REDUCING GIANO SPECTRA with IRAF v1.2.0 (February 2015)

F. Massi[1] (fmassi@arcetri.astro.it), N. Sanna[1], T. Pecchioli[2]

[1]INAF - Osservatorio Astrofisico di Arcetri
[2]Dipartimento di Fisica e Astronomia, Università di Firenze

Abstract

GIANO is a near infrared echelle spectrograph operating at Nasmyth focus A of Telescopio Nazionale Galileo. It is a second generation instrument and allows obtaining spectra that cover most of the range 0.9–2.5 μm *with a single exposure. The spectral interval is covered through 49 orders and only a few small windows fall outside the detector. This document explains how to reduce and extract GIANO calibrated spectra by using routines available in any basic installation of IRAF, the Image Reduction and Analysis Facility developed by the NOAO, plus a few ad-hoc scripts. IRAF is one of the most commonly used scientific suites of software packages for reducing and analysing images and spectra, freely downloadable, and offering many routines which are both fully optimised and tested. Its versatile Command Language also allows scripting and easy customisation.*

Reduction quick guide

Some IRAF basics for new users are dealt with in Sect. 2. The reduction pipeline described in this handbook is summarised below.

1. Install the small set of scripts GIANO_TOOLS as explained in Sect. 3.1.
2. Open an IRAF session and load GIANO_TOOLS (which will have also ASTUTIL, IMRED, and ECHELLE loaded).
3. Move to your working directory where the data are stored.
4. Update the headers of the reference wavelength lamp frames, the target frames, the telluric star frames, and the spectro-photometric star frames (if any) using GIANO_HEADER, as explained in Sect. 3.3.
5. Average together the dark frames with the same integration time with IMCOMBINE.
6. Subtract the corresponding dark frame from the flat-field frames and the single science or calibration frame **which were not taken in AB cycles, i. e. by shifting alternatively the target between fibres**, using IMARITH.
7. Perform bad-pixel correction of all flat-field frames, reference lamp frames, science frames, and calibrator frames with clean_up as explained in Sect. 3.4.
8. Average together the flat-field frames with IMCOMBINE.
9. Set the ECHELLE parameters as discussed in Sect. 3.9.
10. Find the traces after scattered light correction with GIANO_FIND_TRACES as explained in Sect. 3.10 and Appendix A. Set the parameters as in Fig. 11. See Fig. 10 for a typical map of scattered light. See Figs. 20, 21, and 22 for a correct order numbering.
11. Fit the traces with APFLATTEN and construct a normalised 2D flat-field frame from the average flat-field frame as explained in Sect. 3.11 and Appendix B. See Fig. 12 on how to set the parameters for APFLATTEN. See Figs. 24, 26, and 25 for examples of actual trace fits. See Figs. 13 and 14 for an example of a 2D normalised flat-field frame. **Skip the signal fit step if you decide on performing a 1D flat-field correction (see Sect. 3.5 for details).**
12. Divide all corrected reference lamp frames, science frames, and calibration frames by the normalised flat-field frame using IMARITH. **Skip this step if you decide on performing a 1D flat-field correction (see Sect. 3.5 for details).**
13. Subtract the corrected frame B in every AB cycle set from the corresponding corrected frame A using IMARITH.
14. In case of more AB cycles on the same target, average or sum together the subtracted frames using IMCOMBINE. See Appendix E to compute the effective exposure time, gain, and read-out noise. Do the same with dark-corrected frames of sources not observed in AB cycles.
15. Sort the 4×49 traces retrieved to reflect the correct spectral order scheme, using the average flat-field frame input to APFLATTEN and the task SPLIT_FILE, as explained in Sect. 3.14. The task makes four copies of the flat-field frame and fix both the trace definition and the sky windows (see Fig. 22) to use for background subtraction (if requested). The output file names will have

the following string added: “lowest” (i. e., the associated traces are only defined for the lowest tracks in each four-track group with the correct spectral order numbering), “middown”, “midup”, and “topmost”.

16. Make four copies of each corrected reference lamp frame, subtracted science frame and dark-corrected science frames to reflect the lowest-middown-midup-topmost scheme of the reference trace files. Use the task COPY_FILE as explained in Sect. 3.13. The name of the new files will contain the strings “lowest”,‘middown”, “midup”, and “topmost”.

17. Assign the correct trace definition to the newly sorted science frames using APEDIT as explained in Sect. 3.14. See Fig. 15 on how to set the parameters for APEDIT.

18. Extract and wavelength-calibrate the 1D spectra from the newly sorted “lowest” science frames using DOECSLIT as explained in Sect. 3.14. Set the parameters as in Fig. 16 for DOECSLIT and Fig. 17 for SPARAMS. Skip the trace finding and trace fitting steps in the interactive session. **It is mandatory that you use the reference lines shown in Figs. 33, 34, and 35 to obtain an accurate wavelength calibration.** See Fig. 30 for a correct wavelength solution.

19. Use GIANO_REIDENTIFY and the (now) wavelength-calibrated reference lamp “lowest” spectra to also wavelength-calibrate the “middown”, “midup, and “topmost reference lamp spectra as explained in Sect. 3.14.

20. Now extract and wavelength-calibrate the 1D spectra from the newly sorted “middown”, “midup, and “topmost science frames using DOECSLIT as explained in Sect. 3.14. Set the parameters as in Fig. 16 for DOECSLIT and Fig. 17 for SPARAMS. Skip the trace finding and trace fitting steps, and the wavelength calibration step in the interactive session.

21. If you have not flat-field corrected the spectra yet, you can do it now in 1D by using flat_1D, as explained in Sect. 3.15.

Troubleshooting

This handbook also deals with a few problems that may arise during the reduction steps; just look for specific boxes labelled as "Remark" or "Problem solving". Here is a brief summary.

1. **The graphic or image cursor does not work** See the REMARK on page 4.
2. **Error when loading** GIANO_TOOLS **in IRAF** See the REMARK on page 6.
3. **Some** ECHELLE **tasks do not run using correct parameters** See the REMARK on page 13.
4. GIANO_FIND_TRACE **does not find the correct aperture sequence** See the PROBLEM SOLVING section on page 16.
5. **The wavelength calibration is not correct in small intervals inside some orders** Use all the reference lines suggested in Appendix G for the preliminary wavelength fit with DOECSLIT.

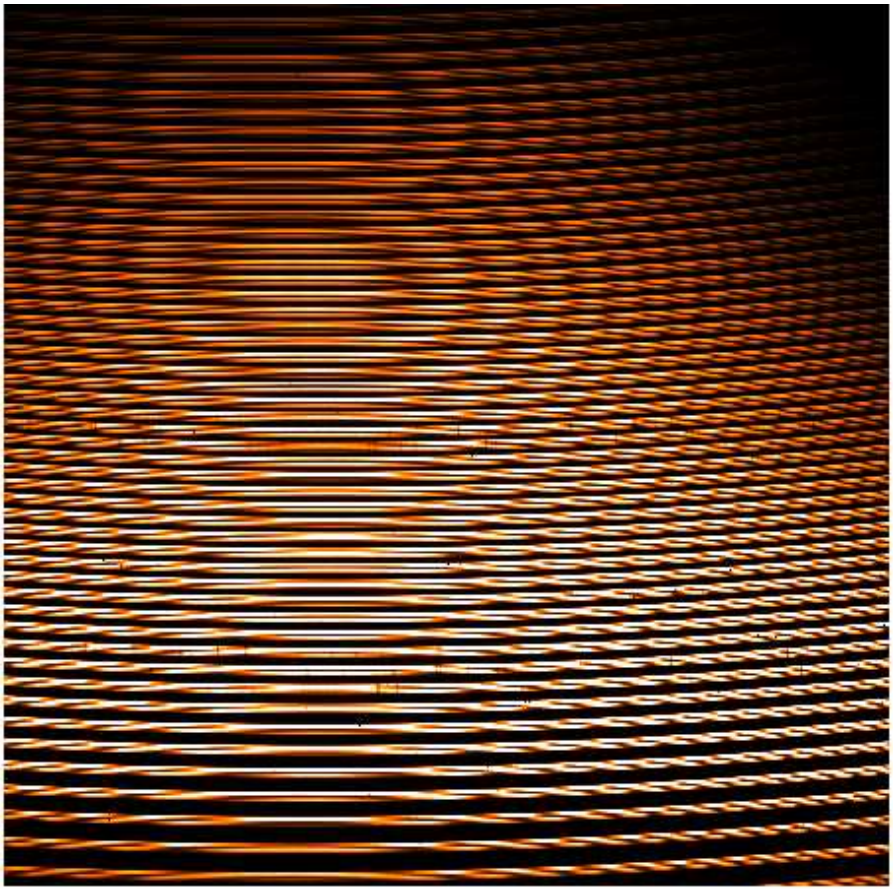

Figure 1: Frame with the spectrum (echellogram) of a continuum lamp obtained with GIANO

1 Starting point: GIANO raw data

A typical 2D output from a GIANO integration is shown in Fig. 1. The frame contains 49 groups of 4 arc-shaped tracks (plus a couple of groups of shorter segments actually), as clear in the zoom-in of Fig. 2. Each group represents a spectral order and contains a part of the spectrum falling inside a sub-range of the total pass-band. The wavelength intervals per order are listed in Appendix H. Any group is composed of two pairs of arcs, each pair corresponding to the output of one of the two optical fibres used to transfer the signal from the telescope focal plane to the spectrograph. Image-slicers split each optical fibre output in two smaller spots arranged along the slit. Through the optical fibres, two different sky areas 3-arcsec apart are fed into the spectrograph. The detector parameters are listed in Table 1.

The lowest order (at the bottom of the frame) is the 32nd and the highest one (at the top of the frame) is the 80st. Although you can count 200 arc-shaped tracks, which includes part of order 81, as well, only orders 32 through 80 are offered. Within each order, wavelength increases from right to left. In addition, wavelength decreases with increasing order (see Appendix H). One of the goals of the reduction procedure is to find the curve that best fits the signal peak in each sub-track and to count the signal as a function of position within "apertures" centred on these curves (called *traces*). More information on the instrument can be found in [1]. The most important detector features are listed in Table 1.

A run with GIANO will yield images recording the target spectra (2D spectra), as long as a few additional "calibration" frames. These are needed to properly extract the target spectra. Below, we list the sets of calibration frames necessary for accurate extraction and calibration.

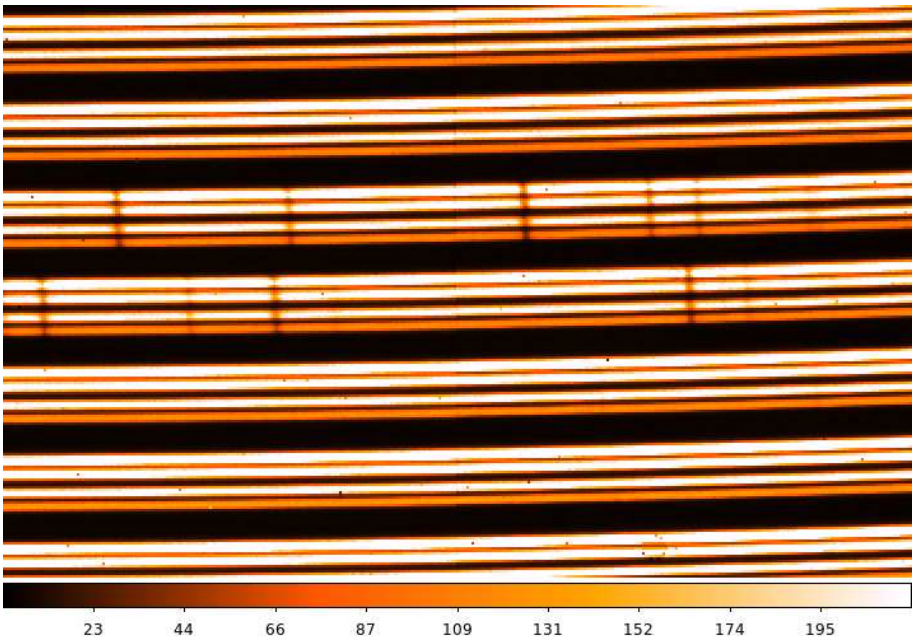

Figure 2: Zoom-in on a Giano image of a continuum lamp. Note that each order is split in four parallel tracks.

DARK FRAMES these are obtained by integrating a signal while holding the shutter closed. Science targets are usually observed by nodding-on-fibres, i. e. by performing an integration with the target in fibre A first and in fibre B next (hereby, a cycle AB). After subtracting B from A, the resulting 2D spectra will both exhibit the target signal in all sub-tracks and have most of the bias removed, so no dark subtraction is required in this case. On the other hand, 2D spectra not taken in AB pairs (flats, reference lamps, etc.) need being dark subtracted. Sets of dark frames will be acquired for all exposure times of non-AB 2D spectra.

FLAT FIELD FRAMES these are obtained by illuminating the fibres with an intense continuum lamp. Flat field frames are needed not only to remove the pixel-to-pixel inhomogeneities in the detector response, but also to correct for the spectrograph response function (which depends on the order and, within each order, on wavelength), and to obtain a trace template that will be then used on the usually noisier and much fainter scientific 2D spectra to extract the 1D spectra.

REFERENCE LAMP SPECTRA reference lamp spectra are obtained by illuminating the fibres with an intense EMISSION LINE lamp, (GIANO is currently equipped with a U-Ne calibration unit). These frames are required for wavelength-calibration of the spectra.

TARGET AND CALIBRATOR (TELLURIC OR SPECTRO-PHOTOMETRIC STARS) SPECTRA these are usually pairs of AB, nodded-on-fibres integrations, but you might also have 2D spectra taken with the target in only one fibre (possibly with corresponding 2D spectra with pure sky emission). If no sky frame is available, subtraction of dark frames with the same integration time are mandatory. See [2, 3] for more details on the observational techniques and the recommended integration times.

REMARK All GIANO observations are saved as FITS files.

Table 1: Main characteristics of the GIANO NIR detector (from Table 1 of [2]).

gain	2.2	e^- ADU^{-1}
Read-Out Noise	5	e^-
Non-linearity	$1500 \times t_{\text{exp}}$	ADU

[1] t_{exp} total integration time in sec.

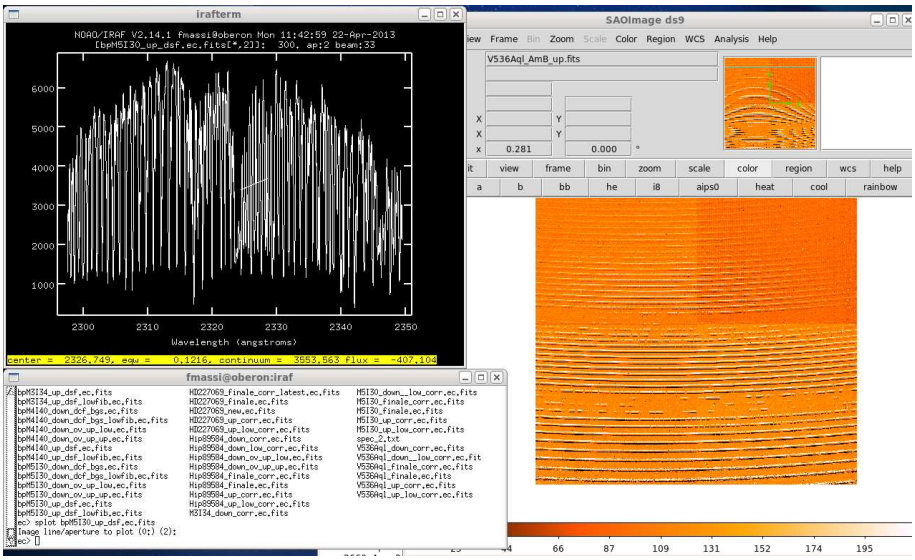

Figure 3: The three basic interfaces used in a typical IRAF session: an xgterm window for running commands and tasks (bottom left), an image tool (here ds9, right), and a graphic window (top left).

2 IRAF basics

This section is only intended as a quick introduction to allow new users to run a simple IRAF session. For more information, we recommend that you should read [4] and [5].

You need to have IRAF installed in your PC or network. We also suggest that you install x11iraf (providing you with xgterm terminal windows) and ds9 (to use as the image tool).

IRAF and x11iraf can be downloaded from
http://iraf.noao.edu
ds9 can be downloaded from
http://hea-www.harvard.edu/RD/ds9/site/Home.html.

Once IRAF is installed on your PC, you just need to create your IRAF home directory, the one which you will always start up IRAF in to work on your data. Then, run **mkiraf** on that directory and answer the questions. The most critical one is that about the terminal type. If you have x11iraf installed, choose "xgterm". Now all is set up for running IRAF.

Basic IRAF session: first, open an xgterm terminal window. Then, start ds9. On the xgterm window, move to your IRAF home directory and type "cl". The IRAF session will open up and you will be shown the prompt "cl>". From the prompt you can run line commands and tasks. A typical

IRAF session uses three windows (see Fig. 3): an xgterm one with the CL running, an image tool (ds9 in figure) and a graphic tool. The graphic tool will open up when you first run a task requiring it.

REMARK Lots of IRAF tasks are interactive and they cause either an image cursor or a graphic cursor to open. These allow you to point specific features inside the image tool (image cursor) or the graphic window (graphic tool) and issue interactive commands (usually by pressing sequences of keys).

BE CAREFUL: if you need to run interactive commands, first take care to click the mouse left button with the cursor inside the image tool or the graphic window, depending on whether you are using the image or the graphic cursor. Then (and only then), issue the command. Remember to always have the proper window active every time you run an interactive command! When you need to return to the CL, use the interactive command "q".

Once cl is running, you can move to your working directory by using "cd". Some shell commands like ls, mkdir, etc. are recognised within cl. In any case you can run any external command or program by typing a "!" in front of the command line to be entered. Type in "logout" to quit cl.

IRAF is a bundle of layered programs divided into several packages. Only a limited number of basic packages will be available and running (i. e. , "loaded") as soon as cl is opened. To run any task from any other package, you first need to load the whole package by simply typing its name and pressing enter. In this case, the prompt changes from "cl>" to another string with the first two characters of the package name. More than one package can be loaded simultaneously (this increases memory allocation; it should be less than a problem on modern PCs but IRAF may meet with some problems). The latest package loaded can be unloaded by typing "bye". Here is an example of loading a package (in this case, noao). Note that as soon as the package is loaded the list of its packages and/or tasks will be shown. Note also that the prompt has changed:

```
cl> noao
            artdata.     digiphot.   nobsolete.   onedspec.
            astcat.      focas.      nproto.      rv.
            astrometry.  imred.      observatory  surfphot.
            astutil.     mtlocal.    obsutil.     twodspec.
no>
```

A help facility is available, just type "help nametask" in cl to find out the help pages about the task NAMETASK.

Most IRAF task need a few parameters being set before running. The simplest way to do this is by using the IRAF parameter editor. You can access the NAMETASK parameter file through the editor by typing "epar nametask" in cl (for an example see Fig. 4). A number of fields will be shown in the cl window. You can select any of them by using the arrow keys and edit it (just type in and press enter). Press CTRL-d or type ":wq" to exit the parameter editor, or type ":go" to exit the parameter editor and make the task run.

