SELECTED STUDIES OF ORBITAL STABILITY AND HABITABILITY IN STAR-PLANET SYSTEMS

by

WILLIAM JASON EBERLE

Presented to the Faculty of the Graduate School of
The University of Texas at Arlington in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2010

I dedicate this to my daughter Astra Chalisa Eberle.

ACKNOWLEDGEMENTS

I would like to thank my supervising professor Dr. Manfred Cuntz for constantly motivating and encouraging me, and also for his invaluable advice during the course of my studies. I wish to thank Dr. Zdzislaw Musielak, Dr. Ramon Lopez, Dr. Alex Weiss and Dr. Qiming Zhang for their interest in my research and for taking time to serve in my committee.

I would also like to express my deep gratitude to my parents, Kip and Darlene Eberle, without whom I wouldn't even be alive. They have motivated me and made many sacrifices to provide me with opportunities that they didn't have. I am also extremely grateful to my wife Sarinya for her encouragement and patience. I also want to thank my sisters Sandy and Nikki for their perspectives that have helped to develop mine.

Finally, I want to acknowledge all of my friends and acquaintances that I have known throughout my education especially Travis Collavo for his invaluable comments which improved this work.

November 15, 2010

ABSTRACT

SELECTED STUDIES OF ORBITAL STABILITY AND HABITABILITY IN STAR-PLANET SYSTEMS

William Jason Eberle, Ph.D.

The University of Texas at Arlington, 2010

Supervising Professor: Manfred Cuntz

The study of planets remaining in orbit around one star with another star interfering is an important topic of orbital mechanics and astrobiology. The onset of instability for a planet which is part of a stellar binary system is investigated by performing numerical simulations of the circular restricted three body problem. Synodic velocity phase space portraits are used with the hodograph in order to obtain a new criterion for determining planetary stability. The stability limits obtained via these methods agree with previous methods that were obtained based on simulations over much longer time intervals. In addition, the timely case of a recently proposed and controversial planet candidate in the binary system ν Octantis is addressed. Finally, the issue of habitability where a giant planet resides in the habitable zone is investigated for the specific case of HD 23079.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

BACKGROUND INFORMATION

The possibility that planets exist outside our solar system has been a topic of speculation since at least the Copernican revolution if not earlier. After all, since the Sun is the center of our planetary system, why shouldn't the other stars in the sky have planets orbiting them too? This was essentially the view put forward by an Italian Dominican friar, philosopher, mathematician and astronomer Giordano Bruno in the sixteenth century before he was burned at the stake in the year 1600, albeit mostly for his religious opinions. In addition, Issac Newton made a comment in his *General Scholium* which was first appended to the second edition of his *Principia*: "And if the fixed Stars are the centers of other like systems, these, being form'd by the like wise counsel, must be all subject to the dominion of One..."[1] although here too is seems to be a matter related more to his religious beliefs than anything else.

Works of science fiction involving extraordinary voyages to the Moon started to appear in the seventeenth century and when rapid improvements to telescopes in the nineteenth century began to make surface features on Mars possible to distinguish, Mars became a popular setting for fictional adventures and peoples. As more information became available however, the prospects for life in the solar system other than on Earth became less and less likely and so almost the only refuge remaining for a science fiction author in the twentieth century was extrasolar planets. The idea that there could be planets orbiting around other stars became so common after the mid-twentieth century that when actual planets were discovered it was exciting but

[1] `http://www.isaacnewton.ca/gen_scholium/scholium.htm`

hardly surprising. What was surprising however was how different these planetary systems are from our own.

1.1 Observation of Extrasolar Planets

Campbell et al. (1988) suggested that a planet orbited the star γ Cephei but this was controversial and would not be confirmed until 2002 (Hatzes et al. 2003). The first widely accepted discovery of an exoplanet was around the pulsar PSR 1257+12 (Wolszczan and Frail 1992). Mayor and Queloz (1995) discovered the first exoplanet around an ordinary main-sequence star (51 Pegasi) and subsequently ushered in the era of exoplanetary discovery that continues to this day. By now, several hundred planets have been found, many in multiple planet systems. In addition, several proto-planetary disks have been discovered which will shed light on planet formation processes.

1.1.1 Detection Methods

Very few exoplanets have been imaged directly and those that have are quite large and far removed from the star they orbit - they would be washed out in the glare of the star otherwise. Thus the majority of exoplanet discoveries have been via indirect methods. Of the various methods used so far to find extrasolar planets the most successful to date has been the radial velocity method. The method works on the principle that a planet orbiting a star will pull the star ever so slightly back and forth and thus a wobble of the star along our line of sight will result in a periodic doppler shifting of the spectral lines of the star. When the star is moving towards us, the spectral lines will be shifted to shorter wavelengths and while moving away to longer wavelengths. This method has become so sophisticated that an oscillation of down to about 1 m/s can be detected.

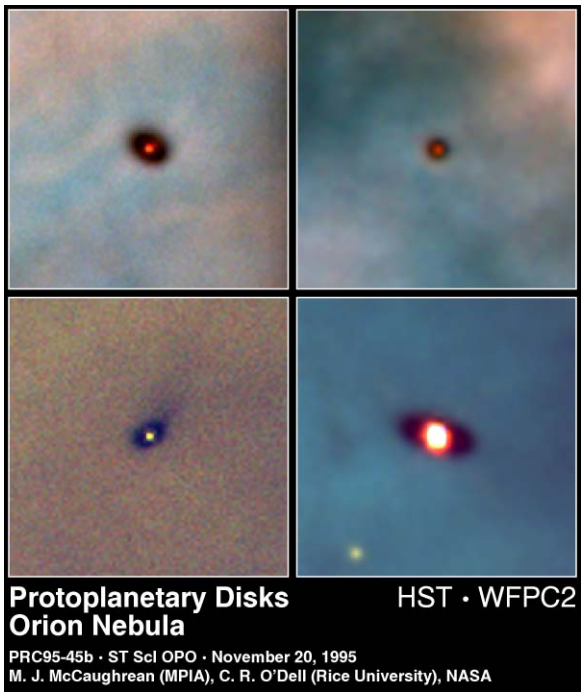


Figure 1.1. Examples of discovered protoplanetary disks.

The second most productive method to date is the transit method, which works by trying to find periodic dips in the luminosity of the star that occur when a planet passes in front of the stellar disk. The amount that the star's luminosity diminishes is dependent on the size of the planet and how far away it is from the star. This method is likely going to become much more productive very soon because of the Kepler space telescope which is simultaneously measuring the brightness of approximately 100,000 stars in the vicinity of Cygnus over a period of 4 years. In order for a transit to be observed, the orbital plane of the planet must be aligned with our line of sight and for an Earth-like orbit the likelihood is about 0.5%. For giant planets in orbits that span less than a week the likelihood is closer to 10%.

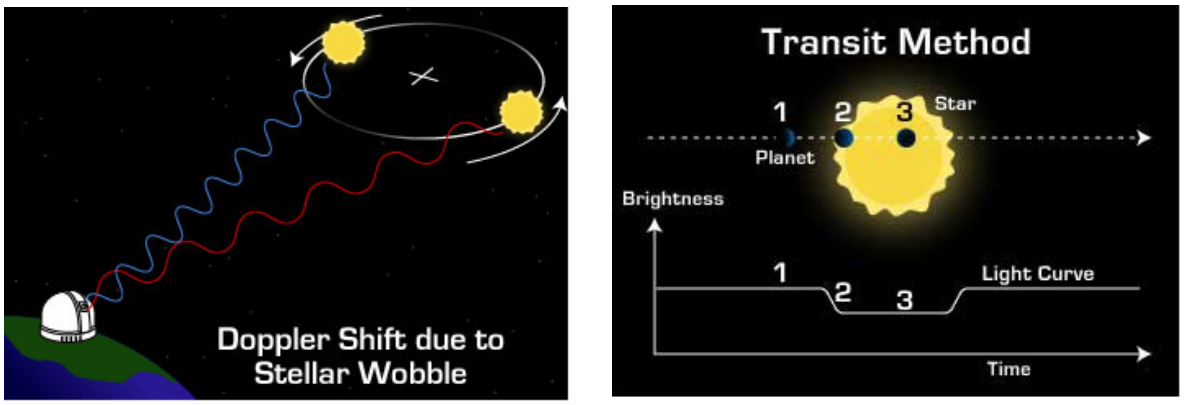


Figure 1.2. Illustrations of the two most productive exoplanet detection methods.

1.1.2 General Properties of Exoplanets

A significant fraction of the stars that have been observed as part of planet search programs have been found to have planets orbiting them. The overall distribution of planets remains uncertain however because the radial velocity method and the transit method both suffer from selection effects that favor systems where the orbital plane of the planets lies nearly along our line of sight and large planets that are close to their parent star. In fact many of the first exoplanets discovered were so called "hot Jupiters": planets of roughly Jupiter mass in orbits with periods that are only a few days. Since no Earth-like planets have been discovered yet, other than pulsar planets, it is difficult to speculate on how common they are. Assuming that the mass distribution of planets follows a power law, it could be suggested that lower mass planets are more common than large mass planets. It also appears that planets on large orbits may be more common than ones on small orbits. Based on such extrapolation, it is estimated that perhaps 20% of sunlike stars have at least one giant planet while at least 40 % may have planets of lower mass (Cumming et al. 2008).

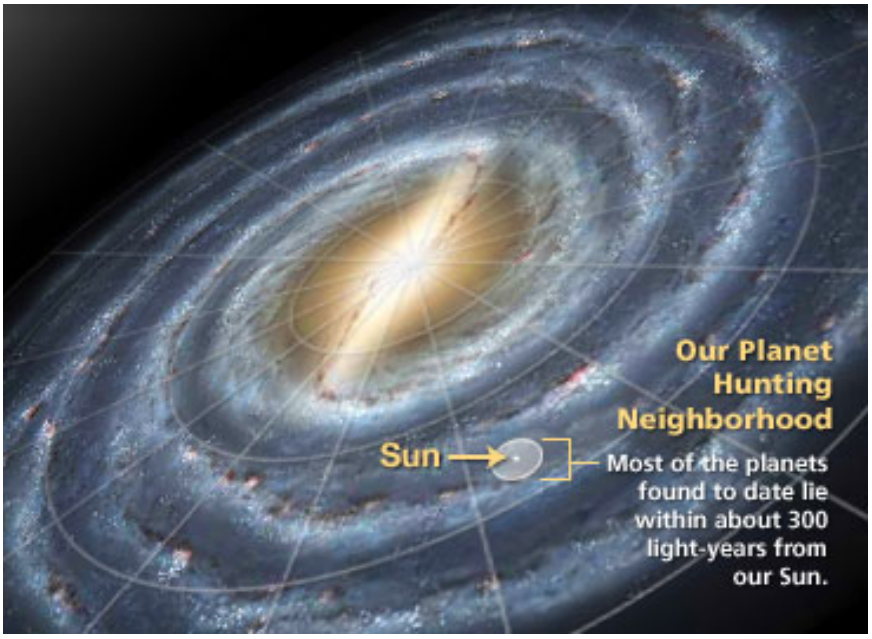


Figure 1.3. Exoplanet search neighborhood. Most of the discovered exoplanets are within 300 lightyears of the Solar System.

1.1.3 Planets in Binary Systems

In the past few years the study of planets in stellar binary systems has reached a heightened level of intensity owing to significant new developments in both observations and theory (e.g., Jones 2008). Observations have shown that planets are able to exist in relatively close binary systems with separation distances of 20 to 25 AU, a result presented by Patience et al. (2002) and Eggenberger et al. (2004). In addition, theoretical simulations have been performed on planet formation in binary systems, which have led to highly favorable outcomes; see, e.g., Kley (2001), Quintana et al. (2002), and subsequent studies. This type of research is also motivated by the ongoing quest to find new extra-solar planets and to analyze their orbital stability, noting that stellar binary systems occur in high frequency in the local Galactic neighborhood (Duquennoy and Mayor 1991; Lada 2006; Raghavan et al. 2006).

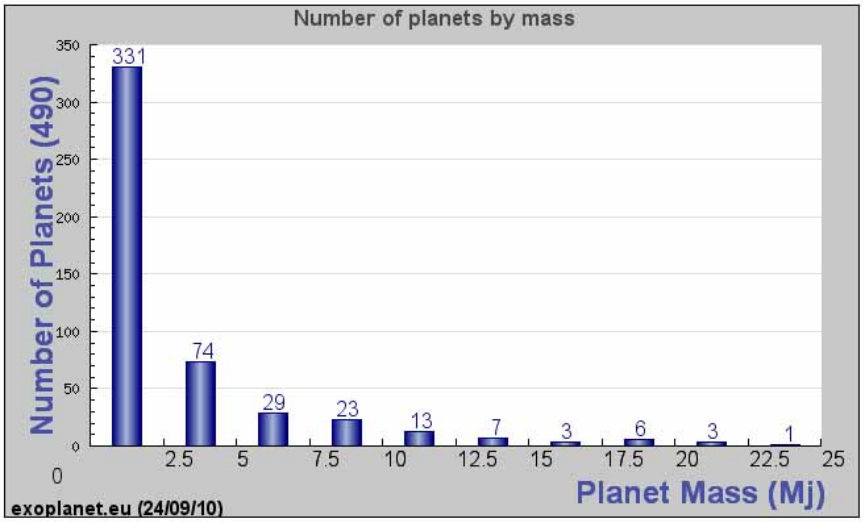


Figure 1.4. Distribution of exoplanet candidates.

1.2 Planetary Orbital Stability

Recent discoveries of planets orbiting stars other than the Sun including planets orbiting stars in multiple star systems has rekindled interest in a centuries old problem: *The Three Body Problem*. Beginning with Sir Isaac Newton's *Philosophiæ Naturalis Principia Mathematica* in 1687 the problem of finding the motion of a set of bodies numbering more than two moving under their mutual gravitational attraction has been difficult to say the least. Generally this is known as the gravitational N-body problem even up to the most extreme cases when the number of bodies is quite large such as the stars in a galaxy for example. These "large N" type problems have been treated quite successfully in recent years using hierarchical methods (Barnes and Hut 1986; Aarseth 2003). On a much smaller scale when the number of bodies is small, less than a dozen or so the problem can be treated efficiently with direct integration of each body.

Since the two body problem is fairly straight forward to solve, it may seem that by adding one more body the overall behavior of the system would not change very

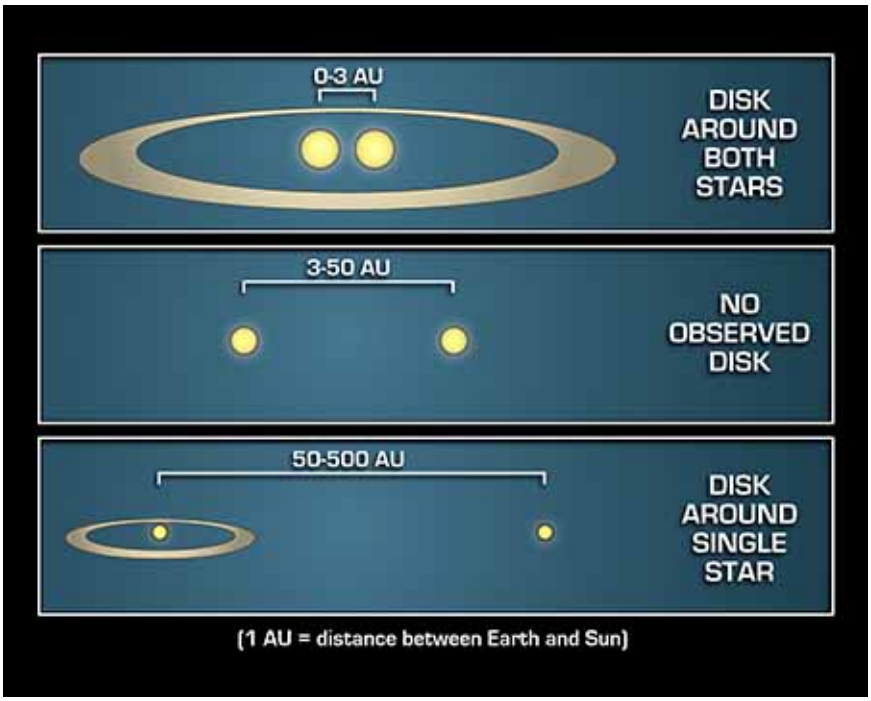


Figure 1.5. Debris disks around binary systems.

much. Numerous approximations and assumptions have been made over the years and have provided great insights, but a complete and practical solution that holds under any circumstances remains elusive.

The assumptions that are relevant to this work are the following:

1. There are only three constant masses and one is small enough that it does not significantly affect the motion of the other two. Further, the masses will be named in order of decreasing mass e.g. M_1, M_2, and M_3.
2. The motion of all of the masses is slow enough that the system is non-relativistic.
3. As a consequence of the previous assumptions, the motion of the two largest masses must be a solution of the two body problem.
4. The motion of the smallest mass lies within the same plane as the two largest masses.

The first three assumptions essentially limit the discussion to the so called restricted three body problem. The last assumption further limits the discussion to the planar restricted three body problem, obviously the full three dimensional problem would be more robust, but a strategic decision was made to limit the domain of study. The problem can be further broken up into subcategories. First, the motion of the two largest masses will be considered as either circular or elliptical since the other two possible options do not result in periodic orbits and are of limited interest. Second, the ratio of the second mass to the sum of it and the first mass ($\mu = \frac{M_2}{M_1+M_2}$) will define two paradigms: $0 < \mu < 0.5$ in which two stars orbit each other and $\mu \approx 10^{-3}$ in which a giant planet similar to Jupiter orbits a star with mass comparable to the Sun. In both of these cases a third much smaller mass orbits primarily around one of the two larger masses. In the case of the binary star system, the smallest mass will be referred to as a planet orbiting the more massive star, whereas in the case of the giant planet system, the smallest mass will be referred to as a moon of the giant planet.

The primary question is: under what circumstances will a planet or moon remain stable under the dynamical gravitational influence of its host stars and giant planets? For the case of the planet in the binary system, a range of mass ratios for the stars as well as a range of initial conditions for the planet will attempt to address this question in as general a way as possible.

1.3 Stellar Habitable Zones

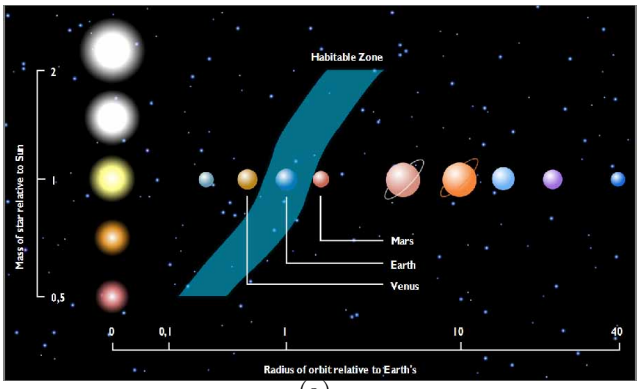
The concept of the stellar habitable zone is based on the assumption that liquid water is a prerequisite for life. The conditions necessary for liquid water to exist on the surface of a planet are dependent on a number of factors such as the spectral class of the star, the size of the planet as well as its orbit, the make up of its atmosphere

and so on. Since we are only familiar with the conditions that support life on Earth, the common assumptions that are used to estimate the range of orbits for a given star correspond to an Earth-like planet with a $CO_2/H_2O/N_2$ atmosphere.

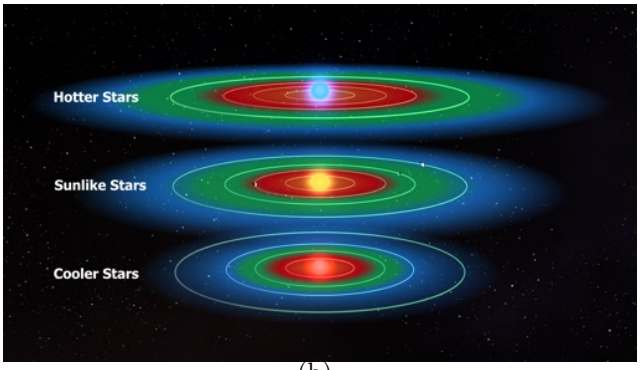
The inner limit of habitability is set by the loss of water from the upper planetary atmosphere through photodissociation and subsequent escape of hydrogen to space associated with a run-away greenhouse effect. The outer limit of habitability is given by the maximum greenhouse effect (Kasting et al. 1993; Underwood et al. 2003); here a surface temperature of 273 K can be maintained by a cloud-free CO_2 atmosphere. In our solar system for example, with these assumptions, the inner and outer edges of the habitable zone are assumed to be located at 0.84 and 1.67 AU from the Sun respectively.

1.4 Projects

The parameter space for investigating three body dynamics is quite large and due to time constraints a strategic decision was made to study the problem through three smaller sub problems. The first of these which is a study of the circular restricted three body problem which introduces a new method of estimating the limits of orbital stability using the hodograph to find the median effective eccentricity is covered in Chapter 2. The second study in Chapter 3 concerns an unexpected oscillation in the radial velocity curve in the ν Octantis binary system in addition to the oscillation of the star's orbit around the other star. If the additional oscillation is in fact due to a planet orbiting one of the stars, the most likely explanation is that the planet is orbiting the primary star in a retrograde orbit. The last project explores the possibility of a trojan type "moon" of the planet HD 23079b which remains within the habitable zone for an extended period of time is covered in Chapter 4.



(a)



(b)

Figure 1.6. The location and extents of the habitable zone is dependent on the mass of the star. (a) Our solar system and its habitable zone with other stars and the extents of their habitable zone for comparison. Note that the horizontal scale is logarithmic. (b) The habitable zones for hot, Sun-like and cool stars. Hot stars have larger habitable zones at larger distances compared to Sun-like stars while cooler stars have a smaller habitable zone at smaller distances.

CHAPTER 2

PROJECT ONE: ADVANCES IN THE CIRCULAR RESTRICTED 3 BODY PROBLEM (CR3BP)

2.1 Introduction

A planet's orbit in a binary system is not always stable. If the planet orbits too far away from one star, the perturbation from the other star could be enough to catastrophically destabilize the planet's orbit. In previous work, Musielak et al. (2005) investigated the stability of both S-type and P-type orbits in stellar binary systems and deduced orbital stability limits for planets. P-type orbits lie well outside the binary system, where the planet essentially orbits the center of mass of both stars, whereas S-type orbits lie near one of the stars, with the second star exerting perturbational influence. The limits of stability were found to depend on the mass ratio between the stellar components.

2.2 Previous Numerical Results

Stability limits for S-type and P-type orbits were found by Holman and Wiegert (1999), who considered orbital simulations for eccentricities of the binary components in the range $0.0 \leq e \leq 0.7$ and mass ratios in the range $0.1 \leq \mu \leq 0.9$ though for the P-type orbits the mass ratio was only taken up to 0.5 since the system would behave the same with the exception of a 180° phase change in the binary stars. Their approach which was first used by Dvorak (1986) was to simultaneously simulate eight massless particles in initially circular prograde orbits at evenly spaced orbital longitudes in the plane of the binary system. The largest stable orbit for which all of the massless

particles survived for the full 10,000 year simulation period was taken as the critical semi-major axis. For S-type orbits the critical semi-major axis was given as

$$\begin{aligned} a_c = [&(0.464 \pm 0.006) + (-0.380 \pm 0.010)\mu \\ &+(-0.631 \pm 0.034)e + (0.586 \pm 0.061)\mu e \\ &+(0.150 \pm 0.041)e^2 + (-0.198 \pm 0.074)\mu e^2]a_b \end{aligned} \tag{2.1}$$

where a_b is the binary semi-major axis. The Holman and Wiegert critical semi-major axis agrees well with both the limit identified by Rabl and Dvorak (1988) for equal mass stars, i.e. $\mu = 0.5$, and Musielak et al. (2005) for circular orbits i.e. e = 0.0. More recently, Mudryk and Wu (2006) concluded that overlapping subresonances of the same mean motion resonance are the dynamical cause of planetary instabilities in the planar restricted three body problem and that these occur for essentially the same conditions that formed the instability boundary (Eq. 2.1) and coincidentally correspond to the Hill criterion as well.

2.3 The Circular Restricted Three Body Problem

The circular restricted 3-body problem (CR3BP) is a well-established topic of celestial mechanics (Szebehely 1967; Roy 2005, and references therein). The CR3BP describes the motion of a body of negligible[1] mass moving in the gravitational field of two massive primaries. The primaries move on circular orbits around their center of mass and their motion is not influenced by the third body, the planet. Additionally, the initial velocity of the planet is usually assumed in the same direction as the orbital velocity of its host star, which is typically the more massive of the two stars. Research

[1]Negligible mass means that although the body's motion is influenced by the gravity of the two massive primaries, its mass is too low to affect the motions of the primaries.

results have also been given for more general orbital problems, including the elliptical restricted 3-body problem (ER3BP), which has been studied by Dvorak (1984), Dvorak (1986), Rabl and Dvorak (1988), and more recently by Pilat-Lohinger and Dvorak (2002) and Szenkovits and Makó (2008). Both Dvorak (1986) and Pilat-Lohinger and Dvorak (2002) investigated the stability domain for S-type orbits, whereas Szenkovits and Makó (2008) presented a sophisticated analytical study.

In the case of the CR3BP, the stability limits previously obtained have recently been updated by Cuntz et al. (2007) and Eberle et al. (2008) [Paper I] based on an analytical approach. Cuntz et al. and Eberle et al. were able to derive a stringent (sufficient) criterion for orbital stability based on Jacobi's constant and Jacobi's integral, akin to the criterion of "Hill stability". It was found that the planetary orbital stability was guaranteed only if its initial starting position is below a well-defined limit $\rho_0^{(1)}$, which can also be found by inspecting the "zero-velocity contour" in the synodic coordinate system. The orbital stability of a planet is ensured if the zero-velocity contour is closed around the star it orbits, which limits the allowable region of the planet as dictated by its available kinetic energy.

It is noteworthy that the property of the planet to remain within the zero-velocity contour is obviously tantamount to orbital stability, although it does not necessarily imply stability in the sense of quasi-periodicity. In fact, Eberle et al. (2008) encountered evidence that quasi-periodic orbits may even occur for planets outside the previously established stability region. This evidence is a compelling reason to revisit the potential of different approaches for determining the onset of orbital instability.

2.4 Theoretical Approach

2.4.1 System Setup with Definitions and Basic Equations

Consider a binary system with the mass $M = M_1 + M_2$, where M_1 and M_2 are the masses of the primary and secondary star, respectively. The mass of the planet is assumed to be very small ("negligible mass") compared to the stellar masses and the two stars are assumed to orbit each other in circular orbits (CR3BP). In general, the two stars can move relative to one another in any form of a conic section; for example, ellipse, parabola or hyperbola. Only elliptical paths are closed and are considered as orbits. The circular orbit is a special case of elliptical orbits and is immediately considered in the following. The origin of the coordinate system is chosen at the

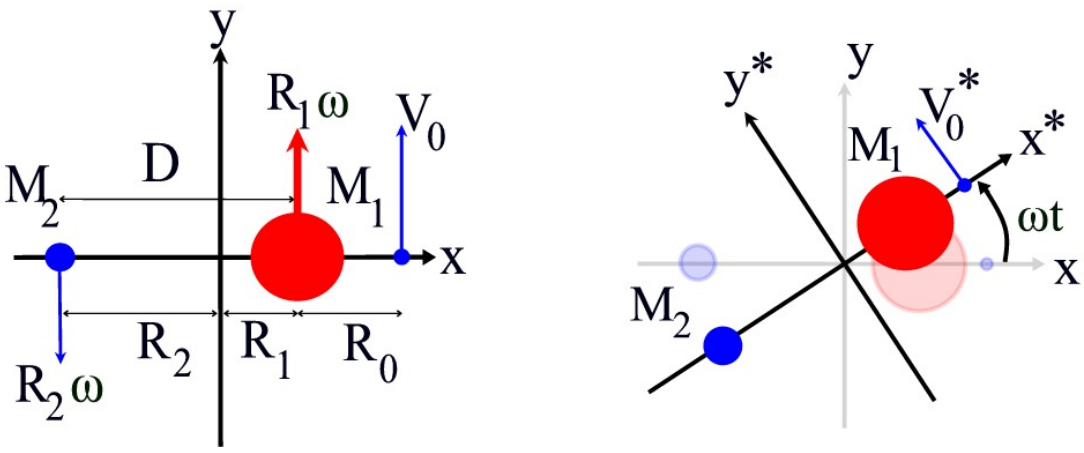


Figure 2.1. Basic setup used for circular restricted three body problem. (a) Initial conditions in sidereal coordinates. (b) Relationship to synodic coordinates.

center of mass of the two binary stars. Therefore, we find

$$M_1 R_1 = M_2 R_2 \tag{2.2}$$

with index 1 referring to the star of greater or equal mass, placed to the right, and index 2 referring to the star placed to the left in all forthcoming figures. All quantities are assumed to have their usual meaning, unless noted otherwise. The basic setup is depicted in Fig. 2.1. The motion of the binary stars is taken to be circular so by equating the gravitational force to the centripetal force on each star we have

$$\frac{GM_1M_2}{D^2} = \frac{M_1{V_1}^2}{R_1} = \frac{M_2{V_2}^2}{R_2} \tag{2.3}$$

where $D = R_1 + R_2$. Since the velocity for circular motion can be expressed as $V_i = R_i\omega$ for $i = 1, 2$ with ω as the angular velocity of the system, and V_i as velocity of star i relative to the center of mass, we can find:

$$V_i^2 \;=\; R_i\frac{GM_j}{D^2} \;=\; R_i^2\omega^2 \tag{2.4}$$

with $j = 3 - i$. Or

$$\omega^2 = \frac{M_j}{R_i}\frac{G}{D^2} \tag{2.5}$$

Now then, because of (2.2):

$$M_1 + M_2 = M_j\left(1 + \frac{R_j}{R_i}\right) \tag{2.6}$$

It is then easy to show that:

$$\frac{M_j}{R_i} = \frac{M_1 + M_2}{D} = \frac{M}{D} \tag{2.7}$$

Substituting this back into equation (2.5); we obtain

$$\omega^2 = \frac{GM}{D^3} \tag{2.8}$$

which is Kepler's third law.

