Gaussian fitting for the Hinode/EIS mission

Peter Young (GMU/NRL), Version 1.2, 16-Apr-2010

A basic requirement for any emission line spectroscopy mission is a set of software suitable for fitting Gaussians to lines in order to derive intensity, velocity and line widths. This document describes a set of IDL routines available for the Hinode/EIS mission that are intended to cover most situations encountered by EIS observers.

Since EIS yields spatially-resolved data that often have sufficiently high signal-to-noise to yield good line profiles in individual spatial pixels, then a common requirement is for a fitting routine that automatically goes through a raster and fits Gaussians at each pixel. We refer to this as Case 1.

The second scenario is when a user wants to obtain a spectrum for a particular spatial feature that spans more than one pixel. Examples include bright points, active regions, flares, quiet Sun and coronal hole. In this case the user wants a single spectrum that has been averaged over many pixels. The situation is complicated with EIS as spatial offsets and mis-alignments within the instrument lead to the spatial feature occurring at different locations in Y on the detector depending on the wavelength. The Gaussian fitting routine in this case will be a manual one (since there is only one spectrum) and should allow the user to sequentially go through the spectrum fitting lines individually or through multi-Gaussian fits. This scenario is referred to as Case 2.

The routines described in the following sections deal with the following situations:

Case 1

- both single and multi-Gaussian fitting (automatic)
- correcting for the slit tilt and orbit variation
- selecting sub-regions within a wavelength window for fitting
- specifying initial parameters and parameter value limits

Case 2

- correcting for spatial offsets as a function of wavelength
- correcting for slit tilt and orbit variation
- choose a pixel mask to identify feature within a raster
- a manual single and multi-Gaussian fitting routine

A key starting point for the Gaussian fitting routines is the extraction of an EIS data window into an IDL structure using the routine 'eis_getwindata.pro'. We often refer to the 'windata' structure in the sections below, which is the output from this routine. There are a number of routines for manipulating the windata structures and these are described in the Appendix.

A separate document is available with worked examples for the Gaussian fitting routines.

Case 1: automatic Gaussian fitting

The basic fitting routine here is 'eis_auto_fit.pro' which deals with both single and multiple Gaussian fitting, however the latter will be dealt with separately in the sections below. There is a varying level of complexity in how the fits are performed, with the velocity requiring the most consideration due to the lack of an absolute calibration for EIS as well as instrumental effects that change the position of the line centroid on the detector. Since an EIS raster may contain many thousands of spatial pixels it is important for the user to be able to assess the quality of the line fits. This is done with the routine 'eis_fit_viewer.pro'.

At the most basic level, an automatic single Gaussian fit can be performed with, e.g.,

```
IDL> wd=eis_getwindata(l1name,195.12)
IDL> eis_auto_fit, wd, fitdata
IDL> eis_fit_viewer, wd, fitdata
```

A more complete (and recommended) set of commands is:

```
IDL> wd=eis_getwindata(l1name,195.12)
IDL> eis_wave_corr, l1name, offset
IDL> eis_wvl_select, wd, offset, wvl_select
IDL> eis_auto_fit, wd, fitdata, offset=offset, wvl_select=wvl_select
IDL> eis_fit_viewer, wd, fitdata
```

that allow the fitting routine to take account of the wavelength offsets due to the slit tilt and orbit correction, and a restricted wavelength range (to prevent nearby emission lines affecting the fit). After performing the fit the user may want to use the fit data to derive an improved wavelength offset array to yield more accurate velocity values. All of these topics are dealt with in the sections below.

1 Single Gaussian fitting

1.1 Dealing with the orbit variation and slit tilt

The centroids of emission lines on the EIS detectors move during the satellite orbit of 100 mins by about 2 pixels. In addition, the tilts of the EIS slits mean that the centroid of a given line can vary by up to 2.5 pixels along the slit. (More information about these instrumental effects can be found on the EIS wiki.)

The most obvious impact of these effects is on velocity maps derived from Gaussian fits, which clearly show the strong orbit variation and the shift of velocity with Y-position. The user can see this by doing:

```
IDL> wd=eis_getwindata(l1name,195.12)
IDL> eis_auto_fit, wd, fitdata
IDL> eis_fit_viewer, wd, fitdata
```

A more subtle effect can occur at the line-fitting stage. Consider the case of an emission line at 200 Å that is isolated in the spectrum in the region 199.7 to 200.3 Å, however there are strong lines either side of this region. To fit a single Gaussian to the line at 200 Å ideally we want to restrict the wavelength range to 199.7-200.3 Å. However the default wavelength

scale for EIS data is fixed and takes no account of the instrumental effects. Thus although a wavelength range of 199.7-200.3 Å is specified the real wavelength range may be 199.6-200.2 Å, which would include part of the strong emission line that we are trying to avoid. The solution is to assign a realistic wavelength vector for every spatial pixel that accounts for the slit tilt and orbit variation effects.

The default wavelength vector for every EIS window structure is stored in 'wd.wvl' ('wd' is the windata structure produced by 'eis_getwindata'). We thus want to create a new wavelength vector for each spatial pixel (x,y):

IDL> wvl[x,y] = wd.wvl – offset[x,y]

such that the emission lines appear (approximately) at the same wavelength position at each pixel. As well as preventing the problem described above, the wavelength correction will also lead to a velocity map without the artifacts due to the instrumental effects.

1.2 Creating the offset array

The simplest (and quickest) way to create an offset array is to use the routine eis_wave_corr:

IDL> eis_wave_corr, filename, offset

This extracts the Fe XII 195.12 line from the FITS file and takes the moments of the profile across the raster in order to work out the orbital variation of the centroid. The standard slit tilts for the 1" and 2" slits are then applied to yield the final offset array.

The user should treat this offset array as a *first approximation* to the instrumental effects, and so the velocity map that is derived from the following fitting process should not be considered definitive. Methods for deriving a more accurate offset array are described later.

