Unsupervised Identification of Open Star Clusters in the Galactic Anticenter with Gaia DR3

Michael Struk

I. INTRODUCTION

The goal for this project is to identify open star clusters in a galactic anticenter region of the Milky Way using unsupervised machine learning techniques. Open clusters are loosely bound groups of stars that formed from the same molecular cloud, often found near the galactic plane where star formation is ongoing. Thus, their member stars are typically similar in composition, distance, kinematic motion, and age. Most are young stars on the main sequence of a Hertzsprung-Russell or Color-Magnitude Diagram (CMD).

The dataset is queried from the Gaia Data Release 3 (DR3) using Astronomical Data Query Language (ADQL). Galactic coordinates are used as initial input to ensure the field is in the desired portion of the galaxy:

$$\text{Galactic Longitude} \quad l \in [210, 230]^{\circ}$$
$$\text{Galactic Latitude} \quad b \in [-10, 10]^{\circ}$$

The longitude measures the angle from the line connecting the Sun and the galactic center (along the plane), and the latitude measures the angle above or below the plane itself. Given the nature of these clusters, the region is centered on the plane; the anticenter region is selected out of convenience to avoid overcrowding from the bulge. These coordinates are converted to the International Celestial Reference System (ICRS) for querying with Gaia. The full query, including the magnitude and distance (parallax) limit, is provided below (Figure 1).

The inputs for the task are the highly accurate astrometric and kinematic measurements from Gaia; these quantities are joined into a multidimensional space, and are what the algorithm will identify clusters with. The primary outputs are the indices of the member points for each cluster, as well as the simple number of unique clusters.

```
# Define total region
l_range = [210, 232] # galactic longitude: approximately anticenter
b_range = [-10, 12] # galactic latitude: centered on the disk

# Define tiling step size (degrees)
l_step = 4
b_step = 3

# Quantity limits
mag_lim = 18
parallax_lim = 0.8 #  d < 1250 pc

# Initialize list to collect tile results
all_results = []

# Tile the full region and query each individually
for l_start in range(l_range[0], l_range[1], l_step): # (210, 214, 218, ...)
    for b_start in range(b_range[0], b_range[1], b_step): # (-10, -7, -4, ...)
        # Define tile
        l_bounds = [l_start, l_start + l_step]
        b_bounds = [b_start, b_start + b_step]
        # Convert tile to ICRS coords
        ra_bounds, dec_bounds = rectangle_gal_to_icrs(l_bounds, b_bounds)
        print(ra_bounds, dec_bounds)
        # Query for tile region
        query = f"""
        SELECT TOP 100000
          source_id, ra, dec, parallax, parallax_error,
          pmra, pmra_error, pmdec, pmdec_error,
          phot_g_mean_mag, phot_bp_mean_mag, phot_rp_mean_mag, radial_velocity
        FROM gaiadr3.gaia_source
        WHERE
          ra BETWEEN {ra_bounds[0]} AND {ra_bounds[1]}
          AND dec BETWEEN {dec_bounds[0]} AND {dec_bounds[1]}
          AND phot_g_mean_mag < {mag_lim}
          AND parallax > {parallax_lim}
          AND parallax IS NOT NULL
          AND pmra IS NOT NULL
          AND pmdec IS NOT NULL
          AND radial_velocity IS NOT NULL
        """
        print(f"Querying tile: l={l_bounds}, b={b_bounds} ...") # print status updates
        # Launch query to Gaia
        job = Gaia.launch_job_async(query)
        result = job.get_results().to_pandas()
        all_results.append(result) # collect results from each tile in all_results list
        print('\ntile complete\n') # print when moving to next iteration
```

Figure 1. ADQL Query to Gaia DR3: a tiling approach is taken to avoid inconsistencies in data acquisition, where step increments of 4 and 3 degrees are taken in the galactic longitude and latitude, respectively. The sample is limited down to the 18th magnitude, with objects no further than 1250 pc (parallax > 0.8 mas). Additionally, the selection includes only objects with real measurements (NOT NULL) of all quantities: right ascension (RA), declination (dec), proper motions (PM), parallax, and radial velocity.

Over the following sections, this paper discusses the specific machine learning methods and algorithms, the diagnosis of the quality of cluster identification, and the physical basis of results.

II. APPROACH & PREPROCESSING

In order to identify unknown groupings, or clusters, of objects in the astrometric spaces, this project employs an unsupervised machine learning approach. The Density-based spatial clustering of applications with noise (DBSCAN) algorithm is particularly useful for this task: it is great for density-based clustering of any shape or size, relatively insensitive to random points and outliers (and identifies them as such), and unlike methods such as K-means, there is no need to pre-define the number of clusters. All of these qualities are ideal for searching a multidimensional feature space for unknown objects with varying appearances. This project utilizes Scikit-learn's implementation of DBSCAN.

After the above query is launched and the data is acquired, we narrow the scope of the field to reduce the computational load (see Figure 2). This project constructs and compares two feature matrices: one including only positional quantities, and the other with both positional and kinematic quantities.

