Clustering analysis for the Gaia XM

Author: Andrés Gúrpide Lasheras
*Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.**

Advisor: Jordi Torra

Abstract: The cross-match (XM) is a sophisticated task that links all the Gaia observations with entries in the source catalogue. The main objective of this project is to perform a comparative analysis of several clustering algorithms that will be part of the XM of the Gaia detections to sources. The clustering results generated by each method will be compared against the *true* XM resolution provided by a simulator. Finally, a decision will be made based on the results of this comparative analysis regarding the selection of the clustering algorithm which best fits the XM requirements to be used in the Gaia data treatment.

I. THE GAIA MISSION

Gaia is a satellite by the European Space Agency (ESA), which main goal is to make the largest, most precise three-dimensional map of our Galaxy by surveying a billion stars with an unprecedented precision in position and motion. It was launched on 19 December 2013 from French Guiana. A first version of the catalogue is scheduled for mid 2016 and the publication of the final catalogue for 2022. The nature of the Gaia mission leads to the acquisition of an enormous quantity of extremely precise data, representing the multiple observations of a billion diverse objects.

A. THE CLUSTERING PROBLEM OF THE GAIA DATA TREATMENT

Gaia will observe each star 80 times on average during the 5 years of the mission. Each time it records its brightness, colour and position. After 6 months, the same region of the sky will be scanned again and another detection will be recorded for the same star, with different sky coordinates due to the accuracy of the systems and the movement of the star during that span of time. Therefore, the processing systems will have to handle a huge amount of observations grouped in clusters of about 80 detections. This is a clear example of clustering data treatment where the observations created by the same source have to be grouped together. Its implementation is inside a huge task called the Cross-Match.

The Cross-Match is a sophisticated task in charge of providing the links between individual Gaia detections and sources. As a result, it provides a single source link for each detection. In this sense, the observations are matched to sources both in a time-ordered and a space-ordered manner, and then clustered using the astrometric data in order to perform the final XM resolution. This process is split in three different stages (see figure 1):

1. The *Obs-Src match*. For each observation, a MatchCandidate (MC) is created containing the basic detection parameters (coordinates, scanning angle, spatial index, etc) and all the candidate sources, if any, according to a distance criteria. Therefore, all sources that fall inside a specified radius computed from the observation position are stored as potential sources that might be matched to it.
2. The *Sky Partitioner*. The purpose of this stage is to create self contained groups of MCs. The algorithm takes the first detection and retrieves all the sources that have been associated with that observation at the previous stage. Then, all the observations that have links to these sources are added to the group and treated like the first one. This recursive process ends when we have a complete group of MCs, meaning that all the observations linked to the sources in the group are in the MatchCandidateGroup.
3. The *Match Resolver*. The basic idea is to do the cluster analysis separately for each isolated MCGs and then link the resulting clusters with the corresponding catalogue entries or create new entries where necessary.

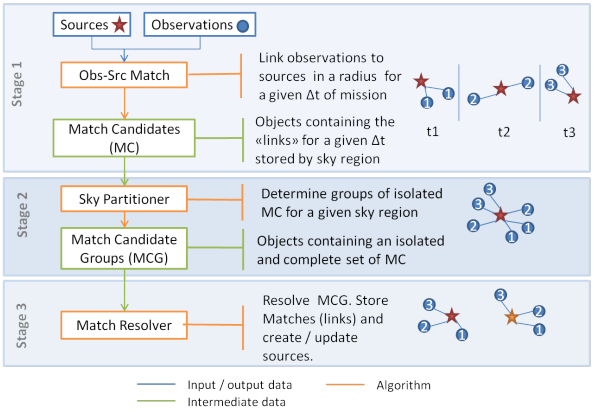

FIG. 1: XM task overview. In the example, three scans with 2 observations each are processed. In the end, 3 observations are linked to a catalogue source (red) and a new source is created (orange) for the remaining three even though they were initially linked to the red one.

*Electronic address: agurpila7@alumnes.ub.edu

The goal of the clustering is thereby to group together observations belonging to the same source. This is currently done using only sky coordinates (α and δ).
The aim of this work is to analyse different clustering algorithms under simulated Gaia-like data to choose the algorithm that best suits the requirements of the clustering process inside the resolution stage of the XM.