The relative size of the two stars in the binary system has a significant affect on the regions of stability for the planet. There are several ways to quantify this mass

ratio such as taking the smaller mass divided by the larger mass. We will however define the mass ratio as the mass of the smaller star divided by the total mass of both stars; $\mu = \frac{M_2}{M}$. With $\alpha = 1 - \mu$ we obtain the mass and position of the two stars in terms of the mass ratio, the total mass of the binary stars, and the distance between them:

$$M_1 = \alpha M \qquad R_1 = \mu D \qquad M_2 = \mu M \qquad R_2 = \alpha D$$

2.4.2 Equations of Motion in Fixed (Sidereal) Coordinates

The following derivation of the equations of motion loosely follows chapter one from Szebehely (1967). Similar information can be found in Symon (1971), and Roy (2005). Since we are assuming the binary stars have circular motion, the positions of star 1 and 2, as functions of time t, denoted as $X_1(t)$, $Y_1(t)$, $X_2(t)$ and $Y_2(t)$, are given as

$$\begin{aligned} X_1(t) &= \mu D \cos \omega t \quad \text{and} \quad Y_1(t) = \mu D \sin \omega t \\ X_2(t) &= -\alpha D \cos \omega t \quad \text{and} \quad Y_2(t) = -\alpha D \sin \omega t \end{aligned} \tag{2.9}$$

We are also considering a small planet with mass m_3 subject to the gravity of both stars. The distance of the small planet to star $i = 1, 2$ is given as $r_i(t) = \sqrt{(X_3(t) - X_i(t))^2 + (Y_3(t) - Y_i(t))^2}$, with X_3 and Y_3 as (sidereal) coordinates of the small planet. The potential energy of the small planet is given as

$$U_3 = -\frac{GM_1 m_3}{r_1} - \frac{GM_2 m_3}{r_2}. \tag{2.10}$$

The planet's acceleration vector $\boldsymbol{a_3}$ is

$$\boldsymbol{a_3} = \frac{1}{m_3}\boldsymbol{F_3} = -\frac{1}{m_3}\boldsymbol{\nabla_3} U_3 \tag{2.11}$$

where $\boldsymbol{F_3}$ is the force acting on the planet and $\boldsymbol{\nabla_3} = <\frac{\partial}{\partial X_3}, \ \frac{\partial}{\partial Y_3}>$ is the gradient operator acting on the coordinates of the planet. We thus obtain

$$\begin{aligned} \ddot{X}_3 &= -\frac{GM_1}{{r_1}^3}(X_3 - X_1) - \frac{GM_2}{{r_2}^3}(X_3 - X_2) \\ \ddot{Y}_3 &= -\frac{GM_1}{{r_1}^3}(Y_3 - Y_1) - \frac{GM_2}{{r_2}^3}(Y_3 - Y_2) \ . \end{aligned} \tag{2.12}$$

2.4.3 Introducing Dimensionless Coordinates

The distance D between the binary stars will be useful as a scale factor to put distances into a dimensionless form based on the substitutions $X_3 = Dx_3$ and $Y_3 = Dy_3$. Thus, the dimensionless position vector for the small mass in the sidereal reference frame is given as $<x_3, \ y_3>$. We also introduce a dimensionless time τ as $\tau = \omega t$. The dimensionless representation of the velocity, $<\dot{x}_3, \ \dot{y}_3>$, and acceleration, $<\ddot{x}_3, \ \ddot{y}_3>$, for the planet can be found from $dX_3/dt = D\omega\dot{x}_3$ and $d^2X_3/dt^2 = D\omega^2\ddot{x}_3$, respectively. Equivalent equations hold for the variables in the y direction and the overdot represents differentiation with respect to the dimensionless time τ. Upon substitution of these quantities into (2.12), we find

$$\begin{aligned} \ddot{x}_3 &= -\left(\frac{\alpha}{{r_1}^3}(x_3 - \mu\cos\tau) + \frac{\mu}{{r_2}^3}(x_3 + \alpha\cos\tau)\right) \\ \ddot{y}_3 &= -\left(\frac{\alpha}{{r_1}^3}(y_3 - \mu\sin\tau) + \frac{\mu}{{r_2}^3}(y_3 + \alpha\sin\tau)\right) \end{aligned} \tag{2.13}$$

with

$$\begin{aligned} r_1(t) &= \sqrt{(x_3(t) - \mu\cos\tau)^2 + (y_3(t) - \mu\sin\tau)^2} \\ r_2(t) &= \sqrt{(x_3(t) + \alpha\cos\tau)^2 + (y_3(t) + \alpha\sin\tau)^2} \ . \end{aligned} \tag{2.14}$$

This set of equations constitutes the dimensionless form of the equations of motion using fixed (sidereal) coordinates.

2.5 Transforming to Rotating (Synodic) coordinates

Next we transform the equations of motion (see Eq. 2.13) into a rotating (synodic) coordinate system that rotates along with the binary stars. Since the binary stars are orbiting each other counterclockwise, the necessary transformation is a rotation in the counterclockwise direction such that the binary stars remain at fixed positions which simplifies analysis of the restricted three body problem. In addition, fixed equilibrium points (Lagrange points) can be found at which the small mass rotates in sync with the binary stars. These will be discussed further in § 2.6.1

The vector transformation between the sidereal $\boldsymbol{V}$ and the synodic coordinate system $\boldsymbol{V^*}$ involves multiplication by the rotation matrix $\mathbf{R}$ and inverse matrix $\mathbf{R}^{-1}$, respectively, according to $\boldsymbol{V^*} = \mathbf{R}\boldsymbol{V}$ and $\boldsymbol{V} = \mathbf{R}^{-1}\boldsymbol{V^*}$, with the two matrices given as

$$\mathbf{R} = \begin{bmatrix} \cos\tau & \sin\tau \\ -\sin\tau & \cos\tau \end{bmatrix} \qquad \textbf{and} \qquad \mathbf{R}^{-1} = \begin{bmatrix} \cos\tau & -\sin\tau \\ \sin\tau & \cos\tau \end{bmatrix}.$$

Thus, the position of the binary stars and the planet in dimensionless synodic coordinates are given as

$$\begin{aligned}
\begin{bmatrix} x_1^* \\ y_1^* \end{bmatrix} &= \mathbf{R}\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \mathbf{R}\begin{bmatrix} \mu\cos\tau \\ \mu\sin\tau \end{bmatrix} = \begin{bmatrix} \mu \\ 0 \end{bmatrix} \\
\begin{bmatrix} x_2^* \\ y_2^* \end{bmatrix} &= \mathbf{R}\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = \mathbf{R}\begin{bmatrix} -\alpha\cos\tau \\ -\alpha\sin\tau \end{bmatrix} = \begin{bmatrix} -\alpha \\ 0 \end{bmatrix} \\
\begin{bmatrix} x_3^* \\ y_3^* \end{bmatrix} &= \mathbf{R}\begin{bmatrix} x_3 \\ y_3 \end{bmatrix} = \begin{bmatrix} x_3\cos\tau + y_3\sin\tau \\ -x_3\sin\tau + y_3\cos\tau \end{bmatrix}.
\end{aligned} \tag{2.15}$$

The latter equation can also be used to obtain relationships for the position, velocity and acceleration of the small mass in the two frames of reference

$$\begin{aligned}
\begin{bmatrix} x_3 \\ y_3 \end{bmatrix} &= \mathbf{R}^{-1}\begin{bmatrix} x_3^* \\ y_3^* \end{bmatrix} = \begin{bmatrix} x_3^*\cos\tau - y_3^*\sin\tau \\ x_3^*\sin\tau + y_3^*\cos\tau \end{bmatrix} \\
\begin{bmatrix} \dot{x}_3 \\ \dot{y}_3 \end{bmatrix} &= \mathbf{R}^{-1}\begin{bmatrix} \dot{x}_3^* - y_3^* \\ \dot{y}_3^* + x_3^* \end{bmatrix} \\
\begin{bmatrix} \ddot{x}_3 \\ \ddot{y}_3 \end{bmatrix} &= \mathbf{R}^{-1}\begin{bmatrix} \ddot{x}_3^* - 2\dot{y}_3^* - x_3^* \\ \ddot{y}_3^* + 2\dot{x}_3^* - y_3^* \end{bmatrix}
\end{aligned} \tag{2.16}$$

By multiplying both sides of Eq. (2.13) by $\mathbf{R}$ and using the above relations we obtain the equations of motion in the synodic coordinate system. If we label all the components of position, velocity, and acceleration in the synodic frame by an asterisk $(*)$, we find

$$\begin{aligned}
\ddot{x}_3^* - 2\dot{y}_3^* &= x_3^* - \frac{\alpha}{{r_1}^3}(x_3^* - \mu) - \frac{\mu}{{r_2}^3}(x_3^* + \alpha) \\
\ddot{y}_3^* + 2\dot{x}_3^* &= (1 - \frac{\alpha}{{r_1}^3} - \frac{\mu}{{r_2}^3})y_3^*
\end{aligned} \tag{2.17}$$

where the dimensionless synodic distances to each star can now be expressed as

$$\begin{aligned} r_1(t) &= \sqrt{(x_3^*(t)-\mu)^2+(y_3^*(t)-\mu)^2} \\ r_2(t) &= \sqrt{(x_3^*(t)+\alpha)^2+(y_3^*(t)+\alpha)^2}\ . \end{aligned} \tag{2.18}$$

2.6 Derivation of Jacobi's Integral

The components on the right side of Eqs. (2.17) are accelerations in the non-inertial reference frame of the rotating binary system. These can be rewritten as forces per unit mass or as the gradient of a scaler pseudo-potential $\overline{\phi}$, leading to

$$x_3^* - \frac{\alpha}{{r_1}^3}(x_3^*-\mu) - \frac{\mu}{{r_2}^3}(x_3^*+\alpha) = \frac{\partial\overline{\phi}}{\partial x_3^*} \tag{2.19}$$

$$y_3^*(1-\frac{\alpha}{{r_1}^3}-\frac{\mu}{{r_2}^3}) = \frac{\partial\overline{\phi}}{\partial y_3^*}\ , \tag{2.20}$$

which is satisfied by

$$\overline{\phi} \;=\; \frac{1}{2}\left[\left(x_3^{*2}+y_3^{*2}\right)+2\left(\frac{\alpha}{r_1}+\frac{\mu}{r_2}\right)\right]\ . \tag{2.21}$$

Now by multiplying the first equation in (2.17) by $\dot{x}_3^*$ and the second by $\dot{y}_3^*$ and adding them together utilising Eqs. (2.19 and 2.20) we obtain

$$\ddot{x}_3^*\dot{x}_3^* + \ddot{y}_3^*\dot{y}_3^* \;=\; \frac{\partial\overline{\phi}}{\partial x_3^*}\dot{x}_3^* + \frac{\partial\overline{\phi}}{\partial y_3^*}\dot{y}_3^*\ , \tag{2.22}$$

which if integrated with respect to τ leads to

$$\frac{1}{2}\left(\dot{x}_3^{*2}+\dot{y}_3^{*2}\right) \;=\; \overline{\phi}-\frac{\overline{C}}{2} \tag{2.23}$$

where $\overline{C}$ constitutes a constant of integration.

For the small mass the dimensionless distance from the center of mass and the synodic velocity are

$$\begin{aligned} r(\tau) &= \sqrt{x_3^{*2}(\tau) + y_3^{*2}(\tau)} \\ v^*(\tau) &= \sqrt{\dot{x}_3^{*2}(\tau) + \dot{y}_3^{*2}(\tau)}\ , \end{aligned} \tag{2.24}$$

which allow us to rewrite Eq. (2.23) as

$$v^{*2} \ = \ 2\overline{\phi} - \overline{C} \ = \ r^2 + 2\left(\frac{\alpha}{r_1} + \frac{\mu}{r_2}\right) - \overline{C}\ . \tag{2.25}$$

This is an integral of motion in the rotating cartesian coordinate system known as Jacobi's integral.

2.6.1 Lagrange Points

At this point in the development, it is beneficial to discuss a few useful solutions to Eqs. (2.19) and (2.20). If we consider the situation when the net force acting on the planet is zero, including centrifugal fictitious forces due to the rotation of the non-inertial synodic reference frame, then the planet will have zero acceleration in that frame and we will have an equilibrium position. Thus if the planet were to be placed exactly at such a point, then it would remain there. As such, Eqs. (2.19) and (2.20) would become

$$x_3^* - \frac{\alpha}{{r_1}^3}(x_3^* - \mu) - \frac{\mu}{{r_2}^3}(x_3^* + \alpha) \quad = \quad 0 \tag{2.26}$$

$$y_3^*\left(1 - \frac{\alpha}{{r_1}^3} - \frac{\mu}{{r_2}^3}\right) \quad = \quad 0. \tag{2.27}$$

There are five solutions to Eq. (2.26) and Eq. (2.27) which are shown in relation to the location of the binary stars in Fig. 2.2. These five equilibrium positions are

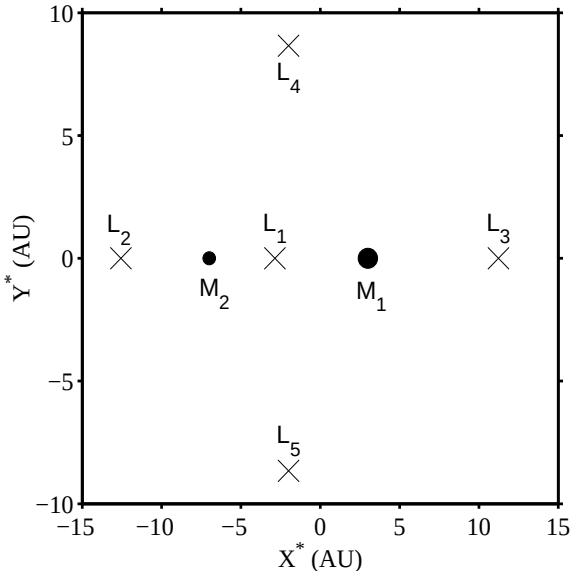


Figure 2.2. The five Lagrange points are shown for the particular case $\mu = 0.3$.

most commonly known as the Lagrange points. From Eq. (2.27) we have two basic situations, either $y_3^* \neq 0$ or $y_3^* = 0$. The first of which implies that the terms contained within the parentheses must equal zero which can easily be verified to be satisfied when $r_1 = r_2 = 1$. This means that the planet is the same distance from both stars as the stars are from each other, thus forming an equilateral triangle. We will refer to these points as L_4 and L_5 which have coordinate solutions $<x_3^*,\ y_3^*>$ $= <x_{(4,5)}^*,\ y_{(4,5)}^*>$. Clearly the $x_{(4,5)}^*$ value of these equilateral solutions will be the midway point between the two stars which is mass ratio dependent. From Eq. (2.15) we can see that the dimensionless position of the stars M_1 and M_2 are μ and $-(1-\mu)$ respectively. It is then easy to see that the midway point of these two positions is $x_{(4,5)}^* = \mu - \frac{1}{2}$. Plugging this back into the equation for either r_1 or r_2 we can easily find that $y_{(4,5)}^* = \pm\frac{\sqrt{3}}{2}$, the plus sign referring to L_4 and the minus sign referring to L_5.

The second situation, $y_3^* = 0$ implies that any remaining solutions lie on the line passing through the two stars. These collinear points will be designated from left to right by L_2, L_1, L_3 as indicated in Fig. 2.2. These three points cannot be solved for analytically and the numerical method outlined in §A.6 was used. The location of these points will be useful in distinguishing different regions of motion available to the planet and we will discuss this further in §2.6.6

2.6.2 Jacobi Constant at L_4 and L_5

The constant $\overline{C}$ can be set by the initial conditions and the system parameters μ and α, recalling that $\alpha = 1 - \mu$. Jacobi's integral (Eq. 2.25) is a relationship between the velocity and position of the planet that remains unchanged during the time-dependent development of the system. In the following, we show how using Jacobi's integral at the L_4 and L_5 equilibrium positions can be used to specify the Jacobi constant C, given as $C = \overline{C} + \mu(1-\mu)$ which has an absolute minimum of $C = 3$ at L_4 or L_5 for any mass ratio.

Since the velocity at L_4 or L_5 is zero and the distance from each star is 1, Eq. (2.25) gives

$$\overline{C}_{(4,5)} = 2\overline{\phi}_{(4,5)} = r^2 + 2\,(\alpha + \mu) = r^2 + 2\ . \tag{2.28}$$

Since $x^*_{(4,5)} = \mu - \frac{1}{2}$ and $y^*_{(4,5)} = \pm\frac{\sqrt{3}}{2}$, and with $\alpha = 1 - \mu$, we find the planet's distance from the center of mass at L_4 and L_5 is

$$r^2 = 1 - \mu\alpha. \tag{2.29}$$

This implies that the scalar pseudo-potential $\overline{\phi}_{(4,5)}$ and the constant $\overline{C}_{(4,5)}$ become

$$\begin{aligned} \overline{\phi}_{(4,5)} &= \frac{3}{2} - \frac{1}{2}\mu\alpha \\ \overline{C}_{(4,5)} &= 3 - \mu\alpha\ . \end{aligned} \tag{2.30}$$

Adding $\mu\alpha$ to $\overline{C}$ and adding $\frac{1}{2}\mu\alpha$ to $\overline{\phi}$ does not change the value of Jacobi's integral, and the new constant and pseudo-potential will be the same at L_4 or L_5 for any mass ratio μ. Therefore, we can define a new pseudo-potential and constant in reference to the equilateral equilibrium points as:

$$\begin{aligned} \phi &= \overline{\phi} + \frac{1}{2}\mu(1-\mu) \\ C &= \overline{C} + \mu(1-\mu) \end{aligned} \tag{2.31}$$

which are valid for any point and not only at L_4 or L_5, thus we will refer to C as the Jacobi constant. Furthermore, with the help of the equation $\alpha r_1^2 + \mu r_2^2 = r^2 + \mu\alpha$, the pseudo-potential ϕ at any point can be written in a more elegant form,

$$\phi = (1-\mu)\Big(\frac{{r_1}^2}{2} + \frac{1}{r_1}\Big) + \mu\Big(\frac{{r_2}^2}{2} + \frac{1}{r_2}\Big). \tag{2.32}$$

We can now rewrite Eq. (2.25) as

$$v^{*2} = 2\phi - C \tag{2.33}$$

and after specifying the initial conditions, the Jacobi constant can be given as

$$C = 2\phi_0 - v_0^{*2}. \tag{2.34}$$

It is worth noting that this is analogous to the energy available to the planet but it is most certainly **not** a statement saying that the energy of the planet is conserved. The total energy of the system is, but if we think back to how Eq. (2.34) was derived, we only considered the potential and kinetic energy of the planet. The potential energy between the stars and their kinetic energies are not a part of this equation.

2.6.3 Extremal Value

We already know that the equilateral equilibrium points yield an extreme value of the pseudo-potential, but is it a minimum or a maximum? If we take the first derivative of ϕ (Eq. 2.32) with respect to r_1 and/or r_2 we find that

$$\frac{\partial \phi}{\partial r_i} \;=\; k_i \left(r_i - \frac{1}{{r_i}^2} \right) \tag{2.35}$$

where $i = 1, 2$ and $k_i = 1 - \mu, \mu$. We can therefore easily confirm the extremal values of $r_i = 1$ from this. Now if we take the second derivative with respect to r_1 and/or r_2 we find that

$$\frac{\partial^2 \phi}{\partial {r_i}^2} \;=\; k_i \left(1 + \frac{2}{{r_i}^3} \right). \tag{2.36}$$

This is always positive for any r_i and ϕ approaches infinity as r_i approaches zero or infinity, so the equilateral equilibrium points are absolute minima and the pseudo-potential and Jacobi constant at these points are:

$$\phi_{min} \;= \frac{3}{2} \qquad\qquad C_{min} \;=\; 3 \tag{2.37}$$

2.6.4 The Jacobi Constant at L_1, L_2, and L_3

Just as the equilateral equilibrium positions served as excellent reference points, the collinear equilibrium positions are useful points of reference for the possible motion allowed to the planet. If a mass were to be stationary at any one of these three points, then we can find the Jacobi constant at such a point from Eq. (2.34) where clearly $v_0^* = 0$ and ϕ_0 is determined at the points L_1, L_2, or L_3. In Fig. 2.3 we show the Jacobi constant for a *stationary* particle lying on the x^* axis for three different mass ratios, $\mu = 0.5$ (red solid line), $\mu = 0.3$ (green dashed line), and $\mu = 0.1$ (blue dash-dotted line). Note that the Jacobi constant diverges as the position approaches either star. These can be considered as *inverted wells* which the planet can be caught in if the Jacobi constant is sufficiently high. This corresponds to the planet being in

orbit around one star or the other. For smaller values of the Jacobi constant, the *wells* become wider until eventually the barrier between the two separate wells no longer exists and the planet is free to move between the two stars. We also recognize that the local minima of the graph correspond to the collinear equilibrium positions. Therefore the middle minimum corresponds to the L_1 point and the value of the Jacobi constant there is C_1. Similarly, the minimum on the left corresponds to L_2 and the one on the right corresponds to L_3.

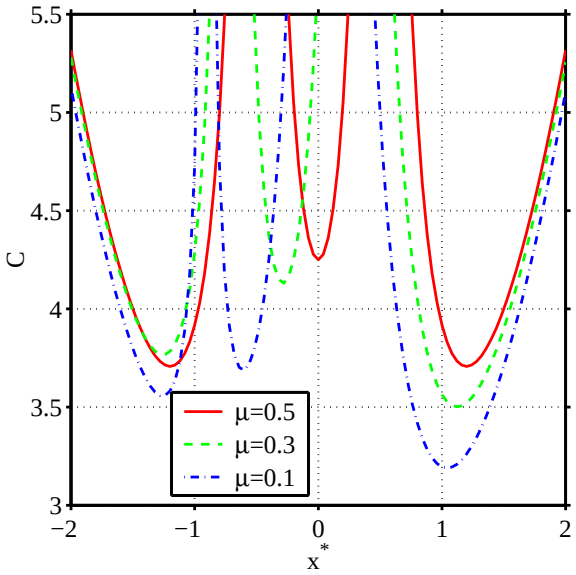


Figure 2.3. The Jacobi constant C as a function of stationary points on the x^* axis for three different mass ratios.

Clearly, the location of the minimum and the value of the Jacobi constant at these minima is dependent on the mass ratio. Starting with the largest mass ratio considered, we can see that the curves are symmetric when $\mu = 0.5$ which is reasonable since this is when the two stars have equal mass. We can see from the figure that as the mass ratio is reduced from 0.5, the L_3 point moves ever closer to the primary star as C_3 gets smaller and smaller. The L_1 point shifts in the same direction, but it is

approaching the secondary star while C_1 decreases. The behavior of the L_2 point is a little harder to determine from Fig. 2.3, but it is clear that C_2 has a maximum value at some point. In Fig. 2.4 we show the mass ratio dependence of the Jacobi constant at the three collinear equilibrium points as well as the differences between them. It is now clear that C_2 has a maximum near $\mu = 0.3$ and we numerically find that the maximum occurs at $\mu \approx 0.334$.

If we look at the differences between the Jacobi constants at the collinear points, we can see that the difference between C_1 and C_2 is a nearly linear monotonically increasing function of the mass ratio. This implies that the larger the mass ratio is, the larger the difference between planets that have orbits that can transfer between the stars and planets that have the possibility of being ejected from the system. Thus for the largest mass ratios, initial conditions that give a Jacobi constant that is less than C_1 and greater than C_2 have a good chance of being flung between the stars while still being confined to the inner part of the binary system.

The difference between C_2 and C_3 increases rapidly from $\mu = 0$ and reaches a maximum at $\mu \approx 0.136$, then decreases almost linearly as μ approaches 0.5. It is clear that the difference between C_1 and C_2 is relatively small for μ near 0.136 and this implies that, for μ near this value, there are few initial conditions yielding a Jacobi constant that is less than C_1 and greater than C_2. Therefore, initial conditions that allow the planet to transfer between the two stars are not very far from those that will allow the planet to escape via the opening of the allowable region near the L_2 point. The Jacobi constant must be significantly less in order for the planet to escape through the L_3 point, therefore the planet is more likely to escape by passing by the secondary star for mass ratios near 0.136.

The difference between C_1 and C_3 is indicative of the relative separation between the L_1 and L_3 points, and the maximum value occurs at $\mu \approx 0.286$, which is fairly

close to $\mu \approx 0.334$ which is where the maximum for C_2 is reached. This means that while there is a wide range of possibilities for initial conditions to give Jacobi constants between C_1 and C_3, the value C_2 is "easiest" to reach in this interval $(0.286 < \mu < 0.334)$. This would seem to imply that escapes through L_2 would be common, or at least would occur well before the Jacobi constant approaches C_3, but this is not the case at least for the initial conditions which will be specified in the next section.

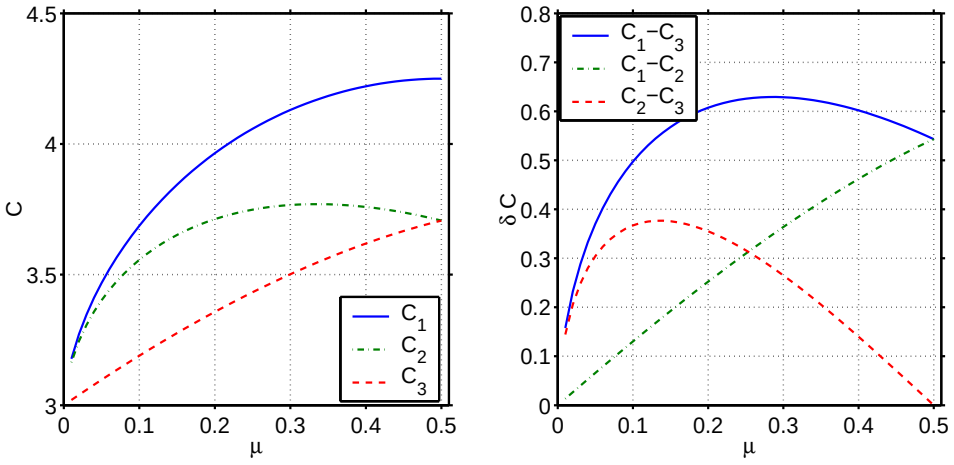


Figure 2.4. The dependance of the Jacobi constant at the collinear points on the mass ratio μ. (a) Jacobi constant C_1, C_2, and C_3 at the collinear points as a function of mass ratio. Note that C_2 has a maximum at $\mu \approx 0.334$. (b) Differences between collinear Jacobi constants. Note that $C_2 - C_3$ has a maximum at $\mu \approx 0.136$, whereas $C_1 - C_3$ has a maximum at $\mu \approx 0.286$.

2.6.5 Initial Conditions to Set the Jacobi Constant

In order to find the Jacobi constant for a particular orbit we are interested in, we will need to specify the initial conditions. The planet is started at the three-o-clock position a dimensionless distance of $\rho_0 = R_0/D$ (see Fig. 2.1(a)) relative to the

primary star and v_0^* (see Fig. 2.1(b)) is the perpendicular initial velocity of the small body in the rotating coordinate system which will be specified in the next paragraph. The system parameter μ and initial conditions, can now be used to give an expression for the Jacobi constant C, which is

$$C = 2\left[(1-\mu)\left(\frac{{\rho_0}^2}{2}+\frac{1}{\rho_0}\right)+\mu\left(\frac{(1+\rho_0)^2}{2}+\frac{1}{1+\rho_0}\right)\right]-{v_0^*}^2 \tag{2.38}$$

The initial velocity is set by adding the initial velocity of the primary star $R_1\omega$ and the velocity that would result in a circular orbit about a stationary star. Therefore, for the initial velocity in sidereal and synodic coordinates, we find

$$\begin{aligned} V_0 &= R_1\omega+\sqrt{\frac{GM_1}{R_0}} \\ V_0^* &= V_0-(R_1+R_0)\omega\ , \end{aligned} \tag{2.39}$$

respectively. The dimensionless form of these quantities can be obtained by dividing by $D\omega$, resulting in $v_0 = V_0/D\omega$ and $v_0^* = V_0^*/D\omega$. This also allows us to rewrite v_0 and v_0^* as

$$\begin{aligned} v_0 &= \mu+\sqrt{\frac{1-\mu}{\rho_0}} \\ v_0^* &= \sqrt{\frac{1-\mu}{\rho_0}}-\rho_0\ . \end{aligned} \tag{2.40}$$

The latter equation allows us to express the Jacobi constant C (see Eq. (2.38)) as

$$C = \mu+2\mu\rho_0+\frac{1-\mu}{\rho_0}+2\frac{\mu}{1+\rho_0}+2\sqrt{\rho_0(1-\mu)}\ . \tag{2.41}$$

Obviously, the Jacobi constant solely depends on the system parameters μ and ρ_0, and not on any orbital parameter of the planet, as attained during the simulation, a further indication that C will not change with time.

