11:54:42 OCA PAD AMENDMENT - PROJECT HEADER INFORMATION 02/06/96

Active

Project #: E-16-M45 Cost share #: Rev #: 6
Center # : 10/24-6-R7552-0A0 Center shr #: OCA file #:
Work type : RES
Contract#: NAG8-921 Mod #: SUPPLEMENT 4 Document : GRANT
Prime #: Contract entity: GTRC

Subprojects ? : N CFDA: 43.002
Main project #: PE #: N/A

Project unit: AERO ENGR Unit code: 02.010.110
Project director(s):
CHUANG C H AERO ENGR (404)894-3075

Sponsor/division names: NASA / MARSHALL SPACE FLT CTR, AL
Sponsor/division codes: 105 / 005

Award period: 920707 to 960630 (performance) 960930 (reports)

Sponsor amount	New this change	Total to date
Contract value	0.00	119,459.00
Funded	0.00	119,459.00
Cost sharing amount		0.00

Does subcontracting plan apply ?: N

Title: THEORY & COMPUTATION OF OPTIMAL LOW AND MEDIUM THRUST ORBIT TRANSFERS

PROJECT ADMINISTRATION DATA

OCA contact: Anita D. Rowland 894-4820

Sponsor technical contact

DR. JOHN M. HANSON, EL58
(205)544-2289

NASA MARSHALL SPACE FLIGHT CENTER
MARSHALL SPACE FLIGHT CENTER, AL
35812

Sponsor issuing office

MARLYCE ALEXANDER, GP32-L
(205)544-8344

NASA MARSHALL SPACE FLIGHT CENTER
MARSHALL SPACE FLIGHT CENTER, AL
35812

Security class (U,C,S,TS) : U ONR resident rep. is ACO (Y/N): Y
Defense priority rating : N/A N/A supplemental sheet
Equipment title vests with: Sponsor GIT X
NONE PROPOSED.
Administrative comments -
SUPPLEMENT 4 AWARDS ANCE THRU 6.30.96

GEORGIA INSTITUTE OF TECHNOLOGY
OFFICE OF CONTRACT ADMINISTRATION

4

NOTICE OF PROJECT CLOSEOUT

Closeout Notice Date 10/28/96

Project No. E-16-M45_____ Center No. 10/24-6-R7552-0A0_

Project Director CHUANG C H_____ School/Lab AERO ENGR_____

Sponsor NASA/MARSHALL SPACE FLT CTR, AL_____

Contract/Grant No. NAG8-921_____ Contract Entity GTRC

Prime Contract No. _____

Title THEORY & COMPUTATION OF OPTIMAL LOW AND MEDIUM THRUST ORBIT TRANSFERS____

Effective Completion Date 960630 (Performance) 960930 (Reports)

Closeout Actions Required:	Y/N	Date Submitted
Final Invoice or Copy of Final Invoice	Y	960906
Final Report of Inventions and/or Subcontracts	Y	_____
Government Property Inventory & Related Certificate	Y	_____
Classified Material Certificate	N	_____
Release and Assignment	N	_____
Other _____	N	_____

Comments_____

Subproject Under Main Project No. _____

Continues Project No. _____

Distribution Required:

Project Director	Y
Administrative Network Representative	Y
GTRI Accounting/Grants and Contracts	Y
Procurement/Supply Services	Y
Research Property Managment	Y
Research Security Services	N
Reports Coordinator (OCA)	Y
GTRC	Y
Project File	Y
Other _____	N
_____	N

NOTE: Final Patent Questionnaire sent to PDPI.

SEMI-ANNUAL STATUS REPORT

Submitted to: NASA Marshall Space Flight Center

Grant Title: Theory and Computation of Optimal Low- and Medium-Thrust Transfers

Grant Number: NAG8-921

Principal Investigator /Project Director: Dr. C.-H. Chuang
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, GA 30332-0150
(404) 894-3075

Research Assistant: Troy Goodson
School of Aerospace Engineering
Georgia Institute of Technology

Period Covered: July 7, 1992 to January 6, 1993

Date of Submission: January 5, 1993

TABLE OF CONTENTS

SUMMARY OF THE RESEARCH RESULTS ii
ABSTRACT 1
I. INTRODUCTION 1
II. THE PROBLEM 3
III. THE FIRST-ORDER NECESSARY CONDITIONS 8
III.1. THE HAMILTONIAN 10
III.2. THE COSTATES 13
IV. SOLVED PROBLEMS 14
IV.1. SIMPLIFICATIONS 14
IV.2. THE TWO-POINT BOUNDARY VALUE PROBLEM 15
IV.3. NON-DIMENSIONALIZATION 17
IV.4. ATMOSPHERE MODEL 19
IV.5. THE MULTIPLE POINT SHOOTING METHOD OF BOUNDSCO 20
IV.6. THE MINIMIZING-BOUNDARY-CONDITION METHOD 21
IV.7. SAMPLE PROBLEMS AND SOLUTIONS 21
V. CONCLUSIONS 23
REFERENCES 24

SUMMARY OF RESEARCH RESULTS

1. The following report titled "Computation of Optimal Low- and Medium- Thrust Orbit Transfers" gives the detail of the research results. We summary these results and future research plan here.

2. The first-order necessary conditions for a general final mass maximization problem has been set up. In the problem formulation we include second-harmonic oblateness, atmospheric drag, and allow three-dimensional, non-coplanar, non-aligned elliptic orbits. In order to ease the numerical calculation we transform the original free final-time problem to a fixed final-time problem, and non-dimensionalize the state variables.

3. Although we can use the constant angular momentum equation, the conservative energy equation, and the orbit equation to specify the boundary conditions for the terminal orbit, we notice that this set of boundary conditions does not uniquely determine an orbit. This is due to the fact that for a given point in space we can have two different velocity vectors (difference in direction only) and yet have the same angular momentum and energy. Proper boundary conditions should be three eccentricity vector equations plus three angular momentum vector equations. Since both eccentricity and angular momentum equations specify the same orbit plane, one of these equations is redundant. That is for a three dimensional problem we only need five equations out of both sets of equations. For two dimensional problem we need two eccentricity equations and one angular momentum equation.

4. We have applied two indirect optimization methods: BOUNDSCO and MBCM (minimizing-boundary-condition method) successfully to several simplified examples. The examples are two dimensional with oblateness effect and atmospheric drag force. Both methods converge to the solutions with about the same sensitivity in the initial guess. Although we have more freedom in selecting the initial guess at every node points, BOUNDSCO does not adjust the number of switching points and the switching pattern during the iteration. On the other hand, MBCM implements the switching function into the integrator and adjust the switching points and the switching pattern automatically during the iteration.

5. Our current plan is to combine advantageous features of BOUNDSCO and MBCM into a new algorithm. The new algorithm will use the idea of the multiple-point shooting method to spread the unknowns among the node points, and between two node points applies the minimizing-boundary-condition method.

6. There is still a question about the local optimum or global optimum for free final time problem. We have some difficulty in converging the transversality condition for the free final time case. In Edelbaum's paper, he shows that three impulses control is usually minimum. However, such claim for low and

medium thrust has not been shown anywhere. Our current hypothesis suggests that the global minimum solution will be at infinite final time and local minimum solutions exist for finite final time. We expect to answer this question by obtaining all the local minimum solutions (if they exist) and compare their cost functions along the final time axis.

Computation of Optimal Low- and Medium-Thrust Orbit Transfers

ABSTRACT

This report presents the formulation of the optimal low- and medium-thrust orbit transfer control problem and methods for numerical solution of the problem. The problem formulation is for final mass maximization and allows for second-harmonic oblateness, atmospheric drag, and three-dimensional, non-coplanar, non-aligned elliptic terminal orbits. We setup some examples to demonstrate the ability of two indirect methods to solve the resulting TPBVPs.

The methods demonstrated are the multiple-point shooting method as formulated in H. J. Oberle's subroutine BOUNDSCO, and the minimizing boundary-condition method (MBCM). We find that although both methods can converge solutions, there are trade-offs to using either method. BOUNDSCO has very poor convergence for guesses that do no exhibit the correct switching structure. MBCM, however, converges for a wider range of guesses. However, BOUNDSCO's multi-point structure allows more freedom in guesses by increasing the node points as opposed to only guessing the initial state in MBCM. Finally, we note an additional drawback for BOUNDSCO: the routine does not supply information to the users routines for switching function polarity but only the location of a preset number of switching points.

I. INTRODUCTION

The ability to perform any given orbit transfer with a minimum use of fuel is obviously desirable. Useful solutions to this problem will account for

at least some approximation to real-life. Therefore, a formulation that includes second-harmonic oblateness and atmospheric drag will be useful.

This report follows such a derivation all the way through to the establishment of a two-point boundary-value problem for optimal low- and medium- thrust orbit transfer. The core cost function is defined simply as the final mass of the spacecraft plus fuel, setting the tone for the maximization problem. The differential constraint is thoroughly defined in terms of the oblateness model and an assumed atmosphere model.

The thrust (control) appears linear in the differential constraint. This results in bang-bang control or singular-arc solutions for the final mass maximization problem. Although bang-bang control is assumed here the possibility of having a singular arc has not been ruled out for a general case. In order to ensure the singular arc solution does not occur, we check the derivative of the switching function at each switching point. However, when our programs reach a non-optimal solution high frequency chattering solutions do occur occasionally. This could indicate that singular-arc solutions are possible for some modification of system parameters and models.

The final mass maximization problem should be a free final time optimal control problem. For impulsive thrust, the Hohmann transfer gives minimum fuel but maximum transfer time. Although the three-impulse Hohmann transfer performs better than a two-impulse Hohmann transfer, Edelbaum[1] shows that the number of impulses may be finite for a global minimum. for low- and medium-thrust orbit transfer the same conclusion has not be shown anywhere. One hypothesis is that the global minimum will be at infinite final time and local minimum solutions exist for finite final time. In other words, this assumes for a given number of switching points (must be at least two) there is a local minimum with finite final time. We do have difficulties in converging the transversality condition corresponding to optimal final time.

We present solutions to three specific optimization problems. These solutions represent the ability of the two TPBVP solvers. The methods

considered are (1) BOUNDSCO, a multi-point shooting algorithm devised by H. J. Oberle and (2) the minimizing boundary-condition method (MBCM), a modification to the shooting method devised by the authors of ref(9).

Both methods converge solutions for about equal sensitivity in initial guesses. In order to achieve the same accuracy along the path, BOUNDSCO needs to converge the boundary conditions at every node point to the same accuracy as the integration routine. the number of switching points and the switching pattern need to be assumed and stay unchanged when BOUNDSCO is used. On the other hand, MBCM does not constrain the number of switching points and MBCM updates the switching pattern along the integrated path.

II. THE PROBLEM

The problem discussed herein is the following: maximize the final mass of a thrusting spacecraft for a given orbital transfer. The craft can be considered as under the influence of some planet's gravitational field and atmospheric drag. The thrust of the spacecraft is limited between zero and some T_{max}. The transfer will be defined by the terminal orbits. Solutions are sought for both fixed and free final time problems and both the case of fixed and free terminal points.

II. 1. *The Cost Functional*

The core cost functional must be defined. We shall define the cost as

$$J = m(t_f) \tag{1}$$

where $m(t_f)$ represents the mass of the spacecraft plus its fuel at the end of the orbital transfer. We shall use the methods of optimal control to maximize the cost functional, thereby maximizing the final mass and solving the problem.

II. 2. *Differential constraint: System Dynamics*

We represent the spacecraft by a point mass and assume it to be a thrusting craft acted upon the by the aerodynamic drag and oblate-body gravity forces of a central body. We also represent the central body, or planet, as a point mass positioned at its own center of gravity. We restrict the problem to crafts of mass much smaller than that of the central body, allowing us to fix the planet in inertial space. We shall describe this inertial space with a rectangular Cartesian inertial reference frame. All motion within this frame of reference agreeing with the above assumptions must satisfy the well-known Newton's equation:

$$\vec{F} = \frac{d(m\vec{v})}{dt} \tag{2}$$

where m is the spacecraft mass and $\vec{v}$ is its velocity with respect to the reference frame.

In this case, gravity, drag, and thrust make up the total force acting on the craft. The thrust on the craft is composed of two separated thrusts, the pressure thrust and the thrust created by the expulsion of mass. That is,

$$\vec{F} \equiv \vec{F}_{\text{pressure thrust}} - \vec{F}_{\text{drag}} - \vec{F}_{\text{gravity}} = m\dot{\vec{v}} - \dot{m}\vec{v}_e \tag{3}$$

where $\vec{v}_e$ is the expulsive velocity of mass. Therefore,

$$\vec{F}_{\text{total thrust}} \equiv \vec{F}_{\text{pressure thrust}} + \dot{m}\vec{v}_e \tag{4}$$

and

$$m\dot{\vec{v}} = \vec{F}_{\text{thrust}} - \vec{F}_{\text{drag}} - \vec{F}_{\text{gravity}} \tag{5}$$

We write the total thrust, herein referred to as just thrust, as some time-varying magnitude, T, independent of a time-varying direction, $\vec{e}$:

$$\vec{F}_{thrust} = T\vec{e} \tag{6}$$

Note that $\vec{e}$ is expressed as a unit vector. For a three dimensional thrust vector the control requires three components. For two dimensional problems only two independent control components are required.

The mass will decrease according to

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{7}$$

We assume that the atmosphere surrounding the central body can be described by an exponential model of the standard atmosphere[2]. The following equation[3] describes such a drag force:

$$\vec{F}_{drag} = \frac{1}{2}\rho_o e^{-\beta(r-r_o)} S C_D |\vec{v}|^2 \frac{\vec{v}}{|\vec{v}|} = \frac{1}{2}\rho_o e^{-\beta(r-r_o)} S C_D |\vec{v}| \vec{v} \tag{8}$$

where β and r_o are constants from the atmosphere model describing air density variation in the prescribed altitude region, ρ_o is the atmosphere density for the altitude r_o, S is the wetted area of the craft, C_D is the craft's drag coefficient, and $\vec{v}$ is craft's current velocity with respect to the inertial reference frame. We are assuming that no matter the orientation of the craft the product of SC_D remains the same and that the craft always remains in a region where the chosen exponential atmosphere model is valid.

Within the confines of this study, the only other influence on the craft is gravitational potential energy. The gravitational potential energy to the second harmonic is[4]:

$$U = -\frac{m\mu}{r} - \frac{1}{3} J R^2 \frac{m\mu}{r^3}(1 - 3\cos^2\theta) \tag{9}$$

Where R is the equatorial radius of the central body, θ is the latitude angle of the current position from the equator, and r is the distance from the central

body's center of gravity to the current position of the craft with respect to the inertial reference frame, μ is the gravitational constant for the central body, m is the mass of spacecraft, and J is a constant describing the mass distribution of the planet. There are additional mass distribution terms but we shall truncate the series here.

We now assume that (1) the central body is fixed at the center of the reference frame and (2) that the plane of the central body's equator is aligned with the x-y plane. The assumption (1) means that the position, velocity, and acceleration of the craft are now measured with respect to the central body. The assumption (2) means that we may describe r with Cartesian coordinates by

$$r = \sqrt{x^2 + y^2 + z^2} \tag{10}$$

and we may describe θ with Cartesian coordinates by

$$z = r \cos \theta \tag{11}$$

We may now write the gravitation potential as

$$U = - \frac{m\mu}{r} - \frac{1}{3} J R^2 \frac{m\mu}{r^3} \left(1 - 3 \left(\frac{z}{r} \right)^2 \right) \tag{12}$$

The force experienced by moving in a potential field U is $\frac{\partial U}{\partial \vec{r}}$. Performing this operation on the gravitational potential yields

$$\vec{F}_{gravity} = \left(\frac{\partial U}{\partial \vec{r}} \right)^T \tag{13}$$

and

$$\frac{\partial U}{\partial x} = \frac{m\mu}{r^3}x + J R^2 \frac{m\mu}{r^5}x\left(1 - 5\left(\frac{z}{r}\right)^2\right) \tag{14}$$

$$\frac{\partial U}{\partial y} = \frac{m\mu}{r^3}y + J R^2 \frac{m\mu}{r^5}y\left(1 - 5\left(\frac{z}{r}\right)^2\right) \tag{15}$$

$$\frac{\partial U}{\partial z} = \frac{m\mu}{r^3}z + J R^2 \frac{m\mu}{r^5}z\left(3 - 5\left(\frac{z}{r}\right)^2\right) \tag{16}$$

All of the dynamics combines to form the following equations of motion

$$m\ddot{x} = T e_x - \frac{m\mu}{r^3}x - J R^2 \frac{m\mu}{r^5}x\left(1 - 5\left(\frac{z}{r}\right)^2\right) - \frac{1}{2}\rho_o e^{-\beta(r-r_o)} S C_D v \dot{x} \tag{17a}$$

$$m\ddot{y} = T e_y - \frac{m\mu}{r^3}y - J R^2 \frac{m\mu}{r^5}y\left(1 - 5\left(\frac{z}{r}\right)^2\right) - \frac{1}{2}\rho_o e^{-\beta(r-r_o)} S C_D v \dot{y} \tag{17b}$$

$$m\ddot{z} = T e_z - \frac{m\mu}{r^3}z - J R^2 \frac{m\mu}{r^5}z\left(3 - 5\left(\frac{z}{r}\right)^2\right) - \frac{1}{2}\rho_o e^{-\beta(r-r_o)} S C_D v \dot{z} \tag{17c}$$

which can be written in vector-matrix form as

$$\ddot{\vec{r}} = \frac{T}{m}\vec{e} - \frac{\mu}{r^3}\vec{r} - \left\{ J R^2 \frac{\mu}{r^5} \begin{bmatrix} 1 - 5\left(\frac{z}{r}\right)^2 & 0 & 0 \\ 0 & 1 - 5\left(\frac{z}{r}\right)^2 & 0 \\ 0 & 0 & 3 - 5\left(\frac{z}{r}\right)^2 \end{bmatrix} \right\} \vec{r} - \frac{1}{2}\frac{\rho_o}{m} e^{-\beta(r-r_o)} S C_D v \dot{\vec{r}} \tag{18}$$

To conform to convention we make the change from J to J_2 as described in ref(4):

$$\ddot{\vec{r}} = \frac{T}{m}\vec{e} - \frac{\mu}{r^3}\vec{r} - \left\{\frac{3}{2}\mu J_2 \frac{R^2}{r^5}\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix} - 5\left(\frac{z}{r}\right)^2\right)\right\}\vec{r} - \frac{1}{2}\frac{\rho_o}{m} e^{-\beta(r-r_o)} S C_D v \dot{\vec{r}} \tag{19}$$

This can also be rewritten as a first-order system:

$$\dot{\vec{r}} = \vec{v} \tag{20a}$$

$$\dot{\vec{v}} = \frac{T}{m}\vec{e} - \frac{\mu}{r^3}\vec{r} - \left\{\frac{3}{2}\mu J_2 \frac{R^2}{r^5}\left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix} - 5\left(\frac{z}{r}\right)^2\right)\right\}\vec{r} - \frac{1}{2}\frac{\rho_o}{m} e^{-\beta(r-r_o)} S C_D v \vec{v} \tag{20b}$$

III. The First-Order Necessary Conditions

All problems herein perform a maximization of the final mass. Now, in order to write the adjoined cost functional we need to know what is included in the state, in the control, and what the constraints on these are.

First, however, we note that the problems herein are also free-final-time problems. The three differential equations above are written with respect to the independent variable t (time). For ease in numerical methods, we want to transfer the problem from free- to fixed- final time. This means that we must define a new independent variable τ (non-dimensional time) to be used in the place of t (dimensional time). This allows t_f to become a state variable. We make the following scaling:

$$t = t_f \tau \tag{21}$$

Therefore, to translate this to a fixed-time problem, t_f must multiply the derivatives of the states. The dot above a variable now means a derivative with respect to τ.

We know what to include in the state, $\vec{x}(\tau)$:

$$\vec{x}(\tau) = \begin{bmatrix} \vec{r}^T(\tau) & \vec{v}^T(\tau) & m(\tau) & t_f \end{bmatrix}^T \tag{22}$$

We also know that our state is confined by the system dynamics so that

$$\dot{\vec{x}}(\tau) = t_f \begin{bmatrix} \vec{v} \\ \frac{T}{m}\vec{e} - \frac{\mu}{r^3}\vec{r} - \left\{ \frac{3}{2}\mu J_2 \frac{R^2}{r^5} \left(\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{bmatrix} - 5\left(\frac{z}{r}\right)^2 \right) \right\} \vec{r} - \frac{1}{2}\frac{\rho_o}{m} e^{-\beta(r - r_o)} S C_D v \vec{v} \\ -\frac{T}{g_o I_{sp}} \\ 0 \end{bmatrix} \tag{23}$$

for all time $\tau \in [0,1]$. This is the differential constraint of the control problem.

The thrust magnitude has both an upper and a lower bound. The upper bound we shall call T_{max}, the lower bound is obviously zero. We, therefore, also have an inequality constraint that must be satisfied for all time $\tau \in [0,1]$:

$$(T - T_{max})T \leq 0 \tag{24}$$

and Eqn (25) can be rewritten as an equality constraint

$$(T - T_{max})T + \alpha^2 = 0 \tag{25}$$

where α is a slack variable, free to change with time. Finally, we need to specify the terminal orbits. We will do so by writing a vector equation

$$\vec{\psi}(\vec{x}(0),\vec{x}(1)) = 0 \tag{26}$$

that is only satisfied when our initial and final states both lie on their respective orbits.

Now we know enough to write the adjoined cost functional for this problem:

$$J = m(1) + \vec{\nu}^T\vec{\psi}(\vec{x}(0),\vec{x}(1)) + \int_0^1 \left\{\vec{\lambda}^T\left[\vec{f}(\vec{x}(t),\vec{u}(t)) - \dot{\vec{x}}\right] + \mu\left[(T - T_{max})T + \alpha^2\right]\right\} d\tau \qquad (27)$$

where m(1) is the final mass and $\vec{\psi}(\vec{x}(0),\vec{x}(1)) = 0$ represents the boundary conditions.

The $\vec{\lambda}$ shown in the cost functional is the costate vector, also called the Lagrange multipliers. This vector will be of the same dimensions as the state. For simplification's sake, we will segment this vector as follows:

$$\vec{\lambda}(t) = \begin{bmatrix} \vec{\lambda}_r^T(t) & \vec{\lambda}_v^T(t) & \lambda_m(t) & \lambda_{t_f} \end{bmatrix}^T \qquad (28)$$

Also, in Eqn (27), $\vec{\nu}$ is the Lagrange multiplier corresponding to the boundary conditions $\vec{\psi}$. $\vec{\nu}$ is a constant in time.

III.1. *The Hamiltonian*

With the pertinent dynamics defined, we are now able to write the Hamiltonian for the system. We take the Hamiltonian from the cost functional as

$$H(\vec{x}(t),\vec{u}(t)) = \vec{\lambda}^T\left[\vec{f}(\vec{x}(t),\vec{u}(t))\right] + \mu\left[(T - T_{max})T + \alpha^2\right] \qquad (29)$$

A major simplification can be made now. Notice that, excluding the constraint on the thrust, the Hamiltonian is linear with respect to the control T (but we shall see it is not linear with respect to $\vec{e}$):

$$H(\vec{x}(t),\vec{u}(t)) = \frac{T}{m}\vec{\lambda_v}^{T}\vec{e} - \lambda_m\frac{T}{g_o I_{sp}} + \cdots \tag{30}$$

This, in conjunction with the structure of the thrust constraint, means that we may assume that this is a 'bang-bang' control problem. Enough is known about this type of problem so that we may do the following:

(1) Define a new Hamiltonian that differs from the original only in the omission of the thrust constraint.

$$H(\vec{x}(t),\vec{u}(t)) = \vec{\lambda}^{T}\left[\vec{f}(\vec{x}(t),\vec{u}(t))\right] \tag{31}$$

(2) Establish what will be called the switching function. In general, the switching function is defined by the partial derivative of the Hamiltonian with respect to the control by which it is linear. For this problem, This is done by evaluating $\frac{\partial H}{\partial T}$:

$$\frac{\partial H}{\partial T} = H_T = \frac{\vec{\lambda_v}^{T}\vec{e}}{m} - \frac{\lambda_m}{I_{sp}\, g_o} \tag{32}$$

This, Eqn (32), is the switching function.

(3) Evaluate a restricted case of the well-known Euler-Lagrange equations. Most of these determine the costate dynamics and we shall see these in section III.2, however, the last one determines part of the control for the problem. This equation is

$$\frac{\partial H}{\partial \vec{e}} = 0 \tag{33}$$

Evaluating (33), it appears that H is linear with respect to $\vec{e}$. However, we must remember that $\vec{e}$ represents only the direction of the thrust. If we

exchange $\vec{e}$ for some angle θ and define this angle as between $\vec{e}$ and $\vec{\lambda}_v$ we may write

$$H = \frac{T}{m} \vec{\lambda}_v^T \vec{e} + \cdots = \frac{T}{m} |\vec{\lambda}_v| |\vec{e}| \cos\theta + \cdots \tag{34}$$

$$H = \frac{T}{m} |\vec{\lambda}_v| \cos\theta + \cdots \quad \text{since} \quad |\vec{e}| = 1 \tag{35}$$

Evaluating H_θ we find that

$$\frac{\partial H}{\partial \theta} = -\frac{T}{m} |\vec{\lambda}_v| \sin\theta \tag{36}$$

and this equals zero only when the vectors are parallel. There are only two choices for $\vec{e}$: in the direction of $\vec{\lambda}_v$ or in the exact opposite direction. Since we are maximizing the final vehicle mass, we need to have $H_{\theta\theta}$ negative (one of the sufficient conditions for the second variation). This is only satisfied with $\vec{e}$ in the direction of $\vec{\lambda}_v$ or

$$\vec{e} = \frac{\vec{\lambda}_v}{|\vec{\lambda}_v|} \tag{37}$$

We must obey this for all time $\tau \in [0,1]$. This result is consistent with that of Lawden's primer vector[5].

(4) Perform bang-bang control with T. This means that T is always on-boundary, i.e. T=0 or $T=T_{max}$ at any $\tau \in [0,1]$. We know which value to use for T by evaluating the switching function, which we can now write as

$$H_T = \frac{|\vec{\lambda}_v|}{m} - \frac{\lambda_m}{I_{sp}\, g_o} \tag{38}$$

The bang-bang control law is

$$H_T \geq 0 \quad T = T_{max}$$
$$H_T < 0 \quad T = 0 \tag{39}$$

This switching structure satisfies the Pontryagin maximum principle by maximizing the Hamiltonian using T.

III.2. *The Costates*

The costate dynamics can be found from the following Euler-Lagrange equations, relating them to the Hamiltonian:

$$\dot{\vec{\lambda}}_r = -\frac{\partial H^T}{\partial \vec{r}} \tag{40}$$

$$\dot{\vec{\lambda}}_v = -\frac{\partial H^T}{\partial \vec{v}} \tag{41}$$

$$\dot{\lambda}_m = -\frac{\partial H}{\partial m} \tag{42}$$

To evaluate these, we must first substitute the equations of motion into the Hamiltonian

$$H = t_f \{ \vec{\lambda}_r^T \vec{v} + \vec{\lambda}_v^T [\frac{T}{m} \vec{e} - [\frac{\mu}{r^3} + \frac{3}{2} J_2 \frac{\mu}{r^5} R^2 \overline{N} -$$

$$- \frac{15}{2} J_2 \frac{\mu}{r^7} R^2 (z^2)] \vec{r} - \frac{1}{2} \frac{\rho_o}{m} e^{-\beta (r - r_o)} S C_D v \vec{v}] \tag{43}$$

$$+ \lambda_m \frac{T}{g_o I_{sp}} \}$$

When evaluated, these become the following vector and scalar differential equations:

$$\dot{\vec{\lambda}}_r = t_f \{ \mu \left[\frac{\vec{\lambda}_v}{r^3} - 3 \frac{\left(\vec{\lambda}_v^T \vec{r}\right)\vec{r}}{r^5} \right] - \frac{1}{2} \frac{\rho_o}{m} \frac{\beta}{r} e^{-\beta (r - r_o)} S C_D v \left(\vec{\lambda}_v^T \vec{v}\right)\vec{r}$$

$$+ \frac{3}{2} J_2 \mu R^2 \left[\frac{\bar{N} \vec{\lambda}_v}{r^5} - 5 \frac{\left(\vec{\lambda}_v^T \bar{N} \vec{r}\right)\vec{r}}{r^7} \right] - \frac{15}{2} J_2 \mu R^2 \left[\frac{z^2}{r^7} \vec{\lambda}_v - \left(\vec{\lambda}_v^T \vec{r}\right)\left(\frac{7 z^2}{r^9} \vec{r} - \frac{2 z}{r^7} \vec{k} \right) \right] \} \quad (44)$$

where $\vec{k} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$

$$\dot{\vec{\lambda}}_v = t_f \left[- \vec{\lambda}_r^T + \frac{1}{2} \frac{\rho_o}{m} e^{-\beta (r - r_o)} S C_D \left(\vec{\lambda}_v v + \frac{\left(\vec{\lambda}_v^T \vec{v}\right)\vec{v}}{v} \right) \right] \quad (45)$$

$$\dot{\lambda}_m = t_f \left[- \frac{T}{m^2} \vec{\lambda}_v^T \vec{e} + \frac{1}{2} \frac{\rho_o}{m^2} e^{-\beta (r - r_o)} S C_D v \vec{\lambda}_v^T \vec{v} \right] \quad (46)$$

IV. SOLVED PROBLEMS

IV.1. *Simplifications*

We have made a few simplifications that ease the formulation of the numerical problem and its solution. The first of these is the reduction from a three-dimensional problem to a two-dimensional problem. To remove this dimension, we simply remove the z-component to all equations. Because of the chosen coordinate system, this also means that all orbit transfers considered are equatorial. Unfortunately, the effect of oblateness is substantially decreased for this case. The other simplification is the restriction

of problems to fixed-initial points. This also greatly eases the problem formulation. The third and final simplification is the fixing of the final time.

IV.2. *The Two-Point Boundary Value Problem*

As a result of the simplifications, the boundary conditions have been stated in two dimensions. The starting orbit determines the initial conditions on position and velocity. The final conditions, however, require a more abstract specification as we do not know exactly at what point the craft will enter this orbit. The following relations specify the final orbit: (All of the following conditions is to be evaluated at the final time, t_f, or τ=1.)

$$\text{(Angular Momentum)} \quad \psi 1: \left\{ \begin{array}{c} \vec{R} \times \vec{V} = \vec{H} \\ \langle x, y \rangle \times \langle u, v \rangle = x v - y u = h \\ x v - y u - h = 0 \end{array} \right\} \tag{47a}$$

$$\text{(eccentricity vector (x))} \quad \psi 2: \left\{ \begin{array}{l} e_x = \frac{1}{\mu}\left[\left(V^2 - \frac{m}{r}\right) x - \left(\vec{r}^T \vec{V}\right) u\right] \\ \\ \frac{1}{\mu}\left[\left(V^2 - \frac{m}{r}\right) x - \left(\vec{r}^T \vec{V}\right) u\right] - e_x = 0 \end{array} \right\} \tag{47b}$$

$$\text{(eccentricity vector (y))} \quad \psi 3: \left\{ \begin{array}{l} e_y = \frac{1}{\mu}\left[\left(V^2 - \frac{m}{r}\right) y - \left(\vec{r}^T \vec{V}\right) v\right] \\ \\ \frac{1}{\mu}\left[\left(V^2 - \frac{m}{r}\right) y - \left(\vec{r}^T \vec{V}\right) v\right] - e_y = 0 \end{array} \right\} \tag{47c}$$

Note that the orbit equation for x-axis aligned orbits and the energy equation can replace (47b) and (47c). However, the combined constraints of angular, momentum, orbit, and energy equations do not uniquely specify an

x-axis aligned final orbit. There are two possible velocity vectors at one point with the same angular momentum and energy.

These conditions completely determine the final orbit. However, these conditions do not complete the two-point boundary-value problem. To complete the TPBVP, the methods of optimal control supply use with a set of natural boundary conditions found by evaluating

$$\vec{\lambda}(1) = \left(\frac{\partial G}{\partial x(1)}\right)^T \tag{48}$$

where G is the constructed from the function portion of the cost functional, e.g. for the cost functional

$$J = m(1) + \vec{\nu}^T\vec{\psi}(\vec{x}(0),\vec{x}(1)) + \int_0^1 \left\{\vec{\lambda}^T\left[\vec{f}(\vec{x}(t),\vec{u}(t)) - \dot{\vec{x}}\right] + \mu\left[(T - T_{max})T + \alpha^2\right]\right\} d\tau \tag{49}$$

G is

$$G = m(1) + \vec{\nu}^T\vec{\psi}(\vec{x}(0),\vec{x}(1)) \tag{50}$$

Constructing G with the above conditions on the states, we can find conditions on the costates at τ=1:

$$G = m + \nu_1(x\,v - y\,u - h) + \begin{bmatrix} \nu_2 & \nu_3 \end{bmatrix}\left[\frac{1}{\mu}\left(\left(V^2 - \frac{m}{r}\right)\vec{r} - \left(\vec{r}^T\vec{V}\right)\vec{V}\right) - \vec{e}\right] \tag{51}$$

evaluating Eqn(49) gives

$$\lambda_x = \frac{\partial G}{\partial x} = \nu_1(v) + \nu_2\left(V^2 - \frac{\mu}{r} + \frac{x^2}{r^3} - \frac{u^2}{\mu}\right) + \nu_3\left(\frac{x\,y}{r^3} - \frac{u\,v}{\mu}\right) \tag{52a}$$

$$\lambda_y = \frac{\partial G}{\partial y} = \nu_1(-u) + \nu_2\left(\frac{x\,y}{r^3} - \frac{u\,v}{\mu}\right) + \nu_3\left(V^2 - \frac{\mu}{r} + \frac{y^2}{r^3} - \frac{v^2}{\mu}\right) \tag{52b}$$

$$\lambda_u = \frac{\partial G}{\partial u} = \nu_1 (- y) + \nu_2 \left(\frac{- y v}{\mu} \right) + \nu_3 \frac{1}{\mu} (2 y u - x v) \tag{52c}$$

$$\lambda_v = \frac{\partial G}{\partial v} = \nu_1 (x) + \nu_2 \frac{1}{\mu} (2 x v - u y) + \nu_3 \left(\frac{- x u}{\mu} \right) \tag{52d}$$

$$\lambda_m = \frac{\partial G}{\partial m} = 1 \tag{52e}$$

note that the constant Lagrange multipliers ν_i are additional unknown's.

The last condition deals with the final time. If the final time were free we would use the transversality condition

$$H(\vec{x}(1),\vec{u}(1),\vec{\lambda}(1)) = - \frac{\partial G}{\partial t_f} \tag{52f}$$

or, for this problem

$$H(\vec{x}(1),\vec{u}(1),\vec{\lambda}(1)) = 0 \tag{52g}$$

However, all the solutions presented in this report are fixed-final time. Note, however, that the same algorithm can be used for both types of problems, all that is required is that equation (53g) be replaced by the specification of t_f.

IV.3. *Non-Dimensionalization*

To improve accuracy, we have non-dimensionalized the problem. This aids in a few ways. First, the integration of the state is more accurate because all variations are on the same order. Second, convergence is improved because all the boundary conditions are immediately placed at or near the same order. Our non-dimensionalized parameters are as follows:

$$\vec{\mathbf{r}} \equiv \frac{\vec{r}}{r^{\star}} \tag{53a}$$

$$\vec{\mathbf{V}} \equiv \frac{\vec{V}}{\sqrt{\frac{\mu}{r^{\star}}}} \tag{53b}$$

$$\mathbf{m} \equiv \frac{m}{m^{\star}} \tag{53c}$$

$$\mathbf{t_f} \equiv \frac{t_f}{\sqrt{\frac{r^{\star 3}}{\mu}}}. \tag{53d}$$

and they require the following

$$\mathbf{T} \equiv \frac{T / m^{\star}}{\mu / r^{\star 2}} \tag{53e}$$

$$(\mathbf{g_o\, I_{sp}}) \equiv (g_o\, I_{sp}) \sqrt{\frac{r^{\star}}{\mu}} \tag{53f}$$

$$\mathbf{r_o} \equiv \frac{r_o}{r^{\star}} \tag{53g}$$

$$\beta \equiv \beta\, r^{\star} \tag{53h}$$

$$(\rho_o \mathbf{C_D S}) \equiv (\rho_o C_D S) \frac{r^{\star}}{m^{\star}} \tag{53i}$$

$$\mathbf{R} \equiv \frac{R}{r^{\star}} \tag{53j}$$

The choices of $r^\star$ and $m^\star$ are completely arbitrary. However, it needs to be said that after a problem is solved by these nondimensionalizations, rescaling must be excersized with caution. This is a direct result of the atmosphere model; if the rescaling is not consistent with the atmosphere model, the results are invalid, e.g. rescaling also rescales the atmosphere model (note Eqn (8)).

If we solve Eqs (53a-j) such that the dimensional parameter is on the left-hand side and then substitute into the original dynamics we find equations that are *exactly* equal to the original equations with μ=1 (The value of J_2, however, has no dimensions and is not changed). This can be extended to the boundary equations and the costate differential equations. A special note is required for the costates: the costates resulting from the solution to the problem with this transformation will be some scalar multiplied by the 'dimensional' costates, e.g.

$$\vec{\lambda} \equiv \frac{\vec{\lambda}}{\lambda^\star} \tag{53k}$$

which requires

$$\vec{\nu} \equiv \frac{\vec{\nu}}{\lambda^\star} \tag{53l}$$

where $\lambda^\star$ is completely arbitrary. This is easily verified by substitution into the differential equations and boundary conditions.

IV.4. Atmosphere Model

Any atmosphere is usable by simple substitution early in the derivation of the differential constraint. For the purposes of this report we have chosen a very simple atmosphere model. The model is not intended to accurately represent the Earth's atmosphere, or any other planet for that

matter. It is implemented only for the purpose of demonstrating the methods for solving the optimization problem.

Our model is defined at 450km altitude above the planet's equator. The entire atmosphere is assumed isothermal. The temperature is 1000K. The density at the definition altitude is 1.184×10^{-12} kg/m^3. This definition point for this model is taken from the 1976 U.S. Standard Atmosphere[6]. The atmosphere is assumed spherical above the oblate planet. For real-world solutions, we strongly recommend the use of the latest standard atmosphere or some appropriate approximation thereof. The contemporary standard atmosphere can be found in ref (7).

IV.5. *The Multiple Point Shooting Method of BOUNDSCO*

One method we are currently using to attempt to solve the TPBVP is the multiple point shooting method. The specific algorithms we are currently using are those given by H. J. Oberle in his subroutine BOUNDSCO[7], written in FORTRAN. His method, a complete description of which can be found in ref(8), is a modification of the traditional well-known multiple point shooting method.

The use of this method requires the writing of a few routines that define the problem. These routines include, of course, the calling program itself, a subroutine defining the differential constraint (or system dynamics), and a subroutine that defines the constraints on the problem.

The state used in BOUNDSCO differ slightly from the state defined in this report. We have simply adjoined the $\vec{\nu}$ vector to the state. This requires also that the system dynamics includes a corresponding number of zero derivatives. We justify for this by noting that it allows the statement of the absolute and natural boundary conditions exactly as they are in this report. If we did not implement this, we would have to solve the system of three of the natural boundary conditions for the $\vec{\nu}$ and substitute the result into the fourth equation, using it in place of the four. This may seem desirable, one equation in the place of four, however, the simple structure of the four

equations is much more desirable than the complex structure of the one equation.

There is one particular feature that makes BOUNDSCO attractive: the explicit inclusion of switching points in the problem formulation. Oberle allows the user to specify the switching function outside of the system dynamics. This simplifies integration and improves convergence. There is a tradeoff; the user must assume a switching structure and verify it outside of BOUNDSCO.

IV.6. *The Minimizing-Boundary-Condition Method*

The second method we are using is called the Minimizing-Boundary-Condition Method (MBCM)[8]. It is described in ref(9). This method is a modification to the shooting method. It expands the set of available solutions by removing one boundary-condition. The choice of this boundary-condition is arbitrary. Since there is a much larger set of solutions, it is much easier to solve the resulting boundary-value problem. Once this is accomplished, the search for the solution that incorporates the final boundary conditions is treated as a minimization problem. The gradient is numerically calculated and used to update the initial state until the last boundary condition is satisfied. This method is at least as effective as BOUNDSCO in solving the two-point boundary-value problems for the current solved optimal orbit transfers.

The switching structure of optimal control is included in MBCM. The program checks the switching function at each integration step. If the switching function alters sign at one integration step, the program stops the integration and restores all the states to the beginning of the step. A secant method then calculates a smaller step size for integrating the switching function to an exact zero point. From our experience with MBCM some sensitive problems need fourteen digits of accuracy in their switching function. Once the integration passes the switching points the program switches the control and uses a normal step size for integration.

IV.7. *Sample Problems and Solutions*

Several solutions are presented in this section, all of which both methods were able to converge. As a matter of fact, in most cases, the solution to one problem can be used as the guess to a different problem and the program(s) will converge. All problems have been nondimensionalized and use the atmosphere model presented above.

The first problem presented is a fixed-final-time circle-to-circle orbit transfer:

Find an extremal for the maximum final-mass problem which travels from a circular orbit of $a=3.847$ *at* $y=3.72$ *to another circular orbit of* $a=1.5$. *The available thrust is (a) 0.9, (b) 0.2 and* $g_0 I_{sp}=51.254$. *The initial mass is 1.527. The allowed time for transfer is 12.5.* $\rho_0 S C_D=3.894\times10^{-17}$.

The optimal trajectories are shown in Fig. 1 for T=0.9 and in Fig. 3 for T=0.2. Their switching functions are shown in Fig. 2 for T=0.9 and in Fig. 4 for T=0.2.

The second problem presented is a fixed-final-time apse-aligned ellipse-to-ellipse orbit transfer:

Find an extremal for the maximum final-mass problem which travels from an orbit of $a=3.847$ *and* $r_p=3.756$ *at* $y=3.76$ *to another orbit of* $a=1.5$ *and* $r_p=1$. *The apses of the orbits are aligned with the x-axis. The available thrust is (a) 0.9, (b) 0.2 and* $g_0 I_{sp}=51.254$. *The initial mass is 1.527. The allowed time for transfer is 12.* $\rho_0 S C_D=3.894\times10^{-17}$.

The optimal trajectories are shown in Fig. 5 for T=0.9 and in Fig. 7 for T=0.2. Their switching functions are shown in Fig. 6 for T=0.9 and in Fig. 8 for T=0.2.

The third problem presented is a fixed-final-time non-apse-aligned ellipse-to-ellipse orbit transfer:

Find an extremal for the maximum final-mass problem which travels from an orbit of a*=3.847 and* r_p*=3.756 at* y*=3.76 to another orbit of* a*=1.5 and* r_p*=1. The apses of the initial orbit is at an angle of 153° with the x-axis, clockwise. The apse of the final orbit is at an angle of 109° with the x-axis, counter-clockwise. The available thrust is (a) 0.9, (b) 0.2 and* $g_0 I_{sp}$*=51.254. The initial mass is 1.527. The allowed time for transfer is 10.* $\rho_0 S C_D = 3.894 \times 10^{-17}$.

The optimal trajectories are shown in Fig. 9 for T=0.9 and in Fig. 11 for T=0.2. Their switching functions are shown in Fig. 10 for T=0.9 and in Fig. 12 for T=0.2.

V. CONCLUSIONS

The performance of BOUNDSCO was mixed. The ability of the routine to converge solutions is quite strong, however, there is a flaw. BOUNDSCO does not supply information to the user's routine concerning the polarity of the switching function. The user must assume in all his/her code that the desired switching structure is correct. The result of this is that BOUNDSCO often allows itself to converge solutions with inconsistent switching functions. This would not be so bad, except for one other difficulty with BOUNDSCO: the routine does not attempt to aid the user in any way with the initial guess. For example, one finds it nearly impossible to converge a two-burn solution without the insight to guess an initial state that, when integrated, produces two crossings of the switching function (this is actually, not too difficult, if one pays attention to the sign of the switching function and its derivative when making guesses). However, when BOUNDSCO does produce correct solutions, they are as accurate as the user can specify. The solutions presented above satisfy their boundary conditions within 10^{-14} absolute error.

The performance of the minimizing-boundary condition method was also quite promising. This method has one distinct advantage over BOUNDSCO, it explicitly disallows inconsistent switching functions. The method checks the switching function during, but separately from, integration to determine where the switching points are and, most

importantly, what the switching function polarity is. This method is, however, currently a simple shooting method and it exhibits the difficulty of the same. It is expected that if the method is extended to a multiple-point shooting method, it's performance will rival, if not exceed that of BOUNDSCO.

And thereby we come to the recommendation of this study: the development of a method that is a hybrid of multiple-shooting and the minimizing-boundary-condition method.

---

[1]Edelbaum, T.N., "How Many Impulses?" Aeronautics and Astronautics, Nov. 1967.

[2]Anderson, J.D., Introduction to Flight, McGraw-Hill Book Co., New York, 1989.

[3]Anderson, J.D., Fundamentals of Aerodynamics, McGraw-Hill Book Co., New York, 1984.

[4]Space Technology Laboratories, Flight Performance Handbook for Orbital Operations, Wiley, New York, 1963.

[5]Lawden, D.F., Optimal Trajectories for Space Navigation, Butterworths, London, 1963.

[6]COESA, U.S. Standard Atmosphere, 1976, U.S. Government Printing Office, Washington, D.C. 1976

[7]Oberle, H. J., BOUNDSCO - Hinweise zur Benutzung des Mehrzielverfahrens für die numerische Lösung von Randwerproblemen mit Schaltbedingungen, Hamburger Beiträge zur Angewandten Mathematik, Berichte 6, 1987.

[8]Chuang, C.-H., and Speyer, J.L., "Periodic Optimal Hypersonic Scramjet Cruise," Optimal Control Applications & Methods, Vol. 8, pp. 231-242 (1987)

