UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

MASTER THESIS

Astrophysical Parameter Determination Using Chaotic Orbital Regions

Jorge Nicolás Álvarez

SUPERVISED BY

Josep J. Masdemont Soler

Universitat Politècnica de Catalunya
Master in Aerospace Science & Technology
February 2018

Astrophysical Parameter Determination Using Chaotic Orbital Regions

BY

Jorge Nicolás Álvarez

DIPLOMA THESIS FOR DEGREE

Master in Aerospace Science and Technology

AT

Universitat Politècnica de Catalunya

SUPERVISED BY:

Josep J. Masdemont Soler
Mathematics Department

ABSTRACT

The goal of this thesis is to show that chaotic orbits of a dynamical system are advantageous (compared with regular ones) to determine the characteristic parameters of the system, and to give an estimate about the behaviour of the error as a function of the number of observations.

The utility of chaotic trajectories for parameter determination in a complex dynamical system such as the restricted full three-body problem is assessed. The model takes into consideration the motion of a particle of negligible mass around a binary system where the main body is modeled as a tri-axial ellipsoid, taking into account its mass distribution, and the smaller one as point mass sphere. It is studied by means of a program coded in Python, capable of simulating the binary system gravity field and propagating trajectories in three dimensions around it. It returns verification values such as the Jacobi constant, the state transition matrix and crashing event assessment allowing lots of flexibility in the parameter settings.

A combination of Poincaré maps, which allow to study the periodicity of a trajectory in a visual way; and mathematical tools such as the finite time Lyapunov exponents and the Pesin entropy formula, which indicate how sensitive a trajectory is to initial conditions (main condition for chaotic behaviour) is used. They return similar and reliable values providing redundancy and robustness to the procedure.

For the determination of parameters we have developed a procedure, based on least squares approximations, that determines how much the uncertainty in the parameter determination of the system decreases as more measurements are taken. There is a strong correlation between the error in the parameter determination and the length of the trajectory, obviously, the error is reduced as more measurements are taken. If N denotes the number of measurements, for a regular orbit the uncertainty in the determination behaves as $N^{-\frac{1}{2}}$. For chaotic orbits we have found scalings of up to N^{-3}, which are better than the scalings found in previous works for simple systems, which were between N^{-1} and N^{-2}.

ACKNOWLEDGEMENTS

I would like to express my most sincere gratitude to all the members in the Advanced Concepts Team of the European Space Agency in ESTEC (The Netherlands) for granting me the opportunity to have fun, hang out, work and develop most of this work together with them. In particular to Leopold Summerer, Dario Izzo, Alexander Wittig, Jai Grover, Alessio Mereta and, above all, Alexander Hickey who taught me special British expressions to assess nonsense scientific results and trusted me to develop and demonstrate his original and chaotic idea allowing me to live a life-changing adventure which has just started and who knows where it will lead me.

Back in Barcelona I want to thank Josep J. Masdemont and Gerard Gómez for supervising the project and allowing me to be part of their team sharing and providing new ideas.

Last but not least, I really appreciate the support from my always present family, wherever I may roam, parents, grandparents, uncles, cousins and sister; my partner by that time who was my strongest support during my first weeks in the Netherlands; and the marvelous friends I met in Leiden from all around Europe, sharing lots of wonderful experiences, knowledge, culture and Estonian lies, gifts and promises.

CONTENTS

Abstract i

Acknowledgements iii

Introduction 1

CHAPTER 1. Dynamical Model 4

1.1. Circular Restricted Three-Body Problem 4

1.2. Full Two-Body Problem 5

1.2.1. Gravity potential of a tri-axial ellipsoid 7

1.2.2. Normalization 8

1.3. Restricted Full Three-Body Problem 11

1.3.1. Jacobi integral 13

CHAPTER 2. Chaos Definition and Measurement Tools 15

2.1. Poincaré Maps 15

2.2. Lyapunov Exponents 17

2.3. Pesin Entropy Formula 19

CHAPTER 3. Numerical Methods for Parameter Determination . . 21

3.1. Least Squares Method for Data Fitting 21

3.2. Optimization 23

3.3. Scaling Dependence 25

Conclusions 29

Bibliography 31

APPENDIX 1: Variational Equations 32

APPENDIX 2: Markley’s Method **35**

APPENDIX 3: Python Code **36**

LIST OF FIGURES

1.1 Comparison between the real F2BP system and the restricted one with a point mass sphere as the smaller primary. (Credit: Julie Bellerose and Daniel J. Scheeres) 6

1.2 F2BP under sphere restriction: geometry of the problem. (Credit: Julie Bellerose and Daniel J. Scheeres) 6

1.3 Gravity potential of a tri-axial body with $a = 400$ m, $b = 300$ m and $c = 200$ m and a density $\rho = 1.7$ g/cm^3 in separated 2D planes. 9

1.5 RF3BP: geometry of the problem. (Credit: Julie Bellerose and Daniel J. Scheeres) 11

1.4 Gravity potential in separated 2D planes of a binary system being the bigger body a tri-axial ellipsoid with $a = 400$ m, $b = 300$ m and $c = 200$ m and a density $\rho = 1.7$ g/cm^3 and the smaller one a sphere with point mass. 12

1.6 Normalized zero-velocity surfaces in the $x-y$ coordinate frame for an ellipsoid-sphere system with dimensional values of distance between the bodies $q = 1180$ m; ellipsoid parameters $a = 400$ m, $b = 300$ m, and $c = 200$ m; sphere radius $r_s = 75$ m; and uniform density of the system $\rho = 1.7$ g/cm^3. The colorbar displays the Jacobi constant value for each surface in state space or curve in the plane. 14

2.1 Representation of a Poincaré section (red line in Fig. a) normal to the plane of the orbit and transversal to the flow. The resulting Poincaré map (Fig. b) represents the x coordinate and velocity ($\dot{x}$) the flow had when crossing the section. 16

2.2 Poincaré map showing the intersection of several planar trajectories propagated up to $t = 1000$ with different initial conditions around the same binary system ($a = 400$, $b = 300$, $c = 200$ and $\rho = 1.7$ g/cm^3). Trajectories intersect a Poincaré section in the $y = 0$ plane for all negative x. The closed curves depicted in the center of the figure are related to quasi-periodic behaviours of the orbits. The ones closer to the edge start diverging and behaving more chaotically. 17

2.3 Representation of the evolution of finite time Lyapunov exponents for three different trajectories. Each trajectory tends to a different asymptotic value, λ_{100}. The samples have been taken in steps of $\Delta t = 0.01$ up to $t = 100$ where the trajectory already has a stable behaviour. Initial conditions for trajectories, I: $[-2.00, 0.00, 0.00, 0.00, 2.50, 0.00]$; II: $[-0.75, 0.00, 0.00, 0.00, 1.75, 0.00]$; III: $[-0.45, 0.00, 0.00, 0.00, -2.00, 0.00]$. 19

2.4 Representation of the Pesin entropy and finite time Lyapunov exponents evolution for a single trajectory. The samples have been taken in steps of $\Delta t = 0.01$ up to $t = 100$ where the trajectory already has a stable behaviour. It shows a strong similarity in both results, proving they are both valid to the determination of the sensitivity to initial conditions of a trajectory. Trajectory I: $[-2.00, 0.00, 0.00, 0.00, 2.50, 0.00]$ 20

3.1 For the sake of simplicity, the ellipsoid parameters (a, b, c) and the initial conditions vector [x,y,z,$\dot{x}$,$\dot{y}$,$\dot{z}$] have been fixed. The only parameter to be determined is μ. This plot represents the error, Q, resulting from the least squares data fitting routine. It depends on the mass parameter, $\mu_{estimated}$, used to get the estimated data. The closer the estimated parameter to the true one, the better the estimated trajectory fits to the measured one. Thus, the minimum of the function corresponds to the best fitting estimated trajectory as well as to the closest estimated mass parameter, $\mu_{estimated}$, with respect to the true one, μ. The measurements are taken at a constant sampling rate of $\Delta t = 0.01$. For the same deviation from the true value, the error is larger as more measurements are taken, making the parameter determination more accurate. 23

3.2 Absolute error, $\Delta\mu$, obtained from the difference between the abscissa of the $Q(\mu_{estimated})$ minimum and the true parameter, μ; against the length of the trajectory in time. Since the sampling rate is constant, $\Delta t = 0.01$, the length of the trajectory in time is directly proportional to the number of measurements taken, N. It shows an improvement on the accuracy of the parameter determination as more points are compared. 24

3.3 Taking logarithms in both axes allows to perform a least-squares linear regression and getting a scaling exponent, α, which indicates how fast the uncertainty in the parameter determination decreases as the trajectory gets longer and more measurements are taken. 25

3.4 Each point corresponds to a different trajectory propagated up to t=100. They represent the scaling exponent, α, it has (how fast the error in the determination of the parameter decreases as more measurements are taken) related to the finite time Lyapunov exponent, λ_{100} which represents how sensitive to initial conditions a trajectory is. The higher λ_{100}, the more chaotic the trajectory is. . . 27

3.5 Outlier trajectory. Initial conditions: $[0.50, 0.00, 0.00, 0.00, -2.10, 0.00]$; $\lambda_{100} = 0.0369; \alpha = -2.8828$ 28

3.6 Outlier trajectory. Initial conditions: $[1.20, 0.00, 0.00, 0.00, -2.10, 0.00]$; $\lambda_{100} = 0.1411; \alpha = -2.8294$ 28

LIST OF TABLES

1.1 Confocal ellipsoid coefficients 7

3.1 Summary of the sampled trajectories. Shot with a vertical velocity $\dot{y}$ from different horizontal positions x along the x-axis. They are related with their finite time Lyapunov exponents and the resulting scalings. 26

INTRODUCTION

Placing satellites in orbit around celestial bodies is used to examine their properties. As they collect more measurements of the trajectory, N, the uncertainty in the determination of the parameters, σ, reduces with a scaling of $\sigma = N^{\alpha}$, with $\alpha < 0$. Whilst non-chaotic systems present scalings in the order of $N^{-\frac{1}{2}}$, previous research on simple systems found out that chaotic behaviours can reach scalings between N^{-1} and N^{-2}.

Previous research on chaos in astrodynamics was mostly focused on the determination of the orbit for later times [1], but this behaviour could be very useful to characterise the properties of celestial bodies. Prof. Andrea Milani has shown the scaling one could achieve is N^{-1}, but other work has shown in some systems the scaling can reach N^{-2} [2]. This method requires chaos, so only special trajectories will exhibit this behaviour. Luckily these are not too hard to find as the restricted three body problem is well known to be chaotic, and it is found all over the Solar System with satellites passing moons and binary asteroids. The scaling behaviour is strongly dependent on the chaos, so this method works best in systems with a short chaotic time period. These will be systems with a shorter characteristic time such as orbits of moons and asteroids.

State-of-the-art parameter determination

There are two main motivations for the study of asteroids and comets. The first one involves the understanding of the origin and evolution of the Solar System [3]. The comet's composition reflects the composition of the pre-solar nebula out of which the Sun and the planets of the Solar System formed, more than 4.6 billion years ago. Therefore, an in-depth analysis of comet 67P/Churyumov-Gerasimenko, as the Rosetta mission and its lander did, would provide essential information to understand how the Solar System formed.

There is convincing evidence that comets played a key role in the evolution of the planets, because cometary impacts are known to have been much more common in the early Solar System than today. Comets, for example, probably brought much of the water in today's oceans. They could even have provided the complex organic molecules that may have played a crucial role in the evolution of life on Earth.

When Rosetta arrived at comet 67P/Churyumov-Gerasimenko in early August 2014, not much was known about the comet [4]. The orbit of the comet had been determined from years of tracking from ground observatories and a few months of optical tracking by Rosetta during approach. The lander Philae was scheduled to land in about three months at a date chosen as a compromise between the time required to acquire sufficient knowledge about the comet and the risk of a rising comet activity worsening the navigation accuracy. During these three months, the comet had to be characterized.

A rough estimate for the comet mass was obtained in early August but it was not until October, one month to landing, when Rosetta was orbiting at 20 kilometers distance and below that reliable estimates for the mass distribution in the form of gravitational spherical harmonics coefficients of degree and order 2 then 3 were derived. The last mass distribution update, including a 7 meters shift in the center of mass position along the Z axis, was performed in early November less than 2 weeks to landing.

By means of a non-chaotic trajectory, the characterization of the comet performed mainly in the three months from arrival to lander delivery has allowed Rosetta to navigate safely and accurately around the comet.

The second interest for the study of asteroids and comets is related to planetary defense [5]. It encompasses all the capabilities needed to detect the possibility and warn of potential asteroid or comet impacts with Earth, and then either prevent them or mitigate their possible effects. It involves characterizing those objects to determine their orbit trajectory, size, shape, mass, composition, rotational dynamics and other parameters, so that experts can determine the severity of the potential impact event, warn of its timing and potential effects, and determine the means to mitigate the impact.

The coming Asteroid Impact and Deflection Assessment (AIDA) mission concept is an international collaboration among the European Space Agency (ESA), NASA, Observatoire de la Côte d'Azur (OCA), and the Johns Hopkins University Applied Physics Laboratory (JHU/APL). It will be the first demonstration of the kinetic impact technique to change the motion of an asteroid in space. AIDA is a dual-mission concept, involving two independent spacecraft – NASA's Double Asteroid Redirection Test (DART), and ESA's Asteroid Impact Mission (AIM).

AIDA is a science-driven test of one of the technologies for preventing the Earth impact of a hazardous asteroid: the kinetic impactor. AIDA's primary objective is to demonstrate, and to measure the effects of, a kinetic impact on a small asteroid. Its target is the binary near-Earth asteroid (65803) Didymos, which consists of a primary body approximately 800 meters across, and a secondary body (or "moonlet") whose 150-meter size is more typical of the size of asteroids that could pose a more common hazard to Earth.

Chaotic determination technique

Our contribution consists of the development of a technique which performs the parameter determination of a dynamical system by means of chaotic trajectories. The presence of chaos is expected to reduce the time required for the characterization of the system.

With the success of several past missions such as Rosetta, Hayabusa or New Horizons as well as the coming AIDA mission measuring the properties of these small bodies has renewed relevance. It is unlikely that any large mission would intentionally place itself in a chaotic orbit, but with the advent of the use of CubeSats for scientific missions, this method of examining celestial bodies could become cheaper in future and as ChipSats are now being used in an expendable way this technique could still bare fruit.

One of the ideal applications for this technique would consist of placing the mother ship of the mission in a safe and ordered orbit around the body to be studied. From there, some expendable small satellites will be shot inserting themselves in the most suitable chaotic trajectories in order to perform a highly efficient determination of the system.

Those small determination satellites would just require to send their telemetry, containing their own position with respect to the system, to the mother ship which will send the data back to Earth to be analyzed.

The project is divided into three chapters. The first two are based on previous research and set the basis for the final study. The last one comprehends original numerical methods for the determination of the parameters of the system.

Chapter 1 defines the problem to be studied describing the dynamical model that is used. It starts from the simplest circular restricted three-body problem (CR3BP), which considers a binary system with two point masses. Then, complexity increases by considering the mass distribution of the bigger primary while leaving the smaller one as a point mass sphere. A program has been coded in Python simulating the gravity field of the binary system and propagating the trajectory of a spacecraft of negligible mass from its initial conditions. It returns verification values and allows lots of flexibility in the parameter settings.

Chapter 2 describes three different tools for the definition and classification of trajectories around a certain system according to how chaotic they are. Poincaré maps represent the intersection of the trajectory with a selected section in state space. Depending on the shape of the plot, the behaviour of the trajectory can be considered ordered or chaotic. Lyapunov exponents and the Pesin entropy formula provide a measure on how sensitive to initial conditions a trajectory is. Trajectories with the highest and least Lyapunov exponents are used to compare the effect of the chaos.

Chapter 3 consists of finding trajectories which have particularly helpful statistical properties by means of the design and performance of different numerical methods based on a least-squares estimation routine. The aim is to get a correlation between the error in the parameter determination and the length of the trajectories compared. The longer the trajectory, the more measurements are taken for a constant sampling rate of $\Delta t = 0.01$. Analysis based on the output of the simulation is done to calculate the parameters based on various initial conditions. The program is then be tested by assessing if ordered and chaotic trajectories produce Gaussian and non-Gaussian scaling respectively. The final objective of the chapter is to get a correlation between the scaling in the determination and the finite time Lyapunov exponents. Therefore, an assessment on how chaos affects the reduction in the uncertainty of the determination is performed. The ultimate aim of this project is to show that chaotic behaviour can be advantageous to determine the properties of celestial bodies.

Finally, a discussion on the results obtained and their effects and causes as well as some future work propositions and ideas is done.

CHAPTER 1. DYNAMICAL MODEL

As described by Newton's law of universal gravitation, the general three-body problem describes the motion of three different bodies under their mutual gravitational interactions. Despite the easiness of the two-body problem, which has well-known closed form solutions, it becomes so complicated when adding a third body. The attempts of many physicists, astronomers and mathematicians to find closed solutions have been unsuccessful for over three hundred years. Those solutions do not exist due to the motion of the three bodies is in general unpredictable, making the general three-body problem one of the most challenging problems in the history of science.

