Lagrange Points and Stable Equilibrium: Studying the Orbits of Jupiter and its Asteroid Clusters

Jeremy Schiff

Lent 2021

Abstract

The orbits of asteroids bound to Lagrange points L_4 and L_5 of the Sun Jupiter system were simulated. Numerical integration was used to solve the equation of motion. The simulation was tested by calculating the energy of the system, observing the orbits and finding the stability of an asteroid placed exactly at L_4, which was found to have maximum fractional wander of 10^{-10}. The stability of L_4 and L_5 due to perturbations of the initial conditions was studied by finding the maximum wander of the angle between the asteroid-Sun and Jupiter-Sun vectors (from $60°$ - the theoretical *fixed* angle). When calculated for several hundred orbits, the stable region was found to have a width of 0.15 AU. A quadratic relation between the planet's mass and asteroid range of wander was fitted by linear regression. For Jupiter/Sun mass ratios greater than 0.02, L_4 and L_5 were seen to be unstable. This is consistent with other work in the field.

1 Introduction

The three body problem in gravitational physics refers to taking the initial positions and velocities of three point masses, and solving for their subsequent motion according to Newton's laws of motion and universal gravitation. This was long thought to be analytically unsolvable; nearly all configurations are chaotic and eventually eject one of the masses. Although on the whole this remains true - there is no general solution[1]- the advent of modern computers has allowed, for very specific configurations, some periodic solutions to be found [1]. However, both Euler and Lagrange previously found exact analytic solutions to configurations of the three body problem. These solutions occur when a small mass is added to a two body problem, in one of five very specific locations [2] (Figure 1).

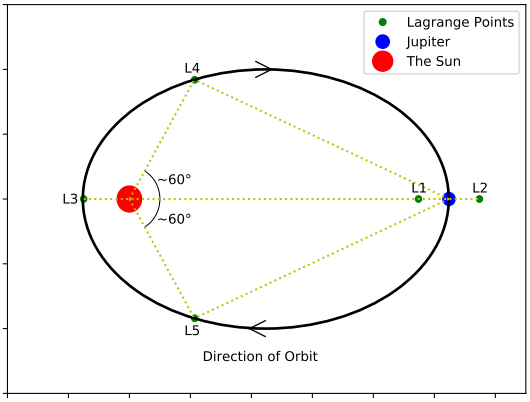

Figure 1: The 5 Lagrange Points shown for Jupiter's orbit. If the orbit were a perfect circle, the angles shown would be exactly 60°, the triangle between $L_{4/5}$, the Sun and Jupiter would be equilateral, and $L_{4/5}$ would be exact points. However, for elliptical orbits, the equilateral triangle becomes isosceles and the points are not as fixed, becoming more of a Lagrangian "area" [4]. The Python script used to make this can be found in Appendix B Sun and Jupiter not to scale.

These points are where the gravitational pull of the two bodies is precisely equal to the centripetal force required for a third small mass to orbit with the same period as the planet. Therefore, with respect to the other two bodies, a small mass placed exactly at one of the points should remain in stationary equilibrium. All five occur at maxima of effective potential and thus should be unstable equilibria. But in practice L_4 and L_5 are stable. This is because, if the ratio of the two large masses is much less than 0.04, the Coriolis force (as viewed from the frame rotating about the barycentre) is great enough to pull the objects at L_4 and L_5 back into equilibrium from small perturbations [3]. The planet Jupiter has two asteroid clusters in stable Lagrange points; Greeks and Trojans (Figure 2). This report describes the development and analyses of Python code, built to model the orbits of these three bodies relative to the Sun, and investigate their stability. The code is also briefly extended to other cases, where different mass planets are considered. In the Analysis Section, the necessary computational considerations to solve the problem are discussed, including a discussion of the method used, and the scaling of units. The Implementation Section describes the basic structure and necessary algorithms used in the code, and considers the efficiency of the program, and how it was optimised. The Results and Discussion Section investigates the tests performed to demonstrate the program creates an accurate simulation, and presents the final results with a discussion of errors. Finally, a Conclusion will summarise the important takeaways.

