Bandmerge User's Guide

Spitzer Heritage Archive Documentation

Version 16.2.1+, February 2011

Contents

Chapter 1. Introduction to Bandmerge

Bandmerge is a software package developed at the Spitzer Science Center. It is designed to take in ASCII tables of positions and fluxes of detected astronomical sources in 2-7 different wavebands, and write out a single table of the merged data. The tool was designed to work with source lists generated by the Spitzer Science Center's MOPEX software, although it can be "fooled" into running on other data as well.

This manual describes the software and interface in full. Bandmerge is compatible with Solaris, MacOSX and most flavors of Linux, although we only provide full user support for Linux RedHat.

1.1 Important Documentation

The following documents and webpages are recommended reading, and are referred to throughout this User's Guide:

Spitzer Data Analysis Cookbook:
http://irsa.ipac.caltech.edu/data/SPITZER/docs/dataanalysistools/cookbook/
IRAC Instrument Handbook:
http://irsa.ipac.caltech.edu/data/SPITZER/docs/irac/iracinstrumenthandbook/
MIPS Instrument Handbook:
http://irsa.ipac.caltech.edu/data/SPITZER/docs/mips/mipsinstrumenthandbook/
MOPEX main page:
http://irsa.ipac.caltech.edu/data/SPITZER/docs/dataanalysistools/tools/mopex/
Helpdesk: http://irsa.ipac.caltech.edu/data/SPITZER/docs/spitzerhelpdesk/

1.2 Getting Help

The Spitzer Helpdesk also contains a KnowledgeBase of frequently asked questions.

If you still have questions about Bandmerge then you can get help by submitting a ticket to the Helpdesk. Please help us by including the following information:

- The operating system you are running (e.g. Mac Intel OS10.5/Linux Redhat/Solaris 10);
- Which version of Bandmerge you're using;
- The AOR ID of your data (if Spitzer data);
- Attach your namelist;
- A description of the error message you saw;
- As much information as possible about the problem, including any processing steps you carried out since downloading the data from the archive.

Please note that our system does not autoreply, so if you receive a reply from us within a few minutes then please check it for further information.

1.3 Overview of Bandmerge

Bandmerge was written as a tool to combine source lists of the same field, at different wavelengths, into a single merged data table. Spitzer users frequently observe their target fields at more than one wavelength but, since each channel is reduced separately, this results in a target list for each channel, almost always with a different number of detected sources in each one. Bandmerge reads in these source lists, matches the positions of the detected sources in each of them, and writes out a single merged table with the source position and flux at each wavelength. Because images at different wavelengths can have very different pixel scales, Bandmerge reads in source lists in the galactic coordinates and projects the source positional (RA, Dec) information in the 7 different bands onto a common 2-dimensional plane with corresponding (x,y) positions. The bandmerging is performed in the fiducial plane in (x,y) positions, and the merged catalog with the (x,y) positions is then converted back to the galactic coordinates during the final output. When performing positional matching, Bandmerge takes into account the positional uncertainties, and more importantly, it can derive the systematic offsets between two different bands, correct this offset and perform more accurate source matching.

1.4 Getting Started

1.4.1 Downloading and Installing the Software

Bandmerge is available at
http://irsa.ipac.caltech.edu/data/SPITZER/docs/dataanalysistools/tools/bandmerge/. Follow the platform-specific instructions on the download page to install the software.

Once you have downloaded and unpacked the Bandmerge software, you need to set it up to run on your system. In the Bandmerge installation directory, you will find the file *bandmerge.csh*. You need to edit the first line of this file to match your installation directory as follows:

```
setenv BANDMERGE_INSTALLATION /your/installation/dir/bandmerge
```

Finally, run the setup script as follows:

```
unix% source bandmerge.csh
```

1.4.2 Setting up the directory structure

Bandmerge can be run from any directory, but wherever you choose to run it from, there must be a subdirectory called *cdf/*. This subdirectory must contain the input parameter files required to run Bandmerge (see §Chapter 4).

1.5 User Interface

Bandmerge is run on the command-line by calling a single Perl wrapper script called *bandmerge.pl*, and uses an input parameter "namelist" (*.nl) file to control the input and output files and parameters. A detailed description of the Perl script can be found in the file *README_bandmerge*, which is distributed with the software. The namelist file is a simple ASCII text file, described in §2.2. An example can be found in Appendix A the *cdf/* directory of your Bandmerge installation.

The syntax for running Bandmerge is as follows:

```
unix% bandmerge.pl –n bandmerge.nl
```

where the *bandmerge.nl* parameter namelist file must be located in a subdirectory *cdf/* of the current working directory where you are running Bandmerge (<*working_dir*>). Table 1.1 lists all of the files that must be present for Bandmerge to run. See §3 for a description of all the input files. If the full path to the input files is specified in the namelist then the <*data_dir*> can be anywhere. If the path is not specified then <*data_dir*> must be the same as <*working_dir*>.

Table 1.1: Required input files, with examples and locations

File	Example	Location
Input source lists	irac1_extract.tbl; mips24_extract.tbl	<data_dir>
Primary namelist	bandmerge.nl	<working_dir>/cdf/
Secondary namelists	bmg1.nl; bmg2.nl	<working_dir>/cdf/
Bandpair uncertainty file	bpru.tbl	<working_dir>/cdf/
Fiducial Image Frame or reference image	fif.tbl or irac1_mosaic.fits	<data_dir>

1.6 Bandmerging Overview

Bandmerging consists of four processing stages, each of which is described briefly below. For a more complete discussion, please see the technical document "Bandmerging Spitzer Point Sources" by J.W. Fowler: http://irsa.ipac.caltech.edu/data/SPITZER/docs/dataanalysistools/tools/bandmerge/bandmerge_technical.pdf.

1.6.1 Cross-Band Linkage Processing

The first bandmerging phase is *cross-band linkage*, in which every qualifying detection has an opportunity to select up to three matching qualifying detections from each band other than its own. Initial matching is done on a position-only basis; a coarse spatial window is used to select possible matches for each *seed* detection, and any *candidates* in the coarse window are subjected to the *fine position test*, which requires that the position-discrepancy chi-square have a value below a user-controlled threshold. If more than one candidate passes the fine position test, an alternative figure of merit called the *pseudo-chi-square* is evaluated, and used instead of the position chi-square value. The pseudo chi-square parameter is intended to have the same property as the chi-square in that larger values indicate less desirable matches. The computational definition of the pseudo-chi-square is currently not implemented, and the code uses the position chi-square as the pseudo-chi-square value. In each candidate band, up to three candidate detections with the lowest pseudo-chi-square values are kept for the seed by storing pointers to these detections. A counter also maintains the information about the total number of candidates that passed the match tests, and this is used in the confusion-flag information in the output file. All bands are processed as seed bands. When cross-band linkage is finished, every detection has pointers to any acceptable matches in all other bands, up to three per band.

