ARC Echelle Spectrograph (ARCES) Data Reduction Cookbook

Karen Kinemuchi
Apache Point Observatory, Sunspot, NM

Version 1.3 - September 2014

DISCLAIMER: I have made this write up to reduce my raw spectra from the ARCES spectrograph using IRAF/PyRAF. It is assumed the user of this cookbook is familiar with the IRAF environment and the basic steps of data reduction for CCD images. If not, please refer to the excellent manuals provided by the folks at IRAF[1]. My nomenclature for many of the input and output files were made to keep things simple and less confusing for me, and my choices may not be the best for you.

A data reduction pipeline exists at NMSU Department of Astronomy. On the APO wiki, a tar ball exists for the ARCES data reduction pipeline. Please contact Jean McKeever (email: jeanm12@nmsu.edu) for any questions regarding the NMSU ARCES reduction pipeline.

If you use this manual for reducing data from other echelle spectrographs, please keep in mind that Steps # 8, 9, 12, 14 are only for ARCES. Applying these steps may mess up your spectra and your analysis.

Please visually inspect your data throughout the processing to make sure you are getting things that are sensible. If not, step back and check if the previous task did not screw up or introduce something wrong.

To use this cookbook, all IRAF tasks are in ALL CAPS for readability. I also provide the parameters for many of the IRAF tasks for convenience, but the user is encouraged to play around with some of the parameters to optimize the steps for their individual science. Keep in mind, I do variable stars, so I have optimized the parameters for my science.

** Many of the tips and tricks for reducing ARCES data comes from the original data reduction many written by Julie Thorburn. It is recommended that this document be first read before using this cookbook. You can download this manual from the APO wiki page or from the APO website.

If you would like to cite this document in a publication, please cite Julie Thorburn's ARCES data reduction document along with this cookbook. A footnote to the APO wiki page should be sufficient, or an acknowledgement to the NMSU Astronomy Department for providing these documents.

If you have any questions or suggestions, feel free to contact me: kinemuchi@apo.nmsu.edu

[1]http://iraf.noao.edu/docs/spectra.html – *A User's Guide to CCD Reductions with IRAF* and *A Beginner's Guide to Using IRAF*

To get started, you will need to load the following IRAF packages:

NOAO, IMRED, CCDRED, ECHELLE, CRUTIL, ASTUTIL, IMAGES, IMGEOM, TWODSPEC, APEXTRACT, and ONEDSPEC

Also have a ds9 or saoimage window up to visually inspect all your data.

1. Setting the DISPAXIS header. Set the header keyword DISPAXIS to equal 1 by using HEDIT.

```
hedit *.fits dispaxis 1 add=yes verify=no show=yes update=yes
```

2. First-pass cosmic ray removal. Run COSMICRAYS for the first time on all the object and flat field images. Be sure to set fluxratio=10.0 in this first pass. See the Thorburn manual (see note in the introduction about this document) to get more details regarding this process of removing cosmic rays from your 2-D images.

The Input @-file contains obj and blue/red flat field images. Output @-files contain files called CR1*.fits

```
epar cosmicrays
        input = "@objflat"      List of images in which to detect cosmic rays
       output = "@outobjflat"   List of cosmic ray replaced output images (optional)
       answer = "yes"           Review parameters for a particular image?
     (crmasks = "")             List of bad pixel masks (optional)
   (threshold = 50.0)           Detection threshold above mean
   (fluxratio = 10.0)           Flux ratio threshold (in percent)
     (npasses = 20)             Number of detection passes
      (window = "5")            Size of detection window
 (interactive = no)             Examine parameters interactively?
       (train = no)             Use training objects?
     (objects = "")             Cursor list of training objects
    (savefile = "")             File to save train objects
    (plotfile = "")             Plot file
    (graphics = "stdgraph")     Interactive graphics output device
      (cursor = "")             Graphics cursor input
        (mode = "ql")
```

Cosmic ray removal can be done in many different ways. Another useful discussion of robustly removing cosmic rays can be found in Pych 2003 (arXiv:0311290). This paper can be found also on the APO wiki page.

3. Creating the master bias. Check if your biases have reasonable numbers by running IMSTAT and then make a master bias frame.

```
imstat bias*
#                IMAGE      NPIX      MEAN    STDDEV       MIN       MAX
       bias.0001.fits   4400704     1282.     7.136     1269.     4749.
       bias.0002.fits   4400704     1282.     6.011     1268.     4026.
       bias.0003.fits   4400704     1281.     5.942     1269.     4547.
       bias.0004.fits   4400704     1282.     7.495     1268.     5047.
       bias.0005.fits   4400704     1282.      7.43     1269.     5770.
       bias.0006.fits   4400704     1282.     46.48     1270.    39423.
       bias.0007.fits   4400704     1282.      6.75     1268.     4577.
       bias.0008.fits   4400704     1282.     41.07     1269.    37410.
       bias.0009.fits   4400704     1281.      15.8     1269.    25718.
       bias.0010.fits   4400704     1282.     7.311     1269.     6800.
```

To make the master bias frame, use ZEROCOMBINE. The input @-file contains all the bias frame taken during the observing run. The output is Zero.fits. Visually inspect this image to make sure no abnormal 2-d structure or other anomalies are seen.

```
 epar zerocombine
        input = "@inmkbias"     List of zero level images to combine
     (output = "Zero")          Output zero level name
    (combine = "average")       Type of combine operation
     (reject = "avsigclip")     Type of rejection
    (ccdtype = "")              CCD image type to combine
    (process = no)              Process images before combining?
     (delete = no)              Delete input images after combining?
    (clobber = no)              Clobber existing output image?
      (scale = "none")          Image scaling
    (statsec = "")              Image section for computing statistics
       (nlow = 0)               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 = 3.0)             Lower sigma clipping factor
     (hsigma = 3.0)             Upper sigma clipping factor
    (rdnoise = "7")             ccdclip: CCD readout noise (electrons)
       (gain = "3.8")           ccdclip: CCD gain (electrons/DN)
     (snoise = "0.")            ccdclip: Sensitivity noise (fraction)
      (pclip = -0.5)            pclip: Percentile clipping parameter
      (blank = 0.0)             Value if there are no pixels
       (mode = "ql")
```

4. Bias subtract, bad pixel fix, and trim. Run CCDPROC to perform the bias subtraction to the rest of the calibration and object images. In addition to the bias subtraction, do the bad pixel mask correction and trimsec on the arcs, objects, and flats. The input @-files are called inprocobj, inprocarc, and inprocflat, in my example. There are corresponding output @-files and all the processed files have a prefix "P".

Note that the trim section is set to [200:1850, 1:2048] for 1x1 binned data.

For the bad pixel mask, you can create your own or use ones provided on the APO wiki (file "badpix.txt").

```
lpar ccdproc
       images = "@inprocobj"    List of CCD images to correct
      (output = "@outprocobj")  List of output CCD images
     (ccdtype = "")             CCD image type to correct
   (max_cache = 0)              Maximum image caching memory (in Mbytes)
      (noproc = no)             List processing steps only?
      (fixpix = yes)            Fix bad CCD lines and columns?
    (overscan = no)             Apply overscan strip correction?
        (trim = yes)            Trim the image?
     (zerocor = yes)            Apply zero level correction?
     (darkcor = no)             Apply dark count correction?
     (flatcor = no)             Apply flat field correction?
    (illumcor = no)             Apply illumination correction?
   (fringecor = no)             Apply fringe correction?
     (readcor = no)             Convert zero level image to readout correction?
     (scancor = no)             Convert flat field image to scan correction?
    (readaxis = "line")         Read out axis (column|line)
     (fixfile = "badpix.txt")   File describing the bad lines and columns
     (biassec = "")             Overscan strip image section
     (trimsec = "[200:1850,1:2048]") Trim data section
        (zero = "Zero")         Zero level calibration image
        (dark = "")             Dark count calibration image
        (flat = "")             Flat field images
       (illum = "")             Illumination correction images
      (fringe = "")             Fringe correction images
  (minreplace = 1.0)            Minimum flat field value
    (scantype = "shortscan")    Scan type (shortscan|longscan)
       (nscan = 1)              Number of short scan lines
 (interactive = no)             Fit overscan interactively?
    (function = "legendre")     Fitting function
       (order = 3)              Number of polynomial terms or spline pieces
      (sample = "*")            Sample points to fit
    (naverage = 1)              Number of sample points to combine
```

```
    (niterate = 3)              Number of rejection iterations
  (low_reject = 3.0)            Low sigma rejection factor
 (high_reject = 3.0)            High sigma rejection factor
        (grow = 0.0)            Rejection growing radius
        (mode = "ql")
```