1.3 Restricting the wavelength range

EIS studies are usually designed so that a wavelength window contains a single emission line and so the user will simply use the entire wavelength range of the window for fitting. Sometimes, however, a study designer may deliberately have chosen a wide wavelength range to encompass two or more lines, or a line may be very close to another line in the spectrum. In these cases a user can choose either to do a multi-Gaussian fit (see later), or he/she can choose to only fit a subset of the spectral range containing the emission line and the nearby continuum. This latter step is performed with the routine eis_wvl_select.pro:

IDL> eis_wvl_select, windata, wvl_select, offset=offset

An IDL widget will appear with two graphics window (see Figure 1). The left panel shows an image derived from the data window (summed over wavelength). Clicking on any position in the image produces a spectrum from that spatial pixel in the right-hand panel. Initially each wavelength pixel in the spectrum has a *. This indicates that the pixel will be included in the fit performed by eis_auto_fit. By holding-and-dragging the left mouse button the user can draw a rubber-band box on the spectrum. All wavelength pixels within the X-range of this box (Y-range is not important) will be de-selected, and the * symbols will disappear. This means that those pixels will not be included in the fit. Figure 1 shows the case of a large wavelength window where the pixels from around 263.5 Å and higher have been de-selected. Therefore only the line around 263.0 Å will actually be fit by 'eis_auto_fit'.

After selecting a wavelength range, the user can try clicking on different spatial pixels to check if any lines pop up at different locations that could interfere with the fit. When the user is happy with the fit region, clicking 'EXIT' will return the session to the IDL prompt and the selected wavelength region(s) will be stored in the structure 'wvl_select'.

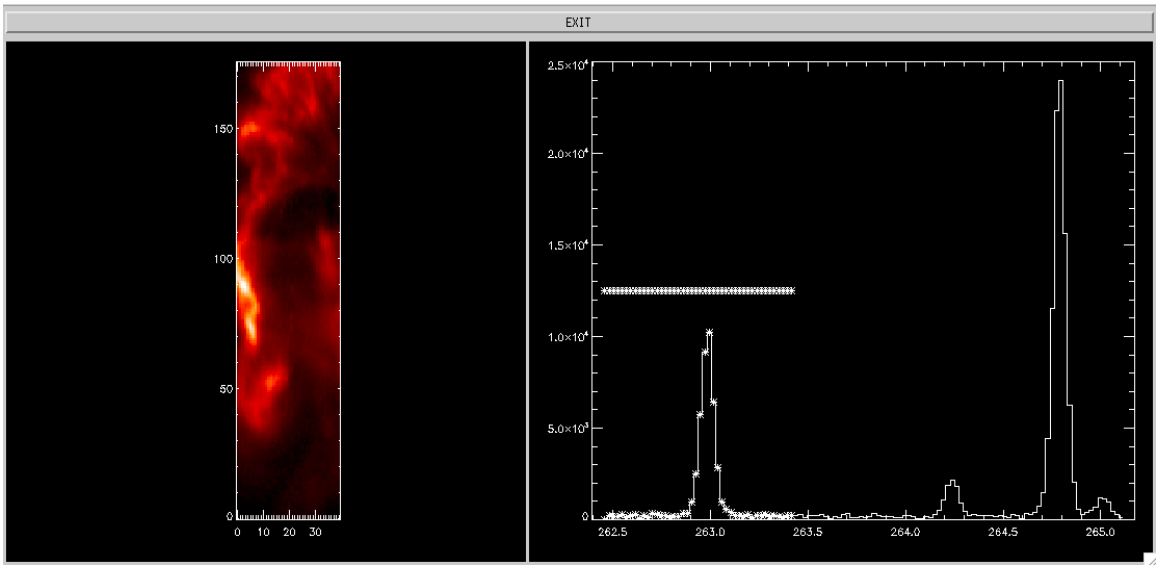

Figure 1 A screen grab of the eis_wvl_select widget. The left panel shows the image derived from the raster, and right panel shows the spectrum from the user-selected spatial pixel. Pixels to be included in the fit are denoted by * symbols on the spectrum.

1.4 Performing the fits

With 'offset' and 'wvl_select' defined the automatic fitting routine can be used:

IDL> eis_auto_fit, windata, fitdata, wvl_select=wvl_select, offset=offset

fitdata is an IDL structure. For information about how to extract intensity, velocity and line width arrays from this structure, see the routine *eis_get_fitdata* that is discussed later. A full description of the contents of *fitdata* is given in the header of the *eis_auto_fit* routine.

One important tag within 'fitdata' is 'refwvl'. This gives the reference wavelength against which velocities are computed. The call to 'eis_auto_fit' sets 'refwvl' to be the average of the centroids across the whole raster.

1.5 Viewing the fits

The widget-based routine 'eis_fit_viewer' allows the user to study the intensity, velocity and line width maps, and to assess the quality of the fits at individual pixels. It also shows histograms of the fit quantities across the raster. The IDL call is:

IDL> eis_fit_viewer, windata, fitdata

The top three windows in the GUI contain the intensity map, velocity map and line width map. Text boxes allow the user to specify upper and lower limits for scaling the images.

With the 'Zoom' option selected, you can zoom into any of the three images by clicking-and-dragging the mouse over the windows to create a rubber-band box that selects a region.

With the 'Pixel' option selected, you can click on any pixel in any of the images, and the line profile for that pixel will be shown in the bottom-left graphics window. In the bottom-right window a histogram plot of the selected line parameter (intensity, velocity, width) is shown for the spatial region that is being displayed in the upper windows.

1.6 Re-assessing the tilt and orbit corrections (with fitted line)

After the line fits have been performed it is possible to use the fits themselves to make an improved characterization of the orbital variation of the line centroid. A common scenario is for active regions where the user may choose to use a patch of quiet Sun at the top or bottom of the slit to derive the correction.