The three-body problem was formulated and studied by Newton in his *Principia* (1687), where he considered the motion of the Earth and the Moon around the Sun. This is just an example among the many cases of three-body problem we can encounter in the Solar System, such as Sun–planet–planet, Sun–planet–moon, or Sun–planet–asteroid systems. The latter can be simplified neglecting the mass of the asteroid against the mass of either the Sun or the planet, meaning the gravitational influence of the asteroid on the planet and the Sun can be omitted. This condition can also be satisfied in many ways, such as when an artificial satellite is orbiting a binary system. Then, the general three-body problem becomes simpler and termed as restricted three-body problem. The primaries or two bodies with dominant masses can move around their center of mass either along circular or elliptic orbits, which leads to the respective circular or elliptic restricted three-body problems.

The circular restricted three-body problem (CR3BP) is going to be the problem studied during the present project due to it is the most extensively studied and well documented. Moreover, it is highly present in the Solar System and it is well known that it produces chaos.

1.1. Circular Restricted Three-Body Problem

The CR3BP considers a body of negligible mass m moving under the gravitational influence of two point masses m_1 and m_2, called primaries, in circular motion around their common center of mass.

A synodic reference system, which rotates with the same angular velocity as the primaries, with origin at the barycenter, being the x-axis the line joining them, y-axis perpendicular to it and z-axis perpendicular to the orbital plane of the primaries. Thus, the coordinates of m_1, m_2 and m are $(x_1, 0, 0)$, $(x_2, 0, 0)$ and (x, y, z) respectively and the distances between m and the primaries m_1 and m_2 are defined as r_1 and r_2:

$$\begin{aligned} r_1^2 &= (x-x_1)^2 + y^2 + z^2 \\ r_2^2 &= (x-x_2)^2 + y^2 + z^2 \end{aligned} \tag{1.1}$$

A suitable set of units for mass and distance are chosen such that the sum of the primary masses and the distance between them is unity. Therefore, $m_1 = 1-\mu$ and $m_2 = \mu$, where the mass parameter is defined as $0 < \mu = \frac{m_2}{m_1+m_2} \leq \frac{1}{2}$. Thus, the coordinates of the primaries m_1 and m_2 in the barycentric coordinate system result as $(-\mu,0,0)$ and $(1-\mu,0,0)$ respectively.

Being the effective potential defined as [6]:

$$\Omega(x,y,z) = \frac{1}{2}(x^2+y^2) + \frac{1-\mu}{r_1} + \frac{\mu}{r_2} + \frac{\mu(1-\mu)}{2} \tag{1.2}$$

where the first term on the right-hand-side generates the centrifugal force. The second and third terms are the gravitational potentials for masses m_1 and m_2.

The equations of motion for the circular restricted three-body problem result as:

$$\ddot{x} - 2\dot{y} = \frac{\mathrm{d}\Omega}{\mathrm{d}x} \tag{1.3}$$

$$\ddot{y} + 2\dot{x} = \frac{\mathrm{d}\Omega}{\mathrm{d}y} \tag{1.4}$$

$$\ddot{z} = \frac{\mathrm{d}\Omega}{\mathrm{d}z} \tag{1.5}$$

The CR3BP considers both primaries as point masses. For the sake of accuracy, a slight variation will be introduced in further sections, increasing complexity. The mass distribution of the bigger body is going to be considered by modeling it as a tri-axial ellipsoid. The smaller primary will still be considered as a point mass.

1.2. Full Two-Body Problem

The Two-Body Problem considers the dynamics of two spherical bodies in orbit about each other. We refer to the Full Two-Body Problem (F2BP) [7] when we consider the mass distribution of at least one of the two bodies. The general situation is shown in Fig. 1.1a. Without any approximations, this system involves 12 degrees of freedom. The sphere restriction shown in Fig. 1.1b reduces the problem to lower dimension. In total, six degrees of freedom can be removed from the conservation of angular momentum and the rotational dynamics of the sphere while keeping the interesting dynamical features.

In the current work, we study the dynamics of an ellipsoid–sphere system with uniform density, and refer to it as the F2BP. In Fig. 1.2, we define M_1 and M_2 as the masses of the ellipsoidal and spherical shapes, respectively, with a mass fraction defined as:

$$\mu = \frac{M_2}{M_1+M_2} \tag{1.6}$$

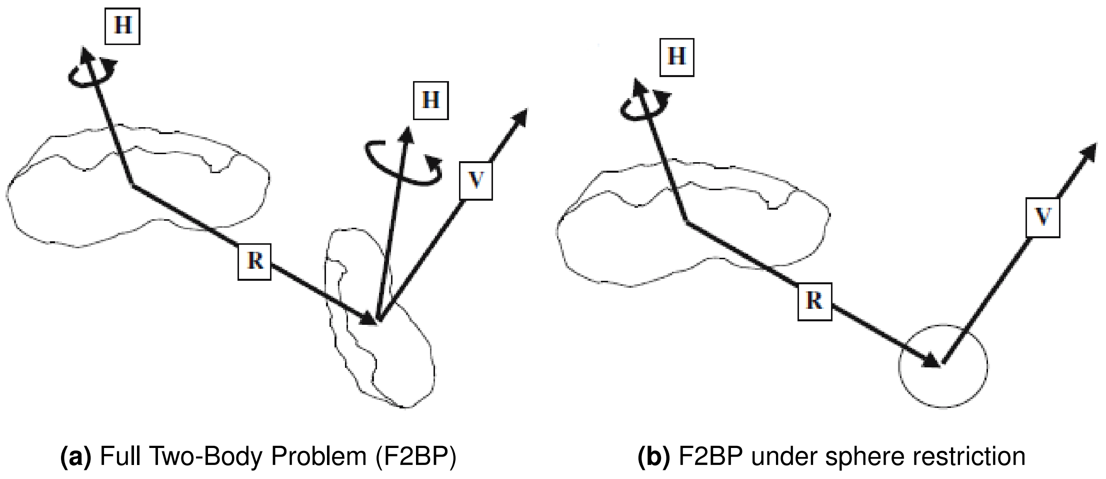

H
H
R
V
H
R
V
(a) Full Two-Body Problem (F2BP)
(b) F2BP under sphere restriction

Figure 1.1: Comparison between the real F2BP system and the restricted one with a point mass sphere as the smaller primary. (Credit: Julie Bellerose and Daniel J. Scheeres)

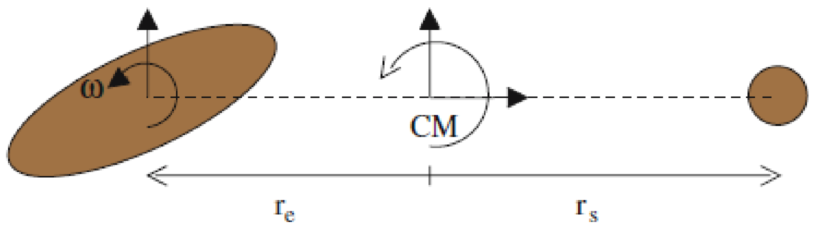


Figure 1.2: F2BP under sphere restriction: geometry of the problem. (Credit: Julie Bellerose and Daniel J. Scheeres)

In dimensional form, the position of the two bodies relative to their center of mass (located at origin) are:

$$\mathbf{r}_e = -\mu \mathbf{r}_b \tag{1.7}$$

and

$$\mathbf{r}_s = (1-\mu)\mathbf{r}_b \tag{1.8}$$

where subscripts e and s refer to the ellipsoid and the spherical body, respectively, and $\mathbf{r}_b$ is the position vector of the sphere relative to the ellipsoid.

For a general body in a sphere restricted binary system, the dynamics of the binary system in the general body-fixed frame is defined by

$$\ddot{\mathbf{r}}_b + 2\Omega \times \dot{\mathbf{r}}_b + \dot{\Omega} \times \mathbf{r}_b + \Omega \times (\Omega \times r_b) = G(M_1 + M_2)\frac{\partial \tilde{U}}{\partial \mathbf{r}_b} \tag{1.9}$$

and the rotational dynamics of the general body are described in the general body-fixed frame by

$$\mathbf{I} \cdot \dot{\Omega} + \Omega \times \mathbf{I} \cdot \Omega = -GM_1 \mathbf{r}_b \times \frac{\partial \tilde{U}}{\partial \mathbf{r}_b} \tag{1.10}$$

where Ω is the angular velocity of the general body, $\mathbf{I}$ is its inertia matrix normalized by its mass and $\tilde{U}$ is the mutual potential.

1.2.1. Gravity potential of a tri-axial ellipsoid

The aim of this section is to represent the dimensional gravitational potential coming from a tri-axial body of uniform density [8].

Since the density ρ is uniform, the gravitational parameter μ for the body can be expressed as:

$$\mu = G\rho\frac{4}{3}\pi abc \tag{1.11}$$

where $G = 6.6695 \times 10^{-11}[m^3/kg/s^2]$ is the universal gravitational constant. The gravitational potential V of a solid homogeneous tri-axial ellipsoid in R_s can be expressed as:

$$V(x,y,x) = \frac{3}{4}\mu\int_{\kappa_0}^{\infty}\left(1-\frac{x^2}{a^2+\kappa}-\frac{y^2}{b^2+\kappa}-\frac{z^2}{c^2+\kappa}\right)\cdot\frac{d\kappa}{\sqrt{(a^2+\kappa)(b^2+\kappa)(c^2+\kappa)}} \tag{1.12}$$

where κ_0 is the largest root of the confocal ellipsoid C defined as

$$C(\kappa) = \frac{x^2}{a^2+\kappa}+\frac{y^2}{b^2+\kappa}+\frac{z^2}{c^2+\kappa}-1 \tag{1.13}$$

In order to get the roots we have to solve a cubic polynomial

$$c_3k^3+c_2k^2+c_1k+c_0 = 0 \tag{1.14}$$

whose coefficients are derived from Eq. 1.13:

Order	Coefficient
c_3	-1
c_2	$x^2+y^2+z^2-(a^2+b^2+c^2)$
c_1	$(b^2+c^2)x^2+(a^2+c^2)y^2+(a^2+b^2)z^2-(a^2c^2+b^2c^2+a^2b^2)$
c_0	$(b^2c^2)x^2+(a^2c^2)y^2+(a^2b^2)z^2-(a^2b^2c^2)$

Table 1.1: Confocal ellipsoid coefficients

By relating the gravitational potential to the Carlson elliptic integrals, the expression for the gravitational potential of a tri-axial ellipsoid becomes:

$$\begin{aligned} V(x,y,z) = &\frac{3}{2}\mu R_F(a^2+\kappa,b^2+\kappa,c^2+\kappa)\\ &-\frac{1}{2}\mu x^2R_D(b^2+\kappa,c^2+\kappa,a^2+\kappa)\\ &-\frac{1}{2}\mu y^2R_D(a^2+\kappa,c^2+\kappa,b^2+\kappa)\\ &-\frac{1}{2}\mu z^2R_D(a^2+\kappa,b^2+\kappa,c^2+\kappa)\end{aligned} \tag{1.15}$$

Where R_D is one of the Carlson symmetric forms, the explicit cartesian components of the gravitational acceleration $\mathbf{a}_g$ in (x,y,z) can be derived by taking the gradient of V which yields:

$$\begin{aligned} a_x &= \frac{\partial V}{\partial x} = -\mu x R_D(b^2+\kappa_0, c^2+\kappa_0, a^2+\kappa_0) \\ a_y &= \frac{\partial V}{\partial y} = -\mu y R_D(a^2+\kappa_0, c^2+\kappa_0, b^2+\kappa_0) \\ a_z &= \frac{\partial V}{\partial z} = -\mu z R_D(a^2+\kappa_0, b^2+\kappa_0, c^2+\kappa_0) \end{aligned} \tag{1.16}$$

Implementing the previously described equations in Python results in the gravity field graphs for separated planes represented in Fig. 1.3.

1.2.2. Normalization

To simplify the analysis, the following normalizations are introduced. The distance between the primaries, denoted by q, and the mean motion of the system at this radius:

$$n = \sqrt{\frac{G(M_1+M_2)}{q^3}} \tag{1.17}$$

are taken as length and time scales, respectively. The normalized position and angular velocity are then

$$\mathbf{r} = \frac{\mathbf{r}_b}{q} \quad \text{and} \quad \omega = \frac{\Omega}{n}$$

Thus, Eqs. 1.9 and 1.10 now become

$$\ddot{\mathbf{r}} + 2\omega\times\dot{\mathbf{r}} + \dot{\omega}\times\mathbf{r} + \omega\times(\omega\times\mathbf{r}) = \frac{\partial V}{\partial \mathbf{r}} \tag{1.18}$$

and

$$\mathbf{I}\cdot\dot{\omega} + \omega\times\mathbf{I}\cdot\omega = -\mu\mathbf{r}\times\frac{\partial V}{\partial \mathbf{r}} \tag{1.19}$$

As previously commented, the general body is modeled as an ellipsoid. The normalized expression for an ellipsoid potential energy is written in therms of elliptic integrals. It can be expressed as

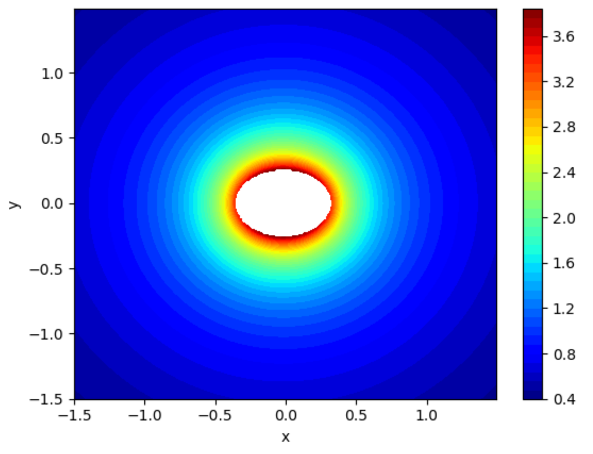
$$U_e = \frac{3}{4}\int_{\kappa_0}^{\infty} \phi(\mathbf{r},\kappa)\frac{d\kappa}{\Delta(\kappa)} \tag{1.20}$$

$$\phi(\mathbf{r},\kappa) = 1 - \frac{x^2}{\alpha^2+\kappa} - \frac{y^2}{\beta^2+\kappa} - \frac{z^2}{\gamma^2+\kappa} \tag{1.21}$$

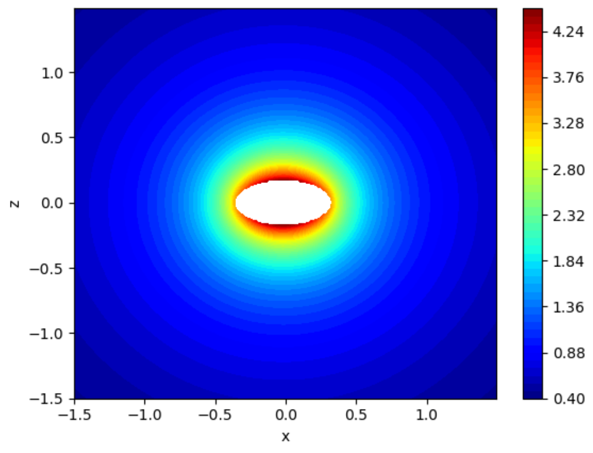
$$\Delta(\kappa) = \sqrt{(\alpha^2+\kappa)(\beta^2+\kappa)(\gamma^2+\kappa)} \tag{1.22}$$

where $0 < \gamma \leq \beta \leq 1$ and κ_0 is the largest root (satisfies $\phi(\mathbf{r},\kappa_0) = 0$) of the confocal ellipsoid C defined as

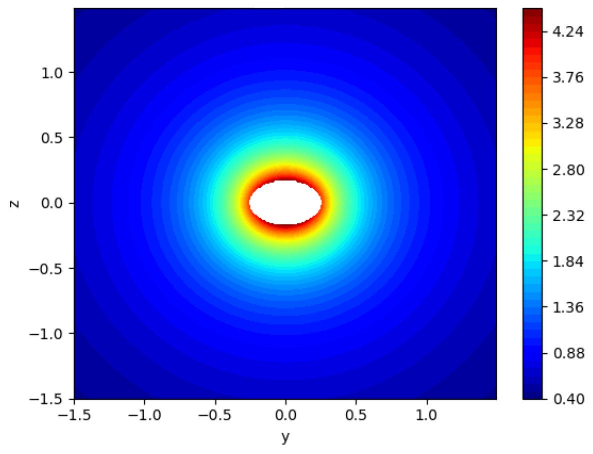
$$C(\kappa) = \frac{(x+\mu r)^2}{\alpha^2+\kappa} + \frac{y^2}{b^2+\kappa} + \frac{z^2}{c^2+\kappa} - 1 \tag{1.23}$$



(a) XY projection in RTBP coordinates



(b) XZ projection in RTBP coordinates



(c) YZ projection in RTBP coordinates

Figure 1.3: Gravity potential of a tri-axial body with $a = 400$ m, $b = 300$ m and $c = 200$ m and a density $\rho = 1.7$ g/cm^3 in separated 2D planes.

