Modelling the Trojan Asteroids

Christopher Gallagher *University of Cambridge*

April 27, 2020

Trojan asteroids orbit the Sun in resonance with Jupiter at Lagrange points, and have their own stable orbits about these points. This can be treated as a restricted three-body problem, which is numerically evaluated in this report using a stiff Radau method to solve the system of coupled differential equations. Without perturbations, a maximal deviation of 4.68×10^{-13} AU from the Lagrange points was observed over 1000 orbits (11,852 years). The application of perturbations to these asteroids demonstrated the predicted existence of stable tadpole and horseshoe orbits, and suggested that wander from the Lagrange point depends quadratically on perturbation size. Such orbits were observed to be stable up to a planetary mass of 0.04 $M_\odot$, also in agreement with theoretical predictions. Word Count: XXXX

1 Introduction

The Jupiter Trojans, commonly known as the Trojan asteroids, are two large groups of asteroids that share the planet Jupiter's orbit around the Sun in a 1:1 orbit resonance. These two groups are called the Greeks and the Trojans, named after opposing sides in the mythological Trojan war, and lead/trail Jupiter respectively in its orbit. They correspond to Jupiter's two stable Lagrange points: L_4, lying 60° ahead of the planet in its orbit, and L_5, 60° behind, with asteroids distributed in two elongated, curved regions around these Lagrange points.

The first Jupiter Trojan, 588 Achilles, was discovered in 1906 by the German astronomer Max Wolf [1], and a total of 7642 Jupiter Trojans have been found as of February 2020 [2].

Research into Jupiter's Trojan asteroids continues, with the particular focus on their origins reliant on an understanding of their orbit stability [3] [4]. This informs studies into their composition [5], as travel to these asteroids is considered for their potential in mineral mining [6] [7].

The purpose of this report is to use numerical simulation techniques to investigate the stability of orbits about the Lagrange points, demonstrating that asteroids oscillate about these points with small perturbations and quantifying the absolute distance of the asteroids from the Lagrange point (the wander) during their orbits. The impact of variation in planetary/solar mass on asteroid orbit stability will also be considered.

2 Theoretical Background

2.1 Lagrange Points

The asteroids exist at or near Lagrange points, defined in Lagrange's initial analysis of the three-body problem in 1772 [8], in which he demonstrated the existence of five equilibrium points for an object of negligible mass orbiting under the gravitational effect of two larger masses. Three of these equilibrium points, L_1–L_3 lie on the line joining the two masses, while each of the remaining two points, L_4 and L_5, lie at the apex of an equilateral triangle with base equal to the separation of the two masses (see Figure 1). Although these points are all potential maxima, stable motion is possible around L_4 and L_5 due to the Coriolis force

[9].

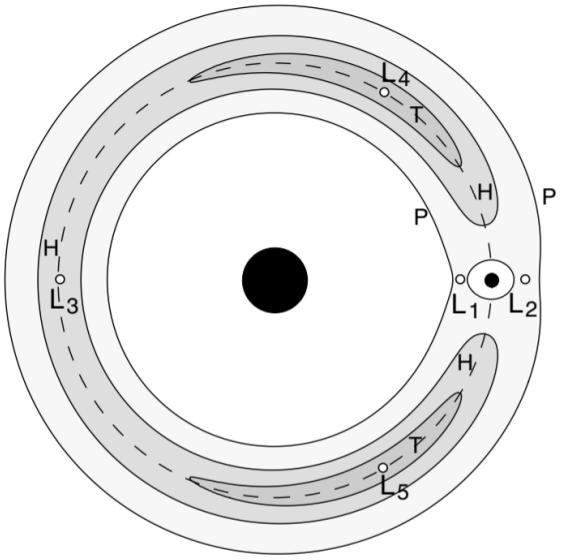


Figure 1: *The location of the five Lagrange equilibrium points in the circular-restricted three-body problem. The solar and planetary masses are denoted by the large and small filled circles, and the letters P, H, and T denote passing, horseshoe, and tadpole orbits respectively. Note that the two massive bodies form an equilateral triangle with each of the L_4 and L_5 points. Reproduced from Marzari et al. [10]*

Figure 1 depicts orbits about L_4 and L_5 (known as tadpole orbits), as well as horseshoe orbits between Lagrange points, described by Murray et al. [11].

This report will focus on tadpole orbits, a well-documented feature of Trojan orbits [12] [13] and of the restricted three-body problem in general. Their distinctive shape results from a long-period motion about the equilibrium point combined with a short-period oscillation due to the Keplerian motion of the asteroid.

Szebehely et al. [14] predict that this short period tends to the planetary period in the small planetary mass limit, while the long period is given by

$$T_{long} = T_P \sqrt{\frac{4}{27\mu_2}} \tag{1}$$

where $\mu_2 = m_2/(m_1+m_2)$ and T_P is the period of planetary orbit.

2.2 Theoretical Model

The three-body problem, where the dynamics of three interacting bodies are determined from their initial positions and velocities, has no analytical (closed-form) solution in the general case [15].

In this report, I will consider the circular, restricted, three-body problem, where two of the bodies move in circular, coplanar orbits about their common centre of mass (CoM), unaffected by the negligible mass of the third body. I will also assume that all interactions are via Newtonian gravity.

The system of differential equations determines the position and velocity of the asteroids, with two equations per spatial coordinate.

$$\frac{dr_i}{dt} = v_i, \frac{dv_i}{dt} = g_i, \quad i = x, y \tag{2}$$

In this, g_i is given by:

$$\mathbf{g} = -\frac{GM_s}{|\mathbf{r}-\mathbf{r}_s|^3}(\mathbf{r}-\mathbf{r}_s) - \frac{GM_p}{|\mathbf{r}-\mathbf{r}_p|^3}(\mathbf{r}-\mathbf{r}_p) \tag{3}$$

where the subscripts *s* and *p* refer to solar and planetary properties respectively.