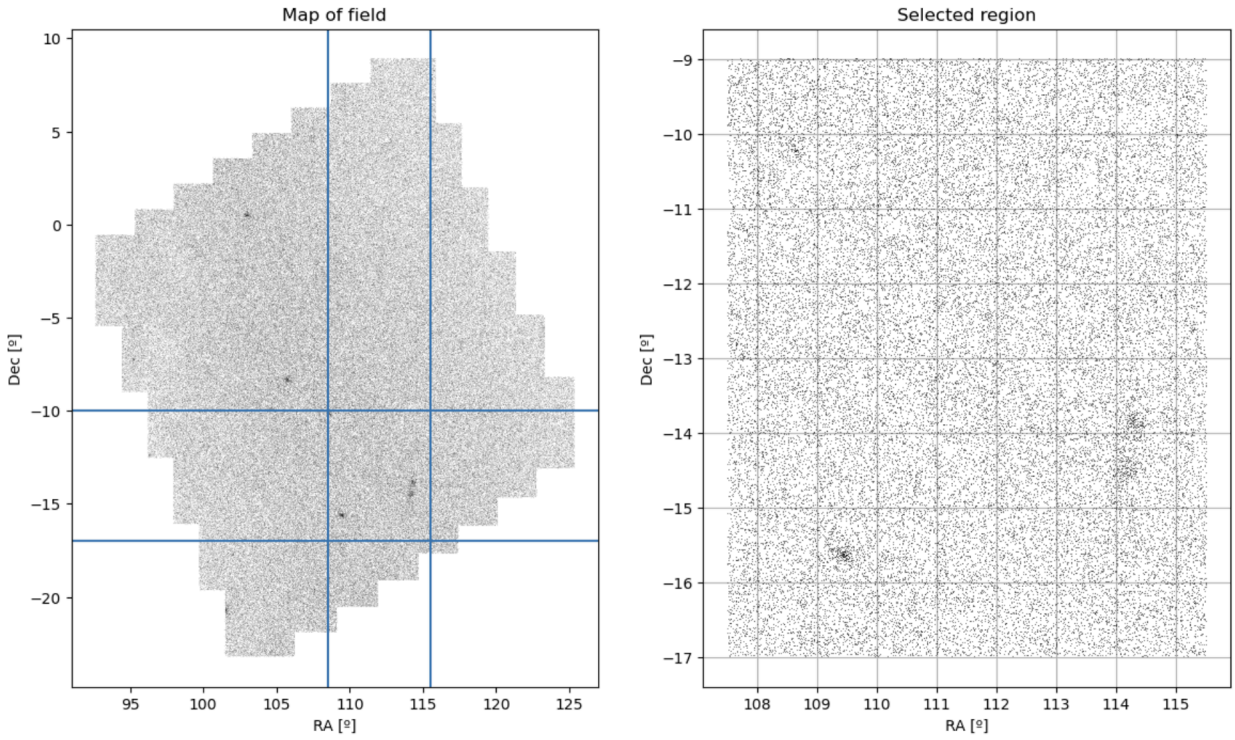

Figure 2. The map of the processed Gaia field in RA/Dec space. The area enclosed in blue lines is plotted to the right, and this is the region of interest for clustering. The full region contains 376,303 objects, and the selected region contains only 42,695. Plotted using Matplotlib's Pyplot library.

II.A. Positions only (features: RA, Dec, Parallax)

The first run is with a feature matrix that contains only the locational measurements angular position (RA, Dec) and parallax, which corresponds to the object's distance from the Sun. Each of them are scaled using Scikit-learn's StandardScaler(), which normalizes the values by their z-score, or number of standard deviations, from the mean, where the mean value is defined to be 0. This ensures that the algorithm won't favor clustering in any particular feature due to the absolute values being larger.

II.B. Kinematics included (features: RA, Dec, Parallax, PMs, Radial Velocity)

The second run is with a feature matrix including the proper motion and radial velocity kinematic quantities in addition to the first set of features. The preprocessing is effectively identical to that of §II.A, with the scaling of parameters. Additionally, though, we perform sigma clipping at the 5 standard deviation threshold on the kinematic parameters. The higher dimensional space can cause computational overexhaustion; since we are using z-score scaling, the range of the feature values is highly dependent on how many standard deviations they span, and sigma clipping helps to avoid the algorithm getting lost.

Within the scope of the algorithm, we tune the *eps* and *min_samples* hyperparameters to adjust the density sensitivity of the clusters. The *eps* hyperparameter defines the maximum distance between two points, or samples, for one to be "in the neighborhood" of the other, meaning points in this distance are considered directly reachable and potentially part of the same cluster. The *min_samples* hyperparameter defines the number of samples in a point's neighborhood to be considered a core point, or a unique cluster; higher values find denser clusters, and lower values make clusters more sparse (Scikit-learn DBSCAN documentation).

The output of DBSCAN is a label for each point indicating its cluster association, or lack thereof (a noise point). This is especially important for (1) calculating the silhouette score, a quality metric that requires excluding noise points, and (2) assessing if the clustered points are physically close, comoving, and on the main sequence.

III. INITIAL RESULTS

Using DBSCAN effectively for this problem requires rooting in both machine learning quality metrics and astrophysical properties for open clusters.

The primary machine learning-based metric is the (average) silhouette score, which is calculated after labeling the cluster assignments. The score for each point is calculated by

$$\text{Silhouette score} = \frac{b_i - a_i}{max(a_i, b_i)}$$

where the cohesion a_i is the average distance to all other points within the same cluster as point i, and the separation b_i is the average distance to all the points in the nearest neighboring cluster for point i. The full score is the average of the score for each point and can provide an overall evaluation of the clustering quality. This score is bounded between -1 and 1: a higher score indicates that the clusters are well-defined, while a lower score can mean that points are poorly clustered due to an inappropriate number of clusters or other data characteristics.

The astrophysical properties of open star clusters, among others, include being lightly bound and approximately in the same location (angular position and distance), having comoving stars with similar proper motions and radial velocities, and being mostly composed of main sequence stars. Analyzing (1) the RA/Dec map of points labeled by cluster association, (2) the distributions of the parallax, proper motions, and radial velocity, and (3) the CMD of cluster stars provides an accurate diagnosis by these characteristics.

III.A.

The default hyperparameters of Scikit-learn's DBSCAN will not create meaningful results for real astronomical data due to the complexity of clusters and the intrinsic, overwhelming noise. Therefore, it is important that we tune these hyperparameters beforehand. The silhouette score is useful to get an initial estimate of them.

