R3D

A Package for Reducing IFS data

www.caha.es/sanchez/r3d

Centro Astronomico Hispano Aleman A.I.E.

MPG/CSIC

R3D, PMAS/PPAK reduction – v0.1

User Guide

Issue 1.2

10/12/2007

DRAFT

Authors S.F.Sánchez

Contact S.F.Sánchez (sanchez@caha.es)

1 Introduction

R3D (Sánchez 2006) is a software package for reducing fiber-based spectroscopic data, focused on the reducion of IFS (integral field spectroscopy) of IFUs (Integral Field Units) that uses this technology. This package is currenly used at CAHA to reduce PMAS data, and it has been tested with many other IFUs, including INTEGRAL, VIMOS, GMOS and ARGUS.

In this document we explain how to reduce this kind of data, focused on the reduction of PMAS data on both setups (Larr and PPAK). PMAS (Roth et al. 2005) is an IFU currently mounted at the 3.5m telescope of Calar Alto. The procedures described here can be easily adapted to reduce any other IFS data.

For reducing the data we will assume that you have also installed other reduction packages, like IRAF or IDL, for basic routines (like image combination or image arithmetic), and that you have installed the Euro3D visualization tool (E3D, Sánchez 2004) to visualize the IFS data. Through this document we have assumed the nomenclatures described in the E3D documentation regarding several concepts, like the concept of row-stacked spectra or the definition of a position table. We strongly recomend to read the definitions included in the E3D User Guide before to read the current document.

2 Installation

R3D is freely distributed at the CAHA webpage:

http://www.caha.es/sanchez/r3d

The instalation instructions are listed there.

There are two different distributions of the code, a Perl and a C version. The Perl version is slower, but it is able to reduce PMAS data in a reasonable time with current computers. It is the only complete version. Its installation is easy, but requires to install a set of Perl libraries.

To install the Perl version just download the corresponding compressed file from the webpage quoted above (`R3D.tgz`). Once you have download it, place it in the directory you want to install all the software (e.g., `/opt/3Dsoft`), and decompress the file (`tar zxvf R3D.tgz`). A new directorey named R3D will be created. Once created enter in the directory and run the script (`./test_r3d.pl`). If you got a sequence of error messages the most probable reason is that you require to install a set of Perl Libraries. These libraries are:

Astro-FITS-CFITSIO-1.01
Astro-FITS-Header-2.2
ExtUtils-F77-1.14
Math-Approx-0.200
Math-Derivative-0.01
Math-FFT-0.25
Math-Matrix-0.4
Math-Spline-0.01
Math-Stat-0.1
PDL-2.4.3
PGPLOT-2.18
Statistics-OLS-0.07

You can find all these libraries (or their last current versions) in the CPAN webpage: `http://www.cpan.org/`. Once you install all the routines, you have to change the location of the Perl library `my.pl` that is used by all the R3D routines following the instructions listed in the `INSTALL.txt` file included in the distribution.

In addition you requite to have installed `pgplot` and `cfitsio` to install all these libraries.

To install the C version you have two options, one is to download the source code from the R3D webpage and compile it. To do you require to have installed in your computer the following libraries:

LCL Lyon-C library, distributed by the Euro3D network at its webpage `http://www.aip.de/Euro3D/`
`pgplot` that you can download from `http://www.astro.caltech.edu/ tjp/pgplot/`
The Tcl/Tk development libraries

The compiling and installing instructions are included in the compressed file.

The second option (highly recomended) is that you download the pre-compiled version of the code available in the webpage. These pre-compiled versions consist of a compresed tar-file with the R3D routines currently translated to C (the most time consuming). You can perform 90% of the reduction using these routines. The installation in this case is very simple, you just donwload the tar-file, copy it to the directory where you want to install it, decompress it,

and then a new directory will be created (`R3D_c`). You just have to add it to your PATH to have the program installed.

It is our purpose to tranlate all the code to C, but we cannot guarantee in which time-scale that will happens.

For more instructions about the installation process, please consult the R3D webpage and the documents included in the distributed tar-files.

3 Data Reduction

The reduction procedure described in this document follows the techniques and sequence described in sánchez (2006). We highly recomend to read this article before to procede with the data reduction, since we will assume that the user knows the basic concepts of the fiber-based IFS reduction.

Most of the time, the raw data from a fiber-feed spectrographs consist in a collection of spectra, stored as a 2D frame, aligned along the so called dispersion axis. This axis can be the X or Y-axis or none of them, although it uses to be almost aligned with one of them. Figure 1 shows an example of IFS raw data, corresponding to PMAS (Roth et al. 2005) in the PPAK mode (Kelz et al. 2006), illustrating this distribution. Each spectrum is spread over a certain number of pixels along the perpendicular direction to the dispersion (cross-dispersion axis). The shape of the profile along the cross-dispersion direction is similar to an asymetrical gaussian or a kurtosis function. Therefore, there is a contamination from each spectrum to the adjacent ones. This is what is well-known as the cross-talk (see Fig. 1). As we quoted before, the spectra are not perfectly aligned along the dispersion axis, due to the configuration of the instrument, its setup, the optical distorsions, the instrument focus and the mechanical flexures (if any). Therefore, it is needed to find the location of the projection of each spectrum at each wavelength along the CCD in order to extract its corresponding flux. For doing so we use continuum illuminated exposures at any location where we are pointing with the telescope. Since the fleasures also affect the wavelength solution, it is also required an ARC exposure obtained at the same location of the science targets. Due to misalignements of the fibers with the pseudo-slit it is needed to correct for distorsion and find a wavelength solution of each individual spectrum. Finally, the differential transmission fiber-to-fibers should be corrected.

All these data reduction steps can be sumarized in the following sequential list:

(a) Pre-reduction.

(b) identification of the position of the spectra on the detector for all pixels along the dispersion axis,

(c) extraction of each individual spectrum,

(d) distortion correction of the extracted spectra and determination of the wavelength solution (i.e., dispersion correction)

(e) fiber-to-fiber transmission correction,

(f) flux-calibration,

(g) sky-subtraction,

(h) re-arranging of the spectra in the sky position: Mosaics and/or dither reconstruction.

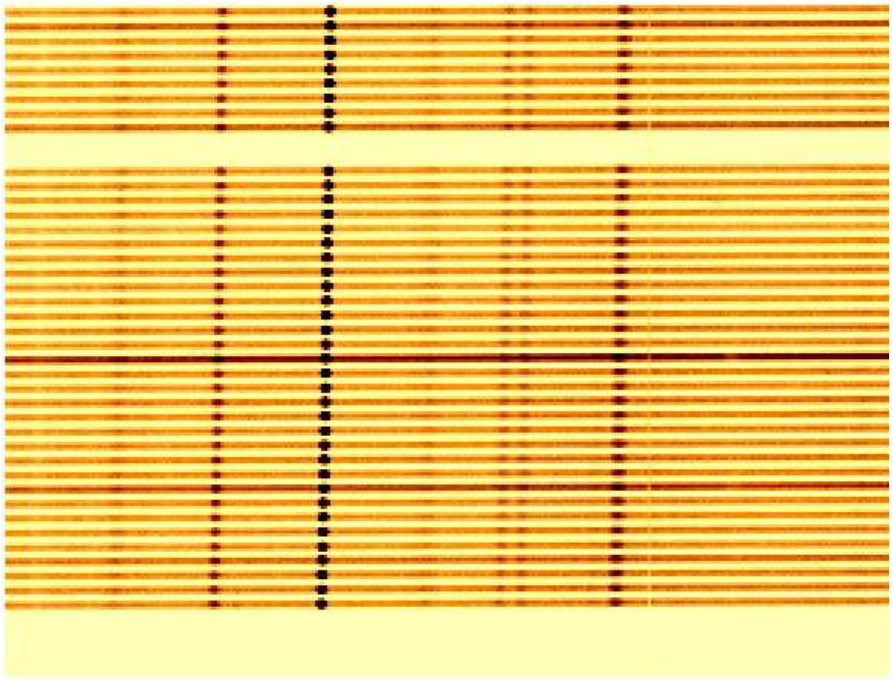

Figure 1: View of a section of IFS raw data obtained with PMAS in the PPAK mode, in inverse colors (i.e., more intense areas are shown in darker colors). Each dark line corresponds to the projection of a spectrum along the dispersion axis, which, in this case, corresponds to the X-axis. Spectra are separated by 7 pixels across the Y-axis, and projected in a few pixels, contaminating the adjacent spectra.

We briefly describe hereafter the different steps required to perform a reduction.

3.1 Pre-reduction

It consists in all the corrections applied to the CCD data that are shared with the reduction of any other CCD-based data. They do not require any special algorithm and for this reason we have not included them in R3D. You can use any external reduction tool to do them, like IRAF, IDL or MIDAS. It comprises:

- The creation of a master bias.
- The creation of a master CCD flat.
- Aplication of the master bias: Bias subtraction.
- Aplication of the master CCD flat. FlatFielding.
- Combination of different exposures on the same target.
- Cosmic ray rejection (it not performed by the previous step).

After these steps are performed the reduction can be performed using R3D.

3.2 Identification of the position of the spectra on the detector

To do that, you need a continuum exposure (e.g., a domeflat), that, in the case of instruments that suffer from fleaxures (i.e., displacement of the spectra in the CCD at different positions of the telescope), should be taken with the telescope pointing to the same location of your science exposure. The location of the spectra is found to be at a given column in the CCD by comparing the intensity at each row along the column (or a coadded set of rows around it) with those of n adjacent pixels, checking for those pixels that they verify the maximum criteria, using `peak_find`.