1.6.2 Inconsistent Chain Processing

Ideally, cross-band linkage would end such that for any given detection there would be at most one pointer to another detection in each other band, with the source being pointed to having a reciprocal pointer back to the given detection. This is the most common case, in fact. But close sources, especially in regions of high density (typically low Galactic latitudes), often get tangled up with each other because of position estimation errors. In such cases, *inconsistent chains* can develop. These are chains in which a given detection's preferred match in a different band points to a different source in the given detection's band. Figure 1.1 shows an example of an inconsistent chain in band B1 (with sources S1, S3 and S5) and band B2 (with sources S2, S4 and S6). Because the match-quality parameters are required to be symmetric under interchange of seed and candidate, such chains cannot be closed loops; instead they must terminate in a reciprocal relationship. The program follows such chains until it finds the reciprocal relationship. Then breaks the chain at the point just prior to this relationship. Wherever a linkage is broken, any existing scond-choice pointer is elevated to first-choice status, and any existing third-choice pointer is elevated to second-choice status. This processing is done for all pointer linkages until only reciprocal linkages remain.

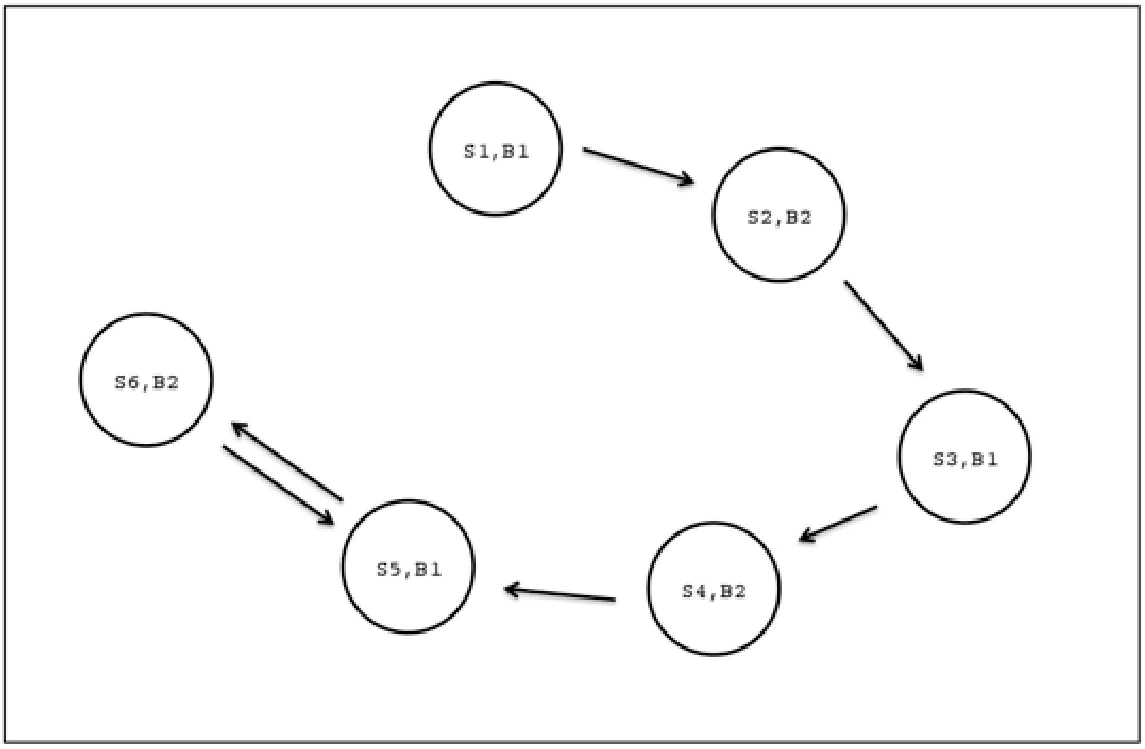


Figure 1.1: An Inconsistent Chain

1.6.3 Excess Linkage Processing

Even after all linkages have been made reciprocal, it is still possible for a different kind of inconsistency to exist in the chain. For example, a detection in one band may have a reciprocal relationship with a detection in band two, which has a reciprocal relationship with a detection in band three, which has a reciprocal relationship in band one that is not the detection in band one with which we started the chain. Thus two band-one detections are in the chain; this is called *excess linkage*. Figure 1.2 shows an example of excess linkage with two sources (S1 and S4) in band one, one source (S2) in band two, and one source (S3) in band 3. One of the two band-one detections must be removed. This is done by breaking the weakest link, i.e. the link with the largest pseudo-chi-square value. This processing is done until all chains have no excess linkages. Unlike inconsistent chain processing, breaking a link does not involve elevating second-choice or third-choice pointers, because this would necessitate a complete reprocessing for inconsistent chains.

