LRIS-B Longslit Data Analysis

Eric Gawiser, 12/5/00

I. Preliminary reduction

(a) I describe data analysis procedures for the blue and red sides of LRIS (LRIS-B and LRIS-R). The current numbers are specialized for observations of bright (m=18) quasars with the 600/4000 grism and 600/5000 grating with the D500 dichroic. The final section describes the search for Damped Lyman α Absorption systems (DLAs) towards those quasars; until then these reduction procedures should be rather general and easily adaptable to other science goals and instrument settings.

(b) Philosophy for what follows: I want the statistics of the sigma spectra to be correct. Since we may use different instruments with different gains the safe thing is to turn ADUs back into electrons as our first step of processing. Note that *italics* describe IRAF routines and parameters and **boldface** describes UNIX commands and files. All non-standard iraf packages described here can be found in **~gawiser/iraf/bogus** or **~gawiser/iraf/wmkolris** and the **lamoffset150** code and examples of other relevant files are in **~gawiser/iraf/bogus/examples**. You will need to adapt the relevant portions of **~gawiser/iraf/loginuser.cl** in order for IRAF to see those packages.

(c) We will OV-subtract, gain-correct, and trim all images (2-D multislit spectra, arc-lamps, flats, and zeros/darks) using *nglccdproc* (ngl stands for no-gain, lris). Always work in a directory other than the raw directory since IRAF will overwrite files in defiance of all protections. *nglccdproc* calls *nglrisbias* which uses the ratio of input parameters *gain_left* and *gain_right* to set both to *gain_left*. *nglccdproc* then multiplies both by *gain_left* and sets header card *gain*=1.0. You can determine gain_left/gain_right empirically using a domeflat or skyflat but not from halogen flats of course. Current LRIS-R gains are listed as 1.97 (left), 2.10 (right) on their web page. With the engineering CCD, LRIS-B has gain=2.25, rdnoise=5.5 according to Steidel's notes, and our LRIS-B frames are consistent with those numbers.

(d) First *nglccdproc* the zeros/darks. Iraf might complain that **instr_lris_img.dat** is not found - ignore that. But if that file is present you should remove it from the current directory since all it does here is cause trouble. If they appear relevant, zeros/darks can be subtracted from the rest of the frames using the *zerocor* and *zero* parameters of *nglccdproc*, or by first running *nglccdproc* without zero correction and then saying *imarith frame - zero_bias frame* if there is a uniform bias *zero_bias*. Even a couple-pixel offset is worth fixing - especially if it's not uniform. If it is, it will be subtracted by *apall* during sky subtraction and this uniform offset should not significantly affect the uncertainty of sky subtraction. Even a uniform offset would affect the normalized flat produced by *response*. Think about how accurate you need image/flat to be... Either way, go ahead and run *nglccdproc* on all other object frames now.

(e) If you need to edit single header cards, the easiest way is with *hedit header_card value=new_value add+ update+*. Don't ask me why but you need to say *add+* in order to add a new header card **or** update an old one. An alternative that's good for altering a lot of the header at once is the unix command **hdedit -novi**.

(f) All subsequent IRAF routines must now be fed *gain=1.0*. You'll also need to adjust *saturation* in routines that have it to reflect saturation in electrons instead of ADU (both LRIS-B and LRIS-R CCDs have saturation=65535 ADU at present). You should update the *rdnoise* value in the header if multiple images are combined and enter the updated value in the corresponding iraf parameter for routines like *apall* that use it. Remember the rdnoise is given in electrons so it doesn't change when we set the gain to 1 (currently rdnoise=5.5 e- for LRIS-B and 6.5 for LRIS-R). Always check IRAF routines for gain, rdnoise, and saturation because many routines have these parameters although they may not always use them. This really shouldn't matter until doing the statistics of a sigma spectrum in *apall*, though.

II. Flatten the 2-D images

(a) We will also combine the 2-D spectra if multiple exposures were taken of a given object. If you want to do fringe subtraction as well I recommend using the *bogus* software described in the accompanying LRIS Multislit Data Analysis cookbook by pretending that you're reducing a slitmask with only one slitlet. That cookbook then covers everything here except for the last section on finding DLAs.

(b) We need to make a flat and choose an arc_lamp. For the arc_lamp, choose the best-looking one that was taken at the closest possible telescope position to the data. We will use night skylines to correct the wavelength solution for flexure if the arc was taken at a different telescope position. If multiple flats are available, check for cosmic rays. If none are found, just add the flats. Note that if there are ≥ 2 flats you can *imcombine* them using *combine=average,reject=crreject* (or *combine=median* for ≥ 3). Unless the medians of the flats are identical you should scale by the median before combining them.

