A post-pipeline processing algorithm for Short Wavelength Spectra from the *ISO*[1]

G. C. Sloan[2,3], Kathleen E. Kraemer[4,5], Stephan D. Price[4],

and

Russell F. Shipman[6]

ABSTRACT

We describe in detail an algorithm to generate a database of all valid SWS full-scan spectra (2.4–45.4 μm). The algorithm is designed to process these data in as uniform a manner as possible. It consists of a series of procedures written primarily in the Interactive Data Language (IDL), and these procedures are available to the community on-line.

1. Introduction

The STARTYPE program began as a series of proposals to obtain spectra with the Short Wavelength Spectrometer (SWS) on the Infrared Space Observatory (*ISO*) with two objectives: (1) to obtain spectra of as many different classes of astronomical objects as possible and to use these as the basis for an infrared spectral classification system, and (2) to make all full-scan spectra obtained with the SWS available to the astronomy community.

---

[1]The *Infrared Space Observatory* is a project of the European Space Agency (ESA), funded by ESA Member States (especially the Principal Investigator countries: France, Germany, the Netherlands, and the United Kingdom) and with the participation of the Institute of Space and Astronautical Science (ISAS) and the National Aeronautics and Space Administration (NASA).

[2]Institute for Scientific Research, Boston College, Chestnut Hill, MA 02467

[3]Infrared Spectrograph Science Center, Cornell University, Ithaca, NY 14853-6801
sloan@isc.astro.cornell.edu

[4]Air Force Research Laboratory, Space Vehicles Directorate, 29 Randolph Rd., Hanscom AFB, MA 01731
kathleen.kraemer@hanscom.af.mil, steve.price@hanscom.af.mil

[5]Institute for Astrophysical Research, Boston University, Boston, MA 02215

[6]SRON, Space Research Institute of the Netherlands, Groningen, the Netherlands
russ@sron.rug.nl

These tasks have been accomplished by Kraemer et al. (2002,hereafter Paper I)and Sloan et al. (2003,hereafter Paper II) respectively.

As described in Paper II, the atlas of SWS spectra is available from the *ISO* Data Archive (IDA). The Uniform Record Locator (URL) on the web for this site is:

http://www.iso.vilspa.esa.es/ida/index.html

This document describes in detail the algorithm which generated the atlas, and it explains how to access both the spectra which comprise the atlas and the procedures used to generate them. The procedures which make up the algorithm are known collectively as the *swsmake* code[7]. Paper II provides a general overview of the algorithm in the main body, and in the appendix it elaborates over some of the details. The purpose of this document is to relate the steps in the algorithm to the specific procedures used to enable the potential user of the atlas to understand how the spectral files were generated, and if necessary, how to modify the algorithm to suit their own purposes. The file *swsmake.txt*, which accompanies the software, provides detailed instructions on running the various procedures.

2. A Detailed Description of the Post-Pipeline Processing

The *swsmake* algorithm begins with output from the *ISO* Off-Line Pipeline (OLP), version 10.1. A typical output file has a name with the format swaaNNNNNNNN.fit, a FITS file where the string NNNNNNNN is the Time-Designated Target (TDT) number. The TDT number uniquely identifies each observation with *ISO* and is used here to distinguish individual spectra.

2.1. Initial steps

The initial processing of the FITS data involves three programs. The initial step uses the "First Look" program *flslws* developed at the Infrared Processing and Analysis Center (IPAC) to translate the binary FITS tables of the AAR files into ASCII format. This program is available from the IPAC website in executable form for a variety of Unix-based or VMS-based operating systems[8]. IPAC requests that third parties do not distribute the First Look software, so interested users should obtain the *flslws* directly from them at:

---

[7]Names of programs and procedures are indicated with *italics*

[8]We have not had a lot of success with the Linux version.

http://www.ipac.caltech.edu/iso/firstlook/first_look.html

Two further routines read the output from *fltslws*, and sort it by segment, scan direction, and detector. The program *isotrim.c* performs the first two tasks, separating the text file produced by *fltslws* into 24 files, two for each of the 12 segments, one for each scan direction. The program *isosort.c* sorts the data in each of these 24 files by detector number and wavelength.

