OPTIMAL TRAJECTORIES FOR CONSTRAINED STATION CHANGE IN GEO USING A LEGENDRE PSEUDOSPECTRAL METHOD

A Dissertation in Aerospace Engineering
by
Seung Pil Kim

© 2012 Seung Pil Kim

Submitted in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

August 2012
The dissertation of Seung Pil Kim was reviewed and approved* by the following:

Robert G. Melton
Professor of Aerospace Engineering
Dissertation Advisor, Chair of Committee

David B. Spencer
Professor of Aerospace Engineering

Joseph F. Horn
Associate Professor of Aerospace Engineering

Julio A. Urbina
Associate Professor of Electrical Engineering

George A. Lesieutre
Professor of Aerospace Engineering
Head of the Department of Aerospace Engineering

*Signatures are on file in the Graduate School.
Abstract

In this dissertation a method for determining optimal trajectories for constrained geostationary station change is presented. If a satellite failure occurs, an emergency station relocation maneuver could be required to replace a malfunctioning satellite and continue to maintain the operational capabilities for a particular mission. The algorithm for numerical approximation of the optimal control is formulated using a Legendre pseudospectral method. The state equations are enforced at each node using a differentiation matrix and forcing the state derivatives from the Legendre polynomial representation to be equal to the state equation derivatives evaluated at the Legendre-Gauss-Lobatto points. An advantage of a pseudospectral method is that it can determine the optimal control using fewer unknown parameters (the states and controls and the node points) than other direct methods. The spacecraft dynamics equations are formulated in 2-D polar form; and perturbation forces are neglected except for the low-thrust control. The collision avoidance term, which is included in the objective function as an integral form, is considered during transfer by specifying a maximum or minimum radius depending on whether the transfer
is in the east or west-direction. Several east-and-west-direction station change transfer maneuvers are simulated; the histories of the states and control behavior are obtained and all the variables are found to be within feasible ranges. Multiple revolutions and large longitude station change cases are also demonstrated using this method. The transfer time is found to be only weakly affected by the initial thrust acceleration for the multi-revolution change cases. This method would be possible to implement using very low-thrust engines, in particular, by employing attitude control thrusters.
Table of Contents

List of Figures viii
List of Tables x
List of Symbols xi
Acknowledgments xiii

Chapter 1
  Introduction 1
  1.1 Geostationary Satellites .............................. 1
  1.2 Station Change ........................................ 3
  1.3 Station-change Maneuvers ............................. 4
    1.3.1 Station Keeping .................................. 4
    1.3.2 Two-Impulse Maneuver ............................ 4
  1.4 Literature Review ..................................... 11
  1.5 Dissertation Objectives ............................... 13
  1.6 Thesis Outline ........................................ 14

Chapter 2
  Optimal Control Theory 17
  2.1 Optimal Control Applications .......................... 17
  2.2 The Optimal Control Problem ........................... 18
    2.2.1 Mathematical formulation .......................... 19
  2.3 Trajectory Optimization ............................... 23
    2.3.1 Indirect Method .................................. 23
    2.3.2 Direct Methods .................................. 25
    2.3.3 Collocation ...................................... 25
List of Figures

1.1 Geostationary Satellite ............................................. 2
1.2 Two-impulse geostationary orbit longitude change (80 deg.) west-direction for various number of revolutions .......................... 6
1.3 Two-impulse drift rate .................................................. 9
1.4 Two-impulse total $\Delta V$ ............................................ 9
1.5 Semi-major axis with respect to transfer time for west 80-degree station change .......................................................... 10
1.6 Delta V (km/s) profile with respect to the number of revolutions for various station changes .............................................. 10

2.1 Hermite Cubic Segment Collocation .................................. 30
2.2 State, control and co-state history from the TPBVP ................. 34
2.3 State and control history from the direct method ..................... 34
2.4 Comparison of direct and indirect results ................................. 35

3.1 Legendre polynomial $L_N(\tau)$ for $N = 0, 1, 2, 3, 4$ .................. 45
3.2 LGL points $N = 10, 20, 40$ ............................................ 45
3.3 Pseudospectral differentiation error order N=20, $f(t) = \cos(2t^2)$ .. 51
3.4 Pseudospectral differentiation error order N=30, $f(t) = \cos(2t^2)$ .. 51
3.5 State for minimum cost problem ....................................... 54
3.6 Control for minimum cost problem .................................... 55
3.7 Co-state vs. state for minimum cost problem .......................... 55
3.8 Radius history for maximum final radius problem ..................... 60
3.9 Radial velocity history for maximum final radius problem ........... 60
3.10 Tangential velocity history for maximum final radius problem ..... 61
3.11 Control angle history for maximum final radius problem .......... 61
3.12 Trajectories for the orbit raising problem ............................ 62
3.13 Trajectories for maximum energy at final time for 30 LGL nodes . 65
3.14 Trajectories for maximum energy at final time for 64 LGL nodes . 65
3.15 Trajectories in XY coordinates for maximum energy at final time . 66
3.16 Energy for maximum energy at final time ............................ 66
3.17 Radial velocity history for maximum energy at final time . . . . . . . 67
3.18 Tangential velocity history for maximum energy at final time . . . . 67
3.19 Control angle for maximum energy at final time . . . . . . . . . . . 68
3.20 Trajectories for maximum energy at final time, explicit integral results 68

4.1 GEO satellite motion . . . . . . . . . . . . . . . . . . . . . . . . . . 73
4.2 Trajectory for east 10 degrees: 2 rev, acc=0.01 . . . . . . . . . . . 81
4.3 Radius for east 10 degrees: 2 rev, acc=0.01 . . . . . . . . . . . . . 81
4.4 Radial velocity for east 10 degrees: 2 rev, acc=0.01 . . . . . . . . . 82
4.5 Tangential velocity for east 10 degrees: 2 rev, acc=0.01 . . . . . . . 82
4.6 Control for east 10 degrees: 2 rev, acc=0.01 . . . . . . . . . . . . . 83
4.7 Trajectory for west 10 degrees: 2 rev, acc=0.01 . . . . . . . . . . . 86
4.8 Radius for west 10 degrees: 2 rev, acc=0.01 . . . . . . . . . . . . . 86
4.9 Radial velocity for west 10 degrees: 2 rev, acc=0.01 . . . . . . . . . 87
4.10 Tangential velocity for west 10 degrees: 2 rev, acc=0.01 . . . . . . . 87
4.11 Trajectory for west 10 degrees: 2 rev, acc=0.01 . . . . . . . . . . . 88
4.12 Radius for 180 degrees: 6 rev, acc=0.01 . . . . . . . . . . . . . . . 92
4.13 Radius for east 180 degrees: 4 rev, acc=0.01 . . . . . . . . . . . . . 92
4.14 Trajectory for 180 degrees: 6 rev, acc=0.01 . . . . . . . . . . . . . 93
4.15 Trajectory for 180 degrees: 4 rev, acc=0.01 . . . . . . . . . . . . . 93
4.16 Radial velocity for 180 degrees: 6 rev, acc=0.01 . . . . . . . . . . . 94
4.17 Radial velocity for 180 degrees: 4 rev, acc=0.01 . . . . . . . . . . . 94
4.18 Tangential velocity for 180 degrees: 6 rev, acc=0.01 . . . . . . . . . 95
4.19 Tangential velocity for 180 degrees: 4 rev, acc=0.01 . . . . . . . . . 95
4.20 Control for 180 degrees: 6 rev, acc=0.01 . . . . . . . . . . . . . . . 96
4.21 Control for 180 degrees: 4 rev, acc=0.01 . . . . . . . . . . . . . . . 96
4.22 Energy for 180 degrees: 6 rev, acc=0.01 . . . . . . . . . . . . . . . 97
4.23 Energy for 180 degrees: 4 rev, acc=0.01 . . . . . . . . . . . . . . . 97
List of Tables

1.1 Two-impulse phasing orbit for west 80 deg. maneuver 8
1.2 Delta V (km/s) with respect to the number of revolutions for various west-direction station changes 8

2.1 Initial guess, transversality condition and tolerance for the TPBVP 33
2.2 Representative value of thrust, exhaust velocity, specific impulse and mass rate (From Ref. [1], p 105) 38

3.1 LION-XI system specifications 52
3.2 Given initial data for the orbit raising problem 57
3.3 Non-dimensional data for the orbit raising problem 58
3.4 The number of variables and nonlinear constraints for the orbit raising problem 59
3.5 The number of variables and nonlinear constraints for maximum energy at final time 64

4.1 Bounds for states and control (non-dimensional units) 78
4.2 East 10 degrees input conditions 79
4.3 Number of variables and CPU time for east 10 deg. 79
4.4 West 10 degrees conditions 84
4.5 Number of variables and CPU time for west 10 degrees 85
4.6 East 180 degrees input conditions 89
4.7 Number of variables and CPU time for 180 degrees 89
4.8 Time of flight for East 180 degree 6 revolution cases with varying initial acceleration 91
List of Symbols

\( r \) radius
\( \theta \) reference angle
\( u \) radial velocity
\( v \) tangential velocity
\( \dot{r} \) radial velocity
\( \dot{\theta} \) angular rate of spacecraft
\( \dot{u} \) radial acceleration
\( \dot{v} \) tangential acceleration
\( x \) state vector
\( \dot{x} \) derivative of state vector
\( \phi \) control (thrust) angle from flight-path angle to force vector
\( x^* \) optimal state vector
\( u^* \) optimal control (thrust) vector
\( f(x, u, t) \) state vector derivative
\( \mathcal{H} \) Hamiltonian
\( t \) time
\( \omega_\oplus \) Earth rotation rate
\( \lambda \) Lagrange multiplier vector
\( c \) boundary condition
\( J \) performance index
\( J_a \) augmented performance index
\( e \) eccentricity
\( M \) mean anomaly
\( E \) eccentric anomaly
\( r_{geo} \) radius of geostationary orbit
\( a \) semi-major axis
\( \delta \) variation
\( \nu \) KKT constant multiplier
ζ   defect
µ   gravitational constant
F   force of thruster of a spacecraft
T   orbital period
I_{sp} specific impulse
v_e exhausted velocity
m mass of the spacecraft
\dot{m} fuel consumption rate
\tau computational time domain
N order for Legendre polynomial
J Jacobian matrix
f residual value vector
F force
\Delta x Variable displacement vector
w Legendre Gauss Lobatto weight
D_{ij} differentiation matrix

Subscripts
i variable number
0 initial value
f final value

Acronyms
GEO Geosynchronous Equatorial Orbit
GTO Geostationary Transfer Orbit
TPBVP Two-Point Boundary Value Problem
SQP Sequential Quadratic Programming
NLP Nonlinear Programming
Acknowledgments

I would like to acknowledge the advice and guidance of my advisor, Dr. Robert G. Melton, the chairman of dissertation committee. It would not have been possible to write this doctoral thesis without his patience and encouragement during the course of this research. His guidance and ever optimistic attitude have been an inspiration to me. I also thank to suggestions and advice of the members of my dissertation committee, Dr. David B. Spencer, Dr. Joseph F. Horn, and Dr. Julio A. Urbina, Special thanks to interest of Dr. George A. Lesieutre, head of the department aerospace engineering.

I would also like to express sincere appreciation to the R.O.K. Air Force offering me the opportunity to study abroad for the doctoral program. It would not have been possible to finish this study without the help, interest and unyielding support of all people around me.
Dedication
Chapter 1

Introduction

1.1 Geostationary Satellites

In the 1960’s, NASA began to extend satellite operations up to the altitude for geosynchronous orbits. By 1964, the communication satellite SYNCOM-3 was transmitting the Tokyo Olympic games live to stations in North America and Europe [2]. Many satellites have been placed in geosynchronous orbits since the first geosynchronous communication satellite, SYNCOM II, launched in the 1960s. A geosynchronous equatorial orbit (GEO) satellite is positioned approximately 35,786 km above the ground of the Earth (42,164.2 km radius from the center of the Earth) as shown in Figure (1.1). The gravitational acceleration from the Earth is 0.2242 m/s² at this altitude. This orbit must have a period of one sidereal day (23 hrs 56 min 4 sec) and zero inclination. The satellite moves around the Earth over the equatorial plane with a transverse velocity of 3.705 km/s, to remain synchronized with Earth’s rotation rate. That allows a satellite to stay continuously over one position on the ground. In fact, GEO is a theoretical orbit since in reality it is affected by small perturbation forces due to solar and lunar
gravity and Earth’s mass distribution.

The main advantage of a geostationary satellite is having continuous contact between the ground station and the satellite. GEO satellites can be used to provide constant communication systems and transmit atmospheric imagery and severe weather conditions such as floods, tsunamis, tornadoes, and hurricanes. In addition, they can perform various scientific purposes, and other special missions.

Figure 1.1. Geostationary Satellite
Note: Cited University Corporation for Atmospheric Research and NASA
Date of cite: June 25, 2012
http://www.rap.ucar.edu/~djohnson/satellite/coverage.html
http://www.nasa.gov/multimedia/imagegallery/image_feature_388.html
1.2 Station Change

The cost for placing a satellite in GEO is so expensive, several hundred million dollars for the satellites plus launch costs, that the owners must have back up procedures in case of failure of GEO satellite.

A satellite in GEO can be retasked and relocated via a phasing maneuver, which is also called a station change or relocation maneuver, to move from the primary position to a new longitude slot. This phasing maneuver is similar to a rendezvous problem except the terminal point is empty space and there is no docking process. A ground operator could expect to prepare the station change maneuvers for a GEO satellite several times over the entire mission lifetime.

There are several cases in which a station change must be performed. If a satellite failure occurs, an emergency station relocation maneuver would be required to replace a malfunctioning satellite and continue to maintain the operational capability. When a scientific test mission ends for one longitude slot, a relocation maneuver to another longitude position has to be accomplished by the operations staff. If space debris approaches the satellite, a temporary longitude change maneuver to get out of a potential collision course would be necessary to preserve satellite service life.

The operator should consider an appropriate action to let the system return to normal condition after a certain amount of time. In addition the satellite must be moved along a path that maintains a distance from other satellites in GEO. For these reasons, developing a scheme for station change is important.
1.3 Station-change Maneuvers

1.3.1 Station Keeping

An ideal geostationary mission, the orbit inclination and eccentricity are zero but the solar and lunar perturbations keep changing inclination and eccentricity. Since external forces are continuously affecting a GEO satellite, the operator has to keep maintaining a satellite inside a designated latitude and longitude rectangular box that is called the deadband [2]. North-south station keeping is the control for inclination and east-west station keeping is the control for longitude drift. Within the deadband no control is required. Station keeping maneuvers compensate for long term perturbations and are applied when the satellite drifts to the edge of the box (the deadband) [3], [4], [5].

1.3.2 Two-Impulse Maneuver

In contrast, a station change maneuver is used to move a satellite out of the station keeping longitude dead-band. The longitude deadband is defined in the form of its midpoint $\lambda_m$ and its half-width $\delta \lambda$, which means that $\lambda$ shall be in the interval

$$\lambda_m - \delta \lambda \leq \lambda_m \leq \lambda_m + \delta \lambda \quad (1.1)$$

Typical range of $\delta \lambda$ is from a few times of 0.01° up to 1°. To relocate the satellite, it is necessary to change the spacecraft’s orbital energy by using any form of passive and active external forces such as difference of gravity gradient, solar radiation pressure, or installed propulsion thrusters. An optimal relocation maneuver plan should be considered. For the simplest case, consider a two-impulse Hohmann
transfer maneuver in which the phasing orbit period is predetermined by the value of semi-major axis needed to increases or decrease an orbital period. Semi-major axis is given by

\[ a = \left( \frac{T \sqrt{\mu}}{2\pi} \right)^{\frac{2}{3}} \tag{1.2} \]

where \( T \) is the orbit period and \( \mu \) is the Earth’s gravitational constant. In order to relocate the satellite, an orbit’s size has to be changed to conduct a phasing maneuver. Firstly, to move a satellite from the current longitude at the geostationary orbit, it must begin to start drifting either in the east or west, depending on whether the transfer orbit is lower or higher than the geostationary orbit radius.

The longitude change maneuver is done by activation of thrust; it is assumed that small low-thrust rockets are placed on the side of the satellite, aligned in different directions to allow thrust to be applied in the desired direction based on the obtained optimal control vectors. Total \( \Delta V \), time of flight and semi-major axis are determined by setting the number of revolutions and by using Kepler’s equation \[ M = nt = \frac{2\pi}{T} t = E - esinE \tag{1.3} \]

where \( n \) is the mean motion, \( M \) is the mean anomaly and \( E \) is the eccentric anomaly. An east-west orbital maneuver involves applying thrust in the orbital plane, either with or against the direction of orbital motion. An east direction maneuver lowers the periapsis of the orbit by adding \( \Delta V \) in the opposite direction of the satellite’s motion, making the satellite lose energy. In contrast, a west direction maneuver speeds up the satellite to increase the apoapsis, increasing the orbit period. Two or more such maneuvers are used to change the longitude slot. When the satellite reaches the desired longitude position, an equal and opposite
burn must be performed, putting the satellite into the geostationary orbit again.