We may also consider a frame rotating at the same velocity as the massive bodies. As there is 1:1 orbital resonance between Jupiter and the asteroids, all three bodies are stationary in this frame. This significantly increases the accuracy of numerical simulations, as the exact solution is stationary with no explicit time dependence, rather than requiring an infinite power series [16], [17].

When transforming into this rotating non-inertial frame, g_i gains an additional virtual force term with coupling between the spatial coordinates. This is given below as the sum of the centripetal and Coriolis forces:

$$\Delta g_i = \Omega^2 r_i - 2[\mathbf{\Omega} \times \mathbf{v}]_i \tag{4}$$

where Ω is the angular speed of the rotating frame, and $\mathbf{v}$ is the velocity of the asteroid within this frame.

2.3 Symmetry

This problem contains a number of symmetries, which were employed to simplify the problem and reduce the computational load. The Trojan and

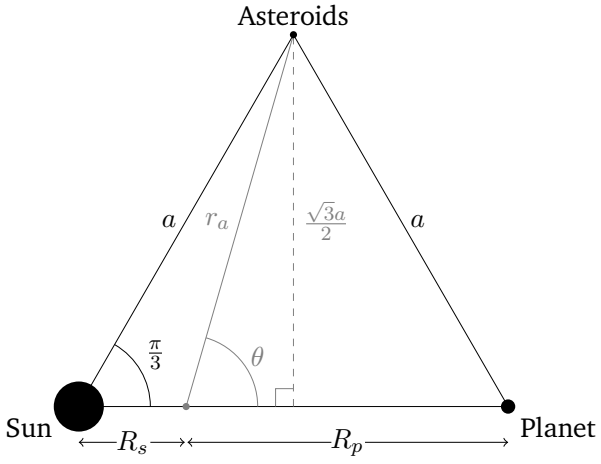


Figure 2: *A geometric depiction of the three-body system, in the case where the planet has a mass equal to one-third of the sun, and asteroids are considered at the L_4 point. R_s and R_p denote the (fixed) radii from the centre of mass (the grey point) to the Sun and the planet respectively, while r_a denotes the radius of the asteroids.*

Greek asteroids are in equivalent positions, so experience the same forces, and the system has rotational and inversion symmetry, so the choice of initial point and orbit direction is arbitrary. Therefore, only the Greeks, orbiting counter-clockwise with perturbations applied at t = 0, need be considered.

2.4 Orbit Geometry

As the three bodies considered here form an equilateral triangle in the initial equilibrium state, as depicted in Figure 2, we can easily derive the polar coordinates of each body with respect to the centre of mass about which the bodies orbit.

Using standard trigonometric relations, it is simple to show that the values r_a and θ are given by:

$$r_a = \sqrt{a^2 + R_s R_p}, \quad \theta = \tan^{-1}\left(\frac{a\sin(\frac{\pi}{3})}{R_p - \frac{a}{2}}\right) \tag{5}$$

Furthermore, the Lagrange point in Cartesian coordinates based about the CoM is easily found to be:

$$(x, y) = \left(R_p - \frac{a}{2}, \frac{\sqrt{3}a}{2}\right) \tag{6}$$

Finally, equating the gravitational and centripetal forces on the planet allows the derivation of its (and all other bodies') orbital velocity:

$$\Omega = \sqrt{\frac{G(M_s + M_p)}{a^3}} \tag{7}$$

3 Methodology

This system of coupled first-order ordinary differential equations (ODEs) was solved using the scipy *solve_ivp* function. The time span was taken as 100 orbits (with 100 points sampled per orbit) unless otherwise stated; this corresponds to 1185 Earth years. Rescaled solar system units are used for mathematical ease, so distances are measured in astronomical units (AU), time in Earth years and mass in multiples of the solar mass, to prevent floating point overflows due to the magnitude of the quantities considered in SI units.

The wander was defined as the maximum distance of the asteroid from the initial point during the orbit (for small perturbations this is also approximately the separation from the Lagrange point). Initial conditions are defined by the Lagrange point in each frame, with the initial velocity in the stationary frame defined by the period of Jupiter's orbit, and split into Cartesian components.

3.1 Integration Method

Within *solve_ivp*, the default solver is RK45 (an explicit Runge-Kutta method of order 5(4) [18]) which gives a deviation in asteroid position (from the Lagrange point) in the order of 10^{-4} AU in the rotating frame over 50 orbits. This is larger than expected, suggesting that the system of equations may require an unreasonably small step-size for numerical stability in this method, even in regions where the solution curve is smooth [19]. Such systems are known as stiff, and solvers designed for these systems typically do more work per step, allowing them to take much larger steps, and have improved numerical stability compared to non-stiff solvers, such as RK45 [20].

Therefore the stiff Radau solver (an implicit Runge-Kutta method from the Radau IIA family of order 5 [21]) is used for increased stability [22], and achieves a deviation in asteroid position in the order of 10^{-13} AU instead. This also ensures stability in the stationary frame, with deviations of 0.76% in asteroid separation from Jupiter over 10^3 years, compared to 53% for the best non-stiff solvers.

3.2 Programme Structure

Global constants, such as solar mass, and sun–planet separation, along with derived values from these, such as orbital period and solar radius from the CoM, are given in an importable python module, "*constants*".

Functions to evaluate these coupled differential equation systems are defined in the module "*orbits*", while additional functions to evaluate the wander during the orbit (under different sampling routines) are implemented in "*wander*". Complete code listings for these files are given in Appendix C. Further files then import these modules and produce the plots given in this report, fully detailed in Appendix B.

When varying planetary mass, I considered it preferable to avoid reconstructing all functions to take planetary mass as an argument, as this requires re-evaluating all initial derived constants in the "*orbits*" module. Therefore, I iterate over alternative planetary masses, re-defining constant values in each instance, and directly import the required functions to compute the orbit.

