NUMERICAL METHODS FOR LOW-THRUST TRAJECTORY OPTIMIZATION

A Thesis

Submitted to the Faculty

of

Purdue University

by

Robert E. Pritchett

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

August 2016

Purdue University

West Lafayette, Indiana

For Grandma, Grandpa, Nana, and Pop

ACKNOWLEDGMENTS

I could fill another thesis thanking everyone who has made it possible for me to reach this stage, however I am told I have a deadline so I must be brief. I must first and foremost thank Mom, Dad, and Sam for their steadfast love and support. Mom and Dad your example of love, faith, and dedication has made me the person I am today and this achievement is just as much a reflection of your hard work as mine. Thank you for all that you have done for me. And Sam, your friendship, encouragement, and wrestling matches, have been a continual source of joy, may we never outgrow any of it. Additionally, the love and support of my grandparents, both Grandma and Grandpa as well as Nana and Pop, has positively shaped me in innumerable ways.

I owe nearly as much thanks to my extended "family" for whom I am deeply grateful. The friendship, guidance, and love that has been offered me by the Millards, Buckners, Ogletrees, Reds, Mitras, Pinegars and so many more families at Nassau Bay Baptist Church has carried me through every period of my life thus far. I cannot think of a more loving community for a child to grow up in and my gratitude runs deeper than words can express.

I am also incredibly thankful for the support and guidance of my advisor Professor Kathleen Howell. Professor Howell you have afforded me numerous opportunities to learn and grow and followed these with a confidence in my ability that has propelled me past my own diffidence. You are both a superb mentor and scholar and I look forward to continuing to learn from your example. I would also like to thank my committee members Professor James Longuski and Professor Carolin Frueh, the time you have taken to offer feedback on this thesis has significantly improved the quality of the final result. In addition, I am grateful to the School of Aeronautics and Astronautics for the past two years of financial support and the opportunity to work and learn at Purdue University.

Furthermore, it has been a true privilege to work with and learn from Dan Grebow and Tom Pavlak during my internships at JPL. Much of the work in this thesis began with what I learned under Dan's exceptional mentorship and with Toms's unselfish support. Dan your passion for astrodynamics along with your humility and character are genuine sources of inspiration for me, and I hope to continue to learn from you. Tom thank you for your kindness and constant willingness to lend a hand. The examples both of you provide have shown me not only how to be a better astrodynamicist, but a better friend and coworker as well. I am also thankful for the many other exceptional individuals I have met at JPL, such as Jeff Stuart and Mar Vaquero, who have generously shared their time and knowledge with me.

My graduate school experience thus far would be quite empty if not for the outstanding friends I have found along the way. To my research groupmates Andrew, Alex, Ash, Bonnie, Emily, Ted, Natasha, Cody, Rohan, Shota, Chris, Davide, Wayne, Kia, and Loic, thank you for your friendship and all that you have taught me. I owe a special thank you to Ash and Bonnie for taking the time to explain so many low-thrust and optimization concepts to me. To all of my Graduate InterVarsity friends, thank you for your friendship and encouragement. Time spent with you has been a constant source of discovery and rejuvenation. I am excited to continue working and enjoying life with all of you.

These are simply the many individuals who have helped me in graduate school thus far, not to mention the profoundly influential mentorship of Professor Lightsey and Susan Winnitoy, as well as the support of so many friends from Longhorn Band and AAE during my time at The University of Texas. I am truly thankful for the many hands that have made it possible for me to complete this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Pritchett, Robert E. MSAA, Purdue University, August 2016. Numerical Methods for Low-Thrust Trajectory Optimization . Major Professor: Kathleen C. Howell.

The spacecraft trajectory design process frequently includes the optimization of a quantity of importance such as propellant consumption or time of flight. A variety of methods for trajectory optimization are available, however the efficiency of an approach is dependent on the problem scenario it is applied to. Indirect and direct trajectory optimization methods are examined in this investigation with the goal of assessing the characteristics of each approach, and thereby determining the problem scenarios each is best suited for. Insight is gained from application of each optimization method to three sample problems; a circular-to-circular orbit transfer as well as two variants of a halo-to-halo orbit transfer, one that leverages manifold arcs and one that does not. The analytical theory underlying indirect optimization methods is presented as is the adjoint control transformation for determining initial costate values. Results from application of the indirect optimization approach to each of the sample problems are offered. The framework of a direct optimization scheme employing collocation is described including a mesh refinement process based on the de Boor update method. The direct optimization method is applied to the three sample problems and results are supplied. Quantitative comparisons of the results of the optimization methods are made based on the categories of accuracy, robustness, and efficiency. Findings from quantitative and qualitative comparisons of the optimization methods are employed to formulate guidelines on the problem scenarios each technique is most applicable to.

1. INTRODUCTION

The recent success of missions employing low thrust propulsion systems has demonstrated the promise for this technology over a wide array of future applications, from transportation networks within the Earth-Moon neighborhood, to Mars missions, to tours of the Trojan asteroids. The high specific impulse of low-thrust engines yields orders of magnitude more fuel efficiency than their conventional high-thrust counterparts. Of course, this decrease in the rate of fuel consumption is offset by increased times of flight. Nevertheless, low-thrust engines remain an excellent option for many potential mission scenarios. Only in the last few years, NASA's Dawn spacecraft emerged as the first spacecraft to orbit two different extraterrestrial bodies when it encountered the dwarf planets Ceres and Vesta located in the asteroid belt. This mission would not have been feasible without the three low-thrust ion engines Dawn was equipped with. The potential of low-thrust spacecraft has only begun to be realized, further astrodynamics research will expand the space attainable by these spacecraft, opening up new regions of space for science and exploration.

Trajectory design for low-thrust propulsion systems presents unique new challenges to the mission design community. Perhaps not surprisingly, one approach to addressing these challenges is incorporating optimization methods into the process. The benefits that optimization methods provide are numerous, but they also bring with them new challenges. The field is vast with applications far beyond trajectory design, therefore selecting the appropriate optimization method for a given problem can be challenging. A complete understanding of both the problem and the potential optimization approaches is necessary to ensure an efficient design process.

1.1 Problem Definition and Motivation

Spacecraft motion is governed by a sensitive system of nonlinear differential equations and, incorporating low-thrust forces into this system, adds a new layer of challenges in constructing desirable trajectories. Impulsive thrust maneuvers are traditionally modeled as instantaneous, however the fundamental nature of low-thrust propulsion systems necessitates an alternative formulation. A single low-thrust maneuver requires continuous thrusting, and therefore, a continuous control history to define the pointing, and possibly the thrust level, at each instant during the maneuver. This difference in comparison to an instantaneous, impulsive thrust model results in many more design variables and a less intuitive problem overall.

The new challenges involved in low-thrust trajectory design lead directly to the frequent introduction of optimization techniques into the design process. Optimization methods yield state and control variables along a path that minimize a scalar cost function. These types of strategies are especially useful in the low-thrust trajectory design process because they offer guidance in the selection of values for numerous control variables. To leverage the advantages of optimization techniques, the best-suited method is sought. But, the field of trajectory optimization is diverse and expanding. At present, most optimization schemes fall into one of three categories: indirect methods, direct methods, and evolutionary algorithms. Indirect methods using Euler-Lagrange theory and direct approaches employing collocation are both explored for the applications in this investigation.

As with any design process, trajectory design is most efficient and effective when the proper tools are employed. Each optimization technique, as well as the underlying numerical algorithm, offers advantages and disadvantages. Two specific optimization techniques are described and demonstrated; their relative strengths and weaknesses are highlighted by comparing the results for several sample problems. The process of comparing and contrasting the techniques should yield a deeper understanding of them and thus enables mission designers to use these tools appropriately.

1.2 Previous Work

The benefits of low-thrust propulsion were understood long before the technology was sufficiently mature for an actual mission. Therefore much literature is available on the topics of trajectory optimization and low-thrust transfer design. However, it is first important to understand the context in which it will be applied, i.e., the dynamical model.

1.2.1 History of the Three-Body Problem

A description of the time-dependent behavior of three gravitationally interacting bodies was first mathematically formalized in 1687 by Issac Newton in his foundational work *PhilosohphæNaturalis Principae Mathematica* [1]. Overtime, the formulation of this problem has come to be denoted the three-body problem. The fundamentals established by Newton have served as the basis for investigation over the subsequent centuries by numerous individuals, some motivated by mathematical curiosity and others by engineering necessity. Nearly one hundred years after Newton, in 1772, Leonhard Euler offered further insight into the problem via several simplifying assumptions leading to the *restricted* three-body problem. Euler's principal contributions to the restricted three-body problem included the introduction of a rotating (synodic) reference frame and the solution for the three collinear equilibrium points. Mere months after the publication of Euler's work Joseph-Louis Lagrange identified the remaining two equilibrium points known as the triangular or equilateral points. As a part of this work, Lagrange defined all five equilibrium points specifically within the context of the restricted three-body problem consequently, these five points are often termed Lagrange points.

The rotating frame introduced by Euler resulted in a formulation that allowed an integral of the motion which was formally identified by Carl Gustav Jacob Jacobi. This integral, eventually labelled the Jacobi integral, is extraordinarily useful because it allows qualitative statements about behavior in the restricted three-body problem without the solution to the differential equations. George William Hill leveraged the qualitative applications of the Jacobi integral in his 1878 work *Researches in the Lunar Theory* [2] where he demonstrated that forbidden regions in the Sun-Earth-Moon system bound the Earth Moon distance for all time and that the geometry of these regions is dependent on the value of the Jacobi integral. The borders of these forbidden regions are defined by zero-velocity surfaces, a useful qualitative tool for investigating the three-body problem. Surfaces of section, introduced by Henri Poincaré in his three-volume work *Methodes Nouvelles* [3], have proven another invaluable qualitative tool for this purpose. Poincaré's 1899 magnum opus has proven foundational to much of modern dynamical systems theory. Subsequently, the mathematician George Birkhoff expanded upon the ideas of Poincaré in his article *Proof of Poincaré's geometric theorem* [4]. Finally, in 1967, Victor Szebehely consolidated much of the fundamental work on the restricted three-body problem in his seminal 1967 work *The Theory of Orbits* [5].

1.2.2 Optimization Methods

At the current time, astrodynamics is motivated not only to understand the motion of the heavenly bodies, but also to successfully navigate throughout the solar system. Any trajectories moving throughout the Earth-Moon neighborhood or further into the solar system must satisfy specific constraints. Ideally, such trajectories minimize parameters such as the propellant consumed or the time in transit. Incorporating optimization strategies into the design process enables the identification of trajectories that extremize parameters such as these. The focus in this investigation is twofold, i.e., indirect and direct optimization.

Indirect Optimization:

Indirect optimization approaches originate with the calculus of variations. Many define the origin of the calculus of variations as an intriguing problem posed by Johann

Bernoulli to the mathematical community in 1696. The objective of this problem, titled the brachistochrone problem, is to determine a path (function) that minimizes a scalar function of that path (functional). The brachristochrone problem was solved after 6 months (and was solved by Newton in one day), however, it continued to intrigue mathematicians and remains a useful example problem [6]. Early in the 18th century, a form of this problem inspired Lagrange to develop a method of determining a function that minimizes a functional; the resulting strategy is, today, the essential definition of the calculus of variations. After the initial development, the calculus of variations approach was refined through a correspondence between Lagrange and Euler that ultimately led to the Euler-Lagrange Theorem.

A recent application of the calculus of variations, i.e., transfer of a satellite between circular orbits, is similar to the original brachistochrone problem. However, such a transfer is complicated by the addition of a control variable that determines the pointing direction of the satellite thrust vector. Moreover, when it is initially posed as a trajectory optimization problem, some of the final boundary conditions are free. In his 1963 book *Optimal Spacecraft Trajectories* [7] Lawden demonstrated that such problems can be transformed to two-point boundary value problems (TPBVP). Two-point boundary value problems are often be solved numerically and, in fact, Bryson and Ho demonstrate the proper application of the Euler-Lagrange theorem to produce a well defined TPBVP [8]. The same basic methodology is employed to solve more complex problems such as transfers from the Earth to the Moon [9] as well as transfers between periodic orbits employing invariant manifolds [10].

Direct Optimization:

Indirect methods have proven effective in solving a variety of continuous optimal control problems by transforming them to two-point boundary value problems, however, such approaches possess several drawbacks that are addressed by direct optimization methods. Direct optimization strategies discretize the continuous optimal control problem thereby reformulating it as a nonlinear programming problem (NLP) and, thus, making it tractable to a wider range of numerical optimization approaches. The process of discretizing an optimal control problem is denoted as direct transcription, a term coined by Canon et al. in 1970 [11]. While the process was familiar to mathematicians such as Canon in the 1960s and 1970s, it was not until the mid-1970's that the technique gained prominence in the aerospace community originating with a paper by Dickmanns and Wells [12]. Dickmanns and Wells used direct transcription to solve optimal control problems, but formulated these problems using analytical techniques from indirect methods. Over a decade later, Hargraves and Paris [13] demonstrated that the step of formulating an optimal control problem as a TPBVP could be skipped all together. This realization was extraordinarily useful as it eliminated the sensitive adjoint variables in the process of solving an optimal control problem. Now, direct solution methods are assumed to be approaches that avoid the use of adjoint or costate variables. The adoption of direct transcription methods for solving optimal control problems within the aerospace community increased in parallel with computational power. Since its initial introduction, numerous schemes to implement direct transcription have been proposed; the primary differences between these various strategies are the type of integration rules employed. One of the most popular schemes for direct transcription is collocation.

Direct Transcription with Collocation:

Collocation methods are one of the primary techniques for solving direct transcription problems. These methods fit piecewise polynomials to a discretized optimization problem, with the polynomial fit governed by the problem dynamics and other constraints. A formulation based on the collocation technique was perhaps first accomplished by de Boor in 1966 [14] when he used it to solve boundary value problems for linear differential equations. Russell and Shampine [15] expanded the application of the method to boundary value problems with ordinary differential equations in

1972 and coincident with Richard Weiss demonstrating that collocation can produce results equivalent to those produced from implicit Runge-Kutta methods [16]. These developments led directly to Dickmann and Wells [12] application of collocation to optimal control problems posed as TPBVP.

Initial applications of collocation schemes to optimal control problems primarily employed cubic polynomials, however, research in the 1990s improved the method's robustness by utilizing higher order polynomials and more accurate node placement rules. Enright and Conway [17] applied a low order Gauss Lobatto rule to define the discretization of an optimal control problem and this rule yields more accurate results than simply fitting a polynomial to an equally spaced discretization. Herman and Conway [18] demonstrated that the error associated with a discretization decreased as the order of the polynomial used to fit the discretization increased, offering results up through seventh degree polynomials. Williams then developed an approach to embed polynomials of any order in a collocation algorithm [19]. Direct transcription can produce nonlinear programming problems involving extremely sparse matrices, especially as the degree of the polynomial increases. Betts and Huffman leveraged this feature in numerical techniques to decrease computation times [20] and developed a software package to solve optimal control problems [21]. The increasingly robust approaches to the collocation technique described were incorporated into a variety of software packages for solving optimal control problems, for example, Optimal Trajectories by Implicit Simulation (OTIS) [22], used by the US Air Force and NASA.

1.2.3 Thesis Overview

The focus of this work is a general dynamical model and set of compact collocation strategies that are subsequently employed to examine several types of spacecraft trajectory optimization problems.

- **Chapter 2:** The dynamical model used throughout the current investigation, namely, the circular-restricted three body problem, is presented. The differential equations that govern motion are developed originating with Newton's general N-body problem. The assumptions that result in equations of motion for the three-body problem are summarized and justified. Unique features of this formulation, such as equilibrium points, the Jacobi constant, and zero velocity surfaces are also discussed. Linear variational equations are constructed relative to the equilibrium solutions and used to analyze the stability of these points.

- **Chapter 3:** Analytical and numerical approaches useful for exploring the dynamical model are examined. The linearized variational equations are the basis to develop the state transition matrix and the differential corrections processes for single and multiple shooting. A general framework for these strategies is presented. Differential corrections methods enable the construction of periodic orbits. The stability of these orbits is analyzed and a continuation method is developed to compute families of periodic orbits. A procedure for constructing the invariant manifolds associated with the equilibrium points and periodic orbits is detailed.

- **Chapter 4:** Indirect optimization as formulated via the Euler-Lagrange theory is presented. The specific application of Euler-Lagrange theory to low-thrust transfer design is detailed along with the advantageous adjoint control transformation. Low-thrust circular to circular orbit transfer and halo to halo orbit transfer problems are included to demonstrate the methodology.

- **Chapter 5:** The foundational theory for direct optimization methods is briefly summarized. The primary focus of this chapter is the specific method of direct transcription with collocation. The implicit integration scheme, i.e., collocation, is described first, then, its application to an optimization strategy with direct transcription is demonstrated. Finally, the procedure involving direct transcription is demonstrated using two examples: (i) low-thrust circular to circular orbit transfer and (ii) halo-to-halo orbit transfer problems.

- **Chapter 6:** The results from low-thrust transfer problems solved using indirect and direct optimization methods are compared. The sample cases include a circular to circular orbit transfer and halo-to-halo orbit transfer design. Comparisons between solutions are based on the cost function as well as other problem parameters including accuracy, robustness, and efficiency. The qualitative advantages and disadvantages of both approached are discussed and a framework for selecting the method best-suited for a given scenario is proposed.

- **Chapter 7:** A brief summary of the work is presented, including the applicability of the two optimization strategies to different sample scenarios. Finally, recommendations for future work are proposed.

2. DYNAMICAL MODEL

Prior to any analysis and development of trajectory control strategies for the motion of a spacecraft, a dynamical model must be constructed. A dynamical model offers a mathematical description of the laws that govern the motion and interaction of bodies. Over time, mathematicians and physicists have improved the accuracy and efficiency with which dynamical models describe the motion. However, the purpose is not always to describe the motion of bodies with the greatest degree of accuracy; rather, simplified models that roughly approximate the motion of bodies are often useful because their simplicity allows for greater insight into the essential interactions occurring within a system. Simplified models, for example the two or three body problems, are constructed given a set of reasonable assumptions. Once a dynamical model is available, examination of the system for integrals and equilibrium solutions is useful for understanding the underlying structure of the solution space.

2.1 The N-Body Problem

The most general dynamical model to incorporate all gravitational forces as point mass sources is the N-Body problem. This model was formally introduced in 1687 by Issac Newton in his groundbreaking work *PhilosohphæNaturalis Principae Mathematica* [1]. In Book I, Newton introduced his three laws of motion that serves as the foundation for much of modern dynamics. The law of motion states that the force impressed on a body is proportional to, and in the same direction, as the derivative of the body's momentum. The law is expressed mathematically in vector form as,

$$\boldsymbol{F} = m\ddot{\boldsymbol{r}} \tag{2.1}$$

where $\boldsymbol{F}$ is the vector sum of all forces acting on the particle mass m and $\ddot{\boldsymbol{r}}$ is the vector acceleration of the mass as observed from an inertial reference frame. Vector quantities are always denoted using boldface type; scalar quantities are italic. Note in equation (2.1), that m is a constant scalar, thus, this relationship applies only to fixed-mass systems. Elsewhere within the *Principia*, Newton formulated his Universal Law of Gravitation.

$$|\boldsymbol{F}| = \left| -\frac{GMm\boldsymbol{d}}{d^3} \right| \tag{2.2}$$

This model for the gravitational force on a single particle mass m due to the existence of mass M when the relative distance between the bodies is d. A single particle m_i is located within a system of N other bodies, as demonstrated in Figure 2.1. Note that the position of m_i relative to particle m_j is denoted $\boldsymbol{d}_{ji}$. Thus, the force on m_i due to the existence of particle j is directed as described by the direction $-\boldsymbol{d}_{ji}/d_{ji}$. The total gravitational force acting on particle i is then obtained by summing the individual gravitational forces. This generalized form of Newton's Universal Law of Gravitation is,

$$\boldsymbol{F}_i = -G \sum_{j=1, j\neq i}^{n} \frac{m_i m_j}{d_{ji}^3} \boldsymbol{d}_{ji} \tag{2.3}$$

where $j = i$ is excluded from the summation because the body obviously cannot exert a force on itself.

Newton's model for gravity the law of motion are combined to produce the differential equation of motion for particle i in a system of N bodies,

$$m_i \ddot{\boldsymbol{r}}_i = -G \sum_{j=1, j\neq i}^{n} \frac{m_i m_j}{d_{ji}^3} \boldsymbol{d}_{ji} \tag{2.4}$$

as appears in Figure 2.1, the vector $\boldsymbol{r}_i$ in equation (2.4) is the position vector from an inertially fixed origin to the mass m_i, while the vector $\boldsymbol{d}_{ji}$ is the vector extending from m_j to m_i. Thus, the relative displacement $\boldsymbol{d}_{ji}$ is given by $\boldsymbol{d}_{ji} = \boldsymbol{r}_i - \boldsymbol{r}_j$.

Collecting differential equations of the form in (2.4) for each particle in an N-body system can become quickly intractable when N is large. The simplest nontrivial case of this is the two-body problem, a focus for mathematicians for hundreds of years. This simple model allows for analytical solutions, some of which were described by Johann Kepler and his predecessors even before the time of Newton. The closed form

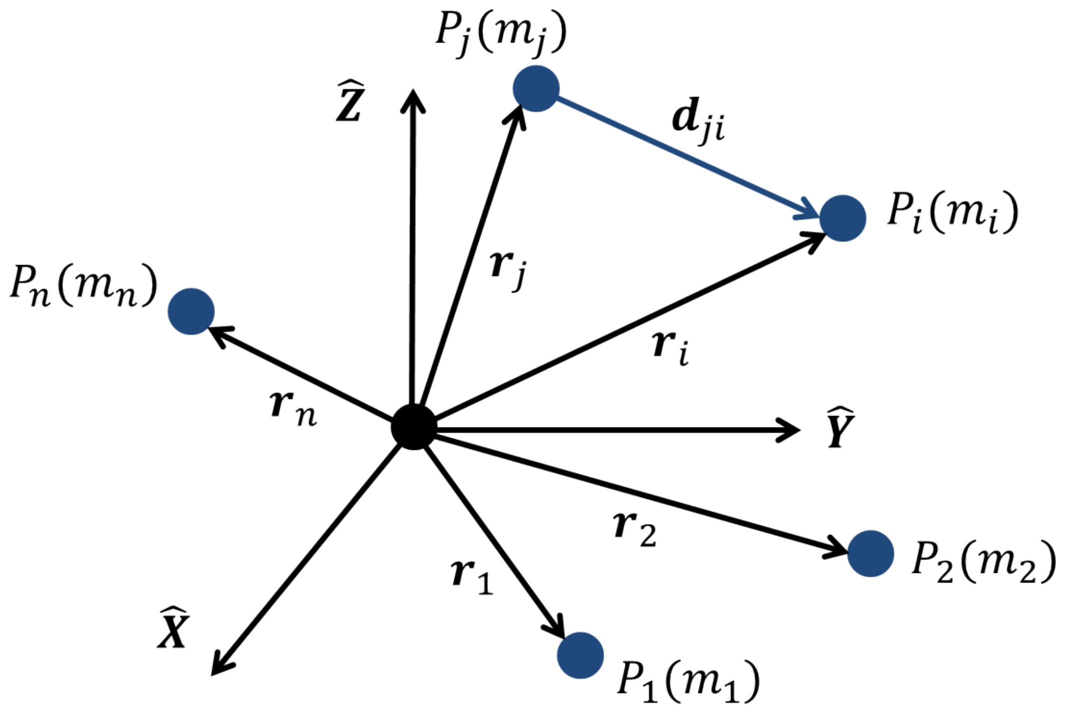


Figure 2.1.: N-Body System

solutions in the two body problem are readily applicable to celestial mechanics and render reasonably accurate approximations for the motion of many celestial bodies. Additionally, these solutions were especially useful when computational capabilities were much more limited. However, the rapid advancement of computing power over the last 75 years, has enabled feasible examination of motion in more complex dynamical models and a new and diverse set of tools for understanding gravitational interactions has emerged.

2.2 The Circular-Restricted Three Body Problem

Although the two-body model has proven extraordinarily useful, a primary limitation is a piecemeal approach to mission design. Additionally, the simultaneous effects of multiple gravitational fields not in the two-body model are typically only included as perturbations. While this approach is frequently effective and successful in particular dynamical regimes, it obscures the impact of these additional forces when they contribute more than perturbations and can be leveraged to achieve mission design objectives. Admitting, just one additional gravitational field into the dynamical model yields the three-body problem (3BP). This expanded model possesses no analytical solution and even by 1900 it was clear that insight and understanding into the three-body problem requires a fundamental shift in approach. First, the problem is reduced to its most essential elements, a process that yields the circular restricted three-body problem (CR3BP). Analysis in the CR3BP has produced a wealth of dynamical insights that, in turn, have resulted in innovative approaches to mission design. The CR3BP model is particularly useful for low-thrust trajectory design, and for this reason, it is the primary dynamical model explored in this investigation.

2.2.1 Assumptions

Derivation of the equations of motion for the CR3BP begins by selecting $N = 3$ in equation (2.4). This specification limits the number of active gravitational fields to three, corresponding to three particles P_1-P_3. Assume that the motion of P_3 is the focus. Then,

$$m_3 \ddot{\boldsymbol{r}}_3 = -\frac{G m_3 m_1}{d_{13}^3}\boldsymbol{d}_{13} - \frac{G m_3 m_2}{d_{23}^3}\boldsymbol{d}_{23} \tag{2.5}$$

represents the differential equation to model the behavior of P_3. Solving for the motion of P_3 from equation (2.7) requires knowledge of the time histories of P_1 and P_2 but, since these particles are themselves influenced by the motion of P_3, such information is generally not available *a priori*. Therefore, to solve equation (2.7) analytically, the equations of motion for all three particles must be solved simultaneously. Using Cartesian coordinates, this integration requires six integrals per particle, three for position and three for velocity, necessitating 18 total constants of integration. However, only 10 constants of integration are known to exist in this problem; six are obtained from conservation of linear momentum, three from conservation of angular momentum, and one from conservation of energy. Due to the insufficient number of integration constants, a time history for the motion of all three bodies is not available analytically. Nevertheless, several assumptions reduce the problem to a more tractable form.

Three key assumptions reduce the complexity of the three-body problem. A useful assumption in the simplification process is that the mass of the third particle, P_3, is infinitesimal compared to the masses of P_1 and P_2, denoted the "primaries". This assumption implies that the motion of the primaries is not influenced by P_3. Such an assumption is reasonable for some important applications, for example, the path a spacecraft or comet under the gravitational impact of the Sun and a planet. The resulting assumption allows the primary motion to be modeled in terms of conics. Finally, from the large set of potential closed conics, assume that the primary system orbit is closed but also circular. Once again this assumption is reasonable for many celestial systems of interest such as the Earth-Moon or Sun-Jupiter systems where the relative orbit eccentricity is very small. Conventionally, the mass of the first primary is assumed to be greater than that of the second, $m_1 > m_2$. Therefore, the primaries orbit about a common barycenter located near P_1 as shown in Figure 2.2. Together, these assumptions reduce the three-body problem to the circular-restricted three-body problem (CR3BP) and simplify the dynamical model while reasonably approximating motion in a three body system.

2.2.2 Coordinate Frames

Motion in a dynamical model is defined relative to a reference frame, and intelligent selection of this frame, rather than an arbitrary choice, produces a more tractable and intuitive problem definition. Many dynamical models include an inertial reference frame that is theoretically at rest or moving at a constant linear velocity. Define an inertial reference frame, I, with origin fixed at the barycenter of the primary system and unit vectors $\hat{X}$ and $\hat{Y}$ spanning the fixed plane of motion of P_1 and P_2. A third unit vector, $\hat{Z}$, is defined such that I is a right-handed coordinate system as

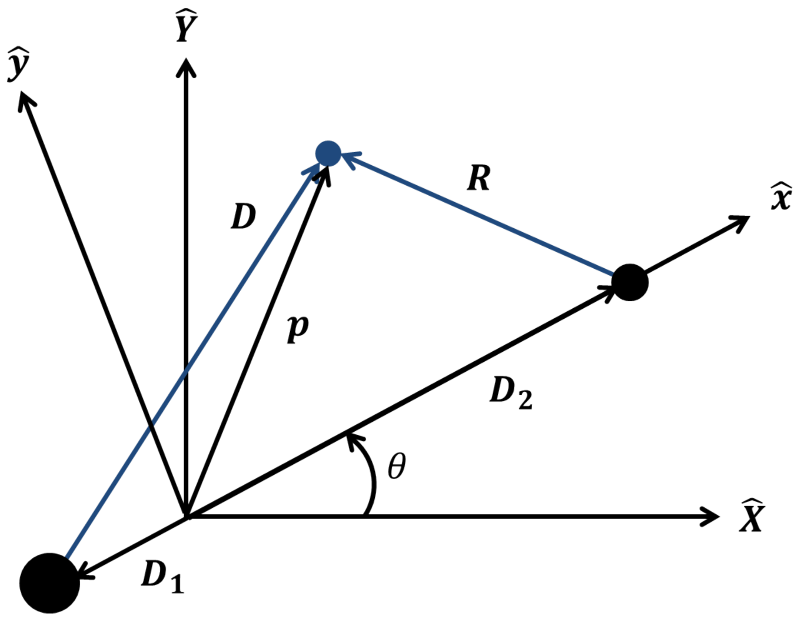


Figure 2.2.: Circular-Restricted Three-Body Problem

illustrated in Figure 2.2. The out-of-plane unit vector, $\hat{Z}$, is aligned with the orbital angular momentum vector of the primary system.

Viewing motion from a reference frame that rotates with the primary system assists the understanding and analysis of dynamical behavior. Define a rotating coordinate system, R, one that is initially aligned with the inertial system, I. The frame R represents a simple rotation about the out-of-plane direction, i.e., $\hat{z}$ and through the angle θ. The frame R is defined by the orthonormal triad $\hat{x}$,$\hat{y}$,$\hat{z}$. The $\hat{x}$ axis of R is defined along the the line passing through the primaries and is directed toward P_2; $\hat{z}$ remains aligned with $\hat{Z}$. Finally, $\hat{y}$ completes the right handed coordinate system, thus, it is perpendicular to $\hat{x}$ and in the plane of motion of the primaries. The time rate of change $\dot{\theta}$, is the magnitude of the angular velocity of the primary system, ${}^{I}\boldsymbol{\omega}^{R} = \dot{\theta}\hat{z}$. Because the path of the primaries is circular, $\dot{\theta}$ is a constant value. The inertial and rotating coordinate frames are related by the angle θ, therefore vectors are transformed between frames using a simple direction cosine matrix (DCM).

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix} = \begin{bmatrix} \cos(\dot{\theta}t) & -\sin(\dot{\theta}t) & 0 \\ \sin(\dot{\theta}t) & \cos(\dot{\theta}t) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x \\ y \\ z \end{Bmatrix} \tag{2.6}$$

For convenience, equation (2.5) is rewritten using the notation of the rotating coordinate frame depicted in Figure 2.2,

$$m_3\ddot{\boldsymbol{p}} = -\frac{Gm_3m_1}{D^3}\boldsymbol{D} - \frac{Gm_3m_2}{R^3}\boldsymbol{R} \tag{2.7}$$

where $\boldsymbol{D} = \boldsymbol{d}_{13}$ and $\boldsymbol{R} = \boldsymbol{r}_{23}$ and vectors from the barycenter to P_1, P_2, and P_3 are denoted D_1, D_2, and p respectively. Defining the motion of P_3 relative to the rotating frame R simplifies the expression of the equations of motion for the CR3BP.

2.2.3 Equations of Motion

Analytical and numerical analysis in the CR3BP is simplified by nondimensionalizing the quantities in equation (2.7). Several characteristic quantities are defined, one for each type of fundamental measurement encountered in the differential equations. Distance is nondimensionalized using the characteristic length, l^*, defined as the constant distance between the primaries.

$$l^* = D_1 + D_2 \tag{2.8}$$

where D_i is the scalar distance from the barycenter to the primary P_i. Likewise, the characteristic mass, m^*, is defined as the sum of the masses of P_1 and P_2, i.e,

$$m^* = m_1 + m_2 \tag{2.9}$$

The characteristic time, t^*, is evaluated such that the nondimensional value of the universal gravitational constant, $\widetilde{G}$, is equal to one. Then,

$$t^* = \left[\frac{(D_1 + D_2)}{\widetilde{G}(m_1 + m_2)}\right]^{1/2} = \left[\frac{l^*}{\widetilde{G}m^*}\right]^{1/2} \tag{2.10}$$

This choice for the definition of characteristic time also simplifies nondimensional mean motion. Recall that the motion of P_2 with respect to P_1 is assumed to be circular, therefore, the dimensional mean motion, N, is computed as,

$$N = \left(\frac{\widetilde{G} m^*}{l^{*3}} \right) \tag{2.11}$$

It follows, then, that the nondimensional mean motion, n, is equal to unity,

$$n = Nt^* = \left(\frac{\widetilde{G} m^*}{l^{*3}} \right) \left(\frac{l^{*3}}{\widetilde{G} m^*} \right) = 1 \tag{2.12}$$

The period, P, of a circular orbit is related to mean motion as $P = 2\pi/n$, thus the nondimensional period of the primaries in the CR3BP is 2π.

Once all the necessary characteristic quantities are defined, the equations of motion are nondimensionalized and simplified. The nondimensional mass of P_2 is defined as the mass ratio, μ, and this ratio is also used to express the nondimensional mass of P_1. Nondimensional time is also defined, such that all derivatives are evaluated with respect to nondimensional time:

$$\mu = \frac{m_2}{m^*} \tag{2.13}$$

$$m_1 + m_2 = m^* = m_1 + \mu m^* \rightarrow 1 - \mu = \frac{m_1}{m^*} \tag{2.14}$$

$$\tau = \frac{t}{t^*} \tag{2.15}$$

In addition to simplifying the equations of motion, the mass ratio, μ, is useful to characterize a CR3BP system. The Earth-Moon system, the primary focus of this investigation, is represented in terms of a mass ratio $\mu = .01215$ which is relatively large compared to other CR3BP systems, for examples, Saturn-Titan where $\mu = 0.000237$ and Sun-Jupiter with $\mu = 0.000954$. This difference in μ implies that dynamic features that appear in the Earth-Moon system may not appear in lower mass ratio systems and vice versa.

The equations of motion are simplified by leveraging characteristic quantities in the expression of vectors. Vector quantities that appear in the equations of motion are nondimensionalized, i.e.,

$$\boldsymbol{d}_i = \frac{\boldsymbol{D}_i}{l^*} \tag{2.16}$$

$$\boldsymbol{d} = \frac{\boldsymbol{D}}{l^*} \tag{2.17}$$

$$\boldsymbol{r} = \frac{\boldsymbol{R}}{l^*} \tag{2.18}$$

The nondimensional vectors $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ describe the position relative to the barycenter of P_1 and P_2, respectively, from the perspective of the rotating coordinate frame. The geometry of the system depicted in Figure 2.2 and the definition of the barycenter indicate that the nondimensional relative positions of P_1 and P_2 are expressed in terms of the mass ration; μ, that is,

$$\boldsymbol{d} = (x + \mu)\hat{x} + y\hat{y} + z\hat{z} \tag{2.19}$$

$$\boldsymbol{r} = (x - 1 + \mu)\hat{x} + y\hat{y} + z\hat{z} \tag{2.20}$$

The final vector quantity in equation (2.7) is $\boldsymbol{p}$ which is nondimensionalized and locates the inifnitesimal particle as,

$$\boldsymbol{\rho} = \frac{\boldsymbol{p}}{l^*} = x\hat{x} + y\hat{y} + z\hat{z} \tag{2.21}$$

where the time derivative of $\boldsymbol{\rho}$ with respect to the rotating reference frame R is,

$$\frac{d^R\boldsymbol{\rho}}{dt} = \dot{x}\hat{x} + \dot{y}\hat{y} + \dot{z}\hat{z} \tag{2.22}$$

The nondimensional counterparts of the components of equation (2.7) are assembled into the nondimensional representation of second-order vector equation of motion,

$$\frac{{}^I d^2\boldsymbol{\rho}}{d\tau^2} = \boldsymbol{\rho}'' = -\frac{(1-\mu)}{d^3}\boldsymbol{d} - \frac{\mu}{r^3}\boldsymbol{r} \tag{2.23}$$

Note that equation (2.23) denotes $\boldsymbol{\rho}''$ as the acceleration observed relative to the inertial frame.

Analysis in the CR3BP is simplified by working in the rotating coordinate frame, therefore, it is more convenient to express $\boldsymbol{\rho}''$ relative to the rotating coordinate frame. The Basic Kinematic Equation (BKE) relates derivatives relative to two different coordinate frames. The BKE is applied twice to obtain an expression that relates $\boldsymbol{\rho}''$ as viewed in the inertial and rotating frames, i.e.,

$$\frac{{}^{I}d\boldsymbol{\rho}}{d\tau} = \frac{{}^{R}d\boldsymbol{\rho}}{d\tau} + {}^{I}\boldsymbol{\omega}^{R} \times \rho \tag{2.24}$$

$$\frac{{}^{I}d^2\boldsymbol{\rho}}{d\tau^2} = \frac{{}^{R}d^2\boldsymbol{\rho}}{d\tau^2} + 2{}^{I}\boldsymbol{\omega}^{R} \times \frac{{}^{R}d\boldsymbol{\rho}}{d\tau} + {}^{I}\boldsymbol{\omega}^{R} \times {}^{I}\boldsymbol{\omega}^{R} \times \rho \tag{2.25}$$

The angular velocity of the rotating frame relative to the inertial frame reflects the nondimensional mean motion, n, hence ${}^{I}\boldsymbol{\omega}^{R} = n\hat{z}$ with constant magnitude. This value is substituted into equation (2.25), along with the expression for $\boldsymbol{\rho}$ defined in equation (2.21), such that

$$\frac{{}^{I}d^2\boldsymbol{\rho}}{d\tau^2} = (\ddot{x} - 2n\dot{y} - n^2x)\hat{x} + (\ddot{y} + 2n\dot{x} - n^2y)\hat{y} + \ddot{z}\hat{x} \tag{2.26}$$

Equation (2.26) reflects the inertial acceleration of $\boldsymbol{\rho}$ expressed in terms of rotating coordinates. This equation is substituted into the left side of equation (2.23) and equations (2.19) and (2.20) are substituted for $\boldsymbol{d}$ and $\boldsymbol{r}$, respectively. These substitutions allow the vector equation to be split into its component parts and written as three scalar equations. Recall from equation (2.12) that the nondimensional mean motion equals one; it is left as a variable in the following equations for completeness.

$$\ddot{x} - 2n\dot{y} - n^2x = -\frac{(1-\mu)(x+\mu)}{d^3} - \frac{\mu(x-1+\mu)}{r^3} \tag{2.27}$$

$$\ddot{y} + 2n\dot{x} - n^2y = -\frac{(1-\mu)y}{d^3} - \frac{\mu y}{r^3} \tag{2.28}$$

$$\ddot{z} = -\frac{(1-\mu)z}{d^3} - \frac{\mu z}{r^3} \tag{2.29}$$

Equations (2.27)-(2.29) are expressed in rotating coordinates and the scalar acceleration and velocity terms are evaluated relative to the rotating coordinate frame. The definitions of $\boldsymbol{d}$ and $\boldsymbol{r}$ in equations (2.19) and (2.20) indicate that their magnitudes reflect the relative distance from each primary to P_3. These magnitudes are evaluated as,

$$d = \left[(x+\mu)^2 + y^2 + z^2\right]^{1/2} \tag{2.30}$$

$$r = \left[(x-1+\mu)^2 + y^2 + z^2\right]^{1/2} \tag{2.31}$$

The motion of P_3 under the influence of the primary system is described by the system of first-order differential equations in (2.27-2.29).

The equations of motion in the CR3BP model can also be represented in terms of the gravitational potential function. Equations (2.27)-(2.29) are formulated relative to the rotating frame. These equations allow the introduction of gravitational potential written as a new pseudo-potential, U^*, i.e.,

$$U^* = \frac{(1-\mu)}{d} + \frac{\mu}{r} + \frac{1}{2}n^2(x^2+y^2) \tag{2.32}$$

The newly defined pseudo-potential incorporate terms that accomodate the rotation of the coordinate frame. The equations of motion are then available in a more succinct form,

$$\ddot{x} - 2\dot{y} = \frac{\partial U^*}{\partial x} \tag{2.33}$$

$$\ddot{y} + 2\dot{x} = \frac{\partial U^*}{\partial y} \tag{2.34}$$

$$\ddot{z} = \frac{\partial U^*}{\partial z} \tag{2.35}$$

This formulation also lends insight into the existence of an integral of motion and equilibrium solutions.

2.2.4 Jacobi Constant

In theory equations (2.27)-(2.29), supply all of the necessary information to solve for the motion of P_3. But, the equations are coupled and nonlinear; no general closed-form solution is currently known. Nonetheless, to gain insight into the problem, the existence of U^* suggests the potential for an integral of motion. To derive such a quantity, the dot product of acceleration with velocity is taken.

$$ {}^{R}\boldsymbol{\rho}'' \cdot {}^{R}\boldsymbol{\rho}' = \dot{x}\ddot{x} + \dot{y}\ddot{y} + \dot{z}\ddot{z} = \frac{\partial U^*}{\partial x}\dot{x} + \frac{\partial U^*}{\partial y}\dot{y} + \frac{\partial U^*}{\partial z}\dot{z} \tag{2.36} $$

The pseudo-potential, U^*, is autonomous and only a function of position, therefore, the right side of equation (2.36) equals the total scalar derivative $\frac{dU^*}{d\tau}$. Consequently, equation (2.36) is integrated resulting in,

$$ \frac{1}{2}\left(\dot{x}^2 + \dot{y}^2 + \dot{z}^2\right) = U^* + integration\ const. \tag{2.37} $$

$$ \frac{1}{2}\left(\dot{x}^2 + \dot{y}^2 + \dot{z}^2\right) = U^* - C \tag{2.38} $$

where the integration constant C is defined with a negative sign by convention. Equation (2.38) is more succinctly expressed as,

$$ V^2 = 2U^* - C \tag{2.39} $$

where V is the scalar magnitude of the velocity of P_3, $V = |{}^{R}\boldsymbol{\rho}|$, as viewed by a rotating observer. Equation (2.39) is denoted Jacobi's integral and the integration constant C labeled the Jacobi constant after the mathematician it Carl Gustav Jacob Jacobi. This constant represents an energy-like quantity in the CR3BP. The Jacobi constant has various uses, for example, approximating the energy change necessary for transfers and as a check on the accuracy of numerical integration. The Jacobi constant yields powerful insights into behavior within the CR3BP particularly when combined with particular solutions to the differential equations.

