Optimal Orbital Transfer Using a Legendre Pseudospectral Method

by

Stuart Andrew Stanton

B.S. Astronautical Engineering
United States Air Force Academy, 2001

SUBMITTED TO THE DEPARTMENT OF AERONAUTICS AND ASTRONAUTICS IN
PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN AERONAUTICS AND ASTRONAUTICS
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

JUNE 2003

Signature of Author

Department of Aeronautics and Astronautics
May 23, 2003

Certified by

Christopher N. D'Souza, Ph.D.
The Charles Stark Draper Laboratory, Inc.
Thesis Supervisor

Certified by

Ronald J. Proulx, Ph.D.
The Charles Stark Draper Laboratory, Inc.
Thesis Supervisor

Certified by

George T. Schmidt, Sc.D.
Lecturer, Department of Aeronautics and Astronautics
Director, Education
The Charles Stark Draper Laboratory, Inc.
Thesis Advisor

Accepted by

Edward M. Greitzer, Ph.D.
H.N. Slater Professor of Aeronautics and Astronautics
Chair, Committee on Graduate Students

[Except for this sentence, this page intentionally left blank.]

Optimal Orbital Transfer Using a Legendre Pseudospectral Method

by

Stuart A. Stanton

Submitted to the Department of Aeronautics and Astronautics
on May 23, 2003, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

Orbital transfer problems are solved successfully for several classes of transfers, including coplanar transfer, simple plane change, and complex orbital size/shape/plane changes. Both impulsive and finite-burn orbital transfers are constructed which minimize fuel costs. Using the direct Legendre pseudospectral technique software package (DIDO) developed by Ross and Fahroo of the Naval Postgraduate School, the effects of complex perturbations (like J_2) are naturally included in the optimal solution. Written in Matlab, the DIDO package allows the user to incorporate complex cost functions, bounds, and constraints on the optimal transfer trajectory through a standardized interface.

Thesis Supervisor: Christopher N. D'Souza, Ph.D.
Title: The Charles Stark Draper Laboratory, Inc.

Thesis Supervisor: Ronald J. Proulx, Ph.D.
Title: The Charles Stark Draper Laboratory, Inc.

Thesis Advisor: George T. Schmidt, Sc.D.
Title: Lecturer, Department of Aeronautics and Astronautics
Director, Education
The Charles Stark Draper Laboratory, Inc.

[Except for this sentence, this page intentionally left blank.]

ACKNOWLEDGMENT
May 23, 2003

Stuart A. Stanton, 2Lt., USAF

5

[Except for this sentence, this page intentionally left blank.]

Contents

List of Figures

[Except for this sentence, this page intentionally left blank.]

List of Tables

Chapter 1

Introduction

1.1 Motivation

Even before artificial satellites were in space, the concept of getting from one point in space to another was of interest to those in the astrodynamics community. For example, the desire to understand the orbits that connect two points dates back centuries. Until the launch of Sputnik in 1957, however, orbital maneuvering was not even a practical problem. Now, the ability to maneuver in space is a necessity.

Clearly, a satellite must operate in a specific orbit to accomplish its mission; a communications satellite may require a geostationary orbit, a navigation satellite a semi-synchronous orbit, or a reconnaissance satellite a sun-synchronous, low earth orbit. Rarely, however, is a satellite launched directly into its operating orbit. Generally, the launch vehicle places the satellite in a parking orbit, and then the satellite performs a series of maneuvers to get to the orbit necessary for the mission.

Any number of cases can be described in which some sort of maneuver is necessary to transfer a satellite from one orbit to another. Like any trajectory problem, though, some trajectories that solve the problem are better than others. Limitations in technology and time motivate an understanding of what constitutes a good trajectory. If a satellite should be moved from one orbit to another, it is important to understand the trajectories that minimize the time or the fuel expended to complete the transfer. In space, a vehicle's resources are limited like nowhere else. Thus, it is crucial to ask the question, What is the optimal trajectory to transfer from orbit A to orbit B?

In this thesis, a new set of computational tools are used to answer this question. These tools apply an extremely accurate direct optimization technique. Minimum fuel transfer problems are solved under several different conditions. Transfers employing both impulsive and finite-burn maneuvers are considered.

15

1.2 Background

A large amount of work has been accomplished over the last forty or more years in the realm of orbital transfer optimization, using both direct and indirect methods. The following is intended to recognize some of the research that has happened in this field, with more emphasis on that of the last ten (or so) years. This summary of contributions is far from exhaustive, but should provide some insight into the context of this thesis.

Early researchers focused on calculus of variation approaches to design optimal trajectories. Lawden used primer vector theory in 1963 for minimum-fuel trajectory optimization problems [25]. Prussing and Chiu used the primer vector to find the optimal number of impulses for time-fixed transfer between circular orbits [32]. Early work by Edelbaum also focused on understanding the number of impulses that optimize an orbital transfer [12].

London showed in 1962, that for certain orbital maneuvers, the use of aerodynamic forces may lead to transfer solutions that minimize fuel better than pure propulsive maneuvers [27]. This discovery has inspired a substantial amount of research in what has been termed aeroassisted orbital transfer (AOT). Both indirect and direct methods have been used to investigate such problems. Naidu has used an indirect, multiple shooting method on an impulsive thrusting AOT scenario [30, 31]. Miele, Wang, and Lee used a sequential gradient-restoration algorithm on a similar problem [29]. In 2001, Baumann examined finite-burn AOTs with an indirect, multiple shooting method [2].

Kechichian has explored the constant acceleration, minimum-time orbital transfer problem using various coordinate systems. In 2000, he presented work in trajectory optimization using the nonsingular equinoctial orbit elements as coordinates, and the polar (Euler-Hill) frame for thrust perturbation vector resolution. Finding greater success replacing the mean longitude, λ, with the true longitude, L, this formulation was then used in an augmented system including the first-order oblateness effect, J_2 [21].

Redding and Breakwell [33]; Zondervan, Wood and Caughey [51]; Spencer and Culp [40]; Matogawa [28]; Ilgen [20]; and Sackett, Malchow, and Edelbaum [37] have considered solar electric propulsion (SEP) in minimum-fuel and minimum-time orbital transfer problems. Both direct and indirect methods have been used to solve these continuous-thrust problems. Thorne and Hall solved 2- and 3-dimensional problems indirectly by developing approximate models for initial co-states and time of flight for minimum-time transfers [42]. More recently, Kluever and Oleson have considered the effects of Earth shadowing, oblateness, and solar cell degradation on the SEP problem by applying a direct method using the sequential quadratic programming code, NPSOL [24]. Thorvaldsen, Proulx, and Ross recently applied pseudospectral techniques to study SEP transfer between Earth and Mars [43].

For shooting methods to solve the two point boundary value problem, guesses

16
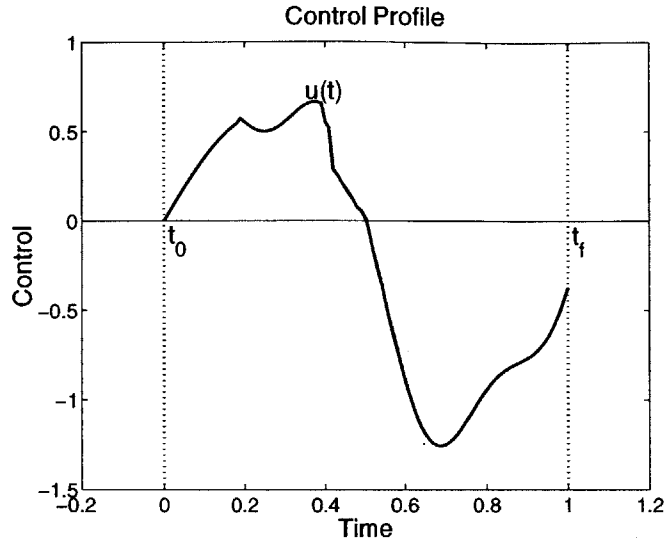
Figure 1-1: Continuous Controls

are required for the states, co-states, and controls. The co-states, with no physical meaning, are difficult to guess because one cannot have a clear intuition of what their values should be. Many have addressed this issue, including Dixon and Biggs, who presented the adjoint-control transformation method for dealing with these guesses, which was applied to optimal orbital transfer by Yan and Wu [11, 50].

Chuang, Goodson, and Ledsinger applied the conditions that come from the second variation of the cost functional for a minimum-fuel transfer problem. The second-order sufficient conditions were used in a neighboring optimal feedback guidance scheme [7].

1.3 Thesis Overview

The orbit transfer problem is an optimal control problem: a vehicle at a set of initial conditions (an initial orbit and location defined by the initial terminal constraints) must be moved to a different orbit (identified by the final terminal constraints) by some control profile. A performance index, J, must be minimized: here we minimize the accumulated effort, which will ultimately lower design and mission costs.

Generally, the controls of an optimal control problem are presented continuously; they are a function of time, $u(t)$ (see Figure 1-1). To minimize effort, a performance index would generally include a term that integrates the magnitude of the control.

$$J = \int_{t_0}^{t_f} |u(t)| \, dt$$

17
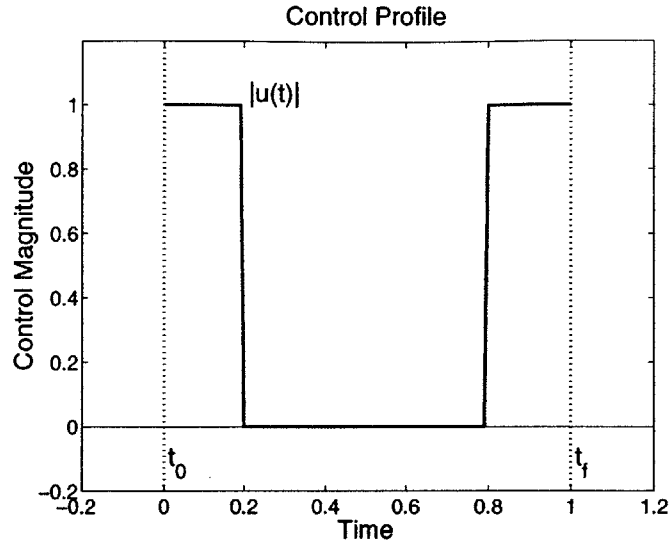
Figure 1-2: "Bang-Off-Bang" Continuous Controls

Orbital transfer may be presented as a constrained control problem: technology limitations will constrain the thrust acceleration available to a vehicle. The typical solution of a minimum-fuel, constrained-control problem will have what is called a "bang-off-bang" control profile, demonstrated in Figure 1-2. In the orbital transfer problem, the number of thrusting arcs and their duration will depend on the problem specifics.

In Chapter 2, the optimal control problem is outlined. The theory is developed by stepping through concepts like function and functional minimization. Optimal control problems with constrained control are presented, as well. However, most optimal control problems are difficult, if not impossible, to solve analytically; the orbital transfer problem is no exception. Two classes of computational methods exist for approaching these problems: indirect methods and direct methods.

This thesis applies a direct pseudospectral technique to solve general orbital transfer problems. Like other direct methods, it requires states and controls to be discretized in order to approximate a solution. By applying the theory of Gaussian quadrature to the development of a differentiation matrix, the pseudospectral method places the nodes at the basis roots to increase accuracy. The Legendre-Gauss-Lobatto Pseudospectral Method places the nodes at the roots of the derivatives of the Legendre polynomials and at the endpoints. We use DIDO, a MATLAB optimization toolbox developed by Fahroo and Ross which implements their direct Legendre Pseudospectral method. Elnagar, et. al. first applied the direct pseudospectral method to the linear quadratic control problem [13]. Since then, it has been further developed by Fahroo and Ross, who have documented the fundamentals of their pseudospectral

method in a series of recent papers [14, 16, 34, 35, 36], with active research underway. DIDO is a powerful package which allows a user to solve both continuous and impulsive transfer problems in a common environment. It transcribes the continuous optimal control problem into a discrete NLP problem, which can be solved using an NLP solver. The version of DIDO employed in this work uses SNOPT [17].

The basic theory behind the Legendre Pseudospectral Method is outlined in Chapter 3. The user interface to DIDO is briefly outlined in Chapter 4. Finally, before presenting results, how DIDO is used for the orbital transfer problem is discussed in Chapter 5.

Two classes of orbital problems are solved in this thesis: impulsive and finite-burn. Quite often, astrodynamicists are interested in an impulsive solution to an orbital transfer problem because it represents a compact and convenient approximation to a high-thrust solution. For problems in which the ratio is small between the actual maneuver duration and the orbital period, an impulsive solution may be a reasonably accurate approximation. Methods have been used previously to find optimal impulsive solutions, most noteably are the methods employing primer vectory theory.

In Chapter 6, the impulsive problem is solved using the direct pseudospectral method which is used in DIDO. Primarily, the directions, magnitudes, and times of impulsive maneuvers are found by optimizing static parameters. With this parameter optimization capability, the continuous controls desribed above are not included. Instead, the controls are represented purely by parameters.

For impulsive solutions, however, the continous control solution is shown to still have merit. For problems that may be difficult to solve, an impulsive-approximation solution consisting of high-magnitude, bang-off-bang controls may serve to provide insight into the solution form, paving the way to find an exact impulsive solution.

In Chapter 7, the finite-burn problem is explored using DIDO to solve for nominally continuous, constrained controls. Here, we solve a potentially more realistic problem: in real-world scenarios, finite-burn control must always be used, as chemical thrusters cannot produce impulsive maneuvers.

The impulsive solution represents an idealization: Δv maneuvers appear to be more efficient (less costly in terms of total Δv) than finite burns, but the impulsive solution cannot be realized. The cost of the impulsive solution can be approached by limiting the duration of finite burns (which generally requires less strict constraints on thrust magnitude). However, this is not to suggest that an impulsive solution is more cost effective than a finite-burn solution, since an impulse cannot actually be applied. It is demonstrated in Chapter 7 that it is far from optimal to implement an impulsive solution with finite burns. Therefore, the finite-burn capability of solving directly for finite-burn solutions is extremely valuable. Since there must always be constraints on thrust capabilities, finding the optimal, finite-thrust, time-bounded solution is far more beneficial than a "cheaper" impulsive solution.

In both Chapters 6 and 7, results are presented for a series of transfer scenarios. Some of the transfers are quite standard: they are the problems that can be solved

Academic Transfers

Hohmann Transfer

Circular-Circular
Elliptical-Circular
Elliptical-Elliptical

Plane Change

Circular: Inclination Change
Circular: Inclination &
  Ascending Node Change
Circular: Nodal Regression
General Elliptical: Inclination
  Change

Combined Plane Change
(Orientation and Energy)

Circular-Circular: Inclination
  Change
Elliptical-Elliptical: Inclination
  Change

Practical Transfers

LEO (ISS)-LEO (Sun-Synchronous)
LEO (ISS)-Molniya

Figure 1-3: Transfer Series

using basic equations from astrodynamics. The impulsive and finite-burn capabilities are also applied to complicated problems, where the terminal constraints are gathered from existing satellite ephemeris data. A summary of the kinds of problems examined in this thesis are presented in Figure 1-3.

Chapter 2

Optimal Control Theory

Necessary to understanding the work in this thesis towards solving orbital transfer problems is a basic knowledge of the theory of optimal control. The objective of optimal control theory is to find the set of controls that cause a system to meet all of its specified constraints while minimizing (or maximizing) a performance criterion, or cost function [22, p. 3].

It would be an enormous undertaking to attempt to develop the fundamentals of optimal control in this chapter. Instead, the intention of this chapter is to suggest how the theory is developed by looking at smaller concepts, and building up to the concepts of optimal control. Further depth can be found in any of a number of different optimal control texts, including [5] or [22].

Since the task of an optimal control problem is to minimize a performance index, basic function minimization is addressed first. Reviewing some basics of ordinary calculus, one is reminded how to find the minimum of a function. Then, we introduce constraints to see how that changes the minimization problem. Next, functional minimization is described. The calculus of variations is employed first for unconstrained problems, and then for constrained problems. The basic concepts of function and functional minimization can be applied directly to the optimal control problem. Since the derivations are long, the section is limited to simply presenting the problem, along with the necessary conditions for optimality. After introducing the optimal control problem in a general form, Pontryagin's Minimum Principle will be discussed for problems with constrained control.

2.1   Function Minimization

Implied above is the fact that optimal control involves minimizing (or maximizing) something: perhaps the amount of time or fuel required to get from point A to point B, or maybe a combination of different quantifiable factors. Therefore, it is reasonable to begin by reviewing the simpler function minimization concepts familiar from ordinary calculus. This will serve to introduce some of the terminology and

solution methodology that is employed in optimal control.

Start with a continuous function, $J(\vec{x})$. The variable, J, is chosen to represent the function to be minimized, as J is traditionally used as the symbol to represent the cost function to be minimized in an optimal control problem. The argument, $\vec{x}$ [1], is a vector, allowing multiple inputs to affect the output of J, even though J, itself, is a scalar. A local minimum exists at the point $\vec{x}^*$ if the following statement is true:

$$J(\vec{x}) \geq J(\vec{x}^*) \text{ for all } \vec{x} \text{ in the neighborhood of } \vec{x}^*. \tag{2.1}$$

A global minimum exists at $\vec{x}^*$ if

$$J(\vec{x}) \geq J(\vec{x}^*) \text{ for all admissible values of } \vec{x}. \tag{2.2}$$

A minimum satisfying one of these statements can exist at any of three places:

1) a state $(\vec{x})$ boundary,

2) a point of discontinuous derivative in $\vec{x}$, or

3) a stationary point.

The rest of the discussion focuses on finding local minima at stationary points. From Equation 2.1, a local minimum is found by looking in the neighborhood, $\vec{x}^* + \Delta\vec{x}$, of the minimum. From the expansion of $J(\vec{x}^* + \Delta\vec{x})$ around $\vec{x}^*$, it is evident that the necessary conditions for a stationary point to be a minimum are

$$\frac{d}{d\vec{x}}J(\vec{x}^*) \;=\; 0 \text{ (First order necessary condition)} \tag{2.3}$$

$$\frac{d^2}{d\vec{x}^2}J(\vec{x}^*) \;\geq\; 0 \text{ (Second order necessary condition)}. \tag{2.4}$$

This result should be familiar from calculus: an extreme exists where the first derivative is zero. (For a minimum, the function's second derivative must be nonnegative.)

A distinction should be made between *necessary* and *sufficient* conditions.

If condition A is *necessary* for condition B, then if A is not true, B cannot not true.

If condition A is *sufficient* for condition B, then if A is true, B must be true.

For example, Equations 2.3 and 2.4 do not guarantee that a minimum exists when they are satisfied, but these conditions must be satisfied if a minimum does exist.

---

[1] For consistency throughout this thesis, the vector arrow in $\vec{x}$ or $\vec{x}(t)$ denotes multiple variables contained in a single vector. The boldface vector **x** will denote a discretization of the single continuous variable $x(t)$. The capitalized boldface **X** is a matrix, which would result from the discretization of $\vec{x}(t)$.

The second order *sufficient* condition for a minimum is that $\frac{d^2}{d\vec{x}^2}J(\vec{x}^*) > 0$ if the first order condition is met. That is, the $n \times n$ matrix that results from taking the second derivative must be positive definite. If $\frac{d^2}{d\vec{x}^2}J(\vec{x}^*) = 0$, then we require the third derivative to be zero and look for the fourth derivative to be positive definite.

To find a stationary point, then, one may solve for $\vec{x}^*$ using the first order condition. The first order condition is satisfied by taking the differential of the function and setting it equal to zero.

$$dJ = \frac{\partial}{\partial x_1}J\,dx_1 + \frac{\partial}{\partial x_2}J\,dx_2 + \ldots + \frac{\partial}{\partial x_n}J\,dx_n = 0 \tag{2.5}$$

The stationary point occurs when the differential, dJ, equals zero, although the differentials of each argument (dx_i) are not required to be zero (they are allowed to vary in the neighborhood of the stationary point). Therefore, we set equal to zero the partial derivative of J with respect to each element of $\vec{x}$, and this makes a set of n equations to solve for the n unknowns of $\vec{x}^*$.

The process is a little more complicated, however, if constraints are added to the problem. For example, what if it is required that $x_1 + x_2 = 5$ or $x_3 \geq 0$? If $f(\vec{x})$ is the constraint function, then let $f(\vec{x}) = x_1 + x_2 - 5$ or $f(\vec{x}) = x_3$. Now, we are solving for the minimum of $J(\vec{x})$ such that, in general, either $\vec{f}(\vec{x}) = 0$ (first case) or $\vec{f}(\vec{x}) \geq 0$ (second case), where $\vec{f}$ is of dimension $m < n$. If the problem has equality constraints, it may be easiest to solve the problem using elimination (using the equality constraints to reduce the problem to $n - m$ equations and unknowns). However, a more general solution method is the Method of Lagrange. This is the method that will be applied with the optimal control problem, so it is applied here as well. If a condition actively constrains the cost function, we would like to find the point where the gradient of the cost function is parallel to the gradient of the boundary of each of the constraint functions (see Figure 2-1). That is,

$$\frac{d}{d\vec{x}}J = -\vec{\lambda}^T\frac{d}{d\vec{x}}\vec{f}. \tag{2.6}$$

Therefore, the Lagrangian, L, is defined.

$$L = J + \vec{\lambda}^T\vec{f} \tag{2.7}$$

The first order necessary conditions for a stationary point when equality constraints are present become

$$\frac{\partial}{\partial\vec{x}}L(\vec{x}^*,\vec{\lambda}^*) = 0 \tag{2.8}$$

$$\vec{f}(\vec{x}) = \frac{\partial}{\partial\vec{\lambda}}L^T = 0 \tag{2.9}$$

23
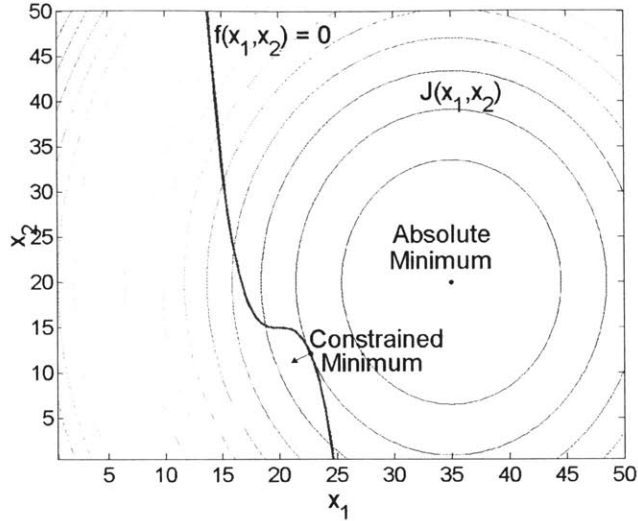
Figure 2-1: Constrained Minimization: The Method of Lagrange

which gives $n + m$ equations to solve for the n dimensions of the stationary point and for the m multipliers. For inequality constraints, the condition in Equation 2.9 is simply modified to say that for each constraint,

$$\lambda_i \frac{d}{d\lambda_i} L = 0 \tag{2.10}$$

so that when an inequality constraint is not active, then its corresponding multiplier equals żero. Notice that because f should equal zero, this augmentation to the minimizing function found in the Lagrangian will not affect the cost when the constraints are met.

The information presented in this section is generalized by the Kuhn-Tucker Condition:

Let $\vec{x}^*$ be a relative minimum for the problem

Minimize J subject to $\vec{f}_1(\vec{x}) = 0$ and $\vec{f}_2(\vec{x}) \geq 0$

where $\vec{f}_{1_x}$ and $\vec{f}_{2_x}$ are linearly independent. Then there are Lagrange multipliers $\vec{\lambda}_1$ and $\vec{\lambda}_2$, with $\vec{\lambda}_2 \leq 0$ such that

$$J_x(\vec{x}^*) + \vec{\lambda}_1^T \vec{f}_{1_x}(\vec{x}^*) + \vec{\lambda}_2^T \vec{f}_{2_x}(\vec{x}^*) = 0 \tag{2.11}$$

$$\vec{\lambda}_2^T \vec{f}_2(\vec{x}^*) = 0 [23]. \tag{2.12}$$

24

2.2 Functional Minimization

A step closer to the optimal control problem is functional minimization, where the *function*, $J(\vec{x})$, is now replaced by the *functional*, $J[\vec{x}(t)]$, and the argument is itself a function of another variable. The basic concepts applied to function minimization still apply here, but now the calculus of variations is applied. In Section 2.1, the stationary point was found by taking the *differential* of the cost function and setting it equal to zero (Equation 2.5). Now, we take the *variation* of the functional and set that equal to zero. Kirk [22, pp. 114–117] defines the variation in terms of the increment:

The increment of the functional J, ΔJ, can be defined as

$$\Delta J(x, \delta x) \equiv J(x + \delta x) - J(x). \tag{2.13}$$

It can also be written as

$$\Delta J(x, \delta x) = \delta J(x, \delta x) + g(x, \delta x)|\delta x|, \tag{2.14}$$

where δJ is linear in δx. If

$$\lim_{|\delta x| \to 0} \{g(x, \delta x)\} = 0, \tag{2.15}$$

then J is said to be *differentiable* on x and δJ is the *variation* of J evaluated for the function x.

The variation, δJ, is a linear approximation to the difference in J caused by changing x. The variation of x (δx) changes x in such a way that $x + \delta x$ is an admissible function (x and $x + \delta x$ must be in the domain, Ω).

As an example, consider the cost functional, $J = \int_{t_0}^{t_f} g[\vec{x}(t), t]dt$. The variation of J with respect to $\vec{x}$, t_0, and t_f is

$$\delta J = \int_{t_0}^{t_f} g_x[\vec{x}(t), t]\delta\vec{x} \, dt + g[\vec{x}(t_f), t_f]\delta t_f - g[\vec{x}(t_0), t_0]\delta t_0 \tag{2.16}$$

where the subscript, x, on the integrand function denotes the partial derivative with respect to $\vec{x}$.

With function minimization, we set the differential equal to zero. Now, To find an extremal $\vec{x}^*(t)$, each term in the variation must equal zero. As $\delta\vec{x}$ cannot be zero over the entire interval (t_0, t_f), then the rest of the integrand must equal zero. The statement,

$$g_x[\vec{x}(t), t] = 0, \tag{2.17}$$

is a differential equation which establishes the form of the solution for $\vec{x}(t)$. This will be called the Euler Equation.

Anything outside of the integral creates boundary conditions to solve for constants of integration, unknown variables (such as t_0 and t_f in the above case), or multipliers (should constraints be imposed).

Now, we look at a general functional minimization problem, including constraints. The cost functional can be put in the Bolza form, noting that the integrand function now also includes $\dot{\vec{x}}$ as an argument:

$$J = h[\vec{x}(t_f), t_f] + \int_{t_0}^{t_f} g[\vec{x}(t), \dot{\vec{x}}(t), t]dt \qquad (2.18)$$

with terminal and point constraints,

$$\begin{aligned} \vec{m}[\vec{x}(t_f), t_f] &= 0 \\ \vec{f}[\vec{x}(t), \dot{\vec{x}}(t), t] &= 0, \end{aligned} \qquad (2.19)$$

respectively. Let us assume, as is commonly the case, that the initial conditions and the initial time are fixed. Even though the final time and states are not fixed, there are some conditions, $\vec{m}$, that we wish to impose at the final time. As well, there are some restrictions, $\vec{f}$, that apply over the entire time interval. Using the same Lagrange method described in function minimization, these constraints are included in an augmented cost functional with appropriate Lagrange multipliers, $\vec{\nu}$ and $\vec{\lambda}$. Therefore, h is replaced with the augmented terminal cost,

$$l[\vec{x}(t_f), \vec{\nu}, t_f] = h[\vec{x}(t_f), t_f] + \vec{\nu}^T \vec{m}[\vec{x}(t_f), t_f], \qquad (2.20)$$

and the integral cost is replaced by

$$g_a[\vec{x}(t), \dot{\vec{x}}(t), \vec{\lambda}, t] = g[\vec{x}(t), \dot{\vec{x}}(t), t] + \vec{\lambda}^T \vec{f}[\vec{x}(t), \dot{\vec{x}}(t), t]. \qquad (2.21)$$

The new, augmented cost functional is

$$J_a = l[\vec{x}(t_f), \vec{\nu}, t_f] + \int_{t_0}^{t_f} g_a[\vec{x}(t), \dot{\vec{x}}(t), \vec{\lambda}, t]dt. \qquad (2.22)$$

The extremal solution is found by taking the variation, δJ_a, and setting each term equal to zero. This results in the following set of necessary conditions, which must be applied in addition to the constraints in Equations 2.19.

$$\begin{aligned} \frac{\partial}{\partial \vec{x}} g_a - \frac{d}{dt}\left[\frac{\partial}{\partial \dot{\vec{x}}} g_a\right] &= 0 \text{ (Euler Equation)} \\ \frac{\partial}{\partial \vec{x}} l(t_f) + \frac{\partial}{\partial \dot{\vec{x}}} g_a(t_f) &= 0 \qquad (2.23) \\ \frac{\partial}{\partial t} l(t_f) + g_a(t_f) - \frac{\partial}{\partial \dot{\vec{x}}} g_a(t_f)\dot{\vec{x}}(t_f) &= 0 \text{ (Transversality Condition)} \end{aligned}$$

26

Notice how the Euler Equation reduces to Equation 2.17 if $\dot{\vec{x}}$ is not an argument of g_a. Solving the differential Euler Equation determines the form of the solution, and then the other boundary conditions solve for any constants left in the solution, the final time, and the Lagrange multipliers $\vec{\nu}$ and $\vec{\lambda}$.

2.3 The Optimal Control Problem

The form of the optimal control problem does not differ too much from the functional minimization problem. The major difference involves introducing the variable, $\vec{u}$, to represent the controls. Optimal control theory is used to solve the following problem:

> Find an admissible control $\vec{u}^*(t)$ which causes the system $\dot{\vec{x}}(t) = \vec{f}[\vec{x}(t), \vec{u}(t), t]$ with initial conditions $\vec{x}(t_0) = \vec{x}_0$ to follow an admissible trajectory $\vec{x}^*(t)$ that minimizes the performance measure $J[\vec{x}(t), \vec{u}(t), t] = h[\vec{x}(t_f), t_f] + \int_{t_0}^{t_f} g[\vec{x}(t), \vec{u}(t), t]dt$ subject to the terminal constraints $\vec{m}[\vec{x}(t_f), t_f] = 0$.

Notice that the problem is to find an *admissible* control and trajectory, implying that there may be bounds on the states and controls that complicate the problem. Assume at first, however, that there are no constraints on the controls, and that the only condition on the trajectory is that it follows the system dynamics expressed in $\dot{\vec{x}} = \vec{f}$. Section 2.3.1 will explore the realistic case where control limitations exist.

For now, though, we use the same procedure to solve the problem: apply the Lagrange Method to the cost functional to include the system dynamics and constraints, and determine necessary first order conditions by taking the variation of the cost functional and forcing it to zero. To simplify the process, the Hamiltonian is defined:

$$\mathcal{H}(\vec{x}(t), \vec{u}(t), \vec{\lambda}(t), t) = g[\vec{x}(t), \vec{u}(t), t] + \vec{\lambda}(t)^T \vec{f}[\vec{x}(t), \vec{u}(t), t] \qquad (2.24)$$

where the Lagrange multipliers, $\vec{\lambda}$, are also known as the costates. Define the augmented terminal cost:

$$l[\vec{x}(t_f), \vec{\nu}, t_f] = h[\vec{x}(t_f), t_f] + \vec{\nu}(t_f)^T \vec{m}[\vec{x}(t_f), t_f]. \qquad (2.25)$$

The first order necessary conditions for optimality at an interior point can be expressed as:

$$\begin{aligned} \dot{\vec{x}}^*(t) &= \mathcal{H}_\lambda^T = \vec{f}[\vec{x}(t), \vec{u}(t), t] \\ \dot{\vec{\lambda}}^*(t) &= -\mathcal{H}_x^T \\ 0 &= \mathcal{H}_u. \end{aligned} \qquad (2.26)$$

The terminal constraints are:

$$\begin{aligned} \vec{x}(t_0) &= \vec{x}_o \\ \vec{m}[\vec{x}(t_f), t_f] &= 0. \end{aligned} \qquad (2.27)$$

The transversality conditions come from two terms in the variation of the cost function.

$$\left[\frac{\partial}{\partial \vec{x}} l(t_f) - \vec{\lambda}^{*T}(t_f)\right] \delta \vec{x}_f + \left[\mathcal{H}(t_f) + \frac{\partial}{\partial t} l(t_f)\right] \delta t_f = 0 \qquad (2.28)$$

Most generally, $\delta \vec{x}_f$ and δt_f are not zero (because neither of these has been fixed). To make the transversality equation true, then,

$$\vec{\lambda}^*(t_f) = \left[\frac{\partial}{\partial \vec{x}} l(t_f)\right]^T \qquad (2.29)$$

$$\mathcal{H}(t_f) = -\frac{\partial}{\partial t} l(t_f). \qquad (2.30)$$

These conditions may simplify based on the nature of the constraints. For example, if the final states are fixed, then $\delta \vec{x}_f = 0$ and Equation 2.29 is unnecessary. If the terminal states lie on a manifold identified by $\vec{\theta}(t)$, then

$$\vec{x}(t_f) = \vec{\theta}(t_f)$$

$$\delta \vec{x}_f = \left[\frac{d}{dt}\vec{\theta}(t_f)\right] \delta t_f$$

and Equation 2.28 reduces to

$$\left[\frac{\partial}{\partial \vec{x}} l(t_f) - \vec{\lambda}^{*T}(t_f)\right] \frac{d}{dt}\vec{\theta}(t_f) + \mathcal{H}(t_f) + \frac{\partial}{\partial t} l(t_f) = 0. \qquad (2.31)$$

2.3.1 Pontryagin's Minimum Principle

In the above setup for the optimal control problem, constraints are placed on the states through the dynamics equations and through the terminal constraints, without any constraints on the controls. In most real problems, however, we must constrain the controls in some way. Unfortunately, one cannot calculate the optimal control with free controls, and then impose the constraints after the problem is solved.

Pontryagin's Minimum Principle simply states that the optimal control, $\vec{u}^*(t)$, can be found as

$$\vec{u}^*(t) = \arg \left[\min_{\vec{u}(t) \in \mathcal{U}(t)} [\mathcal{H}(\vec{x}^*, \vec{u}, \vec{\lambda}^*, t)]\right] \qquad (2.32)$$

which is equivalent to

$$\mathcal{H}(\vec{x}^*, \vec{u}^*, \vec{\lambda}^*, t) \leq \mathcal{H}(\vec{x}^*, \vec{u}, \vec{\lambda}^*, t) \qquad (2.33)$$

where $\mathcal{U}(t)$ denotes the domain of the control variables.

As a simple example to illustrate this principle, consider the following optimal control problem:

Drive the position and velocity of a double integrator system with bounded

control from some initial condition to the origin in minimum time.

In equation form, the problem becomes,

$$\begin{aligned}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= u \\
x(t_0) &= x_0 \\
x(t_f) &= 0
\end{aligned}$$

(2.34)

$$J = \int_{t_0}^{t_f} 1 \, dt$$

(2.35)

subject to

$$u_{min} \le u \le u_{max}$$

(2.36)

where $u_{min} = -u_{max}$. The Hamiltonian for this problem becomes

$$\mathcal{H} = 1 + \lambda_1 x_2 + \lambda_2 u$$

(2.37)

where $\dot{\lambda}_1 = 0$ and $\dot{\lambda}_2 = -\lambda_1$. Therefore, λ_2 will be a linear function with constants for slope and intercept to be determined. Applying Pontryagin, the Hamiltonian will be minimized if u takes its maximum value when λ_2 is negative, and u takes its minimum value when λ_2 is positive. That is,

$$u = \begin{cases} u_{max}, & \lambda_2 < 0 \\ u_{min}, & \lambda_2 > 0 \end{cases}$$

(2.38)

Because λ_2 is linear in time, then over the interval, u will remain u_{max}, remain u_{min}, or switch once from one to the other at some point in the interval. This creates what is known as a "bang-bang" solution, characteristic of minimum time problems.