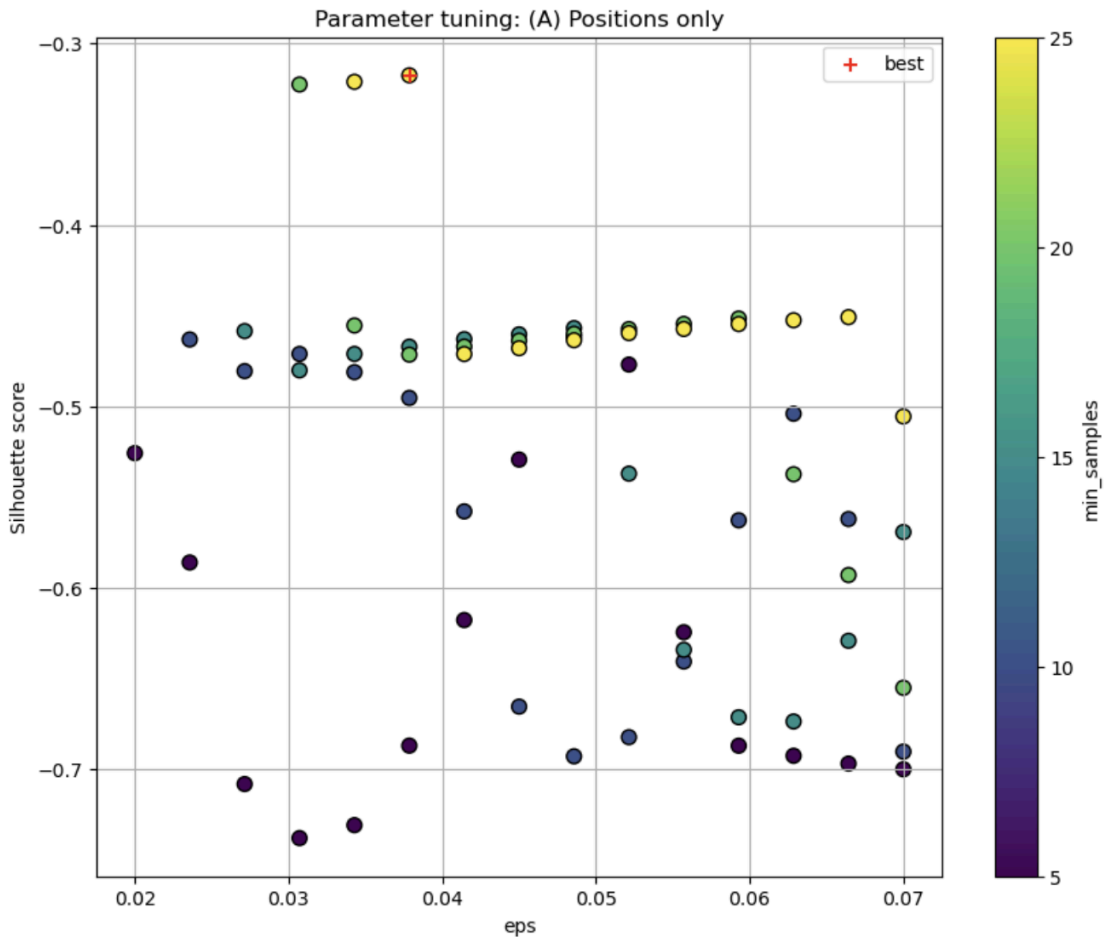

Figure 3. Hyperparameter (*eps*) optimization for A section. A systematic gridsearch with *eps* ϵ [0.02, 0.07] and *min_samples* ϵ [5,25] calculates the silhouette score for 75 combinations of the hyperparameters with DBSCAN. Higher silhouette scores indicate better-defined clusters, so by this metric, the best combination of *eps* and *min_samples* is shown as the tallest point on the graph (marked with a red cross, here). The combination (*eps, min_samples*) identified here is (0.0379, 25), with a silhouette score of -0.3176; this will be the benchmark. The silhouette scores are abnormally negative due to the very high noise point fraction in our set, misrepresenting distances for scoring; the clusters aren't textbook blobs in real astrophysical data.

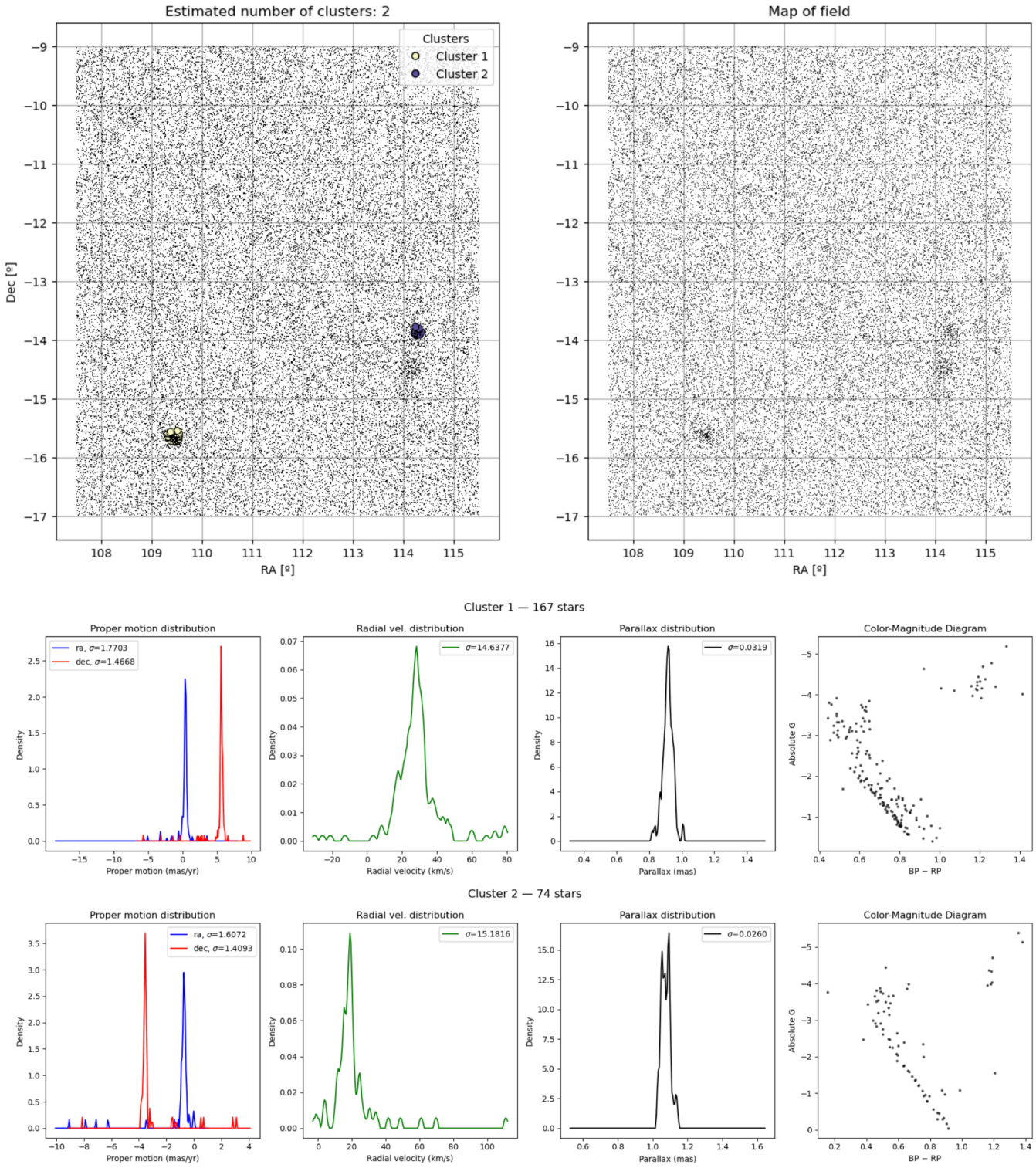

Figure 4. Benchmarking RA/Dec map and diagnostic plots for A section. Using the benchmarking parameters identified in Figure 3, we plot the RA/Dec map, feature distributions, and CMD of the identified clusters. It is clear from the lower sets of plots that the identified stars are cluster members: the vast majority of points the distributions are concentrated around a similar value. The spreads in proper motions and parallax are quite narrow, but the radial

velocities aren't as promising. Additionally, most of the stars lay on or near the main sequence. Having two clusters seems conservative, and visually, there appears to be at least one more significant cluster on the map of the field.