Introducing the following variable transformation $\kappa' = \kappa - \kappa_0$ in Eq.1.20 the expression for the normalized gravitational potential of a tri-axial ellipsoid becomes:

$$\begin{aligned} U_e(x,y,z) = &\frac{3}{2}R_F(\alpha^2+\kappa_0, \beta^2+\kappa_0, \gamma^2+\kappa_0) \\ &-\frac{1}{2}x^2 R_D(\beta^2+\kappa_0, \gamma^2+\kappa_0, \alpha^2+\kappa_0) \\ &-\frac{1}{2}y^2 R_D(\alpha^2+\kappa_0, \gamma^2+\kappa_0, \beta^2+\kappa_0) \\ &-\frac{1}{2}z^2 R_D(\alpha^2+\kappa_0, \beta^2+\kappa_0, \gamma^2+\kappa_0) \end{aligned} \tag{1.24}$$

where $R_F(x,y,z)$ is the Carlson symmetric elliptic integral of the first and $R_D(x,y,z)$ is a special case of the third kind.

The explicit cartesian components of the gravitational acceleration $\mathbf{a}_g$ in (x,y,z) can then be derived by taking the gradient of U_e which yields,

$$\begin{aligned} a_x &= \frac{\partial U_e}{\partial x} = xR_D(\beta^2+\kappa_0, \gamma^2+\kappa_0, \alpha^2+\kappa_0) \\ a_y &= \frac{\partial U_e}{\partial y} = yR_D(\alpha^2+\kappa_0, \gamma^2+\kappa_0, \beta^2+\kappa_0) \\ a_z &= \frac{\partial U_e}{\partial z} = zR_D(\alpha^2+\kappa_0, \beta^2+\kappa_0, \gamma^2+\kappa_0) \end{aligned} \tag{1.25}$$

A particular solution of the F2BP is for the two bodies to be in relative equilibrium, whose conditions are found by setting all velocities and accelerations to zero in Eqs. 1.18, 1.19. With these equations it is possible to show that there exists equilibria when one of the principal axes of the ellipsoid is pointed towards the sphere, as acceleration and position vectors are parallel.

These equilibria are independent of the principal axis the ellipsoid is rotating about; solutions exist along the x, y and z axis. Given a solution along a q axis, the square of the spin rate is expressed as

$$\omega^2 = \frac{3}{2}\int_\lambda^\infty \frac{d\kappa}{(\alpha_q^2+\kappa)\Delta(\kappa)} \tag{1.26}$$

α_q represents the radius along which the sphere is located. Here, $\lambda = q^2 - \alpha_q^2$, in which q is the distance between the primaries (unity).

Introducing the following variable transformation $\kappa' = \kappa - \lambda$ in Eq.1.26 the expression for the square of the spin rate becomes:

$$\omega^2 = R_J(a^2+\lambda, b^2+\lambda, c^2+\lambda, \alpha_q^2+\lambda) \tag{1.27}$$

where $R_J(x,y,z,p)$ is the Carlson symmetric elliptic integral of the third kind.

The relative equilibria for an ellipsoid-sphere system and their stability have been mapped in [7] as a function of the mass ratio and distance between the bodies. This work studied two configurations in which the α or β ellipsoid parameter is aligned with the sphere, referred to as the long-axis and short-axis configurations, respectively. We choose to consider the long-axis configuration, as it was shown to be the only energetically stable configuration for certain parameters. Thus, α_q = α.

1.3. Restricted Full Three-Body Problem

Having a long-axis configuration in the F2BP, we look at the dynamics of a particle in this gravitational field represented in Fig. 1.4. [9]

We now consider a spacecraft of negligible mass in the gravitational field of the binary system. For the current problem we assume relative equilibrium of the F2BP as defined by Eqs. 1.18 and 1.19. As shown in Fig. 1.5 we now let ρ be the position of a particle relative to the center of mass of the system.

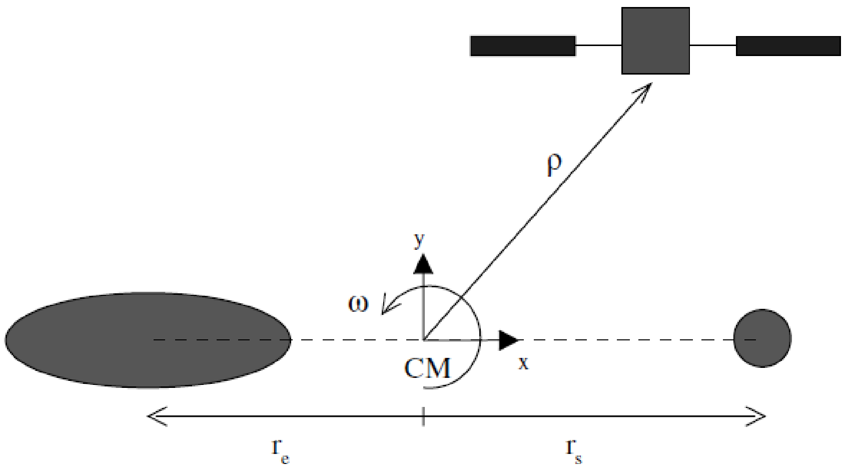


Figure 1.5: RF3BP: geometry of the problem. (Credit: Julie Bellerose and Daniel J. Scheeres)

The dynamics of the problem are expressed as

$$\ddot{\rho} + 2\Omega \times \dot{\rho} + \Omega \times (\Omega \times \rho) = G(M_1 + M_2)\frac{\partial \tilde{U}_{12}}{\partial \rho} \tag{1.28}$$

In normalized units Eq.1.28 becomes

$$\ddot{\tilde{\rho}} + 2\omega \times \dot{\tilde{\rho}} + \omega \times (\omega \times \dot{\rho}) = \frac{\partial U_{12}}{\partial \dot{\tilde{\rho}}} \tag{1.29}$$

where

$$\tilde{\rho} = \frac{\rho}{q}$$

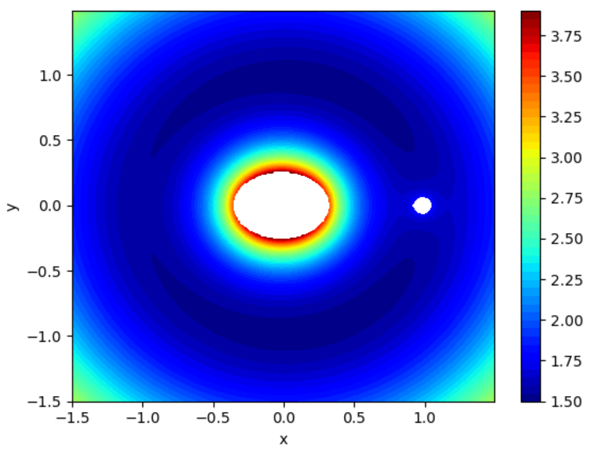
is the normalized distance of the particle. Because we take a frame fixed to the ellipsoid, U_{12} is a time-invariant potential energy expression, as the bodies are in mutual equilibrium. It is expressed as

$$U_{12}(x,y,z) = (1-\mu)U_e(\tilde{\rho} + \mu\mathbf{r}) + \frac{\mu}{|\tilde{\rho} - (1-\mu)\mathbf{r}|} \tag{1.30}$$

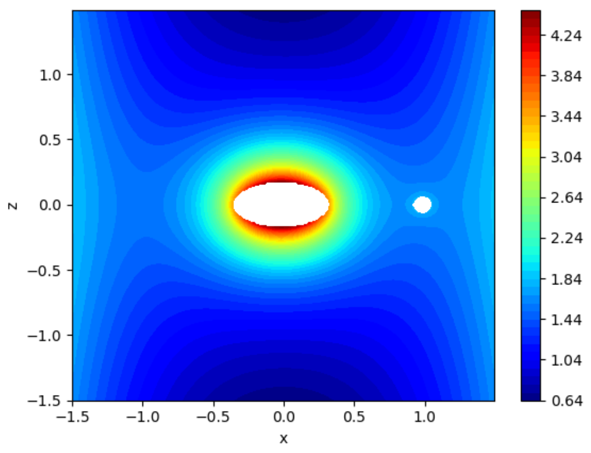
U_e represents the normalized expression for the ellipsoid body defined in Eq. 1.24

Adding the centrifugal force term to the mutual potential expression in Eq. 1.30 we get the effective potential equation

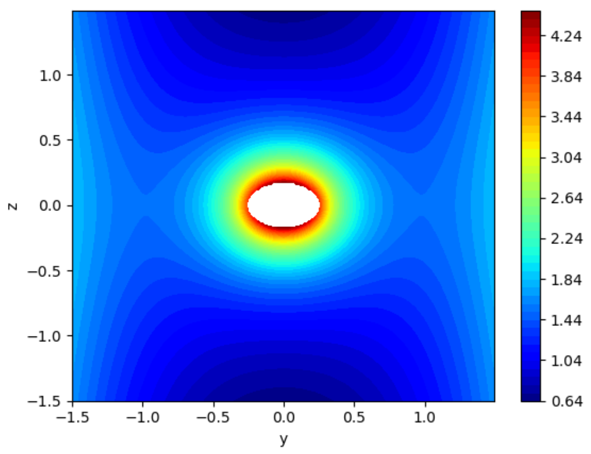
$$V(x,y,z) = \frac{1}{2}\omega^2(x^2 + y^2) + U_{12} \tag{1.31}$$



(a) XY projection in RTBP coordinates



(b) XZ projection in RTBP coordinates



(c) YZ projection in RTBP coordinates

Figure 1.4: Gravity potential in separated 2D planes of a binary system being the bigger body a tri-axial ellipsoid with $a = 400$ m, $b = 300$ m and $c = 200$ m and a density $\rho = 1.7$ g/cm^3 and the smaller one a sphere with point mass.

The free parameters of this system are the mass ratio, μ, and the size parameters of the ellipsoid, α, β and γ. Given these parameters, the spin rate, ω, is given by Eq.1.27.

In an (x, y, z) coordinate system, the equations of motion are the result of writing the x, y, and z components of the dynamics equation Eq. (1.29)

$$\ddot{x} - 2\omega\dot{y} = \frac{\partial V}{\partial x} \tag{1.32}$$

$$\ddot{y} + 2\omega\dot{x} = \frac{\partial V}{\partial y} \tag{1.33}$$

$$\ddot{z} = \frac{\partial V}{\partial z} \tag{1.34}$$

1.3.1. Jacobi integral

Being the two bodies in relative equilibrium, the system allows for one integral of motion, the Jacobi integral. For a spacecraft navigating in this system, the Jacobi integral indicates the regions in which it can move and provides necessary conditions for when it may escape the system.

Furthermore, the Jacobi integral is a good indicator to assess whether the code and the computations are working properly since its value must remain constant along the whole trajectory when equilibrium conditions are fulfilled.

To derive the Jacobi integral we multiply the equations of motion by $\dot{x}$, $\dot{y}$, and $\dot{z}$ respectively and add them up to find

$$\ddot{x}\dot{x} + \ddot{y}\dot{y} + \ddot{z}\dot{z} - \omega^2(\dot{x}x + \dot{y}y) = \frac{\partial V}{\partial x}\dot{x} + \frac{\partial V}{\partial y}\dot{y} + \frac{\partial V}{\partial z}\dot{z} \tag{1.35}$$

Substituting

$$\ddot{x} = \frac{d\dot{x}}{dt} = \frac{d\dot{x}}{dx}\frac{dx}{dt} \tag{1.36}$$

and integrating with respect to time it yields

$$C_J = 2V(x,y,z) - (\dot{x}^2 + \dot{y}^2 + \dot{z}^2) \tag{1.37}$$

Its value dictates regions in the state space where the particle may and may not move. Contours in the state space representing $v^2 = 0$, the so-called zero velocity surfaces, determine boundaries which the infinitessimal mass may not cross because v^2 would be positive on one side of the boundary and negative on the other, therefore v would be an imaginary number.

Zero-velocity surfaces will allow us to place bounds in the infinitesimal mass motion. For a given Jacobi integral, one can calculate the curve in space where the velocity would go to zero. Such areas are equivalent to turning points for potential wells in inertial frames of reference.

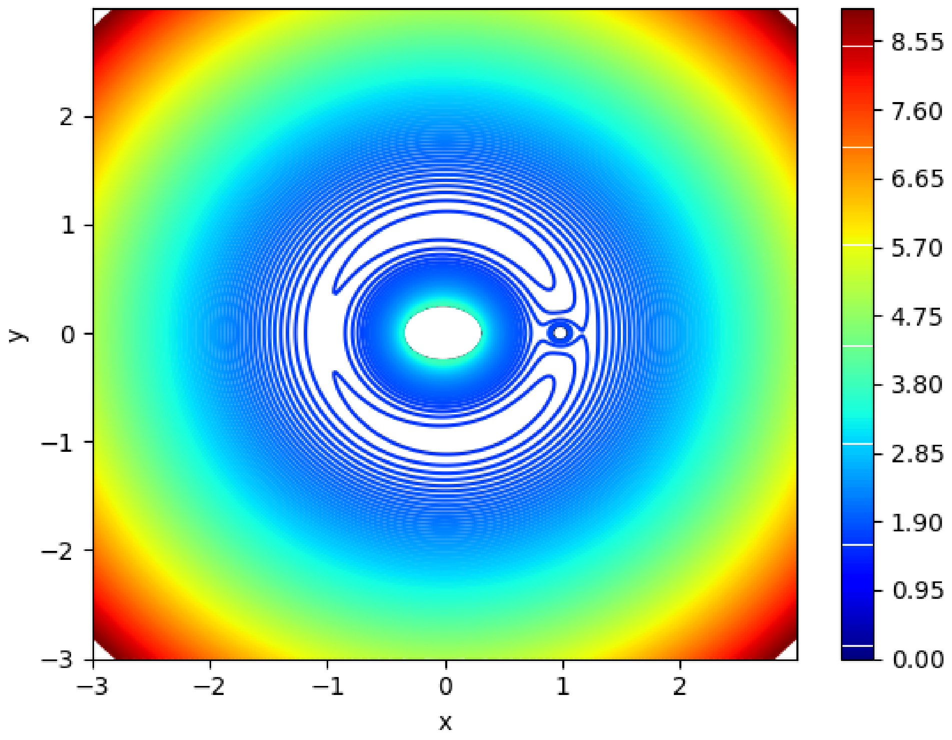


Figure 1.6: Normalized zero-velocity surfaces in the $x-y$ coordinate frame for an ellipsoid-sphere system with dimensional values of distance between the bodies $q = 1180$ m; ellipsoid parameters $a = 400$ m, $b = 300$ m, and $c = 200$ m; sphere radius $r_s = 75$ m; and uniform density of the system $\rho = 1.7$ g/cm^3. The colorbar displays the Jacobi constant value for each surface in state space or curve in the plane.

CHAPTER 2. CHAOS DEFINITION AND MEASUREMENT TOOLS

Chaos is commonly defined as the state of total confusion with no order. Nevertheless, chaos theory is a branch of mathematics focused on the study of dynamical systems with high sensitivity dependence with respect to the initial conditions. It comprehends underlying patterns within the apparent randomness of chaotic systems.

Related to chaos, the butterfly effect describes how a small change in one state of a deterministic nonlinear system can result in large differences in a later state. For instance, a butterfly flapping its wings in Brazil can cause a tornado in Texas. Defining a deterministic system as the one in which no randomness is involved in the development of future states of the system. A deterministic model will thus always produce the same output from a given starting condition or initial state.

If the initial state was known exactly, the future state of such a system could theoretically be predicted. However, in practice, knowledge about the future state is limited by the precision at which the initial state can be measured, making the system unpredictable from a certain point in time on. This measurement noise, by itself, will create some sensitivity to initial conditions that could trigger positive measurements of chaos, such as positive Lyapunov exponents, even for systems that are not chaotic at all.

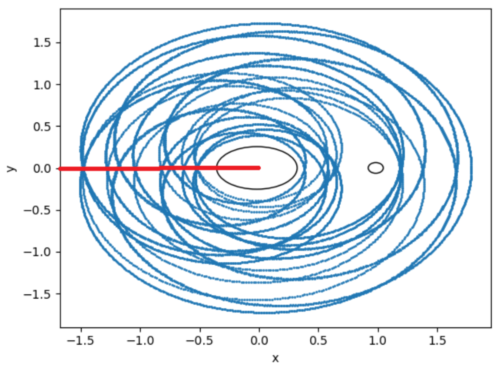
The definition and measurement of chaos is far from trivial and lots of discrepancies are found related to them. In general, chaotic systems are characterized by a sensitive dependence on the initial conditions. They also have to be bounded such as to exclude the trivial case of an unstable linear system where trajectories diverge exponentially for all times [10]. The first statement is satisfied if a Lyapunov exponent is larger than zero. The latter can be verified using Poincaré maps guaranteeing that trajectories fold back close to where they where before.

As this project concerns, as long as the trajectory is bounded, the higher the Lyapunov exponent, the more chaotic the system will be considered. The more sensitive to initial conditions, the more information small changes will provide about the system and the less measurements will be needed to determine its parameters for a certain accuracy.

