Chapter 3

EXTRACTING AND FLUXING EUVE SPECTRA

All EUVE spectra delivered to Guest Observers have been extracted using the IRAF/EUV task **euvred.euvextract**. The same set of spectral and background apertures is used for all GO extractions. Extraction and background subtraction are done on a rectangular aperture in each 2-d spectral image, whose dimensions have been chosen to ensure that as much of the source flux as possible falls within the spectral aperture, assuming that the spectrum is perfectly centered[1]. A detailed description of EGO Center extractions may be found in section 4.5 of the *EGO Center Data Products Guide*. No effort is made to optimize the extractions on a case-by-case basis. It is therefore recommended that all GO's perform a re-extraction of each spectrum, to correct any possible omission of source counts.

The aperture used (25 rows) may also seem quite wide for faint spectra, as the source counts in the tails of the distribution are often overwhelmed by noise. The signal-to-noise of such spectra may be improved by "tailoring" a narrower aperture, but if a narrower aperture is used, some source flux will be eliminated. Any absolute flux measurements based on the extraction will then need to be corrected (if possible) or accompanied by some estimate of the systematic errors introduced. Relative fluxes from different parts of the spectrum will have different levels of systematic error, as the spectral profile changes slightly with wavelength.

IRAF spectral terminology:

It will be helpful in reading this chapter to have the following definitions firmly in mind. In IRAF the term "dispersion axis" means the image dimension along which wavelengths are dispersed. Each spectral image contains the header card `DISPAXIS`, which is assigned a value of 1 for dispersion along lines or rows (X) of the image, and 2 for dispersion along columns (Y). The dispersion direction must also be indicated in the parameter files for many of the tasks in **twodspec**, such as **apextract.apall**.
The term "dispersion line", on the other hand, means pixels along the dimension perpendicular to the dispersion axis, whether that be along rows or columns. Therefore, since for EUVE images the dispersion axis is along X (rows, DISPAXIS=1), dispersion lines correspond to image *columns* along Y. "Dispersion line" is used in a number of help pages and parameter files, usually in reference to producing a spectrum profile for aperture centering.

[1] Empirical tests indicate that the 25-row aperture includes roughly 98% of the source flux in a centered spectrum.

This chapter describes the use of the IRAF/EUV extraction task **euvextract**, as well as extraction with the standard IRAF aperture-editing task **apextract.apall.** Tasks are introduced which can produce estimated fluxes and subtract the effects of multiple spectral orders.

3.1 Extracting EUVE Spectra

It is assumed throughout that spectra are being extracted from a two-dimensional spectral image created by filtering the event list of an EUVE QPOE file. Two-dimensional exposure-corrected spectral images are supplied on the Guest Observer data tape, and can be used without accessing the QPOE files if no other data filters are needed for GO science. The spectral images have been trimmed to the physical coordinate section [1:2048,720:1299], and have logical coordinates of [1:2048,1:580][2]. All locations specified on images with the names *sw.imh, mw.imh,* or *lw.imh* are in these coordinates. If one is extracting spectra from 2-d spectral images made by some other means, such as using **qpmkim** with a custom data selection filter, aperture centers and dimensions may differ from the examples below.

Related topics

◇ See section 4.5.1 of the *Data Products Guide* for a diagram illustrating the image sections used to create the 2-d spectral images and the extracted spectra.

3.1.1 EUVEXTRACT creates and documents EUVE spectra

The task **euvred.euvextract** performs a spectral extraction with subtraction of averaged background, and produces a $[2048 \times 1 \times 3]$ dimensional spectrum image with the background-subtracted spectrum, standard deviation, and average background in three separate bands. The user may define the aperture center and size, may apply a simple centering algorithm, or let the center value default to the center of the image. The centering algorithm sums a specified set of dispersion lines (columns) to find the row corresponding to the highest resulting pixel value. Spectral and background apertures are defined in terms of image rows above and below the center of the spectral aperture [-low:+high], following standard practice in IRAF. The sides of the spectral aperture but need not be symmetric about the center. Multiple background apertures may be specified as a comma-separated list of sections in Y, with the same syntax and using the aperture center as the reference.

[2]Logical coordinates are the default for most plotting tasks in IRAF.

Do I need to run euvextract?

The EGO Center advises GO's to re-extract their spectra, and most GO's will find the task **euvextract**, which uses a straight rectangular aperture and averaged background, is adequate for this purpose. Since EUVE spectra are typically faint and difficult to trace, optimal extraction is not usually practical. Those with bright spectra, especially continua, or long exposures may wish to use the more flexible and complete task **apall**, which is described in section 3.1.2.

A sample parameter file for **euvextract** is shown in figure 3-1.

```
eu> lpar euvextract


      input = "lw"             Input image
     output = "alcen_lwsp"     Output spectrum
    (center = 294)             Spectral pixel position
   (spec_ap = "-7:7")          Spectrum aperture
  (seekspec = yes)             Find the spectral position?
   (cntr_ap = "168:179")       Centering aperture
    (radius = 100)             Radius from center to seek
     (thres = 4)               Threshold, sigma above the noise
(background = "smooth")        Background computation method
    (bkg_ap = "-84:-10,10:84") Background sample region
(bkg_fltr_sz = 7)              Size in pixels of bkg smoothing filters
   (verbose = no)              Print informational messages?
      (mode = "al")
```

Figure 3-1: Parameter file for extracting a spectrum of Alpha Centauri with **euvextract**, using automatic centering and background smoothing.

Notes on running **euvextract**

- Use **euvextract** after examining the spectral image or QPOE using **display** and a 2-d image tool, or by plotting profiles with **splot** or **plot.implot**. Note the apparent location of the spectrum, and determine the areas you want to use for background subtraction, making sure they are free of hotspots, deadspots or other local variations.
- The value of `center` may be specified, or left as "INDEF", which defaults to the center line of the input.
- Specify the spectrum and background apertures as $\pm$ limits with respect to `center`. You may use multiple background apertures, such as ranges of lines above the spectrum (+a:+b,+c:+d) and one below (-e:-f,-g:-h).
- If the centering feature of **euvextract** is used, `seekspec` must be "yes" and `cntr_ap` should be a section of columns where the spectrum is bright. Use columns that

span one or more visible features A comma-separated list of sections can be used to select several features.

- □ The value of `radius` must be large enough to include both the spectrum and a representative sample of the background above and below for the centering algorithm to be effective. It is recommended that the default value of 100 not be changed.
- □ The `background` computation method may be "average", "smooth", or "none". For the first two, the background is averaged in Y and scaled by the width of the spectral aperture. If "smooth" is chosen, boxcar and median filters of width `bkg_fltr_sz` are also applied before subtraction, and the background subtraction is assumed not to add additional noise. Samples of the averaged and smoothed backgrounds for an SW spectrum are shown in figure 3-2.

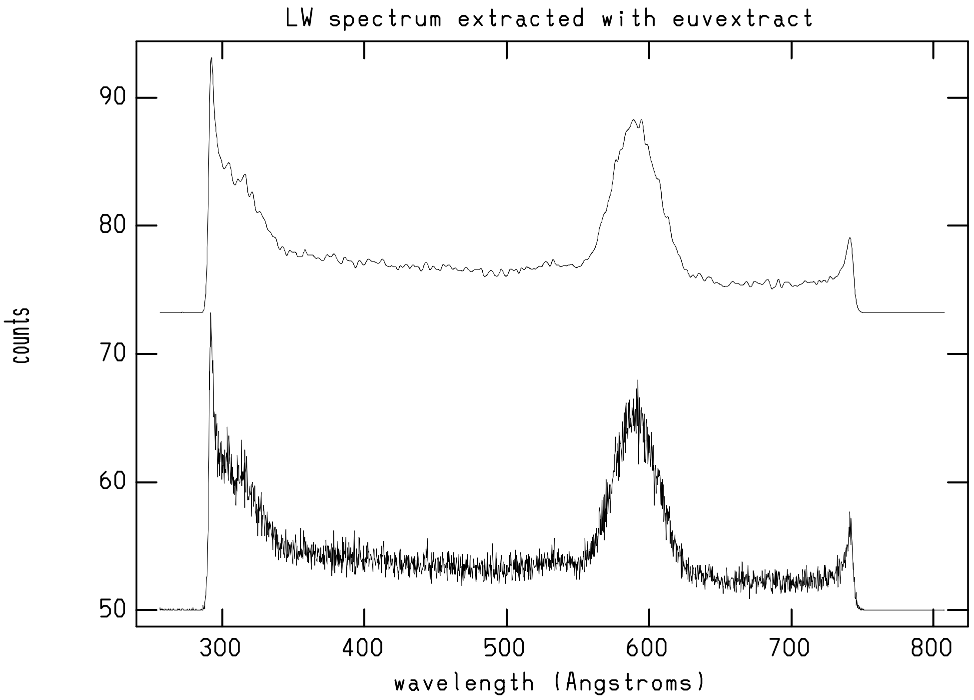

Figure 3-2: Averaged (bottom) and smoothed (top) backgrounds computed with **euvextract** for a spectrum in the SW Spectrometer.

