2014

Particle Swarm Optimization of Low-Thrust, Geocentric-to-Halo-Orbit Transfers

Andrew Abraham
Lehigh University

Particle Swarm Optimization of Low-Thrust, Geocentric-to-Halo-Orbit Transfers

by

Andrew J. Abraham

Presented to the Graduate and Research Committee
of Lehigh University
in Candidacy for the Degree of
Doctor of Philosophy

in
Mechanical Engineering

Lehigh University
May, 2014

Approved and recommended for acceptance as a dissertation in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Date

Professor Terry J. Hart
Dissertation Adviser
Mechanical Engineering
Lehigh University

Accepted Date

Dr. David B. Spencer
Technical Adviser
Aerospace Engineering
The Pennsylvania State University

Dr. Nader Motee
Committee Chairperson
Mechanical Engineering
Lehigh University

Dr. Philip A. Blythe
Committee Member
Mechanical Engineering
Lehigh University

Dr. Gary G. DeLeo
Committee Member
Physics
Lehigh University

Acknowledgments

Over the course of my graduate studies I have encountered many burdensome obstacles that I could not have overcome without significant help from family, friends, and advisers. I would like to take a moment to personally thank those who have helped make this dissertation a reality. First I would like to thank my parents for their support and encouragement during my graduate studies (especially low-cost housing). I would also like to thank my parents-in-law for their generous support as well. Finally, I would like to thank my wife for all her hard work and support as well as the endless nights of proof-reading my papers, articles, and dissertation.

I would also like to recognize significant help that I received from professionals in the Astrodynamics community. Dr. Marty Ozimek and Dr. Chris Scott, from the Johns Hopkins Applied Physics Lab, spent hours via email helping me learn many of the concepts discussed in the first few chapters of this dissertation. To them I owe a large debt of gratitude for the time they have invested in me. I also would like to thank Dr. David B. Spencer, from Penn State University, for acting as a second academic adviser during my dissertation research. To him I owe a great deal of gratitude for lending me his technical expertise, guidance on papers and journal articles, and introducing me to many other experts within the Astrodynamics community. And finally, I would like to thank my adviser, Professor Hart, for all his time, support, and effort during my graduate studies.

Contents

List of Tables

List of Figures

Abstract

Missions to Lagrange points are becoming increasingly popular amongst spacecraft mission planners. Lagrange points are locations in space where the gravity force from two bodies, and the centrifugal force acting on a third body, cancel. To date, all spacecraft that have visited a Lagrange point have done so using high-thrust, chemical propulsion. Due to the increasing availability of low-thrust (high efficiency) propulsive devices, and their increasing capability in terms of fuel efficiency and instantaneous thrust, it has now become possible for a spacecraft to reach a Lagrange point orbit without the aid of chemical propellant. While at any given time there are many paths for a low-thrust trajectory to take, only one is optimal. The traditional approach to spacecraft trajectory optimization utilizes some form of gradient-based algorithm. While these algorithms offer numerous advantages, they also have a few significant shortcomings. The three most significant shortcomings are: (1) the fact that an initial guess solution is required to initialize the algorithm, (2) the radius of convergence can be quite small and can allow the algorithm to become trapped in local minima, and (3) gradient information is not always assessable nor always trustworthy for a given problem. To avoid these problems, this dissertation is focused on optimizing a low-thrust transfer trajectory from a geocentric orbit to an Earth-Moon, L_1, Lagrange point orbit using the method

of Particle Swarm Optimization (PSO). The PSO method is an evolutionary heuristic that was originally written to model birds swarming to locate hidden food sources. This PSO method will enable the exploration of the invariant stable manifold of the target Lagrange point orbit in an effort to optimize the spacecraft's low-thrust trajectory.

Examples of these optimized trajectories are presented and contrasted with those found using traditional, gradient-based approaches. In summary, the results of this dissertation find that the PSO method does, indeed, successfully optimize the low-thrust trajectory transfer problem without the need for initial guessing. Furthermore, a two-degree-of-freedom PSO problem formulation significantly outperformed a one-degree-of-freedom formulation by at least an order of magnitude, in terms of CPU time. Finally, the PSO method is also used to solve a traditional, two-burn, impulsive transfer to a Lagrange point orbit using a hybrid optimization algorithm that incorporates a gradient-based shooting algorithm as a pre-optimizer. Surprisingly, the results of this study show that "fast" transfers outperform "slow" transfers in terms of both Δv and time of flight.

1 Introduction

Lagrange Point Orbits (LPOs) have become increasingly important destinations for spacecraft missions. In the past, spacecraft would travel to these orbits via high-thrust, impulsive maneuvers. Unfortunately, these maneuvers utilize a great deal of fuel due to the relatively low efficiency of chemical propellant. Recently, attention has shifted to low-thrust propulsion technology that would allow a spacecraft to utilize a high-efficiency engine to transport itself to an LPO. This method, while slower than the chemical approach, offers a more fuel efficient method of transportation to/from an LPO and can maximize the payload mass transported. This type of system is ideal for cargo transportation where the cargo or payload is not harmed by long-duration spaceflight. In this dissertation, a method is developed to optimize a low-thrust trajectory from a geocentric orbit to an LPO using an evolutionary algorithm called Particle Swarm Optimization (PSO).

1.1 History of Lagrange Point Orbits

The Lagrange points (drawn in Figure 2.2) were discovered in the early 1770's by Leonhard Euler and Joseph Louis Lagrange [1]. Over time, it was discovered that the dynamics of the collinear Lagrange points were unstable, therefore any

object placed near a Lagrange point would eventually drift away. Fortunately, however, it was a bit easier to control orbits about these Lagrange points, although they, too, are generally unstable. In 1967 Szebehely [2] wrote a detailed explanation of analytic approximations of LPOs that were very useful in qualitatively describing the shape of these orbits and quantitatively describing their size and period. Farquahr [3] described the shape of a *halo* orbit in 1968 as a near-circular trajectory that orbited Lagrange point L_1 or L_2 in the Earth-Moon system. An observer on Earth who looked at the moon would see a *halo* shape traced around the moon by the spacecraft's trajectory. Other trajectories are also possible LPOs. These include a two-dimensional *Lyapunov* orbits (kidney bean shaped orbits) and three-dimensional *Lissajous* orbits which are semi-periodic in nature.

The first practical application of LPOs came in 1968 during the height of the space race. Farquahr [3, 4, 5] noted that halo orbits around the Earth-Moon L_2 point could be used by a single communications satellite to relay data back to Earth from the far-side of the Moon. This would enable communication with Apollo spacecraft in lunar orbit. The Apollo spacecraft would regularly loose line-of-sight radio contact with Earth each time it passed behind the "far side" of the Moon. It would also have allowed for a lunar landing on the far side, since it was a mission requirement that all lunar landing attempts be made while in communication with Earth. Unfortunately, the final Apollo 18 & 19 missions were canceled along with any hope of constructing a LPO communications satellite or landing on the lunar far side. Since then, other uses for Earth-Moon Lagrange points have been considered. These include observation and surveillance of cis-lunar space [6], navigation aids in cis-lunar space, and other human operations and telerobotics near Lagrange points [7, 8]. The placement

of fuel depots at Lagrange points [9] is also being considered for deep-space travel of human and robotic cargo.

There have also been a number of missions that have entered LPOs of the Sun-Earth system. The first spacecraft to enter an LPO was the International Sun-Earth Explorer (ISEE) mission in 1978. This mission was sent to a *halo* orbit about the Sun-Earth L_1 point to study the Sun [10]. Other missions include the Wilkinson Microwave Anisotropy Probe (WMAP) and the Planck Space Observatory. The Genesis mission trajectory was designed almost exclusively using Dynamical System Theory via heteroclinic connections between LPOs [11]. A NASA flagship mission called the James Webb Space Telescope (JWST) is also scheduled to be launched to the Sun-Earth L_2 point in 2018 [12]. LPOs in the Earth-Moon system have also been utilized. The ARTEMIS mission (flown in 2011) [13, 14] was sent to study the distant regions of Earth's magnetotail by entering Earth-Moon L_1 and L_2 *Lyapunov* orbits (in practice they were technically *Lissajous* orbits because various perturbations from the sun and planets prevented them from being perfectly periodic). To date, this is the first and only mission to an Earth-Moon Lagrange point.

1.2 History & Background of Low-Thrust Propulsion

In addition to standard, chemical, impulsive, high-thrust propulsion (studied in the section labeled "Two-Maneuver, Impulsive Transfers via a Hybrid PSO/Shooting Method") a second method exists to deliver spacecraft to Lagrange point orbits. Low-thrust propulsion is useful for changing the orbit of a

spacecraft that is already in space (and is useless in the Earth's atmosphere due to drag and gravity losses) and can have many advantages over chemical (high-thrust) technology. The main advantage of low-thrust technology is its high specific impulse when compared with traditional chemical propulsion. According to the rocket equation $\Delta v = I_{sp} g_0 \ln\left(\frac{m_0}{m_f}\right)$ where Δv is the change in speed of the rocket, g_0, is the acceleration due to gravity at sea level (a constant) and m_0 and m_f are the rocket's initial and final masses (including fuel). The I_{sp} is a measure of the efficiency of the rocket and is an intrinsic property of the rocket and the fuel it burns. An I_{sp} of $3000 - 9000$ seconds is possible with low-thrust technology as opposed to $200 - 400$ seconds with chemical technology; obviously a far more fuel efficient technology than that required for the same change in velocity as chemical propellants. The main disadvantage of low-thrust propulsion is implied by its name: "low-thrust." It might take hours, days, months, or even years of continuous low-thrust operation to acquire the same Δv as a chemical rocket can provide in a matter of minutes. Most low-thrust devices produce far less than one newton of force during their operation.

In general there are three broad categories of low-thrust propulsion: solar-electric, variable specific impulse, and solar sailing (which, technically, requires no fuel but can be difficult to implement and control). This dissertation will focus on solar-electric low-thrust propulsion and assume a simple case of constant specific impulse and thrust which is generally valid in the Earth-Moon system if avoiding Earth's shadow. A solar-electric platform uses an array of solar cells to produce the electricity necessary (in the kilowatt range [15]) to accelerate charged particles away from the spacecraft to produce thrust. Today, modern electric thrusters provide for station keeping for the most part, but also can be used for minor orbital maintenance of communication satellites. According to

Sovey et al. [15], the higher specific impulse of low-thrust technology saves so much fuel mass that it nearly doubles the mass available for communications equipment. It has also been used to propel spacecraft such as Deep Space 1 and Dawn. Deep Space 1 was launched in 1998 with the NSTAR engine that was capable of generating 92 mN of thrust with an $I_{sp}$of $3,100$ seconds while using 2.3kW of power generated by solar panels and running for $9,241$ hours [16]. Later, in 2007, NASA launched the Dawn mission to study asteroid Vesta and dwarf planet Ceres using a similar engine [17, 18]. As of 2004, nearly 200 solar-electric satellites in Earth orbit utilize some form of low-thrust, electric propulsion [19]. This growth has expanded rapidly as new technology has been developed.

Low-thrust ion propulsion was first studied by Robert H. Goddard in 1916 while experimenting with ionized thrust in near-vacuum conditions at high altitudes [20]. Then, in 1923, Hermann Oberth discussed possible fuel savings of electric propulsion of charged gases in his book "Wege zur Raumschiffahrt" or "Ways to Spaceflight" [21]. Unfortunately, the development of low-thrust technology did not proceed very quickly because the first man-made satellites in space did not occur until the late 1950's. In 1959 Harold Kaufman built and tested the first working ion thruster at NASA Glenn (then Lewis) Research Center [15]. Suborbital and orbital testing of this concept occurred in the 1960's and 1970's with the Space Electric Rocket Test (SERT) program. A similar program, the Solar Electric Propulsion System Technology (SEPST), was also established by Jet Propulsion Laboratory around the same time [22]. Its goal was to develop low-thrust ion technology for interplanetary spacecraft and it succeeded by developing and ground-testing a 2.5kW, 88mN ion engine with an I_{sp} of $3,600$ seconds [22]. Of course, the first active mission to utilize

low-thrust technology was Deep Space 1, as previously stated.

1.3 Previous Work

Initial low-thrust optimization work dealt with the indirect method of locally minimizing an objective function and solving a Two Point Boundary Value Problem (TPBVP). These initial studies (1960's) focused primarily on geocentric trajectories consisting of a low-thrust spiral and interplanetary transfers [23, 24, 25]. Over time, many authors have discussed gradient-based methods to optimize low-thrust transfers in two-body environments [26, 27, 28, 29]. Still, the optimization of low-thrust trajectories in a three-body environment using stable manifold theory was not thoroughly studied until Mingotti's series of papers in the late 2000's [30, 31, 32]. Mingotti's work paved the way for the blending of Dynamical Systems Theory (DST) with low-thrust, gradient-based optimization pioneered in the 1960's and 1970's. Senet and Ocampo also contributed to this field by using similar techniques to study low-thrust delivery of spacecraft to Lagrange point orbits [33]. Unfortunately, however, all of these methods rely on gradient-based optimization techniques that are susceptible to poor initial guess solutions and can have a narrow radius of convergence.

The idea of heuristic spacecraft trajectory optimization has often been used to overcome some of the inherent limitations of gradient-based optimizers. One evolutionary method called Particle Swarm Optimization (PSO) is becoming increasingly popular amongst spacecraft mission designers. Pontani and Conway explored the application of the PSO method to various spacecraft trajectory optimization problems [34, 35, 36]. Interplanetary trajectories were optimized via PSO by Bessette and Spencer [37, 38] in the late 2000's. Unfortunately,

no researcher to date has attempted to use PSO in conjunction with, or as a substitute for, traditional gradient-based approaches when designing a transfer (either impulsive or low-thrust) to a Lagrange point orbit. The remainder of this dissertation will focus on the solution to this problem using the PSO method.

1.4 Present Work

In light of the previous work performed in trajectory optimization, this dissertation will focus on using gradient based methods, evolutionary algorithms, and Dynamical Systems Theory to optimize a low-thrust trajectory from a geocentric orbit to an Earth-Moon Lagrange point orbit. The application of the PSO method is intended to enable the global exploration of the search-space without a need for an *a priori* guess trajectory that is oftentimes too difficult for the researcher to generate. It is a fast, easy to use and understandable algorithm that can be programmed and run quickly. This is in contrast with many gradient based algorithms that require detailed programming and guess solutions and are limited to the optimization of a local search-space. The approach designed in this dissertation can be used independently of other methods or as a preliminary, global optimization algorithm whose results are used as initial guess solutions for the gradient-based algorithms.

A brief summary of each chapter in this dissertation is presented here:

- Chapter 2 - Details of the dynamics of the problem are presented. The coordinate system and equations of motion of the Circular Restricted Three-Body Problem (CR3BP) are derived as well as the concept of the Jacobi energy, Lagrange points, and the stability of the Lagrange points.

The State Transition Matrix (STM) is also defined and discussed.

- Chapter 3 - Focuses on gradient-based optimization methods. Generalized free variable and constraint vectors are defined to enable the discussion of shooting algorithms. Single shooting and multiple shooting algorithms are derived, as well as the method of pseudo-arclength continuation. The chapter concludes with a discussion of a full-ephemeris integrator as well as the SPICE/Mice packages used to implement such an integrator.

- Chapter 4 - Implements the theory covered in Chapter 3 on real-world examples of Earth-Moon Lagrange point orbits. Various trajectories are generated which include Lyapunov, halo, and Lissajous orbits.

- Chapter 5 - Introduces Dynamical Systems Theory in order to derive the existence of the invariant stable manifold of a Lagrange point orbit. The chapter begins with a discussion about the linearization of motion about an LPO and uses Floquet's theorem to map the dynamics to an autonomous form. The monodromy matrix is defined and used to investigate the stable and unstable manifolds of an LPO of interest. Finally, a method to generate the stable manifold is discussed and demonstrated on a target LPO.

- Chapter 6 - Particle Swarm Optimization is introduced. The chapter begins with a brief background on the PSO method. It then defines the search-space and objective function used throughout the rest of this dissertation as well as the boundary conditions associated with this search-space.

- Chapter 7 - This chapter presents original research performed using a one-

dimensional formulation of the PSO method. A low-thrust trajectory is defined by a simple control law and is optimized to transport a spacecraft from a Geostationary-altitude orbit to an Earth-Moon, L_1, northern halo orbit.

- Chapter 8 - This chapter builds on the content of Chapter 7 by extending the PSO algorithm into a two-dimensional form. Attention is focused on methods to size various parameters of the PSO algorithm to obtain high algorithmic performance. An optimal low-thrust trajectory is obtained and results are compared with current research.

- Chapter 9 - This chapter covers the optimization of two-maneuver, impulsive trajectories from Low Earth Orbit (LEO) to the same Earth-Moon, L_1, northern halo orbit using a hybrid PSO/shooting method. "Fast" and "slow" transfers are studied and results are compared with other published work.

- Chapter 10 - Conclusions made in Chapters 7-9 are presented as well as general conclusions about the application of the PSO method to Dynamical Systems Theory. Potential future work is also discussed.

2 Dynamics

This chapter describes the dynamics associated with a spacecraft traveling in the presence of a gravity field from both the Earth and Moon, simultaneously. The differential equations associated with the Circular Restricted Three Body Problem (CR3BP) are the dynamics of choice in this dissertation. The CR3BP was chosen because it is the simplest dynamical model of the Earth-Moon system that preserves the dynamics associated with Lagrange point orbits; a key focus of this dissertation. Despite the simplicity of the CR3BP, it has been shown to be too complex for analytical solutions of the CR3BP differential equations of motion to exist [2]. This is in stark contrast with the differential equations of motion of traditional spacecraft orbiting a point mass which have analytic solutions expressed in terms of conic sections. Unfortunately, these "Keplerian" conic orbits preclude the possibility of Lagrange point orbits due to their low fidelity in the three-body environment of cislunar space.

The chapter begins with a discussion defining the coordinate system of the CR3BP. This synodic reference frame is constructed in non-dimensional units to simplify the problem and increase the accuracy of the numeric integration of the system's equations of motion. Next, the equations of motion are derived using the Euler-Lagrange equations and written using state vector notation. An integral constant of the system, known as the Jacobi integral, is defined

and demonstrated as the only constant of the three body system. Equilibrium points of the CR3BP are derived and called "Lagrange Points" after the mathematician who first discovered them in the 1700's [2]. The Jacobi integral can be calculated at each Lagrange point and used to define "forbidden regions" where the spacecraft does not have enough energy to enter. Finally the chapter closes with a discussion of the stability characteristics of the Lagrange points as well as an introduction to the State Transition Matrix (STM) which will be used heavily in later chapters.

2.1 Coordinate System and Definitions

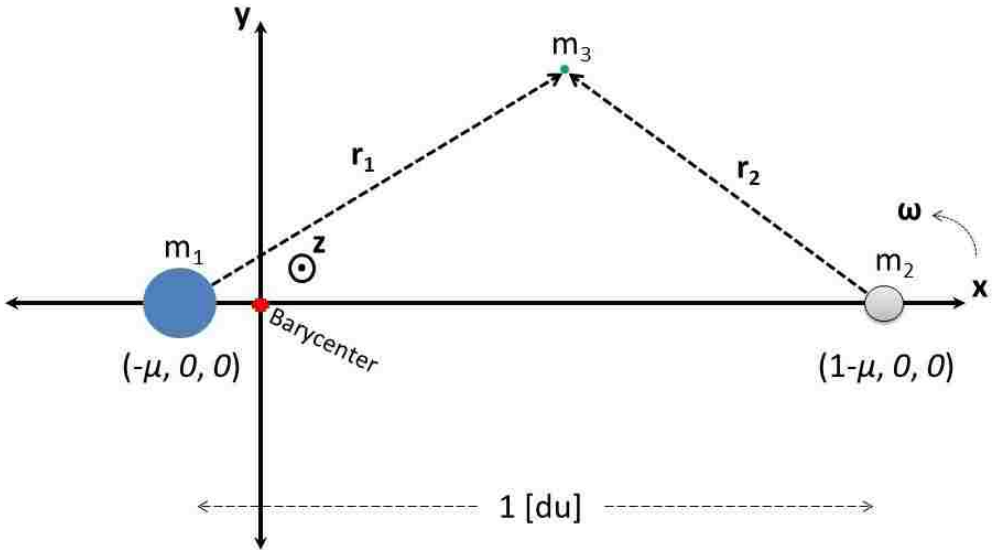


Figure 2.1: CR3BP coordinate system.

In this derivation, an attempt will be made to non-dimensionalize as many quantities as possible in order to simplify the system. First, the distance between Primary 1 (Earth) and Primary 2 (Moon) is set to one "distance unit,"

[du] that is equal to 388,400 km. Next, the reduced mass, μ, is defined as

$$\mu \equiv \frac{m_2}{m_1 + m_2} \tag{2.1.1}$$

with $\mu \in \left(0, \ \frac{1}{2} \right]$. The masses follow the convention $m_1 \geq m_2 \gg m_3$ with m_3, the spacecraft mass, being negligible compared with either mass m_1 or m_2. Mass m_3 is so insignificant, in fact, that it does not affect the motion of either primary (hence the "restriction" in the CR3BP). Due to this assumption, it can be stated that $m_1 + m_2 + m_3 \cong m_1 + m_2$. Furthermore, for the sake of simplicity, define the units of mass in such a way as to set the total mass of the system $m_1 + m_2 = 1$. The Center-of-Mass (CM) of the system (commonly known as the barycenter) can be calculated in the usual way:

$$CM_{Relative\,to\,m_1} = \frac{m_1}{m_1 + m_2}(0) + \frac{m_2}{m_1 + m_2}(1) = \mu \tag{2.1.2}$$

or

$$CM_{Relative\,to\,m_2} = 1 - CM_{Relative\,to\,m_1} = 1 - \mu. \tag{2.1.3}$$

A coordinate system is chosen with the barycenter as its origin as can be seen in Figure 2.1. The positive x-axis is defined as the direction from the larger primary to the smaller primary. The positive y-axis is defined to be in the same direction as the velocity vector of the smaller primary (which is perpendicular to the x-axis due to the circular nature of its orbit). Finally, the positive z-axis is defined in such a way as to complete the right-handed triad based on the x and y unit vectors. A non-inertial reference frame will be utilized; this frame shall rotate with an angular velocity, ω, that exactly matches that of the two primaries which are orbit about their common barycenter. Note that m_1 is

located at $(-\mu, 0, 0)$ and m_2 at $(1-\mu, 0, 0)$ in this frame. This co-rotating reference frame restricts the two primaries to remain in fixed positions along the x-axis. A non-inertial reference frame may seem, at first, to be a needless complication. Fortunately, however, this choice of coordinate system greatly simplifies the problem as will be demonstrated in the subsequent sections of this chapter.

2.2 System Dynamics

The kinetic energy and gravitational potential energy of the spacecraft, m_3, can be written as

$$T = \frac{1}{2}m_3 v^2 = \frac{1}{2}m_3\left[(\dot{x}-\omega y)^2 + (\dot{y}+\omega x)^2 + \dot{z}^2\right] \tag{2.2.1}$$

$$V = -\frac{Gm_3m_1}{r_1} - \frac{Gm_3m_2}{r_2} \tag{2.2.2}$$

and, when added, constitute the total mechanical energy of the system

$$E = T + V \tag{2.2.3}$$

$$E = m_3\left(\frac{1}{2}\left[(\dot{x}-\omega y)^2 + (\dot{y}+\omega x)^2 + \dot{z}^2\right] - \frac{Gm_1}{r_1} - \frac{Gm_2}{r_2}\right) \tag{2.2.4}$$

with the distances $r_1 = \sqrt{(x+\mu)^2 + y^2 + z^2}$ and $r_2 = \sqrt{(x-[1-\mu])^2 + y^2 + z^2}$ being the distances from the spacecraft to the Earth and Moon, respectively. Note that the total energy of the system needs further simplification. Because the gravitational influence of m_3 is negligible, the motion of mass $m_1$and m_2 are governed by classical two-body dynamics. From a thorough study of Kepler's laws and Newtonian mechanics, it can be shown that the angular velocity ω of

two, circular, co-rotating masses is given by

$$\omega = \sqrt{\frac{G(m_1 + m_2)}{r_{1,2}^3}}. \tag{2.2.5}$$

This can be further simplified by utilizing the dimensionless units and quantities as defined above

$$\omega = \sqrt{\frac{G(1)}{(1)}} = \sqrt{G}. \tag{2.2.6}$$

Notice that the dimensionless definitions helped to significantly reduce the angular velocity equation. Further reduction of this equation is possible by cleverly defining the dimensionless time units [tu] of the system to be

$$1\,[\text{tu}] = \frac{T}{2\pi} \tag{2.2.7}$$

with T being the period of the orbits of the primaries. For the Earth-Moon system the sidereal period (orbital period) of the moon is 27.3 days. This translates to $1\,[\text{tu}] = 4.345\,[\text{days}]$ in the Earth-Moon system. This substitution will define

$$\sqrt{G} = \omega = \frac{2\pi}{2\pi}\left[\frac{\text{rad}}{\text{tu}}\right] = 1 \tag{2.2.8}$$

and will further simplify the total energy per unit mass as

$$\frac{E}{m_3} = \varepsilon = \frac{1}{2}\left[(\dot{x} - y)^2 + (\dot{y} + x)^2 + \dot{z}^2\right] - \frac{1-\mu}{r_1} - \frac{\mu}{r_2} \tag{2.2.9}$$

which is a greatly simplified expression. Note that all simplifying assumptions are system specific. For example, the non-normalized units in the Earth-Moon system are very different from that of the Sun-Earth system even though their normalized (three-body) units are identical.

One can also write the Lagrangian, L, of the system by noting that

$$T = \frac{1}{2} m_3 \left[(\dot{x} - y)^2 + (\dot{y} + x)^2 + \dot{z}^2 \right] \tag{2.2.10}$$

$$V = -m_3 \left\{ \frac{1-\mu}{\sqrt{(x+\mu)^2 + y^2 + z^2}} + \frac{\mu}{\sqrt{(x-[1-\mu])^2 + y^2 + z^2}} \right\} \tag{2.2.11}$$

$$L = T - V \tag{2.2.12}$$

$$L = m_3 \left\{ \frac{1}{2} \left[(\dot{x} - y)^2 + (\dot{y} + x)^2 + \dot{z}^2 \right] + \frac{1-\mu}{\sqrt{(x+\mu)^2 + y^2 + z^2}} + \frac{\mu}{\sqrt{(x-[1-\mu])^2 + y^2 + z^2}} \right\}.$$

Using the Euler-Lagrange equations one can write the equations of motion of the spacecraft as

$$\frac{\partial L}{\partial x} - \frac{d}{dt} \frac{\partial L}{\partial \dot{x}} = 0 \tag{2.2.13}$$

$$\dot{y} + x - \frac{\partial V}{\partial x} - \frac{d}{dt} \left[\dot{x} - y \right] = 0 \tag{2.2.14}$$

$$\ddot{x} - 2\dot{y} = -\frac{\partial V}{\partial x} + x \tag{2.2.15}$$

$$\frac{\partial L}{\partial y} - \frac{d}{dt} \frac{\partial L}{\partial \dot{y}} = 0 \tag{2.2.16}$$

$$-(\dot{x} - y) - \frac{\partial V}{\partial y} - \frac{d}{dt} \left[\dot{y} + x \right] = 0 \tag{2.2.17}$$

$$\ddot{y} + 2\dot{x} = -\frac{\partial V}{\partial y} + y \tag{2.2.18}$$

$$\frac{\partial L}{\partial z} - \frac{d}{dt}\frac{\partial L}{\partial \dot{z}} = 0 \tag{2.2.19}$$

$$-\frac{\partial V}{\partial z} - \frac{d}{dt}\left[\dot{z}\right] = 0 \tag{2.2.20}$$

$$\ddot{z} = -\frac{\partial V}{\partial z}. \tag{2.2.21}$$

For one final simplification, the potential function present in Equation 2.2.15 and Equation 2.2.18 can be re-defined. Let

$$U = V + f_{(x,\,y)}. \tag{2.2.22}$$

Then

$$-\frac{\partial U}{\partial x} \Longrightarrow -\frac{\partial V}{\partial x} - \frac{\partial f}{\partial x} = -\frac{\partial V}{\partial x} + x \longrightarrow \frac{\partial f}{\partial x} = -x \longrightarrow f = -\frac{1}{2}x^2 + g_{(y)} \tag{2.2.23}$$

and

$$-\frac{\partial U}{\partial y} \Longrightarrow -\frac{\partial V}{\partial y} - \frac{\partial f}{\partial y} = -\frac{\partial V}{\partial y} + y \longrightarrow \frac{\partial f}{\partial y} = -y \longrightarrow f = -\frac{1}{2}y^2 + h_{(x)} \tag{2.2.24}$$

where $g_{(y)}$ is an arbitrary function of y only and $h_{(x)}$ is an arbitrary function of x only. Matching these results one obtains an expression for U as

$$f = -\frac{1}{2}\left(x^2 + y^2\right) \quad \longrightarrow \quad U = V - \frac{1}{2}\left(x^2 + y^2\right) \tag{2.2.25}$$

$$U = -\frac{1-\mu}{\sqrt{(x+\mu)^2 + y^2 + z^2}} - \frac{\mu}{\sqrt{(x-[1-\mu])^2 + y^2 + z^2}} - \frac{1}{2}\left(x^2 + y^2\right) \tag{2.2.26}$$

with U known as the "pseudopotential." Now Equation 2.2.15, 2.2.18, and

2.2.21 simplify to

$$\begin{aligned} \ddot{x} - 2\dot{y} &= -\frac{\partial U}{\partial x} \\ \ddot{y} + 2\dot{x} &= -\frac{\partial U}{\partial y} \\ \ddot{z} &= -\frac{\partial U}{\partial z} \end{aligned} \tag{2.2.27}$$

which are the equations of motion of the system. These ballistic equations of motion can be re-written in ballistic state-space form as

$$\mathbf{X} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \tag{2.2.28}$$

with

$$\dot{\boldsymbol{X}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ 2\dot{y} - \frac{\partial U}{\partial x} \\ -2\dot{x} - \frac{\partial U}{\partial y} \\ -\frac{\partial U}{\partial z} \end{bmatrix}. \tag{2.2.29}$$

As an augmentation to the ballistic Equations 2.2.28 and 2.2.29 one can also

include the influence of a rocket-propelled thruster as

$$\mathbf{X} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ m \end{bmatrix} \tag{2.2.30}$$

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \dot{m} \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ 2v_y + U_x + u_x \\ -2v_x + U_y + u_y \\ U_z + u_z \\ -\frac{T}{I_{sp} g_o} \end{bmatrix} \tag{2.2.31}$$

with thrust, T, specific impulse, I_{sp}, and the constant gravitational acceleration at sea level of, g_0. The thruster will consume mass at an instantaneous mass flow rate of $\dot{m}$ and produce an acceleration of u that can be broken up into vector components. Obviously Equations 2.2.30 and 2.2.31 degenerate into the ballistic Equations 2.2.28 and 2.2.29 under the absence of propulsive devices. Note that in this dissertation the state $\mathbf{X}$ and $\dot{\mathbf{X}}$ can be of length six or seven depending on the presence of thrust. If no thrust is present then the final (seventh) state is always a constant and is dropped from the state vector notation.

2.3 Jacobi Energy Integral

Now that the equations of motion have been derived, one can study the only conserved quantity in the CR3BP; the Jacobi Energy. Unlike the two-body problem where both energy and angular momentum are conserved, the three-body problem will conserve only the energy of the system (i.e. the Jacobi energy, Jacobi integral, or Jacobi constant). To begin this analysis, first look at the time derivative of the square of the velocity

$$\frac{d}{dt}\,v^2 = \frac{d}{dt}\left(\dot{x}^2 + \dot{y}^2 + \dot{z}^2\right) = 2\left(\dot{x}\ddot{x} + \dot{y}\ddot{y} + \dot{z}\ddot{z}\right). \tag{2.3.1}$$

By substituting the equations of motion (Equation 2.2.27) one will obtain

$$\begin{aligned}\frac{d}{dt}\,v^2 &= 2\left[\dot{x}\left(2\,\dot{y} - \frac{\partial U}{\partial x}\right) + \dot{y}\left(-2\,\dot{x} - \frac{\partial U}{\partial y}\right) + \dot{z}\left(-\frac{\partial U}{\partial z}\right)\right] \\ &= 2\left[\dot{x}\left(-\frac{\partial U}{\partial x}\right) + \dot{y}\left(-\frac{\partial U}{\partial y}\right) + \dot{z}\left(-\frac{\partial U}{\partial z}\right)\right].\end{aligned} \tag{2.3.2}$$

Recall the fact that

$$\frac{dU}{dt} = \dot{x}\left(\frac{\partial U}{\partial x}\right) + \dot{y}\left(\frac{\partial U}{\partial y}\right) + \dot{z}\left(\frac{\partial U}{\partial z}\right) \tag{2.3.3}$$

which simplifies Equation 2.3.2 to

$$\frac{d}{dt}\,v^2 = -2\frac{dU}{dt} \quad \longrightarrow \quad \frac{d}{dt}\left[-v^2 - 2U\right] = 0. \tag{2.3.4}$$

Integrating with respect to time gives

$$J = -v^2 - 2U \tag{2.3.5}$$

to within an arbitrary constant that is assumed to be zero. The constant J is known as the Jacobi constant and represents the only known constant of the CR3BP. It can be seen that J is related to the total energy per unit mass (as measured in the rotating frame) [2], ε by:

$$\varepsilon = \tfrac{1}{2}\, v^2 + U = -\tfrac{1}{2}\left(-v^2 - 2U\right) = -\tfrac{1}{2} J \tag{2.3.6}$$

$$J = -2\varepsilon. \tag{2.3.7}$$

One final note concerning the Jacobi energy centers on the fact that the square of the velocity cannot, by definition, be negative. If it were negative then the velocity would have to be imaginary which doesn't make physical sense. If a projectile was launched, at a speed less than the escape speed, in a rectilinear trajectory from the surface of one of the primaries and into space, then eventually that projectile would slow down, stop, and then turn around. The farthest distance accessible to the projectile would occur when its velocity was exactly zero. Any region of space beyond that point is forbidden by the physics of the system (i.e. the system's energy). Therefore, it becomes possible to map both the accessible and forbidden regions of any third-body if given its associated Jacobi Energy by the relation

$$J = -2U_{max}. \tag{2.3.8}$$

This equation implicitly defines an energy surface or energy manifold; no body with a given energy value can escape this manifold. Historically, the manifold of accessible space is known as "Hill's Region" with the boundary of this region known as the "zero velocity curve."

2.4 Equilibrium Points

The equations of motion (Equation 2.2.27) have five equilibrium points, labeled L_1-L_5, that are plotted in Figure 2.2 for the Earth-Moon system of $\mu = 0.01215$.

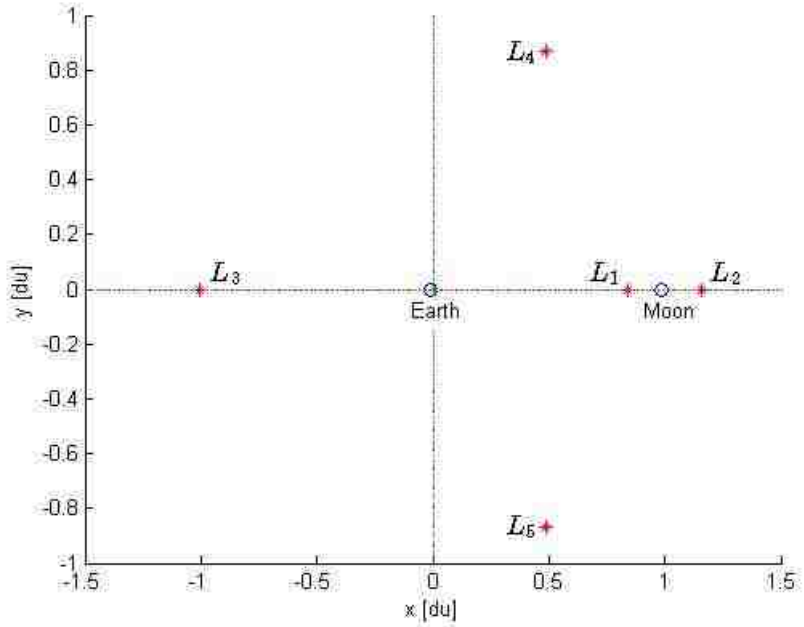


Figure 2.2: Five Lagrange points for the Earth-Moon system.

It is easy to calculate the locations of these Lagrange points. This occurs whenever $\ddot{x} = \dot{x} = \ddot{y} = \dot{y} = 0$ and Equation 2.2.27 becomes

$$0 = -\frac{\partial U}{\partial x} = \frac{(\mu - 1)(x + \mu)}{\left[(x+\mu)^2 + y^2 + z^2\right]^{\frac{3}{2}}} - \frac{\mu(x + \mu - 1)}{\left[(x+\mu-1)^2 + y^2 + z^2\right]^{\frac{3}{2}}} + x \quad (2.4.1)$$

$$0 = -\frac{\partial U}{\partial y} = \frac{(\mu - 1)\, y}{\left[(x+\mu)^2 + y^2 + z^2\right]^{\frac{3}{2}}} - \frac{\mu y}{\left[(x+\mu-1)^2 + y^2 + z^2\right]^{\frac{3}{2}}} + y \quad (2.4.2)$$

$$0 = -\frac{\partial U}{\partial z} = \frac{(\mu - 1)\, z}{\left[(x+\mu)^2 + y^2 + z^2\right]^{\frac{3}{2}}} - \frac{\mu z}{\left[(x+\mu-1)^2 + y^2 + z^2\right]^{\frac{3}{2}}}. \quad (2.4.3)$$

Note that Equation 2.4.2 and 2.4.3 are immediately satisfied anytime $y = 0$ and $z = 0$, respectively. Thus, it becomes prudent to begin exploring the existence of a stationary point that lies upon the x-axis. When $y = z = 0$, Equation 2.4.1 simplifies to

$$x + \frac{(\mu - 1)(x + \mu)}{|x + \mu|^3} - \frac{\mu(x + \mu - 1)}{|x + \mu - 1|^3} = 0. \tag{2.4.4}$$

It is very important to note that the absolute value signs in Equation 2.4.4 must be present as they are a direct consequence of the squaring of the terms in the denominator. But how can one mathematically deal with the absolute value in Equation 2.4.4? The solution is to break up Equation 2.4.4 into three different regions based on the magnitudes of the terms within the absolute value operator.

1. Region I consists of the section of the $x-$axis between Primary 1 and Primary 2 where L_1 is defined by $-\mu < L_1 < 1 - \mu$. Note that $|L_1 + \mu| = L_1 + \mu$ and $|L_1 + \mu - 1| = -(L_1 + \mu - 1)$. Substituting this into Equation 2.4.4 gives

$$L_1(L_1 + \mu)^2(L_1 + \mu - 1)^2 + (\mu - 1)(L_1 + \mu - 1)^2 + \mu(L_1 + \mu)^2 = 0. \tag{2.4.5}$$

   If Equation 2.4.5 is expanded a fifth order polynomial is obtained in terms of L_1. Unfortunately, this polynomial can not be solved for algebraically; only a numerical solution using a numeric value for μ can be found. The numerical solution will yield five roots; one of which is real. This real-root is the location of the stationary point corresponding to L_1. In the case of the Earth-Moon system, where $\mu = 0.0122$, the position of $L_1 = 0.8369$[du].

2. Region II consist of the $x-$axis lying to the right of Primary 2 where L_2 is defined by $-\mu < 1-\mu < L_2$. Note that $|L_2+\mu| = L_2+\mu$ and $|L_2+\mu-1| = L_2+\mu-1$. Substituting this into Equation 2.4.4 gives

$$L_2\left(L_2+\mu\right)^2\left(L_2+\mu-1\right)^2+(\mu-1)\left(L_2+\mu-1\right)^2-\mu\left(L_2+\mu\right)^2=0. \tag{2.4.6}$$

Again, this equation is a fifth order polynomial that contains only one real-root and can only be solved for numerically. In the case of the Earth-Moon system the position of $L_2 = 1.1557\,[\text{du}]$.

3. L_3 is located in Region III. This region will lie to the left of Primary 1 and is defined by $L_3 < -\mu < 1-\mu$. Note that $|L_3+\mu| = -\left(L_3+\mu\right)$ and $|L_3+\mu-1| = -\left(L_3+\mu-1\right)$. Substituting this into Equation 2.4.4 gives

$$L_3\left(L_3+\mu\right)^2\left(L_3+\mu-1\right)^2-(\mu-1)\left(L_3+\mu-1\right)^2+\mu\left(L_3+\mu\right)^2=0. \tag{2.4.7}$$

As before, this equation is a fifth order polynomial that contains only one real-root and can only be solved for numerically. In the case of the Earth-Moon system the position of $L_3 = -1.00506\,[\text{du}]$.

