Appendix A′

Software

The following text is a direct translation from Appendix A located in my thesis, which is written in Greek:

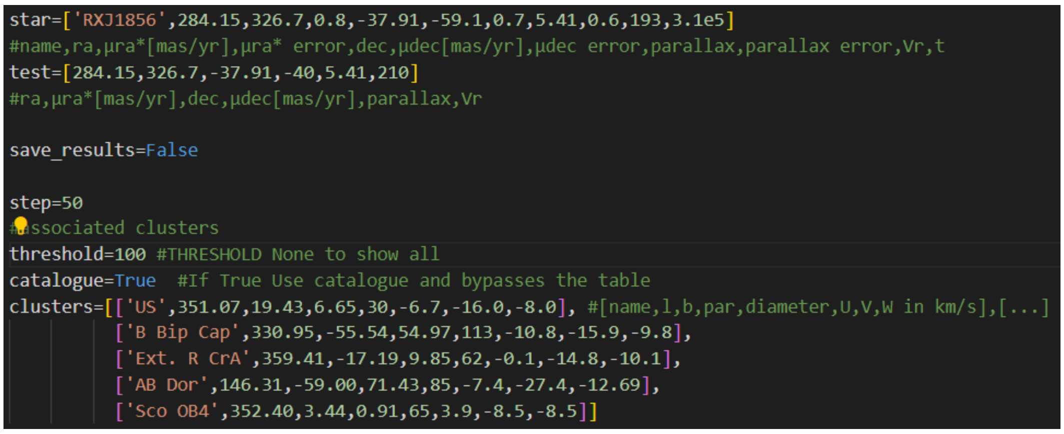
Using Python, I created a code that utilizes simple kinematic equations, the present coordinates, and velocities of isolated stars in order to calculate in a first approximation their orbit in the Galaxy and their kinematic age. My aim was to find the birthplaces of isolated neutron stars and their kinematic age.

The code has four files:

- ▶ Main Body
  - Data entry
  - Parameterization
  - Export results
- ▶ Calculation
- ▶ Graphs
- ▶ Load data and save results from/to files

1. Main Body

Data entry and parameterization take place between the double lines of (#)



```
star=['RXJ1856',284.15,326.7,0.8,-37.91,-59.1,0.7,5.41,0.6,193,3.1e5]
#name,ra,μra*[mas/yr],μra* error,dec,μdec[mas/yr],μdec error,parallax,parallax error,Vr,t
test=[284.15,326.7,-37.91,-40,5.41,210]
#ra,μra*[mas/yr],dec,μdec[mas/yr],parallax,Vr

save_results=False

step=50
#associated clusters
threshold=100 #THRESHOLD None to show all
catalogue=True  #If True Use catalogue and bypasses the table
clusters=[['US',351.07,19.43,6.65,30,-6.7,-16.0,-8.0], #[name,l,b,par,diameter,U,V,W in km/s],[...]
          ['B Bip Cap',330.95,-55.54,54.97,113,-10.8,-15.9,-9.8],
          ['Ext. R CrA',359.41,-17.19,9.85,62,-0.1,-14.8,-10.1],
          ['AB Dor',146.31,-59.00,71.43,85,-7.4,-27.4,-12.69],
          ['Sco OB4',352.40,3.44,0.91,65,3.9,-8.5,-8.5]]
```

Figure A′.1: The environment in which data is entered and parameterization is done

1..1 Data entry

The following data are imputed here:

For the star:
Name
Right Ascention α [°]
Angular Velocity of Right Ascention μ_{ra*} [mas/yr] διορθωμένη από την απόκλιση
Angular Velocity of Right Ascention Error [mas/yr] $\mu_{\alpha*error}$
Declination δ [°]
Angular Velocity of Declination μ_δ [mas/yr]
Angular Velocity of Declination Error $\mu_{\delta error}$ [mas/yr]
Parallax π [parsec]
Parallax Error π_{error} [parsec]
Radial Velocity v_r
Travel time (potential age) τ [kyr]

All of the above are inserted in a tuple with the following order: star=['Name',α,μ_{ra*},$\mu_{\alpha*error}$,δ,μ_δ,$\mu_{\delta error}$,π,π_{error},v_r,τ]

The values for the test trajectory are entered in a corresponding tuple called test. The change of the movement time is only done in the star tuple because otherwise, it creates a problem in the final tables. test=[α,$\mu_{ra*}\delta$,μ_δ,π,v_r].

