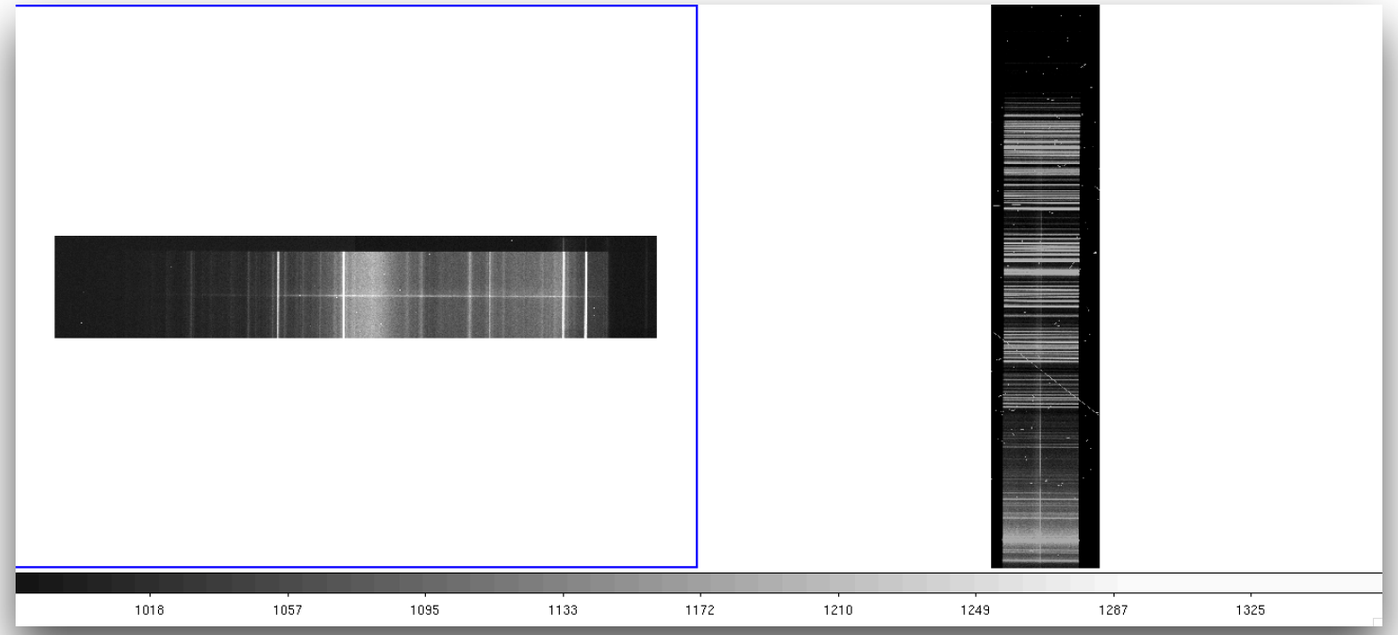
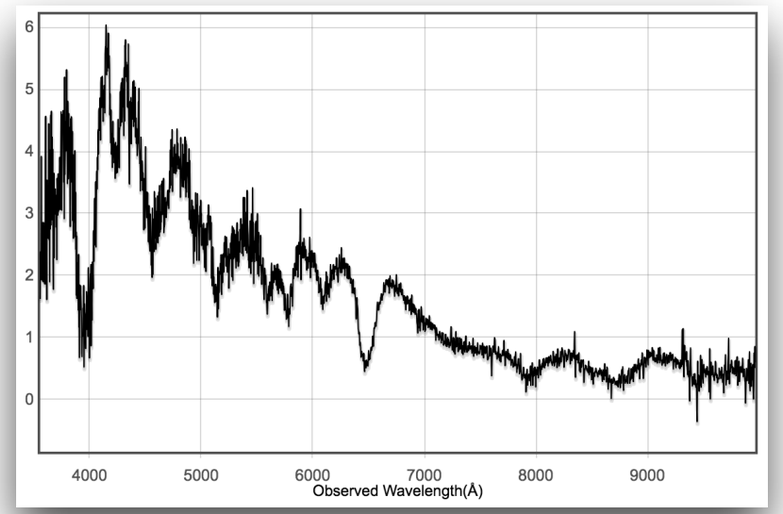
An intro to the quick look spectral pipeline

How to go from this:



To this:



How to install it

- Install astroconda, via anaconda. More info on https://astroconda.readthedocs.io/en/latest/
- Create two python environments (name them what you like):
    - iraf27: This environment will include pyraf, a command language for running IRAF tasks with python. Pyraf only works with python 2
    - astroconda: This environment uses python 3
- Clone the repo:
    - git clone https://github.com/msiebert1/UCSC_spectral_pipeline.git
- In your bash_profile (or whatever you use), add the following lines:
    - export UCSC_SPECPIPE=<the new directory just created by git>
    - export PATH=$UCSC_SPECPIPE/spectral_reduction:$PATH
- Copy the disp.cl file (located in the extra_files folder) into your iraf folder (this was hopefully created when you installed astroconda). If you cannot find it, copy the disp.cl file into ~/iraf and run mkiraf in that directory.
    - If you cannot make iraf(pyraf) work on your machine, ask someone who knows
- At the end of your iraf login.cl file, add the following line with the appropriate path:
    - task disp=‘<your_iraf_directory>/disp.cl'

Description of the files

- Spectral_reduction. This includes the scripts + functions, used in the pipeline. There may be some left-over functions from other codes, or functions that are not called at all. In time, this will be corrected. Moreover, there is a folder called trunk, where the archival files, used in the pipeline are included.

- Extra_files. This includes 2 plots with emission arc lines for KAST, for your convenience, and the disp.cl file, for which we will mention later on.

- Test_data. Contains arcs (b1003, r1003), flats (b1020-b1030, r1020-r1030), and science frames (b1064, r1073-r1075).

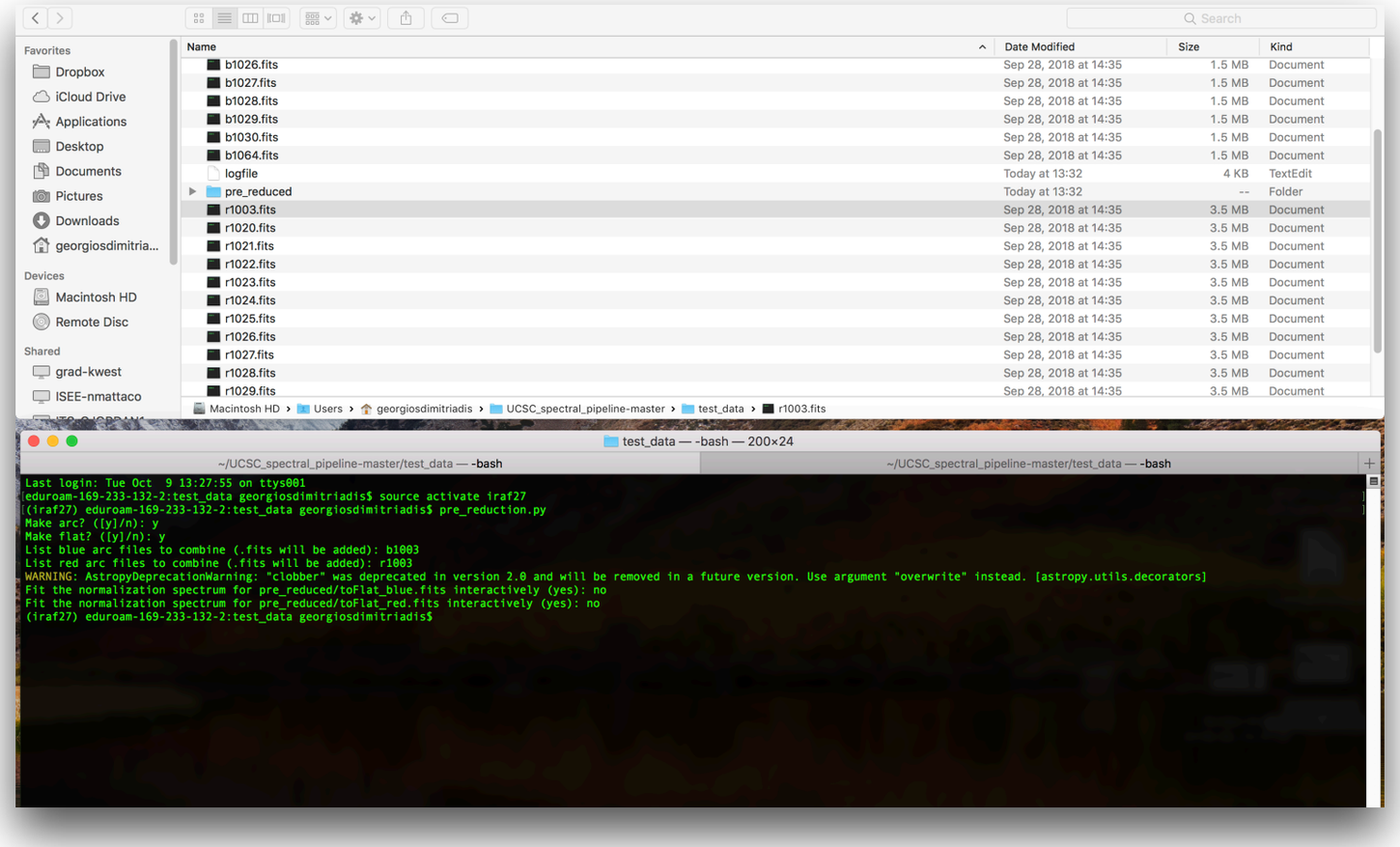
The 3 steps

- **Pre-reduction** (with the iraf27 environment)
    - The script is pre_reduction.py
    - Overscan correction, trimming, master arcs, master flats, and object folder organizing
    - A "pre_reduced" directory is created containing these new calibration files and folders for individual targets
- **Extraction** (with the iraf27 environment)
    - The script is QUICKLOOK.py
    - Recommended syntax: QUICKLOOK.py -i -a -c
        - -i means interactive
        - -a means that we will use the reidentify iraf task to calculate a zero-order shift of the master arc wavelength solution with our night’s arcs
        - -c triggers the python implementation of LACOS for cosmic ray removal
    - Extracts a spectrum using the apall task from IRAF
    - Maps the extracted spectrum with the calculated wavelength solution
    - The procedure is done for both blue and red arm of the spectrograph
        - If more than one red exposures are provided (usually that’s the case), the images are combined prior to reduction
    - Creates "target_ex" directory containing the d*_ex.fits file used in flux calibration
- **Flux Calibration** (with the astroconda environment)
    - The scripts are headerfix.py, cal.py, wombat.py
    - Flux calibrates and telluric corrects the extracted spectrum
    - Creates 2 fits files, one for each arm of the spectrograph

An example with our test data - SN 2018gdg

- The code will ask us some questions. Normally, there is a suggested answer
- We will start in the folder that contains the test data and a terminal with 2 tabs, one with iraf27 and one with astroconda
- We start with **pre_reduction.py**
    - Make arc? Answer Y
    - Make flat? Answer Y
    - List blue/red arc files to combine
        - Here, we need to provide the names of the blue and red arc files of the night. For the test data, first we answer b1003 and then r1003
    - Fit the normalization spectrum for pre_reduced/toFlat_blue.fits interactively? This will fit the normalization (flat spectrum) to create the response function. The default answer is yes, but for the purpose of the quick look, no is fine. The default fit is using Legendre functions of order 60 (blue) and 80 (red)

An example with our test data - SN 2018gdg

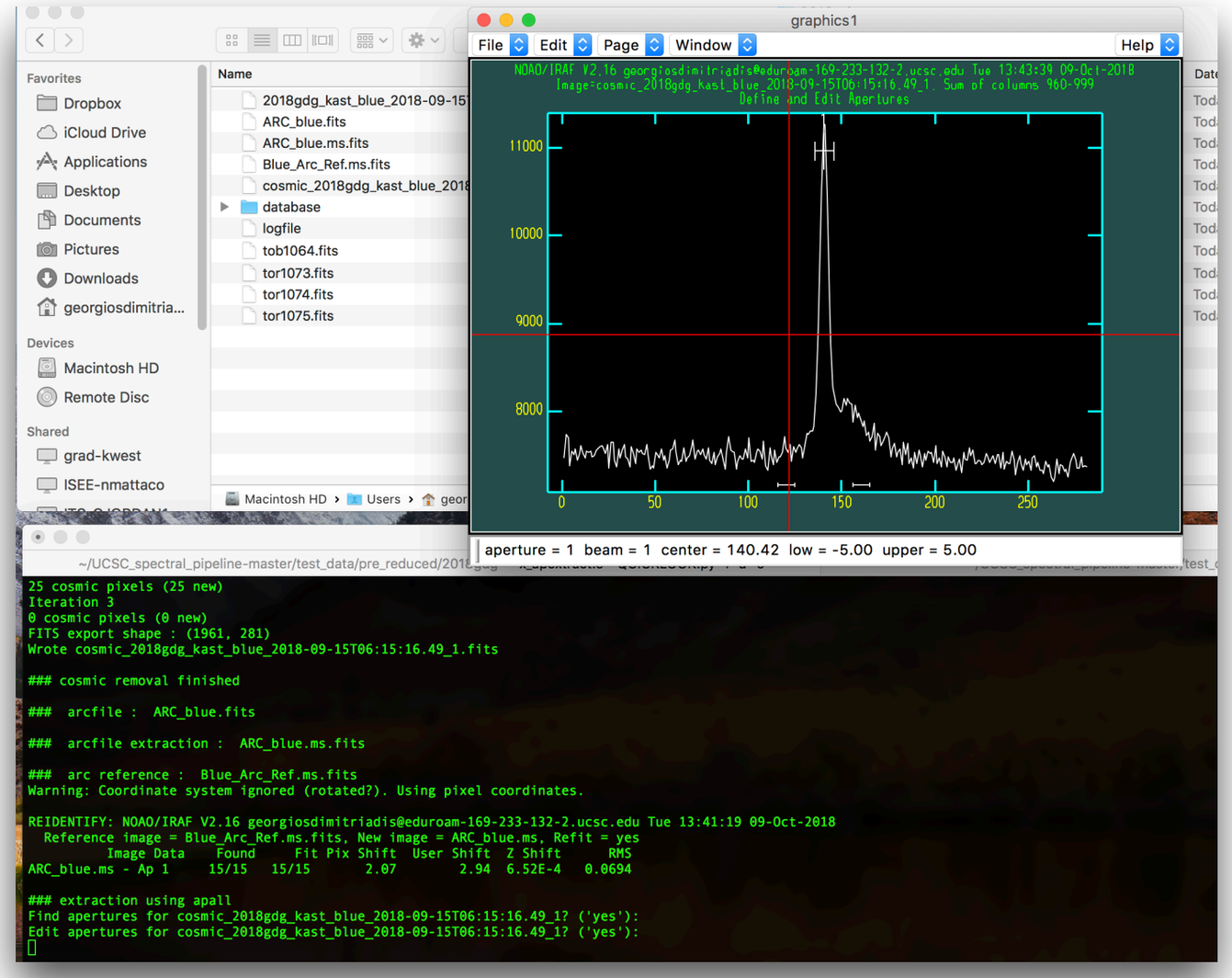


- The pre_reduction.py script created a folder called pre_reduced. We move to that folder and we see a folder called 2018gdg and 4 fits files: The master files of Arcs and response, for each arm

- We move to the target directory and we see the 4 science fits files, with the prefixes t (trimmed) and o(overscan corrected)

An example with our test data - SN 2018gdg

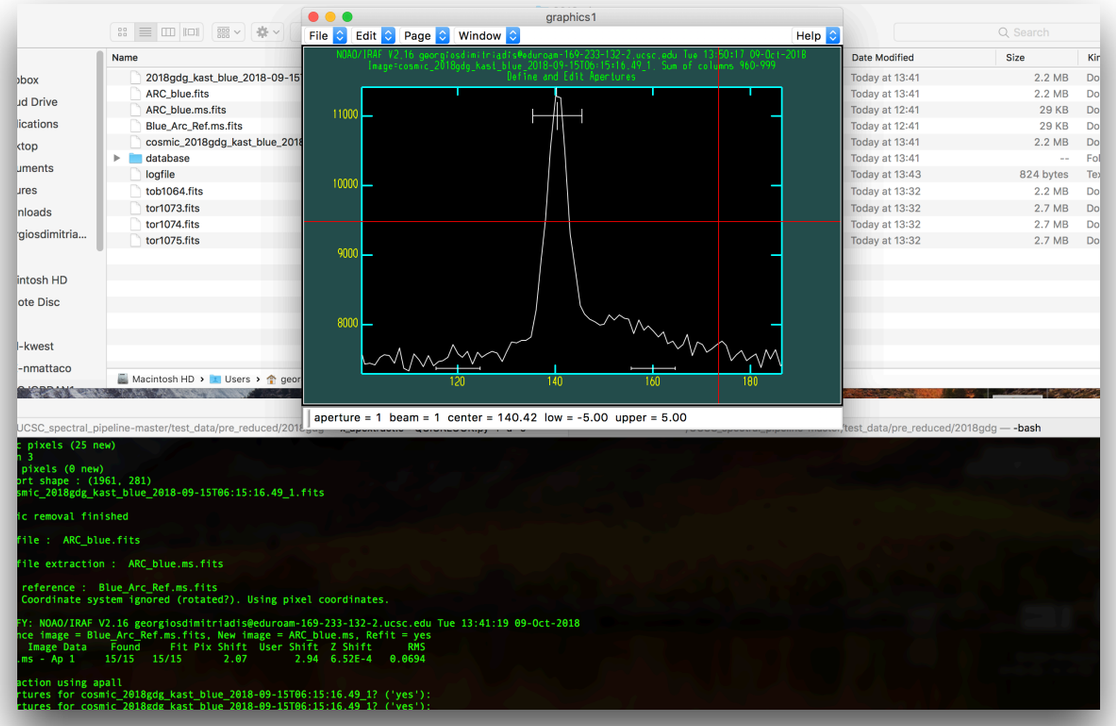
- We move to the spectrum extraction. Still within the iraf27 environment, we type **QUICKLOOK.py** -i -a -c
- Reduce which side? Answer both
- The cosmic ray removal starts and finishes, and the reidentify task found a shift of ~ 2 pixels when comparing the archival arc with tonight's arc
- Then, the apall task starts
  - Find apertures for … Answer yes (with this apall attempts to find automatically the trace of the spectrum)
  - Edit apertures for … Answer yes (we want to change the size of the aperture, the background, etch…)
- We end up with something like this:



