Reducing Spectra with IRAF

Heidi Lietzen*and Sami-Matias Niemi

September 7, 2007

1 Introduction

Since we noticed that there are no simple instructions on how to reduce spectra with IRAF we made one ourselves. These instructions are written so that they can be used by someone who hasn't used IRAF before, and therefore we start with a few most important basic things about using IRAF in general. More information about using IRAF can be found at http://iraf.net/irafdocs, especially for spectra see "A User's Guide to Reducing Slit Spectra with IRAF" by Massey, Valdes and Barnes. Also the help pages of IRAF contain a lot of useful information.

There are some differences between the different versions of IRAF. We have used versions 2.12.2 and 2.13-beta for doing the work we describe here, but we have not checked if every single default parameter is the same for these versions. If you get some strange looking results in some command, see the `help` page for that command.

These instructions are for a simple case of spectral observations done with a good instrument. If for example there are several targets on the slit, you have to do things a little differently.

As an example we use the spectrum of quasar QSO2112+059. The spectrum was observed at Nordic Optical Telescope with instrument ALFOSC on August 16th 2007. The files observed include the spectrum of the quasar itself, ALqh160035.fits, a spectrum of a wavelength calibration lamp, ALqh160036.fits, and a spectrum of a continuum lamp, ALqh160037.fits. These are shown in figure 1.

*email:heilie@utu.fi

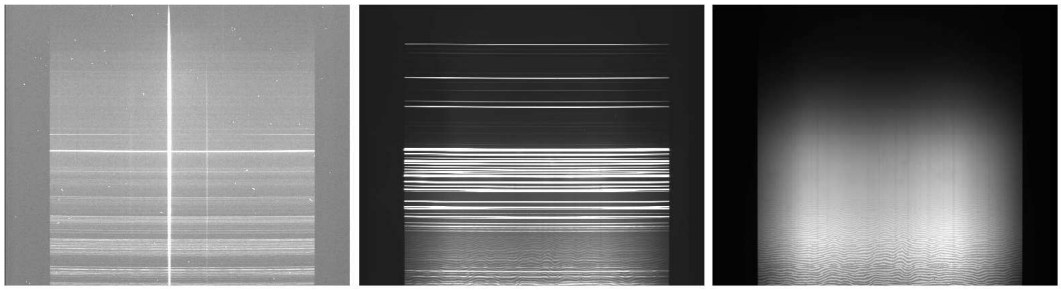

Figure 1: The images needed for a spectrum. *Left:* the target, *middle:* wavelength calibration lamp, and *right:* continuum lamp.

2 IRAF in general

When you start using IRAF for the first time, edit the file login.cl: find the line that says `#set imtype ="imh"`. Delete `#`, and replace `"imh"` with `"fits"`. After this is done you can use images in fits form.

If 'delete' key does not work with `epar` command, edit login.cl: find the line `#set editor = vi`. Delete `#`, and replace `vi` with `emacs`. After this 'delete' key should function properly.

To start IRAF open xterm window and ds9. Then type cl in your choise of terminal window, and the program will start. You should see something like figure 2. The words `dataio`, `dbms` etc. are names of packages. They

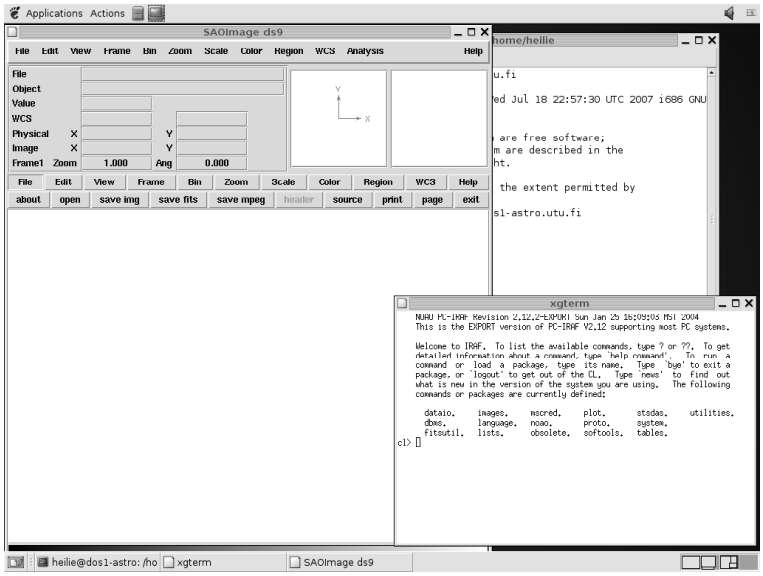

Figure 2: IRAF just after starting the program.

contain the commands that are used in IRAF. To use a command in a certain package go to the right package by typing its name. There can be more than one commands with the same name but in different package, so make sure you have the right one. To get out of a package type `bye`.