This program requires as an input the continuum raw data frame, a flag indicating the principal direction of the dispersion axis (0 for X, 1 for Y), the width around the central column where to coadd the intensities (for low-level signal-to-noise frames), a flag indicating if the results are to be plotted (1) or not (0), the number of subframes to plot in the screen, the number of adjacents pixels to check for the maximum criterium, the minimum distance between adjacent spectra (projected in the cross-dispersion axis), the fraction of the intensity peak to use as a threshold for detecting intensity peaks (i.e., spectra), and the name of an ASCII output file containing the location along the cross-dispersion axis of the detected peaks.

syntax: `peak_find RAW.fits Spectral_axis[0/1] Coadd_width plot nplot nsearch DMIN IMIN(% of the MAX) OUTFILE [colum_search]`

[NOTE: The meaning of each individual parameter is explained in the last section of this document]

If you execute the program in the iterative mode it will show you a plot of the intensities along the cross-dispersion axis corresponding to the central column (or the column you selected with

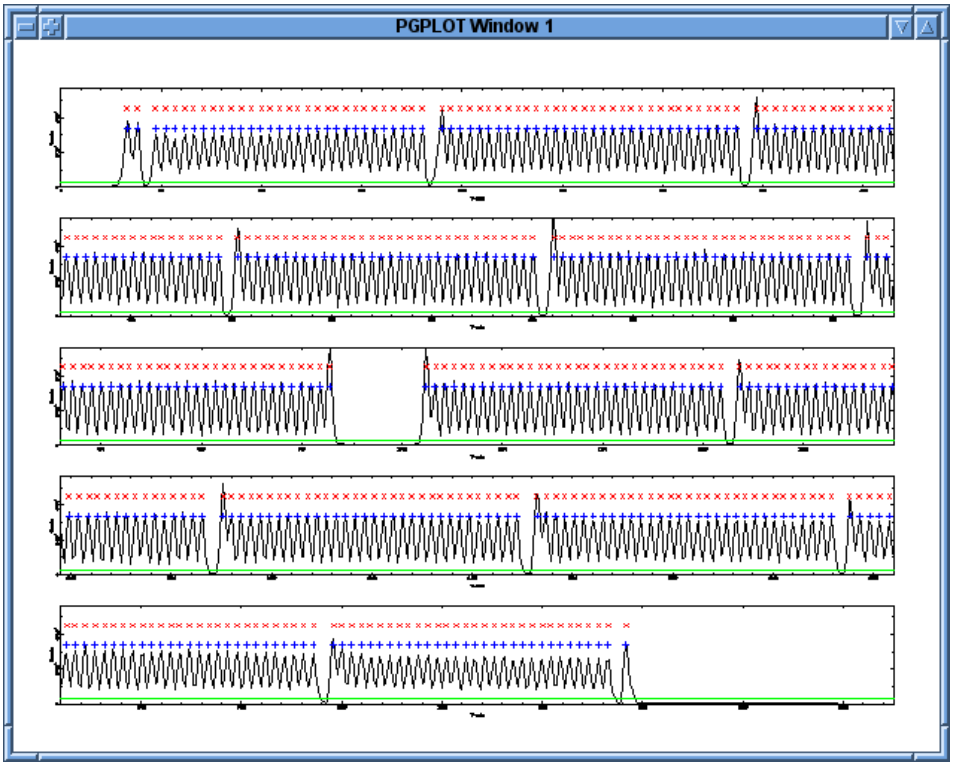

Figure 2: Output of the peak_find routine over PMAS data in the PPAK mode

the comand line options). It will overplot a green horizontal line indicating the threshold used (IMIN parameters), and a set of dots. The red dots marks the location of the pixels where the program has found a peak, using the parameters defined by the command line. The blue dots marks the location of the centroid of these peaks, using an hyperbolic fitting using the peak pixel and the two adjacent ones. All these values are stored in the output file.

For PMAS the tipical parameters used in this routine are:

```
peak_find tjunk.fits 0 1 1 5 1 3 0.05 tjunk.peaks
```

In the case of the LARR mode it should find 256 individual spectra, and in the case of PPAK 382. The distribution of the spectra are always the same. In the case of the LARR mode you should see 16 groups of 16 individual spectra, separated by a hole. In the case of PPAK the distribution is more complex. It starts with two calibration spectra (of 15 in total), the a hole, a group of science+sky spectra, the a hole, one calibration spectra, another group of science+sky and so on as illustrated in Figure 2.

Then, the tracing of the peak intensities along the dispersion axis is done by looking for maxima around each original location within a given window, using `trace_peaks_recursive`. This program requires as an input the continuum raw data frame, a flag indicating the dispersion direction (0 means along X, 1 along Y), the file containing the location of the intensity peaks (i.e., the output of the previous program), the width of the window where to coadd around each column the intensities along the dispersion axis, the width (in pixels) where to look for a new centroid of the peak along the cross-dispersion axis, a flag indicating if the results are to be plotted (1) or not (0), the number of subframes to plot in the screen, the number of adjacents pixels to check for the maximum criterium, and the output file, a 2D FITS file where the X-axis corresponds to the original dispersion axis and the Y-axis indicates the number of traced spectrum. The value stored in each pixel is the location of the peak of the centroid in the original frame of the corresponding spectrum at the corresponding spectral pixel. This file is name the *trace*.

syntax: `trace_peaks_recursive RAW.fits Spectral_axis[0/1] PEAKS_FILE coadd_width search_width plot nplot nsearch TRACE.fits [y_shift] 2nd_search_width]`

The typical values for PMAS are:

```
trace_peaks_recursive tjunk.fits 0 tjunk.peaks 60 4 1 5 1 tjunk.trc.fits 0 2
```

It you execute it in the interactive mode, the program will show you an initial plot showing a similar cut as shown by `peak_find`, with the blue and red dots indicating the same values. It also plot in yellow the initial location of the peaks, as read from the peaks-file. Once you allow the program to continue plotting it will show a similar plot for each column, starting from the central one and finish with the edges of the CCD frame, the program will plot you the *trace* of each individual spectrum (in blue). That is, the location of the peak intensity corresponding to each spectrum in the CCD. It is important that this *trace* is continuous. If you see jumps at certain wavelengths most probable there was an error and the program has confused two adjacent spectra.

3.2.1 Extraction of individual spectra

After tracing the location of the spectra on the CCD, the next reduction step is to extract, for each spectrum, the flux corresponding to each spectral pixel along the dispersion axis. The simpliest method to perform this extraction is to coadd the flux within a certain aperture around the 'trace' of the spectra in the raw data, and store it in a 2D image (RSS or Row-stacked-spectra). The X-axis of the resulting image corresponds to the original dispersion axis, while the Y-axis corresponds to the ordering of the spectra along the pseudo-slit. This is the so-called row-stacked spectra representation or RSS [**?**]. The extraction is performed using `extract_aper`.

This program requires as an input a raw data frame, a flag indicating the direction of the dispersion axis (0 for X, 1 for Y), the trace frame, the aperture width for the extraction and the name of the output file.

syntax: `extract_aper RAW.fits Spectral_axis[0/1] TRACE.fits Aperture OUTPUT.fits [SHIFT]`

The typical parameters for PMAS are:

```
extract_aper junk.fits 0 tjunk.trc.fits 5 junk.ms.fits 0
```

in the case of the LARR mode you can increase the width of the co-adding pixels to 7, due to the largest distance between adjacent spectra in this mode.

Once you have extracted the data you can visalize them using E3D (tk_e3d.tcl command), by importing the RSS (in the file menu), and selecting the corresponding position table: pmas_pt.txt for the LARR mode or ppak_382_arc.txt for the PPAK mode. At this level, in the case of PPAK, the calibration fibers are placed in an artificial location in the sky, since they are not poiting to any sky location.

To continue with the reduction it is needed that you first reduce at this level the skyflat exposures (or the corresponding dome-flats) that you are going to use to determine the fiber-flat (see Sánchez 2006). For each frame, you need to extract both the science frame (or skyflat frame), and the corresponding ARC frame. You will need the ARC frame to perform the next steps of the reduction.

3.2.2 Cross-talk

By contruction the projected spectrum of each fiber in the CCD contaminates the adjacent one in an amount that depends on the distance between adjacent spectra and the width of the projected distribution along the cross-dispersion. This effect it is known as cross-talk, and it produce a cross-correlation of the signal that can blur signal of faint objects or create artificial structures. In many instruments, like PMAS in the Larr mode it is not important (less than 1%), while in others, like VIMOS, may be critical.

Different methods has been implemented in R3D to reduce the effect of the cross-talk:

- `extract_aper_CT` This routine perform a non analitical correction of the cross-talk by performing an iterative subtraction of the guess contribution of the contamination by adjacent fibers to the considered one, by assuming that the have approximatelly a

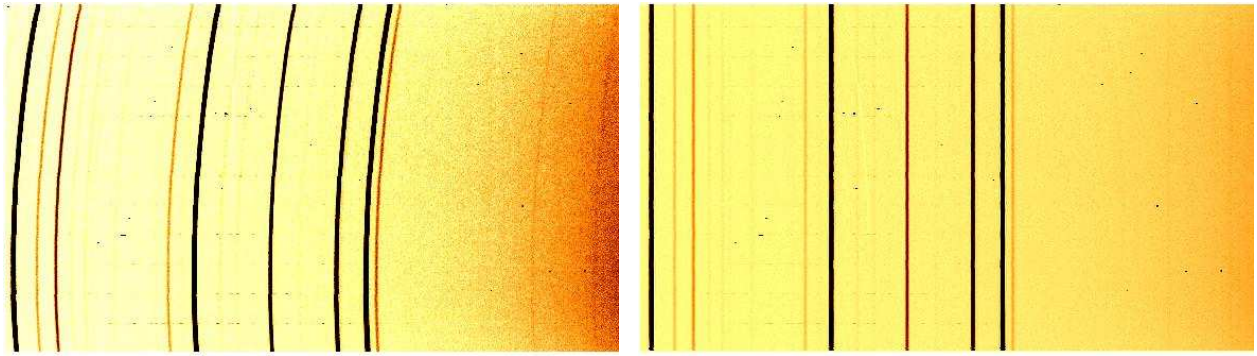

Figure 3: Example of extracted spectra from an arc exposure obtained with PMAS in the PPAK mode, before correcting for the distorsion in the dispersion along the cross-dispersion axis (Left panel) and after (right panel). The X-axis corresponds to the dispersion axis

