Detection of Open Clusters using Data Mining techinques

Author: Carlos Alegre Aldeano
Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.

Advisors: Carme Jordi Nebot, Alfred Castro-Ginard

Abstract: With the arrival of the third Data Release from the *Gaia* mission, we receive new information that can be really helpful to detect new clusters in the Galactic disc. So far, cluster hunting methods used five parameters, which are position, parallax and proper motions of the stars, in order to identify clusters. The new release arrives with the mean radial velocity (RV) measured for 33 million stars, which can be added as the sixth dimension in order to improve the efficiency of these methods.
In this work, we implement a six-parameter detection method based on the work developed for previous releases. The method searches for clusters in a region of the sky with two input hyper-parameters, the size of the box and the number of neighbour stars considered to form a cluster ($L, minPts$). We have run the algorithm for 81 different pairs, to determine which one performs better through several regions of the sky. The most efficient pair has been $(L, minPts) = (13°, 11)$, followed closely by $(16°, 12)$. Both had near 60% of the clusters found and 70% of correctly clustered stars while having a low number of field stars clustered, which means the results had lower noise.

I. INTRODUCTION

On 19th December 2013, the European Space Agency (ESA) launched the *Gaia* space telescope [1] with the objective of mapping more than one billion stars located in the Milky Way, gathering information about their position, motion, magnitude and several other parameters. Until this year, we have received different data releases from the *Gaia* telescope (*Gaia* DR1 [2], *Gaia* DR2 [3]), and early release EDR3 [4] with stars described homogeneously by astrometric parameters (α, δ, ϖ, μ_α, μ_δ). The third release (DR3) [5], has been published in 13th June 2022.

All of this information is key to study and understand several properties about our Galaxy, for instance stellar formation. One way to tackle these studies is using groups of stars with a common origin and formed from the same material as the main tracers. These groups of stars are called clusters, concretely open clusters if we are talking about the those born in the Galactic disc. Open clusters offer the opportunity to obtain more reliable astrophysical parameters than the single field stars, since these parameters are estimated averaging over the whole cluster stellar population.

With the different Data Releases, researchers have been able to apply data mining techniques in order to identify clusters contained in these large data sets (for instance, the method from Castro-Ginard et al. 2018, CG18 from now on [6]). CG18 method to detect new open clusters includes the five astrometric parameters available (α, δ, ϖ, μ_α, μ_δ). One can note that, out of the six-dimensional space formed by position and velocity, there is one missing parameter, the radial velocity (RV). The DR2 included mean radial velocities for almost 8 million stars [8], being the first time the data collected by the Radial Velocity Spectrometer was published. But it is with the *Gaia* DR3 when we receive more precise data, among other products, providing the mean radial velocities for 33 million stars.

In this work we describe the addition of the RV to existing methodologies to prepare the arrival of the DR3. We followed the method described in CG18. The algorithm has been developed from scratch in Python programming language and using existing Astronomy and data-mining modules that will be described in the following sections. We have tested how it performs in some selected regions of the sky, accounting for different density regions along the Galactic disc. These regions contain field stars simulated by the *Gaia* Object Generator (GOG) [7], in which we added simulated open clusters to check how the algorithm performed given a pair of hyper-parameters: the size of the square region in which we search, L, and the minimum number of neighbours we consider to conform a cluster, $minPts$.

II. DATA PROCESSING

The first step we perform is the preparation of the data that is going to be used. This is an essential step in any data-mining task. We obtained the simulated cluster data from the GOG. For each cluster, we have information for each of their member stars corresponding to:

- Position (x, y, z) in [pc].
- Velocity (u, v, w) in [km/s].
- Visual Magnitude ($vMag$).
- Colour index ($V - I$).
- Spectral Type (SpT).

This serves us to obtain the data we want in a more suitable format. Using the Astropy module [9], we transform the position and velocity to the ICRS coordinate system, the same system used in the GOG. After this transformation, we calculate the observational errors using the PyGaia package [10], in order to recreate how the *Gaia* telescope would observe these sources at the time of the DR3.