5. Second-pass cosmic ray removal. Rerun COSMICRAYS on all the flat fields and object frames. This time set the parameter fluxratio to 10.95. Don't forget to modify the input and output @-files, in case you do not want to overwrite your files! Using the example in my dataset, the input files have the prefix "P" and the output files will have the prefix "CR2".

```
epar cosmicrays
       input = "@objflat"       List of images in which to detect cosmic rays
      output = "@outobjflat"    List of cosmic ray replaced output images (optional)
      answer = "yes"            Review parameters for a particular image?
    (crmasks = "")              List of bad pixel masks (optional)
  (threshold = 50.0)            Detection threshold above mean
  (fluxratio = 10.95)           Flux ratio threshold (in percent)
    (npasses = 20)              Number of detection passes
     (window = "5")             Size of detection window
(interactive = no)              Examine parameters interactively?
      (train = no)              Use training objects?
    (objects = "")              Cursor list of training objects
   (savefile = "")              File to save train objects
   (plotfile = "")              Plot file
   (graphics = "stdgraph")      Interactive graphics output device
     (cursor = "")              Graphics cursor input
       (mode = "ql")
```

6. Create red and blue flat fields. Create separate master red and blue flat field frames using FLATCOMBINE. The input files are the flat fields that have been bias subtracted, trimsec'd, and cosmic ray cleaned twice. In my example, the input files have the prefix "CR2". The output files are "redFlat" and "blueFlat". Please visually inspect the blue and red master flats.

```
epar flatcombine
       input = "@inbflat"       List of flat field images to combine
     (output = "blueFlat")      Output flat field root name
    (combine = "median")        Type of combine operation
     (reject = "avsigclip")     Type of rejection
    (ccdtype = "")              CCD image type to combine
```

```
(process = no)              Process images before combining?
(subsets = no)              Combine images by subset parameter?
 (delete = no)              Delete input images after combining?
(clobber = no)              Clobber existing output image?
  (scale = "mode")          Image scaling
(statsec = "")              Image section for computing statistics
   (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 = 3.0)             Lower sigma clipping factor
 (hsigma = 3.0)             Upper sigma clipping factor
(rdnoise = "0.")            ccdclip: CCD readout noise (electrons)
   (gain = "1.")            ccdclip: CCD gain (electrons/DN)
 (snoise = "0.")            ccdclip: Sensitivity noise (fraction)
  (pclip = -0.5)            pclip: Percentile clipping parameter
  (blank = 1.0)             Value if there are no pixels
   (mode = "ql")
```

7. Create a combined flatfield. Now we do some image arithmetic to create a super master flat field. This super flat field combines the blue and red flats. The output file is called "superFlat".

At the IRAF prompt, do the following:

```
imarith blueFlat + redFlat junk

imarith junk / 2.0 superFlat

imdel junk
```

Please visually inspect the superFlat to make sure it looks okay. Figure 1 shows an example of the red and blue flat and a combined master "super flat".

8. ARCES SPECIFIC STEP: Magnify the combined flat field. Please refer to the Thorburn data reduction document regarding the magnification that needs to be done on the ARCES data. The magnification process is specific to ARCES data only.

We will first magnify the super flat field we just created. To do this, we will use the IRAF task MAGNIFY, which is located within IMAGES and IMGEOM. In this example, the output file is called superFlatmag.

```
epar magnify
```

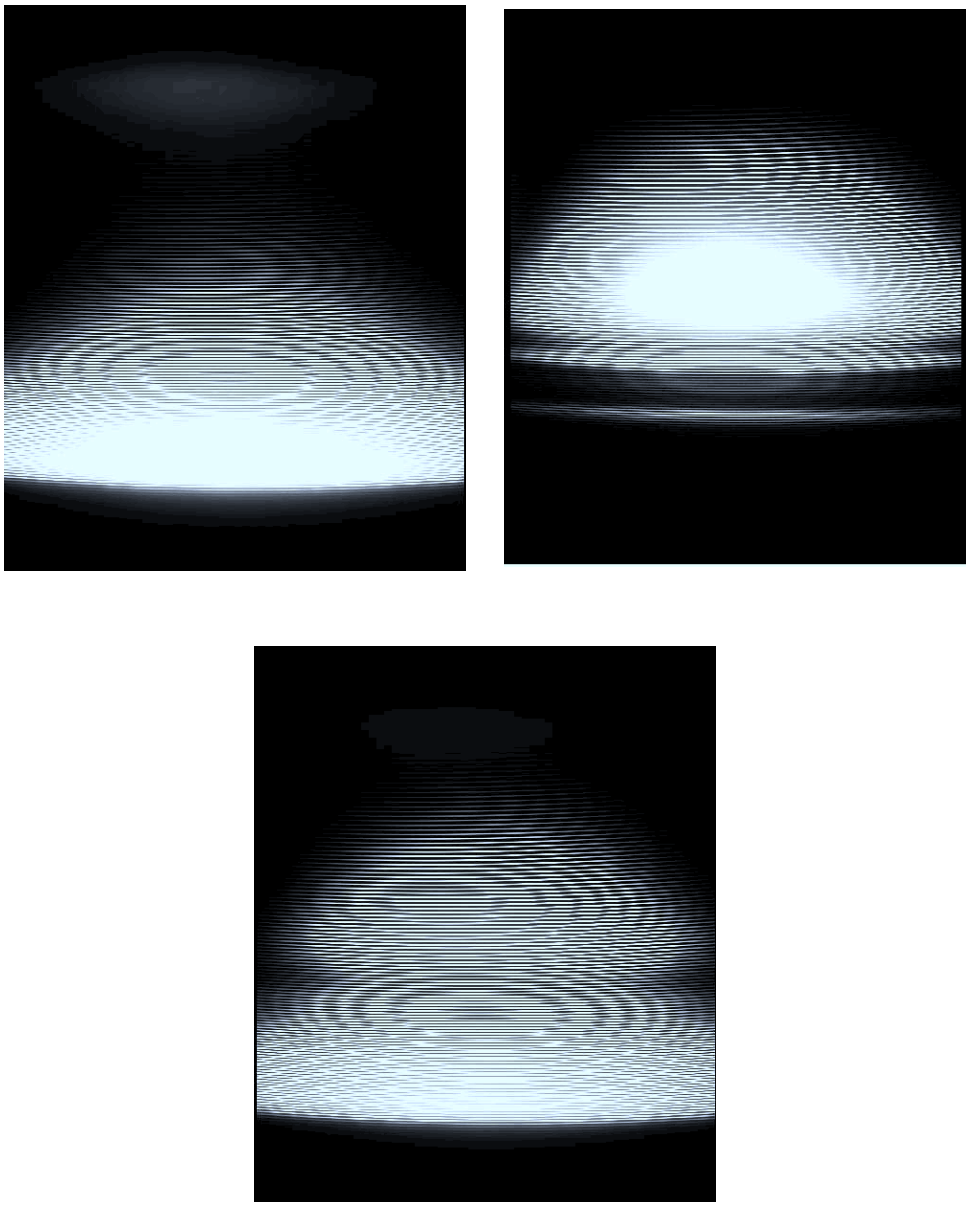


Figure 1: Upper left: Red master flat. Note the reflected light in this flat field frame. Upper right: Blue Master flat. Lower: Combined master flat (blue + red flat field)

```
        input = "superFlat"     Input two dimensional images
       output = "superFlatmag"  Output magnified images
         xmag = 1.0             X magnification factor
         ymag = 4.0             Y magnification factor
          (x1 = INDEF)          X window origin relative to input image
          (x2 = INDEF)          X window end point relative to input image
          (dx = INDEF)          X Pixel interval relative to input image
          (y1 = INDEF)          Y window origin relative to input image
          (y2 = INDEF)          Y window end point relative to input image
          (dy = INDEF)          Y Pixel interval relative to input image
(interpolatio = "linear")       Interpolation type(nearest,linear,poly3,
                                    poly5,spline3,sinc,lsinc,drizzle
    (boundary = "nearest")      Boundary extension type (constant,nearest,
                                    reflect,wrap)
    (constant = 0.0)            Boundary extension constant
(fluxconserve = yes)            Preserve total image flux?
     (logfile = "STDOUT")       Log file
        (mode = "ql")
```

In your IRAF window, you will see output that looks something like this:

```
superFlatmag
  Magnify image superFlat to image superFlatmag.
  Interpolation is linear.
  Boundary extension is nearest.
  Output coordinates in terms of input coordinates:
    x1 =         1., x2 =      1651., dx =         1.
    y1 =         1., y2 =      2048., dy =       0.25
```

You can check the dimension of your file to see that it has indeed increased in size by using IMHEAD:

```
imhead superFlatmag
      superFlatmag[1651,8189][real]:
```

9. ARCES SPECIFIC STEP: Modify CCDSEC headers. To modify the FITS header to show this magnification, you must modify the header with HEDIT.

```
epar hedit
       images = "superFlatmag"  images to be edited
       fields = "CCDSEC"        fields to be edited
```

```
       value = "[200:1850,1:8189]" value expression
        (add = yes)              add rather than edit fields
    (addonly = yes)              add only if field does not exist
     (delete = no)              delete rather than edit fields
     (verify = no)              verify each edit operation
       (show = yes)             print record of each edit operation
     (update = yes)             enable updating of the image header
       (mode = "ql")
```

10. Run APALL on the master flat field. To find the apertures (center, trace, and extract), we use the magnified flat field. We use the task APALL in the TWODSPEC or ECHELLE package. If you are using this cookbook and want to save some time, we have determined an optimized aperture trace file for the ARCES data. Of course, you may rigorously determine this file yourself if our file does not work for your science.