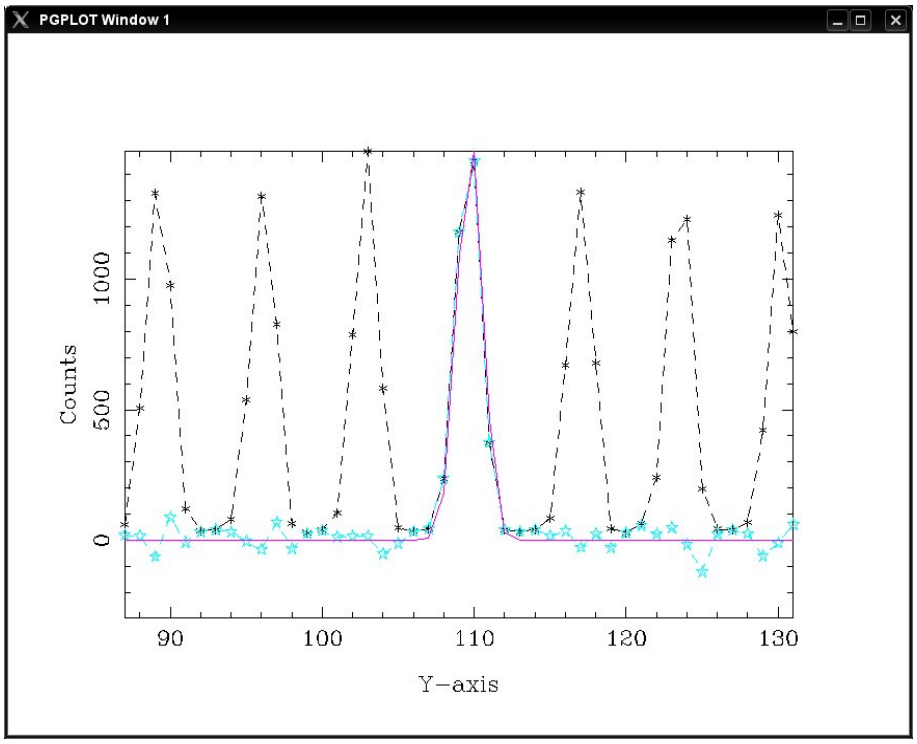

Figure 4: Example of the extraction using `extract_gauss_simple` for correcting the effects of cross-talk. The black line shows the original data, while the blue one shows the decontaminated one. The final fit is shown in magenta.

gaussian distribution along the cross-dispersion axis. It is very slow.

- `extract_gauss_multi` This routine perform a multigaussian fitting to the projected distribution of the each spectra in the CCD for each pixel in the spectral axis. The center and the gauss is defined by the tracing, while the width has to be provided by the user. Although it does produce good results, it is extremelly slow.

- `ectract_gauss_simple` This routine perform a single gaussian fit to extract the considered spectrum. In a 1st approximation it fits individually each spectrum, storing the results, which are used to decontaminate each spectrum from the contamiantion from the adjacent ones by assuming a gaussian shape. The decontaminated cut across the cross-dispersion axis is then fitted again to a single gaussian function. The result is stored again and used to decontaminate the cut, that it is finally fitted again by a new single gaussian function.

  The reason for this three iteration process is that the 1st fit, without any decontamination produces an overestimation of the flux. Using it to decontaminate the spectrum allows to a better estimation of the flux, but it is still overestimated. The final iteration produce reasults as good as the two previous methods, but with a much faster speed. We strongly recomend to use this method, that we have tested with PPAK and VIMOS data.

Figure 4 shows an example of the decontamination of the cross-talk by using the last described procedure.

3.2.3 Distorsion correction

Most of the spectrographs do not disperse the light homogeneously along the cross-dispersion axis. The dispersion is distorted, being larger in the edges of the slit than in the center. This distorsion is sometimes called the C distorsion, due to its shape on the CCD. In the case of fiber-feed spectrographs, additional distorsions are introduced due to the way the fibers are placed in the pseudoslit. Once the spectra extracted, these distorsions are self-evident. Figure 3, left panel, shows an example of extracted spectra, corresponding to an arc exposure obtained with PMAS in the PPAK mode. The C distorsion and the discontinuities in the dispersion solution along the cross-dispersion may be clearly seen. These discontinuities and the slight differences in the dispersion from fiber-to-fiber, do not allow to perform a 2D modelling of the distorsion map with an analytical function, nor at the level of the extracted spectra, neither at the level of the raw data. The distorsions have to be corrected fiber-to-fiber before finding a common wavelength solution.

`R3D` performs this correction by a two steps procedure, using arc calibration lamp exposures (like the one in Fig.3), once extracted its corresponding RSS as indicated in the previous section. First, the peak intensity of a single emission line is traced along the cross-dispersion axis (in the RSS ARC frame), and shifted to a common reference, by a linear shift. This is done by the program `dist_cor`. This program require as input the aperture extracted file, the output corrected file, the output first order distorsion file, a flag indicating if to smooth (1) or not (0) the solution along the cross-dispersion axis, the reference pixel in the dispersion axis to look for the emission line peak, the width of the window in pixels to look for this peak, the number of adjacents pixels to check for the maximum criterium and a flag indicating if to plot (1) or not (0) the results.

syntax: `dist_cor EXTRACTED.fits CORRECTED.fits DISTORSION_CORRECTION.txt SMOOTH[0/1] start_index delta_index nsearch plot [n_fib] [nsigma] [center_index]`

The typical values for PMAS are:

```
dist_cor arc.ms.fits arc.dc0.fits arc.dist.txt 0 550 50 2 0 2
```

although the aproximate location of the emission line (550 pix in the X-axis), and the selected width to look for peaks (50 pixels) strongly depends on the observing mode (LARR or PPAK), grating, and calibration lamp used. These values have to adjusted by inspecting the RSS ARC frame, just using a FITS visualization tool (like ds9) or any similar one. The file arc.dc0.fits can also be inspected after correction to check for the quality of the correction.

If you run the program in the interactive mode (plot=1), it will show you first a plot of the spectrum at the considered range (500-600 pixels, with the previous listed values), marking the location of the peak intensity of the selected emission line. In the next step the location of this peak intensity is determined for each spectrum. The plots will show you the corresponding spectrum compared to the original one. After running through all the spectra, the program shows you a plot of the determined shift. Finally, it will show you the applied offset (in blue), the interpolated final data (in green), and the original one (in red).

Once the process is finished, the intensity peak of a set of selected emission lines is traced, and a polynomial distorsion correction is determined to recenter all the lines to a common reference. This is done using the program `mdist_cor_sp`. This program requires as input the output of the first order correction, an aperture to look for emission lines, a nσ threshold over the average intensity to look for emission line peaks, the order of the polynomial function to fit

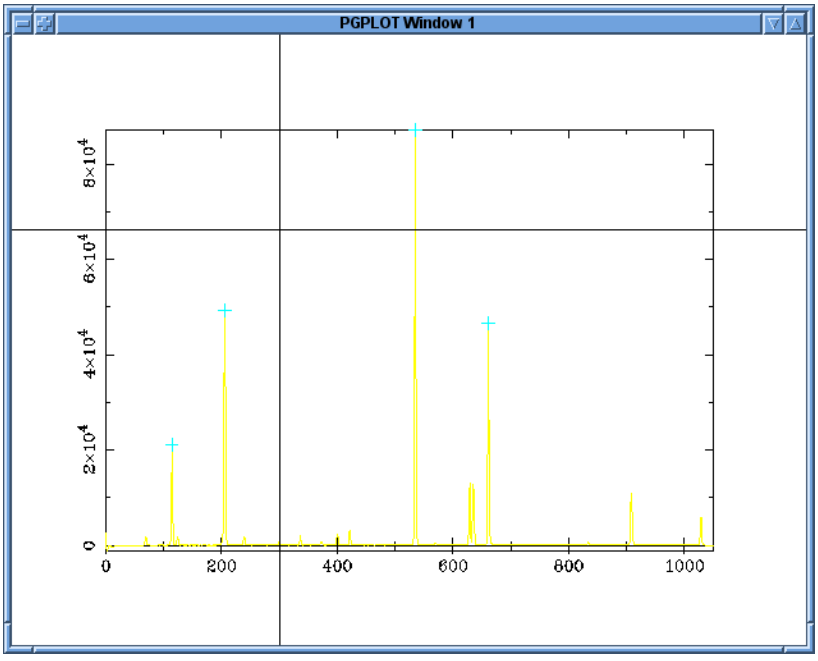

Figure 5: Example of the interactive screen of the mdist_cor_sp.pl routine

the distorsion correction (spectrum to spectrum), the output file with the distorsion corrected frame, the output file containing the 2D map of the corrections, the range of pixels in the dispersion direction to consider for the analysis, and a flag indicating if results will be plotted (1) or not (0).

syntax: `mdist_cor_sp EXTRACTED.fits APERTURE NSIGMA NPOLY OUTPUT.fits DISTORSION.fits NX_min NX_max plot [NSIGMA2] [BOX] [NY_CEN]`

Typical values for PMAS are:

```
mdist_cor_sp arc.dc0.fits 5 30 4 arc.dc1.fits arc.dist.fits 22 10100 0 0 0 2
```

PMAS is a well behaved instrument in terms of the distorsion, and low order polynomial functions correct them very well. In this example we use a 4-order polynomial function.