(c) Now run *response*, which "normalizes" the flat so that it only represents pixel-to-pixel variations. You can have it fit the response interactively, which is recommended! (*func=spline3 order=15* is a reasonable default; *spline3* works better than *legendre* or *chebyshev* for this.) The idea is to match how well the response was fit for the spectrophotometric standard in making a sensitivity function (see below). If the fit is indistinguishable from the data on the full window plot, it's probably sufficient, although you often have to zoom around the peak to decide what order is appropriate. Be careful with *func=spline3* not to go too high in order since it can fit almost anything, but order 15 to 20 is reasonable here!

(d) Now just divide by the normalized flat, *imarith object_frame / norm_flat*. This should correct for pixel-to-pixel variations.

(e) If there are multiple exposures to combine, combine with imcombine and a shifts(offsets?) file. Again, if not using something fancy like *xzap* you can combine using *reject=crreject* but now you want to average not median. During *imcombine*, we scale by ELAPTIME so that sigma clipping makes sense - the images should be similar in counts/s not total counts if they have different exposure times. One danger is varying sky level - we use *reject=sigclip* to get an empirical sense of how much this is varying; as long as we don't reject values less than 2 sigma from the mean (*epar imcombine* and set *lsigma=hsigma=2*) this should allow for reasonable sky variation but still nail residual cosmic rays, 0 values, or similar calamities. You really want to add, not average so proper statistics requires multiplying the averaged images output by *imcombine* by the number of input images (which is shown in the NCOMBINE header keyword added by *imcombine*), updating the ELAPTIME header card to account for that, and updating the RDNOISE header card to equal RDNOISE*sqrt(NCOMBINE). You will want to *epar hedit* and set *show=yes,enable=yes* in order to allow header updating and have a record printed to screen. If you want to avoid having to approve each action, set *verify=no*.

(f) Make sure *xzap* has *min,max* = -+ 127100 (almost saturation). This is a Mark Dickinson custom routine that you're officially supposed to get his permission in order to use. We can get rid of cosmic rays by calling *szap* which calls *xzap*, or we can use clean=yes in apall below. NEED TO TEST IF A LIST OF IMAGES CAN BE USED, ALSO IF NEED TO DELETE CRMASK*PL BEFORE STARTING OR RENAME CRMASK* AFTER EACH RUN.

III. Extract the spectrum (do this section once for each object)

(a) Using *apall* is a bit of a black art.

i. *epar apall* to check parameters - recommendations: *format=multispec, interac=yes, b_grow=1, t_nsum=10, t_step=5, t_order=2, backgro=fit.* All other early yes/no parameters are yes except for *resize=no.* The t_ stuff is for tracing. Don't forget to update *rdnoise* to match the value in the image header if multiple exposures were combined.

ii. set *clean=yes* unless Cosmic Rays were subtracted using *szap/xzap* above.

iii. Dan Stern recommends using *lower=-3.5, upper=3.5* for an aperture for LRIS and says that the "optimal" extraction usually doesn't make much difference. I agree - the unweighted version of the spectrum is usually quite similar to the variance-weighted extraction because they're sky-noise dominated, but the weighted version should be slightly better. The aperture should be bigger for extended objects or bad seeing - in general we want the fwhm of the aperture to equal the fwhm of the seeing.

iv. Another parameter to check is *skybox=1.* This would perform boxcar smoothing of the sky background before subtracting it, but setting it to one does nothing. The signal to noise for m¡=20 quasars is good enough in regions of sky continuum that such smoothing would only serve to broaden the impact of skylines, which is the opposite of what we want to do.

v. Important note: I recommend using 2 sigma for both *low_reject* and *high_reject* during the background subtraction phase of *apall.* Once you get rid of any bright sources, the sky outline is noise dominated, and using asymmetric rejection criteria creates a bias, e.g. an underestimation of the true sky level. You know that things went well if the residual skylines in the output stay equally bright when you invert the color table - this means that they are just regions of higher noise but have been subtracted well leaving a zero mean.

vi. Deciding what order polynomial (I recommend avoiding splines here) to use in sky subtraction is subtle. For longslit don't be afraid to use order 6 or higher. Edge effects could justify a higher order but it's better to just section those out. You can use the rms as a guide but make sure that you believe the interpolation across the object location - that's all we really care about, after all! A clue that the order is too high is oscillation of the 2nd derivative - we don't really expect the illumination pattern to have any inflection points within a slitlet. On LRIS-B there are no skylines bright enough to worry about, although there is one apparently dim one that is probably a reflection of a red skyline.