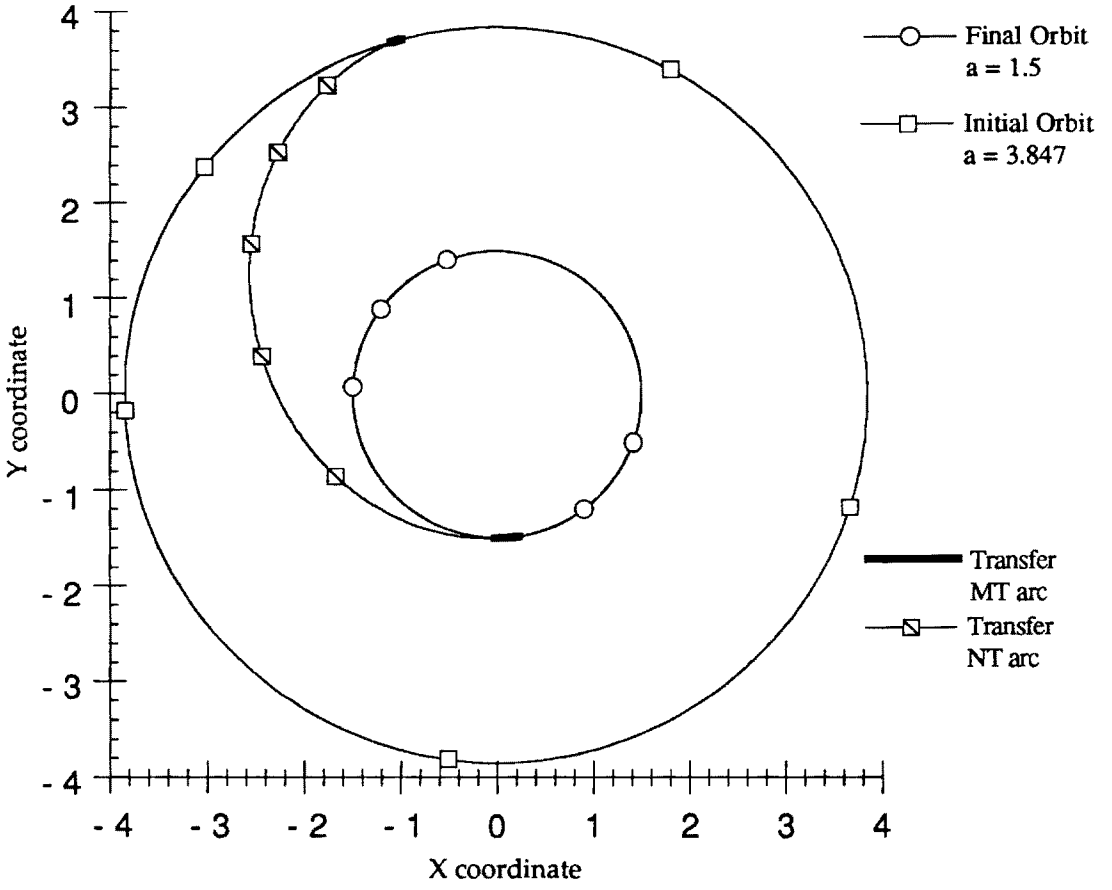


Fig 1 Mass-Optimal Circle-to-Circle Orbit Transfer, T=0.9

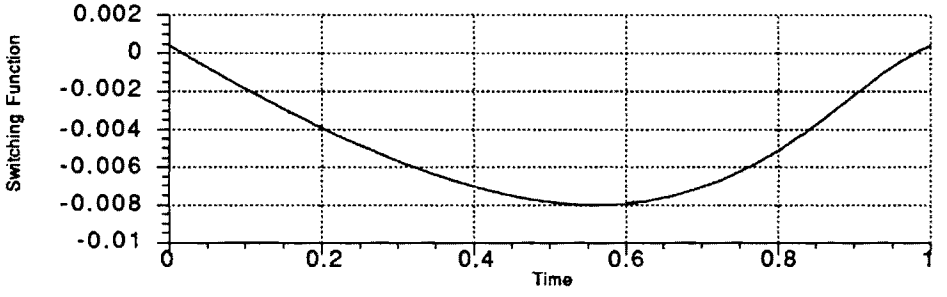


Fig. 2 Graph of Switching Function Corresponding to Figure 1, T=0.9

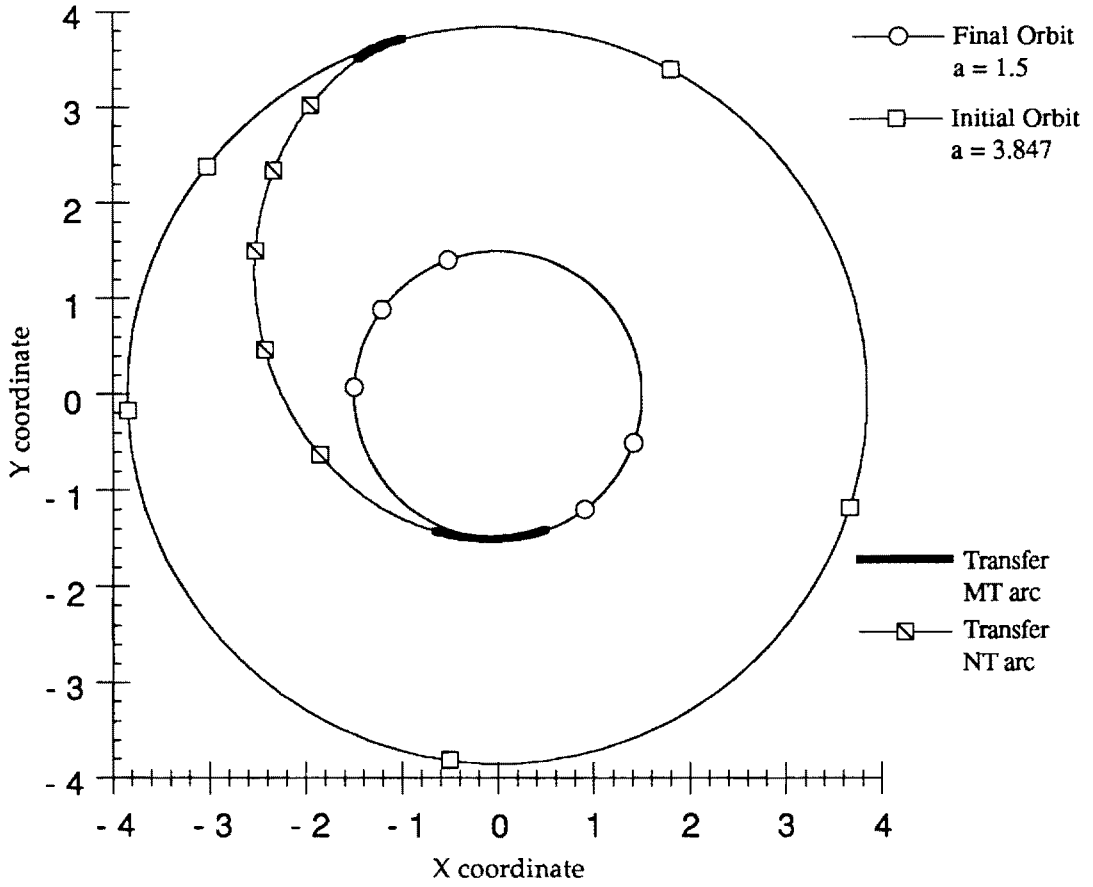


Fig 3 Mass-Optimal Circle-to-Circle Orbit Transfer, T=0.2

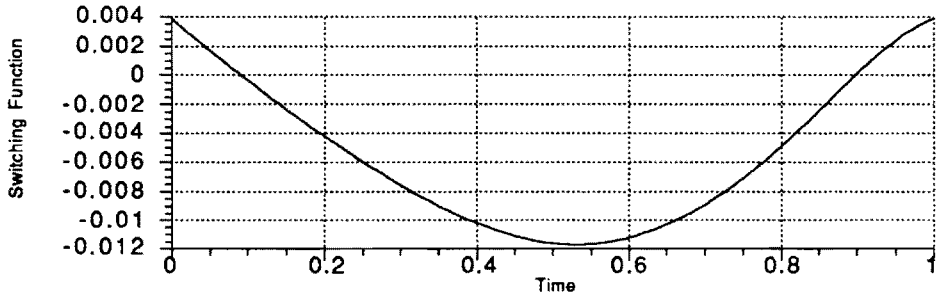


Fig. 4 Graph of Switching Function Corresponding to Figure 3, T=0.2

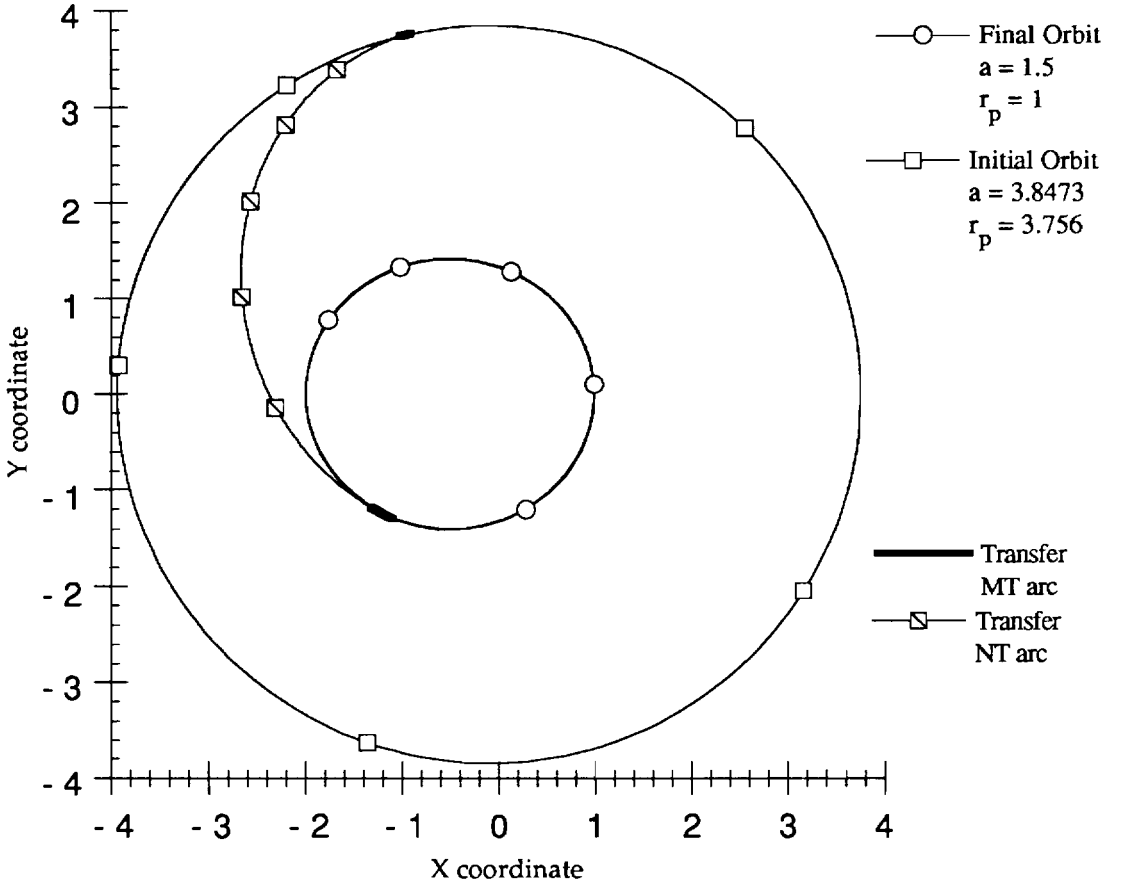


Fig. 5 Mass-Optimal Aligned-Ellipse-to-Ellipse Orbit Transfer, T=0.9

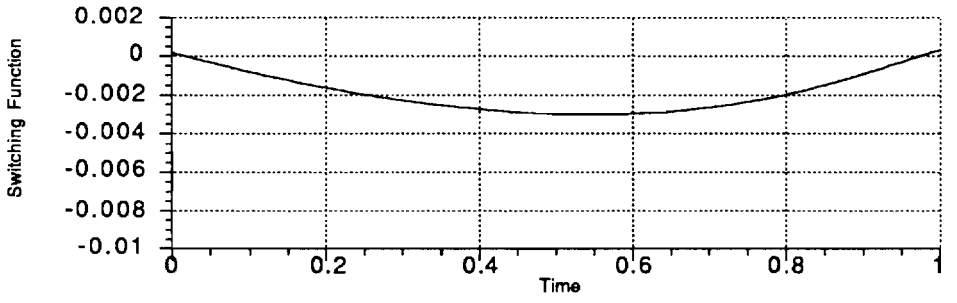


Fig. 6 Graph of Switching Function Corresponding to Figure 5, T=0.9

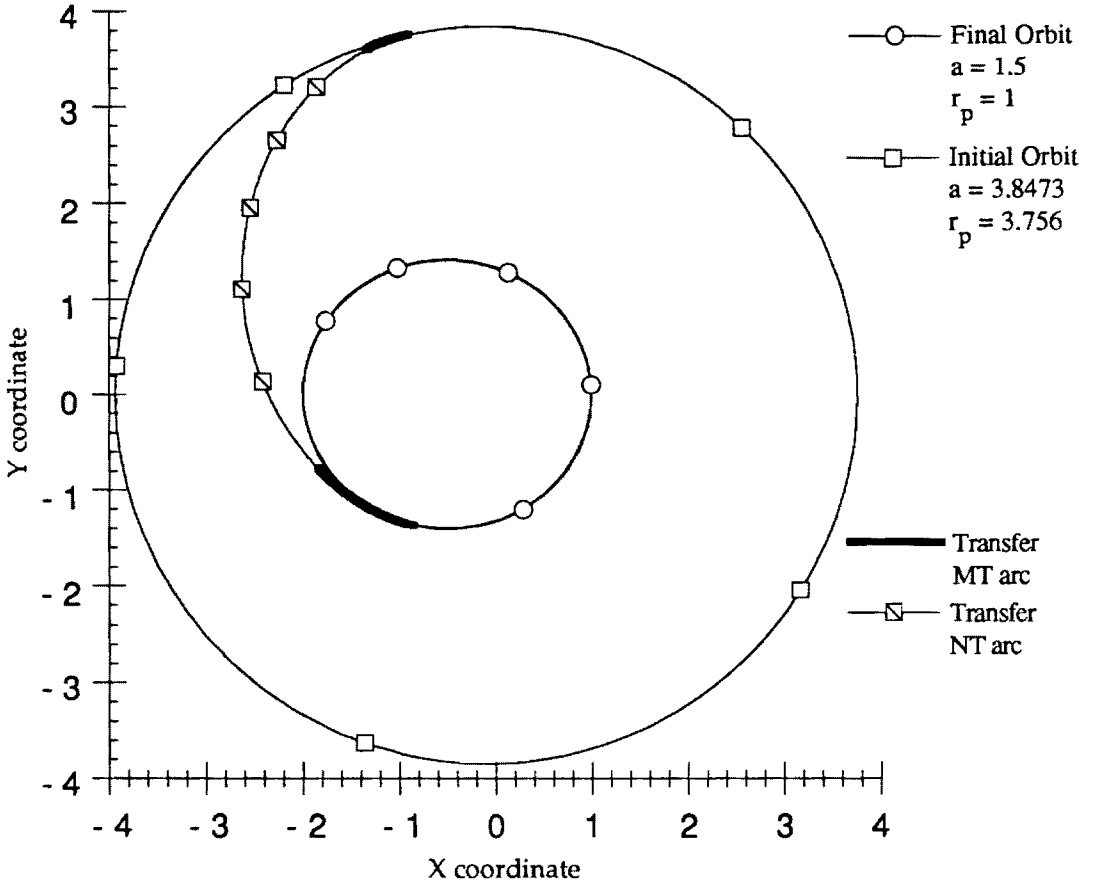


Fig. 7 Mass-Optimal Aligned-Ellipse-to-Ellipse Orbit Transfer, T=0.2

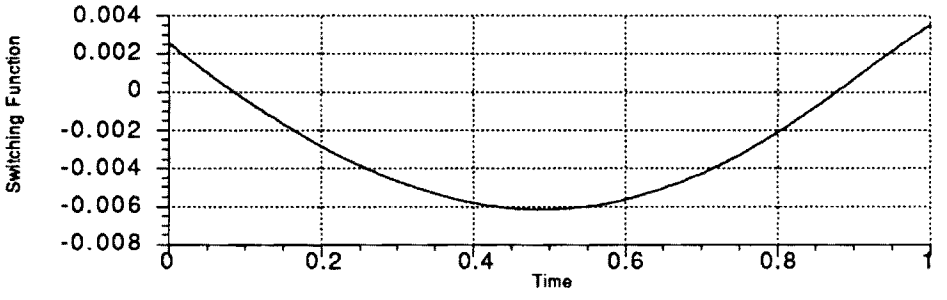


Fig. 8 Graph of Switching Function Corresponding to Figure 7, T=0.2

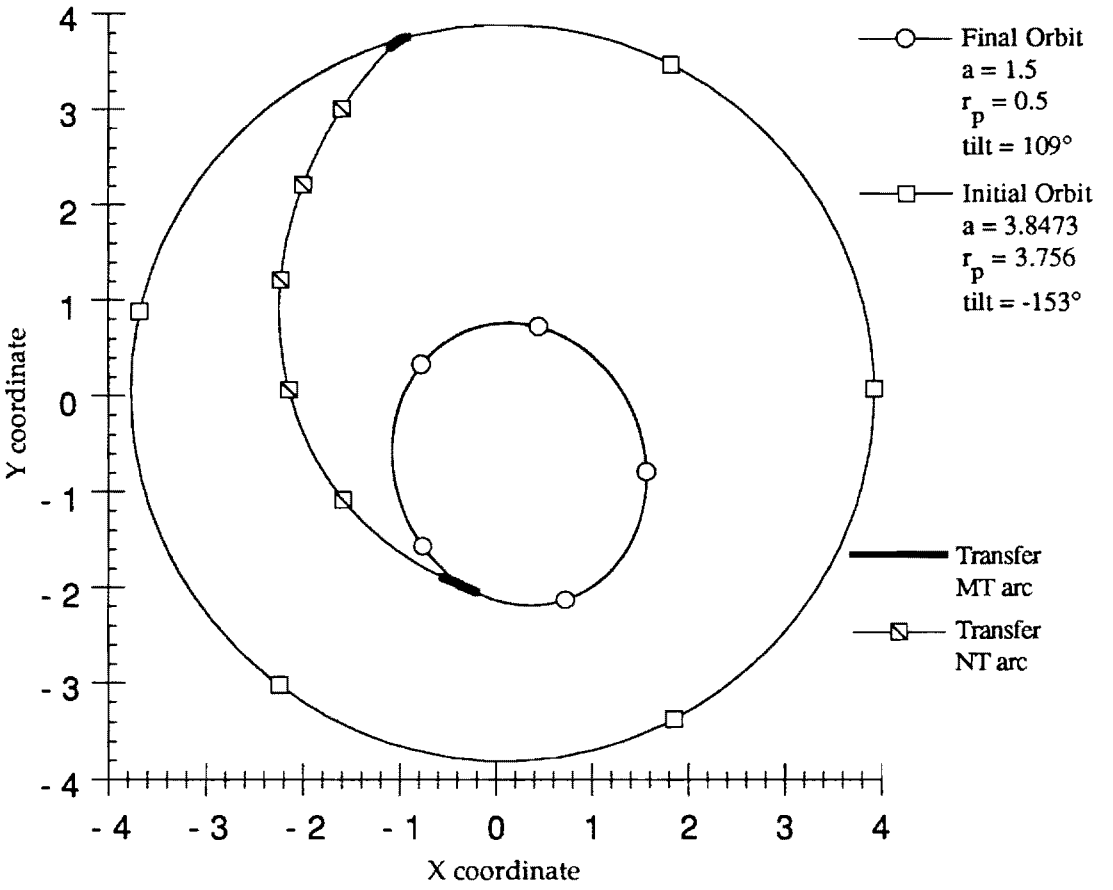


Fig. 9 Mass-Optimal Non-Aligned-Ellipse-to-Ellipse Orbit Transfer, T=0.9

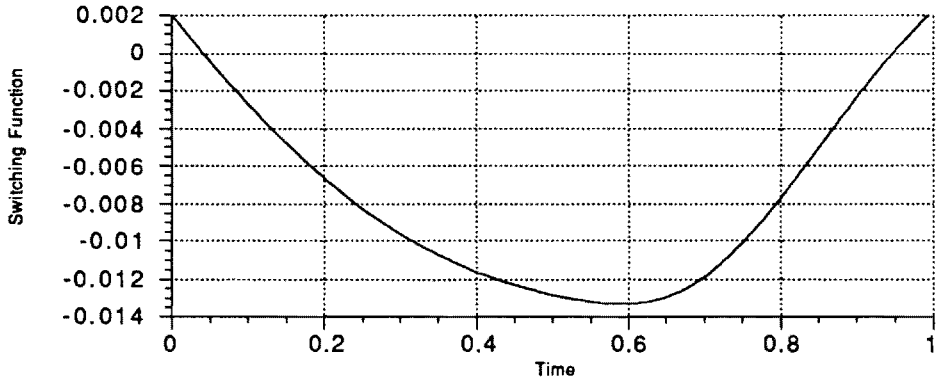


Fig. 10 Graph of Switching Function Corresponding to Figure 9, T=0.9

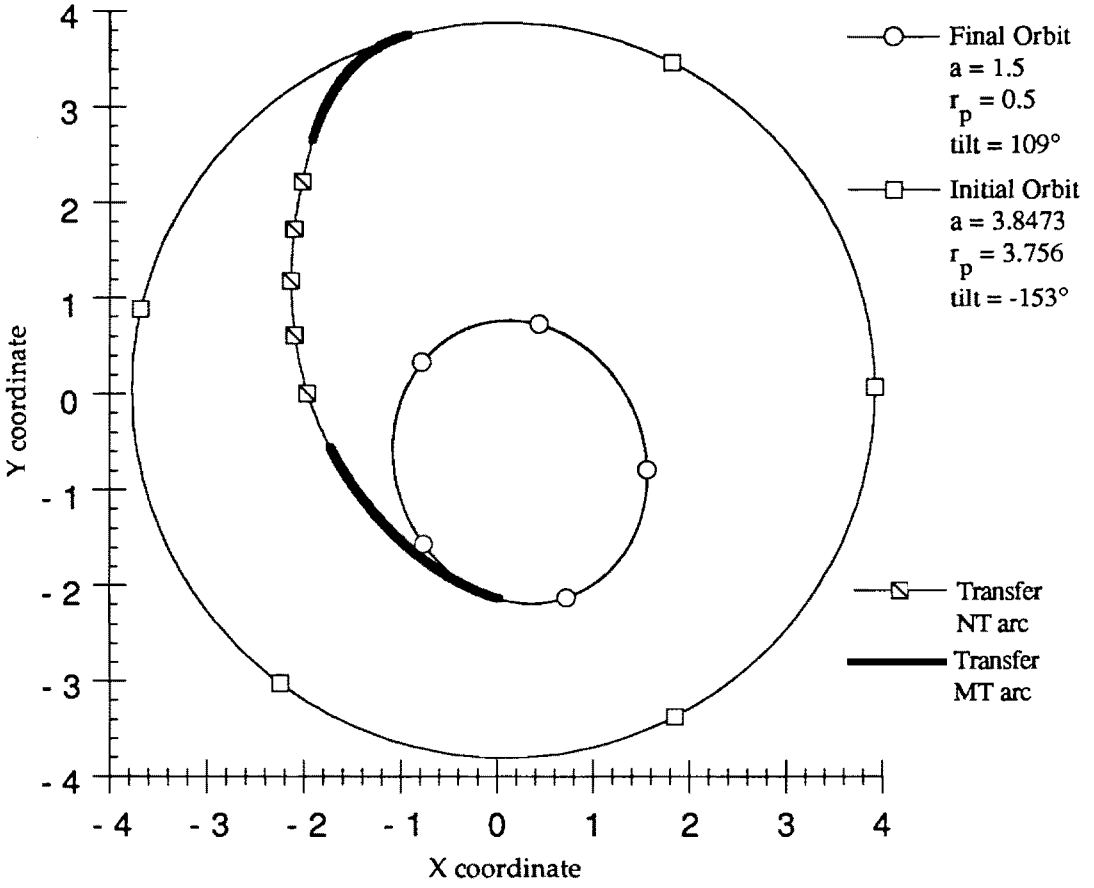


Fig. 11 Mass-Optimal Non-Aligned-Ellipse-to-Ellipse Orbit Transfer, T=0.2

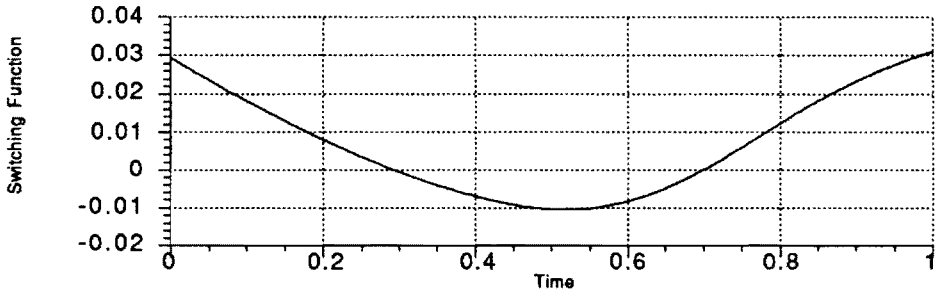


Fig. 12 Graph of Switching Function Corresponding to Figure 11, T=0.2

E16-1126

-11 2

SEMI-ANNUAL STATUS REPORT

Submitted to:	NASA Marshall Space Flight Center
Grant Title:	Theory and Computation of Optimal Low- and Medium-Thrust Transfers
Grant Number:	NAG8-921
Principal Investigator /Project Director:	Dr. C.-H. Chuang School of Aerospace Engineering Georgia Institute of Technology Atlanta, GA 30332-0150 (404) 894-3075
Research Assistant:	Troy Goodson School of Aerospace Engineering Georgia Institute of Technology
Period Covered:	January 7, 1993 to July 7, 1993
Date of Submission:	July 1, 1993

TABLE OF CONTENTS

SUMMARY OF THE RESEARCH RESULTS ii
ABSTRACT 1
I. INTRODUCTION 1
II. THE PROBLEM 3
II.1. THE COST FUNCTIONAL 3
II.2. DIFFERENTIAL CONSTRAINT: SYSTEM DYNAMICS 3
III. THE FIRST-ORDER NECESSARY CONDITIONS 5
III.1. THE HAMILTONIAN 6
III.2. THE COSTATES 8
IV. SOLVED PROBLEMS 9
IV.1. SIMPLIFICATIONS 9
IV.2. THE TWO-POINT BOUNDARY VALUE PROBLEM 9
IV.3. NON-DIMENSIONALIZATION 11
IV.4. ATMOSPHERE MODEL 12
IV.5. THE MULTIPLE POINT SHOOTING METHOD OF BOUNDSCO 13
IV.6. THE MINIMIZING-BOUNDARY-CONDITION METHOD 13
IV.7. OPTIMIZING THE INITIAL POINT 14
IV.8. WHAT IS THE OPTIMAL TRANSFER TIME 15
IV.9. SAMPLE PROBLEMS AND SOLUTIONS 17
V. CONCLUSIONS 21

SUMMARY OF RESEARCH RESULTS

1. The following report titled "Computation of Optimal Low- and Medium- Thrust Orbit Transfers" gives the detail of the research results. The report is an extension of the previous semi-annual report submitted January 6, 1993. The following is a summary of the extended results.
2. At the time of the last report we had formulated the optimal orbit transfer problem for the three-dimensional, time free cases with atmospheric drag effect and earth oblateness effect. Numerical solutions were obtained for two dimensional and fixed transfer time cases by using BOUNDSCO and MBCM. These transfers have a fixed initial orbit exit point and have thrust levels on the order of 10^{-1}. Following the submittal of that report we have done work on finding out how well the methods we have chosen (BOUNDSCO & MBCM) handle optimizing the initial orbit exit point and lowering the thrust level.
3. We have used a continuation method to optimize the initial orbit exit point which has worked satisfactorily in test cases. Specifically, this method has been used successfully for multiple examples in two completely different orbit classes: a two-burn descent transfer and a three-burn ascent transfer. In both classes, an optimal exit point was found without much difficulty.
4. During the effort of lowering the thrust level we have found that the TPBVP solvers require more time to do computations on lower thrust problems. However, the TPBVP solvers have so far been able to handle these problems. We have taken the family of solutions presented in our last report and reduced the thrust by at least a factor of 10, bringing us just above the desired 10^{-3} level.
5. Since the previous report we have also begun work on the free final time problem. This work has become connected with lowering the thrust level and increasing the number of burns. It seems that our previous hypothesis that the global minimum solution will be at infinite final time and local minimum solutions exist for finite final time is not completely true. Our current results indicate that at least for two burn transfers extremal solutions exist only up to a maximum transfer time. For a given final time longer than this maximum transfer time we could not obtain any extremal solution. However, the cost functional becomes very flat with a nearly zero derivative at the maximum transfer time.
6. The next major step is development of a guidance scheme so that a spacecraft will be able to follow an optimal space curve with some degree of accuracy. At this time we are considering using neighboring optimal feedback control. This approach will use a time-varying feedback control law. Our experience in this area indicates that this feedback controller may be very robust for the orbit transfer problems.
7. In the meantime, we are using the TPBVP solvers to produce lower thrust transfers. The difficulty in producing these trajectories has so far shown itself in the fact that low thrust transfers need longer transfer times. We expect that an even longer transfer times can be performed with less fuel and have found that optimal solutions with longer transfer times require more burns.
8. The process of adding or subtracting burns is well-suited to indirect methods

because of the accessibility of the switching function. We can refer to this function to verify the need for more burns. The difficulty has been in actually adding this burn to the transfer. We have found that removing burns from a solution is relatively straightforward task. As the allowed transfer time is decreased, the number of burns required for optimality also decreases. This has manifested itself in a general decrease in burn time for one burn in the transfer. Once this burn time is sufficiently short we remove it altogether.

9. Finally, we are working on expanding the class of solutions to escape trajectories. This requires a modification of the TPBVP but we are not expecting insurmountable difficulty.

COMPUTATION OF OPTIMAL LOW- AND MEDIUM-THRUST ORBIT TRANSFERS

ABSTRACT

This report presents the formulation of the optimal low- and medium- thrust orbit transfer control problem, numerical methods for solution, and numerical solutions of the problem. The problem formulation is for final mass maximization and allows for second-harmonic oblateness, atmospheric drag, and three-dimensional, non-coplanar, non-aligned elliptic terminal orbits. We show examples that demonstrate the ability of two indirect methods to solve the resulting two point boundary value problems (TPBVP). The methods demonstrated are the multiple-point shooting method as formulated in H. J. Oberle's subroutine BOUNDSCO, and the minimizing boundary-condition method (MBCM). We find that although both methods can converge solutions, there are trade-offs to using either method. We present numerical solutions of two burn planar transfers in which both the initial orbit exit and final orbit entry points have been optimized. The methods used show an ability to handle thrust down to at least $T/W_O=O(10^{-2})$. They also show similar convergence abilities with or without the oblateness and drag terms. We discuss the issue of maximizing with respect to the final time and provide evidence that implies a local optimum at a maximum final time for a given number of burns.

I. INTRODUCTION

The minimum fuel orbit transfer optimization problem has been studied for many years now. Some of the most complete early work on impulsive transfers can attributed to Lawden[1], who presented a direct solution to the costate differential equations on a coasting arc. He also verified the optimality of the Hohmann transfer using early optimal control theory. In the finite thrust arena, many papers have been submitted, some with direct and some with indirect methods. For example, using collocation once then direct transcription later, Enright and Conway[2,3] examined circular, point-to-point planar transfers with ideal gravity. Zondervan, Wood, and Caughey[4] used a hybrid direct/indirect method to study three-burn transfers with plane changes in ideal gravity and for thrust levels down to $T/W_O=0.04$. Vulpetti and Montreali[5] used nonlinear programming to optimize transfers between circular orbits with inclinations. They did include oblateness and drag in their gravity model; their thrust level was about 40N. All these studies mentioned above either used fixed final time, fixed entry/exit positions in orbits, or both.

---

[1]Lawden, D.F., *Optimal Trajectories for Space Navigation*, London: Butterworths, 1963.

[2] Enright, P.J. and Conway, B.A., "Optimal Finite-Thrust Spacecraft Trajectories Using Collocation and Nonlinear Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 5, 1991, pp. 981-985

[3]Enright, P.J. and Conway, B.A., "Discrete Approximations to Optimal Trajectories Using Direct Transcription and Nonlinear Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992, pp. 994-1002

[4]Zondervan, K.P., Lincoln, L.J., and Caughey, T.K., "Optimal Low-Thrust, Three-Burn Orbit Transfers with Large Plane Changes," *Journal of the Astronautical Sciences*, Vol. 32, No. 3, 1984, pp. 407-427

[5]Vulpetti, G. and Montereali, R.M., "Hight-Thrust and Low-Thrust Two-Stage LEO-LEO Transfer" *Acta Astronautica*, Vol. 15, No. 12, 1987, pp.973-979 (84-354),

Among the studies using indirect methods, we find the work by Redding[6] which handles point-to-point low-thrust transfers with plane changes. The study was limited to transfers to geosynchronous orbits in an ideal gravity field. Horsewood, Suskin, and Pines[7] modified earlier work by Edelbaum, Sackett and Malchow[8] (a minimum time formulation using equinoctial orbital elements) to produce code for the optimization of point-to-point orbit transfers with plane changes between circular orbits with relatively low thrust in an ideal gravity field. Redding did formulate the problem with transfer time optimization while Horsewood, Suskin, and Pines fixed the final time.

With this report we examine further the increased difficulty of a formulation that includes realistic effects such as oblateness and drag. We also examine the question of optimizing the final time.

This report follows the derivation all the way through to the establishment of the two-point boundary-value problem (TPBVP) for optimal low- and medium- thrust orbit transfers. The cost function is defined simply as the final mass of the spacecraft including fuel, setting the tone for the maximization problem. The differential constraint is stated in terms of the gravity model, including oblateness and an assumed atmosphere model.

The thrust (control) appears linear in the differential constraint. This results in bang-bang control or singular-arc solutions for the final mass maximization problem. Although bang-bang control is assumed here the possibility of having a singular arc has not been ruled out for a general case. In order to ensure the singular arc solution does not occur, we check the derivative of the switching function at each switching point. However, when our programs reach a non-optimal solution high frequency chattering solutions may occur. This could indicate that singular-arc solutions are possible for some modification of system parameters and models. For the solutions presented in this report singular-arc solutions do not occur.

The final mass maximization problem should be a free transfer time optimal control problem. For impulsive thrust, the Hohmann transfer gives minimum fuel use. The three-impulse bi-elliptic transfer performs better than the two-impulse Hohmann transfer for radius ratios greater than 15.58[9]. A similar conclusion for low- and medium- thrust transfers has not been shown anywhere to our knowledge. One hypothesis is that the global extremum will be at infinite transfer time and local extremum solutions exist for finite transfer time. In other words, this assumes for a given number of switching points (must be at least two for two burns) there is a local extremum with finite transfer time. Our numerical results imply another qualification to this hypothesis: the transfer time that produces the local extremum is just short of the transfer time with which one more burn improves fuel savings. Another important observation is that, when searching among optimal solution, when the transfer time is increased, the final mass monotonically increases. This is discussed with more detail in Section IV.8.

We present numerical solutions to specific optimal transfer problems. These solutions represent the ability of two TPBVP solvers. The methods considered are (1) BOUNDSCO, a multi-point shooting algorithm devised by H. J. Oberle, described in Ref. 15 and (2) the minimizing boundary-condition method (MBCM), a modification to the shooting method devised by the authors of Ref. 8.

---

[6]Redding, D.C., "Optimal Low-Thrust Transfers to Geosynchronous Orbit," NASA Lewis SUDAAR 539, Cleveland, Ohio 44135, Sept. 1983.

[7]Horsewood, J.L., Suskin, M.A., and Pines, S., "Moon Trajectory Computational Capability Development," NASA Lewis TR-90-51, Cleveland, Ohio 44135, July 1990

[8]Edelbaum, T.N., Sackett, L.L., and Malchow, H.L., "Optimal Low Thrust Geocentric Transfer" AIAA Paper 73-1074, AIAA 10th Electric Propulsion Conference, Lake Tahoe, Nevada, November 1973

[9]Chobotov, Vladimir A., *Orbital Mechanics,* Washington: AIAA, 1991, pp. 116

II. THE PROBLEM

The problem discussed herein is the following: maximize the final mass of a thrusting spacecraft for a given orbital transfer. The craft can be considered as under the influence of some oblate planet's gravitational field and atmospheric drag. The thrust of the spacecraft is limited between zero and some $T_{\max}$. The transfer will be defined by two terminal orbits.

II. 1. *The Cost Functional*

The cost functional is

$$J = m(t_f) \tag{1}$$

where $m(tf)$ represents the mass of the spacecraft including its fuel at the end of the orbital transfer. We shall use the methods of optimal control to write the conditions necessary for maximizing the cost functional.

II. 2. *Differential Constraint: System Dynamics*

We represent the spacecraft by a point mass and assume it to be a thrusting craft acted upon the by the aerodynamic drag and oblate-body gravity forces of a central body. We also represent the central body, or planet, as a point mass positioned at its own center of gravity. We restrict the problem to crafts of mass much smaller than that of the central body, allowing us to fix the planet in inertial space. We shall describe this inertial space with a rectangular Cartesian inertial reference frame (Oxyz). The central body is fixed at the center O of this reference frame and the z-axis is perpendicular to that body's equator. All motion within this frame of reference agreeing with the above assumptions must satisfy Newton's equation:

$$\mathbf{F} = \frac{d(m\mathbf{v})}{dt} \tag{2}$$

where m is the spacecraft mass and v is its velocity with respect to the reference frame.

In this case, gravity, drag, and thrust make up the total force acting on the craft. This gives us

$$m\dot{\mathbf{v}} = \mathbf{F}_{thrust} - \mathbf{F}_{drag} - \mathbf{F}_{gravity} \tag{3}$$

We write the thrust as some time-varying magnitude T independent of a time-varying direction $\mathbf{e}_T$:

$$\mathbf{F}_{thrust} = T\mathbf{e}_T \tag{4}$$

Note that $\mathbf{e}_T$ is expressed as a unit vector. For a three dimensional thrust vector the control requires a magnitude and three components or two angles. For two dimensional problems, the one magnitude and only two independent control components or one angle are required.

The mass will decrease according to

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{5}$$

We assume that the atmosphere surrounding the central body can be described by an exponential model of the standard atmosphere. The following equation[10] describes such a drag force:

$$\mathbf{F}_{drag} = \frac{1}{2}\rho_o e^{-\beta(r-r_o)} S C_D v \mathbf{v} \tag{6}$$

\where β is a constant from the atmosphere model describing air density variation in the prescribed altitude region, ρ_o is the atmosphere density at the altitude r_o, S is the wetted area of the craft, C_D is the craft's drag coefficient, v is the craft's current velocity vector with respect to the inertial reference frame, v is the magnitude of that velocity, and r is the magnitude of the position vector with respect to the inertial reference frame. We assume that the product SC_D is not a function of time and that the craft always remains in a region where the chosen exponential atmosphere model is valid.

Within the confines of this study, the only other influence on the craft is gravitational potential energy. The gravitational potential energy to the second harmonic is[11]

$$U = -\frac{\mu m}{r} - \frac{1}{3} J R^2 \frac{\mu m}{r^3}(1 - 3\cos^2(\theta)) \tag{7}$$

where R is the equatorial radius of the central body, θ is the latitude angle of the current position from the equator, and r is the distance from the central body's center of gravity to the current position of the craft with respect to the inertial reference frame, μ is the gravitational constant for the central body, m is the total mass of the spacecraft, and J is a constant describing the mass distribution of the central body. There are additional mass distribution terms but we shall truncate the series here. θ is described with Cartesian coordinates by

$$z = r\cos(\theta) \tag{8}$$

Eq. (7) now becomes

$$U = \frac{-\mu m}{r} - \frac{1}{3} J R^2 \frac{\mu m}{r^3}\left(1 - 3\left(\frac{z}{r}\right)^2\right) \tag{9}$$

---

[10]Anderson, J.D., *Fundamentals of Aerodynamics*, New York: McGraw-Hill Book Co., 1984.

[11]Space Technology Laboratories, *Flight Performance Handbook for Orbital Operations*, New York: Wiley, 1963.

We can write the equations of motion by evaluating $\partial U/\partial \mathbf{r}$, and, along with Eq(4) and Eq(6) substitute into Eq(3). The equations of motion are then

$$m\ddot{x} = Te_x - \frac{\mu m}{r^3}x - JR^2\frac{\mu m}{r^5}x\left(1-5\left(\frac{z}{r}\right)^2\right) - \frac{1}{2}\rho_o e^{-\beta(r-r_o)}SC_D v\dot{x} \tag{10a}$$

$$m\ddot{y} = Te_y - \frac{\mu m}{r^3}y - JR^2\frac{\mu m}{r^5}y\left(1-5\left(\frac{z}{r}\right)^2\right) - \frac{1}{2}\rho_o e^{-\beta(r-r_o)}SC_D v\dot{y} \tag{10b}$$

$$m\ddot{z} = Te_z - \frac{\mu m}{r^3}z - JR^2\frac{\mu m}{r^5}z\left(3-5\left(\frac{z}{r}\right)^2\right) - \frac{1}{2}\rho_o e^{-\beta(r-r_o)}SC_D v\dot{z} \tag{10c}$$

which, after conforming to convention by changing from J to J_2 as described in Ref. 11, can be written as a first-order system in vector-matrix form as

$$\dot{\mathbf{r}} = \mathbf{v} \tag{11a}$$

$$\dot{\mathbf{v}} = \frac{T}{m}\mathbf{e}_T - \frac{\mu}{r^3}\mathbf{r} - \left\{\frac{3}{2}\mu J_2\frac{R^2}{r^5}\left(\overline{\mathbf{N}}-5\left(\frac{z}{r}\right)^2\right)\right\}\mathbf{r} - \frac{1}{2}\frac{\rho_o}{m}e^{-\beta(r-r_o)}SC_D v\mathbf{v} \tag{11b}$$

where $\overline{\mathbf{N}}$=diag{1,1,3}.

III. The First-Order Necessary Conditions

The necessary conditions require the formulation of an adjoined cost functional. The state vector, x(t), shall be defined as

$$\mathbf{x}(t) = \left[\mathbf{r}^T(t) \quad \mathbf{v}^T(t) \quad m(t)\right]^T \tag{12}$$

The state is confined by the system dynamics, Eqs. (11a-b) and Eq. (5), for all time $t \in [0,t_f]$. The right-hand sides of these form the differential constraint of the control problem and shall be referred to collectively as $\mathbf{f}(\mathbf{x}(t),\mathbf{u}(t))$.

The thrust magnitude has both an upper and a lower bound. The upper bound we shall call $T_{\max}$, the lower bound is zero. We, therefore, also have an inequality constraint that must be satisfied for all time $t \in [0,t_f]$:

$$0 \le T \le T_{\max} \tag{13}$$

Finally, we need to specify the terminal orbits. The terminal orbits are specified by a vector equation

$$\psi(\mathbf{x}(0),\mathbf{x}(t_f)) = 0 \tag{14}$$

that is only satisfied when our initial and final states both lie on their respective prescribed orbits.

These constraints come together to form the adjoined cost functional:

$$J = m(t_f) + \nu^T \psi(\mathbf{x}(0),\mathbf{x}(t_f)) + \int_0^{t_f} \left\{ \lambda^T [\mathbf{f}(\mathbf{x}(t),\mathbf{u}(t)) - \dot{\mathbf{x}}] \right\} dt \tag{15}$$

where *m(tf)* is the final mass and ν, the Lagrange multiplier corresponding to the boundary conditions.

The λ shown in the cost functional is the costate vector, also called the Lagrange multiplier or adjoint vector. This vector will be of the same dimensions as the state. For simplification's sake, we will segment this vector much as we did the state:

$$\lambda(t) = \left[\lambda_r^T(t) \quad \lambda_v^T(t) \quad \lambda_m(t) \right]^T \tag{16}$$

III.1. *The Hamiltonian*

With the pertinent dynamics and the cost functional defined, we are now able to write the Hamiltonian function for the optimal control problem. We take the Hamiltonian from the cost functional as

$$H(\mathbf{x}(t),\mathbf{u}(t)) = \lambda^T \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t)) \tag{17}$$

A major simplification can be made now. Notice that the Hamiltonian is linear with respect to the control *T*:

$$H(\mathbf{x}(t),\mathbf{u}(t)) = \frac{T}{m} \lambda_v^T \mathbf{e}_T - \lambda_m \frac{T}{g_o I_{sp}} + \mathrm{L} \tag{18}$$

This, in conjunction with the structure of the thrust constraint, means this is a 'bang-bang' control problem. Enough is known about this type of problem so that we may do the following:

(1) Establish what will be called the switching function. In general, the switching function is defined by the partial derivative of the Hamiltonian with respect to the control.

$$H_T = \frac{\partial H}{\partial T} = \frac{\lambda_v^T \mathbf{e}_T}{m} - \frac{\lambda_m}{g_o I_{sp}} \tag{19}$$

(2) Evaluate a restricted case of the well-known Euler-Lagrange equations. Most of these determine the costate dynamics and we shall see these in Section III.2, however, the last one determines part of the control for the problem. This equation is

$$\frac{\partial H}{\partial \mathbf{e}_T} = 0 \tag{20}$$

Evaluating (20), we must remember that $\mathbf{e}_T$ represents only the direction of the thrust. If we define γ as the angle between $\mathbf{e}_T$ and λ_v we may write

$$H = \frac{T}{m}\lambda_v^T \mathbf{e}_T + \mathrm{L} = \frac{T}{m}|\lambda_v|\cos(\gamma) + \mathrm{L} \tag{21}$$

Evaluating H_γ we find that

$$\frac{\partial H}{\partial \gamma} = -\frac{T}{m}|\lambda_v|\sin(\gamma) \tag{22}$$

and this equals zero only when (1) the thrust is off, (2) the vectors $\mathbf{e}_T$ and λ_v are parallel, or (3) $|\lambda_v|=0$. The only case that we can apply for all time and that will not be trivial is (2). This means that there are only two choices for $\mathbf{e}_T$: in the direction of λ_v or in the exact opposite direction. Since this is a mass-maximization problem, we need to have $H_{\gamma\gamma}$ negative (one of the sufficient conditions for the second variation). It can be easily shown that this is satisfied with $\mathbf{e}_T$ in the direction of λ_v or

$$\mathbf{e}_T = \frac{\lambda_v}{|\lambda_v|} \tag{23}$$

The thrust must obey this for all time $t \in [0,tf]$. The result is consistent with that of Lawden's primer vector, Ref. 1.

(3) Perform bang-bang control with T. This means that T is always on-boundary, i.e. T=0 or $T=T_{\max}$ at any $t \in [0,tf]$. We know which value to use for T by evaluating the switching function, Eq. (19), which is written as

$$H_T = \frac{|\lambda_v|}{m} - \frac{\lambda_m}{g_o I_{sp}} \tag{24}$$

The bang-bang control law is

$$\begin{aligned} H_T > 0, \quad T = T_{max} \\ H_T < 0, \quad T = 0 \end{aligned} \tag{25}$$

This switching structure satisfies the Pontryagin maximum principle by maximizing the Hamiltonian using T. If H_T were to be zero for a finite time the control would be singular. Higher order derivatives of H_T would then be needed to calculate T.

III.2. *The Costates*

The costate dynamics can be found from the following Euler-Lagrange equations, relating them to the Hamiltonian:

$$\dot{\lambda}_r = -\left(\frac{\partial H}{\partial \mathbf{r}}\right)^T \tag{26}$$

$$\dot{\lambda}_v = -\left(\frac{\partial H}{\partial \mathbf{v}}\right)^T \tag{27}$$

$$\dot{\lambda}_m = -\frac{\partial H}{\partial m} \tag{28}$$

To evaluate these, we must first substitute the equations of motion into the Hamiltonian:

$$\begin{aligned} H = \lambda_v{}^T \mathrm{v} + \lambda_r{}^T \left\{ \frac{T}{m} \mathrm{e}_T - \frac{\mu}{r^3} \mathrm{r} - \left\{ \frac{3}{2} \mu J_2 \frac{R^2}{r^5} \left(\bar{\mathrm{N}} - 5 \left(\frac{\mathrm{z}}{r} \right)^2 \right) \right\} \mathrm{r} \right. \\ \left. - \frac{1}{2} \frac{\rho_o}{m} e^{-\beta(r-r_o)} S C_D v \mathrm{v} \right\} - \lambda_m \frac{T}{g_o I_{sp}} \end{aligned} \tag{29}$$

When evaluated, these become the following vector and scalar differential equations:

$$\dot{\lambda}_r = \mu\left[\frac{\lambda_v}{r^3} - 3\frac{(\lambda_v{}^T r)r}{r^5}\right] - \frac{1}{2}\frac{\rho_o}{m}\frac{\beta}{r}e^{-\beta(r-r_o)}SC_D v(\lambda_v{}^T v)r$$

$$+\frac{3}{2}\mu J_2 R^2\left[\frac{\overline{N}\lambda_v}{r^5} - 5\frac{(\lambda_v{}^T\overline{N}r)r}{r^7}\right] - \frac{15}{2}\mu J_2 R^2\left[\frac{z^2}{r^7}\lambda_v - (\lambda_v{}^T r)\left(\frac{7z^2}{r^9}r - \frac{2z}{r^7}\begin{bmatrix}0\\0\\1\end{bmatrix}\right)\right] \tag{30}$$

$$\dot{\lambda}_v = -\lambda_r + \frac{1}{2}\frac{\rho_o}{m}e^{-\beta(r-r_o)}SC_D\left[\lambda_v v + \frac{(\lambda_v{}^T v)v}{v}\right] \tag{31}$$

$$\dot{\lambda}_m = \frac{T}{m^2}\lambda_v{}^T e_T - \frac{1}{2}\frac{\rho_o}{m^2}e^{-\beta(r-r_o)}SC_D v\lambda_v{}^T v \tag{32}$$

IV. Solved Problems

IV.1. *Simplifications*

We have reduced the problem from three-dimensions to two-dimensions. To remove the dimension, we simply remove the z-component to all equations. Because of the chosen coordinate system, this also means that all orbit transfers considered are equatorial. Unfortunately, the effect of oblateness is substantially decreased for this case.

We have also fixed the transfer time for the solutions presented here. By examining the behavior of the solution as the transfer time is varied we are able to see why the transversality condition is difficult to converge.

The differential equations above are written with respect to the independent variable t (time). Since neither method used is written to iterate on the transfer time we need to make this variable part of the state. The state here is for the numerical code and not for the optimal control problem. Now, this means that we must define a new independent variable τ (non-dimensional time) to be used in the place of t (dimensional time). The following scaling is made:

$$t = t_f\tau \tag{33}$$

Therefore, to accomplish this, t_f must multiply the derivatives of the states and costates and the time interval must be changed to [0,1].

IV.2. *The Two-Point Boundary Value Problem*

As a result of the simplifications, the boundary conditions have been stated in two dimensions. The orbit conditions, however, require a more abstract formulation as we cannot specify exactly at what point the craft will enter or leave an orbit. The following relations completely specify an orbit: (All of the following conditions are to be evaluated at the appropriate time, τ=0 or 1)

$$\psi_1 = \mathbf{r} \times \mathbf{v} - \mathbf{h} = \mathrm{x}\mathrm{v} - \mathrm{y}\mathrm{u} - h = 0 \tag{34a}$$

$$\psi_2 = \left[\left(v^2 - \frac{\mu}{r}\right)\mathrm{x} - (\mathbf{r}^T\mathbf{v})\mathrm{u}\right] - \mu e_x = 0 \tag{34b}$$

$$\psi_3 = \left[\left(v^2 - \frac{\mu}{r}\right)\mathrm{y} - (\mathbf{r}^T\mathbf{v})\mathrm{v}\right] - \mu e_y = 0 \tag{34c}$$

All that is required to specify an orbit without specifying a position on that orbit is the angular momentum and eccentricity vectors. In a full three-dimensional case, these conditions will form six equations. Since the two vectors are perpendicular, one of these equations will be redundant and thus removable. It follows that it does not matter which equation is removed, but there is no doubt that one must be removed and no more. In the two-dimensional case, as demonstrated above, the angular momentum vector has two zero components while the eccentricity vector has one zero component. Since there are only four orbital elements for the two-dimensional case, we do not have to worry about removing one of the equations; orthogonality has already removed them.

It would seem that the ellipse equation and the energy equation can replace (34b) and (34c). However, the combined constraints of angular momentum, orbit geometry, and energy equations do not uniquely specify an orbit. Angular momentum and energy do determine the pair of orbital elements a and e. However, this orbit may be rotated to find more than one argument of perigee such that the specified conic section is intersected and the proper velocity magnitude, but not direction, is found. This results because there are two possible velocity vectors at one point with the same angular momentum and energy.

Eqns (34) completely determine the final orbit without setting the position on the orbit. However, these conditions do not complete the two-point boundary-value problem. To complete the TPBVP, the methods of optimal control supply a set of natural boundary conditions

$$\lambda(1) = \left(\frac{\partial G}{\partial \mathbf{x}(1)}\right)^T \tag{35a}$$

$$\lambda(0) = -\left(\frac{\partial G}{\partial \mathbf{x}(0)}\right)^T \tag{35b}$$

where G is constructed from the function portion of the cost functional, e.g. for the cost functional in Eq. (15) G is

$$G = m(1) + \nu^{\mathrm{T}}\psi(\mathbf{x}(0), \mathbf{x}(1)) \tag{36}$$

Constructing G with the above conditions on the states, we can find conditions on the costates:

$$G = m(1) + [\nu_1\psi_1 + \nu_2\psi_2 + \nu_3\psi_3]\big|_{\tau=1} + [\nu_4\psi_1 + \nu_5\psi_2 + \nu_6\psi_3]\big|_{\tau=0} \tag{37}$$

Evaluating Eq.(35a) gives, at τ=1:

$$\lambda_x = \nu_1 v + \nu_2\left(v^2 - \frac{\mu}{r} + \frac{\mu x^2}{r^3}\right) + \nu_3\left(\frac{xy}{r^3} - uv\right) \tag{38a}$$

$$\lambda_y = -\nu_1 u + \nu_2\left(\frac{xy}{r^3} - uv\right) + \nu_3\left(u^2 - \frac{\mu}{r} + \frac{\mu y^2}{r^3}\right) \tag{38b}$$

$$\lambda_u = -\nu_1 y + \nu_2(-yv) + \nu_3(2yu - xv) \tag{38c}$$

$$\lambda_v = \nu_1 x + \nu_2(2xv - uy) + \nu_3(-xu) \tag{38d}$$

$$\lambda_m = 1 \tag{38e}$$

Note that the constant Lagrange multipliers ν_i are additional unknowns. At τ=0 we get the expressions identical to Eqs. (38a-d) but with a negative sign placed as indicated by Eq. (35b).

The last condition deals with the final time. If the final time was to be optimized indirectly we would use the transversality condition

$$H(\mathbf{x}(1), \mathbf{u}(1), \lambda(1)) = -\frac{\partial G}{\partial t_f} = 0 \tag{38f}$$

However, we choose to specify the final time in the solutions presented. By looking at the characteristics of the solution as the transfer time is varied, we may gain some important insights as to the nature of this optimization.

Since the transfer time has been included in the state for BOUNDSCO, Eq. (38f) is replaced by a condition on *tf.* for the fixed transfer time problem.

IV.3. *Non-Dimensionalization*

To improve accuracy, we have non-dimensionalized the problem. This aids in a few ways. First, the integration of the state is more accurate because all variations are on the same order. Second, convergence is improved because all the boundary conditions are immediately placed at or near the same order. Our non-dimensionalizations follow:

$$\hat{\mathbf{r}} \equiv \frac{\mathbf{r}}{r^{\star}} , \qquad \hat{\mathbf{v}} \equiv \frac{\mathbf{v}}{\sqrt{\mu/r^{\star}}} , \qquad \hat{m} \equiv \frac{m}{m^{\star}} , \qquad \hat{t}_f \equiv \frac{t_f}{\sqrt{r^{\star 3}/\mu}} \tag{39a-d}$$

and they require the following

$$\hat{T} \equiv \frac{T/m^{\star}}{\mu/r^{\star 2}} , \qquad (\hat{g}_o \hat{I}_{sp}) \equiv g_o I_{sp} \sqrt{r^{\star}/\mu} , \qquad (\hat{\rho}_o \hat{S} \hat{C}_D) \equiv \rho_o S C_D \frac{r^{\star}}{m^{\star}} , \tag{39e-g}$$

$$\hat{r}_o \equiv \frac{r_o}{r^{\star}} , \qquad \hat{\beta} \equiv \beta r^{\star} , \qquad \hat{R}_o \equiv \frac{R_o}{r^{\star}} \tag{39h-j}$$

The choices of $r^{\star}$ and $m^{\star}$ are completely arbitrary. However, it needs to be said that after a problem is solved by these nondimensionalizations rescaling must be exercised with caution. This is a direct result of the atmosphere model; if the rescaling is not consistent with the atmosphere model, the results are invalid, e.g. rescaling also rescales the atmosphere model (note Eq. (6)).

If we solve Eqs (39a-j) such that the dimensional parameter is on the left-hand side and then substitute into the original dynamics we find equations that are *exactly* identical to the original equations with μ=1 (The value of J_2, however, has no dimensions and is not changed). This can be extended to the boundary equations and the costate differential equations. A special note is required: the costates resulting from the solution to the problem with this transformation will be some scalar multiplied by the 'dimensional' costates, e.g.

$$\hat{\lambda} \equiv \frac{\lambda}{\lambda^{\star}} \tag{39k}$$

which requires

$$\hat{\nu} \equiv \frac{\nu}{\lambda^{\star}} \tag{39l}$$

where $\lambda^{\star}$ is completely arbitrary. This is easily verified by substitution into the differential equations and boundary conditions.

IV.4. *Atmosphere Model*

Any atmosphere is usable by simple substitution early in the derivation of the differential constraint. For the purposes of this report we have chosen a simple atmosphere model. The model is not intended to accurately represent the Earth's atmosphere, or any other planet for that matter. It is implemented only for the purpose of demonstrating the methods for solving the optimization problem.

The model is defined from a reference altitude of 450km above the planet's equator and is

assumed spherical above the oblate planet. The entire atmosphere region is assumed isothermal with a temperature of 1000K. The density at the definition altitude is 1.184×10^{-12} kg/m^3. This definition point for our model is taken from the 1976 U.S. Standard Atmosphere[12].

IV.5. *The Multiple Point Shooting Method of BOUNDSCO*

One method we are currently using to solve the TPBVP is the multiple point shooting method. The specific algorithms are those given by H. J. Oberle in his subroutine BOUNDSCO[13], written in FORTRAN. His method is a modification of the multiple point shooting method.

The use of this method requires the writing of a few routines that define the problem. These routines include, of course, the calling routine itself, a subroutine defining the differential constraint (or system dynamics), and a subroutine that defines the constraints on the problem.

The state defined for the optimal control problem differs slightly from the state used in BOUNDSCO. The state used in BOUNDSCO includes the ν vector, from the natural boundary conditions, to the state. This requires also that the system dynamics includes a corresponding number of zero derivatives. We justify this by noting that it allows the statement of the absolute and natural boundary conditions exactly as they are in this report. If we did not do this, we would have to solve a system of three of the four natural boundary conditions for ν and substitute the result into the fourth equation, using it in place of the four. This may seem desirable, one equation in the place of four, however, the simple structure of the four equations is much more tractable for BOUNDSCO than the complex structure of the one equation.

There is one particular feature that makes BOUNDSCO very attractive: the explicit inclusion of switching points in the problem formulation. Oberle allows the user to guess the switching points outside of the system dynamics. In doing so, he provides the equivalent of guessing the burn times and lengths as usually only done in direct methods. This simplifies integration and improves convergence because it removes a lot of sensitivity between the trajectory and the guessed switching times. The switching times are no longer determined by integrating from the guessed initial state; they are specified directly. There is a tradeoff: the user must assume a switching structure and verify it outside of BOUNDSCO. If the switching structure assumed is not correct, BOUNDSCO will not modify the switching structure to achieve optimality.

The guessing of the switching times requires a brief explanation. It makes no sense to guess an initial burn-on time because then the first point in the initial guess is indeterminate, i.e. it doesn't matter where you start coasting from but it does matter when you start burning. Therefore, we always assume switching structures that begin with a thrusting arc and that end with a thrusting arc. The same reasoning holds for the final point.

IV.6. *The Minimizing-Boundary-Condition Method*

The second method we are using is called the Minimizing-Boundary-Condition Method (MBCM)[14]. This method is a modification to the simple shooting method. It expands the set of available solutions by removing one boundary condition. The choice of this boundary condition is

---

[12]United States. COESA. *U.S. Standard Atmosphere, 1976*, Washington: GPO, 1976

[13]Oberle, H. J., BOUNDSCO - Hinweise zur Benutzung des Mehrzielverfahrens für die numerische Lösung von Randwerproblemen mit Schaltbedingungen, Hamburger Beiträge zur Angewandten Mathematik, Berichte 6, 1987.

[14]Chuang, C.-H., and Speyer, J.L. "Periodic Optimal Hypersonic SCRAMjet Cruise," *Optimal Control Applications & Methods*, Vol. 8, 1987, pp. 231-242.

arbitrary. The number of unknowns is unchanged; he solutions become a one-dimensional family. Since there is a much larger set of solutions, it is much easier to solve the resulting boundary-value problem. Once this is accomplished, the search for the solution that incorporates the final boundary conditions is treated as a minimization problem. The gradient is numerically calculated and used to update the initial state until the last boundary condition is satisfied. This method is at least as effective as BOUNDSCO in solving the two-point boundary-value problems for the current solved optimal orbit transfers.

The switching structure of the optimal control is implicit in MBCM. The program checks the switching function at each integration step. If the switching function alters sign at one integration step, the program stops the integration and restores all the states to the beginning of the step. A secant method then calculates a smaller step size for integrating the switching function to a zero point. From our experience with MBCM some sensitive problems need fourteen digits of accuracy in their switching function. Once the integration passes the switching points the program switches the control and uses a normal step size for integration.

IV.7. *Optimizing the Initial Point*

The TPBVP we have described is for optimizing both the exit point on the initial orbit and the entry point on the final orbit. However, it is much easier to obtain a solution between two orbits if one fixes at least the initial point at first. This is because the natural B.C.'s are replaced by the specification of a terminal position and velocity. With the fixed-point problem solved, the simplest way of optimizing the fixed point, of course, is to, holding all other parameters fixed, take a range of true anomaly values on that orbit and look at the resulting final mass. We performed this task on the transfer shown in Fig. 4 and the results are displayed in Fig. 1. The results are quite satisfactory. There is one obvious local maximum in the range of values searched. This is the best initial point.

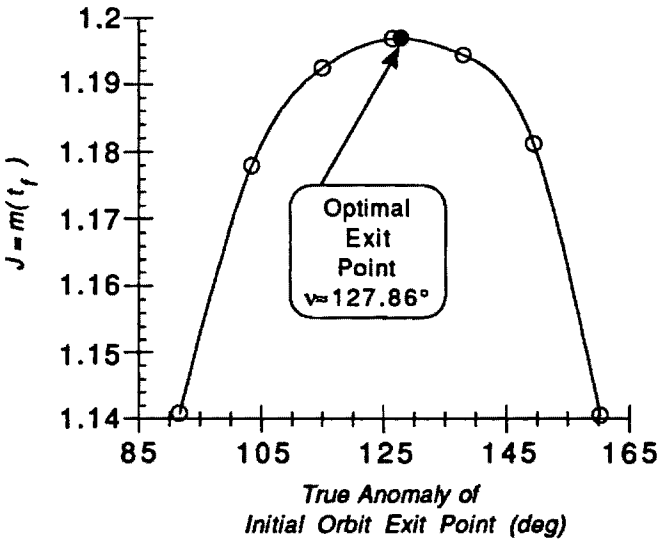


Figure 1 Plot of the final mass as the true anomaly on the initial orbit is varied for the transfer in Fig. 4

However, if we desire to know the optimal initial orbit exit true anomaly with any accuracy we must satisfy the necessary conditions. This is accomplished by simply using one of the solutions we already found as the guess for converging a solution constrained by the full set of necessary conditions. We choose from the results of our search one that landed closest to the optimum, we then use BOUNDSCO or MBCM to converge the necessary conditions.

