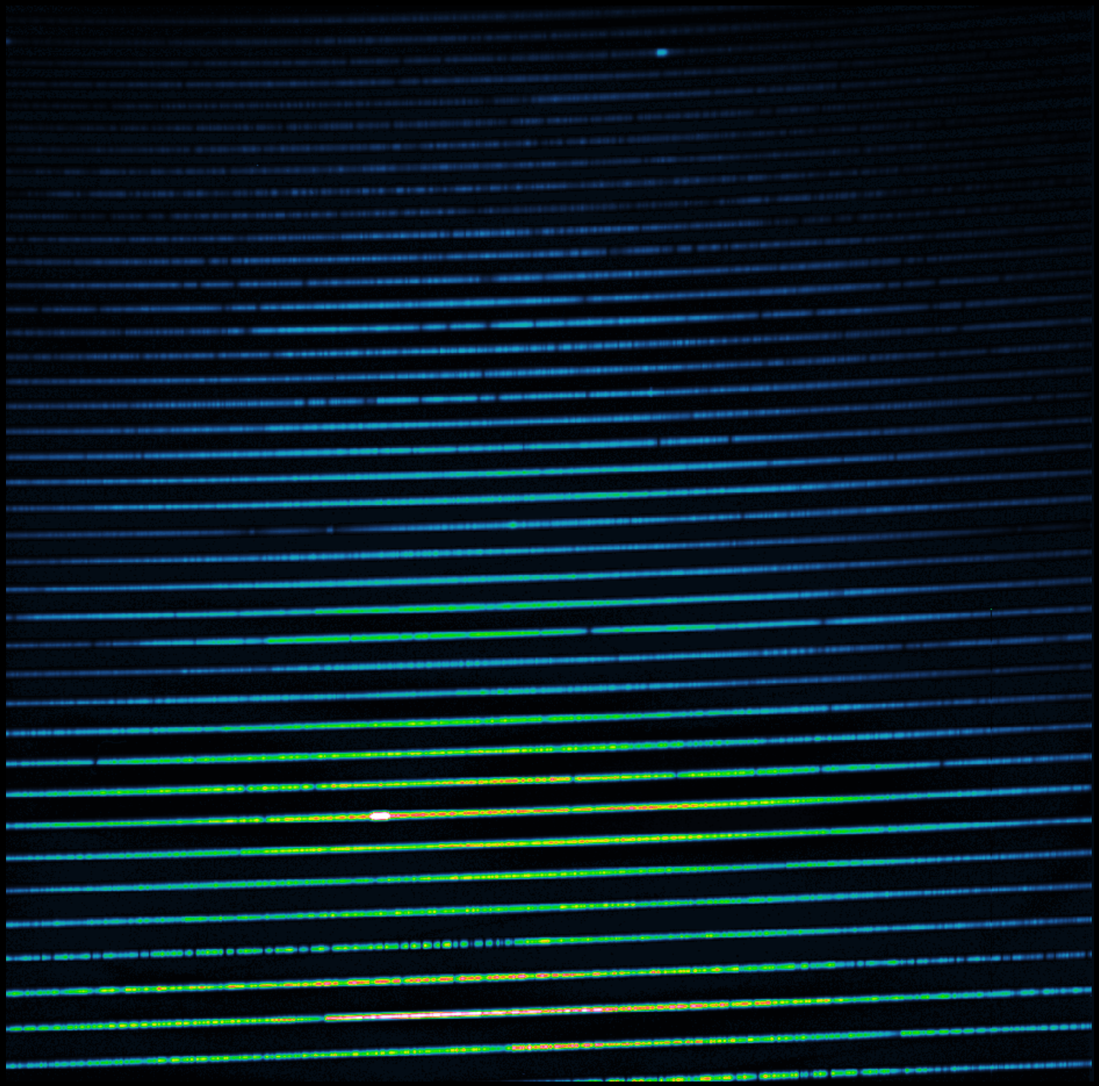
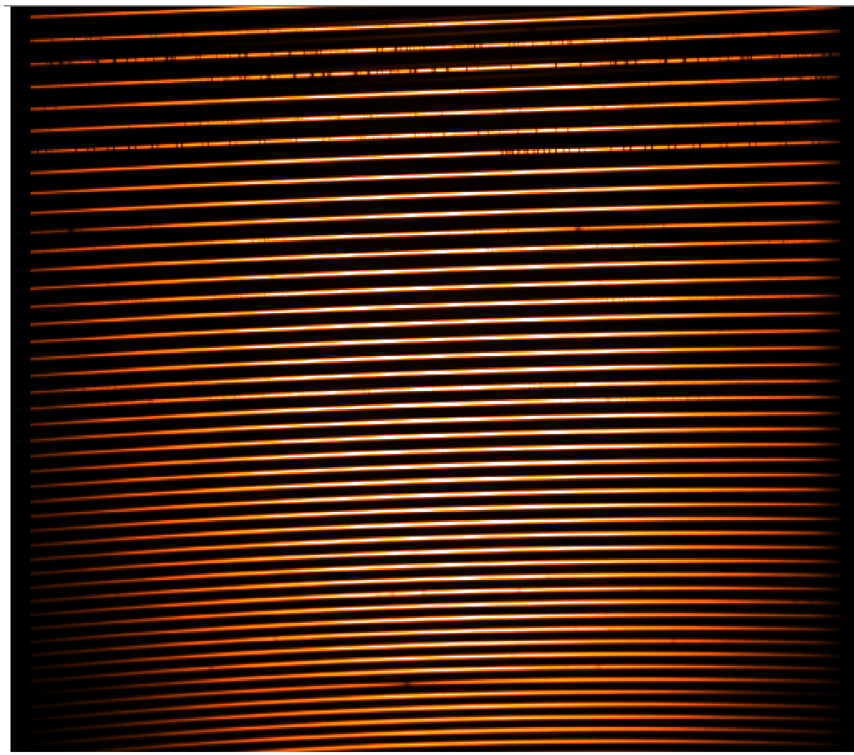
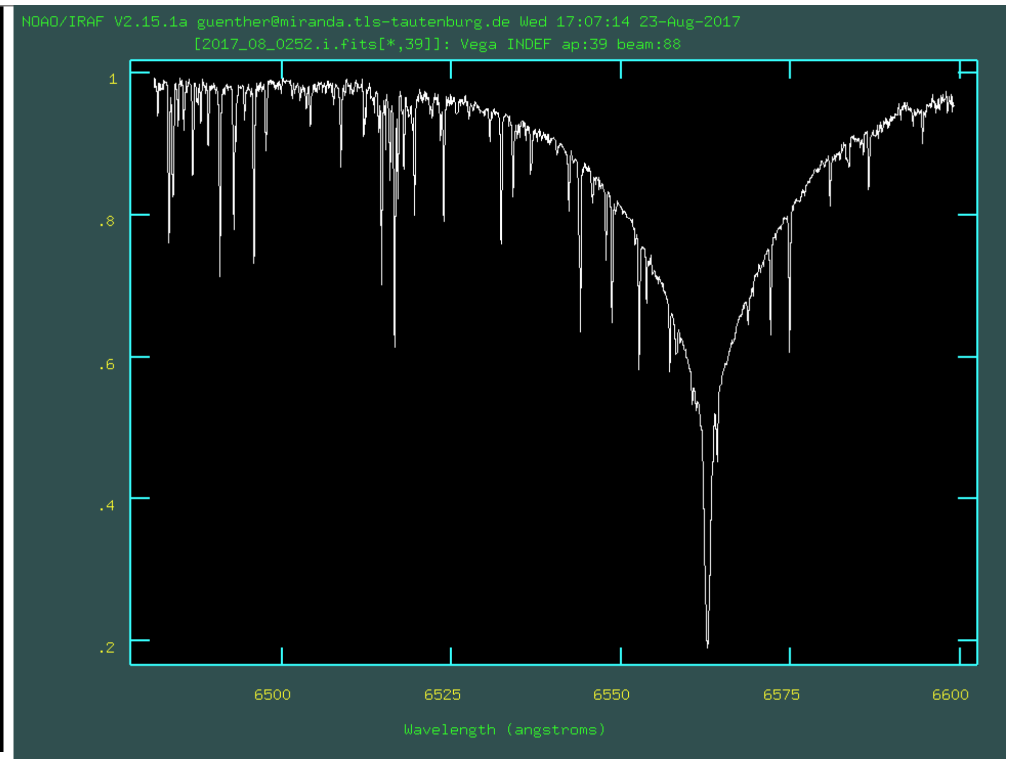
How to reduce Echelle spectra with IRAF to measure the RVs



What we want to do





IRAF package echelle

- The package that contains the necessary commands can be found in **noao.imred.echelle**
- We will now go through all steps necessary to get a 1D-spectrum out of the image that we observed, here are some calibration files for simplification
- To understand what the single steps do, do it by hand the first time!
- You can speed up the process by creating your own script file later containing the commands you use (you can use **mkscript** to do it) and using lists!
- Scripts are loaded using **cl < scriptname.cl &**
- (to go out of your window or to the next step type "q" in the window)
- You are always encouraged to use "**help** task"