In Fig. 2.5 we show the Jacobi constant as a function of the initial distance ratio for the various mass ratios considered in this study. The domain chosen for the figure corresponds to the relevant parameter space that was explored because for $\rho_0 \ll 0.2$ we can see that the circular portion of the initial velocity (See Eq. 2.40) will dominate thus the planet is secure in its orbit around the primary star and Eq. (2.41) approaches infinity. There is no question of the stability in this case and in fact the greater C is, the tighter the orbit of the planet is. Of course in reality the planet cannot orbit with a radius of zero because it would be inside the star so Eq. (2.41) will no longer be valid.

For $\rho_0 > 1.0$ the Jacobi constant continues to increase without bound and the orbit is no longer confined to a satellite type orbit around the primary star (S-type) and it begins to take on a planetary type orbit about both stars (P-type). Again, the greater C is, the more securely the planet is in a circum-binary orbit. In the intermediate region, $0.2 < \rho_0 < 1.0$ we see that the curves have relative minima and in fact the ρ_0 that are within this interval result in orbits that are difficult to predict the long term behavior of.

2.6.6 Zero Velocity Curves and the Allowable Regions of Motion

Now that C has been set by the initial conditions as described in the previous section, the two terms in Eq. (2.33) will maintain a constant difference, the magnitude of which is just C. Both of these terms, v^{*2} and 2ϕ are always real numbers that are greater than or equal to 0. In the case where $v^{*2} = 0$, then we have $\phi = C/2$. Since ϕ is only a function of position, this allows us to delineate a contour for which the velocity of the planet would have to be equal to zero. If the planet were to somehow move into a region where $\phi < C/2$, then the velocity would be imaginary and this cannot occur. Thus these curves of zero velocity set an allowable region of motion

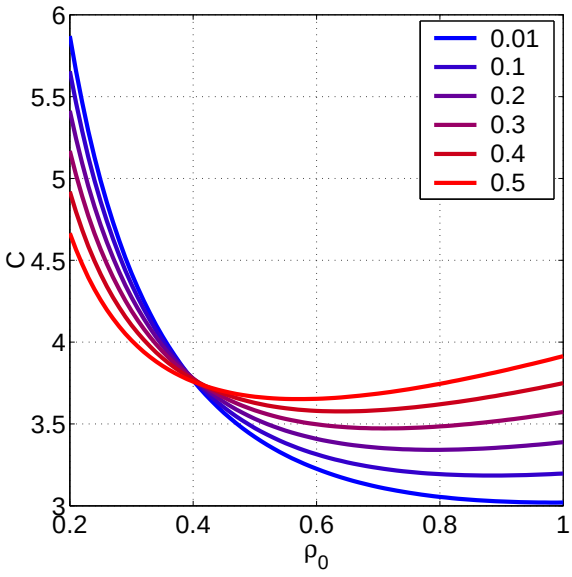


Figure 2.5. The Jacobi constant C as determined by Eq. 2.41 is shown as a function of the initial distance ratio for various mass ratios.

for the planet. The size and shape of the allowable region depends on the initial conditions. From Fig. 2.5, we can see that for relatively small ρ_0 the Jacobi constant is high and therefore from Fig. 2.3, the planet is tightly bound to the primary star. As ρ_0 is increased the Jacobi constant decreases until at one point it is equal to C_1. From this point on, greater values of ρ_0 will give the planet a Jacobi constant that is less than C_1 and then the planet has enough relative energy that it can transfer from the primary star to the secondary star. Just because it *can* does not necessarily mean it *does*. Thus the initial conditions that result in a Jacobi constant that is equal to C_1 are sufficient for an upper limit for "guaranteed stability", but there is not an implication of instability.

As ρ_0 is further increased, the Jacobi constant continues to fall until it is less than C_2. At this point the planet has enough relative energy to escape from the vicinity of the binary stars. It would seem that this would be an adequate way of gauging where instabilities occur. Once again however, just because it *can* escape

does not imply that it actually does. Continuing to increase ρ_0 will further decrease the Jacobi constant until it falls below C_3 at which point the planet can escape from the binaries through an opening that appears in the zero velocity curves near the vicinity of the primary star. This is a relatively easy escape route and for all but the most robust systems, the planet has become unstable well before reaching this limit.

We didn't investigate the behavior of the system much beyond this point because after a moderate increase of ρ_0 the Jacobi constant actually starts to increase (See Fig. 2.5. This implies that the opening near L_3 closes up, and then the opening at L_2 and at this point the planet is excluded from the interior of the binary system. The further out the planet starts its orbit, the more stable this circum-binary orbit is. The value of the Jacobi constant is obviously dependent on the position and the magnitude of the velocity but it does not depend on the direction so one may think that the direction will not affect the behavior of the planet's orbit. This is not the case however as can be seen in Fig 2.6

2.7 Introduction of the Effective Eccentricity

Sir William Rowan Hamilton is credited with the notion of the hodograph as a way of describing a planetary orbit based on the variation of the velocity vector (see Hamilton 1847). Simply put, the hodograph is the path that the tip of the velocity vector traces out as it varies about the fixed origin. In particular, Hamilton showed that under the influence of the central inverse square gravitational force, the hodograph is always a circle. For a more recent and accessible discussion see Butikov (2000).

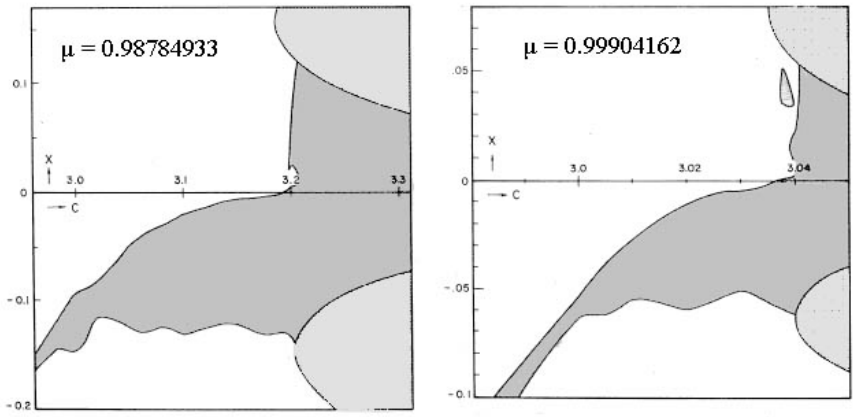


Figure 2.6. Henon stability diagrams from Jefferys (1974) for a lunar orbiter on the left and Jupiter orbiter on the right. In the heavily shaded region the orbits are stable; in the unshaded region, unstable; and the lightly shaded region is the forbidden region bounded by the zero velocity curves. Positive and negative x correspond to prograde and retrograde orbits respectively. Lower values of C correspond to larger orbits, thus it is clear from these figures that retrograde orbits are stable for larger orbits than prograde orbits.

We consider polar coordinates by starting from the familiar form of the Kepler orbit

$$r \;=\; \frac{p}{1+e\cos\phi} \tag{2.42}$$

where $\phi = \theta - \theta_0$. Here r denotes the radius, e the eccentricity, p the semi-latus rectum, and ϕ and θ denote angles with θ_0 indicating the initial value and measured from the $+x$ axis.

In cartesian vector form, we find $\boldsymbol{r} = < r\cos\theta, r\sin\theta >$. Now, we differentiate with respect to time to obtain the velocity vector $\boldsymbol{v} =< v_x, v_y >$, which can be found with simple algebra as

$$\boldsymbol{v} \;=\; \frac{r^2}{p}\frac{d\theta}{dt} < -\sin\theta - e\sin\theta_0, \cos\theta + e\cos\theta_0 > \tag{2.43}$$

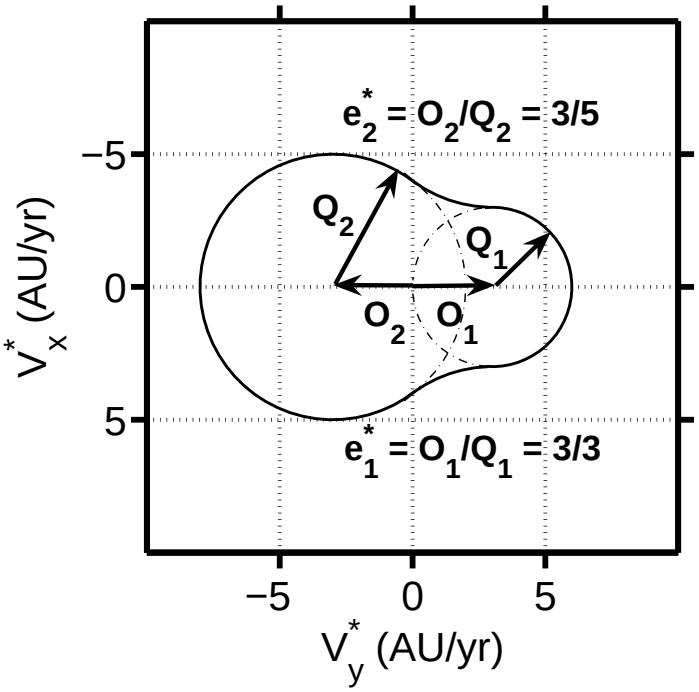


Figure 2.7. Two examples of a circle of curvature for a hodograph depicting the motion of a planet in the synodic coordinate system (V_y^*, V_x^*).

and can be expressed in two parts $\boldsymbol{v} = \boldsymbol{Q} + \boldsymbol{O}$, noting that

$$\boldsymbol{Q} = \frac{h}{p} < -\sin\theta, \cos\theta > \tag{2.44}$$

$$\boldsymbol{O} = \frac{he}{p} < -\sin\theta_0, \cos\theta_0 >, \tag{2.45}$$

where $\boldsymbol{Q}$ is the rotating component and $\boldsymbol{O}$ is the constant center. In the latter transformation, we made use of the identity $r^2 d\theta/dt = L/m = h$ indicating the conservation of angular momentum, denoted as L, where h is sometimes referred to as specific angular momentum. It is also assumed that the mass of the planet m does not change during the planetary orbital motion.

The velocity vector can be considered as a vector of constant length $Q = |\boldsymbol{Q}| = h/p$ rotating about a center located at $\boldsymbol{O}$, where $O = |\boldsymbol{O}| = he/p$. To find a more appropriate representation, we consider the velocity axes rotated 90° clockwise from

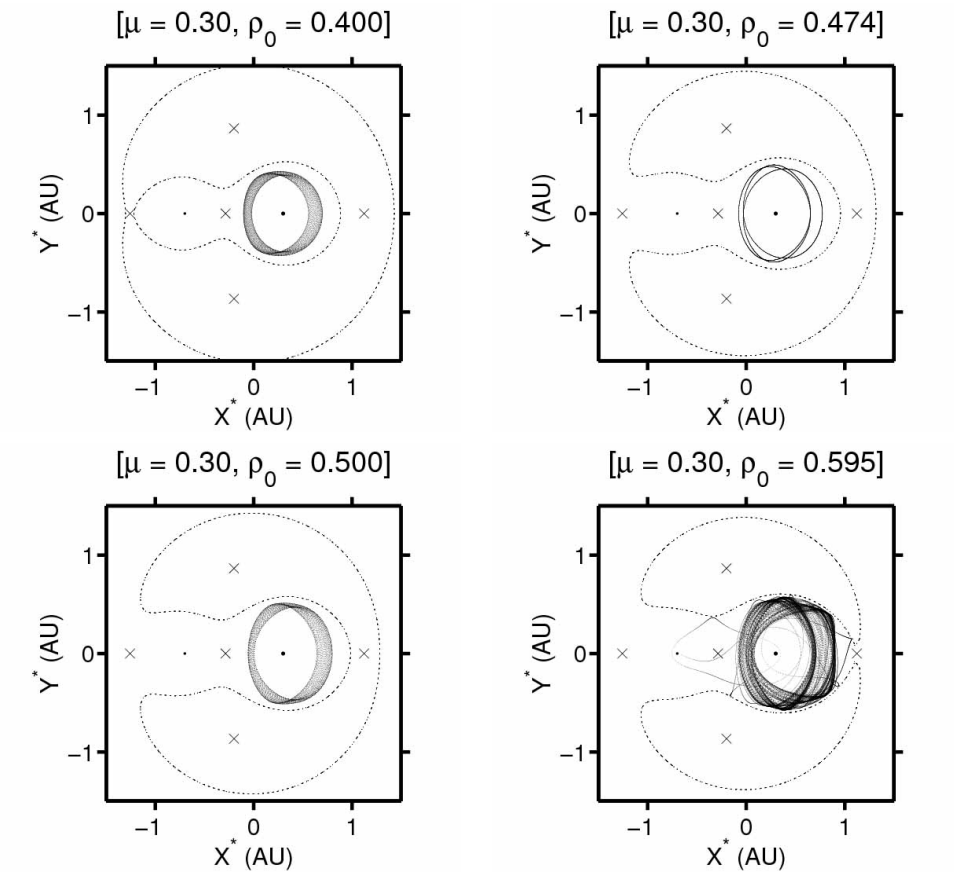


Figure 2.8. Planetary orbits in the synodic coordinate system with $\mu = 0.3$ and $\rho_0 = 0.400$, 0.474, 0.500, and 0.595.

the position axes. In this case, the hodograph will be traced out in the same manner as the orbit, e.g., if the orbit is counterclockwise, then the hodograph will be traced out counterclockwise as well. Moreover, the length and position of the velocity vector allow us to identify the effective eccentricity (i.e., locally defined along-side the hodograph), as we find $e = O/Q$.

For a Keplerian orbit, the situation is as follows: If $e < 1$ then we have a closed elliptical orbit and the origin of the coordinate system lies within the circle; in the special case of $e = 0$, we have a circular orbit and the center in velocity space coincides with the origin. If $e = 1$, we have a parabolic orbit and the circular hodograph passes

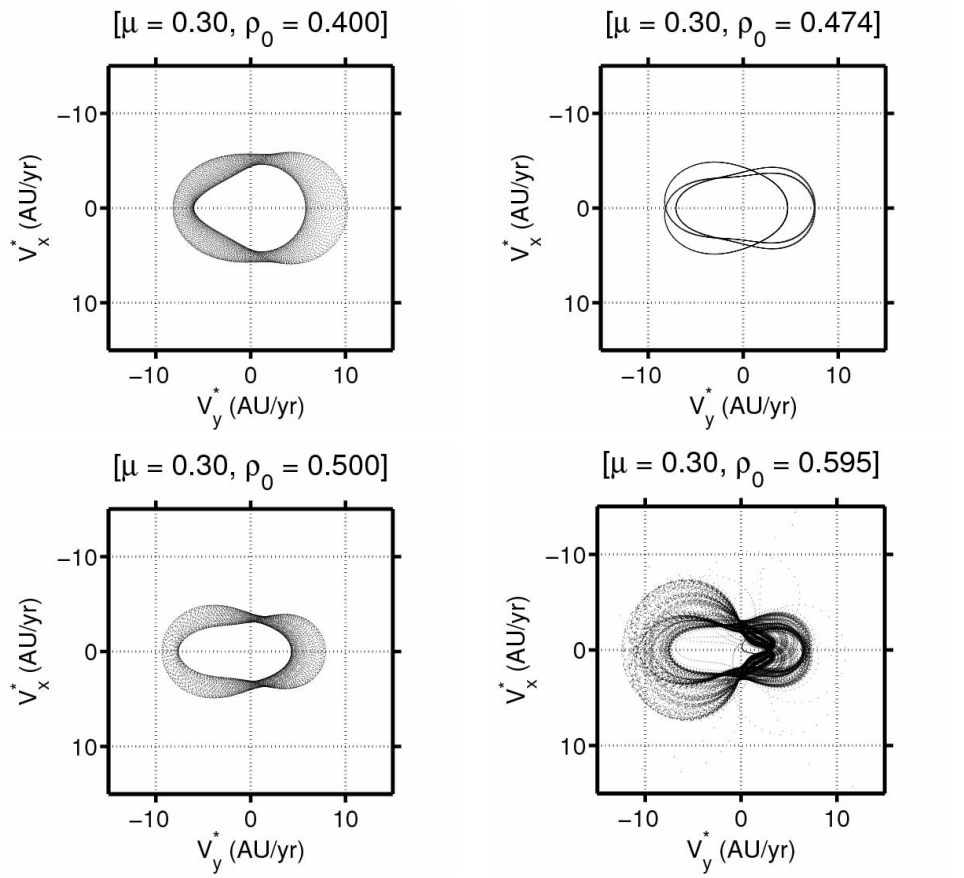


Figure 2.9. Same as Fig. 2.8, but now the behavior of the hodograph is shown.

through the origin in velocity space. If $e > 1$, the orbit is hyperbolic and the origin in velocity space lies outside the perimeter of the hodograph.

For a non-Kepler orbit, as obtained in conjunction with the restricted three body problem, it is again possible to determine at any given moment the eccentricity from the radius of curvature and the center of curvature of the hodograph. The radius of curvature of a parameterized function such as $x = f(t)$ and $y = g(t)$ can be obtained as

$$Q \;=\; \frac{1}{A}\left(\dot{f}^2 + \dot{g}^2\right)^{3/2} \tag{2.46}$$

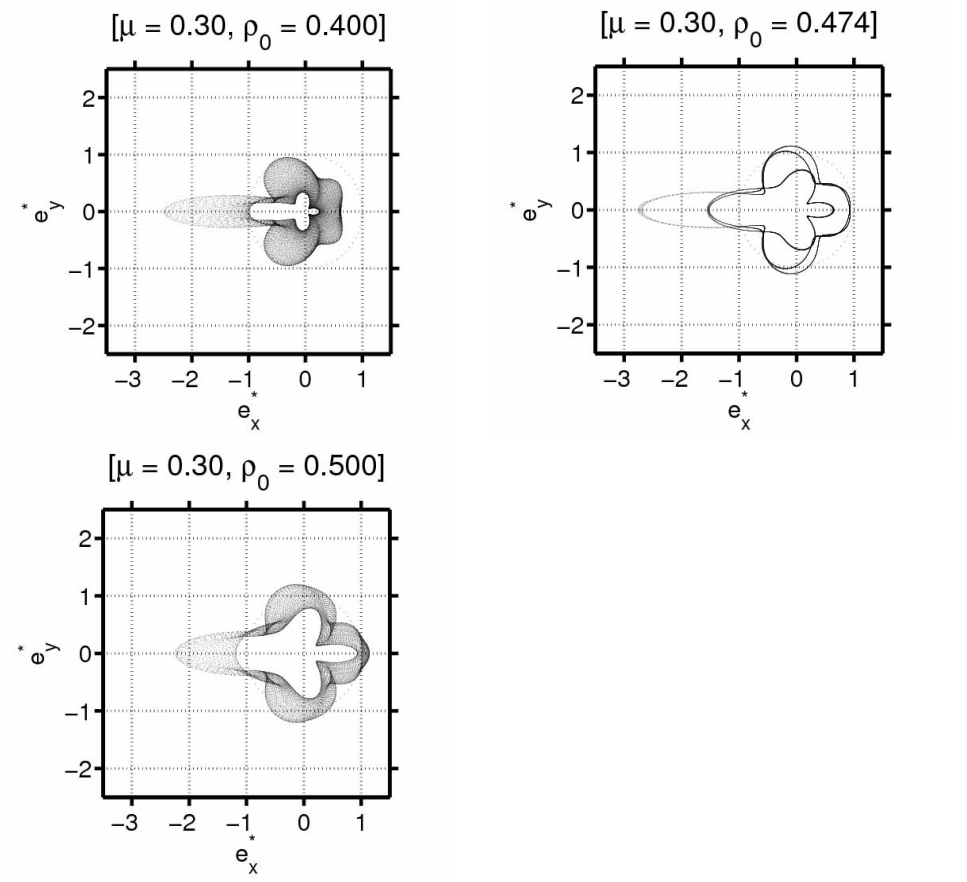


Figure 2.10. Same as Fig. 2.8 except for the case of $\rho_0 = 0.595$, but now the behavior of the effective eccentricity is shown. The magnitude of the effective eccentricity is projected onto the unit velocity vector.

whereas the location of the center of curvature (ξ, η) is given as

$$\xi = f - \frac{1}{A}\left(\dot{f}^2 + \dot{g}^2\right)\dot{g} \tag{2.47}$$

$$\eta = g + \frac{1}{A}\left(\dot{f}^2 + \dot{g}^2\right)\dot{f} \tag{2.48}$$

with $A = |A'|$ and

$$A' = \begin{vmatrix} \dot{f} & \dot{g} \\ \ddot{f} & \ddot{g} \end{vmatrix} \tag{2.49}$$

(see, e.g., Bronshtein and Semendyayev 1997). The distance of the center of curvature from the origin can thus be obtained as $O = \sqrt{\xi^2 + \eta^2}$. For the restricted three body

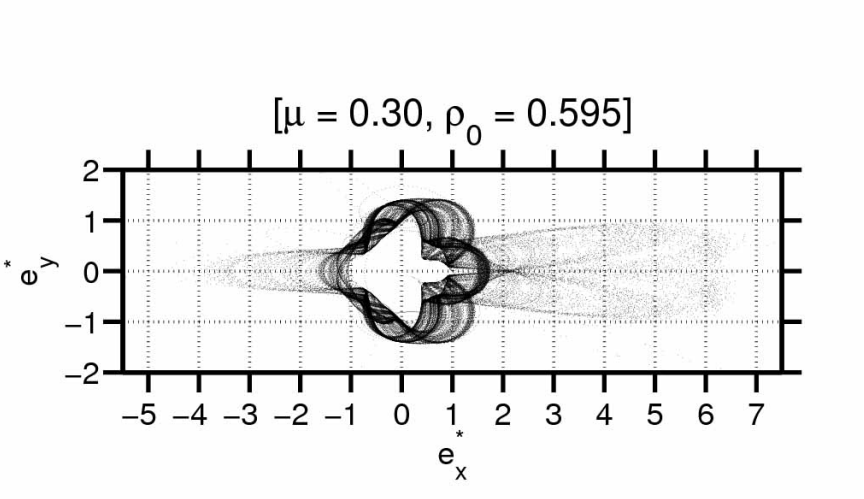


Figure 2.11. Effective eccentricity projected onto the unit velocity vector for the case $\rho_0 = 0.595$.

problem, we take $f(t) = V_y{}^*$ and $g(t) = -V_x{}^*$ with the asterisk denoting the usage of the synodic (rotating) coordinate system. Therefore, the effective eccentricity at any instant is given as $e^* = O(t)/Q(t)$ (see Fig. 2.7).

2.8 Results and Discussion

In the following, we consider the CR3BP (Szebehely 1967; Roy 2005, and references therein). For our study, we assume that the planetary mass is 1×10^{-6} of the mass of the star it orbits. Additionally, we assume that the initial velocity of the planet is in the same direction as the orbital velocity of its host star, which is the more massive of the two stars, and that the starting position of the planet is to the right of the primary star along the line joining the binary components (3 o'clock position). Obviously, the planetary mass deviates from a zero test mass. Nevertheless, it is not expected to have a significant effect on the motion of the two stars, implying that the conclusions derived from the CR3BP concept remain valid.

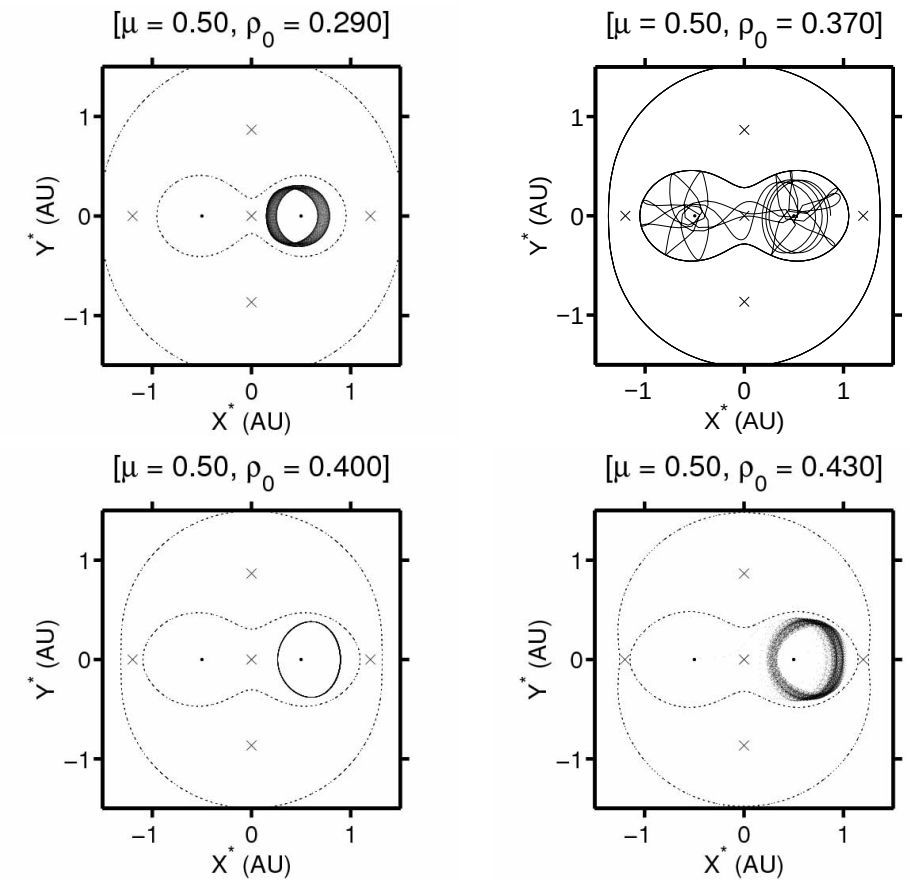


Figure 2.12. Planetary orbits in the synodic coordinate system with $\mu = 0.5$ and ρ_0 = 0.290, 0.370, 0.400, and 0.430.

We performed simulations for stellar mass ratios from $\mu = 0.0$ to $\mu = 0.5$ in increments of 0.01. As a measure of the precision of the integration scheme, we note that a time step of $\epsilon = 10^{-4}$ yrs is used as it has been carefully tested by inspecting the orbital paths under a variety of initial conditions based on time steps of $\epsilon = 10^{-5}$ and 10^{-6} yrs with no discernable difference. We also set an upper time limit of 10^3 yrs for our simulations. Although this value is very small, it is sufficient for models of orbital stability if the stability can be proven analytically (Cuntz et al. 2007; Eberle et al. 2008), or for determining and evaluating orbital instability if the planet is ejected from the system or captured by the secondary star prior to the pre-set time

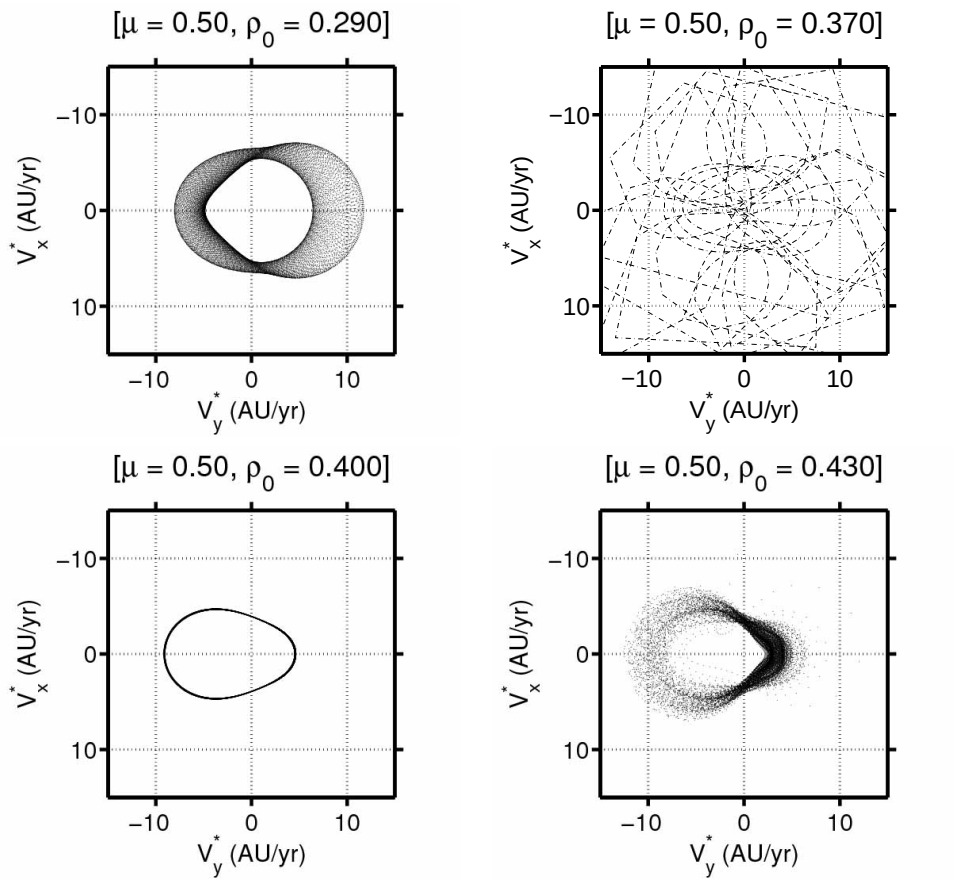


Figure 2.13. Same as Fig. 2.12, but now the behavior of the hodograph is shown.

of simulation. Surely, there is no guarantee that the orbit of a potentially unstable planet that appears stable over the time of integration will remain stable beyond the allotted time as this would require longer simulations.