There are two ways of running a command. First one is simply to type the name of the command and all its parameters that you need. For example `display ALqh160035[1] 2` will display image in the file ALqh160035.fits in frame 2 of ds9. If you need to change many parameters in the command you can type `epar display`, and you will get the list of all the parameters for the command. Change them in the way you want, and then run the command by typing `:go`. If you don't want to run the command you can get out of this parameter environment with Ctrl-d. For display it may be a good idea to set `fill=yes`. Then the whole image will be displayed, not just a small part of it.

3 Basic reductions

3.1 Bias

This has to be done for all of your images, the spectrum, the wavelength calibration lamp and the continuum lamp image. If you have an overscan area in your images, find out where it is (in x and y pixel values). Also find out the area of the "useful" part of the image. This means the area with the spectrum and without the overscan area and the useless edges that might have a bad image quality. The overscan area and the useful area are shown in figure 3. In this example the overscan is at columns (x coordinate) 2000–2170 and lines (y) 10–2000. The useful image area is at columns 300–1950 and lines 10–2000.

Go to the package noao/imred/ccdred and use the command `ccdproc` to subtract overscan and trim the images. To change the parameters write `epar ccdproc`. Important parameters:

- `images`: The names of the images you want to use. In our example `ALqh160035[1],ALqh160036[1],ALqh160037[1]`.

- `output`: New names for the processed images. If nothing here the original images will be overwritten. If you give new names you have to give a name for every image in the input. The program doesn't run if you use names of already existing images. The names could be for example `b35,b36,b37`.

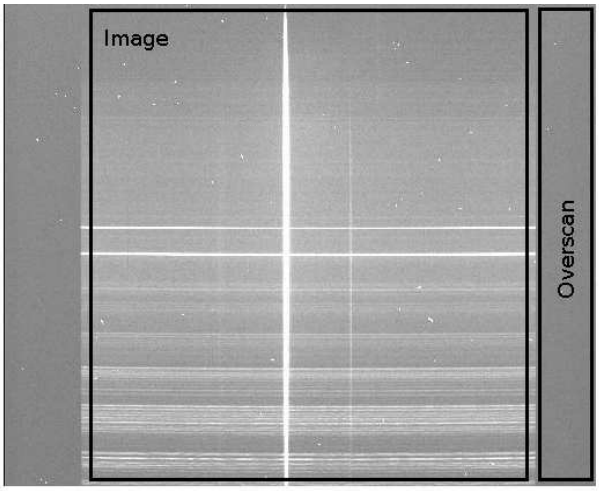

Figure 3: Original image with overscan area.

- `ccdtype`: This should be empty. You can define the type of an image in the header and use this to choose images of a certain type, but if you have not defined the type and have something written here the program doesn't do anything.
- `oversca` and `trim`: `yes`.
- `fixpix, zerocor, darkcor, flatcor, illumco, fringec, readcor` and `stancor`: `no`.
- `biassec`: coordinates of the overscan area in the format [lower x:higher x,lower y:higher y], for example `[2000:2170,10:2000]`
- `trimsec`: The area you want to leave when you trim off the unnecessary edges of the image, for example `[300:1950,10:2000]`

If you need to use bias images and not just the overscan, combine the images using command `zerocombine`. Then use `ccdproc` in the same way but set `zerocor = yes`, `overscan = no`, and `zero =` the name of your combined bias image.

3.2 Flat-field

Flat-fielding is done using the continuum lamp. The illumination of the lamp is not even, but some areas of the image are brighter than others. This has to be corrected before using the image.