Another classic problem is the minimum fuel problem. If this is illustrated with the same system as in Equation 2.34, only the following changes must be made:

$$J = \int_{t_0}^{t_f} |u| \, dt$$

(2.39)

$$\mathcal{H} = |u| + \lambda_1 x_2 + \lambda_2 u$$

(2.40)

$$u = \begin{cases} u_{max}, & \lambda_2 < -1 \\ u_{min}, & \lambda_2 > 1 \\ 0, & -1 < \lambda_2 < 1 \end{cases}$$

(2.41)

The result now is a "bang-off-bang" solution. Therefore, to minimize the amount of fuel, it is optimal to use control at its extreme for a short time interval, then coast for another interval, then use control in the other extreme for another interval as short as possible.

29

Sometimes it is the case that Pontryagin's Minimum Principle stated in Equation 2.32 fails to directly define a unique optimal control, $\vec{u}^*$, over a nonzero interval of time. Consider a well defined control set, and suppose the optimal control is on the interior of the domain ($\vec{u}^*$ is not on the bounds of $\mathcal{U}$). Let us define a singular arc:

> A singular arc exists in an extremal solution to an optimal control problem when
>
> $$\det\left[\mathcal{H}_{uu}\right] = 0 \qquad (2.42)$$
>
> for some nonzero time interval.

A control function is partially singular if one or more, but not all, of its elements are singular over a nonzero arc. If the control elements are independent, then u_i is a singular element over an interval when

$$\mathcal{H}_{u_i u_i} = 0 \qquad (2.43)$$

on that interval. Singular problems can occur in many applications, including orbital transfer. Bell and Jacobson [3] outline methods for handling this sort of situation.

The basics of optimal control theory were presented in this chapter by first developing the concepts of function and functional minimization. This established the foundation for the optimal control problem, which was introduced next. The necessary conditions for optimality were also summarized. For problems with constrained control, the principle of Pontryagin was applied to two examples. These classic examples will help to provide insight into the optimal control setting of the orbital transfer problem developed in subseqent chapters. The direct method used for solving optimal control problems in this thesis, however, does not apply the theory presented in this chapter. Nevertheless, in some cases it would be possible to use the costates to construct the Hamiltonian presented above, thus being able to verify the necessary conditions of optimality listed in Equation 2.26. The Hamiltonian could also be used to test for singular arcs.

Chapter 3

Numerical Methods for Solving Optimal Control Problems

In the previous chapter, the basic theory of optimal control was presented. Using the calculus of variations, we found the first order necessary conditions for extremal solutions. Collectively, these conditions established the form of the solution, $\vec{x}(t)$, and solved for the constants of integration and other parameters. Unfortunately, the process of finding an analytical solution becomes impossible for everything but the simplest of problems. Consequently, numerical techniques must be used to obtain the solution to most optimal control problems.

For this thesis, the Legendre Pseudospectral Method is used to solve orbital transfer problems. This chapter is devoted to understanding where this method fits into the realm of numerical methods that could be used. To do this, we first distinguish between indirect and direct methods at a high level. Since the Legendre Pseudospectral Method is a direct method, the theory of direct methods will be the focus of the remainder of the chapter. Before addressing direct optimization in detail, it is important to understand some of the basic numerical techniques used in direct methods, such as interpolation and quadrature. Then, collocation, spectral, and pseudospectral methods are described.

Indirect Methods

The indirect methods of solving optimal control problems are those that use the necessary conditions established in the last chapter. Therefore, indirect methods require one to introduce Lagrange multipliers, construct the Hamiltonian, and apply Pontryagin's maximum/minimum principle [46, p. 358]. Various gradient and shooting methods fall into the category of indirect methods.

Generally, indirect methods can be more accurate than the direct methods that will be discussed below, but there are several disadvantages that must be considered. First, the very nature of indirect methods implies that a background in optimal control theory is necessary to apply these methods effectively. One must know how

to construct the Hamiltonian, $\mathcal{H}$, and then be able to compute the partials, $\mathcal{H}_x$ and $\mathcal{H}_u$. In some applications this may be difficult. As well, indirect methods require the user to have substantial intuition regarding the solution form. Guesses must be made *a priori* for control switching points and sequences of constrained and unconstrained arcs, and this may be difficult to do. Most importantly, indirect methods require guesses for the multipliers, $\vec{\lambda}$ and $\vec{\nu}$, which are non-physical parameters, the values for which one can have little to no intuition [4, p. 74]. Additionally, the convergence of these techniques requires the user to supply good guesses for $\vec{\lambda}$ and $\vec{\nu}$. This is a severe limitation of indirect methods.

Direct Methods

With direct methods, the optimal control problem is transformed from a functional minimization problem to a function minimization problem by discretizing the states and controls. Through direct transcription, the optimal control problem becomes a nonlinear programming problem. The Legendre Pseudospectral Method is a direct method.

One of the primary advantages of using direct methods is that a deep understanding of optimal control theory or of the nature of the problem is not necessarily required to solve the problem. Switching structures and adjoint variables are not as much of a concern. However, the approximation to the solution is generally less accurate than what comes from an indirect method (provided that the indirect method can obtain a solution). Direct methods are successful in solving for local minima, but it is not guaranteed that the local minimum that they find will be the global minimum or that a solution will satisfy the optimal control necessary conditions (Pontryagin's principle). But even if a direct method converges to the global solution, there are limitations in the discretization of the states and controls. The accuracy of a discrete solution does not necessarily improve by simply increasing the dimension of the discretization [46, p. 360].

The objective of the remainder of this chapter is to develop an understanding of some progressively more complicated direct methods, starting with basic collocation and moving through to pseudospectral methods. Necessary to this discussion is an understanding of some numerical techniques which include interpolation, orthogonality, and quadrature. Consequently, this material is presented first, and it is followed by description of the various direct methods.

3.1 Numerical Techniques

DIDO employs several numerical techniques to solve optimal control problems. This section is devoted to summarizing the theory behind those techniques. First, the Lagrange form of the interpolating polynomial is covered. This is followed by a discussion on orthogonal functions to include one of the classical orthogonal polynomials.

This leads well into a section on Gaussian quadrature, a highly accurate method of integral approximation.

3.1.1 Lagrange Interpolation

Approximating a polynomial to a given set of data is necessary when information beyond the data set is required. This may be the case when one is interested in the value between two points in a set. A polynomial approximation may also be used to replace an intractable function with a tractable one [18, p. 91]. A polynomial approximation may be found with data from a single point, as in a Taylor series, where one would look at a given function and its derivatives at one evaluation point to approximate values in the neighborhood of that point. Alternatively, an interpolating polynomial takes values of a function, $f(x)$, at several points and fits a polynomial to it. This polynomial, $P(x)$, will have the property that it is exact at the N points that were chosen for the interpolation:

$$P(x_j) = f(x_j) \text{ for } j = 1, 2, \ldots, N. \tag{3.1}$$

The interpolating polynomial, then, will be of degree $N - 1$, and it is unique. From this, it is clear that the interpolating polynomial will be exact when approximating a polynomial of any degree less than or equal to $N - 1$. This is supported by a corollary to the Fundamental Theorem of Algebra:

Let P and Q be polynomials of degree at most N. If $x_1, x_2, \ldots, x_k$ with $k > N$, are distinct points such that $P(x_i) = Q(x_i)$ for $i = 1, 2, \ldots, k$, then $P(x) = Q(x)$ for all values of x [6, p. 61]

The polynomial can take on several forms. One is based on a sum of polynomials, each of which corresponds to one of the data points, where the polynomial equals 1 at that point and equals 0 at any of the other points. This is the Lagrange form of the interpolating polynomial, which is written as:

$$P(x) = \sum_{j=1}^{N} f(x_k)\phi_j(x) \tag{3.2}$$

where the Lagrange polynomial is

$$\phi_j(x) = \prod_{\substack{i=1 \\ i \neq j}}^{N} \frac{(x - x_i)}{(x_j - x_i)} \quad \text{for} \quad j = 1, 2, \ldots, N. \tag{3.3}$$

In Equation 3.3, the subscript on the Lagrange polynomial "j" indicates that it is the polynomial corresponding to the jth data point. Each Lagrange polynomial will be of degree no greater than $N - 1$. As a simple example, the interpolating polynomial

for a set of two points $\{(x_1, y_1), (x_2, y_2)\}$ is the line that connects those two points, and in the Lagrange form would be expressed as

$$P(x) = \frac{(x - x_2)}{(x_1 - x_2)}y_1 + \frac{(x - x_1)}{(x_2 - x_1)}y_2. \tag{3.4}$$

The error for the interpolating polynomial is bounded, and this fact can be used to determine the number of points necessary to find a polynomial that interpolates to a given level of accuracy [6, 18].

3.1.2 Orthogonal Functions

Two functions are orthogonal if their inner product is zero. Define a weight function, $w(x)$, as integrable on $[a, b]$, greater than or equal to zero on (a, b), but not identically zero on any subinterval of (a, b). The inner product of two functions with respect to that weight function, then, is

$$(f, g) = \int_a^b w(x)f(x)g(x)dx \tag{3.5}$$

and this equals zero if f and g are orthogonal. Therefore, from [6, p. 156], one can define an orthogonal set of functions $\{p_0, p_1, \ldots, p_N\}$ on an interval $[a, b]$ with respect to the weight function $w(x)$ if

$$\int_a^b w(x)p_i(x)p_j(x)dx = \left\{ \begin{array}{ll} 0, & i \neq j \\ \alpha_i > 0, & i = j. \end{array} \right. \tag{3.6}$$

If those functions are polynomials, where $\deg(p_i) = i$, then a linear combination of the elements of the orthogonal set can be used to construct any polynomial with degree less than or equal to $N - 1$. Gram and Schmidt outline a process by which to construct an orthogonal set of polynomials on $[a, b]$ with respect to $w(x)$, and that is summarized below [6, p. 157].

$$\begin{array}{rcl} p_0(x) & = & 1 \\ p_1(x) & = & x - B_1 \\ p_k(x) & = & (x - B_k)p_{k-1}(x) - C_k p_{k-2}(x), \quad k \geq 2 \end{array} \tag{3.7}$$

where

$$B_1 = \frac{\int_a^b xw(x)\left[p_0(x)\right]^2 dx}{\int_a^b w(x)\left[p_0(x)\right]^2 dx}$$

$$B_k = \frac{\int_a^b xw(x)\left[p_{k-1}(x)\right]^2 dx}{\int_a^b w(x)\left[p_{k-1}(x)\right]^2 dx}$$

$$C_k = \frac{\int_a^b xw(x)p_{k-1}(x)p_{k-2}(x)dx}{\int_a^b w(x)\left[p_{k-2}(x)\right]^2 dx}.$$

This process can be modified to meet other requirements. For example, orthonormalization forces $(p_k, p_k) = 1$, or in general,

$$(p_i, p_j) = \delta_{ij} \tag{3.8}$$

where δ_{ij} is the Kronecker delta function. It is also common to require the coefficient in $p_i(x)$ for x^i to be greater than zero.

Legendre Polynomials

For the special case that the interval $[a, b] \equiv [-1, 1]$, and the weight function $w(x) \equiv 1$, the orthogonal set of polynomials is the Legendre Polynomials, so named as they are the solution to the Legendre differential equation. Using the process described above to construct these polynomials, the first four are listed below in their most convenient form (where the first coefficient of each polynomial is 1).

$$
\begin{aligned}
P_0(x) &= 1 \\
P_1(x) &= x \\
P_2(x) &= x^2 - \frac{1}{3} \\
P_3(x) &= x^3 - \frac{3}{5}x
\end{aligned}
\tag{3.9}
$$

A closed form of this sequence also exists, although the polynomials that are produced by the closed form are scaled differently. The Rodrigues form [26, p. 44] is used later in this section:

$$
\begin{aligned}
\tilde{P}_0(x) &= 1 \\
\tilde{P}_k(x) &= \frac{(-1)^k}{2^k k!} \frac{d^k}{dx^k} \left[\left(1 - x^2\right)^k\right] \quad k \geq 1.
\end{aligned}
\tag{3.10}
$$

In this form, $\tilde{P}_k(1) = 1$ and $(\tilde{P}_k, \tilde{P}_k) = \frac{2}{2k+1}$.

3.1.3 Gaussian Quadrature

Gaussian quadrature is an extremely accurate numerical method for integral approximation. The concept is to approximate

$$\int_a^b w(x)f(x)dx \approx \sum_{i=1}^N w_i f(x_i) \tag{3.11}$$

by choosing the nodes $\{x_1, x_2, \ldots, x_N\}$ and the weights $\{w_1, w_2, \ldots, w_N\}$ in some optimal fashion: to minimize the error (or eliminate it entirely) for polynomials f of the highest possible degree. Because there are $2N$ free parameters (the nodes and the weights), this integral approximation should be able to achieve a $2N - 1$ degree of precision.

Gauss-Legendre Integration

It can be shown that of the set of orthogonal polynomials $\{p_0, p_1, \ldots, p_N\}$, each p_k has exactly k distinct roots in the interval (a, b) [6, p. 212]. If these roots are chosen as the nodes for an integral approximation, then the $2N - 1$ degree of precision is achieved. When the integration interval is $[-1, 1]$ and $w(x) = 1$, the Legendre polynomials form the optimal polynomial set. Thus, we approximate

$$\int_{-1}^1 f(x)dx \approx \sum_{i=1}^N w_i f(x_i) \tag{3.12}$$

where the nodes are the N roots of $P_N(x)$ and the weights are calculated by

$$
\begin{aligned}
w_k &= \int_{-1}^1 \prod_{\substack{i=1 \\ i \neq k}}^N \frac{(x - x_i)}{(x_k - x_i)} dx \\
&= \frac{-2}{(N+1)\tilde{P}_N'(x_k)\tilde{P}_{N+1}(x_k)} \quad k = 1, 2, \ldots, N.
\end{aligned} \tag{3.13}
$$

The second equation uses the closed form scaling of the Legendre polynomials from Equation 3.10 [1, p. 237]. Generally, however, the nodes and weights for Gauss-Legendre integration are found in tables.

When it is necessary to integrate over an interval other than $[-1, 1]$, the following linear shift can be used:

$$\int_a^b f(t)dt = \left(\frac{b-a}{2}\right) \int_{-1}^1 f\left(\frac{a+b+x(b-a)}{2}\right) dx. \tag{3.14}$$

36

Gauss-Lobatto Integration

Lobatto integration is very similar to Legendre integration. This time, however, the nodes will include the endpoints a and b. To ensure that there are still N points, the remaining nodes must be the roots of the derivative of the $(N - 1)$ Legendre polynomial. Gauss-Lobatto integration, then, is as follows [10, p. 104]:

$$\int_{-1}^{1} f(x)dx \approx \frac{2}{N(N - 1)}\left[f(1) + f(-1)\right] + \sum_{i=2}^{N-1} w_i f(x_i) \tag{3.15}$$

where

$$w_k = \frac{2}{N(N - 1)\left[P_{N-1}(x_k)\right]^2} \text{ for } x_k \neq \pm 1. \tag{3.16}$$

3.2 Direct Optimization Methods

To get into some of details of direct methods, let us first put forth several definitions.

> 1. **Collocation Methods** enforce conditions (generally that residuals be zero) at as many (if not all) chosen spatial points. 2. **Spectral Methods** use high-degree, non-zero polynomials as global basis functions over the entire computational interval. 3. **Pseudospectral Methods** use these global basis functions and enforce conditions at chosen points. The points or nodes are obtained from a Gauss quadrature formula.

The following discussion should clear up these definitions while providing some description as to how these methods can be implemented.

3.2.1 Collocation Methods

Recall the form of an optimal control problem:

> Find an admissible control $\vec{u}^*(t)$ which causes the system $\dot{\vec{x}}(t) = \vec{f}[\vec{x}(t), \vec{u}(t), t]$ with initial conditions $\vec{x}(t_0) = \vec{x}_0$ to follow an admissible trajectory $\vec{x}^*(t)$ that minimizes the performance measure $J[\vec{x}(t), \vec{u}(t), t] = h[\vec{x}(t_f), t_f] + \int_{t_0}^{t_f} g[\vec{x}(t), \vec{u}(t), t]dt$ subject to the terminal constraints $\vec{m}[\vec{x}(t_f), t_f] = 0$.

With any direct method, the states and controls are discretized in some way:

$$\vec{x}(t) \rightarrow \mathbf{X} \tag{3.17}$$

$$\vec{u}(t) \rightarrow \mathbf{U} \tag{3.18}$$

Let the states and controls be discretized into N evaluation points, or nodes. If there are n states and m controls, then $\mathbf{X}$ is $N \times n$ and $\mathbf{U}$ is $N \times m$. For the remainder

of the chapter, let the discussion be simplified to one state at a time. A single state $x(t)$ becomes $\mathbf{x} = [x_1 \cdots x_N]^T$. The theory can be easily expanded to multi-state problems.

Once the problem has been discretized, any path constraints that exist are imposed at each of the nodes. A nonlinear program results:

Minimize $\Phi(Y)$, $Y = \{x_1, \ldots, x_N, u_1, \ldots, u_N, t_f\}$, $Y \in \Re^{2N+1}$, subject to $a(Y) = 0$, $b(Y) \leq 0$

where a and b are equality and inequality constraints, respectively [46, p. 360].

Collocation is the method by which conditions are enforced. Specifically, when solving an optimal control problem, one is looking for a trajectory that obeys the dynamics of the problem, $\dot{x}(t) = f[x(t), u(t), t]$. In the discretization, an approximation for the time derivative is required at selected points. Therefore, we want to use the discrete values of $x(t)$ to force the dynamics to be satisfied. Define the residual of the dynamics to be

$$r_i = f(x_i, u_i, t_i) - \dot{x}'_i(\mathbf{x}, \mathbf{f}) \tag{3.19}$$

at some evaluation point, i, where $\dot{x}'_i$ is an approximation of $\dot{x}(t_i)$ in terms of the values of $\mathbf{x}$ and $\mathbf{f}$ at that evaluation point and several neighboring points. Collocation, then, is the process of driving that residual, r_i, to zero by varying the $\mathbf{x}$ arguments of $\dot{x}'_i(\mathbf{x}, \mathbf{f})$ [19, p. 339]. This is illustrated below using three different collocation methods: Euler's method, the midpoint formula, and Simpson's Rule, each approximating $\dot{x}'$ with increasing accuracy. For all of these, assume that the discrete points are selected at equal intervals, Δt.

Euler's Method

Recall that a Taylor expansion can be expressed as

$$x(t_{i+1}) = x(t_i) + \Delta t \ \dot{x}(t_i) + \frac{\Delta t^2}{2} \ddot{x}(\theta_i) \tag{3.20}$$

where θ_i is some value between t_i and t_{i+1}. The simplest derivative approximation uses Euler's integration rule, which comes from this Taylor expansion, ignoring the final (error) term:

$$x_{i+1} = x_i + \Delta t \ \dot{x}'_i \tag{3.21}$$

where $x_i = x(t_i), x_{i+1} = x(t_{i+1})$, etc. So an approximation to $\dot{x}$ at time t_i would be

$$\dot{x}'_i = \frac{(x_{i+1} - x_i)}{\Delta t}. \tag{3.22}$$

Equivalently, this is simply the slope of a linear interpolation formula connecting the two points x_i and x_{i+1}. This slope can now be forced into the dynamics of the problem

38

at t_i. Therefore, the dynamics residual is

$$r_i = f_i - \dot{x}'_i = f(x_i, u_i, t_i) - \frac{(x_{i+1} - x_i)}{\Delta t}, \qquad (3.23)$$

and the object is to vary x_i and x_{i+1} to force r_i to zero. Euler's method is of first order, and therefore is accurate to the order of Δt (the error term is in Δt^2).

Midpoint Method

It can be shown that a higher degree of accuracy can be achieved if we simply perform our evaluations at the midpoint between two interpolation points [8, p. 280]. The midpoint formula is derived from a single point quadrature, for simplicity shown here on the interval (0,1):

$$\int_0^1 x(t)dt = w_1 x(t_1). \qquad (3.24)$$

To make this exact for $x(t) = 1$ and $x(t) = t$, we find that $w_1 = 1$ and $t_1 = \frac{1}{2}$. Thus,

$$\int_0^1 x(t)dt \approx x\left(\frac{1}{2}\right). \qquad (3.25)$$

The error term in this approximation is $\frac{\Delta t^3}{24}\ddot{x}(\theta)$, where θ is some value between 0 and 1, indicating a higher degree of accuracy than Euler [18, p. 120]. Taking the derivative of both sides of Equation 3.25,

$$x(1) - x(0) \approx \dot{x}\left(\frac{1}{2}\right). \qquad (3.26)$$

If the interval is generalized to (a,b),

$$x(b) - x(a) \approx (b-a)\dot{x}\left(\frac{b-a}{2}\right). \qquad (3.27)$$

Now in the context of discretization,

$$x_{i+1} = x_i + \Delta t\, \dot{x}'_{i+\frac{1}{2}}. \qquad (3.28)$$

Notice that this is the same formula as in Euler's method, except for where the derivative approximation is evaluated. Thus collocation using the midpoint is more accurate, and requires minimal additional computation than Euler. Let $c = i + \frac{1}{2}$ and $x_c = \frac{(x_i + x_{i+1})}{2}$, etc. Now the residual is calculated between the nodes and is defined as

$$r_c = f_c - \dot{x}'_c = f(x_c, u_c, t_c) - \frac{(x_{i+1} - x_i)}{\Delta t}. \qquad (3.29)$$

39

Simpson's Method

In a final step to improve the accuracy of collocation, Simpson's Rule for integration is applied.

$$\int_{t_i}^{t_{i+1}} x(t)dt = \frac{\Delta t}{6}[x(t_i) + 4x(t_c) + x(t_{i+1})] - \frac{\Delta t^5}{90}x^{(4)}(\theta_i) \qquad (3.30)$$

where θ_i is a value between t_i and t_{i+1} [18, p. 156]. The error term in this expression indicates that this is accurate to the fourth order of Δt. As before, we want to take the derivative, collecting the error in the approximation of the midpoint derivative:

$$x_{i+1} = x_i + \frac{\Delta t}{6}(\dot{x}_i + 4\dot{x}'_c + \dot{x}_{i+1}). \qquad (3.31)$$

Rearranging for $\dot{x}'_c$,

$$\dot{x}'_c = -\frac{3}{2\Delta t}(x_i - x_{i+1}) - \frac{1}{4}(\dot{x}_i + \dot{x}_{i+1}). \qquad (3.32)$$

Allowing $\dot{x}_i = f_i$ and $\dot{x}_{i+1} = f_{i+1}$, the residual equation at a segment midpoint becomes

$$r_c = f_c - \dot{x}'_c = f(x_c, u_c, t_c) + \frac{3}{2\Delta t}(x_i - x_{i+1}) + \frac{1}{4}(f_i + f_{i+1}) \qquad (3.33)$$

where, again, x_i and x_{i+1} are varied to make the residual zero.

Thus, three collocation methods have been described here: Euler, midpoint, and Simpson. They are different in the level to which they can approximate a derivative, but the principle for each is the same. The dynamics are rearranged to solve for a residual, which must be driven to zero by varying the values of several interpolating points.

3.2.2 Spectral Methods

In the collocation methods discussed above, two points are used to help approximate a derivative. With the Midpoint Method and with Simpson's Method, a derivative is evaluated at a point other than at the data points or interpolation points (the points where state values are identified). With Euler's method, the derivative is evaluated at a data point. Variations of the other two methods, for example by using the set $\{x_{i-1}, x_i, x_{i+1}\}$ instead of $\{x_i, x_c, x_{i+1}\}$, can also eliminate the introduction of evaluation points outside of the data set. When this is the case, it may be convenient to express the differentiation in matrix form. If $\mathbf{x}$ is the vector $[x_1 \cdots x_N]^T$, then

$$\dot{\mathbf{x}} \approx \mathbf{D}^N \mathbf{x} \qquad (3.34)$$

where $\mathbf{D}^N$ is the $N \times N$ differentiation matrix. We can develop the differentiation matrix for the methods described above. Remembering the approximation from Euler's method,

$$\dot{x}_i' = \frac{(x_{i+1} - x_i)}{\Delta t}, \tag{3.35}$$

and applying it over the entire grid, the matrix would look like:

$$\mathbf{D}^N = \frac{1}{\Delta t} \begin{bmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \\ 1 & & & & -1 \end{bmatrix}. \tag{3.36}$$

Again, recall that the grid is uniformly spaced. Notice also that this particular matrix is circulant—the diagonals wrap around. To simplify this development, assume that our function is periodic: $x_{N+1} = x_1$. If this were not the case, then we would not allow the diagonals to wrap around, and different approximations would be necessary at the boundaries.

Using the modified Midpoint Method, we see the standard second-order finite difference approximation. Now,

$$\dot{x}_i' = \frac{(x_{i+1} - x_{i-1})}{2\Delta t} \tag{3.37}$$

and applying it over the entire grid, the differentiation matrix is:

$$\mathbf{D}^N = \frac{1}{2\Delta t} \begin{bmatrix} 0 & 1 & & & -1 \\ -1 & 0 & 1 & & \\ & -1 & \ddots & \ddots & \\ & & \ddots & 0 & 1 \\ 1 & & & -1 & 0 \end{bmatrix} \tag{3.38}$$

By considering x_{i-1} as well, we have achieved an additional order of accuracy to our derivative approximation. An alternative derivation of this matrix exists by using a process of local interpolation and differentiation. The same equation for the second-order finite difference results by finding the second-order polynomial that interpolates x_{i-1}, x_i, and x_{i+1}, differentiating, and evaluating at t_i.

Using this method begs the question: Can one achieve even higher accuracy by differentiating a higher order interpolating polynomial? The answer is Yes, and it motivates the development of spectral methods, where the idea is taken to the limit. Given a set of discrete data, find the global interpolating polynomial, and evaluate the derivative of that interpolant at each point on the grid. This yields a dense

differentiation matrix, which can be used for collocation of the entire interval:

$$\mathbf{r} = \mathbf{f} - \dot{\mathbf{x}}' = \mathbf{f} - \mathbf{D}^N \mathbf{x} \to \mathbf{0}. \tag{3.39}$$

3.2.3 The Legendre Pseudospectral Method

Until now, all of the developments have used equally spaced data points for collocation and derivative approximations. However, this is not the optimal spacing of nodes to achieve the highest accuracy approximation. It is wise to apply the theory of Gaussian quadrature to the development of a differentiation matrix to realize intelligent node spacing. Recall that by placing the nodes at the roots of the Legendre polynomials, integration could be accomplished exactly for polynomials of degree $2N - 1$. This sets the stage for the derivation of the $\mathbf{D}^N$ matrix for the Legendre Pseudospectral Method, which will place nodes at the zeros of the derivative of the $N - 1$ degree Legendre polynomial, reserving $i = 1$ and $i = N$ for the endpoints (i.e. Gauss-Lobatto integration). What follows concerns $i = 2, \ldots, N - 1$.

To find the differentiation matrix, again we will differentiate the interpolating polynomial. Specifically, if $x^N(t)$ is the polynomial approximation of $x(t)$, then

$$x^N(t_i) = \sum_{j=1}^{N} x(t_j)\phi_j(t_i) \tag{3.40}$$

and

$$\dot{x}^N(t_i) = \sum_{j=1}^{N} x(t_j)\dot{\phi}_j(t_i) = \sum_{j=1}^{N} D_{ij}x_j \tag{3.41}$$

if

$$D_{ij} = \dot{\phi}_j(t_i). \tag{3.42}$$

To glean information based on the new node spacing, it is preferable to have the interpolating polynomial in terms of the Legendre polynomial, $P_{N-1}(t)$. Define an intermediate function, $z(t)$, as

$$z(t) = \prod_{i=1}^{N} t - t_i. \tag{3.43}$$

Its derivative, when evaluated at one of the roots, j, becomes

$$\dot{z}(t_j) = (t_j - t_1) \ldots (t_j - t_{j-1})(t_j - t_{j+1}) \ldots (t_j - t_N) = \prod_{\substack{i=1 \\ i \neq j}}^{N} t_j - t_i. \tag{3.44}$$

The Lagrange interpolating function can be expressed in terms of the intermediate

42

function.

$$\phi_j(t) = \prod_{\substack{i=1 \\ i \neq j}}^{N} \frac{(t - t_i)}{(t_j - t_i)} = \frac{z(t)}{(t - t_j)\dot{z}(t_j)} \tag{3.45}$$

Recall that the points $\{t_2, \ldots, t_{N-1}\}$ are chosen to be the roots of $P'_{N-1}(t)$. Also notice that $z(t)$ contains the same roots, with two additional roots at the endpoints (1 and −1).

$$z(t) = (t - t_1)P'_{N-1}(t)(t - t_N) = (t + 1)P'_{N-1}(t)(t - 1) = (t^2 - 1)P'_{N-1}(t) \tag{3.46}$$

Using the Legendre differential equation,

$$\frac{d}{dt}\left[(t^2 - 1)P'_{N-1}(t)\right] = N(N - 1)P_{N-1}(t), \tag{3.47}$$

it is clear that

$$\dot{z}(t) = N(N - 1)P_{N-1}(t). \tag{3.48}$$

Combining the results of Equations 3.46 and 3.48,

$$\phi_j(t) = \frac{(t^2 - 1)P'_{N-1}(t)}{(t - t_j)N(N - 1)P_{N-1}(t_j)}. \tag{3.49}$$

Differentiating and evaluating at node i yields

$$\dot{\phi}_j(t_i) = \frac{1}{N(N - 1)P_{N-1}(t_j)}\left[\frac{2t_i P'_{N-1}(t_i)}{t_i - t_j} + \frac{(t_i^2 - 1)P''_{N-1}(t_i)}{t_i - t_j} - \frac{(t_i^2 - 1)P'_{N-1}(t_i)}{(t_i - t_j)^2}\right]. \tag{3.50}$$

Looking again at Equation 3.47,

$$\begin{aligned} N(N - 1)P_{N-1}(t) &= \frac{d}{dt}\left[(t^2 - 1)P'_{N-1}(t)\right] \\ &= 2tP'_{N-1}(t) + (t^2 - 1)P''_{N-1}(t), \end{aligned} \tag{3.51}$$

and Equation 3.50 simplifies to

$$\dot{\phi}_j(t_i) = \frac{1}{N(N - 1)P_{N-1}(t_j)}\left[\frac{N(N - 1)P_{N-1}(t_i)}{t_i - t_j} - \frac{(t_i^2 - 1)P'_{N-1}(t_i)}{(t_i - t_j)^2}\right]. \tag{3.52}$$

Since the product, $(t_i^2 - 1)P'_{N-1}(t_i) = 0$, we find that

$$\begin{aligned} D_{ij} &= \dot{\phi}_j(t_i) \\ &= \frac{P_{N-1}(t_i)}{(t_i - t_j)P_{N-1}(t_j)}. \end{aligned} \tag{3.53}$$

Additional manipulation will yield the values of the differentiation matrix on the boundaries and when $i = j$. L'Hôpital's Rule is applied to Equation 3.52 to clear the $t_i - t_j$ terms out of the denominator. The resulting terms for the differentation matrix are:

$$D_{ij} = \begin{cases} -\frac{N(N-1)}{4}, & i = j = 1 \\ \frac{N(N-1)}{4}, & i = j = n \\ 0, & \text{otherwise.} \end{cases} \tag{3.54}$$

The elements D_{ij} collect to make the $N \times N$ differentation matrix, $\mathbf{D}^N$, used in the Legendre Pseudospectral Method.

This chapter has summarized the basic numerical tools that are necessary for understanding the theory applied in DIDO. The concepts of interpolation, orthogonality, and quadrature are combined with the technique of collocation to make a highly accurate direct transcription method called the Legendre Pseudospectral Method. Although it is certainly beneficial to understand the nature of the routine, DIDO implements this method without requiring the user to have this theoretical background. The user only needs to understand how to use DIDO, and this is described in the next chapter.

Chapter 4

General DIDO Problem Setup

In the last chapter, the Legendre Pseudospectral method was described in the context of other methods of optimization, both direct and indirect. Previously, it has been stated that this is the method used for the work in this thesis, as it is implemented in DIDO by Ross and Fahroo. DIDO can be used for any optimization problem, aerospace or otherwise. Therefore, this chapter is devoted to summarizing the basic essentials of DIDO as it would be applied to a general optimization problem. Additional information can be found in the DIDO User's Manual [36]. In the next chapter, specifics to the orbital transfer problem will be addressed.

4.1 Functions

A series of MATLAB routines are called by DIDO, and must be written (or modified) by the user appropriate to the given problem. DIDO calls functions to calculate the cost, represent the dynamics, establish the required events, and restrict the path. As well, a routine which we shall call "Main" glues everything together and calls the DIDO procedure.

4.1.1 Main Function

The main function in a DIDO setup is what is used to initiate the DIDO routine. The most important line is the call to the function dido with the appropriate inputs and outputs.

```
[cost, primal, dual] = dido(problem, knots, bounds, guess);
```

This is the complete interface to the DIDO optimization package. Prior to this line, the four structured inputs must be defined. The line serves as a good starting point for describing the essentials to using DIDO for any optimal control problem.

45

Node Locations
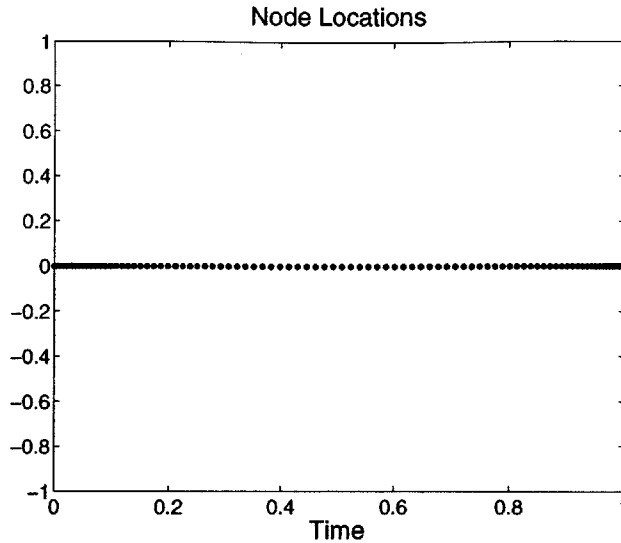
Figure 4-1: Standard Node Distribution for the Legendre Pseudospectral Method (100 nodes)

Defining the Problem

The structured input, `problem` contains four string variables called `cost`, `dynamics`, `events`, and `path`. Each string contains the user-defined name of one of the four functions that will be called by the DIDO procedure.

Knots and Nodes

An important feature of DIDO is the user's ability to describe the *knots* and *nodes* of a problem. A *node* is a point in time where states and controls are defined. These are the interpolating points. Clearly, more nodes equates to a smaller discretization of the time interval, increasing the accuracy of the solution. However, each node adds dimension to the differentiation matrix that must be evaluated to meet the constraints (50 nodes implies a 50×50 dense differentation matrix). Recall, though, that since DIDO uses the Legendre Pseudospectral technique, these nodes are not equally spaced. The node locations correspond to the roots of the derivative of the Legendre polynomials, consequently concentrating more nodes at the endpoints. Figure 4-1 gives an example of how 100 nodes would be spaced on an interval from 0 to 1.

