOCWDEG*2 coefficients

3.2.6 girOzpoz - Positioner binary table

Purpose: Provides fibre assignation and configuration
Reference Function:
Filename format/extention: in the image header

girOzpoz: description

Updated from data taken during the Science Verification. Please, refer to your data for comlete list of KWs. (Obsolte: This is the FITS binary table of the positioner, part of the input *rawFrame*, describing fibre configuration see "FLAMES FIBRE POSITIONER, Instrument Control Software Design Description" INS-SPE-AUS-13721-0042 for more details.)

girOzpoz: working status

table name:	girOzpoz
date:	20000613
status:	From SV data
table source:	**any image**
source comment:	Binary table, extension 2 of the rawImage and present in all data products
examples:	

girOzpoz: archiving technical informations

Total size is negligible (below 1MB)

girOzpoz: Columns

Column	Description	Type	Unit	Comment
OBJECT	Name or identifier of the observed object	string	none	
PRIORITY	object's allocated priority	integer	none	
ORIENT	Pivot-to-button angle on the plate	float	radians	
IN_TOL	Flag if fibre placed within tolerance	logical		'T' or 'F'
MAGNITUDE	object's magnitude	real	none	
RA	J200 mean right ascension	double	degree	
DEC	J200 mean declinaison	double	degree	
R	R position of the fibre	float	microns	
R_ERROR	error in R	float	microns	
THETA	theta position of the fibre	float	degree	
THETA_ERROR	error in theta	float	degree	
TYPE	fibre type: F=FACB, I=IFU, T=test, U=UVES, M=Medusa	character	none	FIBRE_TYPE=T refers to fibres allocated to internal positioner test fibres
BUTTON	positioner button number	integer	none	Equivalent to `girSlitGeo` :*RP*

girOzpoz: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	type of table data	string	none	TAB: 'OZPOZTAB'
CENRA	FPMISS	field centre right ascension	real	radians	input value of P2PP
CENDEC	FPMISS	field centre declination	real	radians	input value of P2PP

girOzpoz: Header

Alias	DIC	Description	Type	Unit	Comment
CENEQNX	FPMISS	equinox of the above (FK5 Julian)	real	none	input value of P2PP
APPRA	FPMISS	apparent right ascension	real	radians	value calculated by P2PP
APPDEC	FPMISS	apparent declination	real	radians	value calculated by P2PP
APPEPOCH	FPMISS	epoch of the above	real	none	value calculated by P2PP
CONFMJD	FPMISS	MJD for which the field was configured	real	none	
ACTMJD	PR_FITS	DIC: Obs start AL: MJD at which the field was observed	double	days	DIC: Obs start

3.2.7 girRVMask - Binary mask for Correlation

Purpose: Provide mask defintion used to obtain Radial Velocity by Correlation
Reference Function: giCrossC
Filename format/extention: girXXX.tfits

girRVMask: description

Tables define the masks used to correlation with extracted, wavelength rebinned spectra. The mask is generated as binary structure with ones for WLEN1 $\leq \lambda \leq$ WLEN2. The column DEEP is included for possible future use as weight. WIDTH and WLENCENTER are derived quantities included for confort of graphical inspection only.

girRVMask: working status and source informations

table name:	girRVMask
date:	Wed 02/05/03 10:19:09
status:	OK
table source:	**archive**
source comment:	Naming convention is gir_Sptype__RES_.tfits
examples:	girF0H9.tfits

girRVMask: archiving technical informations

comment concerning the size:		ThAr all setups; for 3 spectral star type F0, G2, K0 H1-15 L1-5; Sky for H10-22 L4-8
total number of tables in MEDUSA mode:	174	
size per table	0.01	MBytes
total size of tables in MEDUSA mode:	**1.74**	MBytes
total number of tables in ARGUS/IFU mode:	0	
size per table	0.00	MBytes
total size of tables in ARGUS/IFU mode:	**0.00**	MBytes
Total for all modes	**1.74**	MBytes

girRVMask: Columns

Column	Description	Type	Unit	Comment
WLEN1	wavelength	float	nm	Beginning of the spectral window
WLEN2	wavelength	float	nm	End of the spectral window
DEEP	relatif weight of the window	float	none	
WIDTH	wavelength	float	nm	Width of the window
WLENCENTER	wavelength	float	nm	Center of the window

girRVMask: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	TAB: type of data type of table data	string	none	TAB: 'RVMASKTAB'
GRATING	GIR_ICS	DIC: ESO name for grating unit.	string	none	DIC: Grating common name. AL: HR or LR
WLEN0	GIR_ICS	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength	double	nm	DIC: Grating central wavelength [nm].

3.2.8 girSlitGeo - Geometry of slits - position of fibers

Purpose: Provide the slit fibre position used for localisation and PSF modeling, wavelength solution and rebinning.
Reference Function: giCrossC
Filename format/extention: girSlitGeoXXX.tfits

girSlitGeo: description

This table gives the description of the slit geometry. The table selection is made by the match on KW SLIT. It is necessary to establish the wavelength calibration. This is a static table provided initially by the mechanical (columns NS, NSUBSLIT, XF, YF and ZF) and optical (ZFDEFOCUS) designs with possible adjustment during the commissioning. The column BUTTON is the only link between the physical geometry of the slit and the positionner. The BUTTON value gives access key to the informations contained in the `ozpoz` table concerning the specific fibre. Fibres and slits are numbered starting from the fibre with most positive XF coordinate (presumably the top of the slit).

girSlitGeo: working status and source informations

table name:	girSlitGeo
date:	20011108
status:	Operational within Python version
table source:	**GIR_CALIB**
source comment:	Naming convention is gir_SLIT__RES_.tfits Table is produced by wcalMast.
examples:	girSlitGeoMedusa1H21.tfits, girSlitGeoIFU1L4.tfits

girSlitGeo: archiving technical informations

comment concerning the size:		
total number of tables in MEDUSA mode:	58	
size per table	0.01	MBytes
total size of tables in MEDUSA mode:	**0.58**	MBytes
total number of tables in ARGUS/IFU mode:	87	
size per table	0.02	MBytes
total size of tables in ARGUS/IFU mode:	**1.74**	MBytes
Total for all modes	**2.32**	MBytes

girSlitGeo: Columns

Column	Description	Type	Unit	Comment
NSPEC	running number of the spectrum/fibre from 1 to 135(320)	integer		
RP	running number of positionner retractor	integer		This is the only link between the physical geometry of the slit and the positionner. The `girSlitGeo`:RP = `OzPoz`:$BUTTON$ gives access key to the informations co+
STATUS	numerical code giving the permanent fibre status (1 for valid fibre, 0 unused)	integer		Other codes could be used in the future
COMMENT	any comment concerning this fibre	string		
NSUBSLIT	running number of the subslit from 1 to 13(15)	integer		
XF	Almost constant fiber position in x (perpendicular to the slit)	float	mm	center of the slit is [0,0,0]
YF	Almost regularly decreasing fiber position in y (along the slit)	float	mm	
ZF	Fiber position in z on the slit	float	mm	control and visualisation; Reference point is focal surface at the center of the slit [XF=0,YF=0] TBC
ZFDEFOCUS	Fiber distance from the focal plane	float	mm	used for simulation; + toward the spectrograph
XIND	Fiber position in x in focal plane image	integer	pixel	Image reconstruction; Reference point TBD
YIND	Fiber position in y in focal plane image	integer	pixel	Image reconstruction; Reference point TBD
NFSUB	Fiber running number within the subslit	integer		1 on the top

girSlitGeo: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	type of table data	string	none	TAB: 'SLTGEOMTAB'
SLIT	GIR_ICS	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus	string	none	DIC: Slit name.
GRATING	GIR_ICS	DIC: ESO name for grating unit.	string	none	DIC: Grating common name. AL: HR or LR
WLEN0	GIR_ICS	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength	double	nm	DIC: Grating central wavelength [nm].

3.2.9 girWavCoef - Coefficients of the Polynomial Wavelength Solution

Purpose: Provide and save wavelengt solution
Reference Function: giGetWaveSolution
Filename format/extention: girWavCoefXXX.tfits

girWavCoef: description

This table contains the coefficients of the standard wavelength solution for each GIRAFFE slit and setup. The table selection is made by the match on KWs SLIT, GRATING and WLEN0. The header KWs give the parameter of the optical model while the table provide coefficients of the Chebytchev correction. There is 1 line/subslit. Currently a full slit solution is implemented and therefore the table has just 1 line. The coefficient are given in the order ...GILLES.
Note that the complete wavelength solution is defined by this table and the girSlitGeo table and, optionaly, the temporary table ...wlenRes.tfits (simultaneous calibration adjustement).

girWavCoef: working status

table name:	girWavCoef
date:	20000605
status:	OK for most setups
table source:	**GIR_CALIB**
source comment:	Naming convention is girWavCoef_SLIT__RES_.tfits Table is produced by wcalMast.
examples:	girWavCoefH9Medusa2.tfits

girWavCoef: archiving technical informations

Total size is negligible (below 1MB)

girWavCoef: Columns

Column	Description	Type	Unit	Comment
SUBSLIT	Running number of the subslit	integer	none	set 0 for full slit solution
CIJ	Coefficients of the wavelength solution (for the residuals relative to the optical model)	double	nm	same value for a given subslit (residuals)

girWavCoef: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	type of table data	string	none	TAB: 'WSOLTAB'
SLIT	GIR_ICS	DIC: Slit name. AL: one of Medusa1 Medusa2 IFU1 IFU2 Argus	string	none	DIC: Slit name.
GRATING	GIR_ICS	DIC: ESO name for grating unit.	string	none	DIC: Grating common name. AL: HR or LR
WLEN0	GIR_ICS	DIC: Wavelength that the grating transmits along the optical axis of the instrument. AL: grating central wavelength	double	nm	DIC: Grating central wavelength [nm].

girWavCoef: Header

Alias	DIC	Description	Type	Unit	Comment
FOCLEN	DRS	focal length of the collimator	double	m	
CAMGFACT	DRS	magnification factor of the camera	float	none	Name changed from GCCDSCALE to CAMGFAC
GRATROT	GIR_ICS	DIC: Rotation angle of grating. The reference frame is given in the corresponding instrument specification. The rot angle ROT is used in the conversion formula: ENC=ZORDER+ AL+	double	deg	DIC: Grating rot angle [deg].
GSPACE	-				
WSWINDOW	DRS	width in pixel for line detection and fit	float	pixel	
WSWLENXDEG	DRS	polynomial degree along the x direction for wavelength solution	integer	none	
WSWLENYDEG	DRS	polynomial degree along the y direction for wavelength solution	integer	none	
WSWSOLUTION	DRS	flag 1/0 fit optical parameters or residuals t	integer	none	
WSWMODEL	DRS	'xoptmod2' dispersion optical model name	string	none	
WSWDIRECTION	DRS	1/0 optical model dispersion direction	integer	none	
WSWSUBSLITFIT	DRS	0 / slit geometry fit of subslit	integer	none	
WSWFCOLL	DRS	optical model focal lengt	float	mm	
WSWGCAM	DRS	optical model camera magnification factor	float	none	
WSWTHETA	DRS	optical model grating angle	float	radian	
WSWSLITDX	DRS	optical model slit X (horizontal) shift	float	mm	
WSWSLITDY	DRS	optical model slit Y (vertical) shift	float	mm	
WSWSLITPHI	DRS	optical model slit tilt	float	radian	
WSWLINEFILE	DRS	girLineXXX.tfits reference lines table name	string	none	
WSWLINEMODEL	DRS	'psfexp' / line profile model name	string	none	
WSWLINETYPE	DRS	lines for wavelength solution	string	none	
WSWLINE	DRS	width in pixel for line detection and fit	float	pixel	
WSWLINETHRESHOLD	DRS	line detection threshold	float	none	
WSWLINENITER	DRS	50 / number of iterations line detection and	integer	none	
WSWLINENTEST	DRS	7 / number of chisquare test line detection a	integer	none	
WSWLINEDCHISQ	DRS	0.0001 / delta chisquare line detection and	float	none	
WSWXRCLIPSIGMA	DRS	Chebytshev corection sfwsigclip:multiple of sigma	float	none	
WSWXRCLIPNITER	DRS	Chebytshev corection psfwsigclip:number of iterations	integer	none	

girWavCoef: Header

Alias	DIC	Description	Type	Unit	Comment
WSWXRCLIPMFRAC	DRS	0.9 / Chebytshev corection psfw-sigclip:min fraction of accept	float	none	
WSWXRPOLYDEG	DRS	'7:5' / Chebytshev corection 'XDEG:YDEG' polynom order for	string	none	

3.2.10 girWres - Wavelength spectra shift obtained from 5 SIMCAL spectra

Purpose: Adjust wavelength solution before final rebinning
Reference Function: giCrossC
Filename format/extention: .wlenres.tfits

girWres: description

This temporary table created in the working directory by cross-correlation of simultaneous calibration rebinned spectra and interpolation for all spectra of the current frame. It reflects spectra shift compared to the standard solution.
The complete wavelength solution is defined by girWaveCoef table, the girSlitGeo table and, this table.

girWres: working status

table name:	girWres
date:	Thu 10/02/03 14:51:44
status:	OK
table source:	**current directory**
source comment:	Computed by cross-correlation with ThAr mask
examples:	none

girWres: archiving technical informations

Total size is negligible (below 1MB)

girWres: Columns

Column	Description	Type	Unit	Comment
WLRES	Wavelength shift of the corresponding spectrum	float	mm	$\lambda_{corrected} = \lambda_{row} + WLRES$

girWres: Header

Alias	DIC	Description	Type	Unit	Comment
TITLE	DRS	Title of data structure	string	none	Standard table KW
PURPOSE	DRS	Purpose of data structure	string	none	Standard table KW
AUTHOR	DRS	Source of data (responsible or processing recipe)	string	none	Standard table KW
DATE	PR_FITS	DIC: Date this file was written	string	none	DIC: Date this file was written
STATUS	DRS	Comment on last update	string	none	Standard table KW
GIRTTYPE	DRS	type of table data	string	none	TAB: 'WRESTAB'

Chapter 4

Processing recipes (pipes)

4.1 INTRODUCTION

(Source file **/net/obssb55/export/diskB1/giraffe/SV/pipe/doc/README.intro**)

```
INTRODUCTION

  DISCLAIMER

    The integration of the BLDRS into VLT-compliant environment was from the very
    beginning of the FLAMES project under ESO responsibility. Following ESO suggestion we
    use Python environment as prototypic environment and we have set up few tools to make
    automatic processing simple and flexible in the development and commissioning
    environment. When we have been asked to provisionally make our tools available to
    astronomers, we stitched together test scripts and we added crude GUI. We did not
    spend any time cleaning the scripts and the python code and we tested the script
    portability on few systems only. For us the focus was and remains on C-code testing on
    real data.

  CALIBRATION STATUS

    This first release should do a fair data processing in most cases. Note that it makes
    a big difference to start from scratch (no standard calibration available) or from
    standard calibration (standard localization and W-solution available). Unfortunately,
    to this date, we have only very limited calibration dataset. As soon as the
    calibration will be available we will upgrade the distributed calibration database.
    Specifically the present release was not tested with data from ARGUS mode.

  COMMING NEXT

    We expect to upgrade the pipe at rapid pace. Following elements are coming in a matter
    of 1-2 months:

    - argus validation
    - sky subtraction

  MAINTENANCE and BUG correction policy

    We will be interested by any report of your problems but we could not help in
    installation difficulties linked to your local environment or personal setup.
    Concerning the problems linked to the data-reduction itself please send the detailed
    description with at least:

    full offending command (pipe AND python function call)
    cut-and-paste of error message

    In case of unexpected or wrong results we may want to have a complete copy of relevant
    input data (these listed with flag -t and actual pipe parameters). If you have not
    your FTP site for such purpose we will give you instructions where to send it.
```

```
  PLEASE do not send huge amount of data by e-mail (some graphs may be several Mbytes).

HOW to use this documentation

  You should [you may] read paragraphs (and run examples whenever possible):

  GENERAL INFORMATIONS ON PIPES
  DATA REDUCTION COOK-BOOK
  CALIBRATION REDUCTION COOK-BOOK
  [ RDB COOK-BOOK ]

  Following pipe and python function description is better understood when running
  specific pipes with '-t' flag and specific data (RAW=xxx) and options. Since available
  parameters are option-dependent some parameters (typicaly PLOTOPT,PLOTOUT) could remain
  hidden.

  All the documentation is available on line through various calls to the script 'pipe'


  Last update AB  Wed Sep 17 12:17:51 MEST 2003
```