First go to package `noao/imred/kpnoslit`, and there `epar response`. Use the following parameters:

- `calibrat`: The name of the continuum lamp image, in our example `b37`.
- `normaliz`: The name of the continuum lamp image again, `b37`. These two first parameters should be the same.
- `response`: Name for the new response image, for example `resp37`
- `functio = legendre`
- `low_rej = 3`
- `high_rej = 3`

Type `:go` and the program will ask if you want to fit interactively. Answer `yes`. Now the idea is to fit the dash-line on the curve. Change the order of the function by typing `:order 15`, and press `f` to fit. The default order is 1 and this changes it to 15. You will see the dash-line change towards the curve. You can also change the number of rejection iterations by typing for example `:niter 5` and fit with `f`. If you still are not satisfied with the result, change the order higher, for example 20. When the curve fits perfectly press `q` to quit. The result image should look like the one in figure 4.

Next step is to determine the illumination of the lamp. Use the command `illumination`, i.e. `epar illumination`. Start with the following parameters:

- `images`: The result of `response`, `resp37`
- `illumina`: Name for the new image, for example `illu37`
- `functio = legendre`
- `order = 20`

Run this interactively. For the first plot just press `q`. Next the program asks if you want to determine illumination interactively for image at bin 1. Answer yes. Then change the order in the same way as in `response`, by typing for example `:order 25` and fit with `f`. When the curve fits well enough press `q`. Then you have to do the same for bins 2, 3, 4 and 5. They should all be approximately similar. The result image can be seen in figure 5.

Now you have all you need for correcting the flat-field image. go back to `ccdred` package and `ccdproc`. Use it for doing the illumination correction to

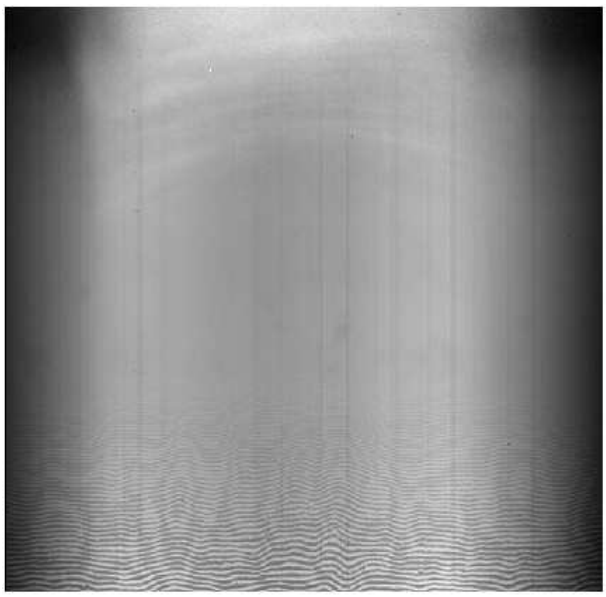

Figure 4: The result image of the task `response`.

the result image of `response`, the one we called `resp37`. The output name could be `red37` because this will be the reduced flat-field image. Set `illumco = yes`, and other corrections no, and give the name of the illumination image `illum = illu37`. Run the command, and you should get the final flat-field image as in figure 6. If you have taken two flat-field images, one before and one after the spectrum, you can combine them by taking an average. You can use the command `imarith` to do the basic arithmetic for the images.

You can now do the flat-field correction for the calibration and the science spectra. For the science spectrum you need only flat-fielding, so set `images = b35`, `output = red35` and `flatcor = yes` in `ccdproc` and the name of the flat-field image at `flat = red37`. For the calibration lamp spectrum you have to do both, illumination and flat-field correction. Set `images = b36`, `output = red36`, `flatcor = yes`, `illumco = yes`, `flat = red37` and `illum = illu37`.

If `ccdproc` doesn't do flat-fielding the problem may be that the flat-field image has the value 0 somewhere. This happens at the overscan area if you haven't trimed the images after subtracting the bias level. You can solve the problem by trimming. Another option is to divide with the flat-field "manually", using `imarith`. In this case you have to normalize the flat-field image first by dividing it by its mean value. You get this value by `imstatistics`.