Knots divide the problem in some way, specifying the endpoints of a set of nodes. The term *knot* is used to describe "the tying together" of two subintervals. The knot is the point at which the two subintervals are tied. Hence, at an interior knot the time at the last node of the previous interval is identical to the first node of the subsequent interval. Knots are located (at the least) at the terminal times, and the nodes fall

between them. These knots come in two different forms.

A *hard knot* allows for discontinuity in the states of the problem. Hard knots are used at the terminals, but they may also be placed at interior locations. For example, a hard knot would be useful in a rocket staging problem, where there is an instantaneous change in mass when a stage is dropped. An interior hard knot would require two nodes to be placed at the specified time, easily capturing discontinuities. Hard knots are also necessarily located at the terminal times (therefore, there is always a minimum of two hard knots—one at either endpoint).

A *soft knot* does not allow for state discontinuities (although control discontinuities are allowed). Soft knots can be used to divide the problem into multiple smaller time domains. For example, a 100-node problem can be changed into two connected 50-node problems, and this will be smaller in total computational space. A soft knot can also be placed at an interior time to concentrate nodes in a particular region. For example, if there is a high amount of control activity in a region where there are ordinarily few nodes, a soft knot placed in that region will change the distribution of nodes to highlight the activity in the controls.

Knots locations can either be fixed or allowed to float with some interval of time, allowing the optimizer to find the (numerically) optimal knot and node placement. User input with regard to knots and nodes is specified in the knots structure, which has five variables defining the numbers and locations of both knots and nodes for a problem. A user specifies the types of knots needed for the problem, a guess location for those knots, upper and lower bounds for knots to float, and the numbers of nodes that fall between the knots.

Bounds

Upper and lower bounds must also be set for key values, including the states, controls, path constraints, event constraints, and other parameters. For every optimization problem, limits must be set for the states and controls. Most of the time these limits will apply to both states and controls throughout the entire trajectory, but DIDO also allows for specific bounds to be applied at the endpoints as well. Path constraints are also bounded for more complicated requirements on the states and controls. For the boundary events defined by the user in the Events function, limits are defined here as well. If the events are set up as equality constraints, then the upper and lower bound for the event must be the same. A bound may also be set at $\pm\infty$.

Guess

The user must also supply a starting point for the optimization routine. The guess must contain all of the parameters for which DIDO must solve. The guess structure, then, must contain guess values for the states, controls, and other parameters over time. The guess is often the single most important driving factor in producing good results in DIDO, so a user should invest significant time to improve the quality of

47

the guess. In subsequent chapters containing results, details are discussed on what constitutes a good guess in the context of the problem solved.

Outputs

The DIDO procedure yields the outputs `cost` and `primal` with the optional output `dual`. The first output is simply the scalar evaluation of the cost function on the converged solution. The structure `primal` contains the (converged) trajectory states (and their time derivatives) and controls at the nodes, the node locations, and any other parameters. The output `dual` contains the values of the costates and the Hamiltonian at the nodes.

4.1.2 Cost Function

The user must create a function that evaluates the chosen performance index. This is a function that will be called by DIDO during the iteration process. The function uses the input structure `primal`, which contains all of the necessary information on the current iteration of the trajectory. The function must have two outputs, the endpoint cost and the integrand cost. The scalar value for the endpoint cost represents the portion of the cost functional found outside of the integral (which usually involves a cost evaluation at the terminal time). The integrand cost is a vector with length equal to the number of nodes, and it represents the values found within the integrand at each node. The DIDO routine uses this second output to evaluate the total cost.

4.1.3 Dynamics Function

The user-created Dynamics function accounts for the equations of motion describing the states of the problem. The same input structure `primal` is an input into the function, containing the states, their derivatives, and the controls. Recall that the state time derivatives are evaluated within the DIDO procedure using the Legendre pseudospectral differentation matrix. The output of this function must be an $n \times 1$ vector representing the residual difference between the left and right hand sides of the n state equations.

4.1.4 Events Function

The function containing the events takes in `primal` and outputs a vector of boundary conditions. This is a vector of user-specified length, whose boundaries are specified in the Main function (see Section 4.1.1). To illustrate the use of the Events function, consider a one-dimensional problem with two states: x and $\dot{x}$. If the optimization problem requires that the particle must terminate at the origin, (0,0)—zero displacement and zero velocity—then the output of the Events could simply be a vector of length 2, whose values are the states at the final time. By setting both the upper

and lower bounds of each to be zero in the Main function, DIDO will force the final states to be zero as it iterates new values of the states.

4.1.5 Path Function

The final function called by DIDO is the Path function, again bringing in `primal` as an input and outputting an $m \times N$ matrix, with a value for each of the m constraints imposed by the user at each of the N nodes. Path constraints may be appropriate for conditions that must be satisfied throughout the trajectory. For example, if there is a maximum limit for a vehicle's velocity, and the velocity is described by a vector of two or three states, then a path constraint would be appropriate. (If the velocity was described by a single state, then it could be handled in the state bounds without using the Path function.)

This function is set up in very much the same way as the Events function, however, it is handled somewhat differently in the DIDO procedure. While a feasible solution must meet both the event and path constraints, as DIDO iterates through different sets of states and controls, the event constraints may not always be satisfied. However, the path constraints will always be satisfied.

4.2 Scaling

A final issue of importance in setting up an optimal control problem in DIDO is scaling. It is not necessarily the best option to use standard units for scaling, such as meters, degrees, or seconds. It is difficult for the optimizer to find a minimum when one is working with variables whose ranges differ in orders of magnitude. As a general rule of thumb, all units should be scaled to units that put the variables in the same order of magnitude, most likely between 1 and 10 (zeroeth order). Therefore, instead of meters, degrees, and seconds, appropriate units for a heliocentric orbital transfer problem may be astronomical units, radians, and years.

49

[Except for this sentence, this page intentionally left blank.]

Chapter 5

Orbital Transfer Optimization

In the last chapter, it was demonstrated how one might set up DIDO to solve a general optimization problem. This chapter focuses attention on the orbital transfer optimization problem. The details of the problem are presented, including the dynamics, cost, and constraints as they are implemented in DIDO. Variations in setup used in this thesis are also introduced.

The dynamics of the problem, the performance index, and the constraints are parts of an optimization problem that exist regardless of the toolset used to solve it. It is easy to think that a user simply inputs the problem specifics into his toolset of choice, and that is the end of it. However, consideration must be given to the toolset when designing the problem. Sometimes, there are multiple ways of representing the same problem, and while problem setup alternatives may be equivalent on paper, they may not necessarily be the same when they are implemented. Some of the alternatives for the orbital transfer problem are presented here along with the lessons from experience regarding their advantages or disadvantages.

5.1 Problem Dynamics

Within the dynamical equations of a problem are the governing laws that describe the motion of a vehicle with or without control. Here, the states and controls, along with forces of nature, must be defined.

The dynamics of the orbital transfer problem are the same as any other orbital problem. Two-body mechanics must be applied, and depending on the nature of the problem, it may also be useful to include other perturbing forces, like J_2 or drag. All of the problems considered in this thesis are Earth-centered.

5.1.1 State Definitions

The states can be defined in a number of ways to represent an orbital problem. The major design choice to be made is what coordinate system to use to define either a

vehicle's position and velocity or its orbit and location within that orbit. Cartesian coordinates and modified equinoctial elements were tested in this work.

Cartesian Coordinates

A Cartesian coordinate system is probably the easiest form for defining states of an orbital problem. In the Geocentric-Equatorial Inertial Frame, the origin is at the center of the Earth, the x-axis points towards Aries (the vernal equinox direction), and the z-axis points towards the North Pole. The y-axis completes the right-hand rule and is in the equatorial plane with the x-axis. The dynamical equations come directly from the restricted two-body equation of motion derived from Newton's Law of Universal Gravitation:

$$\ddot{\vec{r}} + \frac{\mu}{r^3}\vec{r} = 0 \tag{5.1}$$

where $\vec{r}$ is the position vector of a vehicle from the origin (the center of the planet or the barycenter of the planetary system), and μ is the gravitational parameter. The gravitational parameter for Earth is

$$\mu_\oplus = 398600.5 \ km^3/s^2. \tag{5.2}$$

When controls or perturbations are included, they will be applied as additional accelerations, simply adding terms to Equation 5.1. Six states are necessary for sufficiently describing the problem: three states for position and three for velocity. Therefore, the six Cartesian equations of motion are as follows.

$$\begin{aligned} \dot{\vec{r}} &= \vec{v} \\ \dot{\vec{v}} &= -\frac{\mu}{r^3}\vec{r} + \vec{a} \end{aligned} \tag{5.3}$$

where $\vec{a} = a_x i_x + a_y i_y + a_z i_z$ is the additional acceleration imposed through either controls or perturbations.

Scaling is important when the problem is set in a system representing the position and velocity of a satellite. A satellite's position may vary in magnitude from around 6500 kilometers in low-Earth orbit to over 42,000 kilometers in a geosynchronous orbit, while it's velocity may be on the order of 1 to 10 kilometers per second. When the optimizer is looking for a trajectory that meets the dynamics in Equations 5.3, the residuals of the first three equations can be dramatically larger than those of the last three equations if the problem is scaled in kilometers and seconds. Therefore, it is logical to use the radius of the Earth and the Schuller period for scaling. Thus, the scaling is defined as follows:

$$\begin{aligned} \text{Distance Unit} &= R_\oplus = 6378.137 \ km \\ \text{Time Unit} &= 2\pi\sqrt{\frac{R_\oplus^3}{\mu}} = 5069.343 \ s \end{aligned} \tag{5.4}$$

52

The time unit is simply the period of a circular orbit whose semi-major axis is the radius of the Earth. Notice also, that in the scaled units, the gravitational constant reduces simply to $4\pi^2$.

Modified Equinoctial Elements

An element set has a significant advantage over a coordinate frame in which position and velocity are measured. Of the six elements that define a vehicle's movement, only one element varies without the presence of disturbing accelerations. Generally, the first five elements define the orbit (which is presumably constant), and the last element defines its position within the orbit. In the presence of perturbing accelerations, the first five elements still do not vary to the extreme that position and velocity states would; they are still considered "slow" variables.

Several different element sets exist that could be used to define the orbit and a vehicle's position in that orbit. The equinoctial elements were chosen over the classical orbital elements because of their ability to handle circular and equatorial orbits without special accomodations. The equinoctial elements were modified by using the semi-latus rectum instead of the semi-major axis to more gracefully support the transition from elliptical to parabolic orbits. The semi-latus rectum, usually defined as p will be symbolized as ϕ to reserve p for one of the other equinoctial elements.

$$
\begin{aligned}
\phi &= a(1 - e^2) \\
h &= e\sin(\Omega + \omega) \\
k &= e\cos(\Omega + \omega) \\
p &= \tan\left(\frac{i}{2}\right)\sin(\Omega) \\
q &= \tan\left(\frac{i}{2}\right)\cos(\Omega) \\
L &= \Omega + \omega + \nu
\end{aligned}
\tag{5.5}
$$

The equations of motion with this set states is summarized in Equations 5.6, where in the absence of control or perturbations, five of the six elements are conveniently

constants.

$$\dot{\phi} = \frac{2\phi}{w}\sqrt{\frac{\phi}{\mu}}a_\theta$$

$$\dot{h} = \sqrt{\frac{\phi}{\mu}}\left\{-a_r\cos L + [(w+1)\sin L + h]\frac{a_\theta}{w} + (q\sin L - p\cos L)\frac{ka_h}{w}\right\}$$

$$\dot{k} = \sqrt{\frac{\phi}{\mu}}\left\{a_r\sin L + [(w+1)\cos L + k]\frac{a_\theta}{w} - (q\sin L - p\cos L)\frac{ha_h}{w}\right\}$$

$$\dot{p} = \sqrt{\frac{\phi}{\mu}}\frac{s^2 a_h}{2w}\sin L \tag{5.6}$$

$$\dot{q} = \sqrt{\frac{\phi}{\mu}}\frac{s^2 a_h}{2w}\cos L$$

$$\dot{L} = \sqrt{\mu\phi}\left(\frac{w}{\phi}\right)^2 + \frac{1}{w}\sqrt{\frac{\phi}{\mu}}(q\sin L - p\cos L)a_h$$

where

$$w = 1 + k\cos L + h\sin L$$
$$s^2 = 1 + q^2 + p^2$$

and

$$\vec{a} = a_r i_r + a_\theta i_\theta + a_h i_h$$

is the disturbing acceleration vector in the polar frame. This includes both perturbations and controls [47, pp. 409–413].

The scaling used for distance and time in Cartesian coordinates is also appropriate here. The equinoctial elements h, k, p, and q do not require additional scaling. The true longitude, L, is well sized if the angle is measured in radians. Hence, only ϕ and t require scaling.

5.1.2 Perturbations

It is well known that a satellite in Earth orbit does not follow the equations of two-body motion because of various perturbations that disturb the two-body approximation. Disturbing accelerations are caused by third body forces (from the Sun or the Moon) or solar-radiation pressure, but chief among the list of perturbations are those caused by the non-spherical geometry of the central body and atmospheric drag at low altitudes. When solving an orbital optimization problem, it is important to be aware of these perturbations and prepared to account for them if necessary.

54

The J_2 Effect

The Earth is not a perfect sphere, so the point approximation of the central body in the two-body equations is not entirely accurate. Spherical harmonics are used to divide the Earth into sections separated by lines of latitude and longitude to account for these effects. J_2 is the perturbation that accounts most for the oblateness of the Earth—the fact that the Earth is fatter near the equator. While higher order harmonics can account for additional distortions, J_2 is certainly the most dominant of the shape perturbations, three orders of magnitude greater than the next largest perturbation coefficient (J_3). Therefore, only the effect due to the J_2 perturbation is considered here.

The implementation of that perturbation is fairly straightforward in either of the state forms discussed in Section 5.1.1, since in both sets of dynamical equations there was a place for additional acceleration terms. In the Cartesian coordinates, the disturbing acceleration in each direction is as follows:

$$
\begin{aligned}
a_{x(J_2)} &= -\frac{3J_2\mu R_\oplus^2 r_x}{2r^5}\left(1 - \frac{5r_z^2}{r^2}\right) \\
a_{y(J_2)} &= -\frac{3J_2\mu R_\oplus^2 r_y}{2r^5}\left(1 - \frac{5r_z^2}{r^2}\right) \\
a_{z(J_2)} &= -\frac{3J_2\mu R_\oplus^2 r_z}{2r^5}\left(3 - \frac{5r_z^2}{r^2}\right)
\end{aligned}
\tag{5.7}
$$

where $\vec{r} = [r_x\ r_y\ r_z]$ is the position vector of the satellite and

$$
J_2 = 1.0826269 \times 10^{-3}[45, p.528].
\tag{5.8}
$$

In polar coordinates, the same acceleration is

$$
\begin{aligned}
a_{r(J_2)} &= -\frac{3\mu J_2 R_\oplus^2}{2r^4}\left[1 - 12\frac{(q\sin L - p\cos L)^2}{(1+p^2+q^2)^2}\right] \\
a_{\theta(J_2)} &= -\frac{12\mu J_2 R_\oplus^2}{r^4}\frac{(q\sin L - p\cos L)(q\cos L + p\sin L)}{(1+p^2+q^2)^2} \\
a_{h(J_2)} &= -\frac{6\mu J_2 R_\oplus^2}{r^4}\frac{(q\sin L - p\cos L)(1 - p^2 - q^2)}{(1+p^2+q^2)^2}
\end{aligned}
\tag{5.9}
$$

and this form is convenient for when the states are defined as the equinoctial elements [21, p. 596].

Drag

The second most significant perturbation on satellite motion near the Earth is due to atmospheric drag. At extremely low altitudes, the effects of drag may actually be

greater than those of J_2. It is important that the drag perturbation be recognized, so a basic model is presented here as it could be implemented within the confines of the problem setup discussed in this chapter. However, drag is not considered in the rest of the work of this thesis. Implementation of the drag perturbation is left to those interested in continuing in the direction of this thesis.

As with J_2, the atmospheric drag perturbation can be included in the dynamics as an additional acceleration. Aerodynamic drag acceleration is dependent upon several factors:

$$\vec{a}_{(drag)} = -\frac{1}{2}\frac{c_D A}{m}\rho v_{rel}^2 \frac{\vec{v}_{rel}}{|\vec{v}_{rel}|} \tag{5.10}$$

where

$$
\begin{aligned}
c_D &= \text{coefficient of drag} \\
A &= \text{cross-sectional area} \\
m &= \text{satellite mass} \\
\rho &= \text{atmospheric density} \\
\vec{v}_{rel} &= \text{velocity relative to the atmosphere.}
\end{aligned}
$$

Including the effects of drag complicates the problem significantly. Notice that three of the variables affecting the drag acceleration are dependent on the shape and size of the satellite in orbit. The problems considered in this thesis are satellite-independent; to consider drag, a user must specify several characteristics of a satellite. The relative velocity, $\vec{v}_{rel}$, also suggests a level of complexity, as this is not the velocity captured in the last three states of a Cartesian dynamics system. The velocity vector in Equation 5.10 is relative to the atmosphere, suggesting that knowledge of the motion of the atmosphere is required. This motion is due to the rotation of the Earth; additionally,winds might be considered on top of that rotation.

The atmospheric density, ρ, also poses its own complications, as it is extremely difficult to represent the density accurately at high altitudes. There are many factors that affect the density, including latitudinal and longitudinal variations, changes in solar radiation on account of the cycle of Sun spots, the atmospheric bulge in the direction of the Sun, winds and tides. Most significantly, ρ varies nearly exponentially with altitude. One reasonable model for atmospheric density uses the exponential:

$$\rho = \rho_0 \exp^{\left(-\frac{h-h_0}{H}\right)} \tag{5.11}$$

where ρ_0 and h_0 are a reference density and altitude, respectively. The actual altitude, h, presumably calculated by subtracting the Earth's radius from the satellite's position from the center of the Earth, may consider the non-spherical characteristics of the Earth. H is the scale height, and H, ρ_0, and h_0 can be acquired from Vallado (510), which reprints from Wertz. Many other models exist from tabulated data from several sources [45, pp. 497–514].

5.1.3 Controls

Along with the disturbing accelerations presented already, the controls contribute to the dynamical equations. The controls can be defined in multiple ways, and one must take care in defining them with the optimization routine in mind.

At the least, three components are necessary to represent the control accelerations in three orthogonal directions (whether they are in Cartesian or polar coordinates is of no consequence for this discussion). It would be easiest to use a three-vector for the thrust acceleration,

$$\vec{a}_T = [u_1 \ u_2 \ u_3] \tag{5.12}$$

where u_1, u_2, u_3 are the control values in each of the principle directions. As there is generally a restriction on the magnitude of control acceleration (thrust) that can be applied at any given time, a path constraint could be applied such that

$$\sqrt{u_1^2 + u_2^2 + u_3^2} \leq T_{max} \tag{5.13}$$

where T_{max} is the maximum allowed thrust acceleration magnitude. Most likely, the control magnitude $J' = \sqrt{u_1^2 + u_2^2 + u_3^2}$ will also appear in the cost function, as it is the magnitude of the thrust that we generally seek to minimize.

Unfortunately, this control definition creates difficulty for the optimizer, which evaluates the gradient of the cost functional. If the cost functional includes a term with the control magnitude, then the terms of the gradient coming from the partial derivatives of the controls put the controls in the denominator. For example,

$$
\begin{aligned}
\frac{\partial J'}{\partial u_1} &= \frac{\partial}{\partial u_1}(u_1^2 + u_2^2 + u_3^2)^{\frac{1}{2}} \\
&= \frac{1}{2}(u_1^2 + u_2^2 + u_3^2)^{-\frac{1}{2}} 2u_1 \\
&= \frac{u_1}{\sqrt{u_1^2 + u_2^2 + u_3^2}}
\end{aligned}
\tag{5.14}
$$

The result in Equation 5.14 may be disastrous, since the magnitude of the controls may equal zero during some interval of the time period, leading to a singularity in the gradient.

To avoid such an inconvenience, the controls can be represented with four variables instead of three, where the first three constitute a unit vector establishing the thrust direction in either Cartesian or polar coordinates, and the last variable represents the magnitude of the control acceleration.

$$\vec{a}_T = u_4[u_1 \ u_2 \ u_3] \tag{5.15}$$

The magnitude is bounded by $0 \leq u_4 \leq T_{max}$, and a path constraint is included to

constrain the unit vector.

$$\sqrt{u_1^2 + u_2^2 + u_3^2} = 1 \tag{5.16}$$

At the expense of adding an additional control variable to the problem, the term of the cost functional involving the thrust magnitude becomes

$$J' = u_4 \tag{5.17}$$

whose gradient poses no problems to the optimizer.

5.2 The Performance Index

In this work, control is minimized for orbital transfer problems. Practically, the goal is to find the most fuel-efficient way to perform an orbital maneuver. Equivalently, an optimal transfer is the one that requires the least maneuvering, the least accumulated acceleration or change in velocity. The cost functional can be set up in several different ways to accomplish this.

One way to minimize fuel is to specifically measure the fuel mass in the performance index. The mass can appear as a seventh state,

$$\dot{m} = -\beta = -\frac{T}{v_e} \tag{5.18}$$

where β is the mass flow rate, equal to the thrust divided by the exhaust velocity. Then, the performance index is simply the amount of fuel used during the maneuver:

$$J = m_0 - m_f. \tag{5.19}$$

The disadvantage of measuring the fuel mass is that the problem becomes somewhat overspecified for the scope of this research. The optimization problem now requires some knowledge on the size of the vehicle (m_f is the satellite mass without fuel) and the efficiency of the rocket thrusters (inherent in the user-defined value of the exhaust velocity). If we wish to generalize the problem, an equivalent cost functional involves the acceleration instead.

$$J = \int_{t_0}^{t_f} |\vec{a}_T| dt \tag{5.20}$$

Recall that $\vec{a}_T$ is the controlled acceleration due to thrust.

When the performance index has arguments of mass or acceleration as they do in Equations 5.19 and 5.20, then the problem is set up to allow for continuous controls as in any traditional optimal control problem. Sometimes, it may be interesting to solve directly for Δv, the instantaneous velocity change. (The distinction between these two solution forms is elaborated in Section 5.4.) In the case of instantaneous

maneuvers, the appropriate performance index minimizes total Δv.

$$J = \sum_{i=1}^{k} \Delta v_i \tag{5.21}$$

where k is the number of allowable impulsive maneuvers.

If it is desired to consider mass after a problem has been solved, then one may use the ideal rocket equation,

$$\Delta v = v_e \ln \left(\frac{m_0}{m_f} \right). \tag{5.22}$$

By specifying the initial mass and the exhaust velocity, one may determine the amount of fuel necessary to achieve a certain Δv.

5.3 Event Constraints

Whenever a condition must be satisfied at a particular instant, an event constraint is used. For example, event constraints are necessary to achieve the initial and final conditions of the problem. In the case of interior events, constraints in the Events function must also account for them. Because both Cartesian coordinates and modified equinoctial elements are used, the event constraints have some variations based on the set of states employed.

The initial conditions require six event constraints to restrict the initial value of each of the six states, whether they represent position and velocity vectors or six modified equinoctial elements.

The transfer problem solved in this thesis requires a vehicle to reach a final orbit, but it does not specify where on that orbit it must be. Therefore, at the final time, only five constraints are necessary. Whether the dynamics are expressed in Cartesian coordinates or in an element set, the final constraints are best presented in an element set, where the first five elements define the orbit. Therefore, if the dynamics are presented in Cartesian coordinates, they must be transformed within the Events function into elements (the modified equinoctial elements in this thesis) in order to formulate the final event constraints.

When solving directly for impulsive maneuvers to perform an orbital transfer, a set of additional parameters are defined in the Events function. Similar to the formulation for thrusting controls, four parameters are used to define any given Δv: three for direction and one for magnitude. An instantaneous change in velocity necessarily requires discontinuity in the states, so Δvs must occur at hard knots, which are allowed to float freely so that the optimizer can pick when the maneuvers should

occur. Six additional constraints are required at interior knots.

$$\vec{r}\,^{+} = \vec{r}\,^{-}$$
$$\vec{v}\,^{+} = \vec{v}\,^{-} + \Delta v[u_1 \; u_2 \; u_3]^T \tag{5.23}$$

Since the constraints on an impulsive maneuver are formulated in Cartesian coordinates, additional transformations would be necessary if the states were defined in equinoctial elements.

5.4 Design Choices for Specific Problems

As presented above, there are essentially two different ways to create a maneuver in the setups explored in this work. Continuous controls, which take on values at the nodes just like the states, are used to represent accelerations due to variable thrusting, continuous or finite in duration. Parameters are used to represent purely impulsive maneuvers.

Three different forms of the problem are explored under the setup restrictions. Pure Impulses are found using the parameter optimization solution technique. Impulsive Approximation uses a continuous control setting to find high-thrust solutions that approximate impulses. Finally, Finite-Burn solutions are found using the continuous control setting with significantly bounded acceleration levels, forcing maneuvers to take place over longer durations.

5.4.1 Pure Impulses

When pure impulsive solutions are desired, parameter optimization is used. The four variables for controls can either be taken out of the dynamics equations or zeroed out through bounds, so that the parameters in the event constraints solely perform the orbital transfer.

While the dynamics can be formulated in either Cartesian coordinates or orbital elements, when using parameters as the control, there is a distinct advantage to using Cartesian coordinates. Recall from Equation 5.23 that the event constraints related to instantaneous velocity changes must be formulated in Cartesian coordinates ($\vec{r}$ and $\vec{v}$). If the dynamical equations relate the equinoctial elements, then a nonlinear transformation is required for each knot to take the states from elements to vectors and then back to elements again. This is not only an inconvenience that slows down the algorithm; it also degrades the performance of the optimizer. Optimization is difficult when boundary constraint values vary nonlinearly with the states or controls. Therefore, the dynamics should be represented in Cartesian coordinates when solving for pure impulses, simplifying the formulation of event constraints and improving DIDO performance.

The four controls for continuous acceleration are not necessary for this setup alternative. In DIDO, there must be at least one control variable, however it may simply be a dummy variable in this setup. If it does not appear in the dynamics, it will not affect the solution in any way. It is acceptable to allow the useless control variable to float, but good practice would dictate forcing the variable to a single value so that the optimizer does not waste effort in changing its value.

5.4.2 High Thrust Impulsive Approximation

If continuous controls are allowed, and the upper bound on the magnitude of thrust is fairly high, then the solution of the orbital transfer problem will have an impulsive flavor. Thrusting arcs will be large in value and short in duration. Coasting arcs will constitute the majority of the transfer time. This essentially serves as the first approximation to an impulsive solution. This problem formulation can be used independently, but can work well in parallel with the purely impulsive setting. In Chapter 6, solutions with impulsive approximations are used to generate guesses for the Pure Impulse technique.

The dynamics can be written in terms of either position and velocity vectors or equinoctial elements. If it is the case that the Impulsive Approximation setup is used to supplement the Pure Impulse setup, it is logical to write the dynamics in Cartesian coordinates. In this thesis, it is used in a supplemental fashion, and consequently varies little from the Pure Impulse setup. Here, the effects of Δv parameters are zeroed out in the same way as continuous controls were zeroed out before.

5.4.3 Finite-Burn

The Finite-Burn capability is essentially the same form as the High Thrust Impulse Approximation, except for the key fact that the upper bound on thrust is lowered to the extent of showing dramatic differences in the optimal trajectory.

In Chapter 7, the finite-burn solutions are presented using dynamics in both Cartesian-coordinate and element-set dynamics. It is demonstrated that the modified equinoctial elements are significantly more flexible in the continouous-control setting. With this advantage, an initial guess can be quite poor, and DIDO will still be able to find a solution. In Cartesian coordinates, the amount of variation between the guess and the solution is more limited. While the disadvantage of this is clear, the reduced flexibility can actually be a benefit if the user wants to find a solution with characteristics not defined within the constraints. With the Finite-Burn capability, it is wise to use both sets of dynamics in tandem.

Soft knots are quite useful when seeking finite burn solutions. The trajectory time is generally longer, and it is beneficial to break up the nodes with some interior knots to more easily increase the number of nodes in the discretization.

5.5 Other Tools

It is implied in Secion 5.4.2 that there is a benefit to running the Pure Impulse setup in parallel with the High Thrust Impulsive Approximation setup. Consequently, it was beneficial to develop a routine that easily takes a solution from the Impulsive Approximation and converts to a guess for the Pure Impulse setup. The routine used here employs quadrature techniques to transform a finite high-thrust burn into an impulse. While the resulting trajectory will not meet the constraints due to approximation errors, the transformed controls serve as a good guess for the parameters used in the Pure Impulse setup.

In this chapter, the orbital transfer problem is defined as it is solved in DIDO. In the next two chapters, DIDO's capability to solve orbital transfer problems is demonstrated for several types of transfer scenarios. The impulsive problem is examined first, and the finite-burn problem follows.

Chapter 6

Impulsive Burn Orbital Transfer

No maneuver is truly impulsive; any thruster must burn for some duration of time at some finite level of thrust. For an orbital transfer problem, where velocity change requirements are very large, thrusters may be required to burn for minutes at a time. If this is the case, then why is it important to understand orbital maneuvers in terms of fictitious impulsive burns? Depending on the transfer being accomplished, even a full 60 seconds of burning may likely account for less than one percent of the total transfer time. In this context, an impulsive solution might serve as a reasonable first approximation.

Equally as important, solving an impulsive problem helps one gain a better intuition as to the nature of the solution. The velocity change, or Δv, necessary for a transfer generally provides a good understanding of the cost of a solution. It is with this motivation that this chapter is devoted to impulsive burn solutions to transfer problems.

The Legendre Pseudospectral Method, as implemented in DIDO, is used to find optimal impulse times, magnitudes, and directions. DIDO was initially designed, however, for use in continuous models. Therefore, it is well suited for high- or low-thrust, continuous- or finite-burn problems. In a high-thrust, finite-burn regime, though, even a continuous set of controls begin to look like impulses. Knowing that an optimal solution would approach an impulse, one is motivated to stretch the capability of DIDO to actually find impulses. Initially, one would think to do this by increasing the number of knots and nodes used in a problem, but through experience it is determined that the addition of parameters is ideal for representing impulses.

In this chapter, the capability is demonstrated first by exploring a range of problems to which an optimal solution is already known. Then the method is applied to more difficult problems to display its practicality in real world situations. The chapter is concluded by documenting some lessons learned in providing reasonable guesses to increase confidence in the value of a solution.

6.1 Capability Demonstration

In this section, results are presented on a series of transfer problems the solution to which can be found analytically. The Legendre Pseudospectral technique is applied to several variations of a Hohmann Transfer problem at first, then to some simple plane change problems, and finally to some combined size and plane change problems. Since the optimal Δv sequence is already known for each of these problems, the object of this study was to assess the robustness of the problem setup. Can this method find the optimal solution to an impulsive orbital transfer problem? If so, under what conditions can a solution be found? This question refers not only to setup conditions, like the number of nodes or knots used to solve a problem, but also to the issue of how good an *a priori* guess must be in order to find a solution. In the demonstration below, optimal solutions are found under a wide range of conditions, and the imposed constraints and conditions are identified. Section 6.4 is devoted to summarizing the conclusions as to the spectrum of appropriate guesses for finding optimal solutions with this method.

Recall from Chapter 5, that four parameters are set to represent the magnitude and direction of an impulsive burn at each hard knot. All of the knots are allowed to float freely (therefore not specifying burn times) with the exception of the first knot, which must occur at $t_0 = 0$. However, burns are not required at any knot (the parameters representing the magnitude of Δv at each knot are allowed to equal 0). This fact is useful for two reasons:

1. A transfer may require an initial coasting arc, so the burn magnitude at the first knot must be zero.

2. A user may not know the number of impulsive burns required for optimal transfer. Therefore it is important to allow for more knots than anticipated impulses.

6.1.1 Hohmann Transfer

The Hohmann Transfer is a well understood maneuver used to change the size and shape of an orbit. Walter Hohmann suggested that the most efficient transfer between circular, coplanar orbits used tangential burns. Later, others discovered that the Hohmann transfer is also the most efficient for coapsidal elliptical orbits. (Vallado 278) The transfer ellipse shares its periapsis with the smaller orbit and its apoapsis with the larger orbit. A tangential burn occurs at both of these two locations; the first takes a vehicle from its original orbit and places it on the transfer ellipse, and the second places the vehicle in the final orbit.

Several types Hohmann transfers are investigated below, each starting from a smaller orbit and going to a larger one. The ideal solution to each uses the exact same transfer ellipse, and the difference between each case lies in the eccentricity of initial and final orbits.

From Circular Orbit to Larger Circular Orbit

In this first example, DIDO is used to transfer between the orbits defined in Table 6.1. The transfer requirements are displayed in Table 6.2.

Table 6.1: Circular-Circular Hohmann Transfer Element Set

Orbital Elements	Initial	Final
a	6,570,000 m	42,160,000 m
e	0.0	0.0
i	0.0°	0.0°
L	0.0°	(Not Specified)

Table 6.2: Circular-Circular Hohmann Transfer Requirements

Δv_1	2456.90 m/s
Δv_2	1478.13 m/s
$t_f - t_0$	315.41 min

Since this problem requires two distinct burns, it can be solved with a minimum of two hard knots (recall that a hard knot must exist at each terminal), with a burn occurring at each knot. A reasonable number of nodes (perhaps 100) between the two knots will give DIDO sufficient flexibility to converge to the solution provided a good guess. It is more interesting, however, to present results where several interior knots also exist.

In Figure 6-1, an example is presented where a total of four hard knots were used. A random guess is generated using integer values for scaled units of time, velocity, and direction to simulate a transfer to which the solution is not known, but there is an intuition as to how the solution might look. Not shown in the figure is that the

Table 6.3: Circular-Circular Hohmann Transfer: Random Guess

	Guess		Solution	
Knot	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	2831.10	0	2456.90
2	140.82	0	45.23	0
3	281.63	0	207.55	0
4	422.45	0	315.43	1478.13
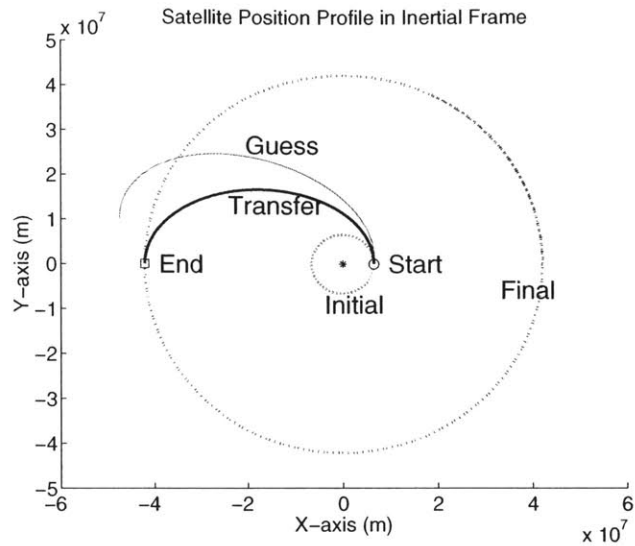
Figure 6-1: Circular-Circular Hohmann Transfer: Random Guess

guess transfer trajectory contains an out-of-plane component that is nulled out by the optimizer.

The time and Δv values of the same problem are presented in Table 6.3. It is clear that the optimizer was given four opportunities (knots) to place impulsive manuevers, and it opted to use only the first and the last of them. (Since this is only a two-body problem, one must expect an impulse at the last knot: once the final Δv is imposed, the transfer is over.)

In the second example, depicted in Figure 6-2, the guess trajectory and parameters match the initial orbit. This scenario represents a case where one may have no *a priori* intuition as to the nature of the solution. Therefore, the guess calls for no impulses, and the satellite remains on the initial orbit.