In addition, it is defined through a tuple array the specified clusters that I wish to associate:

Cluster Name
Galactic Longitude l [°]
Galactic Latitude b [°]
Cluster Parallax π [parsec]
Galactic X-Axis Velocity U [km/sec]
Galactic Y-Axis Velocity V [km/sec]
Galactic Z-Axis Velocity W [km/sec]

And they are entered as follows: clusters=[['Name1',l,b,π,r,U,V,W],...]

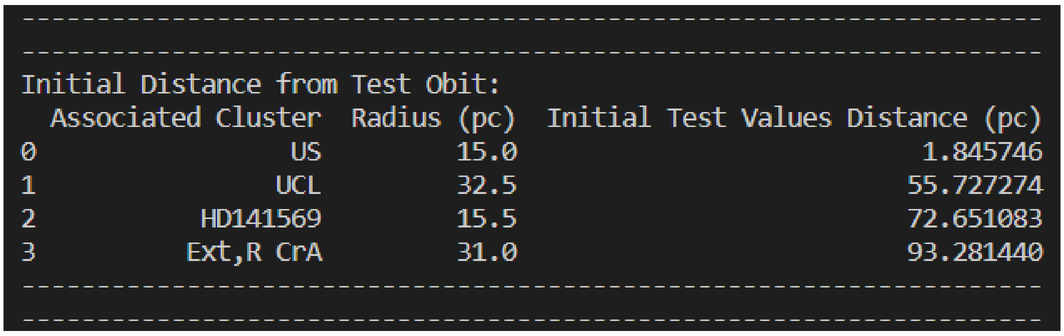
1..2 Parameterization

- save_file = True/False : Save or not the final results
- step = Integer : Number of steps in the computation iteration in terms of time. That is, the number of points to be calculated on the trajectory.
- catalogue = True/False : Whether or not to use the entire directory
- threshold = Number/None : In case of None there is no limit to how many clusters will appear in the chart and results. For a specific number that will be equal to parsec it will show us the maximum distance from each center. It is always recommended to avoid the None option if the directory is used.

1..3 Results Extract and Explaination

Running the program will output a screen that will read:

Initial Distance from Test Obit	Explanation
Associated Cluster	Cluster Name
Radius (pc)	Theoretical Radius (Half Diameter)
Initial Distance from Test Values (pc)	Distance of creation position to cluster center (Test Orbit)



```
-------------------------------------------------------------------
-------------------------------------------------------------------
Initial Distance from Test Obit:
  Associated Cluster  Radius (pc)  Initial Test Values Distance (pc)
0                 US         15.0                           1.845746
1                UCL         32.5                          55.727274
2           HD141569         15.5                          72.651083
3           Ext,R CrA        31.0                          93.281440
-------------------------------------------------------------------
-------------------------------------------------------------------
```

Figure A′.2: Output of the results for the test track

2. Calculation

In the calculations.py file there are 2 functions, kinematics and cartesian_cluster. For fast computing and more accurate conversion of coordinates and measurement units, I use NumPy and astropy packages:

- Kinematics will input the conditions for the 4 orbits as listed below 3., first convert the Celestial coordinates (ICRS) to Cartesian and then via a step function produce the number of points of the orbit. This is where the step integer will be used.
- cartesian_cluster will convert the data for each cluster to Cartesian from Galactic. If catalog is True the whole catalog will be imported, otherwise only the clusters table. If the threshold is different from None, only the clusters whose centers will be in a distance less than the threshold from the end of each of the four trajectories will be displayed and stored.

As it should be shown in the manual, I do not import the galactic potential in my calculation. If time travel is about some million years or less, the potential may be ignored, yet importing would give us better results. So the current velocities that are imported above are considered constant. I would be very happy to help anyone who can import it in the calculations and will answer to every question.

