



Multi-Catalog Matching

Eric Charles

Science Verification Meeting, 2020/5/25



- Context
- Existing Software: *MultiMatch* and *SimpleAssociationTask*
- *NWayMatch*
 - Algorithm
 - Inputs and Outputs
 - Results
- Next Steps

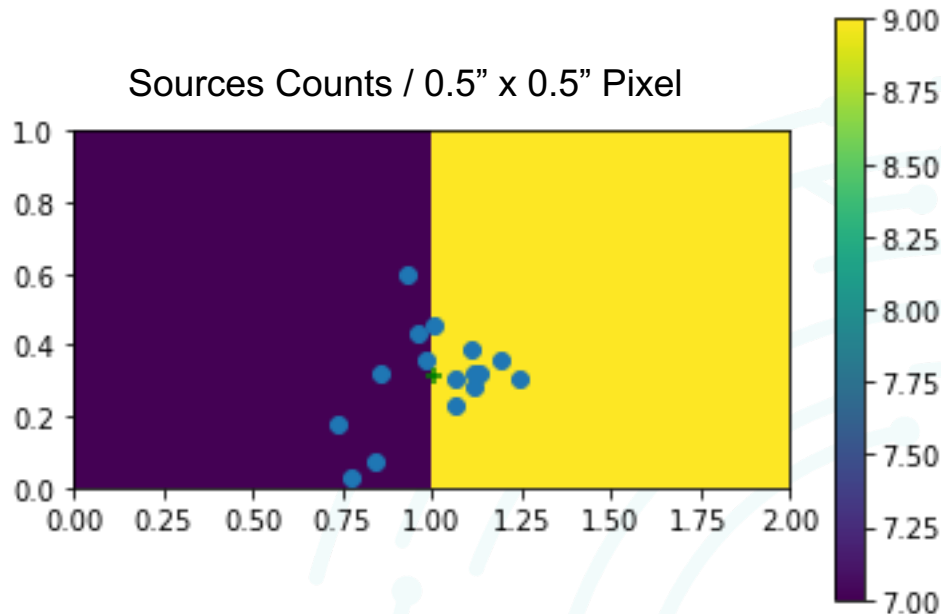
- Both Faro and dia_pipe (difference imaging) use N-catalog matching.
- Both have simple pieces of code that implement N-catalog matching as a series of 2-catalog matches and generate a simple table of “source – object” associations.
- In both cases, the authors of the code have said something to the effect that the matcher was a simple place-holder that they wrote quickly.
- Implementing N-catalog matching as a series of 2-catalog matches has a few limitations:
 - Order dependent: results change with order of input catalog
 - Forces choices about what to do with “ambiguities” before you have the full information

- *MultiMatch*:
 - Does a series of 2-catalog matches, keeping all matches
 - Updates reference catalog after each match
 - Optionally removes all associations with ambiguities
- *SimpleAssociationTask*
 - Does a series of 2-catalog matches, keeping best match
 - Updates reference catalog after each match
 - Keeps only best match, so no ambiguities