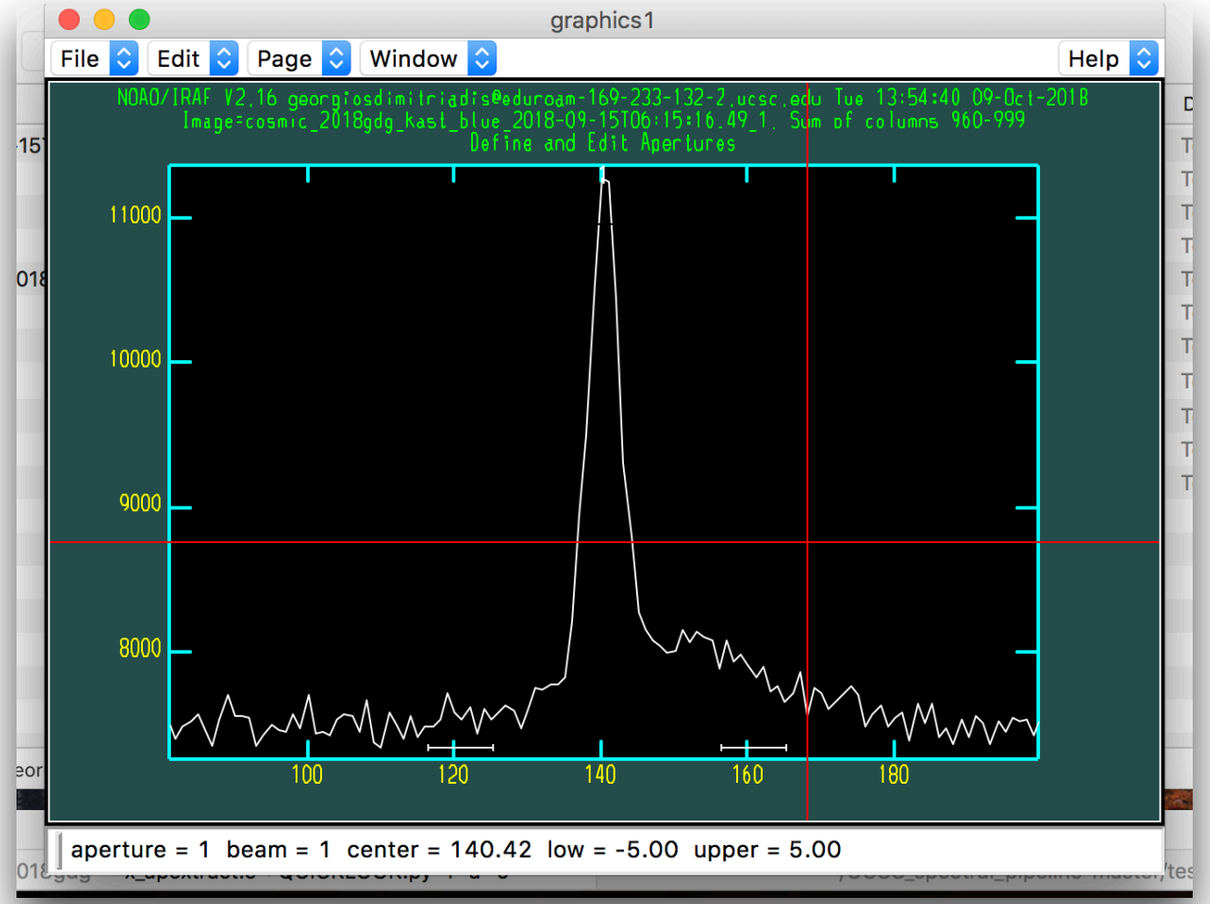
- For a complete tutorial on the iraf apall, see the iraf documentation. Some useful commands are:
  - For zooming, we press "w" and then we press "e" on the 2 corners of our zooming window (bottom left and top right, with this order)
  - We zoom out by pressing "w" and then "a"
  - Lets zoom to our spectrum trace

An example with our test data - SN 2018gdg

- Firstly, we will put the aperture down. In this example, the automated finding of the aperture looks good. We can press “r” to re-plot. Our cursor appears at the center of the aperture.

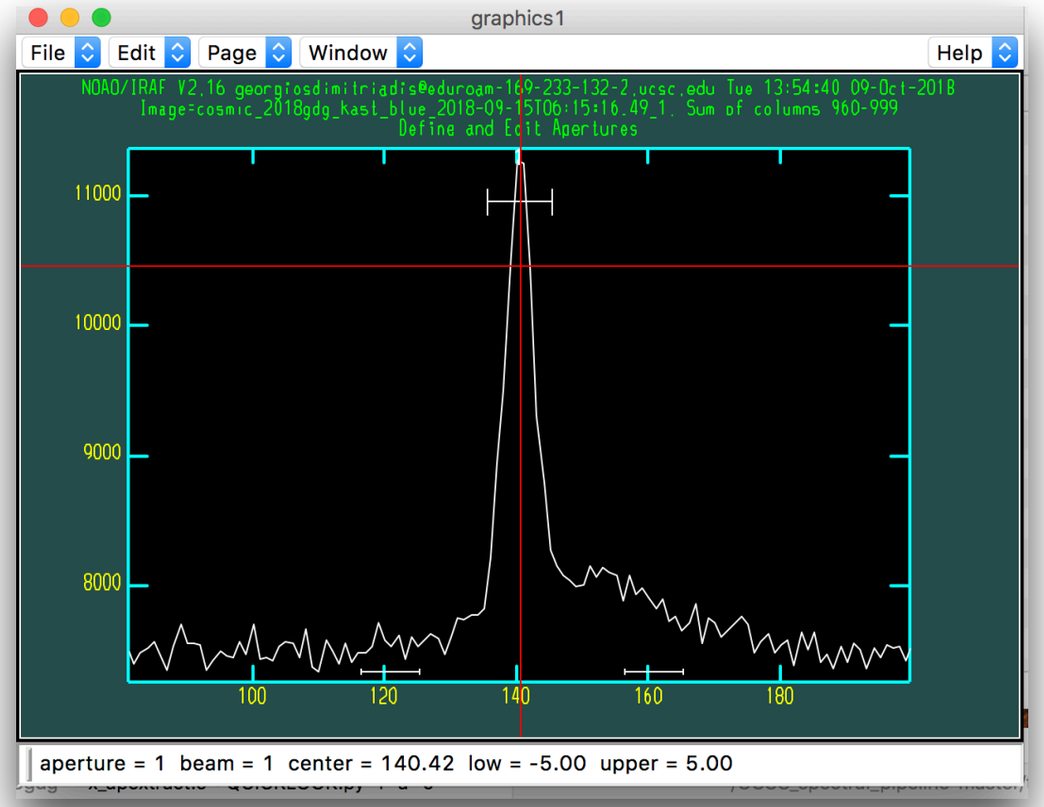


- We can delete out aperture by moving the cursor to the aperture and press “d” (if, for example, we believe that the aperture is not at the correct trace)

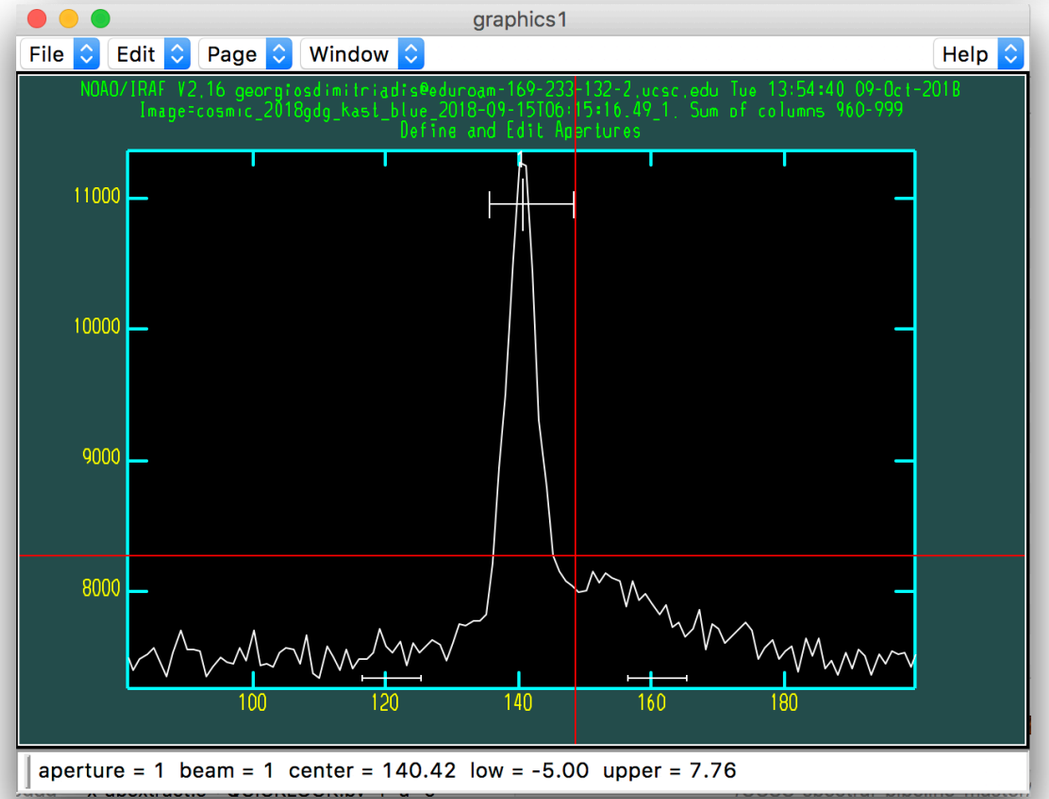


An example with our test data - SN 2018gdg

- We mark a new aperture by going close to the peak of the trace with our cursor and press “m”. Let’s try it for the aperture we deleted at the previous step



- We can change the width of the aperture. For the right side, we move the cursor to the desired position and press “u” (this comes from “upper”) and for the left side we do the same and press “l” (this comes from “lower”)

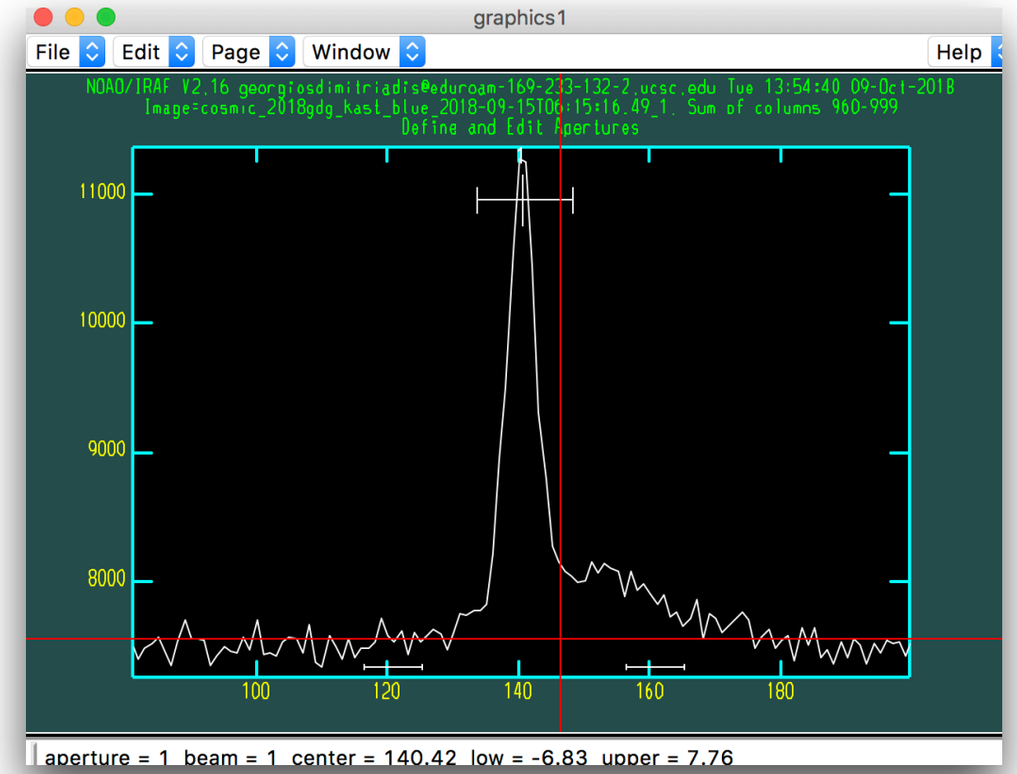


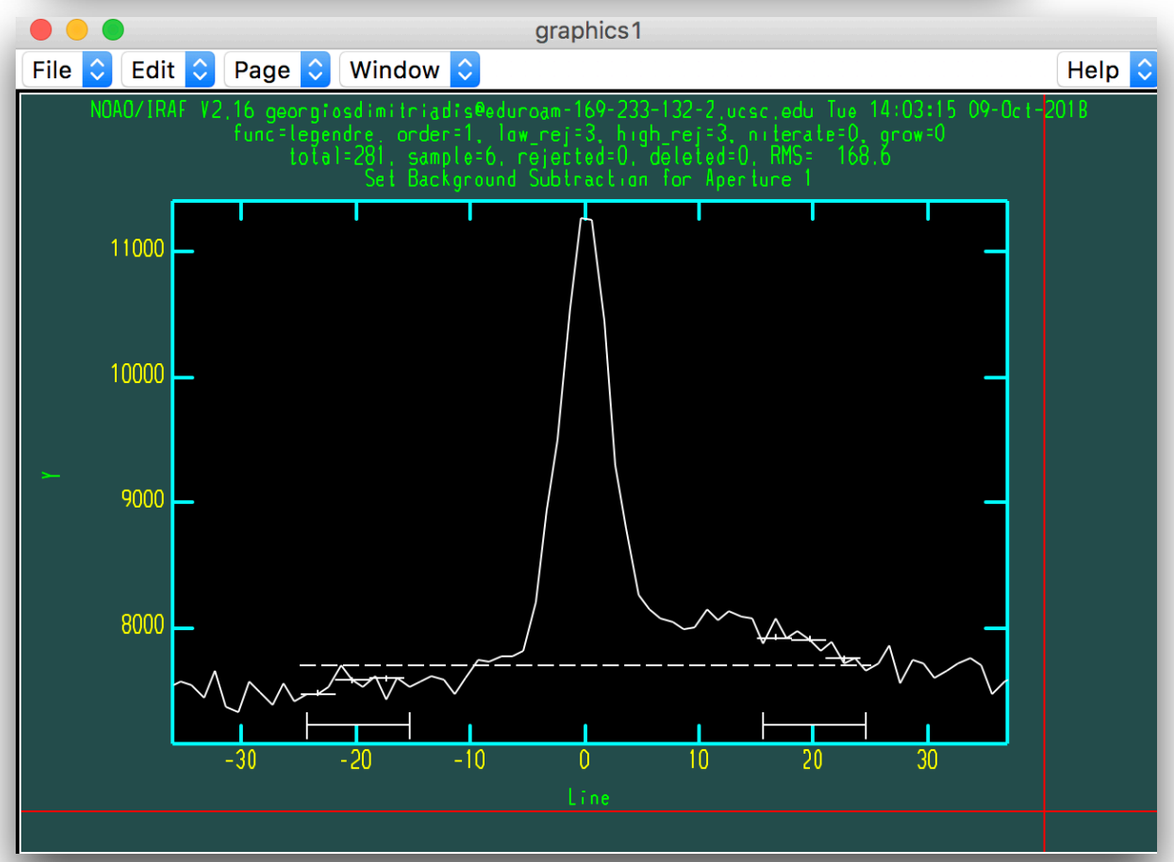
An example with our test data - SN 2018gdg

- When we finish changing the width, we are ready to move to the background fit. The background is indicated by the two horizontal lines at the bottom.