When the task is run, the spectrum pixels are summed along each column. All the background pixels at the same column are averaged and scaled by the spectrum aperture width. The background spectrum may be smoothed by filters of user-specified width before being subtracted. The spectrum, minus the average background ($\bar{B}$) at each column, is placed in the first image band. The 1-σ standard deviation from counting statistics is calculated from the spectrum and background counts, using the definition:

$$\sigma_S = \sqrt{\mathrm{S} + \bar{B}(1 + 1/n)}$$

where S is the subtracted spectrum, and n is the ratio of the background to spectrum aperture heights. The errors are placed in the second band. The averaged background spectrum is placed in the third band.

The error calculation assumes that background subtraction adds no noise ($n \rightarrow \infty$) when `background` is "smooth." Subtracting a smoothed background is therefore advantageous for relatively bright spectra with substantial backgrounds (average pixel > 1 count), as it produces a higher S/N spectrum. However, smoothing may cause under- or over-subtraction near sharp spectral features and edges, so users must think carefully about what background regions they should use.

When `seekspec` is "yes," the parameters `seekspec`, `cntr_ap`, `radius`, and `thresh` govern the spectrum-finding algorithm. The algorithm sums together the columns specified by `cntr_ap`. Then the highest valued pixel within `radius` of `center` (in the Y direction) is chosen as the peak position, subject to the requirement that its value is more than `thresh` standard deviations above the mean of the nearby background.

The background mean and standard deviation for the area within `radius` of `center` are computed by an iterative process. First they are computed for the entire distribution, and any points more than `thresh` standard deviations from the mean (e.g. the bright columns with the spectrum) are eliminated. Then the process is repeated until no more eliminations occur, and the last mean is assumed to represent the background. The reason why the peak finding `radius` should include mostly background area is that the background on either side of the spectrum should dominate calculations of the mean after a small number of iterations. If there are no bins within `radius` that exceed the mean by `thresh`σ, the task cannot distinguish any peak from the background, and the aperture center defaults to `center`.

Example 3.1 : Extracting a spectrum with euvextract

Suppose that a display of the short wavelength spectral image for an observation of the target EQ Pegassi shows a faint but visible emission spectrum with a height in Y of 15 rows at its brightest. The GO may be able to improve the signal-to-noise over the automated extraction, without losing an appreciable quantity of signal, by re-extracting the spectrum with a narrower aperture than was used in the nominal extraction.

To get a more precise estimate of the aperture width, the GO notes the approximate locations of some features at different points along the spectrum and plots several sections of columns in **implot**. Figure 3-3 shows three profiles, centered at columns 1025, 795, and 446, overplotted in the graphics window. Each profile is averaged over 10–12 columns. The GO uses cursor reads to determine that an aperture from 298 to 313 will include the vast majority of source counts.

The GO also notes the locations of two sections that are well separated from the spectrum, and each 100 lines in height, to give an overall background-to-spectrum ratio of 12.5. Since this is an emission spectrum, the section used for the profile-finding algorithm in **euvextract** must be on at least one of these features; the GO uses all three lines for peak-finding.

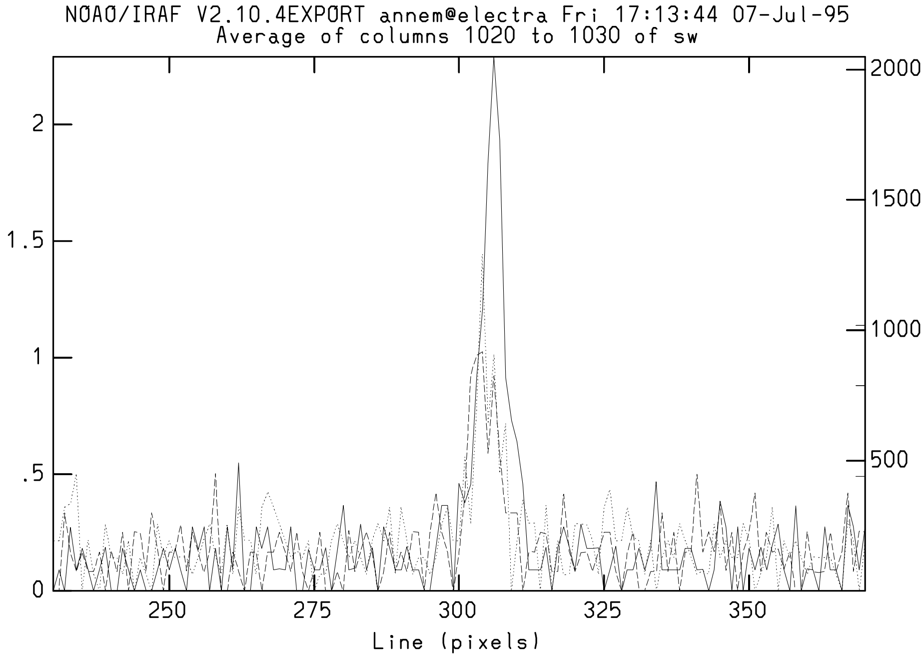

Figure 3-3: Spectrum profiles for several lines, plotted with **implot**. Each profile is an average of several columns from a different part of the spectrum.

The GO enters the following on the command line to perform the extraction:

```
eu> euvextract lw eqpeg_swsp center=306 spec_ap=-8:+8 seekspec+ 440:453,
789:800,1020:1030 bkg_ap=-114:-15,15:114 bkg_fltr_sz=7
```

The extracted spectrum, error band, and background are plotted in figure 3-4.

The size of the background smoothing filter used corresponds roughly to the size (in pixels) of the resolution element for each of the spectrometer detectors. The background begins outside the width of the default spectrum aperture used in nominal processing (±12 pixels), so that source counts in the tails of the spectrum distribution do not add to the background subtraction. The profile finding algorithm located the peak of the profile at row 306.

3.1.2 Extracting EUVE spectra with APALL

Some GO's may wish to use the tasks in the **twodspec.apextract** package to extract their spectra, to make use of the aperture editing, tracing and background fitting functions available in these tasks. Using **apextract.apall**, one may define an aperture, set background sampling regions and subtraction methods, and attempt tracing the spectrum interactively before extraction. If these tools are applied carefully, an

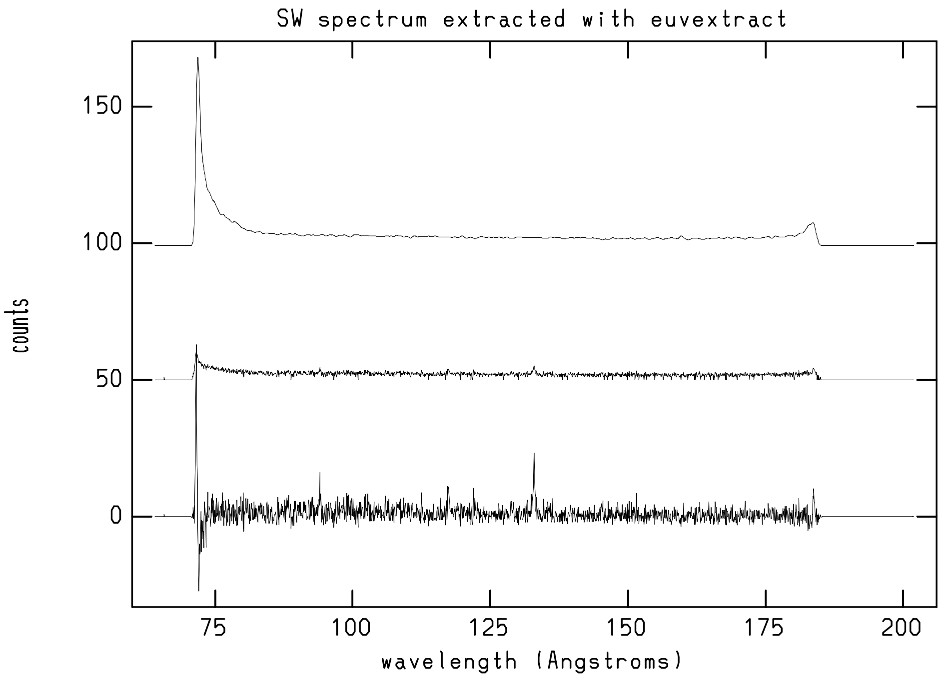

Figure 3-4: Background-subtracted spectrum of EQ Pegassi (top), the standard deviation (center), and the smoothed background (bottom). Extracted with **euvextract**.

increase in signal-to-noise can sometimes be achieved. Users who are not already familiar with **apall** should read the help pages for the individual tasks **apedit**, **aptrace**, and **apsum**. More detailed information on the algorithms used can be found in the IRAF User's Guides on **apextract** tasks, listed in Appendix D.

Since the **apextract** package was written for CCD data, the algorithms are not well suited to the low signal and high noise levels often present in EUVE data. In particular it is very difficult to trace most spectra. Most EUVE spectra have noise and small amounts of residual curvature near the ends, caused by varying detector field effects and poor distortion correction for these areas. Even with a bright continuum, some pixels may be lost to background subtraction noise at the ends of any spectrum. The overlapping spectral bandpasses of the channels should allow extraction of every wavelength in at least one detector. Bright continuum spectra are the best prospects; they may produce an accurate spectrum trace, which will follow small deviations, but the tracing algorithm may still need help.

Parameter setup for apall

An example of the parameter file for the interactive task **apall** is shown in figure 3-5, parts (a-d). It is partitioned, and shown in the format of **epar**, rather than **lpar**, because it is quite long. However, most users will probably want to edit the parameter file at least once. It actually includes (as the name suggests) the parameters for nearly all of the other tasks in the **apextract** package, which are called from within **apall**.