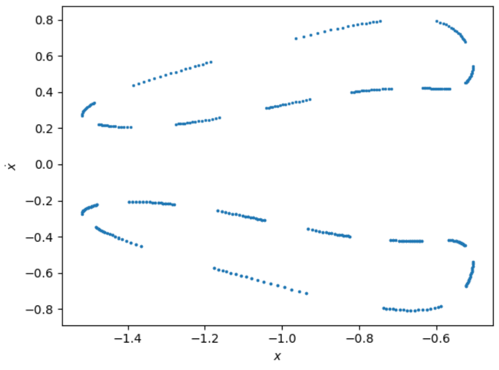
2.1. Poincaré Maps

Henri Poincaré was a French mathematician who, in his research of the three-body problem, discovered a chaotic deterministic system which laid the foundations of modern chaos theory. A first recurrence or Poincaré map, as named after him, is used in dynamical systems to reduce the study of continuous time systems or flows to the study of an associated discrete time system or map.

Instead of analyzing the whole trajectory we look at the intersection of the orbit in the state space of the continuous system with a certain lower-dimensional subspace, called the Poincaré section. The section is transversal to the flow of the system as shown by the red line in Fig. 2.1a representing a plane normal to the plane of the orbit.



(a) Planar trajectory with Poincaré section



(b) Poincaré map

Figure 2.1: Representation of a Poincaré section (red line in Fig. a) normal to the plane of the orbit and transversal to the flow. The resulting Poincaré map (Fig. b) represents the x coordinate and velocity ($\dot{x}$) the flow had when crossing the section.

Certain properties of the underlying dynamics of the system translate to the Poincaré map. For instance, if the trajectory is perfectly periodic, the map will show a single point since the particle will intersect the section at the point every period. Chaotic dynamics will show an erratic distribution of points in the map and quasi-periodic behaviours will be represented as a closed curve.

Several trajectories with different initial conditions around the same binary system are represented in Fig. 2.2. The ones in the center describe a quasi-periodic behaviour since the points intersecting the Poincaré section represent closed curves in the map. They become more chaotic as they get closer to the edge of the map, diverging from the elliptic curves and plotting more erratic behaviours.

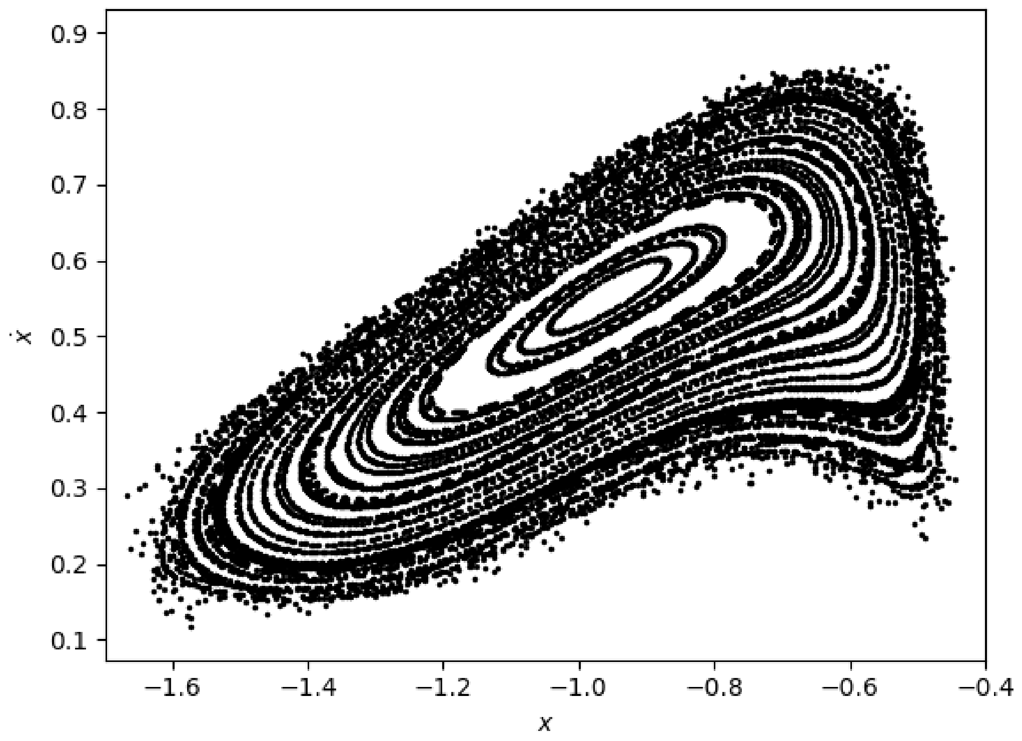


Figure 2.2: Poincaré map showing the intersection of several planar trajectories propagated up to $t = 1000$ with different initial conditions around the same binary system ($a = 400$, $b = 300$, $c = 200$ and $\rho = 1.7\ g/cm^3$). Trajectories intersect a Poincaré section in the $y = 0$ plane for all negative x. The closed curves depicted in the center of the figure are related to quasi-periodic behaviours of the orbits. The ones closer to the edge start diverging and behaving more chaotically.

It is important to bear in mind that, despite a purely erratic behaviour of the trajectories is not seen in the plot, they might be very useful for our objective. What matters is the sensitivity to the initial conditions determined by the Lyapunov exponents described as follows.

2.2. Lyapunov Exponents

It is a characteristic feature of chaotic systems that initially nearby trajectories separate exponentially in time. The mean rate of separation is called the Lyapunov exponent of the system.

Lyapunov exponents have proven to be the most useful dynamical diagnostic for chaotic systems quantifying the degree of sensitivity to initial conditions. They represent the average exponential rates of divergence or convergence of nearby orbits in phase space. Since nearby orbits correspond to nearly identical states, exponential orbital divergence means that systems whose initial differences we may not be able to resolve will soon behave quite differently-predictive ability is rapidly lost. Any system containing at least one positive Lyapunov exponent is defined to have sensitivity dependence on the choice of initial conditions, with the magnitude of the exponent reflecting the time scale on which system dynamics become unpredictable.

Having a flow $\dot{y} = f(x)$, a trajectory x(t) and the variational equations,

$$\dot{Dx} = Df(x(t))Dx, \tag{2.1}$$

where $Df(x(t))$ is the differential of the vectorfield evaluated on the trajectory and Dx the state transition matrix further developed in Appendix 1 3.3..

The effective Lyapunov exponents for a given initial state and time result from the singular-value decomposition (SVD) of the state transition matrix (Dx) of the system. The singular values of a $M \times N$ matrix X are the square roots of the eigenvalues of the $N \times N$ resulting from $X^T X$.

$$\lambda^i_{eff}(\mathbf{x_0}, T) = \frac{1}{T} \log\left(SVD^i \left[Dx(T) \right] \right) \tag{2.2}$$

The λ_{eff} quantify directly the mean loss of information about the localization of the trajectory in state space during the forecasting time [11].

The maximum Lyapunov exponent of the system corresponds to the maximum singular value of the state transition matrix which equals to take the euclidean norm of Dx

$$\lambda_\infty = \lim_{t\to\infty} \frac{1}{T} \log \left\| Dx(T) \right\|_2, \tag{2.3}$$

for almost all trajectories and initial displacements. For segments of a trajectory, one can obtain a finite time Lyapunov exponent from

$$\lambda_T(x(t)) = \frac{1}{T} \log \left\| Dx(T) \right\|_2 \tag{2.4}$$

It is impossible in practise to propagate a trajectory up to infinity. Therefore, we consider finite time Lyapunov exponents at $t = 100$ for the measurement of the sensitivity to initial conditions of a trajectory. The finite time Lyapunov exponent is computed at each time step of $\Delta t = 0.01$ up to $t = 100$ to study their evolution with time as represented in Fig. 2.3. It is clearly seen how the Lyapunov exponents of different trajectories tend to different asymptotic values as time tends to a hundred units.

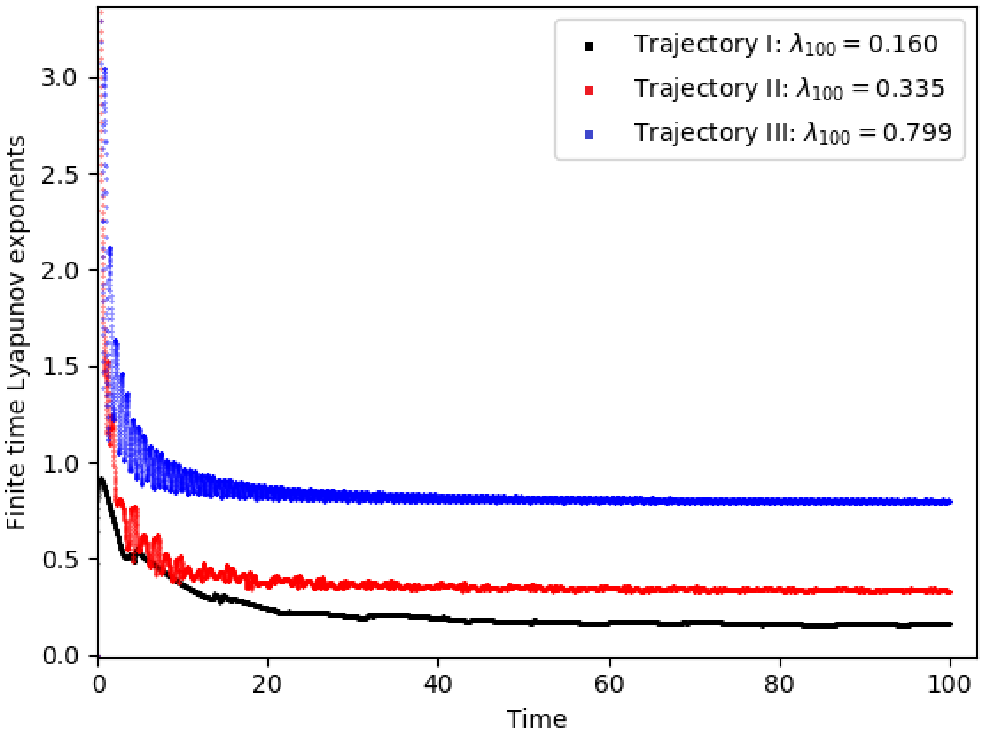


Figure 2.3: Representation of the evolution of finite time Lyapunov exponents for three different trajectories. Each trajectory tends to a different asymptotic value, λ_{100}. The samples have been taken in steps of $\Delta t = 0.01$ up to $t = 100$ where the trajectory already has a stable behaviour. Initial conditions for trajectories, I: $[-2.00, 0.00, 0.00, 0.00, 2.50, 0.00]$; II: $[-0.75, 0.00, 0.00, 0.00, 1.75, 0.00]$; III: $[-0.45, 0.00, 0.00, 0.00, -2.00, 0.00]$.

2.3. Pesin Entropy Formula

Another approach, the Pesin entropy formula, is a formula according to which the entropy of a measure that is invariant under a dynamical system is given by the total asymptotic expansion rate present in the dynamical system [12].

It suggests that entropy is created by the exponential divergence of nearby orbits. In a conservative system, all the expansion goes back to the system to make entropy, leading to the equality in Pesin's entropy formula in Eq. 2.5. It claims that the amount of topological entropy in a conservative dynamical system is proportional to the sum of the positive Lyapunov exponents [13]. Therefore, the trajectories with highest local Lyapunov exponents should be the most entropic, and will provide more information by measuring them.

In non-conservative systems, some of the expansion would be "wasted", meaning there is a "leakage" or dissipation from the system bringing forth the Margulis-Ruelle inequality, out of the scope of this work.

The local entropy can be calculated along a trajectory, or just at a point [11].

$$h(\mathbf{x_0}, T) = \sum \lambda^{i+}_{eff}(\mathbf{x_0}, T) \tag{2.5}$$

As stated before, since it is impossible in practise to compute the effective Lyapunov exponents at infinity, the Pesin entropy formula is computed up to $t = 100$. By that time the system should have reached a pretty steady state. Fig. 2.4 shows the evolution of the Pesin entropy formula, computed at each time step of $\Delta t = 0.01$ up to $t = 100$, to study its evolution together with the finite time Lyapunov exponents.

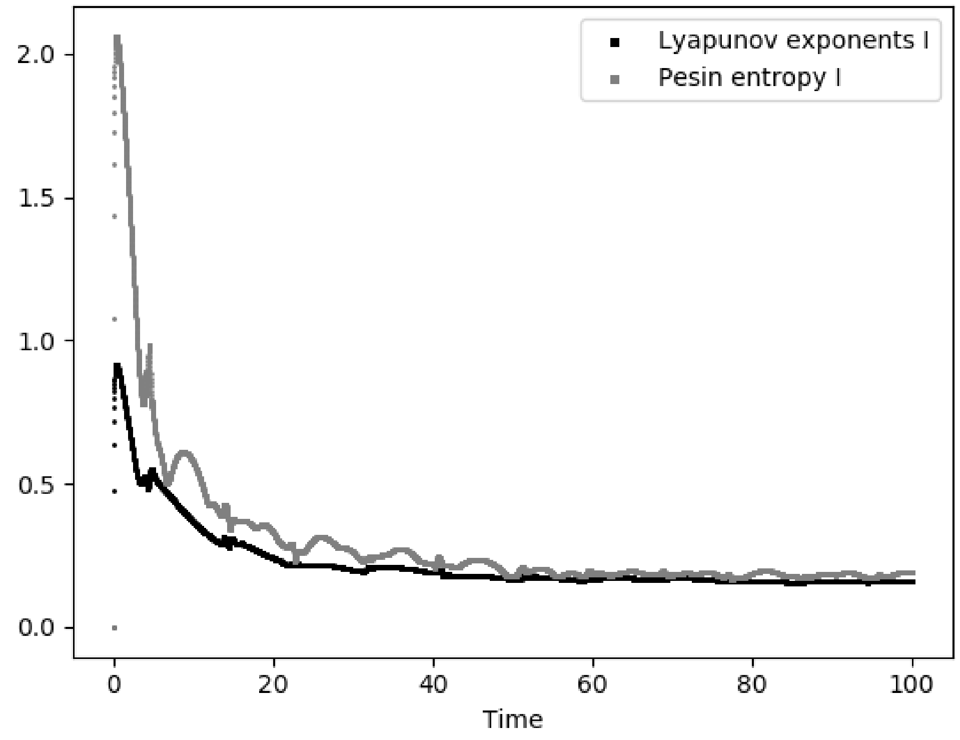


Figure 2.4: Representation of the Pesin entropy and finite time Lyapunov exponents evolution for a single trajectory. The samples have been taken in steps of $\Delta t = 0.01$ up to $t = 100$ where the trajectory already has a stable behaviour. It shows a strong similarity in both results, proving they are both valid to the determination of the sensitivity to initial conditions of a trajectory. Trajectory I: $[-2.00, 0.00, 0.00, 0.00, 2.50, 0.00]$

As seen in the comparison, since the Pesin entropy formula is based on the effective Lyapunov exponents, the results are very similar. They tend to a very close value, making them both valid to classify trajectories according to their sensitivity to initial conditions. Nevertheless, due to Lyapunov exponents are the most extended measure, we are going to use them further on to give trajectories a single measure of chaos.

CHAPTER 3. NUMERICAL METHODS FOR PARAMETER DETERMINATION

Assuming the dynamical model described in Chapter 1, a binary system where the larger primary is modeled as a tri-axial ellipsoid with constant density, and the smaller one as point mass sphere, the purpose is to determine the four parameters that define the binary system: the mass parameter μ, and the length of the ellipsoid axes: a, b, and c. The determination will be performed by means of the so called measured data a satellite would send back to Earth while orbiting around the system. The data would contain its position with respect to the barycenter of the system.

Some noise in the measured data is required. Otherwise the system would be analytically determinable. The two types of noise are measurement noise and system noise. System noise on a nonlinear system can lead to correlated errors in the results. This means that the behaviour of the noise would have to be considered, which will drastically increase the complexity of the problem. Considering this, only measurement noise is added at each sample of the measured data.

Given a binary system (μ, a, b, c), a trajectory is uniquely identified according to its initial conditions vector, $[x, y, z, \dot{x}, \dot{y}, \dot{z}]$. It indicates the initial position (x, y, z) and velocity $(\dot{x}, \dot{y}, \dot{z})$ of the spacecraft with respect to the center of mass of the binary system.

From a fixed vector of initial conditions, the measured data is generated by propagating a trajectory using the true parameters of the system. Pseudo-random generated noise is added at each step of the integration to simulate measurement noise. The word pseudo-random means that despite the noise generated by the computer might seem random, the process is still deterministic. We set a seed in order to generate the exact same noise for different trajectories. Thus, we avoid noise related errors when comparing different trajectories. The considered Gaussian noise scale is $\sigma = 10^{-4}$.

For the sake of simplicity and computing time, the determination of a single parameter, μ, is assessed. The rest of them (a, b, c) remain fixed for the preliminary study. We assume the initial conditions of the measured trajectory are known exactly.

A set of estimated trajectories is simulated using the constant parameters of the system (a, b, c) and the same initial conditions $[x, y, z, \dot{x}, \dot{y}, \dot{z}]$ as the measured trajectory. Each estimated trajectory is related to a single $\mu_{estimated}$ variable within a certain range, subsequently defined.

Finally, each estimated trajectory is compared against the measured one. The best fitting estimated trajectory corresponds to the one generated from the closest $\mu_{estimated}$ parameter with respect to the true one, μ.