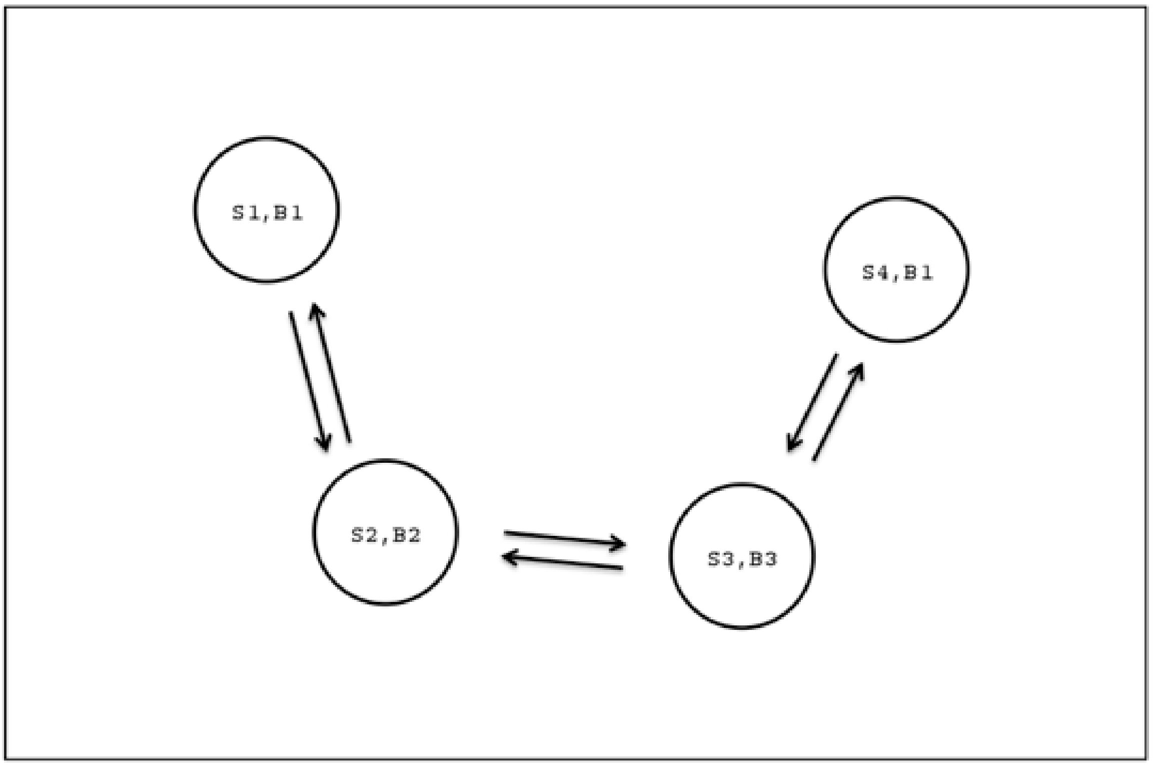


Figure 1.2: Excess Linkage

1.6.4 Linkage Rejection Processing

After excess-linkage processing, only one kind of problem can remain: not all detections in the chain are linked to each other. A simple example is: a band-one detection is linked to a band-two detection, and the latter is linked to a band-three detection; but the band-one and band-three detections are not linked to each other (see Figure 1.3; left). This is referred to as *linkage rejection*; at least one detection is linked to two others who have rejected each other. Since the mutually-rejecting detections are indirectly linked through their common detection(s), there is some indication that perhaps they *should* be allowed to link to each other, and so a looser threshold is applied. This involves scaling the threshold by the user-controlled parameter *LR_Scale_Fact* with default value = 1.5. If confusion has been diagnosed then the pseudo-chi-square threshold is scaled by this factor, otherwise the unconfused threshold is scaled. If this fails, the entire chain is searched for the detection with the fewest links and the link(s) to this source is/are broken. If more than one detection has the fewest links, then the link to the detection with the weakest link(s) (even for detections *not* involving a band with more than one detection in the chain) is/are broken. This process is illustrated in Figure 1.3 (right). The sources in bands B2, B3, B4 and B5 have links to more bands than the source in band B1; therefore the link to S1 is severed (unless its link passes the looser threshold test). Then, the processing restarts from the beginning for the detection that revealed the linkage rejection originally.

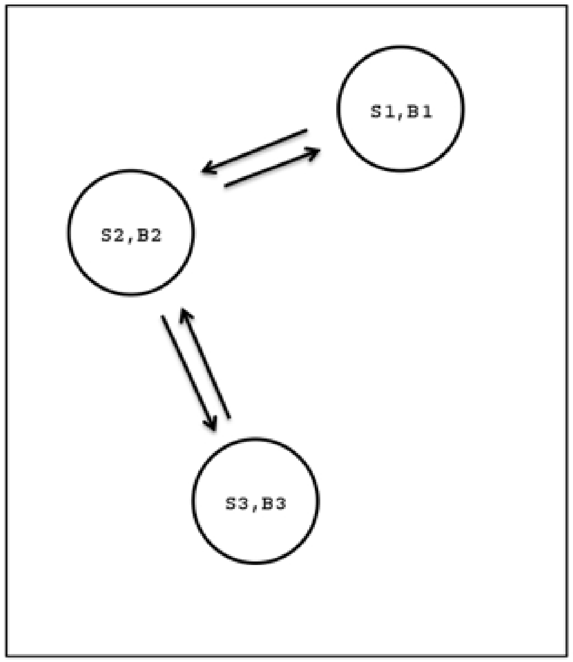
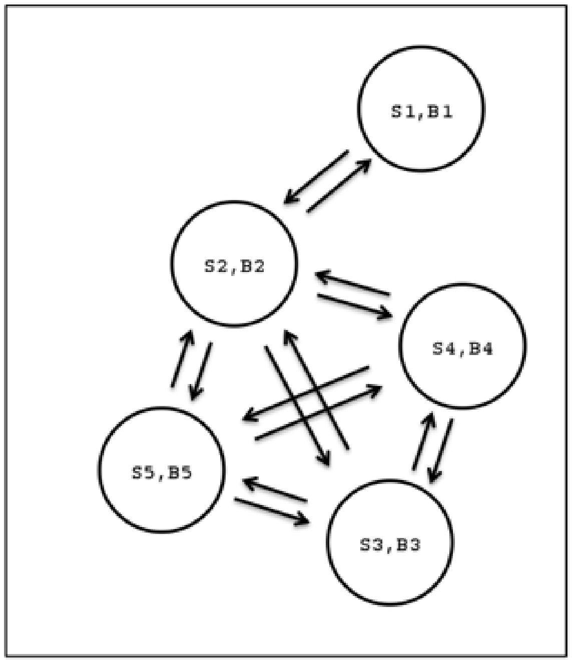


Figure 1.3: Linkage Rejection.



5-Band Linkage Rejection

1.7 Position Refinement

After all bandmerging processing is finished, each group of merged detections is processed for position refinement. This proceeds by identifying which detections are eligible to contribute their positional information to the refinement; if none are eligible, then the detection with the largest positional uncertainty (i.e. determinant of the position error covariance matrix) defines the final position. If only one detection qualifies, then its position parameters are used for the merged source. Otherwise, Gaussian refinement is performed using all qualified detections. Currently eligibility determination is not implemented, and all merged detections contribute their information to the refinement.

1.8 Statistical Summary Computation

The final processing step consists of producing a statistical summary. This is only done for Spitzer data. All multi-band (merged) sources are examined, one at a time, and only those that are completely free from confusion are considered for inclusion in the statistical summary. If the optional input column signal-to-noise ratio has been included as part or the current processing run, the SNR values for all point sources in each merged source are examined. If at least one of the point sources included in a merged source has an SNR value above the threshold for that band, this merged source is included in the statistical data. For all band-pair combinations included in the current merged source, average delta-X and delta-Y, standard deviation in X and Y, as well as chi-square in X, Y, and X/Y, are calculated. This information is binned by signal-to-noise ratio and the totals (for all SNR bins) are also calculated. If, on the other hand, SNR values are not part of the current processing run, only the "Total" category is calculated. If any of the bands in a merged source has a null SNR value then this source is *not* included in the statistical data.