The user sets several boolean parameters in the first section to call each of these tasks or not, and subsequent parts of the parameter list become useful or not, depending on the choices. The parameters are not used if the task is not called, and can then be ignored. The parts of figure 3-5 associated with functions that EUVE GO's will not use may be ignored. Instructions for each of the sections EUVE GO's might use are included in the notes after each part of the figure.

```
    ap> epar apall

  PACKAGE = apextract
  TASK = apall

input   =  sw_hz43[*,720:1299]  List of input images
(output =            hz43_swsp) List of output spectra
(format =           multispec) Extracted spectra format
(referen=                    ) List of aperture reference images
(profile=                    ) List of aperture profile images

(interac=                 yes) Run task interactively?
(find   =                  no) Find apertures?
(recente=                  no) 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 =                 yes) Extract sky, sigma, etc.?
(review =                 yes) Review extractions?

(line   =                 745) Dispersion line
(nsum   =                 100) Number of dispersion lines to sum
```

Figure 3-5: a) Global parameters for the task **apall**. These govern the calling of other **apextract** tasks, and the image section used to show the spectrum profile.

Notes on global **apall** parameters

- If you are not using one of the two-dimensional spectral images supplied on your data tape, it can be helpful to use an image trimmed to similar dimensions. This omits the bright stim pin events near the detector edges, which tend to divert spectrum traces.
- The name for the output file is a hidden parameter, so it must either be entered in the parameter file, or given on the command line with the `pname=value` syntax.
- The `format` of the output should be "multispec"; which allows the subtracted background spectrum to be stored in a second image band.

- Aperture `reference` and `profile` images are not generally available for EUVE data. Leave these values as null strings.
- Run the task interactively. The IRAF graphics capital letter commands are always available when you are interacting with a display in `STDPLOT`.
- Turn off the calls to tasks for automatic aperture finding, centering, and resizing, as these generally fail for EUVE spectra.
- Turn on the call to **apedit**, the aperture editing task. Use **aptrace** in interactive mode if at least part of the spectrum is bright enough to be seen in **display**. Turn on extraction, background extraction and the ability to review the spectra.
- Set `line` to a dispersion line (column) in a bright area of the spectrum, away from any airglow features. For an emission spectrum, choose a line in the brightest feature. Set the number of columns to be summed, `nsum`, to span the feature, or to at least 20 for continuum sources.

```
                                # DEFAULT APERTURE PARAMETERS

(dispaxi=                      1) Dispersion axis (1=along lines, 2=along columns)
(lower  =                    -7.) Lower aperture limit relative to center
(upper  =                     7.) Upper aperture limit relative to center
(apidtab=                       ) Aperture ID table (optional)

                                # DEFAULT BACKGROUND PARAMETERS

(b_funct=                spline1) Background function
(b_order=                      1) Background function order
(b_sampl=       -114:-15,15:114) Background sample regions
(b_naver=                     20) Background average or median
(b_niter=                      3) Background rejection iterations
(b_low_r=                     3.) Background lower rejection sigma
(b_high_=                     3.) Background upper rejection sigma
(b_grow =                     3.) Background rejection growing radius
```

Figure 3-5: b) Parameters for editing apertures using **apedit** in the interactive mode of **apall**.

Notes on **apedit** parameters

- The `dispaxis` parameter determines the definition of dispersion lines (see box on IRAF spectral terms on page 3-1). Since dispersion is along image lines, `dispaxis` should be set to 1.
- The values of `upper` and `lower` for spectral and `b_sample` for background apertures are specified relative to aperture center, which is NOT set anywhere in the parameter file. The upper and lower limits of the aperture may be changed at will in the interactive mode, so these values can be guesses.
- `apidtable` is not used in EUVE applications.

- Always set the background sampling region. Whether or not the other BACKGROUND parameters are used depends on whether the EXTRACTION parameter background is set to "fit" (figure 3-5 d). The sample file sets b_sample to two 100-line regions. The b_naverage has been set to average (+)20 points together (- produces the median) for each point in the fit. Background parameters can also be changed interactively.
- Skip over parameters for APERTURE CENTERING, FINDING AND ORDERING, RECENTERING, and RESIZING.

The task which uses the next parameter set, **aptrace**, locates a spectrum profile at a number of places along the dispersion axis, and fits a function through the peak locations. All the parameters can be changed while running in interactive mode.

```
                                  # TRACING PARAMETERS

(t_nsum =                     50) Number of dispersion lines to sum
(t_step =                     10) Tracing step
(t_nlost=                     10) Number of consecutive times profile is lost befo
(t_funct=               legendre) Trace fitting function
(t_order=                      2) Trace fitting function order
(t_sampl=               300:1750) Trace sample regions
(t_naver=                     -5) Trace average or median
(t_niter=                      3) 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
```

Figure 3-5: c) Parameters for tracing the spectrum: using **aptrace** in the interactive mode of **apall**.

Notes on **aptrace** paramters

- EUVE spectra are fairly straight, but noisy, so averaging more peaks per point can smooth out the trace. Set t_naverage, the number of columns averaged together for each profile, to 50, or at least as wide as your widest emission feature. The step size between fit points, set by t_step should be fairly small.
- The number of trace points with no profile that are allowed before the trace gives up, t_nlost, should be generous, as this happens quite often.
- The polynomial form is of little importance, as most traces don't involve terms higher than second order.
- The trace sampling region, t_sample, should include only the detector. The example parameters assume a bright continuum, in keeping with our HZ43 example. The areas outside and just inside the detector edges (about 120-140 and 1750-1780 in X), should generally be excluded.

The next set of parameters defines the conditions of the spectrum extraction, including the method used to estimate the background.

```
                                # EXTRACTION PARAMETERS

(backgro=               average) Background to subtract
(skybox =                     7) Box car smoothing length for sky
(weights=                  none) Extraction weights (none|variance)
(pfit   =                 fit1d) Profile fitting type (fit1d|fit2d)
(clean  =                    no) 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
(nsubaps=                     1) Number of subapertures per aperture
(mode   =                    ql)
```

Figure 3-5: d) Parameters for extraction and summing with **apsum** in **apall**.

Notes on **apsum** parameters

- ◻ The type of `background` subtraction may be "none," "average," "median," "minimum," or "fit." This example illustrates the use of averaged background subtraction, so that the sampling region is averaged and scaled to the aperture size as in **euvextract**, and all other `BACKGROUND` parameters (figure 3-5 b, above) are ignored. Background fitting can also be used, and may be better for long exposures, if there is some appreciable variation in the Y-direction.
- ◻ Set `weights` and `clean` to "no," as optimized extraction is usually impractical with EUVE spectra. Other parameters can be safely ignored for EUVE data.

Running apall

The task **apall** combines the steps of finding, editing, and tracing an aperture and extracting a spectrum. It is made mostly of ordered calls to the other tasks in the **apextract** package, any of which can be activated or skipped over either in the parameter file setup or at runtime. A lot of the business of running **apall** is keeping your orientation. While running in interactive mode, **apall** and its sub-tasks issue a number of queries at prospective branch points in the process, making it possible to bypass any single function during the execution. The queries signal that the task is ready move on, and when the user responds, the current subtask, and usually the display, will change.

Besides the different task calls, there are various modes within some tasks. For example, both **apedit** and **aptrace** make use of an IRAF routine called **icfit**, for "interactive curve fitting". Each interactive session with one of the tasks or **icfit** ends when the user types `q` to return to the calling task. Keeping track of the number of nested task calls is important; both for avoiding mistakes and for being sure you know what processing path created the output. After some practice, you will learn to associate the different displays with the various "layers" of the task.

There are four basic steps to extracting most spectra, as enumerated below. You may decide to skip the tracing step if your spectrum is very faint, but it isn't a bad idea to try it once as it illustrates nicely where the spectral profile is and is not obvious. The task can be started from the parameter file by typing `:go`, or typing:

```
ap> apall input_imagename
```

and answering "yes" to the edit query. If you need help on the command options at any stage, type `?` to scroll a help page in the text window. Remember the help page you get depends on (and will tell you) what subtask you are in. Return to interactive mode by typing `q` and then `q` or <cr>.

1. **Define the aperture.** **apedit** has no parameter for the aperture center in **apall**. The task forms the sum of `nsum` columns about the column `line` ("line" is short for "dispersion line", which in the case of EUVE images, is a column). The sum is displayed on `STDGRAPH` as a plot of counts vs. row number. Make a new aperture by placing the cursor on the peak and typing `m`. As soon an aperture is made, the limits are set using the values of `upper` and `lower`. The display should look something like figure 3-6.

   Typing `:parameter` in **apall** causes the current parameter value to be displayed. Typing `:parameter value` sets the parameter to "value". For example, you can adjust the center, upper, and lower limits of the aperture with `:center`, `:upper`, and `:lower`. Or position the cursor and type `c`, `u`, or `l`, and the cursor position is read as the new value.