Tuning *eps* with the silhouette score is reliable because *eps* directly controls the separation between clusters, which is exactly what the silhouette score measures. A well-chosen *eps* produces tightly grouped, well-separated clusters that maximize the silhouette score, while an *eps* that is too small or large causes the score to drop sharply. Conversely, *min_samples* controls the density required to form a cluster, not the separation between clusters, so it is less directly tied to silhouette. Therefore, *min_samples* should be re-tuned using astrophysical quality checks like parallax, proper motion, and radial velocity dispersion, rather than relying on silhouette alone. The final chosen value balances achieving small feature spreads while avoiding excessive fragmentation of real physical structures.

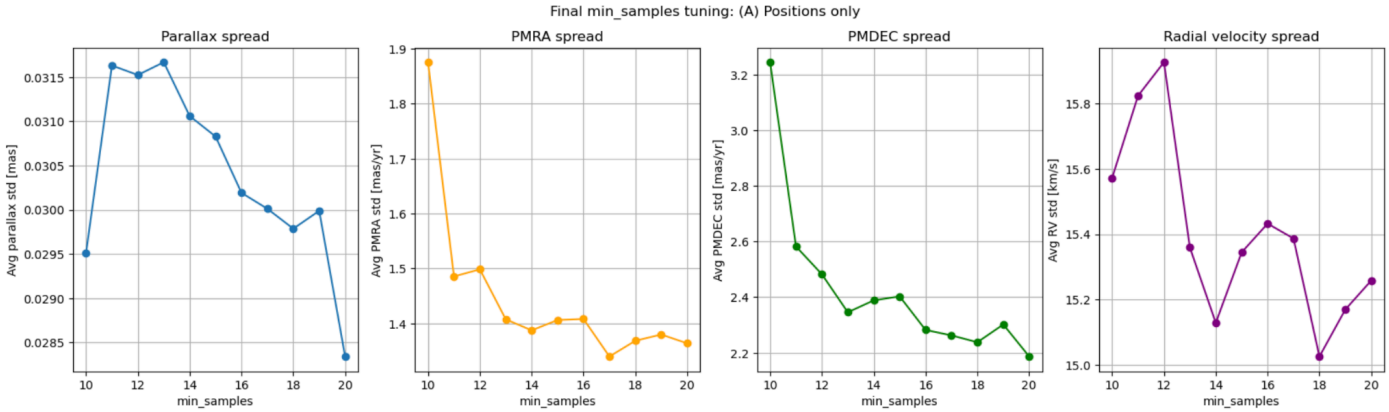

Figure 5. Justification of *min_samples* value for A section. Each of the above plots is the spread, or average standard deviation, among the identified clusters for each measured quantity for 10 values of *min_samples* between [10, 20], holding the previously identified *eps* = 0.0379 constant. These plots provide clarity on how tight the distributions of each non-positional feature are, indicating how closely related the cluster stars are. Though the spreads are collectively the lowest around *min_samples* = 18, we opt to select the second best at *min_samples* = 14 to avoid a distinct double peak in the radial velocity distribution. This is not the lowest for the parallax, but its range of values is so small that it is virtually insignificant.

III.B.

We approach the feature matrix with both positional and kinematic quantities with the exact same process as above (§III.A). The initial results, including the benchmark and hyperparameter optimization, are shown below.

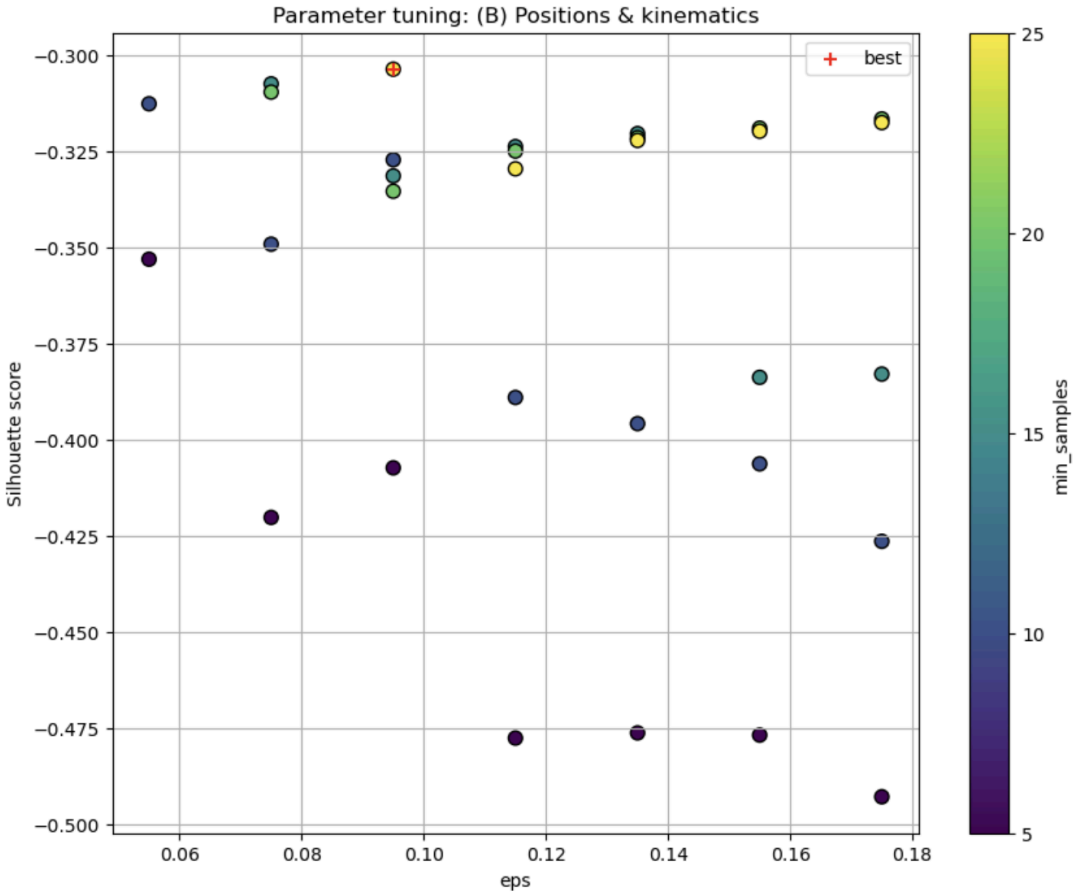

Figure 6. Hyperparameter (*eps*) optimization for B section. A systematic grid search for the feature matrix including kinematic quantities. The silhouette score is calculated across a grid of eps and min_samples values. The best combination (0.095, 25) is identified as the point of maximum silhouette score = -0.3037 (marked by a red cross). As in §III.A, the silhouette scores are affected by the high fraction of noise points, but still guide an initial estimate of clustering parameters.