[1]Excluding K.F. Sundman's series solution, (ignored here due to its *very* slow convergence) the full explanation can be explored in M´emoire sur le probl'eme des trois corps, Acta Math 36 (1913), (French)

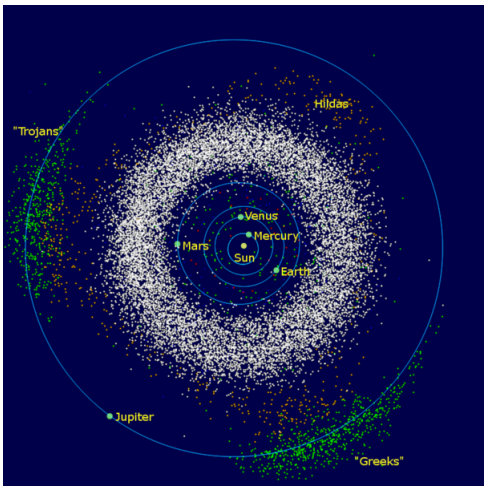

Figure 2: The Greek and Trojan asteroids (green). [5]

2 Analysis

2.1 Physical Model

2.1.1 Assumptions, Approximations and Simplifications

For the purposes of this project, assumptions were made to simplify the problem. Firstly, no bodies other than the four mentioned above were accounted for, and the asteroids were taken as having no mass. This meant the only bodies exerting gravity were the Sun and Jupiter; a valid assumption as the gravitational force from all external masses, and due to the asteroids' mass, are comparatively negligible.

The asteroids were also considered as points, rather than clusters of many objects, allowing the details of the Lagrange points to be considered, without the complication of individual asteroids.

The Lagrange points L_4 and L_5 are mathematically identical, because the maxima of effective potential are identical [3]. Therefore, in some later sections, only one of the asteroid clusters will be considered, but the results are valid for both.

2.1.2 Mathematical Analysis

Using Newtons laws of Gravitation, the gravitational equations of the system can be written as:

$$m_i \frac{d^2\mathbf{r}_i}{dt^2} = -\sum_{i \neq j} \frac{G m_i m_j}{r_{ij}^3} \mathbf{r}_{ij} \tag{1}$$

We then transform to a new set of variables, Y_i, for $0 \leq i \leq 15$, each Y representing one of the degrees of freedom of the system; x, y, z positions ($0 \leq i \leq 7$) and velocities ($8 \leq i \leq 15$) of the Sun, Jupiter, Greeks, and Trojans. Then, in preparation for the computational integration, we split this into 8 sets of coupled first order differential equations;

$$\dot{Y}_k = Y_{k+8} \tag{2}$$

$$\dot{Y}_{k+8} = \frac{F_{\mathbf{x},0}}{m_0} \tag{3}$$

where subscript 0 refers to the sun, and for $0 \leq k \leq 11$ and $\mathbf{x} = x, y, z$ [6].

2.1.3 Frame of Reference

Many frames of reference can be used to study Lagrange points. Commonly, a frame rotating about the barycentre is chosen, thus the Sun and Jupiter are at rest [3]. Here, that convention is *not* used, rather all calculations are done in the Sun's rest frame. This was chosen to produce more familiar orbital shapes for the bodies being studied; in this frame, the Sun is at rest, and Jupiter forms the familiar elliptical orbit. A notable drawback is that it is an accelerating frame; the Sun is orbiting the centre of mass of the system. This will cause some problems, discussed in 4.1.3.

2.2 Initial Conditions

To find the most accurate results, the program must be given accurate initial conditions. When Jupiter is at the apogee of its

orbit, its speed must be at a minimum to conserve angular momentum. Therefore, this is a good place to take known, accurate initial conditions; $distance = 5.455AU$ [7] and $velocity = 2.622AU/year$ [8]. The axes can be defined such that the initial position is on the y-axis, and thus the initial velocity is purely in the x direction. Then, using the definition that the two relevant Lagrange points lead and trail Jupiter by 60°, the initial conditions of the asteroids can be found.