We display runs for selected initial conditions (i.e., starting distances ρ_0 of the planet from the stellar primary) corresponding to $\mu = 0.3$ in Figs. 2.8 to 2.11 and to $\mu = 0.5$ in Figs. 2.12 to 2.14. In Fig. 2.8 and 2.12, we present the planetary orbits in the synodic coordinate system (X^*, Y^*). In Figs. 2.9 and 2.13, the same simulations are depicted by using the hodograph $(V_y{}^*, -V_x{}^*)$ and in Figs. 2.7, 2.11 and 2.14 by using the effective eccentricity $(e_x{}^*, e_y{}^*)$ projected onto the unit velocity vector. The selected starting distance ratios ρ_0 of the planet are 0.40, 0.474, 0.50, and 0.595 for

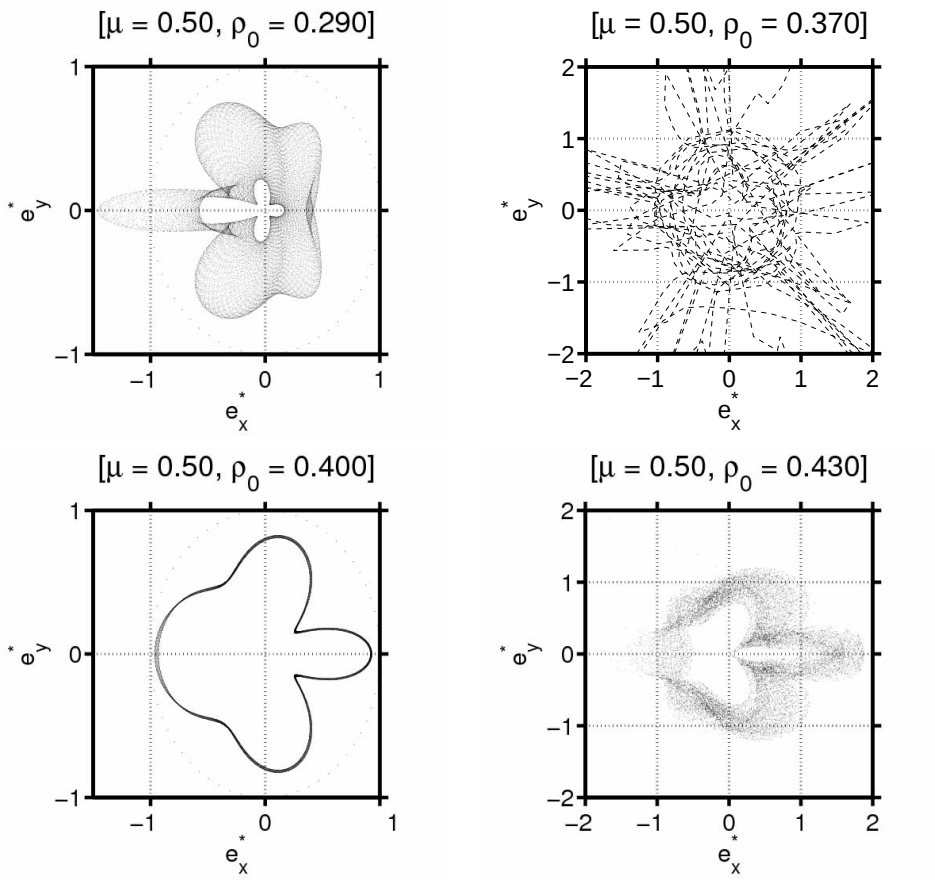


Figure 2.14. Same as Fig. 2.12, but now the the effective eccentricity is shown. The magnitude of the effective eccentricity is projected onto the unit velocity vector.

$\mu = 0.3$ and 0.29, 0.37, 0.40, 0.43 for $\mu = 0.5$. Note that ρ_0 indicates the relative initial distance of the planet, given as R_0/D, with $D = 1$ AU as the distance between the two stars and R_0 as the initial distance of the planet from its host star, the primary star of the binary system.

As this system has been studied before, both analytically and numerically, we have solid information on the conditions for planetary orbital stability. Absolute orbital stability occurs if $\rho_0 < \rho_0^{(1)}$ where $\rho_0^{(1)}$ represents the point at which the initial conditions result in a zero-velocity contour that intersects L_1. For larger values of ρ_0, stability is not guaranteed, because the zero-velocity contour opens at the Lagrange

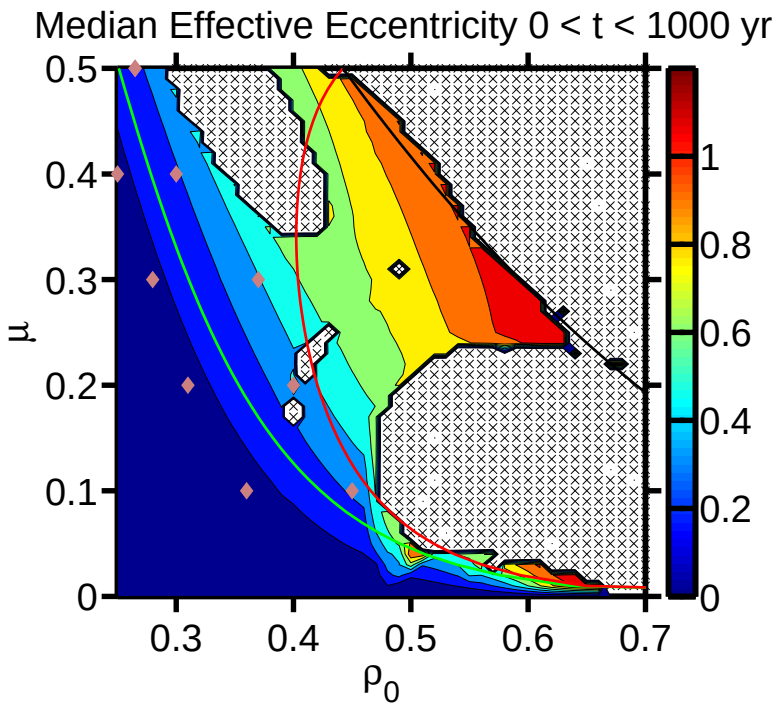


Figure 2.15. The median effective eccentricity (color coded) for various mass ratios μ and initial conditions ρ_0. The crosses are cases where the simulation was terminated due to the planet captured by one of the stars or being ejected from the system. The twelve white spots are cases where the planet was ejected from the system less "severely" and the simulation was not halted. The green, red, and black curves represent the initial conditions which cause the ZVC to intersect L_1, L_2, and L_3, respectively. The pink diamonds are limits of stability given by Pilat-Lohinger and Dvorak (2002).

points L_1, L_2, and L_3 for $\rho_0^{(1)}$, $\rho_0^{(2)}$, and $\rho_0^{(3)}$, which for $\mu = 0.3$ are given as 0.311, 0.404, and 0.593, and for $\mu = 0.5$ are given as 0.251, 0.442, and 0.442, respectively (e.g., Cuntz et al. 2007; Eberle et al. 2008). Note that for $\mu = 0.5$ the zero-velocity contour opens at L_2 and L_3 simultaneously for obvious reasons. Any such opening, in principle, allows the planet to be captured by the secondary star or to escape from the system entirely.

However, in certain cases this may not happen after all. Eberle et al. (2008) identified a domain of quasi-periodic orbital stability, embedded in the domain of

Table 2.1. Effective eccentricity study for the models of $\mu = 0.3$

ρ_0	e_{mean}	e_{median}	σ_{e}	ZVC
0.20	0.026	0.027	0.01	...
0.30	0.17	0.17	0.07	...
0.40	0.62	0.58	0.34	L_1
0.474	0.82	0.75	0.35	L_1, L_2
0.50	0.88	0.85	0.29	L_1, L_2
0.595	1.34	1.14	0.93	L_1, L_2, L_3
0.60	1.54	1.10	3.79	L_1, L_2, L_3

Note: The last column indicates the Lagrange point(s) where the zero-velocity contour (ZVC) is open.

instability in the (ρ_0, μ) diagram. A key example studied here is $\mu = 0.3$ and $\rho_0 = 0.474$, where the orbit and hodograph show periodic behavior (see Figs. 2.8 and 2.9). The orbital data indicate that the planet crosses the X^* axis every quarter of the binary period. The planet's orbit starts at the right outermost position (i.e., 0.474 AU from the primary star), then continues on to the innermost crossing on the left. After another quarter period the planet is crossing on the innermost path on the right. The rest of the orbit follows as outermost left, innermost right again, then innermost left again and then finally after a total of one and a half times the binary period the planet returns to where it started and the orbital pattern repeats.

The corresponding hodograph (Fig. 2.9) shows an equivalent behavior except that at distances further away from the primary star slower velocities are generally attained and vice versa. There is no indication of planetary escape at all, although no firm statement can be made about the long-term nature of the orbit. Note that the effective eccentricity if projected onto the unit velocity vector also traces out a periodic pattern (see Fig. 2.7). Moreover, the point at which the effective eccentricity is at its largest, the planet is passing closest to the L_1 point. The eccentricity distribution

Table 2.2. Effective eccentricity study for the models of $\mu = 0.5$

ρ_0	e_{mean}	e_{median}	σ_{e}	ZVC
0.25	0.19	0.20	0.08	...
0.29	0.47	0.45	0.23	L_1
0.30	0.90	0.67	1.10	L_1
0.35	1.26	0.82	1.58	L_1
0.37	1.27	0.98	1.96	L_1
0.40	0.68	0.71	0.17	L_1
0.43	0.91	0.87	0.31	L_1
0.50	0.40	0.089	1.23	L_1, L_2, L_3
0.55	2.30	1.01	10	L_1, L_2, L_3

Note: The last column indicates the Lagrange point(s) where the zero-velocity contour (ZVC) is open.

indicates a mean value of 0.82 and a median value of 0.75 (with a standard deviation of $\sigma_e = 0.35$; see Table 2.1), consistent with the previous interpretation. Note that the case of $\rho_0 = 0.4$ also shows quasi-stability, but there is no quasi-periodicity obtained for the planetary orbit.

A different situation occurs for $\rho_0 = 0.5$ and 0.595. In these cases, no multi-periodicity if found, and the hodograph shows a highly complex topology. Additionally, larger values for the median values of the eccentricity distribution are attained, which are 0.85 and 1.14, respectively. However, even though the model $\rho_0 = 0.5$ should correspond to an unstable orbit, as previously pointed out, no evidence of planetary escape surfaces. This situation is very different for $\rho_0 = 0.595$, a case where the planet escapes from the system after nearly 211 yrs. This outcome is also reflected by the extreme behavior of the eccentricity vector $(e_x{}^*, e_y{}^*)$ (see Fig. 5), exhibiting values beyond 6.3 and 1.4 for $e_x{}^*$ and $e_y{}^*$, respectively.

A further analysis shows that the transition of the median in the eccentricity distribution beyond unity occurs at about $\rho_0 = 0.58$, which is surprisingly close to

$\rho_0 = \rho_0^{(3)}$, where the zero-velocity contour is open at all Lagrange points L_1, L_2, and L_3. For the sake of curiosity, we also pursued simulations with relatively close planetary starting positions ρ_0, which were 0.2 and 0.3 (see Table 2.1). In both cases, orbital stability is guaranteed because of the stringent stability criterion, attained for $\rho_0 < \rho_0^{(1)}$ with $\rho_0^{(1)} = 0.311$. This result is also supported by analyses invoking histograms of the eccentricity distributions which again reveal the mean and median values of the effective eccentricity during the onset of planetary orbital instability.

The development of instability as ρ_0 increases is different for $\mu = 0.5$. The zero-velocity contour opens up at L_1 for $\rho_0 = 0.251$; therefore, unstable orbits occur for smaller ρ_0 values as expected. For $\rho_0 = 0.29$ the planetary orbit is just on the verge of being unstable as it is slowly spiraling away from the primary star and toward the secondary star (see Fig. 2.12). For slightly larger ρ_0 values, the planet spirals away quickly and transfers to the secondary star and possibly back again before it crashes into one of the stars. For ρ_0 between 0.38 and 0.42 there appears to be a quasi-stable region; note that $\rho_0 = 0.4$ is particularly interesting because it seems to correspond to a periodic case. The upper limit of this quasi-stable region corresponds to ρ_0 where the zero-velocity contour close to the L_2 and L_3 points opens up and the planetary orbits once again exhibit instability.

If the planet orbits close to the primary star, the effective eccentricity remains low for the entire duration of the simulation. When the parameters are such that the zero-velocity contour is nearly intersecting the L_1 equilibrium point (C_1 contour, green in Fig. 2.15), the effective eccentricity increases. For mass ratios less than 0.1 the zero-velocity contour corresponding to L_2 (C_2, red in Fig. 2.15) is a suitable limit for instability. For initial conditions where the planet is far enough away that the zero-velocity curve as already opened up at L_3 (C_3 contour, black in Fig. 2.15),

the planet is insufficiently bound to be able to orbit the primary star. Therefore, it quickly escapes from the region of the binary stars and for most initial conditions, the planet quickly returns and rapidly approaches one of the stars.

In Fig. 2.15 we show the median effective eccentricity over the 1000 year time period for all μ and ρ_0 that we considered in our simulations. The white regions with black crosses represent simulations that were terminated before the maximum 1000 year limit due to captures by one of the stars or ejection from the system. One unstable region is in the upper mass ratio range, roughly from $\mu = 0.35$ to 0.50 and another is from 0.00 to 0.23. Between these two regions ($\mu = 0.24$ to 0.34) there is a quasi-stable region in which the effective eccentricity increases almost uniformly. The boundaries of this region remain basically the same for the duration of the simulation as obtained by the inspection of the median effective eccentricity over the course of the binary period. Of course, we should not be surprised if the quasi-stable region were to shrink over a longer time period as 1000 orbits is not a very long time when considering long term stability. In fact, there is already an instability 'island chain' slowly forming right in the middle of this quasi-stable region. The reasons for this instability 'archipelago' are still under investigation.

As a comparison to other work, we show the minimum and maximum extension of the stable zone for the circular case taken from Pilat-Lohinger and Dvorak (2002) as pink diamonds. These limits were determined by setting various initial conditions for the third body such that it would have various initial eccentricities and then using the Fast Lyapunov Indicator method (Froeschlé et al. 1997) to determine the extents for which the orbit was unstable. This work conveys the minimum and maximum stable extensions of the stable zones, which apparently group around the C_1 curve associated with the stringent limit of stability given by the CR3BP.

2.9 Conclusion

We present a new method that allows to identify the onset of orbital instability for planets in binary systems. The method can be used "along the way" of detailed numerical simulations, and does not require a piece-wise secondary integration as, e.g., methods based on the Lyapunov exponent. Thus, it is an alternative to other methods such as, e.g., the Fast Lyapunov Indicator method (Fouchard et al. 2002) as it allows important insight into the properties of dynamic systems such as quasi-periodicity and multi-periodicity of orbits (as gauged by the hodograph), besides the identification of the onset of orbital instability. If a planet becomes orbitally unstable, high effective eccentricities are attained, which facilitates the new criterion. When the planet has left the system, it typically moves in a slowly expanding spiral if viewed in the synodic coordinate system corresponding to an effective eccentricity of close to zero. In this case, our criterion has become defunct, and if needed should be replaced by an assessment of the semimajor axis of the planet, which is expected to reach relatively high values.

Our method relies on a differential geometrical approach that analyzes the curvature of the planetary orbit in the synodic coordinate system. Based on our simulations, it is found that when the median of the eccentricity distribution exceeds unity, orbital instability occurs. This criterion can be included in detailed simulations in an automated mode allowing the identification of planetary ejection in a straightforward manner. Intuitively, this criterion also agrees with the most basic property of conic sections representing closed orbits for $e < 1$ and open orbits for $e \geq 1$, although it utilizes a modified definition of eccentricity.

A comparison with the zero-velocity contour of the planetary orbit shows that this criterion, which is a sufficient criterion for orbital instability, closely coincides with the opening of the zero-velocity contour at the Lagrange point L_3, located to

the right of the primary star, as previously discussed by Cuntz et al. (2007) and Eberle et al. (2008). If instability occurs, extremely large effective eccentricities are found, indicating a highly hyperbolic planetary orbit in the synodic coordinate system, a stark indication of planetary escape. Although our results have been described for the special case of the CR3BP, we expect that it may also be possible to augment this criterion to planets in generalized stellar binary systems, and to used it for the assessment of long-term simulations. Clearly, the CR3BP is an idealized case which is barely (if at all) realized in nature. Desired generalizations should include studies of the ER3BP (e.g., Pilat-Lohinger and Dvorak 2002; Szenkovits and Makó 2008) as well as of planets on inclined orbits (Dvorak et al. 2007), noting that especially the former have important applications to real existing systems in consideration of the identified stellar and planetary orbital parameters.

CHAPTER 3

PROJECT TWO: ON THE REALITY OF THE PROPOSED PLANET IN THE ν OCTANTIS SYSTEM

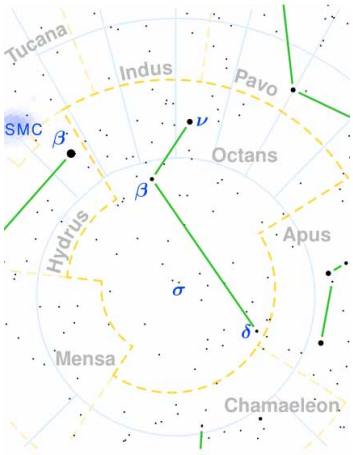


Figure 3.1. The Octans constellation.

The aim of this study is to explore an enigmatic finding about the ν Octantis binary system that indicates the possible existence of a Jupiter-type planet even though the planet seems to be located outside the zone of orbital stability. We perform a detailed analysis of orbital stability based on previous studies that carefully considers the ν Octantis system parameters including their observationally deduced uncertainties. In our analysis we confront the probability distribution of the parameter space of the system with the requirements of planetary orbital stability. Our results indicate that the suggested planet if in a prograde orbit with respect to the motion of the binary components is virtually impossible. However, the estimated probability of existence for a planet in a retrograde orbit is nearly 60%, an estimate

that encapsulates the probability distribution of the mass ratio of the stellar components. This estimate increases if a relatively low stellar mass ratio (within the error bars) is assumed. The principal possibility of a planet in a retrograde orbit is also consistent with long-term orbital stability simulations pursued as part of our study. Thus the existence of the suggested planet in the ν Octantis system constitutes a realistic possibility.

3.1 Introduction

A topic of fundamental importance to astrophysics and astrobiology is the study of orbital stability of theoretical and observed planets in star-planet systems; see, e.g., Roy (2005) for textbook information on planetary orbital stability. A significant controversy has emerged concerning the ν Octantis binary system where observational evidence indicates the possible existence of a Jupiter-type planet even though the planet (if existing) seems to be located outside the zone of orbital stability; see Ramm et al. (2009) for details. Previous observations have shown that planets are, in principle, able to exist in relatively close binary systems with separation distances of 20 to 25 AU, as pointed out by, e.g., Patience et al. (2002) and Eggenberger et al. (2004). Additionally, theoretical simulations have been pursued on planet formation in binary systems with different parameters that have led to highly favorable outcomes (e.g., Kley 2001; Quintana et al. 2002). The idea that planets in binary systems are now viewed to be possible is furthermore consistent with the detection of debris disks in various main-sequence stellar binary systems obtained by the *Spitzer Space Telescope* (e.g., Trilling et al. 2007).

The ν Octantis binary system consists of a primary of spectral type K1 III (Houk and Cowley 1975) and a secondary of spectral type K7–M1 V, see discussion by Ramm et al. (2009), with a separation distance of 2.55 ± 0.13 AU (see Table 3.4; note that all

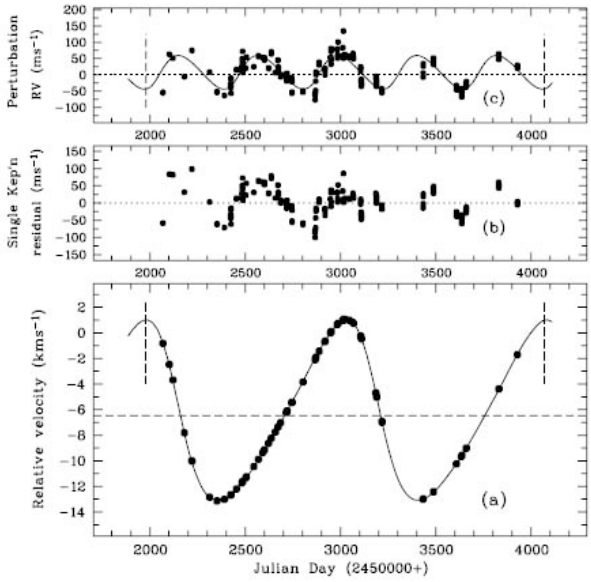


Figure 3.2. Figure 4 from Ramm et al. (2009). (a) relative RVs of ν Oct with the curve corresponding to the orbital solution from Ramm et al. (2009). (b) Residual velocities from a single-Keplerian fit. (c) Perturbation velocities derived using a double-Keplerian fit, including the corresponding best-fitting curve. The vertical dashed lines in (a) and (c) illustrate that every second maximum of the primary stars RV curve is in phase with every fifth minimum of the perturbations curve.

parameters have their usual meaning). Its parallax has been estimated as 47.18 ± 1.93 mas (van Leeuwen 2007), corresponding to a distance of 21.10 ± 0.87 pc. Following the analysis by Ramm et al. (2009), the masses of the primary and secondary star are given as 1.4 ± 0.3 and 0.5 ± 0.1 $M_{\odot}$, respectively. Ramm et al. (2009) discovered low-amplitude periodic behavior in the residuals of the orbital solution for ν Oct, which they very tentatively attribute to a Jupiter-type planet. If existent, this planet would be located at 1.2 ± 0.1 AU from the primary, i.e., almost midway between the two stellar components, which is very unlikely based on standard orbital stability considerations. Alternative interpretations of the perturbation encompass rotational modulation of surface phenomena or pulsations, which however have been ruled out

with high certainty, implying that the observed signals are caused by a planetary mass.

Ramm et al. (2009) also pursued a tentative estimate of the orbital stability of the suspected planet; however, they did not entertain the possibility of a retrograde orbit for the suspected planet. This will be the main focus of the present analysis. Our paper is structured as follows: In § 3.2, we provide a detailed estimate of the orbital stability regime of the planet, if existing. Additionally, we present orbital simulations of the planet by considering both prograde and retrograde orbits about the main component of the binary system. Our conclusions are given in § 3.4.

3.2 Estimates of the Orbital Stability Regime

The outer limit of orbital stability with respect to the planetary host star, i.e., the primary star of the ν Octantis system, can be estimated based on a statistical fitting formula given by Holman and Wiegert (1999). Previous work on the deduction of fitting formulas as well as subsequent studies of orbital stability of planets in binary systems were given by Dvorak (1986), Rabl and Dvorak (1988), Pilat-Lohinger and Dvorak (2002), Musielak et al. (2005), and others. Holman and Wiegert (1999) obtained empirical expressions based on a large number of simulations for various stellar mass ratios and binary eccentricities with the length of integration given by 10^4 binary periods. They found that the ratio between the critical semimajor axis (to be interpreted as orbital stability limit) and the semi-major binary axis $a_{\rm bin}$ depend roughly linearly in both the mass ratio $\mu = M_2/(M_1+M_2)$ and the binary eccentricity $e_{\rm bin}$, although terms of higher order exist. Note that the given formula is only available for prograde planetary orbits.

Using the parameters for ν Octantis (see Table 3.4), we found that the outer limit of planetary orbital stability is given as 0.240 with a 1σ uncertainty of 0.066.

This is consistent with the value deduced by Ramm et al. (2009), given as 0.25 ± 0.01, and also appears to be commensurate with the results from the circular and elliptical restricted three body problem (e.g., Cuntz et al. 2007; Eberle et al. 2008; Szenkovits and Makó 2008). Note that the uncertainty bar deduced in our study was obtained through a detailed Monte-Carlo approach based on the uncertainties in μ, $a_{\rm bin}$, and $e_{\rm bin}$. In order to maximize the uncertainty bar in the stability limit we also pursued a computation based on an extreme exhaustion of the uncertainties in μ, $a_{\rm bin}$, and $e_{\rm bin}$. In this case, the resulting uncertainty bar in the stability limit is obtained as 0.114; note that although this approach is unrealistic, it has been pursued anyway for the sake of curiosity. The attained stability limit with the two versions for associated uncertainty bars are given in Fig. 3.3.

This approach allows us to estimate the probability of the suggested planet in the ν Octantis system for the particular case of a prograde orbit relative to the orbital motion of the binary components. Figure 3.3 depicts the various limits with respect to the distance ratio ρ_0, which in the circular case denotes the ratio of the initial distance of the planet from its host star (i.e., the stellar primary) to the distance between the binary components. In the elliptical case, ρ_0 denotes the ratio of the initial distance of the planet from the primary (a_p) relative to the semimajor axis of the binary components ($a_{\rm bin}$). Our analytic approach shows that for the realistic, i.e., Monte-Carlo–type estimate the 1σ uncertainty bar barely penetrates the 2σ uncertainty regime of distance ratio ρ_0. The latter has been deduced based on the uncertainties of the stellar masses M_1 and M_2 following the approach by Cuntz et al. (2007) and Eberle et al. (2008). Thus assumed that the obtained uncertainties are consistent with Gaussian distributions, the implied probability for the existence of the suggested planet in the ν Octantis system is only about 1%. Therefore, if in a prograde orbit, the previously suggested planet in ν Octantis is virtually impossible.

3.3 Orbital Stability Simulations

In addition to the usage of statistical fitting formulas for obtaining estimates of orbital stability, we also pursued detailed short-term and long-term simulations. The method of integration was based on a sixth-order symplectic integration scheme as found in, e.g., Yoshida (1990). Previous versions of our code have been extensively tested against known analytical solutions, including the two-body and restricted three-body problem (see Noble et al. 2002; Cuntz et al. 2007; Eberle et al. 2008, for detailed results). In the framework of our simulations that extended up to 1×10^7 yrs, we applied a time-step of 10^{-3} yrs per step. Even for the longest term simulations, this seems to be adequate as there were no close encounters of any of the bodies. Furthermore, the total system energy was found to be conserved within less than 5×10^{-7} accuracy.

We selected three different primary masses (1.1, 1.4, 1.7 $M_{\odot}$) and, furthermore, for each primary mass three different mass ratios (0.2593, 0.2754, 0.2908) that are consistent within the uncertainty limits conveyed by Ramm et al. (2009) for a total of nine mass parameter combinations. In regard to prograde planetary orbital stability, we ran 10^3 year simulations (slightly less than 350 binary orbits), whereas for retrograde planetary orbits, we ran 10^4 year simulations (slightly less than 3.5×10^3 binary orbits). The aim of this effort was to contest and improve any analytically deduced stability limits (only available for prograde orbits; see §3.2) and to motivate long-term orbital stability simulations (if appropriate), which will also be targeted in future studies.

The initial conditions of the system were chosen such that the secondary star and the planet were both at apoapsis relative to the primary star and at the 9 o'clock position. The semimajor axis of the secondary star was chosen so as to keep the period of the binary system fixed at 2.875 years. The distance ratio was varied from 0.22

to 0.54 in 30 equal increments for both prograde and retrograde initial conditions. We then estimated the upper limits of stability based on the cut-off point where simulations survived the 10^3 year duration. The next greater distance ratio resulted in an earlier simulation termination due to close approaches of the planet to one of the stars or ejection from the system altogether. In conclusion, for the prograde setting, the average last stable orbit occurred for an initial distance ratio of 0.251 with a standard deviation of 0.005. The attained distance ratio is very similar to our analytic estimate of 0.240 (see § 3.2).