The correction is performed using the routine 'eis_update_fitdata' which is called as, e.g.,

```
IDL> newfitdata=eis_update_fitdata(fitdata,yrange=yrange)
```

where 'yrange' is used to indicate the range of Y-pixels that you want to use as the reference. E.g., if you believe a good patch of quiet Sun occurs between Y-pixels 20 to 44 then you would specify yrange=[20,44]. If 'yrange' is not specified then the full height of the raster will be used. A single pixel can be specified by doing yrange=20, for example.

'eis_update_fitdata' removes the original wavelength correction ('offset') used by 'eis_auto_fit' and computes a new 'offset' array by using the standard slit tilt value and the orbit variation obtained from the Y-pixels specified by 'yrange'. The output 'newfitdata' will contain the new 'offset' array. In addition the tag 'refwvl' will be updated to be the average wavelength in the Y-pixel region specified by 'yrange'. When the measured centroids are converted to velocities using the 'refwvl' wavelength it means that the average velocity of the pixels in 'yrange' will be zero.

A useful optional output from 'eis_update_fitdata' is the 2D array 'offset'. This is the same size as the original raster size and is suitable for using as an input into the fitting routines as a wavelength offset, i.e., a substitute for the 'offset' array generated by 'eis_wave_corr'. Examples that make use of this array are given in the separate Example document.

2 Multi-Gaussian fitting

The main difference over single Gaussian fitting is in the specification of initial parameters for each of the Gaussians. For the single Gaussian case, the initial parameters are automatically calculated by 'eis_auto_fit' by finding the pixel with the maximum intensity in the spectrum, and using this pixel as a first guess for the amplitude and centroid of the emission line. Since emission line widths are dominated by the instrumental width, an initial width of 70 mÅ is assumed for all lines. The initial guess for the spectrum background is assumed to be the minimum intensity value in the spectrum being fitted.

For the multi-Gaussian case it is necessary for the user to manually specify the approximate location and amplitude of each of the lines, and this is performed with the widget routine 'eis_fit_template'. With this extra step, the sequence of commands to perform multi-Gaussian fitting is:

```
IDL> wd=eis_getwindata(l1name,195.12)
IDL> eis_wave_corr, l1name, offset
IDL> eis_fit_template, wd, template, offset=offset
```

```
IDL> eis_wvl_select, wd, wvl_select, offset=offset
IDL> eis_auto_fit, wd, fitdata, offset=offset, template=template,
     wvl_select=wvl_select
IDL> eis_fit_viewer, wd, fitdata
```

2.1 Creating a fit template

'eis_fit_template' is a widget-based routine and the sequence of operations for using it is are as follows:

1. The left-hand graphic window shows an image derived from 'windata'. Draw a rubber-band box (hold-and-drag the left mouse button) to select a sub-region within the image.
2. A spectrum will appear in the right-hand graphics window. It has been averaged over the spatial region selected in Step 1.
3. Click on the 'Choose lines' button underneath the right-hand graphics window.
4. With single clicks of the left mouse button, click at the approximate locations of the peak of each line you want to include in the fit. A thick vertical line will appear with each click.
5. When you're finished selecting lines, click on the 'End selection' button underneath the right-hand graphics window.
6. You can now go back to the image in the left-hand window and select different spatial regions to see if your original region gives a typical spectrum, or if there are additional lines not apparent in the original region.
7. When you're happy with your line selection, click on the 'Exit' button at the top of the widget.

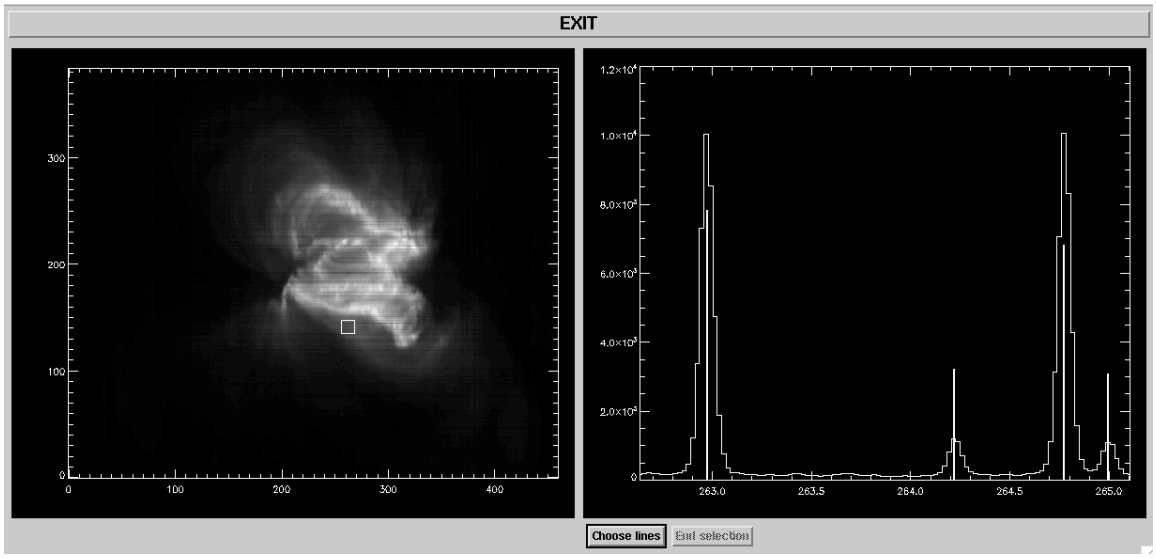

Figure 2 Example of the 'eis_fit_template' widget, showing the EIS image on the left and the spectrum from the selected region (denoted by white square on image) on the right. On the spectrum four vertical lines denote the initial parameters selected by the user.

Some recommendations for using 'eis_fit_template':

1. Over-estimating the peaks of weak lines and under-estimating the peaks of strong lines helps yield better fits.
2. When choosing spatial regions, don't make them too large. Around 10x10 pixels is good. Also making the box bigger in Y is better than making it bigger in X.
3. Try to avoid choosing the brightest region for selecting the template – try to use a weak-to-average brightness region as this will be a better representation of the raster as a whole.