IV.8. *What is the Optimal Transfer Time?*

The only optimization that remains to be discussed is with respect to the transfer time. We have found this to be somewhat more complicated than optimizing with respect to the initial or final orbit true anomalies. Figure 2 shows the results of searching our example transfer for a range of final times. In doing this search we discovered two interesting things: (1) as the transfer time was lowered, the coasting arc was shortened; in fact, at the shortest transfer time examined, there was no coasting arc; and (2) after a certain transfer time BOUNDSCO did not return an optimal two-burn, and the cost appears to be locally maximized. Fig. 3 shows the switching times as the transfer time was varied and illustrates the property described above.

Now when we say that BOUNDSCO did not return an optimal two-burn solution, we mean that the Pontryagin Maximum Principle was violated (as we have mentioned before that BOUNDSCO will allow this). The lower bound makes perfect intuitive sense: this is the quickest that one can do the transfer in an optimal fashion, it is also the least fuel-efficient - the motor is on for the entire transfer. Now, as we relax the time constraint and allow for longer transfers, we approach an optimal transfer time. However, we only *approach* the optimal transfer time. We can claim to be quite near the optimal transfer time as also indicated by Fig. 2 . This figure shows that, as required by optimal control theory, the transversality is close to being satisfied when the mass is close to being optimized. We can meet the transversality condition for the two-burn transfer with an accuracy of 10^{-4}.

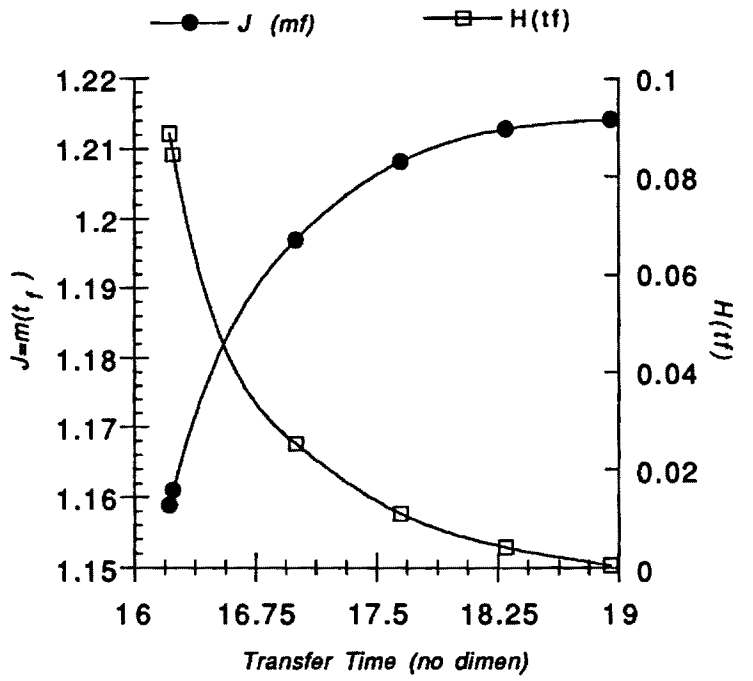


Figure 2 Plot of the final mass and Hamiltonian (for the transversality condition) as the transfer time is varied for the transfer in Fig. 4

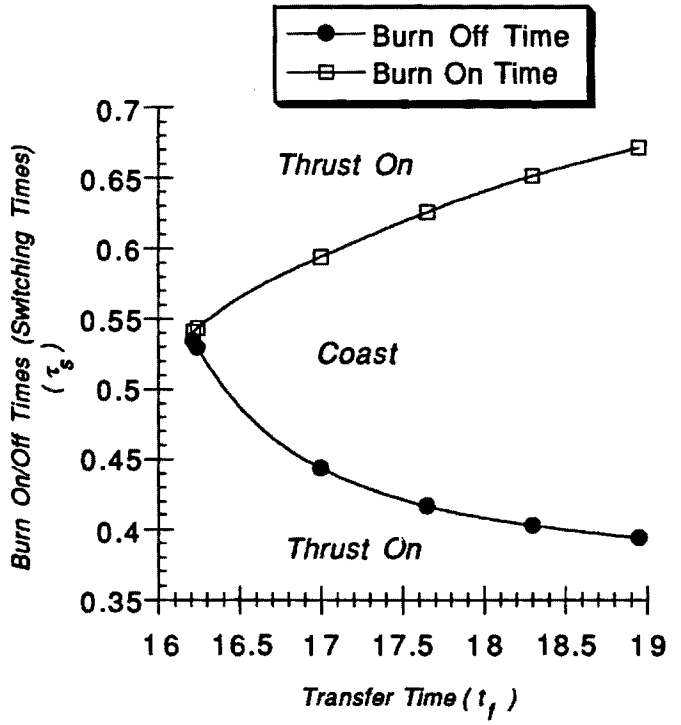


Figure 3 Plot of the switching times of the optimal two-burn solution as the transfer time is varied (for the transversality condition) for the transfer in Fig. 4

We must be very careful in attempting to explain why BOUNDSCO did not return an optimal transfer after t_f=18.9. Our speculation is that this transfer represents a point where both the two- and three-burn solutions have the same cost, and that for transfers with a greater transfer time, the three-burn gives the greater final mass.

IV.9. *Sample Problems and Solutions*

Two solutions are presented in this section, all of which both methods were able to converge. All have been nondimensionalized and use the atmosphere model presented above. All include a drag and oblateness model as mentioned above, with J_2=1082.61x10^{-6}. The drag model is based on a reference state from the 1976 standard atmosphere. At an altitude of 450 km the temperature is 1000k and the density is 1.184x10^{-12} kg/m^3. This gives us $\rho_o SC_D$=7.632x10^{-7}. Now, for the same reasons as mentioned earlier, we are forced to specify r^I=200 km. We do not have to specify m^I.

The first solution's tranjectory is shown in Fig 4, it's velocity portrait is shown in Fig. 5, and a plot of the thrust vector angle is shown in Fig. 6. This is a fixed transfer time transfer and corresponds to the trajectory represented by the leftmost point in Fig. 3. The transfer time for this trajectory is the shortest allowed an optimal transfer with these terminal orbits. And, as such, it has the lowest final mass of the allowed transfers. This is a descent trajectory; the initial orbit is higher than the final orbit. The initial orbit is: a=3.847, e=0.02378. The final orbit is: a=1.5, e=0.3333. The initial and final orbits are the same as for Fig. 4. The transfer time is 16.2. The initial mass is 1.608 and the final mass is 1.1547.

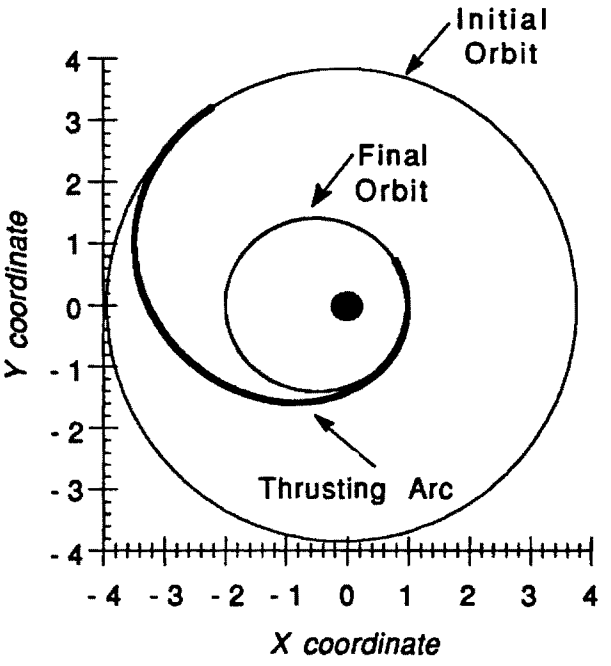


Figure 4 Mass-Optimal, Ellipse-to-Ellipse Transfer with Exit and Entry Points Optimized. T=0.03, $g_o Isp$=1.313, t_f=16.2

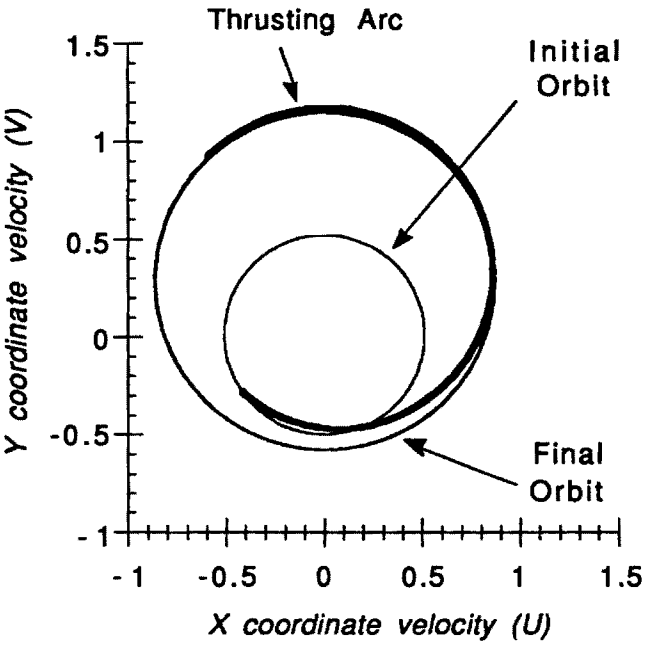


Figure 5 Velocity Phase Portrait for the Mass-Optimal, Ellipse-to-Ellipse Transfer of Figure 4

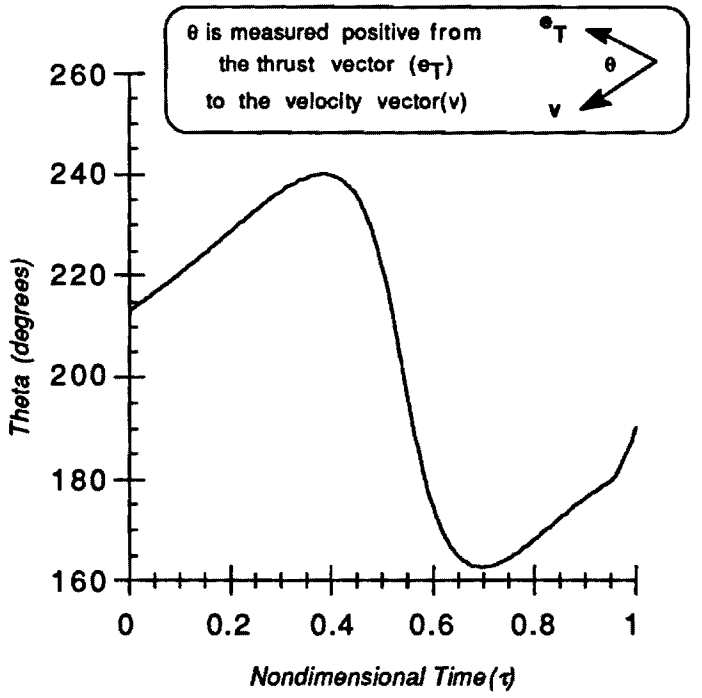


Figure 6 Plot of Angle Between Thrust and Velocity Vectors Versus Nodimensional Time

This transfer is accomplished in one burn, however, at $\tau_s \approx 0.54$, the switching function comes extremely close to zero. We shall, therefore, speak of this as an instantaneous transfer orbit, The first burn, then, is about Δt=8.748 and imparts ΔE=-0.0578 and Δh=-0.5282. The instantaneous coast is over an orbit of a=2.663, e=0.4786, and ω=-39.06°. The second burn is about Δt=7.452 and imparts ΔE=-0.1456 and Δh=-0.278.

The second solution presented, Figs 7-9, is similar to the previous one.. It corresponds to the trajectory represented by the rightmost point in Fig. 3. The transfer time for the this trajectory is the longest allowed an optimal transfer with these terminal orbits. And, as such, it has the highest final mass of the allowed transfers. It has the same initial and final orbits as the previous transfer. The transfer time is 18.95. The initial mass is 1.608 and the final mass is 1.214.

Two burns are used to complete the transfer. The first burn is about Δt=7.47 and imparts ΔE=-0.0672 and Δh=-0.598. The coast is Δt=5.26 over an orbit of a=2.536, e=0.518, and ω=-10.5°. The second burn is about Δt=6.22 and imparts ΔE=-0.136 and Δh=-0.2079. Again, this transfer has many similarities to the other two.

Looking at the transfers above, we find just what we expect. The quickest transfer has to impart the most energy to complete its transfer and the longest transfer is just the opposite. We also found that after a certain maximum transfer time there was no optimal two-burn. We also expect to find that a family of optimal three-burn transfers occupy the next range of transfer times.

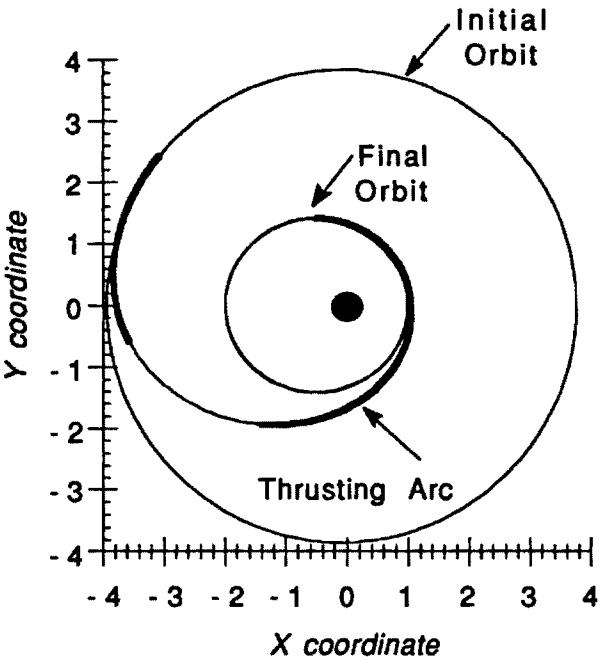


Figure 7 Mass-Optimal, Ellipse-to-Ellipse Transfer with Exit and Entry Points Optimized. T=0.0143, $g_o Isp$=1.313, t_f=18.95

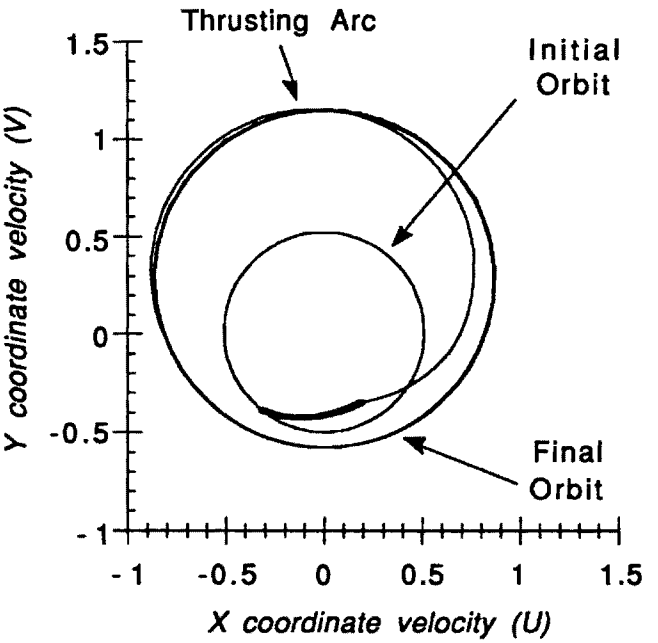


Figure 8 Velocity Phase Portrait for the Mass-Optimal, Ellipse-to-Ellipse Transfer of Figure 4

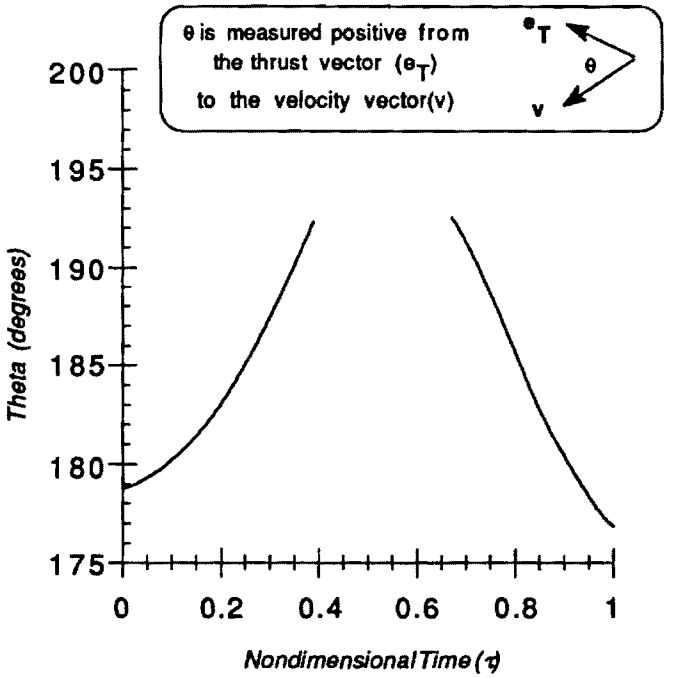


Figure 9 Plot of Angle Between Thrust and Velocity Vectors Versus Nodimensional Time

V. Conclusions

The performance of both BOUNDSCO and MBCM was acceptable. There are significant tradeoffs between the two, however. The main distinguishing feature of BOUNDSCO, in the interests of this study, is the implementation of switching conditions. BOUNDSCO does not supply information to the user about the polarity of the switching function. This leaves the user uncertain as to whether the solution satisfies the Pontryagin Maximum Principle. However, this weakness of BOUNDSCO is also an advantage in that it allows the burn times to be guessed directly. MBCM applies the switching conditions directly and, therefore, every solution that it converges satisfies all the necessary conditions. From this viewpoint, MBCM has an advantage. But the difference that matters more is that MBCM has a greater region of convergence than just simple shooting; this increase being the result of removing a boundary condition.

In this report we have demonstrated the ability of two indirect methods to solve the orbit transfer mass-optimization problem. In particular, we have presented two and three burn solutions, one low and one medium thrust. Both of these transfers were created with indirect methods.

Through the use of these methods we have uncovered some information about the optimal transfer time for finite burn transfer. We have evidence that implies that as the transfer time is increased, the final mass of an optimal transfer monotonically increases.

Addressing the question of difficulty in including drag and oblateness effects we have found that these are not difficult to include. We were able to obtain solutions with drag and oblateness terms via BOUNDSCO and MBCM.

E-16-M45
3

SEMI-ANNUAL STATUS REPORT

Submitted to:	NASA Marshall Space Flight Center
Grant Title:	Theory and Computation of Optimal Low- and Medium-Thrust Transfers
Grant Number:	NAG8-921
Principal Investigator /Project Director:	Dr. C.-H. Chuang School of Aerospace Engineering Georgia Institute of Technology Atlanta, GA 30332-0150 Phone: (404) 894-3075 Fax: (404) 894-2760 E-mail: ch.chuang@aerospace.gatech.edu
Research Assistant:	Troy Goodson School of Aerospace Engineering Georgia Institute of Technology
Period Covered:	July 7, 1993 to January 6, 1994
Date of Submission:	January 4, 1994

TABLE OF CONTENTS

ABSTRACT 1

I. INTRODUCTION 1

II. DIRECTION CORRECTION METHOD 4

III. PATCHED TRANSFER METHOD 17

IV. CONCLUSIONS 23

V. REFERENCES 24

Numerical Computation of Fuel-Optimal, Low- and Medium- Thrust Orbit Transfers in Large Numbers of Burns

ABSTRACT

This report presents two numerical methods considered for the computation of fuel-optimal, low-thrust orbit transfers in large numbers of burns. The origins of these methods are observations made with the extremal solutions of transfers in small numbers of burns; there seems to exist a trend such that the longer the time allowed to perform an optimal transfer the less fuel that is used. These longer transfers are obviously of interest since they require a motor of low thrust; however, we also find a trend that the longer the time allowed to perform the optimal transfer the more burns are required to satisfy optimality. Unfortunately, this usually increases the difficulty of computation.

Both of the methods described use small-numbered burn solutions to determine solutions in large numbers of burns. One method is a homotopy method that corrects for problems that arise when a solution requires a new burn or coast arc for optimality. The other method is to simply patch together long transfers from smaller ones. An orbit correction problem is solved to develop this method. This method may also lead to a good guidance law for transfer orbits with long transfer times.

I. INTRODUCTION

Electric propulsion, with its high specific impulse, promises very low fuel consumption but it produces less thrust than its counterparts. If one wants to use electric propulsion, one needs to be prepared to tolerate the long transfer times that will be incurred. The greater time spent thrusting must be spent wisely if fuel savings are to

realized. Furthermore, the effects of Earth's oblateness and atmospheric drag become more significant on the orbits of long transfer times.

To spend the thrusting time wisely, we form an optimal control problem to maximize the mass at the end of the transfer. This, therefore, is our cost function

$$J = m(t_f) \tag{1}$$

subject to the boundary conditions

$$\psi\left(\mathbf{r}(0), \mathbf{v}(0), \mathbf{r}\left(t_f\right), \mathbf{v}\left(t_f\right)\right) = 0 \tag{2}$$

and the state dynamics

$$\dot{\mathbf{r}} = \mathbf{v} \tag{3}$$

$$\dot{\mathbf{v}} = \frac{T}{m}\mathbf{e}_T - \frac{\mu}{r^3}\mathbf{r} \tag{4}$$

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{5}$$

where $\mathbf{e}_T$ is the thrust direction, a unit vector, and the thrust magnitude, T, is limited between zero and some maximum value T_{max}, μ is the gravitational constant, g_o is the gravitational acceleration at sea-level, and I_{sp} is the specific impulse of the motor. Sometimes $g_o I_{sp}$ is referred to as the exit velocity of the motor. If the boundary conditions referred to in Eqn. (2) are designed for the rendezvous problem, this results in the well-known bang-bang optimal control problem, discussed in detail by Lawden[1]. However, herein the boundary conditions are designed such that the initial and final points lie on the desired orbits without specifying the position, or true anomaly, on either orbit.

As a brief review, the optimal thrust direction for this problem is

$$\mathbf{e}_T = \frac{\lambda_\mathbf{v}}{|\lambda_\mathbf{v}|} \tag{6}$$

where $\lambda_\mathbf{v}$ is found from the following differential equations

$$\dot{\lambda}_r = \mu\left[\frac{\lambda_v}{r^3} - 3\frac{(\lambda_v{}^T r)r}{r^5}\right] \tag{7}$$

$$\dot{\lambda}_v = -\lambda_r \tag{8}$$

$$\dot{\lambda}_m = \frac{T}{m^2}\lambda_v{}^T e_T = \frac{T}{m^2}|\lambda_v| \tag{9}$$

The optimal thrust magnitude for this problem is a bang-bang solution. This is determined by applying the following switching law, Eqn. (10), to the switching function, Eqn. (11).

$$\begin{aligned} &H_S > 0, \quad T = T_{max} \\ &H_S < 0, \quad T = 0 \end{aligned} \tag{10}$$

$$H_S = \frac{|\lambda_v|}{m} - \frac{\lambda_m}{g_o I_{sp}} \tag{11}$$

We are interested in solutions of this problem with long transfer times and, therefore, large numbers of burns. There are many methods that have been successively used to compute n-burn transfers, where n is anywhere from 1 to about 6. However, fewer methods successively compute transfers for larger values of n. These methods for the former attempt to solve the optimal control problem either directly, indirectly, or with a hybrid of the two. In this report, we will assume that a mostly indirect method, such as BOUNDSCO or MBCM or that of Brusch[2], et. al, or of Redding[3] is being used.

One idea to obtain interesting solutions is to first compute some n-burn transfer, where n is generally less than the number of burns initially desired. Using this as a starting point, increase the allowed transfer time and compute the new transfer. Obviously, it is expected that the desired transfer is relatively similar to the starting transfer. This homotopy method seems to work well as long as the number of burns performed in the transfer do not need to increase so that optimality is satisfied. For example, in many cases BOUNDSCO is unable to find a three burn solution when the two burn solution to an almost identical problem is given as the initial guess. The Direction Correction Method has been developed to attempt to alleviate this difficulty. It's purpose is to find an n burn solution to an orbit transfer problem with allowed

transfer time $t_f + \delta t_f$ using an n-1 burn solution to the same problem but with allowed transfer time t_f.

Another idea is to patch together a set of n-burn transfers, where n is a small integer, usually unity, to produce an m-burn transfer, where m is the desired number of burns. This method requires that the sequence of transfer orbits be either guessed and iterated upon for optimality, or simply prespecified. From the theory of optimal control, this patched solution will be a suboptimal solution. However, possible analytical solutions for the one burn solution of two very close orbits may give a feedback guidance law. Since the drag model is only approximate for large numbers of burns it may be more important to have a good guidance law in terms of fuel-savings.

II. Direction Correction Method

The first idea, referred to herein as the Direction Correction Method, is based on the common homotopy strategy. A homotopy method, though slow in producing results, would be considered effective here as long as the number of burns does not change. It is expected, however, that one is going to be using this method to increase the transfer time so that the fuel consumed will be less. To understand the ensuing difficulty, we must study the history of a successful implementation of this homotopy method.

All parameters describing transfers in this section and below have been nondimensionalized such that the gravitational constant, μ, is unity. This nondimensionalization is accomplished through two parameters, $r^{\star}$ and $m^{\star}$ with units of length and mass, respectively. These are chosen appropriately to the problem and may be, for example, initial semimajor axis and initial mass, respectively. The following equations detail the calculation of nondimensional parameters, denoted by the '^' symbol, describing the transfer:

$$\hat{T} \equiv \frac{T/m^{\star}}{\mu/r^{\star 2}} \tag{12a}$$

$$(\hat{g}_o \hat{I}_{sp}) \equiv g_o I_{sp} \sqrt{r^{\star}/\mu} \tag{12b}$$

$$\hat{t}_f \equiv \frac{t_f}{\sqrt{r^{\star 3}/\mu}} \tag{12c}$$

The optimal transfer we will examine is a planar transfer under ideal gravity conditions. The transfer leaves an initial orbit with a semimajor axis of 2.239, eccentricity of 0.1160, and an argument of perigee of -85.94°. The orbit to be entered has a semimajor axis of 7.000, eccentricity of 0.7332, and an argument of perigee of 114.6°. The motor used to perform the transfer delivers a thrust of 0.01386 with an exit velocity of 0.3898. The allowed transfer time is 73.33. This transfer performed in two burns is shown in Figure 1 with its corresponding parameters in Table I. It was computed using the multiple-shooting method of BOUNDSCO[4]. The switching function for this transfer is shown in Figure 2a.

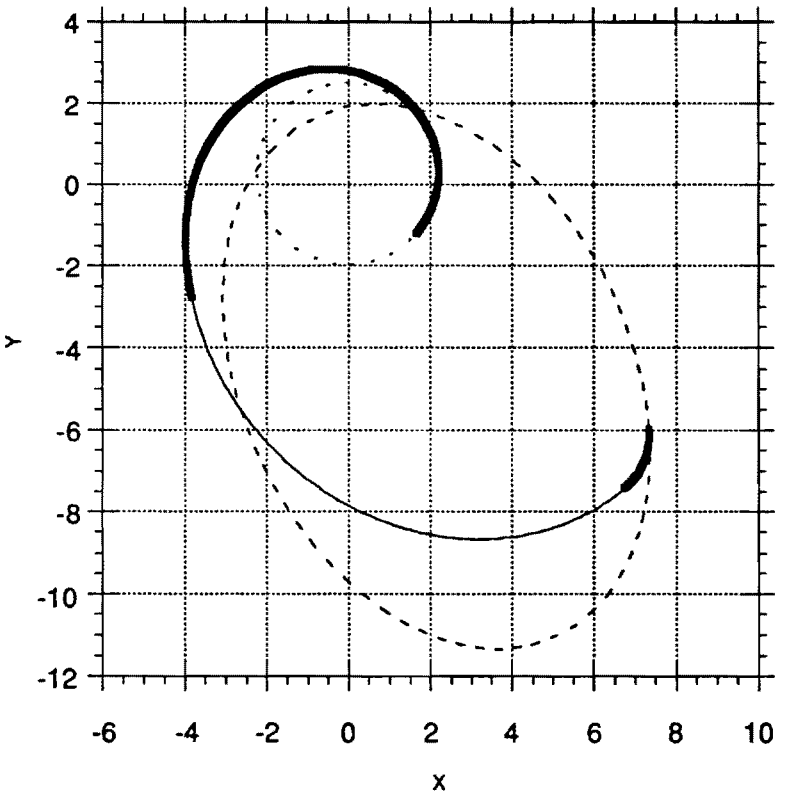


Figure 1. Transfer in Two Burns for Burn Addition Demonstration

$g_o I_{sp}$ =	0.3898	a_i=	2.239	ω_i=	-85.94°	a_f=	7.000	ω_f=	114.6°
T=	0.01386	e_i=	0.1160	t_f=	73.33	e_f=	0.7332	m_f=	0.5545

Table I. Parameters of the transfer shown in Figure 1

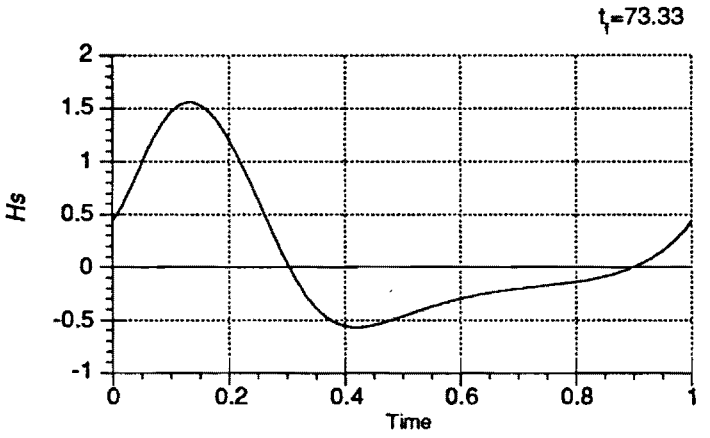


Figure 2a Switching Function for a Two Burn Transfer

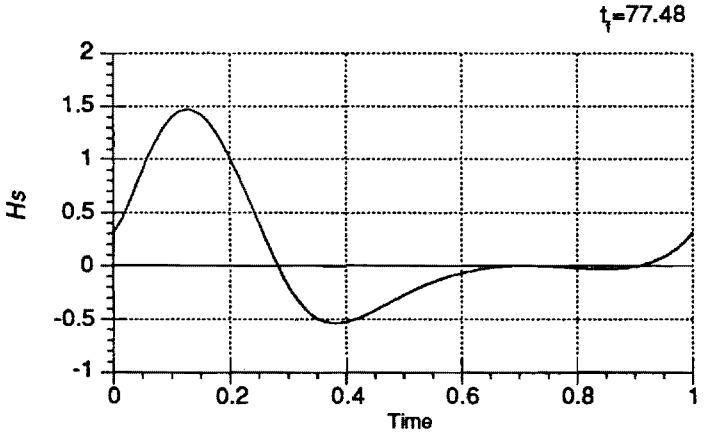


Figure 2b Switching Function for a Two or Three Burn Transfer

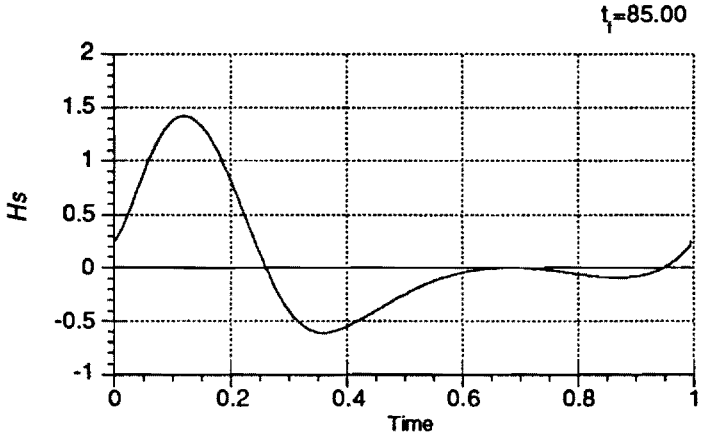


Figure 2c Switching Function for a Three Burn Transfer

The initial mass of the spacecraft was 1.6, the final mass is 0.5545. Now, suppose that a greater fuel savings is desired. As the allowed transfer time is increased from 73.33 to 77.48 and then to 85.00, the shown sequence of switching functions ($H_S(t)$ in Figs. 2a-c) will result. These show a clear indication of a new burn/coast being anticipated in the optimal solution. The orbit transfer corresponding to the switching function in Fig. 2c is plotted in Figure 3. The parameters of this transfer are identical to that of Fig. 1 except that the transfer time is longer, t_f=85, see Table II for the listing. Also, note that the final mass of this longer transfer is indeed larger than the shorter transfer, indicating a greater fuel savings.

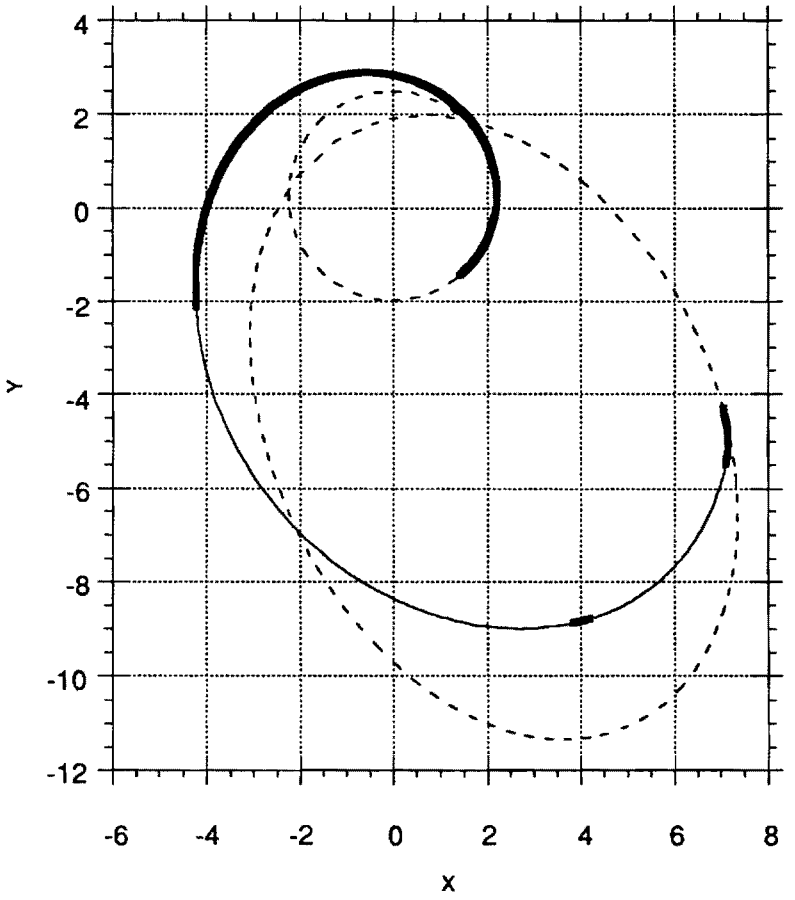


Figure 3 Transfer in Three Burns for Burn Addition Demonstration

$g_o I_{sp}$ =	0.3898	a_i=	2.239	ω_i=	-85.94°	a_f=	7.000	ω_f=	114.6°
T=	0.01386	e_i=	0.1160	t_f=	85.00	e_f=	0.7332	m_f=	0.6056

Table II. Parameters of the transfer shown in Figure 3

It has been seen in many cases that local minima and maxima of the switching function will move down or up on the graph as we examine successive solutions. As in Fig. 2b, once this critical point becomes a root of the switching function, we reach a point where the number of burn/coasts is somewhat indeterminate. Is this, in Fig. 2b, a two- or three- burn extremal? There are only two burns of finite length but there is a third that is infinitely small. This indeterminacy shows itself as a discontinuity in the slope of a plot of the initial guess versus the homotopy variable, transfer time, Figure 4.

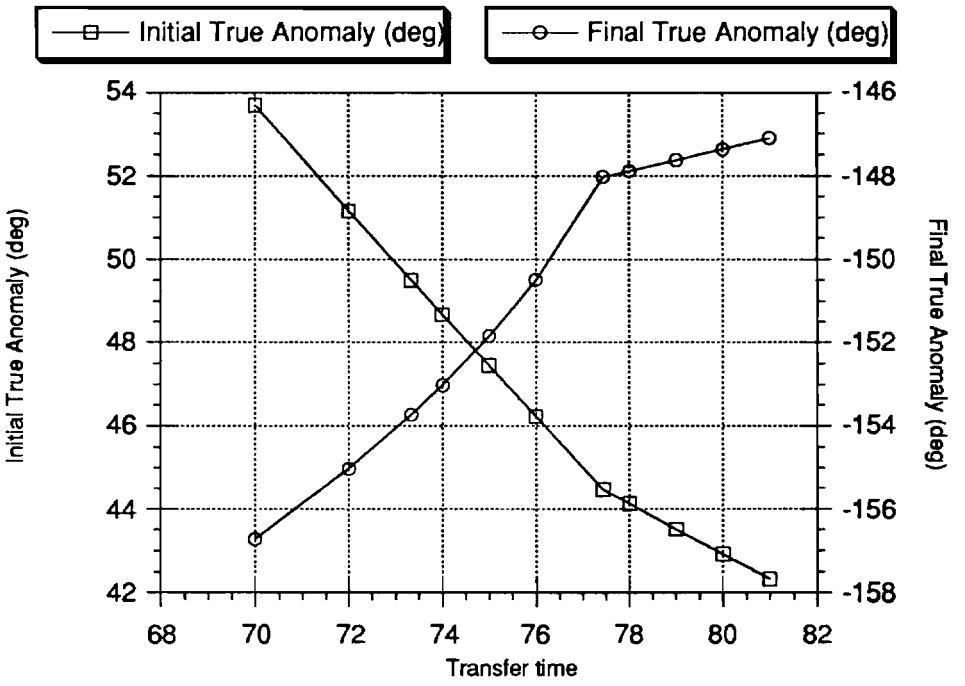


Figure 4 Plot of Initial and Final True Anomaly Values of Successive Solutions which Differ only in Transfer Time, t_f.

Figure 4 shows the initial and final true anomalies as a function of the allowed transfer time. The feature of interest here is the slope discontinuity (note that there is no point discontinuity) in both curves. The effect is not as prominent for the initial true anomaly as it is for the final, but it is still noticeable. As a result of this discontinuity there is difficulty in the homotopy method: the next solution may not converge because the method being used, based on the linear slope of previous points, is not calculating the correct initial state. To overcome this difficulty we must be able to compute the correct

slope, which should be the slope after $t_f = 77.48$, so that the homotopy method can continue.

The change in the initial state needs to be computed such that satisfaction of the boundary conditions is maintained and optimality is preserved. This problem shall be approached for the following general Two-Point-Boundary-Value-Problem (TPBVP):

$$\mathbf{C}(\mathbf{z}(0)) = 0 \quad m \text{ equations} \tag{13a}$$

$$\mathbf{D}\left(\mathbf{z}\left(t_f\right)\right) = 0 \quad m - n \text{ equations} \tag{13b}$$

$$\dot{\mathbf{z}}(t) = \mathbf{f}(\mathbf{z}(t)) \quad n \text{ equations} \tag{13c}$$

where $\mathbf{z}(t)$ is the state consisting of the original state plus the Lagrange multipliers, $\mathbf{f}(t)$ is the right-hand side of the original state dynamics plus the Euler-Lagrange equations, and $\mathbf{C}(\mathbf{z}(0))$ and $\mathbf{D}(\mathbf{z}(t_f))$ are the boundary conditions for the initial and final orbits, respectively.

Now, since we are interested in maintaining the boundary conditions, we set their variations equal to zero. First, the initial conditions from Eqn. (13a):

$$\delta\mathbf{C} = \left.\frac{\partial \mathbf{C}}{\partial \mathbf{z}}\right|_{\mathbf{z}(0)} \delta\mathbf{z}(0) = 0 \tag{14}$$

Next, a similar operation is performed on a vector describing the final conditions from Eqn. (13b). However, so that the initial state is referred to, it is necessary to invoke the transition matrix.

$$d\mathbf{D} = \left.\frac{\partial \mathbf{D}}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_f)} d\mathbf{z}\left(t_f\right) = 0 \tag{15a}$$

$$= \left.\frac{\partial \mathbf{D}}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_f)} \left(\delta\mathbf{z}\left(t_f\right) + \dot{\mathbf{z}}\left(t_f\right)dt_f\right) \tag{15b}$$

$$= \left.\frac{\partial \mathbf{D}}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_f)} \Phi\left(0, t_f\right)\delta\mathbf{z}(0) + \left.\frac{\partial \mathbf{D}}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_f)} \dot{\mathbf{z}}\left(t_f\right)dt_f = 0 \tag{15c}$$

Here, $d(\cdot)$ denotes a variation with variable time and $\Phi(0,t_f)$ is defined as the transition matrix, initialized at $t=0$, and evaluated at $t=t_f$, where

$$\dot{\Phi}(t_o,t) = \left.\frac{\partial \mathbf{f}(\mathbf{z}(t))}{\partial \mathbf{z}(t_o)}\right|_{\mathbf{z}(t)} \Phi(t_o,t) \tag{16a}$$

$$\Phi(t_o,t_o) = \mathbf{I} \tag{16b}$$

Now at each switching time, t_i (i=0,1...q), the switching function must be satisfied. So, we set the variation of the switching function, $H_S(\mathbf{z})$, equal to zero at each switching time, giving q scalar equations:

$$dH_s = \left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} d\mathbf{z}(t_i) = 0 \tag{17a}$$

$$= \left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} \left(\Phi(0,t_i)\delta\mathbf{z}(0) + \dot{\mathbf{z}}(t_i)dt_i\right) = 0 \tag{17b}$$

$$= \left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} \Phi(0,t_i)\delta\mathbf{z}(0) + \left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} \dot{\mathbf{z}}(t_i)dt_i = 0 \tag{17c}$$

Consideration of the switching function also calls attention to a necessary correction in the transition matrix calculation. At each switching point, there is a discontinuity in $\mathbf{f}(\mathbf{z}(t))$ due to the thrust being turned on or off. This discontinuity results in a 'jump' term for $\Phi(0,t_f)$. To calculate this term, we again must set the total change in H_S equal to zero.

$$H_s(\mathbf{z}(t_i)) = 0 \tag{18a}$$

$$dH_s = \left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} d\mathbf{z}(t_i) = 0 \tag{18b}$$

Now, recognize that the total variation in the state at the switching time t_i must be the same looking from either direction. This situation is illustrated in Figure 5.

$$d\mathbf{z}(t_i) = \delta\mathbf{z}(t_i^-) + \dot{\mathbf{z}}(t_i^-)dt_i \tag{19a}$$

$$= \delta\mathbf{z}(t_i^+) + \dot{\mathbf{z}}(t_i^+)dt_i \tag{19b}$$

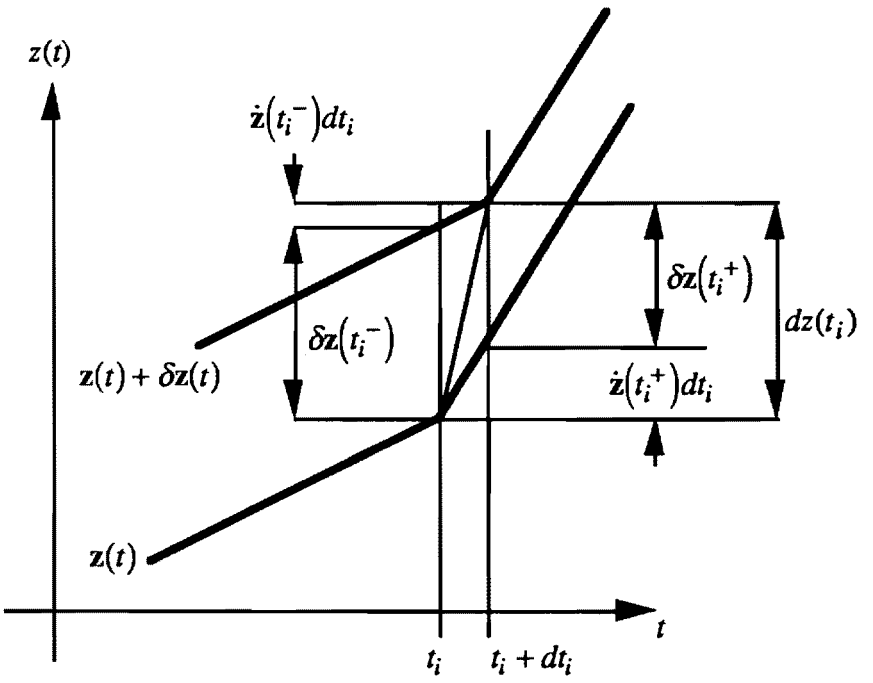


Figure 5. Illustration of Equations 19a and 19b

Substitute Eqn. (19b) into Eqn. (18b)

$$dH_s = \left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} \left(\delta\mathbf{z}(t_i^-) + \dot{\mathbf{z}}(t_i^-)dt_i\right) = 0 \tag{20}$$

Equation (13) can be solved immediately for dt_i which is then substituted into Eqns. (19a-b). This can manipulated to produce

$$\delta\mathbf{z}(t_i^+) = \delta\mathbf{z}(t_i^-) + \left(\dot{\mathbf{z}}(t_i^+) - \dot{\mathbf{z}}(t_i^-)\right) \frac{\left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} \delta\mathbf{z}(t_i^-)}{\left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} \dot{\mathbf{z}}(t_i^-)} \tag{21}$$

Equation (21) can be rewritten by inspection in terms of the transition matrix:

$$\Phi\left(t_i^+, t_i^-\right) = \left(\mathbf{I} + \frac{\left(\dot{\mathbf{z}}\left(t_i^+\right) - \dot{\mathbf{z}}\left(t_i^-\right)\right) \left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)}}{\left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_i)} \dot{\mathbf{z}}\left(t_i^-\right)} \right) \tag{22}$$

This is the jump matrix across the switching point t_i.

We must recognize that these variations are considered in a range of transfer times across which the number of switching points changes. Specifically, this is an addition of a burn or coast arc. The situation is illustrated in Figure 6. The assumed change in the switching function is shown at the top of the figure. The nominal solution's switching function has a touch point at t_c=t_a=t_b. The solution with a slight different transfer time has two new switching points, $t_a + dt_a$ and $t_b + dt_b$. The assumed change in one element of the state vector is shown at the bottom of Figure 6. The derivative, $\dot{\mathbf{z}}(t)$, is assumed equal before and after the new addition and to the nominal value, $\dot{\mathbf{z}}(t_c)$. The slope during the new burn is denoted c. To relate the two solutions across the arc, we write the following equation.

$$\delta\mathbf{z}(t_b + dt_b) = \delta\mathbf{z}(t_a + dt_a) + \left(c - \dot{\mathbf{z}}(t_b)\right)\left(dt_b - dt_a\right) \tag{23}$$

This relation has been verified using data from the example presented here.

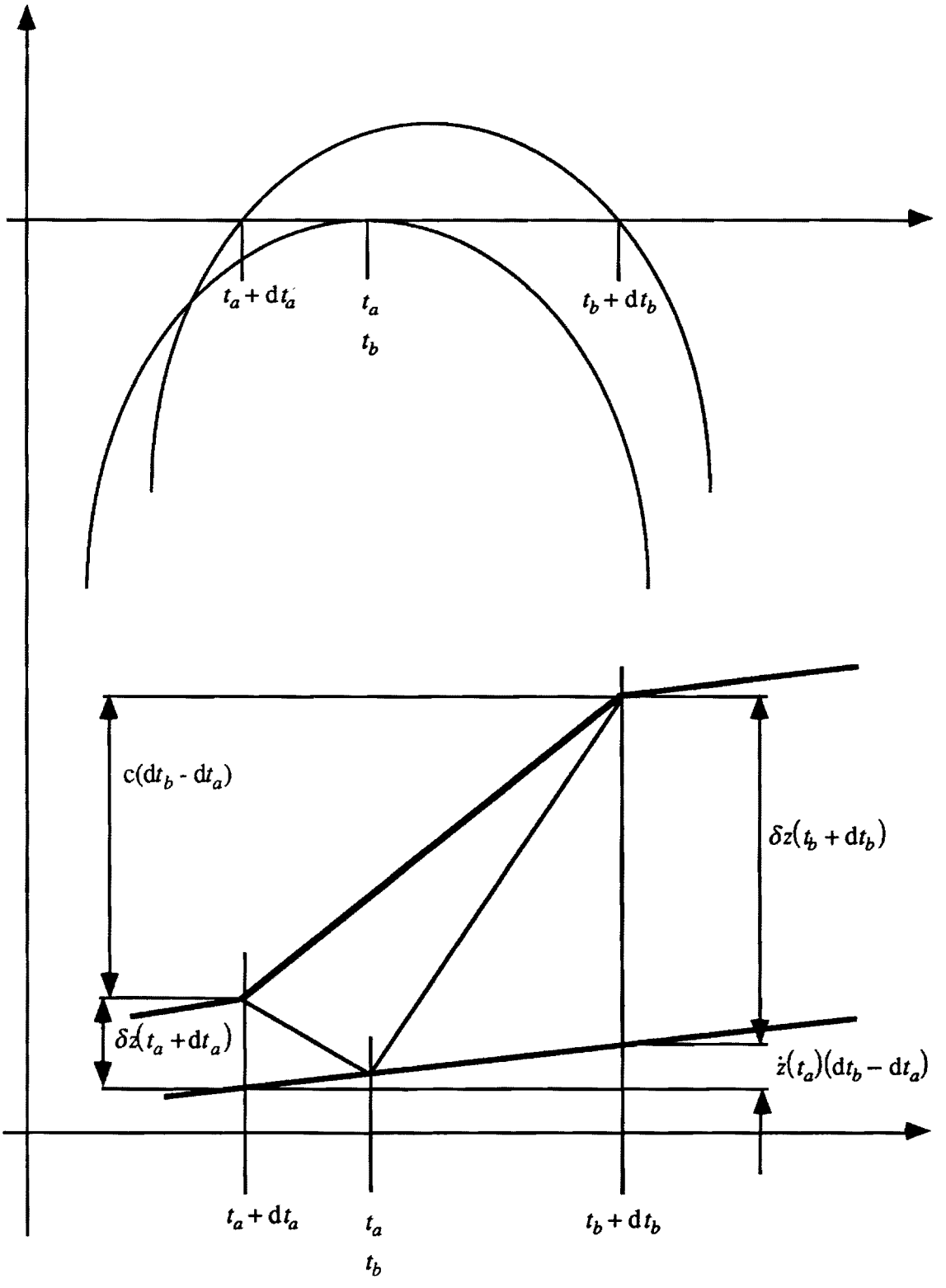


Figure 6: Model Describing Changes Incurred Between *n* and *n+1* Burn Solutions

Now, a model is required to locate the new switching points. We have looked into different models for this. The first model is a simple variational model, but unlike Eq (17a), second-order terms are considered. The equation of this model is

$$\Delta H_s \approx \frac{1}{2} d^2 H_s = \frac{1}{2} d\mathbf{z}(t_a)^{\mathrm{T}} \left.\frac{\partial^2 H_s}{\partial \mathbf{z} \partial \mathbf{z}}\right|_{\mathbf{z}(t_a)} d\mathbf{z}(t_a) = 0 \tag{24a}$$

$$= \frac{1}{2} \left(\delta \mathbf{z}(t_a) + \dot{\mathbf{z}}(t_a) dt_a\right)^{\mathrm{T}} \left.\frac{\partial^2 H_s}{\partial \mathbf{z} \partial \mathbf{z}}\right|_{\mathbf{z}(t_a)} \left(\delta \mathbf{z}(t_a) + \dot{\mathbf{z}}(t_a) dt_a\right) \tag{24b}$$

$$= \frac{1}{2} \left[\delta \mathbf{z}(t_a)^{\mathrm{T}} \left.\frac{\partial^2 H_s}{\partial \mathbf{z} \partial \mathbf{z}}\right|_{\mathbf{z}(t_a)} \delta \mathbf{z}(t_a) + 2 \delta \mathbf{z}(t_a)^{\mathrm{T}} \left.\frac{\partial^2 H_s}{\partial \mathbf{z} \partial \mathbf{z}}\right|_{\mathbf{z}(t_a)} \dot{\mathbf{z}}(t_a) dt_a \right.$$

$$\left. + \dot{\mathbf{z}}(t_a)^{\mathrm{T}} \left.\frac{\partial^2 H_s}{\partial \mathbf{z} \partial \mathbf{z}}\right|_{\mathbf{z}(t_a)} \dot{\mathbf{z}}(t_a) (dt_a)^2 \right] \tag{24c}$$

where the lesser of the two solutions is dt_a leaving the other to be dt_b. Unfortunately this model does not result in a sufficiently accurate answer for dt_a and dt_b.

We have also attempted to model the situation through the information on the placement of $t_c + dt_c$. Since this point can be defined as the point of zero slope, we can find with an analog of Eq. (17). The solution is, therefore,

$$dt_c = -\frac{\left.\frac{\partial \dot{H}_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_c)} \delta \mathbf{z}(t_c)}{\left.\frac{\partial \dot{H}_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_c)} \dot{\mathbf{z}}(t_c)} \tag{25}$$

To complete the model we need to have a point on the graph of ΔH_S and we need the curvature of H_S. The former can be had by rewriting Eq (17) for t_c and evaluating it at dt_c. We assume that the latter is well represented through a curve fit to the original switching function in the neighborhood of $H_S(t_c)$, denoted by k. in the following equation for ΔH_S.

$$\Delta H_s \approx k(t - t_c)^2 + \left(\left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_c)} d\mathbf{z}(t_c) \right) \tag{26}$$

The solutions we are interested in are

$$dt_a = dt_c - \sqrt{\frac{\left(\left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_c)} d\mathbf{z}(t_c)\right)}{k}} \tag{27a}$$

$$dt_b = dt_c + \sqrt{\frac{\left(\left.\frac{\partial H_s}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_c)} d\mathbf{z}(t_c)\right)}{k}} \tag{27b}$$

We have found that the solutions with this model are better than that with the previous model, but still not very accurate with errors greater than 10%. However, this accuracy may still prove to be well enough for BOUNDSCO to produce solutions. The intention here is merely to provide the TPBVP solver an initial guess closer to the n+1 burn solution.