3.3 Performance

Sampling 100 points per orbit for 100 orbital periods takes a mean time of 16.97 ± 34 ms, with sublinear scaling for sampling rate and approximate linear scaling with orbit number up to the array memory limit, achieved through the optimised integration routines within *solve_ivp*. Computing the wander over position/velocity space with larger perturbations was, however, more time consuming, averaging 2.04 ± 0.16 s per point with the same parameters.

The main computational load was within the *solve_ivp* function, which is already optimised well beyond the capabilities of the author. However, assumptions made in Section 2.3, such as considering only one group of asteroids, reduces this time somewhat, and vectorisation of other aspects of the *orbit* functions reduced the running time of the whole module. These approaches were not taken universally however, due to the size of the arrays required, and the dominant effect of the ODE solver on the running time; instead, the orbit number was reduced when only a comparison of the maximum wander was required.

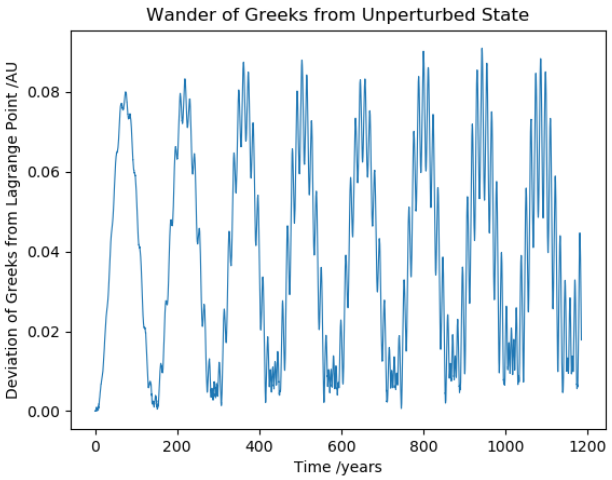


Figure 3: *The wander of Greek asteroids from the Lagrange point in the stationary frame. Note the two oscillation components and constant maximum oscillation amplitude over time, demonstrating this point is indeed stable.*

3.4 Fourier Analysis

For periodic oscillations and orbits, the Fast Fourier Transform (FFT) implemented in *scipy* is used to obtain the period, with *find_peaks* in *scipy.signal* identifying the exact values. The errors were estimated using the associated peak widths of each frequency component.

4 Results

4.1 Unperturbed Stability of Lagrange Points

Without perturbations applied, the Greeks' orbit has a maximum deviation from L_4 of 4.68×10^{-13} AU over 100 orbits (1185 years). This value is unchanged if 1000 orbits are considered instead, confirming the stability of this Lagrange point.

In the stationary frame, this wander from the (now moving) Lagrange point is depicted in Figure 3. The wander oscillates with a magnitude of 9.10×10^{-2} AU, over a period of 148 ± 4 years. This is modulated with a faster oscillation component of 11.85 ± 0.27 years, equivalent to the orbital period of the asteroids.

These much more significant deviations are due to time-dependence in the exact solution, as detailed in Section 2.2. Energy can also be evaluated, and conserved, in this inertial frame; asteroid specific energy varies within only 0.113% of the initial value, and with a similar periodicity to wander.

Energy is also negative, confirming the asteroids are located within a bound orbit.

Animations produced to demonstrate the orbit in the stationary frame are included in Supplementary Material I-II. Animation I depicts the orbit as evaluated by the Radau solver, while II depicts it with LSODA, demonstrating the drift present over time with non-stiff solvers.

4.2 Wander Analysis

Wander from the initial point was calculated for random perturbations with a maximum magnitude of 1% of the displacement from the origin. The perturbation components parallel and perpendicular to the position vector from the CoM (hereafter referred to as radial and tangential components respectively) were considered separately. By considering these perturbations across position space, it was clear that the wander is fully determined by the radial component, with no tangential dependence (as shown in Figure 4).

Figure 4a demonstrates a polynomial dependence on perturbation size in the radial direction, with a negligible constant term. While a quadratic has been fitted here, the possibility of higher order terms could not be eliminated, as increasing perturbation size beyond 0.06 AU can lead to unstable orbits.

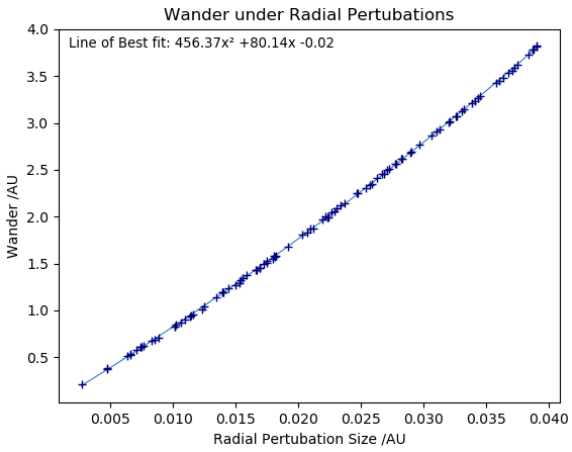
We may consider the wander resulting from perturbations in position and velocity space. Figure 5a clearly shows dependence on radial position perturbations only, while Figure 5b shows a similar dependence on tangential velocity perturbations.

4.3 Orbit Types

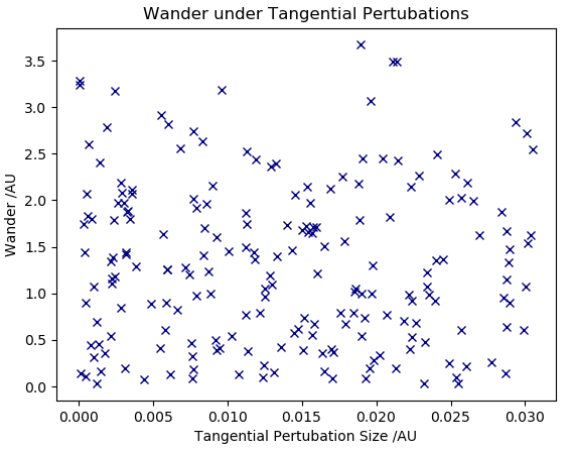
Figure 6 displays orbits resulting from small perturbations in the radial and tangential directions.

