Bandmerging SIRTF Point Sources

John W. Fowler

6 August, 2003

1.) Introduction

This paper discusses the current design for the downlink bandmerging module for point sources extracted from SIRTF IRAC and MIPS images. The discussion is concerned primarily with the algorithmic aspects of the processing rather than implementation. Some unresolved issues involved in this design are also discussed. The bandmerging module will be called *Bandmerge* herein.

1.1) Functional Requirements of the Bandmerge Module

The requirements levied on the Bandmerge module include the following:

1.1.1.) Merge point sources extracted in different bands from IRAC and MIPS images so that apparitions of a given celestial object in different bands are grouped together.

1.1.2.) Refine the position information for bandmerged sources (NOTE: there is currently a lien regarding user control over a given extraction's contribution to the refined position information, e.g., S/N threshold; no requirements on how a detection must qualify for contribution to position refinement have been received).

1.1.3.) Produce a statistical summary that includes: (a.) counters for the spectral combinations produced, binned by brightest-band magnitude; (b.) mean position offsets and variances on both position axes for all band-pair combinations; (c.) chi-square values for all band-pair position discrepancies, binned by brightest-band magnitude. Items (b.) and (c.) will use only bandmerged sources with at least one detection whose S/N is above 10 and containing at least a user-specifiable minimum number of bands merged (to reduce contamination of the statistics by false merges).

1.1.4.) Be able to operate with a variable number of bands to be merged, with a minimum of two and a maximum of seven bands; the output format will vary in length according to the number of input bands, i.e., there will be no filler or empty fields in the format (empty bands, however, will of course be indicated by standard fields).

Other requirements also exist which are not relevant to the discussions herein; these may be found in the SDS (674-SO-43 SSC-PD-4079) and deal with debugging capabilities, precision protection, error handling, and other aspects of implementation.

1.2) Input to the Bandmerge Module

For a detailed description of the input files and parameters, see the SDS document 674-SO-43 SSC-PD-4079. The descriptions below are meant only to support the rest of the discussion herein.

1.2.1) Point-Source Input: Bandmerge will read from two to seven input ASCII table files of single-band point-source information; all such files will be in the same format; the information will consist of the flux in the corresponding band (template-fit magnitudes and optional aperture magnitudes), the position in FIF (Fiducial Image Frame X and Y) coordinates corrected for distortion as necessary, and uncertainties in these quantities. The position uncertainties will include only the uncertainty due to extraction and distortion correction, not absolute telescope pointing reconstruction error which is common to all bands, and not band-to-band relative position error (see 1.2.3 below). In addition to these parameters, housekeeping quantities such as S/N, extraction goodness-of-fit, saturation flags, etc., may be carried in the point-source records; up to 32 table columns may be passed through. Note that mixtures of IRAC and MIPS bands are never processed by the Ops Net pipelines; they may be processed offline, in which case the users will prepare the FIF definition to be used for all bands. In all cases, the Bandmerge module may assume that all bands input during a single execution have position coordinates defined in terms of a common FIF, however that common FIF has been prepared.

1.2.2) NAMELIST Control Input: If specified on the command line, Bandmerge will read a NAMELIST control input that allows overriding of internal defaults on all processing control and thresholding parameters. Some NAMELIST parameters can also be specified on the command-line, in which case such values override the NAMELIST values (see the SDS for details).

1.2.3) Band-to-band relative position error is taken into account via the BPRU file (see SOSDL-SIS-PD-3023, Band-Pair Registration Uncertainties). The values of these uncertainties must depend on whether absolute astrometric references are used for pointing refinement of some or all bands.