4.2 GENERAL INFORMATIONS ON PIPES

(Source file **/net/obssb55/export/diskB1/giraffe/SV/pipe/doc/README.pipe**)

```
COMMAND-LINE PIPE
  1.03 (Fri May 30 07:10:52 CEST 2003)

  PREREQUISITES:

    - girBLDRS installed
    - gnu UNIX commands:        gawk cat touch sort join uniq head tail
      (check using     : which  gawk cat touch sort join uniq head tail)

  INFOS + HELP :

    pipe -h               # help
    pipe -l               # short list of pipes
    pipe -t name_of_pipe  # show short description of the specified  pipe

  TEST

    Check that pipe was correctly installed. If any test a.-c. fail it is useless to
    continue, you must check the prerequistes.

     a.  pipe -l         # will display a short list of available recipes
     b.  pipe  test      # will test pipe execution and prints following line

       You should have GIRAFFExxxNGCnnnn.fits after : GIRAFFExxxNGCnnnn.fits

     c. pipe -t test  # you should have following output

     | Pipe definition files used: your_installation_path/pipe/config/t0.lst
     |                         and: your_installation_path/config/pipe0.r
     |
     | Command:  lt < $tfile \
     |         | ficom test,_TESTNAME_ \
     |         | rr RAW=_RAW_ NAM=_NAM_ \
     |         | pyrectest - -v \
     | (Begin source: file pipeD0.lst/formated by pipe.fmt.txt)
     | PIPE parameters
     |
     |   Name     : test
     |   Section  : Test
     |   Purpose  : test pipe name replacement
```

```
   |
   |   Priority : Any
   |
   |   Inputs:  file_name - relevant_pythone_function(internal_nu)
   |
   |     01 INPUT1                                          -               test (0)
   |     02 INPUT2                                          -               test (0)
   |
   |   Outputs: file_name - relevant_pythone_function(internal_nu)
   |
   |     01 OUTPUT1                                         -               test (0)
   |     02 OUTPUT2                                         -               test (0)
   |
   |   Quality control
   |
   |
   |   Examples executables from ..../NGC2243/reduced  directory (data in ..../NGC2243/raw assumed)
   |
   |     pipe test RAW=GIRAFFExxx NAM=NGCnnnn
   |
   |   Options: nu name[default]            (possible values) - short description
   |
   |      01 TESTNAME=[test1]               (test1) -  pipe test
   |
   |   Parameters: nu name[default]         (possible values) - short descriptions
   |
   |      01 NAM=[NGCnnnn]                  (Any name) -  last part of the filename
   |      02 RAW=[GIRAFFExxx]               (Filename) -  Full name of the RAW input image (.fits ...
   |
   | (End source: file pipeD0.lst/formated by pipe.fmt.txt)
   |
   | command
   | -------
   | echo You should have GIRAFFExxxNGCnnnn.fits after : GIRAFFExxxNGCnnnn.fits

Run pipe

  Before the execution it is recommended to run:

    pipe -t name_of_pipe

  To   execute, remove  the  '-t'  option.  Note  that  you  can  also  cut-and-paste
  commands displayed by 'pipe -t name_of_pipe'

    pipe [-t/T/l] name_of_pipe OPT2=opt2 PAR1=par1 OPT1=opt1 ... PAR3=par3 PAR2=par2

  name_of_pipe must come first just after flags; PARi-s and OPTi-s in any order

  OPTs Control the data flow (skip or include some processing). Note that PARs (and also
       its defaults values) and I/Os displayed with -t/T flags are OPTION-dependent.

  PARs control individual processing steps

       Some  parameters  are  intended  for  expert tuning only. Execept for preliminary
       calibrations,  you  should  run  'extract'  pipeline  which  do the full standard
       data-processing, with default parameters

  BEWARE: command are CASE SENSITIVE

    Most of options are given in the format 'OPT=opt' or 'OPT=no' where the string 'opt'
    is just the lower-case copy of 'OPT'

I/O data

  List of I/Os is displayed upon 'pipe -t ....'
```

```
    PUTZ option

      Some  pipes+options  could  produce  several  tens  of  control+garbage files. It is
      recommended  to  use  the  option 'PUTZ=putz' which removes all the garbage upon the
      completion  of  the  processing.  Note that you must not use this option if you want
      re-execute some individual commands later.

    CLEAN option !!! Beware !!!

      This removes all data in the current directory except README.r file (use for test only)

 CONFIGURATION DATA and SCRIPTS

  $GIR_PIPE/config contains files defining the pipes:

      t0.lst
      pipe0.r

  $GIR_PIPE/doc contains mostly documentation and formating files

      format files *fmt*
      README.*
      doc*

  $GIR_ROOT/share/scripts  contains  all  scripts necessary to run the pipe + some utilities
  scripts. It must be the first directory in your path.

 SPURIOUS GARBAGE

  Pipe  when  interrupted could leave 'tmp.pipe.[1-9][0-9]*' files in current directory;
  do not hesitate remove them. The script 'putz' removes most of it.

GUI - Graphical User Interface (provisional)

  PREREQUISITES: Perl Tcl/Tk installed

  The  GUI  is  intended to help to understand pipe functions and parameters. It is useful
  when 'tuning' usage of pipe for specific situation. For the mass production we recommend
  to  prepare  the  files  with  command-line calls to pipe and 'source' it (see also data
  handling)

  example below are given for pipe 'locMast' - testData/reduced (NGC2243)

  cd  testData/reduced
  girp.pl &             # when called 1-st time it takes some time to open

    click (thumbnail) 'locMast'             # shows options for master localization and
                                            # parameters associated to default options
    point to window of specific parameter   # shows detailed description of the pram.

    click help                              # show help for pipe

      fill Option/Params for test data

        MAST   no
        LOCPSF no
        RAW    ../raw/GIRAF.2003-01-28T10:07:04.309.fits

    click 'list Commands'      # see commands TB executed within the pipe

      Note  that  all  parameters of commands have been set to default or assigned values.
      YOU  MUST  NOT  SEE any parameter enclosed by '_' like _XXXX_ when executing. Since
      the  available  parameters  and  help are both OPTION-dependent, when you change any
      option, you must click the button 'List Commands' to update the GUI panel.
```

```
    click 'Run current pipe'   # it executes all commands

    click 'Run' left to 'biggLoc' command   # executes specific command (input data must exists)
    click 'Helps and Logs' in front of executed command, modify parameters and
          re-execute command

  Known drawback of GUI:

    pipe can not be interrupted
    GUI is 'dead' (becomes blank if moved) during the execution

  Tested

     Red Hat Linux 8.0 3.2-7 / perl v5.8.0 built for i386-linux-thread-multi
```

4.3 DATA REDUCTION COOK-BOOK

(Source file **/net/obssb55/export/diskB1/giraffe/SV/pipe/doc/README.data**)

```
DATA

 Data Organisation

  We recommend following data organization

    data -| object_or_day#1 | raw
          |                 | reduced
          | object_or_day#2 | raw
          |                 | reduced
          |                 | ...
          ...
   [allreduced]

   If  you  have  large  set  of  data with several 'raw' directories you would better also
   made directory 'allreduced' from which you can run mass data reduction with comfort.

   . Fill ../raw directory with your data
   . From all 'reduced' directory run command

     dfits ../raw/*.fits* | fhead > README.r ; tro < README.r > README.txt

     README.r/txt  will  be  ASCII  comprehensive summary of all your data+KWs; with small
     character set it should make 1 line/image (see 'rdb' files for handling '.r' format).
     For more details see 'fhead -h' and/or 'dfits -h'

    The  README.r  is  ASCII file with TAB separated fields and 2-line header. This is the
    'rdb'  format  which  is  supported  by  most  of  utility  scripts.  To  have  nice
    column-aligned output you format with

      tro < README.r

    'rdb'  files  could  be  manipulated  using  both unix and/or specific rdb commands

  . Run pipe from 'reduced' directory.

    The  pipe  mimics  what  is  expected  the  VLT  pipeline  will  do. Most of necessary
    calibration  data  (all  excepted  girSlitGeo,  girLine,  girGrating,  girXX...-masks,
    girWavCoef) are copied from GIR_CALIB [/FF] and pipe mostly runs on local data.

    Local calibration data may be modified during the processing.

    Unless  'MAST'  option is used all output data go to the local directory. All data are
    fully typed by structured filename which is formatted as follows:
```

```
    _OBJ__SLIT__RES__NAM_processing_suffix1processing_suffix2....[t]fits

    The  table  below  summarizes  suffix  of  final  data  products  (not  removed  by
    'PUTZ=putz' option)

    suffix          Processing
    ----------      ---------------------------------------
    Mast            standard calibration
    dbSub           preprocessing
    .clocy/w        RAW localization position/width
    .plocy/w        PSF localization  -"-
    .psfxxx         PSF localization widt,expo,cent
    .e/oxtsp        extracted spectra SUMM/OPTIMAL method
    .e/oxtsperr     associated error
    .rebinlin       rebinned linear
    .rebinlog       rebinned logarithmic

  By  default, _OBJ_, _SLIT_, _RES_ are derived from image KWs and _NAM_ is omitted. For
  more information on _OBJ_ _SLIT_ _RES_ _NAM_ do 'pipe -T any_pipe'

DATA HANDLING and MANAGEMENT

 Files names from VLT archive do not carries information of data content. It is therefore
 mandatory  to  maintain  a  look-up table where some relevant data information are kept.
 This is done by running from 'reduced' directory only once (and each time the content of
 'raw' directory changes):

   dfits ../raw/*.fits* | fhead > README.r

 If you have 'allreduced' directory (large set of data with several 'raw' directories) you also
 construct README.r file at that level. You will need do something like:

   cd .../allreduced
   dfits ../*/raw/*.fits* | fhead > README.r

 Note  that  'fhead'  calls  the  python  script  'fitshow' which provides VERY POWERFUL
 general  conversion  of  'dfits'  output  to 'rdb' file (see also 'fitshow -h'). You may
 produce  and manipulate table of any KWs of any set of FITS images.

   dfits -x1 *.oxtsp.rebinlin.fits | fitshow '*REBIN*' | tro

 Note that KWs produced by DRS are in 1-st extension. You can see all DRS KWs with:

   dfits -x1 `ls *.oxtsp.rebinlin.fits|head -1` | egrep "OGL PRO"

 or with shorter 'alias' names of all KWs relevant to the DRS

   dfits -a -x1 `ls *.oxtsp.rebinlin.fits|head -1` | egrep -v ESO

 Few more examples:

  dfits -a -x1 *.oxtsp.rebinlin.fits | fitshow "*BIN*" | tro        # rebinning prams
  dfits -a -x1 *.oxtsp.rebinlin.fits |egrep -v ESO|fitshow "E*"|tro # extraction

 The  MASS  DATA REDUCTION is facilitated by few scripts described below. Note that those
 operate on README.r file in the current directory which is used as look-up to data.

   showf # get table of specific files

     showf  W   R  | tro   - all raw W-calibrations sorted by setup
     showf  W   RU | tro   - same but only 1 file/setup
     showf FF   R  | tro   - all raw FF-calibrations
     showf SCIT R  | tro   - all science exposures with calsim
     showf SCI  R  | tro   - all science exposures without calsim
     showf FF   CC | tro   - all FF in $GIR_CALIB compared to raw FF files in 'raw'
```

```
    showf W   CC | tro   - all FF in $GIR_CALIB compared to raw FF files in 'raw'

  row #- row selection
    if you may prefer direct control on the selection using the 'rdb' command 'row'

    row 'UTYPE=="FF"' < README.r | sorttable SLIT FILT | tro

  fsplit  # split structured filenames and make 'rdb' table with SLIT and FILT columns

    Localization, W-solution and FFs we have in CALIB:

      ls -1  $GIR_CALIB/FF/*.clocy.fits   | fsplit | sorttable SLIT FILT | tro
      ls -1 $GIR_CALIB/*Coef*.tfits       | fsplit | sorttable SLIT FILT | tro
      ls -1 $GIR_CALIB/FF/Fff*.extsp.fits | fsplit | sorttable SLIT FILT | tro

    Extracted rebinned spectra

      ls -1 *.extsp.rebinlin.fits | fsplit | sorttable SLIT FILT | tro

  files [-n] # convert table to list of files

    the  following  will  take  first  3  raw FF files of specified setup and produce new
    master localization

      pipe -t locMast RAW=''UTYPE=="FF"&&SLIT=="Medusa1"&&FILT=="H14"' < README.r | files -3'

  compute # modify table content

    The following will prepare the processing of all science data covered by the current
    README.r

     showf SCIT|compute 'File="pipe extract NAM="SEQ " RAW="File' | column File | headoff > SCIT.src

    'source SCIT.src' will make data reduction of all science exposures

SPURIOUS GARBAGE

 Pipe  when  interrupted could leave 'tmp.pipe.[1-9][0-9]*' files in current directory;
 do not hesitate remove them. The script 'putz' removes most of it.
```

4.4 CALIBRATION REDUCTION COOK-BOOK

(Source file **/net/obssb55/export/diskB1/giraffe/SV/pipe/doc/README.calib**)

```
CALIBRATION COOK_BOOK

  CALIBRATION strategy

    We  suggest to carry on all calibrations for all your setups, prior to run the science
    data  reduction.  If  you  want  to  use  specific  calibration  files  for a specific
    scientific  target  you  may  create  several  calibration  directory as a copy of the
    standard one and switch between them using the environment variable $GIR_CALIB. In the
    following  we  describe standard calibration to be done in routine work when the basic
    Master  calibrations  are  available  (this  is  currently the case for setups used in
    science  verification).  More  detailed  description  is  given  under  description of
    calibration pipes biasMast,locMast and wcalMast.

  CALIBRATION DATA

    Naming convention

     Tables are named as follows: GenericName[_OBJ_]_SLIT__RES_.(tfits/r/txt)
     where _SLIT_ = (Medusa/IFU)(1/2) or Argus
           _RES_  = H1-H22 or L1-L8
```

```
  Only '.tfits' version of tables are used for processing though most of them are
  generated from pure ASCII '.txt' or '.r' rdb source.

 Location

  $GIR_CALIB -| xxxMast.fits                      # Setup independent Master images
              | girWavCoef_SLIT__RES_.tfits       # Master W-solutions
              | girSlitGeo_SLIT__RES_.tfits       # Master slit geometry adjustment
              | girLine_OBJ_.tfits                # Spectra lines where _OBJ_: ThAr,Sun
              | gir_OBJ__RES_.tfits               # CC masks where _OBJ_: ThAr,F0,G2,K0
              | girGrating(H/L)R(316/600).tfits   # Initial prams of optical model
              |
              | FF  .  .  .  .  .  .  .  .  .  . -| Fff__SLIT__RES_Mast.fits
                                                   | _SLIT__RES_Mast.cloc(y/w).fits
                                                   | _SLIT__RES_Mast.psf(ampl/expo/widt).fits


  To see ordered list of all instances:

    (cd $GIR_CALIB ; ls -1 girSlitGeo*.tfits) | fsplit | sorttable SLIT RES | tro
    (cd $GIR_CALIB ; ls -1 girWav*.tfits) | fsplit | sorttable SLIT RES | tro
    ...
    (cd $GIR_CALIB/FF ; ls -1 *Mast.clocy.fits) | fsplit | sorttable SLIT RES | tro
    ...
  This could be also done with more confort using:

    showf W CC [1] | tro       # option 1 makes new README.r (see below)

 Calibration Data selection

  Only '.tfits' tables are used for processing ('.r' version table being in some cases
  a  source  table for '.tfits'). Last table with appropriate GenericName and matching
  fits KWs is used. To force use of a specific table do: 'touch table_name'

BIAS calibration

 In this description the standard readout mode is assumed.

 ESO DET READ CLOCK = '225kpx_1x1_LG' / Readout clock pattern used
 ESO DET CHIP NX = 2148 / # of pixels along X
 ESO DET CHIP NY = 4096 / # of pixels along Y
 ESO DET OUT OVSCX = 50 / Overscan region in X
 ESO DET OUT OVSCY = 0 / Overscan region in Y
 ESO DET OUT PRSCX = 50 / Prescan region in X
 ESO DET OUT PRSCY = 0 / Prescan region in Y

 Under  these conditions the standard biasMast could be used and it is not necessary to
 produce  new  master. Note that the bias subtraction is based on differential polynomial
 model  built on PRE/OVER scans and active area. During the bias subtraction, the model
 is  adjusted  based on the model built on PRE/OVER scans of the science exposure and the
 biasMast  is  used  only  on  bad  column  where  mask is set.

 Maintenance

  For more details see pipe 'biasMast'

MASTER LOCALIZATION (locMast)

 See your calibration data (assumed README.r exists in current directory)

  showf FF | tro

 Check that the master localization exists for your set-up.
```

```
     (cd $GIR_CALIB/FF ; ls -1 *Mast.clocy.fits) | fsplit | sorttable SLIT RES | tro
     You should have a file with RES column = FILT KW of your set-up

   If we have the master, run:

     pipe [-t] locMast RAW=FFRawFile LOCRAW=no

     this  does  everything  and  produces all necessary masters; do not hesitate to run it
     with  option  MAST=no  and  compare  masters  created  in  the  current  directory  to
     $GIR_CALIB/FF.

   else drop option 'LOCRAW=no'

   Maintenance

     For more details see pipe 'locMast'

  MASTER Wavelength calibration (wcalMast)

   See your calibration data (assumed README.r exists in current directory)

     showf W | tro          # your W-calibration files
     showf W CC | tro       # complete status of the calibration ($GIR_CALIB + your files)
                            # shows flags FF,LOC,PSF,WAVE - set to 1 if standard solution
                            # exists in $GIR_CALIB

   Check that the master W-calibration exists for your set-up.

     (cd $GIR_CALIB ; ls -1 girWav*.tfits) | fsplit | sorttable SLIT RES | tro

   If we have the master, run:

     pipe [-t] wcalMast RAW=ThArRawFile

   This  does  everything  and produces complete update of standard solution usable for any
   subsequent science processing of a given setup.

   The standard wavelength solution has 2 components:

     1. $GIR_CALIB/girSlitGeo_SLIT__RES_.tfits  - model of the slit
     2. $GIR_CALIB/girWavCoef_SLIT__RES_.tfits  - 2-D polynomial W-solution

     girWavCoef is obtained by line fitting of extracted ThAr spectra
     girSlitGeo is obtained by correlation on rebinned ThAr spectra

   If  the  simultaneous W-calibration is used a temporary table '....wlenRes.tfits' is
   created and used for the subsequent rebinning.

# blecha 24/06/2003
```