This program need to be run interactively, if you do not know the location of the emission lines (Identification file, named .id in the adopted nomenclature). When you run the program it will show you, in a PGPLOT /XWINDOW, a plot of the 1st (or the selected) spectrum in the frame corresponding to the 1st order corrected ARC RSS image, as shown in Figure 5. In this screen is possible to mark emission lines (although you do not need to identify them at this level). The different interactive options that you can apply over the screen are:

a => Unzoom
x => X-Zoom
y => Y-Zoom
m => Mark a peak
d => delete a peak
r => Move right
l => Move left
R => Load a file
W => Save a file
F => Load a file and recenter the peaks
q => Quit

After marking a set of emission lines (more than the selected order of the polinomial function), it is needed to store the identification, if you want to use them later for any other data. This is down with the "W" option, and indicating the name of the file. We recomend this file to be named *.dist.id, in order to identify its nature later on. Once marked enough lines, you can click "q", and the program will find the distorsion correction for each spectrum, using the marked lines.

If you already have an identification file, you can indicate the program to use it, adding it to the corresponding command-line option, and it will automatically recenter the emision lines.

Once you have the 1st and 2nd order distorsion corrections (arc.dist.txt and arc.dist.fits in the previous examples), obtained using the RSS ARC exposures using the procedures described before, you have to applied them to your science data. For doing so you require as input the RSS of your science data, extracted using the procedures described in the previous section (these files are normally named *.ms.fits). To apply this correction we use the program mdist_cor_external.pl:

syntax: `mdist_cor_external.pl EXTRACTED.fits DISTORTION.txt DISTORSION.fits OUTPUT.fits plot [offset]`

A tipical use (for PMAS and any other instrument) would be:

```
mdist_cor_external junk.ms.fits arc.dist.txt arc.dist.fits junk.dc1.fits 0
```

Although you still do not have a wavelength solution, you can inspect the quality of your data using E3D. At this level all the spectra have a common wavelength solution (that you still do not know), and therefore you can use the images to create maps.

3.2.4 Wavelength solution

To derive the wavelength coordinate system, the emission lines of an arc exposure are first identified, using an interactive routine. The distorsion-corrected spectra of the arc (the output of the routines described in the previous section) are then transformed to a linear wavelength coordinate system by a 1D spline interpolation, assuming a polynomial transformation between both coordinate systems. The required transformation is stored in an ASCII file to be applied later on the science data. This procedure is done using the `disp_cor.pl` script.

This program requires as an input the output of the distorsion correction analysis, the starting wavelength and the wavelength step wanted for the final spectral pixels, the aperture to look for emission lines, the row number of the central spectrum over which perform the analysis, width of pixels in the cross-dispersion axis for coadding the spectra (to increase the signal-to-noise ratio), the order of the polynomial function to fit the wavelength solution, the output file containing the spectra transformed to a linear wavelength coordinate system, the output file containing the dispersion or wavelength solution applied and a flag indicating if results will be plotted (1) or not (0).

syntax: `disp_cor.pl EXTRACTED.fits CRVAL CDELT APERTURE NY_SPECTRA WITDH NPOLY OUTPUT.fits DISPERSION.txt plot [NMAX] [file.id]`

You need to run the program interactively if you do not have a file with the identification lines. Once you run the program, it will open an interactive XWINDOWS with the following options:

a => Unzoom
x => X-Zoom
y => Y-Zoom
m => Mark a peak
d => delete a peak
r => Move right
l => Move left
R => Load a file
W => Save a file
F => Load a file and recenter the peaks
q => Quit

The mechanism is similar to the one of mdist_cor_sp, but in this case when you mark an emission line it will open a prompt in the xterm requesting you the corresponding wavelength. Once you have identified enough emission lines, you have to store the identification file (named *.disp.id), by clicking “W” in the graphical terminal. You can restore any previous saved identification file o retore and recenter by clicking “R” or “F” in the graphical terminal. You can use this option to use a previous identification over a new arc exposure. You can also use a previous identification adding the name of the identification file in the corresponding command line. To test the quality of your wavelength solution you can just click “q” on

the graphical window, and check the *rms* of the fitting. Whenever you are happy with the wavelength solution you enter "Q" in the text terminal and apply the wavelength solution over the spectra.

The typical values of the parameters to use this routine with PMAS data are:

```
disp_cor arc.dc1.fits 3700 3.2 3 100 20 4 arc.disp_cor.fits arc.disp.txt 1 1075
```

Of course, most of these parameters depends strongly on the grating and the grating angle of the data (do not use them as default in all the cases!!!).

Once you have a wavelegnth solution you have to apply over your science data. For doing so, you need to use the `disp_cor_external` task. This task uses as imput the distorsion corrected RSS frame, i.e., the output of the `mdist_cor_external` task (EXTRACTED.fits), the starting wavelength and the wavelength step per pixel of the adopted wavelength solution (the same values used in the `disp_cor` task), and the dispersion solution found with the previous task:

syntax: `disp_cor_external EXTRACTED.fits CRVAL CDELT OUTPUT.fits DISPERSION.txt plot`

An example of its use of the low resolution mode of PMAS, in both configurations, using the V300 at a grating angle of -74.9 is:

```
disp_cor_external junk.dc1.fits 3700 3.2 junk.disp_cor.fits arc.disp.txt 0
```

This task will produce a distorsion corrected, wavelength calibrated, RSS frame of the science data.

4 Fiber-to-fiber transmission correction: FiberFlat

The light-path from the focal plane of the telescope to the detector is different spectrum to spectrum. They suffer from differencial positioning with respect the the axis of the optical system the entrance pupil of the instrument, the entrance cone in each of the fibers is different due to small differences in the positioning of each fiber with respect to the focal plane of the telescope, each fiber, although done with the same material, have slightly different transmissions one-to-other, with a clear wavelenght dependecy, the location of the final edge of the fiber is different one each other along the slitview of the spectrograph, with slight misalignments, and finally, each spectrum suffer by different distorsions due to the differential position with respect to the optical axis of the spectrograph. In general all these effects produces a differential transmission optical element to optical element that are translated in what is normally named a differential transmission fiber-to-fiber. Figure 6 shows the differential transmission fiber-to-fiber for PMAS in the PPAK mode, illustrating the problem.

To correct for this effect it is required to obtain an exposure of a continous illuminated source, like the twilight sky (although a domeflat is in many times good enough). This exposure has to be reduced, tracing the location of each spectrum, extracting, correcting the distorsions, and applying a wavelength solution. Once performed these steps of the reduction, the output file can be used to determine the differential transmission fiber-to-fiber, what it normally called the *FiberFlat*.

For doing so it is needed to use the `fiber_flat.pl` routine:

syntax: `fiber_flat.pl SKYFLAT.fits FIBERFLAT.fits`

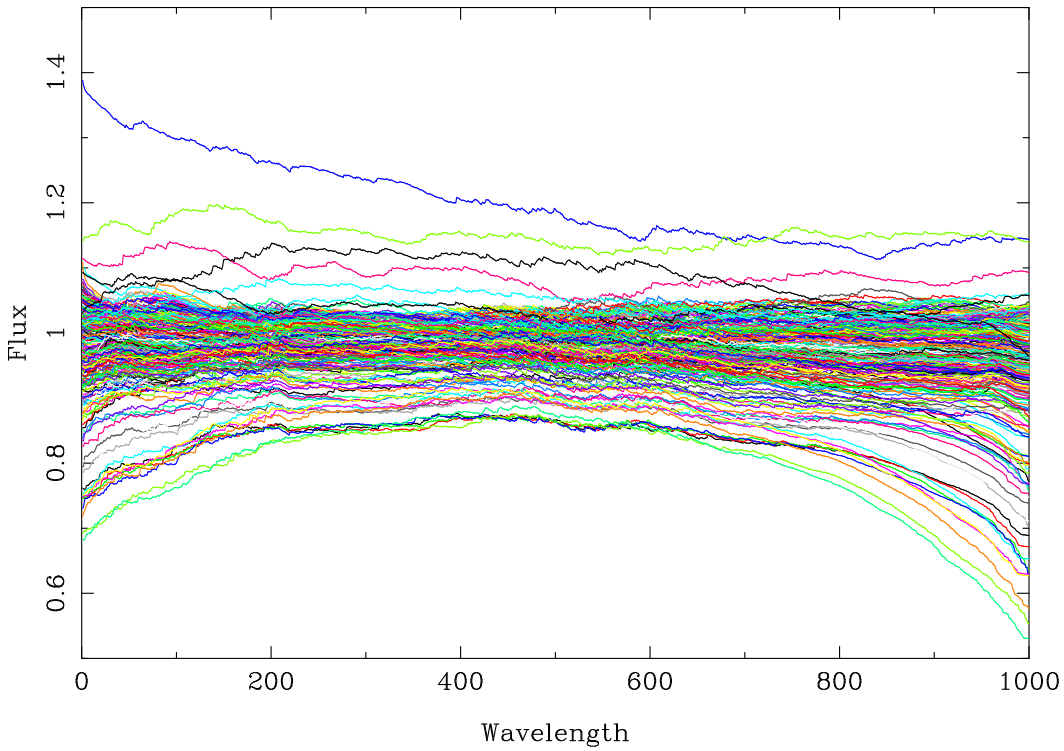

Figure 6: Relative transmission of each fiber along the wavelength corresponding to PMAS in the PPAK mode. Each fiber transmission is plotted in a different color to illustrate the differences in the transmission fiber-to-fiber at each particular wavelength.

this routine uses the previously indicated frame, obtain a median spectrum using all the spectra in the frame and divide each spectrum by this median one, obtaining a file that can be used to correct from the differential transmission fiber-to-fiber by dividing the science frames by this file. With this procedure the transmission of all the fibers is normalized to the median transmission. If this procedure is applied well, it is latter possible to know the transmission of each any fiber by measuring the transmission through one or a reduce number of them (by observing a calibration star).

In the case of PMAS in the PPAK mode R3D also provides with a different tool:

syntax: `fiber_flat_ppak.pl SKYFLAT.fits FIBERFLAT.fits`

This tool performs the same calculation as the previous one, but in the calculation the calibration fibers (which input is different than the one of the remaining fibers) are masked. The created fiber-flat is only useful to correct for the differential transmission of the science and sky fibers, that in any case are the only useful ones at the end of the reduction process.