Making the average bias frames

The bias frames

- The bias has two parts:
- 1.) a structure that is always the same
- 2.) An offset that changes from frame to frame.
- It is thus not very smart to do it like that: science_frame - bias = science_frame_corrected.
- Better do it like this:
- Science_frame - structure = dummy
- Measure off-set using the overscan and then
- Dummy - value_of_overscan = science_frame_corrected.
- The frame “structure” should be made in such a way that the value

  in the overscan is 0.0

First step: average all bias frames using a list:

```
echelle> ls *Bias.fits > bias.lis
echelle>
```

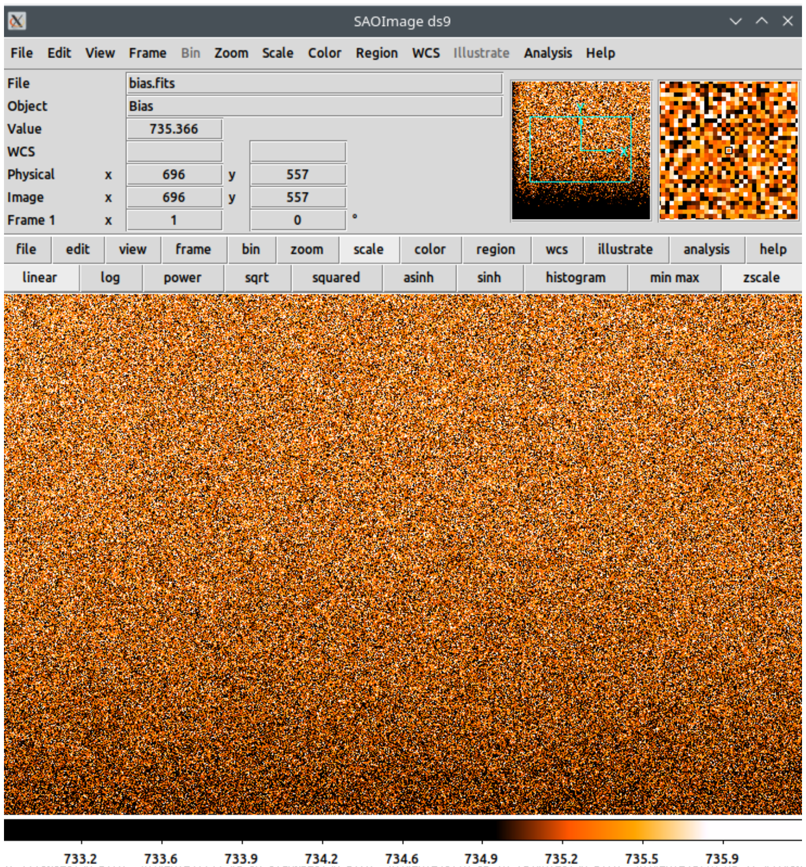
And then combine it using incombine

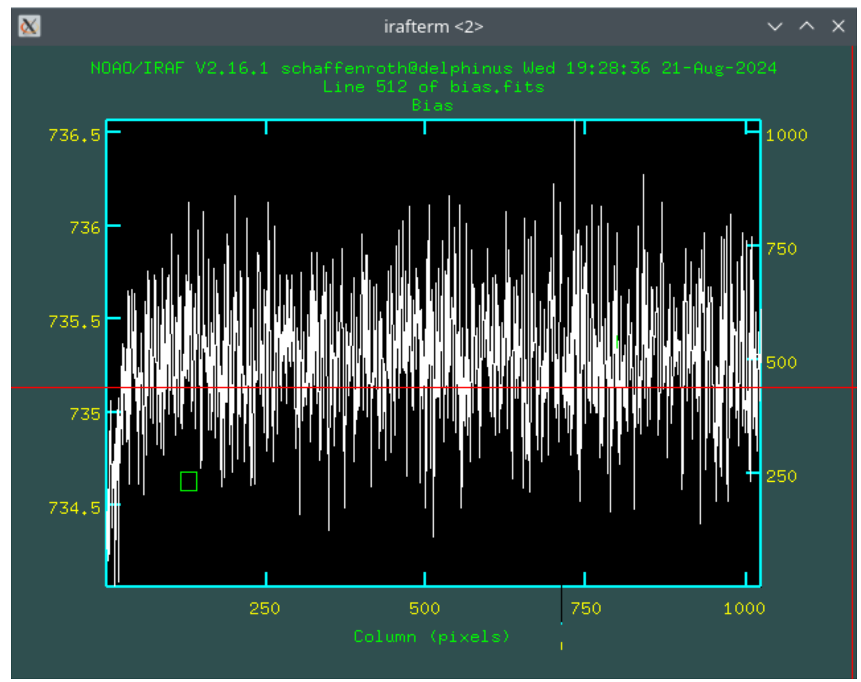
```
PACKAGE = immatch
   TASK = imcombine

input   = []         @bias.lis  List of images to combine
output  =            bias.fits  List of output images
(headers=                    ) List of header files (optional)
(bpmasks=                    ) List of bad pixel masks (optional)
(rejmask=                    ) List of rejection masks (optional)
(nrejmas=                    ) List of number rejected masks (optional)
(expmask=                    ) List of exposure masks (optional)
(sigmas =                    ) List of sigma images (optional)
(imcmb  =                  $I) Keyword for IMCMB keywords
(logfile=              STDOUT) Log file

(combine=             average) Type of combine operation
(reject =             sigclip) Type of rejection
(project=                  no) Project highest dimension of input images?
(outtype=                real) Output image pixel datatype
(outlimi=                    ) Output limits (x1 x2 y1 y2 ...)
(offsets=                none) Input image offsets
(masktyp=                none) Mask type
(maskval=                   0) Mask value
(blank  =                  0.) Value if there are no pixels

(scale  =                none) Image scaling
(zero   =                none) Image zero point offset
(weight =                none) Image weights
(statsec=                    ) Image section for computing statistics
(expname=                    ) Image header exposure time keyword

(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 neighbor rejection
(mode   =                  ql)
```

Determine bias level

Most CCDs are so good that the bias does not have a small-scale structure. In this case, you can just ignore the structure and just subtract the average value in the bias frame

We can derive it using **imstat**, save the mean value and then just subtract this value from the science and the flat field frame later using **imarith**



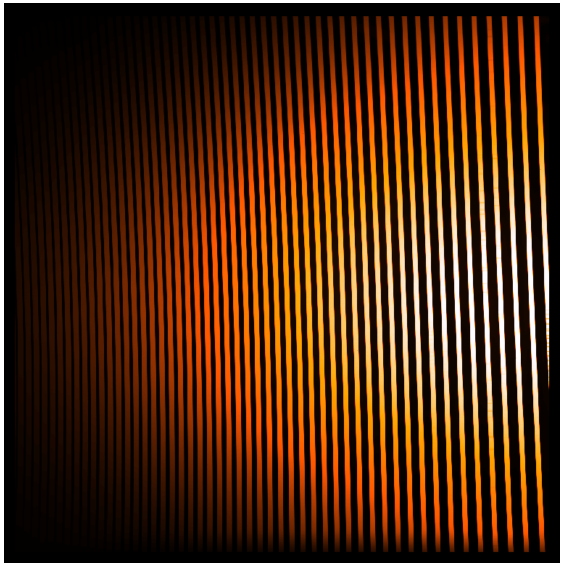


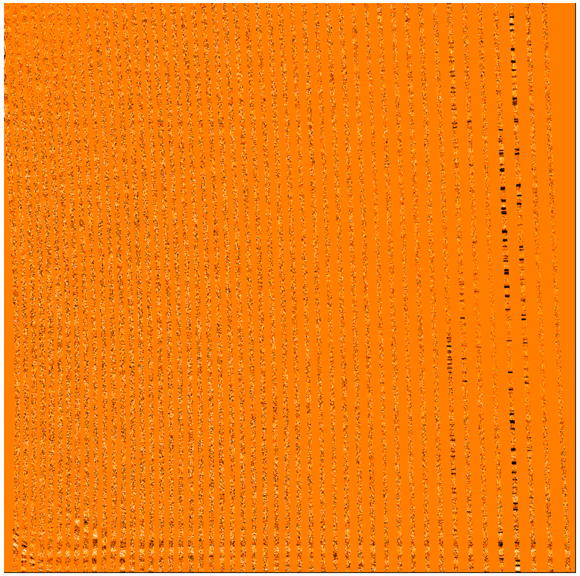

```
echelle> imstat bias.fits
#          IMAGE       NPIX      MEAN    STDDEV       MIN       MAX
       bias.fits    1048576      735.    0.7174     730.3      737.
```

Making an flat for an Echelle spectrum

How to make a flat I?

- Problem: we can not simply divide by the flat, because the flat has a lot of flux in the orders but nothing outside the orders and dividing by zero does nor really give you great results. We thus need a flat that contains 1.0 between the orders and the relative pixel-to-pixel variations (values like 0.8...1.2) in the orders.
- The first step is to average all flats (make sure that you do not have cosmics) and subtract the bias.