4.5 RDB COOK-BOOK

(Source file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc/README.rdb)

```
RDB related scripts
--------------------

addcol         add new columns
column         select columns
compute        modify content of columns
fitstotable    convert FITS table to rdb file
headoff        remove rdb head+possible comments
jointable      join 2 rdb tables on sorted column
justify        complete columns with blanks so as to have allignement
listtotable    convert format '1 labelled column per line' to TAB separated line with all columns
```

```
lt              better 'listtotable'
number          add 'number' column
rename          change name of column[s]
report          format table according the form
rmcol           remove column[s]
row             select column[s] according the expression
see             show TAB and End-of-lines
sorttable       sort table according to 1 or more columns
tabletofits     convert to FITS table (use -b option)
tabletolist     convert rdb to '1 labelled column per line' format
trim            replace TABs by blanks
tro             convert to pure ASCII
txt2r           convert ASCII with commented header to rdb

The best way to understand is run and modify following examples:

echo "qq" | number nu 10      # make test table
echo "qq" | number nu 10 | addcol text | compute 'text="mytest"number' | see
echo "qq" | number nu 10 | addcol text | compute 'text="mytest"number' > qq1
echo "qq" | number nu 10 | addcol text | compute 'text="my2test"number' | compute 'nu=10-nu' > qq2
tro < qq1
tro < qq2
sorttable -n nu < qq2 | jointable -j nu - qq1 | tro
sorttable -n nu < qq2 | rename nu nu2 | jointable -j1 nu2 -j2 nu - qq1 | tro
row 'text~"t4"' < qq1
row 'nu>2&&nu<9' < qq1 | tro

dfits ../*/raw/*.fits | fitshow > allKW     # get all KWs of main headers of all your data
head -3 < allKW | tabletolist | more     # see all KWs names
column File ESO.TEL.AIRM.END ESO.TEL.AIRM.START < allKW | tro   # see all airmasses
column File -r.AIRM < allKW | tro   # see all airmasses
column File -rFWHM < allKW | row 'length(ESO.TEL.AMBI.FWHM.END)>0' | tro | more

#AB  Mon 05/05/03 10:36:41
```

4.6 Calibration Pipes

LIST of Pipes

Name	Purpose
biasMast	Get master BIAS from raw biase[s]
darkMast	Get master DARK from raw dark[s]
locMast	Get master localisation from raw FF[s]
wcalMast	Get standard Wavelength calibration from Raw ThAr W-calibration image[s]
wcalOnly	Get standard Wavelength calibration from extracted ThAr spectra
wcalSlit	Update slit gemetry using rebinned Wavelength calibration

4.6.1 biasMast pipe

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                        and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | ficom biasMast,_MAST_,_AVER_,_CHECK_,_PYCO_,_PUTZ_,_CLEAN_ \
        | rr VERB=_VERB_ BIASHOTAREAS=_BIASHOTAREAS_ BIASOVSC=_BIASOVSC_ BIASCHIP=_BIASCHIP_ ZRANGE=_ZRANGE_ \
         MAST=_MAST_ AVER=_AVER_ CHECK=_CHECK_ NAM=_NAM_ OBJ=bias SLIT='' RES='' NSP='' GRAT='' RAW=_RAW_ \
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.biasMast)
    DESCRIPTION

       In  the  first  step,  the  input  biases are averaged through simple average or through
```

```
    cosmic removal averaging. The second option is presently limited by memory and disk
    space available to 6 images on PC with 1GB memory.

    Two polynomial bias models are built using the average bias:

      1. OvscBias.fits - model on PRE/OVER scans only
      2. ChipBias.fits - model on active area

    The diference between model 1 and 2, the biasDiff.fits, is stable and is used during the
    bias subtraction on all preprocessed frame.

  MAINTENANCE

    Unless the CCD readout (OVER/PRECscans, readout speed ...) or CCD itself change it is
    not necessary to build new bias master files since model of bias is adjusted using the
    PRE/OVER scan areas of science image.
    Actual (24/06/2003) masters have been built from 3 biases.

  # blecha 24/06/2003
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.biasMast)
```

Pipe definition

Name	biasMast
Section	Calibration/Maintenance
Purpose	Get master BIAS from raw biase[s]
Priority	Before any other processing or $GIR_CALIB/biasMast.fits must exists

Inputs

nu	file_name	relevant_pythone_function(internal_nu)
01	../raw/GIRAF.2003-01-27T23:56:48.363.fits	giCheck.py (20)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	$GIR_CALIB/maskMast.fits	giArithm.py (104)
02	$GIR_CALIB/biasMast.fits	copy Master (190)
03	$GIR_CALIB/biasDiffMast.fits	copy Master (190)

Quality control: use RTD and compared RMS of biasMast.fits and individual biases biasxx.fits

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe biasMast RAW=../raw/GIRAF.2003-01-27T23:56:48.363.fits
pipe biasMast RAW=../raw/GIRAF.2003-01-27T23:56:48.363.fits,../raw/GIRAF.2003-01-27T23:57:35.983.fits ,../raw/GIRA
01-27T23:58:23.524.fits
pipe biasMast RAW=‘showf BIAS R|files -3‘

Options

nu name=[default] **Option possible values**; option description
01 AVER=[average]

nu name=[default]
Option possible values; option description

average,cosmic; how we compute the average; cosmic should not be used with < 3 files on input. The averaging is used whenever a coma-separated list of files is given as input (RAW).

02 CHECK=[check]
check,nochek,noprep; Do the check-in; and/or preprocessing (write missing KW's ...) necessary with raw giraffe files

03 CLEAN=[no]
clean,noclean; !!! delete all files except README.r in current directory at start; this is mainly for tests

04 MAST=[biasmast,maskmast]
biasmast,maskmast,no; Produce new master bias and/or master mask and replace existing masks in CALIB

*05 PUTZ=putz[no]
putz,no; remove all intermediate data

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 BIASCHIP=[100:200:0:4096,1900:2000:0:4096]
Fixed for EEV chip - only for test; CHIP reference area; where the bias is computed on the chip

02 BIASHOTAREAS=[1450:4096:414:422]
Fixed for EEV chip - only for test; bias hot areas; where we subtract average master bias instead of model

03 BIASOVSC=[10:40:0:4096,2120:2130:0:4096]
Fixed for EEV chip - only for test; PRE/OVER scan reference area; where the bias is computed on prescan and overscan

04 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_

*05 RAW=../raw/GIRAF.2003-01-27T23:56:48.363.fits[None]
Filename; Full name of the RAW input image (.fits may be ommitted); In some cases coma-separated list is allowed - see description of the specific pipe. Note that even if you do not start from raw image (wcalOnly,calsim,...), the RAW could be given instead of RES OBJ and SLIT which will be then taken from KWs of raw image.

06 VERB=[None]
None,-v; general verbosity for commands;

07 ZRANGE=[-3s:3s]
use > 3s; intensity range [sigma-multiples]; The biasMast MASK is set for points out of thhe range average+ZRANGE

Call to functions

nu	call to Python or UNIX function
2	rm bias[0-9][0-9].fits
20	**giCheck.py** -w -K DARKVALUE= 0. - -on bias.fits ../raw/GIRAF.2003 -01 -27T23:56:48.363.fits
40	**giArithm.py** - -STAT -w 'bias.fits= average(bias??.fits)'
102	**giRecBias.py** - -blimits 10:40:0:4096, 2120:2130:0:4096 -w - -bmethod CURVE - -xorder 10 - -yorder 1 -d -o OvscBias.fits bias.fits; giRecBias.py - -blimits 100:200:0:4096, 1900:2000:0:4096 -w - -bmethod CURVE - -xorder 10 - -yorder 1 -d -o ChipBias.fits bias.fits
103	**giRecBias.py** - -blimits 10:40:0:4096, 2120:2130:0:4096 - -nobremove -w - -bmethod CURVE - -xorder 10 - -yorder 1 -o biasMast.fits bias.fits

nu	call to Python or UNIX function
104	**giArithm.py** - -STAT -Z -3s:3s:out -S 1450:4096:414:422 -w 'maskMast.fits= biasMast.fits'; cp maskMast.fits $GIR_CALIB
105	**giArithm.py** - -STAT -Y 50: -50 -w - -gain - 'biasDiff.fits= OvscBias.fits -ChipBias.fits'
190	cp biasMast.fits $GIR_CALIB/biasMast.fits; cp biasDiff.fits $GIR_CALIB/biasDiffMast.fits
100000	rm bias.fits OvscBias.fits ChipBias.fits biasMast.fits biasDiff.fits bias??.fits maskMast.fits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

4.6.2 darkMast pipe

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                      and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | ficom darkMast,_MAST_,_AVER_,_CHECK_,_PYCO_,_PUTZ_,_CLEAN_ \
        | rr VERB=_VERB_ BIASHOTAREAS=_BIASHOTAREAS_ BIASOVSC=_BIASOVSC_ BIASCHIP=_BIASCHIP_ ZRANGE=_ZRANGE_ \
         MAST=_MAST_ CHECK=_CHECK_ NAM=_NAM_ OBJ=dark SLIT='' RES='' NSP='' GRAT='' RAW=_RAW_ \
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.darkMast)
    DESCRIPTION

      The  darks  are averaged through cosmic removal averaging or through simple average (not
      recommended).  The  first option is presently limited by memory and disk space available
      to 6 images on PC with 1GB memory.  Averaged darks are then bias-subtracted

    # blecha 24/06/2003
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.darkMast)
```

Pipe definition

Name	darkMast
Section	Calibration/Maintenance
Purpose	Get master DARK from raw dark[s]
Priority	Before any processing using option 'dark'

Inputs

nu	file_name	relevant_pythone_function(internal_nu)
01	../raw/GIRAF.2003-01-27T23:56:48.363.fits	giCheck.py (21)
02	$GIR_CALIB/maskMast.fits	giRecBias.py (100)
03	$GIR_CALIB/biasMast.fits	giRecBias.py (100)
04	$GIR_CALIB/biasDiffMast.fits	giArithm.py (101)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	darkbSub.fits	giRecCosmicMC.py (160)

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe darkMast RAW=../raw/GIRAF.2003-01-27T23:56:48.363.fits MAST=no
pipe darkMast RAW=‘showf BIAS R | files -3‘ MAST=no PUTZ=putz CLEAN=clean

Options

nu name=[default]
Option possible values; option description

01 AVER=[cosmic]
cosmic,average; how we compute the average; cosmic should not be used with < 3 files on input. The averaging is used whenever a coma-separated list of files is given as input (RAW).
02 CHECK=[check]
check,nochek,noprep; Do the check-in; and/or preprocessing (write missing KW's ...) necessary with raw giraffe files
03 CLEAN=[no]
clean,noclean; !!! delete all files except README.r in current directory at start; this is mainly for tests
*04 MAST=no[darkmast]
darkmast,no; Produce new master dark and replace existing masks in CALIB
*05 PUTZ=putz[putz]
putz,no; remove all intermediate data

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 BIASOVSC=[10:40:0:4096,2120:2130:0:4096]
Fixed for EEV chip - only for test; PRE/OVER scan reference area; where the bias is computed on prescan and overscan
02 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_
*03 RAW=../raw/GIRAF.2003-01-27T23:56:48.363.fits[None]
Filename; Full name of the RAW input image (.fits may be ommitted); In some cases coma-separated list is allowed - see description of the specific pipe. Note that even if you do not start from raw image (wcalOnly,calsim,...), the RAW could be given instead of RES OBJ and SLIT which will be then taken from KWs of raw image.
04 VERB=[None]
None,-v; general verbosity for commands;

Call to functions

nu	call to Python or UNIX function
2	rm dark[0-9][0-9].fits
11	rm /tmp/mmap*
21	**giCheck.py** -w -K DARKVALUE= 0. - -on dark.fits ../raw/GIRAF.2003 -01 -27T23:56:48.363.fits
100	**giRecBias.py** - -bsigma 100 - -blimits 10:40:0:4096, 2120:2130:0:4096 -w - -bmethod ZMASTER +CURVE - -xorder 10 - -yorder 1 -b $GIR_CALIB/maskMast.fits - -on darkbSub1.fits 'dark??.fits' $GIR_CALIB/biasMast.fits
101	**giArithm.py** -w - -STAT - -on "darkbSub.fits= darkbSub1??.fits +$GIR_CALIB/biasDiffMast.fits"
160	**giRecCosmicMC.py** -w - -crsigma 4.0 'darkbSub??.fits' $GIR_CALIB/maskMast.fits; mv darkbSub00.crmavg.fits darkbSub.fits
100005	rm dark.fits OvscBias.fits darkMast.fits dark??.fits darkbSub??.fits darkbSub1??.fits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

4.6.3 locMast pipe

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                      and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | ficom locMast,_LOCPSF_,_MAST_,_SLM_,_PLOT_,_CHECK_,_NSP0_,_AVER_,_LOCRAW_,_PUTZ_,_CLEAN_,_DARK_ \
        | rr VERB=_VERB_ LOCYORD=_LOCYORD_ PLOUT=_PLOUT_ PRT=_PRT_ NSP0=_NSP0_ CHECK=_CHECK_ PLOTOPT=_PLOTOPT_ \
         PLOT=_PLOT_ PSFE=_PSFE_ SLM=_SLM_ LOCPSF=_LOCPSF_ LYWIDTH=_LYWIDTH_ LNOISE=_LNOISE_ NAM=_NAM_ OBJ=Fff \
         SLIT=_SLIT_ RES=_RES_ RAW=_RAW_ \
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.locMast)
    DESCRIPTION

      This is THE COMPULSORY step before any extraction (Th-Ar or any scientific spectra) in a
      given  setup. It produces the localisation necessary to any extraction, the model of the
      transverse  PSF  required  by OPTIMAL extraction and the spectroscopic extracted (called
      'narrow') flat-field.

      To check existing standard preliminary localisation do:

         (cd $GIR_CALIB/FF; ls *clocy*fits) | fsplit | tro

        If  no  master  exists,  you must use option LOCRAW=locraw (default) otherwise you can
        specify  LOCRAW=no

      To check existing standard PSF localisation do:

        (cd $GIR_CALIB/FF; ls *.psfwidt*fits) | fsplit | tro

      Only  raw  input frame are required as input data.

    MAINTENANCE

        When  starting  from  scratch,  use  at  least  the  option  PLOT=loc which shows in a
        comprehensive  way  the cross-section of the spectra over-plotted to localization in 9
        significant  positions  of  the  CCD.  In case of doubt use PLOT=locfull, respectively
        'locfullA' for IFU/Argus (beware many plots); see also under 'PLOT-COOKBOOK'.

        Spectra  listed  in  the  OzPoz  table and existing in the fibreTable + 5 simultaneous
        calibration  spectra  MUST  BE DETECTED DURING THE PRELIMINARY LOCALIZATION. Presently
        (24/06/2003)  the  fibreTable  in  image  header  is  not  used,  instead  we  use the
        $GIR_CALIB/slitGeo_SLIT__RES_.tfits  which  contains the same information but has been
        corrected  for  recurrent  errors in fibreTable. Once the fibreTable table is safe and
        validated  (Argus  pending)  we  will  switch to it. For purely technical reasons, the
        inital  version  of $GIR_CALIB/slitGeo_SLIT__RES_.tfits must exists. Tables up to date
        according the SV data are listed below (all other tables have been copied from closest
        setup of the same SLIT):

           IFU1   : L4,5,6,7   H12
           IFU2   : L4,6,7     H12,21
           Medusa1: L3,4,8     H2,3,4,5,6,9,13,14,15
           Medusa2: L3,4,8     H2,3,4,5,6,9,13,14,15

        The    slitGeoTable    are    updated    automatically    by    wcalMast    when    option
        MAST=wcalmast,slitmast  is  specified. Only moderately accurate table is necessary for
        the localisation.

    MAINTENANCE (expert)

        The slitGeoTable is the only piece of calibration data which gives the DRS the list of
        available  spectra  TB detected - OzPoz table knowing only buttons (several spectra in
        IFU/Argus).  While  missing  buttons are detected through OzPoz table, missing spectra
        (broken  fibre  in  IFU/Argus  bundle or image substantially shifted on CCD) are known
```

```
only  throug slitGeoTable. This is the reason why it must be updated mmanually in such
a case.
slitGeoTable manual update

  The  simplest,  though  tedious  way,  is to update 22+8=30 setupts for a given slit
  using  any  .tfits manipulation tool (fitsview for example - not supplied with DRS).
  In case of general misallignement all setups should be re-calibrated anyway. In this
  case we recommend to modify one table/slit (for example H9) and copy it to all other
  setups.  After  that  you  must  run  locMast  and  wcalMast  for  all setups on new
  calibration data.