II. ANALYSED ALGORITHMS

The algorithms chosen for this analysis are: k-means, DBSCAN, Hierarchical Agglomerative and a modified version of the Nearest Neighbour Chain.
k-means, DBSCAN and Hierarchical Agglomerative clustering algorithms are deployed using Weka[4]. Weka is a workbench that contains a collection of visualization tools and algorithms for data analysis and predictive modeling, together with graphical user interfaces for easy access to these functions.
The Nearest Neighbour Chain algorithm is implemented in IDTools, an algorithms library for raw and intermediate data processing for Gaia, and is a modified version of the algorithm proposed by L.Lindegren in the early stages of the mission[1].

k-means

The basic idea of the k-means is to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. To achieve this, the algorithm minimizes the within-cluster sum of squares (WCSS):

$$WCSS = \Sigma_{i=1}^{k} \Sigma_j ||x_j - \mu_i||^2 \quad (1)$$

where μ is the mean value of all the observations in a given cluster (i.e. the center if only positions are used).
The algorithm starts by creating a set of k clusters randomly distributed. From this initial setup the algorithm reduces the distance between the members of the cluster and its center at each iteration.
The biggest drawback of the k-means algorithm is that the number of clusters must be set beforehand by the user. It is clear that this information will not be known beforehand in the case of the Gaia XM.
The most important configuration options for the k-means algorithm are:

- Number of clusters, namely k.
- Initialization method.

In particular, the initial k centroids can be set using some criteria instead of randomly. The final result may depend on the position of the initial position of the cluster centres as the solution may converge to a local optima. Therefore, the different initialization methods will be tested to select the method which best fits the XM. The available initialization methods in our setup are: *Random*,*k-means++*,*Canopy* and *Farthest first.*

DBSCAN

Density-based spatial clustering of applications with noise (DBSCAN) is a density-based clustering algorithm. In DBSCAN, points are classified as core points and outliers. A given point becomes a *core point* if it has more than a predefined number of points in its surrounding area, in other words a certain number of points are found within a defined radius. These points are called *reachables.* If p is a core point, it forms a cluster with all points that are reachable from it.
One of the features of this algorithm is that if points are isolated they might be considered as noise. In the Gaia XM all points should be treated so a specific configuration will be used to avoid this behaviour.
The most relevant configuration options for the DBSCAN algorithm are:

- Threshold distance, ϵ: The distance within, for a given point, the others are considered reachables.
- Minimum points, N: The minimum number of points to consider within a defined radius ϵ to start grouping points together.

Hierarchical Agglomerative

Hierarchical clustering methods try to build a hierarchy of clusters. In the case of agglomerative algorithms, the process starts with each observation creating a cluster. From there, pairs of clusters are merged. This process is repeated until all the observations are grouped into a single cluster unless a stop threshold is configured.
In this implementation, the stop threshold is the desired number of clusters to be produced. Variants of this method rely on how the distance between clusters is considered, the so called the *linkage criteria.* As clusters are extended objects, the distance between a pair of them can be computed from their centers, their farthest elements, their closest elements, etc. The most important configuration options are:

- Number of clusters.
- Linkage criteria.

Regarding the linkage criteria, the following configurations are available: *Single*, *Complete*, *Average*, *Mean*, *Centroid*, *Ward*, *Adjusted complete* and *Neighbour Joining.*

Nearest Neighbour Chain

The Nearest Neighbour Chain (NNC) algorithm is classified as a hierarchical algorithm. Differences with the algorithms presented previously rely on how the clusters are joined and the stopping rule criteria. In the NNC, clusters are agglomerated by mutual nearest neighbours. This means that when a couple of clusters are the nearest clusters from each other, they are merged. The distance considered in this implementation between two objects is the centroid criteria i.e. the distance from the cluster centres.
In the modified version presented in [1], the minimum

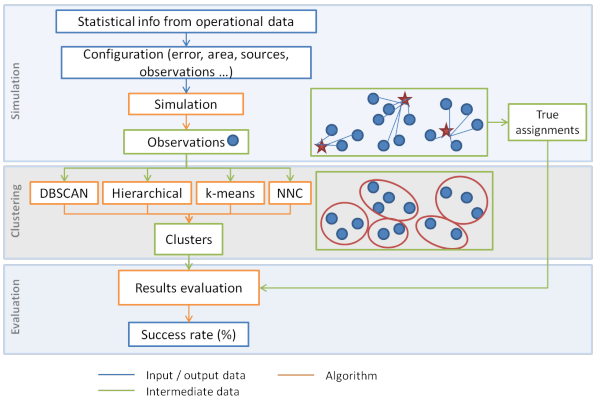

FIG. 2: Workflow of the algorithms evaluation process.