Instead of letting APALL create the "database" directory in your working directory, you must make it yourself. Download the file *apechtrace130522* from the wiki page and place in your newly created database directory. This trace file was determined by J. McKeever of NMSU.

The extracted spectrum is called superFlatmag.ec.fits.

```
epar apall
        input = "superFlatmag"  List of input images
        nfind =                 Number of apertures to be found automatically
      (output = "")             List of output spectra
   (apertures = "")             Apertures
      (format = "echelle")      Extracted spectra format
  (references = "echtrace130522") List of aperture reference images
    (profiles = "")             List of aperture profile images
 (interactive = no)             Run task interactively?
        (find = no)             Find apertures?
    (recenter = yes)            Recenter apertures?
      (resize = yes)            Resize apertures?
        (edit = no)             Edit apertures?
       (trace = yes)            Trace apertures?
    (fittrace = no)             Fit the traced points interactively?
     (extract = yes)            Extract spectra?
      (extras = no)             Extract sky, sigma, etc.?
      (review = no)             Review extractions?
        (line = INDEF)          Dispersion line
        (nsum = 10)             Number of dispersion lines to sum or median

                                # DEFAULT APERTURE PARAMETERS
```

```
       (lower = -14.0)          Lower aperture limit relative to center
       (upper = 14.0)           Upper aperture limit relative to center
   (apidtable = "")             Aperture ID table (optional)

                                # DEFAULT BACKGROUND PARAMETERS

  (b_function = "chebyshev")    Background function
     (b_order = 2)              Background function order
    (b_sample = "-22:-15,15:22") Background sample regions
  (b_naverage = -3)             Background average or median
  (b_niterate = 3)              Background rejection iterations
(b_low_reject = 3.0)            Background lower rejection sigma
(b_high_rejec = 3.0)            Background upper rejection sigma
      (b_grow = 0.0)            Background rejection growing radius

                                # APERTURE CENTERING PARAMETERS

       (width = 18.0)           Profile centering width
      (radius = 18.0)           Profile centering radius
   (threshold = 0.0)            Detection threshold for profile centering

                                # AUTOMATIC FINDING AND ORDERING PARAMETERS

      (minsep = 5.0)            Minimum separation between spectra
      (maxsep = 100000.0)       Maximum separation between spectra
       (order = "increasing")   Order of apertures

                                # RECENTERING PARAMETERS

  (aprecenter = "")             Apertures for recentering calculation
      (npeaks = INDEF)          Select brightest peaks
       (shift = no)             Use average shift instead of recentering?

                                # RESIZING PARAMETERS

      (llimit = INDEF)          Lower aperture limit relative to center
      (ulimit = INDEF)          Upper aperture limit relative to center
      (ylevel = 0.05)           Fraction of peak or intensity for automatic width
        (peak = yes)            Is ylevel a fraction of the peak?
         (bkg = yes)            Subtract background in automatic width?
      (r_grow = 0.0)            Grow limits by this factor
   (avglimits = no)             Average limits over all apertures?
```

```
                                # TRACING PARAMETERS

      (t_nsum = 10)             Number of dispersion lines to sum
      (t_step = 10)             Tracing step
     (t_nlost = 3)              Number of consecutive times profile
                                    is lost before quitting
  (t_function = "legendre")     Trace fitting function
     (t_order = 10)             Trace fitting function order
    (t_sample = "1:1651")       Trace sample regions
  (t_naverage = 1)              Trace average or median
  (t_niterate = 0)              Trace rejection iterations
(t_low_reject = 3.0)            Trace lower rejection sigma
(t_high_rejec = 3.0)            Trace upper rejection sigma
      (t_grow = 0.0)            Trace rejection growing radius

                                # EXTRACTION PARAMETERS

  (background = "fit")          Background to subtract
      (skybox = 1)              Box car smoothing length for sky
    (weights = "variance")      Extraction weights (none|variance)
       (pfit = "fit1d")         Profile fitting type (fit1d|fit2d)
      (clean = yes)             Detect and replace bad pixels?
  (saturation = 35000.0)        Saturation level
   (readnoise = "RDNOISE")      Read out noise sigma (photons)
       (gain = "GAIN")          Photon gain (photons/data number)
     (lsigma = 4.0)             Lower rejection threshold
     (usigma = 4.0)             Upper rejection threshold
    (nsubaps = 1)               Number of subapertures per aperture
       (mode = "ql")
```

11. Normalize your extracted flat field. Next we normalize the extracted spectrum of the master flat field with SFIT. This IRAF task is located in the ONEDSPEC package.

```
lpar sfit
        input = "superFlatmag.ec" Input images
       output = "normFlat.ec"   Output images
          ask = "yes"
       (lines = "*")            Image lines to be fit
       (bands = "1")            Image bands to be fit
        (type = "ratio")        Type of output
     (replace = no)             Replace rejected points by fit?
   (wavescale = no)             Scale the X axis with wavelength?
```

```
    (logscale = no)             Take the log (base 10) of both axes?
    (override = yes)            Override previously fit lines?
    (listonly = no)             List fit but don't modify any images?
    (logfiles = "logfile")      List of log files
 (interactive = no)             Set fitting parameters interactively?
      (sample = "*")            Sample points to use in fit
    (naverage = -5)             Number of points in sample averaging
    (function = "spline3")      Fitting function
       (order = 9)              Order of fitting function
  (low_reject = 3.5)            Low rejection in sigma of fit
 (high_reject = 3.5)            High rejection in sigma of fit
    (niterate = 10)             Number of rejection iterations
        (grow = 1.0)            Rejection growing radius in pixels
     (markrej = yes)            Mark rejected points?
    (graphics = "stdgraph")     Graphics output device
      (cursor = "")             Graphics cursor input
        (mode = "ql")
```

The following step is optional. If you want to view your normalized, extracted 1-D spectrum of the master flat field, you can adjust the y-axis range. To do this, you will use the IRAF task IMREPLACE.

For the upper values:

```
 epar imreplace
       images = "normFlat.ec"   Images to be edited
        value = 2.0             Replacement pixel value
   (imaginary = 0.0)            Imaginary component for complex
       (lower = 2.0)            Lower limit of replacement window
       (upper = INDEF)          Upper limit of replacement window
      (radius = 0.0)            Replacement radius
        (mode = "ql")
```

For the lower range (and to avoid any extreme negative values):

```
epar imreplace
       images = "normFlat.ec"   Images to be edited
        value = 0.5             Replacement pixel value
   (imaginary = 0.0)            Imaginary component for complex
       (lower = INDEF)          Lower limit of replacement window
       (upper = 0.5)            Upper limit of replacement window
      (radius = 0.0)            Replacement radius
        (mode = "ql")
```

12. ARCES SPECIFIC STEP: Magnify your arc calibration files. Now we magnify the arc calibration images using the same procedure as we used for the master flat field.

For the arcs, we create the input @-file containing Parc*.fits files and the output @-file has file names with the prefix "M" (Marc*.fits).

```
epar magnify
        input = "@inmagarc"     Input two dimensional images
       output = "@outmagarc"    Output magnified images
         xmag = 1.0             X magnification factor
         ymag = 4.0             Y magnification factor
          (x1 = INDEF)          X window origin relative to input image
          (x2 = INDEF)          X window end point relative to input image
          (dx = INDEF)          X Pixel interval relative to input image
          (y1 = INDEF)          Y window origin relative to input image
          (y2 = INDEF)          Y window end point relative to input image
          (dy = INDEF)          Y Pixel interval relative to input image
(interpolatio = "linear")       Interpolation type(nearest,linear,poly3,
                                    poly5,spline3,sinc,lsinc,drizzle
    (boundary = "nearest")      Boundary extension type (constant,nearest,
                                    reflect,wrap)
    (constant = 0.0)            Boundary extension constant
(fluxconserve = yes)            Preserve total image flux?
     (logfile = "STDOUT")       Log file
        (mode = "ql")
```

Example output from IRAF:

```
Marc.0001.fits
  Magnify image Parc.0001.fits to image Marc.0001.fits.
  Interpolation is linear.
  Boundary extension is nearest.
  Output coordinates in terms of input coordinates:
    x1 =         1., x2 =      1651., dx =         1.
    y1 =         1., y2 =      2048., dy =      0.25

Marc.0002.fits
  Magnify image Parc.0002.fits to image Marc.0002.fits.
  Interpolation is linear.
  Boundary extension is nearest.
  Output coordinates in terms of input coordinates:
    x1 =         1., x2 =      1651., dx =         1.
    y1 =         1., y2 =      2048., dy =      0.25
```

As with the master flat field, we must modify the CCDSEC header parameter in all the magnified arc images. Use HEDIT to do this.

```
epar hedit
       images = "@outmagarc"    images to be edited
       fields = "CCDSEC"        fields to be edited
        value = "[200:1850,1:8189]" value expression
         (add = yes)             add rather than edit fields
     (addonly = yes)             add only if field does not exist
      (delete = no)             delete rather than edit fields
      (verify = no)             verify each edit operation
        (show = yes)            print record of each edit operation
      (update = yes)            enable updating of the image header
        (mode = "ql")
```