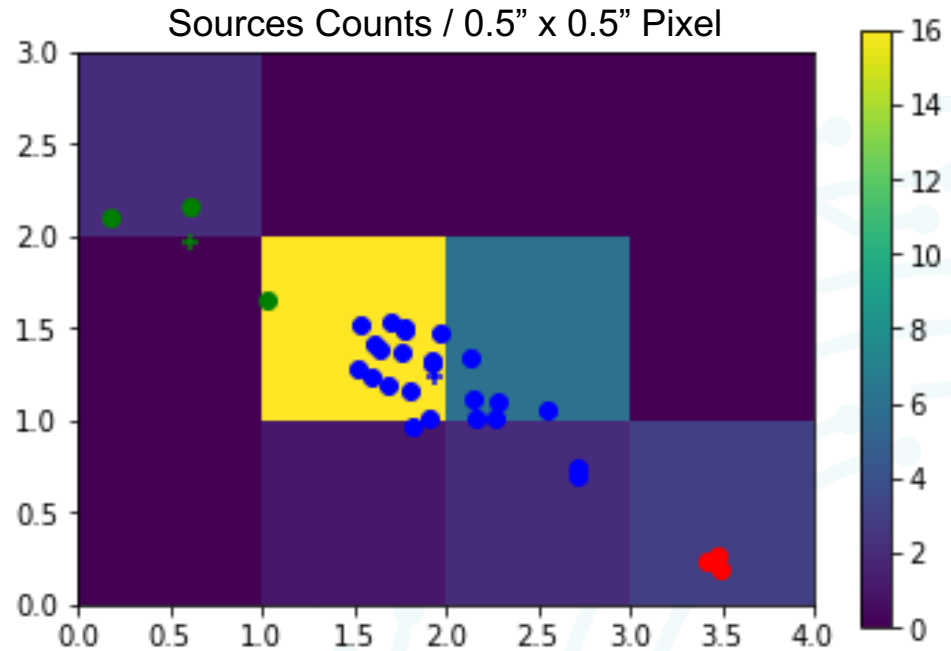
- Does *clustering* using sources from all catalogs
 - Footprint based source detection on source counts maps with match-radius sized pixels.
- Splits *clusters* in *objects* by removing outliers and resolving ambiguities
 - Currently uses brute-force recursion, but could easily use smarter clustering, e.g., minimal spanning tree.

- Inputs from source detection catalogs
 - Results are shown for 33 single exposure input catalogs covering $\sim 2^\circ \times 2^\circ$ region
 - <https://lsst.ncsa.illinois.edu/~yusra/nway-matcher/>
 - Take 5' to 10' to run code, depending on configuration (roughly the same as other algorithms)
- NWayMatch outputs four *astropy* tables:
 - Cluster statistics (cluster position, number of sources, objects & unique input catalogs per cluster)
 - Object statistics (object position, number of source per cluster)
 - Cluster associations
 - Object associations
- Comparisons are shown w.r.t. *MultiMatch* in default configuration

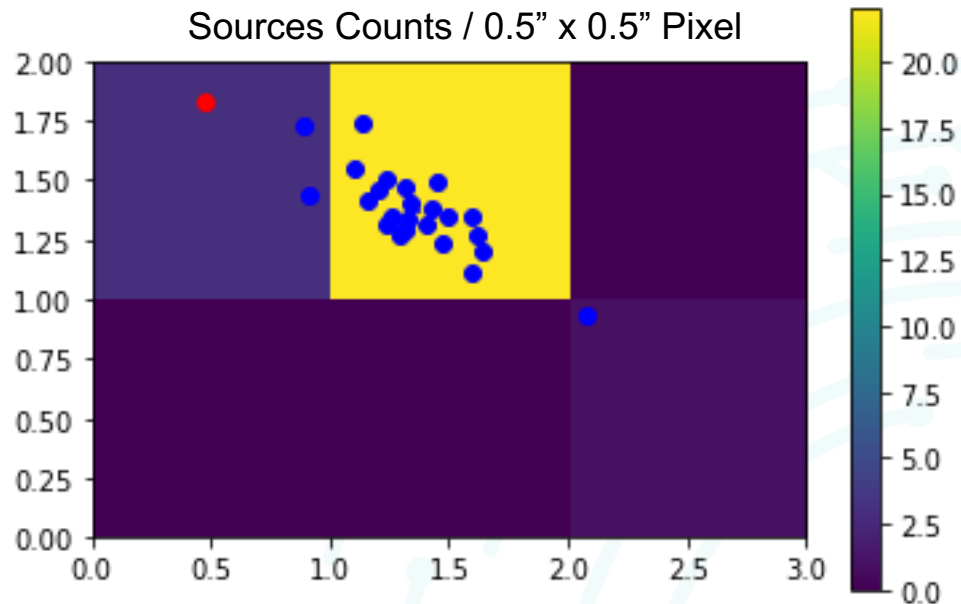
- This figure shows how the object detection works:
 - We project all the sources from all the input catalog into a sky map with the pixel size equal the to match radius
 - We the use the AFW Detection to find cluster of source counts
- In this case all the sources are with the match radius and from different input catalogs, so we are done.