# blecha 24/06/2003
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.locMast)
```

Pipe definition

Name	locMast
Section	Calibration
Purpose	Get master localisation from raw FF[s]
Priority	Before any extraction of the setup

Inputs

nu	file_name	relevant_pythone_function(internal_nu)
01	../raw/GIRAF.2003-01-28T10:07:04.309.fits	giCheck.py (21)
02	$GIR_CALIB/maskMast.fits	giRecBias.py (100)
03	$GIR_CALIB/biasMast.fits	giRecBias.py (100)
04	$GIR_CALIB/biasDiffMast.fits	giArithm.py (101)
05	$GIR_CALIB/girSlitGeoMedusa1H14.tfits	giRecNLoc2.py (400)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	FffMedusa1H14bSub.fits	giArithm.py (150)
02	$GIR_CALIB/FF/Medusa1H14Mast.clocy.fits	copy giRecNLoc2.py (433)
03	$GIR_CALIB/FF/Medusa1H14Mast.clocw.fits	copy giRecNLoc2.py (433)
04	$GIR_CALIB/FF/Medusa1H14Mast.plocy.fits	copy giRecPsfLoc.py (444)
05	$GIR_CALIB/FF/Medusa1H14Mast.plocw.fits	copy giRecPsfLoc.py (444)
06	$GIR_CALIB/FF/Medusa1H14Mast.psfcent.fits	copy giRecPsfLoc.py (444)
07	$GIR_CALIB/FF/Medusa1H14Mast.psfexpo.fits	copy giRecPsfLoc.py (444)
08	$GIR_CALIB/FF/Medusa1H14Mast.psfwidt.fits	copy giRecPsfLoc.py (444)
09	$GIR_CALIB/FF/FffMedusa1H14Mast.extsp.fits	copy giArithm.py (505)
10	$GIR_CALIB/FF/FffMedusa1H14Mast.extsperr.fits	copy giArithm.py (505)
11	$GIR_CALIB/FF/FffMedusa1H14Mast.oxtsp.fits	copy giArithm.py (507)
12	$GIR_CALIB/FF/FffMedusa1H14Mast.oxtsperr.fits	copy giArithm.py (507)

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe locMast RAW=../raw/GIRAF.2003-01-28T10:07:04.309.fits PLOT=loc
pipe locMast RAW=‘showf FF R | files -3‘ PLOT=loc PUTZ=putz

Options

nu name=[default]
Option possible values; option description

01 AVER=[average]
average,cosmic; how we compute the average; cosmic should not be used with < 3 files on input. The averaging is used whenever a coma-separated list of files is given as input (RAW).
02 CHECK=[check]
check,nochek,noprep; Do the check-in; and/or preprocessing (write missing KW's ...) necessary with raw giraffe files
03 CLEAN=[no]
clean,noclean; !!! delete all files except README.r in current directory at start; this is mainly for tests
04 DARK=[no]
dark, no; remove dark; we remove also dark
05 LOCPSF=[locpsf]
locpsf,nolocpsf; Accurate localisation and PSF model; Produce model of the transverse PSF used for optimal extraction
06 LOCRAW=[locraw]
locraw,no; do preliminary localisation from scratch; If 'no' is specified, the Master localisationmust exist
07 MAST=[locmast,ffmast]
locmast,ffmast; Produce new master localisation; Various $GIR_CALIB/...Master are produced depenting whether 'locraw' and/or 'locpsf' options are specified (see I/Os)
*08 PLOT=loc[loc]
loc,locfull,psfpar3,locfullA; coma-separated list of localisation plots; loc produces partial plot while locfull/locfullA (IFU/Argus) show all spectra
*09 PUTZ=putz[no]
putz,no; remove all intermediate data

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 LNOISE=[-0.2]
-0.2 to -1 for L1=-0.3; Raw detection threshold; (Relative to average level of spectra)
02 LOCYORD=[4]
0-10; Ordre of the Y-position localisation polynomial; Chebyshev 1-D polynomial fitted on the threshold localisation
03 LYWIDTH=['-']
'-' or a number; Equalization Filter size [px]; If set to '-' we take 14 in Medusa and 6-7 in Argus/Ifu. This the width [px] of the transverse equlisation filter used in raw localisation.
04 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_
05 NSP0=[0]
Fixed - only for test; force number of spectra on the frame to this values. TBU ONLY when OzPoz tables is missing.
06 PLOTOPT=[None]
'-Lnnn'; any ad-option to 'bigg' commands; Posible usage: '-Lnnn' log scale with nnn offset
07 PLOUT=[X11]
[X11],yes; list of graphic outputs; 'X11,yes' will produce plot to screen and to .eps file while 'yes' to file only. File name are generated automatically. After execution do 'ls -lt *.eps' to see files
08 PSFE=[-3]
-3 to 3; PSF exponent (- means start with |PSFE| and fit)
*09 RAW=../raw/GIRAF.2003-01-28T10:07:04.309.fits[None]

nu name=[default]
Parameter possible values; parameter description

Filename; Full name of the RAW input image (.fits may be ommitted); In some cases coma-separated list is allowed - see description of the specific pipe. Note that even if you do not start from raw image (wcalOnly,calsim,...), the RAW could be given instead of RES OBJ and SLIT which will be then taken from KWs of raw image.

*10 RES=H14[None]
H1,H2,...H22 or L1,l2,...L8; Resolution range; If not given derived from KW: 'INS FILT NAME'. See also $GIR_CALIB/girGratingHR316/LR600.tfits (do: egrep -v "#" < $GIR_CALIB/girGratingHR316.tfits

*11 SLIT=Medusa1[-]
Medusa1/2,IFU1/2,Argus; Geometry of input slit. If not given is taken from KW: 'INS2 SLIT NAME'.

12 VERB=[None]
None,-v; general verbosity for commands;

Call to functions

nu	call to Python or UNIX function
2	rm FffMedusa1H14[0-9][0-9].fits
21	**giCheck.py** -w -K DARKVALUE= 0. - -on FffMedusa1H14.fits ../raw/GIRAF.2003 -01 -28T10:07:04.309.fits
100	**giRecBias.py** - -bsigma 100 - -blimits 10:40:0:4096, 2120:2130:0:4096 -w - -bmethod ZMASTER +CURVE - -xorder 10 - -yorder 1 -b $GIR_CALIB/maskMast.fits - -on FffMedusa1H14bSub1.fits 'FffMedusa1H14??.fits' $GIR_CALIB/biasMast.fits
101	**giArithm.py** -w - -STAT - -on "FffMedusa1H14bSub.fits= FffMedusa1H14bSub1??.fits +$GIR_CALIB/biasDiffMast.fits"
150	**giArithm.py** -w - -STAT 'FffMedusa1H14bSub.fits= average(FffMedusa1H14bSub??.fits)'
195	mv FffMedusa1H14bSub.fits FffMedusa1H14dbSub.fits
400	**giRecNLoc2.py** - -lnoise -0.2 - -lywidth - - -lewidth 1.0 - -yorder 4 - -worder 2 -w FffMedusa1H14dbSub.fits
433	cp FffMedusa1H14dbSub.clocy.fits Medusa1H14Mast.clocy.fits; cp FffMedusa1H14dbSub.clocw.fits Medusa1H14Mast.clocw.fits; cp Medusa1H14Mast.cloc[yw].fits $GIR_CALIB/FF
430	**biggLoc.py** - -OUTPUT= , X11FffMedusa1H14.loc1 -W 1200 -N 1::3, c: -:3, -3: -0 -X 0:4001:1:2000 - -norecenter FffMedusa1H14dbSub FffMedusa1H14dbSub.clocy.fits
440	**giRecPsfLoc.py** - -yorder 4 - -worder 3 -w - -pexpwid -3 - -pmodel psfexp2 FffMedusa1H14dbSub.fits
444	cp FffMedusa1H14dbSub.plocy.fits $GIR_CALIB/FF/Medusa1H14Mast.clocy.fits; cp FffMedusa1H14dbSub.plocw.fits $GIR_CALIB/FF/Medusa1H14Mast.clocw.fits;cp FffMedusa1H14dbSub.psfcent.fits $GIR_CALIB/FF/Medusa1H14Mast.psfcent.fits;cp FffMedusa1H14dbSub.psfexpo.fits $GIR_CALIB/FF/Medusa1H14Mast.psfexpo.fits;cp FffMedusa1H14dbSub.psfwidt.fits $GIR_CALIB/FF/Medusa1H14Mast.psfwidt.fits
451	**biggLoc.py** - -OUTPUT= , X11FffMedusa1H14.locpsf -N 1:4, c: -:3 , -3: -0 -X 10, c, -10 - -norecenter - -psfnormsum FffMedusa1H14dbSub FffMedusa1H14dbSub.plocy.fits
500	**giRecNExt.py** -w - -nograph FffMedusa1H14dbSub.fits FffMedusa1H14dbSub.clocy.fits FffMedusa1H14dbSub.clocw.fits
503	**giRecNExt.py** -w - -nofbkg - -pmaxdw 0.05 - -pdword 2 - -emethod OPTIMAL - -hsigma 6 - -nograph FffMedusa1H14dbSub.fits FffMedusa1H14dbSub.clocy.fits FffMedusa1H14dbSub.clocw.fits - FffMedusa1H14dbSub
504	**giArithm.py** - -STAT -w - -norm 'FffMedusa1H14Mast.extsp.fits= FffMedusa1H14dbSub.extsp.fits'; giArithm.py -w 'FffMedusa1H14Mast.extsperr.fits= FffMedusa1H14dbSub.extsperr.fits/DIVFffMedusa1H14dbSub.extsp.fits'
505	cp FffMedusa1H14Mast.extsp.fits FffMedusa1H14Mast.extsperr.fits $GIR_CALIB/FF
506	**giArithm.py** - -STAT -w - -norm 'FffMedusa1H14Mast.oxtsp.fits= FffMedusa1H14dbSub.oxtsp.fits'; giArithm.py -w 'FffMedusa1H14Mast.oxtsperr.fits= FffMedusa1H14dbSub.oxtsperr.fits/DIVFffMedusa1H14dbSub.oxtsp.fits'
507	cp FffMedusa1H14Mast.oxtsp.fits FffMedusa1H14Mast.oxtsperr.fits $GIR_CALIB/FF

nu	call to Python or UNIX function
100010	rm FffMedusa1H14??.fits FffMedusa1H14bSub??.fits FffMedusa1H14bSub1??.fits FffMedusa1H14dbSub.plocy.fits FffMedusa1H14dbSub.plocw.fits FffMedusa1H14dbSub.plocc.tfits FffMedusa1H14dbSub.psfampl.fits FffMedusa1H14dbSub.psfback.fits FffMedusa1H14dbSub.psfexpo.fits FffMedusa1H14dbSub.psfcent.fits FffMedusa1H14dbSub.psfwidt.fits Medusa1H14Mast.plocy.fits Medusa1H14Mast.plocw.fits FffMedusa1H14.fits FffMedusa1H14bSub1.fits FffMedusa1H14dbSub.fits FffMedusa1H14dbSub.clocy.fits FffMedusa1H14dbSub.clocw.fits FffMedusa1H14dbSub.clocc.tfits Medusa1H14Mast.clocy.fits Medusa1H14Mast.clocw.fits
100011	rm FffMedusa1H14dbSub.extsp.fits FffMedusa1H14dbSub.extsperr.fits FffMedusa1H14dbSub.extspyct.fits FffMedusa1H14dbSub.extspnix.fits FffMedusa1H14Mast.extsp.fits FffMedusa1H14Mast.extsperr.fits FffMedusa1H14dbSub.oxtsp.fits FffMedusa1H14dbSub.oxtsperr.fits FffMedusa1H14dbSub.oxtspyct.fits FffMedusa1H14dbSub.oxtspnix.fits FffMedusa1H14Mast.oxtsp.fits FffMedusa1H14Mast.oxtsperr.fits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

4.6.4 wcalMast pipe

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                      and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | ficom wcalMast,_MAST_,_REB_,_SLM_,_EMETHOD_,_PLOT_,_CHECK_,_PUTZ_,_CLEAN_ \
        | rr VERB=_VERB_ REBSCALE=_REBSCALE_ OBJ=_OBJ_ PLOTOPT=_PLOTOPT_ WSLIT=_WSLIT_ CHECK=_CHECK_ \
         BACK=bkremove RAW=_RAW_ OBJ=ThAr PLOT=_PLOT_ SLM=_SLM WLTHRESH=_WLTHRESH_ LINES=_LINES_ WXRES=_WXRES_
         WPLOTSC=_WPLOTSC_ WNMAXLINE=_WNMAXLINE_ WSSIGMA=_WSSIGMA_ WIND=_WIND_ RSTEP=_RSTEP_ WGFLAG=_WGFLAG_ \
         WFFLAG=_WFFLAG_ NAM=_NAM_ SLIT=_SLIT_ RES=_RES_ WCHEBXY=_WCHEBXY_ EMETHOD=_EMETHOD_ CCGRAPH=_CCGRAPH_
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.wcalMast)
    This  is the full processing in order to get full master wavelength solution from raw ThAr
    exposure[s].  It  does extraction, w-calibration and rebinning. When working repeatedly on
    preliminary  W-solution  from scratch (maintenance), the 'wcalOnly' which starts after the
    extraction   is  much  more  appropriate  since  it  does  not  repeat  the
    preprocessing and extraction.

    CAUTION:  if  MAST=wcalmast  is  specified,  the resulting w-solution overwrites
    existing solution in $GIR_CALIB.

    STANDARD OPERATION - update existing solution

      Only input file has to be specified  RAW=xxx

      It  produces  master W-solution $GIR_CALIB/girWavCoefIFU1L4.tfits but keeps slitGeometry
      unchanged.

    MAINTENANCE (STARTING FROM SCRATCH)

      you   can   check  existing  standard  solution  and  parameters  produced  by  previous
      W-calibrations:

          (cd $GIR_CALIB; ls -1 *Wav*tfits) | fsplit | tro
          (cd $GIR_CALIB; dfits -a *Wav*.tfits) | fitshow 'WSOPT*' | tro

      This    procedures    is    TB    used   when   standard   solution,   i.e.   the   file
      $GIR_CALIB/girWavCoefxxxx.tfits  does  not exists for a setup, or after the spectrograph
      adjustement. The option 'WXRES=no' tells the giRecNWcal2.py to start from scratch.

      1. You must first run wcalMast with options:

        pipe wcalMast REB=no WXRES=no WIND=25,20 RAW=xxx
```

```
   The     initial     optical     solution     is     taken     from     grating     tables
   $GIR_CALIB/girGratingHR316/LR600.tfits. It computes optical model parameters and
   Chebyshev correction using the current slit geometry and establishes 1-st guess for
   complete solution in the table $GIR_CALIB/girWavCoefxxx.tfits

   in case of problem you may:

     - increase the first search window WIND=35,25,20
     - increase the WSSIGMA=500

   if the initial solution is clearly wrong, run with options

   REB=no WGFLAG=-1 WXRES=no MAST=no PLOT=no

     It will stop processing at graphic display of line search.

     Read current slit offset from the graphics (1-st value after 'slit:'), set parameter
     WSLIT= to new value and re-run giRecNWcal2.py command (button 'List Command' with
     GUI or flag -t in command-line mode). INCREASING the WSLIT moves search windows to
     the left on the graphics. Note that you can see line selection graphics in great
     details by just adding the option '-X 0:-0:1050:1050' or '-X 0:-0:1050:1050 -L 1' to
     command giRecNWcal2.py; also the giRecNWcal2.py gives the list of selected
     lines+expected fluxes making easy to reckognize whether the line-search boxes are
     correctly positionned. Repeat till you have most lines well within the window. You
     may also increase window width using the WIND parameter.


 2. Run calibration only using the preliminary solution with new option (narrow last window).

   pipe wcalOnly WIND=20,15,15 RAW=xxx

 3. Adjust slit geometry for this setup (2 or more times)

    pipe wcalSlit RAW=xxx

MAINTENANCE (expert)

  The following description is for expert use only. It imply the 'manual' handling of some
  critical static data in $GIR_CALIB and we recommend strongly to make a copy of $GIR_CALIB
  top directory (not necessary for large FF sub-directory) prior to proceed.

  The W-solution has 2-3 components (example given for Medusa1H9 setup):

    1.The simple optical model defined by few parameters (girGratingHR316.tfits).
      Two tables, one per grating, supply these parameters. These tables are COMPLETELY
      STATIC and could only be modified through edition (fits or ASCII editors)

    2.The slit geometry (girSlitGeoMedusa1H9.tfits).
      150 tables, one per setup, are MAINTAINED BY wcalMast using the option
      MAST=wcalmast,slitmast. This table is used to build initial optical model which takes
      into account the relative fibre position on the slit.

    3.The master W-solution (girWavCoefMedusa1H9.tfits)
      150 tables, one per setup, are MAINTAINED BY wcalMast using the option
      MAST=wcalmast . This table, if exists, is used instead of girGratingHR316.tfits to
      built optical model and contains also Chebyshev correction. TOGETHER WITH SLIT
      GEOMETRY IT COMPOSES FULL MASTER W-SOLUTION.

    4.Simultaneous calibration adjustement (xxMedusa1H9.wlenRes.tfits - LOCAL DIRECTORY)
      This - temporary file - is created by extract with option SIMCAL=simcal and is valid
      only for the file being extracted. It is the correction to master W-solution.

  It is desirable, that the girGratingHR316/LR600.tfits provide the initial optical model accurate
  enough to make automatic search for lines efficent. Here are few hints how to
  maintain table (example for HR grating).
```

```
        See parameters of all currently available optical models (setups) for HR

          ( cd $GIR_CALIB ; dfits girWavCoef*.tfits ) | whead | sorttable SETUP GR SLIT \
          | row 'SLIT~" GR=="H"' | uniq -f 9 | tro

        Or directly update girGratingHR316 table (version rdb)

           egrep '^#' girGratingHR316.r > girGratingHR316.r.new

           ( cd $GIR_CALIB ; dfits girWavCoef*.tfits ) | whead | sorttable SETUP GR SLIT \
          | row 'GR=="H"' | uniq -f 9 | jointable -a1 -j1 SETUP -j2 RES girGratingHR316.r - \
          |   compute  \
          '{if(length(DATE)>0){LASTUPDATE=DATE;FCOLL=FCOLL_2;GCAM=GCAM_2;SDX=SDX_2;SDY=SDY_2;SPHI=SPHI_2}}'\
          | rmcol FCOLL_2 GCAM_2 THETA_2 SDX_2 SDY_2 SPHI_2 DATE SLIT SETUP_2 GR >> girGratingHR316.r.new

        Inspect old and new table (you may also want edit the header KW 'COMMENT')

          tro < girGratingHR316.r.new
          tro < girGratingHR316.r

        Convert new table to '.tfits'

          tabletofits -b girGratingHR316.r.new > girGratingHR316.tfits

        Do the same for LR grating

           egrep '^#' girGratingLR600.r > girGratingLR600.r.new

          ( cd $GIR_CALIB ; dfits girWavCoef*.tfits ) | whead | sorttable SETUP GR SLIT \
          | row 'GR=="L"' | uniq -f 9 | jointable -a1 -j1 SETUP -j2 RES girGratingLR600.r - \
          | compute  \
          '{if(length(DATE)>0){LASTUPDATE=DATE;FCOLL=FCOLL_2;GCAM=GCAM_2;SDX=SDX_2;SDY=SDY_2;SPHI=SPHI_2}}'\
          | rmcol FCOLL_2 GCAM_2 THETA_2 SDX_2 SDY_2 SPHI_2 DATE SLIT SETUP_2 GR >> girGratingLR600.r.new

          tabletofits -b girGratingLR600.r.new > girGratingLR600.tfits

    #AB Sun May  4 20:29:33 CEST 2003
    #   Thu Jun  5 14:26:20 MEST 2003 corrige XRES->WXRES + escription clarified
    #   blecha 26/06/2003  added expert maintenance
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.wcalMast)
```