In this way three Lagrange points, L_1, L_2 and L_3 can be found when $y = z = 0$. A simple process is employed to find the remaining Lagrange points when $y \neq 0$. The solution is actually intractable without taking advantage of the symmetry of the system. Referring to Equation 2.4.1, this equation is actually very difficult to deal with unless the values of the two denominators are equal to each other. This is the key to making the problem tractable. By inspection, one

will note that the values $x = \frac{1}{2} - \mu,\ z = 0$ would be an appropriate substitution that allows for further simplification of Equation 2.4.1 and Equation 2.4.2 to $y = \pm\sqrt{\frac{3}{4}}$. Thus the remaining two Lagrange points are symmetric and have values of $L_4 = \left(\frac{1}{2} - \mu,\ \sqrt{\frac{3}{4}}\right)$ [du] and $L_4 = \left(\frac{1}{2} - \mu,\ -\sqrt{\frac{3}{4}}\right)$ [du].

2.5 Jacobi Energy at Equilibrium Points

Interestingly enough, one can calculate the Jacobi energy of a particle at rest (in the co-rotating system) and positioned at any of the five Lagrange points. Table 2.1 summarizes the locations and energies of Lagrange Points for the Earth-Moon system.

Table 2.1: Summary of Jacobi energies of the Lagrange points in the Earth-Moon system.

	x [du]	y [du]	z [du]	Stability	J	E $[Kg^2s^{-2}]$
L_1	0.8369	0	0	saddle	3.1883	-1.6735
L_2	1.15569	0	0	saddle	3.1722	-1.665
L_3	-1.00506	0	0	saddle	3.0121	-1.581
L_4	0.487846	0.86602	0	stable	2.9879	-1.5683
L_5	0.487846	-0.86602	0	stable	2.9879	-1.5683

Correspondingly, Figure 2.3 is a series of graphs that plot the Jacobi energies J_1-$J_{4,5}$ for their corresponding Lagrange Points. The black area in Figure 2.3 represents the "forbidden region" in which the third body is forbidden to exist given a fixed amount of energy. At energies below J_1 there are three disconnected realms: the Earth-centered realm, the Moon-centered realm, and the exterior realm. Once the energy is equal to that of J_1, the Earth and Moon realms merge at the location of L_1. It now becomes possible for the third body to move from the Earth realm to the Moon realm (and visa versa) through the

"neck" region located around L_1. As the energy increases further to that of L_2, the Earth, Moon, and exterior realms all become connected. At this energy level it becomes possible for the third body to freely move between all realms by traveling through the L_1-Moon-L_2 neck region. As the energy is further increased to that of L_3, another neck opens in the opposite direction. Finally, once the energy approaches $J_{4,5}$ all of the "forbidden" regions cease to exist. This study of Jacobi energy implies that, from an energy standpoint, it is easiest for the spacecraft to get to the L_1 point, and the most difficult for it to get to the triangular points.

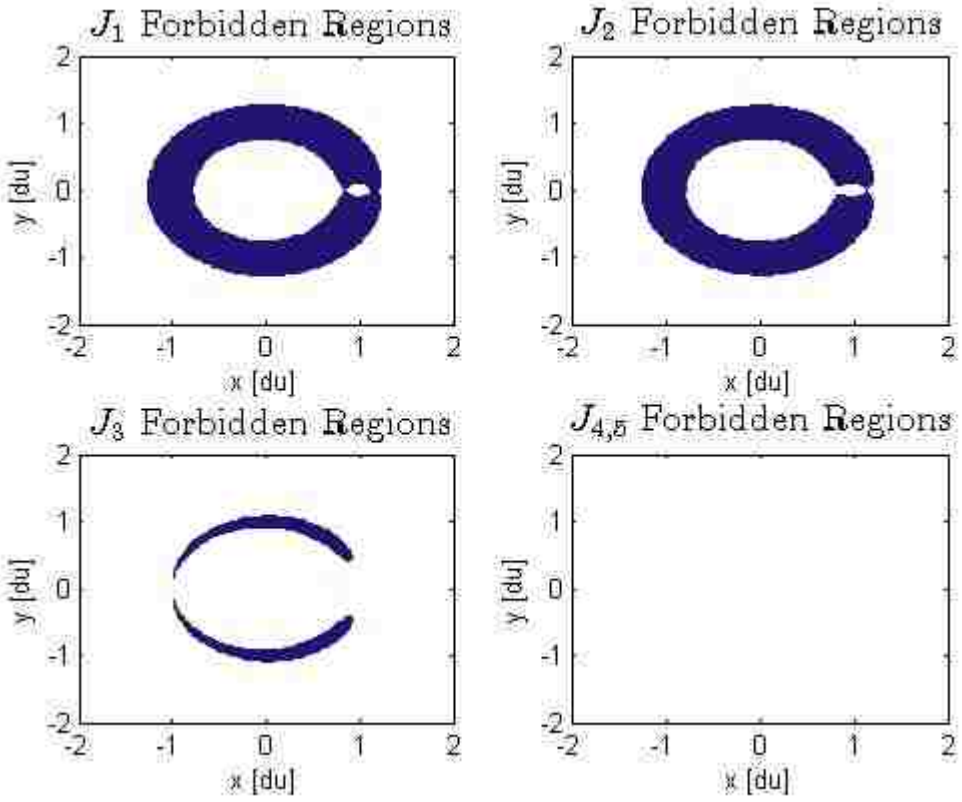


Figure 2.3: Forbidden regions with Jacobi energies (from upper left to lower right) corresponding to L_1, L_2, L_3, and $L_{4,5}$.

2.6 Linearization, Stability, and Bifurcation

Now that the positions of the Lagrange points are known, it is helpful to characterize their stability. According to Equation 2.2.29 the time derivative of the ballistic state vector is

$$\dot{\boldsymbol{X}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \vec{f}(\boldsymbol{X}) = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ 2\dot{y} - \frac{\partial U}{\partial x} \\ -2\dot{x} - \frac{\partial U}{\partial y} \\ -\frac{\partial U}{\partial z} \end{bmatrix}. \tag{2.6.1}$$

The Jacobian Matrix, $A(t)$, can be written as

$$A(t) = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -\frac{\partial^2 U}{\partial x^2} & -\frac{\partial^2 U}{\partial x \partial y} & -\frac{\partial^2 U}{\partial x \partial z} & -\frac{\partial^2 U}{\partial x \partial \dot{x}} & 2 - \frac{\partial^2 U}{\partial x \partial \dot{y}} & \frac{\partial^2 U}{\partial x \partial \dot{z}} \\ -\frac{\partial^2 U}{\partial y \partial x} & -\frac{\partial^2 U}{\partial y^2} & -\frac{\partial^2 U}{\partial y \partial z} & -2 - \frac{\partial^2 U}{\partial y \partial \dot{x}} & -\frac{\partial^2 U}{\partial y \partial \dot{y}} & -\frac{\partial^2 U}{\partial y \partial \dot{z}} \\ -\frac{\partial^2 U}{\partial z \partial x} & -\frac{\partial^2 U}{\partial z \partial y} & -\frac{\partial^2 U}{\partial z \partial z} & -\frac{\partial^2 U}{\partial z \partial \dot{x}} & -\frac{\partial^2 U}{\partial z \partial \dot{y}} & -\frac{\partial^2 U}{\partial z \partial \dot{z}} \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ -\frac{\partial^2 U}{\partial x^2} & -\frac{\partial^2 U}{\partial x \partial y} & -\frac{\partial^2 U}{\partial x \partial z} & 0 & 2 & 0 \\ -\frac{\partial^2 U}{\partial y \partial x} & -\frac{\partial^2 U}{\partial y^2} & -\frac{\partial^2 U}{\partial y \partial z} & -2 & 0 & 0 \\ -\frac{\partial^2 U}{\partial z \partial x} & -\frac{\partial^2 U}{\partial z \partial y} & -\frac{\partial^2 U}{\partial z^2} & 0 & 0 & 0 \end{bmatrix}$$

which reduces to

$$A_{(t)} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ a & b & d & 0 & 2 & 0 \\ b & c & e & -2 & 0 & 0 \\ d & e & f & 0 & 0 & 0 \end{bmatrix} \tag{2.6.2}$$

with the following definitions:

$$a = -\frac{\partial^2 U}{\partial x^2} \quad b = -\frac{\partial^2 U}{\partial x \partial y} = -\frac{\partial^2 U}{\partial y \partial x} \quad c = -\frac{\partial^2 U}{\partial y^2} \quad d = -\frac{\partial^2 U}{\partial x \partial z} = -\frac{\partial^2 U}{\partial z \partial x}$$
$$e = -\frac{\partial^2 U}{\partial y \partial z} = -\frac{\partial^2 U}{\partial z \partial y} \quad f = -\frac{\partial^2 U}{\partial z^2}.$$

Expanding these terms gives

$$a = \frac{3(1-\mu)(x+\mu)^2}{((x+\mu)^2+y^2+z^2)^{\frac{5}{2}}} - \frac{1-\mu}{((x+\mu)^2+y^2+z^2)^{\frac{3}{2}}} + \frac{3\mu(x+\mu-1)^2}{((x+\mu-1)^2+y^2+z^2)^{\frac{5}{2}}}$$
$$- \frac{\mu}{((x+\mu-1)^2+y^2+z^2)^{\frac{3}{2}}} + 1$$

$$b = \frac{3(1-\mu)(x+\mu)y}{((x+\mu)^2+y^2+z^2)^{\frac{5}{2}}} + \frac{3\mu y(x+\mu-1)}{((x+\mu-1)^2+y^2+z^2)^{\frac{5}{2}}} \tag{2.6.3}$$

$$c = \frac{3(1-\mu)y^2}{((x+\mu)^2+y^2+z^2)^{\frac{5}{2}}} - \frac{1-\mu}{((x+\mu)^2+y^2+z^2)^{\frac{3}{2}}} + \frac{3\mu y^2}{((x+\mu-1)^2+y^2+z^2)^{\frac{5}{2}}}$$
$$- \frac{\mu}{((x+\mu-1)^2+y^2+z^2)^{\frac{3}{2}}} + 1$$

$$d = \frac{3(1-\mu)(x+\mu)z}{((x+\mu)^2+y^2+z^2)^{\frac{5}{2}}} + \frac{3\mu z(x+\mu-1)}{((x+\mu-1)^2+y^2+z^2)^{\frac{5}{2}}} \tag{2.6.4}$$

$$e = \frac{3\left(1-\mu\right)yz}{\left((x+\mu)^2+y^2+z^2\right)^{\frac{5}{2}}} + \frac{3\mu yz}{\left((x+\mu-1)^2+y^2+z^2\right)^{\frac{5}{2}}} \tag{2.6.5}$$

$$f = \frac{3\left(1-\mu\right)z^2}{\left((x+\mu)^2+y^2+z^2\right)^{\frac{5}{2}}} - \frac{1-\mu}{\left((x+\mu)^2+y^2+z^2\right)^{\frac{3}{2}}} + \frac{3\mu z^2}{\left((x+\mu-1)^2+y^2+z^2\right)^{\frac{5}{2}}} - \frac{\mu}{\left((x+\mu-1)^2+y^2+z^2\right)^{\frac{3}{2}}}.$$

Overall, it is possible to write the nonautonomous linearized equations of motion (in the vicinity of the initial state) as

$$\dot{\mathbf{X}} = A(t)\boldsymbol{X}. \tag{2.6.6}$$

Of course the Jacobian matrix, $A(t)$, becomes constant at, and very near, a Lagrange point. This is because the velocity of the Lagrange point is zero and its position is fixed in the CR3BP reference frame. This implies that the nonautonomous linearized approximation of Equation 2.6.6 simplifies to the autonomous form of

$$\dot{\boldsymbol{X}} = A\boldsymbol{X} \tag{2.6.7}$$

at any of the five Lagrange points of the CR3BP (note that $\dot{\boldsymbol{X}} = \mathbf{0}$ is the location of the Lagrange point in this autonomous Linearization). Fortunately, it is well known that the solution to Equation 2.6.7 is

$$\boldsymbol{X} = \boldsymbol{X}_0 e^{At}. \tag{2.6.8}$$

From this solution it becomes possible to characterize the stability of a particle placed at a Lagrange point with zero relative velocity. Note that Equation

2.6.8 can also be expressed as the summation of the eigenvalues of A. If the real part of all eigenvalues of A is negative then the solution will exponentially decay back to the Lagrange point as time progresses. This is called "stable" since any minor disturbance away from a Lagrange point will be dampened out and return the particle back to the original Lagrange point. Conversely, if the real part of any eigenvalue is positive, then the solution will tend toward infinity as time progresses. This condition is called "unstable" since a minor disturbance will cause a particle to exponentially depart the vicinity of the original Lagrange point. Finally, a "neutrally stable" point will exist if all eigenvalues are identically zero. This will cause a closed orbit to form about the Lagrange point.

Unfortunately, due to the complexity of the problem, the only way to continue the study of the stability of collinear Lagrange points is through direct numerical computation with a given value of μ. Upon numerical computation, summarized in Table 2.2, one finds that for the Earth-Moon system with $\mu = 0.01215$, all three collinear Lagrange points are unstable because at least one eigenvalue is real and positive. Also note that all three collinear points are saddle points as they are neutrally stable in two directions and unstable in a third.

Table 2.2: Top: Coefficients of A matrix for collinear Lagrange points in the Earth-Moon system. Bottom: Eigenvalues of A matrix for colinear Lagrange points in the Earth-Moon system.

	a	b	c	d	e	f
L_1	11.295	0	-4.147	0	0	-5.147
L_2	7.381	0	-2.190	0	0	-3.190
L_3	3.021	0	-0.011	0	0	-1.011

	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
L_1	2.932	-2.932	$2.334i$	$-2.334i$	$2.268i$	$-2.268i$
L_2	2.158	-2.158	$1.862i$	$-1.862i$	$1.786i$	$-1.786i$
L_3	0.178	-0.178	$1.010i$	$-1.010i$	$1.005i$	$-1.005i$

Similarly, the stability of the collinear Lagrange points in the Sun-Earth system ($\mu \approxeq 3 \times 10^{-6}$) are also all unstable saddle points as evidenced by the λ_1 eigenvalues. Figure 2.4 illustrates the stability of each collinear Lagrange point as a function of μ. Note that all collinear points always have one unstable eigenmode regardless of the value of μ. This supports the previous results from the Earth-Moon and Sun-Earth cases.

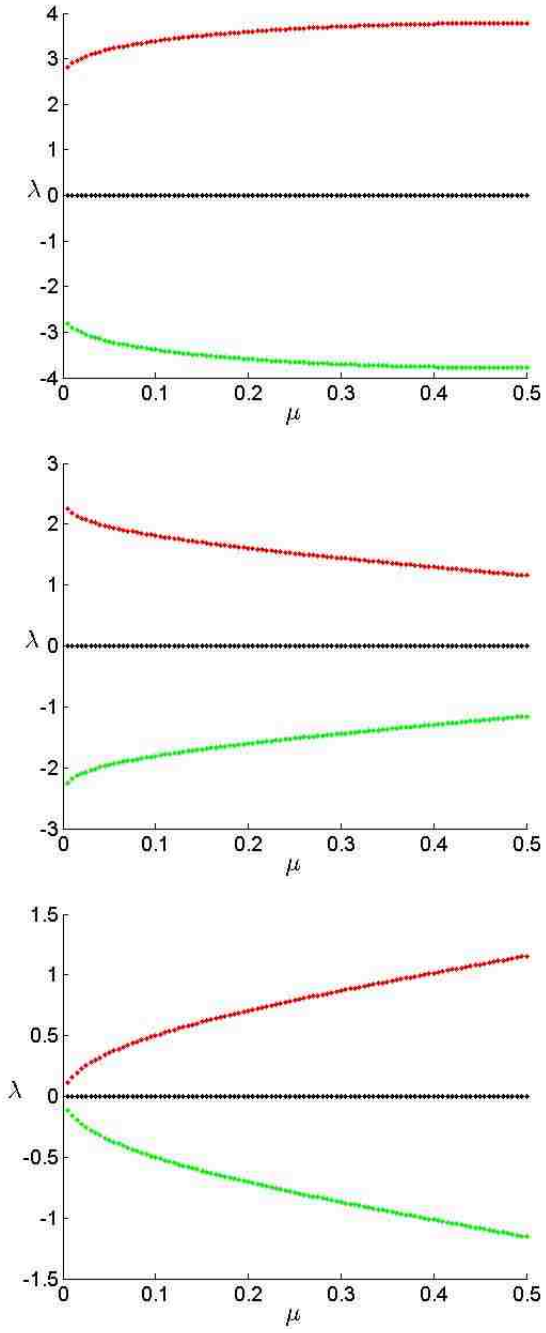


Figure 2.4: *Top* (L_1), *middle* (L_2), and *bottom* (L_3) plots of the real part of the eigenvalues of A vs. μ. All six eigenvalues are plotted for each case (four are on the $x-$axis). Note that the positive values of λ make this system unstable.

2.7 State Transition Matrix

When undertaking numerical calculations it is often necessary to use gradient-based, differential correction methods. Many of these methods, used in chapters 3, 4, and 5 rely on the State Transition Matrix (STM). The STM relates the sensitivity of the final state $\boldsymbol{X}(\boldsymbol{X}_0, t)$ (obtained via integration of Equation 2.2.27) to the initial state $\boldsymbol{X}_0$. In other words, the STM$\equiv \Phi(t, t_0) = \frac{\partial \boldsymbol{X}(\boldsymbol{X}_0, t)}{\partial \boldsymbol{X_o}}$ or

$$\Phi(t, t_0) = \begin{bmatrix} \frac{\partial x}{\partial x_0} & \frac{\partial x}{\partial y_0} & \frac{\partial x}{\partial z_0} & \frac{\partial x}{\partial \dot{x}_0} & \frac{\partial x}{\partial \dot{y}_0} & \frac{\partial x}{\partial \dot{z}_0} \\ \frac{\partial y}{\partial x_0} & \frac{\partial y}{\partial y_0} & \frac{\partial y}{\partial z_0} & \frac{\partial y}{\partial \dot{x}_0} & \frac{\partial y}{\partial \dot{y}_0} & \frac{\partial y}{\partial \dot{z}_0} \\ \frac{\partial z}{\partial x_0} & \frac{\partial z}{\partial y_0} & \frac{\partial z}{\partial z_0} & \frac{\partial z}{\partial \dot{x}_0} & \frac{\partial z}{\partial \dot{y}_0} & \frac{\partial z}{\partial \dot{z}_0} \\ \frac{\partial \dot{x}}{\partial x_0} & \frac{\partial \dot{x}}{\partial y_0} & \frac{\partial \dot{x}}{\partial z_0} & \frac{\partial \dot{x}}{\partial \dot{x}_0} & \frac{\partial \dot{x}}{\partial \dot{y}_0} & \frac{\partial \dot{x}}{\partial \dot{z}_0} \\ \frac{\partial \dot{y}}{\partial x_0} & \frac{\partial \dot{y}}{\partial y_0} & \frac{\partial \dot{y}}{\partial z_0} & \frac{\partial \dot{y}}{\partial \dot{x}_0} & \frac{\partial \dot{y}}{\partial \dot{y}_0} & \frac{\partial \dot{y}}{\partial \dot{z}_0} \\ \frac{\partial \dot{z}}{\partial x_0} & \frac{\partial \dot{z}}{\partial y_0} & \frac{\partial \dot{z}}{\partial z_0} & \frac{\partial \dot{z}}{\partial \dot{x}_0} & \frac{\partial \dot{z}}{\partial \dot{y}_0} & \frac{\partial \dot{z}}{\partial \dot{z}_0} \end{bmatrix}. \tag{2.7.1}$$

This can be demonstrated in the following way. Suppose two trajectories exist: trajectory $\boldsymbol{X}$, and its variation, trajectory $\boldsymbol{Y}$ (i.e. $\boldsymbol{Y}$ is in the neighborhood of $\boldsymbol{X}$). According to Vallado [39], one can express the derivatives and initial conditions as

$$\begin{aligned} \boldsymbol{X}(t_0) &= \boldsymbol{X}_0, \quad \dot{\boldsymbol{X}} = \vec{f}(\boldsymbol{X}) \\ \boldsymbol{Y}(t_0) &= \boldsymbol{Y}_0, \quad \dot{\boldsymbol{Y}} = \vec{f}(\boldsymbol{Y}) \end{aligned} \tag{2.7.2}$$

where f is defined in Equation 2.6.1 as the state vector expression of the equations of motion of the CR3BP. The difference in these two trajectories is small and can be expressed as

$$\boldsymbol{Y} = \boldsymbol{X} + \delta\boldsymbol{X}. \tag{2.7.3}$$

As with Equation 2.7.2 one can write $\dot{\boldsymbol{Y}}$ as

$$\dot{\boldsymbol{Y}} = \vec{f}(\boldsymbol{X} + \delta\boldsymbol{X}) \tag{2.7.4}$$

based on Equation 2.7.3. This can be expanded using a Taylor series centered about $\boldsymbol{X}$, using Equation 2.7.3

$$\dot{\boldsymbol{Y}} = \vec{f}(\boldsymbol{X}) + \frac{\partial\vec{f}(\boldsymbol{X})}{1!\partial\boldsymbol{X}}\delta\boldsymbol{X} + \frac{\partial\vec{f}(\boldsymbol{X})^2}{2!\partial^2\boldsymbol{X}}\delta\boldsymbol{X}^2 + ... \tag{2.7.5}$$

Note that $\dot{\boldsymbol{Y}} = \dot{\boldsymbol{X}} + \delta\dot{\boldsymbol{X}}$ via a direct differentiation of Equation 2.7.3 and when substituted into Equation 2.7.5 yields

$$\delta\dot{\boldsymbol{X}} = \frac{\partial\vec{f}(\boldsymbol{X})}{\partial\boldsymbol{X}}\delta\boldsymbol{X} + H.O.T. = A(t)\,\delta\boldsymbol{X} + H.O.T. \tag{2.7.6}$$

where $H.O.T.$ represents all the higher order terms of the expansion. If the $H.O.T.$ are ignored, one can say that the system has been "linearized" because the non-linear terms have been disregarded. This approximation is valid in regions where the dynamics are only weakly nonlinear, such as the vicinity of a Lagrange point. Assume that a solution to the differential Equation 2.7.6 has the form

$$\delta\boldsymbol{X} = \Phi(t,\, t_0)\,\delta\boldsymbol{X}_0. \tag{2.7.7}$$

If Equation 2.7.7 is inserted into Equation 2.7.6 the result is

$$\dot{\Phi}(t,\, t_0) = A(t)\,\Phi(t,\, t_0) \tag{2.7.8}$$

which can be numerically integrated from an initial condition to determine the STM at any moment in time. The initial condition is found by inserting in $t = t_0$ into Equation 2.7.7. When doing so, one will find that $\Phi(t_0,\, t_0) = I$.

The dynamics presented in this chapter have laid the groundwork for many of the subsequent studies presented in later chapters of this dissertation. This is especially true in the next chapter where traditional, gradient-based optimization methods are formulated, described, and demonstrated. These methods rely on the STM as well as the dynamics of the CR3BP and are used to quickly solve a variety of trajectory optimization problems using low-thrust or impulsive maneuvers. Examples of this type of optimization are illustrated in subsequent chapters of this dissertation.

3 Gradient-Based Optimization Techniques

This chapter focuses on the general optimization of two-point boundary value problems that are commonly found in astrodynamics and spaceflight mission planning. The scope of this chapter is limited to the traditional "gradient" based optimization approaches that are commonly applied to spacecraft trajectory optimization problems. A gradient-based algorithm utilizes information about the gradient of an objective function (or related derivatives or a subset of those derivatives) in order to solve it quickly and intelligently. In general, gradient-based optimization is considered to be a more desirable optimization method than other options (such as stochastic processes or evolutionary algorithms). This is because gradient-based methods use differential calculus to guide the solution from an initial guess (sometimes called a "seed") to a final, optimal solution. These algorithms converge very fast because they attempt to utilize the maximum amount of information about a problem. They are considered to be more "intelligent" than other methods and can be mathematically proven to locate the optimal solution given a set of boundaries [36].

Unfortunately, they also have a number of disadvantages that make them undesirable for certain applications. First of all, gradient-based optimization

can converge on a local, rather than the global, minima of an objective function if a poor guess is supplied or the objective function if particularly onerous. Worse yet are situations where the gradient of an objective function is undefined. Such situations arise from discontinuities, abrupt changes in curvature, and singularities. In general, if an objective function has multiple minima and poorly behaved gradients, then it is not a good candidate for gradient-based methods, because they will either converge on a local minima, or will fail to converge altogether. Fortunately, other optimization methods, such as Particle Swarm Optimization (Chapter 6), can be used in these situations.

This chapter begins with a short discussion on free variables and constraint vectors. With the knowledge of these vectors, it is possible to outline single shooting and multiple shooting gradient-based techniques. Various continuation methods are introduced that are capable of constructing a new and different solution from an old solution that was found using a different method. Finally, a discussion of a full-ephemeris integrator is given and is followed up by an introduction to the SPICE/Mice package used in implementing this integrator. Concrete examples of each section of this chapter can be found in corresponding sections of Chapter 4.

3.1 Generalized Free Variable & Constraint Vectors

When using a differential corrections algorithm to solve a two-point boundary value problem (TPBVP), one can use free variable and constraint vectors based

on Newton's method [40, 41]. The free variable vector

$$\boldsymbol{X} = \begin{bmatrix} x_1 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} \tag{3.1.1}$$

consists of n elements that are allowed to freely and independently change in value during the course of a differential corrections procedure. Typically, these elements consist of the state vector, integration times, T_i, and slack variables. The slack variable is introduced to transform an inequality constraint into an equality constraint. The system is also subject to m constraint equations. These constraint equations form the Constraint Vector

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{bmatrix} F_1(\boldsymbol{X}) \\ \vdots \\ \vdots \\ F_m(\boldsymbol{X}) \end{bmatrix} = 0 \tag{3.1.2}$$

and must be written in such a way that they satisfy the equation $\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{0}$.

The goal is to find a solution, $\boldsymbol{X}^*$, which satisfies the equation $\boldsymbol{F}(\boldsymbol{X}^*) = \boldsymbol{0}$ given an acceptable error tolerance, ε. In order to accomplish this task, one may begin with an initial free variable vector $\boldsymbol{X^0}$. One can express $\boldsymbol{F}(\boldsymbol{X})$ by using a Taylor Series centered about $\boldsymbol{X}^0$and dropping all higher order terms

$$\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{F}\left(\boldsymbol{X}^0\right) + D\boldsymbol{F}\left(\boldsymbol{X}^0\right)\left(\boldsymbol{X} - \boldsymbol{X}^0\right) \tag{3.1.3}$$

with the $m \times n$ Jacobian Matrix $D\boldsymbol{F}\left(\boldsymbol{X}^0\right)$expressed as

$$D\boldsymbol{F}\left(\boldsymbol{X}^0\right) = \frac{\partial \boldsymbol{F}\left(\boldsymbol{X}^0\right)}{\partial \boldsymbol{X}^0} = \left[\begin{array}{ccc} \frac{\partial F_1}{\partial x_1} & \cdot\cdot & \frac{\partial F_1}{\partial x_n} \\ \vdots & \cdot & \vdots \\ \vdots & \cdot & \vdots \\ \frac{\partial F_m}{\partial x_1} & \cdot\cdot & \frac{\partial F_m}{\partial x_n} \end{array}\right]_{\boldsymbol{X}=\boldsymbol{X}^{\mathbf{0}}} . \tag{3.1.4}$$

Next an iterative process will be introduced into Equation 3.1.3. Note that the substitution $\boldsymbol{F}(\boldsymbol{X}) = \mathbf{0}$ can be used to reduce Equation 3.1.3 to

$$\mathbf{0} = \boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right) + D\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right)\left(\boldsymbol{X}^{(j+1)} - \boldsymbol{X}^{(j)}\right) \tag{3.1.5}$$

where $\boldsymbol{X}^{(j)}$ represents the current iteration and $\boldsymbol{X}^{(j+1)}$ represents the next iteration. If the algorithm is convergent, then $\left\|\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right)\right\| > \left\|\boldsymbol{F}\left(\boldsymbol{X}^{(j+1)}\right)\right\|$ and will stop iterating once $\left\|\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right)\right\| < \varepsilon$.

There are two ways to solve for $\boldsymbol{X}^*$ depending on the situation. If $n = m$ then $D\boldsymbol{F}$ is square and invertible. In this situation, a multivariate version of Newton's Method is appropriate

$$\boldsymbol{X}^{(j+1)} = \boldsymbol{X}^{(j)} - D\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right)^{-1}\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right). \tag{3.1.6}$$

If, however, $n > m$, there exists an infinite number of solutions and $D\boldsymbol{F}$ cannot be inverted. In this circumstance, the minimum-norm solution is used

$$\boldsymbol{X}^{(j+1)} = \boldsymbol{X}^{(j)} - D\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right)^T\left[D\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right)D\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right)^T\right]^{-1}\boldsymbol{F}\left(\boldsymbol{X}^{(j)}\right). \tag{3.1.7}$$

This solution minimizes the difference between the current iteration and the

previous one. This is desirable because it typically finds the solution, $\boldsymbol{X}^*$, that is closest to the initial guess, $\boldsymbol{X}^0$, given the fact that there are an infinite number of solutions.

In summary, to solve a general optimization problem using free variable and constraint vectors one must

1. Define $\boldsymbol{X}$ and $\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{0}$,
2. Calculate the Jacobian Matrix, $D\boldsymbol{F}(\boldsymbol{X})$, and the associated partial derivatives therein,
3. Iteratively solve for $\boldsymbol{X}^*$ using either Equation 3.1.6 or Equation 3.1.7 depending on the relationship between n and m.

3.2 Variable-Time, Single Shooting

One application of free variables and constraint vectors is that of Variable-Time, Single Shooting [41]. In this application the position of the initial state of a trajectory is fixed but the initial velocity and Time of Flight, T, are free. Thus the free variable vector becomes

$$\boldsymbol{X} = \begin{bmatrix} \dot{x}_0 \\ \dot{y}_0 \\ \dot{z}_0 \\ T \end{bmatrix}. \tag{3.2.1}$$

The constraint vector will be written as

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{bmatrix} x_T - x_d \\ y_T - y_d \\ z_T - z_d \end{bmatrix} \tag{3.2.2}$$

with $\boldsymbol{X}_d$ representing the desired state at the end of time T. The Jacobian Matrix can be represented as

$$D\boldsymbol{F}(\boldsymbol{X}) = \frac{\partial \boldsymbol{F}(\boldsymbol{X})}{\partial \boldsymbol{X}} = \begin{bmatrix} \frac{\partial x_T}{\partial \dot{x}_0} & \frac{\partial x_T}{\partial \dot{y}_0} & \frac{\partial x_T}{\partial \dot{z}_0} & \frac{\partial x_T}{\partial T} \\ \frac{\partial y_T}{\partial \dot{x}_0} & \frac{\partial y_T}{\partial \dot{y}_0} & \frac{\partial y_T}{\partial \dot{z}_0} & \frac{\partial y_T}{\partial T} \\ \frac{\partial z_T}{\partial \dot{x}_0} & \frac{\partial z_T}{\partial \dot{y}_0} & \frac{\partial z_T}{\partial \dot{z}_0} & \frac{\partial z_T}{\partial T} \end{bmatrix} = \begin{bmatrix} \Phi_{1,4} & \Phi_{1,5} & \Phi_{1,6} & \dot{x}_T \\ \Phi_{2,4} & \Phi_{2,5} & \Phi_{2,6} & \dot{y}_T \\ \Phi_{3,4} & \Phi_{3,5} & \Phi_{3,6} & \dot{z}_T \end{bmatrix} \tag{3.2.3}$$

in accordance with Equation 2.7.1. Since the Jacobian is not a square matrix, the minimum-norm solution may be found using Equation 3.1.7. Figure 3.1 illustrates the iterative process that begins with an initial guess trajectory (in green) and iteratively refines that guess until the desired constraint conditions are satisfied (red trajectory with red $\times$).

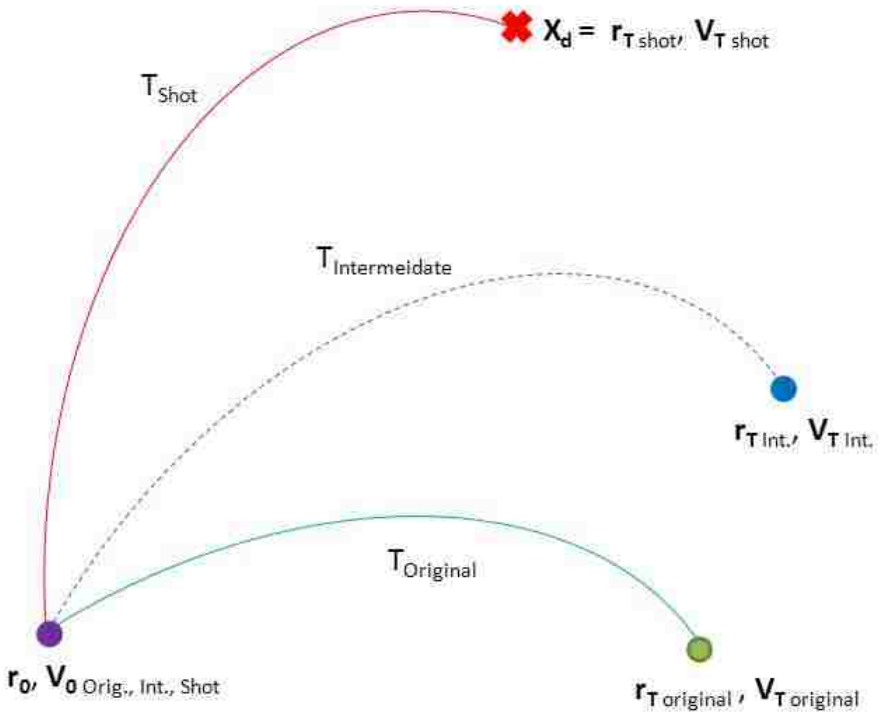


Figure 3.1: The Variable-Time, Single Shooting method in action.

3.3 Variable-Time, Multiple Shooting

Next the concept of Variable-Time, Multiple Shooting will be explored. This is a highly useful technique to use in finding sensitive trajectories, because a number of "way-points" can be used between the initial and final states [41]. Figure 3.2 shows a trajectory that is beginning to form using the Variable-Time, Multiple Shooting method. This trajectory is comprised of n segments

with each segment being numerically propagated from its initial state, $\boldsymbol{X}_n$, to its final state, $\boldsymbol{X}_{n+1\,T_n}$. A converged trajectory needs to be fully continuous along its entire path. It is therefore necessary for the final state of a segment to match the initial state of its succeeding segment. To begin, a free variable vector is defined as

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \\ \vdots \\ \boldsymbol{X}_n \\ T_1 \\ \vdots \\ T_{n-1} \end{bmatrix} \tag{3.3.1}$$

with $7n - 1$ components. A constraint vector is defined as

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{bmatrix} \boldsymbol{X}_{2,T_1} - \boldsymbol{X}_2 \\ \vdots \\ \boldsymbol{X}_{n,T_{n-1}} - \boldsymbol{X}_n \end{bmatrix} \tag{3.3.2}$$

which dictates the need for each new segment of the trajectory to start/stop with an identical state as its preceding/proceeding segment. The constraint vector has $6(n-1)$ components. The Jacobian can be expressed as:

$$
DF(\boldsymbol{X}) = \begin{bmatrix}
\Phi_{(t_2,t_1)} & -I_{6\times6} & 0_{6\times6} & \cdots & & & & & \cdots & 0_{6\times6} & \dot{x}_{2,\,T_{1\,(6\times1)}} & 0_{6\times1} & \cdots & 0_{6\times1} \\
0_{6\times6} & \Phi_{(t_3,t_2)} & -I_{6\times6} & 0_{6\times6} & \cdots & & & & \ddots & \cdots & 0_{6\times1} & \dot{x}_{3,\,T_{2\,(6\times1)}} & \cdots & \cdots \\
\cdots & & 0_{6\times6} & \Phi_{(t_4,t_3)} & -I_{6\times6} & 0_{6\times6} & \cdots & & & \cdots & \cdots & 0_{6\times1} & \cdots & \\
 & & \cdots & 0_{6\times6} & \ddots & \ddots & & & & \cdots & \cdots & \cdots & \cdots & \\
 & & & \cdots & & & & & & \cdots & \cdots & & \cdots & \\
 & & & & & & & & \ddots & 0_{6\times6} & \cdots & & & \\
 & & & & & & & & \ddots & -I_{6\times6} & \cdots & & \cdots & \\
 & & & & & & & \Phi_{(t_{n-1}t_{n-2})} & & -I_{6\times6} & \cdots & & 0_{6\times1} & \\
0_{6\times6} & 0_{6\times6} & & & & & & & & 0_{6\times6} & \cdots & 0_{6\times1} & \cdots & \dot{x}_{n,\,T_{n-1\,(6\times1)}} \\
0_{6\times6} & 0_{6\times6} & \cdots & & & & & \cdots & & 0_{6\times6} & \Phi_{(t_n,t_{n-1})} & 0_{6\times1} & \cdots &
\end{bmatrix} \tag{3.3.3}
$$

which is a $6\,(n-1)\ \times\ 7n-1$ matrix. This method is generalized to account for n way-points that flow together using a continuous trajectory with a total time of $T = \sum_{i=1}^{n} T_i$.

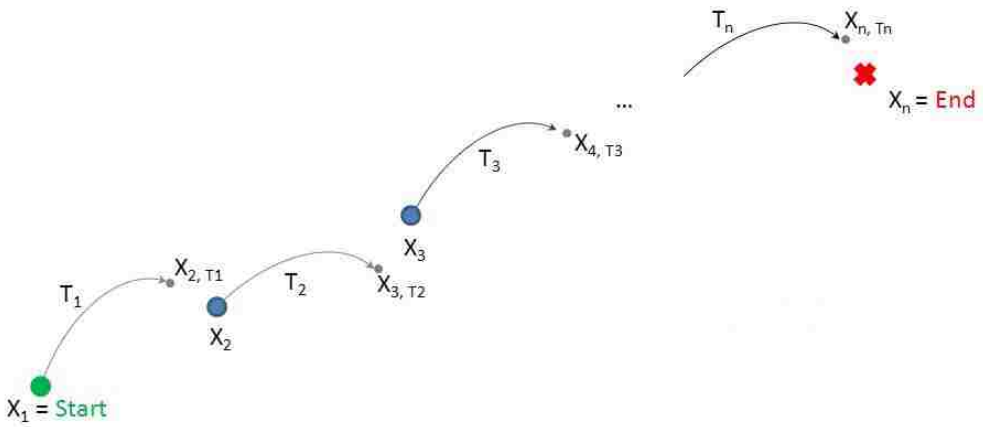


Figure 3.2: The Variable-Time, Multiple Shooting in action.

3.4 Single Parameter Continuation

The underlying concept behind Single-Parameter Continuation is relatively straightforward. Assuming a (desired) baseline trajectory can be found using a standard optimization method, such as those outlined above, it should also be possible to use the same techniques to find related solutions. This can be accomplished by using the same initial state as was used with the baseline

solution but with a small change in one of the parameters. This new initial state will then be run through the given optimization method until a new solution is reached. This new solution will differ from the baseline trajectory because of the small change in one of the parameters of the initial state. This process can be repeated numerous times to build up a "family" of trajectories that are all based on the "baseline" trajectory and a continuation of a parameter of interest. Traditionally, parameters that are changed can involve position, velocity, energy, period, and so on. Normally, only one parameter is used in this continuation scheme.