vii. OK, try *apall objectname.* Typing e.g. *:line 560* takes you to column 560 and then *:nsum 80* makes it sum over 80 columns to see if any objects are there. For LRIS-B try :nsum 800, :line 1400 to use everything from the Lyman limit on. Some strange stuff happens blueward of that, possibly from reflections, so this is a safe range to find the apertures in.

viii. *b* checks background, *w a* shows whole background region, *t* erases the sample regions, *s s* chooses a new sample region and can be used repeatedly, *f* refits, and *q* declares victory and moves on. Remember, we have already run *background* once as part of *bogus* so the sky should have been subtracted. This is a chance for us to fix any clear under- or oversubtractions but only a low order polynomial (1-2, at most 3) is justified since we're probably looking at mostly noise. GIVE INSTRUCTIONS.

ix. Hit *q* to trace. You definitely want to trace the aperture interactively. You may need to use *s s* to section and to delete several points using *d* in order to make the trace go where you want.

x. Beware of *iraf*'s tendency to overwrite things when you don't want to and quit without overwriting when you did want to.

(b) When the spectrum is extracted, display it using *splot objectname.ms*. You can make *apall* review the extracted spectrum but it's not interactive. In *splot* you can blow up regions using *w e e*, use *o g* to overplot, and use *s 3* to smooth by e.g. 3 pixels. More than 5 pixel smoothing degrades the resolution significantly (any smoothing degrades the resolution somewhat) but can be useful for comparison with lower-resolution spectra.

IV. Wavelength and flux calibrate

(a) Make sure wavelength ranges of *dispcor* and *scopy* are set equal! Otherwise fnd_DLA will blow up for some reason later on.

(b) Making a wavelength solution (do this at least once)

i. *apall arc_lamp refer=objectname out=lampref_objectname int- fit- tra- rec- back-* extracts an arc spectrum from the arc_lamp slitlet in exactly the same way that you just extracted the object spectrum. If you haven't tried extracting any object spectra yet, then ignore the refer=objectname part and call the output lampref; apall will extract the center line of the 2-D image. Probably the best idea is to extract the standard star spectrum (see below) and then refer to that since this is the first object we will use the wavelength solution on.

ii. Now *epar identify* and put in the list of arc lines you want to use e.g. arc_lins.dat. Run *identify lampref_objectname*, use *w e e* to zoom, *m* to mark a few line IDs by comparing with the arcline plots from the LRIS website and entering your guess at their wavelength, *l* to go to the list of arc lines and have IRAF guess at the rest, *f* to fit, and *h* to give you a plot of residuals that should be a straight line. Be careful - Xe 8819 is **bright** but not on the arc line plots and is therefore easily confused with Ar 9122. Use *z* to check all of the lines fit by IRAF, hit *d* to delete a blend or really weak line, *n* to move on to the next one, and *p* to exit the zoom mode. Once the rms is less than a pixel (1.3 Å for the LRIS-R 600 line/mm grating, 1 Å for the LRIS-B 600 line/mm grism) that's probably good enough.

(c) Making a sensitivity function (do this once per observing run)

i. You really want to observe the spectrophotometric standard star both with and without a dichroic filter so that 2nd order light doesn't ruin the red half of your sensitivity function. If you're using both sides of LRIS just observe the spectrophotometric standard star with the same setup you're using for science.

ii. We must process the spectrum of the spectrophotometric standard star analogously to how we have processed the other science objects i.e. run *nglccdproc* on spectroscopic image(s) of the spectrophotometric standard star, divide by an appropriate normalized flat, and extract the standard star spectrum using *apall*.

iii. Now we use *refspec* and *dispcor* to wavelength calibrate each image using the wavelength solution we made above. It's best to check linear vs. non-linear in *dispcor*, but it seems to make no difference. *flux=yes* because you want to conserve photon counts.

iv. In *noao.onedspec* you will find the tasks *standard* and *sensfunc* which you will now use to make the sensitivity function. *epar standard* and input the name of your spectrophotometric standard star. *standard* reads in a one-dimensional wavelength-calibrated spectrum, averages that spectrum into ∼50 Å bins, and then writes out a file listing your counts in those bins and the calibrated flux in those bins. To be careful, you should use the extinction correction here and during *calibrate* below in order to properly account for a difference in airmass between when the standard star was measured and the calibrated spectrum and for a difference in airmass between standard star and science observations. *extinct=yes file=keckextinction.dat*. This doesn't seem to make a big difference, though. *sensfunc* then reads in this output file (there can be multiple if you took multiple observations) and determines the average sensitivity function to transform counts into flux. You should control the choice of function and order interactively; *func=spline3* is probably necessary to get a good fit. Note that it is very important to have the right exposure time listed in the header for each standard star frame and to have the gain in this image be the same as in everything you will apply the sensitivity function to (should be 1).

v. *epar calibrate* and put in the name of the sensitivity function to be used for flux calibration. Make sure that *fnu=no* since we use f_lambda units for quasars (but f_nu i.e. *fnu=yes* for galaxies) and *extinct=yes* to handle airmass differences.

vi. There's this cool routine called *sbands* that you can use to figure out what the photometry of your standard star should be and thereby cross-check this sensitivity function.

