A software package for astrometric memberships calculation in open star clusters research

Simón Pedro González[1], Myriam Herrera[1], Susana Ruiz[2]

[1] Instituto de Informática, Facultad de Ciencias Exactas, Físicas y Naturales, Universidad Nacional de San Juan, San Juan, Argentina
[2] Departamento de Geofísica, Facultad de Ciencias Exactas, Físicas y Naturales, Universidad Nacional de San Juan, San Juan, Argentina

Abstract. One of the first steps in open star cluster studies is members identification. For each star in the sample, researchers need to know the probability that it actually belongs to the cluster. In the project portrayed in this paper, the problem of membership calculation is approached from a Data Science perspective, and a solution is being designed and implemented as a Python software package named Opencluster. The aim is of this package to make star cluster exploratory analysis faster and easier for individual researchers.

Keywords: Data Science, Scientific Software, Stellar Cluster Memberships.

1 Introduction

Since the lauch of the GAIA (Global Astrometric Interferometer for Astrophysics) space probe in 2013, the ESA (European Space Agency) GAIA mission releases data in catalogues of increasing accuracy and completeness. According to the GAIA Collaboration (2020), the third Data Release (DR3) is expected to be complete by 2022, and an Early Data Release (EDR3) catalogue has been released in December 2020. Anticipating each release, the scientific community presents different approaches for data processing and analysis.

In this context, a free and easy-to-integrate package for cluster membership probability calculations with astrometric data can facilitate exploratory analysis. In this work, a practical, implementation-oriented perspective is taken, as to mitigate some of the difficulties scientific software development portrays. To begin with, more often than not, software development is not the main focus of the research, and therefore not much time is spent to ensure desiderable characteristics such as testability, usability, code readability and efficiency. Secondly, developers are generally problem domain experts, but have limited knowledge regarding software engineering and programming techniques.

Section II describes the fundamentals of the open star cluster membership problem and presents a brief background review of both analytic methods and software implementations. In Section III, the progress of the Opencluster project is reported, and some preliminary conclusions are given in Section IV.

2 Background

2.1 Problem Statement

Star clusters are collections of stars formed from large clouds of dust and gas that collapse under the action of gravity (Stott, 2018). During the formation of young, massive stars, a large amount of gas is expelled, forming more stars that become gravitically associated with each other, and present similar composition and age. Sets of stars formed in this way are known as open clusters.

In a large sample , a significant number of stars do not belong to the cluster and present more extended parameter distributions, the so called field stars. Distinguishing between cluster stars and field stars with a probabilistic approach is of utmost importance for further studies on the structure, origin and evolution of a cluster.

From a Data Science perspective, and with a focus on unsupervised Machine Learning algorithms, the problem of star cluster membership calculation is of particular interest due to the challenges it poses. Distinguishing star clusters from field stars is not straightforward: the ratio of the number of cluster stars to the number of field stars in the sample can be variable, and this is a critical factor that can affect the results of many methods. In addition, from a celestial coordinates standpoint, it is possible to find clusters of different sizes, shapes and densities, which makes their identification more difficult. Choosing the right approach (parametric or non-parametric) and appropriate distributions for modelling star populations (in the parametric case) is a non-trivial task, as different approaches have different limitations and applicability conditions. Moreover, it is very important for algorithms to properly account for accuracy measures, so as to provide confidence estimates of the results. Finally, the treatment of missing data must also be considered, so as to avoid biases.

2.2 Background on astrometric membership estimation methods

One of the main approaches to the astrometric membership problem using unsupervised learning techniques consists of modelling the field and the cluster in the proper motion variable space, using a maximum likelihood parametric fit (Sanders, 1971).

Cabrera-Caño & Alfaro (1990) analysed Sanders' method, and found that it works correctly only if the ratio between the number of cluster stars and field stars in the sample is sufficient. They also proposed two alternative solutions, a parametric method (Cabrera-Caño and Alfaro, 1985) and a non-parametric one (Cabrera-Caño and Alfaro, 1990). In the non-parametric method, the distributions are computed using Gaussian kernel functions. This allows a priori assumptions to be less restrictive.

Sampedro and Alfaro (2016) analysed previous methods and proposed a new one, the geometric method. It is based on a parametric modelling of star populations using as input the euclidean distance calculated from the different variables chosen. The geometric method generally outperforms previous methods, being able to provide valid clustering in cases where the previous methods could not. However, it was concluded

that some limitations are common to all of them: the sample taken needs to be centred in the cluster and restricted to the pre-calculated cluster radius. Also, the ratio of cluster stars to field stars must be adequate.