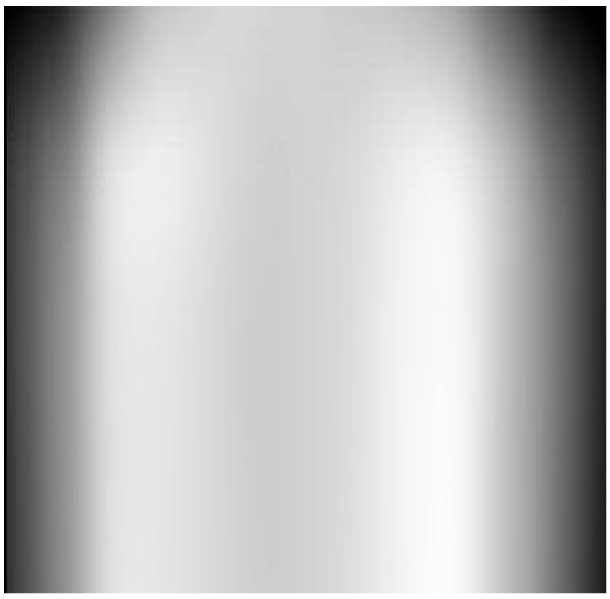

Figure 5: The result image of the task `illumination`.

4 Background

In the spectral image the object itself is the vertical bright line in the center of the image. All the horizontal lines are emission lines of the atmosphere. They have to be removed to see anything in the object spectrum.

First display the image, you will need to look at it. Go to `kpnoslit` and use the command `background`. Parameters to use:

- `input = red35`: the name of the image
- `output = sub35`: the name for the new image
- `axis = 1`: this means whether the background is subtracted along the lines or columns. The other possibility would be `2`.
- `interac = no`
- `sample = 690:720,780:810` Look at the image and find about 30 columns on each side of the object spectrum. Type the x coordinates limiting the chosen columns. See figure 7. This defines the area where the background is determined.
- `naverage = -5`
- `functio = legendre`

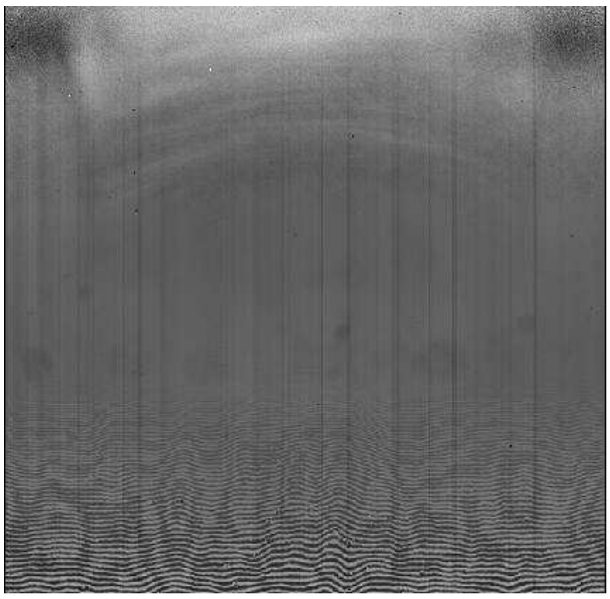

Figure 6: The final flat-field image.

- `order = 3`
- `low_rej = 3`
- `high_re = 3`
- `niterat = 3`

When you run this the result image should look like the one in figure 8. If you still have background emission lines near the target spectrum your sample columns were too far from the spectrum.

5 1-dimensional spectra

Now we can turn the two dimensional image into a one dimensional spectrum. This is done by the command `apall` in `kpnoslit`. This command has lots of parameters:

- `input = sub35` The name of the image.
- `output = 351d` Name for one-dimensional spectrum.
- `interac = yes`
- `find = yes`

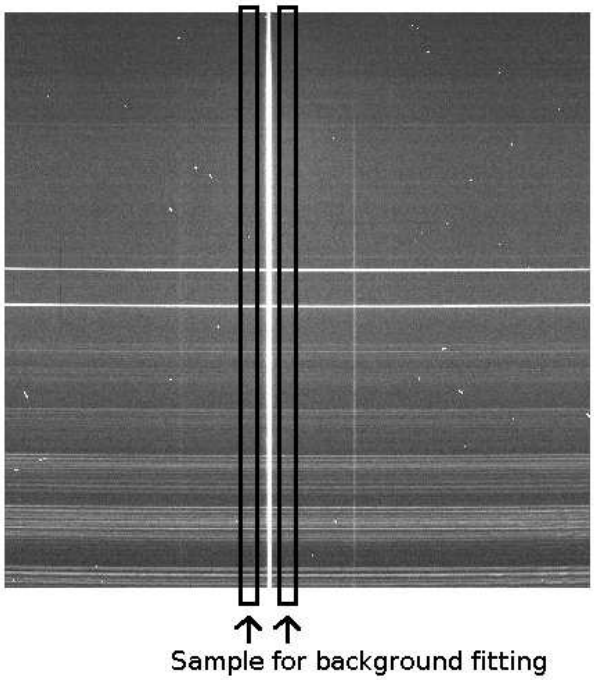

Figure 7: Areas for background subtracting sample.