3.5 Pseudo-Arclength Continuation

In contrast to Single Parameter Continuation, Pseudo-Arclength Continuation can vary multiple parameters during one iteration. The way in which these parameters are altered cause the solution to move in a direction that is tangent to the baseline solution by a user defined amount $\triangle s$. In this way, the continuation scheme moves away from the current family of trajectories in search for new families of trajectories [41]. Let $\boldsymbol{X}_{i-1}^{*}$ be a (converged) baseline trajectory that satisfies the equation $\mathbf{F}\left(\boldsymbol{X}_{i-1}^{*}\right) = \mathbf{0}$. The goal is to find the next member of the continuation scheme, $\boldsymbol{X}_{i}$, which often defines a new family of trajectories. The null vector of the Jacobian Matrix, $D\mathbf{F}\left(\boldsymbol{X}_{i-1}^{*}\right)$, is used to generate an orthonormal null vector, $\triangle\boldsymbol{X}_{i-1}^{*}$, that is tangent to the family at $\boldsymbol{X}_{i-1}^{*}$. To ensure that the new member, $\boldsymbol{X}_{i}$, is found by moving an amount $\triangle s$ in the tangent direction, a pseudo-arclength constraint is appended to the existing constraint vector, $\boldsymbol{F}\left(\boldsymbol{X}_{i}\right)$. The new, augmented, constraint vector is

written as

$$\boldsymbol{G}\left(\boldsymbol{X}_i\right) = \begin{bmatrix} \boldsymbol{F}\left(\boldsymbol{X}_i\right) \\ \left(\boldsymbol{X}_i - \boldsymbol{X}^*_{i-1}\right)^T \triangle \boldsymbol{X}^*_{i-1} - \triangle s \end{bmatrix} = \boldsymbol{0} \tag{3.5.1}$$

with the augmented Jacobian Matrix is written as

$$D\boldsymbol{G}\left(\boldsymbol{X}_i\right) = \frac{\partial \boldsymbol{G}\left(\boldsymbol{X}_i\right)}{\partial \boldsymbol{X}_i} = \begin{bmatrix} D\boldsymbol{F}\left(\boldsymbol{X}_i\right) \\ \left(\triangle \boldsymbol{X}^*_{i-1}\right)^T \end{bmatrix}. \tag{3.5.2}$$

These equations can be solved using either Newton's Method or the minimum norm Equation 3.1.7. This process can be repeated numerous times to generate a large number of trajectory families.

3.6 Full Ephemeris

While the equations of motion in the CR3BP are a useful starting point for accurate trajectory propagation, they are not accurate enough for high-fidelity modeling. Perturbations from other gravitational bodies (i.e. Sun, Venus, Jupiter, etc.) can cause significant changes in the trajectory of a satellite in High Earth Orbit (HEO). These perturbations become increasingly important in the vicinity of Lagrange points since much of the three-body effects cancel and the resultant force is astonishingly small. In order to accurately account for the gravitational perturbations of extra bodies, the exact position of each body with respect to a reference point must be known at a given epoch. Once this information is known, Equation 3.6.1, the $N-$body equation, can be used to determine the acceleration on a spacecraft (sc) from N gravitational bodies

[39].

$$\ddot{\boldsymbol{r}}_{1,\,sc} = -\frac{G\left(m_1 + m_{sc}\right)}{r_{1,\,sc}^3}\boldsymbol{r}_{1,\,sat} + G\sum_{j=3}^{N} m_j \left(\frac{\boldsymbol{r}_{sc,\,j}}{r_{sc,\,j}^3} - \frac{\boldsymbol{r}_{1,\,j}}{r_{1,\,j}^3}\right) \tag{3.6.1}$$

3.7 SPICE/Mice

If a full ephemeris force model is to be employed, the exact positions of all major gravitational objects in the Solar System must be known at a given epoch. While it is possible to use Keplerian models to derive algebraic approximations of the positions of these bodies, it is much more accurate to glean this information from direct astronomical observation. This is where the capabilities of the SPICE kernel produced by NASA's Navigation and Ancillary Information Facility (NAIF) becomes paramount [42]. The primary goal of SPICE (Spacecraft, Planet, Instrument, C-Matrix "pointing", and Events kernel) is to define, develop, and utilize software standards/protocols that can store data gathered from spacecraft missions from any agency, nation, or organization in a uniform way. One of the key functions of SPICE is its ability to quickly retrieve ephemeris data (i.e. position and velocity relative to a coordinate system at a given moment in time) from any time period between the 1970's and the 2050's. All data is based on astronomical observation, spacecraft reconnaissance, or advanced mathematical modeling, where appropriate. This data represents some of the most accurate ephemeris data available to the public and is used as the standard in many software applications [43]. This SPICE kernel is supported by four major programing languages: C, FORTRAN, IDL, and MATLAB. The MATLAB version, called "Mice," is the primary language used for this dissertation work.

The theory of gradient-based optimization presented in this chapter is utilized throughout the remainder of this dissertation. It is relied upon heavily in Chapter 4 to successfully create a Lagrange point orbit from a set of initial conditions. Both single and multiple shooting are used as well as continuation techniques which are capable of identifying multiple LPO families from a single initial guess. Additionally, gradient based optimization can be hybridized with non-gradient optimization, as seen in Chapter 9 of this dissertation. This technique was used to optimize a two-burn transfer to a particular LPO of interest.

4 Gradient-Based Optimization of Lagrange Point Orbits

According to a linear stability analysis of Lagrange points (see Chapter 2) in the Earth-Moon three-body system, there exists at least one or more center manifolds. These center manifolds permit the existence of periodic orbits about a Lagrange point or in the vicinity of a Lagrange point. This chapter focuses on the construction of such Lagrange point orbits using techniques of gradient-based optimization described in Chapter 3. Note that the use of an optimization technique to discover and describe a particular Lagrange point orbit is very common in present day research [41, 44, 45, 46]. The only alternative to optimization techniques are analytic approximations of Lagrange point orbits as discussed in Szebehely [2] and elsewhere. While these approximations make excellent initial guess solutions for optimization algorithms, they cannot replace these solutions due to an intrinsic lack of fidelity of the medium and long-term dynamics.

This chapter begins by discussing the construction of a Lyapunov orbit about the Earth-Moon, L_1 point via the application of the single shooting algorithm introduced in Chapter 3. Next, this Lyapunov orbit can be utilized as a seed orbit to generate an entire family of Lyapunov orbits about the L_1 point using

pseudo-arclength continuation. The application of a multiple shooting algorithm is discussed and applied to construct a northern halo orbit about L_1. Finally, the chapter concludes with a demonstration of these techniques under a full-ephemeris dynamics model instead of the CR3BP.

4.1 Variable Time, Single Shooting Program

The goal of the Variable Time, Single Shooting program is to compute trajectories that are closed and periodic about the Earth-Moon L_1 point. This program will take advantage of the symmetry in the dynamics found across the $x-z$ plane and restrict the motion of the spacecraft to the $x-y$ plane for simplicity. The $x-z$ plane symmetry will ensure that any trajectory that begins on the $x-$axis and then crosses the $x-$axis with a velocity vector tangent to the $x-$axis (and still within the $xy-$plane), must produce a new trajectory that is a reflection of the original about the $x-$axis [47]. This is diagrammed in Figure 4.1.

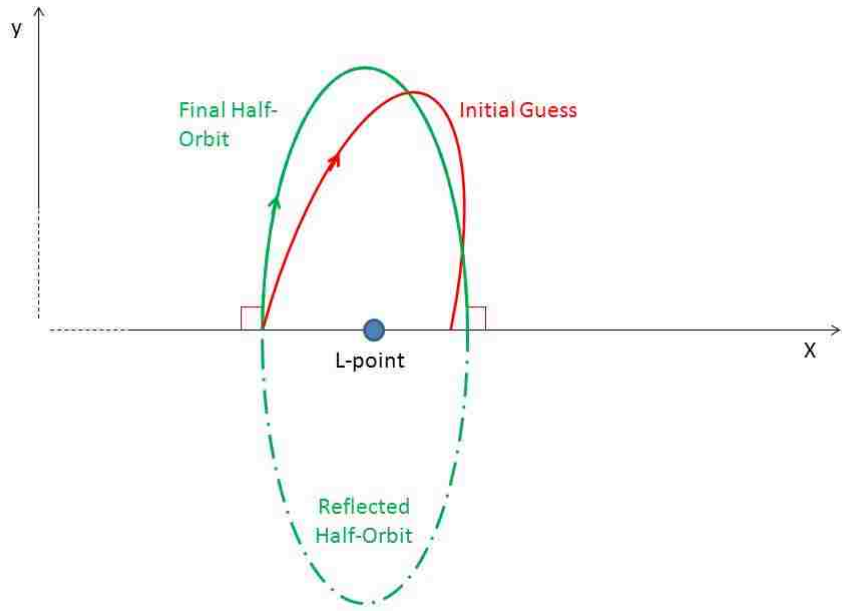


Figure 4.1: Reflection of the converged trajectory across the x-axis.

This program uses a Free Variable Vector of the form

$$\boldsymbol{X} = \begin{bmatrix} x_0 \\ \dot{y}_0 \\ \frac{1}{2}T \\ \beta \end{bmatrix} \tag{4.1.1}$$

Note that the first two entries of Equation 4.1.1 are the initial x-position and initial y-velocity (assumed to be positive). All other components of the state vector are assumed to be zero. This implies that the initial position is located somewhere on the x-axis. The third component of the Free Variable Vector is the half-period of the orbit. This indicates that the intention of the shooting algorithm is to find a half-orbit rather than a full orbit. Finally, if the initial

velocity is positive, then the velocity at the end of a half-period must be negative (or else there would not be a closed orbit). This implies a constraint condition of $\dot{y}\left(\frac{1}{2}T\right) < 0$. Unfortunately, this constraint condition cannot be written into the constraint vector in its present form. Instead the condition must be modified to

$$\dot{y}\left(\frac{1}{2}T\right) + \beta^2 = 0 \tag{4.1.2}$$

with β being real number known as the "slack variable." Since β is real, the Equation 4.1.2 can be expressed within the Constraint Vector as

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{bmatrix} y\left(\frac{1}{2}T\right) \\ \dot{x}\left(\frac{1}{2}T\right) \\ \dot{y}\left(\frac{1}{2}T\right) + \beta^2 \end{bmatrix} = \mathbf{0} \tag{4.1.3}$$

and the Jacobian Matrix as

$$\begin{aligned} D\boldsymbol{F}(\boldsymbol{X}) &= \frac{\partial \boldsymbol{F}(\boldsymbol{X})}{\partial \mathbf{X}} \\ &= \begin{bmatrix} \frac{\partial y\left(\frac{1}{2}T\right)}{\partial x_0} & \frac{\partial y\left(\frac{1}{2}T\right)}{\partial \dot{y}_0} & \frac{\partial y\left(\frac{1}{2}T\right)}{\partial T} & \frac{\partial y\left(\frac{1}{2}T\right)}{\partial \beta} \\ \frac{\partial \dot{x}\left(\frac{1}{2}T\right)}{\partial x_0} & \frac{\partial \dot{x}\left(\frac{1}{2}T\right)}{\partial \dot{y}_0} & \frac{\partial \dot{x}\left(\frac{1}{2}T\right)}{\partial T} & \frac{\partial \dot{x}\left(\frac{1}{2}T\right)}{\partial \beta} \\ \frac{\partial\left(\dot{y}\left(\frac{1}{2}T\right)+\beta^2\right)}{\partial x_0} & \frac{\partial\left(\dot{y}\left(\frac{1}{2}T\right)+\beta^2\right)}{\partial \dot{y}_0} & \frac{\partial\left(\dot{y}\left(\frac{1}{2}T\right)+\beta^2\right)}{\partial T} & \frac{\partial\left(\dot{y}\left(\frac{1}{2}T\right)+\beta^2\right)}{\partial \beta} \end{bmatrix} \\ &= \begin{bmatrix} \Phi_{2,1} & \Phi_{2,5} & \dot{y}\left(\frac{1}{2}T\right) & 0 \\ \Phi_{4,1} & \Phi_{4,5} & \ddot{x}\left(\frac{1}{2}T\right) & 0 \\ \Phi_{5,1} & \Phi_{5,5} & \ddot{y}\left(\frac{1}{2}T\right) & 2\beta \end{bmatrix} \end{aligned} \tag{4.1.4}$$

where $\Phi_{i,j}$ represents the i^{th} row and j^{th} column of the STM, $\Phi\left(\frac{1}{2}T, 0\right)$. Note that the STM is numerically propagated from time 0 to time $\frac{1}{2}T$ in order to populate the first six elements of the Jacobian Matrix. Three of the remaining

elements are populated by evaluating the equations of motion at the final state of the trajectory.

As an example, the program was seeded with an initial state taken from tabulated values in [2]. The initial state was

$$\boldsymbol{X}_{0\,guess} = \begin{bmatrix} 0.78 \\ 0.000 \\ 0.000 \\ 0.000 \\ 0.443 \\ 0.000 \end{bmatrix} \tag{4.1.5}$$

with an initial $T_{guess} = 3.9$. All values are in the non-dimensional units of the CR3BP. By exploring the tradespace through many trial-and-error attempts, the initial guess of the slack variable was set to $\beta = 0.7$. The Variable-Time, Single Shooting algorithm ran for just over one second and used 10 iterations to produce a closed, periodic orbit about the Earth-Moon L_1 point with a tolerance of 10^{-12}. The integration was performed on a 3GHz Intel Core 2 Duo processor using MATLAB's ODE113 function. The orbit can be seen in Figure 4.2

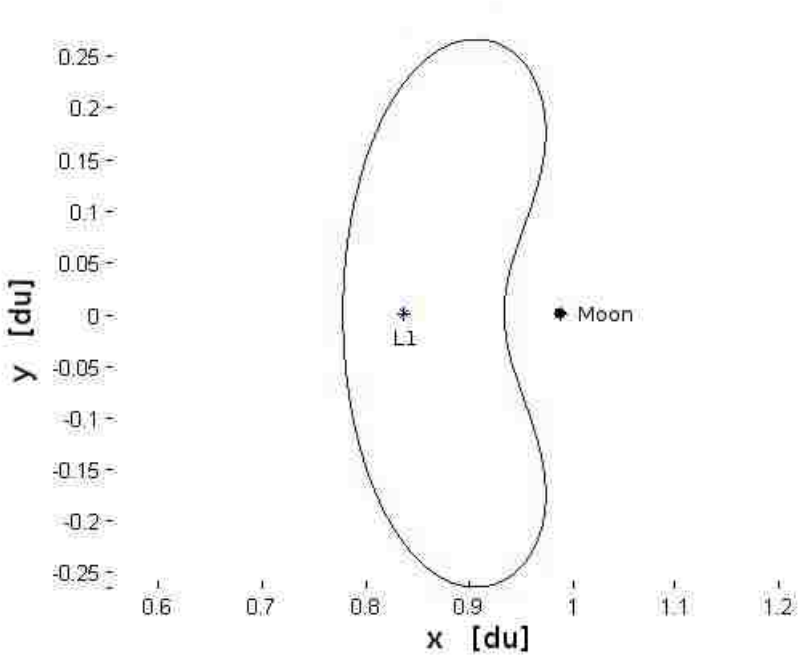


Figure 4.2: Closed Earth-Moon L_1 orbit generated by the Variable-Time, Single Shooting method.

The algorithm found the desired initial state to be

$$\boldsymbol{X}_{0\,shot} = \begin{bmatrix} 0.777910486548393 \\ 0.0 \\ 0.0 \\ 0.0 \\ 0.455080899040143 \\ 0.0 \end{bmatrix} \tag{4.1.6}$$

with an orbital period of $T = 4.063544575624369$. Note the classic "kidney-bean" shape of Figure 4.2 with curvature that is especially evident in the vicinity near the Moon (shown to the right of the orbit). Also note the 15 significant figures associated with each value reported (if the value is exactly zero, only

two digits are reported for sake of simplicity). These 15 significant figures correspond to the machine precision of MATLAB's 64 bit, double-precision, floating-point arithmetic. All values reported in this dissertation should be assumed to be double-precision. This degree of precision is needed because the equations of motion are numerically integrated. As the integration time increases, so does the integration error, which is primarily due to limitations in machine precision.

4.2 Pseudo-Arclength Continuation & Single Shooting

The orbit found in the previous section by method of Single Shooting will now be used as a "seed trajectory" for a Pseudo-Arclength Continuation program. In this program the Free Variable Vector,$\boldsymbol{X}$, was unchanged and

$$\begin{aligned} \boldsymbol{G}(\boldsymbol{X}) &= \begin{bmatrix} F(\boldsymbol{X}_i) \\ \left(\boldsymbol{X}_i - \boldsymbol{X}^*_{i-1}\right)^T \triangle \boldsymbol{X}^*_{i-1} - \triangle s \end{bmatrix} \\ &= \begin{bmatrix} y\left(\frac{1}{2}T\right) \\ \dot{x}\left(\frac{1}{2}T\right) \\ \dot{y}\left(\frac{1}{2}T\right) + \beta^2 \\ \left(\boldsymbol{X}_i - \boldsymbol{X}^*_{i-1}\right)^T \triangle \boldsymbol{X}^*_{i-1} - \triangle s \end{bmatrix} = 0 \end{aligned} \tag{4.2.1}$$

with

$$D\boldsymbol{G}(\boldsymbol{X}_i) = \frac{\partial \boldsymbol{G}(\boldsymbol{X}_i)}{\partial \boldsymbol{X}_i} = \begin{bmatrix} D\boldsymbol{F}(\boldsymbol{X}_i) \\ \left(\triangle \boldsymbol{X}^*_{i-1}\right)^T \end{bmatrix}. \tag{4.2.2}$$

The baseline trajectory, $\boldsymbol{X}_1^*$, is taken from the solution given by the Single Shooting algorithm from the previous section. Note that the orthonormal null vector $\triangle \boldsymbol{X}_1^*$ was also calculated from the previous trajectory by using the MATLAB code "$deltaXstar = null\,(DF)$". All subsequent null vectors are calculated using the same command. Based on trial-and-error experience, the value of $\Delta s = 0.012$ and was constant throughout the continuation process. The code generated 100 orbits and ran for a total of 23 seconds. Every 10^{th} orbit is plotted in Figure 4.3.

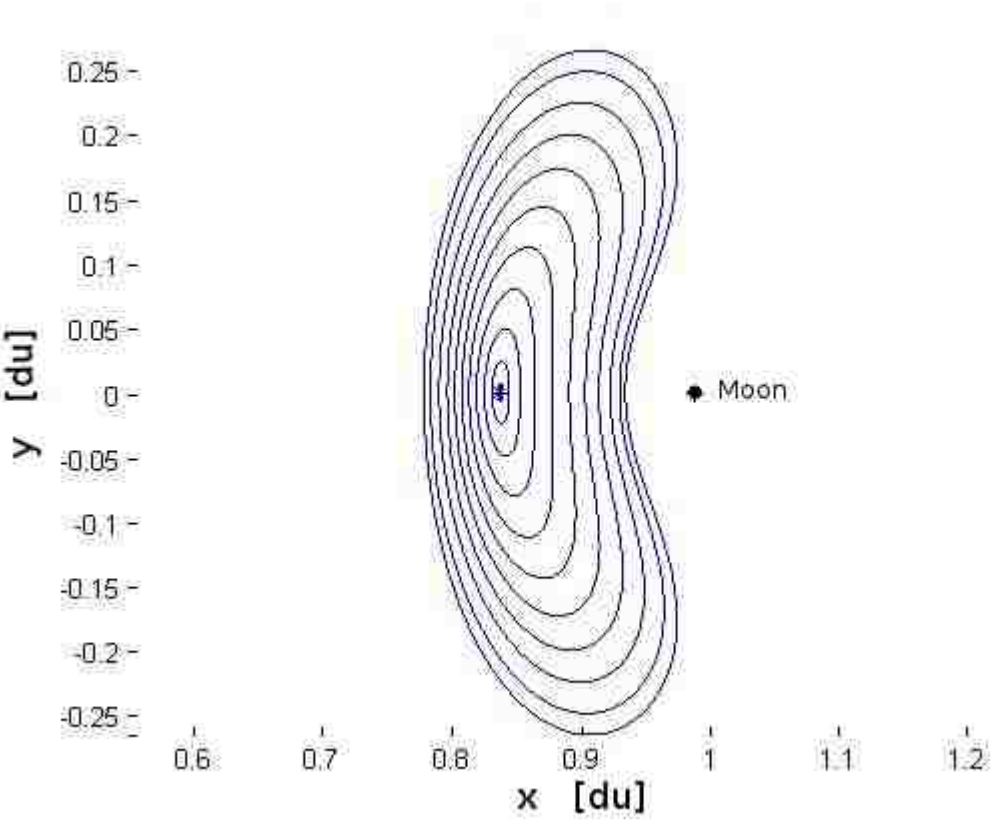


Figure 4.3: Orbits automatically found using Pseudo-Arclength Continuation (outside orbit is the initial orbit).

4.3 Variable Time, Multiple Shooting Program

In this program, the Multiple Shooting technique is used to find a trajectory that converges into periodic motion about the Earth-Moon L_1 point. The algorithm was tested using a known periodic solution which can be found in Table 4.1.

Table 4.1: Ten known patch points.

	x	y	z	$\dot{x}$	$\dot{y}$	$\dot{z}$
$\boldsymbol{X}_1$	0.781600000000000	0.000000000000000	0.0...	0.000000000000000	0.443081213954370	0.0...
$\boldsymbol{X}_2$	0.792032292050977	0.101076863157104	0.0...	0.085186531444790	0.395828151489133	0.0...
$\boldsymbol{X}_3$	0.911369923345768	0.252455690570353	0.0...	0.167745047434767	-0.020992012461727	0.0...
$\boldsymbol{X}_4$	0.955433598468470	0.106151039746667	0.0...	-0.112048667589987	-0.372540477511669	0.0...
$\boldsymbol{X}_5$	0.930684549931070	0.006511131265643	0.0...	-0.024803940496956	-0.601702268361602	0.0...
$\boldsymbol{X}_6$	0.946269109073130	-0.077568038927116	0.0...	0.142596656908626	-0.436462539017418	0.0...
$\boldsymbol{X}_7$	0.965129475968441	-0.148829574963555	0.0...	0.039514208529445	-0.301012106276146	0.0...
$\boldsymbol{X}_8$	0.923183854471848	-0.249543637080857	0.0...	-0.157340342992445	-0.059034756744953	0.0...
$\boldsymbol{X}_9$	0.783201387251299	-0.040266198269010	0.0...	-0.034848333296306	0.435583142613420	0.0...
$\boldsymbol{X}_{10}$	0.781600000001444	-0.000000000002147	0.0...	0.000000000001834	0.443081213952850	0.0...

	T
$\boldsymbol{X}_1$	0.000000000000000
$\boldsymbol{X}_2$	0.236788879638363
$\boldsymbol{X}_3$	1.010884466987540
$\boldsymbol{X}_4$	1.756927859260430
$\boldsymbol{X}_5$	1.963474218828280
$\boldsymbol{X}_6$	2.120557712057200
$\boldsymbol{X}_7$	2.319415057331600
$\boldsymbol{X}_8$	2.865106111638700
$\boldsymbol{X}_9$	3.857147386090900
$\boldsymbol{X}_{10}$	3.948543114845460

Note that the program is capable of accepting an arbitrary number of points, n, with $n \geq 3$; in this case, $n = 10$. The solutions in Table 4.1 are truncated to one significant digit for all values and then used as the baseline "patch points" for the Multiple Shooting algorithm. After the Multiple Shooting Algorithm

is run the ten truncated states should return to the ten known solutions in Table 4.1. If the trajectory successfully converged, it would prove that the Multiple Shooting algorithm was indeed a highly robust algorithm since the baseline trajectory was so badly degraded by data loss. The Variable Time, Multiple Shooting algorithm followed most of the methodology outlined in Chapter 3 with a few minor exceptions that allowed for a closed, periodic trajectory instead of an open one. The free-variable vector was defined in the usual way as a $7n-1$ vector

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X}_1 \\ \boldsymbol{X}_2 \\ \vdots \\ \boldsymbol{X}_n \\ T_1 \\ \vdots \\ T_{n-1} \end{bmatrix} \tag{4.3.1}$$

but the constraint vector is slightly different

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{bmatrix} \mathbf{X}_{2,T_1} - \mathbf{X}_2 \\ \vdots \\ \boldsymbol{X}_{n-1,T_{n-2}} - \boldsymbol{X}_{n-1} \\ \boldsymbol{X}_{n,T_{n-1}} - \boldsymbol{X}_1 \end{bmatrix} \tag{4.3.2}$$

yet still a $6(n-1)$ vector. Note that the final patch-point is designed to match the initial patch-point as shown by the last line in Equation 4.3.2. Also note that, on occasion, the program will not fully converge using this definition of the constraint vector. If this is the case, then the solution is to remove the

constraint on the y-velocity component of the final patch point. Because of numerical integration error, it may become difficult for the solver to exactly match the initial patch point with the final patch point. The introduction of a "slack" variable, which allows a small amount of leeway in the y-velocity component, will typically resolve this issue. Finally, the Jacobian matrix will change slightly to

$$
D\boldsymbol{F}(\boldsymbol{X}) = \left[\begin{array}{ccccccccccccccc}
\Phi_{(t_2,t_1)} & -I_{6\times6} & 0_{6\times6} & \cdots & & & & & & \cdots & 0_{6\times6} & \dot{\mathbf{X}}_{2,\,T_{1\,(6\times1)}} & 0_{6\times1} & \cdots & 0_{6\times1} \\
0_{6\times6} & \Phi_{(t_3,t_2)} & -I_{6\times6} & 0_{6\times6} & \cdots & & & & & \ddots & 0_{6\times1} & & \dot{\mathbf{X}}_{3,\,T_{2\,(6\times1)}} & \cdots & \cdots \\
\cdots & 0_{6\times6} & \Phi_{(t_4,t_3)} & -I_{6\times6} & 0_{6\times6} & \cdots & & & & & \cdots & 0_{6\times1} & \cdots & & \\
& \cdots & 0_{6\times6} & 0_{6\times6} & \ddots & & & & & & \cdots & \cdots & & & \\
& & & \cdots & & & & & & & \cdots & \cdots & & & \\
& & & & & & & & & \ddots & 0_{6\times6} & \cdots & & & \\
& & & & & & & & & \ddots & -I_{6\times6} & \cdots & & \cdots & \\
& & & & & & & & \Phi_{(t_{n-1}t_{n-2})} & & -I_{6\times6} & \cdots & & 0_{6\times1} & \\
0_{6\times6} & 0_{6\times6} & \cdots & & & & & & 0_{6\times6} & & \Phi_{(t_n,t_{n-1})} & 0_{6\times1} & \cdots & \dot{\mathbf{X}}_{n,\,T_{n-1\,(6\times1)}} & \\
I_{6\times6} & 0_{6\times6} & \cdots & & & & & & \cdots & & & & & &
\end{array}\right] \tag{4.3.3}
$$

which is a $6\,(n-1)\ \times\ 7n-1$ matrix. Note the change in the matrix at the bottom left corner of Equation 4.3.3 from a null matrix to an identity matrix. This reflects the insertion of the closed-orbit boundary condition.

The results of this program can be seen in Figure 4.4. The thick lines indicate trajectory segments that were propagated using the ten truncated patch points computed from patch points in Table 4.1. The colors correspond to identical trajectory segments with one set belonging to the truncated points (thick lines) and the other belonging to the newly computed trajectory segments (thin lines). Obviously, they are all highly undesirable and discontinuous trajectories. This program ran for 2.2 seconds with a total of 8 iterations before converging all segments/patch points to within 10^{-12} of each other. The converged patch points can be seen as color-coded stars with their corresponding trajectory segments of the same color. Notice that each trajectory segment matches the one before and after. In this way, it has been demonstrated that a continuous trajectory can be formed from a highly fragmented guess trajectory using the Variable-Time, Multiple Shooting technique.

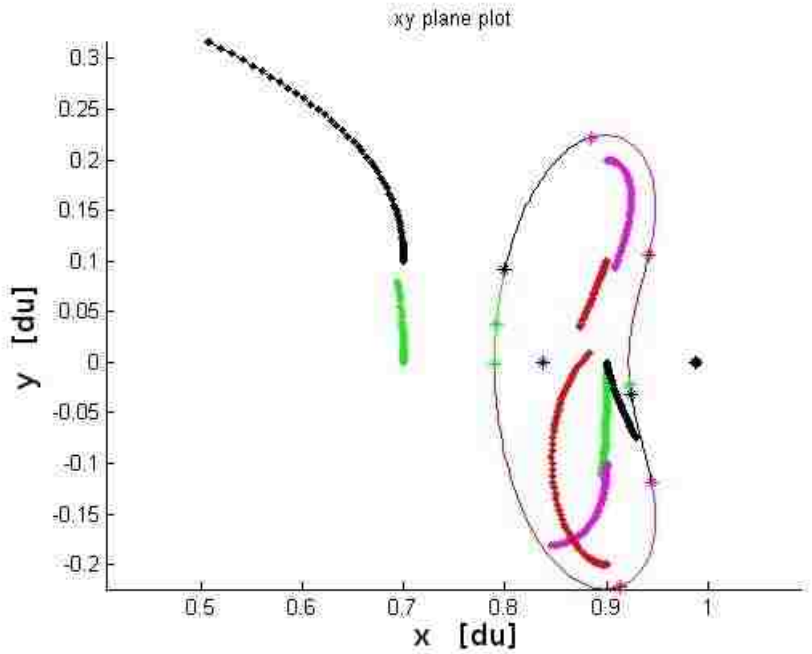


Figure 4.4: Data from Variable Time, Multiple Shooting program. Thick font represents the propagation of the initial guess, while thin font represents the propagation of the final solution.

4.4 Example of Three-Dimensional Orbits

In general, a quasi-periodic trajectory can usually be found that maintains a spacecraft within a fixed volume of space, for a long period of time, in the vicinity of a Lagrange point. These trajectories are fully three-dimensional and may or may not consist of one or more closed orbits (typically they are nearly closed but not fully closed). These trajectories are called "Lissajous" orbits. A special case of Lissajous orbits exist when the orbits are fully closed and periodic. Such trajectories are known as "halo" orbits; named after the shape they trace when viewed by an Earth-bound observer looking directly at the Moon and observing the shape of the spacecraft's trajectory [3]. Finally, if the

orbit is entirely two-dimensional and lies completely within the $x - y$ plane of the CR3BP, it is known as a "Lyapunov" orbit. All of the orbits displayed above are two-dimensional, Lyapunov orbits. However, there is no reason why a multiple shooting technique cannot be used to find halo orbits. Figure 4.5 shows the results of a multiple shooting program that calculates a three-dimensional halo orbit. This orbit was found using the same program that was used in the previous section, only a periodic z-component of small amplitude was added to the initial conditions. The result was a fully converged halo orbit that has been plotted in three-dimensions without the initial conditions displayed for simplicity's sake. All ten segments begin at a colored star and propagate forward to the next trajectory segment, beginning with a different colored star. Note that the sphere just to the right of the orbit is the Moon; plotted to the correct scale.

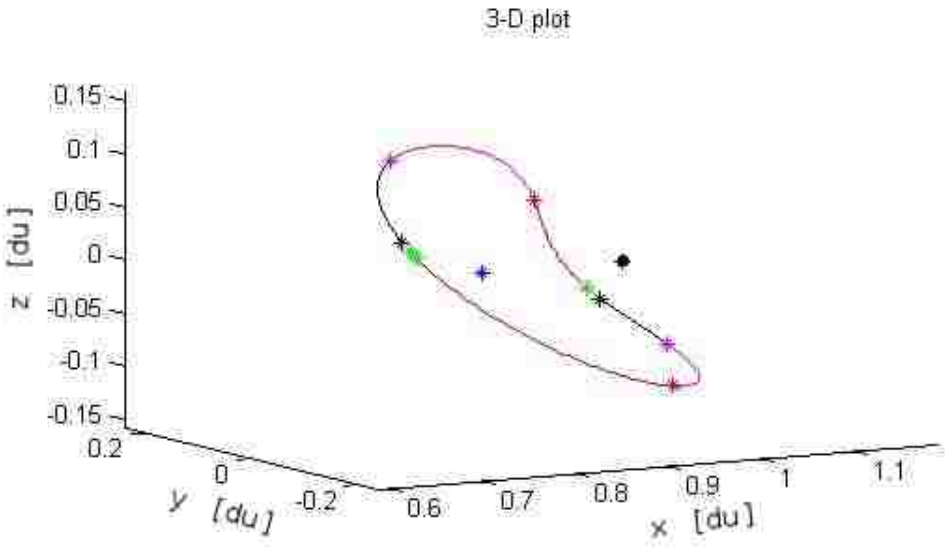


Figure 4.5: A three-dimensional Lagrange point orbit.

.

4.5 Full Ephemeris EOM Program with SPICE

The final program discussed in this chapter attempts to utilize SPICE commands to accurately propagate trajectories under a full-ephemeris force model using Equation 3.6.1. Of course, it is not enough to simply make a trajectory propagator; it must be validated as well. Validation is accomplished by comparing two trajectories propagated using the author's MATLAB ephemeris model with identical trajectories propagated using System Tool Kit (STK) software[1]. STK software is a commercial product that has been on the market for many

[1]http://www.agi.com

years and is well established as mature, tested, and validated [43]. Because of its established history, STK is ideal benchmarking platform.

The first trajectory is based on an arbitrary initial state of

$$\mathbf{X_0} = \begin{bmatrix} 350000 & [\text{km}] \\ 0 & [\text{km}] \\ 0 & [\text{km}] \\ 0 & [\text{km/s}] \\ 0.5 & [\text{km/s}] \\ 0 & [\text{km/s}] \end{bmatrix} \tag{4.5.1}$$

and was propagated for one year (365 days). This initial state represents a very high Earth orbit with a trajectory that is bound to a region slightly beneath the Moon's orbit. A custom full-ephemeris propagator was created in STK. For this study, the propagator used 10 bodies and modeled each body as a point mass. The bodies involved are the eight planets (Mercury, Venus, Earth, Mars, Jupiter, Saturn, Uranus, Neptune), the Sun, and the Moon. The Earth was chosen as the "central body." A satellite was then created and the propagator "Astrogator" was used to generate the baseline trajectory. Note that the coordinate system used by STK/Astrogator was "Earth Inertial," which is based on a geocentric J2000 coordinate system. The final state as computed by STK

was

$$\mathbf{X_f} = \begin{bmatrix} 228638.0987 & [\text{km}] \\ 279042.4137 & [\text{km}] \\ 20770.87224 & [\text{km}] \\ -0.714259 & [\text{km/s}] \\ 0.107036 & [\text{km/s}] \\ -0.010484 & [\text{km/s}] \end{bmatrix}. \tag{4.5.2}$$

If compared to the final state generated by the custom MATLAB ephemeris model, the magnitude of the difference in position is 0.1682 [km] and the magnitude of the difference in velocity is 0.00074 [m/s]. This is an exceptionally small difference between the MATLAB ephemeris model and STK's model and demonstrates the high degree of accuracy and precision of the MATLAB model created by the author of this dissertation. For the sake of comparison, the MATLAB model was re-run three times and compared with STK. Each time the model was run, more and more bodies were disregarded in the calculation. The results are shown in Table 4.2. Note that the propagator is essentially useless after the loss of the Sun.

Table 4.2: MATLAB custom propagator error.

MATLAB Propagator	Earth + Moon + Sun	Earth + Moon	Earth
Position Error [km]	18.89	263892.56	167715.65
Velocity Error [m/s]	0.04	580.94	602.80

It is important to recognize that validation of the custom MATLAB ephemeris model using only the trajectory described above may not be sufficient. The trajectories that are of primary interest in this proposal are located in the vicinity

of Lagrange points. Since Lagrange points are equilibrium points, it is implied that all the forces from the major primaries (Earth, Moon, and centripetal force) roughly sum to zero. Because of this fact, perturbations from other sources (i.e. the Sun and planets) may play an even larger role than that demonstrated in the first trajectory discussed above. It seems appropriate that a Libration orbit should also be verified in the custom MATLAB model.

Since an infinite number of Libration point orbits exist, a reasonable Libration point orbit to model is one that has been flown before by a recent spacecraft. To date, only one mission and two spacecraft have ever visited any Earth-Moon Lagrange points. The THEMIS (Time History of Events and Macroscale Interactions during Substorms) mission was launched in 2007 and were originally intended to study the interactions between the Sun and the Earth's magnetic field. The mission consisted of five spacecraft and was originally intended to be terminated at the end of 2010. It was then discovered that two of the original spacecraft had just enough fuel to insert themselves into Lissajous orbits about Earth-Moon L_1 and L_2 [14, 44]. THEMIS B was re-named ARTEMIS P1 and THEMIS C was re-named ARTEMIS P2 [48]. The ARTEMIS (Acceleration, Reconnection, Turbulence and Electrodynamics of Moon's Interaction with the Sun) mission then collected data in the vicinity of the Lagrange points for a few months. This data is available to the public and can be accessed on UC Berkely's website[2] [49].

The second trajectory propagated by both STK and the custom MATLAB model is based on the state vector of ARTIMIS P1 (THEMIS B) on May 1,

[2]http://themis.ssl.berkeley.edu/data/themis/ Acessed on 11-11-2012

2011 00:00:00.000 UTC. The state vector

$$\mathbf{X_0} = \begin{bmatrix} -346924.683095239 & [\text{Km}] \\ 120980.805483469 & [\text{Km}] \\ 16050.3697336028 & [\text{Km}] \\ -0.372332927758167 & [\text{Km/s}] \\ -0.882566329832206 & [\text{Km/s}] \\ -0.408755815977972 & [\text{Km/s}] \end{bmatrix} \tag{4.5.3}$$

was propagated for exactly 12 days, ending on May 13, 2011 00:00:00.000 UTC. Unfortunately the propagation time could not be extended beyond 12 days because the initial state vector was slightly off in its targeting and only allowed for the propagation of ¾ of an orbit before escaping the vicinity of the Lagrange point. The ARTIMIS spacecraft would typically perform a thruster burn every few days in order to correct its trajectory and keep it on a proper Lissajous orbit. For that reason, it was impossible for the author to compare a later state vector with one predicted by either STK or the custom MATLAB model (there was no model for thruster dynamics). Nevertheless a direct comparison between STK and the MATLAB model was performed. The magnitude of the difference in position was around 2.6 [m] while the magnitude of the velocity difference is around 10^{-5} [m/s]. These small values indicate that over relevant propagation times (nearly one orbit), the custom MATLAB model has been validated against STK. The trajectory of the spacecraft can be seen in CR3BP coordinates in Figure 4.6 and the same trajectory can also be plotted in a geocentric J2000 coordinate frame shown in Figure 4.7.

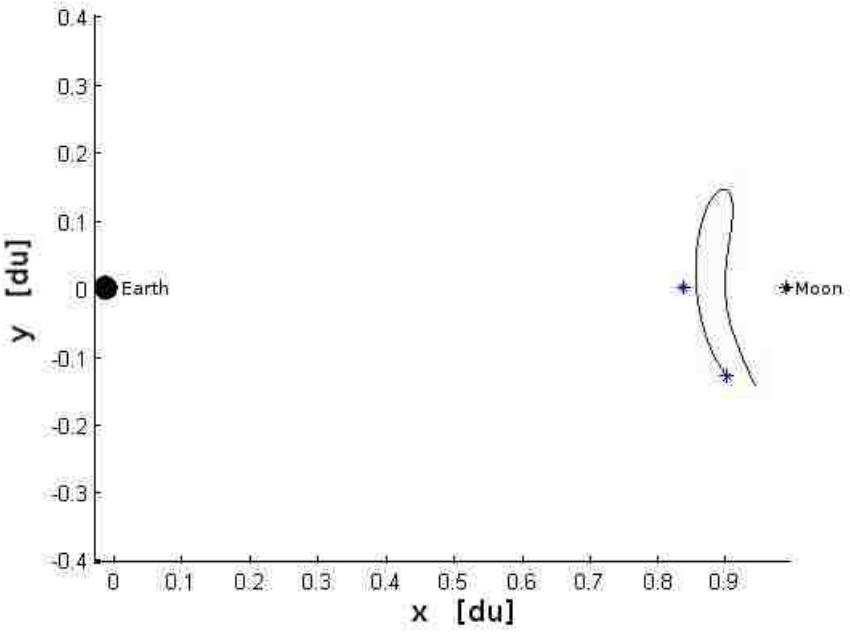


Figure 4.6: Propagated trajectory of ARTIMIS, P1 in the CR3BP reference frame. From *Left* to *Right*: Earth, L_1, Moon.

