Setting Up the Pipeline

- Untar nifs_pipeline_quick.tgz. There will be 4 directories: py_scripts, idl_scripts, iraf_scripts, ref_files.

- The contents of py_scripts/ can be moved wherever you want. There is a variable at the top of nifs_main_LP_quick.py for you to enter the location of these files. For example, I put the contents of py_scripts/ in /Users/jlwalsh/Data/py_scripts/LP_2016/quick_reduce/.

- The contents of idl_scripts/ can be moved wherever you want, just be sure they are in your IDL path.

- The contents of iraf_scripts/ can be moved anywhere, but update your iraf login.cl file to include the path to the scripts. For example, to my login.cl file I added:
    - task   nffixbad_anil = home$scripts/nffixbad_anil.cl
    - task   nftelluric_anil = home$scripts/nftelluric_anil.cl

  In this case, the two iraf scripts are located in /Users/jlwalsh/iraf/scripts/, and at the top of my login.cl file, I specify that home is /Users/jlwalsh/iraf/.

- The contents of ref_files/ can be moved wherever you want. There is a variable at the top of nifs_main_LP_quick.py for you to enter to the location of these files.

Setting Up the Pipeline

- You'll need to have the IDL astro library in your IDL path, found here: http://idlastro.gsfc.nasa.gov/
- You'll need to have Craig Markwardt's mpfit.pro, mpfitfun.pro, mpfit2dfun.pro, and mpfit2dpeak.pro in your IDL path. These can be found here: http://cow.physics.wisc.edu/~craigm/idl/fitting.html
- You'll need to have sigfig.pro in your IDL path, found here: http://w.astro.berkeley.edu/~johnjohn/idlprocs/sigfig.pro
- You'll need to get pIDLy, found here: https://github.com/anthonyjsmith/pIDLy, and install by typing 'pip install pidly'.

Running the Pipeline

- Make a directory to hold all the raw data. In that directory, make subdirectories for each date following a yyyymmdd format. Put the NIFS raw data into the appropriate subdirectories.

- Edit the top of nifs_main_LP_quick.py (originally located in py_scripts/). This part is well documented. Note, to determine the names of the galaxies and telluric stars that were observed, see the observing logs for the night(s) you are working on.

- If you have set reduce_tellurics or reduce_galaxies to ‘yes’ at the top of nifs_main_LP_quick.py, then you need to have an opened ds9 window. Before the pipeline starts the part that requires ds9, it will remind you to open ds9 if you haven’t already.

- Go to your rootdir (variable set at the top of nifs_main_LP_quick.py). Open pyraf, then type:
    - pyexecute(“[directory path]/nifs_main_LP_quick.py”)

  For example, I would open pyraf in a terminal, then type pyexecute(“/Users/jlwalsh/Data/py_scripts/LP_2016/quick_reduce/nifs_main_LP_quick.py”).

Data Directory Structure

- rootdir [e.g., /Users/jlwalsh/Data/LP_2016/nifs_data/2016b]
  - datadir [directory holding the raw data, e.g., raw_data]
    - yyyymmdd [raw data from different dates, e.g., 20121227]
  - reducedir [directory holding all the data reductions, e.g., reduced_data_quick]
    - daycals [all baseline daytime calibrations]
      - yyyymmdd
        - observational setup [filter and central wavelength, e.g., hk_2.20]
    - tellurics [telluric star observations]
      - yyyymmdd
        - observational setup
          - star name [e.g., hip10848]
    - galaxy [science object name, e.g., ngc1277]
      - yyyymmdd
        - observational setup
      - merged [merged science cubes]
        - observational setup

- rootdir and datadir must be made by the user (see step #1 of "Running the Pipeline"). reducedir and all it's contents will be made by the pipeline, if the directories don't yet exist. The user has control over naming rootdir, datadir, reducedir - see the top of nifs_main_LP_quick.py. Also, the dates in yyyymmdd format, the telluric stars, and galaxies entered by the user at the top of nifs_main_LP_quick.py will be used to name the appropriate directories.

Summary of the Procedure

- Using header information, sort raw files into the appropriate subdirectories (e.g., daycal, telluric, and galaxy). Create lists of files that will be fed into pyraf (e.g., flatlist contains the names of all the flats for a given night and observational setup).

- Reduce the calibrations.
    - Using one of the flats, locate the spectra by matching to the mask definition file. Save the offset.
    - Update the flats and the flat darks with the offset. Create variance and data quality extensions.
    - Combine flats using an average, including an avsigclip rejection method. Do the same for the flat darks.
    - Normalize the flat and create a bad pixel mask from the dark.
    - Update the arcs and the arc darks with the offset. Create variance and data quality extensions.
    - Combine arcs (if more than 1) using an average, including an avsigclip rejection method. Do the same for the arc darks.
    - Extract slices from the combined arc, apply approximate wavelength calibration to headers, apply combined arc dark, apply combined flat.
    - Identify arc lines and determine the dispersion function for each slice. Creates a database/ subdirectory with the wavelength solution for each slice.
    - Update the ronchi masks with the offset. Create variance and data quality extensions.

Summary of the Procedure

- Reduce the calibrations (continued).
  - Combine ronchi masks (if more than 1) using an average, including an avsigclip rejection method.
  - Extract slices from the ronchi mask, apply approximate wavelength calibration to headers, apply combined flat dark, apply combined flat.
  - Determine the correction for distortions along slices and registration between slices by identifying peaks in the ronchi mask. Writes to the database/ directory.

- Reduce the telluric stars.
  - Update the star exposures and the sky exposures with the offset. Create variance and data quality extensions.
  - Subtract each star exposure by the neighboring sky exposure.
  - Extract slices from each star exposure, and apply flat.
  - Replace bad pixels with values interpolated from fitting neighbors.
  - Derive the 2D spectral and spatial transformation for each slice and then apply the transformation.
  - Generate slices that are sampled in constant steps of wavelength and arcsec. Basically a data cube.
  - Extract a 1D spectrum from the data cube for each of the star exposures.
  - Combine the 1D spectra and save to a final output file.
  - Make a blackbody spectrum for a temperature of 9480 K (for an A0V star) and save to an output file.
  - Fit the Brγ absorption line and remove it from the combined 1D spectrum.
  - Divide by the blackbody spectrum and save to the final output file.

