Monte Carlo Simulation for Nemesis Stars with Gaia DR2*

Rose Kong
University of Pennsylvania
(Dated: August 14, 2019)

Passing stars may perturb the Oort cloud and affect the evolution of our solar system. I use the data in Gaia DR2 to re-analyze the results of Bailer-Jones et al. (2018)'s stellar encounters in six-dimensional phase space data. I reevaluate the 31 stars that come within 1 pc of the Sun and propagate the correlated uncertainties via a Monte Carlo sampling of 4000 surrogate stars for a larger statistical sample. The cloud of surrogate stars are then integrated through a Miyamoto-Nagai Galactic potential. Perihelion distances to the star and perihelion times were then generated. Due to computational limitations, this study remains incomplete.

I. INTRODUCTION

Stars which pass close to the Sun may have significant contributions to the evolution of our Solar System. Passing stars may perturb the Oort cloud and inject comets into the inner solar system, threatening life on Earth. The degree of perturbation from the close encounters also depends on not only the distance of the encountering star, but also the mass and speed.

With the second Gaia data release (DR2), we obtain very precise astrometric and photometric properties for more than one billion sources, radial velocities for millions, variability information for half a million stars from selected variability classes, and orbits for thousands of solar system objects. Out of the data set, 7.2 million stars in Gaia DR2 have 6D phase space information. Bailer-Jones et al. (2018) completed a comprehensive survey on close stellar encounters to the Sun with all 7.2 million stars in Gaia DR2 with complete six-dimensional kinematic data, including position, parallax, proper motion, and radial velocity. There is potential for refinement in study of nemesis stars and close encounters.

Here I would like to build off the reported results of the close encounters of Gaia DR2 and investigate the statistical nature of such encounters using a Monte Carlo resampling of the 6D likelihood of the distribution of data. My goal is to perform a statistical analysis on the nature of the encounters as reported by Bailer-Jones et al. (2018).

Gaia DR2 source ID
4270814637616488064
955098506408767360
5571232118090082816
2946037094755244800
4071528700531704704
510911618569239040
154460050601558656
6608946489396474752
3376241909848155520
1791617849154434688
4265426029901799552
5261593808165974784
5896469620419457536
4252068750338781824
1949388868571283200
1802650932953918976
3105694081553243008
5231593594752514304
4472507190884080000
3996137902634436480
3260079227925564160
5700273723303646464
5551538941421122304
2924378502398307840
6724929671747826816
3972130276695660288
5163343815632946432
2926732831673735168
2929487348818749824
939821616976287104
3458393840965496960

TABLE I. From Table 2 of Bailer-Jones et al. (2018) which reports the 31 forecasted encounters within 1 pc.

II. METHODS

The release of the Bailer-Jones et al. (2018) paper identified the closest stellar encounters to the Sun among the 7.2 million stars in Gaia DR2 that have available 6D phase space information. Uncertainties were accounted for via a Monte Carlo resampling of the likelihood distribution for 2000 surrogate particles, then integrated for their orbits over a smooth gravitational potential. The results define 31, 8, and 3 stars which are likely to come within 1 pc, 0.5 pc, and 0.25 pc of the Sun, respectively. Star selection in my analysis will be limited to the 31 stars that have been calculated to come within 1 pc. Gaia DR2 source identification numbers are shown in Table 1.

Selection for these 31 stars was made in Gaia DR2 according to their source IDs. The query made in AQDL can be found in Appendix A.

* With the tremendous help of Dr. Robyn Sanderson and colleague Nicholas Scarsdale

A. Monte Carlo Resampling

The Gaia data are used as they are, and transformed into Galactocentric coordinates. To account for uncertainties in measurements, I have applied Monte Carlo resampling method to draw 4000 samples from the 6D covariant probability density function (PDF) over the dimensions of position, parallax, proper motion, and radial velocity $(\alpha, \delta, \pi, \mu_\alpha, \mu_\delta, R_v)$. These values are given in Gaia, with the exception of the row and column of radial velocity to be empty except its own error; it is uncorrelated to the other five parameters. The PDF is initialized as a 6×6 matrix. This PDF is used for each star to determine and integrate the orbits of the 4000 surrogate stars through the Galactic potential.

B. Orbit Integration

In order to obtain encounter parameters and improve the accuracy of the perihelion parameters for these "clouds" of 4000 surrogate stars stemming from 31 sources and the Sun were integrated through a Galactic potential. The Galactic potential utilized was the 3D, time-independent, asymmetric model from Miyamoto and Nagai (1975) in cylindrical coordinates (r, ϕ, z).

$$\Phi = \Phi_d + \Phi_g + \Phi_h \tag{1}$$

$$\Phi_d = \frac{-GM}{\sqrt{r^2 + (a + \sqrt{z^2 + b^2})^2}} \tag{2}$$

$$\Phi_{g,h} = \frac{-GM}{\sqrt{r^2 + z^2 + b^2}} \tag{3}$$

The d, g, h refer to the Galactic disc, bulge, and halo, respectively. a and b represent length scales and M is the mass of the component. The Miyamoto-Nagai potential can be directly initialized in the *galpy* package that I have chosen to use. a and b are set to 0.5 and 0.0375, respectively, and the potential is normalized to 0.6 such that the radial force is a fraction. Effects of Sun-star interaction or the possibility of deflection will be assumed to be negligible.