5 Flux-calibration

Once you have performed the previous steps of the data reduction you have a distorsion corrected, wavelength calibrated and fiber-to-fiber differential transmission corrected RSS of your data. The final step of the reduction, at the level of the CCD frames, is to determine the transformation between counts measured and real flux per second, that is normally called flux calibration. For doing so it is required to have observed an standard spectrophotometric calibration star (or a set of them) during the night. You should apply all the previous reduction steps to the calibration star frame, in order to use it to measure the ratio between the counts per second and the flux per second. This ratio will be latter applied to the science data to "flux-calibrate" them.

The flux calibration is done in R3D using the routine `flux_ratio.pl`:

syntax: `flux_ratio.pl UNCALIBRATED_SPEC.txt CALIBRATED_SPEC.txt ratio.txt FACTOR MEDIAN_BOX plot SMOOTH_BOX min_wave max_wave`

This routine requires as inputs a 1D uncalibrated spectrum of the standard spectrosphotometric star observed during the night, a calibrated spectrum in the standard format used in different observatories (ASCII file, 1st colum wavelength, 2nd column flux), and a FACTOR to transform the values in the input spectrum in counts per second (normally 1/EXPTIME). The uncalibrated and calibrated spectra can be smoothed to match their resolutions using the MEDIAN_BOX and SMOOTH_BOX parameters. The result is stored in a 1D spectrum file, named "ratio.txt", that contains the required transformation.

Assuming that you have a reduced frame of an standard star observed during the night it is needed to extract 1D spectrum. For doing so we recomend to use E3D. A RSS can be uplodaded into this tool by importing it. It will require the position table, that in the case of PMAS for both the lensarray (pmas_pt.txt) and PPAK (ppak_382_arc.txt or ppak_pt_arc.txt) are included in the E3D distribution. Once you import the RSS, you have to subtract the sky. For doing so you open the spaxels inspector is select areas clean of source. Then in the spectral inspector you store the median sky spectrum obtained with this method (in the Spectra menu), and you apply the sky subtraction option in the same menu. After that you clean the selected fibers using the clean option in the spaxels inspector, and select the fibers with flux from the calibration star. Once you plot the corresponding 1D spectrum in the spectral inspector you can save it as an ASCII file (using the corresponding option in the File

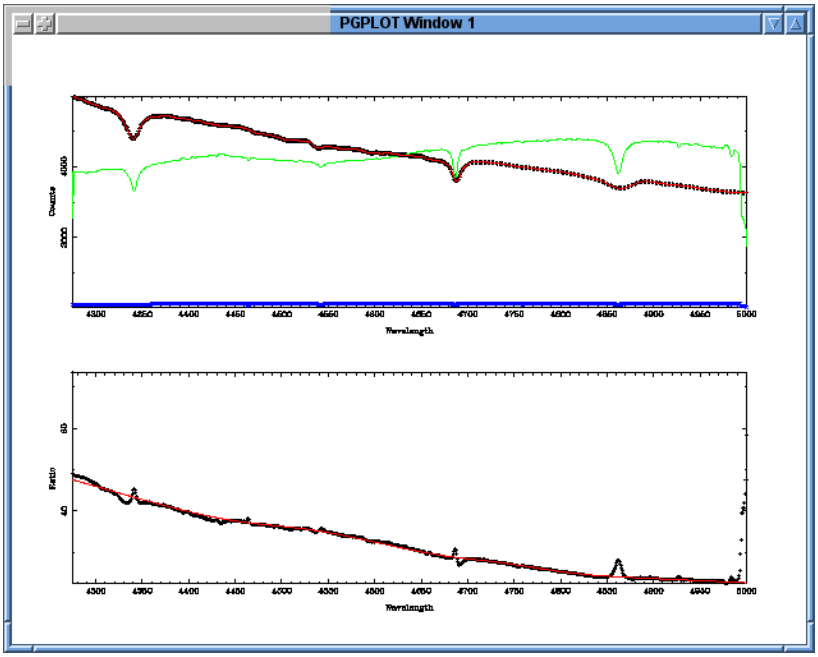

Figure 7: Graphical output of the flux_ratio.pl task. In the top panel is plotted in black the calibrated spectrum of the spectrophotometric star, and in red its interpolation to the resolution of the science data. In blue is is shown the uncalibrated 1D spectrum, and in green an scaled version of this one. The bottom-panel shows the ratio between the two spectra, in black, and in red the smoothed version that it is finally stored in the output file.

menu). It is required that you keep in mind the number of selected fibers. The reason for that is because E3D average the spectra of the selected fibers when it plots a single spectrum, and therefore, it is needed to multiply the fluxes in the stored spectrum by the number of selected fibers to recover the real counts of the calibration star. This can be done by selecting the FACTOR to be N.FIB/EXPTIME, in the `flux_ratio.pl` task.

Once you have extracted the 1D spectrum of the calibration star, it is easy to apply the corresponding task to derive the flux calibration. It is recomended to visualize the output since this ratio should be an smooth function. In some cases the discrepancies in the resolutions of the uncalibrated and calibrated spectra can create wirlets that it is needed to be smoothed.

To apply the flux calibration over the science data, it is required to multiply them by the ratio determined with the `flux_ratio.pl` task. For doing so it is convinient to create a RSS file of the same size of the science data with each spectrum being the 1D spectrum of the ratios to apply. After that you can just use the `imarith.pl` task to apply the calibration over the science data (or the similar one in IRAF or any other reduction tool). These data have to be divided by the exposure time, in the final step of the flux calibration process.

To create the 2D RSS file it is required to use the task `spec2img.pl`:

syntax `spec2img.pl spec.txt image.fits ny [crval] [cdelt]`

This task requires as input a 1D spectrum (in the ASCII file format), and creates the corresponding 2D spectrum with the same spectrum in each row, and a `ny` number of rows.

An example of the the flux calibration in PMAS is:

```
flux_ratio.pl spec_BD28_1.txt fbd28d4211.dat ratio_1.txt 0.0033 3 1 50 0 10000
spec2img.pl ratio.txt ratio.fits 382 3700 3.2
imarith junk.disp_cor.fits * ratio.fits junk.FC.fits
imarith junk.FC.fits / 600 junk.FC.fits
```

Figure 7 shows an example of the graphical output of the `flux_ratio.pl` task.

6 Sky Subtraction

The procedure to subtract the sky emission is different for each instrument and setup and for each kind of observation. If the observed target does not fill all the field of view of the IFU, the best option is to use E3D, following the instructions given in the previous section for the sky subtraction of the spectrophotometric standard star frames. This techique can be used both for the Lensarray and PPAK. If your target fill all the field of view of the IFU it is required to take an independent observation of the sky during the night, as near as possible (both in time and location) to the science target. In this case the sky exposure is reduced following the same procedure described before, and then subtracted to the science frames (once normalized by the exposure time).

In the case of PPAK there are a set of 36 fibers located at ∼90" of the center of the science bundle, grouped in 6 small bundles. The purpose of these fibers is to sample the sky, what they do for targets of the order of ∼1 arcmin in size. In this case is possible to select the spectra of these fibers by using the procedure `split_ppak.pl` :

syntax: `split_ppak.pl INPUT OBJECT CAL SKY PLOT`

where the input is the RSS frame, and the output are three RSS, one correponding to the

central hexagonal bundle (OBJECT), which comprises 331 fibers, another corresponding to the 15 calibration fibers, and a last one corresponding to the 36 sky fibers. As a 1st approach the median spectrum of these 36 sky spectra should produce a reasonable representation of the sky spectrum. A simple method to create a RSS frame with this spectrum is to apply the routine `median_spec.pl`:

syntax: `median_spec.pl RSS.fits MEDIAN_SPEC.fits [NY]`

which determines the median spectrum of the input spectra (RSS.fits), and store it in a new RSS of NY rows. In the case of PPAK the use should be:

`median_spec.pl sky.fits median_sky.fits 331`

where sky.fits is the output of the previous routine, and median_sky.fits is the derived sky to be subtracted to the central bundle RSS frame.

However, as described in sánchez (2006) it is possible to create a better representation of the sky that takes into account the differences in the distorsion and dispersion of the spectra at different locations of the CCD. This is done in the case of PPAK using the routine `create_sky_ppak.pl`:

syntax: `create_sky_ppak.pl INPUT OBJECT SKY npoly PLOT`

this routine creates a better representation of the sky frame for the central hexagon by fitting polynomial functions of the sky intensity at each wavelength taking into account the location of the spectra across the CCD. The input frame in this routine is the original RSS frame, containing 382 individual spectra. It requires an order for the polynomial function to fit, and produce two frames, one corresponding to the central bundle of 331 spectra, prior to sky subtraction, and another frame of 331 spectra comprising the best reconstruction of the sky for each spectra in the bundle. The sky is then corrected by subtracting the latter to the former frame.

If the science target is distributed inhomeneously through the field of view of the IFU, but the exposure is statistically dominated by the sky, it is possible to derive a good representation of the sky using the routine `create_sky_clip.pl` :

syntax: create_sky.pl INPUT SKY WIDTH_BOX NSIGMA PLOT

This routine creates an sky RSS spectrum of the same number of rows of the input RSS spectrum by obtaining the median between a certain number of adjacent spectra (defined by WIDTH_BOX), clipping those ones with a flux over a certain threshold of the standard deviation (NSIGMA, where the threshold is NSIGMA*STDDEV).

Which approach is the best depends on the nature of your science observations.

7 Locating the spectra in the sky: Mosaics and dithers

The location of the spectra in the sky is given by a certain position table that relates each spectrum with a certain fiber, that is located at a fix position with respect to each other in the IFU configuration. To know at which location corresponds each spectrum the best option is to use E3D, following the procedures described above, when we described the sky subtraction for the calibration stars.