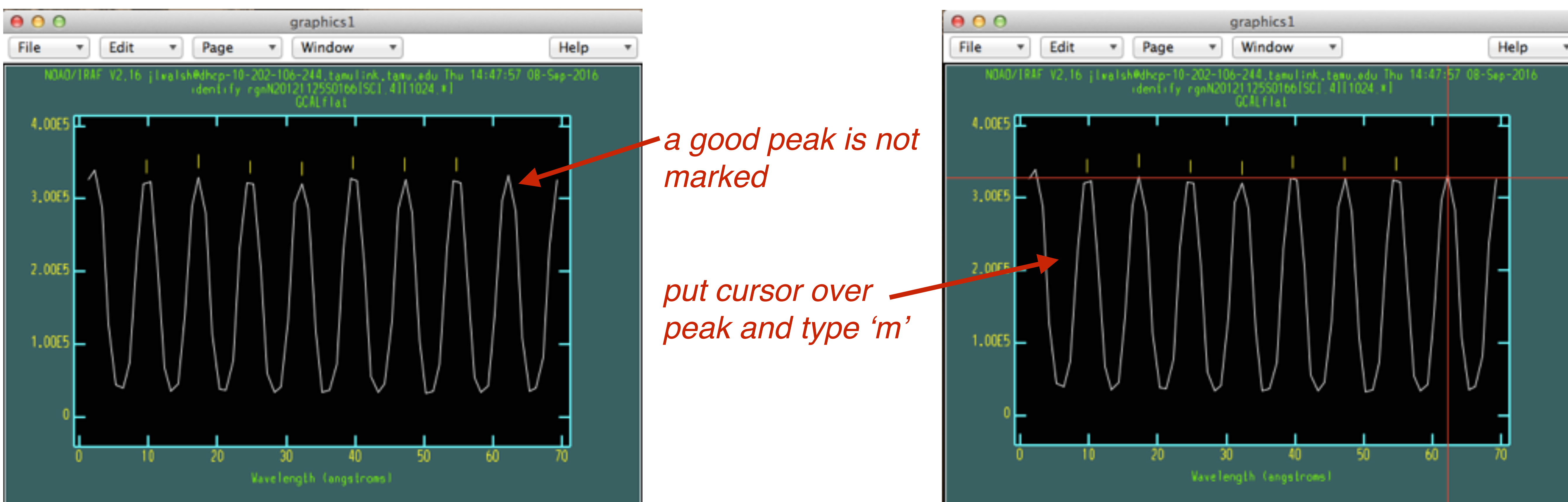
Summary of the Procedure

- Reduce the galaxy.
    - Update the galaxy exposures and the sky exposures with the offset. Create variance and data quality extensions.
    - Subtract each galaxy exposure by the neighboring sky exposure.
    - Extract slices from each galaxy exposure, apply approximate wavelength calibration to headers, and apply flat.
    - Replace bad pixels with values interpolated from fitting neighbors (using a modified version of nffixbad code to work with the variance extension).
    - Derive the 2D spectral and spatial transformation for each slice and then apply the transformation.
    - Generate slices that are sampled in constant steps of wavelength and arcsec. Basically a data cube.
    - Correct galaxy spectra for telluric features (using a modified version of nftelluric to work with the variance extension).
    - Create the data cube (using a modified version of nifcube to work with the variance extension).
    - Copy the galaxy cubes that need to be combined to the merged/ directory.
    - Determine the barycentric correction for each galaxy cube.
    - Put all the galaxy cubes onto the same wavelength array.
    - Collapse the cubes and fit a 2D Gaussian to determine the center of the object for each cube.
    - Combine the galaxy cubes accounting for the spatial offsets. During the combination, a meanclip is used to further clean cosmic rays. Also, works with the variance extensions.

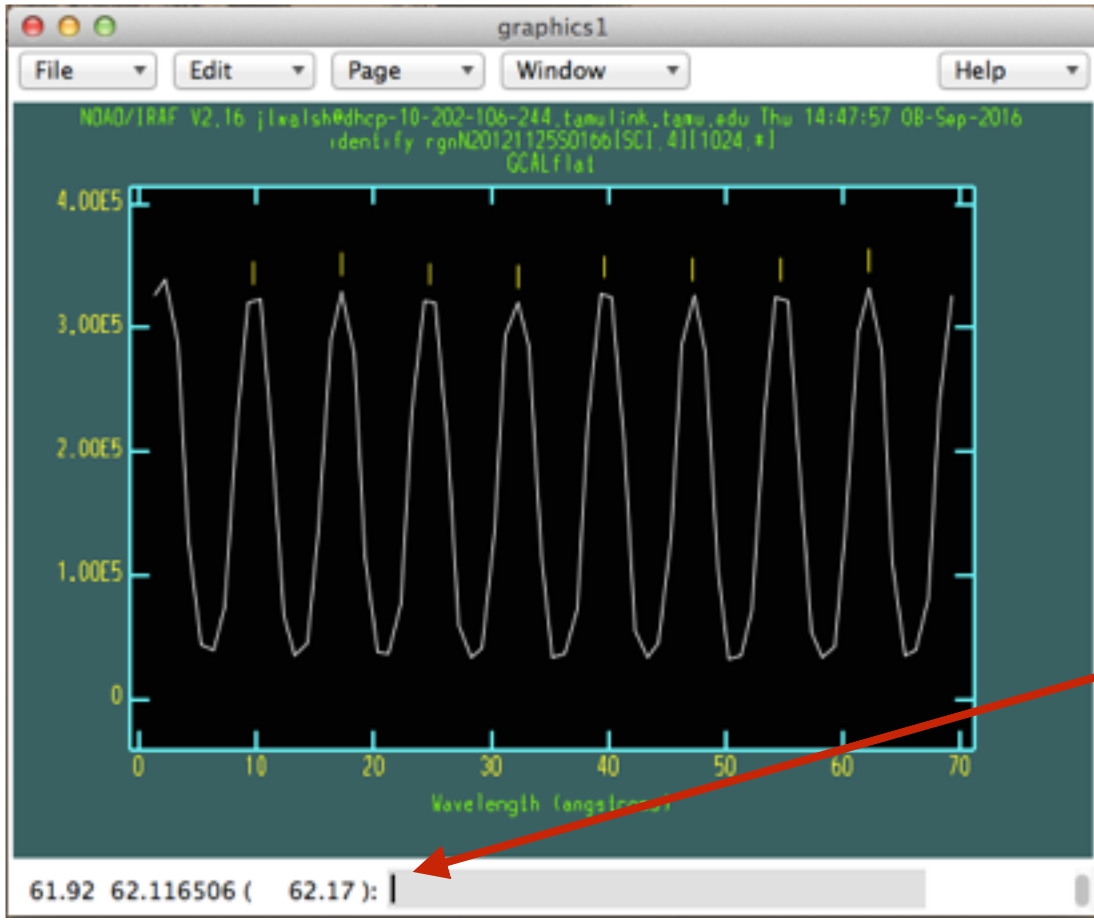
Instructions While the Pipeline is Running