2. **Set background parameters.** If you have set the parameter `background` to "average," skip this step and type "q" to exit from **apedit**. Otherwise, you are fitting the background (`background = fit`), so use the `b` key command to enter the background fitting mode of **apedit**. This calls the IRAF-internal task, **icfit**, which performs the fitting specified by the `BACKGROUND` parameters, and displays a plot of the background sampling region, summed over all rows in the background aperture(s), with the fit points and the resulting function. The command `:vshow` lists all current **icfit** parameter values. The fit points are averages of `naverage` rows (perpendicular to spectrum). All fit parameters can be changed with colon commands, as in step 1 above. Type `f` after changing parameters to redo the fit and see the new results.

   When satisfied with the appearance of the background function, type `q` to return to **apedit**, which called **icfit**. Now the aperture and background subtraction are set up, so type `q` again to return to **apall**. You will be queried about the next step, which in our example is **aptrace**. Type `y` if you want to trace the spectrum, or `n` to skip on to the extraction in step 4.

3. **Trace the spectrum.** Enter the interactive fitting mode of **aptrace** by answering "y" to the next query (or two). An initial trace and fit are performed immediately using the **aptrace** parameter values (figure 3-5 c). **aptrace** sums each `t_nsum` columns and searches for a peak in the resulting profile. If for any particular profile, no peak is found, the task reports the failure and moves on. It can fail `t_nlost` times in succession before it "gives up". Every `t_naverage`

peak locations are averaged together, and a fit is again calculated by **icfit** to the averaged points. Since `fittrace` is set to "yes" in figure 3-5 c), the display will go back to **icfit**'s interactive mode, and wait for you to adjust the fit.

You will see a plot spanning all the columns of the image, but only a few rows near where the profile peaks were found. A sample display of the first trace for a bright continuum spectrum is shown in figure 3-7 of example 3.2 below. Small crosses mark each of the peak locations, and larger crosses represent the averaged points used in the fit. A dotted line represents the function fitted to the averaged points.

Tracing the spectrum across the entire detector is difficult with EUVE data, as most emission spectra have gaps, and many continuum spectra cut off in mid-bandpass. It may even be possible to see features in a display of the spectral image which **aptrace** will fail to find. As a quick diagnostic, note the scale of the Y-axis: if it spans more than 10 pixels or so, the trace has probably wandered off-course, as will also be visible from the fit line. If the Y-scale spans only 1–2 pixels, you are off to a good start.

You are using **icfit**, the same subtask used for the background fitting, with the same commands: `:vshow` to display fit parameters in the text window, and `?` to display the available key and colon commands. You may wish to adjust the `sample` region, `naverage`, `order`, or `grow`, among others, to change the fit, using `f` each time to recalculate. Aim for a peak function that shifts by no more than 1-2 pixels from left to right on the display. We offer the following (EUVE specific) suggestions:

- In regions where no peak is found, you must add some points yourself to guide **aptrace.** Use the key command `a`, which also allows you to assign a weight to the new point. The commands `x` and `y` can be used to adjust point locations. At least one or two artificial fit points may be needed to make the trace behave reasonably at the ends.
- If there are profiles or points with large deviations at the ends of the fit, the sampling region may include bright rim areas, and should be shortened.
- Increasing the number of profiles combined with `naverage` can smooth out noise in the profile locations. Setting a small `step` with a large `naverage` can help spread the influence of bright parts of the spectrum.
- Single points whose profiles vary greatly from the average can be eliminated by moving the cursor to that point and typing a "d." Points where the peak-finding algorithm has found the bright detector rim or a stim pin (check the locations) are obvious candidates. Deleting individual profiles (small crosses) forces recomputation of the averages used as the fit points the next you type `f`.
- One can also vary the parameters `width`, `radius`, and `threshold` in order to prevent the trace from wandering off of the spectrum. Values of 10, 2, and 0 are good for a start.

To further explore the intricacies of **icfit**, type `?` to display a full list of the available commands. Always use `q` to return from the help listing to interactive mode (it may require two keystrokes) before typing more commands. Please read the **icfit** help page in IRAF for a full description of how the remaining task parameters affect the fit.

4. **Extract the spectrum.** When satisfied with the trace, once again type `q` to leave **aptrace**. **apall** will ask you if you want to save the aperture and trace in a *database* directory; if you want to continue, type "y". You will then be queried whether if you want to extract the spectrum and review the results (the display will not change until you answer). If you answer "yes," **apsum** is run with the `EXTRACTION` parameters (figure 3-5, d). After another query, you can look at the spectrum that has been extracted, to decide whether to save it in the output image with the background in the second band. **apall** has now done its job; the task exits when you type "q" in the spectrum display.

apall and the **apextract** package have a number of other capabilities that have not been introduced here. Read the **apextract** task help pages and IRAF User's Guides if you wish to try variations on any of the steps.

Example 3.2 : Extracting a continuum spectrum with apall

This example illustrates the extraction of a long-wavelength spectrum of HZ43, as set up in our sample parameter file. The GO has remade the two-dimensional spectral image with a user-defined time filter. A section of the [2048×2048] image is used for extraction, to eliminate stim pins and most of the detector rim. After editing the parameters as shown in figure 3-5, she types:

```
ap> apall sw_hz43[*,720:1299]
```

The graphics display shows summed profile of 100 columns centered about column 745. The GO makes an aperture with an `m` command, and uses `:upper` and `:lower` to widen it to ± 8 rows. The resulting display is shown in figure 3-6.

To continue, the GO types `b` to enter background parameter setting mode, in **icfit**. The display is not wide enough to include the entire background region selected, but using `vshow` to display the fit parameters gives:

```
Wed 18:03:08 29-Mar-95
Set Background Subtraction for Aperture 1
function = spline1
grow = 3.
naverage = 20
order = 1
low_reject = 3.
high_reject = 3.
niterate = 3
sample = -114:-15,15:114
```

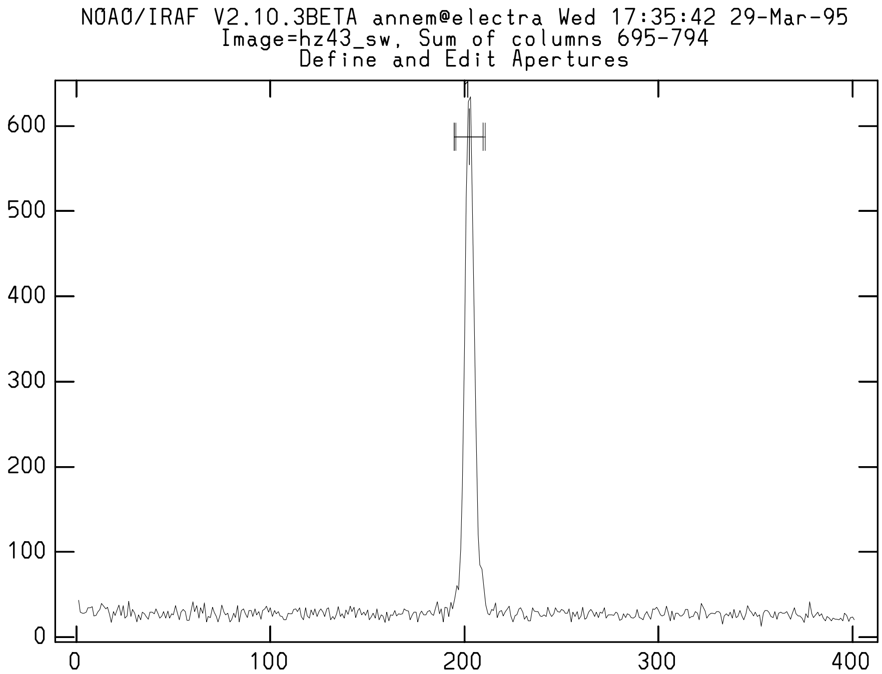

Figure 3-6: Display in **apedit** with a user-defined aperture.

```
total points = 401
sample points = 10
nrejected = 0
deleted = 0
RMS =   1.089518
square root of reduced chi square =   5.425122
          coefficient      error
            2.839199E1     1.352863E-1
            2.695773E1     1.354585E-1
```

The number of sample points is $10 = (100 + 100)/20$ as expected. Redoing the fit or using the graph keys `hijkl` shifts to the **icfit** display, which shows all of the sample area. The GO uses a few `Z` and `M` commands to zoom in on a section of the fit, decides that it looks adequate, and types `q` to return to **apedit** and the aperture display. Another `q` elicits the prompt lines:

```
Trace apertures for hz43_sw?  (yes):

Fit traced positions for hz43_sw interactively?  (yes):
```

The GO types a <cr> (default affirmative) for each query to enter **aptrace**, and the display blinks and returns with yet another query. The GO, now mildly irritated, again types a <cr> and the display returns, showing the initial spectrum trace (figure 3-7).