Taking all of this together, a system of linear and non-linear equations can be written, starting with Eqs (14) and (15)

$$\left.\frac{\partial C}{\partial \mathbf{z}}\right|_{\mathbf{z}(0)} \Phi(0, t_a + dt_a)\delta\mathbf{z}(t_a + dt_a) = 0 \tag{28a}$$

$$\left.\frac{\partial D}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_f)} \Phi(t_b + dt_b, t_f)\mathbf{f}(\delta\mathbf{z}(t_a + dt_a)) = -\left.\frac{\partial D}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_f)} \dot{\mathbf{z}}(t_f) dt_f \tag{28b}$$

where $\mathbf{f}(\delta\mathbf{z})$ refers to the right-hand side of Eq. (23) as a function of $\delta\mathbf{z}(t_a + dt_a)$. The solution to this system is $\delta\mathbf{z}(t_a + dt_a)$, The transition matrix can be used to give the change in the initial state required to produce the desired solution. Then the variation of each switching point can be found one at time using Eqn. (17c).

The solution information can easily be put into a form useful for a variety of numerical methods. For example, the change $\delta\mathbf{z}(0)$ can be propagated through the transition matrix to calculate the changes at each node point for a multiple point shooting method. This method is still under development but shows promise as relatively simple way of getting to the n+1 burn solution in the right direction.

Once we have the ability to find optimal solutions with successively increasing transfer times, there is another characteristic of the extremals that may be encountered. Experience has shown that the length of the new burn will increase monotonically as the

transfer time is increased and usually the situation detailed above will be repeated so that the number of burns will increase again. However, there are cases where the cycle ends and the transversality condition, giving the optimal transfer time, is satisfied and there may be no nearby solution that has better performance. The following solution is an example. It is a descent trajectory from an orbit with a semimajor axis of 3.847, eccentricity of 0.02378, and an argument of perigee of 0°. The transfer terminates at an orbit with a semimajor axis of 1.500, eccentricity of 0.3333, and an argument of perigee of 0°. The motor used to perform the transfer delivers a thrust of 0.03 with an exit velocity of 1.313. The allowed transfer time is 19.05. This transfer performed in two burns is shown in Figure 7a. It also was computed using the multiple-shooting method of BOUNDSCO. The switching function for this solution is shown in Figure 7b.

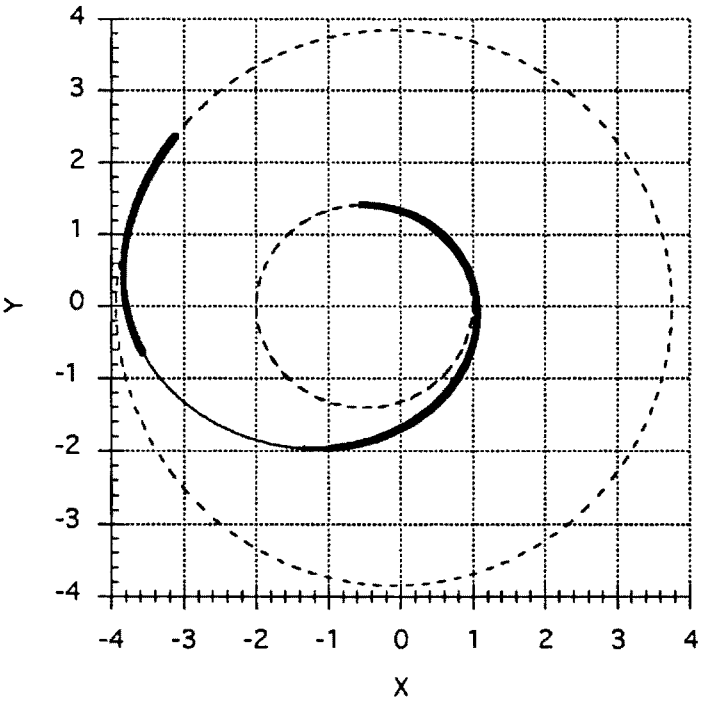


Figure 7a: Two Burn Extremal with Transversality Converged

$g_o I_{sp}$ =	1.313	a_i=	3.847	ω_i=	0.000°	a_f=	1.500	ω_f=	0.000°
T=	0.03	e_i=	0.02378	t_f=	19.05	e_f=	0.3333	m_f=	1.214

Table III. Parameters of the Transfer Shown in Figure 7

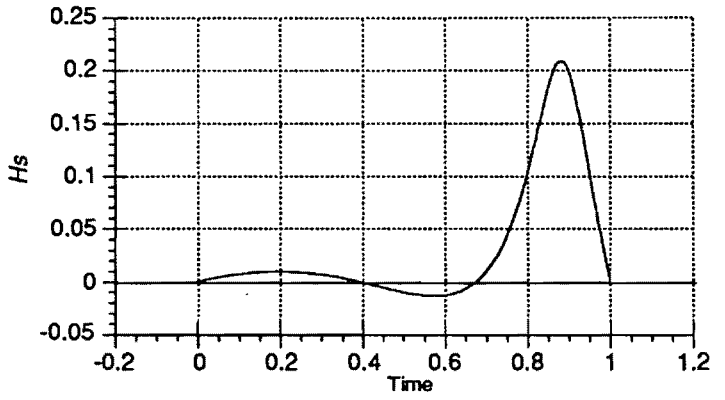


Figure 7b: Switching Function for Two Burn Extremal in Figure 7a

This solution was presented previously[5], however, with one difference, oblateness and atmospheric drag were included in the dynamics. It was found that with these terms removed, the transversality condition could be converged. It was also observed that the initial and final points of the switching function were driven to zero. There is certainly no conflict here in terms of optimality: the initial and final points are now switching points.

III. PATCHED TRANSFER METHOD

The second idea was inspired in part by the work of others. Zondervan, et. al made some simple guidance observations[6], specifically that in some regions the primer vector is relatively constant in a velocity-fixed reference frame. This implies that a simple control law is available in some cases. Marec presents a solution to the orbit correction problem[7]. This motivated a notion that solutions to linearized and/or approximated problems were available. In this spirit a solution was obtained for the optimal transfer between two close orbits. The transfer leaves a circular initial orbit with a radius of 1.038. The orbit to be entered has a semimajor axis of 1.069, eccentricity of 0.02633, and an argument of perigee of -50°. The motor used to perform the transfer delivers a thrust of 0.01438 with an exit velocity of 0.3861. The allowed transfer time is 1.553. This transfer is performed in one burn and is shown in Figure 8a. It was computed using the multiple-shooting method of BOUNDSCO. The switching function for this transfer is shown in Figure 8b.

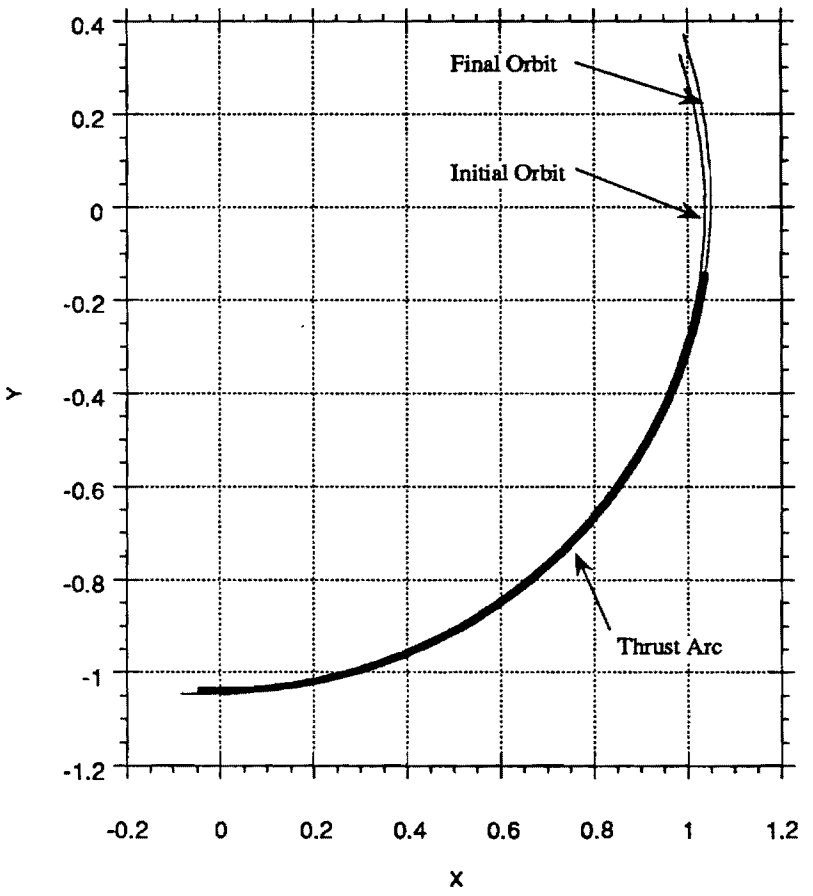


Figure 8a: One Burn Transfer Between Close Orbits: An Example of a Solution with a Simple Optimal Control

$g_o I_{sp}$ =	0.3861	a_i=	1.038	ω_i=	n/a	a_f=	1.069	ω_f=	-50°
T=	0.03	e_i=	0.000	t_f=	1.553	e_f=	0.02633	m_f=	1.542

Table IV. Parameters of the Transfer Shown in Figure 8a

Most interesting about this transfer is the simplicity of the control. Over this short transfer between a circular orbit and a close target orbit, the optimal control of the thrust angle is linear in time. And, in addition, we find that the control direction is almost coincident with the velocity direction.

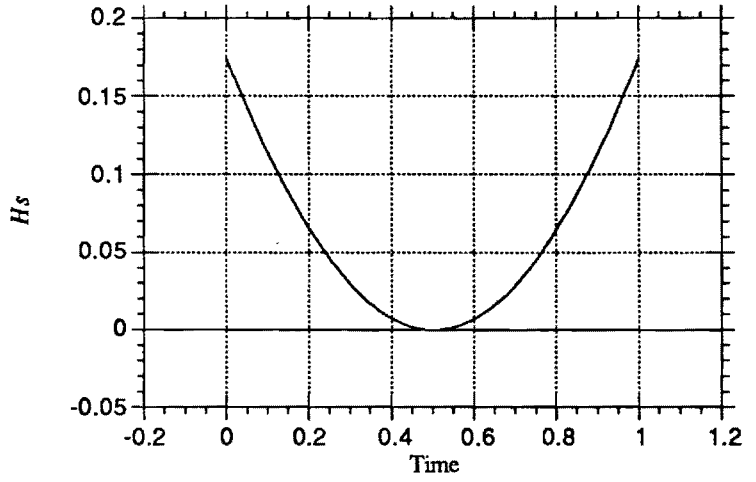


Figure 8b: Switching Function for One Burn Transfer Shown in Figure 8a

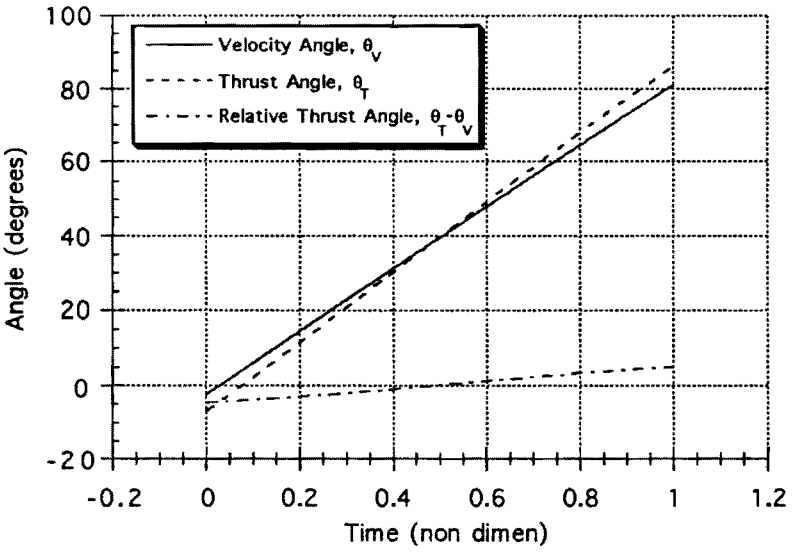


Figure 8: Plot of Thrust Direction, the Optimal Control, Alongside the Angle of the Velocity Vector.

To match this transfer analytically, a modified optimal control problem is considered. The dynamics for this problem are again the equations of orbital motion, however, this time the state is defined relative to the initial orbit. Assuming that the distance from the initial orbit is small compared to the radius of the initial orbit, we

ignore all terms to the order of $(\delta r/\rho)^2$. This assumption results in the following dynamics:

$$\delta\dot{\mathbf{r}} = \delta\mathbf{v} \tag{29a}$$

$$\delta\dot{\mathbf{v}} = \frac{T}{m}\mathbf{e}_T + 3\frac{\mu(\delta\mathbf{r}\bullet\rho)}{\rho^5}\rho - \frac{\mu}{\rho^3}\delta\mathbf{r} \tag{29b}$$

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{29c}$$

Here, $\delta\mathbf{r}=[x\ y]^T$ and $\delta\mathbf{v}=[u\ v]^T$, $\mathbf{e}_T$ is the thrust direction, T is the thrust, m is the mass, μ is the gravitational constant, and ρ represents the initial orbit which satisfies identical dynamics but without the thrust term. Now, assuming that the initial orbit is circular, these can be rewritten as:

$$\dot{x} = u \tag{30a}$$

$$\dot{y} = v \tag{30b}$$

$$\dot{u} = \frac{T}{m}e_x + \frac{\mu}{\rho^3}\left[3(x\cos(\omega t) + y\sin(\omega t))\cos(\omega t) - x\right] \tag{30c}$$

$$\dot{v} = \frac{T}{m}e_y + \frac{\mu}{\rho^3}\left[3(x\cos(\omega t) + y\sin(\omega t))\cos(\omega t) - y\right] \tag{30d}$$

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{30e}$$

Writing the Hamiltonian for this system and evaluating the Euler-Lagrange equations results in the following differential equations involving the costates:

$$\dot{\lambda}_u = -\lambda_x \tag{31a}$$

$$\dot{\lambda}_v = -\lambda_y \tag{31b}$$

$$\dot{\lambda}_x = -\lambda_u\frac{\mu}{\rho^3}\left(3\cos^2(\omega t) - 1\right) - \lambda_v\frac{\mu}{\rho^3}3\cos(\omega t)\sin(\omega t) \tag{31c}$$

$$\dot{\lambda}_y = -\lambda_u\frac{\mu}{\rho^3}(3\cos(\omega t)\sin(\omega t)) - \lambda_v\frac{\mu}{\rho^3}\left(3\sin^2(\omega t) - 1\right) \tag{31d}$$

$$\dot{\lambda}_m = -\frac{T}{m^2\sqrt{\lambda_u{}^2 + \lambda_v{}^2}} \tag{31e}$$

We also learn that the control, $\mathbf{e}_T$ is

$$\mathbf{e}_T = \frac{1}{\sqrt{\lambda_u{}^2 + \lambda_v{}^2}} \begin{bmatrix} \lambda_u \\ \lambda_v \end{bmatrix} \tag{32}$$

and the control T is bang-bang, governed by the switching function, H_T, as

$$H_T = \frac{\sqrt{\lambda_u{}^2 + \lambda_v{}^2}}{m} - \frac{\lambda_m}{g_o I_{sp}} \tag{33}$$

$$\begin{aligned} &H_T > 0, \quad T = T_{max} \\ &H_T < 0, \quad T = 0 \end{aligned} \tag{34}$$

Pleasantly, Eqns. (31) happen to be the equations for the costates on a coast arc coinciding with the initial orbit. In fact, this result is not limited to the assumption of a circular orbit. The coast arc costates have been solved by Lawden and other authors[8,9]. It also can be shown that Eqns. (31) are, in fact, identical to Eqns. (30), without the thrust terms, up to sign. Therefore, once we solve the system in Eqns. (31) we have the homogeneous solution to the system in Equations (30). Now to solve the differential Eqns (26), they must first be rewritten in a more useful form:

$$\begin{bmatrix} \dot{\lambda}_x \\ \dot{\lambda}_y \end{bmatrix} = -\begin{bmatrix} \ddot{\lambda}_u \\ \ddot{\lambda}_v \end{bmatrix} = -3l \begin{bmatrix} cos(\omega t) \\ sin(\omega t) \end{bmatrix} \begin{bmatrix} cos(\omega t) & sin(\omega t) \end{bmatrix} \begin{bmatrix} \lambda_u \\ \lambda_v \end{bmatrix} + l \begin{bmatrix} \lambda_u \\ \lambda_v \end{bmatrix} \tag{35}$$

where $l \equiv \mu/\rho^3 = \omega^2$. Now, define vectors $\mathbf{e}_\rho(t)$ and $\mathbf{e}_\omega(t)$, as the radial and circumferential directions associated with the initial orbit over time t. This can now be written as

$$\ddot{\boldsymbol{\lambda}} = 3l\mathbf{e}_\rho \mathbf{e}_\rho{}^T \boldsymbol{\lambda} - l\boldsymbol{\lambda} \tag{36}$$

where $\boldsymbol{\lambda} \equiv \begin{bmatrix} \lambda_u & \lambda_v \end{bmatrix}^T$. Multiply Eqn. (36) by $\mathbf{e}_\rho{}^T$ and $\mathbf{e}_\omega{}^T$, respectively to obtain

$$\mathbf{e}_\rho{}^T \ddot{\boldsymbol{\lambda}} = 3l\mathbf{e}_\rho{}^T \boldsymbol{\lambda} - l\mathbf{e}_\rho{}^T \boldsymbol{\lambda} = 2l\mathbf{e}_\rho{}^T \boldsymbol{\lambda} = 2\omega^2 \mathbf{e}_\rho{}^T \boldsymbol{\lambda} \tag{37a}$$

$$\mathbf{e}_\omega{}^T \ddot{\boldsymbol{\lambda}} = 3l\mathbf{e}_\omega{}^T \mathbf{e}_\rho \mathbf{e}_\rho{}^T \boldsymbol{\lambda} - l\mathbf{e}_\omega{}^T \boldsymbol{\lambda} = -l\mathbf{e}_\omega{}^T \boldsymbol{\lambda} = -\omega^2 \mathbf{e}_\omega{}^T \boldsymbol{\lambda} \tag{37b}$$

To complete the simplifications, it is necessary to obtain an expression for left-hand side of Eqn. (36) in terms of $\mathbf{e}_\rho$ and $\mathbf{e}_\omega$. That expression is

$$\ddot{\boldsymbol{\lambda}} = \frac{d^2}{dt^2}\left(\lambda_\rho \mathbf{e}_\rho + \lambda_\omega \mathbf{e}_\omega\right) = \frac{d}{dt}\left(\left(\dot{\lambda}_\rho - \omega\lambda_\omega\right)\mathbf{e}_\rho + \left(\dot{\lambda}_\omega + \omega\lambda_\rho\right)\mathbf{e}_\omega\right)$$
$$= \left(\ddot{\lambda}_\rho - 2\omega\dot{\lambda}_\omega - \omega^2\lambda_\rho\right)\mathbf{e}_\rho + \left(\ddot{\lambda}_\omega + 2\omega\dot{\lambda}_\rho - \omega^2\lambda_\omega\right)\mathbf{e}_\omega \tag{38}$$

Using this expression, Eqns. (37) become

$$\ddot{\lambda}_\rho - 2\omega\dot{\lambda}_\omega - \omega^2\lambda_\rho = 2\omega^2\lambda_\rho \tag{39a}$$
$$\ddot{\lambda}_\omega + 2\omega\dot{\lambda}_\rho - \omega^2\lambda_\omega = -\omega^2\lambda_\omega \tag{39b}$$

This can be represented with a matrix differential equation,

$$\begin{bmatrix} \dot{\lambda}_\rho \\ \dot{\lambda}_\omega \\ \dot{\lambda}_1 \\ \dot{\lambda}_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 3\omega^2 & 0 & 0 & 2\omega \\ 0 & 0 & -2\omega & 0 \end{bmatrix} \begin{bmatrix} \lambda_\rho \\ \lambda_\omega \\ \lambda_1 \\ \lambda_2 \end{bmatrix} \tag{40}$$

where $\lambda_1 \equiv d\lambda_\rho/dt$ and $\lambda_2 \equiv d\lambda_\omega/dt$. The solution to this system is

$$\begin{bmatrix} \lambda_\rho \\ \lambda_\omega \\ \lambda_1 \\ \lambda_2 \end{bmatrix} = a\begin{bmatrix} 2 \\ -3\omega t \\ 0 \\ 3\omega \end{bmatrix} + b\begin{bmatrix} cos(\omega t) \\ -2sin(\omega t) \\ -\omega sin(\omega t) \\ -2\omega cos(\omega t) \end{bmatrix} + c\begin{bmatrix} sin(\omega t) \\ 2cos(\omega t) \\ \omega cos(\omega t) \\ -2\omega sin(\omega t) \end{bmatrix} + d\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \tag{41}$$

where a, b, c, and d are independent constants. The vector $\boldsymbol{\lambda}$ can be interpreted as the thrust direction in a reference frame fixed to the radius of the initial orbit, referred to here as the initial orbit reference frame. From the solution above, Eqn. (41), we see that there are four modes of the thrust direction. The mode associated with d is fixed with respect to the initial orbit reference frame. The mode associated with a is not fixed to that frame but is very simply described in it. The last two modes do not seem as well described in this frame.

To be sure, we would like to see that the approximate state dynamics given in Eqns (29) and (30) closely match those given in Eqns (3,4,5). To validate the approximate dynamics, it was simplest to simulate both systems using the same control. The most obvious choice for this control is the optimal control from the transfer in Fig.

7a. Figure 9 shows the results of the simulation. In this figure, "Delta-" states refer to the states from Fig. 7a with the initial orbit states subtracted, producing the desired plot for $\delta \mathbf{r}$. The "X1,Y1," etc. states refer to the states obtained by integrating Eqns. (30). The results seen in this figure are very promising: there is almost exact agreement between the two state histories. In fact, the worst error between the two at the end of the transfer is only about 1.5%.

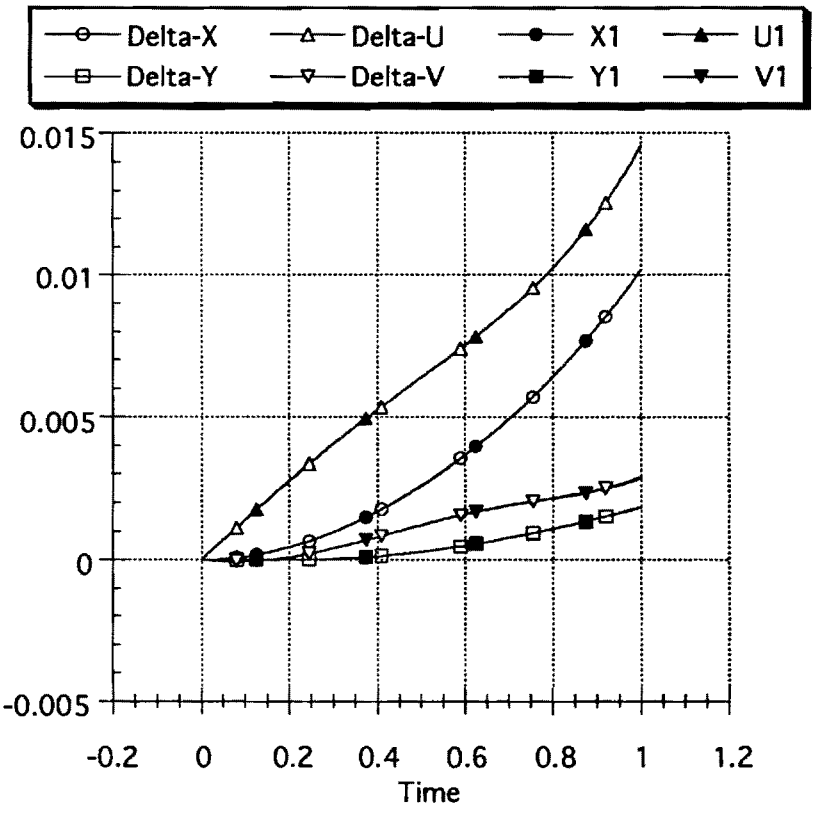


Figure 9: Validation Plot for the Dynamics in Equations (29) for the Transfer shown in Figure 7a

IV. CONCLUSIONS

The development of the Direction Correction Method is proceeding rather well. The ideas that it is based upon have been validated individually. At this point, the only weak link is the prediction of the new switching points. Testing of the method will be required in order to determine just how critical is the accuracy of that prediction.

The Patched Transfer Method is very promising. The dynamics resulting from assumptions made closely matches the dynamics before the approximations. Also, the simplicity of the resulting optimal control problem promises a state feedback guidance law. The usefulness of these results will outweigh the loss in accuracy. However, much more analysis must be performed to completely validate the linearized problem and its solution. Specifically, the approximate optimal control solution must be compared to exact solution; based on the agreement of the state, positive results are expected, but they must be verified.

V. REFERENCES

[1]Lawden, D.F., Optimal Trajectories for Space Navigation, Butterworths, London, 1963.

[2]Brusch, R.G. and Vincent, T.L., "Low-Thrust, Minimum-Fuel, Orbital Transfers," *Astronautica Acta*, Vol. 16, pp 65-74.

[3]Redding, D.C., "Optimal Low-Thrust Transfers to Geosynchronous Orbit," NASA Lewis SUDAAR 539, Cleveland, Ohio 44135, Sept. 1983.

[4]Oberle, H. J., BOUNDSCO - Hinweise zur Benutzung des Mehrzielverfahrens für die numerische Lösung von Randwerproblemen mit Schaltbedingungen, Hamburger Beiträge zur Angewandten Mathematik, Berichte 6, 1987.

[5]Chuang, C.-H., Goodson, T.G., and Hanson, J. "Theory and Computation of Optimal Low- And Medium-Thrust Orbit Transfers," AIAA Guidance, Navigation, and Control Conference Paper, September 1992.

[6]Zondervan, K.P., Lincoln, L.J., and Caughey, T.K., "Optimal Low-Thrust, Three-Burn Orbit Transfers with Large Plane Changes," *Journal of the Astronautical Sciences*, Vol. 32, No. 3, 1984, pp. 407-427

[7]Marec, J.P., Optimal Space Trajectories, Elsevier Scientific Publishing, New York, 1979.

[8]Glandorf, D.R., "Lagrange Multipliers and the State Transition Matrix for Coasting Arcs," *AIAA Journal*, Vol. 7, No. 2, Feb. 1969, pp 363-365.

[9]Eckenwiler, M. W., "Closed-Form Lagrangian Multipliers for Coast Periods of Optimum Trajectories," *AIAA Journal*, Vol. 3, No. 6, June 1965, pp 1149-1151.

SEMI-ANNUAL STATUS REPORT

Submitted to:	NASA Marshall Space Flight Center
Grant Title:	Theory and Computation of Optimal Low- and Medium-Thrust Transfers
Grant Number:	NAG8-921
Principal Investigator /Project Director:	Dr. C.-H. Chuang School of Aerospace Engineering Georgia Institute of Technology Atlanta, GA 30332-0150 Phone: (404) 894-3075 Fax: (404) 894-2760 E-mail: ch.chuang@aerospace.gatech.edu
Research Assistant:	Troy Goodson School of Aerospace Engineering Georgia Institute of Technology
Period Covered:	January 7, 1993 to July 6, 1994
Date of Submission:	July 6, 1994

TABLE OF CONTENTS

ABSTRACT 1
I. INTRODUCTION 1
II. DIRECTION CORRECTION METHOD PROGRESS 4
III. PATCHED TRANSFER METHOD PROGRESS 5
IV. CONCLUSIONS 7
V. REFERENCES 8

Progress in Computing Fuel-Optimal Orbit Transfers in Large Numbers of Burns

C.-H. Chuang and Troy D. Goodson
Georgia Institute of Technology, Atlanta, Georgia 30332

ABSTRACT

This report describes the current state of development of methods for calculating optimal orbital transfers with large numbers of burns. Reported on first is the homotopy-motivated and so-called Direction Correction method. So far, this method has been partially tested with one solver, the final step has yet to be implemented. Second is the Patched Transfer method. This method is rooted in some simplifying approximations made on the original optimal control problem. The transfer is broken up into single-burn segments, each single-burn solved as a predictor step and the whole problem then solved with a corrector step.

I. INTRODUCTION

Electric propulsion, with its high specific impulse, promises very low fuel consumption but it produces less thrust than its counterparts. If one wants to use electric propulsion, one needs to be prepared to tolerate the long transfer times that will likely be incurred. The greater time spent thrusting must be spent wisely if fuel savings are to be realized. Furthermore, the effects of Earth's oblateness and atmospheric drag become more significant on the orbits of long transfer times.

To spend the thrusting time wisely, form an optimal control problem to maximize the mass at the end of the transfer. This, therefore, is the cost function

$$J = m(t_f) \tag{1}$$

subject to the boundary conditions

$$\psi\left(\mathbf{r}(0), \mathbf{v}(0), \mathbf{r}\left(t_f\right), \mathbf{v}\left(t_f\right)\right) = 0 \tag{2}$$

and the state dynamics

$$\dot{\mathbf{r}} = \mathbf{v} \tag{3}$$

$$\dot{\mathbf{v}} = \frac{T}{m}\mathbf{e}_T - \frac{\mu}{r^3}\mathbf{r} \tag{4}$$

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{5}$$

where $\mathbf{e}_T$ is the thrust direction, a unit vector, and the thrust magnitude, T, is limited between zero and some maximum value T_{max}, μ is the gravitational constant, g_o is the gravitational acceleration at sea-level, and I_{sp} is the specific impulse of the motor. Sometimes $g_o I_{sp}$ is referred to as the exit velocity of the motor.

This results in the well-known bang-bang optimal control problem, discussed in detail by Lawden[1]. However, where the boundary conditions are often designed for the rendezvous problem, herein the boundary conditions are designed such that the initial and final points lie on the desired orbits without specifying the position, or true anomaly, on either orbit.

As found using the Euler-Lagrange necessary conditions, the optimal thrust direction for this problem is

$$\mathbf{e}_T = \frac{\boldsymbol{\lambda}_\mathbf{v}}{|\boldsymbol{\lambda}_\mathbf{v}|} \tag{6}$$

where $\boldsymbol{\lambda}_\mathbf{v}$ is found from the following differential equations

$$\dot{\boldsymbol{\lambda}}_\mathbf{r} = \mu\left[\frac{\boldsymbol{\lambda}_\mathbf{v}}{r^3} - 3\frac{(\boldsymbol{\lambda}_\mathbf{v}{}^{\mathrm{T}}\mathbf{r})\mathbf{r}}{r^5}\right] \tag{7}$$

$$\dot{\boldsymbol{\lambda}}_\mathbf{v} = -\boldsymbol{\lambda}_\mathbf{r} \tag{8}$$

$$\dot{\lambda}_m = \frac{T}{m^2}\boldsymbol{\lambda}_\mathbf{v}{}^{\mathrm{T}}\mathbf{e}_T = \frac{T}{m^2}|\boldsymbol{\lambda}_\mathbf{v}| \tag{9}$$

The optimal thrust magnitude for this problem is a bang-bang solution. Polarity for the on-off control is determined by applying the following switching law, Eqn. (10), to the switching function, Eqn. (11).

$$\begin{aligned} &H_S > 0, \quad T = T_{max} \\ &H_S < 0, \quad T = 0 \end{aligned} \tag{10}$$

$$H_S = \frac{|\lambda_v|}{m} - \frac{\lambda_m}{g_o I_{sp}} \tag{11}$$

Solutions of this problem with long transfer times and, therefore, large numbers of burns are desired. There are many methods that have been successfully used to compute n-burn transfers, where n is anywhere from 1 to about 6. However, fewer methods successfully compute transfers for larger values of n. Methods for the former attempt to solve the optimal control problem either directly, indirectly, or as a hybrid of the two. In this report, assume that a mostly indirect method, such as BOUNDSCO[2] or MBCM[3] or that of Brusch[4], et. al, or of Redding[5] is being used.

One idea to obtain interesting solutions is to first compute some n-burn transfer, where n may be less than the number of burns desired. Using this as a starting point, increase the allowed transfer time and compute the new transfer. It is expected that the new transfer is relatively similar to the starting transfer. New transfers are then successfully produced this way until the desired transfer is reached. This homotopy method seems to work well as long as the number of burns performed in the transfer does not need to increase so that optimality is satisfied. For example, in many cases BOUNDSCO is unable to find a three burn solution when the two burn solution to an almost identical problem is given as the initial guess. Introduced in this report, the Direction Correction Method is an attempt to alleviate this difficulty. Its purpose is to find an n+1 burn solution to an orbit transfer problem with allowed transfer time $t_f + dt_f$ using an n burn solution to the same problem but with allowed transfer time t_f.

Another idea is to patch together a set of n-burn transfers, where n is a small integer, perhaps unity, to produce an m-burn transfer, where m is the desired number of burns. This method requires that the sequence of transfer orbits be either guessed and iterated upon for optimality, or simply prespecified. From the theory of optimal control, this patched solution will be a suboptimal solution. This idea will be referred to herein as the Patched Transfer method.

More than likely, once an optimal transfer has been computed, interest will shift to developing a guidance law. Possible analytical solutions found from consideration of the patched transfer method for the one burn solution of two very close orbits may give a simple guidance law.

II. DIRECTION CORRECTION METHOD PROGRESS

The first idea, referred to herein as the Direction Correction method, is based on the common homotopy strategy. The Direction Correction is designed to aid a homotopy strategy in calculating successive optimal transfers. In particular, the difficulty arises when the desired transfer has one more burn arc than the current computed transfer.

The method is attractive because it only requires the solution of a relatively small set of nonlinear equations. These equations are of the following form

$$\left.\frac{\partial C}{\partial \mathbf{z}}\right|_{\mathbf{z}(0)} \Phi(0, t_a + dt_a)\delta\mathbf{z}(t_a + dt_a) = 0 \tag{12a}$$

$$\left.\frac{\partial D}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_f)} \Phi(t_b + dt_b, t_f)\mathbf{f}(\delta\mathbf{z}(t_a + dt_a)) = -\left.\frac{\partial D}{\partial \mathbf{z}}\right|_{\mathbf{z}(t_f)} \dot{\mathbf{z}}(t_f)dt_f \tag{12b}$$

For reasons given in a previous report and a paper submitted to the 1994 AIAA Guidance, Navigation, and Control conference[6], both equations are evaluated at time t_a + dt_a. The first equation propagates a guess made for this instant in time back to the initial time, using it to check a condition on the boundary conditions at the initial time, denoted $C(\mathbf{z})$ where $\mathbf{z}$ is the state vector. The second equation is a similar situation, except that it is applied to the boundary conditions at the final time, denoted $D(\mathbf{z})$. The function $\mathbf{f}(\mathbf{z})$ takes into account the fact that the number of burns in the desired transfer, $\mathbf{z}(t)$ + $\delta\mathbf{z}(t)$, is one greater than in the computed transfer, $\mathbf{z}(t)$.

The solution information can easily be put into a form useful for a variety of numerical methods. For example, the change $\delta\mathbf{z}(0)$ can be propagated through the transition matrix to calculate the changes at each node point for a multiple point shooting method. This method is still under development but shows promise as relatively simple way of getting to the n+1 burn solution.

Using the IMSL routine DNEQBF to solve Eqns. (12a-b), the method has been used to predict the correct change, or 'direction,' for an example. The algorithm starts with information from a given transfer. Then, it iteratively improves upon an initial guess, using DNEQBF. The method has produced an approximate solution to Eqns. (12a-b). Comparing this solution to the correct answer, errors are only about 4%. The final step is to add the solution to the computed transfer and attempt to converge the desired transfer with a solver such as BOUNDSCO. Although this has not been implemented yet, success is expected.

III. PATCHED TRANSFER METHOD PROGRESS

The second method was inspired in part by the work of others. Zondervan, et. al made some simple guidance observations[7], specifically that in some regions the primer vector is relatively constant in a velocity-fixed reference frame. This implies that a simple control law is available in some cases. Marec presents a solution to the orbit correction problem[8]. This motivated a notion that solutions to linearized and/or approximated problems were available. In this spirit a solution was obtained for the optimal transfer between two close orbits. This solution has been presented in [6].

Most interesting about this transfer was the simplicity of the control. Over this short transfer between a circular orbit and a close target orbit, the optimal control of the thrust angle was almost linear in time. And, in addition, the control direction was almost coincident with the velocity direction.

To review, a modified optimal control problem is considered. The dynamics for this problem are again the equations of orbital motion; however, this time the state is defined relative to the initial orbit. Assume that the distance from a reference orbit is small compared to the radius of the reference orbit and ignore all terms to the order of $(\delta r/\rho)^2$. This assumption results in the following dynamics:

$$\delta\dot{\mathrm{r}} = \delta\mathrm{v} \tag{13a}$$

$$\delta\dot{\mathrm{v}} = \frac{T}{m}\mathrm{e}_T + 3\frac{\mu(\delta\mathrm{r}\bullet\rho)}{\rho^5}\rho - \frac{\mu}{\rho^3}\delta\mathrm{r} \tag{13b}$$

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{13c}$$

Here, $\delta\mathbf{r}$ and $\delta\mathbf{v}$ represent displacement from an osculating orbit or the initial orbit, $\mathbf{e}_T$ is the thrust direction, T is the thrust, m is the mass, μ is the gravitational constant, and ρ represents the initial orbit which satisfies identical dynamics but without the thrust term.

Writing the Hamiltonian for this, the approximated system, gives

$$H = \lambda_{\mathrm{r}}{}^{\mathrm{T}}\delta\mathrm{v} + \lambda_{\mathrm{v}}{}^{\mathrm{T}}\left[\frac{T}{m}\mathrm{e}_T + 3\frac{\mu(\delta\mathrm{r}\bullet\rho)}{\rho^5}\rho - \frac{\mu}{\rho^3}\delta\mathrm{r}\right] - \lambda_m\frac{T}{g_o I_{sp}} \tag{14}$$

Evaluating the Euler-Lagrange equations results in the following differential equations involving the costates:

$$\dot{\lambda}_r = -3\frac{\mu(\lambda_v{}^T\rho)}{\rho^5}\rho + \frac{\mu}{\rho^3}\lambda_v \tag{15a}$$

$$\dot{\lambda}_v = -\lambda_r \tag{15b}$$

$$\dot{\lambda}_m = -\frac{T}{m^2|\lambda_v|} \tag{15c}$$

The control, $\mathbf{e}_T$ is

$$\mathbf{e}_T = \frac{\lambda_v}{|\lambda_v|} \tag{16}$$

and the control T is bang-bang, governed by the switching function, H_T, as

$$H_T = \frac{|\lambda_v|}{m} - \frac{\lambda_m}{g_o I_{sp}} \tag{17}$$

$$\begin{aligned} H_T > 0, \quad T = T_{max} \\ H_T < 0, \quad T = 0 \end{aligned} \tag{18}$$

Pleasantly, Eqns. (15) happen to be the differential equations for the costates on a coast arc coinciding with the initial orbit or the osculating orbit. The coast arc costates have been solved by Lawden and other authors[9,10]. However, it is Glandorf's formulation, actually based on work by Pines[11], that is currently being considered. His formulation is in the following form:

$$\begin{bmatrix} \lambda_v(t) \\ -\lambda_r(t) \end{bmatrix} = \mathrm{P}(t)\left[\mathrm{P}(t_0)\right]^{-1}\begin{bmatrix} \lambda_v(t_0) \\ -\lambda_r(t_0) \end{bmatrix} \tag{19}$$

Considering the form of the state dynamics, their solution can then be written as

$$\begin{bmatrix} \delta r(t) \\ \delta v(t) \end{bmatrix} = \mathrm{P}(t)\left[\mathrm{P}(t_0)\right]^{-1}\begin{bmatrix} \delta r(t_0) \\ \delta v(t_0) \end{bmatrix} + T\mathrm{P}(t)\int_{t_0}^{t}\left[\frac{\left[\mathrm{P}(\tau)\right]^{-1}\begin{bmatrix} 0 \\ \lambda_v(\tau) \end{bmatrix}}{m(\tau)|\lambda_v(\tau)|}\right]d\tau \tag{20}$$

An analytical expression for the integral has been rather elusive. Currently, work is focused on approximating the integral. For example, if the magnitude of the Lagrange multiplier is approximated as

$$|\lambda_v(\tau)| \approx g(\tau)\left\|\begin{bmatrix}\lambda_v(t_0)\\ -\lambda_r(t_0)\end{bmatrix}\right\| \tag{21}$$

where the function $g(t)$ represents a "curve fit" of sorts, then Eq.(20) becomes

$$\begin{bmatrix}\delta r(t)\\ \delta v(t)\end{bmatrix} \approx P(t)[P(t_0)]^{-1}\begin{bmatrix}\delta r(t_0)\\ \delta v(t_0)\end{bmatrix} + TP(t)\left[\int_{t_0}^{t}\left[\frac{[P(\tau)]^{-1}\begin{bmatrix}0\\ [I\ \ 0]P(\tau)\end{bmatrix}}{m(\tau)g(\tau)}\right]d\tau\right][P(t_0)]^{-1}\frac{\begin{bmatrix}\lambda_v(t_0)\\ -\lambda_r(t_0)\end{bmatrix}}{\left\|\begin{bmatrix}\lambda_v(t_0)\\ -\lambda_r(t_0)\end{bmatrix}\right\|} \tag{22}$$

Now, the integral only has to be evaluated once for each choice of burn, saving a considerable about of computation time. Finally, note that burns are not restricted in length, using osculating orbits (much as in Encke's method) the burn lengths are actually rather arbitrary. The only consideration for burn length, then, is the error accumulated by approximated functions during integration.

To formulate a method for computing the transfer, the above discussion hints to a burn-by-burn approach. Burns would be guessed by a user via a set of transfer orbits and burn times. Each burn would then be approached as a single-burn rendezvous problem. This produces a sub-optimal transfer and can be thought of as a predictor step. The corrector step would then consist of iterations to make it an optimal transfer; either a direct optimization of the transfer orbit elements and burn times or an indirect optimization by multiple-shooting.

IV. CONCLUSIONS

The development of the Direction Correction method is proceeding rather well. At the time of this report we are not prepared to say whether the method will be successful. The ideas that it is based upon have been validated individually. It has also produced a fair approximation to the solution of a known problem. Further testing of the method is required in order to determine just how robust it is; but at this point it seems pretty clear that method will work.

The Patched Transfer Method is very promising. Glandorf's formulation for the Lagrange multipliers been checked numerically and a suitable approximation for the

Lagrange multiplier magnitude is forthcoming. The next steps are to refine the predictor-corrector idea, code the method, and test it.

V. REFERENCES

[1]Lawden, D.F., Optimal Trajectories for Space Navigation, Butterworths, London, 1963.

[2]Oberle, H. J., BOUNDSCO - Hinweise zur Benutzung des Mehrzielverfahrens für die numerische Lösung von Randwerproblemen mit Schaltbedingungen, Hamburger Beiträge zur Angewandten Mathematik, Berichte 6, 1987.

[3]Chuang, C.-H., and Speyer, J.L. "Periodic Optimal Hypersonic SCRAMjet Cruise," *Optimal Control Applications & Methods*, Vol. 8, 1987, pp. 231-242.

[4]Brusch, R.G. and Vincent, T.L., "Low-Thrust, Minimum-Fuel, Orbital Transfers," *Astronautica Acta*, Vol. 16, pp 65-74.

[5]Redding, D.C., "Optimal Low-Thrust Transfers to Geosynchronous Orbit," NASA Lewis SUDAAR 539, Cleveland, Ohio 44135, Sept. 1983.

[6]Chuang, C.-H., Goodson, T.D., and Hanson, J. "Optimal Trajectories of Low- And Medium-Thrust Orbit Transfers with Drag and Oblateness," submitted to *Journal of the Astronautical Sciences*, 1994

[7]Zondervan, K.P., Lincoln, L.J., and Caughey, T.K., "Optimal Low-Thrust, Three-Burn Orbit Transfers with Large Plane Changes," *Journal of the Astronautical Sciences*, Vol. 32, No. 3, 1984, pp. 407-427

[8]Marec, J.P., Optimal Space Trajectories, Elsevier Scientific Publishing, New York, 1979.

[9]Glandorf, D.R., "Lagrange Multipliers and the State Transition Matrix for Coasting Arcs," *AIAA Journal*, Vol. 7, No. 2, Feb. 1969, pp 363-365.

[10]Eckenwiler, M. W., "Closed-Form Lagrangian Multipliers for Coast Periods of Optimum Trajectories," *AIAA Journal*, Vol. 3, No. 6, June 1965, pp 1149-1151.

[11]Pines, S., ""Constants of the Motion for Optimal Thrust Trajectories in a Central Force Field" *AIAA Journal*, Vol. 2, No. 11, Nov. 1964, pp 2010-2014.

SEMI-ANNUAL STATUS REPORT

Submitted to:	NASA Marshall Space Flight Center
Grant Title:	THEORY AND COMPUTATION OF OPTIMAL LOW- AND MEDIUM-THRUST TRANSFERS
Grant Number:	NAG8-921
Principal Investigator /Project Director:	Dr. C.-H. Chuang School of Aerospace Engineering Georgia Institute of Technology Atlanta, GA 30332-0150 Phone: (404) 894-3075 Fax: (404) 894-2760 E-mail: ch.chuang@aerospace.gatech.edu
Research Assistant:	Troy D. Goodson Laura A. Ledsinger School of Aerospace Engineering Georgia Institute of Technology
Period Covered:	July 7, 1994 to January 6, 1995
Date of Submission:	January 5, 1995

TABLE OF CONTENTS

	ABSTRACT	1
I.	INTRODUCTION	1
II.	FIRST VARIATION CONSIDERATIONS	3
III.	CHECKING THE SECOND VARIATION	11
IV.	NEIGHBORING OPTIMAL FEEDBACK GUIDANCE	25
V.	SIMULATION RESULTS FOR THE FREE FINAL TIME SOLUTION USING THE ONE BURN CASE	27
VI.	PATCHED TRANSFER METHOD PROGRESS	29
VII.	CONCLUSIONS	33
VIII.	REFERENCES	34

Multiple Burn Fuel-Optimal Orbit Transfers: Numerical Trajectory Computation and Neighboring Optimal Feedback Guidance

C.-H. Chuang, Troy D. Goodson, and Laura A. Ledsinger
School of Aerospace Engineering
Georgia Institute of Technology, Atlanta, Georgia 30332

ABSTRACT

This report describes current work in the numerical computation of multiple burn, fuel-optimal orbit transfers and presents an analysis of the second variation for extremal multiple burn orbital transfers as well as a discussion of a guidance scheme which may be implemented for such transfers. The discussion of numerical computation focuses on the use of multivariate interpolation to aid the computation in the numerical optimization. The second variation analysis includes the development of the conditions for the examination of both fixed and free final time transfers. Evaluations for fixed final time are presented for extremal one, two, and three burn solutions of the first variation. The free final time problem is considered for an extremal two burn solution. In addition, corresponding changes of the second variation formulation over thrust arcs and coast arcs are included. The guidance scheme discussed is an implicit scheme which implements a neighboring optimal feedback guidance strategy to calculate both thrust direction and thrust on-off times.

I. INTRODUCTION

The necessary conditions which result from analyzing the first variation of a cost functional are widely used. These are commonly referred to as the Euler-Lagrange equations. Many problems require additional considerations, for example, the problem considered herein, fuel-optimal orbit transfer, requires consideration of Pontryagin's Maximum Principle.

Many researchers have used the first variation to compute extremal solutions to the fuel-optimal orbit transfer problem. Some have used them to apply two-point boundary value problem solvers to optimization problems, forming indirect methods.[1,2,3] Others have used a partial set of the conditions to form hybrid indirect/direct methods where certain highly sensitive parameters are optimized directly.[4,5] However, to the knowledge of the authors, few, if any, have made use of the conditions related to the second variation of the cost functional. These provide sufficient conditions which, when met, declare an extremal solution as a local, weak optimal solution.

Once the second variation of the cost functional is verified so that it is known whether the sufficient conditions are met, the information obtained can then be used to implement a guidance scheme. Guidance is defined to be the determination of a way to follow an optimal trajectory when presented with obstacles such as environmental disturbances or uncertainties in navigation data. Two different types of guidance schemes exist: implicit and explicit. Implicit guidance systems are characterized by the fact that the vehicle's motion must be precomputed on the ground and then compared to the actual motion. The equations which need to be solved are based upon the difference between these measured and precomputed values. The solutions to these equations are used in the vehicle's steering and velocity control. Explicit guidance systems are generalized by the fact that the vehicle's equations of motion are modeled and solved for by on-board computers during its motion. The solutions for the equations are solved continuously and are used to determine the difference between the vehicle's current motion and its destination. Commands are then generated to alleviate the anticipated error.

Existing guidance schemes have been presented in various papers. An iterative guidance scheme which is implemented using a linear tangent steering law is presented by Smith.[6] This guidance scheme has been used for the Saturn V and is in currently used by the Space Shuttle, the Atlas-Centaur, and the Titan-Centaur. In a paper by Lu[7], a general nonlinear guidance law is developed using two different strategies. One strategy

uses optimal control theory to generate a new optimal trajectory onboard from the start, while the other uses flight-path-restoring-guidance to bring the trajectory back to the nominal. A guidance scheme that is developed using inverse methods for unthrusted, lift-modulated vehicles along an optimal space curve is presented by Hough.[8] Linearized guidance laws applicable to many different types of space missions are presented by Tempelman.[9] These guidance laws are based on fixed and free final time arrivals. Naidu[10] presents a guidance scheme applicable to aeroassisted orbital transfers. This scheme is developed by implementing neighboring optimal guidance and linear quadratic regulator theory. Some interesting techniques for making the neighboring optimal guidance converge about the nominal path are introduced in a paper by Powers.[11]

The guidance scheme proposed in this report is an implicit one which implements neighboring optimal feedback guidance. An implicit guidance system was chosen due to the fact that that type of guidance system handles disturbances well.[10] The neighboring optimal feedback guidance was chosen because it inherently uses the nominal solutions. Also, it has the advantage of being a feedback system, as a opposed to open-loop guidance.

In this scheme, the initial orbit exit point is assumed to be perturbed from the nominal point but the boundary condition, specifying the final orbit, is assumed unchanged. The goal is to use the controller to bring the trajectory back to the nominal path at some point by using minimal fuel.

II. First Variation Considerations

Within this report results are restricted to the planar case, no plane changes are considered at this stage of development. The solutions examined in this report satisfy the conditions related to the first variation. In the next section, the conditions sufficient for declaring a minimizing solution will be checked for these transfers. Some of these transfers are multi-burn transfers and in order to simplify initial analysis new nominal

solutions have been constructed from these which are single-burn transfers, i.e. the thrust is kept on for all time between the initial orbit exit point and the final orbit entry point.

Only that part of the original trajectory which is contained in the last burn is taken to constitute the new extremal solution. This new extremal solution has a fixed initial point and fixed transfer time but the final point is only constrained in that it must lie on the final orbit.

II.1. CONDITIONS FROM THE FIRST VARIATION

The first order conditions for this problem have been stated many times[12] and will be given here only briefly. The optimization problem consists of a cost functional (Eq. [1]), state dynamics (Eqs. [2-4]), fixed initial point conditions (Eq. [5]), and boundary conditions on the terminal point (Eq. [6]); each of these is expressed below.

$$J = -m(t_f) \tag{1}$$

$$\dot{\mathbf{r}} = \mathbf{v} \tag{2}$$

$$\dot{\mathbf{v}} = \frac{T}{m}\mathbf{e}_T - \frac{\mu}{r^3}\mathbf{r} \tag{3}$$

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{4}$$

$$\mathbf{r}(t_o) = \mathbf{r}_o, \quad \mathbf{v}(t_o) = \mathbf{v}_o, \quad m(t_o) = m_o \tag{5}$$

$$\begin{bmatrix} \psi_1 \\ \psi_2 \\ \psi_3 \end{bmatrix} = \begin{bmatrix} \text{xv - yu} \\ (v^2 - \mu/r)\text{x} - (\mathbf{r}^T\mathbf{v})\text{u} \\ (v^2 - \mu/r)\text{y} - (\mathbf{r}^T\mathbf{v})\text{v} \end{bmatrix} - \begin{bmatrix} h \\ \mu e_x \\ \mu e_y \end{bmatrix} \tag{6}$$

where $\mathbf{r}=[\text{x y}]^T$ is the radius vector, $\mathbf{v}=[\text{u v}]^T$ is its time derivative, $\mathbf{e}_T$ is the thrust direction, a unit vector, T is thrust magnitude (limited between zero and some maximum

value T_{max}), μ is the gravitational constant, g_0 is the gravitational acceleration at sea-level, I_{sp} is the specific impulse of the motor. The quantity $g_0 I_{sp}$ is often referred to as the exit velocity of the motor. The Euler Lagrange conditions are then

$$\mathbf{e}_T = \frac{-\lambda_v}{|\lambda_v|} \tag{7}$$

$$\dot{\lambda}_r = \mu\left[\frac{\lambda_v}{r^3} - 3\frac{(\lambda_v{}^T\mathbf{r})\mathbf{r}}{r^5}\right] \tag{8}$$

$$\dot{\lambda}_v = -\lambda_r \tag{9}$$

$$\dot{\lambda}_m = \frac{T}{m^2}\lambda_v{}^T\mathbf{e}_T = \frac{T}{m^2}|\lambda_v| \tag{10}$$

where $\lambda_r=[\lambda_x\ \lambda_y]^T$ and $\lambda_v=[\lambda_u\ \lambda_v]^T$. The natural boundary conditions are

$$\lambda_x(t_f) = \nu_3 v + \nu_4(v^2 - \mu/r + \mu x^2/r^3) + \nu_5(\mu xy/r^3 - uv)\big|_{t=t_f} \tag{11}$$

$$\lambda_y(t_f) = -\nu_3 u + \nu_4(\mu xy/r^3 - uv) + \nu_5(u^2 - \mu/r + \mu y^2/r^3)\big|_{t=t_f} \tag{12}$$

$$\lambda_u(t_f) = -\nu_3 y - \nu_4 yv + \nu_5(2yu - xv)\big|_{t=t_f} \tag{13}$$

$$\lambda_v(t_f) = \nu_3 x + \nu_4(2xv - uy) - \nu_5 xu\big|_{t=t_f} \tag{14}$$

$$\lambda_m(t_f) = -1 \tag{15}$$

The conditions resulting from applying Pontryagin's Minimum Principle are

$$\begin{aligned} H_S < 0, \quad T = T_{max} \\ H_S > 0, \quad T = 0 \end{aligned} \tag{16}$$

where

$$H_S = -\left(\frac{|\lambda_v|}{m} + \frac{\lambda_m}{g_o I_{sp}} \right) \tag{17}$$

Note that when the derivatives of H_S are zero, singular arc solutions may exist. This has been checked numerically.[12]

Finally, the free final time problem will also be considered here. For these extremal solutions the final, or transfer, time is selected such that the transversality condition is satisfied, i.e. the Hamiltonian is zero at $t=t_f$.

$$H(t_f) = \lambda_r^T \dot{r} + \lambda_v^T \dot{v}\Big|_{t=t_f} = 0 \tag{18}$$

In a previous paper[12] this problem was given as a maximization problem. To conform to the convention used for the second variation[13], it is now stated as a minimization problem. If an extremal solution to the maximization problem is given as state time history $\mathbf{x}(t)$, Lagrange-multiplier time history $\lambda(t)$, and Lagrange multipliers ν, associated with boundary conditions, then the extremal solution of the above minimization problem is $\mathbf{x}(t)$, $(-1)*\lambda(t)$, and $(-1)*\nu$.