1.9 Iterative Processing

One of the main challenges with bandmerging is dealing with the positional offset between the different bands (especially when taken with different detectors). Bandmerge is designed to run as an iterative process, whereby the bandmerging module and the position refinement steps are run using a crude initial estimate of the offset between bands (given in the Bandpair uncertainty file; default *bpru.tbl*), and the output of the position refinement step is then used to modify the positional information before re-running the bandmerging module. This loop of bandmerging - positional offset modification - bandmerging is defined as a single iteration.

Chapter 2. Input Files

Bandmerge requires four main inputs to run:

1) Source detection tables, containing the lists of sources to be merged. There should be one table per wavelength, and they must be in IPAC table format (see Table 2.1).

2) Parameter (namelist) files, used to control the input parameters to Bandmerge. There are two types of namelist file – a primary namelist, which controls the overall input, and a number of secondary namelists, which are used for each successive iteration of the source matching algorithm (see §2.2).

3) Bandpair uncertainty file giving the first crude guess of the offsets of the source coordinates between the different bands (see §2.3 and Table 2.2).

4) Either a Fiducial Image Frame table, defining the common grid onto which all of the source tables can be projected, or a reference FITS image containing all relevant WCS header keywords, that can be used to create this table. The *FIF.tbl* file is standard output from MOPEX (see §2.4)

2.1 Source Detection Tables

The source detection tables are the ASCII tables containing the lists of sources to be merged. There should be one source table per wavelength band. These tables must be in IPAC table format, which is the default output format of APEX, the Spitzer Science Center's Astronomical Point-source Extraction software (distributed as part of MOPEX). The table must contain the columns and header keywords as shown in Table 2.1.

The Bandmerge software will check for (and if absent will add) two additional header keywords during processing – INSTRUME and CHNLNUM, which describe the instrument used to take the data, and the instrument channel number respectively (e.g. INTRUMEN = MIPS, CHNLNUM = 2 identifies the data as being from the MIPS 70 micron detector). If they are missing from your table headers, the values for these keywords must be included in the namelist file (see §4 for more information on how to do this, and the allowed values for these parameters). The data from the input bands will internally be assigned the following indices (and they will be included in the bandmerged output file in this order):

```
1 = IRAC-3.6
```

```
2 = IRAC-4.5
3 = IRAC-5.8
4 = IRAC-8.0
5 = MIPS-24
6 = MIPS-70
7 = MIPS-160
```

The indices are used throughout the Bandmerge module to sort, process and identify the input bands. For example, if a processing run has three input bands, given in the order MIPS-70, IRAC-3.6 and IRAC-8.0, these will be assigned index values 6, 1 and 4, respectively, and will be re-ordered as 1, 4 and 6 in the output.

Bandmerge will also check and update a number of other header keywords. Specifically, it will count the actual number of sources listed in the file, and update *TOTAL_PS_NUMBER* to match. Similarly, keywords CDELT and NAXIS are compared to the values in either *FIF.tbl* or the reference FITS image, and are updated to be consistent.

The source detection tables can be kept in any directory, as long as they are specified in the namelist with their full or relative paths.

Table 2.1: Example of a source detection table

```
\int Total_PS_Number = 1315
\float CDELT1 = -0.000167
\float CDELT2 = 0.000167
\int NAXIS1 = 501
\int NAXIS2 = 501
|srcid        |detid    |RA           |delta_RA   |Dec          |delta_Dec   |delta_RAD  |
|i            |i        |d            |r          |d            |r           |r
|1             1         149.4873834   6.54e-06    2.7578815    7.69e-06     -6.20e-07
|2             2         149.4859230   1.95e-06    2.7584803    1.97e-06     -4.53e-07
```

2.2 Namelist Files

2.2.1 Primary Namelist

The input parameters for Bandmerge are specified in the primary namelist file (e.g. *bandmerge.nl*). It must be located in a subdirectory *cdf/* of the directory in which Bandmerge is going to be run, i.e. if you plan to run Bandmerge from */Users/joe/*`data` then the namelist must be in */Users/joe/data/cdf/*. The namelist file is a simple ASCII text file, with two main sections:

1) Global Parameters section, which controls input/output files, header keywords etc.

2) Module Parameter Block, which controls the values of the parameters to be used in the bandmerging process.

A quick-look annotated example is given in Appendix A, and assumes that you are trying to merge data from IRAC channels 1 and 3, and MIPS-24. The input parameters are described in detail in Chapter 3.

2.2.2 Secondary namelists

Bandmerge runs as an iterative process. It starts with a crude estimate of the positional offset between different bands (given in the Bandpair uncertainty file – see §2.3), runs the bandmerging module, followed by the positional offset calculation, and then uses the calculated offsets to refine the source positions and re-run the bandmerging module. This whole process of bandmerge - positional refinement - bandmerge is defined as a single iteration. In the primary namelist, you can set the number of iterations you wish to run (must be 1 or more), and for each iteration you must provide a secondary namelist file to control the bandmerging parameters. For example, Bandmerge runs the first iteration using the parameters in the file *bmg1.nl*, and the second by using *bmg2.nl*, and so on. These secondary namelists are a stripped-down version of the primary namelist file, and only contain the information for the Module Parameter Block. An example of a secondary namelist is given in Appendix A, and can be found in the *cdf/* directory of your Bandmerge installation.

2.3 Bandpair Uncertainty File

Bandmerge requires an initial crude estimate of the offsets of the source positions between bands to start the iterative matching process. This file is called *bpru.tbl*, and a default version is provided in the *cdf/* subdirectory of your Bandmerge installation. Most users will find this default file is fine for their needs. Columns 1 and 2 in the file specify the pair of bands that are being compared. Columns 3, 4 and 5 are the positional shifts between the pair of bands in X, Y and a cross term shift (XY). This file can include initial values for all 7 bands, even though you may only merge a subset of them. The values in columns 3, 4 and 5 will be updated by Bandmerge during the iterative source-matching process.