In order to probe the cut-off point between stability and instability for the retrograde case, we pursued a set of simulations where the initial distance ratio was varied from 0.42 to 0.50 in ten equal increments with the nine mass parameters previously used; the appropriateness of this interval was obtained by previous trial calculations. For this set, the maximum simulation time was set as 10^4 years. In this case, the outermost stable orbit distance ratio was identified as 0.479 with a standard deviation of 0.008, implying an estimated probability of existence for a planet in a retrograde orbit of about 60%, an estimate that also encapsulates the probability distribution of the mass ratio of the stellar components. Evidently, the upper limit of orbital stability for retrograde orbits is about double that for prograde orbits, which is also in line with earlier work by Jefferys (1974). Most importantly, this value is highly consistent with the possibility of a planet in the ν Octantis system in concert with the stellar mass ratio indicated by observations, including its error bar (see Fig. 3.3).

Figure 3.4 depicts examples of detailed time-dependent orbital stability simulations for the ν Octantis system. The masses of the stellar components were assumed as $M_1 = 1.4 M_\odot$ and $M_2 = 0.532 M_\odot$, corresponding to a mass ratio of $\mu = 0.28$. Figure 3.4a shows the prograde planetary orbit as well as the secondary's motion about the primary where the primary is at the origin. After only a few orbits around the

primary it is evident that the planet's orbit is unstable, as expected. A close approach of the secondary star pulls the planet into a more elliptical orbit for a few more passes until another close approach of the secondary pulls the planet away from the primary and to the eventual collision of the planet with the secondary. The total time of this simulation is only 9.88 years, corresponding to approximately three binary periods.

Figure 3.4b shows the retrograde planetary orbit for the same system parameters motivated by the prospect of orbital stability based on our previous short-term investigation. This simulation was continued for a timespan of 1×10^7 yrs, corresponding to about 3.4×10^6 binary periods. The orbit of the planet about the primary obviously remains within a ring that has a center that is slightly to the left of the origin. This is an unequivocal indication of long-term orbital stability of the planet, if existent, for system parameters compatible with the observational constraints. To further strengthen this interpretation, we also assessed the time line data for the distances between the primary star and the secondary star as well as the primary star and the planet for the retrograde planetary orbit for a 1×10^7 yrs time period. The motion of the secondary relative to the primary is essentially a two-body Keplerian motion along an ellipse with an eccentricity equal to 0.236. These data were sampled in 100 year time increments, resulting in very well defined upper and lower limits corresponding to the planetary apoapsis and periapsis, respectively. The distances between the stars vary between 1.9288 and 3.1125 AU, whereas the distances between the planet and the primary star vary between 0.6452 and 1.2879 AU, an effect due to the ellipticity of the stellar and planetary orbits and by no means indicative of any long-term changes in the system.

3.4 Conclusions

Our study has been aimed at exploring an enigmatic finding about the ν Octantis binary system that indicates the possible existence of a Jupiter-type planet, even though the planet seems to be located outside the zone of orbital stability. Ramm et al. (2009) discovered low-amplitude periodic behavior in the residuals of the orbital solution for this star, which they very tentatively attribute to a Jupiter-type planet. Alternative interpretations of the perturbation consist in rotational modulation of surface phenomena or pulsations, which however have been ruled out with high certainty.

In our study, we performed a detailed analysis of orbital stability based on previous studies that carefully considers the ν Octantis system parameters including their observationally deduced uncertainties. We confronted the probability distribution of the parameter space of the system with the requirements of planetary orbital stability. Our results indicate that the suggested planet if in a prograde orbit with respect to the motion of the binary components is virtually impossible; however, the estimated probability of existence for a planet in a retrograde orbit is nearly 60%. This estimate further increases if a relatively low stellar mass ratio (within the error bars) is assumed. The possibility of a planet in a retrograde orbit is also consistent with long-term orbital stability simulations spanning 1×10^7 yrs. Based on time-dependent studies, the stability limit of prograde orbits was identified as 0.251, whereas stability limit of retrograde orbits was identified as 0.479; note that the latter value is fully consistent with observationally deduced masses of the stellar companions including the conveyed uncertainty bars. Furthermore, the possibility of a planet in a retrograde planetary orbit in a binary system like ν Octantis is also consistent with earlier studies by Jefferys (1974) indicating significantly enlarged regions of stability

for planets in retrograde orbits. This study had been based on a Henon stability analysis while considering the circular restricted three body problem.

The distance ratio that would correspond to the 5:2 resonance obviously falls within the error bars for the case of an unstable prograde orbit; note that it is related to how Ramm et al. (2009) determined the semimajor axis of the proposed planet in the first place. While we did not specifically deduce the exact parameters for this resonance for the various considered models, we made a strong case for the retrograde orbit as potentially the most likely candidate explanation. The fact that the planet, if existing, appears to be in a resonance orbit with the binary system could be due to the unknown mechanism of how this proposed planet came to be in this orbit. Thus, it is still possible, albeit highly unlikely, that the planet, if existing, could be in a prograde orbit and the particular orbital elements conspire to produce a stable orbit with the observed resonance. Evidently, additional observational and theoretical studies are required concerning ν Octantis to elucidate the possible existence of a planet in that system.

There is another interesting observation with some relevance to our study. In our Solar system the planets have orbits with relatively low inclination, both in relation to the Sun's equator and each other. It came as a bit of a surprise when Queloz et al. (2010) reported the discovery of WASP-8b, a retrograde transiting planet located in a multiple stellar system, although in this system the planetary retrograde motion occurs with respect to the stellar rotation of the planetary host star rather than the orbital motion of any of the two stellar companions. Akin to the suspected planet in the ν Octantis system, WASP-8b is a Jupiter-type planet of $2.25 \pm 0.08\ M_J$ in a significantly eccentric orbit.

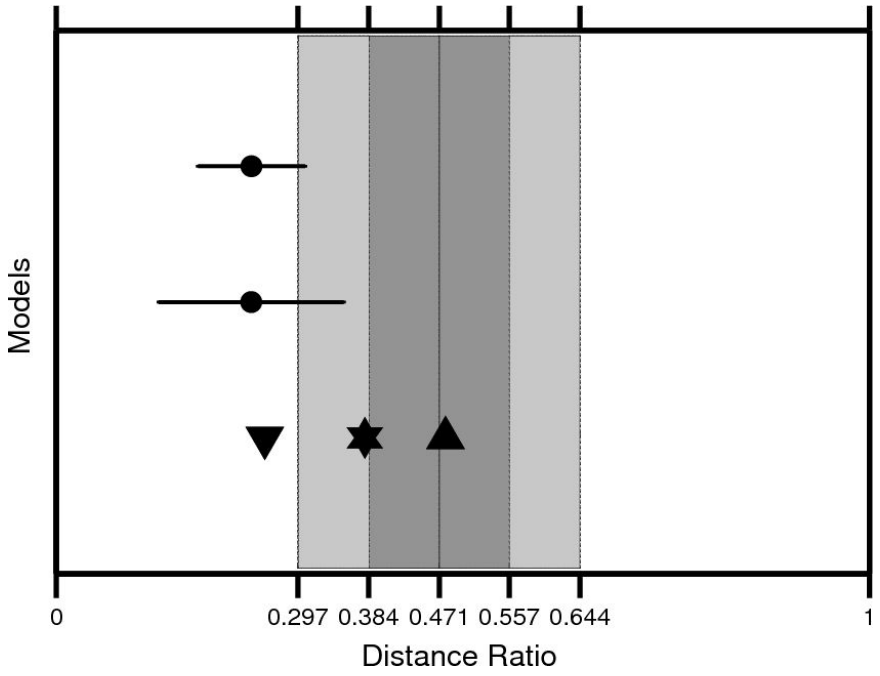


Figure 3.3. Models used for ν Octantis. The two dots indicate detailed estimates of the outer orbital stability limit for the suspected planet in the ν Octantis system in prograde orbit based on the statistical fitting formula by Holman and Wiegert (1999); here we adopted two different approaches for estimating the uncertainty. Note that the stellar primary is placed at the origin, whereas the stellar secondary is positioned at 1. The two triangular symbols highlight the deduced stability limits of the suspected planet in prograde (*downward triangle*) and retrograde (*upward triangle*) orbits based on short-term simulations, whereas the stellated hexagon indicates the ρ_0 parameter used in our long-term simulation (10^7 yrs). The five symbols are superimposed on domains depicting the estimated region of orbital stability for the planet based on the stellar masses. With $M_1 = 1.4M_\odot$ and $M_2 = 0.5M_\odot$, the limit of orbital stability is given as $\rho_0 = 0.471$. The dark and light gray domains depict the 1σ and 2σ uncertainty regimes of ρ_0, respectively, given by the uncertainty in the stellar mass ratio (see Table 3.4). The latter uncertainty regimes are thus inherently separate from the facilitation of the planetary orbital stability simulations.

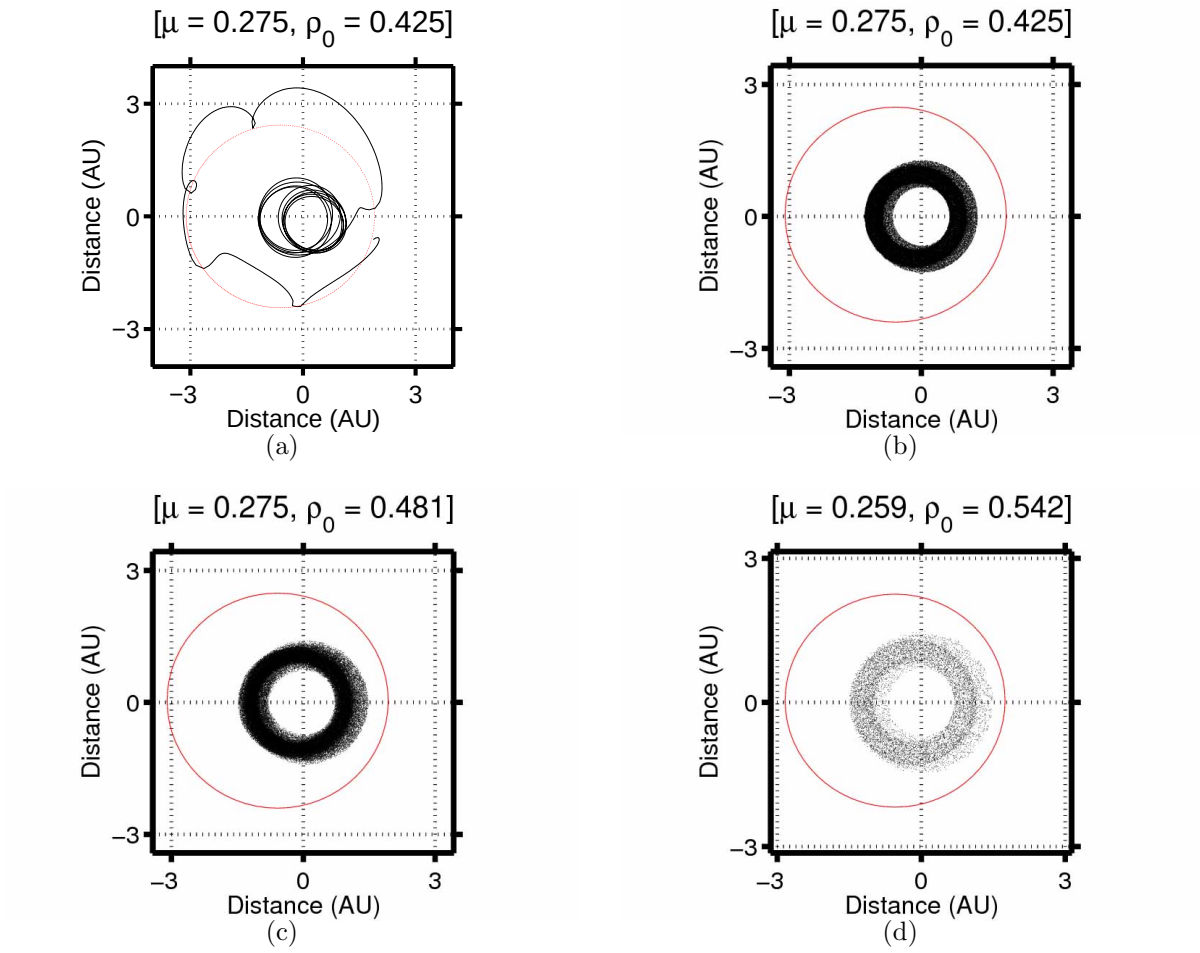


Figure 3.4. Four simulations of orbital stability for a Jupiter-type planet in the ν Octantis system using slightly different parameters. (a) Example of a prograde planetary orbit (*black*) together with the resulting secondary's motion about the primary (*red ellipse*) where the primary is at the origin. Evidently, the planet's orbit is highly unstable, a result obtained after a very short amount of time (9.88 yrs). (b) Example of a retrograde planetary orbit based on the same system parameters as before. The simulation spans 1×10^7 yrs, corresponding to 3.4×10^6 binary periods. Note that the planet about the primary remains within a ring centered slightly to the left of the origin. (c) The only thing that has changed is that the planet starts out further from the primary. (d) Starting at an even larger distance, but the mass ratio has been reduced and the simulation only lasts 1×10^6 yrs, but the behavior is similar to the previous figure.

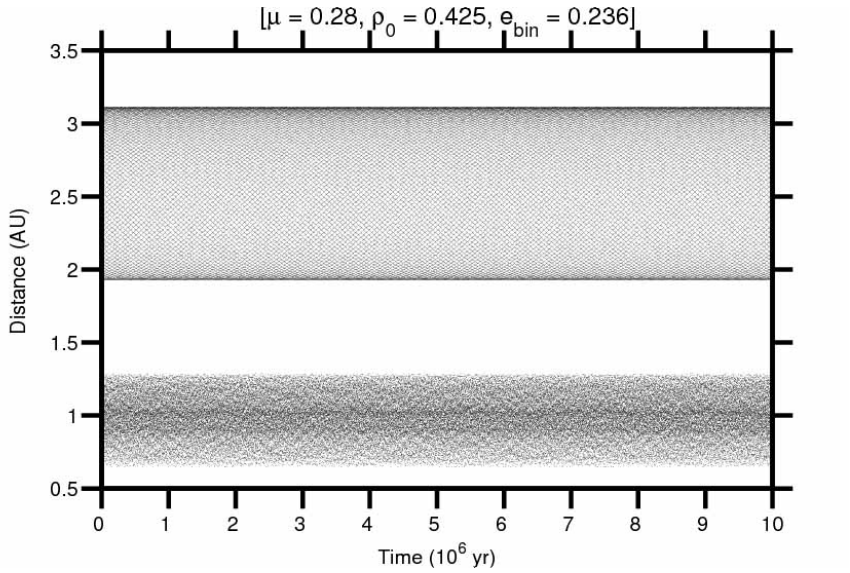


Figure 3.5. Distance between primary star and the secondary star (upper) and the planet (lower) over time.

Table 3.1. Stellar and Planetary Parameters for ν Octantis

Parameter	Value	Reference
Spectral Type (1)	K1 III	Houk and Cowley (1975)
Spectral Type (2)	K7–M1 V	Ramm et al. (2009)
RA	21 41 28.6463	European Space Agency (1997)[a,b]
DEC	− 77 23 24.167	European Space Agency (1997)[a,b]
Distance (pc)	21.20 ± 0.87	van Leeuwen (2007)
M_V (1)	2.10 ± 0.13	[c]
M_V (2)	$\sim$ 9.9	Drilling and Landolt (2000)
M_1 ($M_\odot$)	1.4 ± 0.3	Ramm et al. (2009)
M_2 ($M_\odot$)	0.5 ± 0.1	Ramm et al. (2009)
$T_{\rm eff,1}$ (K)	4790 ± 105	Prieto and Lambert (1999)
R_1 ($R_\odot$)	5.9 ± 0.4	Prieto and Lambert (1999)
$P_{\rm bin}$ (d)	1050.11 ± 0.13	Ramm et al. (2009)
$a_{\rm bin}$ (AU)	2.55 ± 0.13	Ramm et al. (2009)
$e_{\rm bin}$	0.2358 ± 0.0003	Ramm et al. (2009)
$M_p{\rm sin}i$ (M_J)	2.5 ± ...	Ramm et al. (2009)[d]
a_p (AU)	1.2 ± 0.1	Ramm et al. (2009)[d]
e_p	0.123 ± 0.037	Ramm et al. (2009)[d]

Note. — [a]data from SIMBAD, `http://simbad.u-strasbg.fr`

[b]adopted from the *Hipparcos* catalogue

[c]derived from the stellar parallax, see Ramm et al. (2009)

[d]controversial

CHAPTER 4

PROJECT THREE: THE POSSIBILITY OF HABITABLE TROJAN PLANETS IN THE SYSTEM HD 23079

4.1 Introduction

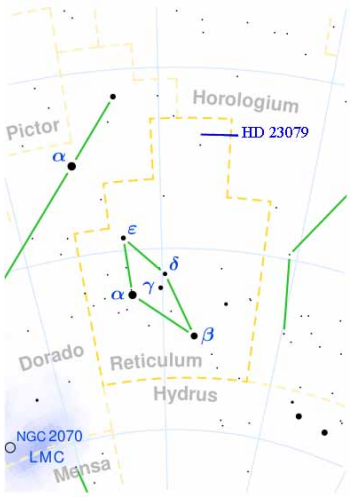


Figure 4.1. The Reticulum constellation.

About one and a half decades have passed since the detection of the first planet hosted by a solar-type star other than the Sun. The field of research on extra-solar planets and associated aspects of exobiology has considerably matured in the meantime. One important aspect with significant ramifications for the search of life outside the Solar System concerns the identification of Earth-mass planets in stellar habitable zones (HZs), including detailed investigations regarding their orbital stability.

The Anglo-Australian Planet Search (AAPS) is a planet detection program performing extrasolar planet detection and measurement with a long-term, systematic radial velocity precision of 3 m/s or better and discovered the first extrasolar planet orbiting beyond 0.2 AU that is in a circular orbit similar to solar system planets (Butler et al. 2001). HD 23079 is a target star of this program, and it too was found to have a Jupiter mass planet in a relatively large circular orbit (Tinney et al. 2002). HD 23079 is an inactive main-sequence star; Houk and Cowley (1975) classify it as an intermediate between F8 and G0 star. Based on this and a stellar effective temperature of $T = 6030 \pm 52$ K and a stellar radius of $R = 1.106 \pm 0.022\ R_{\odot}$ (Ribas et al. 2003), the stellar mass can be estimated as $M = 1.10 \pm 0.15\ M_{\odot}$; HD 23079 is very similar to our Sun. The planet (HD 23079 b) has a minimum mass of $M_p \sin i = 2.45 \pm 0.21\ M_J$ and a semimajor axis of $a_p = 1.596 \pm 0.093\ (AU)$ and eccentricity of $e_p = 0.102 \pm 0.031$ with an orbital period of $P = 730.6 \pm 5.7$ days (Butler et al. 2006). These orbital parameters are very similar to those of Mars, thus making it difficult for a terrestrial planet to orbit HD 23079 at a similar distance without being affected by the influence of the giant planet. Dvorak et al. (2004) investigated the dynamical stability of a hypothetical terrestrial planet at the equilateral points of this system and found that there is a good chance for Trojan planets in the 1:1 resonance with the giant planet to stay on stable orbits.

There is one aspect, however, that has received little attention by previous studies, which is the dependency of the ejection time on the starting position of the planet, especially in systems where the orbital stability is not guaranteed. An existing example includes the work by Noble et al. (2002). They investigated the orbital stability of terrestrial planets inside the HZs of three stellar systems, which are: 51 Peg, 47 UMa and HD 210277. The center stars are relatively similar to the Sun concerning mass, spectral type, and effective temperature. Orbital stability was

attained for terrestrial planets in the HZ of 51 Peg and a significant portion of the HZ of 47 UMa; however, no orbital stability was found for hypothetical Earth-mass planets in the HZ of HD 210277. In this case, a Jupiter-type planet crosses into the stellar HZ, thus effectively thwarting habitability for this system.

In regard to HD 210277, Noble et al. (2002) varied the starting position of the terrestrial planet, originally placed at the center of the HZ, in 30° increments from 0° to 180°. They found that the ejection time of the terrestrial planet was roughly five to six decades with one exception that was less than a year. It is the main motivation of the present work to extend this type of study to other systems, namely HD 23079. We intend to identify properties of the ejection times of the terrestrial planet depending on the starting positions of the giant and terrestrial planets, including the relative angles (phases) between these objects.

Note that our current study exhibits some inherent similarities to the previous work by Fatuzzo et al. (2006). They provided a detailed statistical analysis of ejection times, encompassing about 500 orbital configurations, for Earth-mass planets in binary systems. This study included variations of the companion mass, the companion eccentricity, the companion periastron, and the planet's inclination angle i relative to the stellar binary plane. The authors identified distributions of ejection times τ_s (in yrs). With respect to log τ_s they typically turned out to be Gaussian in width, although the longest-lived systems displayed substantial (non-Gaussian) tails. This latter result indicates that seemingly unstable systems may still hold some merit in the study and realization of planetary habitability.

4.2 Methods and System Parameters

For our simulations concerning the HD 23079 system, we consider both the observed giant planet and a hypothetical terrestrial planet of one Earth-mass, i.e.,

$3.005 \times 10^{-6}\ M_{\odot}$, which allows us to execute a grid of model simulations. The method of integration uses a fourth-order Runge-Kutta integration scheme (Garcia 2000). The code has been extensively tested against known analytical solutions, including the two-body and restricted three-body problem (see Noble et al. 2002; Cuntz et al. 2007; Eberle et al. 2008, for detailed results). In the framework of our simulations that are limited to about 10^6 yrs due to the ejection of the Earth-mass planet from the system, we apply a time-step of 10^{-4} yrs for the integration scheme. This value is found to be fully appropriate. A case study comparing the planetary orbits based on three different integration time-steps, i.e., 10^{-3}, 10^{-4} and 10^{-5} yrs, is given in Fig. 4.2 where ΔR_{ij} is the magnitude of the difference between the position of the planet when different step sizes 10^{-i} and 10^{-j} are used. The initial conditions for the orbit of the Earth-mass is chosen such that it is assumed to be in a circular orbit about the star at the midpoint of the HZ (1.4779 AU), although it is evident that it will be significantly affected immediately by the giant planet preventing from continuing its circular motion. The Earth-like planet was placed in an initially circular orbit in the middle of the HZ at 8 starting angles in increments of 45°. The maximum extent of the error bounds in the giant planet's semimajor axis and eccentricity were used, in addition the giant planet is started in either its periastron or apastron position. In total there are 144 initial conditions that were simulated.

4.2.1 Stellar Habitable Zones

Following Kasting et al. (1993), the inner and outer edges of the HZs are assumed to be located at 0.9896 and 1.9662 AU, respectively, from the star. The underlying definition of habitability is based on the assumption that liquid surface water is a prerequisite for life, a key concept that is also the basis of ongoing and future searches for extrasolar habitable planets (e.g., Catanzarite et al. 2006; Cock-

Table 4.1. Stellar and Planetary Parameters for HD 23079

Parameter	Value	Reference
Spectral Type	F8/G0 V	Houk and Cowley (1975)
RA	3 39 43.0952	European Space Agency (1997)[a,b]
DEC	− 52 54 57.017	European Space Agency (1997)[a,b]
$T_{\rm eff}$ (K)	6030 ± 52	Ribas et al. (2003)
R ($R_\odot$)	1.106 ± 0.022	Ribas et al. (2003)
M ($M_\odot$)	1.10 ± 0.15	[c]
M_V	4.42 ± 0.05	European Space Agency (1997)[a,b]
$M_{\rm bol}$	4.25 ± 0.05	European Space Agency (1997)[a,b]
Distance (pc)	34.60 ± 0.67	European Space Agency (1997)[a,b,d]
$M_p \sin i$ (M_J)	2.45 ± 0.21	Butler et al. (2006)
P (days)	730.6 ± 5.7	Butler et al. (2006)
a_p (AU)	1.596 ± 0.093	Butler et al. (2006)
e_p	0.102 ± 0.031	Butler et al. (2006)

Note. — [a]data from SIMBAD, `http://simbad.u-strasbg.fr`

[b]adopted from the *Hipparcos* catalogue

[c]based on spectral type

[d]based on parallax 28.90 ± 0.56 mas

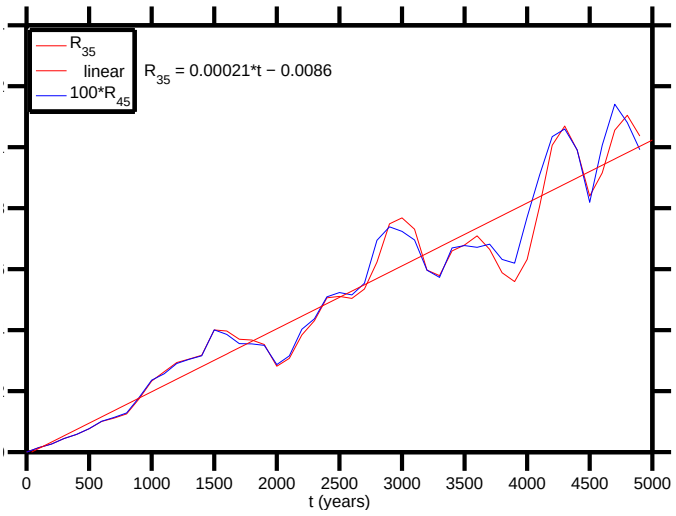


Figure 4.2. The time dependence of ΔR_{ij}. ΔR_{45} retains the same character as ΔR_{35} but is 100 times smaller.

ell et al. 2009). Note that the numerical evaluation of these limits is based on an Earth-type planet with a $CO_2/H_2O/N_2$ atmosphere.

The inner limit of habitability is set by the loss of water from the upper planetary atmosphere through photodissociation and subsequent escape of hydrogen to space associated with a run-away greenhouse effect. The outer limit of habitability is given by the maximum greenhouse effect (Kasting et al. 1993; Underwood et al. 2003); here a surface temperature of 273 K can be maintained by a cloud-free CO_2 atmosphere. We would also like to point out that concerning the outer edge of habitability, even less conservative limits have been proposed in the meantime (e.g., Forget and Pierrehumbert 1997; Mischna et al. 2000). They are based on the assumption of relatively thick planetary CO_2 atmospheres and invoke strong backwarming that is further enhanced by the presence of CO_2 crystals and clouds. However, as these limits, which can be as large as 2.4 AU for solar-like stars, depend on distinct properties of the planetary atmosphere, they are not relevant for our study. Moreover, the

significance of these limits have recently been challenged based on detailed radiative transfer simulations (Halevy et al. 2009).

4.3 Results and Discussion

4.3.1 Case Studies of Habitable Trojan Planets

Table 4.2 and Table 4.3 summarize the time that the Earth-mass planet remained within the HZ. Of the 144 total considered initial conditions 13 survived at least 1 million years, 93 crossed the upper limit of the HZ, 28 crossed the lower limit of the HZ, and 10 collided with the giant planet. Of the 13 survivors, 12 are Trojan types, that is they exist in stable orbits around the equilateral equilibrium positions much like what what demonstrated in Dvorak et al. (2004). In the cases where the giant planet is initially in the periastron position, only the smallest semimajor axis and eccentricity combination results in habitable Trojan planets and these are shown in Fig. 4.3. It is clear that the Earth-mass planet is safely inside of the HZ but it is a snug fit. Under similar circumstances but with the next larger eccentricity of 0.102 the Earth-mass remains within the HZ for several hundred-thousands of years before finally crossing the upper limit of the HZ and when the eccentricity is increased again to 0.133 the Earth-mass planet remains within the HZ for only hundreds of years. The situation is similar for the cases where the giant planet was started in the apastron position with the exception that the Earth-mass planet remained in the HZ for at least a million years for two eccentricities and these are shown in Fig. 4.4 and Fig. 4.5. Comparing Fig. 4.5 to Fig. 4.4 it is clear that the Earth-mass planet moves in a wider area and approaches the edges of the HZ for the larger eccentricity, thus illustrating how the planet remained in the HZ for such a long time before exiting in the periastron case with the same parameters. It is possible that the planet was outside of the

Table 4.2. Time terrestrial planet remained within HZ (years) with HD 23079 b starting at periastron

a \ e	0°	45°	90°	135°	180°	225°	270°	315°
1.503 \ 0.071	3.26E-01^{U}	>1.00E+06	>1.00E+06	1.67E+02^{L}	1.05E+02^{L}	5.38E+01^{L}	>1.00E+06	>1.00E+06
1.503 \ 0.102	1.84E+00^{C}	2.30E+05^{U}	7.09E+05^{U}	8.51E+01^{U}	4.84E+01^{L}	4.27E+01^{U}	8.78E+05^{U}	2.59E+05^{U}
1.503 \ 0.133	1.07E+00^{L}	3.46E+02^{L}	6.13E+02^{L}	9.72E+01^{U}	6.04E+01^{U}	4.30E+01^{U}	4.78E+02^{U}	1.85E+03^{U}
1.596 \ 0.071	1.30E-03^{C}	2.33E+02^{U}	1.54E+01^{U}	9.48E+00^{U}	7.93E+00^{U}	5.59E+00^{U}	3.89E+00^{U}	1.95E+00^{U}
1.596 \ 0.102	4.83E-01^{U}	1.53E+01^{U}	1.53E+01^{U}	1.23E+01^{U}	7.80E+00^{U}	5.56E+00^{U}	3.73E+00^{U}	1.77E+00^{U}
1.596 \ 0.133	>1.00E+06	3.38E+01^{L}	1.52E+01^{U}	9.48E+00^{L}	7.82E+00^{U}	6.02E+00^{U}	3.90E+00^{U}	1.42E+00^{C}
1.689 \ 0.071	1.09E-01^{C}	8.47E+00^{U}	8.99E+00^{U}	7.25E+00^{L}	4.86E+00^{U}	4.05E+00^{U}	2.64E+00^{U}	3.90E+01^{L}
1.689 \ 0.102	2.85E-02^{C}	8.02E+00^{C}	8.56E+00^{L}	6.73E+00^{U}	4.78E+00^{U}	4.38E+00^{L}	2.45E+00^{U}	2.18E+00^{L}
1.689 \ 0.133	5.90E-03^{C}	8.93E+00^{U}	1.91E+01^{U}	6.55E+00^{L}	4.59E+00^{U}	8.97E+00^{U}	2.23E+00^{U}	8.13E+00^{U}

Note. — ^{U}Earth-like planet crossed the upper limit of the HZ at 1.9662 AU.

