EUV IMAGING SPECTROMETER

Hinode

EIS SOFTWARE NOTE No. 21

Version 1.2 18 February 2020

The WINDATA routines for EIS data analysis

Peter Young
NASA Goddard Space Flight Center
Greenbelt, MD 20771
U.S.A.

1. Introduction

The routine *eis_getwindata* extracts a wavelength window from an EIS data file and puts it into an IDL structure along with a set of metadata. A number of routines make use of the *windata* structure, for example the *eis_auto_fit* routine described in EIS Software Note No. 16. In addition there are a number of routines that perform operations on the *windata* structure, and these routines are described in the present Software Note.

Routine	Purpose
eis_join_windata	Join two windatas together in wavelength direction.
eis_bin_windata	Spatially bin windata in X and/or Y directions.
eis_trim_windata	Reduce the wavelength range of a windata.
eis_combine_sitstare_windata	Join two sit-and-stare windatas in time direction.
eis_shift_spec	Interpolate all spatial pixels onto common wavelength scale.
eis_fix_windata	Remove data anomalies from windata.
eis_sat_windata	Set saturated data to missing when using /refill.

2. eis_join_windata (concatenate two windata structures)

An EIS study may contain two wavelength windows that are very close to each other or even directly adjacent. In some circumstances it may be useful to concatenate the two windows in the wavelength direction. For example, if the spectrum in a window is congested and it is not easy to identify a good background region for performing line fitting. Two wavelength windows can be concatenated by doing:

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

3. eis_bin_windata (spatial binning)

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 (see Software Note No. 16), 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.

4. eis_trim_windata (trim wavelength range)

This routine is intended for when *eis_auto_fit* is used for full CCD data-sets. The usage is:

```
IDL> wd=eis_getwindata(l1name,195.12)
```

```
IDL> wdnew=eis_trim_windata(wd,[194.5,196])
```

The initial *wd* structure contains data arrays that cover the complete CCD range of 1024 pixels, while *wdnew* contains arrays that only span the wavelength range 194.5 to 196.0 Å.

The new structure, *wdnew*, can be used with the *eis_auto_fit* suite of routines in the same way as other *windata* structures.

5. eis_combine_sitstare_windata (combining multiple sit-and-stare data-sets)

Often a long sit-and-stare sequence will be split over multiple FITS files and it is useful to combine the individual data windows into one large window. This can be done as follows:

```
IDL> wd1=eis_getwindata(fname1,wvl)
IDL> wd2=eis_getwindata(fname2,wvl)
IDL> wdnew=eis_combine_sitstare_windata(wd1,wd2)
```

This new windata structure can then be processed with *eis_auto_fit* in the normal way.

6. eis_shift_spec (interpolate onto common wavelength scale)

The slit tilt and spectrum drift mean that the wavelength scale for EIS rasters changes across the raster, both in X and Y. The offset relative to the default wavelength scale (stored in *windata.wvl*) is stored in *windata.wave_corr*. The routine *eis_auto_fit* operates by fitting each spatial pixel by taking the intensity from *windata.int* and using the wavelength vector that has been corrected for the *wave_corr* value at that pixel.

An alternative way of doing things is to interpolate the intensity array to make each spatial pixel have the same wavelength scale. This can be done with *eis_shift_spec*:

```
IDL> wdnew=eis_shift_spec(wd)
```

which uses the *wave_corr* wavelength offsets to interpolate the intensity spectrum at each spatial pixel. After doing this step, one can then send the new *windata* structure to *eis_auto_fit* as follows:

```
IDL> wdnew.wave_corr=0.
IDL> eis_fit_template, wdnew, template
IDL> eis_wvl_select, wdnew, wvl_select
IDL> eis_auto_fit, wdnew, fit, template=template, wvl_select=wvl_select
```

Note that it is necessary to set *wdnew.wave_corr* to zero otherwise *eis_auto_fit* will use it to perform the wavelength correction, which is not necessary after calling *eis_shift_spec*.

7. eis_fix_windata (fix data problems)

There are some data anomalies that are not identified by eis_prep, and two of them can be fixed within the windata structure using eis_fix_windata. The first is a problem where data columns are set to 2048 DN, giving anomalously bright intensities, and the second is where the exposure time for a column is zero. The call to fix these is:

```
IDL> wdnew=eis_fix_windata(wd)
```

You will see a plot, showing the average intensity as a function of column. You should see a smooth variation, but if there is a "2048 column" then it will be appear as a single spike. The routine tries to identify such spikes automatically, but if it fails then you can adjust the 'thresh' optional input. The zero exposure time columns are automatically corrected.

8. eis_sat_windata (fix saturated data problem)

The /refill option to *eis_getwindata* imputes the values of missing data, but does not differentiate between the types of missing data (warm pixels, cosmic rays, saturated pixels, etc.). For saturated data we prefer to leave these as missing, as it is not possible to guess what the real value of the intensity is for these pixels. This problem can be fixed by calling *eis_sat_windata* **instead of** *eis_getwindata*:

```
IDL> wd=eis_sat_windata(file, wvl, /refill)
```

Note that this routine requires that you have the original level-0 file in your $HINODE_DATA directory tree.

9. Pointing information

The tags 'xcen' and 'ycen' give the heliocentric coordinates of the center of the raster. These are *not* the best values to use. Please check the EIS Wiki page:

http://solarb.mssl.ucl.ac.uk:8080/eiswiki/Wiki.jsp?page=EISPointing

for more details about EIS pointing. The routine *eis_getwindata* uses the "First approximation" method listed on this webpage. To get more accurate values, you should use the routine *eis_aia_offsets* to obtain a correction that can be applied to the *eis_getwindata* values. See the Wiki page for more details.

Update history

Version 1.2, 18-Feb-2020 – added Sect. 9 (pointing); fixed formatting problem on front page.
Version 1.1, 7-Jan-2015 – some minor text edits.