- We enter the background fit module by pressing “b”

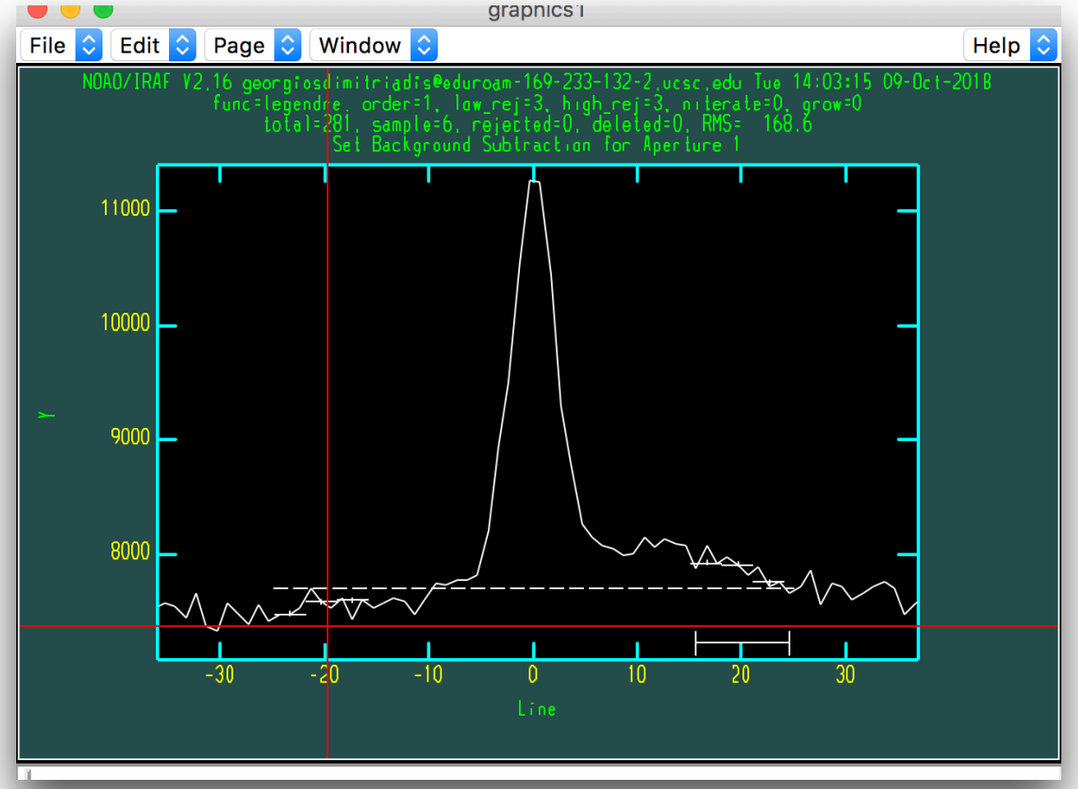
- The background is represented as a dashed line. Currently it’s a straight line fit to the points indicated by the 2 horizontal lines.



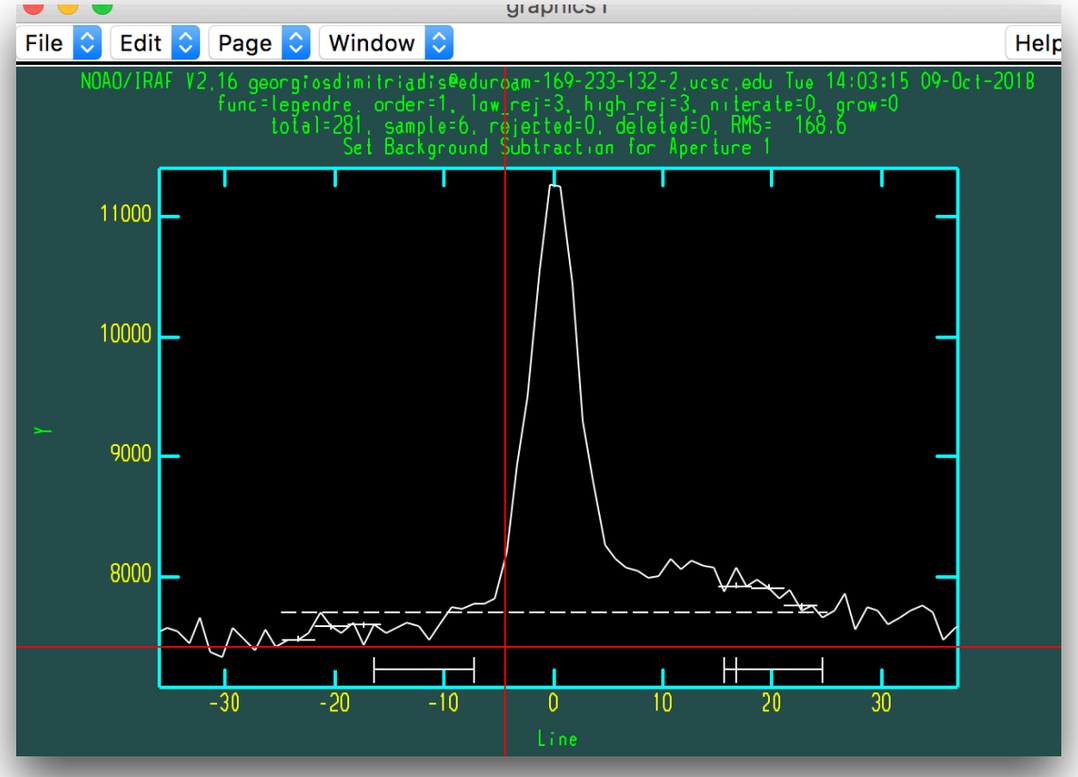


An example with our test data - SN 2018gdg

- We will first change the points we want to include in the fit. We can delete a horizontal line (in this case the left background region) by moving the cursor close to it and pressing “z”



- We will mark a new background region by pressing “s” two times, for the two edges of the back ground region

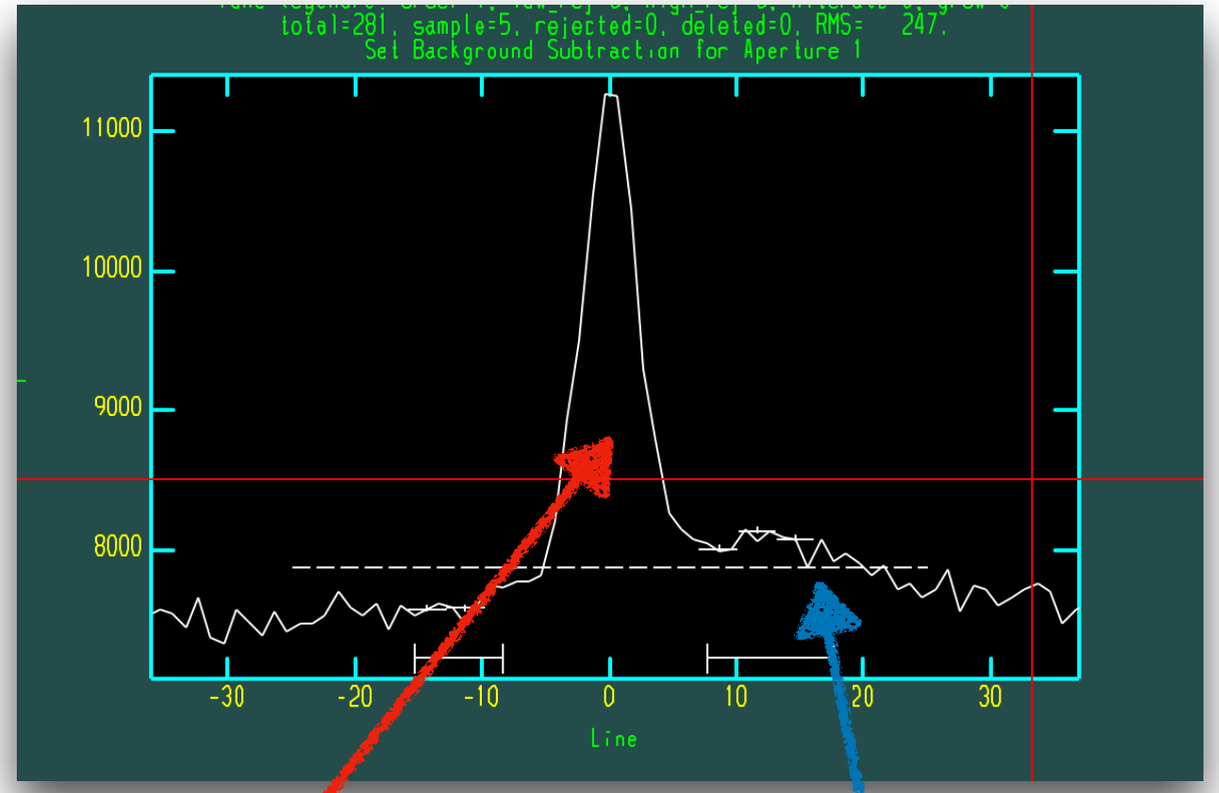


An example with our test data - SN 2018gdg

- We do the same for the right region, and when we are done, we press “f” to fit. The new fit appears (the dashed line moves a little bit up)

- Next, we will attempt to fit a straight line with a slope (this is what we normally do when we want to extract correctly a supernova which is embedded in galaxy light). See the image for what we mean.

- So, ideally we want the background to “represent” the galaxy.

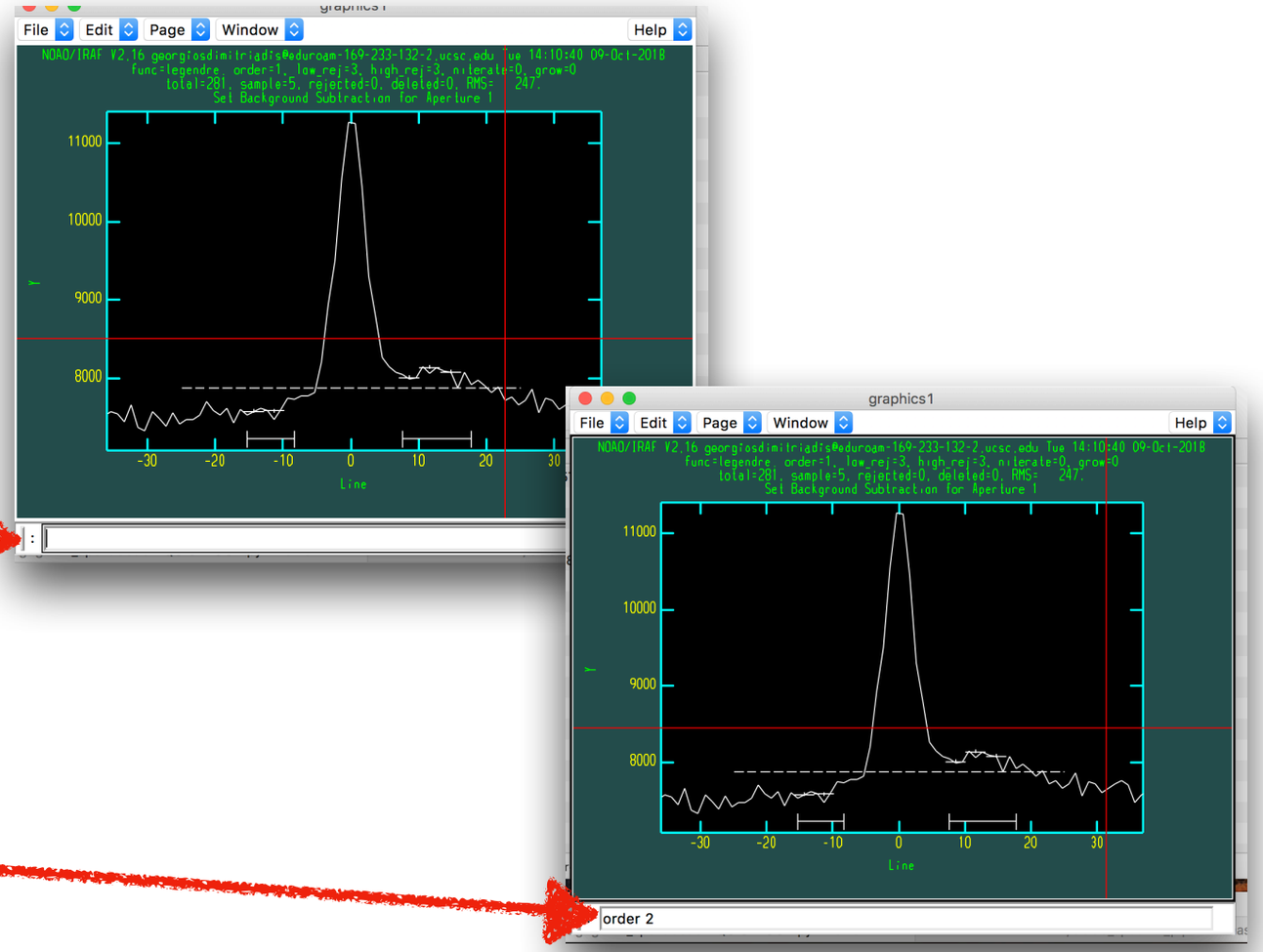


Supernova light

Galaxy light

An example with our test data - SN 2018gdg

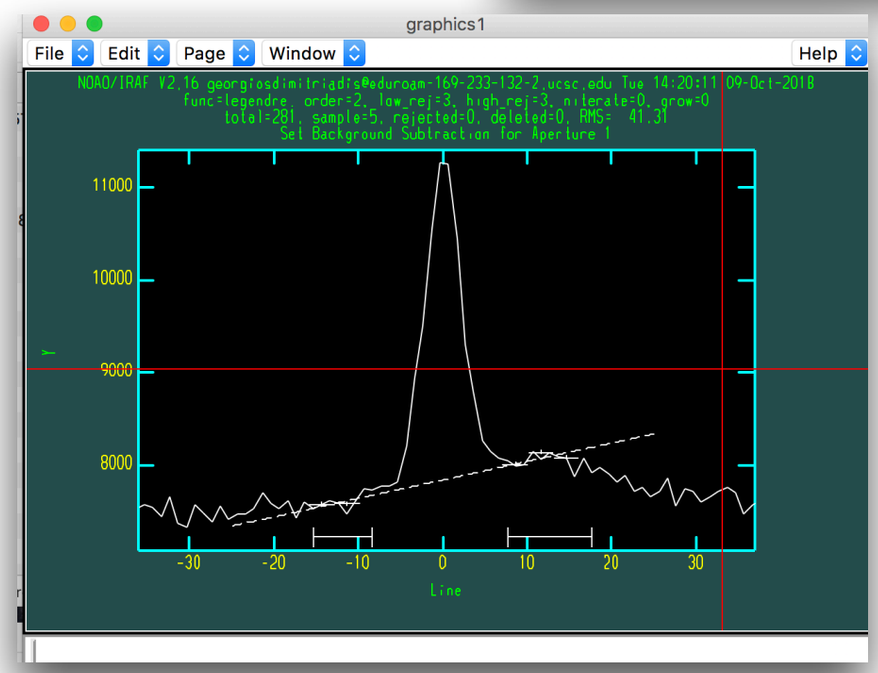
- We press “:” A cursor appears at the bottom of the window



- We type “order 2” and then enter

- We press “f” for re-fitting

- Looks better

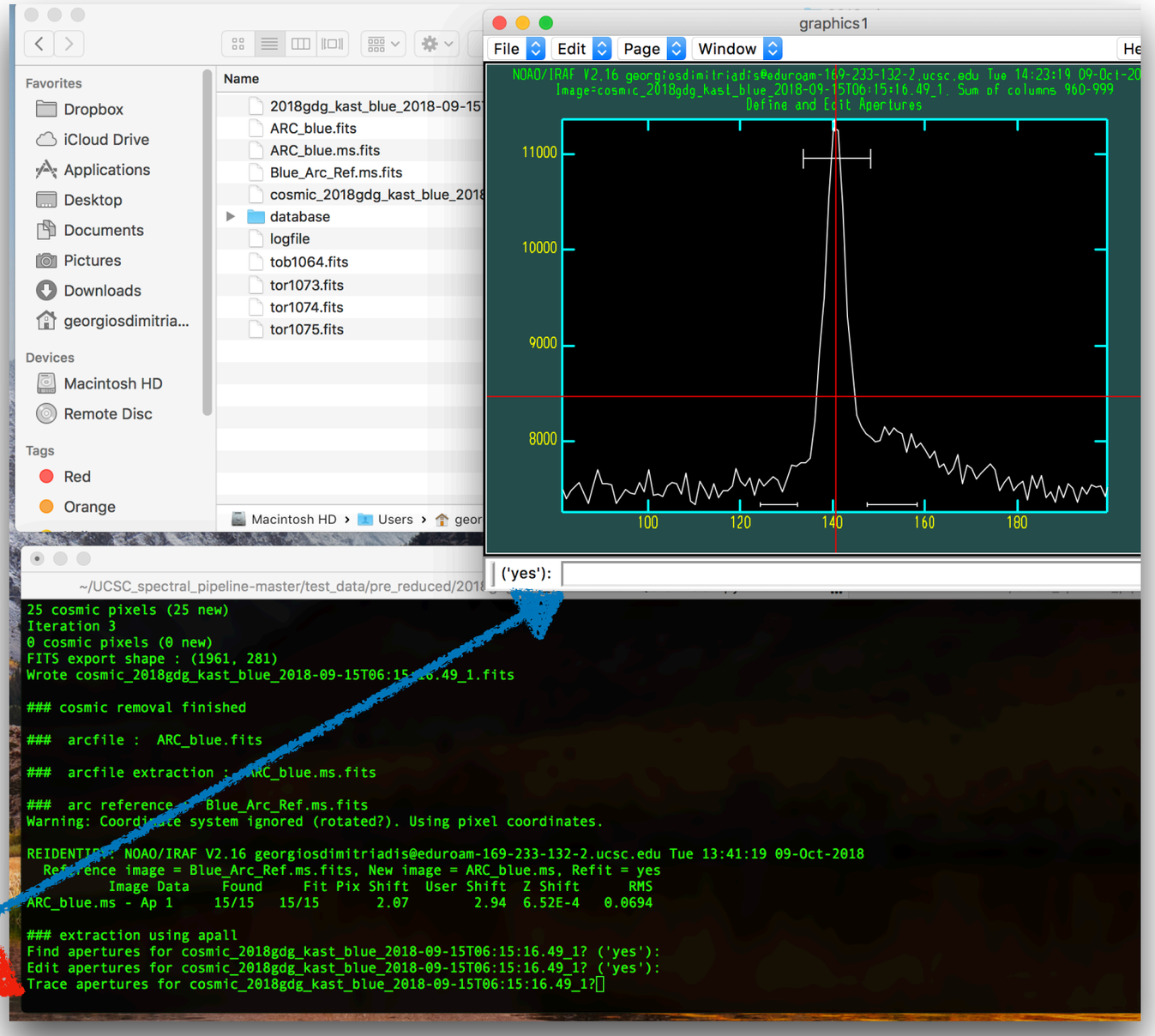


An example with our test data - SN 2018gdg

- We press “q” to quit the background module, returning to the aperture module.