3. Graphs and explanation

The file graph.py there are 2 functions graph_from_clusters and orbits. This is where matplotlib and Tkinter are needed. In more detail:

- Graph_from_clusters outputs the graph of clusters over time entered by the main program. Additionally, a legend window will be produced where the name of each cluster will match the color of the cluster on the chart. Each new run will randomly generate other colors so that a large number of clusters can be entered and varied. The clusters that will be displayed are the result from the clusters function for a given threshold. Finally, the Sun is represented by a black dot in the center of Cartesian coordinates.
- Orbits will display 4 orbits, where the first time point will be the most prominent. The trajectories that will be displayed are the result of the 4 kinematics functions, one for each trajectory:
  - Main Orbit: The red Track will be calculated from the average values entered at the beginning of the program.
  - Minimum Orbit: The green track will be calculated from the minimum values entered at the beginning of the program (Main - Errors).
  - Maximum Orbit: The blue trajectory will be calculated from the maximum values entered at the beginning of the program (Main + Errors).
  - Test Orbit: The black trajectory will be calculated from the test values from the tuple test values.

4. Load data and save results from/to files

In save_prints.py pandas is used to organize the tables and the following functions are included:

- The xl_save which saves in an xlsx file in the results subfolder the points of the Main, Minimum, Maximum trajectories and the distances of the clusters from the final point of the trajectory (previous).

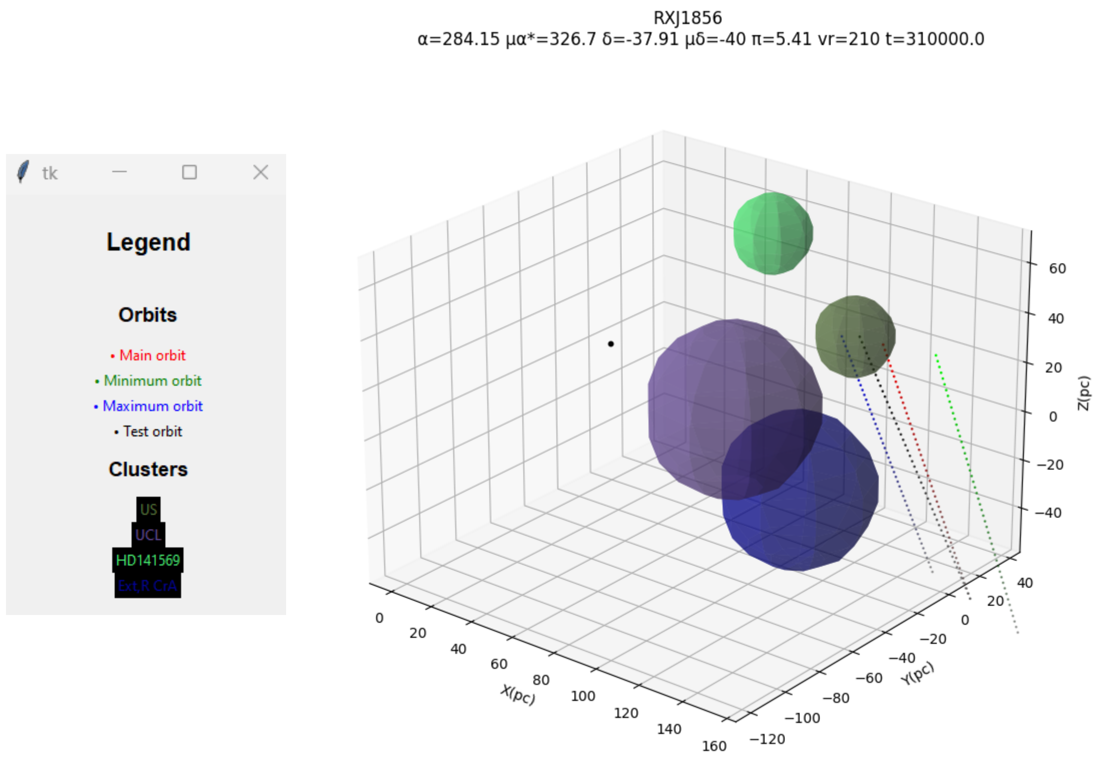


Figure A′.3: Graph and Legend.

- from_catalogue which imports the clusters catalogue.xlsx catalog with the clusters if catalog is True.
- results_print prints the results for the test trajectory as reported in the Main Body 1..3

5. Prerequisite Packages

- NumPy (Simple calculation)
- matplotlib (Graphs)
- Tkinter (Legend)
- Astropy (Coordinate conversion)
- Pandas (Save/export results and import data)