2.2.5 Equilibrium Solutions

Another strategy for gaining insight into the CR3BP is the search for equilibrium solutions. These particular solutions are determined as the states for which the differential equations evaluate to zero. Equilibrium states are located by recognizing that at any equilibrium points the velocity and acceleration of P_3 relative to the rotating frame equals zero. For the set of scalar differential equations this is equivalent to the gradient of the psuedo-potential function equaling the zero vector, i.e., $\boldsymbol{\nabla} U^* = \mathbf{0}$. Thus applied to equations (2.27)-(2.29),

$$\frac{\partial U^*}{\partial x} = -\frac{(1-\mu)(x_{eq}+\mu)}{d_{eq}^3} - \frac{\mu(x_{eq}-1+\mu)}{r_{eq}^3} + n^2 x_{eq} = 0 \tag{2.40}$$

$$\frac{\partial U^*}{\partial y} = -\frac{(1-\mu)y_{eq}}{d_{eq}^3} - \frac{\mu y_{eq}}{r_{eq}^3} + n^2 y_{eq} = 0 \tag{2.41}$$

$$\frac{\partial U^*}{\partial z} = -\frac{(1-\mu)z_{eq}}{d_{eq}^3} - \frac{\mu z_{eq}}{r_{eq}^3} = 0 \tag{2.42}$$

The subscript *eq* on the state variables indicates that these equations are satisfied at the equilibrium points. It is clear that equation (2.42) is only completely satisfied when $z_{eq} = 0$, indicating that all of the equilibrium solutions are planar. Similarly, equation (2.41) is satisfied with $y_{eq} = 0$, therefore, at least one or more of the equilibrium solutions are located on the x-axis of the rotating coordinate frame. To locate these collinear equilibrium solutions, recall that $n = 1$ and substitute $z_{eq} = y_{eq} = 0$ into equation (2.40).

$$0 = -\frac{(1-\mu)(x_{eq}+\mu)}{|x_{eq}+\mu|^3} - \frac{\mu(x_{eq}-1+\mu)}{|x_{eq}-1+\mu|^3} + x_{eq} \tag{2.43}$$

Equation (2.43) yields five solutions for x_{eq} two of which are imaginary and are neglected for the purposes of this evaluation. No closed form solutions exist for equation (2.43), therefore, the remaining three values of x_{eq} are solved iteratively. This numerical process is aided by reformulating equation (2.43) in terms of the displacement, γ_i, from the nearest primary. The three possible values of x_{eq} are defined as follows,

$$x_1 = 1 - \mu - \gamma_1 \tag{2.44}$$

$$x_2 = 1 - \mu + \gamma_2 \tag{2.45}$$

$$x_3 = -\mu - \gamma_3 \tag{2.46}$$

These definitions indicate that one equilibrium point, x_1, is located between the two primaries on the x-axis, while x_3 and x_2 are outside P_1 and P_2, respectively. Such a configuration appears in Figure 2.3, where the equilibrium points are numbered consistent with the typical NASA convention. Equations (2.44)-(2.46) are substituted into equation (2.43) yielding three possible scalar equations,

$$0 = -\frac{(1-\mu)}{(1-\gamma_1)^2} + \frac{\mu}{(\gamma_1)^2} + 1 - \mu - \gamma_1 \tag{2.47}$$

$$0 = -\frac{(1-\mu)}{(1+\gamma_2)^2} - \frac{\mu(\gamma_2)}{(\gamma_2)^2} + 1 - \mu + \gamma_2 \tag{2.48}$$

$$0 = \frac{(1-\mu)}{(\gamma_3)^2} + \frac{\mu}{(\gamma_3+1)^2} + -\mu + \gamma_3 \tag{2.49}$$

Newton's method is used, in combination with a reasonably accurate initial guess, to solve for γ_i in equations (2.47)-(2.49). The resulting values of γ_i are substituted into equations (2.44)-(2.46) to compute the positions of the collinear equilibrium points, L_1, L_2, and L_3.

Two additional equilibrium points are located for $y_{eq} \neq 0$ in equation (2.41). When the equilibrium points are off the x-axis the values of d and r must be equivalent to satisfy equations (2.40) and (2.41) to be satisfied. When $d = r$, two possible values for the location of the equilibrium point exist, i.e.,

$$x_{4,5} = \frac{1}{2} - \mu \tag{2.50}$$

$$y_{4,5} = \pm\frac{\sqrt{3}}{2} \tag{2.51}$$

The points L_4 and L_5 are located on either side of the x-axis equidistant from the primaries and, for this reason, they are denoted the equilateral or triangular points. The locations of these points are also depicted in Figure 2.3, where they are labeled such that, in an inertial frame, L_4 appears to lead P_2 by 60° while L_5 lags by 60°.

2.2.6 Zero Velocity Surfaces

The Jacobi constant and equilibrium solutions lead to another concept that aids qualitative understanding of motion in the CR3BP. When $C > 2U^*$ in equation (2.39), the magnitude of velocity is an imaginary number and, since an imaginary velocity is not physically possible, all natural motion in the CR3BP must satisfy $C \leq 2U^*$.

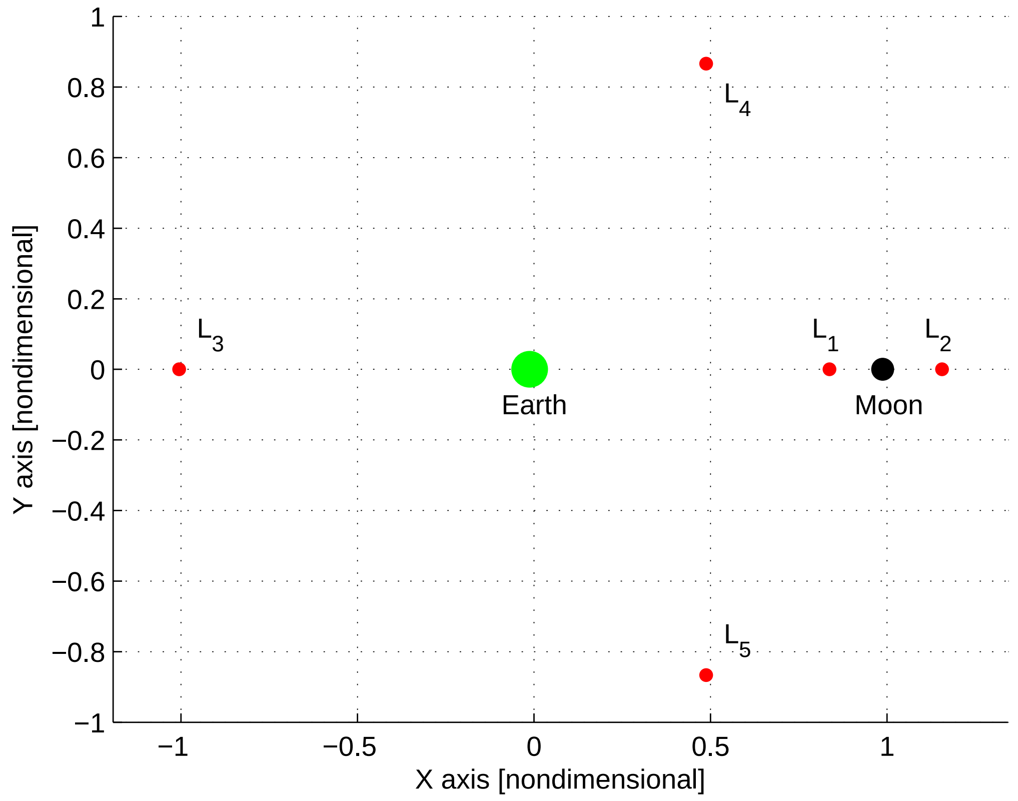


Figure 2.3.: Lagrange Points Configuration

The boundary between the two domains is defined as $C = 2U^*$ which occurs when P_3 possesses zero speed. Expanded, this expression is,

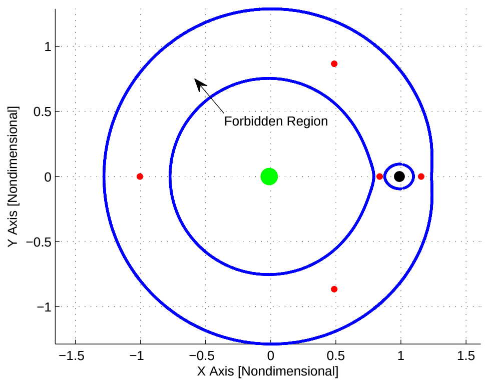
$$C = x^2 + y^2 + \frac{2(1-\mu)}{d} + \frac{2\mu}{r} \tag{2.52}$$

where the values of d and r are evaluated by equations (2.30) and (2.31), respectively. At a given value of C, an infinite variety of locations, or (x, y, z) combinations, satisfy equation (2.52), and this infinite set of points defines a three-dimensional surface termed the zero-velocity surface (ZVS). A cross section of such a zero-velocity surface is a plane reflecting zero-velocity curves (ZVC). In Figure 2.4, ZVC in the $x-y$ plane are depicted for several values of C. Some areas in Figure 2.4 are enclosed by the ZVC and are labeled forbidden regions because, in these zones, $C > 2U^*$ and P_3 cannot enter these regions. As the Jacobi constant value decreases, the energy of P_3

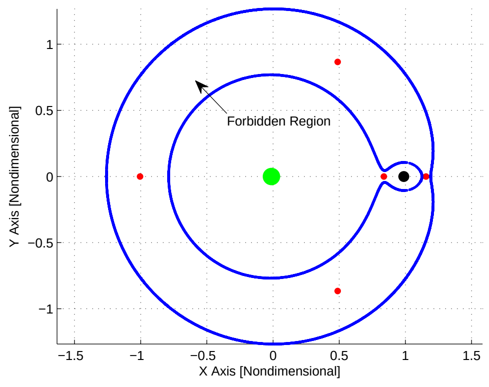
increases, and the ZVS contracts out of the $x - y$ plane. Therefore, the forbidden regions in Figure 2.4 decrease along with C.

Each equilibrium point possesses an associated Jacobi constant value (C_{L_i}). It is convenient to track the evolution of the ZVC as C passes through these values. When $C > C_{L_1}$ the space interior to the forbidden regions is split into two separate zones surrounding each of the primaries, as evident in Figure 2.4(a). When P_3 is located in one of these interior zones it cannot pass from one region into the other; alternately, when P_3 is located exterior to the ZVC it cannot traverse the ZVC and enter the zones near the primaries. When the Jacobi constant is decreased to the range $C_{L_2} < C < C_{L_1}$, as in Figure 2.4(b), the L_1 gateway opens and connects the regions surrounding the two primaries. Within this range of C values, when P_3 is interior to the ZVC, it can pass between the regions immediately surrounding the primaries, but it cannot escape the $P_1 - P_2$ system. As C is further decreased into the range $C_{L_3} < C < C_{L_2}$ the L_1 gateway widens and a new gateway opens at L_2, e.g., Figure 2.4(c). The new gateway links the interior and exterior regions of the ZVC enabling P_3 to escape the $P_1 - P_2$ system entirely. The opportunities for P_3 to escape the system increase as the C value decreases into the range $C_{L_{4,5}} < C < C_{L_3}$ and the forbidden regions recede toward the $L_{4,5}$ equilibrium points, as seen in Figure 2.4(d). Finally, when $C < C_{L_{4,5}}$ the ZVS leave the $x-y$ plane and the ZVC disappear. The ZVS continue to exist as two distinct three dimensional surfaces, however these continue to shrink further from the primary plane of motion as C decreases.

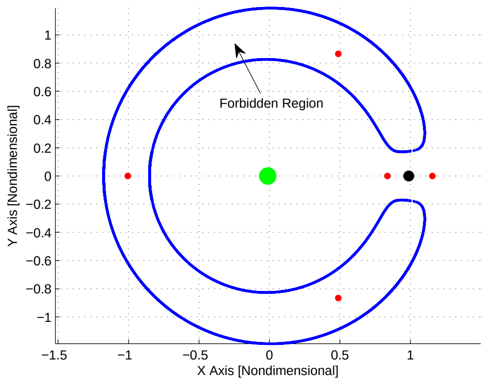
The zero velocity surfaces offer a unique type of guidance for mission design in the CR3BP because they indicate regions of space that allow access of a spacecraft if the spacecraft state attains a particular “energy level”. For example, a spacecraft enroute to the Moon must attain a Jacobi constant value less than C_{L_1} or the L_1 gateway is closed and the spacecraft cannot reach the vicinity of the Moon. A spacecraft can modify its Jacobi constant value via an additional force, e.g., a source of thrust such as an engine or solar sail. Therefore, without sufficient energy after launch, a thrusting maneuver is required to reach the intended destination.



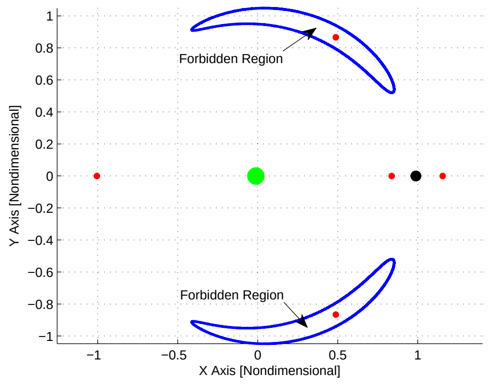
(a) $C > C_{L_1}$



(b) $C_{L_2} < C < C_{L_1}$



(c) $C_{L_3} < C < C_{L_2}$



(d) $C_{L_4} < C < C_{L_3}$

Figure 2.4.: Zero-Velocity Curves in the $x - y$ Plane at Multiple Energy Levels

2.2.7 Linearized Variational Equations of Motion

Some insight into behavior in the vicinity of a particular solution can be explored through the variational equations for motion relative to the reference solution. In the CR3BP, the nonlinear equations of motion are linearized relative to the equilibrium points to examine motion near these points. The resulting system of linear variational equations is subsequently analyzed to assess the stability of the equilibrium points. Recall the complete nonlinear differential equations, i.e., $\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}, t)$. Perturbations relative to an equilibrium solution are introduced into the equations of motion to derive the linear variational equations. The states at equilibrium are defined as $\boldsymbol{x} = \boldsymbol{x}_{eq}$, therefore, the states perturbed from equilibrium are,

$$\boldsymbol{x} = \boldsymbol{x}_{eq} + \delta\boldsymbol{x}_{eq} \tag{2.53}$$

Equation (2.53) is substituted into the nonlinear equations of motion,

$$\dot{\boldsymbol{x}} = \dot{\boldsymbol{x}}_{eq} + \delta\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}_{eq} + \delta\boldsymbol{x}, t) \tag{2.54}$$

Then, the right side of equation (2.54) is expanded about the equilibrium solution using a Taylor series.

$$\begin{aligned} \dot{\boldsymbol{x}}_{eq} + \delta\dot{\boldsymbol{x}} &= \boldsymbol{f}(\boldsymbol{x}_{eq}, t) + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_{eq}} \delta\boldsymbol{x} + \text{H.O.T} \\ \delta\dot{\boldsymbol{x}}_{eq} &\approx \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}_{eq}} \delta\boldsymbol{x} \end{aligned} \tag{2.55}$$

The higher order terms (H.O.T) in the Taylor series are neglected resulting in a first order approximation for the derivative of the variation $\delta\boldsymbol{x}$, termed the variational equations. The partials are of course, evaluated on the reference solution; in this case, the reference solution is the constant equilibrium point. Note that $\boldsymbol{f}$ and $\boldsymbol{x}$ are $n \times 1$ vector quantities, where n is the number of scalar coordinates. For natural motion in the CR3BP, $n = 6$. Thus, the partial derivative $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$ represents a $n \times n$ matrix of partials, that is denoted $\boldsymbol{A}_6$ for conciseness, and

$$\delta\dot{\boldsymbol{x}}_{eq} \approx \boldsymbol{A}_6(t)\delta\boldsymbol{x} \tag{2.56}$$

The matrix $\boldsymbol{A}_6$ consists of partial derivatives of the equations of motion with respect to the state variables with each evaluated at the equilibrium point. The results of the partial derivative evaluations are provided when equation (2.56) is expanded into its matrix representation. The components of the vector $\delta\boldsymbol{x} = \{\xi\ \eta\ \zeta\ \dot{\xi}\ \dot{\eta}\ \dot{\zeta}\}^T$ noting the scalar variations relative to the equilibrium point,

$$\begin{Bmatrix} \dot{\xi} \\ \dot{\eta} \\ \dot{\zeta} \\ \ddot{\xi} \\ \ddot{\eta} \\ \ddot{\zeta} \end{Bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ U^*_{xx} & U^*_{xy} & U^*_{xz} & 0 & 2 & 0 \\ U^*_{xy} & U^*_{yy} & U^*_{yz} & -2 & 0 & 0 \\ U^*_{xz} & U^*_{yz} & U^*_{zz} & 0 & 0 & 0 \end{bmatrix} \begin{Bmatrix} \xi \\ \eta \\ \zeta \\ \dot{\xi} \\ \dot{\eta} \\ \dot{\zeta} \end{Bmatrix} \tag{2.57}$$

In equation (2.57), $U^*_{ij} = \frac{\partial^2 U^*}{\partial i \partial j}$ represents the second partial derivative of the pseudo-potential function, first with respect to the variable i and then with respect to j. The expressions for the second partial derivatives of the psuedo-potential function are,

$$U^*_{xx} = 1 - \frac{(1-\mu)}{d^3} - \frac{\mu}{r^3} + \frac{3(1-\mu)(x+\mu)^2}{d^5} + \frac{3\mu(x-1+\mu)^2}{r^5} \tag{2.58}$$

$$U^*_{yy} = 1 - \frac{(1-\mu)}{d^3} - \frac{\mu}{r^3} + \frac{3(1-\mu)y^2}{d^5} + \frac{3\mu y^2}{r^5} \tag{2.59}$$

$$U^*_{zz} = -\frac{(1-\mu)}{d^3} - \frac{\mu}{r^3} + \frac{3(1-\mu)z^2}{d^5} + \frac{3\mu z^2}{r^5} \tag{2.60}$$

$$U^*_{xy} = \frac{3(1-\mu)(x+\mu)y}{d^5} + \frac{3\mu(x-1+\mu)y}{r^5} = U^*_{yx} \tag{2.61}$$

$$U^*_{xz} = \frac{3(1-\mu)(x+\mu)z}{d^5} + \frac{3\mu(x-1+\mu)z}{r^5} = U^*_{zx} \tag{2.62}$$

$$U^*_{yz} = \frac{3(1-\mu)yz}{d^5} + \frac{3\mu yz}{r^5} = U^*_{zy} \tag{2.63}$$

The matrix $\boldsymbol{A}_6$, evaluated at the equilibrium points, is constant and offers stability information at these points. This information is more easily analyzed when equation (2.57) is solved to obtain a compact form of the variational equations.

$$\ddot{\xi} - 2\dot{\eta} = U^*_{xx}\xi + U^*_{xy}\eta + U^*_{xz}\zeta \tag{2.64}$$

$$\ddot{\eta} + 2\dot{\xi} = U^*_{yx}\xi + U^*_{yy}\eta + U^*_{yz}\zeta \tag{2.65}$$

$$\ddot{\zeta} = U^*_{zx}\xi + U^*_{zy}\eta + U^*_{zz}\zeta \tag{2.66}$$

The linear variational equations approximate motion near the reference solution, and this lends insight into the stability of a particular solution.

2.2.8 Stability of the Equilibrium Solutions

The stability information available via the linear variational equations cannot be properly interpreted unless stability is first defined. The types of motion that occur when a particle is perturbed from an equilibrium point are characterized by the concept of stability. Many definitions of stability are available with the best choice depending upon the objective in a particular problem. It is convenient to consider an equilibrium point and define it as stable if, when a particle at the point is perturbed, its subsequent motion remains bounded within a "small" neighborhood of the equilibrium point. This notion of stability corresponds to the definition of Lyapunov stability. Mathematically, a solution, $\psi(t)$, is Lyapunov stable if, given any $\varepsilon > 0$, there exists a $\delta > 0$ such that any solution $\phi(t)$ satisfying,

$$|\phi(t_0) - \psi(t_0)| < \delta \tag{2.67}$$

also satisfies

$$|\phi(t) - \psi(t)| < \varepsilon, \quad \text{for } t > t_0 \tag{2.68}$$

In short, given a perturbation by an amount δ relative to a reference solution, the subsequent path will diverge from the reference solution by no more than an amount ε for all time. Moreover, a solution is considered asymptotically stable if,

$$|\phi(t) - \psi(t)| \rightarrow 0, \quad \text{at } t \rightarrow \infty \tag{2.69}$$

While this definition of Lyapunov stability is useful when an equilibrium point is the reference solution, it is not as useful when evaluating the stability of an orbit.

The Lyapunov stability of a linear variational system, defined $\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x}$, and assuming that the matrix A is constant, is determined by the roots of the characteristic equation of $\boldsymbol{A}$, i.e. the eigenvalues denoted by λ. Clearly, these roots can be of three types and each lead to different conclusions regarding the stability of the nonlinear, system.

A. Unstable: If any eigenvalues have a positive real component, that is $R(\lambda_i) > 0$, the linear system and any corresponding nonlinear system are unstable.

B. Marginally Stable: If all eigenvalues are purely imaginary, that is $R(\lambda_i) = 0$, the linear system and any corresponding nonlinear system are marginally stable. A marginally stable solution is bounded but not asymptotically stable. No conclusions about the stability of a corresponding nonlinear system can be made.

C. Asymptotically Stable: If all eigenvalues have negative real parts, that is $R(\lambda_i) < 0$, the linear system and any corresponding nonlinear system are asymptotically stable.

The three types of stability conclusions are useful classifications that enable the selection of desirable solutions.

The linear variational equations for motion relative to the equilibrium solutions are employed to assess their stability. Analysis in Section 2.2.5 demonstrated that all five of the equilibrium solutions in the CR3BP are planar, and when $z = 0$ in equations (2.62) and (2.63), then $U^*_{xz} = U^*_{zx} = U^*_{yz} = U^*_{zy} = 0$. Thus, the linear variational equations are further simplified, i.e.,

$$\ddot{\xi} - 2\dot{\eta} = U^*_{xx}\xi + U^*_{xy}\eta \tag{2.70}$$

$$\ddot{\eta} + 2\dot{\xi} = U^*_{yx}\xi + U^*_{yy}\eta \tag{2.71}$$

$$\ddot{\zeta} = U^*_{zz}\zeta \tag{2.72}$$

Equation (2.72) governing out-of-plane, decouples from equations (2.70) and (2.71) which govern in-plane motion. Equation (2.60) indicates that, at all five of the equilibrium points, $U_{zz} < 0$; hence, equation (2.72) represents a simple harmonic oscillator. The corresponding scalar characteristic equation yields two eigenvalues, i.e. $\lambda_{\text{out-of-plane}} = \pm\sqrt{-|U^*_{zz}|}$. These eigenvalues are always purely imaginary, thus, the equilibrium points are marginally stable.

The second-order variational equations governing the in-plane motion, equations (2.70) and (2.71), are examined separately and, ultimately, the characteristic equation possesses four roots. Based on the form of equations (2.70) and (2.71), the solution for the in-plane motion is represented as,

$$\xi = \sum_{i=1}^{4} A_i e^{\lambda_i t} \tag{2.73}$$

$$\eta = \sum_{i=1}^{4} B_i e^{\lambda_i t} \tag{2.74}$$

where A_i and B_i are constants of integration and λ_i are roots of the characteristic equation. The eigenvalues are evaluated from the determinant of the matrix $(\lambda\boldsymbol{I}-\boldsymbol{A}_4)$ as follows,

$$|(\lambda\boldsymbol{I}-\boldsymbol{A}_4)| = \begin{vmatrix} \lambda & 0 & -1 & 0 \\ 0 & \lambda & 0 & -1 \\ -U^*_{xx} & -U^*_{xy} & \lambda & -2 \\ -U^*_{yx} & -U^*_{yy} & 2 & \lambda \end{vmatrix}$$
$$= \lambda^4 + (4 - U^*_{xx} - U^*_{yy})\lambda^2 + (-2U^*_{xy} - 2U^*_{xy})\lambda + (U^*_{xx}U^*_{yy} - U^*_{yx}U^*_{xy}) = 0 \tag{2.75}$$

The fourth-order polynomial in equation (2.75) is denoted the characteristic equation.

First, examine the collinear equilibrium points, where $y_{eq} = z_{eq} = 0$. Equation (2.61) demonstrates that at these points, $U_{xy} = U^*_{yx} = 0$. Thus, equation (2.75) reduces to,

$$\lambda^4 + (4 - U^*_{xx} - U^*_{yy})\lambda^2 + U^*_{xx}U^*_{yy} \tag{2.76}$$

Equation (2.76) is more easily factored when represented as,

$$\Lambda^2 + 2\beta_1\Lambda + \beta_2^2 \tag{2.77}$$

where $\Lambda = \lambda^2$, $\beta_1 = 2 - \frac{U_{xx}^* + U_{yy}^*}{2}$, and $\beta_2^2 = -U_{xx}^* U_{yy}^* > 0$. This simplified representation is then factored into two roots,

$$\Lambda_1 = -\beta_1 + (\beta_1^2 + \beta_2^2)^{1/2} \tag{2.78}$$

$$\Lambda_2 = -\beta_1 - (\beta_1^2 + \beta_2^2)^{1/2} \tag{2.79}$$

Therefore, the four eigenvalues of this system of equations are expressed,

$$\lambda_{1,2} = \pm\sqrt{\Lambda_1} \tag{2.80}$$

$$\lambda_{3,4} = \pm\sqrt{\Lambda_2} \tag{2.81}$$

Evaluated at the equilibrium points, equations (2.58) and (2.59) indicate that $U_{xx} > 0$ and $U_{yy}^* < 0$, respectively. These relationships ensure that the eigenvalues in equation (2.80) are real roots with opposite signs while equation (2.81) yields imaginary roots.

The eigenvalues from this analysis supply information concerning the Lyapunov stability characteristics at each of the collinear equilibrium points. Because the eigenvalues associated with the out-of-plane motion are always purely imaginary, none of the equilibrium points is asymptotically stable. Two of the eigenvalues corresponding to the in-plane motion are also purely imaginary, while the remaining two are real with opposite signs. Because one of the eigenvalues associated with the collinear equilibrium points is positive and real these points are unstable by the definition of Lyapunov stability. However, the imaginary roots suggest some oscillatory behavior. When initial conditions are selected carefully to excite only the oscillatory modes in the linear approximation, motion near the collinear points can appear stable or oscillatory for some time before diverging.

A similar analysis is conducted for the in-plane motion of the triangular equilibrium points. The eigenvalues resulting from this analysis are all purely imaginary indicating that the linear motion near the equilateral points is marginally stable.

Other approaches such as higher order analysis or numerical propagation, are necessary to assess the nonlinear motion near the equilateral points. Szebehely [5] offers extensive linear and some nonlinear analysis to explore the stability of motion near all of the the equilibrium solutions. Nonlinear analysis concerning the stability of the equilateral points has been conducted by Leontovic [23].

The linear variational equations derived in Section 2.2.7 approximate motion in the vicinity of the equilibrium points and supplies some stability information. These equations can be further developed to provide analytical approximations of periodic and quasi-periodic motion relative to the equilibrium points. These approximations serve as convenient initial guesses for numerical strategies that enable nearby motion to be analyzed and targeted.

3. DYNAMICAL SYSTEMS THEORY

Once a dynamical model is developed, dynamical systems theory offers a variety of schemes for analyzing behaviors. The equilibrium solutions associated with the dynamical differential equations identified in section 2.2.5, serve as a guide for obtaining periodic orbits that exist relative to the rotating frame. Continuation methods expand single periodic orbits into branching families that exhibit a wide range of behaviors. The theory of invariant manifolds is applied to equilibrium solutions as well as periodic orbits to compute stable and unstable manifolds that indicate natural flow consistent with the dynamical model. When applied within the context of the CR3BP, these approaches yield new formulations to be employed in conjunction with powerful numerical techniques.

Differential corrections methods techniques an understanding of motion in the CR3BP to produce desired trajectories. The variational equations are assembled into a matrix that linearly maps changes in the initial states along a trajectory to changes in the end state. This sensitivity approximation is implemented in a general free-variable and constraint scheme to iteratively construct trajectories that satisfy desired constraints. Single and multiple shooting techniques can be numerically formulated within this framework, and offer different degrees of computational robustness and accuracy.

3.1 State Transition Matrix

Trajectories within the CR3BP are computed via numerical integration of the equations of motion as derived in Chapter 2. Given that an infinite number of trajectories exist within the CR3BP it is challenging to identify a set of initial conditions to produce an orbit or even a trajectory arc with specific desired characteristics. The state transition matrix (STM) guides this search by providing sensitivity information in the form of a linear map that predicts the impact of changes in the initial states on the final position and velocity states. The STM is employed in combination with numerical methods to deliver a robust and efficient approach for design and analysis.

3.1.1 Linear Variational Equations Approach

A trajectory arc or orbit within the CR3BP is constructed by numerically integrating the dynamic equations of motion $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}_0, t)$ from a set of initial states, $\boldsymbol{x}_0$, for a specified span of time $[t_0 \ t]$. When the states are expressed as Cartesian coordinates within the rotating frame, the initial state vector has the form, $\boldsymbol{x}_0 = \{x_0 \ y_0 \ z_0 \ \dot{x}_0 \ \dot{y}_0 \ \dot{z}_0\}$, and the states at the final time are defined as $\boldsymbol{x}(\boldsymbol{x}_0, t) = \{x(x_0, t) \ y(y_0, t) \ z(z_0, t) \ \dot{x}(\dot{x}_0, t) \ \dot{y}(\dot{y}, t) \ \dot{z}(\dot{z}_0, t)\}$. If additional dynamical force models are incorporated, for example, thrust forces from an engine, then extra states are included in the state vector.

The trajectory design process is initiated with a set of states $\boldsymbol{x}_0^*$ that produce a reference path, $\boldsymbol{x}^*(t)$. This reference path serves as a first guess in producing a desired final path. The initial guess rarely exhibits all of the characteristics desired in the final solution, but, if it is a reasonable approximation, then the states corresponding to the desired solution are likely nearby. Because simply propagating $\boldsymbol{x}_0^*$ in isolation yields no information concerning the behavior of nearby trajectories, variational equations are derived relative to a reference trajectory to obtain insight concerning the behavior of trajectories nearby the reference. First, to derive the appropriate relationships, a variation relative to the reference trajectory is introduced and the initial states corresponding to a nearby trajectory are defined such that,

$$\boldsymbol{x}_0 = \boldsymbol{x}_0^* + \delta\boldsymbol{x}_0 \tag{3.1}$$

The variation $\delta\boldsymbol{x}_0$ is assumed to be small and contemporaneous with respect to the reference, therefore, the reference path and the variation are related as depicted in

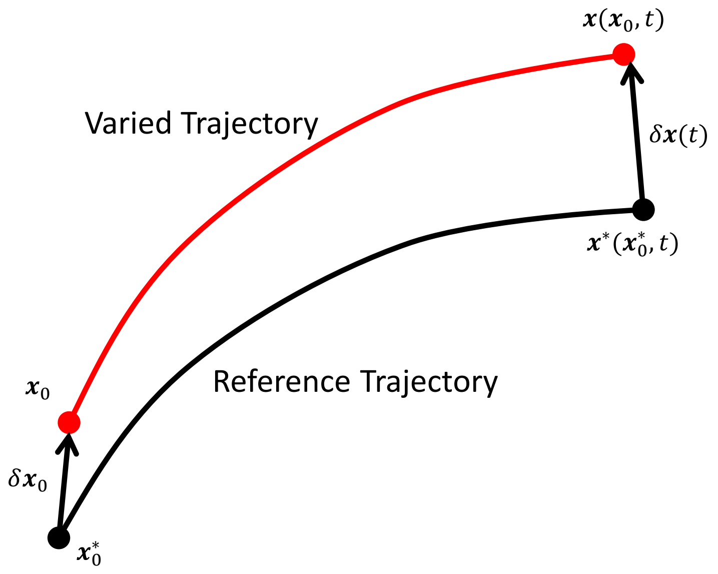


Figure 3.1.: Reference and Variation Trajectory

Figure 3.1. The final state along the trajectory at a later time is then represented in a similar manner,

$$\boldsymbol{x}(\boldsymbol{x}_0^* + \delta\boldsymbol{x}_0, t) = \boldsymbol{x}^*(\boldsymbol{x}_0^*, t) + \delta\boldsymbol{x}(t) \tag{3.2}$$

Because the variation is assumed to be small, a first order Taylor series expansion is used to reflect the left side of equation (3.2) relative to the baseline path,

$$\begin{aligned}
\boldsymbol{x}^*(\boldsymbol{x}_0^*, t) + \frac{\partial\boldsymbol{x}}{\partial\boldsymbol{x}_0}\delta\boldsymbol{x}_0 + \text{H.O.T's} &= \boldsymbol{x}^*(\boldsymbol{x}_0^*, t) + \delta\boldsymbol{x}(t) \\
\boldsymbol{x}^*(\boldsymbol{x}_0^*, t) + \frac{\partial\boldsymbol{x}}{\partial\boldsymbol{x}_0}\delta\boldsymbol{x}_0 &\approx \boldsymbol{x}^*(\boldsymbol{x}_0^*, t) + \delta\boldsymbol{x}(t) \\
\frac{\partial\boldsymbol{x}}{\partial\boldsymbol{x}_0}\delta\boldsymbol{x}_0 &\approx \delta\boldsymbol{x}(t) \qquad (3.3)
\end{aligned}$$

In equation (3.3), $\boldsymbol{x}$ and $\boldsymbol{x}_0$ are $n \times 1$ vector quantities, where n is the number of states, therefore, the partial $\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}_0}$ is a $n \times n$ matrix of partial derivatives denoted as the state transition matrix (STM),

$$\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}_0} = \boldsymbol{\Phi}(t, t_0) = \begin{bmatrix} \frac{\partial x}{\partial x_0} & \frac{\partial x}{\partial y_0} & \frac{\partial x}{\partial z_0} & \frac{\partial x}{\partial \dot{x}_0} & \frac{\partial x}{\partial \dot{y}_0} & \frac{\partial x}{\partial \dot{z}_0} \\ \frac{\partial y}{\partial x_0} & \frac{\partial y}{\partial y_0} & \frac{\partial y}{\partial z_0} & \frac{\partial y}{\partial \dot{x}_0} & \frac{\partial y}{\partial \dot{y}_0} & \frac{\partial y}{\partial \dot{z}_0} \\ \frac{\partial z}{\partial x_0} & \frac{\partial z}{\partial y_0} & \frac{\partial z}{\partial z_0} & \frac{\partial z}{\partial \dot{x}_0} & \frac{\partial z}{\partial \dot{y}_0} & \frac{\partial z}{\partial \dot{z}_0} \\ \frac{\partial \dot{x}}{\partial x_0} & \frac{\partial \dot{x}}{\partial y_0} & \frac{\partial \dot{x}}{\partial z_0} & \frac{\partial \dot{x}}{\partial \dot{x}_0} & \frac{\partial \dot{x}}{\partial \dot{y}_0} & \frac{\partial \dot{x}}{\partial \dot{z}_0} \\ \frac{\partial \dot{y}}{\partial x_0} & \frac{\partial \dot{y}}{\partial y_0} & \frac{\partial \dot{y}}{\partial z_0} & \frac{\partial \dot{y}}{\partial \dot{x}_0} & \frac{\partial \dot{y}}{\partial \dot{y}_0} & \frac{\partial \dot{y}}{\partial \dot{z}_0} \\ \frac{\partial \dot{z}}{\partial x_0} & \frac{\partial \dot{z}}{\partial y_0} & \frac{\partial \dot{z}}{\partial z_0} & \frac{\partial \dot{z}}{\partial \dot{x}_0} & \frac{\partial \dot{z}}{\partial \dot{y}_0} & \frac{\partial \dot{z}}{\partial \dot{z}_0} \end{bmatrix} \tag{3.4}$$

When $t = t_0$ the partials along the diagonal of the STM are all equal to one while the remaining elements equal zero; therefore, $\boldsymbol{\Phi}(t_0, t_0) = \boldsymbol{I}$, where $\boldsymbol{I}$ is the 6×6 identity matrix. A compact representation of the 6×6 STM is,

$$\boldsymbol{\Phi}(t, t_0) = \begin{bmatrix} \boldsymbol{\Phi}_{rr} & \boldsymbol{\Phi}_{rv} \\ \boldsymbol{\Phi}_{vr} & \boldsymbol{\Phi}_{vv} \end{bmatrix} \tag{3.5}$$

where r and v represent position and velocity states, respectively. The STM is essentially a linear map that relates a variation in the initial state $\boldsymbol{x}(\boldsymbol{x}_0, t_0)$ along a path to the resulting variation at the final state $\boldsymbol{x}(\boldsymbol{x}_0, t)$. Thus, the STM is also labelled the sensitivity matrix. Because the STM is derived using a first order Taylor series expansion relative to the reference trajectory, the mapping accuracy is dependent upon the size of the initial variation.

A set of first order differential equations governing the evolution of the STM are derived that are then integrated along with the dynamical equations of motion for the position and velocity states. As a result, variational information with respect to the propagated trajectory is available at each time step, given the augmented state vector, the additional differential equations are,

$$\begin{aligned}\dot{\boldsymbol{\Phi}}(t,t_0) &= \frac{d}{dt}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}_0}\\ \frac{\partial}{\partial \boldsymbol{x}_0}\frac{\boldsymbol{x}}{dt} &= \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}_0}\\ &= \frac{\boldsymbol{\partial f}}{\partial \boldsymbol{x}}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{x}_0}\end{aligned} \tag{3.6}$$

Recall from the derivation of the linear variational equations, equation (2.56), that the matrix of partial derivatives $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}$ is denoted $\boldsymbol{A}_6$. Thus, equation (3.6) is rewritten,

$$\dot{\boldsymbol{\Phi}}(t,t_0) = \boldsymbol{A}_6\boldsymbol{\Phi}(t,t_0) \tag{3.7}$$

When evaluated at the equilibrium points, $\boldsymbol{A}_6$ is a constant matrix but, in general, the matrix changes with time as the states evolve along a trajectory arc. Equation (3.7) results in an $n \times n$ matrix, therefore, for natural motion in the CR3BP, propagation of the STM along with the EOM requires the numerical integration of 42 total differential equations. While the partial derivatives required to compute the STM using equation (3.7) are relatively straightforward to derive for natural motion, the difficulty increases as the dynamics grow more complex, leading to the development of alternate strategies to construct the STM.

3.1.2 Numerical Approximation for Partial Derivatives

Analytically determining the partial derivatives necessary to construct the STM can be challenging, moreover, modification of the dynamical force models necessitate an update to the derivation of the partials. Therefore, it is sometimes advantageous to numerically approximate partial derivatives for the STM because such approximations are typically straightforward and easily implemented. However, such an approach warrants caution because the STM is approximated with varying degrees of accuracy. Additionally, when explicit numerical integration of the EOMs is included in the process the computation time required to produce the approximation can be large.

Moreover, insight into relationships between the state vector at different times can be lost when the user relies solely on numerical approximations. Nonetheless, when used appropriately, numerical approximation methods for partial derivatives are a powerful approach.

The first-order central difference approximation is a finite difference method commonly employed to numerically approximate partial derivatives. This method is derived by differencing two first order Taylor-series expansions, resulting in

$$\frac{\partial f_j}{\partial x_i} = \frac{f_j(x_i + h) - f_j(x_i - h)}{2h} + O(h^2) \tag{3.8}$$

where h is the step size and $O(h^2)$ indicates that this method has a truncation error on the order of h^2. A small step size, h, must be selected for a low truncation error. However, if h is too small, excessive round off error occurs due to the subtraction step in the numerator. Strategies for selecting a value of h that mitigate these two sources of error have been developed [24].

Complex step differentiation affords an even more powerful method for numerical approximation of partial derivatives. This approach is also derived from a Taylor-series expansion but, in this case, a step is taken along the imaginary axis [25].

$$f(x + ih) = f(x) + ihf'(x) - h^2 f''(x)/2! - ih^3 f^{(3)}/3! + \dots \tag{3.9}$$

Focus on the imaginary part of both sides of equation (3.9) and solve for $f'(x)$.

$$\begin{aligned} Im\left(f(x + ih)\right) &= hf'(x) - h^3 f^{(3)}/3! + \dots \\ f'(x) &= Im\left(f(x + ih)\right)/h + O(h^2) \end{aligned} \tag{3.10}$$

The division required in equation (3.10) does not induce round-off error, therefore, h can be set arbitrarily small. The step size, h is typically selected such that the truncation error $O(h^2)$ is below the numerical precision of the computational tool in use, i.e. Matlab, rendering the resulting approximation equally accurate to a numerically implemented analytical method.

A third method approach to numerically computing partial derivatives is automatic (algorithmic) differentiation (AD). While not implemented in this investigation, such a technique is increasingly applied to trajectory design and optimization problems [26]. The term AD describes a variety of techniques for computing derivatives by implementing basic differentiation rules within the source code of a numerical method. Its primary advantages include the avoidance of truncation errors and automatic computation of the derivatives of a function in parallel with the computation of the function itself [27]. While this procedure is more complex to implement than finite differencing approaches, it is less computationally expensive. Automatic differentiation is facilitated by object-oriented programming and has been implemented in many languages with this capability, including Matlab [28].

3.2 Differential Corrections

Construction of a trajectory within the context of a specific dynamical model that satisfies a specific set of constraints is typically formulated as a two-point boundary value problem (TPBVP). Differential corrections strategies are frequently employed to solve TPBVPs; they employ sensitivity information to render a solution in an iterative process. These strategies can be implemented in a multitude of ways, but a general scheme adaptable to a variety of problems is incorporated in this investigation.