The tadpole orbit in Figure 6a consists of two oscillating components, as described in Section 2.1. The short period, measured at 11.85 ± 0.61 years, is in excellent agreement with planetary orbital period as expected; meanwhile the long period is measured to be 148 ± 5 years, consistent with the analytical value of 144 years.

For a narrow range of larger perturbations, stable horseshoe orbits encompassing both Lagrange points may also be observed, as depicted in Figure 7. This has a full period of 353 years, in agreement with Taylor et al.'s numerical result of 358 ± 7 years [23].



(a) *Radial*



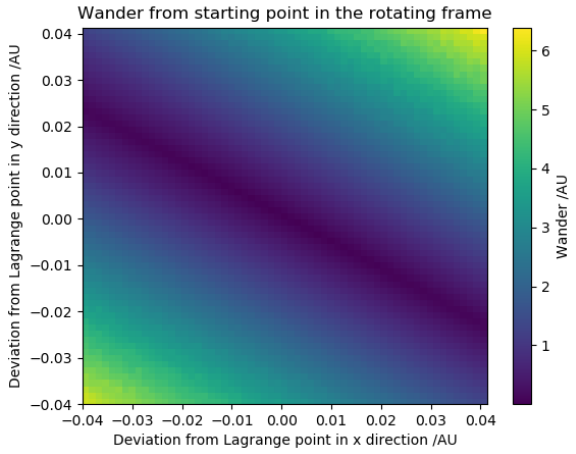
(b) *Tangential*

Figure 4: *Wander from an initial perturbation with maximum relative magnitude of 0.1% of the displacement from the origin, given in radial and tangential directions.*

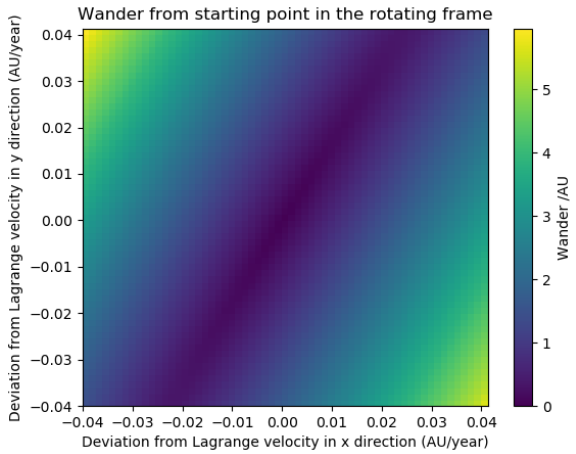
4.4 Perturbations in Z direction

Perturbations in the Z direction are aligned with the angular velocity vector for the rotating frame, so experience no virtual forces in this direction, and oscillate under the influence of gravity alone. Considering small perturbations in the Z direction (so that the distance from the CoM can be considered constant), the period of such oscillations tend to the orbital period of the rotating frame under Newtonian gravity, as detailed in Appendix A. These oscillations are observed to be approximately sinusoidal, with a measured period of 11.85 ± 0.48 years, in strong agreement with the theoretical predictions.

The overall wander, however, is observed to follow Figure 8. It is suggested that, while the maximum deviation from the Lagrange point in the Z direction is simply twice the initial perturbation,



(a) *Position*



(b) *Velocity*

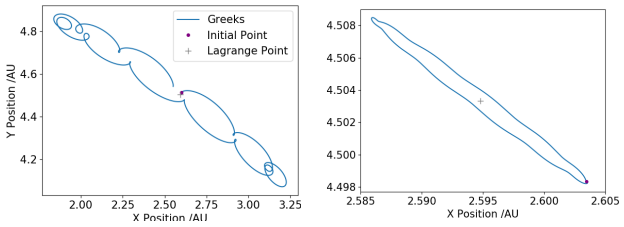
Figure 5: *Wander from an initial point in position and velocity space, only calculated over 50 orbits with 30 points per orbit to reduce computational load. Note the dependence on radial position and tangential velocity perturbation components.*

wander in the XY plane increases quadratically. This can be initiated from a radial displacement due to the reduced gravitational force under the Z perturbation, demonstrated in Figure 9 or, more significantly, from an initial perturbation in the XY plane. However, the oscillation of gravitational forces due to perturbations in the Z direction make this problem significantly more complicated and are a possible cause of the observed deviations from the quadratic relations.

4.5 Variation of Planetary Mass

The variation of wander with planetary mass is given in Figure 10.

For small planetary masses (up to approximately



(a) *Radial* **(b)** *Tangential*

Figure 6: *Orbits from an initial perturbation of magnitude 0.01 AU from the origin, in radial and tangential directions, over 12 orbital periods. Note the tadpole features resulting from the radial perturbation, and the significantly larger wander than from the tangential perturbation.*

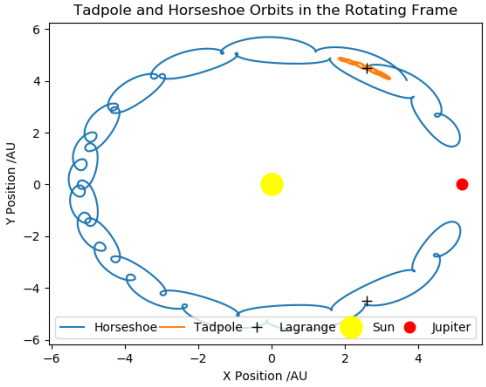


Figure 7: *Horseshoe and tadpole orbits in the rotating frame, from radial perturbations of 0.07 and 0.01 AU respectively, over 30 orbital periods.*

$M_p = 0.01M_s$), the wander follows a $M_P^{-\frac{1}{2}}$ dependence, tending towards the initial perturbation for larger masses. However, wander increases unbounded and orbits become unstable beyond $M_p = 0.04M_s$, as depicted in Figure 11, in agreement with theoretical predictions by Darwin [24].