Table 6.4 shows that at every knot of the guess, 0 Δv is imposed. The solution, however, differs from the first solution presented. With this one, there are still two impulsive maneuvers, but the first one occurs on the second knot instead of the first. Therefore, the satellite trajectory has a coasting arc before it initiates the first transfer maneuver.

The results presented here bring up an interesting point. For a Hohmann Transfer between two circular orbits, there are an infinite number of optimal transfer trajectories. The transfer may be initiated from any point on the initial orbit, and it will still be optimal if the transfer angle is 180°. In the cost function, only Δv is being minimized, so the time of flight is allowed to float freely without affecting the cost. Because time is not a factor in this problem, both solutions are equally feasible and equally optimal.

66
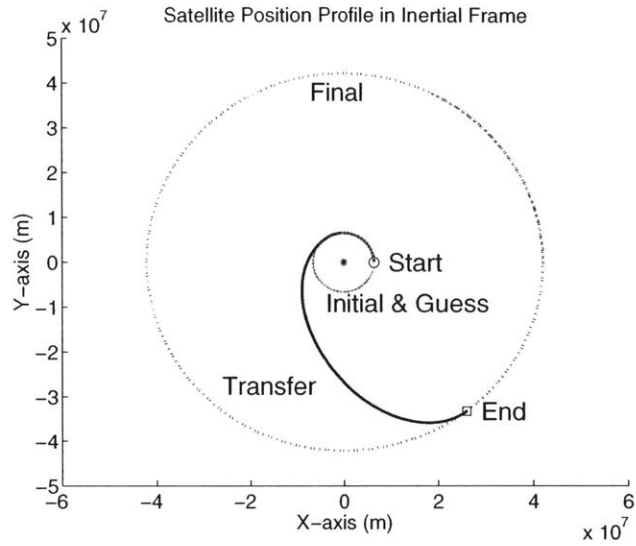
Figure 6-2: Circular-Circular Hohmann Transfer: Guess Trajectory = Initial Orbit

Table 6.4: Circular-Circular Hohmann Transfer: Guess Trajectory = Initial Orbit

Knot	Guess		Solution	
	$t\ (min)$	$\Delta v\ (m/s)$	$t\ (min)$	$\Delta v\ (m/s)$
1	0	0	0	0
2	28.16	0	31.42	2456.89
3	56.33	0	70.90	0
4	84.49	0	346.83	1478.13

67

From Elliptical Orbit to Larger Elliptical Orbit

If elliptical orbits become the initial and target orbits, then there is no longer an infinite number of optimal trajectories (provided that time is bounded in a reasonable fashion). The two coplanar orbits are placed along the same line of apses and are sized such that the same trajectory is required as shown in the Circular-Circular example.

Table 6.5: Elliptical-Elliptical Hohmann Transfer Element Set

Orbital Elements	Initial	Final
a	7,300,000 m	24,800,000 m
e	0.1	0.7
i	0.0°	0.0°
$\tilde{\omega}$	0.0°	0.0°
ν	270.0°	(Not Specified)

For a proper Hohmann Transfer, the most desired requirement is that the satellite initiate the transfer at the perigee of the smaller orbit. To test the package's capability to find that point, the satellite has a true anomaly of 270 degrees at the epoch of the DIDO run. Therefore, we now require an initial coasting arc in the optimal trajectory; in Table 6.6, that coasting arc is the time between t_0 and t_1.

DIDO results are posted in Figure 6-3 and Table 6.7. The *a priori* guess used in this example again reflects having some intuition into the problem. Specifically, that intuition results in a guess with an initial coasting arc, although the duration of the arc is unknown.

The numbers in Tables 6.6 and 6.7 do not exactly match up, although they are quite close. The run presented here, using 200 nodes and 3 knots, produced a result whose cost is accurate to 0.015 m/s. The transfer angle in the DIDO converged solution is 180.74 degrees. If this solution were resubmitted as the guess for a new DIDO run, the accuracy of this solution would surely improve.

Table 6.6: Elliptical-Elliptical Hohmann Transfer Requirements

Δv_1	2076.72 m/s
Δv_2	87.46 m/s
$t_f - t_1$	315.41 min
$t_1 - t_0$	22.58 min
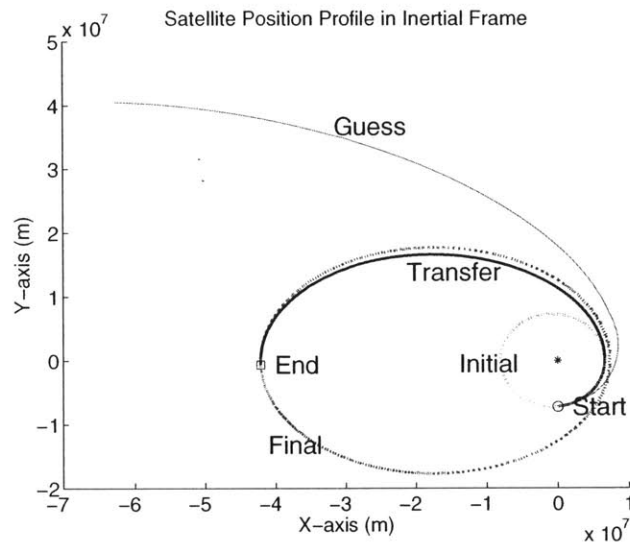
Figure 6-3: Elliptical-Elliptical Hohmann Transfer: Random Guess

Table 6.7: Elliptical-Elliptical Hohmann Transfer: Random Guess

	Guess		Solution	
Knot	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	1258.18	0	0
2	42.24	2516.36	22.60	2076.71
3	422.45	1258.18	343.91	87.49

69

6.1.2 Plane Change

Another type of standard problem worth investigating to validate the method is the simple plane change. Here, the initial and final orbits of the transfer have the same size and shape; they only vary in their orientation. Most importantly, each orbit pair has two points in common, making a locally optimal transfer a one-burn maneuver at an intersection. Two circular orbit plane changes are examined: one which changes only the inclination and one that changes the inclination and the ascending node. Additionally, an elliptical orbit with an inclination change is shown.

Inclination Change on a Circular Orbit

For a simple inclination change, the element set is presented in Table 6.8 [1]. In Figure 6-4, the initial and final orbits are shown to depict more clearly the orientations of the orbits, as this plays into generating a reasonable guess for a solution. Note that there are (again) an infinite number of optimal solutions. Theoretically, a satellite can coast for many revolutions, provided that a single maneuver of the right direction and Δv occurs at an intersection. Limiting the solution space to transfers less than a revolution in length, then there are two optimal solutions, single impulses at either orbit intersection. The specifics of these two solutions are summarized in Table 6.9.

Table 6.8: Inclination Change Transfer Element Set

Orbital Elements	Initial	Final
a	6,570,000 m	6,570,000 m
e	0.0	0.0
i	0.0°	10.0°
Ω	45.0°	0.0°
u		(Not Specified)

Table 6.9: Inclination Change Transfer Requirements

Δv	1357.73 m/s
$t_f - t_0$	33.12 min
$t_f - t_0$	77.29 min

Three different guesses are presented in this section to demonstrate what constitutes a good guess. In this particular case, the guessed direction of an impulsive maneuver has a significant impact on the likelihood for convergence.

---

[1] Note that u is the alternate orbital element, the argument of latitude, which equals $\omega + \nu$
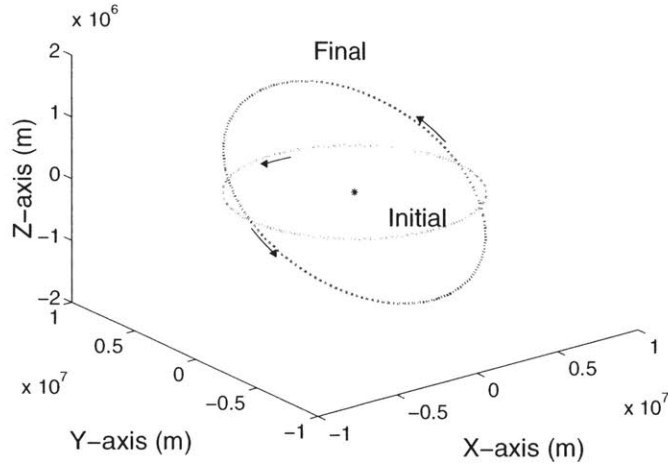
Figure 6-4: Inclination Change Transfer: Initial & Final Orbits

The first guess is presented in Figure 6-5 with results in Table 6.10. (As the converged trajectory simply remains along the initial orbit until it crosses the final orbit, it is not shown in the figure.) A guess is provided for a 3-knot problem expecting two coasting arcs and a burn at the last knot in the $-z$ direction in the vicinity of the descending node. The Δv of the guess, while not exactly correct, does place the guess trajectory in an orbit visually similar to the targeted final orbit. When the guess is processed through DIDO, the converged solution conducts a single maneuver at the first intersection between the initial and final orbits.

Table 6.10: Inclination Change Transfer: Single Impulse Guess $(-z)$

	Guess		Solution	
Knot	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	0	0	0
2	21.12	0	21.66	0
3	42.24	629.09 $(-z)$	33.12	1357.73
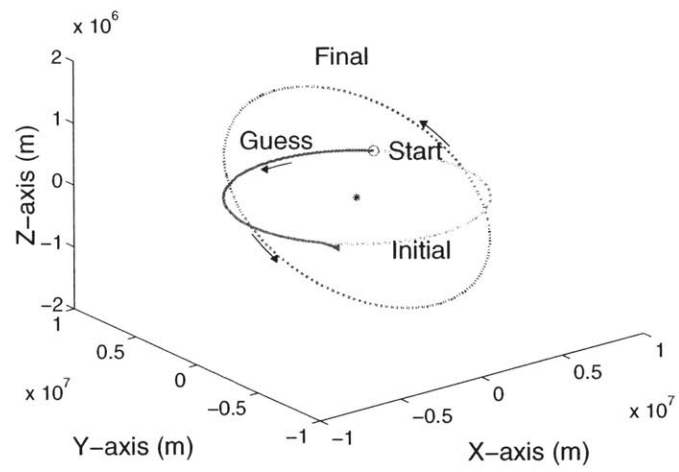
Figure 6-5: Inclination Change Transfer: Single Impulse Guess ($-z$)
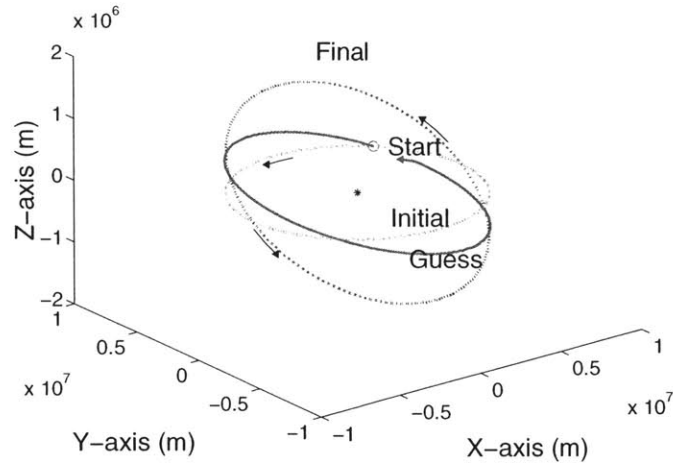
Figure 6-6: Inclination Change Transfer: Two-Impulse Guess $(+z, -z)$

In the next guess, presented in Table 6.11, a 3-knot guess applying two burns is used. DIDO was unable to find a solution for the problem given this guess, and it is important to understand why this happened.

Now, the knots are placed further out in time, with the final manuever happening much closer to the second intersection of the initial and final orbits. This is shown more clearly in Figure 6-6. The guessed burn direction is extremely important for this, especially with the burn occuring on the third knot. (Understanding that the burn on the first knot is going to be zeroed out, anyway, it's direction is not as much of a concern.) Like before, the final burn is guessed in the $-z$ direction, but because of the guess in the time of that knot, that burn is essentially in the opposite direction of the target orbit in that vicinity (its ascending node). For this case, DIDO was not able to find a solution.

It is understood why DIDO was not able to find a solution to the problem given

Table 6.11: Inclination Change Transfer: Two-Impulse Guess $(+z, -z)$

	Guess		Solution	
Knot	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	629.09 $(+z)$	—	—
2	42.24	0	—	—
3	84.49	629.09 $(-z)$	—	—

73

this guess: the guess provides no support through engineering intuition regarding the nature of the solution. This conclusion can be supported easily in two ways. First, leaving the burn directions as they are, the guessed times of those knots could be altered to put the final knot more near the intersection at the descending node. Alternatively, leaving the times as they are now, the guessed burn direction could be reversed. Results from the second option are presented next.

To understand why the solution was infeasible provided the guess from Table 6.11, the guess from Table 6.12 was processed. The only difference between the two guesses is the direction in which the burns are guessed: their magnitudes and times are unaltered. Figure 6-7 shows this guess.

DIDO has no difficulty finding a solution to this transfer. Now, the final burn takes place in a direction more reasonable as compared to the closest optimal solution. The converged solution appropriately coasts to the second intersection (the ascending node of the final orbit) and conducts a proper burn whose cost is accurate to 2.2×10^{-6} m/s.

Results like the ones presented on the simple inclination change provide a great deal of understanding into what it takes to ensure that a solution can be found to a given transfer problem. See Section 6.4 for a summary of the conclusions found in this work on guess selection.
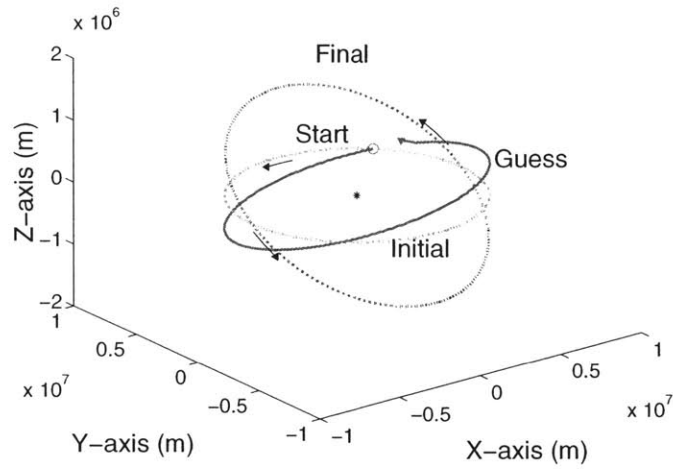
Figure 6-7: Inclination Change Transfer: Two-Impulse Guess $(-z, +z)$

Table 6.12: Inclination Change Transfer: Two-Impulse Guess $(-z, +z)$

Knot	Guess		Solution	
	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	629.09 $(-z)$	0	0
2	42.24	0	42.22	0
3	84.49	629.09 $(+z)$	77.29	1357.73

Inclination and Ascending Node Change on a Circular Orbit

The method's ability to solve an inclination change is stretched only slightly by solving an inclination *and* ascending node change problem. Essentially, it is the same problem: two intersecting points exist, and the optimizer must find one of them and place the proper Δv there in the proper direction. This problem is just more difficult for the user because the intersection points between the initial and final orbits no longer lie along the ascending or descending nodes.

The specifics of this problem are enumerated in Table 6.13, and the requirements for an optimal solution are listed in Table 6.14.

Table 6.13: Inclination/Node Change Transfer Element Set

Orbital Elements	Initial	Final
a	6,570,000 m	6,570,000 m
e	0.0	0.0
i	45.0°	10.0°
Ω	0.0°	90.0°
u	0.0°	(Not Specified)

Table 6.14: Inclination/Node Change Transfer Requirements

Δv	6069.85 m/s
$t_f - t_0$	40.73 min

Figure 6-8 shows the initial and final orbits (and their directions of travel), and the trajectory corresponding to a guess applying limited engineering judgment. As presented in Table 6.15, the guess calls for two burns when only one is needed. A solution is easily found, placing a burn at the first orbit intersection, whose cost is accurate to 3.0×10^{-5} m/s.
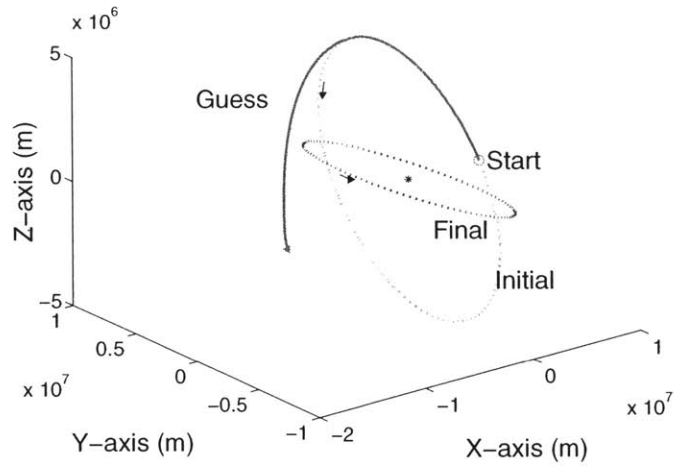
Figure 6-8: Inclination/Node Change Transfer: Random Guess

Table 6.15: Inclination/Node Change Transfer: Random Guess

	Guess		Solution	
Knot	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	0	0	0
2	21.12	2516.36 $(-x)$	22.64	0
3	50.69	1258.18 $(+x)$	40.73	6069.85

77

Inclination Change on an Elliptical Orbit

In this last plane change example, we perform an inclination change on an elliptical orbit. This particular example becomes interesting because, in the interval of one revolution, there is only one optimal solution, even though there are as before two intersections between the initial and final orbits. The orbital elements for the two orbits are listed in Table 6.16.

Table 6.16: Inclination Change Transfer Element Set (Elliptical)

Orbital Elements	Initial	Final
a	24,800,000 m	24,800,000 m
e	0.7	0.7
i	0.0°	10.0°
Ω	45.0°	0.0°
ω		45.0°
ν	270.0°	(Not Specified)

As with the circular problems, two intersections exist between the initial and final orbits. However, these two elliptical orbits have been oriented in such a way so that the orbital radius (and consequently, velocity) is different at each. Because the cost of plane change is directly proportional to the orbital velocity at the point of the change, it will be optimal to conduct the plane change maneuver for this tranfer at the higher-altitude intersection. Table 6.17 lists the transfer requirements at both intersections. The true anomalies of the low-altitude and high-altitude intersections are, respectively, 315 degrees and 135 degrees. Notice that at epoch the satellite is at 270 degrees. Intentionally, the problem has been set up so that the optimal solution requires the trajectory to coast through the first intersection and conduct the burn at the second.

Table 6.17: Inclination Change Transfer Requirements (Elliptical)

Δv	494.19 m/s
$t_f - t_0$	121.67 min
Δv	1462.91 m/s
$t_f - t_0$	19.32 min

In Table 6.18, the guess and solution parameters are presented for the inclination change on an elliptical orbit. A random guess, using integer values of scaled units of
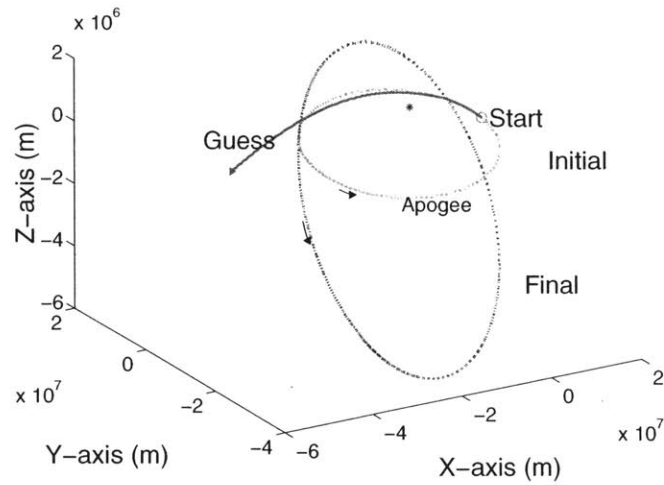
Figure 6-9: Inclination Change Transfer (Elliptical): Random Guess

Table 6.18: Inclination Change Transfer (Elliptical): Random Guess

Knot	Guess		Solution	
	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	0	0	0
2	21.12	2516.36 $(-x)$	26.43	0
3	168.98	1258.18 $(+x)$	121.70	494.19

time and velocity, is used in the process, and without any difficulty DIDO resolves the trajectory into the correct solution. The cost is accurate to 1.0×10^{-7} m/s.

79

6.1.3 Combined Plane Change

The final set of standard problems that were tested under the parameter optimization setup of DIDO is the combined plane change. For this set of problems, the transfer requires a size (and maybe shape) change, as well as a plane change. It is basically a combination of the two sets presented already. The problems presented here are still coapsidal (a factor for the elliptical problem), but more importantly now that inclination changes will also be made, the line of apses is in union with the line of nodes.

The solution to a problem like this will resemble that of a Hohmann transfer. A two-burn solution is required, and those burns will occur along the line of nodes. What makes these problems interesting, however, is how the inclination change is made. It is well understood that a single burn should contribute both to the size change and to the inclination change (as opposed to conducting individual burns for each), but the real question is how. Specifically, to how much of the inclination change should each of the two "Hohmann" burns contribute? Because plane change maneuvers are cheaper at higher altitudes, one may expect all of the inclination change to occur on the second (higher-altitude) burn. In reality, a combined plane change follows more of a 10-90 rule, where it is optimal to conduct about 10% of the plane change on the lower burn, and the other 90% on the higher burn.

The next two problems were solved to determine whether the optimizer could discover this result on its own. Results are presented for a circular-circular transfer and an elliptical-elliptical transfer.

From Circular Orbit to Larger, Inclined Circular Orbit

In this transfer, a satellite begins in an equatorial low Earth orbit and must end in an inclined geosynchronous orbit. The elements associated with each are summarized in Table 6.19.

Table 6.19: Size and Inclination Change Transfer Element Set (Circular)

Orbital Elements	Initial	Final
a	6,570,000 m	42,160,000 m
e	0.0	0.0
i	0.0°	10.0°
Ω	45.0°	0.0°
u		(Not Specified)

For an impulsive manuever which combines size and plane change, the Law of Cosines is used to determine the magnitude of the burn. At the location of the first

burn,

$$\Delta v_1 = \sqrt{v_1^2 + v_{1*}^2 - 2v_1 v_{1*} \cos(\Delta i_1)} \tag{6.1}$$

where v_1 is the velocity of the initial orbit, v_{1*} is the velocity of the transfer orbit at perigee, and Δi_1 is the amount of inclination change made with the burn. A similar equation is used for the second burn with v_{2*} (the transfer velocity at apogee) and v_2 (the final orbit velocity). To determine the amount of inclination change per burn, the following equation is iterated.

$$\sin(s\Delta i) = \frac{\Delta v_1 v_2 v_{2*} \sin\left[(1-s)\Delta i\right]}{\Delta v_2 v_1 v_{1*}} \tag{6.2}$$

where $\Delta i_1 = s\Delta i$ and $\Delta i_2 = (1-s)\Delta i$, and s varies from 0 to 1. These calculations are optimized independent of DIDO, minimizing total Δv.

In Table 6.20, the optimal numbers for this transfer are presented, including inclination changes. Notice also that a coasting arc is required before the satellite reaches the line of nodes from which to burn.

Table 6.20: Size and Inclination Change Transfer Requirements (Circular)

Δi_1	0.9027°
Δi_2	9.0973°
Δv_1	2460.92 m/s
Δv_2	1519.34 m/s
$t_f - t_1$	315.41 min
$t_1 - t_0$	33.12 min

In Figure 6-10 and Table 6.21, results are presented from processing this case in DIDO. The figure shows both a three-dimensional plot and a variable evolution plot, both of which convey the same information. Notice that three trajectory lines are included. The original guess (Guess 1) consists of random values for Δv times, directions, and magnitudes. In fact, the original guess places the satellite on a hyperbolic trajectory. From this guess, DIDO converges to a feasible, but not optimal, solution (called "2"). This solution does not conduct burns along the line of nodes, nor does it divide the inclination change correctly. However, when this solution is used as a new *a priori* guess, then DIDO is able to find the globally optimal solution (3).

Satellite Position Profile in Inertial Frame
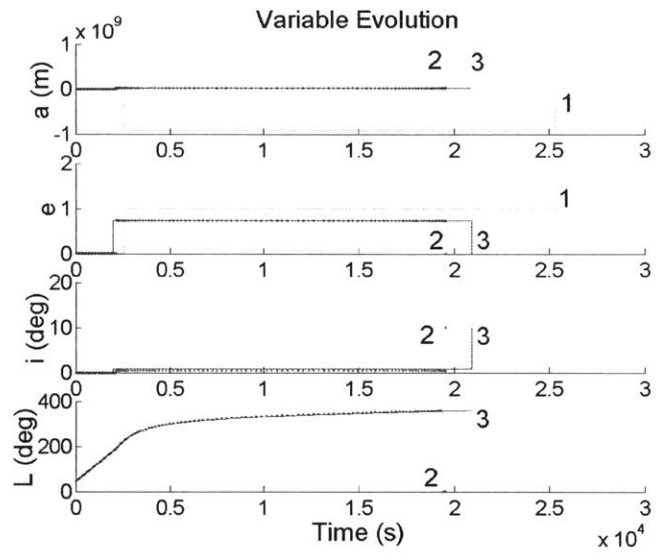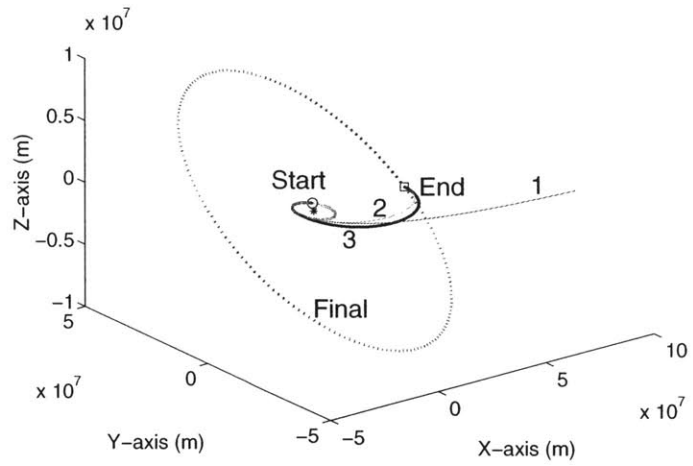
Figure 6-10: Size and Inclination Change Transfer: Random Guess

The characteristics seen in this example are not unusual. When a relatively un-realistic guess is used as a starting point (as in this case), the optimizer's primary goal is to find *a* solution that meets the constraints (i.e. a feasible solution). It is quite likely that the optimizer will settle on a trajectory that merely meets the KKT conditions when it must start far away from feasibility. However, the trajectory may not be an optimal solution to the control problem. If the optimizer can begin from a set of states and controls that already meet the constraints (as in Guess 2), there is more freedom for the routine to search the realm of feasible solutions for the truly optimal solution. In general, it is good practice to reprocess a converged solution to increase the chances that it has landed on a global minimum.

Table 6.21: Size and Inclination Change Transfer: Random Guess

Knot	1		2		3	
	t (*min*)	Δv (*m/s*)	t (*min*)	Δv (*m/s*)	t (*min*)	Δv (*m/s*)
1	0	0	0	0	0	0
2	42.24	3774.53	35.91	2511.07	33.12	2460.87
3	422.45	1258.18	326.11	1553.98	348.53	1519.38

From Elliptical Orbit to Larger, Inclined Elliptical Orbit

A similar combined plane change is next performed on elliptical orbits. The elements of the orbits are listed in Table 6.22.

Table 6.22: Size and Inclination Change Transfer Element Set (Elliptical)

Orbital Elements	Initial	Final
a	7,300,000 m	24,800,000 m
e	0.1	0.7
i	0.0°	10.0°
Ω	0.0°	0.0°
ω		0.0°
ν	270.0°	(Not Specified)

To discover analytically what the optimal transfer is for this problem, the same method is used as before, employing Equations 6.1 and 6.2. Because the velocity requirements for the elliptical problem are so very different from the circular problem, so are the inclination change requirements. It is interesting, as shown in Table 6.23, how much more inclination change is applied at the first burn.

Table 6.23: Size and Inclination Change Transfer Requirements (Elliptical)

Δi_1	2.1965°
Δi_2	7.8035°
Δv_1	2106.13 m/s
Δv_2	239.69 m/s
$t_f - t_1$	315.41 min
$t_1 - t_0$	22.58 min

In Figure 6-11 and Table 6.24, the results of this transfer from DIDO are presented. A random guess is used, and from here the optimizer is able to converge on the correct solution immediately (processing additional guesses is not necessary here).

Comparing the results here to that of the circular combined plane change, there are two primary reasons why this run was somewhat more successful. First, the guess was technically a little better. The guess for the elliptical case did not place the satellite into a hyperbolic trajectory. Although the guess was far from meeting the constraints, it did maintain an ellipse (although highly eccentric), and because of this it was a better guess. As well, the optimal solution to the elliptical problem falls into a deeper valley in the spectrum of feasible solutions than does the circular problem.

84

Satellite Position Profile in Inertial Frame
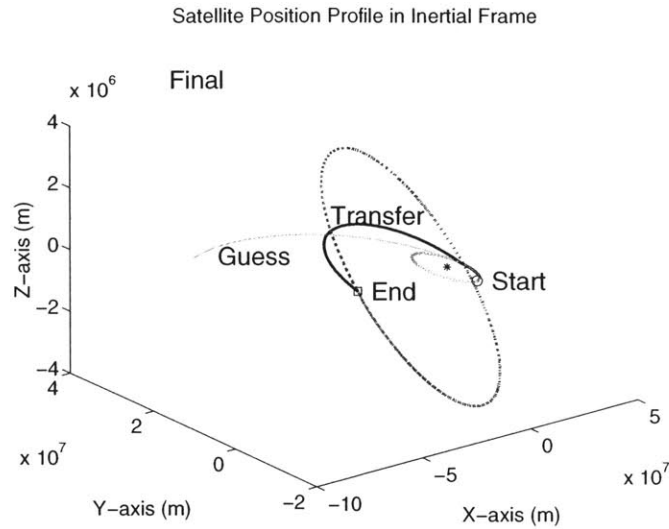
Figure 6-11: Size and Inclination Change Transfer (Elliptical): Random Guess

Specifically, two factors, instead of one, mandate the burn locations. The maneuvers must occur on the line of nodes, as before, but because of the introduced eccentricity, maneuvers now must also occur on the lines of apsides, which is conveniently the same. Therefore, to optimize both inclination and size/shape changes, even if it were to do so separately, the optimizer is more firmly motivated to drive the maneuver times to the right places.

Table 6.24: Size and Inclination Change Transfer (Elliptical): Random Guess

Knot	Guess		Solution	
	$t\ (min)$	$\Delta v\ (m/s)$	$t\ (min)$	$\Delta v\ (m/s)$
1	0	0	0	0
2	21.12	2516.36 $(-y)$	22.58	2106.17
3	422.45	1258.18 $(+y)$	338.00	239.65

85

6.1.4 The J_2 Perturbation

Nodal Regression on an Inclined Circular Orbit

Until now, the transfer problems presented have employed two-body mechanics only. In Chapter 5, however, a way to implement the J_2 perturbation was presented. Here, the implementation is validated with a simple example. In Table 6.25, the element sets are summarized for the example.

Table 6.25: Nodal Regression Transfer Element Set

Orbital Elements	Initial	Final
a	6,570,000 m	6,567,082.663994 m
e	0.0	0.001567467542855
i	45.0°	44.9872031511858°
Ω	270.0°	266.284984189246°
ω .	0.0°	203.215201836724°
ν		(Not Specified)

A set of initial conditions pertains to an orbit with a substantial inclination to ensure that the effects due to J_2 cannot be ignored. Using a Runge-Kutta integrator, the conditions are propagated forward without control for three Schuller periods, accounting for the perturbation, and the final states from the propagation establish the final orbital elements listed in the table. Notice that even in the absence of controls, there is a significant alteration in the orbital elements, most notably in the semi-major axis and in the longitude of the ascending node (which is expected when J_2 is present).

The object of this validation case was to provide DIDO with a guess that included some sort of impulsive maneuver, and ensure that it can zero it out; DIDO should find a coasting solution with zero cost. Table 6.26 presents the results from this test.

Table 6.26: Nodal Regression Transfer

Knot	Guess		Solution	
	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	0	0	0
2	84.49	125.82	80.49	0
3	253.47	0	253.47	0

This is clearly a success: DIDO was able to completely zero out the Δv for this case. The J_2 perturbation certainly affects the solution, but it does not affect the

optimizer's ability to find it.

6.2 Application

So far, a series of standard problems have been presented. They have been academic in nature, in that the impulsive solution to each could be found by hand by applying the equations found in a standard astrodynamics textbook. This has been useful to validate the technique, to be sure that the solutions that this technique yields are indeed optimal. However, most problems are generally not as easy to solve.

Also implied throughout the development so far has been the fact that some sort of engineering judgment or problem intuition may be required. But if the problem is not standard, from where does this intuition come? For a general real-world problem, one may not even know how many impulses will be required for an optimal transfer or what the basic shape of the transfer trajectory will be.

The solution technique presented so far may not be successful without some sort of intuition. And while it is sometimes the case that a bad guess will yield a nonoptimal, but feasible, solution, often it is the case that a bad guess will produce no solution at all. This, of course, is the situation we wish to avoid.

In this section, a solution method is presented for these potential real-world problems the solutions to which may not be readily apparent. Element sets have been taken from NORAD Two Line Element Sets to give more of a flare of reality. For each of the transfers shown here, a satellite begins from the orbit for the International Space Station (ISS). The orbital elements were recorded on June 16, 2002.

6.2.1 From LEO (ISS) to LEO (Sun-Synchronous)

In this scenario, a satellite is transferring from the ISS orbit to a sun-synchronous low Earth orbit. The data for the target orbit in Table 6.27 was recorded on June 13, 2002 for a British nanosatellite.

Table 6.27: LEO-LEO Transfer Element Set

Orbital Elements	Initial	Final
a	6,772,290.24534 m	7,062,996.85533 m
e	0.0007083	0.0011147
i	51.6390°	98.2208°
Ω	58.5505°	120.0745°
ω	238.2837°	282.0257°
ν	261.4820°	(Not Specified)

Satellite Position Profile in Inertial Frame
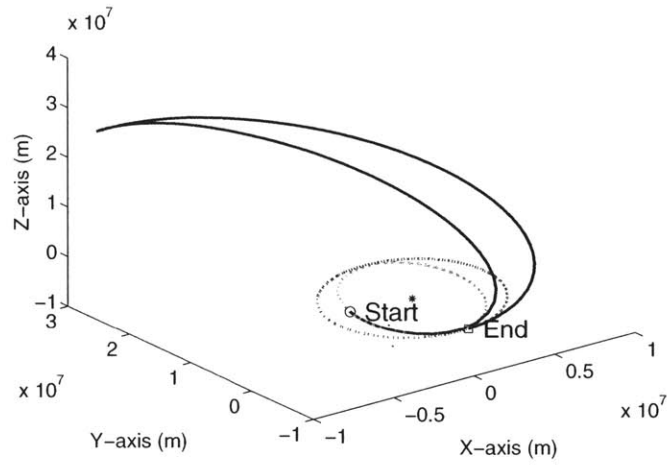
Thrust Acceleration Profile
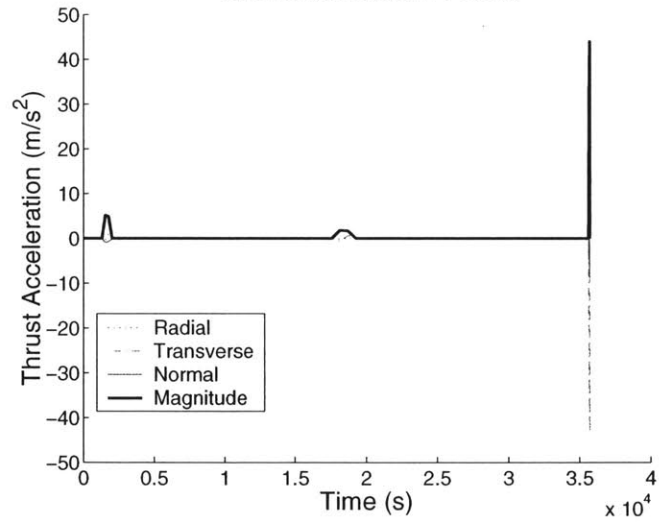
Figure 6-12: LEO-LEO Transfer: Continuous Solution

Here are two orbits, although both LEO, with nothing in common. They differ slightly in size and shape, and significantly in plane and orienation. The transfer requires a moderately inclined orbit to go retrograde. Clearly, the proper way to perform this transfer is not intuitive.