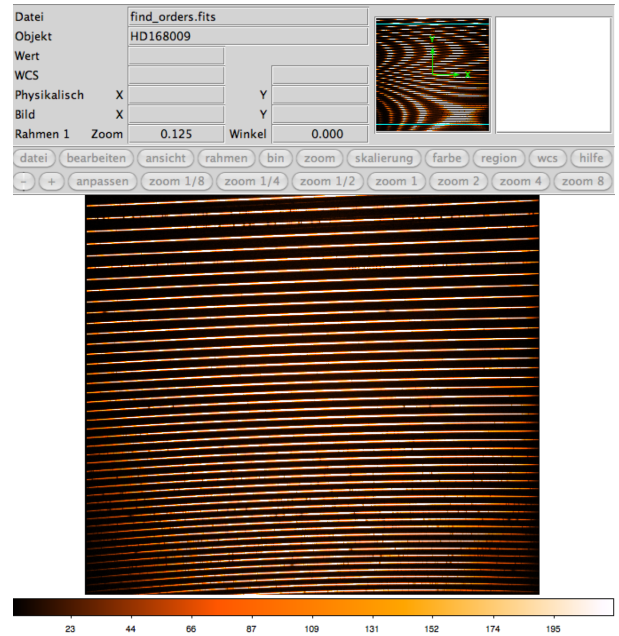




```
echelle> ls *Flat.fits > flat.lis
echelle> imcombine @flat.lis flat.fits

Aug 21 15:54: IMCOMBINE
  combine = average, scale = none, zero = none, weight = none
  reject = sigclip, mclip = yes, nkeep = 1
  lsigma = 3., hsigma = 3.
  blank = 0.


  echelle> imarith flat.fits - 735 flat_b.fits
```

How to make a flat II?

- In order to make a flat, IRAF has to know were the orders are. We also need this information later for the stellar spectra. I call the spectrum of the brightest star “find_orders.fits”.
- Take a spectrum of a very bright star and use “apall” and switch on all interactive modes, except the width of the orders (measure them and define them), you can use the one given in the calibration files

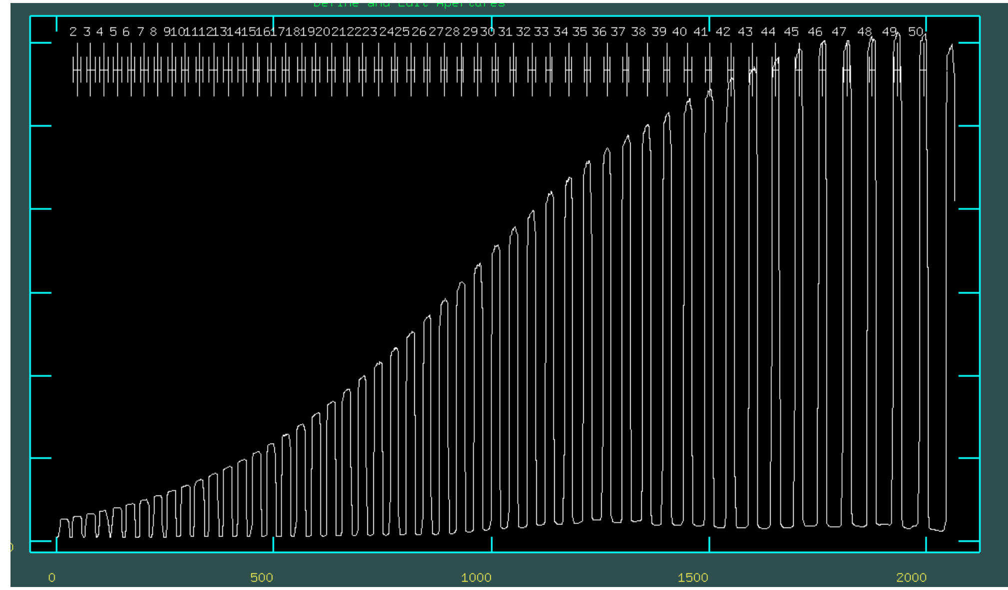



```
                                  I R A F
                    Image Reduction and Analysis Facility
PACKAGE = echelle
   TASK = apall

input   =       find_ordes.fits  List of input images
(output =            dummy.fits) List of output spectra
(apertur=                      ) Apertures
(format =                echelle) Extracted spectra format
(referen=                      ) List of aperture reference images
(profile=                      ) List of aperture profile images

(interac=                   yes) Run task interactively?
(find   =                   yes) Find apertures?
(recente=                   yes) Recenter apertures?
(resize =                    no) Resize apertures?
(edit   =                   yes) Edit apertures?
(trace  =                   yes) Trace apertures?
(fittrac=                   yes) Fit the traced points interactively?
(extract=                   yes) Extract spectra?
(extras =                    no) Extract sky, sigma, etc.?
```

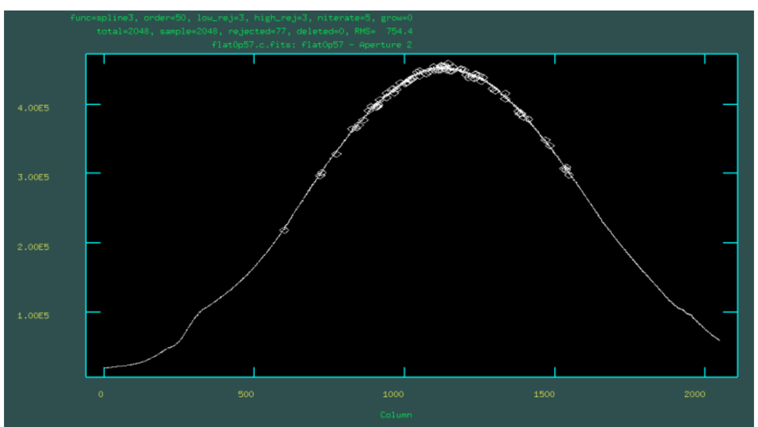
```
                 # AUTOMATIC FINDING AND ORDERING PARAMETERS

nfind   =                    48  Number of apertures to be found automatically
(minsep =                    5.) Minimum separation between spectra
(maxsep =               100000.) Maximum separation between spectra
(order  =            increasing) Order of apertures


                 # TRACING PARAMETERS

(t_nsum =                   100) Number of dispersion lines to sum
(t_step =                   113) Tracing step
(t_nlost=                     3) Number of consecutive times profile is lost before quitting
(t_funct=              legendre) Trace fitting function
(t_order=                     7) Trace fitting function order
(t_sampl=                     *) Trace sample regions
(t_naver=                     1) Trace average or median
(t_niter=                     0) Trace rejection iterations
(t_low_r=                    3.) Trace lower rejection sigma
(t_high_=                    3.) Trace upper rejection sigma
(t_grow =                    0.) Trace rejection growing radius
```

How to make a flat III?

- After the orders are defined using the “find_orders.fits” frame:
- Use “apflatten” to make flat. Make sure that you have the width of the orders defined and use the correct fitting function.

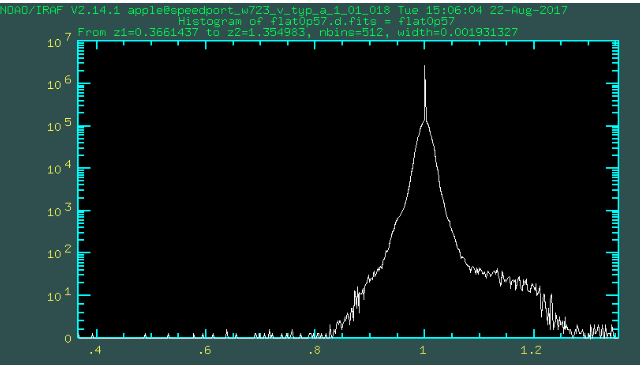


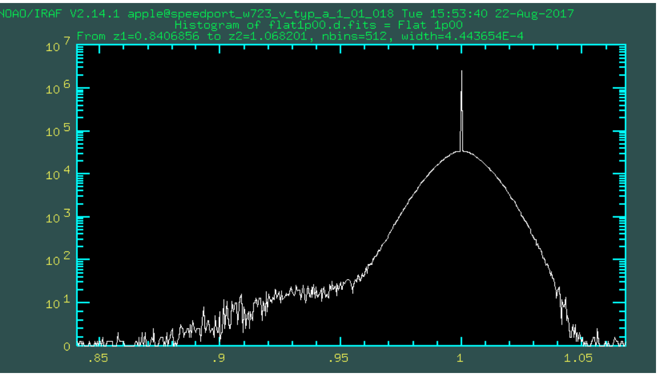

```
PACKAGE = echelle
   TASK = aptrace

input   =                        List of input images to trace
(apertur=                      ) Apertures
(referen=                      ) List of reference images
(interac=                   yes) Run task interactively?
(find   =                   yes) Find apertures?
(recente=                    no) Recenter apertures?
(resize =                    no) Resize apertures?
(edit   =                    no) Edit apertures?
(trace  =                   yes) Trace apertures?
(fittrac=                   yes) Fit the traced points interactively?

(line   =                 INDEF) Starting dispersion line
(nsum   =                    10) Number of dispersion lines to sum
(step   =                    10) Tracing step
(nlost  =                     3) Number of consecutive times profile is lost before quitting

(functio=              legendre) Trace fitting function
(order  =                     7) Trace fitting function order
(sample =                     *) Trace sample regions
(naverag=                     1) Trace average or median
(niterat=                     0) Trace rejection iterations
(low_rej=                    3.) Trace lower rejection sigma
(high_re=                    3.) Trace upper rejection sigma
(grow   =                    0.) Trace rejection growing radius
(mode   =                    ql)
```

