VERA C. RUBIN OBSERVATORY

# Multi-Catalog Matching

Eric Charles

Science Verification Meeting, 2020/5/25

NSF AURA U.S. DEPARTMENT OF ENERGY SLAC CHARLES AND LISA SIMONYI FUND FOR ARTS AND SCIENCES LSST CORPORATION

# Outline

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

# Context

- 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 table of “source – object” associations.
- In both cases,  the authors of the code have said something to the effect that the matcher was a 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
    - Potentially inefficient: updating object positions after each merge

# Existing Software

- *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”

# *NWayMatch* Algorithm

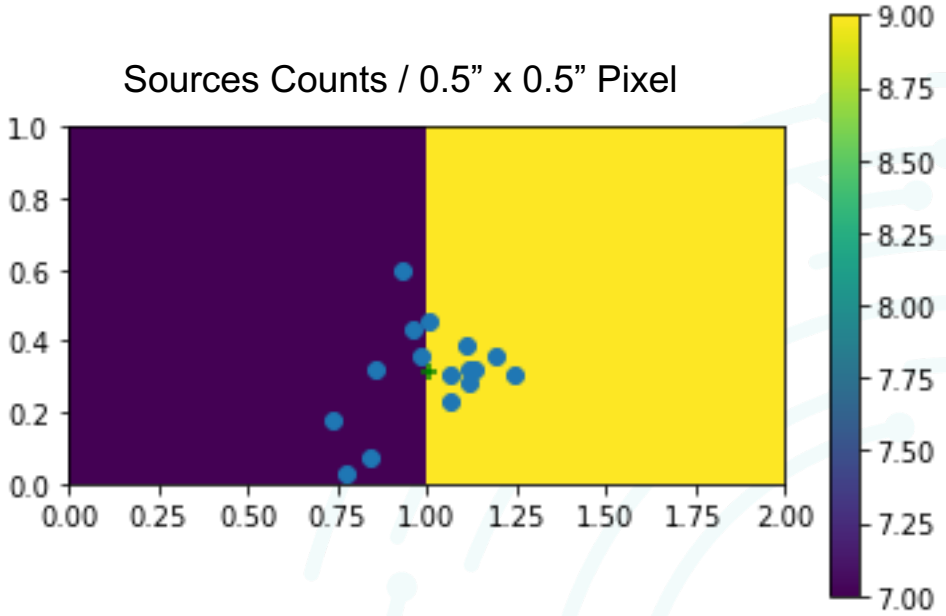
- Does *clustering* using sources from all catalogs
  - Footprint based source detection on source counts maps with match-radius sized pixels.
- Splits *clusters* in *objects* by removing outliers and resolving ambiguities
  - Currently uses brute-force recursive outlier rejection/ ambiguity resolution,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

# *NWayMatch* input and output

- Inputs from source detection catalogs
  - Results are shown for 33 single exposure input catalogs covering ~ $2^{o}$ x $2^{o}$ 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 (objection position, number of source per cluster)
  - Cluster associations
  - Object associations (equivalent to output of existing algorithms)
- Comparisons are shown w.r.t. *MultiMatch* in default configuration