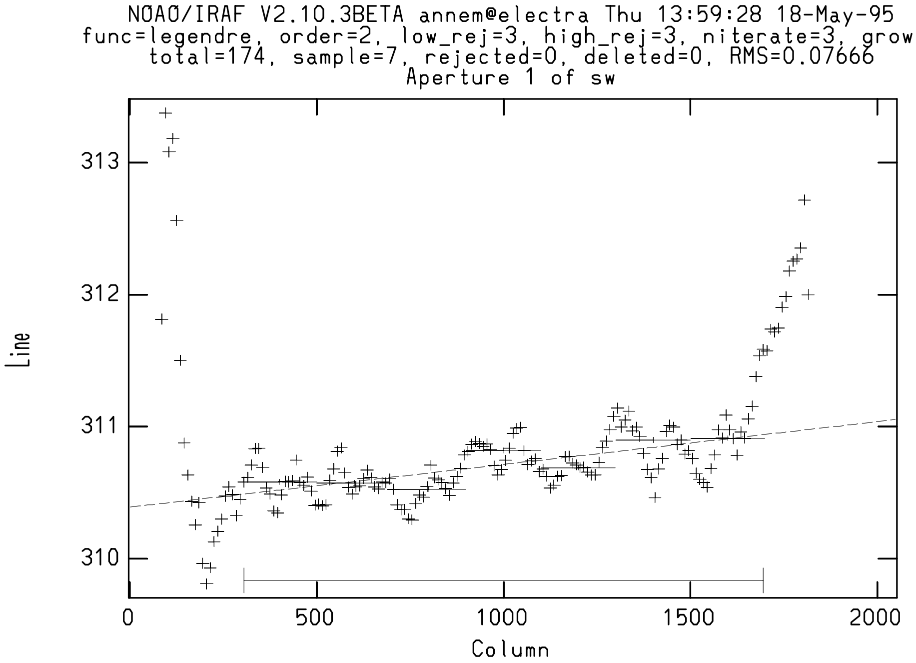

Figure 3-7: Spectrum trace of a bright continuum displayed in **aptrace**. Sample region is indicated by horizontal bar, and small crosses are individual spectrum profile peaks. Large crosses are averaged points used in the fit.

The initial results are fairly acceptable. The trace shifts by only about one pixel from one end of he image to the other; less than a pixel over the length of the spectrum. Note the long horizontal hash marks, which show the sample area where the profiles are averaged to form one fit point. The GO tries several methods of improving the fit. Increasing `naverage` to 20 averages out some of the oscillations in the profile locations. Redefining the sample region to exclude points at the long end and add some at the short end also reduces the RMS a bit. After plotting the residuals using the `j` key command, the GO tries deleting all the profiles with residuals of $|r| > .5$ (figure 3-8), and finds that since only two are within the sample region, this makes very little difference. Since there is no obvious reason why these profiles might not be valid, the GO undeletes them (`u`), and reruns the fitting routine one more time to return to the previous fit. This seems quite good enough, so the GO enters a `q` to move on.

The GO types <cr> to the next queries, writing the aperture and trace to a new *database* directory file. The contents of *database/aphz43_sw*, are shown below:

```
# Wed 18:49:47 29-Mar-95
begin   aperture hz43_sw 1 745. 202.6001
        image   hz43_sw
        aperture        1
        beam    1
```

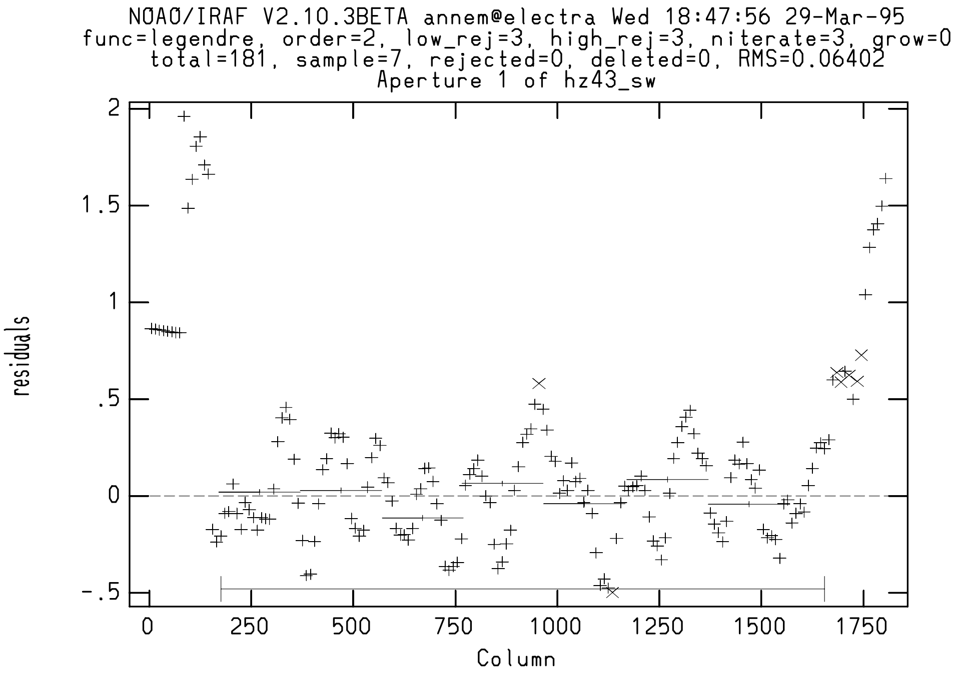

Figure 3-8: Plot of fit point residuals in **aptrace**. ×'s show where profiles were deleted from the fit.

```
center  745. 202.6001
low     -744. -8.
high    1303. 8.
background
        xmin -114.
        xmax 114.
        function spline1
        order 1
        sample -114:-15,15:114 41.12367:46.12561
        naverage 20
        niterate 3
        low_reject 3.
        high_reject 3.
        grow 3.
axis    2
curve   6
        2.
        2.
        1.
        2048.
        0.2402294
        0.3580123
```

The spectrum then is extracted, after several more queries, and the result is displayed, before being written to the output image. The peaks at the ends are normal; as the detector is round, the lines of most background regions don't extend as far as those of the spectrum aperture, causing the under-subtractions at both ends. Also, the particularly bright rim at the short wavelength end of the SW detector is fairly broad in X, and often produces some background over-subtractions in this area. The spectrum is written to the multispec format output image *hz43_swsp*, with the background in the second image band. Both the spectrum and background are plotted in figure 3-9.

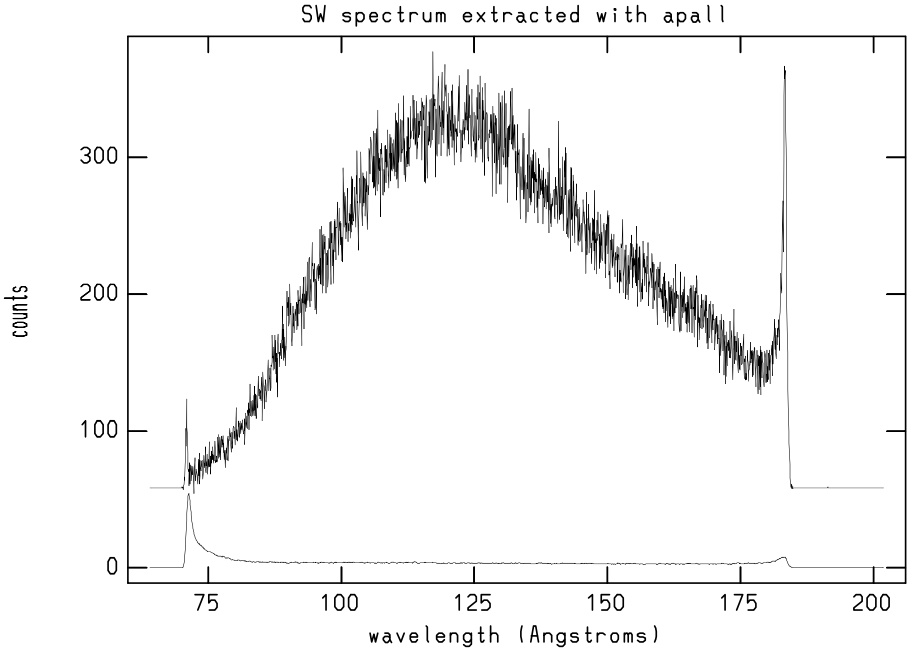

Figure 3-9: SW spectrum for HZ43, extracted with **apall**. The bottom curve is the averaged background subtracted. The spectrum plot is offset by 60 counts.

3.1.3 MKPROFILE fits profiles at specified intervals

The tool **euvtools.mkprofile** is written to create spatial profiles of spectra for use in performing optimal extraction. A sample parameter file is shown in figure 3-10.

The user can specify either the sample section and size of profile sections or the number of profiles to make. The specified number of dispersion lines are summed, a background level is subtracted, and each profile is normalized over a specified FWHM. Background fitting can be done interactively. The profiles are output as lines of a two-dimensional image.

```
eu> lpar mkprofile

       input = "sw"           input image
      output = "prof_sw"      output image
        (x1 = 350)            starting column of profiles
        (x2 = 1500)           ending column of profiles
        (nx = INDEF)          number of columns per profile
  (nprofiles = 10)            number of profiles
   (aperture = "35:535")      profile aperture
   (function = "chebyshev")   Fitting function
      (order = 2)             Number of polynomial terms or spline pieces
     (sample = "*")           Sample points to fit
    (naverage = 3)            Number of sample points to combine
    (niterate = 4)            Number of rejection iterations
  (low_reject = 2.5)          Low sigma rejection factor
 (high_reject = 2.5)          High sigma rejection factor
       (grow = 0.)            Rejection growing radius
       (trim = yes)           Trim low S/N regions of profile?
  (threshold = 0.02)          Profile trim threshold
      (nfwhm = 4)             Number of fwhm's to use in normalization
(interactive = yes)           Interactive display?
   (graphics = "stdgraph")    Device for display
     (cursor = "")            Cursor
       (mode = "al")
```

Figure 3-10: Parameter file for **mkprofile**.