Pipe definition

Name	wcalMast
Section	Calibration
Purpose	Get standard Wavelength calibration from Raw ThAr W-calibration image[s]

Inputs

nu	file_name	relevant_pythone_function(internal_nu)
01	../raw/GIRAF.2003-01-28T10:17:25.736.fits	giCheck.py (21)
02	$GIR_CALIB/FF/Medusa1H14Mast.clocy.fits	copy from CALIB (31)
03	$GIR_CALIB/FF/Medusa1H14Mast.clocw.fits	copy from CALIB (31)
04	$GIR_CALIB/maskMast.fits	giRecBias.py (100)
05	$GIR_CALIB/biasMast.fits	giRecBias.py (100)
06	$GIR_CALIB/biasDiffMast.fits	giArithm.py (101)
07	$GIR_CALIB/girSlitGeoMedusa1H14.tfits	giRecNWcal2.py (600)
08	$GIR_CALIB/girWavCoefMedusa1H14.tfits	giRecNWcal2.py (600)
09	$GIR_CALIB/girLineThAr.tfits	giRecNWcal2.py (600)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	ThArMedusa1H14dbSub.clocy.fits	copy from CALIB (31)
02	ThArMedusa1H14dbSub.clocw.fits	copy from CALIB (31)
03	ThArMedusa1H14bSub.fits	giArithm.py (150)
04	ThArMedusa1H14dbSub.fits	rename giRecBias.py (195)
05	ThArMedusa1H14dbSub.extsp.fits	giRecNExt.py (500)
06	ThArMedusa1H14dbSub.extsperr.fits	giRecNExt.py (500)
07	$GIR_CALIB/girWavCoefMedusa1H14.tfits	copy giRecNWcal2.py (605)
08	ThArMedusa1H14dbSub.extsp.rebinlin.fits	giRecBinn2.py (630)
09	ThArMedusa1H14dbSub.extsperr.rebinlin.fits	giRecBinn2.py (630)

Quality control: Mainly graphics 'wcal3'

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe wcalMast RAW=../raw/GIRAF.2003-01-28T10:17:25.736.fits
pipe wcalMast RAW=‘showf W R | files -2‘

Options

nu name=[default]
Option possible values; option description

01 CHECK=[check]
check,nochek,noprep; Do the check-in; and/or preprocessing (write missing KW's ...) necessary with raw giraffe files

02 CLEAN=[no]
clean,noclean; !!! delete all files except README.r in current directory at start; this is mainly for tests

03 MAST=[wcalmast]
wcalmast,slitmast,no; Produce new master wavelength solution and/or if 'slitmast' new slit geometry masters new master wavelength solution and/or if 'slitmast' new slit geometry master. The later must be used with option REB=rebin

04 PLOT=[wcal3]
ext,wcal1,wcal3,rebin1,rebin5,rebin10,rebin20,wcalline1,wcalline2; coma-separated list of plot selection

*05 PUTZ=putz[no]
putz,no; remove all intermediate data

06 REB=[rebin]
rebin,no; W-rebinning after the W-calibration;

07 WGFLAG=[1]
0,1,-1; Flag of linesearc control plot in W-cal; 1 - we plot line search, 0-we do not plot, -1 only plot (no attempt to get W-solution)

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 EMETHOD=[SUMM]

nu name=[default]
Parameter possible values; parameter description

SUMM, OPTIMAL; Extraction method

02 LINES=[girLine_OBJ_.tfits]
(girLineThAr.tfits,girLineSun.tfits); table of spectral line; see also 'ls $GIR_CALIB/girLine*.tfits'; Line table name for plots or W-calibrations located in $GIR_CALIB

03 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_

04 OBJ=[ThAr]
Fixed to ThAr; This is the first part of the names of I/O images

05 PLOTOPT=[None]
'-Lnnn'; any ad-option to 'bigg' commands; Posible usage: '-Lnnn' log scale with nnn offset

*06 RAW=../raw/GIRAF.2003-01-28T10:17:25.736.fits[None]
Filename; Full name of the RAW input image (.fits may be ommitted); In some cases coma-separated list is allowed - see description of the specific pipe. Note that even if you do not start from raw image (wcalOnly,calsim,...), the RAW could be given instead of RES OBJ and SLIT which will be then taken from KWs of raw image.

07 REBSCALE=[lin]
lin,log; W-rebinning scale linear or logarithmic; Must be cosistent with slitGeometry adjustement scale

*08 RES=H14[None]
H1,H2,...H22 or L1,l2,...L8; Resolution range; If not given derived from KW: 'INS FILT NAME'. See also $GIR_CALIB/girGratingHR316/LR600.tfits (do: egrep -v "#" < $GIR_CALIB/girGratingHR316.tfits

09 RSTEP=[-]
'-' or number; Rebinnig step in [nm]; '-' resolution-depedent default will use 0.005 in HR and 0.02 in LR

*10 SLIT=Medusa1[-]
Medusa1/2,IFU1/2,Argus; Geometry of input slit. If not given is taken from KW: 'INS2 SLIT NAME'.

11 VERB=[None]
None,-v; general verbosity for commands;

12 WCHEBXY=[6,4]
0-10,0-10; W-calibration Chebyshev correction; degree of 2D polynomyal correction to optical model in W-calibration

13 WFFLAG=[0,0,0,0,0,1,1,1]
Fixed - only for maintenance; Flags governing the W-fit; in the following order: colimator focale, camera magn., grating angle, ordre, groove density, slit diplacements: dx,dy,angle. In Normal operation only slit diplacements flags should be set to fit. The colimator focale and the grating angle should only be adjusted after a major changes on instrument.

14 WIND=[15]
25,20,15; Width [px] of search window; coma-separated list of width of window for repeated search of line

15 WLTHRESH=[1]
1-10; multiple of BIASSIGMA; Threshold for use of the spectral line

16 WNMAXLINE=[80]
10-200; maximum number of spectral lines; During the W-calibration, take only WNMAXLINE number of brigtest spectral lines

17 WPLOTSC=[5]
1-20; Control of W-residuals plot; Max/min [px] of the graphic showing the residuals

18 WSLIT=[-]
Maintenance only; Initial x,y,phi slit displacement in wavelength solution after spectro adjustement; 1-3 coma-separated params.

19 WSSIGMA=[200]
200; Sigma clipping for Chebyshev; Spectral lines with position |Chebyshev model - position | > WSSIGMA are rejected - TB found better parameter

20 WXRES=[None]

nu name=[default] **Parameter possible values**; parameter description
no; if 'no' we do not use existing W-calibration; if nothing specified we use existing W-calibration in $GIR_CALIB, if we do not want it (or if no calibration exists) it must be 'no'

Call to functions

nu	call to Python or UNIX function
21	**giCheck.py** -w -K DARKVALUE= 0. - -on ThArMedusa1H14.fits ../raw/GIRAF.2003 -01 -28T10:17:25.736.fits
31	cp $GIR_CALIB/FF/Medusa1H14Mast.clocy.fits ThArMedusa1H14dbSub.clocy.fits;cp $GIR_CALIB/FF/Medusa1H14Mast.clocw.fits ThArMedusa1H14dbSub.clocw.fits
100	**giRecBias.py** - -bsigma 100 - -blimits 10:40:0:4096, 2120:2130:0:4096 -w - -bmethod ZMASTER +CURVE - -xorder 10 - -yorder 1 -b $GIR_CALIB/maskMast.fits - -on ThArMedusa1H14bSub1.fits 'ThArMedusa1H14??.fits' $GIR_CALIB/biasMast.fits
101	**giArithm.py** -w - -STAT - -on "ThArMedusa1H14bSub.fits= ThArMedusa1H14bSub1??.fits +$GIR_CALIB/biasDiffMast.fits"
150	**giArithm.py** -w - -STAT 'ThArMedusa1H14bSub.fits= average(ThArMedusa1H14bSub??.fits)'
195	mv ThArMedusa1H14bSub.fits ThArMedusa1H14dbSub.fits
500	**giRecNExt.py** -w - -nograph ThArMedusa1H14dbSub.fits ThArMedusa1H14dbSub.clocy.fits ThArMedusa1H14dbSub.clocw.fits
600	**giRecNWcal2.py** - -noquiet - -lthresh 1 - -xres - -graph 1 - -wssigma 200 - -linmod psfexp - -lexpwid -3 - -lwidth 15 - -fflags 0, 0, 0, 0, 0, 1, 1, 1 - -lbright 80 - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -wltable girLineThAr.tfits - -nosubsfit - -xworder 2, 2 - -wsorder 6, 4 -w - -slit - ThArMedusa1H14dbSub.extsp.fits ThArMedusa1H14dbSub.clocy.fits
605	cp ThArMedusa1H14dbSub.extsp.xores.tfits $GIR_CALIB/girWavCoefMedusa1H14.tfits
610	**biggExt.py** - -OUTPUT= , X11ThArMedusa1H14.Wopmodel -A -S 3 -Y 0: -0 -N 0: -0:1:0.21 -Z -5:5 - -xlabel 'X pixel' - -ylabel "X optModel -gausFit in px Medusa1H14" -D ThArMedusa1H14dbSub.extsp.xmres.fits ThArMedusa1H14dbSub.extsp.xxres.fits
611	**biggExt.py** - -OUTPUT= , X11ThArMedusa1H14.Wcheb -A -S 3 -Y 0: -0 -N 0: -0:1:0.21 -Z -1:1 - -xlabel 'X pixel' - -ylabel "X Chebyshev (6, 4) -gausFit in px (wind 15) Medusa1H14" -D ThArMedusa1H14dbSub.extsp.xcres.fits ThArMedusa1H14dbSub.extsp.xxres.fits
630	**giRecBinn2.py** -w - -bnstep - - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -bnlin - -bnrange= setup - -bnmethod linear ThArMedusa1H14dbSub.extsp.fits - ThArMedusa1H14dbSub.clocy.fits -
100020	rm ThArMedusa1H14??.fits ThArMedusa1H14bSub??.fits ThArMedusa1H14bSub1??.fits Medusa1H14Mast.clocy.fits Medusa1H14Mast.clocw.fits ThArMedusa1H14dbSub.extspyct.fits ThArMedusa1H14dbSub.extspnix.fits ThArMedusa1H14dbSub.extsp.bbres.fits ThArMedusa1H14dbSub.extsp.iires.fits ThArMedusa1H14dbSub.extsp.lines.tfits ThArMedusa1H14dbSub.extsp.lprms.dump ThArMedusa1H14dbSub.extsp.ocres.fits ThArMedusa1H14dbSub.extsp.scres.fits ThArMedusa1H14dbSub.extsp.sdres.fits ThArMedusa1H14dbSub.extsp.wdres.fits ThArMedusa1H14dbSub.extsp.wsx.fits ThArMedusa1H14dbSub.extsp.wsx.tfits ThArMedusa1H14dbSub.extsp.xcres.fits ThArMedusa1H14dbSub.extsp.xmres.fits ThArMedusa1H14dbSub.extsp.xores.fits ThArMedusa1H14dbSub.extsp.xores.tfits ThArMedusa1H14dbSub.extsp.xxres.fits ThArMedusa1H14dbSub.extsp.xmres.fits ThArMedusa1H14dbSub.extsp.ymres.fits ThArMedusa1H14dbSub.extsp.yores.fits ThArMedusa1H14dbSub.extsp.yyres.fits xcres.fits xmres.fits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

4.6.5 wcalOnly pipe

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                      and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | row 'section>=4' \
        | ficom wcalMast,_MAST_,_REB_,_PLOT_,_PUTZ_,_RAW_,_EMETHOD_ \
```

```
        | rr VERB=_VERB_ REBSCALE=_REBSCALE_ MAST=_MAST_ OBJ=_OBJ_ PLOTOPT=_PLOTOPT_ WSLIT=_WSLIT_ RAW=_RAW_ \
         OBJ=ThAr PLOT=_PLOT_ WLTHRESH=_WLTHRESH_ LINES=_LINES_ WXRES=_WXRES_ WPLOTSC=_WPLOTSC_ \
         WNMAXLINE=_WNMAXLINE_ WSSIGMA=_WSSIGMA_ WIND=_WIND_ RSTEP=_RSTEP_ WGFLAG=_WGFLAG_ WFFLAG=_WFFLAG_ \
         NAM=_NAM_ SLIT=_SLIT_ RES=_RES_ WCHEBXY=_WCHEBXY_ _EMETHOD_ CCGRAPH=_CCGRAPH_ \
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.wcalOnly)
    This is the last step of 'wcalMast', mainly for maintenance purpose. You may run this pipe
    after  'wcalMast  PUTZ=putz  ...'  ; wcalMast do not remove files necessary for subsequent
    re-calibrations.

    CAUTION:  if  MAST=wcalmast  is  specified,  the resulting w-solution overwrites
    existing solution in $GIR_CALIB.