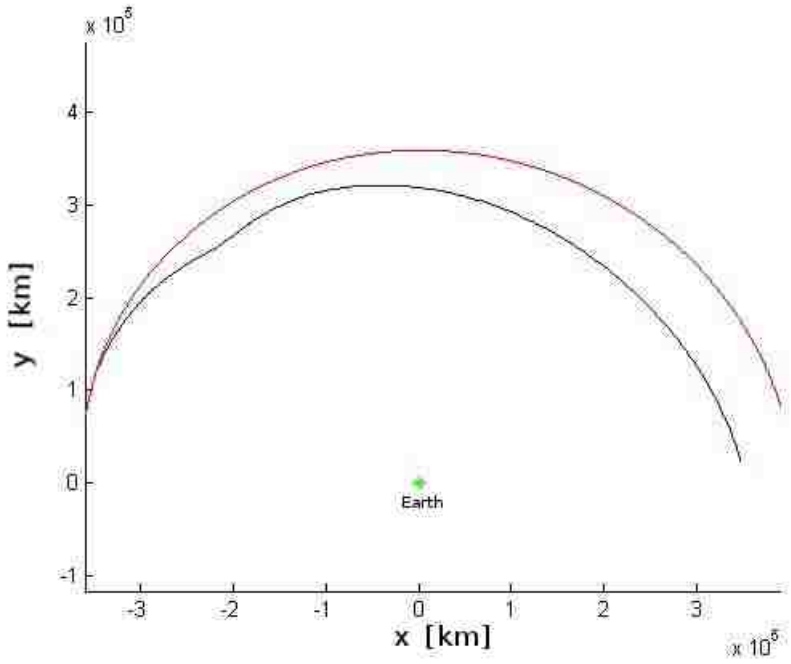


Figure 4.7: Propagated trajectory of ARTIMIS, P1 in the J2000 geocentric reference frame. *Key*: Earth = green, Moon = red, spacecraft = blue.

The methods outlined in this chapter reflect the current state-of-the-art in trajectory construction and mission planning for Lagrange point orbits. As demonstrated, the application of shooting methods allows for the quick identification of Lagrange point orbits. Of course these LPOs are typically unstable and require a spacecraft to expend some stationkeeping propellant in order to remain on the orbit over long time periods. In the next chapter, a closer examination of trajectories exiting and entering the LPO are discussed at length. These trajectories, otherwise known as *manifolds*, offer insight into the dynamical behavior of the spacecraft. The study of these stable and unstable manifolds can offer great insights into the navigation, optimization, and orbital maintenance of Lagrange point orbiting spacecraft.

5 Dynamical Systems Theory and Invariant Manifolds

This chapter focuses on Dynamical Systems Theory as applied to periodic orbits about Lagrange points in the Circular Restricted Three Body Problem (CR3BP). A more detailed explanation can be found Perko's book [50] and complemented by other references [51, 52, 53, 54, 55], but a useful summary is presented here. Dynamical Systems Theory (DST) is useful in evaluating the stability of Lagrange Point Orbits (LPO) in the CR3BP. It is based on an extrapolation of the stability analysis done in Chapter 2 for the individual Lagrange points themselves, but is a bit more involved, because the focus is on the trajectory of a periodic orbit instead of a single point in space. This chapter begins by linearizing the motion about an LPO and applying Floquet's theorem to a nonautonomous mapping. Next, the monodromy matrix is used to characterize the stability of the LPO and identify the directions of the local stable and unstable manifolds. Finally, a procedure for the generation of the stable manifold of a LPO is outlined and demonstrated. This manifold serves as the search-space for the Particle Swarm Optimization algorithm as described in Chapter 6 and demonstrated in Chapters 7, 8, and 9.

5.1 Linearized Motion About A Periodic Orbit

The dynamics of this problem follow

$$\dot{\boldsymbol{X}} = f(\mathbf{X}) \tag{5.1.1}$$

where $\mathbf{X}$ is the state vector as defined in Equation 2.2.28 and $f(\mathbf{X})$ represents the dynamics of the system as defined by the equations of motion in Equation 2.6.1. Assuming that Equation 5.1.1 has a periodic orbit of period P one may define

$$\mathbf{X}_\gamma(t) = \boldsymbol{\gamma}(t) \tag{5.1.2}$$

as a closed, periodic path of the orbit with $0 \leq t \leq P$. A first-order linearization of Equation 5.1.1 is given as

$$\dot{\boldsymbol{X}}_\gamma(t) = A(t)\mathbf{X}_\gamma(t) \tag{5.1.3}$$

with $A(t) = Df(\gamma(t))$ being a continuous, differentiable, and $T-$periodic matrix. Equation 5.1.3 is a non-autonomous, first order, linear system of differential equations. The solution to Equation 5.1.3 is of the form

$$\mathbf{X}_\gamma(t) = \Phi\left(t, \ t_0\right)\Phi^{-1}\left(t_0, \ t_0\right)\mathbf{X}_{\gamma\mathbf{0}} \tag{5.1.4}$$

where the state transition matrix, $\Phi\left(t, \ t_0\right)$, is known as the fundamental matrix for the system (Equation 5.1.3). Inserting Equation 5.1.4 into Equation 5.1.3 yields

$$\dot{\Phi}\left(t, \ t_0\right) = A(t)\Phi\left(t, \ t_0\right) \tag{5.1.5}$$

with $\Phi\left(\ t_0,\ \ t_0\ \right) = \Phi^{-1}\left(\ t_0,\ \ t_0\ \right) = I$, as was stated in an earlier chapter by Equation 2.7.8.

5.2 Floquet's Theorem & the Non-Autonomous Mapping

According to *Floquet's Theorem* [50], the solution for Equation 5.1.5 can be expressed in the form

$$\Phi\left(\ t,\ \ t_0\ \right) = Q\left(t\right) e^{Bt} \tag{5.2.1}$$

with $Q(t)$ being a non-singular, differentiable, $P-$periodic matrix and B being a constant matrix. Also, $Q(0) = I$ since $\Phi\left(\ t_0,\ \ t_0\ \right) = I$. According to Perko [50], the $Q(t)$ matrix enables the mapping of the non-autonomous Equation 5.1.3 into the autonomous form

$$\dot{\boldsymbol{Y}}_\gamma = B\boldsymbol{Y}_\gamma \tag{5.2.2}$$

which is identical to the form of Equation 2.6.7. This mapping is formulated as

$$\boldsymbol{Y}_\gamma = Q^{-1}(t)\boldsymbol{X}_\gamma \tag{5.2.3}$$

and is shown to be a valid mapping that reduces the non-autonomous linear system of Equation 5.1.3 to the autonomous linear system of Equation 2.6.7. The proof of the validity of the mapping described in Equation 5.2.3 comes primarily from Perko [50] and is summarized next.

Mapping Proof

According to Equation 5.2.1

$$Q(t) = \Phi(t)e^{-Bt}. \tag{5.2.4}$$

Taking the first derivative of $Q(t)$ yields

$$\dot{Q}(t) = \dot{\Phi}(t)e^{-Bt} - \Phi(t)e^{-Bt}B \tag{5.2.5}$$

and substituting Equation 5.1.5 gives

$$\dot{Q}(t) = A(t)\Phi(t)e^{-Bt} - \Phi(t)e^{-Bt}B \tag{5.2.6}$$

and an additional substitution using Equation 5.2.1 yields (note that B and e^{-Bt} commute)

$$\dot{Q}(t) = A(t)Q(t) - Q(t)B. \tag{5.2.7}$$

The mapping of Equation 5.2.3 can be easily rewritten as

$$\boldsymbol{X}_\gamma(t) = Q(t)\boldsymbol{Y}_\gamma(t) \tag{5.2.8}$$

and differentiated with respect to time as

$$\dot{\boldsymbol{X}}_\gamma(t) = \dot{Q}(t)\boldsymbol{Y}_\gamma(t) + Q(t)\dot{\boldsymbol{Y}}_\gamma(t). \tag{5.2.9}$$

Equation 5.2.7 can be used as a substitution for $\dot{Q}(t)$ and yields

$$\dot{\boldsymbol{X}}_\gamma(t) = A(t)Q(t)\boldsymbol{Y}_\gamma(t) - Q(t)B\boldsymbol{Y}_\gamma(t) + Q(t)\dot{\boldsymbol{Y}}_\gamma(t) \tag{5.2.10}$$

which can be simplified as

$$\dot{\boldsymbol{X}}_\gamma(t) = A(t)\boldsymbol{X}_\gamma(t) + Q(t)\left(\dot{\boldsymbol{Y}}_\gamma(t) - B\boldsymbol{Y}_\gamma(t)\right) \tag{5.2.11}$$

with the substitution of the mapping equation (Equation 5.2.3). Examining Equation 5.2.11 shows that the only way this equation is true is if both the non-autonomous Equation 5.1.3 and the autonomous Equation 5.2.2 are both true. Since this proof began using the result of *Floquet's Theorem* it demonstrates that the utilization of *Floquet's Theorem* reduces a non-autonomous system into an autonomous system that is both simple and has a known solution. This is accomplished via the definition of a $Q(t)$ matrix that is assumed to be non-singular, differentiable, and $P-$periodic. In practice, it is very difficult to precisely define the form of the $Q(t)$ matrix given a specific problem. Fortunately, however, it is not necessary to identify the exact form of the $Q(t)$ matrix because the simple knowledge of its intrinsic properties is more than enough information to gain a great deal of insight into the dynamics of a problem.

5.3 Monodromy Matrix & Stability

Now that the validity of *Floquet's Theorem* has been established the question of the utility of this theorem remains. In theory, Equation 5.2.1 could be used for any value of t. In practice, however, most values of t are unhelpful because the exact form of the $Q(t)$ matrix is unknown. Fortunately, however, there are special values of t that allow one to take advantage of the unique properties of the $Q(t)$ matrix. One such value is $t = t_0 = 0$. Plugging this value into

Equation 5.2.1 gives the result

$$\Phi\left(t_0, \quad t_0 \right) = Q(t_0). \tag{5.3.1}$$

Note that $\Phi\left(t_0, \quad t_0 \right) = I$, as stated before. This implies that

$$Q(t_0) = I \tag{5.3.2}$$

which is an important result of the special case of $t = t_0 = 0$. Furthermore, because $Q(t)$ is $P-$periodic,

$$Q(t_0 + nT) = I \tag{5.3.3}$$

for any value of n, with n being a positive integer value. The monodromy matrix, M, is defined as $M = \Phi\left(P, \quad t_0 \right)$ (with $n = 1$) or the value of the state transition matrix exactly one cycle after its initial value of $\Phi\left(t_0, \quad t_0 \right) = I$. Substituting $n = 1$ into Equation 5.2.1 gives

$$M = \Phi\left(P, \quad t_0 \right) = Q\left(t_0 + P\right) e^{BP} \tag{5.3.4}$$

which simplifies to

$$M = e^{BP} \tag{5.3.5}$$

via Equation 5.3.3. Equation 5.3.5 is an important result because of its implications to the stability of orbit γ. The eigenvalues of e^{BP} are given by $e^{\lambda_j P}$ with $j = 1, ..., n$ where n is the rank of B. The eigenvalues of B, λ_j, are called the *characteristic exponents* of orbit $\gamma(t)$ while the values $e^{\lambda_j P}$ are called *characteristic multipliers*. Referring to Equation 5.3.5 it is obvious that the eigenvalues

of the monodromy matrix, M, are identical to these characteristic multipliers.

It turns out that the characteristic exponents, λ_j, determine the stability of the orbit. This can be demonstrated by the solution of the autonomous linear system of Equation 5.2.2 which is

$$\boldsymbol{Y}_{\gamma}(t) = \boldsymbol{Y}_{\gamma_0} e^{Bt}. \tag{5.3.6}$$

It is also possible to write related solutions of Equation 5.2.2 by exciting eigenmodes of the system. Let $\boldsymbol{\nu}_j$ be an eigenvector of B that is associated with eigenvector λ_j. The solution of Equation 5.2.2 can also be expressed as the summation of all eigenmodes of the system

$$\boldsymbol{Y}_{\gamma}(t) = \sum_{j=1}^{n} c_j e^{\lambda_j t} \boldsymbol{\nu}_j \tag{5.3.7}$$

with the coefficients c_j determined from $\boldsymbol{Y}_{\gamma_0}$. One can excite a single eigenmode of the system by intentionally setting $\boldsymbol{Y}_{\gamma_0} = \boldsymbol{\nu}_j$ and plugging the eigenmode solution

$$\boldsymbol{Y}_{\gamma}(t) = \boldsymbol{\nu}_j e^{\lambda_j t} \tag{5.3.8}$$

into Equation 5.2.2. This substitution yields the familiar equation $B\boldsymbol{\nu}_j = \lambda_j \boldsymbol{\nu}_j$, which is the definition of an eigenvalue and is intrinsically true. Since the eigenmode solution of Equation 5.3.8 has been verified as a valid solution of Equation 5.2.2 (because it is a single term of Equation 5.3.7 and all terms are linearly independent) it can offer a vast amount of insight into the stability of the system. The most general form of an eigenvalue is a complex number. Allow $\lambda_j = a_j + ib_j$ where a_j and b_j are both real numbers. Substitution of this

complex number into Equation 5.3.8 and using Euler's formula gives

$$\boldsymbol{Y}_{\gamma}(t) = \boldsymbol{\nu}_j e^{a_j t}(\cos b_j t + i \sin b_j t). \tag{5.3.9}$$

It can be seen from a simple inspection of Equation 5.3.9 that $\lim_{t\to\pm\infty} \boldsymbol{Y}_{\gamma}(t)$ becomes unbounded if $a_j \geq 0$, is bounded if $a_j \leq 0$, and does not change amplitude if $a_j = 0$. Therefore, these stability conditions yield insight into the stable, center, and unstable manifolds, respectively. Furthermore, this stability relationship can be extended to the eigenvalues of the monodromy matrix via Equation 5.3.5. The eigenvalues, Λ_j, of the monodromy matrix, M, are equal to the characteristic multipliers defined above. Allowing for the substitution of a complex number for λ_j gives

$$\Lambda_j = e^{\lambda_j t}|_{t=T} = e^{a_j t + i b_j t}|_{t=P} = e^{a_j t}(\cos b_j t + i \sin b_j t)\,|_{t=P}. \tag{5.3.10}$$

The absolute value of a complex number is defined as the square root of its complex conjugate. Following this definition, the absolute value of the eigenvalues of the monodromy matrix are

$$\|\Lambda_j\| = e^{a_j t}|_{t=P}. \tag{5.3.11}$$

Since the stability behavior relative to a_j is known, the stability as relative to $\|\Lambda_j\|$ can also be directly inferred. Note that when $a_j = 0$ the critical value of $\|\Lambda_j\| = 1$. If $\|\Lambda_j\| < 1$, the eigenmode is stable, and if $\|\Lambda_j\| > 1$, the eigenmode is unstable. These results are summarized in Table 5.1.

Table 5.1: Summary of eigenmode stability characteristics where Λ_j is an eigenvalue of the monodromy matrix, M.

Name	Symbol	Stable If:	Center If:	Unstable If:
Characteristic Exponents	λ_j	$Re\left(\lambda_j\right) < 0$	$Re\left(\lambda_j\right) = 0$	$Re\left(\lambda_j\right) > 0$
Characteristic Multipliers	$\Lambda_j = e^{\lambda_j P}$	$\|\Lambda_j\| < 1$	$\|\Lambda_j\| = 1$	$\|\Lambda_j\| > 1$

5.4 Characteristics of Monodromy Matrix Eigenvalues

It can be shown that at least one of the characteristic multipliers of the closed, periodic orbit is equal to one. This is accomplished via the following proof.

5.4.1 $\Lambda = 1$ Proof

By definition, the periodic orbit $\mathbf{X}_\gamma(t) = \boldsymbol{\gamma}(t)$ satisfies Equation 5.1.1 as

$$\dot{\boldsymbol{X}}_\gamma\left(t\right) = f\left(\mathbf{X}_\gamma\left(t\right)\right). \tag{5.4.1}$$

Differentiating Equation 5.4.1 yields

$$\ddot{\boldsymbol{X}}_\gamma = Df\left(\mathbf{X}_\gamma\left(t\right)\right)\dot{\boldsymbol{X}}_\gamma\left(t\right) \tag{5.4.2}$$

but $A(t) = Df\left(\mathbf{X}_\gamma\left(t\right)\right)$ so

$$\ddot{\boldsymbol{X}}_\gamma = A\left(t\right)\dot{\boldsymbol{X}}_\gamma\left(t\right) \tag{5.4.3}$$

which has an identical form to Equation 5.1.3 (assuming the second derivative is replaced by the first). Recall that Equation 5.1.4 is the solution to Equation 5.1.3. Likewise,

$$\dot{\boldsymbol{X}}_{\gamma}(t) = \Phi(t, t_0)\,\Phi^{-1}(t_0, t_0)\,\dot{\boldsymbol{X}}_{\gamma_0} \tag{5.4.4}$$

with $\Phi^{-1}(t_0, t_0) = I$. Note that $\dot{\boldsymbol{X}}_{\gamma_0} = f(\mathbf{X}_{\gamma_0})$ by Equation 5.4.1. This reduces the solution to

$$\dot{\boldsymbol{X}}_{\gamma}(t) = \Phi(t, t_0)\, f(\mathbf{X}_{\gamma_0}). \tag{5.4.5}$$

Recall that $\dot{\boldsymbol{X}}_{\gamma}(T) = \dot{\boldsymbol{X}}_{\gamma_0} = f(\mathbf{X}_{\gamma_0})$, due to the periodic nature of the orbit, and $\Phi(t, t_0) = M$. Substituting this into Equation 5.4.5 when $t = P$ gives

$$(1)\, f(\mathbf{X}_{\gamma_0}) = M f(\mathbf{X}_{\gamma_0}) \tag{5.4.6}$$

which is the definition of an eigenvector/eigenvalue equation. Note that the eigenvalue of the monodromy matrix is 1 with an eigenvector of $f(\mathbf{X}_{\gamma_0})$ based on this equation.

5.4.2 Reciprocal Pairs of Λ_j in the CR3BP

It was shown by Breakwell [47] and Howell [56] that the eigenvalues of M in periodic orbits in the CR3BP occur in reciprocal pairs as $\left(\Lambda_1,\ \frac{1}{\Lambda_1},\ \Lambda_2,\ \frac{1}{\Lambda_2},\ 1,\ 1\right)$ due to the invariance of the equations of motion of the CR3BP (Equation 2.6.1) under the transformation $t \to -t$ and $y \to -y$. This transformation is simply another way to state the fact that the CR3BP is time-invariant ($t \to -t$) and symmetric across the $y-$axis ($y \to -y$). Under this transformation, the

autonomous linear system of Equation 5.2.2 can be rewritten as

$$\dot{\boldsymbol{Y}}_{\gamma}(-t) = B\boldsymbol{Y}_{\gamma}(-t) \tag{5.4.7}$$

and the solution rewritten as

$$\boldsymbol{Y}_{\gamma}(-t) = \boldsymbol{Y}_{\gamma_0} e^{-Bt}. \tag{5.4.8}$$

As before, the summation of eigenmode solutions is

$$\boldsymbol{Y}_{\gamma}(-t) = \sum_{k=1}^{n} c_k e^{-\lambda_k t} \boldsymbol{\nu}_k \tag{5.4.9}$$

which is nearly unchanged from Equation 5.3.7 except for the negative sign in the exponent and indexing with respect to k instead of j. It is important to note that the eigenvalues of B are both λ_j and λ_k since B remains unchanged under the transformation (because it is constant) and both Equation 5.3.7 and Equation 5.4.9 are valid solutions of the CR3BP dynamics. This can only be true in two cases:

1. the trivial case where $\lambda_j = \lambda_k = 0$ or

2. the negative pair case where
   $\lambda_j = \left(\lambda_1, \quad \lambda_2 = -\lambda_1, \quad \lambda_3, \quad \lambda_4 = -\lambda_3, \quad \lambda_5, \quad \lambda_6 = -\lambda_5 \right)$ and
   $\lambda_k = \left(\lambda_1 = -\lambda_2, \quad \lambda_2, \quad \lambda_3 = -\lambda_4, \quad \lambda_4, \quad \lambda_5 = -\lambda_6, \quad \lambda_6 \right)$.

This result is sometimes referred to as Lyaponov's Theorem [55]. Assuming the nontrivial case where $\lambda_1 = -\lambda_2$ and using Equation 5.3.10, one can write

$$\begin{aligned} \Lambda_1 &= e^{\lambda_1 P} \\ \Lambda_2 = e^{\lambda_2 P} = e^{-\lambda_1 P} &= \left(e^{-\lambda_1 P}\right)^{-1} = \tfrac{1}{\Lambda_1} \end{aligned} \quad . \tag{5.4.10}$$

This means that all of the eigenvalues of the monodromy matrix occur in reciprocal pairs in the CR3BP. This, along with the $\Lambda = 1$ Proof from above, indicate that the eigenvalues of the monodromy matrix of any closed periodic orbit in the CR3BP are of the form

$$\Lambda_j = \left(\Lambda_1, \ \tfrac{1}{\Lambda_1}, \ \Lambda_2, \ \tfrac{1}{\Lambda_2}, \ 1, \ 1 \right) \tag{5.4.11}$$

or, equivalently

$$\Lambda_j = \left(\Lambda_1, \ \tfrac{\Lambda_1^*}{\Lambda_1 \Lambda_1^*}, \ \Lambda_2, \ \tfrac{\Lambda_2^*}{\Lambda_2 \Lambda_2^*}, \ 1, \ 1 \right) \tag{5.4.12}$$

where Λ_j^* is the complex conjugate of Λ_j.

5.4.3 The Stability Index in the CR3BP

Now that the reciprocal pair relationship of the eigenvalues of the monodromy matrix has been established, a new "stability index" can be defined [57] that will summarize the stability of a closed, periodic orbit in the CR3BP. The stability index, S_i, is defined as

$$S_i = \frac{1}{2}\left(\|\Lambda_i\| + \left\| \frac{1}{\Lambda_i} \right\| \right) \tag{5.4.13}$$

for $i = 1, 2$. A periodic orbit of the CR3BP is considered "stable" (actually a center subspace) if $S_1 = 1$ and $S_2 = 1$, as opposed to the orbit being unstable for either $S_1 > 1$ or $S_2 > 1$. Note the stability index can never be less than 1. A stability index near 1 indicates an orbit that will have low Δv requirements for station-keeping but large Δv requirements if a change in the orbit is desired. Conversely, transfer costs will be low for large values of the stability index and Δv costs will be high for station-keeping. The stability index is just one of many characteristics of a Lagrange point orbit that determines the orbits utility given a specific mission profile.

5.5 Stable Manifold Generation

A Lagrange point orbit can be identified via either tabulated values [2] or a numeric computation that solves a two-point boundary value problem [41, 36]. The Lagrange point orbit is then defined by an arbitrary state vector (anywhere along the orbit), $\mathbf{X}_{\mathbf{orbit}}$, and the period of the orbit, P. Under controlled CR3BP dynamics, the state vector for the spacecraft is expressed in terms of position, velocity, and spacecraft mass as

$$\mathbf{X} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ m \end{bmatrix} \tag{5.5.1}$$

with the controlled CR3BP expressed in terms of a first-order differential equation as

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \dot{m} \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ 2v_y + U_x + u_x \\ -2v_x + U_y + u_y \\ U_z + u_z \\ -\frac{T}{I_{sp} g_o} \end{bmatrix}. \tag{5.5.2}$$

Note that the controlled form of Equation 5.5.2 degenerates to the uncontrolled Equation 2.6.1 when the control is removed ($T = 0$). Numeric integration of Equation 5.5.2 (with ballistic assumptions of $\mathbf{u} = \mathbf{T} = 0$ and $m = constant$) as well as the state transition matrix for one period with the initial condition $\mathbf{X_0} = \mathbf{X_{orbit}}$ enables the calculation of the monodromy matrix associated with the initial state, $\mathbf{X_{orbit}}$. The stable eigenvector, $\boldsymbol{\nu_s}$, of the monodromy matrix is multiplied by a very small number, ϵ, with $\epsilon = 10^{-10}$ in this dissertation. A perturbed initial state, $\boldsymbol{X}_{pert} = \boldsymbol{X}_{orbit} \pm \epsilon \boldsymbol{\nu_s}$, is then integrated backward in time using the ballistic version of Equation 5.5.2. As in Abraham et al.,[58] the integration is terminated when the spacecraft crosses the $y - z$ plane from the negative $x-$direction. This defines a trajectory that is a member of the stable manifold of the nominal Lagrange point orbit bounded by the $y - z$ plane. The nominal Lagrange point orbit can be discretized into M states defined as $\mathbf{X}^{(\mathbf{k})}_{\mathbf{orbit}}$ with $k \in [1, M]$. The process is then repeated for other values of $\mathbf{X}^{(\mathbf{k})}_{\mathbf{orbit}}$ (all of which are on exactly the same orbit) which, in turn, constructs a stable manifold consisting of M trajectories. If a spacecraft's state lies along a trajectory within this manifold then the ballistic flow forward in time will take

it to the state $\mathbf{X}_{\mathbf{orbit}}^{(\mathbf{k})}$ and the spacecraft will be automatically inserted into the nominal Lagrange point orbit. In this way, any state within the manifold, $\mathbf{X}_{\mathbf{s.m.}}(\tau_{01},\, k)$, can be expressed via two parameters:

1. An integer $k \in [1,\, M]$ that corresponds to a state on the nominal orbit $\mathbf{X}_{\mathbf{orbit}}^{(\mathbf{k})}$ with M being the total number of states that represent a discretization of the orbit.

2. A time parameter τ that represents the time remaining for a ballistic flow of Equation 5.5.2 to reach the state $\mathbf{X}_{\mathbf{orbit}}^{(\mathbf{k})}$.

Note that the eigenvalues remain constant for a given CR3BP orbit [46, 45, 52] regardless the value of $\mathbf{X}_{\mathbf{orbit}}^{(\mathbf{k})}$ (as long as it is still a state on the orbit). Only the eigenvectors are unique to each value of $\mathbf{X}_{\mathbf{orbit}}^{(\mathbf{k})}$.

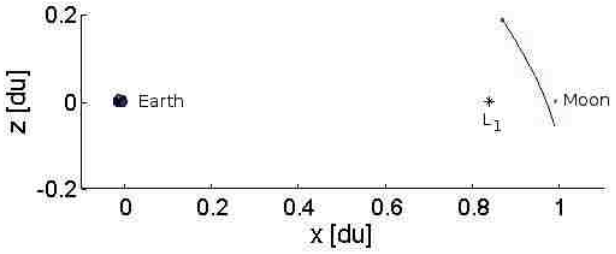
5.6 Example: L_1 Halo Orbit and its Associated Manifold

This example uses an Earth-Moon, L_1, northern halo orbit. In this case a "northern" halo orbit spends the majority of flight time above the northern lunar hemisphere. This orbit, and its associated stable manifold, are used as the destination orbit in chapters 7, 8, and 9. The orbit has a period of

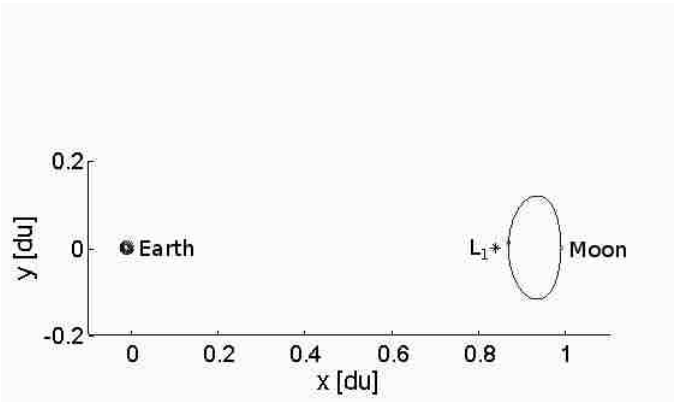
$P = 2.31339\,[\text{tu}]$ (10 days) and an initial state of

$$\mathbf{X_0} = \mathbf{X_{orbit}} = \begin{bmatrix} 0.866224052875085 \\ 0.011670195668094 \\ 0.186912185139037 \\ 0.013870554690931 \\ 0.245270168936540 \\ -0.021792775971957 \end{bmatrix} \tag{5.6.1}$$

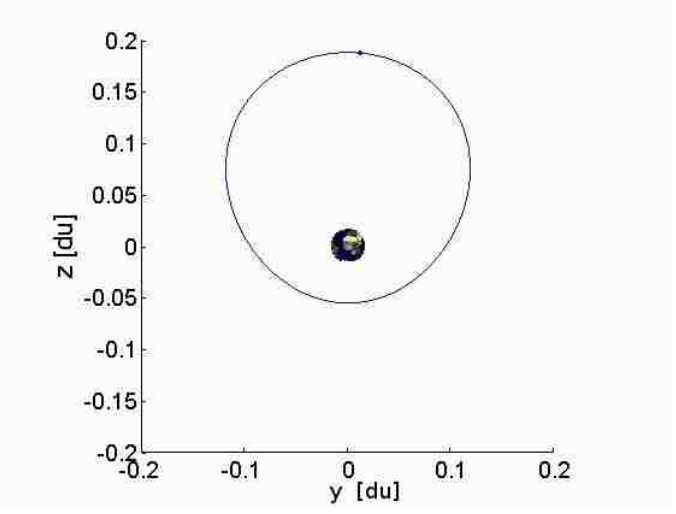
expressed in [du] and [vu]. Figure 5.1 and Figure 5.2 display multiple views of this example orbit.



(a)



(b)



(c)

Figure 5.1: Example northern halo orbit about Earth-Moon L_1 (drawn as an asterisk). (a) side view, (b) top view, (c) $x-$axis view.

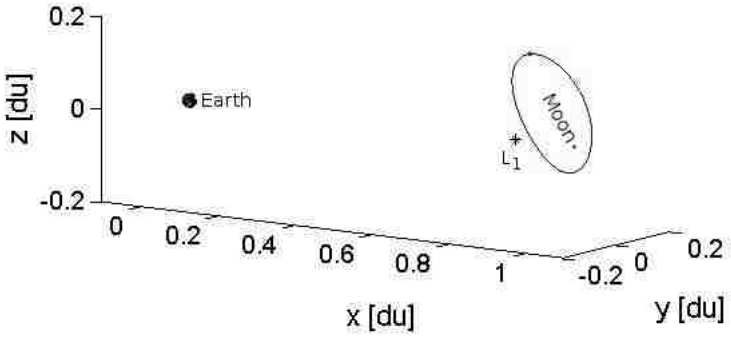


Figure 5.2: Off-axis view of the example, northern, halo orbit about Earth-Moon L_1.

The orbit is also displayed in Figure 5.3 with the states being discretized into $k_{max} = M = 791$ points with each point corresponding to a particular trajectory, k, on the invariant stable manifold.

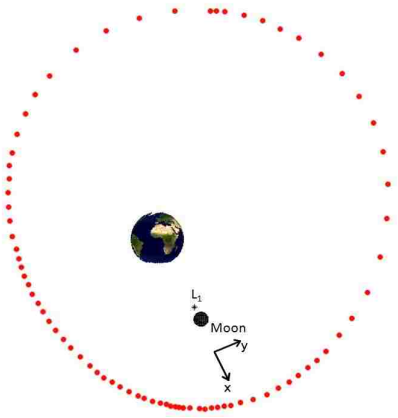


Figure 5.3: The nominal Earth-Moon L_1 northern halo orbit. The orbit is broken into $M = 791$ points with every 10^{th} point displayed in this figure.

This orbit was arbitrarily chosen as an example of how the PSO technique can optimize a transfer trajectory to a complicated and highly three dimen-

sional orbit. According to Dynamical Systems Theory (DST) the invariant stable manifold of this northern halo orbit may be generated by the following procedure:

1. Select a point on the orbit, $\mathbf{X}^{(\mathbf{1})}_{\mathbf{orbit}}$, and integrate the State Transition Matrix forward in time for one period.

2. Calculate the eigenvectors associated with the direction of the stable manifold at $\mathbf{X}^{(\mathbf{1})}_{\mathbf{orbit}}$.

3. Multiply this eigenvector by a small number, ϵ, and add/subtract this small perturbation to $\mathbf{X}^{(\mathbf{1})}_{\mathbf{orbit}}$.

4. Propagate this perturbed state vector backward in time until it crosses the $y - z$ plane from the $-x$ direction.

This procedure can be repeated numerous times; once for each point on the nominal orbit for $k \in [1,\ N]$. All trajectories generated by points on the halo orbit comprise the invariant stable manifold, W^s, of the halo orbit. Figure 5.4 illustrates the invariant stable manifold of the Earth-moon, L_1, northern halo orbit.

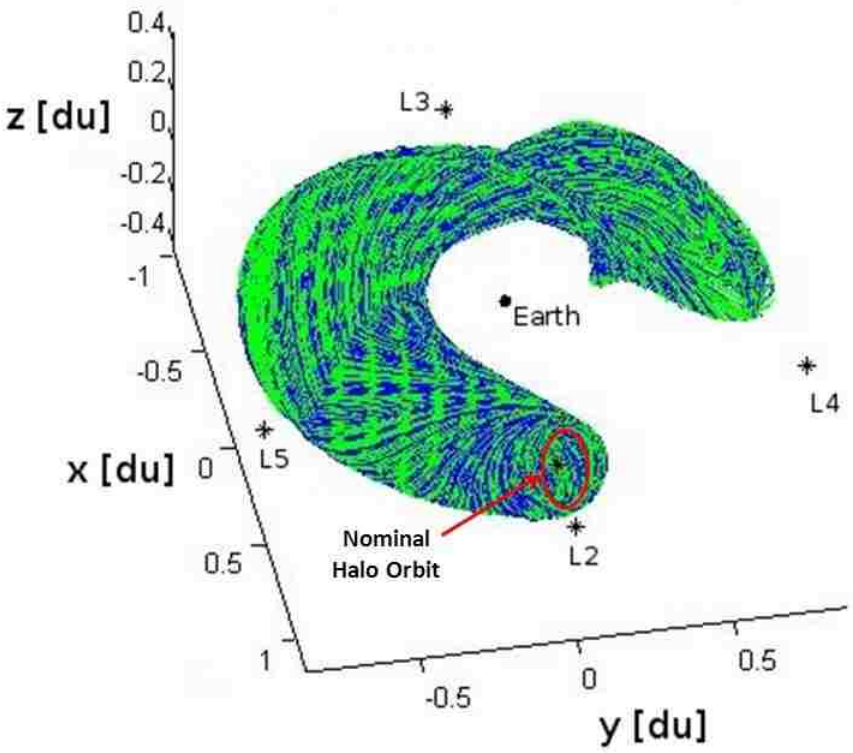


Figure 5.4: The invariant stable manifold of the nominal, L_1 northern halo orbit (green-blue trajectories). Note that the manifold never approaches the vicinity of low Earth orbit.

Figure 5.5 illustrates the locations of the eigenvalues in the complex plane. The eigenvalues for this example, northern, halo orbit are

$$\Lambda = \begin{bmatrix} 4.020036768696304 \\ -0.354045921058469 + 0.935228039454473i \\ -0.354045921058469 - 0.935228039454473i \\ 1.022630802058527 \\ 0.977870017200811 \\ 0.248753943692300 \end{bmatrix} \tag{5.6.2}$$

with the two stability indices as $S_1 = 2.13$ and $S_2 = 1.00$ quantifying the overall instability of this example orbit.

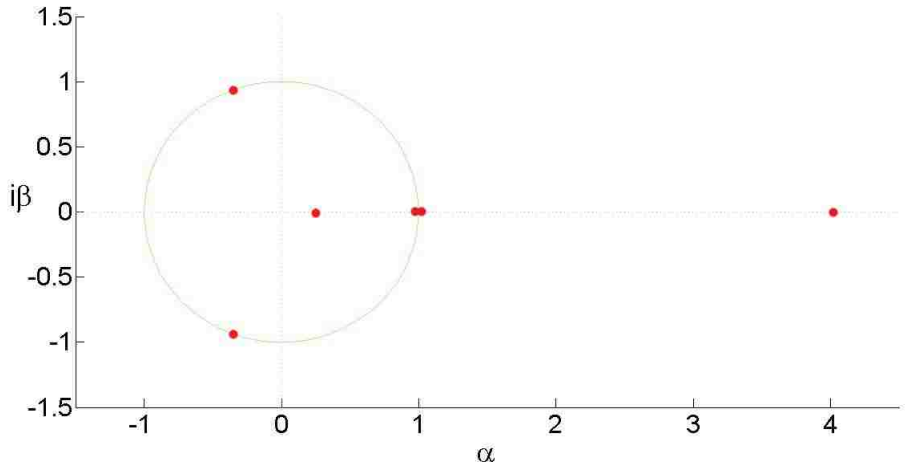


Figure 5.5: Eigenvalues (in red) of the example northern halo orbit are plotted with the unit circle in the complex plane.

Now that the existence of invariant manifolds has been firmly defined and discussed, it becomes possible to use them as part of an optimization algorithm. The next chapter focuses on the application of Particle Swarm Optimization to solve many difficult problems without utilizing gradient-based information. In later chapters, the particle swarm algorithm is applied to the invariant stable manifold to search for optimal "patch points" that join the manifold to a transfer trajectory that originates near the Earth.

6 Particle Swarm Optimization

This chapter outlines the main method of optimization used in this dissertation: Particle Swarm Optimization (PSO). PSO is a heuristic, evolutionary-based algorithm that can be used when gradient-based optimization techniques fail. The chapter begins with a brief history of the PSO method and discusses the formation of a search-space from the invariant stable manifold of a parent Lagrange point orbit. The search-space is parametrized for the sake of algorithmic simplicity and an "objective", "cost", or "fitness" function will be defined. The mechanisms of Particle Swarm Optimization are discussed and applied to a hypothetical fitness function. Finally, a "local" version of PSO is defined and boundary conditions for either version of PSO are examined in detail.

6.1 History and Background of PSO

Particle Swarm Theory is a branch of evolutionary computation known as "swarm intelligence" and is classified as a heuristic algorithm. According to Back [59], the history of evolutionary computation dates back to the late 1950's when computers were in their infancy. Since computational power was very limited, simple heuristic approaches to problem solving were favored over more complex and computationally expensive methods – even though they are usually

more precise. Examples from this period are found in references [60, 61, 62]. As Moore's Law [63] increased computational capabilities, evolutionary algorithms became commonplace in the engineering world.

Particle Swarm Optimization (PSO), in particular, is a relatively new algorithm that was originally developed by Kennedy and Eberhart in 1995 [64]. Kennedy, a social psychologist, and Eberhart, an electrical engineer, originally developed this algorithm to study the flocking/swarming behavior of birds. Until that time, the dominant thought was that bird flocking was controlled by the individual bird's tendency to maintain a constant separation distance between itself and neighboring birds [65, 66]. Kennedy and Eberhart abandoned that reasoning and focused instead on the social dynamics of the situation. Through much trial and error they found that they could write a very robust algorithm using only two simple equations (Equations 6.5.2 and 6.5.3) that kept the complexity of the algorithm to a minimum. They sought an algorithm that was able to model the swarming behavior of a flock of birds attempting to locate food within a cornfield. The "cornfield," in this case, became known as the search-space, which, for a given problem, is a subset of $\mathbb{R}^n$ and the "food" was the minimization of an objective, fitness, or cost function such as Equation 6.4.1.

These equations involved three simple terms that controlled the movement of each individual "particle" (a.k.a. "bird"). The first term was a simple momentum term that modeled Newtonian physics: a particle in motion will stay in motion with constant velocity. The second term mimics the cognitive nature of the bird. This term redirects the particle's velocity vector towards the best location yet found by that individual particle in the search-space (i.e. the location visited with the most food in the cornfield). The final term mimics

the social intelligence of birds. This "social" term informs each particle in the swarm of the best location yet found by any individual member of the swarm. The particle's velocity vector is then redirected in that direction. This social property was inspired by a paper written by sociobiologist Wilson [67] who said, "*In theory at least, individual members of the school can profit from the discoveries and previous experience of all other members of the school during the search for food. This advantage can become decisive, outweighing the disadvantages of competition for food items, whenever the resource is unpredictably distributed in patches.*" In addition to the influence of these three terms, a final parameter, initially called "craziness," was introduced into the system. This parameter is manifested in the form of three stochastic weights that are applied to each of the three terms mentioned above. The effect of these weights was to introduce "crazy," random velocities that were slightly unpredictable in nature. As in evolutionary computation, these random variables allow the particles to effectively explore the search-space without becoming prematurely trapped in local minima. Kennedy and Eberhart found that this stochastic property was essential to the efficacy of the PSO algorithm and could not remove this feature without vastly diminishing its capability [64]. An extensive description of the PSO algorithm can be found in references [64, 36, 68, 69].