(d) Now wavelength and flux calibrate each object

i. *apall arc_lamp output=object_lamp.ms refer=object int- fit- tra- rec- back-* to extract the arc_lamp spectrum the exact same way as the object spectrum was extracted.

ii. *reidentify refer=lampref images=object_lamp.ms int-* to use an existing wavelength solution (lampref) and alter it slightly to fit the object_lamp.ms spectrum.

iii. *refspec object.ms refer=object_lamp.ms* adds a header card that points to the wavelength solution in object_lamp.ms.

iv. *dispcor object.ms object.w* finishes wavelength calibration by assigning the dispersion function to the object spectrum.

v. *splot object.w band=3* to show spectrum of sky extracted during apall. Mark skylines at 5577.3, 5892.9 (blended), 6300.3, 6363.8, and 6864.0 Å (the 2nd of the last set of lines for 600/5000 grating) using *k k* in splot (*w e e* can be used to zoom); this writes the apparent wavelengths to **splot.log**. Hit q when done. **splot.log** should have 2 blank lines, a header, and 5 data lines. Run a fortran program called **lamoffset600** (you can call this with *!lamoffset600*); this reads **splot.log** and calculates the pixel shift for each of these skylines and the average pixel shift. If there is more than about 1 Å dispersion in these shifts then it is time to make a new lampref either by using reidentify interactively or starting with identify and making a new wavelength solution from scratch.

vi. Decide what pixel shift you believe and execute it with *specshift object.w shift* (same sign as lamoffset600 suggests).

vii. If the EXPOSURE and ELAPTIME header cards are different, use *hedit* to alter EXPOSURE to equal ELAPTIME. For some reason, *calibrate* insists on using EXPOSURE so that is what will matter next.

viii. *calibrate object.w object.f* flux calibrates.

ix. When you examine the flux- and wavelength-calibrated spectrum object.f in *splot*, *$* switches between wavelength and pixel number as x-coordinates.

x. If you want an ascii version of the spectrum, use *scopy* with *format = 'onedspec'* and then *wspectext.*

V. Search for Damped Lyman α Absorption systems

(a) Getting set up (do this once)

i. printenv PGPLOT_FONT to make sure it's the equivalent of /usr/local/pgplot/REAL/grfont.dat

ii. Launch pgdisp with
/usr/local/bin/pgdisp -geometry 692x540+450+0 -lgCrosshair >&! /dev/null &

iii. In a separate window, install Jason's ESI tools package. ftp ftp.ociw.edu, cd pub/xavier/ESI, get ESI_tools.tar.gz (also found in gawiser/esi). If trouble do make clean, then make. Now need to setenv PGPLOT_FONT lib/pgplot/grfont.dat to use his version.

(b) Continuum fit using **xplot** function of **makee**

i. *scopy object.f[*,1,1] object_f* to make one-D spectrum.

ii. *scopy object.f[*,1,4] object_e* to make error array.

iii. **xplot object_f**. h -> histogram mode. zoom is left-mouse, right-mouse. Vertical scale is right-mouse, right-mouse. w -> whole spectrum. space-bar, space-bar picks a region and it prints the data value of your x-position. Then c -> continuum fit mode, d deletes region you just picked. left-mouse adds points. enter a 2-digit order number to fit. s -> save. Can do continuum piecemeal, s after each section. ? gets commands, ESC to stop, 1 to write. Save as con_object_f.fits.

iv. **opim div object_f con_object_f div_object_f** at unix prompt. This creates continuum-divided spectrum.

v. **opim div object_e con_object_f div_object_e** for new error array.

(c) Find DLAs with **fnd_DLA**

i. **emacs Input/fnd_DLA.inp &** and put div_object_f.fits in first row. Rest should be blah then 60.0 (dispersion of instrument) then 4000,6600 which is an adjustable wavelength range.

ii. **fnd_DLA**. Use l r to zoom, n for new DLA center, then l r for EW range - it plots a Voigt profile starting at 20.3 or most recent value. c moves the center, $+-$ keys (without shift) change the N_HI value. w -> whole spectrum. P for postscript, x to quit.