Alternatively, you can run a task by simply typing its name and pressing enter. IRAF remembers the latest set of parameters entered for a given task *through the parameter editor*, so by default the task will run using that set. Some tasks can run in verbose mode; in this case you will be asked to confirm each parameter of the set. You will be able to either just confirm a parameter or enter a new value. Parameters can also be passed to a task via a line command (e. g., by typing
"nametask par1 par2 ...",

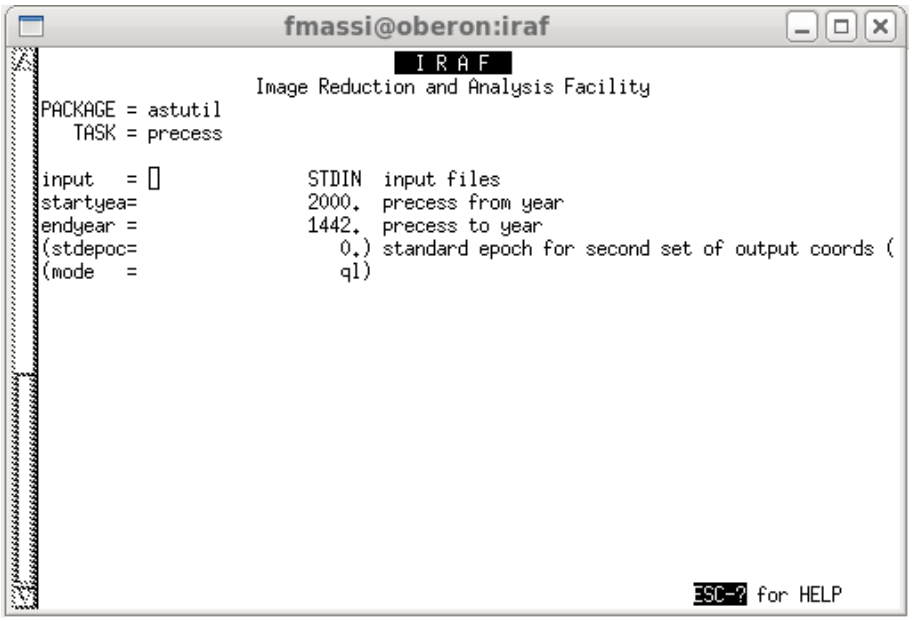

Figure 4: IRAF parameter editor opened on the parameter file of the task "precess". You can move through the fields by using the arrow keys and you can input some text in the field evidenced by the cursor. The old text will automatically be replaced with the latest one after pressing ENTER.

or
"nametask field1=value field2=value ...").

You can reset all the parameters of a given task to their default values by typing "unlearn nametask" at the cl prompt.

REMARK Most IRAF tasks can accept, as input and output fields: single file names, multiple file names separated by commas, single text file names including lists of files. The latter mode is useful to operate on whole blocks of spectra in a single shot. Files have to be listed one per line in the text file, then the task has to be informed that the given name refers to a list of files by typing a "@" in front of the file name. For example, by typing:
cl> imstatistics @image.list
the task IMSTATISTIC is run to yield the statistics of all images listed in the file image.list.

3 Data reduction with IRAF

3.1 Setting up an IRAF session to reduce data from GIANO

After running MKIRAF on your IRAF home directory (see page 3), you will be able to find a file called "login.cl" there, containing some basic paths, a few settings, and the tasks to be loaded whenever cl is run. Lines commented out have a hashtag # as the line first character, so they are not read by cl at start time.

Edit your command.cl file, find the line

set stdimage = imt...

check that it is **not** commented out and if necessary modify it to:

set stdimage = imt2048

Then check that the line

set imtype = "imh"

is **commented out**, so that the default image file format is fits and not imh. You will no longer need to type in the full name of an image including the extension "fits", IRAF will now be able to recognise a fits file unless it is a fits file using a different extension (e. g., fts; see page 7 to change extension).

You will also need to install a small package (called GIANO_TOOLS) containing a few scripts which we have developed to the purpose and is provided in the form of a tar file. Make a new directory to store the package, move the tar file there and untar it. One of the extracted files is called giano_tools.cl. Edit giano_tools.cl and change the line

set gianot = "put here the correct path to giano_tools"

by inserting the actual full path between quotes. **Make sure that the last character of the path is a "/".** Finally, edit the login.cl file in your IRAF home directory and add the line

task $giano_tools = path/giano_tools.cl

where path is the full path to the directory where you have untarred the package. Now you will be able to load the package by simply typing "giano_tools" in cl. Note that the package also contains a fits file with an updated map of detector bad pixels (badpix_mask.fits), a fits file with an updated wavelength-calibrated lamp spectrum (cl_29_jul_UNe_300_ds_lowest.ec.fits) and its wavelength-solution file (eccl_29_jul_UNe_300_ds_lowest.ec), and a list of reference wavelengths for calibrating spectra using a U-Ne-Ar lamp (U_Ne_Ar_lines_list_cl.dat).

REMARK: INSTALLATION ERRORS Sometimes, after installing it we got error messages when loading GIANO_TOOLS, due to hidden characters inserted in giano_tools.cl by the text editor used to edit it. Should this happen, just try editing the original file with a different text editor.

3.2 Giano_tools

GIANO_TOOLS is a collection of IRAF scripts we have developed to adapt IRAF tasks and reduce GIANO spectra. They are provided as a tar file and include the following files:
badpix_mask.fits
cl_29_jul_UNe_300_ds_lowest.ec.fits
clean_up.cl
copy_file.cl
eccl_29_jul_UNe_300_ds_lowest.ec
fits2text.cl
flat_1D.cl
giano_find_trace.cl
giano_header.cl
giano_reidentify.cl
giano_tools.cl
split_file.cl
U_Ne_Ar_lines_list_cl.dat

3.3 First reduction steps

Each night run will yield a number of dark frames, flat-field frames, line reference lamp frames and scientific frames. We suggest that you should make a directory for each night and save the corresponding frames there. We assume that you have taken one complete set of calibration frames per each night. If possible, keep a copy of each of these directories in your disk, since a few tasks you are going to use will modify the frames they operate on. If something goes wrong, it will sometimes be quicker to just erase the directory, e. g. with "rm -r /path-to-yourdirectory/*", and copy there the original frames again.

Open an xgterm window, change the current directory to your IRAF home directory in the new window, and run cl. Within the cl prompt, change directory to the one where the data are stored. If their extension is not fits, e. g. fts, you need to change it to fits. This is easy done by the line command:

```
cl> rename *.fts fits field=extn
```

Now, you need to load the packages ASTUTIL, IMRED, ECHELLE, and GIANO_TOOLS. But typing giano_tools is enough, since it will automatically load the other three packages.

Before starting to work on the spectra, you will also need to add a few fields to the headers of the reference lamp spectra, the target spectra, and any calibrator spectra. Just list all lamp reference frames in a text file (e. g., lamp.list), one per line. Then list all target and calibrator spectra in another one (e. g., target.list). Edit the parameter file of the GIANO_TOOLS task called GIANO_HEADER (running "epar giano_header") and set *list_obj* to target.list, *list_com* to lamp.list, *obsvt* to lapalma, and *equi* to 2000. Therefore, the editor page must look as follows:

```
PACKAGE = giano_tools
  TASK = giano_header
      list_obj=    target.list   File listing the target spectra:
      list_com=    lamp.list     File listing the comparison spectra:
      obsvt =      lapalma       Observatory (from the IRAF observatory db):
      equi =       2000.         Equinox which RA and DEC in the header are refferred to:
    (mode = q)
```

Exit starting the task by typing ":wq". Run the task twice, just to make sure everything is updated as needed.

3.4 Dark subtraction and bad-pixel correction

Dark subtraction is only needed by those frames that have not been taken in AB cycles (i. e., pairs of frames obtained by shifting the target from one fibre to the other will be subtracted from each other and do not need dark-subtraction). These include flat-fields, lamp reference spectra, and every target frame with spectra from one fibre only. **We stress again that the dark frames to subtract must have been taken with the same integration time as the image to be dark-corrected!** Thus, group together all dark frames according to their integration time. Check that none of the dark frames exhibit any kind of problems by displaying them (use the task DISPLAY) and having a look at the image statistics (use the task IMEXAMINE). Let us assume that all darks with an integration time of 300 s (let us call them dark_300_1.fits, dark_300_2.fits, dark_300_3.fits) are listed in a file called dark_300.list:

```
cl> display dark_300_1 1
cl> display dark_300_2 2
cl> display dark_300_3 3
cl> imstat @dark_300.list
```

The last number in each of the display line command is a frame identifier. E. g., in practice IRAF allows nine frames (1,2,3,4,5,6,7,8,9) to be displayed simultaneously on ds9. You can use the ds9 key "blink" (in the menu "frame" of ds9, see Fig. 5) to iteratively display all open frames, or you can shift from one frame to another by using "next" (in the menu "frame" of ds9). Drop any file from the dark_300.list that exhibits some strange patterns or statistics. Then combine the remaining ones together, using the task IMCOMBINE after updating the list of frames to be combined.

```
cl> imcombine @dark_300.list dark_300_combined combine=average
```

This will yield the file dark_300_combined.fits, which has to be subtracted from each frame with an integration time of 300 s needing dark correction.

The quickest way of dark-correcting all frames that need it, is by listing them (one per line) in a text file (let us call it to_be_dcorrected.list). Then do a copy of to_be_dcorrected.list (e. g., let us call it dcorrected.list), edit this new file, and change the name of each item listed (e.g., by adding the string "dc_" in front of each file name). Now you will be able to dark-subtract all files just by typing:

```
cl> imarith @to_be_dcorrected.list   -   dark_300_combined @dcorrected.list
```

This construct the dark-subtracted frames and saves them adding a "dc_" in front of their original file names (e. g.,target_1.fits will yield dc_target_1.fits).

After dark subtraction, We suggest that you apply bad-pixel corrections to all flat-field frames, calibration-lamp frames, target frames and calibrator frames. Bad pixels cause a lot of annoyance when using tasks that perform fits to pixel rows or columns. The correction can be done by using the GIANO_TOOLS task CLEAN_UP. Just make a text file listing all frames to be bad-pixel corrected (let us call it to_be_flatted.list). Then open the clean_up parameter file by typing "epar clean_up". Set the fields as below:

```
PACKAGE = giano_tools
  TASK = clean_up
```

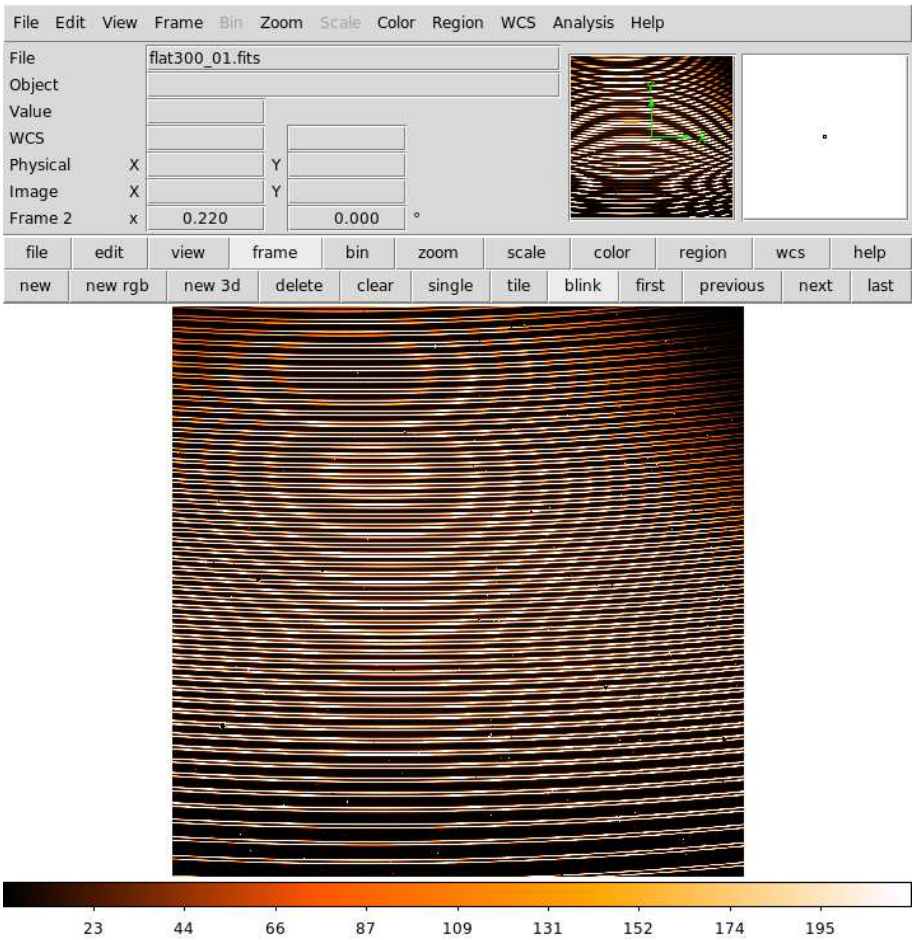

Figure 5: How to run "blink" on ds9: first click on *frame* on the upper part of the bar above the image. This will change the lower part of the bar and allow you to click on *blink*. Click on *single* to exit from blink.

imlist=	to_be_flatted.list	File with list of images:
outstr=	cl_	String to be added to input image name:
badima=	gianot$badpix_mask.fits	Bad pixel image:
xbox=	40	Size of median-filter box along X:
ybox=	1	Size of median-filter box along Y:
fix=	no	Use fixpix?:
fixfile =		List of bad pixel masks for fixpix:
(mode = q)		

When you have finished, exit by typing ":go" to start the task. It will construct bad-pixel corrected frames and save them adding the string "cl_" (you can choose any string you want, just update the field *outstr*) in front of the names of the input frames. The task uses a bad-pixel map already provided with GIANO_TOOLS (badpix_mask.fits) to replace each mapped bad pixel with an xbox-by-ybox median value around it. If you want, you can also have clean_up run the IRAF task FIXPIX (by setting *fix* to yes) to correct groups of pixels whose coordinates are to be listed in a file (by entering its name in *fixfile*). See the IRAF help for further information on fixpix and the file format.

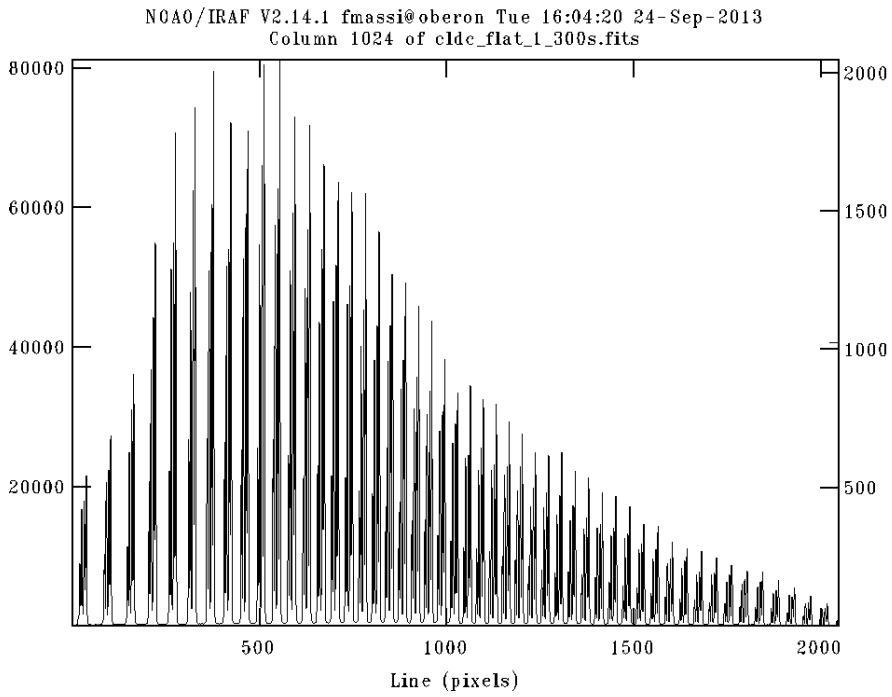

Figure 6: Cut along the middle column of a typical 300 s flat-field frame.

3.5 Learning more about flat-field frames

A typical flat-field frame is shown in Fig. 1. You can have a look at a 1D section of the frame by using the task IMPLOT. Just choose one of your flat image, e. g., flat_1.fits, and type "implot flat_1.fits" followed by enter. This will open the graphic window. Activate the window by clicking on it, and type ":c 1024" followed by enter. This will display an image cut along the middle column of the frame, like in Fig. 6. This flat was exposed for 300 s, thus the top counts are roughly those you should expect when using the same integration time. Note how the spectrograph efficiency varies with the spectral order (remember that the lowest order corresponds to the bottom-most track, hence the left-most in figure).

To look more carefully, just zoom-in on a smaller section: move the graphic cursor to the left-bottom corner of the area you want to display and press "e". Then move it to the right-top corner and press "e". You will be displayed something like in Fig. 7. Note the four different sub-tracks in each order, as explained at the beginning. The differences in peak counts between the four tracks of each order are due to the different efficiencies of the two fibres and to the positioning of the source in the fibre. To go back to the previous display, type "a", and then "a" again. Type "q" to return to the command language prompt.

There is another important effect to be aware of about flat-field frames. Try subtracting a flat-field frame form a different one, provided both belong to the same set of exposures (e. g., by running "imarith flat_1.fits - flat_2.fits flat_1-2.fits"). You will obtain an image like the one shown in Fig. 8. Clearly, each track in each order exhibits both a positive and a negative peak, although with different values. This is due to very small intensity source variations between the two exposures (less than few percent), and to micro-shifts of the fibres (fraction of a pixel). These are acceptable and you should not worry about it.

3.6 2D or 1D flat field correction?

CCDs and NIR detectors exhibit pixel-to-pixel variations in their quantum efficiency. These need being corrected for in order to increase the signal-to-noise ratio. The issue is more complicated for images

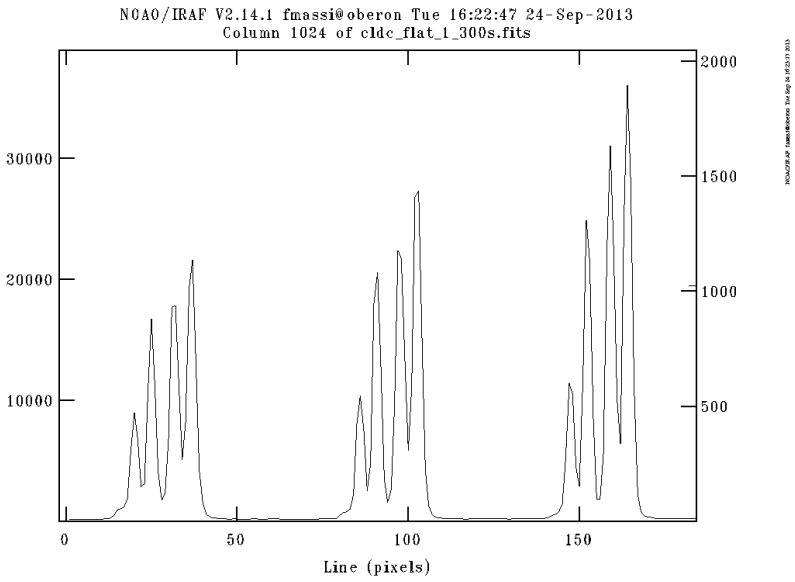

Figure 7: Zoom-in on a segment of the middle column of a typical 300 s flat-field frame.

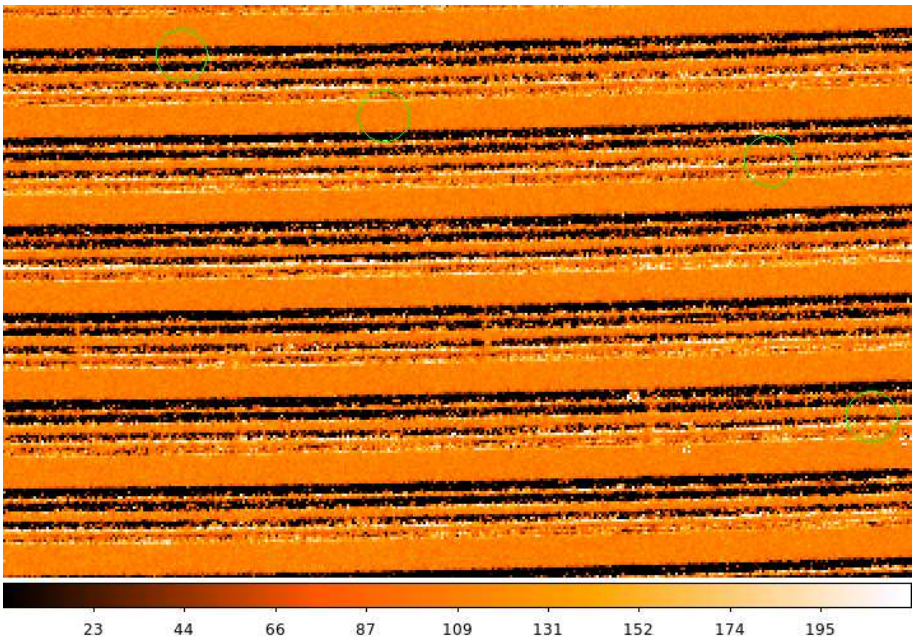

Figure 8: Zoom-in on a small area around the middle column of a typical 300 s flat-field frame.