^{L}Earth-like planet crossed the lower limit of the HZ at 0.9896 AU.

^{C}Earth-like planet made a close approach or collided with giant planet.

HZ for only a brief time and the planet could still be at least marginally habitable but once the planet crossed the HZ the simulation was halted so this is impossible to quantify.

4.3.2 On the Possibility of Habitable Moons

The remaining survivor occurred for an intermediate value of the giant planet's semimajor axis and highest eccentricity when it was initially at periastron (see Fig. 4.6). In this case the Earth-like planet turned out to orbit the giant planet with a perigee of 0.004 AU and an apogee of 0.098 AU. The Earth-like planet's semimajor axis relative to the giant planet is ≈ 0.051 AU with an eccentricity of ≈ 0.8. With a uniform data sampling rate, the moon will be more likely to be recorded at or near apogee. From the right side of Fig. 4.6 it is evident that there is a precession of the apogee in a retrograde sense with a period of ≈ 30 years.

Table 4.3. Time terrestrial planet remained within HZ (years) with HD 23079 b starting at apastron

a \ e	0°	45°	90°	135°	180°	225°	270°	315°
1.503 \ 0.071	8.50E-01^{U}	>1.00E+06	>1.00E+06	2.95E+01^{U}	5.08E+01^{L}	4.01E+01^{U}	>1.00E+06	>1.00E+06
1.503 \ 0.102	1.45E+00^{U}	>1.00E+06	>1.00E+06	9.94E+01^{L}	2.32E+01^{L}	2.11E+01^{U}	>1.00E+06	>1.00E+06
1.503 \ 0.133	9.33E-01^{L}	1.28E+03^{L}	1.46E+03^{U}	2.54E+01^{L}	4.99E+01^{U}	8.52E+00^{U}	4.19E+03^{U}	7.58E+02^{U}
1.596 \ 0.071	7.07E+00^{L}	2.57E+02^{U}	3.27E+01^{L}	5.60E+01^{C}	6.61E+00^{U}	4.97E+00^{U}	2.86E+01^{U}	4.18E+01^{U}
1.596 \ 0.102	1.33E+01^{U}	3.44E+01^{U}	5.33E+01^{U}	5.34E+01^{U}	6.24E+00^{C}	4.77E+00^{U}	6.49E+01^{U}	2.44E+01^{U}
1.596 \ 0.133	7.07E+00^{U}	2.05E+01^{U}	7.08E+01^{U}	3.23E+01^{U}	1.73E+01^{U}	5.07E+00^{U}	2.26E+01^{U}	2.08E+01^{U}
1.689 \ 0.071	1.89E+01^{C}	3.61E+01^{U}	7.76E+00^{U}	6.45E+00^{U}	5.17E+00^{U}	3.61E+00^{U}	3.61E+00^{U}	1.38E+00^{U}
1.689 \ 0.102	7.48E+00^{L}	1.00E+01^{U}	7.59E+00^{U}	7.32E+00^{U}	5.52E+00^{L}	3.46E+00^{U}	3.32E+00^{U}	1.23E+00^{U}
1.689 \ 0.133	1.87E+01^{U}	9.64E+00^{U}	7.41E+00^{U}	7.02E+00^{L}	5.53E+01^{L}	3.27E+00^{U}	3.44E+00^{L}	1.08E+00^{U}

Note. — ^{U}Earth-like planet crossed the upper limit of the HZ at 1.9662 AU.

^{L}Earth-like planet crossed the lower limit of the HZ at 0.9896 AU.

^{C}Earth-like planet made a close approach or collided with giant planet.

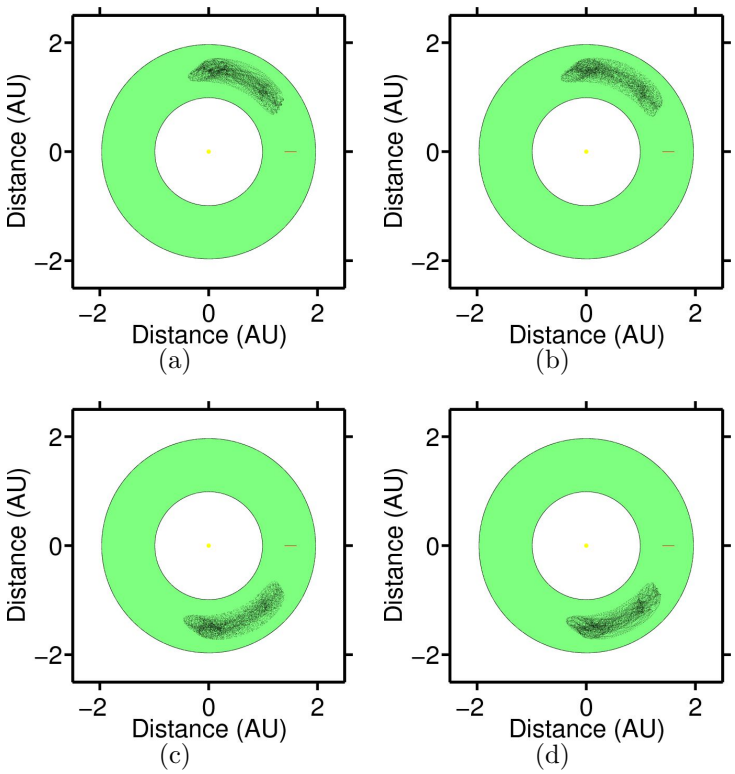


Figure 4.3. HD 23079 b initially at periastron (a: 1.503, e: 0.071) with Trojan planets initially at 4 different starting angles. Earth type planet initially at 45° (a), 90° (b), 270° (c), and 315° (d). Using a rotated system of coordinates, HD 23079 b moves along the red line and HD 23079 is the yellow dot at the origin. The Earth-like planet remained within the HZ for at least 1e6 years.

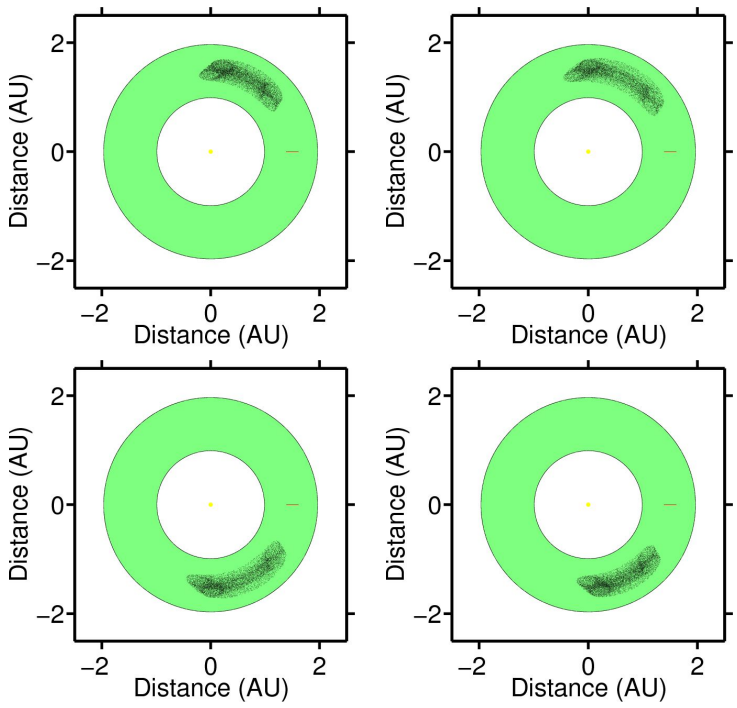


Figure 4.4. HD 23079 b initially at apastron (a: 1.503, e: 0.071) with Trojan planets initially at 4 different starting angles. Earth type planet initially at 45° (top left), 90° (top right), 270° (bottom left), and 315° (bottom right). Using a rotated system of coordinates, HD 23079 b moves along the red line and HD 23079 is the yellow dot at the origin. The Earth-like planet remained within the HZ for at least 1e6 years.

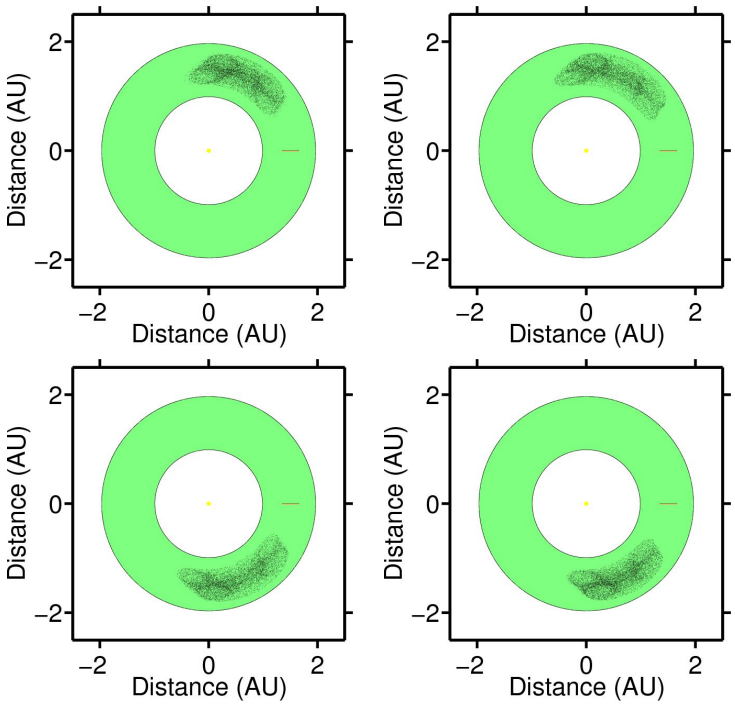


Figure 4.5. HD 23079 b initially at apastron (a: 1.503, e: 0.102) with Trojan planets initially at 4 different starting angles. Earth type planet initially at 45° (top left), 90° (top right), 270° (bottom left), and 315° (bottom right). Using a rotated system of coordinates, HD 23079 b moves along the red line and HD 23079 is the yellow dot at the origin. The Earth-like planet remained within the HZ for at least 1e6 years.

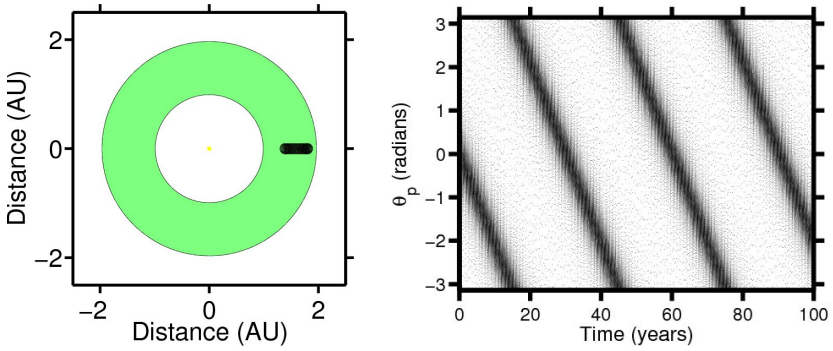


Figure 4.6. HD 23079 b initially at periastron (a: 1.596, e: 0.133) with planet initially at 0°. Using a rotated system of coordinates, HD 23079 b moves along the red line and HD 23079 is the yellow dot at the origin. The Earth-like planet remained within the HZ for at least 1e6 years. In this case, the Earth-like planet was captured by the giant planet and became a moon.

CHAPTER 5

SUMMARY

The first and most comprehensive project of this dissertation depends on the fact that the two stars are orbiting each other in a circle; if the two stars orbit each other in an ellipse, the problem is more difficult. The second project, which studies the real elliptical binary system of ν Octantis with a possible giant planet orbiting one of the stars, is just a small example. The constant of motion that we were able to use to describe the available region of motion no longer exists in the elliptical problem, but Holman and Wiegert (1999) were able to find empirical limits through long term numerical simulations. This and many other studies of planets in binary systems have neglected the possibility that the planet has a retrograde orbit with respect to the orbital motion of the binaries despite the observed phenomenon that retrograde orbits can stably exist for a wider range (Jefferys 1974). The first two projects both assumed that the smallest mass was in orbit of the largest which is a bit of an over restriction. If we consider M_2 to be the "other" star even if it is the larger of the two, then the mass ratio range is extended to include $0.5 < \mu < 1.0$ and in this scenario, the planet would be in orbit around the smaller star so as the mass ratio increases, we can expect that the stable region would continue to decrease. If the planet is in a circumbinary orbit then the mass ratio can be kept at one half or lower as the only difference is a matter of phase. All of the studies presented in this dissertation suffer from the prejudice that all three masses should remain in the same plane, real systems are obviously more complicated than this. The difficulties involved are well worth taking on since, as recent discoveries have indicated, real extra solar planetary

systems are more diverse than we would have previously imagined. The number of extrasolar planetary systems continues to rise and there are still many questions as to how these systems form. Knowing how planets form in single or multiple star systems, and whether they have stable orbits or not is fundamental to our understanding of our place in the universe.

APPENDIX A

COMPUTER CODE

A.1 Introduction

As mentioned previously, a 6^{th} order symplectic integration scheme (Yoshida 1990) was used in order to investigate the restricted three body problem. The code used is given in the following sections and was built upon the code in my masters thesis which was reworked from the example that are provided in Garcia (2000). As can be seen from the following verbatim transcript of the computer code used to integrate the three body problem, there is no implementation of any kind of stability criteria or estimation whatsoever. The only generous escape protocol that was introduced is if the kinetic energy exceeds the magnitude of the potential energy by a factor of two then the program is halted.

A.2 Main body of code

```
#include <cstdlib>
#include <math.h>
#include <ctime>       //for timestamp data files
#include "mkdirtree.cpp"
#include "CBody.cpp"
#include "LagrangePoints.cpp"
#include "RunSim.cpp"

int main(int argc, char *argv[]) {
const int nBodies = 3; //number of bodies
const double pi = acos(-1);
const double G = 4*pi*pi;
CBody plan[nBodies+1];   //the +1 is so I can count more intuitively
double param[10];
double TotMass = 1;
double mu;
double TimePeriod = 1000;    //Sim duration default 1000 years
double DSP = 0.00;           //Data Sample period
double tau = 1e-4;
```

```
    cout<<"Enter type of initial condition:\n\n"
        <<"\t1: Initial Circle about more massive star.\n"
        <<"\t2: Initial Circle about less massive star.\n"
        <<"\t3: Starting stationary at L4 point.\n"
        <<"\t4: From Data.in file.\n"
        <<"\t5: From Data.in file and for range of Impossible Habitable Planets.";
    int type;
    cin>>type;
    switch(type)
    {
    case 1:
    param[0] = 1;       //Orbit around Primary
    break;
    case 2:
    param[0] = 2;       //Orbit around Secondary
    break;
    case 3:
    param[0] = 3;       //Starting stationary at L4 point
    break;
    case 4:
    param[0] = 4;       //From Data.in file
    break;
    case 5:
    param[0] = 5;       //From Data.in and 5 distances and several angles.
    break;
    default:cerr<<"You didn't enter a valid option.\n";
            cout<<"Press any key to quit."; cin.ignore(2); exit(1);
    }

if(param[0] == 1 || param[0] == 2) {
    param[7] = 0;  //no flags
      cout<<"Would you like to enter in a range of Mass Ratios? (Y/N): ";
    char muchoice;
    double mulowlimit, muuplimit, muinc;
      while((muchoice != 'N' && muchoice != 'n') && (muchoice != 'Y' && muchoice != 'y'))
      {
      cin>>muchoice;
      if(muchoice == 'N' || muchoice == 'n')
```

```
  {
  double tempor;
  cout<<"Enter mass of primary star in Solar mass: ";
  cin>>tempor; plan[1].SetMass(tempor);
  cout<<endl<<"Enter mass of secondary star in Solar mass: ";
  cin>>tempor; plan[2].SetMass(tempor);
  TotMass = plan[1].GetMass()+plan[2].GetMass();
  mulowlimit = plan[2].GetMass()/(TotMass);
  muuplimit = mulowlimit;
  muinc = 1;
  }
  else if(muchoice == 'Y' || muchoice == 'y')
  {
  cout<<"Enter total mass in Solar mass: ";
  cin>>TotMass;
  cout<<"Enter mass ratio lower limit: ";
  cin>>mulowlimit;
  cout<<"Enter mass ratio upper limit: ";
  cin>>muuplimit;
  cout<<"Enter mass ratio increment size: ";
  cin>>muinc;
  mu = mulowlimit; //just to start
  plan[1].SetMass((1-mu)*TotMass);
  plan[2].SetMass(mu*TotMass);
  }
  else
      cout<<"Y or N only please!\n";
  }//while choice isn't yes or no.

cout<<endl<<"Enter semimajor axis of stars in AU: ";
cin>>param[5];
cout<<endl<<"Enter eccentricity of stars orbit: ";
cin>> param[6];

double lowlimit, uplimit, inc;
char choice;

  cout<<"Would you like to enter in a range of Initial Conditions? (Y/N): ";
```

```
      while((choice != 'N' && choice != 'n') && (choice != 'Y' && choice != 'y'))
      {
      cin>>choice;

      if(choice == 'N' || choice == 'n')
      {
      cout<<"Enter distance of planet from primary star in AU: ";
      cin>>param[4];

      lowlimit = param[4];
      uplimit = param[4];
      inc = 1;
      }
      else if(choice == 'Y' || choice == 'y')
      {
      cout<<"Enter lower limit: ";
      cin>>lowlimit;
      cout<<"Enter upper limit: ";
      cin>>uplimit;
      cout<<"Enter increment size: ";
      cin>>inc;
      }
      else
          cout<<"Y or N only please!\n";
      }//while choice isn't yes or no.

char Tchoice;

while(DSP < tau) { while((Tchoice != 'Y' && Tchoice != 'y')) {
 cout<<"Enter Desired simulation time in years: ";
 cin>>TimePeriod;

           cout<<"\nEnter DSP (data sample period) in years. ";
           cin>>DSP;
           cout<<"\nEnter size of timestep (years/step): ";
           cin>>tau;
           if(DSP < tau)
           cout<<"\nThe DSP (data sample period) must be >= "<<tau<<endl;
```

```
 cout<<"\nThis will result in "<<TimePeriod/DSP<<" lines of data.";
 cout<<"\nIs this acceptable? ";
 cin>>Tchoice;
} }

for(mu = mulowlimit; mu <= muuplimit; mu += muinc) {
        plan[1].SetMass((1-mu)*TotMass);
        plan[2].SetMass(mu*TotMass);
        //plan[3].SetMass(1e-6*plan[1].GetMass());
        plan[3].SetMass(2.387e-3);  //for nu Octantis
        //plan[3].SetMass(0.0);
    for(int m=1; m<=3; m++)
        param[m]=plan[m].GetMass();

for(param[4] = lowlimit; param[4] <= uplimit; param[4] += inc) {

int num;
double phi = 0.0;
double a = param[5];
double factor = G*(TotMass)/a;
//double estar = 0.0;
//double eplan = 0.0;
//for nu octantis:
double estar = param[6];
double eplan = 0.12;

     //elliptical ap-start
     plan[1].SetPos(mu*a*(1+estar), 0.0, 0.0);
     plan[1].SetVel(0.0, sqrt(factor*((1-estar)/(1+estar)))*mu, 0.0);

     plan[2].SetPos((mu-1)*a*(1+estar), 0.0, 0.0);
     plan[2].SetVel(0.0, -sqrt(factor*((1-estar)/(1+estar)))*(1-mu), 0.0);

     if(param[0] == 1)
     //inside elliptical ap-start
     plan[3].SetPos(-param[4]*(1+eplan), 0.0, 0.0);
     //outside elliptical ap-start
     //plan[3].SetPos(param[4]*(1+eplan), 0.0, 0.0);
     else if(param[0] == 2 || param[0] == 3)
```

```
      plan[3].SetPos(-param[4],0.0,0.0);

      if(param[0] == 1)
      phi = 90;
      else if(param[0] == 2 || param[0] == 3)
      phi = 270;
      double stitch = G*plan[1].GetMass()/fabs(plan[3].GetPos(X));
      //elliptical ap-start
      stitch = sqrt(stitch*((1-eplan)/(1+eplan)));
      //inside retrograde or //outside prograde
      plan[3].SetVel(stitch*cos(phi*pi/180), stitch*sin(phi*pi/180), 0.0);
      //inside prograde or //outside retrograde
      //plan[3].SetVel(-stitch*cos(phi*pi/180), -stitch*sin(phi*pi/180), 0.0);

     cout<<setprecision(5);
     cout<<endl<<"Mass of Star planet orbits:     "<<plan[1].GetMass()<<" Mo.";
     cout<<endl<<"Orbit Radius of the same star:  "<<plan[1].GetPos(X)<<" AU.";
     cout<<endl<<"Star's velocity: "<<plan[1].GetVel(Y)<<" AU/year."<<endl;
     cout<<endl<<"Planet's Orbit radius:  "<<plan[3].GetPos(X)<<" AU.";
     cout<<endl<<"Planet's Relative velocity: "<<plan[3].GetV()<<" AU/year."<<endl<<endl;
      if(param[0] == 1)
      {
       plan[3].SetPos(X,plan[1].GetPos(X) + plan[3].GetPos(X));
          plan[3].SetVel(Y,plan[1].GetVel(Y) + plan[3].GetVel(Y));
      }
      else if(param[0] == 2)
      {
       plan[3].SetPos(X,plan[2].GetPos(X) + plan[3].GetPos(X));
          plan[3].SetVel(Y,plan[2].GetVel(Y) + plan[3].GetVel(Y));
      }

     for(int p = 1; p <=3; p++)    //Resets the tangent space matrix to identity
         plan[p].ResetTan();

mkdirtree(param, DSP, TimePeriod, tau, pi/180*(phi-90));

     LagrangePoints(plan[1].GetMass(),plan[2].GetMass(),a);
     double endtime;
     endtime = RunSim(plan, nBodies, param, tau, TimePeriod, DSP);
```

```
}//for lower limit to upper limit in specified increments.

}//for mu lower limit to mu upper limit in specified increments.
}//Type 1 and Type 2
////////////////////////////////// Type 3 ///////////////////////////////////////
else if(param[0] == 3) {
double a = param[5];

param[7] = 0;   //no flags

     //distance from center of mass to initial point for mass 3:
     param[4] = sqrt(pow(a*(mu-0.5),2)+pow(a*(sqrt(3)/2),2));
     double TotMass = plan[1].GetMass() + plan[2].GetMass();
     double omega = sqrt(G*TotMass/(a*a*a));
     mkdirtree(param, DSP, TimePeriod, tau,1.047197551196597746154214461O932);

     double Lpoints[5+1][2];
     LagrangePoints(plan[1].GetMass(),plan[2].GetMass(),a,Lpoints);

     plan[1].SetPos(mu*a, 0.0, 0.0);
     plan[1].SetVel(0.0, mu*a*omega, 0.0);

     plan[2].SetPos((mu-1)*a, 0.0, 0.0);
     plan[2].SetVel(0.0, (mu-1)*a*omega, 0.0);

     plan[3].SetPos(Lpoints[4][0], Lpoints[4][1], 0.0);
     plan[3].SetVel(-Lpoints[4][1]*omega, Lpoints[4][0]*omega, 0.0);

    cout<<setprecision(5);
    cout<<endl<<"Primary Star mass:     "<<plan[1].GetMass()<<" Mo.";
    cout<<endl<<"Primary Star Orbit radius:  "<<plan[1].GetPos(X)<<" AU.";
    cout<<endl<<"Star velocity: "<<plan[1].GetVel(Y)<<" AU/year."<<endl;
    cout<<endl<<"Planet Orbit radius:  "<<param[4]<<" AU.";
    cout<<endl<<"Planet X velocity: "<<plan[3].GetVel(X)<<" AU/year.";
    cout<<endl<<"Planet Y velocity: "<<plan[3].GetVel(Y)<<" AU/year.";

     double endtime;
     endtime = RunSim(plan, nBodies, param, tau, TimePeriod, DSP);
}//Type 3
```

```
////////////////////////////////Type 4/////////////////////////////////////
else if(param[0] == 4) {
 param[7] = 1;  //use flags
 int t_p,body_num = 0;
//get the initial conditions from "data.in"
initial_con (TimePeriod, body_num, tau, plan);

cout <<
"-------------------------------EXOPLANET-------------------------------"
<< endl;
cout << "\nEnter lower (HZ) boundary [AU]: " ;

cin >> param[8];
cout << "\nEnter upper (HZ) boundary [AU]: " ;

cin>> param[9]; cout << endl;

cout << "Enter the order number of the terrestrial planet as defined
in [data.in]." << endl;

cout << "The star is [0], first planet [1], ...etc. "; cin >> t_p ;
cout<<"\n\n";

char Tchoice;

while((Tchoice != 'Y' && Tchoice != 'y')) {
 while(DSP < tau || (Tchoice != 'Y' && Tchoice != 'y'))
 {
 cout<<"\nEnter DSP (data sample period) in years. ";
 cin>>DSP;
 if(DSP < tau)
 cout<<"\nThe DSP (data sample period) must be >= "<<tau<<endl;
 cout<<"\nThis will result in "<<TimePeriod/DSP<<" lines of data.";
 cout<<"\nIs this acceptable? ";
 cin>>Tchoice;
 }
}

param[1]=plan[1].GetMass(); //Star
```

```
param[2]=plan[3-t_p+1].GetMass(); //Giant Planet
param[3]=plan[t_p+1].GetMass(); //Terrestrial Planet
double TotMass = param[1] + param[2];
mu = param[2]/TotMass;
param[4] = sqrt(pow(plan[t_p+1].GetPos(X),2) +
                pow(plan[t_p+1].GetPos(X),2));
param[5] = plan[1].GetPos(X) + plan[3-t_p+1].GetPos(X);
double angle = atan2(plan[t_p].GetPos(Y),plan[3].GetPos(X));
mkdirtree(param, DSP, TimePeriod, tau, angle);

cout<<"Time Period: "<<TimePeriod<<endl;

cout << "COMPUTING......" << endl;

double endtime;

endtime = RunSim(plan, nBodies, param, tau, TimePeriod, DSP);
//should take directory redirects out of runsim and put them here.
//Print to screen.
cout << "\n\nElapsed time: " << endtime << "yrs" << endl;

/* return elapsed orbital time to screen */

cout << "The output data file is [results.txt]" << endl;

cout << "------COMPLETE------.\n"; }//Type 4
////////////////////////////////////Type 5/////////////////////////////////////
else if(param[0] == 5) {
 int body_num = 0;
 param[7] = 1;  //use flags


cout <<
"-------------------------------EXOPLANET-------------------------------"
<< endl;

cout << "\nEnter lower (HZ) boundary [AU]: " ;

cin >> param[8];
```

```
cout << "\nEnter upper (HZ) boundary [AU]: " ;

cin >> param[9] ; cout << endl;

double distance[5];

distance[0] = param[8] + 0.1*(param[9]-param[8]);
distance[1] = param[8] + 0.2*(param[9]-param[8]);
distance[2] = param[8] + 0.5*(param[9]-param[8]);
distance[3] = param[8] + 0.8*(param[9]-param[8]);
distance[4] = param[8] + 0.9*(param[9]-param[8]);

for(int i = 0; i < 5; i++)

cout<<"\nDistance #"<<i<<" of planet from Star is: "<<distance[i];

const int numang = 8;
double angle[numang];
angle[0] = 0;

cout<<"\nStarting radian angles:\n"<<angle[0];

for(int i = 1; i < numang; i++) {
angle[i] = (angle[i-1] + 2*pi/numang);

cout<<", "<<angle[i]; } cout<<endl;

char Tchoice;

while((Tchoice != 'Y' && Tchoice != 'y')) {
           while(DSP < tau)
           {
           cout<<"\nEnter DSP (data sample period) in years. ";
           cin>>DSP;
           if(DSP < tau)
           cout<<"\nThe DSP (data sample period) must be >= "<<tau<<endl;
            }
 cout<<"\nThis will result in "<<TimePeriod/DSP<<" lines of data.";
```

```
 cout<<"\nIs this acceptable? ";
 cin>>Tchoice;
}

for(int i = 0; i < 5; i++)
for(int j = 0; j < numang; j++) {
        initial_con (TimePeriod, body_num, tau, plan);
        param[1]=plan[1].GetMass(); //Star
        param[2]=plan[2].GetMass(); //Giant Planet
        param[3]=plan[3].GetMass(); //Terrestrial Planet
        double TotMass = param[1] + param[2];
        mu = param[2]/TotMass;
        plan[3].SetPos(X,distance[i]*cos(angle[j]));
        plan[3].SetPos(Y,distance[i]*sin(angle[j]));
        plan[3].SetPos(Z,0);
        double factor = sqrt(G*(param[1]+param[3])/distance[i]);   //initially circular
        plan[3].SetVel(X,-factor*cos(pi/2-angle[j]));
        plan[3].SetVel(Y, factor*sin(pi/2-angle[j]));
        plan[3].SetVel(Z,0);
param[4] = sqrt(pow(plan[1].GetPos(X) - plan[3].GetPos(X),2) +
                pow(plan[1].GetPos(Y) - plan[3].GetPos(Y),2) +
                pow(plan[1].GetPos(Z) - plan[3].GetPos(Z),2));

param[5] = sqrt(pow(plan[1].GetPos(X) - plan[2].GetPos(X),2) +
                pow(plan[1].GetPos(Y) - plan[2].GetPos(Y),2) +
                pow(plan[1].GetPos(Z) - plan[2].GetPos(Z),2));

mkdirtree(param, DSP, TimePeriod, tau, angle[j]);

cout<<"COMPUTING......"<<endl;

double endtime;

endtime = RunSim(plan, nBodies, param, tau, TimePeriod, DSP);
}

cout << "------COMPLETE------.\n"; }//Type 5

    cout<<"The Program Completed Successfully without incident.\n";
```

```
    cout<<"Press any key to quit.\n";
    cin.ignore(1);
    return EXIT_SUCCESS;
}
////////////////////////////////////End of Main//////////////////////////////
```