Table 2.2: Example of a bandpair uncertainty file

```
\char comment = Spitzer band-pair registration uncertainties
\ Generated by getoff vsn 1.91 A60816 on Thu Jan 11 12:16:05 2007
|seed_index|cand_index| X_sigma  | Y_sigma  |XY_csd  |
|    int   |    int   |  real    |  real    |  real  |
          1          2   0.15      0.05       0.0
          1          3   0.15      0.05       0.0
          1          4   0.15      0.05       0.0
          1          5   0.15      0.05       0.0
          2          3   0.15      0.05       0.0
          2          4   0.15      0.05       0.0
          2          5   0.15      0.05       0.0
          3          4   0.15      0.05       0.0
          3          5   0.15      0.05       0.0
          4          5   0.15      0.05       0.0
```

2.4 Fiducial Image Frame Table or FITS Image

In order to merge data taken with different detectors, the source positions in each band must be converted to a common reference frame. The source lists from Spitzer give the source positions in RA and Dec, which must be converted to Cartesian (X,Y) coordinates. This is done using a Fiducial Image Frame (FIF) – a table defining the independent coordinate system and the spatial boundaries onto which the sources from each waveband can be projected. The FIF is a standard output from MOPEX (*FIF.tbl*, found in the output directory). If the user does not have an FIF table then Bandmerge must create one from one of the data frames that was used to extract the source list. In the example below we set the IRAC Channel 1 mosaic image to define the FIF. The choice of reference frame must be made carefully, and should usually be the waveband with the largest spatial coverage. Bandmerge will ignore any sources that fall outside the spatial coverage of the reference frame, regardless of the waveband.

The user must set one of the following in the primary namelist:

```
FIF_FILE_NAME = FIF.tbl
REFERENCE_FITS_FILENAME = irac_channel1_mosaic.fits
```

Chapter 3. Namelist Parameters

This section gives a full listing and description of all the input and output parameters available in Bandmerge. An annotated example of a Bandmerge primary namelist can be found in Appendix A. A similar example of a secondary namelist (essentially a subset of the primary namelist) can be found in the same section. See §2.2 for an explanation of the namelist types. The namelist is split into two sections – Global Parameters and Module Parameter Block. The primary namelist contains both sections, while the secondary namelists only include the Module Parameter Block section.

IMPORTANT NOTE: all input variables in the namelists require a space preceding and following the equals sign, i.e. "variable = value". An entry like "variable=value" will not be read and the hard-coded default will be used. You will not be warned that your input value is not being read.

3.1 Global Parameters

These are the parameters listed at the top of the namelist, and control the input and output flow for the Bandmerge Perl script. The variables for input into the Bandmerging algorithm are given in §4.2 Module Parameter Block.

run_bandmerge = 1 (switch)
Run the Bandmerge module. This must be set to 1. If set to 0, Bandmerge will not run.

convert_sky_to_cartesian = 1 (switch)
Convert the source positions from RA and Dec into Cartesian coordinates for source matching. Necessary if your input source lists give positions in RA and Dec. To turn this off, set convert_sky_to_cartesian = 0. See also convert_cartesian_to_sky.

convert_cartesian_to_sky = 1 (switch)
Convert the calculated source positions from Cartesian coordinates used internally by Bandmerge into RA and Dec for the final source output list. Set to 0 to get the output source list in X and Y instead of RA and Dec.

PointSourceList_Filename* = <input file *> (string)
File name of the input source detection tables. The namelist should contain a separate similar line for each input table, e.g.

```
PointSourceList_Filename1 = irac1_extract.tbl
PointSourceList_Filename2 = irac2_extract.tbl
PointSourceList_Filename3 = mips70_extract.tbl
```

etc…

Instrument_Channel* = <Instrument and channel number> (string)
A line for each of the input source detection tables telling Bandmerge which instrument and channel the data were taken with. Unless the table headers list the INSTRUME and CHNLNUM keywords, you must include these lines for each input table e.g.

```
Instrument_Channel1 = IRAC_1
Instrument_Channel2 = IRAC_2
Instrument_Channel3 = MIPS_2
etc…
```

MIPS-24, MIPS-70 and MIPS-160 are referred to as MIPS_1, MIPS_2 and MIPS_3, respectively. IRAC channels 1, 2, 3 and 4 refer to 3.6, 4.5, 5.8 and 8.0 microns, respectively. If your data were not taken with Spitzer then you must still assign them a Spitzer instrument and channel number to fool the Bandmerge software into running properly.

FIF_FILE_NAME = <name of Fiducial Image Frame table> (string)
File name of the input Fiducial Image Frame table (see §2.4). Either this keyword or REFERENCE_FITS_FILENAME is required.

REFERENCE_FITS_FILENAME = <name of reference FITS image> (string)
File name of the FITS image to be used as the common coordinate reference image (see §2.4). Either this keyword or FIF_FILE_NAME is required.

Input_BPRU_Filename = <bandpair uncertainty file name> (string)
File name of the initial Bandpair Uncertainty File (see §2.3).

Number_of_iterations = 2 (int)
Number of iterations that Bandmerge should use to refine the source positions. For each iteration, the user must also supply a secondary namelist file (see §2.2.2). This cannot be set to zero.

clean_type = 3 (int)
Tells Bandmerge which columns of the input source detection tables to update when running the source position refinement (see §2.2.2). Choices are 1 (update source positions), 2 (update source position uncertainties) and 3 (update both source positions and uncertainties).

s2c_prefix = 'cnv' (string)

Sets the prefix to be added to the filenames after converting the positions from RA and Dec to Cartesian X,Y. Normally the user has no reason to change this.

OUTPUT_DIR = <output dir> (string)

Sets the name of the output directory.

OUTPUT_FILE_NAME = <name for output merged source table> (string)

Sets the name of the final output merged source table. Intermediate iteration output tables are given the prefix "iter1_", "iter2_" etc. This file is written to the output directory.

3.2 Module Parameter Block

The Module Parameter Block contains all of the variables for input into the Bandmerging algorithm. The block is delineated by the following lines:

```
&BANDMERGEIN
&END
```

See Appendix A for examples. Most users will find that the default values for the module are satisfactory for their needs. Input parameters are as follows, given in alphabetical order:

Chi_Sq_Denom_Min (double, default 1.0e-10, optional)

Minimum values allowed for the denominator in chi-square and pseudo-chi-square calculations (see §1.6).

Chi_Sq_Max (float, default 36.0, optional)

Maximum allowed chi-square value used to specify the match threshold (see §1.6).