- We press again “q” to start the fit

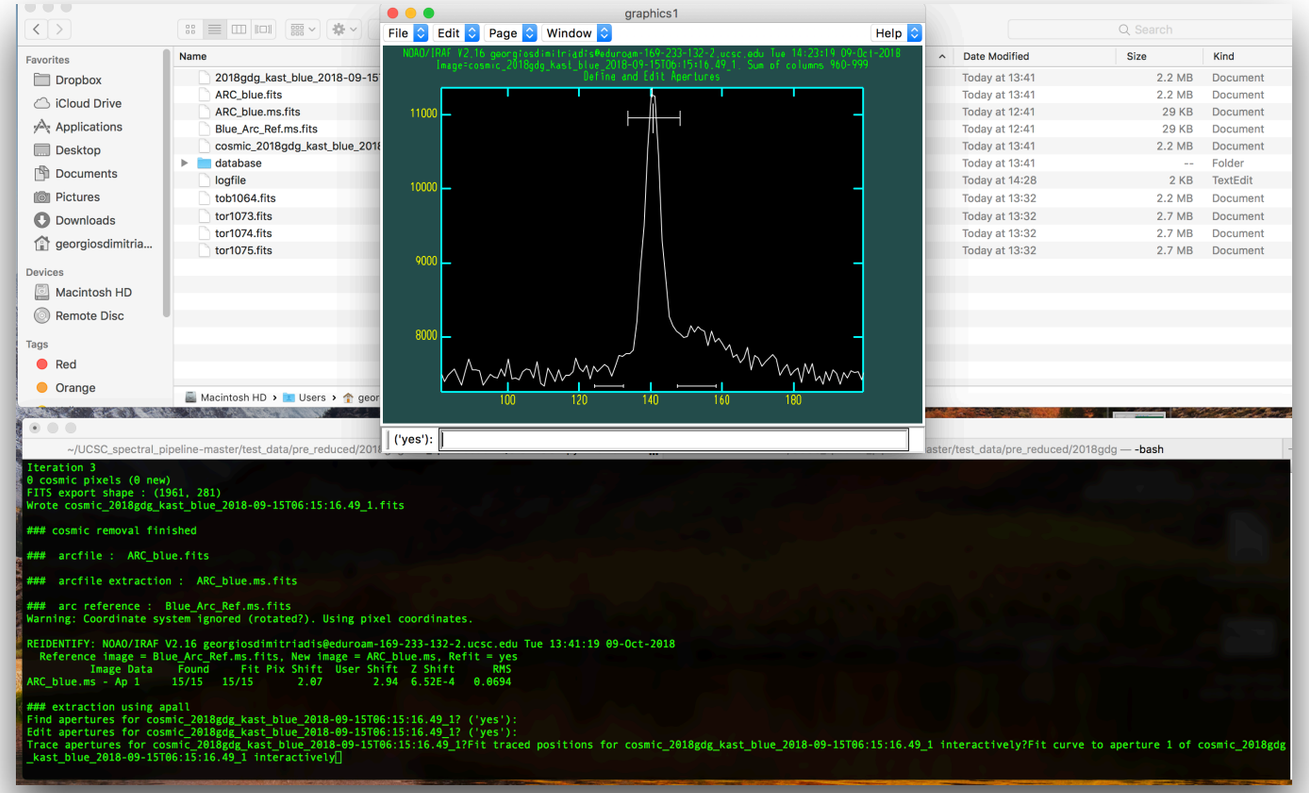
- Again, apall asks us some questions (questions appear at the terminal and answers at the bottom of the apall window)

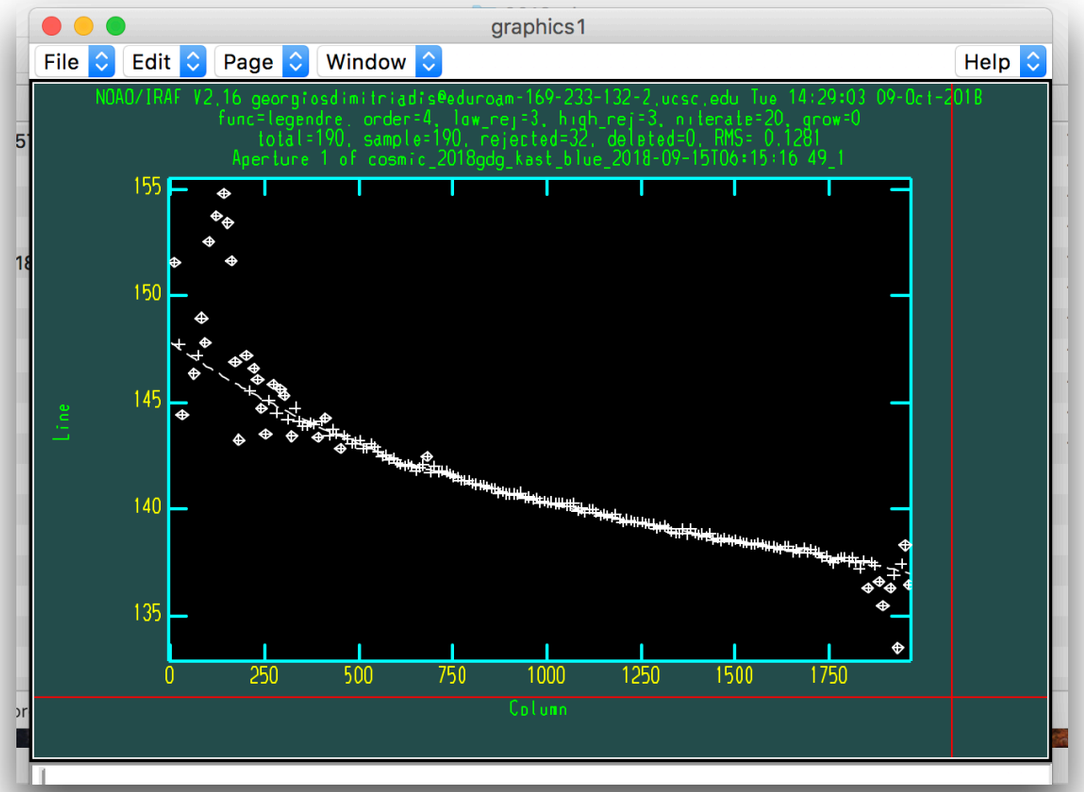


An example with our test data - SN 2018gdg

- So, in order to answer the question, we make sure we answer at the apall window (not the terminal). The cursor must blink. The default answer is “yes”. We simply press enter to all of them.

- We end up in the trace fitting module



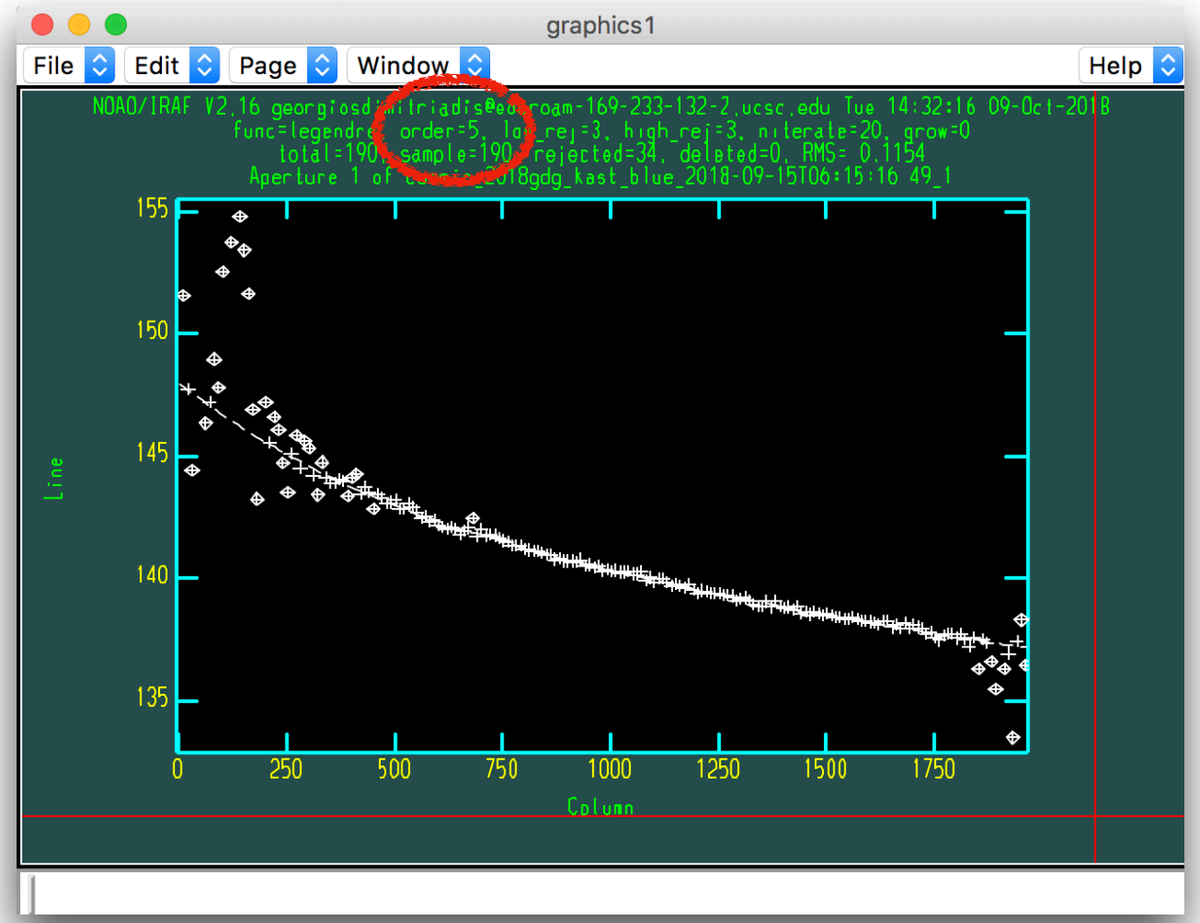


An example with our test data - SN 2018gdg

- The automated fitting routine fit appears as a dashed line. Some info appears at the top of the window (for example, some points were excluded from the fit)

- We can again change the parameters of the fit just like for the aperture. For example, I will increase the order of the fit to 5.

- We can delete points (in order to exclude them from the fit) by moving the cursor close to them and pressing “d”. We can un-delete points by moving the cursor close to them and pressing “u”

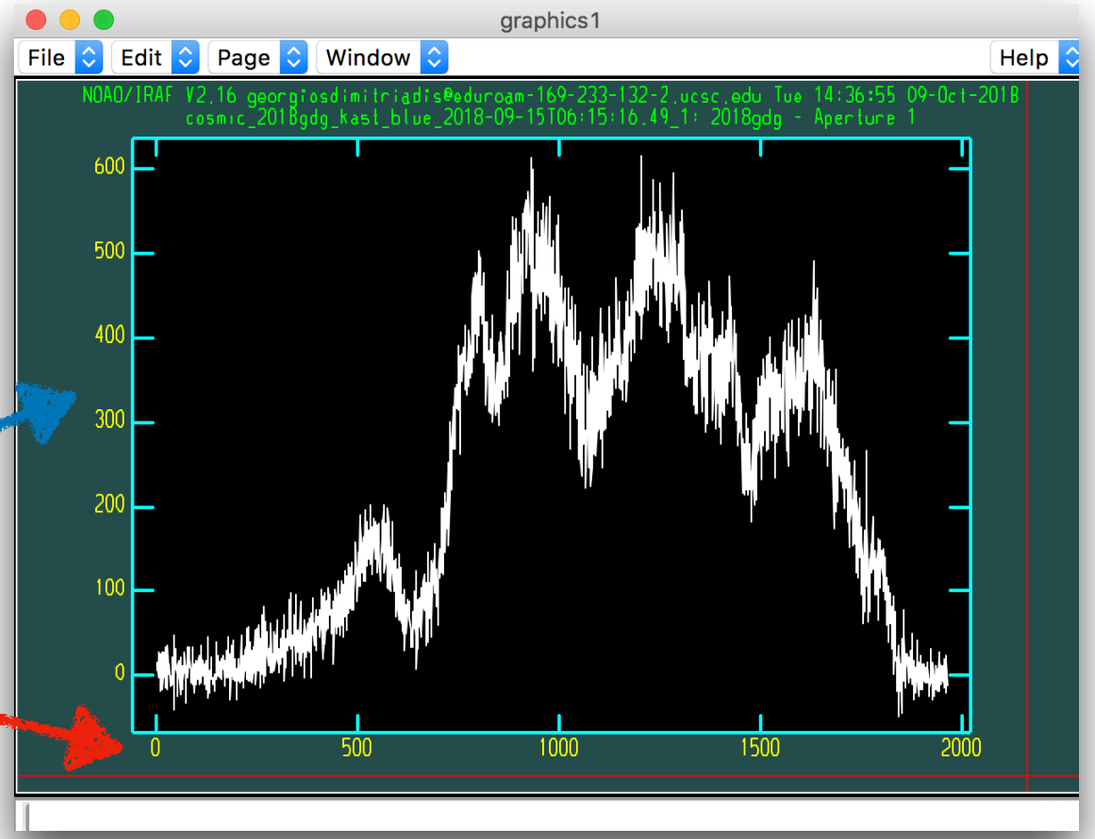


- When we are done, we press “q”

- And we answer “yes”, in the same way as before

An example with our test data - SN 2018gdg

- We end up with the extracted uncalibrated (both in wavelength and flux space) spectrum.