Examples for using eis_fit_template are presented in the accompanying Examples document.

The user-created template is returned to IDL as an IDL structure that can then be input to eis_auto_fit. The template can be saved in the form of a text file by doing:

IDL> eis_write_template, 'template.txt', template

This file can be edited manually be the user, and then read back into an IDL structure by doing:

IDL> template=eis_read_template('template.txt')

This may be useful if the user plans to use a template for many data-sets.

2.2 Viewing the multi-Gaussian fits (eis_fit_viewer)

To view the multi-Gaussian fits produced by 'eis_auto_fit', simply call 'eis_fit_viewer' in the same manner as before:

IDL> eis_fit_viewer, wd, fitdata

The widget shows one significant difference over the single Gaussian version: below the 'Unzoom' button a set of widget buttons is shown giving the 'refwvl' values for each line of the multi-Gaussian fit. By clicking on different lines you will see the images change, giving the intensity, velocity and width for the selected line.

For the spectrum plot (bottom left graphic window), there are also some plot options to make viewing the fits easier. In particular:

- For 'X-range options', clicking on 'Selected line' shows the spectrum for +/- 0.20 Å either side of the selected emission line.
- For 'Y-range options', clicking on 'Selected line' changes the Y-range to better display the selected line.

2.3 Re-assessing the orbit correction

The orbit correction can be re-assessed using the line fit data in the same manner as for single Gaussian fits, i.e., with the routine 'eis_update_fitdata'. An important difference, however, is that the emission line to be used to determine the orbit correction is specified with the line= keyword, i.e.,

IDL> newfitdata=eis_update_fitdata(fitdata,yrange=yrange,line=line)

The average value of the selected line centroid over the region specified by 'yrange' is used to update the value of the 'refwvl' tag within 'fitdata'. For example, suppose the selected line originally had a 'refwvl' value of 262.00. The result of the call to 'eis_update_fitdata' is to update this value to 262.15. If there are two other lines in the fit with 'refwvl' values of 263.50 and 264.20, then these will be updated to 263.65 and 264.35.

2.4 Advanced feature: setting parameter limits

Parameter limits can be useful for preventing bad fits. For example EIS emission lines typically have widths of around 60-80 mÅ. A line cannot have a width of < 50 mÅ as it would be narrower than the instrumental width, while widths of > 200 mÅ are highly unusual. One can thus specify

that line widths should lie within the range 50-200 mÅ. Bounds on each of the three emission line parameters (peak, centroid and width) can be specified through the 'template' structure.

'template' has the tag 'lines' which contains a number of tags that apply to each of the Gaussians. The parameter limits are stored in the following tags (for the example of Gaussian 0):

```
IDL> print,template.lines[0].peak_lim
0.00000     -100.000
IDL> print,template.lines[0].cen_lim
256.534      256.834
IDL> print,template.lines[0].wid_lim
0.0200000    0.0900000
```

These are the default parameter limits set by 'eis_fit_template'. For each parameter there are two numbers: the lower and upper limits, respectively. If a limit is set to -100 then it means the parameter is not limited. Therefore the line peak must be ≥ 0, but there is no upper limit. The centroid must lie between 256.534 and 256.834 Å, i.e., ± 0.15 Å of the wavelength selected by the user when using 'eis_fit_template'. The Gaussian width must lie between 0.020 and 0.090 Å. Note that the Gaussian width is related to the FWHM by the factor 2.35 so this range corresponds to a FWHM range of 47-212 mÅ.

By simply editing the numbers in 'template' one can adjust the parameter limits. E.g., suppose you want to remove the upper limit on line widths for Gaussian 4, then you do:

```
IDl> template.lines[4].wid_lim[1]=-100.0
```

If the template structure has been written to a text file using 'eis_write_template' (see Section 2.1) then the limits can be adjusted by opening the template file in a text editor and modifying the lines:

```
Allowed range for wavelength:     256.534     256.834
Allowed range for peak:             0.000    -100.000
Allowed range for width:            0.020       0.090
```

Note that a fixed format of '(f12.3)' is used for these numbers. The new template file can be read into IDL using:

```
IDL> template=eis_read_template('template.txt')
```

2.5 What happens if a parameter limit is reached by the fitting routine?

If a parameter limit is reached at one spatial location, then the 1σ error on that parameter is set to zero and the parameter value is set to the limit. All other parameters will be optimized as normal. *eis_auto_fit* prints a summary of all parameter limits that were reached during the fitting. If there are a lot of such cases, then the user has the following options:

1. Modify the parameter limits in the *template* structure.
2. Bin the data using *eis_bin_windata* and re-do the fit.
3. Omit weak lines from the fit using *eis_wvl_select*.

Generally noisy data are the cause of the parameter limits being reached and so options 2 or 3 should be used, however there may be cases where the parameter limits can be tweaked, e.g., if two lines are close together in the spectrum.

2.6 Advanced feature: tying parameters

In addition to prescribing parameter limits, the user can also tie the parameters of one emission line to another. Three cases are considered:

1. A line has a fixed intensity ratio relative to another line (for example a branching ratio) so the user forces the two lines to have the same width, and their peaks to have a fixed ratio.
2. Two lines are emitted from the same ion so the widths can be set to have the same value.
3. Two lines have a fixed wavelength separation from each other.

The user should generally not use this feature unless the use of fully independent line parameters fails to yield satisfactory results. The accompanying Examples document shows how parameter tying is used for the Fe XII λ195.12+λ195.18, and Fe XII λ203.72+Fe XIII λ203.82 blends.