5 Conclusion

This report demonstrates the stability of Lagrange points L_4 and L_5, with a maximum deviation from the Lagrange points of 4.68×10^{-13} AU over 1000 orbits (11,852 years), in a frame rotating with the asteroids and without perturbations applied. Stability under perturbations in both the XY plane and in the Z direction is also demonstrated, with a finite wander from these points independent of orbit duration. It was possible to further quantify this wander, suggesting a strong quadratic dependence on radial perturbation magnitude and

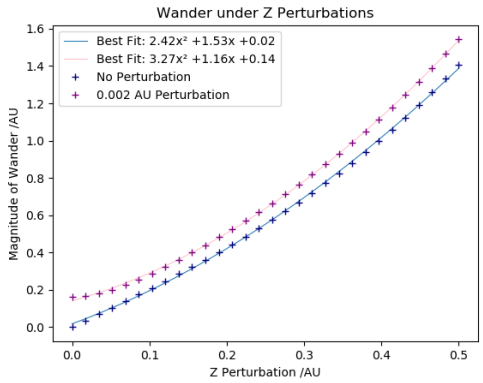


Figure 8: *Variation of wander with increasing Z perturbations. A strong linear component is observed initially in the un-perturbed case, but then tends towards the perturbed quadratic case, while the linear offsets result from the initial perturbation in the XY plane.*

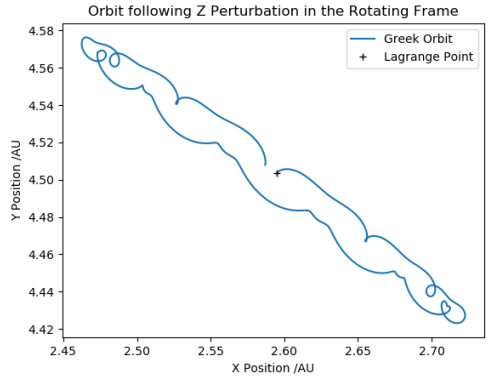


Figure 9: *Orbit of Greek asteroids following a 0.2 AU perturbation in the positive Z direction, over 12 orbital periods. Note the initial outwards radial displacement despite the lack of perturbation in the XY plane, and subsequent tadpole-like orbit.*

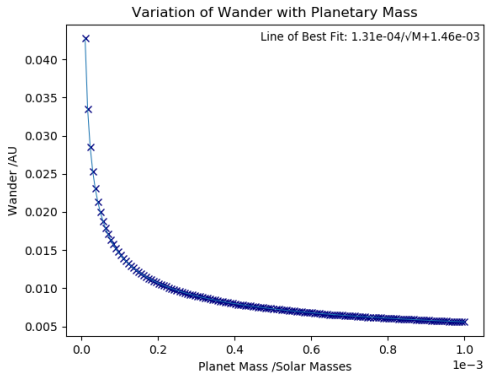


Figure 10: *Variation of maximum wander with planetary mass, under a 0.001 AU radial perturbation and 0.001 AU/year tangential boost. Note the tendency*

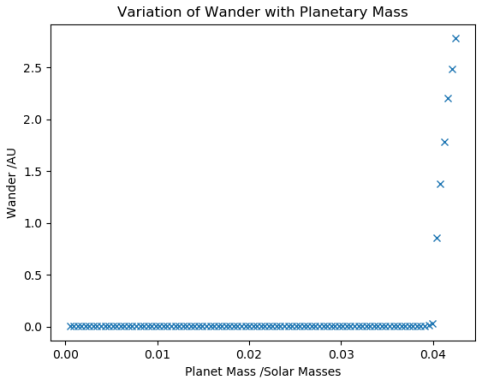


Figure 11: *Variation of maximum wander with planetary mass, under a 0.01 AU radial perturbation and 0.01 AU/year tangential boost. Note the orbit stability up to $M_p = 0.04M_s$, and instability beyond this.*

independence of tangential perturbations, which did not grow beyond the initial perturbation size. It was also possible to replicate both tadpole and horseshoe orbits from radial perturbations, and frequency components were obtained for these orbits in agreement with previous literature. It was also found that orbits become unstable when the planetary–solar mass ratio is greater than 0.04, in agreement with theoretical predictions, and there is little mass dependence below this point.

Further work is required to evaluate coefficient values for polynomial relationships observed, and to demonstrate a theoretical basis for such dependencies. Further research would also be beneficial on the categorisation of different orbit types, and examination of initial conditions corresponding to formation mechanisms of these orbits.

References

[1] S. B. Nicholson, "The trojan asteroids", Leaflet of the Astronomical Society of the Pacific **8**, 239 (1961).

[2] Minor Planet Centre, International Astronomical Union, *Trojan minor planets*, (2020) `https://minorplanetcenter.net/iau/lists/Trojans.html`.

[3] R. P. D. Sisto, X. S. Ramos, and T. Gallardo, "The dynamical evolution of escaped jupiter trojan asteroids, link to other minor body populations", Icarus **319**, 828–839 (2019).

[4] D. Nesvorný, D. Vokrouhlický, W. F. Bottke, and H. F. Levison, "Evidence for very early migration of the solar system planets from the patroclus–menoetius binary jupiter trojan", Nature Astronomy **2**, 878–882 (2018).

[5] M. E. Brown, "The 3–4 micrometer spectra of jupiter trojan asteroids", The Astronomical Journal **152**, 159 (2016).