2.3 Computational Analysis

Once the equation has been represented as in equations (2) and (3), built-in scipy functions can be used to solve it computationally. The 'integrate' package within scipy contains numerous methods of integration. In this case, the odeint function worked well. This integrates using initial conditions (here, of the 16 Y_i's defined above) over a given array of times, using the LSODA algorithm [9] [10]. This returns a matrix of results, each of the 16 columns describing a Y_i's progression through time.

The `odeint` function from `scipy.integrate` was used, implementing LSODA, rather than `solve_ivp` , which uses the Runge-Kutta (RK) integration method [11]. This was done to improve performance.

2.4 Scaling

In order to keep with industry convention, the problem was scaled into Astronomical Units of distance (AU) and Solar Masses ($M_\odot$), leading to $G = 4\pi^2$ [12]. This scaling ensures the numerical values in the simulation remain reasonable, reducing the possibility of arithmetic under/overflows.

3 Implementation

3.1 The Core Functions

The code uses two main functions to find a solution to the problem. First, a function to create the 16 coupled ODE's (in the form of equations (2) is created, and (3)) defined as 'derivatives', which is a function of time and Y_i. Then, the integration was computed. As explained in 2.3.2, odeint was chosen, producing a matrix of results; one column for each variable, Y_i, and one row for each time the integral was calculated. Transposing this result created a matrix with each row representing a single variable journey through time, making plotting the results straightforward.

3.2 Uses of the Core Functions

Once the core functions were defined, they could be used to find a variety of results. Here, they were called on for 5 programs; to plot the orbits of the 3 bodies around the Sun, to find the stability of the Lagrange points, to plot the energy of the system, to see how the system changes due to perturbation, and to consider the effects of changing the mass of Jupiter. The scripts for all programs can be found in Appendix B, written in Python 3.7.

4 Results and Discussion

4.1 Testing the Program

4.1.1 Plotting the Orbits

The simplest test to ensure the program was working properly was to confirm that it produced a stable, elliptical orbit around the Sun for all three bodies. Since one orbit of Jupiter takes 11.86 years [13], to get a few hundred orbits (and thereby establish stability), the code was run for 3000 years (Figure 3).

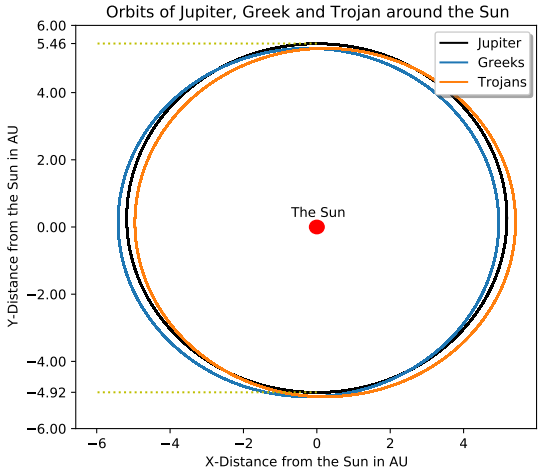

Figure 3: The orbits around the Sun of Jupiter, Greek, and Trojan over 3000 years; approximately 250 orbits, plotted in the Sun's frame of reference. The overlap of each orbit implies stability, and the annotated points (shown by dotted yellow lines) show the orbit is an ellipse. Sun not to scale.

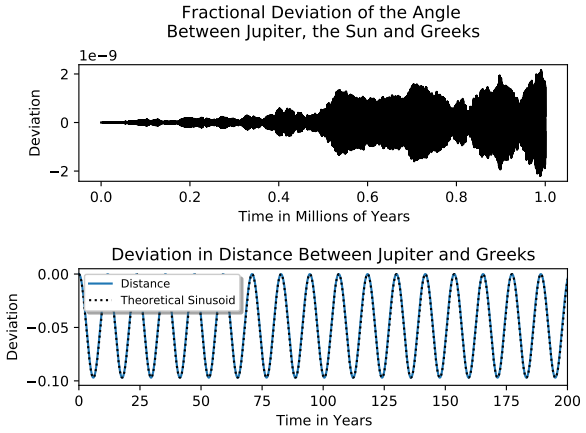

Figure 4: Top: Fractional angle deviation (from 60°) of the angle between Greek, the Sun and Jupiter. Bottom; fractional (relative to theoretical maximum) distance deviation between Jupiter and Greek, compared to a perfect sinusoid with period of 11.8. To save space, only the first 200 years are shown.