Once we have the data from clusters correctly stored, we download from the GOG a rectangular region in the ICRS coordinate system, containing the Galactic center. We will later add to this region, containing simulated field stars, the clusters we obtained previously, and then we will proceed to search clusters in different sub-regions, storing the results as a function of the input hyper-parameters. In the next section we give a detailed description of this method.

III. METHODOLOGY

Following the methodology developed by CG18, we have used the clustering algorithm DBSCAN [11] to search for clusters using a dataset where we add the sixth parameter RV added. DBSCAN has two input hyper-parameters, $minPts$ and ϵ, and once they are given it can search for over-densities in the six-parameter space. The algorithm starts on one source of a $(L \times L)$ region and calculates the euclidean distance in the six-dimensional space between the source and every other source. If the distance is less than the value (ϵ), it considers the two sources neighbours. When the algorithm ends the computation for a single star, if the number of neighbours counted is greater or equal than $minPts$, the star is considered to be a core member and all the neighbours will be labeled as members. The algorithm will jump to one of those neighbours and run the same process. When the algorithm reaches a source where it cannot find at least $minPts$ neighbours, and there is no more unexplored members, it randomly jumps to another source and searches a new cluster.

Figure 1 shows a sketch of how DBSCAN works in a two-dimensional space. Core stars are in red, the ones that verify the $minPts$ condition, considered to be the core points of the cluster. In yellow, boundary stars that have been clustered but have less than $minPts$ in their ϵ-neighbourhood. We consider both core and boundary as cluster members. Finally, the purple stars are the outliers, field stars that are not clustered.

Once the neighbours of the last star have been computed, all the stars are labeled either with -1, which means they are considered field stars, or with a number equal or greater than zero, corresponding to each of the clusters found. In Figure 2, we can see a region of sky before and after running the DBSCAN algorithm.

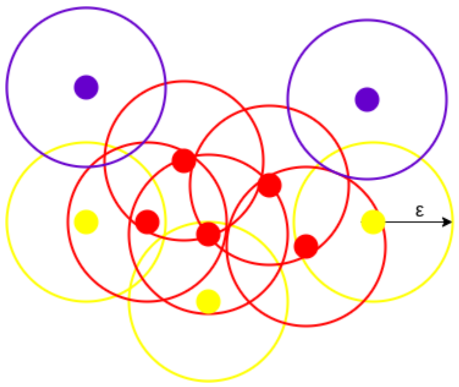

FIG. 1: DBSCAN search in a dataset [12]. Red dots are core points of the cluster, yellow dots are in the neighbourhood of the cluster but have not enough points to be a core member, and blue ones are not in the cluster.

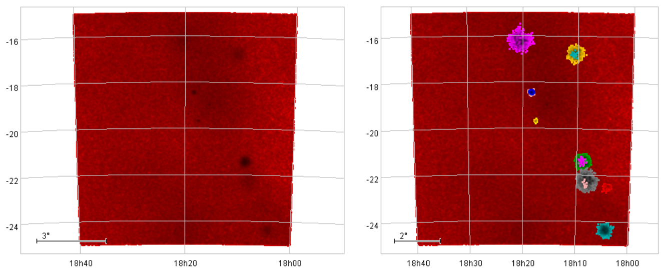

FIG. 2: TOPCAT [13] plot of a region of the sky $\alpha =$ (18h00m, 18h40m), $\delta = (-25°, -15°)$ before and after applying the clustering algorithm with parameters $(L, N) = (10°, 4)$. The algorithm found 16 clusters in total.

In the left plot we can clearly see over-densities in the darker spots, and these are confirmed in the right plot with the clusters identified in a different colour. In Figure 3 we plot the stars corresponding to the added clusters, the grey ones are the undetected and in blue the correctly clustered. In addition, there are a few red dots that show the field stars that were clustered.

To reduce the free hyper-parameters that we need to give to DBSCAN, we automatically determine which ϵ value will be chosen for each region for each $minPts$ value, as done in CG18. We compute the k_{th} Nearest Neighbour (kNN) for every field star of a region of the sky first. Then, we do the same, including the simulated clusters and calculating the distance for their stars too. We choose the ϵ parameter by averaging the minimum k_{th} distance from both regions. Once we choose an input $minPts$ value for the DBSCAN, and knowing that $k = minPts - 1$, we can calculate the input ϵ value.