2.) Unresolved Issues

Some aspects of the design cannot be resolved with certainty at this time, and so certain ambiguities will be retained and defaults assumed. For example, some components of a bandmerged source will probably be judged too dubious to be permitted to contribute position information to the position refinement of the bandmerged source; it will be assumed that such criteria will be resolved later, and for now this will be pictured as depending on obvious characteristics such as the source's signal-to-noise ratio. It is also assumed that the questions of how and where to compute the noise, what parameters to use for band-filling, how and whether to use density-dependent merge thresholds, etc., will be resolved later. Note, however, that it *is* assumed that bandmerging decision thresholds will be symmetric with respect to bands; that is, the threshold for processing two bands will not depend on which band supplies the *seed* and which supplies the *candidate*. Currently density-dependent thresholds are not used, but if this changes, then the density must be computed

from the sum of the densities in the two bands estimated at the two corresponding point source positions. The reason why this symmetry is desired is that it precludes closed circles of *inconsistent chains* (see the algorithm description, especially section 3.3, below). There has also been discussion of various confusion-processing modes and the possibility of an alternate fine-position test match criterion based on position probability density cross-correlation; these are the topics of ongoing study and are not implemented at this time. Finally, deblended detections may need special handling, as in 2MASS.

3.) Algorithm

The algorithm is essentially that used in the 2MASS processing generalized to a maximum of seven bands, with modifications only in the parameters used for sorting out confusion, since these are mission-dependent. The advantage of the 2MASS algorithm is that it has no *horizon*, i.e., its results do not depend on the order in which bands or sources within bands are processed, and it does not fail to recognize potential confusion the way methods based on processing windows occasionally do.

The algorithm may be summarized as follows. Let S_{ij} denote source number i in band number j, where by "source" we mean a data record containing the source parameters in high-speed memory. Besides photometric and position parameters, the source record also contains an array of pointers (i.e., indexes) to sources in other bands which have been found acceptable for merging; up to three pointers are kept for each candidate band, with the pointers in a given band arranged in descending order of merge acceptability (e.g., increasing position-discrepancy chi-square) and candidate-band pointer triples arranged in increasing order of wavelength. Thus the source S_{ij} has an array of pointers **P** in which the element P_{qm} is the index to the q^{th} acceptable candidate in the m^{th} candidate band for the i^{th} seed band, where q runs from 1 to 3, m runs from 1 to N_{band} - 1 in increasing wavelength order, and N_{band} is the number of input bands. The number of sources in bands j and k will be denoted N_j and N_k, respectively.

A loop over band is performed in which each band serves as a supplier of seed sources; for each seed band, a loop over all other bands is performed in which these bands serve as suppliers of candidate sources for merging with the seeds. For each seed band, a loop over all sources in that band is performed in which each source is processed as a seed for merging with candidate sources in the other bands. The result of these loops is that each source's pointers are set to the most acceptable merge candidates in all other bands. A source's pointers in any one band may contain from zero to three nonzero values, where a zero value indicates that the pointer is not set. If the first pointer is set, then its value is the i index of a candidate in the corresponding band with which it is most acceptable for merging. If the second pointer for the same candidate band is set, then this points to another less-favored source in the same band as the prime candidate, but the fact that more than one pointer is set will trigger confusion processing. Similarly, a third pointer may be set. The limit of three pointers is arbitrary and based on the idea that if more than three candidates pass the merge criteria, the situation is probably beyond recovery, but a counter for the total number of such candidates is kept in order to quantify the seriousness of the confusion, and this counter is included in the output merged record. No actual merging or any other irreversible process takes place until

after all sources have served as seeds and candidates, and any confusion processing required has taken place. The confusion processing may result in rejecting the most favored merge, in which case the second-best merge is elevated by moving its pointer up one slot (and similarly a third choice, if any, becomes a second choice).

As one would expect, most of the algorithmic complexity is in the confusion processing. For the sake of simplicity, and also to keep the role played by photometry minimal, confusion is resolved strictly within the two bands for which it has been diagnosed, without recourse to information in other bands. In other words, although a candidate that has pointers set to additional bands is probably more likely to be real than another candidate in the same band for which no pointers are set (or a smaller number of pointers is set), this aspect will not be used in the confusion processing because it introduces photometric preconceptions and also could become extremely complicated to use (e.g., the additional bands could be confused themselves, and this would have to be resolved before the credibility argument could be made; with up to seven bands, the combinations to be unraveled would be numerous). Thus two-band confusion will be resolved strictly with information pertinent to the two bands involved. There *are* two other forms of multi-band confusion that *will* be dealt with; these are called *excess linkage* and *linkage rejection*, and will be discussed below.