In recent years, Gaussian mixture models combined with the Expectation-Maximization algorithm have become widespread as a generalised solution to this type of problem. One example of the application of such methods can be seen in Olivares et al. (2018) paper. However, this approach is not without flaws. When populations do not naturally fit normal distributions, the models obtained do not adjust well to the data unless they are very complex, with many free parameters. As a result, they tend to require significant computational resources to fit, which makes them impractical in some cases.

2.3 Background on free open source software for studying open clusters

In recent years, significant progress has been made regarding free scientific software for star clusters research. Krone-Martins & Moitinho (2014) developed UPMASK, an R package able to compute membership probabilities based on photometric data. Another package worth mentioning is ASTECA, under development by Perren, Vázquez & Piatti since 2015, which can be used to calculate the star cluster parameters, among other tasks. In 2019, Lei Liu and Xiaoying Pang presented SHiP (Star cluster Hunting Pipeline), a set of Python 2.7 scripts integrated with MPI that can identify clusters using data files from the second GAIA release. More recently, Balaguer-Núñez, López del Fresno, Solano, et al. (2020) developed CLUSTERIX 2.0, an interactive web-based virtual observatory for proper motion-based membershipe determination.

Unlike CLUSTERIX and SHiP, UPMASK and ASTECA are designed as software packages, which offers certain advantages. Opencluster is intended to have a similar format, so that it can be easily installed, used, extended, and integrated into Python workflows. In terms of functionality, Opencluster differs from UPMASK and ASTECA in that it works on astrometric data, including proper motions, parallax and spatial coordinates. Finally, unlike SHiP, Opencluster focuses on the problem of membership calculation, not detection, hence the methods used differ.

3 Progress

3.1 Considerations on the software design process

According to Oliveira and Stewart (2006), priorities regarding scientific software design are the following: software correctness, numerical stability, discretisation accuracy, flexibility and finally, CPU and memory efficiency. To the former list, it is worth adding usability. The code must be easy to use, even if users have little Python knowledge. Sufficient documentation must be provided to facilitate understanding of the different functionalities, how they work and their syntax.

To address software correctness, one of the most convenient approaches is to use Test-Driven Development (TDD). In this technique, each test case is written prior to implementing the functionality it is associated with, so that the developer ensures that each component is testable and correct. For Opencluster development, TDD was adopted, using the Tox environment manager, Pytest and the automatic deployment service Travis.

To provide software flexibility, a modular design was chosen and rigorous style controls were implemented using Flake-8, Black and Pydocstyle. In terms of discretization accuracy and numerical stability, it is worth mentioning that most of the operations are done using SciPy and Numpy, which have been tested for accuracy and numerical stability. Consistency and correctness their use in Opencluster is accounted for when testing. Finally, basic efficiency issues have been addressed through good coding practices. More advanced aspects, such as code pre-compilation, sparse-matrix use and processing parallelization, will be considered at a later stage, as deemed necessary.

3.2 Data collection and storage

In Opencluster, data is retrieved using the Gaia Astroquery package, which allows access to the European Space Agency Gaia Archive, through the Gaia TAP+ service. Functions were implemented to easily build queries, download, save and read data in VOTable format, which achieves a higher compression ratio than other Astroquery supported formats such as CSV, FITS and JSON. Reading and querying speed has not been measured, and the possibility of optimizing data persistence related operations by using other formats, such as parquet, remains to be explored.

3.3 Outlier detection

Outlier detection is critical, especially when parametric fits are to be applied, because they interfere in the calculation of the parameters, resulting in unrealistic solutions. Selecting outliers involves some subjectivity because there are different ways of defining them. Cabrera-Caño and Alfaro (1985) defined outliers as those objects located in regions of low probability density, and described a method for outlier detection called OUTKER. This non-parametric method, applied on the proper motion in right ascension and declination, was implemented in Opencluster. However some changes regarding the method were included, for instance, proper motions measurement errors are used to define dynamic bandwidths for each of the Gaussian bells used to approximate the distributions, instead of using a single fixed bandwidth, like in the original method. This is expected to provide a more accurate Probability Density Function (PDF) approximation.

4 Preliminary Conclusions

This work reports the status of the Opencluster project, a Python-based software package for membership probabilities computation in open star clusters. The software is

under development: the main calculation algorithm is yet to be implemented, but the package already includes features for local and remote data management and outlier detection using OUTKER. It is hoped that Opencluster will be useful for open star clusters exploratory analysis, as there are no package with its exact features available to date. The software progress can be tracked though the following repository link: https://github.com/simonpedrogonzalez/opencluster.