Parameter tying is implemented through the template structure (Section 2.1). The tags of the 'lines' structure of 'template' include the following:

IDL> help, template.lines[0]

```
PEAK_TIE         INT              -1
PEAK_TIE_VAL     FLOAT          -100.000
CEN_TIE          INT              -1
CEN_TIE_VAL      FLOAT          -100.000
WID_TIE          INT              -1
```

These are the parameters that control the parameter tying. The values of -1 for 'peak_tie', 'cen_tie' and 'wid_tie' indicate that parameter-tying is switched off. Suppose Gaussian 1 is emitted from the same ion as Gaussian 0, and that atomic theory predicts that Gaussian 1 is 0.3 times the strength of Gaussian 0, and occurs at a wavelength 0.32 Å longer than Gaussian 0. These facts can be implemented in the template structure by doing

IDL> template.lines[1].peak_tie=0
IDL> template.lines[1].cen_tie_val=0.30
IDL> template.lines[1].cen_tie=0
IDL> template.lines[1].peak_tie_val=0.32
IDL> template.lines[1].wid_tie=0

The '_tie' values are set to 0 indicating that the three parameters of Gaussian 1 are tied to the parameters of Gaussian 0. The value 'peak_tie_val' is set to the ratio of Gaussian 0 to Gaussian 1, and 'cen_tie_value' is set to the wavelength offset of the two lines. Note that there is no 'wid_tie_val' as the only option available is to set two lines to have the same width.

The above commands set the parameter tie values for the current session. If you want to set them for multiple data-sets, then you can manually edit the template text file. First write out the template structure to a text file:

IDL> eis_write_template,'template.txt',template

and then open this file in a text editor. For each Gaussian in the file there will be the following text:

```
Parameter tying:
```

```
Amplitude:  -1    -100.000
Centroid:   -1    -100.000
Width:      -1
```

where the default values can be manually replaced with the new values. Note that the format is fixed to '(i3,f12.3)'.

To see how the parameter tie values are used by 'eis_auto_fit', try doing:

IDL> expr=eis_fit_function(template)

the resulting string gives the complete fit function that will be used by 'eis_auto_fit' in the call to the minimization routine 'mpfitexpr'. The p[0], p[1], etc., are the free parameters for the fit. Each Gaussian has a corresponding 'gauss_sg' function, and the background is given by either a 'line_sg' function, or a single parameter, depending on if a linear or flat background is specified.

3 Extracting parameters from the fitdata structure

The structure *fitdata* stores the fit parameters in an array of size (*nparam, nx, ny*) under the tag *aa*. It is possible to pick out the peak, centroid and width arrays from this array, however it is easier to use the routine *eis_get_fitdata*. E.g.,

IDL> cen=eis_get_fitdata(fitdata, /cen, line=line)

which extracts the centroids into a 2D array (the optional input line= is used to specify which emission line, in the case of multi-Gaussian fits, is to be extracted). The associated error array can be extracted by using the optional input 'error='. E.g.,

IDL> cen=eis_get_fitdata(fitdata,/cen,line=line,error=cenerr)

Velocities are not stored explicitly in *fitdata*, and so *eis_get_fitdata* is recommended for extracting these:

IDL> vel=eis_get_fitdata(fitdata,/vel,line=line)

Note that the reference wavelength stored in *fitdata.refwvl* is used to compute the velocities.

4 Technical details

This section gives additional technical details of the software that may be of interest to some users.

4.1 Specifying the fit function

The MPFIT procedures used for the line fitting require a text string that specifies the fit function. For the present case this function is a linear combination of Gaussians and a straight line for the background. For the case of multi-Gaussian fits the function string is generated by the routine *eis_fit_function*:

IDL> expr=eis_fit_function(template)

The Gaussian function is *gauss_sg* and the linear function is *line_sg*. These functions are stored in the $SSW/idl/gen/fitting directory. The parameter tying described in Section 2.6 is implemented through the function string.

Case 2: Gaussian fitting for spatially-averaged spectra

Creating spatially-averaged spectra is, at first glance, straightforward: simply average the spectra over the spatial region, and average the errors in quadrature. The situation is complicated for EIS, however, due to the spatial offsets within the EIS instrument which means that a spatial region selected from the Fe XII 195.12 line will not necessarily correspond to the same region, in terms of Y-pixels and even exposure numbers, with another line.

Described below is a set of IDL routines that account for the spatial offsets to create an averaged spectrum that correctly represents the observed spatial structure. The basic steps and corresponding IDL routines are:

1. Create an image for identifying a pixel mask (eis_make_image.pro)
2. Create the pixel mask that defines the spatial region (eis_pixel_mask.pro)
3. Create a 1D spectrum derived from the spatial pixel mask (eis_mask_spectrum.pro)
4. Fit the emission lines (spec_gauss_eis.pro)

We use the following data set as an example:

eis_l0_20061209_113031.fits

which is a 256x256 raster that takes 15 spectral windows with 40s exposures. It is recommended that the user download and calibrate this data with eis_prep:

IDL> eis_prep, l0name, /save, /default, /retain

where l0name is the name of the level-0 FITS file. A level-1 FITS file will be created together with an associated error file.

1 Step 1: make an image

If you already know which wavelength you want to use to identify the spatial structure that you're interested in, do:

IDL> eis_make_image, l1name, 185.21, im185

where l1name is the name of the level-1 FITS file, and 185.21 is the wavelength I'm using for this example. The 2D array im185 contains an image derived from the Fe VIII 185.21 emission line (7 wavelength pixels centered on the 185.21 wavelength have been averaged to generate the image).