A.3 Directory Tree Creator [mkdirtree.cpp]

The function mkdirtree creates a directory tree based on the input parameters.

```
#include <cstdio>
#include <dirent.h>    //functions for opening, reading, rewinding
                       //and closing a directory
#include <dir.h>  //creating, deleting, and changing a directory

int mkdirtree(double *parameter, double dsp, double Period,
              double timestep, double angle)
{
char buffer[50];
double type = parameter[0];
double rhonot = parameter[4];
double dist = parameter[5];
double totalmass = parameter[1]+parameter[2];
double massratio = parameter[2]/totalmass;

sprintf(buffer,"Type %1.0f [%.0f@%1.4f] DSP{%1.3f}",type,Period,timestep,dsp);
mkdir(buffer); chdir(buffer);  //going in one dir

sprintf(buffer,"[mu, M] = [%0.4f, %1.3f]",massratio,totalmass);
mkdir(buffer); chdir(buffer);  //going in one more dir

sprintf(buffer,"[rhonot, D] = [%0.5f, %1.5f] @ %1.5f",rhonot/dist,dist,angle);
mkdir(buffer); chdir(buffer);  //going in one more dir

mkdir("Data");
 return 1;
 }
```

A.4 Celestial Body Class [Cbody.h]

This file defines the class CBody which neatly packages all the information about a celestial body such as a star or planet into one data type.

```
//Class for celestial bodies
#ifndef CBODY_H
#define CBODY_H
//This isn't entirely necessary, but it makes the code
//immensly more readable
enum State{X, Y, Z};       //X = 0, Y = 1, Z = 2
class CBody {
    private:
        double mass;
        double pos[3];
        double Tpos[3];           //temp position vector
        double vel[3];
        double Tvel[3];           //temp velocity vector
        double TanSpace[6][6];    //tangent vector space for Wolf method
        double TempTans[6][6];
        double acc[3];
        double magR;
        double magV;
        double magA;
        double Radius;
        double semimajor;
        double ecc;

    public:
        CBody(void);
        ~CBody(void);
        void   ResetTan(void);
        double magdem[6];       //for lyapunov exponents
        double lyapunov[6]; //lyapunov exponents
        void   SetMass(double);
        double GetMass(void);
        int    SetPos(double, double, double);
        int    SetPos(int, double);
        double GetPos(int);
        void   SetTPos(double, double, double);
        int    SetTPos(int, double);
```

```
        double GetTPos(int);
        double GetR(void);
        void   SetVel(double, double, double);
        int    SetVel(int, double);
        double GetVel(int);
        void   SetTVel(double, double, double);
        int    SetTVel(int, double);
        double GetTVel(int);
        void   SetTanPos(int, double, double, double);
        void   SetTanVel(int, double, double, double);
        void   SetTanPos(int, int, double);
        void   SetTanVel(int, int, double);
        double GetTanPos(int, int);
        double GetTanVel(int, int);
        void   SetTempTanPos(int, double, double, double);
        void   SetTempTanVel(int, double, double, double);
        void   SetTempTanPos(int, int, double);
        void   SetTempTanVel(int, int, double);
        double GetTempTanPos(int, int);
        double GetTempTanVel(int, int);
        double GetV(void);
        double TanDot(int,int);
        double MagTan(int);
        void   GSO(void);
        void   SetAcc(double, double, double);
        int    SetAcc(int, double);
        double GetAcc(int);
        double GetA(void);
        void   SetSemimajor(double);
        double GetSemimajor(void);
        void   SetEcc(double);
        double GetEcc(void);
        void   Prepare(void);
        void   TanPrepare(void);
        void   Update(void);
        void   TanUpdate(void);
};
#endif
```

A.5 Implementation of CBody class [CBody.cpp]

This file contains the definitions of the member function of class CBody.

```
#include "CBody.h"
#include <cstdlib>
#include <math.h>

CBody::CBody(void) {
    mass = 0;
    magR = 0;
    magV = 0;
    magA = 0;
    Radius = 0;

    for(int i = 0; i < 3; i++)
    {
            pos[i] = 0;
            vel[i] = 0;
            acc[i] = 0;

            for(int k = 0; k < 6; k++) //6 components
            {
             if(i == k)           //space: X,Y,Z
             TanSpace[i][k] = 1;
             else
             TanSpace[i][k] = 0;
             if(i + 3 == k)       //velocity: u,v,w
             TanSpace[i+3][k] = 1;
             else
             TanSpace[i+3][k] = 0;
            }
    }
    for(int i = 0; i < 6; i++)
    lyapunov[i] = 0;

}

CBody::~CBody(void) {

}
```

```
void CBody::ResetTan(void) { for(int i = 0; i < 3; i++)
    {
            for(int k = 0; k < 6; k++) //6 components
            {
             if(i == k)           //space: X,Y,Z
             TanSpace[i][k] = 1;
             else
             TanSpace[i][k] = 0;
             if(i + 3 == k)       //velocity: u,v,w
             TanSpace[i+3][k] = 1;
             else
             TanSpace[i+3][k] = 0;
            }
    }
    for(int i = 0; i < 6; i++)
    lyapunov[i] = 0;
}

void CBody::SetMass(double M) {
     mass = M;
}

double CBody::GetMass(void) {
       return mass;
}

int CBody::SetPos(double x, double y, double z) {
     pos[0] = x;
     pos[1] = y;
     pos[2] = z;

     return 1;

}

int CBody::SetPos(int s, double val) {
    if(s >= 0 && s < 3)
    {
```

```
     pos[s] = val;
     return 1;
    }
   return 0;
}

double CBody::GetPos(int s) {
       if(s >= 0 && s < 3)
           return pos[s];
       else
       {
           exit(1);
       }
}

void CBody::SetTPos(double tx, double ty, double tz) {
     Tpos[0] = tx;
     Tpos[1] = ty;
     Tpos[2] = tz;

}

int CBody::SetTPos(int s, double val) {
    if(s >= 0 && s < 3)
    {
     Tpos[s] = val;
     return 1;
    }
   return 0;
}

double CBody::GetTPos(int s) {
       if(s >= 0 && s < 3)
           return Tpos[s];
       else
       {
           exit(1);
       }
}
```

```
double CBody::GetR(void) {
    magR = sqrt(pos[0]*pos[0] + pos[1]*pos[1] + pos[2]*pos[2]);
    return magR;
}

void CBody::SetVel(double vx, double vy, double vz) {
     vel[0] = vx;
     vel[1] = vy;
     vel[2] = vz;
}

int CBody::SetVel(int s, double val) {
    if(s >= 0 && s < 3)
    {
     vel[s] = val;
     return 1;
    }
   return 0;
}

double CBody::GetVel(int s) {
       if(s >= 0 && s < 3)
           return vel[s];
       else
       {
           exit(1);
       }
}

void CBody::SetTVel(double tvx, double tvy, double tvz) {
     Tvel[0] = tvx;
     Tvel[1] = tvy;
     Tvel[2] = tvz;
}

int CBody::SetTVel(int s, double val) {
    if(s >= 0 && s < 3)
    {
```

```
     Tvel[s] = val;
     return 1;
    }
   return 0;
}

double CBody::GetTVel(int s) {
       if(s >= 0 && s < 3)
           return Tvel[s];
       else
       {
           exit(1);
       }
}

void   CBody::SetTanPos(int k, double x, double y, double z) {
       if(k >= 0 && k <6 )
       {
       TanSpace[0][k] = x;
       TanSpace[1][k] = y;
       TanSpace[2][k] = z;
       }
       else
       {
           exit(1);
       }
}

void   CBody::SetTanVel(int k, double u, double v, double w) {
       if(k >= 0 && k <6 )
       {
       TanSpace[3][k] = u;
       TanSpace[4][k] = v;
       TanSpace[5][k] = w;
       }
       else
       {
           exit(1);
       }
```

```
}

void   CBody::SetTanPos(int k, int s, double val) {
       if((k >=0 && k <6) && (s >=0 && s < 3))
           TanSpace[s][k] = val;
       else
           exit(1);
}

void   CBody::SetTanVel(int k, int s, double val) {
       if((k >=0 && k <6) && (s >=0 && s < 3))
           TanSpace[s+3][k] = val;
       else
           exit(1);
}

double CBody::GetTanPos(int k, int s) {
       if((k >= 0 && k <6) && (s >= 0 && s < 3))
       return TanSpace[s][k];
       else
       exit(1);
}

double CBody::GetTanVel(int k, int s) {
       if((k >= 0 && k <6) && (s >= 0 && s < 3))
       return TanSpace[s+3][k];
       else
       exit(1);
}

void   CBody::SetTempTanPos(int k, double x, double y, double z) {
       if(k >= 0 && k <6)
       {
       TempTans[0][k] = x;
       TempTans[1][k] = y;
       TempTans[2][k] = z;
       }
       else
       {
```

```
            exit(1);
        }
}

void    CBody::SetTempTanVel(int k, double u, double v, double w) {
        if(k >= 0 && k <6)
        {
        TempTans[3][k] = u;
        TempTans[4][k] = v;
        TempTans[5][k] = w;
        }
        else
        {
            exit(1);
        }
}

void    CBody::SetTempTanPos(int k, int s, double val) {
        if((k >=0 && k <6) && (s >=0 && s < 3))
            TempTans[s][k] = val;
        else
            exit(1);
}

void    CBody::SetTempTanVel(int k, int s, double val) {
        if((k >=0 && k <6) && (s >=0 && s < 3))
            TempTans[s+3][k] = val;
        else
            exit(1);
}

double CBody::GetTempTanPos(int k, int s) {
        if((k >= 0 && k <6) && (s >= 0 && s < 3))
        return TempTans[s][k];
        else
        exit(1);
}

double CBody::GetTempTanVel(int k, int s) {
```

```
        if((k >= 0 && k <6) && (s >= 0 && s < 3))
        return TempTans[s+3][k];
        else
        exit(1);
}

double CBody::GetV(void) {
    magV = sqrt(vel[0]*vel[0] + vel[1]*vel[1] + vel[2]*vel[2]);
    return magV;
}

double CBody::TanDot(int k1, int k2) {
        double inrprod = 0;
        for(int i = 0; i < 6; i++)
        inrprod += TanSpace[i][k1]*TanSpace[i][k2];

        return inrprod;
}

double CBody::MagTan(int k) {
        double mag = 0;
        if(k >=0 && k < 6)
        {
             mag = sqrt(TanDot(k,k));
             return mag;
        }
        else
        exit(1);
}

void CBody::GSO(void) {
    for(int k = 0; k < 6; k++)
    {
        for(int i = 0; i < k; i++)
        {
             double projnum = 0;
             double projdem = 0;
             for(int s = 0; s < 6; s++)
             {
```

```
            projnum += TanSpace[s][i]*TanSpace[s][k];
            projdem += TanSpace[s][i]*TanSpace[s][i];
            }

            for(int s = 0; s < 6; s++)
            {
            TanSpace[s][k] -= projnum*TanSpace[s][i]/projdem;
            }
        }
       double projdem = 0;
       for(int s = 0; s < 6; s++)
       projdem += pow(TanSpace[s][k],2);
       magdem[k] = sqrt(projdem);
        for(int s = 0; s < 6; s++)
        TanSpace[s][k] /= magdem[k];
    }
}

void CBody::SetAcc(double ax, double ay, double az) {
     acc[0] = ax;
     acc[1] = ay;
     acc[2] = az;
}

int CBody::SetAcc(int s, double val) {
    if(s >= 0 && s < 3)
    {
     acc[s] = val;
     return 1;
    }
   return 0;
}

double CBody::GetAcc(int s) {
       if(s >= 0 && s < 3)
           return acc[s];
       else
       {
           exit(1);
```

```
        }
}

double CBody::GetA(void) {
    magA = sqrt(acc[0]*acc[0] + acc[1]*acc[1] + acc[2]*acc[2]);
    return magA;
}

void CBody::SetSemimajor(double sm) {
    semimajor = sm;

}

double CBody::GetSemimajor(void) {
        return semimajor;
}

void CBody::SetEcc(double e) {
    ecc = e;

}

double CBody::GetEcc(void) {
        return ecc;
}

void CBody::Prepare(void) {
    for(int i=0; i < 3; i++)
    {
        Tpos[i] = pos[i];
        Tvel[i] = vel[i];
    }
}

void CBody::TanPrepare(void) {
    for(int i = 0; i < 6; i++)
    for(int j = 0; j < 6; j ++)
    TempTans[i][j] = TanSpace[i][j];
}
```

```
void CBody::Update(void) {
    for(int i=0; i < 3; i++)
    {
        pos[i] = Tpos[i];
        vel[i] = Tvel[i];
    }
}

void CBody::TanUpdate(void) {
    for(int i = 0; i < 6; i++)
    for(int j = 0; j < 6; j ++)
    TanSpace[i][j] = TempTans[i][j];
}
```

A.6 Coordinates and Jacobi Constant at the Five Lagrange Points

The following is a stand alone program that calculates the coordinates of the five Lagrange points and the value of the Jacobi constant C at these points.

```
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <iomanip>
#include <math.h>

using namespace std;

void LagrangePoints( double mass1, double mass2, double d,
                     double Return[][2])
{
double L[5+1][2];
double eps=0;            // epsilon term used in derivation of Lagrange points
double oldep =0;         //storage variable for epsilon
double third = 1.0/3;    //just to make things easier
double Lprec = 1e-8;     //precision for Lagrange Points
double Laxplot, Layplot;

ofstream LaxplotOut("Data/Laxplot.txt"),
```

```
        LayplotOut("Data/Layplot.txt");

   if(!LaxplotOut || !LayplotOut)
   {
    cerr<<"Lagrange point files could not be opened!"<<endl
    <<"Make sure this directory contains a folder named 'Data'."<<endl;
    cout<<"Press any key to continue.\n";
    cin.ignore(1);
    exit(1);
   }

//  cout<<"Enter desired precision for Lagrange points, ( 1e-16 < prec < 1e-3 ): ";
//  cin>>Lprec;

//  Calculation of Lagrange Points
//  See Szebehely "Theory of Orbits"  Sections 4.3 - 4.4
//  for algorithm details and derivation
//  Note that the locations of L1 and L2 are switched

double  u = mass2/(mass1 + mass2);      //mass ratio

    L[0][0] = mass1;            //convienent way to use array to
    L[0][1] = mass2;            //store up the masses :D

    //Begin looping for L1...
    eps = pow((u/(3*(1-u))),third);

    do
    {
    oldep = eps;
    eps = pow(u*pow(1-eps,2)/(3-2*u-eps*(3-u-eps)),third);
    }while(fabs((oldep-eps)/eps)>Lprec);

    cout<<endl<<"L1: ["<<d*(u-1+eps)<<","<<0<<"]";
    L[1][0] = d*(u-1+eps);
    L[1][1] = 0;

    //Begin looping for L2...
    eps = pow((u/(3*(1-u))),third);
```

```
    do
    {
    oldep = eps;
    eps = pow(u*pow(1+eps,2)/(3-2*u+eps*(3-u+eps)),third);
    }while(fabs((oldep-eps)/eps)>Lprec);

    cout<<setprecision(6);
    cout<<endl<<"L2: ["<<d*(u-1-eps)<<","<<0<<"]";
    L[2][0] = d*(u-1-eps);
    L[2][1] = 0;

    //Begin looping for L3...
    eps = 1 - 7.0/12*u;

    do
    {
    oldep = eps;
    eps = pow((1-u)*pow(1+eps,2)/(1+2*u+eps*(2+u+eps)),third);
    }while(fabs((oldep-eps)/eps)>Lprec);

    cout<<endl<<"L3: ["<<d*(u+eps)<<","<<0<<"]";
    L[3][0] = d*(u+eps);
    L[3][1] = 0;

    //Simple calculation for L4
    cout<<endl<<"L4: ["<<d*(u-0.5)<<","<<d*(sqrt(3)/2)<<"]";
    L[4][0] = d*(u-0.5);
    L[4][1] = d*(sqrt(3)/2);

    //Simple calculation for L5
    cout<<endl<<"L5: ["<<d*(u-0.5)<<","<<-d*(sqrt(3)/2)<<"]";
    L[5][0] = d*(u-0.5);
    L[5][1] =-d*(sqrt(3)/2);

    cout<<endl<<endl;

for(int p=0;p<=5;p++) {
///////////////Gets the current state variables///////////////
```

```
////////////////////////////////////////////////////////////
Laxplot = L[p][0];                                        //
Layplot = L[p][1];                                        //
Return[p][0] = L[p][0];                                   //
Return[p][1] = L[p][1];                                   //
////////////////////////////////////////////////////////////

///////////////Then stores them in data files///////////////
////////////////////////////////////////////////////////////
    LaxplotOut<<Laxplot<<endl;                            //
    LayplotOut<<Layplot<<endl;                            //
////////////////////////////////////////////////////////////
}

LaxplotOut.close(); LayplotOut.close();
}

void LagrangePoints(double mass1, double mass2, double d) {
double L[5+1][2];

LagrangePoints(mass1,mass2,d, L);

}
```

A.7 Execution of the Simulation [RunSim.cpp]

```
#include <iostream>
#include "CBody.h"
#include "energy.cpp"
#include "wolfsymp6.cpp"   //includes lyapunov exponent calculation
#include "wolfgrav.cpp"
#include "ShiftToCM.cpp"
#include "initial_con.cpp"

using namespace std;

double RunSim(CBody* plan, const int nBodies, double *param,
              double tau,double TPeriod, double dsp)
{ double r1xplot, r1yplot,
```

```
        r2xplot, r2yplot,
        r3xplot, r3yplot,
        tplot, tauplot,
        v1xplot, v1yplot,
        v2xplot, v2yplot,
        v3xplot, v3yplot;

int printtoscreen = 25;    //Time between print to screen operations
double GSOPeriod = 1;          //Time between GSO operations


const double adaptErr = 1e-12;   //for rka

//Output files Data.txt BinaryStarData.txt and ExtraData.txt
ofstream Data1Out("Data/Data1.txt"),
         Data2Out("Data/Data2.txt"),
         Data3Out("Data/Data3.txt"),
         ExtraOut("Data/ExtraData.txt"),
         ParameterOut("Data/Parameter.txt"),
         LyapunovOut("Data/Lyapunov.txt");

   if(!Data1Out || !Data2Out || !Data3Out || !ExtraOut || !ParameterOut || !LyapunovOut)
   {
      cerr<<"One or more combination files could not be opened for writing!/n"
          <<"Press any key to quit.";
      cin.ignore(2);
      exit(1);
   }
const int max = 30; //hope strftime() will never produce more than 30 characters

char buf[max]; time_t t = time(0);
strftime(buf,max,"%d-%b-%Y %H:%M:%S", localtime(&t));


  Data1Out<<"The data in this file is output from the program.\n";
  Data1Out<<"Date and time file created: "<<buf<<"\nMass:"<<plan[1].GetMass()<<"\n";
  Data1Out<<"Time                      R1x                      R1y                      ";
  Data1Out<<"V1x                       V1y\n";
```

```
Data2Out<<"The data in this file is output from the program.\n";
Data2Out<<"Date and time file created: "<<buf<<"\nMass:"<<plan[2].GetMass()<<"\n";
Data2Out<<"Time                      R2x                      R2y                      ";
Data2Out<<"V2x                       V2y\n";

Data3Out<<"The in this file is output from the program.\n";
Data3Out<<"Date and time file created: "<<buf<<"\nMass:"<<plan[3].GetMass()<<"\n";
Data3Out<<"Time                      R3x                      R3y                      ";
Data3Out<<"V3x                       V3y\n";

ExtraOut<<"The Energy and Time Step data in this file is output from the program.\n";
ExtraOut<<"Date and time file created: "<<buf<<"\n\n";
ExtraOut<<"Time                      Potential                Kinetic                  ";
ExtraOut<<"Total                     Tau\n";

LyapunovOut<<"The lyapunov exponent data in this file is output from the program.\n";
LyapunovOut<<"Date and time file created: "<<buf<<"\n\n";
LyapunovOut<<"Time                      ";
LyapunovOut<<"P1L1                      P1L2                        P1L3                        ";
LyapunovOut<<"P1L4                      P1L5                        P1L6                        ";
LyapunovOut<<"P2L1                      P2L2                        P2L3                        ";
LyapunovOut<<"P2L4                      P2L5                        P2L6                        ";
LyapunovOut<<"P3L1                      P3L2                        P3L3                        ";
LyapunovOut<<"P3L4                      P3L5                        P3L6\n";

////////////////////////////////////////////////////////////////////////////////////////////
   ParameterOut<<"Type: \t"<<param[0]<<endl;                    //Type of initial conditions
   ParameterOut<<"\nFirst mass: \t"<<param[1]<<endl;            //First Mass
   ParameterOut<<"\nSecond mass: \t"<<param[2]<<endl;           //Second Mass
   ParameterOut<<"\nThird mass: \t"<<param[3]<<endl;            //small mass
   ParameterOut<<"\nR0: \t"<<param[4]<<endl;                    //R0
   ParameterOut<<"\nD: \t"<<param[5]<<endl;                     //D
   ParameterOut<<"\nBinary Eccentricity: \t"<<param[6]<<endl;   //binary eccentricity
   ParameterOut<<"\nFlags? \t"<<param[7]<<endl;                 //Flags? 1: using flags, 0: not
   ParameterOut<<"\nHZ lower limit: \t"<<param[8]<<endl;        //HZ lower limit
   ParameterOut<<"\nHZ upper limit: \t"<<param[9]<<endl;        //HZ upper limit

   //shift coordinates to Center of Mass coordinates for simulation
   ShiftToCM(plan);
```

```
cout<<"Starting simulation,...."<<endl<<endl; double time = 0;
double tdLast = 0;    //time to take data

double tpsLast = 0;   //time to print data to screen

double tgsoLast = 0;  //time to do GSO
//*************************************************************************
while(time <= TPeriod) {
double kinetic = 0;
double potential = 0;
double total = 0;
double Sep13 = 0;
                                          //Defined in energy.cpp
    kinetic = Kinetic(plan,3);            //kinetic of body 3
    potential = Potential(nBodies,plan,3);  //potential of body 3
    total = Total(nBodies,plan);          //total energy of system

if(kinetic>2*fabs(potential)) {

cout<<"\nKinetic Energy is greater than Potential energy by a factor
of 2,"
    <<"\n...no longer bound.\n";
cout<<"Possible run-time error or collision....halting
simulation.\n";
 Data1Out.close();
 Data2Out.close();
 Data3Out.close();
 ExtraOut.close();
ParameterOut<<"\nSimulation ended due to Energy criterion at:
"<<time<<" years."; ParameterOut.close(); LyapunovOut.close();
 //".." refers to parent dir
 chdir("..");        //go back to mu dir
 chdir("..");        //go back to Type dir
 chdir("..");        //go back to main program dir
return time; }

for(int i = 0; i < 3; i++)
Sep13 += pow(plan[1].GetPos(i)-plan[3].GetPos(i),2);
```

```
Sep13 = sqrt(Sep13);

if(param[7] == 1 && Sep13 < param[8]) {
 cout<<"\nPlanet crossed lower HZ limit at "<<time<<" years!\n";
 Data1Out.close();
 Data2Out.close();
 Data3Out.close();
 ExtraOut.close();
 ParameterOut<<"\nSimulation ended due to crossing lower HZ limit at: "<<time<<" years.";
 ParameterOut.close();
 LyapunovOut.close();
 //".." refers to parent dir
 chdir("..");        //go back to mu dir
 chdir("..");        //go back to Type dir
 chdir("..");        //go back to main program dir
 return time;
}

if(param[7] == 1 && Sep13 > param[9]) {
 cout<<"\nPlanet crossed upper HZ limit at "<<time<<" years!\n";
 Data1Out.close();
 Data2Out.close();
 Data3Out.close();
 ExtraOut.close();
 ParameterOut<<"\nSimulation ended due to crossing upper HZ limit at: "<<time<<" years.";
 ParameterOut.close();
 LyapunovOut.close();
 //".." refers to parent dir
 chdir("..");        //go back to mu dir
 chdir("..");        //go back to Type dir
 chdir("..");        //go back to main program dir
 return time;
}

//Print to screen only once every printtoscreen years
if((time - tpsLast) > printtoscreen) {
          double p1k = Kinetic(plan,1);
          double p2k = Kinetic(plan,2);
```

```
            double p3k = kinetic;
            double p1p = Potential(nBodies,plan,1);
            double p2p = Potential(nBodies,plan,2);
            double p3p = potential;
    cout<<setprecision(5)<<endl;

    cout<<"[mu,rhonot] = ["<<param[2]/(param[1]+param[2])
        <<", "<<param[4]/param[5]<<"] after "<<time<<" years: \n";
    cout<<"     KIN: "<<p1k<<" "<<p2k<<" ";
    cout<<setprecision(3)<<p3k<<setprecision(5)<<endl;
    cout<<"     POT: "<<p1p<<" "<<p2p<<" ";
    cout<<setprecision(3)<<p3p<<setprecision(17)<<endl;
    cout<<"     TOT: "<<total;
    cout<<"\nComplete: "<<setprecision(5)<<time/TPeriod<<"\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n";
    tpsLast = time;


}

/* if(((time - tgsoLast) > (GSOPeriod - 0.5*tau)) && ((time -
tgsoLast) < (GSOPeriod + 0.5*tau)) || time == TPeriod) {

for(int p = 1; p <=3; p++) {
    plan[p].GSO();

    for(int k = 0; k < 6; k++)
    {
    plan[p].lyapunov[k] += log(plan[p].magdem[k]);   //running sum
    }
}//p
    tgsoLast = time;
}//if test for gso period
*/

//Record position, velocity and energy every DSP years for plotting.

if(((time - tdLast) > (dsp - 0.5*tau)) && ((time - tdLast) < (dsp +
0.5*tau)) || time == 0 || time == TPeriod) {
```

```
///////////////Gets the current state variables///////////////
//////////////////////////////////////////////////////////////
r1xplot = plan[1].GetPos(X);                                  //
r1yplot = plan[1].GetPos(Y);                                  //
                                                              //
r2xplot = plan[2].GetPos(X);                                  //
r2yplot = plan[2].GetPos(Y);                                  //
                                                              //
r3xplot = plan[3].GetPos(X);                                  //
r3yplot = plan[3].GetPos(Y);                                  //
                                                              //
v1xplot = plan[1].GetVel(X);                                  //
v1yplot = plan[1].GetVel(Y);                                  //
                                                              //
v2xplot = plan[2].GetVel(X);                                  //
v2yplot = plan[2].GetVel(Y);                                  //
                                                              //
v3xplot = plan[3].GetVel(X);                                  //
v3yplot = plan[3].GetVel(Y);                                  //
                                                              //
tplot = time;                                                 //
tauplot = tau;                                                //
                                                              //
//////////////////////////////////////////////////////////////

///////////////Then stores them in data files///////////////
//////////////////////////////////////////////////////////////

    Data1Out.precision(16);
    Data1Out.setf(ios_base::left,ios_base::adjustfield);
    Data1Out.setf(ios_base::scientific,ios_base::floatfield);
    Data1Out.unsetf(ios_base::showpos);
    Data1Out.width(26);    Data1Out<<tplot;
    Data1Out.setf(ios_base::showpos);
    Data1Out.width(26);     Data1Out<<r1xplot;
    Data1Out.width(26);     Data1Out<<r1yplot;
    Data1Out.width(26);     Data1Out<<v1xplot;
    Data1Out.width(26);     Data1Out<<v1yplot<<endl;
```

```
    Data2Out.precision(16);
    Data2Out.setf(ios_base::left,ios_base::adjustfield);
    Data2Out.setf(ios_base::scientific,ios_base::floatfield);
    Data2Out.unsetf(ios_base::showpos);
    Data2Out.width(26);    Data2Out<<tplot;
    Data2Out.setf(ios_base::showpos);
    Data2Out.width(26);     Data2Out<<r2xplot;
    Data2Out.width(26);     Data2Out<<r2yplot;
    Data2Out.width(26);     Data2Out<<v2xplot;
    Data2Out.width(26);     Data2Out<<v2yplot<<endl;

    Data3Out.precision(16);
    Data3Out.setf(ios_base::left,ios_base::adjustfield);
    Data3Out.setf(ios_base::scientific,ios_base::floatfield);
    Data3Out.unsetf(ios_base::showpos);
    Data3Out.width(26);    Data3Out<<tplot;
    Data3Out.setf(ios_base::showpos);
    Data3Out.width(26);     Data3Out<<r3xplot;
    Data3Out.width(26);     Data3Out<<r3yplot;
    Data3Out.width(26);     Data3Out<<v3xplot;
    Data3Out.width(26);     Data3Out<<v3yplot<<endl;

    ExtraOut.precision(16);
    ExtraOut.setf(ios_base::left,ios_base::adjustfield);
    ExtraOut.setf(ios_base::scientific,ios_base::floatfield);
    ExtraOut.unsetf(ios_base::showpos);
    ExtraOut.width(26);    ExtraOut<<tplot;
    ExtraOut.setf(ios_base::showpos);
    ExtraOut.width(26);    ExtraOut<<potential;
    ExtraOut.width(26);    ExtraOut<<kinetic;
    ExtraOut.width(26);    ExtraOut<<total;
    ExtraOut.unsetf(ios_base::showpos);
    ExtraOut.width(26);    ExtraOut<<tau<<endl;
/*
    LyapunovOut.precision(16);
    LyapunovOut.setf(ios_base::left,ios_base::adjustfield);
    LyapunovOut.setf(ios_base::scientific,ios_base::floatfield);
    LyapunovOut.unsetf(ios_base::showpos);
    LyapunovOut.width(26);    LyapunovOut<<time;
```

```
    LyapunovOut.setf(ios_base::showpos);
    for(int p = 1; p <= 3; p++)
    {
    for(int k = 0; k < 6; k++)
    {
    if(time != 0)
    {
    LyapunovOut.width(26);    LyapunovOut<< (plan[p].lyapunov[k]/time);
    }
    else
    {
    LyapunovOut.width(26);    LyapunovOut<< time;
    }
    }//k
    }//p
    LyapunovOut<<endl;
  */  //Don't forget to remove comment in front of TanUpdate below...

    tdLast = time;
}//if test for DSP

    //When newstate is passed it is the same as state
    //wolfsymp6(time, tau, plan, wolfgrav);
    //rk4(time, tau, plan, gravrk);                         //integrate using same timestep
    symp6(time, tau, plan, gravrk);                         //integrate using same timestep
    //tau = symp6a(time, tau, plan, adaptErr, gravrk);      //integrate using varying timestep

    for(int p=1; p<=3; p++)
    {
    plan[p].Update();   //the idea here is change after all steps are made instead
    //plan[p].TanUpdate();
    }                   //of changing while steps are being made,
                        //which hopefully reduces error propagation

    //Now newstate has been integrated

    time+=tau;

}// while(time <= TPeriod)
```

```
Data1Out.close();
Data2Out.close();
Data3Out.close();
ExtraOut.close();

//".." refers to parent dir
chdir("..");        //go back to mu dir
chdir("..");        //go back to Type dir
chdir("..");        //go back to main program dir

return time;

}//RunSim
```