- In this case, not all the sources are within the match radius, so we end up splitting the cluster into three objects
 - Object centroids marked with '+'

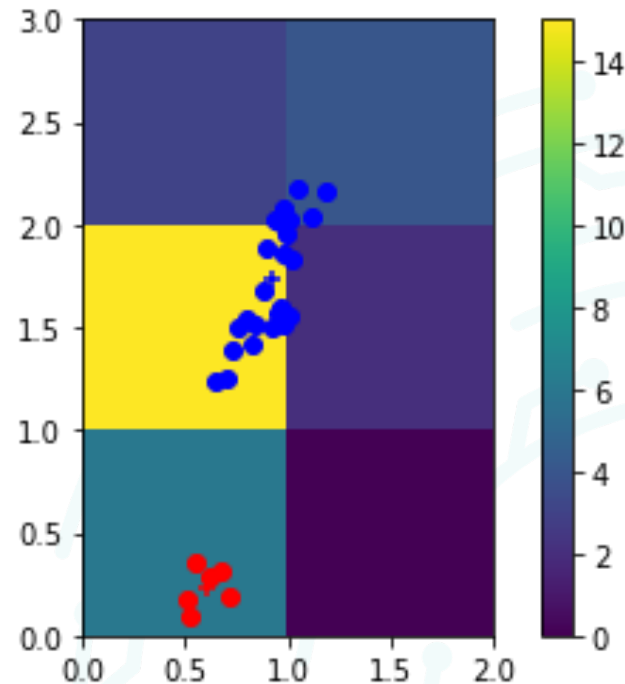


- This is a common failure mode:
 - The same source has been found in all the images except for one, in which it was split into two sources
 - In this case the one closer to the centroid of the cluster was included, and the other source was put into its own object

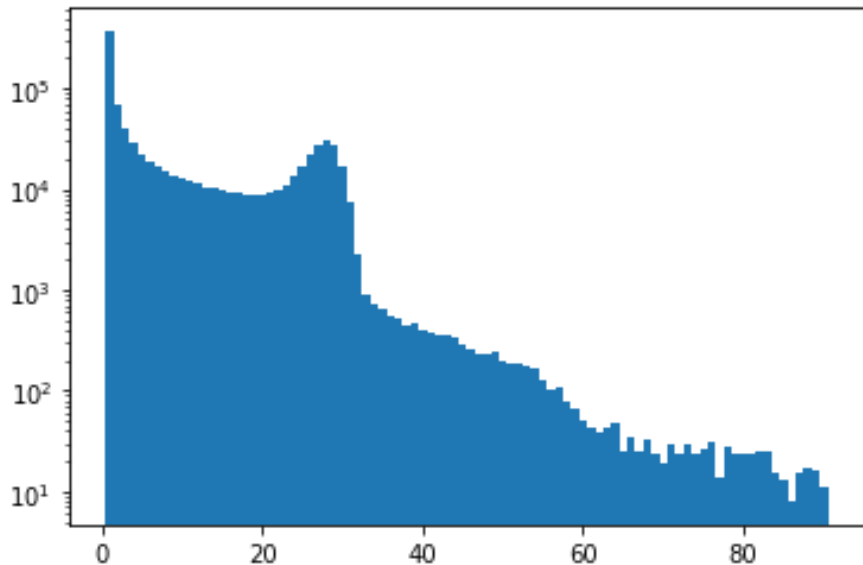


- In the case, MultiMatch removes **all** the associations for this object because of the ambiguity remove condition
- SimpleAssociationTask would probably do much better, giving about the same results as shown in the figure