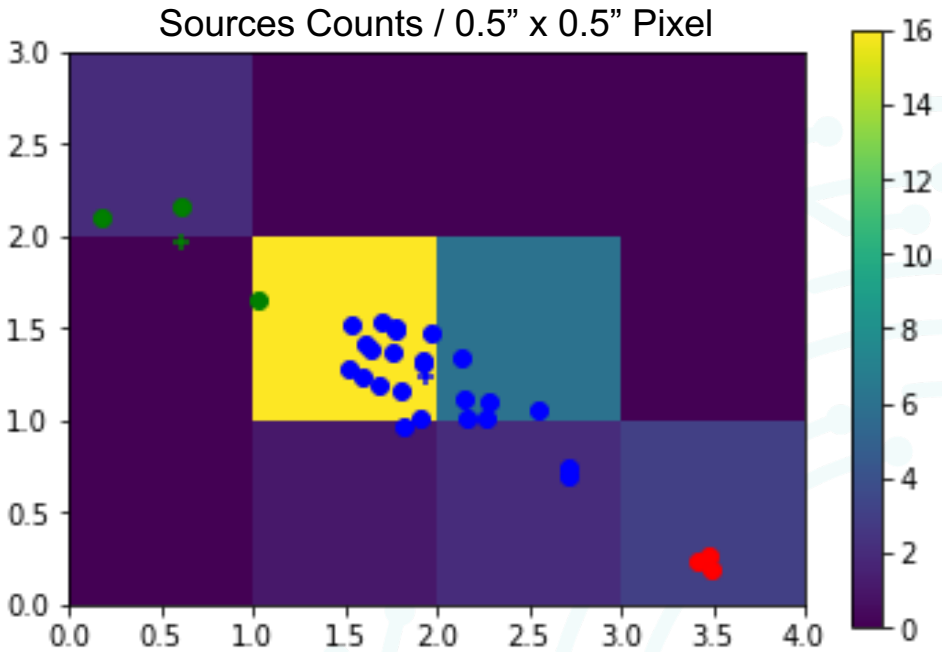
# Source Clustering

- 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 to the match radius
  - We then use AFW Detection to find cluster of source counts
- In this case all the sources are within the match radius and from different input catalogs, so we are done.



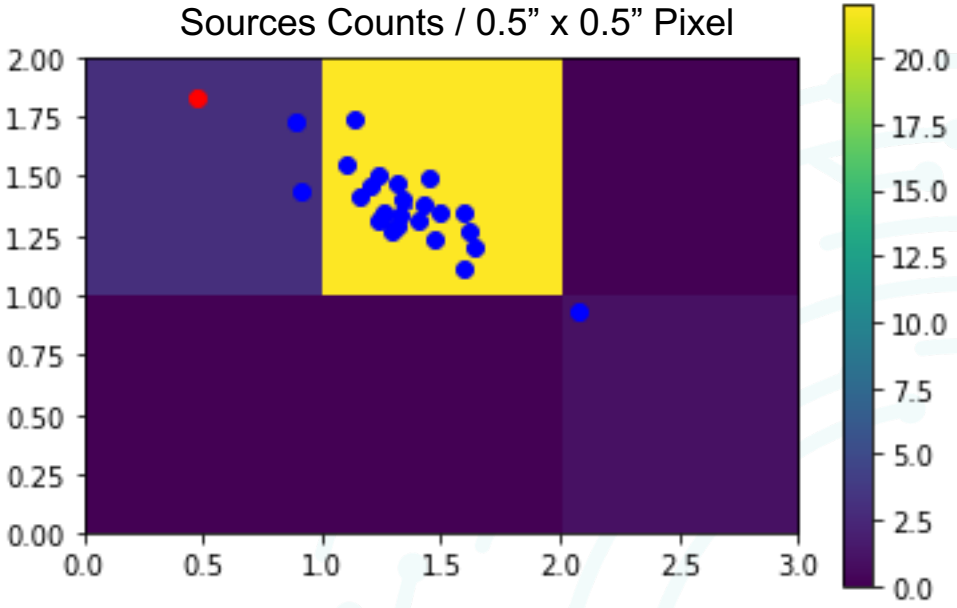
# Cluster Splitting

- 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 '+'