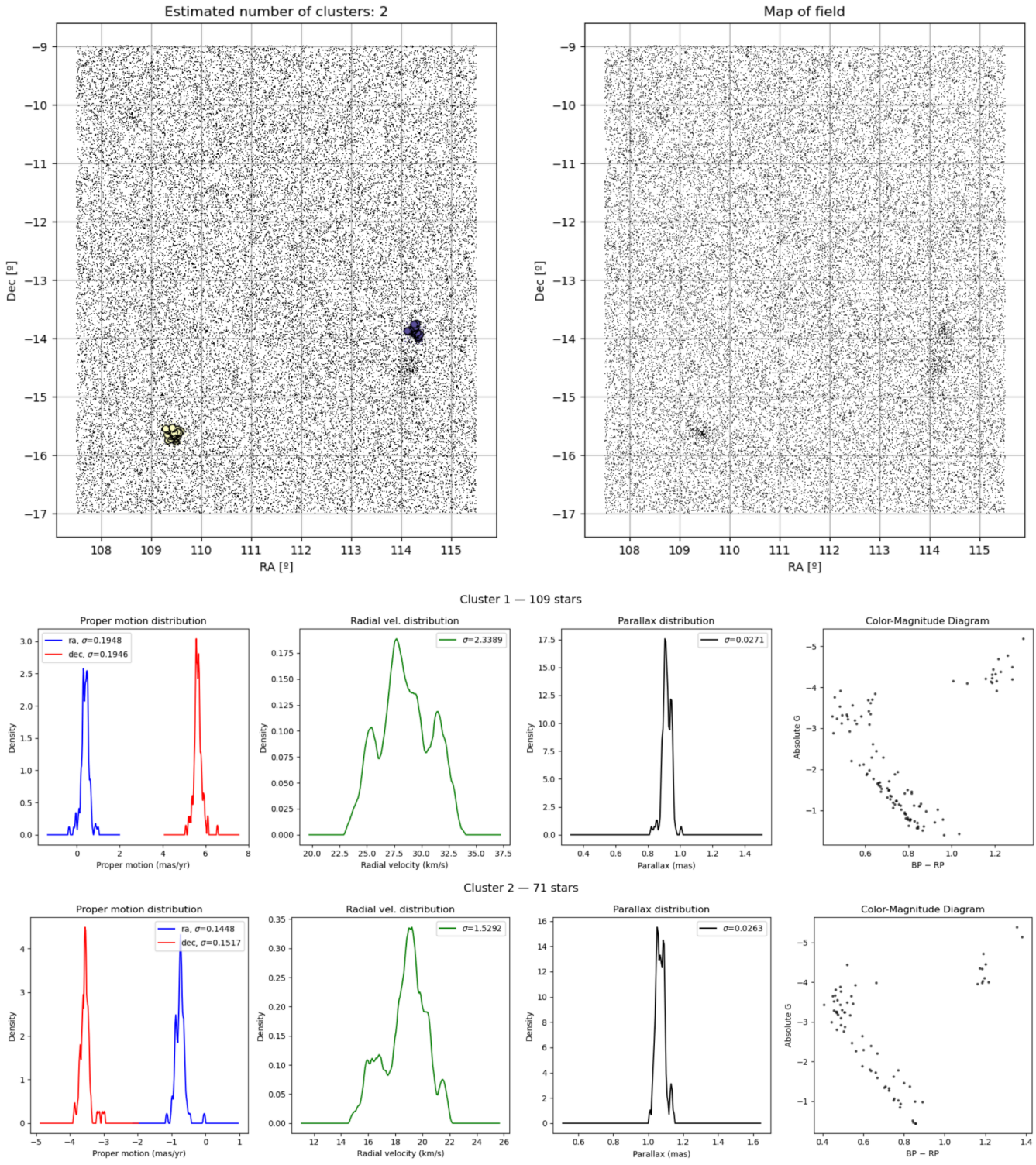

Figure 7. Benchmarking RA/Dec map and diagnostic plot for B section. The RA/Dec map, parallax, proper motion, and radial velocity distributions, and the CMD of the identified clusters using the benchmarking parameters identified in Figure 6. The cluster distributions are more compact than in §III.A as a result of the additional information provided by proper motions and radial velocity, though their shapes are slightly more jagged.

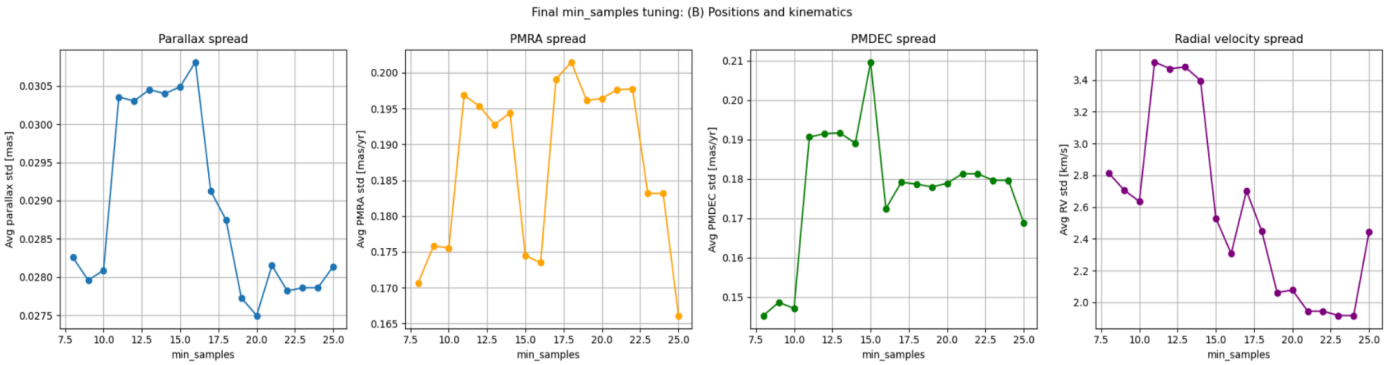

Figure 8. Justification *min_samples* value for B section. Plots of the spread (average standard deviation) of parallax, proper motions, and radial velocity across clusters as a function of *min_samples*. These plots diagnose the tightness of cluster properties and inform the selection of the final *min_samples* value, balancing small spreads and coherent physical grouping. We select *min_samples* = 10 for this set of features, which is the highest value — producing higher-density clusters — before a stark increase in the average spread across all the quantities.

We tuned *eps* and *min_samples* with the silhouette score and analyzed the benchmark results. Then, we included astrophysical logic to refine the value of *min_samples*. The final results are visualized in the following §IV, and discussed in §V.

IV. FINAL RESULTS

Following the hyperparameter tuning and physical validation process outlined in §III.A and §III.B, the final clustering results are presented here for both feature matrices.