Orbits are obtained through integrating the equations of motion using a fourth-order symplectic integrator, called $symplec4_c$ in the *galpy* package. The reason for choosing this integrator rather than Runge-Kutta, Dormand-Prince, or ODE integrators is due to the speed of computation. Fast orbit integration is needed for batch integration of large numbers of orbits. The symplectic integrator is an order of 10^3 faster than the leapfrog integration method, which is an enormous margin faster than *odeint*. Initial conditions for the integration are the six astrometric coordinates of a star. These are automatically transformed into Galactocentric phase space coordinates. The Sun's integration is done by utilizing the same method. The coordinate transformation and the computation of the solar orbit requires us to define the phase space coordinates of the Sun, which is built into *galpy.Orbits*.

C. Timestep

Orbit integration starts at time $t = 0$, representing the presen,t and proceeds to progress to 10 Gyrs in 1000 steps, for both the star "clouds" and the Sun, separately. Due to its integration with the symplectic integration, energy was able to be decently conserved. The step sizes were kept relatively large to be able to trace the orbit without extreme computational needs but not small enough to determine perihelion with high accuracy.

$$ts = numpy.linspace(0, 10, 1000) * units.Gyr \tag{4}$$

D. Calculating Perihelion Distance

The calculation for perihelion, the point of closest distance for any star in the "clouds" to the Sun, was made to determine the proximity and probability of close encounters. Perihelion distance was taken utilizing the distance formula in cylindrical coordinates.

$$d_{ph}^2 = (\Delta R)^2 + (\Delta z)^2 + (R * \Delta\phi)^2 \tag{5}$$

To find the minimum distance, this distance formula was used between every star and the Sun's coordinates, for every iteration with an increase in timestep as mentioned in equation 4. The time of when the orbit reaches d_{ph} was also taken.

III. RESULTS AND DISCUSSION

Data for this study remains incomplete. Failure to complete computation and calculation stems from many factors, such as computing power, run time, algorithmic efficiency, and other miscellaneous errors.

The main problem stems from the lack of computational power. The 6D PDF from which 4000 surrogate stars were generated became 4000 entries of 6-dimensional phase space data for 31 stars, which in itself is sizable for a personal computer. Data manipulation also took more time than expected. Integration of orbits using *galpy.Orbit* had an exceptionally high run time, due to the relatively numerous time-steps for a large total time, and usage of a less efficient integration method, $'odeint'$ before the discovery of a quicker, and reliable integration method.

Other complications include algorithmic efficiency. Due to the shape of the data, most loops had the algorithmic efficiency of, in Big O notation, $O(n^2)$, up to $O(n^3)$, which is universally considered undesirable, especially in cases of large data sets. Combinations of hefty orbit integrations along with an inefficient programming style have caused a tremendous backlash in total computational time.

I have tried many solutions to mitigate these problems. Attempts have been made to increase Jupyter Notebook's data rate to 10^9. Then, I drastically lowered the sample sizes in order to test the computational speeds. A test was made to integrate 1000 samples, which is not ideal since it is lower than the Bailer-Jones et al. (2018) paper, and failed to complete. Instead of attempting integration with 4000 samples, I had drastically lowered it to 100, then steadily raised the number of samples. One star at 100 samples took more than 1 hour to complete. Integrating more stars has overloaded my personal computer's memory and forced restarts.

Additional steps I took were more on a fundamental level. I replaced the slow 'odeint' integrator with a faster method, the symplectic integrator, which also has fairly high accuracy, speed, and energy conservation. To help alleviate the algorithm problem, I have taken out many nested for loops and rewritten them with different methods. Other details include decreasing the total time in the timestep and number of steps. As a last resort to save memory, I have rounded all the generated samples from the 6D PDF to 5 decimal places. Admittedly, there was also a large amount of human error as well.

Unfortunately, my study on this topic is currently at this status. For my next steps, I hope to be able to use a different processing machine for large computations. Additionally, the change in integrator is potentially a much faster way to process the data. If given more time, I would like to plot the perihelion distance (pc) versus perihelion time (kyr) and error bars to calculate its confidence interval. I would also like to plot the perihelion distance versus median perihelion velocities. Colour-magnitude diagrams for stellar encounters could also reveal what classification potential nemesis stars may belong to.

IV. REFERENCES

- Bailer-Jones, C. A. L. 2015a, A& A, 575, A35
- Bailer-Jones, C. A. L., Rybizki, J., Andrae, R., & Fouesneau, M. 2018a, A& A, 616, A37

Appendix A: Gaia archive query

Below is the ADQL query used to select stars from Gaia DR2 according to Bailer-Jones et al. (2018)'s results.

```
SELECT TOP 31 * FROM gaiadr2.gaia_source
WHERE source_id = 4270814637616488064
OR source_id = 955098506408767360
OR source_id = 5571232118090082816
OR source_id = 2946037094755244800
OR source_id = 4071528700531704704
OR source_id = 510911618569239040
OR source_id = 154460050601558656
OR source_id = 6608946489396474752
OR source_id = 3376241909848155520
OR source_id = 1791617849154434688
OR source_id = 4265426029901799552
OR source_id = 5261593808165974784
OR source_id = 5896469620419457536
OR source_id = 4252068750338781824
OR source_id = 1949388868571283200
OR source_id = 1802650932953918976
OR source_id = 3105694081553243008
OR source_id = 5231593594752514304
OR source_id = 4472507190884080000
OR source_id = 3996137902634436480
OR source_id = 3260079227925564160
OR source_id = 5700273723303646464
OR source_id = 5551538941421122304
OR source_id = 2924378502398307840
OR source_id = 6724929671747826816
OR source_id = 3972130276695660288
OR source_id = 5163343815632946432
OR source_id = 2926732831673735168
OR source_id = 2929487348818749824
OR source_id = 939821616976287104
OR source_id = 3458393840965496960
```