- The pipeline requires a little bit of user interaction, and the steps are summarized below.

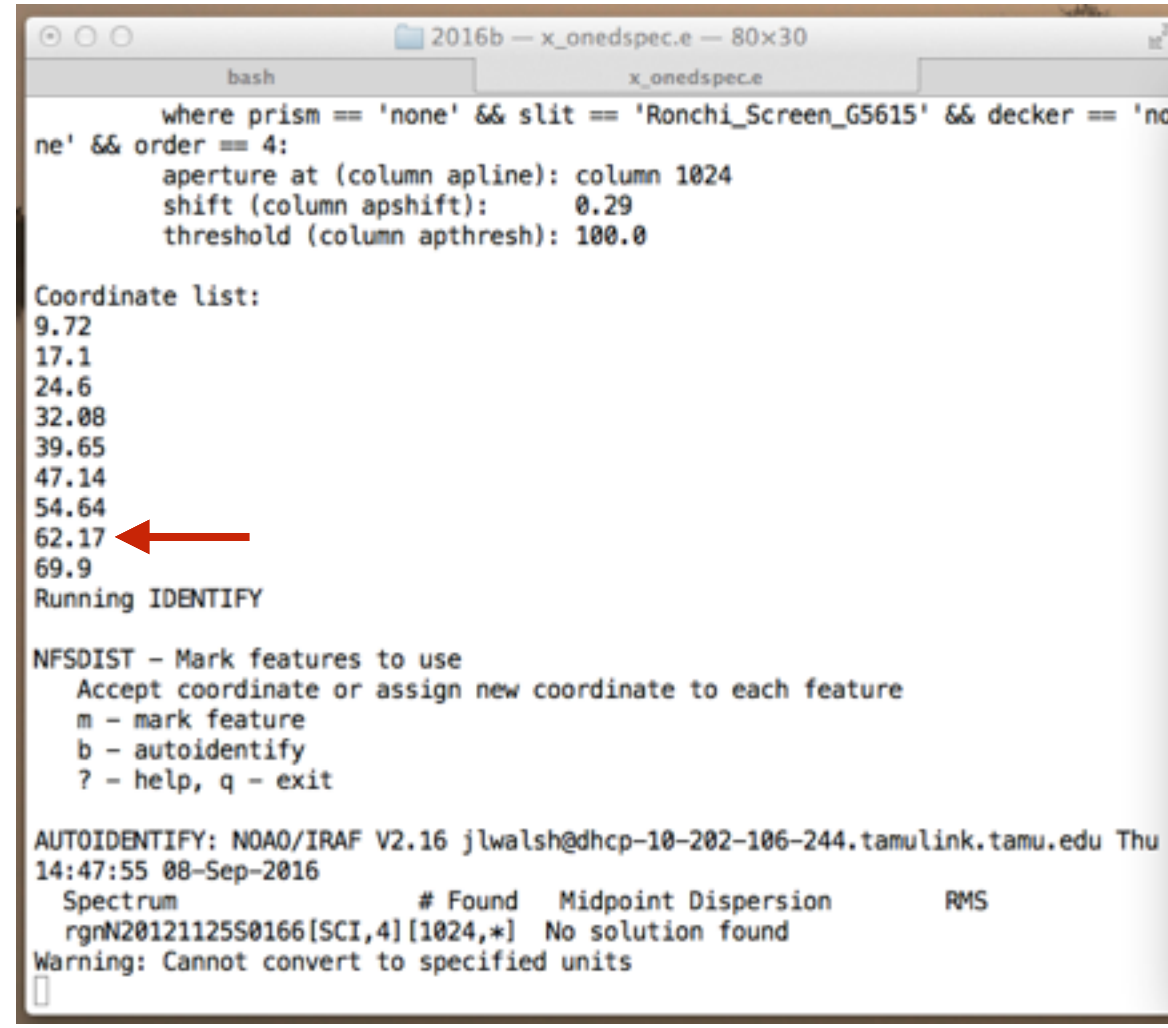
- When reducing the ***daycals***:
  - To correct for distortions along slices/registration between slices, you'll need to identify peaks in the ronchi mask and match to a reference list (like wavelength calibration, but spatially).
  - For each slice, a graphics window will pop up and some of the peaks will be marked. In the graphics window, type 'i' to initialize, then 'x' to fit. Alternatively, you can simply type 'e'.
  - If some good peaks are still not marked, put your cursor over the peak and type 'm'. At the bottom of the graphics window will be the location of the peak selected from the reference list. Hit return to accept (iraf should identify the correct number from the reference list).
  - If there is a bad peak that is marked, put your cursor over the peak and type 'd' to delete. When you are satisfied, type 'q' to move onto the next slice.