The stability can be seen because each successive orbit overlaps the previous - making it appear as if only 1 ellipse is plotted. The orbit's elliptical shape can be seen by the annotated points, showing the shape's asymmetry.

4.1.2 Stability of the Lagrange Point

The stability of the Lagrange points was also tested (Figure 4). To do this, the fractional deviation in the angle between Greek, the Sun, and Jupiter were plotted. Notably, over *10,000* years, the mean angle is exactly 60°, until the *13th* decimal, with a standard deviation of the order of 10^{-11}, and a maximum angle of fractional wander of the order of just 10^{-10}. The centre of oscillations stays at 60°, and the oscillations are not consistently increasing. Therefore, it would appear that Greek is staying fixed to the Lagrange point, with only small oscillatory movement away. Thus, objects at this Lagrange point are in stable orbit.

The distance between Jupiter and Greek was also plotted. As expected, it oscillates sinusoidally because the orbit is an ellipse; two objects 60° apart will have changing distance. Importantly, the sinusoid should have a period equal to the orbit, since when Jupiter and Greek complete each orbit, they should return to their original positions. This indeed was the case; the sinusoid had a period of 11.8 years, approximately matching the orbit [2].

4.1.3 Energy of the System

The final test carried out was on the energy of the system. Since the calculations are done in the Sun's frame (an accelerating frame, if only slightly), the total energy does have some variation. However, the important result is - as expected for a bound orbit - the form of the energy oscillates between kinetic and potential, while total energy is always negative.

[2] The discrepancy is likely due to imperfect numerical integration, and simplifications to the problem, including the point mass assumption.

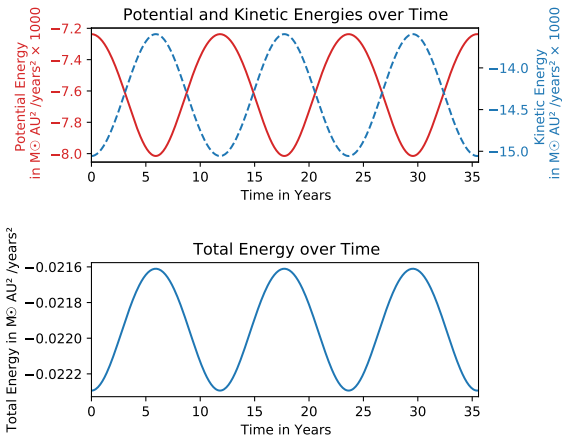

Figure 5: Top: the potential and kinetic energies of Jupiter. Bottom; total energy of the system. To save space, only the first 35 years are shown, but the pattern repeats for any time plotted. Note: over all 3000 years, the maximum value of total energy was $-0.004(M_{\odot})AU/year$

.

4.2 Varying Initial Conditions

The dependence of the Lagrange points on the initial conditions was also investigated. To do this, the program was run with a variety of initial positions, varying the radial distance from the Sun and angular distance from Jupiter, and speeds, varying the x and y velocities. To test whether the resulting motion was just wandering from the Lagrange point, or completely unbound, the following reasoning was used:

1. If the asteroid is bound to the Lagrange point, it may wander, but will always be on the same orbital cycle, with a comparable period to Jupiter.

2. Therefore, if this is *not* the case, it can be inferred that the asteroid is *not* bound to the Lagrange point.

3. If the asteroid has a different orbital period to Jupiter (or is not orbiting at all), then at some point in time, the angle between Trojan, the Sun, and Jupiter will be 180°.

4. Therefore, if a large number of Jupiter orbits are considered, any initial position where the maximum of this angle is less than 180° must be bound to the Lagrange point.