The solution techique presented in this chapter so far has used the parameter optimization capability of DIDO to solve directly for Δvs. Without a good guess, though, this technique should be avoided. The method used to find an impulsive solution to this problem, then, is to first approximate an impulsive solution in a high-thrust, continuous control setting. To simplify the entire process, the necessary terminal hard knots are the only knots used. Using this setting, a first approximation to the impulsive solution results, identifying a shape to the trajectory, the number of burns that may be necessary, and an approximation for the times of those burns.

For example, in Figure 6-12, the resulting trajectory-control set from a run in the continuous setting is shown. Just by looking at these results, it is clear that DIDO prefers a 3-burn solution: the first places the satellite in a large transfer orbit, the second conducts the majority of the plane change, and the third completes the transfer. From this result, it is clear that to run this case in the parameter optimization setting, four knots are necessary. One will be at the initial terminal (with zero Δv), and the other three should capture the burns.

A quadrature routine has been developed to transform the high-thrust, continuous solution into an approximate impulsive solution. The resulting parameterized trajectory will have values for Δv representative of the continuous controls (the accumulated Δv will be identical), but will most likely not meet the constraints. However, it can serve as a reasonable first guess in the parameter-optimization setup. In Figure 6-13, the original continuous-control trajectory is shown with its transformed parameter-control trajectory.

Figure 6-14 and Table 6.28 summarize the results for the LEO-LEO transfer when solving for impulses directly. Four different sets of data are represented.

> 1. Parameterized Guess from Continuous. This is the trajectory that results from running a quadrature routine on a continuous result to approximation an impulsive solution. (Cost: 6722.15 m/s)
>
> 2. DIDO Solution and Improved Guess. Processing (1) yields this first impulsive solution. Unlike (1), this result meets the constraints, although its cost for doing so is steep. This transfer trajectory also takes significantly less time to complete. (Cost: 8724.22 m/s)
>
> 3. DIDO Solution and Improved Guess. Processing (2) yields a second impulsive solution. This trajectory meets the constraints, while increasing the time of flight and lowering the cost. This is a good solution, and it hits the maximum bound for time. (Cost: 6562.63 m/s)
>
> 4. DIDO Solution. Opening up the time bound and processing (3) yields this final solution, believed to be very near optimal. (Cost: 6549.56 m/s)
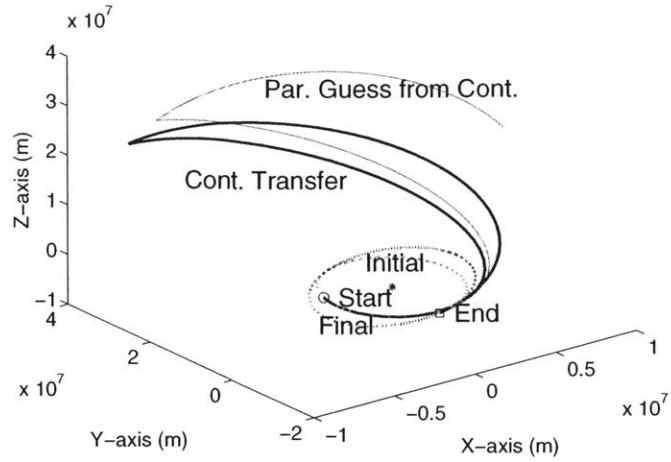
Figure 6-13: LEO-LEO Transfer: Continuous Solution with Transformed Parameterized Guess)

In the figure, the trajectory associated with (3) is not shown, but its trajectory is very similar in shape to the final solution.

The progression of solutions is similar to results presented in Section 6.1.3. When the guessed trajectory/parameter pair does not meet the constraints, the optimizer's priority is only to find a feasible solution. In this case, to converge on trajectory (2), the transfer time was lowered in the solution. After the constraints were met, then there was latitude to increase the transfer time again. Having a continuous, high-thrust solution helps one to understand that trajectory (2) is not yet optimal; its cost was significantly higher than the original. Notice also that the final impulsive cost (4) is less than the cost of the continuous solution (1). One should expect this, as the accuracy of the continuous solution is limited by the necessary control bounds and by the discretization.

Satellite Position Profile in Inertial Frame
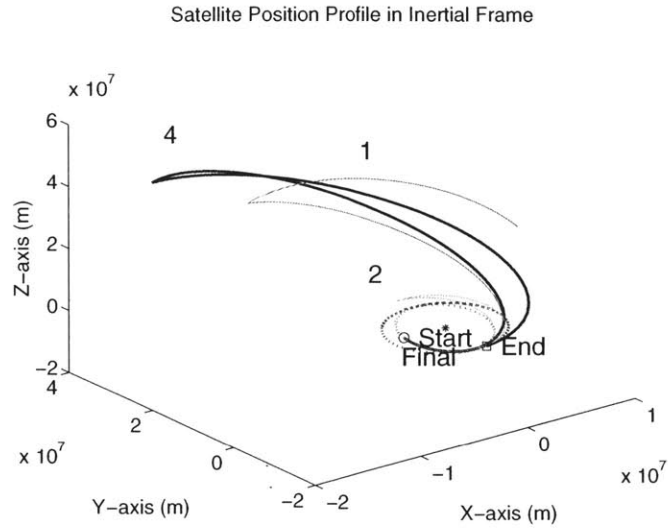
Figure 6-14: LEO-LEO Transfer: Parameterized Guess and Two Solutions

Table 6.28: LEO-LEO Transfer: Parameterized Guess and Three Solutions

Knot	1		2		3		4	
	t (min)	Δv (m/s)	t (min)	Δv (m/s)	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	0	0	0	0	0	0	0
2	25.66	2446.83	34.67	736.32	26.86	2545.10	26.78	2574.92
3	302.36	1997.15	80.15	6839.71	448.28	1548.30	485.56	1475.05
4	595.10	2278.17	151.37	1148.19	844.89	2469.23	906.58	2499.59

91

6.2.2 From LEO (ISS) to Molniya Orbit

Now, a satellite must transfer from the ISS orbit to a Molniya orbit, which varies significantly from the initial LEO in size, shape and orientation. The final elements in Table 6.29 were collected from a Two Line Element Set for a Russian communications satellite dated June 13, 2002.

Table 6.29: LEO-Molniya Transfer Element Set

Orbital Elements	Initial	Final
a	6,772,290.24534 m	26,554,926.3973 m
e	0.0007083	0.7271244
i	51.6390°	62.9383°
Ω	58.5505°	203.4832°
ω	238.2837°	280.7294°
ν	261.4820°	(Not Specified)

The same solution technique is used on the transfer as on the LEO-LEO transfer. Namely, the problem will be solved with a continuous set of controls whose high thrust values approximate impulses. The solution is then transformed to a parameterized set of impulses via a quadrature routine. The parameterized trajectory serves as a first guess in the parameter setup to solve directly for impulses.

This transfer problem is an interesting one because of the extreme changes required in all of the elements. In studying this problem using DIDO with continuous controls, it is discovered that the problem has many solutions with extremely different characteristics (trajectory and cost, primarily). Therefore, one has to be extremely careful when providing a guess to the optimizer. It is important not to give a guess that overly restricts the solution space. In this particular problem, DIDO was given a number of different starting points, and the most reasonable of the resulting solutions from the continuous regime was transformed and used for later processing.

In Figure 6-15 a continuous result is shown, with a trajectory on the left and a thrust acceleration profile on the right. A series of small burns occur in the beginning which raise the orbital altitude. After about one revolution, an extremely large impulsive maneuver conducts most of the orientation change. Finally, the transfer is completed near perigee of the Molniya orbit.

There is no guarantee that the continuous solution shown here represents the optimal LEO-Molniya transfer. However, it is a reasonable-looking solution with little (apparent) superfluous maneuvering, so it serves to be transformed into a parameterized trajectory. Figure 6-16 shows the continuous trajectory laid against its transformed trajectory. Six burns are conducted in the continuous solution (although some are very small), and so the parameterized trajectory uses six knots, with Δv

contributions on each of them. The cost of the parameterized trajectory is nearly identical to the continuous trajectory (10,263.37 m/s).

Satellite Position Profile in Inertial Frame
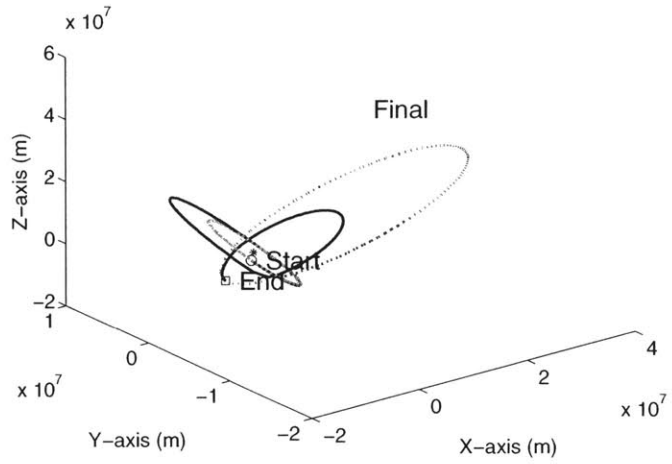
Thrust Acceleration Profile

Figure 6-15: LEO-Molniya Transfer: Continuous Solution

Satellite Position Profile in Inertial Frame

Figure 6-16: LEO-Molniya Transfer: Continuous Solution with Transformed Parameterized Guess)

It is clear from the figure that the parameterized trajectory fails to look much like the original for the second half of the transfer interval. This is an artifact of the quadrature routine on this particular continuous trajectory. The plane change maneuver, occurring around 10,000 seconds into the transfer, accumulates a large amount of Δv over a large amount of time. Quadrature is used to capture that accumulation into a single impulse, fo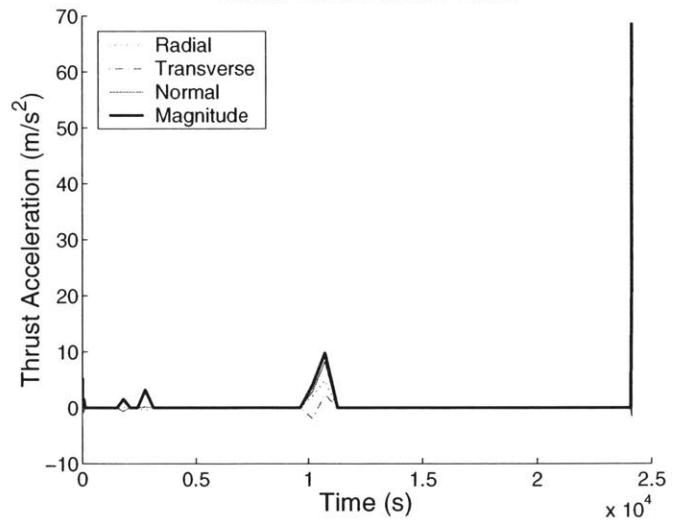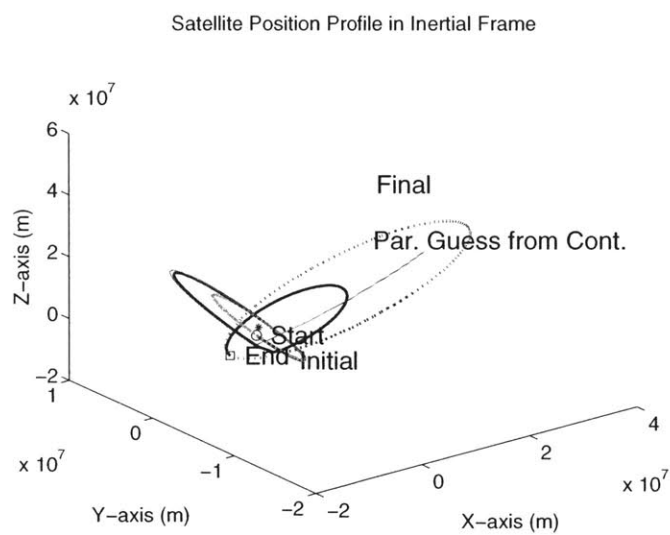rced to occur at the node of highest thrust. Because the continuous solution burns over a large amount of time, it is not unreasonable that the parameterized transform displays such variation. This variation is not considered an issue for a couple of reasons. Primarily, the parameterized transform is merely a guess; it is good enough to give the optimizer something to work with. In this particular case, as well, it seems to open up the solution space: while the original trajectory as a guess would probably continue to drive the solution to meet the final orbit at perigee, this transform opens up the possibility of finding other entry points into the final orbit.

Three trajectories are shown against the initial and final orbits in Figure 6-17, with their parameters summarized in Table 6.30.

1. Parameterized Guess from Continuous. This is the trajectory that results from running a quadrature routine on a continuous result to approximation an impulsive solution. (Cost: 10,263.37 m/s)

2. DIDO Solution and Improved Guess. Processing (1) yields this first impulsive solution. This result meets the constraints and quite significantly reduces the cost in doing so. Although six Δv maneuvers were in the guess, the solution has zeroed out three of them, leaving three maneuvers to follow a predictable transfer pattern (orbit raising, plane change, completion). The transfer requires the maximum amount of time, implying that the cost could be lowered by increasing the time of flight. (Cost: 6631.92 m/s)

3. DIDO Solution. Opening up the time bound and processing (2) yields this final solution. The bound on time is now twice as large as in (2), and while maintaining the same basic shape in trajectory, conducts a larger orbit raising maneuver in order to reduce the cost of the plane change. The solution again requires the maximum amount of time allowed. The overall cost is less, and probably could be improved more by raising the time bound even more. Naturally, there will be diminishing returns in cost for increasing the time, and it is up to the user to put a balance these as desired. (Cost: 5923.13 m/s)

The resulting solution for this transfer scenario is ultimately much different than what was first produced in the continuous domain. The cost has nearly been cut in half, and the trajectory has no resemblance to the original. To satisfy the curious, the final parameterized solution was used to contruct a guess in the continuous regime

to see if this new trajectory could be recreated there. The continuous result has a similar trajectory to its guess from the parameterized solution but costs over 7800 m/s. Further processing could, perhaps, reduce this maneuver cost.

Satellite Position Profile in Inertial Frame
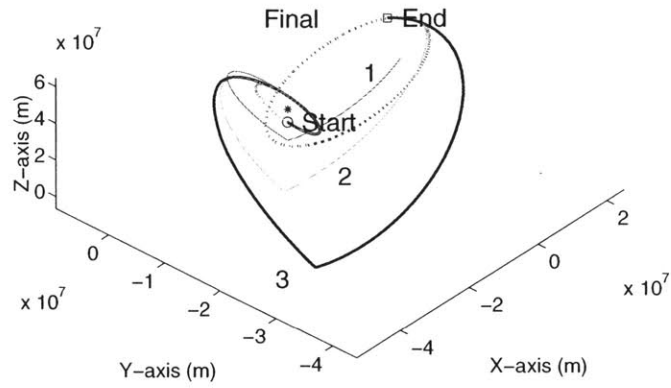
Figure 6-17: LEO-Molniya Transfer: Parameterized Guess and Two Solutions

Table 6.30: LEO-Molniya Transfer: Parameterized Guess and Two Solutions

	1		2		3	
Knot	t (min)	Δv (m/s)	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	33.64	0	0	0	0
2	1.02	98.07	0.23	0	0.11	0
3	30.06	494.79	49.57	2482.29	50.19	2740.00
4	45.94	1185.27	75.83	0	85.28	0
5	178.54	7729.57	292.50	3233.49	528.08	2025.64
6	402.48	722.03	844.89	916.14	1689.78	1157.50

6.3 Uncertainties in Solution Optimality

Previously, it has been demonstrated that a solution coming from DIDO may not be a globally optimal solution, as it has taken several iterations on the DIDO solution to actually find the global solution. Indeed, with any direct method of optimization, only a locally optimal solution can be guaranteed. One must be careful: sometimes a local minimum can cost much more than than the global minimum. In this section, some analysis is presented to understand the realm of feasible solutions, and where minimum solutions fit into that realm. The study was motivated when the optimizer failed to find the known global minimum for a particular transfer pair.

6.3.1 Motivation

In Section 6.1.1, successful results were presented for both a circular-circular Hohmann transfer and an elliptical-elliptical Hohmann transfer. Another transfer pair studied in this series, however, was an elliptical-circular transfer. Because of the nature of the solutions found with this pair, the results are presented here.

From Elliptical Orbit to Larger Circular Orbit

For the elliptical-circular Hohmann transfer, a satellite begins from a point on an elliptical orbit, and it is expected to coast to perigee before initiating a transfer. Then, 180 degrees later, it should terminate on the circular orbit. Using a mix of element sets from other scenarios, the elements for these two orbits are summarized in Table 6.31. The familiar requirements for this transfer can be seen in Table 6.32.

Table 6.31: Elliptical-Circular Hohmann Transfer Element Set

Orbital Elements	Initial	Final
a	7,300,000 m	42,160,000 m
e	0.1	0.0
i	0.0°	0.0°
$\tilde{\omega}$	0.0°	(Not Specified)
ν	270.0°	

Several different guesses were used as starting points for DIDO for the elliptical-circular transfer. In each case, DIDO converged on a feasible solution, but most often, it was not the solution outlined in Table 6.32. (The optimal solution was only obtained when the guess, itself, was near-optimal.) The trajectory characteristics were similar when DIDO converged on a non-optimal solution.

One such result is presented in Figure 6-18 with the corresponding data in Table 6.33. At the initial epoch, the satellite was at a true anomaly of 270 degrees. Instead

Table 6.32: Elliptical-Circular Hohmann Transfer Requirements

Δv_1	2076.72 m/s
Δv_2	1478.13 m/s
$t_f - t_1$	315.41 min
$t_1 - t_0$	22.58 min

of coasting for 90 degrees and burning at perigee, the satellite burned at some time before perigee. In this particular case, it burned at $\nu = 349.11°$. It placed the satellite on a transfer ellipse which reached the final orbit after a transfer angle, θ, of 175.55 degrees (instead of $\theta = 180°$ for a true Hohmann transfer). The figure shows both the DIDO transfer solution and the correct solution.

The most logical next step when faced with such a result is to use the solution presented here as an *a priori* guess for another DIDO run. However, additional runs did not improve the results. With frustration, it was determined that the optimizer had found a locally (but not globally) optimal solution. Once at this local minimum, DIDO could not move away from it in order to find the global minimum.

To test this hypothesis, analysis was conducted to understand the solution space, to find where the global minimum exists in the context of the other feasible solutions to this problem. Do characterstics exist in the solution space that make it difficult for an optimizer to find the global minimum?
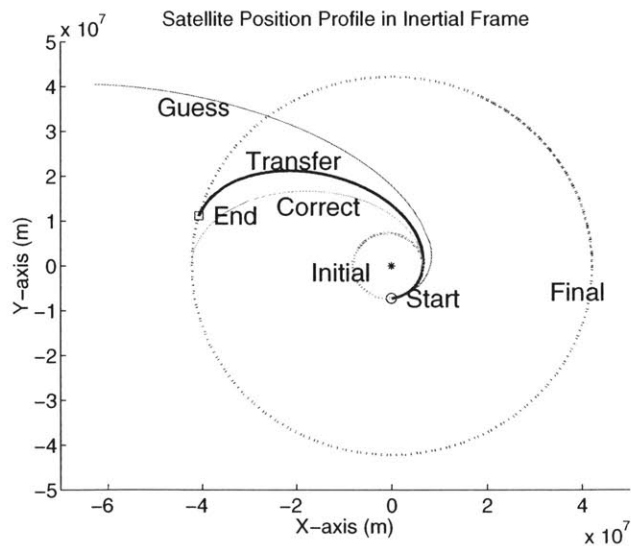
Figure 6-18: Elliptical-Circular Hohmann Transfer: Random Guess

Table 6.33: Elliptical-Circular Hohmann Transfer: Random Guess

Knot	Guess		Solution	
	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	1258.18	0	0
2	22.58	2516.36	20.03	2102.71
3	422.45	1258.18	318.39	1478.02

6.3.2 Understanding the Solution Space

To search the solution space for a Hohmann transfer problem, Lambert's Theorem was used to find all two-burn solutions. Three quantities—departure point (the true anomaly on the initial orbit), transfer angle, and transfer time—were varied parametrically to note their effects on the solution cost (total Δv for the two burns).

Table 6.34: Cost Analysis Variable Ranges

Variable	Range		Resolution
	(Lower)	(Upper)	
Departure Point (ν)	$0°$	$360°$	$5°$
Transfer Angle (θ)	$45°$	$315°$	$5°$
Transfer Time (t)	$15,000\ s$	$20,000\ s$	$5\ s$

Because only two of the three variables at a time can be presented graphically against the cost, figures in the next sections are presented in sets of surface plots. For a given transfer, three different sets were analyzed:

1. Departure Point, Transfer Angle vs. Cost *given best transfer time for each Departure Point/Angle pair*

2. Departure Point, Transfer Time vs. Cost *given best transfer angle for each Departure Point,Time pair*

3. Transfer Angle, Transfer Time vs. Cost *given best departure point for each Angle/Time pair*

The "best" transfer time for a Departure Point/Angle pair is the transfer time that minimizes the cost for those particular values of ν and θ. This pattern follows for the other pairs, as well. Occasionally, it is useful to show the variation in that "best" variable, so some plots are also presented in that regard. For example, plotting ν and θ versus the best t for transfer.
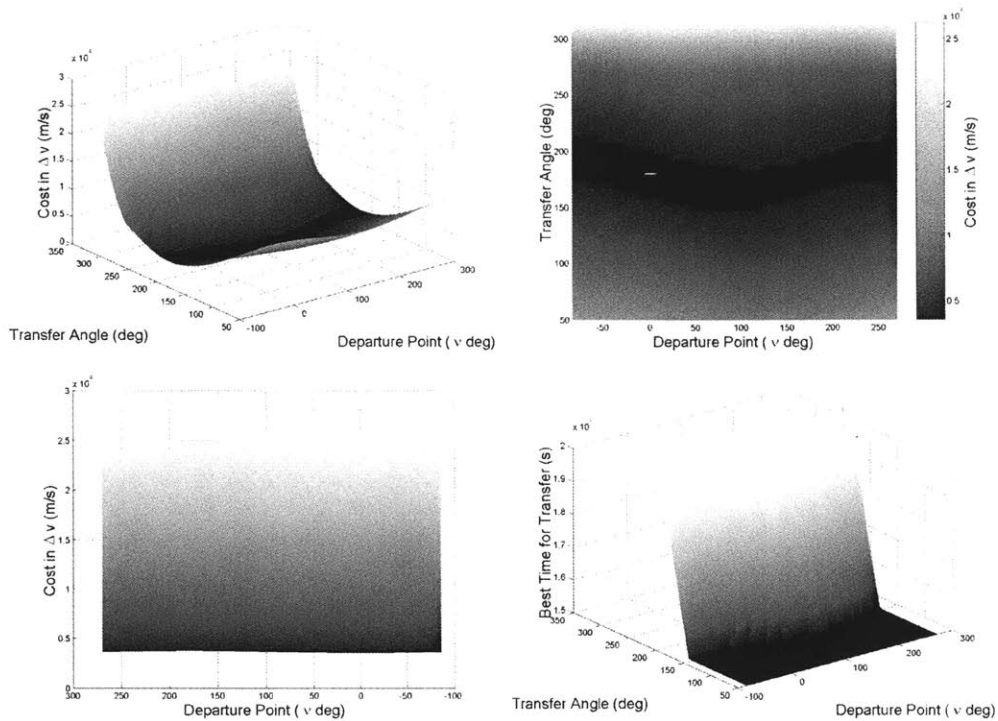
Figure 6-19: Dep. Point, Angle vs. Cost Given Best Time (Elliptical-Circular)

From Elliptical Orbit to Larger Circular Orbit

Presented in Figure 6-19, we look at departure point and transfer angle against cost for the elliptical-circular transfer. The first three plots are different views of the same surface; the plot in the bottom-right corner puts the variable against the best time for transfer.

Apparent in the first three plots is the fact that the cost is fairly insensitive to changes in the departure point. The trough would indicate that the transfer angle is a much bigger factor. However, in the final plot, the best time for transfer is at either lower or upper bound for many of the departure point/angle pairs. Therefore, information on cost is much more reliable in the region of where the best transfer time is not at a bound.

103
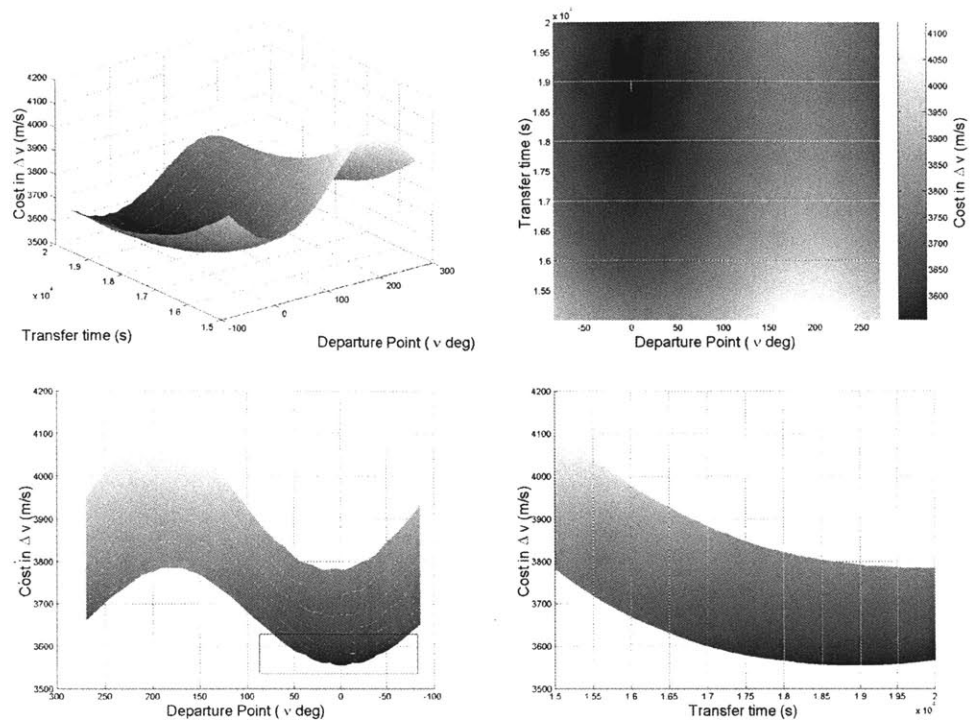
Figure 6-20: Dep. Point, Time vs. Cost Given Best Angle (Elliptical-Circular)

In Figure 6-20, all four surface plots show the departure point and transfer time against the cost given the best transfer angle for each ν/t pair. Not shown but of note is the fact that the best transfer angle ranges from 155° to 195°.

In these plots, the minimum seems to be more defined that in the previous set, but the variation in cost is much smaller. Therefore, transfer angle contributes more to the size of the cost than either of the other variables.

Very important to notice are the "bumps" that appear near the minimum cost in the bottom-left plot. This is not an artifact of the granularity of the analysis: this shows clearly that additional stationary points exist at departure points other than perigee. This helps to explain the DIDO results presented on this transfer.
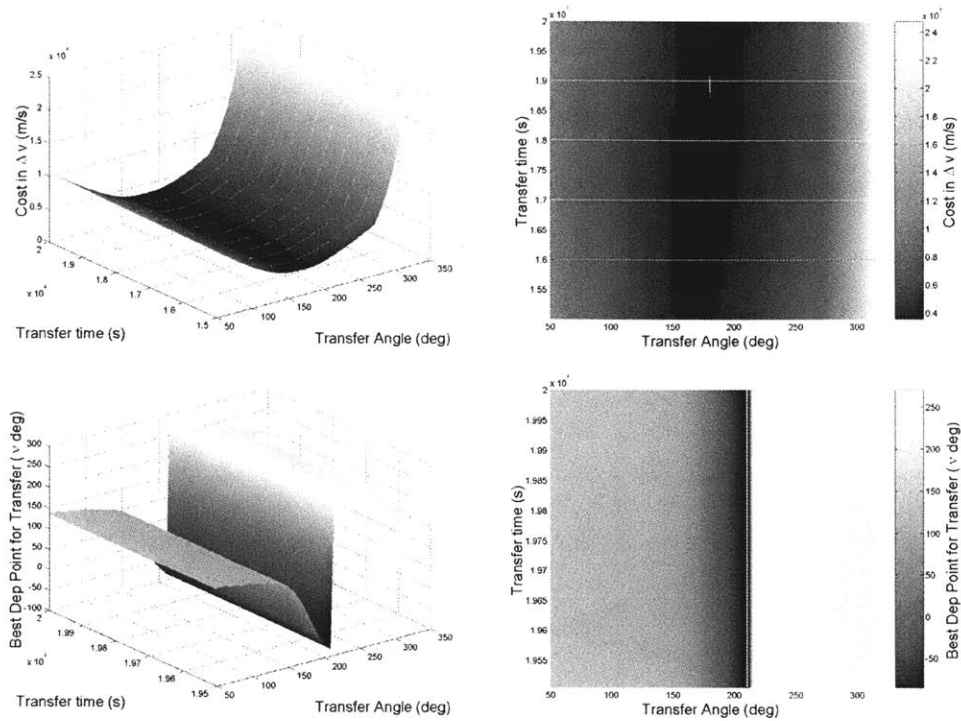
Figure 6-21: Angle, Time vs. Cost Given Best Dep. Point (Elliptical-Circular)

Transfer angle and transfer time are plotted against cost, given the best departure point for a pair, in the top two surface plots of Figure 6-21. In the bottom two, the best departure point for transfer is plotted against changes in angle and time.

The bottom plots indicate that for a given θ/t pair, the best departure point is not necessarily at perigee ($\nu = 0°$), although it is when $\theta = 180°$. The transfer time has limited effect on the cost or the departure point as compared to the transfer angle.
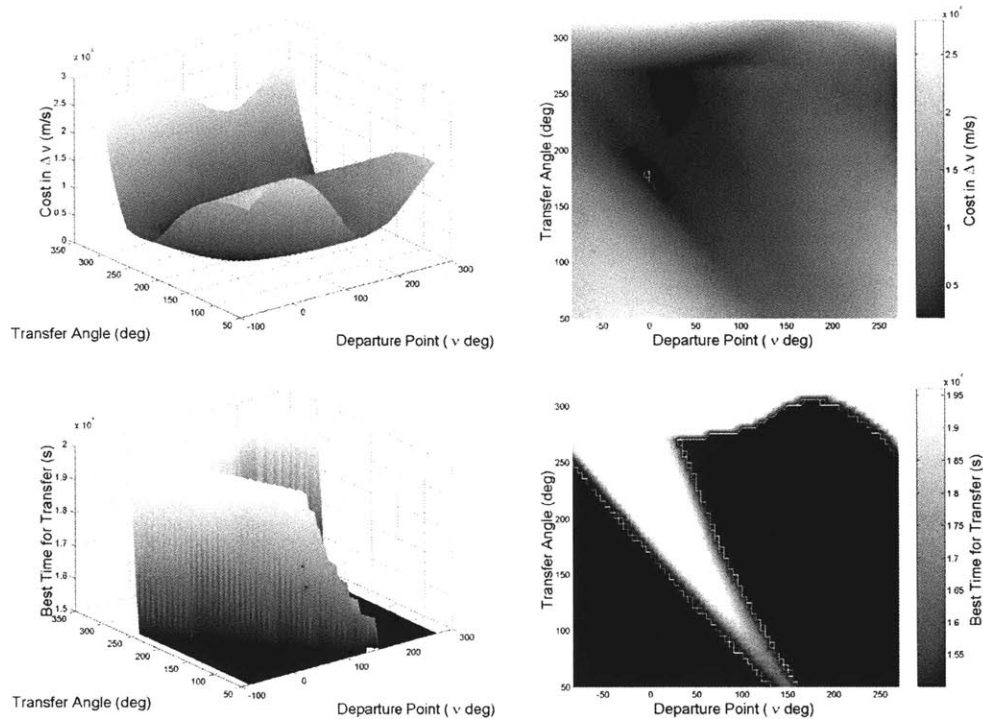
Figure 6-22: Dep. Point, Angle vs. Cost Given Best Time (Elliptical-Elliptical)

From Elliptical Orbit to Larger Elliptical Orbit

With the elliptical-circular transfer, two major characteristics can be seen: there exist alternative stationary points that do not represent the absolute minimum, and the global minimum sits along a fairly flat curve in some variables. This clearly provides some explanation into the DIDO results that came out of this scenario. However, it was important to conduct this analysis with another problem to provide a reference.

If the final orbit was elliptical as well, oriented along the same line of apsides as the initial orbit, it is reasonable to suspect that the global minimum will be better defined. With the Elliptical-Elliptical transfer shown in Section 6.1.1, the optimizer was able to find an optimal solution: does this result make sense in the context of the solution space?

In Figure 6-22, departure point and transfer angle are plotted against cost in the top two graphics; the bottom two show the same variables against the best time for transfer. As one would expect, the first two show a more definable minimum than in Figure 6-19. One must again be aware that for some ν/θ pairs, the transfer time is at a bound.
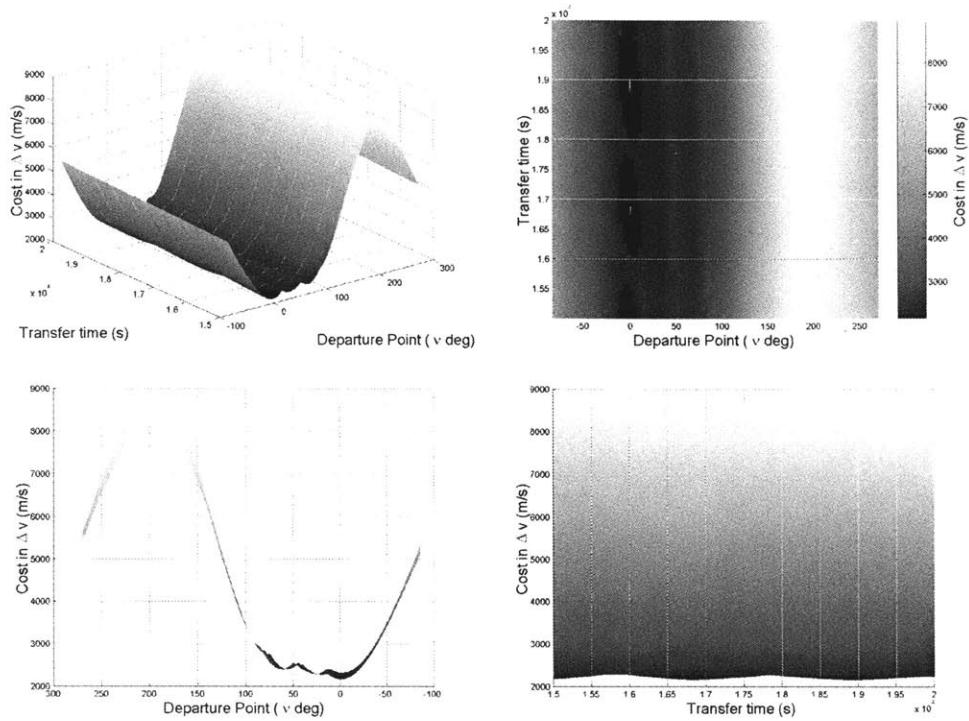
Figure 6-23: Dep. Point, Time vs. Cost Given Best Angle (Elliptical-Elliptical)

Figure 6-23 shows departure point and transfer time against cost, given the best transfer angle. Each graphic represents the same information from a different perspective.