Figure (1.2) shows an example of a two-impulse station relocation maneuver with several conditions for the GEO satellite. This is the case for an 80° west-direction maneuver from 0° to 260° longitude. The biggest orbit is one revolution transfer maneuver. This Figure shows that as the number of revolution increases the transfer orbit size becomes smaller. Thrust is added to the east direction for increasing orbit size and decreasing an orbit period. Figures (1.3) and Figure (1.4) show the drift rate and total $\Delta V$ with respect to the number of revolutions. The drift rate is inversely proportional to the number of revolutions; and the required $\Delta V$ for an 80° longitude change becomes small as the number of revolutions increases.

Figure 1.2. Two-impulse geostationary orbit longitude change (80 deg.) west-direction for various number of revolutions
Figure (1.5) shows the semi-major axis for the corresponding transfer time. The $\Delta V$ profile with respect to transfer time for a different station change is shown in Figure (1.6). As can be seen in Figure (1.6), when the multiple revolutions are applied to the various station changes, the required $\Delta V$ for a large angle is not much different from a small angle station change compared to the a one-revolution maneuver.

From the results, when we specify the number of revolutions, it will affect the drift rate and total $\Delta V$ for the relocation maneuver. The energy needed for the transfer is dependent on the number of revolutions. Tabulated data shown in Table (1.1) indicates how much total $\Delta V$ is needed for each programmed maneuver. Table (1.2) shows the magnitude of delta $\Delta V$ for a different station change angle. From the table we can also determine that magnitude of semi-major axis would be a dominant factor for the number of revolutions, total $\Delta V$ and total time of flight. A key problem with this type of station change maneuver is that the satellite remains very close to the GEO belt for most of the transfer, raising the possibility of collision with the other satellites in GEO.

When drift angle and drift rate are within the required bounds, only the eccentricity needs to be changed. This change can be accomplished by a Hohmann-type transfer with two impulses. These two impulses are equal and the directions are opposite. The change in drift rate due to the first impulse is nullified by the second one of the opposite sign.
<table>
<thead>
<tr>
<th>Number of revolutions</th>
<th>Total $\Delta V$ [km/s]</th>
<th>Time of flight [day]</th>
<th>Semi-major axis [km]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3736</td>
<td>1.2189</td>
<td>48,199</td>
</tr>
<tr>
<td>2</td>
<td>0.2051</td>
<td>2.2161</td>
<td>45,232</td>
</tr>
<tr>
<td>3</td>
<td>0.1413</td>
<td>3.2134</td>
<td>44,221</td>
</tr>
<tr>
<td>5</td>
<td>0.0872</td>
<td>5.2079</td>
<td>43,404</td>
</tr>
<tr>
<td>10</td>
<td>0.0445</td>
<td>10.1942</td>
<td>42,786</td>
</tr>
<tr>
<td>15</td>
<td>0.0298</td>
<td>15.1805</td>
<td>42,579</td>
</tr>
<tr>
<td>20</td>
<td>0.0224</td>
<td>20.1668</td>
<td>42,476</td>
</tr>
<tr>
<td>30</td>
<td>0.0150</td>
<td>30.1394</td>
<td>42,372</td>
</tr>
</tbody>
</table>

**Table 1.1.** Two-impulse phasing orbit for west 80 deg. maneuver

<table>
<thead>
<tr>
<th>N. of rev.</th>
<th>20°</th>
<th>30°</th>
<th>40°</th>
<th>50°</th>
<th>60°</th>
<th>70°</th>
<th>80°</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.10782</td>
<td>0.15766</td>
<td>0.20505</td>
<td>0.25018</td>
<td>0.29321</td>
<td>0.33429</td>
<td>0.3736</td>
</tr>
<tr>
<td>2</td>
<td>0.05532</td>
<td>0.08192</td>
<td>0.10782</td>
<td>0.13306</td>
<td>0.15766</td>
<td>0.18165</td>
<td>0.2051</td>
</tr>
<tr>
<td>3</td>
<td>0.03719</td>
<td>0.05532</td>
<td>0.07313</td>
<td>0.09063</td>
<td>0.10782</td>
<td>0.12472</td>
<td>0.1413</td>
</tr>
<tr>
<td>5</td>
<td>0.02244</td>
<td>0.03352</td>
<td>0.04448</td>
<td>0.05532</td>
<td>0.06604</td>
<td>0.07665</td>
<td>0.0872</td>
</tr>
<tr>
<td>10</td>
<td>0.01124</td>
<td>0.01686</td>
<td>0.02244</td>
<td>0.028</td>
<td>0.03352</td>
<td>0.03901</td>
<td>0.0445</td>
</tr>
<tr>
<td>15</td>
<td>0.00748</td>
<td>0.01124</td>
<td>0.01499</td>
<td>0.01872</td>
<td>0.02244</td>
<td>0.02615</td>
<td>0.0298</td>
</tr>
<tr>
<td>20</td>
<td>0.00559</td>
<td>0.00842</td>
<td>0.01124</td>
<td>0.01405</td>
<td>0.01686</td>
<td>0.01965</td>
<td>0.0224</td>
</tr>
<tr>
<td>30</td>
<td>0.00371</td>
<td>0.00559</td>
<td>0.00748</td>
<td>0.00936</td>
<td>0.01124</td>
<td>0.01312</td>
<td>0.015</td>
</tr>
</tbody>
</table>

**Table 1.2.** Delta $V$ (km/s) with respect to the number of revolutions for various west-direction station changes
Figure 1.3. Two-impulse drift rate

Figure 1.4. Two-impulse total $\Delta V$
Figure 1.5. Semi-major axis with respect to transfer time for west 80-degree station change

Figure 1.6. Delta V (km/s) profile with respect to the number of revolutions for various station changes
1.4 Literature Review

Edelbaum [7] suggested that continuous low-thrust can be employed for the circle-to-circle transfer; the optimal method is to fire tangentially in one direction until half of the desired change is completed and then thrust in the opposite direction until the full longitude change is achieved. In his solution, the mass flow is negligible and he did not consider the eccentricity changes generated by tangential thrust.

Landis and Hrach proposed the satellite relocation by implementing a tether deployment [8]. A tether is a long flexible cable which connects one of a satellite with another. The tether was applied to achieve orbital longitude change without fuel expenditure using only a mass extension performed by a spool motor powered by the satellite’s solar array. The masses are placed at the end of the tether. The tether extension outward increases centrifugal force \((r\dot{\theta}^2)\) linearly with distance; on the other hand, if the mass that extends inward experiences, the center of mass of the orbit is pulled inward due to an increase in gravity that increases faster than linearly \((-\frac{\mu}{r^2})\). The tether can be used for changing the orbital eccentricity but energy is not conserved; angular momentum is conserved. The system also allows correction of small changes in orbital period and eccentricity by changing the semi-major axis. Maximum relocation can be obtained when the masses are equal at both ends. In reality, however, the length of tether they assumed is too long to be practical. In order to obtain an angular rate of 0.22 deg/day, a 1000 km long tether is needed.

Porcelli considered fast relocation for geostationary longitude relocation as well as accuracy of positioning [9], [10]. He presented an algorithm generating the tra-
jectory control requirements for the low-thrust maneuver: the total time and the required velocity increase. The two-point boundary value problem is solved by an iterative routine using forward and backward integration. He demonstrated the \( \Delta v \) requirement for given values of the relocation angle and initial acceleration. As the initial acceleration increases, the relocation angle also increases proportionally for the same amount of time.

Titus [11], [12] presented optimal-control techniques for the problem of the relocation in GEO satellites with constant low-thrust; and derived the converted \( \Delta V \) expression into simple analytical forms of fuel consumption, \( \Delta m \), for both two-impulse and continuous low-thrust burn cases. The analytical form of \( \Delta m \) indicates that high specific impulse propulsion can be dominant for the fuel consumption. He showed that constant low-thrust employing an electric propulsion system is superior for making sure near-zero final eccentricity is achieved (a constraint for the final position). For the advantage of less computational time, the objective function is the integral form of the satellite tangential rate, maximizing the total station relocation over the total time maneuver with a fixed final position. However, the cost function with a maximum angular rate for a given time is somewhat impractical in comparison to the final time-free problem from the perspective of minimum fuel expenditure. Generally, the first priority for the ground operator in station relocation is minimum fuel consumption for the transfer maneuver to a desired longitude position.

Free and Babuska [13] studied the propellant usage for station relocation, high altitude orbit raising, and north-south station keeping. In that paper a proper application could be chosen based on ratio of power to satellite mass. The angular
velocity rate formula is derived for total longitude change when using chemical propulsion. They suggested that multiple electric propulsion systems with high specific impulse can give several advantages to enhance relocation maneuvering and other maneuvers.

1.5 Dissertation Objectives

There are several optimal control software tools, DIDO[14], GPOPS [15], PSOPT [16], SOCS [17], BNDSCO [18], and DIRCOL[19] to obtain optimal solutions in aerospace engineering problems area such as interplanetary transfer, attitude control, launching, re-entry and autonomous flight problems.

DIDO is well known MATLAB-based optimal control software using a pseudospectral method associated with the SNOPT [20] nonlinear programming software, developed by Ross et al. While DIDO is recognized for its ability to generate high-fidelity numerical solutions (along with a reconstructed Hamiltonian and co-states), it is often difficult to use. A weakness is that DIDO does not indicate progress towards convergence, and it provides almost no useful diagnostic messages. The other drawbacks are that DIDO does not give an opportunity to control NLP solver, i.e. none of the user supplied derivative function is used. The user has no authority to adjust the optimality tolerances and iteration limits.

The main goal of this dissertation is to study the GEO station change problem with a constraint to avoid collisions. Previous research on geostationary station change problems used an indirect method, two-point boundary value problems, limited cases for near geostationary belt transfer[9] and time-fixed conditions[12].
Novel software for high performance optimal control for GEO satellite station change maneuvers is developed to obtain the control history and trajectories. The implementation of this research is associated with pseudospectral methods, one of the direct collocation methods. In order to enhance user control, the pseudospectral method algorithm is developed and the results are demonstrated via the Legendre pseudospectral method using global discretization.

In this research, the solution of the optimal problem, a set of trajectories and control vectors, can be evaluated depending on the desired transfer mission without the need to accurately guess, the Lagrange coefficients (co-state variables). A low-thrust acceleration is employed via an electric propulsion system, requiring a spacecraft to make multiple-revolution maneuvers with lower fuel expenditure.

1.6 Thesis Outline

In Chapter 2, the basic optimal control and conventional methods (calculus of variations) are introduced. The direct collocation techniques, discretizing time domain to approximate a differential equation to solve optimal control problems, are presented. In addition, a brief explanation for the efficiency of low-thrust with of the high specific impulse is discussed.

In Chapter 3, the pseudospectral methods and the mathematical formulation for a Legendre pseudospectral method in optimal control problems are introduced. The accuracy of the derivative function is described as a feature of the differentiation matrix. The developed Legendre pseudospectral algorithm is demonstrated to verify for some example problems.
In Chapter 4, the motion of a geostationary satellite is defined with some assumptions. The collision avoidance term is included in a performance index as an integral form. The integral term of the performance index is evaluated by the Gauss quadrature rule by using the Legendre-Gauss-Lobatto weights. The control and state histories are shown for the east and west-direction longitude change maneuvers. The comparison of energy variations for multiple revolutions is presented; and the thrust magnitude effect to the total transfer time is also considered.

Conclusions of this dissertation and recommendations for future study are presented in Chapter 5.
This page is blank intentionally
Chapter 2

Optimal Control Theory

The brachistochrone problem, determining the minimum time path for a bead sliding on a frictionless wire between two points under the constant-gravity condition, is well known as the first optimal control problem [21]. This problem can be solved by the classical method, using the calculus of variations with the Euler-Lagrange equations, for obtaining optimal solutions [22]. The classical optimization techniques have drawbacks due to some of the practical problems with performance indices that are not continuous or differentiable.

2.1 Optimal Control Applications

The optimal control problem is applicable in a wide variety of subject areas, ranging from automatic control systems and keeping operational cost low during the mission lifetime for aerospace engineering and automatic control [23],[24],[25],[26],[27] to pharmaceutical delivery systems for killing fast-growing cancer cells or strengthening the immune system of the human body [28].
Trajectory optimization is simply defined as finding the best way one can take where there are many selective potential cases leading a path to the solution while satisfying the equality and inequality constraints. In general, one wishes to find the maximum performance or minimum expense. Approximate solutions for a complex optimization can be obtained by numerical methods. For this reason iterative numerical methods are used to generate a series of progressively improved solutions to the optimization problem without an initial estimate of a set of trajectories. After several decades of innovations, many numerical methods have been developed with benefits from advanced computer technologies and increased computational speed. In this chapter we briefly review several methods related in the field of trajectory optimization.

2.2 The Optimal Control Problem

An optimal control problem may be formulated as an extended form of the nonlinear programming (NLP) problem with an infinite number of variables. Nonlinear programming is often called parameter optimization. In modern control theory, the optimal control problem is to find a control that causes the dynamical system to reach a target or follow a state variable (or trajectory) and at the same time maximize a performance index which may take several forms [22]. For solving a nonlinear problem, one should find a set of variables such that the performance index (also called cost the function or objective function), typically a nonlinear form, is optimized while satisfying a set of dynamic constraints. The form of an optimal control problem is similar to the functional extremum problem.
2.2.1 Mathematical formulation

The optimal trajectory of a spacecraft’s motion can be determined by the calculus of variations using Hamilton-Lagrange theory. The optimal control problem is to find an optimal $m$-dimensional control vector $u^*$ for a nonlinear system of differential equations. The dynamic state equations are described by $n$ first-order differential equations

$$\dot{x} = f(x, u, t), \quad t \in [t_0, t_f] \quad (2.1)$$

where $x$ is a vector of states, $u$ is a vector of controls, and $t$ is time. The initial and final boundary conditions are given by

$$\psi_0(x(t_0), t_0)^T = 0 \quad (2.2)$$

$$\psi_f(x(t_f), t_f)^T = 0 \quad (2.3)$$

where $\psi$ is a set of algebraic functions representing the states at the initial and final times. The dynamic system of equations is to follow an admissible trajectory of an $n$-dimensional state $x^*(t)$ such that the associated performance index is minimized [21]. Equation (2.1) is an equality constraint, generally nonlinear, and it should be satisfied over the entire time domain. In contrast, Eq. (2.2) and Eq. (2.3) are discrete because boundary conditions are imposed at the specific times for the initial and final points.

The first step for setting up the optimal control problem is to describe a performance index to be maximized or minimized depending on the problem conditions. The performance index may consist of two terms, an algebraic function of the final state and time, and an integral form of the states and controls.
\[
min \ J = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} \mathcal{L}(x, u, t) dt
\]  \hspace{1cm} (2.4)

Once we get the optimal solution, the control vector, \((u^*)\), will generate the optimal state vectors \(x^*\) while minimizing \(J\). In spacecraft problems, minimum time, fuel consumption or maximum energy at the final time would be taken as a scalar performance index. If the performance index contains the first term of Eq. (2.4) only, it is called the Mayer type. On the other hand, if it has only an integral term, it is called the Lagrange type. Equation (2.4) represents the Bolza form containing a final cost function and general cost function.