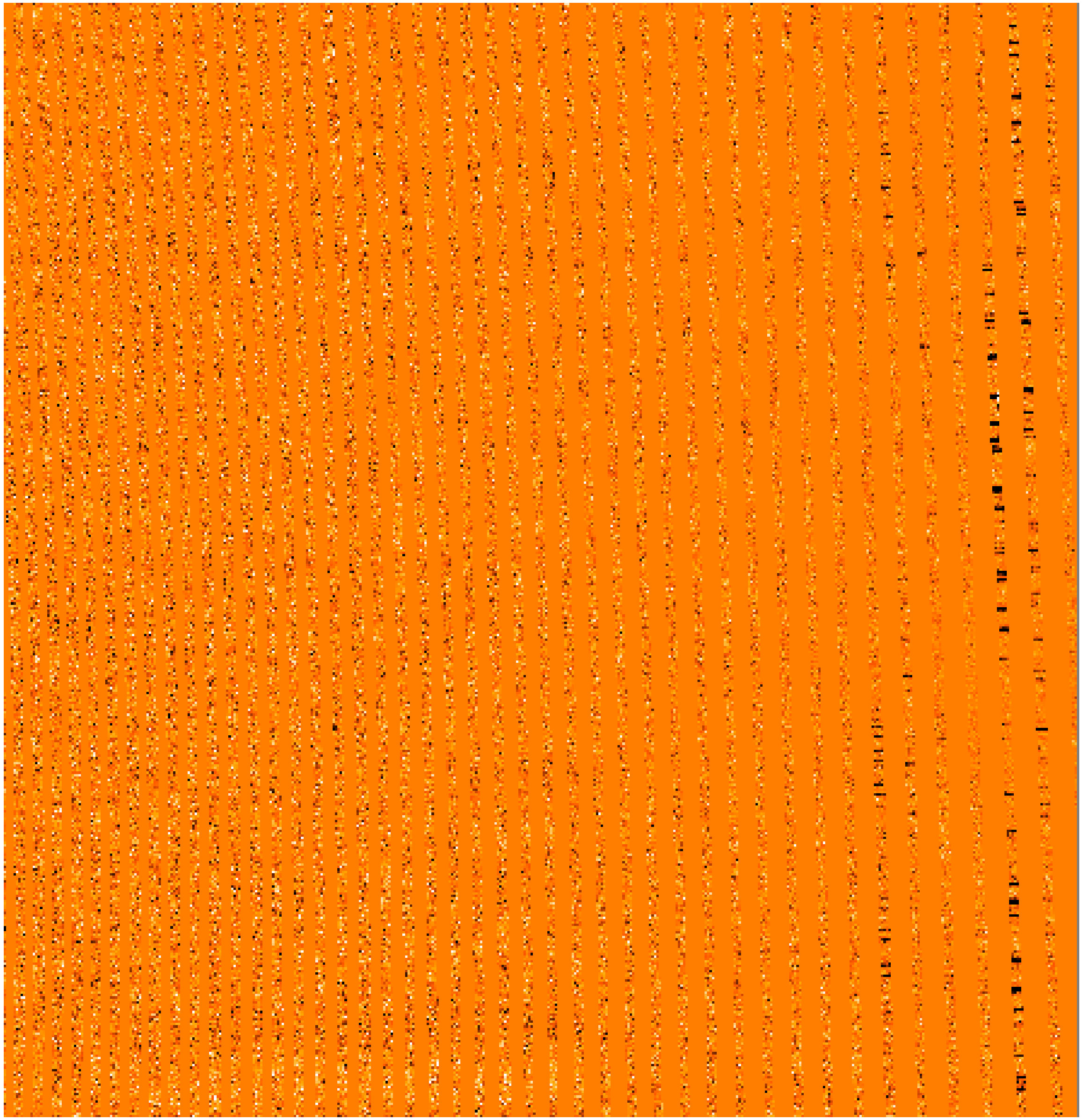



```
PACKAGE = echelle
   TASK = apflatten

input   =          flat_b.fits  List of images to flatten
output  =      masterflat.fits  List of output flatten images
(apertur=                     ) Apertures
(referen=      find_orders.fits) List of reference images

(interac=                  yes) Run task interactively?
(find   =                  yes) 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=                   no) Fit normalization spectra interactively?

(line   =                INDEF) 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  =                  yes) 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=             legendre) Fitting function for normalization spectra
(order  =                   20) Fitting function order
(sample =                    *) Sample regions
(naverag=                    1) Average or median
(niterat=                    0) Number of rejection iterations
(low_rej=                   3.) Lower rejection sigma
(high_re=                   3.) High upper rejection sigma
(grow   =                   0.) Rejection growing radius
(mode   =                    q)
```

Check with “imhist” that there no strange values, use “imreplace” if needed.







```
ecl> imreplace flat1p00.d.fits value=1.0 lower=1.07 upper=INDEF
ecl> imreplace flat1p00.d.fits value=1.0 lower=INDEF upper=0.84
```

Preparing the science frames

First steps

- Subtract bias:
- Science_frame - value_of_overscan = science_frame_corrected.

- Devide by flat
- Science_frame / Flat = Science_frame_corrected

```
echelle> imstat bias.fits
#               IMAGE      NPIX      MEAN    STDDEV       MIN       MAX
            bias.fits   1048576      735.    0.7174     730.3      737.
```

```
echelle> imarith HD10700_2024-07-27T10_09.fits - 735 HD10700_2024-07-27T10_09a.fits
echelle> imarith HD10700_2024-07-27T10_09a.fits / masterflat.fits HD10700_2024-07-27T10_09b.fits
```

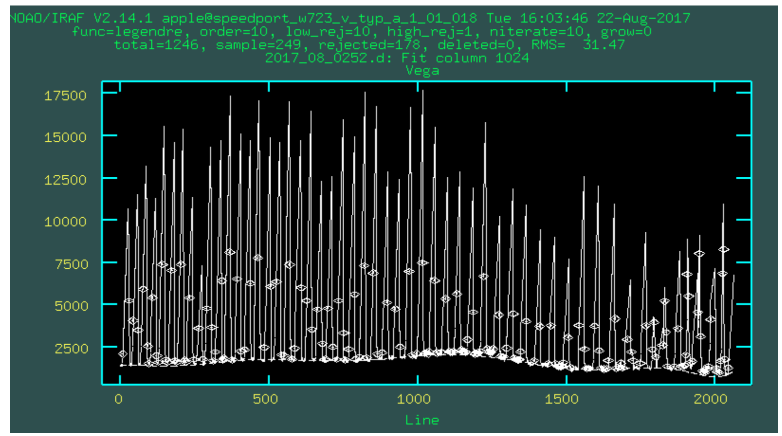
- same needs to be done to the ThAr lamps later!!