It is apparent from the figure that making the final orbit elliptical dramatically increases the potential variation in cost due to the departure point (compare these plots with Figure 6-20). This makes sense: now it is much more important for the transfer to be completed at the apogee of the final orbit. When the best angle for transfer is generally around 180 degrees, the departure point must sink into perigee.

The "bumps" of the elliptical-circular case are even more pronounced now, as seen especially in the plot in the lower-left corner.
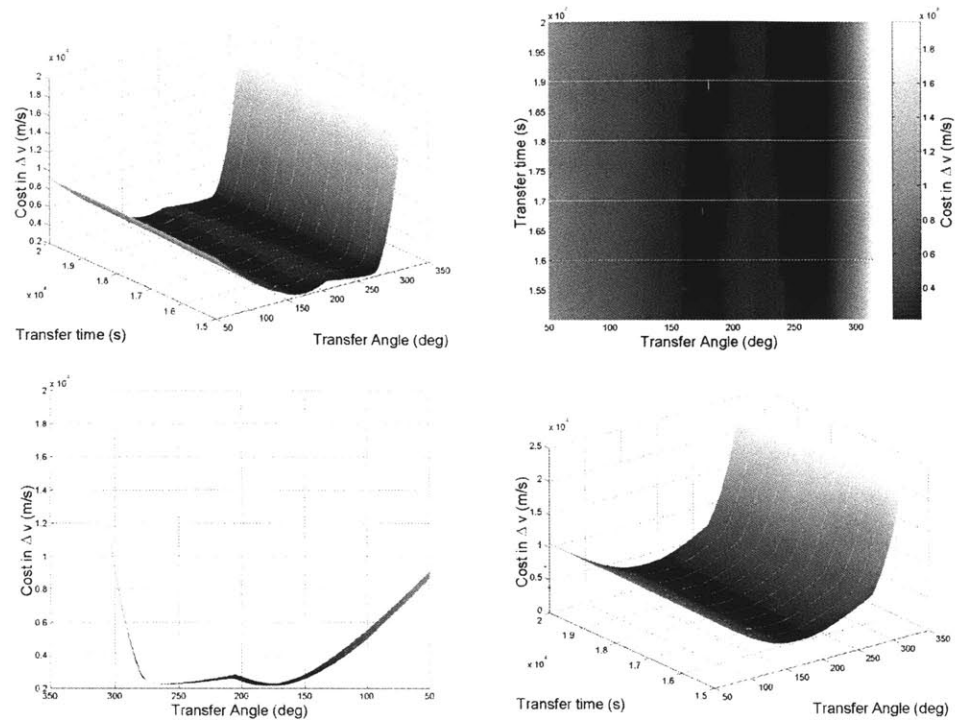
Figure 6-24: Angle, Time vs. Cost Given Best Dep. Point (Elliptical-Elliptical)

The first three surface plots of Figure 6-24 show the transfer angle and transfer time against cost for the elliptical-elliptical transfer. The plot in the bottom-right corner is a repeat of the same surface plot from the elliptical-circular case for comparison.

An additional valley has appeared, comparing the top-left (Elliptical-Elliptical) with the bottom-right (Elliptical-Circular). This valley is located at $\theta \approx 270°$ and $\nu \approx 20°$.

6.4 Providing a Good Guess for DIDO

The *a priori* guess provided to DIDO can have a serious impact on the solution. The results presented in this chapter should, at the least, allude to this conclusion. Ways to improve the guess and to avoid problems with the solution (either nonoptimal or infeasible solutions) have only been subtly mentioned so far. This section is devoted to summarizing the conclusions that came from the examples presented above, as well as others not covered yet.

6.4.1 Guess Consistency

Having consistency within the guess is probably one of the most important qualities in a good guess. This is an obvious conclusion, but certainly worth addressing. Simply put, the elements of a guess—the states, the controls, the times, the parameters— must be consistent. Specifically, if the guessed parameters impose a Δv maneuver at some time, then the guessed states should reflect the same maneuver. In other words, the guess should meet the dynamical constraints of the problem.

While it is certainly convenient to allow some latitude in the guess, trusting in the optimizer's ability to "straighten out" the trajectory, the likelihood of convergence on a locally optimal solution is greatly reduced when the elements of the guess do not line up. It has been shown before that with an infeasible guess, the optimizer seems to focus primarily on achieving feasibility before optimality. In the same way, if an optimzer must focus on meeting dynamical constraints (that could already be met by the user), then its efforts cannot be set on achieving feasibility or optimality.

All of the data presented in this chapter have resulted from consistent guesses.

6.4.2 Engineering Judgment

The most important conclusion that comes from the material presented in this chapter is that a level of engineering judgment is required in order to find optimal solutions. If the solution does not make intuitive sense, it very well may not be optimal. The first question to ask should always be: is the solution sensible?

In general, it is good practice to resubmit a solution as a guess to improve a solution. If it does improve, there is an indication of how near or far one is away from optimality; if it doesn't improve, then there can be more comfort in the quality of the solution as it is.

As well, engineering intuition should be applied in developing an initial guess for the optimizer. The "shape" of the guess is important.

The Shape of the Guess

The shape of the guess trajectory also appears to have some sort of impact on the nature of the solution that results. In general, the resulting trajectory from a DIDO
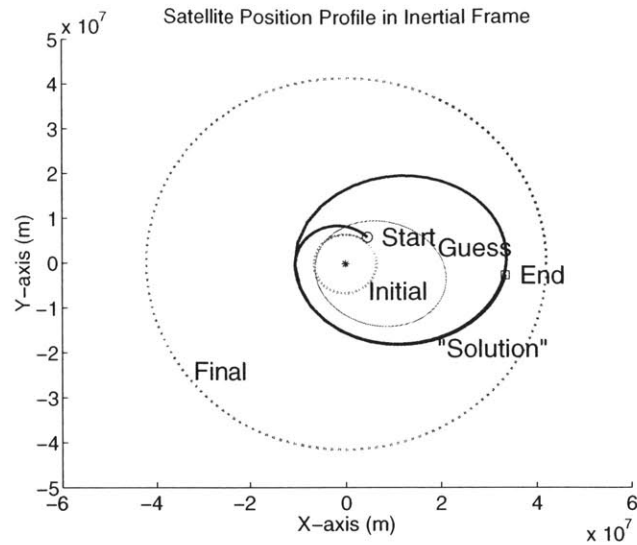
Figure 6-25: Combined Plane Change Transfer: Bad Guess

run will likely have a similar shape to that of the guess. As an example, Figure 6-25 shows a guess/"solution" set from the combined plane change transfer (circular). In the guess, the time of flight is probably a reasonable value, but because of the nature of the guessed parameters, the vehicle does not reach the terminal altitude. Instead, it stays in a relatively small orbit and in the allotted time conducts several revolutions, ultimately not reaching the target orbit.

Consequently, the "solution" that results from DIDO, which is clearly an infeasible solution, highly resembles the guess in shape. The trajectory takes place over several orbital periods, and although the optimizer does raise the orbit to some extent, it never gets to the final orbit. Even if it did, however, it probably would not have taken out the extra revolutions.

We can support this last conjecture with an equivalent case from the other combined plane change (elliptical). In Figure 6-26, a coasting guess is used on the elliptical transfer. This problem differs in the fact that that the perigee of the final orbit is in reasonable proximity to the initial orbit (and the guess trajectory). A feasible trajectory results from this guess because of the nature of this transfer scenario, but it is clear that the trajectory is not optimal. Rather, the trajectory resembles the previous trajectory in shape.

This particular characteristic makes sense in the context of the problem setup. The dynamics for this problem are described in Cartesian coordinates, and there is a significant change in state values when the vehicle flies for a full revolution. It is difficult, then, for the optimizer to "undo" or add revolutions to find an optimal solution. One way to get around this problem would be to describe the dynamics in

110
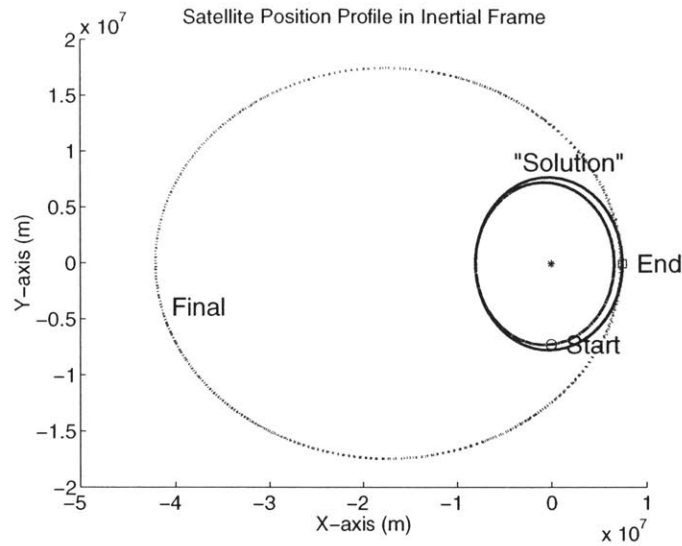
Figure 6-26: Combined Plane Change Transfer (Elliptical): Bad Guess

orbital elements instead. However, as presented in Chapter 5, this creates additional difficulties for the parameter optimization problem. Instead, the best way to avoid this problem is to provide a guess that has as many (or few) orbital revolutions as the user thinks are necessary for the optimal solution. A user should look at the shape of his guessed trajectory before processing it.

A good guess is important not only to increase the chances that the optimizer will find a solution, but also to increase the chances that it finds the correct solution. The first way to improve a guess is to ensure that the guess is consistent: the states and controls of the guess should line up with the dynamics of the problem. Engineering judgment is also important: a solution that does not look reasonable is probably not optimal. As well, the user must be aware of the fact that a DIDO solution may not vary greatly from the guess in its general trends or characteristics. The shape of the guess must be considered.

In this chapter, the impulsive problem was presented as it is implemented in DIDO. The capability was presented for basic problems, like the Hohmann transfer, but it was also applied to several practical problems. Next, DIDO will be used to solve finite-burn problems.

111

[Except for this sentence, this page intentionally left blank.]

Chapter 7

Finite Burn Orbital Transfer

In the previous chapter, the Legendre Pseudospectral Method was applied to the impulsive orbital transfer problem. Orbital transfers were solved by performing one or more impulsive maneuvers on a vehicle to move it from its starting position and velocity to a target orbit. It is recognized, however, that truly impulsive maneuvers are not possible; a Δv must take place over a finite amount of time. When the thrust limit is high, that maneuver resembles an impulse, and so it makes sense to treat it as one.

This chapter focuses on finite-burn transfers, where the maneuver cannot reasonably be approximated by an impulse. Specifically, an upper bound is placed on the thrust in such a way as to force a maneuver (like those performed with impulsive transfer) to take place over a substantial amount of time. The parameterized controls used to solve for Δvs will not be used when solving for finite burns. Instead, the standard, continuous controls will be applied; the controls will take values at each node, just as the states do, and the controls will represent the magnitude and direction of a thrust acceleration. It is for this form of problem that DIDO was originally designed, and so the capability and limitations are demonstrated here using familiar orbital transfer problems. The method is then applied to a real-world problem, applying the available features to make the solution as realistic as possible.

A discussion of solution feasibility is also included in this chapter. When the optimizer converges on a solution, is that trajectory truly feasible? Equivalently, this is a question of the accuracy of the solution, or more specifically, of how well the discrete state and control values at the nodes represent a continuous solution (i.e. from a shooting method). It is important that a user be able to trust the solution that comes out of the black box (DIDO), so the issue is covered in detail.

7.1 Capability Demonstration

A series of transfers were explored when the Pseudospectral technique was applied to impulsive maneuvers. Hohmann transfers, plane changes, and combined plane

changes were studied, as these are the types of impulsive problems that could be solved by hand for Δvs.

The impulsive problem has a major advantage over the finite-burn problem: there is little question as to what the solution should look like for the simple orbit-raising or plane-change problems using impulsive maneuvers. In contrast, it is not necessarily known what a finite-burn solution should look like, even for these basic, single-element transfers.

When studying finite-burn problems, one must understand what the solution should look like *as compared to an impulsive solution*. How is the trajectory affected by imposing thrust limitations on the vehicle? Once the intuition to answer this question is developed, then practical problems can be solved. With this in mind, it would be an exercise in tedium to demonstrate the finite-burn capability on the entire series of transfer problems discussed in Chapter 6. Instead, two from the series are studied, the Hohmann (now Orbit-Raising) Transfer and the Inclination Change, and it is assumed that the conclusions drawn here would apply to the other problems from the original series, as well.

In Chapter 5, there is discussion of the dynamics setup; the problem dynamics can be defined in terms of Cartesian position and velocity or in terms of orbital elements (in this thesis, the modified equinoctial elements). While only Cartesian dynamics were used for impulsive transfers because of the nature of the event contraints, there are no such event limitations for the finite-burn problem, allowing for setup in either of the two dynamical forms. The strengths and weaknesses of these design choices are also discussed here as the results of the two setups are compared side to side.

7.1.1 Orbit-Raising Transfer

Recall the transfer from a low-Earth circular orbit to a geosynchronous circular orbit, first solved with two tangential impulsive maneuvers. The element set is summarized in Table 7.1. The cost of the Hohmann transfer was 3935.03 m/s, and transfer occurred over a 315-minute interval.

Table 7.1: Circular-Circular Orbit-Raising Transfer Element Set

Orbital Elements	Initial	Final
a	6,570,000 m	42,160,000 m
e	0.0	0.0
i	0.0°	0.0°
L	0.0°	(Not Specified)

In this section, the transfer is re-examined using finite burns. A significant bound is now placed on the thrust acceleration. Picked somewhat arbitrarily, the maximum

114

acceleration is set at 0.6 m/s^2, as if a 150-N thruster is operating on a 250-kg satellite. With the Hohmann transfer, the first Δv was 2456.90 m/s. With the acceleration bounded as it is, a maneuver of that magnitude would now take around 4095 seconds— over an hour—to accomplish. This bound is certainly enough to induce noticeable changes in the solution.

Before demonstrating the method on the finite-burn problem, a word is in order regarding the expected nature of the solution, as compared to the Hohmann transfer. The essential characteristic of the Hohmann transfer is the fact that the impulses occur tangential to the velocity vector of the satellite, at the transfer orbit's perigee and apogee. However, to find a finite-burn solution similar in form, the thruster would hypothetically be turned on for 4095 seconds, and then again for another long period of time near the apogee of the resulting orbit. For this problem, the first burn would occur over more than half of an orbital revolution! For these long duration burns, the elegance of the Hohmann transfer is lost. With this, it is probably not optimal to conduct such a long duration burn.

Instead, it is more advantageous to make each finite burn as much like the Hohmann transfer's impulses as possible by limiting the duration of the burns and spacing them out over mutiple revolutions. An initial burn places the satellite in a slightly elliptical orbit; when the satellite again reaches the position of the original burn (perigee), it burns again. Over many revolutions, then the orbit is raised to that of the Hohmann transfer ellipse. Once it is reached, a series of maneuvers at apogee re-circularize the orbit at the new, higher altitude. Assuming that time is not an issue, a finite-burn solution can approach the cost of the Hohmann transfer. When time is bounded, however, near-perigee and near-apogee burns must take on longer durations.

Dynamics in Cartesian coordinates and modified equinoctial elements are tested for their ability to find a solution similar to that described above. The numbers of knots and nodes necessary to produce a clean solution are just as important as the cost of the solution in accumulated acceleration (or Δv).

Cartesian Coordinates

The orbit-raising problem was used extensively for testing of the finite-burn DIDO setup. Two of the results of that testing are presented here as a sample of the quality of solutions found using Cartesian coordinates.

In describing the nature of a finite-burn solution of the orbit-raising problem, it is implied that the solution should take place over multiple orbital revolutions, as the Δvs that must be imposed at the transfer orbit's perigee and apogee should be split over several burns. Therefore, a reasonable solution is going to have a rather large time of flight, hopefully somewhere near the maximum allowed time, and the six states, representing position and velocity, must go through several periods of oscillation. Because the states do oscillate rather quickly from negative to positive
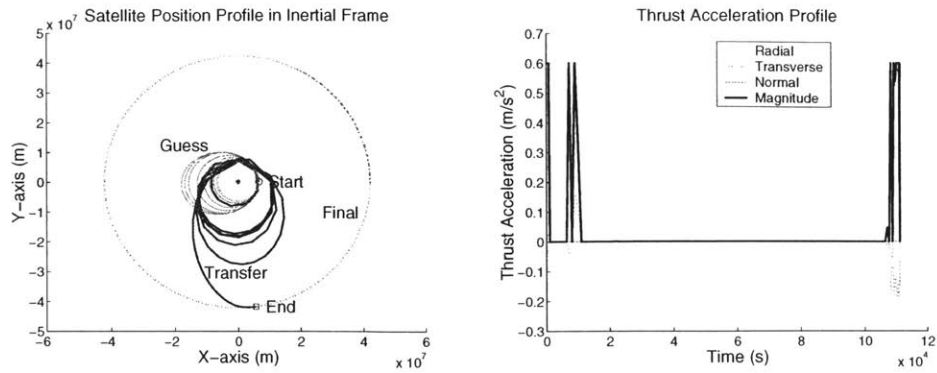
Figure 7-1: Orbit-Raising Transfer (Cartesian Coordinates): Solution # 1 (100 nodes)

and back, the quality of the guess has a dramatic impact on the solution. Specifically, a DIDO solution generated through Cartesian coordinates will generally have an equal number of orbital revolutions as the guess.

In the first sample result, the *a priori* guess is a trajectory that has small perigee burns over 7 revolutions. The transfer solution shown in Figure 7-1, also takes place over approximately 7 revolutions, using 100 nodes; both the trajectory and the acceleration profile are shown.

A number of details from this example should be noticed. It is easy to see that the position profile is not very smooth, indicating that the node count is probably not high enough to accurately represent the converged solution found by DIDO. This conclusion is supported by looking at the acceleration profile. While the controls clearly hit the maximum bound (at $0.6 \ m/s^2$), the controls do not have a clean appearance. A "clean" set of controls would have a well defined rectangular shape: the transition from zero thrust to maximum thrust would be practically instantaneous, and the duration of a burn would be clear.

Another detail of interest is in the cost, the accumulated Δv. By integrating the control magnitude, the cost of this solution is supposedly 3550.74 m/s, almost 400 m/s *less* than the Hohmann transfer solution. This is clearly not correct; it is not possible for a finite-burn solution to improve upon the performance of its corresponding impulsive solution. While the constraints are met at the nodes, they are most likely not met in between them; the nodes are either too few or too poorly located to accurately represent a feasible solution.

The logical next step is to add additional nodes to the problem, using the converged solution above as an initial guess. However, the first converged solution also provides some insight into where it would be useful to have more nodes: the times of control activity would benefit from additional nodes. In the next sample, whose trajectory and controls are shown in Figure 7-2, five interior soft knots and 180 nodes were used.
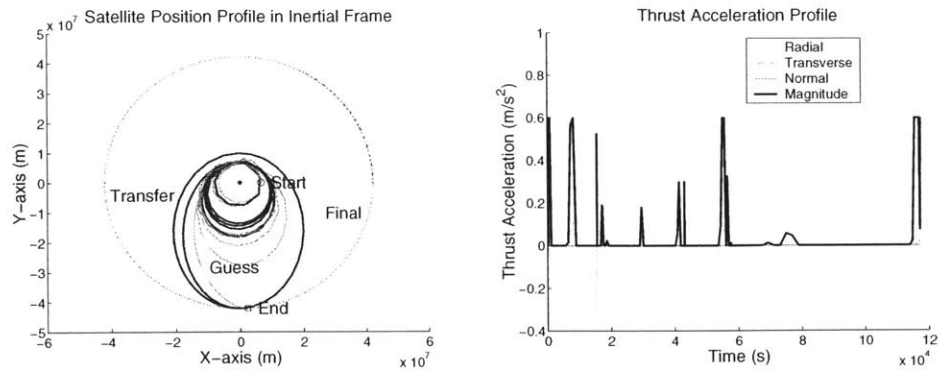
116

Figure 7-2: Orbit-Raising Transfer (Cartesian Coordinates): Solution # 2 (180 nodes)

Now, it is easy to see that the solution has smoothed out significantly, while still maintaining the general shape of its guess. Consistent with the expected solution, there are several perigee burns (about six) to raise the apogee of the transfer ellipse. Once the apogee height reaches the radius of the final orbit, two apogee burns re-circularize the orbit. The cost of this solution is 4085.25 m/s in accumulated Δv, which seems quite reasonable.

However, although the trajectory has smoothed out significantly, the controls are still not very clean. Again, the spirit of the solution includes six perigee burns and two apogee burns. The first six burning regions occur before 60,000 seconds, but they are not necessarily clean. Some burn regions have several spikes, a burn-off-burn control in a short amount of time, which is not accurate. As well, not all of the burns reach the maximum acceleration limit. The first apogee burn, occurring at about 70,000 seconds, is a good example of this. An optimal burn would be thinner (placing more of the burn near the apsis), and it would reach the bound.

To highlight this problem some more, Figure 7-3 places markers at each node in the trajectory when the control magnitude is not zero. Ideally these markers should be very near the apses. They are not.

Even more knots and nodes could be added to the problem, using this second solution as a guess. The result could certainly be cleaned up and smoothed out more, but a penalty is paid in the size of the problem as those knots and nodes are added.

Finally, it is important to notice that knots are more easily implemented when the guess comes from an already-converged solution (presumably from a DIDO run that did not have interior knots). This makes for additional difficulty for a problem of this size which requires a large number of nodes to capture each of the revolutions, as it should first be solved without interior knots. For example, the first example used 100 nodes, requiring a 100×100 dense differentiation matrix. A great deal of computational space is necessary only to come up with an initial estimate to the solution.
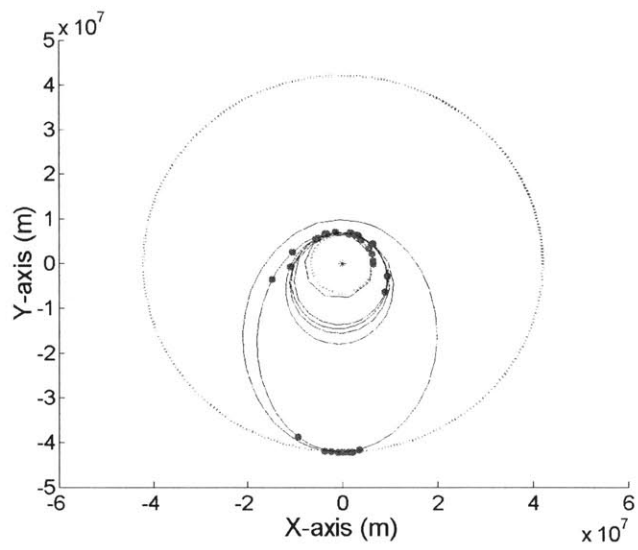
117

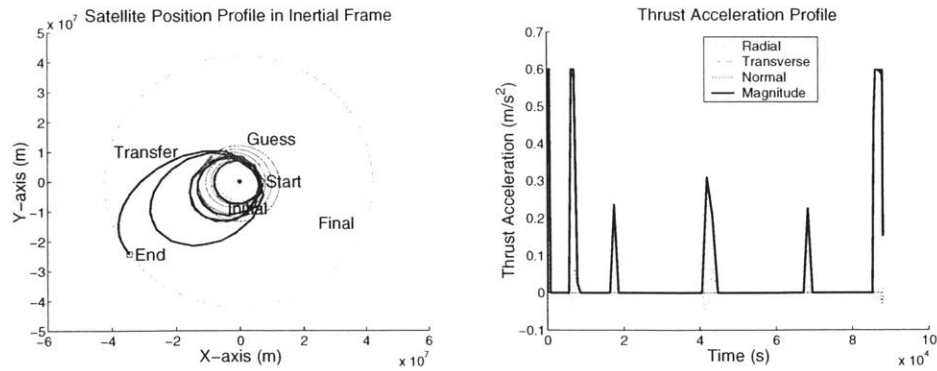Figure 7-3: Orbit-Raising Transfer (Cartesian Coordinates): Burning Nodes for Solution # 2

Figure 7-4: Orbit-Raising Transfer (Modified Equinoctial Elements): DIDO Solution # 1 (100 nodes)

Modified Equinoctial Elements

The dynamics of an orbital problem can also be represented in DIDO with orbital elements. Earlier, the modified equinoctial elements were defined, and it was shown that they do not break down with circular or equitorial orbits like the Keplerian elements do. As well, the modified elements smoothly handle both elliptical and parabolic orbits. Because five of the six elements are slow-varying, the modified equinoctial elements are more flexible than Cartesian coordinates in a direct optimization setting. This is now shown for the orbit-raising transfer. Using two similar examples—one without and one with interior soft knots—some of the advantages of the modified equinoctial elements become apparent.

The initial guess used in these sample equinoctial results is similar to that from the Cartesian example: a few low-Δv maneuvers are spaced over several revolutions, but the guess is no where near a feasible solution. In this particular example, the guess included four small perigee burns, followed by four apogee burns; the trajectory covered approximately six orbital revolutions.

First, the guess is submitted to find a solution without interior knots. One hundred nodes were used to find the first presented converged solution, shown in Figure 7-4. The position trajectory shown on the left is not remarkably better than the case without interior knots with Cartesian coordinates. However, there is a dramatic improvement in the "cleanness" of the control profile shown on the right. It is very clear where each of the burns occurs, and there is very little noise around the burns. Several of them do not hit the acceleration bound, no doubt due to a shortage of nodes in the vicinity of the burn, and this can easily be remedied.

Because of the smooth controls from the original case, adding strategically-placed knots is quite straightforward. Interior soft knots were added at most of the switching point of the control (where the control switched either on or off). Nine soft knots are

119
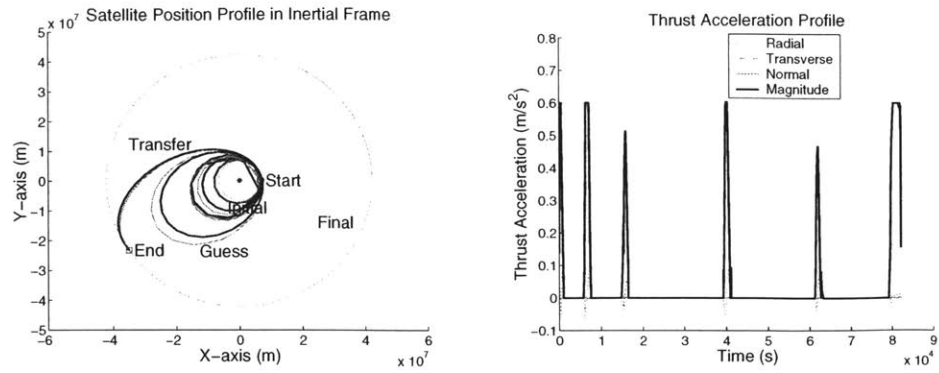
Figure 7-5: Orbit-Raising Transfer (Modified Equinoctial Elements): DIDO Solution # 2 (200 nodes)

included in the 200-node solution shown in Figure 7-5, where the previous converged trajectory served as a starting point for the optimizer. In this case, each of the finite burns that existed in the first profile has tightened into a smaller time interval; at most of them the burn stays at the maximum acceleration bound.

On the left side of Figure 7-5, the position profile is shown. It can be distinguished that during one particular revolution (the third), there are relatively few nodes distributed over the orbital period. This occurs during the coasting arc between 17,000 and 39,000 seconds. Again, it is understood that additional nodes in this particular region would take care of this discretization issue.

For the second Cartesian case, Figure 7-3 indicated on the trajectory where the nodes of thrusting occurred. Compare this result with Figure 7-6, which shows how the thrusting nodes for this second equinoctial case are much more focused at the apses of the transfer ellipses.

120

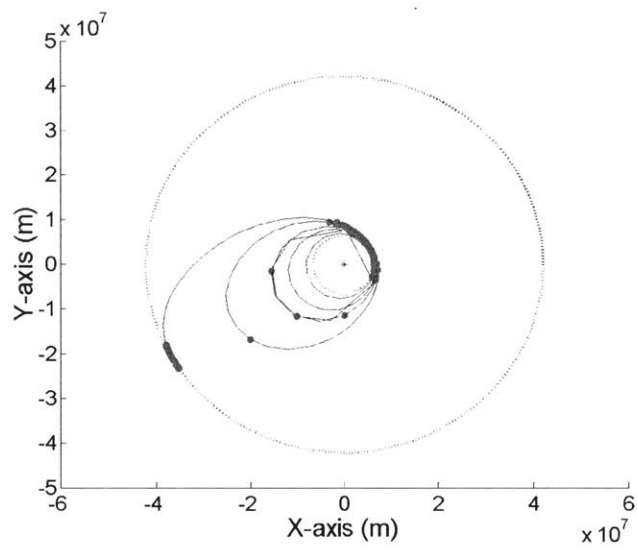Figure 7-6: Orbit-Raising Transfer (Modified Equinoctial Elements): Burning Nodes for Solution # 2

7.1.2 Inclination Change on a Circular Orbit

The simple plane change was another orbit transfer of interest when studying impulsive maneuvers. In this transfer, the size and the shape of the final orbit remains unchanged from that of the initial, but there is a change in either the inclination or the ascending node (or both). For the orbits chosen in Chapter 6, the initial and final orbits had two points of intersection, making it possible to conduct the transfer in a single Δv maneuver.

Now for the finite-burn problem, the inclination change problem for a circular orbit is resurrected. The orbital elements of the initial and final orbits can be seen again in Table 7.2. Recall also that the single-impulse solution costs 1357.73 m/s, found with

$$\Delta v = 2v_i \sin\left(\frac{\theta}{2}\right) \tag{7.1}$$

where v_i is the initial velocity (the velocity of the circular orbit), and θ is the required plane change of $10°$.

In the orbit-raising problem, the maneuvers can be divided into smaller Δvs without affecting the cost of the solution. An infinite number of very small Δvs in an orbit-raising problem would accumulate to the same cost, as long as they all occurred at the perigee and apogee of the Hohmann transfer ellipse, in the tangential direction. This is not the case for the plane change. For example, consider dividing the plane change maneuver into n impulsive burns, each occurring at a node and contributing an equal amount of plane change. In general,

$$2nv_i \sin\left(\frac{\theta}{2n}\right) \nleq 2v_i \sin\left(\frac{\theta}{2}\right) \tag{7.2}$$

so in theory it will not be beneficial to divide the plane change over many small-duration finite burns. However, a single finite burn, occurring over a substantial duration of time, would also raise the cost significantly by moving the ends of the burn further away from the orbital intersection (both before and after). Without knowing exactly what an optimal finite-burn solution would look like, it is understood that

Table 7.2: Inclination Change Transfer Element Set

Orbital Elements	Initial	Final
a	6,570,000 m	6,570,000 m
e	0.0	0.0
i	0.0°	10.0°
Ω	45.0°	0.0°
u		(Not Specified)

there will have to be some sort of compromise between the number of burns and the duration of the burns.

Again, the acceleration is bounded at 0.6 m/s^2. The problem is examined in both Cartesian coordinates and modified equinoctial elements.

Cartesian Coordinates

It has already been stated that because of the nature of Cartesian dynamics, the scope of DIDO solutions is generally limited to those equal in orbital revolutions to the guess. This can be seen as either an advantage or a disvantage. It is possible for a user to indirectly define the time range of a solution. On the other hand, it makes the task of discovering the balance between burn quantity and duration much more difficult.

Understanding the limitations of DIDO under Cartesian dynamics, three sets of results are briefly introduced. In each, the initial guess is a trajectory which implements virtually no thrusting (the guess remains along the intial orbit). The cases only differ in the number of orbital periods of the guess; solutions are presented with one, two, and three revolutions.

Figure 7-7 shows the trajectory of the single revolution solution with two control profiles. The first set of controls corresponds to a converged DIDO solution with 100 nodes. When 3 interior soft knots are added and the node count is increased to 160, the second set of controls results. Included above the second set of controls is a line of individual points, representing the times corresponding to each node. Knots are located where the nodes tend to bunch up, and it is not arbitrary that the knots occur near the control switches. Although the control profiles differ in an obvious way, the resulting position profiles are practically identical.

In the case where there are no interior knots, the controls are quite noisy; this is expected. When interior knots are added, the nature of the solution becomes quite clear. There are essentially two major burning segments, and the centroids of these segments occur very near the times of intersection between the initial and final orbits. (A satellite remaining on the initial orbit would intersect the final orbit at 1987.42 and 4637.32 seconds.) Two peculiarities exist in the clean solution, however. There is clear evidence of a small burn at the starting terminal. As well, the first major burn (at the descending node) has a definite split, where the control magnitude, starting at the maximum bound, dips down and comes back up. Interestingly, these same peculiarities will be seen with equinoctial dynamics, as well. The second set of controls still contains some noise, noticably at the final terminal where the control oscilates below the maximum bound. Further tightening the discretization would most likely reduce the noise characteristics.

123
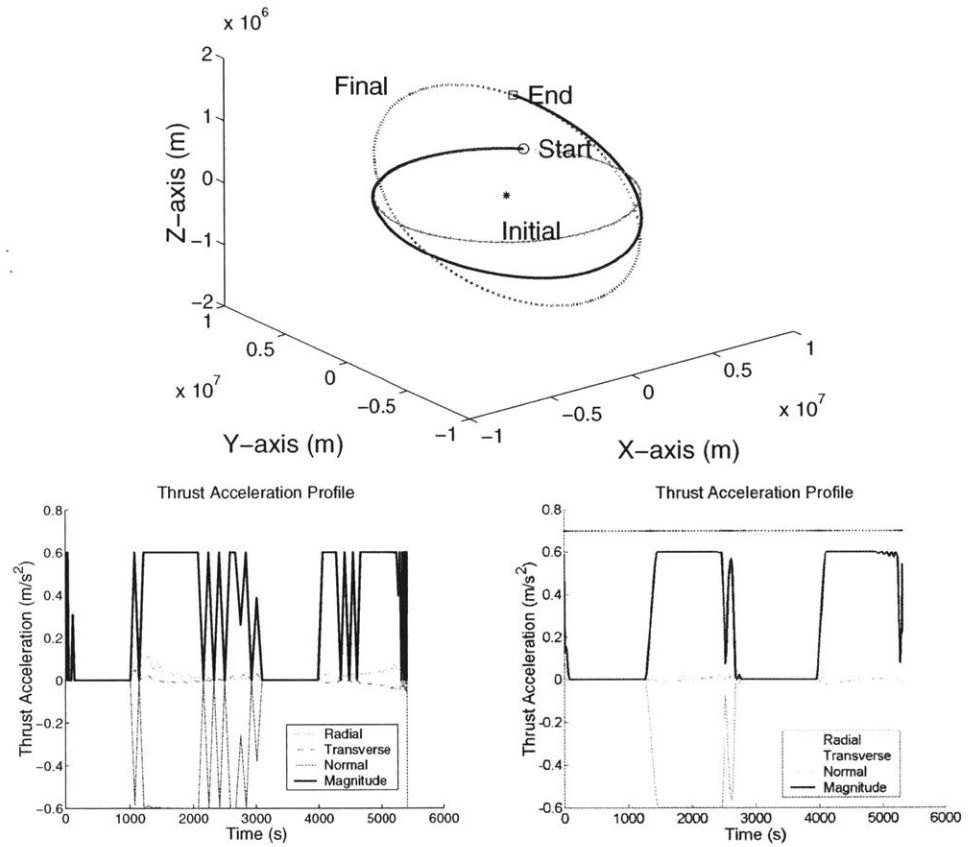
Figure 7-7: Inclination Change Transfer (Cartesian Coordinates): Single Revolution Solutions (100,160 nodes)

Two-revolution solutions are shown in Figure 7-8; 100 nodes are used in the first case, whose controls are on the left, and 200 nodes with seven interior soft knots are used in the second case, right. An extremely smooth control profile results with well placed knots and nodes in the two-revolution solution, placing two burns at the decending nodes and two at the ascending nodes of the final orbit.