3.2 Spectral Corrections and Flux Calibration

This section covers the use of tasks for processing one-dimensional extracted spectra. The EUVE spectra output by **euvextract**, as well as extractions done with **apall** are binned counts vs. wavelength. The tasks in this section can be used to divide counts by effective area to create flux spectra, and to subtract counts from so-called "higher" inside spectral orders ($n < -1$) from the appropriate bins. In addition to the IRAF/EUV tasks described here, there are spectral analysis tasks in **noao.onedspec** which may be used on EUVE spectra as well. See Appendix D for a listing.

All examples of reference data path names in this section assume that the reference data is located in a directory called *reference*, located immediately below the current working directory. This is in keeping with the standard directory setup illustrated in the *Data Products Guide*, and provided for GO's working at the EGO Center. When using any task which requires access to any reference data file, make sure the path to be used file is complete and accurate for your system. Newer versions of the reference data set can be obtained as needed from the CEA anonymous *ftp* site, in the directory */pub/reference*. See section D.2 in Appendix D for the *ftp* site address.

3.2.1 DIVEA divides by effective area

The task **euvtools.divea** provides the basic functionality for converting first-order EUVE spectra in counts to flux units. It performs a straightforward division by the first-order effective area and the exposure time, and rebins the spectrum at a user-specified resolution. A sample parameter file is shown in figure 3-11.

```
eu> lpar divea

   input = "sw_spec"     Input spectrum
  output = "sw_flux"     Output spectrum
     (dx = 0.1)            Bin width in Angstroms
(exptime = 1.)             Exposure time
(refdata = "reference/detector") Reference file
(detector = "sw")          Detector name [sw|mw|lw]
   (mode = "al")
```

Figure 3-11: Parameter file for fluxing a short wavelength spectrum with **divea**.

Notes on running **divea**

- The value of `dx` determines the binning of the output spectrum. Setting `dx` to "INDEF" suppresses rebinning of the spectrum. Otherwise, the spectrum, and spectrum error band if one exists, will both be rebinned.
- When the input is rebinned, the error band will be divided by exposure time and effective area, and corrected using standard error propagation. The correction does not include the cross-correlation terms introduced in the spectrum by rebinning, and the error band will no longer fully account for the standard deviation. Do any rebinning before extraction, and set `dx` to "INDEF" if preserving the validity of the error band is important.
- The parameters `exptime` and `detector` are used only if the task fails to find this information in the image header. These header cards should be present in any spectrum extracted from an EUVE two-dimensional spectral image.
- **divea** needs the reference data table *detector.tab*, so the parameter `reference` must contain the complete path, including the filename, for your directory setup.

GO's whose data do not contain appreciable contributions from higher spectral orders can use **divea** from the command line to convert their extracted spectra to flux units without going through the order-subtraction process in **suborder**, for example by typing:

```
eu> divea lw_spec lw_spdiv dx=.2
```

If the header card `INSTRUME` is not found for some reason, and the value of `detector` does not match that of the input spectrum (recorded or not), the task attempts to do the division for any part of the spectral bandpasses that overlap. The input spectrum is divided by the effective area of the `detector`, or if the error involves the LW and SW detectors, the task fails with the error message: " divea: Current parameters yield no pixels." Neither of these outputs is especially useful, but they may demonstrate to the user that something is amiss.

3.2.2 SUBORDER removes higher order counts

Because of the fact that each channel spans a wavelength range greater than $2 \times \lambda_{min}$, the Spectrometer has little or no free spectral range. Light from wavelengths shorter than λ_{min} in each channel is collected by the mirror, and may contribute to spectra in inside orders other than the principal order, $n = -1$. Contamination from higher order light is sometimes a significant problem in continuum spectra, where higher order counts form an underlying continuum, rather than discrete features.

The task **euvtools.suborder** uses the EUVE spectra and reference data for higher-order effective areas to calculate the number of counts due to all possible higher orders in each bin. The higher order contributions to shorter wavelengths are calculated first. They are then subtracted from the spectrum, and the subtracted part of the spectrum is used to find the higher order intensities at longer wavelengths with a "bootstrapping" technique.

suborder uses the exposure time in the card `EXPTIME` and first-order effective area[3] to convert the subtracted count spectrum to flux units before creating the output images. The end results are corrected spectra for each of the spectrometer channels input, which can be written to `equispec` type spectral images in flux units (photons/cm^2/sec/Å). **suborder** does not provide any facility to merge overlapping parts of the spectra, or to join the spectra from the different channels[4].

[3]The effective areas for the first two inside orders in each spectrometer are plotted in the *EUVE Guest Observer Program Handbook*. The data for all measured orders are available from the CEA *ftp* site.

[4]**suborder** replaces part of the functionality of the task **euvcombine**, which is now in the **euv.obsolete** package.

> **Will running suborder improve my spectrum?**
>
> When used with version **1.9** or later of the EUVE reference data on smooth continuum spectra, **suborder** can produce relatively good order-subtracted spectra with reasonable agreement at the join points. However there are still some inconsistencies in the effective area functions in regions where the bandpasses overlap. In contrast, **suborder** does NOT work well near emission features. Line profiles in the higher-order spectra are not well characterized, and so **suborder** effectively constructs scaled versions of first order features. These often do not match the shapes predicted by ray tracing the higher orders, and may be asymmetrical. When higher orders calculated near such features are subtracted, spurious features can be generated at the wings. The same problem arises in attempts to subtract higher orders from continuum spectra with absorption features. It is usually only worthwhile to run **suborder** on spectra from bright sources with fairly smooth continua; they are the most affected by higher order light, and are corrected the most successfully.

A sample parameter set for **suborder** is shown in figure 3-12. Higher order counts will be subtraced from three EUVE spectra of the white dwarf GD153.

suborder can be run interactively, allowing the user to view each higher-order contribution as it is calculated. After viewing the display, the user can vary many of the task parameters before any subtractions are made. Each order is then calculated in turn, allowing the user to elect whether to perform each subtraction. Refer to the IRAF help page for complete information on the parameters and how higher-order contributions are calculated.

Notes on running **suborder**

- When operating on spectra such as those made by **euvextract** the task looks for a `BANDID` card in the header. If the spectrum band is not identified in some header card, band 1 is assumed.
- The spectra from all three channels should be included as input, unless the user only wants a subtracted spectrum for the SW channel. The task will not abort if only the medium and long spectra are input, but counts in both the short and medium wavelength channels should be available to predict any higher order counts in the longer channels.
- The parameter `putout` is used to determine whether the results of the task should be saved in one-dimensional spectral images. If `putout` is "no", the task will exit without writing, and the subtracted spectra are discarded. This feature saves cleaning up after trial runs.
- The bin widths used to rebin the spectra are given in Å units, and can be smaller than the existing spectral bins.
- All wavelength values must be given in Å. Users may wish to plot each spectrum in **splot** first to find the desired limits.

```
eu> lpar suborder

     sw_in = "sw_spec"       sw input spectrum
     mw_in = "mw_spec"       mw input spectrum
     lw_in = "lw_spec"       lw input spectrum
    sw_out = "GD153_swsub"   sw output spectrum
    mw_out = "GD153_mwsub"   mw output spectrum
    lw_out = "GD153_lwsub"   lw output spectrum
   (putout = yes)            out put results
 (sw_delta = 0.1)            sw delta
 (mw_delta = 0.1)            mw delta
 (lw_delta = 0.1)            lw delta
   (sw_low = 75.)            sw low range
  (sw_high = 180.)           sw high range
   (mw_low = 145.)           mw low range
  (mw_high = 365.)           mw high range
   (lw_low = 300.)           lw low range
  (lw_high = 735.)           lw high range
    (sw_mw = 160.)           sw mw join point
    (mw_lw = 340.)           mw lw join point
  (refdata = "reference/detector") reference file
  (exptime = "exptime")      exposer time imheader
 (graphics = "stdgraph")     graphics
     (mode = "ql")
```

Figure 3-12: Parameter file for starting the task **suborder** in interactive mode.

- Set `lw_high` or `sw_low` *inside* the spectrometer bandpass limits, as shown in figure 3-12, rather than use the default values. This is necessary because including bins from too near the detector edges, which may be under- or over-subtracted, can produce pathological results. The output spectra will then have unsubtracted portions at each end which can be trimmed or masked off.
- The join points `sw_mw` and `mw_lw` are used to specify which spectrum is used to calculate higher order fluxes in regions where the bandpasses overlap. The join points should also be well into the longer wavelength spectral range, and inside the upper limit of the shorter range, to ensure that no over-bright or over-subtracted edges are used for order calculations.

Starting suborder

When started, **suborder** first rebins the input spectra, and divides the counts in each bin by the bin width specified in `sw_delt`, `mw_delt`, or `lw_delt`. The initial display shows all spectra plotted as counts/Å vs. Å. All calculations are carried out on the binned fluxes, and the results are converted to counts/Å for display. Figure 3-13 shows the initial display for a set of GD153 spectra.