13. Extract your arc calibration spectra with APALL. Extract the spectra from the magnified arc images with APALL. Use your master flat field frame (unextracted) as your reference image. The extracted spectra of the arc images will have the suffix *.ec.fits.

```
epar apall
        input = "@outmagarc"    List of input images
        nfind =                 Number of apertures to be found automatically
      (output = "")             List of output spectra
   (apertures = "")             Apertures
      (format = "echelle")      Extracted spectra format
  (references = "superFlatmag") List of aperture reference images
    (profiles = "")             List of aperture profile images
 (interactive = no)             Run task interactively?
        (find = no)             Find apertures?
    (recenter = no)             Recenter apertures?
      (resize = no)             Resize apertures?
        (edit = no)             Edit apertures?
       (trace = yes)            Trace apertures?
    (fittrace = no)             Fit the traced points interactively?
     (extract = yes)            Extract spectra?
      (extras = no)             Extract sky, sigma, etc.?
      (review = no)             Review extractions?
        (line = INDEF)          Dispersion line
        (nsum = 10)             Number of dispersion lines to sum or median

                                # DEFAULT APERTURE PARAMETERS

       (lower = -14.0)          Lower aperture limit relative to center
```

```
      (upper = 14.0)           Upper aperture limit relative to center
   (apidtable = "")            Aperture ID table (optional)

                               # DEFAULT BACKGROUND PARAMETERS

  (b_function = "chebyshev")   Background function
     (b_order = 2)             Background function order
    (b_sample = "-22:-15,15:22") Background sample regions
  (b_naverage = -3)            Background average or median
  (b_niterate = 3)             Background rejection iterations
(b_low_reject = 3.0)           Background lower rejection sigma
(b_high_rejec = 3.0)           Background upper rejection sigma
      (b_grow = 0.0)           Background rejection growing radius

                               # APERTURE CENTERING PARAMETERS

       (width = 18.0)          Profile centering width
      (radius = 18.0)          Profile centering radius
   (threshold = 0.0)           Detection threshold for profile centering

                               # AUTOMATIC FINDING AND ORDERING PARAMETERS

      (minsep = 5.0)           Minimum separation between spectra
      (maxsep = 100000.0)      Maximum separation between spectra
       (order = "increasing")  Order of apertures

                               # RECENTERING PARAMETERS

  (aprecenter = "")            Apertures for recentering calculation
      (npeaks = INDEF)         Select brightest peaks
       (shift = no)            Use average shift instead of recentering?

                               # RESIZING PARAMETERS

      (llimit = INDEF)         Lower aperture limit relative to center
      (ulimit = INDEF)         Upper aperture limit relative to center
      (ylevel = 0.05)          Fraction of peak or intensity for automatic width
        (peak = yes)           Is ylevel a fraction of the peak?
         (bkg = yes)           Subtract background in automatic width?
      (r_grow = 0.0)           Grow limits by this factor
   (avglimits = no)            Average limits over all apertures?

                               # TRACING PARAMETERS
```

```
      (t_nsum = 10)             Number of dispersion lines to sum
      (t_step = 10)             Tracing step
     (t_nlost = 3)              Number of consecutive times profile
                                    is lost before quitting
  (t_function = "legendre")     Trace fitting function
     (t_order = 10)             Trace fitting function order
    (t_sample = "1:1651")       Trace sample regions
  (t_naverage = 1)              Trace average or median
  (t_niterate = 0)              Trace rejection iterations
(t_low_reject = 3.0)            Trace lower rejection sigma
(t_high_rejec = 3.0)            Trace upper rejection sigma
      (t_grow = 0.0)            Trace rejection growing radius

                                # EXTRACTION PARAMETERS

  (background = "none")         Background to subtract
      (skybox = 1)              Box car smoothing length for sky
     (weights = "none")         Extraction weights (none|variance)
        (pfit = "fit1d")        Profile fitting type (fit1d|fit2d)
       (clean = yes)            Detect and replace bad pixels?
  (saturation = 35000.0)        Saturation level
   (readnoise = "RDNOISE")      Read out noise sigma (photons)
        (gain = "GAIN")         Photon gain (photons/data number)
      (lsigma = 4.0)            Lower rejection threshold
      (usigma = 4.0)            Upper rejection threshold
     (nsubaps = 1)              Number of subapertures per aperture
        (mode = "ql")
```

14. ARCES SPECIFIC STEP: Magnify the object spectra. Now we magnify the object images as we have done with the master flat field and arc images. The input files are the CR2obj*.fits, in this example, and the output files are called Mobj*.fits. Again, we utilize the @-file for the input and output.

```
epar magnify
        input = "@inmagobj"     Input two dimensional images
       output = "@outmagobj"    Output magnified images
         xmag = 1.0             X magnification factor
         ymag = 4.0             Y magnification factor
          (x1 = INDEF)          X window origin relative to input image
          (x2 = INDEF)          X window end point relative to input image
          (dx = INDEF)          X Pixel interval relative to input image
          (y1 = INDEF)          Y window origin relative to input image
```

```
          (y2 = INDEF)          Y window end point relative to input image
          (dy = INDEF)          Y Pixel interval relative to input image
(interpolatio = "linear")       Interpolation type(nearest,linear,poly3,
                                    poly5,spline3,sinc,lsinc,drizzle
    (boundary = "nearest")      Boundary extension type (constant,nearest,
                                    reflect,wrap)
    (constant = 0.0)            Boundary extension constant
(fluxconserve = yes)            Preserve total image flux?
     (logfile = "STDOUT")       Log file
        (mode = "ql")
```

Example IRAF output:

```
Mobj.0001.fits
  Magnify image CR2obj.0001.fits to image Mobj.0001.fits.
  Interpolation is linear.
  Boundary extension is nearest.
  Output coordinates in terms of input coordinates:
    x1 =         1., x2 =      1651., dx =         1.
    y1 =         1., y2 =      2048., dy =      0.25
```

And as before, we must modify the CCDSEC header parameter with HEDIT.

```
lpar hedit
       images = "@outmagobj"    images to be edited
       fields = "CCDSEC"        fields to be edited
        value = "[200:1850,1:8189]" value expression
         (add = no)             add rather than edit fields
     (addonly = no)             add only if field does not exist
      (delete = no)             delete rather than edit fields
      (verify = no)             verify each edit operation
        (show = yes)            print record of each edit operation
      (update = yes)            enable updating of the image header
        (mode = "ql")
```

15. Create some dummy files in /database with APALL. To extract the 1-D spectra of the object images, we do this in two steps. First we create empty entries in the database directory of your working data directory. We can do this with APALL but turn off all the subtasks for extraction. We want to correct for scatter light separately, so we are essentially prepping the object files to do this outside of APALL.

```
epar apall
```

```
        input = "@outmagobj"    List of input images
        nfind =                 Number of apertures to be found automatically
      (output = "")             List of output spectra
   (apertures = "")             Apertures
      (format = "echelle")      Extracted spectra format
  (references = "superFlatmag") List of aperture reference images
    (profiles = "")             List of aperture profile images
 (interactive = no)             Run task interactively?
       (find = no)              Find apertures?
    (recenter = no)             Recenter apertures?
      (resize = no)             Resize apertures?
        (edit = no)             Edit apertures?
       (trace = no)             Trace apertures?
    (fittrace = no)             Fit the traced points interactively?
     (extract = no)             Extract spectra?
      (extras = no)             Extract sky, sigma, etc.?
      (review = no)             Review extractions?
        (line = INDEF)          Dispersion line
        (nsum = 10)             Number of dispersion lines to sum or median

                                # DEFAULT APERTURE PARAMETERS

       (lower = -14.0)          Lower aperture limit relative to center
       (upper = 14.0)           Upper aperture limit relative to center
   (apidtable = "")             Aperture ID table (optional)

                                # DEFAULT BACKGROUND PARAMETERS

  (b_function = "chebyshev")    Background function
     (b_order = 2)              Background function order
    (b_sample = "-22:-15,15:22") Background sample regions
  (b_naverage = -3)             Background average or median
  (b_niterate = 3)              Background rejection iterations
(b_low_reject = 3.0)            Background lower rejection sigma
(b_high_rejec = 3.0)            Background upper rejection sigma
      (b_grow = 0.0)            Background rejection growing radius

                                # APERTURE CENTERING PARAMETERS

       (width = 18.0)           Profile centering width
      (radius = 18.0)           Profile centering radius
   (threshold = 0.0)            Detection threshold for profile centering
```

```
                                # AUTOMATIC FINDING AND ORDERING PARAMETERS

      (minsep = 5.0)            Minimum separation between spectra
      (maxsep = 100000.0)       Maximum separation between spectra
       (order = "increasing")   Order of apertures

                                # RECENTERING PARAMETERS

  (aprecenter = "")             Apertures for recentering calculation
      (npeaks = INDEF)          Select brightest peaks
       (shift = no)             Use average shift instead of recentering?

                                # RESIZING PARAMETERS

      (llimit = INDEF)          Lower aperture limit relative to center
      (ulimit = INDEF)          Upper aperture limit relative to center
      (ylevel = 0.05)           Fraction of peak or intensity for automatic width
        (peak = yes)            Is ylevel a fraction of the peak?
         (bkg = yes)            Subtract background in automatic width?
      (r_grow = 0.0)            Grow limits by this factor
   (avglimits = no)             Average limits over all apertures?

                                # TRACING PARAMETERS

      (t_nsum = 10)             Number of dispersion lines to sum
      (t_step = 10)             Tracing step
     (t_nlost = 3)              Number of consecutive times profile
                                    is lost before quitting
  (t_function = "legendre")     Trace fitting function
     (t_order = 10)             Trace fitting function order
    (t_sample = "1:1651")       Trace sample regions
  (t_naverage = 1)              Trace average or median
  (t_niterate = 0)              Trace rejection iterations
(t_low_reject = 3.0)            Trace lower rejection sigma
(t_high_rejec = 3.0)            Trace upper rejection sigma
      (t_grow = 0.0)            Trace rejection growing radius

                                # EXTRACTION PARAMETERS

  (background = "fit")          Background to subtract
      (skybox = 1)              Box car smoothing length for sky
    (weights = "variance")      Extraction weights (none|variance)
       (pfit = "fit1d")         Profile fitting type (fit1d|fit2d)
```

```
      (clean = yes)           Detect and replace bad pixels?
 (saturation = 35000.0)       Saturation level
  (readnoise = "RDNOISE")     Read out noise sigma (photons)
       (gain = "GAIN")        Photon gain (photons/data number)
     (lsigma = 4.0)           Lower rejection threshold
     (usigma = 4.0)           Upper rejection threshold
    (nsubaps = 1)             Number of subapertures per aperture
       (mode = "ql")
```