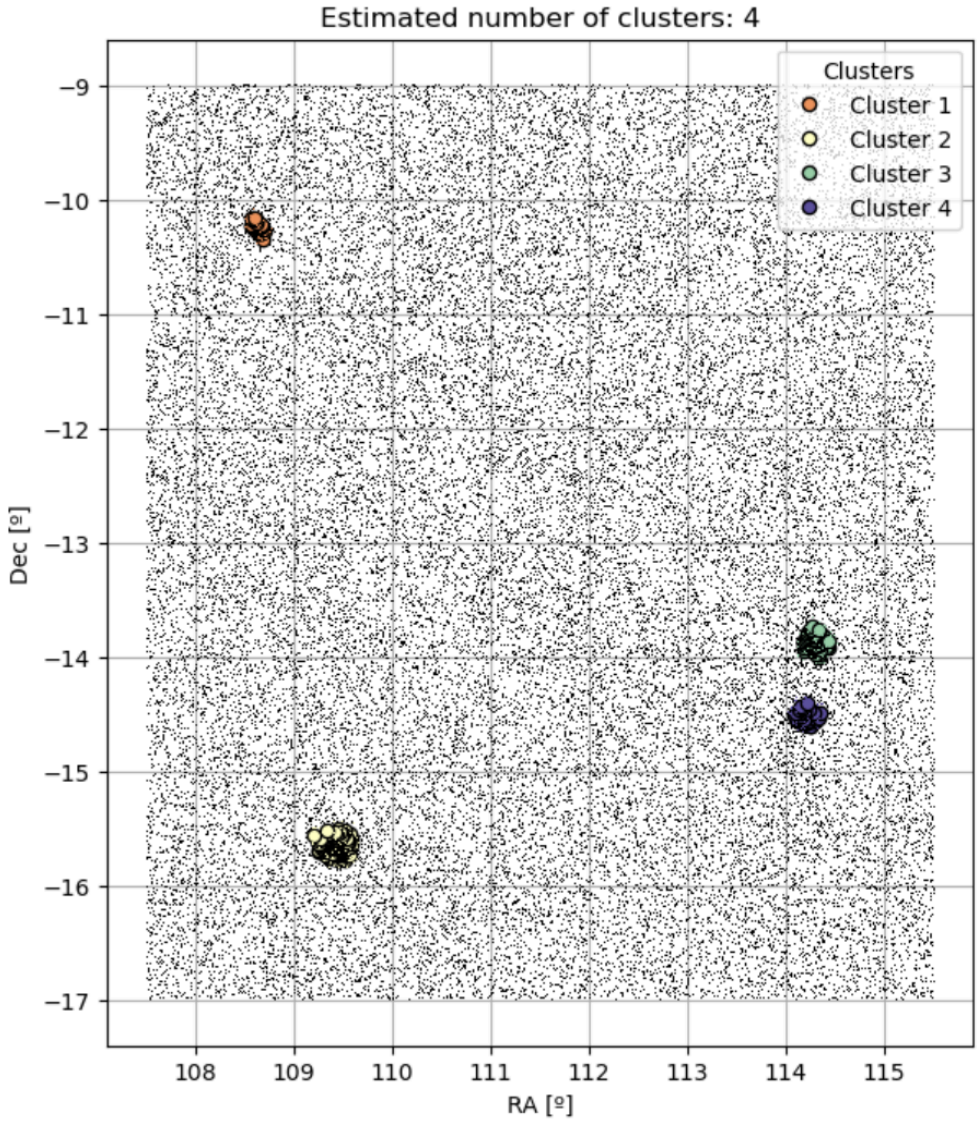

Figure 9. Final A section results: RA/Dec map of clusters. The RA/Dec map of the field after applying DBSCAN with the final selected hyperparameters (*eps*=0.0379, *min_samples*=14) for the positional-only feature matrix. Each cluster is color-coded according to its assigned label, while noise points are plotted in black. Four clear groupings are visible, corresponding to candidate open clusters. Two supplementary groups to the previous two identified groups in §III.A, which are still present.

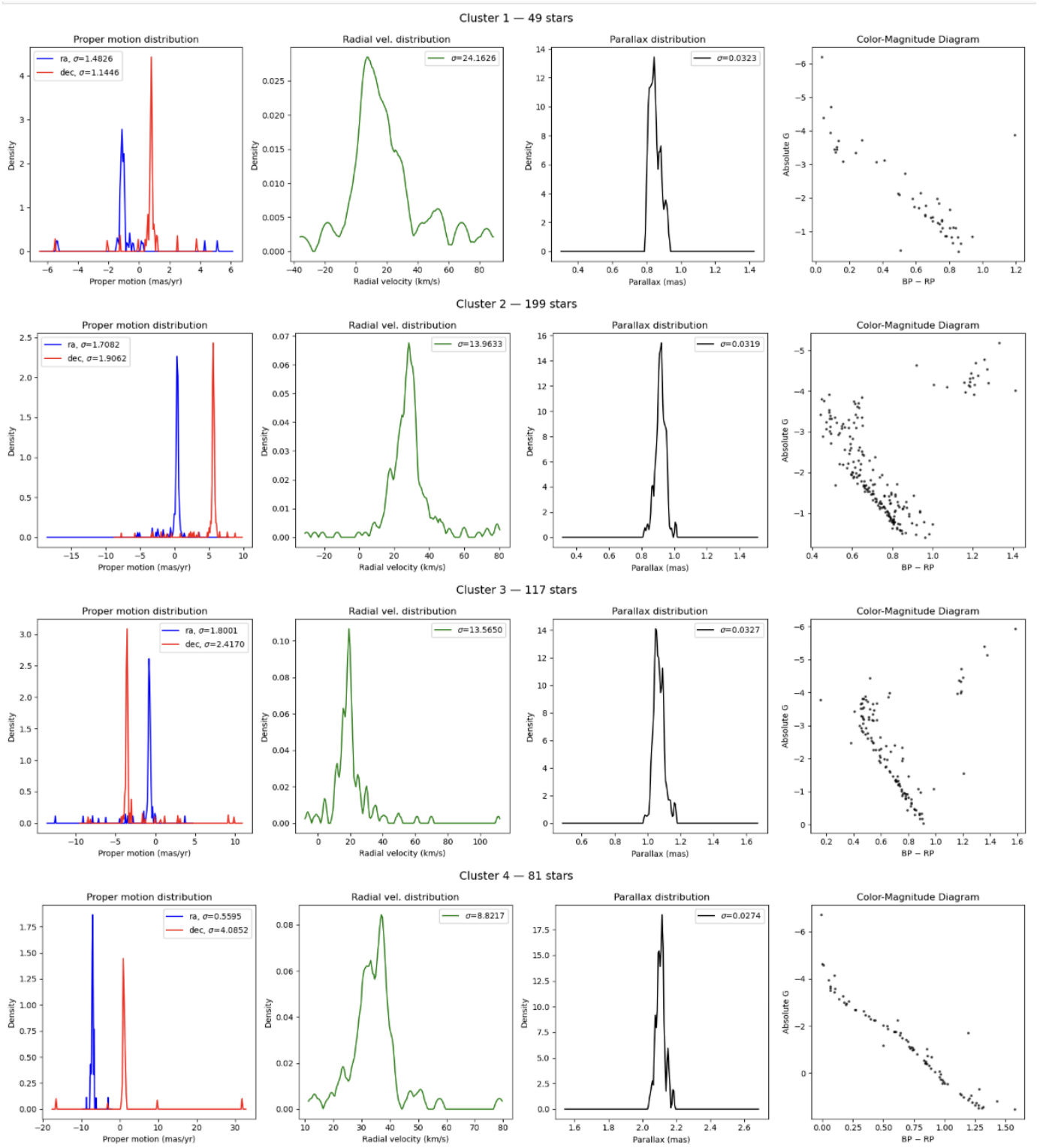

Figure 10. Final A section (positional features) results: Feature distributions and CMDs. Plots showing the parallax, proper motions, and radial velocity distributions for each identified cluster, along with the CMD for each group. Tight parallax and proper motion distributions and