Over the past decade, the PSO algorithm has made its way into a plethora of scientific and engineering applications. According to a recent survey of papers by Poli [68] within the IEEE Xplore database, the PSO algorithm has been applied to: antenna design, biomedical applications, communication networks, control systems, engines and motors, entertainment (games and music), and scheduling problems – just to name a few. In fact, the application of the PSO algorithm has seen a near exponential growth between 1995 and

2006. It is favored by so many different fields because of its simplicity, ease of programming, flexibility, and ability to be hybridized with more traditional methods. A more recent application of PSO is applied to spacecraft trajectory optimization problems. Interplanetary trajectories were optimized by Bessette and Spencer [37, 38] while Pontani and Conway focused on multiple-burn rendezvous problem [35] as well as general spacecraft trajectory optimization [34, 36]. Other spacecraft trajectory optimization problems include solar sails [70], low-thrust asteroid missions [71], and satellite coverage problems [72] among others [73, 74].

6.2 Search Space & Low-Thrust Control Law

Using Dynamical Systems Theory, the invariant stable manifold of the nominal Earth-Moon Lagrange point orbit was constructed as described in Chapter 5. Next, a particular trajectory segment, k, is randomly selected from the stable manifold, W^s. From this trajectory segment a random state, $\mathbf{X_{s.m.}}(\tau_{01},\, k)$, is chosen to be a "patch point." The "Manifold Time," τ , is a measure of the amount of time that is required for the spacecraft to ballistically travel from $\mathbf{X_{s.m.}}$ to the nominal Lagrange point orbit given a particular value of k. Similarly, the value τ_{01} represents a "normalized" Manifold Time (given k), because this time is expressed in a range between 0 and 1, with 0 indicating a point on the LPO, and 1 indicating a point a the far end of the manifold where the manifold crosses the $y-z$ plane. A low-thrust trajectory is then propagated backwards in time, according to Equation 2.2.31, from this patch point until the Jacobi energy of the spacecraft is equal to the Jacobi energy of a Geosynchronous Earth Orbit (GEO) (or some other user-defined stopping condition);

then the integrator stops. The control law for the low-thrust acceleration during this propagation is quite straightforward

$$\begin{aligned} \|\mathbf{u}(t)\| &= \tfrac{T_{max}}{m(t)} \\ \hat{\mathbf{u}}(t) &= \hat{\mathbf{v}}(t) \end{aligned} \tag{6.2.1}$$

with T_{max} being the magnitude of the maximum thrust of the engine and $\hat{\mathbf{v}}\,(\mathbf{t})$ being a unit vector oriented in the same direction as the instantaneous, three-body velocity of the spacecraft. This control law was successfully used by Mingotti et al. [30] and was shown to be the most fuel and time efficient control law obtainable. A summary of this method for an arbitrarily chosen trajectory and patch point is described below:

1. Create an invariant stable manifold, W^s, based on the nominal L-point orbit by using DST and backwards integrating the Equation 2.2.29 of the CR3BP

2. Arbitrarily select a trajectory, k, from this manifold and then arbitrarily select a patch point, $\mathbf{X_{s.m.}}\,(\tau_{01},\ k)$, from within this trajectory

3. Using Equation 2.2.31, propagate the low-thrust trajectory from this patch point backwards in time until the Jacobi energy matches that of GEO.

It is now possible to describe a feasible continuous trajectory from a GEO-energy orbit to a Lagrange point orbit given a patch point, $\mathbf{X_{s.m.}}$. It, therefore, becomes necessary to optimize this trajectory by identifying the optimal patch point, $\mathbf{X^*_{s.m.}}$. Thus, the search space for the PSO technique becomes all the

possible points within the stable manifold.

6.3 Parametrization of the Search Space

The Particle Swarm Optimization (PSO) algorithm requires the *a priori* definition of a "search space" where it is permitted to search for an optimal solution. In this study, the search space is defined as all states within the invariant stable manifold, $\mathbf{X_{s.m.}}(\tau_{01},\, k) \in W^s$, and within certain bounds. Each state is uniquely defined by exactly two parameters: k and τ. The parameter k represents an individual trajectory member of the stable manifold ($k \in W^s$) that is generated via the method outlined in Section 6.2. The parameter τ_{01} is the second parameter that defines $\mathbf{X_{s.m.}}$ and is defined relative to τ via a simple mapping function. The time of flight, τ, represents the amount of time required to get from an initial state to a state on the nominal Lagrange point orbit, as defined in the Section 6.2. Unfortunately, it is impossible to define the entire stable manifold as a search space because it is infinite in nature and a search space (by definition) must be finite. The bounds of τ, therefore, are carefully chosen such that a wide swath of relevant manifold states are captured within the search space and irrelevant manifold states are excluded.

In this study, the bounds of τ are $\tau_{L.B.} \leq \tau_{s.m.} \leq \tau_{U.B.}$ with:

- $\tau_{L.B.}$ being the time that trajectory k crosses the $y-z$ plane from the positive x direction
- $\tau_{U.B.}$ being the time that trajectory k crosses the $y-z$ plane located at $x = L_1$.

The $\tau_{U.B.}$ bound was chosen to ensure that the fitness function (Equation 6.4.1)

can be evaluated at any point within the search space. If, for example, a patch point was located on the moon side of L_1, the control law (Equation 6.2.1) would cause the spacecraft to flow towards a lunar orbit instead of an Earth orbit; thus invalidating Equation 6.4.1. The opposite bound, $\tau_{L.B.}$, was chosen as a matter of convenience and practicality. While it is true that the trajectories of W^s continue to flow for an infinite amount of time, one needs to cut off this flow after a finite amount of time due to the limitations of computing power. Since the run times of the PSO method can become quite large, a smaller search space is needed to adequately converge on an optimal solution. While this limited search space does preclude the possibility of lunar phasing maneuvers, such as those used by Mingotti et al.[30], it is more than adequate to address a fitness function that attempts to minimize time of flight. The values of τ are mapped to τ_{01} using the simple relationship that $\tau_{L.B.} = 1$ and $\tau_{U.B.} = 0$. Therefore, the values of τ for a given value of k are mapped to a normalized range of $0 \leq \tau_{01} \leq 1$. This mapping ensures that values of τ between $\tau_{L.B.}$ and $\tau_{U.B.}$are treated equally, regardless of the value of k and the time of flight between the nominal Lagrange point orbit and the $y - z$ plane located at $x = L_1$.

In a similar fashion, the values of k are mapped between $1 \leq k \leq M$ via a modulus function. In this study, $k = k_{desired}$ **mod** (M). This means, for example, that if $k_{desired} = M + x$ then $k = x$ assuming $0 \leq x \leq M$. Using this technique, no value of $k_{desired}$ is ever excluded from the search space but is instead looped back onto itself in k-space. In summary, any value of $\mathbf{X_{s.m.}}(\tau_{01},\ k)$ can be uniquely parametrized, in $k\tau$-space, in terms of τ_{01} and k with the boundaries of these parameters being real numbers, $0 \leq \tau_{01} \leq 1$, and positive integers, $1 \leq k \leq M$.

6.4 Fitness Function

The fitness function used in this dissertation can be expressed as a function of a state on the stable manifold,

$$J\left(\mathbf{X}_{\mathbf{s.m.}}\right) = c_1 \left\| e_{GEO}\left(\mathbf{X}_{\mathbf{s.m.}}\right) - e_{desired} \right\| + c_2 \Delta m\left(\mathbf{X}_{\mathbf{s.m.}}\right) + c_3 \Delta T\left(\mathbf{X}_{\mathbf{s.m.}}\right) \quad (6.4.1)$$

where e_{GEO} is the eccentricity of the GEO-energy orbit, $e_{desired}$ is the desired eccentricity of the GEO-energy orbit, Δm is the propellant mass used by the low-thrust maneuver, and $\triangle T$ is the total Time of Flight (TOF) from the initial Earth orbit to the nominal Lagrange point orbit. The constants $c_1,\ c_2,\ c_3$ are all weighting constants controlled by the researcher. In this study, the goal is to minimize the eccentricity of the original orbit and make it as circular as possible; therefore $e_{desired} = 0$. A circular orbit is desirable because most very low-thrust spirals, from LEO to GEO, begin and end with circular orbits. It is well known [30] that the most efficient control law for constant thrust (Equation 6.2.1) is the tangential thrust strategy, which will increase the specific energy of the spacecraft while leaving its eccentricity very near zero for most low-thrust trajectories. Thus, it is important that the target GEO-energy orbit also be near-circular if a mission planner desires to utilize a connecting low-thrust transfer from LEO as well.

Because no one term in Equation 6.4.1 should dominate the others, the choice of the constants $c_1,\ c_2,\ c_3$ is of critical importance. In this study, the constants

c_2 and c_3 are chosen such that the ratio

$$\frac{c_2 \Delta m\left(\mathbf{X}_{\mathbf{s.m.}}\right)}{c_3 \Delta T\left(\mathbf{X}_{\mathbf{s.m.}}\right)} \tag{6.4.2}$$

is on the order of 1. For example, if $\Delta m \approx 60$, $\Delta T \approx 30$ and the desired ratio of Equation 6.4.2 was roughly $1:1$ an appropriate choice of constants would be $c_2 = 0.001$, $c_3 = 0.002$. This selection method does not require *a priori* knowledge of the value of Δm or ΔT, but rather an *a priori* estimate of their relative magnitudes. Once c_2 has been determined, it is possible to set a value for c_1 by setting some target threshold for the first term in Equation 6.4.1. Values below this threshold would be overwhelmed by the other terms in Equation 6.4.1 while values above this threshold would dominate Equation 6.4.1. In this study, a threshold value of 1% seems reasonable because it represents a relatively small error in eccentricity. Thus the ratio

$$\frac{c_2 \Delta m\left(\mathbf{X}_{\mathbf{s.m.}}\right)}{c_1 \left\| e_{GEO}\left(\mathbf{X}_{\mathbf{s.m.}}\right) - e_{desired} \right\|} \tag{6.4.3}$$

should be roughly $1:1$ assuming $c_2 = 0.001$, $\Delta m \approx 60$, and the threshold value of $\left\| e_{GEO}\left(\mathbf{X}_{\mathbf{s.m.}}\right) - e_{desired} \right\|$ is roughly 10^{-2}. This gives a value of $c_1 = 1$. In summary, following the procedure outlined in this example yields the values $c_1 = 1,\ c_2 = 0.001,\ c_3 = 0.002$ for the weighting constants. This setup ensures convergence on an optimal solution that rigidly enforces the eccentricity constraint while minimizing the fuel usage and time of flight.

6.5 Particle Swarm Optimization

The PSO algorithm is simple yet powerful. It consists of N_p particles which are, initially (i.e. $j = 1$), randomly distributed throughout the search space with a position, $\boldsymbol{\chi} = [\tau_{01},\ k]^T$, and velocity, $\boldsymbol{\omega} = \begin{bmatrix} V_\tau, & V_k \end{bmatrix}^T$. Note that the position maps to the state vector in the following manner:

$$\boldsymbol{\chi} = [\tau_{01},\ k]^T \Rightarrow$$
$$\mathbf{X_{s.m.}}(\tau_{01},\ k) = \begin{bmatrix} x_{s.m.}, & y_{s.m.}, & z_{s.m.}, & \dot{x}_{s.m.}, & \dot{y}_{s.m.}, & \dot{z}_{s.m.}, & m_{s.m.} \end{bmatrix}^T. \tag{6.5.1}$$

Both the velocity and position of each particle in the search space is calculated by Equations 6.5.2 and 6.5.3, respectively

$$\boldsymbol{\omega}_i^{(j+1)} =$$

$$C_I\left(1 + R_1(i,\ j)\right)\boldsymbol{\omega}_i^{(j)} + C_C R_2(i,\ j)\left(\boldsymbol{\psi}_i^{(j)} - \boldsymbol{\chi}_i^{(j)}\right) + C_S R_3(i,\ j)\left(\mathbf{Y}^{(j)} - \boldsymbol{\chi}_i^{(j)}\right) \tag{6.5.2}$$

$$\boldsymbol{\chi}_i^{(j+1)} = \boldsymbol{\chi}_i^{(j)} + \boldsymbol{\omega}_i^{(j+1)} \tag{6.5.3}$$

with the superscripts indicating the j^{th} iteration ($1 \leq j \leq j_{max}$) of the PSO algorithm and the subscripts representing the i^{th} particle ($1 \leq i \leq N_p$) . Note that $R_{1,2,3}(i,j)$ represents a random number $0 \leq R_{1,2,3}(i,\ j) \leq 1$ following a uniform distribution, and the constants $C_I,\ C_C,\ C_S$, represent the "Inertial," "Cognitive," and "Social" weighting coefficients, respectively. The fitness function, Equation 6.4.1, is evaluated for each particle and $\boldsymbol{\psi}_i^{(j)}$ and $\mathbf{Y}^{(j)}$ are recorded. The "personal best" value, $\boldsymbol{\psi}_i^{(j)} = \begin{bmatrix} \tau_i^{(best)}, & k_i^{(best)} \end{bmatrix}$, represents the best known value of the fitness function recorded by particle i from iteration 1 to j. The "global best" value, $\mathbf{Y}^{(j)} = \begin{bmatrix} \tau^{(best)}, & k^{(best)} \end{bmatrix}$, represents the best

known value of the fitness function, recorded by any particle in the swarm, from iteration 1 to j. In this way, the position and velocity of each particle can be calculated for iteration $j+1$ based on the information contained in iteration j using Equations 6.5.2 and 6.5.3.

The inertial coefficient directs the particle's motion according to Newtonian mechanics (i.e. motion directed along the current velocity vector). The cognitive coefficient allows each particle to "remember" the best location it has visited and acts as an attractor to that location. Finally, the social coefficient allows each particle to "communicate" with the others in the swarm and attracts particle i to that location. The values of the coefficients used in this study have been inspired by the work of Pontani and Conway [34] and were only modestly modified, via trial and error, to achieve reasonable convergence while still identifying obvious local minima. They are summarized as follows:

$$\begin{aligned} C_I &= 0.15 \\ C_C &= 1.00 \\ C_S &= 1.00 \end{aligned} \quad . \tag{6.5.4}$$

6.6 Local PSO

The method outlined above is excellent for identifying local minima when only a few minima are present. Unfortunately, if a large number of local minima exist then the global PSO algorithm displays a tendency to converge on a non-optimal local minima instead of the best local minima discoverable. This algorithmic shortcoming exists because the global best solution, $\boldsymbol{Y}^{(j)}$, draws other particles away from what is oftentimes the vicinity of a better local minimum. This is especially true when the depth of multiple local minima are very similar. To

avoid this problem a "local" version of the PSO algorithm has been developed and utilized in this research. This is accomplished by limiting the ability of the particles to communicate over distances greater than some cutoff distance, $\mathbf{r}_{local}$. This mirrors conditions found in nature where collaborating swarms of animals have an inability (or a retarded ability) to communicate over vast distances, thus allowing more time to explore nearby local minima. In this study, $\boldsymbol{Y}^{(j)}$ is modified to $\boldsymbol{Y}^{(j)}_{local(i)}$ by utilizing the best value of a local swarm defined as all particles within radius, $\mathbf{r}_{local}$, of particle i. If particle i can not "see" a distant particle then that distant particle has no influence over the value of $\boldsymbol{Y}^{(j)}_{local(i)}$. Typically, $\mathbf{r}_{local} = \left[\begin{array}{cc} \frac{1}{20}, & \frac{1}{16}M \end{array} \right]^T$ in this study and is noted in the text otherwise.

As the PSO algorithm evolves over j iterations, the particles in the swarm begin to collect around various local minima. It becomes possible to define a convergence metric, γ, for the entire system. This metric is defined by

$$\gamma^{(j)} = \frac{N_C^{(j)}}{N_p} \tag{6.6.1}$$

where $N_C^{(j)}$ represents the number of particles that have converged to the vicinity of $\boldsymbol{Y}^{(j)}_{local(i)}$. The vicinity of $\boldsymbol{Y}^{(j)}_{local(i)}$is defined to be a circular area of $k\tau$-space, centered on $\boldsymbol{Y}^{(j)}_{local(i)}$, with a radius that is roughly 14% the size of r_{local} and an area that is roughly 2% the area of a circle of radius r_{local}. Using this definition of convergence it becomes possible to track the convergence of the PSO algorithm as a function of j and even terminate the algorithm early if a sufficient value of γ is reached.

6.7 Search Space Boundary Conditions:

Occasionally, a particle attempts to exit the permissible search space to which it must be bound. In such a case, a series of rules is followed to gently guide the particle back into the search space and continue its search for the best local minima discoverable. One reason why a particle may attempt to exit the search space is because its velocity is too large. In this case, a saturation limit is imposed on the velocity vector such that $\boldsymbol{\omega} \leq \boldsymbol{\omega}_{max} = \begin{bmatrix} \pm\tau_{max}, & \pm k_{max} \end{bmatrix}^T = \begin{bmatrix} \pm\frac{1}{2}, & \pm\frac{1}{2}M \end{bmatrix}^T$. In general, this velocity saturation limit is very high and only acts upon extremely unreasonable velocity values. As a consequence, some values of $\boldsymbol{\chi}$ still fall outside the search space. If this is the case, then the τ_{01} component of position is bounded by the saturation limits

$$\tau_{01} = \begin{cases} \tau_{max} = 1 & for\ \tau_{01} > \tau_{max} \\ \\ \tau_{min} = 0 & for\ \tau_{01} < \tau_{min} \end{cases} \tag{6.7.1}$$

and the k component is bounded by the modulus function $k = k_{requested}\ \mathbf{mod}\ M$. The velocity of this particle is also reset to $\boldsymbol{\omega} = \mathbf{0}$ to prevent the particle from exiting the search space during the subsequent iteration.

This concludes the chapter introducing Particle Swarm Optimization. Recall that in Chapter 2, the dynamics of the problem were first introduced. Chapter 3 and Chapter 4 introduced the concepts needed to construct an LPO, while Chapter 5 discussed the theory governing the stable and unstable manifolds. Finally, this chapter introduced the concept of Particle Swarm Optimization. In the next few chapters, these concepts will be unified in order to optimize transfer trajectories from an initial Earth orbit to a target LPO.

7 Application of One-Dimensional PSO

This chapter describes the application of a basic, one-dimensional Particle Swarm Optimization (PSO) algorithm attempting to optimize a low-thrust trajectory from a geosynchronous Earth orbit (or close to it) to an Earth-Moon, L_1, northern halo orbit. Note that some content for this chapter was taken from a publication by Abraham et al. [58] and reproduced with the author's and publisher's consent. This study holds parameter k fixed and optimizes with respect to the time parameter, τ, on a given segment of the invariant stable manifold. Three cases are considered: (A) a basic case where the eccentricity of the GEO-like orbit is zero, (B) a case where the eccentricity of the geocentric orbit is zero and the propellant consumption is minimized and, (C) a case where eccentricity, propellant, and time of flight are all, simultaneously optimized.

The cost function and PSO algorithm are based on those found in Chapter 6 while the search-space and destination Lagrange point orbit are based on those developed in Chapter 5. An individual point in the search-space (manifold) is evaluated by propagating, backwards in time, from that point using Equation 2.2.31 with the control law defined by Equation 6.2.1. The equations of motion are propagated back until the Jacobi energy (Equation 2.3.5) of the spacecraft

matches that of a spacecraft in Geosynchronous Earth Orbit (GEO). Then the integration terminates and the fitness function is evaluated based on the final conditions of the propagated trajectory (in geocentric orbit). Using this method a "cost" can be assigned to the manifold patch point under consideration. This process is repeated for each location visited by an individual particle in the swarm. The swarm terminates its search when the maximum number of iterations are reached.

7.1 Study A: Eccentricity-Only Fitness Function

7.1.1 Fitness Function

In this study, a very simple fitness function was chosen in such a way as to focus the goal of the PSO algorithm on obtaining a circular GEO-energy orbit. This was accomplished by setting $e_{desired} = 0$ and $c_1 = 1,\ c_2 = c_3 = 0$ in Equation 6.4.1. Therefore, the fitness function for Study A becomes

$$J(\chi) = e_{GEO}(\chi). \tag{7.1.1}$$

A circular GEO orbit is the most likely orbit attainable by low-thrust technology for its given specific orbital energy. Study A focuses on the exploration of the solution space in an effort to identify all GEO-energy orbits that have a low eccentricity ($e_{GEO} \leq 0.01$). It is important to prove that the PSO method is able to find a large number of circular GEO-energy orbits because any usable fuel-optimal solutions must be a subset of the acceptable solution space found in Study A.

7.1.2 Data

Numerous trials were conducted using the cost function found in Equation 7.1.1 with different amounts of particles, iterations, and function evaluations per trajectory k_{fixed}. In total, four trials were conducted and directly compared to the stochastic Monte Carlo method. The results are summarized in Table 7.1 and Table 7.2.

Table 7.1 compares the PSO method with the Monte Carlo method via function evaluations. Note that the $^{+}/_{-}$ signs correspond to W^{s+} and W^{s-}, respectively. Function evaluations are defined as the number of patch points visited per trajectory k_{fixed}. Note that an entire low-thrust trajectory is computed for each patch point that is visited by either method. For example, the "300 Samples +" trial of the Monte Carlo method will randomly sample 300 patch points on each trajectory of W^{s+} for a total of 300 function evaluations per k_{fixed}. The table then reports the optimal value of the cost function, Equation 7.1.1, the $\tau_{s.m.}$ and k of that value, and the amount of time it took for the computer to complete the trial. Likewise, the same can be said for the PSO method. The "30 Particles, 10 Iterations +" trial samples a total of 300 patch points on each trajectory of W^{s+} for a total of 300 function evaluations per k_{fixed}. The remaining columns are analogous to the Monte Carlo method. Note that it is fair to directly compare the PSO method to the Monte Carlo method given the same number of function evaluations. Looking at Table 7.1, one may note that the PSO method usually finds a better optimal solution than the Monte Carlo method regardless of the number of function evaluations used by both methods.

Table 7.1: *Top*: Results from eight Monte Carlo trials. Both the negative perturbation (-) and positive perturbation (+) are investigated. *Bottom*: Results from eight PSO trials. Both perturbations are also investigated. The optimal case is shown in bold.

Monte Carlo Trial	Optimal $J = e_{GEO}$	$\tau_{s.m.}$ $[tu]$	$k^{(\pm)}$	CPU Time (hrs.)
300 Samples +	0.012573	-19.050677	764^{+}	23.0
300 Samples -	0.012573	-19.050677	764^{-}	23.0
160 Samples +	**0.000930**	**-18.101890**	$\mathbf{610^{+}}$	**9.1**
160 Samples -	0.001960	-19.034836	763^{-}	9.9
80 Samples +	0.002907	-26.353585	607^{+}	4.6
80 Samples -	0.002931	-32.545428	610^{-}	5.0
40 Samples +	0.001741	-19.935733	141^{+}	2.6
40 Samples -	0.005998	-19.996901	239^{-}	2.5

PSO Trial	Optimal $J = e_{GEO}$	$\tau_{s.m.}$ $[tu]$	$k\,(\pm)$	CPU Time (hrs.)
30 Particles, 10 Iterations +	0.001741	-19.938450	141^{+}	23.0
30 Particles, 10 Iterations -	0.001960	-19.030223	763^{-}	23.0
16 Particles, 10 Iterations +	**0.000930**	**-18.099478**	$\mathbf{610^{+}}$	**9.5**
16 Particles, 10 Iterations -	0.001960	-19.033287	763^{-}	9.3
8 Particles, 10 Iterations +	0.003136	-21.995468	618^{+}	4.8
8 Particles, 10 Iterations -	0.002907	-26.348258	607^{-}	4.7
8 Particles, 5 Iterations +	0.000930	-18.095750	610^{+}	2.4
8 Particles, 5 Iterations -	0.004073	-35.545405	503^{-}	2.6

To further illustrate this point, refer to Table Table 7.2. The second and third column of this table identifies the number of patch points with $e_{GEO} \leq 0.01$ for the PSO and Monte Carlo method, respectively. Note that for a given number of function evaluations and perturbation (i.e. $^{+}/_{-}$), the PSO method significantly outperforms the Monte Carlo method in the number of useful patch points that

it has identified. The fourth column indicates the ratio of the values of column two to column three; or the ratio of useful patch points identified by the PSO vs. that of the Monte Carlo method. Finally, the fifth column identifies the percentage of trajectories within $W^{s(\pm)}$ that obtained a better optimum value using the PSO method vs. the Monte Carlo method. The sixth column is simply the complement of the fifth. Note that the PSO method significantly outperformed the Monte Carlo method in terms of the number of suitable patch points it was able to identify according to both measures.

Table 7.2: Comparison of PSO vs. Monte Carlo (MC) methods. Both the negative perturbation (-) and positive perturbation (+) are investigated. The PSO and MC columns indicates the number of solutions found with an eccentricity less than 0.01 using the Particle Swarm Optimization and Monte Carlo methods, respectively.

Function Evaluations	PSO	MC	PSO/MC	% Dominated by PSO	% Dominated by MC
300+	948	17	55.8	94.4	5.6
300-	740	14	52.8	96.9	3.1
160+	553	16	34.5	90.7	9.3
160-	280	9	31.1	90.3	9.7
80+	162	4	10.5	80.1	19.9
80-	104	4	26.0	82.8	17.2
40+	16	3	5.3	71.8	28.2
40-	7	1	7	76.5	23.5

Figure 7.1 illustrates the effectiveness of the PSO method very clearly. The figure plots the percentage of patch points in W^s with $e_{GEO} \leq 0.01$ (as identified by the PSO and Monte Carlo methods) given a set amount of function evaluations. The figure shows that the percentage of suitable patch points identified by the Monte Carlo method was fairly uniform with regard to the number of function evaluations per k_{fixed}. The percentage of suitable patch points identified by the Monte Carlo method is on the order of 0.001% which is fairly low

and is to be expected from a random sampling of the search space. In contrast, the percentage of suitable patch points identified by the PSO method varies between 0.02% and 0.44% and generally increases with more function evaluations. This provides convincing evidence that the PSO method quickly and efficiently identifies suitable patch points far better than random chance would allow. Additionally, it would seem that all data originating from W^{s+} contains more suitable patch points than that from W^{s-} although no clear explanation for this phenomena currently exists.

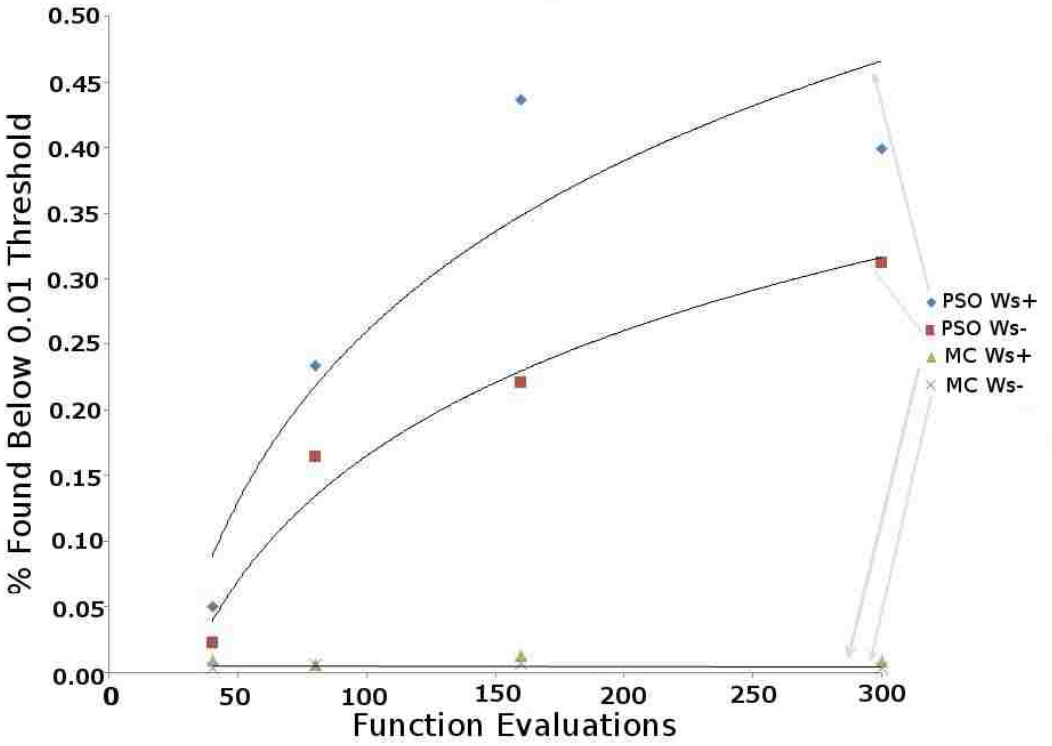


Figure 7.1: PSO vs. Monte Carlo. Percentage of function evaluations that yield an eccentricity less than 0.01. PSO (diamond and squares) clearly outperformed the Monte Carlo (triangles and X's) method. Logarithmic curve fit (PSO) and linear curve fit (MC) are drawn for reference.

Overall, the optimal trajectory found has an eccentricity of $e_{GEO} = 0.0093$

and is located at time $\tau_{s.m.} = -18.10189\,[\text{tu}]$ on trajectory number $k = 610^{+}$ of W^{s+} and was identified by both the PSO and Monte Carlo methods. Figure 7.2 displays the cost of each patch point visited by the PSO algorithm on trajectory 610^{+}. Note that there are three local minima and two local maxima. The global minimum of $k = 610^{+}$ occurs at time $-18.10189\,[\text{tu}]$ and is a rather sharp minimum, with the fitness function increasing rapidly from both the left and right sides. Indeed, the approach towards this minimum is so abrupt that traditional, gradient-based solvers may never reach it; instead becoming trapped within the other two local minima. This is not the case, however, with the evolutionary PSO technique. Note that the blue points in Figure 7.2 are the initial guess solutions of the PSO algorithm, while the red points are the solutions of the intermediate iterations, and the green points are the solutions of the final iteration. While the initial points are randomly distributed throughout the search space, the PSO algorithm quickly converges to the global minimum of trajectory $k = 610^{+}$. The majority of the function evaluations (i.e. patch points) are located near the global minimum, which is a much more efficient use of computational resources as opposed to a random distribution with the Monte Carlo method.

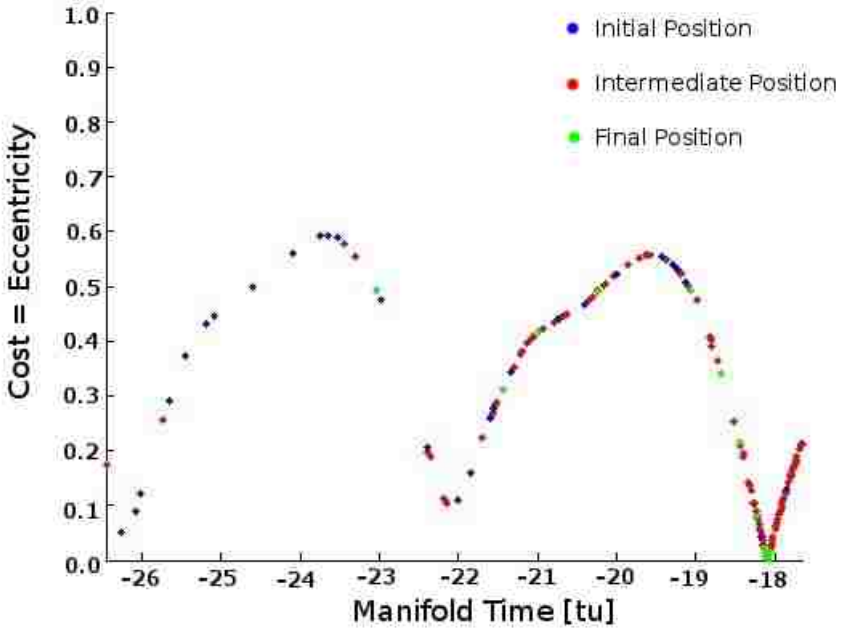


Figure 7.2: Fitness vs. Manifold Time: Fitness (eccentricity) of different locations along trajectory $k = 610^+$ of the positively perturbed stable manifold of the nominal halo orbit. A blue point indicates the location and cost of an initial particle, red points indicate that of a particle during the optimization process, and green points indicate that of a particle at the end of the PSO algorithm.

Figure 7.3 shows two different views of the optimal low-thrust trajectory (red) and its associated trajectory $k = 610^+ \in W^{s+}$ shown in green (the figure includes the entire green trajectory and not just the portion traversed by the spacecraft). The low-thrust spiral begins at the radius of a circular GEO-energy orbit where a blue trajectory indicates the GEO-energy orbit that the spacecraft has originated from. The spacecraft began with an initial mass of $m_0 = 1064.8\,[\text{kg}]$ and has a final mass of $m_f = 1000.0\,[\text{kg}]$ with a mass fraction of $\frac{m_f}{m_0} = 0.939$ at the optimal patch point. The low-thrust spiral was integrated using a maximum thrust of 500 [mN] and an I_{sp} of 3000 seconds. The controlled flight took 10.14 [tu] while the total time of flight took 28.24 [tu] (or 122 days).

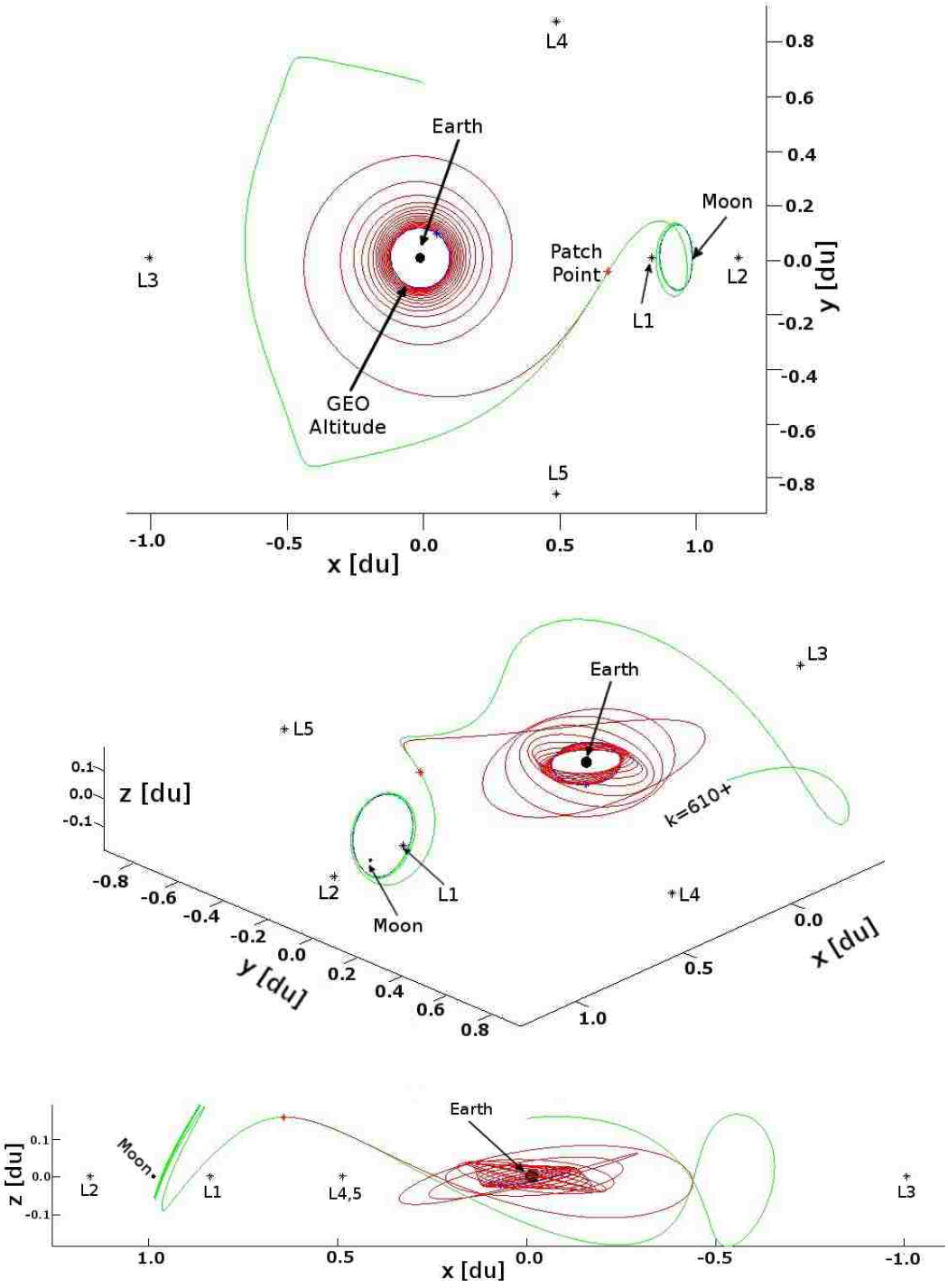


Figure 7.3: Optimal Low-Thrust Trajectory with $e_{GEO} = 0.0093$. Shown with parent trajectory $k = 610^{+}$ as well as the nominal halo orbit. Three separate views of this trajectory are given in order to show perspective.

7.1.3 Optimization of an Arbitrary k_{fixed}

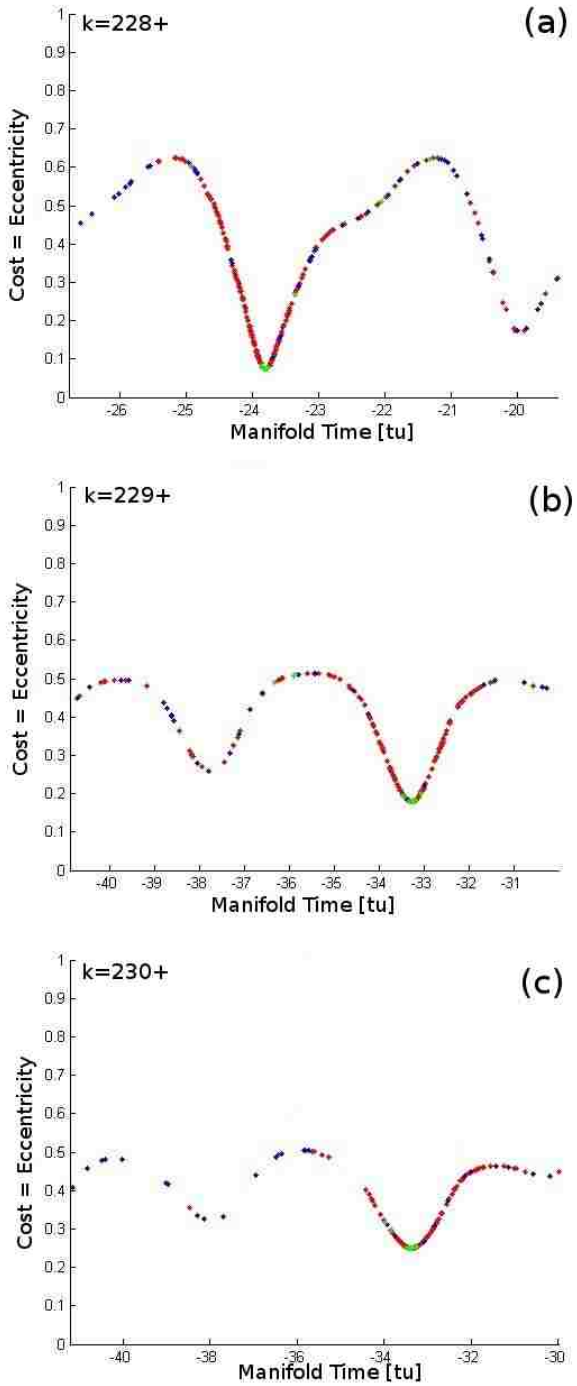
Table 7.3 illustrates the optimization of nine independent trajectories of W^{s+}, namely $k_{fixed} = 228^+$ through 233^+. Note that many trajectories have relatively high values of e_{GEO} as well as high values of the optimal e_{GEO} for that trajectory.

Table 7.3: Optimal patch points found for nine trajectories (228^+-233^+) using PSO.

k_{fixed}	Optimal $\tau_{s.m.}$ $[tu]$	Optimal $J = e_{GEO}$
228^+	-23.799467	0.073514
229^+	-33.269355	0.179901
230^+	-33.367848	0.250249
231^+	-33.545388	0.299980
232^+	-30.501083	0.318315
233^+	-23.892618	0.007784

Figure 7.4 represents the fitness/eccentricity generated from each patch point visited by the PSO algorithm for $k_{fixed} = 228^+ - 233^+$. The figure is broken into six plots; one for each trajectory. A blue point indicates the location and cost of an initial particle, red points indicate the location and cost of a particle during the optimization process, and green points represent the final location and cost of a particle at the end of the PSO algorithm. Note that the structure of the cost functions changes significantly between successive values of k_{fixed}. Some functions have large fluctuations with low minima and high maxima while others, such as 232^+ have far less variability. All trajectories display a semi-cyclic nature with periodic extrema that occur in a semi-periodic manner. While the magnitude of each extrema can vary significantly, some can have very similar magnitudes such as the local minima found in trajectory 233^+. Also note that this simple fitness function yielded a highly non-convex (nor concave) cost function that makes it difficult, if not impossible, for more

traditional convex optimization methods to be used.