16. Scatter light correction. To remove the scatter light, we use APSCATTER. First, we rename the data files. At the IRAF prompt, do the following:

```
imcopy Mobj*.fits  noscat*.fits  # or just use @-files here.

files Mobj*.fits > outscat

files noscat*.fits > inscat

imdel Mobj*.fits
```

Don't worry, we will recreate the Mobj*.fits again.

Now we run APSCATTER. You may want to set APSCAT1 and APSCAT2 parameters before running APSCATTER. You can set these subtasks within APSCATTER. The header parameters "order", "low_reject" and " high_reject" values presented here were determined by Doug Gies of Georgia State University.

```
lpar apscat1
    (function = "spline3")      Fitting function
       (order = 25)             Order of fitting function
      (sample = "*")            Sample points to use in fit
    (naverage = 1)              Number of points in sample averaging
  (low_reject = 6.0)            Low rejection in sigma of fit
 (high_reject = 1.5)            High rejection in sigma of fit
    (niterate = 5)              Number of rejection iterations
        (grow = 1.0)            Rejection growing radius in pixels
        (mode = "ql")

 lpar apscat2
    (function = "spline3")      Fitting function
       (order = 5)              Order of fitting function
      (sample = "*")            Sample points to use in fit
```

```
    (naverage = 1)              Number of points in sample averaging
  (low_reject = 3.0)            Low rejection in sigma of fit
 (high_reject = 3.0)            High rejection in sigma of fit
    (niterate = 2)              Number of rejection iterations
        (grow = 0.0)            Rejection growing radius in pixels
        (mode = "ql")

epar apscatter
        input = "@inscat"       List of input images to subtract scattered light
       output = "@outscat"      List of output corrected images
   (apertures = "")             Apertures
     (scatter = "")             List of scattered light images (optional)
  (references = "@outscat")     List of aperture reference images
 (interactive = no)             Run task interactively?
        (find = no)             Find apertures?
    (recenter = no)             Recenter apertures?
      (resize = no)             Resize apertures?
        (edit = no)             Edit apertures?
       (trace = no)             Trace apertures?
    (fittrace = no)             Fit the traced points interactively?
    (subtract = yes)            Subtract scattered light?
      (smooth = yes)            Smooth scattered light along the dispersion?
  (fitscatter = yes)            Fit scattered light interactively?
   (fitsmooth = yes)            Smooth the scattered light interactively?
        (line = INDEF)          Dispersion line
        (nsum = -10)            Number of dispersion lines to sum or median
      (buffer = 0.4)            Buffer distance from apertures
     (apscat1 = "")             Fitting parameters across the dispersion
     (apscat2 = "")             Fitting parameters along the dispersion
        (mode = "ql")
```

17. Extract object spectra with APALL. After correcting for the scattered light, we can finish the process of extracting the 1-D spectra of your object spectra. Again we use APALL. The input file contains all the Mobj*.fits files and the output file contains files with the extra extension of *.ec.fits (i.e. Mobj*.ec.fits).

```
epar apall
        input = "@inexobj"      List of input images
        nfind =                 Number of apertures to be found automatically
      (output = "")             List of output spectra
   (apertures = "")             Apertures
      (format = "echelle")      Extracted spectra format
  (references = "")             List of aperture reference images
```

```
   (profiles = "")            List of aperture profile images
(interactive = no)            Run task interactively?
       (find = no)            Find apertures?
   (recenter = no)            Recenter apertures?
     (resize = no)            Resize apertures?
       (edit = no)            Edit apertures?
      (trace = no)            Trace apertures?
   (fittrace = no)            Fit the traced points interactively?
    (extract = yes)           Extract spectra?
     (extras = no)            Extract sky, sigma, etc.?
     (review = no)            Review extractions?
       (line = INDEF)         Dispersion line
       (nsum = 10)            Number of dispersion lines to sum or median

                              # DEFAULT APERTURE PARAMETERS

      (lower = -14.0)         Lower aperture limit relative to center
      (upper = 14.0)          Upper aperture limit relative to center
  (apidtable = "")            Aperture ID table (optional)

                              # DEFAULT BACKGROUND PARAMETERS

 (b_function = "chebyshev")   Background function
    (b_order = 2)             Background function order
   (b_sample = "-22:-15,15:22") Background sample regions
 (b_naverage = -3)            Background average or median
 (b_niterate = 3)             Background rejection iterations
(b_low_reject = 3.0)          Background lower rejection sigma
(b_high_rejec = 3.0)          Background upper rejection sigma
     (b_grow = 0.0)           Background rejection growing radius

                              # APERTURE CENTERING PARAMETERS

      (width = 18.0)          Profile centering width
     (radius = 18.0)          Profile centering radius
  (threshold = 0.0)           Detection threshold for profile centering

                              # AUTOMATIC FINDING AND ORDERING PARAMETERS

     (minsep = 5.0)           Minimum separation between spectra
     (maxsep = 100000.0)      Maximum separation between spectra
      (order = "increasing")  Order of apertures
```

```
                            # RECENTERING PARAMETERS

  (aprecenter = "")         Apertures for recentering calculation
      (npeaks = INDEF)      Select brightest peaks
       (shift = no)         Use average shift instead of recentering?

                            # RESIZING PARAMETERS

      (llimit = INDEF)      Lower aperture limit relative to center
      (ulimit = INDEF)      Upper aperture limit relative to center
      (ylevel = 0.05)       Fraction of peak or intensity for automatic width
        (peak = yes)        Is ylevel a fraction of the peak?
         (bkg = yes)        Subtract background in automatic width?
      (r_grow = 0.0)        Grow limits by this factor
   (avglimits = no)         Average limits over all apertures?

                            # TRACING PARAMETERS

      (t_nsum = 10)         Number of dispersion lines to sum
      (t_step = 10)         Tracing step
     (t_nlost = 3)          Number of consecutive times profile
                                is lost before quitting
  (t_function = "legendre") Trace fitting function
     (t_order = 10)         Trace fitting function order
    (t_sample = "1:1651")   Trace sample regions
  (t_naverage = 1)          Trace average or median
  (t_niterate = 0)          Trace rejection iterations
(t_low_reject = 3.0)        Trace lower rejection sigma
(t_high_rejec = 3.0)        Trace upper rejection sigma
      (t_grow = 0.0)        Trace rejection growing radius

                            # EXTRACTION PARAMETERS

  (background = "fit")      Background to subtract
      (skybox = 1)          Box car smoothing length for sky
    (weights = "variance")  Extraction weights (none|variance)
       (pfit = "fit1d")     Profile fitting type (fit1d|fit2d)
      (clean = yes)         Detect and replace bad pixels?
  (saturation = 35000.0)    Saturation level
   (readnoise = "RDNOISE")  Read out noise sigma (photons)
       (gain = "GAIN")      Photon gain (photons/data number)
      (lsigma = 4.0)        Lower rejection threshold
      (usigma = 4.0)        Upper rejection threshold
```

```
    (nsubaps = 1)              Number of subapertures per aperture
       (mode = "ql")
```

18. Flat field your object spectra. Flat field correct the 1-D object spectra with the 1-D, normalized master flat field. Use IMARITH to do this image arithmetic.

```
imarith @inflatten / normFlat.ec @outflatten
```

(ex: Mobj*.ec.fits in @inflatten and FMobj*.fits in @outflatten)

19. Reidentify arc lines in your arcs. To identify all the lines in the extracted arc spectra, we use ECREIDENTIFY. J. McKeever has produced a reference file which allows us to skip the ECIDENTIFY task. You may, of course, create your own arc line identification file.

To obtain the reference spectrum file, download *arcnewref.ec* from the APO wiki page.