3.3 Constraint and Free-Variable Formulation

The constraint and free-variable formulation provides a general framework for solving TPBVP's using differential corrections. The framework is popular because of its simplicity and proven success in accurately solving complex problems. However, when a general methodology is applied to a particular problem, some insight can be lost, care to ensure that problem specific details are not obscured is necessary. The constraint and free-variable formulation is employed in conjunction with the simple, yet powerful, Newton's method in order to iteratively solve a TPBVP. Consider a design variable vector $\boldsymbol{X}$ of n free-variables,

$$\boldsymbol{X} = \begin{Bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{Bmatrix} \tag{3.11}$$

which is subject to m constraint equations of the form

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{Bmatrix} F_1(\boldsymbol{X}) \\ F_2(\boldsymbol{X}) \\ \vdots \\ F_m(\boldsymbol{X}) \end{Bmatrix} = \boldsymbol{0} \tag{3.12}$$

In general, the objective is the determination of the design variable vector, $\boldsymbol{X}_c$, that satisfies $\boldsymbol{F}(\boldsymbol{X}_c) = \boldsymbol{0}$. A simple example of application is the determination of the initial position and velocity to obtain a periodic orbit. An iterative process to determine $\boldsymbol{X}_c$ is derived by applying a Taylor-series expansion to $\boldsymbol{F}(\boldsymbol{X}_0)$ where $\boldsymbol{X}_0$ is the initial guess for the design variable vector, i.e.,

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{X}) &= \boldsymbol{F}(\boldsymbol{X}_0) + D\boldsymbol{F}(\boldsymbol{X}_0) \cdot (\boldsymbol{X} - \boldsymbol{X}_0) + H.O.T.S \\ 0 &\approx \boldsymbol{F}(\boldsymbol{X}_0) + D\boldsymbol{F}(\boldsymbol{X}_0) \cdot (\boldsymbol{X} - \boldsymbol{X}_0) \end{aligned} \tag{3.13}$$

Then, $\boldsymbol{X}_0$ effectively serves as the reference for the variational relationships. Neglecting higher order terms and substituting the definition of $\boldsymbol{F}(X)$ from equation (3.12) reduces the expansion to equation (3.13). Application of the Taylor-series expansion does imply that the initial guess $\boldsymbol{X}_0$ is relatively close to the final design variable vector $\boldsymbol{X}_c$. The matrix $D\boldsymbol{F}(\boldsymbol{X}_0)$ is an $m \times n$ matrix typically denoted the Jacobian with elements that are partial derivatives of each constraint vector element with respect to each design variable vector element,

$$D\boldsymbol{F}(\boldsymbol{X}_0) = \left.\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{X}}\right|_{\boldsymbol{X}_0} = \begin{bmatrix} \frac{\partial \boldsymbol{F}_1}{\partial \boldsymbol{X}_1} & \frac{\partial \boldsymbol{F}_1}{\partial \boldsymbol{X}_2} & \cdots & \frac{\partial \boldsymbol{F}_1}{\partial \boldsymbol{X}_n} \\ \frac{\partial \boldsymbol{F}_2}{\partial \boldsymbol{X}_1} & \frac{\partial \boldsymbol{F}_2}{\partial \boldsymbol{X}_2} & \cdots & \frac{\partial \boldsymbol{F}_2}{\partial \boldsymbol{X}_n} \\ \vdots & \vdots & \ddots & \\ \frac{\partial \boldsymbol{F}_m}{\partial \boldsymbol{X}_1} & \frac{\partial \boldsymbol{F}_m}{\partial \boldsymbol{X}_2} & \cdots & \frac{\partial \boldsymbol{F}_m}{\partial \boldsymbol{X}_n} \end{bmatrix} \tag{3.14}$$

Note that each particle is evaluated on the reference path. Most useful TPVBP are nonlinear and iteration based upon equation (3.13) is generally required to produce a solution, therefore, it is convenient to express the relationship in a general form,

$$\boldsymbol{F}(\boldsymbol{X}_j) + D\boldsymbol{F}(\boldsymbol{X}_j) \cdot (\boldsymbol{X}_{j+1} - \boldsymbol{X}_j) = \boldsymbol{0} \tag{3.15}$$

Solving for $\boldsymbol{X}_{j+1}$ yields the final form of the update equation, but the actual method of solution depends upon the dimensions of $D\boldsymbol{F}(\boldsymbol{X}_j)$. If the number of design variables equals the number of constraints, that is $n = m$, then one unique solution to equation (3.15) exists. In such a case, $D\boldsymbol{F}(\boldsymbol{X}_j)$ can simply be inverted to solve equation (3.15).

$$\boldsymbol{X}_{j+1} = \boldsymbol{X}_j - D\boldsymbol{F}(\boldsymbol{X}_j)^{-1}\boldsymbol{F}(\boldsymbol{X}_j) \tag{3.16}$$

The solution in equation (3.16) is a multi-dimensional form of the well-known Newton's method. In addition to its adaptability to numerous problems, this approach is powerful because it achieves quadratic convergence properties.

When there are more design variables than constraints, that is $n > m$, infinitely many solutions exist to equation (3.15). Therefore, a unique solution emerges by isolating the update $\boldsymbol{X}_{j+1}$ that is closest to $\boldsymbol{X}_j$. This choice acknowledges the fact that gradient searches are more successful when the updated solution is close to the reference, i.e., when $\delta\boldsymbol{X}_j$ is small. Additionally, seeking a solution nearby $\boldsymbol{X}_j$ implies that $\boldsymbol{X}_{j+1}$ likely inherits some of the characteristics of $\boldsymbol{X}_j$ which is presumably beneficial, particularly if the initial guess is selected to possess characteristics that are desired in the final solution. The unique solution $\boldsymbol{X}_{j+1}$ closest to $\boldsymbol{X}_j$ is labelled the minimum-norm and is determined via the equation,

$$\boldsymbol{X}_{j+1} = \boldsymbol{X}_j - D\boldsymbol{F}(\boldsymbol{X}_j)^T \left[D\boldsymbol{F}(\boldsymbol{X}_j) \cdot D\boldsymbol{F}(\boldsymbol{X}_j)^T\right]^{-1} \boldsymbol{F}(\boldsymbol{X}_j) \tag{3.17}$$

Whether equation (3.16) or equation (3.17) is employed as the update equation, both are applied iteratively given an initial guess $\boldsymbol{X}_0$. In general, at each iteration, $\|F(\boldsymbol{X}_{j+1})\| < \|F(\boldsymbol{X}_j)\|$, and iterations are continued until $F(\boldsymbol{X}_{j+1}) = F(\boldsymbol{X}_c) = 0$. In practice, $F(\boldsymbol{X}_{j+1})$ will never equal exactly zero, of course so a tolerance, ε, is set and the update equation is iterated until $\|F(\boldsymbol{X}_{j+1})\| < \varepsilon$. In summary, the general structure of the constraint and free variable formulation allows four basic steps:

1. Select free-variables and construct the $n \times 1$ design variable vector, $\boldsymbol{X}$, with an initial value $\boldsymbol{X}_0$.

2. Define constraints as equality constraints and formulate the $m \times 1$ constraint vector, $F(\boldsymbol{X}) = 0$.

3. Compute the partial derivatives of the constraint vector with respect to each of the design variables and assemble the results into the $m \times n$ Jacobian matrix, $D\boldsymbol{F}(\boldsymbol{X})$.

4. Depending upon the relationship between m and n, use either equation (3.16) or (3.17) to compute $\boldsymbol{X}_{j+1}$. The new $\boldsymbol{X}_{j+1}$ vector then is defined as the reference $\boldsymbol{X}_j$ and the process repeats. Iteratively, re-compute $\boldsymbol{X}_{j+1}$ until $\|F(\boldsymbol{X}_{j+1})\| < \varepsilon$, or the number of iterations exceeds a pre-defined limit.

The free-variable and constraint formulation is a convenient framework for numerous astrodynamics problems and is incorporated in this investigation for shooting and collocation methods.

3.4 Single Shooting

The general scheme for the implementation of differential corrections as formulated in Section 3.3 is applied in a variety of applications including shooting schemes. Shooting algorithms require repeated propagations to recover one or more trajectory arcs. After each propagation, the differential corrections scheme adjusts the initial conditions to satisfy a given set of constraints. This process is repeated until the constraints are satisfied to within a specified tolerance. A sample single shooting problem is initiated with a set of initial conditions $\{\boldsymbol{r}_0 \ \boldsymbol{v}_0\}^T = \{x_0 \ y_0 \ z_0 \ \dot{x}_0 \ \dot{y}_0 \ \dot{z}_0\}^T$, at time t_0. The objective is to determine the change in the initial velocity, $\boldsymbol{v}_0$, that is necessary to deliver the final point along the path to the specified location, i.e., $\boldsymbol{r}_d = \{x_d \ y_d \ z_d\}$, at the fixed time t, as depicted in Figure 3.2. The variables, those

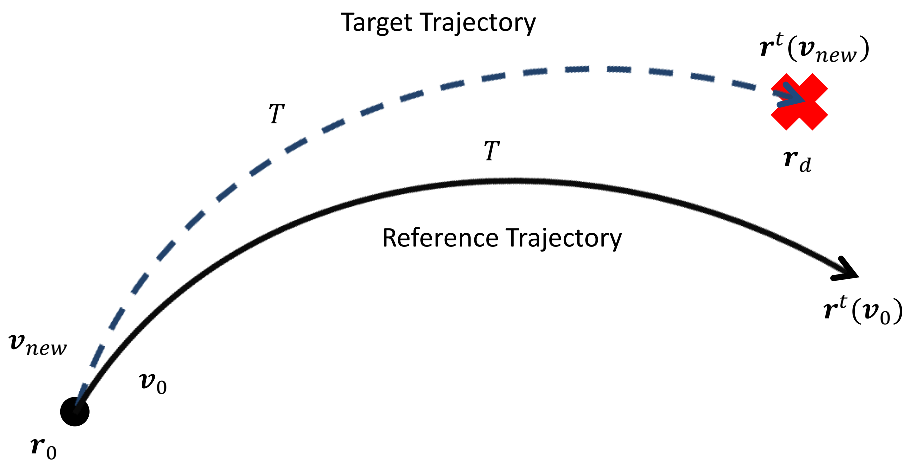


Figure 3.2.: Single Shooting Problem

allowed to be altered, termed the design variables, are the components of the initial velocity, as collected,

$$\boldsymbol{X} = \boldsymbol{v}_0 = \begin{Bmatrix} \dot{x}_0 \\ \dot{y}_0 \\ \dot{z}_0 \end{Bmatrix} \tag{3.18}$$

The objective is satisfied when the position at the end of propagation is equivalent to the desired position. Therefore, the equality constraints are written such that they equal zero when the final position states are equal to the target position, i.e.,

$$\boldsymbol{F}(\boldsymbol{X}) = \boldsymbol{r}_d - \boldsymbol{r}(\boldsymbol{v}_0) = \begin{Bmatrix} x_d - x(\boldsymbol{v}_0) \\ y_d - y(\boldsymbol{v}_0) \\ z_d - z(\boldsymbol{v}_0) \end{Bmatrix} \tag{3.19}$$

The Jacobian consists of the partial derivatives of the constraints with respect to the design variables. The chain rule is applied because the constraints are implicitly dependent upon the design variables,

$$D\boldsymbol{F}(\boldsymbol{X}) = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{r}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{v}_0} = \begin{bmatrix} \frac{\partial \boldsymbol{F}_1}{\partial x} & \frac{\partial \boldsymbol{F}_1}{\partial y} & \frac{\partial \boldsymbol{F}_1}{\partial z} \\ \frac{\partial \boldsymbol{F}_2}{\partial x} & \frac{\partial \boldsymbol{F}_2}{\partial y} & \frac{\partial \boldsymbol{F}_2}{\partial z} \\ \frac{\partial \boldsymbol{F}_m}{\partial x} & \frac{\partial \boldsymbol{F}_m}{\partial y} & \frac{\partial \boldsymbol{F}_m}{\partial z} \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial x}{\partial \dot{x}_0} & \frac{\partial x}{\partial \dot{y}_0} & \frac{\partial x}{\partial \dot{z}_0} \\ \frac{\partial y}{\partial \dot{x}_0} & \frac{\partial y}{\partial \dot{y}_0} & \frac{\partial y}{\partial \dot{z}_0} \\ \frac{\partial z}{\partial \dot{x}_0} & \frac{\partial z}{\partial \dot{y}_0} & \frac{\partial z}{\partial \dot{z}_0} \end{bmatrix} \tag{3.20}$$

The position of the target position is constant, therefore, all partials of the components of $\boldsymbol{r}_d$ with respect to the states equal zero; the left matrix in equation (3.20) simplifies to, $\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{r}} = -\boldsymbol{I}$. The right matrix in equations (3.20) is equal to the upper right quadrant of the STM in equation (3.5), thus, the Jacobian simplifies to

$$D\boldsymbol{F}(\boldsymbol{X}) = -\boldsymbol{I} \cdot \boldsymbol{\Phi}_{rv}(t_0, t) = -\boldsymbol{\Phi}_{rv}(t_0, t) \tag{3.21}$$

Given the case $n = m$, the Jacobian is square and equation (3.16) is used as the update equation. Substituting in the results from equations (3.18), (3.19), and (3.21), the unique update equation is

$$\begin{Bmatrix} \dot{x}_0 \\ \dot{y}_0 \\ \dot{z}_0 \end{Bmatrix}_{j+1} = \begin{Bmatrix} \dot{x}_0 \\ \dot{y}_0 \\ \dot{z}_0 \end{Bmatrix}_j - \left(-\boldsymbol{\Phi}_{rv}(t_0, t)\right)^{-1} \begin{Bmatrix} x_d - x(\boldsymbol{v}_0) \\ y_d - y(\boldsymbol{v}_0) \\ z_d - z(\boldsymbol{v}_0) \end{Bmatrix}_j \tag{3.22}$$

The update equation is applied iteratively until the norm of the difference between the target point location and the final position along the propagated path is less than ε, or $\left\| F(\boldsymbol{X}_{j+1}) \right\| < \varepsilon$. This example is easily modified to accomodate a variable time single shooting problem by adding the propagation time t to the design variable vector in equation (3.18). This addition results in $n = 4$, and adding an extra column to the matrix $D\boldsymbol{F}(\boldsymbol{X})$ that contains the derivatives of the position states with respect to time. The Jacobian then becomes a 3×4 matrix, requiring equation (3.17) as the update equation.

3.5 Multiple Shooting

Multiple shooting is, fundamentally, a single shooting strategy applied to subarcs along a path. Figure 3.3 depicts a trajectory subdivided into multiple arcs, ready for correction via a multiple shooting scheme. Subdividing a trajectory into multiple subarcs offers several advantages including increased accuracy and robustness. The errors associated with numerical integration, including approximation and truncation, increase with time, therefore, decomposing a trajectory into multiple arcs with shorter propagation times can improve the accuracy of the final converged result. Subdivision may also improve the convergence properties of a shooting method. The states along a trajectory arc also change rapidly as it passes near a primary, therefore, trajectories are particularly sensitive in these areas. Decomposing a path into multiple segments near a primary reduces the sensitivity of each sub-arc thereby reducing the number of iterations required to achieve convergence. Finally, subdividing a path allows path constraints to be applied at specific nodes rather than along an entire trajectory. For example, an altitude constraint can be applied at nodes that occur near a primary, but removed from nodes elsewhere.

The design variable vector in a fixed time multiple shooting scheme includes the initial state vectors for each subarc along a trajectory. In variable time multiple

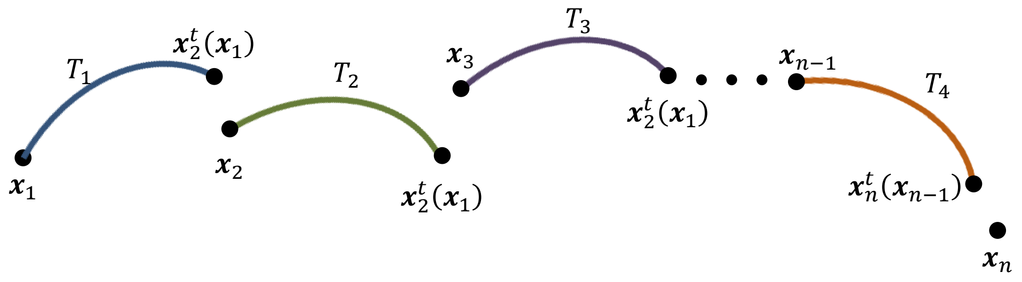


Figure 3.3.: Multiple Shooting Problem

shooting, the integration time along each subarc is also added to the design variable vector,

$$\boldsymbol{X} = \begin{Bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_{n-1} \\ \boldsymbol{x}_n \end{Bmatrix} \tag{3.23}$$

In addition to satisfying specific boundary and path constraints, the sub-arcs are also constrained to be continuous. Therefore, the constraint vector always contains continuity constraints that link the individual segments. The continuity constraints must also enforce position continuity for a single feasible trajectory, however, continuity in velocity can generally be omitted. Any discontinuities in velocity that occur as a result of this omission are incorporated as Δv's along the path,

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{Bmatrix} \boldsymbol{x}_2^t(\boldsymbol{x}_1) - \boldsymbol{x}_2 \\ \boldsymbol{x}_3^t(\boldsymbol{x}_2) - \boldsymbol{x}_3 \\ \vdots \\ \boldsymbol{x}_2^t(\boldsymbol{x}_{n-1}) - \boldsymbol{x}_n \end{Bmatrix} \tag{3.24}$$

The result of a multiple shooting scheme is, a trajectory defined by discrete sub-arcs that are explicitly propagated from a set of initial state values.

Numerical methods offer powerful techniques capable of producing trajectories that can satisfy specific constraints. However, for the most part, these numerical strategies can only supply information on a specific orbit for a finite time interval. Initially, the equilibrium solutions in the CR3BP supply insight into fundamental behaviors in the CR3BP model. It is generally advantageous to seek additional particular solutions and assess stability to illuminate the CR3BP behavior.

3.6 Generating Periodic Orbits

Because the Hamiltonian in the CR3BP model is not an explicit function of time, periodic solutions to the differential equations of motion exist. The differential corrections techniques identified in Section 3.2 allow periodic orbits, Poincaré's [3] "windows into the R3BP", to be constructed and investigated. Periodic orbits, of course, supply information on behavior over infinite time but require only finite integration time. An infinite amount of periodic orbits exist in the CR3BP, however, these solutions can be organized into distinct categories based on common characteristics. One of the simplest type of periodic orbits are the planar Lyapunov orbits that exist in the vicinity of each collinear equilibrium point.

3.6.1 Planar Lyapunov Orbits

The computation of a periodic orbit is easily posed as a TPBVP where the 6-D boundary points are equivalent, therefore, symmetry is exploited to simplify the problem. The EOM in the CR3BP model remain unchanged when time runs backwards; this fact permits application of the *Mirror Theorem* for the computation of periodic orbits. The theorem states that, if n-point masses are moving under mutual gravitational forces, then their orbits are periodic, if at two separate epochs, a mirror configuration occurs [29]. In the CR3BP, several types of periodic orbits exhibit mirror configurations across planes defined by combinations of the coordinate axes such that, for every trajectory, there is a mirror image path across the plane in backwards time. A consequence of this property is a symmetric periodic orbit that is determined by only integrating for precisely one-half of an orbital period, thus, reducing the computational time and improving convergence. A Lyapunov orbit is planar, existing in the x-y plane, and symmetric across the x-axis. Thus, to construct a Lyapunov orbit, initial conditions are selected to yield a perpendicular crossing of the x-axis. The target state reflects a similar perpendicular crossing of the x-axis that occurs after precisely one-half period of integration. Thus, the initial position and velocity states to represent a perpendicular crossing at the x-axis possess the form $\boldsymbol{x}_0 = \{x_0 \ 0 \ 0 \ 0 \ \dot{y}_0 \ 0\}$. To construct the orbit, the initial position is assumed to be fixed, but the velocity $\dot{y}_0$ and the propagation time t are design variables and allowed to vary, thus,

$$\boldsymbol{X} = \begin{Bmatrix} \dot{y}_0 \\ t \end{Bmatrix} \tag{3.25}$$

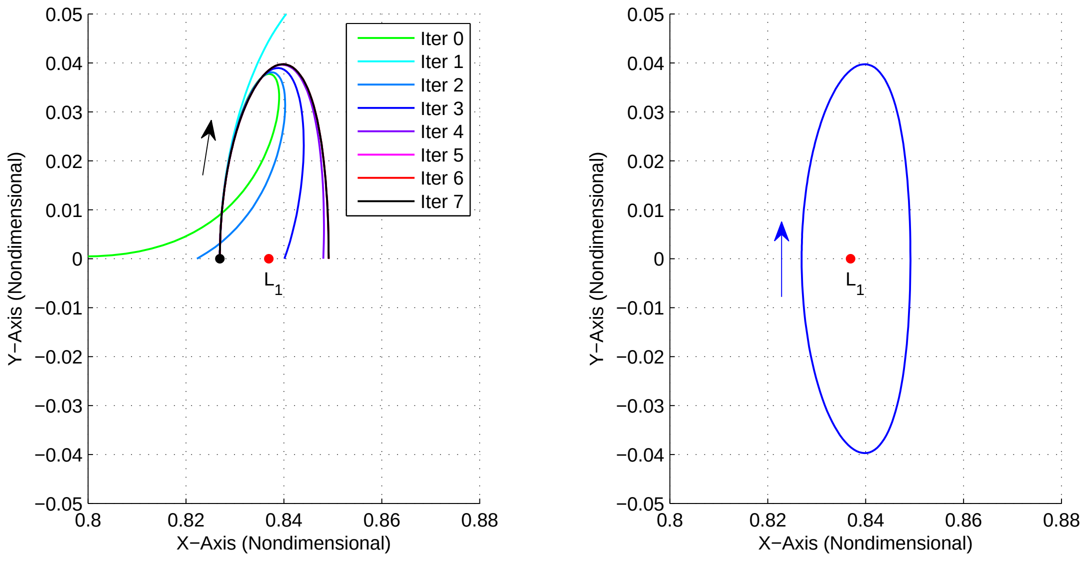
At a later time, a perpendicular crossing on the x axis occurs when $y(t) = z(t) = \dot{x}(t) = \dot{z}(t) = 0$. However, because this problem is strictly planar, no out-of-plane constraints are necessary to include in the constraint vector,

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{Bmatrix} y(t) \\ \dot{x}(t) \end{Bmatrix} \tag{3.26}$$

Additionally, excluding out-of-plane components from the constraint vector ensures that the resulting Jacobian matrix is square, thus guaranteeing a unique solution in the targeting problem. The $D\boldsymbol{F}(\boldsymbol{X})$ matrix is then comprised of partial derivatives that are easily computed using the STM and EOM, i.e.,

$$D\boldsymbol{F}(\boldsymbol{X}) = \begin{bmatrix} \frac{\partial y(t)}{\dot{y}_0} & \frac{\partial y(t)}{\partial t} \\ \frac{\partial \dot{x}(t)}{\dot{y}_0} & \frac{\partial \dot{x}(t)}{\partial t} \end{bmatrix} \tag{3.27}$$

Once the problem is formulated consistent with the free-variable and constraint equation (3.16), iterations continue until the desired tolerance is met. Figure 3.4(a) demonstrates various iterations of the targeting scheme; note that each subsequent iteration



(a) Targeter Iterations (b) Corrected L_1 Lyapunov Orbit

Figure 3.4.: Targeting L_1 Lyapunov Orbit

shifts closer to satisfying the required condition for a perpendicular crossing. Thus, the resulting trajectory is one-half of the Lyapunov orbit originating at x_0, and the final time t is precisely one-half of the orbital period. The final converged initial state is subsequently propagated for $2t$ to produce the orbit in Figure 3.4(b) which is a Lyapunov orbit about L_1. The same process is also used to compute Lyapunov orbits about L_2 and L_3. The family of Lyapunov periodic orbits remain entirely in the $x-y$ plane, and evolve to large orbits that pass very near the smaller primary.

3.6.2 Three-Dimensional Halo Orbits

Halo orbits represent set of characteristics that define a family of periodic orbits that exist in the CR3BP. These orbits are three-dimensional but, because they are symmetric across the x-z plane, they can be targeted using a process very similar to one exploited for Lyapunov orbits. In this case, the initial conditions reflect a perpendicular crossing at the x-z plane, $\boldsymbol{x}_0 = \{x_0\ 0\ z_0\ 0\ \dot{y}_0\ 0\}^T$. Consistent with the Lyapunov orbits, an initial position state is assumed constant; either x_0 or z_0 is selected depending on which direction is less sensitive at the given location along the family. In this example, x_0 is assumed fixed. Then, the design variable vector is expressed,

$$\boldsymbol{X} = \begin{Bmatrix} z_0 \\ \dot{y}_0 \\ t \end{Bmatrix} \tag{3.28}$$

Because the halo orbit is three dimensional the out-of-plane velocity component must also be constrained to ensure a perpendicular crossing,

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{Bmatrix} y(t) \\ \dot{x}(t) \\ \dot{z}(t) \end{Bmatrix} \tag{3.29}$$

Due to the omission of x_0 from the design vector, $n = m$ producing a square Jacobian matrix and a unique solution to the problem exists,

$$D\boldsymbol{F}(\boldsymbol{X}) = \begin{bmatrix} \frac{\partial y(t)}{\dot{z}_0} & \frac{\partial y(t)}{\dot{y}_0} & \frac{\partial y(t)}{\partial t} \\ \frac{\partial \dot{x}(t)}{\dot{z}_0} & \frac{\partial \dot{x}(t)}{\dot{y}_0} & \frac{\partial \dot{x}(t)}{\partial t} \\ \frac{\partial \dot{z}(t)}{\dot{z}_0} & \frac{\partial \dot{z}(t)}{\dot{y}_0} & \frac{\partial \dot{z}(t)}{\partial t} \end{bmatrix} \tag{3.30}$$

Equation (3.16) serves to update the design variable vector until the constraint vector meets the desired tolerance. Once again, the converged result is precisely half of the periodic halo orbit; the successful initial conditions are propagated for $2t$ to construct the full orbit. Halo orbits also exist in the vicinity of the other two collinear points. In fact, the symmetric properties of the CR3BP are also exploited to construct orbits in the axial and vertical families. Periodic orbits also exist about the equilateral points but these tyoes of trajectories are not symmetric, therefore, targeting algorithms to numerically determine such orbits must be modified to satisfy different types of constraints [30].

3.6.3 Stability Analysis for Periodic Orbits

Periodic orbits are a particular solution to the CR3BP and the concept of stability is important in understanding behavior in their vicinity. In Section 2.2.7, a system of linear variational equations with constant matrix $\boldsymbol{A}_6$ is obtained by linearizing relative to the equilibrium points. The same linearization process about a periodic orbit yields a set of linear variational equations, $\delta\dot{\boldsymbol{x}} = \boldsymbol{A}_6(t)\delta\boldsymbol{x}$; however the matrix $\boldsymbol{A}_6(t)$ now varies with time since the reference path is a function of time. Application of Floquet theory, along with the concept of maps, allows the continuous time linear variational equations along the periodic orbit to be discretized enabling a type of stability analysis similar to that employed for equilibrium points. The STM after precisely one revolution of the periodic orbit is denoted the monodromy matrix. This matrix essentially delivers a linear map that facilitates discretization of the system differential equations. The new linear variational equations for the discrete time system emerge as, $\delta\boldsymbol{x}(t) = \boldsymbol{\phi}(t, t_0)\delta\boldsymbol{x}_0$ and, using the monodromy matrix, they evolve as, $\delta\boldsymbol{x}(T) = \boldsymbol{\phi}(T, t_0)\delta\boldsymbol{x}_0$. Thus, the monodromy matrix is used to calculate the values of the linear variational equations at any time along a periodic orbit.

The eigenvalues of the monodromy matrix supply stability information for a fixed point along a periodic orbit and, thus, the entire orbit. However, because the system is a discrete time linear system, the eigenvalues correlation with stability differ from the continuous case that is,

1. If all $|\lambda_i| < 1$ then the system is <u>asymptotically stable</u>.

2. If all $|\lambda_i| \leq 1$ then the system is <u>marginally stable</u>.

3. If all $|\lambda_i| > 1$ then the system is <u>unstable</u>.

Additionally, systems where some $|\lambda_i| < 1$ and some $|\lambda_i| > 1$ are considered non-stable, and a non-stable fixed-point is called a saddle point. Thus, the stability of a periodic orbit is indicated by the combination of eigenvalues yielded by the monodromy matrix.

For the set of differential equations and the periodic orbits constructed, thus far, the monodromy matrix is characterized by predictable types of eigenvalues. First, the Lyapunov theorem states that an eigenvalue λ of the monodromy matrix, $\boldsymbol{\phi}(T, t_0)$, of a time-invariant system also possesses a reciprocal counterpart λ^{-1} with the same structure of elementary divisors. Furthermore, because $\boldsymbol{\phi}(T, t_0)$ is a real matrix, its eigenvalues are either real or in complex conjugate pairs. The first consequence of these observations is a set of eigenvalues of $\boldsymbol{\phi}(T, t_0)$ that only exist in certain configurations. Complex conjugate pairs are generally not reciprocal, therefore, these eigenvalues, occur as complex pairs on the unit circle. It is also deduced that, because the λ occur in reciprocal pairs, any $|\lambda| < 1$ that induces stability is accompanied by a counterpart $|\lambda| > 1$ supporting instability. Therefore, a periodic orbit is only ever marginally stable, and this stability occurs when all of its eigenvalues equal one. Finally, given the unique eigenstructure of the monodromy matrix, all periodic orbits possess at least two eigenvalues equal to one. This characteristic is due to the fact that for a periodic orbit to exist, at least one λ must equal one, and since eigenvalues of the monodromy matrix come in reciprocal pairs, it follows that a second λ also equals one.

3.7 Generating Families of Periodic Orbits

The periodic orbits in Sections 3.6.1 and 3.6.2 are single members of larger families of similar periodic orbits. Each orbit along the family is characterized by some parameter α (for example, x_0, T, J, etc.) where α evolves along the family. Thus, these families are evolved from a single orbit using natural parameter continuation based on α. This strategy employs a solution at one value α as an initial guess to construct a solution at another value, i.e., $\alpha + \Delta\alpha$, where $\Delta\alpha$ is a step size. In this manner, new solutions are produced by stepping through values of the single parameter α. The value for the step size $\Delta\alpha$ is selected so that the initial guess yielded by each step is sufficiently accurate to deliver a new solution to the problem.

Each specific parameter α that can be used to continue Lyapunov orbits evolves the family differently. The simplest and, arguably the most insightful evolution is based on $\alpha = x_0$. In this formulation, x_0 is constant during each evaluation of the single shooting problem, and a fixed step size is possible. New orbits in the Lyapunov family are constructed by stepping along the x-axis towards or away from the Lagrange point. The stability characteristics of the Lyapunov orbits then change as the family evolves and these changes in stability may necessitate changes in step size Δx_0. Portions of the L_1, L_2, and L_3 Lyapunov orbit families in the Earth-Moon system are plotted in Figure 3.5.

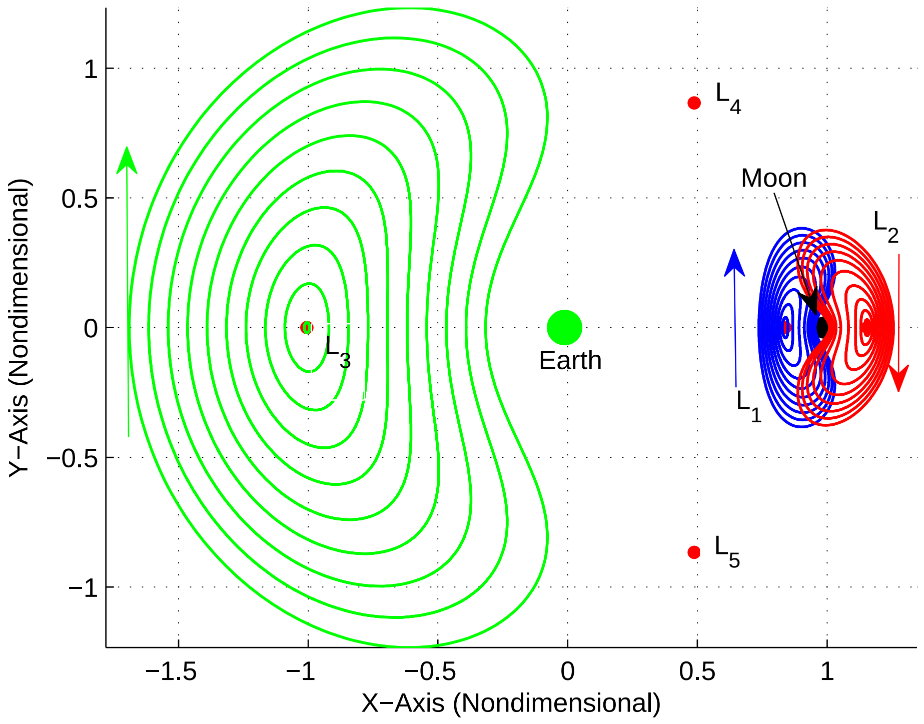


Figure 3.5.: L_1, L_2, and L_3 Lyapunov Families

To apply natural parameter continuation to other families of periodic orbits a similar or modified choice for α is deliberately specified. Either x_0 or z_0, for example, is frequently employed as the continuation parameter to evolve the halo family of orbits. At some stages, the evolution of the family may necessitate switching between x_0 and z_0 as the continuation parameter. Similar trends occur in the axial and vertical families of orbits; Figure 3.6 offers examples of selected regions of the halo and vertical families. The halo family is expanded above and below the x-y plane in Figure 3.6(a), and the resulting orbits are denoted as northern and southern halo orbits, respectively. The sub-sets of the families in Figures 3.5 and 3.6 could be extended with further iterations of the natural parameter continuation process but, eventually, each family reaches a maximum size. However, each family includes an infinite number of orbits and Figures 3.5 and 3.6 display sample orbits at discrete intervals.

3.8 Invariant Manifolds

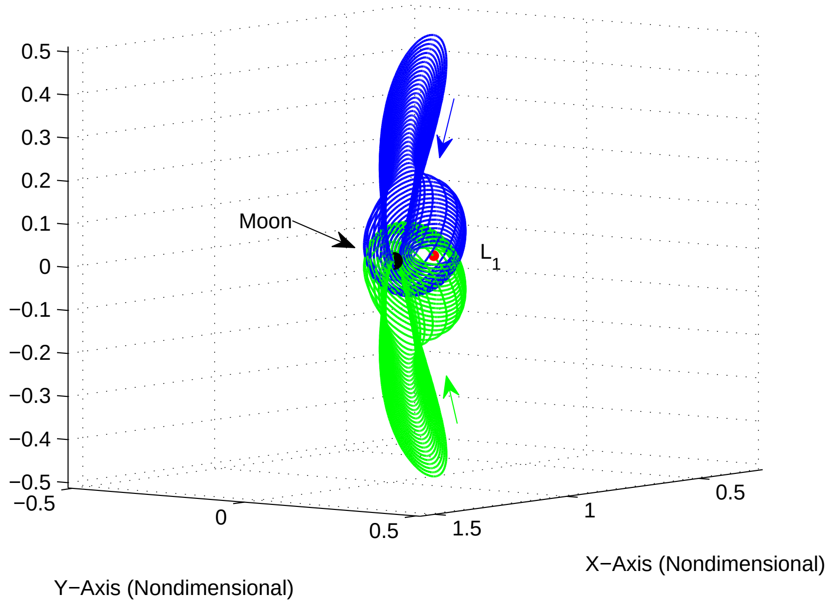
The stability analyses for the equilibrium solutions and periodic orbits in Sections 2.2.8 and 3.6.3 indicate the general behavior of motion near the equilibrium point or periodic orbit. This behavior is further examined via the concept of invariant manifolds. The phase portrait of the flow, supplied by the invariant manifolds, is valuable for designing transfers throughout the P_1-P_2 region, particularly applicable for sun-planet or planet-moon systems.

3.8.1 Manifolds Associated with the Equilibrium Points

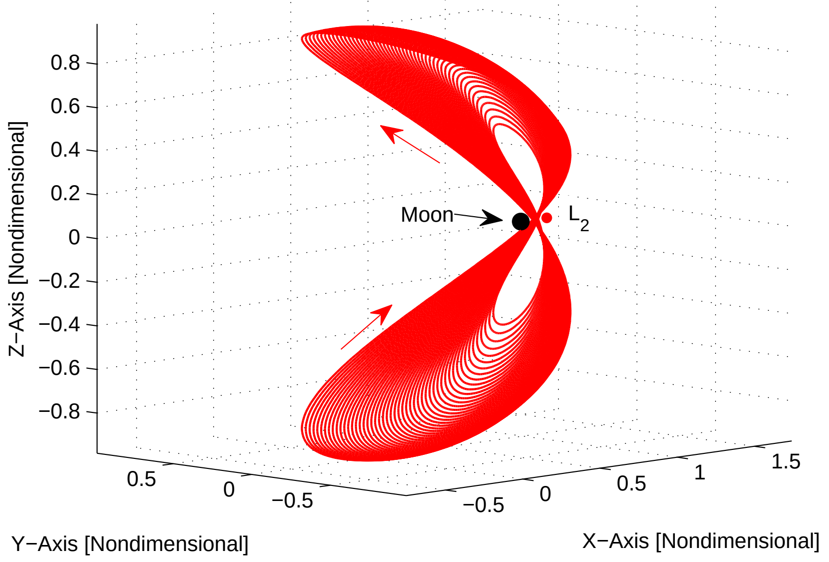
The eigenvalues used to assess stability in Section 2.2.8 emerge from the matrix $\boldsymbol{A}_6$ corresponding to each of the equilibrium solutions. The matrix is defined consistent with equation (2.56), a first order variational equation with a solution such that,

$$\delta\boldsymbol{x}(t) = \sum_{j-1}^{n} c_j e^{\lambda_j(t-t_0)}\boldsymbol{v}_j \tag{3.31}$$

where the coefficients c_j are determined from the initial conditions. Because $\boldsymbol{A}_6$ possesses 6 unique eigenvalues, equation (3.31) is a summation of six distinct terms, or modes. The form of the eigenvalues determines the behavior of the exponential in each of these terms and, thus, the overall behavior of the solution. The modes lead to



(a) Northern (Blue) and Southern (Green) L_1 Halo Families



(b) L_2 Vertical Family

Figure 3.6.: L_1 Halo and L_2 Vertical Periodic Orbit Families

converging, diverging, or oscillatory behavior corresponding to stable, unstable, and marginally stable descriptions, respectively, for solutions to equation (3.31). Assuming a total of n eigenvalues, the λ_j that lead to these three conclusions are classified as follows,

$$n = n_S + n_U + n_C \begin{cases} n_S & \lambda_j \text{ with negative real parts} \\ n_U & \lambda_j \text{ with positive real parts} \\ n_C & \lambda_j \text{ with zero real parts} \end{cases} \tag{3.32}$$

The subscript 'c' in n_c represents 'center' because solutions with this type of eigenvalue evolve on center manifolds. If $n_c = 0$, corresponding to an equilibrium point, then no center manifold exists and the equilibrium point is described as hyperbolic. The eigenvectors associated with each λ_j are linearly independent, therefore, they span the entirety of the space $\mathbb{R}^6$. This six-dimensional space is decomposed into three linear subspaces corresponding to the three types of eigenvalues, E^S, E^U, and E^C.

The eigenspaces are linear structures with counterparts in the nonlinear system of equations. The *Stable Manifold Theorem* states that, if a nonlinear system of first-order differential equations possesses a hyperbolic equilibrium point, then local stable, $W^S_{loc}(\boldsymbol{x}_{eq})$, and unstable manifolds, $W^U_{loc}(\boldsymbol{x}_{eq})$ exist and are of the same dimension n_S and n_U as that of the linear eigenspaces E^S and E^U [31]. Furthermore, the manifolds $W^S_{loc}(\boldsymbol{x}_{eq})$ and $W^U_{loc}(\boldsymbol{x}_{eq})$ are tangent to E^S and E^U, respectively, near the equilibrium point. It is emphasized that the eigenspaces are structures in the linear system defined in equation (2.56) while the manifolds are structures in the corresponding nonlinear system that are tangent to the eigenspaces nearby the equilibrium solution. These relationships are demonstrated in Figure 3.7 which depicts the stable and unstable manifolds corresponding to the L_1 equilibrium solution. The stable and unstable manifolds possess global analogs that flow toward and away from the equilibrium solutions, respectively, and extend far beyond the vicinity of these points. Therefore, propagating backwards in time along $W^S_{loc}(\boldsymbol{x}_{eq})$ produces W^S while propagating forwards in time along $W^U_{loc}(\boldsymbol{x}_{eq})$ results in W^U. The manifolds in the linear and nonlinear problems are invariant, that is, a particle initially placed exactly on one of the manifolds will not depart from it; rather, the state will follow the flow indicated by the path of the manifold. Finally, if the equilibrium point is non-hyperbolic, $n_C \neq 0$, then a similar theorem, the *Center Manifold Theorem*, supports the existence of a center manifold tangent to the center eigenspace E^C in the linearized system [31]. The center manifold is not necessarily unique and may indicate the existence of periodic orbits or quasi-periodic trajectories in the vicinity of the equilibrium point.

Manifolds corresponding to equilibrium points possess a dimension equal to $n_S + n_U$. The numerical computation of the manifolds consists of two parts, determining a point on the local manifold and propagating from that point, both forwards and backwards. Because they are tangent to global manifolds, the eigenspaces are used to compute an initial point on the manifold. The eigenspaces E^S and E^U are line segments along the stable, $\pm\boldsymbol{\nu}_S$, and unstable, $\pm\boldsymbol{\nu}_U$, eigenvectors of $\boldsymbol{A}_6$ respectively. The eigenspaces extend in the positive and negative directions along the eigenvectors. Both components are necessary to compute a full manifold; a half manifold is constructed in each direction. The eigenvectors have the form,

$$\boldsymbol{\nu}_S = \{x_S \; y_S \; z_S \; \dot{x}_S \; \dot{y}_S \; \dot{z}_S\}^T \tag{3.33}$$

$$\boldsymbol{\nu}_U = \{x_U \; y_U \; z_U \; \dot{x}_U \; \dot{y}_U \; \dot{z}_U\}^T \tag{3.34}$$

Only the direction of the eigenvector is required to define the eigenspace, therefore each eigenvector is normalized by its magnitude,

$$\hat{\boldsymbol{\nu}}_S = \frac{\boldsymbol{\nu}_S}{|\boldsymbol{\nu}_S|} \tag{3.35}$$

$$\hat{\boldsymbol{\nu}}_U = \frac{\boldsymbol{\nu}_U}{|\boldsymbol{\nu}_U|} \tag{3.36}$$

A point along a manifold is approximated by taking a step away from the equilibrium point in the direction of one of the eigenspaces. The eigenspaces represent only a first-order approximation of the manifolds, therefore, a nondimensional step, d, along the eigenspace that is too large could produce a point far from one on the actual nonlinear manifold. Conversely, manifolds approach and depart from the equilibrium point asymptotically, therefore, choosing a value for d that is too small requires long propagation times. The sensitivity of the equilibrium point is typically incorporated in selecting a value of d, however, typically in the CR3BP a dimensional value of $d = 100$ is sufficient to achieve accurate results,

$$\boldsymbol{x}_S = \boldsymbol{x}_{eq} \pm d\hat{\boldsymbol{\nu}}_S \tag{3.37}$$

$$\boldsymbol{x}_U = \boldsymbol{x}_{eq} \pm d\hat{\boldsymbol{\nu}}_U \tag{3.38}$$

Once an initial point along a manifold is located, it can serve as an initial condition for a numerical integration process to that propagate the manifold states forward or backwards in time. The initial points in both the positive and negative directions must be located and propagated to produce a full manifold.