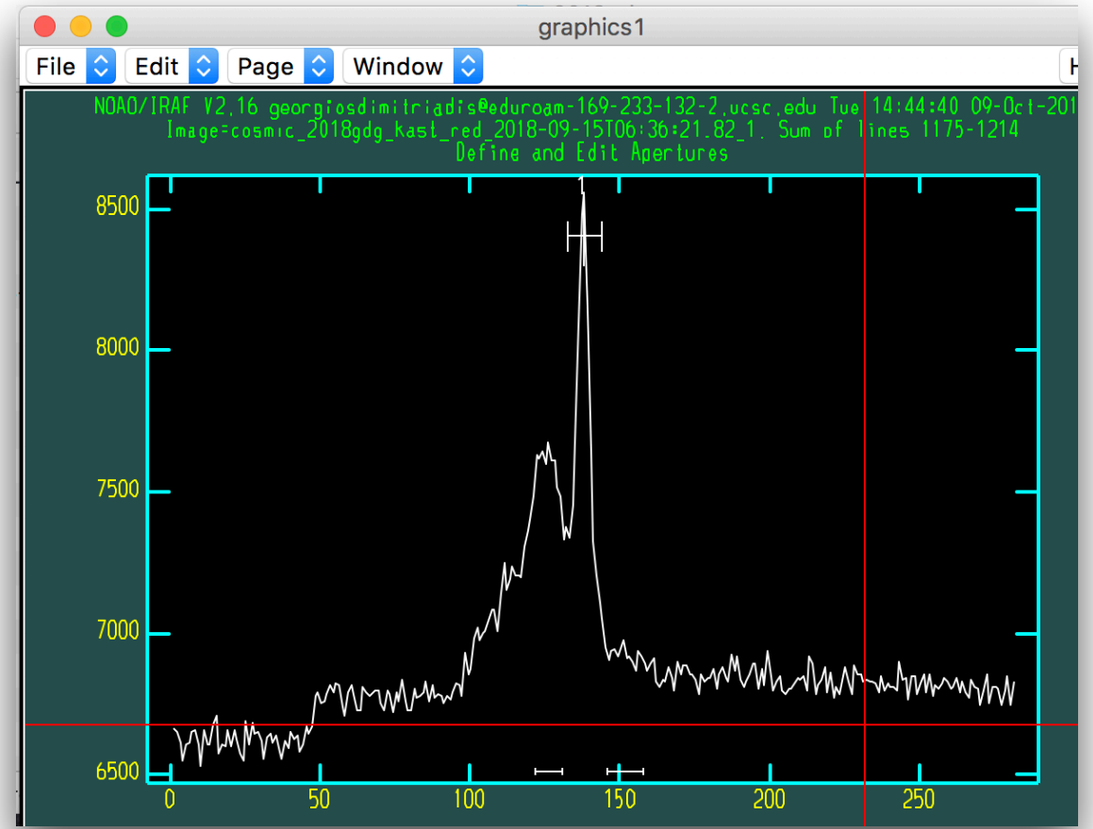


- We press “q” to quit and leave apall. Back to the pipeline.

- Use archival flux calibration? Answer yes

An example with our test data - SN 2018gdg

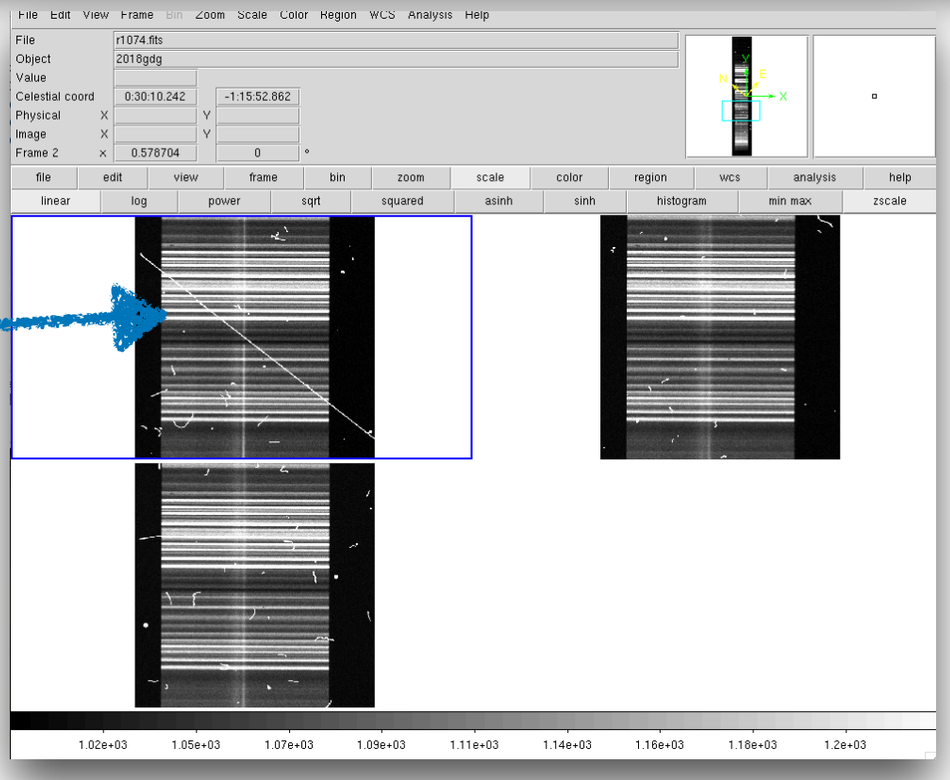
- The blue side is done. The pipeline automatically starts the red side.
- Repeat the same as before
- It does take time to familiarize with how apall works, but after practice, it becomes easy
- There are subtle differences between the two arms, for example, in this case the galaxy is red, so the galaxy light is more at the red arm
- Fit the background accordingly!



An example with our test data - SN 2018gdg

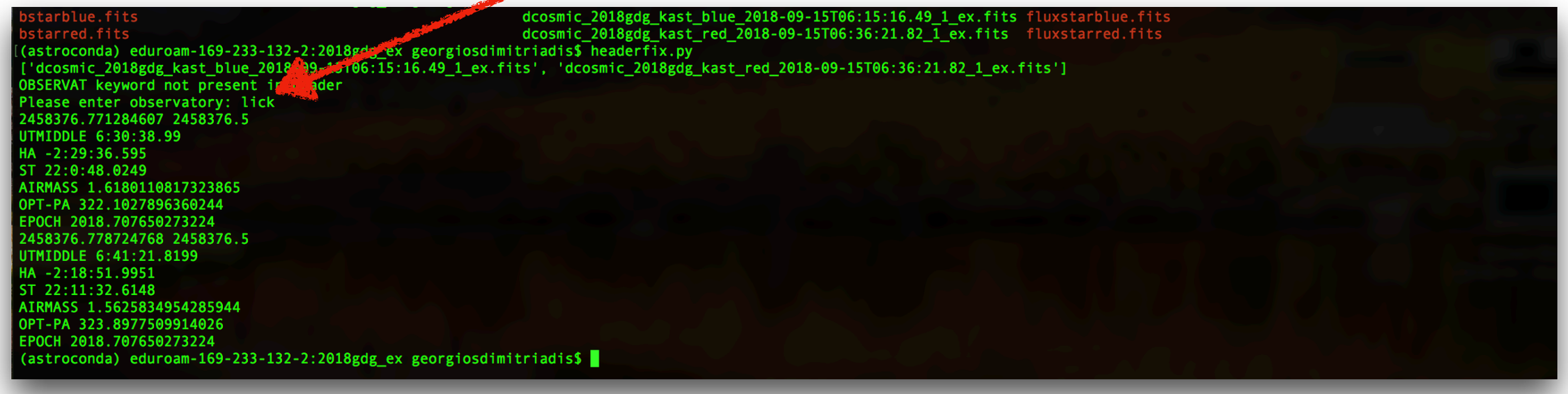
- The red side looks like this.

- Around pixel 1500, there is an issue. There is a huge cosmic ray at the red, that it is not removed correctly



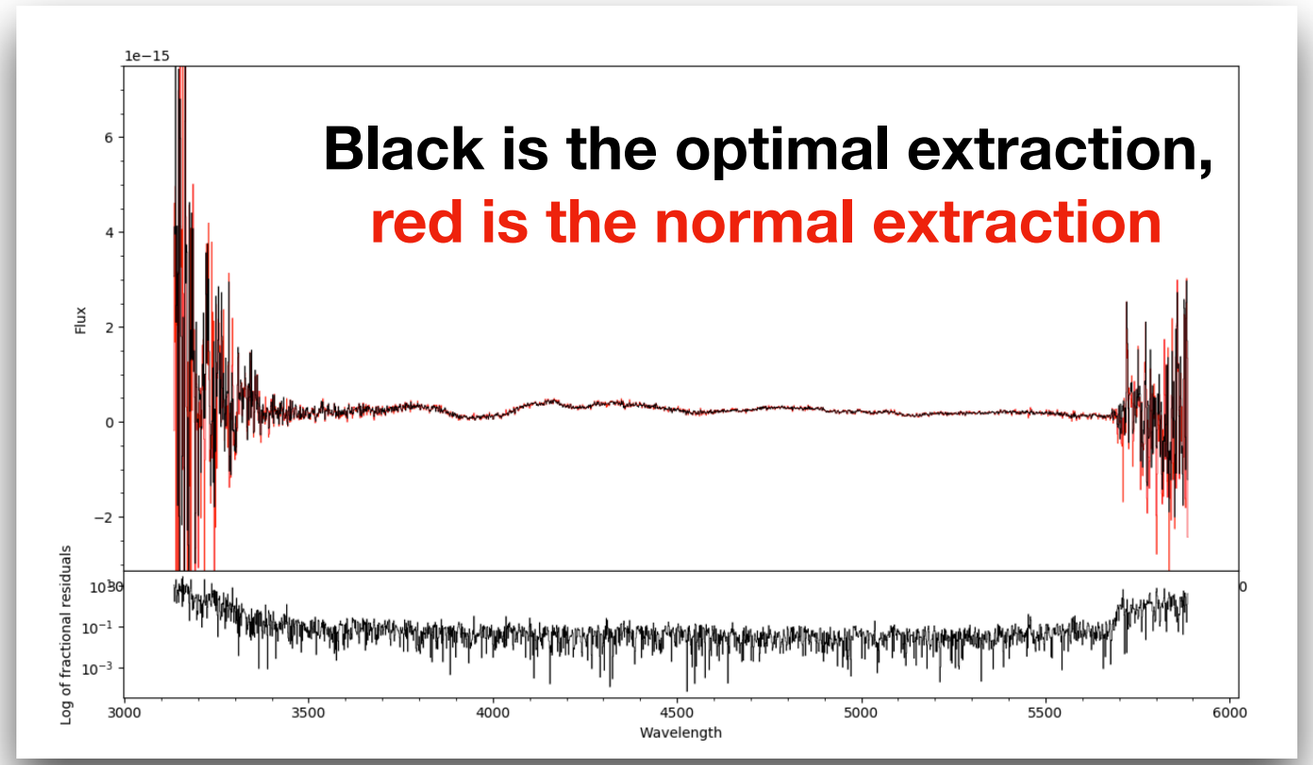
An example with our test data - SN 2018gdg

- The result of the **QUICKLOOK.py** script is a new folder called 2018gdg_ex
- We move to that folder for the flux calibration. It contains the wavelength calibrated 1d spectra of the object (with the prefix “d”) and the 4 files of the archival standard star that cal.py needs
- Now, we switch to the astroconda environment
- We first run headerfix.py
    - This simply adds the information of the observatory to the header
    - We type “lick” when asked



An example with our test data - SN 2018gdg

- Then, at the same directory, we type cal.py
- This routine asks us some questions (again, there are default answers)
    - Do you want to use second-order correction? Answer no
    - Then, it asks the grating code. We type “blue” to do the blue arm flux calibrations
    - Do you want to fit a flux star (this is the standard star)? Default answer is yes. Since we have the archival standard, we answer no
    - Do you want to apply the flux star(s) to the data? Answer yes
    - Do you want to fit a b-star? Default answer is yes, we answer no (we have the b-star in the archive)
    - Final calibration (atmos. removal, sky line wave. adjust, etc.)? Answer yes
- We end up with the wavelength and flux calibrated spectrum

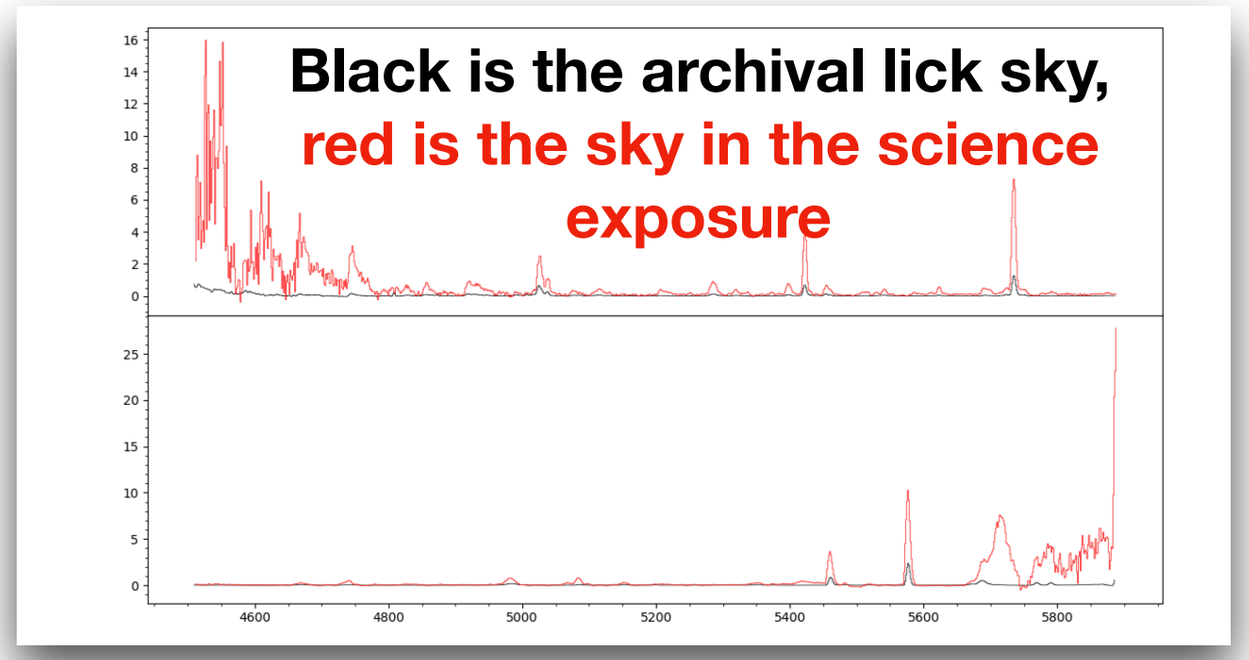


- Back at the terminal, the code asks us to chose the extraction we want. Chose one (say the normal, so type “n”)

An example with our test data - SN 2018gdg

- The code attempts to find a shift of the skylines with the archival lick sky night lines.

- In this case, it found a shift of ~2.5 angstroms. Look at the plot



- The code ask us if the shift is ok. Answer yes

- Then, it plots the spectrum before and after the atmospheric correction. For the case of the blue side (where there are not a lot of tellurics) they don't look that different

- Again, it asks us if it's ok. Answer yes

An example with our test data - SN 2018gdg

- The code then plots the edges of the spectrum

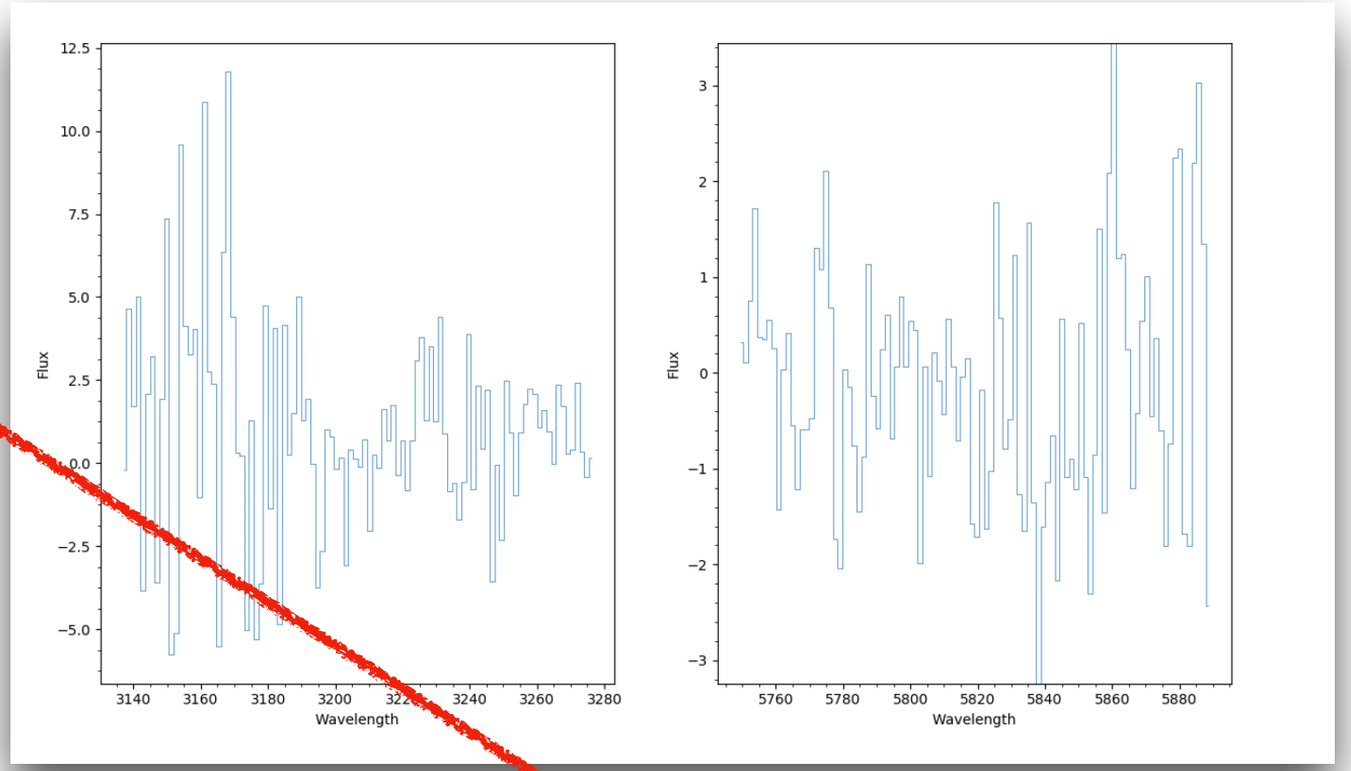
- At the terminal, it prints the current A/pix resolution and asks us if we want to rebin it. If we hit enter, it leaves it as it is. Type 1.4 for 1.4 angstroms