[6] T. Okada, T. Iwata, J. Matsumoto, J.-P. Bibring, S. Ulamec, R. Jaumann, R. Nakamura, H. Yano, Y. Kebukawa, J. Aoki, Y. Kawai, K. Terada, M. Toyoda, M. Ito, K. Yabuta, H. Yurimoto, Y. Saito, S. Yokota, C. Okamoto, S. Matsuura, K. Tsumura, D. Yonetoku, T. Mihara, A. Matsuoka, R. Nomura, T. Hirai, N. Grand, H. Cottin, L. Thirkell, C. Briois, T. Saiki, H. Kato, O. Mori, and J. Kawaguchi, "Science and exploration of a jupiter trojan asteroid in the solar-power sail mission", in 48th lunar and planetary science conference (Mar. 2017).

[7] H. F. Levison and Lucy Science Team, "Lucy: Surveying the Diversity of the Trojan Asteroids, the Fossils of Planet Formation", in Lunar and planetary science conference, Lunar and Planetary Science Conference (Mar. 2016), p. 2061.

[8] J.-L. Lagrange, "Essai sur le probléme des trois corps", Prix de l'Académie Royale des Sciences de Paris **IX** (1772).

[9] J. Lissauer and C. Murray, "Solar system dynamics: regular and chaotic motion", in *Encyclopedia of the solar system (third edition)* (2014).

[10] F. Marzari, H. Scholl, C. Murray, and C. Lagerkvist, "Origin and evolution of trojan asteroids", Asteroids III (2002).

[11] C. D. Murray and S. F. Dermott, *Solar system dynamics* (Cambridge University Press, 1999), pp. 95–102.

[12] B. Garfinkel, "Theory of the trojan asteroids, iv", Celestial Mechanics **30**, 373–383 (1983).

[13] S. F. Dermott and C. Murray, "The dynamics of tadpole and horseshoe orbits", Icarus **48**, 1–11 (1981).

[14] V. Szebehely and E. Grebenikov, "Theory of orbits-the restricted problem of three bodies", Soviet Astronomy **13**, 364 (1969).

[15] J. Barrow Green, "The princeton companion to mathematics", in *The three-body problem*, edited by T. Gowers (Princetown University Press, 2008) Chap. V, pp. 726–728.

[16] N. Guglielmi and E. Hairer, "Implementing radau IIA methods for stiff delay differential equations", Computing **67**, 1–12 (2001).

[17] R. LeVeque, *Finite difference methods for ordinary and partial differential equations : steady-state and time-dependent problems* (Society for Industrial and Applied Mathematics, Philadelphia, PA, 2007), pp. 131–153.

[18] J. Dormand and P. Prince, "A family of embedded runge-kutta formulae", Journal of Computational and Applied Mathematics **6**, 19–26 (1980).

[19] J. D. Lambert, *Numerical methods for ordinary differential systems : the initial value problem* (Wiley, Chichester New York, 1991), pp. 217–220.

[20] G. D. Byrne and A. C. Hindmarsh, "Stiff ODE solvers: a review of current and coming attractions", Journal of Computational Physics **70**, 1–62 (1987).

[21] E. Hairer, *Solving ordinary differential equations II* (Springer, Berlin, 2010).

[22] R. Frank, J. Schneid, and C. W. Ueberhuber, "Stability properties of implicit runge–kutta methods", SIAM Journal on Numerical Analysis **22**, 497–514 (1985).

[23] D. B. Taylor, "Horseshoe periodic orbits in the restricted problem of three bodies for a sun-Jupiter mass ratio", Astronomy and Astrophysics **103**, 288–294 (1981).

[24] G. H. Darwin, "Periodic orbits", Acta Mathematica **21**, 99–242 (1897).

Appendix A Z Perturbation Period

Figure 12 depicts a Z perturbation out of the XY plane, from the L_4 point.

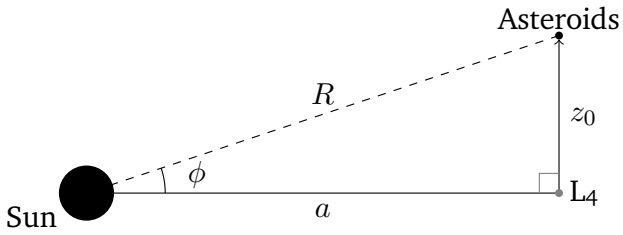


Figure 12: *A geometric description of asteroids at the L_4 point undergoing a perturbation of magnitude z_0 out of the XY plane. The fixed equilibrium distance between the asteroids and the Sun is denoted by a, while R represents the separation between the asteroids and the Sun as a result of the perturbation*

As the asteroid is equidistant from the Sun and the planet, the magnitudes of these forces depend on these bodies' mass only, and gravitational force components from both massive bodies are similarly orientated in the Z direction. The total gravitational force acting on the asteroid is therefore given by:

$$F = -\frac{G(M_s + M_p)M_a}{R^2} \tag{8}$$

where M_s, M_p and M_a represent the solar, planetary and asteroid masses respectively. Resolving this force in Cartesian components, the Z component is given by:

$$F_z = Fcos(\phi) = F\frac{z}{R} = -\frac{G(M_s + M_p)M_a z}{R^3} \tag{9}$$

In conjunction with Newton's Second Law, we therefore obtain a second order differential equation for z:

$$\ddot{z} + \frac{G(M_s + M_p)}{R^3} z = 0 \tag{10}$$

which can be treated as simple harmonic motion in the small perturbation limit where $R \approx a$. This has a angular frequency equal to (7):

$$\Omega = \sqrt{\frac{G(M_s + M_p)}{a^3}} \tag{11}$$

Appendix B Program Structure

Global constants, along with derived values from these, are given in a python module "*constants*", from which relevant variables are imported in all other scripts.

Functions to evaluate the coupled differential equation systems are defined in the module "*orbits*", while additional functions to evaluate the wander during the orbit are implemented in "*wander*". These modules are both imported into the scripts detailed in Table 1 for plotting and analysis, and given in the listings in Appendix C.