Now plot the image using

IDL> plot_image,sigrange(im185)

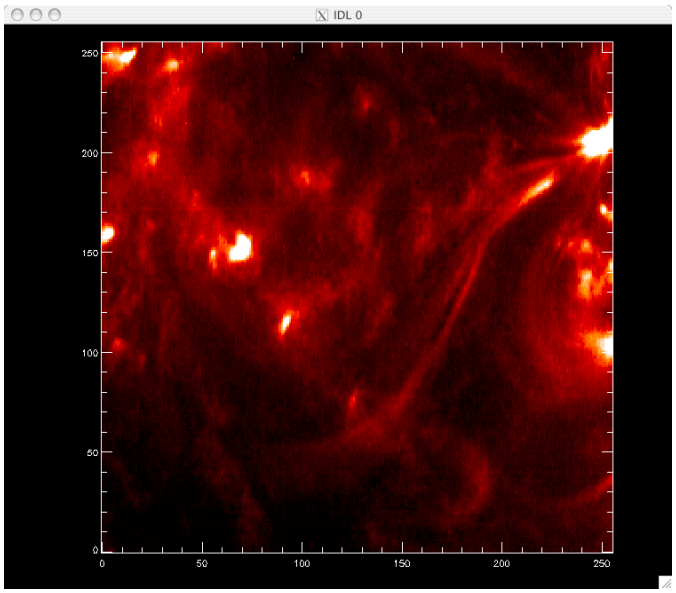

Figure 3. Fe VIII 185.21 image from the 12-Dec-2006 data set.

Since Fe VIII is a cool line (log T = 5.8) then the image consists of a number of fairly small brightenings with a few narrow loop structures.

If you want to browse the data set to choose a different emission line, try using eis_raster_browser:

IDL> eis_raster_browser, l1name

2 Step 2: Create a pixel mask

A pixel mask is an array of same size as the image described earlier but containing only 1's and 0's. Pixels marked with a 1 indicate that they will be used to generate the averaged spectrum.

To create a pixel mask from your image, do

IDL> eis_pixel_mask, im185, mask185, 185.21, 1

where im185 is the image created in the previous section, mask185 will contain the output mask, 185.21 is the wavelength of the line you've specified, and 1 is the size of the slit (should be either 1 for the 1" slit, or 2 for the 2" slit).

Upon calling eis_pixel_mask, you will see the image plotted again (with color table 5) and in the IDL window you will see a menu:

IDL> eis_pixel_mask,im185,mask185,185.21,1

*** CHOOSE A MODE ***
left: polygon mode
middle: painting mode
right: exit

The mouse is used to create a pixel mask by clicking on the image. *You need to use a 3 button mouse for the routine to work!* There are two options for creating the mask:

1. Polygon mode. Click on a number of points in the image and, when complete, the routine will join the dots to create a polygon. The polygon will be filled, and the enclosed pixels will be set to 1 in the pixel mask.
2. Painting mode. By holding down the left mouse button, you can 'paint' over the image, selecting pixels as you go. Selected pixels can be removed by clicking with the middle mouse button.

Generally for small spatial features you will use the Painting mode to accurately select the pixels you need, while Polygon mode is used for choosing large spatial regions.

While clicking on the image, make sure to always check back to the IDL window to see what mode you are currently in. Note that the right mouse button is always used to exit out of a mode.

For our example we are going to choose a small brightening in the image so it is necessary to zoom in on the region of interest:

IDL> eis_pixel_mask,im185,mask185,185.21,1,xsc=[50,100],ysc=[120,170], col=255

which displays X-pixel range 50 to 100 and Y-pixel range 120 to 170. Note that col=255 is used to set the color of the over-plotted pixels to white.

Use the Painting mode to select the pixels. In Figure 4 the selected pixels are displayed in white.

If you want to reset the mask, then simply exit out of the routine and do:

IDL> mask185=0

The Appendix to this document explains how the Polygon option in eis_pixel_mask works.

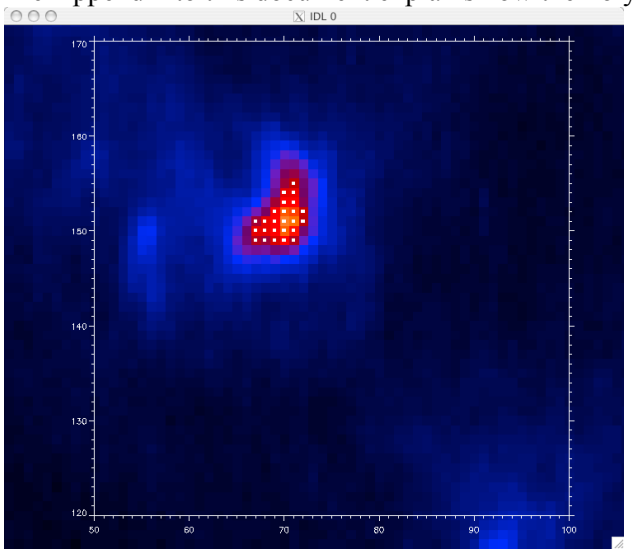

Figure 4. Fe VIII image showing the selected pixels (small white squares).

2.1 Viewing the pixel map in different wavelengths

It is useful to check how the pixel mask looks for other wavelengths. To create the mask for, e.g., Si VII 275.35, do

IDL> mask275=eis_adjust_mask(mask185,275.35)

eis_adjust_mask is an important routine as it takes into account the various spatial offsets to move the selected pixels to spatially match the original wavelength.

To give an indication of the shifts involved in the present case, do:

IDL> plot_image,mask185.image+mask275.image,xra=[50,100],yra=[120,170]

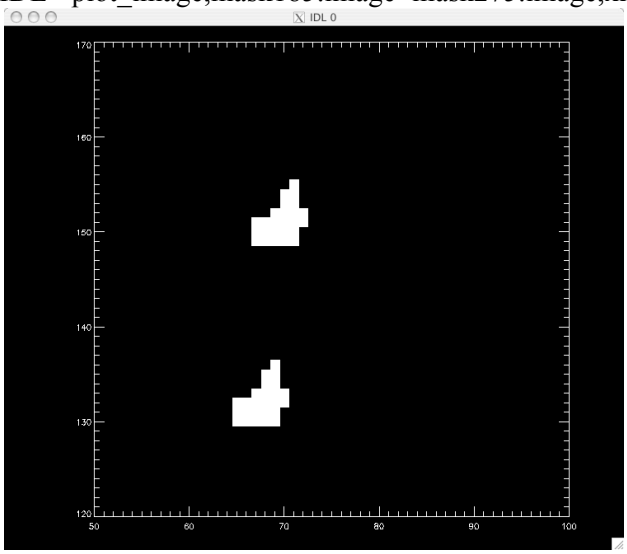

Figure 5. Pixel masks for Fe VIII 185 (upper) and Si VII 275 (lower).