- Then, it prints the wavelength range, and asks you if you want a new one (hit enter if you want the full range). I normally trim the spectrum, but keep in mind that we need enough coverage in the blue side, to merge the 2 arms. Type “3200 5800”

- It finds non integer number of bins and suggests a new range. Answer yes

- Finally it asks you the name of the final file. Type “SN2018gdg_blue”

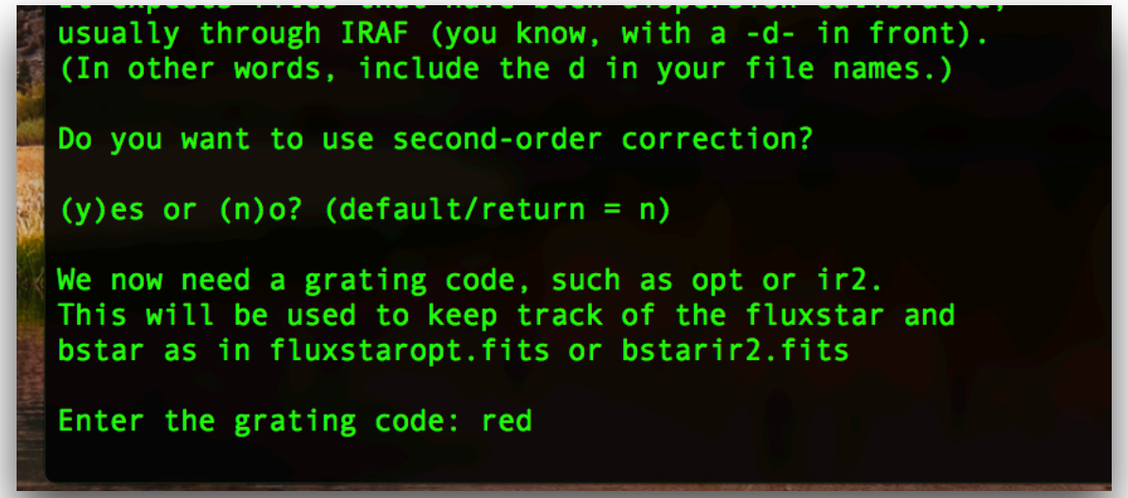
- The code saves the spectrum as a fits file with the addition of the UT time of the observation, in this case SN2018gdg_blue-20180915.fits

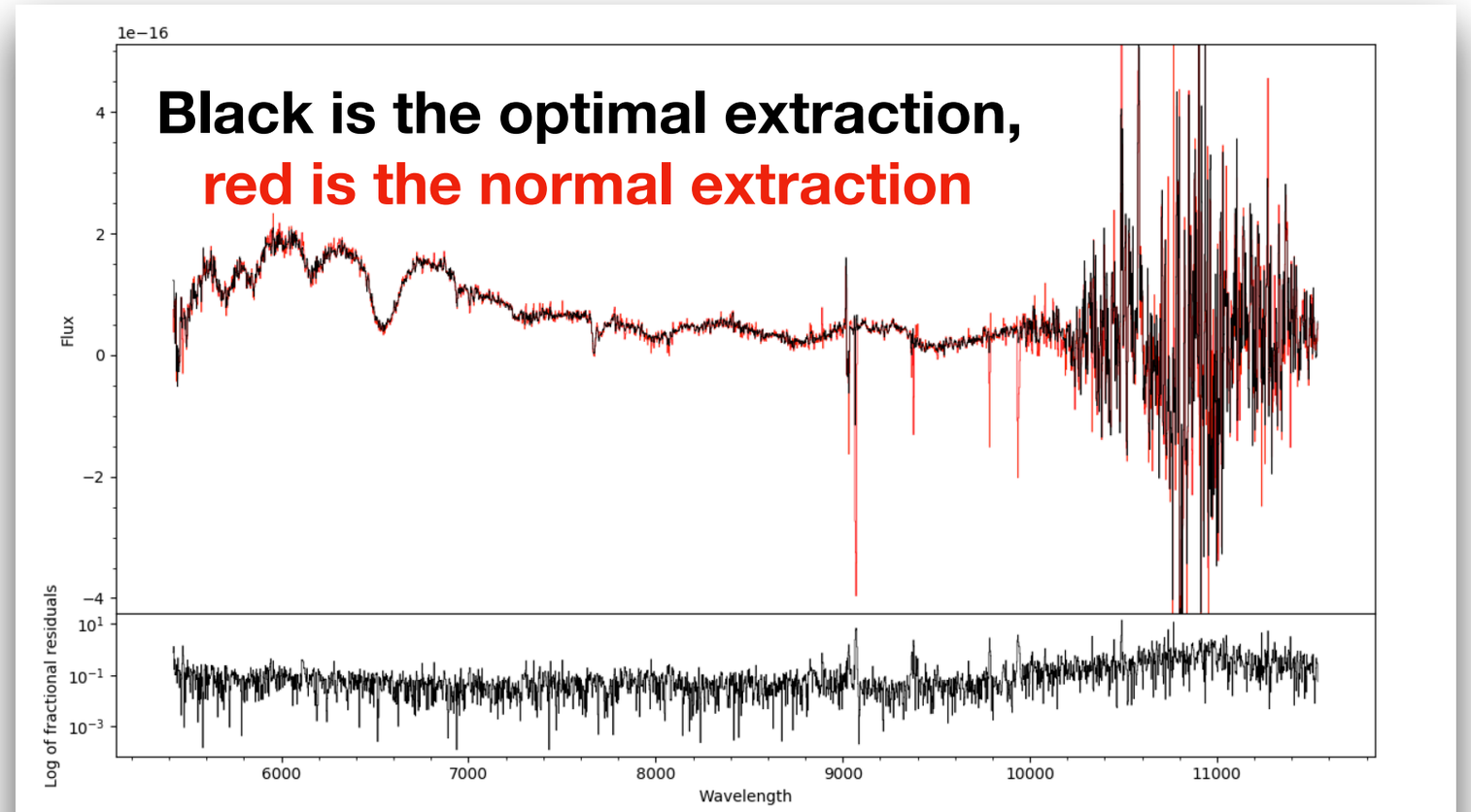



```
Rebin to how many Angstroms per pi
        <CR> selects previous choice: 1.4

Current range: 3137.1602041618175 5888.524889652802
Enter the new wavele   h range desired:
Enter wavelength range: 3200 5800
NON-INTEGER number of bins
Closest match is: 3200.0 to 5801.2
Would you like this wavelength range?
(y)es or (n)o? (default/return = y)

The file is: dcosmic_2018gdg_kast_blue_2018-09-15T06:15:16.49_1_ex.fits
The object is: 2018gdg
The DATE-OBS is: 2018-09-15T06:15:16.49
The aperture is: 1
The previous name was:

Enter the object name for th    nal fits file:
(UT date and .fits will be ad    ): SN2018gdg_blue
final
dcosmic_2018gdg_kast_blue_2018-09-15T06:15:16.49_1 blue False

There, was that so hard?
(astroconda) eduroam-169-233-132-2:2018gdg_ex georgiosdimitriadis$
```

An example with our test data - SN 2018gdg

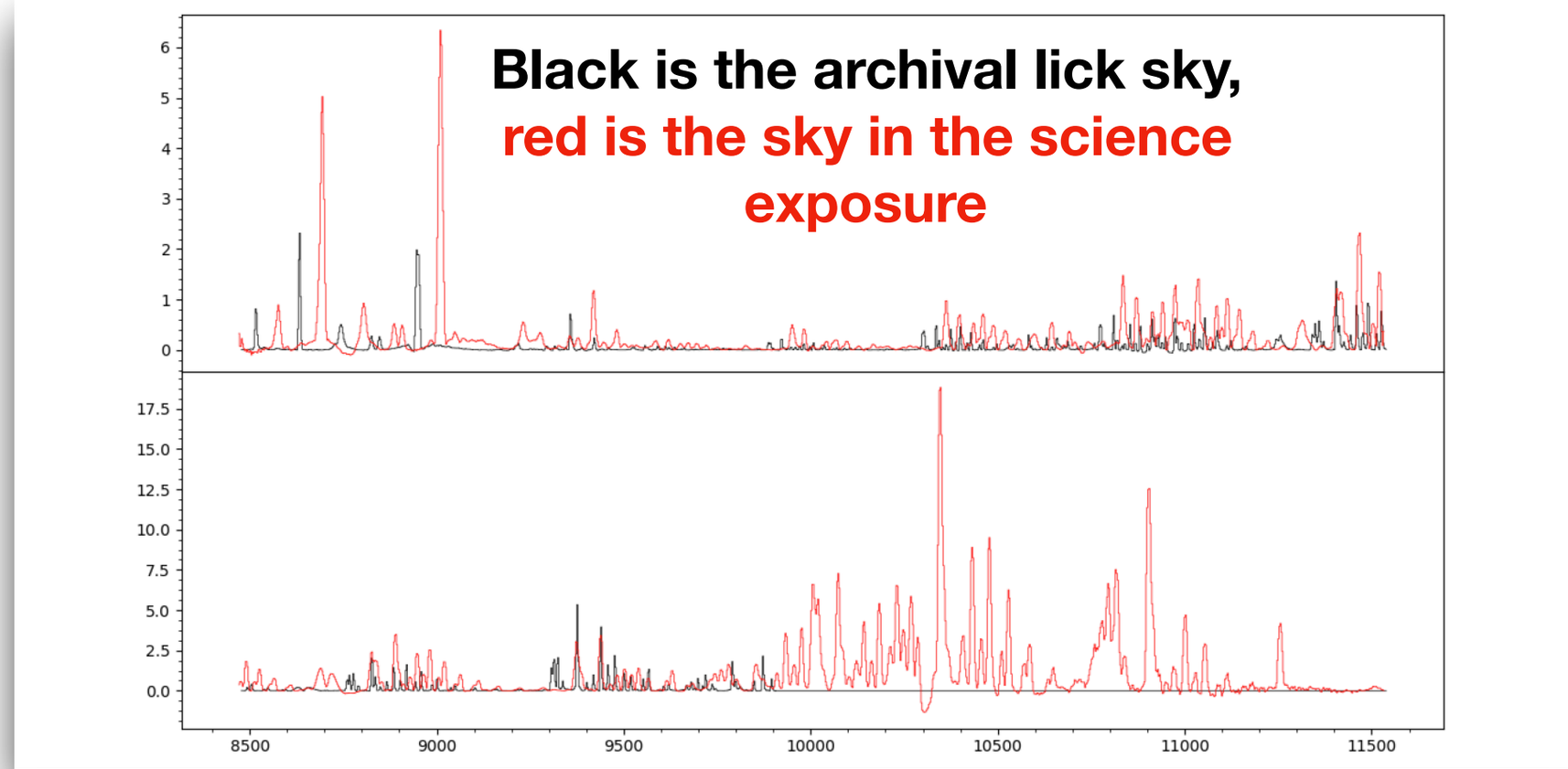
- Repeat the same for the red. The only difference is that when it asks you the grating at the beginning, type “red”





An example with our test data - SN 2018gdg

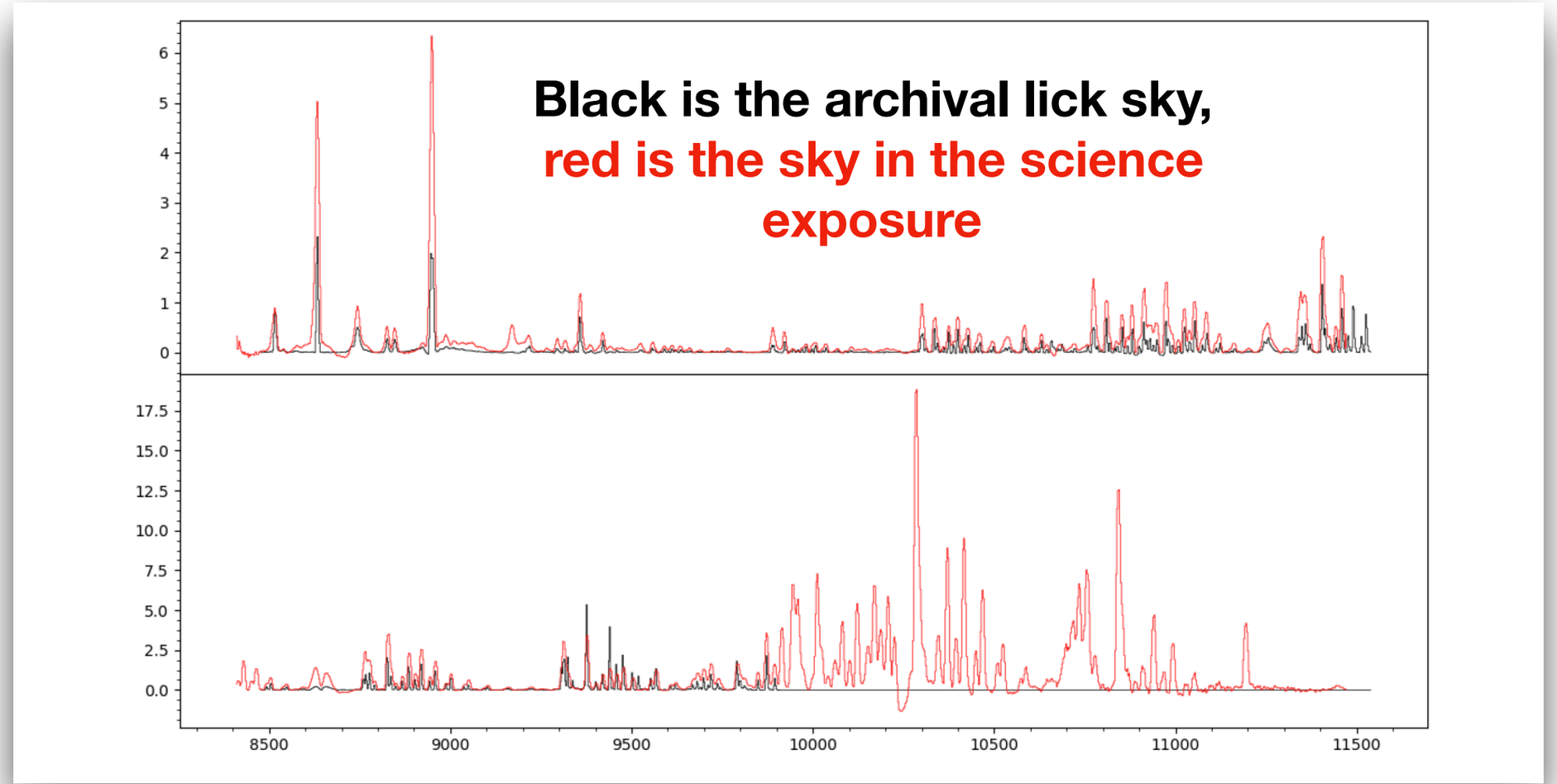
- In this case, we have an issue with the skylines. It finds a shift of -6.14 angstroms, which is wrong



- When it asks us if it's ok, we answer no
- It asks us for a desired shift. Type "-67"
- Normally, this will not happen. We will investigate.