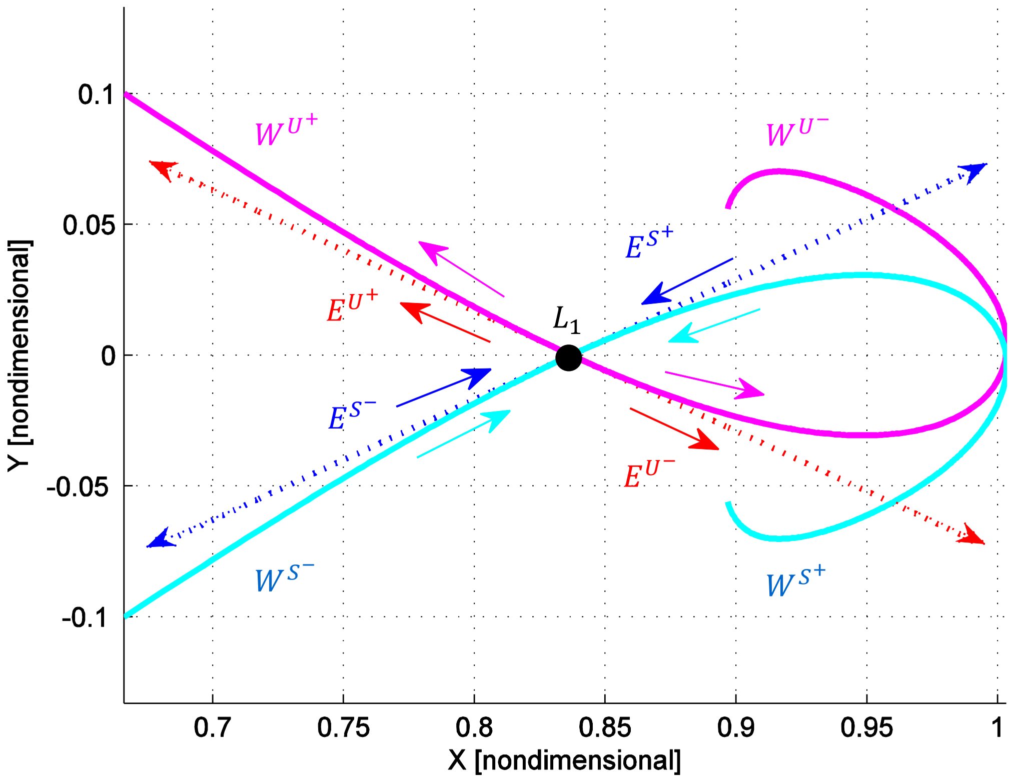


Figure 3.7.: Stable and Unstable Local Manifolds at L_1

3.8.2 Manifolds Associated with Periodic Orbits

Manifold structures also exist for fixed points along periodic orbits and are computed in a manner similar to the manifolds associated with equilibrium points. These manifolds asymptotically approach or depart a periodic orbit, therefore, such structures are useful for designing transfers to or from periodic orbits. However, the natural flow reflected in the evolution of manifolds can also be leveraged for other spacecraft destinations.

Computation of periodic orbit manifolds follows from the discussion of periodic orbit stability in Section 3.6.3. The eigenvalues in that analysis signal stable, unstable, or marginally stable behaviors depending upon the modulus of the eigenvalues. The number of each eigenvalue type is once again assigned to three different categories and sum to n, the total number, i.e.,

$$n = n_S + n_U + n_C \begin{cases} n_S & |\lambda_j| < 1 \\ n_U & |\lambda_j| > 1 \\ n_C & |\lambda_j| = 1 \end{cases} \tag{3.39}$$

The three eigenvalue types correspond to stable, unstable, and center subspaces, each with dimensions indicated by n_S, n_U, and n_C, respectively. Analogs of the stable and center manifold theorems exist for fixed points along periodic orbits. These theorems support the existence of local stable $W^S_{loc}(\Gamma)$, unstable $W^U_{loc}(\Gamma)$, and center $W^C_{loc}(\Gamma)$ manifolds associated with a periodic orbit Γ. These local manifolds are then tangent to the stable, unstable, and center eigenspaces, $E^S(\Gamma)$, $E^U(\Gamma)$, and $E^C(\Gamma)$. The rate at which manifolds approach or depart a periodic orbit is determined by the stability of the orbit. When an orbit is highly stable the unstable manifold, defined by a real eigenvalue with magnitude greater than one, requires a longer time to depart the orbit while the stable manifold takes longer to approach it. The opposite is true for highly unstable periodic orbits where the unstable and stable manifolds, respectively, depart and approach the periodic orbit more quickly.

The computation of invariant manifolds are initiated from any fixed point along a periodic orbit and, because the theory is largely analogous, the computation of periodic orbit manifolds generally follows the same steps as in Section 3.8.1. The eigenvalues of the monodromy matrix are independent of the fixed point along a periodic orbit, however, the eigenvectors evolve from one fixed point to the next. These eigenvectors are computed at each fixed point from the monodromy matrix. Recall that the monodromy matrix is defined as the STM after one complete revolution. A new monodromy matrix at each fixed point could be calculated by propagating over a complete revolution at each fixed point, but such an approach is inefficient. Rather, a similarity transformation is employed to generate the eigenvectors at any point "downstream" from the initial fixed point on the periodic orbit through the following relationship,

$$\boldsymbol{\nu}_i(t_1) = \boldsymbol{\phi}(t_1, t_0)\boldsymbol{\nu}(t_0) \tag{3.40}$$

In Equation (3.40), the STM from t_0 to t_1 advances the eigenvectors from t_0 to t_1. The eigenvectors are then nondimensionalized to calculate the directions of the eigenspaces.

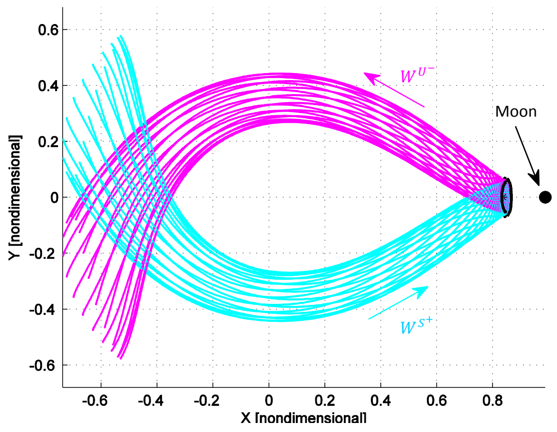
$$\hat{\boldsymbol{\nu}}\boldsymbol{(t_1)}_S = \frac{\boldsymbol{\nu}_S(t_1)}{|\boldsymbol{\nu}_S(t_1)|} \tag{3.41}$$

$$\hat{\boldsymbol{\nu}}_U(t_1) = \frac{\boldsymbol{\nu}_U(t_1)}{|\boldsymbol{\nu}_U(t_1)|} \tag{3.42}$$

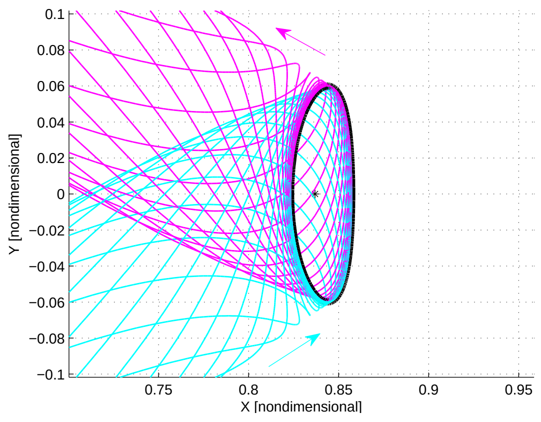
Recall the 6-D eigenvectors are nondimensionalized using their magnitude. When the directions of the eigenspaces are constructed, an initial point on the manifold surface is determined by stepping off of the periodic orbit in the direction of one of the eigenspaces. The value of the step size is sufficiently small such that the first-order approximation remains valid, but also large enough that the numerical integration time is reasonable, i.e.,

$$\boldsymbol{x}_S(t_1) = \boldsymbol{x}_{eq}(t_1) \pm d\hat{\boldsymbol{\nu}}_S(t_1) \tag{3.43}$$

$$\boldsymbol{x}_U(t_1) = \boldsymbol{x}_{eq}(t_1) \pm d\hat{\boldsymbol{\nu}}_U(t_1) \tag{3.44}$$



(a) Global Manifolds of L_1 Halo Orbit



(b) Region about L_1 Periodic Orbit

Figure 3.8.: Stable and Unstable Manifolds Associated with an L_1 Halo Orbit as Viewed in Configuration Space; the Manifold Structures are Represented by Trajectories Along the Surface

The initial conditions for the stable and unstable manifolds approaching and departing an L_1 halo orbit are constructed and then propagated for $t = 5$ nondimensional time units. The resulting 6D paths are projected onto configuration space and the two-dimensional x-y position history is viewed in Figure 3.8. The invariant manifolds in position space actually form a three-dimensional invariant surface with a tube-like structure. The three dimensional velocity space can also be represented but are not plotted here. The manifolds corresponding to other periodic orbits form similar surfaces that suggest low-cost structures that may be exploited to transfer between regions of space in the CR3BP.

4. INDIRECT OPTIMIZATION

Mission design for spacecraft equipped with low-thrust engines typically requires the use of optimization techniques; and indirect methods are an approach that has yielded useful results and significant insights. Indirect optimization methods are generally based on analytical techniques derived from the calculus of variations and have been developed since the conception of this field in 1696. Indirect optimization employs the necessary conditions from the calculus of variations to formulate a two-point boundary value problem (TPBVP) that renders an optimal solution. These techniques reflect the classical approach to optimization having been employed long before the ubiquity of computers. However, computational power and numerical methods have enabled the evaluation of more complex TPBVPs, broadening the scope of problems for application via indirect optimization techniques.

The theory behind indirect optimization is introduced, and applied to the general low-thrust variable specific impulse (VSI) problem. One of the primary disadvantages of an indirect approach is the non-physical nature of the costate variables. This challenge is alleviated by introducing an adjoint control transformation, that is, a scheme that relates the costates to physical values and, thus, enables a simpler initial guess. Presentations of indirect optimization techniques and the adjoint control transformation are found in numerous textbooks, however this content is reviewed to set up several sample problems that demonstrate the strengths and weaknesses of an indirect optimization method.

4.1 Euler-Lagrange Theory

The familiar Euler-Lagrange (E-L) theory transforms the problem of optimizing a cost function into a well-defined two-point boundary value problem (TPBVP). Originating from the calculus of variations, Lagrange developed the approach in response to the challenges of the brachistochrone problem. The E-L Theorem provides a set of necessary conditions to be satisfied that enable the optimal control, defined in terms of the control vector, $\boldsymbol{u}^*(t)$, to minimize the scalar cost function, J. The number of these conditions is sufficient to ascertain a unique local solution to the TPBVP, thus, it is denoted as "well-defined". The E-L theorem is an indirect optimization approach because an optimal solution satisfies a set of boundary conditions rather than one that directly minimizes the cost function. A solution that satisfies the necessary conditions of the E-L theorem is guaranteed to be a locally extremal solution, and additional rules are available to verify that the solution is a minimum or a maximum. Greater detail on the derivation and application of the E-L theory are available in the text *Optimal Control with Aerospace Applications* by Longuski, et al. [6].

An optimization problem is distinguished from a general trajectory design problem by the inclusion of an objective function, this expression incorporates the quantities of importance that determine the optimality of a solution. The objective function, or cost function, produces a scalar value, J, to be minimized,

$$\text{Min. } J = \underbrace{\phi(t_f, \boldsymbol{x}_f)}_{\text{Terminal Cost}} + \underbrace{\int_{t_0}^{t_f} L(t, \boldsymbol{x}, \boldsymbol{u}) dt}_{\text{Path Cost}} \tag{4.1}$$

The integrand, L, is termed the Lagrangian and is dependent upon m control variables, compromising the vector $\boldsymbol{u}$, in addition to n states, that is, $\boldsymbol{x}$ and time, t. The cost function in equation (4.1) is generally formulated as a problem of Bolza, containing both terminal and path components of the cost. A cost function consisting only of a terminal cost is in a Mayer form while a cost function with only a path component is in Lagrange form.

Application of the E-L Theorem to an optimal control problem is divided into a series of three distinct steps. The first of these steps is forming the Hamiltonian, i.e.,

$$H = L(t, \boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^T \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}) \tag{4.2}$$

where $\boldsymbol{f}(t,\boldsymbol{x},\boldsymbol{u}) = \dot{\boldsymbol{x}}$ is compromised of the n system scalar equations and the $\boldsymbol{\lambda}$ vector contains Lagrange multipliers also called costates. The second step in the process of applying the E-L theorem to an optimal control problem is to evaluate the partial derivatives of the Hamiltonian according to the E-L necessary conditions. This computation results in both differential and algebraic equations,

$$\dot{\boldsymbol{\lambda}} = \frac{\partial H}{\partial \boldsymbol{x}} = H_{\boldsymbol{x}} \tag{4.3}$$

$$\frac{\partial H}{\partial \boldsymbol{u}} = H_{\boldsymbol{u}} = \mathbf{0} \tag{4.4}$$

Equation (4.3) requires a partial derivative with respect to each of the n state variables, therefore, this necessary condition results in n differential equations for the costates. Equation (4.4) produces m algebraic equations from the partial derivatives of H with respect to the control variables. A TPBVP is formed from the application of the first two steps of the Euler-Lagrange theory to the optimal control problem, however the resulting problem may be ill-defined due to an insufficient number of boundary conditions.

The third step in the process of applying the Euler-Lagrange theory to an optimal control problem supply additional boundary conditions. An optimal control problem includes a set of p boundary conditions that define necessary initial or final conditions for the problem, and these are collected into a vector such that,

$$\boldsymbol{\psi}(t,\boldsymbol{x}) = \mathbf{0} \tag{4.5}$$

To pose the optimal control problem as a well-defined TPBVP, a total of $2n+2$ boundary conditions are required. The initial number of boundary conditions rarely satisfies this requirement, therefore, additional conditions are typically added. The initial conditions for the state variables and time supply $n+1$ conditions; therefore, the necessary number of additional conditions is $2n+2-(n+1)-p = n+1-p$. To derive the $n+1-p$ boundary conditions for a well-defined TPBVP, the terminal boundary constraints are first differentiated, i.e.,

$$d\boldsymbol{\psi} = \boldsymbol{\psi}_{t_f} dt_f + \boldsymbol{\psi}_{x_f} dx_f = \mathbf{0} \tag{4.6}$$

The known boundary conditions are substituted into equation (4.6), producing p independent equations expressed in terms of the $n+1-p$ remaining unknown differentials. The p independent equations are substituted into the transversality condition that initially includes $n+1$ differential terms,

$$H_f dt_f - \boldsymbol{\lambda}_f^T d\boldsymbol{x}_f + d\phi = 0 \tag{4.7}$$

Following this substitution, the transversality condition contains $n+1-p$ terms. The coefficients of each term in equation 4.7 are set to zero and these $n+1-p$ expressions supply the final boundary conditions to complete the required number. The result of these operations is $2n+2$ boundary conditions that allow a well-defined TPBVP to be formulated. This method for deriving the additional boundary conditions required for the TPBVP is termed the un-adjoined method and is one of two approaches. The alternate method involves the introduction of extra adjoining variables and, hence, is denoted the adjoined method. The usage of the adjoined technique has increased since its inclusion in several key texts such as Bryson and Ho [8] in 1969 and it is the predominate method taught in modern textbooks [32]. However, both methods are presented by Citron [33] and Longuski [6]. The popularity of the adjoined method is largely due to the ease with which it is implemented in various types of numerical methods, which are required to solve all but the most elementary of optimal control problems.

Application of the E-L theorem alone is sometimes insufficient to fully solve the optimal control problem. In such circumstances, the minimum principle is also applied to determine a control law that determines $\boldsymbol{u}^*(t)$ and, thus, minimizes the cost function, J. The most general form of this condition is Pontryagin's Minimum Principle [34], often simply identified as the Minimum Principle,

$$H[t, \boldsymbol{x}^*(t), \boldsymbol{u}^*(t), \boldsymbol{\lambda}(t)] \leq H[t, \boldsymbol{x}^*(t), \boldsymbol{u}(t), \boldsymbol{\lambda}(t)] \tag{4.8}$$

Bryson and Ho [8] offer McShane's succinct summary of the Minimum Principle as: "H^* must be minimized over the set of all admissible $\boldsymbol{u}$". Admissible controls for equation (4.8) include a control vector that is continuous or piecewise as well as bounded or unbounded. The Weierstrass necessary condition is a similar, although more restrictive version of the Minimum Principle that applies only to scenarios with unbounded continuous control. However, many concepts in spacecraft trajectory optimization require piecewise bounded control, for example, the "bang-bang" control history of a constant specific impulse low-thrust spacecraft, therefore, the general Minimum Principle is more frequently applied. Finally, if a maximum value of the cost function is sought, the inequality in equation (4.8) is reversed to produce the corresponding Maximum Principle. Application of the Minimum or Maximum Principles ensures that the desired type of extremal is obtained when the TPBVP is solved.

The E-L algebraic necessary condition in Equation (4.4) is only applicable to optimal control problems with unbounded control. When the value of a scalar control variable u is bounded, for example $|u| \leq u_{max}$, then the partial derivative of the Hamiltonian with respect to the control may not be zero, rather,

$$\frac{\partial H}{\partial u} = H_1 \tag{4.9}$$

where the variable H_1 corresponds to a Hamiltonian assumed to be linear with respect to u, that is,

$$H = H_0(t, \boldsymbol{x}, \boldsymbol{\lambda}) + H_1(t, \boldsymbol{x}, \boldsymbol{\lambda})u \tag{4.10}$$

The sign of H_1 is employed, in combination with the minimum principle, to determine a value for u. According to the maximum principle u is such that H is maximized, thus the following rule guides the choice of u.

$$u = \begin{cases} -u_{max} & \text{if} \quad H_1 < 0 \\ \text{undetermined} & \text{if} \quad H_1 = 0 \\ u_{max} & \text{if} \quad H_1 > 0 \end{cases} \tag{4.11}$$

If an optimal control problem is formulated to minimize a given objective, then the minimum principle applies and the first and last elements of equation (4.11) are reversed. Note that when a bounded control variable is a vector the approach just described for scalar control values applies to each component of the vector variable. Therefore, algebraic equations for the control variables can be derived even when the control is bounded.

4.2 Application of Euler-Lagrange Theory to the Low-Thrust VSI Transfer Problem

The literature on low-thrust mission design generally focuses on two types of engine model based upon constant specific impulse (CSI) or variable specific impulse (VSI). Constant Specific Impulse engines operate at fixed thrust and power levels, therefore, the impulse is constant. Trajectories delivered via spacecraft with CSI engines are decomposed into individual thrusting and coasting segments during which the engine is either turned on or off, respectively. This type of thrusting scheme results in a "bang-bang" control structure for the engine control variables, and the optimal pattern for this structure can be difficult to predict *a priori*. Recent work on low-thrust mission design utilizing CSI engines has been conducted by Russell [35], Tang and Jiang [36], as well as Rasotto et al [37]. Variable Specific Impulse engines allow the thrust and power levels of an engine to vary which eliminates the need to determine a specific thrust-coast control structure *a priori*. While an initial guess for the path is still required, numerical convergence for spacecraft trajectories incorporating VSI engines is aided by the lack of control discontinuities. Additionally, Ranieri and Ocampo [38] have demonstrated via simulations that, in some scenarios, VSI engines offer greater fuel efficiency than CSI. This advantage has spurred further investigation into development and demonstrations of VSI engines and missions which employ them. However, to date no spacecraft equipped with VSI engines have yet been implemented in flight. All low-thrust spacecraft, thus far, have employed CSI engines with the Hayabusa and Dawn missions being two of many successful examples. Ultimately, the VSI engine model is employed in this investigation due to its amenable numerical properties.

4.2.1 Framework for the Low-Thrust VSI Optimal Transfer Problem

The indirect optimal control problem for VSI equipped spacecraft is formulated by applying the Euler-Lagrange theory, detailed in Section 4.1, to the dynamical model of a VSI spacecraft. When the VSI engine is in operation, a mass state, m, is added to the six position and velocity states typically associated with natural dynamical motion, i.e.,

$$\boldsymbol{\chi} = \left\{ \begin{matrix} \boldsymbol{r} \\ \boldsymbol{v} \\ m \end{matrix} \right\} \tag{4.12}$$

Acceleration terms reflecting an engine thrust model are incorporated into the first order differential equations governing the motion of a particle representing the spacecraft to obtain equations of motion for a thrusting spacecraft.

$$\dot{\boldsymbol{\chi}} = \boldsymbol{f}_T(t, \boldsymbol{\chi}, \boldsymbol{u}) = \left\{ \begin{matrix} \dot{\boldsymbol{r}} \\ \dot{\boldsymbol{v}} \\ \dot{m} \end{matrix} \right\} = \left\{ \begin{matrix} \boldsymbol{v} \\ \boldsymbol{f}_N(\boldsymbol{r}, \boldsymbol{v}) + \frac{T}{m}\boldsymbol{u}_T \\ -\frac{T^2}{2P} \end{matrix} \right\} \tag{4.13}$$

Note that five scalar scalar control variables appear in equation 4.13, i.e.,

The natural dynamical vector force term, $\boldsymbol{f}_N(\boldsymbol{r}, \boldsymbol{v})$, is determined based on the dynamical model employed in the problem, for example, the 2BP or CR3BP models. Note that up to five scalar scalar control terms appear in equation 4.13, i.e.,

$$\boldsymbol{u} = \left\{ \begin{matrix} \hat{\boldsymbol{u}}_T \\ T \\ P \end{matrix} \right\} \tag{4.14}$$

these determine the motion of a low-thrust spacecraft, in addition to the seven states, $\boldsymbol{\chi}$. Only three of the five control terms are independent for any given optimal control problem. Relationships between the control terms enable the values of the two dependent control terms to be determined from the values of the three independent control variables. The thrust pointing vector, $\hat{\boldsymbol{u}}_T$, is a unit vector that defines the direction of thrust, thus the components of this vector are related to one another by the requirement that $\hat{\boldsymbol{u}}_T$ have unity magnitude. The magnitude of the full thrust vector is defined by the scalar value T, where,

$$T = \frac{2P}{I_{sp} g_0} \tag{4.15}$$

where g_0 is the acceleration due to gravity on the surface of the Earth, approximated as $9.80665 \frac{m}{s^2}$. Equation 4.15 relates the control term T to the control term, P, that defines the engine power level. While T is left unconstrained the power level, P, is bounded to be within the range, $0 \leq P \leq P_{max}$. Clearly, changes to the values of T and P in equation (4.15) impact the value of engine specific impulse, I_{sp}. Note that when the optimal control problem is setup such that dependent control terms are allowed to vary then additional constraints are included to enforce the relationships between the control terms. Alternately, these same relationships can be leveraged to eliminate the dependent control terms form the framework. Together the state and control variables describe the trajectory of a spacecraft within the dynamical model.

In the optimal control problem for low-thrust spacecraft a trajectory is desired that extremizes a scalar cost. A common cost, or objective, of low-thrust mission design is the minimization of fuel consumption over the duration of a transfer, this corresponds to a cost function in Mayer form.

$$\text{Max } J = m(t_f) \tag{4.16}$$

The Euler-Lagrange theory from Section 4.1 is applied to the optimal control problem defined by equation (4.16) to determine the state and control variables that produce an optimal solution. First, the Hamiltonian is formed, consistent with equation (4.2). Equation (4.16) includes no path cost term, thus, $L = 0$ and the Hamiltonian is, written

$$H = \boldsymbol{\lambda}^T \dot{\boldsymbol{\chi}} = \boldsymbol{\lambda}_{\boldsymbol{r}}^T \boldsymbol{v} + \boldsymbol{\lambda}_{\boldsymbol{v}}^T \left[\boldsymbol{f}_N(\boldsymbol{r}, \boldsymbol{v}) + \frac{T}{m} \boldsymbol{u}_T \right] - \lambda_m \frac{T^2}{2P} \tag{4.17}$$

where H is not an explicit function of time. Because the Hamiltonian is autonomous with respect to time, H is constant for the duration of the problem; thus the Hamiltonian is a useful value for determining the error in a numerical solution to the optimal control problem.

The constrained and unconstrained control variables in the optimal control problem are addressed differently by Euler-Lagrange theory. The thrust pointing vector $\hat{\boldsymbol{u}}_T$ is constrained, therefore, the partial derivative of the Hamiltonian with respect to $\boldsymbol{u}_T$ appears as in equation (4.9), i.e.,

$$\frac{\partial H}{\partial \hat{\boldsymbol{u}}_T} = \boldsymbol{\lambda}_{\boldsymbol{v}}^T \frac{T}{m} = \boldsymbol{H}_{\hat{\boldsymbol{u}}_T} \tag{4.18}$$

where $\boldsymbol{H}_{\hat{\boldsymbol{u}}_T}$ is analogous to H_1 in equation 4.9, but is a vector because the partial of H is calculated with respect to $\hat{\boldsymbol{u}}_T$. The cost function in equation (4.16) reflects a goal to maximize J, therefore, H must also be maximized and equation (4.11) guides the choice of $\hat{\boldsymbol{u}}_T$. Assuming the thrust magnitude and mass are positive values, the value of the Hamiltonian is maximized when $\hat{\boldsymbol{u}}_T$ is parallel to and oriented as $\boldsymbol{\lambda}_{\boldsymbol{v}}$, i.e.,

$$\boldsymbol{u}_T = \frac{\boldsymbol{\lambda}_{\boldsymbol{v}}}{\|\boldsymbol{\lambda}_{\boldsymbol{v}}\|} \tag{4.19}$$

This observation was first noted by Lawden who termed $\boldsymbol{\lambda}_{\boldsymbol{v}}$ the *primer vector* due to its significance [7]. Engine power is also constrained, thus, the partial derivative of the Hamiltonian with respect to P is,

$$\frac{\partial H}{\partial P} = \lambda_m \frac{T^2}{2P^2} = H_P \tag{4.20}$$

where H_P corresponds to H_1 in equation 4.9. If λ_m is assumed to be a positive value, then,

$$P = P_{max} \tag{4.21}$$

maximizes the Hamiltonian. The remaining control variable, T, is unconstrained, therefore, the partial derivative of the Hamiltonian with respect to T takes the form of equation (4.4),

$$\boldsymbol{\lambda}_{\boldsymbol{v}}^T \frac{\boldsymbol{u}_T}{m} - \frac{2\lambda_m}{2P} = 0 \tag{4.22}$$

$$T = \frac{\|\boldsymbol{\lambda}_{\boldsymbol{v}}\| \, P}{\lambda_m m} \tag{4.23}$$

The values obtained for the control variables $\boldsymbol{u}_T$ and P are substituted into equation (4.22) and the resulting expression is solved for T resulting in equation (4.23). The relationship $\boldsymbol{\lambda}_{\boldsymbol{v}}^T \hat{\boldsymbol{u}}_T = \|\boldsymbol{\lambda}_{\boldsymbol{v}}\|$ is used in this simplification process. Similar to equation 4.15, equation 4.23 demonstrates the relationship between T and P and indicates that these control terms cannot both be independent variables.

Several observations simplify the setup and analysis of the expressions resulting from application of the first two steps of the Euler-Lagrange theory. First, the control variable expressions obtained in equations (4.19), (4.21), and (4.23) aid in reformulating the Hamiltonian in terms of a switching function S,

$$H = \boldsymbol{\lambda}_{\boldsymbol{r}}^T \boldsymbol{v} + \boldsymbol{\lambda}_{\boldsymbol{v}}^T \boldsymbol{f}_N + S \cdot T \tag{4.24}$$

$$S = \frac{\|\boldsymbol{\lambda}_{\boldsymbol{v}}\|}{m} - \frac{\lambda_m T}{2P_{max}} \tag{4.25}$$

The switching function is a useful formulation for understanding the total effect of the control values on the Hamiltonian. Next, the Euler-Lagrange necessary condition in equation (4.3) indicates that differential equations for the costates are derived from the partial derivatives of the Hamiltonian with respect to the state variables,

$$\dot{\boldsymbol{\lambda}} = -\frac{\partial H}{\partial \boldsymbol{\chi}}^T = \begin{Bmatrix} -\boldsymbol{\lambda}_{\boldsymbol{v}}^T \left(\frac{\partial \boldsymbol{f}_N}{\partial \boldsymbol{r}}\right) \\ -\boldsymbol{\lambda}_{\boldsymbol{r}}^T - \boldsymbol{\lambda}_{\boldsymbol{v}}^T \left(\frac{\partial \boldsymbol{f}_N}{\partial \boldsymbol{v}}\right) \\ \|\boldsymbol{\lambda}_{\boldsymbol{v}}\| \frac{T}{m^2} \end{Bmatrix} \tag{4.26}$$

The differential equation for λ_m, the final one in equation 4.26, indicates that this costate is monotonically increasing; moreover, further analysis suggests that the initial mass costate is equal to one. Thus, the value of λ_m is always positive. The final step of applying Euler-Lagrange theory to the optimal VSI problem is to construct the additional boundary conditions via transversality. The added conditions allow a well-defined TPBVP to be posed. The selection of boundary conditions is highly problem-dependent, thus, the final step is reserved for sample problems.

4.2.2 Example: Circular Orbit Transfer

An efficient transfer between the circular orbits can be posed in terms of optimal propellant usage. The form of the optimal propellant transfer between two circular orbits is dependent upon the type of engine employed. The optimal propellant impulsive transfer between two circular orbits in the 2BP was first developed by Hohmann in 1925 [39]. This transfer requires two thruster burns assumed to be instantaneous, however, this assumption does not apply for low-thrust burns, thus, Hohmann transfers are not practical for application to low-thrust spacecraft. Circular orbit transfers for low-thrust spacecraft generally employ spiral trajectories for orbit lowering and raising maneuvers. Spiral transfers can use weeks to months to accomplish the objective, however, the total propellant consumption may be far less than the propellant required for impulsive transfers [40]. Clearly, an obvious trade-off is apparent. Optimal low-thrust circular-to-circular orbit transfer is a useful example for a straightforward optimal control problem. Yet, the spiral behavior in this problem represents non-trivial dynamical and numerical challenges for a variety of optimization methods.

The framework for the circular-to-circular orbit transfer optimal control problem for VSI engines is detailed in terms of the formulation from in Section 4.2.1. The planar circular orbit transfer problem is posed in a two-body dynamical model, thus, the equations of motion for the natural dynamics are,

$$\begin{gathered}\dot{\boldsymbol{\chi}} = \boldsymbol{f}_N(\boldsymbol{r}, \boldsymbol{v}) \\ \begin{Bmatrix} \dot{\rho} \\ \dot{\theta} \\ \dot{p} \\ \dot{q} \end{Bmatrix} = \begin{Bmatrix} p \\ \frac{q}{\rho} \\ \frac{q^2}{\rho} - \frac{1}{\rho^2} \\ \frac{-pq}{\rho} \end{Bmatrix}\end{gathered} \tag{4.27}$$

In this application, rather than than Cartesian coordinates, polar quantities are better suited for numerical computation. The position components in terms of the polar coordinate are $\boldsymbol{r} = \{\rho, \theta\}$, where ρ is the radial distance from the origin and θ is the angle measured counter-clockwise from the $\hat{\boldsymbol{X}}$ axis. The velocity components are defined as $\boldsymbol{v} = \{p, q\}$, where p and q are the radial and tangential components of velocity, respectively. The transfer is assumed to be entirely planar, thus, no out-of-plane components are included in the equations of motion. Figure 4.1 illustrates the configuration variables.

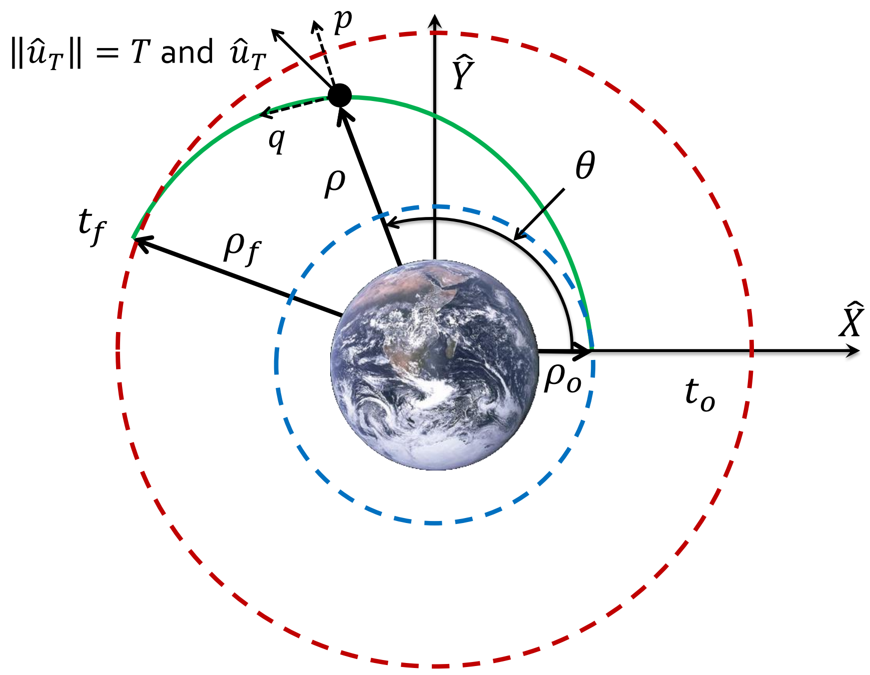


Figure 4.1.: Low-Thrust Circular-to-Circular Orbit Transfer Diagram

The boundary conditions for the circular-to-ciruclar orbit transfer problem distinguish it from the general formulation of the VSI optimal control problem. The transfer is constrained to originate on a circular orbit with known initial conditions, such that,

$$\boldsymbol{\psi}_0 = \begin{Bmatrix} \rho_0 - R_0 \\ \theta_0 \\ q_0 - \sqrt{\frac{1}{R_0}} \\ p_0 \\ m_0 - M_0 \end{Bmatrix} = \mathbf{0} \tag{4.28}$$

where R_0 and M_0 represent a known initial radius and mass, respectively. The final circular orbit is fixed, however, the final values of θ and m remain free,

$$\boldsymbol{\psi}_f = \begin{Bmatrix} \rho_f - R_f \\ p_f \\ q_f - \sqrt{\frac{1}{R_f}} \end{Bmatrix} = \mathbf{0} \tag{4.29}$$

where R_f is a known final radius. Equations (4.28), (4.29), and the fixed initial and final transfer times, $\begin{bmatrix} t_0 \ t_f \end{bmatrix}$, supply 10 boundary conditions. The planar circular orbit transfer problem possesses five equations of motion, thus $n = 5$, and $2(5) + 2 = 12$ boundary conditions are required to pose a well-defined TPBVP.

Two more boundary conditions are necessary to pose a well-defined TPBVP, and these conditions are obtained by applying the transversality condition. The general transversality condition supplied in equation (4.7), is applied to the end of the circular-to-circular orbit transfer to obtain,

$$H_f dt_f - \lambda_{\rho_f} d\rho_f - \lambda_{\theta_f} d\theta_f - \lambda_{p_f} dp_f - \lambda_{q_f} dq_f - \lambda_{m_f} dm_f + dm_f = 0 \tag{4.30}$$

The transversality condition is subject to the expression $d\boldsymbol{\psi}_f = \mathbf{0}$. Therefore, the differential terms corresponding to the fixed final boundary conditions in equation (4.30) equal zero, and these terms are removed from the expression to yield,

$$-\lambda_{\theta_f} d\theta_f + \left(-\lambda_{m_f} + 1\right) dm_f = 0 \tag{4.31}$$

The terms $d\theta_f$ and dm_f are both nonzero because these quantities can vary during the iterations for solving the optimization problem. Therefore, for equation (4.31) to be true, the coefficients of the remaining differential terms must equal zero. The coefficient expressions provide the final two boundary conditions that are required to pose a well-defined TPBVP.

The TPBVP is then solved using the single shooting method described in Section 3.4. The 5×1 design vector corresponding to the single-shooting formulation includes the initial valeus of the costates,

$$\boldsymbol{X} = \begin{Bmatrix} \lambda_{\rho_0} \\ \lambda_{\theta_0} \\ \lambda_{p_0} \\ \lambda_{q_0} \\ \lambda_{m_0} \end{Bmatrix} \tag{4.32}$$

The corresponding 5×1 constraint vector is then comprised of the final boundary conditions,

$$\boldsymbol{F}(\boldsymbol{X}) = \begin{Bmatrix} \rho_f - R_f \\ p_f \\ q_f - \sqrt{\frac{1}{R_f}} \\ \lambda_{\theta_f} \\ \lambda_{m_f} - 1 \end{Bmatrix} = \boldsymbol{0} \tag{4.33}$$

Iterating using Newton's method, convergence to the design variable values that satisfy the constraints, to within a given tolerance, is achieved. The resulting design variables are employed to propagate and produce the final solutions for a continuous low-thrust VSI transfer.

4.2.3 Example: Halo to Halo Transfer

A direct transfer between halo orbits in the vicinity of different libration points is inherently a three-dimensional transfer problem. Additional complexities are introduced in comparison to the circular-to-circular planar transfer problem as formulated in the 2BP. However, the invariant manifolds associated with periodic orbits in the

CR3BP can be leveraged to yield low-energy transfers between different regions in the primary system. Because manifolds approach and depart from periodic orbits asymptotically, these structures are particularly useful for designing transfers to and from periodic orbits. Invariant manifolds offer heteroclinic and homoclinic connections between orbits that require no change in energy implying that the originating and destination orbits possess the same energy level. Alternately, an engine is exploited to change energy levels and facilitate connections between manifolds associated with a wide range of initial and final orbits, expanding transfer options. The continuous and gradual change in energy supplied by low-thrust engines enable this type of propulsion and are particularly well suited for attaining efficient transfers using invariant manifolds.

A low-thrust transfer between periodic orbits leveraging invariant manifolds can involve any of the infinite number of trajectories along a manifold surface that depart from a periodic orbit. The specific manifold trajectory selected is designated by a nondimensional time parameter, τ, that denotes the "time" along a periodic orbit at which a manifold trajectory is constructed. The efficiency of a transfer may also benefit from a period of coasting along a departure or arrival manifold trajectory; this coast time is represented by the nondimensional time parameter, γ. Figure 4.2 illustrates the definition of τ and γ define the manifolds used for a transfer; these definitions are demonstrated to be very effective by both Stuart and Howell [10] as well as Haapala and Howell [41]. The variables τ and γ are incorporated into the problem formulation for a mass optimal transfer between periodic orbits leveraging manifold arcs. The general formulation detailed in Section 4.2.1 and is posed in the

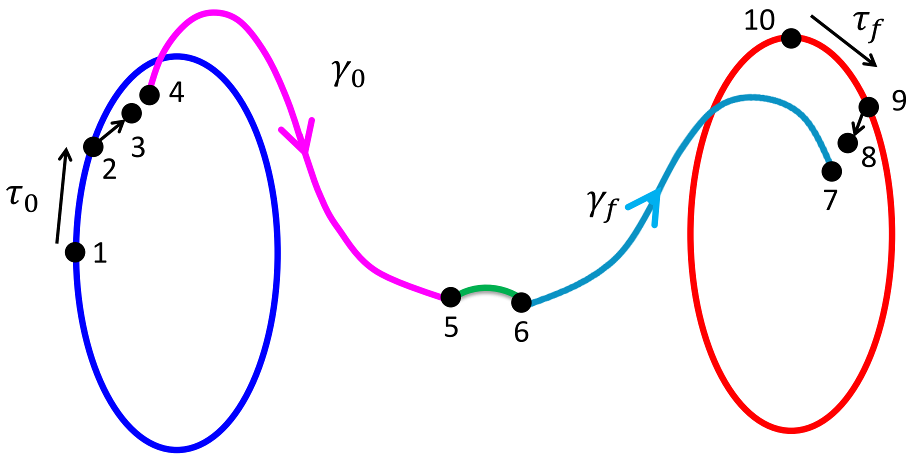


Figure 4.2.: Periodic Orbit Transfer Leveraging Invariant Manifolds

CR3BP dynamical model. Therefore, the equations of motion incorporating only gravitational forces are,

$$\dot{\boldsymbol{\chi}} = \boldsymbol{f}_N(\boldsymbol{r}, \boldsymbol{v})$$
$$\begin{Bmatrix} \dot{\boldsymbol{r}} \\ \dot{\boldsymbol{v}}_x \\ \dot{\boldsymbol{v}}_y \\ \dot{\boldsymbol{v}}_z \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{v} \\ 2\dot{y} + \frac{\partial U^*}{\partial x} \\ -2\dot{x} + \frac{\partial U^*}{\partial y} \\ \frac{\partial U^*}{\partial z} \end{Bmatrix} \tag{4.34}$$

as originally derived in Section 2.2.3. To complete the definition setup of the optimal control problem boundary constraints, define the following,

$$\boldsymbol{\psi}_0 = \boldsymbol{x}_0(t_0) - \boldsymbol{x}_I(\tau_0, \gamma_0) = \boldsymbol{0} \tag{4.35}$$

$$\boldsymbol{\psi}_f = \boldsymbol{x}_f(t_{TD}, \boldsymbol{x}_0, \boldsymbol{\lambda}) - \boldsymbol{x}_F(\tau_f, \gamma_f) = \boldsymbol{0} \tag{4.36}$$

A continuous transfer is ensured by constraining the initial and final states along the thrust segment, i.e., $\boldsymbol{x}_0(t_0)$ and $\boldsymbol{x}_f(t_{TD}, \boldsymbol{x}_0, \boldsymbol{\lambda})$, to equal the end states corresponding to the periodic orbit coast phases, $\boldsymbol{x}_I(\tau_0, \gamma_0)$ and $\boldsymbol{x}_F(\tau_f, \gamma_f)$. A fixed thrust duration TD also constrains the transfer. If the thrust duration or a minimum mass consumption value are not specifically defined, an optimization procedure simultaneously drives TD to infinity and T to zero, producing an impractical result. Therefore, a fixed TD is typically selected *a priori*.