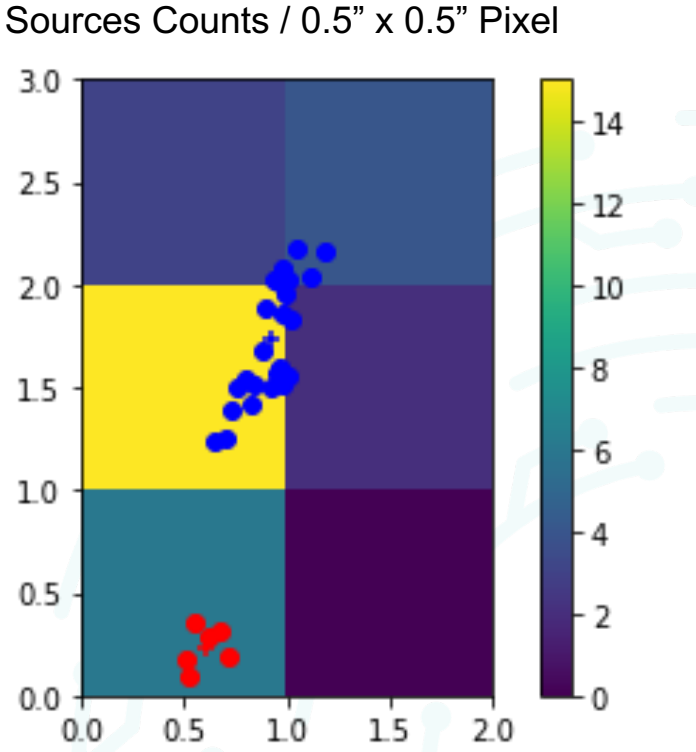
# Source Splitting

- 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



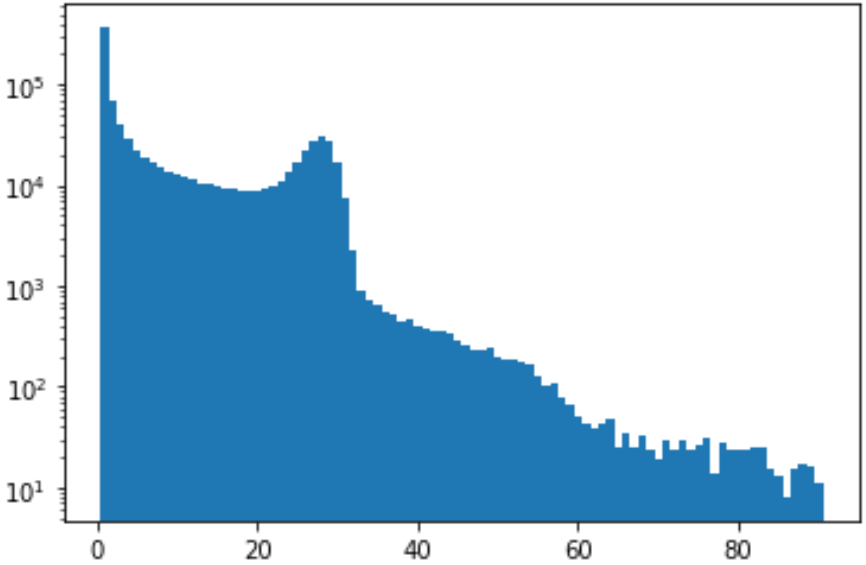
# *MultiMatch* ambiguity removal

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

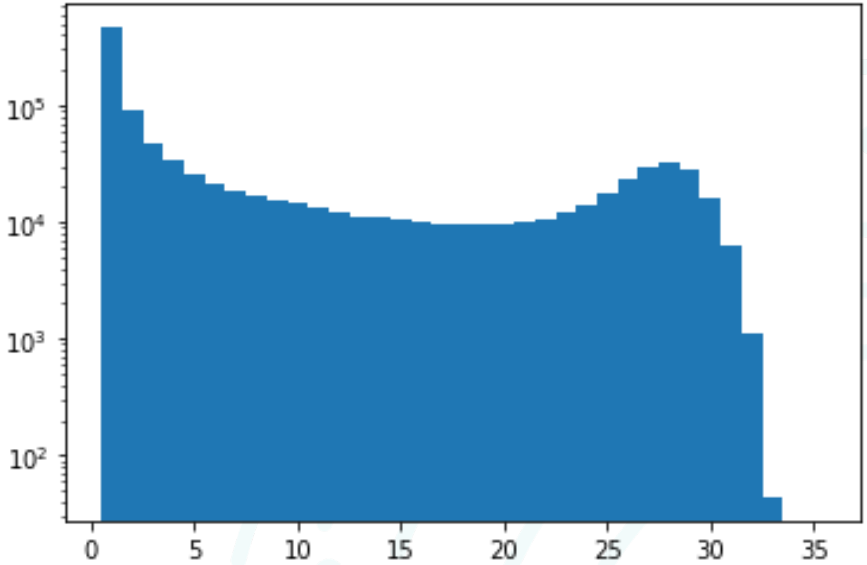


# Some stats from NWayMatch

Number of Sources / Cluster




Number of Sources / Object