Table 1: *Program files used to analyse and plot data for each section, with a description of their role.*

Section	Description	Code File
3.3	ODE solver performance	*lagrange_stability*
3.4	Fourier analysis	*lagrange_stability*
4.1	Wander from unperturbed initial conditions	*lagrange_stability*
4.1	Animations of unperturbed orbits	*stat_frame_video*
4.2	Perturbations restricted to the XY plane	*perturbation_2D*
4.3	Plotting of orbit types in XY plane	*perturbation_2D*
4.4	Perturbations in the Z direction	*perturbation_3D*
4.5	Variation of planetary mass	*planet_mass_variation*

Appendix C Code Listings

C.1 *constants.py*

```
import math
import numpy as np

# USER DEFINED CONSTANTS
G = 4 * np.pi ** 2  # gravitational constant in astronomical units
M_S = 1  # solar mass
M_P = 0.001  # planetary mass as a fraction of solar mass
R = 5.2  # average planetary radius from sun in AU

PRECISION = 100  # evaluation points per orbit
ORBIT_NUM = 100  # number of orbits

# DERIVED CONSTANTS
solar_rad = R * M_P / (M_S + M_P)  # distance from origin to sun in CoM frame
planet_rad = R * M_S / (M_S + M_P)
period = math.sqrt(R ** 3 / (M_S + M_P))
omega = 2 * np.pi / period  # angular velocity of frame
time_span = np.linspace(0, ORBIT_NUM * period, int(ORBIT_NUM * PRECISION))
lagrange = (planet_rad - R / 2, R * math.sqrt(3) / 2, 0)
greek_theta = np.arctan((R * math.sqrt(3) / 2) / (R / 2 - solar_rad))
```

C.2 *orbits.py*

```
import math
import numpy as np
from scipy import integrate
import matplotlib.pyplot as plt

from constants import G, M_S, M_P, R, ORBIT_NUM, PRECISION  # User defined constants
from constants import (
    solar_rad,
    planet_rad,
    period,
    omega,
    time_span,
)  # Derived constants

# DERIVED QUANTITIES
```

```
# refers to greeks in this case, symmetric to trojans
greek_theta = np.arctan((R * math.sqrt(3) / 2) / (R / 2 - solar_rad))
greek_rad = math.sqrt(R ** 2 - solar_rad * planet_rad)
greek_v = omega * (greek_rad)

# Initial conditions for Greeks, given as equillibrium state
cos, sin = np.cos(greek_theta), np.sin(greek_theta)
rcos, rsin = greek_rad * cos, greek_rad * sin
vcos, vsin = greek_v * cos, greek_v * sin

# EXACT SOLUTIONS FOR POSITION
def solar_pos(t=0):
    """Position of sun in 3-dimensions at time t

    Note that z coordinate is fixed to 0,
    so movement is constrained to xy plane in this exact solution
    """
    return np.array([-solar_rad * np.cos(omega * t), -solar_rad * np.sin(omega * t), 0])


def planet_pos(t=0):
    """Position of planet in 3-dimensions at time t

    Note that z coordinate is fixed to 0,
    so movement is constrained to xy plane in this exact solution
    """
    return np.array([planet_rad * np.cos(omega * t), planet_rad * np.sin(omega * t), 0])


def lagrange_pos(t=0):
    """Position of Greeks' lagrange point in 3-dimensions at time t

    Note that z coordinate is fixed to 0,
    so movement is constrained to xy plane in this exact solution"""
    return np.array(
        [
            greek_rad * np.cos(omega * t + greek_theta),
            greek_rad * np.sin(omega * t + greek_theta),
            0,
        ]
    )


# ODE SOLVER IN THE ROTATING FRAME
def rot_derivatives(t, y):
    """Gives derivative of each term of y at time t in the rotating frame

    y should be of the form (x_pos, y_pos, z_pos, x_vel, y_vel, z_vel)
    """

    position, velocity = np.array(y[0:3]), np.array(y[3:6])

    # Factors defined for simplicity in acceleration term:
    solar_dr3 = np.linalg.norm(solar_pos(0) - position) ** 3
    planet_dr3 = np.linalg.norm(planet_pos(0) - position) ** 3

    virtual_force = (
        position[0] * omega ** 2 + 2 * omega * velocity[1],
```

```
        position[1] * omega ** 2 - 2 * omega * velocity[0],
        0,
    )

    acceleration = (
        -G
        * (
            M_S * (position[0] - solar_pos(0)[0]) / solar_dr3
            + M_P * (position[0] - planet_pos(0)[0]) / planet_dr3
        )
        + virtual_force[0], # x component
        -G * (M_S * position[1] / solar_dr3 + M_P * position[1] / planet_dr3)
        + virtual_force[1], # y component
        -G * (M_S * position[2] / solar_dr3 + M_P * position[2] / planet_dr3)
        + virtual_force[2], # z component
    )

    return np.concatenate((velocity, acceleration))


def rotating_frame(y0_rot=(rcos, rsin, 0, 0, 0, 0)):
    """Gives position and velocity of asteroids in the rotating frame

    Uses scipy solve_ivp method with Radau, taking an input in the form
    of (x_pos, y_pos, z_pos, x_vel, y_vel, z_vel) for the initial state.
    This takes a default value of the equillibrium position for y0_rot
    if this is not given.
    """

    return integrate.solve_ivp(
        fun=rot_derivatives,
        method="Radau",  # Or LSODA for non-stiff alternative
        t_span=(0, ORBIT_NUM * period),
        y0=y0_rot,
        t_eval=time_span,  # selects points for storage
    )


# ODE SOLVER IN THE STATIONARY FRAME
def stat_acceleration(position, t):
    """Gives acceleration in stationary frame at a given position and time"""
    # factors defined for convenience
    solar_dr3 = np.linalg.norm(solar_pos(t) - position) ** 3
    planet_dr3 = np.linalg.norm(planet_pos(t) - position) ** 3

    return (
        -G
        * (
            M_S * (position[0] - solar_pos(t)[0]) / solar_dr3
            + M_P * (position[0] - planet_pos(t)[0]) / planet_dr3
        ),
        -G
        * (
            M_S * (position[1] - solar_pos(t)[1]) / solar_dr3
            + M_P * (position[1] - planet_pos(t)[1]) / planet_dr3
        ),
        -G * (M_S * position[2] / solar_dr3 + M_P * position[2] / planet_dr3),
    )
```

```
def stat_derivatives(t, y):
    """Gives derivative of each term of y at time t in the stationary frame

    y should be of the form (x_pos, y_pos, z_pos, x_vel, y_vel, z_vel)
    """
    position, velocity = np.array(y[0:3]), np.array(y[3:6])
    return np.concatenate((velocity, stat_acceleration(position, t)))


def stationary_frame(y0_stat=(rcos, rsin, 0, -vsin, vcos, 0)):
    """Gives position and velocity of asteroids in the stationary frame

    Uses scipy solve_ivp method with LSODA, taking an input in the form
    of (x_pos, y_pos, z_pos, x_vel, y_vel, z_vel) for the initial state.
    This has a default value of the equillibrium state if not given.
    """
    return integrate.solve_ivp(
        fun=stat_derivatives,
        method="Radau",  # Or LSODA for non-stiff alternative
        t_span=(0, ORBIT_NUM * period),
        y0=y0_stat,
        t_eval=time_span,
    )


def specific_energy(t, y):
    """ Gives energy per unit mass of asteroids

    t is the time after the start of the orbit
    y has six components; first three for position, second three for velocity
    These are in the form of the output of a scipy integrate function
    """

    radius_p = np.linalg.norm(y[0:3] - planet_pos(t), axis=0)
    radius_s = np.linalg.norm(y[0:3] - solar_pos(t), axis=0)
    velocity = np.linalg.norm(y[3:], axis=0)
    return -G * M_P / radius_p - G * M_S / radius_s + 1 / 2 * velocity ** 2
```