Create an @-file for all the extracted arc files:

```
files Marc.*.ec.fits > inreident
```

And now run ECREIDENTIFY:

```
epar ecreidentify
       images = "@inreident"    Spectra to be reidentified
    reference = "arcnewref.ec"  Reference spectrum
       (shift = INDEF)          Shift to add to reference features
     (cradius = 2.0)            Centering radius
   (threshold = 50.0)           Feature threshold for centering
       (refit = yes)            Refit coordinate function?
    (database = "database")     Database
    (logfiles = "STDOUT,logfile") List of log files
        (mode = "ql")
```

Here is example output from ECREIDENTIFY:

```
ECREIDENTIFY: NOAO/IRAF V2.14EXPORT
  Reference image = arcnewref.ec, Refit = yes
       Image    Found     Fit Pix Shift  User Shift  Z Shift      RMS
Marc.0001.ec  1214/1370 1214/1214     0.432       -3.19  -5.6E-6   0.0147
```

```
Marc.0002.ec  1242/1370 1242/1242     0.269         -2.  -3.5E-6    0.014
Marc.0003.ec  1242/1370 1242/1242     0.225       -1.67  -3.0E-6   0.0161
Marc.0004.ec  1253/1370 1253/1253   -0.0131      0.0722  1.20E-7   0.0173
Marc.0005.ec  1225/1370 1225/1225   -0.0392       0.265  4.60E-7   0.0104
Marc.0006.ec  1250/1370 1250/1250    -0.159        1.14  2.00E-6   0.0153
```

20. Assign corresponding arcs to your object spectra. Run REFSPEC to assign the reference arc spectra to your object spectra. There are different ways to run this task. Here is an alternate way of running REFSPEC by providing a list of reference spectra. The list will explicitly assign the spectra rather than letting IRAF determine which arc spectra is appropriate based on time indices in the FITS headers. ARCES data do not have have the UTMIDDLE header parameter. You can also calculate UTMIDDLE from a variety of IRAF tasks (see step 22 of this manual).

The input file in this example, "@inwave" contains files with the extracted spectra with the prefix "FM".

Here is an example of a list of reference spectra. The name of this file (referspec.txt) goes into the "references" input parameter for REFSPEC.

```
FMobj.0001.ec  Marc.0001.ec,Marc.0002.ec
FMobj.0002.ec  Marc.0003.ec,Marc.0004.ec
FMobj.0003.ec  Marc.0005.ec,Marc.0006.ec
```

```
epar refspec
        input = "@inwave"       List of input spectra
       answer = "YES"           Accept assignment?
  (references = "referspec.txt") List of reference spectra
   (apertures = "")             Input aperture selection list
      (refaps = "")             Reference aperture selection list
   (ignoreaps = yes)            Ignore input and reference apertures?
      (select = "interp")       Selection method for reference spectra
        (sort = "")             Sort key
       (group = "")             Group key
        (time = yes)            Is sort key a time?
    (timewrap = 17.0)           Time wrap point for time sorting
    (override = yes)            Override previous assignments?
     (confirm = no)             Confirm reference spectrum assignments?
      (assign = yes)            Assign the reference spectra to the input spectrum?
    (logfiles = "STDOUT,logfile") List of logfiles
     (verbose = no)             Verbose log output?
        (mode = "al")
```

Here is the output on screen from REFSPEC:

```
[FMobj.0001.ec] refspec1='Marc.0001.ec 0.5'
[FMobj.0001.ec] refspec2='Marc.0002.ec 0.5'
[FMobj.0002.ec] refspec1='Marc.0003.ec 0.5'
[FMobj.0002.ec] refspec2='Marc.0004.ec 0.5'
[FMobj.0003.ec] refspec1='Marc.0005.ec 0.5'
[FMobj.0003.ec] refspec2='Marc.0006.ec 0.5'
```

21. Dispersion correction of your object spectra. Calculate the dispersion correction by running DISPCOR. The input @-file contains all the normalized, extracted spectra (FMobj*.ec.fits) and the output @-file contains files with the prefix "EX" (e.g. EXobj.*.ec.fits).

```
epar dispcor
       input = "@indispcor"    List of input spectra
      output = "@outdispcor"   List of output spectra
  (linearize = no)             Linearize (interpolate) spectra?
   (database = "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 = no)             Conserve total flux?
      (blank = 0.0)            Output value of points not in input
   (samedisp = no)             Same dispersion in all apertures?
     (global = no)             Apply global defaults?
  (ignoreaps = no)             Ignore apertures?
    (confirm = no)             Confirm dispersion coordinates?
   (listonly = no)             List the dispersion coordinates only?
    (verbose = yes)            Print linear dispersion assignments?
    (logfile = "")             Log file
       (mode = "ql")
```

22. Fill your headers with some useful information. If you need it, you can run SETJD to assign the Julian date information into the FITS headers. If you require the UTMIDDLE keyword to be in the headers, you can run SETAIRMASS.

```
epar setjd
       images = "EXobj*.fits"   Images
 (observatory = ")_.observatory") Observatory of observation
       (date = "date-obs")      Date of observation keyword
       (time = "ut")            Time of observation keyword
```

```
    (exposure = "exptime")      Exposure time keyword
          (ra = "ra")           Right ascension (hours) keyword
         (dec = "dec")          Declination (degrees) keyword
       (epoch = "equinox")      Epoch (years) keyword
          (jd = "jd")           Output Julian date keyword
         (hjd = "hjd")          Output Helocentric Julian date keyword
         (ljd = "ljd")          Output local Julian date keyword
      (utdate = yes)            Is observation date UT?
      (uttime = yes)            Is observation time UT?
    (listonly = no)             List only without modifying images?
        (mode = "al")
```

Here is the output to the screen after running SETJD on your extracted object spectra.

```
#               Image              jd            hjd       ljd
# SETJD: Observatory parameters for Apache Point Observatory
#       timezone = 7
  EXobj.0001.ec.fits  2456768.89360  2456768.89344   2456768
  EXobj.0002.ec.fits  2456768.94037  2456768.94087   2456768
  EXobj.0003.ec.fits  2456768.99016  2456768.98938   2456768
```

To obtain the UTMIDDLE parameter, you can run SETAIRMASS. I modified certain parameters that are read in by this IRAF task so that the original values are not overwritten (i.e. "airmass2").

```
epar setairmass
       images = "test.ec"       Input images
 (observatory = ")_.observatory") Observatory for images
     (intype = "beginning")     Input keyword time stamp
    (outtype = "effective")     Output airmass time stamp\n
         (ra = "ra")            Right acsension keyword (hours)
        (dec = "dec")           Declination keyword (degrees)
    (equinox = "equinox")       Equinox keyword (years)
         (st = "LST")           Local siderial time keyword (hours)
         (ut = "date-obs")      Universal time keyword (hours)
       (date = "date-obs")      Observation date keyword
   (exposure = "exptime")       Exposure time keyword (seconds)
    (airmass = "airmass2")      Airmass keyword (output)
   (utmiddle = "utmiddle")      Mid-observation UT keyword (output)
      (scale = 750.0)           The atmospheric scale height\n
       (show = yes)             Print the airmasses and mid-UT?
     (update = yes)             Update the image header?
```

```
    (override = yes)            Override previous assignments?
        (mode = "al")
```

Some output to screen:

```
#              Image    UT middle  effective  begin   middle     end   updated
# SETAIRMASS: Observatory parameters for Apache Point Observatory
#       latitude = 32:46.8
             test.ec    9:26:47.3   1.2267   1.2839   1.2249   1.1765  yes
```

Check your FITS headers to see if the header parameter UTMIDDLE has been created.

23. Continuum flatten your object spectra. *Optional:* There are many robust ways to continuum flatten your spectra. I provide one method of performing this task. A nice write up exists for Subaru spectra, written by Wako Aoki, and his write up is available as a reference on the APO wiki page.

To do my prescription of continuum flattening, I recommend doing this step interactively and change the function/order/niter/low_rej/high_rej parameters for each echelle order as necessary. This step takes a lot of time as you will be inspecting each echelle order and checking/modifying the IRAF determined continuum is good enough.

In this example exercise, I have IMCOPY'd my EXobj*.ec.fits files to a generic filename of "orig" for each extracted spectrum file. You can use the original filename for the input of CONTINUUM. My generic output spectrum name is "cont". If you use my generic file names, the CONTINUUM parameters do not need to be changed each time you run it. However, you will have to remember to delete the orig.fits and cont.fits files before you copy a new file to orig.fits and create a new cont.fits each time.

```
imcopy EXobj.0001.ec.fits orig.fits
```

Adjust the function and its order for each echelle order to find the best continuum fit. As I am going through the echelle orders, I increase the function order, but this depends on your features contaminating the generated fit (e.g. broad Balmer or Ca II lines). If a fitting function of spline3 doesn't work for you, try :func chebyshev and :order 7. Don't try to fit every single wiggle in the spectrum at a high function order, otherwise your flattened spectrum will have unintended wiggles in the continuum.