Using Lagrange multipliers, adjoining the dynamic equations and boundary conditions to the performance index, the augmented performance index can be written

\[
J_a = \phi[x(t_f), t] + \nu^T \psi[x(t_f), t]
\]
\[
+ \int_{t_0}^{t_f} [\mathcal{L}[x(t), u(t), t] + \lambda^T \{f[x(t), u(t), t] - \dot{x}\}] dt
\]  \hspace{1cm} (2.5)

The first-order necessary condition can be derived by applying the variational method as
\[\delta J_a = \frac{\partial \phi}{\partial x(t_f)} \delta x_f + \frac{\partial \phi}{\partial t_f} \delta t_f + \delta \nu^T \psi + \nu^T \frac{\partial \psi}{\partial x(t_f)} \delta x_f + \nu^T \frac{\partial \psi}{\partial t_f} \delta t_f + (\mathcal{L} + \lambda^T (f - \dot{x}))|_{t=t_f} \delta t_f + (\mathcal{L} + \lambda^T (f - \dot{x}))|_{t=t_0} \delta t_0 \]

\[\delta J_a = \delta x_f + \delta \nu^T \psi + \nu^T \frac{\partial \psi}{\partial x(t_f)} \delta x_f + \nu^T \frac{\partial \psi}{\partial t_f} \delta t_f + (\mathcal{L} + \lambda^T (f - \dot{x}))|_{t=t_f} \delta t_f + (\mathcal{L} + \lambda^T (f - \dot{x}))|_{t=t_0} \delta t_0 \]

Rewriting the last term of the above equation

\[\int_{t_0}^{t_f} -\lambda^T \delta \dot{x} = -\lambda^T (t_f) \delta x(t_f) + \lambda^T (t_0) \delta x(t_0) + \int_{t_0}^{t_f} \lambda^T \delta x dt \]

When the final time is free, the final variation between the final state and the state can be expressed as follows

\[\delta x_f = x(t_f) + \dot{x}(t_f) \delta t_f \]

The augmented performance index can be rewritten by substituting Eq. (2.7) and Eq. (2.8) into Eq. (2.6)
When the first variation $\delta J_a$ goes to zero, the extrema of the performance index $J$ is obtained. For solving this problem we can apply the Euler-Lagrange equation to the functional to determine the first-order necessary conditions by using the variation of the cost functional and forcing it to zero \cite{22} \cite{29} \cite{30}. The Hamiltonian is defined from the augmented performance index as

\begin{equation}
H(x(t), u(t), \lambda(t), t) = L(x, u, t) + \lambda^T f(x, u, t) \tag{2.10}
\end{equation}

The first-order necessary conditions for optimality can be defined by setting the variations $\delta x, \delta u, \delta \lambda$ equal to zero. The state, co-state, and control equations are solved along with the given initial and final conditions which lead to a two-point boundary value problem. Constraints for the final state variables can be expressed by final values of the Lagrange multipliers with what is called the transversality conditions. In summary, necessary conditions for the stationary value for $J$, can be described as

\begin{equation}
\dot{x}(t) = \frac{\partial H}{\partial \lambda} = f(x, u, t) \tag{2.11}
\end{equation}

\begin{equation}
\dot{\lambda}(t) = -\frac{\partial H}{\partial x} = -(\frac{\partial f}{\partial x})^T \lambda - (\frac{\partial L}{\partial x})^T \tag{2.12}
\end{equation}

\begin{equation}
(\frac{\partial H}{\partial u})^T = -(\frac{\partial f}{\partial u})^T \lambda - (\frac{\partial L}{\partial u})^T = 0 \tag{2.13}
\end{equation}

\begin{equation}
\lambda^T(t_f) = (\frac{\partial \phi}{\partial x} + \nu^T \frac{\partial \psi}{\partial x})_{t=t_f} \tag{2.14}
\end{equation}

\begin{equation}
\psi[x(t_f), t_f] = 0 \tag{2.15}
\end{equation}

The $m$-control vector $u(t)$ is determined by stationary conditions Eq. (2.13). For solving the two point boundary value problem, we need $n$ state equations,
Eq. (2.11), \( n \) co-state equations and Eq. (2.12) and boundary conditions.

### 2.3 Trajectory Optimization

In optimal control problems, one can obtain the analytical solutions only for limited cases [31]. Thus we must determine an iterative way to get an approximate solution for finite-thrust spacecraft trajectories. The two major categories for solving optimal control problems are known as the direct and an indirect method [32], [33].

#### 2.3.1 Indirect Method

An indirect method for solving optimal control problem is based on the calculus of variations and Pontryagin’s maximum principle [21]. The maximum principle states that the control variable has to be chosen to optimize the Hamiltonian subject to limitations on the control imposed by state and control path constraints. Indirect methods employ the first-order necessary optimality conditions obtained from the calculus of variations. The optimality condition and co-state equations result from Euler-Lagrange theory. The finite dimensional optimization problem is converted into a \( n \)-dimensional discrete optimization problem for the unknown co-states. In finite-dimensional case the Lagrange multipliers can be obtained from algebraic equations.

\[
\dot{\lambda} = -H_x^T
\]  

(2.16)

When we use an indirect method, the control variables are obtained indirectly through the associated two-point boundary value problem (TPBVP) which is made
of the state and co-state equations together with the initial and final conditions. In the TPBVP, one should determine the dependent variables such that they have specified values at two or more points. Thus, this approach is called the indirect method, because the optimal control is found by solving the auxiliary co-state equations, instead of using the original equations of motion. However, it is well known that there are some difficulties in implementing an indirect method:

- It is necessary to construct explicitly the adjoint equations, control equations and all of the transversality (boundary) conditions at the initial and final time.

\[-\frac{\partial H}{\partial x}, \frac{\partial H}{\partial u}\]

\[\lambda(t_i) = 0, \quad \lambda(t_f) = \frac{\partial \phi^T}{\partial x}\]

- There is a small region of convergence: the user must make a good initial guess for the adjoint variables, \(\lambda\). These are non-physical parameters, and poor guess can easily lead to ill-conditioning.

Using an indirect method can provide highly accurate results but solving the associated TPBVP using gradient and shooting methods can be difficult because of the aforementioned reasons. Many of TPBVPs are difficult to solve by gradient and shooting methods because of sensitivity, but this approach gives more accurate results [?].
2.3.2 Direct Methods

In direct methods, unknown state and control variables can be solved directly; a direct method constructs a sequence of trajectories \( x_1, x_2, \ldots, x^* \) subject to minimizing the performance index. The governing equations of motion are integrated implicitly. It requires mathematical programming to approximate the continuous problem with a parameter optimization problem, but it does not require any explicit derivation of the necessary and transversality conditions [32]. In order to avoid the sensitivity problem for indirect methods, several direct methods have been proposed including a hybrid formulation to solve transfer problems [34], [35]. The goal of any direct method for solving the optimal control problem is to approximate the continuous problem with a parameter optimization problem. Since the problem state is considered only at discrete nodes, the state history must be integrated between these nodes using some form of quadrature rule. Direct solution methods are categorized according to the type of integration scheme they use.

2.3.3 Collocation

There have been many attempts to substitute a continuous optimal control problem into a discretization to obtain a finite solution by implementing numerical techniques. Collocation techniques have been introduced and developed to solve TPBVPs by using polynomials as approximations for the implicit solution which is usually based on equally-spaced nodes in the time domain.

Reddien [36] used collocation at Gauss points and showed increased convergence rate compared to splines and Ritz methods [37], [27]. Such discretization
showed that it is possible to obtain high-order accurate discretization of certain unconstrained optimal control problems. Hargraves et al. [38] introduced the state and control histories using Chebyshev polynomials and used integral penalty functions to enforce equations of motion, then converted the optimal control problem into an unconstrained minimization. Approximating the states using third-order Hermite polynomial functions was proposed for solutions to the differential equations of TPBVPs to an optimal control problem by Dickmanns and Well [39].

In late 1980’s, Hargraves and Paris developed a collocation technique that converts optimal control problems into nonlinear programming problems [40]. Both control and state variables are regarded as independent variables in forming the problem; and the state history between nodes (discrete times at which the controls and states are determined) are approximated using a cubic polynomial fitted between neighboring nodes. The control history is assumed to be linear between nodes. This direct implicit method for converting an optimal control problem into a parameter optimization problem is commonly called Direct Collocation with Nonlinear Programming (DCNLP) [41].

2.4 Collocation Formulation

In collocation methods, the time domain is commonly divided into several segments and we choose two end points at each segment to formulate a polynomial function to derive an approximation to the center point derivative. The state and control values are calculated at discrete times (nodes), and piecewise-continuous polynomials are used to provide implicit solutions to the governing differential equations.
There are several different methods to discretize the problem and to ensure satisfaction of the system governing equations. Using the finite difference method, the equations of motion are defined as nonlinear dynamic constraints:

- Dynamic equality constraints

\[
c(x) = [D_1, D_2, ... D_{n-1}, D_n]^T = 0
\]  

(2.17)

- Inequality constraints

\[
c_L \leq c(x) \leq c_U
\]

(2.18)

In this section, some of the collocation methods (also called transcription method) are introduced.

### 2.4.1 Basic Schemes

In this section, some basic methods are introduced to formulate an implicit numerical integration form to propagate discrete state values.

- **Euler Method**

  A set of variables approximation is defined

  \[
  X^T = \{x_1, x_2, \ldots, x_N, u_1, u_2, \ldots, u_N\}
  \]

  (2.19)

  and a differential equation is expressed as

  \[
  \frac{dx}{dt} = f(x)
  \]

  (2.20)

  For obtaining states vector, using known values \(x_i, t_i\) and \(t_{i+1}\) Euler’s method
is defined to approximate $x_{i+1}$ explicitly

$$x_{i+1} = x_i + \Delta t_i f(x_i) \tag{2.21}$$

From Eq. (2.21), the defect can be defined at $i$th node

$$\zeta_i = x_{i+1} - x_i - \Delta t_i f(x_i) \tag{2.22}$$

- **Trapezoidal Method**

The $x_{i+1}$ is to be calculated by using the trapezoidal rule and the differential equation Eq. (2.20) can be integrated implicitly

$$x_{i+1} = x_i + \frac{\Delta t_i}{2} [f(x_i) + f(x_{i+1})] \tag{2.23}$$

Similarly the constraint defect can be written

$$\zeta_i = x_{i+1} - x_i - \frac{\Delta t_i}{2} [f(x_i) + f(x_{i+1})] \tag{2.24}$$

### 2.4.2 Hermite Cubic Polynomial

The Hermite-cubic method used for direct trajectory optimization is introduced by Hargraves and Paris [40]. In this method, the time history is divided into segments, the $i$th segment having width $H$. A cubic polynomial interpolated value and the slope at the midpoint can be described as

$$x_c = -\frac{(x_i - x_{i+1})}{2} + \frac{H}{8} (f_i - f_{i+1}) \tag{2.25}$$

where $f_i$ is evaluated from the equation of motion, Eq. (2.1), at $x_i$
\[ \dot{x}_c = -\frac{3}{2H}(x_i - x_{i+1}) - \frac{1}{4}(f_i + f_{i+1}) \]  

\text{(2.26)}

For obtaining state values at the mid point, the residual equation at the center is to evaluate the defect

\[ \zeta_i = f(x_c, u_c, t_c) - \dot{x}_c = f(x_c, u_c, t_c) + \frac{3}{2H}(x_i - x_{i+1}) + \frac{1}{4}(f_i + f_{i+1}) \]

The system state and control variables are assumed known only at the left and right sides of each segment; and the governing equation of the dynamic systems is expressed in Eq. (2.1). Iterative calculation (nonlinear programming) is used to make the defect \( \zeta_i \) go to zero in each segment.

The four quantities, i.e., the states and their time rates of change at the two boundaries of the segment \((x_i, x_{i+1}, f_i, f_{i+1})\), determine uniquely a cubic polynomial approximation for the state within the segment. By forcing the slope of the cubic to equal the system time rate of change evaluated at the center of the segment, i.e., by requiring

\[ f_c = \dot{x}((t_i + t_{i+1})/2, u_c) \]  

\text{(2.27)}

a constraint is created on each state.

Implicit integration is performed to collocate the equations of motion at the center of the segment. Figure (2.1) shows that a Hermite cubic polynomials can be determined such that it matches the state values at the nodes, and its derivative matches the state derivative at the center point. Note that the time interval \( T \) is divided into a number of nodes over the entire trajectory and the number of
segments will be \(N - 1\). A segment has fixed length \(H\):

\[
H = \frac{t_f - t_0}{N - 1}
\]

### 2.4.3 Hermite-Simpson Method

Simpson’s rule is formulated by a quadratic integrand of \(f(x)\). Using Eq. (2.25), the center point state value is approximated to construct a Hermite-Simpson method. The Hermite-Simpson discretization is widely used in the direct collocation algorithm that is fourth order. The constraint at the center point of the Hermite-Simpson’s polynomial is

\[
\zeta_i = x_i - x_{i+1} - \frac{\Delta t_i}{6} [f(x_i) + 4f(x_c) + f(x_{i+1})]
\]  \hspace{1cm} (2.28)
where
\[ f(x_{i+1}) = f[x_{i+1}, u_{i+1}, t_i + \frac{\Delta t_i}{2}], \quad (2.29) \]
\[ x_{i+1} = \frac{1}{2}(x_{i+1} + x_i) + \frac{\Delta t_i}{8}(f_i - f_{i+1}) \quad (2.30) \]

The finite difference method approximates the function derivatives using local discretized data. For obtaining iterative solution, implicit integration methods require at a minimum an initial guess for the problem state at every node. The initial state guess most likely will not satisfy the problem’s differential constraint, and the degree of constraint violation between nodes is quantified as a set of residuals or defects which are driven to zero as part of the optimization process.

### 2.4.4 Solution For An Indirect and Direct Method

Consider the optimal control maximization problem [42]. The objective function is defined as

\[ \max \ J = \int_0^{t_f} (4x^2 - 2u^2) dt \quad (2.31) \]

subject to
\[ \dot{x} = -\frac{1}{2}x^2 + 3u \quad (2.32) \]

with initial condition \( x(0) = 1 \) and final time \( t_f = 1 \). The Hamiltonian is

\[ \mathcal{H} = L + \lambda^T f(x, u, t) = 4x^2 - 2u^2 + \lambda(-\frac{1}{2}x^2 + 3u) \quad (2.33) \]
Using the optimality and tranversality conditions

\[ 0 = \frac{\partial \mathcal{H}}{\partial u} = -4u + 3\lambda, \quad u = \frac{3\lambda}{4} \tag{2.34} \]

\[ \dot{\lambda} = -\frac{\partial \mathcal{H}}{\partial x} = -4 + \lambda x \tag{2.35} \]

\[ \lambda(t_f) = 0 \tag{2.36} \]

Using two differential equations Eq. (2.32) and Eq. (2.34) and the optimality condition, TPVBP solutions are obtained as shown in Figure (2.2). The initial guesses for the control and co-state are shown in Table (2.1) for the TPBVP solution. The iterative procedure is performed by forward and backward integration with the Runge-Kutta 4th order method.

Because of the small region of convergence, when the arbitrary initial co-state guess was chosen the result was far from the optimal solution. To compare the two methods, the direct method results (solved by the Legendre pseudospectral method introduced in the next chapter) are shown in Figure (2.3). The thirty one Legendre-Gauss-Lobatto (LGL) nodes are used for this problem and direct solution results are shown in Figure (2.4). From the figure, the direct results are almost identical to the TPBVP solution. When using the direct method, it is not necessary to derive the co-state equations and any arbitrary initial guess gives converged state and control variables.
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x(t_0)$</td>
<td>0</td>
</tr>
<tr>
<td>$\lambda(t_0)$</td>
<td>0</td>
</tr>
<tr>
<td>$\lambda(t_f)$</td>
<td>0</td>
</tr>
<tr>
<td>$u(0)$</td>
<td>0</td>
</tr>
<tr>
<td>tolerance $\Delta$</td>
<td>1.e-3</td>
</tr>
</tbody>
</table>

**Table 2.1.** Initial guess, transversality condition and tolerance for the TPBVP
Figure 2.2. State, control and co-state history from the TPBVP

Figure 2.3. State and control history from the direct method
2.5 Thrust for Optimal Trajectories

In spacecraft operations we need an optimal trajectory solution requiring minimum propellant expenditure for lengthening the spacecraft’s operational life. Thrust management should be taken into account in the trajectory plan by formulating a mathematical model. Thrust is expressed by the mass-flow rate and an effective exhaust velocity, $v_e$

$$ F = |\dot{m}|v_e $$  \hspace{1cm} (2.37)

Most thrusters have a constant flow rate. Attitude thrusters are generally fired in multiple pairs to create torque. In contrast, for changing semi-major axis, orbit plane, and other transfer maneuvers, a thruster is activated over a finite time
ranging from a few seconds to several hours. When the thruster is deactivated, the spacecraft moves on a coast arc governed only by gravity (assuming no perturbations). Transfer duration may be defined as fixed or free, depending on the mission objectives.

2.5.1 High-Thrust

When high-thrust propulsion is used, it is assumed that the thrust burns are performed over a short interval relative to the orbital period. Trajectory optimization for high-thrust propulsion can be expressed in a simple mathematical form. This burn process can be represented as one burn each at the beginning and the ending of the maneuver. After the first burn is performed, a spacecraft’s motion is governed by gravity alone during the coast arc. Thus numerical integration is not necessary to solve a high thrust problem, if we know the position and velocity at the end points.

The high-thrust systems are normally characterized by low specific impulse and high fuel consumption. The maneuvers of this system are optimized so as to minimize fuel consumption.

2.5.2 Low-Thrust

Low-thrust trajectory optimization problems have been considered for transfer maneuvers between a parking orbit and a mission orbit [43], [44], [45]. The main advantage of a low-thrust propulsion system is the payload efficiency. As opposed to high-thrust systems, the low-thrust systems tend to have high specific impulse and low fuel consumption. But the low-thrust system requires a large amount of thrusting time during the entire mission and it should be formulated as a contin-
uous system.

Another type of propulsion system is electric propulsion or ion propulsion. It employs an on-board electric field to accelerate ions that are subsequently neutralized before being ejected. The advantage is that more $\Delta V$ relative to launch mass can be obtained for missions of long duration because only a small amount of mass is ejected with a high velocity, typically $30 \text{ km/sec}$ . This is combined with the fact that the operations power is generally obtained from the solar cells instead of being carried on-board as chemicals [46].