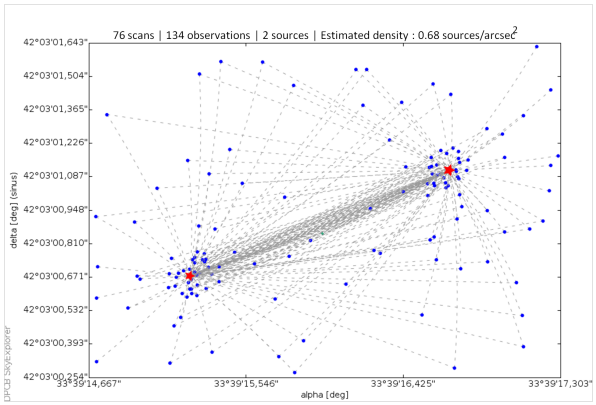

FIG. 3: Simulated MCG with 2 sources and some overlap between the observations. Note how all the possible source-detections links are contained in this region of the sky.

variance method is used to define the stopping criteria. Ward's minimum variance method defines the intrinsic dissimilarity of a given cluster as the sum of the squared residuals (SSR) with respect to the cluster mean:

$$R(C) = \sum_{O \in C} ||x(O) - x(C)||^2 \tag{2}$$

The coordinates of the cluster center, $x(C)$, are chosen to minimize the SSR. It can be seen that the agglomeration of two disjoint clusters C_i, C_j results in a cluster $C = C_i \cup C_j$ with SSR:

$$R(C) = R(C_i) + R(C_j) + \frac{n(C_i)n(C_j)}{n(C_i) + n(C_j)} ||x(C_i) - x(C_j)||^2 \tag{3}$$

As clusters are built up by agglomeration, their R value increase by the accumulation of the corresponding dissimilarities of agglomerated clusters. Thus we can then introduce the rule that an agglomeration is only allowed if the resulting internal variance $R(C)/n(C)$ of the resulting cluster is below a given limit.
The implementation tested in this study has been specifically adapted for the XM purposes and introduces the following features: identification and handling of duplicated detections and clustering by time compatibility. This latter only allows the agglomeration of two clusters if their observations are time compatible, i.e. they were detected in different scans. Thus, observations seen in the same scan will always end up in different clusters.

III. CLUSTERING SETUP

The clustering algorithms were evaluated under controlled test scenarios generated by a MCG simulator[2]. In this way, the performance of the algorithms can be evaluated by comparing the *true* information (i.e. the correct source for each observation) provided by the simulation against the generated clustering results. The workflow of the clustering algorithm evaluation process is shown in figure 2.
The simulator has the following relevant configuration options: number of sources per MCG, number of observations per source, distance between sources, maximum area of the MCG and observational error of the observations.
Modifying these parameters, one can generate a specific MCG situation. As an example, figure 3 shows the plot of a simulation with 2 sources and with some overlapping between the observations.

A. Success Rate Definition

It is desirable to quantify the success rate of the clustering in a single number. This can be used, not only to characterize the performance in a given situation, but to tune the parameters of a clustering algorithm in order to optimize the performance.
From the simulations we know the correct assignment of every observation to a unique source i.e. the *true clusters*. In this analysis, the success rate has been defined as the ratio between the correct clustered observations by the total number of them.
In order to compute the correctly clustered observations, a correspondence between true clusters and the resulting clusters from each method has to be found. This has been done by the maximum number of coincidences criteria i.e. clusters are matched with sources by maximum number of coincident observations. Once a source has been matched with a cluster, either of them can not be matched anymore. This can lead either to sources without corresponding cluster or the other way around, clusters that do not have an assigned source.

IV. CLUSTERING EVALUATION

The evaluation has been carried out in two parts: in the first part, algorithm performance under different XM scenarios has been studied depending on the configuration used. In the second part, their performance have been evaluated using more realistic scenarios.

