Summer 2021

Stability of Earth's Trajectory inside a Binary Star System

Michael Krunic

San Jose State University

STABILITY OF EARTH'S TRAJECTORY INSIDE A BINARY STAR SYSTEM

A Thesis

Presented to

The Faculty of the Department of Physics

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Michael S. Krunic

August 2021

The Designated Thesis Committee Approves the Thesis Titled

STABILITY OF EARTH'S TRAJECTORY INSIDE A BINARY STAR SYSTEM

by

Michael S. Krunic

APPROVED FOR THE DEPARTMENT OF PHYSICS

SAN JOSÉ STATE UNIVERSITY

August 2021

Patrick Hamill, Ph.D.	Department of Physics and Astronomy
Thomas Madura, Ph.D.	Department of Physics and Astronomy
Aaron Romanowsky, Ph.D.	Department of Physics and Astronomy

ABSTRACT

STABILITY OF EARTH'S TRAJECTORY INSIDE A BINARY STAR SYSTEM

by Michael S. Krunic

The evolution of the solar system is an interesting dynamical problem in celestial mechanics. Computer simulations have shown that planetary bodies in multiple-body systems become unstable after a long time. In this thesis, through numerical simulations, we investigate the stability of the Earth in a binary star system. In the Sun, Earth, and Jupiter three-body system, we treat Jupiter as the second star (Star 2) and we simulate the Earth's orbit over many orbital periods. In our program, we generate the positions, velocities, accelerations, and other orbital elements of each body using Newton's laws of motion. We explore four types of simulations including when the system is run with the usual locations and masses, when the mass of Star 2 is changed at the position of Jupiter, when the stars have equal mass and the location of Star 2 is changed, and when the position and mass of Star 2 changed. We plot the position coordinates of the Earth and determine when the Earth exhibits unpredictable or erratic behaviors.

ACKNOWLEDGMENTS

First and foremost, I want to thank my thesis advisor and teacher, Dr. Patrick Hamill, for sharing his expertise and for his support throughout this project. I am grateful to him and appreciate his patience during the struggles and frustrations completing this thesis. I will never forget the intellectual freedom he granted me to create this work.

I can't thank Dr. Thomas Madura enough for agreeing to join the committee. I want to thank him because his computational physics course was instrumental to this project. It was in this class that I learned to appreciate programming and its relevance in physics. I will always be grateful for the conversations that I had with Dr. Madura about celestial mechanics, astrophysics, and in general, programming.

A warm and special thanks to Dr. Aaron Romanowsky for participating on my committee. Although, most of our conversations were brief, I appreciate the thoughtful advice and the genuine interest in my project and in the possible Ph.D. programs I hope to attend in the near future.

Thanks to my family, friends, and colleagues at school for all of the support you have provided me over the years. I appreciate everyone and will always remember the support, conversations, and continuous motivation for completing this project.

DEDICATION

My thesis is dedicated to my family and friends, my colleagues and professors, and everyone else who has come along with my journey of becoming a scientist.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

Order and regularity are paradigms for describing the motion of planets in our solitary star Solar System: The Sun's trajectory is placed at the center while the eight planets revolve around it in stable trajectories. However, once a second star is introduced and certain parameters are changed, instability may result. In this thesis, we explore the orbital stability of Earth in the presence of an additional star as a three-body problem involving Earth, the Sun, and the hypothetical second star. We consider the following: What effect does the second star's mass have on Earth's trajectory? What effect does the second star's position have on Earth's trajectory? What are the conditions for which Earth is stable and those for which Earth is unstable?

1.1 Overview of the Thesis

In the introduction we review the history and elements of celestial mechanics and discuss the motivation for studying stability problems. We present previous stability studies involving the three-body problem and discuss how they relate to this thesis. We briefly address a three-body study involving the Alpha Centauri system, the Planet Nine Hypothesis, and discuss how a binary star system may be able to capture a planet. We present a calculation of the apparent brightness of a hypothetical second star in our Solar System.

In Chapter 2 we discuss the physics and derive the constants of the two-body problem. Using the two-body model, we individually simulate each of the eight planets around the Sun. We use the results of this simulation to validate our computer program. We derive the stability condition for the two-body problem and show that the inverse-square law leads to the stability condition.

In Chapter 3 we present the equations that are used in the three-body analysis, including those derived from the disturbing function. We present Cowell's perturbation method and show the results of how perturbation influence Earth's and Jupiter's orbit around the Sun. We compare the simulations of the orbital energy and angular momentum of the two-body and three-body problems.

In Chapter 4 we cover the numerical methods and algorithms used throughout the thesis. We discuss the Euler-Cromer and the Runge Kutta 4^{th} order methods. We outline the method of computation by including the algorithms we used to generate the orbital elements.

In Chapter 5 we present the results from our simulations. We discuss how changing the mass and perihelion position of Star 2 affects the trajectory of Earth's orbit in a binary star system.

In Chapter 6 we provide a discussion for the results and address the questions listed in the introduction.

1.2 Historical Review of Celestial Mechanics

Celestial mechanics was developed in the 16^{th} and 17^{th} centuries. Johannes Kepler discovered his laws of planetary motion which described the orbits of the planets about the Sun as being elliptical rather than circular, and demonstrated that a planet's velocity changes depending on its distance from the Sun. Kepler's three empirical laws of planetary motion are:

1) A planet follows an elliptical orbit about the Sun positioned at one of the foci.
2) The radius vector from the Sun to a planet sweeps out equal areas in equal times.
3) A planet's orbital period T is proportional to the planet's semi-major axis a by

$$T^2 \,\alpha\, a^3. \tag{1}$$

Mathematically, an ellipse can be expressed as

$$r = \frac{a(1-e^2)}{1+e\cos\theta}, \tag{2}$$

where r is the distance from the Sun to the planet, e is the eccentricity of the ellipse, θ is the angle between the planet's current position and the perihelion position, and a is the semi-major axis.

Like Kepler, Issac Newton investigated the dynamics of our Solar System. In his *Principia*, he states his laws of motion and the law of universal gravitation, and provides the derivation of Kepler's law of planetary motion. Newton's three laws of motion are:

1) A body remains in a state of rest or in uniform motion unless a force acts upon it.
2) The force experienced by a body is equal to the rate of change of its momentum.
3) To every action there is an equal and opposite reaction.

His law of universal gravitation states that every particle with a mass attracts every other particle with a force that is directly proportional to the mass and inversely proportional to the square of the distance; the magnitude of the force $|\mathbf{F}|$ between any two masses, m_1 and m_2 separated by a distance r can be expressed as

$$|\mathbf{F}| = G\frac{m_1 m_2}{r^2}, \tag{3}$$

where G is the gravitational constant. Using equation (3), Newton confirmed Kepler's empirical laws and showed that the resulting motion of a planet around the Sun is a mathematical curve called a conic section.

The motion of two bodies moving around their common center of mass can be solved analytically and the resulting motion is closed in inertial space. The motion of the bodies becomes complicated when an additional body is introduced into the system. In this three-body problem, each set of initial conditions leads to different outcomes, making the trajectories of the bodies unpredictable and the solution unobtainable. Between $1784 - 1786$, the French mathematician, Pierre Simon de Laplace, attempted to solve the instability problem by studying the Sun, Jupiter, and Saturn system. Laskar's (2013) historical review of planetary stability in our Solar System provides the observations that showed that Jupiter was gradually moving closer to the Sun while Saturn was moving away from it. This was based on the previous work by Kepler, Lagrange, and other astronomers working on the problem. Laplace attempted to understand the behavior of these planets by studying the irregularities in the semi-major axes of Jupiter and Saturn. He looked at the orbital energies and found that the quantity

$$\frac{m_J}{a_J} + \frac{m_S}{a_S}, \tag{4}$$

must remain constant according to Newton's laws, where m_J and m_S are the masses and a_J and a_S are the semi-major axes of Jupiter and Saturn, respectively. Using Kepler's third law, in the form of $n^2 a^3 = \mu$, the ratio between the mean motions, n_J and n_S, is

$$\frac{n_S}{n_J} = -\frac{m_J}{m_S}\sqrt{\frac{a_J}{a_S}}. \tag{5}$$

The mean motions of Jupiter and Saturn are commensurate with each other in a $5:2$ ratio between their orbital periods; that is, Jupiter orbits the Sun five times while Saturn orbits two in a given amount of time. This discovery was called the Great Jupiter-Saturn Inequality and was claimed to be the most important advance in astronomy since Issac Newton. Despite its importance, the methods of Laplace remained inconclusive when applied to the motion of the three-body problem: When the system is subjected to different initial conditions, the outcome is unpredictable.

In the 19^{th} century, Henri Poincaré showed that the three-body problem does not lead to a closed-form solution by looking at a special case. In his restricted circular three-body problem (RC3BP), he showed that there are an infinite number of periodic solutions. In the RC3BP, two of the bodies follow circular orbits around their common center of mass while the mass of the third body is neglected. Not only are there many solutions, but the sensitivity to the initial conditions impacts the trajectories of the bodies. If the initial conditions are varied slightly, the trajectory leads to a different path every time.

Poincaré's work on the three-body problem and RC3BP became the mathematical basis for chaos theory, which involves the sensitivity of dynamical systems to initial conditions.

A disadvantage of Poincaré's era was the lack of computing power necessary to calculate the equations of motions for the three-body problem. With the advent of modern powerful computers, we are equipped to apply experimental and numerical methods to multi-body systems and solve nonlinear equations of motion, including those of the planetary instability problem. Modern computing power also allows for simulations of these dynamical systems over large time scales, which may be used to predict future events such as the evolution of planetary systems.

1.3 Previous Studies of the Planetary System Stability

We now provide the definition of stability, background on the previous studies of instability in planetary and binary star systems, the consequences of stability, and a calculation of the apparent brightness of a binary star companion.

1.3.1 Definition of Stability

The definition of stability depends on the type of study being conducted or the deviations that are being tracked with the orbital elements. Graziani and Black (1981) defined stability in terms of secular changes in the planetary orbits, which depend on the separation distance and the mass ratio of the companion planets. Pendleton and Black (1983) defined a lack of stability to be when the orbit of a planet develops a clear secular trend or when erratic behavior can be observed. Weigert and Holman (1997) defined stability in terms of the change in inclination of the orbital plane of the binary pairs in the system. Stability depends on the orbit elements of the system including the semi-major axis (a), the eccentricity (e), the inclination (i), and the ratio between the masses (μ) in the system. There are other important orbital elements used for describing stability such as the time of pericenter passage (τ), the angle of the ascending node (Ω), and the argument of pericenter (ω). The reader is referred to any standard text of celestial mechanics such as Danby (1962) or Murray and Dermott (1999) for information about the orbital elements. In this thesis, when we refer to an orbit being unstable, we mean the planet deviates from the initial path or shows erratic behavior.

1.3.2 Instability in Three-body Systems

Graziani and Black studied the stability constraints on three-body systems, in particular, a system composed of a single star and two planetary companions. Based on the numerical experiments they performed, they showed that a system made up of the Sun and two masses like Jupiter, separated by 0.28 AU located at the present day positions of Venus and Earth, were stable for up to 10^4 years. This result extended the findings by Donnison and Williams (1978) who showed such a system was stable for 10^3 years. Graziani and Black also showed that a system made up of the Sun, Jupiter, and Saturn becomes unstable when the masses of the planets increase by a factor of 30 greater than the original mass values and the system remains stable when the masses are a factor of 20 or less. They confirmed the result of Nacozy (1976) who showed the factor was 29.5 when the planets were separated by a distance of 4.34 AU. In their own study, they explored the stability constraints of the Sun, Neptune, and Uranus system when the planets were separated by 10.9 AU. The main result that they determined was the condition for orbital stability. They compared the critical mass μ_{crit} and the mean mass μ of the planets and the star. When $\mu \geq \mu_{\text{crit}}$, the companion planets becomes unstable within a few orbits. For this condition, they determined the expression

$$\mu_{\text{crit}} = 0.175\Delta^3(2-\Delta)^{-\frac{3}{2}}, \tag{6}$$

and used

$$\mu = \frac{1}{2}\frac{(m_1+m_2)}{M_{\text{star}}}, \tag{7}$$

to determine the ratio of the masses, where m_1 and m_2 are the masses of the planetary companions, M_{star} is the mass of the star, and $\Delta = \frac{2(R-1)}{R+1}$ is the minimum (initial) separation between the companions in units of their mean distance from the central star where $R = \frac{R_2}{R_1}$. For this system, they varied the distance between $0.39 \leq \Delta R \leq 14.4$ and found instabilities when $5.7 \times 10^{-3} \leq \mu \leq 6.7 \times 10^{-3}$. They found when one of the masses in the system is five times larger than the mass of Jupiter, the system displays instabilities.

Pendleton and Black (1983) furthered the criterion for dynamical instability in the three-body problem. They constructed their system as a binary star system with a planet orbiting both within and outside of the binary pair and confirmed the stability criterion derived by Graziani and Black. They investigated different types of orbital configurations including inner and outer planet configurations and tested how inclination and eccentricity of the binary pair influences the stability of the orbit of the tertiary body. In their studies, they showed that prograde orbits are stable while the same system is not stable for retrograde orbits. To test the orbital stability on inclination, they set the binaries to have equal mass where the tertiary mass was equal to the mass of Jupiter. Their results showed that inclination has a strong effect on the inner planetary configuration and instability occurs between $50^\circ \leq i \leq 70^\circ$. Inclination did not have an effect on stability for the outer planet configuration. Their main experiment was studying the effects of eccentric binary orbits on the orbital stability of a planetary body in the inner planet configuration. When the tertiary mass was much greater or much less than the reduced mass of the binary pair, orbital stability was a strong function of eccentricity.

1.3.3 Stability in Binary Star Systems

In this section, we consider the stability of a planet in a binary star system. Two important studies that have been carried out are by R. Dvorak et al. (1989) and Wiegert and Holman (1997). Both of these studies consider the eccentricity of a binary star system as a function of the ratio of the binary masses and treat their systems as an elliptical three-body problem. Dvorak et al. derived the stability limits of a planetary orbit around the binary pair and focused on the relationship between the eccentricity and the mass ratio of the binary pairs. Wiegert and Holman calculated the limits on the semi-major axis. The critical semi-major axis was determined for the ranges $0 \leq e \leq 0.8$ and $0.1 \leq \mu \leq 0.9$ and can be expressed as

$$a_{\text{crit}} = [(0.464 \pm 0.006) + (-0.380 \pm 0.010)\mu + (-(0.631 \pm 0.034)e \\ + (0.586 \pm 0.061)\mu e + (0.150 \pm 0.041)e^2 + (-0.198 \pm 0.074)\mu e^2]a_b, \quad (8)$$

where a_b is the binary semi-major axis, e is the binary eccentricity, and μ is the mass ratio. Using this result, they were able to identify regions of phase space where planets can persist in a binary star system over large time scales.

Alpha Centauri is located 1.3 parsecs from the Sun which makes it a prime system to look for the existence of planets and a logical starting point for theoretical investigations of stability. Wiegert and Holman (1996) studied the Alpha Centauri system and they investigated the long-term stability of small bodies near the central binary stars, Alpha and Beta. They considered a system composed of stars forming a binary pair, with a semi-major axis of 23 AU. The third body is orbiting the pair at a distance of $12,000$ AU. The smaller body began on a circular orbit and the simulation was carried out for $32,000$ binary periods or approximately 2.5 Myr. Their results showed that a body is unstable in the interior and exterior regions of the binary pair when the semi-major axis of the particle is $a_p < 3a_b$ and $a_p > 0.2a_b$, where a_p and a_b are the semi-major axes of the particle and the binary pair. The simulation cannot assure the stability of a planet, but they identified important unstable regions where the planet cannot exist. In the Alpha Centauri system, the planet cannot exist where $a_p < 3.5$ AU and $a_p > 70$ AU from the primary.

1.3.4 Consequences of Stability

Due to the difficulties of solving problems involving the dynamics of interacting bodies, we must apply numerical methods to study these systems. Numerical methods have become a fundamental technique for understanding physical problems in celestial mechanics. Pakter and Levin (2018) use Runge-Kutta methods with an adaptive step size to demonstrate that planetary systems are susceptible to events such as the collision or the ejection of planets. They claimed that finding initial conditions that lead to the stability of planetary orbits is impossible because of the finite time limit of the simulations. We cannot say, with certainty, how long our Solar System will be stable. Another study, by Batygin and Laughlin (2002), integrated the orbital motions of the solar system over 20 Gyr. Their experiments showed that Mercury falls into the Sun approximately 1.261 Gyr from now, and in another they showed that Mercury and Venus collide in 862 Gyr. Mars was also ejected from the system in the experiment as a result of Mercury's unstable orbit.

1.3.5 The Planet Nine Hypothesis

Planet Nine is a hypothetical planet that could exist in the outer region of the solar system between 400 – 800 AU. It is proposed that the addition of a planet like Planet Nine could explain the gravitational effects on the clustering of the Extreme Trans-Neptunian Objects and the high inclinations of the orbits of these objects. The Planet Nine hypothesis proposed by Batygin and Brown (2019) is inconclusive; however, there are interesting features about the stability of the solar system if Planet Nine were to exist. One of these features is that our Solar System could have contained two stars during formation of the solar system. In a binary star system, it is plausible for a planet like Planet Nine to exist in the outer region of the system. Siraj and Loeb (2020) investigated how a binary star model could explain the existence of Planet Nine or the possibility that Planet Nine is a rogue planet from another system and was captured by our Solar System. In their study, they showed that an equal-mass, temporary companion to the Sun increased the likelihood of a binary star system capturing Planet Nine. Computer simulations have shown binary star systems are efficient at capturing planets and objects similar to those in the Oort Cloud, including Planet Nine. The model proposed by Siraj and Loeb favors capturing Planet Nine over previously studied lone star models.

1.4 Apparent Brightness of a Binary Star Companion

If a companion star exists, then we would be able to see if it is close enough or if it is bright enough. An illustrative exercise (Romanowsky, 2021) shows us the luminosity of the companion star mentioned by Siraj and Loeb. We assume the Sun and the hypothetical companion star are identical light sources. We call the apparent magnitudes of the Sun and the companion star, m_{sun} and m_{star}, and the distance the stars are located from the Earth, d_{sun} and d_{star}. The relationship between the luminosity of a star and its apparent magnitude can be expressed as

$$\frac{I_a}{I_b} = (2.512)^{(m_b - m_a)}, \tag{9}$$

where m_a, m_b are the apparent magnitudes of the stars and I_a, I_b are the intensities of those stars.

The apparent magnitude depends on the distance from the Earth to the Star. A light source becomes fainter as the star moves away from the Earth or becomes brighter the closer it is. We can use the relationship between the intensity and the distances of the star. Since the light sources are identical, the ratio of the intensities can be expressed as

$$\frac{I_a}{I_b} = \left(\frac{d_a}{d_b}\right)^2, \tag{10}$$

where d_a, d_b are the distances from the observed position.

We combine equations (9) and (10) to obtain a relationship between the apparent brightness and the distance

$$m_b - m_a = 5.0 \times \log_{10}\left(\frac{d_b}{d_a}\right). \tag{11}$$

The values from the Sun, $m_{\text{sun}} = -26.5$ and $d_{\text{sun}} = 1$ AU, are used to calculate the apparent brightness of Star 2. The distance to Star 2 will be the distance proposed from Siraj and Loeb, $d_{\text{star}} = 1500$ AU. We can calculate the apparent brightness of the Star 2:

$$\begin{aligned}
m_{\text{star}} &= m_{\text{sun}} + 5.0 \times \log_{10}\left(\frac{d_{\text{star}}}{d_{\text{sun}}}\right), \\
m_{\text{star}} &= -26.5 + 5.0 \times \log_{10}\left(\frac{1500\text{AU}}{1\text{AU}}\right), \\
m_{\text{star}} &= -10.61.
\end{aligned} \tag{12}$$

The result shows the apparent magnitude of Star 2 is -10.61. The apparent magnitude falls between the apparent magnitude of a full Moon (-12.5) and Venus at its brightest (-4.4). A quick calculation evaluates the brightness of Star 2 compared to the Moon, yielding the following result:

$$\begin{aligned}
\frac{b_{\text{star}}}{b_{\text{moon}}} &= (2.512)^{(m_{\text{moon}} - m_{\text{star}})}, \\
\frac{b_{\text{star}}}{b_{\text{moon}}} &= (2.512)^{[-12.5-(-10.6)]}, \\
\frac{b_{\text{star}}}{b_{\text{moon}}} &= 0.175.
\end{aligned} \tag{13}$$

The brightness of the star is about 17.5% the brightness a full Moon. A star with this brightness would be visible in the night sky, even at a distance of 1500 AU. However, we do not observe such a star. This analysis does not answer whether the star exists but gives us some intuition about the brightness of a hypothetical star in the night sky. The hypothetical star has to be orders of magnitudes less in luminosity, the star must be farther away, or it does not exist at all.