- `recente = yes`
- `resize = no`
- `edit = yes`
- `trace = yes`
- `fittrac = yes`
- `extract = yes`
- `extras = yes`
- `review = yes`
-
- `line = 1200` This is the line (y coordinate) at which the spectrum is wanted to be found. Pick a line where the spectrum looks bright and wide.

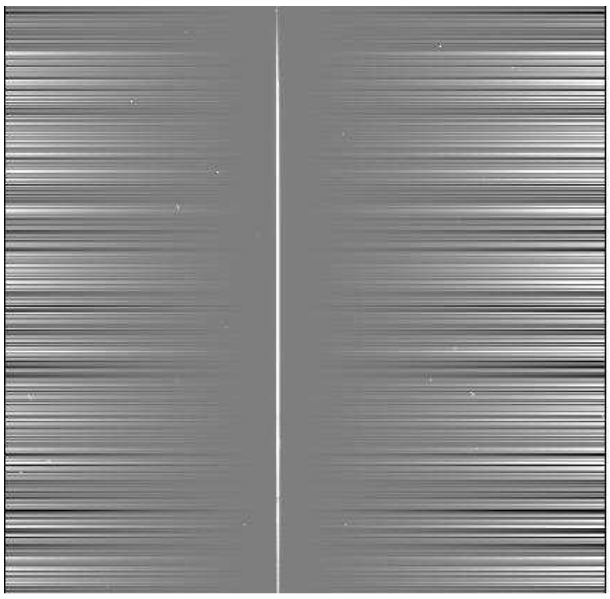

Figure 8: Image after subtracting the background.

- `nsum = 30` The number of lines around the line chosen to be used in finding the spectrum.
- APERTURE PARAMETERS
- `lower = -10`
- `upper = 10` These two should be the same number, but negative for lower and positive to upper limit. This number is the distance from the center of the spectrum that is clearly outside the spectrum (but not too far).
- APERTURE CENTERING PARAMETERS
- `width = 20` The width of the spectrum.
- `radius = 12` This is an error radius for centering the spectrum and it should be slightly larger than half of `width`.
- TRACING PARAMETERS
- `t_nsum = 20`
- `t_step = 20`
- `t_order = 3`

- `t_niter = 1`
- EXTRACTION PARAMETERS
- `weights = variance` This is important for optimal extraction.
- `clean = no`
- `readnoi = 3.2` Readnoise of your instrument. NOT ALFOSC: 3.2 e^-/pixel
- `gain = 0.726` Gain of your instrument. NOT ALFOSC: 0.726 e^-/ADU

Run the command, and answer `yes` to the question of finding apertures. Next question is how many spectra you want the program to find automatically. Answer 1, and `yes` for editing apertures. You will see a plot like the one in figure 9. The spectrum has been found and marked with `1`. If the spectrum was not found automatically you could choose it yourself by pointing it with the mouse and pressing `m`. When you have the spectrum marked press `q` to continue.

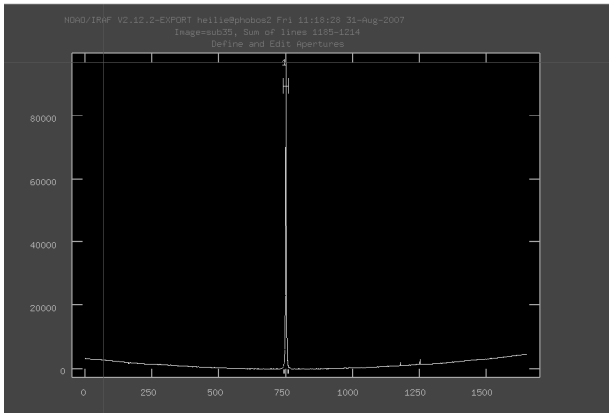

Figure 9: Plot for finding the spectrum.