    STANDARD OPERATION - update existing solution

      Only input file has to be specified  RAW=xxx

    STARTING FROM SCRATCH (Maintenance):

      see description inder wcalMast

    # blecha 24/06/2003
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.wcalOnly)
```

Pipe definition

Name	wcalOnly
Section	Calibration
Purpose	Get standard Wavelength calibration from extracted ThAr spectra

Inputs

nu	file_name	relevant_pythone_function(internal_nu)
01	$GIR_CALIB/girSlitGeoMedusa1H14.tfits	giRecNWcal2.py (600)
02	$GIR_CALIB/girWavCoefMedusa1H14.tfits	giRecNWcal2.py (600)
03	$GIR_CALIB/girLineThAr.tfits	giRecNWcal2.py (600)
04	ThArMedusa1H14dbSub.extsp.fits	giRecNWcal2.py (600)
05	ThArMedusa1H14dbSub.extsperr.fits	giRecNWcal2.py (600)
06	ThArMedusa1H14dbSub.clocy.fits	giRecNWcal2.py (600)
07	ThArMedusa1H14dbSub.clocw.fits	giRecNWcal2.py (600)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	$GIR_CALIB/girWavCoefMedusa1H14.tfits	copy giRecNWcal2.py (605)
02	ThArMedusa1H14dbSub.extsp.rebinlin.fits	giRecBinn2.py (630)
03	ThArMedusa1H14dbSub.extsperr.rebinlin.fits	giRecBinn2.py (630)

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe wcalOnly RES=H14 SLIT=Medusa1
pipe wcalOnly RAW=../raw/GIRAF.2003-01-28T10:17:25.736.fits
pipe wcalOnly RAW=../raw/GIRAF.2003-01-28T10:17:25.736.fits PUTZ=putz MAST=wcalmast,slitmast PLOT=no

Options

nu name=[default]
Option possible values; option description

01 MAST=[wcalmast]
wcalmast,no; produces new master wavelength solution and/or if 'slitmast' new slit geometry master. The later must be used with option REB=rebin

02 PLOT=[wcal3]
ext,wcal1,wcal3,rebin1,rebin5,rebin10,rebin20,wcalline1,wcalline2; coma-separated list of plot selection

*03 PUTZ=putz[no]
putz,no; remove all intermediate data

04 REB=[rebin]
rebin,no; W-rebinning after the W-calibration;

05 WGFLAG=[1]
0,1,-1; Flag of linesearc control plot in W-cal; 1 - we plot line search, 0-we do not plot, -1 only plot (no attempt to get W-solution)

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 EMETHOD=[SUMM]
SUMM, OPTIMAL; Extraction method

02 LINES=[girLine_OBJ_.tfits]
(girLineThAr.tfits,girLineSun.tfits); table of spectral line; see also 'ls $GIR_CALIB/girLine*.tfits'; Line table name for plots or W-calibrations located in $GIR_CALIB

03 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_

04 OBJ=[ThAr]
Fixed to ThAr; This is the first part of the names of I/O images

05 PLOTOPT=[None]
'-Lnnn'; any ad-option to 'bigg' commands; Posible usage: '-Lnnn' log scale with nnn offset

06 RAW=[None]
Filename; Full name of the RAW input image (.fits may be ommitted); In some cases coma-separated list is allowed - see description of the specific pipe. Note that even if you do not start from raw image (wcalOnly,calsim,...), the RAW could be given instead of RES OBJ and SLIT which will be then taken from KWs of raw image.

07 REBSCALE=[lin]
lin,log; W-rebinning scale linear or logarithmic; Must be cosistent with slitGeometry adjustement scale

*08 RES=H14[None]
H1,H2,...H22 or L1,l2,...L8; Resolution range; If not given derived from KW: 'INS FILT NAME'. See also $GIR_CALIB/girGratingHR316/LR600.tfits (do: egrep -v "#" < $GIR_CALIB/girGratingHR316.tfits

09 RSTEP=[-]
'-' or number; Rebinnig step in [nm]; '-' resolution-depedent default will use 0.005 in HR and 0.02 in LR

*10 SLIT=Medusa1[-]
Medusa1/2,IFU1/2,Argus; Geometry of input slit. If not given is taken from KW: 'INS2 SLIT NAME'.

11 VERB=[None]
None,-v; general verbosity for commands;

nu name=[default]
Parameter possible values; parameter description

12 WCHEBXY=[6,4]
0-10,0-10; W-calibration Chebyshev correction; degree of 2D polynomyal correction to optical model in W-calibration

13 WFFLAG=[0,0,0,0,0,1,1,1]
Fixed - only for maintenance; Flags governing the W-fit; in the following order: colimator focale, camera magn., grating angle, ordre, groove density, slit diplacements: dx,dy,angle. In Normal operation only slit diplacements flags should be set to fit. The colimator focale and the grating angle should only be adjusted after a major changes on instrument.

14 WIND=[15]
25,20,15; Width [px] of search window; coma-separated list of width of window for repeated search of line

15 WLTHRESH=[1]
1-10; multiple of BIASSIGMA; Threshold for use of the spectral line

16 WNMAXLINE=[80]
10-200; maximum number of spectral lines; During the W-calibration, take only WNMAXLINE number of brigtest spectral lines

17 WPLOTSC=[5]
1-20; Control of W-residuals plot; Max/min [px] of the graphic showing the residuals

18 WSLIT=[-]
Maintenance only; Initial x,y,phi slit displacement in wavelength solution after spectro adjustement; 1-3 coma-separated params.

19 WSSIGMA=[200]
200; Sigma clipping for Chebyshev; Spectral lines with position |Chebyshev model - position | > WSSIGMA are rejected - TB found better parameter

20 WXRES=[None]
no; if 'no' we do not use existing W-calibration; if nothing specified we use existing W-calibration in $GIR_CALIB, if we do not want it (or if no calibration exists) it must be 'no'

Call to functions

nu	call to Python or UNIX function
600	**giRecNWcal2.py** - -noquiet - -lthresh 1 - -xres - -graph 1 - -wssigma 200 - -linmod psfexp - -lexpwid -3 - -lwidth 15 - -fflags 0, 0, 0, 0, 0, 1, 1, 1 - -lbright 80 - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -wltable girLineThAr.tfits - -nosubsfit - -xworder 2, 2 - -wsorder 6, 4 -w - -slit - ThArMedusa1H14dbSub.extsp.fits ThArMedusa1H14dbSub.clocy.fits
605	cp ThArMedusa1H14dbSub.extsp.xores.tfits $GIR_CALIB/girWavCoefMedusa1H14.tfits
610	**biggExt.py** - -OUTPUT= , X11ThArMedusa1H14.Wopmodel -A -S 3 -Y 0: -0 -N 0: -0:1:0.21 -Z -5:5 - -xlabel 'X pixel' - -ylabel "X optModel -gausFit in px Medusa1H14" -D ThArMedusa1H14dbSub.extsp.xmres.fits ThArMedusa1H14dbSub.extsp.xxres.fits
611	**biggExt.py** - -OUTPUT= , X11ThArMedusa1H14.Wcheb -A -S 3 -Y 0: -0 -N 0: -0:1:0.21 -Z -1:1 - -xlabel 'X pixel' - -ylabel "X Chebyshev (6, 4) -gausFit in px (wind 15) Medusa1H14" -D ThArMedusa1H14dbSub.extsp.xcres.fits ThArMedusa1H14dbSub.extsp.xxres.fits
630	**giRecBinn2.py** -w - -bnstep - - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -bnlin - -bnrange= setup - -bnmethod linear ThArMedusa1H14dbSub.extsp.fits - ThArMedusa1H14dbSub.clocy.fits -

nu	call to Python or UNIX function
100020	rm ThArMedusa1H14??.fits ThArMedusa1H14bSub??.fits ThArMedusa1H14bSub1??.fits Medusa1H14Mast.clocy.fits Medusa1H14Mast.clocw.fits ThArMedusa1H14dbSub.extspyct.fits ThArMedusa1H14dbSub.extspnix.fits ThArMedusa1H14dbSub.extsp.bbres.fits ThArMedusa1H14dbSub.extsp.iires.fits ThArMedusa1H14dbSub.extsp.lines.tfits ThArMedusa1H14dbSub.extsp.lprms.dump ThArMedusa1H14dbSub.extsp.ocres.fits ThArMedusa1H14dbSub.extsp.scres.fits ThArMedusa1H14dbSub.extsp.sdres.fits ThArMedusa1H14dbSub.extsp.wdres.fits ThArMedusa1H14dbSub.extsp.wsx.fits ThArMedusa1H14dbSub.extsp.wsx.tfits ThArMedusa1H14dbSub.extsp.xcres.fits ThArMedusa1H14dbSub.extsp.xmres.fits ThArMedusa1H14dbSub.extsp.xores.fits ThArMedusa1H14dbSub.extsp.xores.tfits ThArMedusa1H14dbSub.extsp.xxres.fits ThArMedusa1H14dbSub.extsp.xmres.fits ThArMedusa1H14dbSub.extsp.ymres.fits ThArMedusa1H14dbSub.extsp.yores.fits ThArMedusa1H14dbSub.extsp.yyres.fits xcres.fits xmres.fits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

4.6.6 wcalSlit pipe

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                      and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | ficom wcalSlit,_RAW_,_MAST_,_EMETHOD_,_REB_ \
        | rr VERB=_VERB_ RAW=_RAW_ REBSCALE=_REBSCALE_ NSP=_NSP_ OBJ=_OBJ_ CDOM=_CDOM_ CORDV=_CORDV_ \
         CDOM=_CDOM_ RSTEP=_RSTEP_ NAM=_NAM_ SLIT=_SLIT_ RES=_RES_ CCGRAPH=_CCGRAPH_ \
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.wcalSlit)
      We  assume  that  we  have  ThAr wavelength-calibrated and rebinned spectra, obtained by
      wcalMast with 'REB=rebin' option.

      Using the Crosscorelation, we produce new updated girSlitGeo_SLIT__RES_.tfits table. The
      inversion  of  the  optical model being only approximate, when used the first time after
      optical  modification  of  the  spectrograph,  this  procedure has to be repaeted till a
      negligible RV difference between individual spectra is reached.

    #AB Sun May  4 20:29:33 CEST 2003
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.wcalSlit)
```

Pipe definition

Name	wcalSlit
Section	Calibration
Purpose	Update slit gemetry using rebinned Wavelength calibration
Priority	After, or as an option, successfull wcalMast

Inputs

nu	file_name	relevant_pythone_function(internal_nu)
01	$GIR_CALIB/girThArH14.tfits	giCrossC.py (650)
02	ThArMedusa1H14dbSub.extsp.rebinlin.fits	giCrossC.py (650)
03	$GIR_CALIB/girWavCoefMedusa1H14.tfits	giRecBinn2.py (700)
04	ThArMedusa1H14dbSub.extsp.fits	giRecBinn2.py (700)
05	ThArMedusa1H14dbSub.extsperr.fits	giRecBinn2.py (700)
06	ThArMedusa1H14dbSub.clocy.fits	giRecBinn2.py (700)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	$GIR_CALIB/girSlitGeoMedusa1H14.tfits	copy giCrossC.py (655)
02	ThArMedusa1H14dbSub.extsp.rebinlin.fits	giRecBinn2.py (700)
03	ThArMedusa1H14dbSub.extsperr.rebinlin.fits	giRecBinn2.py (700)

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe wcalSlit RAW=../raw/GIRAF.2003-01-28T10:17:25.736.fits

Options

nu name=[default]
Option possible values; option description

01 MAST=[slitmast]
slitmast,no; Produce new slit geometry master
02 REB=[no]
rebin,no; W-rebinning before Correlation; If it was not done during wcalMast

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 CCGRAPH=[5]
(0-5); list of graphs for Correlation; see giCrossC.py help for more explanations
02 CDOM=[0.1:-0.1]
Correlation domain [px] or fractions of domain (0:-0 for full spectra); Correlation domain [px] or fractions of domain (0:-0 for full spectra);
03 CORDV=[200]
Correlation: +-DV[km/sec] where we compute the correlation; Correlation: +-DV[km/sec] where we compute the correlation
04 EMETHOD=[SUMM]
SUMM, OPTIMAL; Extraction method
05 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_
06 NSP=[0:-0]
SP1:SP2:SPSTEP; Spectra TB processed (0:-0 for all spectra) see also help for giCrossC.py
07 OBJ=[ThAr]
Fixed to ThAr; This is the first part of the names of I/O images
*08 RAW=../raw/GIRAF.2003-01-28T10:17:25.736.fits[None]
Filename; Full name of the RAW input image (.fits may be ommitted); In some cases coma-separated list is allowed - see description of the specific pipe. Note that even if you do not start from raw image (wcalOnly,calsim,...), the RAW could be given instead of RES OBJ and SLIT which will be then taken from KWs of raw image.
09 REBSCALE=[lin]
lin,log; W-rebinning scale linear or logarithmic; Must be cosistent with slitGeometry adjustement scale
*10 RES=H14[None]
H1,H2,...H22 or L1,l2,...L8; Resolution range; If not given derived from KW: 'INS FILT NAME'. See also $GIR_CALIB/girGratingHR316/LR600.tfits (do: egrep -v "#" < $GIR_CALIB/girGratingHR316.tfits

nu name=[default]
Parameter possible values; parameter description

11 RSTEP=[-]
'-' or number; Rebinnig step in [nm]; '-' resolution-depedent default will use 0.005 in HR and 0.02 in LR

*12 SLIT=Medusa1[-]
Medusa1/2,IFU1/2,Argus; Geometry of input slit. If not given is taken from KW: 'INS2 SLIT NAME'.

13 VERB=[None]
None,-v; general verbosity for commands;

Call to functions

nu	call to Python or UNIX function
650	**giCrossC.py** - -titles 'RV of ThAr with current master slit for Medusa1 H14' -V 200 - -Vcorr no - -graph 5 - -cdomain 0.1: -0.1 - -xf H14 - -txts -T girThArH14.tfits -N '0: -0' ThArMedusa1H14dbSub.extsp.rebinlin.fits
655	cp girSlitGeoMedusa1H14.tfits $GIR_CALIB
700	**giRecBinn2.py** -w - -bnstep - - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -bnlin - -bnrange= setup - -bnmethod linear ThArMedusa1H14dbSub.extsp.fits - ThArMedusa1H14dbSub.clocy.fits -
703	**giCrossC.py** - -titles 'RV of ThAr with NEW master slit for Medusa1 H14' -V 200 - -Vcorr no - -graph 5 - -cdomain 0.1: -0.1 - -txts -T girThArH14.tfits -N '0: -0' ThArMedusa1H14dbSub.extsp.rebinlin.fits
100031	rm girSlitGeoMedusa1H14.tfits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

4.7 Preprocessing Pipes

LIST of Pipes

Name	Purpose
preproc	Preprocess raw image

4.7.1 PREPROC PIPE

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                      and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | row 'section<3 \
        | \
        | section>=10' \
        | ficom extract,_SLM_,_CHECK_,_NSP0_,_NSP0_,_SLIT_,_COSMICM_,_PLOT_,_PUTZ_,_DARK_ \
        | rr VERB=_VERB_ PLOT=_PLOT_ NSP0=_NSP0_ CHECK=_CHECK_ SLM=_SLM_ SLSTEP=_SLSTEP_ SLXORDER=_SLXORDER_ \
         SLYORDER=_SLYORDER_ NAM=_NAM_ OBJ=_OBJ_ SLIT=_SLIT_ RES=_RES_ RAW=_RAW_ \
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.preproc)
    Preprocess  RAW  image.  This  is  a  first part of 'extract' which ends BEFORE extraction
    itself. If you want to try various extraction, W-calibration and binning, it could be a
    convenient way to split processing.