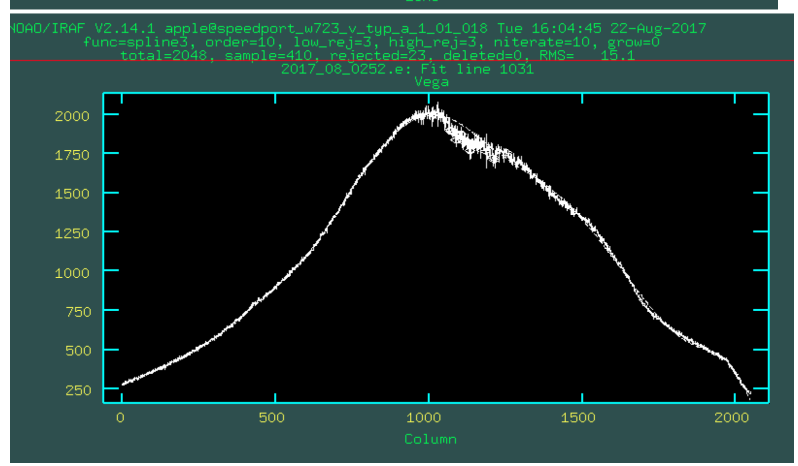
Subtracting the scattered light

- Subtracting the scattered light
- The scattered light is created by the grating. The light is a kind of diffuse background that is distributed over the whole frame (brightest in the middle). It is subtracted using the tool “apscatter” which measures the flux between the orders.
- A fit is made in X and Y-direction using a tool that is very similar to fit1d.

```
PACKAGE = echelle
   TASK = apscat1

(apertur= )apscatter.apertures) >apall.apertures
(functio=                spline3) Fitting function
(order  =                      7) Order of fitting function
(sample =                      *) Sample points to use in fit
(naverag=                      1) Number of points in sample averaging
(low_rej=                     5.) Low rejection in sigma of fit
(high_re=                     2.) High rejection in sigma of fit
(niterat=                      5) Number of rejection iterations
(grow   =                     0.) Rejection growing radius in pixels
(mode   =                     ql)
```

```
PACKAGE = echelle
   TASK = apscatter

input   = HD10700_2024-07-27T10_09b.fits  List of input images to subtract scattered light
output  = HD10700_2024-07-27T10_09c.fits  List of output corrected images
(apertur=                        ) Apertures
(scatter=                        ) List of scattered light images (optional)
(referen=             find_orders) List of aperture reference images

(interac=                     yes) Run task interactively?
(find   =                      no) Find apertures?
(recente=                      no) Recenter apertures?
(resize =                      no) Resize apertures?
(edit   =                      no) Edit apertures?
(trace  =                      no) Trace apertures?
(fittrac=                      no) Fit the traced points interactively?
(subtrac=                     yes) Subtract scattered light?
(smooth =                     yes) Smooth scattered light along the dispersion?
(fitscat=                     yes) Fit scattered light interactively?
(fitsmoo=                     yes) Smooth the scattered light interactively?

(line   =                   INDEF) Dispersion line
(nsum   =                      10) Number of dispersion lines to sum or median
(buffer =                      1.) Buffer distance from apertures
(apscat1=                        ) Fitting parameters across the dispersion
(apscat2=                        ) Fitting parameters along the dispersion
(mode   =                      ql)
```

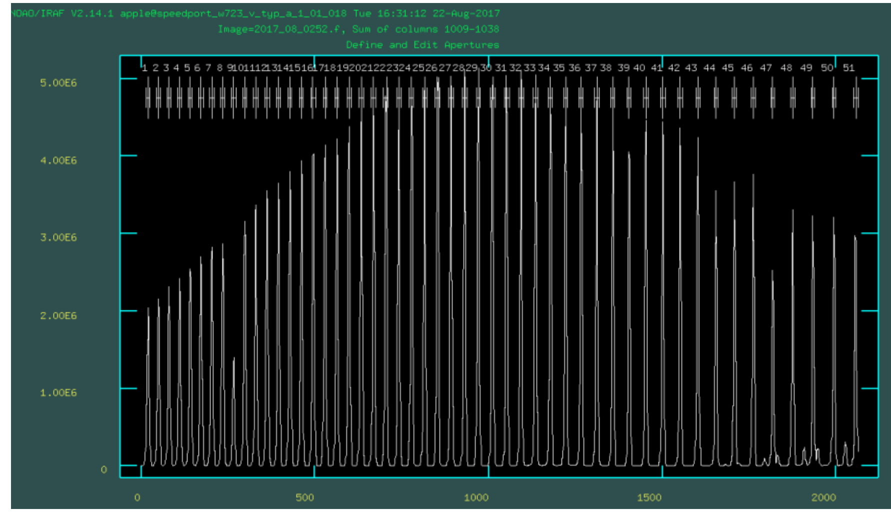




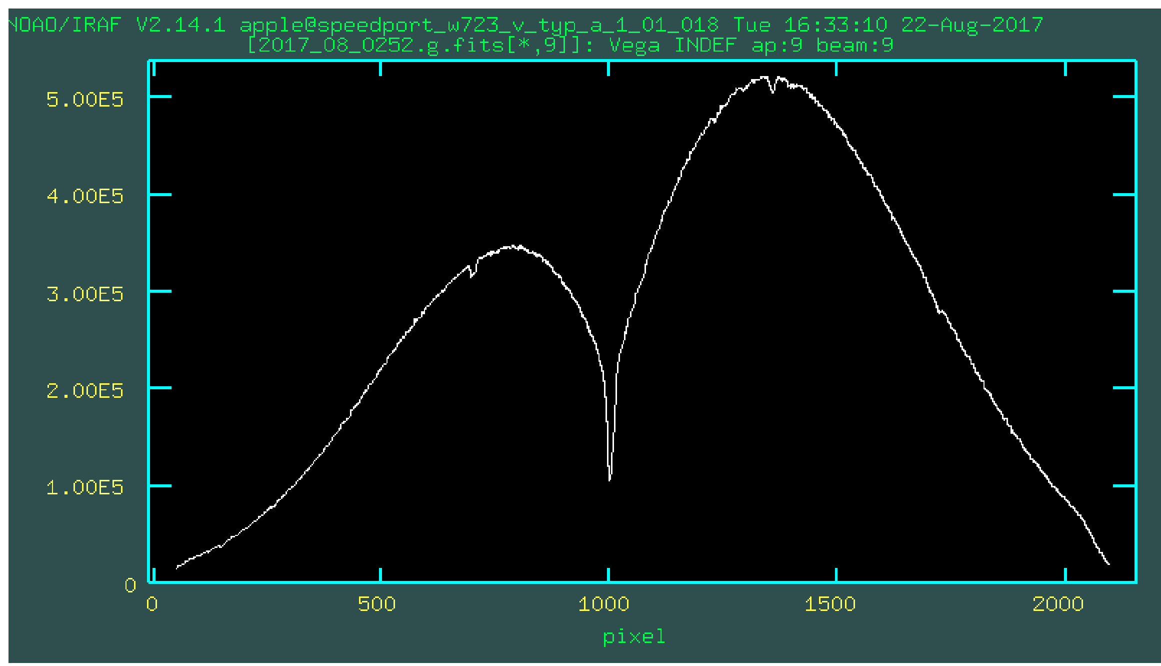
Extracting the spectra

Extracting the spectra is quick, because we have already identified were the orders are. If you want to fine-adjust it, set “find” and “recenter” to yes.

Now we have extracted spectra were each order is one line.




```
PACKAGE = echelle
   TASK = apall

input   = HD10700_2024-07-27T10_09c.fits  List of input images
(output = HD10700_2024-07-27T10_09d.fits) List of output spectra
(apertur=                            ) Apertures
(format =                     echelle) Extracted spectra format
(referen=                 find_orders) List of aperture reference images
(profile=                            ) List of aperture profile images

(interac=                         yes) Run task interactively?
(find   =                         yes) Find apertures?
(recente=                         yes) Recenter apertures?
(resize =                          no) Resize apertures?
(edit   =                         yes) Edit apertures?
(trace  =                          no) Trace apertures?
(fittrac=                          no) Fit the traced points interactively?
(extract=                         yes) Extract spectra?
(extras =                          no) Extract sky, sigma, etc.?
(review =                         yes) Review extractions?

(line   =                       INDEF) Dispersion line
(nsum   =                          10) Number of dispersion lines to sum or median

                                       # DEFAULT APERTURE PARAMETERS

(lower  =                         -5.) Lower aperture limit relative to center
(upper  =                          5.) Upper aperture limit relative to center
(apidtab=                            ) Aperture ID table (optional)

                                       # DEFAULT BACKGROUND PARAMETERS

(b_funct=                   chebyshev) Background function
```



The wavelength calibration

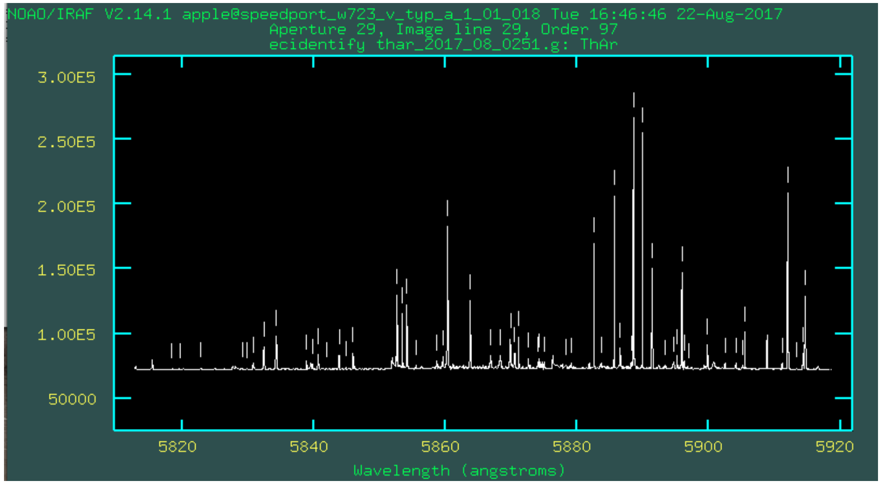
- The wavelength calibration has several steps.
- The first one is to extract the thar-spectrum with apall in the same way as the science frame (bias, flat and scattered light- correction also has to be done first the same way as for the science frames)

```
PACKAGE = echelle
   TASK = apall

input   = 2024-07-26T22_42_43.601769-Comp.c.fits  List of input images
(output = 2024-07-26T22_42_43.601769-Comp.d.fits) List of output spectra
(apertur=                      ) Apertures
(format =                echelle) Extracted spectra format
(referen=            find_orders) List of aperture reference images
(profile=                      ) List of aperture profile images

(interac=                    no) Run task interactively?
(find   =                    no) Find apertures?
(recente=                    no) Recenter apertures?
(resize =                    no) Resize apertures?
(edit   =                    no) Edit apertures?
(trace  =                    no) Trace apertures?
(fittrac=                    no) Fit the traced points interactively?
(extract=                   yes) Extract spectra?
(extras =                    no) Extract sky, sigma, etc.?
(review =                   yes) Review extractions?

(line   =                 INDEF) Dispersion line
(nsum   =                    10) Number of dispersion lines to sum or median

                                 # DEFAULT APERTURE PARAMETERS

(lower  =                   -5.) Lower aperture limit relative to center
(upper  =                    5.) Upper aperture limit relative to center
(apidtab=                      ) Aperture ID table (optional)
```