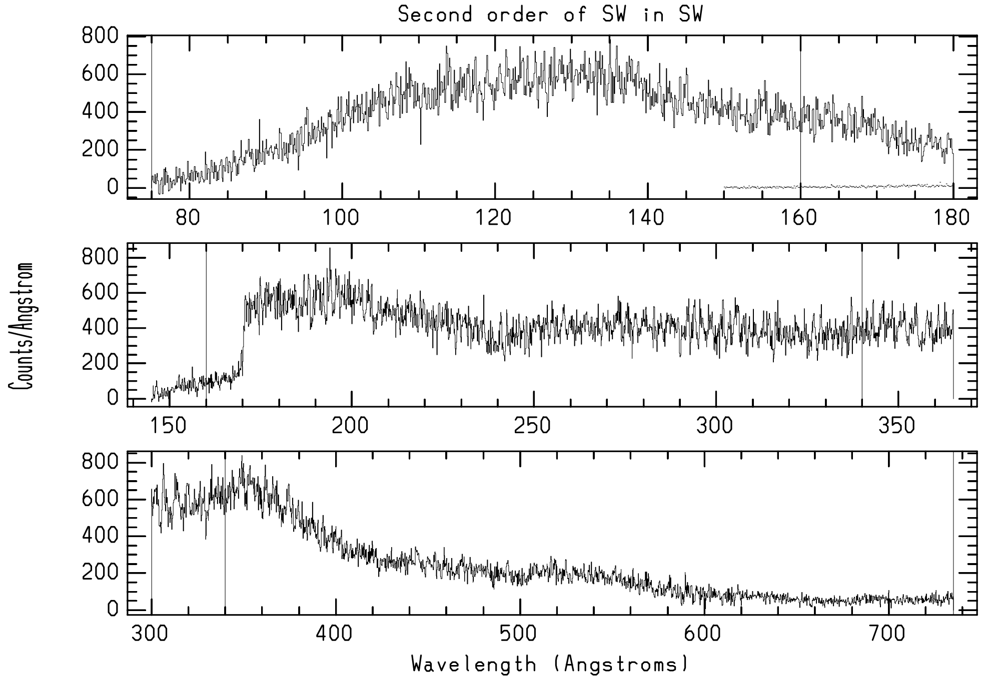

Figure 3-13: Start-up display of GD153 spectra in **suborder**, showing input spectra and the 2nd order contribution from light at 75–150 Å to the short wavelength spectrum.

The task starts by assuming that the spectrum from `sw_low` to [`sw_low` $\times$ 2] is free of any higher-order contributions. Although the short wavelength (SW) spectrometer can produce orders -2, -3, etc. for wavelengths shortward of 75 Å, their contribution is assumed to be negligible because the throughput of the Lexan filter at these wavelengths is very low. Therefore, if the spectrum limits are as in figure 3-12, the 2nd order contribution to SW from SW begins at $2 \times 75 = 150$ Å.

IRAF shifts into interactive cursor mode when the task starts. The command line shifts to the graphics screen, and the task waits for the first command. Any adjustments to the spectrum limits or the join points must be made at this point, before the first subtraction. To change the spectrum limits, place the cursor at the desired lower limit within the spectrum you want to change, and type the command `z`. The task will record the X-position as the new lower limit and you will be prompted to enter a new upper limit by repeating the process. To change the join points, position the cursor in the same way and type the command `j`. The 2nd order contribution of SW to SW will be recalculated with each change of parameters.

Running in interactive mode

Once in interactive mode, the user views each calculated order overplotted on the appropriate spectrum, and elects to subtract the contribution, or to move on to the next calculation without subtracting, or to undo the last subtraction. Order contributions are subtracted by giving the `:subtract` command after each new calculation is displayed. Subtraction of any single order can be bypassed in interactive mode by typing the `c` command (for "continue"). The next contribution is then calculated immediately from the current versions of the subtracted spectra. The `u` command undoes the last subtraction, returning the spectrum its former state.

The task proceeds from short to long wavelengths, calculating and applying each successive order for each channel. After the initial subtraction of 2nd order from SW, there should be no more higher orders in this channel, as $3 \times$ `sw_low` $= 225$ is outside thw SW bandpass. Contributions of SW light to second order in the medium wavelength (MW) are then calculated from the entire spectrum between `sw_low` and `sw_mw`, the first join point.

The process continues, calculating all higher orders in each Spectrometer channel from the shorter ones until some order misses the bandpass entirely. All contributions from wavelengths longer than `sw_mw` and shorter than `mw_lw` are calculated from the MW spectrum. Contributions from wavelengths longer than `mw_lw` are calculated from the long wavelength (LW) spectrum. All subtractions are performed on the entire spectrum between the low and high limits, including regions between join points and limits, like that between `sw_mw` and `sw_high`.

When all possible orders have been computed and subtracted, the final subtracted spectra are written out in flux units, obtained by dividing the subtracted counts in the display by `exptime` and the first order effective area. The user can command the task to complete the sequence without further reviews (or choices) at any time by typing `:auto`. One can quit from the task at any time by typing `q`. Spectra are recorded if `putout` is "yes".

Manipulating the display

The following commands can be given at any time, to change some aspect of the display. They involve no subtractions or order calculations. Complete definitions of the commands can be displayed in the text window by typing a `?`.

- To view one spectrum at a time, position the cursor over the spectrum you want and type `l`.
- The `a` command restores all the spectra to the display.
- After any single subtraction, the `s` command will overplot the spectrum before the last subtraction on the subtracted result, allowing the user to view the effects of the subtraction. Type `s` again to remove the overplot from the display. The `s` command is actually a toggle which is always turned "off" (unsubtracted spectrum not displayed) when **suborder** is started.
- When the difference between subtracted and unsubtracted spectra is small, it may be necessary to enlarge part of the plot with the standard IRAF `window` commands to see the difference. Typing `w`(indow) `a` will restore the plot under the cursor to its default range and scale. The Y axis will be autoscaled when `subtract` or `undo` are issued, but other window commands remain in effect.

- The `d` and `f` commands are also toggles; they turn the display of the current subtracted spectrum and the next higher-order contribution off and on, respectively. They are both "on" at startup.

The user should always be aware of the status of the display and the task algorithm, to avoid being confused about the results of the subtractions. For example, a combination of `d` and `s` make it possible to create a display in which the version of the spectrum *before the last subtraction* is plotted along with the next higher-order contribution to be subtracted. Then the next `:subtract` command produces a spectrum showing the affects of the last *two* subtractions.

If you do become confused, using the undo (`u`) and various window commands to back up and compare the subtracted and unsubtracted states may be helpful.

Example 3.3 : Subtracting higher-order contributions with suborder

The spectra from an observation of hot DA white dwarf GD153 are to have higher order contributions subtracted. The GO sets up the parameter file as in figure 3-12, and types `:go` in **epar** to start the task. After viewing the initial display of the second order contribution to the short wavelength spectrum shown back in figure 3-13, the GO types `:subtract` to perform subtraction of second order contributions of SW light from the SW spectrum.

The SW spectrum has no third order contributions from wavelengths with $\lambda > 75$ Å, as $75 \times 3 >$ `sw_high`, and shorter wavelengths were assumed to be negligible. So the next contribution calculated is the second order contribution from SW light to the MW spectrum. The contribution is quite small, but the GO subtracts this order also, as the subtraction adds relatively little noise and the compounded effects of several orders may be significant. The same is repeated for the third and fourth orders of SW in MW, and then the second order MW light in the MW spectrum, which begins at $2 \times$ `sw_mw` $= 320$ Å.

After this is subtracted, the next contribution is from second order SW light to the LW spectrum. The GO now displays only the LW plot by using the `l` command. This eliminates waiting for the (now static) SW and MW spectra to be redrawn after each order. The second order contribution of SW in LW is small, and all at wavelengths lower than the join point `mw_lw`, so the subtracted bins would not affect later order calculations. The GO decides to skip over the subtraction by typing `c` (for "continue").

The third and fourth order contributions of SW light in LW are very small. After subtracting the fourth order, the GO tries to assess the difference in the spectrum. He types `s` to overplot the previous spectrum, then a series of window commands enlarge a small region of the spectrum. There is no detectible difference in the spectrum, so it does not seem that the fourth order (or higher orders) is significant. The GO types `w a` to return to the autoscaled axes, and skips over all of the rest of the SW orders ($n = 5,6,7$, and 8) in LW using `c`. The next significant contribution is the second order of MW in the LW spectrum, shown in figure 3-14.