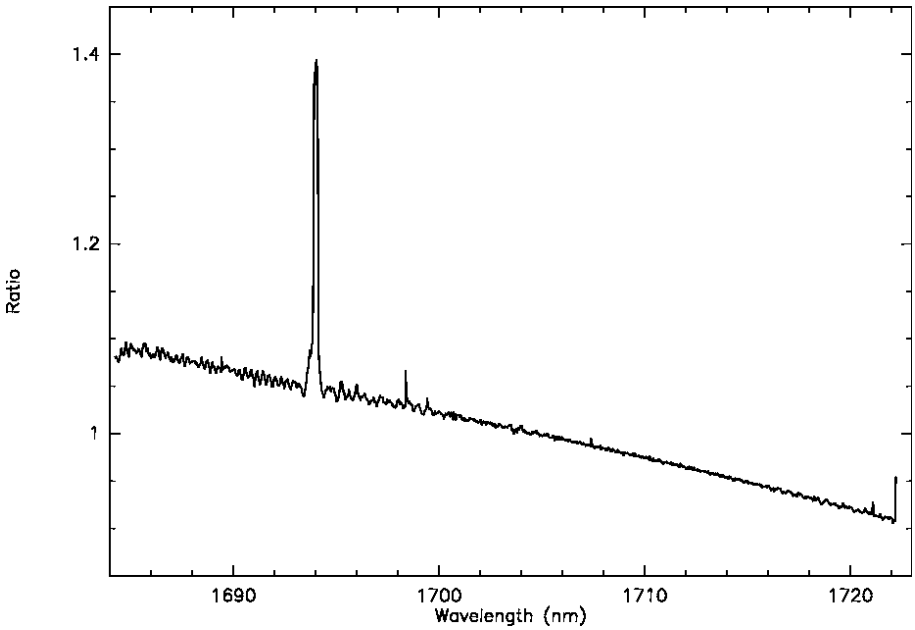

Figure 9: Ratio of spectra from the same exposure on Deneb, after 1D flat correction and 2D flat correction. The spike at $\sim$ 1694 nm is due to bad pixels.

of echelle spectra, due to the large fraction of the detector area (between orders) exposed to a very low flux. To avoid various degrading effects on the noise statistics, flat-field corrections should only be made inside each aperture (i. e., the detector areas impinged by the signal from the fibres and image slicers).

In addition, since GIANO is a fibre spectrometer, no spatial information is preserved inside each order. So, two ways of flat-field correcting the spectra are possible:

1. construct a 2D flat-field map by deriving a flat field correction only inside the apertures (setting the pixels outside the apertures to 1) and divide each frame by this map;

2. extract the 1D spectrum from each aperture of the flat-field image, normalise it, and divide all extracted (1D) unflatted spectra by the corresponding normalised flat-field 1D spectra (see Sect. 3.15 for detail).

The results are quite similar, so the 2D technique is redundant. This is demonstrated in Fig. 9; a spectrum of the star Deneb was extracted and flat-field corrected in the two ways. The ratio of the resulting spectra is displayed in Fig. 9. Disregarding the low frequency component (the large scale, smooth variation from 1.1 to 0.9), the high-frequency variations are below $\sim$ 1 %, which means that both techniques lead to the same level of accuracy. None the less, making a 2D flat-field frame allows you to define the traces for all orders and check that the subsequent spectrum extraction is ok. So, We will describe here the 2D technique. Due to the very high signal-to-noise ratio in all orders of the continuum lamp frames, they are the best images to derive the traces that will be used to extract any other spectrum. This works fine since the spectrograph unit does not move during the observing run and the traces are then not expected to change. On the other hand, science target spectra are often faint in most orders and are thus not suitable to derive their own accurate traces.

3.7 Averaging the flat-field frames

First of all, you have to combine together all flat-field frames (after dark subtraction and bad-pixel correction) to enhance the signal-to-noise ratio. Using the image tool and the task IMEXAMINE, check

all flat-field frames and discard those displaying any problem. Then list the good files (one per line) in a text file (let us call it "to_be_flatted.list"). The frames can be combined together by using the task IMCOMBINE:

cl> imcombine @to_be_flatted.list flat_cmb combine=average

In this case, the averaged final frame will be called "flat_cmb.fits" (second argument in the line command). This is the image you will use not only to construct the 2D flat-field map, but also to find and fit the traces.

3.8 Handling spectra: IRAF packages to use

The package we will use to handle GIANO spectra (i. e., to construct flat-field frames, extract spectra, etc.) is called ECHELLE. So remember to load it. To do it, first you have to load IMRED, then you can load ECHELLE (just type "imred", enter, "echelle", enter). However, if you already have GIANO_TOOLS loaded, ECHELLE has been automatically loaded, as well (see page 7).

REMARK IRAF has other packages that make use of some of the routines managed by ECHELLE (e. g., ONEDSPEC, TWODSPEC, etc.). Be careful, if you have one of these packages loaded over ECHELLE and run one of this common routines, IRAF will read its parameters from a different parameter file than the one associated with ECHELLE! Thus MAKE SURE that either GIANO_TOOLS or ECHELLE are the only spectrum analysis packages loaded when using ECHELLE routines.

3.9 Setting an ECHELLE session to extract GIANO spectra

After loading ECHELLE (or having GIANO_TOOLS already loaded), a few critical parameters need to be set once and for all. Hereafter, a list of the necessary steps are summarised. Be aware that many ECHELLE tasks will output new fits files, write some new information in the frame headers, and save a lot of parameters in text files. These text files are saved in a subdirectory called (by default) "database" that will be made by ECHELLE itself. This is why removing a newly-created frame is often not enough to go one step back in the reduction process. ECHELLE keeps memory of what has been done through header fields and text files, and ALWAYS uses this information any time you re-run a given task from the same directory.

Setting the dispersion axis (i. e., is dispersion along rows or columns ?) and changing the name of the subdirectory "database": run "epar echelle" and set the field *dispaxis* to 1 (mandatory) and the field *database* to the name you want to assign to the database subdirectory. **In the following, we will always assume that the subdirectory name is "database" (recommended).**

Setting the number of apertures to be found. Run "epar apfind" and set the following fields as shown:

(nfind =	196)	Number of apertures to be found automatically
(minsep =	3.)	Minimum separation between spectra
(maxsep =	52.)	Maximum separation between spectra

Setting the parameters for background subtraction. Run "epar apdefault" and set the following fields as shown:

(lower = -2.)	Lower aperture limit relative to center	
(upper = 2.)	Upper aperture limit relative to center	
...	...	...
(b_funct =	chebyshev)	Background function
(b_order =	2)	Background function order
...	...	...
(b_naver =	-100)	Background average or median
(b_niter =	3)	Background rejection iterations
(b_low =	1.)	Background lower rejection sigma
(b_high =	1.)	Background upper rejection sigma

Setting the aperture resize parameters. If *resize* is set to yes, the corresponding aperture boundaries along each column will be defined for each order by the pixels showing a number of counts equal to 10 % of the corresponding peak. However, we do not use this algorithm, so run "epar apresize" and set the following fields as shown below:

(resize =	no)	Resize apertures?
...	...	...
(nsum =	1)	Number of dispersion lines to sum or median
...	...	...
(peak =	yes)	Is ylevel a fraction of the peak?

Setting the parameter for the aperture editor. Run "epar apedit" and set the following fields as shown:

(nsum =	10)	Number of dispersion lines to sum or median
(width =	4.)	Profile centering width
(radius =	4.)	Profile centering radius

Setting the parameters for aperture tracing . Run "epar aptrace" and set the following fields as shown:

(line =	1000)	Starting dispersion line
(nsum =	10)	Number of dispersion lines to sum
(step =	10)	Tracing step
...	...	...
(functio=	spline3)	Trace fitting function
(order =	3)	Trace fitting function order

3.10 How to find traces

Following Sect. 3.7, you will have obtained a final, average flat-field image. Hopefully, you will have one per night. Flat-field images have by far the highest signal-to-noise ratio of all and none of the orders is depressed, as it happens to science frames due to the strong atmospheric absorption between the YJHK bands. Since, unlike the telescope, the spectrograph does not move at all during an observation run, flat-field images are the best choice to fit polynomials to the signal tracks with accuracy. These polynomials define the traces associated to each track. Usually, traces can be easily retrieved from any flat-field image. We ran across few cases, however, where a high level of scattered light, following a smooth distribution which peaks at the centre of the detector (see Fig 10), can drive small spurious peaks between the central orders to exhibit more counts (resulting from peak plus scattered component) than the peaks at the highest order. These spurious peaks are therefore counted as real signals and this leads to a wrong aperture numbering. **This occurrence is extremely rare, nevertheless we have developed a simple script which removes the scattered component before trace finding.** This task is called GIANO_FIND_TRACE and combines 1D fits (along image columns), obtained with the IRAF task *fit1d*, and trace detection performed by the task APFIND. The output of GIANO_FIND_TRACE is needed as a reference file (storing the trace information) by APFLATTEN (Sect. 3.11).

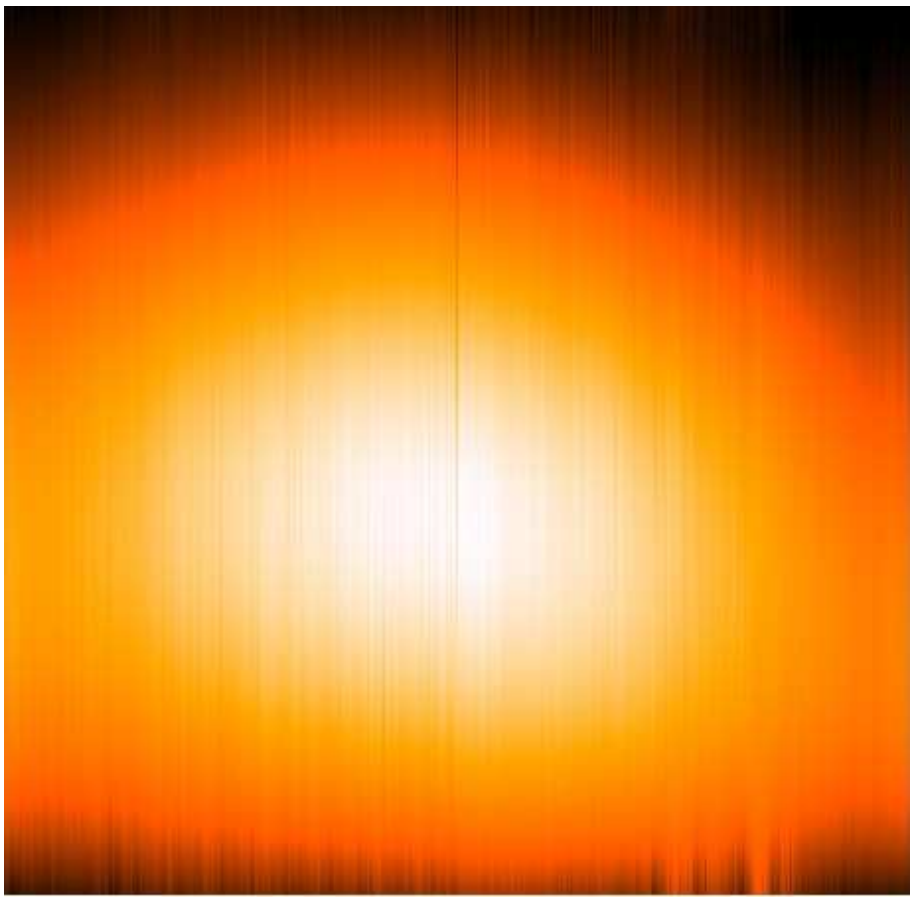

Figure 10: Map of scattered light over the GIANO detector area obtained by a simple spline3 fit of order 2 along columns (after discarding the signal inside the orders).

Fig. 11 explains how to set the parameters (with epar) of GIANO_FIND_TRACE. Set the first field to the name of your final flat-field frame. You can also set *save_sc* to yes, so that the derived map of scattered light can be saved in a fits file (whose name is output on the cl window by the task in run time). This can be then compared with Fig 10 for consistency. You will now be able to start GIANO_FIND_TRACE. At the beginning, it will take a few minutes to construct the scattered light map. Then, you will be asked some questions through the prompt, answer yes to all of them. You will have the graphic window opened and an interactive graphic session started. Check aperture locations and widths; if you want you can change them interactively see appendix A. However, we do not advice changing widths, just check that each aperture is centred on a track peak and that all apertures follow the correct numbering scheme. The correct aperture numbering is shown in Figs. 20, 21, and 22 (i. e., all four sub-tracks inside all orders but the highest one must be labelled and numbered, no aperture must be labelled between orders. **Note that the current version of** GIANO_FIND_TRACE **erases the highest order, unused peaks from the image, so you will not be able to see them anymore in the graphic window**). When satisfied, press "q" with the graphic window active.

```
                                   I R A F
                        Image Reduction and Analysis Facility
PACKAGE = giano_tools
   TASK = giano_extract_trace

input_fr= []          flat_cmb  Frame to use as a reference to extract traces:
out_fram=     scatcor_flat_cmb  Output frame after scattered light removal:
save_sc =                   no  Save scattered light map?
interac_=                  yes  Fit traces interactively?
displine=                 1000  Dispersion line:
gnsum   =                   10  Number of dispersion lines to sum or median:
gminsep =                   3.  Minimum separation between spectra:
gmaxsep =                  52.  Maximum separation between spectra:
(mode   =                    q)

                                                          ESC-? for HELP
```

Figure 11: Parameters to be set before running GIANO_FIND_TRACE.

GIANO_FIND_TRACE problem solving Check that all four apertures in every four-track group have been found and correctly numbered. The apertures must be numbered from 1 to 196 (i. e., 4×49).

Most of the problems we have found were caused by frames that had not been cleaned from bad-pixels appropriately and so exhibited spikes that interfered with the peak finding algorithm. Should any peaks in any order group not happen to be marked, or should some peaks other than the real ones happen to be marked, try first to change the parameter *displine* (but keep it in the range 900–1010). This will change the column searched for by the trace-finding algorithm and fix the problem if it is caused by a bad pixel. Should this not fix the problem, try changing *gminsep* (keep it in the range 3–6).

Sometimes we had all relevant peaks detected, but incorrectly numbered (e.g., 1–2–3–4, then 6–7–8–9, 5 missing). This was caused by spectral profile variations that resulted in slightly larger separations between the top track of an order group and the bottom one of the next order group than set by default in parameter *gmaxsep*. Try setting *gmaxsep* to a slightly larger value.

3.11 Using APFLATTEN to fit traces and construct a 2D flat-field frame

REMARK The next step is critical, not so much in order to obtain a reliable flat-field map of the detector, but especially because APFLATTEN will determine the traces that will then be used to extract the spectra from any other frame.

Having found the positions, along the reference column, of the traces with GIANO_FIND_TRACE, you have to derive a polynomial curve fitting the corresponding signal across the detector by using APFLATTEN. By default, the width of the apertures centred on the traces has been set in Sect. 3.9 (via APDEFAULT) to be 4 pixel in the direction orthogonal to the dispersion axis. APFLATTEN will allow you to construct a 2D flat-field image where the normalised values are only set inside the apertures, while the pixels outside them are set to 1. As explained in [6], this preserves the noise distribution all over a flat-field corrected frame, as needed by any bad-pixel removal algorithm based on a knowledge of the noise distribution. This is actually pointless in the NIR: NIR frames usually undergo several modifications and do not maintain the original information on the noise. So, we recommend that you do not use

any of these algorithms. In addition, a 2D flat is useful only if some spatial information is still present inside the apertures. Using fibres as GIANO does, such information is erased. Anyway, hereby we will show how to apply the 2D flat correction as an alternative to the 1D flat-field correction (see Sect. 3.6). **Furthermore, you can take advantage of the spectrum fit session to check that all apertures are correctly retrieved, by comparing the extracted 1D spectra to those shown in Figs. 27, 28, and 29**.

Summary of APFLATTEN actions: First, based on the aperture locations found by GIANO_FIND_TRACE, it runs an algorithm which sums all image columns *nsum* by *nsum* and finds all peaks along every *step* columns following the signals along the various orders. Secondly, it fits a polynomial curve to each string of peaks corresponding to a given order sub-track. It displays the fit results on the graphic window, and you can control this step interactively, as well. Finally, it extracts a spectrum from each trace and fits a polynomial to it. Again, you can review the fit results interactively. **These fits are used inside each trace to normalise the signal.** If you want the spectrograph response to be also removed by the flat, just use a polynomial of order 1 (see Sect. 3.16 and Appendix C). In principle, the wavelength range of each order is narrow enough so that the spectral density of a continuum source does not change appreciably inside.

To begin with, have a look at Fig. 12, which lists how to set the parameters (with epar) for APFLATTEN. Only five field settings have to be adapted: *input* (set it to the name given to the final combined flat-field frame to normalise), *output* (set it to the name you want to give to the normalised flat-field frame), and *referen* (set it to the output file of GIANO_FIND_TRACE). If you want to use a polynomial of order 1 (as said above), also set *functio* to chebyshev and *order* to 1.

All other fields should be set as in figure. We note that, being the field *clean* set to "no", there is no need to set the fields *saturat*, *readnoise*, and *gain*. In addition, the imaged GIANO orders are actually 49, but split in four traces each (two per fibre). So, APFLATTEN must be tricked into looking for 196 orders. At this stage, this is irrelevant.

You are now able to start APFLATTEN. Answer yes to all questions that are asked at the cl prompt. You will have the graphic window opened and an interactive graphic session started. In the following, the session steps are summarised. Each one is further detailed in the appendix. Just follow the appendix descriptions.

1. You can skip this step by pressing "q" with the graphic window active, you just did it with GIANO_FIND_TRACE. Alternatively, you can change aperture locations and widths interactively (see appendix A). The correct aperture numbering is shown in Figs. 20, 21, and 22 (i. e., all four sub-tracks of all orders but the last one labelled, no aperture labelled between orders)). When satisfied, press "q" with the graphic window active.

2. Check the trace fits (one per sub-track, i. e. four per order), you can change the parameters interactively (see appendix B). An example of correct fit for most of the 196 traces is shown in Fig. 24. Some of the highest number apertures exhibit a little problem like that shown in Fig. 25. Orders 48–49 will look like Fig. 26. In both cases, just delete the deviant points and redo the fit as explained in Appendix B. When done, press "q" with the graphic window active.