distinct radial velocity groupings, along with coherent main sequences on the CMDs, suggest successful identification of physically associated stars.

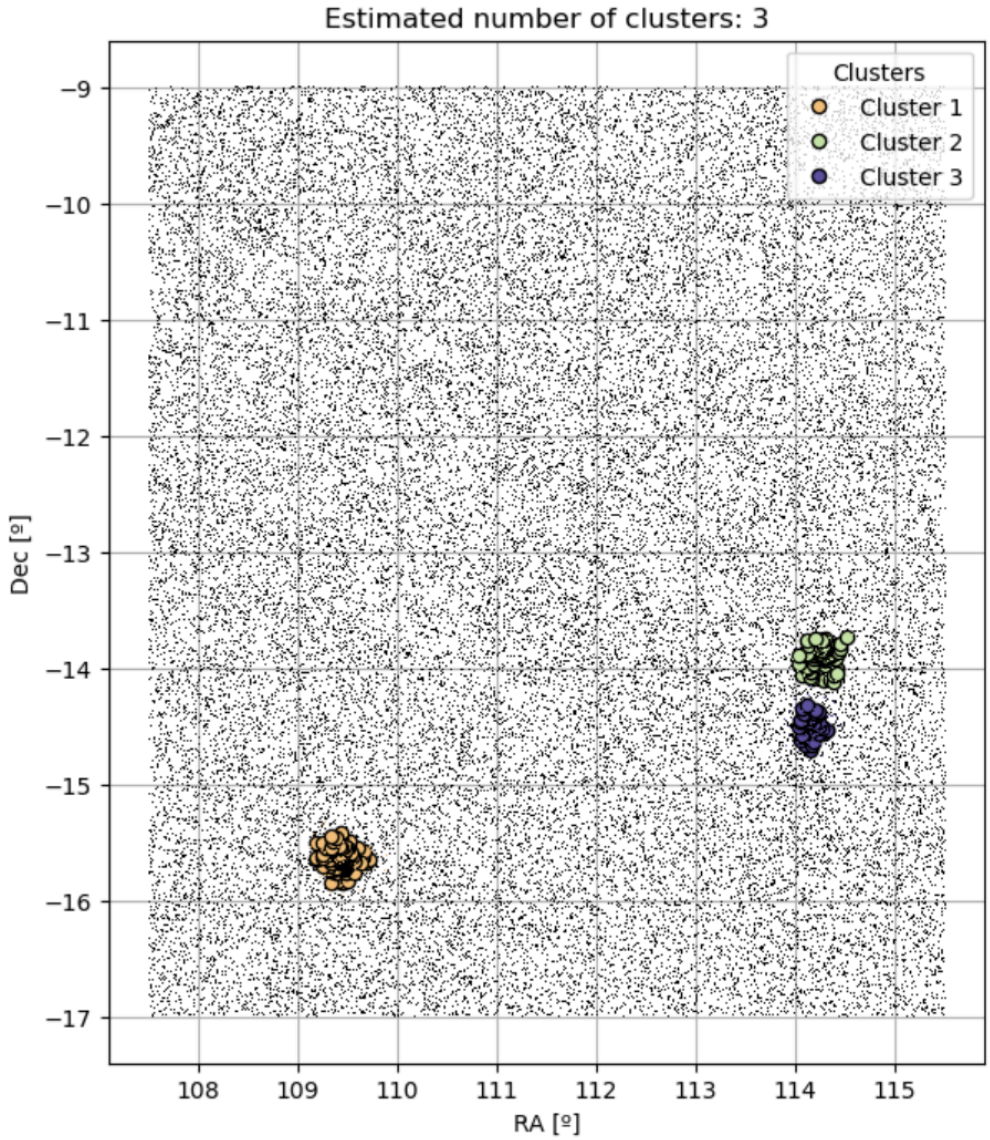

Figure 11. Final B section (positional and kinematic features) results: RA/Dec map. The RA/Dec map for the feature matrix including both positional and kinematic quantities using the final selected hyperparameters (*eps*=0.095, *min_samples*=10). Three compact, well-defined clusters are identified, highlighting the incorporation of proper motions and radial velocity to restrictively select candidates. One additional cluster to the two previously identified in §III.B.