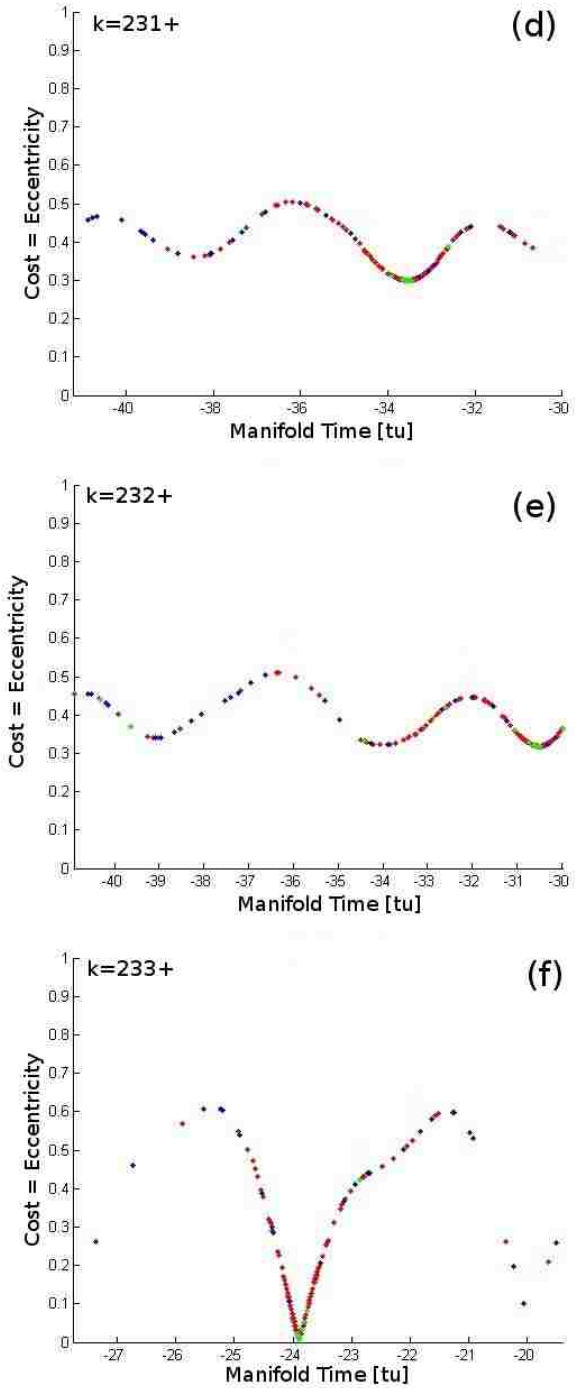


Figure 7.4: Cost Map of Trajectory 228-233; 228-Top; 233-Bottom. A blue point indicates the location and cost of an initial particle, red points indicate that of a particle during the optimization process, and green points indicate that of a particle at the end of the PSO algorithm. Note how the result of the fitness function varies dramatically between adjacent trajectories (a) - (f). This demonstrates the discontinuous nature of the cost function if viewed as a function of trajectory number.

7.2 Study B: Eccentricity and Fuel Optimizing Fitness Function

7.2.1 Fitness Function

The fitness function in Study B becomes slightly more complex with the addition of the fuel-optimization constraint $c_2 \neq 0$. Therefore the fitness function becomes

$$J\left(\mathbf{X}_{\mathbf{s.m.}}\right) = c_1 e_{GEO}\left(\mathbf{X}_{\mathbf{s.m.}}\right) + c_2 \Delta m\left(\mathbf{X}_{\mathbf{s.m.}}\right) \tag{7.2.1}$$

with the values $c_1 = 1$ and $c_2 = 10^{-3}$ in accordance with Equation 6.4.3. This choice of weighting constants ensures convergence on trajectories with very low eccentricity values for their initial orbits, while simultaneously minimizing fuel consumption.

7.2.2 Data

Numerous trials were conducted using the cost function found in Equation 7.2.1 with different amounts of particles, iterations, and function evaluations per trajectory k_{fixed}. In total, four trials were conducted and directly compared to the stochastic Monte Carlo method. The results are summarized in Table 7.4 and Table 7.5.

Table 7.4: *Top*: Results from eight Monte Carlo trials. Both the negative perturbation (-) and positive perturbation (+) are investigated. *Bottom*: Results from eight PSO trials. Both perturbations are also investigated. The optimal case is shown in bold. A 3GHz Core 2 Duo processor was used for the computations.

Monte Carlo Trial	Opt. e_{GEO}	Opt. $\triangle m\ [kg]$	Opt. J	$\tau_{s.m.}\ [tu]$	$k^{(\pm)}$	CPU Time [hrs.]
300 Samples +	**0.000930**	**64.771**	**0.065701**	**-18.0903**	**610^{+}**	**19.2**
300 Samples -	0.002931	64.759	0.067680	-32.5446	610^{-}	18.8
160 Samples +	0.001741	64.852	0.066594	-19.9362	141^{+}	9.2
160 Samples -	0.001960	64.851	0.066811	-19.0354	763^{-}	9.9
80 Samples +	**0.000930**	**64.771**	**0.065701**	**-18.1020**	**610^{+}**	**5.0**
80 Samples -	0.002936	64.749	0.067684	-20.0128	239^{-}	6.4
40 Samples +	0.010044	64.794	0.074838	-20.5978	550^{+}	2.6
40 Samples -	0.009919	64.757	0.074676	-23.0817	760^{-}	3.2

PSO Trial	Opt. e_{GEO}	Opt. $\triangle m\ [kg]$	Opt. J	$\tau_{s.m.}\ [tu]$	$k^{(\pm)}$	CPU Time [hrs.]
30 Particles, 10 Iterations +	**0.000930**	**64.771**	**0.065701**	**-18.0997**	**610^{+}**	**23.3**
30 Particles, 10 Iterations -	0.002931	64.749	0.067680	-32.5417	610^{-}	22.3
16 Particles, 10 Iterations +	**0.000930**	**64.771**	**0.065701**	**-18.0995**	**610^{+}**	**9.5**
16 Particles, 10 Iterations -	0.001960	64.851	0.066811	-19.0363	763^{-}	11.0
8 Particles, 10 Iterations +	**0.000930**	**64.771**	**0.065701**	**-18.0951**	**610^{+}**	**4.6**
8 Particles, 10 Iterations -	0.002931	64.749	0.067680	-32.5474	610^{-}	4.6
8 Particles, 5 Iterations +	**0.000930**	**64.771**	**0.065701**	**-18.0961**	**610^{+}**	**2.4**
8 Particles, 5 Iterations -	0.004526	64.920	0.069446	-30.6836	148^{-}	2.3

Table 7.5: Comparison of PSO vs. Monte Carlo (MC) methods. Both the negative perturbation (-) and positive perturbation (+) are investigated. The PSO and MC column indicates the number of solutions found with a fitness function less than $J = 0.1$ using the Particle Swarm Optimization and Monte Carlo methods, respectively.

Function Evaluations	PSO	MC	PSO/MC	% Dominated by PSO	% Dominated by MC
300+	5237	244	21.46	93.1	6.9
300-	3776	168	22.5	96.5	3.5
160+	2584	120	21.53	76.2	23.8
160-	1496	86	17.4	89.9	10.1
80+	1159	64	18.1	81.7	18.3
80-	860	59	14.5	74.3	25.7
40+	189	28	6.7	76.1	23.9
40-	92	25	3.7	77.8	22.2

Table 7.4 compares the PSO method with the Monte Carlo method via function evaluations and is organized in much the same way as Table 7.1 of Study A. The only new addition is the third column corresponding to the optimal amount of propellant used during the low-thrust maneuver. Based on the results of Table 7.4 it can be clearly seen that the PSO method matched or outperformed the Monte Carlo method in every case. Note that the PSO method clearly dominates the Monte Carlo method for a low number of function evaluations; such as 40 Monte Carlo samples vs. eight particles and five iterations via the PSO method. Table 7.5 further demonstrates the superiority of the PSO method. The table records the number of patch points with $J \leq 0.1$ which are considered to be suitable candidates for further study. Note that the PSO method significantly outperformed the Monte Carlo method just as was observed in Study A.

The globally optimal patch point found in this study occurred on trajectory $k = 610^{+}$with a total propellant consumption of $\triangle m = 64.771\,kg$, an eccentric-

ity of $e_{GEO} = 0.000930$, and a fitness value of $J = 0.065701$. Coincidentally, this also happens to be the globally optimal patch point found in Study A and is graphed in Figure 7.3. Figure 7.5 plots the propellant consumption verses eccentricity of the patch points used in Study B. Note that the maximum propellant used by any point is around $74\,kg$ and is roughly 15% more propellant than the globally optimal value found in Study B. The figure also suggests that the globally minimal propellant usage does not correspond to an eccentricity of zero but closer to $e_{GEO} = 0.25$ which may lower the propellant used to only $63.7\,kg$. Finally, at nearly zero eccentricity the range of feasible propellant consumption is very narrow, differing by less than a kilogram. This suggests that optimization with respect to propellant may not be the most fluid parameter, given a zero eccentricity restriction, and optimization with respect to Time of Flight (TOF) should be given more consideration.

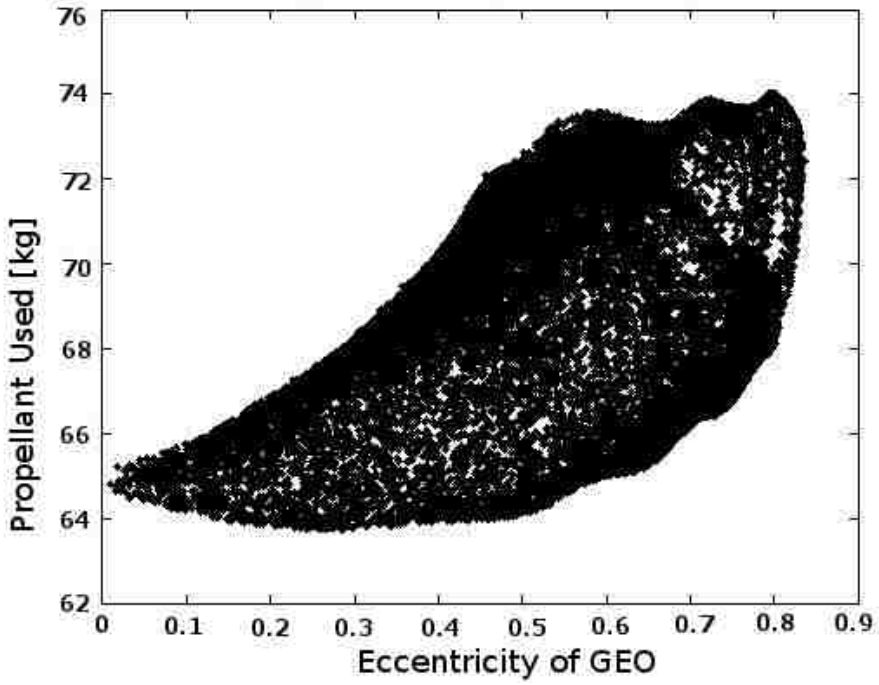


Figure 7.5: Propellant consumption vs. eccentricity for patch points generated in Study B.

7.3 Study C: Eccentricity, Propellant, and Time of Flight Optimizing Fitness Function

7.3.1 Fitness Function

The fitness function in Study C becomes slightly more complex with the addition of the Time-of-Flight-optimization constraint $c_3 \neq 0$. Therefore the fitness function becomes

$$J\left(\mathbf{X}_{\mathbf{s.m.}}\right) = c_1 e_{GEO}\left(\mathbf{X}_{\mathbf{s.m.}}\right) + c_2 \Delta m\left(\mathbf{X}_{\mathbf{s.m.}}\right) + c_3 \Delta T\left(\mathbf{X}_{\mathbf{s.m.}}\right) \tag{7.3.1}$$

with the values $c_1 = 1$, $c_2 = 10^{-3}$, and $c_3 = 10^{-4}$ in accordance with Equations 6.4.3 and 6.4.2. This choice of weighting constants ensures convergence on trajectories with very low eccentricity values for their initial orbits while simultaneously minimizing fuel consumption and TOF.

7.3.2 Data

Numerous trials were conducted using the fitness function found in Equation 7.3.1 with different amounts of particles, iterations, and function evaluations per trajectory k_{fixed}. In total, four trials were conducted and directly compared to the stochastic Monte Carlo method. The results are summarized in Table 7.6 and Table 7.7.

Table 7.6: *Top*: Results from eight Monte Carlo trials. Both the negative perturbation (-) and positive perturbation (+) are investigated. *Bottom*: Results from eight PSO trials. Both perturbations are also investigated. The optimal case is shown in bold and has been independently identified twice by two separate trials with differing amounts of particles. This demonstrates the robustness of the PSO method.

Monte Carlo Trial	Opt. e_{GEO}	Opt. $\triangle m\,[Kg]$	Opt. TOF $[tu]$	Opt. J	$\tau_{s.m.}$ $[tu]$	$k^{(\pm)}$	CPU Time [hrs.]
300 Samples +	0.001926	64.666	39.40	0.070533	-29.2770	743^{+}	16.8
300 Samples -	0.001960	64.851	29.18	0.069729	-19.0247	763^{-}	18.8
160 Samples +	0.003221	64.773	54.42	0.073436	-44.2802	653^{+}	10.4
160 Samples -	0.002936	64.749	30.14	0.070699	-20.0033	239^{-}	9.9
80 Samples +	0.006493	64.731	30.31	0.074255	-20.1688	428^{+}	5.5
80 Samples -	0.005365	65.026	51.42	0.075533	-41.2405	452^{-}	6.3
40 Samples +	0.005513	64.750	40.72	0.074335	-30.5743	237^{+}	2.6
40 Samples -	0.004217	64.901	47.32	0.073850	-37.1541	640^{-}	3.2

PSO Trial	Opt. e_{GEO}	Opt. $\triangle m\,[Kg]$	Opt. TOF $[tu]$	Opt. J	$\tau_{s.m.}$ $[tu]$	$k^{(\pm)}$	CPU Time [hrs.]
30 Particles, 10 Iterations +	**0.000930**	**64.771**	**28.23**	**0.068525**	**-18.0903**	**610^{+}**	**15.4**
30 Particles, 10 Iterations -	0.001960	64.851	29.18	0.069729	-19.0240	763^{-}	16.8
16 Particles, 10 Iterations +	**0.000930**	**64.771**	**28.23**	**0.068525**	**-18.0909**	**610^{+}**	**8.5**
16 Particles, 10 Iterations -	0.001960	64.851	29.18	0.069729	-19.0262	763^{-}	11.1
8 Particles, 10 Iterations +	0.001741	64.852	30.09	0.069603	-19.9342	141^{+}	6.1
8 Particles, 10 Iterations -	0.002936	64.749	30.14	0.070699	-20.0032	239^{-}	5.6
8 Particles, 5 Iterations +	0.001741	64.852	30.10	0.069604	-19.9450	141^{+}	2.4
8 Particles, 5 Iterations -	0.004526	64.920	40.85	0.073531	-30.6827	148^{-}	3.0

Table 7.7: Comparison of PSO vs. Monte Carlo (MC) methods. Both the negative perturbation (-) and positive perturbation (+) are investigated. The PSO and MC column indicates the number of solutions found with a fitness function less than $J = 0.1$ using the Particle Swarm Optimization and Monte Carlo methods, respectively.

Function Evaluations	PSO	MC	PSO/MC	% Dominated by PSO	% Dominated by MC
300+	4750	211	22.51	92.4	7.6
300-	3685	146	25.24	95.1	4.9
160+	2325	92	25.27	87.5	12.5
160-	1383	72	19.21	89.4	10.6
80+	879	59	14.89	78.8	21.2
80-	626	40	15.65	81.6	18.4
40+	184	22	8.36	70.2	29.8
40-	89	13	6.84	74.9	25.1

Table 7.6 compares the PSO method with the Monte Carlo method via function evaluations and is organized in much the same way as Table 7.1 of Studies A and B. The only new addition is a column corresponding to the optimal spacecraft Time of Flight (TOF). Based on the results of Table 7.6 it can be clearly seen that the PSO method outperformed the Monte Carlo method in nearly every case. Table 7.7 further demonstrates the superiority of the PSO method. The table records the number of patch points with $J \leq 0.1$ which are considered to be suitable candidates for further study. Note that the PSO method significantly outperformed the Monte Carlo method just as was observed in Studies A and B.

The globally optimal patch point found in this study occurred on trajectory $k = 610^{+}$with a total propellant consumption of $\triangle m = 64.771\,kg$, an eccentricity of $e_{GEO} = 0.000930$, TOF of 28.23 [tu], and a fitness value of $J = 0.065701$. Coincidentally, this also happens to be the globally optimal patch point found in Studies A and B and is graphed in Figure 7.3. Figure 7.6 plots the TOF versus eccentricity of the patch points used in Study C. Note the cyclical nature

of TOF at zero eccentricity. This suggests that the TOF is in great need of optimization and has, indeed, been accomplished in Study C.

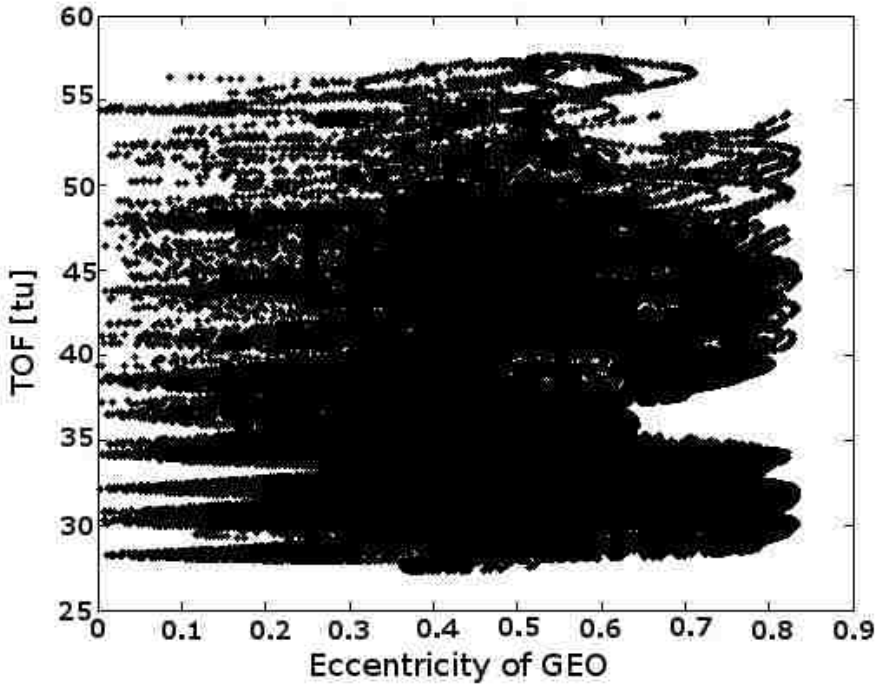


Figure 7.6: Plot of Time of Flight (TOF) vs. eccentricity for patch points generated in Study C.

This concludes the one-dimensional study of Particle Swarm Optimization. Conclusions of this research can be found in Section 10.1. The next chapter focuses on the generalization of the technique found in this chapter by allowing each particle to freely move in two dimensions instead of only one. This is accomplished by allowing the parameter, k, to be a variable (or a degree of freedom) within the PSO method. In this case, the searchspace becomes the entire $k\tau$-plane instead of τ only. This has the effect of increasing the degrees of freedom of the searchspace and results in a significant decrease in CPU time.

8 Application of Two-Dimensional PSO

A low-thrust trajectory from a geosynchronous Earth orbit (or close to it) to an Earth-Moon, L_1, northern halo orbit is optimized via a two-dimensional Particle Swarm Optimization (PSO) algorithm in this chapter. Note that some content for this chapter was taken from a publication by Abraham et al. [75] and reproduced with the author's and publisher's consent. In contrast with Chapter 7, this study allows both parameter k and parameter, τ, to simultaneously vary as the PSO algorithm runs. The shape of the fitness function is first discussed followed by the sizing considerations of the swarm and the number of iterations of the PSO algorithm to perform. A convergence parameter is then defined and used as an optional exiting flag to prevent unproductive CPU resource utilization. Finally, varying parameters of the "local" version of the PSO algorithm is tested and the influence on the optimal results is discussed.

The search-space and destination Lagrange point orbit are based on what is developed in Chapter 5 while the cost function and PSO algorithm are taken from Chapter 6. An individual point in the search-space (manifold) is evaluated using the control law defined by Equation 6.2.1 and by propagating, backwards in time, from that point using Equation 2.2.31. These equations of motion are

propagated until the Jacobi energy (Equation 2.3.5) of the spacecraft matches that of a spacecraft in Geosynchronous Earth Orbit (GEO). Once the integration terminates, the fitness function is evaluated based on the final conditions of the propagated trajectory (in geocentric orbit). A "cost" is assigned, using this method, to the manifold patch point under evaluation. This process is repeated for each location visited by an individual particle in the swarm. When the maximum number of iterations are reached the PSO method terminates and the optimized result is obtained.

8.1 Shape of the Fitness Function

The primary motivation for the application of evolutionary algorithms, in this study, is the lack of convexity of the fitness function. Figure 8.1 was generated via $150,000$ random samples of the search space (with a 24 hour CPU runtime). The image on the top shows the raw data while the image on the bottom illustrates a curve fit using two dimensional, cubic interpolation from MATLAB's "Curve Fit" toolbox. While this fit is too crude to use in the optimization algorithm (and takes far too long to generate) it is helpful in visualizing the complexity of the fitness function. This function has a very large number of local maxima and minima with a wide variation in their values. To make matters worse, many of the extrema's values differ significantly with other points in their immediate vicinity. This yields nearly infinite gradients in the vicinity of these points and could lead astray even the most robust of gradient based optimization routines unless a near-perfect initial guess is given. Finally, a few of the best local minima have nearly identical values, making the consistent identification of the global minimum extremely difficult.

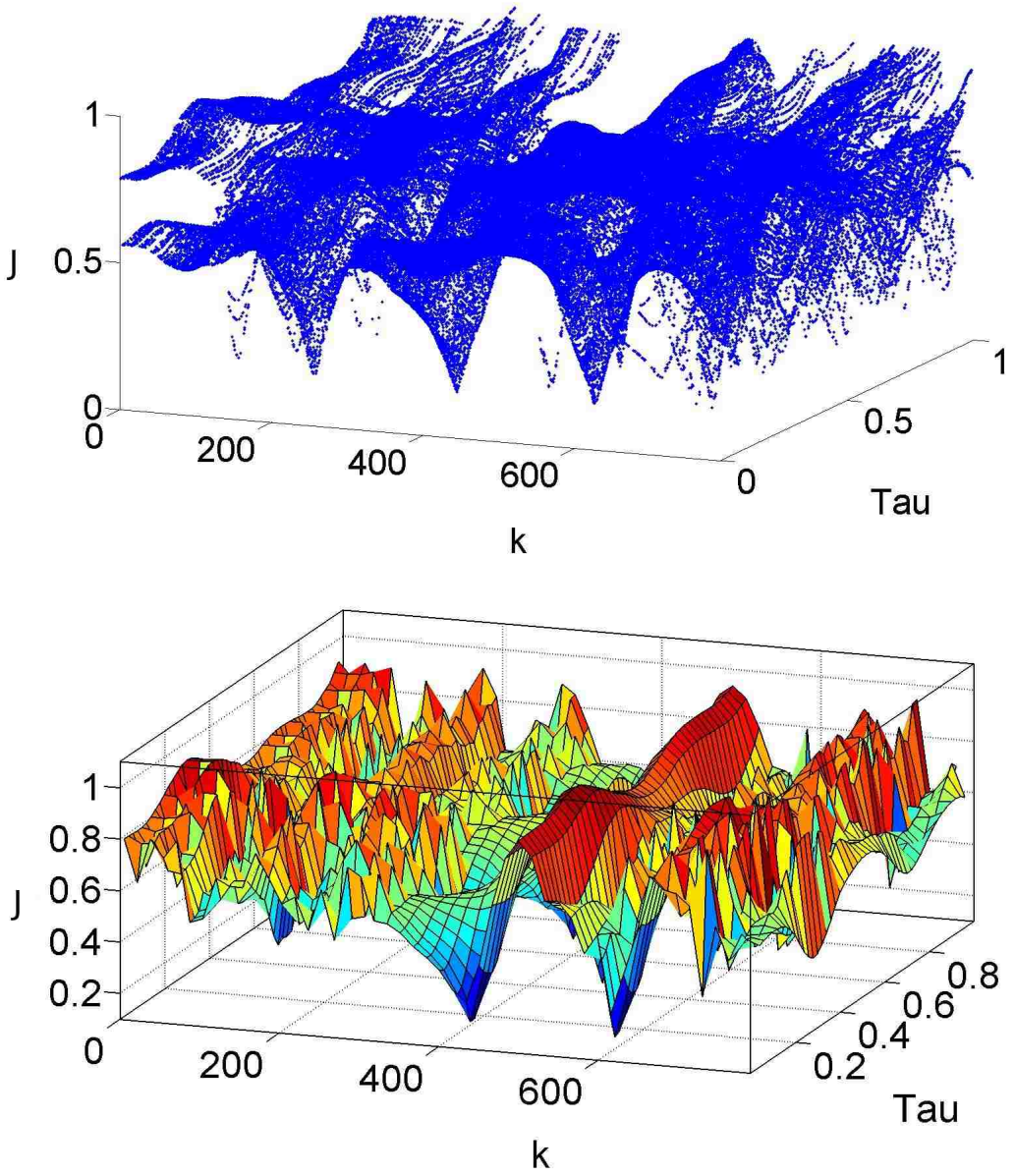


Figure 8.1: *Top*: The fitness function is plotted in $k\tau$-space using roughly $150,000$ samples generated over a 24-hour Monte Carlo trial. *Bottom*: A curve fit of this data was generated via MATLAB's "Curve Fit" toolbox to illustrate the non-convexity and irregular behavior of the fitness function.

8.2 Sizing N_p

The PSO algorithm was run multiple times with $\mathbf{r}_{local} = \left[\frac{1}{20}, \frac{1}{16}N \right]^T$ and $j_{max} = 15$. Each data point in Table 8.1 was averaged over 10 identical trials with only the difference being the initial $\boldsymbol{\chi}$ and $\boldsymbol{\omega}$ values that are chosen randomly. Referring to Figure 8.2, one will note that the average value of the cost function, J_{best} decreases as a function of N_p, as expected. Initially, the decrease is substantial but the amount of increase diminishes as each additional particle is added to the system – especially for values of $N_p > 400$. Based on this data, it seems reasonable to conclude that the ideal range of N_p lies somewhere between 200 and 400 particles. A PSO algorithm with less particles experiences a huge amount of variability and offers a relatively poor average J_{best} (i.e. the $N_p = 50$ data point). Conversely, a PSO algorithm utilizing more than 400 particles will consume copious amounts of CPU time without a significant decrease in the average J_{best} identified.

Table 8.1: Cost, convergence, and CPU time for $j_{max} = 15$ iterations and various number of particles (N_p). Each entry consists of the average of 10 identical trials with error bounds of $\pm 1\sigma$ standard deviation.

N_p	Avg. Best Cost, J_{best}	Avg. Final Convergence, γ_{final}	Avg. CPU Time [min]
50	$0.1357 \pm 8.8\%$	$0.71 \pm 14\%$	$9.0 \pm 6\%$
100	$0.1273 \pm 2.8\%$	$0.69 \pm 7\%$	$20.5 \pm 4\%$
200	$0.1260 \pm 2.8\%$	$0.60 \pm 10\%$	$42.8 \pm 3\%$
400	$0.1235 \pm 2.4\%$	$0.61 \pm 11\%$	$84.4 \pm 3\%$
800	$0.1221 \pm 1.4\%$	$0.61 \pm 11\%$	$160.0 \pm 4\%$

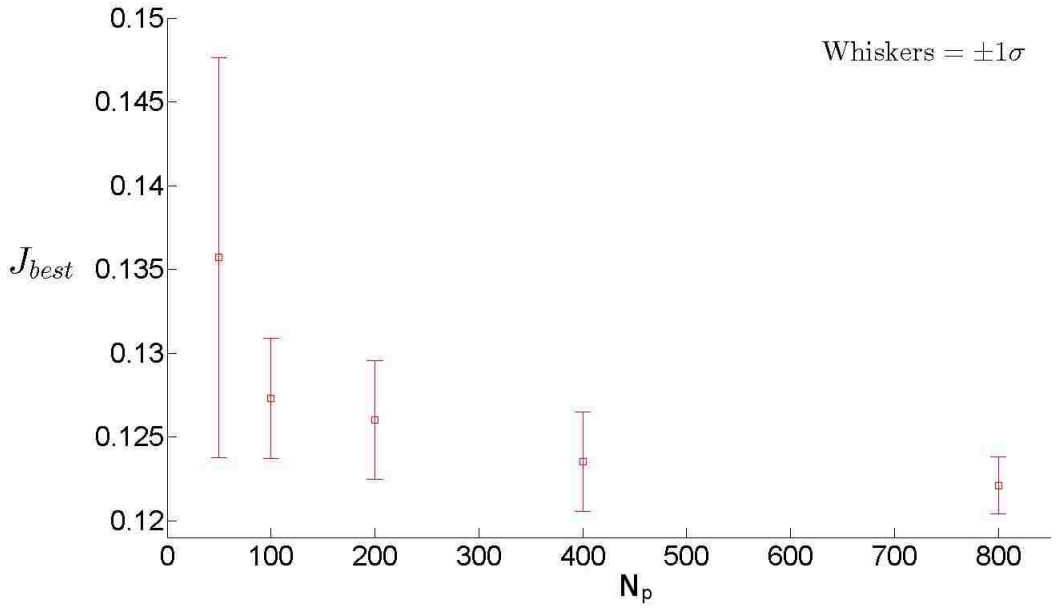


Figure 8.2: Values of J_{best} as a function of N_p for a fixed value of $j_{max} = 15$ iterations. Data taken from Table 8.1.

8.3 Sizing j_{max} via the Convergence Metric, γ

In the previous section, a method for determining the appropriate value of N_p was presented. This section develops a new metric that is used to determine an appropriate number of iterations to use when terminating the PSO algorithm. In general, there are two ways to accomplish this task. The first method is to simply select a value for j_{max} before running the PSO algorithm. An appropriate value of j_{max} is highly problem dependent and relies greatly on the experience and intuition of the researcher [34, 36]. A new metric, γ, is proposed, in this study, to replace j_{max} as the trigger that terminates the PSO algorithm.

The top plot in Figure 8.3 displays convergence (γ) as a function of PSO iteration (j) for a system of 50 particles. Each point is averaged over 10 (identical) data trials and is plotted with $\pm 1\sigma$ error bars. Note that the convergence seems

to initially increase exponentially with low values of j, but then logarithmically approaches an asymptote for large values of j. This seems to indicate that, for large numbers of PSO iterations, an increasingly diminishing rate of marginal convergence is achieved. This relationship has been successfully fitted to the following curve using MATLAB's curve fitting toolbox:

$$c \exp\left(\frac{-b}{(j+a)^3}\right) \tag{8.3.1}$$

with the fitted values of the constants being $a = 7$, $b = 2421$, and $c = 0.8671$ with a "goodness of fit" measured as $R^2 = 0.9957$. In general, $\gamma_{max} \leq c \leq 1$ since the value of Equation 8.3.1 is c when $j = \infty$. Of course, theory dictates that $\gamma_{max} \to 1$ as $j_{max} \to \infty$. In practice, however, this is not always the case, since a small number of particles can become trapped in cycles that oscillate them between the neighborhood of one local minima and another. Also, on occasion, a particle will oscillate for a very long time between a personal best and global best value that are in opposite directions.

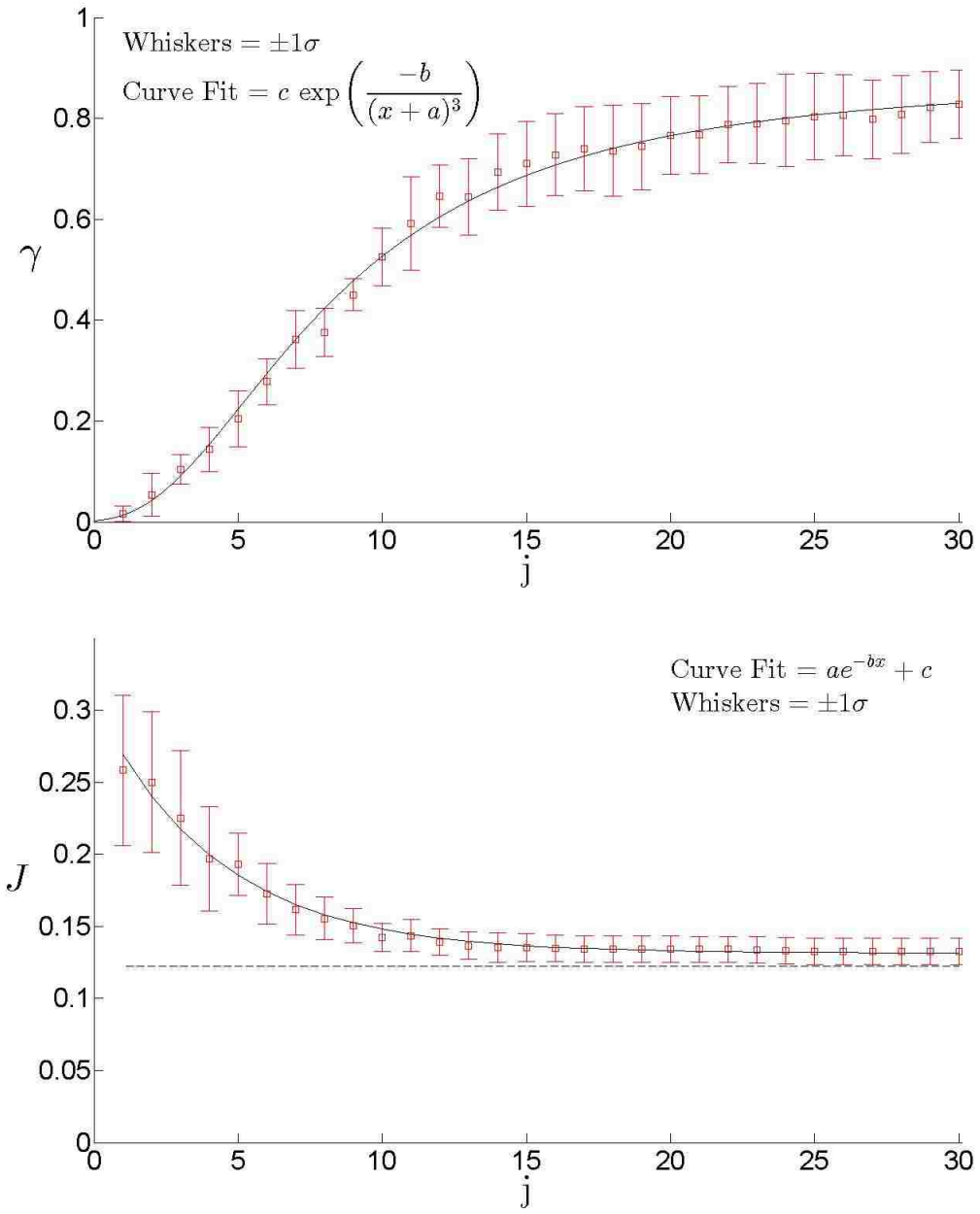


Figure 8.3: *Top*: Convergence (γ) as a function of iteration number (j). *Bottom*: Cost (J) as a function of iteration number (j).

The bottom plot in Figure 8.3 displays the cost (J) as a function of PSO iteration (j) for the same system of 50 particles and 10 trials. The plot describes the evolution of the best value of the cost function. Note the exponential decay

displayed in the cost function and fitted to the following curve:

$$ae^{-bj} + c \tag{8.3.2}$$

where the values of the fitted constants are $a = 0.174$, $b = 0.2344$, and $c = 0.1314$ with $R^2 = 0.9879$. As $j \to \infty$ the limit of Equation 8.3.2

$$\lim_{j\to\infty} \left(ae^{-bj} + c\right) = c \tag{8.3.3}$$

is equal to 0.1314 which is a greater value than $J_{best} = 0.1225$. J_{best} is indicated by the dashed line in the bottom plot of Figure 8.3, where J_{best} represents the lowest value of the cost function encountered by any particle from any of the 10 trials. The difference between J_{best} and c is reflected in the variability of the last few data points in the plot where $J = 0.1327 \pm 0.0092$ $(\pm 1\sigma)$ or a range of $J\,(j = 30)_{range} = [0.1225 \leftrightarrow 0.1520]$.

Based on the data found in Figure 8.3 it seems reasonable to conclude that a value of $\gamma \geq 0.75$ is a suitable choice of terminal convergence to end the PSO algorithm. This roughly correlates to $j = [15 \leftrightarrow 20]$ iterations and places the value of the cost function very near its asymptotic limit without wasting CPU time on additional and unproductive iterations. It, therefore, seems reasonable to use a value of $\gamma_{stop} = 0.8$ as a conservative termination metric for the PSO algorithm. Inserting this value into Equation 8.3.1 and solving for j yields $j_{stop} = 24$ iterations on average.

8.4 Effect of the Product $N_p\, j_{max}$ on the Optimal Solution

Since the most significant issue with PSO is the CPU run-time, and CPU run-time is proportional to the product $N_p\, j_{max}$, it is reasonable to investigate the trade between N_p and j_{max} while holding the number of integrated trajectories (and CPU run-time) fixed. This data is presented in Figure 8.4 which plots J_{best} vs. the log of N_p (semi-log plot was chosen for illustrative purposes) given a fixed value of the product $N_p\, j_{max} = 3000$ evaluations (trajectories integrated according to Equation 2.2.29). Values of N_p included [50, 100, 200, 400, 800]. Note that the lowest value of J_{best} was captured when $N_p = 200$ and $j_{max} = 15$. This roughly agrees with the recommended values of N_p and j_{max} (or γ_{stop}) developed in the two previous sections of this study.

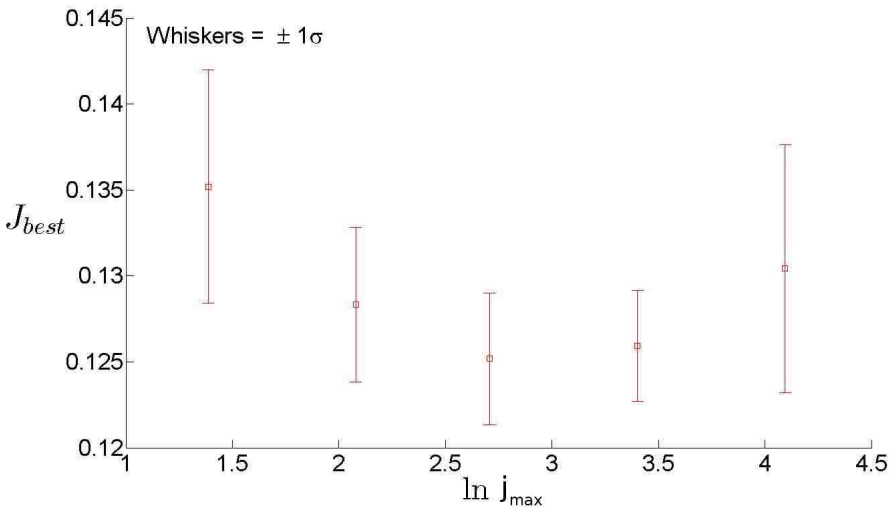


Figure 8.4: Plot of J_{best} as a function of $\ln(j_{max})$ for a fixed value of $3,000$ integrations per point. Each point was averaged over 10 identical trials with $\pm 1\sigma$ error bounds drawn.