A. Optimal calibration of the algorithms

Each clustering algorithm has a specific set of configuration options and the clustering performance depends on how these parameters are configured. A simplistic study was carried out to see which of the different available configuration options of each algorithm was the most suitable for the XM i.e. their optimal calibration for this purpose. In this study, the NNC was not considered as its calibration is out of the scope of this work.
The results obtained from the analysis are summarized as follows:

- **k-means**: The best performance was again obtained when the number of clusters was set to the number of *true* clusters. The best initialization method found was *farthest first*. This method sets the first centroid randomly and starts picking iteratively the farthest possible points from the already picked ones until the k-points are obtained. Then, the rest of the points are assigned to the cluster with closest center.
- **Hierarchical Agglomerative**: The best performance was obtained when the number of cluster was set to the number of true clusters, as expected. In reference to the linkage criteria, the *average* criteria was the one that gave the best results. The distance considered between two clusters in the this criteria is the average distance of all the distances between all the elements of the clusters considered.
- **DBSCAN**: Cases with higher density, required higher minimum number of points to achieve the optimal results. Consequently, in cases where there were no overlap between the sources, the required values were lower. For the threshold distance, the situation was similar. Higher density cases required higher values around 0.16 arcsec while for no overlap scenarios optimal values were found around 0.08 arcsec.

In general, k-means and Hierarchical Agglomerative performance were above DBSCAN. However, this performance was obtained when the number of *true* clusters was known in advance. Although this information will not be available in the Gaia-XM, this might be estimated using the scan information available. These test do not confirm if k-means and the Hierarchical Agglomerative can work properly without the *true* number of clusters set in advance.

B. Algorithms performance evaluation

To study the performance of the algorithms, thousands of different scenarios with variations in the density and distribution of observations and sources were generated. These were produced according to the statistical information obtained from the operational Gaia data in order to provide realistic cases. The algorithms were run under these scenarios and the success rate was computed in each case.

In order to plot the results, a dimensionless parameter was defined to measure the *clustering case complexity*. It was defined as $\sigma^2\Sigma$ where Σ is the density of sources and σ the observational error. This number gives the expected number of sources in a square of side σ and takes into account both the overlap between observations of different sources and the spread of the observations in a given cluster.
The observational error was randomly set between $-0.5 : 0.5$ pixels according to Gaia instrument. For the density, the area was computed using a convex hull algorithm to enclose all observations.
The number of sources was iterated from 2 to 20. These are the typical values found in MCGs, as observed in real data processing[3](in particular Figure 25). Note that the case with 1 source was not considered as it is trivial. The number of observations per source was fixed to a random value between 70 and 80 since this is the expected values at the end of the mission.
For a given number of sources, to generate different scenarios in terms of clustering complexity the area of the group was modified to increase the density. Hence, the clustering scenario becomes more difficult as the density increases, as it does the $\sigma^2\Sigma$ parameter.

C. Results

As one of the key configuration options in some of the algorithms is actually the number of clusters to be generated, we present two test scenarios.
First, we test the algorithms assuming that the number of clusters (sources) is known in advance. In this case the results reflect the ideal performance of each algorithm. Then a more realistic situation is presented where only the information that will be available in the Gaia XM is used to configure the algorithm. In this case the configuration is extracted from the actual data such as the number of observations per scan or the density of expected sources in the MCG. Taking into account only the information provided by the data, the number of clusters is estimated as the weighted average of the number of observations per scan.
For the DBSCAN, different methods were tested to adjust the calibration using information provided by the data itself but none provided reasonable results. The conclusion found after the tests was that the optimal parameters must fall around a threshold distance of 0.095 arcsec and 3 as the minimum number of points. However, the correct adjustment for each realization would require excessive complexity and effort, if possible at all.