Satellite Position Profile in Inertial Frame

Figure 7-8: Inclination Change Transfer (Cartesian Coordinates): Two-Revolution Solutions (100,200 nodes)

In Figure 7-9, three-revolution solutions are introduced. Again, the controls on the left, from a 100-node DIDO solution, are improved dramatically by increasing the number of nodes to 200, adding 9 interior soft knots to effectively capture the control switches. The solution presented on the right is nearly as clean as the interior-knot case with two revolutions (there is still a little bit of noise at the final terminal: a short coasting arc followed by a small duration and small magnitude non-zero control segment).
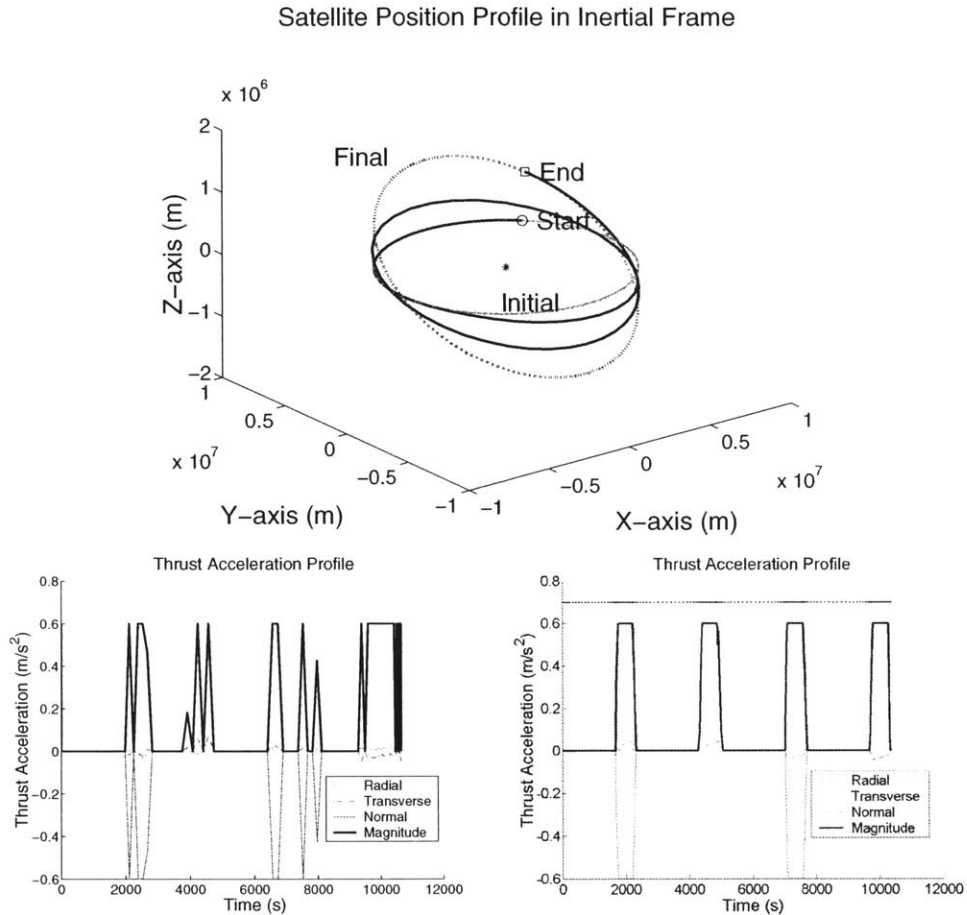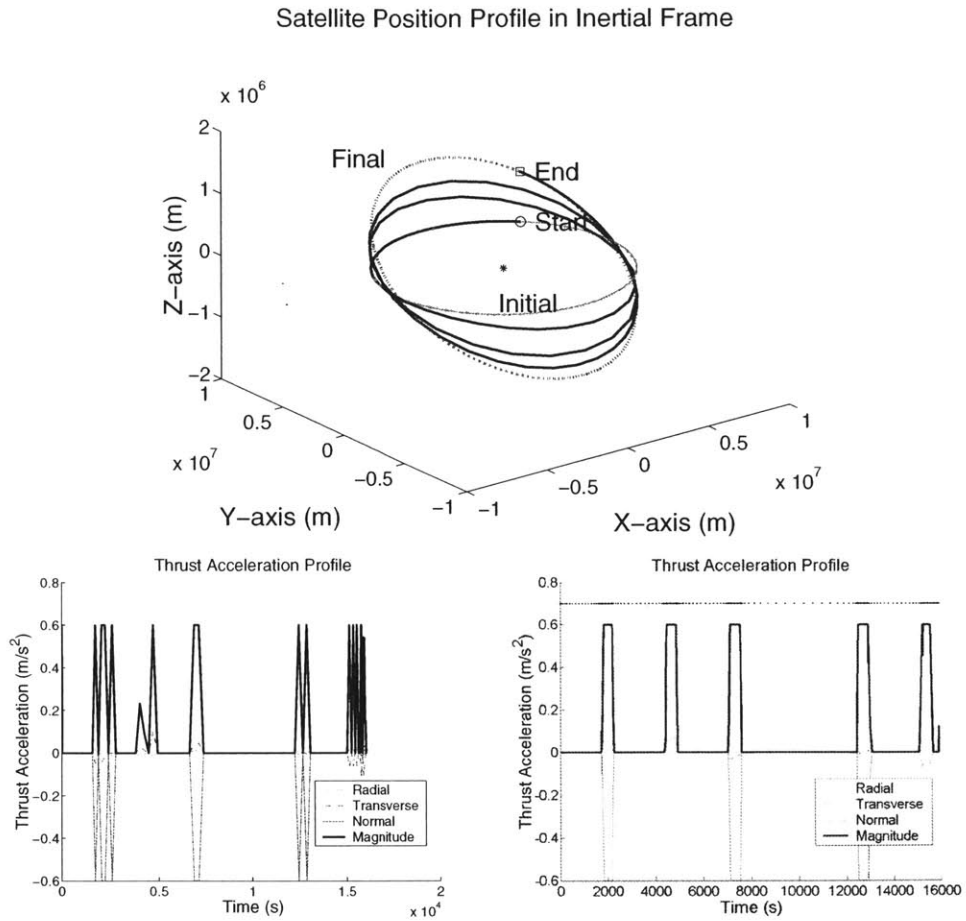
Satellite Position Profile in Inertial Frame

Figure 7-9: Inclination Change Transfer (Cartesian Coordinates): Three-Revolution Solutions (100,200 nodes)
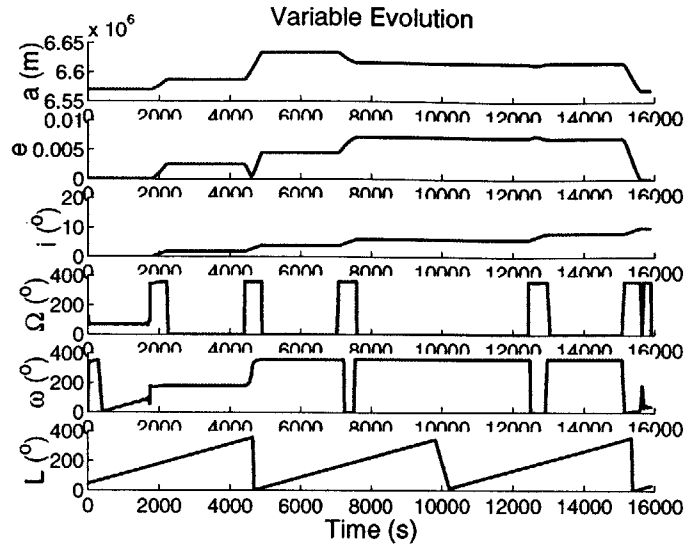
Figure 7-10: Inclination Change Transfer (Cartesian Coordinates): Variable Evolution of the Three-Revolution Solution (200 nodes)

There is one point of interest from the preceding result. The three-revolution solution contains five burn sequences, taking place around three descending nodes and two ascending nodes. For the three-revolution solution, one might initially think that there should be six burns, one at each ascending and descending node. However, the maneuver that might have occurred at the ascending node at 10,000 seconds is skipped.

In Figure 7-10, the five slow Keplerian variables and the true longitude are plotted against time. It is important to see that although there is no change in a and e between the initial and final orbits, a change definitely occurs during the transfer. The semi-major axis is raised during the first few burns, and a substantial eccentricity is introduced into the orbit. The last burns lower and circularize the orbit again. From the information provided in the figure, it is evident that at the ascending node at approximately 10,000 seconds, the satellite is at the perigee of the transfer orbit, when the orbital velocity is greatest. Conducting a plane change maneuver would be more expensive at perigee than anywhere else according to Equation 7.1. Clearly, the optimizer has recognized that since the optimal solution will require multiple burns, it is beneficial to raise the altitude of the satellite to make the plane change cheaper. It is also recognized that it is better to coast through that ascending node which occurs close to perigee in order to make use of the acquired eccentricity.

To conclude the Cartesian analysis on the plane change transfer, a cost comparison between the results presented above is necessary, and it is provided in Table 7.3. Recall that the single-impulse solution was 1357.73 m/s. There is a dramatic

127

Table 7.3: Inclination Change Transfer (Cartesian Coordinates): Cost in Accumulated Δv

Interior Knots?	1 Revolution	2 Revolutions	3 Revolutions
No	1626.35 m/s	1560.08 m/s	1460.27 m/s
Yes	1496.18 m/s	1388.69 m/s	1380.89 m/s

improvement to the cost when knots are included, and this goes hand in hand with the cleanness of the control profiles when knots are well placed. Notice also how the cost improves by adding revolutions to the solution. Comparing the solutions with knots, the improvement in cost is probably not linear with respect to the number of burns; the major improvement probably comes by adding a third finite burn, allowing the plane change to occur at a higher altitude than the initial and final orbits.

Modified Equinoctial Elements

When the DIDO capability was demonstrated on the orbit-raising problem using the modified equinoctial elements, there were some appealing characteristics to the solutions that resulted as compared to those generated using Cartesian dynamics. Overall, there was a cleaner appearance to the equinoctial solutions. It was also stated that employing the element set might lend themselves better to changing the number of orbital periods in the solution as compared to the guess. Whether for good or bad, this characteristic can be seen with the results presented here.

As in the three-revolution case in Cartesian coordinates, a guess is provided to DIDO that takes place over three orbital periods. Interestingly, the resulting DIDO solution is a one-revolution solution, and it is shown in Figure 7-11. As before, the control profile is shown first with 100 nodes (no interior knots), and then with knots. In this example, the accelerations shown on the right came from a converged solution using 120 nodes, with three interior soft knots.
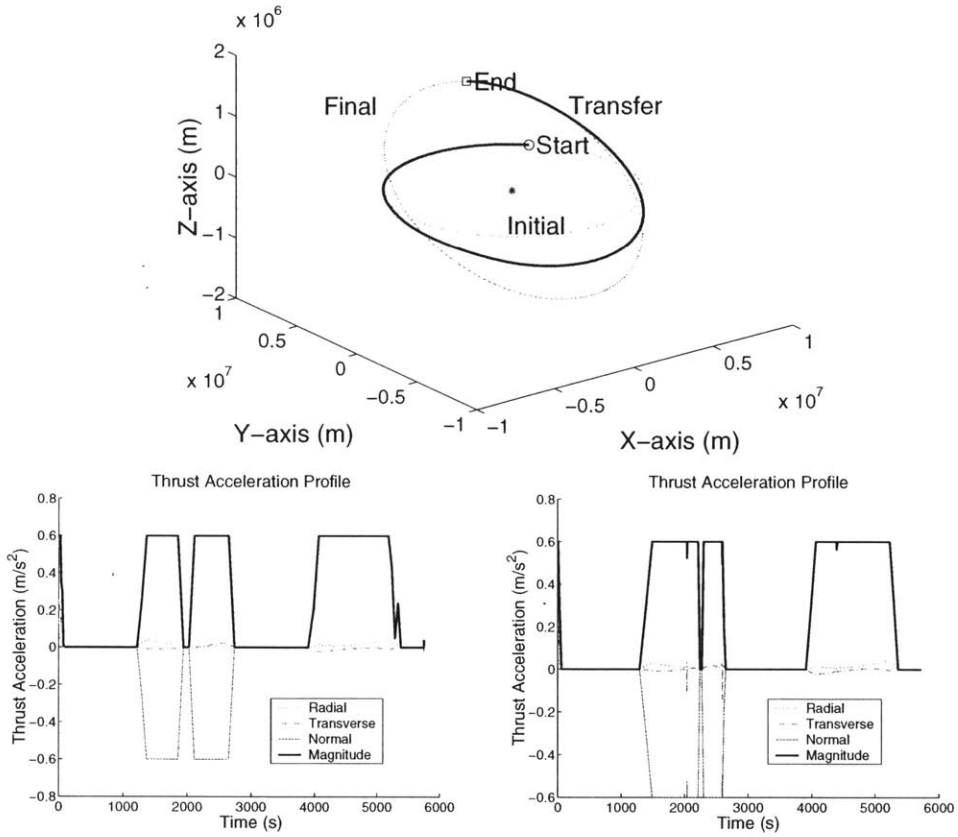
Figure 7-11: Inclination Change Transfer (Modified Equinoctial Elements): Single Revolution Solutions (100,120 nodes)

Table 7.4: Inclination Change Transfer (Cartesian Parameters): Impulsive Solution (8 knots)

Knot	Guess		Solution	
	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	0	0	0
2	34.41	193.01	33.12	180.05
3	78.38	163.22	77.85	992.61
4	114.70	317.30	115.34	0
5	207.32	123.34	212.02	180.19
6	258.91	59.03	258.33	0
7	262.75	30.18	263.59	0
8	267.25	0.22	268.00	0

Consistent with previous results with the modified equinoctial elements, the acceleration profile—even without additional knots—contains much less noise than its Cartesian counterpart; for a one-revolution solution without knots, its cost is respectable: the accumulated Δv is 1532.35 m/s. When the knots are included, the cost is reduced to 1505.14 m/s. Notice that the pecularities of the Cartesian, one-revolution trajectory are exhibited here as well. There is a burn at the initial terminal, and the first major burn contains a quite noticeable division.

It is clear that this is not the optimal solution: in Cartesian coordinates it has been shown that smaller costs can be obtained by solutions with more finite burns. However, with several different starting points, DIDO using equinoctial dynamics was only able to find the one-revolution solution. A point of emphasis in Chapter 6 was that DIDO only promises to find *locally* optimal solutions which may or may not be globally optimal. Quite likely, it is the case for this particular problem that the single revolution trajectory is an easier local minimum to find over the spectrum of solutions. Therefore, the supposed disadvantage of the Cartesian dynamics—that solutions are restricted more heavily by the starting guess—coincidentally worked in the favor of finding a better solution in this case. Since Cartesian coordinates and equinoctial elements both produce feasible solutions (shown in Section 7.3.2), it is recommended that general problems be run in both sets of dynamics in order to capitalize on the advantages of each method.

Cartesian Parameters

An interesting spin on the finite-burn problem comes by taking the results already presented from the finite-burn setup and using them to again find an impulsive solution. In the previous chapter, an impulsive solution was found to the inclination change problem: it was the single-impulse transfer, whose cost was 1357.73 m/s, whether the

maneuver was conducted at the ascending or descending node of the final orbit. Now, what kind of impulsive solution would be found if a multiple finite-burn solution was used to generate an *a priori* guess for the parameter setup?

For this experiment, the three-revolution trajectory found with Cartesian coordinates was used to generate a parameterized, impulsive guess. (This was the solution whose results were presented in Figure 7-9.) Eight hard knots went into the guess, and DIDO was successful in finding a solution, summarized in Table 7.4. Note that only noise causes the final coasting arc.

As the data indicates, a three-impulse solution results from DIDO, whose cost is a Δv of 1352.85 m/s. This is less than the single impulse solution. Analysis indicates that the converged solution has a familiar form: the first burn changes the plane only slightly but raises the altitude, the second conducts the majority of the plane change, the last finishes off the maneuver. There is a symmetry to the maneuver: the exterior burns contribute a Δi of 1.3° each while changing the semi-major axis approximately 55 km. The middle maneuver contributes $\Delta i = 7.4°$. The variable evolution in Figure 7-12 supports this.
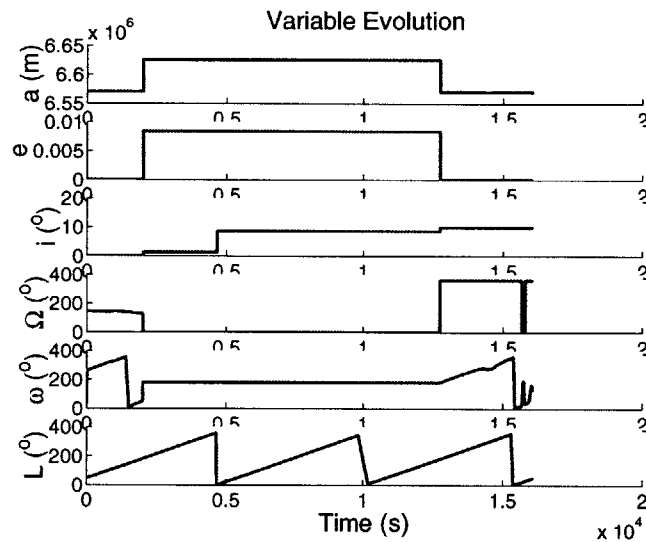

Figure 7-12: Inclination Change Transfer (Cartesian Parameters): Variable Evolution of the Impulsive Solution

131

This too-good-to-be-true result motivated an additional test to understand the benefits of the three-impulse solution to a simple plane change problem. The result just presented was used as a starting point to find a solution to a similar inclination change problem, where the final inclination was changed to 45°, as Table 7.5 shows.

Table 7.5: New Inclination Change Transfer Element Set

Orbital Elements	Initial	Final
a	6,570,000 m	6,570,000 m
e	0.0	0.0
i	0.0°	45.0°
Ω	45.0°	0.0°
u		(Not Specified)

Such a large plane change ($\Delta\theta = 45.0°$), requires an enormous amount of Δv if conducted in a single maneuver. From Equation 7.1, the cost is 5961.50 m/s. In Table 7.6, the DIDO results of this case are presented. It is clear that with multiple maneuvers, this cost is improved to 5473.23 m/s. This is over an 8% improvement! Notice that the impulsive solution has four burns instead of three. The fourth and fifth burns actually occur at the same point in space and in the same direction; they could be combined into one burn. The sum of the last two Δvs is virtually the same as the Δv at the second knot, and their sum also contributes the same amount of inclination change. Therefore the symmetry witnessed in the 10° transfer is preserved.

Table 7.6: New Inclination Change Transfer (Cartesian Parameters): Impulsive Solution (5 knots)

Knot	Guess		Solution	
	t (min)	Δv (m/s)	t (min)	Δv (m/s)
1	0	0	0	0
2	33.12	180.05	33.12	1083.33
3	77.85	992.61	98.14	3306.57
4	115.34	0	163.15	377.54
5	212.02	180.19	274.61	705.80

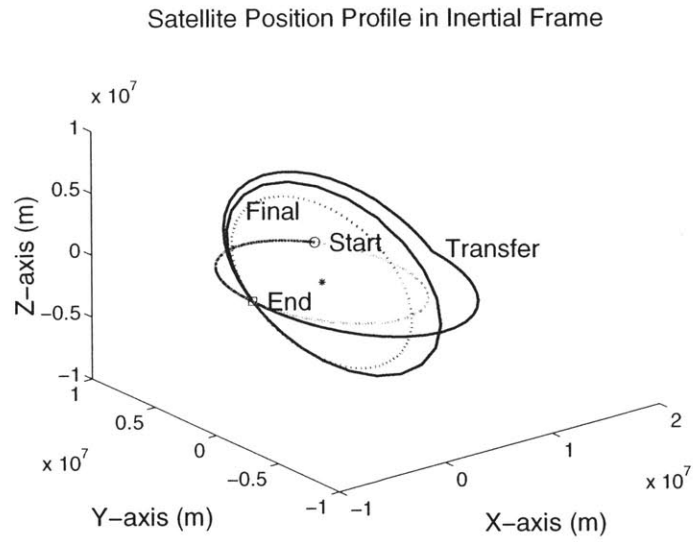Satellite Position Profile in Inertial Frame

Figure 7-13: New Inclination Change Transfer (Cartesian Parameters): Impulsive Solution
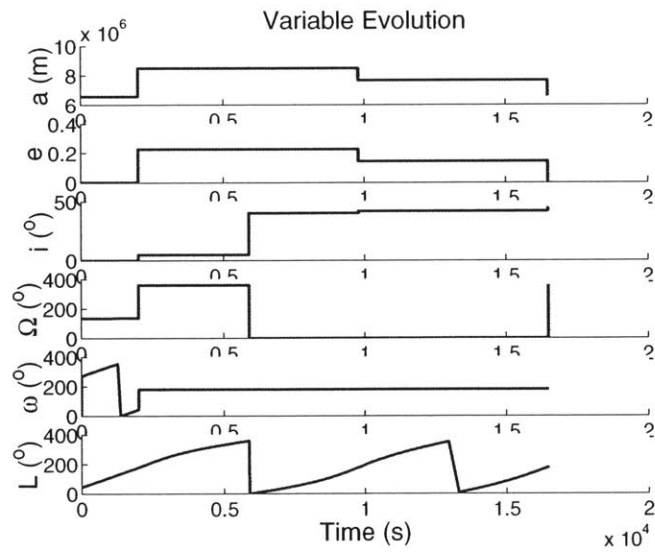

Figure 7-14: New Inclination Change Transfer (Cartesian Parameters): Variable Evolution of the Impulsive Solution

7.2 Application: From LEO (ISS) to LEO (Sun-Synchronous) with J_2

The capability of the finite-burn setup has been demonstrated with two types of problems; both of which yielded very different solutions as compared to their impulsive counterparts. In this section, we apply the technique to more of a real-world problem.

The LEO-LEO transfer described by the element sets in Table 7.7 was used in Chapter 6 when DIDO solved for impulsive solutions. Now, it is appropriate to use this transfer as an example employing finite burns. The dynamics are supplemented to include the perturbations associated with the oblateness of the Earth (J_2).

Table 7.7: LEO-LEO Transfer Element Set

Orbital Elements	Initial	Final
a	6,772,290.24534 m	7,062,996.85533 m
e	0.0007083	0.0011147
i	51.6390°	98.2208°
Ω	58.5505°	120.0745°
ω	238.2837°	282.0257°
ν	261.4820°	(Not Specified)

With the orbit-raising and plane-change problems presented in Section 7.1, the finite-burn solution looked significantly different than the impulsive solution. This was somewhat intentional: the acceleration bound was set low enough as to induce a change in the general shape of the solution. Specifically, impulsive Δv manuevers had to be split up among several finite burns. This does not have to be the case.

In a practical situation, acceleration limitations may not change the total number of burns. It is still interesting to study not only the duration and timing of a burn, but the direction of the thrusting, as it may change over the duration of the burn. Certainly, the impulsive solution is easier to understand in terms of cost, but the finite-burn solution can be extemely interesting, perhaps having more value than the impulsive solution as it also provides a guidance law to follow as a Δv is imposed over some duration of time (however small that might be).

7.2.1 Basic Problem

The LEO-LEO problem is first solved under the same conditions that were imposed on the problems presented in the chapter. The acceleration magnitude is bounded by 0.6 m/s^2. The trajectory that results from the DIDO finite-burn setup has some similar features to those previously seen on this problem.

The problem is solved in both Cartesian coordinates and modified equinoctial elements, as it is beneficial to capitalize on the advantages of both sets of dynamics. It will be shown that the resulting solutions are practically identical with both set of dynamics, thereby validating both solutions. They also come from drastically different starting points.

Cartesian Coordinates

The LEO-LEO transfer was solved with Cartesian dynamics using a reasonable guess, based on the knowledge obtained through the impulsive analysis of the same problem. A series of four iterations on the solution were obtained by progressively refining the descretization by wisely placing knots and nodes:

1. 50 nodes, no interior knots. (Cost: 8844.99 m/s)

2. 100 nodes, no interior knots. (Cost: 8432.51 m/s)

3. 125 nodes, 4 interior soft knots. (Cost: 8019.48 m/s)

4. 150 nodes, 4 interior soft knots. (Cost: 8027.51 m/s)

With each, the solution improved upon the previous trajectory by tightening the control switches and reducing the noise. In Figure 7-15, the results of the fourth converged solution are summarized. The cost of the final converged solution, believed to be the most accurate, is 8027.51 m/s. For this case, knots were absolutely necessary to effectively capture the optimal solution.

Modified Equinoctial Elements

Using the element set affords greater flexibility in providing an initial guess. To text this flexiblity to the extreme, the guess for the equinoctial results presented here was quite poor: a trajectory that remains along the initial orbit due to zero thrusting. Again, a series of iterations were obtained to parallel the process used in Cartesian coordinates:

1. 50 nodes, no interior knots. (Cost: 8045.90 m/s)

2. 100 nodes, no interior knots. (Cost: 8038.31 m/s)

3. 125 nodes, 4 interior soft knots. (Cost: 8032.79 m/s)

4. 150 nodes, 4 interior soft knots. (Cost: 8034.04 m/s)

The position and control profiles are shown in Figure 7-16. Within the position profile, the Cartesian trajectory (labeled "Cartesian") is also included to provide a means of comparison.

Notice that there was little variation in the cost through out each of the equinoctial runs (the cost varies by less than 15 m/s between the solutions; it was over 800 m/s
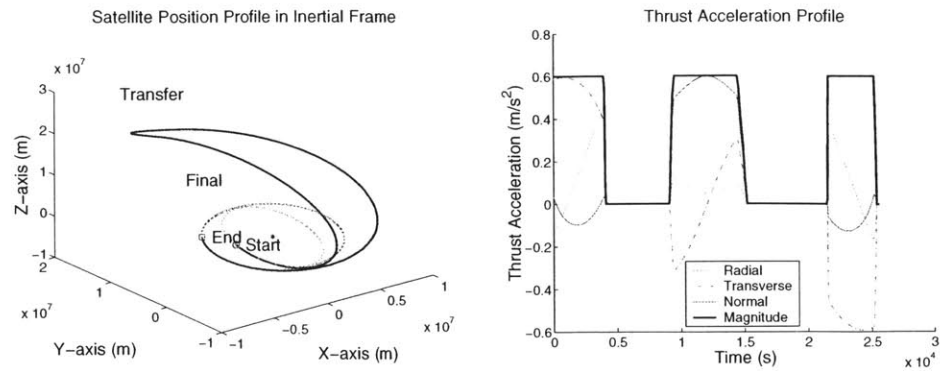
Figure 7-15: LEO-LEO Transfer (Cartesian Coordinates): Final Solution (150 nodes)
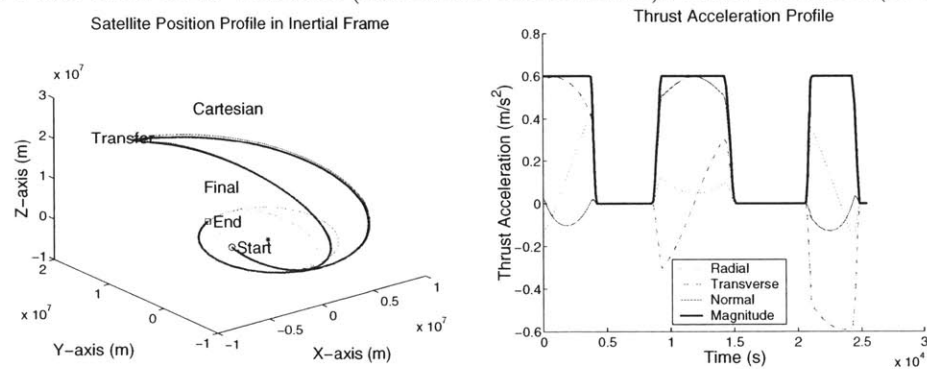

Figure 7-16: LEO-LEO Transfer (Modified Equinoctial Elements): Final Solution (150 nodes)

in Cartesian coordinates). As well, there was little variation in the shape of the trajectory or the control profile. Essentially, using equinoctial elements, DIDO was able to find something very close to the optimal solution with only 50 nodes (and no interior knots), starting from an extremely bad guess. It is still improved with additional knots and nodes, but not to the extent as in Cartesian coordinates.

7.2.2 One Burn at a Time

Throughout this chapter, finite-burn transfer problems have been solved by constraining the initial and final orbits. This results in a control profile consisting of multiple finite burns. The LEO-LEO transfer just presented consisted of three finite burns.

Recall from Chapter 6, that the impulsive solution to this problem was comprised of three impulsive burns. The impulsive solution is useful, as it conveys some information regarding the form and cost of the solution, but it is not practical. In a real situation, a vehicle would conduct some maneuver over a finite amount of time,
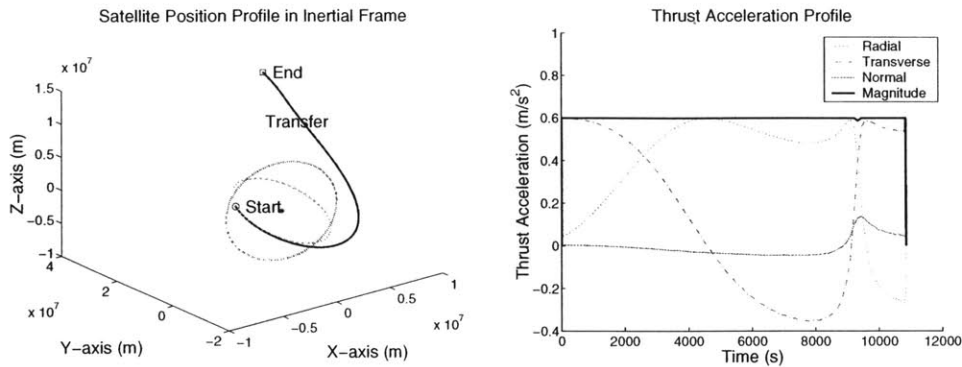
Figure 7-17: LEO-LEO Transfer (Cartesian Coordinates): Finite-Burn Approximation to the First Maneuver of the Impulsive Solution

perhaps approximating the impulsive solution with high-thrust, short-duration burns. This poses two problems.

The first problem is that by changing an impulsive solution to finite-burn control, the trajectory will no longer be optimal. Essentially, impulses equate to unconstrained control, and the impulse is approximated with a finite burn because there is a realistic constraint on the magnitude of thrust allowed. Since Pontryagin's principle for constrained control was not applied, the trajectory cannot be optimal.

The second problem is practical. An impulsive solution introduces a Δv at some time in one direction. However, when an impulse is turned into a finite burn, it is not likely that the vehicle should burn in that one direction the entire time. A steering law is required for the interval of the burn.

DIDO can be used to find the steering law by looking at the impulsive solution, one Δv at a time. Before, we solved the finite-burn problem globally (from initial to final orbit); now we are solving an intermediate problem by using intermediate orbits from a reference trajectory as the initial and final constraints. To demonstrate this, we will look at finding a finite-burn approximation to the first impulse of the LEO-LEO problem solved impulsively last chapter. In Figure 7-17, a single finite burn takes a vehicle from its initial orbit to the transfer orbit of the coasting arc after the first impulsive burn.

This is an important result for several reasons. First of all, it defines the time-varying thrust direction of a finite-burn. Therefore, if in any situation an impulsive solution is known, but a steering law is necessary when it is implemented as a finite-burn impulsive approximation, then this capability can be used to provide it.

The cost of the single finite burn is 6502.39 m/s. When the same maneuver was conducted impulsively, the cost was 2574.92 m/s. In this example, the thrust level is limited significantly, accounting for much of this cost discrepency. As well, the initial time has been fixed and the finite burn begins without an initial coasting arc; it is

137

likely that the finite burn is not optimized because the burn is not allowed to begin any earlier. However, even if the local transfer was optimized, the finite burn will always be more expensive.

More importantly, however, is the cost of the finite-burn solution just evaluated in the last section. The cost in Cartesian coordinates was 8027.51 m/s for the entire transfer; the contribution of the first finite burn of that transfer was 2418.96 m/s, even less than the cost of the first impulsive burn! The reason behind this discrepency is that before, we solved the finite-burn problem globally, finding the optimal solution from initial orbit to final while considering the control constraints. The consequence is that the finite-burn solution does not extend as far in altitude as the impulsive solution. It is not optimal for the finite-burn solution to extend as far. This clearly demonstrates how far from optimal it may be to approximate an impulsive solution with finite burns.

This section serves to connect the capabilities presented both in this chapter and in the previous. However, it also serves to demonstrate the benefits of using DIDO for finite-burn solutions. Since finite burns will always be used in real transfers, it is extremely beneficial to use this capability to find the optimal way to conduct that transfer. Steering laws are inherent in the solution, but also, by solving the actual problem (which is finite-burn, not impulsive), the cost can be drastically improved.

7.3 Solution Feasibility

It is not good engineering practice to simply accept a DIDO solution at first glance. There has already been discussion on the *optimality* of a solution, as an optimizer can easily fall into a local minimum which is not optimal. However, it is also important to question the *feasibility* of a DIDO result. A trajectory is *feasible* if it satisfies the event and dynamical constraints. A feasible trajectory is a solution to the problem, getting a vehicle from the initial conditions to the final conditions through the imposed dynamics, and the optimal solution is just one element out of the subset of feasible trajectories.

When DIDO converges on a solution, all of the constraints are met to some degree of precision at the nodes. However, the constraints are not necessarily met between the nodes, and this is why feasibility becomes an issue. Limitations introduced by node and knot placement can have a more dramatic effect on feasibility.

DIDO solutions contain sets of states and controls at the nodes, and testing feasibility is essentially ensuring that the states match up with the control sequence of the converged solution. This is accomplished by first interpolating the controls over the interval using a spline interpolation function. Then, the interpolated controls are included in the dynamics to propagate the initial states to the terminal time using a Runge-Kutta integration routine. If the R-K routine uses the same dynamics used by the optimizer, then a simple comparison of the final DIDO and Propagated states

offers a great deal of information on the value of the solution [1]. In this section, several figures are presented to compare the DIDO states with the Propagated states (labeled in each figure as "DIDO" and "Propagated") and the test is to see whether the propagated trajectory meets the final terminal constraints (the final orbit). Note that in each of the examples presented here, DIDO converged to what it considered to be a locally optimal solution.

There is a significant difference in solution accuracy between solutions using Cartesian dynamics and solutions using the modified equinoctial elements. It is interesting to compare the solutions of similar problems solved under different dynamics.