    #AB Mon May  5 09:21:44 MEST 2003
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.preproc)
```

Pipe definition

Name	preproc
Section	Preprocessing
Purpose	Preprocess raw image

Inputs

nu	file_name	relevant_pythone_function(internal_nu)
01	../raw/GIRAF.2003-01-28T05:49:49.211.fits	giCheck.py (21)
02	$GIR_CALIB/maskMast.fits	giRecBias.py (100)
03	$GIR_CALIB/biasMast.fits	giRecBias.py (100)
04	$GIR_CALIB/biasDiffMast.fits	giArithm.py (101)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	NGC2243KAL5Medusa1H14bSub.fits	giArithm.py (150)

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe preproc RAW=../raw/GIRAF.2003-01-28T05:49:49.211.fits
pipe preproc RAW=‘showf FF R| files -5‘ PUTZ=putz

Options

nu name=[default]
Option possible values; option description

01 CHECK=[check]
check,nochek,noprep; Do the check-in; and/or preprocessing (write missing KW’s ...) necessary with raw giraffe files
02 COSMICM=[nocosmicm]
cosmicm or nocosmicm; remove cosmic before extraction on multiple images
03 DARK=[no]
dark, no; remove dark; we remove also dark
04 PLOT=[-]
ext,slmlplot (with SLM option),rebin1,rebin5,rebin10,extreb; coma-separated list of plot selection
*05 PUTZ=putz[no]
putz,no; remove all intermediate data
06 SLM=[slmnoremove]
slmnoremove,slmremove; Remove scattered ligth

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_

nu name=[default]
Parameter possible values; parameter description

02 NSP0=[0]
Fixed - only for test; force number of spectra on the frame to this values. TBU ONLY when OzPoz tables is missing.

*03 OBJ='NGC2243KAL5'[None]
ThAr,Fff,Sun,Any_object; taken from KW 'OBS TARG NAME' if not given; This is the first part of the names of I/O images

*04 RAW=../raw/GIRAF.2003-01-28T05:49:49.211.fits[None]
Filename; Full name of the RAW input image (.fits may be ommitted); In some cases coma-separated list is allowed - see description of the specific pipe. Note that even if you do not start from raw image (wcalOnly,calsim,...), the RAW could be given instead of RES OBJ and SLIT which will be then taken from KWs of raw image.

*05 RES=H14[None]
H1,H2,...H22 or L1,l2,...L8; Resolution range; If not given derived from KW: 'INS FILT NAME'. See also $GIR_CALIB/girGratingHR316/LR600.tfits (do: egrep -v "#" < $GIR_CALIB/girGratingHR316.tfits

*06 SLIT=Medusa1[-]
Medusa1/2,IFU1/2,Argus; Geometry of input slit. If not given is taken from KW: 'INS2 SLIT NAME'.

07 VERB=[None]
None,-v; general verbosity for commands;

Call to functions

nu	call to Python or UNIX function
21	**giCheck.py** -w -K DARKVALUE= 0. - -on NGC2243KAL5Medusa1H14.fits ../raw/GIRAF.2003 -01 -28T05:49:49.211.fits
100	**giRecBias.py** - -bsigma 100 - -blimits 10:40:0:4096, 2120:2130:0:4096 -w - -bmethod ZMASTER +CURVE - -xorder 10 - -yorder 1 -b $GIR_CALIB/maskMast.fits - -on NGC2243KAL5Medusa1H14bSub1.fits 'NGC2243KAL5Medusa1H14??.fits' $GIR_CALIB/biasMast.fits
101	**giArithm.py** -w - -STAT - -on "NGC2243KAL5Medusa1H14bSub.fits= NGC2243KAL5Medusa1H14bSub1??.fits +$GIR_CALIB/biasDiffMast.fits"
150	**giArithm.py** -w - -STAT 'NGC2243KAL5Medusa1H14bSub.fits= average(NGC2243KAL5Medusa1H14bSub??.fits)'
195	mv NGC2243KAL5Medusa1H14bSub.fits NGC2243KAL5Medusa1H14dbSub.fits
100030	rm FffMedusa1H14Mast.extsp.fits NGC2243KAL5Medusa1H14??.fits NGC2243KAL5Medusa1H14bSub??.fits NGC2243KAL5Medusa1H14dbSub.fits NGC2243KAL5Medusa1H14bSub1??.fits NGC2243KAL5Medusa1H14bSubWithSL.fits NGC2243KAL5Medusa1H14dbSub.slimg.fits NGC2243KAL5Medusa1H14dbSub.locd.fits NGC2243KAL5Medusa1H14dbSub.extspyct.fits NGC2243KAL5Medusa1H14dbSub.extspnix.fits NGC2243KAL5Medusa1H14dbSub.wlenRes.tfits Medusa1H14Mast.psfcent.fits Medusa1H14Mast.psfexpo.fits Medusa1H14Mast.psfwidt.fits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

4.8 Extraction Pipes

LIST of Pipes

Name	Purpose
extract	From raw image to W-calibrated rebinned spectra
rebin	Rebin extracted spectra

4.8.1 EXTRACT PIPE

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                      and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | ficom \
        | rr VERB=_VERB_ REBSCALE=_REBSCALE_ LOCADJUST=_LOCADJUST_ PLOTOPT=_PLOTOPT_ IMAGOPT=_IMAGOPT_ \
         RSTEP=_RSTEP_ NSP0=_NSP0_ NFF=_NFF_ CHECK=_CHECK_ SLSTEP=_SLSTEP_ CCGRAPH=_CCGRAPH_ \
         SLXORDER=_SLXORDER_ SLYORDER=_SLYORDER_ NAM=_NAM_ OBJ=_OBJ_ SLIT=_SLIT_ RES=_RES_ RAW=_RAW_ \
         EMETHOD=_EMETHOD_ HSIGMA=_HSIGMA_ \
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.extract)
    DESCRIPTION

      This  carry  on  the  extraction  from raw file to extracted spectra. Optionally, and if
      wavelength solution and/or extracted FFs for this SLIT+SETUP exist, we do also rebinning
      and  simultaneous  W-calibration adjustment using the simultaneous W-calibration spectra
      and  finaly for IFU/Argus mode the image reconstruction (option IMAG=imag). Probably the
      best use of image reconstruction is directly through single Python command giRecImag.py.

      This pipe includes through options all available processing of science data:

        .Input of multiple files averaged after the initial check-in and bias subtraction.

        . Build  of  scattered-light  2-D polynomial model using the inter-spectra regions and
          model subtraction

        . Localization adjustment using 5 - simultaneous W-calibration spectra.

        . Virutual slit extraction (SUMM) and optimal extraction (OPTIMAL)

        . 'Narrow' flat-fielding on extracted spectra

        . Rebinning at linear or logarithmic scale

        . W-calibration  adjustment  through  CrossCorrelation of 5 simultaneous W-calibration
          spectra and  final rebinning

      Note that option compatibility with data is not checked and it is under responsibility
      of the user to fulfill the following requirements:

        option             requirement
        ------------------ ------------------------------------------
        AVER=cosmic        at least 3 spectra on input
        LOCADJUST          simultaneous  W-calibration must be present and standard W-solution
                           must exist for the setup
        CALSIM,LOCADJUST   simultaneous W-calibration must be present
        EMETHOD=OPTIMAL    we recommend LOCADJUST=locadjust

    # blecha Tue 06/24/03
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.extract)
```

Pipe definition

Name	extract
Section	Extraction
Purpose	From raw image to W-calibrated rebinned spectra

Inputs

nu	file_name	relevant_pythone_function(internal_nu)

nu	file_name	relevant_pythone_function(internal_nu)
01	../raw/GIRAF.2003-01-28T05:49:49.211.fits	giCheck.py (21)
02	$GIR_CALIB/FF/Medusa1H14Mast.clocy.fits	copy from CALIB (32)
03	$GIR_CALIB/FF/Medusa1H14Mast.clocw.fits	copy from CALIB (32)
04	$GIR_CALIB/FF/Medusa1H14Mast.psfcent.fits	copy from CALIB (32)
05	$GIR_CALIB/FF/Medusa1H14Mast.psfexpo.fits	copy from CALIB (32)
06	$GIR_CALIB/FF/Medusa1H14Mast.psfwidt.fits	copy from CALIB (32)
07	$GIR_CALIB/maskMast.fits	giRecBias.py (100)
08	$GIR_CALIB/biasMast.fits	giRecBias.py (100)
09	$GIR_CALIB/biasDiffMast.fits	giArithm.py (101)
10	$GIR_CALIB/girSlitGeoMedusa1H14.tfits	giRecBinn2.py (630)
11	$GIR_CALIB/girWavCoefMedusa1H14.tfits	giRecBinn2.py (630)
12	$GIR_CALIB/girThArH14.tfits	giCrossC.py (705)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	NGC2243KAL5Medusa1H14dbSub.clocy.fits	copy from CALIB (32)
02	NGC2243KAL5Medusa1H14dbSub.clocw.fits	copy from CALIB (32)
03	NGC2243KAL5Medusa1H14bSub.fits	giArithm.py (150)
04	NGC2243KAL5Medusa1H14dbSub.oxtsp.fits	giRecNExt.py (502)
05	NGC2243KAL5Medusa1H14dbSub.oxtsperr.fits	giRecNExt.py (502)
06	NGC2243KAL5Medusa1H14dbSub.oxtsp.rebinlin.fits	giRecBinn2.py (630)
07	NGC2243KAL5Medusa1H14dbSub.oxtsperr.rebinlin.fits	giRecBinn2.py (630)
08	Control_plot	giCrossC.py (808)

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe extract RAW=../raw/GIRAF.2003-01-28T05:49:49.211.fits
pipe extract RAW=‘showf FF R | files -3‘ AVER=cosmic
pipe extract REB=no RAW=../raw/GIRAF.2003-01-28T05:49:49.211.fits SLM=slmremove

Options

nu name=[default]
Option possible values; option description

01 AVER=[average]
average,cosmic; how we compute the average; cosmic should not be used with < 3 files on input. The averaging is used whenever a coma-separated list of files is given as input (RAW).

02 CALSIM=[calsim]
calsim,no; apply simultaneous W-calibration; We crosscorrelate calsim spectra with ThAr mask and we rebinning using current correction _OBJ__SLIT__RES_dbSub.wlenRes.tfits

03 CHECK=[check]
check,nochek,noprep; Do the check-in; and/or preprocessing (write missing KW’s ...) necessary with raw giraffe files

04 CLEAN=[no]
clean,noclean; !!! delete all files except README.r in current directory at start; this is mainly for tests

05 DARK=[no]
dark, no; remove dark; we remove also dark

06 IMAG=[no]

nu name=[default]
Option possible values; option description

imag,no; image reconstruction; If 'imag' given we reconstruct the image and display IFU layout (IFU/Argus mode only)

07 LOCADJUST=[locadjust]
locajust,no; adjust localisation before extraction; we adjust the master localisation using the simultaneous calibration spectra - recommended when using optimal extraction

08 NFF=[nonff]
nff,nonff; Do FFielding on extracted spectra (narrow flat-fielding)

09 PLOT=[-]
ext,slmlplot (with SLM option),rebin1,rebin5,rebin10,extreb; coma-separated list of plot selection

*10 PUTZ=putz[no]
putz,no; remove all intermediate data

11 REB=[rebin]
rebin,no; W-rebinning using standard solution

12 SLM=[slmnoremove]
slmnoremove,slmremove; Remove scattered ligth

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 CCGRAPH=[5]
(0-5); list of graphs for Correlation; see giCrossC.py help for more explanations

02 EMETHOD=[OPTIMAL]
SUMM, OPTIMAL; Extraction method

03 HSIGMA=[6]
6-10; Optimal extraction sigma clipping; data with (psf-data)**2 > VAR*HSIGMA replaced by psf

04 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_

05 NSP0=[0]
Fixed - only for test; force number of spectra on the frame to this values. TBU ONLY when OzPoz tables is missing.

*06 OBJ='NGC2243KAL5'[None]
ThAr,Fff,Sun,Any_object; taken from KW 'OBS TARG NAME' if not given; This is the first part of the names of I/O images

*07 RAW=../raw/GIRAF.2003-01-28T05:49:49.211.fits[None]
Filename; Full name of the RAW input image (.fits may be ommitted); In some cases coma-separated list is allowed - see description of the specific pipe. Note that even if you do not start from raw image (wcalOnly,calsim,...), the RAW could be given instead of RES OBJ and SLIT which will be then taken from KWs of raw image.

08 REBSCALE=[lin]
lin,log; W-rebinning scale linear or logarithmic; Must be cosistent with slitGeometry adjustement scale

*09 RES=H14[None]
H1,H2,...H22 or L1,l2,...L8; Resolution range; If not given derived from KW: 'INS FILT NAME'. See also $GIR_CALIB/girGratingHR316/LR600.tfits (do: egrep -v "#" < $GIR_CALIB/girGratingHR316.tfits

10 RSTEP=[-]
'-' or number; Rebinnig step in [nm]; '-' resolution-depedent default will use 0.005 in HR and 0.02 in LR

*11 SLIT=Medusa1[-]

nu name=[default] **Parameter possible values**; parameter description
Medusa1/2,IFU1/2,Argus; Geometry of input slit. If not given is taken from KW: 'INS2 SLIT NAME'.
12 VERB=[None] **None,-v**; general verbosity for commands;

Call to functions

nu	call to Python or UNIX function
21	**giCheck.py** -w -K DARKVALUE= 0. - -on NGC2243KAL5Medusa1H14.fits ../raw/GIRAF.2003 -01 -28T05:49:49.211.fits
32	cp $GIR_CALIB/FF/Medusa1H14Mast.clocy.fits NGC2243KAL5Medusa1H14dbSub.clocy.fits; cp $GIR_CALIB/FF/Medusa1H14Mast.clocw.fits NGC2243KAL5Medusa1H14dbSub.clocw.fits; cp $GIR_CALIB/FF/Medusa1H14Mast.psfcent.fits $GIR_CALIB/FF/Medusa1H14Mast.psfexpo.fits $GIR_CALIB/FF/Medusa1H14Mast.psfwidt.fits .
100	**giRecBias.py** - -bsigma 100 - -blimits 10:40:0:4096, 2120:2130:0:4096 -w - -bmethod ZMASTER +CURVE - -xorder 10 - -yorder 1 -b $GIR_CALIB/maskMast.fits - -on NGC2243KAL5Medusa1H14bSub1.fits 'NGC2243KAL5Medusa1H14??.fits' $GIR_CALIB/biasMast.fits
101	**giArithm.py** -w - -STAT - -on "NGC2243KAL5Medusa1H14bSub.fits= NGC2243KAL5Medusa1H14bSub1??.fits +$GIR_CALIB/biasDiffMast.fits"
150	**giArithm.py** -w - -STAT 'NGC2243KAL5Medusa1H14bSub.fits= average(NGC2243KAL5Medusa1H14bSub??.fits)'
195	mv NGC2243KAL5Medusa1H14bSub.fits NGC2243KAL5Medusa1H14dbSub.fits
410	**giRecDLocfit.py** -w - -xorder 1 - -yorder 1 NGC2243KAL5Medusa1H14dbSub.fits NGC2243KAL5Medusa1H14dbSub.clocy.fits NGC2243KAL5Medusa1H14dbSub.clocw.fits; giArithm.py -w 'NGC2243KAL5Medusa1H14dbSub.clocy.fits= NGC2243KAL5Medusa1H14dbSub.clocy.fits +NGC2243KAL5Medusa1H14dbSub.locd.fits'
502	**giRecNExt.py** -w - -nofbkg - -pmaxdw 0.05 - -pdword 2 - -emethod OPTIMAL - -hsigma 6 - -nograph NGC2243KAL5Medusa1H14dbSub.fits NGC2243KAL5Medusa1H14dbSub.clocy.fits NGC2243KAL5Medusa1H14dbSub.clocw.fits - Medusa1H14Mast
630	**giRecBinn2.py** -w - -bnstep - - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -bnlin - -bnrange= setup - -bnmethod linear NGC2243KAL5Medusa1H14dbSub.oxtsp.fits - NGC2243KAL5Medusa1H14dbSub.clocy.fits -
705	**giCrossC.py** - -wres - -O CALSIM - -Vcorr no - -graph 5 - -cdomain 0.1: -0.1 -T girThArH14.tfits -N '0: -0' NGC2243KAL5Medusa1H14dbSub.oxtsp.rebinlin.fits
800	**giRecBinn2.py** - -wres - -w - -bnstep - - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -bnlin - -bnrange= setup - -bnmethod linear NGC2243KAL5Medusa1H14dbSub.oxtsp.fits - NGC2243KAL5Medusa1H14dbSub.clocy.fits -
808	**giCrossC.py** -O CALSIM - -Vcorr no - -graph 5 - -cdomain 0.1: -0.1 -T girThArH14.tfits -N '0: -0' NGC2243KAL5Medusa1H14dbSub.oxtsp.rebinlin.fits
100030	rm FffMedusa1H14Mast.oxtsp.fits NGC2243KAL5Medusa1H14??.fits NGC2243KAL5Medusa1H14bSub??.fits NGC2243KAL5Medusa1H14dbSub.fits NGC2243KAL5Medusa1H14bSub1??.fits NGC2243KAL5Medusa1H14bSubWithSL.fits NGC2243KAL5Medusa1H14dbSub.slimg.fits NGC2243KAL5Medusa1H14dbSub.locd.fits NGC2243KAL5Medusa1H14dbSub.oxtspyct.fits NGC2243KAL5Medusa1H14dbSub.oxtspnix.fits NGC2243KAL5Medusa1H14dbSub.wlenRes.tfits Medusa1H14Mast.psfcent.fits Medusa1H14Mast.psfexpo.fits Medusa1H14Mast.psfwidt.fits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

4.8.2 REBIN PIPE

```
Pipe definition files used: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst
                      and: /net/obssb55/export/diskB1/giraffe/SV/pipe/config/pipe0.r

Command:  lt < /net/obssb55/export/diskB1/giraffe/SV/pipe/config/t0.lst \
        | row 'section>3 \
```

```
        | \
        | section>=10' \
        | ficom extract,rebin,_EMETHOD_,_PLOT_,_SLIT_,_CALSIM_,_PUTZ_,_CLEAN_,_IMAG_ \
        | rr VERB=_VERB_ REBSCALE=_REBSCALE_ PLOTOPT=_PLOTOPT_ IMAGOPT=_IMAGOPT_ RSTEP=_RSTEP_ \
         CCGRAPH=_CCGRAPH_ NAM=_NAM_ OBJ=_OBJ_ SLIT=_SLIT_ RES=_RES_ RAW=_RAW_ EMETHOD=_EMETHOD_ \
        | pyrectest - -v \
PIPE short description
(Begin source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.rebin)
    Do rebinning using current master solution. See help for the rebinn giRecBinn2.py
    function.

    Comment to rebinning parameters.

      Default RSTEP has been chosen in a way to keep number of output pixels similar to the
      original sampling and also to have same sampling over all setups using the same grating
      (possibility to join rebinned spectra). It is 0.005 [nm] in high and 0.02 [nm] in low
      resolution. If you decide to use other then default value you should do it in a
      cosistent way for W-calibration and science data.

      Default REBSCALE is 'linear' and also the cross-correlation when adjusting the slit
      gometry for the setup is done on linearly rebined spectra (with an appropriate
      correction of the resulting shift). When the simultaneous calibrationn adjustement is
      made, it seems that the difference between the 'full log processing' - i.e. wcalMast and
      'extract' made with REBSCALE=log and the 'hybrid log processing' - i.e. only extract
      made with REBSCALE=log is negligible.

    #AB Mon May  5 09:21:44 MEST 2003
(End source: file /net/obssb55/export/diskB1/giraffe/SV/pipe/doc//README.rebin)
```