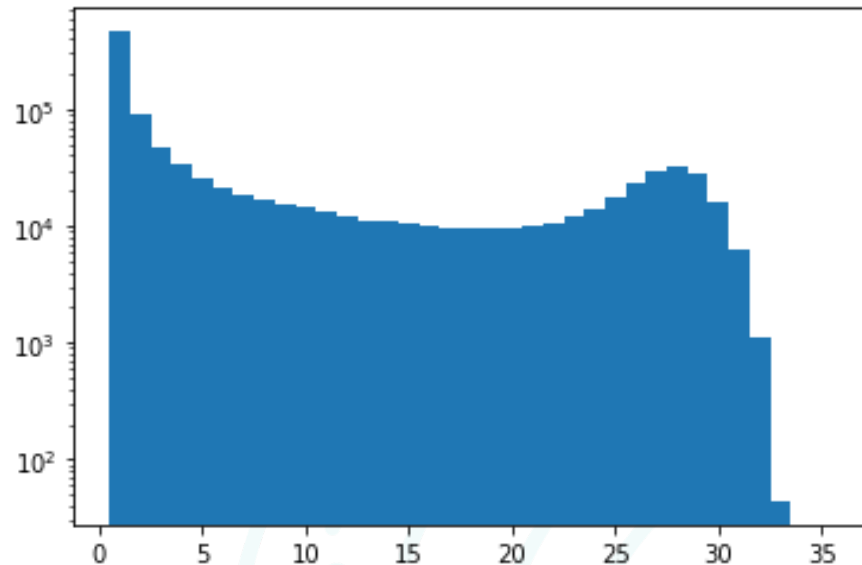
Sources Counts / 0.5" x 0.5" Pixel



Number of Sources / Cluster

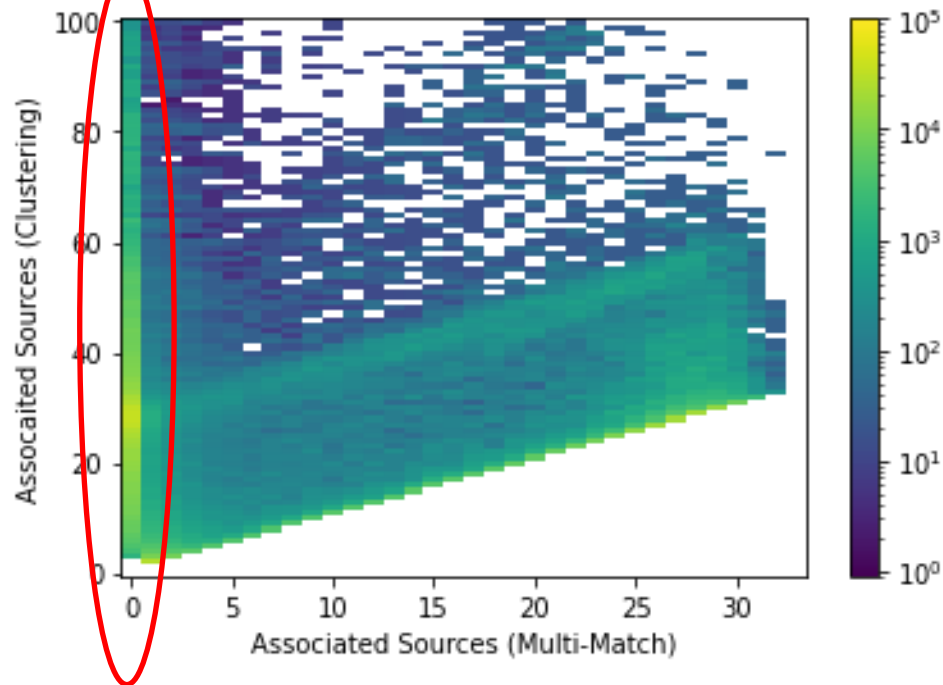


Number of Sources / Object

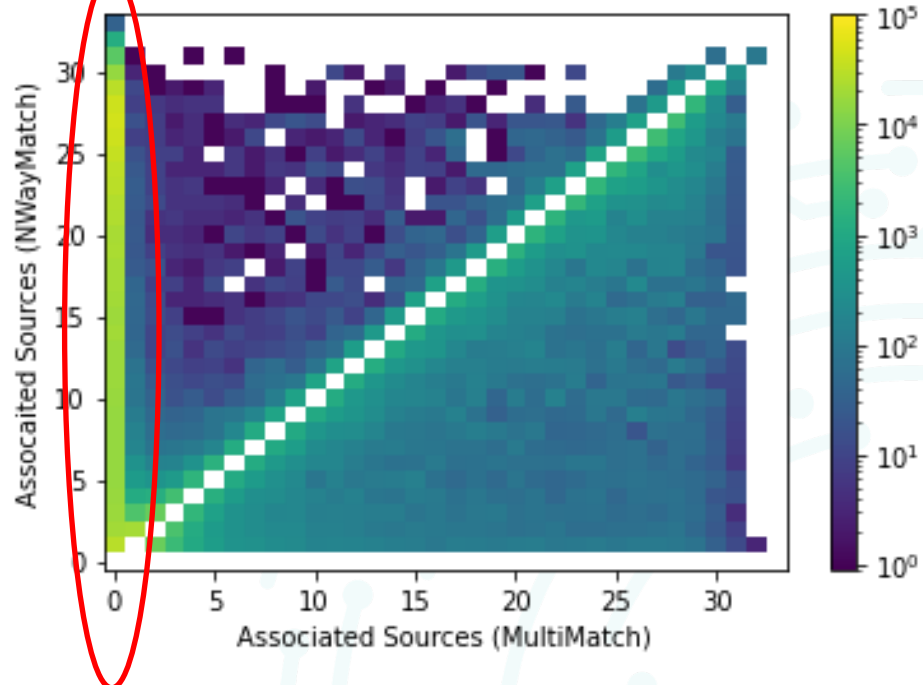


- 8636020 total sources from all the input catalogs
- 910162 total clusters are found, of which:
 - 367735 are single source clusters
 - 47776 have “ambiguities”
 - 64065 ended up split into more than one object
- 1050857 total objects found, of which:
 - 462058 are single source objects

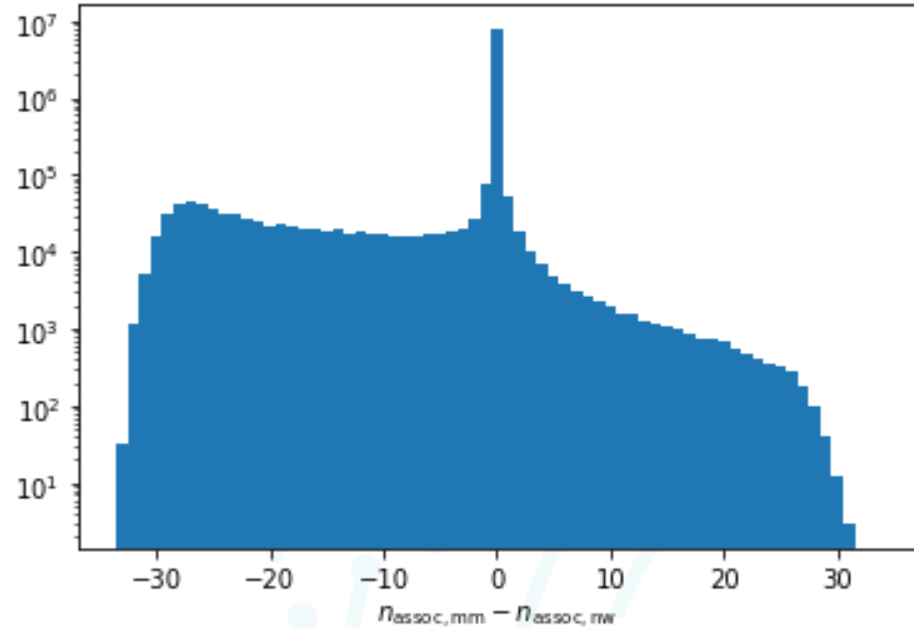
MultiMatch associations v. Clusters



MultiMatch associations v. Objects



- This histogram shows the difference in the number other sources each source is association with under the two algorithms



- In the large majority of cases the matching do the same thing
- Most (almost all?) of the cases with difference involve either blending sources, or incorrectly split sources
- The two advantage of using something like NWayMatch are:
 - Determinism, input catalog order doesn't matter
 - Gives you more control over how to present information for complicated cases, allowing you to explore blending / source splitting performance

- Make code available / merge into LSST code base:
 - Where?
- Compare results and performance with *SimpleAssociationTask*
- Test simple improvements to cluster splitting phase:
 - Minimal Spanning Tree clustering
 - Peak-finding inside AFW footprints