Results with true number of clusters (sources)

Figure 4 shows the performance of the studied algorithms when we used the true information for the configuration as a function of $log(\sigma^2\Sigma)$. The general trend shows a satisfactory performance from all the algorithms (except for the DBSCAN) until the dimensionless parameter reaches -1, where the density is approximately of 10 sources/arcsec^2. At this point, the performance of the

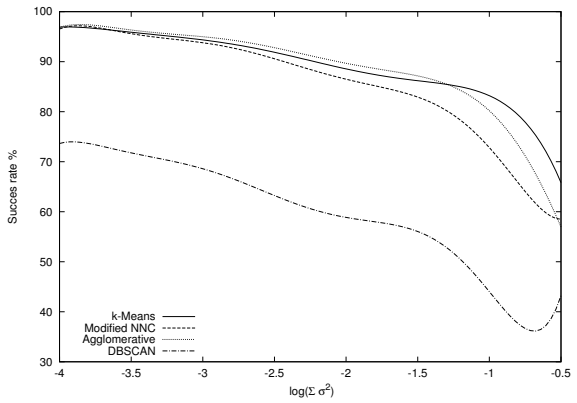

FIG. 4: Performance comparison of the algorithms when using true data in the configuration.

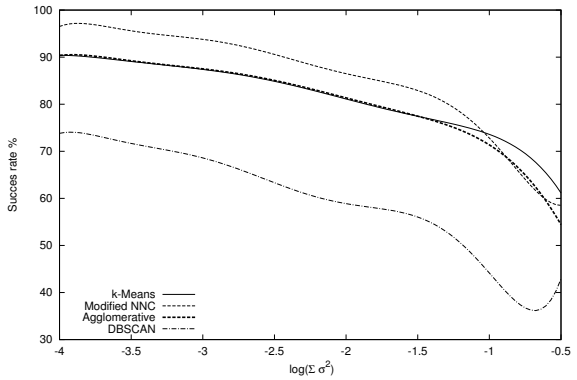

FIG. 5: Performance comparison of the algorithms when using only real data in the configuration.

algorithms starts dropping until about 60% for the most complex cases. Note that the performance of the NNC is comparable to the ideal performance of the k-means and the Hierarchical Agglomerative algorithm in most of the cases, but once the complexity of the case increases its performance drops much faster.
As can be seen, DBSCAN performance is well below the others probably due to the noise feature. Too many observations are treated as noise which reduces considerably the success rate.

Results without the true number of clusters (sources)

For these tests an estimation of the number of clusters was used to configure the algorithms.

As seen from figure 5, NNC performance is in general better than the other algorithms. At -1 its performance starts dropping below k-means. This is due to the increase of the overlap between observations. At this point, k-means is slightly better but in general NNC is preferable since no estimation of the number of clusters is needed. Again, DBSCAN is far from the performance of the other algorithms.

V. Conclusions

Considering the test results shown in the previous section we can conclude:

- DBSCAN is not a suitable alternative for the Gaia XM. The calibration in real scenarios is too complex and the performance results were below all the other alternatives. The main problem with this algorithm is that depending on the configuration some observations might be treated as noise, which is not a valid option for the XM task.

- Both k-means and Hierarchical Agglomerative algorithms show good performance in the tests. However, the configuration when only using information available to the XM task might lead to sub-optimal performance in both cases. In this case, not knowing the number of desired clusters k a priori makes both options not recommendable. However, it has been shown that we can obtain a good estimation of the k parameter just using available data.

- The modified NNC algorithm based on the proposal by L. Lindegren in [1] is still the most suitable clustering algorithm for the XM. The ratios obtained are amongst the best, and at the same time it does not require previous knowledge of the number of clusters. On top of that, its performance can be improved if properly calibrated as it was done for the other algorithms.

Acknowledgments

I would like to thank my tutor, Jordi Torra, for giving me the opportunity to develop my work on such a motivating field. I also would like to thank the DPCB team and in special Marcial Clotet and Juanjo González for their support and guidance throughout this semester of work. Finally, I express my thanks to my parents who have always encourage me to work hard and to fulfil my goals.

[1] [LL-060] L.Lindegren, *Cross-Matching Gaia objects*, GAIA-C3-LL-060 (August 2005)
[2] [MCL-029] M. Clotet, J.González, *MCG Simulator*, GAIA-C3-TN-UB-MCL-029 (April 2015)
[3] [JC-074] J. Castañeda, *IDT Re-Crossmatch Processing Report*, GAIA-DB-TN-UB-JC-074
[4] The University of Waikato, URL http://www.cs.waikato.ac.nz/ml/weka/ [Online; accessed 29/08/2015]