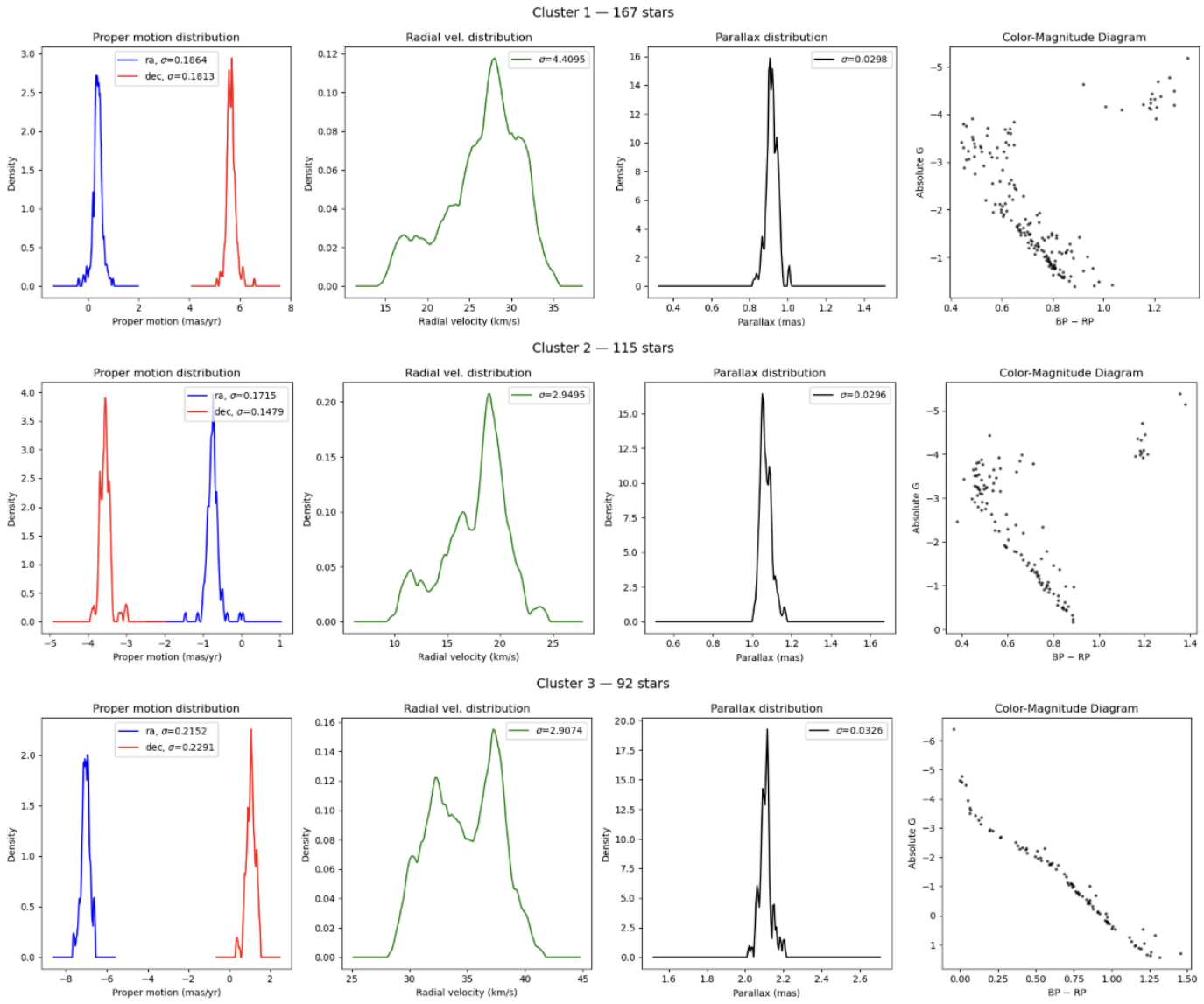

Figure 12. Final B section (positional and kinematic features) results: Feature distributions and CMDs. Diagnostic plots for clusters identified with the full positional and kinematic feature set. Parallax, proper motion, and radial velocity spreads are noticeably narrower than in the positional-only case, while CMDs reveal well-defined main sequence characteristics for the majority of cluster members.

V. REFLECTIONS

The results from the two different feature matrices, including either just positional quantities (A) or both positional and kinematic quantities (B), highlight both important strengths, limitations, and similarities of including or excluding kinematic information in the clustering process.

First, the clustering of both feature sets produced compact, well-defined clusters in RA/Dec space with substantial numbers of objects. It was expected that including the kinematic information would reduce contamination from unrelated field stars and create a more restrictive selection. This was true in terms of both the number of clusters identified, with the positional feature model defining four clusters while the positional/kinematic feature model defining three, and the isolation, with the positional model having a silhouette score of -0.4671 and the positional/kinematic having a score of -0.3272.

Second, the physical coherence of the clusters, as measured by the spreads in parallax and proper motions, improved when kinematic features were included. Parallax, proper motion, and radial velocity dispersions were generally smaller, indicating that physically realistic clusters consisting of stars at similar distances and moving coherently, were better identified with the expanded feature set. The radial velocity distributions struggled the most with tightness and a single-peaked shape, likely due to the higher levels of uncertainty associated with these measurements, which were avoided from the start by selecting non-NULL values in the ADQL query. For future projects of this nature, these problems may be remedied by weighting or scaling this feature differently.

Third, the Color-Magnitude Diagrams (CMDs) showed slightly sharper main sequence structures for the clusters identified with kinematic features. This suggests that including additional dimensions enhances DBSCAN's ability to isolate stars of similar age and composition.

Both of the feature sets identified the same primary three clusters, with the positional set finding an additional one. This additional cluster appeared to be physically real from the diagnostics. For a higher level of clustering sophistication, the higher-dimensional feature set is ideal. For a more exploratory approach to identification, the lower-dimensional feature set may be beneficial.

An additional complication was the overwhelmingly negative silhouette scores in both clustering runs. The score is generally supposed to be positive if not close to 1, but it was still a useful metric for tuning the *eps* hyperparameter. This in complement with spread diagnostics for the *min_samples* hyperparameter created scientific justification for the quality of our models.

Overall, both feature sets successfully identified physically meaningful open clusters, though with differing levels of confidence for cluster membership and boundary definition. The combination of unsupervised machine learning validation and astrophysical diagnostics provided a robust framework for confirming the reality of the detected structures.

VI. FUNCTIONS

This project employed Scikit-learn functions and classes that were covered in the course. The key components included the StandardScaler for feature processing, DBSCAN for unsupervised clustering, and the silhouette_score for hyperparameter tuning and cluster validation.

The StandardScaler class was responsible for normalizing the input features by removing the mean and scaling to unit variance. This was crucial to ensure that features with larger numerical values, such as radial velocity or parallax, did not dominate during the clustering process. The scaling was applied to both feature matrices prior to clustering to create balanced comparison across the dimensions.

The clustering algorithm used was DBSCAN, which identifies clusters as overdensities of points, separated by areas of lower density. Its two primary hyperparameters are *eps*, which defines the neighborhood radius for core points, or points where *min_samples*, the other hyperparameter, points lie within the radius. The latter is also the minimum number of points required to form a cluster. This algorithm was useful because it doesn't require pre-specification of the number of clusters, and is effective at identifying arbitrary clusters, including noise points.

For cluster evaluation, the silhouette_score was useful because it compares how similar each point is to its own cluster relative to other clusters. It was mainly used to guide the tuning of *eps*. Although the scores in this project were abnormally negative due to the high fraction of noise points and the complex nature of astronomical data and structure, they still helped provide a consistent initial estimate of the clustering behavior.

The functions for creating adaptive histograms using Kernel Density Estimation (KDE), were written for previous projects.

VII. AI STATEMENT

The use of the generative AI ChatGPT was limited, though present, in this project.

It was used for model visualization in two cases. The first was for creating a legend identifying which colors correspond to which cluster numbers, which was useful for comparing the points on the RA/Dec maps to the subsequent diagnostic plots. The second was for the *min_samples* tuning plots, where I struggled to only select calculations where the clustering quality was relevant.

Both of these instances were purely Pythonic issues that hindered simple progress. They are also clearly labeled in the Jupyter Notebook.

VIII. LINK TO REPOSITORY

https://github.com/mstruk12/Astro-416-Final-mstruk#