3.1. Least Squares Method for Data Fitting

Batch state estimation is where all the data is assessed simultaneously. It can work in a very similar way to a Kalman filter. To analyze the data in this system we use least squares estimation. This involves rerunning the simulation based on a set of estimated parameters, and comparing the generated (estimated) data to the original (measured) one.

Being a residual the difference between an observed value and the fitted value provided by a model, the least squares function returns the sum of squared residuals normalized over the number N of samples from a data series and a scaling factor σ.

Simulating a trajectory with some given parameters provide us with a set of estimated (fitted) data, position and velocity. Comparing the estimated positions with the measured or observed ones, as Eqs. 3.1, 3.2 and 3.3 describe, eventually returns the quality of fit as Eq. 3.4 shows. Meaning the more similar the two sets of data are, the better the quality of the fit and the lower the error, Q, will be.

$$Q_x = \frac{1}{N}\sum_{i=0}^{N}\frac{(x_{i,measured} - x_{i,estimated})^2}{\sigma^2} \tag{3.1}$$

$$Q_y = \frac{1}{N}\sum_{i=0}^{N}\frac{(y_{i,measured} - y_{i,estimated})^2}{\sigma^2} \tag{3.2}$$

$$Q_z = \frac{1}{N}\sum_{i=0}^{N}\frac{(z_{i,measured} - z_{i,estimated})^2}{\sigma^2} \tag{3.3}$$

$$Q = \frac{Q_x + Q_y + Q_z}{3} \tag{3.4}$$

A measured trajectory is generated using the true parameters of the system ($a = 400$ m, $b = 300$ m, $c = 200$ m and $\mu = 0.017274$) and the following initial conditions: $x = -0.75$, $y = 0$, $z = 0$, $\dot{x} = 0$, $\dot{y} = 1.75$, $\dot{z} = 0$. These initial conditions provide a bounded and quasi-periodic orbit close to the bigger primary. Many other trajectories are tested in further sections with different properties.

The initial conditions and the length of the axes of the ellipsoid are assumed as known. The only parameter to be determined is μ. Thus, a set of 32 estimated trajectories are simulated from a range of equidistant parameters between $0.010 \leq \mu_{estimated} \leq 0.026$.

Fig. 3.1 shows the result of comparing those estimated trajectories against the measured one. Each value of Q corresponds to the quality of the fit resulting from the comparison. The closer the estimated parameter to the true one, the better the estimated trajectory fits to the measured one. Therefore, the minimum of the function corresponds to the best fitting estimated trajectory. Its abscissa corresponds to the closest $\mu_{estimated}$ value with respect to the true one, μ.

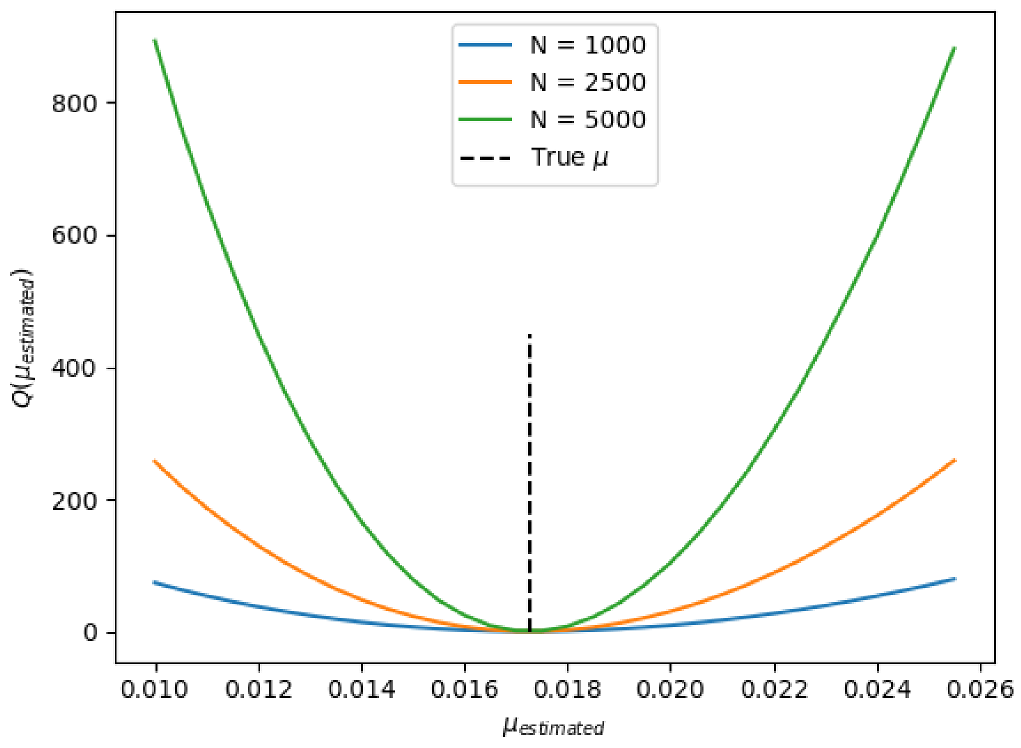


Figure 3.1: For the sake of simplicity, the ellipsoid parameters $(a,\ b,\ c)$ and the initial conditions vector $[x,y,z,\dot{x},\dot{y},\dot{z}]$ have been fixed. The only parameter to be determined is μ. This plot represents the error, Q, resulting from the least squares data fitting routine. It depends on the mass parameter, $\mu_{estimated}$, used to get the estimated data. The closer the estimated parameter to the true one, the better the estimated trajectory fits to the measured one. Thus, the minimum of the function corresponds to the best fitting estimated trajectory as well as to the closest estimated mass parameter, $\mu_{estimated}$, with respect to the true one, μ. The measurements are taken at a constant sampling rate of $\Delta t = 0.01$. For the same deviation from the true value, the error is larger as more measurements are taken, making the parameter determination more accurate.

As expected, the abscissa of the minimum of the $Q(\mu_{estimated})$ function corresponds to the true value of the parameter of the system, μ, previously used to generate the measured data. Furthermore, the longer the compared trajectories are, the more samples taken and points compared, the narrower the plot gets. For the same deviation from the true value, the error is larger as more points are compared, making the parameter determination more accurate.

3.2. Optimization

The estimated parameters are now optimized to reduce the difference between the estimated and measured data. This is a very concrete way to produce the best estimation of the variables but computationally very slow since a whole trajectory propagation has to be performed at every optimization step.

A Nelder-Mead optimization routine is run to find the minimum of the $Q(\mu_{estimated})$ objective function. It is run at different lengths for each trajectory. Despite we lack its gradient, our objective function has a pretty smooth shape and it is easy to minimize. The abscissa of this minimum value is compared to the true one. Therefore we have a measure of the error in the determination as a function of the length of the trajectories compared.

For a given number of measurements, each optimization step returns the difference between the abscissa of the minimum of the objective function and the true parameter, μ. It results in the absolute error, $\Delta\mu$, of the determination of the mass parameter. The length of the trajectories increases in 1 measurement per optimization step. It is expected that $\Delta\mu$ decreases as the length of the trajectory increases. We keep optimizing and increasing the length up to a certain accuracy. A tolerance of 10^{-2} with respect to the true parameter is set as convergence criteria for the method.

At this point, a trajectory which performs a slow determination is preferable in order to get enough points to assess the correlation between the error in the determination and the length of the trajectory in time. Therefore, given the previously defined binary system ($a = 400$ m, $b = 300$ m, $c = 200$ m and $\mu = 0.017274$), the following initial conditions are chosen to get an orbit far from the primaries, causing the subsequent slow determination: $x = -10$, $y = 0$, $z = 0$, $\dot{x} = 0$, $\dot{y} = 10.525$, $\dot{z} = 0$.

One should notice that, owing to the sampling rate is constant, $\Delta t = 0.01$, the length of the trajectory in time is directly proportional to the number of measurements taken, N.

Fig. 3.2 shows how the absolute error, $\Delta\mu$, in the determination of the parameter depends on the propagation time of the compared trajectories for a given initial state and system.

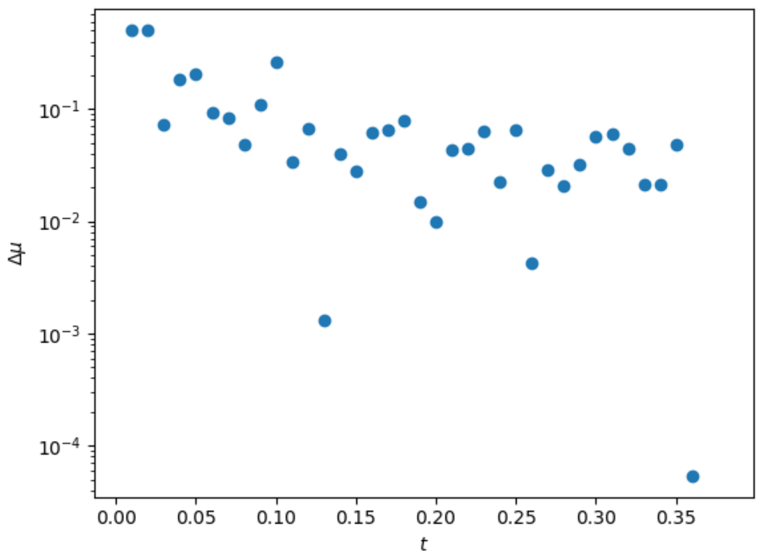


Figure 3.2: Absolute error, $\Delta\mu$, obtained from the difference between the abscissa of the $Q(\mu_{estimated})$ minimum and the true parameter, μ; against the length of the trajectory in time. Since the sampling rate is constant, $\Delta t = 0.01$, the length of the trajectory in time is directly proportional to the number of measurements taken, N. It shows an improvement on the accuracy of the parameter determination as more points are compared.

By taking logarithms as Fig. 3.3 shows, with a least-squares linear regression one can get the scaling exponent, α, of the absolute error in the determination, $\Delta\mu$, with respect to the propagation time. In other words, how fast the error decreases when the longer the trajectory is propagated and so more measurements are considered, $\Delta\mu = t^{\alpha}$.

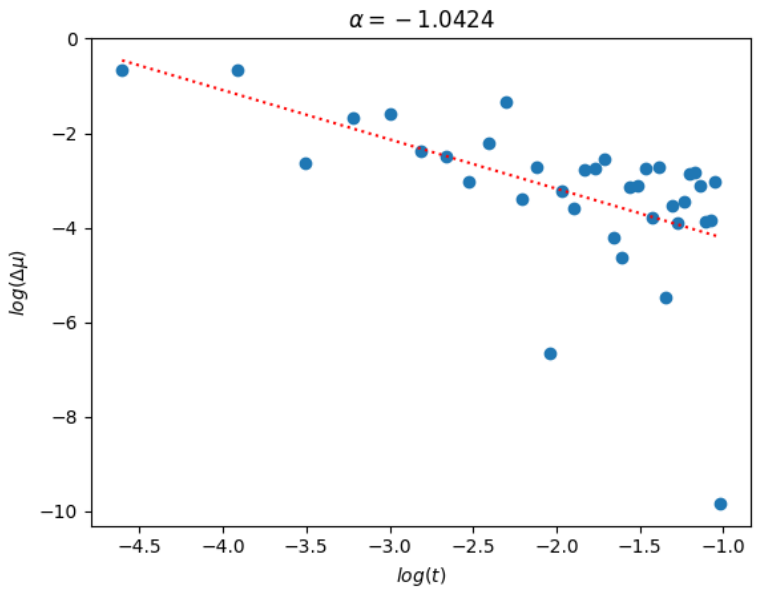


Figure 3.3: Taking logarithms in both axes allows to perform a least-squares linear regression and getting a scaling exponent, α, which indicates how fast the uncertainty in the parameter determination decreases as the trajectory gets longer and more measurements are taken.

3.3. Scaling Dependence

The sensitivity to initial conditions of a trajectory can be determined by the computation of its Lyapunov exponents. The convergence rate in the determination of the parameter is defined by the scaling exponent, α. Making use of both tools, we can study the dependence of the scaling in the determination of the parameters in a complex dynamical system as described in Chapter 1.

According to previous research [1] [2] in simple dynamical systems, the higher the Lyapunov exponent of a trajectory, the faster the determination will be for a fixed accuracy. The scaling exponents for chaotic trajectories were bounded between $\alpha = -1$ and $\alpha = -2$.

One should bear in mind that, in the present work, the complexity of the dynamical system studied might introduce many sources of perturbation in the determination of the parameters apart from chaos. Therefore, the data resulting will have to be very carefully studied.

For the same dynamical system, several trajectories with different initial conditions are sampled. Their scaling and finite time Lyapunov exponents are computed as plotted in Fig. 3.4 with the aim of observing some correlation as literature suggests. The trajectories are sampled by shooting with different vertical velocities ($\dot{y}$) from different horizontal positions (x) along the x-axis. They are all identified and summarized in Table 3.1.

Initial conditions, [x, y, z, $\dot{x}$, $\dot{y}$, $\dot{z}$]	**Finite time Lyapunov exponent, λ_{100}**	**Scaling, α**
$[-0.50, 0.00, 0.00, 0.00, 2.00, 0.00]$	0.614369551	-3.1077
$[-0.75, 0.00, 0.00, 0.00, 1.75, 0.00]$	0.328471276	-1.3029
$[-2.00, 0.00, 0.00, 0.00, 2.50, 0.00]$	0.160539669	-1.7407
$[-1.00, 0.00, 0.00, 0.00, 2.20, 0.00]$	0.122819670	-1.4357
$[-1.00, 0.00, 0.00, 0.00, 2.32, 0.00]$	0.167948320	-1.4357
$[1.50, 0.00, 0.00, 0.00, -1.00, 0.00]$	0.149160510	-1.1020
$[0.50, 0.00, 0.00, 0.00, -2.00, 0.00]$	0.331510662	-2.8828
$[0.45, 0.00, 0.00, 0.00, -2.00, 0.00]$	0.799480112	-2.6202
$[0.45, 0.00, 0.00, 0.00, -2.50, 0.00]$	0.224440783	-2.6202
$[0.75, 0.00, 0.00, 0.00, -1.75, 0.00]$	0.335904778	-2.9495
$[0.70, 0.00, 0.00, 0.00, -1.75, 0.00]$	0.270254064	-2.5841
$[0.73, 0.00, 0.00, 0.00, -1.75, 0.00]$	0.324828086	-3.0543
$[1.50, 0.00, 0.00, 0.00, -2.20, 0.00]$	0.087525327	-1.4238
$[0.61, 0.00, 0.00, 0.00, -1.80, 0.00]$	0.448072606	-2.6441
$[0.60, 0.00, 0.00, 0.00, -1.80, 0.00]$	0.539200981	-2.6202
$[0.615, 0.00, 0.00, 0.00, -1.80, 0.00]$	0.376661544	-2.6441
$[0.55, 0.00, 0.00, 0.00, -1.95, 0.00]$	0.299866951	-3.0948

Table 3.1: Summary of the sampled trajectories. Shot with a vertical velocity $\dot{y}$ from different horizontal positions x along the x-axis. They are related with their finite time Lyapunov exponents and the resulting scalings.

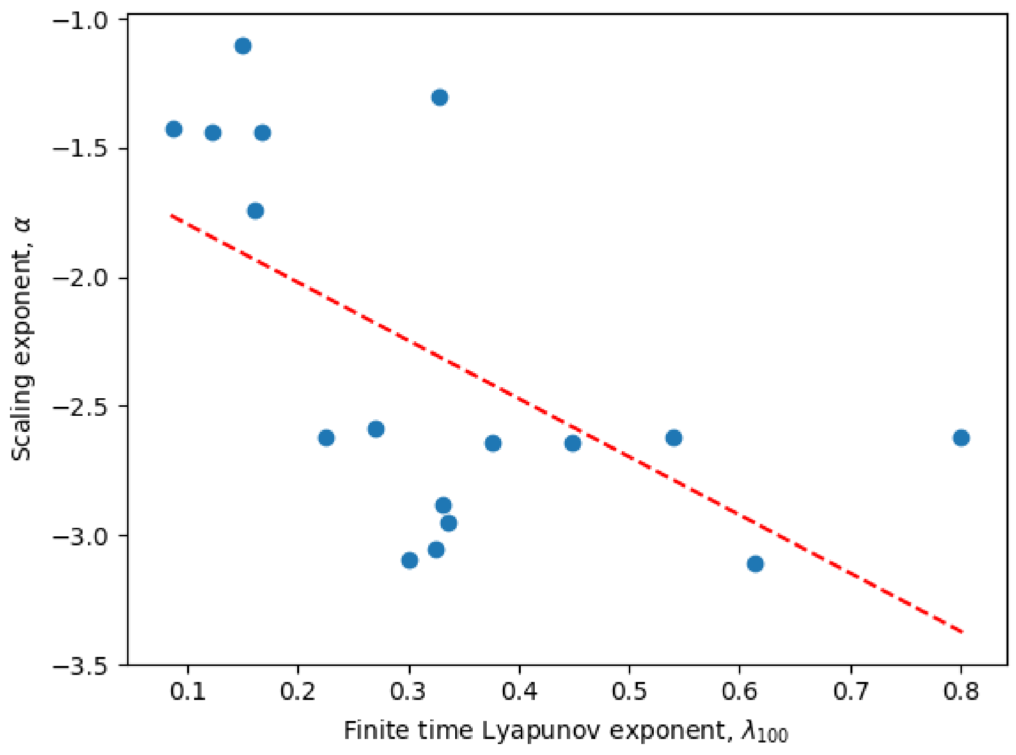
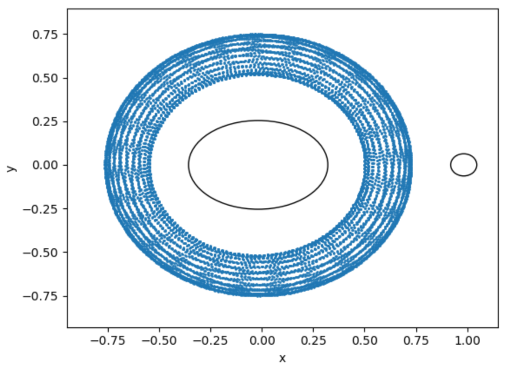


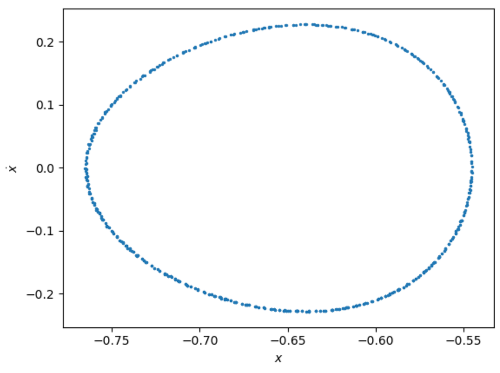
Figure 3.4: Each point corresponds to a different trajectory propagated up to t=100. They represent the scaling exponent, α, it has (how fast the error in the determination of the parameter decreases as more measurements are taken) related to the finite time Lyapunov exponent, λ_{100} which represents how sensitive to initial conditions a trajectory is. The higher λ_{100}, the more chaotic the trajectory is.