We will run the algorithm for $minPts \in [4, 12]$. For each value, we will also evaluate different sizes of region, in order to assess the effect of having different number of clusters in a region. The values will be $L \in [8°, 16°]$. In the following section the processed results obtained by

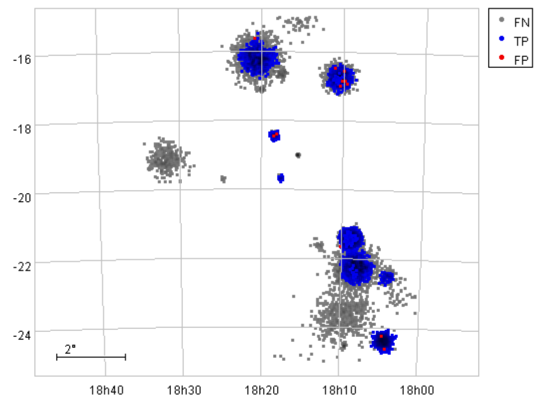

FIG. 3: TOPCAT [13] plot of the same region, showing the stars from simulated clusters in grey if they have not been detected and in blue if they are correctly clustered. We also see the field stars that were clustered in red.

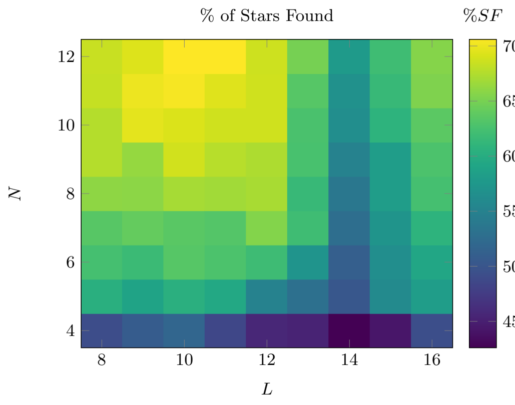

FIG. 4: Matrix plot with a heat map of the percentage of correctly clustered stars by every pair of parameters $(L, minPts)$.

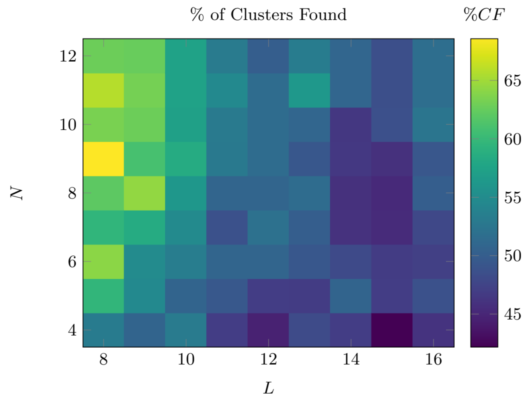

FIG. 5: Matrix plot with a heat map of the percentage of clusters found by every pair of parameters $(L, minPts)$.

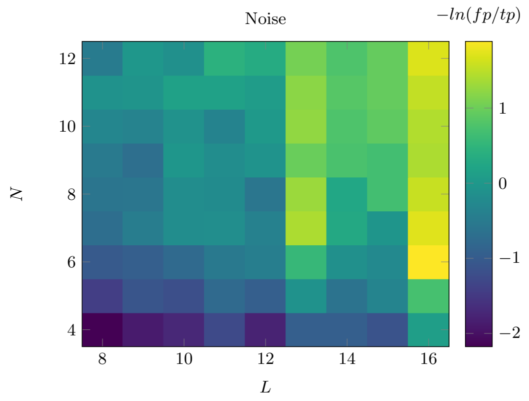

FIG. 6: Matrix plot with a heat map of noise (the false positives-true positives ratio in logarithmic scale) captured by every pair of parameters $(L, minPts)$. The pairs with less noise are in yellow.

each pair of parameters $(L, minPts)$ are shown, as well as a summary of their performance.