An example with our test data - SN 2018gdg

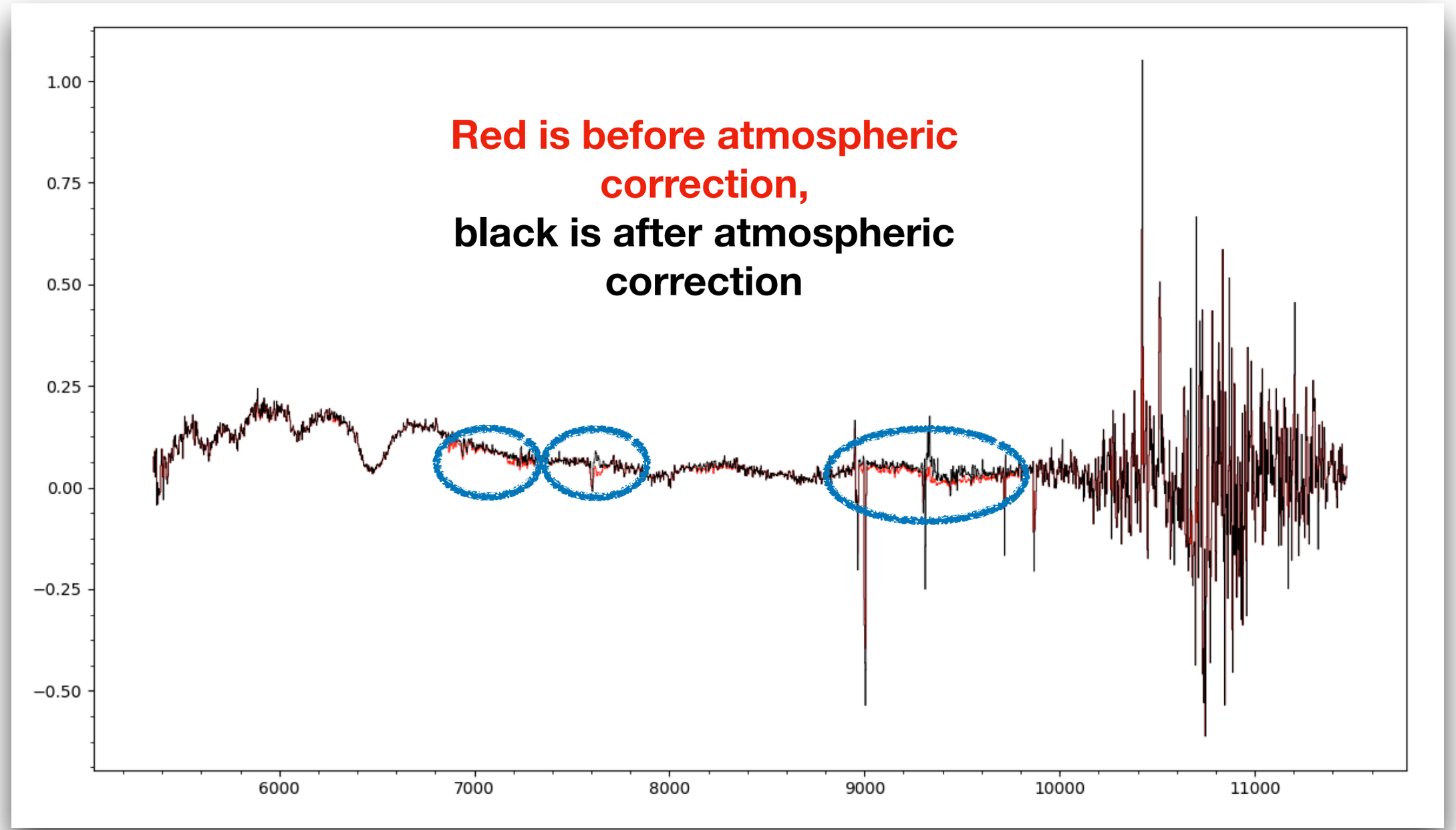
- Now, it looks better



- Again, it asks us if it’s ok, now we answer yes

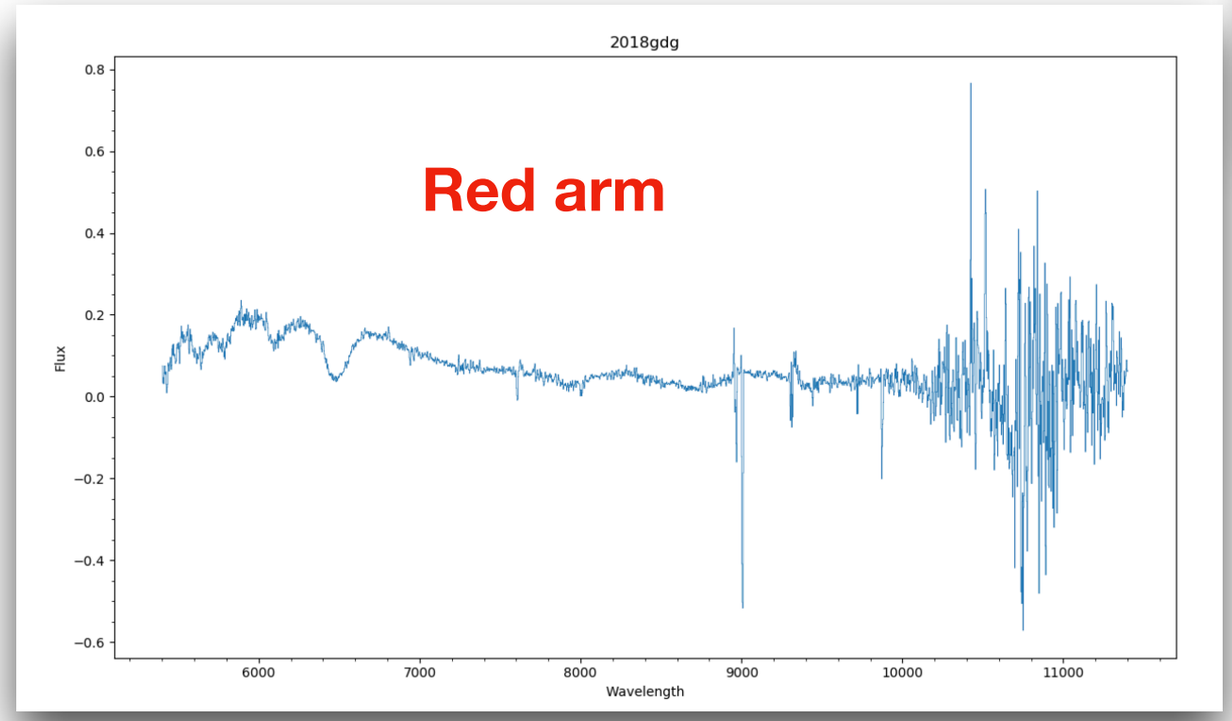
An example with our test data - SN 2018gdg

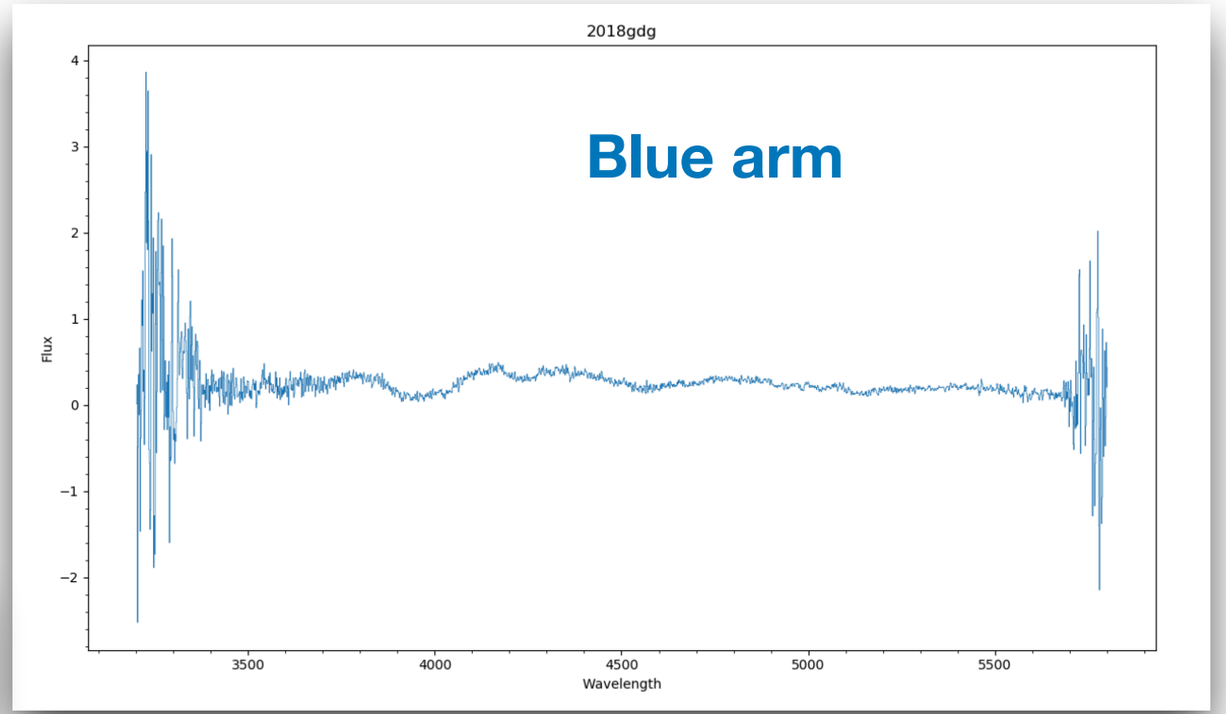
- Then, it removes the tellurics and plots



An example with our test data - SN 2018gdg

- I rebin to 2.6 angstroms and with wavelength range 5400 - 11400
- I name the spectrum SN2018gdg_red
- The spectra look like this





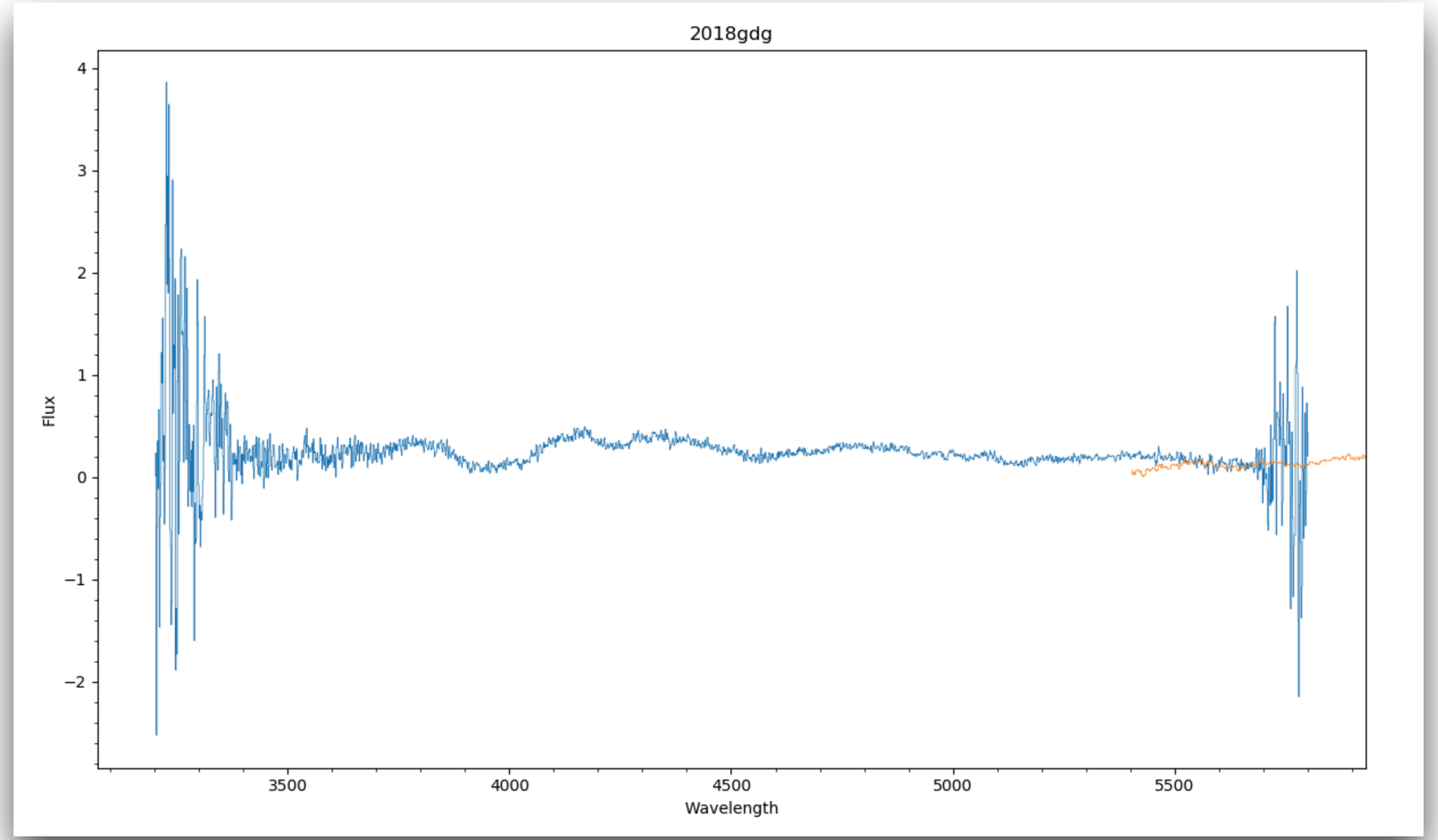
- We need to combine the two arms
- We do this with wombat.py
- Wombat needs some time to familiarize with, but when you learn it, it’s straight forward to rebin, combine or merge spectra

An example with our test data - SN 2018gdg

- We start by typing “wombat.py”
- Wombat uses the concept of “hopper”. What you need to do is read a spectrum and store it in a hopper.
- Then, you work on the hopper (bin a spectrum, combine two hoppers-spectra to a new one etch…)
- The terminal switches to the wombat module. It says: Enter command
- We read a fits file spectrum by typing “rfits”. It asks the name of the fits file, we type “SN2018gdg_blue-20180915”
- Now, we need to save it in a hopper. The command is “hop”
- It asks us where to store it. We type “1”. Now, hopper 1 has the SN2018gdg_blue spectrum
- We do the same for the red. Type “rfits”, then “SN2018gdg_red-20180915”. Then, “hop”, but now store it in hopper 2
- We load a hopper with the command “rh”. W type “rh” and asks us “read from which hopper?”. We type 1.
- We plot a hopper with the command “plot”. Type “plot” and the plot appears. In the terminal, it asks “Change scale?”. If you want, you can answer yes and zoom just like in iraf (clicking on the bottom left and upper right of the zooming window). If you like it, type “yes”

An example with our test data - SN 2018gdg

- Lets try to overplot the red arm to our plot. We type “rh” and then 2. Now, the active hopper is 2. Then, type “oplot”. The red arm appears in orange



An example with our test data - SN 2018gdg

- Now, we will try to merge the 2 arms. We use the command “cat”.
- We type “cat”. It says “This will combine blue and red pieces from two hoppers”, and then asks us the first hopper. We type “1”
- Then, it asks us for the next hopper (there is a typo, and the new questions is the same as before). We type 2.
  - Hmmm, we have this:

```
Enter command: cat

This will combine blue and red pieces from two hoppers

Enter first hopper: 1
Enter first hopper: 2
Spectra do not have same Angstrom/pixel
Blue side: 1.400000000000091
Red side: 2.600000000000364

Enter command:
```

- The spectra have different resolutions. We need to rebin to a common one. We pick the largest one, so we should rebin the blue to the red.
- We type “rh” and then 1 (we load in the active hopper the blue spectrum)
- We rebin the spectrum with the command “bin”. Type “bin” and then it asks “Rebin to how many Angstroms per pixel?”. Type 2.6 (the red resolution). Then it asks the wavelength range. Type the full range (3200 5800) and type “yes”

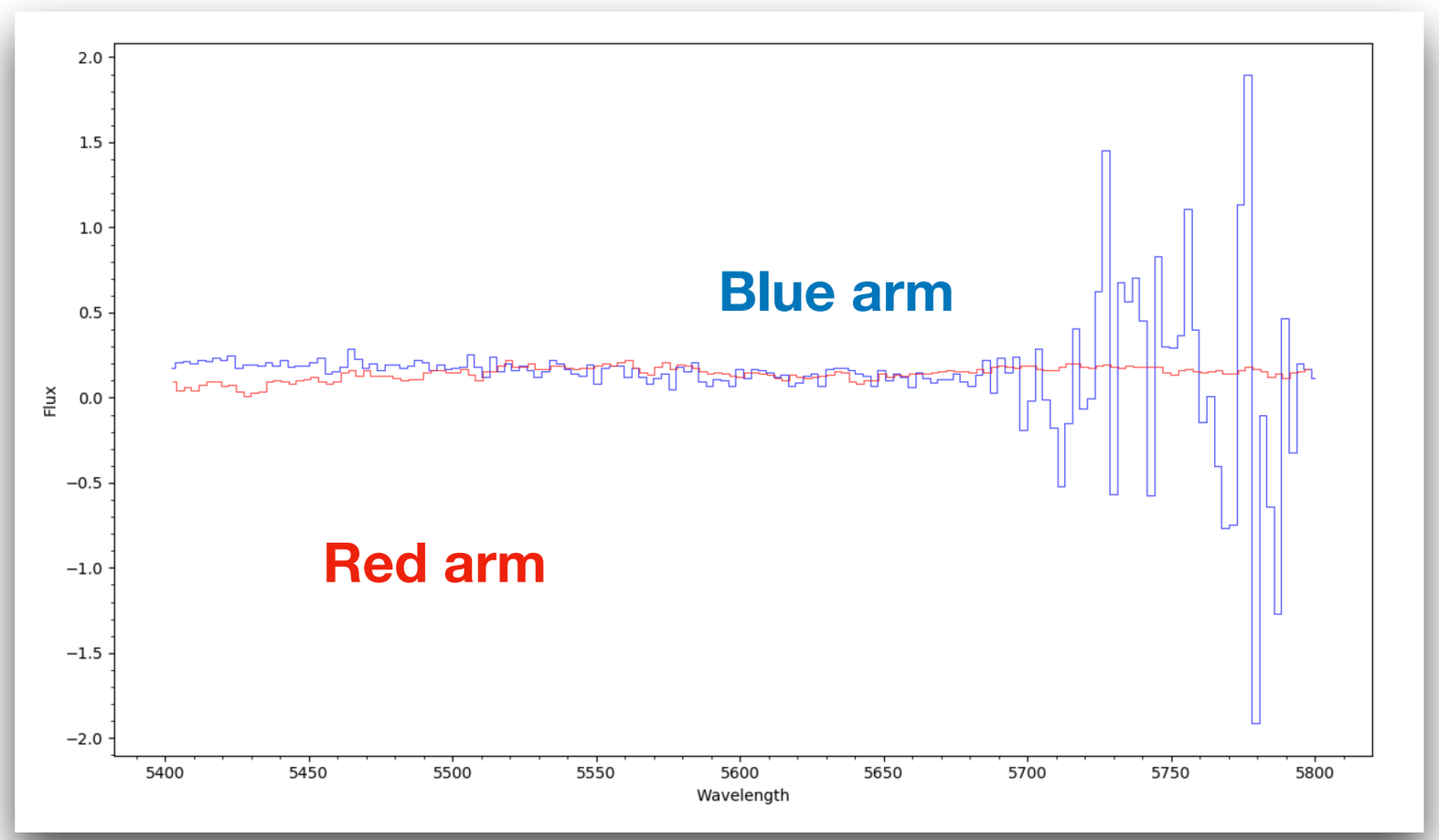
An example with our test data - SN 2018gdg

- Now, in the active hopper we have the rebinned blue spectrum. We need to save it to another hopper.