Dump (int, default 0, optional)

Produce optional dump output. Options are Dump = 0 (no additional output - default); Dump = 1 (basic output); Dump = 2 (expanded output; Spitzer data only). If non-zero, the namelist parameter Output_DUMP_Filename_Base must be specified.

Epsilon (double, default 1.0e-12, optional)

Minimum value allowed for determinant in calculation of the inverse of the covariance matrix (used in Position Refinement – see §1.7).

Input_Table_Column<*i*> = <column name> (string, <*i*> = 1-32, default is no optional columns, optional)

Allows the inclusion of extra columns from the input file to be copied verbatim to the bandmerged output file (e.g. if you want to include SNR in the output). <i> is a number running from 1-32 that controls the order in which the optional columns are written to the output file. <column name> is the name of the column header in the input file that you wish to include, and must be specified in single quotes, e.g. Input_Table_Column1 = 'SNR' will write the SNR column from the input file into the output file in the first column after all of the standard outputs.

LR_Scale_Fac (float, default 1.5, optional)

Scale factor used during Linkage Rejection Processing (§1.6.4) to calculate the looser threshold to be applied to bandpair matching. The new threshold is calculated by multiplying the original match threshold (either *Chi_Sq_Max* or *Pseudo_Chi_Sq_Max*) by this scale factor.

NL_print (switch, default 0, optional)

Switch indicating whether (= 1) or not (= 0) the contents of the namelist file should be printed to stdout.

Output_DUMP_Filename_Base (string, default is no dump file, optional)

Base of the output dump file (see Chapter 4). The suffix *#.tbl* (where # is the band number) will be added to the base filename. If *Dump* is non-zero then this parameter must be set.

Output SPCMB_Filename (string, required)

Name of the (QA) spectral combination counts output file (see §4.2).

Output_STAT_Filename (string, required)

Name of the output statistical summary table (see §1.8 and §4.3).

Pseudo_Chi_Sq_Max (float, default 36.0, optional)

Merge threshold used in confusion processing (see §1.6.4).

SigMin (double, units arcsec, default 0.03, optional)

Minimum sigma value allowed in position calculations (see §1.7).

SNR_Bin<n> (float, default SNR_Boin1 = 20.0, SNR_Bin2 = 50.0, SNR_Bin3 = 100.0, SNR_Bin4 = 1000.0, SNR_Bin5 = -999.9, SNR_Bin6 = -999.9, SNR_Bin7 = -999.9, optional)

Upper limit for signal-to-noise ration value for bin <n> used in statistical summary computations (see §1.8; Spitzer data only).

SNR_Threshold<n> (float, <n> = 1-7, default 10.0, optional)
Signal-to-noise ratio threshold for input band <n> used in statistical summary computations (see §1.8; Spitzer data only).

Status_Check_Flag (integer, default 0, optional)
May be set to 0, 1 or 2. Any other values will be ignored and a WARNING message will be printed. See Appendix B for details on status flags.

Status_Threshold (integer, default 0, optional)
May be set to 0 through 6. Any other values will be ignored and a WARNING message will be printed. See Appendix B for details on status flags.

Trace_PS<n> (int, <n> = 1-7, default 0, optional)
Source identification for a point source in input band <n> that should be traced through the processing. A value of 0 or the absence of this keyword indicates that no source should be traced in this band. This functionality has not been implemented, and so users should set this option to 0.

Warning_Max (integer, default 10, optional)
Maximum number of warnings that may be printed regarding divisions by zero, negative square roots, and/or negative (pseudo-) chi-square calculations found.

X_Window (float, units arcsec, default 10.0, optional)
Coarse window for search in X direction for merge Candidates (see §1.6.1)

Y_Window (float, units arcsec, default 10.0, optional)
Coarse window for search in Y direction for merge Candidates (see §1.6.1)

Chapter 4. Output Files

Bandmerge will always produce three (mandatory) output files. They will contain the bandmerged data, the spectral combination counts (QA) and the statistical summary. Other output files (e.g. the Dump files) are optionally produced.

4.1 Bandmerged Data Output File

The bandmerged data output file will include the following header keywords:

```
\  BANDMERGE Vsn 1.39 A50707 run on 07/01/08 at 15:02:28
\INT NBands = 3
\INT Total_Merged_Number =  50939
\CHAR Infile-1 = Output_1/iter2_cnv_irac1sci_extract.tbl
\CHAR Infile-2 = Output_1/iter2_cnv_irac3sci_extract.tbl
\CHAR Infile-3 = Output_1/iter2_cnv_mips24sci_extract.tbl
\INT N_Apertures-1 = 3
\INT N_Apertures-2 = 3
\INT N_Apertures-3 = 3
\CHAR Sort_&_Search_Direction = Y
\CHAR Outband-1 = Infile-1
\CHAR Outband-3 = Infile-2
\CHAR Outband-5 = Infile-3
\INT IntBlank = -999
\REAL RealBlnk = -9.9e+09
\CHAR CharBlnk = null
```

where:

- **Nbands:** the number of bands that have been merged,
- **Total_Merged_Number:** the number of merged point sources (Spitzer data only),
- **NSrc:** the number of merged sources (for 2MASS),
- **Infile-#:** the (path and) file names of the input source detection tables,

- **N_Apertures-#:** the number of aperture#bad_pix# columns included (Spitzer only),

- **Sort_&_Search_Direction:** which axis the output bandmerged sources are sorted on (Spitzer only),

- **Outband-#:** the output band numbers (see §3.1) and the corresponding input files (Spitzer only).

The Bandmerged output file will include the following columns:

- **IDNum:** a unique identification number for each bandmerged point source

- **X, Y, Delta_X, Delta_Y:** the refined coordinates with the corresponding uncertainties

- **CnfFlag:** the confusion flags; the character field has a variable width of NBands*(NBands – 1) where NBands is the number of bands being merged; thus the width varies from 2 – 42 characters. If only 2 bands are being merged, the column header is abbreviated to "CF". The field is composed of NBands sub-fields of width NBands – 1 each. The Nth subfield gives the number of sources in each band that passed the fine position test (see §2.1), in increasing order of band and excluding band N itself. The subfields are also given in increasing order of band. If more than 9 sources in a candidate band passed the fine position test, the symbol "X" is used. For example, assuming we are merging 3 bands (IRAC-3.6, IRAC-4.5, IRAC-5.8):

  If CnfFlag = 112132:
  "11": the lowest band (IRAC-3.6) found one candidate each in bands 2 and 3 that passed the fine position test.
  "21": the middle band, band 2 (IRAC-4.5), found 2 candidates that matched in band 1, and 1 candidate in band 3.
  "32": the highest band, band 3 (IRAC-5.8), found three matching candidates in band 1, and two in band 2.