2 THE TWO-BODY PROBLEM

The two-body problem (2BP) is important in celestial mechanics because we can approximate important orbital dynamics using a point mass moving under the gravitational influence of a central, dominant mass. For example, we can analyze a smaller body like the Earth orbiting around a larger central body like the Sun. The mass of Earth, which is $m_E = 5.94 \times 10^{24}$ kg, moves around the Sun, $m_S = 1.99 \times 10^{30}$ kg, in an elliptical orbit with a semi-major axis a. When we investigate the 2BP, we obtain constants of motion which are crucial to the analysis. In this chapter, we present the physics of the 2BP and derive the constants of motion. These are the orbital specific energy, specific angular momentum, and the eccentricity or the Laplace-Runge-Lenz vector. Then we look at the results from simulating the 2BP to verify that our code runs correctly. Lastly, we derive the stability condition in the 2BP using the analogy of the harmonic oscillator.

2.1 The Physics of the Two-body Problem

The physics of the 2BP is well known and we will discuss its main components. We will first look at the gravitational forces between two point masses in Cartesian space. Figure 1 shows the configuration of the system relative to a fixed origin. The masses, m_1 and m_2, are initially located at some positions in space, which we label as vectors, $\mathbf{r}_1$ and $\mathbf{r}_2$. The positions of the masses are relative to a fixed origin and the motion is observed in an inertial reference frame.

We assume the gravitational interaction between the masses is an attractive force, dependent on the separation distance between the pair. We label the force as a function of distance

$$\mathbf{F} = F(\mathbf{r})\hat{\mathbf{r}}. \tag{14}$$

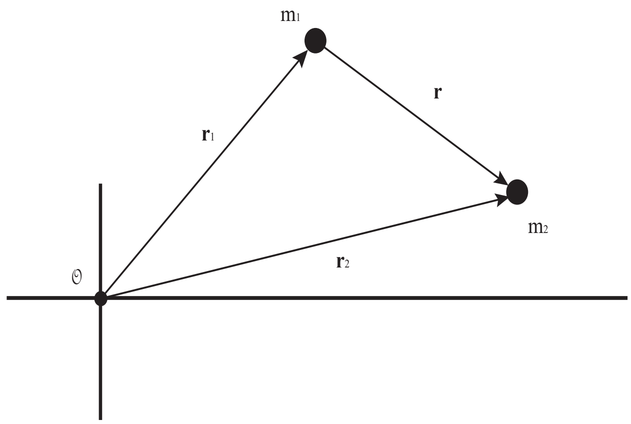

Fig. 1. Two point masses, m_1 and m_2 relative to a fixed origin in space. The position vectors are labeled $\mathbf{r}_1$ and $\mathbf{r}_2$, and $\mathbf{r}$ is the difference in the position vectors.

The separation $\mathbf{r}$ between the bodies is the difference between their position vectors

$$\mathbf{r} = \mathbf{r}_2 - \mathbf{r}_2. \tag{15}$$

The unit vector in the direction of $\mathbf{r}$ is

$$\hat{\mathbf{r}} = \frac{1}{r}\mathbf{r}. \tag{16}$$

Using Newton's second and third law, we obtain the pair of equations used to analyze the 2BP. We end up with the equations below:

$$-\mathbf{F} = m_2\ddot{\mathbf{r}}_2, \tag{17}$$

and

$$\mathbf{F} = m_1\ddot{\mathbf{r}}_1. \tag{18}$$

The accelerations, $\ddot{\mathbf{r}}_1$ and $\ddot{\mathbf{r}}_2$, are the second derivatives of the position with respect to time of the masses; these are the relative accelerations of the masses. The terms can be written so that the equations can be expressed as the difference between accelerations

$$\ddot{\mathbf{r}}_2 - \ddot{\mathbf{r}}_1 = \Big(-\frac{\mathbf{F}}{m_2} + \frac{\mathbf{F}}{m_1}\Big) = \Big(-\frac{1}{m_2} + \frac{1}{m_1}\Big)\mathbf{F}. \tag{19}$$

The reduced mass of the system can be expressed as μ_{reduced}, simplifying equation (19)

$$\mu_{\text{reduced}} = \frac{1}{m_1} + \frac{1}{m_2} = \frac{m_1 + m_2}{m_1 m_2}. \tag{20}$$

For simplicity, the smaller of the two masses of the system is treated as a point-like mass. The force is the gravitational force between the masses and the magnitude of the force is a product of the masses and the inverse-square of the separation distance

$$|\mathbf{F}| = F = G\frac{m_1 m_2}{r^2}, \tag{21}$$

where $G = 6.674 \times 10^{-11}\text{Nm}^2\text{kg}^{-2}$ is the universal gravitational constant. Inserting equation (21) into equation (19) yields the relative equation of motion for the 2BP

$$\ddot{\mathbf{r}} = -\frac{m_1 + m_2}{m_1 m_2} G\frac{m_1 m_2}{r^2}\hat{\mathbf{r}}. \tag{22}$$

We set $\mu = G(m_1 + m_2)$ to express the equation in a simpler form and we obtain the equation of motion used in our analysis of the 2BP

$$\ddot{\mathbf{r}} + \frac{\mu}{r^3}\mathbf{r} = 0. \tag{23}$$

Later in the discussion, we apply the equation of motion to the Sun-Planet system, where the planet is each of the planets in the solar system.

2.2 The Constants of the Two-body Problem

The integrals of the 2BP are derived by using the equation of motion (23). The important constants are the energy integral (C), the angular momentum integral $(\mathbf{h})$, and the Laplace-Runge-Lenz vector or the eccentricity vector $(\mathbf{B})$. For the analysis, we assume m_1 will be located at a fixed origin in space and the motion of m_2 will be with respect to m_1. This is valid because $m_1 >> m_2$.

2.2.1 The Energy Constant

The energy constant is derived by taking the dot product of the velocity vector, $\dot{\mathbf{r}}$, and the equation of motion (23)

$$\dot{\mathbf{r}} \cdot \left(\ddot{\mathbf{r}} + \frac{\mu}{r^3} \mathbf{r} \right) = 0. \tag{24}$$

Using dot product rules, we distribute the velocity vector across the equation of motion. This leads to the equation

$$\dot{\mathbf{r}} \cdot \ddot{\mathbf{r}} + \left(\frac{\mu}{r^3} \right) \dot{\mathbf{r}} \cdot \mathbf{r} = 0. \tag{25}$$

The first term of equation (25) can be re-written as

$$\dot{\mathbf{r}} \cdot \ddot{\mathbf{r}} = \frac{1}{2} \frac{d}{dt} (\dot{\mathbf{r}} \cdot \dot{\mathbf{r}}). \tag{26}$$

The second term on the right of equation (25) can be re-written as

$$\dot{\mathbf{r}} \cdot \frac{\mu}{r^3} \mathbf{r} = \frac{1}{2} \frac{\mu}{\sqrt{(\mathbf{r} \cdot \mathbf{r})^3}} \frac{d}{dt} (\mathbf{r} \cdot \mathbf{r}) = \frac{d}{dt} \left(- \frac{\mu}{\sqrt{\mathbf{r} \cdot \mathbf{r}}} \right), \tag{27}$$

where $|r| = \mathbf{r} \cdot \mathbf{r}$. Next, we combine equations (26) and (27) to obtain

$$\frac{1}{2} \frac{d}{dt} (\dot{\mathbf{r}} \cdot \dot{\mathbf{r}}) + \frac{d}{dt} \left(- \frac{\mu}{\sqrt{\mathbf{r} \cdot \mathbf{r}}} \right) = \frac{d}{dt} \left(\frac{1}{2} (\dot{\mathbf{r}} \cdot \dot{\mathbf{r}}) - \frac{\mu}{r} \right) = 0. \tag{28}$$

Integrating equation (28) we obtain the first constant in our problem

$$\frac{1}{2} (\dot{\mathbf{r}} \cdot \dot{\mathbf{r}}) - \frac{\mu}{r} = C. \tag{29}$$

Equation (29) is the energy constant in the 2BP. In this equation C is a constant and $\dot{\mathbf{r}} \cdot \dot{\mathbf{r}} = v^2$. Equation (29) shows that the orbital energy per unit mass of a body is a conserved quantity.

2.2.2 The Angular Momentum Constant

The next constant in the problem is the angular momentum constant. The approach we take to derive the momentum constant is similar to the method used to derive the energy constant. We begin by taking the cross product of $\mathbf{r}$ with the equation of motion (23)

$$\mathbf{r} \times \left(\ddot{\mathbf{r}} + \frac{\mu}{r^3}\mathbf{r}\right) = 0, \tag{30}$$

or,

$$\mathbf{r} \times \ddot{\mathbf{r}} + \left(\frac{\mu}{r^3}\right)\mathbf{r} \times \mathbf{r} = 0. \tag{31}$$

When two vectors are in the same direction, the angle between them is zero and the magnitude of the cross product will be zero. So the second term on the right hand side of equation (31) is zero ($\mathbf{r} \times \mathbf{r} = 0$). The left hand term of equation (31) can be re-written by taking the anti-derivative of the acceleration vector. We obtain the expression:

$$\mathbf{r} \times \ddot{\mathbf{r}} = \frac{d}{dt}(\mathbf{r} \times \dot{\mathbf{r}}). \tag{32}$$

We expand equation (32) by applying the time derivative crossed with the position and velocity vectors

$$\frac{d}{dt}(\mathbf{r} \times \dot{\mathbf{r}}) = \dot{\mathbf{r}} \times \dot{\mathbf{r}} + \mathbf{r} \times \ddot{\mathbf{r}}. \tag{33}$$

The cross product, $\dot{\mathbf{r}} \times \dot{\mathbf{r}} = 0$, and we can simplify further

$$\mathbf{r} \times \ddot{\mathbf{r}} = \frac{d}{dt}(\mathbf{r} \times \dot{\mathbf{r}}) = 0. \tag{34}$$

We integrate equation (34) with respect to time to obtain the angular momentum integral

$$\mathbf{r} \times \dot{\mathbf{r}} = \mathbf{h}. \tag{35}$$

Equation (35) is the angular momentum constant in the 2BP. In this equation, $\mathbf{h}$ is a constant.

The motion of m_2 around m_1 lies in a plane perpendicular to the direction of $\mathbf{h}$. See Figure 2 below. This implies the position and velocity of the planet will always lie the same orbital plane and the motion is confined to lie in this plane only. We have found four constants of motion for the 2BP: the energy integral (C) and the three components (h_x, h_y, h_z) of the specific angular momentum ($\mathbf{h}$).

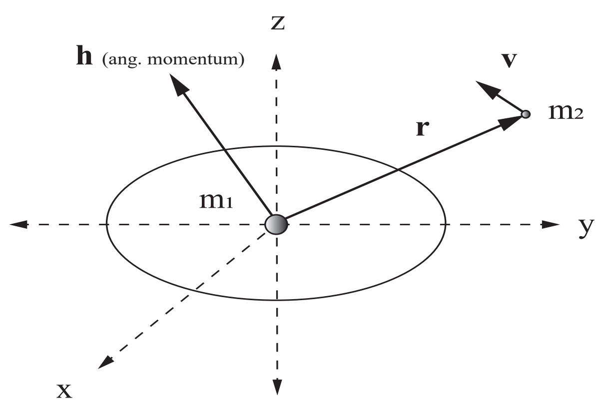

Fig. 2. The smaller mass, m_2, in motion around m_1, in the x-y plane. The angular momentum vector is always perpendicular to the orbital plane in the z-direction.

2.2.3 The Laplace-Runge-Lenz Vector

The last constant in the problem is the Laplace-Runge-Lenz vector, or the eccentricity vector, $\mathbf{B}$. We begin by taking the cross product between the angular momentum vector and the equation of motion (equation 23),

$$\mathbf{h} \times \left(\ddot{\mathbf{r}} + \frac{\mu}{r^3}\mathbf{r} \right) = 0. \tag{36}$$

Expanding, we see that

$$\mathbf{h} \times \ddot{\mathbf{r}} + \left(\frac{\mu}{r^3} \right) \mathbf{h} \times \mathbf{r} = 0. \tag{37}$$

The left term of equation (37) is re-written by taking the anti-derivative of the acceleration

$$\mathbf{h} \times \ddot{\mathbf{r}} = \frac{d}{dt}(\mathbf{h} \times \dot{\mathbf{r}}). \tag{38}$$

The definition of angular momentum, $\mathbf{h} = \mathbf{r} \times \dot{\mathbf{r}}$, is used in the right term of equation (37),

$$\left(\frac{\mu}{r^3}\right)\mathbf{h} \times \mathbf{r} = \frac{\mu}{r^3}(\mathbf{r} \times \dot{\mathbf{r}}) \times \mathbf{r} = 0. \tag{39}$$

Using the vector triple product rule,

$$(\mathbf{A} \times \mathbf{B}) \times \mathbf{C} = -\mathbf{C} \times (\mathbf{A} \times \mathbf{B}) = -(\mathbf{C} \cdot \mathbf{B})\mathbf{A} + (\mathbf{C} \cdot \mathbf{A})\mathbf{B}. \tag{40}$$

We can re-write equation (39) as

$$\frac{\mu}{r^3}\Big[-(\mathbf{r} \cdot \dot{\mathbf{r}})\mathbf{r} + (\mathbf{r} \cdot \mathbf{r})\dot{\mathbf{r}}\Big] = 0. \tag{41}$$

The dot product of the position and velocity vector is zero ($\mathbf{r} \cdot \dot{\mathbf{r}} = 0$) because $\cos\frac{\pi}{2} = 0$. These vectors in circular motion under a central force is zero because the vectors are always perpendicular to each other. The equation becomes

$$\frac{\mu}{r^3}(\mathbf{r} \cdot \mathbf{r})\dot{\mathbf{r}} = \frac{\mu}{r^3} r\dot{\mathbf{r}} = \frac{\mu}{r^2}\dot{\mathbf{r}} = 0. \tag{42}$$

We obtain the equation and the anti-derivative of the velocity,

$$\frac{\mu}{r^2}\dot{\mathbf{r}} = \frac{d}{dt}\left(\frac{\mu}{r}\mathbf{r}\right). \tag{43}$$

We can combine the results from equations (38) and (43).

$$\begin{aligned} \mathbf{h} \times \ddot{\mathbf{r}} + \frac{\mu}{r^3}(\mathbf{r} \times \dot{\mathbf{r}}) \times \mathbf{r} &= \mathbf{h} \times \ddot{\mathbf{r}} + \frac{\mu}{r^3}\Big[(\mathbf{r} \cdot \mathbf{r})\dot{\mathbf{r}} - (\dot{\mathbf{r}} \cdot \mathbf{r})\mathbf{r}\Big], \\ &= \frac{d}{dt}(\mathbf{h} \times \dot{\mathbf{r}}) + \frac{d}{dt}\left(\frac{\mu}{r}\mathbf{r}\right), \\ &= \frac{d}{dt}\left(\mathbf{h} \times \dot{\mathbf{r}} + \frac{\mu}{r}\mathbf{r}\right). \end{aligned} \tag{44}$$

Equation (44) is in a form that we can integrate with respect to time

$$\frac{d}{dt}\left(\mathbf{h}\times\dot{\mathbf{r}}+\frac{\mu}{r}\mathbf{r}\right)=0. \tag{45}$$

After integrating the equation (45), we obtain last constant in the 2BP

$$\mathbf{B}=\mathbf{h}\times\dot{\mathbf{r}}+\frac{\mu}{r}\mathbf{r}. \tag{46}$$

This is the Laplace-Runge-Lenz vector.

The eccentricity vector is a dimensionless vector that describes the shape and orientation of the orbit of one astronomical body around the other. This vector is the last constant of motion and the eccentricity vector is a constant quantity wherever the planet is in the orbit. In the 2BP, if we assume the planet is in a circular orbit around the central mass, the eccentricity is zero $e=0$.

2.3 Results from the Two-body Problem

We created a program to simulate each of the planet's orbits in the solar system around the Sun over 165 Earth years, or approximately the time it takes for Neptune to orbit the Sun once. The program generates the positions, velocities, orbital energies, and angular momenta of each of the planets and plots the data. Using the equation of motion (23) from the 2BP, the planet's accelerations are determined from the gravitational force of the Sun on the planet. The velocities and the positions of the planet are determined by solving a set of second-order differential equations. In Figure 3 below, we plot the x and y positions of the inner and outer planets with the Sun at the center (red dot). In the left image, the orbits of Mercury (black), Venus (yellow), Earth (blue), and Mars (red) are plotted. In the image on the right, the orbits of Jupiter (green), Saturn (cyan), Uranus (pink), and Neptune (black) are plotted. In each of the cases, the eight planets in the 2BP are assumed to be circular orbits revolving around the Sun. This means we can take any mass, up to the mass of Jupiter, and simulate the trajectory of the body, and we will obtain a circular orbit.

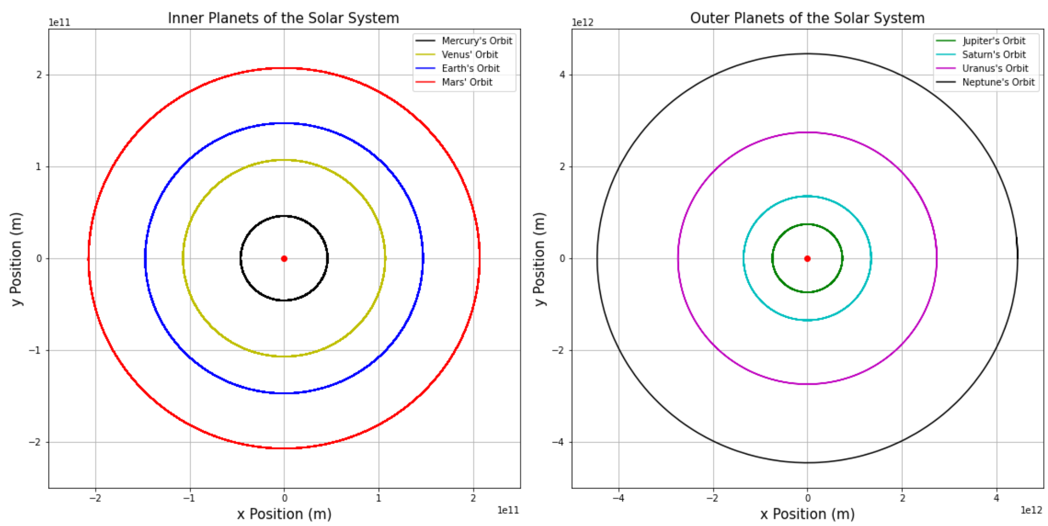

Fig. 3. The orbits are shown for the inner rocky planets: Mercury's, Venus', Earth's, and Mars' orbit are shown (on the left), and the outer gas planets: Jupiter's, Saturn's, Uranus', and Neptune's orbit (on the right) in the 2BP. The simulation was run for 165 Earth years and each of the planets display circular orbits.

We also use our program to calculate the orbital energy and angular momentum of Earth in the 2BP. In Figure 4, the Earth's energy and angular momentum are plotted over time. In the figure, we see that the values remain constant and do not change over the course of the simulation. In the previous sections, we derived the equations (29) and (35) and showed that the energy and momentum integrals were conserved quantities in the 2BP. Using these results, we also confirm that our program runs successfully.

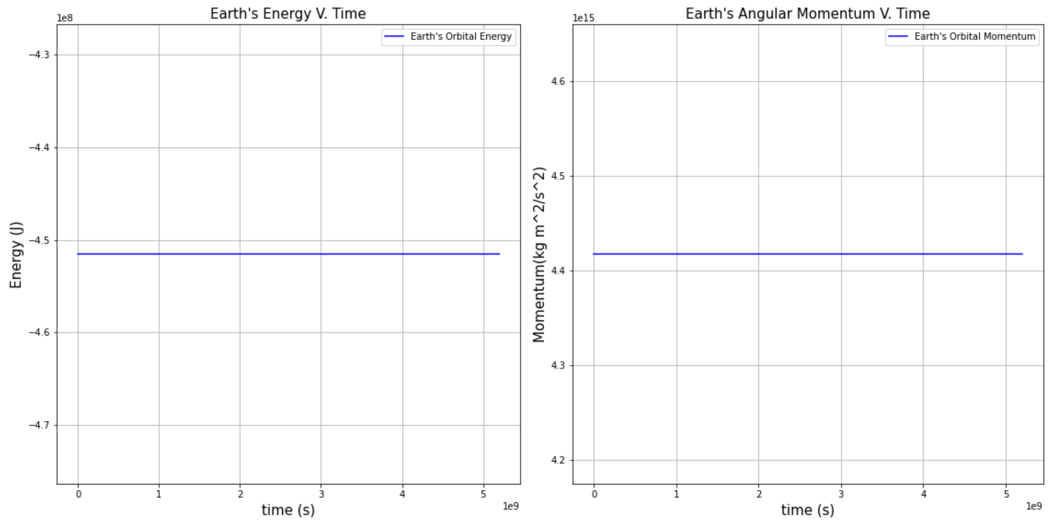

Fig. 4. Earth's orbital energy and angular momentum remain constant over the entire simulation, 165 Earth years, in the 2BP.