The least-squares linear regression performed returns a correlation coefficient of -0.590 and a p-value of 0.012, meaning there is a moderate correlation between the scaling exponent, α, and the sensitivity to initial conditions measured by the finite time Lyapunov exponents, λ_{100}.

Some trajectories as the ones analyzed in Fig. 3.5 and Fig. 3.6, considered as outliers, present high scaling exponents despite having low Lyapunov exponents.

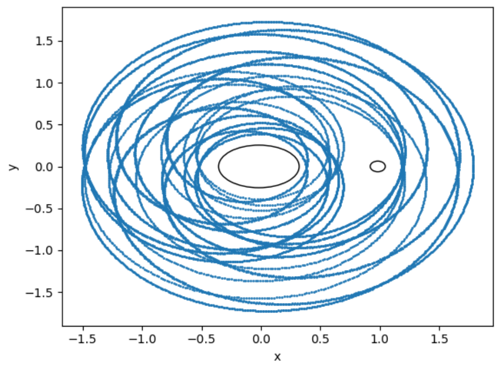


(a) Planar trajectory

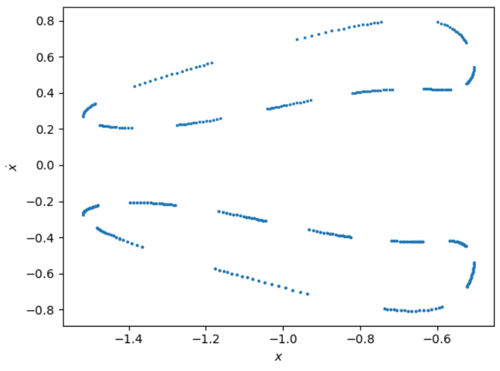


(b) Poincaré map

Figure 3.5: Outlier trajectory. Initial conditions: $[0.50, 0.00, 0.00, 0.00, -2.10, 0.00]$; $\lambda_{100} = 0.0369; \alpha = -2.8828$



(a) Planar trajectory



(b) Poincaré map

Figure 3.6: Outlier trajectory. Initial conditions: $[1.20, 0.00, 0.00, 0.00, -2.10, 0.00]$; $\lambda_{100} = 0.1411; \alpha = -2.8294$

Fig. 3.5 shows a very stable quasi-periodic behaviour, corresponding to a very low Lyapunov exponent and a closed curve in the Poincaré map, though it has a quite large α due to it orbits very close to the main body getting lots of information in a short time. On the other hand, Fig. 3.6 presents a more unstable and chaotic behaviour with a higher λ_{100} but still quasi-periodic with a very long period as the first recurrence map shows. Nevertheless, in a short time the particle has passed many times around and very close to both bodies, being able again to get lots of information in a few measurements.

CONCLUSIONS

The goal of this thesis is to show that chaotic orbits of a dynamical system are advantageous (compared with regular ones) to determine the characteristic parameters of the system, and to give an estimate about the behaviour of the error as a function of the number of observations.

Placing satellites in orbit around celestial bodies is used to examine their properties. As they collect more measurements of the trajectory, N, the uncertainty in the determination of the parameters, σ, reduces with a scaling of $\sigma = N^{\alpha}$, with $\alpha < 0$. Whilst non-chaotic systems present scalings in the order of $N^{-\frac{1}{2}}$, previous research on simple systems found out that chaotic behaviours can reach scalings between N^{-1} and N^{-2}.

The utility of chaotic trajectories for parameter determination in a complex dynamical system such as the restricted full three-body problem has been assessed. The model takes into consideration the motion of a particle of negligible mass around a binary system where the main body is modeled as a tri-axial ellipsoid, taking into account its mass distribution, and the smaller one as point mass sphere. It has been studied by means of a program coded in Python, capable of simulating the binary system gravity field and propagating trajectories in three dimensions around it. It returns verification values such as the Jacobi constant, the state transition matrix and crashing event assessment allowing lots of flexibility in the parameter settings.

Chaos can be defined in many different ways along a trajectory, which might lead to confusion and errors in the analysis of the data. Thus, a common and reliable mathematical measure of chaos is paramount. A combination of Poincaré maps, which allow to study the periodicity of a trajectory in a visual way; and mathematical tools such as the finite time Lyapunov exponents and the Pesin entropy formula, which indicate how sensitive a trajectory is to initial conditions, has been used. They return similar and reliable values providing redundancy and robustness to the procedure.

A strict procedure has been followed during the design and testing of the numerical methods used, mainly in order to avoid correlated errors. Some noise had to be inserted into the system, otherwise it would be analytically determinable. Therefore, a least-squares estimation routine for data fitting has been used in order to perform the parameter determination comparing real and estimated trajectories. Since the generation of noise from a computer is pseudo-random, meaning that although it might seem random, the process to get them is still deterministic, we have been able to compare results coming from determinations with the exact values of noise, avoiding correlated errors and proving its reliability.

Numerical methods have been developed in order to determine the parameters of a chaotic system up to a given accuracy while returning the scaling value computed from the last-squares linear regression of the log-log $\Delta\mu$ vs. time plot, which relates the logarithm in the determination of the parameter of the system with the logarithm of the length of the compared trajectories (the longer the trajectory, the more measurements have been taken since the sampling rate is a constant of $\Delta t = 0.01$). As expected, a strong correlation between the error in the determination and the length of the trajectory has been proven, being reduced as more measurements were taken.

Finally, a moderate correlation between the finite time Lyapunov exponents, λ_{100}, which indicate the chaoticity or sensitivity to initial conditions of a trajectory, and the scaling exponent, α, that measures the rate at which the uncertainty in the determination of the parameters of the system decreases as more measurements are taken, has been shown. Even though further research is required, those results allow to conjecture that trajectories with a higher level of chaos will tend to perform better in the determination of the parameters of the system showing higher scaling values than less chaotic and non-chaotic ones.

Nevertheless, some outliers have been found, such as ordered trajectories scaling in chaotic expected levels. This indicates that, apart from chaos, the scaling seems to depend on other factors such as the distance to the main body or the ground track projected over the system, opening new topics and insights to expand the study on astrophysical chaotic systems.

The scaling values obtained for chaotic trajectories with positive finite time Lyapunov exponents lie between N^{-1} and N^{-3}, being the results greater than N^{-2} not predicted by previous research. These unexpected results might be due to those previous studies were focused on very simple systems such as the standard map. This work is based on a more complex dynamical system, the restricted full three-body problem, which might introduce other factors affecting the scaling apart from just chaos.

Future work and research that will be worth to consider involve the previously commented outliers as well as the high scaling values. Besides, it will be interesting to study the behaviour and scaling presenting when trying to perform a full system determination (a, b, c, and μ), which in this project has been limited to a single parameter since we were seeking a correlation between the scaling and the chaotic behaviour.

A more in depth theoretical study on the scaling dependence on chaos would bring some light when analyzing practical situations more efficiently. Furthermore, the use of techniques such as the Space Manifold Dynamics (SMD) [14] will allow to study the chaotic regions with greater accuracy.

Finally, once further research on the topic has been performed and succeeded, the design of future applications on mission analysis will be able to take advantage of the chaocity of some trajectories, together with artificial intelligence algorithms, making the spacecraft search for and insert itself in the most suitable trajectory for the most efficient determination. Many trade-offs will arise from here, such as different levels of chaos, closer and further trajectories with respect to the primary bodies and the security factor among many others.

BIBLIOGRAPHY

[1] F. Spoto and A. Milani. Shadowing lemma and chaotic orbit determination. *Celestial Mechanics and Dynamical Astronomy*, 124(3):295–309, 2016.

[2] E.S. Hung. Parameter estimation in chaotic systems. 1995.

[3] European Space Agency. Rosetta's frequently asked questions. Online:`http://www.esa.int/Our_Activities/Space_Science/Rosetta/Frequently_asked_questions`. Accessed: February 2018.

[4] P. Muñoz T. Morley B. Godard, F. Budnik and V. Janarthanan. Orbit determination of rosetta around comet 67p/churyumov-gerasimenko. In *Proceedings 25th International Symposium on Space Flight Dynamics–25th ISSFD, Munich, Germany*, 2015.

[5] NASA. Asteroid impact and deflection assessment (aida) mission. Online:`https://www.nasa.gov/planetarydefense/aida`. Accessed: February 2018.

[6] G. Gómez, W.S. Koon, M.W. Lo, J.E. Marsden, J.J. Masdemont, and S.D. Ross. Connecting orbits and invariant manifolds in the spatial restricted three-body problem. *Nonlinearity*, 17(5):1571, 2004.

[7] J. Bellerose and D.J. Scheeres. Energy and stability in the full two body problem. *Celestial Mechanics and Dynamical Astronomy*, 100(1):63–91, 2008.

[8] S. Willis, D. Izzo, and D. Hennes. Reinforcement learning for spacecraft maneuvering near small bodies. *American Institute of Aeronautics and Astronautics Paper 16-*.

[9] J. Bellerose and D.J. Scheeres. Restricted full three-body problem: application to binary system 1999 kw4. *Journal of guidance, control, and dynamics*, 31(1):162, 2008.

[10] S. Wiggins. *Introduction to Applied Nonlinear Dynamical Systems and Chaos*. Texts in Applied Mathematics Series. Springer-Verlag GmbH, 1990.

[11] R. Doerner, B. Hübinger, W. Martienssen, S. Grossmann, and S. Thomae. Predictability portraits for chaotic motions. *Chaos, Solitons & Fractals*, 1(6):553–571, 1991.

[12] B. Hasselblatt and Y. Pesin. Pesin entropy formula. *Scholarpedia*, 3(3):3733, 2008. revision #91644.

[13] L.S. Young. Entropy in dynamical systems. *Entropy*, pages 313–327, 2003.

[14] G. Gómez and E. Barrabés. Space manifold dynamics. *Scholarpedia*, 6(2):10597, 2011.

[15] D.J. Scheeres. Stability of relative equilibria in the full two-body problem. *Annals of the New York Academy of Sciences*, 1017(1):81–94, 2004.

[16] A.P.M. Chiaradia, H.K. Kuga, and A.F.B. de Almeida Prado. Comparison between two methods to calculate the transition matrix of orbit motion. *Mathematical Problems in Engineering*, 2012, 2011.

APPENDIX 1: VARIATIONAL EQUATIONS

Being the state transition matrix, Dx, a matrix whose product with the state vector at an initial time returns the state at a latter time, whose derivative is represented by the variational equations, $\dot{D}x$, expressing the change in the final state of a propagation when a change or perturbation is applied in the initial conditions of the flow.

$$\dot{D}x = Df(x) \cdot Dx \tag{3.5}$$

Being the first term on the right hand side, $Df(x)$, the differential of the vectorfield evaluated on the trajectory and Dx the state transition matrix,

$$\dot{D}x = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{\partial^2 V}{\partial x^2} & \frac{\partial^2 V}{\partial x \partial y} & \frac{\partial^2 V}{\partial x \partial z} & 0 & 2 & 0 \\ \frac{\partial^2 V}{\partial y \partial x} & \frac{\partial^2 V}{\partial y^2} & \frac{\partial^2 V}{\partial y \partial z} & -2 & 0 & 0 \\ \frac{\partial^2 V}{\partial z \partial x} & \frac{\partial^2 V}{\partial z \partial y} & \frac{\partial^2 V}{\partial z^2} & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial x}{\partial x_0} & \frac{\partial x}{\partial y_0} & \frac{\partial x}{\partial z_0} & \frac{\partial x}{\partial \dot{x}_0} & \frac{\partial x}{\partial \dot{y}_0} & \frac{\partial x}{\partial \dot{z}_0} \\ \frac{\partial y}{\partial x_0} & \frac{\partial y}{\partial y_0} & \frac{\partial y}{\partial z_0} & \frac{\partial y}{\partial \dot{x}_0} & \frac{\partial y}{\partial \dot{y}_0} & \frac{\partial y}{\partial \dot{z}_0} \\ \frac{\partial z}{\partial x_0} & \frac{\partial z}{\partial y_0} & \frac{\partial z}{\partial z_0} & \frac{\partial z}{\partial \dot{x}_0} & \frac{\partial z}{\partial \dot{y}_0} & \frac{\partial z}{\partial \dot{z}_0} \\ \frac{\partial \dot{x}}{\partial x_0} & \frac{\partial \dot{x}}{\partial y_0} & \frac{\partial \dot{x}}{\partial z_0} & \frac{\partial \dot{x}}{\partial \dot{x}_0} & \frac{\partial \dot{x}}{\partial \dot{y}_0} & \frac{\partial \dot{x}}{\partial \dot{z}_0} \\ \frac{\partial \dot{y}}{\partial x_0} & \frac{\partial \dot{y}}{\partial y_0} & \frac{\partial \dot{y}}{\partial z_0} & \frac{\partial \dot{y}}{\partial \dot{x}_0} & \frac{\partial \dot{y}}{\partial \dot{y}_0} & \frac{\partial \dot{y}}{\partial \dot{z}_0} \\ \frac{\partial \dot{z}}{\partial x_0} & \frac{\partial \dot{z}}{\partial y_0} & \frac{\partial \dot{z}}{\partial z_0} & \frac{\partial \dot{z}}{\partial \dot{x}_0} & \frac{\partial \dot{z}}{\partial \dot{y}_0} & \frac{\partial \dot{z}}{\partial \dot{z}_0} \end{bmatrix} \tag{3.6}$$