In the early 1960's, Edelbaum [7] studied the problem of optimal low-thrust transfer between inclined circular orbits. He assumed that thrust is regarded as providing constant acceleration and a constant thrust vector at each revolution, and he linearized the Lagrange planetary equations of orbital motion for the circular orbit [47]. Edelbaum proposed that several impulses are necessary to obtain the minimum $\Delta V$ transfer [48].

2.5.3 Specific Impulse

When we consider the remaining life time of a satellite, an application for constant low thrust is very useful to perform this research. Low-thrust electric propulsion systems have been studied widely in station keeping, interplanetary missions and LEO to GEO transfer maneuvers [49],[50],[45],[51]. The rocket equation is

$$\frac{m_f}{m_0} = e^{-\frac{\Delta V}{v_e}}$$

(2.38)

where $v_e = I_{sp}g_0$ is the effective velocity, $m_f$ is the final time mass and $m_0$ is the initial mass. A low-thrust engine, having very high levels of specific impulse, does have much more fuel efficiency than conventional high-thrust engines. The
high-thrust systems are normally characterized by low specific impulse and high fuel consumption. Table (2.2) [1] shows the forces, the relations between specific impulse and effective exhaust velocity, and how much mass rate is needed for an appropriate thruster.

| Propulsion system               | F     | $v_e$   | $I_{sp}$ | $|\dot{m}|$ |
|---------------------------------|-------|---------|----------|-----------|
| Solid propellant booster        | 40 kN | 3,000 m/s | 300 s    | 1.3 kg/s  |
| Liquid propellant booster       | 400 N | 3,500 m/s | 300 s    | 130 g/s   |
| Station keeping thruster        | 10 N  | 3,500 m/s | 350 s    | 3 g/s     |
| Ion thruster                    | 200 mN| 25 km/s  | 2,500 s  | 0.8 mg/s  |

**Table 2.2.** Representative value of thrust, exhaust velocity, specific impulse and mass rate (From Ref. [1], p 105)

The commonly used hydrazine thrusters are often designed to produce forces on the order of 0.5 to 20 N. A typical arc jet can have a specific impulse of at least 1,000 s [12]. Ion thrusters can achieve very high exit velocities, and have typical specific impulses up to 4,000 s.
This page is blank intentionally
Chapter 3

Pseudospectral Methods

Up to this point we have discussed optimal control theory and the types of methodologies have been developed since collocation methods were introduced. A direct collocation introduced in the previous chapter uses equally spaced data points for collocation and derivative approximation. However, as the number on nodes is increased, equally spacing will lead to large error especially at the end points. The pseudospectral methods require that the differential equation be exactly satisfied at the collocation points [52]. By formulating high-order differentiation matrix and applying it over the Legendre Gauss Lobatto nodes, one can obtain higher accuracy by differentiating a higher order interpolating polynomial. In this chapter, we will introduce the features of pseudospectral methods, the spectral accuracy of this method and some example problems that were solved by Legendre pseudospectral methods.
3.1 Pseudospectral Methods in Optimal Control

Pseudospectral methods have been proposed as one of the approximation techniques for computational fluid mechanics in the 1980’s for solving ordinary and partial differential equations [53], [54]. The basis functions are assumed to be polynomials and integration is approximated by Gauss quadrature rule formulations. The principle of pseudospectral methods is the employment of properties of orthogonal polynomial expansions, which have been shown to be mathematically robust [55], [56].

Finite difference methods approximate the derivative of a function using local information, whereas pseudospectral methods use the whole domain (for example, the entire transfer time in a trajectory problem) to approximate the derivative functions. The functions are expanded in terms of interpolating polynomials so that the expansion coefficients are the values of the function at the node points. Differential constraints are imposed only at the nodes (Legendre-Gauss-Lobatto points) by way of a differentiation operator. In this sense, the pseudospectral methods are different from other collocation schemes where high-order numerical integration techniques are used to approximate differential constraints not only at the nodes but in between [57], [58].

Elnagar et al. introduced Legendre pseudospectral methods to solve optimal control problems in nonlinear dynamic systems and demonstrated spectral accuracy [59] and also presented an alternative form to solve the controlled Duffing oscillator [60], [61] using the Chebyshev second-derivative differentiation matrix. In this formulation, it is not necessary to solve a TPBVP or to find a closed-form expression for the necessary and sufficient optimality conditions.
Fahroo and Ross [62],[63] investigated the relationship between the co-state variables and the Lagrange multipliers used in the direct method. By applying the Karush-Kuhn-Tucker (KKT) theorem, they proved that the co-state is determined quite accurately at the LGL points by simply dividing the KKT multipliers by the LGL weights,

\[ \omega_k = \frac{2}{N(N+1)} \frac{1}{[L_N(t_k)]^2} \]  

(3.1)

To obtain a better solution for an optimal control problem, it is necessary to estimate accurate co-state values in order to solve the TPBVP. They also developed Chebyshev methods and knotting techniques for discontinuous problem such as rocket launch trajectories and feedback control for attitude control [64],[65],[66],[56],[66],[67].

Huntington [68] and Huntington et al. [69] compared the performance of three different pseudospectral methods (Legendre-Gauss, Legendre-Gauss-Radau, and Legendre-Gauss-Lobatto) and presented the state, co-state and control error; and convergence rate for the single state case.

### 3.2 Legendre Polynomials

#### 3.2.1 Orthogonal functions

To introduce orthogonal polynomials over the interval (a,b), we can define the scalar product of two functions \( f(x) \) and \( g(x) \) over a weight function \( w(x) \), which is greater than or equal to zero on (a,b), as

\[ \langle f | g \rangle \equiv \int_a^b w(x) f(x) g(x) dx \]  

(3.2)
The scalar product is a number and the two functions are said to be orthogonal if their scalar product is zero. Thus one can define an orthogonal set of functions \( \{p_0, p_1, \cdots, p_N\} \) on an interval \([a,b]\) with respect to the weight function \( w(x) \)

\[
\int_a^b w(x) p_i(x) p_j(x) dx = \begin{cases} 
0, & i \neq j \\
\beta_i > 0, & i = j
\end{cases}
\]

(3.3)

### 3.2.2 Legendre Polynomials

The Legendre polynomials are the eigenfunctions of a singular Sturm-Louville problem

\[
\frac{d}{dx} [(1 - x^2) L_N(x)]' + N(N + 1) L_N(x) = 0, \quad N \geq 0
\]

(3.4)

Let \( L_N(x) \) denote the Legendre polynomial of order \( N \)

\[
L_N(x) = \frac{1}{2^N N!} \frac{d^N}{dx^N} (x^2 - 1)^N
\]

(3.5)

Legendre polynomials are orthogonal over the interval \([-1, 1]\), with the weight function \( w(x) = 1 \).

### 3.3 Legendre Pseudospectral Collocation

The collocation of differential equations is performed at orthogonal collocation points. In this thesis, Legendre Gauss Lobatto (LGL) points are selected as the collocation points. The LGL nodes are theoretically the roots of the derivative of the \( Nth \) degree Legendre polynomial, including the end points. The Examples of Legendre polynomials are:
Legendre polynomials and LGL nodes are represented in Figure (3.1) and Figure (3.2), respectively. In Figure (3.2), we can see that the distribution of the nodes, i.e., the density of the LGL nodes at the ends is higher than at the middle.

When it comes to error estimation, Legendre pseudospectral approximation is well known to have a rate of convergence at the collocation points [52] faster than any power of $1/N$. For the $N$th degree of interpolation, the error increases exponentially in equally spaced nodes, which is called the Runge phenomenon. In contrast, when using LGL nodes for interpolation, the error decreases exponentially.
Figure 3.1. Legendre polynomial $L_N(\tau)$ for $N = 0, 1, 2, 3, 4$

Figure 3.2. LGL points $N = 10, 20, 40$
3.4 Mathematical Formulations

The pseudospectral methods are approximation techniques for the computation of the solutions to ordinary and partial differential equations. They are based on a polynomial expansion of the solution. The unknown function $f(x)$ is defined by approximating it with a truncated series expansion using a basis trial function $\phi_i(x)$

$$f(x) \approx f_i(x) = \sum_{i=0}^{N} a_i(x)\phi_i(x)$$

We can approximate the function $f(x)$ by solving for the spectral coefficients $a_i(x)$

In the Legendre pseudospectral method, the state is approximated using a basis function. Consider the following optimal control problem. Determine the state trajectories $x(\tau) \in \mathbb{R}^n$, and control, $u(\tau) \in \mathbb{R}^m$, that minimize the performance index. The problem in Eq.(2.4) is defined over the time interval $[\tau_0, \tau_f]$. The $\tau_f$ denotes the final time which may be free or fixed. We can convert the time domain using LGL points that lie in the new mapping domain [-1, 1]. Now we can employ following transformation to express the problem for $t \in [t_0, t_f] = [-1, 1]$

$$\tau = [(\tau_f - \tau_0)t + (\tau_f + \tau_0)]/2$$

The objective performance can be replaced by the new computational time domain

$$J = \phi(x(-1), \tau_0, x(1), \tau_f) + \frac{\tau_f - \tau_0}{2} \int_{-1}^{1} \mathcal{L}(\tau, u, t)dt$$
subject to the state dynamics and boundary conditions

\[ \dot{x}(t) = \frac{\tau_f - \tau_0}{2} f[x(\tau), u(\tau)], \quad \tau \in [\tau_0, \tau_f] \]  

(3.10)

\[ \psi_0[x(-1), \tau_0] = 0 \]  

(3.11)

\[ \psi_f[x(1), \tau_f] = 0 \]  

(3.12)

Let \( L_N(t) \) be the Lagrange polynomial of degree \( N \) over the interval \([-1 1]\). Thus we can use LGL nodes using relations Eq. (3.8) i.e., \( t_0 = -1 \) and \( t_f = 1 \) in the Legendre collocation approximation for Eq. (3.9) - Eq. (3.12). For \( 1 \leq l \leq N - 1 \), the points \( t_l \) are the zeros of the derivative of the Legendre polynomial, \( \dot{L}_N \). Now we can define the approximating \( N \text{th} \) degree polynomial function

\[ x(t) \approx x^N(t) = \sum_{l=0}^{N} x(t_l) \phi_l(t) \]  

(3.13)

\[ u(t) \approx u^N(t) = \sum_{l=0}^{N} u(t_l) \phi_l(t) \]  

(3.14)

where for \( t = 0, ..., N \)

\[ \phi_l(t) = \frac{(t^2 - 1) \dot{L}_N(t)}{(t - t_l) N(N + 1) L_N(t_l)} \]  

(3.15)

are Lagrange polynomials of order \( N \). The Lagrange polynomials of order \( N \) can be shown as

\[ \phi_l(t_k) = \delta_{lk} = \begin{cases} 0, & k \neq l \\ 1, & k = l \end{cases} \]  

(3.16)
From this relation of \( \phi_t \), states and control can be defined as

\[
\phi_t = \begin{pmatrix}
\phi^T(t) & \cdots & 0 \\
0 & \phi^T(t) & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & \phi^T(t)
\end{pmatrix}
\]

where \( \phi = [\phi_0(t) \quad \phi_1(t) \quad \cdots \quad \phi_N(t)] \)

\[
x^N(t_l) = x(t_l), \quad u^N(t_l) = u(t_l) \tag{3.17}
\]

We impose the conditions exactly at the LGL nodes \( t_k \). The derivatives of the state and control vectors are approximated by matrix multiplication forms as

\[
\dot{x}^N(t_k) = \sum_{l=0}^{N} D_{kl} x(t_l) \tag{3.18}
\]

\[
\dot{u}^N(t_k) = \sum_{l=0}^{N} D_{kl} u(t_l) \tag{3.19}
\]

where \( D_{kl} \) is the \((N + 1) \times (N + 1)\) differentiation matrix

\[
D_{kl} = \begin{cases}
\frac{L_n(t_k)}{L_N(t_l) t_k - t_l}, & k \neq l \\
\frac{-N(N+1)}{4}, & k = l = 0 \\
\frac{N(N+1)}{4}, & k = l = N \\
0, & \text{otherwise}
\end{cases} \tag{3.20}
\]

Now performance index in the pseudospectral domain performance index can
be replaced using the Gauss Lobatto integration form

\[ J = \phi(x(-1), \tau_0, x(1), \tau_f) + \frac{\tau_f - \tau_0}{2} \sum_{k=0}^{N} L(x_k, u_k) \omega_k \]  

(3.21)

subject to dynamic constraints and boundary conditions

\[ \frac{\tau_f - \tau_0}{2} \sum_{k=0}^{N} L(x_k, u_k) \omega_k - \sum_{l=0}^{N} D_{kl} x(t_l) = 0 \]  

(3.22)

\[ \psi_0 [x(-1), \tau_0] = 0 \]  

(3.23)

\[ \psi_f [x(1), \tau_f] = 0 \]  

(3.24)

From Eq (3.21) to Eq (3.24), the formulation of this problem becomes a simple to solve nonlinear program.

### 3.4.1 Differentiation Matrix

In the pseudospectral method, the derivatives of the state functions at the discretization points are easily calculated by multiplying the differentiation matrix as the coefficient states or control column matrix. The derivative functions \( f^N(\tau) \) can be expressed as

\[ \dot{f}(\tau_k) \approx \dot{f}^N(\tau_k) = \sum_{l=0}^{N} D_{kl} f(\tau_l) \]  

(3.25)

The derivation of the differentiation matrix is introduced in Appendix A.

\[ D_{kl} = \dot{\phi}(t_k) = \frac{L_{N-1}(t_k)}{(t_k - t_l)L_{N-1}(t_l)} \]  

(3.26)

The differential equations of the optimal control problem are defined by a set of algebraic equations. The use of the differentiation matrix, which is a key idea in
pseudospectral methods, has provided us indirect collocation methods for solving for states and co-states variable simultaneously [62].

Figure (3.3) and Figure (3.4) show the derivative function values at the LGL nodes for \( f(t) = \cos(2t^2) \) for \( N = 20 \) and \( N = 30 \), respectively. For \( N = 20 \), the maximum error is as high as order \( 10^{-8} \) and then the error decreases rapidly after leaving the first node. Figure (3.4) for \( N=30 \) shows that the differentiation error is near zero in most regions except at the two end points. This is the reason why the Legendre pseudospectral method is employed in this thesis. This method can give us high accuracy results without the need for more computational storage space for state variables which is required in the classical direct collocation methods.
Figure 3.3. Pseudospectral differentiation error order $N=20$, $f(t) = \cos(2t^2)$

Figure 3.4. Pseudospectral differentiation error order $N=30$, $f(t) = \cos(2t^2)$
3.5 Legendre Pseudospectral Method

After describing the polynomial approximation for states and control values at the Legendre Gauss Lobatto points, we can set up the equations to satisfy the constraints. In order to satisfy the equations of motion we impose the conditions at the LGL points, thus we can approximate the values of states and control values at the global domain. For solving an optimal control problem using Eq. (3.8) we can convert the physical domain into LGL nodes.