Besides input, initialization, and output, the algorithm will consist of six phases:

A.) Cross-band linkage
B.) Simple confusion processing
C.) Inconsistent-chain processing
D.) Excess linkage processing
E.) Linkage-rejection processing
F.) Position refinement

There are several computations in which small denominators and matrix determinants may occur; in such cases, the program will check against specific limits for each type of quantity and clip at minimum values if necessary. See the SDS for details.

3.1) Cross-Band Linkage

This processing phase is described above; a pseudocode sketch is given below, wherein it is assumed that relevant counters and pointers are properly initialized at the correct locations.

```
Loop on band j from 1 to N_bands     { seed band loop }
  Loop on source i from 1 to N_j       { seed source loop }
    Loop on band k from 1 to N_bands    { candidate band loop }
      If k = j then skip to end of loop    { skip seed band }
      N_w = 1                              { initialize coarse window trailing edge }
      Loop on source n from N_w to N_k      { candidate source loop }

        If S_nk behind coarse window, increment N_w            { slide coarse window }
        If S_nk in front of coarse window, exit candidate band  loop
        If S_nk passes merge threshold with respect to S_ij then       { see Appendix A }
          Increment counter for candidates in this band
          Insert pointer to S_nk in S_ij record as appropriate       { see section 3.2 }
        End If

      End of loop on source n from N_w to N_k     { candidate source loop }
    End of loop on band k from 1 to N_bands    { candidate band loop }
  End of loop on source i from 1 to N_j       { seed source loop }
End of loop on band j from 1 to N_bands     { seed band loop }
```

3.2) Simple Confusion Processing

Simple confusion occurs when a seed source has more than one candidate source that passes the main bandmerging test in the same candidate band. The main test employs position information only; other information may be used to resolve simple confusion once it is diagnosed.

In the pseudocode recipe in section 3.1 above, the step "Insert pointer to S_{nk} in S_{ij} record as appropriate" is performed as follows. The S_{ij} source record contains a counter N_{Ck} for the number of candidates in band k that pass the main merge test; this counter is incremented. If S_{ij} has no pointers set to band k (i.e., if N_{Ck} becomes one after it is incremented), then the index n (i.e., the index of the candidate source record within its own band's array of source records) is simply placed in the first-choice slot (position 1) of the pointer array P_{qm}, where the index q has the value one, and the band indicator m has the value k if $k < j$ and otherwise $m = k-1$.

In the case where S_{ij} already has one or more pointers set to band k (i.e., N_{Ck} is greater than zero before it is incremented), the quality of the new match must be compared to those which already exist. These matches are bookkept via the P_{qm} array and a corresponding array of match parameters, C_{qm}, which contain information about the quality of the match, e.g., position chi-squares, which probably take on a different form once confusion is diagnosed. We will assume herein that as soon as confusion is diagnosed, information in addition to position will be used to compute a *pseudo-chi-square* parameter that, like chi-squares in general, indicates a poorer match when it has a larger value.

When N_{Ck} takes the value 2, confusion is diagnosed for the first time, and the previous match must be reevaluated in the context of confusion. The exact nature of this reevaluation will be left undefined for now, but for illustration, in the 2MASS processing it involves flux differences and differences in the parameters generated during the point-source extraction process such as indications of possible saturation, origin from detector persistence, profile-fit quality, etc. Differences in such parameters are squared and individually scaled, then summed with the position chi-square, which also has a rescale parameter, to form the pseudo-chi-square. If the second match has a smaller pseudo-chi-square than the first, it replaces it as the first-choice match in band k; this involves shifting the previous match's pointer and quality value into P_{2m} and C_{2m}, respectively, and placing the corresponding values for the new match in P_{1m} and C_{1m}, respectively. Otherwise the new match is placed in the second-choice position.

When N_{Ck} takes a value greater than 2, confusion has already been diagnosed, and the previous matches have already been recast into pseudo-chi-square form. The newest match must also be evaluated in that form, and then its rank relative to those already being bookkept must be determined. If there is already a third-choice match and its pseudo-chi-square is less than the new match's, then the new match is rejected. Otherwise the new match will be accepted, and its position in the choice order is found as follows: if the new pseudo-chi-square is less than C_{2m}, then the current second-choice match is demoted to third choice; otherwise the new match becomes the third choice, and the pointer-insertion processing is finished. If the new pseudo-chi-square is less than C_{1m}, then the current first-choice match is demoted to second choice, and the new match becomes

the first choice; otherwise the new match becomes the second choice.