References

1 Author, F.: Article title. Journal 2(5), 99–110 (2016).
2 Author, F., Author, S.: Title of a proceedings paper. In: Editor, F., Editor, S. (eds.) CONFERENCE 2016, LNCS, vol. 9999, pp. 1–13. Springer, Heidelberg (2016).
3 Author, F., Author, S., Author, T.: Book title. 2nd edn. Publisher, Location (1999).
4 Author, F.: Contribution title. In: 9th International Proceedings on Proceedings, pp. 1–2. Publisher, Location (2010).
5 LNCS Homepage, http://www.springer.com/lncs, last accessed 2016/11/21.
6 E.-M. Arvanitou, A. Ampatzoglou, A. Chatzigeorgiou et al., Software engineering practices for scientific software development: A systematic mapping study. The Journal of Systems & Software (2020), DOI: https://doi.org/10.1016/j.jss.2020.110848
7 Gaia Collaboration (2020). Gaia early data release 3. URL https://www.cosmos.esa.int/web/gaia/newsletter/contents#GaiaNewsletter14
8 Stott, J. J. (2018). Determining open cluster membership: a bayesian framework for quantitative member classification. Astronomy and Astrophysics, 609, A36.
9 Kelleher, J.D., Tierney, B. (2018). What is Data Science?, MIT Press, 1-38.
10 Sanders, W.L. (1971). An improved method for computing membership probabilities in open clusters. Astronomy and Astrophysics, 14, 226-232.
11 Cabrera-Caño, J. y Alfaro E. J. (1985). Analysis of relative proper motion - an improved method to assign membership probabilities in open clusters. Astronomy and Astrophysics, 150, 298-301.
12 Cabrera-Caño, J. y Alfaro E. J. (1990). A non-parametric approach to the membership problem in open clusters. Astronomy and Astrophysics, 235, 94-102.
13 Sampedro, L. M., y Alfaro E. J. (2016). Caracterización de Sistemas Estelares en Espacios de N-Dimensiones: Simulaciones y Aplicación al Catálogo Astrométrico UCAC4. (Doctoral thesis). Universidad de Granada, Granada, España.
14 Cabrera-Caño, J. y Alfaro E. J. (1990). A non-parametric approach to the membership problem in open clusters. Astronomy and Astrophysics, 235, 94-102.
15 Olivares, J. et al. (2018). The seven sisters DANCe - IV. Bayesian hierarchical model. Astronomy and Astrophysics, 617, A15.
16 Krone-Martins, A. y Moitinho, A. (2013). UPMASK: Unsupervised photometric membership assignment in stellar clusters. Astronomy and Astrophysics, 561, 10. DOI: 1051/0004-6361/201321143.
17 G. I. Perren, R. A. Vázquez, A. E. Piatti, (2015). ASteCA: Automated Stellar Cluster Analysis. Astronomy and Astrophysics, 576, A6. DOI: 10.1051/0004-6361/201424946.
18 Lei Liu and Xiaoying Pang 2019 The Astrophisics Journal Supplement Series, 245, 32.
19 Balaguer-Núñez L., López del Fresno M., Solano E., et al. (2020) Clusterix 2.0: a virtual observatory tool to estimate cluster membership probability, Monthly Notices of the Royal Astronomical Society, vol. 492, 5811–5843, DOI: https://doi.org/10.1093/mnras/stz3610

20 Oliveira S., Stewart D. (2006) Writing Scientific Software: A Guide For Good Style, Cambridge University Press.
21 Tox Development Team. URL https://tox.readthedocs.io/en/latest/
22 Pytest Development Team. URL https://docs.pytest.org/en/6.2.x/
23 Travis Development Team (2018). URL https://travis-ci.com/
24 Flake8 Development Team (2016). Flake8: Your Tool For Style Guide Enforcement. URL https://flake8.pycqa.org/en/latest/index.html
25 Black Development Team (2018). Black: The uncompromising code formatter. URL https://black.readthedocs.io
26 Pydocstyle Development Team (2020). URL http://www.pydocstyle.org/en/stable/
27 Charles R. et al. (2020) Array programming with NumPy, Nature, 585, 357–362. DOI:10.1038/s41586-020-2649-2
28 Virtanen, P. et al. (2020) SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17(3), 261-272.