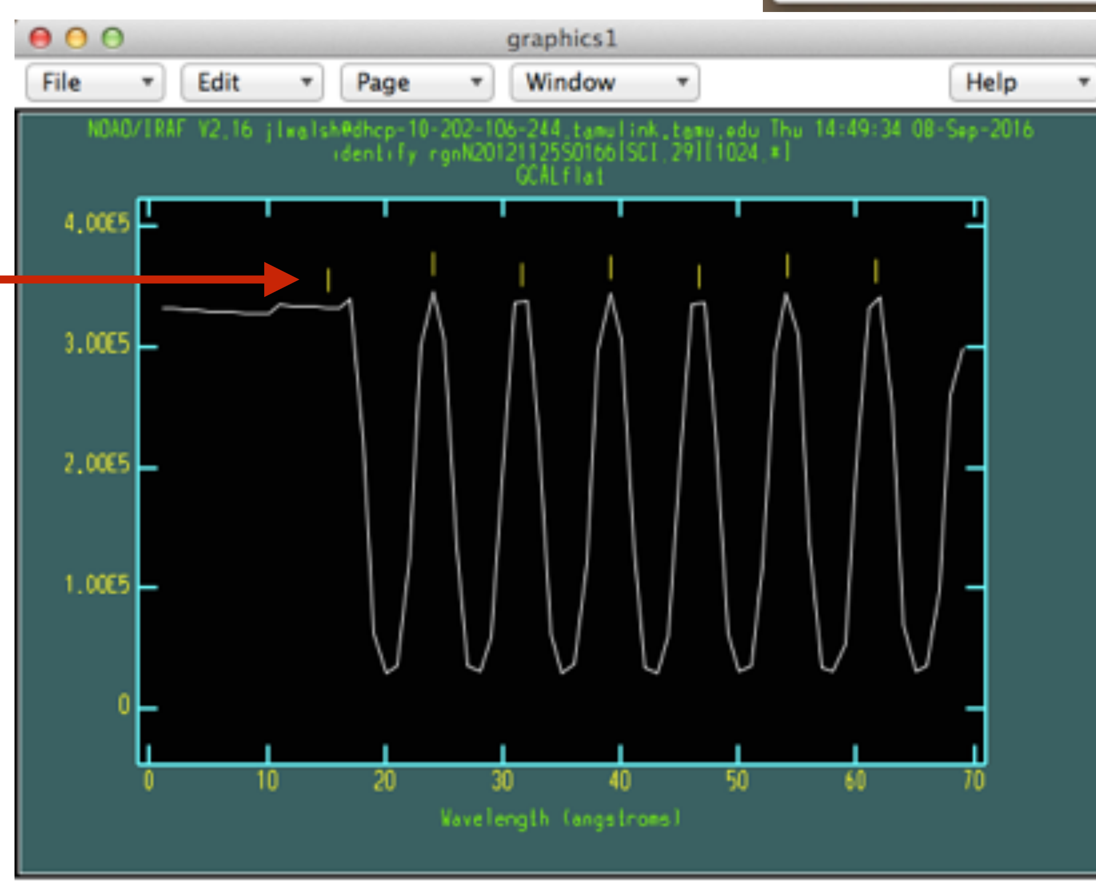
Instructions While the Pipeline is Running



after typing ‘m’, iraf will match the peak with the coordinate list. hit return to accept the value.



example of a bad peak. place cursor over the peak and type ‘d’ to remove.