3.6 Legendre Pseudospectral Code Test Cases

3.6.1 Test Environment

The Legendre pseudospectral method has been implemented in FORTRAN-90 code combined with nonlinear optimizer code NPSOL [70]. The code was examined by solving several optimal control problems to determine if it worked properly. The simulations were performed using LION-XI cluster [71], one of the high-performance computing systems in The Pennsylvania State University. The system specifications are shown in Table (3.1)

| Server                  | Dell Power Edge 1950 |
|                        |                      |
| Processor              | Intel Xeon E5450 Quad-Core 3.0 Ghz |
| Num. of Processor Cores| 8                     |
| Memory                 | 32 Gb                |

*Table 3.1. LION-XI system specifications*
3.6.2 Minimization Cost Problem

The first example is a simple case for one state and one control which is introduced in reference [72]. Consider the following optimal control problem. Minimize the cost function

$$J = \int_0^{\tau_f} (x^2 + u^2)dt$$

subject to the differential constraint

$$\dot{x} = -x^3 + u$$

and boundary conditions

$$x(0) = 1, \quad x(t_f) = 1.5$$

This optimal control problem is simple to study using the Legendre pseudospectral code. The physical time domain is converted into LGL nodes and the integral form of the cost function is integrated by the Legendre Gauss Lobatto quadrature rule. The results are for the case of $t_f = 1, t_f = 5, t_f = 25$ using 41 LGL nodes. The state, control and co-state histories are shown in Figure (3.5) and Figure (3.6) and Figure (3.7) respectively. This results are in perfect agreement with those in the reference [72]. Note that all units in the figures are non-dimensional.
Figure 3.5. State for minimum cost problem
Figure 3.6. Control for minimum cost problem

Figure 3.7. Co-state vs. state for minimum cost problem
### 3.6.3 Maximum Radius Orbit Transfer Problem

Consider the problem, introduced in reference [21], of transferring a spacecraft from a given unit circular orbit to the largest possible circular orbit in a fixed time using a low-thrust rocket engine. This problem is a continuous low-thrust coplanar orbit transfer from Earth to Mars. The total trip time is 193 days. The goal to maximize the radius in a given time requires a negative sign in the cost function to make this a minimization problem.

\[ J = -r(t_f) \]  

(3.30)

The equations of motion for this problem are defined

\[
\begin{align*}
\dot{r} &= u \\
\dot{u} &= \frac{v^2}{r} - \frac{\mu}{r^2} + \frac{F}{m_0 - \dot{m} t} \sin \phi \\
\dot{v} &= -\frac{uv}{r} + \frac{\mu}{r} + \frac{F}{m_0 - \dot{m} t} \cos \phi
\end{align*}
\]  

(3.31a, 3.31b, 3.31c)

where \( r \) is the radius, \( u \) is the radial velocity, \( v \) is the tangential velocity, \( \mu \) is the gravitational constant of the attracting center, \( F \) is the thrust, \( \dot{m} \) is the constant fuel consumption rate, \( m \) is an initial spacecraft mass and \( \phi \) is the thrust angle direction measured from the local horizontal plane to the thrust vector. The initial and final boundary conditions are defined as:

\[
\begin{align*}
r(0) &= r_0, & r(t_f) &= \text{unknown} \\
u(0) &= u_0 = 0, & u(t_f) &= 0
\end{align*}
\]  

(3.32a, 3.32b)
\[ v(0) = \sqrt{\frac{\mu}{r(t_0)}}, \quad v(t_f) = \sqrt{\frac{\mu}{r(t_f)}} \] (3.32c)

The given initial mass, propellant flow rate, thrust and other constants applied in this problem are shown in Table (3.2)

<table>
<thead>
<tr>
<th>Given dimensional values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial mass [kg]</td>
</tr>
<tr>
<td>Propellant rate [kg/day]</td>
</tr>
<tr>
<td>Thrust [N]</td>
</tr>
<tr>
<td>Sun’s gravitational constant ([km^3/s^2])</td>
</tr>
<tr>
<td>Initial Earth’s circular radius [km]</td>
</tr>
</tbody>
</table>

Table 3.2. Given initial data for the orbit raising problem

Note that non-dimensional units for numerical computation are defined as:

- The non-dimensional acceleration unit:

\[ \frac{\mu}{r_0^2} \]

- The non-dimensional time unit:

\[ \sqrt{r_o^3/\mu} \]
• The non-dimensional total flight time:

\[ \frac{t_f}{\sqrt{r_0^3/\mu}} \]

• The non-dimensional thrust:

\[ \frac{F/m_0}{\mu/r_0^2} \]

• The non-dimensional constant of the Sun: \( \mu = 1 \)

• The non-dimensional initial distance: \( r_o = 1 \)

All data used are converted into non-dimensional parameters in order to avoid calculating large numerical data. The advantages of scaling for computation are introduced in Chapter 4.

<table>
<thead>
<tr>
<th>( m_0 )</th>
<th>Thrust</th>
<th>Final time</th>
<th>( \dot{m} )</th>
<th>( \mu )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>1.0145</td>
<td>3.32</td>
<td>-0.0749</td>
<td>1</td>
</tr>
</tbody>
</table>

**Table 3.3.** Non-dimensional data for the orbit raising problem

The number of LGL nodes is 61 so that the total number of variables is 254 (192 for the states, 61 for the control and 1 for the time). The size of variables is shown in Table (3.4). The final optimal value, maximum radius, is \( r(t_f) = 1.5256 \) which shows good agreement with reference [21]. Three state histories for radius, radial velocity, tangential velocity are presented in Figure (3.8), Figure (3.9) and Figure (3.10). In Figure (3.11), the control angle is seen to vary smoothly over the entire time (ignoring the change in angle definition at +/- 180 deg). The
trajectory propagation shown in Figure (3.12) is obtained from explicit integration using control angle results from the pseudospectral code. Explicit integration was performed using a Runge-Kutta 4th order method with the control angle linearly approximated between the nodes.

<table>
<thead>
<tr>
<th>Number of variables for vector space</th>
</tr>
</thead>
<tbody>
<tr>
<td>LGL nodes</td>
</tr>
<tr>
<td>States</td>
</tr>
<tr>
<td>Control</td>
</tr>
<tr>
<td>Time</td>
</tr>
<tr>
<td>Nonlinear constraints</td>
</tr>
</tbody>
</table>

**Table 3.4.** The number of variables and nonlinear constraints for the orbit raising problem
**Figure 3.8.** Radius history for maximum final radius problem

**Figure 3.9.** Radial velocity history for maximum final radius problem
Figure 3.10. Tangential velocity history for maximum final radius problem

Figure 3.11. Control angle history for maximum final radius problem
3.6.4 Maximum Energy for Final Time Orbit Raising

For the third example, an optimal orbit raising problem for maximum energy at a
given final time was examined.

\[ X = [r, \theta, u, v]^T \]  \hspace{1cm} (3.33)

The control variable is the thrust direction angle (the same as in the previous
example). Thus, the objective function is defined by a scalar function of final
parameters

\[ J = M(t_f) = -\left[ \frac{u(t_f)^2 + v(t_f)^2}{2} - \frac{1}{r(t_f)} \right] \]  \hspace{1cm} (3.34)
subject to the dynamic equations

\[
\begin{align*}
\frac{dr}{dt} & = u \quad (3.35a) \\
\frac{d\theta}{dt} & = v \quad (3.35b) \\
\frac{du}{dt} & = \frac{v^2}{r} - \frac{1}{r^2} + Acc(t)\sin\phi \quad (3.35c) \\
\frac{dv}{dt} & = -\frac{uv}{r} + Acc(t)\cos\phi \quad (3.35d)
\end{align*}
\]

where constant acceleration, \(Acc(t) = 0.01\), is applied. Before using the Legendre pseudospectral method, the time domain \([0, t_f]\) must be converted into the computational domain using the relations in Eq (3.8).

The numerical value for the objective function, \(J = -0.09515\), is attained using 30 LGL nodes. From Figure (3.13) and Figure (3.14), the optimal trajectory obtained from 30 LGL nodes shows good agreement the result from 64 LGL nodes. The number of variables and controls is shown in Table (3.5). The number of state and control variables is smaller than that used by the Herman's [57]. Thus the results show that the Legendre pseudospectral code applied in this problem performs well enough to predict optimal results using only a small number of unknowns. Accuracy can be improved by increasing the number of nodes used in the discretized problem.

The orbit raising trajectory result in XY coordinates is shown in Figure (3.15). The energy is increased gradually from the initial time and reaches the maximum value at the final time as shown in Figure (3.16). The state histories for radial velocity and tangential velocity are shown in Figure (3.17) and Figure (3.18) re-
respectively. In order to make the size of orbit larger and reach maximum energy, the radial velocity is increased gradually whereas the tangential velocity becomes smaller. The control angle behavior varies smoothly and the frequency becomes larger as the semi-major axis increases as shown in Figure (3.19). Since these optimal control problems examples being solved by implicit integration, it is recommended to perform explicit integration to verify that the states histories have good agreement with the implicit results. If that shows almost identical results within a specified tolerance, then the optimal solutions are assumed to be reasonable. Explicit trajectory for the 200 nodes result is represented in Figure (3.20) showing that the explicit integration trajectory is almost identical to the implicit result during the orbit raising maneuver.

<table>
<thead>
<tr>
<th>Number of variables for vector space</th>
</tr>
</thead>
<tbody>
<tr>
<td>LGL nodes</td>
</tr>
<tr>
<td>States</td>
</tr>
<tr>
<td>Control</td>
</tr>
<tr>
<td>Time</td>
</tr>
<tr>
<td>Nonlinear constraints</td>
</tr>
</tbody>
</table>

**Table 3.5.** The number of variables and nonlinear constraints for maximum energy at final time
Figure 3.13. Trajectories for maximum energy at final time for 30 LGL nodes

Figure 3.14. Trajectories for maximum energy at final time for 64 LGL nodes
Figure 3.15. Trajectories in XY coordinates for maximum energy at final time

Figure 3.16. Energy for maximum energy at final time
Figure 3.17. Radial velocity history for maximum energy at final time

Figure 3.18. Tangential velocity history for maximum energy at final time
Figure 3.19. Control angle for maximum energy at final time

Figure 3.20. Trajectories for maximum energy at final time, explicit integral results
3.7 Summary

In this chapter, we introduced the pseudospectral methods and the feature of this methods: the high accuracy of the derivative functions and orthogonality of the Legendre polynomials. The finite difference methods approximate the derivative functions using local information, in contrast, the pseudospectral methods are global in nature and use the entire domain to approximate the derivative function using the differentiation matrix. The developed algorithms of the Legendre pseudospectral method were evaluated for several problems; and the test results show good agreement with references [21], [57]. For the maximum energy at the final time problem, the efficiency of this algorithm can be found even though it used a smaller number of variables than that of the reference.
Chapter 4

Constrained Station Change for a Geostationary Satellite

In this chapter we determine the optimal control for a constrained station change in GEO. For solving station change problems, some assumptions have been made to define the motion of the spacecraft: the spacecraft is regarded as a point mass; perturbation effects due to solar radiation pressure, lunar and solar gravity, and magnetic forces are all ignored; the only additional force is the constant low thrust of the satellite’s engine.

4.1 Problem Statement

The geostationary satellite station change maneuver involves moving from a current longitude to a target longitude in a minimum time. Since the thrust will be constant, this implies that the problem is also a minimum propellant one. The longitude difference must be larger than the station keeping deadband. If a satellite is to be relocated, the repositioning maneuver will be performed using finite low-
thrust along the flight path. The motion of the body can be expressed by a system of second-order differential equations [47]

\[ \ddot{r} + \mu \frac{r}{r^3} = A \]  

(4.1)

where the radius \( r = \|r\| \) is the magnitude of the inertial position vector \( r \), and \( \mu \) is the gravitational constant. We can define \( A \) as a low-thrust acceleration

\[ A = \frac{F}{m_0 - \dot{m}t} \]

If the external force is zero, \( \|A\| = 0 \), Eq. (4.1) becomes a two-body problem. The general acceleration components representing radial and tangential direction in two dimensional motion can be described in polar form

\[ \ddot{r} = r\dot{\theta}^2 - \frac{\mu}{r^2} + A\sin\phi \]  

(4.2a)

\[ \ddot{\theta} = \frac{2\dot{r}\dot{\theta}}{r} + A\cos\phi \]  

(4.2b)

\[ \dot{m} = -c \]  

(4.2c)

The spacecraft has a fixed propellant mass-flow rate which is a negative value. A GEO satellite geometry for this problem is shown in Figure (4.1) and the longitude position of satellite is the angle from the reference line. The thrust angle, \( \phi \), is measured from the transverse direction at the spacecraft position and this angle is also called control angle in the optimal control problem. We can transform Eq. (4.2) into a set of first order ordinary different equations for variable thrust motion as follows
\[ \dot{r} = u \]  
(4.3a)

\[ \dot{\theta} = \frac{v}{r} \]  
(4.3b)

\[ \dot{u} = \frac{v^2}{r} - \frac{\mu}{r^2} + Asin\phi \]  
(4.3c)

\[ \dot{v} = -\frac{uv}{r} + \frac{\mu}{r} + Acos\phi \]  
(4.3d)

\[ \dot{A} = \frac{A^2}{v_e} \]  
(4.3e)

where \( r \) is the distance from Earth’s center, \( \theta \) the angle from the reference line, \( u \) the velocity along the radial direction, \( v \) the transverse velocity, \( A \) the thrust acceleration and \( v_e \) the effective velocity. In this thesis, a thrust acceleration is
represented as one of the state variables. For the case of constant thrust, the thrust acceleration varies with time because of the consumption of fuel.

### 4.2 Optimal Control Formulation

For a better approximation of a practical system using thrust propulsion, a constant thrust is chosen and the thrust acceleration varying with respect to time is expressed in terms of exhaust velocity, which depends on the specific impulse. The system dynamics are modeled using equations of motions Eq. (4.3) and then implemented in an optimal control formulation using the Legendre pseudospectral collocation method. The DCNLP method was attempted for this optimal control problem, but it did not converge. In satellite relocation problems, we can consider two choices as an objective function. One is the solution for the minimum time of flight with fixed longitude change

\[
J = t_f
\]  

(4.4)

and the other one is maximizing the angular rate [12] of the spacecraft with a given time.

\[
J = \int_0^{t_f} (\frac{v}{r} - \omega_\oplus) dt
\]  

(4.5)

From the user’s perspective, obtaining a minimum time solution is more practical than the fixed time problem. In addition, to ensure a clearance between the satellite and other satellites in the geostationary belt, the objective function can be reformulated by adding an integral term as follows:
\[ J = t_f + K \int_0^{t_f} [r(t) - R_{\text{min/max}}]^2 dt \]  

(4.6)

where \( R_{\text{min/max}} \) is an lower or upper radius to which the satellite must be moved during the transfer maneuver to avoid collisions with other satellites. For an east direction change \( R_{\text{min}} \) would be chosen; and \( R_{\text{max}} \) is for a west direction change. Note that this formulation treats the desired transfer radius \( R_{\text{min/max}} \) as a soft constraint, which allows the spacecraft to move to \( R_{\text{min/max}} \) in a finite amount of time. Instantaneous transfer from GEO radius to \( R_{\text{min/max}} \) is physically impossible.

The constant \( K \) is a weight factor to be chosen to represent the relative importance of the integral term. For computational purposes, the time \( \tau \in [0 \ \tau_f ] \) must be converted into the LGL domain i.e., \( t \in [-1 \ 1] \) using Eq. (3.8). Then Eq. (4.6) can be written as

\[ J = \tau_f + K \frac{\tau_f - \tau_0}{2} \int_{-1}^{1} [r(\tau) - R_{\text{min/max}}]^2 d\tau \]  

(4.7)

Now the performance index is formulated in Gauss quadrature form as

\[ J = \tau_f + K \frac{\tau_f - \tau_0}{2} \sum_{k=0}^{N} [r(t) - R_{\text{min/max}}]^2 \omega_k \]  

(4.8)

where the \( \omega_k \) are the LGL weights obtained from the LGL pseudospectral routine.

### 4.3 Nonlinear Programming Solver

With the equations of motion expressed as a set of discrete algebraic constraints, the optimal control problem has been converted into an NLP problem that mini-
mizes the performance index subject to the dynamic constraints. In this thesis the satellite transfer motion and associated NLP calculations are coded in FORTRAN, which calls the NPSOL [70] optimizer to solve the nonlinear programming problem.

An optimal solution to the NLP problem can be approximated by using the optimization algorithm NPSOL. NPSOL is a set of FORTRAN subroutines for minimizing a smooth function subject to constraints, which may include simple bounds on the variables, linear constraints and smooth nonlinear constraints. NPSOL uses a sequential quadratic programming algorithm, in which each search direction is the solution of a quadratic programming sub-problem.

In the process of optimal solutions, NPSOL needs an initial guess for the NLP parameters of states, control, and time. Further to this, a user must provide subroutines that evaluate the performance index, the algebraic constraint equations, and the first derivative with respect to time of the vector of unknowns (i.e. the Jacobian) for the cost function and constraints. If the user does not define the Jacobian, NPSOL will approximate it using a finite difference method.

4.4 Scaling

When solving NLP problems, it is important to use scaling so that all of the parameters have roughly the same order of magnitude. This reduces roundoff error and also improves convergence properties in the optimization process. The initial spacecraft mass is assumed to be 1,000 kg with an electric propulsion system. All numerical data used in this thesis are normalized by setting the gravitational parameter $\mu = 1$ and the distance unit $DU = r_{geo}$. This results in the following
units:

- Time unit
  \[ T_U = \frac{1 \text{ day}}{2\pi} \]

- Velocity unit
  \[ D_U/T_U = 3.0746619 \text{ km/s} \]

- Acceleration unit
  \[ \frac{\mu}{r_{\text{geo}}^2} \]

4.5 Multiple Revolutions

In a relocation mission, a multiple-revolution maneuver may be necessary to perform an optimal transfer between locations when low-thrust acceleration is employed. Collyer [73] proposed a non-integer transfer orbit for circular orbit phasing maneuvers using impulsive thrust, which leads to lower \( \Delta V \) expense. He introduced a discrete transfer method, inserting a burn to rotate the line of apsides during the last phasing maneuver to reduce the time of flight.

Wiesel and Alfano [74] showed a closed-form solution for many revolutions for a low-thrust, circle-to-circle orbit transfer. Alfano and Thorne [75] calculated a constant low-thrust orbit raising transfer, showing the relationship between initial acceleration and time of flight. In this research, multiple revolutions are applied to the station change maneuvers. The number of revolutions is specified for a given transfer.
4.6 Simulation Results

4.6.1 Constraints and Bounds

The station change maneuver starts the thrust with the initial acceleration prescribed depending on the level of initial thrust. If $2.24\, N$ acts on a $1,000\, kg$ satellite, the initial acceleration becomes $0.01\, [DU/TU^2]$. The initial acceleration input can be varied by changing the magnitude of the thrust. The bounds for the states and control are shown in Table (4.1).