In the case of IFUs coupled with rectangular lensarrays it is possible to rearrange the RSS spectra in a datacube, since their position tables are just pixels of a rectangular grid. This is the case of PMAS in the lensarray mode. In these cases it is possible to use the task `rss2cube_pt.pl`

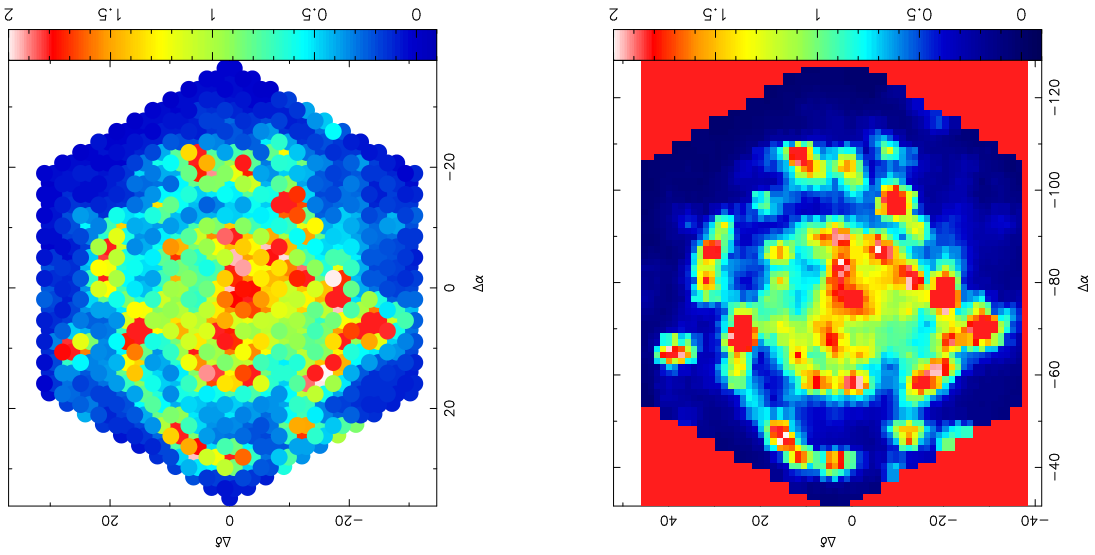

Figure 8: Left panel: Result of the `Mosaic_rss.pl` routine applied over a 3 dither exposure with PPAK on a nearby espiral galaxy. Right panel: Result of the `rss2grid_new.pl` routine over the previous data shown in the left panel.

syntax: `rss2cube_pt.pl input_RSS.fits pos_table.txt output_cube.fits`

this routine requires as input the reduced RSS spectra and the associated position table and produced as output a FITS cube where the spectra are ordered according to the locations defined in the position table.

In the case of PMAS in the PPAK mode it is not possible to do such simple operation. However it is possible to create a cube of data for the central hexagonal IFU (the OBJECT in the previous task descriptions), by interpolating at each wavelenght the intensities at each spatial location creating an artificial squared grid pattern. This is not a part of R3D, but you can use tasks like E3D for doing so. This program includes two different methods to interpolate and create final datacube. One is just to load the RSS in the GUI, following the procedures described before, and latter export it as a datacube. The other is to use the task `rss2cube.tcl` distributed in the E3D package. Both methods are well described in the E3D User Guide.

In some situtations it is interesting to do mosaics of a certain large target, that clearly exceed the size of the field-of-view of the IFU, or to perform dithering exposures to increase the sampling and/or fill the holes between adjacent fibers (like in the case of PPAK). Both cases are strongly experimental and they depend on the science case and the nature of the targets and/or the observations. However, we have included some tasks in R3D that can be used (or modified) to handle this situation.

The most simple method to joint different exposures taken at different locations of the same target (mosaics or dithers), is just to create single RSS and position tables which includes the individual exposures, by using the task `Mosaic_rss.pl` :

syntax: `Mosaic_rss.pl CONFIG.txt RSS.FITS POSITION_TABLE.txt [NO_FITS]`

This task uses as input a configuration file that consist of an ASCII table where in each row it is listed the name of the individual RSS frames, their position tables, and the X and Y offset with respect to the reference (that in most of the cases is the central exposure of the mosaic/dither). It produces as an output a single RSS which includes all the spectra included in each of the individual ones included in the configuration file, plus a position table that includes all the individual ones taking into account the offsets between pointings. However this routine does not have into account if the exposures of the mosaic have certain overlap. In the case of overlapping mosaics it is possible to use the routine `Mosaic_rss_overlap.pl` which the contiguous exposures using the overlaping spectra, replacing them by the average found after scaling:

syntax: `Mosaic_rss_overlap.pl CONFIG.txt RSS.FITS POSITION_TABLE.txt`

For a proper use of this task it is important that the offsets between adjacent exposures were selected in a way that the overlap is complete between spectra. An example of the use of this technique is the integral field survey of the Orion nebula presented in Sánchez et al. (2007).

In the case of dithering exposures that cover the complete field of view (e.g., three exposures with PPAK of 0,0 ; 1.56,0.78 and 1.56,-0.78 arcsec offsets), it is possible to construct a datacube without interpolating the data (sánchez 2006). This is done by a regularization of the data: over the field of view of the dithered exposures is constructed a grid of a certain pixel scale (smaller than that of the original fibers). Then, for each pixel of this grid is averaged the spectra that enters in this pixel, obtaining a final datacube. This is done with the procedure `rss2grid_new.pl`

syntax: `rss2grid_new.pl INPUT_RSS position_table.txt DPIX OUTPUT_cube [OFFSET_X Y]`

This tool requires as inputs the combined RSS and position table of the dither, i.e., the output of the `Mosaic_rss.pl` tool. It request the final pixel scale (DPIX), and produce an output datacube. Figure 8 illustrate the use of both the routines over dithered exposures with PPAK.

8 Creating pipelines: R3D_pmas.pl

One of the basic adventages of R3D is that being based on separeted routines, executed on the shell, and with a command-based system to change the input parameters, it is very easily to create pipelines to reduce the data.

There are two basic pipelines that can be created, the interactive ones, that request you some of the input parameters for each of the tasks to run at each step of the reduction, fixing the rest of the parameters to the default values for a certain instrument an setup. These pipelines are useful for testing the quality of the data (quick on-line reductions), and to reduce well behaved datasets. We have included one of them in the actual distribution of R3D, `R3D_pmas.pl`. This pipeline is able to perform the basic steps of the data reduction for PMAS data in any of the available configurations (lensarray or PPAK).

Once you have tested the required parameters for reducing a certain night dataset, it is possible to create non interactive pipelines. These ones can use configuration files to create the required routines to run R3D completely automatic.

9 Annotated list of routines and parameters

List of scripts (the corresponding C-program have similar name without the .pl, and share the same command line options):

- add_dist.pl
- check_trace.pl
- clean_cr.pl
- clean_ThAr.pl
- comp_spec.pl
- correct_AtExt.pl
- create_mask_borders.pl
- create_mask.pl
- create_mask_vimos.pl
- create_NS_trace.pl
- create_sky_clip.pl
- create_sky.pl
- create_sky_ppak_new.pl

- create_sky_ppak.pl
- create_sky_vimos_med.pl
- create_sky_vimos.pl
- cube2rss.pl
- cube_spec_arith.pl
- disp_cor_external.pl
- disp_cor.pl
- dist_cor_block.pl
- dist_cor_cross_ascii.pl
- dist_cor_cross.pl
- dist_cor_external.pl
- dist_cor.pl
- dist_cor_same.pl
- edit_peaks.pl
- extract_aper_CT.pl
- extract_aper.pl
- extract_gauss_external.pl
- extract_gauss_jacobi.pl
- extract_gauss_old.pl
- extract_gauss.pl
- extract_gauss_simple.pl
- extract_gauss_weight.pl
- extract_gauss_multi
- fiber_flat.pl
- fiber_flat_ppak.pl
- fiber_trans_flat.pl
- fiber_trans_flat_twin.pl
- flux_ratio.pl
- get_vimos_quad.pl
- glue_vimos_HR.pl
- glue_vimos_LR.pl

- imarith.pl
- img_spec_arith.pl
- img_spec_rat.pl
- mask_spec.pl
- match_peaks.pl
- mdist_cor_external.pl
- mdist_cor.pl
- mdist_cor_ppak.pl
- mdist_cor_sp.pl
- mdist_cor_ThAr.pl
- median_clean.pl
- median_column.pl
- median_cut.pl
- median_spec.pl
- min_spec.pl
- Mosaic_cubes.pl
- Mosaic_rss.pl
- mspec2img.pl
- my.pl
- new_split_vimos.pl
- old_rss2cube.pl
- peak_find.pl
- peaks_cross.pl
- peaks_mask.pl
- polyfit_mask.pl
- polyfit_no.pl
- polyfit.pl
- read_Euro3D.pl
- read_img_header.pl
- remove_cal_ppak.pl
- rename_vimos_org.pl

- rss2cube.pl
- rss2cube_pt_grid.pl
- rss2cube_pt_mask.pl
- rss2cube_pt.pl
- rss2grid_new.pl
- rss2grid.pl
- shift_spectra_cross.pl
- sigma_spec.pl
- smooth_trace.pl
- spec2img.pl
- split_ppak.pl
- split_vimos_ms_LR_b.pl
- split_vimos_ms_LR.pl
- split_vimos.pl
- straight_light.pl
- substract_NS_sky_extracted.pl
- substract_NS_sky.pl
- trace_peaks_cross.pl
- trace_peaks.pl
- trace_peaks_recursive.pl
- undo_split_vimos_ms_LR_b.pl
- undo_split_vimos_ms_LR.pl
- write_img_header.pl

Description of each scripts an its corresponding command line inputs (uncomplete):

* check_trace.pl TRACE.fits y_width

  Visualize every tracing in a trace file, over a width of "y_width" points.

  It requires:

  TRACE.fits -> Tracing file (output of trace_peaks.pl or trace_peaks_recursive.pl)
  y_width -> Plotting width.

  e.g., check_trace.pl VIMOS_IFU_LAMP018_0022_B.1.trc.fits 4

* comp_spec.pl SPEC1.txt SPEC2.txt ratio.txt

  Compare two spectra, stored in two differen ascii files, and derive the ratio between them.