Answer `yes` to the question of tracing apertures, to the question about doing it interactively and to the one about fitting interactively (that is three times `yes`). You will see a plot like the one in figure 10. Change the order of the fitting function by typing `:order 4` and fit with `f`. If there are points that don't fit on the curve with others, delete them by moving the pointer on the point and pressing `d`. Continue until the dash-line fits well on the points, and then press `q`. Answer `yes` for writing the apertures to database and for extracting apertures and finally for reviewing the spectrum. You will then

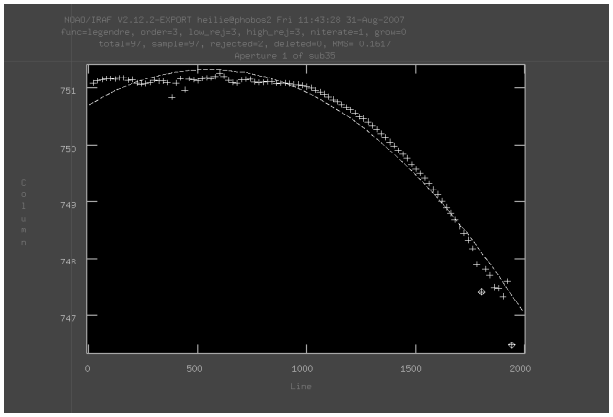

Figure 10: Tracing plot.

see your spectrum. Since the wavelength calibration has not been done yet you will see only pixel numbers on the x-axis and red is on the left and blue is on the right. Our result can be seen in figure 11. Press `q` to exit.

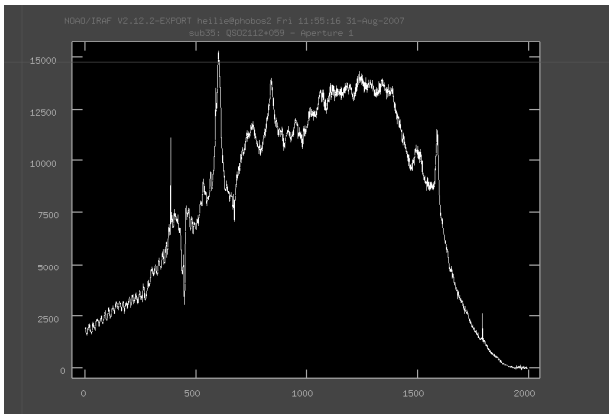

Figure 11: Spectrum extracted with `apall`.

Extract also the calibration lamp spectrum. You can use the same parameters for `apall` as for the target spectrum. Change the input name to `red36` and output name `361d`, set the parameter `referen = sub35` (your science image name before transforming it into a one-dimensional spectrum) and set `recente = no` and `trace = no`. Choose the spectrum somewhere in the center of the first plot, press `q` and answer `yes` four times.

6 Wavelength calibration

6.1 Identify

Now it is time to start the wavelength calibration. First task is to find the correct wavelengths for the emission lines of the calibration lamp. First find the correct calibration spectra for the lamps you have used. For example for NOT they are at http://geena.not.iac.es/instruments/alfosc/lamps/. You may have used two lamps, like helium and neon, and then you have lines from both in your lamp spectrum. Make sure you pick the right grism too. If you don't have a calibration lamp spectrum for some reason you can use the sky background lines in your target image for calibration.

The command to use here is `identify` (in `kpnoslit`). Write the name of the one-dimensional lamp spectrum as `images` (here `361d`). Set `functio = spline3` and `order = 1` for now. You will change these later to see if some other function with some other order is better, but this combination should be a good start. Type `:go` and you will see the calibration spectrum. Now you should mark some lines and give their real wavelengths. This may be a bit difficult at first because your lamp spectrum is a reflection of the true spectrum. Start with finding at least five lines on different parts of the spectrum. Mark a line by placing the pointer on the line and pressing `m`. Then type the wavelength of the line you have chosen. In figure 12 you can see the helium–neon spectrum with nine lines marked.