<table>
<thead>
<tr>
<th>State /Control</th>
<th>bl</th>
<th>bu</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>0.9</td>
<td>1.1</td>
</tr>
<tr>
<td>$\theta$</td>
<td>0</td>
<td>Max</td>
</tr>
<tr>
<td>$V_r$</td>
<td>-2</td>
<td>2</td>
</tr>
<tr>
<td>$V_t$</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Acc</td>
<td>Initial acc</td>
<td>0.05</td>
</tr>
<tr>
<td>Control</td>
<td>$-\pi$</td>
<td>$\pi$</td>
</tr>
</tbody>
</table>

Table 4.1. Bounds for states and control (non-dimensional units)

4.6.2 East Two-Revolution

Consider the 10-degree east-direction relocation maneuver case. Figure (4.2) shows the trajectory for two revolutions with anti-collision weight factor $K = 10$ and $K = 0.01$. The trajectory history behavior indicates that lowering orbital energy reduces the orbit size close to the minimum radius bound, $r = 0.98$. Note that in
this figure (and in all other trajectory plots in this chapter) the radial scale is not zero at the center of the figure. The number of LGL nodes and input conditions are shown in Table (4.2). The calculation time and the step size are shown in Table (4.2).

---

**Input values for east direction maneuver**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LGL Nodes</td>
<td>41</td>
</tr>
<tr>
<td>Num. of rev.</td>
<td>2</td>
</tr>
<tr>
<td>Thrust [N]</td>
<td>2.24</td>
</tr>
<tr>
<td>Weight factor</td>
<td>10</td>
</tr>
<tr>
<td>$I_{sp}$ [sec]</td>
<td>1000</td>
</tr>
<tr>
<td>$R_{min}$</td>
<td>0.98</td>
</tr>
</tbody>
</table>

**Table 4.2.** East 10 degrees input conditions

---

**Number of variables and CPU time**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. of Max Variables</td>
<td>247</td>
</tr>
<tr>
<td>Num. of Constraints</td>
<td>206</td>
</tr>
<tr>
<td>CPU Time [min]</td>
<td>0.97656</td>
</tr>
<tr>
<td>Objective Value</td>
<td>12.14706</td>
</tr>
</tbody>
</table>

**Table 4.3.** Number of variables and CPU time for east 10 deg.

From Figure (4.3), the radius variation shows steeper behavior at the starting and end points of the maneuver. That is sufficient to avoid collision in the geostationary belt over most of the transfer maneuver for this case.
States for the velocity component are shown in Figure (4.4) and Figure (4.5). The radial velocity starts to decrease into negative regions at the beginning and then becomes near zero. It and then shows positive magnitude at the final stage of the maneuver. The tangential velocity does not enter the negative regions because transfer motion is always in the counterclockwise direction. The tangential velocity history begins with unit velocity at the starting point and ends with the same initial velocity to become a circular orbit.

The desired control angle is shown in Figure (4.6). Because the thrust magnitude is constant the control angle must change to prevent the semi-major axis from varying too much (which would result in excessive longitude change, or exceed the bound \( R_{\text{min/max}} \)).
Figure 4.2. Trajectory for east 10 degrees: 2 rev, acc=0.01

Figure 4.3. Radius for east 10 degrees: 2 rev, acc=0.01
Figure 4.4. Radial velocity for east 10 degrees: 2 rev, acc=0.01

Figure 4.5. Tangential velocity for east 10 degrees: 2 rev, acc=0.01
4.6.3 **West Two-Revolution**

We have seen the east-direction station change maneuver. The west-direction case begins by enlarging semi-major axis to increase the orbital period. The station change maneuver begins with initial conditions as shown in Table (4.4), and state and control variables are obtained. The size of state and control variables, calculation time for this transfer data are shown in Table (4.5).

Figure (4.7) and Figure (4.8) are the trajectory and radius histories showing maneuvers from the initial position to the outer region of the geostationary ring. The radial and tangential velocity components are shown in Figure (4.9) and Fig-
Input values for west-direction maneuver

<table>
<thead>
<tr>
<th>LGL Nodes</th>
<th>31</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. of rev.</td>
<td>2</td>
</tr>
<tr>
<td>Thrust [N]</td>
<td>2.24</td>
</tr>
<tr>
<td>Weight factor</td>
<td>1</td>
</tr>
<tr>
<td>$I_{sp}$ [sec]</td>
<td>1,000</td>
</tr>
<tr>
<td>$R_{max}$</td>
<td>1.1</td>
</tr>
</tbody>
</table>

Table 4.4. West 10 degrees conditions

Since this transfer is maneuvering outer region of GEO to increase the orbit size, the radial and tangential velocity variations are opposite at the beginning and end points of the maneuver compared to the east-direction transfer case. The control angle behavior shows that there are some constant values, $180^\circ$, during the second half of the transfer as shown Figure (4.11). Note that this case differs from the east-direction transfer in that the weight factor $\mathcal{K}$ is ten times less. The resulting trajectory for the west-direction case therefore does not approach or depart the maximum bound as rapidly as in the east-direction case.
<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num. of Max Variables</td>
<td>187</td>
</tr>
<tr>
<td>Num. of Constraints</td>
<td>156</td>
</tr>
<tr>
<td>CPU Time [min]</td>
<td>0.3324</td>
</tr>
<tr>
<td>Objective Value</td>
<td>13.55065</td>
</tr>
</tbody>
</table>

**Table 4.5.** Number of variables and CPU time for west 10 degrees
Figure 4.7. Trajectory for west 10 degrees: 2 rev, acc=0.01

Figure 4.8. Radius for west 10 degrees: 2 rev, acc=0.01
Figure 4.9. Radial velocity for west 10 degrees: 2 rev, acc=0.01

Figure 4.10. Tangential velocity for west 10 degrees: 2 rev, acc=0.01
4.6.4 East 180-degree Change

Consider the $180^\circ$ longitude change case. Six-revolution and four-revolution results are obtained to compare the transfer maneuver performance. We employed 31 LGL nodes and the initial input conditions are shown in Table (4.6). The number of state and control variables and the calculation times are shown in Table (4.7). Figure (4.12) and Figure (4.13) give the radius histories. The trajectory history is shown in Figure (4.14) and Figure (4.15). From the figures, we can see that the satellite moves near $R_{\min}$ most of the transfer time. Both cases show similar behavior except that the satellite arrives at $R_{\min}$ earlier in the six-revolution case.
The history of the radial velocities shows that the six-revolution case has a near zero velocity when it arrives at the minimum radius. The tangential velocity in a six-revolution case has a steady region between one and three days but the four-revolution case has an oscillating radial velocity as shown in Figure (4.18) and Figure (4.19).

---

**Input values for east direction maneuver**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>LGL Nodes</td>
<td>31</td>
</tr>
<tr>
<td>Num. of rev.</td>
<td>6 and 4</td>
</tr>
<tr>
<td>Thrust [N]</td>
<td>2.4</td>
</tr>
<tr>
<td>Weight factor</td>
<td>1</td>
</tr>
<tr>
<td>$I_{sp}$ [sec]</td>
<td>1,000</td>
</tr>
<tr>
<td>$R_{min}$</td>
<td>0.9</td>
</tr>
</tbody>
</table>

**Table 4.6.** East 180 degrees input conditions

---

**Number of variables and CPU time**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num of Max Variables</td>
<td>187</td>
</tr>
<tr>
<td>Num. Constraints</td>
<td>156</td>
</tr>
<tr>
<td>CPU Time(6rev)[min]</td>
<td>0.300804</td>
</tr>
<tr>
<td>CPU Time(4rev)[min]</td>
<td>0.3017207</td>
</tr>
</tbody>
</table>

**Table 4.7.** Number of variables and CPU time for 180 degrees

From the six-revolution case, shown in Figure (4.20), the control angle begins
with a negative value; and then becomes positive after crossing the switching point in the middle of the transfer. In contrast, the four-revolution control history lies at a positive angle except at the beginning of the maneuver as shown in Figure (4.21).

Now we consider the orbital energy variation for both cases shown in Figure (4.22) and Figure (4.23). The energy becomes diminished because of the shrinking semi-major axis; the energy decreases from -0.5 to below -0.56 for both cases. It can be seen that there is less energy loss for the six-revolution case in the first half of the transfer compared to the four-revolution case. That behavior is somewhat expected because the four-revolution case must complete the station change in less time, and therefore requires a greater change in semi-major axis (and orbital energy).

4.6.5 Effect of Thrust Magnitude on Transfer Time

In this problem we can examine the effects of different thrust magnitudes on the transfer time. The initial acceleration and time of flight relations are shown in Table (4.8). The results correspond to east 180-degree longitude change with six-revolution maneuver. Three cases were obtained by varying the initial acceleration from 0.0107 to 0.023. Because the station change maneuvers are time-free optimal control problem, as the initial acceleration increases the total time of flight decreases as expected. It is interesting to note in comparing the first and third cases that the thrust is doubled, but the transfer time decreases by only 2%. This is because the dynamics of the motion are dominated by the gravitational force.
<table>
<thead>
<tr>
<th>Initial acceleration $[DU/TU^2]$</th>
<th>Time of flight [day]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0107</td>
<td>5.66845</td>
</tr>
<tr>
<td>0.01388</td>
<td>5.63294</td>
</tr>
<tr>
<td>0.023</td>
<td>5.58586</td>
</tr>
</tbody>
</table>

Table 4.8. Time of flight for East 180 degree 6 revolution cases with varying initial acceleration
Figure 4.12. Radius for 180 degrees: 6 rev, acc=0.01

Figure 4.13. Radius for east 180 degrees: 4 rev, acc=0.01
Figure 4.14. Trajectory for 180 degrees: 6 rev, acc=0.01

Figure 4.15. Trajectory for 180 degrees: 4 rev, acc=0.01
**Figure 4.16.** Radial velocity for 180 degrees: 6 rev, acc=0.01

**Figure 4.17.** Radial velocity for 180 degrees: 4 rev, acc=0.01
Figure 4.18. Tangential velocity for 180 degrees: 6 rev, acc=0.01

Figure 4.19. Tangential velocity for 180 degrees: 4 rev, acc=0.01
Figure 4.20. Control for 180 degrees: 6 rev, acc=0.01

Figure 4.21. Control for 180 degrees: 4 rev, acc=0.01
Figure 4.22. Energy for 180 degrees: 6 rev, acc=0.01

Figure 4.23. Energy for 180 degrees: 4 rev, acc=0.01
4.7 Summary

In this chapter, we defined the equations of motion for a GEO satellite with some assumptions, i.e. a low-thrust engine of the satellite is the only applied force to the spacecraft and other perturbation forces are neglected. Before solving a constrained geostationary station change problem, the physical time domain is converted into the Legendre pseudospectral domain to collocate the first-order differential equation at the global LGL nodes using orthogonality conditions. The performance index is formulated by the summation of the minimum transfer time and an integral term as a path constraint to avoid collision with other satellites stationed in the geostationary ring. The maximum radius and minimum radius are specified by the user for each relocation maneuver. For obtaining an iterative solution, NPSOL is used to calculate the nonlinear program and it provides converged optimal control angle histories to predict states over the entire transfer time.

The optimal trajectory solutions are gathered for east-and-west-direction relocation maneuvers with some assumptions. The obtained control angle histories are feasible to keep the state trajectories in bound ranges to meet the constraints. The control and state histories for multiple-revolution are also obtained. It is found that the decrement of energy loss for four-revolution is larger than six-revolution case. The magnitude of acceleration effect is also tested and indicated that as the magnitude of acceleration increases, the total transfer time is decreased, however, it does not much affected for reducing time of flight because of the Earth gravity. A constrained geostationary station change using a Legendre pseudospectral method can predict the control and state trajectories with low-thrust in multiple-revolution maneuvers.
This page is blank intentionally
Conclusions and Recommendations for Future Work

This thesis has developed a method for determining optimal station-change maneuvers for satellites in geosynchronous equatorial orbits (GEO), with a constraint on the transfer radius to reduce the possibility of collisions with other satellites in GEO. Constant low-level thrust was assumed, with the in-plane thrust angle as the only control. The problem was formulated as a Bolza form in optimal control theory and solved using a Legendre pseudospectral method to calculate the control.

The resulting control histories and trajectories are feasible and within practical limits for implementation. Multiple-revolution station-change transfers are possible, which would allow very low thrust engines to be employed. This would make it possible to complete the station change by using a satellite’s attitude thrusters in the event that the orbital maneuvering system had failed.

An advantage of the pseudospectral method is that it can determine the op-
timal control using fewer unknown parameters (the states and controls at the node points) than other direct methods; however, it does not handle discontinuous controls very well. For example, the direct method DCNLP can easily handle a sequence of thrust and coast arcs during a transfer (but requires many more state and control values).

Future work on the low-thrust station-change problem should study the use of coast arcs as part of the solution, with a trade-off between reduced propellant usage and increased transfer time. The formulation used in this thesis treated the minimum/maximum radius (for avoiding collisions with other satellites in GEO) as part of the objective function, rather than as a rigid constraint. This resulted in smooth trajectories, since an instantaneous change in radius from GEO is not physically possible. But this formulation also resulted in some solutions that violated this minimum/maximum radius requirement. Future work should examine ways to include the constraint exactly over some specified portion of the transfer.
This page is blank intentionally
Differentiation Matrix for Legendre Pseudospectral Method

A.1 Differentiation Matrix

Note that polynomial approximation of \( x(t) \) is defined as
\[
x^N(\tau_i) = \sum_{j=1}^{N} x(t_j) \phi_j(t_i)
\]  
(A.1)

Taking the derivative of the above polynomial leads to
\[
\dot{x}^N(\tau_i) = \sum_{j=1}^{N} x(t_j) \dot{\phi}_j(t_i) = \sum_{j=1}^{N} D_{ij} x(t)
\]  
(A.2)

where
\[
D_{ij} = \dot{\phi}_j(t_i)
\]  
(A.3)

Solving for the derivative matrix \( D_{ij} \) for the Lagrange interpolating function
can be described as

\[ \phi_j(t) = \prod_{i=1, i \neq j}^{N} \frac{t - t_i}{t_j - t_i} = \frac{z(t)}{(t - t_j) \dot{z}(t_j)} \]  

(A.4)

where \( z(t) \) is defined as

\[ z(t) = \prod_{i=1}^{N} (t - t_i) \]

The node points, \( t_1, \ldots, t_{N-1} \), are the zeros of the derivative of the \( N \)th degree Legendre polynomial, \( L_N(t) \) with end points 1 and -1. The differential form of \( z(t) \) at \( j \) is

\[ \dot{z}(t_j) = (t_j - t_1) \cdots (t_j - t_{j-1})(t - t_{j+1}) \cdots (t - t_N) = \prod_{i=i \neq j}^{N} (t - t_i) \]  

(A.5)

Rewriting \( z(t) \) using the differential form of \( N - 1 \) degree the Legendre polynomial is

\[ z(t) = (t - t_1)L'_{N-1}(t)(t - t_N) = (t + 1)L'_{N-1}(t)(t - 1) = (t^2 - 1)L'_{N-1}(t) \]  

(A.6)

From the differential equation of Sturm-Liouville form

\[ \frac{d}{dt}[(t^2 - 1)L'_{N-1}(t)] = N(N - 1)L_{N-1}(t) \]  

(A.7)

we can obtain the derivative form

\[ \dot{z}(t) = N(N - 1)L_{N-1}(t) \]  

(A.8)

Thus the Lagrange interpolating basis function can be expressed by the derivative form of the Legendre polynomial by substituting Eq. (A.5) and Eq. (A.7) into
Eq. (A.4)
\[ \phi_j(t) = \frac{(t^2 - 1)L'_{N-1}(t)}{(t - t_j)(N - 1)L_{N-1}(t)} \] (A.9)

The derivative form of Eq. (A.12) at \( i \) is evaluated

\[ \dot{\phi}_j(t_i) = \frac{1}{N(N - 1)L_{N-1}(t_j)} \left[ \frac{2t_iL'_{N-1}(t_i)}{t_i - t_j} + \frac{(t_i^2 - 1)L''_{N-1}(t_i)}{t_i - t_j} - \frac{(t_i^2 - 1)L'_{N-1}(t_i)}{(t_i - t_j)^2} \right] \] (A.10)

From Eq. (A.7)

\[ N(N - 1)L_{N-1}(t) = 2tL'_{N-1}(t) + (t^2 - 1)L''_{N-1}(t) \] (A.11)