If you want to run CONTINUUM in batch, just use the usual @-files for both input and output. However, you will need to inspect your spectra carefully to make sure the continuum flattening is acceptable for your analysis.

```
echelle> lpar continuum
        input = "orig"          Input images
       output = "cont"          Output images
          ask = "YES"           Fit [1,1] of orig.fits w/ graph?
       (lines = "*")            Image lines to be fit
       (bands = "*")            Image bands to be fit
        (type = "fit")          Type of output
     (replace = no)             Replace rejected points by fit?
   (wavescale = yes)            Scale the X axis with wavelength?
    (logscale = no)             Take the log (base 10) of both axes?
    (override = no)             Override previously fit lines?
    (listonly = no)             List fit but don't modify any images?
    (logfiles = "logfile")      List of log files
 (interactive = yes)            Set fitting parameters interactively?
      (sample = "*")            Sample points to use in fit
    (naverage = 1)              Number of points in sample averaging
    (function = "spline3")    Fitting function
       (order = 3)              Order of fitting function
  (low_reject = 2.)             Low rejection in sigma of fit
 (high_reject = 5.)             High rejection in sigma of fit
    (niterate = 20)             Number of rejection iterations
        (grow = 1.)             Rejection growing radius
     (markrej = yes)            Mark rejected points?
    (graphics = "stdgraph")     Graphics output device
      (cursor = "")             Graphics cursor input
        (mode = "ql")
```

In Figure 2-4 below, you can see how well or how difficult it is to determine the continuum per order. Wavelength ranges are provided in the captions.

After determining the continuum, run SCOMBINE for both the original and the continuum spectrum you just created.

```
epar scombine
        input = "orig"          List of input spectra
       output = "orig.wav"      List of output spectra
     (noutput = "")             List of output number combined spectra
     (logfile = "STDOUT")       Log file
   (apertures = "")             Apertures to combine
        (group = "images")      Grouping option
     (combine = "sum")          Type of combine operation
      (reject = "none")         Type of rejection
       (first = no)             Use first spectrum for dispersion?
          (w1 = INDEF)          Starting wavelength of output spectra
```

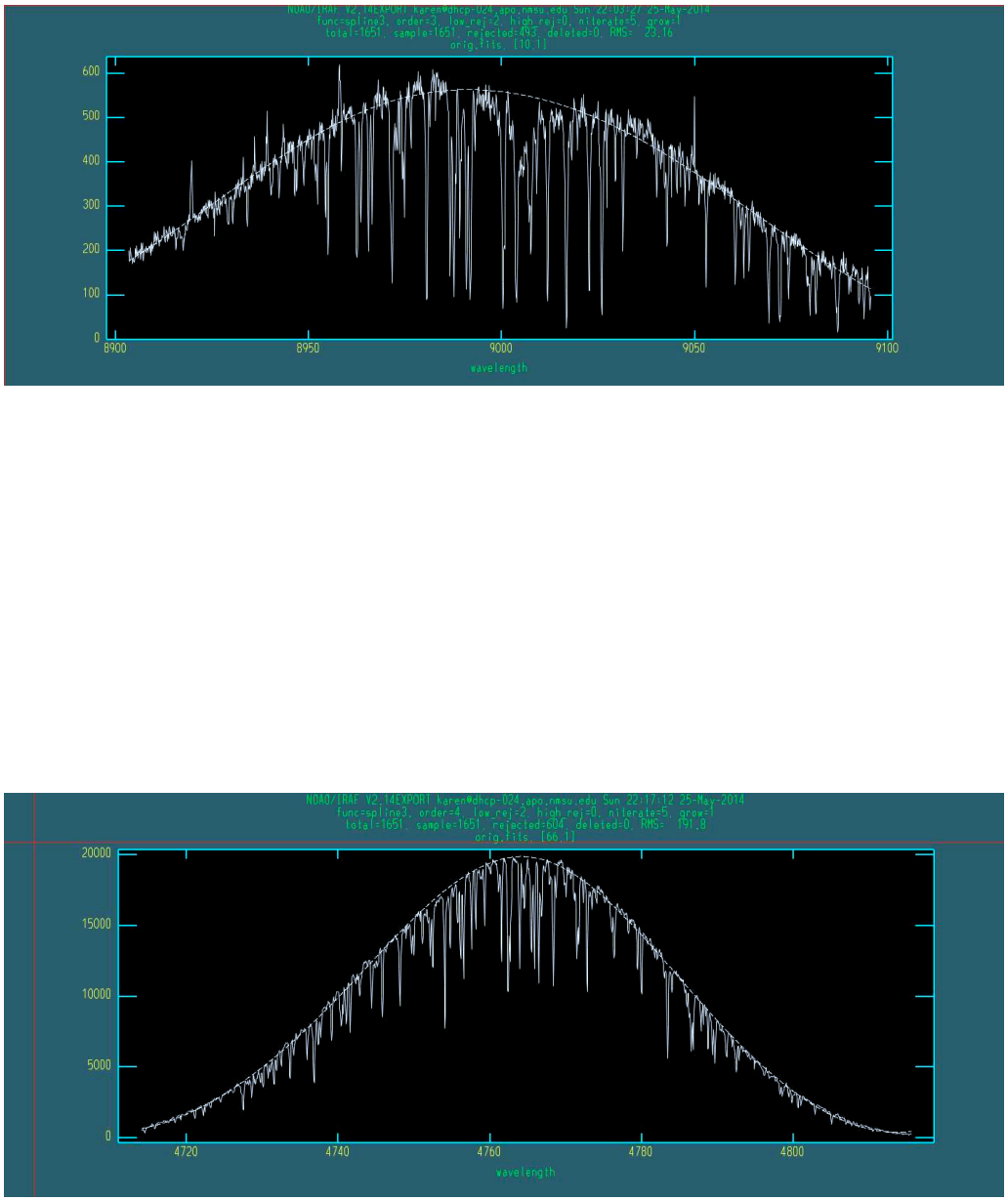
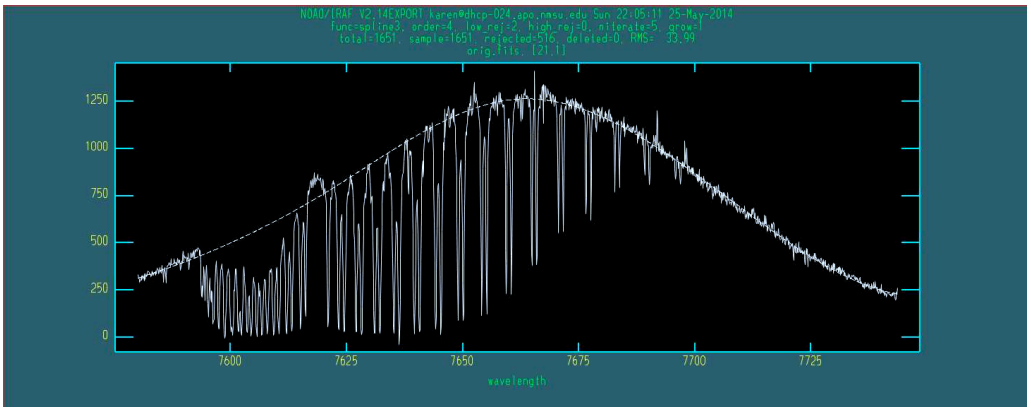


Figure 2: Upper: A red order spanning 8900 to 9100 Å. Note the density of lines can depress the continuum level. Lower: An example of an order with lots of metal lines and a good continuum determination. Wavelength range is between 4715 to 4820Å.



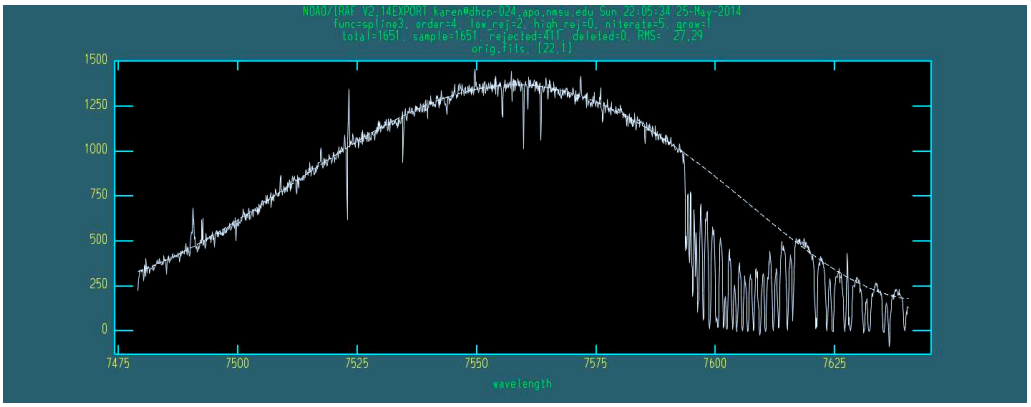


Figure 3: Upper: An order that contains the atmospheric telluric lines. Wavelength range is betwen 7575 to 7750Å. Lower: Another order with atmospheric telluric lines. Wavelength range is betwwen 7475 to 7650Å.