The VSI optimal control problem, as posed in the CR3BP, employs seven state variables, such that $n = 7$, and thus, $2(7) + 2 = 16$ boundary conditions are necessary to pose a well-defined TPBVP. The constraints in equation (4.35) and (4.36) supply 12 boundary conditions; therefore, the transversality condition, i.e., equation (4.7), is used to deliver four additional boundary conditions. Differentials of the boundary constraints are constructed consistent with the form in equation (4.5). The first derivatives of $\boldsymbol{x}_I(\tau_0, \gamma_0)$ and $\boldsymbol{x}_F(\tau_f, \gamma_f)$ with respect to the nondimensional time parameters, τ and γ, yield differential relationships that are substituted into the transversality condition producing,

$$\boldsymbol{\lambda}_{\boldsymbol{rv}_0}^T \frac{\partial \boldsymbol{x}_I(\tau_0, \gamma_0)}{\partial \tau_0} = 0 \tag{4.37}$$

$$\boldsymbol{\lambda}_{\boldsymbol{rv}_0}^T \frac{\partial \boldsymbol{x}_I(\tau_0, \gamma_0)}{\partial \gamma_0} = 0 \tag{4.38}$$

$$\boldsymbol{\lambda}_{\boldsymbol{rv}_f}^T \frac{\partial \boldsymbol{x}_F(\tau_f, \gamma_f)}{\partial \tau_f} = 0 \tag{4.39}$$

$$\boldsymbol{\lambda}_{\boldsymbol{rv}_f}^T \frac{\partial \boldsymbol{x}_F(\tau_f, \gamma_f)}{\partial \gamma_f} = 0 \tag{4.40}$$

Note that $\boldsymbol{\lambda}_{\boldsymbol{rv}_0}^T$ and $\boldsymbol{\lambda}_{\boldsymbol{rv}_f}^T$ are six element vectors composed of the initial and final position and velocity costates, respectively. When a transfer between periodic orbits does not utilize manifold arcs, equations (4.38) and (4.40) are unnecessary and are therefore, omitted as constraints. The change in $\boldsymbol{x}$ with respect to the manifold propagation parameter γ is derived by the chain rule,

$$\frac{\partial \boldsymbol{x}}{\partial \gamma} = \begin{Bmatrix} \boldsymbol{v} \\ \boldsymbol{f}_N \end{Bmatrix} \frac{\partial t}{\partial \gamma} \tag{4.41}$$

When thrust time, TD, and manifold propagation time, γ, are nondimensionalized by the same scaling factor, the relationship $\frac{\partial t}{\partial \gamma}$ is equal to unity. The change in $\boldsymbol{x}$ with respect to the periodic orbit coast parameter τ is more complex because the relationship is dependent upon γ and the manifold step-off calculation through the step-off quantity d. Senent et al. [42] derived the expression for this differential as,

$$\begin{aligned}\frac{\partial \boldsymbol{x}}{\partial \tau} &= \boldsymbol{\Phi}(\gamma, 0)\boldsymbol{\Upsilon}(\tau) \\ &= \boldsymbol{\Phi}(\gamma, 0)\left[\boldsymbol{f}(\boldsymbol{x}(\tau, 0)) \pm d[\boldsymbol{I}_6 - \hat{\boldsymbol{\nu}}\hat{\boldsymbol{\nu}}^T]\boldsymbol{A}_6(\boldsymbol{x}(\tau, 0))\hat{\boldsymbol{\nu}}(\tau)\right]\frac{dt}{d\tau} \qquad (4.42)\end{aligned}$$

where $\boldsymbol{\Phi}(\gamma, 0)$ is the STM evaluated at the end of the manifold trajectory propagation arc and $\boldsymbol{\Upsilon}(\tau)$ relates a change in τ to a change in the vector that defines the initial manifold propagation point. The $\pm$ in equation (4.42) is determined by the sign used to define the half-manifold to be used in constructing the transfer.

With the additional boundary conditions supplied by the transversality condition, the VSI optimal control problem is posed as a well-defined TPBVP. The TPBVP is solved via the single shooting method from Section 3.4. This formulation is simplified by noting that the constraint in equation (4.35) is inherently satisfied when the result of propagation for the coast times τ_0 and γ_0 is employed to deliver the initial thrust propagation point, $\boldsymbol{x}_0(t_0)$. This choice removes six constraints and allows the six design variables that define the initial point along the thrust arc to be replaced by two: τ_0 and γ_0. Similarly, the final point along the thrust arc is represented by τ_f and γ_f rather than six Cartesian coordinates. The resulting design vector contains ten design variables, i.e.,

$$\boldsymbol{X} = \left\{\begin{array}{c} \tau_0 \\ \gamma_0 \\ \boldsymbol{\lambda}_{\boldsymbol{rv}_0} \\ \tau_f \\ \gamma_f \end{array}\right\} = \boldsymbol{0} \qquad (4.43)$$

The corresponding constraint vector is also 10×1, thus, the Jacobian is square and, from Newton's method, i.e., equation (3.16), is applied to update the design variables.

$$\boldsymbol{F}(\boldsymbol{X}) = \left\{ \begin{matrix} \boldsymbol{x}_f(t_{TD}, \boldsymbol{x}_0, \boldsymbol{\lambda}) - \boldsymbol{x}_F(\tau_f, \gamma_f) \\ \boldsymbol{\lambda}^T_{\boldsymbol{rv}_0} \frac{\partial \boldsymbol{x}_I(\tau_0, \gamma_0)}{\partial \tau_0} \\ \boldsymbol{\lambda}^T_{\boldsymbol{rv}_0} \frac{\partial \boldsymbol{x}_I(\tau_0, \gamma_0)}{\partial \gamma_0} \\ \boldsymbol{\lambda}^T_{\boldsymbol{rv}_f} \frac{\partial \boldsymbol{x}_F(\tau_f, \gamma_f)}{\partial \tau_f} \\ \boldsymbol{\lambda}^T_{\boldsymbol{rv}_f} \frac{\partial \boldsymbol{x}_F(\tau_f, \gamma_f)}{\partial \gamma_f} \end{matrix} \right\} = \boldsymbol{0} \tag{4.44}$$

When a satisfactory initial guess is supplied the Newton's update equation eventually converges upon design variable values that satisfy the constraints to within a given tolerance and the resulting design variables are used to propagate the final solution, producing a continuous low-thrust VSI transfer. This method is employed to obtain continuous transfers for each of the example problems, however results are not supplied until Section 4.4.

4.3 Adjoint Control Transformation

The sensitivity of nonlinear differential equations can render the convergence and solution of a TPBVP dependent upon the initial values of the costates. Because the costates are non-physical variables, it can be challenging to determine a viable initial guess. The adjoint control transformation (ACT), originally developed by Dixon and Biggs, alleviates this problem [43] by linking the costates as formulated in the frame of the system differential equations with physical values in a spacecraft-centered frame.

Consider a spacecraft-centered frame defined such that one axis is in the direction of the instantaneous spacecraft velocity, another axis is parallel to the instantaneous angular momentum direction, and the third axis completes the dextral orthonormal set, i.e.,

$$\widehat{\boldsymbol{V}} = \frac{\boldsymbol{v}}{|\boldsymbol{v}|}, \quad \hat{\boldsymbol{h}} = \frac{\boldsymbol{r} \times \boldsymbol{v}}{|\boldsymbol{r} \times \boldsymbol{v}|}, \quad \hat{\boldsymbol{b}} = \hat{\boldsymbol{h}} \times \widehat{\boldsymbol{V}} \tag{4.45}$$

The chain rule is applied to evaluate the derivatives of these quantities,

$$\dot{\widehat{\boldsymbol{V}}} = \frac{\dot{\boldsymbol{v}}}{|\boldsymbol{v}|} - \frac{\boldsymbol{v}\dot{v}}{|\boldsymbol{v}|^2}, \quad \dot{\hat{\boldsymbol{h}}} = \frac{\dot{\boldsymbol{h}}}{|\boldsymbol{h}|} - \frac{\boldsymbol{h}\dot{h}}{|\boldsymbol{h}|^2}, \quad \dot{\hat{\boldsymbol{b}}} = \dot{\hat{\boldsymbol{h}}} \times \widehat{\boldsymbol{V}} + \hat{\boldsymbol{h}} \times \dot{\widehat{\boldsymbol{V}}} \tag{4.46}$$

where the scalar derivatives $\dot{v}$ and $\dot{h}$ are defined,

$$\dot{v} = \boldsymbol{v} \cdot \frac{\dot{\boldsymbol{v}}}{|\boldsymbol{v}|} \quad \text{and} \quad \dot{h} = \boldsymbol{h} \cdot \frac{\dot{\boldsymbol{h}}}{|\boldsymbol{h}|} \tag{4.47}$$

Therefore, equations (4.46) and (4.47) offer the rate of change of the spacecraft-centered frame with respect to time. The spacecraft-centered frame defined by the unit vectors $\hat{\boldsymbol{V}}$,$\hat{\boldsymbol{h}}$, and $\hat{\boldsymbol{b}}$ is denoted the velocity frame, V. The thrust pointing vector expressed in the velocity frame, $\hat{\boldsymbol{u}}_{T_V}$, is oriented by the spherical angles α and β as depicted in Figure 4.3. The expression for $\boldsymbol{u}_{T_V}$ in terms of these angles is,

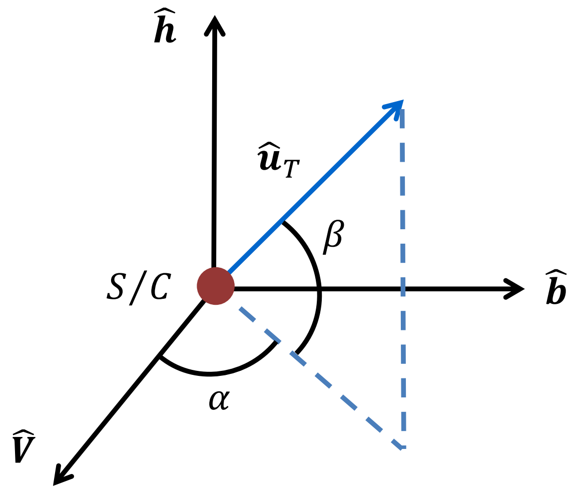

$\widehat{\boldsymbol{h}}$
$\widehat{\boldsymbol{u}}_T$
β
S/C
$\widehat{\boldsymbol{b}}$
α
$\widehat{\boldsymbol{V}}$

Figure 4.3.: Adjoint Control Transformation Velocity Frame

$$\hat{\boldsymbol{u}}_{T_V} = \begin{pmatrix} \cos\alpha\cos\beta \\ \sin\alpha\cos\beta \\ \sin\beta \end{pmatrix} \tag{4.48}$$

The derivative of equation (4.48) is then the rate of change of the thrust pointing vector as viewed in the velocity frame,

$$\dot{\hat{\boldsymbol{u}}}_{T_V} = \begin{pmatrix} -\dot{\alpha}\sin\alpha\cos\beta - \dot{\beta}\cos\alpha\sin\beta \\ -\dot{\alpha}\cos\alpha\cos\beta - \dot{\beta}\sin\alpha\sin\beta \\ \dot{\beta}\cos\beta \end{pmatrix} \tag{4.49}$$

The Cartesian coordinates that are defined in the CR3BP are typically expressed in terms of the rotating frame, therefore, a direction cosine matrix is required to transform the thrust pointing vector and its derivative from the velocity to the rotating frame,

$$\hat{\boldsymbol{u}}_{T_R} = \boldsymbol{D}\hat{\boldsymbol{u}}_{T_V} \tag{4.50}$$

$$\hat{\boldsymbol{u}}_{T_R} = \dot{\boldsymbol{D}}\hat{\boldsymbol{u}}_{T_V} + \boldsymbol{D}\dot{\hat{\boldsymbol{u}}}_{T_V} \tag{4.51}$$

The direction cosine matrix, $\boldsymbol{D}$, and its associated derivative are defined,

$$\boldsymbol{D} = \begin{bmatrix} \hat{x}\cdot\widehat{V} & \hat{x}\cdot\hat{b} & \hat{x}\cdot\hat{n} \\ \hat{y}\cdot\widehat{V} & \hat{y}\cdot\hat{b} & \hat{y}\cdot\hat{n} \\ \hat{z}\cdot\widehat{V} & \hat{z}\cdot\hat{b} & \hat{z}\cdot\hat{n} \end{bmatrix} \tag{4.52}$$

$$\dot{\boldsymbol{D}} = \begin{bmatrix} \hat{x}\cdot\dot{\widehat{V}} & \hat{x}\cdot\dot{\hat{b}} & \hat{x}\cdot\dot{\hat{n}} \\ \hat{y}\cdot\dot{\widehat{V}} & \hat{y}\cdot\dot{\hat{b}} & \hat{y}\cdot\dot{\hat{n}} \\ \hat{z}\cdot\dot{\widehat{V}} & \hat{z}\cdot\dot{\hat{b}} & \hat{z}\cdot\dot{\hat{n}} \end{bmatrix} \tag{4.53}$$

The spherical angles as defined in the velocity frame in Figure 4.3 are related to the velocity costate values by the definition of the primer vector in equation (4.18), that is,

$$\boldsymbol{\lambda_v} = \lambda_v \boldsymbol{u}_{T_R} \tag{4.54}$$

The time derivative of equation (4.54) is required to relate the spherical angles to the position costates,

$$\dot{\boldsymbol{\lambda}}_v = \dot{\lambda}_v \boldsymbol{u}_{T_R} + \lambda_v \dot{\boldsymbol{u}}_{T_R} \tag{4.55}$$

To relate the position costates to the spherical angles, the differential equation for the velocity costates in equation (4.26) is solved for $\boldsymbol{\lambda}_r$ and equation (4.55) is substituted into the resulting expression.

$$\boldsymbol{\lambda_r} = -\dot{\lambda}_v \boldsymbol{u}_{T_R} - \lambda_v \dot{\boldsymbol{u}}_{T_R} - \frac{\partial \boldsymbol{f}_N}{\partial \boldsymbol{v}} \boldsymbol{\lambda_v} \tag{4.56}$$

An expression for the magnitude of the rate of change of the velocity costate is obtained by assuming that the initial value of the Hamiltonian is equal to zero.

$$\dot{\lambda}_v = -\frac{1}{\boldsymbol{u}_{T_R}^T \boldsymbol{v}} \left[\lambda_v \dot{\boldsymbol{u}}_{T_R}^T \boldsymbol{v} + \boldsymbol{\lambda_v}^T \frac{\partial \boldsymbol{f}_N}{\partial \boldsymbol{v}} \boldsymbol{v} - \boldsymbol{\lambda_v}^T \boldsymbol{f}_N \right] \tag{4.57}$$

Equations (4.54) and (4.56) relate the position and velocity costate vectors to the values and rates of change of the spherical angles and the magnitude of the velocity costate vector. These quantities then replace the original six costates as design variables in the shooting formulation of the optimal VSI control problem, i.e., the new design variable vector becomes

$$\boldsymbol{X}_{ACT} = \left\{ \begin{array}{c} \tau_0 \\ \gamma_0 \\ \alpha_0 \\ \dot{\alpha}_0 \\ \beta_0 \\ \dot{\beta}_0 \\ \lambda_{v0} \\ \dot{\lambda}_{v0} \\ \gamma_f \\ \tau_f \end{array} \right\} \tag{4.58}$$

Employing the ACT, the determination of an initial guess for the design variables is linked to the designer's intuition concerning the initial magnitude and direction of the thrust vector in the velocity frame.

4.4 Example Problems:

The framework for sample optimal VSI problems are detailed in Section 4.2.1. A simple circular-to-circular orbit transfer demonstrates fundamental behaviors in the VSI optimal control problem, while more complex transfers leverage invariant manifolds and demonstrate the versatility of the indirect optimization approach. The transfer path is constructed and the result appears in configuration space below.

4.4.1 Circular Orbit Transfer

An orbit raising maneuver, from an initial parking orbit to a desired final circular orbit is a common application of the low-thrust circular-to-circular orbit transfer problem. The scenario here is a transfer from an initial low Earth orbit (LEO) to a geosynchronous Earth orbit (GEO). These orbits are both well within Earth's sphere of influence, thus, the dynamical model includes only Earth's gravitational field. Parameters and initial conditions are included in Table 4.1. A transfer time of 75 days is selected consistent with Seywald et al. in an investigation of the efficiency of VSI for coplanar circular orbit transfers given fixed transfer times ranging from 50 to 100 days [44]. The VSI transfer involves continuous thrusting, thus, the total transfer time is equivalent to the thrust duration (TD).

Table 4.1.: LEO to GEO Circular-to-Circular Orbit Transfer Parameters

Parameter	Value	Units
Earth Gravitational Parameter	3.986004418×10^5	$\frac{km^3}{s^2}$
Initial Orbital Radius	6671	km
Final Orbital Radius	42164	km
Earth Gravitational Acceleration	9.80665	$\frac{m}{s^2}$
Initial Spacecraft Weight	500	kg
Maximum Spacecraft Power	2000	W

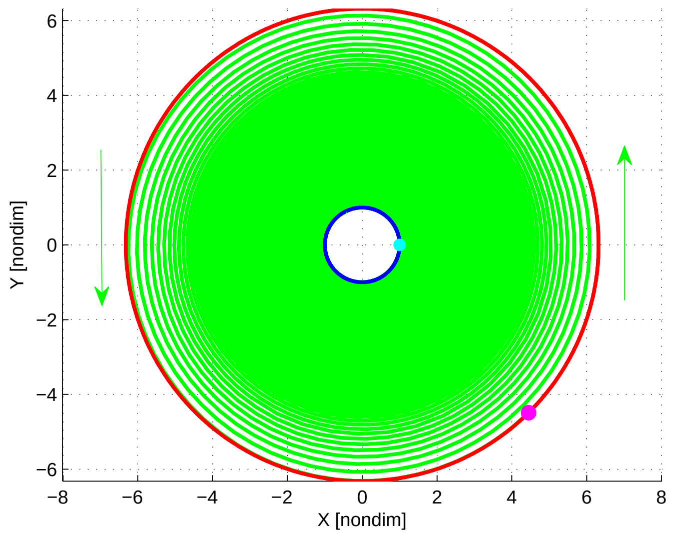


Figure 4.4.: Orbit Transfer from Initial State (cyan) on LEO (blue) to Final State (magenta) on GEO (red) for $TD = 75$ days

A single shooting strategy is applied in the problem formulation as detailed in Section 4.2.2 and the algorithm converges to a local optimal low-thrust transfer, as plotted in Figure 4.4. This transfer requires 483 revolutions about Earth; over the interval, the thrust pointing vector is aligned nearly precisely in the anti-velocity direction as illustrated in Figure 4.5. The converged or final values for the design variable vector representing the optimal solution appear in Table 4.2. Optimal solutions for longer or shorter transfer times can be constructed using a continuation process originating with the solution for $TD = 75$ days. Convergence to the optimal solution is extremely sensitive to the initial costate values, thus, natural parameter continuation proved too coarse as a primary methodology, even with a small step size. A more accurate continuation approach, e.g., pseudo-arc length continuation, is necessary to successfully employ continuation with respect to the transfer time.

The initial costate values in Table 4.2, along with the known initial orbit conditions, are used to propagate the circular-to-circular orbit transfer. The final spacecraft mass is $m_f = 352.6081\ kg$, thus, roughly 147 kg of propellant is consumed over the

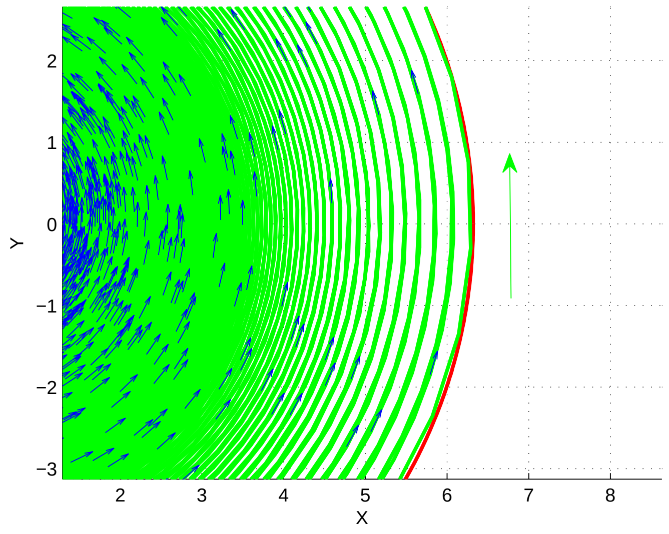


Figure 4.5.: Circular Orbit Transfer, with $\hat{\boldsymbol{u}}_T$

Table 4.2.: Solution to TPBVP for Circular Orbit Transfer

Parameter	Solution
λ_{r_0}	0.69027033
λ_{θ_0}	0
λ_{p_0}	−0.00085971
λ_{q_0}	0.69109078
λ_{m_0}	0.49733000

course of the transfer. The plot for the total mass as a function of time is plotted in Figure 4.6 and appears to decrease linearly, however, close examination reveals oscillations about this linear approximation with a period consistent with the orbital revolutions of the spacecraft. All of the time responses plotted in Figure 4.6 display similar oscillations, including thrust and specific impulse that are inversely proportional, as indicated by equation (4.15). The difference in the Hamiltonian with respect to its initial value also appears plotted because it serves to assess the numerical accuracy of the propagation. The total variations in the value of the Hamiltonian in Figure 4.6 are consistent with the order of machine precision, thus, the numerical integration approach to propagate the other quantities is assumed to be reasonably accurate.

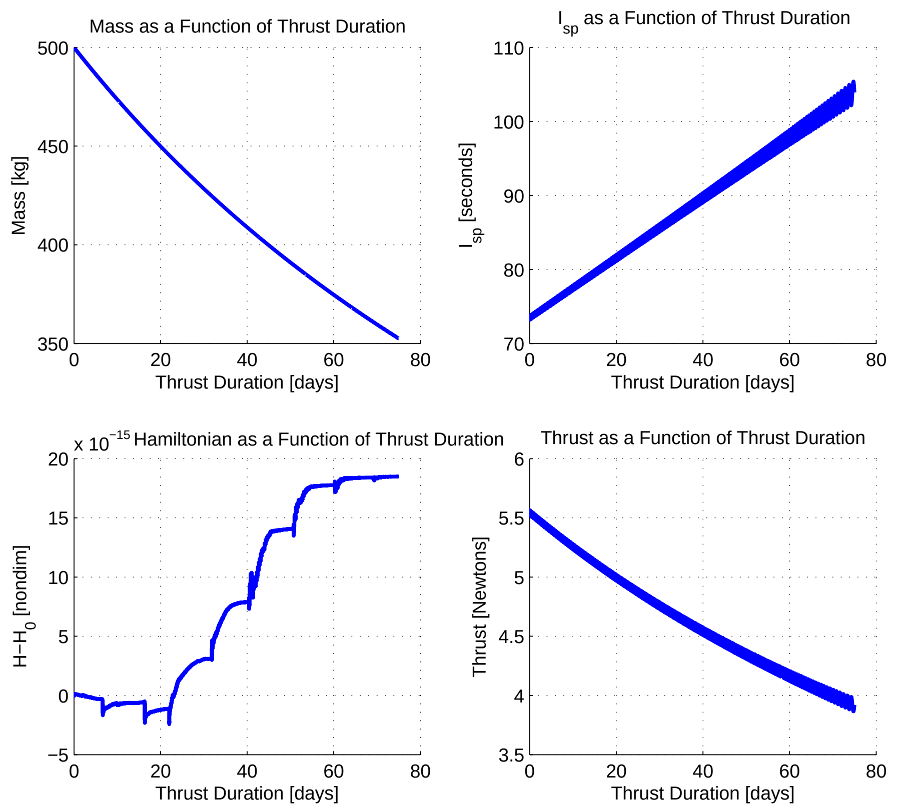


Figure 4.6.: Key Parameters for Circular-to-Circular Orbit Transfer

The velocity costates are related to the thrust pointing vector by equation (4.18) therefore, the behavior of this vector in the spacecraft centered velocity frame is determined by observing the costates trends. Both components of $\boldsymbol{\lambda}_v$ appear constant in Figure 4.7, although closer examination reveals low amplitude oscillations in both values. The mean values of $\boldsymbol{\lambda}_v$ indicate that the thrust pointing vector only has a significant component in the tangential direction of the spacecraft centered velocity frame. Therefore, in the inertial frame the thrust vector constantly points in the

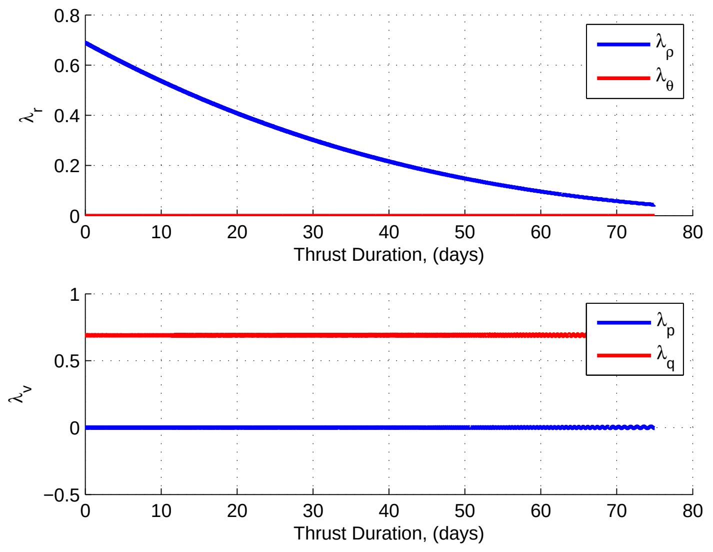


Figure 4.7.: Costates for Circular Orbit Transfer

positive tangential direction of the circular motion, exactly the behavior observed in Figure 4.5. The consistent oscillatory nature of the costates and transfer spiral of the low-thrust circular orbit transfer lend themselves to the development of analytical approximations and control laws for these types of transfers [45]. More complex transfers incorporating additional dimensions and forces yield transfers less amenable to description by analytical methods.

4.4.2 Halo to Halo Transfers

Halo to halo orbit transfers add complexity to the optimal control problem by introducing additional state variables and parameters. While circular orbit transfers were kept planar, halo to halo orbit transfers possess significant out-of-plane components. The manifolds of halo orbits can offer efficient transfer solutions, but it is not always beneficial to use these structures, especially when the destination orbit is centered about the same Lagrange point. Whether or not manifold arcs are employed the approach for obtaining halo to halo orbit transfers is applicable to a broad range

Table 4.3.: Halo to Halo Orbit Transfer Parameters

Parameter	Value	Units
Earth Mass	5.97200×10^{24}	kg
Lunar Mass	7.34600×10^{22}	kg
Mass Parameter	0.01215	NA
Earth Moon Distance	384400.00	km
Characteristic Time	375208.35	seconds
Initial Spacecraft Mass	500	kg
Maximum Spacecraft Power	2000	W

of similar transfer scenarios. These scenarios, for example transfer from an Earth parking orbit to a halo orbit [46], often utilize phasing parameters analogous to the nondimensional time parameters τ_i and α_i. The two halo to halo transfer problems chosen for this study closely follow those originally examined by Stuart [47]. The CR3BP model is used to determine these transfers and standard parameters for this model are provided in Table 4.3.

Halo to Halo Transfer Without Manifolds

A transfer between two nearby northern L_1 halo orbits is not significantly aided by employing a manifold because both orbits are centered around the same equilibrium point. Therefore, the two periodic orbits are connected only by a transfer arc which occurs over a fixed thrust duration, $TD = 2.388364$ days. Because manifolds are not leveraged, the parameters γ_0 and γ_f are omitted from the single shooting algorithm used to obtain the optimal transfer. Initial conditions for the selected Northern halo orbits are provided in Table 4.4.

The single shooting algorithm converges upon a local optimal low-thrust transfer for a thrust duration, $TD = 2.388364$ days, shown in Figure 4.8. Euler-Lagrange

Table 4.4.: Initial Conditions for Halo Orbits in No Manifold Case

Parameter	Initial Orbit	Final Orbit
x_0 (km)	316625.9094	318038.1661
z_0 (km)	17304.8239	36521.8311
$\dot{y}_0$ (km/s)	0.1582	0.2153
Jacobi Constant	3.1577	3.1091
Period (days)	11.9711	12.0892

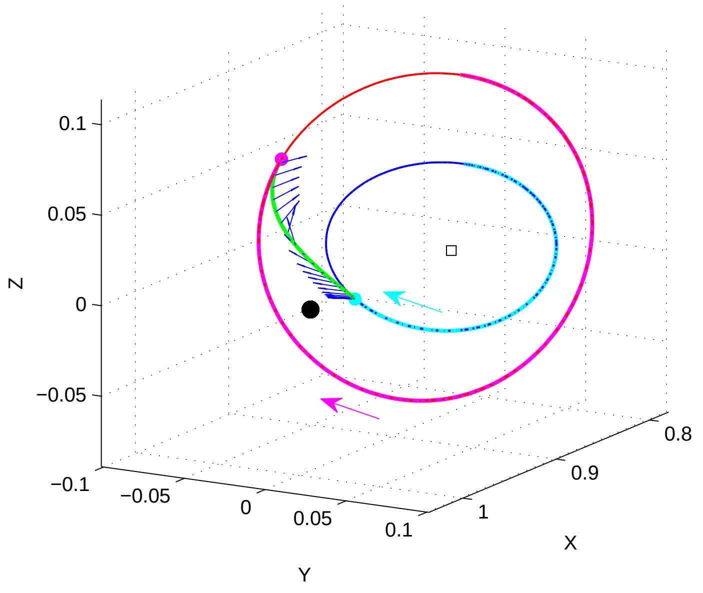


Figure 4.8.: Transfer (green) from Initial L_1 Halo (blue) to Final L_1 Halo (red) without Manifold, Including Thrust Pointing Vectors (blue)

theory does not guarantee a solution to the TPBVP is a global optimal. Therefore, other more optimal transfers may exist for the chosen thrust duration and halo orbits, this is demonstrated by Stuart [47]. Stuart also located optimal solutions for other thrust durations by performing continuation with respect to the thrust duration. The continuation process results in a family of locally optimal transfers for a range of thrust durations. The sensitivity inherent in longer thrust arcs eventually requires that a multiple shooting form of the original single shooting algorithm be used to obtain convergence.

Table 4.5.: Solution to TPBVP for No Manifold Case

Parameter	Solution
τ_0	1.66824171
λ_{rx_0}	1.31297242
λ_{ry_0}	-1.22244717
λ_{rz_0}	-0.25358164
λ_{vx_0}	0.40087507
λ_{vy_0}	-0.31836465
λ_{vz_0}	0.070576740
τ_f	2.19435230

The design variable vectors converged upon for the optimal solution are provided in Table 4.5, where the coast times along each periodic orbit, τ_0 and τ_f, are measured from from the northern crossing of the $X - Z$ plane. This solution to the optimal control problem is used to propagate the thrust arc for the chosen thrust duration. The spacecraft mass at the end of the thrust arc propagation is $m_f = 480.6426\ kg$, thus roughly 20 kg of fuel is consumed over the course of the transfer. Figure 4.9 shows that mass is consumed most quickly at the beginning and end of the transfer, this trend corresponds with the relatively high thrust values used at those times. As indicated in equation (4.14), the thrust trend line is the inverse of the specific impulse trend; the maximum specific impulse occurs at the midpoint of the thrust duration when thrust is at a minimum. The difference in the Hamiltonian with respect to its initial value is also plotted as a function of thrust duration because it serves as a check of numerical accuracy. The variations of the Hamiltonian shown in Figure 4.9

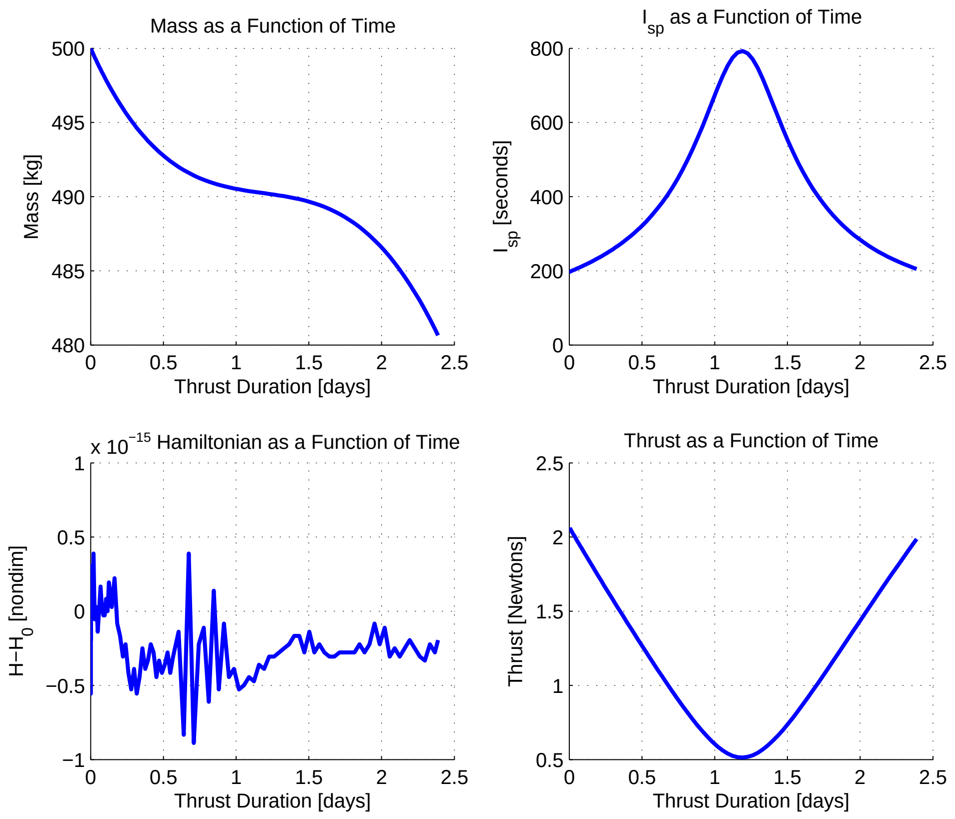


Figure 4.9.: Key Parameters for L_1 Halo to L_1 Halo Transfer Without Manifold

are close to machine precision, thus the propagation used to obtain the other trends is once again assumed accurate.

The velocity costates are related to the thrust pointing vector by equation (4.18) and all of the costates are related to physical angles by the ACT scheme, thus, useful insight is gained from noting the costates trends. Figure 4.10 shows that over the duration of the transfer λ_{vx} decreases while λ_{vy} increases. This trend is consistent with the thrust pointing vectors plotted in Figure 4.8 which switch direction with respect to the x and y axes while remaining relatively constant in z.

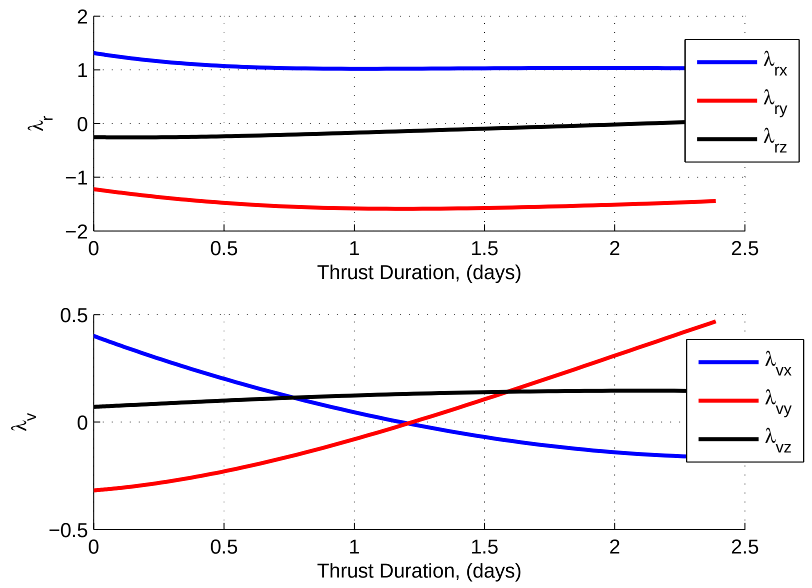


Figure 4.10.: Costate Trends for L_1 Halo to L_1 Halo Transfer Without Manifold

Halo to Halo Transfer With Manifolds

The fuel efficiency of a transfer between periodic orbits surrounding different equilibrium points is increased by leveraging manifold structures. An unstable manifold is used to depart the initial orbit while a stable manifold assists insertion into the final orbit. The coasting period spent on each manifold is governed by the nondimensional time parameters γ_0 and γ_f, now included in the single shooting algorithm. Initial conditions for the selected Northern halo orbits are provided in Table 4.6. Note, that the Jacobi constant values supplied for the selected halo orbits are identical up to four digits. Initial conditions for halo orbits corresponding to the provided Jacobi constant values are targeted, however the halo orbits produced by the targeting scheme possess slightly different Jacobi constant values when more decimal places are shown. If this difference were not present then a heteroclinic connection between the two orbits would be available.

Table 4.6.: Initial Conditions for Halo Orbits in Manifold Case

Parameter	Initial Orbit	Final Orbit
x_0 (km)	318014.12	449363.6
z_0 (km)	36287.36	37632.76
$\dot{y}_0$ (km/s)	0.2146	-0.1994
Jacobi Constant	3.1149	3.1149
Period (days)	12.0806	14.4809

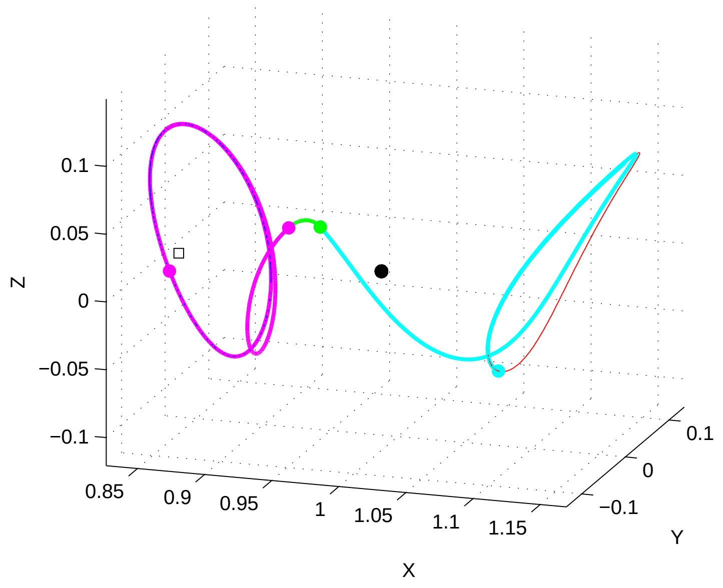


Figure 4.11.: Transfer (green) from Initial L_1 Halo (blue) to Final L_2 Halo (red) Employing Unstable (Magenta) and Stable (Cyan) Manifolds

The single shooting algorithm converges upon a local optimal low-thrust transfer for a thrust duration of $TD = .651372$ days, shown in Figure 4.11. Once again, continuation can be used to obtain a family of locally optimal transfers for a range of thrust durations. The design variable vectors for the optimal solution, provided in Table 4.7, now include the coast times along each manifold, γ_0 and γ_f, where γ_f is

Table 4.7.: Solution to TPBVP for Manifold Case

Parameter	Solution
τ_0	2.05210772
α_0	2.85602244
λ_{rx_0}	0.12807031
λ_{ry_0}	-0.38761577
λ_{rz_0}	1.38237663
λ_{vx_0}	0.22623444
λ_{vy_0}	-1.18940955
λ_{vz_0}	-0.63957151
α_f	-3.80619756
τ_f	1.47574058

negative because the stable manifold is propagated in reverse time. This solution to the optimal control problem is used to propagate the thrust arc for the chosen thrust duration.

The spacecraft mass at the end of the thrust arc propagation is $m_f = 415.5340\ kg$, thus roughly 85 kg of fuel is consumed over the course of the transfer. Figure 4.12 illustrates that the spacecraft mass decreases in a nearly linear manner over the thrust duration. The thrust magnitude, also presented in Figure 4.12 follows a similar trend while the specific impulse reflects the reverse trend. The I_{sp} values obtained for this transfer are relatively low compared to current low-thrust engines, however transfers with greater I_{sp} and lower T values can be obtained by extending the thrust duration via continuation. The variations in the Hamiltonian with respect to H_0 in Figure 4.9 are once again close to machine precision, therefore, the transfer propagation is assumed accurate.

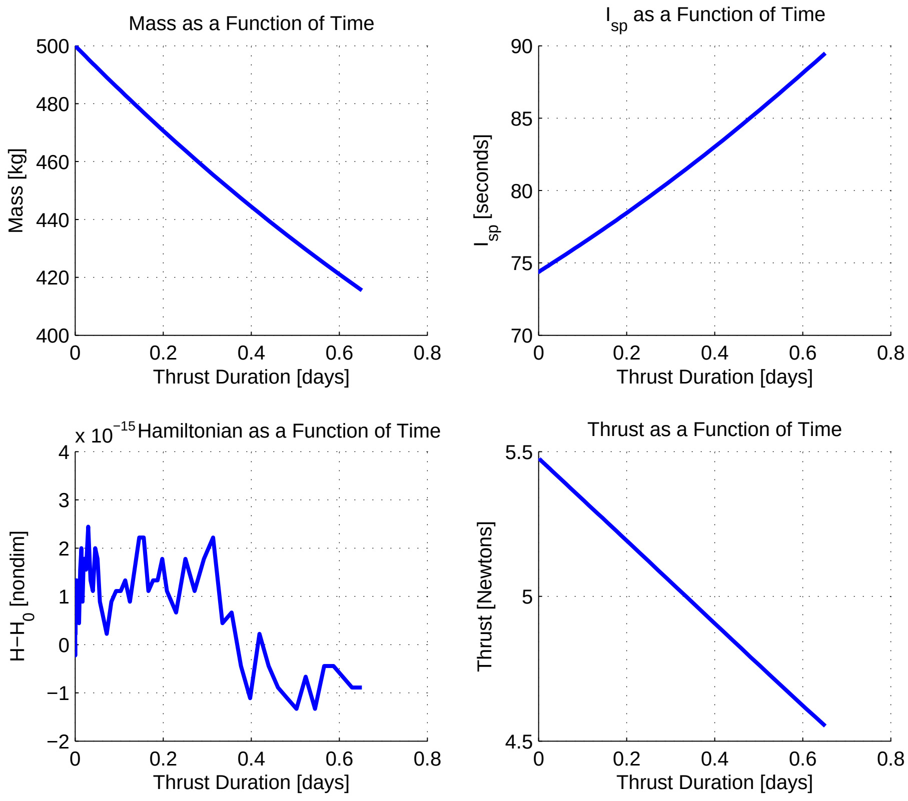


Figure 4.12.: Key Parameters for L_1 Halo to L_2 Halo Transfer Employing Manifolds

The velocity costates remain relatively constant throughout the thrust duration, as shown in Figure 4.13. The λ_{vx} costate possesses the largest slope and switches sign during the thrust duration. The relatively steady thrust pointing is reflected in the unit vectors plotted in Figure 4.14 which all point in the negative y directions.