The remainder of the analysis is performed by routines written in Interactive Data Language (IDL) in two major steps[9]. First, the *swsmake* software processes the data to reduce each spectral segment to a single spectrum with the IDL procedure *swsmake1*. Second, the software combines the 12 separate spectral segments into a single continuous spectrum using the IDL procedure *swsmake2*.

2.2. *swsmake1*

The *swsmake1* procedure saves the un-normalized spectra as a "pws" file. The *swsmake1* procedure reads this file in, normalizes the segments to each other, trims the overlap, and saves the result as a "sws" file. Both procedures write information to a "log" file which accompanies the two data files.

The *swsmake1* procedure processes each of the 12 spectral segments by making 12 successive calls to the *segment* procedure, which performs the following steps:

The procedure *segmentread* reads the files containing data in the two scan directions and then generates a low-resolution wavelength grid (matching the wavelength spacing of the data from a single detector) for use by the following procedures.

The procedure *segmentmed* generates a median spectrum for each scan direction by first regridding all data to the low-resolution grid generated in the previous procedure and then finding the median at each wavelength. This median is converted into a pseudo-continuum by the elimination of all narrow spikes with a median filter, whether these spikes arise from glitches or actual emission lines. If the minimum flux in a segment is less than 15 Jy, or the maximum is less than 20 Jy, *segmentmed* sets a flag to use additive corrections in the rest of the algorithm instead of multiplicative.

---

[9]The IDL procedures are compatible with IDL versions 5.4 or 5.5; but have not been tested with later versions. Earlier versions of IDL will not work due to the use of new commands such as "BREAK" not previously implemented in IDL.

The procedure *segmentmrg* combines the medians from the positive and negative scan directions (illustrated by Fig. 3 in Paper II). For spectral segments from Bands 1–3, each direction receives equal weight, but in Band 4 (segment 13), the lack of any memory correction requires a different approach. Only the negative scan is used, since it occurs after the positive scan, allowing more time for recovery from any hysteresis. When both scan directions are used, the procedure may still discard portions of one scan direction if the two disagree by more than 20%. When this happens, it prints a statement to the log file that it is using variable weights to combine the medians.

The median generated by *segmentmrg* serves multiple purposes. First, it is used to generate coefficients to correct the spectrum from each detector and scan direction in a spectral segment. It also serves as the basis for filtering data and estimating uncertainties (in *segmentpass* and *segmenterr* described below).

If *segmentmed* has not flagged the spectrum as low-flux, then *segmentmod* determines the multiplicative corrections needed to force the spectrum from each detector and scan direction to the shape of the median spectrum. To preserve detailed features in the actual data, the multiplicative corrections vary as a cubic function of wavelength. If the low-flux flag has been set, then *segmentmad* is called instead of *segmentmod*. This procedure makes additive corrections, and these corrections can only vary linearly with wavelength. The coefficients generated by *segmentmod* and *segmentmad* for each detector and scan direction are saved in the log file. They indicate the reliability of a detector. (For example, detectors 34 and 36 are often problematic, which is reflected in their having significantly different coefficients compared to the other detectors.) These routines finally apply the polynomial corrections to the original unregridded data, bringing the 24 spectra into line with each other.

Next, *segmentpass* determines the acceptable range of deviations of the data from the median at each wavelength. This step is applied independently to each scan direction. The final array of acceptable ranges is produced by taking the minimum range from both scan direction at each wavelength. This ensures that an atomic line, which will appear in both scan directions survives, while a glitch which simultaneously affects all detectors in one scan direction will not.

The main procedure (*segment*) then combines the unregridded data corrected by *segmentmod* or *segmentmad* from both scan directions into a single array, which is passed to *segmentsort*. This procedure sorts the data by wavelength, rejecting data which fall outside the range appropriate for their wavelength (as defined by *segmentpass*). The procedure *segmenthigh* regrids these data onto a uniform wavelength grid based on the segment number and the speed of the observation (as detailed in Table 2). Finally, *segmenterr* estimates the uncertainty at each wavelength, using data in the low-resolution grid and re-gridding these

estimated uncertainties to the higher-resolution grid generated by *segmenthigh*.