C.3 *wander.py*

```
import numpy as np

import orbits
from constants import omega, lagrange, greek_theta  # Derived constants

# INITIAL CONDITIONS
initial_cond_rot = np.array((lagrange[0], lagrange[1], 0, 0, 0, 0))
# in rotating frame
initial_cond_stat = np.array(
    (lagrange[0], lagrange[1], 0, -omega * lagrange[1], omega * lagrange[0], 0)
)

# DEFINED FUNCTIONS
def perturb(initial_cond, max_pertubation_size=0.01):
    """ Returns perturbed version of initial conditions array
```

```
    init_cond is the initial conditions, submitted as a numpy array
    pertubation_size is the relative magnitude of the perturbation, measured as
    the percentage deivation from the lagrange point (for position)

    Perturbations are only chosen randomly from within a circle of radius one,
    to give a more uniform radial distribution than from a square.
    Note that this function will give a different value each time it is run
    """
    rand_array = np.random.uniform(-1, 1, (np.shape(initial_cond)))
    while np.linalg.norm(rand_array[0:2]) > 1: # reject points outside circle
        rand_array = np.random.uniform(-1, 1, (np.shape(initial_cond)))
    return initial_cond + initial_cond * (
        max_pertubation_size * rand_array
    )  # random between -1 and 1, selects points in circle of radius 1


def rand_sample(max_pertubation_size, samples):
    """Returns maximum deviation over orbit in the
    rotating frame for given number of random samples

    This calculates the distange from the initial point (not the lagrange point)
    It returns a samples*3 array, giving the radial and tangential components
    of the pertubation, and then the maximum wander over this timespan

    pertubation_size determies the relative magnitude of the perturbation
    Samples denotes the number of random pertubations sampled at the given size
    """
    output = np.zeros((samples, 3))  # size of pertubation; size of wander
    for n in range(samples):
        initial_cond = perturb(initial_cond_rot, max_pertubation_size)
        orbit = orbits.rotating_frame(initial_cond)
        sample_wander = np.zeros((len(orbit.t)))

        perturb_theta = np.arctan(
            (initial_cond[1] - initial_cond_rot[1])
            / (initial_cond[0] - initial_cond_rot[0])
        )

        for i in range(len(sample_wander)):
            sample_wander[i] = np.linalg.norm(
                orbit.y[0:3, i] - initial_cond[0:3]
            )  # deviation in pos only
        output[n, 0] = np.linalg.norm((initial_cond[0:3] - lagrange[0:3])) * np.abs(
            np.cos(greek_theta - perturb_theta)
        )  # radial component of pertubation
        output[n, 1] = np.linalg.norm((initial_cond[0:3] - lagrange[0:3])) * np.abs(
            np.sin(greek_theta - perturb_theta)
        )  # tangential component of pertubation
        output[n, 2] = np.max(sample_wander)  # overall magnitude of wander
    return output


def initial_point(pertubation, pertubation_type="position"):
    """Maximum wander over orbit in the rotating frame for given initial point

    Wander is the maximum deviation from the initial point
    (not the lagrange point) over this timespan
    perturbation is the initial point in 2D position/velocity space
```

```
    perturbation_type can be "position" or "velocity"
    """
    if pertubation_type == "position":
        initial_cond = initial_cond_rot + np.array(
            (pertubation[0], pertubation[1], pertubation[2], 0, 0, 0)
        )  # add position pertubation
    elif pertubation_type == "velocity":
        initial_cond = initial_cond_rot + np.array(
            (0, 0, 0, pertubation[0], pertubation[1], pertubation[2])
        )  # add velocity pertubation
    else:
        raise ValueError(
            "Unknown perturbation_type: Please use 'position' or 'velocity'"
        )

    orbit = orbits.rotating_frame(initial_cond)
    wander_t = np.zeros((len(orbit.t)))
    for i in range(len(orbit.t)):
        wander_t[i] = np.linalg.norm(
            orbit.y[0:3, i]
            - initial_cond[0:3]
            # - lagrange[0:3]
        )  # deviation in pos only
    return np.max(wander_t)
```