Due to the differential equations of the vectorfield are numbered from 1 to 6, we are ordering the 36 variational equations from 7 to 42

$$\begin{bmatrix} \dot{x}_{7} & \dot{x}_{13} & \dot{x}_{19} & \dot{x}_{25} & \dot{x}_{31} & \dot{x}_{37} \\ \dot{x}_{8} & \dot{x}_{14} & \dot{x}_{20} & \dot{x}_{26} & \dot{x}_{32} & \dot{x}_{38} \\ \dot{x}_{9} & \dot{x}_{15} & \dot{x}_{21} & \dot{x}_{28} & \dot{x}_{33} & \dot{x}_{39} \\ \dot{x}_{10} & \dot{x}_{16} & \dot{x}_{22} & \dot{x}_{29} & \dot{x}_{34} & \dot{x}_{40} \\ \dot{x}_{11} & \dot{x}_{17} & \dot{x}_{23} & \dot{x}_{30} & \dot{x}_{35} & \dot{x}_{41} \\ \dot{x}_{12} & \dot{x}_{18} & \dot{x}_{24} & \dot{x}_{31} & \dot{x}_{36} & \dot{x}_{42} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ \frac{\partial^2 V}{\partial x^2} & \frac{\partial^2 V}{\partial x \partial y} & \frac{\partial^2 V}{\partial x \partial z} & 0 & 2 & 0 \\ \frac{\partial^2 V}{\partial y \partial x} & \frac{\partial^2 V}{\partial y^2} & \frac{\partial^2 V}{\partial y \partial z} & -2 & 0 & 0 \\ \frac{\partial^2 V}{\partial z \partial x} & \frac{\partial^2 V}{\partial z \partial y} & \frac{\partial^2 V}{\partial z^2} & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} x_{7} & x_{13} & x_{19} & x_{25} & x_{31} & x_{37} \\ x_{8} & x_{14} & x_{20} & x_{26} & x_{32} & x_{38} \\ x_{9} & x_{15} & x_{21} & x_{28} & x_{33} & x_{39} \\ x_{10} & x_{16} & x_{22} & x_{29} & x_{34} & x_{40} \\ x_{11} & x_{17} & x_{23} & x_{30} & x_{35} & x_{41} \\ x_{12} & x_{18} & x_{24} & x_{31} & x_{36} & x_{42} \end{bmatrix} \tag{3.7}$$

where the second derivatives of the system are defined as

$$\frac{\partial^2 V}{\partial x^2} = \omega^2 + a_{xx} - \mu \frac{r_2^2 - 3(x - x_s)^2}{r_2^5} \tag{3.8}$$

$$\frac{\partial^2 V}{\partial y^2} = \omega^2 + a_{yy} - \mu \frac{r_2^2 - 3(y - y_s)^2}{r_2^5} \tag{3.9}$$

$$\frac{\partial^2 V}{\partial z^2} = \omega^2 + a_{zz} - \mu \frac{r_2^2 - 3(z - z_s)^2}{r_2^5} \tag{3.10}$$

$$\frac{\partial^2 V}{\partial x \partial y} = \frac{\partial^2 V}{\partial y \partial x} = a_{xy} - 3\mu \frac{(x - x_s)(y - y_s)}{r_2^5} \tag{3.11}$$

$$\frac{\partial^2 V}{\partial x \partial z} = \frac{\partial^2 V}{\partial z \partial x} = a_{xz} - 3\mu \frac{(x - x_s)(z - z_s)}{r_2^5} \tag{3.12}$$

$$\frac{\partial^2 V}{\partial y \partial z} = \frac{\partial^2 V}{\partial z \partial y} = a_{yz} - 3\mu \frac{(y-y_s)(z-z_s)}{r_2^5} \tag{3.13}$$

being the second partials of the ellipsoid potential [15]

$$a_{xx} = \frac{\partial^2 U_e}{\partial x^2} = -\frac{3}{2}\int_{\kappa_0}^{\infty} \frac{d\kappa}{(\alpha^2+\kappa)\Delta(\kappa)} + \frac{3x^2}{(\alpha^2+\kappa_0)^2\Delta(\kappa_0)} \frac{1}{\left[\frac{x^2}{(\alpha^2+\kappa_0)^2} + \frac{y^2}{(\beta^2+\kappa_0)^2} + \frac{z^2}{(\gamma^2+\kappa_0)^2}\right]} \tag{3.14}$$

$$a_{yy} = \frac{\partial^2 U_e}{\partial y^2} = -\frac{3}{2}\int_{\kappa_0}^{\infty} \frac{d\kappa}{(\beta^2+\kappa)\Delta(\kappa)} + \frac{3y^2}{(\beta^2+\kappa_0)^2\Delta(\kappa_0)} \frac{1}{\left[\frac{x^2}{(\alpha^2+\kappa_0)^2} + \frac{y^2}{(\beta^2+\kappa_0)^2} + \frac{z^2}{(\gamma^2+\kappa_0)^2}\right]} \tag{3.15}$$

$$a_{zz} = \frac{\partial^2 U_e}{\partial z^2} = -\frac{3}{2}\int_{\kappa_0}^{\infty} \frac{d\kappa}{(\gamma^2+\kappa)\Delta(\kappa)} + \frac{3z^2}{(\gamma^2+\kappa_0)^2\Delta(\kappa_0)} \frac{1}{\left[\frac{x^2}{(\alpha^2+\kappa_0)^2} + \frac{y^2}{(\beta^2+\kappa_0)^2} + \frac{z^2}{(\gamma^2+\kappa_0)^2}\right]} \tag{3.16}$$

$$a_{xy} = \frac{\partial^2 U_e}{\partial x \partial y} = \frac{3xy}{(\alpha^2+\kappa_0)(\beta^2+\kappa_0)\Delta(\kappa_0)} \frac{1}{\left[\frac{x^2}{(\alpha^2+\kappa_0)^2} + \frac{y^2}{(\beta^2+\kappa_0)^2} + \frac{z^2}{(\gamma^2+\kappa_0)^2}\right]} \tag{3.17}$$

$$a_{xz} = \frac{\partial^2 U_e}{\partial x \partial z} = \frac{3xz}{(\alpha^2+\kappa_0)(\gamma^2+\kappa_0)\Delta(\kappa_0)} \frac{1}{\left[\frac{x^2}{(\alpha^2+\kappa_0)^2} + \frac{y^2}{(\beta^2+\kappa_0)^2} + \frac{z^2}{(\gamma^2+\kappa_0)^2}\right]} \tag{3.18}$$

$$a_{yz} = \frac{\partial^2 U_e}{\partial y \partial z} = \frac{3yz}{(\beta^2+\kappa_0)(\gamma^2+\kappa_0)\Delta(\kappa_0)} \frac{1}{\left[\frac{x^2}{(\alpha^2+\kappa_0)^2} + \frac{y^2}{(\beta^2+\kappa_0)^2} + \frac{z^2}{(\gamma^2+\kappa_0)^2}\right]} \tag{3.19}$$

The variational equations result as:

$$\dot{x_7} = x_{10} \tag{3.20}$$

$$\dot{x_8} = x_{11} \tag{3.21}$$

$$\dot{x_9} = x_{12} \tag{3.22}$$

$$\dot{x_{10}} = x_7\frac{\partial^2 V}{\partial x^2} + x_8\frac{\partial^2 V}{\partial x \partial y} + x_9\frac{\partial^2 V}{\partial x \partial z} + 2x_{11} \tag{3.23}$$

$$\dot{x_{11}} = x_7\frac{\partial^2 V}{\partial y \partial x} + x_8\frac{\partial^2 V}{\partial y^2} + x_9\frac{\partial^2 V}{\partial y \partial z} - 2x_{10} \tag{3.24}$$

$$\dot{x_{12}} = x_7\frac{\partial^2 V}{\partial z \partial x} + x_8\frac{\partial^2 V}{\partial z \partial y} + x_9\frac{\partial^2 V}{\partial z^2} \tag{3.25}$$

$$\dot{x_{13}} = x_{16} \tag{3.26}$$

$$\dot{x_{14}} = x_{17} \tag{3.27}$$

$$\dot{x_{15}} = x_{18} \tag{3.28}$$

$$\dot{x_{16}} = x_{13}\frac{\partial^2 V}{\partial x^2} + x_{14}\frac{\partial^2 V}{\partial x \partial y} + x_{15}\frac{\partial^2 V}{\partial x \partial z} + 2x_{17} \tag{3.29}$$

$$\dot{x}_{17} = x_{13}\frac{\partial^2 V}{\partial y \partial x} + x_{14}\frac{\partial^2 V}{\partial y^2} + x_{15}\frac{\partial^2 V}{\partial y \partial z} - 2x_{16} \tag{3.30}$$

$$\dot{x}_{18} = x_{13}\frac{\partial^2 V}{\partial z \partial x} + x_{14}\frac{\partial^2 V}{\partial z \partial y} + x_{15}\frac{\partial^2 V}{\partial z^2} \tag{3.31}$$

$$\dot{x}_{19} = x_{22} \tag{3.32}$$

$$\dot{x}_{20} = x_{23} \tag{3.33}$$

$$\dot{x}_{21} = x_{24} \tag{3.34}$$

$$\dot{x}_{22} = x_{19}\frac{\partial^2 V}{\partial x^2} + x_{20}\frac{\partial^2 V}{\partial x \partial y} + x_{21}\frac{\partial^2 V}{\partial x \partial z} + 2x_{23} \tag{3.35}$$

$$\dot{x}_{23} = x_{19}\frac{\partial^2 V}{\partial y \partial x} + x_{20}\frac{\partial^2 V}{\partial y^2} + x_{21}\frac{\partial^2 V}{\partial y \partial z} - 2x_{22} \tag{3.36}$$

$$\dot{x}_{24} = x_{19}\frac{\partial^2 V}{\partial z \partial x} + x_{20}\frac{\partial^2 V}{\partial z \partial y} + x_{21}\frac{\partial^2 V}{\partial z^2} \tag{3.37}$$

$$\dot{x}_{25} = x_{28} \tag{3.38}$$

$$\dot{x}_{26} = x_{29} \tag{3.39}$$

$$\dot{x}_{27} = x_{30} \tag{3.40}$$

$$\dot{x}_{28} = x_{25}\frac{\partial^2 V}{\partial x^2} + x_{26}\frac{\partial^2 V}{\partial x \partial y} + x_{27}\frac{\partial^2 V}{\partial x \partial z} + 2x_{29} \tag{3.41}$$

$$\dot{x}_{29} = x_{25}\frac{\partial^2 V}{\partial y \partial x} + x_{26}\frac{\partial^2 V}{\partial y^2} + x_{27}\frac{\partial^2 V}{\partial y \partial z} - 2x_{28} \tag{3.42}$$

$$\dot{x}_{30} = x_{25}\frac{\partial^2 V}{\partial z \partial x} + x_{26}\frac{\partial^2 V}{\partial z \partial y} + x_{27}\frac{\partial^2 V}{\partial z^2} \tag{3.43}$$

$$\dot{x}_{31} = x_{34} \tag{3.44}$$

$$\dot{x}_{32} = x_{35} \tag{3.45}$$

$$\dot{x}_{33} = x_{36} \tag{3.46}$$

$$\dot{x}_{34} = x_{31}\frac{\partial^2 V}{\partial x^2} + x_{32}\frac{\partial^2 V}{\partial x \partial y} + x_{33}\frac{\partial^2 V}{\partial x \partial z} + 2x_{35} \tag{3.47}$$

$$\dot{x}_{35} = x_{31}\frac{\partial^2 V}{\partial y \partial x} + x_{32}\frac{\partial^2 V}{\partial y^2} + x_{33}\frac{\partial^2 V}{\partial y \partial z} - 2x_{34} \tag{3.48}$$

$$\dot{x}_{36} = x_{31}\frac{\partial^2 V}{\partial z \partial x} + x_{32}\frac{\partial^2 V}{\partial z \partial y} + x_{33}\frac{\partial^2 V}{\partial z^2} \tag{3.49}$$

$$\dot{x}_{37} = x_{40} \tag{3.50}$$

$$\dot{x}_{38} = x_{41} \tag{3.51}$$

$$\dot{x}_{39} = x_{42} \tag{3.52}$$

$$\dot{x}_{40} = x_{37}\frac{\partial^2 V}{\partial x^2} + x_{38}\frac{\partial^2 V}{\partial x \partial y} + x_{39}\frac{\partial^2 V}{\partial x \partial z} + 2x_{41} \tag{3.53}$$

$$\dot{x}_{41} = x_{37}\frac{\partial^2 V}{\partial y \partial x} + x_{38}\frac{\partial^2 V}{\partial y^2} + x_{39}\frac{\partial^2 V}{\partial y \partial z} - 2x_{40} \tag{3.54}$$

$$\dot{x}_{42} = x_{37}\frac{\partial^2 V}{\partial z \partial x} + x_{38}\frac{\partial^2 V}{\partial z \partial y} + x_{39}\frac{\partial^2 V}{\partial z^2} \tag{3.55}$$

APPENDIX 2: MARKLEY'S METHOD

Instead of propagating the varaitional equations the state transition matrix, here defined as ϕ, can be approximated by the Markley's method [16] as

$$\phi(t,t_0) \approx \begin{bmatrix} \phi_{rr} & \phi_{rv} \\ \phi_{vr} & \phi_{vv} \end{bmatrix} \tag{3.56}$$

where

$$\phi_{rr} \equiv I + (2G_0 + G)\frac{(\Delta t)^2}{6} \tag{3.57}$$

$$\phi_{rv} \equiv I\Delta t + (G_0 + G)\frac{(\Delta t)^3}{12} \tag{3.58}$$

$$\phi_{vr} \equiv (G_0 + G)\frac{(\Delta t)}{2} \tag{3.59}$$

$$\phi_{vv} \equiv I + (G_0 + 2G)\frac{(\Delta t)^2}{6} \tag{3.60}$$

and $G(t)$ is a 3×3 sub-matrix of the Jacobian of the system which is obtained by taking the partials of the vector field, resulting as

$$G(t) = \begin{bmatrix} \frac{\partial^2 V}{\partial x^2} & \frac{\partial^2 V}{\partial x \partial y} & \frac{\partial^2 V}{\partial x \partial z} \\ \frac{\partial^2 V}{\partial y \partial x} & \frac{\partial^2 V}{\partial y^2} & \frac{\partial^2 V}{\partial y \partial z} \\ \frac{\partial^2 V}{\partial z \partial x} & \frac{\partial^2 V}{\partial z \partial y} & \frac{\partial^2 V}{\partial z^2} \end{bmatrix} \tag{3.61}$$

The approximation requires less computations at each state of the propagation and so it makes it faster. Though not faster enough to compensate the loss in accuracy required to assess the results of this work.

APPENDIX 3: PYTHON CODE

This appendix includes the code and a brief description of the most important routines used in this project.

Binary System Construction

"make binary system" routine allows to create a binary system object that fits into the restricted full three body problem requirements. Allowing to set the uniform density of the system, the length of the axis of the tri-axial ellipsoid and the radius of the sphere.

```
class BinarySystem(object):
    G             = 0
    rho           = 0
    q             = 0
    w             = 0
    mu            = 0
    ellipsoid_A   = 0
    ellipsoid_B   = 0
    ellipsoid_C   = 0
    sphere_radius = 0

    def __init__(binary_system, G, rho, q, w, mu, ellipsoid_A, ellipsoid_B,
    ellipsoid_C, sphere_radius):
        binary_system.G             = G
        binary_system.rho           = rho
        binary_system.q             = q
        binary_system.w             = w
        binary_system.mu            = mu
        binary_system.ellipsoid_A   = ellipsoid_A
        binary_system.ellipsoid_B   = ellipsoid_B
        binary_system.ellipsoid_C   = ellipsoid_C
        binary_system.sphere_radius = sphere_radius

def make_binary_system(G, rho, q, w, mu, ellipsoid_A, ellipsoid_B, ellipsoid_C,
sphere_radius):
    binary_system = BinarySystem(G, rho, q, w, mu, ellipsoid_A, ellipsoid_B,
    ellipsoid_C, sphere_radius)
    return binary_system
```

Restricted Full Three-Body Problem Vectorfield and Variational Equations

"RF3BP42" returns the 42 differential equations, 6 from the flow and 36 from the variational equations evaluated at the provide state of a binary system at a certain time.

```
def RF3BP_42 (t, x, binary_system):

    # Particle state (position and velocity)
    state = x[0:6]

    x1  = x[0]
    x2  = x[1]
    x3  = x[2]
    x4  = x[3]
    x5  = x[4]
    x6  = x[5]
    x7  = x[6]
    x8  = x[7]
    x9  = x[8]
    x10 = x[9]
    x11 = x[10]
    x12 = x[11]
    x13 = x[12]
    x14 = x[13]
    x15 = x[14]
    x16 = x[15]
    x17 = x[16]
    x18 = x[17]
    x19 = x[18]
    x20 = x[19]
    x21 = x[20]
    x22 = x[21]
    x23 = x[22]
    x24 = x[23]
    x25 = x[24]
    x26 = x[25]
    x27 = x[26]
    x28 = x[27]
    x29 = x[28]
    x30 = x[29]
    x31 = x[30]
    x32 = x[31]
    x33 = x[32]
    x34 = x[33]
    x35 = x[34]
    x36 = x[35]
    x37 = x[36]
```

```
x38 = x[37]
x39 = x[38]
x40 = x[39]
x41 = x[40]
x42 = x[41]

# Angular velocity
w = binary_system.w
w_pow2 = w * w
# Sphere mass
mu_sphere = binary_system.mu
# Sphere position with respect ot CM
position_sphere = [1-mu_sphere, 0, 0]
xs = position_sphere[0]
ys = position_sphere[1]
zs = position_sphere[2]
# Particle position with respect to sphere
pos_x_sphere = x1 - xs
pos_y_sphere = x2 - ys
pos_z_sphere = x3 - zs
# Particle-sphere position squared
pos_x_sphere_pow2 = pos_x_sphere * pos_x_sphere
pos_y_sphere_pow2 = pos_y_sphere * pos_y_sphere
pos_z_sphere_pow2 = pos_z_sphere * pos_z_sphere
# Particle-sphre position modulus
pos_particle_sphere = np.sqrt(pos_x_sphere_pow2 + pos_y_sphere_pow2 +
pos_z_sphere_pow2)
pos_particle_sphere_pow3 = pos_particle_sphere * pos_particle_sphere *
pos_particle_sphere
# Gravitational acceleration due to ellipsoid
acc = ellipsoid_potential(state, binary_system)[1]
ax = acc[0]
ay = acc[1]
az = acc[2]
# Mutual potential first derivatives
Vx = w_pow2 * x1 + ax - (mu_sphere / pos_particle_sphere_pow3) * pos_x_sphere
Vy = w_pow2 * x2 + ay - (mu_sphere / pos_particle_sphere_pow3) * pos_y_sphere
Vz = az - (mu_sphere / pos_particle_sphere_pow3) * pos_z_sphere
# Local Jacobian
G = local_jacobian_G(state, binary_system)
Vxx = G[0,0]
Vxy = G[0,1]
Vxz = G[0,2]
Vyx = G[1,0]
Vyy = G[1,1]
Vyz = G[1,2]
Vzx = G[2,0]
Vzy = G[2,1]
```

```
    Vzz = G[2,2]
    # Vectorfield
    xdot = np.zeros(42)
    xdot[0]  = x4
    xdot[1]  = x5
    xdot[2]  = x6
    xdot[3]  = 2 * w * x5 + Vx
    xdot[4]  = -2 * w * x4 + Vy
    xdot[5]  = Vz
    # Variational equations
    xdot[6]  = x10
    xdot[7]  = x11
    xdot[8]  = x12
    xdot[9]  = x7*Vxx + x8*Vxy + x9*Vxz + 2*x11
    xdot[10] = x7*Vyx + x8*Vyy + x9*Vyz - 2*x10
    xdot[11] = x7*Vzx + x8*Vzy + x9*Vzz
    xdot[12] = x16
    xdot[13] = x17
    xdot[14] = x18
    xdot[15] = x13*Vxx + x14*Vxy + x15*Vxz + 2*x17
    xdot[16] = x13*Vyx + x14*Vyy + x15*Vyz - 2*x16
    xdot[17] = x13*Vzx + x14*Vzy + x15*Vzz
    xdot[18] = x22
    xdot[19] = x23
    xdot[20] = x24
    xdot[21] = x19*Vxx + x20*Vxy + x21*Vxz + 2*x23
    xdot[22] = x19*Vyx + x20*Vyy + x21*Vyz - 2*x22
    xdot[23] = x19*Vzx + x20*Vzy + x21*Vzz
    xdot[24] = x28
    xdot[25] = x29
    xdot[26] = x30
    xdot[27] = x25*Vxx + x26*Vxy + x27*Vxz + 2*x29
    xdot[28] = x25*Vyx + x26*Vyy + x27*Vyz - 2*x28
    xdot[29] = x25*Vzx + x26*Vzy + x27*Vzz
    xdot[30] = x34
    xdot[31] = x35
    xdot[32] = x36
    xdot[33] = x31*Vxx + x32*Vxy + x33*Vxz + 2*x35
    xdot[34] = x31*Vyx + x32*Vyy + x33*Vyz - 2*x34
    xdot[35] = x31*Vzx + x32*Vzy + x33*Vzz
    xdot[36] = x40
    xdot[37] = x41
    xdot[38] = x42
    xdot[39] = x37*Vxx + x38*Vxy + x39*Vxz + 2*x41
    xdot[40] = x37*Vyx + x38*Vyy + x39*Vyz - 2*x40
    xdot[41] = x37*Vzx + x38*Vzy + x39*Vzz

    return xdot
```