The pseudo-code recipe for this processing is as follows, where C represents the chi-square for the newest matched candidate, S_{nk}.

```
Increment N_Ck
If N_Ck = 1 then                    { first match found }
   Set P_1m = n and C_1m = C
Else if N_Ck = 2 then               { second match found }
   Recompute C_1m and C for confusion context
   If C < C_1m  then                { new match is better }
      Set P_2m = P_1m and C_2m = C_1m    { demote 1st choice }
      Set P_1m = n and C_1m = C          { new match is 1st choice }
   Else
      Set P_2m = n and C_2m = C          { new match is 2nd choice }
   End If
Else                                { third or more match found }
   Recompute C for confusion context
   If C < C_3m  or N_Ck = 3 then    { newest better than 3rd choice or no prior 3rd choice }
      If C < C_2m  then
         Set P_3m = P_2m and C_3m = C_2m     { demote 2nd choice }
         If C < C_1m  then
            Set P_2m = P_1m and C_2m = C_1m  { demote 1st choice }
            Set P_1m = n and C_1m = C        { new match is 1st choice }
         Else
            Set P_2m = n and C_2m = C        { new match is 2nd choice }
         End If
      Else
         Set P_3m = n and C_3m = C           { new match is 3rd choice }
      End If
   End If
End If
```

3.3) Inconsistent-Chain Processing

After cross-band linking and simple confusion processing have been completed, each detection may have pointers to other detections in other bands with which it can be merged according to the bandmerging decision and simple-confusion resolution algorithms. At this stage, however, there is no guarantee that a given seed's first-choice candidate in a given band also has a first-choice preference for the seed in the seed's band; there may be another detection in the seed's band that is a better match for the candidate and which was selected when the candidate was itself a seed. In fact, this nonreciprocal relationship could extend over quite a few detections, in principle, if the point-source extraction process happened to leave detections lying about with such unfortunate position relationships. This is depicted in Figure 1, where a *chain* of detections in bands B1 and B2 are shown with schematic positions and arrows indicating first-choice pointers. This is called an *inconsistent chain*, because the pointer relationships are not all *reciprocal*. Such chains must be found and corrected so that all pointers are reciprocal. This is done by finding the end of the chain and breaking the previous link. Because the merge decisions and thresholds are constrained to be symmetrical under seed-candidate interchange, inconsistent chains cannot form closed loops. The end of the chain occurs when a source finally points back to the source which pointed to it; this reciprocal relation is given priority, and the chain is broken at the previous link.

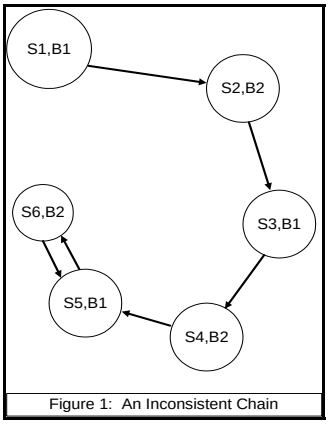

Figure 1: An Inconsistent Chain

Breaking a link involves modifying the pointer information of the source whose first choice was not reciprocated. This is done by elevating any corresponding-band second-choice and third-choice links that may exist and zeroing out the vacated pointer. In the example in Figure 1, the link from source S4 to source S5 would be broken; elevating S4's second-choice pointer to first-choice status, and if there is a third-choice pointer, it is upgraded to second choice. This should result in S4 pointing back to S3 or some other source outside the figure, so that the chain would once again end in a reciprocal relationship or link into another chain which does so.