- **srcid, flux, deltflux, aperture#, bad_pix# and any optional input columns:** all these columns are listed verbatim from the input files, one band at a time. They are given the band number as a suffix.

4.2 Spectral Combination File

The output (QA) spectral combination counts file consists of a table giving all possible spectral combinations and the corresponding number of point sources that were found during bandmerging. An example is given here:

```
Sp.Comb.  Count
========  =====
    001   33424
    010    5982
    011    9521
    100     165
    101     192
    110     146
    111    1509
===============
Total:    50939
```

The information in the above table is interpreted as follows:

- There were 33424 point sources in the first (lowest wavelength) input source detection table that were not merged with any other band.
- There were 5982 point sources in the second input source detection table that were not merged with any other band.
- There were 9521 sources from the first two bands that were merged with each other, but not band 3.
- There were 165 sources from the third input source detection table that were not merged with any other band.
- There were 192 sources that were merged between bands 1 and 3, but not with band 2.
- There were 146 sources merged between bands 2 and 3, but not band 1.
- There were 1509 sources merged in all three bands.

4.3 Statistical Summary File

This file is produced for Spitzer data only. It has the following format:

There are two mandatory header lines:

```
\  BANDMERGE Vsn 1.39 A50707 run on 07/01/08 at 15:02:28
\CHAR BMG_Output_file = Output_1/newtbl.tbl
```

where **BMG_Output_File** indicates the path and file name of the bandmerged output file that corresponds to this statistics file.

This is followed by an ASCII table with the following columns:

Cols	Name	Description	Units	Type	Format
1-4	B-P	Band Pair	-	C*3	1X,A3
5-16	SNR_bin	S/N ratio bin	-	C*8	1X,A11
17-26	AveDX	Average DX	arcsec	R*4	F10.5
27-36	AveDY	Average DY	arcsec	R*4	F10.5
37-46	StdDevX	X standard deviation	-	R*4	F10.5
47-56	StdDevY	Y standard deviation	-	R*4	F10.5
57-66	ChSqX	X Chi-square	-	R*4	F10.5
67-76	ChSqY	Y Chi-square	-	R*4	F10.5
77-86	ChSqXY	XY Chi-square	-	R*4	F10.5
87-95	NSum	Number of points	-	I*4	I19

where

- **B-P column:** the two bands that are given on the current table line (e.g. "1-2", "2-4" etc.)

- **SNR_bin column:** which S/N ration bin is given on the current table line, e.g. "10-20", ">1000". "Total", etc.

If the SNR column is not specified in the namelist file then no binning of statistics will be done, i.e. only the "Total" category will be computed and output. Furthermore, if SNR is present but a given detection has a null value, then that case will be omitted from the statistical computation, no matter how many other bands in the merged source have non-null SNR values (i.e. if any band for a given source has a null SNR then the entire bandmerged point source will be ignored during the statistics processing).

4.4 Dump Output Files

A "dump" output file will be produced after each completion of the cross-band linkage processing step (see §2.1.1) if the parameter "Dump" is included in the namelist file. There will be one dump file per input source detection table. The file output depends on the value of "Dump" (= 1 or 2). If Dump = 0 (default) then no dump file is produced. In both cases 1 and 2, the table is preceded by a one-line header:

```
\INT Band = <b_value>
```

where `<b_value>` = the primary band in this file (a.k.a. the seed band; the bands are numbered from 1-7, in the order of their band index).

If <d_value> = 1 then the output table will have the following format:

```
| srcid | pbnd | ptrid | chisq | pschisq |
| int   | int  | int   | real  | real    |
```

where

- **srcid** is the source identification of the seed (sorted in ascending order).
- **pbnd** is the band of the candidate being pointed to (1-7),
- **ptrid** is the source identification of the candidate (in pbnd),
- **chisq** is the chi-squared value for othe seed-candidate combination,
- **pschisq** is the pseudo-chi-square value for the seed-candidate combination (sorted in ascending order for each srcid).

If <dvalue> = 2 then the output file has the following format:

```
| srcid | nsmch  |  x  |  y  | sflux | pbnd | ptrid | npmch | pflux |
chisq | pschisq |
| int   | int    |real |real | real  | int  | int   | int   | real  |
real  | real    |
```

where

- **nsmch** is the match count for the seed point source,
- **x** and **y** are the coordinate position of the seed,
- **sflux** is the flux of the seed point,
- **npmch** is the match count for the candidate point,
- **pflux** is the flux of the candidate point source.