Additionally, it makes more sense in the guidance problem to consider the control as the angle θ, rather than individual components of a unit vector. This simplifies analysis because the control is now a scalar. Equation [7] must now be restated as

$$tan(\theta) = -\frac{\lambda_v}{\lambda_u} \tag{19}$$

II.2. EXTREMAL SOLUTIONS

All quantities associated with the solutions presented here have been nondimensionalized so that μ=1 and will be presented here in that form. The relations used to nondimensionalize are given below.

$$\hat{\mathbf{r}} \equiv \frac{\mathbf{r}}{r^{\star}} \qquad \hat{\mathbf{v}} \equiv \frac{\mathbf{v}}{\sqrt{\mu/r^{\star}}} \tag{20-21}$$

$$\hat{m} \equiv \frac{m}{m^{\star}} \qquad \hat{t}_f \equiv \frac{t_f}{\sqrt{(r^{\star})^3/\mu}} \tag{22-23}$$

$$\hat{T} \equiv \frac{T/m^{\star}}{\mu/(r^{\star})^2} \qquad \left(\hat{g}_o \hat{I}_{sp}\right) = g_o I_{sp}\sqrt{r^{\star}/\mu} \tag{24-25}$$

where $r^{\star}$ and $m^{\star}$ are indicated in the tables for each case of the extremal solutions.

Each of the transfers given below have both the initial orbit exit point and final orbit entry points free. However, for the guidance problem it makes more physical sense to consider the initial orbit exit point as fixed and equal to the optimal choice, for it cannot be updated once the transfer has begun.

The last burn of any multi-burn transfer below may also be taken as a complete transfer unto itself. The initial point can be chosen as the one at the very first instant (or shortly thereafter) of thrusting for the last burn. The final orbit exit point must remain unchanged. Obviously, for these choices the natural boundary conditions for final orbit entry point are still satisfied. This new transfer can then be considered as a new extremal solution, though to an orbit transfer problem with a different initial orbit.

Figure 1 shows a one burn ascent extremal solution. This trajectory is a transfer leaving an orbit with a semimajor axis a=1.069, eccentricity e=0.02633, and argument of perigee ω= -50°. The transfer ends at a nearby orbit with a=1.038 and e=0. Other pertinent transfer data are given in Table I. This transfer was produced by shortening the time of a two-burn transfer until the coast arc between them vanished. This transfer is

therefore both a minimum mass and minimum time extremal solution because mass and time have an affine relationship in the one-burn case.

Figure 2 displays a two-burn transfer, in fact a descending transfer, from an orbit with elements a=3.847, e=0.02378 to a final orbit with elements a=1.5, e=0.3333. The apses of the terminal orbits are aligned and lie on the 'X' axis of the figure. The initial mass is 1.608 and the final mass is 1.1547. Other pertinent transfer data are given in Table II. This transfer has the transversality condition converged, therefore it is a candidate fuel-optimal free final time solution. By the same right, it can also be considered a candidate optimal solution for the fixed final time problem.

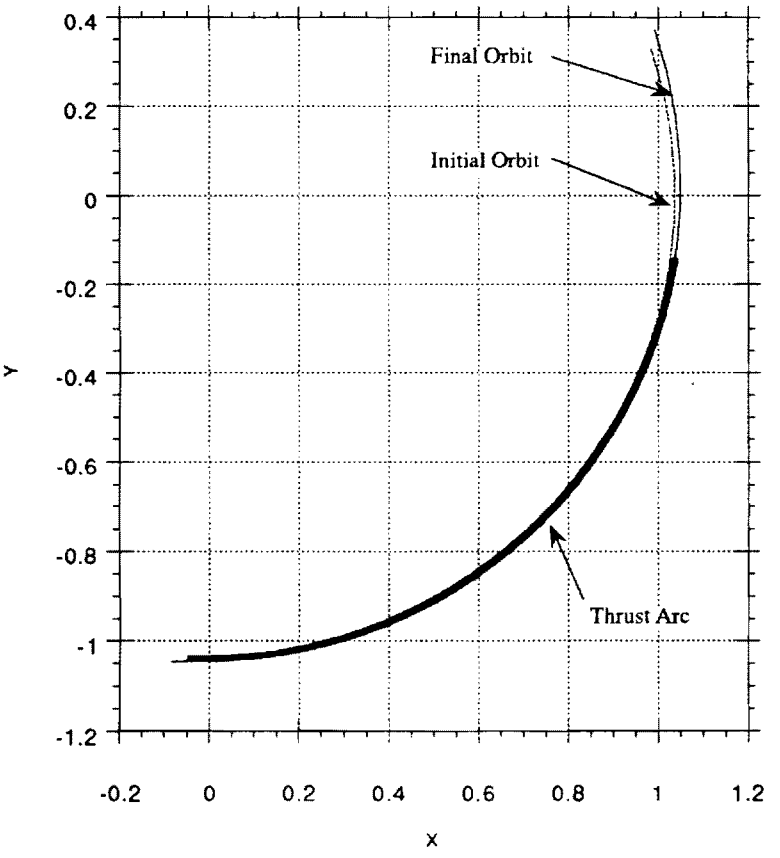


Figure 1: One-Burn Extremal with Fixed Final Time.

$g_o I_{sp}$ =	0.3861	a_i=	1.038	ω_i=	n/a	a_f=	1.069	ω_f=	-50°
T=	0.03	e_i=	0.000	t_f=	1.553	e_f=	0.02633	m_f=	1.542
$r^{\star}$=	6378 *km*	$m^{\star}$=	14 *Mg*						

Table I. Parameters of the Transfer Shown in Figure 1.

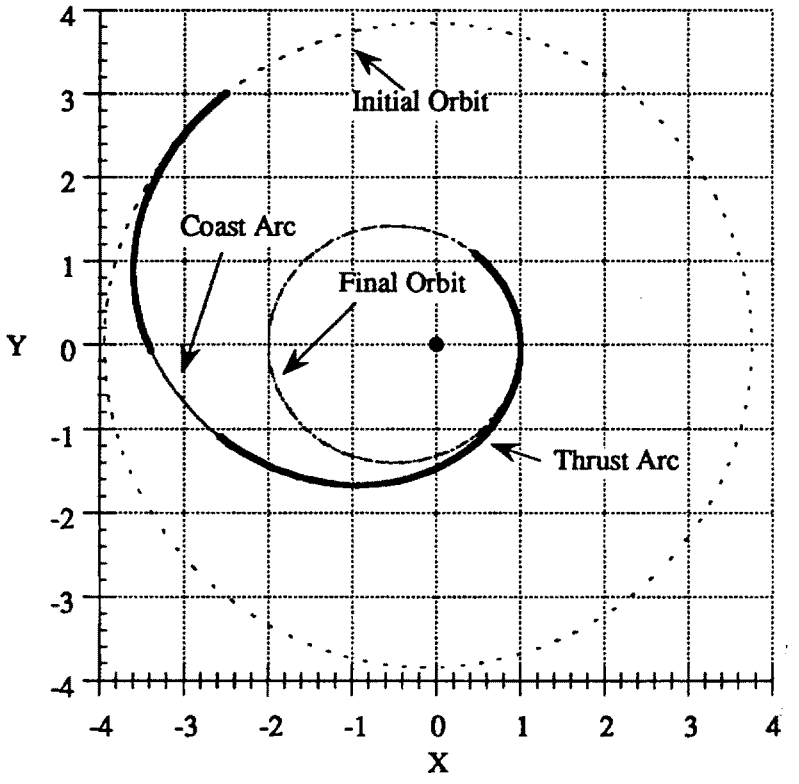


Figure 2: Two-Burn Extremal Orbit Transfer Solution with Free Final Time.

$g_o I_{sp}$ =	1.313	a_i=	3.847	ω_i=	0°	a_f=	1.5	ω_f=	0°
T=	0.03	e_i=	0.02378	t_f=	19.05	e_f=	0.3333	m_f=	1.608
$r^{\star}$=	6878 *km*	$m^{\star}$=	200 *kg*						

Table II Parameters of Transfer in Figure 2.

A three-burn transfer is shown in Figure 2. Since this transfer is between orbits of increasing semimajor axis, it will be referred to as an ascending transfer. The initial orbit

has elements a=2.239, e=0.1160, and ω=-85.94°. The elements of the final orbit are a=7, e=0.7332, and w=114.6°. Other pertinent transfer data are given in Table III.

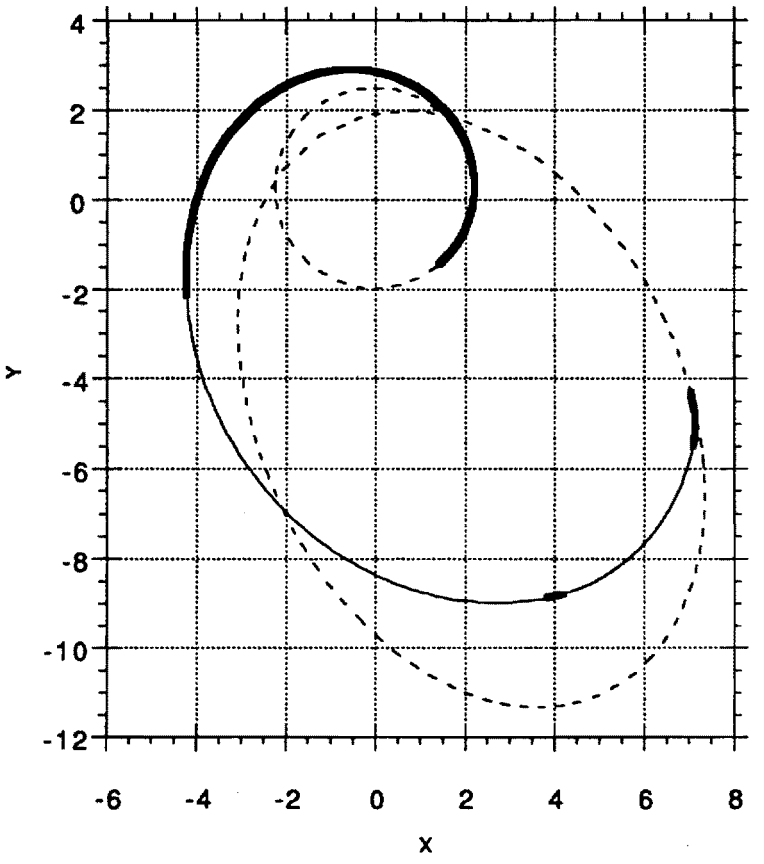


Figure 3: Three-Burn Extremal Orbit Transfer with Fixed Final Time .

$g_o I_{sp}$ =	0.3898	a_i=	2.239	ω_i=	-85.94°	a_f=	7.000	ω_f=	114.6°
T=	0.01386	e_i=	0.1160	t_f=	85.00	e_f=	0.7332	m_f=	0.6056
$r^{\star}$=	6378 *km*	$m^{\star}$=	14 *Mg*						

Table III. Parameters of the transfer shown in Figure 3.

III. CHECKING THE SECOND VARIATION

Extensive derivation of the conditions for the second variation of the cost functional has already been detailed in Ref [13]. Equations given in this section are the results of applying this work to our problem.

Considering the second variation of the augmented cost functional, J, a new optimal control problem can be stated. In this new problem, the state is δx, the control δu, and the Lagrange-multipliers are $\delta\lambda$ and dv.. Thus the new cost function is

$$\delta^2 \bar{J} = \frac{1}{2}\left[\delta x^T \left(\phi_{xx} + \left(v^T \psi_x\right)_x\right)\delta x\right]_{t=t_f} + \frac{1}{2}\int_{t_o}^{t_f}\begin{bmatrix}\delta x^T & \delta u^T\end{bmatrix}\begin{bmatrix}H_{xx} & H_{xu} \\ H_{ux} & H_{uu}\end{bmatrix}\begin{bmatrix}\delta x \\ \delta u\end{bmatrix} dt \tag{26}$$

subject to

$$\delta\dot{x} = f_x \delta x + f_u \delta u \tag{27}$$

$$\delta x(t_o) = \delta x_o \tag{28}$$

where $\mathbf{x}=[\mathbf{r}^T\ \mathbf{v}^T\ m]^T$ and $u=\theta$.

In general, neighboring optimal feedback guidance allows the designer to consider changes in final boundary conditions. We consider no such changes, assuming that the destination orbit was accurately planned well in advance. Formulation will be made below for both the fixed and free final time cases.

Evaluating the terms in Eq. [26-28], for orbital transfer, the partial derivatives for the dynamics and for the Hamiltonian are:

$$\mathbf{f}_x = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ -\left(\frac{\mu}{r^3}\right) - \frac{3\mu x^2}{r^5} & \frac{3\mu xy}{r^5} & 0 & 0 & -\frac{T}{m}\cos(\theta) \\ \frac{3\mu xy}{r^5} & -\left(\frac{\mu}{r^3}\right) - \frac{3\mu y^2}{r^5} & 0 & 0 & -\frac{T}{m}\sin(\theta) \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{29}$$

$$\mathbf{f}_\theta = \begin{bmatrix} 0 \\ 0 \\ -\frac{T}{m}sin(\theta) \\ \frac{T}{m}cos(\theta) \\ 0 \end{bmatrix} \tag{30}$$

$$\mathbf{H}_{xx} = \begin{bmatrix} -\mu\left(\frac{-3(3\lambda_u x + \lambda_v y)}{r^5} + \frac{15(\lambda_v^T r)x^2}{r^7}\right) & -\mu\left(\frac{-3(\lambda_u y + \lambda_v x)}{r^5} + \frac{15(\lambda_v^T r)xy}{r^7}\right) & 0 & 0 & 0 \\ -\mu\left(\frac{-3(\lambda_u y + \lambda_v x)}{r^5} + \frac{15(\lambda_v^T r)xy}{r^7}\right) & -\mu\left(\frac{-3(3\lambda_v y + \lambda_u x)}{r^5} + \frac{15(\lambda_v^T r)y^2}{r^7}\right) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2\frac{T}{m^3}|\lambda_v| \end{bmatrix} \tag{31}$$

$$H_{\theta\theta} = \frac{T}{m}|\lambda_v| \tag{32}$$

$$H_{\theta x} = 0 \tag{33}$$

The fixed and free final time problems have the following equations in common:

$$\delta\dot{\mathbf{x}} = \mathbf{A}(t)\delta\mathbf{x} - \mathbf{B}(t)\delta\lambda \tag{34}$$

$$\delta\dot{\lambda} = -\mathbf{C}(t)\delta\mathbf{x} - \mathbf{A}^T(t)\delta\lambda \tag{35}$$

where

$$\mathbf{A}(t) = \mathbf{f}_x - \mathbf{f}_u\mathbf{H}_{uu}^{-1}\mathbf{H}_{ux} \tag{36}$$

$$\mathbf{B}(t) = \mathbf{f}_u\mathbf{H}_{uu}^{-1}\mathbf{f}_u^T \tag{37}$$

$$\mathbf{C}(t) = \mathbf{H}_{xx} - \mathbf{H}_{xu}\mathbf{H}_{uu}^{-1}\mathbf{H}_{ux} \tag{38}$$

For a multiple-burn solution, one finds that $H_{\theta\theta}$ becomes zero during coast arcs. This makes it impossible to solve for the change in control, $\delta\theta$. However, since the thrust is off during a coast arc, it physically makes no difference what choice is made for the control. Therefore, $\delta\theta$ may be chosen as zero and simpler expressions for **A**(t), **B**(t), and **C**(t) can be written as

$$\mathbf{A}(t) = \mathbf{f}_x \tag{39}$$

$$\mathbf{B}(t) = 0 \tag{40}$$

$$\mathbf{C}(t) = \mathbf{H}_{xx} \tag{41}$$

Using the sweepback method for nonlinear terminal constraints, as is the case for this development, the form for $\delta\lambda$ and $\delta\psi$ are assumed as

$$\delta\lambda(t) = \mathbf{P}(t)\delta\mathbf{x}(t) + \mathbf{S}(t)d\nu \tag{42}$$

$$\delta\psi = \mathbf{S}^{\mathrm{T}}(t)\delta\mathbf{x}(t) + \mathbf{V}(t)d\nu \tag{43}$$

which allows the solution for $d\nu$ to be written as

$$d\nu = \mathbf{V}^{-1}(t_o)\left[\delta\psi - \mathbf{S}^{T}(t_o)\delta\mathbf{x}(t_o)\right] \tag{44}$$

As mentioned above, $\delta\psi$=0 will be considered here. The boundary condition equations are given by:

$$\mathbf{P}(t_f) = \left[\phi_{xx} + \left(\nu^{\mathrm{T}}\psi_x\right)_x\right]_{t=t_f} \tag{48}$$

$$S(t_f) = \left[\psi_x^T\right]_{t=t_f} \tag{49}$$

$$V(t_f) = 0 \tag{50}$$

where in the development for the orbital transfer these are:

$$P(t_f) = \begin{bmatrix} a & b & d & e & 0 \\ b & c & f & g & 0 \\ d & f & h & i & 0 \\ e & g & i & j & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{51}$$

where

$$a = \nu_2\mu\left[\frac{x(t_f)}{R^3} - \frac{3x^3(t_f)}{R^5} + \frac{2x(t_f)}{R^3}\right] + \nu_3\mu\left[\frac{y(t_f)}{R^3} - \frac{3x^2(t_f)y(t_f)}{R^5}\right] \tag{52a}$$

$$b = \nu_2\mu\left[\frac{y(t_f)}{R^3} - \frac{3x^2(t_f)y(t_f)}{R^5}\right] + \nu_3\mu\left[\frac{x(t_f)}{R^3} - \frac{3x(t_f)y^2(t_f)}{R^5}\right] \tag{52b}$$

$$c = \nu_3\mu\left[\frac{y(t_f)}{R^3} - \frac{3y^3(t_f)}{R^5} + \frac{2y(t_f)}{R^3}\right] + \nu_2\mu\left[\frac{x(t_f)}{R^3} - \frac{3x(t_f)y^2(t_f)}{R^5}\right] \tag{52c}$$

$$d = -\nu_3 v(t_f) \tag{52d}$$

$$e = \nu_1 - \nu_2 u(t_f) + 2\nu_3 v(t_f) \tag{52e}$$

$$f = -\nu_1 - \nu_2 v(t_f) + 2\nu_3 u(t_f) \tag{52f}$$

$$g = -\nu_2 u(t_f) \tag{52g}$$

$$h = 2\nu_3 y(t_f) \tag{52h}$$

$$i = -\nu_3 x(t_f) - \nu_2 y(t_f) \tag{52i}$$

$$j = 2\nu_2 x(t_f) \tag{52j}$$

$$\mathbf{S}(t_f)=\begin{bmatrix} v(t_f) & v^2(t_f)-\frac{\mu}{R}+\frac{\mu x^2(t_f)}{R^3} & \frac{x(t_f)y(t_f)}{R^3}-u(t_f)v(t_f) \\ -u(t_f) & \frac{x(t_f)y(t_f)}{R^3}-u(t_f)v(t_f) & u^2(t_f)-\frac{\mu}{R}+\frac{\mu y^2(t_f)}{R^3} \\ -y(t_f) & -y(t_f)v(t_f) & 2y(t_f)u(t_f)-x(t_f)v(t_f) \\ x(t_f) & 2x(t_f)v(t_f)-u(t_f)y(t_f) & -x(t_f)u(t_f) \\ 0 & 0 & 0 \end{bmatrix} \tag{53}$$

Following from the assumptions expressed as Eqs. [46-47], the following nonlinear equations for **P**, **S**, and **V** must be integrated backwards. The results will be used to check the sufficient conditions governing a minimizing solution.

$$\dot{\mathbf{P}}=-\mathbf{PA}-\mathbf{A}^T\mathbf{P}+\mathbf{PBP}-\mathbf{C} \tag{54}$$

$$\dot{\mathbf{S}}=-\left(\mathbf{A}^T-\mathbf{PB}\right)\mathbf{S} \tag{55}$$

$$\dot{\mathbf{V}}=\mathbf{S}^T\mathbf{BS} \tag{56}$$

To satisfy the sufficient conditions, $H_{\theta\theta}$, **P**, **S**, and **V** must be such that

convexity condition: $H_{\theta\theta}(t) > 0$ for $t_o \le t \le t_f$ (57)

normality condition: $\mathbf{V}^{-1}(t)$ exists for $t_o \le t < t_f$ (58)

conjugate point condition: $\mathbf{P}(t)-\mathbf{S}(t)\mathbf{V}^{-1}(t)\mathbf{S}^T(t)$ finite for $t_o \le t < t_f$ (59)

The convexity condition is satisfied for any transfer satisfying the choice of control specified by the Euler-Lagrange equations. This can be seen by noting that Eq.[32] is positive definite, irrespective of the time history for the Lagrange multipliers.

III.1. Numerical Results for Fixed Final Time

The results discussed in this section were obtained for nominal solutions with fixed transfer time. The eigenvalues of the V(t) matrix are plotted in Figure 4 for the 3-burn extremal solution. Note that V(t) is not negative definite, one of the eigenvalues is zero for all time and the other two eigenvalues are positive. Therefore, the normality condition is violated. Furthermore, the conjugate point condition cannot be checked. It must be concluded that the 3-burn extremal does not satisfy the sufficient conditions and cannot be considered an optimal solution for fixed final time. Figure 5 shows the eigenvalues of V(t) over the one-burn extremal constructed from this solution in the manner described earlier. The same conclusions must be made for this transfer.

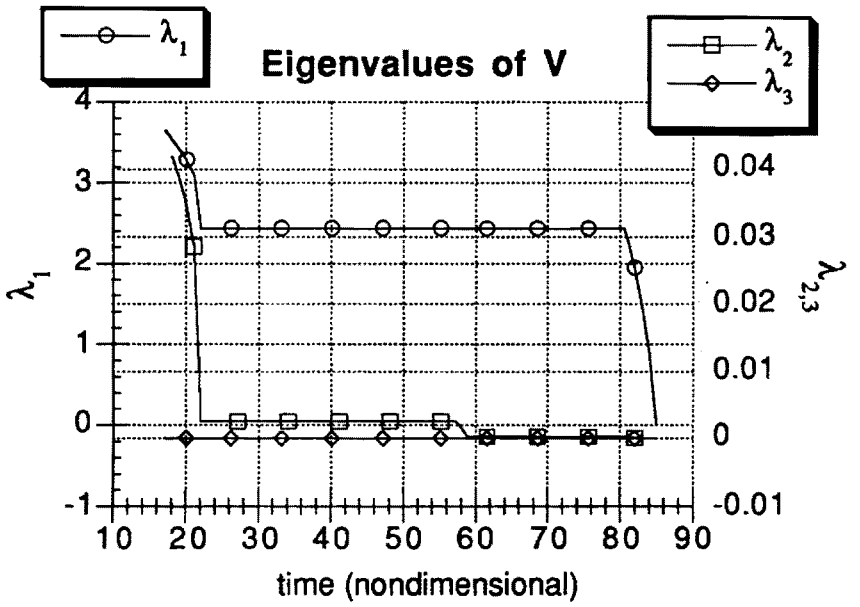


Figure 4: Plot of Eigenvalues of V(t) for Three Burn Extremal.

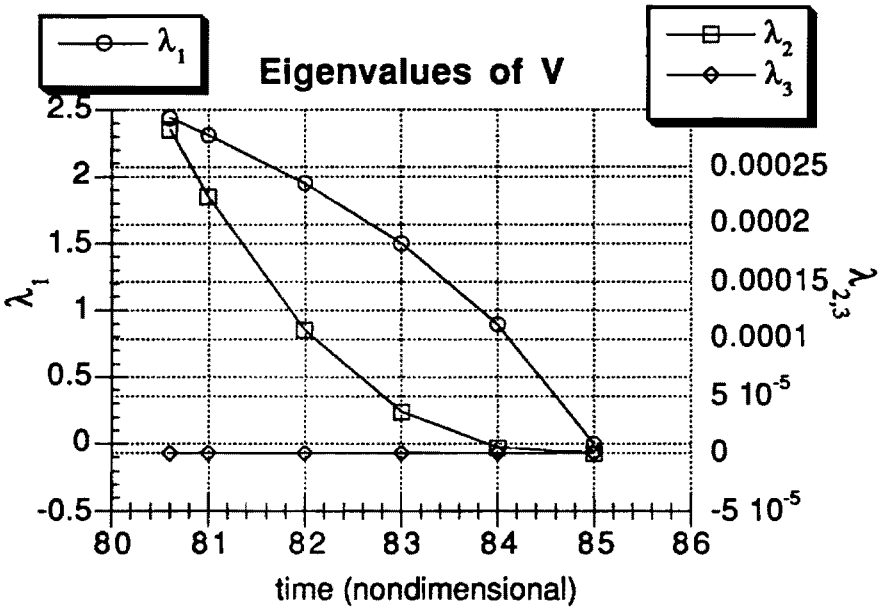


Figure 5: Plot of Eigenvalues of V(t) for Last Burn of Three-Burn Extremal.

The eigenvalues of V(t) for the single-burn transfer are shown in Figure 6. This plot is again made for the two-burn extremal in Figure 7 and a one-burn constructed from it in Figure 8. These figures all show similar results, namely that V(t) is not negative definite, but positive semidefinite. The situation has been repeated, namely that the normality condition has been violated and the conjugate point condition cannot be checked. Therefore, none of the extremal solutions with fixed final time given in this report satisfy the sufficient conditions for a minimizing solution.

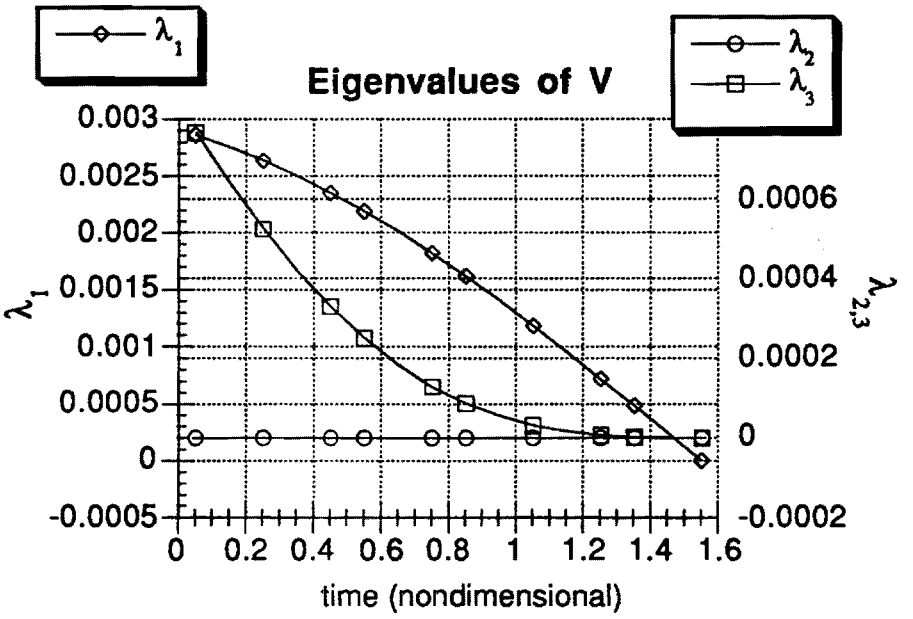


Figure 6: **Plot of Eigenvalues for One-Burn Extremal.**

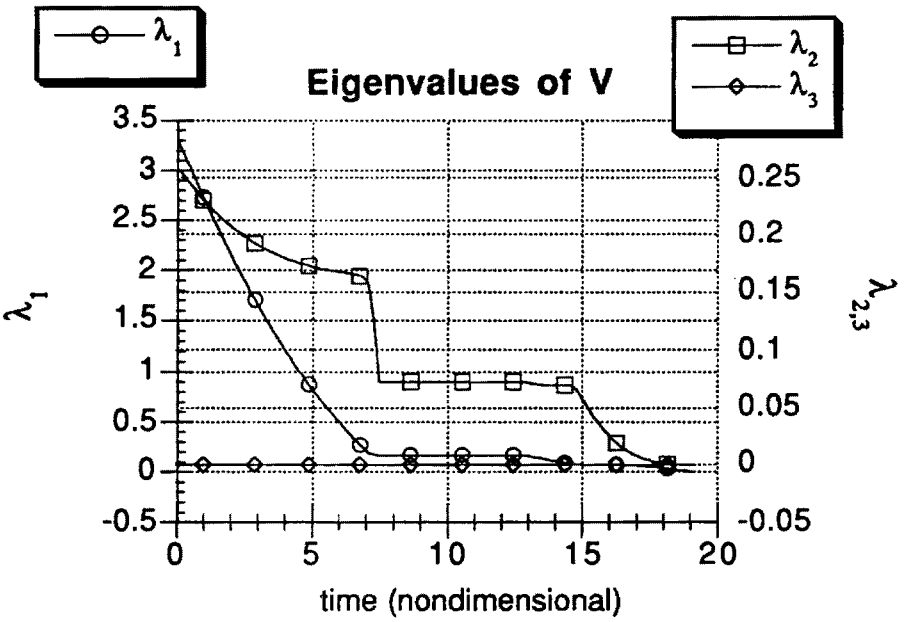


Figure 7: **Plot of Eigenvalues of V(t) for Two-Burn Extremal.**

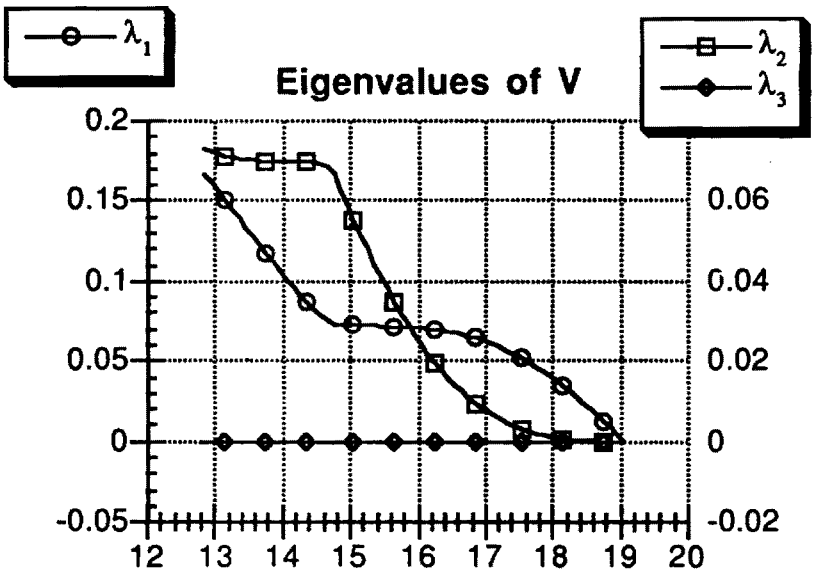


Figure 8: Plot of Eigenvalues of V(t) for Last Burn of Two-Burn Extremal.

III.2. NUMERICAL RESULTS FOR FREE FINAL TIME

When the final time is unspecified, a new condition, the transversality condition, must be satisfied by the nominal solution. This condition is expressed in Eq. [60a]

$$\Omega(\mathbf{x},\mathbf{u},\nu,t)\big|_{t=t_f} \equiv \left(\frac{d\Phi}{dt}+\mathbf{L}\right)_{t=t_f} = 0 \tag{60a}$$

$$\text{where } \Phi = \phi(\mathbf{x},t) + \nu^T\psi(\mathbf{x},t) \tag{60b}$$

$$\frac{d\Phi}{dt} = \frac{\partial\Phi}{\partial t} + \frac{\partial\Phi}{\partial \mathbf{x}}\dot{\mathbf{x}} \tag{60c}$$

This slightly complicates the process of checking the sufficient conditions. The sweepback method can be used with some additions. Three differential equations and thus three boundary conditions must be added to those for **P**, **S**, and **V**.

$$\dot{\mathbf{m}} = -(\mathbf{A}^T - \mathbf{PB})\mathbf{m}, \qquad \mathbf{m}(t_f) = \left(\frac{d\Omega}{d\mathbf{x}}\right)^T_{t=t_f} \tag{61}$$

$$\dot{\mathbf{n}} = \mathbf{S}^{\mathrm{T}}\mathbf{B}\mathbf{m}, \qquad \mathbf{n}(t_f) = \left(\frac{d\psi}{dt}\right)_{t=t_f} \tag{62}$$

$$\dot{\alpha} = \mathbf{m}^{\mathrm{T}}\mathbf{B}\mathbf{m}, \qquad \alpha(t_f) = \left(\frac{d\Omega}{dt}\right)_{t=t_f} \tag{63}$$

These additions are used to form $\overline{\mathbf{P}}$, $\overline{\mathbf{S}}$, and $\overline{\mathbf{V}}$ matrices as follows:

$$\overline{\mathbf{P}} = \mathbf{P} - \frac{\mathbf{m}\mathbf{m}^T}{\alpha} \tag{64}$$

$$\overline{\mathbf{S}} = \mathbf{S} - \frac{\mathbf{m}\mathbf{n}^T}{\alpha} \tag{65}$$

$$\overline{\mathbf{V}} = \mathbf{V} - \frac{\mathbf{n}\mathbf{n}^T}{\alpha} \tag{66}$$

The equations for $d\nu$ and $\delta\lambda$ change by substituting $\overline{\mathbf{P}}$, $\overline{\mathbf{S}}$, and $\overline{\mathbf{V}}$ for **P**, **S**, and **V** respectively, giving

$$d\nu = \overline{\mathbf{V}}^{-1}(t_o)\left[\delta\psi - \overline{\mathbf{S}}^{\mathrm{t}}(t_o)\delta\mathbf{x}(t_o)\right] \tag{67}$$

$$\delta\lambda(t) = \overline{\mathbf{P}}(t)\delta\mathbf{x}(t) + \overline{\mathbf{S}}(t)d\nu \tag{68}$$

Note again, however that $\delta\psi$ has been assumed zero. Now, the sufficient conditions based on the second variation with free final time are:

convexity condition: $H_{\theta\theta}(t) > 0$ for $t_o \le t \le t_f$ (69)

normality condition:
$\overline{\mathbf{V}}^{-1}(t)$ exists for $t_o \le t < t_f$ (70a)
$\alpha^{-1}(t)$ exists for $t_o \le t < t_f$ (71b)

conjugate point condition: $\overline{\mathbf{P}}(t) - \overline{\mathbf{S}}(t)\overline{\mathbf{V}}^{-1}(t)\overline{\mathbf{S}}^{\mathrm{T}}(t)$ finite for $t_o \le t < t_f$ (72)

The eigenvalues of $\overline{\mathbf{V}}$ are plotted in Figure 9. Figures 10-12 plot the elements of the conjugate point condition matrix. Figure 13 is a plot of $\alpha(t)$. Figure 9 shows that $\overline{\mathbf{V}}$ is positive definite in the required interval. Figure 13 shows that $\alpha(t)$ is negative definite in the required interval. Since the normality condition requires that the inverse of $\overline{\mathbf{V}}$ and $\alpha(t)$ exists in the interval, this solution is normal. Figures 10-12 show that the conjugate point condition is satisfied. The elements are bounded in the required interval and grow asymptotically at the final time. Therefore, this solution satisfies the sufficient conditions for minimizing the cost functional with free transfer time.

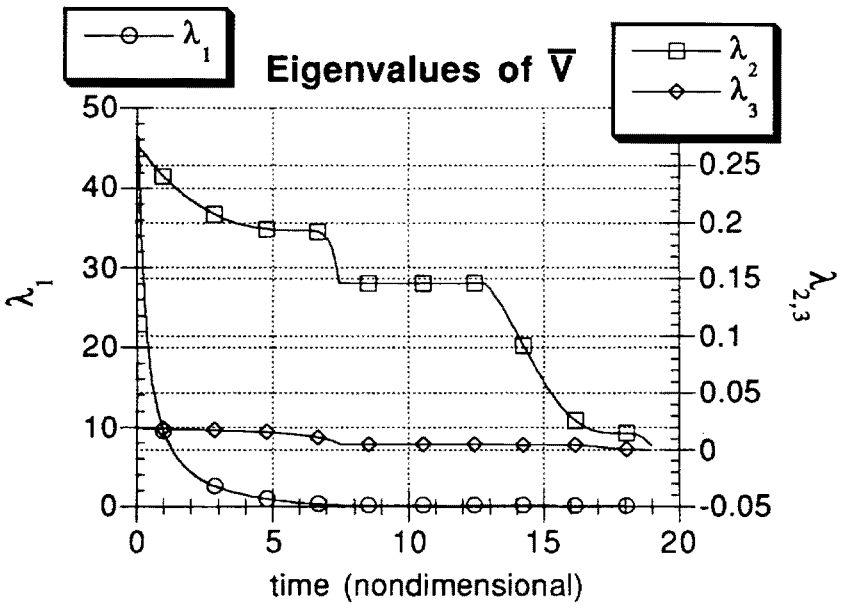


Figure 9 **Plot of Eigenvalues of $\overline{\mathbf{V}}(t)$ for Two Burn Extremal.**

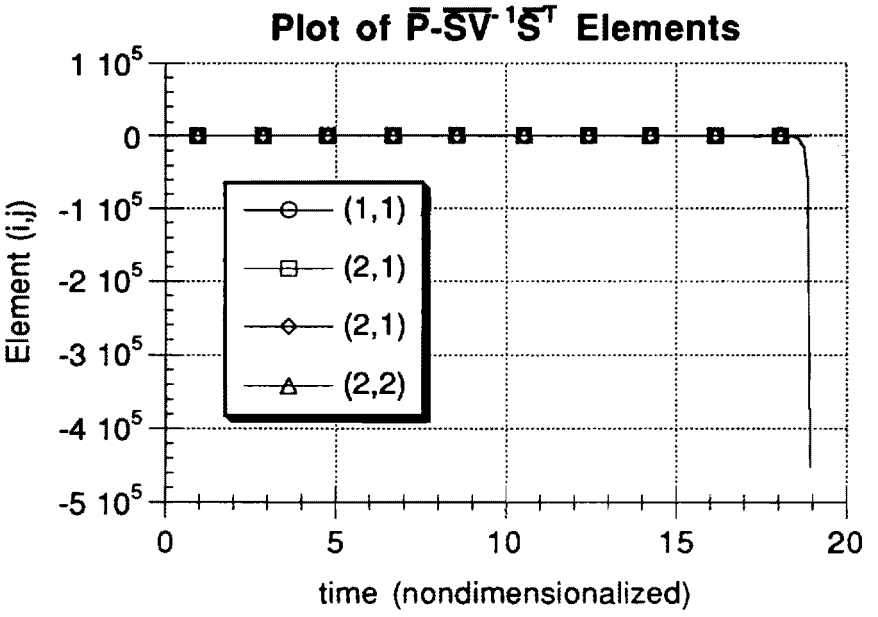


Figure 10: **Plot of Elements of Conjugate Point Condition Matrix for Two Burn Extremal.**

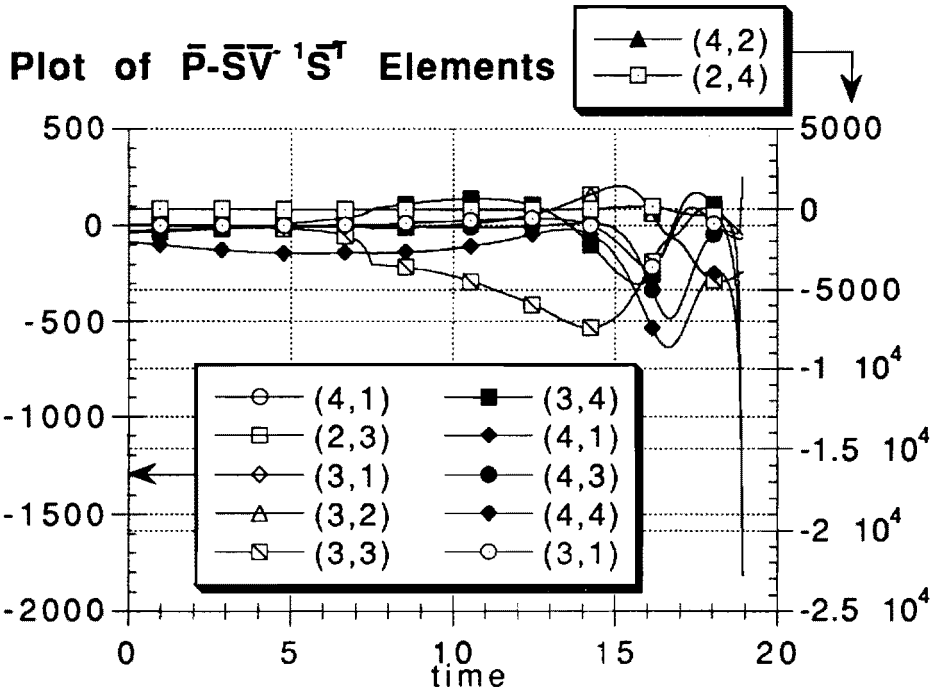


Figure 11: **Plot of Elements of Conjugate Point Condition Matrix for Two Burn Extremal.**

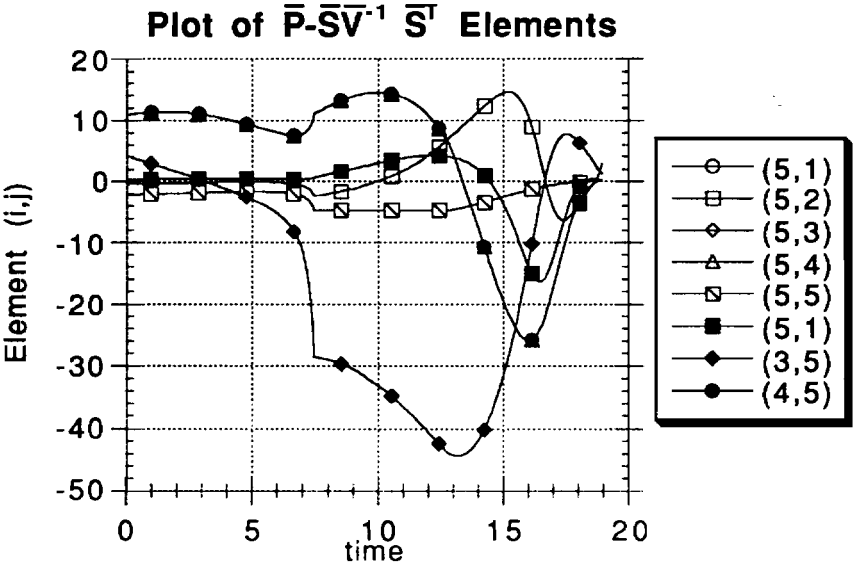


Figure 12: **Plot of Elements of Conjugate Point Condition Matrix for Two Burn Extremal.**

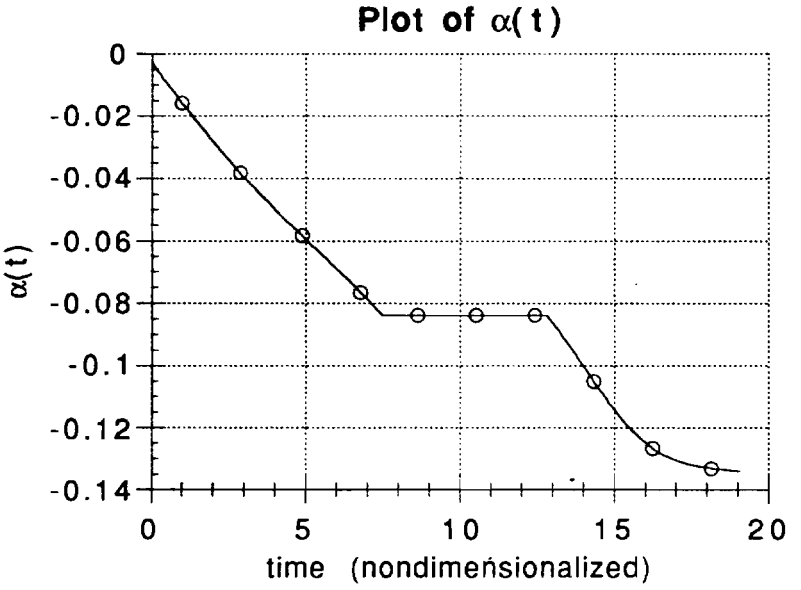


Figure 13: **Plot of α(t) for Two Burn Extremal.**

Numerical results were also obtained for a one burn case using the same free final time solution. Figs. 14 & 15 show that the normality condition is indeed met in that, respectively, $\alpha(t)$ exists and $\overline{V}^{-1}$ exists

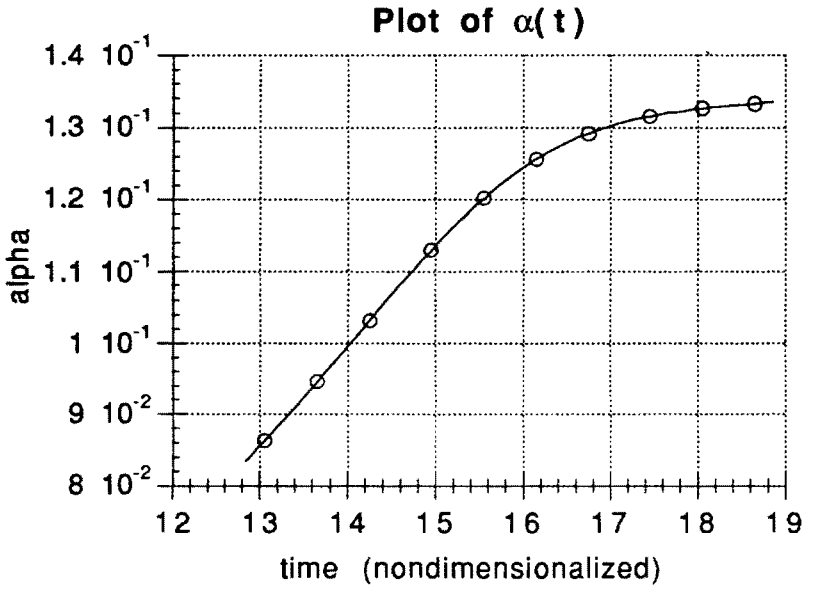


Figure 14: Plot of $\alpha(t)$ for One Burn Extremal.

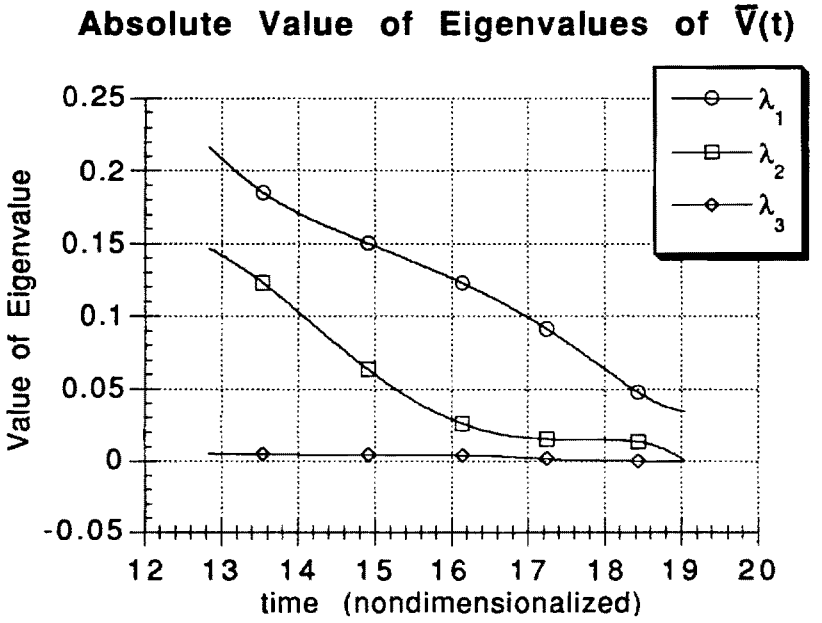


Figure 15: Plot of Eigenvalues of $\overline{V}$ for One Burn Extremal.

As seen above in Figs. 10-12, the conjugate point condition is met for the two burn case in that the elements of the matrix required for that condition are finite; thus the conditions can be met similarly for the one burn case for the same solution.

IV. Neighboring Optimal Feedback Guidance

Conveniently, construction of a neighboring optimal feedback guidance law uses the same information as that required to check the second variation of the cost functional. As a result, much of the derivation required of guidance law has been stated already. The remaining discussion will describe how to form the feedback control law and adjust the characteristics of the bang-bang control in a feedback law.

The control, $\delta\theta$, for the fixed final time problem can be found using

$$\begin{aligned}\delta\theta(t) &= -H_{\theta\theta}^{-1}\left[\left(\mathbf{f}_u^T\mathbf{P}\right)\delta\mathbf{x} + \mathbf{f}_u^T\mathbf{S}d\nu\right] \\ &= -H_{\theta\theta}^{-1}\left[\mathbf{f}_u^T\left(\mathbf{P} - \mathbf{S}\mathbf{V}^{-1}\mathbf{S}^T\right)\right]\delta\mathbf{x}\end{aligned} \tag{73}$$

Note that this continuous feedback law has been constructed by estimating $d\nu$ at each instant of time. The feedback law depends on **P**, **S**, and **V** as functions of time. A particular advantage of the sweepback method is the solution of $P(t_0)$, $S(t_0)$, and $V(t_0)$, allowing the guidance law to store these values and propagate them forward to the current time to calculate the current feedback gain. Propagation of the feedback gain may be by integration or more practically by interpolation between stored values. Use of this control should keep the trajectory on a neighboring optimal solution and deliver the spacecraft to the required orbit in the specified transfer time.

If the transfer time is not fixed, and was chosen optimally for the nominal trajectory. Then the formulation for free final time as stated earlier can be used to obtain the feedback guidance law

As seen above in Figs. 10-12, the conjugate point condition is met for the two burn case in that the elements of the matrix required for that condition are finite; thus the conditions can be met similarly for the one burn case for the same solution.

IV. Neighboring Optimal Feedback Guidance

Conveniently, construction of a neighboring optimal feedback guidance law uses the same information as that required to check the second variation of the cost functional. As a result, much of the derivation required of guidance law has been stated already. The remaining discussion will describe how to form the feedback control law and adjust the characteristics of the bang-bang control in a feedback law.

The control, $\delta\theta$, for the fixed final time problem can be found using

$$\begin{aligned}\delta\theta(t) &= -H_{\theta\theta}^{-1}\left[\left(\mathbf{f}_u^T\mathbf{P}\right)\delta\mathbf{x} + \mathbf{f}_u^T\mathbf{S}d\nu\right] \\ &= -H_{\theta\theta}^{-1}\left[\mathbf{f}_u^T\left(\mathbf{P} - \mathbf{S}\mathbf{V}^{-1}\mathbf{S}^T\right)\right]\delta\mathbf{x}\end{aligned} \tag{73}$$

Note that this continuous feedback law has been constructed by estimating $d\nu$ at each instant of time. The feedback law depends on **P**, **S**, and **V** as functions of time. A particular advantage of the sweepback method is the solution of $P(t_0)$, $S(t_0)$, and $V(t_0)$, allowing the guidance law to store these values and propagate them forward to the current time to calculate the current feedback gain. Propagation of the feedback gain may be by integration or more practically by interpolation between stored values. Use of this control should keep the trajectory on a neighboring optimal solution and deliver the spacecraft to the required orbit in the specified transfer time.

If the transfer time is not fixed, and was chosen optimally for the nominal trajectory. Then the formulation for free final time as stated earlier can be used to obtain the feedback guidance law

$$\begin{aligned}\delta\theta(t) &= -H_{\theta\theta}^{-1}\left[\left(\mathbf{f}_u^T\overline{\mathbf{P}}\right)\delta\mathbf{x} + \mathbf{f}_u^T\overline{\mathbf{S}}d\mathbf{v}\right] \\ &= -H_{\theta\theta}^{-1}\left[\mathbf{f}_u^T\left(-\overline{\mathbf{S}}\overline{\mathbf{V}}^{-1}\overline{\mathbf{S}}^T\right)\right]\delta\mathbf{x}\end{aligned} \tag{74}$$

and the change in the final time, dt_f, is:

$$dt_f = -\left[\left(\frac{\mathbf{m}^T}{\alpha} - \frac{\mathbf{n}^T}{\alpha}\overline{\mathbf{V}}^{-1}\overline{\mathbf{S}}^T\right)\right]\delta\mathbf{x} \tag{75}$$

Evaluating dt_f determines when the thrust will be turned off to complete the transfer.

The block diagram for the feedback controller needed for neighboring optimal feedback guidance is shown in Figure 16.

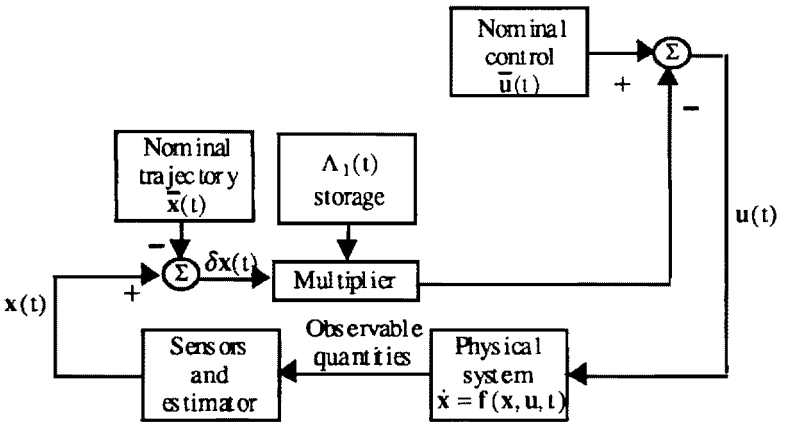


Figure 16: Diagram of Neighboring Optimal Feedback Controller Implementation.[13]

In Figure 16 $\Lambda_1(t)$ is the feedback gain for the $\delta\theta$ equation.

To determine when the new switching times should be, a variation of the switching function must be taken

$$d\mathbf{H}_T = \mathbf{H}_{Tx}d\mathbf{x} + \mathbf{H}_{T\lambda}d\lambda + \mathbf{H}_{T\theta}d\theta = 0 \tag{76}$$

Therefore, the equation to find the change in the switching time is

$$dt_s = \frac{-\mathbf{H}_{Tx}\delta\mathbf{x} - \mathbf{H}_{T\lambda}\delta\lambda - \mathbf{H}_{T\theta}\delta\theta}{\mathbf{H}_{Tx}\dot{\mathbf{x}} + \mathbf{H}_{T\lambda}\dot{\lambda} + \mathbf{H}_{T\theta}\dot{\theta}} = \frac{\partial\left(\dot{\lambda}^{\mathrm{T}}\delta\mathbf{x} - \dot{\mathbf{x}}^{\mathrm{T}}\delta\lambda\right)/\partial T}{\partial\left(-\dot{\lambda}^{\mathrm{T}}\dot{\mathbf{x}} - \dot{\mathbf{x}}^{\mathrm{T}}\dot{\lambda}\right)/\partial T} \tag{77}$$

$$= \frac{\dot{\lambda}^{\mathrm{T}}\delta\mathbf{x} - \dot{\mathbf{x}}^{\mathrm{T}}\delta\lambda}{-2\dot{\mathbf{x}}^{\mathrm{T}}\dot{\lambda}}$$

In order to implement changes in the switching times it will be necessary to predict future errors in the state. The state transition matrix should be sufficient in this matter. Such predictions will provide the foresight to make switching times earlier or later when necessary.