  It requires: SPEC1.txt & SPEC2.txt -> Two three-columns ascii files with two spectra (id,wavelength,flux)
  ratio.txt -> output ratio.

* disp_cor.pl EXTRACTED.fits CRVAL CDELT APERTURE NY_SPECTRA WITDH NPOLY OUTPUT.fits DISPERSION.txt plot

  Program to perform the dispersion correction. It requires: EXTRACTED.fits => Input extracted and distortion corrected 2D spectra.
  CRVAL => Starting wavelength.
  CDELT => Wavelength step.
  APERTURE => Searching aperture for maximun searchings.
  NY_SPECTRA => Index of the spectrum to look for signatures.
  WITDH => Width of the aperture to coadd to generate the searching spectrum (+-width). NPOLY => Order of the polynomical function of the spectral correction.
  OUTPUT.fits => Dispersion corrected 2D spectral image.
  DISPERSION.txt => Dispersion solution.
  plot => Flag for plotting.

  e.g., disp_cor.pl run35_00385b.dist.fits 4406 0.3 5 128 5 4 run35_00385b.disp.fits run35_00385b.disp.
  1

* disp_cor_external.pl EXTRACTED.fits CRVAL CDELT OUTPUT.fits DISPERSION.txt plot

  Program to perform the dispersion correction using an external solution. It requires: EXTRACTED.fits => Input extracted and distortion corrected 2D spectra.
  CRVAL => Starting wavelength.
  CDELT => Wavelength step.
  OUTPUT.fits => Dispersion corrected 2D spectral image.
  DISPERSION.txt => Dispersion solution.
  plot => Flag for plotting.

  e.g., disp_cor_external.pl run35_00387b.dist.fits 4406 0.3 run35_00387b.disp.fits run35_00385b.disp.
  1

* dist_cor.pl EXTRACTED.fits CORRECTED.fits DISTORSION_CORRECTION.txt SMOOTH[0/
  start_index delta_index plot

  Distortion correction, without required a dispersion correction of the spectra. It cross-correlate the spectra in the file with the 1st spectrum to determine a shift that it is applied to each single spectrum.

  It requires:
  EXTRACTED.fits => Input extracted spectra.
  CORRECTED.fits => Output distortion corrected spectra.
  DISTORSION_CORRECTION.txt => Output distortion correction solution.
  SMOOTH[0/1] => Flag to smooth or not the distortion correction
  start_index,end_index => It defines the wavelength range to cross-correlate. NOTE:

end_index must be a power of 2: 16, 32, 128...
plot => Flag for plotting the results.

* dist_cor_cross.pl EXTRACTED.fits CORRECTED.fits DISTORSION_CORRECTION.txt SMOOTH[0/1] start_index delta_index plot

  Distortion correction, without required a dispersion correction of the spectra. It cross-correlate the spectra in the file with the 1st spectrum to determine a shift that it is applied to each single spectrum.

  It requires:
  EXTRACTED.fits => Input extracted spectra.
  CORRECTED.fits => Output distortion corrected spectra.
  DISTORSION_CORRECTION.txt => Output distortion correction solution.
  SMOOTH[0/1] => Flag to smooth or not the distortion correction
  start_index,end_index => It defines the wavelength range to cross-correlate. NOTE: end_index must be a power of 2: 16, 32, 128...
  plot => Flag for plotting the results.

* dist_cor_external.pl EXTRACTED.fits CORRECTED.fits DISTORSION_CORRECTION.txt start_index delta_index plot

  Distortion correction, without required a dispersion correction of the spectra. It use an ascii file as input to define the shifts. It requires: EXTRACTED.fits => Input extracted spectra. CORRECTED.fits => Output distortion corrected spectra. DISTORSION_CORRECTION.txt => Input distortion correction solution. start_index,end_index => It defines the wavelength range to plot. plot => Flag for plotting the results.

* extract_aper.pl RAW.fits Spectral_axis[0/1] TRACE.fits Aperture OUTPUT.fits [SHIFT]

  Algorithm for aperture extraction of the spectra. It requires:
  RAW.fits => The raw fitsfile with the spectra.
  Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
  TRACE.fits => 2D tracing file, output of "trace_peaks.pl"
  Aperture => Aperture for extracting around the traced peak (+-Aperture).
  OUTPUT => Output extracted spectra.
  SHIFT => Global Shift to the tracing Mask.

* extract_gauss_simple RAW.fits Spectral_axis[0/1] TRACE.fits Aperture OUTPUT.fits WIDTH PLOT [NAD] [SHIFT]

  Simple gaussian extraction. It select a certain ragion of +-Aperture width and perform a fit with a simple gaussian of a FWHM=width over the data to extract them. It subtract the contribution of the adjacent fibers to the considered one using this fit (NAD=N.Adjacent fibers), and perform again the fitting.

* extract_gauss_simple.pl RAW.fits Spectral_axis[0/1] TRACE.fits WIDTH OUTPUT.fits NX_MIN NX_MAX plot nplot

  Simple gaussian extraction. For each spectral pixel and each spectral peak the program fits a 1D gaussian extracting the integrated flux.

  It requires:
  RAW.fits => The raw fitsfile with the spectra.

Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
TRACE.fits => 2D tracing file, output of "trace_peaks.pl"
WIDTH => Number of pixels to be fitted around a certain position.
Aperture => Aperture for extracting around the traced peak (+-Aperture).
OUTPUT => Output extracted spectra.
NX_min,NX_max => Range of spectral pixels over which the fitting algorithm is applied.
plot [0/1] => Flag that allows to visualize the results.
nplot => Number of subframes for plotting.

It also produces three 2D images:
back.fits => 2D background image, derived from the fitting.
cen.fits => 2D Tracing, equivalent to the result from "trace_peaks.pl", but obtained from the gaussian fitting (centroid).
fwhm.fits => 2D image containing the FWHM of the fitted gaussian for each spectral pixel and each spectra.

* extract_gauss_weight.pl RAW.fits Spectral_axis[0/1] CEN.fits FWHM.fits ADDING_WIDTH ALLOWED_SHIFT_CEN OUTPUT.fits [SHIFT]

Gaussian weighted extraction. For each spectral pixel and each spectral peak the program extract an gaussian weighted flux. It requires:
RAW.fits => The raw fitsfile with the spectra.
Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
CEN.fits => 2D tracing file, output of "trace_peaks.pl" and/or "extract_gauss_simple.pl".
FWHM.fits => 2D image containing the FWHM of the fitted gaussian for each spectral pixel and each spectra.
ADDING_WIDTH => Number of pixels to be co-added around a certain position.
ALLOWED_SHIFT_CEN => Maximun tolerance for centroid offseting between sucesive centroids.
OUTPUT => Output extracted spectra.
SHIFT => Global Shift to the tracing Mask.

* extract_gauss.pl RAW.fits Spectral_axis[0/1] TRACE.fits WIDTH OUTPUT.fits NX_MIN NX_MAX plot nplot

Gaussian fitted extraction. It is an extension of "extract_gauss_simple.pl" where, after deriving the gaussian parameters for each spectral pixel and spectra, a LSQ fitting algorithm is applied over the 1D array at a certain spectral pixel fitting simultaneously all the gaussians (fixing the centroid and the FWHMs).

It requires:
RAW.fits => The raw fitsfile with the spectra.
Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
TRACE.fits => 2D tracing file, output of "trace_peaks.pl" and/or "extract_gauss_simple.pl".
WIDTH => Number of pixels to be fitted around a certain position.
OUTPUT => Output extracted spectra.
NX_min,NX_max => Range of spectral pixels over which the fitting algorithm is applied.
plot [0/1] => Flag that allows to visualize the results.
nplot => Number of subframes for plotting.

* extract_gauss_external.pl RAW.fits Spectral_axis[0/1] CEN.fits FWHM.fits OUTPUT.fits ALLOWED_SHIFT_CEN plot nplot

  Gaussian fitted extraction. Similar to "extract_gauss.pl", but using as tracing and FWHM an external file, instead of one determined over the data.

  It requires: RAW.fits => The raw fitsfile with the spectra.
  Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
  CEN.fits => 2D tracing file, output of "trace_peaks.pl" and/or "extract_gauss_simple.pl".
  FWHM.fits => 2D image containing the FWHM of the fitted gaussian for each spectral pixel and each spectra.
  OUTPUT => Output extracted spectra.
  ALLOWED_SHIFT_CEN => Maximun tolerance for centroid offseting between sucesive centroids.
  plot [0/1] => Flag that allows to visualize the results.
  nplot => Number of subframes for plotting.

  e.g., extract_gauss_external.pl run35_00385b.cl.fits 0 run35_00385b.strc.fits fwhm.fits run35_00385b.gs.fit
  2 0 6

* fiber_flat.pl SKYFLAT.fits FIBERFLAT.fits

  Creates the fiber-to-fiber transmision frame (wavelength dependent) using an extracted skyflat.

* fiber_trans_flat.pl SKYFLAT.fits FIBERFLAT.fits [NX_min] [NX_max]

  Create the wavelength independent fiber-to-fiber transmission frame, using an extracted skyflat, averaging the fiber-to-fiber transmission over a certain wavelength range.