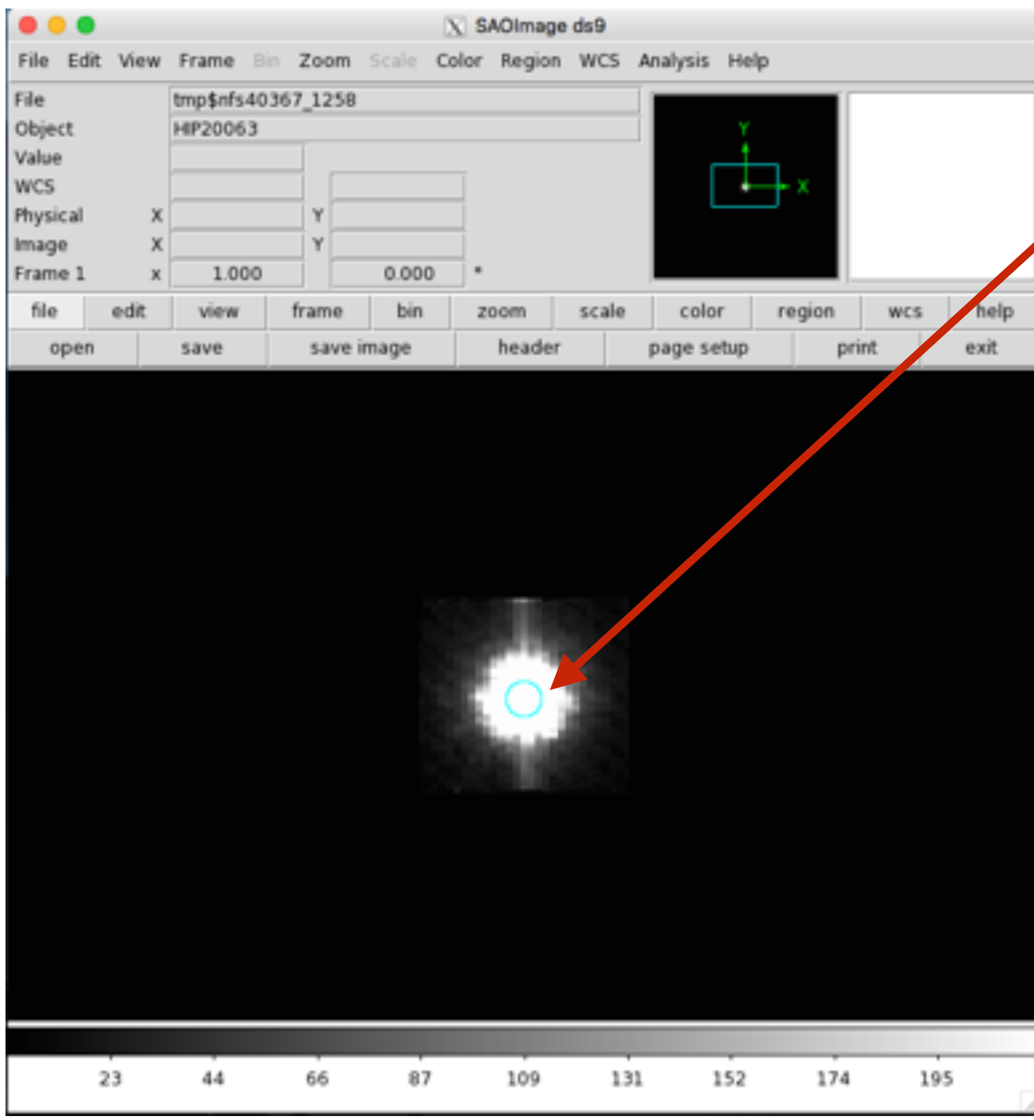


Instructions While the Pipeline is Running

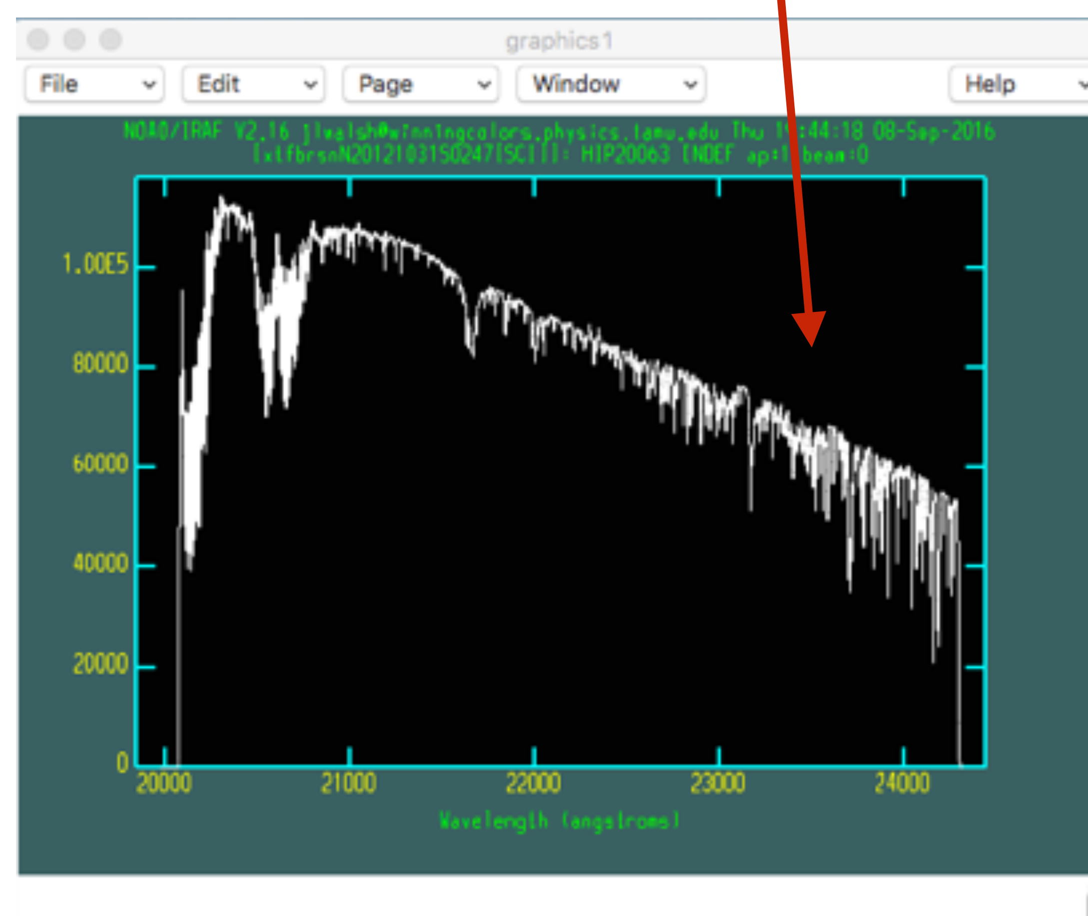
- When reducing the ***tellurics***:
    - The user will be prompted to modify ‘telluriclist’, ’skylist’, and ‘skylist_short’. Initially ‘telluriclist’, ‘skylist’, and ‘skylist_short’ are the same and contain all observations associated with a telluric star sequence (often O-S-O-O-S-O). Modify ‘telluriclist’ so that it contains the telluric star exposures (i.e., the O exposures;1 per line), and ‘skylist_short’ so that it contains the sky exposures (i.e., the S exposures; one per line). Modify ‘skylist’ to have only sky exposures, but the list needs to be the same length as ‘telluriclist’, so repeat sky exposures if necessary. Line #1 from ‘skylist’ will be subtracted from line #1 of ‘telluriclist’. For example, if an O-S-O sequence corresponded to frames 100, 101, 102 then ‘telluriclist’ would list 100 and 102, while ‘skylist’ would list 101 and 101 and ‘skylist_short’ would just list 101.
    - DS9 needs to be opened. When the user is prompted to modify ‘telluriclist’, ‘skylist’, and ‘skylist_short’, a reminder to open DS9 will be given.

Instructions While the Pipeline is Running

- When reducing the ***tellurics*** (continued):
    - The collapsed telluric cube will be shown in a ds9 window. Place cursor over the center of the star and hit any key to extract a 1D spectrum.
    - The spectrum will appear in a graphics window. Place cursor in the graphics display and type ‘q’ to move onto the next telluric exposure.



1D spectrum is extracted from aperture, and displayed in graphics window.