Ellipsoid Gravity Potential

"ellipsoid potential" returns the gravity potential and acceleration a single ellipsoid defined in the BinarySystem object is creating in an implicit position in the state vector of the particle.

```
def ellipsoid_potential (state, binary_system):

    # Input:
    #    - binary_system
    #    - state: particle state [x-position, y-position, z-position, x-velocity,
    y-velocity, z-velocity]
    #
    # Output:
    #    - Ve: Gravitational potential of the tri-axial ellipsoid
    #    - acc: Cartesian components of the gravitational acceleration
    (gradient of Ve) [ax, ay, az]

    # Ellipsoid geometry
    a = binary_system.ellipsoid_A
    b = binary_system.ellipsoid_B
    c = binary_system.ellipsoid_C

    # Ellipsoid semi-axes squared
    a_pow2 = a * a
    b_pow2 = b * b
    c_pow2 = c * c

    # Ellipsoid position respect to CM
    re = [-binary_system.mu, 0, 0]
    xe = re[0]
    ye = re[1]
    ze = re[2]

    # Ellipsoid normalized gravitational parameter
    mu_ellipsoid = 1 - binary_system.mu

    # Particle position respect to CM
    x = state[0]
    y = state[1]
    z = state[2]

    # Particle position respect to ellipsoid
    pos_x = x - xe
    pos_y = y - ye
    pos_z = z - ze

    pos_x_pow2 = pos_x * pos_x
```

```
pos_y_pow2 = pos_y * pos_y
pos_z_pow2 = pos_z * pos_z

# Set exception condition for the points inside the ellipsoid
if (pos_x_pow2 / a_pow2 + pos_y_pow2 / b_pow2 + pos_z_pow2 / c_pow2) <= 1:
    import sys
    sys.exit("Crash against ellipsoid")

else:
    # Confocal ellipsoid coefficients
    coef_3 = -1
    coef_2 = pos_x_pow2 + pos_y_pow2 + pos_z_pow2 - (a_pow2 + b_pow2 + c_pow2)
    coef_1 = (b_pow2 + c_pow2) * pos_x_pow2 + (a_pow2 + c_pow2) * pos_y_pow2 +
    (a_pow2 + b_pow2) * pos_z_pow2 - (a_pow2 * c_pow2 + b_pow2 * c_pow2 +
     a_pow2 * b_pow2)
    coef_0 = b_pow2 * c_pow2 * pos_x_pow2 + a_pow2 * c_pow2 * pos_y_pow2 +
    a_pow2 * b_pow2 * pos_z_pow2 - (a_pow2 * b_pow2 * c_pow2)
    C_coefficients = [coef_3, coef_2, coef_1, coef_0]

    # Confocal ellipsoid roots
    C_roots = np.roots(C_coefficients)

    # Store real roots in a list
    C_realroots = []
    for root in C_roots:
        if np.imag(root) < 1e-7:
            C_realroots.append(np.real(root))

    # Set kappa as the largest real root
    kappa = float(np.max(C_realroots))

    # Carlson elliptic integrals
    Rf0 = float(mpmath.elliprf(a_pow2 + kappa, b_pow2 + kappa, c_pow2 + kappa))
    Rdx = float(mpmath.elliprd(b_pow2 + kappa, c_pow2 + kappa, a_pow2 + kappa))
    Rdy = float(mpmath.elliprd(a_pow2 + kappa, c_pow2 + kappa, b_pow2 + kappa))
    Rdz = float(mpmath.elliprd(a_pow2 + kappa, b_pow2 + kappa, c_pow2 + kappa))

    # Gravitational potential of a tri-axial ellipsoid
    Ve = mu_ellipsoid * ((3/2) * Rf0 - (1/2) * (Rdx * pos_x_pow2 + Rdy *
    pos_y_pow2 + Rdz * pos_z_pow2))

    # Explicit cartesian components of the gravitational acceleration
    (gradient of Ve)
    ax = -mu_ellipsoid * pos_x * Rdx
    ay = -mu_ellipsoid * pos_y * Rdy
    az = -mu_ellipsoid * pos_z * Rdz

    acc = [ax, ay, az]
    return Ve, acc
```

Binary System Potential

"ellipsoid sphere potential" returns the gravity potential and acceleration a whole binary system defined in the BinarySystem object is creating in an implicit position in the state vector of the particle.

```
def ellipsoid_sphere_potential(state, binary_system):

    # Input:
    #    - binary_sytem
    #    - state
    # Output:
    #    - V: Ellipsoid-sphere mutual gravitational potential
    #    - acc: Cartesian components of the gravitational acceleration
    (gradient of Ve) [ax, ay, az]

    # Particle position respect to CM
    x = state[0]
    y = state[1]
    z = state[2]

    # Sphere radius
    radius_sphere = binary_system.sphere_radius
    radius_sphere_pow2 = radius_sphere * radius_sphere

    # Sphere position with respect ot CM
    position_sphere = [1-binary_system.mu, 0, 0]
    xs = position_sphere[0]
    ys = position_sphere[1]
    zs = position_sphere[2]

    # Particle position with respect to sphere
    pos_x_sphere = x - xs
    pos_y_sphere = y - ys
    pos_z_sphere = z - zs
    # Particle-sphere position squared
    pos_x_sphere_pow2 = pos_x_sphere * pos_x_sphere
    pos_y_sphere_pow2 = pos_y_sphere * pos_y_sphere
    pos_z_sphere_pow2 = pos_z_sphere * pos_z_sphere
    # Particle-sphre position modulus
    pos_particle_sphere = np.sqrt(pos_x_sphere_pow2 + pos_y_sphere_pow2
    + pos_z_sphere_pow2)

    # Sphere gravitational parameter
    mu_sphere = binary_system.mu

    # Ellipsoid gravitational parameter
    mu_ellipsoid = 1 - binary_system.mu

    # Sphere potential
    if ((pos_x_sphere_pow2 + pos_y_sphere_pow2 + pos_z_sphere_pow2)/
    radius_sphere_pow2) <= 1:
        import sys
        sys.exit("Crash against sphere")
    else:
        Vs = mu_sphere / pos_particle_sphere

    # Ellipsoid potential
    Ve, acc = ellipsoid_potential(state, binary_system)

    # Centrifugal force
    CF = (1/2) * binary_system.w*binary_system.w * (x*x + y*y)

    # Mutual potential
    V = CF + Ve + Vs + (1/2) * mu_sphere * mu_ellipsoid

    return V, acc
```

Propagation

Given a binary system and the initial conditions, it propagates a trajectory and the variational equations up to a certain final time. It returns the trajectory (position and velocity), time, state transition matrix and Jacobi constant (to check it is constant) vectors as well as a boolean that tells if the trajectory crashes against a body.

"Vode" method from Python's "scipy.integrate.ode" library has been used. It is a Real-valued Variable-coefficient Ordinary Differential Equation solver, with fixed-leading-coefficient implementation. It provides implicit Adams method (for non-stiff problems) and a method based on backward differentiation formulas (BDF) (for stiff problems). The parameters used along this whole work are listed as follows:

- Absolute tolerance: 10^{-7}
- Relative tolerance: 10^{-7}
- First step: 10^{-2}
- Minimum step: 10^{-4}
- Maximum step: 10^{0}
- Order: 8

```
def propagation_time_42 (f, binary_system, initial_state, DX, t0, tf,
dt, noise_scale, at, rt, fs, mins, maxs, int_order):
    # Evaluate if the initial state lies within a body
    crash = crashing_event(initial_state, binary_system)
    if crash == True:
        trajectory = initial_state
        time = t0
        STM = DX
        jacobi = [np.inf, np.inf]
     # If the initial state makes sense (does not lie within a body),
     start the propagation
    else:
        x = np.zeros(42)
        x[0]  = initial_state[0]
        x[1]  = initial_state[1]
        x[2]  = initial_state[2]
        x[3]  = initial_state[3]
        x[4]  = initial_state[4]
        x[5]  = initial_state[5]
        x[6]  = DX[0,0]
        x[7]  = DX[1,0]
        x[8]  = DX[2,0]
        x[9]  = DX[3,0]
        x[10] = DX[4,0]
        x[11] = DX[5,0]
        x[12] = DX[0,1]
        x[13] = DX[1,1]
        x[14] = DX[2,1]
        x[15] = DX[3,1]
        x[16] = DX[4,1]
        x[17] = DX[5,1]
        x[18] = DX[0,2]
        x[19] = DX[1,2]
        x[20] = DX[2,2]
        x[21] = DX[3,2]
        x[22] = DX[4,2]
        x[23] = DX[5,2]
        x[24] = DX[0,3]
        x[25] = DX[1,3]
        x[26] = DX[2,3]
        x[27] = DX[3,3]
        x[28] = DX[4,3]
        x[29] = DX[5,3]
        x[30] = DX[0,4]
        x[31] = DX[1,4]
        x[32] = DX[2,4]
        x[33] = DX[3,4]
        x[34] = DX[4,4]
```

```
x[35] = DX[5,4]
x[36] = DX[0,5]
x[37] = DX[1,5]
x[38] = DX[2,5]
x[39] = DX[3,5]
x[40] = DX[4,5]
x[41] = DX[5,5]

# Evaluate if the initial state lies within a body
crash = crashing_event(initial_state, binary_system)
if crash == True:
    trajectory = initial_state
    time = t0

# If the initial state makes sense (does not lie within a body),
start the propagation
else:
    # Initialize state matrix
    trajectory = initial_state
    # State transition matrix
    STM = x
    # Initialize time vector
    time = t0
    # Set time parameter
    t = t0
    # Set integrator
    solver = integrate.ode(f).set_integrator('vode', atol=at,
    rtol=rt, method = 'adams', first_step=fs, min_step=mins,
    max_step=maxs, order=int_order)
    # Set initial values for the integrator
    solver.set_initial_value(x, t0)
    solver.set_f_params(binary_system)

    while solver.successful() and t < tf:
        # Measurement noise
        noise = np.random.normal(0, noise_scale, len(initial_state))
        # Integration step
        solver.integrate(t + dt)
        # Time at current step
        t = solver.t
        # State at current step
        x = solver.y
        state = x[0:6] + noise
        # Jacobi constant at current step
        potential = ellipsoid_sphere_potential(state, binary_system)[0]
        cj = jacobi_constant(state, potential)

        # Store values in vectors
```

```
            time = np.vstack((time, t))
            trajectory = np.vstack((trajectory, state))
            STM = np.vstack((STM, x))
            jacobi = np.vstack((jacobi, cj))

            # Evaluate crashing event at current state
            crash = crashing_event(state, binary_system)
            if crash == True:
                break

    return trajectory, time, crash, STM, jacobi
```

Poincaré Map

It computes the crossing events of a given trajectory with respect to a Poincaré section which can be set by the user by looking at the change of sign. It also applies a refinement of the points by means of a Newton-Raphson propagation in order to get accurate Poincaré maps.

```
def poincare_map(binary_system, trajectory, time):
    from rkf45 import rkf45
    initial_state = trajectory[0]
    count = 0
    # Poincaré section
    y0 = initial_state[1]
    for state in trajectory:
        # Poincaré section
        yk = state[1]
        if np.dot(y0, yk) < 0:
            if count == 0:
                crossing = state
                count += 1
            else:
                crossing = np.vstack((crossing, state))
        # Poincaré section
        y0 = state[1]

    if count == 0:
        return 0, False
    else:
        # Newton-raphson refinement (y=0)
        f = RF3BP
        count=0
        hmin = 1e-4
        hmax = 1e0
        tol = 1e-7
        for x in crossing:
            t = 0
            while x[1] > 1e-12:
                h = -x[1]/x[4] # h = -g(t)/g'(t)
                [t, x, h, err] = rkf45(t,x,h,hmin,hmax,tol,f,binary_system)
                if count == 0:
                    poincare = x
                    count += 1
                else:
                    poincare = np.vstack((poincare, x))

        return poincare, True
```

Finite Time Lyapunov Exponents

Given a state transition matrix propagation on time, it computes the finite time Lyapunov exponent of a trajectory considering just the last state.

```
def finite_lyapunov(STM, time):
    x = STM[-1]
    t = time[-1]
    # STM at final step
    DX_t = np.matrix([[x[6],x[7],x[8],x[9],x[10],x[11]],
                  [x[12],x[13],x[14],x[15],x[16],x[17]],
                  [x[18],x[19],x[20],x[21],x[22],x[23]],
                  [x[24],x[25],x[26],x[27],x[28],x[29]],
                  [x[30],x[31],x[32],x[33],x[34],x[35]],
                  [x[36],x[37],x[38],x[39],x[40],x[41]]])
    DX = np.transpose(DX_t)
    # Lyapunov exponent
    lyapunov = (1/t) * np.log((np.linalg.norm(DX, ord=2)))

    return float(lyapunov)
```

Scaling

It propagates a measured trajectory with the real values of the binary system and a noise scale of 10^{-4} and stores it. Then, optimizes the dynamical parameter by propagating the estimated trajectories within a certain time range. For each length of the trajectory it returns an optimized parameter which is then compared to the real one.

It keeps optimizing until it gets an accuracy of two orders of magnitude with respect to the real parameter. Finally, it returns the slope of the log-log plot absolute error vs. time which is defined as the scaling exponents, α.

It uses the "Nelder-Mead" method from the Python's "scipy.optimize.minimize" library calling the objective function $Q(\mu)$ and using the real value of the system as initial guess in order to speed the process.

```
# Objective function Q(mu)
def error_vs_mu(e_mu, e_tf, real_binary_system, m_trajectory, scale):
    # Estimated binary system
    estimated_binary_system = make_binary_system(real_binary_system.G,
    real_binary_system.rho, real_binary_system.q, real_binary_system.w,
    e_mu, real_binary_system.ellipsoid_A, real_binary_system.ellipsoid_B,
    real_binary_system.ellipsoid_C, real_binary_system.sphere_radius)
    # Integration data
    f = RF3BP
    initial_state = m_trajectory[0]
    t0 = 0
    dt = 1e-2
    atol = 1e-7
    rtol = 1e-7
    first_step = 1e-2
    min_step = 1e-4
    max_step = 1e0
    order = 8
    # Estimated propagation
    e_trajectory, e_time, e_crash = propagation_time (f, estimated_binary_system, ir

    # Least-squares comparison
    Q = least_squares(m_trajectory, e_trajectory, scale)

    return Q

# Scaling routine
def scaling(initial_state, real_binary_system, noise_scale):
    # Vectorfield to integrate
    f = RF3BP
    # Initial integration time
    t0 = 0
    # Final integration time
    tf = 100
    # Time step
    dt = 1e-2
    # Absolute tolerance
    atol = 1e-7
    # Relative tolerance
    rtol= 1e-7
    # First integration step
    first_step = 1e-2
    # Minimum integration step
    min_step = 1e-4
    # Maximum integration step
    max_step = 1e0
    # Integration order
```