V. Simulation Results For The Free Final Time Solution Using The One Burn Case

The controller was implemented by simulating the one burn case for the free final time solution. The simulation corresponds to a forward integration of the states, costates, and the assumed variables, **P**, **S**, **V**, **m**, **n**, and α from the initial time to the final time.

A comparison of the nominal and actual trajectories is shown in Fig. 17. (The actual trajectory being that generated from the simulation results.) Fig. 18 shows a plot of the actual and approximated errors in the trajectories when each state is perturbed from a value of 10^{-4} (actual refers to the difference between the nominal and actual trajectories and approximated refers to the integrated error). It is seen that the actual and approximated errors are concurrent with one another; however, they do not approach zero which implies that stability in this case is not guaranteed. Figure 19 shows the approximated error during the backward integration in which the error at the final time as set to a small number. This plot seems to show a stable response. In order to examine response on a more general basis, the 2-norm of the system transition matrix was plotted in Fig. 20. Obviously, if the 2-norm went to zero, response to initial conditions would be stable. By this plot, it would seem that in general the response will not be decreasing.

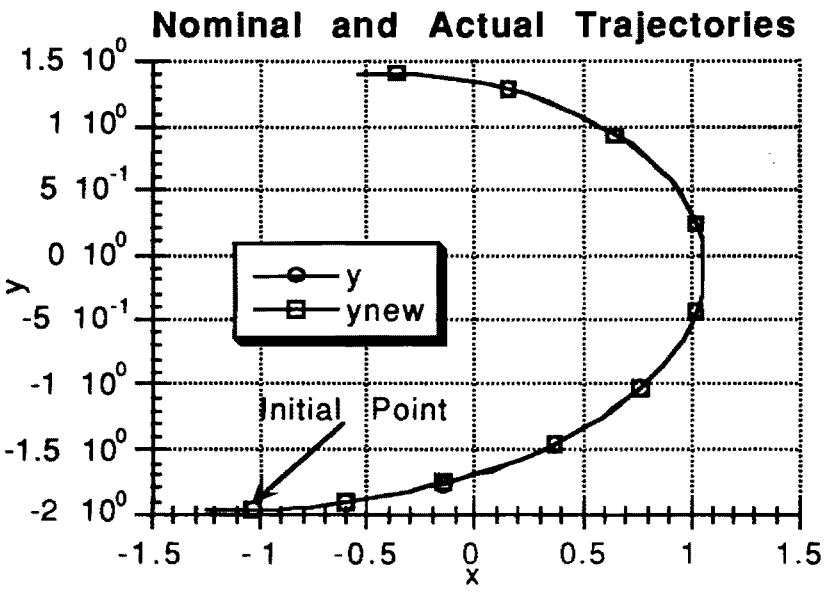


Figure 17: **Plot of the Nominal and Actual Trajectories.**

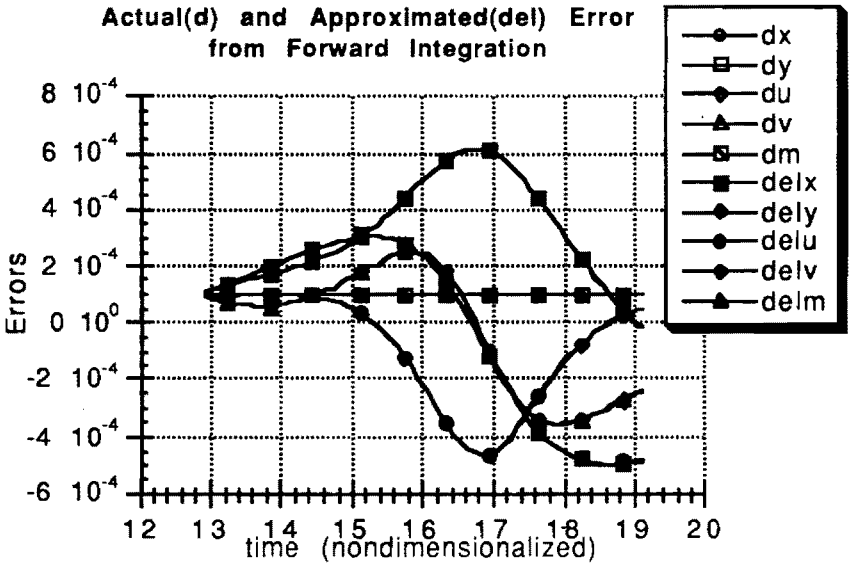


Figure 18: **Plot of the Actual and Approximated Errors.**

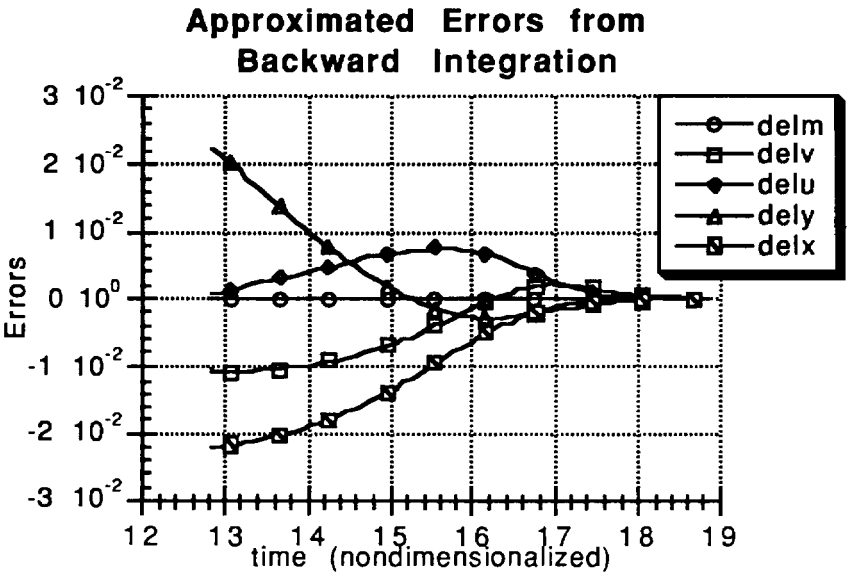


Figure 19: Plot of the Approximated Errors for Backward Integration.

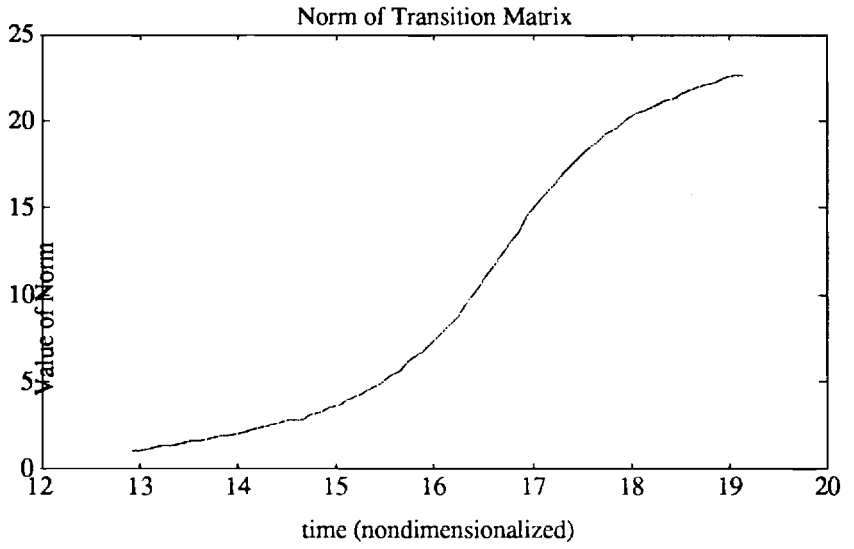


Figure 20: Plot of the Norm of the Transition Matrix.

VI. Patched Transfer Method Progress

Recent work in this research project has been directed toward developing a numerical computation scheme that performs well for a wide range of acceleration levels

and target parameters. Current effort is in assembling a method referred to here as the Patched Method.

The Patched Method is to consist of two phases: a transfer orbit optimization phase and a orbit transfer solver phase. The transfer orbit optimization phase is not so much concerned with what values of the Lagrange multipliers are required to take the craft from orbit A to orbit B as it is with how much time or fuel is required. The orbit transfer solver phase, however, is more concerned with obtaining an accurate representation of the transfer and is equally concerned with the values of the Lagrange multipliers as it is with the transfer time. Finally, it also seems reasonable to desire a method that will search for the optimal solution satisfying the target parameters but will also, if that fails, be able to return a sub-optimal solution satisfying the target parameters. In other words, it would be better to calculate a sub-optimal solution than obtain no solution at all.

Obviously, the key algorithm is one that can quickly determine the minimum transfer time (and fuel requirement) between two given orbits in a single burn. One approach that may give satisfactory results is an application of multivariate interpolation. Interpolation requires calculation and storage of data ahead of time. Therefore, the first question is what needs to be stored? To completely specify a problem the following twelve (12) values are required: semimajor axis $a(t_o)$ and $a(t_f)$, eccentricity $e(t_o)$ and $e(t_f)$, true anomaly $\nu(t_o)$ and $\nu(t_f)$, argument of perigee $\omega(t_o)$ and $\omega(t_f)$, mass $m(t_o)$, thrust T, specific impulse I_{sp}, and transfer time, t_f. To specify the problem's solution storage of the Lagrange multipliers $\lambda_x(t_o)$, $\lambda_y(t_o)$, $\lambda_u(t_o)$, $\lambda_v(t_o)$, $\lambda_m(t_o)$ is required.

The first of the nondimensionalization equations, Eqn. [23], says that $a(t_o)$ can be set to unity ($a(t_o)=1$) for any orbit transfer problem. A simple choice of coordinate axis, aligning it with perigee, will set the initial argument of perigee to be set to zero degrees, which also works for any orbit transfer problem. Neither eccentricity nor true anomaly have such favorable scaling qualities. Additionally, now that the initial values have been scaled, the final values cannot. The influence of specific impulse can be removed by

assuming a constant mass. The mass may be corrected at the end of the burn, so that calculation of the following burn is more accurate. Assuming constant mass also removes the need to store λ_m.

The influence of the thrust level may be removed by a somewhat restrictive assumption, $(\delta r/\rho)^2 << 1$, where δr is distance between the actual position of the craft and a point on a reference orbit with current radius ρ. This assumption is consistent both with low thrust levels, which stay close to a reference orbit for several revolutions, and medium thrust levels, which may only stay close to a reference orbit for a few revolutions. The advantage is that the assumption linearizes the dynamics and allows the solution to be written as

$$\begin{bmatrix} \delta \mathbf{r}(t) \\ \delta \mathbf{v}(t) \end{bmatrix} = \Phi(t,t_o)\begin{bmatrix} \delta \mathbf{r}(t_0) \\ \delta \mathbf{v}(t_0) \end{bmatrix} + T\int_{t_0}^{t}\left[\frac{\Phi(t,\tau)}{m(\tau)}\begin{bmatrix} 0 \\ e_T(\tau) \end{bmatrix}\right]d\tau \tag{78}$$

Where $\Phi(t,t_o)$ is the state transition matrix for the homogeneous solution. If the initial conditions are set to zero, then the solution is linear with respect to the thrust. Now, one solution can easily be scaled for any thrust level; however, the resulting solution must satisfy the assumption. The orbital elements can then be determined using a Jacobian matrix, as shown in Eq. [79] and easily obtained by taking partial derivatives of equations used to convert Cartesian coordinates to orbital elements.

$$\begin{bmatrix} \delta a \\ \delta e \\ \delta v \\ \delta \omega \end{bmatrix} = \mathbf{J}(a_o, e_o, v_o, \omega_o)\begin{bmatrix} \delta \mathbf{r} \\ \delta \mathbf{v} \end{bmatrix} \tag{79}$$

The number of parameters required to specify a given transfer are reduced to seven (7): $\delta a(t_f)$, $e(t_o)$, $\delta e(t_f)$, $\delta \omega(t_f)$, $v(to)$, $\delta v(t_f)$, and t_f; the transfer time stored with this data is the minimum time required for the transfer. To store the solution, it is still required to know all the Lagrange multipliers.

Before proceeding much further with this discussion a few words should be said about multivariate interpolation. It is assumed the reader is familiar with univariate, or single variable, interpolation in which there is only one independent variable and one scalar function of that variable, though any number of such dependent variables is allowed. Bivariate interpolation is then interpolation involving two independent variables and at least one function of these variables. Bivariate interpolation, and this applies equally well to multivariate interpolation, is most easily implemented when the values for independent variables are evenly spaced in a grid[14].

However, in the case of orbital transfer it would be quite difficult to obtain data with the orbital elements of the target orbit evenly spaced because this would require an iterative solver for each data point. If that process were easy, there would be no need for this approach. On the other hand, it is relatively easy to obtain a grid with the Lagrange multipliers, orbital elements of the initial orbit, and the transfer time evenly spaced. The equations of motion can then be integrated and the orbital elements of the destination orbit are known. The difficulty associated with the unevenly spaced grid is evident when one has values for the elements of the target orbit and wants to obtain the values of the Lagrange multipliers and the required transfer time. Currently, both types of grids are being considered.

The spacing of the grid is another issue altogether. The spacing of the grid, or its density, will determine the accuracy of the estimated minimum transfer time. Since speed of the algorithm and storage space required for the software are always important, the grid will need a somewhat wide spacing. There are 7 values to store for each point in the grid; whichever of the other 4 variables are evenly spaced, it is easy to determine their values though proper indexing of the grid points. If each variable is allowed n different values and 8 bytes are used to store each number in the computer, then the grid will occupy $56n^7$ bytes of memory. For the grid to need under one megabyte of space, then n=4 is required. For n=5, the grid would occupy just over 4 megabytes of space. More

than likely, different spacings of each variable would be most efficient but will this will not change the fact that the grid cannot be dense.

This interpolation will most surely produce a quick, though rough estimate of the transfer time between two chosen orbits. The next phase of the method is to obtain accurate solutions for the transfers between these orbits. Estimates for the Lagrange multipliers can be also obtained through the interpolation. These can then be used as an initial guess for a numerical solver. And, if that fails, a homotopy algorithm can be initiated from a nearby grid data point since that data point is already an accurate solution.

VII. CONCLUSIONS

Concerning the calculation of optimal transfers, the current direction has been elaborated upon, which is to test numerical methods within the framework of the Patched Method. Some work from previous reports and borrowing techniques from the literature will be incorporated along with the discussed methods. Results from this work are forthcoming.

A few conclusions, which lend themselves to study in current research work, can be made from the analyses presented here. If there are no algorithm mistakes, then it can be concluded that the extremal solutions examined may not be locally optimal solutions for fixed transfer time. However, some considerations necessary for an accurate examination of the second variation may have been overlooked. Ongoing research work is in examining why these conditions are not met.

It was found that the sufficient conditions were satisfied for the free final time problem. Software has been developed in order to simulate neighboring optimal feedback guidance. Currently, this software is not producing stable solutions. The issue of stability of the response must be investigated further and is an area of current research endeavors.

VIII. REFERENCES

[1]Brusch, R.G. and Vincent, T.L., "Low-Thrust, Minimum-Fuel, Orbital Transfers," *Astronautica Acta*, Vol. 16, pp 65-74.

[2]Edelbaum, T.N., Sackett, L.L., and Malchow, H.L.., "Optimal Low Thrust Geocentric Transfer" *AIAA Paper 73-1074, AIAA 10th Electric Propulsion Conference, Lake Tahoe, Nevada, November 1973*

[3]Redding, D.C., "Optimal Low-Thrust Transfers to Geosynchronous Orbit," NASA Lewis SUDAAR 539, Cleveland, Ohio 44135, Sept. 1983.

[4]Zondervan, K.P., Lincoln, L.J., And Caughey, T.K., "Optimal Low-Thrust, Three-Burn Orbit Transfers with Large Plane Changes," *Journal of the Astronautical Sciences*, Vol. 32, No. 3, 1984, pp. 407-427.

[5]Anderson, G.M., and Smith, E.A., "A Combined Gradient/Neighboring Extremal Algorithm for the Calculation of Optimal Transfer Trajectories between Noncoplanar Orbits using a Constant Low Thrust Rocket," *Journal of the Astornautical Sciences*, Vol. 23, No. 3, 1984, pp. 225-239

[6]Smith, I. E. "General Formulation of the Iterative Guidance Mode," Technical Memorandum X-53414, NASA-George C. Marshall Space Flight Center, March 22, 1966.

[7]Lu, P. "A General Nonlinear Guidance Law...," Guidance, Navigation, and Control Conference, Scottsdale, Arizona, 1994.

[8]Hough, Michael E. "Explicit Guidance Along an Optimal Space Curve," *Journal of Guidance, Control, and Dynamics.* Vol. 12, 1989, pp.495-504.

[9]Tempelman, Wayne. "Linear Guidance Laws for Space Missions," *Journal of Guidance, Control, and Dynamics.* Vol. 9, 1986, pp,495-502.

[10]Naidu, D. Subbaram. *Aeroassisted Orbital Transfer: Guidance and Control Strategies.* New York: Springer-Verlag, 1994.

[11]Powers, William F. "Techniques for Improved Convergence in Neighboring Optimal Guidance," *AIAA Journal.* Vol. 8, 1970, pp. 2235-2241.

[12]Chuang, C.-H., Goodson, T.D., and Hanson, J. "Optimal Trajectories of Low- And Medium-Thrust Orbit Transfers with Drag and Oblateness," submitted to *Journal of the Astronautical Sciences*, 1994

[13]Bryson, A.E., and Ho, Y.-C., *Applied Optimal Control.* Philadelphia: Taylor and Francis, 1975.

[14]Press, W.H., Flannery, B.P., Teukolsky, S.A., and Vetterling, W.T. *Numerical Recipes.* New York: Cambridge University Press, 1990.

E-16-M45
#6

FINAL REPORT

Submitted to:	NASA Marshall Space Flight Center
Grant Title:	THEORY AND COMPUTATION OF OPTIMAL LOW- AND MEDIUM-THRUST ORBIT TRANSFERS
Grant Number:	NAG8-921
Principal Investigator / Project Director:	Dr. Jason C.H. Chuang School of Aerospace Engineering Georgia Institute of Technology Atlanta, GA 30332-0150 Phone: (404)894-3075 Fax: (404)894-2760 E-mail: ch.chuang@aerospace.gatech.edu
Research Assistants:	Troy D. Goodson Laura A. Ledsinger School of Aerospace Engineering Georgia Institute of Technology
Period Covered:	July 7, 1992 to June 30, 1996
Date of Submission:	July 26, 1996

TABLE OF CONTENTS

List of Illustrations iv
List of Symbols vi
Summary xiii

Section I: The Orbit Transfer Problem 1
I.1. Introduction 1
I.2. Orbit Transfer Modeling 3

Section II: Computation of Optimal Orbit Transfers 11
II.1. Literature Review 11
II.1.1. Indirect Methods 13
II.1.2. Direct Methods 14
II.1.3. Hybrid Methods 16
II.2. Using Indirect Methods and Homotopy to Compute Solutions 17
II.2.1. Application of Optimal Control 17
II.2.2. BOUNDSCO 21
II.2.3. The Minimizing-Boundary-Condition Method 23
II.2.4. Example Two-Burn Extremal 23
II.2.5. Example Three-Burn Extremal Considering Perturbation Effects 24
II.3 A New Property of the Optimal Switching Function 28
II.3.1 Family of Extremals 31
II.3.2 Multiple Solutions in the Family 35
II.4. Conclusions 37

Section III: New Methods for Optimizing Orbit Transfers 38
III.1. Introduction 38
III.2. The Patched Method 39
III.2.1. Architecture of the Method 40
III.2.2. Using Direct Method Solutions as Guesses for Indirect Methods 42
III.2.3. Gradient of the Cost Function 44
III.2.4. An Equivalent Set of Necessary Conditions 46
III.2.5. Solution using the Patched Method with Eleven Burns 60
III.3. The Modified Patched Method (MPM) 64
III.3.1. Equivalency of MPM Conditions and Necessary Conditions 69
III.3.2. MPM Example Solutions 73
III.4. Inclusion of Perturbation Terms 80
III.5 Conclusions 83

Section IV: Guidance for Optimal Orbit Transfers 85
IV.1. Introduction 85
IV.2. Literature Review 86
IV.3. Preliminary Considerations 87
IV.4. The Second Variation for One-Burn Problems 89
IV.4.1. Neighboring Optimal Feedback Guidance 94
IV.4.2. Simulation of the Guidance Algorithm 98
IV.4.3. Time-To-Go Implementation 98
IV.5. Multiple Burn Guidance 102
IV.6. Conclusions 105

Section V: Conclusions and Recommendations for further Study 107
V.1. Transfers with Small Numbers of Burns 107
V.3. Transfers with Large Numbers of Burns 108
V.2. Multiple-Burn Guidance 109

Appendix: ORBPACK Users Manual 110

LIST OF ILLUSTRATIONS

Figure	Title	Page
2.1	Two-Burn Extremal Orbit Transfer Solution with Free Final Time.	25
2.2	Projection into X-Y Plane of Three-Burn Transfer in Ideal Gravity	27
2.3	Projection into Z-Y Plane of Three-Burn Transfer in Ideal Gravity	27
2.4	Plot of a Family of Optimal Transfers as Final Mass versus Transfer Time	32
2.5	Extending the Switching Function to Create More Optimal Transfers	33
2.6	Switching Function of Transfer at Point 7 in Figure 2.5	35
3.1	Orbital Elements of Each Transfer Orbit of Eleven Burn Solution	62
3.2	Orbit Transfer Terminal Points Indexed by Ending Orbit	63
3.3	Transfer Time Indexed by Ending Orbit for the Eleven Burn Solution	64
3.4	Diagram Illustrating the Layout of a Two-Burn Transfer	65
3.5	Components of the Angular Momentum Vector for each Transfer Orbit vs. Orbit Number of a 5-Burn Transfer with Plane Changes	74
3.6	Components of the Eccentricity Vector Vs Orbit Number for each Transfer Orbit of a 5-Burn Transfer with Plane Changes	74
3.7	Transfer Time Vs Orbit Number for each Burn of a 5-Burn Transfer with Plane Changes	75
3.8	Orbital Elements for each Transfer Orbit Vs Orbit Number of a 19-Burn Transfer	76
3.9	Transfer Time vs Orbit Number for each Burn of a 19-Burn Transfer	76
3.10	Performance Index, Final Mass, of the Extremal Trajectory vs Number of Burns Executed during the Trajectory	77
3.11	Orbital Elements for each Transfer Orbit Vs Orbit Number of a 27 Burn-Transfer	77
3.12	Transfer Time vs Orbit Number for each Burn of a 27-Burn Transfer	78
3.13	Components of the Angular Momentum Vector for each Transfer Orbit vs Orbit Number of a 27-Burn Transfer with a 63.4° Plane Change	79
3.14	Components of the Eccentricity Vector vs Orbit Number for each Transfer Orbit of a 27-Burn Transfer with a 63.4° Plane Change	79
3.15	Transfer Time vs Orbit Number for each Burn of a 27-Burn Transfer with a 63.4° Plane Change	80
3.16	Decrease in the Components of the Angular Momentum Vector for the 5-Burn Transfer of Figure 3.5 Considering Orbit Perturbations	82
3.17	Decrease in the Components of the Eccentricity Vector for the 5-Burn Transfer of Figure 3.5 Considering Orbit Perturbations	82

3.18	Decrease in the Burn Times for the 5-Burn Transfer of Figure 3.5 Considering Orbit Perturbations	83
4.1	Plot of Eigenvalues of $\overline{V}(t)$ for Two-Burn Extremal, Last Burn	95
4.2	Plot of Elements of Conjugate Point Condition Matrix for Two-Burn Extremal, Last Burn	95
4.3	Plot of Elements of Conjugate Point Condition Matrix for Two Burn Extremal, Last Burn	96
4.4	Plot of Elements of Conjugate Point Condition Matrix for Two Burn Extremal, Last Burn	96
4.5	Plot of a(t) for Two Burn Extremal, Last Burn	97
4.6	Diagram of Neighboring Optimal Feedback Controller Implementation	98
4.7	Plot of State Variation from the Nominal Trajectory vs. Time for Neighboring Optimal Trajectory (()nen) and a Trajectory Without Guidance (no subscript)	99
4.8	Plot of State Variation from the Nominal Trajectory vs. Time for Neighboring Optimal Trajectory (()nen) and a Trajectory With Guidance (no subscript)	99
4.9	Plot of Boundary Condition Error for Continuous Guidance	101
4.10	Plot of Boundary Condition Error for Discrete Guidance with Time-to-Go	101
4.11	Plot of Boundary Condition Error for Discrete Guidance During the First Burn	103
4.12	Plot of Boundary Condition Error for Discrete Guidance During the Second Burn (Continuation of Fig. 4.11)	103
4.13	Plot of Boundary Condition Error for Discrete Guidance During the Second Burn for error in x at the initial time	104
4.14	Plot of Boundary Condition Error for Discrete Guidance During the Second Burn for error in u	104
4.15	Plot of the Two Burn Orbit Transfer (from Fig. 2.1) with Initial Perturbation	105

LIST OF SYMBOLS

Bold type always indicates a vector quantity
For variables, italics indicates a scalar quantity
Vectors with Greek symbols are indicated by plain text with no italics
Unless otherwise specified, subscripts refer to partial derivatives with respect to the subscripted variable.
$\mathbf{a}_\times$ denotes the skew symmetric matrix representation of the cross product:

$$\mathbf{a}_\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \text{ where } \mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

c	Characteristic velocity of the rocket motor. $c=g_o I_{sp}$.
C_D	The drag coefficient of the spacecraft, see Eq. (1.4)
C^i	Denotes the set of i-dimensional vector functions continuous with respect all arguments, vector and/or scalar
$diag\{z_1,...z_N\}$	An $N\times N$ matrix where the jth diagonal element is z_j
$\mathbf{e}_T(t)$	Rocket motor thrust direction at time t (a unit vector)
e_x	The component of the eccentricity vector in the X-direction of OXYZ
e_y	The component of the eccentricity vector in the Y-direction of OXYZ
e_z	The component of the eccentricity vector in the Z-direction of OXYZ
$\mathbf{F}$	A Force acting on the spacecraft; cause of force denoted by subscript
g_o	Earth's gravitational acceleration at sea-level
G	An auxiliary function defined for derivation of the necessary conditions
H	The Hamiltonian, defined in the usual manner for bang-bang optimal control problems
H_T	The switching function, defined in the usual manner for bang-bang optimal control problems
h_x	The component of the angular momentum vector in the X-direction of OXYZ
h_y	The component of the angular momentum vector in the Y-direction of OXYZ

h_z	The component of the angular momentum vector in the Z-direction of OXYZ
$\mathbf{I}$	The identity matrix - if subscripts are given they denote its dimensions
I_{sp}	Specific impulse of the rocket motor
J	A cost functional or performance index
J_2	The constant describing the mass distribution of the central body; for Earth it is often taken as J_2=1082.61×10^{-6}
$m(t)$	Total spacecraft mass at time t
$\overline{\mathbf{N}}$	The 3×3 matrix $diag\{1,1,3\}$
OXYZ	This is a Rectangular Cartesian inertial reference frame. Here O is fixed at the gravitational center. The directions X,Y,Z form a right-handed system; X and Y are in the equatorial plane. Z completes the right-handed system.
R_e	The equatorial radius of the gravitating body - for oblateness effects
R^i	Denotes the set of i-dimensional real numbers
$\mathbf{r}(t)$	Radius vector from origin O of OXYZ to spacecraft's location at time t
$r(t)$	Magnitude of $\mathbf{r}(t)$
r_o	Reference altitude for reference atmospheric density (ρ_o) in atmospheric model, see Eq. (1.4)
S	The cross sectional area of spacecraft used in computing drag, see Eq. (1.4)
T_{max}	The upper bound on rocket motor thrust magnitude
t	Time
t_f	Transfer time, the total length of time required to execute the transfer
$T(t)$	Rocket motor thrust magnitude at time t
U	The gravitational potential function, see Eq. (1.5). This definition only holds in Section I
U	Denotes the set of piece-wise continuous scalar functions with one scalar argument. This definition does not hold in Section I
$u(t)$	The component of $\mathbf{v}$ in the X-direction of OXYZ
$v(t)$	The component of $\mathbf{v}$ in the Y-direction of OXYZ
$\mathbf{v}(t)$	Velocity vector in OXYZ at time t
$v(t)$	Magnitude of $\mathbf{v}(t)$
$w(t)$	The component of $\mathbf{v}$ in the Z-direction of OXYZ
W_o	Weight of spacecraft at initial point of transfer.

$x(t)$	The component of $\mathbf{r}$ in the X-direction of OXYZ
$\mathbf{x}(t)$	Vector $\left[\mathbf{r}^T(t) \quad \mathbf{v}^T(t) \quad m(t)\right]^T$; this definition changes in Section III
$y(t)$	The component of $\mathbf{r}$ in the Y-direction of OXYZ
$z(t)$	The component of $\mathbf{r}$ in the Z-direction of OXYZ
$\mathbf{z}$	State used in numerical computation
α_o, α_f	These vectors contain the orbital elements which are used to specify the initial and final orbits of the transfer, respectively.
α_i	A vector containing the orbital elements of the ith transfer orbit. For an N burn transfer the zeroth orbit is the initial orbit and the Nth orbit is the final orbit. This only applies for numeric subscripts.
β	Constant from the atmosphere model describing air density variation in the prescribed altitude region, see Eq. (1.4); this definition changes in Section III
Γ	Dummy nonsquare matrix used for generality in Lemma III.1
$\Gamma(\mathbf{x})$	Dummy matrix function used for generality in Lemma III.2
θ	The latitude angle of the current position from the equator; thrust angle in plane; this definition changes in Section III
λ_e	The Lagrange multiplier associated with the constraint on $\mathbf{e}_T$ magnitude
λ_m	The Lagrange multiplier associated with m
$\lambda_r(t)$	The Lagrange multiplier associated with $\mathbf{r}$
$\lambda_v(t)$	The Lagrange multiplier associated with $\mathbf{v}$
μ	The gravitational constant for the central body
ν_o, ν_f	Lagrange multipliers associated with boundary conditions at initial and final points
ρ_o	Atmospheric density at reference altitude (r_o) for atmospheric model, see Eq. (1.4)
τ	The independent variable used in numerical computation, represents normalized time
$\psi(\mathbf{x})$	Vector function that calculates the orbital elements for the state $\mathbf{x}$

Symbol Definitions Applying Only to Section III, Subsection III.2.2

t_{fj}	Transfer time for the jth burn

$\mathbf{e}_{T_j}(t)$	Thrust direction function for the *j*th burn
$\lambda_j(t)$	Lagrange multiplier functions for the *j*th burn
ν_j	A vector containing the Lagrange multipliers associated with the respective boundary condition. Even indices indicate association with the final orbit; odd indices indicate association with the initial orbit.
β_j	The initial condition for the mass of the *j*th burn
ξ_j	The Lagrange multiplier associated with the initial mass constraint of the *j*th burn
$m_j(t)$	The mass as a function of time for the *j*th burn
$\mathbf{f}\left(\mathbf{x}_j(t), \mathbf{e}_{T_j}(t)\right)$	Represents the state dynamics with the thrust always on, as in the one-burn problem.
$\bar{J}_j$	Adjoined cost functional for the *j*th burn of an approximate discretized problem; application of a direct optimization method is anticipated
$\overline{m}_{j,i}$	Final mass from the discrete problem, *j*th burn, at the *i*th time node
η_j	Same as ν_j except that these are for the discrete problem's boundary conditions
$\zeta_1\left(\mathbf{y}_{j,1}\right)$	Function that computes the initial orbital elements associated with the initial state $\mathbf{y}_{j,1}$ for the discretized problem
$\zeta_2\left(\mathbf{y}_{j,M}\right)$	Function that computes the initial orbital elements associated with the initial state $\mathbf{y}_{j,M}$ for the discretized problem
σ_j	The Lagrange multiplier associated with the initial mass constraint of the *j*th burn for the discretized problem
$\mu_{j,i}$	Lagrange multipliers associated with the state $\mathbf{y}_{j,i}$ for the *j*th burn at time node *i*, discretized problem
$\omega_{j,i}$	Thrust direction at time node *i*, for the *j*th burn, discretized problem
$\Delta_i\left(\mathbf{y}_{j,i}, \omega_{j,i}\right)$	Constraint at time node *i* for the *j*th burn that enforces implicit integration, discretized problem

Symbol Definitions Applying Only to Section III, except Subsection III.2.2

$\mathbf{f}(\mathbf{x}(t))$	Defined in problem $\{P\}$, the state dynamics without control
$\mathbf{g}(y(t), \mathbf{v}(t))$	Defined in problem $\{P\}$, the controlled portion of the state dynamics

$\{I\}$	Defines a set of necessary conditions that represent a typical application of optimal control theory, excluding Pontryagin's Minimum Principle; defined in subsection III.2.4.
$\{II\}$	Defines a set of necessary conditions that represent the Patched method; defined in subsection III.2.4.
$\{III\}$	Defines a set of necessary conditions that represent the Modified Patched method; defined in subsection III.3.1.
$\{P\}$	A definition for an optimal control problem that generalizes the optimal orbit transfer problem; defined in subsection III.2.4.
t_o	The fixed initial time for the problem $\{P\}$
t_f	The free final time for the problem $\{P\}$
t_{si}	The switching times defined as variables in the conditions $\{I\}$
t_i	For conditions $\{II\}$ and $\{III\}$, the initial time of burn i
t_{fi}	For conditions $\{II\}$ and $\{III\}$, the final time of burn i
$u(t)$	Defined in problem $\{P\}$, the scalar control that appears linearly in the Hamiltonian; this is assumed to be a bang-bang control
u_{max}	The maximum value allowed for the control $u(t)$
$\mathbf{v}(t)$	Defined in problem $\{P\}$, the control vector that optimal control determines to be a continuous function of time.
$\mathbf{x}(t)$	State vector in problem $\{P\}$ that contains all states except $y(t)$. In problem $\{P\}$, conditions $\{I\}$, $\{II\}$, and $\{III\}$ these states are defined for the time interval $[t_o,t_f]$
$y(t)$	One of the states in problem $\{P\}$, separated from the rest of the state vector so that it may be treated separately. In $\{P\}$ and $\{I\}$ this state is Defined for the time interval $[t_o,t_f]$
$\hat{\lambda}(t)$	Defined in conditions $\{I\}$, Lagrange multiplier vector functions associated with the whole state vector - it is partitioned as $\left[\hat{\lambda}_x{}^T(t) \quad \hat{\lambda}_y(t)\right]^T$
$\tilde{\lambda}_i(t)$	Defined in conditions $\{II\}$, Lagrange multiplier vector functions associated with the whole state vector on the ith $u=u_{max}$ arc, which is the interval $t \in \left[t_i,t_{fi}\right]$ for $i=1,..N$ - it is partitioned as $\left[\tilde{\lambda}_{x_i}{}^T(t) \quad \tilde{\lambda}_{y_i}(t)\right]$
$\hat{\lambda}_x(t)$	Defined in conditions $\{I\}$, Lagrange multiplier vector functions associated with the state vector $\mathbf{x}(t)$

$\tilde{\lambda}_y(t)$ Defined in conditions {*I*}, Lagrange multiplier vector functions associated with the state $y(t)$

$\tilde{\lambda}_{u_i}(t)$ Defined in conditions {*II*}, Lagrange multiplier vector functions associated with the state vector $\mathbf{x}(t)$ on the ith $u=u_{max}$ arc, which is the interval $t \in [t_i, t_{fi}]$ for $i=1,..N$

$\tilde{\lambda}_{y_i}(t)$ Defined in conditions {*II*}, Lagrange multiplier vector functions associated with the state $y(t)$ on the ith $u=u_{max}$ arc, which is the interval $t \in [t_i, t_{fi}]$ for $i=1,..N$

$\tilde{\nu}_o$ Defined in conditions {*I*}, the Lagrange multipliers associated with the boundary conditions at the initial time

$\tilde{\nu}_f$ Defined in conditions {*I*}, the Lagrange multipliers associated with the boundary conditions at the final time

$\tilde{\nu}_{oi}$ Defined in conditions {*II*}, the Lagrange multipliers associated with the boundary conditions at the time t_i for $i=1,...N$

$\tilde{\nu}_{fi}$ Defined in conditions {*II*}, the Lagrange multipliers associated with the boundary conditions at the time t_{fi} for $i=1,...N$

$\lambda(t)$ Defined in conditions {*III*}, the Lagrange multiplier associated with the state $\mathbf{x}$ and $\lambda = [\lambda_r^T \quad \lambda_v^T]^T$

$\lambda_r(t)$ Defined in conditions {*III*}, the Lagrange multiplier associated with $\mathbf{r}$

$\lambda_v(t)$ Defined in conditions {*III*}, the Lagrange multiplier associated with $\mathbf{v}$

ν_i Defined in conditions {*III*}, Lagrange multipliers associated with boundary conditions at i; $i=0,N+1$

Symbol Definitions Applying Only to Section IV

Lagrange multiplier symbols are the same as above, however, here they refer to a minimization problem.

$\mathbf{A}(t)$ Matrix in the differential equations for the linear correction to the state and Lagrange multipliers, control correction accounted for

$\mathbf{B}(t)$ Matrix in the differential equations for the linear correction to the state as depending on the Lagrange multiplier corrections, control correction accounted for

$\mathbf{C}(t)$	Matrix in the differential equations for the linear correction to the Lagrange multipliers as depending on the state corrections, control correction accounted for
dt_f	Correction to final time
H	The Hamiltonian
$\mathbf{P}(t)$, $\mathbf{S}(t)$, $\mathbf{V}(t)$	Sweepback matrices used to compute $\overline{\mathbf{P}}(t)$, $\overline{\mathbf{S}}(t)$, $\overline{\mathbf{V}}(t)$
$\overline{\mathbf{P}}(t)$, $\overline{\mathbf{S}}(t)$, $\overline{\mathbf{V}}(t)$	Sweepback matrices for free final time
$\mathbf{m}(t)$, $\mathbf{n}(t)$	Sweepback vectors used to compute $\overline{\mathbf{P}}(t)$, $\overline{\mathbf{S}}(t)$, $\overline{\mathbf{V}}(t)$
$\{t_1,...,t_i,...t_N\}$	Time nodes for discrete guidance with time-to-go
Δt_{gi}	Length of guidance time interval i
dt_{fi}	Correction to final time, computed at start of guidance time interval i
$\delta\mathbf{x}(t)$	Linear correction for the state of the nominal trajectory, control correction accounted for
$\alpha(t)$	Sweepback scalar
$\delta\theta(t)$	Control (thrust direction angle) correction
$\theta(t)$	Thrust direction angle (control)
$\delta\lambda(t)$	Linear correction for the Lagrange multipliers of the nominal trajectory, control correction accounted for
$d\nu$	Linear correction for the constant Lagrange multipliers, control correction accounted for
$\phi(\mathbf{x})$	Cost function for minimization problem
$\delta\psi$	Linear correction to boundary conditions, control correction accounted for
$\Omega(\mathbf{x},\nu,t)$	Hamiltonian for minimization problem

SUMMARY

This report presents new theoretical results which lead to new algorithms for the computation of fuel-optimal multiple-burn orbit transfers of low and medium thrust. Theoretical results introduced herein show how to add burns to an optimal trajectory and show that the traditional set of necessary conditions may be replaced with a much simpler set of equations. Numerical results are presented to demonstrate the utility of the theoretical results and the new algorithms.

Two indirect methods from the literature are shown to be effective for the optimal orbit transfer problem with relatively small numbers of burns. These methods are the Minimizing Boundary Condition Method (MBCM) and BOUNDSCO. Both of these methods make use of the first-order necessary conditions exactly as derived by optimal control theory.

Perturbations due to Earth's oblateness and atmospheric drag are considered. These perturbations are of greatest interest for transfers that take place between low Earth orbit altitudes and geosynchronous orbit altitudes. Example extremal solutions including these effects and computed by the aforementioned methods are presented.

It is a commonly accepted notion in the field of optimal orbit transfer that the more burns an optimal transfer executes, the lower the cost. Unfortunately, many numerical methods are not robust enough to simply "jump" from an N-burn solution to an N+1 burn solution. A new algorithm is presented which greatly eases this process. The method is just as easily implemented in the framework of MBCM as BOUNDSCO, any indirect method, or a hybrid method.

Using this algorithm and the indirect methods mentioned above, the phenomena of multiple solutions is demonstrated for the optimal orbit transfer problem. A simple empirical guideline is proposed which chooses between two or more multiple solutions when using this algorithm. It is not claimed that the algorithm will obtain the globally optimal solution.

Intuitively, one might want to think of an optimal multiple-burn transfer not as one large trajectory, but as a sequence of optimal one-burn transfers between transfer orbits that are optimally chosen. For ideal gravity, a strong relationship is shown to exist between these two problems. Based on this relationship, two new numerical methods are presented which iteratively compute optimal orbit transfers. The first method, named the Patched Method, appears to be very robust yet sluggish in convergence. The second method, named the Modified Patched Method (MPM) seems somewhat less robust but much faster in convergence. For optimal orbit transfers in ideal gravity with large numbers of burns, MPM seems to be superior to the other methods investigated in this report.

Finally, an investigation is made into a suboptimal multiple-burn guidance scheme. This scheme is, in fact, seen to have somewhat less than desirable terminal error. This terminal error is improved through a time-to-go indexing scheme. Future directions for multiple-burn guidance are suggested.

The FORTRAN code developed for this study has been collected together in a package named ORBPACK. ORBPACK and a user manual are provided. The manual is included as an appendix to this report.

SECTION I

THE ORBIT TRANSFER PROBLEM

I.1. Introduction

The most popular motor today for performing orbit transfers is of high thrust and usually a solid, sometimes a liquid rocket motor. These typically have a specific impulse, or I_{sp}, in the lower hundreds of seconds (250s-450s) and thrust in the thousands of Newtons[1] and up. In this range, they can be considered impulsive[2], applying changes in velocity on a time scale much shorter than the orbit period. For many years the study of optimal orbit transfer has focused on these impulsive motors.

With the hopes of lower fuel consumption due to an I_{sp} typically in the thousands of seconds, electric propulsion has recently grown in popularity and many studies have been performed to develop the motors; a major satellite manufacturer is already designing satellites which use a Xenon Ion Propulsion System (XIPS)[3]. The thrust produced by these motors is in the tens to thousandths of Newtons; for example, XIPS produces 18 thousandths of a Newton with an I_{sp} just under 3,000 sec. Obviously, orbit transfer maneuvers with such electric propulsion will take more time and practical transfers can no longer be modeled as impulsive. Since it is necessary to specify the maneuver with continuous functions as opposed to discrete impulsive events, the optimal transfer problem has been too complicated for exact analytical solutions.

---

[1]Hertz, J. R.., and Arson, W. J., *Space Mission Analysis and Design*, Kluwer Academic Publishers, Boston, 1991.

[2]Robbins, H. M., "An Analytical Study of the Impulsive Approximation," *AIAA Journal*, Vol. 4, No. 8, 1966, pp. 1417-1423

[3]Christensen, R. A., ed., "Space Propulsion's Latest Thrust," *Vectors*, Vol 37, No. 1, 1995, Hughes Electronics, Los Angeles.

Numerical methods for the computation of optimal orbit transfers have been widely studied. These numerical methods fall into three categories: direct, indirect, and hybrid methods. Direct methods parameterize the thrust program and then attempt to optimize these parameters while satisfying boundary conditions. Indirect methods employ the mathematics of optimal control to formulate a Two-Point Boundary Value Problem (TPBVP) which can then be approached with a variety of numerical methods. Hybrid methods are a combination of the two. These methods are often formed by simply removing difficult conditions from the TPBVP and optimizing some equivalent cost function over the released parameters.

The main objective of this research was the computation of fuel-optimal low and medium thrust orbit transfers. Here, medium thrust was taken as $1 > T/W_o \geq 0.01$ and low-thrust as $0.01 > T/W_o \geq 0.001$. This particular definition has been made because it is the initial acceleration which the rocket motor produces compared with the gravitational acceleration at that point that determines how easily changes in the initial orbit will be made. In contrast, comparing the initial rocket motor acceleration with the weight of the spacecraft as it would measure on the planet's surface does not directly indicate the motor's ability to move the spacecraft away from a very high orbit.

Of the utmost interest was the ability to compute highly efficient transfers for the ideal case. This will provide mission planners with the ability to compute a "best" transfer which can be used to judge more practical schemes. However, the ideal case does not quite represent reality; the ability to handle orbit perturbations is desirable as this would produce more realistic "best" transfers. For trajectories that spend much time near or beyond geosynchronous orbit, the dominant orbit perturbations will result from either Earth oblateness effects or atmospheric drag.[1]

Software using multiple-point shooting and modified-shooting techniques were used and produced many solutions. Using these, some characteristics of the solution have been observed and studied. Identification of these characteristics has resulted in the development of a new method for improving optimal orbit transfers. The method introduces additional burns to optimal ideal-gravity orbit transfers using an under-exploited property of the switching function. A set of improved transfers were constructed and these uncovered new properties of optimal transfers.

Furthermore, two new methods have been developed. The first is a new hybrid approach called the Patched Method. This method combines the robustness of a direct approach and the greater convergence speed of the multiple-shooting approach in a configuration that can handle transfers with large numbers of burns. However, the Patched Method pays for its robustness with speed.

The second new method is the Modified Patched Method (MPM). MPM trades back some of the sluggishness of the Patched Method for a small loss in robustness. This trade-off is accomplished by making use of properties specific to the orbit transfer problem. Some of these properties appear to be new, developed here for the first time. Overall, MPM seems to be superior to any of the other methods applied in this report.

The other objective of this research was the examination of a capable guidance algorithm for multiple-burn orbit transfer. Work on this has produced a one-burn guidance algorithm using neighboring optimal feedback control. This guidance algorithm could be used on a burn-by-burn basis to produce a sub-optimal trajectory.

I.2. Orbit Transfer Modeling

The spacecraft is represented by a point mass and assumed to be a thrusting craft acted upon by the aerodynamic drag and oblate-body gravity forces of a central body.

The central body, or planet, is also represented as a point mass positioned at its own center of gravity. Furthermore, the problem is restricted to crafts of mass much smaller than that of the central body; therefore, the planet is assumed fixed in inertial space. This inertial space is described with a rectangular Cartesian inertial reference frame (OXYZ). The central body is fixed at the center O of this frame and the z-axis is perpendicular to that body's equator. All motion within this frame agreeing with the above assumptions must satisfy Newton's Second Law:

$$\sum \mathbf{F} = \frac{d(m\mathbf{v})}{dt} \tag{1.1}$$

where m is the spacecraft's mass, v is its velocity with respect to the reference frame, and ΣF represents the sum of forces on the craft.

In this case, gravity, drag, and thrust make up the total force acting on the craft. This gives

$$m\dot{\mathbf{v}} = T\mathbf{e}_T - \mathbf{F}_{drag} - \mathbf{F}_{gravity} \tag{1.2}$$

in which the thrust is some time-varying function $T(t)$ independent of a time-varying direction $\mathbf{e}_T(t)$. This is most clearly derived by considering a momentum balance of the spacecraft as it expells mass to produce thrust; absorbing the dm/dt term into the thrust term produces Equation (1.2).

The thrust direction is expressed as the unit vector $\mathbf{e}_T(t)$. For a three-dimensional thrust vector the control requires a magnitude and three components or two angles. For two dimensional problems, the one magnitude and only two independent control components or one angle is required.

It is assumed that the fuel consumption of the motor is represented by

$$\dot{m} = -\frac{T}{g_o I_{sp}} \tag{1.3}$$

where g_o is Earth's gravitational acceleration at sea level and I_{sp} is the motor's specific impulse.

It is assumed that the atmosphere surrounding the central body can be described by an exponential model as in the standard atmosphere[4] resulting in the following aerodynamic drag force:

$$\mathbf{F}_{drag} = \frac{1}{2}\rho_o e^{-\beta(r-r_o)} S C_D v \mathbf{v} \tag{1.4}$$

where β is a constant from the atmosphere model describing air density variation in the prescribed altitude region, ρ_o is the atmosphere density at the altitude r_o, S is the cross-sectional area of the craft, C_D is the craft's drag coefficient, v is the magnitude of the velocity $\mathbf{v}$, and r is the magnitude of the position vector $\mathbf{r}$.

The gravitational potential energy to the second harmonic is[5]

$$U = \frac{\mu m}{r} + \frac{1}{2} J_2 R_e^2 \frac{\mu m}{r^3}\left(1 - 3\cos^2(\theta)\right) \tag{1.5}$$

where R_e is the equatorial radius of the central body, θ is the latitude angle of the current position from the equator, μ is the gravitational constant for the central body, and J_2 is a constant describing the mass distribution of the central body; for Earth $J_2 = 1082.61 \times 10^{-6}$. There are additional mass distribution terms, but the series is truncated here. θ is

---

[4]Anderson, J. D., *Fundamentals of Aerodynamics*, New York: McGraw-Hill Book Co., 1984.

[5]Space Technology Laboratories, *Flight Performance Handbook for Orbital Operations*, New York: Wiley, 1963.

described with Cartesian coordinates by $z=r\cos(\theta)$. This gravitational potential exerts the following force on the spacecraft:

$$\mathbf{F}_{gravity} = \frac{\partial U}{\partial \mathbf{r}} = -\left\{\frac{\mu}{r^3}\mathbf{I} + \frac{3}{2}\mu J_2 \frac{R_e^2}{r^5}\left(\overline{\mathbf{N}} - 5\left(\frac{z}{r}\right)^2 \mathbf{I}\right)\right\}\mathbf{r} \tag{1.6}$$

where $\overline{\mathbf{N}}$ =diag{1,1,3} and $\mathbf{I}$ is the identity matrix.

The equations of motion for the spacecraft are

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), T(t), \mathbf{e}_T(t)) \tag{1.7}$$

where

$$\mathbf{x}(t) = \begin{bmatrix}\mathbf{r}^T(t) & \mathbf{v}^T(t) & m(t)\end{bmatrix}^T \tag{1.8}$$

and

$$\mathbf{f}(\mathbf{x}(t), T(t), \mathbf{e}_T(t)) = \begin{Bmatrix} \mathbf{v} & \text{(1.9a)} \\ \frac{T}{m}\mathbf{e}_T - \frac{\mu}{r^3}\mathbf{r} - \left\{\frac{3}{2}\mu J_2 \frac{R_e^2}{r^5}\left(\overline{\mathbf{N}} - 5\left(\frac{z}{r}\right)^2\right)\right\}\mathbf{r} - \frac{1}{2}\frac{\rho_o}{m}e^{-\beta(r-r_o)}SC_D v\mathbf{v} & \text{(1.9b)} \\ -T/(g_o I_{sp}) & \text{(1.9c)} \end{Bmatrix}$$

The thrust magnitude has both an upper and a lower bound. The upper bound is called T_{max}, the lower bound is zero. Therefore, the following inequality constraint must be satisfied for all time $t \in [0, t_f]$:

$$0 \le T \le T_{max} \tag{1.10}$$

For the purposes of this study a simple atmosphere model was chosen. The model was not intended to accurately represent the Earth's atmosphere, or any other planet for that matter. It is implemented only for the purpose of demonstrating the methods used herein and to allow examination of its effects on the optimal transfer.

The model was defined from a reference altitude of 450 km above the planet's equator. The entire atmosphere region was assumed isothermal with a temperature of 1,000K. The density at the definition altitude was defined to be 1.184×10^{-12} kg/m^3. The definition point for this model was taken from the 1976 U.S. Standard Atmosphere[6]. Also, it was assumed that C_D=2, a common approximation for spacecraft[7], and the cross sectional area of the satellite was arbitrarily chosen to be 4π m^2.

For problems in which the ideal gravity assumption is acceptable, coasting trajectories are well understood and can be analytically represented. Therefore, it is simplest to optimize the exit, or "thrust on," point on the initial orbit and the entry, or "thrust off," point on the final orbit. A real spacecraft implementing the orbit transfer could simply wait in the initial orbit until arrival at the initial orbit exit point, indicating that the maneuver should begin.

Hence, the boundary conditions must determine all orbital elements except position on orbit, and are written as

$$\psi\left(\mathbf{x}(t_o)\right) = \alpha_o \tag{1.11a}$$

$$\psi\left(\mathbf{x}(t_f)\right) = \alpha_f \tag{1.11b}$$

where the function ψ determines these orbital elements for the state in question and α_o and α_f are vectors containing the desired values at the initial and final points, respectively. Such a determination could be accomplished several different ways.

[6]United States. COESA. *U.S. Standard Atmosphere, 1976*, Washington: GPO, 1976.
[7]King-Hele, D. *Theory of Satellite Orbits in an Atmosphere*, London, Butterworths, 1964.

However, using the angular momentum and eccentricity vectors is perhaps the simplest.[8] For planar transfers, all motion can be placed in X-Y plane and the components of the ψ function are

$$\begin{aligned} \psi_1 &= h = \mathrm{xv} - \mathrm{yu} \\ \psi_2 &= \mu e_x = \left[\left(v^2 - \mu/r\right)\mathrm{x} - \left(\mathbf{r}^T\mathbf{v}\right)\mathrm{u}\right] \\ \psi_3 &= \mu e_y = \left[\left(v^2 - \mu/r\right)\mathrm{y} - \left(\mathbf{r}^T\mathbf{v}\right)\mathrm{v}\right] \end{aligned} \tag{1.12}$$

Where h is the angular momentum, e_x is the X-component of the eccentricity vector, and e_y is the Y-component of the eccentricity vector.

In the three-dimensional case, these vectors will compose six components. Since the angular momentum and eccentricity vectors are always perpendicular, one of these components will be redundant and thus removable. There is one restriction on which component is removed; it can be seen clearly by considering the property that the vectors are always orthogonal, expressed as

$$h_x e_x + h_y e_y + h_z e_z = 0 \tag{1.13}$$

A component of one of the two vectors can be removed if it can be computed using Equation (1.13). In other words, since Eq. (1.13) always holds, knowledge of the removed component is implied and it is unnecessary to explicitly compute it. Another way to state this is to say that the six components are linearly dependent. Therefore, if for the orbit transfer problem in question, $h_z \neq 0$ on a terminal orbit, then the ψ function components can be written as

---

[8]Kaplan, M. H. *Modern Spacecraft Dynamics and Control*, New York, John Wiley & Sons, 1976.

$$\psi_1 = h_x = yw - zv \tag{1.14a}$$

$$\psi_2 = h_y = zu - xw \tag{1.14b}$$

$$\psi_3 = h_z = xv - yu \tag{1.14c}$$

$$\psi_4 = \mu e_x = \left[\left(v^2 - \mu/r\right)x - \left(\mathbf{r}^T\mathbf{v}\right)u\right] \tag{1.14d}$$

$$\psi_5 = \mu e_y = \left[\left(v^2 - \mu/r\right)y - \left(\mathbf{r}^T\mathbf{v}\right)v\right] \tag{1.14e}$$

where x,y, and z are the components of $\mathbf{r}$ in OXYZ and u,v, and w are the components of $\mathbf{v}$ in OXYZ.