8.5 Effect of r_{local} on the Optimal Solution

The effect of the size of $\boldsymbol{r}_{local}$ on the convergence and cost is best captured in Figure 8.5 and Figure 8.6. These plots are produced by multiplying $\boldsymbol{r}_{local}$ by a scaling factor. The scaling factors used in this study are $c = \left[\frac{1}{4}, \frac{1}{2}, 1, 2, 4, 8\right]$ where $\boldsymbol{r}_{resized} = c\boldsymbol{r}_{local}$. Each data point was averaged over 10 identical trials and plotted with $\pm 1\sigma$ error bars using 50 particles and 15 iterations. Note that the convergence is not significantly affected by a change in the magnitude of $\boldsymbol{r}_{local}$, as is evidenced by the fact that any data point lies within the error bars of any of the remaining points. Likewise, the cost typically is not affected by the magnitude of $\boldsymbol{r}_{local}$ except for extremely small values of c, as evidenced by the plot in Figure 8.5; in which case the size of a particle's local range of communication is so small it is effectively "blind and deaf."

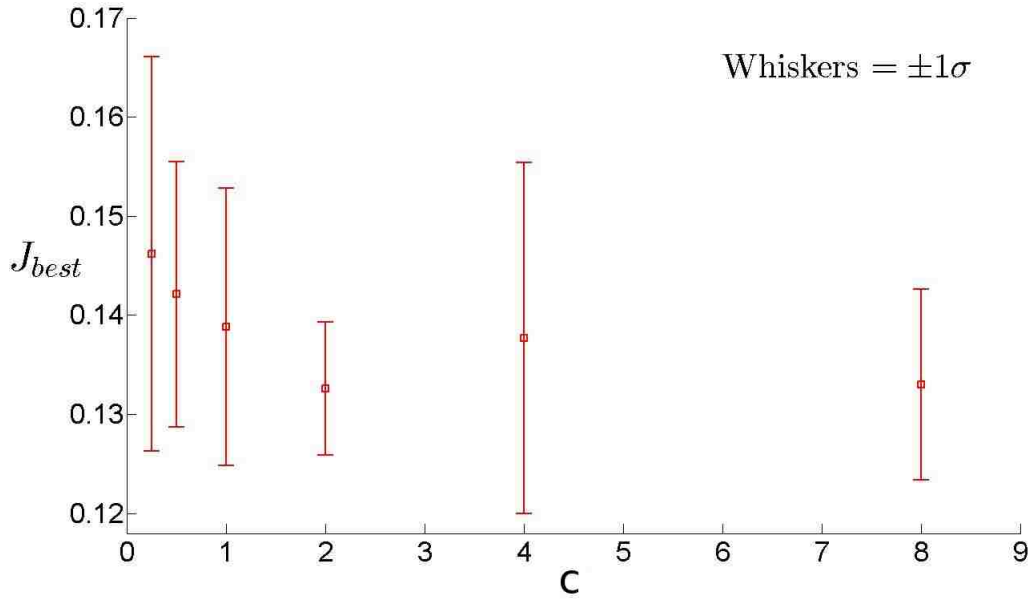


Figure 8.5: Cost as a function of radius.

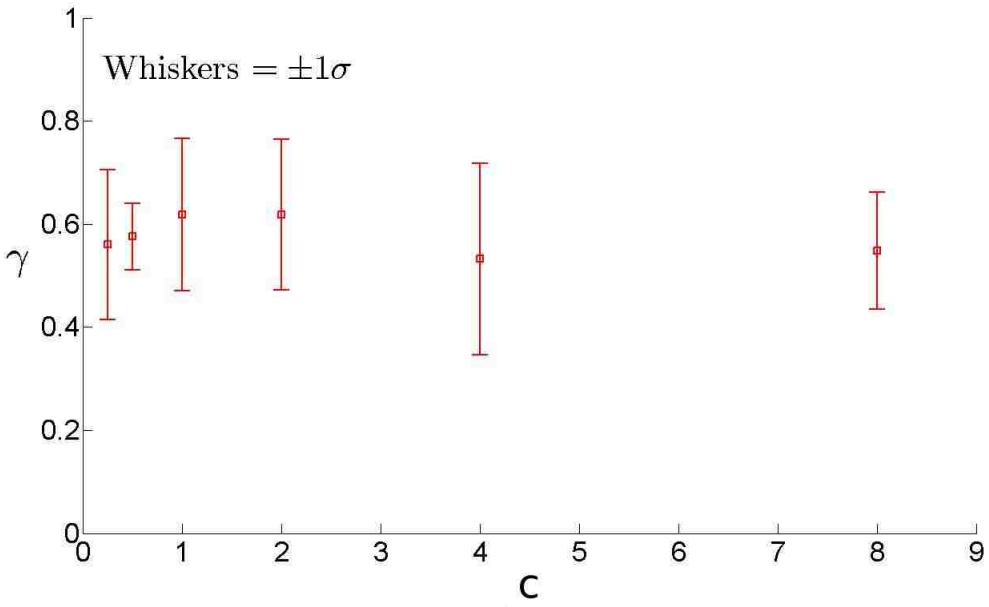


Figure 8.6: Convergence as a function of radius.

Figure 8.7 displays a map of local minima identified by the PSO algorithm as plotted in $k\tau$-space. In this particular example $c = 1$ and seven local minima are identified. A green ellipse of size $\boldsymbol{r}_{local} = \left[\frac{1}{20}, \quad \frac{1}{16}N \right]^T$ is drawn around each local minima to illustrate the communication limit between a particle at the local minima and any nearby particle it can communicate with. This plot illustrates the ability of the local PSO algorithm to simultaneously interrogate multiple local minima instead of being limited to only one minimum using the traditional PSO algorithm.

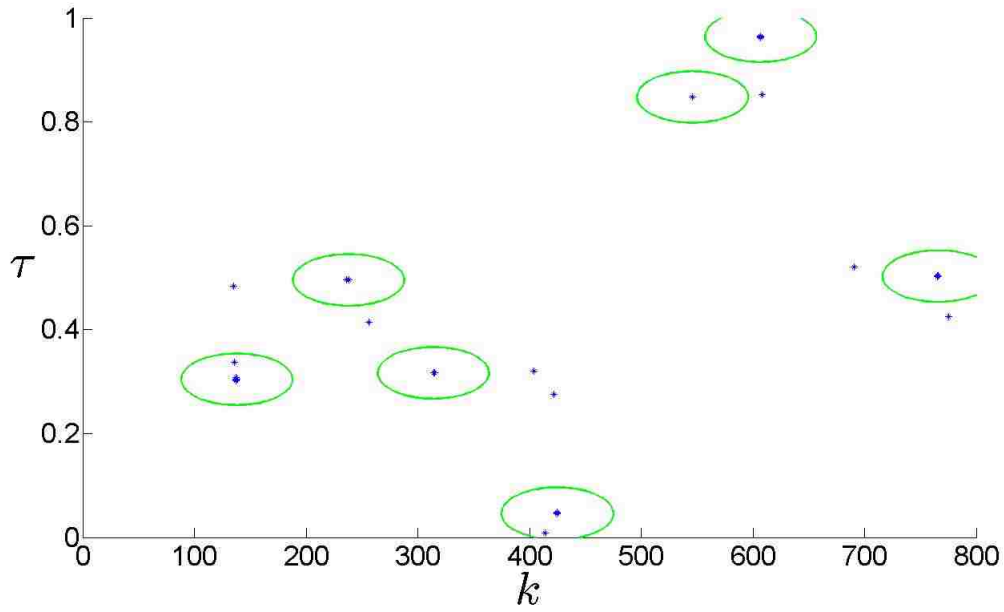


Figure 8.7: A map of the final positions of 50 particles after 15 iterations. The green ellipses are drawn around local swarms of two or more particles indicating a local minimum. The size of the ellipse reflects the dimensions of $\boldsymbol{r}_{local} = \left[\begin{array}{cc} \frac{1}{20}, & \frac{1}{16}N \end{array} \right]^T$.

8.6 Optimal Low-Thrust Trajectory

Mindful of the results of this study's preceding sections, a PSO algorithm was run with the following conditions: $N_p = 50$ particles, $\gamma_{stop} = 0.8$, $\boldsymbol{r}_{local} = \left[\begin{array}{cc} \frac{1}{20}, & \frac{1}{16}N \end{array} \right]^T$, and $j_{max} = 50$ iterations. This setup was repeated over 10 trials to obtain an average cost and standard deviation of $J_{best-median} = 0.1217 \pm 0.7\%$ with a maximum value of $J_{best-max} = 0.1231$ and a minimum value of $J_{best-min} = 0.1201$. Table 8.2 shows a summary of this data with the values of the three terms of the fitness function displayed in addition to the value $j_{discovered}$ and $j_{\gamma=0.8}$. $j_{discovered}$ represents the iteration number when J_{best} was discovered whereas $j_{\gamma=0.8}$ represents the iteration number when $\gamma = \gamma_{stop} = 0.8$.

Note that, on occasion, $\gamma < \gamma_{stop}$ when $j = j_{max}$. In this case, the PSO algorithm terminated before reaching γ_{stop}, as desired.

The two-dimensional PSO algorithm presented in this study was relatively fast in comparison with other published results. Abraham et al. [58] published results that took approximately $8 - 16$ hours to achieve similar values of the fitness function using a 1-D version of the PSO algorithm on a 3GHz Intel Core 2 processor. In this 2-D study, the average CPU run-time was 34 ± 9 minutes ($\pm 1\sigma$) with a maximum value of 46 minutes and a minimum value of 22 minutes using the same processor as with the one-dimensional study. The majority of this variability is due to the fact that the PSO algorithm terminates in as little as $j = 26$ iterations and as many as $j = j_{max} = 50$ iterations. This 2-D PSO algorithm produces an order of magnitude decrease in the CPU run-time when compared with the 1-D result and represents a sizable boost in performance, usability, and practicality.

Table 8.2: Summary of the maximum, minimum, and median values of the fitness function.

	J_{best}	e_{GEO}	$\triangle m$ [kg]	TOF [tu]	$j_{dis.}$	$j_{\gamma=0.8}$	k	τ_{01}
J_{Max}	0.1231	0.0017	64.85	28.27	15	37	141	0.3261
J_{Med}	0.1217	0.0017	64.85	27.57	9	26	141	0.3257
J_{Min}	0.1201	0.0019	64.66	26.73	4	N/A	743	0.0556

The evolution of J and γ as a function of j also agrees well with previous data. Referring to Figure 8.8 one will find a plot of J vs. j for the median case outlined in the center row of Table 8.2. Note that this data does not refute the veracity of Equation 8.3.2 although the variation and limited scope of the data limits a direct comparison. Conversely, γ vs. j in Figure 8.9 illustrates data with only minor variation that agrees well with the form predicted by Equation

8.3.1.

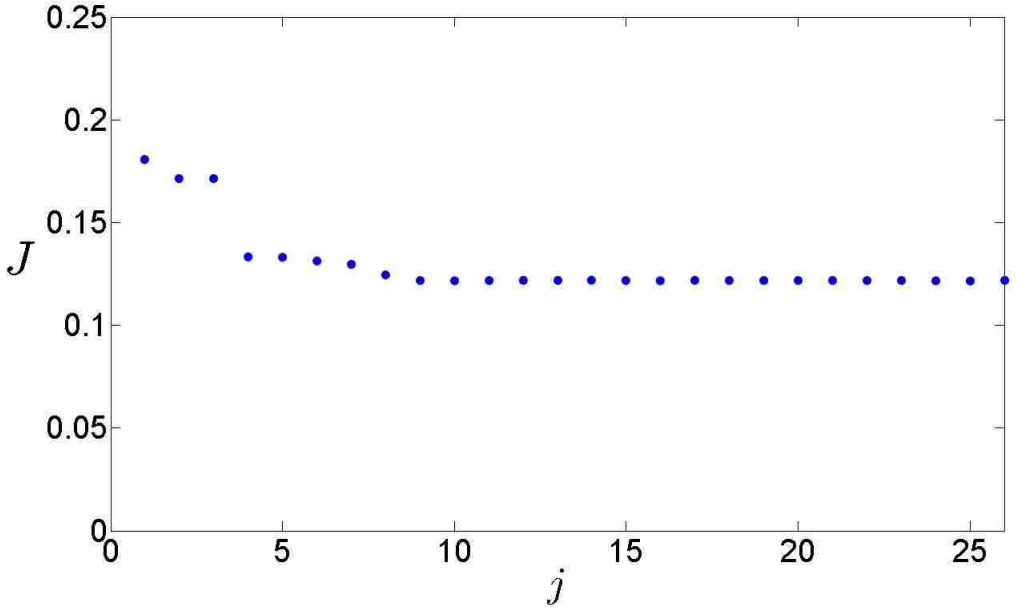


Figure 8.8: Cost vs. iteration number for $J_{best-median}$ case.

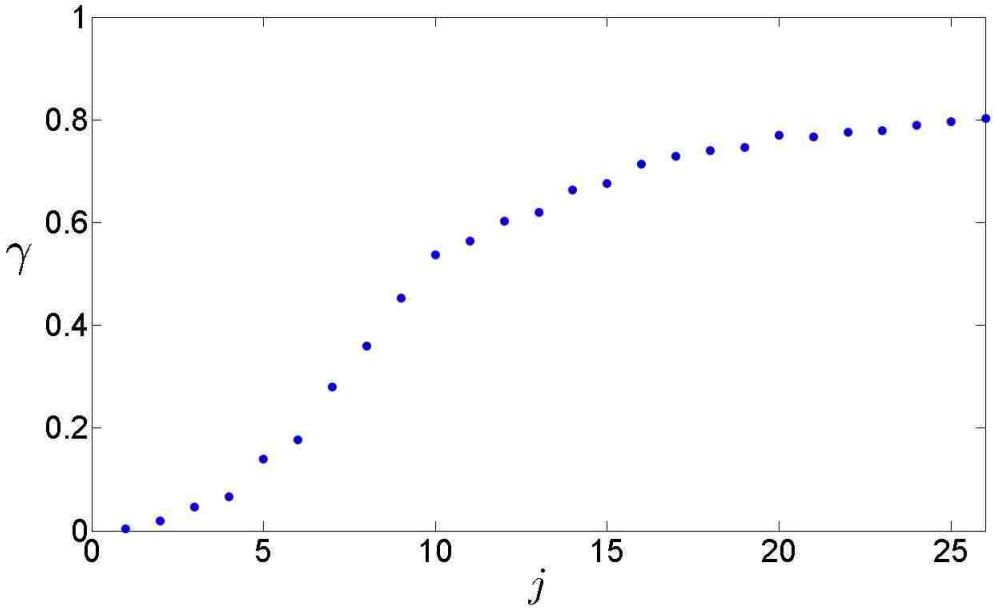


Figure 8.9: Convergence vs. iteration number for $J_{best-median}$ case.

The optimal trajectory corresponding to $J_{best-median}$ has been plotted in Figure 8.10. This trajectory is representative of a typical optimized trajectory discovered by the 2-D PSO algorithm. Note the highly three dimensional nature of this transfer as the spacecraft becomes heavily influence by three-body dynamics. Figure 8.10 is plotted in the synodic, three-body reference system using the traditional non-dimensional units. The spacecraft begins its journey with the geocentric, GEO-energy orbit drawn in blue. The spacecraft then activates its low-thrust engine and tangentially thrusts itself into a spiral pattern of increasing altitude shown in red. The low-thrust spiral is greatly deformed as it enters a region of space that is clearly dominated by three-body effects. The spacecraft terminates this low-thrust arc at the optimized patch point, $\boldsymbol{X}^*_{s.m.}$, which is also a member of the stable manifold (and search space). The spacecraft ballistically coasts along the green trajectory and flows along the stable manifold towards the nominal halo orbit shown in blue. The spacecraft begins its journey with a wet mass of $1,064.85$ [kg] and arrives at the nominal halo orbit 24 months later expending 64.85 [kg] of propellant and delivering 94% of its original mass.

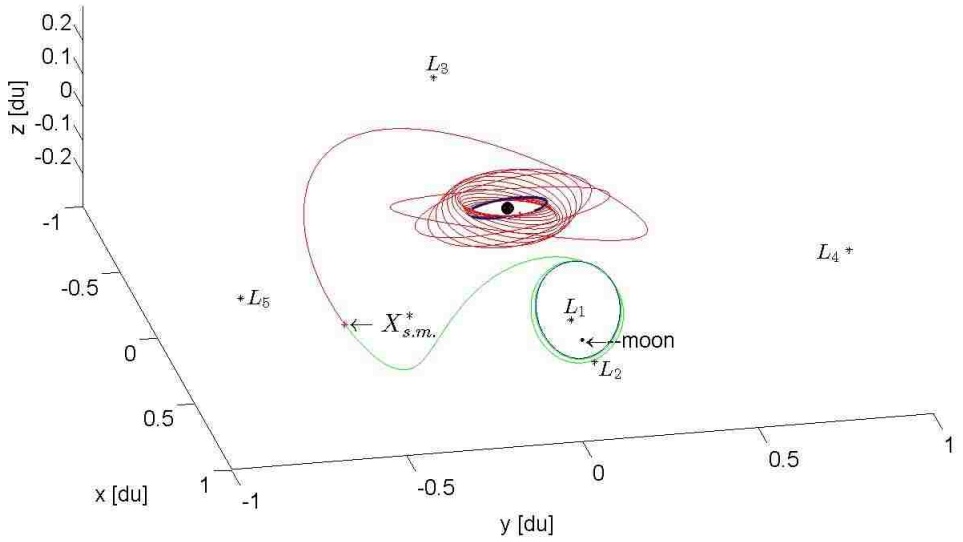


Figure 8.10: Optimal trajectory found using PSO. The low-thrust trajectory is displayed in red, the ballistic coast along the manifold is shown in green, and the geocentric and halo orbits are shown in blue.

This concludes the study found in Chapter 8. Conclusions drawn from this chapter are summarized in Section 10.2. In the next chapter, the PSO method is re-formulated to solve a two-burn, impulsive transfer from a Low Earth Orbit (LEO) to the same LPO discussed in this chapter.

9 Two-Maneuver, Impulsive Transfers via a Hybrid PSO/Shooting Method

The application of a basic, two-dimensional Particle Swarm Optimization (PSO) algorithm attempting to optimize a two-burn, impulsive maneuver trajectory from a Low Earth orbit (LEO) to an Earth-Moon, L_1, northern halo orbit is described in this chapter. Note that some content for this chapter was taken from a publication by Abraham et al. [76] and reproduced with the author's and publisher's consent. In contrast with Chapter 7 and Chapter 8, this chapter is focused on high thrust, impulsive transfers from a geocentric Low Earth Orbit (LEO) to the same Lagrange point orbit constructed in Chapter 5. The cost function and PSO algorithm are loosely based on that found in Chapter 6, but re-designed via Equation 9.2.1 to better reflect the goals of this chapter. The search-space and destination Lagrange point orbit are still based on what is found in Chapter 5. An individual point in the search-space (manifold) is evaluated by propagating, backwards in time, from that point using ballistic Equation 2.2.29. This integration occurs after a velocity discontinuity, representing an impulsive maneuver, is added. A final velocity discontinuity, representing

the LEO departure burn, is added to the end of the propagation to circularize the orbit in a 400 [km] altitude LEO. The fitness function is evaluated based on the final conditions of the propagated trajectory (in geocentric orbit). A "cost" is then assigned to the manifold patch point under consideration. This technique is repeated for each location visited by an individual particle in the swarm, with the swarm terminating its search when the maximum number of iterations has been reached. The method described in this chapter represents a hybrid between a heuristic method (PSO) and gradient based method (shooting).

9.1 Single Shooting a Two-Maneuver, Impulsive Transfer

For a given point on the manifold, a variable time, single shooting algorithm was used to compute the magnitude and direction of two burns: one to exit the LEO and a second to enter the manifold (or LPO). The shooting was conducted in the following manner. The free variable vector

$$\boldsymbol{Y} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \tag{9.1.1}$$

is defined as the pre-burn velocity components of the insertion point into the manifold/LPO. The magnitude of the velocity discontinuity between $\boldsymbol{Y}$ and the velocity components of $\mathbf{X}_{s.m.}$ yield ΔV_{LPO}, or the amount of delta-V required to insert the spacecraft into the manifold/LPO. Note that the units of $\boldsymbol{Y}$ are

[vu] but can easily be converted to [km/s]. The constraint vector is defined as

$$\boldsymbol{F} = \begin{bmatrix} \|r - r_d\| \\ \mathbf{r} \cdot \boldsymbol{v} \end{bmatrix} \tag{9.1.2}$$

where r is the magnitude of the distance between the spacecraft and the center of the Earth and $r_d = r_{Earth} + 400$ [km] is the desired distance from the center of the Earth (a 400 [km] altitude orbit) and $\boldsymbol{v}$ is the spacecraft's post-burn velocity relative to the center of the Earth. Here, $r_{Earth} = 6378$ [km] represents the radius of the Earth. All values of $\boldsymbol{F}$ are geocentric and expressed in [km] and [km/s]. When $\|r - r_d\| = 0$, the spacecraft is located at the desired altitude corresponding to the target LEO. When $\mathbf{r} \cdot \boldsymbol{v} = 0$, the spacecraft's position and velocity vectors are perpendicular to each other. This is the case during apogee, perigee, or a circular orbit, if only considering the two-body dynamics that dominate LEO. Since a circular parking LEO is desired, it is easy to calculate the necessary Δv required to match $\boldsymbol{v}$. The two-body expression for the speed of a circular orbit is given as $v_{LEO} = \sqrt{\frac{GM}{r}}$, where GM is the gravitational parameter of the Earth, and r is the known distance from the center of the Earth during the first burn. Since the spacecraft's orbital energy is maximized when the burn is performed perpendicular to the velocity vector, a simple subtraction of $\|\boldsymbol{v}\| - v_{LEO} = \Delta V_{LEO}$ gives the Δv necessary to exit the LEO and head towards the manifold insertion point. Other characteristics of the LEO can be verified from the state $\boldsymbol{X}_{LEO} = [\boldsymbol{r},\, \boldsymbol{v}_{LEO}]^T$, such as the Keplerian orbital elements; especially eccentricity and inclination.

Given a point on the manifold and the value of $\boldsymbol{Y}$, one can integrate the spacecraft's path backward in time, using Equation 2.2.29. This integration

will terminate at the perigee of the trajectory and yield the values that define $\boldsymbol{F}$ at the end of the trajectory. Of course the value of $\boldsymbol{F}$ is unlikely to be equal to the null vector, even if a reasonable guess of $\boldsymbol{Y}$ is applied. An iterative process, known as Newton's method, is applied here to guide the single shooting algorithm to the correct value of $\boldsymbol{Y}$ that result in $\boldsymbol{F} = \boldsymbol{0}$. To use this method the derivative matrix must be defined

$$DF = \begin{bmatrix} \frac{\partial}{\partial v_x} \|r - r_d\|, & \frac{\partial}{\partial v_y} \|r - r_d\|, & \frac{\partial}{\partial v_z} \|r - r_d\| \\ \frac{\partial}{\partial v_x}\mathbf{r} \cdot \boldsymbol{v}, & \frac{\partial}{\partial v_y}\mathbf{r} \cdot \boldsymbol{v}, & \frac{\partial}{\partial v_z}\mathbf{r} \cdot \boldsymbol{v} \end{bmatrix}. \tag{9.1.3}$$

This matrix can then be used with Newton's method to iteratively solve

$$\boldsymbol{Y}_{new} = \boldsymbol{Y} - DF^T \left(DF\, DF^T\right)^{-1} \boldsymbol{F} \tag{9.1.4}$$

and will terminate when $\|\boldsymbol{F}\| \leq \epsilon$ where $\epsilon = 10^{-10}$ in this study (or any sufficiently small number). Typically, ten iterations or less are needed to converge on the appropriate solution; however, a maximum of 50 iterations are attempted before the algorithm gives up and is unable to converge during single shooting.

In order for this shooting algorithm to converge, an appropriate initial guess solution must be provided. A two phase approach to providing initial guess solutions was used. The first guess is simply a velocity that is identical to the velocity found on the manifold/LPO insertion point itself. This represents no Δv needed for the LPO insertion and is the best case scenario for any insertion burn. From this guess, the shooting algorithm generally increases the necessary Δv and converges on a feasible solution that joins the LEO to the LPO via a transfer arc. Unfortunately, however, an optimal transfer solution is not guaranteed (where "optimal" is defined in terms of Δv). Occasionally,

for a given manifold state, the single shooting algorithm will converge to a feasible solution with a slightly higher Δv than necessary. To compensate for this shortcoming, a recursive algorithm was written for improving the guess solution. The algorithm begins with the output $\left(\boldsymbol{Y}^{(1)}\right)$ of the first run of the single shooting algorithm. This output,

$$\boldsymbol{Y}^{(k+1)} = \left(1 - \frac{1}{4k}\right)\left(\boldsymbol{Y}^{(k)} - \boldsymbol{v}_{LPO}\right) + \boldsymbol{v}_{LPO} \tag{9.1.5}$$

is used to compute the initial guess $\left(\boldsymbol{Y}^{(k+1)}\right)$ that is used in the next iteration of the same shooting algorithm defined above. Equation 9.1.5 serves to decrease the initial guess of the magnitude of the LPO insertion Δv $\left(\Delta v_{LPO}^{(k)} = \left\|\boldsymbol{Y}^{(k)} - \boldsymbol{v}_{LPO}\right\|\right)$. Each successive iteration (with $k \in \left[\begin{array}{cc} 1, & 10 \end{array}\right]$) will decrease the scaling factor of the guess Δv. Initially, when $k = 1$, the scaling factor is 0.75 but that quickly increases to a factor of 0.975 when $k = 10$.

The shooting method and recursion method discussed above may not guarantee that the minimum Δv transfer is found between a LEO and the manifold/LPO insertion point, but it performs very well the majority of the time. Indeed this single recursive single shooting method is easy to program and executes quickly to optimize the transfer to a single manifold/LPO insertion point. This technique, combined with PSO, proves to be a low-cost, reliable, and quick method of optimizing impulsive transfers to LPOs.

9.2 PSO Algorithm

9.2.1 Fitness Function

Once an insertion point has been evaluated using single shooting, the results of that algorithm can be used to assign a "cost" to that point via an objective function known as the fitness function. The fitness function used in this study can be expressed as:

$$J\left(\mathbf{X}_{\mathbf{s.m.}}\right) = c_1 \Delta V\left(\mathbf{X}_{\mathbf{s.m.}}\right) + c_2 \left\| i\left(\mathbf{X}_{\mathbf{s.m.}}\right) - i_{desired} \right\| \tag{9.2.1}$$

where i is the two-body orbital inclination of the LEO, $\Delta V = \Delta V_{LEO} + \Delta V_{LPO}$ is the total Δv required to transfer from LEO to the manifold/LPO insertion point, and c_1 and c_2 are weighting constants chosen by the researcher. Note that the eccentricity of the LEO was not included in the fitness function, as was done in Abraham et al.[75, 58], because a zero eccentricity orbit was guaranteed by default if the single shooting algorithm converged. In this study, typical values of the weighting constants are $c_1 = 1$ and $c_2 = 1$ or $c_2 = 0$ depending on the importance of inclination to the orbit design. When inclination was used, the value of $i_{desired}$ is 28^o with respect to the Moon's orbital plane. This roughly represents the inclination of the International Space Station (ISS) with respect to the Moon. Note that $r_{desired} = r_{Earth} + 400$ [km] also mirrors the typical value of the ISS. In this way, the orbit of the ISS can be used as a baseline LEO orbit with only the longitude of the ascending node and the true anomaly unspecified. If deemed important, these two orbital elements could also be added to Equation 9.2.1.

9.2.1.1 Parametrization of the Search Space

The Particle Swarm Optimization (PSO) algorithm requires the *a priori* definition of a "search space" where it is permitted to search for an optimal solution. In this study, the search space is defined as all states within the invariant stable manifold, $\mathbf{X_{s.m.}}(\tau_{01},\, k) \in W^s$ and within certain bounds. Each state is uniquely defined by exactly two parameters: k and τ. The parameter k represents an individual trajectory member of the stable manifold ($k \in W^s$) that is generated via the method outlined in Chapter 6. The parameter τ_{01} is the second parameter that defines $\mathbf{X_{s.m.}}$ and is defined relative to τ via a simple mapping function. The time of flight, τ, represents the amount of time required to get from an initial state to a state on the nominal Lagrange point orbit, as defined in Chapter 6. Unfortunately, it is impossible to define the entire stable manifold as a search space because it is infinite in nature and a search space (by definition) must be finite. The bounds of τ, therefore, are carefully chosen such that a wide swath of relevant manifold states are captured within the search space, and irrelevant manifold states are excluded.

In this study, the bounds of τ are $\tau_{L.B.} \leq \tau_{s.m.} \leq \tau_{U.B.}$ with each bound being defined in one of two ways: the "Fast Transfer" and "Slow Transfer."

9.2.2 Fast Transfer

In the Fast Transfer the search space is bounded by,

- $\tau_{L.B.}$ being the time that trajectory k crosses the yz-plane located at $x = L_1$ and
- $\tau_{U.B.}$ being the time that trajectory k crosses first leaves the LPO (i.e. $\tau = 0$).

9.2.3 Slow Transfer

In the Slow Transfer the search space is bounded by,

- $\tau_{L.B.}$ being the time that trajectory k crosses the yz-plane from the positive x direction and
- $\tau_{U.B.}$ being the time that trajectory k crosses the yz-plane located at $x = L_1$.

The first search space allows the PSO algorithm to focus on the highly localized manifold that exists very near the LPO. While it is true that a spacecraft inserted into this portion of the manifold may spend a considerable amount of time in the manifold before reaching the destination LPO, this is not a problem. The majority of mission goals (aside from rendezvous) can be achieved in an LPO that is very near the target LPO but not necessarily on it. Since the majority of this search space circles around the target LPO, a spacecraft placed into this portion of the manifold can be considered to be in a Lissajous orbit (of approximate size and shape as the target LPO) that flows into the target LPO as time progresses forward. The exploration of this search space is particularly attractive missions that require short time of flight transfers such as manned missions to LPO's.

The other search space allows the PSO algorithm to focus on the remainder of the stable manifold. Note that any spacecraft inserted into this portion of the manifold will have to spend an appreciable amount of time coasting to the vicinity of the target LPO. The bound, $\tau_{L.B.}$ was chosen as a matter of convenience and practicality. While it is true that the trajectories of W^s continue to flow for an infinite amount of time, one needs to cut off this flow

after a finite amount of time due to the limitations of computing power [58, 75]. Since the run times of the PSO method can become quite large, a smaller search space is needed to adequately converge on an optimal solution.

The values of τ are mapped to τ_{01} using the simple relationship that $\tau_{L.B.} = 1$ and $\tau_{U.B.} = 0$. Therefore, the values of τ for a given value of k are mapped to a normalized range of $0 \leq \tau_{01} \leq 1$. This mapping ensures that values of τ between $\tau_{L.B.}$ and $\tau_{U.B.}$ are treated equally, regardless of the value of k and the time of flight between the nominal Lagrange point orbit and the yz-plane located at $x = L_1$. In a similar fashion, the values of k are mapped between $1 \leq k \leq N$ via a modulus function. In this study, $k = k_{desired}$ **mod** N. This means, for example, that if $k_{desired} = N + x$ then $k = x$ assuming $0 \leq x \leq N$. Using this technique, no value of $k_{desired}$ is ever excluded from the search space but is instead looped back onto itself in k-space. In summary, any value of $\mathbf{X}_{\mathbf{s.m.}}(\tau_{01},\, k)$ can be uniquely parametrized, in $k\tau$-space, in terms of τ_{01} and k with the boundaries of these parameters being real numbers, $0 \leq \tau_{01} \leq 1$ and positive integers, $1 \leq k \leq N$.

9.3 Application of the Fast Transfer Case

The PSO method was used to optimize the "Fast Transfer" search space described above. This was accomplished by setting $c_1 = 1$ and $c_2 = 0$ (in the fitness function, Equation 6.4.1) for the case of optimization with respect to any inclination and $c_1 = 1$ and $c_2 = 1$ for the case of a 28^o desired LEO inclination. In both cases, the optimal trajectories are shown in Figure 9.1. Notice how the spacecraft immediately enters a Lissajous orbit that would be useful for the majority of Earth-Moon L_1 applications such as communications relay,

Earth/Moon observation, and mission staging areas. As noted in Table 9.1, the Lissajous orbit will deliver the spacecraft into the target LPO over a period of $60-90$ days where it can rendezvous with another spacecraft that is on the exact same orbit. Alternatively, to speed up a desired rendezvous, the spacecraft could execute a small maneuver to decrease a rendezvous time using a modest amount of ΔV.

Table 9.1: Summary of fast transfer to (a) any inclination and (b) a 28^o inclination.

	J	ΔV_{LEO} [km/s]	ΔV_{halo} [km/s]	Total ΔV [km/s]	TOF to Insertion [days]	TOF in Manifold [days]	TOF Total [days]
Fast Transfer	3.44	3.07	0.37	3.44	4.98	90.00	94.98
Fast Transfer 28^o	3.58	3.07	0.44	3.51	4.89	60.94	65.83

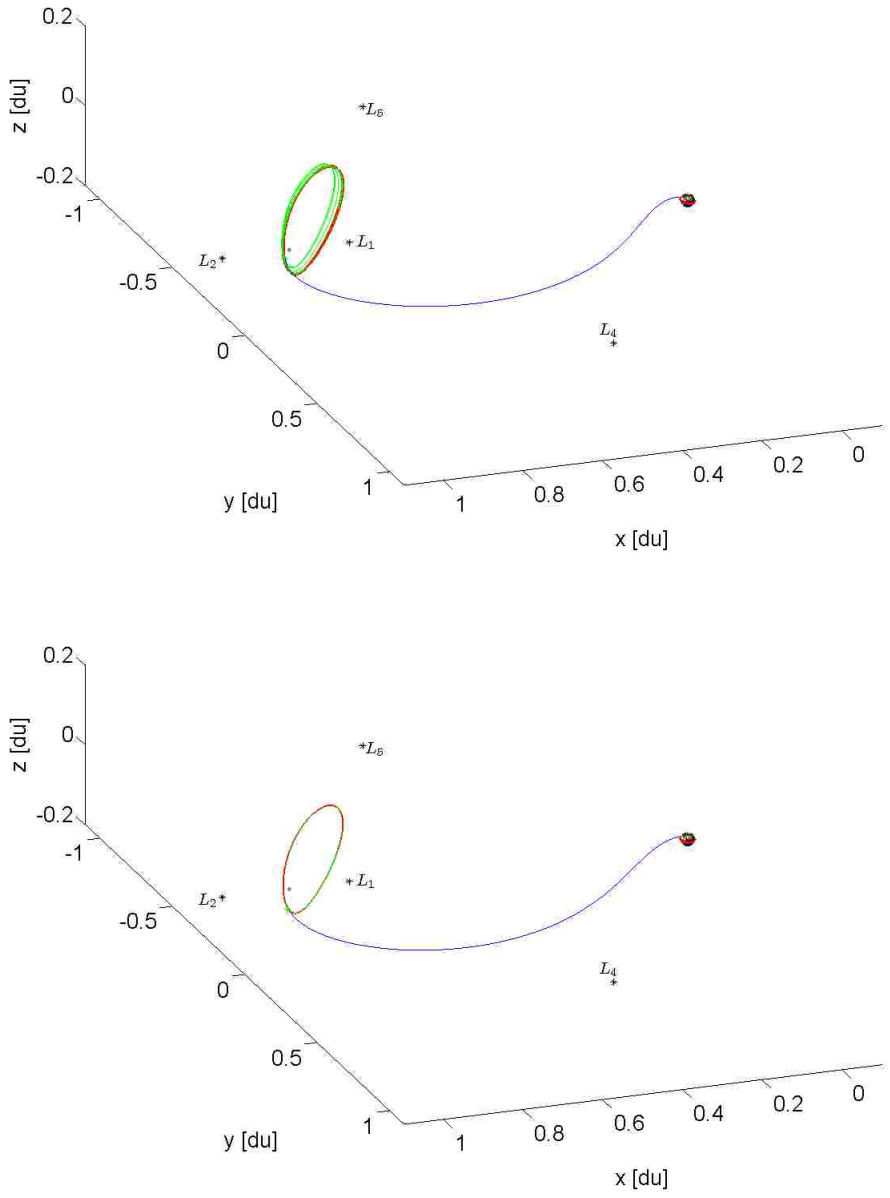


Figure 9.1: Fast transfer from a 400 km altitude LEO (red) to the target LPO (red) via a cislunar coast (blue) following a cislunar injection burn and a stable manifold coast (green) following a manifold injection burn. *Top*: Optimized for any inclination. *Bottom*: Optimized for 28^o inclination.

9.4 Application of the Slow Transfer Case

The PSO method was also used to optimize the "Slow Transfer" search space described above. As before, this was accomplished by setting $c_1 = 1$ and $c_2 = 0$ for the case of optimization with respect to any inclination and $c_1 = 1$ and $c_2 = 1$ for the case of a 28^o desired LEO inclination. In both cases, the optimal trajectories are shown in Figure 9.2. Notice how much longer it takes for the spacecraft to reach the vicinity of L_1. Adding the values of the "TOF to Insertion" and "TOF in pre-L_1 Manifold" columns of Table 9.2 together it is possible to calculate the time of flight required to reach the vicinity of L_1 and directly compare it to that of a fast transfer. Disregarding LEO inclination, an optimized slow transfer requires a 22.93 day flight compared with a 4.98 day flight as seen in Table 9.1. This means that a fast transfer could move a spacecraft to the vicinity of L_1 with approximately 100 [m/s] less Δv and approximately 4 to 5 times quicker. Similar results exist for the 28^o inclination (ISS mission) example. In this case, the fast transfer requires 160 [m/s] less Δv and approximately 3 to 4 times quicker than it's slow counterpart.

It is interesting to note that Alessi et al. [77] found that the optimal manifold insertion point was manifold apogee; that is the point on the manifold that is the maximum distance away from the Earth. In this study, all optimal manifold insertion points discovered using the slow transfer search space were very near apogee. For example, the optimal insertion point found by PSO for the "Slow Transfer, Any Inclination" case was only $30,000$ km away from the manifold apogee. This is only five Earth radii in size away from the apogee condition (which is relatively small when the search space is nearly ± 60 Earth radii in diameter). Even better, was the optimal insertion point for the "Slow Transfer,

28^o Inclination" case at $3,000$ km from manifold apogee. This is less than half an Earth radius and is about as close as one could hope for when matching the results from Alessi et al. Strong agreement with previously published work [77, 78] corroborates the efficacy of the PSO method as applied here.

Table 9.2: Summary of slow transfer.