3. Check the spectrum fits (one per trace), you can change the parameters interactively (see appendix C). Examples of correct fits (and correctly extracted spectra) are shown in Fig. 27 (lower orders), Fig. 28 (central orders), and Fig. 29 (higher orders). Note that the signal peak shifts in position towards the right edge in moving from the lowest to the highest orders. When done, press "q" with the graphic window active.

```
                                        I R A F
                                Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = apflatten

input   =            flat_cmb  List of images to flatten
output  =       flat_cmb_norm  List of output flatten images
(apertur=               1-196) Apertures
(referen=    scatcor_flat_cmb) List of reference images

(interac=                 yes) Run task interactively?
(find   =                  no) Find apertures?
(recente=                  no) Recenter apertures?
(resize =                  no) Resize apertures?
(edit   =                 yes) Edit apertures?
(trace  =                 yes) Trace apertures?
(fittrac=                 yes) Fit traced points interactively?
(flatten=                 yes) Flatten spectra?
(fitspec=                 yes) Fit normalization spectra interactively?

(line   =                1000) Dispersion line
(nsum   =                  10) Number of dispersion lines to sum or median
(thresho=                 10.) Threshold for flattening spectra

(pfit   =               fit1d) Profile fitting type (fit1d|fit2d)
(clean  =                  no) Detect and replace bad pixels?
(saturat=               INDEF) Saturation level
(readnoi=                  0.) Read out noise sigma (photons)
(gain   =                  1.) Photon gain (photons/data number)
(lsigma =                  4.) Lower rejection threshold
(usigma =                  4.) Upper rejection threshold

(functio=             spline3) Fitting function for normalization spectra
(order  =                   5) Fitting function order
(sample =                   *) Sample regions
(naverag=                   1) Average or median
(niterat=                   3) Number of rejection iterations
(low_rej=                  3.) Lower rejection sigma
(high_re=                  3.) High upper rejection sigma
(grow   =                  0.) Rejection growing radius
(mode   =                  ql)

                                                                  ESC-? for HELP
```

Figure 12: Parameters to be set for APFLATTEN. **Note that the fields "input", "output", and "referen" have to be adapted (see text). If necessary, change also "functio" and "order".**

A normalised flat frame obtained from APFLATTEN (and displayed with DISPLAY) will look like the one in Fig. 13. Note the perfectly flat areas between orders. This is why you need to use the same traces for normalised flat images and target 2D spectra, to make sure that all extracted signal is flat-field corrected. You can obtain a view along a column by using IMPLOT, which will look like that in Fig. 14.

3.12 Flat-fielding the targets in 2D, and final steps before spectrum extraction

FLAT-FIELD CORRECTION If you decide on flat-field correcting your spectra in 1D, skip this step and go on to the point FRAME SUBTRACTION below. 1D flatting will then be performed later as described in Sect. 3.15.

As for 2D flatting, following the steps outlined in the above section yields a frame that is hereby named flat_cmb_norm. This is the normalised 2D frame to use for flat-field correction. Now, list all dark-subtracted, bad-pixel cleaned comparison lamp spectra, and all bad-pixel cleaned science frames in a single text file (e.g., to-be-flatted.list). Make a new text file (e. g., call it flatted.list) and list, in the same order, the names you want to allot to the flat-field corrected frames. To flat-field correct them all, just type:

cl> imarith @to-be-flatted.list / flat_cmb_norm @flatted.list

This will divide (pixel-by-pixel) each frame listed in to-be-flatted.list by the normalised flat, assigning the corresponding name found in flatted.list to the output frame.

FRAME SUBTRACTION After you have corrected the science frames, subtract frame B from frame A for every AB cycles with the target alternatively in each of the fibres. This will remove any residual scattered and diffuse light contribution. Do this again using IMARITH, e. g.:

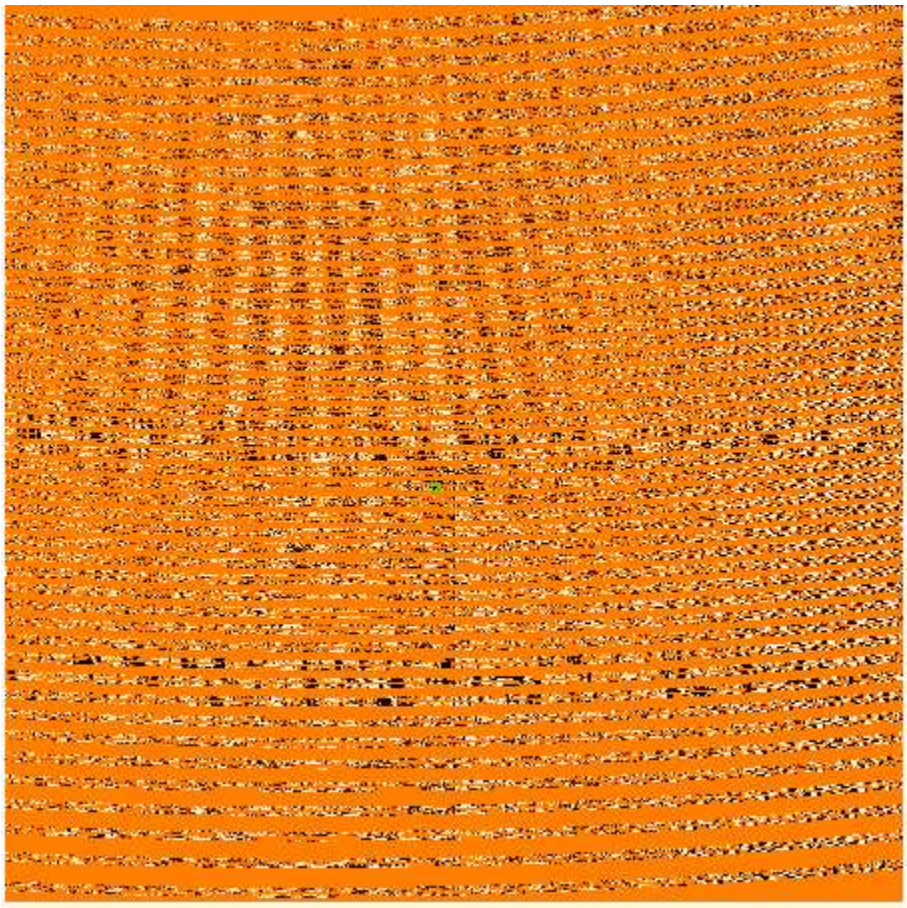

Figure 13: Normalised flat-field image obtained with APFLATTEN.

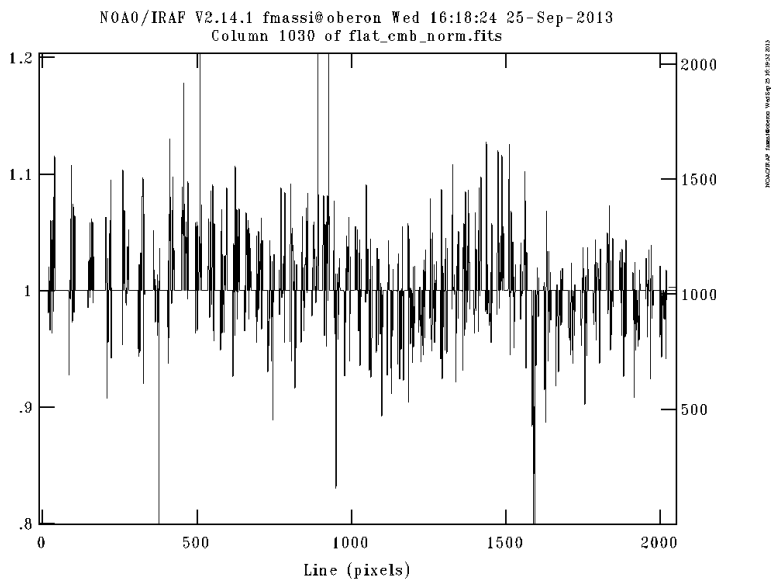

Figure 14: Cut along the central column of a normalised flat image obtained with APFLATTEN.

cl> imarith star1-lower-fibre - star1-upper-fibre star1_sub

Of course, this can be done for all frames simultaneously by using three list files (and an @ in front of the file names in the command line, as usual). The frame star1_sub.fits will have positive and negative peaks. The signal from the lower fibre will be positive (two tracks per order), and the signal from the upper fibre negative (two tracks per order). Thus, you need to multiply star1_sub.fits by −1 before extracting the spectra from the upper-fibre tracks (see Sect. 3.13). You might also have either pairs of on-source and off-source frames or single target frames, i. e. with the source spectrum in only one fibre. In the former case, subtract the off-source frames from the on-source ones, as explained above. In the latter case, you have no other choice but a dark-subtracted, flat-field corrected, background contaminated spectrum.

On the other hand, if you have more than one subtracted AB pair of frames from the same target, combine them together with IMCOMBINE before extracting the spectra. Alternatively, you could first extract the spectra from every single AB frame and then, if necessary, combine the spectra from the same targets with SARITH as explained in Sect. 3.17. As already explained, list them all in a text file (e. g., target1_A-B.list) and run IMCOMBINE:

cl> imcombine @target1_A-B.list.list target1_A-B_comb combine=average

this will yield the file target1_A-B_comb.fits as the average of all frames listed in target1_A-B.list.list.

3.13 Cloning frames

As explained above, we have found 196 traces, but the orders are actually only 49. In fact, for each order there is a bottom trace, a mid-lower one, a mid-upper one and a topmost one. We will need a different frame for each of the 4 sets of traces to extract the 1D spectra, i. e., we need to make four copies of each frame. In addition, all AB subtracted frames will have to be multiplied by −1 before

extracting 1D spectra from the two upper (or lower) trace pairs per order. This may seem redundant, but by allowing the four traces of each order to be separately extracted, they will be independently wavelength-calibrated and this will also correct for the slit image not being perfectly vertical within the traces.

This "frame cloning" can be done in one shot by using the task COPY_FILE of GIANO_TOOLS. First, list all subtracted files in a text file (e. g., subtracted.list) and all frames with only positive traces (which do not need multiplication by −1, e. g., those of the calibration lamps) in another (e. g., lamp.list). Then run "epar copy_file" and set the fields as below:

```
PACKAGE = giano_tools
  TASK = copy_file
     anysub =    yes              Are there any subtracted (A–B) frames?
     inlist =    subtracted.list  File listing subtracted (A–B) frames:
     inmark =    1                Positive peaks are at the lower (1) or upper (2) two tracks?:
     nonodd =    lamp.list        File listing frames with positive peaks at all tracks (calibration lamps, etc.):
    (mode = q)
```

If you only have frames with positive traces (i. e., not subtracted), just set *anysub* to no and disregard *inlist* and *inmark*. Only set *inlamp* to the name of the list of frames. If you also have AB subtracted files, set *anysub* to yes and *inlist* to the name of the list of subtracted frames. If you only have subtracted files, set *anysub* to yes, set *inlist* to the name of the list of subtracted frames, and set *inlamp* to an empty string. Then set *inmark* according to whether the traces with positive counts in each order are the lower two (set it to 1) or the upper two (set it to 2). Finally, run the task.

The task makes four copies of each file found in the two lists. Two of the copies of the AB subtracted frames are also multiplied by −1. The four copies will have a name composed of that of the original frame plus the strings: _lowest, _middown, _midup, _topmost (e. g., target1.fits will be copied into target1_lowest.fits, target1_middown.fits, target1_midup.fits, and target1_topmost.fits). Each of the four copies will be used with the corresponding trace reference frame obtained as explained in Sect. 3.14.

3.14 Spectrum extraction

The task that you will use for extracting spectra is DOECSLIT. However, you will first need to run APEDIT to associate the traces already found through GIANO_FIND_TRACE and APFLATTEN to each cloned frame. As a consequence, DOECSLIT will skip the trace finding session, but will wavelength-calibrate the spectra after extracting them and before saving them in fits format. Flux calibration will not be performed with DOECSLIT.

Remember having tricked APFLATTEN in considering any suitable signal track in a frame (196 in total) as a single spectral order. So you will have to "split" the traces found and re-arrange them in the correct order. To this purpose, you can use the GIANO_TOOLS task SPLIT_FILE. The task operates on the non-normalised flat-field frame, for which trace definitions have been saved by APFLATTEN, and makes four new fits files (and a few associated files in the subdirectory database), the trace parameters for the actual 49 orders of each fibre. It also set the correct "sky" windows around each trace to allow background subtraction, if requested. Run "epar split_file" and set the fields as below:

```
PACKAGE = giano_tools
  TASK = split_file
     imtrace =   flat_cmb   Reference image for traces:
     numf =      4          Number of traces per order (2,4)
     dir =       database   Directory holding the aperture files (default database)
     mode =      ql)
```

remember to set the field *imtrace* to the name of the flat-field frame version before normalisation (i. e., the input to APFLATTEN), and the field *dir* to the name you have assigned to the subdirectory database (of course, only if you have changed it and do not use the default one). The field *numf* must

be set to 4. The other possible value (2) was only used for the very first GIANO observations (which adopted no image slicers).

After the task has run, this will output: a reference frame for the traces from the lowest sub-track of every order group (input name with the string "_lowest" added at the end, e. g. flat_cmb_lowest.fits), a reference frame for the traces from the mid-lower sub-tracks (input name with the string "_middown" added at the end, e. g. flat_cmb_middown.fits), a reference frame for the traces from the mid-upper sub-tracks (input name with the string "_midup" added at the end, e. g. flat_cmb_midup.fits), and a reference frame for the traces from the topmost sub-tracks (input name with the string "_topmost" added at the end, e. g., flat_cmb_topmost.fits). In addition, the trace parameters can now be found in four new files held in the subdirectory database, with the same names as the corresponding new fits file plus an "ap" string in front (and without extension); e. g., apflat_cmb_lowest, apflat_cmb_middown, apflat_cmb_midup, and apflat_cmb_topmost.

It is mandatory that you also make four copies of each flat-field corrected calibration-lamp frame, following Sect. 3.13. In fact, during the wavelength-calibration step, DOECSLIT will assign a set of traces to the lamp frames. So, you need a set of lamp frames per each of the four sub-track sets.

Including the strings added by COPY_FILE and SPLIT_FILE to the frame names, list all the science frames with the "lowest" label in a new file (e.g., lowest.list), all the science frames with the "middown" label in a second file (e.g., middown.list), all the science frames with the "midup" label in a third file (e.g., midup.list), and all the science frames with the "topmost" label in a fourth file (e.g., topmost.list). Do the same for the calibration lamp frames (e.g., lowest_lamp.list, middown_lamp.list, midup_lamp.list, and topmost_lamp.list).

Before extracting the 1D spectra, you need to associate the correct traces to each set of frames. This will be done by using the task APEDIT. First, associate the traces of the lowest sub-tracks of each order to the "lowest" frame set. The main steps are listed below and further described in the appendices.

1. Run "epar apedit" and set the fields as in Fig. 15. Note that *input* and *referen* need to be adapted.

2. Run APEDIT and follow the instructions given in appendix A. Just check that the aperture locations have been set correctly for each frame.

Repeat the two steps for the other three track groups (middown, midup, and topmost), so that every frame will have its trace set associated.

Use the task DOECSLIT to extract and wavelength-calibrate the 1D spectra. First, extract the 1D spectra from the 49 lowest sub-tracks. The main steps are listed below and further described in the appendices.

1. Run "epar doecslit" and set the fields as in Fig. 16. **the fields to adapt to your own setting are: objects, apref, arcs, readnoi, gain, datamax, and backgrou.** As for the first three fields, input the name you have assigned to the file listing the "lowest" science frames, the reference frame for the "lowest" traces (obtained from SPLIT_FILE), and the name you have assigned to the file listing the "lowest" calibration lamp frames. Compute the values to input into fields *readnoi* (effective frame readout noise) and *gain* (effective frame gain) as explained in appendix E. Compute the maximum data value to input into field *datamax* from Table 1. If you do not want to carry out any background subtraction, set *backgrou* to none.

2. Note that the field *sparams* in Fig. 16 is empty. This is because SPARAMS does not require a value, it is just a link to another parameter file. You have to access it by entering ":e" (followed by return) into the field. Alternatively, you can exit the parameter editor and type "epar sparams" to access the file. Edit SPARAMS and set ONLY the field shown in Fig. 17 as shown in figure, keeping the default value for the others. There is no need to change any of the fields not shown

```
                                   I R A F
                        Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = apedit

input   =         @lowest.list  List of input images to edit
(apertur=                 1-49) Apertures
(referen= flat_cmb_lowest.fits) Reference images

(interac=                  yes) Run task interactively?
(find   =                   no) Find apertures?
(recente=                   no) Recenter apertures?
(resize =                   no) Resize apertures?
(edit   =                  yes) Edit apertures?

(line   =                 INDEF) Dispersion line
(nsum   =                    10) Number of dispersion lines to sum or median
(width  =                    4.) Profile centering width
(radius =                    4.) Profile centering radius
(thresho=                    5.) Detection threshold for profile centering
(mode   = ▯                  ql)

                                                          ESC-? for HELP
```

Figure 15: Parameters to be set for APEDIT. **Note that the fields "input" and "referen" have to be adapted (see text).**

in the figure. Only parameters in the "AUTOMATIC ARC ASSIGNMENT PARAMETERS" section can be adapted in alternative ways; see appendix F.

3. Run DOECSLIT and follow the instructions given in appendix G. At the end, the task SPLOT will be invoked by DOECSLIT and you will be shown the final spectra. See appendix D on how to use SPLOT, the 1D spectrum editor.

4. When DOECSLIT has finished, the extracted spectra will be saved in fits files having the same name as the input file plus the string ".ec." at the end (e. g., input file "target1_lowest.fits", output file "target1_lowest.ec.fits").

Let us now turn to extract 1D spectra from the "mid-lower" trace set. The main steps are the same as for the "lowest" set, with few small changes. Note that you should also go through the wavelength assignment stage for all remaining three sets of traces, which may be quite boring. Thus, this step can be made faster by using the ECHELLE task ECREIDENTIFY. Once you have extracted the "lowest" set of spectra with DOECSLIT, you can use ECREIDENTIFY to automatically find a new wavelength solution for the remaining trace sets. To this purpose, we have developed a GIANO_TOOLS script, called GIANO_REIDENTIFY, which makes this step as easy as possible. DOECSLIT will have extracted at least one of the "lowest" lamp spectra and saved it in a file ending in ".ec.fits". E. g., the spectra extracted from the image "lamp_1_lowest.fits" are saved in "lamp_1_lowest.ec.fits". To run GIANO_REIDENTIFY, you need one of the lamp "ec.fits" files as a reference. In addition you need a text file listing the lamp images to calibrate (e. g., middown_lamp.list; **These are images, not spectra, so do not use a label "ec" in the file names!**). Note that the lamp frames input to GIANO_REIDENTIFY must have the corresponding aperture reference image set (e. g., flat_cmb_middown.fits for the "middown" set). Run "epar giano_reidentify" and input the required parameters as follows:

PACKAGE = giano_tools
TASK = giano_reidentify

refimage =	lamp_3_lowest.ec	Wavelength-calibrated lamp reference spectrum:
imlist =	middown_lamp.list	File list with lamp spectra to extract and calibrate:
referen =	flat_cmb_middown	Aperture reference image:
database =	database	Database

```
                                   I R A F
                        Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = doecslit

objects =        @lowest.list  List of object spectra
(apref  = flat_cmb_lowest.fits) Aperture reference spectrum
(arcs   =    @lowest_lamp.list) List of arc spectra
(arctabl=                     ) Arc assignment table (optional)
(standar=                     ) List of standard star spectra

(readnoi=                  8.5) Read out noise sigma (photons)
(gain   =                  2.2) Photon gain (photons/data number)
(datamax=                INDEF) Max data value / cosmic ray threshold
(norders=                   49) Number of orders
(width  =                   4.) Width of profiles (pixels)

(dispcor=                  yes) Dispersion correct spectra?
(extcor =                   no) Extinction correct spectra?
(fluxcal=                   no) Flux calibrate spectra?
(resize =                   no) Resize object apertures?
(clean  =                   no) Detect and replace bad pixels?
(trace  =                   no) Trace object spectra?
(backgro=                  fit) Background to subtract
(splot  =                  yes) Plot the final spectra?
(redo   = ▯                 no) Redo operations if previously done?
(update =                   no) Update spectra if cal data changes?
(quicklo=                   no) Approximate quicklook reductions?
(batch  =                  yes) Extract objects in batch?
(listonl=                   no) List steps but don't process?

(sparams=                     ) Algorithm parameters
(mode   =                   ql)

                                                    ESC-? for HELP
```

Figure 16: Parameters to be set for DOECSLIT. **Note that the fields objects, apref, arcs, readnoi, gain, datamax, and backgrou have to be adapted (see text).**

(mode = q)

Now run the task and check the output information (for each lamp: number of lines reidentified, number of fit reidentified lines, r.m.s., etc.). This will resemble the following:

ECREIDENTIFY: NOAO/IRAF V2.14.1 fmassi@oberon Tue 14:46:24 24-Sep-2013
Reference image = lamp_3_lowest.ec, Refit = yes

Image	Found	Fit	Pix Shift	User Shift	Z Shift	RMS
test.ec	288/290	269/288	−0.387	0.319	4.17E−6	0.00137

In particular, note that 288 out of 290 lines have been retrieved in the new spectrum, of these 269 have been fitted. The r.m.s. is quite low (0.00137) as well. Values like these are ok.