See Table 1 for the parameters of each of the planets used in our simulation. In the table we included the initial positions and orbital velocities of each of the eight bodies.

Table 1. The initial conditions for each of the eight planets in the 2BP. Each of the planets begin at the perihelion position with a velocity in the *y* direction.

Initial Conditions				
Planet	$x_0(10^9 m)$	$y_0(10^9 m)$	$v_{x_0}(m/s)$	$v_{y_0}(m/s)$
Mercury	46.0	0.0	0.0	47,400
Venus	107.5	0.0	0.0	35,000
Earth	147.1	0.0	0.0	29,000
Mars	206.6	0.0	0.0	24,100
Jupiter	740.9	0.0	0.0	13,100
Saturn	1352.6	0.0	0.0	9,700
Uranus	2741.3	0.0	0.0	6,800
Neptune	4444.5	0.0	0.0	5,400

2.4 The Stability Condition

We will derive the stability criterion for the 2BP using methods similar to Hamill (2010). Suppose a small perturbing force acts on a planet orbiting the Sun. The perturbing force will cause the planet to oscillate around the equilibrium position. When the force becomes large enough, the planet can become unstable around the equilibrium position and the orbit is no longer circular. In the 2BP, the central force is a gravitational force with an inverse-square dependence on distance. For the sake of this analysis we assumed that the central force can take on different kinds of radial dependence and in the subsequent discussion, the force will be general until specifically defined. Suppose the force is a function of the radial position only

$$\mathbf{F}(\mathbf{r}) = F(r)\hat{\mathbf{r}}. \tag{47}$$

The equation of motion can be transformed from Cartesian coordinates into polar coordinates, where the scalar equations are

$$\begin{aligned} F(r) &= m(\ddot{r} - r\dot{\theta}^2), \\ 0 &= m(r\ddot{\theta} + 2\dot{r}\dot{\theta}). \end{aligned} \tag{48}$$

Here $\ddot{r}$ is the radial acceleration, $\dot{r}$ is the radial velocity, $\dot{\theta}$ is the angular velocity, and $\ddot{\theta}$ is the angular acceleration. Using the definition of the angular momentum $\mathbf{l} = \mathbf{r} \times \mathbf{p}$ and applying it to a body in motion around another body in a circular path, the following scalar relation can be obtained

$$l = mvr = mr^2\dot{\theta}, \tag{49}$$

where

$$\dot{\theta} = \frac{l}{mr^2}. \tag{50}$$

Using this relation the equation of motion can be reduced to a one-dimensional equation where the dependence is only on distance, r,

$$\begin{gathered} m\left[\ddot{r} - r\left(\frac{l}{mr^2}\right)^2\right] = m\left[\ddot{r} - \frac{l^2}{m^2 r^3}\right] = F(r), \\ m\ddot{r} = F(r) + \frac{l^2}{m^2 r^3}. \end{gathered} \tag{51}$$

The orbit of a planet remains at a constant radius, a, over time. Since a is a constant, the first and second time derivatives with respect to the radius will be equal to zero.

$$r = a = \text{constant} \Rightarrow \dot{r} = \ddot{r} = 0.$$

At radius a, the gravitational force on a planet in a circular orbit is

$$F(r = a) = -\frac{l^2}{ma^3}. \tag{52}$$

Suppose the planet is given a slight nudge from its original trajectory, given by a perturbing force. The planet's radial distance, r, from the central body will be changed by ε where $\varepsilon << a$

$$r = a + \varepsilon. \tag{53}$$

We substitute the approximate radial distance into the equation of motion

$$m\frac{d^2}{dt^2}(a + \varepsilon) = F(a + \varepsilon) + \frac{l^2}{m(a + \varepsilon)^3}. \tag{54}$$

Before we move on, we briefly mention the Taylor series and Binomial expansions. Using these approximations, our equation of motion can be simplified. The approximations we will use in equation (54) are given below

1) Taylor's Series

$$f(a + \Delta x) = f(a) + \Delta x \frac{df(a)}{dx} + \frac{1}{2!}\Delta x^2 \frac{d^2 f(a)}{dx^2} + \ldots,$$

2) Binomial Expansion

$$(1+x)^m = 1 + mx + \frac{m(m-1)}{2!}x^2 + ...$$

Applying these expansions to equation (54), we obtain the resulting expansions

$$f(a+\varepsilon) = f(a) + \varepsilon\frac{df}{dr} + \frac{1}{2!}\varepsilon^2\frac{df^2}{dr^2} + ..., \tag{55}$$

and

$$\frac{1}{(1+\varepsilon/a)^3} = (1+\varepsilon/a)^{-3} = 1 - 3\frac{\varepsilon}{a} + 6\left(\frac{\varepsilon}{a}\right)^2 + ... \tag{56}$$

We re-state the equation of motion and substitute the approximations to obtain the expanded equation of motion,

$$m\ddot{\varepsilon} = F(a+\varepsilon) + \frac{l^2}{ma^3(1+\varepsilon/a)^3}. \tag{57}$$

Applying the expansions, we obtain the following equation

$$\begin{aligned} m\ddot{\varepsilon} &= F(a) + \varepsilon\frac{dF}{dr} + \frac{1}{2!}\varepsilon^2\frac{dF^2}{dr^2} + ... \\ &= \frac{l^2}{mr^3} - 3\frac{l^2}{mr^3}\frac{\varepsilon}{a} + 6\frac{l^2}{mr^3}\left(\frac{\varepsilon}{a}\right)^2 + ... \end{aligned} \tag{58}$$

For our purpose, it is sufficient to neglect the second and higher-order terms

$$m\ddot{\varepsilon} = F(a) + \varepsilon\frac{dF}{dr} + \frac{l^2}{ma^3} - 3\frac{l^2}{ma^3}\frac{\varepsilon}{a}. \tag{59}$$

Using equation (52) with equation (59), we can simplify the expression and find terms that cancel out to yield

$$\ddot{\varepsilon} = \varepsilon\left[\frac{1}{m}\frac{dF}{dr} - 3\frac{l^2}{m^2a^4}\right]. \tag{60}$$

We arrive at an expression identical to the equation that represents simple harmonic motion

$$\ddot{\varepsilon} + \varepsilon\Gamma = 0, \tag{61}$$

where $\Gamma = -3\frac{l^2}{m^2a^4} + \frac{1}{m}\frac{dF}{dr}$.

When $\Gamma < 0$, the general solution has the form dependent on an exponential

$$\varepsilon(t) = \alpha e^{+\sqrt{|\Gamma|}t} + \beta e^{-\sqrt{|\Gamma|}t}. \tag{62}$$

When $\Gamma > 0$, the general solution has the form dependent on a linear combination of sines and cosines

$$\varepsilon(t) = \gamma \sin(\sqrt{\Gamma}t) + \eta \cos(\sqrt{\Gamma}t). \tag{63}$$

According to equation (62) the general solution depends on the behavior of the exponential. As the time increases, the term with the negative exponential will go to zero leaving a solution with the form of a positive exponential. The orbit of a planet will be unstable over time and the distance $r = a + \varepsilon$ will grow without any constraints. On the other hand, if $\Gamma > 0$, the solution involves the behavior of the sine and cosine. The solution will oscillate about the equilibrium radius position $(r = a)$. Consequently, the stability condition for the orbit is

$$0 < 3\frac{l^2}{m^2a^4} - \frac{1}{m}\frac{dF}{dr}. \tag{64}$$

Let us consider an example assuming the central force is inversely proportional to r^2, that is, the form of the gravitational force in the 2BP. Taking the derivative of the force with respect to the radial position, we obtain

$$F(r) = -G\frac{Mm}{r^2} \Rightarrow \frac{dF(r)}{dr} = 2G\frac{Mm}{r^3}. \tag{65}$$

Evaluating equation (65) for $r = a$, the stability condition becomes

$$-\frac{3}{a}F(a) - \frac{dF}{dr} = \frac{3}{a}\left(\frac{GMm}{a^2}\right) - \frac{2GMm}{a^3} = \frac{GMm}{a^3}, \tag{66}$$

so $\Gamma > 0$ and the orbit is stable. That is, the force of gravity leads a stable planetary orbit. Generally, the central force can be extended to any power of r

$$F(r) = -\frac{k}{r^m} \Rightarrow \frac{dF}{dr} = \frac{km}{a^{m+1}}, \tag{67}$$

where k is a constant. Again, using equation (65), we arrive at the stability condition

$$\begin{gathered} -\frac{3}{a}\left(-\frac{k}{a^m}\right) - \frac{km}{a^{m+1}} = \frac{k}{a^{m+1}(3-m)} < 0, \\ m < 3. \end{gathered} \tag{68}$$

The derivation demonstrates there cannot be anything higher than an inverse square law in terms of the stability. The inverse square law leads to stable orbits in the solar system. Anything higher in order, will result in the planet spiraling away from its original orbit.

3 THE THREE-BODY PROBLEM

In this chapter we provide an introduction to the three-body problem (3BP) and derive the disturbing function. We also apply a special perturbation method to show how another body in the system influences the orbit of Earth. Lastly, we briefly compare the 2BP and 3BP results.

3.1 Three-body Problem Equations

In the N-body system, with mutually interacting bodies, the acceleration of a body of interest, i, can be expressed as:

$$\ddot{\mathbf{r}}_i = \sum_{j=1}^{N} \frac{Gm_j(\mathbf{r}_j - \mathbf{r}_i)}{r_{ij}^3}, \tag{69}$$

where $\ddot{\mathbf{r}}_i$ is the acceleration vector of a body i, m_j is the mass of body j, $\mathbf{r}_i$ and $\mathbf{r}_j$ are the position vectors, and r_{ij} is the distance between the mass i and j. For the purpose of this chapter, let $N = 3$ and $i = 1$, representing the acceleration of the body of interest. Then, the acceleration of body 1 due to the other masses in the system, m_2 and m_3, can be expressed as:

$$\begin{aligned} \ddot{\mathbf{r}}_1 &= \sum_{j=2}^{N=3} \frac{Gm_j(\mathbf{r}_j - \mathbf{r}_1)}{r_{1j}^3}, \\ \ddot{\mathbf{r}}_1 &= \frac{Gm_2(\mathbf{r}_2 - \mathbf{r}_1)}{r_{12}^3} + \frac{Gm_3(\mathbf{r}_3 - \mathbf{r}_1)}{r_{13}^3}. \end{aligned} \tag{70}$$

3.2 The Disturbing Function

In the following derivation of the disturbing function, we use methods similar to those found in Murray and Dermott (1999). Suppose an inner mass, m_i, exists in an elliptical orbit around a central mass, m_c. When another mass is introduced into the system, m_j, the system is composed of three bodies. This is the three-body problem. The extra body in the system creates additional gravitational forces between each of the bodies. The disturbing function is used to make the problem easier to analyze and mainly used to look at deviations in the orbital elements.

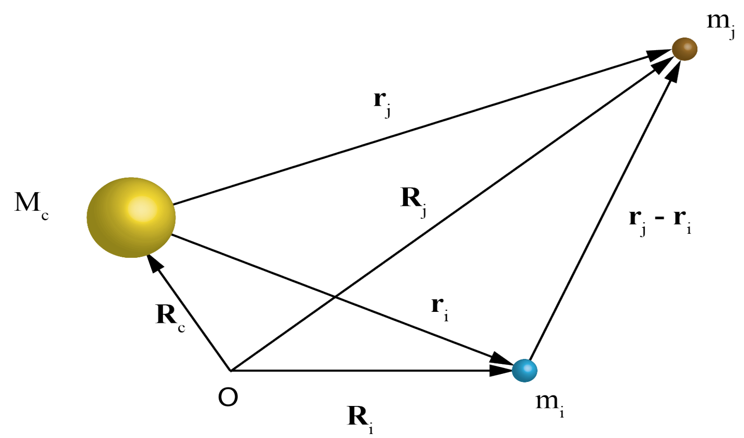

Fig. 5. The position vectors $\mathbf{r}_i$ and $\mathbf{r}_j$ of the masses m_i and m_j, with respect to the central mass, M_c. $\mathbf{R}_c, \mathbf{R}_i$, and $\mathbf{R}_j$ are the position vectors with respect to an arbitrary, fixed origin.

The position vectors $(\mathbf{R_c}, \mathbf{R_i}, \mathbf{R_j})$, give the location of the masses (M_c, m_i, m_j) relative to a fixed origin. Assume $\mathbf{r}_i$ and $\mathbf{r}_j$ are the position vectors of the secondary masses relative to the primary. See Figure 5. By definition, the norms of $\mathbf{r}_i$, $\mathbf{r}_j$, and $\mathbf{r}_j - \mathbf{r}_i$ are

$$\begin{aligned} |\mathbf{r}_i| &= \sqrt{x_i^2 + y_i^2 + z_i^2}, \\ |\mathbf{r}_j| &= \sqrt{x_j^2 + y_j^2 + z_j^2}, \\ |\mathbf{r}_j - \mathbf{r}_i| &= \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2 + (z_j - z_i)^2}. \end{aligned} \tag{71}$$

Using Newton's laws of motion and the law of universal gravitation, the equations of motion for each of the masses in the inertial reference frame are

$$m_c \ddot{\mathbf{R}}_c = G m_c m_i \frac{\mathbf{r}_i}{r_i^3} + G m_c m_j \frac{\mathbf{r}_j}{r_j^3}, \tag{72}$$

$$m_i \ddot{\mathbf{R}}_i = G m_i m_j \frac{\mathbf{r}_j - \mathbf{r}_i}{|\mathbf{r}_j - \mathbf{r}_i|^3} - G m_i m_c \frac{\mathbf{r}_i}{r_i^3}, \tag{73}$$

$$m_j\ddot{\mathbf{R}}_j = Gm_jm_i\frac{\mathbf{r}_i-\mathbf{r}_j}{|\mathbf{r}_i-\mathbf{r}_j|^3} - Gm_jm_c\frac{\mathbf{r}_j}{r_j^3}. \tag{74}$$

The acceleration of m_i and m_j relative to the primary can be expressed as

$$\begin{aligned} \ddot{\mathbf{r}}_i &= \ddot{\mathbf{R}}_i - \ddot{\mathbf{R}}_c, \\ \ddot{\mathbf{r}}_j &= \ddot{\mathbf{R}}_j - \ddot{\mathbf{R}}_c. \end{aligned} \tag{75}$$

By using the equations (73), (74), and (75), we obtain the following set of equations

$$\begin{aligned} \ddot{\mathbf{r}}_i + G(m_c+m_i)\frac{\mathbf{r}_i}{r_i^3} &= Gm_j\Big[\frac{\mathbf{r}_j-\mathbf{r}_i}{|\mathbf{r}_i-\mathbf{r}_j|^3} - \frac{\mathbf{r}_j}{r_j^3}\Big], \\ \ddot{\mathbf{r}}_j + G(m_c+m_j)\frac{\mathbf{r}_j}{r_j^3} &= Gm_i\Big[\frac{\mathbf{r}_i-\mathbf{r}_j}{|\mathbf{r}_i-\mathbf{r}_j|^3} - \frac{\mathbf{r}_i}{r_i^3}\Big], \end{aligned} \tag{76}$$

which can be re-written as the gradient of a scalar function

$$\begin{aligned} \ddot{\mathbf{r}}_i &= \nabla_i\cdot(\mathscr{U}_i+\mathscr{R}_i) = \Big(\hat{\mathbf{i}}\frac{\partial}{\partial x_i} + \hat{\mathbf{j}}\frac{\partial}{\partial y_i} + \hat{\mathbf{k}}\frac{\partial}{\partial z_i}\Big)(\mathscr{U}_i+\mathscr{R}_i), \\ \ddot{\mathbf{r}}_j &= \nabla_j\cdot(\mathscr{U}_j+\mathscr{R}_j) = \Big(\hat{\mathbf{i}}\frac{\partial}{\partial x_j} + \hat{\mathbf{j}}\frac{\partial}{\partial y_j} + \hat{\mathbf{k}}\frac{\partial}{\partial z_j}\Big)(\mathscr{U}_j+\mathscr{R}_j), \end{aligned} \tag{77}$$

where

$$\begin{aligned} \mathscr{U}_i &= G\frac{(m_c+m_i)}{r_i}, \\ \mathscr{U}_j &= G\frac{(m_c+m_j)}{r_j}, \end{aligned} \tag{78}$$

and

$$\begin{aligned} \mathscr{R}_i &= \frac{Gm_j}{|\mathbf{r}_j-\mathbf{r}_i|} - Gm_j\frac{\mathbf{r}_i\cdot\mathbf{r}_j}{r_j^3}, \\ \mathscr{R}_j &= \frac{Gm_i}{|\mathbf{r}_i-\mathbf{r}_j|} - Gm_i\frac{\mathbf{r}_j\cdot\mathbf{r}_i}{r_i^3}. \end{aligned} \tag{79}$$

The term $\mathscr{R}$ is the disturbing function which arises from the potential between the secondary masses m_i and m_j.

3.3 Cowell's Perturbation Method

Cowell's method is a perturbation method applied to the 3BP. In Figure 6, we illustrate two bodies with mass, m_i and m_j, revolving around the central mass, M_c. We have seen in the 2BP, m_i follows an orbit around the central body. When another body is considered in the system, such as m_j, we need to account for the gravitational influence it has on m_i. In the figure, we indicate that m_i deviates from its initial path due to this attraction. We can call this deviation from its initial path a perturbation.

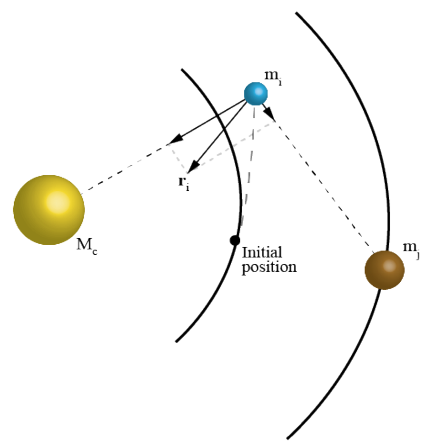

Fig. 6. Cowell's perturbation method. A simple perturbation applied to the bodies, M_c, m_j, and m_i. The perturbation changes the orbit of m_i, showing a deviation from its initial path.

We begin with the equations of motion of the 2BP with a perturbing force as a function of distance. The equation of motion with a perturbation force can be written as

$$\ddot{\mathbf{r}} + \frac{\mu}{r^3}\mathbf{r} = \mathbf{F}(\mathbf{r}), \tag{80}$$

where $\mathbf{F}(\mathbf{r})$ is the perturbing force. For numerical purposes, equation (80) is reduced to a set of differential equations

$$\begin{aligned} \dot{\mathbf{r}} &= \frac{d\mathbf{r}}{dt}, \\ \ddot{\mathbf{r}} &= \mathbf{F}(\mathbf{r}) - \frac{\mu}{r^3}\mathbf{r}, \end{aligned} \tag{81}$$

where $\dot{\mathbf{r}}$ and $\mathbf{r}$ are the velocity and the radius of the perturbed body with respect to the Sun. In vector components, equation (81) becomes

$$\begin{aligned}
\dot{x} &= v_x \qquad \ddot{x} = F(r)_x - \frac{\mu}{r^3}x, \\
\dot{y} &= v_y \qquad \ddot{y} = F(r)_y - \frac{\mu}{r^3}y, \\
\dot{z} &= v_z \qquad \ddot{z} = F(r)_z - \frac{\mu}{r^3}z,
\end{aligned} \tag{82}$$

where $r = \sqrt{x^2 + y^2 + z^2}$.

As an example, we will consider how another planet, such as Jupiter, perturbs the Earth in the 2BP. We will label m_j as Jupiter, treated as the perturbing body, and Earth is the perturbed body, m_i. The equations of motion for the Earth can be written in terms of the x,y, and z components below

$$\begin{aligned}
\dot{x} &= v_x \qquad \ddot{x} = -\eta \left[\frac{x_{SE}}{r_{SE}^3} - \frac{x_{JE}}{r_{JE}}\right] - \frac{\mu}{r^3}x, \\
\dot{y} &= v_y \qquad \ddot{y} = -\eta \left[\frac{y_{SE}}{r_{SE}^3} - \frac{y_{JE}}{r_{JE}}\right] - \frac{\mu}{r^3}y, \\
\dot{z} &= v_z \qquad \ddot{z} = -\eta \left[\frac{z_{SE}}{r_{SE}^3} - \frac{z_{JE}}{r_{JE}}\right] - \frac{\mu}{r^3}z,
\end{aligned} \tag{83}$$

where η is a constant dependent on the masses of the Sun and Jupiter.