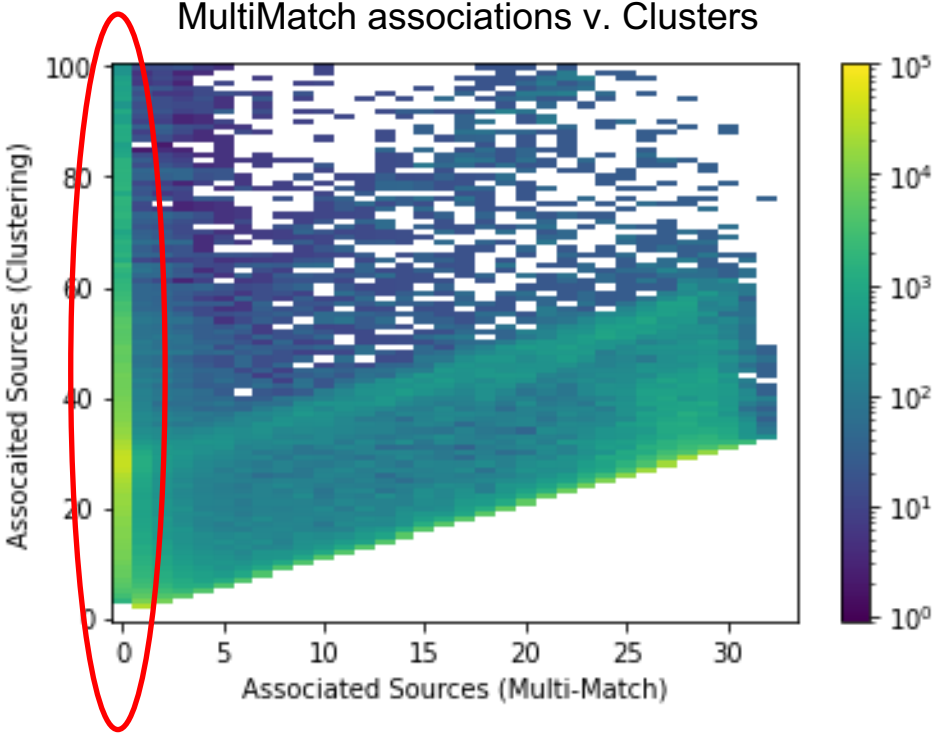


# Some Stats from *NWayMatch* run

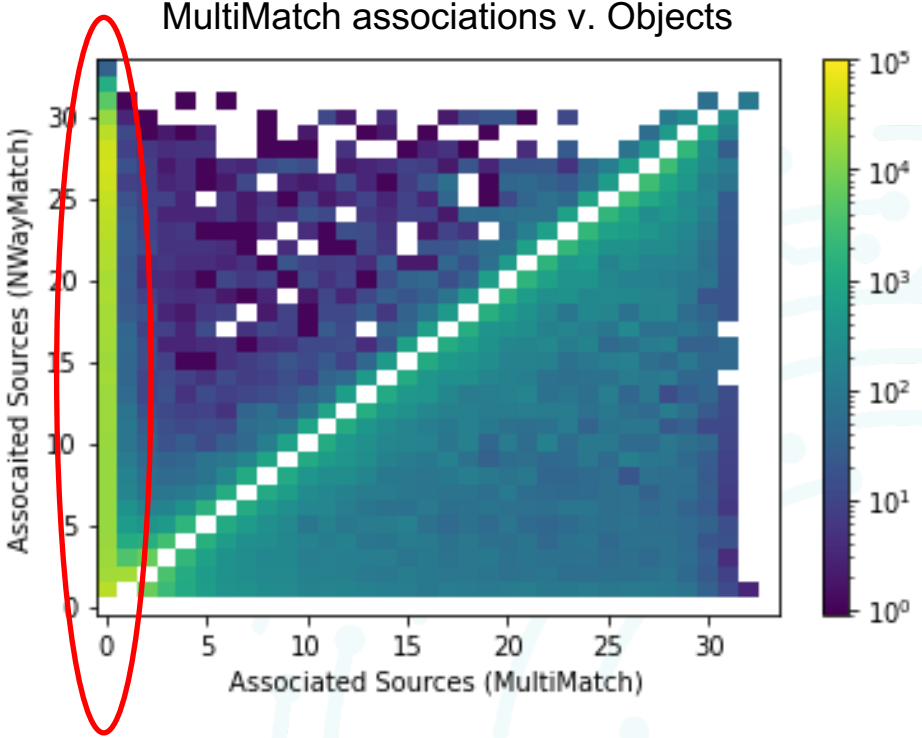
- 8636020 total sources from all the input catalogs
- 910162 total clusters are found, of which:
  - 367735 are single source clusters
  - 47776 have “ambiguities”
  - 64065 ended up split into more than one object
- 1050857 total objects found, of which:
  - 462058 are single source objects