After GIANO_REIDENTIFY has run successfully, you can run DOECSLIT repeating the same steps as for the "lowest" set of tracks:

1. Run "epar doecslit", keep the fields set as in Fig. 16, **but update the fields: objects, apref,and arcs.** Input the name you have assigned to the file listing the "mid-lower" target frames, the reference frame for the "mid-lower" traces, and the name you have assigned to the file listing the "mid-lower" calibration lamp frames.

2. You do not have to change any setting in SPARAMS. Keep them as explained above for the "lowest" case.

3. Run DOECSLIT and follow the instructions given in appendix G. At the end, the task SPLOT will be invoked by DOECSLIT and you will be shown the final spectra. See appendix D to use SPLOT.

4. When DOECSLIT has finished, the extracted spectra will be saved in fits files with the same name as that of the input file plus the string ".ec.".

Rerun GIANO_REIDENTIFY and repeat the above steps (1 through 4) for the mid-upper and topmost traces, as well.

```
                                -- BACKGROUND AND SCATTERED LIGHT PARAMETERS --
(b_funct=             legendre) Background function
(b_order=                    2) Background function order
(b_naver=                 -100) Background average or median
(b_niter=                    3) Background rejection iterations
(b_low  =                   1.) Background lower rejection sigma
(b_high =                   1.) Background upper rejection sigma
(buffer =                   2.) Buffer distance from apertures
(apscat1=                     ) Fitting parameters across the dispersion
(apscat2=                     ) Fitting parameters along the dispersion

                                -- APERTURE EXTRACTION PARAMETERS --
(weights=                 none) Extraction weights (none|variance)
(pfit   =                fit1d) Profile fitting algorithm (fit1d|fit2d)
(lsigma =                   3.) Lower rejection threshold
(usigma =                   3.) Upper rejection threshold

                                -- ARC DISPERSION FUNCTION PARAMETERS --
(thresho=                  10.) Minimum line contrast threshold
(coordli= gianot$U_Ne_Ar_lines_list_cl.dat) Line list
(match  =                  0.1) Line list matching limit in Angstroms
(fwidth =                   6.) Arc line widths in pixels
(cradius=                  10.) Centering radius in pixels
(i_funct=             legendre) Echelle coordinate function
(i_xorde=                    4) Order of coordinate function along dispersion
(i_yorde=                    4) Order of coordinate function across dispersion
(i_niter=                    3) Rejection iterations
(i_low  =                   3.) Lower rejection sigma
(i_high =                   3.) Upper rejection sigma
(refit  =                  yes) Refit coordinate function when reidentifying

                                -- AUTOMATIC ARC ASSIGNMENT PARAMETERS --
(select =              average) Selection method for reference spectra
(sort   =                   jd) Sort key
(group  =                  ljd) Group key
(time   =                   no) Is sort key a time?
(timewra=                  17.) Time wrap point for time sorting

                                -- DISPERSION CORRECTION PARAMETERS --
(lineari=                  yes) Linearize (interpolate) spectra?
(log    =                   no) Logarithmic wavelength scale?
(flux   =                  yes) Conserve flux?
```

Figure 17: Parameters to set in SPARAMS. **Only for "AUTOMATIC ARC ASSIGNMENT PARAMETERS" you might want to use a different set-up (see text).**

giano_reidentify as a shortcut provided you have run DOECSLIT once and have reference images for at least one trace set, you can use GIANO_REIDENTIFY to wavelength-calibrating every other lamp set taken in any other date. E. g., you can use the reference lamp frame already provided with the GIANO_TOOLS package. Just copy cl_jul_UNe_300_ds_lowest.ec.fits to your working directory and eccl_29_jul_UNe_300_ds_lowest.ec to its subdirectory database, and follow Appendix G to see how to use DOECSLIT when a wavelength solution is already available.

3.15 Flat-field correcting the spectra in 1D

If you have decided on performing a flat-field correction on the extracted spectra (1D) rather than on the corresponding 2D images, now it is time to do it. To make things easier, GIANO_TOOLS provides still another task that will enable you to both construct the flat-field 1D spectra and apply the correction to the target 1D spectra. It is based on the ECHELLE task CONTINUUM (see Sect. 3.16) and its name is "flat_1D". As usual, run *epar flat_1D* and set the parameters as follows. Be aware that FLAT_1D requires that you have followed all the reduction steps explained so far, in particular having already used SPLIT_FILE and COPY_FILE. It needs the same information input to SPLIT_FILE and COPY_FILE, but actually uses their output files.

iinput the final reduced flat-field 2D image used as input to SPLIT_FILE. The task will translate the file name into the corresponding "lowest", "middown", etc., names.
jinput either a file listing all target frames to flat-field correct, or a single fits file. It is mandatory that you should use the file names input to COPY_FILE, the task will translate the file names into the corresponding "lowest.ec", "middown.ec", etc., names.
lampname The calibration lamp frame or a file listing the calibration lamp frames. Again, use the file names input to COPY_FILE, the task will translate the file names into the corresponding "lowest.ec", "middown.ec", etc., names.
trac1d_f set it to yes if you want to use the same traces associated to the flat-field frame input to iinput (default). If another frame is to be used, set it to no.
trac1_d Reference frame to use for aperture definition if trac1d_f is set to no. Again, omit "_lowest", "_middown", etc., in the name, but the reference frame needs having passed the steps for aperture definition (i. e., it needs the associated files in the subdirectory database and the labels "_lowest", "_middown", etc., added to its name to denote the actual reference frame names).
inter1d set it to yes if you want to follow interactively the fits to the continuum lamp spectra.
order1d_ set it to yes if you want to normalise the flat-field 1D spectra to a constant value (to remove the spectrograph response) or to no to normalise the flat to a fitted polynomial.
dir1d set it to the name of the subdirectory where the reduction information is stored (default is database).
del1d set it to no if you want to save the normalised flat-field 1D spectra.

By setting *del1d* to no, four fits files with the normalised flat-field 1D spectra will be saved in your working directory: namely, tmp_flat_corr_lowest_nrm.ec.fits, tmp_flat_corr_middown_nrm.ec.fits, tmp_flat_corr_midup_nrm.ec.fits, and tmp_flat_corr_topmost_nrm.ec.fits. Take care to change their names before any further run of flat_1D. If you only want to construct 1D normalised spectra, set *jinput* to a void string and *del1d* to no. Be aware that the flat-field correction is performed trace by trace Afterwards, each flatted 1D spectrum will be saved and assigned the name from the list input to *jinput*, plus the string "_lowest_fl.ec.fits", "_middown_fl.ec.fits", etc., at the end.

3.16 Normalising the extracted 1D spectra to the continuum

You might want to have your spectra normalised to the continuum, i. e. to have the continuum level set to a constant value. You can do it by using the task CONTINUUM. However, it is often difficult to identify the continuum level in a spectrum exhibiting lots of absorption lines. Thus, we suggest two approximate ways of normalising your spectra. Taking into account that the flat-field spectra are those with the best signal-to-noise ratio, since all orders span a limited range in wavelength, the spectral shape in the flat-field apertures is mostly due to the spectrograph efficiency.

The simplest way to operate is by fitting a Legendre polynomial of order 1 (i. e., a horizontal straight line) when constructing a normalised 2D flat (Sect. 3.11) or 1D flat (Sect. 3.15). After flat-field correction, your extracted spectra will already be approximately normalised to the continuum.

Alternatively, if you fit a higher order polynomial to normalise the continuum lamp 1D spectra, you will have to perform a new fit to the non-normalised flat-field frames. The drawback is that you risk adding small oscillations to the extracted 1D spectra in both steps. Try always to keep the order of the fitted polynomial as low as possible. To perform (and save) the fit, we will use the task "continuum". First, make a copy of flat_cmb_lowest.fits, flat_cmb_middown.fits, flat_cmb_midup.fits and flat_cmb_topmost.fits (e. g., call them lowest_continuum.fits, middown_continuum.fits, midup_continuum.fits, and topmost_ continuum.fits). Extract their 1D spectra with DOECSLIT, using the appropriate setting as explained in Sect. 3.14. If you have already wavelength calibrated all traces, this step will be skipped by the task. When you start DOECSLIT, you will have to confirm all the questions you will be asked (no calibration session will be opened). Then, you need to fit all the 49 1D spectra from all the four

```
                                   I R A F
                         Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = continuum

input   = lowest_continuum.ec.fits  Input images
output  = lowest_transmission.ec.fits  Output images
(lines  =                    *) Image lines to be fit
(bands  =                    1) Image bands to be fit
(type   =                  fit) Type of output
(replace=                   no) Replace rejected points by fit?
(wavesca=                  yes) Scale the X axis with wavelength?
(logscal=                   no) Take the log (base 10) of both axes?
(overrid=                   no) Override previously fit lines?
(listonl=                   no) List fit but don't modify any images?
(logfile=              logfile) List of log files
(interac=                  yes) Set fitting parameters interactively?
(sample =                    *) Sample points to use in fit
(naverag=                    1) Number of points in sample averaging
(functio=             spline3) Fitting function
(order  =                    5) Order of fitting function
(low_rej=                   3.) Low rejection in sigma of fit
(high_re=                   3.) High rejection in sigma of fit
(niterat=                    3) Number of rejection iterations
(grow   =                   0.) Rejection growing radius in pixels
(markrej=                  yes) Mark rejected points?
(graphic=             stdgraph) Graphics output device
(cursor =                     ) Graphics cursor input
ask     = ▯                yes
(mode   =                   ql)
```

Figure 18: Parameters to be set in continuum.

trace sets (in this case, lowest_continuum.ec.fits, middown_continuum.fits, midup_continuum.fits, and topmost_ continuum.ec.fits). Run "epar continuum" and set the parameters as in Fig. 18. Note that the field *type* is set to fit in order to save the fitted polynomial in the output frame. Set the fields *functio* and *order* to the same parameters as used in APFLATTEN. Run CONTINUUM and repeat for the other three fibre settings.

To normalise the target spectra, we will use the task SARITH, which is the analogue of IMARITH for spectra; SARITH handles spectra and deals automatically with issues such as differences in sampling intervals and wavelength ranges. List the fits files with the extracted 1D spectra for each of the four trace sets in a text file (e. g., let us call them to_be_normalised_lowest.list, to_be_normalised_middown.list, to_be_normalised_midup.list, and to_be_normalised_topmost.list). Make four other lists with the names to assign to the normalised 1D spectra (e. g., normalised_lowest.list, normalised_middown.list, normalised_midup.list, and normalised_topmost.list). Then run SARITH on all pairs of files:

cl> sarith @to_be_normalised_lowest.list / lowest_continuum.ec.fits @normalised_lowest.list
cl> sarith @to_be_normalised_middown.list / middown_continuum.ec.fits @normalised_middown.list
cl> sarith @to_be_normalised_midup.list / midup_continuum.ec.fits @normalised_midup.list
cl> sarith @to_be_normalised_topmost.list / topmost_continuum.ec.fits @normalised_topmost.list

3.17 Combining the extracted 1D spectra together

At this point, you will have four different 1D spectra of every target (each per sub-track group). You need to combine them together to increase the signal-to-noise ratio. This can easily be done using the task SCOMBINE, which is the analogue of IMCOMBINE for spectra; SCOMBINE handles spectra and deals with issues such as differences in sampling intervals and wavelength ranges. **Actually, we found more suitable using SARITH, which apparently does a better job with handling wavelengths.** Of course, SARITH only allows operations on pairs of spectra, so it needs to be repeated several times when averaging a number of spectra.

First, have a look at the 1D spectra to sum up with SPLOT (see appendix D) to identify the maximum and minimum (in principle, the minimum should be zero) signal level throughout the sample. Be also aware that the two fibres have slightly different efficiencies.

Then, list all spectra to be combined, in a text file (e. g., call it target1_ec.list). **Now you will have to use the fits files of the extracted 1D spectra (the "ec.fits" files)** obtained from DOECSLIT, or after normalisation. Run "epar scombine" and set the parameters as in Fig. 19. **The only fields you may want to set are lthresh and hthresh.** You need to put a value slightly below the lowest signal level you have identified into *lthresh*, and a value slightly above the highest signal level you have identified into *hthresh*. This will clean off a few annoying spikes. The parameter fields listed by the parameter editor lying below *lthresh* are not used in our case, so you can leave them untouched. As for *combine*, We always prefer using the average, **but see the remarks on the effective exposure time.**. And, of course, *output* should be set to the name you want to give to the output combined 1D spectrum. Finally, run SCOMBINE.

REMARK: EFFECTIVE INTEGRATION TIME OF COMBINED SPECTRA

Remember that the output of each fibre is split into two spots, both feeding the slit. This means that the exposure time WILL BE EXACTLY THAT OF THE SINGLE A or B FRAME after summing together the pair of 1D spectra corresponding to the same fibre. Consequently, if you add together all four 1D spectra extracted from an AB subtracted frame, the effective exposure time will be twice that of the single A or B frame. If you average together the four 1D spectra, the exposure time will be HALF that of the single A or B frame (but the effective gain will be larger). See Appendix E to compute effective exposure time, gain, and read-out noise in a number of cases.

3.18 Viewing the final 1D spectra

To look at your final extracted 1D spectra you can use SPLOT, as explained in appendix D. You can also use SPLOT to perform several operations on spectra (see the SPLOT help in IRAF). However, the simplest way to handle a 1D spectrum is translating it into a text file (a two-column list with wavelengths and counts). To do this, you can use the GIANO_TOOLS script called FITS2TXT, which is based on the iraf.onedspec task WSPECTEXT. We will explain how to use FITS2TXT first, and then how to use WSPECTEXT itself. Run "epar fits2txt", the editor page will look like that below:

```
PACKAGE = giano_tools
  TASK = fits2txt
     specin =      @spec.list   Input fits spectrum
     startnum =    yes          Start output from 1(yes) or 32(no)?
     sphed =       no           Include header?
    (mode = q)
```

The field *specin* has to be set either to the name of a single fits file or to a text file (remember to put "@" followed by the file name into the field) listing a number of 1D spectrum fits files. The task output consists of 49 new text files each with one spectral order, having the same name of the input fits file but ending in "_t*N*.txt" instead of ".fits". Note that *N* is a number from one (the lowest order, i. e. the largest wavelength) to 49 (the highest order, i. e. the smallest wavelength). If you want, you can number the text files starting from 32 (i. e., the real number of the lowest order) just setting *startnum* to *no*. If you want to have the fits header saved in the text files, as well, set *sphed* to *no*. For the wavelength ranges covered in each order, see Appendix H.

Alternatively, you can use WSPECTEXT by yourself. You first have to load the package:

cl> onedspec

Note that wspectext allows you to convert only one order of a spectrum at a time. E. g., to convert the first trace of a spectrum named target1.ec.fits into a text file named target1_n1.txt, you will have to set the wspectext parameters as follows (note the [*,1] at the end of the input file name):

```
PACKAGE = onedspec
  TASK = wspectext
```

input =	target1.ec.fits[*,1]	Input list of image spectra
output =	target1_n1.txt	Output list of text spectra
header =	no	Include header?
wformat =		Wavelength format
(mode = q)		

The good news is, you can make a script to convert all traces in one shot. Just write a text files with the following lines (one per aperture):

wspectext target1.ec.fits[*,1] target1_n1.txt header=no
wspectext target1.ec.fits[*,2] target1_n2.txt header=no
wspectext target1.ec.fits[*,3] target1_n3.txt header=no
...

Then run the script typing:
cl> cl < nomefile

Needless to say, there is also a task RSPECTEXT, which allows you to convert text spectra in fits files (see the help in IRAF). **After using wspectext, unload the package onedspec by typing "bye"** to avoid messing up the parameter files associated with the package ECHELLE.

```
                                   I R A F
                         Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = scombine

input   =     @target1_ec.list  List of input spectra
output  = target1_comb.ec.fits  List of output spectra
(noutput=                     ) List of output number combined spectra
(logfile=               STDOUT) Log file

(apertur=                 1-49) Apertures to combine
(group  =            apertures) Grouping option
(combine=             average) Type of combine operation
(reject =                none) Type of rejection

(first  =                  no) Use first spectrum for dispersion?
(w1     =               INDEF) Starting wavelength of output spectra
(w2     =               INDEF) Ending wavelength of output spectra
(dw     =               INDEF) Wavelength increment of output spectra
(nw     =               INDEF) Length of output spectra
(log    =                  no) Logarithmic increments?

(scale  =                 none) Image scaling
(zero   =                 none) Image zero point offset
(weight =                 none) Image weights
(sample =                     ) Wavelength sample regions for statistics

(lthresh=                INDEF) Lower threshold
(hthresh=                INDEF) Upper threshold
(nlow   =                    1) minmax: Number of low pixels to reject
(nhigh  =                    1) minmax: Number of high pixels to reject
(nkeep  =                    1) Minimum to keep (pos) or maximum to reject (neg)
(mclip  =                  yes) Use median in sigma clipping algorithms?
(lsigma =                   3.) Lower sigma clipping factor
(hsigma =                   3.) Upper sigma clipping factor
(rdnoise=                   0.) ccdclip: CCD readout noise (electrons)
(gain   =                   1.) ccdclip: CCD gain (electrons/DN)
(snoise =                   0.) ccdclip: Sensitivity noise (fraction)
(sigscal=                  0.1) Tolerance for sigma clipping scaling corrections
(pclip  =                 -0.5) pclip: Percentile clipping parameter
(grow   =                    0) Radius (pixels) for 1D neighbor rejection
(blank  =                   0.) Value if there are no pixels
(mode   =                   ql)

                                                          ESC-? for HELP
```

Figure 19: Parameters to be set in SCOMBINE. **Fields lthresh and hthresh need being adapted as explained in the text.**

4 Bibliography

[1] Oliva, E., et al. 2012, SPIE, 8446
[2] Origlia, L., Oliva, E., & Scuderi, S. GIANO cookbook for observers,
http://http://www.tng.iac.es/instruments/giano
[3] Origlia, L., & Oliva, E. GIANO cookbook for proposers,
http://http://www.tng.iac.es/instruments/giano
[4] Barnes, J. 1993, "A Beginner's Guide to Using IRAF",
http://iraf.noao.edu/docs/recommend.html
[5] Shames, P., & Tody, D. 1986, "A User's Introduction to the IRAF Command Language",
http://iraf.noao.edu/docs/recommend.html
[6] Wilmarth, D., & Barnes, J. 1994, "A Users Guide to Reducing Echelle Spectra with IRAF",
http://iraf.net/irafdocs/ech.pdf
[7] Oliva, E., et al. 2013, A&A 555, A78
[8] ESO Sky Model Calculator: http://www.eso.org/observing/etc/

Appendix A: Checking the apertures

After starting GIANO_FIND_TRACE, APFLATTEN, or DOECSLIT, you will be asked a few questions, always answer "yes". Then, the graphic window will open and the first thing you will be shown is a section of the frame along the central column, like in Fig. 20. You can see all orders as peaks and, on top of the panel, the aperture intervals marked by horizontal segments joining small vertical bars and labelled by numbers. You may want to check that the aperture intervals are correct, so you had better zoom in on parts of the image. In Fig. 21, the zoom-in shows the correct numbering: all four peaks corresponding to the sub-tracks of the same order must be labelled and numbered for every order (sometimes order 81 is also visible; it must not be labelled), and no aperture interval segment must fall in areas between subsequent orders. Figure 22 shows a zoom-in on the highest orders; order 81 is not marked, this is ok (sometimes even half of order 52 is also visible), and the four sub-tracks of order 80 are numbered from 193 to 196, this is ok as well. If not so, the aperture finding algorithm failed for some reason and you must check in depth.