The results from Cowell's method are highlighted in Figure 7 below. The figure shows the orbits of Earth and Jupiter using the equations of motion when the perturbing force exists and when it is absent. In the image on the left, we can observe a deviation of Earth's orbit when Jupiter is included (red) in the system and when Jupiter is ignored (blue). In the image on the right, we can observe that Jupiter's orbit is not effected by the gravitational influence from Earth. As seen in the figure, the blue and red curves are overlap.

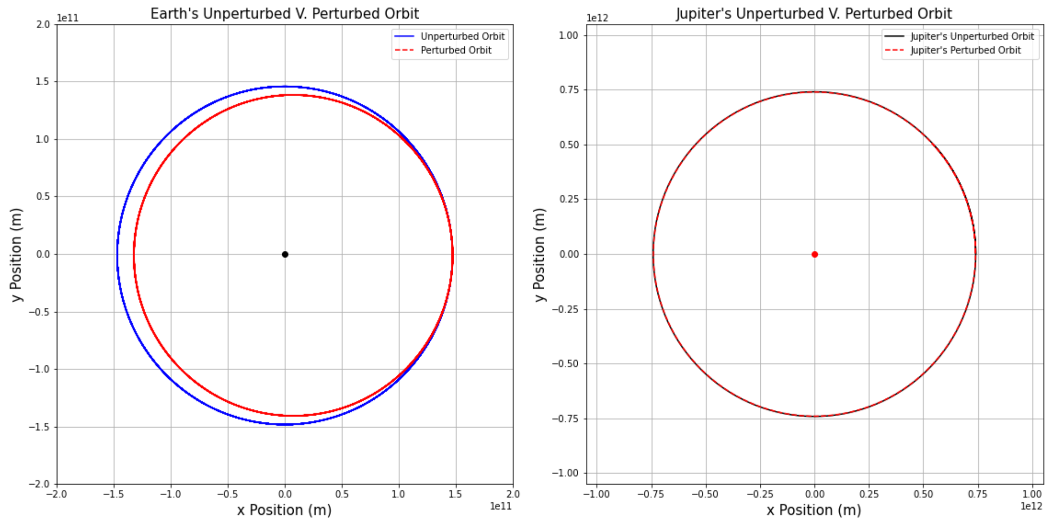

Fig. 7. The image on the left displays Earth's orbit in the unperturbed and perturbed case. The image on the right displays Jupiter's orbit in the unperturbed and perturbed case. Cowell's method is used for the simulation. The blue line represents the initial orbit of Earth in the two-body problem, red presents the perturbation due to Jupiter. There is no effect on the orbit of Jupiter from the Earth as shown in the image on the right.

In general, there are other forces to consider when the number of bodies in the system increase. These will be mutual gravitational forces between the central, larger body and a planet and a mutual force between the planets. The behavior of such a system is largely dependent on the masses and the distances between these masses.

3.4 Two-body vs. Three-body Problem

We will discuss the conserved constants mentioned before. We will limit our the analysis to the energy and angular momentum of the Earth in the 2BP and how these quantities change when another planet is included in the system. Here, we introduce Jupiter into the system. Jupiter is chosen because after the Sun, it is the next most influential body in the solar system, having a mass on the scale of 10^{27} kg. Here we will compare the results of the 2BP versus 3BP.

In the Figures 8 and 9 below, the specific energies and angular momenta as a function of time are plotted. The simulation was run for 165 Earth years, or approximately the time it takes Neptune to orbit the Sun once. As shown in Figure 8, the orbital energy of Earth (green line) in the 2BP remains constant over the entire period. The value of this energy is $C = -4.51 \times 10^8$ Jkg^{-1}. However, when we include Jupiter in the system, the energy of the Earth fluctuates between the values of $-4.75 \times 10^8 < C < -4.65 \times 10^8$.

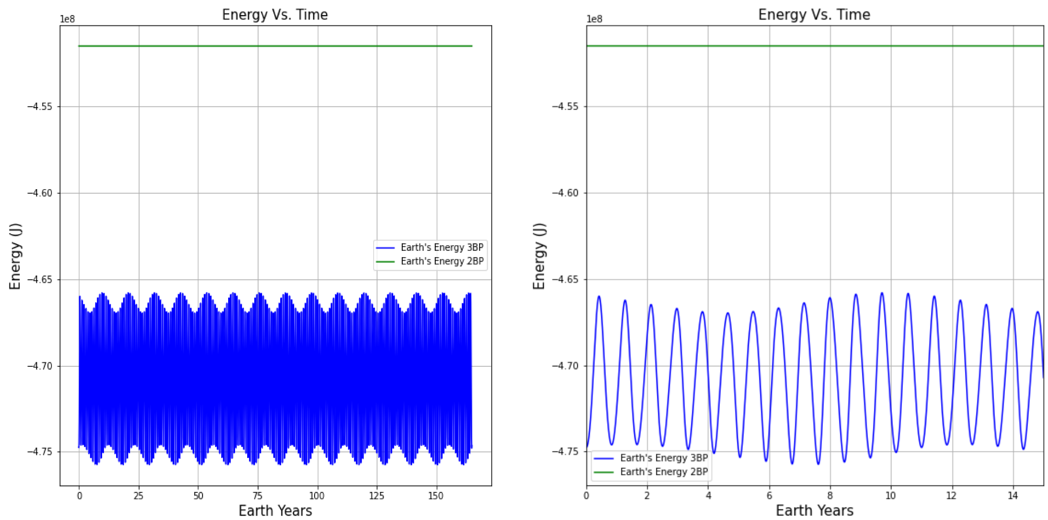

Fig. 8. The image on the left displays the specific energies of Earth in the 2BP and 3BP plotted over 165 Earth years. The blue line represents the specific energy of Earth in the 3BP and the green line represents the specific energy of the Earth in the 2BP. The specific energy in the 2BP remains constant and the specific energy in the 3BP varies. The image on the right is an expanded view of the image on the left, only covering 15 years time. This shows the oscillations in energy quite clearly.

In Figure 9 the specific angular momentum is plotted over time in the 2BP and 3BP. The simulation was run over the same period of time as Figure 8. From the plot, we can see the specific angular momentum value in the 2BP is around $|\mathbf{h}| = 4.42 \times 10^{15}$ kgm^{2}s^{-2} and remains constant over the entire simulation. For the 3BP, the angular momentum varies between $4.30 \times 10^{15} < |\mathbf{h}| < 4.342 \times 10^{15}$.

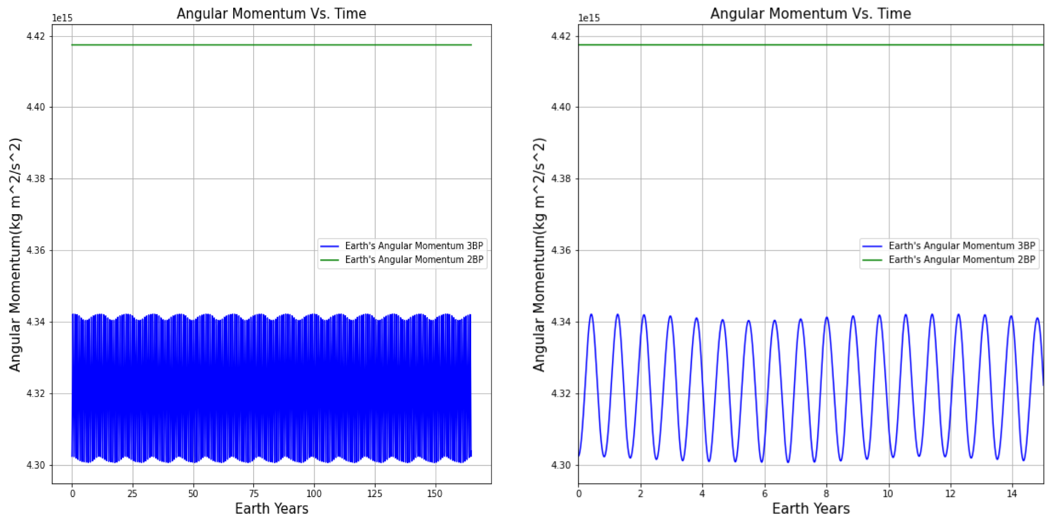

Fig. 9. The image on the left displays the specific momenta of Earth in the 2BP and 3BP plotted over 165 Earth years. The blue line represents the specific momentum of Earth in the 3BP and the green line represents the specific momentum of the Earth in the 2BP. The specific momentum in the 2BP remains constant and the specific momentum in the 3BP varies. The image on the right is an expanded view of the image on the left, only covering 15 years time. This shows the oscillations in angular momentum quite clearly.

In the 3BP, the positions and velocities of the three masses are used to determine the acceleration using Newton's laws. This type of system has no closed-form solution and numerical methods are required to analyze the motion. The problem can be simplified by restricting the mass of interest, moving under the influence of the other two larger masses. The mass of interest is negligible which means the effect that this planet has on the other two larger masses is too small to matter, which we have shown. This type of system can be analyzed using the approach used to study the 2BP where the bodies are moving around the center of mass. Notice the 2BP and 3BP are special cases of the much harder problem, the N-body problem. The additional planetary bodies create mutual gravitational forces between each other, the Sun-Planet interactions and the Planet-Planet interactions, and over a long period of time, lead to unpredictable motion. In this thesis, we will look at the three-body problem containing the Sun-Earth-Jupiter system and observe how the orbits change over time due to these gravitational interactions. We will treat Jupiter as our second star in the system but before we do this, we need to discuss the numerical methods.

4 NUMERICAL METHODS

In this chapter we present two numerical methods used to compute the orbital elements for the 2BP and 3BP. The first method we used is called the Euler-Cromer method and the second is the Runge-Kutta 4^{th} order method. Both methods are used numerically to obtain a solution for a system of ordinary differential equations (ODEs). The type of ODEs that come up in celestial mechanics are second order equations and the best way to handle them is to break them down into a set of two first order equations. The type of accuracy in the solution depends on the numerical method. In this thesis, we compare the Euler-Cromer and Runge-Kutta Method and justify why the Euler-Cromer method is sufficient for our computation. We also outline the algorithm used to obtain a solution of the differential equations presented in Chapter 2.

4.1 Euler-Cromer Method

The Euler-Cromer method is designed to integrate differential equations given an initial set of conditions. This method is used to approximate the solution to an initial-value problem when the differential equation cannot be solved analytically. We begin by stating the general form from the Euler method

$$y_{n+1} = y_n + h\frac{dy}{dx}, \tag{84}$$

which advances the solution from y_n to y_{n+1}. This formula uses the derivative at the beginning of the interval and advances the solution through a step size h. The step size error is one power smaller than the correction $\mathscr{O}(h^2)$. The Euler method is not recommended due to the instability of the solutions. In this thesis, we choose to use the Euler-Cromer and Runge-Kutta 4^{th} order methods because these methods are stable and yield highly accurate solutions.

The Euler-Cromer method can be applied to a pair of differential equations of the form

$$\begin{aligned} \dot{x} &= f(t,x), \\ \ddot{x} &= g(t,x), \end{aligned} \tag{85}$$

where f and g are given functions and x, $\dot{x}$ may be scalars or vectors. The differential equations are solved with the initial conditions $x(t_0) = x_0$ and $\dot{x}(t_0) = \dot{x}_0$. The method produces approximate discrete solutions by iterating

$$\begin{aligned} \dot{x}_{n+1} &= \dot{x}_n + g(t_n, x_n)\Delta t, \\ x_{n+1} &= x_n + f(t_n, \dot{x}_n)\Delta t, \end{aligned} \tag{86}$$

where Δt is the time step and $t_n = t_n + n\Delta t$ is the time after n steps. The functions f depend on the time and position at the n^{th} step. This method is a first-order approximation, where the error of the computer solution depends on the time step.

The Euler-Cromer method uses v_{n+1} in the equation for x_{n+1} while Euler uses v_n. The Euler-Cromer method is a first-order integrator for which the global error depends the time step. As the time step becomes larger, the global error also becomes larger. Lastly, a consequence of the Euler-Cromer method, is that the energy is conserved. When the Euler method is applied, the energy increases steadily over time making the method unstable.

4.2 Runge-Kutta Method

Just like the Euler-Cromer method, the Runge-Kutta methods are used for iterative methods to approximate solutions of ordinary differential equations. In general, let an initial value problem take the following form

$$\dot{y} = f(t,y) \qquad y(t_0) = y_0, \tag{87}$$

where y is an unknown, scalar or vector, function at a time t. A step-size h is chosen to be non-zero. This method defines the expressions

$$\begin{aligned} y_{n+1} &= y_n + \frac{1}{6}h[k_1 + 2(k_2 + k_3) + k_4], \\ t_{n+1} &= t_n + h, \end{aligned} \tag{88}$$

for $n = 1, 2, ..., N$ using

$$\begin{aligned} k_1 &= f(t_n, y_n), \\ k_2 &= f\Big(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\Big), \\ k_3 &= f\Big(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\Big), \\ k_4 &= f(t_n + h, y_n + hk_3), \end{aligned} \tag{89}$$

where N is the number of steps, k_1 is the slope at the beginning of the interval, k_2 and k_3 are the slopes at the midpoint of the interval, and k_4 is the slope at the end of the interval.

The y_{n+1} term is the 4^{th} order approximation of $y(t_{n+1})$ and the next value y_{n+1} is determined by the present value y_n. The k_i's are the weighted average where each increment is the product of the size of the interval, h, and the estimated slope specified by the function f on the right-hand side of the differential equation. The Runge-Kutta 4^{th} order method gives the local truncation error on the order of $\mathrm{O}(h^5)$, while the total accumulated error is on the order of $\mathrm{O}(h^4)$.

4.3 Method of Computation

In this section, we develop an algorithm for computing the position and velocity of m_i and m_j orbiting around m_C at any time. We consider two algorithms in this section. One presents the method for calculating the accelerations in the 2BP and the second presents the method for calculating the accelerations in the 3BP. For both of the methods, the goal is to obtain the positions $\mathbf{r}_i$ and velocity $\dot{\mathbf{r}}_i$ of the bodies in motion around the central body. The initial positions $\mathbf{r}_0$ and velocities $\dot{\mathbf{r}}_0$ of the bodies are given at an initial time t_0. These methods will determine the $\mathbf{r}$ and $\dot{\mathbf{r}}$ at time t.

4.3.1 Algorithm No.1: 2BP and Runge-Kutta Method

The following method is used to determine the position and velocity of a mass with respect to the central body.

1) Determine the masses in the system, m_C and m_i.
2) Initialize the time t_0, position $\mathbf{r}_0$, and velocity $\dot{\mathbf{r}}$ of m_i.
3) Set the time step size (in seconds, days, years) and the length of the simulation.
4) Compute the energy C and angular momentum $\mathbf{h}$ of m_i using the equations from Chapter 2:

$$\begin{aligned} C &= \frac{1}{2}(\dot{\mathbf{r}}_0 \cdot \dot{\mathbf{r}}_0) - \frac{\mu}{r_0}, \\ \mathbf{h} &= \mathbf{r}_0 \times \dot{\mathbf{r}}_0. \end{aligned} \tag{90}$$

5) Compute the semi-major axis and the eccentricity

$$\begin{aligned} a &= -\frac{\mu}{2C}, \\ e &= 0. \end{aligned} \tag{91}$$

6) Use the Runge-Kutta algorithm.
   a) Compute the coefficients $k_{1,x}$ and $k_{1,y}$

$$\begin{aligned} k_{1,x} &= v_{x,n}, \\ k_{1,y} &= v_{y,n}. \end{aligned} \tag{92}$$

   b) Compute the coefficients k_{1,v_x} and k_{1,v_y}

$$\begin{aligned} k_{1,v_x} &= a_x(x_n, y_n), \\ k_{1,v_y} &= a_y(x_n, y_n). \end{aligned} \tag{93}$$

c) Compute the coefficients $k_{2,x}$ and $k_{2,y}$

$$
\begin{aligned}
k_{2,x} &= v_{x,n} + \frac{1}{2}\Delta t \cdot k_{1,v_x}, \\
k_{2,y} &= v_{y,n} + \frac{1}{2}\Delta t \cdot k_{1,v_y}.
\end{aligned} \tag{94}
$$

d) Compute the coefficients k_{2,v_x} and k_{2,v_y}

$$
\begin{aligned}
k_{2,v_x} &= a_x(x_n + \frac{1}{2}\Delta t \cdot k_{1,x}, y_n + \frac{1}{2}\Delta t \cdot k_{1,y}), \\
k_{2,v_y} &= a_y(x_n + \frac{1}{2}\Delta t \cdot k_{1,x}, y_n + \frac{1}{2}\Delta t \cdot k_{1,y}).
\end{aligned} \tag{95}
$$

e) Compute the coefficients $k_{3,x}$ and $k_{3,y}$

$$
\begin{aligned}
k_{3,x} &= v_{x,n} + \frac{1}{2}\Delta t \cdot k_{2,v_x}, \\
k_{3,y} &= v_{y,n} + \frac{1}{2}\Delta t \cdot k_{2,v_y}.
\end{aligned} \tag{96}
$$

f) Compute the coefficients k_{3,v_x} and k_{3,v_y}

$$
\begin{aligned}
k_{3,v_x} &= a_x(x_n + \frac{1}{2}\Delta t \cdot k_{2,x}, y_n + \frac{1}{2}\Delta t \cdot k_{2,y}), \\
k_{3,v_y} &= a_y(x_n + \frac{1}{2}\Delta t \cdot k_{2,x}, y_n + \frac{1}{2}\Delta t \cdot k_{2,y}).
\end{aligned} \tag{97}
$$

g) Compute the coefficients $k_{4,x}$ and $k_{4,y}$

$$
\begin{aligned}
k_{4,x} &= v_{x,n} + \frac{1}{2}\Delta t \cdot k_{3,v_x}, \\
k_{4,y} &= v_{y,n} + \frac{1}{2}\Delta t \cdot k_{3,v_y}.
\end{aligned} \tag{98}
$$

h) Compute the coefficients k_{4,v_x} and k_{4,v_y}

$$
\begin{aligned}
k_{4,v_x} &= a_x(x_n + \frac{1}{2}\Delta t \cdot k_{3,x}, y_n + \frac{1}{2}\Delta t \cdot k_{3,y}), \\
k_{4,v_y} &= a_y(x_n + \frac{1}{2}\Delta t \cdot k_{3,x}, y_n + \frac{1}{2}\Delta t \cdot k_{3,y}).
\end{aligned} \tag{99}
$$

7) Compute the new positions x_{n+1}, y_{n+1} and velocities $v_{x,n+1}, v_{y,n+1}$

$$\begin{aligned}
x_{n+1} &= x_n + \frac{1}{6}\Delta t \cdot [k_{1,x} + 2(k_{2,x} + k_{3,x}) + k_{4,x}], \\
y_{n+1} &= y_n + \frac{1}{6}\Delta t \cdot [k_{1,y} + 2(k_{2,y} + k_{3,y}) + k_{4,y}], \\
v_{x,n+1} &= v_{x,n} + \frac{1}{6}\Delta t \cdot [k_{1,v_x} + 2(k_{2,v_x} + k_{3,v_x}) + k_{4,v_x}], \\
v_{y,n+1} &= v_{y,n} + \frac{1}{6}\Delta t \cdot [k_{1,v_y} + 2(k_{2,v_y} + k_{3,v_y}) + k_{4,v_y}].
\end{aligned} \tag{100}$$

8) Return to step 4 and repeat until the simulation ends.

Note this is done by using the acceleration function in the program. All the following steps which evaluate a_x and a_y use the same functions, but with different x and y values.

4.3.2 Algorithm No.2: 3BP and Euler-Cromer

1) Determine the masses in the system, m_C, m_i, and m_j.
2) Initialize the time t_0, position $\mathbf{r}_0$, and velocity $\dot{\mathbf{r}}$ of m_i and m_j.
3) Set the time step size (in seconds, days, years) and the length of the simulation.
4) Compute the energy C and angular momentum $\mathbf{h}$ of m_i using the equations from Chapter 2:

$$\begin{aligned}
C &= \frac{1}{2}(\dot{\mathbf{r}}_0 \cdot \dot{\mathbf{r}}_0) - \frac{\mu}{r_0}, \\
\mathbf{h} &= \mathbf{r}_0 \times \dot{\mathbf{r}}_0.
\end{aligned} \tag{101}$$

5) Compute the semi-major axis and the eccentricity

$$\begin{aligned}
a &= -\frac{\mu}{2C}, \\
e &= 0.
\end{aligned} \tag{102}$$

6) Use the Euler-Cromer algorithm.

a) Calculate the velocities of m_i and m_j

$$
\begin{aligned}
v_{x_{n+1}} &= v_{x_{n+1}} + a_x(x_n, y_n)\Delta t, \\
v_{y_{n+1}} &= v_{y_{n+1}} + a_y(x_n, y_n)\Delta t.
\end{aligned}
\tag{103}
$$

b) Calculate the positions of m_i and m_j

$$
\begin{aligned}
x_{n+1} &= x_n + v_{x_{n+1}}\Delta t, \\
y_{n+1} &= y_n + v_{y_{n+1}}\Delta t.
\end{aligned}
\tag{104}
$$

5 RESULTS FROM THE SIMULATIONS

5.1 Initial Set-up and Test Case for the Sun-Earth-Star 2 System

The first case we will simulate is the test case. Initially, we will conduct this test to confirm the validity of our C++ program. For the test case simulation, we treat the Sun as the central mass, located at $(-7.062 \times 10^8, 0.0)$ m and other masses in our three-body system will be the Earth and Jupiter. However, we treat Jupiter as the second star in the system and call it Star 2. In all of the simulations we run, Earth will begin on the x-axis at the perihelion position, 1.47×10^{11} m. For the test simulation Star 2 will begin on the x-axis at the perihelion position, 7.41×10^{11} m, but this starting position will change depending on the simulation. Each of the masses in the system are given an initial velocity in the $+y$ direction. Earth's velocity is determined by using Kepler's third law relating the orbital period to the orbital velocity. The velocity for the Sun and Star 2 can be determined by using the equation of motion from the 2BP such that our system begins in a circular orbit configuration. In each of the figures, the green dot will represent the Sun, the blue line will represent Earth's orbit, and the cyan line will represent the orbit of Star 2.