7.3.1 Orbit-Raising Transfer

Cartesian Coordinates

Figure 7-18 displays a DIDO trajectory of an orbit-raising transfer that converged with 100 nodes and no interior knots. Recall from Section 7.1.1 that the accumulated Δv from this finite-burn transfer was 3550.74 m/s, significantly less than the impulsive, Hohmann solution. It is not surprising, then, that when the DIDO controls are propagated through the Cartesian dynamics with the R-K integrator, the resulting trajectory is nowhere near the DIDO solution.

As a measure of accuracy, it is useful to compare the final orbital elements of each trajectory to help quantify the issue of feasibility, as we know that the initial conditions and dynamics will be met with the R-K integration. The final elements of the DIDO and propagated trajectories are listed in Table 7.8.

Table 7.8: Circular-Circular Orbit-Raising Transfer Element Set (End Terminal for DIDO Solution # 1)

Orbital Elements	DIDO	Propagated
a	42,160,000 m	8,226,230.64 m
e	0.0	0.14448
i	0.0°	0.33501°

This disturbing result is on account of the relationship between the trajectory and the node count and placement. With this problem, the solution takes place over more than five orbital revolutions, and DIDO can only account for this with 100 nodes. When knots are added and the node count is increased, how much more feasible, or accurate, will the solution be?

---

[1] In checking feasibility, one must also take into account any peculiarities of the ODE solver. One must be careful in propagating simple two-body dynamics without perturbations. The addition of controls (i.e. perturbing accelerations) can lead to a very difficult ODE problem.

Figure 7-18: Circular-Circular Orbit-Raising Transfer (Cartesian Coordinates): DIDO Trajectory vs. Propagated Trajectory (Solution # 1)

In Figure 7-19 and Table 7.9, the DIDO and propagated trajectories are compared when the solution has 180 nodes separated with five interior soft knots. Interestingly, the discrepancies between the two trajectories are still dramatic. It is clear that, even with the additional nodes and knots, the solution has not really improved very much. Even with 180 nodes, the problem that DIDO is solving is not representative of the actual continuous problem.
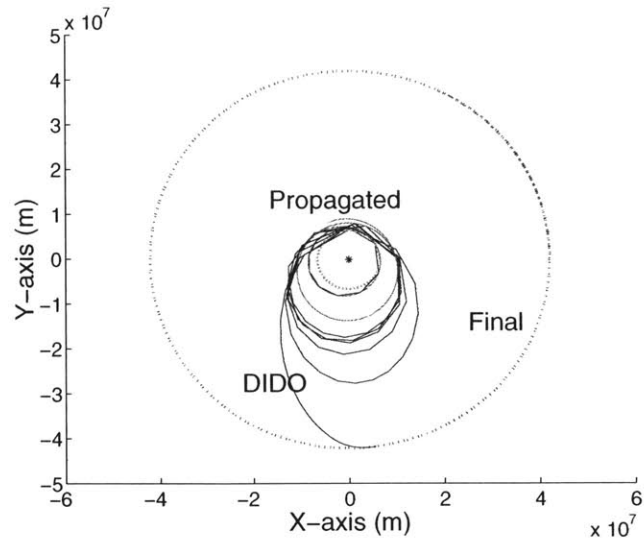
Figure 7-19: Circular-Circular Orbit-Raising Transfer (Cartesian Coordinates): DIDO Trajectory vs. Propagated Trajectory (Solution # 2)

Table 7.9: Circular-Circular Orbit-Raising Transfer Element Set (End Terminal for DIDO Solution # 2)

Orbital Elements	DIDO	Propagated
a	42,160,000 m	7,716,433.33 m
e	0.0	0.19112
i	0.0°	0.38298°

141

Additional knots or nodes do not help. Running this problem in Cartesian coordinates essentially requires a smaller discretization than is convenient when the solution has so many revolutions. To support this conclusion, observe the improved accuracy of propagating the controls from a trajectory that takes place over a single orbital revolution, shown in Figure 7-20 and Table 7.10. In this case, using 150 nodes and two interior knots, DIDO converged on a local minimum far different than the previous cases. However, the control values of this solution more accurately represent a feasible solution. Figure 7-21 compares the results in another way.
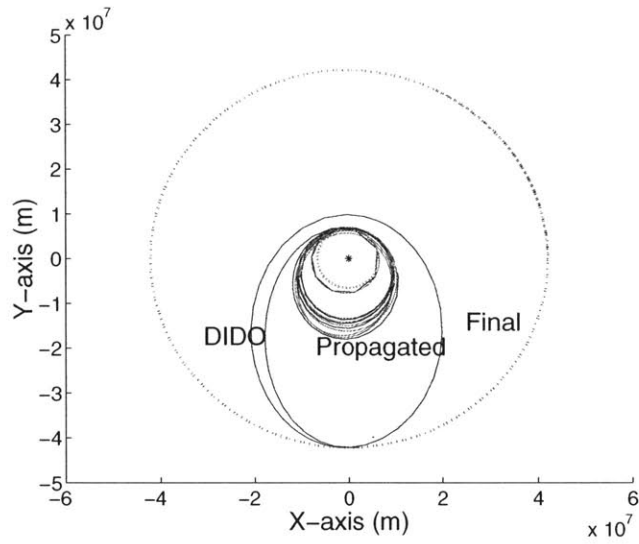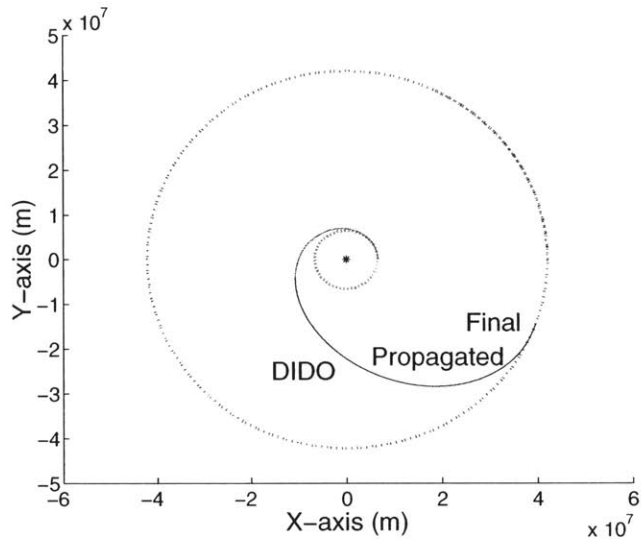

Figure 7-20: Circular-Circular Orbit-Raising Transfer (Cartesian Coordinates): DIDO Trajectory vs. Propagated Trajectory (Solution # 3)

Table 7.10: Circular-Circular Orbit-Raising Transfer Element Set (End Terminal for DIDO Solution # 3)

Orbital Elements	DIDO	Propagated
a	42,160,000 m	42,157,291.69 m
e	0.0	0.00004
i	0.0°	0.00003°

142

Figure 7-21: Circular-Circular Orbit-Raising Transfer (Cartesian Coordinates): DIDO States vs. Propagated States (Solution # 3)

143

Modified Equinoctial Elements

Implied above is that describing the dynamics in orbital elements leads to more accurate results, and this is demonstrated here. First, without knots, a DIDO solution is compared with a R-K propagation. The propagation in Figure 7-22 and Table 7.11 is a significant improvement on any of the multiple-revolution solutions shown above.
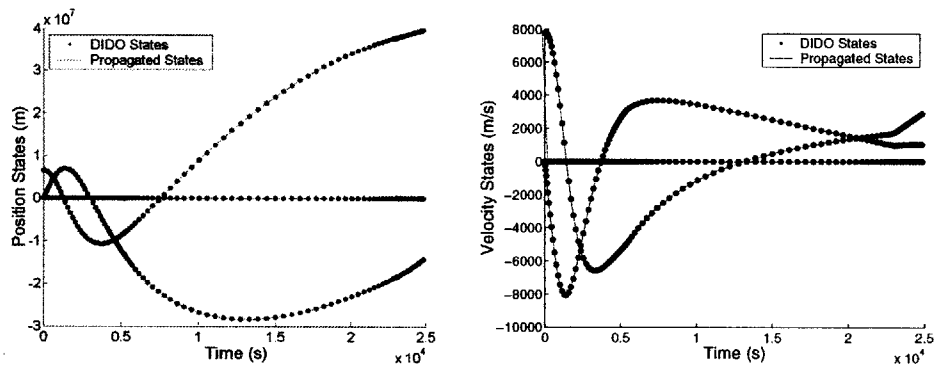

Figure 7-22: Circular-Circular Orbit-Raising Transfer (Modified Equinoctial Elements): DIDO Trajectory vs. Propagated Trajectory (Solution # 1)

Table 7.11: Circular-Circular Orbit-Raising Transfer Element Set (End Terminal for DIDO Solution # 1)

Orbital Elements	DIDO	Propagated
a	42,160,000 m	38,563,927.02 m
e	0.0	0.21960
i	0.0°	0.33637°

Figure 7-23 compares the element sets of the DIDO and propagated trajectories. In general, the control profile from this equinoctial elements solution is much cleaner (recall Figure 7-4). The result of its cleanness is seen in the similarity between DIDO and propagated states. As well, it also facilitates the process of adding knots and nodes (it is much easier for a user to decide where to place knots).
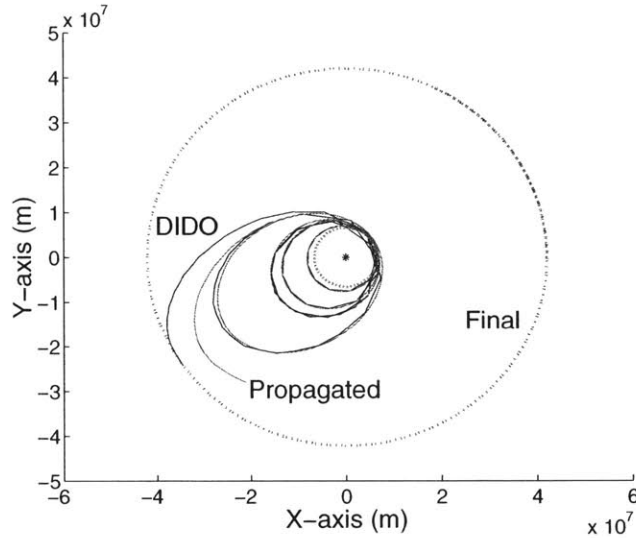
Figure 7-23: Circular-Circular Orbit-Raising Transfer (Modified Equinoctial Elements): DIDO States vs. Propagated States (Solution # 1)

When knots are added to the problem (nine soft knots dividing 200 nodes), the resulting DIDO trajectory matches much more closely the propagated trajectory. Figure 7-24 compares the states with knots. Table 7.12 compares the terminal constraints. Notice that the a in the Propagated solution differs from the constraint by 10 km. Considering the scaling of the problem, a 10 km difference over 42,000 km is mostly within the tolerance of convergence.
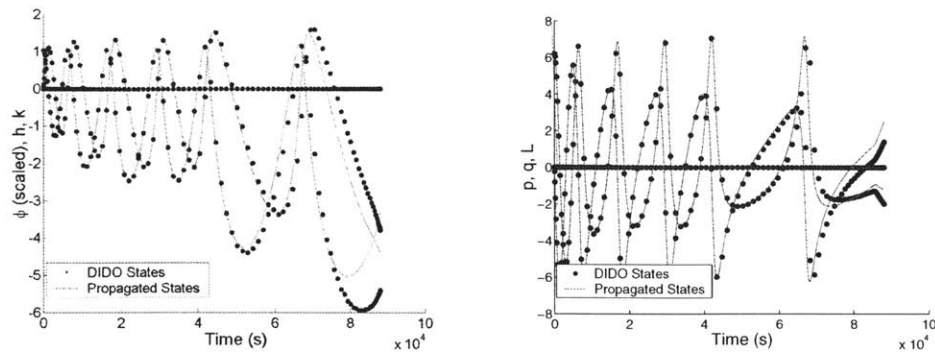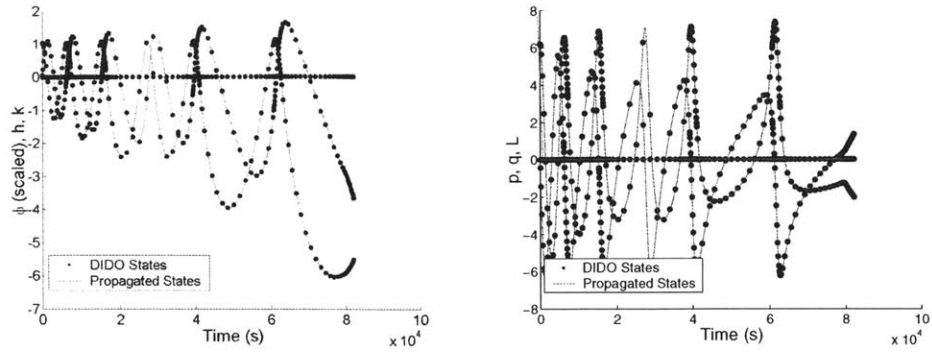
Figure 7-24: Circular-Circular Orbit-Raising Transfer (Modified Equinoctial Elements): DIDO States vs. Propagated States (Solution # 2)

Table 7.12: Circular-Circular Orbit-Raising Transfer Element Set (End Terminal for DIDO Solution # 2)

Orbital Elements	DIDO	Propagated
a	42,160,000 m	42,150,473.59 m
e	0.0	0.00081
i	0.0°	0.00200°

7.3.2 Inclination Change Transfer

The inclination change transfer is on a much smaller scale than the orbit-raising problem: the transfer time is much smaller, whether the transfer occurs over a single revolution or over three revolutions. Consequently, it is much more manageable, and it allows for cleaner control profiles, whether they are determined using Cartesian coordinates or equinoctial elements.

Cartesian Coordinates

To demonstrate the accuracy of this transfer in Cartesian coordinates, the states (position and velocity) of the DIDO and propagated trajectories are compared in Figure 7-25 and Table 7.13. In this case, 200 nodes are sectioned by nine interior soft knots.

146

Figure 7-25: Inclination Change Transfer (Cartesian Coordinates): DIDO States vs. Propagated States

Table 7.13: Inclination Change Transfer Element Set (End Terminal)

Orbital Elements	DIDO	Propagated
a	6,570,000 m	6,569,606.35 m
e	0.0	0.00006
i	10.0°	9.99284°
Ω	0.0°	359.99808°

147

Modified Equinoctial Elements

With the modified equinoctial elements, the resulting solution takes place over a single revolution (regardless of the guessed trajectory). With 120 nodes and three soft knots, the states compare quite respectably (Figure 7-26 and Table 7.14).
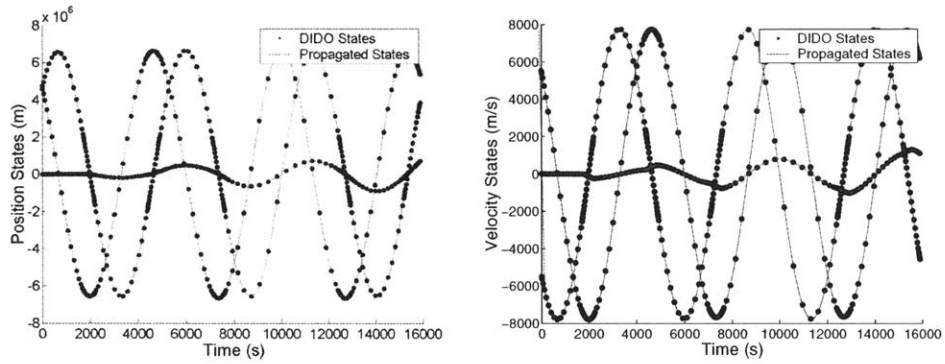

Figure 7-26: Inclination Change Transfer (Modified Equinoctial Elements): DIDO States vs. Propagated States

Table 7.14: Inclination Change Transfer Element Set (End Terminal)

Orbital Elements	DIDO	Propagated
a	6,570,000 m	6,569,058.77 m
e	0.0	0.00030
i	10.0°	9.98668°
Ω	0.0°	0.01533°

Inclination Change with Cartesian Parameters

It has already been shown that a simple inclination change can be conducted over three impulses with a cost less than the traditional single impulse solution. One may question the feasibility of the trajectory from the DIDO solution, since the solution is almost too good to be true. In Figure 7-27, the position and velocity states are compared from the DIDO solution and an R-K integration of the parameter controls found in the DIDO solution. These two trajectories are absolutely consistent, as supported by the data in Table 7.15.
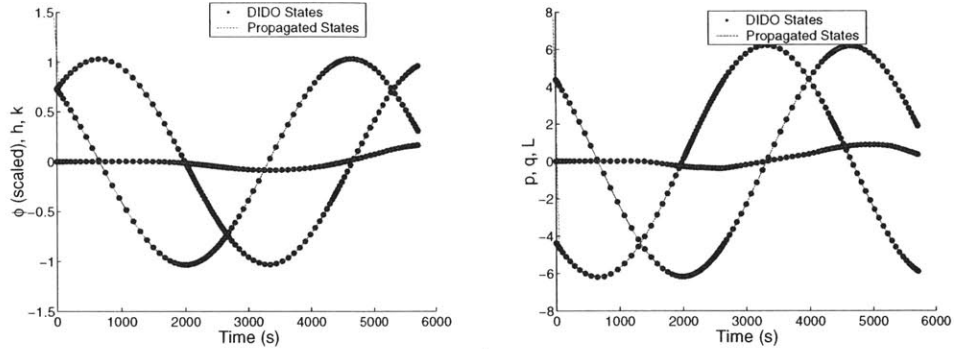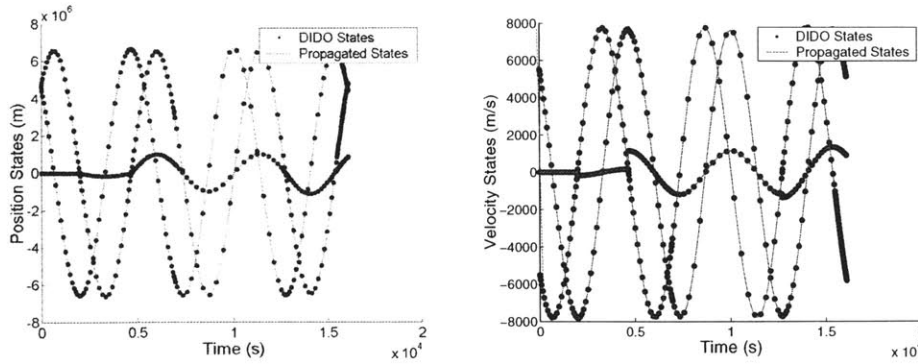
148

Figure 7-27: Inclination Change Transfer (Cartesian Parameters): DIDO States vs. Propagated States

Table 7.15: Inclination Change Transfer Element Set (End Terminal)

Orbital Elements	DIDO	Propagated
a	6,570,000 m	6,570,000.01 m
e	0.0	0.00000
i	10.0°	10.00000°
Ω	0.0°	0°

7.3.3 From LEO (ISS) to LEO (Sun-Synchronous) with J_2

Finally, we check the feasibility of the real-world example from Section 7.2. When 150 nodes were used in DIDO with Cartesian dynamics, the DIDO solution compared to an R-K integration as shown in Figure 7-28 and Table 7.16. Notice that in the table, the final elements are shown for a propagation with J_2 and without. This is to show the level to which the perturbation effects the solution.

In this section, the feasibility concerns associated with using DIDO were presented. Namely, when DIDO converges on a solution, it satisfies constraints at the nodes only, so while it does solve the transcribed, nonlinear programming problem, it may not be an accurate representation of a continuous solution. Therefore, it is important that the user test the feasibility of a DIDO solution by interpolating the controls and propagating the states through the dynamics to ensure that the end conditions are met to some degree of accuracy. This has been accomplished for both of the Capability Demonstration problems, as well as for the Application problem from this chapter.

Throughout the chapter, the finite-burn problem has been demonstrated as it is solved using the Legendre Pseudospectral technique. The advantages and disadvantages of

149
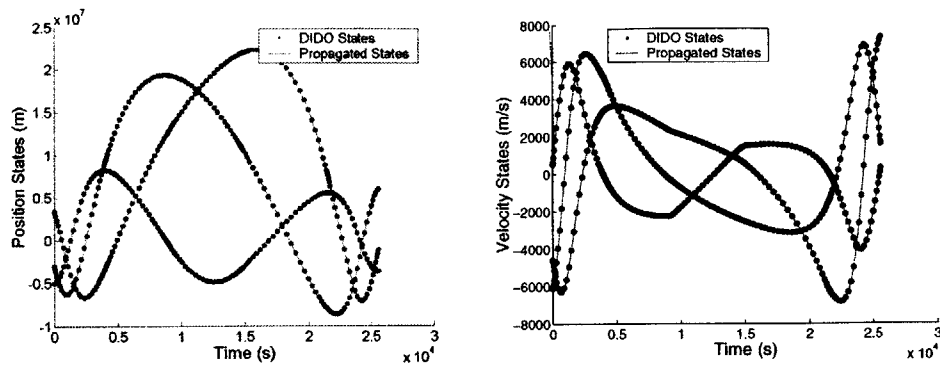
Figure 7-28: LEO-LEO Transfer (Cartesian Coordinates): DIDO States vs. Propagated States

Table 7.16: LEO-LEO Transfer Element Set (End Terminal)

Orbital Elements	DIDO	Prop. (w/ J_2)	Prop. (w/o J_2)
a	7,062,996.85533 m	7,063,069.84570 m	7,070,400.34668
e	0.0011147	0.0009757	0.0065604
i	98.2208°	98.2188°	98.3662°
Ω	120.0745°	120.0752°	119.9503°
ω	282.0257°	278.0714°	348.2986°

two state definitions (Cartesian position and velocity versus modified equinoctial elements) have been presented in the context of three orbital transfer problems. While each set of dynamics has its weaknesses, it is recommended to use both simultaneously to capitalize on their strengths. It is also recommended to be somewhat skeptical when DIDO produces a solution: the feasibility of that trajectory may be in question.

150

Chapter 8

Conclusion

8.1 Summary

Orbital transfer has been a topic of interest within the astrodynamics community for over 40 years. It is commonly necessary for a vehicle to change from one orbit to another through a series of maneuvers. Many different methods have been used in orbital transfer optimization. In this thesis, the orbital transfer problem has been examined using the Legendre Pseudospectral Method.

The Legendre Pseudospectral Method, developed by Ross and Fahroo, has been implemented in the powerful software package DIDO, a MATLAB optimization tool-box also developed by Ross and Fahroo. The method offers a robust, accurate approach to transcribing a continuous optimal control problem into a nonlinear programming problem by optimally placing the discretization points, or nodes.

DIDO was used here to solve two types of orbital transfer problems, impulsive and finite-burn, where trajectories were optimized to minimize fuel. The impulsive problem was solved in two ways. In the first, continuous controls were allowed to approximate impulses as high-thrust, short-duration burns. In the second, DIDO's parameter optimization capability was exploited to solve for exact, impulsive solutions. Continuous controls were also used to find finite-burn solutions, where the magnitude of allowable thrust acceleration was bounded significantly to force longer-duration burns.

The two control settings—continuous and impulsive controls—were used in complement in a number of different ways:

1. Continuous solutions served to pave the way for impulsive solutions. High-thrust, continuous-control (finite-burn) solutions approximated impulses, and they were used to construct guesses for purely impulsive solutions implemented by parameter optimization.

2. Realizable (finite-burn) solutions were compared to theoretical (impulsive) solutions. As the time interval of a finite-burn trajectory was allowed

to increase, the duration of individual finite burns decreased, making the maneuvers more impulsive. The cost appeared to approach that of the impulsive solution. Therefore, a trade-off resulted between trajectory time and cost.

3. The process of increasing the time interval for finite-burn transfer led to a new impulsive solution to a simple plane-change problem. A three-impulse solution proved to be superior in cost to an ordinary single-impulse solution.

4. Impulsive solutions were converted to finite-burn solutions to illustrate how an impulsive solution could be implemented by real thrusters. The cost difference in realizing an impulsive solution one burn at a time indicated the limitations in impulsive transfer solutions. Even if the finite-burn realization is optimized, it is better to solve a global problem considering the constraints imposed by propulsion technology.

The impulsive and finite-burn capabilities were demonstrated with great success in this thesis. By using transfer scenarios to which the solution wass already known, the impulsive capability was validated. Then, it was applied to real-world problems to outline its usefulness when the solution is not as intuitive. Finite-burn solutions were validated by applying the intuition gained through impulsive solutions. The finite-burn capability was effectively applied to demonstrate its use in finding solutions to even more realistic problems (since thrusters can realistically thrust according to a finite-burn profile). It was also demonstrated how the capability could serve to develop steering laws for finite-duration thrusting.

8.2 Future Work

This thesis is in no way an exhaustive work on the topic. The Legendre Pseudospectral Method can be used to solve even more complex orbital transfer problems:

1. Within the minimum-fuel problem, additional complexities could be added.

> a. The implementation of a drag model has been outlined, but it has not been implemented. In its simplest form, drag accelerations are added to the dynamical equations. It is also possible to consider additional event constraints dictating the arcs in which the drag perturbation is an effect.
>
> b. Additional perturbations, like the effects of a third-body (the moon or the Sun for Earth-centered problems) should be considered for extremely high-altitude orbital transfer problems.

c. Other path constraints could be considered. For example, a vehicle may be required to stay within some operating window for part or all of a trajectory interval.

d. Control constraints could be imposed. Realistically, there may be contraints on the direction of a burn, as certain directions may be prohibited due to the impact on a payload or the limitations of an attitude control system.

2. The implications of a variable thrust direction should be considered. Changing thrust direction generally requires a change of vehicle attitude. The costs of attitude control should be taken into account. By considering attitude control, the problem changes from three to six degees of freedom.

3. In this thesis, minimum-effort or minimum-fuel problems were solved. The minimum-time problem is also of significant interest in astrodynamics. This is an easier problem in the absence of thrust direction constraints as there are no coasting arcs.

4. The orbital rendezvous problem could also be examined. In this thesis, the final terminal is defined by five elements, the slow variables of an orbit. Solving for the sixth element, as well, identifying a location on a final orbit, also contributes to problem complexity.

5. Further work should be accomplished in validating the optimality of a DIDO solution. It would be interesting to compare DIDO solutions to indirect method solutions (i.e. two-point boundary-value-problem solutions). In cases without interior knots, DIDO generates estimates of the costates which cn be used to facilitate the effort of optimality validation. Current research by Ross and Fahroo is under way for cases with interior knots.

[Except for this sentence, this page intentionally left blank.]

Bibliography

[1] Atkinson, K.E., *An Introduction to Numerical Analysis.* New York: John Wiley & Sons, 1978.

[2] Baumann, H., "Thrust Limited Coplanar Aeroassisted Orbital Transfer," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 4, July-Aug 2001, pp. 732-738.

[3] Bell, D.J., and Jacobson, D.H., *Singular Optimal Control Problems.* Academic Press, London, 1975.

[4] Betts, J.T., "A Direct Approach to Solving Optimal Control Problems." Computing in Science & Engineering, May-June 1999.

[5] Bryson, A.E., and Ho, Y., *Applied Optimal Control: Optimization, Estimation, and Control.* Taylor & Francis, New York, 1975.

[6] Burden, R.L., Faires, J.D., and Reynolds, A.C., *Numerical Analysis.* Boston: Prindle, Weber, & Schmidt, 1978.

[7] Chuang, C.-H., Goodson, T.D., and Ledsinger, L.A., "Optimality and Guidance for Planar Multiple-Burn Orbit Transfers," *Journal of Guidance, Control, and Dynamics*, Vol. 19, No. 6, Nov-Dec, 1996, pp. 1310-1316.

[8] Conte, S.D., and de Boor, C., *Elementary Numerical Analysis: An Algorithmic Approach.* New York: McGraw-Hill, 1972.

[9] Conway, B. A., and Larson K. M., "Collocation Versus Differential Inclusion in Direct Optimization." Journal of Guidance, Control, and Dynamics, Vol. 21, No. 5, 1998, pp. 780-785.

[10] Davis, P.J., and Rabinowitz, P., *Methods of Numerical Integration.* 2nd ed. Harcourt Brace Jovanovich, Orlando, 1984.

[11] Dixon, L. C. W., and Biggs, M. C., "The Advantages of Adjoint Control Transformations when Determining Optimal Trajectories by Pontryagin's Maximum Principle," *The Aeronautical Journal*, Vol. 76, No. 735, 1972, pp. 169-174.

[12] Edelbaum, T.N., "How Many Impulses?" *Astronautics & Aeronautics*, Nov 1967, pp. 64-69.

[13] Elnagar, J., Kazemi, M. A., and Razzaghi, M., "The Pseudospectral Legendre Method for Discretizing Optimal Control Problems," IEEE Transactions on Automatic Control, Vol. 40, No. 10, 1995.

[14] Fahroo, F., and Ross, I. M., "Costate Estimation by a Legendre Pseudospectral Method," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 2, Jul-Aug 2001, pp. 270-277.

[15] Fahroo, F., and Ross, I. M., "Second Look at Approximating Differential Inclusions." Journal of Guidance, Control, and Dynamics, Vol. 24, No. 1, 2001, pp. 131-133.

[16] Fahroo, F., and Ross, I. M., "A Spectral Patching Method for Direct Trajectory Optimization," *The Journal of the Astronautical Sciences*, Vol. 48, No. 2/3, Apr-Sep 2000, pp. 269-286.

[17] Gill, P.E., Murray, W., and Saunders, M.A., *User's Guide for SNOPT Version 6: A Fortran Package for Large-Scale Nonlinear Programming.* Systems Optimization Laboratory, Standford, CA, Dec 2002.

[18] Hamming, R.W., *Numerical Methods for Scientists and Engineers.* New York: McGraw-Hill, 1962.

[19] Hargraves, C.R., and Paris, S.W., "Direct Trajectory Optimization Using Nonlinear Programming and Collocation" Journal of Guidance, Control, and Dynamics, Vol. 10, No. 4, 1987, pp. 338-342.

[20] Ilgen, M.R., "A Hybrid Method for Computing Optimal Low Thrust OTV Trajectories," American Astronautical Society, AAS Paper 94-129, Feb. 1994.

[21] Kechichian, J.A., "Minimum-Time Constant Acceleration Orbit Transfer with First-Order Oblateness Effect," *Journal of Guidance, Control, and Dynamics*, Vol. 23, No. 4, July–Aug 2000, pp. 595-603.

[22] Kirk, D.E., *Optimal Control Theory: An Introduction.* Prentice Hall, Englewood Cliffs, NJ, 1970.

[23] Kuhn, H.W. and Tucker, A.W., "Nonlinear programming," *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, Ed. J. Neyman, University of California Press, 1951.

[24] Kluever, C.A., and Oleson, S.R., "Direct Approach for Computing Near-Optimal Low-Thrust Earth-Orbit Transfers," *Journal of Spacecraft and Rockets*, Vol. 35, No. 4, Jul-Aug 1998, pp. 509-515.

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

[26] Lebedev, N.N., *Special Functions and Their Applications*. Dover Publications, NY, 1972.

[27] London, H.S., "Change of Satellite Orbit Plane by Aerodynamic Maneuvering," *Journal of the Aerospace Sciences*, Vol. 29, 1962, pp. 323-332.

[28] Matogawa, Y., "Optimal Low Thrust Transfer to Geosynchronous Orbit," *Acta Astronautica*, Vol. 10, No. 7, 1983, pp. 467–478.

[29] Miele, A., Wang, T., and Lee, W. Y. "Optimization and Guidance of Trajectories for Coplanar, Aeroassisted Orbital Transfer," *Journal of the Astronautical Sciences*, Vol. 38, No. 3, Jul-Sep 1990, pp. 311-333.

[30] Naidu, D., *Aeroassisted Orbital Transfer*, Springer-Verlag, Berlin, 1994, Chaps. 2,3.

[31] Naidu, D. S., "Fuel-Optimal Trajectories of Aeroassisted Orbital Transfer with Plane Change," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 27, No. 2, Mar 1991, pp. 361-368.

[32] Prussing, J.E., and Chiu, J.-H., "Optimal Multiple-Impulse Time-Fixed Rendezvous Between Circular Orbits," *Journal of Guidance, Control, and Dynamics*, Vol. 9, No. 1, 1986, pp. 17-22.

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

[34] Ross, I. M., and Fahroo, F., "A Direct Method for Solving Nonsmooth Optimal Control Problems," (in preparation).

[35] Ross, I. M., and Fahroo, F., "A Pseudospectral Transformation of the Covectors of Optimal Control Systems," *Proceedings of the First IFAC Symposium on System Structure and Control*, Prague, Czech Republic.

[36] Ross, I. M., and Fahroo, F., *User's Manual for DIDO 2003a*. Naval Postgraduate School, Monterey, CA, Jul 2001.

[37] Sackett, L.L., Malchow, H.L., and Edelbaum, T.N., "Solar Electric Geocentric Trnasfer with Attitude Constraints: Analysis," NASA CR-134927, Aug. 1975.

[38] Sellers, J.J., Astore, W.J., Crumpton, K.S., Elliot, C., and Giffen, R.B., *Understanding Space: An Introduction to Astronautics*. McGraw-Hill, Inc., New York, 1994.

[39] Seywald, H., "Trajectory Optimization Based on Differential Inclusion." Journal of Guidance, Control, and Dynamics, Vol. 17, No. 3, 1994, pp. 480-487.

[40] Spencer, D.B., and Culp, R.D., "Designing Continuous-Thrust Low-Earth-Orbit to Geosynchronous-Earth-Orbit Transfers," *Journal of Spacecraft and Rockets*, Vol. 32, No. 6, Nov-Dec, 1995, pp. 1033-1038.

[41] Stanford Business Software Inc (SBSI) site for the distribution of SOL Optimization Software, http://www.sbsi-sol-optimize.com.

[42] Thorne, J.D., and Hall, C.D., "Minimum-Time Continuous-Thrust Orbit Transfers," *Journal of the Astronautical Sciences*, Vol. 45, No. 4, Oct-Dec 1997, pp. 411-432.

[43] Thorvaldsen, T.P., Proulx, R.J., and Ross, I.M., "A fast, Accurate Method for Low-Thrus Trajectory Optimization" 16th Spaceflight Dynamics Symposium, Pasadena, CA, Dec 3-7, 2002.

[44] Trefethen, L.N., *Spectral Methods in Matlab*. SIAM Press, Philadelphia, 2000.

[45] Vallado, D.A., *Fundamentals of Astrodynamics and Applications*. New York: McGraw-Hill, 1997.

[46] von Stryk, O., Burlirsch, R., "Direct and Indirect Methods for Trajectory Optimization." 1992.

[47] Walker, M.J.H., Ireland, B., and Owens, J., "A Set of Modified Equinoctial Orbit Elements." *Celestial Mechanics* 36, 1985, pp. 409–419.

[48] Wertz, J.R., *Spacecraft Attitude Determination and Control*. Dordrecht, Holland: D. Reidel Publishing Company, 1978.

[49] Won, C.-H., "Fuel- or Time-Optimal Transfers Between Coplanar, Coaxial Ellipses Using Lambert's Theorem," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 4, Jul-Aug 1999, pp. 536-542.

[50] Yan, H., and Wu, H., "Initial Adjoint-Variable Guess Technique and Its Application in Optimal Orbital Transfer," *Journal of Guidance, Control, and Dynamics*, Vol. 22, No. 3: Engineering Notes, pp. 490-492.

[51] Zondervan, K.P., Wood, L.J., and Caughey, T.K., "Optimal Low-Thrust, Three-Burn Orbit Transfers with Large Plane Changes," *Journal of the Astronautical Sciences*, Vol. 32, No. 3, 1984, pp. 407–427.