IV. RESULTS

Once the analysis has finished, we store our results and proceed to run post-processing scripts to the data. First, we will show the performance of this method depending on the choice of parameters $(L, minPts)$, and then we will show the properties of a selection of clusters, the times they were correctly clustered and the percentage of the stars that were correctly clustered.

In Figure 4, we can see the percentage of stars that were correctly clustered per each $(L, minPts)$. The plot shows that increasing the number of neighbours implies finding more clustered stars, as the radius of the neighbourhood will increase. We can also see that the size of the box considered changes its performance. For smaller sizes, we can study better the inhomogeneity of

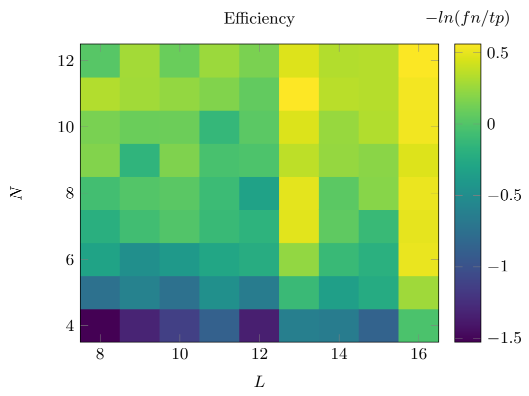

FIG. 7: Matrix plot with a heat map of the efficiency (the false negatives-true positives ratio in logarithmic scale) of every pair of parameters $(L, minPts)$. The pairs with better efficiency are in yellow.

Cluster	α [deg]	δ [deg]	ϖ [mas]	μ_α [mas·yr^{-1}]	μ_δ [mas·yr^{-1}]	RV [km/s]	Stars	%Found	%Stars
NGC6604	274.50 (0.04)	-12.25 (0.04)	0.59 (0.01)	-0.36 (0.19)	-2.68 (0.2)	9.66 (2.21)	2154	100	99.6
NGC6530	271.11 (0.13)	-24.36 (0.12)	0.76 (0.01)	0.58 (0.25)	-2.10 (0.25)	-6.36 (1.71)	630	100	97.8
vdBergh113	272.21 (0.13)	-21.42 (0.12)	0.29 (0.01)	-0.26 (0.1)	-2.47 (0.1)	-16.18 (44.07)	2638	100	88.4
Trumpler27	264.08 (0.07)	-33.52 (0.06)	0.83 (0.01)	-0.93 (0.27)	-0.56 (0.29)	-15.77 (5.46)	340	100	87.4
Ruprecht141	277.82 (0.04)	-12.32 (0.04)	0.18 (0.01)	1.37 (0.06)	-4.67 (0.06)	31.53 (57.25)	746	100	85.8
NGC6242	253.90 (0.11)	-39.46 (0.08)	0.89 (0.01)	0.39 (0.33)	-0.12 (0.3)	9.77 (5.07)	118	100	77.4
Trumpler26	262.14 (0.04)	-29.50 (0.04)	0.36 (0.01)	-3.51 (0.12)	-0.01 (0.12)	13.30 (32.98)	172	100	76.5
NGC6250	254.48 (0.12)	-45.93 (0.1)	1.16 (0.01)	-0.18 (0.4)	-3.33 (0.4)	10.25 (3.47)	186	100	76
NGC6167	248.63 (0.09)	-49.77 (0.06)	0.90 (0.01)	-0.84 (0.28)	-4.53 (0.3)	-35.69 (4.71)	66	100	75.7
NGC6249	254.39 (0.05)	-44.82 (0.04)	1.02 (0.01)	2.19 (0.27)	-5.06 (0.37)	9.89 (4.97)	56	100	75.2
ASCC88	256.68 (0.69)	-35.61 (0.55)	0.53 (0.01)	2.89 (0.18)	-2.00 (0.18)	1.73 (17.04)	7934	100	74.8
NGC6475	268.49 (0.81)	-34.87 (0.71)	3.33 (0.04)	1.64 (1.14)	-3.60 (1.18)	-14.84 (1.99)	285	0	0
Alessi9	266.08 (2.12)	-47.19 (1.34)	5.12 (0.12)	11.69 (1.76)	-9.67 (1.67)	-11.64 (2.1)	223	0	0
ASCC94	273.84 (0.27)	-15.02 (0.24)	1.18 (0.01)	0.34 (0.36)	-0.15 (0.38)	-25.92 (2.13)	82	0	0
ASCC90	264.85 (0.48)	-34.75 (0.32)	2.00 (0.02)	-2.42 (0.62)	-3.74 (0.7)	10.02 (1.68)	40	0	0
NGC6514	270.65 (0.23)	-23.02 (0.23)	1.23 (0.01)	2.74 (0.4)	-0.85 (0.48)	-1.60 (2.79)	38	0	0
NGC6613	274.99 (0.04)	-17.10 (0.04)	0.77 (0.01)	-0.92 (0.29)	-1.26 (0.26)	-4.28 (7.93)	31	0	0
NGC6546	271.87 (0.15)	-23.34 (0.11)	1.06 (0.01)	-2.46 (0.4)	-0.64 (0.38)	-16.37 (2.31)	27	0	0