Appendix A. Example namelists

A.1 Primary Namelist

```
#Bandmerge.nl file example (primary namelist)
#in order to run bandmerge, set run_bandmerge = 1
#if input files list source coordinates in RA and Dec, set
#convert_sky_to_cartesian to 1. To output in RA and Dec, set
#convert_cartesian_to_sky = 1. A value of zero uses X and Y.

convert_sky_to_cartesian = 1
run_bandmerge = 1
convert_cartesian_to_sky = 1

#Input source detection file names. We recommend including the full
#path. For more input files (i.e. to merge more bands), include similar
#lines labeled PointSourceList_Filename4 etc

PointSourceList_Filename1 = /your/data/directory/irac1_extract.tbl
PointSourceList_Filename2 = /your/data/directory/irac3_extract.tbl
PointSourceList_Filename3 = /your/data/directory/mips24_extract.tbl

#Add the following lines for each input file if the keywords INSTRUME
#and CHNLNUM are missing in the header of the above files. The channels
#for MIPS-24, MIPS-70 and MIPS-160 are referred to as MIPS_1, MIPS_2
#and MIPS_3 respectively.

Instrument_Channel1 = IRAC_1
Instrument_Channel2 = IRAC_3
Instrument_Channel3 = MIPS_1

#if you have a pre-made Fiducial Image Frame table then specify it
#here, or...
FIF_FILE_NAME = fif.tbl

#...specify the file to be used to make the FIF
REFERENCE_FITS_FILENAME = irac1sci.fits


#band_pair_registration_uncertanties_file in CDF directory - required
Input_BPRU_Filename = bpru.tbl

#number of iterations to run. For no iteration, set
```

```
#Number_of_iterations = 0
Number_of_iterations = 2

#correction type used in iterations
clean_type = 3

#default is 's2c'
s2c_prefix = 'cnv'

#set output directory name
OUTPUT_DIR = Output_1
#set output file name
OUTPUT_FILE_NAME = 'newtbl.tbl',

&BANDMERGEIN
 Comment = 'Generic namelist file for bandmerge - default values.',
 Output_STAT_Filename = 'statfile.tbl',
 Output_DUMP_Filename_Base = 'bm_dump',
 Output_SPCMB_Filename = 'spcmbfile.bm',
 ChiSq_Denom_Min = 1.0e-06,
 Chi_Sq_Max = 24.0,
 Pseudo_Chi_Sq_Max = 12.0,
 LR_Scale_Fact = 2.5,
 Epsilon = -1.0e+39,
 SigMin = 0.03,
 Pseudo_Pos_SF = 1.0,
 Pseudo_Phot_SF = 0.0,
 X_Window = 20.0,
 Y_Window = 20.0,
 Status_Check_Flag = 0,
 Status_Threshold = 0,
 Dump = 1,
 Trace_PS1 = 0,
 Trace_PS2 = 0,
 Trace_PS3 = 0,
 Trace_PS4 = 0,
 Trace_PS5 = 0,
 Trace_PS6 = 0,
 Trace_PS7 = 0,
 Chi_Sq_Max = 12.0,
 LR_Scale_Fact = 1.5,
 Pseudo_Pos_SF = 1.0,
 Pseudo_Phot_SF = 0.0,
 X_Window = 10.0,
 Y_Window = 10.0,
 Status_Check_Flag = 0,
 Status_Threshold = 0,
 SNR_Threshold1 = 10.0,
 SNR_Threshold2 = 10.0,
 SNR_Threshold3 = 10.0,
 SNR_Threshold4 = 10.0,
 SNR_Threshold5 = 10.0,
 SNR_Threshold6 = 10.0,
 SNR_Threshold7 = 10.0,
 SNR_Bin1 = 20.0,
 SNR_Bin2 = 50.0,
 SNR_Bin3 = 100.0,
 SNR_Bin4 = 200.0,
 SNR_Bin5 = 500.0,
 SNR_Bin6 = 1000.0,
 SNR_Bin7 = 2000.0,
 SNR_Bin8 = 5000.0,
 SNR_Bin9 = 10000.0,
 NL_print = 0,
&END
```

A.2 Secondary namelist

```
#Example of a secondary namelist file, e.g. bmg1.nl

&BANDMERGEIN
 ChiSq_Denom_Min = 1.0e-06,
 Chi_Sq_Max = 12.0,
 Pseudo_Chi_Sq_Max = 12.0,
 LR_Scale_Fact = 1.5,
 Epsilon = -1.0e+39,
 SigMin = 0.03,
 Pseudo_Pos_SF = 1.0,
 Pseudo_Phot_SF = 0.0,
 X_Window = 20.0,
 Y_Window = 20.0,
 Status_Check_Flag = 0,
 Status_Threshold = 0,
 SNR_Threshold1 = 10.0,
 SNR_Threshold2 = 10.0,
 SNR_Threshold3 = 10.0,
 SNR_Threshold4 = 10.0,
 SNR_Threshold5 = 10.0,
 SNR_Threshold6 = 10.0,
 SNR_Threshold7 = 10.0,
 SNR_Bin1 = 20.0,
 SNR_Bin2 = 50.0,
 SNR_Bin3 = 100.0,
 SNR_Bin4 = 200.0,
 SNR_Bin5 = 500.0,
 SNR_Bin6 = 1000.0,
 SNR_Bin7 = 2000.0,
 SNR_Bin8 = 5000.0,
 SNR_Bin9 = 10000.0,
 Warning_Max = 10,
 NL_print = 1,
&END
```

Appendix B. Status_Check_flag and Status_Threshold

Because the point-source extraction program generates detections in a variety of quality classes (tagged by a detection "status" parameter), the project has felt that a capability may be needed to prevent certain classes from being mixed together by the bandmerging process. Therefore, a detection must qualify for consideration as a bandmerge partner with any given candidate, and vice versa. The status classes are as follows, where "mapping" refers to an internal status assigned during initialization.

Status	description	mapping
-3	"no more dof"	5
-2	"converged"	3
-1	"exceeded"	4
0	"no fitting performed"	6
+1	"previous success"	2
+2	"new success, converged"	1
+3	"new success, exceeded"	0

The status parameter, however, has not been declared mandatory; hence if it does not exist in the input detection files, then all detections are treated as qualified to mix with all others in different bands, and the control parameters `Status_Check_Flag` and `Status_Threshold` (see §4.2) are pointless to specify. Assuming that the status parameters are present in all bands, then the following rules for mixing and matching are in effect:

`Status_Check_Flag` is initialized to zero. It may be set to 1 or 2 (as well as 0) by including the `Status_Check_Flag` parameters in the namelist file. Any other values will be ignored and a WARNING message will be printed.

`Status_Threshold` is also initialized to zero. It may be set to 0-6 by including the `Status_Threshold` parameter in the namelist file. Any other values will be ignored and a WARNING message will be printed.

If `Status_Check_Flag` is equal to zero, both `Status_Check_Flag` and `Status_Threshold` (as well as the optional input value "status") are ignored in the processing (but if "status" has been included as an optional input column in the namelist file, this value will always be read from the input source detection tables and copied verbatim to the output bandmerged file).

A `Status_Check_Flag` value of 1 or 2 means that in the Cross-Band Linkage processing (see §2.1.1), all seed candidate pairs that pass the coarse windows tests will have the `Status_Check_Flag` checked.

If `Status_Check_Flag` is equal to 2, the function *`status_check`* is called with the seed and the potential candidate information. The status values for the seed and the candidate should be in the range -3 to +3.; these are mapped as shown in the table above to the range 0-6. The absolute value of the difference between the two mapped status values is returned (i.e. a value between 0 and 6) unless an error occurs, in which case -1 is returned. If `Status_Check_Flag` is equal to 1, the function *`status_check`* is called *twice*, first with the seed information, and second with the potential candidate information. In each case, the status value (originally in the range -3 to +3) is mapped to the range 0-6. This value is returned (unless an error occurs, in which case -1 is returned).

For both cases 1 and 2, the returned value is checked. If the returned value is negative (i.e. -1) then the processing is terminated with an error message. Otherwise, the returned value is compared to the `Status_Threshold` value. If the returned value is *greater than* the `Status_Threshold` value, chi-square for this seed-candidate combination is computed, etc. If the return value is *less than or equal to* the `Status_Threshold` value, this seed-candidate combination is rejected, and processing continues with another seed-candidate combination (assuming more exist).