For the test case scenario, the duration of simulation runs for one Neptune year, or approximately 165 Earth years, in one Earth day increments of 8.64×10^4 seconds. The result of the test case is shown below in Figure 10. In the figure, we can observe that the Earth and Star 2 orbit the Sun at a distance of 1 AU and 4.95 AU from the Sun, which is placed at the center. We observe that the orbits are both circular, or symmetric with respect to both axes, and there is no erratic behavior displayed by the Earth. This result was expected from the theory and we have shown that our code outputs the correct solution.

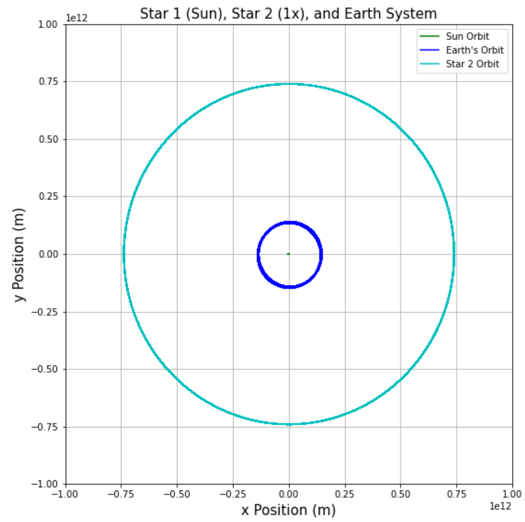

Fig. 10. This image displays the results of the test case simulation. The system contains the Sun at the center and Earth is in an orbit 1 AU from the Sun and Star 2 is in an orbit 4.95 AU from the Sun. The green line represents the Sun's orbit, the blue line represents the Earth's orbit, and the cyan line represents the orbit of Star 2. Earth displays no erratic behavior.

5.2 Star 2 Located at Perihelion Position, Mass Varied

The next simulation we will initialize the Star 2 location at the perihelion position of Jupiter and vary the mass beginning with the original mass of Jupiter, $M_{J_0} = 1.90 \times 10^{27}$ kg, up to the mass of the Sun, $M_S \approx 10^{30}$ kg. Star 2 is placed at 7.41×10^{11} m, or 4.95 AU with the same initial velocity used in the test case. The initial conditions for Earth are set to the test case parameters, where the initial position is the perihelion position of the Earth, with the same initial velocity.

In Figure 11 below, the results show when the mass of Star 2 is $1, 100, 195$, and 310 times M_{J_0} located at 4.95 AU. The top left image is exactly the same as the test case and we refer the reader to the previous section. When Star 2 has mass of 100 times M_{J_0} (top right), we notice the Earth's orbit begins to expand outward in the radial direction and the orbit of Star 2 becomes elongated, no longer symmetric. As the mass of Star 2 increases to 195 times M_{J_0} (bottom left), we can see that the Earth's orbit begins in a circular orbit

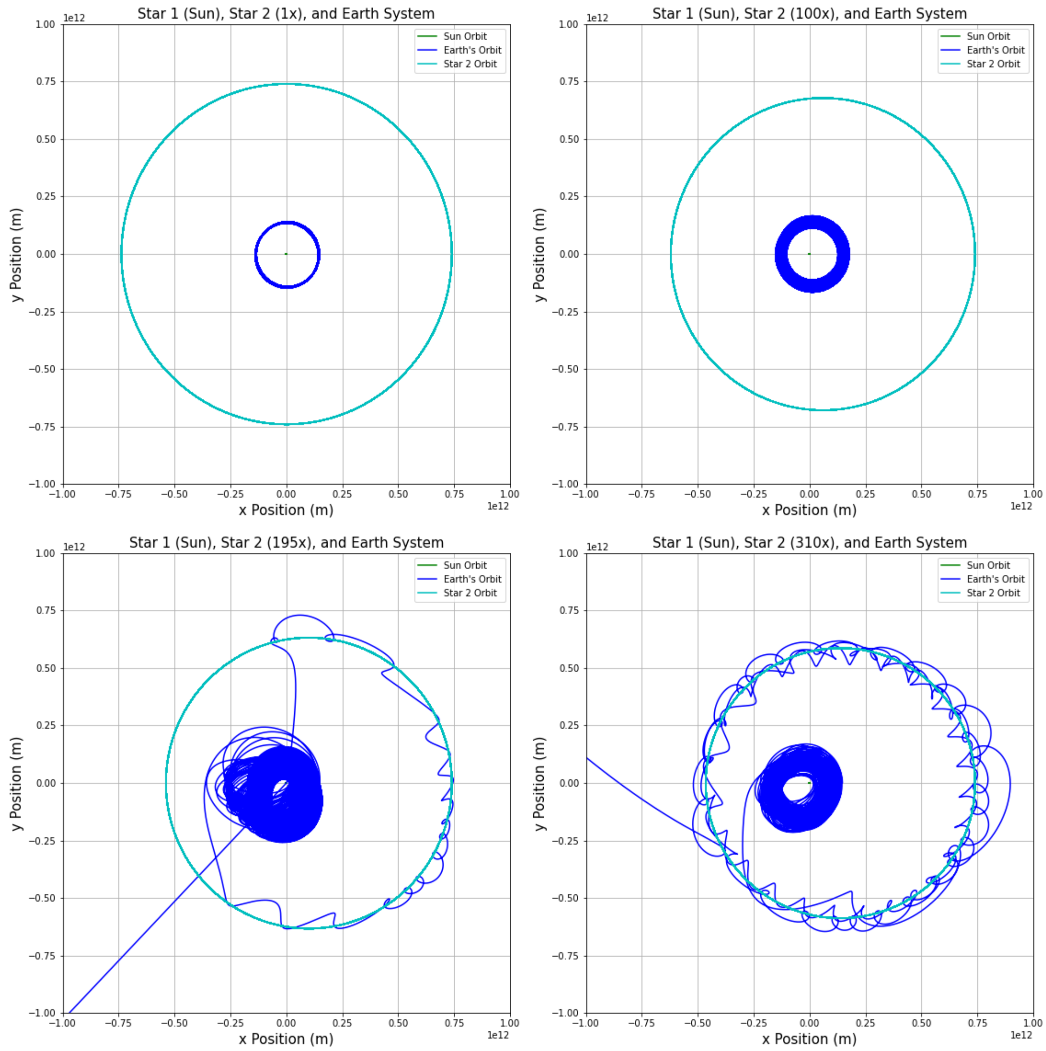

Fig. 11. The stability of Earth's orbit. Star 2 is set at the perihelion position of Jupiter, 7.41×10^{11} m. The mass of Star 2 is changed beginning with the initial mass of Jupiter, $M_{J_0} = 1.90 \times 10^{27}$ kg (top left), $100 \times M_{J_0}$ (top right), $195 \times M_{J_0}$ (bottom left), and $310 \times M_{J_0}$ (bottom right).

around the Sun and expands outward, no longer being circular. Eventually, Earth's trajectory leaves its initial path around the Sun and escapes, then reaches the orbit of Star 2. We can also observe the oscillating patterns of Earth around Star 2 which then does a fly-by the Sun and leaves the binary star system. We continue to increase the mass of Star 2. When the mass of Star 2 reaches 310 times M_{J_0} (bottom right), we observe more instabilities with the orbit of Earth. Earth revolves around the Sun for a while, then eventually leaves to be captured by Star 2. We see that Earth orbits around Star 2 for a long time but eventually escapes the binary system.

We continue to increase the mass of Star 2 and observe the erratic patterns of Earth's orbit in the binary star system. Figure 12 displays the results when the mass of Star 2 is $440, 500, 905$, and 1000 times M_{J_0}. When the mass of Star 2 is 440 times M_{J_0} (top left), we observe that Earth orbits the Sun for a few cycles but eventually takes on a new trajectory. In this case, we see that Earth is flying around the system, in no predictable pattern. Earth leaves the system and the Star 2 orbit is becoming even more elongated. When the mass of Star 2 is half the mass of the Sun, 500 times M_{J_0} (top right), the Earth orbits the Sun roughly four times and then exits the system. The orbit of Star 2 is elliptical. The bottom row images show when the mass of Star 2 is 905 and 1000 times M_{J_0} (the mass of the Jupiter). In the image on the left, we see Earth orbits the Sun three times and then escapes towards Star 2. Star 2 then captures Earth and Earth oscillates around Star 2 until it leaves the system. In the image on the right, we see Earth leave the system relatively quickly when Star 2 has the mass of the Sun. Also, we observe the elliptical shape of the orbit of Star 2.

In our analysis, we have shown only a few figures to highlight the trajectories of Earth's orbit in the binary star system. We generated many other plots showing Earth's orbit, all being unpredictable, displaying no pattern. We felt that the selected figures provide the best representations of Earth's behavior. We refer the reader to Table 2 for each case we simulated. In the table, dark gray represents the stability of Earth's orbit, when it does not escape from the system. The light gray represents situations in which the orbit becomes unstable and Earth escapes the system. Table 2 shows that Earth becomes unstable when the mass of Star 2 is between 165 to 170 times the mass of Jupiter.

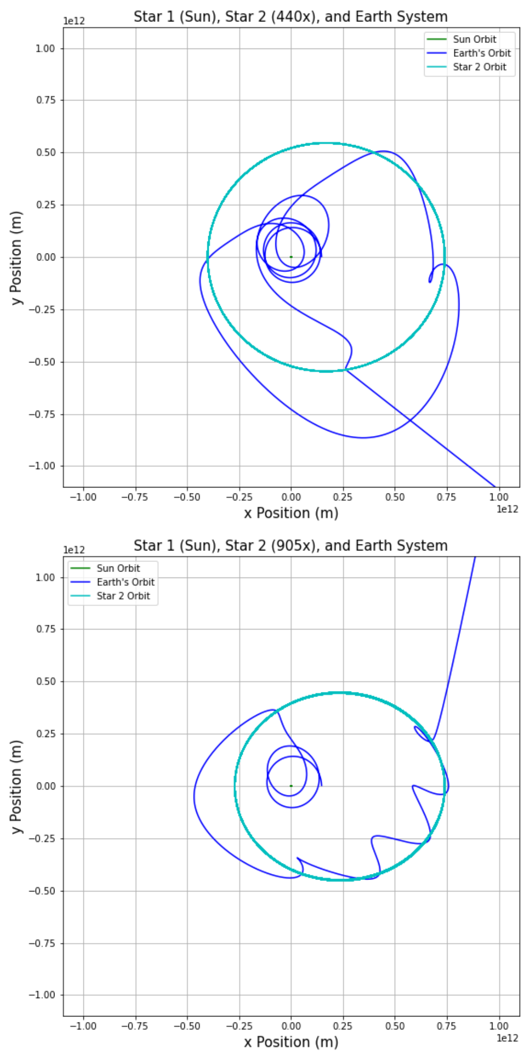

Fig. 12. The stability of Earth's orbit. Star 2 is set at the perihelion position of Jupiter, 7.41×10^{11}m. The mass of Star 2 is changed beginning with $440 \times M_{J_0}$ (top right), $500 \times M_{J_0}$ (top left), $905 \times M_{J_0}$ (bottom left), and $1000 \times M_{J_0}$ (bottom right).

Table 2. Changing the mass of Star 2 in increments of 10 times the mass of Jupiter from 1 – 100, then changing the mass in increments of 5 times from 100 – 1000.

Mass Change of Star 2											
1x	10x	20x	30x	40x	50x	60x	70x	80x	90x	100x	105x
110x	115x	120x	125x	130x	135x	140x	145x	150x	155x	160x	165x
170x	175x	180x	185x	190x	195x	200x	205x	210x	215x	220x	225x
230x	235x	240x	245x	250x	255x	260x	265x	270x	275x	280x	285x
290x	295x	300x	305x	310x	315x	320x	325x	330x	335x	340x	345x
350x	355x	360x	365x	370x	375x	380x	385x	390x	395x	400x	405x
410x	415x	420x	425x	430x	435x	440x	445x	450x	455x	460x	465x
470x	475x	480x	485x	490x	495x	500x	505x	510x	515x	520x	525x
530x	535x	540x	545x	550x	555x	560x	565x	570x	575x	580x	585x
590x	595x	600x	605x	610x	615x	620x	625x	630x	635x	640x	645x
650x	655x	660x	665x	670x	675x	680x	685x	690x	695x	700x	705x
710x	715x	720x	725x	730x	735x	740x	745x	750x	755x	760x	765x
760x	765x	770x	775x	780x	785x	790x	795x	800x	805x	810x	815x
820x	825x	830x	835x	840x	845x	850x	855x	860x	865x	870x	875x
880x	885x	890x	895x	900x	905x	910x	915x	920x	925x	930x	935x
940x	945x	950x	955x	960x	965x	970x	975x	980x	985x	990x	995x
1000x											

5.3 Star 2 Equal to the Sun's Mass, Perihelion Position Varied

We saw in the previous simulation that as the mass of Star 2 increases, Earth's orbit becomes unstable and eventually is ejected from the system. At the largest mass, $M_{J_0} = 1.90 \times 10^{30}$ kg, the most dramatic effects were noticed, in particular, when Earth orbits the Sun twice and then leaves the system. For our next set of simulations, we begin with this set of parameters and let the mass of Star 2 be equal to the mass of the Sun. In this simulation, we vary the perihelion position of Star 2. All of the images in this section represent a binary star system containing the Earth. We discuss the results shown in Figure 13 and Figure 14.

In Figure 13, the results show the orbit of Earth when Star 2 is located at 1 and 2 times the perihelion position of Jupiter. These position values are 7.41×10^{11}m (4.95 AU) and 1.48×10^{12} m (9.89 AU). In the left image, we observe that Earth escapes after a few orbits around the Sun. The orbit of Star 2 also looks elliptical. When the perihelion distance is doubled in the right image, Earth is ejected after many orbital periods.

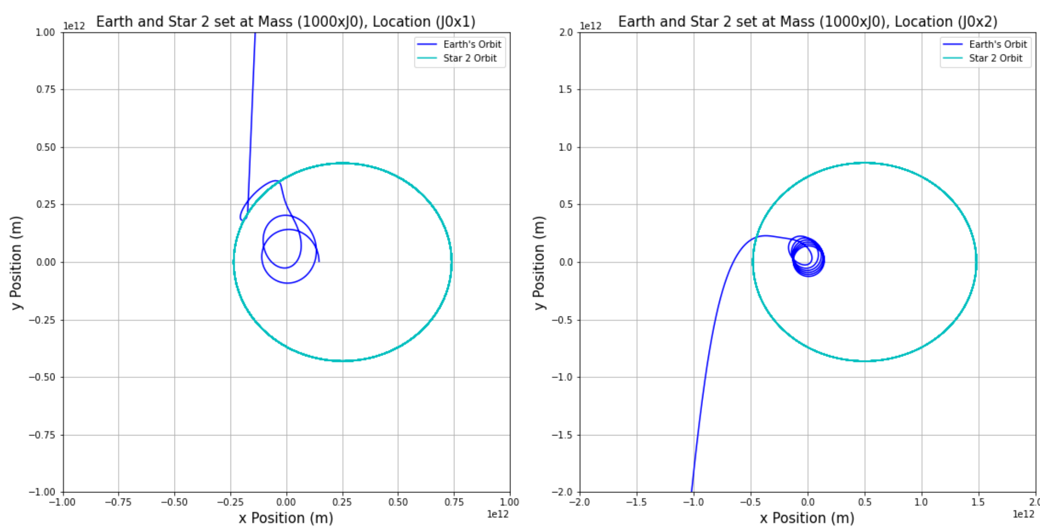

Fig. 13. The stability of Earth's orbit. Star 2 is set to mass of 1.90×10^{30} kg, or $1000 \times M_{J_0}$. The perihelion position is changed beginning at $1.0 \times J_0$ or 4.95 AU (left) and $2.0 \times J_0$ or 9.89 AU (right).

In Figure 14, we increase the distance by $2.5, 3, 10$, and 100 times the distance of Jupiter, J_0. When Star 2 is located at 2.5 times J_0 (top left) or 1.85×10^{12} m (12.4 AU), Earth orbits the Sun over many periods but then escapes out to the orbit of Star 2.

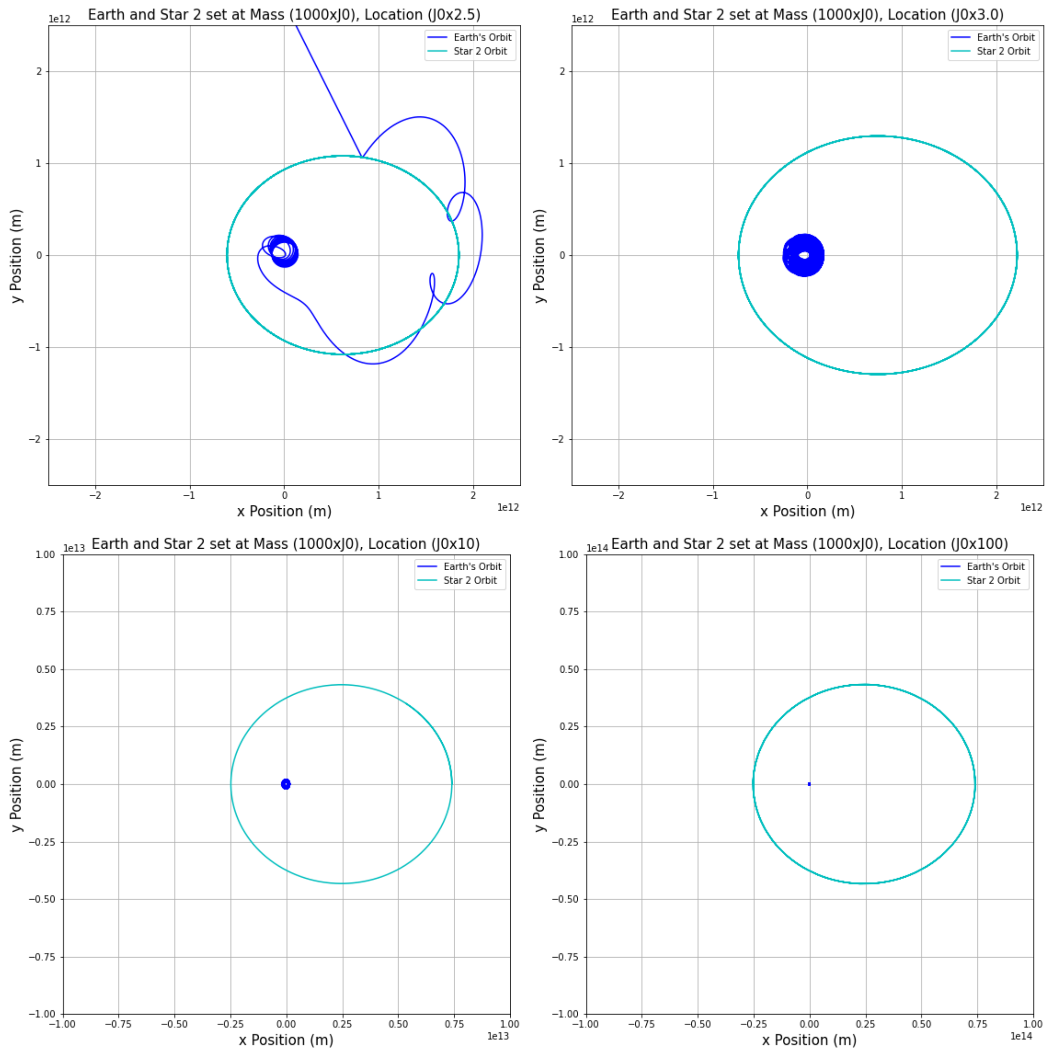

Fig. 14. The stability of Earth's orbit. Star 2 is set to mass of 1.90×10^{30} kg, or $1000 \times M_{J_0}$. The perigee position is $2.5 \times J_0$ or 12.4 AU (top left), $3.0 \times J_0$ or 14.8 AU (top right), $10 \times J_0$ or 49.5 AU (bottom left), and $100 \times J_0$ or 247.4 AU (bottom right).