A.8 Energy Calculators [energy.cpp]

```
double Kinetic(CBody* param, int i);
double Potential(int n, CBody* param, int i);
double Total(int n, CBody* param);

//  n       the total number of Celestial Bodies
//  param   array of CBody data structures
//  i       index number of a specific body

double Kinetic(CBody* param, int i) { double kinetic=0;
    kinetic += 0.5*param[i].GetMass()*param[i].GetV()*param[i].GetV();
    return kinetic;
}//Kinetic

double Potential(int n, CBody* param, int i) {

const double pi = acos(-1);
double G = 4*pi*pi; double potential=0;
double sepex[6];    //seperation vectors #: n(n-1) where n = 3 bodies in this case
                    //(different directions counted)
double sepey[6];    //seperated into x, y and z components
double sepez[6]; double SEP[6];  //magnitude of separation vector
```

```
for(int j = 1; j <= n; j++) { if(i!=j)
    {
        //X, Y and Z components of separation vector for force on i due to j
        sepex[j] = param[j].GetPos(X)-param[i].GetPos(X);  //X
        sepey[j] = param[j].GetPos(Y)-param[i].GetPos(Y);  //Y
        sepez[j] = param[j].GetPos(Z)-param[i].GetPos(Z);  //Z
        SEP[j]=sqrt(sepex[j]*sepex[j]+sepey[j]*sepey[j] + sepez[j]*sepez[j]);//magnitude of separation vector
        potential+=-G*param[i].GetMass()*param[j].GetMass()/SEP[j];//adds up potential of all bodies
    }//if
}//for
    return potential;
}//Potential

double Total(int n, CBody* param) {
    double total=0;

    for(int i=1; i<=n; i++)
    total += Kinetic(param,i) + Potential(n,param,i)/2;

    return total;
}//Total
```

A.9 Symplectic Integrator (6^{th} Order) [wolfsymp6.cpp]

```
#include "Cbody.h"
#include <math.h>

void wolfsymp6(double t, double tau, CBody* planet,
  void (*derivsRK)(double t, CBody* planet, double deriv[][6], double tanderiv[][6][6]))
{
// Symplectic integrator (8th order)
// Inputs
//   t          Independent variable (usually time)
//   tau        Step size (usually time step)
//   planet     State array of CBody objects
//   derivsRK   Right hand side of the ODE; derivsRK is the
//              name of the function which returns dx/dt
//              Calling format derivsRK(t,planet,dxdt).
// Output
```

```
//   planet     New state of planet array after a step of size tau

// The algorithm is from "Construction of higher order symplectic integrators"
// by Haruo Yoshida
// Physics Letters A, Volume 150, #5,6,7, pages 262-268, 1990
// (section 4)

    const double thrdrtwo = pow(2,1/3);
    const double fifthrtwo = pow(2,1/5);
    const double x0 = -thrdrtwo/(2-thrdrtwo);
    const double x1 = 1/(2-thrdrtwo);
    const double y0 = -fifthrtwo/(2-fifthrtwo);
    const double y1 = 1/(2-fifthrtwo);

    const double d[11] = {0,x1*y1,
                          x0*y1,
                          x1*y1,
                          x1*y0,
                          x0*y0,
                          x1*y0,
                          x1*y1,
                          x0*y1,
                          x1*y1,
                          0};

    const double c[11] = {0,0.5*d[1],
                          0.5*(d[1]+d[2]),
                          0.5*(d[2]+d[3]),
                          0.5*(d[3]+d[4]),
                          0.5*(d[4]+d[5]),
                          0.5*(d[5]+d[6]),
                          0.5*(d[6]+d[7]),
                          0.5*(d[7]+d[8]),
                          0.5*(d[8]+d[9]),
                          0.5*d[9]};

    double F[3+1][6], Ft[3+1][6][6];

for(int p=1; p<=3; p++) {
```

```
    planet[p].Prepare();        //sets tempstate equal to state
    planet[p].TanPrepare();
}



for(int j=1; j<=10; j++) { for(int p = 1; p <=3; p++) {
        for(int s = 0; s <3; s++)
        {
        planet[p].SetTPos(s,planet[p].GetTPos(s) + tau*c[j]*planet[p].GetTVel(s));
        for(int k = 0; k < 6; k++)
        planet[p].SetTempTanPos(k,s,planet[p].GetTempTanPos(k,s) + tau*c[j]*planet[p].GetTempTanVel(k,s));
        }
}

        (*derivsRK)(t, planet, F, Ft);  //integrates tempstate

for(int p =1; p <=3; p++) {
        for(int s = 0; s < 3; s++)
        {
        planet[p].SetTVel(s,planet[p].GetTVel(s) + tau*d[j]*F[p][s+3]);
        for(int k = 0; k < 6; k++)
        planet[p].SetTempTanVel(k,s,planet[p].GetTempTanVel(k,s) + tau*d[j]*Ft[p][k][s+3]);
        }
}

}// for over j

}
```

A.10 Gravitational Forces Between All Bodies [wolfgrav.cpp]

```
#include "CBody.h"
#include <math.h>

void wolfgrav(double t, CBody* planet, double deriv[][6], double
tanderiv[][6][6]) {
//  Returns right-hand side of linearized three body ODE;
//  Inputs
//    t           Time (not used)
```

```
//    planet      State array of CBody objects
//  Output
//    deriv       Derivatives [dr(1)/dt dr(2)/dt dv(1)/dt dv(2)/dt]
const double pi = acos(-1);
double sepx[3+1][3+1];//seperation vectors
double sepy[3+1][3+1];//seperated into x, y and z components
double sepz[3+1][3+1];
double SEP[3+1][3+1];  //magnitude of seperation vector
double GM_R3[3+1][3+1];
enum State{ X, Y, Z};
double SEPSQRD[3+1][3+1];

for(int p =1; p <=3; p++)
for(int j =1; j <=3; j++) {
    if(j!=p)
    {
        sepx[p][j] = planet[j].GetTPos(X)-planet[p].GetTPos(X);    //X
        sepy[p][j] = planet[j].GetTPos(Y)-planet[p].GetTPos(Y);    //Y
        sepz[p][j] = planet[j].GetTPos(Z)-planet[p].GetTPos(Z);    //Z

        SEP[p][j]=sqrt(pow(sepx[p][j],2) + pow(sepy[p][j],2) + pow(sepz[p][j],2));//magnitude of separation vector
        //* Compute acceleration
        GM_R3[p][j] = 4*pi*pi*planet[j].GetMass()/pow(SEP[p][j],3);
        SEPSQRD[p][j] = pow(SEP[p][j],2);
    }
}

//////////////////////////////////////////////////////////////////////
for(int p = 1; p <= 3; p++) {
double accelx = 0;
double accely = 0;
double accelz = 0;
//$$$$$$$$$$$$$$$$$$$$$$$$ Calculate the Force due to others $$$$$$$$$$$$$$$$$$
    for(int j=1; j<= 3; j++)
    {
    if(j!=p)
    {
     //X, Y, and Z components of seperation vector for force on p due to j
     //at temporary position state
```

```
        accelx += GM_R3[p][j]*sepx[p][j];
        accely += GM_R3[p][j]*sepy[p][j];
        accelz += GM_R3[p][j]*sepz[p][j];
    }//if j not equal to p
    }//for over j
//$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$


  //* Return derivatives [dx/dt dy/dt dz/dt dvx/dt dvy/dt dvz/dt]
  deriv[p][0] = planet[p].GetTVel(X);
  deriv[p][1] = planet[p].GetTVel(Y);
  deriv[p][2] = planet[p].GetTVel(Z);
  deriv[p][3] = accelx;
  deriv[p][4] = accely;
  deriv[p][5] = accelz;

////////////////////////////////////////////////////////////////////////////////
for(int k = 0; k < 6; k++) { double dudot = 0; double dvdot = 0;
double dwdot = 0;
//$$$$$$$$$$$$ Calculate the Linearized Force due to others $$$$$$$$$$$$$$$$$$$
    for(int j=1; j<= 3; j++)
    {
    if(j!=p)
    {
        dudot += GM_R3[p][j]*((-1+3*pow(sepx[p][j],2)/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,X) +
                              (3*sepx[p][j]*sepy[p][j]/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,Y) +
                              (3*sepx[p][j]*sepz[p][j]/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,Z));
        dvdot += GM_R3[p][j]*((3*sepx[p][j]*sepy[p][j]/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,X) +
                              (-1+3*pow(sepy[p][j],2)/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,Y) +
                              (3*sepy[p][j]*sepz[p][j]/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,Z));
        dwdot += GM_R3[p][j]*((3*sepx[p][j]*sepz[p][j]/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,X) +
                              (3*sepy[p][j]*sepz[p][j]/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,Y) +
                              (-1+3*pow(sepz[p][j],2)/SEPSQRD[p][j])*planet[p].GetTempTanPos(k,Z));
    }//if j not equal to p
    }//for over j
//$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
  tanderiv[p][k][X] = planet[p].GetTempTanVel(k,X);
  tanderiv[p][k][Y] = planet[p].GetTempTanVel(k,Y);
  tanderiv[p][k][Z] = planet[p].GetTempTanVel(k,Z);
```

```
  tanderiv[p][k][X+3] = dudot;
  tanderiv[p][k][Y+3] = dvdot;
  tanderiv[p][k][Z+3] = dwdot;
}//for over k }//for over p }//gravrk
```

A.11 Shift to Center of Mass Coordinates [ShiftToCM.cpp]

```
#include "Cbody.h"

void ShiftToCM(CBody* plan) {
       double TotalMass = 0;
       double CM[3] = {0.0, 0.0, 0.0};
       double VCM[3] = {0.0, 0.0, 0.0};

       for(int i = 1; i <=3; i++)
       TotalMass += plan[i].GetMass();

       for(int p = 0; p <3; p++)
       {
       for(int i = 1; i <=3; i++)
       {
          CM[p] += plan[i].GetMass()*plan[i].GetPos(p);
         VCM[p] += plan[i].GetMass()*plan[i].GetVel(p);
       }
          CM[p] /= TotalMass;
         VCM[p] /= TotalMass;
       }

       for(int p = 0; p <3; p++)
       {
       for(int i = 1; i <=3; i++)
       {
           plan[i].SetPos(p,plan[i].GetPos(p) - CM[p]);
           plan[i].SetVel(p,plan[i].GetVel(p) - VCM[p]);
       }
       }
}
```

A.12 Read Initial Conditions From File [initialcon.cpp]

```
#include "CBody.h"
#include <fstream>
#include <iostream>

using namespace std;

void initial_con (double &t_stop, int &b_num, double &delta_t,
CBody* plan) {

const int LINESIZE = 80;
const int debug = 0;

ifstream fpIn("data.in");

char buff[LINESIZE]; // setup file reading buffers

if (!fpIn) // open "data.in" file or exit gracefully
{
cout << "Data.in could not be opened!!" << std::endl;
cout<<"Press any key to quit.";
cin.ignore(2); exit(1);
}

//This section reads the time duration, step and number
//of bodies from the dat file.
fpIn.getline( buff, LINESIZE, ':'); // read line from the file
fpIn.getline( buff, LINESIZE);
t_stop = atof (buff); // convert to numeric
if ( debug == 1) // debug output
cout << "Time stop: "<<t_stop << endl; // etc.............
fpIn.getline( buff, LINESIZE, ':'); // read line from the file
fpIn.getline( buff, LINESIZE);
delta_t = atof (buff);
if ( debug == 1)
cout << "Time step: "<<delta_t << endl;
fpIn.getline( buff, LINESIZE, ':'); // read line from the file
fpIn.getline( buff, LINESIZE);
b_num = atoi (buff);
```

```
if ( debug == 1)
cout << "Number of bodies: "<<b_num << endl<<endl; /*
* This section reads the planet initial conditions
* from the data file.
*/
for (int counter = 1;counter <= b_num; counter++) {
fpIn.getline( buff, LINESIZE);
fpIn.getline( buff, LINESIZE, ':'); // read line from the file
fpIn.getline( buff, LINESIZE);
plan[counter].SetMass(atof(buff));
cout << "Body "<<counter<<" Mass: "<<plan[counter].GetMass()
<< endl;
cout<<"Body "<<counter<<" Position vector: <";
for(int s = 0; s < 3; s++) {
fpIn.getline( buff, LINESIZE, ':'); // read line from the file
fpIn.getline( buff, LINESIZE);
plan[counter].SetPos(s,atof(buff));
cout << plan[counter].GetPos(s)<<", ";
}

cout<<">"<<endl;
cout<<"Body "<<counter<<" Velocity vector: <";
for(int s = 0; s < 3; s++) {
fpIn.getline( buff, LINESIZE, ':'); // read line from the file
fpIn.getline( buff, LINESIZE);
plan[counter].SetVel(s,atof(buff));
cout << plan[counter].GetVel(s)<<", ";
}
cout<<">"<<endl;
cout<<"End Body # "<<counter<<":\n\n";
}

fpIn.close();
if ( debug == 1) {
cout<<"Take a moment to confirm this is correct.\n"
<<"Press any key to continue: \n"; cin.ignore(2); } }
```

REFERENCES

S. J. Aarseth. *Gravitational N-Body Simulations: Tools and Algorithms*. Cambridge University Press, 1^{st} edition, 2003.

J. Barnes and P. Hut. A Hierarchical O(N log N) Force-Calculation algorithm. *Nature*, **324**:446, 1986.

I. N. Bronshtein and K. A. Semendyayev. *Handbook of Mathematics*. Springer, New York, 3^{rd} edition, 1997.

E. I. Butikov. The Velocity Hodograph for an Arbitrary Keplerian Motion. *European Journal of Physics*, **21**:297, 2000.

R. P. Butler, C. G. Tinney, G. W. Marcy, H. R. A. Jones, et al. Two New Planets from the Anglo-Australian Planet Search. *The Astrophysical Journal*, **555**:410, 2001.

R. P. Butler, J. T. Wright, G. Marcy, D. A. Fischer, et al. Catalog of Nearby Exoplanets. *The Astrophysical Journal*, **646**:505, 2006.

B. Campbell, G. A. H. Walker, and S. Yang. A Search for Substellar Companions to Solar-Type Stars. *The Astrophysical Journal*, **331**:902, 1988.

J. Catanzarite, M. Shao, A. Tanner, S. Unwin, et al. Astrometric Detection of Terrestrial Planets in the Habitable Zones of Nearby Stars with SIM PlanetQuest. *Publications of the Astronomical Society of the Pacific*, **118**:1319, 2006.

C. S. Cockell, A. Lger, M. Fridlund, T. M. Herbst, et al. Darwin-A Mission to Detect and Search for Life on Extrasolar Planets. *Astrobiology*, **9**:1, 2009.

A. Cumming, R. P. Butler, G. W. Marcy, S. S. Vogt, et al. The Keck Planet Search: Detectability and the Minimum Mass and Orbital Period Distribution

of Extrasolar Planets. *Publications of the Astronomical Society of the Pacific*, **120**:531, 2008.

M. Cuntz, J. Eberle, and Z. E. Musielak. Stringent Criteria for Stable and Unstable Planetary Orbits in Stellar Binary Systems. *Astrophysical Journal*, **669**:L105, 2007.

J. S. Drilling and A. U. Landolt. Normal Stars. In A. N. Cox, editor, *Allen's Astrophysical Quantities*, p. 381. Springer, New York, 4^{th} edition, 2000.

A. Duquennoy and M. Mayor. Multiplicity Among Solar-Type Stars in the Solar Neighbourhood. *Astronomy and Astrophysics*, **248**:485, 1991.

R. Dvorak. Numerical Experiments on Planetary Orbits in Double Stars. *Celestial Mechanics*, **34**:369, 1984.

R. Dvorak. Critical Orbits in the Elliptical Restricted Three-Body Problem. *Astronomy and Astrophysics*, **167**:379, 1986.

R. Dvorak, E. Pilat-Lohinger, R. Schwarz, and F. Freistetter. Extrasolar Trojan Planets Close to Habitable Zones. *Astronomy and Astrophysics*, **426**:L37, 2004.

R. Dvorak, R. Schwarz, Á. Süli, and T. Kotoulas. On the Stability of the Neptune Trojans. *Monthly Notices of the Royal Astronomical Society*, **382**:1324, 2007.

J. Eberle, M. Cuntz, and Z. E. Musielak. The Instability Transition for the Restricted 3-Body Problem - I. Theoretical Approach. *Astronomy and Astrophysics*, **489**:1329, 2008.

A. Eggenberger, S. Udry, and M. Mayor. Statistical Properties of Exoplanets. *Astronomy and Astrophysics*, **417**:353, 2004.

European Space Agency. The Hipparcos and Tycho Catalogues (SP-1200). 1997.

M. Fatuzzo, F. C. Adams, R. Guvin, and E. M. Proszkow. A Statistical Stability Analysis of Earth-like Planetary Orbits in Binary Systems. *Publications of the Astronomical Society of the Pacific*, **118**:1510, 2006.

F. Forget and R. T. Pierrehumbert. Warming Early Mars with Carbon Dioxide Clouds That Scatter Infrared Radiation. *Science*, **278**:1273, 1997.

M. Fouchard, E. Lega, C. Froeschlé, and C. Froeschlé. On the Relationship Between Fast Lyapunov Indicator and Periodic Orbits for Continuous Flows. *Celestial Mechanics and Dynamical Astronomy*, **83**:205, 2002.

C. Froeschlé, R. Gonczi, and E. Lega. The Fast Lyapunov Indicator: A Simple Tool to Detect Weak Chaos. Application to the Structure of the Main Asteroidal Belt. *Planetary and Space Science*, **45**:881, 1997.

A. L. Garcia. *Numerical Methods for Physics*. Prentice-Hall, Inc., Upper Saddle River, New Jersey, 2^{nd} edition, 2000.

I. Halevy, R. T. Pierrehumbert, and D. P. Schrag. Radiative Transfer in CO2-Rich Paleoatmospheres. *Journal of Geophysical Research*, **114**:D18112, 2009.

W. R. Hamilton. In *Proceedings of the Royal Irish Academy*, volume 3, pp. 344–353. Royal Irish Academy, 1847.

URL `http://www.maths.soton.ac.uk/EMIS/classics/Hamilton/Hodo.pdf`

A. P. Hatzes, W. D. Cochran, M. Endl, B. McArthur, et al. A Planetary Companion to γ Cephei A. *The Astrophysical Journal*, **599**:1383, 2003.

M. J. Holman and P. A. Wiegert. Long-Term Stability of Planets in Binary Systems. *The Astronomical Journal*, **117**:621, 1999.

N. Houk and A. P. Cowley. *University of Michigan Catalogue of Two-Dimensional Spectral Types for the HD Stars. Volume I.* University of Michigan, Ann Arbor, Michigan, 1975.

W. H. Jefferys. Stability Regions for Quasiperiodic Motion in the Restricted Problem of Three Bodies. *The Astronomical Journal*, **79**:710, 1974.

B. W. Jones. Exoplanets : Search Methods, Discoveries, and Prospects for Astrobiology. *International Journal of Astrobiology*, **7**:279, 2008.

J. F. Kasting, D. P. Whitmire, and R. T. Reynolds. Habitable Zones around Main Sequence Stars. *Icarus*, **101**:108, 1993.

W. Kley. Planet Formation in Binary Systems. In H. Zinnecker and R. Mathieu, editors, *The Formation of Binary Stars*, volume 200 of *IAU Symposium*, pp. 511–518. 2001.

C. J. Lada. Stellar Multiplicity and the Initial Mass Function: Most Stars Are Single. *Astrophysical Journal, Letters*, **640**:L63, 2006.

F. van Leeuwen. Validation of the New Hipparcos Reduction. *Astronomy and Astrophysics*, **474**:653, 2007.

M. Mayor and D. Queloz. A Jupiter-Mass Companion to a Solar-Type Star. *Nature*, **378**:355, 1995.

M. A. Mischna, J. F. Kasting, A. Pavlov, and R. Freedman. Influence of Carbon Dioxide Clouds on Early Martian Climate. *Icarus*, **145**:546, 2000.

L. R. Mudryk and Y. Wu. Resonance Overlap is Responsible for Ejecting Planets in Binary Systems. *Astrophysical Journal*, **639**:423, 2006.

Z. E. Musielak, M. Cuntz, E. A. Marshall, and T. D. Stuit. Stability of Planetary Orbits in Binary Systems. *Astronomy and Astrophysics*, **434**:355, 2005.

M. Noble, Z. E. Musielak, and M. Cuntz. Orbital Stability of Terrestrial Planets Inside the Habitable Zones of Extrasolar Planetary Systems. *The Astrophysical Journal*, **572**:1024, 2002.

J. Patience, R. J. White, A. M. Ghez, C. McCabe, et al. Stellar Companions to Stars with Planets. *Astrophysical Journal*, **581**:654, 2002.

E. Pilat-Lohinger and R. Dvorak. Stability of S-type Orbits in Binaries. *Celestial Mechanics and Dynamical Astronomy*, **82**:143, 2002.

C. A. Prieto and D. L. Lambert. Fundamental Parameters of Nearby Stars from the Comparison with Evolutionary Calculations: Masses, Radii and Effective Temperatures. *Astronomy and Astrophysics*, **352**:555, 1999.

D. Queloz, D. Anderson, A. C. Cameron, M. Gillon, et al. WASP-8b: A Retrograde Transiting Planet in a Multiple System. *Astronomy and Astrophysics*, **517**:L1, 2010.

E. V. Quintana, J. J. Lissauer, J. E. Chambers, and M. J. Duncan. Terrestrial Planet Formation in the Alpha Centauri System. *Astrophysical Journal*, **576**:982, 2002.

G. Rabl and R. Dvorak. Satellite-Type Planetary Orbits in double Stars - A Numerical Approach. *Astronomy and Astrophysics*, **191**:385, 1988.

D. Raghavan, T. J. Henry, B. D. Mason, J. P. Subasavage, et al. Two Suns in The Sky: Stellar Multiplicity in Exoplanet Systems. *Astrophysical Journal*, **646**:523, 2006.

D. J. Ramm, D. Pourbaix, J. B. Hearnshaw, and S. Komonjinda. Spectroscopic Orbits for K giants β Reticuli and ν Octantis: What is Causing a Low-Amplitude Radial Velocity Resonant Perturbation in ν Oct? *Monthly Notices of the Royal Astronomical Society*, **394**:1695, 2009.

I. Ribas, E. Solano, E. Masana, and A. Gimnez. Effective Temperatures and Radii of Planet-Hosting Stars from IR Photometry. *Astronomy and Astrophysics*, **411**, 2003.

A. E. Roy. *Orbital Motion*. Institute of Physics Publishing, London, 4^{th} edition, 2005.

K. R. Symon. *Mechanics*. Addison-Wesley, Reading, Massachusetts, 3^{rd} edition, 1971.

V. G. Szebehely. *Theory of Orbits*. Academic Press, New York and London, 1^{st} edition, 1967.

F. Szenkovits and Z. Makó. About the Hill stability of Extrasolar Planets in Stellar Binary Systems. *Celestial Mechanics and Dynamical Astronomy*, **101**:273, 2008.

C. G. Tinney, R. P. Butler, G. W. Marcy, H. R. A. Jones, et al. Two Extrasolar Planets from the Anglo-Australian Planet Search. *The Astrophysical Journal*, **571**:528, 2002.

D. E. Trilling, J. A. Stansberry, K. R. Stapelfeldt, G. H. Rieke, et al. Debris disks in main-sequence binary systems. *The Astrophysical Journal*, **658**:1289, 2007.

D. R. Underwood, B. W. Jones, and P. N. Sleep. The Evolution of Habitable Zones During Stellar Lifetimes and its Implications on the Search for Extraterrestrial Life. *International Journal of Astrobiology*, **2**:289, 2003.

A. Wolszczan and D. A. Frail. A planetary System Around the Millisecond Pulsar PSR1257 + 12. *Nature*, **355**:145, 1992.

H. Yoshida. Construction of Higher Order Symplectic Integrators. *Physics Letters A*, **150**:262, 1990.

BIOGRAPHICAL STATEMENT

William Jason Eberle was born in Reading, Pennsylvania, in 1976. He received his B.S. in Physics from the University of Texas at Dallas, in 2004, and his M.S. and Ph.D. in Physics from The University of Texas at Arlington in 2007 and 2010 respectively. He became interested in the gravitational three body problem during his junior year at UTD and has worked on it in his spare time ever since. To have been able to compose his thesis on this pet project of his was unexpectedly fortunate. Jason become interested in complexity theory, chaos, and astrobiology while at UTA.