Euler-Lagrange theory provides a general method that, applied to the low-thrust VSI optimal control problem, yields a formulation applicable to several transfer scenarios that exhibit unique features such as spiraling or coasting periods. The costates introduced by the Euler-Lagrange theory provide useful control laws for the spacecraft, that allow the results of the TPBVP to be propagated. However the nonphysical values of the costates make determining an initial guess challenging, although this is alleviated somewhat by the adjoint control transformation. Additionally, the

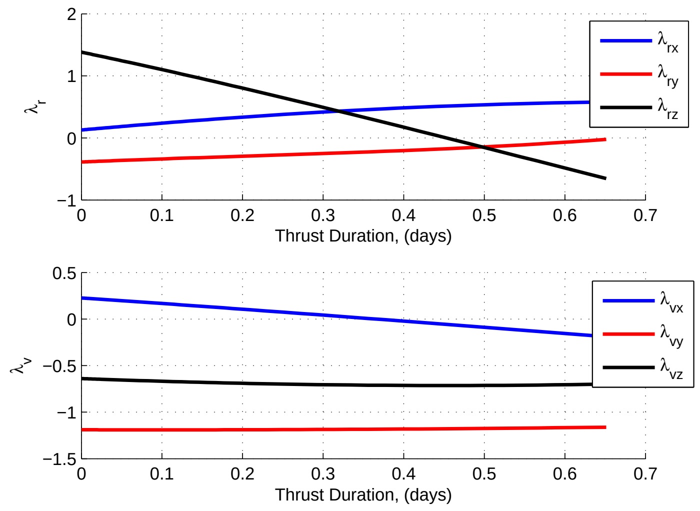


Figure 4.13.: Costate Trends for L_1 Halo to L_2 Halo Transfer Employing Manifolds

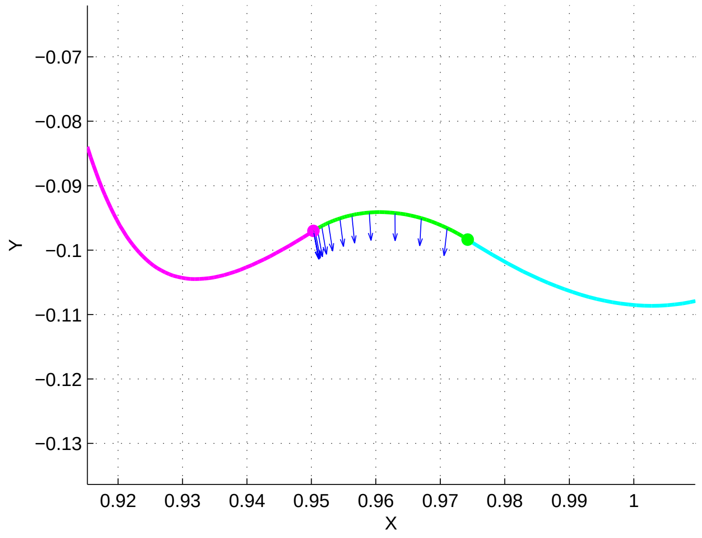


Figure 4.14.: Halo to Halo Orbit Transfer Employing Manifolds with Thrust Vector

transversality condition was used to pose a well-defined TPBVP, however the resulting boundary conditions had to be rederived for each type of transfer examined. While solution to the TPBVP formulated using Euler-Lagrange theory is guaranteed to be a local optimal, changes in the problem require siginificant rederivation. The rigidity of the TPBVP problem formulation and the sensitivity of the non-physical costates are incentives to explore other methods for solving optimal control problems.

5. DIRECT OPTIMIZATION WITH COLLOCATION

The indirect optimization approach from Chapter 4 solves the optimal control problem by transforming it into a TPBVP using the Euler-Lagrange necessary conditions and the transversality conditions; the solution of this problem is then guaranteed to be locally optimal. A cost function is used to setup the TPBVP, but direct operation on the the cost function does not occur in solving the problem. In contrast, direct optimization methods employ the cost function directly during the computation of an optimal solution, typically constructing the gradient of the cost function and ensuring that each iteration shifts the solution in a direction that minimizes the norm of the gradient. Many direct optimization approaches discretize an optimal control problem, thus transforming it into a parameter optimization problem that can be solved using Nonlinear Programming (NLP) methods [48]. Direct optimization schemes are typically categorized based on the strategy to discretize the optimal control problem and the type of NLP method. The discretization process is termed direct transcription; the size and accuracy of the resulting discretization varies with the particular transcription scheme [49]. The NLP problem formulated using direct transcription often involves hundreds or thousands of individual variables. Therefore, NLP algorithms capable of accommodating large, sparse problems, for example, interior-point methods, are employed to solve direct optimization problems parameterized via direct transcription.

5.1 General Optimal Trajectory Design Problem

Stating the optimal trajectory design problem in its most general form reflects the application of direct optimization methods to obtain a solution. A trajectory design problem begins, of course, with a set of differential equations that govern the the system dynamics,

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\kappa}, t\right) \tag{5.1}$$

where the dynamics are a function of n_s variables $\boldsymbol{x}$ in the state vector, n_u control variables defined as the elements of vector $\boldsymbol{u}$, and any parameter values included in the vector $\boldsymbol{\kappa}$ that are not dependent on time t. Initial and final conditions consistent with the system equations are defined at times t_0 and t_f, i.e.,

$$\boldsymbol{\psi}_{0l} \leq \boldsymbol{\psi}\left(\boldsymbol{x}(t_0), \boldsymbol{u}(t_0), \boldsymbol{\kappa}, t_0\right) \leq \boldsymbol{\psi}_{0u} \tag{5.2}$$

$$\boldsymbol{\psi}_{fl} \leq \boldsymbol{\psi}\left(\boldsymbol{x}(t_f), \boldsymbol{u}(t_f), \boldsymbol{\kappa}, t_f\right) \leq \boldsymbol{\psi}_{fu} \tag{5.3}$$

A trajectory or trajectory arc is a solution to the set of dynamic equations that govern motion in the system. This solution or the dynamical flow, is also subject to path constraints that may apply only at specific times, or along the entire path.

$$\boldsymbol{g}_l \leq \boldsymbol{g}\left(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{\kappa}, t\right) \leq \boldsymbol{g}_u \tag{5.4}$$

Additionally, the state and control variables corresponding to a particular solution may be limited by upper and lower bounds,

$$\boldsymbol{x}_l \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_u \tag{5.5}$$

$$\boldsymbol{u}_l \leq \boldsymbol{u}(t) \leq \boldsymbol{u}_u \tag{5.6}$$

The goal of an optimal control problem is the determination of the control variables and parameter values that extremize a scalar cost value, J, while also delivering a path that is consistent with the problem dynamics and satisfies the constraints. The equation for the cost value, J, is termed the objective function and, expressed in the Mayer form, is written,

$$J = \phi(\boldsymbol{x}(t_f), t_f) \tag{5.7}$$

Note, additional variables or parameters can be included in the objective function but, in this representation, J is dependent upon the final states and time. The general optimal trajectory design problem is continuous in time, therefore, an obvious solution strategy is the application of Euler-Lagrange theory which transforms continuous optimization problems into TPBVPs.

This approach is represented by the indirect optimization scheme detailed in Chapter 4. Direct optimization methods offer an alternate solution strategy, however, it is required that the continuous optimal trajectory design problem be discretized.

5.2 The Nonlinear Programming Problem

The field of mathematical optimization encompasses a diverse array of optimization problems along with methods for solving these problems; an excellent overview of this discipline is offered by Bertsekas [50] as well as Boyd and Vandenberghe [51]. The nonlinear programming (NLP) problem is one of the most basic types of mathematical optimization problems and many direct optimization methods use NLP to solve a parameter optimization problem. The goal of a nonlinear programming (NLP) problem within the larger field of mathematical optimization, is the determination of the n variable vector $\boldsymbol{x}$ that minimizes,

$$\min_{\boldsymbol{x}} E(\boldsymbol{x}) \tag{5.8}$$

subject to the m constraints,

$$\boldsymbol{c}_l \leq \boldsymbol{c}(\boldsymbol{x}) \leq \boldsymbol{c}_u \tag{5.9}$$

with bounds,

$$\boldsymbol{x}_l \leq \boldsymbol{x} \leq \boldsymbol{x}_u \tag{5.10}$$

The constraints in equation (5.9) include equality and inequality constraints, where the equality constraints, $\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0}$, appear when $\boldsymbol{c}_L = \boldsymbol{c}_U$. Note, the bounds in equation (5.10) differ from equation (5.5) because, in the former, the variables are no longer continuous in time. The Lagrangian is introduced to secure a solution to the NLP, and is a scalar value determined from,

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = E(\boldsymbol{x}) - \boldsymbol{\lambda}^T \boldsymbol{c}(\boldsymbol{x}) \tag{5.11}$$

Note, though the Lagrangian in the NLP is similar in form to the Hamiltonian, equation (4.2), the Lagrange multipliers, $\boldsymbol{\lambda}$, are not necessarily equivalent to the costates that are familiar in indirect optimization. The partial derivatives of the Lagrangian with respect to the variables in $\boldsymbol{x}$ and the Lagrange multipliers $\boldsymbol{\lambda}$ supply the necessary conditions for solving the NLP,

$$\nabla_x L(\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{g}(\boldsymbol{x}) - \boldsymbol{G}^T(\boldsymbol{x})\boldsymbol{\lambda} = \boldsymbol{0} \tag{5.12}$$

$$\nabla_\lambda L(\boldsymbol{x}, \boldsymbol{\lambda}) = -\boldsymbol{c}(\boldsymbol{x}) = \boldsymbol{0} \tag{5.13}$$

where $\boldsymbol{g} = \nabla_x E$ is the gradient of the objective function, and $\boldsymbol{G}$ is the Jacobian of the equality constraint vector. The system comprised of equations (5.12) and (5.13) is solved using a Newton method, where the linear system for computing the update vector, $\{\Delta\boldsymbol{x},\ \Delta\boldsymbol{\lambda}\}$ is,

$$\begin{bmatrix} \boldsymbol{H}_L & -\boldsymbol{G}^T \\ \boldsymbol{G} & 0 \end{bmatrix} \begin{Bmatrix} \Delta\boldsymbol{x} \\ \Delta\boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} -\boldsymbol{g} \\ -\boldsymbol{c} \end{Bmatrix} \tag{5.14}$$

The matrix $\boldsymbol{H}_L$ in equation (5.14) is denoted the Hessian and is a matrix of the second derivatives of the equality constraints constructed by,

$$\boldsymbol{H}_L = \nabla_x^2 E - \sum_{i=1}^{m} \lambda_i \nabla_x^2 c_i \tag{5.15}$$

The system of equations (5.14) is denoted a Karush-Kunh-Tucker (KKT) system, because its formation is based on the well known Karush-Kuhn-Tucker conditions. The general optimal trajectory design problem is discretized to apply the strategy for solving the NLP that is supplied by the KKT conditions. The discretization process is frequently denoted direct transcription and collocation methods offer one approach for implementing this process. The surveys by Betts ([52] and [53]) provide an overview of the general optimal trajectory, the NLP, and the application of both concepts to direct transcription. Finally, note that solving the NLP via the KKT conditions is one of a variety of approaches for solving a mathematical optimization problem. Other solution strategies fall under the categories of feasible direction methods, Lagrangian methods, and penalty function methods, as outlined by Bertsekas [50]. However, not all of these methods are well-suited for solving the NLP especially the NLP formulated from the trajectory optimization problem.

5.3 Collocation

Collocation is a method for implicitly integrating differential equations, one that is frequently employed to transcribe continuous optimal control problems into NLP problems. The implicit integration approach fits piecewise polynomials into a discretization framework in a system governed by a set of ordinary differential equations. In astrodynamics, a discretization is commonly comprised of discrete points in time, i.e.,

$$\Pi : t_0 < t_1 < \ldots < t_n = t_f \tag{5.16}$$

where Π is a mesh comprised of n mesh points. Each mesh point, $\boldsymbol{Z}_i$, may have associated state and control information, i.e.,

$$\boldsymbol{Z} = \{\boldsymbol{x}_1, \boldsymbol{u}_1, \ldots, \boldsymbol{x}_n, \boldsymbol{u}_n\}^T \tag{5.17}$$

The n mesh points define $n-1$ segments where the time step along each segment is defined as $\Delta t_i = t_{i+1} - t_i$. For numerical convenience, the time along each segment is typically normalized such that the time interval along a segment is $[-1,\ 1]$. The conversion to normalized time is then given by,

$$\boldsymbol{\tau} = \frac{2}{t_{i+1} - t_i}(t - t_i) - 1 \tag{5.18}$$

where τ represent the normalized time on segment i that is equivalent to the non-normalized time, t. Computations relevant to a specific segment that are accomplished on a normalized time interval, improves the scaling and computational efficiency. The general collocation framework follows a scheme developed by Ozimek et al. [54] and refined by Grebow and Pavlak [55].

The fidelity of a collocation scheme is dependent upon the accuracy of the approximation of the solution to a set of differential equations. The simplest collocation approach utilizes Euler's integration rule, i.e.,

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \Delta t_i \boldsymbol{f}(t_i, \boldsymbol{x}_i, \boldsymbol{u}_i) \tag{5.19}$$

where $\boldsymbol{f}(t_i, \boldsymbol{x}_i, \boldsymbol{u}_i)$ represents the system dynamical equations. Equation (5.19) uses the vector field information, or "slope", at mesh point i to predict the states at mesh point $i+1$. The discrepancy between the states predicted by Euler's rule and the actual states at $i+1$ is defined as the defect.

$$\boldsymbol{\Delta}_i = \boldsymbol{x}_i - \boldsymbol{x}_{i+1} + \Delta t_i \boldsymbol{f}(t_i, \boldsymbol{x}_i, \boldsymbol{u}_i) = \boldsymbol{0} \tag{5.20}$$

The collocation problem is solved when all defects equal zero to within a predefined tolerance, thus ensuring state continuity between adjacent segments with a mesh that abides by the system dynamics. The Euler rule method of collocation is depicted in Figure 5.16. Euler's rule provides a simple demonstration of the collocation approach, however, it only approximates a solution to a set of differential equations with first order accuracy, $O(h)$. The low accuracy associated with Euler's method is mitigated by employing shorter discretization segments, but this approach requires a large number of segments to produce an accurate solution. Alternatively, higher order methods typically allow a reduction in the number of segments necessary to accurately solve a collocation problem. A method based on the trapezoidal rule, illustrated in Figure 5.2, supplies an additional order of accuracy over Euler's rule, $O(h^2)$. The defect equation based on the trapezoidal rule is,

$$\boldsymbol{\Delta}_i = \boldsymbol{x}_i - \boldsymbol{x}_{i+1} + \frac{\Delta t_i}{2}\{\boldsymbol{f}(t_i, \boldsymbol{x}_i, \boldsymbol{u}_i) + \boldsymbol{f}(t_{i+1}, \boldsymbol{x}_{i+1}, \boldsymbol{u}_{i+1})\} = \boldsymbol{0} \tag{5.21}$$

Low order of accuracy schemes for implicit integration, for example Euler's method and the trapezoidal rule are useful for demonstrating fundamentals of the collocation approach, however both schemes are generally considered too coarse for practical implementation.

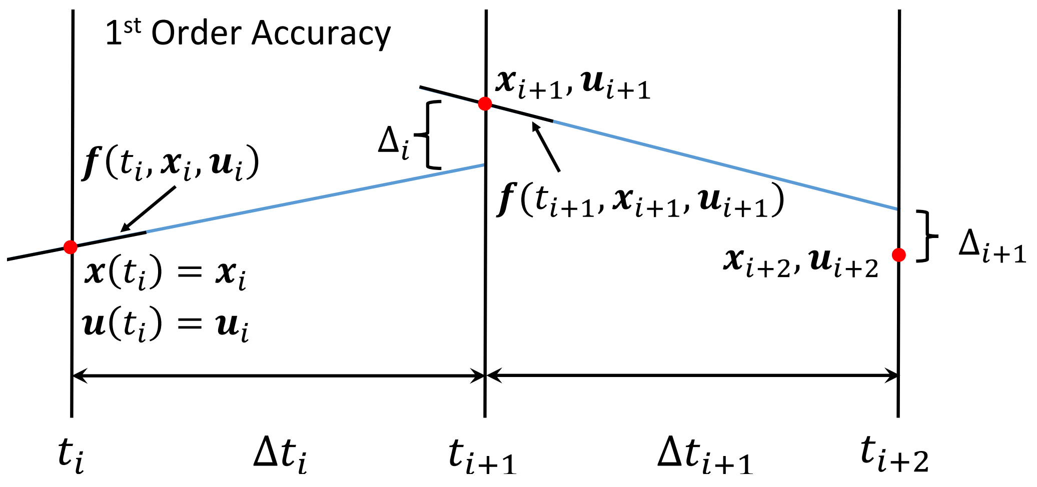

1st Order Accuracy
$\boldsymbol{x}_{i+1}, \boldsymbol{u}_{i+1}$
Δ_i
$\boldsymbol{f}(t_i, \boldsymbol{x}_i, \boldsymbol{u}_i)$
$\boldsymbol{f}(t_{i+1}, \boldsymbol{x}_{i+1}, \boldsymbol{u}_{i+1})$
$\boldsymbol{x}_{i+2}, \boldsymbol{u}_{i+2}$
Δ_{i+1}
$\boldsymbol{x}(t_i) = \boldsymbol{x}_i$
$\boldsymbol{u}(t_i) = \boldsymbol{u}_i$
t_i
Δt_i
t_{i+1}
Δt_{i+1}
t_{i+2}

Figure 5.1.: Collocation Using Euler's Rule

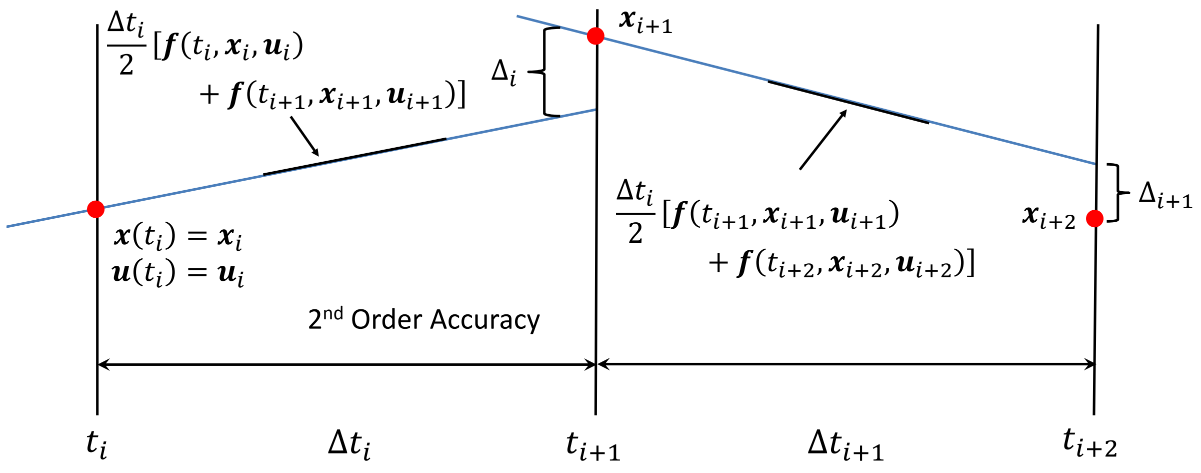

$\frac{\Delta t_i}{2}[\boldsymbol{f}(t_i, \boldsymbol{x}_i, \boldsymbol{u}_i) + \boldsymbol{f}(t_{i+1}, \boldsymbol{x}_{i+1}, \boldsymbol{u}_{i+1})]$
$\boldsymbol{x}_{i+1}$
Δ_i
$\frac{\Delta t_i}{2}[\boldsymbol{f}(t_{i+1}, \boldsymbol{x}_{i+1}, \boldsymbol{u}_{i+1}) + \boldsymbol{f}(t_{i+2}, \boldsymbol{x}_{i+2}, \boldsymbol{u}_{i+2})]$
$\boldsymbol{x}_{i+2}$
Δ_{i+1}
$\boldsymbol{x}(t_i) = \boldsymbol{x}_i$
$\boldsymbol{u}(t_i) = \boldsymbol{u}_i$
2nd Order Accuracy
t_i
Δt_i
t_{i+1}
Δt_{i+1}
t_{i+2}

Figure 5.2.: Collocation Using the Trapezoidal Rule

The accuracy of a particular discretization strategy is influenced by the number of segments and the order of the implicit integration method. Higher degree polynomials typically yield greater accuracy, although errors can arise from very high degree polynomials. The same order of accuracy is achieved by fitting a discretization scheme containing a few segments with higher order polynomials or by fitting many smaller segments with lower degree polynomials. The specific approach implemented to attain a desired degree of accuracy is determined based on problem objectives and computational power. A flexible collocation scheme that allows the polynomial degree N to be easily adjusted to achieve the desired order of accuracy is advantageous, such a scheme was developed by Williams [19]. Grebow and Pavalk [55] refined Williams' variable polynomial degree collocation scheme, the method supplied by these authors that employs only odd degree polynomials is represented here. Grebow and Pavlak offer a similar formulation for even degree polynomials as well, however, odd degree polynomials yield the same accuracy with a less complex formulation. Consider an N^{th} degree polynomial for segment i, i.e.,

$$\boldsymbol{p}_i(\tau) = \boldsymbol{C}_i\{1\ \tau\ \tau^2\ \cdots\ \tau^N\}^T \tag{5.22}$$

where $\boldsymbol{C}_i$ is a matrix of polynomial coefficients with dimensions $l_s \times (N+1)$. The polynomial defining segment i is then a $l_s \times 1$ vector that approximates the states at the normalized time τ. The general formulation offered for representing the polynomials of a collocation scheme accommodates any degree polynomial and an arbitrary number of state variables.

A variety of schemes are available for constructing the polynomials along each segment. The matrix of polynomial coefficients, $\boldsymbol{C}_i$, is constructed using states at one or more nodes on segment i. Each segment is subdivided into $(N+1)/2$ *variable nodes* and $(N-1)/2$ *defect points*. The index of a variable node or defect point is indicated by the subscript $j = 1, 2, \ldots, N$, and the variable nodes and defect points occur at odd and even numbered j, respectively, as demonstrated in Figure 5.3. A node placement scheme determines the normalized times τ_j at which the variable nodes and boundary points are placed. Equally distributing the variable nodes in normalized time is the simplest scheme, however, higher orders of accuracy are delivered by placing the nodes at the roots of Legendre or Chebyshev polynomials. Several node placement schemes prevalent in the literature are listed in Table 5.16, where $P_N(\tau)$ are degree N Legendre polynomials, where N is the degree of the polynomial. The collocation algorithm in this investigation utilizes Legendre-Gauss points. Because LG points do not lie on the boundaries of a segment, computations associated with each segment are independent, and this feature is leveraged to improve computational speed.

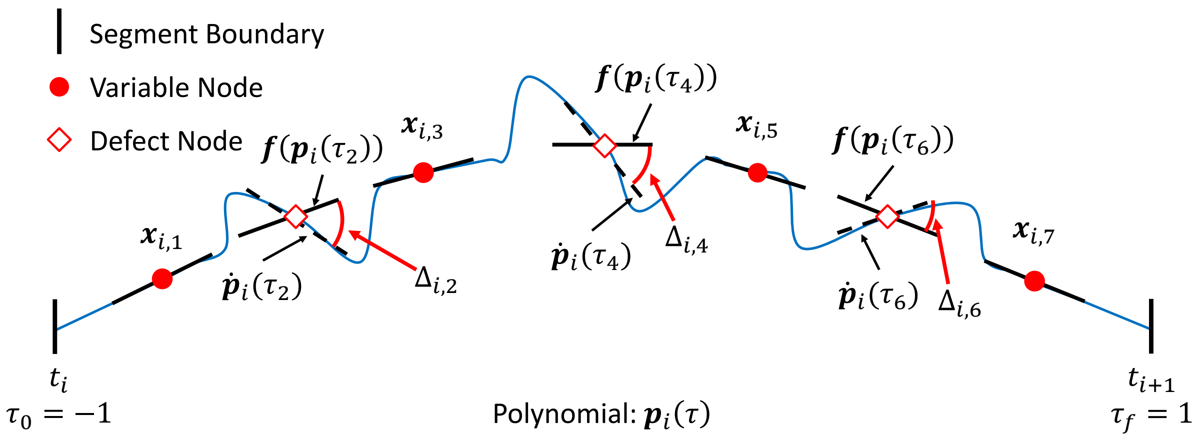


Figure 5.3.: Collocation Using a 7th Degree Polynomial

Table 5.1.: Node Placement Schemes

Method	Description	Order of Accuracy
Legendre-Gauss-Lobatto (LGL)	τ_j at -1 and 1 and at the roots of $\dot{P}_{N-1}(\tau)$	$2N-2$
Legendre-Gauss-Radau (LG)	τ_j at the roots of $P_{N-1}(\tau)$ and $P_N(\tau)$	$2N-1$
Legendre-Gauss (LG)	τ_j at the roots of $P_N(\tau)$	$2N$

A polynomial segment $\boldsymbol{p}_i$ approximates states and derivatives at times τ_j along a discretization segment. The states and derivatives as evaluated by a polynomial at each τ_j are represented as $\boldsymbol{p}_{i,j} = \boldsymbol{p}_i(\tau_j)$ and $\dot{\boldsymbol{p}}_{i,j} = d\boldsymbol{p}_i(\tau_j)/d\tau$, respectively. These values are constrained to equal those values at the variable nodes, that is,

$$\boldsymbol{p}_{i,j} = \boldsymbol{x}_{i,j}, \quad i = 1, \dots, n-1 \tag{5.23}$$

$$\dot{\boldsymbol{p}}_{i,j} = \dot{\boldsymbol{x}}_{i,j}, \quad j = 1, 3, \dots, \frac{N+1}{2} \tag{5.24}$$

where $\boldsymbol{x}_{i,j}$ is the vector state at node j on segment i. The derivatives at these nodes are computed as,

$$\dot{\boldsymbol{x}}_{i,j} = \frac{\Delta t_i}{2}\boldsymbol{f}\left[\tau_j, \boldsymbol{x}_{i,j}, \boldsymbol{u}_{i,j}\right] \tag{5.25}$$

Recall Δt_i is the non-normalized time interval of a segment i, thus, the coefficient $\Delta t_i/2$ transforms the derivatives to normalized time. Substituting equation (5.22) into equations (5.23) and (5.24) produces a matrix representation of the variable node constraints along an entire segment,

$$\boldsymbol{C}_i\left[\boldsymbol{\tau}\ \dot{\boldsymbol{\tau}}\right] = \left[\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,3}, \cdots \boldsymbol{x}_{i,N} | \dot{\boldsymbol{x}}_{i,1}\dot{\boldsymbol{x}}_{i,3}\cdots\dot{\boldsymbol{x}}_{i,N}\right] \tag{5.26}$$

where $\boldsymbol{\tau}$ and $\dot{\boldsymbol{\tau}}$ represent the matrices,

$$\boldsymbol{\tau} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \tau_1 & \tau_3 & \cdots & \tau_N \\ \tau_1^2 & \tau_3^2 & \cdots & \tau_N^2 \\ \vdots & \vdots & \cdots & \vdots \\ \tau_1^N & \tau_3^N & \cdots & \tau_N^N \end{bmatrix} \tag{5.27}$$

$$\dot{\boldsymbol{\tau}} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 1 \\ 2\tau_1 & 2\tau_3 & \cdots & 2\tau_N \\ \vdots & \vdots & \cdots & \vdots \\ N\tau_1^{N-1} & N\tau_3^{N-1} & \cdots & N\tau_N^{N-1} \end{bmatrix} \tag{5.28}$$

The system of equations in equation (5.26) contains $2N \times l_s$ constraints and $l_s \times N + 1$ unknown coefficient values. The states at the variable nodes are supplied as part of the initial guess for the collocation problem. The unknown values are then computed by solving equation (5.26) for $\boldsymbol{C}_i$.

$$\boldsymbol{C}_i = \left[\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,3}, \ldots, \boldsymbol{x}_{i,N} | \dot{\boldsymbol{x}}_{i,1}, \dot{\boldsymbol{x}}_{i,3}, \ldots, \dot{\boldsymbol{x}}_{i,N}\right]\boldsymbol{A}^{-1} \tag{5.29}$$

The matrices $\boldsymbol{\tau}$ and $\dot{\boldsymbol{\tau}}$ are combined into a single matrix $\boldsymbol{A}$ that is square and non-singular,

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & \cdots & 1 & | & 0 & 0 & \cdots & 0 \\ \tau_1 & \tau_3 & \cdots & \tau_N & | & 1 & 1 & \cdots & 1 \\ \tau_1^2 & \tau_3^2 & \cdots & \tau_N^2 & | & 2\tau_1 & 2\tau_3 & \cdots & 2\tau_N \\ \vdots & \vdots & \cdots & \vdots & | & \vdots & \vdots & \cdots & \vdots \\ \tau_1^N & \tau_3^N & \cdots & \tau_N^N & | & N\tau_1^{N-1} & N\tau_3^{N-1} & \cdots & N\tau_N^{N-1} \end{bmatrix} \tag{5.30}$$

If first order differential equations are available for any of the control variables, $\boldsymbol{u}_{i,j}$, then polynomials can be constructed in the same way for these variables.

Construction of the matrix $\boldsymbol{C}_i$ involves the development of a polynomial for each state along the segment i, and this polynomial is leveraged to approximate the states at the defect points. The normalized times at the location of the defect points are used to create the matrices $\boldsymbol{B}$ and $\boldsymbol{D}$. Recall that the defect point locations are determined from the same node placement scheme used to place the variable nodes,

$$\boldsymbol{B} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 & 1 \\ -1 & \tau_2 & \tau_4 & \cdots & \tau_{N-1} & 1 \\ -1 & \tau_2^2 & \tau_4^2 & \cdots & \tau_{N-1}^2 & 1 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ -1 & \tau_2^N & \tau_4^N & \cdots & \tau_{N-1}^N & 1 \end{bmatrix} \tag{5.31}$$

$$\boldsymbol{D} = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 1 \\ 2\tau_2 & 2\tau_4^2 & \cdots & 2\tau_{N-1} \\ \vdots & \vdots & \cdots & \vdots \\ N\tau_2^{N-1} & N\tau_4^{N-1} & \cdots & N\tau_{N-1}^{N-1} \end{bmatrix} \tag{5.32}$$

The first and last columns of $\boldsymbol{B}$ are included to compute the states at the boundary nodes along a segment because, depending upon the node placement scheme, the boundary nodes may not be variable nodes, meaning they are not used to construct the segment polynomial. Therefore, it is convenient to compute the boundary node states as well so that they are available toward the evaluation of other constraints. Note, because time along each segment is normalized, the step sizes to define τ_j are the same along every segment, thus, the matrices $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{D}$ are computed only once, when a collocation problem is initially formulated. The matrix $\boldsymbol{B}$ and the polynomial coefficients are then employed to construct the states at the defect points and boundary nodes.

$$\begin{bmatrix}\boldsymbol{x}_{i,0} \; \boldsymbol{x}_{i,2} \; \boldsymbol{x}_{i,4} \; \cdots \; \boldsymbol{x}_{i,N-1} \; \boldsymbol{x}_{i,f}\end{bmatrix} = \boldsymbol{C}_i\boldsymbol{B} \tag{5.33}$$

If a polynomial representation of the control history is employed, then states at the control points are computed as demonstrated in equation (5.33). Otherwise, control values at the defect point j are assumed to be equivalent to the values at the variable node $j+1$. The polynomials constructed in equation (5.29) are also used to compute the derivatives of the states at the defect points by evaluating $\boldsymbol{C}_i\boldsymbol{D}$. The derivatives of the states at the defect points are also computed using the dynamic equations in equation (5.25). The defect equations, also denoted defect constraints, are defined by the difference between the results obtained from the two approaches for constructing the derivatives at the defect points,

$$\left[\boldsymbol{C}_i\boldsymbol{D} - \begin{bmatrix}\dot{\boldsymbol{x}}_{i,2} \; \dot{\boldsymbol{x}}_{i,4} \; \cdots \; \dot{\boldsymbol{x}}_{i,N-1}\end{bmatrix}\right]\boldsymbol{W} = \boldsymbol{0} \tag{5.34}$$

The diagonal matrix $\boldsymbol{W}$ in equation (5.34) incorporates quadrature weights for each variable node that enable the chosen node placement scheme to provide a higher order of accuracy than the same number of unweighted nodes. The values of the weights are determined by the node placement scheme, for example LG or LGL. Driving the defect constraints to zero, or some acceptable tolerance, ensures that the polynomials constructed for each segment sufficiently approximate the system dynamics. When convergence is obtained, the solution for the collocation problem yields piecewise polynomials that approximate the state at any point along a solution arc. The variables node states that cause the defect constraint equations to evaluate to zero are constructed by posing the collocation problem in terms of differential corrections which is implemented with a free-variable and constraint formulation. The design variables in this framework are the position and velocity states at the variable nodes, i.e.,

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_{1,1} & \boldsymbol{x}_{1,3} & \cdots & \boldsymbol{x}_{1,N} \\ \boldsymbol{x}_{2,1} & \boldsymbol{x}_{2,3} & \cdots & \boldsymbol{x}_{2,N} \\ \vdots & \vdots & \ddots & \boldsymbol{x}_{1,N} \\ \boldsymbol{x}_{n-1,1} & \boldsymbol{x}_{n-1,3} & \cdots & \boldsymbol{x}_{n-1,N} \end{bmatrix} \tag{5.35}$$

The matrix in equation (5.35) is reshaped into a single column vector for application with Newton's method. The defect constraints along each segment are included in the constraint vector, as are continuity constraints that ensure that the segments comprising the final trajectory are continuous in position and velocity,

$$\boldsymbol{F} = \begin{bmatrix} \boldsymbol{x}_{1,0} - \boldsymbol{x}_{0,fix} \\ \boldsymbol{x}_{2,0} - \boldsymbol{x}_{1,f} \\ \vdots \\ \boldsymbol{x}_{n-1,0} - \boldsymbol{x}_{n-2,f} \\ \left[\boldsymbol{C}_1\boldsymbol{D} - \left[\dot{\boldsymbol{x}}_{1,2}\ \dot{\boldsymbol{x}}_{1,4}\ \cdots\ \dot{\boldsymbol{x}}_{1,N-1}\right]\right]\boldsymbol{W} \\ \left[\boldsymbol{C}_2\boldsymbol{D} - \left[\dot{\boldsymbol{x}}_{2,2}\ \dot{\boldsymbol{x}}_{2,4}\ \cdots\ \dot{\boldsymbol{x}}_{1,N-1}\right]\right]\boldsymbol{W} \\ \vdots \\ \left[\boldsymbol{C}_{n-1}\boldsymbol{D} - \left[\dot{\boldsymbol{x}}_{n-1,2}\ \dot{\boldsymbol{x}}_{n-1,4}\ \cdots\ \dot{\boldsymbol{x}}_{n-1,N-1}\right]\right]\boldsymbol{W} \end{bmatrix} = \boldsymbol{0} \tag{5.36}$$

The vector $\boldsymbol{x}_{0,fix}$ in equation (5.36) is a desired initial state. The constraints that define the collocation problem are formulated such that computations along each segment are independent. Therefore, the Jacobian matrix that results from the partial derivatives of $\boldsymbol{F}$ with respect to $\boldsymbol{x}$ is highly sparse. The sparsity of the Jacobian matrix is leveraged to implement strategies that reduce the computation times for general collocation algorithms. These strategies are especially essential when the collocation problem becomes very large. The procedure for solving a NLP problem by collocation is illustrated in the flowchart in Figure 5.4. A converged result to a collocation problem yields the approximation of a solution to a system of differential equations, and this approximation is dependent upon the parameters of the discretization technique and the implicit integration method. The accuracy of the final solution is improved by refining the discretization mesh.

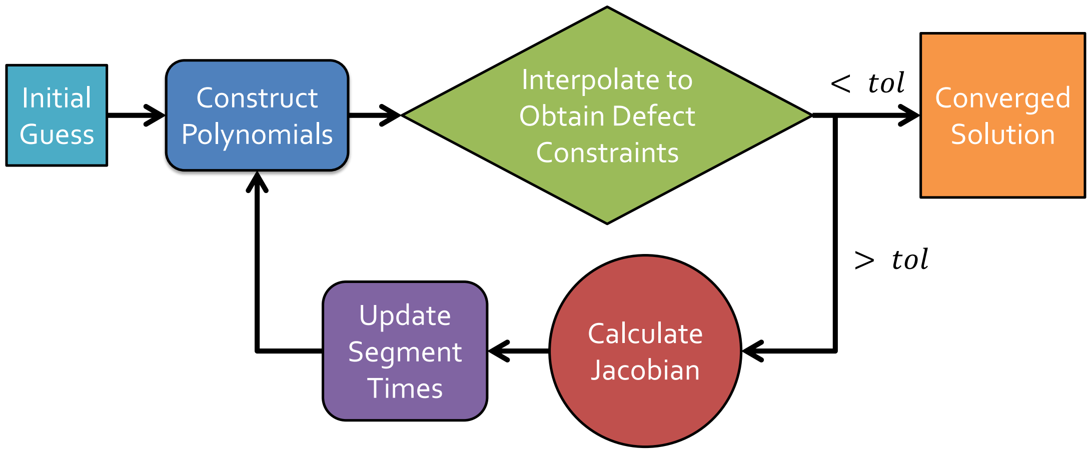


Figure 5.4.: Procedure for Applying Collocation with Newton's Method

5.4 Mesh Refinement

The solution of an optimal control problem via collocation produces a set of piecewise polynomials that approximate a solution to a system of first-order differential equations. The accuracy of this solution is highly dependent on the spacing and number of nodes used in the problem mesh. A variety of methods exist for refining the mesh such that the error in the converged collocation solution is below a desired tolerance; some methods also ensure that the total error is equally distributed across segments. Russell and Christiansen compared several mesh refinement schemes [56], including the de Boor mesh refinement technique. Originally developed by Carl de Boor [57], this approach reduces and equally distributes the error among the segments in the collocation problem; given its computational efficiency, the de Boor method is leveraged in this investigation. Details concerning the de Boor mesh refinement approach follow from a framework employed by Ozimek et al. [54] and then further honed by Grebow and Pavlak [58].

Mesh refinement implies both potential adjustment in the size of the grid as well as a guided approach to locating the nodes that define the mesh. The placement of points that define the mesh is adjusted based on the error predicted for the polynomial approximations of the states in a converged collocation solution. The error in a polynomial approximation of order N for a segment i is,

$$e_i = K\Delta t_i^{N+1}\xi_i + O(\Delta t_i^{N+2}) \tag{5.37}$$

where the scalar error due to higher order terms, $O(\Delta t_i^{N+2})$, is determined by the time interval along a given segment, where ξ_i represents the $N^{\text{th}+1}$ derivative of the constructed polynomial. The constant K is a dimensionless scalar value that is dependent on the degree of the polynomial. A method for computing K is presented in the Appendix of Russell and Christiansen [56] and several pre-computed values are provided in Table 5.2. De Boor states that the variable ξ_i is estimated by the $N^{\text{th}+1}$ derivative of the solution [57], however, this value cannot be calculated directly from the polynomial in the collocation solution. As an alternative, the polynomial and time intervals on the converged collocation solution are used to approximate ξ_i.

$$\xi_i \approx \begin{cases} 2\max\left[\frac{\left|\boldsymbol{p}_1^{(N)} - \boldsymbol{p}_2^{(N)}\right|}{\Delta t_1 + \Delta t_2}\right], & \text{on} \quad (t_1, t_2) \\ \max\left[\frac{\left|\boldsymbol{p}_{i-1}^{(N)} - \boldsymbol{p}_i^{(N)}\right|}{\Delta t_{i-1} + \Delta t_i}\right] + \max\left[\frac{\left|\boldsymbol{p}_{i+1}^{(N)} - \boldsymbol{p}_i^{(N)}\right|}{\Delta t_{i+1} + \Delta t_i}\right], & \text{on} \quad (t_i, t_i + 1), \quad i = 2, \dots, s-1 \\ 2\max\left[\frac{\left|\boldsymbol{p}_{s+1}^{(N)} - \boldsymbol{p}_s^{(N)}\right|}{\Delta t_{s+1} + \Delta t_s}\right], & \text{on} \quad (t_s, t_{s+1}) \end{cases} \tag{5.38}$$

Note, s is a new parameter representing the number of segments in a mesh $s = n - 1$. The error calculated with equations (5.37) and (5.38) is an approximation of the error in the polynomial fit to the segment, it is not equivalent to the error that would result from an explicit propagation of the segment.

The error in the collocation solution for each segment may vary significantly along a path after it is initially converged. However, equally distributing error along a

Table 5.2.: Constant Values for Odd Degree Polynomial Error Calculation

Degree	K Value
1	1.25
2	$8.01875373874480 \times 10^{-3}$
3	$5.20833333333334 \times 10^{-4}$
4	$2.45076190281488 \times 10^{-5}$
5	$1.03339947089947 \times 10^{-6}$
6	$3.59267656090070 \times 10^{-8}$
7	$1.12915151977652 \times 10^{-9}$
8	$3.08927792667408 \times 10^{-11}$
9	$7.74907905728965 \times 10^{-13}$
10	$1.74408359272640 \times 10^{-14}$
11	$3.64148451940432 \times 10^{-16}$

trajectory is desirable because it ensures efficient node placement by locating more boundary points at times where the solution is rapidly changing and vice versa. The times at which each boundary point occurs are updated to achieve error equidistribution, and the updated times are identified by,

$$t_{i+1} = I^{-1}\left[\frac{iI(t_{s+1})}{s}\right], \quad i = 1, \ldots, s-1 \tag{5.39}$$

where I is determined from the integral,

$$I(t) = \int_{t_1}^{t} \xi_i(s)^{\frac{1}{n+1}} ds \tag{5.40}$$

Calculation of the integral in equation (5.40) is simplified by the fact that $\xi_i(s)$ is approximated by a piecewise constant method. Therefore, equation (5.40) can be precisely constructed by the rectangle rule of integration. Note, equation (5.39) is the inverse integral of I, computed at the value indicated by the expression within the brackets. State and control values at the new boundary node times, computed via equation (5.39), are interpolated from the mesh nodes that support the previously converged collocation solution; the NLP problem is then resolved with these new values. The process of computing the quantities in equations (5.37)-(5.40) and re-solving the NLP problem, is iterated until the maximum difference between the segment errors is less than a predefined tolerance.