TABLE I: List of a selected group of clusters with their mean parameters and their standard deviation. The first set of clusters shows the clusters with the highest percentage of stars found. The second set of clusters shows some clusters that have not been found.

the sky, allowing less dense cluster to be more noticeable. This is shown in Figure 5, a matrix plot that shows the percentage of clusters found out of the total we added.

This does not mean that smaller sizes perform better. We have to take into consideration the field stars that were clustered, the false positives. Having low size of box adds a lot of noise, as we can see in Figure 6. Low number of neighbours also show an increase of noise, since a few field stars with similar parameters end up clustered. Overall, the best performing pairs are around $L = 13°$ and $L = 16°$, as we can see in Figure 7, where the efficiency is computed as the division of the false negatives rate by the true positives rate. The rate of false negatives is the percentage of clusters not found out of the ones added, while the rate of true positives is the percentage of correctly found clusters out of the total found. These pairs may not give all the possible clusters that can be found, but the results given have less false positives (the percentage of clusters formed only by field stars out of all the clusters found) while they still have good levels of clusters and stars found.

To understand these results, we need to take into account three main factors that can make a cluster go unnoticed (or less defined):

- The most obvious one, how clustered it is in all of its 6-parameters. If the cluster has very low dispersion in the parameters when compared to the field stars, it is more likely that it will be found. When the dispersion of its parameters is high, member stars end up blending with the ones outside of the cluster.
- The amount of stars forming it, since more stars in a cluster mean more potential neighbours. In some extreme cases, the cluster has less stars than the minimum required, meaning that it does not matter how clustered it is, DBSCAN will not find it. We have discarded these cases, but some small clusters that barely make the cut are not found and counted as a false negative.
- The last one is the presence of very dense cluster(s) in a region. Since the radius of the neighbourhood is determined as described Section III, if there is a very dense cluster in a region, the ϵ will be low. This means we are limited to find the most dense clusters in the region. We explore several L values to check the behaviour of the algorithm. A lower size of the box helps to reduce this effect, being more probable to isolate the small clusters and lowering the chance of having dense clusters in the same box. In exchange, the noise is greater than for a larger box size, as we show in Figure 6.

In Table I, we can see the average performance in a subset of clusters. We have chosen the most defined (the ones with greater % Stars Found) clusters for the first subset, and the biggest not found clusters for the second. At first glance we can state that having small position and proper motion dispersion leads to a higher rate of detection. Comparing the α and δ dispersion of the first and the second subset, its clear that high dispersion ends up with the cluster not being found.

If we compare the ASCC 88 (the biggest cluster) with ASCC 94, we can see that while they have similar dispersions, the first one is always detected and the second one never. If we check in the table, we can see that ASCC 94 ($\alpha = 273.84°, \delta = -15.02°$) is very close to NGC 6604 ($\alpha = 274.50°, \delta = -12.25°$), the most defined cluster. This means that ASCC 94 is eclipsed by the pressence of a denser cluster in his region, and the ϵ value computed there is smaller enough to not detect it.