Hereafter, a list of the relevant graphic commands, **always to input with the graphic window active**.

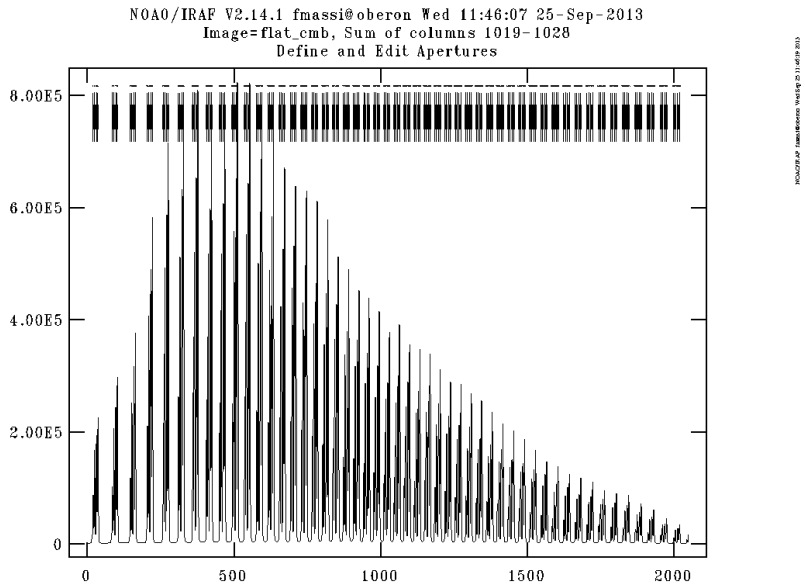

Figure 20: Graphic window showing the apertures selected by the algorithm. The signal is plotted along the reference column; each peak represents a sub-track, each group of four close-by sub-tracks represents a spectral order. The numbered segments on top of the panel mark the selected apertures and must always correspond to the peaks.

1. to start the help page, type "?". The cl window will display a list of commands that you can scroll down. After you have finished, type first "q" and then "return" to return the cursor to the graphic window.

2. to enlarge the view, first type "w"; then move the cursor marker to the lower left corner of the area you want to enlarge and type "e". Finally, move the cursor marker to the upper right corner of the area you want to enlarge and type again "e". Figure 21 shows an example of the result of this sequence of commands.

3. to return to the largest view, type "w" and then "a".

4. to enlarge the width of the current aperture (at the beginning, aperture number 1, i. e. order 32), move the cursor marker to the lower end of the interval and type “l”. Then move the cursor marker to the upper end of the interval and type “u”.

5. to move to the next aperture, type “+”. To move to the previous aperture, type “-”.

6. to enlarge the width of all apertures simultaneously, move the cursor marker to the lower end of the interval at the current aperture and type “L”. Then move the marker to the upper end of the interval and type “U”.

We advice that you do not display another frame column (which is possible) during the trace fitting session, it may cause errors in the trace centring.

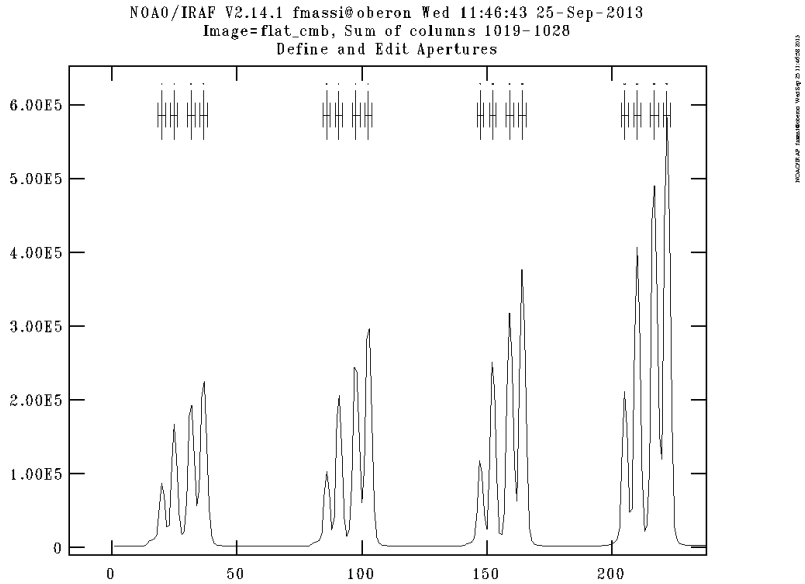

Figure 21: Graphic window showing the apertures found, after zooming using the graphic commands as explained in the text.

If you are using APEDIT, you may also want to review the two intervals outside each aperture that define the windows that will be used to estimate the background to subtract. This can be done for the current aperture by typing “b”. This will move you to the background session. You can enlarge the view as explained above. In Fig. 23, the graphic window displays the background intervals (horizontal segments at the bottom of the panel) for aperture 1. Be aware that the intervals are defined as pixel distances from the aperture centre along a column (e. g., $-a : -b, c : d$). If you want to change background intervals, type “:sample $-a : -b, c : d$”. Then, go back to the aperture editing session by typing “q”. Note that you can only change aperture from the aperture editing session.

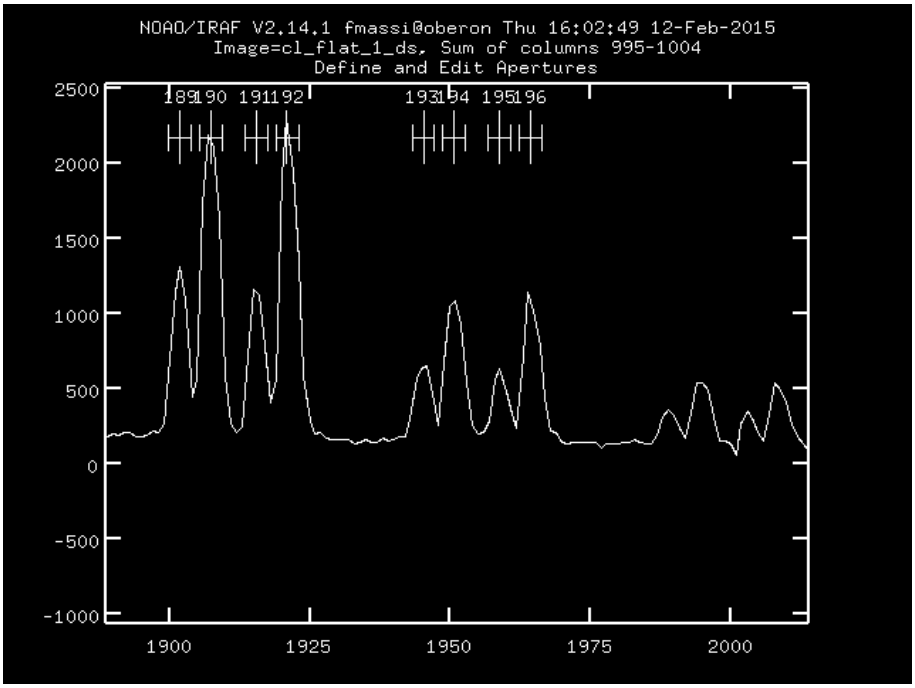

Figure 22: Graphic window showing the highest order apertures found, after zooming using the graphic commands as explained in the text. Note that order 81 is not labelled and the four sub-tracks of order 80 are numbered from 193 to 196.

Having finished with aperture editing, you need to move to the next step by typing "q". You will be asked other questions, just answer yes to all.

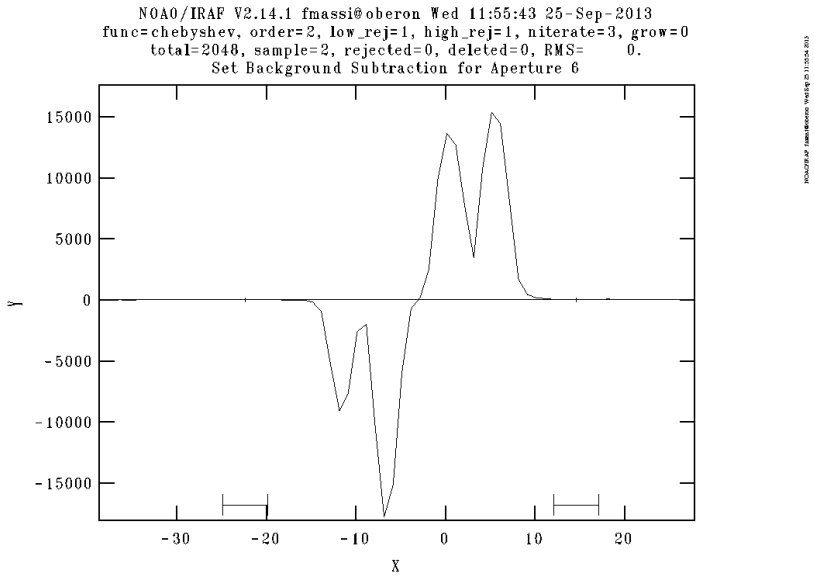

Figure 23: Graphic window showing the selected background window for aperture 1 (i. e., the two horizontal bars at $Y \sim 20000$).

Appendix B: Fitting traces

Once out of the aperture editing display of APFLATTEN, you will be shown the fits to the tracks (trace fits), one by one, in the graphic window; the first display will show the curve fit to aperture 1 (see Fig. 24). By typing "q", it will move to the next aperture. The long and short of it is, an algorithm finds all peaks along columns, then polynomials are fitted to each string of peaks corresponding to the same track. The fit parameters are listed on top of the graphic window. Keep an eye on the RMS, it must be less than few hundredths of a pixel for the fit to be good enough. Otherwise, you may need to increase the fit order. The graphic commands to do this are listed below. Be aware that after changing any parameter, you will have to redo the fit by typing "f". A good fit looks like that in Fig. 24. Large number apertures may exhibit problems like that in Fig. 25; just delete the deviant points on the top right corner and redo the fit as explained below. This is ok and due to the low signal level at the highest orders near the right edge of the frame. Should something like this happen on the central part of the detector, where the signal is always strong, the most likely cause being the presence of bad pixels, check that the frame has been cleaned correctly. A low signal in the central parts of the detector would indicate that the frame has some major problems. Finally, order 80 do not cover the whole range of pixel columns across the detector (Fig. 26), so just delete the deviant pixels as explained below.

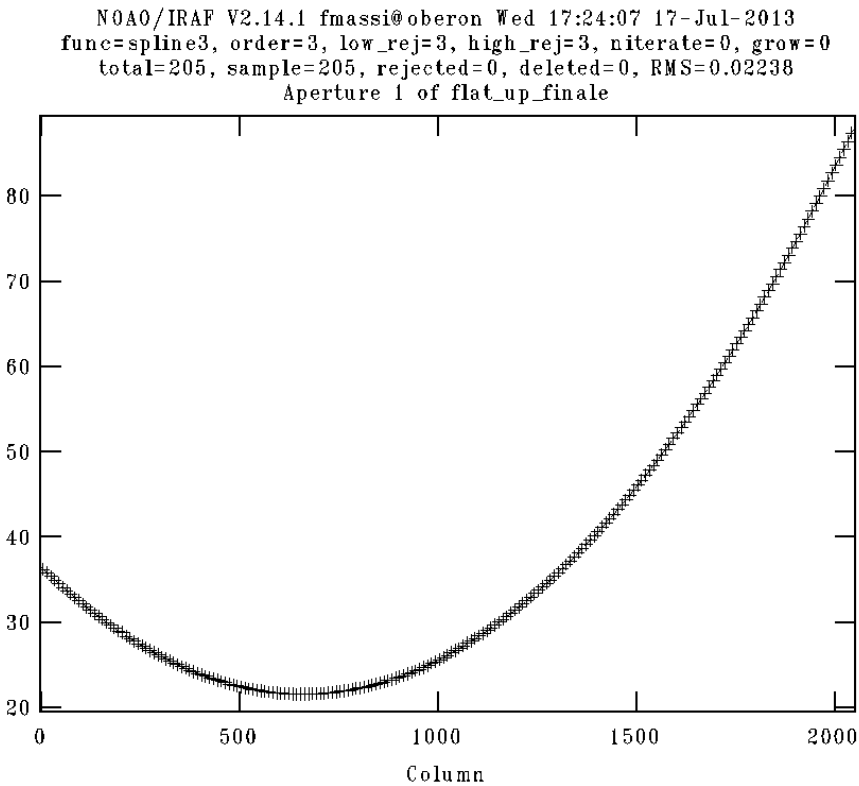

Figure 24: Graphic window showing a typical trace fit.

1. to start the help page, type "?". The cl window will display a list of commands that you can scroll down. After you have finished, type first "q" and then "return" to return to the graphic window.

2. to enlarge the view, first type "w"; then move the cursor marker to the lower left corner of the area you want to enlarge and type "e". Finally, move the cursor marker to the upper right corner

of the area you want to enlarge and type again “e”.

3. to return to the largest view, type “w” and then “a”.
4. to delete a deviant point, put the cursor marker as near as possible to it and type “d”.
5. to undelete a deleted point, put the cursor marker as near as possible to it and type “u”.
6. to change function, type “:function *fname*”, where fname can be chebyshev, legendre, spline3, or spline1.
7. to change polynomial order, type “:order *value*” where value is the chosen order.
8. to change low (high) rejection threshold, type “:low_reject *value*” (“:high_reject *value*”), where value is the rejection threshold in sigma units.
9. to change the number of rejection iterations, type “:niterate *value*”, where value is the number of rejection iterations.
10. to redo the fit after changing any parameter, type “f”.

When you are satisfied with the current fit, type “q” to move to the next aperture (answer yes to every question). You will be shown all trace fits one by one. **Note that the fit parameters are global, so the last parameters input are used to update the fits to all traces.** After the last aperture has been fit, typing “q” will move the display to the next graphic session.

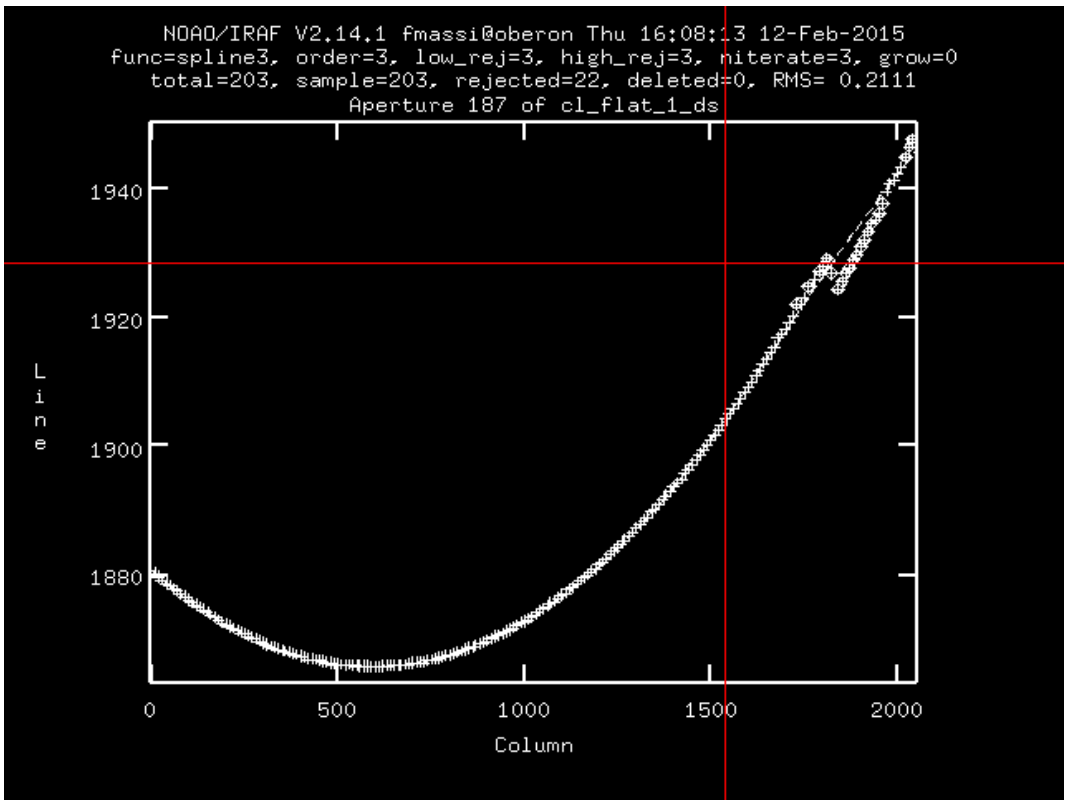

Figure 25: Graphic window showing a typical trace fit at high orders. Note the deviant points on the top right corner of the panel, which need being deleted and refitting.

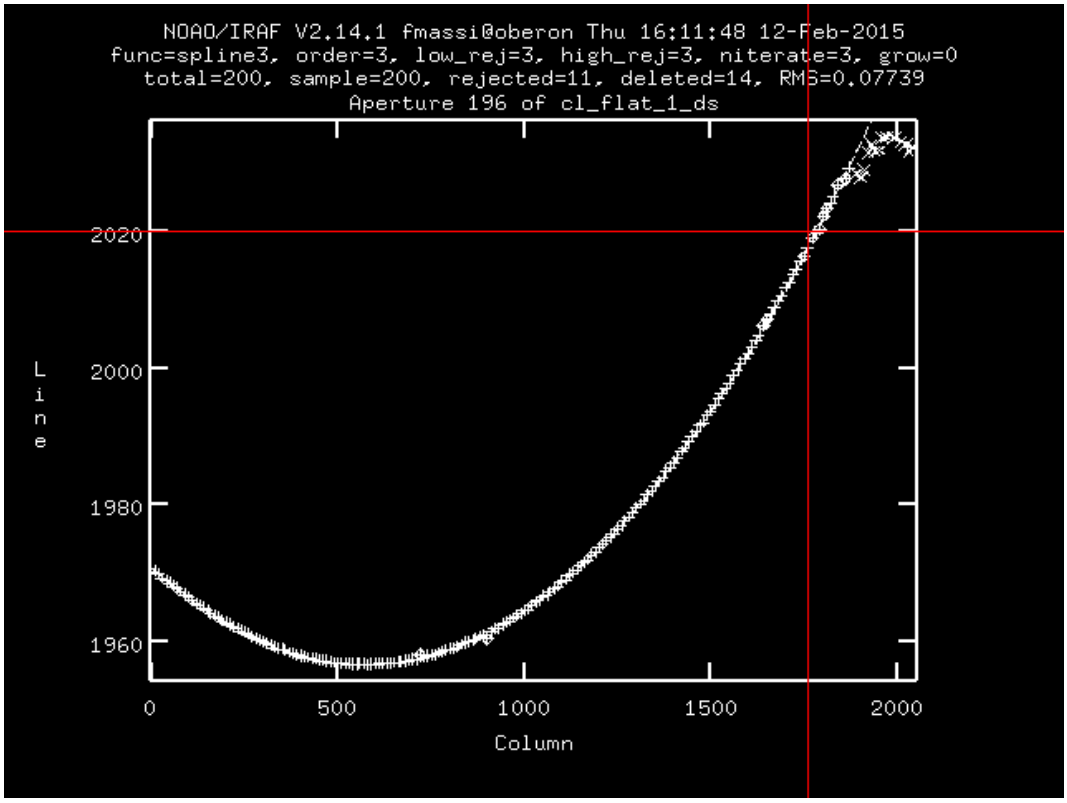

Figure 26: Graphic window showing the curve fit to aperture n. 196. Note that this order does not span the whole range of pixel columns across the detector.

Appendix C: Fitting spectra

When using APFLATTEN, after the trace fit session is over, the spectrum extraction process will start. For each trace, all pixels in a column within a given distance (according to the selected parameters) from the trace centre (i. e., falling inside the corresponding aperture)) are summed together, and a 1D spectrum is constructed. The graphic window will display the spectrum extracted from aperture 1 and the fit results. As can be seen in Figs. 27 and 28, the fit parameters are listed on top of the graphical window, whereas the polynomial fit is overlaid on the spectrum, and the rejected points are marked. If you are not satisfied with the fit results, you may want to change polynomial order and refit as explained below. Otherwise, by typing "q" you will move to the next aperture. Be aware that after changing any parameter, you will need to type "f" to perform a new fit.