This processing is performed by looping over all source records in the memory arrays, examining any first-choice pointers that may exist by accessing the corresponding source record and seeing whether its first choice points back. If so, continue with any other first-choice pointers to other bands; if not, follow the chain of pointers until a reciprocal relation is found, break the previous link, and then begin processing the current source record from the beginning, since the link breakage may or may not have solved its problem. Note that in general, this processing will begin somewhere in the chain other than at its beginning, but this does not matter. In Figure 1, if this chain were first found while processing the source S3, then the S3/S4 link would be fixed and the problems with S1 and S2 would go undiagnosed, but later on these sources would come up in the loop, and their problems would be fixed at that time.

3.4) Excess Linkage Processing

After inconsistent-chain processing has been completed, all pointers will be *reciprocal*, i.e., if a source S1 in band B1 has a first-choice pointer to a source S2 in band B2, then source S2 will have a first-choice pointer to S1 in B1. *Excess linkage* refers to a chain of sources with reciprocal links between sources but inconsistent band linkages within the chain. The simplest example is a chain composed of four sources (see Figure 2): a source S1 in band B1 is linked to a source S2 in band B2, which is linked to a source S3 in band B3, which is linked to a source S4 in band B1. Thus two sources in band B1 are in the chain, and one of the links in the chain must broken. This is done by breaking the link with the lowest match confidence (e.g., position chi-square or some pseudo-chi-square used to resolve confusion). Breaking a link here involves only zeroing out the corresponding-band pointers in each of the two sources being detached; second-choice pointers are not elevated because doing so would necessitate another pass through the inconsistent-chain processing, and this does not appear to offer appealing benefits.

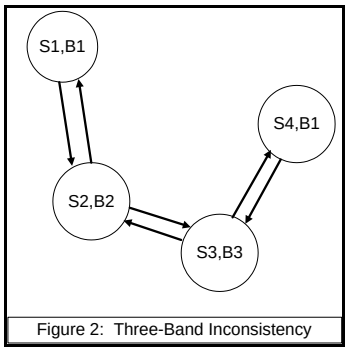

Figure 2: Three-Band Inconsistency

In general it is not always the case that a detection in the excess band should be disconnected from the chain; i.e., one could decide to break the weaker of the two links S1-S2 or S3-S4, leaving a B1 solo and a three-band source, but it may be that two double-band sources constitute the right answer, i.e., that the S2-S3 link should be broken. Furthermore, many other detections may be in the chain, even multiple excess linkages. For algorithmic simplicity and also because it appears to work as well as any other method that avoids the complexities of spectral analysis, the approach taken is simply to break the weakest link found anywhere in the chain, then to begin the analysis of the chain again from the same point.

3.5) Linkage-Rejection Processing

At this point, all linkages are reciprocal, and all groups of linked sources have each band represented no more than once. The only possible remaining problem is that not all sources are linked to each other. Such a situation is depicted in Figure 3, where source S1 in band B1 is reciprocally linked to S2 in band B2 and to S3 in band B3, but S2 and S3 are not linked to each other. In a true multi-band source, each detection must be acceptable to each other detection. The unlinked detections are tested with a separate position-match threshold applied only to candidates which have already been found acceptable to a common detection. This threshold is made separate so that it can be made a bit larger, since the candidates have some credibility as matches for each other by virtue of having already been found acceptable to a common detection. In the 2MASS processing, if this threshold is not passed, then the detection with the lower-confidence match to the common source is disconnected; otherwise the pointers are set between the previously unlinked sources. This works because this type of problem can occur only in three-band linkages in 2MASS. But in the SIRTF processing, up to seven bands may be involved, and a different approach must be used.

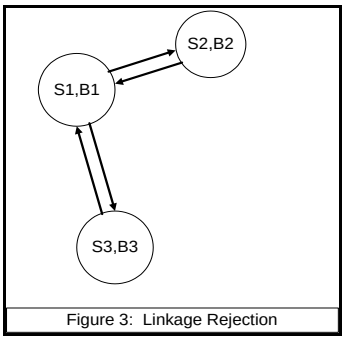

Figure 3: Linkage Rejection

For example, suppose that in Figure 3, the S1/S3 linkage had a lower quality than the S1/S2 linkage, and suppose that the S2/S3 combination was unable to pass the looser match test. Then S3 would become a single-band source, and S1 and S2 would remain linked. But suppose that the situation in Figure 3 is actually a subset of the situation in Figure 4, where other detections in other bands are also involved as shown. Although S1/S3 is still a weaker link than S1/S2, S3 is also linked to S4 and S5, as is S1. Only S2 is sitting out on a limb by itself. Despite the implied broad-band spectrum, S2 is simply unacceptable on the basis of position information, and it must be disconnected from the chain. This is done via the following rules:

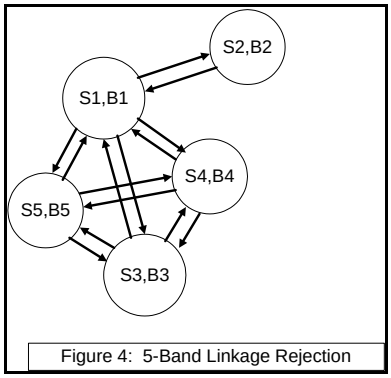

Figure 4: 5-Band Linkage Rejection

A.) Traverse the entire chain and locate the detection(s) possessing the fewest links.

B.) If only one detection has the smallest number of links, sever all of its links.

C.) If two or more detections tie as having the fewest links, sever the weakest link (i.e., maximum chi-square or pseudo-chi-square) among these detections.

D.) Test the chain again for linkage rejection; if found, start again at step (A.).

This processing is applied to all sources in memory until the situation is no longer found to occur.

3.6) Position Refinement

After all the processing described above has been completed, position refinement may be performed. This is done by looping over all bands, and for each detection in the band that has no linked detections at lower band numbers, pairwise refinement is performed, first with the lowest band's detection and the next higher band's detection, then the result of that refinement is paired with the next higher band's detection, if any, and so on until all linked detections that are eligible to contribute position information have been used. As discussed in Appendix A, it is assumed that the source-extraction processing has provided position-uncertainty covariance matrices, denoted Ω herein (note that the band-pair registration uncertainty covariance matrix in Appendix A is *not* used in the refinement processing; it enters only into the decision processing).

The pairwise position refinement is done as follows for each pair of position-parameter sets. The equations for the refined position parameters on the X and Y axes are given below with subscripts 1 and 2 indicating the two intermediate position-parameter sets. The refined parameters are indicated by a subscript "r".

$$\Omega_r = \left(\Omega_1^{-1} + \Omega_2^{-1}\right)^{-1}$$

$$\begin{pmatrix} X_r \\ Y_r \end{pmatrix} = \Omega_r \left[\Omega_1^{-1} \begin{pmatrix} X_1 \\ Y_1 \end{pmatrix} + \Omega_2^{-1} \begin{pmatrix} X_2 \\ Y_2 \end{pmatrix} \right]$$

The inversion of the two-dimensional real symmetric matrix is straightforward. Defining a general covariance matrix Ω and its determinant D as

$$\Omega = \begin{pmatrix} \sigma_x^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_y^2 \end{pmatrix}$$

$$D \equiv \sigma_x^2 \sigma_y^2 - \sigma_{xy}^4$$

the inverse Ω^{-1} is

$$\Omega^{-1} = \frac{1}{D} \begin{pmatrix} \sigma_y^2 & -\sigma_{xy}^2 \\ -\sigma_{xy}^2 & \sigma_x^2 \end{pmatrix}$$

Note that if the covariance term σ^2_{xy} is zero, this reduces to the common result

$$\Omega^{-1} = \begin{pmatrix} \frac{1}{\sigma_x^2} & 0 \\ 0 & \frac{1}{\sigma_y^2} \end{pmatrix}$$

4.) Testing

The test environment for Bandmerge must include dedicated simulators that can create input source files containing situations that will exercise all branches of the code. The emphasis will be on proper code execution, not astrophysical realism. If necessary, the simulators may be designed to generate large-scale astrophysically interesting situations as well, perhaps for threshold tuning purposes, but the main focus will be on smaller special-purpose scenes for exercising specific code branches. The simulator output must include "correct" Bandmerge output files for automated scoring purposes. Situations such as those discussed in section 3.2, 3.3, 3.4, and 3.5 must be constructed and used as test scenarios.