- So, type “hop” and then “3”.

- Now, we have 3 hoppers, hopper 1 is the original blue, hopper 2 is the original red and hopper 3 is the rebinned blue

- We are ready to merge the arms. Type again “cat” but now, for the two hoppers to merge type “3” and then “2”. A plot appears, with the overlapping region. It asks if you want to change scale (zoom in/out). Answer no.

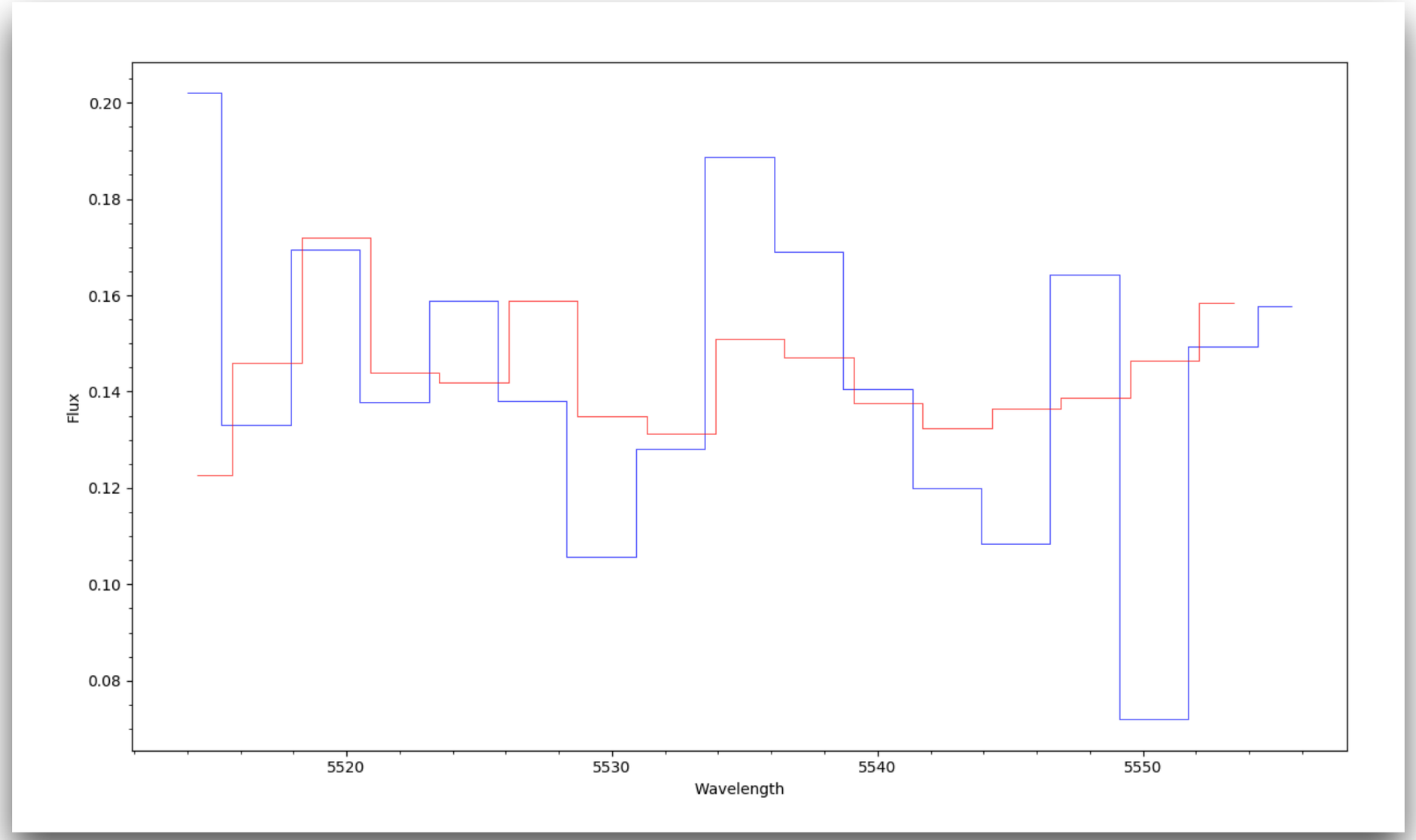


An example with our test data - SN 2018gdg

- The code asks you “Enter (w)avelengths or mark with the (m)ouse? (w/m)”. Type “m”
- The code says “Choose end points of region to compute average”. Use your mouse to mark the region.

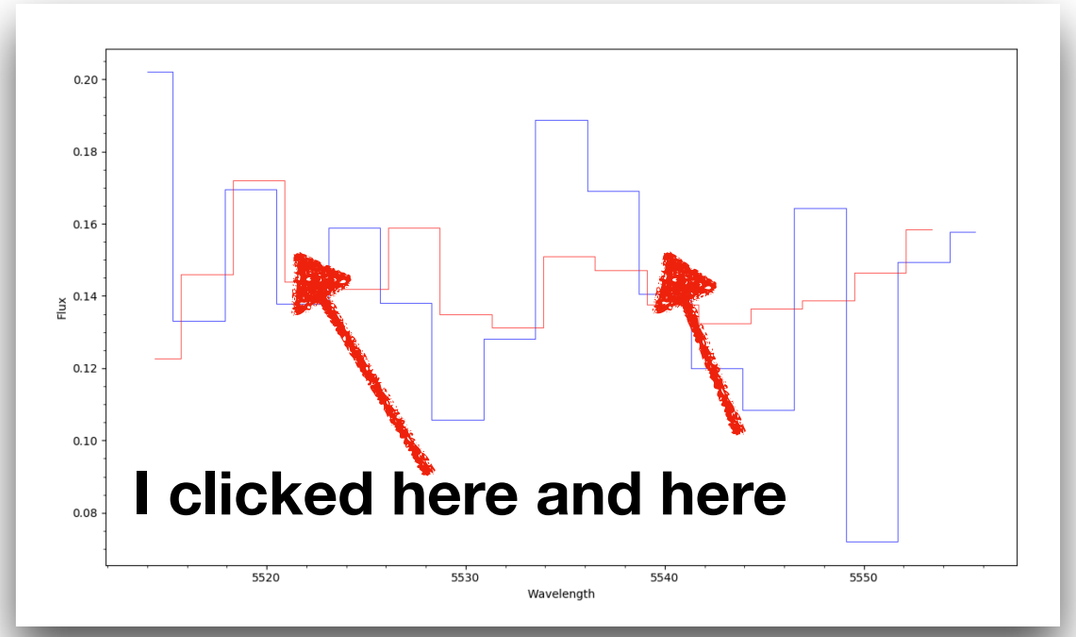
An example with our test data - SN 2018gdg

- The code asks you if the range is correct. Answer yes.
- Then, it asks which arm we want to scale to. Pick one. For example, for scaling the blue arm to the red, we type “r”.
- A plot appears. Answer no to change scale. These is the scaled overlapping region



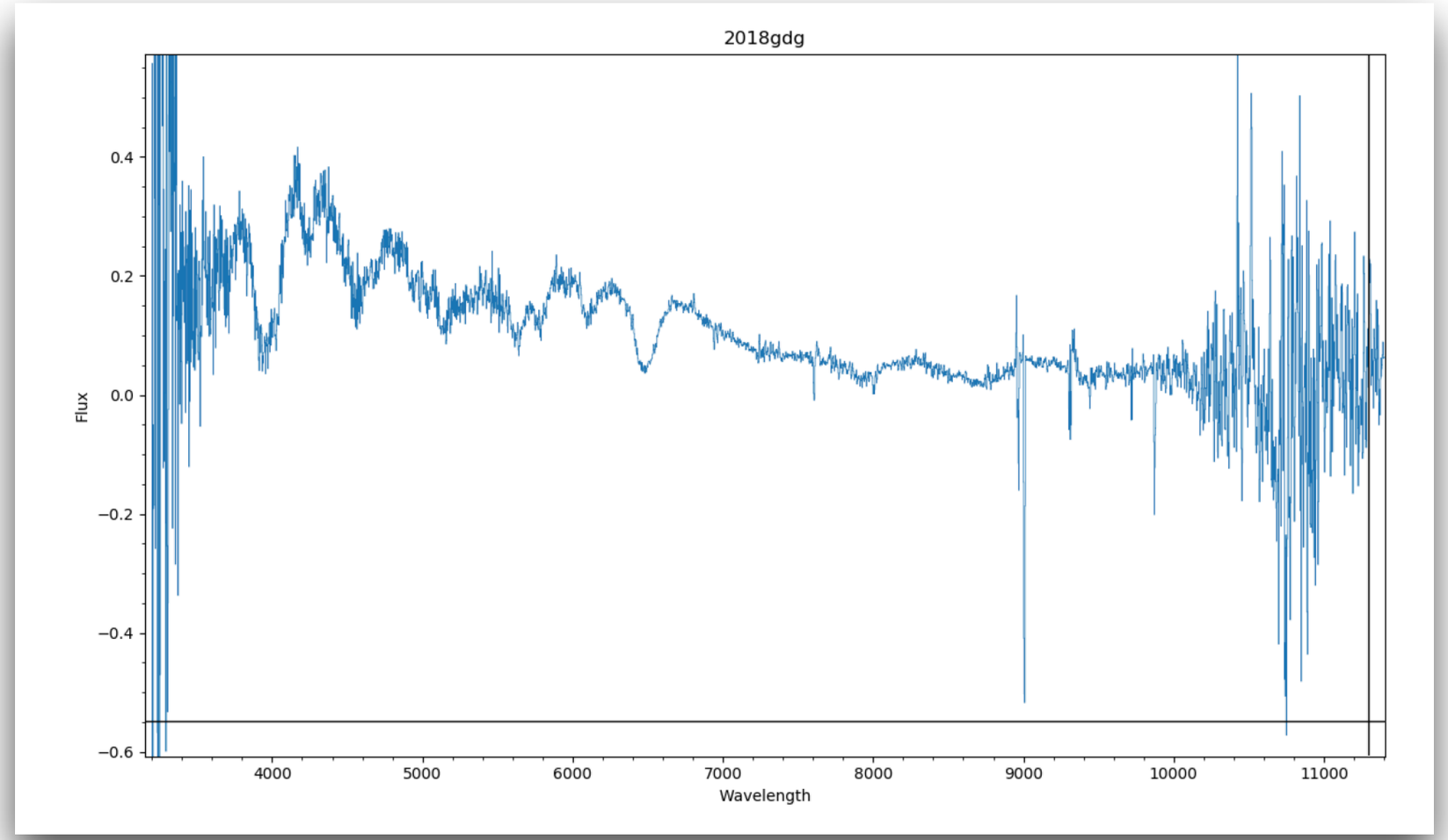
An example with our test data - SN 2018gdg

- The code asks you to choose the endpoints of region to compute the average. Click the two endpoints with the mouse



- The code asks you “Add overlap region (e)qually or with (w)eights (e/w)”. We type “e”
- A plot with the combined spectrum appears.
- The plot asks us to store the new spectrum to a hopper. Type “4”.
- Now, in hopper 4, we have the final merged spectrum. Lets plot it. Type “rh”, then “4”. Type “plot”. Zoom in just like before.

An example with our test data - SN 2018gdg



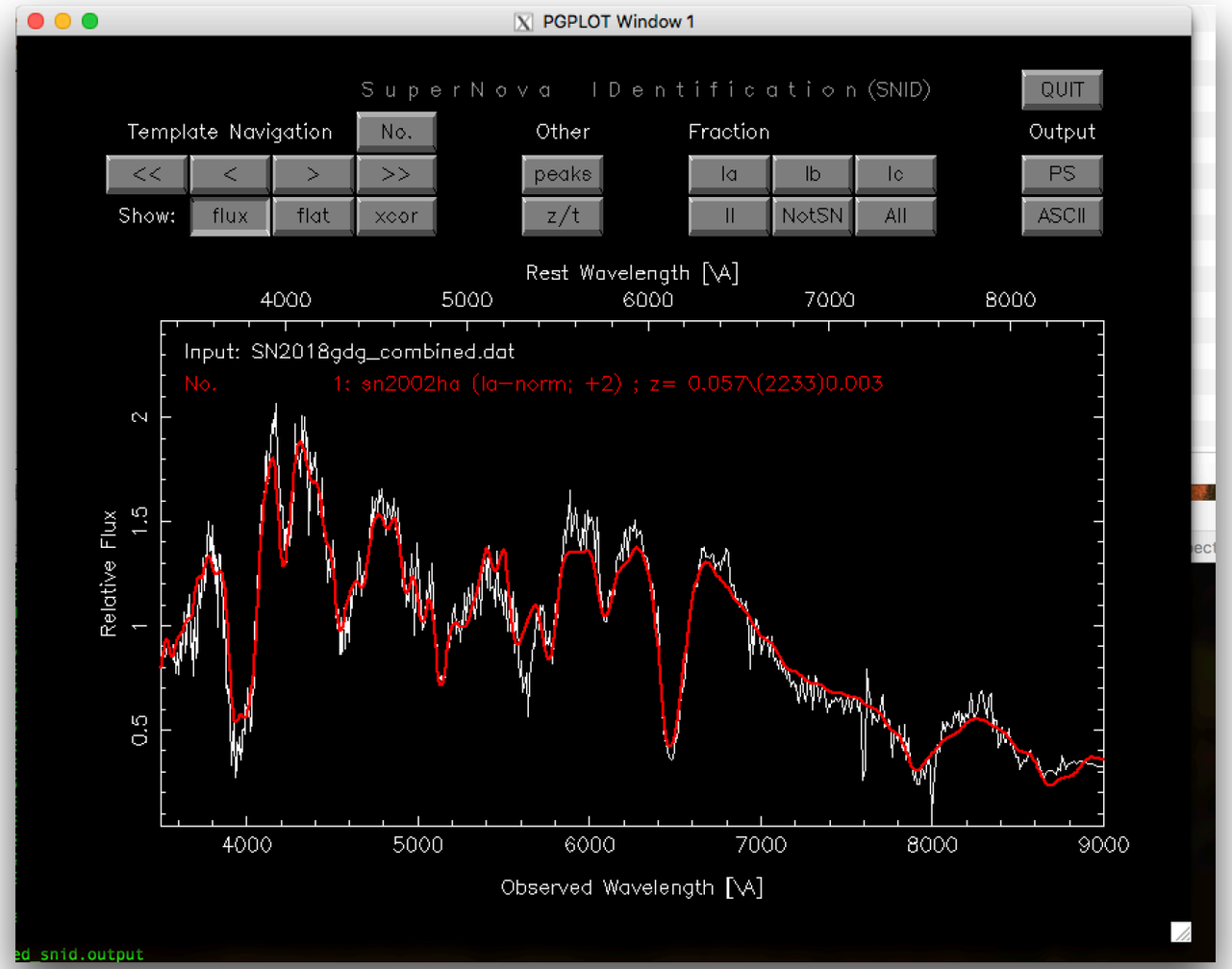
- Looks good, right? Lets save the spectrum in an ascii file. We already have hopper 4 as active, so anything we do happens to that. So, just type “wpl” and for the name, type “SN2018gdg_combined.dat”

- Type “help” to check other wombat commands. It is very useful for quick calculations etch.

- We quit wombat.py by typing “quit”

An example with our test data - SN 2018gdg

- Run your favorite classification code to the .dat file to classify the SN. I use SNID, but there are others too



- Looks like an normal Ia at maximum. You succeeded in identifying the type of stellar explosion million of light years away. You are awesome.

Final remarks

- This pipeline is a work in progress

- We will add better/more archival data (for example standard starts in different airmasses)

- We will soon add the option for quicklook Keck/ LRIS spectra

- Regarding non-quicklook reductions (science quality), the code can do it, but it needs some experience

- Feel free to ask Matt, Georgios or Jon for any help on the pipeline