If the initial or final portion of a transfer traverses altitudes where ideal gravity is not a valid assumption, then the boundary conditions likely need to be reformulated. For example, a trajectory that begins at a very low Earth-altitude cannot truly coast with zero cost because energy would be lost due to atmospheric drag. For such a transfer, it would be more realistic to fix the initial point. Likewise, some missions may be more interested in delivering the spacecraft to a specific point in space, in which case the final condition should be a rendezvous condition.

Anticipating numerical applications, note that the problem can be nondimensionalized. This aided by making all states roughly the same order. In the presentation of example solutions, the hat (^) notation will be dropped and solutions are assumed nondimensionalized unless stated otherwise. The non-dimensionalizations follow:

$$\hat{\mathbf{r}} \equiv \mathbf{r}/r^* \qquad \hat{m} \equiv m/m^* \tag{1.15a-b}$$

$$\hat{t} \equiv t/\sqrt{r^{*3}/\mu} \tag{1.15c}$$

and they require the following:

$$\hat{\mathbf{v}} \equiv \mathbf{v}/\sqrt{\mu/r^*} \qquad \hat{t}_f \equiv t_f/\sqrt{r^{*3}/\mu} \tag{1.15d-e}$$

$$\hat{r}_o \equiv r_o / r^{\star} \qquad \hat{\beta} \equiv \beta r^{\star} \tag{1.15f-g}$$

$$\left(\hat{\rho}_o \hat{S} \hat{C}_D\right) \equiv \rho_o S C_D \left(r^{\star}/m^{\star}\right) \qquad \left(\hat{g}_o \hat{I}_{sp}\right) \equiv g_o I_{sp} \sqrt{r^{\star}/\mu} \tag{1.15h-i}$$

$$\hat{T} \equiv \left(T/m^{\star}\right) \Big/ \left(\mu / r^{\star 2}\right) \qquad \hat{R}_e \equiv R_e / r^{\star} \tag{1.15j-k}$$

The choices of $r^{\star}$ and $m^{\star}$ are completely arbitrary. However, it needs to be said that after a problem is solved by these nondimensionalizations rescaling must be exercised with caution; rescaling changes the atmosphere model and changes the equatorial radius used for the oblateness terms. For example, a given transfer with nondimensionalized parameters must specify the value for $\hat{R}_e$ if oblateness effects were considered. If, after rescaling, one intends this transfer to represent a maneuver about Earth then $r^{\star}$ must be such that R_e is the radius of Earth by Equation (1.15k). Similar arguments may be made concerning the nondimensionalized parameters for atmospheric drag effects.

Substitution of Eqs. (1.15a-k) into Eqs. (1.9a-c) shows that the nondimensional dynamic equations are equivalent to Eqs. (1.9a-c) with μ=1 (the value of J_2, however, has no dimensions and is not changed). In Eq. (1.9a), choosing the scalings for **r** and t, shows that the only consistent scaling for **v** is Eq. (1.15d). Then, in Eq. (1.9b) it is clear that Eqs. (1.15a-h) and (1.15j-k) are required for consistency. Substitution into Eq. (1.9b) also shows that the factor μ appears on both sides of Eq. (1.9b), in the numerator of every term; therefore, it may be dropped from both sides. Finally, substitution into Eq. (1.9c) reveals that Eq. (1.15i) is required for consistent scaling.

SECTION II

COMPUTATION OF OPTIMAL ORBIT TRANSFERS

II.1. Literature Review

One of the earliest and most notable applications of the calculus of variations to the orbit transfer problem was by Lawden[9]. His work established the now widely-used pointer vector theory. Lawden also derived many useful analytical results including an analytical solution for the Lagrange multipliers over coast arcs in ideal gravity[10]; his expression is easily configured to trajectories where the transfer time is unconstrained. He went on to conclude that for the case of escape from a circular orbit, tangential thrusting would be nearly optimal[11]; however, he noted that this thrust program may not fare so well in other cases. Lawden studied the possibility that, in addition to arcs of maximum thrust and null thrust, arcs of intermediate-thrust may exist in an optimal transfer[12]. He later wrote a general review of rocket trajectory optimization[13] and stated that issue of the existence of intermediate-thrust arcs was still unresolved.

After Lawden's formulation was published, many other researchers produced solutions to the Lagrange multipliers over coast arcs in ideal gravity. A set of

---

[9]Lawden, D. F., *Optimal Trajectories for Space Navigation*, London, Butterworths, 1963.
[10]Lawden, D. F., "Fundamentals of Space Navigation," *Journal of the British Interplanetary Society*, Vol. 13, No. 2, 1954, pp. 87-101, 1954.
[11]Lawden, D. F., "Optimal Escape from a Circular Orbit," *Astronautica Acta*, Vol. 4, No. 3, 1958, pp. 218-233.
[12]Lawden, D. F., "Optimal Intermediate-Thrust Arcs in a Gravitational Field," *Astronautica Acta*, Vol. 8, No. 2, pp. 106-123.
[13]Lawden, D. F., "Rocket Trajectory Optimization: 1950-1963," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 4, 1991, pp. 705-711.

expressions derived by Danby[14] appear to be the earliest such work. This was actually for the equivalent problem of determining the matrizant. At almost the same time, Pines published work which derived constants of integration[15], some which apply during any part of the trajectory, even intermediate-thrust arcs, and some in restricted cases. Later, both Eckenwiler[16] and Hempel[17] produced formulations valid in a two-dimensional system. Lion and Handelsman[18] derived equations for a three-dimensional system. Glandorf[19] produced a very useful form for the Lagrange multiplier's that used the current radius, velocity, and angular momentum vectors as reference directions. Vinh[20] developed equations which reduced the solution of the Lagrange multipliers for any central force field to a problem of simple quadratures.

These analytical results have all proved useful in many studies of optimal orbit transfers. However, to date no closed-form expressions have been obtained for optimal orbit transfers, including the fuel-optimal thrust-limited case considered in this report. Therefore, numerical methods are used to produce exact solutions for this problem which has challenged the most sophisticated algorithms. These methods are traditionally divided into three types: indirect, direct, and hybrid.

---

[14]Danby, J. M. A, "The Matrizant of Keplerian Motion," *AIAA Journal*, Vol. 2, No. 1, 1964, pp. 16-19.

[15]Pines, S., "Constants of the Motion for Optimum Thrust Trajectories in a Central Force Field," *AIAA Journal*, Vol. 2, No. 11, 1964, pp. 2010-2014.

[16]Eckenwiler, M. W., "Closed-Form Lagrangian Multipliers for Coast Periods of Optimum Trajectories," *AIAA Journal*, Vol.3, No. 6, June 1965, pp. 1149-1151.

[17]Hempel, P. R., "Representation of the Lagrangian Multipliers for Coast Periods of Optimum Trajectories," *AIAA Journal*, Vol. 4, No. 4, June 1966, pp. 720-730.

[18]Lion, P. M., and Handelsman, M., "Primer Vector on Fixed-Time Impulsive Trajectories," *AIAA Journal*, Vol. 6, No. 1, 1968, pp. 127-132.

[19]Glandorf, D. R., "Lagrange Multipliers and the State Transition Matrix for Coasting Arcs," *AIAA Journal*, Vol. 7, No. 2, 1969, pp. 363-365.

[20]Vinh, N. X., "Integration of the Primer Vector in a Central Force Field," *Journal of Optimization Theory and Applications*, Vol. 9, No. 1, 1972, pp. 51-58.

II.1.1. Indirect Methods

Indirect methods convert the optimization problem into a TPBVP though optimal control theory. The most popular indirect methods by far seem to be the shooting and multiple-point shooting methods.

Among the studies using indirect methods, the work by Brown, Harrold, and Johnson[21] produced an indirect method named OPGUID/SWITCH which handles rendezvous trajectories or free entry/exit points and free final time using a modified set of boundary conditions. Results with OPGUID/SWITCH were presented for medium thrust levels and two to three burns.

Another indirect method, developed by McAdoo, Jezewski, and Dawkins[22] and dubbed OPBURN, was actually a combination of two approaches. The first approximated ideal gravity using a model for gravitational accelerations linearly varying with altitude. This assumption results in a linear steering law and was used to simplify low-accuracy calculation of the transfer. The data from this approach were used as the starting iterate of another, more accurate code. Results with this method were presented for medium thrust acceleration levels and two to three burns.

Edelbaum, Sackett, and Malchow[23] produced computer code to solve minimum time transfers (one burn) using equinoctial orbital elements as state variables. Constraints on exposure to solar radiation were considered. This method relied heavily upon the

---

[21]Brown, K. R., Harrold, E. F., and Johnson, G. W., "Rapid Optimization of Multiple-Burn Rocket Flights," *NASA CR-1430*, Sept., 1969.

[22]McAdoo, S., Jr., Jezewski, D. J., and Dawkins, G. S., "Development of a Method for Optimal Maneuver Analysis of Complex Space Missions," *NASA TN D-7882*, April, 1975.

[23]Edelbaum, T.N., Sackett, L. L., and Malchow, H. L., "Optimal Low Thrust Geocentric Transfer" AIAA Paper 73-1074, *Proceedings of the AIAA 10th Electric Propulsion Conference*, Lake Tahoe, Nevada, November 1973.

method of averaging and was named SECKSPOT. Horsewood, Suskin, and Pines[24] modified SECKSPOT to produce a code for the optimization of multiple-burn rendezvous orbit transfers with plane changes between circular orbits with low-thrust in an ideal gravity field. The transfer times for these trajectories were fixed.

A study by Redding[25] handled point-to-point, or rendezvous, low-thrust transfers with plane changes. The method presented in the study includes the reduced set of boundary conditions established earlier by Brown, et. al.[21] It was limited to transfers to geosynchronous orbits in an ideal gravity field and no results are discussed for elliptical terminal orbits. Solutions with low-thrust were obtained for transfers with two to six burns.

II.1.2. Direct Methods

The most common technique for direct methods is to discretize the control and possibly the state, then optimize the performance index by varying the control and state at each node of the independent variable. This optimization is usually subject to some constraints. In orbit transfer optimization, it obviously makes sense to use any helpful results from the application of optimal control theory. Almost universally, direct methods for orbit transfer optimization make use of a bang-bang assumption which eliminates the possibility of intermediate-thrust arcs. The control is then taken as a combination of switching times and directions.

The Direct Collocation with Nonlinear Programming (DCNLP) technique makes use of polynomial approximation to both perform integration and approximate the control

---

[24]Horsewood, J.L., Suskin, M.A., and Pines, S., "Moon Trajectory Computational Capability Development," *NASA Lewis TR-90-51*, Cleveland, Ohio 44135, July 1990.

[25]Redding, D.C., "Optimal Low-Thrust Transfers to Geosynchronous Orbit," *NASA Lewis SUDAAR 539*, Cleveland, Ohio 44135, Sept. 1983.

at nodes. Dickmanns and Wells[26] made a significant contribution using a DCNLP method based on piece-wise Hermite polynomial approximations for the state and Lagrange multipliers. More recently, Hargraves and Paris[27] used this technique in their OTIS (Optimal Trajectories by Implicit Simulation) program. The Direct Transcription and Nonlinear Programming (DTNLP) technique is very similar to DCNLP, with transcription replacing collocation for implicit integration.

Using DCNLP once then DTNLP later, Enright and Conway[28,29] examined circular, point-to-point planar transfers with ideal gravity. The methods demonstrated in these studies were shown effective for two- and three-burn trajectories. In using the DTNLP method, a technique was developed for calculating the Lagrange multipliers so that Pontryagin's Minimum Principle could be checked. In some cases, it was found that this principle had been violated.

Vulpetti and Montreali[30] used nonlinear programming to optimize transfers between circular orbits with inclinations. They did include oblateness and drag in their gravity model; their thrust acceleration level was about $0.0019g$. Example transfers included from two to four burns. Pourtakdoust and Jalali[31] used DTNLP for three-

---

[26]Dickmanns, F.D., and Well, K.H., "Approximate Solution of Optimal Control Problems Using Third Order Hermite Functions," *IFIP-TC7, VI Technical Conference on Optimization Techniques*, Novosibirsh Springer, 1974.

[27]Hargraves, C.R., Paris, S.W., Vlases, W.G., "OTIS Past, Present, and Future," *Proceedings of the 1992 AIAA conference of Guidance, Navigation, and Control*, Hilton Head, S.C. 1992

[28]Enright, P.J. and Conway, B.A., "Optimal Finite-Thrust Spacecraft Trajectories Using Collocation and Nonlinear Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 5, 1991, pp. 981-985.

[29]Enright, P.J. and Conway, B.A., "Discrete Approximations to Optimal Trajectories Using Direct Transcription and Nonlinear Programming," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 4, 1992, pp. 994-1002.

[30]Vulpetti, G. and Montereali, R.M., "High-Thrust and Low-Thrust Two-Stage LEO-LEO Transfer" *Acta Astronautica*, Vol. 15, No. 12, 1987, pp. 973-979 (84-354)

[31]Pourtakdoust, S.H. and Jalali, M.A., "Optimal Three-Dimensional Orbital Transfer Using Direct Optimization Methods," *Engineering Systems Design and Analysis*, Vol. 64-6, ASME, 1994, pp. 53-58.

dimensional two-burn transfers with a medium thrust level. All these studies mentioned above either used fixed final time, fixed entry/exit positions in orbits, or both.

Another direct method that is gaining in popularity makes use of a technique called differential inclusion.[32] Coverstone-Carroll, V. and Williams, S.N.[33] used differential inclusion concepts in a direct optimization scheme that produced one- and two-burn planar interplanetary rendezvous trajectories. The title of the study states that these trajectories are for low-thrust, but the thrust levels fit in the medium thrust range defined for this report.

II.1.3. Hybrid Methods

Methods are called hybrid if they don't fit neatly into either of the above categories. Typically, hybrid methods for the orbit transfer problem involve some use of the Lagrange multipliers and the Euler-Lagrange equations but also use direct optimization to determine other parameters of the trajectory.

Zondervan, Wood, and Caughey[34] used a hybrid method to study three-burn transfers with plane changes in ideal gravity and for thrust levels in the medium and low-thrust range. Their approach was to take the indirect setup and release the switching function constraint. The switching points were then optimized directly.

---

[32]Kisielewicz., M., *Differential Inclusions and Optimal Control*, Kluwer Academic Publishers, Boston, 1991.

[33]Coverstone-Carroll, V. and Williams, S.N., "Optimal Low Thrust Trajectories Using Differential Inclusion Concepts," *Proceedings of the AAS Rocky Mountain Guidance Conference*, Colorado, 1994.

[34]Zondervan, K.P., Wood, L.J., and Caughey, T.K., "Optimal Low-Thrust, Three-Burn Orbit Transfers with Large Plane Changes," *Journal of the Astronautical Sciences*, Vol. 32, No. 3, 1984, pp. 407-427.

Ilgen[35] used a hybrid scheme called HYTOP to compute low-thrust transfers for an Orbit Transfer Vehicle (OTV) study. The HYTOP algorithm uses the fact from optimal control theory that the pointer vector function is continuous for the duration of the transfer. The pointer vector function, and only this function, is discretized into piece-wise linear functions. The state was represented by equinoctial orbital elements. The final mass was then optimized over the choice of the pointer vector function parameters subject to the TPBVP constraints.

Each hybrid method is unique, these two are by no means representative of all that have been attempted. To date, there does not appear to be any standard hybrid methodology.

II.2. Using Indirect Methods and Homotopy to Compute Solutions

The following subsections describe work in this research effort using indirect methods and homotopy to compute solutions. Modified forms of both shooting and multiple-point shooting were found capable of computing medium thrust transfers with small numbers of burns and some low-thrust transfers. In this domain, a new method for increasing the number of burns in a transfer was developed and is based a new property of the switching function. This new method was used to demonstrate that optimal orbit transfers may have multiple solutions. Also, when using this method there is a rule-of-thumb that may help compute the more efficient of the multiple solutions, thus, avoiding the need to compute all possible transfers and comparing the cost directly. However, there is no guarantee of a global minimum.

II.2.1. Application of Optimal Control

For this problem the choice of performance index is clear:

---

[35]Ilgen, M.R., "A Hybrid Method for Computing Optimal Low-Thrust OTV Trajectories," *Proceedings of the AAS Rocky Mountain Guidance Conference*, Colorado, 1994 (AAS 94-129).

$$J = m(t_f) \tag{2.1}$$

where $m(t_f)$ represents the mass of the spacecraft including its fuel at the end of the orbit transfer. The intention, then, is to maximize the performance index, *viz.* maximize the mass at the end of the transfer.

The TPBVP is constructed using the necessary conditions in the usual manner.[36] Include the steering direction vector constraint in the Hamiltonian, which can be defined for the optimization problem as

$$H(\mathbf{x}(t),T(t),\mathbf{e}_T(t),\lambda(t)) = \lambda^T(t)\mathbf{f}(\mathbf{x}(t),T(t),\mathbf{e}_T(t)) + \lambda_c\left(\mathbf{e}_T{}^T(t)\mathbf{e}_T(t)-1\right) \tag{2.2a}$$

$$H = \lambda_r{}^T\mathbf{v} + \lambda_v{}^T\left(\frac{T}{m}\mathbf{e}_T - \frac{\mu}{r^3}\mathbf{r} - \left\{\frac{3}{2}\mu J_2\frac{R_e^2}{r^5}\left[\bar{\mathbf{N}} - 5\left(\frac{z}{r}\right)^2\right]\right\}\mathbf{r} - \frac{1}{2}\frac{\rho_o}{m}e^{-\beta(r-r_o)}SC_D v\mathbf{v}\right) - \lambda_m\frac{T}{g_o I_{sp}} + \lambda_c\left(\mathbf{e}_T{}^T\mathbf{e}_T - 1\right) \tag{2.2b}$$

from which the Euler-Lagrange equations are obtained as ODEs governing the Lagrange multipliers

$$\dot{\lambda}_r = -\left(\frac{\partial H}{\partial \mathbf{r}}\right)^T = \mu\left[\frac{\lambda_v}{r^3} - 3\frac{(\lambda_v{}^T\mathbf{r})\mathbf{r}}{r^5}\right] - \frac{1}{2}\frac{\rho_o}{m}\frac{\beta}{r}e^{-\beta(r-r_o)}SC_D v(\lambda_v{}^T\mathbf{v})\mathbf{r} + \frac{3}{2}\mu J_2 R_e^2\left[\frac{\bar{\mathbf{N}}\lambda_v}{r^5} - 5\frac{(\lambda_v{}^T\bar{\mathbf{N}}\mathbf{r})\mathbf{r}}{r^7}\right] - \frac{15}{2}\mu J_2 R_e^2\left[\frac{z^2}{r^7}\lambda_v - (\lambda_v{}^T\mathbf{r})\left(\frac{7z^2}{r^9}\mathbf{r} - \frac{2z}{r^7}\begin{bmatrix}0\\0\\1\end{bmatrix}\right)\right] \tag{2.3a}$$

$$\dot{\lambda}_v = -\left(\frac{\partial H}{\partial \mathbf{v}}\right)^T = -\lambda_r + \frac{1}{2}\frac{\rho_o}{m}e^{-\beta(r-r_o)}SC_D\left[\lambda_v v + \frac{(\lambda_v{}^T\mathbf{v})\mathbf{v}}{v}\right] \tag{2.3b}$$

---

[36]Bryson, A.E. and Ho, Y.-C., *Applied Optimal Control*, New York: Hemisphere Publishing Corporation.

$$\dot{\lambda}_m = -\frac{\partial H}{\partial m} = \frac{T}{m^2}\lambda_v^T \mathbf{e}_T - \frac{1}{2}\frac{\rho_o}{m^2} e^{-\beta(r-r_o)} S C_D v \lambda_v^T \mathbf{v} \tag{2.3c}$$

The next Euler-Lagrange equation is easily derived as

$$\frac{\partial H}{\partial \mathbf{e}_T} = \frac{\partial}{\partial \mathbf{e}_T}\left(\frac{T}{m}\lambda_v^T \mathbf{e}_T + \lambda_e\left(\mathbf{e}_T^T \mathbf{e}_T - 1\right)\cdots\right) = \frac{T}{m}\lambda_v^T + 2\lambda_e \mathbf{e}_T^T = 0 \tag{2.4}$$

so that the necessary condition is satisfied if $\mathbf{e}_T = \lambda_v/|\lambda_v|$ and $\lambda_e = (T|\lambda_v|)/(2m)$; in other words, the thrust direction is parallel to λ_v, which Lawden thus referred to as the pointer vector. This choice is further supported by a sufficient condition; note that

$$\frac{\partial H}{\partial \mathbf{e}_T \partial \mathbf{e}_T} = 2\lambda_e \mathbf{I} = \frac{T}{m}|\lambda_v|\mathbf{I} > 0 \tag{2.5}$$

when $|\lambda_v| > 0$, $T>0$, and m finite. Also, note that if any one of these is violated during a burn, the trajectory is immediately indeterminate. The choice for the Lagrange multiplier λ_e has been made and does not need to be solved for.

The switching function is derived by an application of the maximum principle. The thrust magnitude, which has bounds T_{max} and 0, will give H its maximum value if it is at its maximum value when $H_T > 0$ and at its minimum when $H_T < 0$. The switching function is

$$H_T = \frac{|\lambda_v|}{m} - \frac{\lambda_m}{g_o I_{sp}} \tag{2.6}$$

and the switching law is

$$\begin{aligned} &H_T > 0, \quad T = T_{max} \\ &H_T < 0, \quad T = 0 \end{aligned} \tag{2.7}$$

If H_T were to be zero for a finite time the control would be singular. Higher-order derivatives of H_T would then be needed to calculate T. In subsection II.1., it was noted that this singular control has been investigated by many different researchers but no conclusions are widely accepted as to when, or if, it will be part of the optimal control.

Many authors[21,34,25,37] have identified the switching law, and associated switching function, as a source of strong sensitivity in numerical solutions.

To complete the TPBVP, the methods of optimal control supply a set of natural boundary conditions

$$\lambda(t_f) = \left[\frac{\partial G}{\partial \mathbf{x}(t_f)}\left(\mathbf{x}(t_o), \mathbf{x}(t_f), \nu_o, \nu_f\right)\right]^T \tag{2.8a}$$

$$\lambda(t_o) = -\left[\frac{\partial G}{\partial \mathbf{x}(t_o)}\left(\mathbf{x}(t_o), \mathbf{x}(t_f), \nu_o, \nu_f\right)\right]^T \tag{2.8b}$$

where G is defined as

$$G\left(\mathbf{x}(t_o), \mathbf{x}(t_f), \nu_o, \nu_f\right) = m(t_f) + \nu_f{}^T\left[\psi\left(\mathbf{x}(t_f)\right) - \alpha_f\right] + \nu_o{}^T\left[\psi\left(\mathbf{x}(t_o)\right) - \alpha_o\right] \tag{2.9}$$

and $\psi(\mathbf{x})$ was defined in Equations (1.12). Therefore, the natural boundary conditions can be expressed as

---

[37]Chuang, C.-H. and Goodson, T.D. "Optimal Trajectories of Low- and Medium- Thrust Orbit Transfers with Drag and Oblateness," *Submitted to the Journal of the Astronautical Sciences*.

$$\begin{bmatrix}\lambda_r(t_f)\\ \lambda_v(t_f)\end{bmatrix}=\left[\frac{\partial\psi}{\partial \mathbf{x}}(\mathbf{x}(t_f))\right]^T \nu_f$$
$$\begin{bmatrix}\lambda_r(t_o)\\ \lambda_v(t_o)\end{bmatrix}=-\left[\frac{\partial\psi}{\partial \mathbf{x}}(\mathbf{x}(t_o))\right]^T \nu_o \qquad (2.10)$$
$$\lambda_m(t_f)=1$$

where

$$\left[\frac{\partial\psi}{\partial \mathbf{x}}(\mathbf{x})\right]^T=\begin{bmatrix}[\mathbf{r}_\times] & \left[2\mathbf{r}\mathbf{v}^T-(\mathbf{r}^T\mathbf{v})\mathbf{I}-\mathbf{v}\mathbf{r}^T\right]^T\\ [-\mathbf{v}_\times] & \left[(\mathbf{v}^T\mathbf{v})\mathbf{I}-\mathbf{v}\mathbf{v}^T+\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{5/2}}(\mathbf{r}\mathbf{r}^T-(\mathbf{r}^T\mathbf{r})\mathbf{I})\right]\end{bmatrix} \qquad (2.11)$$

and the subscript "X" denotes the skew symmetric matrix representation of the cross product.

The last condition deals with the final time. For free transfer time the transversality condition must be satisfied

$$H(\mathbf{x}(t_f),\mathbf{u}(t_f),\lambda(t_f))=-\frac{\partial G}{\partial t_f}=0 \qquad (2.12)$$

II.2.2. BOUNDSCO

One method used here to solve the TPBVP is a modification of the multiple-point shooting method. The specific algorithms are those given by H. J. Oberle in the subroutine BOUNDSCO[38], written in FORTRAN.

The state defined for the optimal control problem differs slightly from the state used in BOUNDSCO. The state used in BOUNDSCO for numerical computation is

---

[38]Oberle, H. J., "BOUNDSCO - Hinweise zur Benutzung des Mehrzielverfahrens für die numerische Lösung von Randwerproblemen mit Schaltbedingungen", Hamburger Beiträge zur Angewandten Mathematik, Berichte 6, 1987.

$$\mathbf{z}(\tau)=\left[\mathbf{x}^T(\tau) \quad \boldsymbol{\lambda}^T(\tau) \quad t_f \quad \boldsymbol{\nu}_o^T \quad \boldsymbol{\nu}_f^T\right]^T$$

and includes a state denoting the transfer time, t_f, and the $\boldsymbol{\nu}_o$ and $\boldsymbol{\nu}_f$ vectors, from the natural boundary conditions. BOUNDSCO does not allow user-defined parameters that are determined in the iteration process, only functions of time; therefore, these last quantities must be included in the state $\mathbf{z}$ and specified to have zero derivatives with respect to time. Also, BOUNDSCO is restricted to problems with a fixed partition of the independent variable; therefore, the independent variable has been defined as $\tau \in [0,1]$ with $t = \tau t_f$. This requires that the system dynamics be properly transformed to the independent variable τ so that

$$\frac{d}{d\tau}\begin{bmatrix}\mathbf{x}(\tau)\\ \boldsymbol{\lambda}(\tau)\\ t_f\\ \boldsymbol{\nu}_o\\ \boldsymbol{\nu}_f\end{bmatrix}=\begin{bmatrix}\dot{\mathbf{x}}(\tau)\\ \dot{\boldsymbol{\lambda}}(\tau)\\ 0\\ 0\\ 0\end{bmatrix}t_f d\tau$$

and these derivatives with respect to t are Eqs. (1.9a)-(1.9c) and (2.3a)-(2.3c). If $\mathbf{x}$ had N components, then the BOUNDSCO state, $\mathbf{z}$, has $2N+2(N-2)+1$ components.

BOUNDSCO addresses the switching function sensitivity problem by the explicit inclusion of switching points in the problem formulation. The number of switching points is not changed by BOUNDSCO. It iteratively drives the guessed switching points to be zeros of the switching function, Eq. (2.6). The user must then decide in which intervals to have the thrust on and in which to have thrust off. Unfortunately, with this scheme the switching law, Eq. (2.7), may not be satisfied and must be checked after BOUNDSCO claims convergence to a solution.

II.2.3. The Minimizing-Boundary-Condition Method

The second method used herein is called the Minimizing-Boundary-Condition Method (MBCM)[39]. MBCM is a modified shooting algorithm in which the switching structure of the optimal control is implicit. The program checks the switching function and the switching law to ensure that Eqs. (2.6) and (2.7) are satisfied at each integration step.

As a modification to the simple shooting method, MBCM, expands the set of available solutions by removing one boundary condition while keeping the same number of unknowns. The choice of this boundary condition is arbitrary. With the number of unknowns unchanged, the solutions become a one-dimensional family. Since this gives a much larger set of solutions, it is much easier to solve the resulting boundary-value problem. Once that is accomplished, the search for the solution that incorporates the final boundary conditions is treated as a minimization problem. The gradient is numerically calculated and used to update the initial state until the last boundary condition is satisfied. This method is about as effective as BOUNDSCO in solving the two-point boundary-value problems for the solved optimal orbit transfers.

II.2.4. Example Two-Burn Extremal

A solution is presented in this subsection, obtained by both BOUNDSCO and MBCM. It is nondimensionalized and assumes ideal gravity. The transfer is made between two planar, aligned orbits. The solution's trajectory is shown in Figure 2.1. The transfer time has been optimized and is 19.05. The initial mass is 1.608. The initial semimajor axis is 3.847 and eccentricity is 0.02378. The final orbit semimajor axis is 1.5 and eccentricity is 0.333. The product $g_o I_{sp}$ is 1.313 and the thrust level is 0.03.

---

[39]Chuang, C.-H., and Speyer, J.L., "Periodic Optimal Hypersonic SCRAMjet Cruise," *Optimal Control Applications and Methods*, Vol. 8, 1987, pp. 231-242.

Since initial altitude for the transfer is 3.905, the initial T/W_o is 0.2845 and the transfer may be categorized as a medium thrust transfer by the definition stated earlier. With the initial orbit higher than the final orbit, this transfer may be viewed as an optimal descent transfer. However, since atmospheric drag has not been considered, it should not be viewed as an optimal de-orbiting transfer, where the spacecraft would be intentionally placed in an orbit low enough for drag to eventually destroy it.

Two burns are used to complete the transfer. Most of the change in energy occurs in the longer second burn, but most of the change in angular momentum occurs in the first burn.

II.2.5. Example Three-Burn Extremal Considering Perturbation Effects

In this subsection, another example transfer is presented. This transfer was also obtained with both BOUNDSCO and MBCM. However, this is a three-burn transfer whose terminal orbits are not planar. The initial orbit has the same semimajor axis and eccentricity as the transfer from Fig. 1 except now the orbit is inclined 20°, has a right ascension of 13°, and an argument of perigee at 15°. The final orbit is also identical but inclined 1° with 0° right ascension and an argument of perigee at 0°. The thrust level and specific impulse are also the same. This solution includes oblateness effects but excludes drag effects. For the computation of oblateness effects, Earth's value for J_2 (1082.61×10^{-6}) was used along with R_e=0.9696. Since this transfer is intended to be about the earth, $r^{\star}$=6578 *km* must be specified as it ensures the correct equatorial radius scaling.

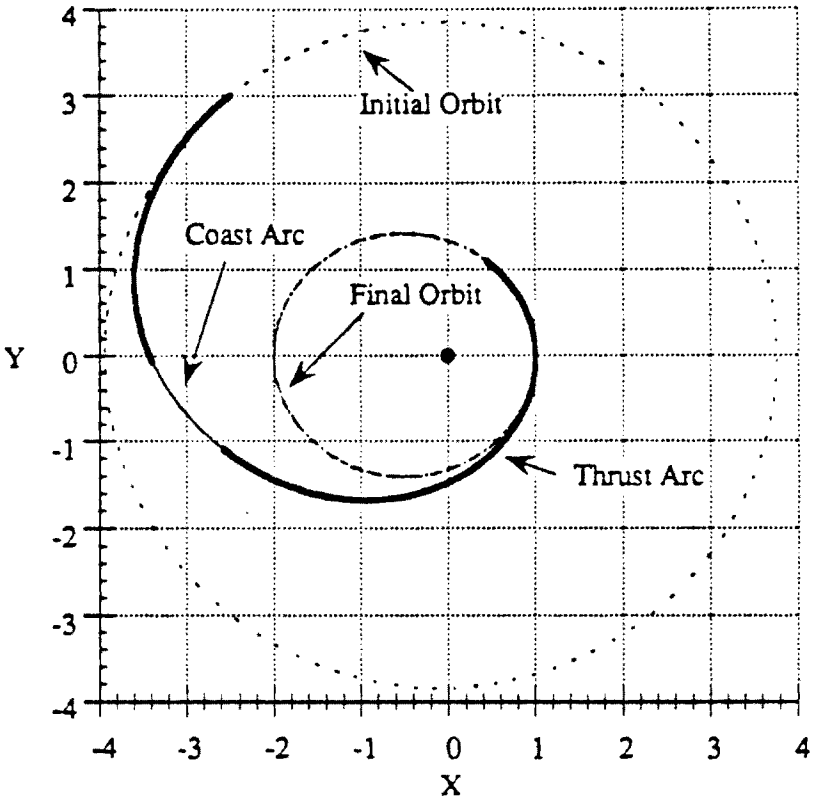


Figure 2.1 Two-Burn Extremal Orbit Transfer Solution with Free Final Time.

The trajectory is shown in Figs. 2.2-2.3. This is a fixed transfer time transfer with t_f=28.75. Recall that this is a descent trajectory; the initial orbit is higher than the final orbit. It is interesting to look at this transfer in terms of the normalized time, τ, the energy, E, the angular momentum, h, the semimajor axis, a, the eccentricity, e, the right ascension, Ω, the argument of perigee, ω, and inclination, i, for certain segments and points on the trajectory. For the first burn $\Delta\tau$=0.3616, ΔE=-0.07760, and Δh=-0.6566. The burn ends at what would be an orbit of a=2.409, e=0.5420, Ω=8.320°, ω=1.123°, and i=1.665°. For the second burn $\Delta\tau$=0.1450, ΔE=-0.1048, and Δh=-0.1310. The second burn ends at what would be an orbit of a=1.601, e=0.3742, Ω=-1.073°, ω=0.3892°, and

i=1.202°. For the third burn $\Delta\tau$=0.02420, ΔE=-0.02101, and Δh=-0.01865. The final mass for this transfer is 1.1656, the initial mass was 1.527. As a result of the oblateness effects, this transfer has poorer performance than if it could be performed in ideal gravity, where it's final mass would be 1.1659.

If drag is considered in the trajectory, performance improves and the final mass is 1.1663. This is consistent with a descending transfer whose final orbit is rather low. The altitude of perigee for the final orbit is 6578 *km* where drag needs to be considered; therefore, atmospheric drag can be used to improve performance. Obviously, with the consideration of atmospheric drag, this transfer could be considered as an optimal de-orbiting transfer.

The loss in performance caused by the oblateness effect is expected. The terminal orbits have their apses aligned; since the oblateness effect causes the line of nodes to regress, the optimal thrust program must fight this effect to return the orientation to that of the initial orbit. The improvement caused by drag is also expected for this is a descending trajectory and drag encourages descending trajectories.

It is interesting to note that the change in right ascension was almost exactly divided between the first two burns while the change in both inclination and argument of perigee happened almost entirely in the first burn. The change in inclination can be most dramatically seen in Fig. 2.3. The burn at the top of the figure is the first burn. The next two burns are difficult to distinguish but not very interesting from this vantage point. The second coasting orbit, or transfer orbit, is quite similar to the final orbit; fittingly, the third burn imparts the least energy of any of the burns.

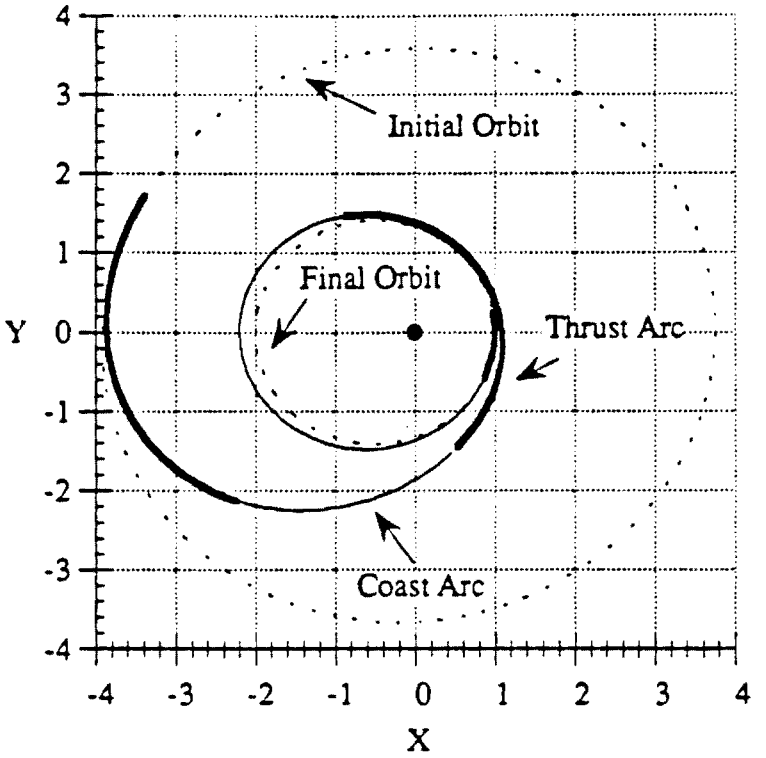


Figure 2.2 Projection into X-Y Plane of Three-Burn Transfer in Ideal Gravity

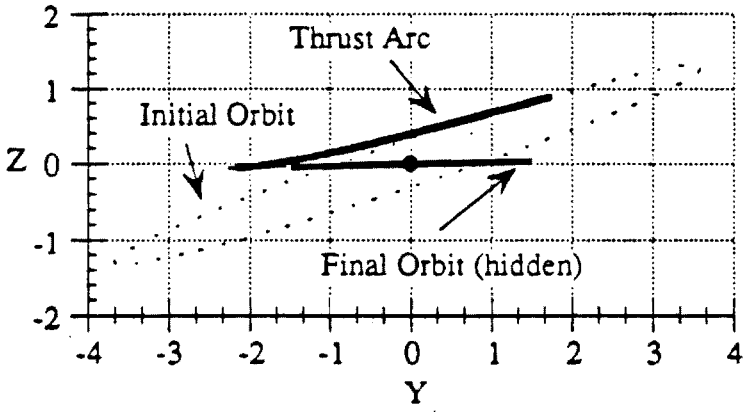


Figure 2.3 Projection into Z-Y Plane of Three-Burn Transfer in Ideal Gravity

This example demonstrates the ability of these methods to obtain exact solutions to the orbit transfer problem for nonplanar trajectories that include perturbing effects. BOUNDSCO typically can obtain such trajectories within the desired tolerance if given

the solution under ideal gravity as the initial guess. However, performance usually becomes unacceptable if the number of burns was increased beyond six; this is an empirical observation and by no means constitutes an absolute limitation of BOUNDSCO. There may well be certain cases in which BOUNDSCO can compute transfers with more than six burns quite easily; however, experience indicates that these cases are uncommon.

II.3 A New Property of the Optimal Switching Function

A very interesting property of the optimal control solution under ideal gravity is that the initial and final values of the switching function are equal. Even more interesting is that for the free transfer time problem they are both equal to zero at the initial and final times.

This property may be explained with the following theorem. In the following, C_i^0 denotes the set of i-dimensional vector functions that are continuous with respect to all arguments, vector and/or scalar, and U denotes the set of piece-wise continuous scalar functions with one scalar argument.

Theorem II.1 : Given a bang-bang optimal control problem of the form:

$$J = \int_{t_i}^{t_f} \left[L(\mathbf{x}(t),t) + M(\mathbf{x}(t),t)u(t) \right] dt \text{ where } L(\mathbf{x}(t),t) \in C_1^0 \text{ and } M(\mathbf{x}(t),t) \in C_1^0$$

and subject to the following:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t),t) + \mathbf{g}(\mathbf{x}(t),\mathbf{v}(t),t)u(t),\ \mathbf{x}(t) \in C_n^0,\ \mathbf{v}(t) \in C_m^0;$$

$$u_{min} \le u(t) \le u_{max},\ u(t) \in U\ ;$$

$$\Psi_i(\mathbf{x}(t_i)) = 0,\ \Psi_f(\mathbf{x}(t_f)) = 0,\ \Psi_i(\mathbf{x}(t_i)) \in C_{q_1}^0,\ \Psi_f(\mathbf{x}(t_f)) \in C_{q_2}^0\ ;$$

t_i and t_f are free for optimization

and satisfying the following assumptions:

(i) $L(\mathbf{x}(t_i),t_i) = L(\mathbf{x}(t_f),t_f)$;
(ii) $[\partial\psi_i(\mathbf{x}(t))/\partial\mathbf{x}(t)]\mathbf{f}(\mathbf{x}(t),t) = 0, [\partial\psi_f(\mathbf{x}(t))/\partial\mathbf{x}(t)]\mathbf{f}(\mathbf{x}(t),t) = 0$;
(iii) $u(t_i)=u(t_f)\neq 0$

then, considering the usual optimal control formulation, introduction of the $\lambda(t)$ functions, and the Hamiltonian $H(\mathbf{x}(t),\mathbf{v}(t),u(t),\lambda(t),t)$ function[36], the following statements are true:

(1) The switching function, $S(\mathbf{x}(t),\lambda(t),t) = \lambda(t)^{\mathrm{T}}\mathbf{g}(\mathbf{x}(t),\mathbf{v}(t),t) + M(\mathbf{x}(t),t)$, satisfies $S(\mathbf{x}(t_i),\lambda(t_i)) = S(\mathbf{x}(t_f),\lambda(t_f)) = -L(\mathbf{x}(t_f),t_f)/u(t_f)$ if and only if $H(\mathbf{x}(t_i),\mathbf{v}(t_i),u(t_i),\lambda(t_i),t_i) = 0$ and $H(\mathbf{x}(t_f),\mathbf{v}(t_f),u(t_f),\lambda(t_f),t_f) = 0$.

(2) If the Hamiltonian is autonomous with t_i and t_f fixed, then $S(\mathbf{x}(t_i),\lambda(t_i)) = S(\mathbf{x}(t_f),\lambda(t_f))$ and $S(\mathbf{x}(t_f),\lambda(t_f)) = \left[H(\mathbf{x}(t_f),\mathbf{v}(t_f),u(t_f),\lambda(t_f)) - L(\mathbf{x}(t_f))\right]/u(t_f)$.

Proof:

In the usual optimal control formulation, the boundary conditions at t_i and t_f result in the familiar natural boundary conditions on the Lagrange multipliers, written as

$$\lambda(t_i) = -(\partial\psi_i/\partial\mathbf{x})^{\mathrm{T}}\nu_i \in R^n$$
$$\lambda(t_f) = (\partial\psi_f/\partial\mathbf{x})^{\mathrm{T}}\nu_f \in R^n$$

which involve the constant Lagrange multiplier vectors $\nu_i \in R^{q_1}$ and $\nu_f \in R^{q_2}$, where R^i denotes the set of i-dimensional vectors with real-valued components. Now, consider the dot product of $\lambda(t_i)$ and $\lambda(t_f)$ with vectors called $\mathbf{n}_1 \in R^n$ and $\mathbf{n}_2 \in R^n$, respectively:

$$\lambda(t_i)^T \mathbf{n}_1 = -\nu_i^T (\partial \psi_i / \partial \mathbf{x}) \mathbf{n}_1$$
$$\lambda(t_f)^T \mathbf{n}_2 = -\nu_f^T (\partial \psi_f / \partial \mathbf{x}) \mathbf{n}_2$$

This shows that, at both the initial and final times, any vector in the null space of the relevant constraint gradient matrix is perpendicular to the corresponding Lagrange multiplier vector. Assumption (ii) indicates appropriate choices for $\mathbf{n}_1$ and $\mathbf{n}_2$ as

$$\mathbf{n}_1 = \mathbf{f}(\mathbf{x}(t_i), t_i)$$
$$\mathbf{n}_2 = \mathbf{f}(\mathbf{x}(t_f), t_f)$$

With these choices, the Hamiltonian at either terminal time may be written in the following form:

$$H(\mathbf{x}(t), \mathbf{v}(t), u(t), \lambda(t), t) = \left[\lambda(t)^T \mathbf{g}(\mathbf{x}(t), \mathbf{v}(t), t) + M(\mathbf{x}(t), t)\right] u(t) + L(\mathbf{x}(t), t)$$

Statements (1) and (2) follow immediately. ■

The theorem is useful because it leads to a method for finding time-optimal extremals with additional u_{max} arcs when u_{min}=0. Although not attempted in this work, it may also lead to a method for finding extremals with fewer u_{max} arcs.

Applied to the orbit transfer problem with ideal gravity and free transfer time, condition (1) implies the switching function must be zero at the entry/exit points. A similar condition was successfully used in the place of Eqs. (2.10) by Brown, et. al.[21] for free transfer time problems in ideal gravity. In that work, however, the condition was used as a boundary condition in order to reduce the number of variables in the problem.

One may make more use of this property of equal switching function values than a boundary condition; it can be used to help add burns, improving the performance of extremal orbit transfers as shall be seen in the following subsections.

II.3.1 Family of Extremals

Exploitation of the property described earlier by Theorem II.1, along with the favorable performance of these indirect methods allowed the study of the characteristics of families of solutions. Herein a family of solutions is defined as a set of solutions whose transfer times and numbers of burns vary but whose terminal orbits do not. The optimal terminal points will vary from solution to solution because they are free for optimization.

Figure 2.4 displays a family of optimal transfers. Each data point in the figure represents an extremal orbit transfer by its total transfer time and final mass. The transfers are planar and the dynamics do not take drag or oblateness effects into account. Furthermore, their terminal orbits are the same as for the transfer shown in Figure 2.1.

Though this family appears quite disjointed, it is actually quite connected. These connections can be best seen by starting at the leftmost transfer (point (1) in Fig. 2.4) and tracing solutions of increasing transfer time. The solutions from point (1) to point (2) are the original set of two-burn solutions, obtained via homotopy and a TPBVP solver (BOUNDSCO and MBCM).

At point (1) the total burn time equals the transfer time; point (1) is a one-burn solution. Point (2) represents a local optimum in transfer time; the Hamiltonian for point (2) is zero and this satisfies the transversality condition.

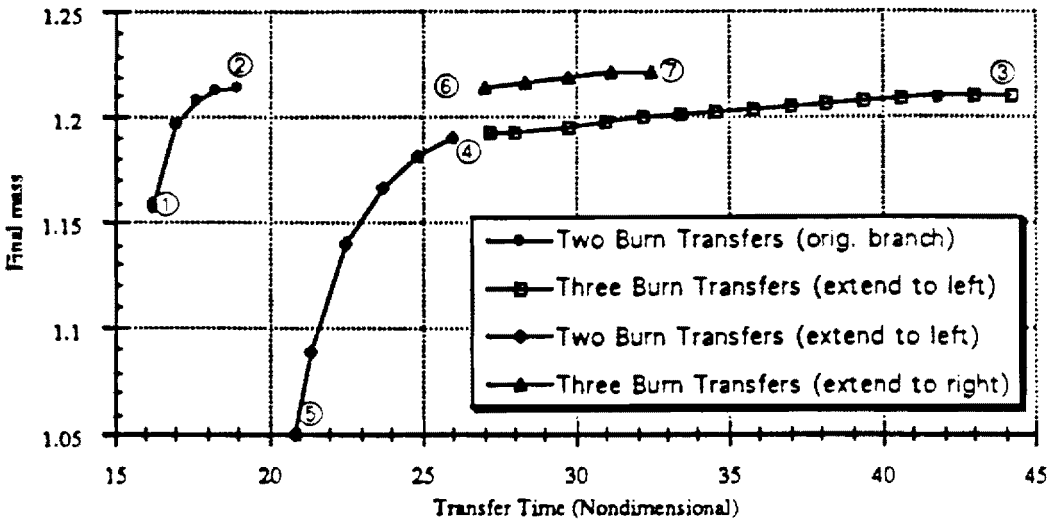


Figure 2.4 Plot of a Family of Optimal Transfers as Final Mass versus Transfer Time

As a result of Theorem II.1, the switching function at point (2) indicates the existence of additional solutions. The situation is shown in Figure 2.5. Because of the slope of H_T and the fact that it is zero at both the initial and final times (from Theorem II.1), the transfer may be extended optimally by the addition of a coast arc at the beginning and/or at the end of the transfer. This may seem trivial; one might observe that coast arcs can always be added; however, this particular situation leads to the addition of *burns*. Lawden's solution[10] to the costates on a coast arc shows that on such an arc with a vanishing Hamiltonian the switching function is periodic. This means that the switching function, once crossing zero, must return to zero. In other words, for an n burn transfer like that represented by Fig. 2.5, the periodicity of the coast arc switching function hints at the existence of two different n+1-burn solutions and an n+2-burn solution; each by different additions of coast arcs.

To optimally extend a transfer with coast arcs such that the switching function will again vanish, it is required that the switching function at a terminal orbit both be

equal to zero and have an appropriate sign for its slope: positive at the initial time and/or negative at the final time. This situation can be seen in Figure 2.5 below, for the portion of the switching function labeled "Original Transfer."

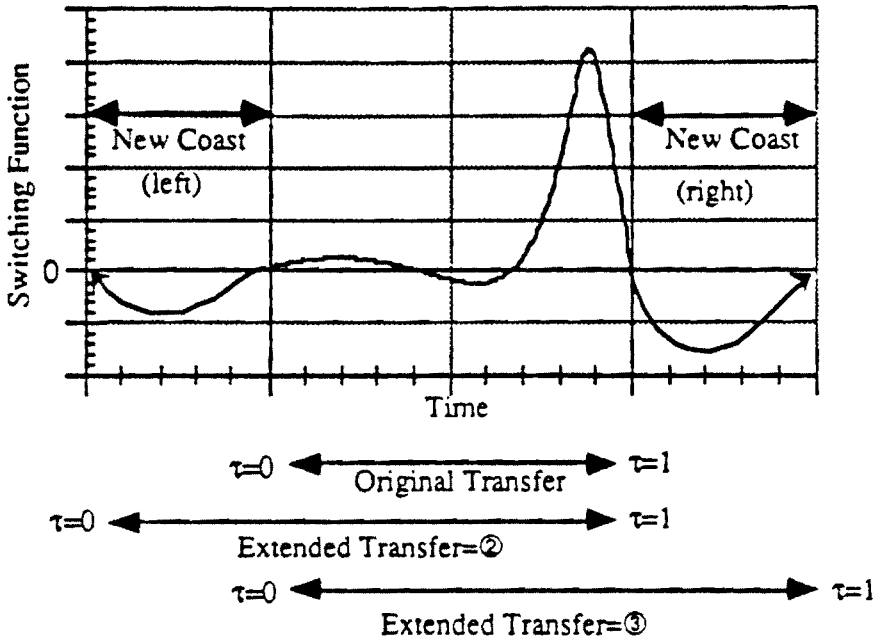


Figure 2.5 Extending the Switching Function to Create More Optimal Transfers; symbols ② and ③ refer to points in Figure 2.4

One may observe that the process does not guarantee a new burn - only a new coast arc. However, using numerical methods, one may discover that the burn can be lengthened.

Adding the coast arc is trivial; lengthening the burn arc is not. The following burn-addition procedure worked well. To add a burn to an n-burn solution with optimal transfer time that begins and ends with a burn arc: Append a coast arc to the solution at the chosen time, initial or final, making sure that states and costates are continuous. This is easily done by integrating forward from the final time or backward from the initial time. At both ends of the new coast arc the switching function must be zero. Use this

extended transfer as a guess for the numerical routine setup for an n+1-burn problem with a slightly longer transfer time. Finally, use homotopy to obtain an n+1-burn solution with a longer transfer time.

For the guesses constructed in this report, the new coast arc was extended so that the switching became positive for a finite time. Since the thrust was set to T_{max} for this new interval, the boundary conditions were violated and the new arc was a non-optimal burn because the natural boundary condition was violated. However, it was found that this new burn aided in the convergence of iterations.

There are three options for creating the next transfer in the family: extend the transfer to right, extend it to the left, or extend it in both directions. However, because of numerical difficulties, this last option was not favored. First, consider extension to the right. Physically, this corresponds to adding the new burn closer to the final orbit. The resulting transfer is represented by point (6) in Figure 2.4. Starting with point (6), solutions with longer transfer times were easily found but solutions with shorter transfer times were not found at all.

Now consider the second option, extension to the left. Physically, this corresponds to adding a burn near the initial orbit. The resulting transfer is represented by point (3) of branch (3-4-5) in Figure 2.5. Numerical difficulty was discovered in attempting to find a solution with a greater transfer time than point (3); however, solutions with lower transfer times were found constituting branch (3-4-5). Additionally, note that this branch, though a branch of optimal solutions, is unfavorable when compared to branch (6-7) of the family. This example of multiplicity may be viewed as a rearrangement of the burns in the trajectory. It has not been shown analytically, but there is likely a connection to a similar result for non-optimal impulsive trajectories[18].

By the above discussion, points (2) and (3) and (6) are, in fact, the same transfer. The only difference between these transfers is the addition of a coast arc, which makes no difference in the performance associated with the transfer. This means that the branches of the family are connected and these connections are as follows, with the transfer time increasing: (1) to (2) (which is identical to (6)) to (7); or (5) to (4) to (3) (which is identical to (2)).

Figure 2.6 shows the switching function corresponding to the transfer represented by point (7). Compare this to Figure 2.5. The situation is repeating itself; the terminal switching points in Fig. 2.6 are close to zero. Clearly, one may attempt to expand this family of transfers from point (7).

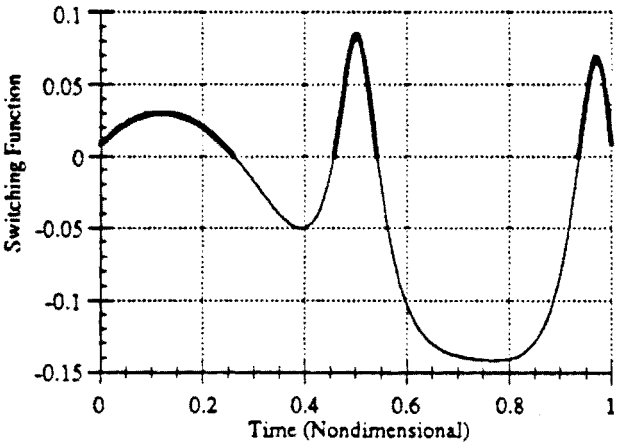


Figure 2.6 Switching Function of Transfer at Point 7 in Figure 2.5

II.3.2 Multiple Solutions in the Family

Evidence of the existence of multiple solutions was found. For a specified problem (including specification of the transfer time and the number of burns) there may exist more than one extremal transfer. Such multiple solutions are represented by any point on branch (3-4) and any point on branch (6-7) which have equal transfer times.

Conditions for multiplicity are not clear, but it is clear that solutions are not necessarily unique. It is also clear that one cannot say that just because the transfer time for one solution is longer than another, the former has a greater final mass; although this is typically an assumption made in the literature.

One cannot help but wonder why the solutions of branch (6-7) are more fuel-conservative than those of branch (3-4). Both branches are extensions of branch (1-2), but the difference is *where the new burn is placed*. When the burn was placed near the initial orbit, far from the attracting body, the branch was unfavorable. When the burn was placed near the final orbit, close to the attracting body, the branch was favorable. A principle often seen in impulsive trajectories seems to carry over in some form to finite burn trajectories; it appears to be better to implement changes in velocity near the attracting body, where changes in velocity will produce large increases in the already large kinetic energy, as opposed to far away from the attracting body, where kinetic energy is lower.