Eq. (A.12) can be rewritten as

\[ \dot{\phi}_j(t_i) = \frac{1}{N(N - 1)L_{N-1}(t_j)} \left[ \frac{N(N - 1)L_{N-1}(t_i)}{t_i - t_j} - \frac{(t_i^2 - 1)L'_{N-1}(t_i)}{(t_i - t_j)^2} \right] \] (A.12)

Since \( (t^2 - 1)L'_{N-1}(t_i) = 0 \), we can obtain \( \dot{\phi}_j(t_i) \) and rewrite Eq. (A.3)

\[ D_{ij} = \dot{\phi}(t_i) = \frac{L_{N-1}(t_i)}{(t_i - t_j)L_{N-1}(t_j)} \] (A.13)
Constrained Station Change Code

B.1 Station Change Main Routine

This is the FORTRAN 90 main routine for the longitude station change maneuver which is introduced in Chapter 4. The continuous optimal control problem is converted into parameterized form; and then the problem is solved as a nonlinear programming problem. The user has to define the constraint and objective functions to be called by NPSOL. The derivatives of the constraints and objective function can be provided; alternatively, NPSOL can evaluate the Jacobian matrix using a central difference method based on a user’s specified option. The constraint defect is set up by the Legendre pseudospectral method.

\[
\frac{\tau_f - \tau_0}{2} \sum_{k=0}^{N} \mathcal{L}(x_k, u_k) \omega_k - \sum_{l=0}^{N} D_{kl} x(t_l) = 0
\]  

(B.1)

B.2 Makefile
jj:  LGL_STCH_Main.o  lib_algb.a
ifort  -o  jj  LGL_STCH_Main.o  lib_algb.a

LGL_STCH_Main.o:  LGL_STCH_Main.FOR
ifort  -c  -r8  LGL_STCH_Main.FOR

BLAS1.o:  BLAS1.FOR
ifort  -c  -r8  BLAS1.FOR

BLAS2.o:  BLAS2.FOR
ifort  -c  -r8  BLAS2.FOR

CHSUBS.o:  CHSUBS.FOR
ifort  -c  -r8  CHSUBS.FOR

CMSUBS.o:  CMSUBS.FOR
ifort  -c  -r8  CMSUBS.FOR

CNPSUBS.o:  CNPSUBS.FOR
ifort  -c  -r8  CNPSUBS.FOR

F06SUBS.o:  F06SUBS.FOR
ifort  -c  -r8  F06SUBS.FOR

LSSUBS.o:  LSSUBS.FOR
ifort  -c  -r8  LSSUBS.FOR

MCSUBS.o:  MCSUBS.FOR
ifort  -c  -r8  MCSUBS.FOR

NPSUBS.o:  NPSUBS.FOR
ifort  -c  -r8  NPSUBS.FOR

OPSUBS.o:  OPSUBS.FOR
B.3 Main Routine

c == Module for basic data --------------------------------------

Module DATA_STCH_Parameter

Implicit None

! Num of States, cont, NODES and others parameters
INTEGER, parameter :: Nstates= 5, nnodes=41, ncv= Nnodes
integer, parameter :: Nlps = Nstates * nnodes, Nlpc = nnodes

integer, parameter :: ldA = 1
integer, parameter :: Ntime = 1
integer, PARAMETER :: Maxn = Nlps + Nlpc + Ntime ! Max states
integer, PARAMETER :: ldR = Maxn
integer, PARAMETER :: NEqual = 1

integer, parameter :: Nclinr = 0
integer, parameter :: ldCj = Nnodes * NSTATES + NEqual

integer, parameter :: liwork = 3 * Maxn + Nclinr + 2 * ldCj
integer, parameter :: lwork = 235298 ! 390000
! (2 * Maxn * Maxn + 2 * Maxn * ldCj + 20 * Maxn + 21 * ldCj)

integer, parameter :: maxbnd = Maxn + ldCj

Real(8), parameter :: zero = 0.0d+0, one = 1.0d+0, two = 2.0d+0
Real(8), parameter :: pi = 3.1415D0
REAL(8), PARAMETER :: BIGBND = 1.0E21

! Earth $ spacecraft data ------------------------------

Real(8), parameter :: omega_e = 7.292116e-5
Real(8), parameter :: F0 = 2.24
Real(8), parameter :: mu = 3.986012e5
Real(8), parameter :: R0 = 42164.2

Real(8), parameter :: m0 = 1000.
Real(8), parameter :: acc0 = F0/m0
Real(8), parameter :: uniacc = mu/(r0*r0)
Real(8), parameter :: DU = R0
Real(8),parameter :: TU = 86400/2/PI

!Real(8),parameter :: unitvel = (mu/r0)**.5
Real(8),parameter :: unitvel = DU/TU
Real(8),parameter :: g = 9.8
Real(8),parameter :: Isp = 1000.
Real(8),parameter :: iniacc = acc0/(uniacc*1.e3)
Real(8),parameter :: Ve = Isp*g*1.E-3/Unitvel

! offset to collision avoidance distance :: 0.001 = 42.1642 km
Real(8), parameter :: Outer = 1.1
Real(8), parameter :: INNER = 0.99

! == Bounds and Direction ------------------------------

INTEGER, PARAMETER :: FLAG = 1
INTEGER, parameter :: NumR = 2
    REAL(8), parameter :: Long = 10
REAL(8), parameter :: Finangle= NumR*2*pi + Long*pi/180
Real(8), parameter :: Tfguess = 100

Real(8), parameter :: WGT = 1

End Module DATA_STCH_Parameter

! ------------------------------------------------------

PROGRAM LGL_stch_Main
Use DATA_STCH_Parameter

implicit DOUBLE PRECISION (a-h,o-z)

* Set the declared array dimensions.
* ldA(>=1) = the declared leading dimension of A.
* ldCJ = the declared leading dimension of cJac.
* ldR = the declared leading dimension of R.
* maxn = maximum no. of variables allowed for.
* maxbnd = maximum no. of variables + linear
* & nonlinear constraints.
* liwork = the length of the integer work array.
* lwork = the length of the double precision work array.
* nnodes = number of nodes
* ncv = number of control variables
* ncp = number of control parameter
* nlps = total number of state variables
* nlpc = total number of control variables

*nstates(=5) = number of states
* ncp = number of control variables

c = ===================================================
! iwork --> iw = is an integer array of dimension liwork
! that provides integer for NPSOL
! liwork --> leniw = is the dimension of iw. It must be at least
! 3n+nclin+2ncnln= 861
! work --> w = is an array of dimension lwork that provides
! real workspace for NPSOL
! lwork --> lenw = is the dimension of w. If there are no general
! linear constraints and
! no nonlinear constraints(nclin=0 anc ncnln=00),
! lenw must be at least
! ldcJ = num of constraints

c = ===================================================
  integer istate(maxbnd)
  integer iwork(liwork)
  double precision xini(nstates)
  double precision A(ldA,maxn)
number of bound array = num of state + total constraints
= maxn + nstates*nnodes + 1

double precision :: bl(maxbnd), bu(maxbnd)
double precision :: c(ldcJ), cJac(ldcJ,maxn), clamda(maxbnd)
double precision :: objgrd(maxn), R(ldR,maxn)
double precision :: X(maxn),U1(nnodes)
double precision :: work(lwork)

! setting vector for pseudo module
real(8), Dimension(nnodes) :: X1, X2, X3, X4, X5
real(8), Dimension(nnodes) :: T, TAU, TDAY
REAL(8), Dimension(nnodes) :: ET, VN, WT, CO, ENER
Real(8) :: DFA(nnodes,nnodes)

External fobj1, fncon1
character*40 lFile, DATE

logical byname, byunit, EAST
iNTEGER NO7
COMMON /DIR/ RMIN, RMAX

print*, iniacc
NO7 = 7
OPEN (NO7, File='CH_STCH.OUP', status='UNKNOWN')
WRITE(NO7,*)'maxbnd=', maxbnd

* Assign file numbers and open files by various means.
* (Some systems don’t need explicit open statements.)
* iOptns = unit number for the Options file.
* iPrint = unit number for the Print file.
iSumm = unit number for the Summary file.

iOptns = 4
iPrint = 10
iSumm = 6
byname = .true.
byunit = .false.

if (byname) then
  lFile = 'NPSOL.OPT'
  open( iOptns, file=lFile, status='OLD', err=800 )

  lFile = 'LGL_STCH_OUP.txt'
  open( iPrint, file=lFile, status='UNKNOWN', err=800 )
else if (byunit) then
  lUnit = iOptns
  open( lUnit , status='OLD', err=900 )

  lUnit = iPrint
  open( lUnit , status='UNKNOWN', err=900 )
end if

nclin = 0
ncnln = ldcJ

N = MAXN
nbnd = N + nclin + ncnln
* Assign the data arrays.
* \( A \) = the linear constraint matrix.
* \( b_l \) = the lower bounds on \( x \), \( a'x \) and \( c(x) \).
* \( b_u \) = the upper bounds on \( x \), \( a'x \) and \( c(x) \).
* bounds \( \geq \) \( \text{bigbnd} \) will be treated as plus infinity.
* bounds \( \leq \) \( -\text{bigbnd} \) will be treated as minus infinity.
* \( x_{ini} \) = the initial estimate of the solution.

---

**Define Transfer Direction**

IF(Flag .EQ. 1) then
    Rmin = iNNER
    Rmax = 1.0
ELSE
    Rmin = 1.0
    Rmax = Outer
ENDIF

---

**Initial condition**

\( x_{ini}(1) = \text{ONE} \)
\( x_{ini}(2) = \text{ZERO} \)
\( x_{ini}(3) = \text{ZERO} \)
\( x_{ini}(4) = \text{ONE} \)
\( x_{ini}(5) = \text{INIACC} \)

---

**Bounds for state variables**
! Do i=1, NLPS
! bl(i) = -Bigbnd
! bu(i) = +Bigbnd
! End do

Do 555 i = 1, NLPS

**c == bound for radius ========**
if(mod(i,nstates) .eq. 1)then
  bl(i) = RMIN
  bu(i) = RMAX
end if

**c == reference angle =========**
  if(mod(i,nstates) .eq. 2)then
    bl(i) = ZERO
    bu(i) = BIGBND
  end if

**c == bound for rad. velocity =======**
  if(mod(i,nstates) .eq. 3)then
    bl(i) = -1.0
    bu(i) =  2.0
  end if

**c == bound for tan vel =======**
  if(mod(i,nstates) .eq. 4)then
    bl(i) =  0.0
    bu(i) =  2.0
  end if
c == bound for thrust acceleration ==
    if(mod(i,nstates) .eq. 0)then
        bl(i) = 0.0000001
        bu(i) = .05
    end if

555 CONTINUE

c ! bound for initial state at time zero ---------------
DO i=1, nstates
    bl(i) = xini(i)
    bu(i) = xini(i)
ENDDO

c Final bounds at tf -----------------------------------
    bl(nlps-4) = xini(1)
    bu(nlps-4) = xini(1)
    bu(nlps-2) = xini(3)
    bl(nlps-2) = xini(3)
    bu(nlps-1) = xini(4)
    bl(nlps-1) = xini(4)

c End of bound for states -------------------------------

c bounds for control variable ---------------------
! nlps = number of state variables
! nlpc = number of control variables
! nscv = total number of state & control variables

nscv = nlps + nlpc
do i=nlps+1, nscv
   bl(i) = -pi + 0.0001
   bu(i) = pi
end do

c bound for Time end -----------------------------------
b(l(n)) = 0.0001
bu(n) = Bigbnd

c bound for Time end -----------------------------------

c ! Specifying bounds for nonlinear constraints
! locate after the control variable bounds
DO j = maxn+1, nbnd
   bl(j) = zero
   bu(j) = zero
ENDDO

c ! Specifying bounds for nonlinear constraints

c == Set the initial guess of x, state vector ===========
! Initial guess for state variables
DO   i=1, nstates
   x(i) = xini(i)
ENDDO

c == initial guess for radius -----------------------------
DO   i = nstates+1, nlps-4, nstates
   x(i) = x(i-nstates) + 0.001
c == initial guess for ref angle
DO i = nstates+2, nlps-3, nstates
  x(i) = x(i-nstates) + 0.01*pi/180
ENDDO

c == initial guess radial velocity
DO i = nstates+3, nlps-2, nstates
  x(i) = x(i-nstates) + 0.1
ENDDO

c == initial guess for tan velocity
DO i = nstates+4, nlps-1, nstates
  X(i) = X(i-nstates) + 0.02
ENDDO

c == initial guess for tan velocity
DO i = nstates+5, nlps, nstates
  X(i) = X(i-nstates) + 0.0001
ENDDO

c == Set the initial guess of x, state vector

c ==== Initial guess for control variables
! nlpc = tot num of control variables
! nscv = tot number of states and control
u10 = 0.001
X(nlps+1) = U10
c nscv = nlps + nlpc
DO i = nlps+2, nscv
  X(i) = X(i-1) + 1*pi/180 ! delu !+ 1*pi/180
ENDDO

c ==== Initial guess for final time ===========
X(N) = Tfguess

c ==== Initial guess for final time ===========

c -----------------------------
* Set a few options in-line.
* The Print file will be on unit iPrint.
* The Summary file will be on the default unit 6
* (typically the screen).
c -----------------------------
call npopti( 'Print file =' , iPrint )
call npoptr( 'Infinite Bound size =', bigbnd )
write(NO7,*) 'Chk 7', 'iPrint=',iPrint,'bigbnd=', bigbnd ! iPrint = 10

c Read the Options file.
call npfile( iOptns, inform )
write(NO7,*) 'Chk 8', 'iOptns=',iOptns,'inform=', inform

! iOptns = 4, inform = 0
  if (inform .ne. 0) then
    write(iPrint, 3000) inform
    stop
  End if
* Solve the problem.

! input ==================================
! n : (>0) the number of variables in the problem
! nclin : (>=0) the number of general linear constraints
! ncnln : (>=0) the number of general nonlinear constraints
! ldA : (>=1 and >=nclin) row dimension of the array A
! ldJ : (>=1 and >=nclin) row dimension of the array cJac
! ldR : (>=n) row dimension of the array R
! A : the array of dimension(ldA,k) for some k>= n.
! It contains the matrix A for the linear constraints.
! bl : array of dimension at least nctotl that contains l,
! the lower bounds for r(x) in the problem NP.

! To specify a non-existent lower bound,
! set bl(j)<= -bigbnd, when bigbnd is inf.
! To specify an equality constraint an equality
! constraint(say r(x)=beta), bl(j) = bu(j) = beta
! bu : array of dimension at least nctotl that
! contains lower bounds for r(x) in NP
!
! funcon : the name of subroutine
!
! output ==================================
! inform :
! iter :
! istate : status of constrain number of iterations
! l< r(x) < u in NP
! c : an array of dimension at least ncnln,
! if ncnln=0, c is not accessed
! jJac : Jacobian matrix
! clamda : contains OP multipliers from the last OP subproblems
! at thr final iterate
! f : objective func
! g : array of dimension at least n that contains
! the obj gradient at the final state
! R : contains information about H, the Hessian of the Lagrangian
! x : final estimation of solution

c --------------------------------------------------------

! --- To show the input data on the screen ---
CALL PRINTOUT_INPUT_SCREEN()

c == 1st call to NPSOL ================================
CALL NPSOL ( n, nclin, ncnln, ldA, ldcJ, ldR,
$ A, bl, bu,
$ fncon1, fnobj1,
$ inform, iter, istate,
$ c, cJac, clamda, objf, objgrd, R, x,
$ iwork, liwork, work, lwork )

c --------------------------------------------
call cpu_time(TEND)
PRINT*, 'CPUTIME:', TEND/60.

c Time of Flight ------------------------------------------
Tau0 = 0.
Tauf = X(N)

c Order of Polynomial -----------------------------------
N1 = Nnodes-1
NM = N1

c Evaluate LGL Nodes calling Pseudo Module
CALL LGL_Pseudo_Module(N1, NM, ET, VN, DFM, WT)
! Et(N1+1) : LGL nodes
! VN(N1+1) : Values at LGL PNTS
! WT(N1+1) : Weights

c Legendre Gauss Lobatto points -----------------------------
t = ET ! ET(nnodes) t=[t0, t_N] = [-1, 1 ]

c Mapping time domain to  t = [-1, 1]
do i =1, nNodes
tau(i) = 0.5*( (tauf-tau0)*t(i) + (tauf+tau0) )
end do

c Time unit for Conversion ------------------------------
TimeU = sqrt(r0*r0*r0/mu) ! time unit [sec]

c Constants ---------------------------------------------
do j= 1, nnodes

tday(j) = tau(j)*TIMEU/(3600*24)

X1(j) = x( (j-1)*nstates + 1)
x2(j) = x( (j-1)*nstates + 2)*180/pi
x3(j) = x( (j-1)*nstates + 3)
x4(j) = x( (j-1)*nstates + 4)
x5(j) = x( (j-1)*nstates + 5)

u1(j) = x(nlps + j)  ! *180/pi
vel2 = x3(j)*x3(j) + x4(j)*x4(j)
Ener(j) = 0.5*vel2 - 1/x1(j) ! energy

END DO
c Constants ---------------------------------------------
do i=1, nnodes
!write(NO7,114) tday(i),x1(i),x2(i),x3(i),x4(i),x5(i),u1(i),ener(i)
write(*,114) tday(i),x1(i),x2(i),x3(i),x4(i),x5(i),u1(i) !,ener(i)
end do

114 format(1x,8(f15.8))

call PRINTOUT(inform, Tend, NO7, tday,
           & Ener, x1, x2, x3, x4,x5, u1)

if (inform .gt. 0) go to 999

* --------------------------------------------------------------
* The following is for illustrative purposes only.
* A second run solves the same problem,
* but defines the objective and constraints via
* the subroutines fnobj2 and fncon2. Some
* objective derivatives and the constant
* Jacobian elements are not supplied.
* We do a warm start using
* istate   (the working set)
* clamda   (the Lagrange multipliers)
* R        (the Hessian approximation)
* from the previous run, but with a slightly perturbed starting
* point. The previous option file must have specified
* Hessian    Yes
* for R to be a useful approximation.
* --------------------------------------------------------------

* Set some new options in-line,
* but stop listing them on the Print file.

call npoptn( 'Nolist'    )
call npoptn( 'Derivative level 0' )
call npoptn( 'Verify No' )
call npoptn( 'Warm Start' )
call npopti( 'Major iterations ', 50 ) !20050
call npopti( 'Major print level ', 10 )

* ! if Derivative level=0/1 NPSOL will approximate unspecified elements fo Jacobian.*
* ! One call to function is needed for each variable for which partial derivatives are not available.
* ! At times central diff are used rather than fwd diff, in which case twice as many calls to funobj and funcon are needed.

if (inform .gt. 0) go to 999
* Error conditions.