After the error of the polynomial approximations is equally distributed along each segment of the mesh the number of segments is updated as well. The number of segments in the mesh is updated as,

$$s_{j+1} = \text{round}\left[s_j \left(\frac{10 e_i}{tol} \right)^{\frac{1}{N+1}} + 5 \right] \tag{5.41}$$

Once the number of segments is updated, both in number and location, the new boundary point times are calculated from equation (5.39) and the NLP problem is solved again. The procedure for equally distributing the error and reducing the total error by adding segments is repeated until the total error is below a desired tolerance. An schematic of the mesh refinement process appears in the flowchart in Figure 5.5. Inclusion of a mesh refinement scheme paired with a collocation method enables solutions to systems of differential equations on the same order of accuracy as explicit integration methods.

5.5 Nonlinear Programming Problem Setup

A collocation and mesh refinement scheme is employed to transcribe the continuous optimal control problem into a nonlinear programming problem. A variety of algorithms and software packages exist for solving NLPs, and many are tailored to efficiently solve specific problem types. Large-scale NLP algorithms, for example, the interior-point method, are well suited to solving NLP problems with a large number of variables and constraints. These algorithms leverage the sparsity of the Jacobian and Hessian matrices in large NLP problems to save computation time. Direct transcription of optimal trajectory design problems often produces large NLPs, thus,

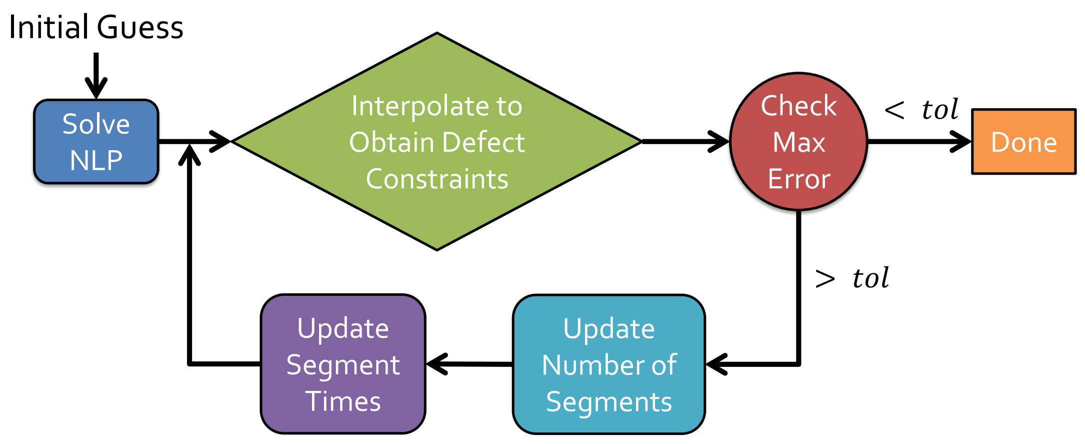


Figure 5.5.: Procedure for Applying Mesh Refinement with Collocation, Based on Figure by Grebow and Pavlak [58]

large-scale algorithms are applied. The widely available software package Matlab® contains an NLP solver, `fmincon`, with an interior-point algorithm to solve optimization problems. Other common software packages that leverage the sparsity of large matrices to solve large-scale NLPs are the Sparse Nonlinear OPTimizer (SNOPT) and the Interior Point OPTimizer (IPOPT) both of which are exploited in trajectory optimization tools. The interior-point algorithm option in the NLP solver `fmincon` was employed in this investigation; the derivation and details of this algorithm are available in Byrd et al. [59] [60].

Continuity and defect constraints from the collocation scheme are incorporated into the NLP, however, additional constraints are required to ensure that the NLP solver converges to a feasible final solution. Constraints on control variables are also incorporated into the derivation of the TPBVP in the indirect optimization method, therefore, these constraints are implicitly satisfied in the solution of the problem. However, in the NLP problem, constraints on the control values must be explicitly included in the constraint vector. One notable constraint involves the maximum power value governing the spacecraft engine. It is guided to remain within the range

$0 \leq P \leq P_{max}$. This constraint is incorporated into the NLP problem formulation using the slack variable σ, i.e.,

$$P_i - P_{max}\sin^2(\sigma_i) = 0 \tag{5.42}$$

$$\hat{\boldsymbol{u}}_{T_i}^T\hat{\boldsymbol{u}}_{T_i} - 1 = 0 \tag{5.43}$$

$$T_i - \eta^2 = 0 \tag{5.44}$$

Additionally, as is apparent in equation 5.44 the slack variable η is added to constrain the thrust magnitude, T, to satisfy $T \geq 0$, and the direction of the thrust vector, $\hat{\boldsymbol{u}}_T$, is also constrained to maintain a magnitude equal to one. These control variable constraints, as formulated in equations (5.42)-(5.43), are applied at each variable node in the discretized optimal control problem.

Continuity is ensured by enforcing boundary constraints on the initial and final points along the trajectory. In the circular orbit transfer problem, the boundary constraints included the initial and final circular orbits; in the halo-to-halo transfer scenario, the thrust arc is constrained to be continuous through the end of the coasting segments, or

$$\boldsymbol{x}_{1,0} - \boldsymbol{\psi}_0 = \boldsymbol{0} \tag{5.45}$$

$$\boldsymbol{x}_{s,f} - \boldsymbol{\psi}_f = \boldsymbol{0} \tag{5.46}$$

Finally, the initial mass state is formulated as a boundary constraint, because without this constraint, the NLP solver increases the initial mass of the spacecraft in an effort to maximize the mass at the end of the transfer arc, therefore,

$$m_{1,0} - M_0 = 0 \tag{5.47}$$

The variable M_0 in equation (5.47) represents a fixed mass value equal to the desired initial mass of the spacecraft. These additional boundary and control variable constraints are combined with constraints associated with the collocation method to comprise the full constraint vector for the NLP.

The augmented design variable vector for implementation of the collocation scheme consists of the state and control values at each variable node. The circular orbit transfer problem is two dimensional and, therefore, requires only five state variables, while the three dimensional halo orbit transfer problems utilize seven state variables. Similarly, the circular orbit transfer employs four control variables while the halo orbit transfer must employ five because an out-of-plane component is not required for the thrust pointing vector in the planar case. Additionally, the slack variable constraints introduced in the constraint equations (5.42) and (5.44), respectively, are incorporated as design variables. The slack variables are included as design variables at each variable node, therefore, they introduce $2 \times (n-1) \times (N+1)/2$ additional variables to the design vector. Furthermore, the values for nondimensional time defining the coasting periods, τ_i and γ_i, that appear in the halo-to-halo transfer sample problems are also incorporated in the design vector. Inclusion of these values adds 2 or 4 additional design variables, depending on the potential to formulate the transfer with manifold arcs. In summary, the circular orbit transfer and halo to halo orbit transfers require 11 and 14 design variables per variable node, respectively. If the coasting parameters are included, these values clearly lengthen the design variable vector.

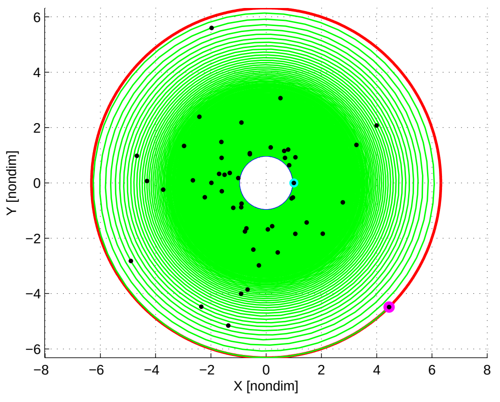
Once the objective function and the problem constraints are identified, the appropriate relationships that deliver these quantities are supplied to the selected NLP problem solver along with an initial guess for the design variable values. Upper and lower bounds on the design variables, as well as the desired tolerances, can also be introduced as inputs to the NLP solver. Convergence of the solver results in an approximation for an optimal trajectory. The accuracy of this solution is dependent upon the order of the polynomial and the number of segments in the collocation scheme.

5.6 Example Problems:

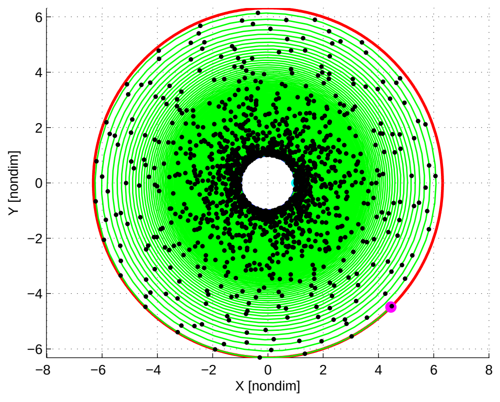
A collocation and mesh refinement scheme is employed in conjunction with the interior-point algorithm option in `fmincon` to solve the same three sample problems that appeared in Chapter 4. Solutions emerging from the indirect optimization approach detailed previously are introduced as initial guesses for the corresponding direct optimization sample problems. The indirect optimization solutions are discretized into boundary nodes spaced equally with respect to time, and the associated variable node locations are computed consistent with the Legendre-Gauss node spacing scheme. The discretized solution to the indirect optimization problem then serves as the initial mesh for the direct optimization problem that is subsequently solved using collocation.

5.6.1 Circular Orbit Transfer

A driving factor in the selection of the continuous low-thrust circular orbit transfer as a sample problem is the expectation that the large number of revolutions seemingly required of this transfer make convergence with a direct optimization method challenging. The circular orbit transfer examined in Chapter 4 required 75 days of continuous thrusting and several hundred spirals expanding away from the Earth. Accurately achieving a similar trajectory using a collocation method, requires a discretization with many segments; which delivers a large dimensioned NLP comprised of thousands of discrete state and control values. Such problems can exhibit extraordinarily slow convergence rates, with the possible lack of convergence, particularly when the required gradients are computed numerically. Such is the case when the direct optimization approach is applied to the 75 day circular orbit transfer. The sparsity of the large dimensioned NLP problem was leveraged to rapidly produce a solution using collocation alone, however, when the direct optimization algorithm `fmincon` is incorporated into this process, the convergence rate is impractically slow. Figure 5.6 demonstrates the number of segments necessary to achieve an accurate non-optimized



(a) Boundary Points Before Mesh Refinement, $s = 50$



(b) Boundary Points After Mesh Refinement, $s = 1468$

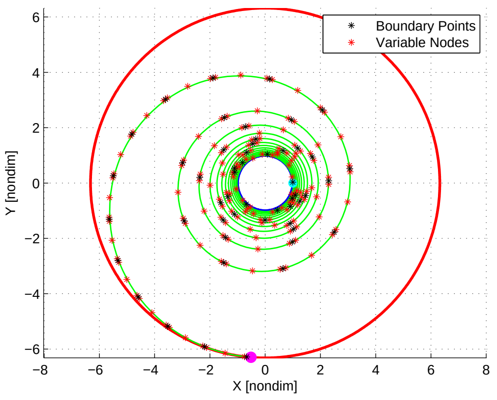
Figure 5.6.: Circular Orbit Transfer, $TF = 75$ $days$; Collocation without Optimization

result, with collocation, over the 75 day transfer. The initial discretization of the indirect optimization result consisted of 50 segments, as noted in Figure 5.6a, however, after mesh refinement the converged collocation solution contained 1468 segments, reflected in Figure 5.6b. Fit with a 5^{th} degree polynomial, the collocation solution consists of a 48444×1 design variable vector and a 35234×1 constraint vector which, together, produce a 35234×48444 Jacobian matrix that is 99.97% sparse. A problem of this size is not tractable by the available computational resources.

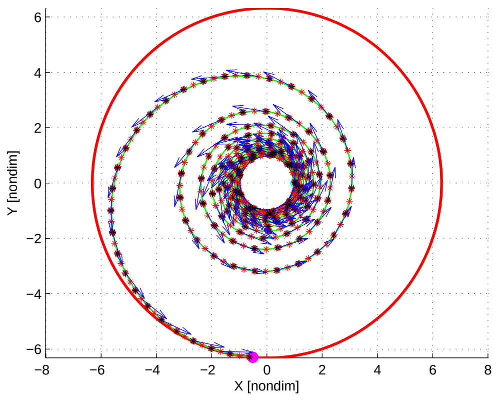
Low-thrust orbit transfers, in particular, those employing many revolutions, are an active area of research and other authors have offered techniques to enable the application of direct optimization methods to this type of transfer. One approach is orbital averaging and a hybrid control formulation to reduce the oscillatory behavior along the many revolution transfer [48] [61]. Such a plan decreases the convergence time required to obtain a solution. An alternate technique is a Runge-Kutta parallel shooting scheme that is incorporated into the direct optimization formulation such that the transfer problem is posed in terms of equinoctial elements that typically vary less rapidly with time than other coordinates [62]. Additionally, direct optimization utilizing psuedospectral methods has been demonstrated to be effective at solving various many revolution orbital transfer problems [63]. The increase in speed afforded by these techniques typically outweighs any corresponding decrease in solution accuracy. This investigation is primarily focused on demonstrating direct optimization approaches, thus, the large dimensioned problem is reduced simply by decreasing the required transfer time to 2 days. While the engine parameters to achieve this transfer time are impractical, the process of constructing the minimum fuel transfer path still provides insight into the characteristics of the optimization methods.

A collocation scheme utilizing 5^{th} order polynomials produces 3 variable nodes per segment resulting in a total of 150 variable nodes. Each variable node is associated with 5 state values, 4 control values, and 2 slack variable values yielding an initial design variable vector with dimensions 1650×1. The direct optimization result, both before and after mesh refinement, appears in Figures 5.7a and 5.7b, respectively. In Figure 5.7a, three red variable nodes are located between each boundary point, consistent with the 5^{th} degree polynomials used to obtain the results. Moreover, the distribution of the variable nodes between the boundary points reflects the Legendre-Gauss node spacing employed to create the polynomials. The boundary points and variable nodes in the optimized solution, closely follow the solution characteristics obtained using indirect optimization and, as expected, the thrust vector is directed parallel to the circular orbital velocity vector. The congruity of the two solutions suggests that both optimization techniques identified the same optimal solution. Following mesh refinement, the maximum error of the polynomial approximation along each segment for the entire solution is less than 1×10^{-8} and the error along each segment differed by no more than two decimal places.

Further correspondence in comparing the results of the two optimization methods is apparent in the time histories of the mass and control variables. The position states of the variable nodes in the direct optimization result are plotted along with the continuous indirect optimization solutions in Figure 5.8. Recall the spacecraft

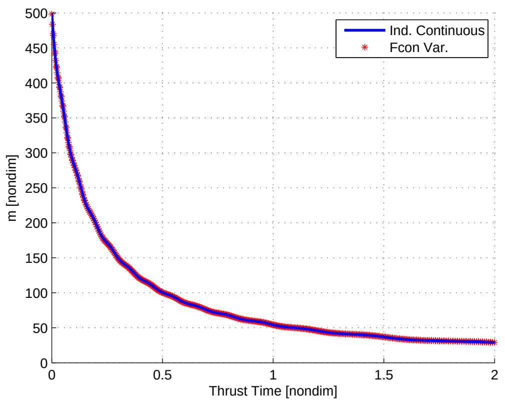


(a) Before Mesh Refinement, $s = 50$

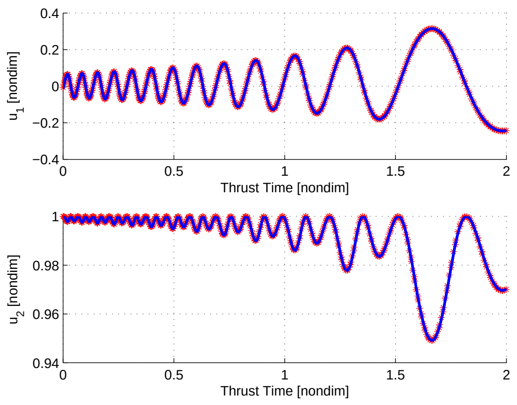


(b) After Mesh Refinement, $s = 170$

Figure 5.7.: Circular Orbit Transfer, $TF = 2\ days$; Direct Optimization Result



(a) Mass History



(b) Pointing Vector Components

Figure 5.8.: Circular Orbit Transfer Direct Optimization Mass and Thrust Pointing Vector Values

centered velocity frame defined in Section 4.3 by the unit vectors of the spacecraft velocity and angular momentum. The components of the thrust pointing vector, $\hat{\boldsymbol{u}}_T$, expressed in the velocity frame exhibit oscillatory behavior similar to some results from the indirect optimization for the same problem. The components of, $\hat{\boldsymbol{u}}_T$, are associated with Cartesian coordinates in the velocity frame, therefore, they are not constant like the components of $\boldsymbol{\lambda}_v$ in Figure 4.7. The mass time history in Figure 5.8a demonstrates that the evolving mass values produced from application of each optimization method are nearly identical.

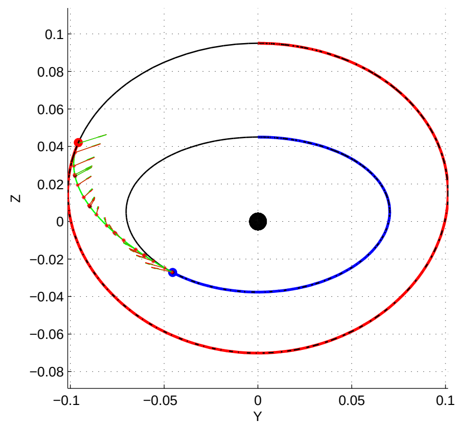
Application of direct optimization with collocation to the continuous low-thrust circular orbit transfer problem initially introduces difficulties due to the problem size, however, strategies are avilable to make this problem tractable. Including additional forces and dimensions in the continuous low-thrust transfer problem increases the practical value of the analysis and inclusion of these effects is simplified with a direct optimization approach. Therefore, further investigation is warranted.

5.6.2 Halo to Halo Transfers

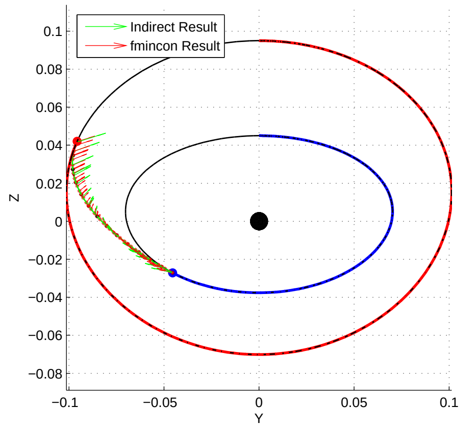
Transfer scenarios that do not involve significant spiraling typically require a lower number of segments to be accurately approximated using collocation methods. Fewer segments yields a lower dimensioned problem and one more immediately suitable to solution via direct optimization with collocation. The transfers between periodic orbits examined in Chapter 4 exhibit a simple geometry, therefore the direct optimization solutions for these problems require far fewer segments than the circular orbit transfer type of problem. Similar to the previous results, direct optimization again proved to rernder an accurate optimal solution that closely coincides with the indirect optimization results.

L_1 Halo to L_1 Halo Transfer

The thrusting portion of the transfer between L_1 halo orbits from Section 4.4.2 is discretized into 6 boundary points and a collocation scheme, utilizing 7^{th} order Hermite polynomials, is used to approximate the transfer over the corresponding 5 segments. The 7^{th} order polynomial produced 4 variable nodes per segment resulting in a total of 20 variable nodes, each associated with 7 state values, 5 control values, and 2 slack variable values. Including the two nondimensional time parameters to define the coasting phases along the halo orbits, the initial number of design variables in this problem is 282. The collocation scheme and `fmincon` converge under this initial discretization as seen in Figure 5.9a. Then a subsequent mesh refinement produces a final result comprised of 13 segments, which is plotted in Figure 5.9b. The error in the polynomial approximations over each segment differs by no more than two decimal places with a maximum value of 2.928737×10^{-9} nondimensional units. The position states and thrust pointing vector of the final solution supplied by the direct optimization method agree closely with the result of the indirect optimization method when both are plotted in configuration space as shown in Figure 5.9b.

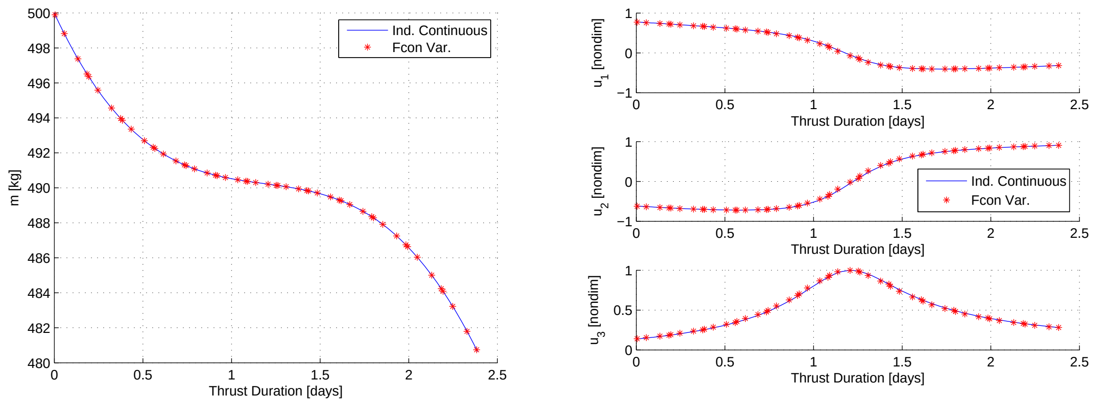


(a) No Mesh Refinement, $s = 5$



(b) Mesh Refinement, $s = 13$

Figure 5.9.: L_1 to L_1 Halo Transfer Direct Optimization Result



(a) Mass History (b) Pointing Vector Components

Figure 5.10.: L_1 to L_1 Halo Transfer Direct Optimization Mass and Thrust Pointing Vector Values

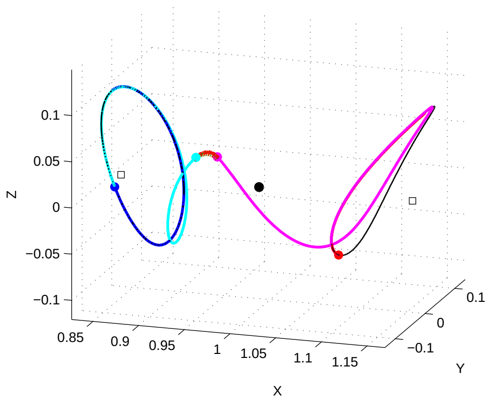
The agreement of the results from application of both optimization methods displayed in Figure 5.9 is also exhibited by the time histories of the state and control values along the transfer. The mass history over the low-thrust transfer obtained from indirect optimization is matched closely by the direct optimization result and the final mass values differ by less than a hundredth of a kilogram, as apparent in Table 5.3. The components of the thrust pointing vector emerging from each optimization strategy also coincide, and the trends observed regarding each component approximately match those of the costates in Figure 4.10. The correspondence in both states and control values between the results of the two optimization methods offers confidence that both approaches converged to the same optimal solution.

All of the state and control variables representing the direct optimization solution closely match the indirect optimization results. This agreement includes the value corresponding to the engine power level. Recall that the direct optimization formulation constrains engine power to remain within the bounds $0 \geq P \geq P_{max}$, however, power is not required to equal P_{max} as required in the indirect optimization approach. Despite this difference, the optimal solution from direct optimization produces a power level approximately equal to P_{max} over the duration of the transfer, confirming that this engine power setting corresponds to the optimal result. The differences that do appear between the results from the two optimization methods are analyzed more closely in Chapter 6 where the results produced by a direct optimization scheme are also examined when a less accurate initial guess is introduced.

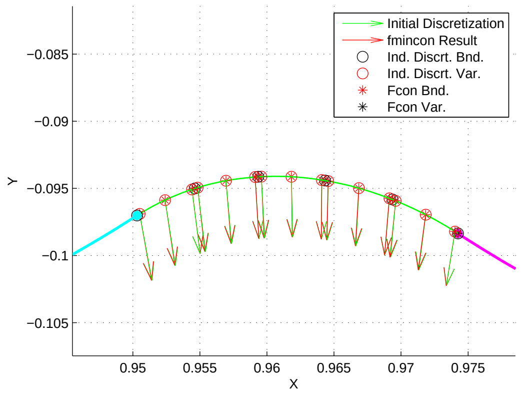
L_1 Halo to L_2 Halo Transfer with Manifold Arcs

The majority of the halo to halo transfer leveraging manifold arcs consists of coasting periods along the periodic orbits and manifolds. The relatively brief duration of the thrusting portion of the transfer means that it is discretized into only five segments, $s = 5$, for the initial guess of the collocation algorithm. Fifth order, $N = 5$, polynomials are chosen to fit this discretization, resulting in 15 variable nodes which, including the four nondimensional time parameters defining the coasting phases, produces 214 initial design variables. The initial design variable vector is passed into the direct optimization algorithm which converges without requiring mesh refinement. The position states of the boundary points and variable nodes of the converged solution are plotted with the continuous results of the indirect optimization method in Figure 5.11, and appear to closely follow this solution. The thrust pointing vector is derived from the values of the costates in the indirect optimization method, but is solved for explicitly in the direct optimization case; Figure 5.11b illustrates that both techniques produce the same thrust vector results. The close correspondence of the transfer geometry in Figure 5.11 suggests the direct optimization method accurately obtains the same optimal solution as the indirect method. The error of the polynomial approximations for each segment differs by no more than three decimal places and has a maximum value of 8.043262×10^{-9} nondimensional units.

The congruity of the results from both optimization methods seen in Figure 5.11 is also evident in the time histories of the state and control values. The mass values at variable nodes overlay the mass history from the indirect optimization result nearly

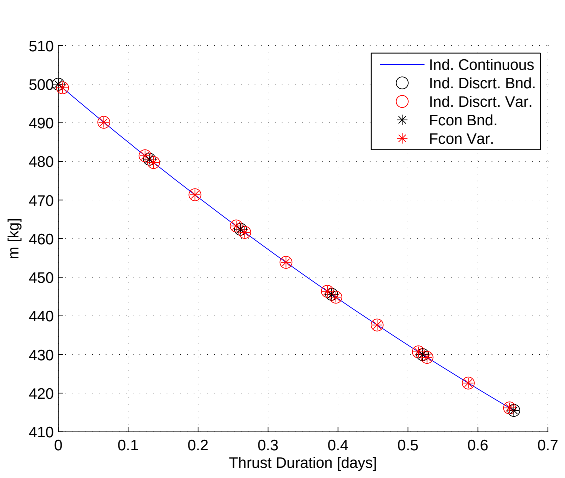


(a) Full Transfer

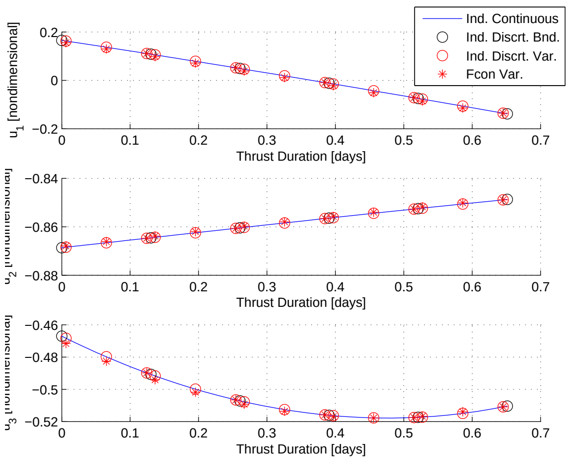


(b) Zoomed View of XY Plane

Figure 5.11.: L_1 to L_2 Halo Transfer Direct Optimization Result



(a) Mass History



(b) Pointing Vector Components

Figure 5.12.: L_1 to L_2 Halo Transfer Direct Optimization Mass and Thrust Pointing Vector Values

exactly in Figure 5.12a, and the final mass values differ by less than a tenth of a kilogram, Table 5.3. Because the number of segments is not updated between the initial and final solutions of this transfer the discrepancies between the results of the two optimization methods are more apparent. Several of the star markers that denote the mass values at the variable nodes of the direct optimization solution are off center from the circular markers that denote the mass values at the variable nodes of the discretized indirect optimization solution. The offset markers indicate differences between the results of each optimization method, and these variations are most noticeable in the plots of the components of the thrust pointing vector in Figure 5.12b. However, the solutions from both methods are in general agreement, and once again the trends of the costates in Figure 4.13 are approximately replicated by the pointing vector components in Figure 5.12b.

The state and control variables of the direct optimization solution closely match the indirect optimization results for the halo to halo orbit transfer problem using manifold arcs. The difference between the final masses obtained with each optimization method for all three sample problems is presented in Table 5.3. The final mass obtained with each method is identical to within at least hundredths of a kilogram. This difference is likely due to the approximations inherent in the collocation method which approximates the low-thrust transfer with piecewise polynomials. If higher order polynomials, or a greater number of segments are used in each case then the final mass values might exhibit even greater correspondence. However, for many practical purposes the mass values in these cases are close enough to be considered identical. Overall, the close agreement between the results of the indirect and direct optimization methods observed in each of the sample problems demonstrates the efficacy of both methods for conducting trajectory optimization. The two optimization methods achieve the same result with dramatically different techniques which may be best suited for specific types of problems. This fact is seen when the direct optimization technique employed in this investigation proves infeasible for solving the $TF = 75$ *days* low-thrust circular orbit transfer. Further comparison of the optimization methods offers insight into the scenarios each method is most applicable to.

Table 5.3.: Final Mass Values for Every Sample Problem

Problem Type	Indirect Optimization	Direct Optimization	Difference
Circular Orbit Transfer	29.091744 kg	29.091189 kg	0.000554 kg
L_1 Halo to L_1 Halo	480.642551 kg	480.639555 kg	0.002996 kg
L_1 Halo to L_2 Halo	415.533967 kg	415.530648 kg	0.003319 kg

6. COMPARISON OF OPTIMIZATION METHODS

The indirect and direct optimization methods presented in Chapters 4 and 5 respectively produced approximately the same results in each of the three sample problems. However, each optimization approach possesses distinct advantages and disadvantages that make it well suited for certain problems and a poor fit for others. Understanding these tradeoffs enables the optimization technique best suited for a particular problem to be chosen. Quantitative comparisons of the indirect and direct optimization results for each sample problem highlight several of the key differences between the optimization methods. The metrics of accuracy, robustness, and computational performance are used to examine these differences. Qualitative comparisons also yield insight, the ease of implementation and modification of each optimization technique is also discussed in this chapter. The two optimization methods inform one another, thus an understanding of one type is gained by contrasting it with the other.

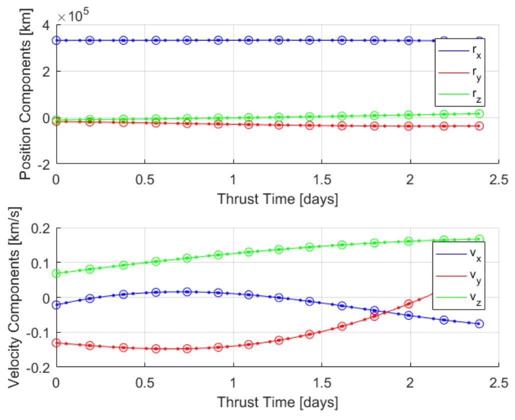
6.1 Quantitative Comparison

6.1.1 Accuracy Comparison

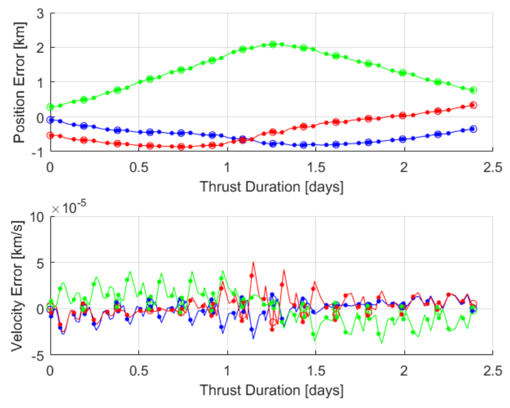
The transcription of a continuous trajectory into a discrete representation that is necessary to apply direct optimization methods, results in a loss of accuracy in the final optimal solution. The solutions produced by direct optimization methods consist of piecewise polynomials that approximate an optimal trajectory, and the convergence tolerances set for Newton's method and mesh refinement determine the accuracy with which these polynomials satisfy the differential equations of the system, but not necessarily the optimal solution. The accuracy of the optimal solution obtained with direct optimization methods is dependent upon the number of segments and degree of the polynomials because these parameters determine the resolution with which the discretization approximates the continuous solution. Indirect optimization methods are applied to the continuous optimal control problem, however when numerical methods are employed to solve the resulting TPBVP error due to discretization is also inevitably introduced. As with the direct optimization approach the extent of the error due to discretization is dependent upon the numerical scheme, however solving the TPBVP with a shooting method that utilizes explicit integration techniques can substantially reduce error. Explicit numerical integration schemes, for example the Adams Bashforth Moulton scheme used in Matlab's `ode113` algorithm, are capable of yielding highly accurate solutions commonly regarded as "truth" when comparing the results of numerical integration techniques. The accuracy of explicit integration combined with the assurance of obtaining a locally optimal solution afforded by Euler-Lagrange theory tends to make the indirect optimization technique the most efficient at reliably determining a locally optimal solution, however this is dependent on the problem formulation.

To assess the difference between the optimal solutions obtained by both methods the polynomials of the solution from the direct optimization approach were used to interpolate state variable values at times along the optimal trajectory. At each time step, including those of the boundary points and variable nodes, the states of the indirect optimization solution were subtracted from the corresponding states of the direct optimization solution. The difference between the two solutions indicates the accuracy with which the direct optimization method computed the locally optimal transfer. The difference between the two solutions is not necessarily a measure of accuracy, rather it simply indicates the extend of the difference between the optimal solutions determined by each method, if the difference is substantial it is likely that one optimization approach has at least converged to a different local optimal if not produced an erroneous result. The difference between the indirect and direct optimization results is calculated for ten separate runs of the direct optimization method, and the average of this difference is plotted for each of the halo to halo transfers. Figure 6.1b plots this difference for the transfer between L_1 and L_2 halo orbits using manifold arcs and plots of the position and velocity components of the transfer are offered in Figure 6.1a to provide context for the scale of the difference between solutions. On average the position and velocity components of the direct optimization solution differ from the indirect optimization solution by around 1 km and 10×10^{-5} km/s respectively. These discrepancies are approximately four orders of magnitude less than the scales over which the position and velocity components of the transfer vary. These differences are not insignificant, however they are small enough that the results of either method are useful for many applications, or as initial guesses for more accurate strategies. Table 5.3 indicated that the difference between the final mass values obtained by the two optimization methods was on the order of several thousandths of a kilogram, therefore despite position component differences on the order of 1 km the final value of the objective function is extremely similar in both cases. The transfer between two L_1 halo orbits exhibited differences between the two solutions with similar orders of magnitude to those seen in the halo orbit transfer employing manifold arcs, Figure 6.2a and 6.2b. These comparisons demonstrate that direct optimization methods provide less accurate, but still useful solutions.

The error shown in Figures 6.1 and 6.2 is not a measure of the integration accuracy of the optimal solution. The tolerances used for the explicit integration and collocation procedures ensure the model dynamics are satisfied with a high degree of accuracy, however this does not guarantee that either optimization method locates the exact optimal solution. The exact optimal solution is continuous, thus the control variables required to obtain this solution are continuous as well. In the both optimization methods the control values are constant in between integration time steps or variable nodes, therefore, some precision is lost when applying the control and the resulting solution is slightly less optimal. The magnitudes of the difference observed in Figures 6.1 and 6.2 are reduced by increasing the number of nodes used in the direct optimization method. A greater number of nodes permits more frequent changes in control, and finer application of control results in a solution that more ac-

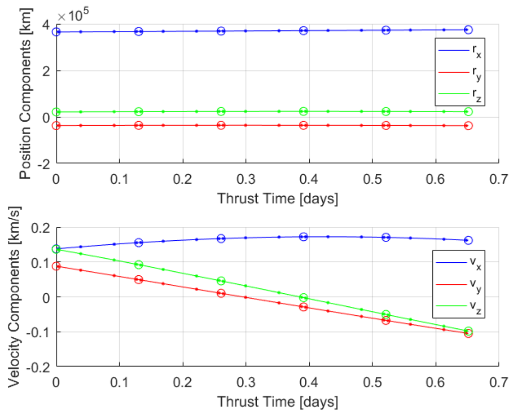


(a) Position and Velocity Components of Direct Optimization Solution

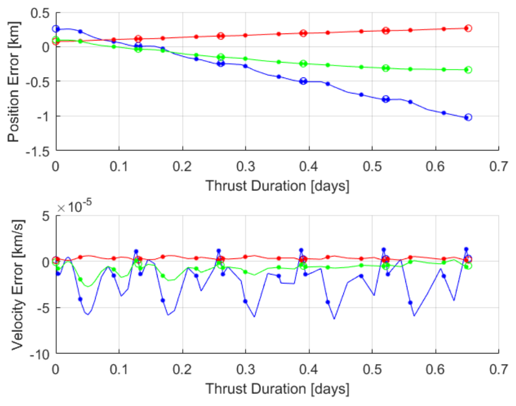


(b) Difference between Indirect and Direct Optimization Solutions

Figure 6.1.: Position and Velocity Components and Associated Error for Direct Optimization Results of Halo to Halo Orbit Transfer with Manifold Arcs



(a) Position and Velocity Components of Direct Optimization Solution



(b) Difference between Indirect and Direct Optimization Solutions

Figure 6.2.: Position and Velocity Components and Associated Error for Direct Optimization Results of Halo to Halo Orbit Transfer with Manifold Arcs

curately approximates the local optimal. Figure 6.3 shows the difference between the two optimization solutions 30 segments are utilized in the direct optimization scheme, employing a larger number of segments reduces the error of the optimal solution by an order of magnitude in this case. These segment numbers are well above what is necessary to satisfy the integration tolerances of the collocation method, however more segments are necessary to achieve correspondence between the solutions obtained by both optimization approaches.

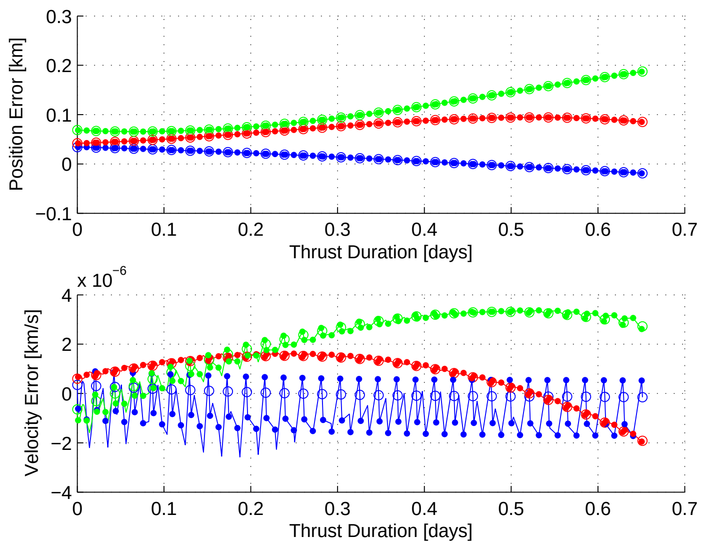
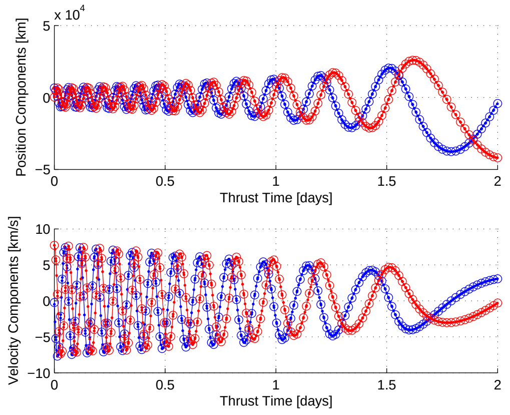
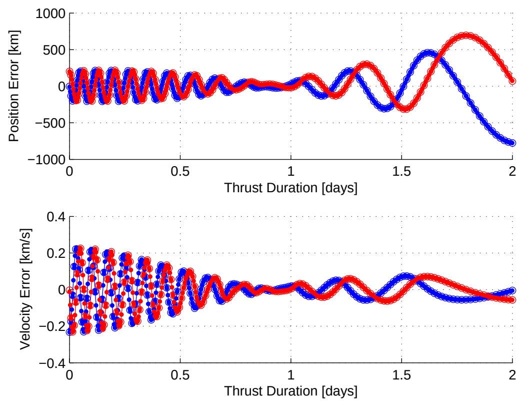


Figure 6.3.: Difference between Indirect and Direct Optimization for L_1 to L_2 Halo Orbit Transfer, $s = 30$

The importance of the number of nodes used in a direct optimization solution is best highlighted by examining the results of the circular orbit transfer problem shown in Figure 6.4. Although the direct optimization solution appears to exactly overlay the indirect optimization result in Figure 5.6, closer examination reveals significant differences between the two solutions. The circular orbit transfer obtained with the direct optimization method is slightly out of phase with the transfer from the indirect optimization transfer, and this difference results in large discrepancies between the two solutions, shown in Figure 6.4. Inspection of the circular orbit transfer indicates



(a) Position and Velocity Components of Direct Optimization Solution



(b) Difference between Indirect and Direct Optimization Solutions

Figure 6.4.: Position and Velocity Components and Associated Error for Direct Optimization Results of Circular Orbit Transfer

that on average less than 20 variable nodes are used per revolution around the central body with the number decreasing as the orbital altitude decreases. This low number of variable nodes per revolution provides too few opportunities to update the control variables for the same optimal solution to be obtained, typically more nodes per revolution are used in order to produce a more optimal result. Figure 6.4a indicates that in the gravity field of a massive central body the position and velocity components of a spacecraft are rapidly changing, therefore the control must be able to adjust rapidly as well if a reasonably optimal solution is to be obtained.

In summary, both optimization techniques require discretization when numerical methods are employed, and this discretization. Generally, the more coarse a discretization is the less optimal the result of an optimization method will be, because fewer opportunities to apply control are available. The tolerance associated with the numerical methods of both optimization approaches can be adjusted such that either method supplies poor optimal results, therefore it is not possible to designate one method as being inherently more accurate. However, the difference between the optimal solutions produced by both optimization approaches is a useful indicator of whether both methods converged to the same local optimal, or if one scheme converged to an incorrect result. The close agreement of the results of the two optimization methods applied to the halo to halo transfer scenarios indicates both approaches converged to approximately the same local optimal. However, the larger difference between the optimal solutions obtained for the circular-to-circular orbit transfer indicate that the two optimization methods may have converged to different locally optimal solutions.