* flux_ratio.pl UNCALIBRATED_SPEC.txt CALIBRATED_SPEC.txt ratio.txt FACTOR MEDIAN_BOX plot SMOOTH_BOX min_wave max_wave

  This program determines the ratio to transform from couts per second in flux per second. It requires:

  UNCALIBRATED_SPEC.txt => An 1D uncalibrated spectrum of a calibration star. It is normally extracted (using E3D) from a RSS distorsion corrected, wavelength calibrated and fiberflat corrected of a spectrophotometric standard star.
  CALIBRATED_SPEC.txt => A 1D calibrated spectrum of a calibration star in the units that we want to calibrate our science data. The format of this file is a ASCII table with the 1st column being the wavelength and the 2nd column being the flux. The formats of the calibrated spectrum of calibration star adopted in major observatories (like ESO or the HST) can be used. The spectra listed in the Calar Alto webpage of calibration stars can also be applied.
  ratio.txt => It a 1D spectrum storing the ratio or transformation between counts and fluxes. It can be latter transformed to a FITs file to apply it to the science data.
  FACTOR => It is the multiplicative factor applied to the uncalibrated data to transform them to fluxes per second. In the case of single fiber extracted spectrum, this factor is 1/EXPTIME. If E3D is used to extract the 1D spectrum, it is required to multiply by the number of coadded fibers: N.FIB/EXPTIME.
  MEDIAN_BOX => It is an smoothing factor to apply to the uncalibrated spectrum to match its resolution with that of the calibrated one.
  plot => A flag to run the program interactively or not [0=No plot, 1=Plot]

SMOOTH_BOX => It is an smoothing factor to apply to the calibrated spectrum to match its resolution with that of the uncalibrated one.

* mdist_cor.pl EXTRACTED.fits APERTURE NSIGMA NPOLY OUTPUT.fits DISTORSION.fits NX_min NX_max plot

  Correct the 2nd order of the distortion, with a polynomical correction for each spectra. It requires:
  EXTRACTED.fits => An extracted spectra, corrected for the 1st order with "dist_cor.pl".
  APERTURE => Searching aperture for maximun searchings.
  NSIGMA => N-Sigma over the background for searching new peaks.
  NPOLY => Order of the polynomical function.
  OUTPUT.fits => OUTPUT file.
  DISTORSION.fits => 2D distortion correction.
  NX_min => minimum spectral index for peak searching.
  NX_max => maximum spectral index for peak searching.
  plot => Interactive plotting.

* mdist_cor_external.pl EXTRACTED.fits DISTORTION.txt DISTORSION.fits OUTPUT.fits plot

  Correct the 2nd order of the distortion, using both the 1st order correction and the 2nd order correction derived from "dist_cor.pl" and "mdist_cor.pl". It requires:
  EXTRACTED.fits => An extracted spectra.
  DISTORTION.txt => The 1st order distortion correction derived from "dist_cor.pl".
  DISTORSION.fits => The 2nd order distortion correction derived from "mdist_cor.pl".
  OUTPUT.fits => Corrected flag.
  plot => Flag for interactive plotting.

* Mosaic_cubes.pl CONFIG.txt OUTPUT

* Mosaic_rss.pl CONFIG.txt RSS.FITS POSITION_TABLE.txt [NO_FITS]

  This routine re-arrange spatially a set of mosaic or dither RSS exposures to create a final RRS and a position table containing all the data of the mosaic or dither. It requires:

  CONFIG.txt => An ASCII configuration file in the format:
  infile in_position_table dx dy
  Where each "infile" is a single RSS fits file, the "in_position_table" is the corresponding position table of each "infile" and "dx" and "dy" are the offsets between this exposure of the mosaic/dither and any other one, in the units of the position table coordinates.
  RSS.fits => The output RSS file.
  POSITION_TABLE.txt => The output position table.

* peaks_cross.pl

  Peaks cross-correlation algorithm. It determines the peak intensity of each spectra in the CCD for the central spectral pixels, using as input the results from "peak_find.pl". It cross-correlate the previously found peak with the new cut determining a shift that accounts for the fleaxures.

It requires:
RAW.fits => The raw fitsfile with the spectra.
Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
PEAKS_FILE => Output file from "peak_find.pl".
x_width => The number of spectral pixels to coadd to look for the peak.
y_width => Pseudo-slit direction region for looking for each peak.
plot [0/1] => Flag that allows to visualize the results.
nplot => Number of subframes for plotting.
nsearch => Number of pixels to compare with the possible maximun.
y_shift_limit => Maximum tolerance of distance between the new peaks and the old ones.

e.g., peaks_cross.pl brun35_00451b.fits 0 brun35_00395b.peaks 0 5 0 4 2 0.1

* peak_find.pl RAW.fits Spectral_axis[0/1] Coadd_width plot nplot nsearch DMIN IMIN(% of the MAX) OUTFILE

Find the peaks of the spectra in the pseudo-list direction of the image. As a result it derives an ASCII file with the position of the peaks in the central spectral pixel. It requires as inputs:
RAW.fits => The raw fitsfile with the spectra.
Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
Coadd_width => The number of spectral pixels to coadd to look for the peak.
plot [0/1] => Flag that allows to visualize the results.
nplot => Number of subframes for plotting.
nsearch => Number of pixels to compare with the possible maximun.
DMIN => Minumun distance between adjacent peaks.
IMIN => Mimumun limit of intensity (in fractions of the maximun), for looking for a peak.
OUTFILE => Output file:

e.g., peak_find.pl run35_00385b.fits 0 1 1 4 2 3.5 0.07 run35_00385b.peaks

* rss2cube.pl input_RSS.fits output_cube.fits nx ny

Transform a Row-Stacked-Spectra fits file into a datacube, for regular grid IFS, when the spectra are order from left-top to right-bottom in the RSS file.

It requires:

input_RSS.fits -> The input RSS file.
output_cube.fits -> The ouput cube.
nx,ny -> size of the cube.

* rss2cube_pt.pl input_RSS.fits pos_table.txt output_cube.fits

Transform a Row-Stacked-Spectra fits file into a datacube, for regular grid IFS, when the spectra are not ordered from left-top to right-bottom in the RSS file, but following a certain position table.

It requires:

input_RSS.fits -> The input RSS file.
pos_table.txt -> position table (E3D format).
output_cube.fits -> The ouput cube.

* smooth_trace.pl TRACE.fits y_width Npoly OUT_TRACE.fits plot NX_min NX_max

  Smooth the 2D tracing, by fitting a N-order polynomial function.

  It requires:
  TRACE.fits => Input tracing file.
  y_width => Pseudo-slit direction region for plotting.
  Npoly => order of the polynomial function.
  OUT_TRACE.fits => Output tracing file.
  plot [0/1] => Flag that allows to visualize the results.
  NX_min,NX_max => Range of spectral pixels over which the straight-light is estimated.

* straight_light.pl RAW.fits Spectral_axis[0/1] TRACE.fits width Npoly staight_light.fits clean.fits NY_min NY_max plot

  Algorithm for determining and removing straight light contamination in the spectra. It requires enough "holes" between the spectra projected in the 2D image to derive a maenful result. It requires:
  RAW.fits => The raw fitsfile with the spectra.
  Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
  TRACE.fits => 2D tracing file, output of "trace_peaks.pl"
  width => The number of pixels in the pseudo-slit direction to mask around the spectral data.
  Npoly => Order of the polynomical function to fit over the non-contaminated data.
  straight_light.fits => 2D image with the estimated straight light.
  clean.fits => 2D image with the decontaminated data.
  NX_min,NX_max => Range of spectral pixels over which the straight-light is estimated.
  plot [0/1] => Flag that allows to visualize the results.

  e.g., straight_light.pl run35_00385b.fits 0 run35_00385b.strc.fits 6 8 run35_00385b.cl.fits run35_00385b. 0 2020 0

* trace_peaks.pl RAW.fits Spectral_axis[0/1] PEAKS_FILE x_width y_width plot nplot nsearch TRACE.fits

* trace_peaks_recursive.pl

  Tracing algorithm. It determines the peak intensity of each spectra in the CCD along the pseudo-slit direction for any spectral pixel, using as input the results from "peak_find.pl".
  It requires:
  RAW.fits => The raw fitsfile with the spectra.
  Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
  PEAKS_FILE => Output file from "peak_find.pl".
  x_width => The number of spectral pixels to coadd to look for the peak.
  y_width => Pseudo-slit direction region for looking for each peak.
  plot [0/1] => Flag that allows to visualize the results.
  nplot => Number of subframes for plotting.
  nsearch => Number of pixels to compare with the possible maximun.
  TRACE.fits => Output file.

e.g., trace_peaks.pl run35_00385b.fits 0 run35_00385b.peaks 0 5 1 4 2 run35_00385b.trc.fits

* trace_peaks_cross.pl

Tracing algorithm. It determines the peak intensity of each spectra in the CCD along the pseudo-slit direction for any spectral pixel, using as input the results from "peak_find.pl". It cross-correlate the previously found peak with the new cut determining a shift that accounts for the fleaxures.

It requires:
RAW.fits => The raw fitsfile with the spectra.
Spectral_axis[0/1] => The axis of the spectral dispersion, 0=X, 1=Y.
PEAKS_FILE => Output file from "peak_find.pl".
x_width => The number of spectral pixels to coadd to look for the peak.
y_width => Pseudo-slit direction region for looking for each peak.
plot [0/1] => Flag that allows to visualize the results.
nplot => Number of subframes for plotting.
nsearch => Number of pixels to compare with the possible maximun.
TRACE.fits => Output file.
y_shift_limit => Maximum tolerance of distance between the new peaks and the old ones.

e.g., trace_peaks_cross.pl brun35_00451b.fits 0 brun35_00395b.peaks 0 5 0 4 2 brun35_00451b.trc.fits 0.1

10 Bibliography

[Kelz et al. 2006] Kelz, A., Verheijen, M.A.W., Roth, M.M. et al., 2006, PASP, 118, 129

[Roth et al. 2005] Roth, M.M., Kelz, A., Fechner, T., et al., 2005, PASP, 117, 620

[Sánchez(2004)] Sánchez, S. F. 2004, Astronomische Nachrichten, 325, 167

[Sánchez(2006)] Sánchez, S. F. 2006, Astronomische Nachrichten, 327, 850

[Sánchez et al.(2007)] Sánchez, S. F., Cardiel, N., Verheijen, M. A. W., Martín-Gordón, D., Vilchez, J. M., & Alves, J. 2007, A& A, 465, 207

__oOo__