	J	ΔV_{LEO} [km/s]	ΔV_{halo} [km/s]	Total ΔV [km/s]	TOF to Insertion [days]	TOF in pre-L_1 Manifold [days]	TOF in post-L_1 Manifold [days]	TOF Total [days]
Slow Transfer	3.55	3.04	0.51	3.55	3.89	19.04	131.80	154.73
Slow Transfer 28^o	3.69	3.05	0.63	3.68	3.76	11.51	121.87	137.14

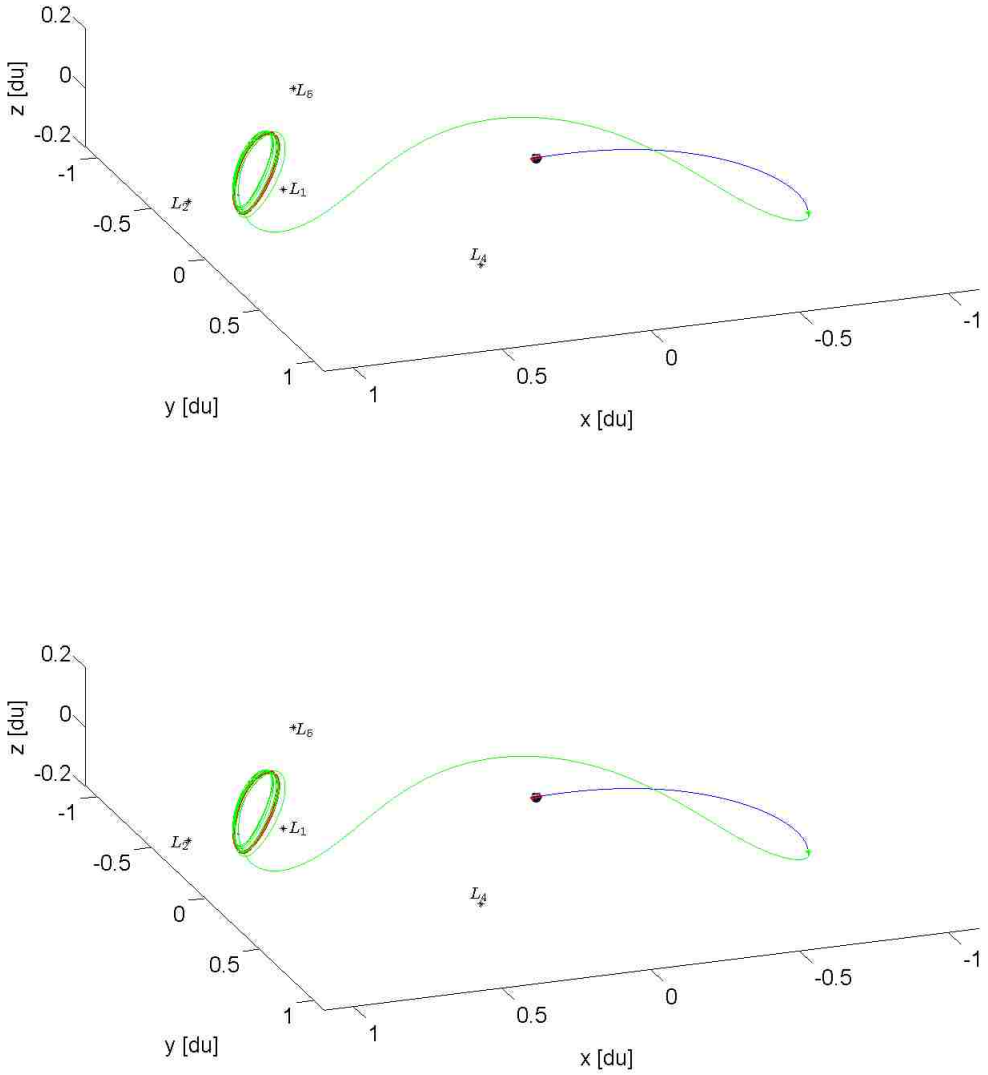


Figure 9.2: Slow transfer from a 400 km altitude LEO (red) to the target LPO (red) via a cislunar coast (blue) following a cislunar injection burn and a stable manifold coast (green) following a manifold injection burn. *Top*: Optimized for any inclination. *Bottom*: Optimized for 28^o inclination.

This concludes the study of the optimization of two-burn transfers from LEO to an LPO using PSO. Conclusions are found in Section 10.3.

10 Conclusions & Future Work

In this dissertation, a successful demonstration of a method to optimize low-thrust (and impulsive) transfer trajectories from a geocentric orbit to an Earth-Moon Lagrange point orbit was presented. This method involves the utilization of the evolutionary algorithm known as Particle Swarm Optimization (PSO) and can be used independently or in conjunction with gradient-based optimization methods. The key principle when attempting to utilize a PSO algorithm for spacecraft trajectory optimization is the ability to translate the problem into one with a clearly defined search space that can be discretely or continuously characterized by an objective function. This was accomplished by utilizing the invariant stable manifold of the target LPO as a suitable search-space. Any spacecraft inserted into this manifold will ballistically coast and insert into the target LPO without additional propulsion. Each point on the manifold was translated into a discrete "cost" value via the clever use of a tangential-thrust control law and a simple "fitness" function. Once this search-space is defined, the Particle Swarm Optimization algorithm is employed to identify the globally best value (minimum) of the fitness function. Although this value is not guaranteed to be the globally optimal solution, the PSO method generally gets very, very close to this solution and may be used as a pruning algorithm or a preliminary search algorithm prior to the application of a different, gradient-based

optimization routine.

10.1 Conclusions for One-Dimensional PSO Applications

This study shows that preliminary optimization of a low-thrust transfer trajectory from a GEO-energy orbit to a nominal Earth-Moon Lagrange point orbit is indeed possible using Particle Swarm Optimization. Since the search space is often non-convex, traditional convex optimization methods do not apply. Additionally, gradient based optimization algorithms would likely have a great deal of difficulty avoiding a local minima. Instead, an evolutionary PSO algorithm has proven to be a highly useful optimization technique that significantly outperforms random chance algorithms, such as the Monte Carlo technique, as evidenced by Figure 7.1. While a large number of function evaluations is important, the percentage of candidate trajectories identified by the PSO method increases only logarithmically with increasing function evaluations. Since run time increases proportionately with function evaluations it seems reasonable to identify 160 function evaluations as a compromise between robust candidate identification and CPU run time. It is also interesting to discover that the positive perturbation of the stable manifold, W^{s+} normally contains more candidate patch points than its counterpart, W^{s-}. It is unknown if this is the result of generalized system dynamics or simply a phenomena unique to the given nominal orbit.

The optimization of the low-thrust trajectory with respect to propellant consumed and TOF was successfully demonstrated in this study. The PSO algo-

rithm is, in general, much more successful in optimizing the trajectory than random chance would allow. Convergence on the discrete, global minimum, as observed in Figure 7.6, demonstrate the robustness of the PSO method; especially compared to gradient based algorithms used by Mingotti et al. [30] which can become trapped in a local minima. Although the PSO method can not guarantee convergence upon the globally optimal solution it does offer the benefits of a wide coverage of the search-space and would be an ideal choice for preliminary trajectory optimization.

10.2 Conclusions for Two-Dimensional PSO Applications

The preliminary optimization of a low-thrust transfer from geocentric orbit to an Earth-Moon, L_1 halo orbit has been demonstrated in this study. Particle Swarm Optimization was used to prune a highly non-convex search space that tends to trap traditional, gradient-based optimization algorithms in non-optimal, local minima. Indeed, since the topology of the fitness function is very rough, being filled with multiple extrema and discontinuities, a pruning method that is capable of obtaining a guess solution very near the optimal solution is warranted. Once the results of this PSO algorithm are obtained, they can be refined using a gradient based approach as demonstrated by Mingotti et al. [30, 31, 32]

It has been demonstrated, in this study, that utilizing a "local" version of the PSO algorithm can greatly improve its ability to simultaneously interrogate multiple minima without becoming trapped in a non-optimal, local minima.

This result is in agreement with past research [36]. Based on the data gathered in this study, an appropriate choice of N_p is in the range of 200 to 400 particles and a suitable value of the convergence metric is $\gamma_{stop} \geq 0.75$. These values offer the best compromise between a thorough interrogation of the search space and prohibitive CPU run-time. This study also demonstrated a vast improvement in CPU run-time when utilizing the 2-D version of the PSO algorithm as opposed to the 1-D version [58] utilized elsewhere. Indeed, an order of magnitude improvement in the CPU run-time has been demonstrated between the 2-D and 1-D versions of the PSO algorithm.

The optimal trajectory found using the 2-D, local PSO algorithm was highly three dimensional in nature – especially when three body effects are at their maximum. This trajectory began on a circular, geocentric orbit with a Jacobi energy equivalent to that of GEO. Three body effects are insignificant below this orbit so lower trajectories were not studied in this research. Instead, a connecting low-thrust spiral or chemical Hohmann transfer was assumed. Overall, this technique is general enough to be applied to many other Lagrange point orbits that utilize a (initially) low-eccentricity, low-thrust transfer.

10.3 Conclusions for Two-Impulsive Burn PSO Applications

In this study, an evolutionary algorithm called Particle Swarm Optimization, was used in conjunction with a traditional, gradient-based optimization method called Single Shooting. The fusion of a gradient-based technique with an evolutionary algorithm attempts to blend the strengths of both methods while

minimizing their weaknesses. While the PSO method may be much slower than gradient-based algorithms, its robust global optimization capability when used to optimize a non-convex objective function far exceeds that of gradient-based methods; especially when the objective function is non-differentiable or nearly so. On the other hand, using a shooting method to optimize a single manifold insertion point is relatively fast (a few seconds) and can determine a very efficient two-burn transfer trajectory. While shooting does not guarantee the optimal transfer solution is found for a given insertion point, the method of iterative shooting does reduce the likelihood of a non-optimal result. The PSO/Shooting method, described above, is relatively simple to program and requires a modest amount of resources to run. Typical run-times for a desktop computer are on the order of $10 - 20$ hours but could be significantly reduced (by one or two orders of magnitude) by utilizing a parallel computing cluster. Due to the ease of programming and reasonable run-time, the PSO/Shooting method is useful in preliminary optimization of space mission design or for trajectory pruning applications.

The optimization of the sample problem given in this study is also noteworthy. In agreement with Alessi et al., the PSO/Shooting method identified the optimal manifold insertion point as apogee when considering the slow transfer search space. This result adds creditability to the PSO/Shooting method since Alessi et al. used an entirely different method to gather their data, yet provide similar results. In light of this published work, however, it is a bit surprising to note that the fast transfer was superior to the slow transfer in terms of both Δv and time of flight. It is currently unknown whether this result is characteristic of all LPOs or is unique to the one chosen for this study. Given enough computational power it may be useful to investigate the influence of the choice in LPO

over the optimization results. For example, LPO's in different systems should be studied (Earth-Moon, Sun-Earth, Sun-Mars, etc.), LPOs about the L_1, L_2, L_3, and $L_{4,5}$ points could be studied, and even various shapes and families of LPO's should be studied to see if there are any commonalities between them. Optimal manifold insertion points at apogee may be a common characteristic of most LPO's but not all. It may be useful to future mission designers to know what characteristics of a LPO lend themselves to optimal fast transfers rather than slow transfers. This is especially true for human spaceflight to LPO's.

10.4 Conclusions From This Work

In this dissertation, three separate studies were conducted to test the ability of the PSO method to optimize geocentric to LPO orbit transfers. The first study, discussed in Chapter 7, optimizes a low-thrust transfer from a circular, geosynchronous-altitude orbit to an LPO using a restricted, 1-D search-space. This study demonstrated that a PSO algorithm could optimize a non-convex fitness function (defined at discrete points) faster and more accurately than a random-guessing, Monte Carlo technique. It also showed that the PSO method was capable of avoiding local minima in a way that gradient-based algorithms cannot. An optimal low-thrust trajectory was generated and tested very quickly using this technique.

The next study, found in Chapter 8, generalized the PSO method into one that used a "local" version of the PSO algorithm and utilized a 2-D search-space. In this study, the fitness function was further characterized as one having numerous minima of similar magnitude and sharp discontinuities. The local version of the PSO algorithm allowed for simultaneous interrogation of multiple

local minima. This is in contrast with the original PSO algorithm which may, prematurely, draw particles away from multiple minima in favor of a single minimum. This has the potential to be detrimental to the goal of optimization, since other minima have not been thoroughly investigated. The choice of the parameters N_p and j_{max} were methodically refined, and a new parameter, γ_{stop}, was introduced to decrease unproductive CPU time. The end result was a 2-D local PSO algorithm that ran an order of magnitude faster than its predecessor, the 1-D version. The end result was an optimal low-thrust trajectory that appeared to be very three-dimensional (which is uncharacteristic of Keplerian orbits) and takes full advantage of the three-body effects of the Earth-Moon system.

The final study, found in Chapter 9, is a bit different from the previous two. This study focuses on impulsive, two-maneuver transfers from Low Earth Orbit (LEO) to the target LPO via a hybrid PSO/shooting technique. This formulation attempts to capture the "best of both worlds" by utilizing the gradient-based shooting method to complement the evolutionary PSO algorithm. The search-space was split into a "fast" and "slow" space where the fast space encompassed the stable manifold near the LPO, and the slow space consisted of the remainder of the manifold. In agreement with previous research, it was found that the optimal manifold insertion point was that of apogee in the slow search-space case. Surprisingly, however, it was discovered that the optimal fast transfer was superior to the optimal slow transfer, both in terms of propellant consumption and time of flight. This is a significant result because propellant consumption and time of flight are typically viewed in terms of a trade-off. Here, however, no such trade is necessary.

10.5 Future Work

Much work could be done to further investigate the techniques, algorithms, and results initially explored in this dissertation. Ideas for future research include:

- Further investigation of tuning parameters for the PSO algorithm. This topic was addressed in Chapter 8 but could be further refined to include an automated method of tuning each parameter. Examples include neural networks, genetic algorithms, or other heuristic-based approaches that allow for the "training" or discovery of optimal tuning parameters.

- This entire study could be repeated using other evolutionary or heuristic approaches. The results could then be compared with that found in this study.

- Other forms of the fitness/cost/objective function could be tested and evaluated. The form of the objective function drives the system to evolve toward a final state that (it is hoped) is optimal. It would be interesting to study what characteristics of a fitness function drive the system to such a state in the shortest number of iterations. It is also important to determine how this performance changes as the target LPO is changed.

- This entire study should be repeated using other LPOs (Lyapunov, halo, or Lissajous) and the performance of the PSO algorithm should be recorded for each type and family. Additionally, orbits about other Lagrange points (L_{2-5}) in the Earth-Moon system should be explored, as well as orbits about Lagrange points of the Sun-Earth, Sun-Mars, Sun-Jupiter, and other three-body systems.

- Additionally, two-maneuver, impulsive transfers should be studied for other LPOs. Fast and slow transfers should be studied and compared to determine if fast transfers are superior to slow transfers for all LPOs or only a subset of them (i.e. a particular family, Lagrange point, three-body system, and so on).

References

[1] W. S. Koon, M. W. Lo, J. E. Marsden, and S. D. Ross, *Dynamical Systems, the Three-Body Problem and Space Mission Design*. Marsden Books, 2008.

[2] V. Szebehely, *Theory of Orbits: The Restricted Problem of Three Bodies*. New York, NY: Academic Press Inc., 1967.

[3] R. W. Farquhar, *The Control and Use of Libration-Point Satellites*. Ph.D. Dissertation, Dept. of Aeronautics and Astronautics, Stanford University, Stanford, CA, 1968.

[4] R. W. Farquhar, "A Halo-Orbit Lunar Station," *Astronautics & Aeronautics*, Vol. 10, 1972, pp. 59–63.

[5] R. W. Farquhar and A. A. Kamel, "Quasi-Periodic Orbit About The Translunar Libration Point," *Celestial Mechanics*, Vol. 7, 1973, pp. 458–473.

[6] E. Davis, "Transfers to the Earth-Moon L3 Halo Orbits," *AAS/AIAA Astrodynamics Specialist Conference*, Minneapolis, MN, August 2012, 10.2514/6.2012-4593.

[7] Lockheed Martin, "Early Human L2-Farside Missions," 2010. Accessed on 3-14-2013,

http://www.lockheedmartin.com/content/dam/lockheed/data/space/documents/orion/LMFarsideWhitepaperFinal.pdf.

[8] T. A. Heppenheimer, "Steps Toward Space Colonization: Colony Location and Transfer Trajectories," *Journal of Spacecraft and Rockets*, Vol. 15, September - October 1978, pp. 305–312, 10.2514/3.28013.

[9] J. A. Goff, B. F. Kutter, F. Zegler, D. Bienhoff, F. Chandler, and J. Marchetta, "Realistic Near-Term Propellant Depots: Implementation of a Critical Spacefaring Capability," Pasadena, CA, AIAA SPACE 2009 Conference & Exposition, September 2009, pp. 13–31, 10.2514/6.2009-6756.

[10] NASA, "Solar System Exploration: ISSEE-3/ICE," January 2011. Accessed on 2-14-2013, http://solarsystem.nasa.gov/missions/profile.cfm?MCode =ISEE-ICE&Display=ReadMore.

[11] W. S. Koon, M. W. Lo, J. Marsden, and S. Ross, "The Genesis Trajectory and Heteroclinic Cycles," *Advances in Astronautical Sciences*, No. 103, 1999, pp. 2327–2343.

[12] ESA, "JWST Fact Sheet," September 2013. Accessed on 4-15-2013, http://www.esa.int/Our-Activities/Space-Science/JWST-factsheet.

[13] V. Angelopoulos, "The ARTEMIS Mission," *Space Science Review*, 2010, pp. 1–23, 10.1007/s11214-010-9687-2.

[14] T. H. Sweetser, S. B. Broschart, V. Angelopoulos, G. J. Whiffen, D. C. Folta, M.-K. Chung, S. J. Hatch, and M. A.Woodard, "ARTEMIS Mis-

sion Design," *Space Science Review*, Vol. 165, March 2012, pp. 27–57, 10.1007/s11214-012-9869-1.

[15] J. S. Sovey, V. K. Rawlin, and M. J. Patterson, "Ion Propulsion Development Projects in U.S.: Space Electric Rocket Test I to Deep Space 1," *Journal of Propulsion and Power*, Vol. 17, May-June 2001, pp. 517–526.

[16] M. Rayman, P. Varghese, D. Lehman, and L. Livesay, "Design of the First Interplanetary Solar Electric Propulsion Mission," *Journal of Spacecraft and Rockets*, Vol. 39, August 2002, pp. 589–595.

[17] J. Polk, R. Kakuda, J. Anderson, J. Brophy, V. Rawlin, M. Patterson, J. Sovey, and J. Hamley, "Displaced Periodic Orbits With Low-Thrust Propulsion in the Earth-Moon System," AIAA-99-2274, Los Angeles, CA, 35th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit, June 1999.

[18] M. Rayman, T. Fraschetti, C. Raymond, and C. Russell, "Dawn: A Mission in Development for Exploration of Main Belt Asteroids Vesta and Ceres," *Acta Astronautica*, Vol. 58, 2006, pp. 605–616.

[19] E. Choueiri, "A Critical History of Electric Propulsion: The First 50 Years (1906-1956)," *Journal of Propulsion and Power*, Vol. 20, March-April 2004, pp. 193–203.

[20] R. Goddard, "Report Concerning Further Developments," 1920. Accessed on 5-1-2013, http://siarchives.si.edu/history/exhibits/stories/march-1920-report-concerning-further-developments-space-travel.

[21] H. Oberth, "Ways to Spaceflight," 1972. Translated from original German in 1929, NASA TT F-622, Accessed on 5-3-2013, https://archive.org/details/nasa-techdoc-19720008133.

[22] T. Masek and E. Pawlik, "Thrust System Technology for Solar Electric Propulsion," AIAA-1968-541, Cleveland, OH, AIAA 4th Propulsion Joint Specialist Conference, June 1968.

[23] D. F. Lawden, *Optimal Trajectories for Space Navigation.* Butterworths, 1963.

[24] W. Melbourn and C. Sauer, "Optimum Interplanetary Rendezvous with Power-Limited Vehicles," *AIAA Journal*, Vol. 1, January 1963, pp. 54–60.

[25] T. Edelbaum, "Optimum Power-Limited Orbit Transfer in Strong Gravity Fields," *AIAA Journal*, Vol. 3, May 1965, pp. 921–925.

[26] W. A. Scheel and B. A. Conway, "Optimization of Very-Low-Thrust, Many Revolution Spacecraft Trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 17, November-December 1994, pp. 1185–1192, 10.2514/3.21331.

[27] A. L. Herman and D. B. Spencer, "Optimal, Low-Thrust Earth-Orbit Transfers Using Higher-Order Collocation Methods," *Journal of Guidance, Control, and Dynamics*, Vol. 25, January-February 2002, pp. 40–47.

[28] D. C. Redding and J. V. Breakwell, "Optimal Low-Thrust Transfers to Synchronous Orbit," *Journal of Guidance, Control, and Dynamics*, Vol. 7, No. 2, 1984, pp. 148–155, 10.2514/3.8560.

[29] D. C. Redding, "Highly Efficent, Very Low-Thrust Transfer to Geosynchronous Orbit: Exact and Approximate Solutions," *Journal of Guidance, Control, and Dynamics*, Vol. 7, 1984, pp. 141–147, 10.2514/3.8559.

[30] G. Mingotti, F. Topputo, and F. Bernelli-Zazzera, "Combined Optimal Low-Thrust and Stable-Manifold Trajectories to the Earth-Moon Halo Orbits," *AIP Conference Proceedings* (E. Belbruno, ed.), American Institute of Physics, 2007, pp. 100–110, 10.1063/1.2710047.

[31] G. Mingotti, F. Topputo, and F. Bernelli-Zazzera, "Numerical Methods to Design Low-Energy, Low-Thrust Sun-Perturbed Transfers to the Moon," *Proceedings of 49th Israel Annual Conference on Aerospace Sciences*, Tel Aviv-Haifa, Israel, 2009, pp. 1–14.

[32] G. Mingotti, F. Topputo, and F. Bernelli-Zazzera, "Transfers to distant periodic orbits around the Moon via their invariant manifolds," *Acta Astronautica*, Vol. 79, October-November 2012, pp. 20–32, 10.1016/j.actaastro.2012.04.022.

[33] J. Senet and C. Ocampo, "Low-Thrust Variable-Specific-Impulse Transfers and Guidance to Unstable Periodic Orbits," *Journal of Guidance, Control, and Dynamics*, Vol. 28, March-April 2005, pp. 280–290, 10.2514/1.6398.

[34] M. Pontani and B. A. Conway, "Particle Swarm Optimization Applied to Space Trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 33, September-October 2010, pp. 1429–1441, 10.2514/1.48475.

[35] M. Pontani, P. Gosh, and B. Conway, "Particle Swarm Optimization of Multiple-Burn Rendezvous Trajectories," *Journal of Guidance, Control, and Dynamics*, Vol. 35, No. 4, 2012, pp. 1192–1207, 10.2514/1.55592.

[36] B. A. Conway, ed., *Spacecraft Trajectory Optimization*. Cambridge, 2010.

[37] C. R. Bessette and D. B. Spencer, "Identifying Optimal Interplanetary Trajectories Through a Genetic Approach," *AIAA/AAS Astrodynamics Conference*, No. AIAA 2006-6306, Keystone, CO, AIAA/AAS Astrodynamics Conference, 2006.

[38] C. R. Bessette and D. B. Spencer, "Optimal Space Trajectory Design: A Heuristic-Based Approach," *Advances in the Astronautical Sciences*, Vol. 124, 2006, pp. 1611–1628.

[39] D. A. Vallado, *Fundamentals of Astrodynamics and Applications*. Space Technology Series, McGraw-Hill, 1997. ISBN 0-07-066834-5.

[40] G. Gomez, J. Masdemont, and C. Simo, "Quasi-Halo Orbits Associated with Libration Points," *Journal of the Astronautical Sciences*, Vol. 46, No. 2, 1998, pp. 135–176.

[41] T. A. Pavlak, "Mission Design Applications In The Earth-Moon System: Transfer Trajectories and Stationkeeping," Master's thesis, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, May 2010.

[42] Navigation and Ancillary Infornmation Facility, Jet Propulsion Laboratory, "The SPICE Concept," August 2012. Accessed on 10-22-2012, http://naif.jpl.nasa.gov/naif/index.html.

[43] Analytical Graphics Inc., "Database for Interplanetary Spacecraft," 2012. Accessed on 9-13-2012, http://www.agi.com/resources/faq-system.

[44] T. A. Pavlak and K. C. Howell, "Strategy for Optimal, Long-Term Libration Point Orbit Stationkeeping in the Earth-Moon System," AIAA 2012-4665, AAS/AIAA Astrodynamics Specialist Conference, 2011, 10.2514/6.2012-4665.

[45] M. Ozimek, "A Low-Thrust Transfer Strategy to Earth-Moon Collinear Libration Point Orbits," Master's thesis, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, December 2006.

[46] M. T. Ozimek, *Low-Thrust Trajectory Design and Optimization of Lunar South Pole Coverage Missions*. PhD thesis, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, May 2010.

[47] J. Breakwell and J. Brown, "The "Halo" Family of 3-Dimensional Periodic Orbits in the Earth-Moon Restricted 3-Body Problem," *Celestial Mechanics*, Vol. 20, November 1979, pp. 389–404, 10.1007/BF01230405.

[48] U.C. Berkeley, "ARTEMIS Misson," 2011. Accessed on 9-20-2012, http://cse.ssl.berkeley.edu/artemis/mission-artemis.html.

[49] U.C. Berkeley, "Index of Data: Themis," 2011. Accesed on 9-20-2012, http://themis.ssl.berkeley.edu/data/themis/.

[50] L. Perko, *Differential Equations and Dynamical Systems*. 175 Fifth Avenue, New York, NY, 10010, USA: Springer, 3rd ed., 2001.

[51] H. Poincare, *Les Methodes Nouvelles de la Mecanique Celeste*. Gauthier-Villars, 1893.

[52] T. Parker and L. Chua, *Practical Numerical Algorithms for Chaotic Systems*. New York: Springer-Verlag Publishing, 1989.

[53] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations*. John Wiley and Sons, 1991.

[54] B. Marchand, "Temporary Satellite Capture of Short-Period Jupiter Family Comets From the Perspective of Dynamical Systems," Master's thesis, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, December 2000.

[55] V. Yakubovich and V. Starzhinski, *Linear Differential Equations with Peridoic Coefficents*. John Wiley and Sons, 1975.

[56] K. C. Howell, "Three-Dimensional, Periodic, 'Halo' Orbits," *Celestial Mechanics*, Vol. 32, No. 1, 1984, pp. 53 – 71.

[57] D. Grebow, "Generating Periodic Orbits in the Circular Restricted Three-Body Problem with Applications to Lunar South Pole Coverage," Master's thesis, School of Aeronautics and Astronautics, Purdue University, West Lafayette, IN, May 2006.

[58] A. J. Abraham, D. B. Spencer, and T. J. Hart, "Optimization of Preliminary Low-Thrust Trajectories From GEO-Energy Orbits to Earth-Moon, L1, Lagrange Point Orbits Using Particle Swarm Optimization," AAS 13-925, Hilton Head, SC, AAS Astrodynamics Specialist Conference, August 2013.

[59] T. Back, U. Hammel, and H. Schwefel, "Evolutionary Computation: Comments on the History and Current State," *IEEE Transactions on Evolutionary Computation*, Vol. 1, April 1997, pp. 3–17.

[60] H. Bremermann, *Self-Organizing Systems*, ch. 7, Optimization Through Evolution and Recombination, pp. 93–106. Washington, D.C.: Spartan Books, 1962.

[61] R. Friedberg, "A Learning Machine: Part I," *IBM Journal*, Vol. 3, July 1959, pp. 282–287.

[62] G. Box, "Evolutionary Operation: A Method for Increasing Industrial Productivity," *Applied Statistics*, Vol. 6, No. 2, 1957, pp. 81–101.

[63] G. Moore, "Cramming More Components Onto Integrated Circuits," *Electronics*, Vol. 38, April 1965, pp. 114–117.

[64] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *Proceedings of IEEE International Conference on Neural Networks*, Vol. 4, Western Australia, Institute of Electrical and Electronics Engeineers, November 1995, pp. 1942–1948, 10.1109/ICNN.1995.488968.

[65] F. Heppner and U. Grenader, *A Stochastic Nonlinear Model for Coordinated Bird Flocks*. AAAS Publications, December 1990. ISBN: 978-0871683502.

[66] C. Reynolds, "Flocks, Herds and Schools: A Distributed Behavioral Model," *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, Vol. 21, New York, NY, ACM SIGGRAPH Computer Graphics, ACM SIGGRAPH Computer Graphics, July 1987, pp. 25–34, 10.1145/37402.37406.

[67] E. Wilson, *Sociobilology: The New Synthesis*. Cambridge, MA: Belknap Press, 1975.

[68] R. Poli, "Analysis of the Publications on the Applications of Particle Swarm Optimization," *Journal of Artifical Evolution and Applications*, Vol. 2008, Article ID 685175, 10.1155/2008/685175.

[69] Y. Shi and R. Eberhart, "A Modified Particle Swarm Optimizer," *Proceedings of the IEEE Congress on Evolutionary Computation*, Piscataway, NY, Institute of Electrical and Electronics Engineers, 1999, pp. 69–73.

[70] C. Spaans and E. Mooij, "Performance Evaluation of Global Trajectory Optimization Methods for a Solar Polar Sail Mission," *AIAA Guidance, Navigation, and Control Conference*, AIAA 2009-5666, Chicago, IL, AIAA Guidance, Navigation, and Control Conference, AIAA Guidance, Navigation, and Control Conference, 2009.

[71] K. Zhu, F. Jiang, J. Li, and H. Baoyin, "Trajectory Optimization of Multi-Asteroids Exploration with Low-Thrust," *Transactions of the Japan Society for Aeronautical and Space Sciences*, Vol. 52, No. 175, 2009, pp. 47–54.

[72] K. Zhu, J. Li, and H. Baoyin, "Satellite Scheduling Considering Maximum Observation Coverage Time and Minimum Orbital Transfer Fuel Cost," *Acta Astronautica*, Vol. 66, 2010, pp. 220–229.

[73] M. Vasile, E. Minisci, and M. Locatelli, "On Testing Global Optimization Algorithms for Space Trajectory Design," *AIAA/AAS Astrodynamics Specialist Conference*, AIAA 2008-6277, Honolulu, HI, AIAA/AAS Astrodynamics Specialist Conference, AIAA/AAS Astrodynamics Specialist Conference, 2008.

[74] R. Sentinella and L. Casalino, "Cooperative Evolutionary Algorithm for

Space Trajectory Optimization," *Celestial Mechanics and Dynamical Astronomy*, Vol. 105, No. 1-3, 2009, pp. 211–227.

[75] A. J. Abraham, D. B. Spencer, and T. J. Hart, "Preliminary Optimization 2-D Optimization of Low-Thrust, Geocentric-to-Halo-Orbit Transfers, via Particle Swarm Optimization," AAS 14-315, Santa Fe, NM, AIAA Space Flight Mechanics Meeting, January 2014.

[76] A. J. Abraham, D. B. Spencer, and T. J. Hart, "Particle Swarm Optimization of 2-Maneuver, Impulsive Transfers From LEO to Lagrange Point Orbits via Shooting," Laurel, MD, Proceedings of the 24th International Symposium on Space Flight Dynamics, May 2014.

[77] E. Alessi, G. Gomez, and J. Masdemont, "Two-Manoeuvres Transfers Between LEOs and Lissajous Orbits in the Earth-Moon System," *Advances in Space Research*, Vol. 45, 2010, pp. 1276–1291, 10.1016/j.asr.2009.12.010.

[78] F. Renk, M. Hechler, and E. Messerschmid, "Exploration Missions in the Sun-Earth-Moon System: A Detailed View on Selected Transfer Problems," *Acta Astronautica*, Vol. 67, July-August 2010, pp. 82–96, http://dx.doi.org/10.1016/j.actaastro.2009.10.023.

[79] N. J. Cornish, "Professor of Physics," Montana State University, 1999. Accessed on 4-18-2012, http://www.physics.montana.edu/faculty/cornish/lagrange.pdf.

Appendix

Additional information pertaining to a more advanced stability analysis of Lagrange points is presented here. The information in this appendix is complementary to the body of this dissertation but is not critical to its understanding.

L_3 Stability Approximation with Low Values of μ

Before the advent of digital computers, it was very difficult to determine the stability of a Lagrange point. In the 19^{th} century it was believed that a hidden planet, known as Planet X, may be hiding behind the Sun at the Sun-Earth L_3 point [79]. The fact that L_3 is continuously hidden from Earth's view made testing this hypothesis nearly impossible. It is, however, possible to make some useful approximations of the stability dynamics of the L_3 point if one assumes that $\mu \lll \frac{1}{2}$. If, for example, if $\mu \leq 0.01$ the x coordinate of the L_3 point is very well approximated by $x \approxeq -1$ (and, of course, $y = z = 0$). Plugging this

result into the expressions for $a - f$ gives

$$\begin{aligned} a &= \tfrac{-24+50\mu-46\mu^2+25\mu^3-8\mu^4+\mu^5}{-8+28\mu-38\mu^2+25\mu^3-8\mu^4+\mu^5} \\ b &= 0 \\ c &= \tfrac{17\mu-34\mu^2+25\mu^3-8\mu^4+\mu^5}{-8+28\mu-38\mu^2+25\mu^3-8\mu^4+\mu^5} \\ d &= 0 \\ e &= 0 \\ f &= \tfrac{8-11\mu+4\mu^2}{-8+28\mu-38\mu^2+25\mu^3-8\mu^4+\mu^5} \end{aligned} \quad . \tag{A-1}$$

Since μ is small, all but the lowest term in each expression can be neglected

$$\begin{aligned} a &= 3 \\ b &= 0 \\ c &= -\tfrac{17\mu}{8} \\ d &= 0 \\ e &= 0 \\ f &= -1 \end{aligned} \quad . \tag{A-2}$$

Now the Jacobian Matrix becomes

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3 & 0 & 0 & 0 & 2 & 0 \\ 0 & -\frac{17\mu}{8} & 0 & -2 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 \end{bmatrix} \tag{A-3}$$

with corresponding eigenvalues of

$$\lambda_{L_3,\,\mu \lll \frac{1}{2}} = \begin{cases} \quad i \\ -i \\ \quad \frac{1}{4}\sqrt{-8-17\mu+\sqrt{64+1904\mu+289\mu^2}} \\ -\frac{1}{4}\sqrt{-8-17\mu+\sqrt{64+1904\mu+289\mu^2}} \\ \quad \frac{1}{4}\sqrt{-8-17\mu-\sqrt{64+1904\mu+289\mu^2}} \\ -\frac{1}{4}\sqrt{-8-17\mu-\sqrt{64+1904\mu+289\mu^2}} \end{cases} . \tag{A-4}$$

Since $\mu \lll \frac{1}{2}$, the μ^2 term can be immediately neglected, since it is very small when compared with the other terms. Next the inner square root term is expanded to Equation A-6 using the binomial expansion (Equation A-5)

$$(1+x)^r = \sum_{k=0}^{\infty} \frac{(-r)_k}{k!}(-x)^k = 1 + rx + \frac{1}{2}r(r+1)x^2 + \frac{1}{6}r(r+1)(r+2)x^3 + ... \tag{A-5}$$

$$\sqrt{64+1904\mu} = 8\left(1+\frac{119}{4}\mu\right)^{\frac{1}{2}} = 8\left[1+\frac{119}{8}\mu+\frac{1}{8}\left(\frac{119}{8}\mu\right)^2+...\right] \approx 8+119\mu. \tag{A-6}$$

Substituting this value back into the expression for the eigenvalues (and noting that $0 \leq \mu \leq \frac{1}{2}$) gives

$$\lambda_{L_3,\,\mu \lll \frac{1}{2}} = \begin{cases} i \\ -i \\ \frac{1}{4}\sqrt{102\mu} \\ -\frac{1}{4}\sqrt{102\mu} \\ \frac{1}{4}\sqrt{-16-136\mu} \\ -\frac{1}{4}\sqrt{-16-136\mu} \end{cases} \approx \begin{cases} i \\ -i \\ \frac{5}{2}\sqrt{\mu} \\ -\frac{5}{2}\sqrt{\mu} \\ \sqrt{1+\frac{17}{2}\mu}\, i \\ -\sqrt{1+\frac{17}{2}\mu}\, i \end{cases} \tag{A-7}$$

Note that there is one positive, real eigenvalue that is a function of the square root of μ. The L_3 point is, indeed, unstable but the amount of instability has now been directly expressed in terms of μ when $\mu \lll \frac{1}{2}$. The motion of a body at this L_3 point will diverge as a function of

$$e^{\lambda_{L_3} t} \approx e^{\frac{5}{2}\sqrt{\mu}\, t}. \tag{A-8}$$

The time-constant τ can be defined as $\tau = \frac{1}{\lambda} = \frac{2}{5\sqrt{\mu}}$. Since $\mu \lll \frac{1}{2}$ we know that τ is relatively large. Indeed for the Sun-Earth system $\tau \approxeq 230\,[\text{tu}]$ (or 1500 years in the Sun-Earth system). While this timescale is far to short too allow for the existence of a planet, it does bode well for space stations and spacecraft that wish to park at the Sun-Earth L_3 point and remain there for a very long time.

L_4 and L_5 Stability and Bifurcation

The stability of the L_4 and L_5 points, on the other hand, can be solved for algebraically without any approximation of μ nor any other parameter. The values of the matrix components $a - f$ are

$$for\ L_4:\ a = \frac{3}{4}, \quad b = \frac{3}{4}\sqrt{3} - \frac{3}{2}\sqrt{3}\mu \quad c = \frac{9}{4} \quad d = 0 \quad e = 0 \quad f = -1 \quad \text{(A-9)}$$

$$for L_5 : a = \frac{3}{4}, \quad b = -\frac{3}{4}\sqrt{3} + \frac{3}{2}\sqrt{3}\mu \quad c = \frac{9}{4} \quad d = 0 \quad e = 0 \quad f = -1. \quad \text{(A-10)}$$

In either case, the eigenvalues for both L_4 and L_5 turn out to be exactly the same

$$\lambda_{L_{4,5}} = \begin{cases} i \\ -i \\ \frac{1}{2}\sqrt{-2 + 2\sqrt{1 - 27\mu + 27\mu^2}} \\ -\frac{1}{2}\sqrt{-2 + 2\sqrt{1 - 27\mu + 27\mu^2}} \\ \frac{1}{2}\sqrt{-2 - 2\sqrt{1 - 27\mu + 27\mu^2}} \\ -\frac{1}{2}\sqrt{-2 - 2\sqrt{1 - 27\mu + 27\mu^2}} \end{cases} . \quad \text{(A-11)}$$

It is now helpful to define a new value, $k = \frac{m_1 - m_2}{m_1 + m_2} = 1 - 2\mu$, to simplify the eigenvalues. Note that $k \in \left(\ 0, \ \ 1 \ \right]$. Upon substitution

$$\lambda_{L_{4,5}} = \begin{cases} i \\ -i \\ \frac{1}{2}\sqrt{-2 + \sqrt{27k^2 - 23}} \\ -\frac{1}{2}\sqrt{-2 + \sqrt{27k^2 - 23}} \\ \frac{1}{2}\sqrt{-2 - \sqrt{27k^2 - 23}} \\ -\frac{1}{2}\sqrt{-2 - \sqrt{27k^2 - 23}} \end{cases} . \tag{A-12}$$

Recall that for an equilibrium point to be stable (or at least not unstable), the real part of each eigenvalue of the Jacobian Matrix must be non-positive. Looking at the eigenvalues above, it becomes apparent that they need to become entirely imaginary to satisfy this condition; if the eigenvalues had any non-zero real part, then half would be negative and half would be positive, which obviously breaks the stability condition. By inspection, note that the following conditions must be met for all eigenvalues to be imaginary

$$\begin{cases} \sqrt{27k^2 - 23} \leq 2 & (1) \\ 27k^2 - 23 \geq 0 & (2) \end{cases} = \begin{cases} k \leq 1 & (1) \\ k \geq \sqrt{\frac{23}{27}} & (2) \end{cases} . \tag{A-13}$$

Since $k \in \left(\ 0, \ \ 1 \ \right]$, condition (1) is always satisfied. Condition (2) implies that $0 \leq k \leq 0.9229$ or $0 \leq \mu \leq 0.0385$ or $0 \leq \frac{m_2}{m_1} \leq 4\%$ (since $\frac{m_2}{m_1} = \frac{\mu}{1-\mu}$). This is a remarkable result. The stability of the L_4 and L_5 equilibrium points is a function of μ only. The system will bifurcate, going from stable to unstable,

once the critical value of $\mu_C = \frac{1}{2} - \frac{\sqrt{69}}{18} \approxeq 0.0385$ has been crossed. Note that most three body systems within the solar system are below μ_C, with the most notable exception being the Pluto-Charon system where $\mu = 0.1$.

Vita

Andrew J. Abraham was born on August 4^{th}, 1986 to Joseph A. Abraham, Jr. and Terry-Ann (Richards) Abraham in Bethlehem, Pennsylvania. He graduated from Northampton Area Senior High School before attending Moravian College where he received a B.A. in Economics and a B.S. in Physics in 2009. Dr. Abraham graduated "summa cum laude" with "honors in physics." He was awarded this honor for his research entitled "*The Crystallite Size Distribution in 2-D Beds of Randomly Close-Packed Binary Beads*." He then spent a summer at NASA Glenn Research Center where he participated in the NASA Space Academy internship program working on a "microwave power combiner" for use in deep space, $k_\alpha-$band communications.

Dr. Abraham attended graduate school at Lehigh University where he finished an M.S. degree in Physics in 2011. He became involved with Penn State University's "Lunar Lion" program via a partnership with Lehigh University's Hopper Spacecraft Simulator. Dr. Abraham's role was to design a preliminary trajectory that would enable the spacecraft to soft-land on the lunar surface and "hop" 500 meters. He spent the summer of 2013 working at NASA Goddard Space Flight Center where he concentrated on proximity operations for the OSIRIS-REx spacecraft as it investigated asteroid 101995 Bennu. He later received the John Mather Nobel Scholar award for this work. Dr. Abraham proceeded to publish three papers on the topic of low-thrust trajectory optimization, using Particle Swarm Optimization, and was awarded his Ph.D. in Mechanical Engineering from Lehigh University in May, 2014.