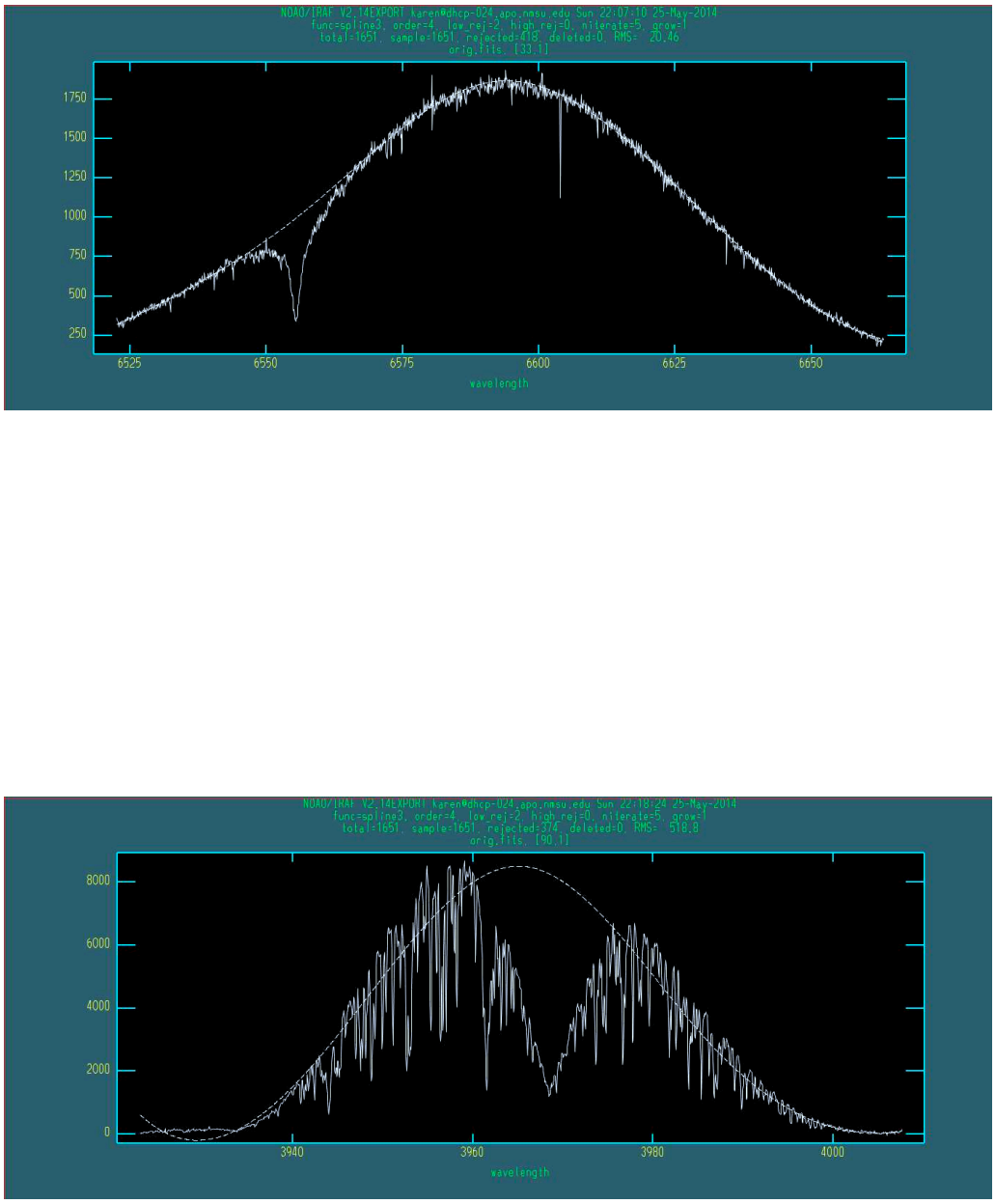


Figure 4: Upper: This order has the broad $H\alpha$ line at 6563Å. Wavelength range is between 6520 to 6675Å. Lower: An example of a broad Ca line (3968Å) that affects the determination of the continuum. Wavelength range is between 3920 to 4010Å.

```
          (w2 = INDEF)          Ending wavelength of output spectra
          (dw = INDEF)          Wavelength increment of output spectra
          (nw = INDEF)          Length of output spectra
         (log = no)             Logarithmic increments?
       (scale = "none")         Image scaling
        (zero = "none")         Image zero point offset
      (weight = "none")         Image weights
      (sample = "")             Wavelength sample regions for statistics
  (lthreshold = INDEF)          Lower threshold
  (hthreshold = 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 = 3.0)            Lower sigma clipping factor
      (hsigma = 3.0)            Upper sigma clipping factor
     (rdnoise = "0.")           ccdclip: CCD readout noise (electrons)
        (gain = "1.")           ccdclip: CCD gain (electrons/DN)
      (snoise = "0.")           ccdclip: Sensitivity noise (fraction)
    (sigscale = 0.1)            Tolerance for sigma clipping scaling corrections
       (pclip = -0.5)           pclip: Percentile clipping parameter
        (grow = 0)              Radius (pixels) for 1D neighbor rejection
       (blank = 0.0)            Value if there are no pixels
        (mode = "ql")


epar scombine
         input = "cont"         List of input spectra
        output = "cont.wav"     List of output spectra
      (noutput = "")            List of output number combined spectra
      (logfile = "STDOUT")      Log file\n
    (apertures = "")            Apertures to combine
       (group = "images")       Grouping option
      (combine = "sum")         Type of combine operation
       (reject = "none")    Type of rejection\n
        (first = no)            Use first spectrum for dispersion?
          (w1 = INDEF)          Starting wavelength of output spectra
          (w2 = INDEF)          Ending wavelength of output spectra
          (dw = INDEF)          Wavelength increment of output spectra
          (nw = INDEF)          Length of output spectra
         (log = no)             Logarithmic increments?\n
       (scale = "none")         Image scaling
        (zero = "none")         Image zero point offset
```

```
     (weight = "none")         Image weights
     (sample = "")             Wavelength sample regions for statistics\n
 (lthreshold = INDEF)          Lower threshold
 (hthreshold = 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)
   (sigscale = 0.1)            Tolerance for sigma clipping scaling corrections
      (pclip = -0.5)           pclip: Percentile clipping parameter
       (grow = 0)              Radius (pixels) for 1D neighbor rejection
      (blank = 0.)             Value if there are no pixels
       (mode = "ql")
```

Some example output from SCOMBINE:

```
Output image = orig.wav, ncombine = 107, exptime = 288900.
 w1 = 3506.452, w2 = 10611.28, dw = 0.0457617, nw = 155258., dtype = 0
```

Then using the output of the SCOMBINE, you can continuum flatten your object spectra via SARITH. At the IRAF prompt, do:

```
sarith orig.wav / cont.wav  foo
```

Check your flattened spectrum "foo" with SPLOT. Below in Figure 5 is an example of the flattened, 1-D spectrum.

You can check beloved lines in the spectra to see how good your wavelength calibration and continuum flattening worked out. If you are viewing your continuum flattened spectrum with SPLOT, you can use the windowing commands "w, e, e" or "a, a" with the cursor is at the lower left and upper right of the region you want to zoom in.

As a guide, here are some wavelengths you can check:

Hα 6563

MgII 5167, 5173, 5184

Na doublet 5890, 5896

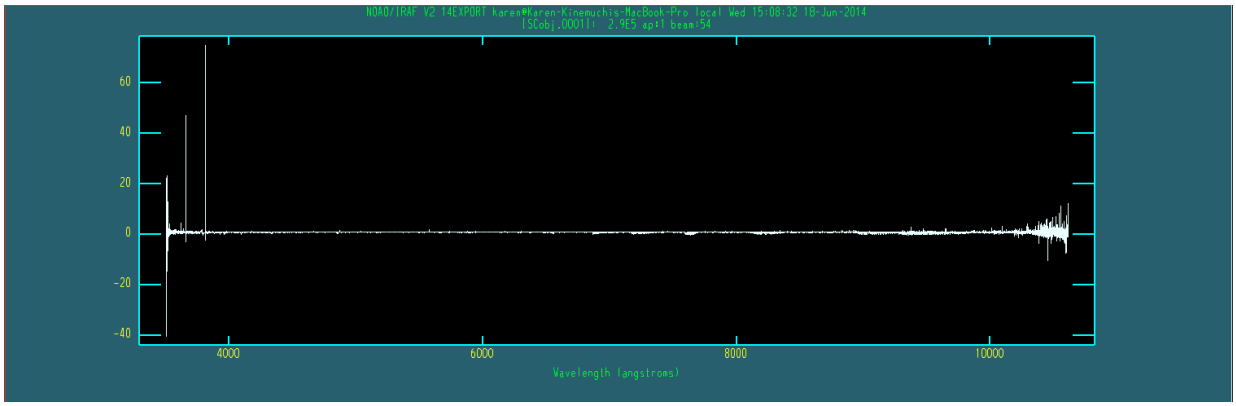


Figure 5: Example of the continuum flattened spectrum from the method described in Step 23.

Optional: If you want to remove any high or low spikes from your continuum flattened spectrum, you can use IMREPLACE like in Step #11.

```
imreplace foo value=2 lower=2 upper=INDEF
imreplace foo value=-0.2 lower=INDEF upper=-0.2
```

24. That's all folks! Your spectra should now be ready for analysis!