Earth oscillates around Star 2 for two cycles, and then leaves the system. When Star 2 is positioned at 3.0 times J_0 (top right) or 2.22×10^{12} m (14.8 AU), Earth displays erratic behavior but never leaves the orbit around the Sun. There is still a circular nature to the orbit. As we continue to place Star 2 further out at distances 10 times J_0 or 7.41×10^{12} m (49.5 AU) and at 50 times J_0 or 3.71×10^{13} m (247.4 AU), we observe Earth remains in orbit around the Sun and does not leave the system or reach the orbit of Star 2.

In Table 3, we display all of the location changes for Star 2. In this table, the light grey represents the instability of Earth's orbit or when erratic behavior can be seen and the dark gray represents the stability of Earth's orbit or when the Earth remains bound to the Sun and does not reach the orbit of Star 2. We can see from the table that Earth's orbit is stable between the range of $2.8 - 2.9$ times J_0.

Table 3. Changing the location of Star 2 when the mass of Star 2 is 1.90×10^{30} kg. The dark grey represents when the orbit of Earth is stable at the mass. The light gray represents when the orbit of Earth is unstable, leading to ejection.

Location Change of Star 2							
1x	1.5x	2.0x	2.1x	2.2x	2.3x	2.4x	2.5x
2.6x	2.7x	2.8x	2.9x	3.0x	3.1x	3.2x	3.3x
3.4x	3.5x	3.6x	3.7x	3.8x	3.9x	4.0x	5.0x
10x	20x	30x	40x	50x	60x	70x	80x
90x	100x	150x					

5.4 Star 2 Mass and Perihelion Position Varied

In the previous sections, we fixed the location of Star 2 and varied the mass or we fixed the mass and varied the location. In each of the simulations, we observed the behavior of Earth. Now, we will investigate how varying both the mass and the location of Star 2 impacts the behavior of Earth. In this section, for each set of positions of Star 2, which are $1, 2.5, 5, 10, 25,$ and 50 times J_0, the mass of Star 2 is varied by $1, 100, 500,$ and 1000 times M_{J_0}. We are only concerned with the orbit of Earth as shown in the following figures 15 - 20 below. In Figure 15, Star 2 is located at 4.95 AU and the mass is varied.

When Star 2 is equal to the mass of Jupiter, Earth's exhibits no sign of erratic behavior and remains in circular motion around the Sun. As the mass increases to 100 times M_{J_0} (top right), Earth's orbit begins to show signs of expanding radially outward but remains in a circular motion around the Sun. As the mass increases to 500 and 1000 times M_{J_0}, Earth's orbit becomes erratic and ultimately, leaves the system. We have already seen these results in Section 5.2.

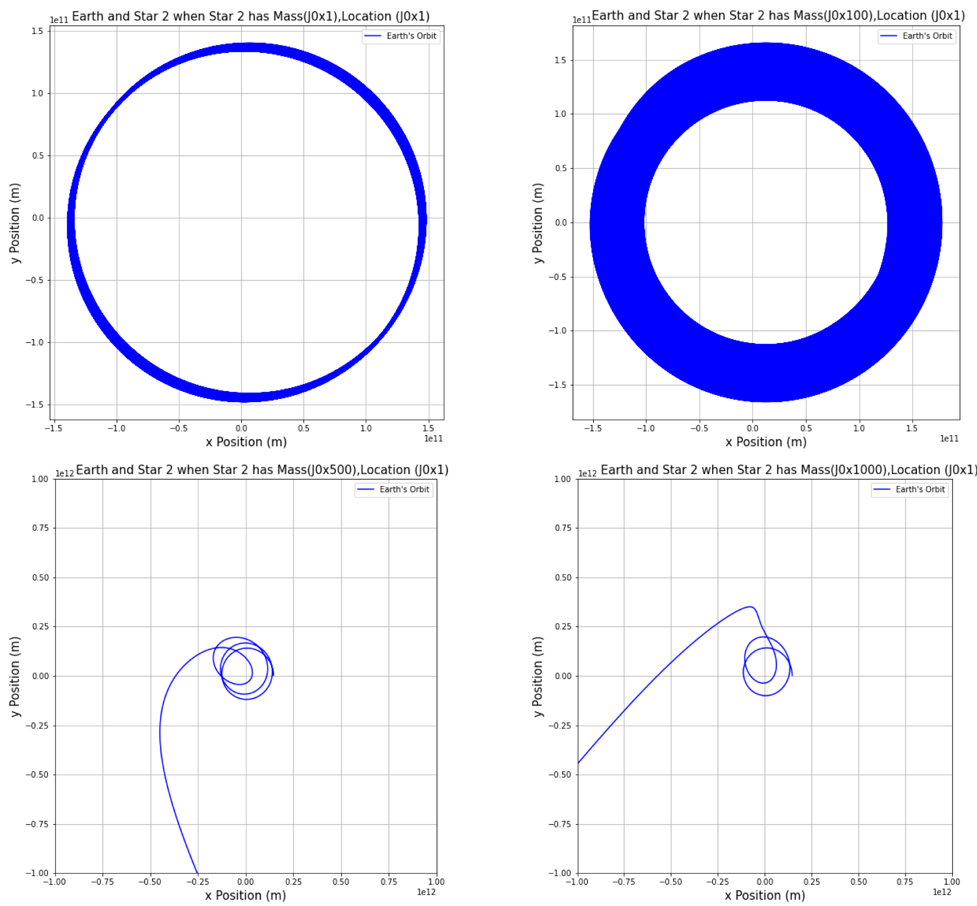

Fig. 15. The stability of Earth's orbit. Star 2 begins at the perihelion location of Jupiter, 7.41×10^{11} m or approximately 4.95 AU. The mass of Star 2 starts at the original mass of Jupiter, $1 \times M_{J_0}$ (top left). We vary the mass by $100 \times M_{J_0}$ (top right), $500 \times M_{J_0}$ (bottom left), and $1000 \times M_{J_0}$ (bottom right), where $M_{J_0} = 1.90 \times 10^{27}$ kg.

In Figure 16, the Star 2 is located 12.4 AU, outside of orbit of Saturn, and the mass is varied. We can see that when the mass of Star 2 is equal to Jupiter (top left), Earth demonstrates circular motion around the Sun and does not leave the system. As the mass increases to 100 (top right) and 500 times M_{J_0} (bottom left), Earth's orbit begins to expand radially outward. When Star 2 is 1000 times M_{J_0}, it is clear that Earth begins to show erratic behavior. Earth orbits the Sun over many periods and eventually leaves the system entirely.

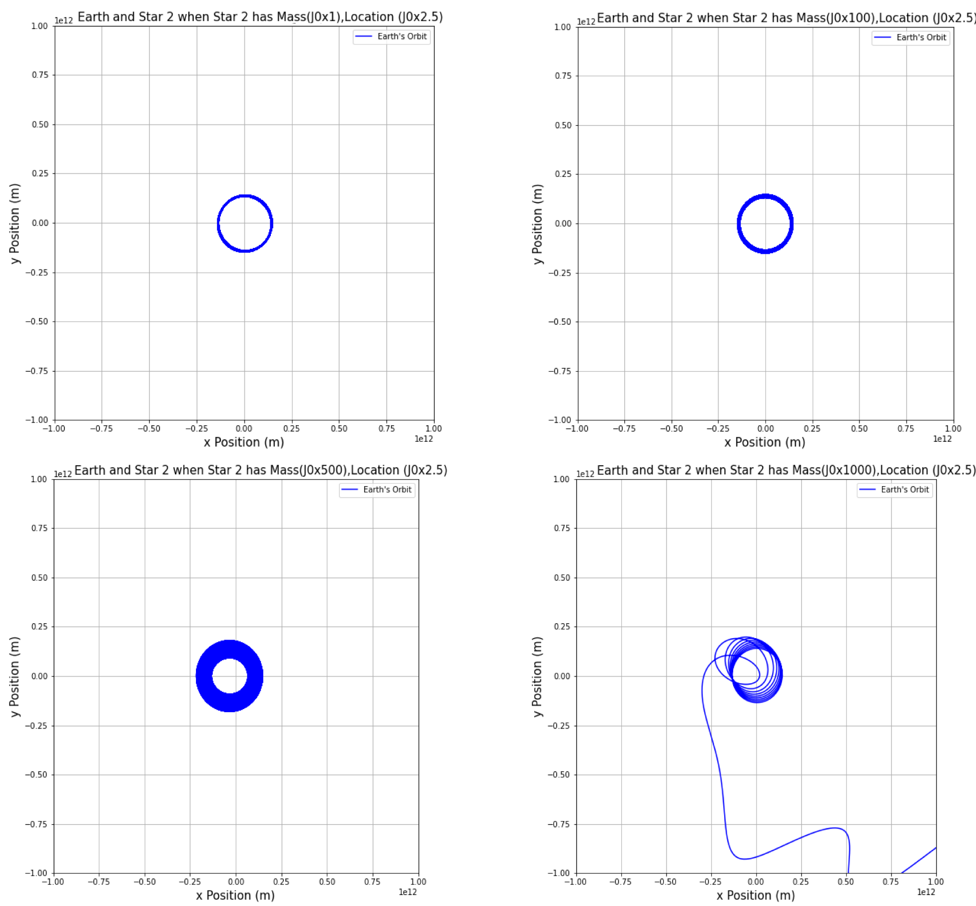

Fig. 16. The stability of Earth's orbit. Star 2 begins at a distance 2.5$\times$ further than the perihelion position of Jupiter, $\approx 1.85 \times 10^{12}$ m or approximately 12.4 AU. The mass of Star 2 begins at the oringal mass of Jupiter $1 \times M_{J_0}$ (top left). We vary the mass by $100 \times M_{J_0}$ (top right), $500 \times M_{J_0}$ (bottom left), and $1000 \times M_{J_0}$ (bottom right), where $M_{J_0} = 1.90 \times 10^{27}$ kg.

In Figure 17, Star 2 is located 24.8 AU, between the orbits of Uranus and Nepture, and the mass is varied. With Star 2 having the mass of Jupiter or 100 times M_{J_0}, we see that it does not have an apparent influence on Earth's orbit. See the top left and right images. Once again, as Star 2 reaches half the mass of the Sun 500 times M_{J_0} or equal to the mass of the Sun, 1000 times M_{J_0}, we see the effects on Earth's orbit. In the bottom left and right images, we notice Earth is beginning to become unstable and the orbit continues to expand outward. However, over the length of this simulation, Earth never escapes the system.

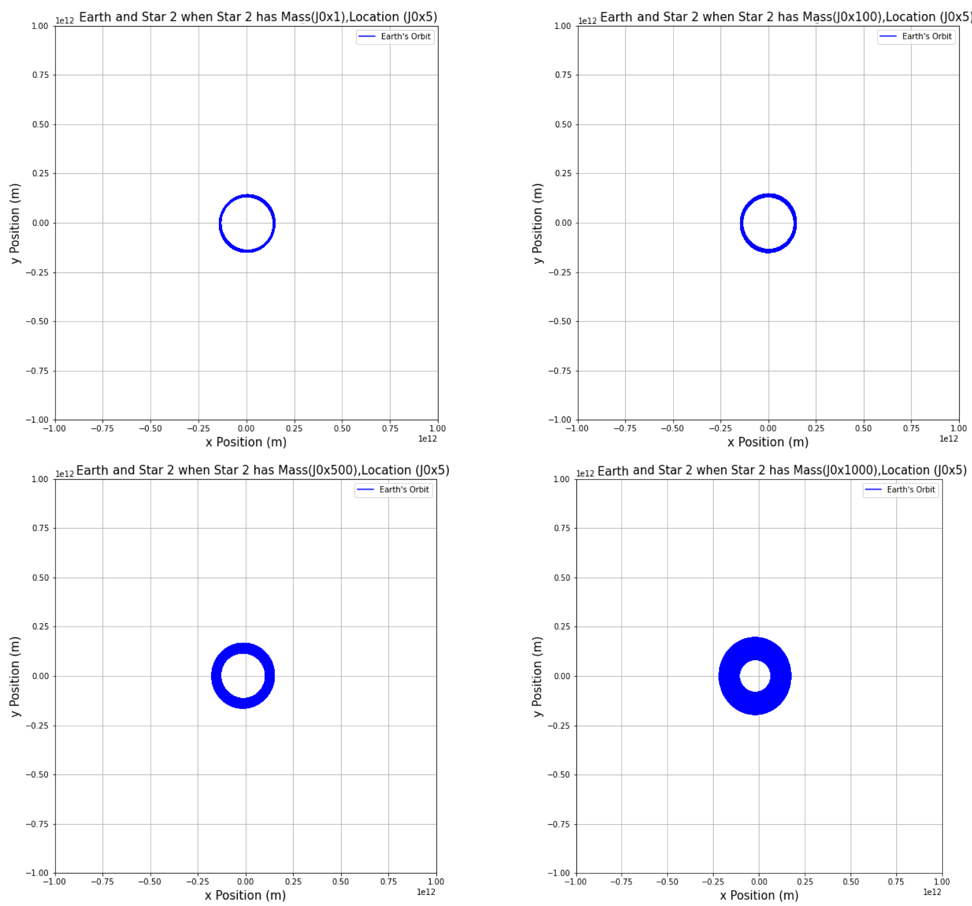

Fig. 17. The stability of Earth's orbit. Star 2 begins at a distance $5\times$ further than the perihelion position of Jupiter, $\approx 3.70 \times 10^{12}$ m, or approximately 24.8 AU. The mass of Star 2 begins at the original mass of Jupiter $1 \times M_{J_0}$ (top left), and then we vary the mass by $100 \times M_{J_0}$ (top right), $500 \times M_{J_0}$ (bottom left), and $1000 \times M_{J_0}$ (bottom right), where $M_{J_0} = 1.90 \times 10^{27}$ kg.

In Figure 18, Star 2 is located 49.5 AU, past Pluto and somewhere in the Kuiper Belt, and the mass is varied. In these images, we can see that Earth remains in a circular orbit around the Sun over the entire simulation. When the Star 2 has a mass of 500 and 1000 times M_{J_0}, Earth's orbit expands outward. At this distance, when the mass of the stars in the system are equal to each other, Earth shows the beginning of erratic behavior.

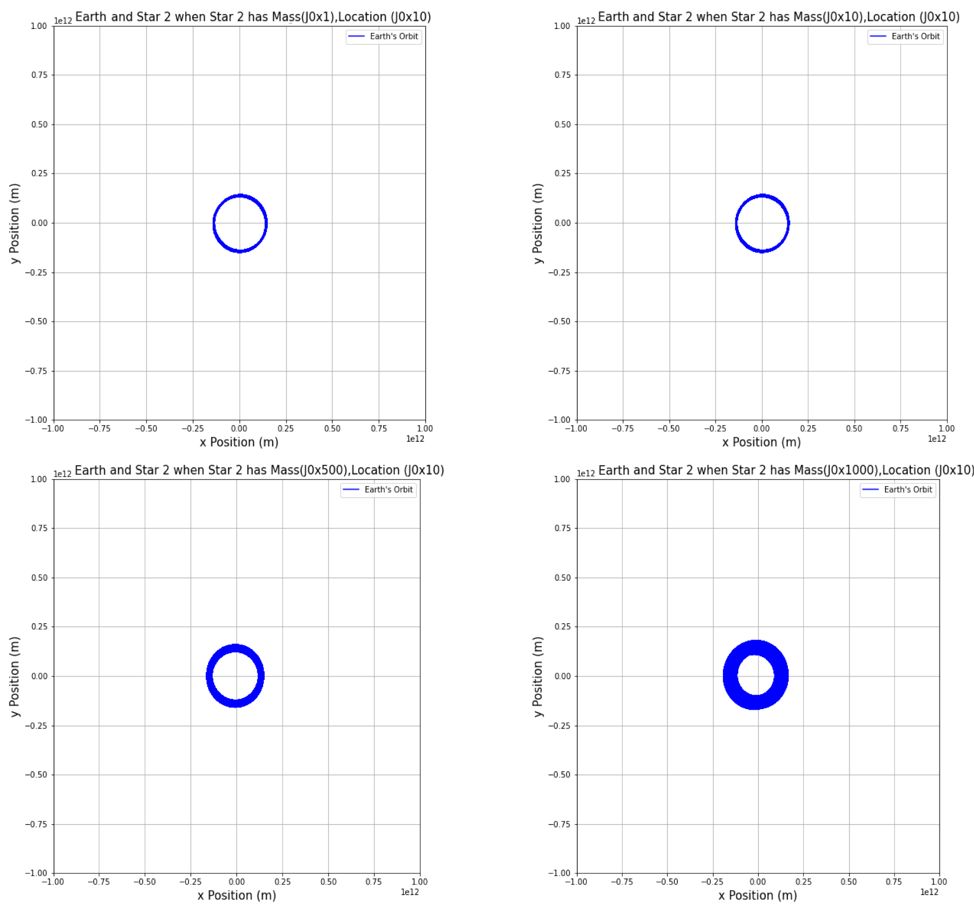

Fig. 18. The stability of Earth's orbit. Star 2 begins at a distance $10\times$ further than the perihelion position of Jupiter, $\approx 7.4\times10^{12}$ m, or approximately 49.5 AU. The mass of Star 2 begins at the original mass of Jupiter $1\times M_{J_0}$(top left), and then we vary the mass by $100\times M_{J_0}$ (top right), $500\times M_{J_0}$ (bottom left), and $1000\times M_{J_0}$ (bottom right), where $M_{J_0} = 1.90\times10^{27}$ kg.

In Figures 19 and 20 below, Star 2 is located 124 AU and 248 AU. These distances are comparable to where the trans-Neptunian object, 2018 VG18 was discovered, and half way to where Planet Nine is supposed to exist (somewhere between $400-800$ AU). We can see in the figures that Earth's orbit remains in circular motion around the Sun.

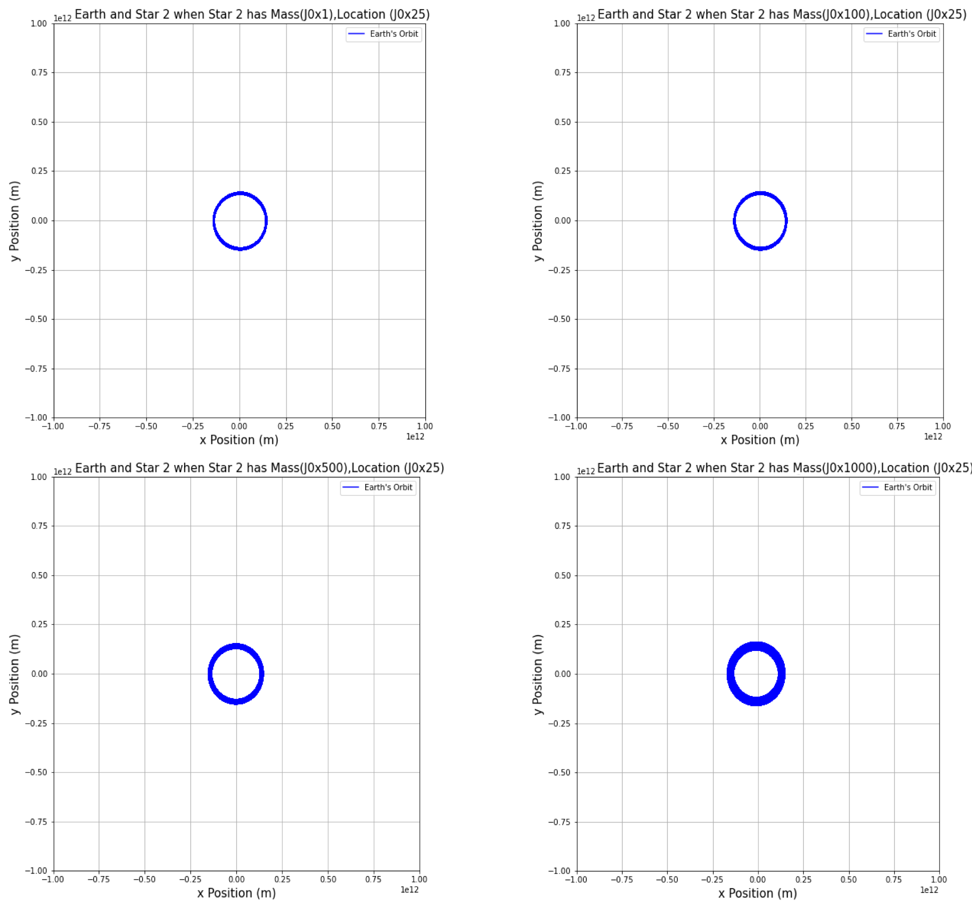

Fig. 19. The stability of Earth's orbit. Star 2 begins at a distance $25\times$ further than the perihelion position of Jupiter, $\approx 1.85\times10^{13}$ m, or approximately 124 AU. The mass of Star 2 begins at the original mass of Jupiter,$1\times M_{J_0}$(top left), and then we vary the mass by $100\times M_{J_0}$ (top right), $500\times M_{J_0}$ (bottom left), and $1000\times M_{J_0}$ (bottom right), where $M_{J_0}=1.90\times10^{27}$ kg.