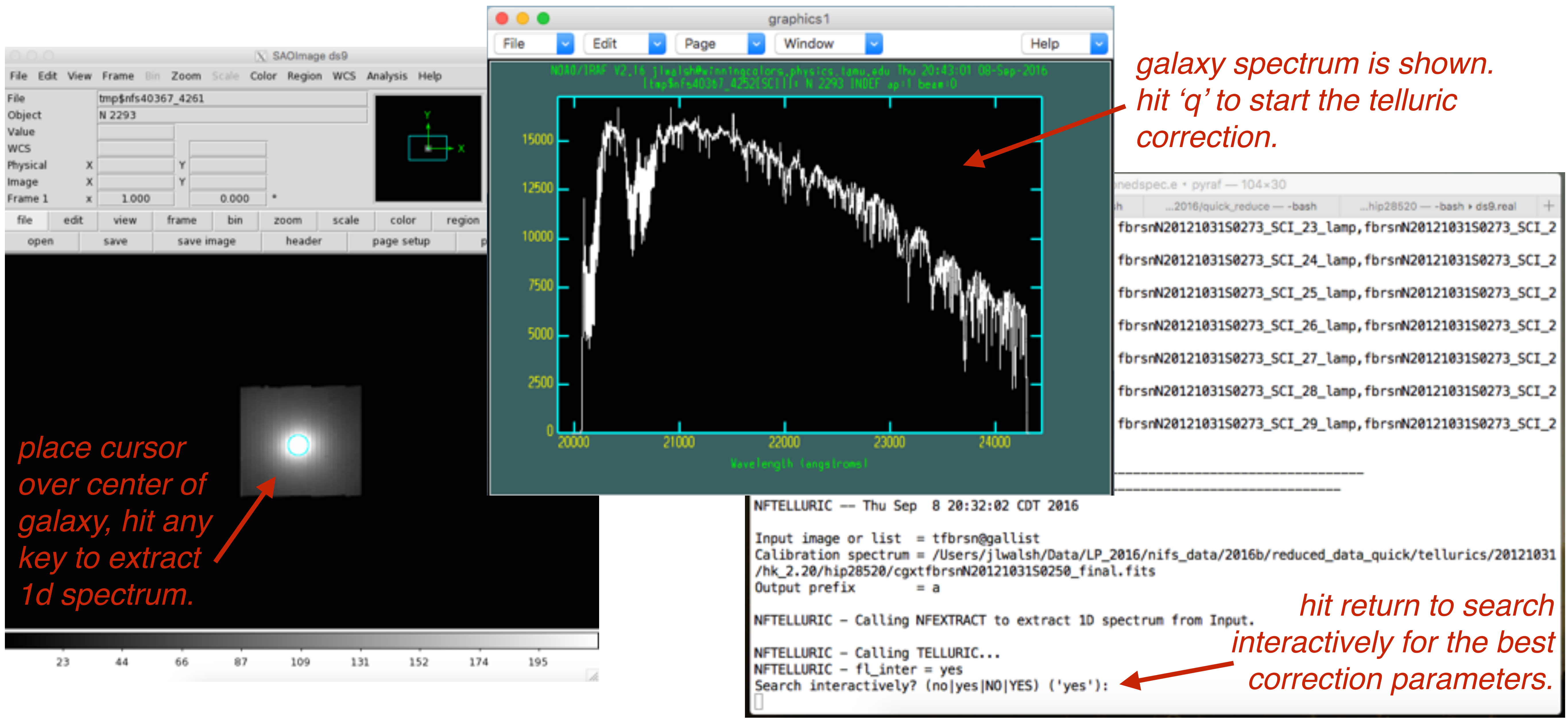


Instructions While the Pipeline is Running

- When reducing the ***galaxy exposures***:
    - The user will be prompted to modify ‘gallist’, ’skylist’, and ‘skylist_short’. Initially ‘gallist’, ‘skylist’, and ‘skylist_short’ are the same and contain all observations associated with a galaxy sequence (e.g., O-S-O-O-S-O). Modify ‘gallist’ so that it contains the galaxy exposures (i.e., the O exposures;1 per line), and ‘skylist_short’ so that it contains the sky exposures (i.e., the S exposures; one per line). Modify ‘skylist’ to have only sky exposures, but the list needs to be the same length as ‘gallist’, so repeat sky exposures if necessary. Line #1 from ‘skylist’ will be subtracted from line #1 of ‘gallist’. For example, if an O-S-O sequence corresponded to frames 100, 101, 102 then ‘gallist’ would list 100 and 102, while ‘skylist’ would list 101 and 101 and ‘skylist_short’ would just list 101.
    - DS9 needs to be opened. When the user is prompted to modify ‘gallist’, ‘skylist’, and ‘skylist_short’, a reminder to open DS9 will be given.