Pipe definition

Name	rebin
Section	Extraction
Purpose	Rebin extracted spectra

Inputs

nu	file_name	relevant_pythone_function(internal_nu)
01	$GIR_CALIB/girSlitGeoMedusa1H14.tfits	giRecBinn2.py (630)
02	$GIR_CALIB/girWavCoefMedusa1H14.tfits	giRecBinn2.py (630)
03	NGC2243KAL5Medusa1H14dbSub.oxtsp.fits	giRecBinn2.py (630)
04	NGC2243KAL5Medusa1H14dbSub.oxtsperr.fits	giRecBinn2.py (630)
05	NGC2243KAL5Medusa1H14dbSub.clocy.fits	giRecBinn2.py (630)
06	$GIR_CALIB/girThArH14.tfits	giCrossC.py (705)

Outputs

nu	file_name	relevant_pythone_function(internal_nu)
01	NGC2243KAL5Medusa1H14dbSub.oxtsp.rebinlin.fits	giRecBinn2.py (630)
02	NGC2243KAL5Medusa1H14dbSub.oxtsperr.rebinlin.fits	giRecBinn2.py (630)
03	Control_plot	giCrossC.py (808)

Examples

Executables from ..../TestData/reduced directory (data in ..../TestData/raw assumed)

pipe rebin RAW=../raw/GIRAF.2003-01-28T05:49:49.211.fits

Options

nu name=[default]
Option possible values; option description

01 CALSIM=[calsim]
calsim,no; apply simultaneous W-calibration; We crosscorrelate calsim spectra with ThAr mask and we rebinning using current correction _OBJ__SLIT__RES_dbSub.wlenRes.tfits

02 CLEAN=[no]
clean,noclean; !!! delete all files except README.r in current directory at start; this is mainly for tests

03 IMAG=[no]
imag,no; image reconstruction; If 'imag' given we reconstruct the image and display IFU layout (IFU/Argus mode only)

04 PLOT=[-]
ext,slmlplot (with SLM option),rebin1,rebin5,rebin10,extreb; coma-separated list of plot selection

05 PLOT=[-]
rebin1,rebin5,rebin10,rebin20; coma-separated list of plot selection

*06 PUTZ=putz[no]
putz,no; remove all intermediate data

Parameters

Note that parameter list is option-dependent. The list of parameter given in this documentation is associated with the first *example* given above. Note also that parameter and option values are preceded by * if default value has been changed (see 1-st exemple).

nu name=[default]
Parameter possible values; parameter description

01 CCGRAPH=[5]
(0-5); list of graphs for Correlation; see giCrossC.py help for more explanations

02 EMETHOD=[OPTIMAL]
SUMM, OPTIMAL; Extraction method

03 NAM=[None]
Any name; last part of the filename; User defined component of the filename of structured part of the filename _OBJ__SLIT__RES__NAM_

*04 OBJ='NGC2243KAL5'[None]
ThAr,Fff,Sun,Any_object; taken from KW 'OBS TARG NAME' if not given; This is the first part of the names of I/O images

05 REBSCALE=[lin]
lin,log; W-rebinning scale linear or logarithmic; Must be cosistent with slitGeometry adjustement scale

*06 RES=H14[None]
H1,H2,...H22 or L1,l2,...L8; Resolution range; If not given derived from KW: 'INS FILT NAME'. See also $GIR_CALIB/girGratingHR316/LR600.tfits (do: egrep -v "#" < $GIR_CALIB/girGratingHR316.tfits

07 RSTEP=[-]
'-' or number; Rebinnig step in [nm]; '-' resolution-depedent default will use 0.005 in HR and 0.02 in LR

*08 SLIT=Medusa1[-]
Medusa1/2,IFU1/2,Argus; Geometry of input slit. If not given is taken from KW: 'INS2 SLIT NAME'.

09 VERB=[None]
None,-v; general verbosity for commands;

Call to functions

nu	call to Python or UNIX function
630	**giRecBinn2.py** -w - -bnstep - - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -bnlin - -bnrange= setup - -bnmethod linear NGC2243KAL5Medusa1H14dbSub.oxtsp.fits - NGC2243KAL5Medusa1H14dbSub.clocy.fits -
705	**giCrossC.py** - -wres - -O CALSIM - -Vcorr no - -graph 5 - -cdomain 0.1: -0.1 -T girThArH14.tfits -N '0: -0' NGC2243KAL5Medusa1H14dbSub.oxtsp.rebinlin.fits
800	**giRecBinn2.py** - -wres - -w - -bnstep - - -sgtable $GIR_CALIB/girSlitGeoMedusa1H14.tfits - -bnlin - -bnrange= setup - -bnmethod linear NGC2243KAL5Medusa1H14dbSub.oxtsp.fits - NGC2243KAL5Medusa1H14dbSub.clocy.fits -
808	**giCrossC.py** -O CALSIM - -Vcorr no - -graph 5 - -cdomain 0.1: -0.1 -T girThArH14.tfits -N '0: -0' NGC2243KAL5Medusa1H14dbSub.oxtsp.rebinlin.fits
100030	rm FffMedusa1H14Mast.oxtsp.fits NGC2243KAL5Medusa1H14??.fits NGC2243KAL5Medusa1H14bSub??.fits NGC2243KAL5Medusa1H14dbSub.fits NGC2243KAL5Medusa1H14bSub1??.fits NGC2243KAL5Medusa1H14bSubWithSL.fits NGC2243KAL5Medusa1H14dbSub.slimg.fits NGC2243KAL5Medusa1H14dbSub.locd.fits NGC2243KAL5Medusa1H14dbSub.oxtspyct.fits NGC2243KAL5Medusa1H14dbSub.oxtspnix.fits NGC2243KAL5Medusa1H14dbSub.wlenRes.tfits Medusa1H14Mast.psfcent.fits Medusa1H14Mast.psfexpo.fits Medusa1H14Mast.psfwidt.fits

(End source: file pipeD0.lst/formated by pipeDFun.fmt.tex)

Functions Index

B

biggExt.py 128, 131
biggLoc.py 122

G

giAdjustLoc 43
giAdjustSL 31
giArithm.py 12, 116–118, 122, 128, 136, 140
giArithmetics 11
giCheck.py 116, 118, 122, 128, 136, 140
giCrossC 80
giCrossC.py 82, 134, 140, 143
giDetectCosmicM 25
giExtractSpectra 56
giFitPsfProfile 49
giGetRemoveBias 15
giGetWaveSolution 64
giLocalSpectra 36
giRebinSpectra 74
giRecBias.py 19, 116, 118, 122, 128, 136, 140
giRecBinn2.py 77, 128, 131, 134, 140, 143
giRecCosmicMC.py 29, 118
giRecDark.py 23
giRecDLocfit.py 47, 140
giRecNExt.py 61, 122, 128, 140
giRecNLoc2.py 39, 122
giRecNWcal2.py 69, 128, 131
giRecPsfLoc.py 54, 122
giRecSLfit.py 33
giSubtractDark 21

Data Tables Index

G

`girDLoccTable`
- chebDY
  - *giAdjustLoc* . . . 46
- girDLocc . . . 92

`girGratingTable`
- girGrating . . . 93
- gratingMast
  - *giGetWaveSolution* . . . 66

`girLineTable`
- girLine . . . 95
- linesWlampMast
  - *giAdjustLoc* . . . 44
  - *giGetWaveSolution* . . . 66

`girLoccTable`
- chebY
  - *giLocalSpectra* . . . 37
- girLocc . . . 95
- pLocc
  - *giFitPsfProfile* . . . 52

`girOzpozTable`
- girOzpoz . . . 97

`girRVMaskTable`
- girRVMask . . . 98
  - *giCrossC* . . . 81

`girSlitGeoTable`
- girSlitGeo . . . 99
  - *giAdjustLoc* . . . 44
  - *giCrossC* . . . 81
  - *giGetWaveSolution* . . . 66
  - *giRebinSpectra* . . . 75

`girWavCoefTable`
- girWavCoef . . . 101
  - *giAdjustLoc* . . . 44
  - *giGetWaveSolution* . . . 66, 67
  - *giRebinSpectra* . . . 75

`girWresTable`
- girWres . . . 103
  - *giAdjustLoc* . . . 44
  - *giCrossC* . . . 81
  - *giRebinSpectra* . . . 75

Inputs/Outputs Index

A

averageCrhBadPixMap
Output of:
giDetectCosmicM . . . 26
averageImage
Output of:
giDetectCosmicM . . . 26

B

badPixMap
Input of:
giAdjustSL . . . 32
giLocalSpectra . . . 36
Output of:
giArithmetics . . . 12
badPixMap1
Input of:
giArithmetics . . . 11
badPixMap2
Input of:
giArithmetics . . . 11
badPixMast
Input of:
giGetRemoveBias . . . 15
giSubtractDark . . . 21
biasMast
Input of:
giGetRemoveBias . . . 15
bSubImage
Input of:
giSubtractDark . . . 21
Output of:
giGetRemoveBias . . . 16

C

chebDY
Output of:
giAdjustLoc . . . 46
chebY
Output of:
giLocalSpectra . . . 37
crhBadPixMap
Output of:
giDetectCosmicM . . . 26
crhmCount
Output of:
giDetectCosmicM . . . 26
curBadPixMap
Input of:
giDetectCosmicM . . . 25
giExtractSpectra . . . 58

D

darkMast
Input of:
giSubtractDark . . . 21
dbSubFrame
Input of:
giAdjustLoc . . . 43
giFitPsfProfile . . . 49
giLocalSpectra . . . 36
dbSubImage
Input of:
giDetectCosmicM . . . 25
giExtractSpectra . . . 56
Output of:
giSubtractDark . . . 22
dLocY
Output of:
giAdjustLoc . . . 45

E

exSp
Input of:
giGetWaveSolution . . . 64
Output of:
giExtractSpectra . . . 59
exSpErr
Input of:
giGetWaveSolution . . . 65
Output of:
giExtractSpectra . . . 59
exSpNpixels
Output of:
giExtractSpectra . . . 59
extLocY
Output of:
giExtractSpectra . . . 59

F

ffSp
Input of:
giRebinSpectra . . . 74
ffSpErr
Input of:
giRebinSpectra . . . 74

G

girRVMask
Input of:

giCrossC 81
girSlitGeo
Input of:
giAdjustLoc 44
giCrossC 81
giGetWaveSolution 66
giRebinSpectra 75
Output of:
giCrossC 81
girWavCoef
Input of:
giAdjustLoc 44
giGetWaveSolution 66
giRebinSpectra 75
Output of:
giGetWaveSolution 67
girWres
Input of:
giAdjustLoc 44
giRebinSpectra 75
Output of:
giCrossC 81
gratingMast
Input of:
giGetWaveSolution 66

I

inFrame1
Input of:
giArithmetics 11
inFrame2
Input of:
giArithmetics 11

L

linesWlampMast
Input of:
giAdjustLoc 44
giGetWaveSolution 66
LocD
Output of:
giAdjustLoc 46
locWy
Input of:
giAdjustLoc 43
giExtractSpectra 57
Output of:
giLocalSpectra 37
locWyMast
Input of:
giAdjustSL 32
locWyMaster
Input of:
giFitPsfProfile 50
locY
Input of:
giAdjustLoc 43
giExtractSpectra 57
giGetWaveSolution 65
Output of:
giLocalSpectra 37
locYMast
Input of:
giAdjustSL 31
locYMaster
Input of:
giFitPsfProfile 49

O

outFrame
Output of:
giArithmetics 12

P

phffImageMast
Input of:
giAdjustSL 32
pLocc
Output of:
giFitPsfProfile 52
plocWy
Output of:
giFitPsfProfile 50
plocY
Output of:
giFitPsfProfile 50
prepImage
Input of:
giAdjustSL 31
psfAmpl
Output of:
giFitPsfProfile 51
psfBack
Output of:
giFitPsfProfile 51
psfCent
Output of:
giFitPsfProfile 51
psfExpo
Input of:
giExtractSpectra 57
Output of:
giFitPsfProfile 50
psflwidth
Output of:
giRebinSpectra 76
psfWidt
Input of:
giExtractSpectra 58
Output of:
giFitPsfProfile 51

R

rawFrame
Input of:
giGetRemoveBias 15

S

sigmaImage
Output of:

giDetectCosmicM 26
slImage
Input of:
giExtractSpectra 58
Output of:
giAdjustSL 33

W

wlSp
Input of:
giCrossC 80
Output of:
giRebinSpectra 75
wlSpErr
Output of:
giRebinSpectra 76

Frame types/FITS Keywords Index

nlWbin

ACTMJD 80
BIASSIGMA 80
BINMETHOD 75
BINRANGE 75
BINWL0 ..... 75, 76, 80
BINWLMAX 75, 80
BINWLMIN 75, 80
BINWNS 75, 76
BINWNX 75, 76
BINWSTEP 75, 76, 80
BINWSTEPLOG 76
CDELT2 75
CREATOR 75
CRPIX2 75
CRVAL2 75
CTYPE2 75
EQUINOX 80
EXPTIME 80
GEOELEV 80
GEOLAT 80
GEOLON 80
GRATING 80
NX 80
NY 80
SLIT 80
WLEN0 80

nxExt

BIASSIGMA 43, 64
BITPIX 37, 45, 46, 59
BSCALE 37, 45, 46, 59
BZERO 37, 45, 46, 59
CONAD 43
CREATOR 37, 45, 46, 50, 59
DATAMAX 59
DATAMIN 59
ECLIPMFRAC 59
ECLIPNITER 59
ECLIPSIGMA 59
EMETHOD 59
EXTNS 31, 32, 59, 74
EXTNX 31, 32, 59
FILT 64
GIRFTYPE 49, 50, 59
GRATGRV 64, 74
GRATING 31, 32, 49, 50, 57, 58, 64, 65, 74
HEWIDH 59
HSIGMA 59
LAMP 64
LAMPID 64
LCLIPMFRAC 37, 45, 46
LCLIPNITER 37, 45, 46
LCLIPSIGMA 37, 45, 46
LEXTRAWID 37, 45, 46
LFULLLOC 37, 45, 46
LMETHOD 37, 45, 46
LNOISEMULT 37, 45, 46
LNOISETHRH 37, 45, 46
LNORMALIZE 37, 45, 46
LOCNS 37, 45, 46
LOCNX 37, 45, 46
LOCWDEG 37, 45, 46
LOCYDEG 37, 45, 46
LWFRAME 59
LYFRAME 59
NAXIS1 ..... 37, 45, 46
NAXIS2 ..... 37, 45, 46
NFIBRES 43
PIXSIZX 64
PIXSIZY 64
PMODEL 59
PSFCOEFFi 50, 51, 57, 58
PSFMODEL 50, 51, 57, 58
PSFPARAM 50, 51, 57, 58
PSFWDEG 50, 51, 57, 58
PSFXBINS 50, 51, 57, 58
PSFYDEG 50, 51, 57, 58
SLFRAME 59
SLIT 31, 32, 49, 50, 57, 58, 64, 65, 74
WLEN0 ..... 31, 32, 49, 50, 57, 58, 64, 65, 74

undef

EXPTIME 32
GRATING 32
NX 32
NY 32
SLIT 32
STARTX 32
STARTY 32
WLEN0 32

xy

BIASSIGMA....25, 36, 49, 56
BIASVALUE....15
BITPIX....22
BSCALE....22
BZERO....22
CONAD....15, 25, 36, 49, 56
CREATOR....22
DARKEXPECT....22
DARKMETHOD....22
DARKTHRESH....22
DARKVALUE....22, 25, 49, 56
DATAMAX....22
DATAMEAN....22
DATAMIN....22
DATAMODE....22
EXPTIME....25, 49, 56
GRATING....43
NFIBRES....36, 49
OVRSCX....15
OVRSCY....15
PRESCX....15
PRESCY....15
SLIT....43
WLEN0....43

xyPrep

ALERTVAL....26
AVGMETHOD....26
BCLIPMFRAC....16
BCLIPNITER....16
BCLIPSIGMA....16
BIASAREAS....16
BIASFILE....16
BIASMETHOD....16
BIASPLANE....16
BIASSIGMA....16
BIASVALUE....16
BITPIX....16, 26
BSCALE....16, 26
BZERO....16, 26
CCDID....15
CREATOR....16
CRHMMFRAC....26
CRHMNITER....26
CRHMSIGMA....26
CRMETHOD....26
DATAMAX....12, 16, 26
DATAMEAN....12, 16
DATAMED....12
DATAMIN....12, 16, 26
DATAMODE....16
DATARMS....12
EXPTIME....21, 31
GRATING....31
NAXIS1....11, 12, 21
NAXIS2....11, 12, 21
NX....31
NY....31
OVRSCX....25
OVRSCY....25
PRESCX....25
PRESCY....25
SCALING....26
SLIT....31
SLMODEL....33
SLPHFFCOR....33
SLPOLYDEG1....33
SLPOLYDEG2....33
STARTX....31
STARTY....31
WLEN0....31

__oOo__