The upper set of pixels are those from the original Fe VIII 185.21 mask, while the lower set of pixels are those from the Si VII 275.35 mask. The latter is seen to be shifted in both X and Y.

Now since Si VII and Fe VIII are formed at the same temperature, then we are able to check how accurately the derived Si VII mask sits on the brightening seen in the Si VII image (remember that mask275 has been derived from the mask chosen in the Fe VIII image).

We first create a 275 image:

IDL> eis_make_image, l1name, 275.35, im275

and then do:

IDL> eis_pixel_mask,im275,mask275,275.35,1,col=255,xsc=[50,100],ysc=[120,170]

which gives the image shown in Figure 6. The match is fairly good although seems to be off by about 1 pixel in the Y-direction. Since the shift of the pixel mask is done in integer pixel numbers, then this is within the uncertainty of the method taking into account also the uncertainty in the measured EIS spatial offsets.

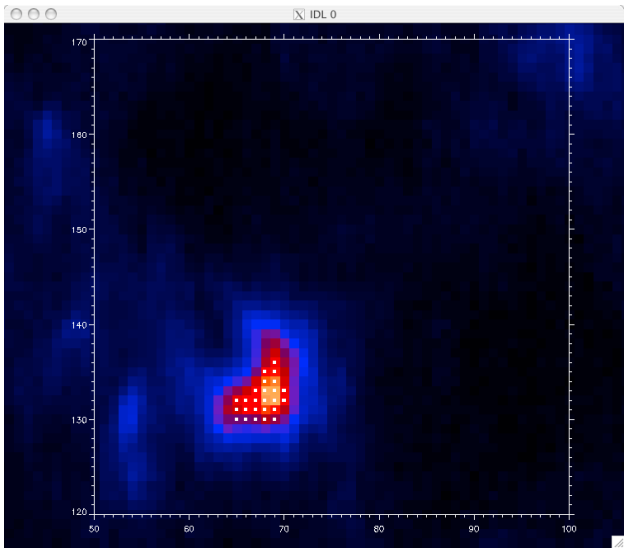

Figure 6. Si VII 275.35 image with pixel mask overplotted.

3 Deriving the averaged spectrum

Once you are happy with your chosen pixel mask, an EIS spectrum averaged over the spatial region can be derived by doing:

IDL> eis_mask_spectrum, l1name, mask185, swspec=swspec, lwspec=lwspec

swspec and lwspec are both structures with the following tags:

```
IDL> help,swspec,/str
** Structure <1539eaa4>, 5 tags, length=36868, data length=36866, refs=1:
   WVL             DOUBLE    Array[2048]
   INT             FLOAT     Array[2048]
   ERR             FLOAT     Array[2048]
   QUAL            INT       Array[2048]
   QUAL_MAX        INT           25
```

You will see that the spectrum is defined for the full size of the CCD (2048 pixels). You can view it by doing:

IDL> plot,swspec.wvl,swspec.int,psym=10,/xsty

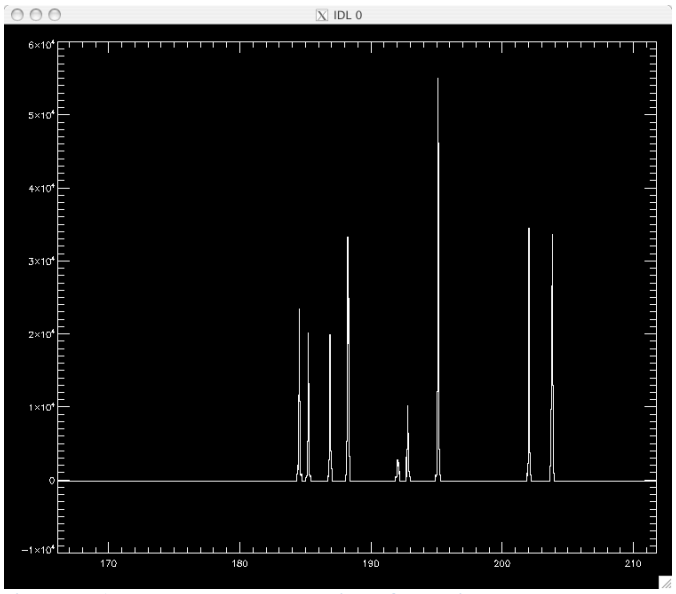

Figure 7. Averaged spectrum derived from pixel mask.

You'll see only a few emission lines that correspond to the wavelength windows of the EIS raster. All other wavelengths are set to an intensity value of -100.

Note that the number of pixels averaged over to generate the spectrum is stored in the tag QUAL_MAX. The significance of the QUAL tag is discussed in the separate tutorial on SPEC_GAUSS_WIDGET, the Gaussian fitting routine.

3.1 Tilt and orbit corrections

The slit tilt and orbit variation will mean that averaging spectra over a spatial area will lead to emission lines being artificially broadened due to the varying positions of the line centroids on the detector. For small spatial areas, say 10x10 pixels, this will be a small affect but for large areas the broadening may be significant. This is particularly so for the 2" slit which has a larger tilt than the 1" slit.

To perform the slit tilt and orbit corrections, first do:

```
IDL> eis_wave_corr, l1name, offset
```

which derives a quick wavelength offset array using the Fe XII 195.12 line. Although it is possible to derive a more accurate offset array, this approximation is generally good enough for the spectrum averaging performed here.