Instructions While the Pipeline is Running

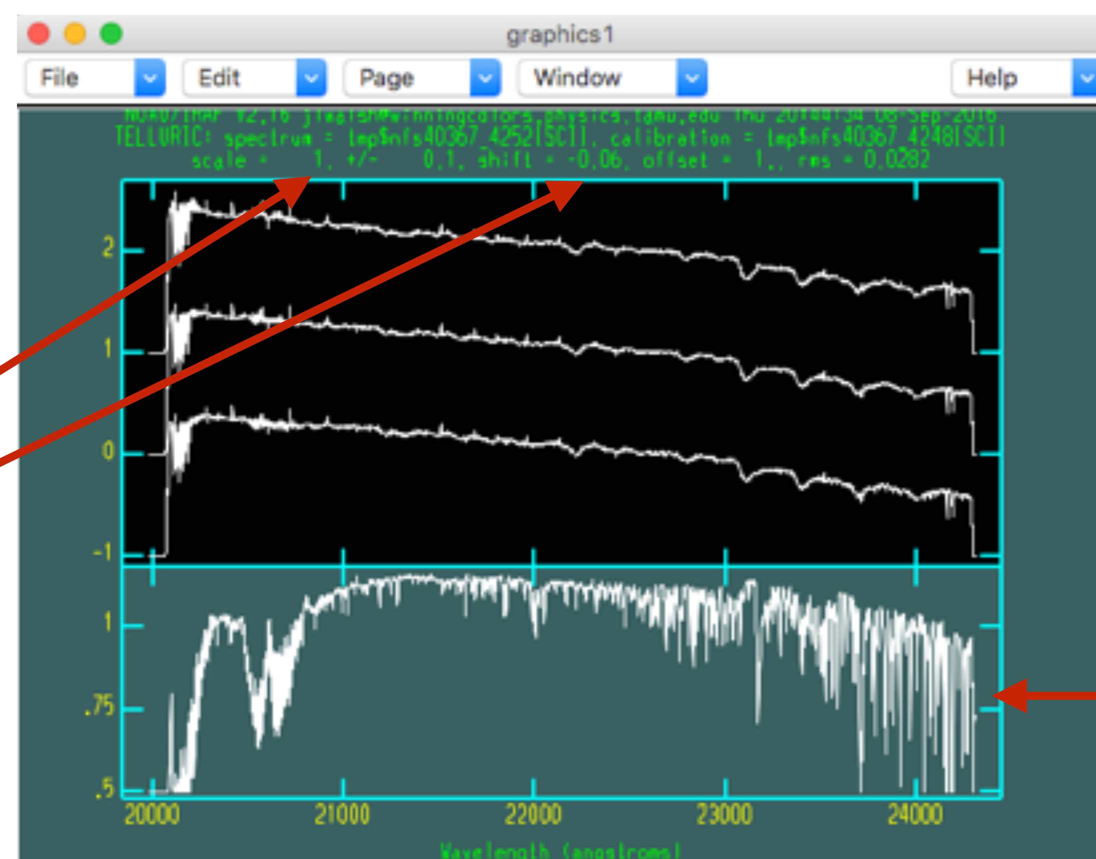
- When reducing the ***galaxy exposures*** (continued):
  - The collapsed galaxy cube will be shown in a ds9 window. Place cursor over the center of the galaxy and hit any key to extract a 1D spectrum.
  - The spectrum will appear in a graphics window. Place cursor in the graphics display and type ‘q’ to start the telluric correction.
  - The terminal window will ask if you want to search interactively for the best telluric correction parameters. Hit return to accept the default answer of ‘yes’.



Instructions While the Pipeline is Running

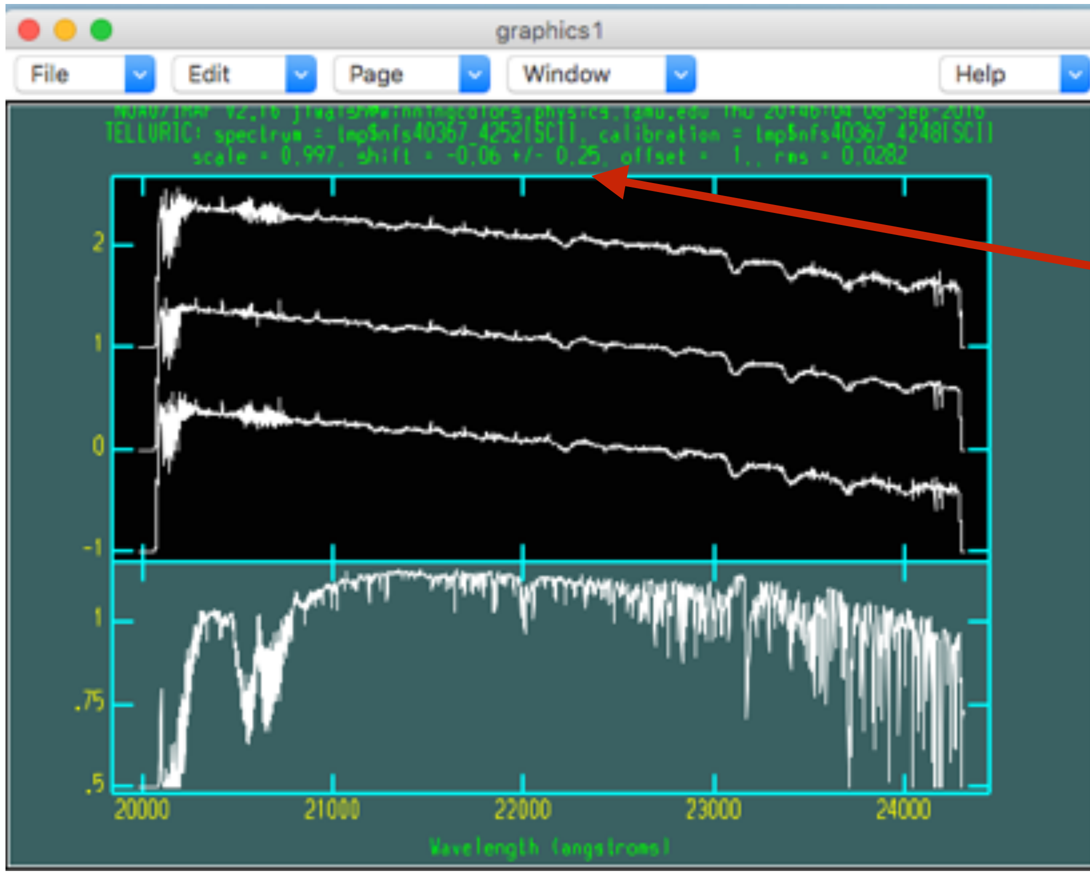
- When reducing the ***galaxy exposures*** (continued):
  - In the graphics window, a corrected 1D galaxy spectrum will be shown (middle plot) for an initial shift and scale values of the telluric. For our quick reduction purposes, just accept this as long as the telluric and galaxy have similar airmass. Place your cursor in the graphics window and type ‘q’ to move onto the next galaxy cube.
  - If you want to adjust the telluric correction parameters:
    - Placing your cursor over the middle spectrum and typing ‘x’ (‘y’) will decrease the step size of the shift (scale) parameter.
    - Applying the adjustment is done by placing your cursor over the top spectrum (for increasing by the step size) or by placing your cursor over the bottom spectrum (for decreasing by the step size), and typing ‘x’ (for the shift) or ‘y’ (for the scale).

telluric-corrected galaxy spectrum for a scale of 1.0 and a shift of -0.06 is shown in the middle. to accept, type ‘q’ in the window.