Figure 6 shows these results for perturbing initial angle, and 7 shows the result of perturbing initial speed. From these we can infer the region of initial positions in which the asteroid will eventually fall into stable orbit at the Lagrange point and clearly see their maximum angular wander. The asteroid will remain in the Lagrange point so long as it starts radially within 0.15 AU, and angularly within 13° clockwise, or 36° anti clockwise, of the Lagrange point, with unchanged initial velocity. Similarly, the asteroid is bound to the Lagrange point if it starts at the original position, with an x velocity within 0.11, and y velocity within 0.08 AU/year of its initial velocities. Beyond these values, at some point over the 3000 years, Trojan forms a straight line with the Sun and Jupiter and thus is unbound from its Lagrange point.

It is also notable that there is a wider range of allowed changes to the x velocity. This is likely due to the initial starting position used; since we start at the farthest point on the ellipse, and define this to be along the y axis, all the velocity is in the x direction. Therefore, larger absolute changes to the x velocity are relatively low percentage changes, whereas small absolute changes in y velocity are proportionally very large to the proper y velocity of 0.

Another interesting result is the symmetry, or lack thereof, in Figure 6. When changing radial distance, the graph is highly symmetric, indicating there is no mathematical difference between perturbation radially in and out. However, the same is *not* true when varying *angular* distance to Jupiter; there appears to be a secondary region in which the asteroid will fall into stable orbit, if perturba-

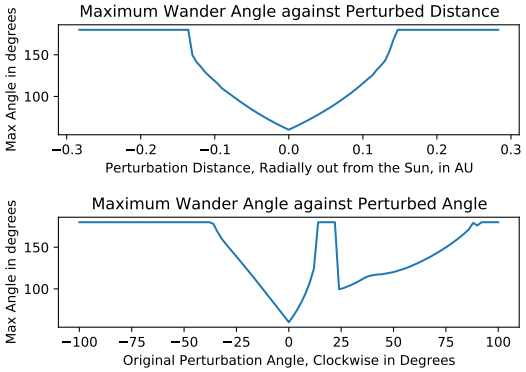

Figure 6: Top: distance perturbation against maximum angle between Trojan, the Sun and Jupiter. Bottom; angle perturbation against maximum angle between Trojan, the Sun and Jupiter. Both plotted for 3000 years.

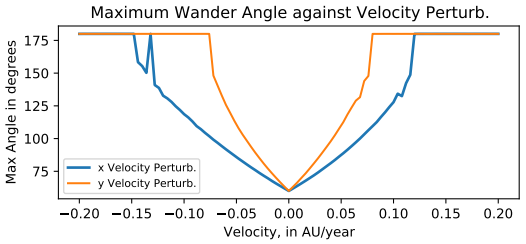

Figure 7: Perturbation of speed in x (blue) and y (orange) directions plotted against maximum angle between Trojan, the Sun and Jupiter. Plotted for 3000 years.

tion occurs between 22° and 91°. The exact physics of why this is the case are beyond this projects' scope, but some possibilities to be explored in future investigations would be due to the angular changes taking Trojan too far off the ellipse, or interesting gravitational effects, possibly due to nearing the other Lagrange points.

4.2.1 Long Timescales

The stability over long timescales was also briefly investigated. Run over 30,000 years, the perturbation program yields a similar graph to figure 6, however the range of stable radial perturbation is smaller by 0.015 AU. To further test this, the result due to a radial perturbation of 0.1 AU (which, for short timescales is predicted to be firmly within the stable region) was plotted for the standard 3,000 years, and for 275,000 years, and was found to be ejected from orbit in the latter. These results suggest not all initially stable orbits remain as such. Therefore, long timescales are required to identify stable orbits (Figure 9 and 10 in Appendix A).

4.3 Varying Planet/Sun Mass Ratio

Twenty simulations were run with varying planet masses (all other parameters were kept as for Jupiter), over 3000 years, and the range of angular wander was plotted (Figure 8). When the mass ratio $\gtrapprox 0.02$, the asteroid was no longer bound to the point, and thus is excluded from the figure. This is consistent with Greenspans's result [3] that stability occurs if the Planet/Sun ratio $\ll 0.04$ [3]. For lower values, a polynomial fits the data. Higher orders fit more accurately, but with diminishing returns for R^2, and thus a simple quadratic was used to model the relation between mass ratio and range of angular wander from a stable Lagrange point.