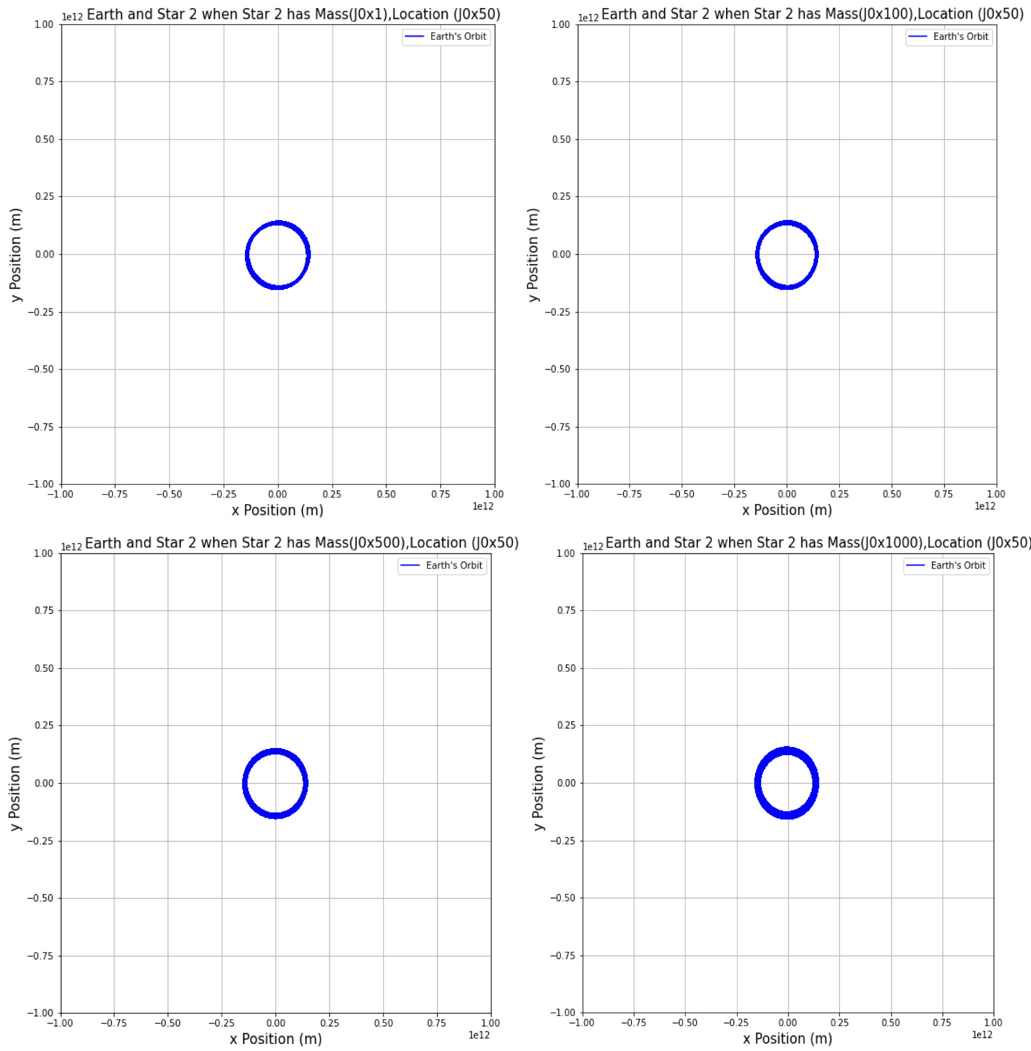

Fig. 20. The stability of Earth's orbit. Star 2 begins at a distance $50\times$ further than the perihelion position of Jupiter, $\approx 3.70 \times 10^{13}$ m, or approximately 248 AU. The mass of Star 2 begins at the original mass of Jupiter, $1 \times M_{J_0}$ (top left), and then we vary the mass by $100 \times M_{J_0}$ (top right), $500 \times M_{J_0}$ (bottom left), and $1000 \times M_{J_0}$ (bottom right), where $M_{J_0} = 1.90 \times 10^{27}$ kg.

However, we can see that Earth's orbit is beginning to expand when Star 2 has the same mass as the Sun. See the bottom left images of Figures 19 and 20.

Table 4 contains all of the simulations we ran. In this table, the dark gray boxes with a S (stable) represents the stability of the orbit of Earth. This means the Earth remains in orbit around the Sun and does not escape the system. The light gray boxes with an U (unstable) is used to present the instability of the orbit of Earth. In these cases, Earth no longer orbits the Sun and escapes the system.

Table 4. Changing the mass of Star 2 located at different positions. The dark gray represents when the orbit of Earth is stable at the mass value of Star 2. The light gray represents when the orbit of Earth is unstable, leading to ejection.

Mass and Location Change of Star 2							
Location/Mass:	1x	10x	50x	100x	250x	500x	1000x
1x	S	S	S	S	U	U	U
2.5x	S	S	S	S	S	S	U
5x	S	S	S	S	S	S	S
10x	S	S	S	S	S	S	S
25x	S	S	S	S	S	S	S
50x	S	S	S	S	S	S	S

6 DISCUSSION

6.1 Overview of the Results

The test case simulation for the Sun-Earth-Star 2 system showed that our program was running properly. We observe that the orbits of Earth and Star 2 show no signs of instability. From the gravitational effects, Earth is bound to the Sun and the interaction between the Earth and Star 2 is not large enough to overcome the attraction from the Sun. This system is the same as the Sun-Earth-Jupiter three-body system and so we expect the orbits to be circular and that Earth remains in orbit around the Sun.

In the first simulation, we varied the mass of Star 2 while placing it in Jupiter's orbit. The mass was varied beginning with the mass of Jupiter and ending with the mass of the Sun. By varying and increasing the mass of Star 2, we noticed that Earth's orbit became increasingly unstable. At lower masses, the orbit of Earth begins to expand radially outward, experiencing the gravitational pull from Star 2. The orbit continues to increase until the mass of Star 2 reaches the mass between 165 and 170 times M_{J_0}. Once Star 2 reaches this range of mass, Earth begins to exhibit erratic trajectories around the binary pair, possibly gaining enough orbital energy to escape the system. These trajectories exhibit no particular pattern. As the mass continues to increase, we can actually observe that Star 2 could capture the Earth after leaving its initial orbit around the Sun. When this happens, we can see that Earth orbits around the second star in our binary system. As the mass of Star 2 approaches the mass of the Sun, in a short time Earth is ejected from the system.

In the next simulation, Star 2 was set to the mass of the Sun and the position of Star 2 was varied. When Star 2 was placed at 4.95, 9.89, and 12.4 AU, we observed that Earth orbits the Sun over a few periods and eventually escapes, even reaching the orbit of Star 2. Then Earth oscillates around Star 2 and after a short time leaves the system. As the distance of Star 2 is further increased to a distance of 14.8, 49.5, and 247.4 AU, we see that Earth can still experience the gravitational effects from Star 2, but does not exit the system in this time interval. From these results, we can see that when Star 2 has the same mass of the Sun, the closer Star 2 is, the greater the gravitational influence it has on Earth. As Star 2 moves further away, these gravitational effects decrease and Earth remains in orbit around the Sun. Specifically, it is in the range of $2.8-2.9$ times J_0 or $13.8-14$ AU of Star 2, where Earth transitions from an unstable orbit to a stable orbit. Earth is in a stable orbit when Star 2, with a mass of 1.90×10^{30} kg, is located outside the orbit of Saturn.

The last simulation we performed involved varying both the mass and locations of Star 2. When Star 2 was placed at 4.95 AU, at Jupiter's orbit, we saw that as the mass increases, Earth's orbit became unstable. Earth begins in an orbit around the Sun but then expands radially outward and leaves the system after a few orbits around the Sun. As we moved Star 2 out, away from the Sun, we saw the gravitational effects on the Earth become less severe. For example, when Star 2 was at 12.4 AU, around the orbit of Saturn, Earth did not leave until the mass of Star 2 was equal to the mass of the Sun. We did observe erratic behavior but it was less pronounced. It was not until Star 2 was located at 24.8 AU that Earth remained in the system, revolving around the Sun. At this distance, which is between the orbits of Uranus and Neptune, Earth remained in a stable orbit. However, we did see a radial expansion of the orbit of Earth, but it never left the system. As the position of Star 2 moved outward to distances $49.5, 124$, and 248 AU, we observed that Earth remained in a stable configuration. At these distances, which is past the orbit of Pluto and out into the Kuiper Belt, Earth felt the gravitational effects when the mass of Star 2 was equal to the mass of the Sun.

6.2 Concluding Remarks

In this thesis, we analyzed the effects of Earth's trajectory in a binary star system. We used the three-body model to explore the behavior of Earth's orbit when the mass of Star

2 is varied, when the position of Star 2 is varied, and when both the mass and position of Star 2 is varied. What we learned is that there are ranges of masses and distances of Star 2 that leads to Earth being unstable over long periods of time. This type of study may be used to analyze other types of binary star systems, like the Alpha Centauri System, to investigate where planets are likely to exist or where they are likely not to exist.

Literature Cited

R. R. Bate, D. D. Mueller, and J. E. White, *Fundamentals of Astrodynamics*, pp. 19–20. Dover Publications Inc., 1971.

K. Batygin, F. C. Adams, M. E. Brown, and J. C. Becker, "The Planet Nine Hypothesis," *Physics Reports*, vol. 805, p. 1–53, 2019.

K. Batygin and G. Laughlin, "On the Dynamical Stability of the Solar System," *The Astrophysical Journal*, vol. 683, no. 2, p. 1207–1216, 2008.

V. R. Bond and M. C. Allman, *Modern Astrodynamics: Fundamentals and Perturbation Methods*, pp. 19–27. Princeton University Press, 1996.

J. Danby, *Fundamentals of Celestial Mechanics*, pp. 155 – 157. New York - The Macmillan Company, 1962.

J. Donnison and I. Williams, "Orbital Stability of Massive Protoplanets in the Terrestrial Planet Region of the Solar System," *Moon and Planets*, vol. 19, no. 4, pp. 421–424, 1978.

R. Dvorak, C. Froeschle, and C. Froeschle, "Stability of Outer Planetary Orbits (P-types) in Binaries," *Astronomy and Astrophysics*, vol. 226, no. 1, pp. 335–342, 1989.

F. Graziani and D. C. Black, "Orbital Stability Constraints on the Nature of Planetary Systems," *The Astrophysical Journal*, vol. 251, no. 1, pp. 337–341, 1981.

P. Hamill, *Intermediate Dynamics*, pp. 364–368. Jones & Bartlett Learning, 2010.

M. J. Holman and P. A. Wiegert, "Long-term Stability of Planets in Binary Systems," *The Astronomical Journal*, vol. 117, no. 1, pp. 621–628, 1999.

J. Laskar, "Is the Solar System Stable?," in *Chaos: Poincaré Seminar 2010* (B. Duplantier, S. Nonnenmacher, and V. Rivasseau, eds.), pp. 239–270, Basel: Springer Basel, 2013.

C. D. Murray and S. F. Dermott, *Solar System Dynamics*, pp. 52, 225–228. Cambridge University Press, 1999.

R. Pakter and Y. Levin, “Stability of Planetary Systems: A Numerical Didactic Approach,” *Americal Journal of Physics*, vol. 87, no. 69, pp. 69–74, 2019.

Y. J. Pendleton and D. C. Black, “Further Studies on Criteria for the Onset of Dynamical Instability in General Three-body Systems,” *The Astronomical Journal*, vol. 88, no. 9, pp. 1415–1419, 1983.

A. Romanowsky. Private Communication, 2021.

A. Siraj and A. Loeb, “The Case for an Early Solar Binary Companion,” *The Astronomical Journal Letters*, vol. 899, no. L24, pp. 1–4, 2020.

P. A. Wiegert and M. J. Holman, “The Stability of Planets in the Alpha Centauri System,” *Astrophysics Journal*, vol. 113, pp. 1445–1450, 1997.