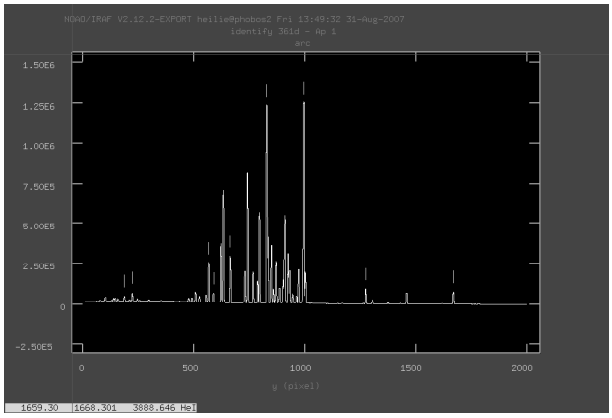

Figure 12: Lines marked for identifying.

After finding and marking some lines press `f`. You will see a plot of residuals. The points should be spread equally on both sides of the line at 0, like in figure 13. If one of the points is very far from the others you should delete it. Press `q` to get back on the line selection screen. Now

the x-axis has changed from pixels into wavelengths and the spectrum has turned around. Find and mark more lines and fit again with `f`. The program should now guess which lines you choose, and you can also mark all the lines automatically with `l`. If however some of the lines have been marked wrong the program might also guess others wrong, and therefore it is always good to check enough of lines yourself.

If the points in the residual plot are not distributed randomly but form some kind of curve or if their residuals are high, change function by typing `:func legendre` and order for example `:order 4`, and fit again. Usually good plots can be made with function spline3, order 1 or 2, or with function legendre, order 3 or 4. When you think you have made the residuals low and random enough quit with `q`, and answer `yes` to the question about writing the data to database.

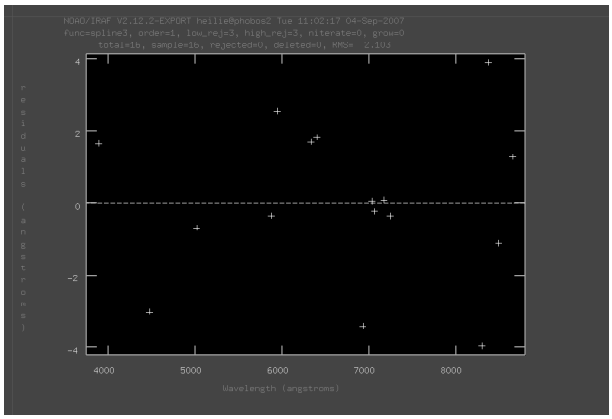

Figure 13: Plot of residuals in `identify`.

6.2 Dispcor

Now you have the information needed for calibration in the database. First you have to add the source of calibration information to the header of the target spectrum. Use the command `hedit`: `images = 351d` (the name of the target spectrum), `fields = refspec1`, `value = 361d` (the name of the calibration spectrum) and `add = yes` (you want to add something new in the header).

After doing this run `dispcor` with `input = 351d` (the input name) and `output = spec35` (an output name). Other parameters can be the default ones.

After running `dispcor` the wavelength calibration should be finished. Type the command `splot spec35` and answer `1` to image band to plot.

This image band question is there because there can be several spectra in one file. You will now see the spectrum. You can zoom in on some part of the spectrum by placing the pointer at the left edge of the wanted area, pressing `a`, and then placing the pointer at the right edge of the wanted area and pressing `a` again. To measure the wavelength of a line set the pointer at the beginning of the line, press `k`, set the pointer at the end of the line, and press `k` again. The spectrum in our example can be seen in figure 14.

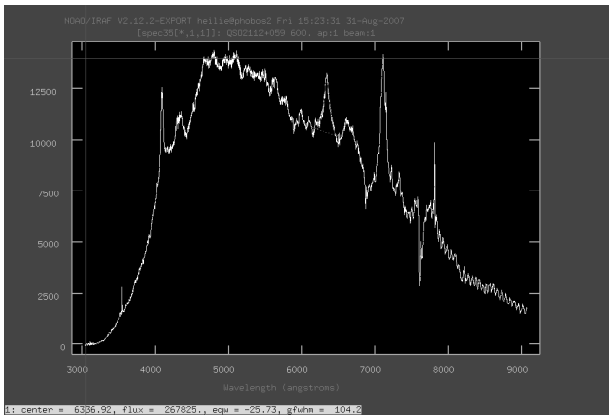

Figure 14: Spectrum after the wavelength calibration is done. An emission line has been measured at 6337 Å.