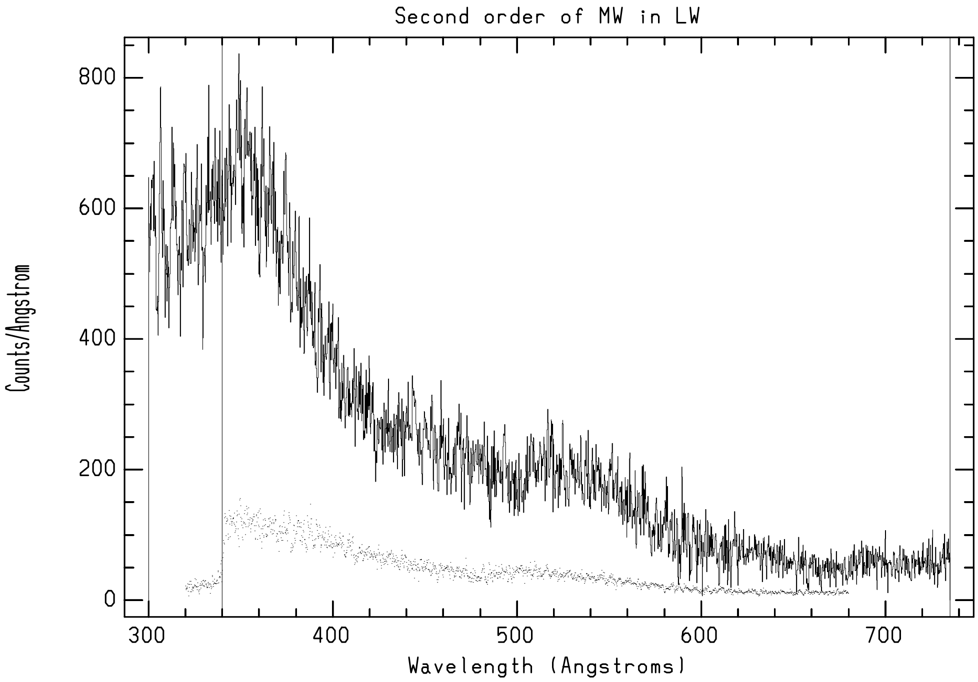

Figure 3-14: Second order MW in LW from **suborder**. The vertical line at 340 Å is the join point between the MW and LW spectra. The spectrum is seen before subtraction of the -2nd order counts.

Near the `mw_lw` join point, it is about 1 count in 6, and so its subtraction makes a visible difference in the spectrum height and shape. Figure 3-15 shows a close-up view of the subtracted result overplotted with the spectrum before subtraction. The plot was made using the following commands:

cursor position	**command**	**effect**
(any,any)	`s`	overplots spectrum before subtraction
(550,350)	`w j`	sets left edge of window
(560,363)	`w k`	sets right edge of window
(any,450)	`w b`	sets bottom edge of window
(any,790)	`w t`	sets top edge of window
(any,any)	`:.snap`	sends screen dump to printer

One can see that the shape of the spectrum is preserved after the subtraction– very little additional noise is added.

The third order of MW in LW is also significant, but does not contribute until 480 Å, or $3 \times$ sw_mw. The fourth order results in some increased over-subtraction between 600 and 700 Å. Subtracting the higher orders from LW itself polishes off most of the signal longward of about 640 Å. The final spectra are divided by the first order effective areas before being written to the output images. They are shown in figure 3-16, after smoothing with a boxcar of $\approx \Delta\lambda$.

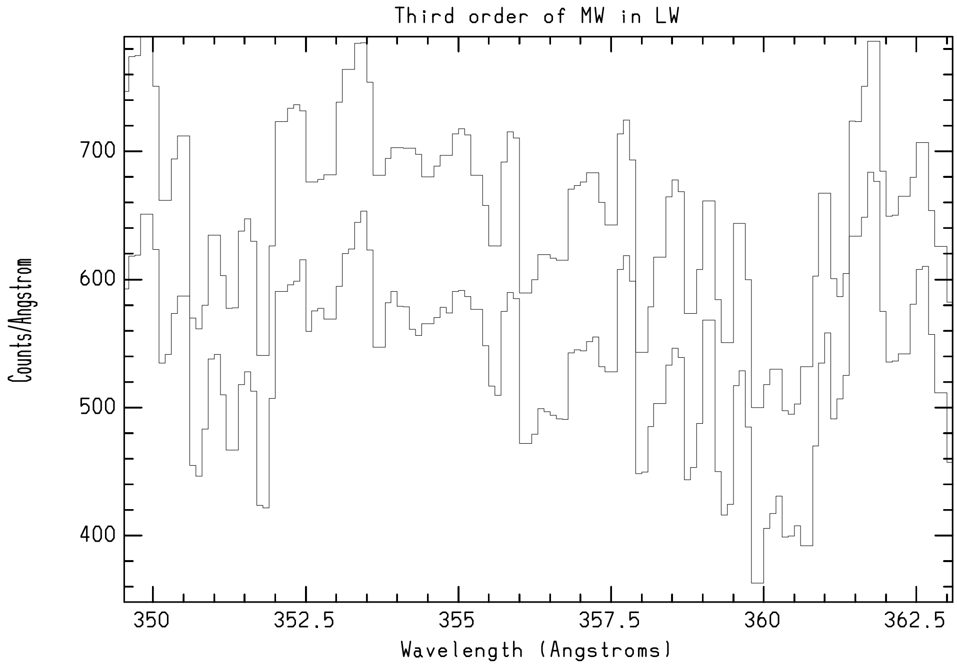

Figure 3-15: Enlarged section of LW spectrum. Top line is the spectrum before subtraction of second order MW light; bottom is after. **suborder** is ready to subtract 3rd order.

Although there has been some increase in the noise levels, especially near the ends, the spectra are now cleaned of higher order contamination, and fluxed.

3.2.3 EUVHCLEAN strips obsolete header cards from spectra

The task **euvtools.euvhclean** is supplied to remove obsolete header cards which are commonly written into EUVE spectra when they are created. The task has only one input parameter: the name of the image to be cleaned. It has a predefined list of header keywords, and removes cards matching a certain template. **euvhclean** is normally used from the command line:

```
eu> euvhcl GD153_swsp
```

Cards removed include the macros inherited from parent QPOE files. These macros occupy a large block of space in the header file and are no longer useful once the QPOE has been binned into an image. Other cards removed are also inherited from the parent QPOE file, and their functions may be obsolete or redundant with other header cards in the image. See the **euvhclean** help page for a full description of the keyword templates applied.

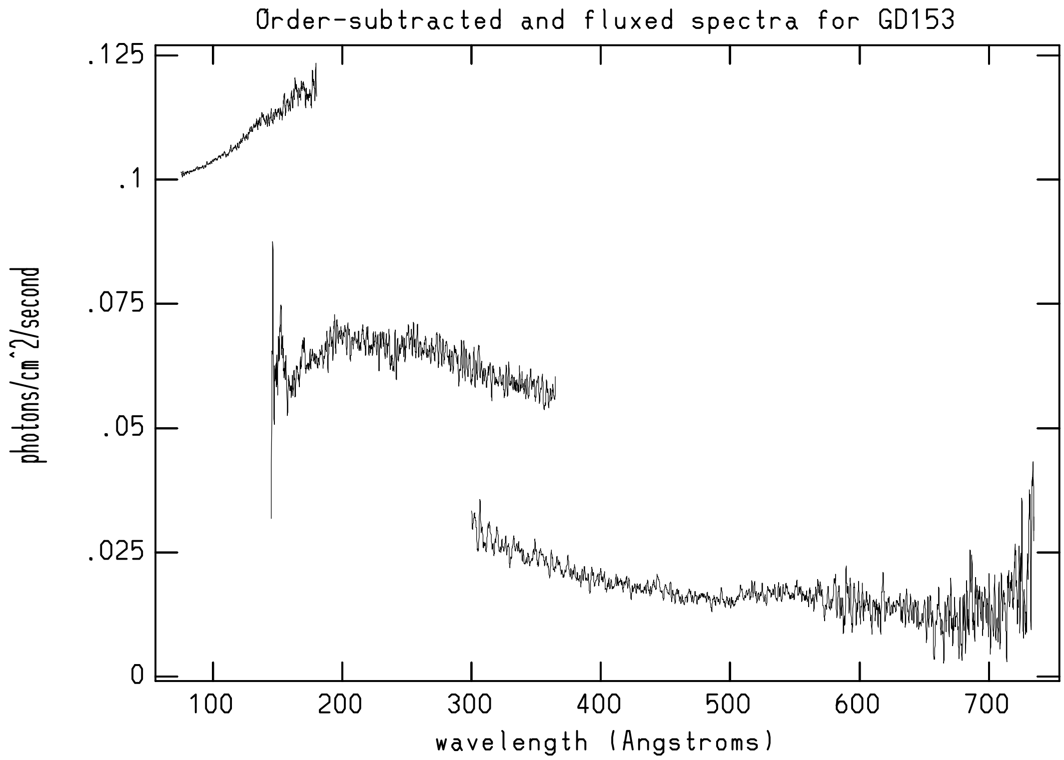

Figure 3-16: Final output from **suborder**. The subtracted spectra have units of photons/cm^2/sec/Å. The spectra have been boxcar smoothed by about 1 resolution element, and are offset in intensity.

3.2.4 SPECPRINT prints spectrum and error band

The task **euvtools.specprint** is a tool for creating ASCII listings of extracted spectra. A spectral image with multiple bands is input, the spectrum and error band numbers are given through parameters, and the task writes the wavelength of each bin, the spectrum counts, and the subtraction errors to STDOUT. Figure 3-17 shows a sample parameter file for printing one of the EGO Center extracted spectra.

```
eu> lpar specprint


    input = "lw_spec"       Input image file
(specband = 1)              Image band with spectrum
 (errband = 2)              Image band with spectrum errors
  (format = "%13.3f %g")    Output formats
    (mode = "al")
```

Figure 3-17: Parameter file for **specprint**

The output from **specprint** can be easily redirected to a file:

```
eu> specprint lw_spec > lwspec.txt
```

The default values of the parameters `specband` and `errband` are correct for the extracted spectrum and error bands of spectra produced with **euvextract**. The parameter `format` may also be used to specify the format of the output:

```
eu> specprint mw_spec specb=1 errb=2 format=%9.3f,%g,%g > mwspec.txt
```

specprint should not be run with either `specband` or `errband` set to "INDEF".