The mask spectrum can then be derived using:

```
IDL> eis_mask_spectrum,l1name,mask185,offset=offset,/refill,swspec=swspec,
   lwspec=lwspec
```

Basically, at each spatial pixel the spectra are interpolated onto a reference spectrum scale using the offsets stored in 'offset'. This aligns the spectra so they can be added. The routine that performs the shifts to the spectra is 'eis_shift_spec.pro' where more details can be found. It is recommended to use the /refill keyword to replace missing data with 'realistic' intensity and error values. This is because a consequence of the interpolation method is that the number of missing

pixels is essentially doubled. More details about warm pixels and the /refill option are available on the EIS wiki.

4 Step 4: Fitting Gaussians

Gaussians can be fit to the output spectrum using the routine SPEC_GAUSS_EIS, which is called as:

IDL> spec_gauss_eis, swspec

it is actually a wrapper to a general purpose Gaussian fitting routine called SPEC_GAUSS_WIDGET present in the /gen branch of Solarsoft. A tutorial explaining how to use 'spec_gauss_widget' is available in Solarsoft at:

$SSW/gen/idl/fitting/spec_gauss_widget_tutorial.pdf

The tutorial is also available through the EIS wiki.

5 Appendix: the Polygon option in eis_pixel_mask

Start up eis_pixel_mask on the whole Fe VIII 185 image:

IDL> eis_pixel_mask, im185, mask185, 185.21, 1, col=255

and click somewhere on the image with the left mouse button – this puts the routine into Polygon mode.

I will choose an approximately rectangular region in the bottom left of the image by clicking three times with the left mouse button. You'll see that after the first button press, subsequent ones will show a line joining the newly added point to the previous. Figure 8 shows the result after 3 button presses.

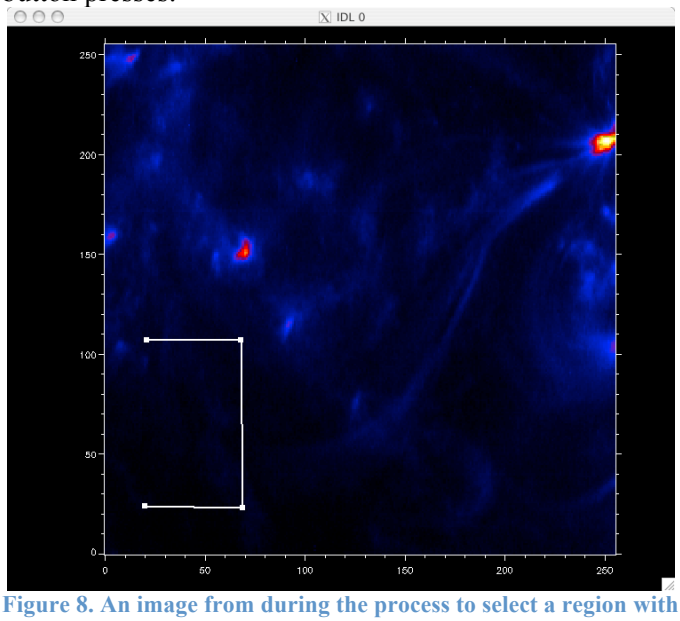

Figure 8. An image from during the process to select a region with the Polygon option in eis_pixel_mask.

To ‘close’ the rectangle, simply click on the RIGHT mouse button, which exits the Polygon mode and joins up the last two dots. Figure 9 shows the result where the selected pixels form a rectangular block in the image.

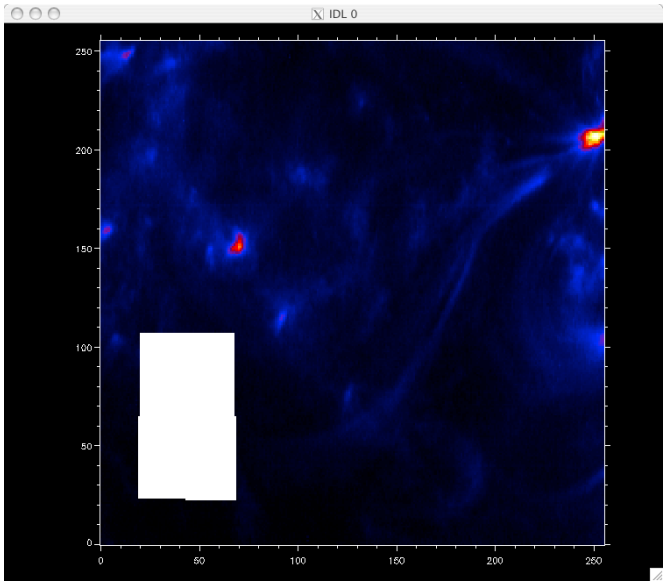

Figure 9. The selected spatial region after using the Polygon option in eis_pixel_mask is shown.

To fine-tune the selected block of pixels, the Painting mode can be used.

Appendix

1 eis_getwindata operations

1.1 Concatenating two windata structures in the wavelength dimension (eis_join_windata)

An EIS study may contain two wavelengths that are very close to each other or even directly adjacent. In some circumstances it may be useful for fitting to concatenate the two windows such that the fitted wavelength region will extend across both windows. This can be done as follows:

```
IDL> wd1=eis_getwindata(l1name, wvl1)
IDL> wd2=eis_getwindata(l1name, wvl2)
IDL> wd=eis_join_windata(wd1,wd2)
```

1.2 Spatial binning of windata structures (eis_bin_windata)

To increase signal-to-noise for weak lines it may be useful to perform spatial binning. This is performed with eis_bin_windata, e.g.,

```
IDL> wd=eis_getwindata(l1name, wvl)
IDL> wdnew=eis_bin_windata(wd,xbin=3,ybin=3)
```

By default the routine starts binning from pixel (0,0), i.e., the bottom left corner of the spatial image. The keywords xstart= and ystart= can be used to change the starting pixel.

If you are using the 'offset' array to characterize the wavelength offsets, then this can also be input to eis_bin_windata:

```
IDL> wdnew=eis_bin_windata(wd,xbin=3,ybin=3,offset=offset)
```

and it will be binned in the same way as 'wd'. Note that 'offset' will be overwritten with the new array.