The second step is to identify the lines using **ecidentify**. A good fit has a scatter of 0.03 AA

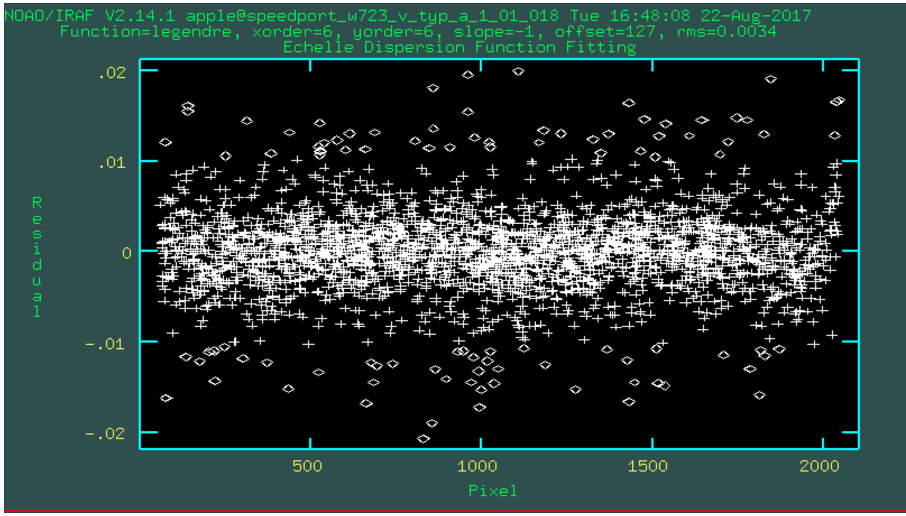
In our case we have already done that!

So you can use **ecreidentify** for your comparison lamps and use the solution given in the calibration files (database/ecThAr)




```
PACKAGE = echelle
   TASK = ecreidentify

images  = 2024-07-26T22_42_43.601769-Comp.d.fits  Spectra to be reidentified
referenc=                   ThAr  Reference spectrum
(shift  =                     0.) Shift to add to reference features
(cradius=                     5.) Centering radius
(thresho=                    10.) Feature threshold for centering
(refit  =                    yes) Refit coordinate function?
(databas=               database) Database
(logfile=         STDOUT,logfile) List of log files
(mode   =                     ql)
```

```
ECREIDENTIFY: NOAO/IRAF V2.7 seaman@puppis Mon 09:15:21 27-Jun-88
  Reference image = f033.ec, Refit = no
             Image     Found  Pix Shift  User Shift  Z Shift       RMS
           f043.ec   561/561       0.11       -1.07  -1.9E-6    0.0131
```



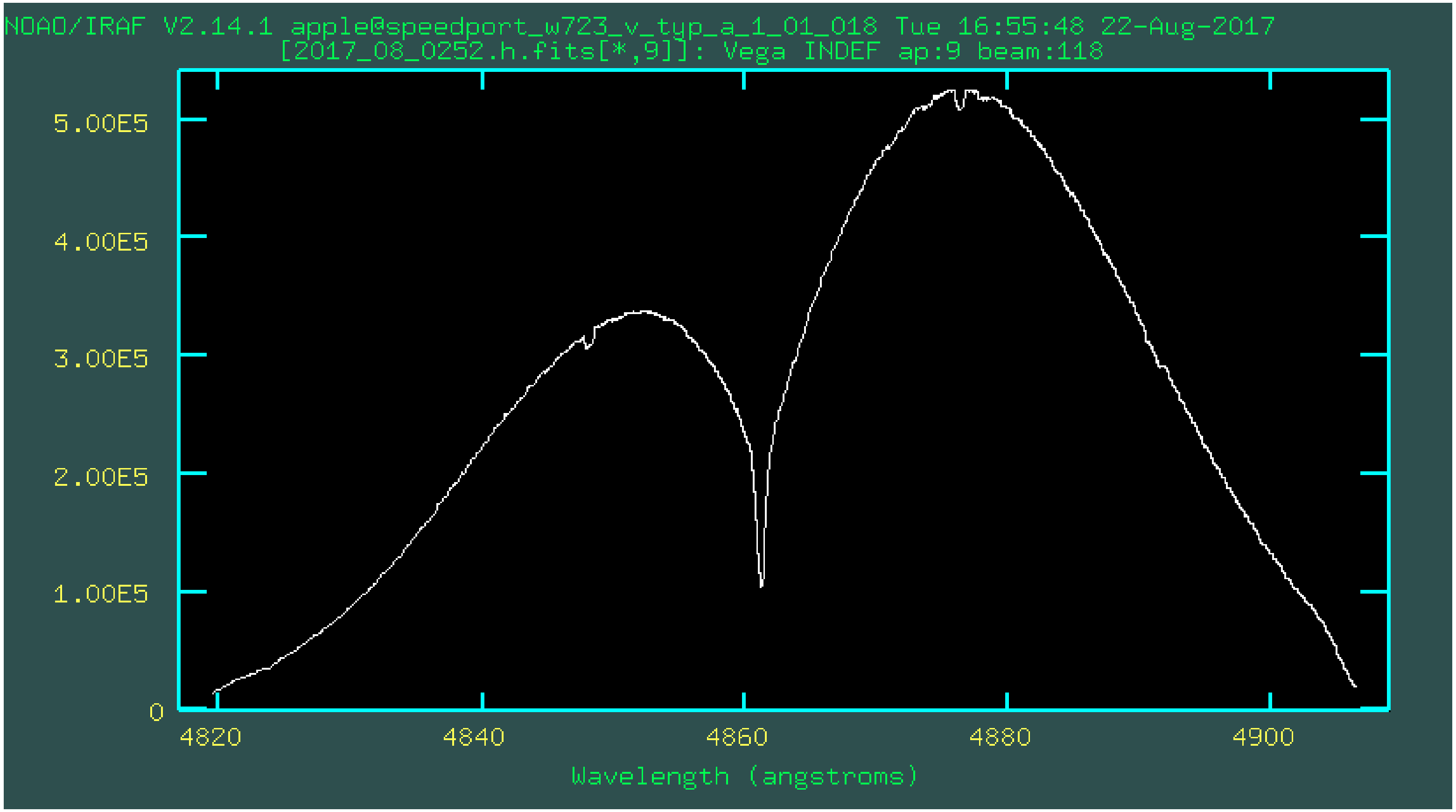
Now assign the ThAr you obtained during the observing night to the science frame using **refspectra**, and do the wavelength-calibration with **dispcor**.

```
PACKAGE = echelle
   TASK = refspectra

input   = HD10700_2024-07-27T10_09d.fits  List of input spectra
(referen= 2024-07-26T22_42_43.601769-Comp.d.fits) List of reference spectra
(apertur=                        ) Input aperture selection list
(refaps =                        ) Reference aperture selection list
(ignorea=                     yes) Ignore input and reference apertures?
(select =                 average) Selection method for reference spectra
(sort   =                        ) Sort key
(group  =                        ) Group key
(time   =                      no) Is sort key a time?
(timewra=                     17.) Time wrap point for time sorting
(overrid=                     yes) Override previous assignments?
(confirm=                     yes) Confirm reference spectrum assignments?
(assign =                     yes) Assign the reference spectra to the input spectrum?
(logfile=          STDOUT,logfile) List of logfiles
(verbose=                     yes) Verbose log output?
answer  =                     yes  Accept assignment?
(mode   =                      ql)
```

```
PACKAGE = echelle
   TASK = dispcor

input   = HD10700_2024-07-27T10_09d.fits  List of input spectra
output  = HD10700_2024-07-27T10_09e.fits  List of output spectra
(lineari=                     yes) Linearize (interpolate) spectra?
(databas=                database) Dispersion solution database
(table  =                        ) Wavelength table for apertures
(w1     =                   INDEF) Starting wavelength
(w2     =                   INDEF) Ending wavelength
(dw     =                   INDEF) Wavelength interval per pixel
(nw     =                   INDEF) Number of output pixels
(log    =                      no) Logarithmic wavelength scale?
(flux   =                     yes) Conserve total flux?
(blank  =                      0.) Output value of points not in input
(samedis=                     yes) Same dispersion in all apertures?
(global =                      no) Apply global defaults?
(ignorea=                     yes) Ignore apertures?
(confirm=                      no) Confirm dispersion coordinates?
(listonl=                      no) List the dispersion coordinates only?
(verbose=                     yes) Print linear dispersion assignments?
(logfile=                        ) Log file
(mode   =                      ql)
```