4.4 Performance

To see the performance of the program, the time library was imported, allowing times to be recorded at different points in the program. All the programs had very short run times, under 10 seconds, with the exception of programs which required the integration to be computed multiple times. When varying initial conditions, plotting 101 points for each line (Figures 6 and 7) took 30 minuets to run on a 2017 Macbook, using a Intel Core m3 processor, and varying mass ratio took 79

[3]Note: the result calculated by Greenspan was done using the inverse ratio, and therefore is expressed as $Sun/Planet \gtrapprox 25$

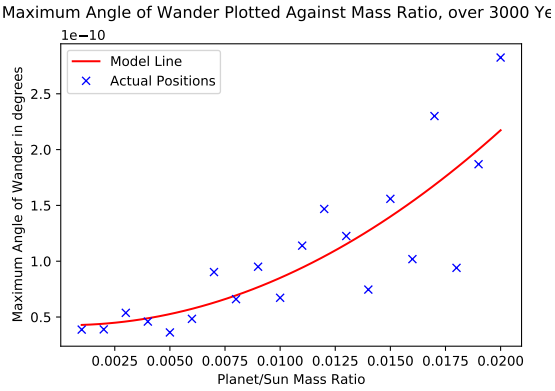

Figure 8: Range of wander plotted against Planet/Sun Mass ratio, with a least square fit. The equation of the fitted line is $(4.51 \times 10^{-7})x^2 + (1.42 \times 10^{-9})x + (3.73 \times 10^{-11})x$, with an R^2 value of 0.83

.

seconds to plot 20 points. Considering that each point required the simulation to be recomputed, these are not unreasonable. Due to these longer run time, the os library was used, allowing the computer to audibly notify the user of completion.

To improve the performance, it was noted that with the given initial conditions, z-displacement should remain constant. Including the z-value calculation would increase the computational time, and thus the simulations was considered a 2 dimensional problem, ignoring the trivial results of $z = 0$ and $V_z = 0$ for all time.

5 Conclusion

The program successfully simulated the binding of objects to the L_4 and L_5 Lagrange points over many cycles. The range of fractional angular wander for an asteroid placed on L_4 or L_5 was found to be of the order of 10^{-12}.

Numerous tests were run on the program to ensure the results produced were as expected. This included tests to ensure the orbits of Jupiter, Trojan and Greek produced, were ellipses, and that Greek (and equivalently Trojan) stays very stable at 60° from Jupiter, and to ensure the total energy of the system was negative. The final test was difficult, because the model was set in an accelerating frame, rather than the centre of mass frame, so total energy varies. Even in this frame, the energy never exceeded 0, so it can be concluded that the orbits must be bound.

The stable region was found to have an allowance of just $\pm 0.15AU$, but a very large angular range of $24° \lessapprox \theta \lessapprox 73°$. This matches well with observations of the real distribution of asteroids [5]. Future work could extend the range of initial conditions covered and fully map out the region by using perturbation combinations of radial and angular displacement, as well as investigate the surprising behaviour resulting from angular displacements clockwise of $22° \lessapprox \theta \lessapprox 91°$. Similarly, the stable region of velocities was found to be perturbations of ± 0.11 or ± 0.08 AU/year for the x and y velocities, and further work could research the stable region of velocity phase space due to perturbation in the x and y direction simultaneously.

The effect of longer timescales was briefly investigated, and the stable region was seen to shrink. Further work could consider this more quantitatively to find the relation between timescale and stable region.

Although the wander was of the order of 10^{-10}, the range of wander of asteroids depended quadraticly on the mass of the planet for planet/sun mass ratios less than 0.02. Once the planet/sun mass ratios exceeded ~ 0.02, no stability was observed. This is in line with earlier analytical findings [14].

Overall, this simple Python script is consistent with with the existing body of works on Lagrange points L_4 and L_5 and stable orbits. It has been shown that they are stable points of orbit for the Greek and Trojan asteroids, and the range of this stability with respect to position, velocity and mass ratio has

been quantified for short and long timescales. Further work could apply this to any three body system and can be extended to three dimensional space, allowing for perturbations in the z direction.