After all the spectra from each detector and scan direction have been combined for each segment, *swsmake1* concludes by writing the results to a "pws" file. This file contains two header lines, each of 12 integers, followed by the data in 4-column format. The first header line contains the number of data for each of the 12 segments, and the second contains a number for each of the segments, set to 0 if the spectra from the detectors were combined multiplicatively and set to 1 if the corrections were additive. The remaining lines in the file contain the wavelength (in μm) in the first column, the flux (in Jy) in the second, the statistical uncertainty in the flux (in Jy) in the third, and the total uncertainty in the flux (in Jy), including normalization errors, in the fourth. Because the normalizations are performed by *swsmake2* after the "pws" file is written, in the "pws" file, these are simply set to be equal to the statistical uncertainties in column 3.

2.3. *swsmake2*

The *swsmake2* procedure completes the post-pipeline processing by performing two tasks. It first normalizes all the segments to their neighbors to eliminate discontinuities in flux, and it then trims the segments to eliminate regions of overlap between them.

The *swsmake2* procedure first determines the starting point for the segment-to-segment normalization: either Band 1E or 3D (segments 4 or 11), depending on which is brighter. The procedure then normalizes from one to the other, and then from Band 1E to Band 1A. In each case the procedure is identical. The two segments are passed to *seg2segin*, which calls *splice_in* to determine the correction factor, which *seg2segin* then applies. The correction factor is based on the average flux in the specified region of overlap. The software determines the normalization uncertainty and propagates this from segment to segment as the total statistical uncertainty (fourth column in the data files). The corrections will be multiplicative or additive for bright and faint sources, respectively; the conditions are identical to those used in *swsmake1*.

Normalization of Bands 4 and 3E to Band 3D suffers from the complication that the good data in Band 3D does not overlap the other bands. Here, *swsmake2* calls *seg2segout*, which calls *splice_out* to fit polynomials to the adjacent segment and determine the correction factor by evaluating the polynomial at a specified wavelength. The uncertainties from normalizations between Bands 1A and 3D are not propagated into Bands 3E and 4, but the errors determined through extrapolation are usually much larger.

Calls to *swsmake2* can modify the standard normalization procedure by invoking the

parameter *type*. As described by Paper II, this parameter determines (1) how the normalization of segment 13 (Band 4) to segment 11 (Band 3D) proceeds, and (2) if any wavelengths in boundary regions need to be avoided when normalizing due to the presence of emission lines. The values of *type* depend on the infrared spectral classifications defined and determined in Paper I. Table 5 in Paper II shows the correspondence of the *type* to the infrared spectral class.

Appendix B.2. in Paper II describes the normalization applied for each value of *type*. Here, we relate these to the routines called by *swsmake2*. Setting *type=0* uses the default extrapolation, while setting *type=1* modifies the extrapolation parameters used by *splice_out* to those appropriate for reddened carbon-rich dust sources, and *type=2* will use parameters appropriate to the red spectra from sources in Groups 4 and 5. Adding 10 to the *type* value (10, 11, or 12) invokes the extrapolation method used by *splice_out* appropriate to the last digit (0, 1, or 2), but with modified wavelength ranges to avoid the Br α line in the Band 1E/2A interface. The result of these higher *type* values is for *seg2segin* to call a modification of *splice_in* (known as *splice_in2*) which expects two wavelength ranges instead of the usual one so as to avoid the wavelengths affected by the emission line. For 4.PN and 4.PU sources, we use *type*=11 to account for Br α and to prevent the [O IV] emission line at 25.9 μm from affecting the Band 3D/4 normalization. Tables 4 and 5 show which values of *type* apply to the different infrared spectral classes defined in Paper I.