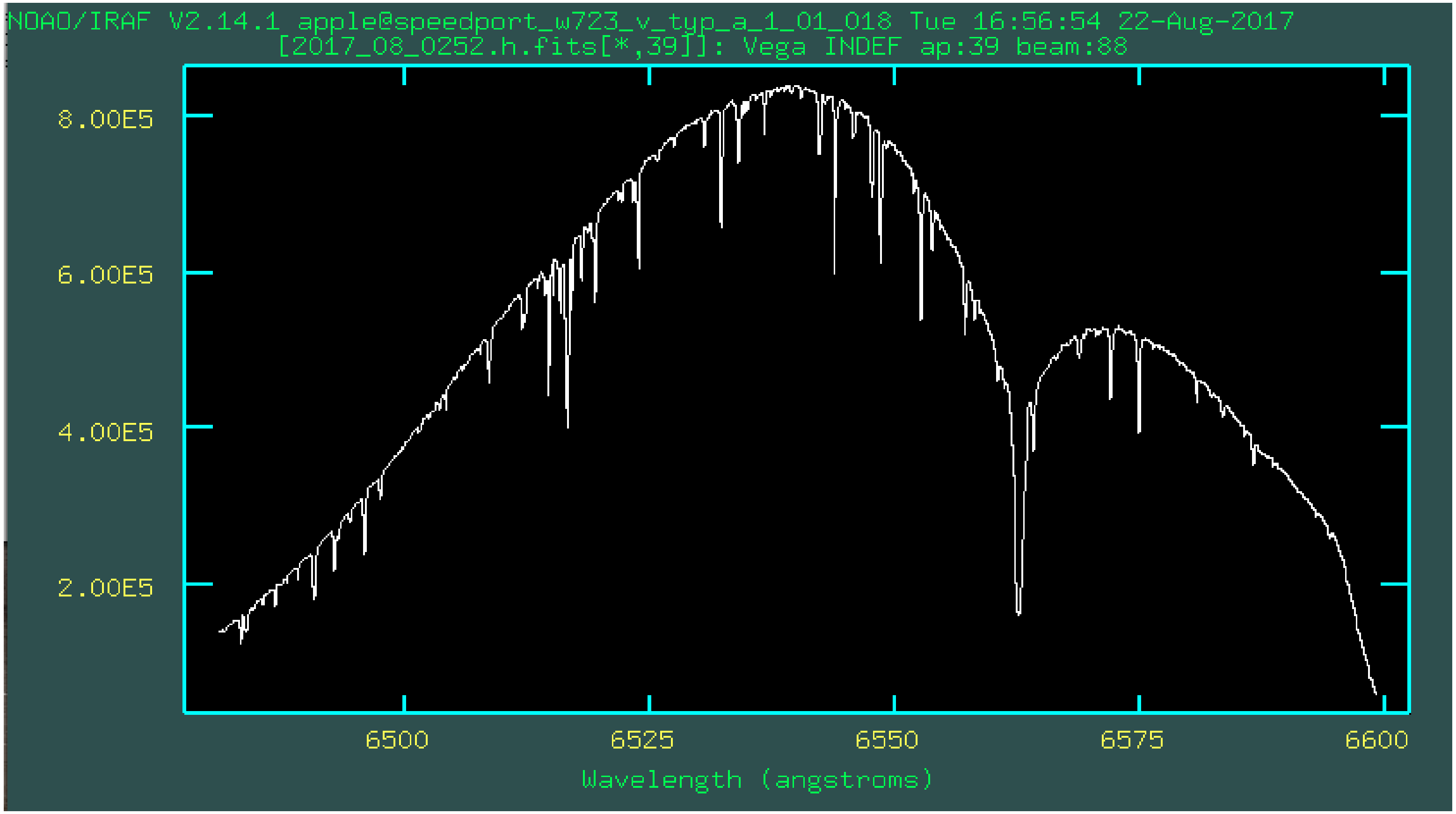
```
HD10700_2024-07-27T10_09d.fits: REFSPEC1 = '2024-07-26T22_42_43.601769-Comp.d 1.'
HD10700_2024-07-27T10_09e.fits: ap = 1, w1 = 4005.401, w2 = 4082.349, dw = 0.075218, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 2, w1 = 4044.209, w2 = 4121.937, dw = 0.075981, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 3, w1 = 4083.781, w2 = 4162.305, dw = 0.076758, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 4, w1 = 4124.141, w2 = 4203.476, dw = 0.077551, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 5, w1 = 4165.312, w2 = 4245.473, dw = 0.078359, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 6, w1 = 4207.317, w2 = 4288.322, dw = 0.079183, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 7, w1 = 4250.183, w2 = 4332.047, dw = 0.080024, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 8, w1 = 4293.934, w2 = 4376.676, dw = 0.080881, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 9, w1 = 4338.599, w2 = 4422.236, dw = 0.081756, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 10, w1 = 4384.207, w2 = 4468.756, dw = 0.082649, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 11, w1 = 4430.786, w2 = 4516.268, dw =  0.08356, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 12, w1 = 4478.369, w2 = 4564.802, dw =  0.08449, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 13, w1 = 4526.988, w2 = 4614.392, dw = 0.085439, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 14, w1 = 4576.676, w2 = 4665.073, dw = 0.086409, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 15, w1 =  4627.47, w2 =  4716.88, dw = 0.087399, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 16, w1 = 4679.407, w2 = 4769.851, dw = 0.088411, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 17, w1 = 4732.526, w2 = 4824.027, dw = 0.089444, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 18, w1 = 4786.867, w2 = 4879.448, dw =   0.0905, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 19, w1 = 4842.473, w2 = 4936.159, dw =  0.09158, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 20, w1 = 4899.388, w2 = 4994.203, dw = 0.092683, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 21, w1 = 4957.661, w2 =  5053.63, dw = 0.093811, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 22, w1 = 5017.339, w2 = 5114.488, dw = 0.094965, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 23, w1 = 5078.474, w2 = 5176.831, dw = 0.096146, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 24, w1 =  5141.12, w2 = 5240.713, dw = 0.097354, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 25, w1 = 5205.335, w2 = 5306.193, dw = 0.098591, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 26, w1 = 5271.176, w2 =  5373.33, dw = 0.099857, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 27, w1 = 5338.708, w2 = 5442.189, dw = 0.101154, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 28, w1 = 5407.996, w2 = 5512.836, dw = 0.102483, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 29, w1 =  5479.11, w2 = 5585.343, dw = 0.103845, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 30, w1 = 5552.121, w2 = 5659.784, dw = 0.105242, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 31, w1 = 5627.108, w2 = 5736.236, dw = 0.106674, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 32, w1 = 5704.152, w2 = 5814.784, dw = 0.108145, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 33, w1 = 5783.337, w2 = 5895.514, dw = 0.109655, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 34, w1 = 5864.755, w2 = 5978.518, dw = 0.111206, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 35, w1 =   5948.5, w2 = 6063.895, dw =   0.1128, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 36, w1 = 6034.674, w2 = 6151.746, dw =  0.11444, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 37, w1 = 6123.383, w2 = 6242.182, dw = 0.116127, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 38, w1 = 6214.742, w2 = 6335.317, dw = 0.117865, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 39, w1 = 6308.868, w2 = 6431.275, dw = 0.119655, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 40, w1 = 6405.891, w2 = 6530.186, dw =   0.1215, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 41, w1 = 6505.944, w2 = 6632.187, dw = 0.123404, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 42, w1 = 6609.173, w2 = 6737.426, dw =  0.12537, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 43, w1 = 6715.728, w2 =  6846.06, dw = 0.127402, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 44, w1 = 6825.773, w2 = 6958.254, dw = 0.129502, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 45, w1 = 6939.482, w2 = 7074.188, dw = 0.131677, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 46, w1 =  7057.04, w2 = 7194.049, dw = 0.133929, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 47, w1 = 7178.643, w2 = 7318.042, dw = 0.136264, nw = 1024
HD10700_2024-07-27T10_09e.fits: ap = 48, w1 = 7304.505, w2 = 7446.383, dw = 0.138688, nw = 1024
```

Hbetha



to look at the reduced spectra: use **splot;** you can change orders using ")" and "("

Halpha



to look at the reduced spectra: use **splot;** you can change orders using ")" and "("

Measuring the RV using the cross-correlation function

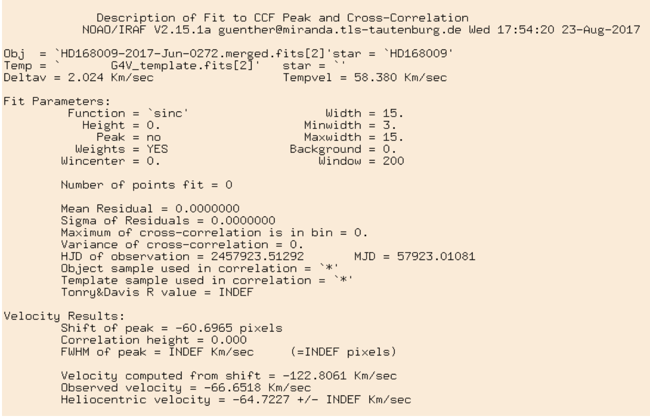
(we want to use viper to do that, but this could also be done for comparison, if time allows)