References

[1] X. L. S. Liao, "On the periodic solutions of the three-body problem," *National Science Review*, p. 1070–1071, 2019. Accessed: 25 March 2021.

[2] C. Marchal, *The Three Body Problem*. Studies in Astronautics 4, Elsevier, 1990.

[3] T. Greenspan, *Stability of the Lagrange Points, L4 and L5*. Cornell University, 2014.

[4] R. R. I. Todoran, "The elliptical three-body problem - triangular solution," *Astronomische Nachrichten*, vol. 313, no. 5, pp. 315–317, 1992.

[5] "The inner solar system." "https://commons.wikimedia.org/wiki/File:InnerSolarSystem-en.png".

[6] D. Buscher, "Part II computational physics projects, week 2 handout," *Cambridge University*, 2021.

[7] D. Olsson-Steel, L. Taaffe, and S. Williams, "Short-period comet production in close encounters with Jupiter," *Proceedings of the Astronomical Society of Australia*, vol. 8, pp. 108–118, Jan. 1989.

[8] "Official NASA website." https://nssdc.gsfc.nasa.gov/planetary/factsheet/jupiterfact.html. Accessed: 18 March 2021.

[9] "Scypy documentation website." https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.ode.html#scipy.integrate.ode. Accessed: 21 March 2021.

[10] A. Hindmarsh, "Odepack, a systematized collection of ode solvers," *Scientific Computing*, 1983.

[11] M. Zeltkevic, "Runge-kutta methods." "https://web.mit.edu/10.001/Web/Course_Notes/Differential_Equations_Notes/node5.html".

[12] D. Buscher, "Part II computational physics projects," *Cambridge University*, 2021.

[13] "Official NASA website." https://solarsystem.nasa.gov/planets/jupiter/in-depth/. Accessed: 18 March 2021.

[14] S. Kemp, *An Examination of the Mass Limit for Stability at the Triangular Lagrange Points for a Three-Body System and a Special Case of the Four-Body Problem*. PhD thesis.

A Long Timescale Figures

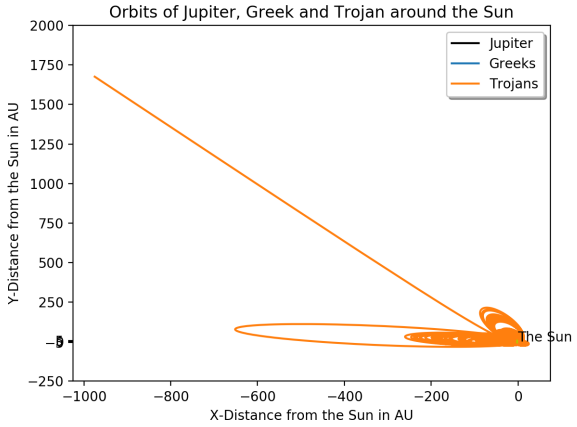

Figure 9: The orbit of the three bodies around the Sun, but only Trojan is visible. Trojan here has been perturbed radially out by 0.1AU, and can be seen to become increasingly more unstable, until ultimate being ejected from the system.

.

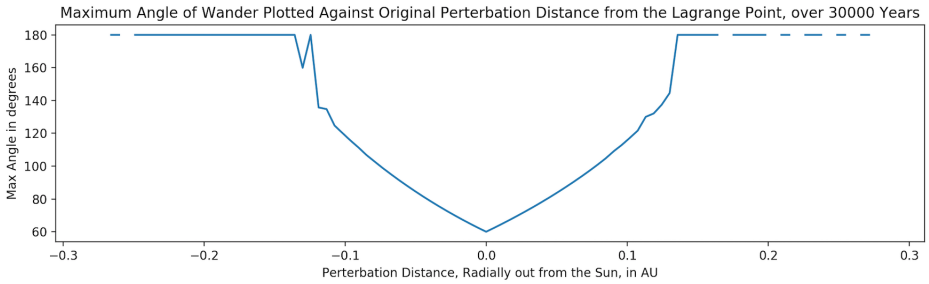

Figure 10: Distance perturbation against maximum angle between Trojan, the Sun and Jupiter plotted for 30000 years. Note that the stable region is 0.015AU smaller on both sides

.