References


```
SOSDL-SIS-PD-3021 : bandmerged point source output file
SOSDL-SIS-PD-3022 : Bandmerge pointer dump file
SOSDL-SIS-PD-3023 : band-pair registration uncertainties
SOSDL-SIS-PD-3024 : Bandmerge statistics
```

Appendix A Merge Decision Algorithm

Whereas the bandmerging algorithm described above does not depend on the order in which seeds and candidates are processed, efficient searching for possible candidates for a seed does depend on such order. For this reason it is assumed that all input source lists have been sorted by the Cartesian position coordinate corresponding to the longest axis of the FIF, so that when sources are accessed in increasing source array index, they are being accessed in monotonically changing value of this position coordinate. This permits coarse searches for possible matches to be done efficiently, without having to examine every record in a candidate band for each seed. A coarse search simply requires a candidate to be within a generous window area centered on the seed. The candidate source list need not be searched from the beginning for each seed; instead a starting index for the search (initialized to one before processing the first seed in each band) can be incremented each time the source at that index is found to be outside the trailing edge of the window as it moves through the candidate array. Coarse searching ends as soon as a candidate is found to be outside the leading edge of the window. Candidates between the leading and trailing edges of the window are also checked against boundaries in the orthogonal direction. Only candidates within the coarse window are subjected to the fine position test, which is a chi-square test based on Gaussian position errors. When processing terminates for a given seed, the next seed in that band inherits the previous seed's trailing-edge indexes for each candidate band, since any candidate behind that band's trailing edge for the previous seed will also be behind it for the next seed, which is further along in the direction that the trailing edge is moving.

The coordinates of a seed will be denoted (X_s, Y_s), and those of a candidate will be denoted (X_c, Y_c). These coordinates are defined by the Fiducial Image Frame. The differences on each axis are used to compute the chi-square as follows.

$$\Delta X \equiv X_C - X_S$$

$$\Delta Y \equiv Y_C - Y_S$$

$$\Omega_s \equiv \begin{pmatrix} \sigma_{sX}^2 & \sigma_{sXY}^2 \\ \sigma_{sXY}^2 & \sigma_{sY}^2 \end{pmatrix}$$

$$\Omega_c \equiv \begin{pmatrix} \sigma_{cX}^2 & \sigma_{cXY}^2 \\ \sigma_{cXY}^2 & \sigma_{cY}^2 \end{pmatrix}$$

$$\Omega_b \equiv \begin{pmatrix} \sigma_{bX}^2 & \sigma_{bXY}^2 \\ \sigma_{bXY}^2 & \sigma_{bY}^2 \end{pmatrix}$$

$$\Omega_{sc} = \Omega_s + \Omega_c + \Omega_b \equiv \begin{pmatrix} \sigma_{scX}^2 & \sigma_{scXY}^2 \\ \sigma_{scXY}^2 & \sigma_{scY}^2 \end{pmatrix}$$

$$\chi^2 = \begin{pmatrix} \Delta X & \Delta Y \end{pmatrix} \Omega_{sc}^{-1} \begin{pmatrix} \Delta X \\ \Delta Y \end{pmatrix}$$

where the covariance matrices subscripted s (seed) and c (candidate) are assumed to have been provided by the source extraction program, and the covariance matrix subscripted b (band-pair registration uncertainty) is obtained from a table for the two bands involved (see SOSDL-SIS-PD-3023). This last covariance matrix expresses the uncertainty in registering any two given bands, which depends on whether the corresponding arrays are coaxial or require a slew to cover the same sky location. The expression above for χ^2 can be expanded to obtain

$$\chi^2 = \frac{\sigma^2_{scX}\,\Delta Y^2 + \sigma^2_{scY}\,\Delta X^2 - 2\sigma^2_{scXY}\,\Delta X\,\Delta Y}{\sigma^2_{scX}\,\sigma^2_{scY} - \sigma^4_{scXY}}$$

which, for the special case $\sigma^2_{scXY} = 0$, reduces to the more familiar form

$$\chi^2 = \frac{\Delta X^2}{\sigma^2_{scX}} + \frac{\Delta Y^2}{\sigma^2_{scY}}$$

This is the quantity that must be less than a user-controlled threshold for the seed/candidate match to be considered acceptable.