  800 write(iSumm , 4000) 'Error while opening file', lfile
  stop

  900 write(iSumm , 4010) 'Error while opening unit', lunit
  stop

  999 write(iPrint, 3010) inform
  stop

  3000 format(/ ' npfile terminated with inform =', i3)
  3010 format(/ ' npsol terminated with inform =', i3)
  4000 format(/ a, 2x, a )
  4010 format(/ a, 2x, i6 )

**** end of the main program No_Coast_Mod
End Program LGL_stch_Main
c User-defined Objective function  --------------
subroutine Fnobj1(mode, n, x, objf, objgrd, nstate )

USE DATA_STCH_Parameter, ONLY: Nstates, Nnodes, NLPS, FLAG, & ZERO, WGT

iMplicit double precision(a-h,o-z)
integer mode, n, nstate
REAL(8) x(n), objgrd(n)

Integer :: Nr, Nc
REAL(8), DIMENSION(NNODES) :: A1, W
REAL(8), DIMENSION(NNODES) :: ET, VN, WT
Real(8), Dimension(nnodes,nnodes):: DMA
COMMON / DIR / RMIN, RMAX

C User-defined Objective function  --------------

C  ____________________________
* fnobj1 computes the value and first derivatives
* of the nonlinear objective function.
C  ____________________________

Nr = Nnodes-1
Nc = Nr
tf = x(n)

C Set up Matrices A(l): from x(i) states at LGL NODES -------
DO j = 1, NNODES
A1(j) = x((j-1)*nstates + 1)
c End Set up Matrices A(l): from x(i)  

---

c Weight at LGL points
N1 = Nnodes-1  ! N1 = Led Poly Order
NM = N1

CALL LGL_Pseudo_Module(N1, NM, ET, VN, DMA, WT)

IF (FLAG. eq. 1) THEN
RT=RMIN
ELSE
RT=RMAX
ENDIF

SUM = ZERO
! Gauss Integral at LGL points: \int_1^N f(x(i))*W(i)dx
DO i = 1, NNODES
SUM = ( RT - A1(i) )*( RT - A1(i) )*WT(i) + SUM
! PRINT*, 'Weighting factor ', I, WT(I)
End DO

C Objf = tf + K*\int_0^tf(R - R_min)^2 dt
! WET : weight factor

Objf = tf + 0.5*tf*WGT*SUM
RETURN
*
end of fnobj1

End subroutine Fnobj1

---
c Constraint Function ---------------------------------------------
* fncon computes the values and first derivatives
* of the nonlinear constraints.
*
* The zero elements of Jacobian matrix are set only once. This
* occurs during the first call to fncon (nstate = 1).
c ----------------------------------------------------------

subroutine Fncon1(mode, ncnln, n, ldcJ,
$ needc, x, c, cJac, nstate)

USE DATA_STCH_Parameter, ONLY: Nstates, Nnodes, NLPS, ZERO
  & ,NUMR, finangle

  == input+++++++++++++++++++++++++++++++++
! nstates = number of state vector for one segment
! ncv = number of control vector for one segment

! resid : state defect vector for node k
!
  == output+++++++++++++++++++++++++++++++++
! mode : can be used to end the solution of the currente problem
! c : array of dimension at least ncnln that contains
! the appropriate values of the nonlinear constraints.
! cJac : array of declared dimension (1dJ,k), where k>= n.
c ----------------------------------------------------------
implicit Double precision(a-h,o-z)
integer mode, ncnln, n, Nstate
integer needc(*)
real(8) :: x(n), c(*), cJac(ldcJ,*)

    ! ---- specify vector space for psdueo module ----
inTEGER :: AROW, ACOL, BROW, BCOL
real(8) :: ti(nnodes), tau(nnodes) 

Real(8), dimension(Nnodes,1) :: A1, A2, A3, A4, A5  
Real(8), DIMENSION(NNODES) :: B1  
Real(8), dimension(NNODES,1) :: CK,CK1,CK2,CK3,CK4,CK5  
real(8), dimension(NNODES) :: F1ab, F2ab,F3ab,F4ab,F5ab  

real(8), dimension(nstates) :: Fx, XS, tk  
real(8) :: resid(nnodes*nstates)  
real(8), dimension(nnodes) :: ET,CO,WT, VN  
real(8) :: DKC(1,NNODES), DMA(nnodes,nnodes)  

PI = 4.0*ATAN(1.0D0) 

c Set up Matrices A(l): from x(i) states at LGL NODES -----  
DO j=1, NNODES  
A1(j,1) = x( (j-1)*nstates + 1 )  
A2(j,1) = x( (j-1)*nstates + 2 )  
A3(j,1) = x( (j-1)*nstates + 3 )  
A4(j,1) = x( (j-1)*nstates + 4 )  
A5(j,1) = x( (j-1)*nstates + 5 )  
B1(j) = X(NLPS+j)  
ENDDO  
c End Set up Matrices A(l) B(l): from x(i) -----  

c Find LGL point from temp tf  
tau0 = ZERO  
tauf = x(N)
c Evaluate LGL Nodes
N1 = Nnodes-1 ! N1 = Led Poly Order
NM = N1

call LGL_Pseudo_Module(N1, NM, ET, VN, DMA, WT)
c LGL points
ti = ET ! ET(nnodes) t=[-1, 1 ]

c Mapping time domain using t(i)
do i =1, nnodes
Tau(i) = 0.5*((tauf-tau0)*ti(i) + (tauf+tau0) )
end do

c ---------------------------------------------------------
c Derivative of state vector x(t), collocated at LGL ponits
! c(tk) - f(tk,uk) = 0
! DMA(nnodes,nnodes) : Differentiation Matrix
c ---------------------------------------------------------
AROW = nnodes
ACOL = AROW
BROW = AROW
BCOL = 1

c Evaluate Derivative at collocated nodes C(K) VECTOR ---------
c === C(k) = Sum_{0}{N} D_(i,j)*A(j)
call matmult(DMA, A1, CK1, arow, acol, brow, bcol)
call matmult(DMA, A2, CK2, arow, acol, brow, bcol)
call matmult(DMA, A3, CK3, arow, acol, brow, bcol)
call matmult(DMA, A4, CK4, arow, acol, brow, bcol)
call matmult(DMA, A5, CK5, arow, acol, brow, bcol)
Do  j=1, nnodes
   tk = tau(j)  ! t = [-1,1]
x(1) = A1(j,1)
x(2) = A2(j,1)
x(3) = A3(j,1)
x(4) = A4(j,1)
x(5) = A5(j,1)
uc = B1(j)
Call Eom(xs, uc, fx)

! tk(1), xs(4), uc(1)/ Exit: fx(4)
f1ab(j) = fx(1)  ! dr
f2ab(j) = fx(2)  ! dtheta
f3ab(j) = fx(3)  ! du
f4ab(j) = fx(4)  ! dv
f5ab(j) = fx(5)  ! dv
ENDDO

c --- Evaluation Defects -------------------------------
c defect : (tauf-tau0)/2 Int_f(a(tk),u(tk),t) - c(k) =0
c -----------------------------------------------

DO 31 k =1, Nnodes
   if(k.eq.1) then  ! first LGL node
      nks = 0
   else
      nks = (k-1)*nstates
   endif

DO 32 i=1, nstates
if(i.eq.1)then
    resid(nks+i) = 0.5*(tauf-tau0)*F1ab(k)-CK1(k,1)
end if

if(i.eq.2)then
    resid(nks+i) = 0.5*(tauf-tau0)*F2ab(k)-Ck2(k,1)
end if

if(i.eq.3)then
    resid(nks+i) = 0.5*(tauf-tau0)*F3ab(k)-CK3(k,1)
end if

if(i.eq.4)then
    resid(nks+i) = 0.5*(tauf-tau0)*F4ab(k)-CK4(k,1)
end if

if(i.eq.5)then
    resid(nks+i) = 0.5*(tauf-tau0)*F5ab(k)-CK5(k,1)
end if

32         END DO
31         END DO

c Nonlinear Equality constraints at LGL points
DO i=1, Nnodes*Nstates
    c(i) = resid(i)
ENDDO

! Equality constraints for referane angle
    c(nnodes*nstates+1) = Finangle - x(nlps-3)
return

! end of funcon1
Subroutine Eom(xs, uc, xdot)
USE DATA_STCH_Parameter

! c input
! t = current time
! xs = current state vector
! uc = current control vector

! c output
! xdot(nstates) = equations of motion

implicit double precision (a-h, o-z)
real(8), intent(in) :: xs(nstates), uc
real(8), intent(out):: xdot(nstates)

! mdot = - F/(Isp*g)
! ve = Isp*g*(1.0e-3)/unitvel
! ve = 3.18734  ! if F=2.4 Newton

! evaluate equations of motion
xdot(1) = xs(3)
xdot(2) = xs(4)/xs(1)
xdot(3) = xs(4)*xs(4)/xs(1) - 1.0/(xs(1)*xs(1)) + xs(5)*sin(uc)
xdot(4) = -xs(3)* xs(4)/xs(1) + xs(5)*cos(uc)
xdot(5) = xs(5)*xs(5)/ve
return
* end of eom
End subroutine EOM

* *** Matrix Multiplication ************
Subroutine Matmult(A,B,C,arow,acol,brow,bcol)
c -------------------------------------------------
c Matrix Multiplication: A(M1,N1)*B(M2,N2) = C(M1,N2)
c -------------------------------------------------
Implicit None
integer :: k, ROW, COL
integer, intent(in) :: Arow, Acol, Brow, Bcol
     real(8), intent(IN) :: A(Arow,Acol), B(Brow,Bcol)
     REAL(8), INTENT(OUT) :: C(Arow,Bcol)

! IMPLEMENTATION
do 11 row=1, Arow
   do 12 col = 1, Bcol
      C(row, col)= 0.0d0
         do 13 k = 1, acol
            C(row, col) = C(row, col)+
               & A(row, k)*B(k,COL)
      13 end do
   12 end do
11 end DO

END SUBROUTINE matmult
c -------------------------------------------------
c --- Print out -----------------------------------------
c --- Print out -----------------------------------------
SUBROUTINE PRINTOUT_INPUT_SCREEN()

Use DATA_STCH_Parameter
iMPLICIT double precision(a-h, o-z)
! parameter (nnodes=64, PI=3.141592D0)

INTEGER :: inform, N07
C integer, intent(in) :: Nnodes,N,NCLIN,NCNLN,LDA,LDCJ,
C & NUMR, INIACC, VE
!REAL(8),DIMENSION(NNODES):: Ener,tau,tday,x1,x2,x3,x4,x5,u1

PRINT*, ' --- Before NPSOL --- ' 
PRINT*, ' ' 
PRINT*, ' iNITIAL iNPUT Data ' 
PRINT*, ' ' 
PRINT*, 'Number of Nodes ', Nnodes
c WRITE(no7,*)'Number of Nodes ', Nnodes
PRINT*, 'Number of variables ', MaxN
c WRITE(no7,*)'Number of variables ', N
PRINT*, 'Number of Linear Constraints ', NCLINR
c WRITE(no7,*)'Number of Linear Constraints ', NCLIN
PRINT*, 'Row dimension of the array A ', LDA
c WRITE(no7,*)'Row dimension of the array A ', LDA
PRINT*, 'Number of Nonlinear Constraints ', LDCJ
c WRITE(no7,*)'Number of Nonlinear Constraints ', LDCJ
PRINT*, 'Row dimension of the array R ', LDR
print*, 'Max bnd ', maxbnd
PRINT*, ' ' ' 
PRINT*, 'Initial Acceleration ', INIACC 
PRINT*, 'Effective velocity ', VE 
print*, ' ' 
print*, 'Thrust(Newton) :', FO 

Print*, ' ' 
Print*, '--- Direction --- ' 
PRINT*, 'FLAG : ', FLAG 
Print*, 'Num of Rev : ', NumR 

if(FLAG .EQ. 0) then 
    print*, 'West Direction ' 
    ! write(NO7,*) ' West Direction ' 
else 
    print*, 'East Direction ' 
    ! write(NO7,*) 'East Direction ' 
endif 
print*, 'Longitude change :', Long 
print*, 'Weight :', WGT 

RETURN 

END SUBROUTINE PRINTOUT_INPUT_SCREEN 

c ------------- 

c Print Out Results Module --------------------------------------- 
SUBROUTINE PRINTOUT(inform,Tend,NO7,tday, 
    & Ener, x1, x2, x3, x4, x5, u1)
Use DATA_STCH_Parameter

iMPLICIT double precision(a-h, o-z)

CHARACTER*20 :: DATE
INTEGER :: inform, NO7

REAL(8) :: TEND
REAL(8), DIMENSION(NNODES):: Ener, tday, x1, x2, x3, x4, x5, u1

C ----------------------------------------------------------
c Print all states and control+++++++++++++++++++++++++++++++
call date_and_time(DATE)

c Screen -----------------------------------------------
WRITE(NO7,*) 'Cpute time(min) :', TEND/60.
WRITE(NO7,*) 'Thrust (Newton) :', F0
WRITE(NO7,*) 'Initial Accel :', x5(1)
WRITE(NO7,*) 'Longitude Change :', Long
WRITE(NO7,*) 'Weight Factor :', WGT

WRITE(NO7,*) 'inform :', inform

C Station change direction -------------------------------
WRITE(NO7,*) ' Dierction ---'
WRITE(NO7,*) 'FLAG :', FLAG
WRITE(NO7,*) 'Num of Rev :', NumR

if(FLAG .EQ. 0)then
   write(NO7,*) ' West Direction '
else
   write(NO7,*) 'East Direction '

endif

C Chk convergence ------------------------------------------

if(inform .eq. 0)then
    write(NO7,*)'Optimal Solution Converged!'
else
    write(NO7,*)'Never Give UP One More try ! '
endif

c Energy variation --------------------------------------------
WRITE(NO7,*) 'Final time', x2(NNODES)

    write(NO7,*)
write(NO7,*)'--------- Print Time, states & control ----------'
write(NO7,*)'Tday  r  ref_ang  u  v  acc  con  ener'
do i=1, nnodes
write(NO7,114) tday(i),x1(i),x2(i),x3(i),x4(i),x5(i),u1(i),ener(i)
end do

114 format(1x,8(f15.8))
write(NO7,*) 'Date :', date
RETURN

END Subroutine PRINTOUT
C End print all states and control -----------------------------
Bibliography


Vita

Seung Pil Kim

Seung Pil Kim was born and raised in Jeju, South Korea, where he graduated from Ohyun high school. He received his Bachelor’s degree in Aeronautical Engineering from R.O.K Air Force Academy in 1994. He has served as an air crew (Weapon System Officer) of the fighter jet, F-4D and RF-4C. He earned his Master’s degree on the experiment of dynamic derivatives for high-performance capability maneuver aircraft model with forced vibration methods from AIMST at Air Force Academy in 2001. He has been employed in aircraft maintenance group and worked as an evaluator for long-term projects of the Air Force. He went to The Pennsylvania State University, University Park, to pursue his terminal degree in Aerospace Engineering under the guidance of Dr. Robert G. Melton. He completed his Ph.D. degree on optimal trajectories for station change in GEO in 2012