7 Flux calibration

7.1 Standard star

To correct the shape of your spectrum like the one in figure 14 you have to do the flux calibration using a standard star spectrum. You should have observed a standard star with the same instrument setup as your actual target. Extract and calibrate this standard star spectrum in the same way as the target spectrum. The spectrum of our standard star Feige 92 is shown in figure 15.

Use the command `standard` in `kpnoslit`. Set the following parameters:

- `input = Feige92_spectrum` The name of your extracted standard star spectrum.
- `output = std` Output name.
- `star_nam = feige92` The name of your standard star. IRAF has a database of the standard stars and it should find the star by its name.

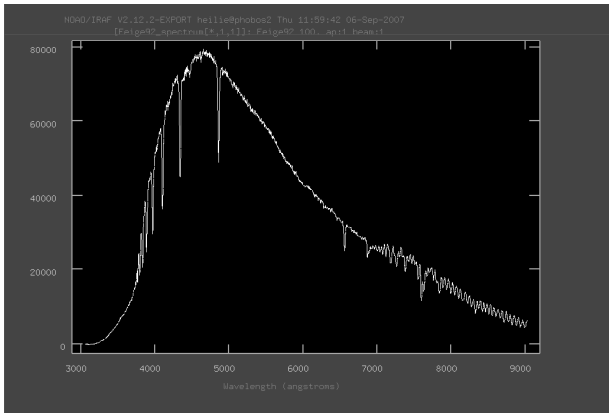

Figure 15: Spectrum of the standard star Feige 92.

If the star is not found you have to search for the flux data and save it on your computer so that the program can use it.

- `caldir = )_.caldir` If your standard data is in your working directory, leave this empty. Otherwise this defines where the data is. The default parameter tells the program to find the information by using the parameter defined in the `kpnoslit` package.
- `airmass = 1.34` Airmass of your standard star observation. Can be usually found in the image header or in the observing log.
- `exptime = 100` Exposure time of your standard star observation.

When you run this you may get an error message of not finding the data. In this case find out where the data of your star is by typing `page onedstds$README` and searching the name of the star. Then type the command `epar kpnoslit` and make sure the parameter `caldir` points to the folder where the right star is. For example in the case of Feige 92 it should be `caldir = onedstds$iidscal/`. You can also write this directory in the parameter `caldir` in `standard`. Then try running `standard` again. You don't need to edit bandpasses.

After `standard` run `sensfunc`. The parameters can probably be the default ones, `standard = std` is the output name of task `standard` and `sensitiv = sens` is the name for the new sensitivity function image. You can run the command interactively and delete some points if they are clearly out of the curve.

7.2 Calibrating the target spectrum

Now we are ready to do the flux calibration. Use the command `calibrate`:

- `input = spec35` The name of the target spectrum.
- `output = cal35` A name for the calibrated spectrum.
- `sensiti = sens` The result of `sensfunc`.
- `airmass = 1.08` The airmass of your target observation.
- `exptime = 600` Exposure time.

Run the command, and the calibration is done. The edges of the spectrum may be useless and look terrible, so when you look at the spectrum with `splot` you may have to zoom with `a`.

Our result spectrum is shown in figure 16. There are three strong emission lines, MgII at $\lambda = 4087$ Å, Hγ at $\lambda = 6338$ Å, and Hβ at $\lambda = 7106$ Å. These give the quasar redshift $z = 0.461$.

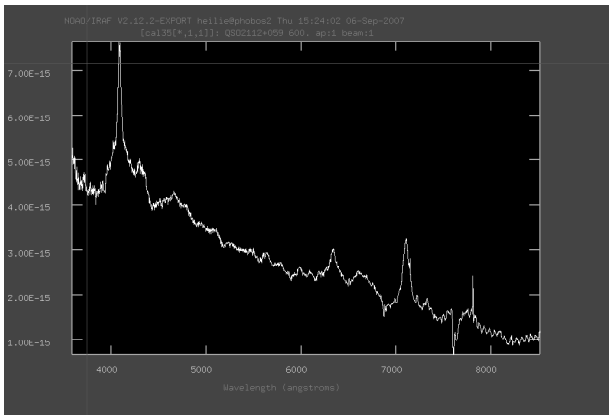

Figure 16: Flux-calibrated spectrum of the quasar.