The header has to be corrected

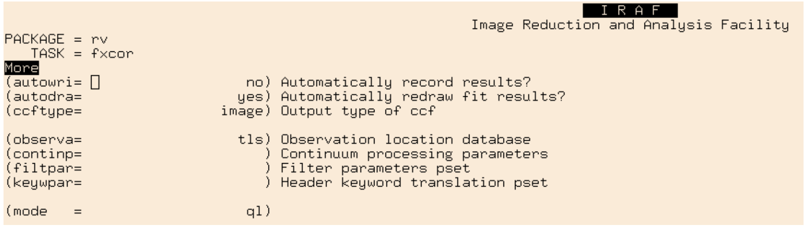
```
                                                                      I R A F
                                                          Image Reduction and Analysis Facility
PACKAGE = rv
   TASK = keywpars

(ra     =                POSTN-RA) Right Ascension keyword
(dec    =                POSTN-DE) Declination keyword
(ut     =                      UT) UT of observation keyword
(utmiddl=                   UTMID) UT of mid-point of observation keyword
(exptime=                EXP-TIME) Exposure time keyword
(epoch  =                   EPOCH) Epoch of observation keyword
(date_ob=                DATE-OBS) Date of observation keyword
```

Use fxcor to calculate the RV

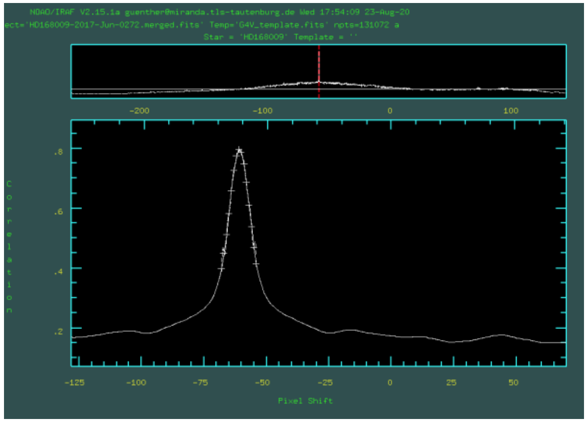
```
                                   I R A F
                   Image Reduction and Analysis Facility
PACKAGE = rv
   TASK = fxcor

objects = HD168009-2017-Jun-0272.merged.fits  List of object spectra
template=    G4V_template.fits  List of template spectra
(apertur=                    *) Apertures to be used
(cursor =                     ) Graphics input cursor

(continu=                 both) Continuum subtract spectra?
(filter =                 none) Fourier filter the spectra?
(rebin  =               object) Rebin to which dispersion?
(pixcorr=                   no) Do a pixel-only correlation?
(osample=                    *) Object regions to be correlated ('*' => all)
(rsample=                    *) Template regions to be correlated
(apodize=                  0.1) Apodize end percentage

(functio=                 sinc) Function to fit correlation
(width  =                  15.) Width of fitting region in pixels
(height =                   0.) Starting height of fit
(peak   =                   no) Is height relative to ccf peak?
(minwidt=                   3.) Minimum width for fit
(maxwidt=                  15.) Maximum width for fit
(weights=                   1.) Power defining fitting weights
(backgro=                   0.) Background level for fit
(window =                 200.) Size of window in the correlation plot
(wincent=                   0.) Center of peak search window

(output =                fxcor) Root spool filename for output
(verbose=                 long) Verbose output to spool file?
(imupdat=                   no) Update the image header?
(graphic=             stdgraph) Graphics output device

(interac=                  yes) Interactive graphics?
```



```
             Description of Fit to CCF Peak and Cross-Correlation
           NOAO/IRAF V2.15.1a guenther@miranda.tls-tautenburg.de Wed 17:54:20 23-Aug-2017

Obj  = `HD168009-2017-Jun-0272.merged.fits[2]'star = `HD168009'
Temp = `       G4V_template.fits[2]'   star = `'
Deltav = 2.024 Km/sec                  Tempvel = 58.380 Km/sec

Fit Parameters:
         Function = `sinc'                    Width = 15.
           Height = 0.                     Minwidth = 3.
             Peak = no                     Maxwidth = 15.
          Weights = YES                  Background = 0.
        Wincenter = 0.                       Window = 200

        Number of points fit = 0

        Mean Residual = 0.0000000
        Sigma of Residuals = 0.0000000
        Maximum of cross-correlation is in bin = 0.
        Variance of cross-correlation = 0.
        HJD of observation = 2457923.51292       MJD = 57923.01081
        Object sample used in correlation = `*'
        Template sample used in correlation = `*'
        Tonry&Davis R value = INDEF

Velocity Results:
        Shift of peak = -60.6965 pixels
        Correlation height = 0.000
        FWHM of peak = INDEF Km/sec     (=INDEF pixels)

        Velocity computed from shift = -122.8061 Km/sec
        Observed velocity = -66.6518 Km/sec
        Heliocentric velocity = -64.7227 +/- INDEF Km/sec
```

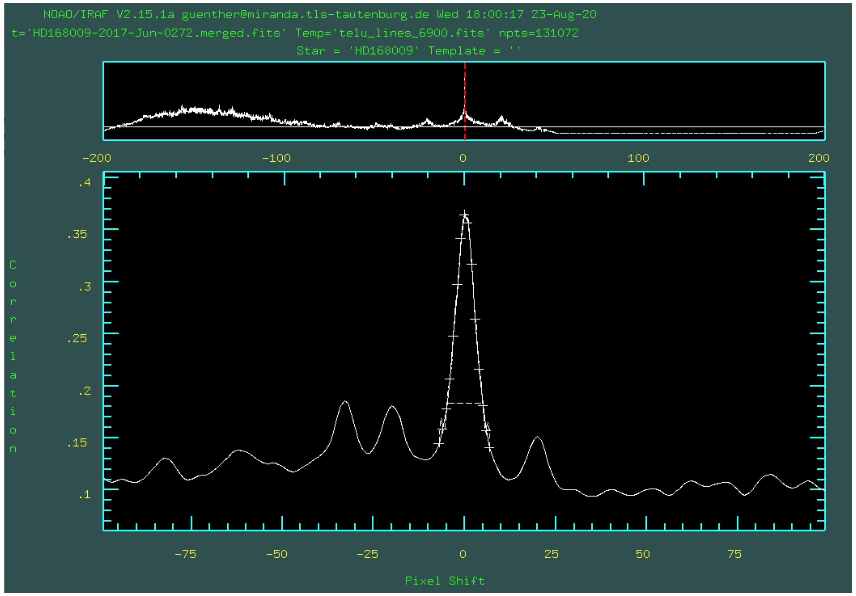


```
                                 I R A F
                  Image Reduction and Analysis Facility
PACKAGE = rv
   TASK = fxcor
More
(autowri=                   no) Automatically record results?
(autodra=                  yes) Automatically redraw fit results?
(ccftype=                image) Output type of ccf

(observa=                  tls) Observation location database
(continp=                     ) Continuum processing parameters
(filtpar=                     ) Filter parameters pset
(keywpar=                     ) Header keyword translation pset

(mode   =                   ql)
```



Computing the instrumental shift

RV = Rvstar - RVtelu




```
          Description of Fit to CCF Peak and Cross-Correlation
        NOAO/IRAF V2.15.1a guenther@miranda.tls-tautenburg.de Wed 18:00:26 23-Aug-2017

Obj   = `HD168009-2017-Jun-0272.merged.fits[2]'star = `HD168009'
Temp = `     telu_lines_6900.fits[2]'   star = `'
Deltav = 2.024 Km/sec                   Tempvel = INDEF Km/sec

Fit Parameters:
         Function = `sinc'                   Width = 15.
           Height = 0.                    Minwidth = 3.
             Peak = no                    Maxwidth = 15.
          Weights = YES                 Background = 0.
        Wincenter = 0.                      Window = 200

        Number of points fit = 15

        Mean Residual = 0.0000000
        Sigma of Residuals = 0.0000000
        Maximum of cross-correlation is in bin = 0.
        Variance of cross-correlation = 0.06143044
        HJD of observation = INDEF                  MJD = INDEF
        Object sample used in correlation = `*'
        Template sample used in correlation = `*'
        Tonry&Davis R value = 4.208615

Velocity Results:
        Shift of peak = 0.2520 pixels
        Correlation height = 0.366
        FWHM of peak = 19.48829 Km/sec  (=9.630037 pixels)

        Velocity computed from shift = 0.5100 Km/sec
        Observed velocity = INDEF Km/sec
        Heliocentric velocity = INDEF +/- 2.939 Km/sec
```