The normalization of segment 12 (Band 3E) to its neighbors also requires special treatment due to its unstable behavior. The procedures *seg2segtwo* and *splice_two* find an average over the wavelength ranges 26.3–27.3 μm (segment 11) and 27.7–28.7 μm (segment 13), then finds the normalization for segment 12 such that its average from 27.3 to 27.7 μm fits the mean of the averages from its neighbors.

Finally, *swsmake2* calls *cut* which truncates data outside of the given wavelength range, eliminating all overlaps between segments. The result is written to the "sws" file.

Three utilities are called by various programs. The utility *gridlo* regrids spectra from one wavelength grid to another, using a Gaussian convolution. The utility *spike* uses a median filter to identify and remove spikes. Both of these routines are called by routines within *swsmake1*. Another utility, *segmentavg* computes the average flux of a spectrum between two wavelengths. It is called by *swsmake2* to determine the starting point for the segment-to-segment normalization and by *splice_two* as part of the normalization of segment 12.

3. Spectral data files

3.1. The preliminary SWS file ("pws")

The results from *swsmake1* are saved in a file with the extension "pws". The first two lines of the "pws" file give the number of wavelength elements for each of the 12 spectral segments, followed by the flag which discriminates between multiplicative and additive corrections (*flag* set to 0 and 1 respectively). Both lines contain integers. The data follow, in four columns of floating-point numbers: wavelength (in μm), flux (in Jy), and two columns of uncertainties in the flux (both in Jy). The first column gives the statistical uncertainty, while the second includes the additional uncertainty generated by the normalization process. Since the spectral segments in the "pws" file are not yet normalized to each other, the fourth column in a "pws" file equals the third.

3.2. The final SWS file ("sws")

The *swsmake2* procedure reads the "pws" file, normalizes the spectral segments to each other, trims the spectra at the boundaries of the segments to eliminate overlaps, and writes the result in a "sws" file. The format of the "sws" file is identical to the "pws" file, except now the fourth column differs from the third where the uncertainty in the normalization is significant.

3.3. The log file

For each TDT, a log file (".log") accompanies the two spectral data files. It contains two sections written separately by *swsmake1* and *swsmake2*.

The *swsmake1* section includes details for each spectral segment, then a brief review giving the number of wavelength elements for each spectral segment and the flag value. For each segment, the log gives details about the low-resolution grid used to generate a median. Then for each scan direction (down scan first, then up scan), either *segmentmod* or *segmentmad* will print out the polynomial coefficients used to fit spectra from individual detectors to the median. These coefficients can be used to determine the correction at a specific wavelength for each detector. For additive offsets, *segmentmad* prints the detector number (0–11 for all segments), the y-intercept, and the muliplicative coefficient. For multiplicative corrections, *segmentmod* prints the detector number followed by the four coefficients for a cubic fit (in order of increasing degree). The final line for each segment repeats the flag and gives the

weight assigned to the down scan when determining the median. In addition, *segmentmrg* will generate a message at the top of the segment report if it had to vary the weights as a function of wavelength when finding the median.

The *swsmake2* section first reports how the segments were spliced together, identifying the routine used for each neighboring set, the wavelength range used, the mode, and where appropriate, the degree of the polynomial for extrapolation. The mode corresponds to the flag used in *swsmake1*; it is 0 for multiplicative corrections and 1 for additive corrections. For extrapolations, two wavelength ranges are printed, along with the wavelength (or wavelength ranges) at which the extrapolations are fitted. The degree of the polynomial usually only applies to segment 11, not segment 13. It also prints the length, flag (or mode) and normalization of each segment, along with the estimated error for this normalization. Finally *swsmake2* prints the wavelength ranges used to trim overlaps from the segments, along with the final number of wavelength elements retained for each.

REFERENCES

Kraemer, K. E., Sloan, G. C., Price, S. D., & Walker, H. J. 2002, ApJS, 140, 389 (Paper I)

Sloan, G. C., Kraemer, K. E., Price, S. D., & Shipman, R. F. 2003, ApJS, in press (Paper II)