# Comparison with *MultiMatch*

MultiMatch associations v. Clusters

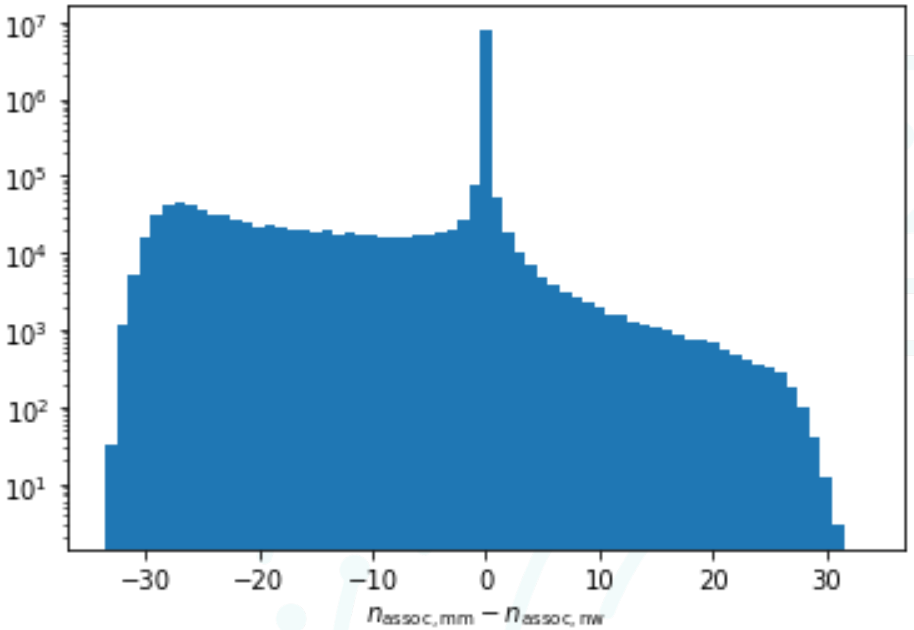



MultiMatch associations v. Objects



# Comparison with *MultiMatch*

- This histogram shows the difference in the number other sources each source is associated with under the two algorithms
- Many cases where *NWayMatch* finds more associations than the *MultiMatch*, these are largely attributable to cases where *MultiMatch* kills the entire cluster b/c of ambiguities



# Comments

- In the large majority of cases the matching algorithms do the same thing
- Most (almost all?) of the cases with differences involve either blending sources, or incorrectly split sources
- The advantages 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
  - Potentially more efficient: don't need to update the object position with each new input catalog

# Next Steps

- Make code available / merge into LSST code base:
  - Where?
- Compare results and performance with *SimpleAssocationTask:*
  - I expect that *SimpleAssociationTask* will find somewhat more associations than *MultiMatch*, as it is not indiscriminately removing all ambiguities
- Test simple improvements to cluster splitting phase:
  - Minimal Spanning Tree clustering
  - Peak-finding inside AFW footprints