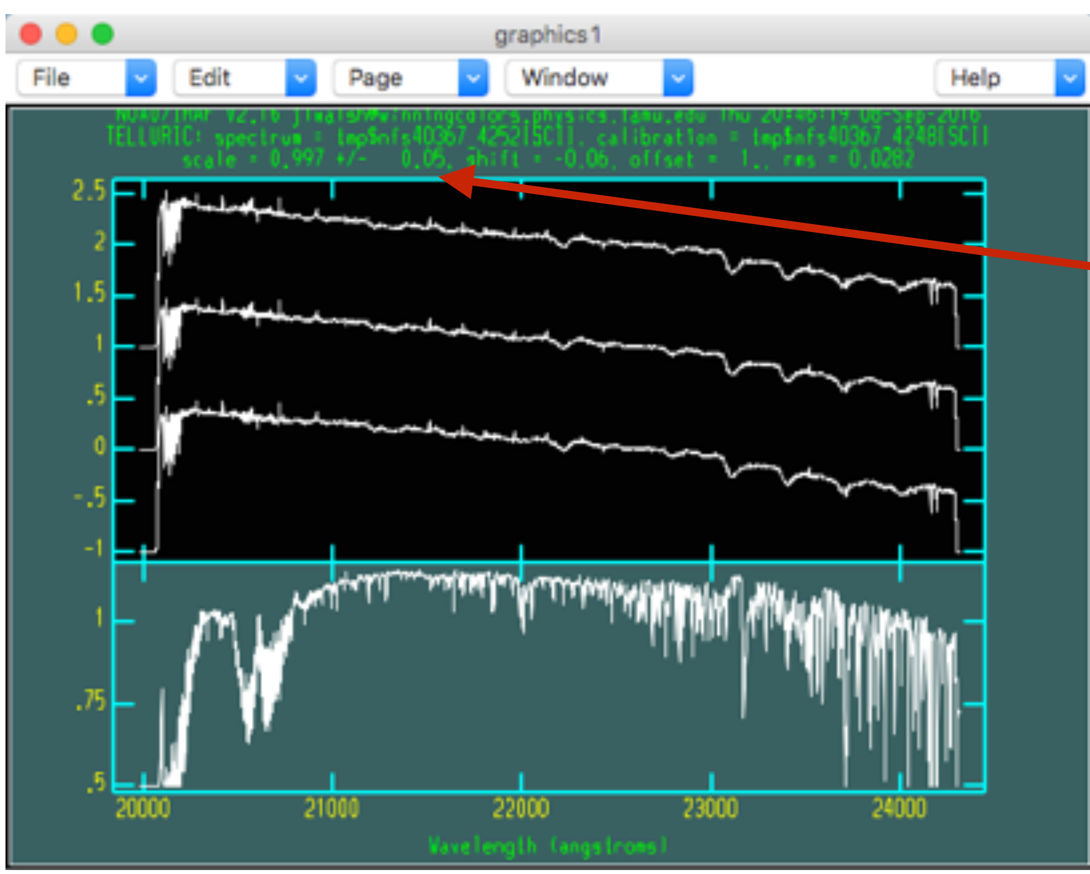
the telluric that is being used is shown here.

Instructions While the Pipeline is Running



decrease the step size in the shift parameter by typing 'x' while your cursor is over the middle spectrum. the step size is shown here.

apply the change by typing 'x' with your cursor over the top spectrum to increase the shift (in this case to -0.06+0.25=0.19) or by typing 'x' with your cursor over the bottom spectrum to decrease the shift (in this case to -0.06-0.25=-0.31)



decrease the step size in the scale parameter by typing 'y' while over the middle spectrum. the step size is shown here.

apply the change by typing 'y' with your cursor over the top spectrum to increase the scale (in this case to 0.997+0.05=1.047) or by typing 'y' with your cursor over the bottom spectrum to decrease the shift (in this case to 0.997-0.05=0.947)

To Do

- The pipeline will produced reduced galaxy cubes good enough for us to determine the S/N and plan for the following night during our priority visiting run. However, the pipeline doesn't yet produce a final sky cube (using the same spatial offsets that were used to make the final galaxy cube) from which we can measure the line spread function. Also, the pipeline doesn't yet reduce any PSF star observations.

- While the pipeline works for K-band NIFS data centered on 2.20 microns, the wavelength calibration fails for data centered on 2.30 microns. There are a couple of objects in the sample where we will use a central wavelength of 2.25 microns. I haven't tested a 2.25 micron NIFS data set, so this part still needs to be figured out.

- We may want to do some parts better, such as the removal of the Brɣ line, the sky subtraction, and treating covariance between spaxels. Also, instead of resampling to make a common wavelength axis/applying the vlsr correction (done in run_shift_anil.pro), perhaps could shift the reference line list during the wavelength calibration, so that only one interpolation is done with nstransform, instead of two interpolations with nstransform and then with run_shift_anil.pro.

- Some other minor stuff to correct: automate the generation of 'telluriclist'/'gallist', 'skylist', and 'skylist_short' and simply have the user check that the files listed are correct; adjust the scale of the displayed data cube during the extraction of 1D spectra; adjust so that a flag will can be turned on to do a "real-time" data reduction at the telescope versus a full data reduction.

- Parts of the pipeline are written in IDL because the codes already existed, while the majority of the pipeline is in python. Perhaps it's better to translate everything into python.

Minor Questions

- Anil has had to manually determine ‘shifty’ (by measuring the gaps in the flats) for old NIFS data, but for recent NIFS data nfprepare appears to work just fine? The current pipeline simply calls nfprepare and does not do any check.

- What should fl_corr be set to (yes/no) when calling nfprepare on the flats, flat darks, arcs, arc darks, and the ronchi mask?

- When running nsfitcoords, is there a preferred lxorder, lyorder, sxorder, and syorder?