Appendix A

C++ PROGRAMS

We present the program used in the simulations. The code provided contains comments about each part of the program.

```
// Source File: Orbital_Code.cpp
// This program simulates the orbital positions, velocities, and
// accelerations as well as the orbital energy, angular momentum
// for the primary body under the gravitational influence of a
// central mass and/or another mass in the two-body and Three-Body
// problem (TBP and 3BP).

// Include the libraries:
        #include <iostream>
        #include <fstream>
        #include <cstdlib>
        #include <string>
        #include <cmath>

// namespace give the freedom to use short, accurate names.
        using namespace std;

// Declaring the size of the array.
        const int Size = 11e6;

// Defining the variable arrays for Star 1, Earth, and Star 2.
// The arrays will store the time and the x,y-directional positions
// and velocities for Star1, Earth, and Star 2.
        double Time[Size];
        double Star1xPosition[Size], Star1yPosition[Size],
                EarthxPosition[Size], EarthyPosition[Size],
                Star2xPosition[Size], Star2yPosition[Size];
        double Star1xVelocity[Size], Star1yVelocity[Size],
                EarthxVelocity[Size], EarthyVelocity[Size],
                Star2xVelocity[Size], Star2yVelocity[Size];
```

```
// The following lines will define the functions used throughout
// the program.
// ================================================================

// Defining the Acceleration Functions for Star 1, Earth, and Star2
// xi's and yi's are the positions of the bodies inputted into the
// functions, where i = 1,2,3.

        double Star1xAcceleration(const double& x1, const double& y1,
                                  const double& x3, const double& y3);
        double Star1yAcceleration(const double& x1, const double& y1,
                                  const double& x3, const double& y3);
        double EarthxAcceleration(const double& x1, const double& y1,
                                  const double& x2, const double& y2,
                                  const double& x3, const double& y3);
        double EarthyAcceleration(const double& x1, const double& y1,
                                  const double& x2, const double& y2,
                                  const double& x3, const double& y3);
        double Star2xAcceleration(const double& x1, const double& y1,
                                  const double& x3, const double& y3);
        double Star2yAcceleration (const double& x1, const double& y1,
                                  const double& x3, const double& y3);

// The energy (C), angular momentum (h), semi-major axis (a), and
// eccentricity (e) functions for the TBP. x and y represent the
// position, vx and vy represent the velocity of the planet.

        double energy(const double & x, const double& y,
                      const double& vx, const double& vy);
        double angularmomentum(const double& x, const double& y,
                      const double& vx, const double& vy);
        double semimajoraxis(const double& C);
        double eccentricity(const double& A);

// The next set of functions are the numerical methods used in this
```

```
// problem. These methods are the Euler-Cromer and the Runge-Kutta 4th
// order method.

// Runge-Kutta (RK4) Function
        void RK4(const int& N,
                const double& InitialTime, const double& FinalTime,
                const double& Star1X, const double& Star1Y,
                const double& Star1VX, const double& Star1VY,
                const double& EarthX, const double& EarthY,
                const double& EarthVX, const double& EarthVY,
                const double& Star2X, const double& Star2Y,
                const double& Star2VX, const double& Star2VY);

//Runge-Kutta Step Function
        void RK4STEP(double& t, const double& dt,
                double& x1, double& y1, double& vx1, double& vy1,
                double& x2, double& y2, double& vx2, double& vy2,
                double& x3, double& y3, double& vx3, double& vy3);

// Euler-Cromer Function
// Uncomment to use this method
        /*void EulerCromer(double& t, const double& dt,
                double& x1, double& y1, double& vx1, double& vy1,
                double& x2, double& y2, double& vx2, double& vy2,
                double& x3, double& y3, double& vx3, double& vy3); */

// Extra functions used to calculate the velocity of the planet and
// the Star. Note that each simulation begins the simulation with
// each of the bodies in a circular configuration.

        double circularvelocity(const double& x);
        double Star1Velocity(const double& x1, const double& y1,
                                const double& x3, const double& y3);

\\Begin the Main Program
int main(){
```

```
// Declaring the time variables:
        double InitialTime, FinalTime;

// number of times the program will iterate
        int N;

// Declaring Star1 Parameters:
        double Star1X, Star1Y, Star1VX, Star1VY;

// Declaring Earth parameters:
        double EarthX, EarthY, EarthVX, EarthVY;

// Declaring Star2 parameters:
        double Star2X, Star2Y, Star2VX, Star2VY;

        double Pi = 3.141592653589793;

// Number of iterations -This number will be varied
// depending how long the program runs for.
        N = 13.0e5; //
        InitialTime = 0.0; // beginning at time 0.
        //FinalTime = 6.0 * 365.0;

//Star1 initial conditions
        SunX = -7.062e8;
        SunY = 0.0;
        SunVX = 0.0;
        //SunVY = SunVelocity(SunX, SunY, JupiterX, JupiterY);

// Earth's initial conditions (beginning the pergiee position)
        EarthX = 1.47e11;
        EarthY = 0.0;
        EarthVX = 0.0;
        EarthVY = (2.0 * Pi * EarthX)/ (365.25 * 24.0 * 3600.0);
        //EarthVY = circularvelocity(EarthX);
```

```
// Star2 initial conditions
// These initial conditions are set at the perigee position
// of Jupiter with the speed.S
        Star2X = 7.40e11;
        Star2Y = 0.0;
        Star2VX = 0.0;
        Star2VY = circularvelocity(JupiterX);

        SunVY = SunVelocity(SunX, SunY, JupiterX, JupiterY);

// If the number of iterations are larger than the
// array size then the program will exit.

        if (N >= P) {cerr << "Error! N >= P \n"; exit (1);}

// Input all the parameters into the Runge-Kutta Algorithm
        RK(N, InitialTime, FinalTime,
                Star1X, Star1Y, Star1VX, Star1VY,
                EarthX, EarthY, EarthVX, EarthVY,
                Star2X, Star2Y, Star2VX, Star2VY);

// Enables writing to a .dat file
        ofstream myfile("OrbitalData.dat");
// Set precision to 6.
        myfile.precision(6);

// Begin the adding the elements to the file, iterate.
        for (i = 0; i < N; i++){
                myfile << Time[i] << " "
                << Star1xPosition[i] << " "
                << Star1yPosition[i] << " "
                << Star1xVelocity[i] << " "
                << Star1yVelocity[i] << " "
                << EarthxPosition[i] << " "
                << EarthyPosition[i] << " "
```

```
<< EarthxVelocity[i] << "␣"
<< EarthyVelocity[i] << "␣"
<< Star2xPosition[i] << "␣"
<< Star2yPosition[i] << "␣"
<< Star2xVelocity[i] << "␣"
<< Star2yVelocity[i] << "␣"
<< sqrt(EarthxPosition[i]*EarthxPosition[i] +
        EarthyPosition[i]*EarthyPosition[i]) << "␣"
<< sqrt(EarthxVelocity[i]*EarthxVelocity[i] +
        EarthyVelocity[i]*EarthyVelocity[i]) << "␣"
<< energy(EarthxPosition[i], EarthyPosition[i],
        EarthxVelocity[i],EarthyVelocity) << "␣"
<< angularmomentum(EarthxPosition[i], EarthyPosition[i],
        EarthxVelocity[i],EarthyVelocity) << "␣"
<< semimajoraxis(energy(EarthxPosition[i],
        EarthyPosition[i], EarthxVelocity[i],
        EarthyVelocity)) << "␣"
<< eccentricity(semimajoraxis(energy(EarthxPosition[i],
        EarthyPosition[i], EarthxVelocity[i],
        EarthyVelocity))) << "␣"
<< sqrt(Star2xPosition[i]*Star2xPosition[i] +
        Star2yPosition[i]*Star2yPosition[i]) << "␣"
<< sqrt(Star2xVelocity[i]*Star2xVelocity[i] +
        Star2yVelocity[i]*Star2yVelocity[i]) << "␣"
<< energy(Star2xPosition[i], Star2yPosition[i],
        Star2xVelocity[i],Star2yVelocity) << "␣"
<< angularmomentum(Star2xPosition[i], Star2yPosition[i],
        Star2xVelocity[i],Star2yVelocity) << "␣"
<< semimajoraxis(energy(Star2xPosition[i],
        Star2yPosition[i], Star2xVelocity[i],
        Star2yVelocity)) << "␣"
<< eccentricity(semimajoraxis(energy(Star2xPosition[i],
        Star2yPosition[i], Star2xVelocity[i],
        Star2yVelocity))) << "␣"
<<endl;
```

```
// Delete the elements in the array/
                delete [] Star1xPosition;
                delete [] Star1yPosition;
                delete [] Star1xVelocity;
                delete [] Star1yVelocity;
                delete [] EarthxPosition;
                delete [] EarthyPosition;
                delete [] EarthxVelocity;
                delete [] EarthyVelocity;
                delete [] Star2xPosition;
                delete [] Star2yPosition;
                delete [] Star2xVelocity;
                delete [] Star2yVelocity;
        }
        myfile.close();
} // end writing and storing to the file.

// Runge Kutta 4th order Function
        void RK (const int& N,
                const double& InitialTime, const double& FinalTime,
                const double& Star1X, const double& Star1Y,
                const double& Star1VX, const double& Star1VY,
                const double& EarthX, const double& EarthY,
                const double& EarthVX, const double& EarthVY,
                const double& Star2X, const double& Star2Y,
                const double& Star2VX, const double& Star2VY){

// Define the time step variables
        double dt, TS;

// Number of iterations:
        int i;

// Stepper variables for Star1's positions and velocities
        double X1S, Y1S, VX1S, VY1S;
```

```
// Stepper variables for Earth's positions and velocities
        double X2S, Y2S, VX2S, VY2S;

// Stepper variables for Star2's positions and velocities
        double X3S, Y3S, VX3S, VY3S;

// Time Step (Using 1 Earth Day)
        //dt = ((FinalTime - InitialTime)/(N- 1)) * 21600.0
        dt = 8.64e4;
        Time[0] = InitialTime;

// Storing the initial positions and velocities of each of the
// bodies in the 0th element spot.
        Star1xPosition[0] =  Star1X;
        Star1yPosition[0] =  Star1Y;
        Star1xVelocity[0] =  Star1VX;
        Star1yVelocity[0] =  Star1VY;

        EarthxPosition[0] = EarthX;
        EarthyPosition[0] = EarthY;
        EarthxVelocity[0] = EarthVX;
        EarthyVelocity[0] = EarthVY;

        Star2xPosition[0] = Star2X;
        Star2yPosition[0] = Star2Y;
        Star2xVelocity[0] = Star2VX;
        Star2yVelocity[0] = Star2VY;

        TS = InitialTime;
        X1S  = Star1X;
        Y1S  = Star1Y;
        VX1S = Star1VX;
        VY1S = Star1VY;

        X2S  = EarthX;
        Y2S  = EarthY;
```

```
        VX2S = EarthVX;
        VY2S = EarthVY;

        X3S  = Star2X;
        Y3S  = Star2Y;
        VX3S = Star2VX;
        VY3S = Star2VY;

// Begin for loop for iterating the positions and velocities of
// each body in the simulation
        for (i = 1; i < N; i++){

//Pass the parameters to the numerical method

                RK(TS, dt, X1S, Y1S, VX1S, VY1S,
                        X2S, Y2S, VX2S, VY2S,
                        X3S, Y3S, VX3S, VY3S);

                // Uncomment to use this method
                /*EulerCromer(TS, dt, X1S, Y1S, VX1S, VY1S,
                        X2S, Y2S, VX2S, VY2S,
                        X3S, Y3S, VX3S, VY3S);*/

                Time[i] = TS;
                Star1xPosition[i] = X1S;
                Star1yPosition[i] = Y1S;
                Star1xVelocity[i] = VX1S;
                Star1yVelocity[i] = VY1S;
                EarthxPosition[i] = X2S;
                EarthyPosition[i] = Y2S;
                EarthxVelocity[i] = VX2S;
                EarthyVelocity[i] = VY2S;
                Star2xPosition[i] = X3S;
                Star2yPosition[i] = Y3S;
                Star2xVelocity[i] = VX3S;
                Star2yVelocity[i] = VY3S;
```

```
        }// End for loop
}// End RK Function

// Begin Euler-Cromer Function
    // Uncomment to use this method
    /*void EulerCromer(double& t, const double& dt,
        double& x1, double& y1, double& vx1, double& vy1,
        double& x2, double& y2, double& vx2, double& vy2,
        double& x3, double& y3, double& vx3, double& vy3){

    double AccelXSun, AccelYSun, AccelXEarth, AccelYEarth,
        AccelXJupiter, AccelYJupiter;

    AccelXSun = SunxAcceleration(x1, y1, x3, y3);
    AccelYSun = SunyAcceleration(x1, y1, x3, y3);
    AccelXEarth = EarthxAcceleration(x1, y1, x2, y2, x3, y3);
    AccelYEarth = EarthyAcceleration(x1, y1, x2, y2, x3, y3);
    AccelXJupiter = JupiterxAcceleration(x1, y1, x3, y3);
    AccelYJupiter = JupiteryAcceleration(x1, y1, x3, y3);

    vx1 = vx1 + dt * AccelXSun;
    vy1 = vy1 + dt * AccelYSun;
    vx2 = vx2 + dt * AccelXEarth;
    vy2 = vy2 + dt * AccelYEarth;
    vx3 = vx3 + dt * AccelXJupiter;
    vy3 = vy3 + dt * AccelYJupiter;

    x1 = x1 + dt * vx1;
    y1 = y1 + dt * vy1;
    x2 = x2 + dt * vx2;
    y2 = y2 + dt * vy2;
    x3 = x3 + dt * vx3;
    y3 = y3 + dt * vy3;

    t = t + dt;
}// End Euler-Comer Method */
```

```
// Begin RK 4th order method
        void RKSTEP (double& t, const double& dt,
         double& x1, double& y1, double& vx1, double& vy1,
         double& x2, double& y2, double& vx2, double& vy2,
         double& x3, double& y3, double& vx3, double& vy3){

// For Star1, the S#vx or S#vy represents the coefficients
// for the acceleration and k#x or k#y represents the coefficients
// for the velocities.
        double S1x, S1y, S2x, S2y, S3x, S3y, S4x, S4y;
        double S1vx, S1vy, S2vx, S2vy, S3vx, S3vy, S4vx, S4vy;

// For Earth, the E#vx or E#vy represents the coefficients
// for the acceleration and k#x or k#y represents the
// coefficients for the velocities.
        double E1x, E1y, E2x, E2y, E3x, E3y, E4x, E4y;
        double E1vx, E1vy, E2vx, E2vy, E3vx, E3vy, E4vx, E4vy;

// For Star1, the k#vx or k#vy represents the coefficients
// for the acceleration and k#x or k#y represents the coefficients
// for the velocities.
        double J1x, J1y, J2x, J2y, J3x, J3y, J4x, J4y;
        double J1vx, J1vy, J2vx, J2vy, J3vx, J3vy, J4vx, J4vy;

        double h, h2, h6;

        h  = dt;
        h2 = 0.5 * h;
        h6 = (1.0/6.0) * h;

// These are the initial velocities that were inputted from the user.
// Note these are the initial velocities of Earth and Jupiter.
        S1x = vx1;
        S1y = vy1;
```

```
        E1x = vx2;
        E1y = vy2;

        J1x = vx3;
        J1y = vy3;

// 1st accelerations for Star1
        S1vx = Star1xAcceleration(x1, y1, x2, y2, x3, y3);
        S1vy = Star1yAcceleration(x1, y1, x2, y2, x3, y3);

// 1st accelerations for Earth
        E1vx = EarthxAcceleration(x1, y1, x2, y2, x3, y3);
        E1vy = EarthyAcceleration(x1, y1, x2, y2, x3, y3);

// 1st accelerations for Star2
        J1vx = Star2xAcceleration(x1, y1, x2, y2, x3, y3);
        J1vy = Star2yAcceleration(x1, y1, x2, y2, x3, y3);

// 2nd velocity for Star1
        S2x = vx1 + h2 * S1vx;
        S2y = vy1 + h2 * S1vy;

// 2nd velocity for Earth
        E2x = vx2 + h2 * E1vx;
        E2y = vy2 + h2 * E1vy;

//2nd velocity for Star2
    J2x = vx3 + h2 * J1vx;
    J2y = vy3 + h2 * J1vy;

//2nd Accelerations for Star1
        S2vx = Star1xAcceleration(t + h2, x1 + h2 * S1x, y1 + h2 * S1y,
                                  x2 + h2 * E1x, y2 + h2 * E1y,
                                  x3 + h2 * J1x, y3 + h2 * J1y);
        S2vy = Star1yAcceleration(t + h2, x1 + h2 * S1x, y1 + h2 * S1y,
                                  x2 + h2 * E1x, y2 + h2 * E1y,
```

```
                                    x3 + h2 * J1x, y3 + h2 * J1y);

// 2nd Accelerations for Earth
        E2vx = EarthxAcceleration(t + h2, x1 + h2 * S1x, y1 + h2 * S1y,
                                    x2 + h2 * E1x, y2 + h2 * E1y,
                                    x3 + h2 * J1x, y3 + h2 * J1y);
        E2vy = EarthyAcceleration(t + h2, x1 + h2 * S1x, y1 + h2 * S1y,
                            x2 + h2 * E1x, y2 + h2 * E1y,
                            x3 + h2 * J1x, y3 + h2 * J1y);

// 2nd Accelerations for Star2
        J2vx = Star2xAcceleration(t + h2, x1 + h2 * S1x, y1 + h2 * S1y,
                                    x2 + h2 * E1x, y2 + h2 * E1y,
                                    x3 + h2 * J1x, y3 + h2 * J1y);
        J2vy = Star2yAcceleration(t + h2, x1 + h2 * S1x, y1 + h2 * S1y,
                            x2 + h2 * E1x, y2 + h2 * E1y,
                            x3 + h2 * J1x, y3 + h2 * J1y);

// 3rd velocity for Star1
        S3x = vx1 + h2 * S2vx;
        S3y = vy1 + h2 * S2vy;

// 3rd velocity for Earth
        E3x = vx2 + h2 * E2vx;
        E3y = vy2 + h2 * E2vy;

// 3rd velocity for Star2
        J3x = vx3 + h2 * J2vx;
        J3y = vy3 + h2 * J2vy;

// 3rd acceleration for Star1
        S3vx = Star1xAcceleration(t + h2, x1 + h2 * S2x, y1 + h2 * S2y,
                                    x2 + h2 * E2x, y2 + h2 * E2y,
                                    x3 + h2 * J2x, y3 + h2 * J2y);
        S3vy = Star1yAcceleration(t + h2, x1 + h2 * S2x, y1 + h2 * S2y,
                                    x2 + h2 * E2x, y2 + h2 * E2y,
```

```
                                        x3 + h2 * J2x, y3 + h2 * J2y);

// 3rd acceleration for Earth
        E3vx = EarthxAcceleration(t + h2, x1 + h2 * S2x, y1 + h2 * S2y,
                                        x2 + h2 * E2x, y2 + h2 * E2y,
                                        x3 + h2 * J2x, y3 + h2 * J2y);
        E3vy = EarthyAcceleration(t + h2, x1 + h2 * S2x, y1 + h2 * S2y,
                                x2 + h2 * E2x, y2 + h2 * E2y,
                                x3 + h2 * J2x, y3 + h2 * J2y);

// 3rd acceleration for Star2
        J3vx = Star2xAcceleration(t + h2, x1 + h2 * S2x, y1 + h2 * S2y,
                                x2 + h2 * E2x, y2 + h2 * E2y,
                                x3 + h2 * J2x, y3 + h2 * J2y);
        J3vy = Star2yAcceleration(t + h2, x1 + h2 * S2x, y1 + h2 * S2y,
                                x2 + h2 * E2x, y2 + h2 * E2y,
                                x3 + h2 * J2x, y3 + h2 * J2y);

// 4th velocities for Star1
        S4x = vx1 + h * S3vx;
        S4y = vy1 + h * S3vy;

// 4th velocities for Earth
        E4x = vx2 + h * E3vx;
        E4y = vy2 + h * E3vy;

// 4th velocities for Star2
        J4x = vx3 + h * J3vx;
        J4y = vy3 + h * J3vy;

// 4th acceleration term for Star1
        S4vx = Star1xAcceleration(t + h2, x1 + h2 * S3x, y1 + h2 * S3y,
                                        x2 + h2 * E3x, y2 + h2 * E3y,
                                        x3 + h2 * J3x, y3 + h2 * J3y);
        S4vy = Star1yAcceleration(t + h2, x1 + h2 * S3x, y1 + h2 * S3y,
                                        x2 + h2 * E3x, y2 + h2 * E3y,
```

```
                                          x3 + h2 * J3x, y3 + h2 * J3y);

// 4th acceleration term for Earth
        E4vx = EarthxAcceleration(t + h2, x1 + h2 * S3x, y1 + h2 * S3y,
                                          x2 + h2 * E3x, y2 + h2 * E3y,
                                          x3 + h2 * J3x, y3 + h2 * J3y);
        E4vy = EarthyAcceleration(t + h2, x1 + h2 * S3x, y1 + h2 * S3y,
                                          x2 + h2 * E3x, y2 + h2 * E3y,
                                          x3 + h2 * J3x, y3 + h2 * J3y);

// 4th acceleration term for Star2
        J4vx = Star2xAcceleration(t + h2, x1 + h2 * S3x, y1 + h2 * S3y,
                                          x2 + h2 * E3x, y2 + h2 * E3y,
                                          x3 + h2 * J3x, y3 + h2 * J3y);
        J4vy = Star2yAcceleration(t + h2, x1 + h2 * S3x, y1 + h2 * S3y,
                                          x2 + h2 * E3x, y2 + h2 * E3y,
                                          x3 + h2 * J3x, y3 + h2 * J3y);

// The new positions and velocities are dependent of the time step,
// current position, and velocity as well as the
// E#x, E#y, E#vx, E#vy,M#x, M#y, M#vx, M#vy,J#x, J#y, J#vx, & J#vy
// parameters.

    t = t + h;

    //new Star1 x and y positions
    x1 = x1 + h6 * (S1x + 2.0 * (S2x + S3x) + S4x);
    y1 = y1 + h6 * (S1y + 2.0 * (S2y + S3y) + S4y);

    //new Star1 x and y velocities
    vx1 = vx1 + h6 * (S1vx + 2.0 * (S2vx + S3vx) + S4vx);
    vy1 = vy1 + h6 * (S1vy + 2.0 * (S2vy + S3vy) + S4vy);

    //new Earth x and y positions
    x2 = x2 + h6 * (E1x + 2.0 * (E2x + E3x) + E4x);
    y2 = y2 + h6 * (E1y + 2.0 * (E2y + E3y) + E4y);
```

```
    //new Earth x and y velocities
    vx2 = vx2 + h6 * (E1vx + 2.0 * (E2vx + E3vx) + E4vx);
    vy2 = vy2 + h6 * (E1vy + 2.0 * (E2vy + E3vy) + E4vy);

    //new Star2 x and y positions
    x3 = x3 + h6 * (J1x + 2.0 * (J2x + J3x) + J4x);
    y3 = y3 + h6 * (J1y + 2.0 * (J2y + J3y) + J4y);

    //new Star2 x and y velocities
    vx3 = vx3 + h6 * (J1vx + 2.0 * (J2vx + J3vx) + J4vx);
    vy3 = vy3 + h6 * (J1vy + 2.0 * (J2vy + J3vy) + J4vy);
} // End RK Algorithm

// Begin Acceleration Functions for Stars and Earth
// x1 and y1 are the positions of Star1,
// x2 and y2 are the positions of Earth,
// x3 and y3 are the positions of Star2.

// Returning the x acceleration of the Star1:
double Star1xAcceleration (const double& x1, const double& y1,
                const double& x3, const double& y3){

    double G = 6.67e-11;
    double Star1Mass = 1.99e30;
    double Star2Mass = 1.90e27;

// normSJ is the norm distance between Star 1 and 2.
    double normSJ = sqrt((x1 - x3) * (x1 - x3) + (y1 - y3) * (y1 - y3));

    double RxStar1 = - G * x1 * (Star1Mass + Star2Mass) /
                                    (normSJ * normSJ * normSJ);
    return RxStar1;
}

// Returning the y acceleration of the Star1:
```

```
double Star1yAcceleration (const double& x1, const double& y1,
                const double& x3, const double& y3){

    double G = 6.67e-11;
    double Star1Mass = 1.99e30;
    double Star2Mass = 1.90e27;

    double normSJ = sqrt((x1 - x3) * (x1 - x3) + (y1 - y3) * (y1 - y3));

    double RyStar1 = - G * y1 * (SunMass + JupiterMass) /
                                (normSJ * normSJ * normSJ);
    return RyStar1;
}

// Returning the x acceleration of the Earth
double EarthxAcceleration (const double& x1, const double& y1,
                           const double& x2, const double& y2,
                           const double& x3, const double& y3){

    double G = 6.67e-11;
    double Star1Mass = 1.99e30;
    double Star2Mass = 1.90e27;

// normES and NormEJ are the distances norms between the
// Earth and Star 1 and between Earth and Star2.
    double normES = sqrt((x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2));
    double normEJ = sqrt((x3 - x2) * (x3 - x2) + (y3 - y2) * (y3 - y2));

    double RxEarth = (G * SunMass * (x1 - x2) /
                (normES * normES * normES)) +
                (G * JupiterMass * (x3 - x2)/
                (normEJ * normEJ * normEJ));
    return RxEarth;
}

//Returning the y acceleration of the Earth.
```

```
double EarthyAcceleration (const double& x1, const double& y1,
                        const double& x2, const double& y2,
                        const double& x3, const double& y3){

        double G = 6.67e-11;
        double Star1Mass = 1.99e30;
    double Star2Mass = 1.90e27;

    double normES = sqrt((x1 - x2) * (x1 - x2) + (y1 - y2)* (y1 - y2));
    double normEJ = sqrt((x3 - x2) * (x3 - x2) + (y3 - y2)* (y3 - y2));

    double RyEarth = (G * SunMass * (y1 - y2) /
                        (normES * normES * normES)) +
                        (G * JupiterMass * (y3 - y2)/
                        (normEJ * normEJ * normEJ));
    return RyEarth;

}

//Returning the x acceleration of Star2.
double Star2xAcceleration (const double& x1, const double& y1,
                        const double& x3, const double& y3){

        double G = 6.67e-11;
    double Star1Mass = 1.99e30;
    double Star2Mass = 1.90e27;

// normJS is the distance norm between the stars
    double normJS = sqrt((x1 - x3) * (x1 - x3) + (y1 - y3) * (y1 - y3));

    double RxStar2 = - G * x3 * (SunMass + JupiterMass) /
                        (normJS * normJS * normJS);
    return RxStar2;
}

//Returning the y acceleration of Jupiter.
```

```
double Star2yAcceleration (const double& x1, const double& y1,
                        const double& x3, const double& y3){

        double G = 6.67e-11;
    double Star1Mass = 1.99e30;
    double Star2Mass = 1.90e27;

    double normJS = sqrt((x1 - x3) * (x1 - x3) + (y1 - y3) * (y1 - y3));

    double RyStar2 = - G * y3 * (SunMass + JupiterMass) /
                        (normJS * normJS * normJS);
    return RyStar2;
} \\ End acceleration Functions

\\ Begin Integrals of TBP

double energy(const double& x, const double& y,
                        const double& vx, const double& vy){

        double G = 6.67408e-11;
        double Star1Mass  = 1.989e30;
        double PlanetMass = 5.97e24;

        double mu = G * (SunMass + PlanetMass);
        double r = sqrt(x1 * x1 + x2 * x2);

        return (0.5 * (v1 * v1 + v2 * v2) - (mu/ r));
}

double angularmomentum(const double& x, const double& y,
                        const double& v1, const double & v2){

        return ( x1 * v2 - x2 * v1);
}

double eccentricity(const double & A ){
```

```
// For circular orbits, the semi-latus is equal to the
// semi-major axis ( p = a ). So p = h^2 / a which means p = a

        double G = 6.67408e-11;
        double SunMass  = 1.989e30;
        double PlanetMass = 5.97e24;
        double mu = G * (SunMass + PlanetMass);

    double h = sqrt(A * mu);
    double x = mu * A;

    double ecc = sqrt(1 - h * h / (mu * A));
    return ecc;
}

// Begin Extra Functions Used
double semimajor(const double & C){

        double G = 6.67408e-11;
        double Star1Mass  = 1.989e30;
        double PlanetMass = 5.97e24;

        double mu = G * (Star1Mass + PlanetMass);
        return - mu/ (2.0 * C);
}

double circularvelocity(const double & x){

        double G = 6.67e-11;
        double Star1Mass = 1.99e30;
                if (x == 0){return 0;}
                 else{return sqrt(G * Star1Mass / x);}
}

double SunVelocity(const double& x1, const double& y1,
                        const double& x3, const double& y3){
```

```
        double G = 6.67e-11;
        double Star2Mass = 1.90e27;

    double normSJ = sqrt((x1 - x3) * (x1 - x3) +
                                (y1 - y3) * (y1 - y3));
    return sqrt(G * Star2Mass * (-x1) / (normSJ * normSJ));

}
// End Functions
// End Program
```