Comparing now ASCC 88 with NGC 6604, the best defined cluster, we see the effect of the dispersion in position. We almost find every single star in NGC 6604, while we leave 25% of ASCC 88 member stars unfound. For the first one, we can also see a radial velocity dispersion almost ten times greater than in NGC 6604, which also reduces the density of the first cluster in the six-parameter space.

Radial velocity dispersion do not seem to affect the result as much as the other parameters. We can compare some clusters that only differ significantly in RV dispersion to find that the definition of the cluster is affected by it. For example, if we compare NGC 6604 with Trumpler 26, we can see a case where all the positional and proper motion dispersion are almost equal, but RV dispersion is 15 times bigger in Trumpler 26 case, leading to a difference in resolution of almost 25% of the percentage of stars found. Another example is NGC 6530 vs vdBergh 113, the second one having equal or less dispersion in all of its parameters except for RV, where it has more than 20 times the dispersion of NGC 6530. Despite having much more stars, the percentage of stars found is almost 10% lower.

V. CONCLUSIONS

We have implemented a method that is capable of finding clusters using the DBSCAN algorithm. It has been tested in a Galactic field simulation of a region of the sky with simulated clusters added. The algorithm has performed a search in the sub-regions sized $(L \times L)°$, calculating the distance between stars in the six-dimensional space of the parameters (α, δ, ϖ, μ_α, μ_δ, RV) and clustering them in groups of at least $minPts$ stars.

The hyper-parameter $minPts$ is the minimum required neighbours to form a cluster, and also defines the ϵ radius that is considered to be the neighbourhood of each star. This radius is obtained by performing the mean between the distance of the kNN of the smooth region of the sky and the distance of the kNN of the region with the clusters added.

We have seen after processing the results how this design hyper-parameters $(L, minPts)$ affect the performance, obtaining from 45% to 70% of the clusters correctly identified, and similar percentages of correctly clustered stars out of the added from the simulation. We checked how parameter dispersion affected to the results, in order to understand why there were some better performing pairs. The best efficiency was obtained for high values of L and $minPts$, obtaining a bit worse results in terms of amount of clusters found but having much less noise than other parameters. The best pair was $(13°, 11)$, although several pairs with similar efficiency should be considered in a real search in order to account for the limitations of this work.

This work has been developed under some limitations, mainly computational. Since we only consider a region of the sky, results may be biased towards similar populated regions, considering we chose the center and part of the disc of our Galaxy. This can be improved using more computational power to cover bigger regions

Acknowledgments

I want to thank my advisors during the development of this paper, Carme Jordi and Alfred Castro-Ginard. They helped me to keep a good pace during the whole process, not needing to rush too much even when I was really busy the last months. They also shared Alfred's publications so I could develop my own implementation of the problem inspired by his previous works.

[1] European Space Agency *Gaia. The mission.* [URL: https://www.esa.int/gaia]
[2] Gaia Collaboration et al. *Gaia* DR1 (2016).
[3] Gaia Collaboration et al. *Gaia* DR2 (2018).
[4] Gaia Collaboration et al. *Gaia* EDR3 (2021).
[5] Gaia Collaboration et al. *Gaia* DR3 (2022).
[6] Castro-Ginard et al. *A new method for unveiling Open Clusters in Gaia* (2018). [arXiv: 1805.03045]
[7] Masana E. et al. *Gaia* Object Generator (2014).
[8] Katz et al.2019*Gaia DR2: Properties and validation of the radial velocities.* [https://doi.org/10.1051/0004-6361/201833273]
[9] Astropy Collaboration et al. *The Astropy Project v2.0* (2018). [DOI:https://doi.org/10.1051/0004-6361/201833273]
[10] Gaia Project Scientist Support Team and Gaia Data Processing and Analysis Consortium. *PyGaia v2.2*
[11] Ester, Martin et al. (1996)*A density-based algorithm for discovering clusters in large spatial databases with noise.*
[12] medium.com *DBSCAN algorithm for fraud outlier detection in a data set.*
[13] Taylor, M.B. *TOPCAT & STIL: Starlink Table/VOTable Processing Software* (2005). [2005ASPC..347...29T]