IMPORTANT REMARK Flat-field spectra are only intended to remove the pixel-to-pixel response variations of the detector. So, you don't have to fit every small oscillation of the spectrum. You will need to use a polynomial of order 5 to follow the spectrum at the highest orders, but be careful not to exceed this value. Alternatively, you can simply use an order 1 to also remove the spectrograph response (during the flat-fielding phase) normalising the target 1D spectra to the continuum.

1. to start the help page, type "?". The cl window will display a list of commands that you can scroll down. After you have finished, type first "q" and then "return" to return to the graphic window.
2. to enlarge the view, first type "w"; then move the cursor marker to the lower left corner of the area you want to enlarge and type "e". Finally, move the cursor marker to the upper right corner of the area you want to enlarge and type again "e".
3. to return to the largest view, type "w" and then "a".
4. to delete a deviant point, put the cursor marker as near as possible to it and type "d".
5. to undelete a deleted point, put the cursor marker as near as possible to it and type "u".
6. to change function, type ":function *fname*", where fname can be chebyshev, legendre, spline3, or spline1.
7. to change polynomial order, type ":order *value*" where value is the chosen order.
8. to change low (high) rejection threshold, type ":low_reject *value*" (":high_reject *value*"), where value is the rejection threshold in sigma units.
9. to change the number of rejection iterations, type ":niterate *value*", where value is the number of rejection iterations.
10. to redo the fit after changing any parameter, type "f".

When you are satisfied with the current fit results, type "q" to move to the next aperture (answer yes to every question). You will be shown the curve fit to all extracted spectra one by one. **Note that the fit parameters are global, so the last parameters input are used to update the fits to all spectra.** After the last spectrum has been fit (aperture 196), typing "q" will move you to the next graphic session. **Note that the topmost four apertures (number 193–196) do not span the whole column range across the detector, so you will obtain something like**

in Fig. 29. You need to delete the points in the area with signal 0 to avoid the fitted polynomial diverging too much from the signal around column 2000. On the other hand, the polynomial oscillation inside the area without signal is irrelevant.

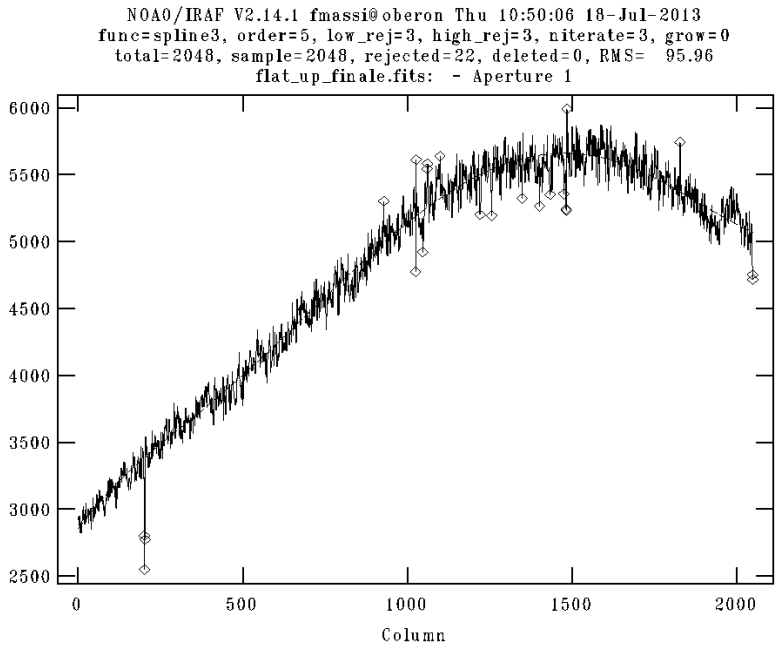

Figure 27: Graphic window showing a polynomial fit to one of the extracted spectra from the lowest spectral orders.

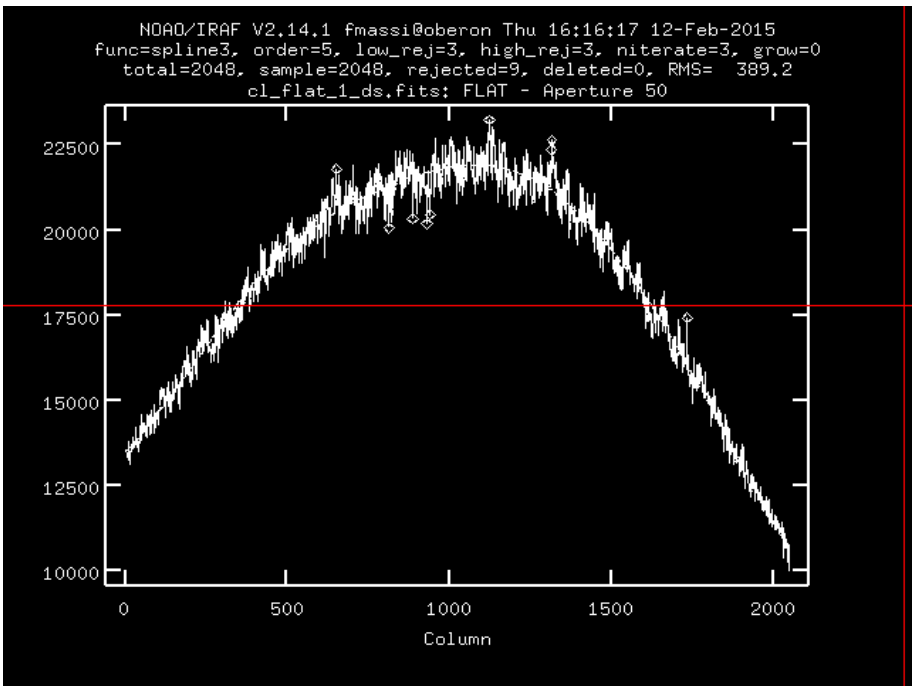

Figure 28: Graphic window showing a polynomial fit to one of the extracted spectra from the central spectral orders.

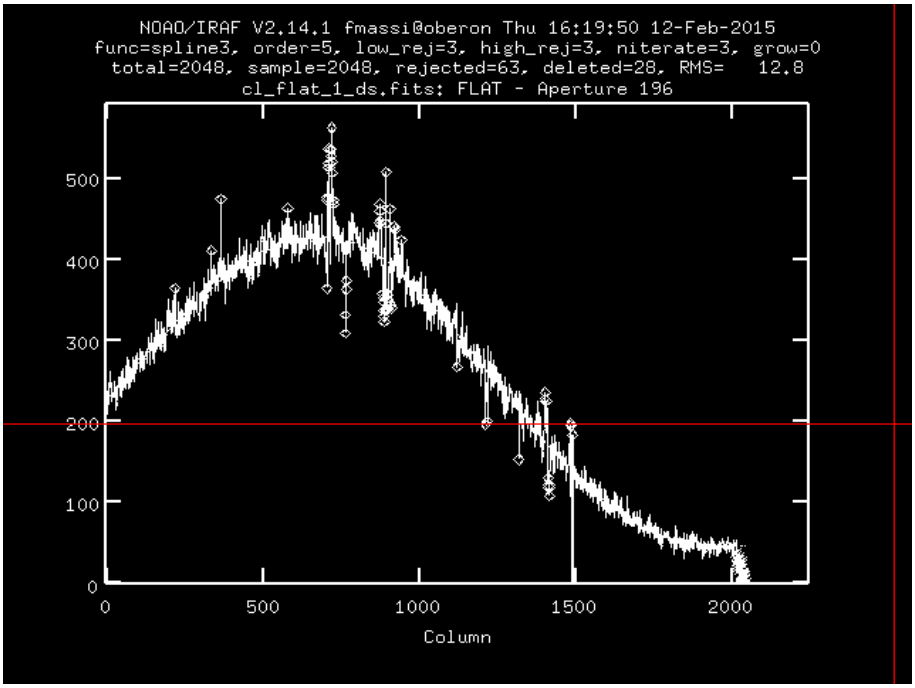

Figure 29: Graphic window showing the polynomial fit to aperture n. 196. Note that the corresponding spectral order (i. e., apertures 193–196, order 80) does not span the whole column range across the detector, so take care to delete the points inside the area without signal and refit, to avoid oscillations at the edge.

Appendix D: Editing spectra using SPLOT

To edit the extracted (wavelength-calibrated) 1D spectra, i. e. the fits files with a string ".ec.fits" added to the name, just use the task SPLOT. This task is also invoked by DOECSLIT at end of the job to allow you to revise the extracted spectra. To run SPLOT on a spectrum, just type "splot SPECNAME", where specname is the name of the fits file with the extracted 1D spectra. You will be asked which aperture you want to review (of course, from 1 to 49). Input its number and enter "return". Below, a list of basic graphic commands to allow handling SPLOT.

1. to start the help page, type "?". The cl window will display a list of commands that you can scroll down. After you have finished, type first "q" and then "return" to return to the graphic window.

2. to enlarge the view, first type "w"; then move the cursor marker to the lower left corner of the area you want to enlarge and type "e". Finally, move the cursor marker to the upper right corner of the area you want to enlarge and type again "e".

3. to return to the largest view, type "w" and then "a".

4. to change aperture, type "# value", where value is the aperture number.

5. to quit, type "q".

The task also allows you to perform some operations on the spectrum, see the help for more information.

Appendix E: Computing effective gain and readnoise

Readout noise and gain of a single raw GIANO frame are listed in Table 1. However, since a final GIANO 2D image gets somewhat modified, due to subtraction, sum, average, etc., its effective readnoise and gain (before 1D spectrum extraction) are also modified. To compute them, simply apply recursively (i. e., after each step) the following rules:

Sum of N frames:
effective exposure time = N x exposure time
effective gain = gain (e^-/ADU)
effective readout noise = $\sqrt{(N)}\times$ readout noise (e^-)

Subtraction of 2 frames:
effective exposure time = exposure time
effective gain = gain (e^-/ADU)
effective readout noise = $\sqrt{(2)}\times$ readout noise (e^-)

Average of N frames:
effective exposure time = exposure time
effective gain = $N\times$ gain (e^-/ADU)
effective readout noise = $\sqrt{(N)}\times$ readout noise (e^-)

Median of N frames:
effective exposure time = exposure time
effective gain = $2 \times N\times$ gain /3 (e^-/ADU)
effective readout noise = $\sqrt{(2 \times N/3)}\times$ readout noise (e^-)

The final subtracted (and combined) frames will then have exposure time $t_{\rm eff}$, gain $g_{\rm eff}$, and readout noise $R_{\rm eff}$. However, when combining together the four sets of 1D spectra extracted from a typical frame you have to keep in mind that the signal from each fibre has been split in two and that each aperture has a width of four pixel. Hence, by applying again the formulae above one obtains for the final 1D spectra:

Single trace out of a four-trace order

effective exposure time = $t_{\rm eff}/2$
effective gain = $g_{\rm eff}$ (e^-/ADU)
effective readout noise = $2 \times R_{\rm eff}$ (e^-)

Summing two traces from the same fibre (out of a four-trace order)

effective exposure time = $t_{\rm eff}$
effective gain = $g_{\rm eff}$ (e^-/ADU)
effective readout noise = $2.83 \times R_{\rm eff}$ (e^-)

Averaging two traces from the same fibre (out of a four-trace order)

effective exposure time = $t_{\rm eff}/2$
effective gain = $2 \times g_{\rm eff}$ (e^-/ADU)
effective readout noise = $2.83 \times R_{\rm eff}$ (e^-)

Summing all four traces out of a four-trace order

effective exposure time = $2 \times t_{\rm eff}$
effective gain = $g_{\rm eff}$ (e^-/ADU)
effective readout noise = $4 \times R_{\rm eff}$ (e^-)

Averaging all four traces out of a four-trace order

effective exposure time = $t_{\rm eff}/2$
effective gain = $4 \times g_{\rm eff}$ (e^-/ADU)
effective readout noise = $4 \times R_{\rm eff}$ (e^-)

Appendix F: Automatic arc assignment parameters

The wavelength-calibration is performed by DOECSLIT. The task allows selecting the lamp spectra to use for calibrating each science spectrum based on several possible criteria (nearest in time, average, etc.). The field *arcs* must be set to a list of calibration lamp 2D spectra. The criterion to select ("assign") a spectrum from the list is controlled by the value in the field *select*, in the "Automatic arc assignment" parameter section of the parameter file SPARAMS. Detailed assignments are possible (see the help), but the most useful values are the following. These are based on the observation date and time (in Julian day, having set *sort* to "jd").

1. *average* The average of the first two spectra in the list is used. If only one reference spectrum is specified then it is assigned with a warning. If more than two reference spectra are specified then only the first two are used and a warning is given.

2. *following* Select the nearest following spectrum in the reference list based on the sorting parameter (in our case, time). If there is no following spectrum it uses the nearest preceding spectrum.

3. *interp* Interpolate between the preceding and following spectra in the reference list based on the sorting parameter. If there is no preceding and following spectrum it uses the nearest spectrum. The interpolation is weighted by the relative distances of the sorting parameter (in our case, time).

4. *nearest* Select the nearest spectrum in the reference list based on the sorting parameter (in our case, time).

5. *preceding* Select the nearest preceding spectrum in the reference list based on the sorting parameter (in our case, time). If there is no preceding spectrum it uses the nearest following spectrum.

Appendix G: Wavelength-calibration

After confirming the choice of the apertures (just confirm, they will have already been selected through APEDIT), DOECSLIT will move you to the wavelength-calibration session. There is no trace-fitting session, since we will use the traces from APFLATTEN. The graphic window will display a 1D spectrum extracted from one of the spectral orders of the calibration lamp frame (see Fig. 30). You need to identify a few lines and input their wavelength for a few orders. We advice you to use aperture number 1, 2, 10, 11, 33, 35, 38, 49 and to input the wavelengths of the lines shown in Fig. 33, 34, and 35. We have checked against the wavelengths of telluric absorption and emission lines ([7,8]) that using all the lines in the figures results in a wavelength accuracy < 0.01 nm on the whole passband (see Fig. 31; note that the pixel sampling interval is ~ 0.02 nm).

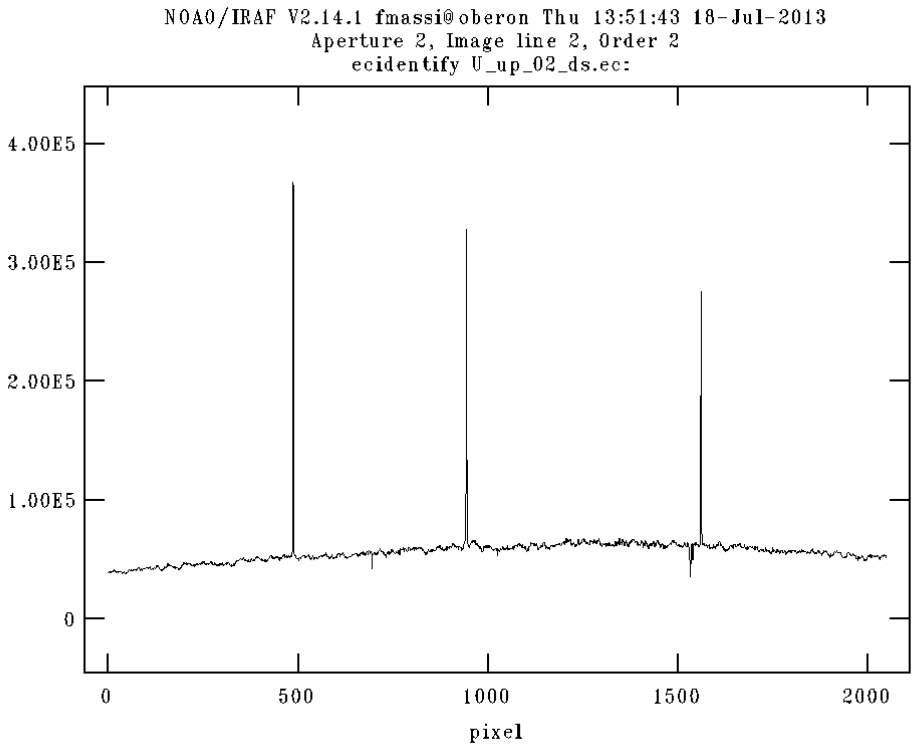

Figure 30: Graphic window showing the spectrum extracted from an aperture of the calibration lamp frame, waiting for the lines displayed to be identified.

The line wavelengths must be input in nm (as they are listed in the reference file list). Note also that DOECSLIT refers to the apertures or traces as orders. Actually, aperture 1 corresponds to order 32 of the echelle spectrum. This task uses the line catalogue file indicated in the field *coordli* of the SPARAMS file. This file is part of the GIANO_TOOLS package, so do not worry about it. **If a wavelength solution already exists (e. g., if you have already used GIANO_REIDENTIFY), then the graphic window will display a calibrated spectra with identified lines already marked. Skip the line identification stage and have a look at the fit results by pressing "f" (as a graphic command). Check the fit results as explained below. If not satisfied, you can unmark all lines and redo the line assignment. But usually you should be able to leave the session without any action at all.**

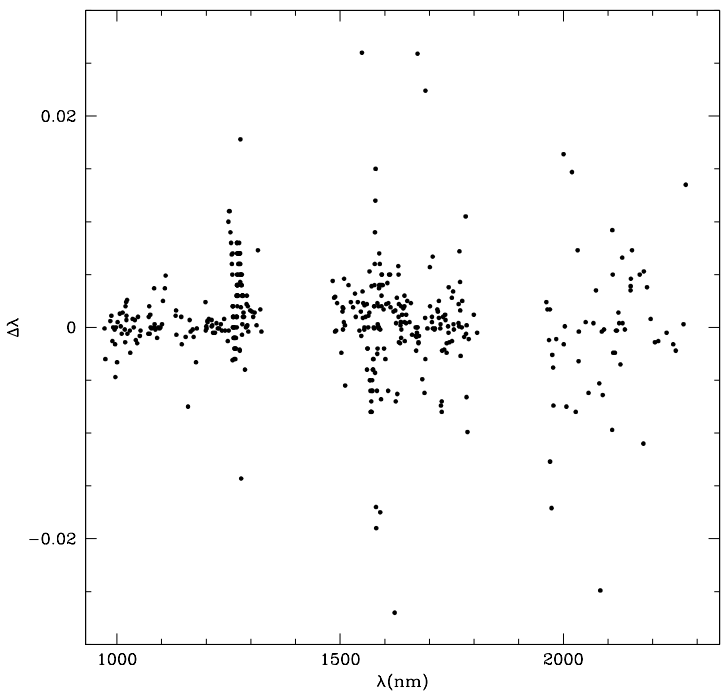

Figure 31: Wavelength difference (in nm) between observed absorption or emission telluric lines and their counterparts identified from [7] or [8]. We note that differences > 0.01 nm usually arise due to heavy blending or low s/n of the line in the spectrum, making it difficult to locate its centre. Note that the pixel sampling interval is ~ 0.02 nm).

In order to identify the lines and input their wavelengths, use the following graphic commands:

1. to start the help page, type "?". The cl window will display a list of commands that you can scroll down. After you have finished, type first "q" and then "return" to return to the graphic window.

2. to enlarge the view, first type "w"; then move the cursor marker to the lower left corner of the area you want to enlarge and type "e". Finally, move the cursor marker to the upper right corner of the area you want to enlarge and type again "e".

3. to return to the largest view, type "w" and then "a".

4. to enter the wavelength of a line, first move the graphic cursor marker onto the line, then type "m". The line will be marked and the graphic window will ask for the wavelength. Type the wavelength (in nm) and enter "return".

5. to delete an identified line, first move the graphic cursor marker onto the line, then type "u".

6. to change aperture, type "o" and enter the aperture number.