B Code

Listing 1: CoreFunctions.py the Core Functions

```
"""
Creates the solver function; the main integral function to solve Newtons laws
of gravity for Jupiter, Greek and Trojan using their positions,
velocities and masses
"""

import numpy as np
import scipy.integrate
import matplotlib.pyplot as plt

mod = lambda c : np.sqrt(c.dot(c))
years = 1000000

def derivatives_helper(y,t,M):
    """
    Outputs (no. evaluation times) by (18) matrix, each collum, y[i],
    represents y[i]'s evolution through time
    """
    G =4*np.pi**2
    Mj = M
    Ms = 1

    #Vik gives the vector pointing from i to k (leading to force on k from i)
    Vjs = y[0:2]-y[2:4]
    Vsg = y[4:6]-y[0:2]
    Vjg = y[4:6]-y[2:4]
    Vst = y[6:8]-y[0:2]
    Vjt = y[6:8]-y[2:4]

    Kjs = G*Mj/(mod(Vjs)**3)
    Kjs_s = G*Ms/(mod(Vjs)**3)
    Ksg = G*Ms/(mod(Vsg)**3)
    Kjg = G*Mj/(mod(Vjg)**3)
    Kjt = G*Mj/(mod(Vjt)**3)
    Kst = G*Ms/(mod(Vst)**3)

    return np.concatenate([
        y[8:],         # first differentials
        -Kjs*Vjs,        # second differential of Sun
        Kjs_s*Vjs,         # second differential of Jupyter
        -Ksg * Vsg - Kjg * Vjg, # second differential of Greece
        -Kst*Vst - Kjt*Vjt      # second differential of Trojhan
    ])

# Outputs (no. evaluation times) by (16) matrix, each collum, y[i],
# represents y[i]'s evolution through time
def solver(times,TrojanX, TrojanY, TrojanVx, TrojanVy, M):
    """
    The function to solve for all subsequent motion of the bodies.
    Inupts: evaluation times, x and y position,
            x and y velocity of Trojan, Jupiter mass
    """
    def derivatives(t,y):
        return derivatives_helper(t,y,M)

    return scipy.integrate.odeint(
        func = derivatives,
        t=times,
        # using maximum distance to sun 5.455 AU, and therefore minimum
        # velocity 2.622 AU/yr (in sun frame) (1AU = 1.496 *10^11 m)
        y0 = (0, 0, #initial values of y[0-1]; sun position
        0, 5.455, #initial values of y[2-3] jupiter position
        -5.455*np.sin(np.pi/3) , 5.455*np.cos(np.pi/3),#initial values of y[4-5]; greek
```

```
         position
64           TrojanX, TrojanY, #initial values of y[6-7]; trojan position
65           0,0,#initial values of y[8-9]; sun velocity
66           2.622, 0, #initial values of y[10-11]; jupiter velocity in AU per year
67           2.622*np.cos(np.pi/3), 2.622*np.sin(np.pi/3), #initial values of y[12-13];
       greek velocity in AU per year
68           TrojanVx, TrojanVy),#initial values of y[14-15]; trojan velocity in AU per year
69       )
70   times = np.arange(0,years,0.1) # 10 years aproximately = 1 orbit
```

Figure	**Scripts**	**Files Generated**
Figure 1	`LagrangePointsDiagram.py`	`LagrangePointsDiagram.pdf`
Figures 3, 9	`MainOrbitPlots.py`	`MainOrbitPlots.pdf`
Figure 4	`StabilityPlot.py`	`StabilityPlot.pdf`
Figure 5	`Energies.py`	`Energies.pdf`
Figures 6,7,10	`Perterb.py`	`PerterbPosition.pdf` `PerterbSpeed.pdf`
Figure 8	`ChangingMass.py`	`ChangingMass.pdf`

Table 1: List of scripts used to perform functions which generated each figure.