6.1.2 Robustness Comparison

One of the greatest strengths of direct optimization methods is the fact that they exhibit greater robustness than indirect optimization techniques. A robust numerical method continues to converge to the same solution despite small changes to the initial guess. Thus, the direct optimization method is considered robust because its convergence to an optimal solution is less sensitive to errors in the initial guess than the indirect optimization method. This quality is also described as the direct optimization method having a wide basin of convergence, where the basin of convergence is the solution space “surrounding” a locally optimal solution in which an initial guess must lie for an optimizer to converge to the local optimal. The quality of robustness was tested by introducing random perturbations to the initial guess of a transfer problem and using the perturbed initial guess with both optimization methods. Testing robustness in this manner demonstrates the wider basin of convergence of the direct optimization method.

The L_1 halo to L_1 halo transfer problem is used to conduct the robustness test of each optimization method. The costates and τ coast times that form the initial guess, $\boldsymbol{X}_0$, for this transfer are perturbed using the relation, $\boldsymbol{X}_0 = \boldsymbol{X}_0(\mathbf{1} + \boldsymbol{h})$, where $\boldsymbol{h}$ is a random vector of the same dimension as $\boldsymbol{X_0}$, and $\mathbf{1}$ is a vector of ones with the same dimensions. The range of random values that occur in $\boldsymbol{h}$ is determined by the selected perturbation level, if a perturbation of $\pm 10\%$ of the initial guess is desired then the random values in $\boldsymbol{h}$ are between $-.1$ and $.1$. Perturbing the initial guess for the L_1 halo to L_1 halo transfer in this manner shifts the transfer insertion points along each halo orbit and alters the direction and magnitude of the thrust arc. Perturbing the initial guess for one of the other two transfer scenarios in this way produces a problem that is either dynamically infeasible with the given parameters or requires an unreasonably long convergence time. Because these added challenges obscure the results of the robustness test only the L_1 halo to L_1 halo transfer problem is examined. Perturbation levels from $\pm 10\%$ to $\pm 50\%$ with an interval of 10% are tested, and at each perturbation level ten runs of the sample problem are conducted with each optimization method employing a different random vector, $\boldsymbol{h}$, each time. This method of randomly perturbing the initial guess for the indirect optimization method within some bounds tested the robustness of each optimization approach when supplied with a poor initial guess.

The optimal L_1 halo to L_1 halo transfer converged upon in Chapter 4 requires approximately 20 kg of propellant, and the ability to re-converge to this solution after the introduction of a perturbation is used as the measure of robustness in this test. No check is conducted to ensure that the original transfer is the global optimal, however no transfer with a smaller Δm is found in this investigation. The perturbed initial guess for the indirect optimization method is passed to a single shooting scheme that solves the associated TPBVP. The single shooting scheme converged to a solution at all perturbation levels tested, however the solution was not always the same as the initial transfer produced in Chapter 4 that required 20 kg of propellant. A thrust arc propagated with an initial guess randomly perturbed by $\pm 40\%$ is shown in Figure 6.5, in configuration space this arc is directed away from the final halo orbit. Despite the poor initial guess the single shooting scheme converges upon a transfer between the two L_1 halo orbits, also shown in Figure 6.5, however the transfer obtained is clearly different than the original transfer displayed in Figure 4.8. In addition to the different configuration space appearance, the L_1 to L_1 halo transfer obtained with the perturbed initial guess required 30 kg of propellant. All of the new transfers obtained in the robustness test of the indirect optimization method required a larger propellant mass than the original transfer, and were therefore less optimal solutions. Out of the ten runs conducted at each perturbation level the number of runs for which the indirect optimization method converged back to the original, and most optimal, halo to halo transfer is recorded in Table 6.1.

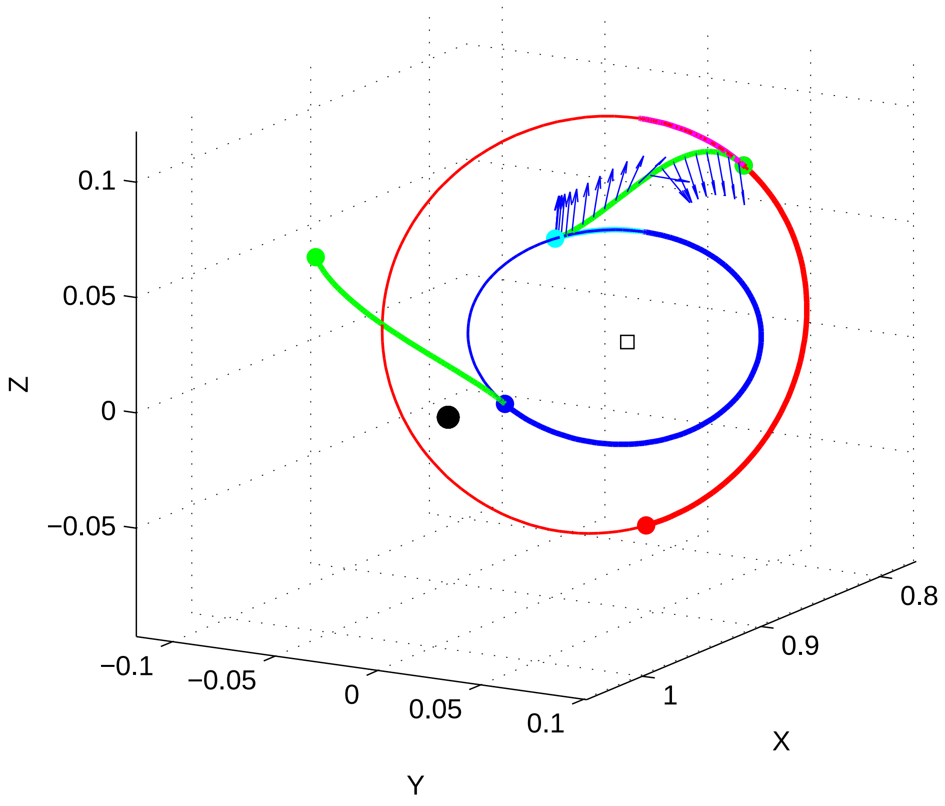


Figure 6.5.: Indirect Optimization Solution Following ±40% Perturbation to the Initial Guess; Perturbed Trajectory (green) and Converged Solution (green with blue thrust pointing vector arrows)

The results in Table 6.1 provide a general sense of the robustness of the indirect optimization method. At a perturbation level of ±10% the indirect optimization method converges back to the original result for nine of the ten runs conducted.

However, for perturbation levels of ±20% or higher no more than half of the ten runs converge back to the optimal solution that requires 20 kg of propellant. The remaining runs out of the ten converge to new transfers that require more propellant, such as the one shown in Figure 6.5. The broad trend indicated by Table 6.1 is that the solution obtained with the indirect optimization is fairly sensitive to changes in the initial guess. Some of this sensitivity could be mitigated by employing a multiple shooting rather than a single shooting scheme, however even with this approach the sensitivity of the solution to the values of the costates remains.

Table 6.1.: Optimization Robustness Analysis for L_1 Halo to L_2 Halo Transfer, Full Perturbation

Method	±10%	±20%	±30%	±40%	±50%
Indirect Optimization	9/10	4/10	5/10	5/10	5/10
Direct Optimization	10/10	10/10	10/10	10/10	8/10

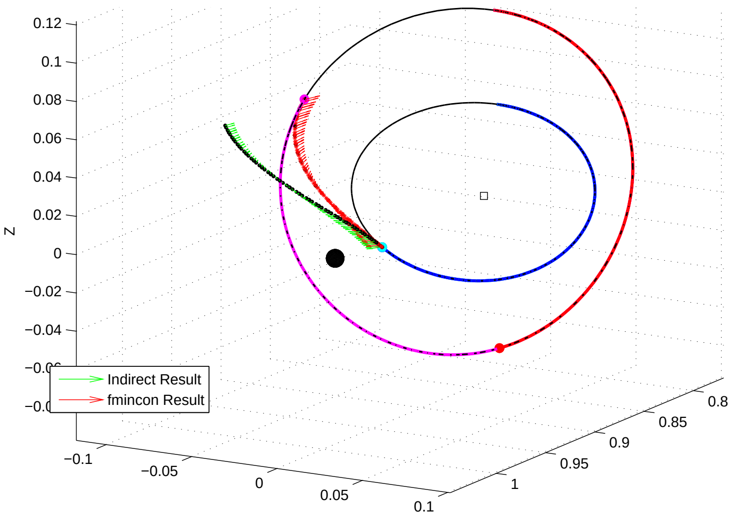


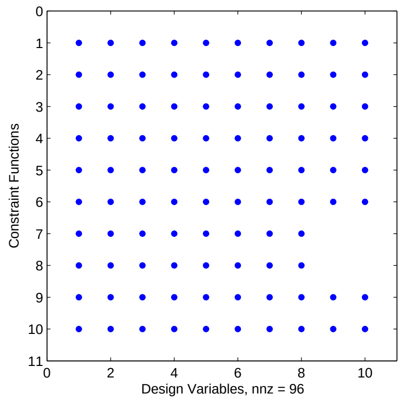
Figure 6.6.: Direct Optimization Solution Following ±40% Perturbation to the Initial Guess; Perturbed Trajectory (green) and Converged Solution (red)

Repeating the robustness test with the same perturbed initial guesses but applying a direct optimization method instead offers a contrast to the test results with the indirect optimization method. The perturbed initial guess for the direct optimization method is created by propagating and discretizing the perturbed initial guess for the indirect optimization method. The discretized trajectory produced from an initial guess perturbed by $\pm 40\%$ is shown in Figure 6.6, note that the same perturbed initial guess used to create Figure 6.5 is employed here. The discretization of the trajectory propagated from the perturbed initial guess is passed into the optimizer and the NLP formulation is solved. The direct optimization formulation converged to a solution at all perturbation levels tested, and in nearly every case the solution was approximately the same as the initial transfer produced in Chapter 4. For example, the transfer obtained with the direct optimization method when it is supplied with the initial guess displayed in Figure 6.6 is approximately the same as the original transfer. The converged transfer is also plotted in Figure 6.6 and requires approximately $20kg$ of propellant. The results offered in the final row of Table 6.1 indicate that the same robust behavior is demonstrated at every perturbation level tested, with only two runs at the $\pm 50\%$ failing to converge to the original solution. The general behavior indicated by Table 6.1 is that the final optimal solution obtained via the direct optimization approach is less susceptible to changes in the initial guess than the solution produced by the indirect optimization method. Overall, the results contained in Table 6.1 support the notion that the direct optimization method is more robust than the indirect optimization approach.

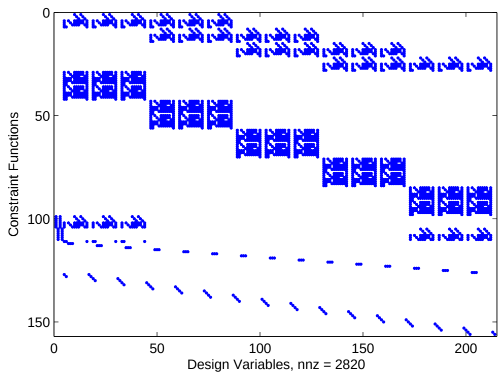
6.1.3 Computational Comparison

Direct optimization methods can be made to provide the same level of accuracy as indirect optimization methods, but doing so can come with high computational costs. The accuracy of direct optimization with collocation is increased by the number of segments in the discretization of the continuous optimal control problem and the order of the polynomial approximations. However, increasing these two parameters also increases the size of the NLP problem, and as a result the necessary computational time. The benefits afforded by direct optimization are negated if the time and computational resources required to solve the NLP are impractical. Unsurprisingly then, adoption of and research on direct optimization methods has increased in parallel with developments in computational technology. Widespread access to powerful computational resources has made direct optimization methods a practical alternative to indirect optimization, however poor implementation of these techniques can still lead to inefficient optimizers with steep computational costs.

The high dimensionality frequently encountered in direct optimization methods can slow down optimization algorithms. This is because the gradients required to obtain a solution are typically computed numerically, and a larger dimensioned problem requires that more gradients be computed. However, due to the discrete nature of the NLP many of the necessary gradients are equal to zero leading to highly sparse gradient matrices. The gradients of the constraint function with respect to the design variables form the Jacobian matrix, and this matrix for the halo to halo transfer using manifold arcs is shown in Figure 6.7. The 10×10 Jacobian matrix calculated in the indirect optimization method contains a total of 100 gradients, and only 4 are equal to zero. The entire matrix is calculated numerically because it has very few elements equal to zero, however because the dimension of the matrix is low this procedure requires a relatively small amount of computational time. The Jacobian matrix calculated in the direct optimization method is significantly larger with dimensions of 214×156 and a total of 33384 elements. Only 2820 of the Jacobian elements in the direct optimization case are nonzero, meaning that 91.55% of the elements in the matrix equal zero. Matrices where the vast percentage of elements equal zero are denoted sparse matrices. The gradients that occur in direct optimization methods tend to be sparse because many elements along a discretization are independent of one another. For example, the defect constraints of a segment in the collocation method are affected only by variables and parameters on that segment, therefore the partial



(a) Indirect Optimization



(b) Direct Optimization

Figure 6.7.: Sparsity Pattern of Jacobian

derivatives of these constraints with respect to variable on other segments are equal to zero. The sparsity of the gradients in direct optimization methods is leveraged to reduce the computational times of these techniques. Algorithms that track the indices of nonzero elements of gradients are developed so that only the values of these elements are computed while the remainder of the gradient elements are assumed to equal zero. When gradient sparsity is not leveraged in this way computational times for direct optimization methods can increase by several orders of magnitude. Therefore, leveraging sparsity decreases the number of computations required for large dimensioned problems, thereby decreasing the required computational time.

Many factors contribute to the computational time required to run a numerical algorithm. The efficiency of the code, accuracy of the initial guess, and hardware of the computer, all play a role, therefore it can be difficult to make one-to-one comparisons between the run times of different numerical algorithms. Nonetheless, a general comparison of the computational times of numerical methods can offer insight, especially if the times differ by several orders of magnitude. The average computational time required for ten runs of each optimization method applied to every sample problem studied is supplied in Table 6.2. Note, due to the relatively large computational time required to solve the circular-to-circular orbit transfer problem via direct optimization the time shown in Table 6.2 is for only one run of this problem.

Table 6.2.: Computational Time Comparison

Problem Type	Indirect Opt. (IO)	Direct Opt. (DO)	$DO = IO \times x$
Circular Orbit Transfer	4 sec	51523 sec	$IO \times 12881$ sec
L_1 Halo to L_1 Halo	9 sec	532 sec	$IO \times 59$ sec
L_1 Halo to L_2 Halo	158 sec	47 sec	$IO \times 0.3$ sec

The computational time of the indirect optimization method is less than that of the direct optimization method in two of the three cases examined, and in the first case it is significantly lower. Even when sparsity is leveraged the large dimensionality of the circular-to-circular orbit transfer problem appears to drastically increases the computational time. Another possible cause for the large computational time of the circular orbit transfer is that too few segments were utilized, this made honing in on an optimal solution more difficult for the optimizer. This possibility is supported by the results of section 6.1.1, where it was shown that the initial number of segments per revolution used for this problem was far too low to achieve an optimal solution that corresponded with the indirect optimization solution. Utilizing an insufficient number of segments to discretize the low-thrust transfer may create an extremely shallow basin of convergence, thus, making it difficult for an optimizer to converge to an optimal solution. Even when the initial state of the circular-to-circular orbit transfer is entirely fixed in an attempt to mitigate this problem the same behavior occurs, therefore, further investigation is needed to resolve this issue. However, the computational times for the other two transfer scenarios demonstrate that direct optimization methods can at least operate on a reasonable and sometimes even similar time-scale compared to indirect optimization methods. The small difference between the computational times for the halo to halo transfer case is almost certainly a result of algorithm implementation and not due to one method being inherently more efficient for this scenario. While well implemented indirect optimization methods are nearly always faster than direct optimization schemes, the latter technique does not necessarily require significantly more time.

6.2 Qualitative Comparison

Some of the differences between indirect and direct optimization methods are best captured by qualitative description. Numerous authors have produced surveys of direct and indirect optimization methods that discuss the advantages and disadvantages of each. Betts' 1997 survey [52] provides a thorough overview of trajectory optimization methods and is still frequently referenced. More recently authors such as Conway [49], Topputo and Zhang [53], and Rao [64] have contributed surveys of the field that include new trends, for example, discussion of genetic algorithms. The text *Spacecraft Trajectory Optimization* [48] provides an excellent primer on this diverse and expanding discipline.

Convergence of a numerical method does not guarantee an optimal solution has been identified, and the approach for ensuring the optimality of a solution depends on the optimization method used. The necessary conditions derived in Euler-Lagrange theory identify an extremal point and conditions such as Pontryagin's minimum principle, 4.8, determine whether the identified extremal is a minimum or a maximum. Other necessary conditions have been formulated to assess the optimality of solutions identified by the Euler-Lagrange theory, however generally these methods are assumed to 'guarantee' locally optimal solutions. Direct optimization methods do not offer this guarantee, thus other strategies to check the accuracy of their solutions are developed, one approach is to adapt the necessary conditions derived for indirect optimization methods to apply to direct optimization results. An alternate method is to perturb and reconverge the initial optimal solution, when the perturbed solution converges back to the original result this suggests the result is at least locally optimal. The robustness analysis conducted in section 6.1.2 is an example of this kind of testing, and the results of this test indicate the solution obtained is a local optimal. Therefore, while it is generally more simple to gauge the optimality of results from indirect optimization, strategies for judging solution optimality exist for both techniques.

The costates introduced in the derivation of the Euler-Lagrange theory are associated with the greatest advantages and disadvantages of the indirect optimization method. The costates indicate the sensitivity of the cost function to changes in their corresponding state variables. This information indicates the variables that have the greatest impact on the objective of an optimization problem, knowledge that is useful in the design and refinement of an optimal control problem. However, because sensitivity information is not directly associated with physical parameters such as length or mass, it can be difficult to determine an initial guess for the values of the costates. Unfortunately, convergence of the TPBVP formulated by the indirect optimization method is highly sensitive to the initial provided for the costates. This sensitivity was demonstrated by the robustness analysis conducted in section 6.1.2. Methods such as the adjoint control transformation help alleviate the difficulty of determining an initial guess for the costates by associating these variables with physical parameters. The challenge of determining an initial guess for the costates is eliminated completely by direct optimization methods which do not explicitly utilize costates to solve the optimal control problem. An additional benefit of eliminating the costates is that the dimension of the problem is reduced by two. The reduced problem size can be leveraged to decrease computational time depending upon the direct transcription method employed. Methods that avoid the use of costates from an optimal control problem are sought because these variables can complicate the problem setup, however, the costates can also provide useful insight into the relationships within a problem.

Continuous equations for the control variables are denoted control laws. In Chapter 4 the costates were employed to formulate Lawden's primer vector, a control law which indicates the direction of the thrust unit vector in the optimal control problem. The procedure for formulating the TPBVP in the indirect optimization method includes derivation of continuous control laws that are functions of the state and costate variables. Therefore, a solution to the TPBVP includes continuous control laws, and these equations are useful for assessing the optimal solution as well as nearby solutions. In contrast, direct optimization solutions provide values for the control variables at discrete points along a solution. While equations, for example polynomials, that approximate control values in between these points can be formulated they are not required in order to obtain a solution. It is possible to develop control laws that estimate the control history of optimal and nearby sub-optimal solutions, however these approximations can be very rough. When a control law is known an initial guess for the time history of the control law variables is still required regardless of the optimization method used. For example, if a control law is known for a CSI low-thrust transfer an initial guess for the pattern of thrust and coast arcs is necessary. Determining control law variables *a priori* can be challenging, therefore direct optimzation methods are sometimes preferred in these scenarios due to their wider basin of convergence. However, despite their disadvantages indirect optimization methods provide useful control and sensitivity information, this ensures that indirect optimization methods remain a viable approach to solving optimal control problems.

The control laws and costates derived for the setup of the TPBVP in indirect optimization methods are useful, however the extent of the derivations required to formulate the problem can be cumbersome, this is regarded as one of the primary weaknesses of the method. Applying the Euler-Lagrange necessary conditions to an optimal control problem requires the formulation of the Hamiltonian and the calculation of the gradient of the Hamiltonian with respect to the state, costate, and control variables. For relatively simple problems, such as the optimal VSI low-thrust transfer, these derivations are straightforward, however, they become more involved as the optimal control problem increases in complexity. Moreover, the TPBVP derived when the indirect optimization method is applied to an optimal control problem is specific to that problem. Changes more drastic than altering parameter values often require that the TPBVP be partially or fully rederived. For example, adding state and control variables to accommodate different propulsion systems, i.e. a solar sail, requires changes to the dynamic equations of motion of an optimal control problem. If indirect optimization is used, gradients for the new variables must be derived and the transversality condition is reapplied to derive additional boundary constraints. These alterations to the problem setup are incorporated with less drastic changes when direct optimization is used; in addition to updating the equations of motion, the design variable vector is expanded and extra constraints may be required. The ease of modification of the direct optimization formulation is indicative of the fact that direct optimization methods are also typically easier to implement. Fundamentally, the code for a direct optimization method requires functions for the objective and constraint equations as well as an initial guess. These three components are passed to an optimizer which uses them to converge upon an optimal solution. While ease of implementation is appealing, the apparent simplicity of direct optimization methods permits them to be applied in a haphazard manner that may ultimately slow down the process of finding an optimal solution. The ease of implementation and modification of direct optimization methods makes them a powerful technique, however the additional insight afforded by indirect optimization can make the method more effective in some scenarios.

The ease with which path constraints are incorporated is another measure of the flexibility of an optimization method. Path constraints are bounds applied along the course of a trajectory, for example a minimum altitude constraint for a planetary flyby. Tabular data, such as atmospheric conditions as a function of altitude may also be included in path constraints. Application of the indirect optimization method formulates a TPBVP, meaning the problem constraints are applied at the trajectories endpoints, posing the optimal control problem in this way is not conducive to applying path constraints. Direct optimization methods require the discretization of a continuous trajectory, and constraints can be applied at any of the discrete nodes. The discrete nature of direct optimization methods simplifies the process of incorporating constraints anywhere along the path of a trajectory.

One of the most difficult parts of applying any optimization method can be developing an initial guess, and while each optimization approach, direct and indirect, has distinct advantages and disadvantages both methods still require an initial guess. The robustness analysis conducted in Section 6.1.2 indicates that direct optimization methods generally exhibit a wider basin of convergence, therefore they can tolerate a less accurate initial guess. This property of direct optimization methods is especially useful when a solution space is unknown. An additional challenge in the process of formulating an initial guess is that both optimization techniques tend to converge towards a solution in the vicinity of the initial guess. The global optimal of a solution space is the most desirable solution, however the optimal solution nearest an initial guess is not necessarily globally optimal. Solution bias introduced by an initial guess is a challenge for both direct and indirect optimization methods. Techniques and other optimization methods, for example genetic algorithms, that address this weakness have been developed. And, these can be used in combination with either direct or indirect solutions to aid the process of locating the globally optimal solution.

7. CONCLUSION

7.1 Summary

The fundamental background required to pose and solve the optimal control problem is presented, with a focus on the dynamical features of the circular restricted three-body problem (CR3BP). Following this, the Euler-Lagrange theory that forms the basis for the majority of indirect optimization methods is presented. This theory is applied to the problem of obtaining orbital transfers with a low-thrust variable specific impulse engine (VSI). Differential and algebraic equations are derived for this problem and the resulting variables provide control laws that describe the behavior of control values throughout the transfer. A circular-to-circular orbit transfer in the two-body problem as well as two types of low-thrust halo to halo orbit transfers in the CR3BP are produced which maximize the final mass of the spacecraft at the end of the fixed transfer time. Configuration space plots of each transfer are plotted along with the histories of key variables and parameters. The background established early in the document provides the context for understanding the behavior of each locally optimal transfer obtained.

The low-thrust VSI transfers obtained with the indirect optimization method are discretized and used as an initial guess to reproduce the same transfers with direct optimization methods. The direct optimization approach used utilizes collocation to transcribe the continuous optimal control problem into a nonlinear programming problem (NLP). The collocation scheme employed uses odd order polynomials with Legendre-Gauss node spacing along with a mesh refinement technique to approximate a trajectory with acceptable integration accuracy. The low-thrust transfers obtained with the direct optimization and collocation approach appear to accurately approximate the results obtained by the indirect optimization approach. Comparison of the final masses obtained by each optimization method demonstrates the similarity of their results because the percent differences are on the order of thousandths of a percent, and this correspondence supports the conclusion that both methods produced the same result.

Finally, quantitative and qualitative comparisons of the results of each method are made to provide additional insight on the characteristics of each method. Quantitative comparisons based on the metrics of accuracy, robustness, and computational efficiency are made with the result that the indirect optimization method is demonstrated to be more computationally efficent, although not necessarily more accurate. However, the direct optimization method proves to posses greater robustness than the indirect optimization method because convergence is obtained even when larger perturbations are given to the initial guess. Qualitative assessment of both methods highlights additional strengths of each approach and stresses the ease of modification of direct optimization methods. Overall, the comparisons of both methods demonstrate the key traits of each approach.

7.2 Selecting a Method

Quantitative and qualitative analysis of indirect and direct optimization methods reveals the strengths and weaknesses of each approach. The strengths of one technique often address the deficiencies of the other, therefore no clear "best" optimization method can be chosen. However, general guidelines can be offered that guide the process of choosing an optimization strategy to apply to a particular optimal control problem. Formation of these guidelines is informed by the results of the implementation and analysis of each optimization method.

The speed of indirect optimization methods, demonstrated by the results of Chapter 6, make these methods efficient at quickly determining optimal trajectories within a specific problem context. Several types of low-thrust VSI transfers are obtained by the indirect optimization method with relatively minor changes required for each transfer scenario. The costate variables introduced by the indirect optimization method readily offer information on the behavior of the control values throughout each transfer, and provide dynamical insight on the behavior of trajectories in the vicinity of the optimal solution. Satisfaction of the Euler-Lagrange necessary conditions and the shooting method utilized to solve the TPBVP together offer confidence that the solution is an accurately propagated local optimal. For these reasons, the indirect optimization solution is used as the baseline in Chapter 6's accuracy comparison. Furthermore, the rapid computational time exhibited by the indirect optimization method in comparison with the direct optimization method demonstrates the relative speed numerical implementations of the indirect method are capable of. For this reason the software tools initially developed to design and fly low-thrust spacecraft missions were based on indirect optimization methods. The efficiency of the indirect optimization method makes it an excellent option for solving certain types of optimal control problems. Specifically, when an optimal control problem is unlikely to change significantly and some intuition about the problem is known, for example the circular orbit raising problem, indirect optimization methods offer a powerful solution approach for spacecraft optimal control problems.

The relative complexity of the derivations required to formulate the TPBVP and the poor robustness of the indirect optimization method make it a less appealing option when there is little intuition about an optimal control problem. The strengths of the indirect optimization method are evident when the optimal control problem is well formulated and a good initial guess is provided, however the testing in Chapter 6 demonstrates that the method lacks robustness. The efficiency of the method is only evident when the indirect optimization method converges which is difficult when the initial guess is perturbed by a significant percentage. The poor robustness of the indirect optimization method limits its ability to find local optimal solutions in other basins of convergence, although this capability could be improved by implementing a multiple shooting rather than a single shooting scheme. Additionally, formulation of the low-thrust VSI optimal control problem using Euler-Lagrange theory is an involved process and significantly altering the problem, for example adding new boundary or path constraints, will likely require significant rederivation. The deficiencies of the indirect optimization method demonstrated in Section 6.1.2 can be severely limiting, making this approach a poor fit for scenarios where little intuition about a problem is known.

When a solution space is unfamiliar the robustness and flexibility of direct optimization methods make these schemes effective tools for gaining insight on the range of possible optimal solutions. The robustness of the direct optimization method is demonstrated in Section 6.1.2 when convergence is obtained for even very poor initial guesses. The fact that the direct optimization method can converge to a locally optimal solution even when the initial guess is directed opposite to the target in configuration space, as shown in Figure 6.6 evidences the versatility of this method and how it can be applied in problems where good initial guesses are not available. Furthermore, the constraints detailed in Section 5.5 are easily incorporated into the direct optimization scheme and this ease of modification is also desirable when a problem is not yet well defined. An example of a less well understood solution space could be the convergence of low-thrust periodic orbits about artificial equilibrium points. This problem and others on the frontiers of trajectory design offer excellent opportunities for the application of robust and adaptable direct optimization methods.

7.3 Future Work

A variety of avenues for further investigation are immediately apparent. Comparison of the indirect and direct optimization methods shows that the weaknesses of one method are often met by the strengths of the other. This complementary nature suggests that research into strategies for combining the best aspects of both methods may yield even more profitable optimization schemes. Indeed, Betts [52] and Conway [48] writing over a decade apart have both noted this potential trend. Some researchers have already demonstrated how methods such as collocation can be utilized to facilitate the process of obtaining an initial guess for the costates [65]. Alternately, the necessary conditions of the indirect optimization method have been used to verify the optimality of results obtained via direct optimization method. Regardless of the route taken a variety of opportunities for combining the best features of both approaches are available and may yield even more efficient methods obtaining optimal trajectories.

The VSI low-thrust engine model utilized in this study is numerically useful, however the technology readiness level of VSI engines is still low and all low-thrust spacecraft currently in flight utilize CSI engines. The CSI optimal control problem is more challenging to solve due to the existence of thrust and coast arcs, and this is especially true for the indirect method because the desired sequence of thrust and coast arcs must be a part of the initial guess. The direct optimization method does not require that a control structure be specified *a priori*, therefore demonstrating this difference by the inclusion of a CSI optimal control problem would strengthen the final assessment of both methods.

Spiral trajectories are ubiquitous in low-thrust mission design problems, therefore more efficient direct optimization methods for solving this type of problem must be developed. In this study polar coordinates were employed to solve the circular-to-circular orbit transfer problem, however may researchers use alternate coordinate systems, for example equinoctial elements, in order to solve this type of problem. Reformulating, the current circular orbit transfer problem in a different coordinate system and including motion in the out-of-plane direction may improve the convergence properties of the direct optimization approach. Moreover, spiraling trajectories are often only a phase in the overall flight plan of a low-thrust spacecraft, therefore the capability to connect each phase and optimize the whole is necessary. Ozimek [54], among other researchers, has demonstrated solutions to this problem. The ability to optimize spiraling low-thrust trajectories is an essential part of any comprehensive low-thrust trajectory design strategy.

The robustness demonstrated by the direct optimization with collocation method, will offer a powerful tool for exploring unfamiliar solution spaces for low-thrust transfers. Transfers in the CR3BP utilizing manifold arcs offer a particularly intriguing and useful area for investigation. The manifolds of natural periodic orbits, or perhaps of artificial periodic orbits maintained by a low-thrust engine, could unlock a vast new solution space and enable previously infeasible missions. The first step towards this goal is to employ the direct optimization method to produce artificial periodic orbits in the CR3BP. Relatively few solutions for this type of orbit are known at present, thus the robustness of the direct optimization method makes it well-suited for addressing this type of problem.

A multitude of challenging optimal control problems are found in the discipline of spacecraft trajectory design, particularly when the focus is low-thrust spacecraft. No single trajectory optimization method is capable of solving all of these problems, rather the unique strengths and weaknesses of a variety of optimization methods must be brought to bear on the problems each method is best suited for. At times the complementary nature of different combinations of optimization methods may also provide unique solutions to optimal control problems. A deep understanding of the theory and characteristics of each optimization method is necessary, this knowledge will ensure the efficient application of optimization techniques to any of the many available intriguing problems.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Issac Newton. *The Principia: Mathematical Principles of Natural Philosophy.* University of California Press, Berkely California, 1999.

[2] George William Hill. *Collected Mathematical Works of G. W. Hill*, volume 1. Carnegie Institution of Washington, Washington, 1905.

[3] Henri Poincaré. *Les Methodes Nouvelles de la Mecanique Celeste*, volume 2. Gauthier-Villars, Paris, 1893.

[4] George David Birkhoff. *Collected Mathematical Papersl*, volume 1. American Mathematical Society, New York, 1950.

[5] Victor Szebehely. *Theory of Orbits.* Academic Press, New York, 1967.

[6] James M. Longuski, José J. Guzmán, and John E. Prussing. *Optimal Control with Aerospace Applications.* Microcosm Press and Springer, El Segundo, California, 2014.

[7] Derek F. Lawden. *Optimal Trajectories for Space Navigation.* Butterworths, London, 1963.

[8] Arthur Bryson and Yu-Chi Ho. *Applied Optimal Control.* Blaisdell Publishing Company, Waltham, Massachusetts, 1969.

[9] G. Mingotti, F. Topputo, and F. Bernelli-Zazzera. Combined optimal low-thrust and stable-manifold trajectories to the earth-moon halo orbits. In *AIP Conference Proceedings*, volume 886, pages 100–112, May 2007.

[10] Jeff R. Stuart, Martin T. Ozimek, and Kathleen C. Howell. Optimal, low-thrust, path-constrained transfers between libration point orbits using invariant manifolds. In *AIAA/AAS Astrodynamics Specialist Conference*, Toronto, Ontario, 2010.

[11] Michael D. Canon, Clifton D. Cullum, and Elijah Polak. *Theory of Optimal Control and Mathematical Programming.* McGraw-Hill, New York, New York, 1970.

[12] Ernst D. Dickmanns and Klaus H. Well. Approximate solution of optimal control problems using third-order hermite polynomial functions. In *Proceedings of the 6th Technical Conference on Optimization Techniques*, New York, 1975. IFIP-TC7, Springer-Verlag.

[13] Charles R. Hargraves and Stephen W. Paris. Direct trajectory optimization using nonlinear programming and collocation. *Journal of Guidance, Control, and Dynamics*, 10(4):338–342, 1987.

[14] Carl-Wilhelm de Boor. *The Method of Projections as Applied to the Numerical Solutions of Two Point Boundary Value Problems Using Cubic Splines.* Ph.d. dissertation, University of Michigan, Ann Arbor, Michigan, 1966.

[15] R. D. Russell and L. F. Shampine. A collocation method for boundary value problems. *Numerische Mathematik*, 28:1–28, 1972.

[16] Richard Weiss. The application of implicit runge-kutta and collocation methods to boundary-value problems. *Mathematics of Computation*, 28(126):449, 1974.

[17] Bruce A. Conway Paul J. Enright. Discrete approximations to optimal trajectories using direct transcription and nonlinear programming. *Journal of Guidance, Control, and Dynamics*, 15(4):994–1002, 1992.

[18] Albert L. Herman and Bruce A. Conway. Direct optimization using collocation based on high-order gauss-lobatto quadrature rules. *{AIAA} Journal of Guidance, Control, and Dynamics*, 19(3):522–529, 1996.

[19] Paul Williams. Hermite-legendre-gauss-lobatto direct transcription in trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 32(4):1392–1395, 2009.

[20] John T. Betts and William P. Huffman. Mesh refinement in direct transcription methods for optimal control. *Optim. Control Appl. Meth*, 19(April 1997):1–21, 1998.

[21] John T. Betts and William P. Huffman. Sparse optimal control software socs. Technical Report Rept. MEA-LR-085, The Boeing Co., Seattle, Washington, 1997.

[22] Charles R. Hargraves and Stephen W. Paris. Optimal trajectories by implicit simulation (otis). (AFWAL-TR-88-3057), 1988.

[23] A. M. Leontovic. On the stability of the lagrange periodic solutions for the reduced problem of three bodies. *Dokl. Akad. Nauk USSR*, 143:525, 1962.

[24] P. Gill, W. Murray, M. Saunder, and M. Wright. *Practical Optimization.* Academic Press, London, England, 1991.

[25] Joaquim R. R. A. Martins, Peter Sturdza, and Juan J. Alonso. The complex-step derivative approximation. *ACM Transaction on Mathematical Software (TOMS)*, 29(3):245–262, 2003.

[26] David Garza. *Application of Automatic Differentiation to Trajectroy Optimization via Direct Multiple Shooting.* Ph.d. dissertation, University of Texas, Austin, Texas, 2003.

[27] Andreas Griewank. *Evaluating Derivatives Principles and Techniques of Algorithmic Differentiation.* SIAM, Philadelphia, Pennsylvania, 2 edition, 2008.

[28] Richard D. Neidinger. Introduction to automatic differentiation and matlab object-oriented programming. *SIAM Review*, 52(3):545–563, 2012.

[29] A. E. Roy and M. W. Ovenden. On the Occurence of Commensurable Mean Motions in the Solar System: II. The Mirror Theorem. *Monthly Notices of the Royal Astronomical Society*, 115(3):296–309, 1955.

[30] Daniel J. Grebow. *Generating Periodic Orbits in the Circular Restricted Three-Body Problem with Applications to Lunar South Pole Coverage.* M.s. dissertation, Purdue University, West Lafayette, Indiana, 2006.

[31] Amanda F. Haapala. Trajectory design using periapse maps and invariant manifolds. M.s. dissertation, Purdue University, West Lafayette, Indiana, 2010.

[32] D.G. Hull. *Optimal Control Theory for Applications.* Springer, New York, 2003.

[33] S. J. Citron. *Elements of Optimal Control.* Holt, Rinehart, and Winston, New York, 1969.

[34] L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze, and E.F. Mishchenko. *The Mathematical Theory of Optimal Processes.* Wiley, New York, 1962.

[35] Ryan P. Russell. Primer vector theory applied to global low-thrust trade studies. *Journal of Guidance, Control and Dynamics*, 30(3):460–472, 2007.

[36] Gao Tang and Fanghua Jiang. Capture of near-earth objects with low-thrust propulsion and invariant manifolds. *Astrophysics and Space Science*, 361(1), 2016.

[37] P. Di Lizia M Rasotto, R. Armellin. Multi-step optimization strategy for fuel-optimal orbital transfer of low-thrust spacecraft. *Engineering Optimization*, 48(3):519–542, 2016.

[38] Chris L. Ranieri and Cesar A. Ocampo. Optimization of roundtrip, time-constrained, finite burn trajectories via an indirect method. *Journal of Guidance, Control, and Dynamics*, 28(2):306–314, 2005.

[39] Walter Hohmann. *The Attainability of Heavenly Bodies*, volume F-44 of *NASA Technical Translation.* National Aeronautics and Space Administration, Washington, DC, 1960.

[40] James E. Pollard. Evaluation of low-thrust orbital maneuvers. In *34th AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, Cleveland, Ohio, 1998.

[41] Amanda F. Haapala and Kathleen C. Howell. Trajectory design using poincaré maps and invariant manifolds. In *Proceedings of the 21st AAS/AIAA Space Flight Mechanics Meeting*, New Orleans, LA, 2011.

[42] Juan Senent, Cesar Ocampo, and Antonio Capella. Low-thrust variable-specific-impulse transfers and guidance to unstable periodic orbits. *Journal of Guidance, Control and Dynamics*, 28(2):280–290, 2005.

[43] L.C. W. Dixon and M.C. Bartholomew-Biggs. Adjoint-control transformations for solving practical optimal control problems. *Optimal Control Applications and Methods*, 2:365–381, 1981.

[44] Hans Seywald, Carlos M. Roithmayr, Patrick Troutman, and Sang-Young Park. Fuel-optimal transfers between coplanar circular orbits using variable-specific-impulse engines. *Journal of Guidance, Control and Dynamics*, 28(4):795–800, 2005.

[45] Jon A. Sims Anastasios .E. Petropoulos. A review of some exact solutions to the planar equations of motion of a thrusting spacecraft. In *2nd International Symposium Low Thrust Trajectories*, Toulouse, France, 2002.

[46] Martin T. Ozimek. A low-thrust transfer strategy to earth-moon collinear libration point orbits. M.s. dissertation, Purdue University, West Lafayette, Indiana, 2006.

[47] Jeffrey R. Stuart. Fuel-optimal, low-thrust transfers between libration point orbits. M.s. dissertation, Purdue University, West Lafayette, Indiana, 2011.

[48] Bruce A. Conway. *Spacecraft Trajectory Optimization.* Cambridge University Press, New York, New York, 2010.

[49] Bruce A. Conway. A survey of methods available for the numerical optimization of continuous dynamic systems. *Journal of Optimization Theory and Applications*, 152(2):271–306, 2012.

[50] Dimitri P. Bertsekas. *Constrained Optimization and Lagrange Multiplier Method.* Athen Scientific, Belmont, 1996.

[51] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, New York, 2004.

[52] John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1997.

[53] F. Topputo and C. Zhang. Survey of direct transcription for low-thrust space trajectory optimization with applications. *Abstract and Applied Analysis*, 2014:15, 2014.

[54] Martin T. Ozimek, Daniel J. Grebow, and Kathleen C. Howell. A collocation approach for computing solar sail lunar pole-sitter orbits. *The Open Aerospace Engineering Journal*, 3, 2010.

[55] Daniel J. Grebow and Thomas A. Pavlak. Ammos report on collocation methods. Technical report, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 2015.

[56] R. D. Russell and J. Christiansen. Adaptive mesh selection strategies for bounday value problems. *SIAM Journal of Numerical Analysis*, 15(2):59–80, 1978.

[57] Carl-Wilhelm de Boor. Good approximation by splines with variable knots ii. In *Conference on the Numerical Solution of Differential Equations*, volume 363 of *Lecture Notes in Mathematics*, pages 12–20, New York, 1973. Springer.

[58] Daniel J. Grebow and Thomas A. Pavlak. Ammos report on mesh refinement for collocation methods. Technical report, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California, 2015.

[59] Richard H. Byrd, Mary E. Hribar, and Jorge Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 1999.

[60] Richard H. Byrd, Jean Charles Gilbert, and Jorge Nocedal. A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming*, 89(1):149–185, 2000.

[61] Robert D. Falck and John W. Dankanich. Optimization of low-thrust spiral trajectories by collocation. Technical report, Glenn Research Center, Cleveland, Ohip, 2012.

[62] Wayne A. Scheel and Bruce A. Conway. Optimization of very-low-thrust, many-revolution spacecraft trajectories. *Journal of Guidance, Control and Dynamics*, 17(6):1185–1192, 1994.

[63] Kathryn F. Graham and Anvil V. Rao. Minimum-time trajectory optimization of multiple revolution low-thrust earth-orbit transfers. *Journal of Spacecraft and Rockets*, 52(3), 2015.

[64] Anvil V. Rao. Trajectory optimization: A survey. In Harald Waschl, Ilya Kolmanovsky, Maarten Steinbuch, and Luigi del Re, editors, *Optimization and Optimal Control in Automotive Systems*, chapter 1, pages 3–21. Springer, The address of the publisher, 2014.

[65] O. von Stryk and R. Bulirsch. Dirrect and indirect methods for trajectory optimization. *Annals of Operations Research*, 37:357–373, 1992.