After all the lines recommended have been identified, type "f" to perform a preliminary fit. The graphic window will display the fit results (fit residuals vs. pixel scale) as in Fig. 32. As usual, the fit parameters are listed on the top of the graphic window. It is mandatory that you check that the field *offset* displayed in the graphic windows is offset=31 (i. e., aperture 1 corresponds to order $31 + 1 = 32$, as it must be).

If the offset is not 31, return to the previous step by typing "q". Check the line identifications and, where needed, delete the incorrect ones by typing "u", then mark them again with "m" and assign the correct wavelengths.

The next result to check is the rms, which must be less than few hundredth of a nm. To get a better fit, you can perform the following operations:

1. to start the help page, type "?". The cl window will display a list of commands that you can scroll down. After you have finished, type first "q" and then "return" to return to the graphic window.

2. to change the x-axis scale from pixel to order, type "x" and then "o".

3. to change the x-axis scale from pixel to wavelength, type "x" and then "w".

4. to go back to the x-axis scale in pixel, type "x" and then "p".

5. to delete a deviant point, put the cursor marker as near as possible to it and type "u".

6. to change function, type ":function *fname*", where fname can be chebyshev or legendre.

7. to change polynomial orders, type ":xorder *value*" or/and ":yorder *value*" where value means the order.

8. to change low (high) rejection threshold, type ":lowreject *value*" (":highreject *value*"), where value means the rejection threshold in sigmas.

9. to change the number of rejection iterations, type ":niterate *value*", where value means the number of rejection iterations.

10. to redo the fit after changing any parameter, type "f".

Be aware that you need to redo the fit (by typing "f") after any change of parameters. When you are satisfied with the fit, return to the line identification display by typing "q".

Now you can run the automatic finding procedure that browses into the line catalogue and try to identify as many lines as possible. Just type:
":maxfeatures 500" followed by ENTER and then
"l".
After this, you can move the display within orders (with "o" as explained above) to verify that many more lines have been marked. Perform a new fit by typing "f". You will return to the fit result display. Follow the explanations above to obtain a better fit. **Check again that offset is equal to 31**. When satisfied, type "q". You will be again moved to the line identification display (like Fig. 32). Type "q" again and you will be through.

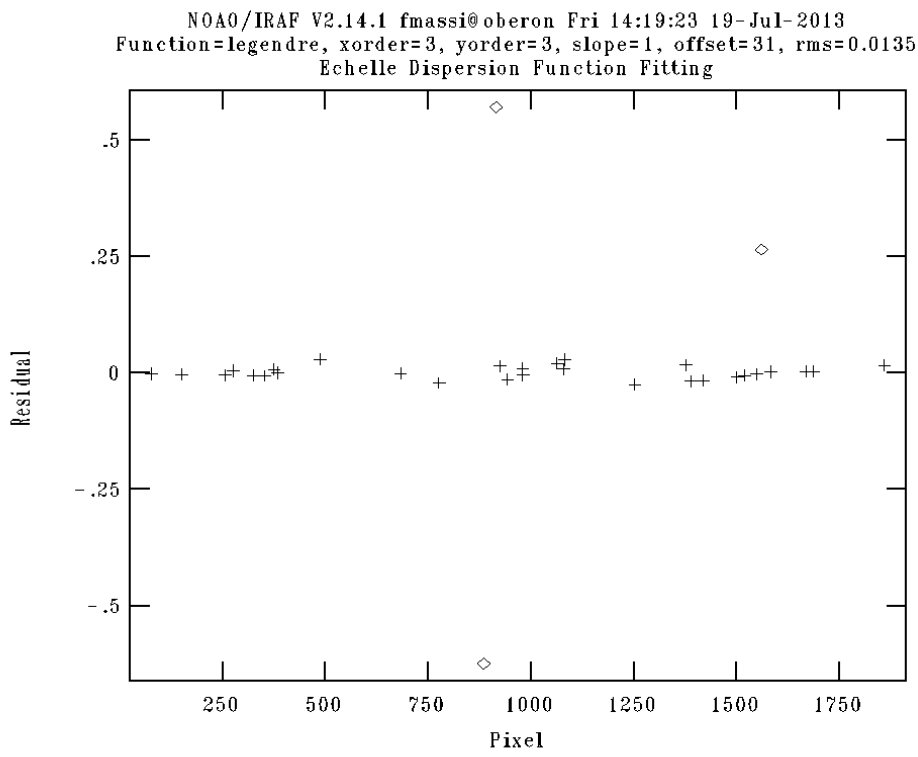

Figure 32: Graphic window showing the preliminary wavelength solution after the reference line identification step.

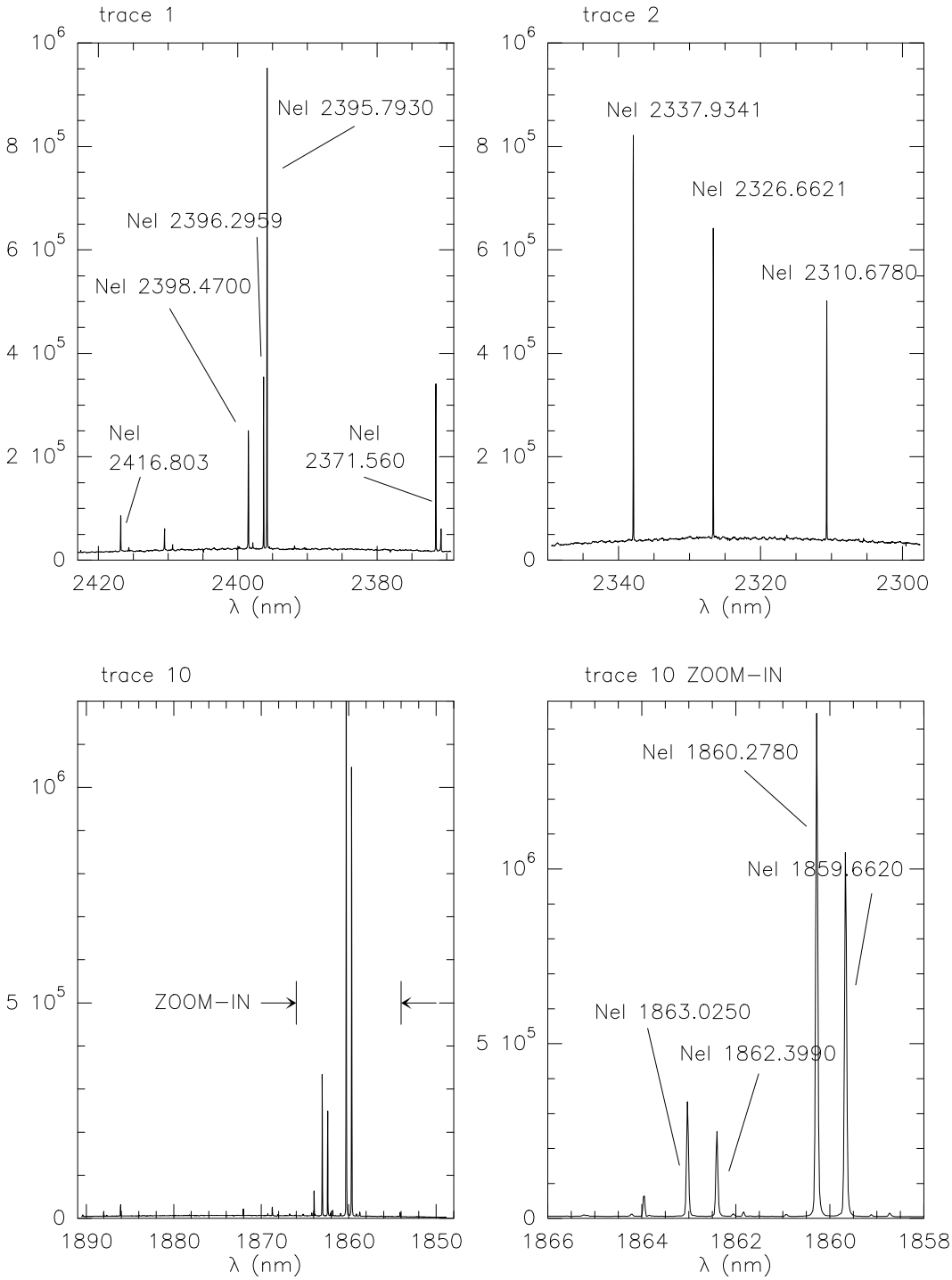

Figure 33: Calibration lamp lines to be used for wavelength calibration.**Note that the wavelength scale is reversed so that it mimics the pixel scale shown in the graphic windows.**

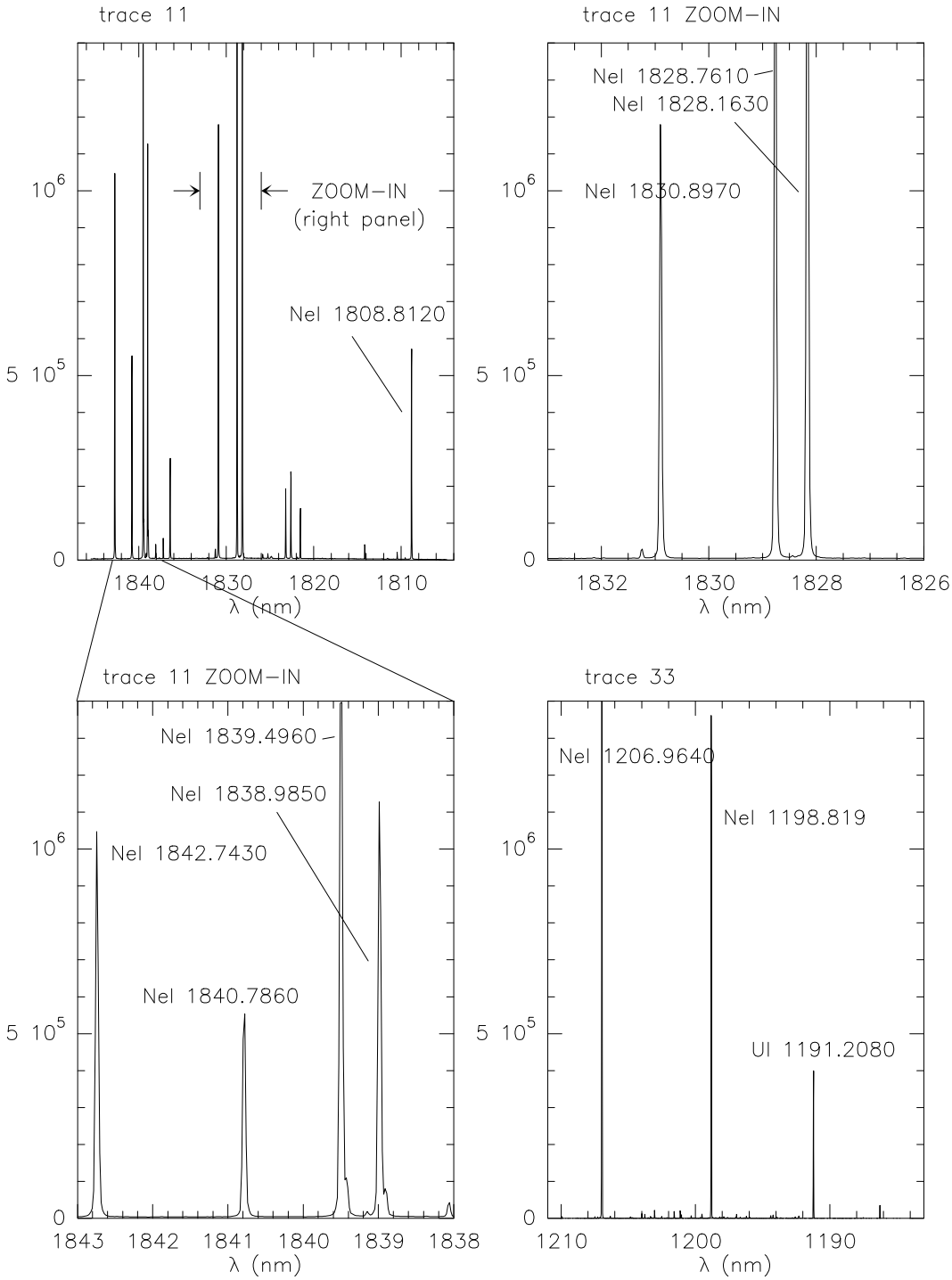

Figure 34: Calibration lamp lines to be used for wavelength calibration. **Note that the wavelength scale is reversed so that it mimics the pixel scale shown in the graphic windows.**

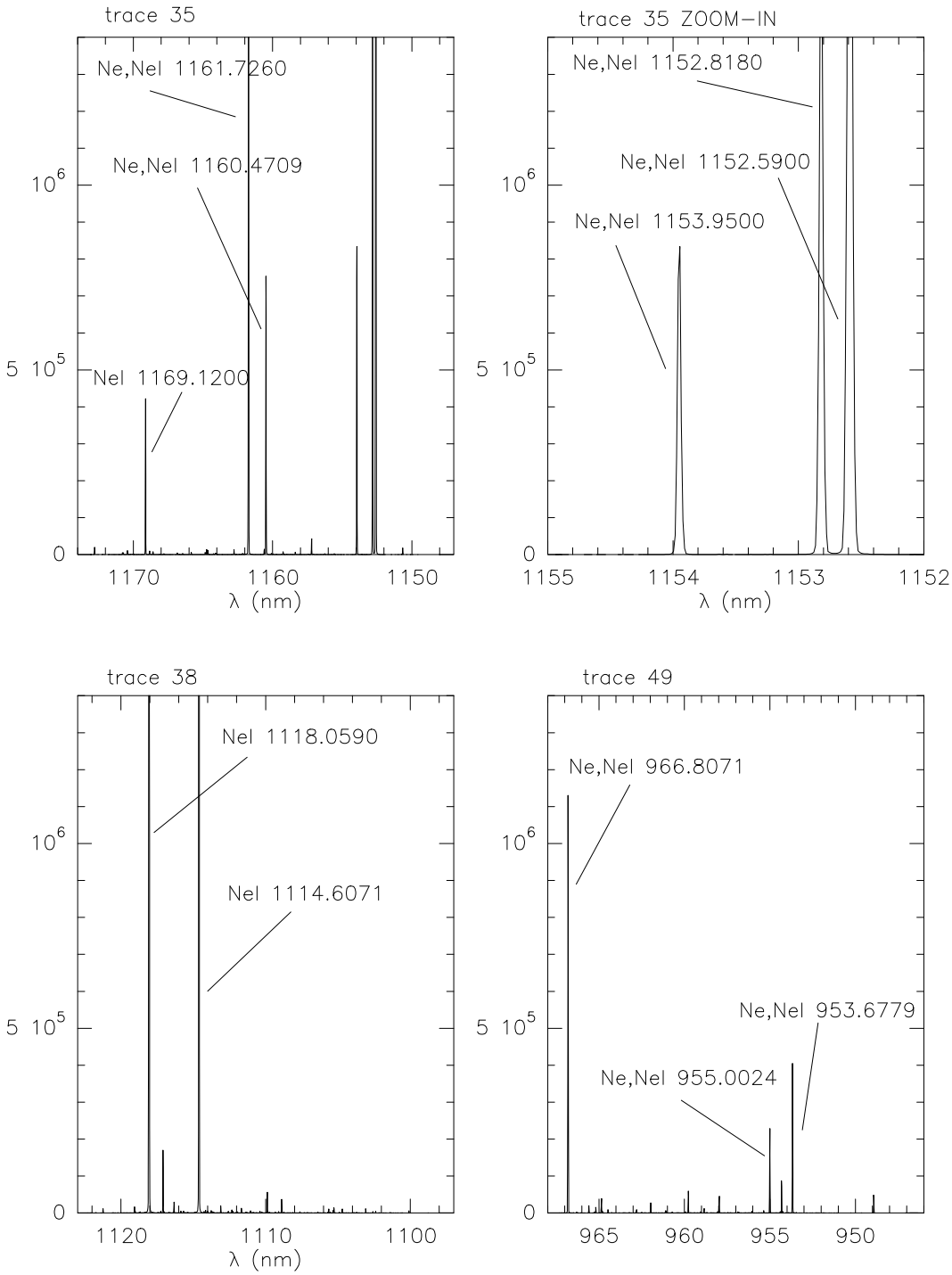

Figure 35: Calibration lamp line to be used for wavelength calibration.**Note that the wavelength scale is reversed so that it mimics the pixel scale shown in the graphic windows.**

Appendix H: Wavelength range per order

Order 32	2.3699–2.4224 μm
Order 33	2.2979–2.3490 μm
Order 34	2.2302–2.2799 μm
Order 35	2.1664–2.2146 μm
Order 36	2.1062–2.1531 μm
Order 37	2.0492–2.0948 μm
Order 38	1.9952–2.0396 μm
Order 39	1.9440–1.9872 μm
Order 40	1.8953–1.9375 μm
Order 41	1.8490–1.9902 μm
Order 42	1.8049–1.8451 μm
Order 43	1.7629–1.8021 μm
Order 44	1.7228–1.7611 μm
Order 45	1.6844–1.7219 μm
Order 46	1.6478–1.6844 μm
Order 47	1.6126–1.6486 μm
Order 48	1.5790–1.6142 μm
Order 49	1.5467–1.5812 μm
Order 50	1.5157–1.5495 μm
Order 51	1.4860–1.5191 μm
Order 52	1.4573–1.4898 μm
Order 53	1.4298–1.4616 μm
Order 54	1.4033–1.4345 μm
Order 55	1.3777–1.4084 μm
Order 56	1.3531–1.3832 μm
Order 57	1.3293–1.3589 μm
Order 58	1.3063–1.3354 μm
Order 59	1.2841–1.3127 μm
Order 60	1.2627–1.2908 μm
Order 61	1.2419–1.2696 μm
Order 62	1.2219–1.2491 μm
Order 63	1.2024–1.2292 μm
Order 64	1.1836–1.2100 μm
Order 65	1.1654–1.1913 μm
Order 66	1.1477–1.1732 μm
Order 67	1.1305–1.1557 μm
Order 68	1.1138–1.1387 μm
Order 69	1.0977–1.1221 μm
Order 70	1.0819–1.1060 μm
Order 71	1.0667–1.0904 μm
Order 72	1.0518–1.0753 μm
Order 73	1.0374–1.0605 μm
Order 74	1.0233–1.0461 μm
Order 75	1.0096–1.0321 μm
Order 76	0.9963–1.0185 μm
Order 77	0.9833–1.0053 μm
Order 78	0.9707–0.9923 μm
Order 79	0.9584–0.9797 μm
Order 80	0.9464–0.9675 μm

Wavelengths are in vacuum.
(from Table 2 of *GIANO cookbook for proposers*)

Index

2D or 1D flat field correction?, 10

Appendix A: Checking the apertures, 31
Appendix B: Fitting traces, 34
Appendix C: Fitting spectra, 37
Appendix D: Editing spectra using SPLOT, 40
Appendix E: Computing effective gain and readnoise, 41
Appendix F: Automatic arc assignment parameters, 43
Appendix G: Wavelength-calibration, 44
Appendix H: Wavelength range per order, 51
Averaging the flat-field frames, 12

Bibliography, 30

Cloning frames, 20
Combining the extracted 1D spectra together, 27

Dark subtraction and bad-pixel correction, 8
Data reduction with IRAF, 6

First reduction steps, 7
Flat-field correcting the spectra in 1D, 25
Flat-fielding the targets in 2D, and final steps before spectrum extraction, 18

Giano_tools, 7

Handling spectra: IRAF packages to use, 13
How to find traces, 14

IRAF basics, 3

Learning more about flat-field frames, 10

Normalising the extracted 1D spectra to the continuum, 26

Reduction quick guide, I

Setting the ECHELLE session to extract GIANO spectra, 13
Setting up an IRAF session to reduce data from GIANO, 6
Spectrum extraction, 21
Starting point: GIANO raw data, 1

Troubleshooting, III

Using APFLATTEN to fit the traces and construct a 2D flat-field frame, 16

Viewing the 1D final spectra, 28