Finally, it is clear that during the burn addition process, one may control the placement of new burns. By tending to place new burns closer to the attracting body, undesirable solutions might be avoided.

The possibility of multiple solutions was recognized by Brusch[40] for one-burn low-thrust transfers originating from a circular orbit. Brusch also provides some excellent analysis concerning this phenomenon. In this research, it was found that multiple solutions exist for multiple-burn low-thrust transfers originating from an elliptical orbit. That the phenomenon may occur for the more general case indicates that there are likely many cases with multiple solutions.

---

[40]Brusch, R.G. and Vincent, T.L., "Low-Thrust, Minimum-Fuel, Orbital Transfers," *Astronautica Acta*, Vol. 16, pp. 65-74.

II.4. Conclusions

In this section the indirect methods BOUNDSCO and MBCM have demonstrated the ability to solve the optimal orbit transfer problem for small numbers of burns and small numbers of revolutions. Particular solutions have been presented in some detail. These solutions demonstrate some effects of drag and oblateness on the optimal transfer.

A new method for adding burns to time-optimal orbit transfers has been presented. This method is based on a newly observed property of the optimal switching function and a proof has been given for this property. The method has proven its practical utility by generating a family of solutions.

This family of solutions is a set of fixed-time optimal transfers with identical terminal orbits and parameterized by transfer time. Using this family, some new properties of optimal orbit transfers have been seen: multiple-burn transfers are not necessarily unique, transfers with greater transfer time do not necessarily have greater final mass, and local optima do not necessarily occur at transitions between N and $N+1$ burns when using homotopy to increase the transfer time.

Addressing the inclusion of orbit perturbations, neither BOUNDSCO nor MBCM had difficulty obtaining solutions with atmospheric drag or oblateness terms.

SECTION III

NEW METHODS FOR OPTIMIZING ORBIT TRANSFERS

III.1. Introduction

The bang-bang structure of the optimal orbit transfer solution is well-known. This means the optimal transfer is made up of a series of individual interior transfers between a sequence of orbits beginning with the specified initial orbit and ending with the desired final orbit. However, the fact that these transfers are, individually, optimal transfers has not yet been widely exploited. In this section, this notion is expressed concisely in a mathematical sense and shown to be quite useful for numerical methods.

Two methods that originated with this notion are presented. First, the Patched Method is a hybrid method with a greatly reduced number of parameters. In fact, not only are the number of parameters reduced, but they are all free for optimization.

The Patched Method also takes advantage of another simple idea: any interior one-burn transfer taken between two neighboring interior orbits of an N-burn transfer should be easier to solve than the N-burn transfer as a whole. It then makes sense to consider using the orbital elements of each intermediate transfer orbit as free parameters. Given these parameters, the performance (final mass) is computed by solving each individual one-burn problem in succession.

The Patched Method, however, pays for its robustness in speed. Therefore, it seems to be most useful as a way of refining and developing initial guesses for the second method, the Modified Patched Method (MPM). MPM is an indirect method; no variables

are directly optimized. It enforces conditions necessary for the transfer to be an extremal solution. MPM assumes a bang-bang structure; however, as in BOUNDSCO, the Patched Method, and many other methods found in the literature, MPM does not enforce satisfaction of Pontryagin's Maximum Principle. For this problem, Pontryagin's Maximum Principle supplies the switching law as Eqs (2.6) and (2.7). These methods only guarantee that the thrust will switch values at the zeros of the switching function, Eq. (2.6); they do not guarantee that the polarity will be consistent with Eq. (2.7). However, this turns out to be an easy condition to check after iterations converge.

A few reasonable and common assumptions are made in both methods. It is assumed that the only forces on the spacecraft are ideal gravity and the thrust from the rocket motor. The number of arcs of maximum thrust is assumed fixed; choosing the number of burns is often desirable and makes the problem easier to solve. The first and last arcs are assumed to be of maximum thrust; however, no generality is lost here under the assumption of ideal gravity. Arcs of intermediate thrust are assumed not to exist in the trajectory because numerical experience indicates that such arcs are rare if they exist at all. It is assumed that no part of the trajectory will be rectilinear; in other words, the angular momentum vector never vanishes. Rectilinear trajectories are unlikely to ever be of interest in an orbit transfer problem and, if they are of interest, the implications of zero angular momentum should motivate the development of specialized software.

III.2. The Patched Method

Usually, when a hybrid method is formulated the assumption is made that the solution to this new problem is always a solution to the original problem. Intuitively, this is often easy to accept. However, it is even more reassuring to prove whatever equivalency exists between the original formulation and that used by the hybrid method.

This subsection describes the architecture of the Patched Method, explaining how it functions. Also, it is shown that necessary conditions from the traditional problem statement are, in fact, equivalent to the necessary conditions which arise from the optimization loop of the Patched Method.

III.2.1. Architecture of the Method

The architecture of the Patched Method is best described as an inner and an outer loop. Given a choice of orbital elements, the inner loop solves each one-burn problem in succession. Each one-burn transfer has its terminal points and transfer time free for optimization. However, the result is a suboptimal transfer; it lacks the optimal choice of intermediate transfer orbits. The choice of transfer orbits is made by the outer loop via unconstrained minimization of the complete trajectory's fuel consumption.

The method that has been chosen for the outer loop is the conjugate gradient method. Since such methods tend to have better performance if they are supplied with an analytical gradient, such a gradient was formulated for this case; the formulation will be presented in this section. The particular FORTRAN code is taken from a common reference[41].

The architecture of this method indicates a useful new paradigm for the orbit transfer problem. One might think of the multiple-burn transfer optimization problem as optimizing the fuel used by choice of the intermediate transfer orbits, expressed as

$$\text{given } \alpha_0, \alpha_N, m_o, c, T; \quad \min_{\alpha_i,\, i=1,N-1} \sum_{i=1}^{N} t_{fi}\left(\alpha_{i-1}, \alpha_i, T, c, m_o - c\sum_{j=1}^{i} t_{f(j-1)}\right) \tag{3.1}$$

[41]Press, W.H., et al. *Numerical Recipes: the Art of Scientific Computing*, New York: Cambridge University Press, 1989.

where $t_{f_0}=0$ and $t_f(\alpha_{i-1},\alpha_i,T,m_i)$ shall be called the *transfer time function* which computes the optimal transfer time for the orbit transfer problem defined by the initial orbital elements α_{i-1}, the final orbital elements α_i, the thrust level T, the initial mass m, and the fuel consumption rate c. In (3.1), the value for the initial mass of each burn is calculated knowing the transfer times for the burns before, giving an unconstrained minimization problem; alternatively this could have been expressed as a constraint on the minimization.

In this section it will be proven that certain conditions necessary to solve (3.1) are equivalent to certain conditions necessary to solve the orbit transfer fuel-optimization problem, under certain assumptions. It will be seen that the restrictions imposed are few and quite practical; however, it is not claimed that the two problems themselves are equivalent; this may or may not be true. Nevertheless, this paradigm has certain advantages. The problem expressed in (3.1) is a parameter optimization problem. If an expression for the transfer time function were available, this would quite likely be easier to solve than the TPBVP.

Unfortunately, there are no analytical expressions or approximations for the transfer time function. The Patched Method must compute it numerically in the inner loop. The inner loop uses both Direct Collocation with Nonlinear Programming (DCNLP) and multiple-shooting to solve the one-burn transfer. Each time the optimal solution for a one-burn trajectory is required, either method may be used. For the first iteration, the choice is up to the user. If DCNLP is requested, the solution is found for a high tolerance. Once this tolerance is achieved, a multiple-shooting guess is constructed. Multiple-shooting is then used to reduce the error to the desired, lower, tolerance. If multiple-shooting was requested as the initial method and it fails, a DCNLP guess is constructed and DCNLP is attempted. If DCNLP is successful, then multiple-shooting is

used again. This structure was chosen because it was found that DCNLP was typically much too slow to use with each outer-loop iteration but multiple-shooting typically could not converge rough guesses. The failure of multiple-shooting typically occurred with the first iteration if the initial guess for the transfer was poor or the failure would occur if the outer loop took too large a step.

III.2.2. Using Direct Method Solutions as Guesses for Indirect Methods

At this point, the question of converting the solution from a direct method to the guess for an indirect method arises (the inverse process is trivial because the solution obtained by an indirect method inherently contains more information). The adjoined performance index for the jth of N one-burn problems (j=1,...,N) is

$$J_j = m_j(t_j) + \nu_{2,j-1}^T\left[\psi(\mathbf{x}_j(0)) - \alpha_{j-1}\right] + \nu_{2,j}^T\left[\psi(\mathbf{x}_j(t_{fj})) - \alpha_j\right] + \xi_j^T\left[m_j(0) - \beta_j\right] + \int_0^{t_{fj}} \lambda_j^T(t)\left[\mathbf{f}(\mathbf{x}_j(t), \mathbf{e}_{T_j}(t)) - \dot{\mathbf{x}}_j(t)\right]dt \tag{3.2}$$

where $\mathbf{x}_j(t)$ is the state, $u_j(t)$ is the control, t_{fj} is the free final time (the initial time is fixed at 0), α_{j-1} and α_j are the initial and final boundary parameters, $\psi_1(\mathbf{x})$ and $\psi_2(\mathbf{x})$ are the boundary constraint vector functions, $m_j(t)$ is the spacecraft mass, $\mathbf{f}(\mathbf{x}_j(t), \mathbf{e}_{T_j}(t))$ is the state dynamics, and $m_j(t_j)$ is the performance index to be maximized. The parameter β_j is fixed while solving each one-burn; its value is equal to initial mass constraint (m_o) or the final mass of the previous burn:

$$\beta_j = m_{j-1}\left(t_{f(j-1)}\right) \tag{3.3}$$

The discretized version for the same problem, divided into M nodes indexed by i and designed for a direct method, follows:

$$\bar{J}_j = \bar{m}_{j,M} + \eta_{2j-1}^T\left[\zeta_1(\mathbf{y}_{j,1}) - \alpha_{j-1}\right] + \eta_{2j}^T\left[\zeta_2(\mathbf{y}_{j,M}) - \alpha_j\right] + \sigma_j^T\left[\bar{m}_{j,1} - \beta_j\right] + \sum_{i=1}^{M} \mu_{j,i}^T \Delta_i(\mathbf{y}_{j,i}, \omega_{j,i}) \tag{3.4}$$

where $\mathbf{y}_i$ is the state, ω_i is the control, $\zeta_1(\mathbf{y})$ and $\zeta_2(\mathbf{y})$ are the boundary constraint functions, $\Delta_i(\mathbf{y}_i,\omega_i)$ are integration constraints, $\bar{m}_{j,i}$ is the spacecraft mass, and $\bar{m}_{j,M}$ is the performance index to be maximized Assignment of β_j, in this case, is similar to Eqn. (3.3) as follows:

$$\beta_j = \bar{m}_{j-1,M} \tag{3.5}$$

Since, for any $1<k<M$, both formulations solve the same problem with $j=k$, one can assume that $J_k = \bar{J}_k$ for any choice of α_k and α_{k+1} with $\bar{m}_{j-1,M} = m_{j-1}\left(t_{f(j-1)}\right)$, then $\frac{\partial J_j}{\partial \alpha_j} = \frac{\partial \bar{J}_j}{\partial \alpha_j}$, $\frac{\partial J_j}{\partial \alpha_{j+1}} = \frac{\partial \bar{J}_j}{\partial \alpha_{j+1}}$, and $\frac{\partial J_j}{\partial m_{j-1}\left(t_{f(j-1)}\right)} = \frac{\partial \bar{J}_j}{\partial \bar{m}_{j-1,M}}$. The implications of this are best seen in the first-order changes for both performance indices:

$$\begin{aligned}
\delta J_j = {} & \delta m_j(t_j) \\
& + \nu_{2j-1}^T\left[\psi_{1x}\left(\mathbf{x}_j(0)\right)\delta\mathbf{x}_j(0) - \delta\alpha_{j-1}\right] \\
& + \nu_{2j}^T\left[\psi_{2x}\left(\mathbf{x}_j(t_j)\right)\delta\mathbf{x}_j(t_j) - \delta\alpha_j\right] \\
& + \xi_j^T\left[\delta m_j(0) - \delta\beta_j\right] \\
& + H\left(\mathbf{x}_j(t_j), \mathbf{e}_{Tj}(t_j), \lambda_j(t_j)\right)dt_j \\
& + \int_0^{t_j}\left[H_x\left(\mathbf{x}_j(t), \mathbf{e}_{Tj}(t), \lambda_j(t)\right)\delta\mathbf{x}_j(t)\right. \\
& \left. + H_u\left(\mathbf{x}_j(t), \mathbf{e}_{Tj}(t), \lambda_j(t)\right)\delta\mathbf{e}_{Tj}(t) - \lambda_j^T\dot{\mathbf{x}}_j(t)\right]dt
\end{aligned} \tag{3.6}$$

$$
\begin{aligned}
\delta\bar{J}_j = {} & \delta\overline{m}_{j,M} \\
& + \eta_{2j-1}^{T}\left[\zeta_{1_j}\left(\mathbf{y}_{j,1}\right)\delta\mathbf{y}_{j,1} - \delta\alpha_{j-1}\right] \\
& + \eta_{2j}^{T}\left[\zeta_{2_j}\left(\mathbf{y}_{j,M}\right)\delta\mathbf{y}_{j,M} - \delta\alpha_{j}\right] \\
& + \sigma_j^{T}\left[\delta\overline{m}_{j,1} - \delta\beta_j\right] \\
& + \sum_{i=1}^{M}\mu_{j,i}^{T}\Delta_{1_{y_{j,i}}}\left(\mathbf{y}_{j,i},\omega_{j,i}\right)\delta\mathbf{y}_{j,i} + \sum_{i=1}^{M}\mu_{j,i}^{T}\Delta_{1_{\omega_{j,i}}}\left(\mathbf{y}_{j,i},\omega_{j,i}\right)\delta\omega_{j,i}
\end{aligned}
\tag{3.7}
$$

Knowing the solutions for both optimal control problems, one can substitute for the state and control of the local extremals into Eqs. (3.6)-(3.7), respectively. The resulting equations are simply:

$$\delta J_j = -\nu_{2j-1}^{T}\delta\alpha_{j-1} - \nu_{2j}^{T}\delta\alpha_j - \xi_j^{T}\delta\beta_j \tag{3.8}$$

$$\delta\bar{J}_j = -\eta_{2j-1}^{T}\delta\alpha_{j-1} - \eta_{2j}^{T}\delta\alpha_j - \sigma_j^{T}\delta\beta_j \tag{3.9}$$

It is now quite clear that since the gradients were surmised to be approximately equal, then $\nu_{2j-1}=\eta_{2j-1}$, $\nu_{2j}=\eta_{2j}$, and $\xi_j=\sigma_j$.

A simple approach to converting a solution obtained with a direct method into an appropriate guess for an indirect method is now clear. One may use a direct method to compute η_{2j-1}, η_{2j}, and σ_j; then use Eq. (2.8b) to obtain an approximation of the costates at the initial time. Knowing the states and the costates at the initial time, obtaining an approximate time history merely requires the solution of an initial value problem.

III.2.3. Gradient of the Cost Function

For this application, the gradient of the cost is required. The cost for the entire transfer is

$$J_{overall} = \sum_{j=1}^{N} t_{fj} = -\frac{g_o I_{sp}}{T}\left[m_N\left(t_{fN}\right) - m_1(0)\right] \tag{3.10}$$

where the mass at the end of the jth burn is a function of α_j, α_{j-1}, and m_{j-1}. This is obviously equivalent expression to (3.1). Omitting some simple steps of calculus and algebra, the gradient of the cost functional $J_{overall}$, may easily be written as

$$\begin{aligned}
\frac{\partial J}{\partial \alpha_i} &= \frac{-g_o I_{sp}}{T}\left[\prod_{j=i+1}^{N-1}\frac{\partial m_{j+1}\left(t_{f(j+1)}\right)}{\partial m_j\left(t_j\right)}\right]\left[\frac{\partial m_{i+1}\left(t_{f(i+1)}\right)}{\partial \alpha_i} - \frac{\partial m_{i+1}\left(t_{f(i+1)}\right)}{\partial m_i\left(t_{fi}\right)}\frac{\partial m_i\left(t_{fi}\right)}{\partial \alpha_i}\right], i = 1,\ldots,N-2 \\
\frac{\partial J}{\partial \alpha_{N-1}} &= \frac{-g_o I_{sp}}{T}\left[\frac{\partial m_N\left(t_{fN}\right)}{\partial \alpha_{N-1}} - \frac{\partial m_N\left(t_{fN}\right)}{\partial m_{N-1}\left(t_{f(N-1)}\right)}\frac{\partial m_{N-1}\left(t_{f(N-1)}\right)}{\partial \alpha_{N-1}}\right]
\end{aligned} \tag{3.11}$$

Equations (3.11) are not yet sufficient to implement the Patched Method. Expressions for evaluating the terms in Eqs. (3.11) are required. To begin, note that m_j is the performance index of the jth burn. Referring back to Eq. (3.8), one observes that

$$\frac{\partial J_j}{\partial \alpha_{j-1}} = \frac{\partial m_j\left(t_{fj}\right)}{\partial \alpha_{j-1}} = -\nu_{2j-1}{}^{\mathrm{T}} \tag{3.12a}$$

$$\frac{\partial J_j}{\partial \alpha_j} = \frac{\partial m_j\left(t_{fj}\right)}{\partial \alpha_j} = -\nu_{2j}{}^{\mathrm{T}} \tag{3.12b}$$

$$\frac{\partial J_j}{\partial \beta_j} = \frac{\partial m_j\left(t_{fj}\right)}{\partial m_{j-1}\left(t_{f(j-1)}\right)} = -\xi_j \tag{3.12c}$$

so that Eqs. (3.11) can be restated as

$$\begin{aligned}
\frac{\partial J}{\partial \alpha_i} &= \frac{g_o I_{sp}}{T}\left[\prod_{j=i+1}^{N-1}\left(-\xi_{j+1}\right)\right]\left[\nu_{2i+1}{}^{\mathrm{T}} + \xi_{i+1}\nu_{2i}{}^{\mathrm{T}}\right], i = 1,\ldots,N-2 \\
\frac{\partial J}{\partial \alpha_{N-1}} &= \frac{g_o I_{sp}}{T}\left[\nu_{2N-1}{}^{\mathrm{T}} + \xi_N \nu_{2(N-1)}{}^{\mathrm{T}}\right]
\end{aligned} \tag{3.13}$$

Which, simply, gives the gradient of the overall cost function in terms of the Lagrange multipliers from each respective one-burn problem. It is interesting to note that zeroing this gradient supplies simple relations between the Lagrange multipliers associated with the beginning of one burn to those associated with the termination of the previous burn. It is the "patching" together of optimal burns implied by these relations that inspired the name of the Patched Method.

III.2.4. An Equivalent Set of Necessary Conditions

The following results will prove useful to showing the practicality of the Patched Method conditions and, later, the practicality of the Modified Patched Method conditions:

Lemma III.1: If the matrix $\Gamma \in R^{(n-1)\times n}$ yields $rank(\Gamma) = n-1$ and satisfies $\Gamma \mathbf{f} = 0$, $\mathbf{f} \in R^n$ while $\mathbf{f}$ satisfies $\lambda^T \mathbf{f} = 0$, $\lambda \in R^n$ and $\mathbf{f}^T \mathbf{f} \neq 0$, then λ may be expressed as $\lambda = \Gamma^T \nu$ where $\nu \in R^{n-1}$.

Proof:

If $rank(\Gamma) = n-1$, $\Gamma \mathbf{f} = 0$, and $\mathbf{f}^T \mathbf{f} \neq 0$, then $\mathbf{f}$ is in the null space of Γ and it is obvious that $rank\left(\begin{bmatrix} \Gamma^T & \mathbf{f} \end{bmatrix}\right) = n$. This in turn implies that there exists a $\nu \in R^{n-1}$ and $\beta \in R$ such that

$$\lambda = \begin{bmatrix} \Gamma^T & \mathbf{f} \end{bmatrix} \begin{bmatrix} \nu \\ \beta \end{bmatrix}$$

Now, $\lambda^T \mathbf{f} = 0 \Rightarrow \nu^T \Gamma \mathbf{f} + \beta \mathbf{f}^T \mathbf{f} = 0 \Rightarrow \beta \mathbf{f}^T \mathbf{f} = 0 \Rightarrow \beta = 0.$ ■

Lemma III.2: Consider the following system of ordinary differential equations:

(i) $\dfrac{d}{dt}\mathbf{x}(t) = \mathbf{f}(t)$

(ii) $\frac{d}{dt}\lambda(t) = -\left[\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t))\right]^{\mathrm{T}}\lambda(t)$

and a matrix function $\Gamma(\mathbf{x})$, if $\frac{d}{dt}\Gamma(\mathbf{x}(t)) + \Gamma\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t)) = \mathbf{0}$, then the vector function $\lambda(t) = \Gamma(\mathbf{x}(t))^{\mathrm{T}}\nu$ is a solution to the differential equation (ii).

Proof:

To show that a function is a solution to (ii), it suffices to substitute the function into both sides of (ii) and show that equality holds.

$$L.H.S. = \frac{d}{dt}\left(\Gamma(\mathbf{x}(t))^{\mathrm{T}}\nu\right) = \left[\frac{d}{dt}\Gamma(\mathbf{x}(t))\right]^{\mathrm{T}}\nu$$

$$R.H.S. = -\left[\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t))\right]^{\mathrm{T}}\Gamma(\mathbf{x}(t))^{\mathrm{T}}\nu$$

The left hand side will equal the right hand side if $\frac{d}{dt}\Gamma(\mathbf{x}(t)) + \Gamma\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t)) = \mathbf{0}$. ■

The following definitions are precursors to a theorem that will prove the equivalence between necessary conditions for the Patched Method, which will be expressed in the definition of conditions $\{II\}$, and necessary conditions derived from the usual application of optimal control theory, which will be expressed in the definition of conditions $\{I\}$. The specific problem formulation for which such conditions are equivalent will be defined as $\{P\}$.

In what follows, C_i^0 denotes the set of i-dimensional vector functions that are continuous with respect to all arguments, vector and/or scalar, and U denotes the set of piece-wise continuous scalar functions with one scalar argument.

Definition: The optimal control problem (P) is of the form:

minimize $J = y(t_f)$ subject to the following constraints:

$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(y(t), \mathbf{v}(t))u(t)$, $\mathbf{x}(t) \in C_n^0$, $\mathbf{v}(t) \in C_m^0$;
$\dot{y}(t) = cu(t)$, $y(t) \in C_1^0$
$0 \le u(t) \le u_{max}$, $u(t) \in U$;
$\psi(\mathbf{x}(t_o)) - \alpha_o = 0$, $\psi(\mathbf{x}(t_f)) - \alpha_f = 0$, $\psi(\mathbf{x}(t)) \in C_{n-1}^0$;
$y(t_o)=y_o$;
t_f is free for optimization, t_o is fixed

and satisfying the following assumptions:

(i) $\left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t))\right]\mathbf{f}(\mathbf{x}(t)) = 0$;
(ii) $u(t_i)\neq 0$, $u(t_f)\neq 0$, and the number of arcs with $u=u_{max}$ is N
(iii) $\mathbf{g}(\mathbf{x}(t), y(t), \mathbf{v}(t))$ is not linear in $\mathbf{v}(t)$
(iv) the solution only contains arcs with $u=0$ or $u=u_{max}$;
(v) $rank\left(\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t))\right) = n-1$;
(vi) $\left(\frac{d}{dt}\frac{\partial}{\partial \mathbf{x}}\psi(\mathbf{x}(t)) + \frac{\partial}{\partial \mathbf{x}}\psi(\mathbf{x}(t))\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t))\right) = 0$ when $\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t))$
(vii) $\mathbf{f}^T(\mathbf{x}(t))\mathbf{f}(\mathbf{x}(t)) \neq 0 \ \forall t \in [t_o, t_f]$

Consider the usual optimal control formulation, introduction of the Lagrange multiplier functions $\hat{\lambda}(t)$, the Hamiltonian $H(\mathbf{x}(t), y(t), \mathbf{v}(t), u(t), \hat{\lambda}(t))$ function, and the following partition of $\hat{\lambda}(t)$

$$\hat{\lambda}(t) = \begin{bmatrix} \hat{\lambda}_x(t) \\ \hat{\lambda}_y(t) \end{bmatrix}, \ \hat{\lambda}_x(t) \in C_n^0 \ , \ \hat{\lambda}_y(t) \in C_1^0$$

Definition: For optimal control problem $\{P\}$, the conditions $\{I\}$ are

$$H\left(\mathbf{x}(t), y(t), \mathbf{v}(t), u(t), \hat{\boldsymbol{\lambda}}(t)\right) = \hat{\boldsymbol{\lambda}}_{\mathbf{x}}{}^{\mathrm{T}}(t)\,\mathbf{f}(\mathbf{x}(t)) + \left[\hat{\boldsymbol{\lambda}}_{\mathbf{x}}{}^{\mathrm{T}}(t)\,\mathbf{g}(y(t), \mathbf{v}(t)) + c\hat{\lambda}_{y}(t)\right]u(t) = 0 \tag{3.14}$$

$$\frac{d}{dt}\hat{\boldsymbol{\lambda}}_{\mathbf{x}}(t) = -\left[\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t))\right]^{\mathrm{T}}\hat{\boldsymbol{\lambda}}_{\mathbf{x}}(t) \tag{3.15}$$

$$\frac{d}{dt}\hat{\lambda}_{y}(t) = -\hat{\boldsymbol{\lambda}}_{\mathbf{x}}(t)^{\mathrm{T}}\left[\frac{\partial}{\partial y}\mathbf{g}(y(t), \mathbf{v}(t))\right]u(t) \tag{3.16}$$

$$\hat{\boldsymbol{\lambda}}_{\mathbf{x}}(t)^{\mathrm{T}}\left[\frac{\partial}{\partial \mathbf{v}}\mathbf{g}(y(t), \mathbf{v}(t))\right] = 0 \tag{3.17}$$

$$\hat{\boldsymbol{\lambda}}_{\mathbf{x}}(t_f) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_f))\right]^{\mathrm{T}}\hat{\boldsymbol{\nu}}_f \tag{3.18}$$

$$\hat{\boldsymbol{\lambda}}_{\mathbf{x}}(t_o) = -\left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_o))\right]^{\mathrm{T}}\hat{\boldsymbol{\nu}}_o \tag{3.19}$$

$$\hat{\lambda}_{y}(t_f) = 1 \tag{3.20}$$

$$\hat{\boldsymbol{\lambda}}_{\mathbf{x}}(t_{x_i})^{\mathrm{T}}\,\mathbf{g}(y(t_{x_i}), \mathbf{v}(t_{x_i})) + c\hat{\lambda}_{y}(t_{x_i}) = 0,\ i = 1,\ldots 2(N-1) \tag{3.21}$$

These are the transversality condition, Eq. (3.14); the Euler-Lagrange differential equations, Eqs. (3.15)-(3.17); the natural boundary conditions, Eqs. (3.18)-(3.20); and that the switching function vanishes at the switching points, Eq. (3.21). It is also required by conditions $\{I\}$ that the control $u(t)$ switch values across each switching point, in a pattern consistent with assumption (ii).

Definition: For optimal control problem $\{P\}$, the conditions $\{II\}$ are

$$H_i\left(\mathbf{x}(t), y(t), \mathbf{v}(t), u(t), \tilde{\lambda}_i(t)\right) = \tilde{\lambda}_{\mathbf{x}_i}(t)^T \mathbf{f}(\mathbf{x}(t)) + \left[\tilde{\lambda}_{\mathbf{x}_i}(t)^T \mathbf{g}(y(t), \mathbf{v}(t)) + c\tilde{\lambda}_{y_i}(t)\right] u(t) = 0 \tag{3.22}$$

$$\frac{d}{dt}\tilde{\lambda}_{\mathbf{x}_i}(t) = -\left[\frac{\partial}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}(t))\right]^T \tilde{\lambda}_{\mathbf{x}_i}(t) \tag{3.23}$$

$$\frac{d}{dt}\tilde{\lambda}_{y_i}(t) = -\tilde{\lambda}_{\mathbf{x}_i}(t)^T \left[\frac{\partial}{\partial y} \mathbf{g}(y(t), \mathbf{v}(t))\right] u(t) \tag{3.24}$$

$$u(t) = u_{max} \tag{3.25}$$

$$\tilde{\lambda}_{\mathbf{x}_i}(t)^T \left[\frac{\partial}{\partial \mathbf{v}} \mathbf{g}(y(t), \mathbf{v}(t))\right] = 0 \tag{3.26}$$

$$\tilde{\lambda}_{\mathbf{x}_i}(t_i) = -\left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_i))\right]^T \tilde{\nu}_{oi} \tag{3.27}$$

$$\tilde{\lambda}_{\mathbf{x}_i}(t_{fi}) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_{fi}))\right]^T \tilde{\nu}_{fi} \tag{3.28}$$

$$\tilde{\lambda}_{y_i}(t_{fi}) = 1 \tag{3.29}$$

$$\tilde{\nu}_{o(i+1)} + \tilde{\lambda}_{y(i+1)}(t_{i+1})\tilde{\nu}_{fi} = 0 \tag{3.30}$$

$$\left\{ \begin{array}{c} \mathbf{x}(t_{i+1}) = \mathbf{x}(t_{fi}) + \int_{t_{fi}}^{t_{i+1}} \mathbf{f}(\mathbf{x}(t))dt \\ y(t) = y(t_{i+1}) = y(t_{fi}) \\ u(t) = 0, \quad t \in [t_{fi}, t_{i+1}] \end{array} \right\} \tag{3.31}$$

where Eqs. (3.22)-(3.26) are defined for $t \in [t_i, t_{fi}]$ and the following partition is defined

$$\tilde{\lambda}_i(t) = \begin{bmatrix} \tilde{\lambda}_{\mathbf{x}_i}(t) \\ \tilde{\lambda}_{y_i}(t) \end{bmatrix}, \quad \tilde{\lambda}_{\mathbf{x}_i}(t) \in C_n^0, \quad \tilde{\lambda}_{y_i}(t) \in C_1^0$$

All conditions in $\{II\}$ are defined for i=1..N except Eqs. (3.30)-(3.31) which are only defined for i=1..N-1. Finally, $t_1=t_o$ is assigned and the value for t_f is seen to be t_{fN}.

Theorem III.1: If and only if

$$\left\{\mathbf{x}(t),y(t),\mathbf{v}(t),u(t),\hat{\lambda}(t)\middle|\, t\in\left[t_o,t_f\right]\right\},\hat{\nu}_o,\hat{\nu}_f,t_f,\left\{t_{si}\middle|\, i=1,\ldots 2(N-1)\right\} \tag{3.32}$$

satisfies $\{I\}$ then

$$\left\{\mathbf{x}(t),y(t),\mathbf{v}(t),u(t)\middle|\, t\in\left[t_1,t_{fN}\right]\right\},\left\{\left(\tilde{\lambda}_i(t),\ t\in\left[t_i,t_{fi}\right]\right),t_i,t_{fi},\tilde{\nu}_{oi},\tilde{\nu}_{fi}\middle|\, i=1,\ldots N\right\} \tag{3.33}$$

satisfies $\{II\}$, assuming that the constraints and assumptions from $\{P\}$ are satisfied.

Proof:

It will be shown, for both the necessary and sufficient parts of the theorem, that if one condition holds, then a construction may be made such that the other is satisfied.

Assume that (3.32) satisfies $\{I\}$. A solution to $\{II\}$ will be constructed from (3.32) going backwards in time. For the last u=u_{max} arc, where $t\in\left[t_N,t_{fN}\right]$, define

$$\tilde{\nu}_{fN}=\hat{\nu}_f \tag{3.34}$$

$$t_N=t_{s2(N-1)} \tag{3.35}$$

$$\tilde{\lambda}_N(t)=\hat{\lambda}(t),\ t\in\left[t_N,t_{fN}\right] \tag{3.36}$$

These definitions allow Eqs. (3.14)-(3.18) and Eq. (3.20) to imply satisfaction of Eqs. (3.22)-(3.26), (3.28), and (3.29) for $t\in\left[t_N,t_{fN}\right]$ and i=N. Eq. (3.21) for i=2(N-1) specifies that the switching function is zero at the beginning of this interval, where t=t_N. Therefore, satisfaction of Eq. (3.22) for i=N clearly implies that $\tilde{\lambda}_{sN}{}^{T}(t_N)\,\mathbf{f}\left(\mathbf{x}(t_N)\right)=0$. Considering this result, Lemma III.1 with $\Gamma\left(\mathbf{x}(t_N)\right)=\dfrac{\partial\psi}{\partial\mathbf{x}}\left(\mathbf{x}(t_N)\right)$ and assumptions (i), (v),

and (vii), implies that there exists a $-\tilde{\nu}_{oN} \in R^{n-1}$ such that Eq. (3.27) is satisfied for $i=N$. This completes the definitions for the final $u=u_{max}$ arc.

Consider the next interval, where $t \in \left[t_{f(N-1)}, t_N\right]$, the definitions will now be extended into this interval. Define $t_{f(N-1)}=t_{s(2N-3)}$. The conditions $\{I\}$ specify that $u(t)=0$ for t in this interval. This implies that Eqs. (3.31) with $i=N-1$ are consistent with the switching structure of $\{I\}$. Define

$$\tilde{\lambda}_{x_N}(t) = \hat{\lambda}_x(t),\ t \in \left[t_{f(N-1)}, t_N\right]$$

With this definition and that Eq. (3.27) is satisfied for $i=N$, Lemma III.2 with $\Gamma(\mathbf{x}(t)) = \frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t))$ and assumption (vi) implies that the Lagrange multipliers satisfy

$$\tilde{\lambda}_x\left(t_{f(N-1)}\right) = \left[\frac{\partial \psi}{\partial \mathbf{x}}\left(\mathbf{x}\left(t_{f(N-1)}\right)\right)\right]^T \left[-\tilde{\nu}_{oN}\right] = \hat{\lambda}_x\left(t_{s(2N-3)}\right)$$

The definition $\tilde{\nu}_{f(N-1)} = -\frac{1}{\hat{\lambda}_y(t_N)}\tilde{\nu}_{oN}$ then implies that Eq. (3.30) for $i=N-1$ is satisfied. The construction for the last $u=0$ arc is complete.

Define

$$t_{N-1} = t_{s(2N-4)}$$

$$\tilde{\lambda}_{N-1}(t) = \frac{1}{\hat{\lambda}_y(t_N)}\hat{\lambda}(t),\ t \in \left[t_{N-1}, t_{f(N-1)}\right]$$

Note that this definition implies satisfaction of (3.29) for $i=N-1$ because $\hat{\lambda}_y(t_N) = \hat{\lambda}_y\left(t_{f(N-1)}\right)$. This also makes satisfaction of Eq. (3.30) for $i=N-1$ imply satisfaction of (3.28) for $i=N-1$. After establishing these constructions, the arguments for the previous $u=u_{max}$ and $u=0$ arc may be repeated. With each repeat, the construction is

made with scaling by an even earlier value from $\hat{\lambda}_y(t)$ in the following sequence $\hat{\lambda}_y(t_i), i = N,\ldots 2$. Such repetition may be continued until the beginning of the first burn is reached. At this point, the definition

$$\tilde{\nu}_{o1} = \frac{1}{\hat{\lambda}_y(t_2)}\hat{\nu}_o$$

implies satisfaction of (3.27) with i=1 and completes the proof of the "if" part of the theorem.

Assume that (3.33) satisfies $\{II\}$. The construction of the solution to $\{I\}$ will proceed backwards in time. Consider the last $u=u_{max}$ arc, where $t \in [t_N, t_{fN}]$. Define

$$\hat{\nu}_f = \tilde{\nu}_{fN}$$

$$t_{s2(N-1)} = t_N$$

$$\hat{\lambda}(t) = \tilde{\lambda}_N(t),\ t \in [t_N, t_{fN}]$$

For $t \in [t_N, t_{fN}]$ and i=N, this construction lets Eqs. (3.22)-(3.26) and (3.28) and (3.29) imply satisfaction of Eqs. (3.18) and (3.20) at the final point and Eqs. (3.14)-(3.17) during the interval. Now, it is obvious that satisfaction of Eqs. (3.14) and (3.27) with i=N in this interval under assumption (i) implies that Eq. (3.21) is satisfied for i=2(N-1); in other words $t_{s2(N-1)}$ is a switching point. This completes the construction for the last $u=u_{max}$ arc.

The definitions will now be extended into the interval $[t_{f(N-1)}, t_N]$. With Eqs. (3.31), the conditions $\{II\}$ specify that $u(t)$=0 for t in this interval. Define $t_{s(2N-3)}=t_{f(N-1)}$. This implies that Eqs. (3.31) are consistent with this switching structure of $\{I\}$ up to and including this interval. Now define

$$\hat{\lambda}_x(t) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t))\right]^T \left[-\tilde{\nu}_{oN}\right]$$

for all t in this interval. Knowing $u(t)=0$ and that Eqs. (3.31) are satisfied in this interval, Lemma III.2 with assumption (vi) and $\Gamma(\mathbf{x}(t)) = \frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t))$ implies satisfaction of Eq. (3.15) in this interval. Define

$$\hat{\lambda}_y(t) = \tilde{\lambda}_{yN}(t_N)$$

for all t in this interval. Knowing $u(t)=0$, this immediately implies satisfaction of Eq. (3.16) in the interval. Finally, since Eq. (3.14) was satisfied in the previous interval, Eqs. (3.15)-(3.16) are satisfied continuously from $t=t_f$ to any point in the current interval, and since the control switched values at a switching point, then Eq. (3.14) is satisfied in this interval. This completes the construction for the last $u=0$ arc.

Define $t_{s(2N-4)}=t_{N-1}$. Consider the interval $[t_{N-1},t_{f(N-1)}]$. Conditions $\{II\}$ specify that this is a $u=u_{max}$ interval which, by the definitions, is consistent with the switching structure of $\{I\}$. Define

$$\hat{\lambda}_x(t) = \tilde{\lambda}_{yN}(t_N)\tilde{\lambda}_{x(N-1)}(t)$$

$$\hat{\lambda}_y(t) = \tilde{\lambda}_{yN}(t_N)\tilde{\lambda}_{y(N-1)}(t)$$

in this interval. Equations (3.22) and (3.28) with $i=N-1$ imply that $t_{f(N-1)}$ is a switching point. Considering the definitions, Eq. (3.28) with $i=N-1$ and Eq. (3.30) with $i=N-2$ obviously imply continuity of the Lagrange multipliers $\hat{\lambda}_x(t)$ across the switching point $t_{f(N-1)}$; continuity of $\hat{\lambda}_y(t)$ across this point is immediately implied by the definition. Therefore, Eqs (3.15) and (3.16) are satisfied across the switching point.

The previous arguments for the final $u=u_{max}$ and $u=0$ arcs may be repeated, implying satisfaction of the conditions in $\{I\}$ for each interval. After repeating the arguments and reaching the beginning of the trajectory, the following definitions will have been made and are presented for the sake of clarity:

$$\hat{\lambda}_x(t)=\left[\prod_{j=i+1}^{N}\tilde{\lambda}_{yj}(t_j)\right]\left[\frac{\partial\psi}{\partial\mathbf{x}}(\mathbf{x}(t))\right]^T\left[-\tilde{\nu}_{oi}\right],\ t\in\left[t_{f(i-1)},t_i\right],\ i=2,\ldots N-1$$

$$\hat{\lambda}_y(t)=\left[\prod_{j=i+1}^{N}\tilde{\lambda}_{yj}(t_j)\right],\ t\in\left[t_{f(i-1)},t_i\right],\ i=2,\ldots N-1$$

$$\hat{\lambda}(t)=\left[\prod_{j=i+1}^{N}\tilde{\lambda}_{yj}(t_j)\right]\tilde{\lambda}_i(t),\ t\in\left[t_i,t_{fi}\right],\ i=1,\ldots N-1$$

Finally, for the first $u=u_{max}$ interval, one more definition is required. The definition

$$\hat{\nu}_o=\left[\prod_{i=2}^{N}\lambda_{yi}(t_i)\right]\tilde{\nu}_o$$

forces satisfaction of Eq. (3.27) with $i=1$ to imply satisfaction of Eq. (3.19). ■

The theorem does not assure satisfaction of Pontryagin's Minimum Principle. This principle requires that

$$\begin{aligned} &u(t)=0 \text{ when } \hat{\lambda}_x(t)^T\mathbf{g}(y(t),\mathbf{v}(t))+c\hat{\lambda}_y(t)>0\\ &u(t)=u_{max} \text{ when } \hat{\lambda}_x(t)^T\mathbf{g}(y(t),\mathbf{v}(t))+c\hat{\lambda}_y(t)<0 \end{aligned} \tag{3.37}$$

It should be noted that in the application of the Patched Method to the optimal orbit transfer problem, a second-order condition was taken into account. Lawden's pointer vector theory is a second-order condition and is explicitly specified. Also, note

that this condition was determined considering the maximization problem instead of the equivalent minimization problem.

To apply Theorem III.1 to the orbit transfer optimization problem, the assumptions of the theorem must be satisfied. Assumptions (i), (iii), and (vii) are obviously satisfied. There may still be debate over assumption (iv); however, based on numerical experience, orbit transfers that violate (iv) are rare if they exist at all.

Assumption (ii) is made in anticipation of the ideal gravity assumption. In such a case, coasting before the first burn contributes zero cost and coasting after the final burn contributes zero cost. It therefore makes no sense to allow such arcs as part of the trajectory to be calculated. If an initial and/or final coast arc is desired, it may be added to the computed trajectory without affecting optimality.

Rectilinear orbits will be explicitly excluded from candidate orbit transfer trajectories. Such orbits intersect the center of gravitation and are, therefore, rarely of interest for the orbit transfer problem. With this exclusion made, assumptions (v) and (vi) may now be shown true for the orbit transfer optimization problem.

It is desired that if $h = |\mathbf{r} \times \mathbf{v}| \neq 0$, then the vector function

$$\psi(\mathbf{x}) = \psi\left(\begin{bmatrix}\mathbf{r}\\ \mathbf{v}\end{bmatrix}\right) = \begin{bmatrix}\mathbf{I}_{5\times 5} & \mathbf{0}_{5\times 1}\end{bmatrix}\begin{bmatrix}\mathbf{r}\times\mathbf{v}\\ \mathbf{v}\times(\mathbf{r}\times\mathbf{v}) - \dfrac{\mu}{\sqrt{\mathbf{r}^T\mathbf{r}}}\mathbf{r}\end{bmatrix} \in C_S^0$$

yields $rank\left(\dfrac{\partial\psi(\mathbf{x})}{\partial\mathbf{x}}\right) = 5$. Note that this formulation for $\psi(\mathbf{x})$ calculates the angular momentum and eccentricity vectors, then removes the third component of the eccentricity vector. $\psi(\mathbf{x})$ as defined above yields

$$\frac{\partial\psi(\mathbf{x})}{\partial\mathbf{x}} = \begin{bmatrix} \mathbf{I}_{5\times5} & \mathbf{0}_{5\times1} \end{bmatrix} \begin{bmatrix} [-\mathbf{v}_\times] & [\mathbf{r}_\times] \\ \left[(\mathbf{v}^T\mathbf{v})\mathbf{I} - \mathbf{v}\mathbf{v}^T + \frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}\left(\mathbf{r}\mathbf{r}^T - (\mathbf{r}^T\mathbf{r})\mathbf{I}\right) \right] & \left[2\mathbf{r}\mathbf{v}^T - (\mathbf{r}^T\mathbf{v})\mathbf{I} - \mathbf{v}\mathbf{r}^T \right] \end{bmatrix}$$

where the subscript "X" denotes the skew symmetric matrix representation of the cross product. The result, $rank\left(\frac{\partial\psi(\mathbf{x}(t))}{\partial\mathbf{x}}\right) = 5$ is desired. The task is simplified by the following simple manipulation

$$\frac{\partial\psi(\mathbf{x}(t))}{\partial\mathbf{x}} = \begin{bmatrix} \mathbf{I}_{5\times5} & \mathbf{0}_{5\times1} \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{v}_X & \mathbf{I}_{3\times3} \end{bmatrix} \begin{bmatrix} [-\mathbf{v}_\times] & [\mathbf{r}_\times] \\ \left[\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}\mathbf{r}_X\mathbf{r}_X \right] & \left[\mathbf{r}\mathbf{v}^T - \mathbf{v}\mathbf{r}^T \right] \end{bmatrix}$$

which makes use of the identity $\mathbf{a}_X\mathbf{b}_X = \mathbf{b}\mathbf{a}^T - (\mathbf{a}^T\mathbf{b})\mathbf{I}$. This, in combination with

$$rank\left(\begin{bmatrix} \mathbf{I}_{5\times5} & \mathbf{0}_{5\times1} \end{bmatrix} \begin{bmatrix} \mathbf{I}_{3\times3} & \mathbf{0}_{3\times3} \\ \mathbf{v}_X & \mathbf{I}_{3\times3} \end{bmatrix} \right) = 5$$

implies

$$rank\left(\frac{\partial\psi(\mathbf{x}(t))}{\partial\mathbf{x}}\right) = min\left\{ 5 \, , \, rank\left(\begin{bmatrix} [-\mathbf{v}_\times] & [\mathbf{r}_\times] \\ \left[\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}\mathbf{r}_X\mathbf{r}_X \right] & \left[\mathbf{r}\mathbf{v}^T - \mathbf{v}\mathbf{r}^T \right] \end{bmatrix} \right) \right\}$$

It is most convenient to consider, without loss of generality, the following rotation of vectors $\mathbf{r}$ and $\mathbf{v}$ into the X-Y plane via an orthonormal matrix $\mathbf{W}$ defined such that

$$\mathbf{W}\mathbf{r} = \begin{bmatrix} x \\ y \\ 0 \end{bmatrix} \text{ and } \mathbf{W}\mathbf{v} = \begin{bmatrix} u \\ v \\ 0 \end{bmatrix}$$

It is easy to show that this rotation does not affect $rank\left(\frac{\partial\psi(\mathbf{x}(t))}{\partial\mathbf{x}}\right)$. Substitution reveals that, after rotation,

$$rank\left(\begin{bmatrix}[-\mathbf{v}_x] & [\mathbf{r}_x]\\ \left[\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}\mathbf{r}_x\mathbf{r}_x\right] & [\mathbf{r}\mathbf{v}^T-\mathbf{v}\mathbf{r}^T]\end{bmatrix}\right)=rank\left(\begin{bmatrix}0 & 0 & -v & 0 & 0 & y\\ 0 & 0 & u & 0 & 0 & -x\\ v & -u & 0 & y & x & 0\\ -gy^2 & gxy & 0 & 0 & -h & 0\\ gyx & -gx^2 & 0 & h & 0 & 0\\ 0 & 0 & 0 & 0 & 0 & 0\end{bmatrix}\right)$$

where $h=xv-yu$ and $g=\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}$. It can be shown that

$$det\left(\begin{bmatrix}0 & 0 & -v & 0 & y\\ 0 & 0 & u & 0 & -x\\ v & -u & 0 & x & 0\\ -gy^2 & gxy & 0 & -h & 0\\ gxy & -gx^2 & 0 & 0 & 0\end{bmatrix}\right)=-gxh^3$$

$$det\left(\begin{bmatrix}0 & 0 & -v & 0 & y\\ 0 & 0 & u & 0 & -x\\ v & -u & 0 & y & 0\\ -gy^2 & gxy & 0 & 0 & 0\\ gxy & -gx^2 & 0 & h & 0\end{bmatrix}\right)=gyh^3$$

so that as long as $h\neq0$, $\frac{\partial\psi(\mathbf{x}(t))}{\partial\mathbf{x}}$ has a nonzero minor of order 5. In other words, as long as the orbit is not rectilinear, $rank\left(\frac{\partial\psi(\mathbf{x}(t))}{\partial\mathbf{x}}\right)=5$.

Now, for assumption (vi) it must be shown that if the vector function $\mathbf{f}(\mathbf{x})$ is

$$\mathbf{f}\left(\begin{bmatrix}\mathbf{r}\\ \mathbf{v}\end{bmatrix}\right)=\begin{bmatrix}\mathbf{v}\\ -\dfrac{\mu}{\left(\mathbf{r}^{\mathrm{T}}\mathbf{r}\right)^{3/2}}\mathbf{r}\end{bmatrix}$$

and $\psi(\mathbf{x})$ is as already defined, then when $\dfrac{d}{dt}\mathbf{x}(t)=\mathbf{f}(\mathbf{x}(t))$,

$$\left(\frac{d}{dt}\frac{\partial}{\partial \mathbf{x}}\psi(\mathbf{x}(t))+\frac{\partial}{\partial \mathbf{x}}\psi(\mathbf{x}(t))\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t))\right)=\mathbf{0}$$

It is easy to show that

$$\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x})=\begin{bmatrix}[0] & [\mathrm{I}]\\ \left[-\dfrac{\mu}{\left(\mathbf{r}^{\mathrm{T}}\mathbf{r}\right)^{3/2}}\mathbf{I}+3\dfrac{\mu}{\left(\mathbf{r}^{\mathrm{T}}\mathbf{r}\right)^{5/2}}\mathbf{r}\mathbf{r}^{\mathrm{T}}\right] & [0]\end{bmatrix}$$

Note that the time notation has been dropped for convenience. Evaluating $\left[\dfrac{\partial\psi(\mathbf{x})}{\partial \mathbf{x}}\right]\left[\dfrac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x})\right]$ gives

$$\frac{\partial}{\partial \mathbf{x}}\psi(\mathbf{x})\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x})=\begin{bmatrix}\mathbf{I}_{5\times 5} & \mathbf{0}_{5\times 1}\end{bmatrix}\begin{bmatrix}\mathbf{M}_{11} & \mathbf{M}_{12}\\ \mathbf{M}_{21} & \mathbf{M}_{22}\end{bmatrix}\text{ where}$$

$$\mathbf{M}_{11}=-\mathbf{v}_{\times}$$

$$\mathbf{M}_{12}=-\frac{\mu}{\left(\mathbf{r}^{\mathrm{T}}\mathbf{r}\right)^{3/2}}\mathbf{r}_{\times}$$

$$\mathbf{M}_{21}=\left(\mathbf{v}^{\mathrm{T}}\mathbf{v}\right)\mathbf{I}-\mathbf{v}\mathbf{v}^{\mathrm{T}}+\frac{\mu}{\left(\mathbf{r}^{\mathrm{T}}\mathbf{r}\right)^{3/2}}\left(\mathbf{r}\mathbf{r}^{\mathrm{T}}-\left(\mathbf{r}^{\mathrm{T}}\mathbf{r}\right)\mathbf{I}\right)$$

$$\mathbf{M}_{22}=\frac{\mu}{\left(\mathbf{r}^{\mathrm{T}}\mathbf{r}\right)^{3/2}}\left\{-\left(\mathbf{r}^{\mathrm{T}}\mathbf{v}\right)+2\left(\mathbf{r}\mathbf{v}^{\mathrm{T}}\right)+2\left(\mathbf{v}\mathbf{r}^{\mathrm{T}}\right)-\frac{3\left(\mathbf{r}^{\mathrm{T}}\mathbf{v}\right)}{\left(\mathbf{r}^{\mathrm{T}}\mathbf{r}\right)}\left(\mathbf{r}\mathbf{r}^{\mathrm{T}}\right)\right\}$$

Next, the time derivative of each term in $\dfrac{\partial}{\partial \mathbf{x}}\psi(\mathbf{x})$ can be expressed as:

$$\frac{d}{dt}(\mathbf{v}_*) = -\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{1/2}}\mathbf{r}_*$$

$$\frac{d}{dt}(\mathbf{r}_*) = \mathbf{v}_*$$

$$\frac{d}{dt}(\mathbf{v}^T\mathbf{v}) = -2\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}\mathbf{r}\mathbf{v}^T$$

$$\frac{d}{dt}(\mathbf{v}\mathbf{v}^T) = -\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}(\mathbf{v}\mathbf{r}^T + \mathbf{r}\mathbf{v}^T)$$

$$\frac{d}{dt}\left(\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}\mathbf{r}\mathbf{r}^T\right) = \frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}(\mathbf{r}\mathbf{v}^T + \mathbf{v}\mathbf{r}^T) - \frac{3\mu}{(\mathbf{r}^T\mathbf{r})^{5/2}}(\mathbf{r}^T\mathbf{v})\mathbf{r}\mathbf{r}^T$$

$$\frac{d}{dt}\left(\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{1/2}}\right) = -\frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}(\mathbf{r}^T\mathbf{v})$$

$$\frac{d}{dt}(\mathbf{r}\mathbf{v}^T) = \mathbf{v}\mathbf{v}^T - \frac{\mu}{(\mathbf{r}^T\mathbf{r})^{3/2}}(\mathbf{r}\mathbf{r}^T)$$

$$\frac{d}{dt}(\mathbf{v}\mathbf{r}^T) = \frac{d}{dt}(\mathbf{r}\mathbf{v}^T)$$

$$\frac{d}{dt}(\mathbf{r}^T\mathbf{v}) = \mathbf{v}^T\mathbf{v} - \frac{\mu}{(\mathbf{r}^T\mathbf{r})^{1/2}}$$

With these expressions it can easily be shown that

$$\frac{d}{dt}\frac{\partial}{\partial \mathbf{x}}\psi(\mathbf{x}(t)) + \frac{\partial}{\partial \mathbf{x}}\psi(\mathbf{x}(t))\frac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t)) = \mathbf{0}$$

This is more than just satisfaction of a simple condition that proves useful to the theorem. In fact, this shows that Eq. (2.12) is the solution of the ODEs for the Lagrange multipliers, Eqs. (2.3a-c), when the Hamiltonian vanishes and ideal gravity is assumed. As reviewed earlier, many previous research efforts have focused on obtaining such solutions, but the form found herein is different from those.

III.2.5. Solution using the Patched Method with Eleven Burns

The plots below represent the current capability of the Patched Method. The eleven-burn solution represented by these plots has a larger number of burns than

obtained BOUNDSCO or MBCM, in this study. Few solutions, if any, with this number of burns have been obtained in the literature. However, the Modified Patched Method, introduced in the next subsection, has produced solution with even larger numbers of burns.

Also indicative of the Patched Method, the convergence tolerance for the outer loop was set relatively high, 10^{-3}, to prevent prohibitively long computation times.

For this example, the thrust level is 0.09698, the product $g_o I_{sp}$ is 0.3929, the initial mass is 10. The initial orbit is circular with a radius of 1; the final orbit has an eccentricity of 0.398 and a final semimajor axis of 1.708. With this information the value of T/W_o for this transfer is calculated to be 0.009698, placing it in the low-thrust transfer range.

Figure 3.1 is a plot the transfer orbit elements, *viz.* angular momentum, eccentricity vector x-component, and eccentricity vector y-component, versus transfer orbit number. The shape of the angular momentum and eccentricity x-component curves seem to indicate a second order polynomial fit could be used to reduce the number of variables in the problem. The eccentricity y-component is always small in this transfer; suggesting that it could be assumed zero or, more generally, the same parameterization may be used. The zeroth orbit is the fixed initial orbit and the eleventh orbit is the fixed final orbit.

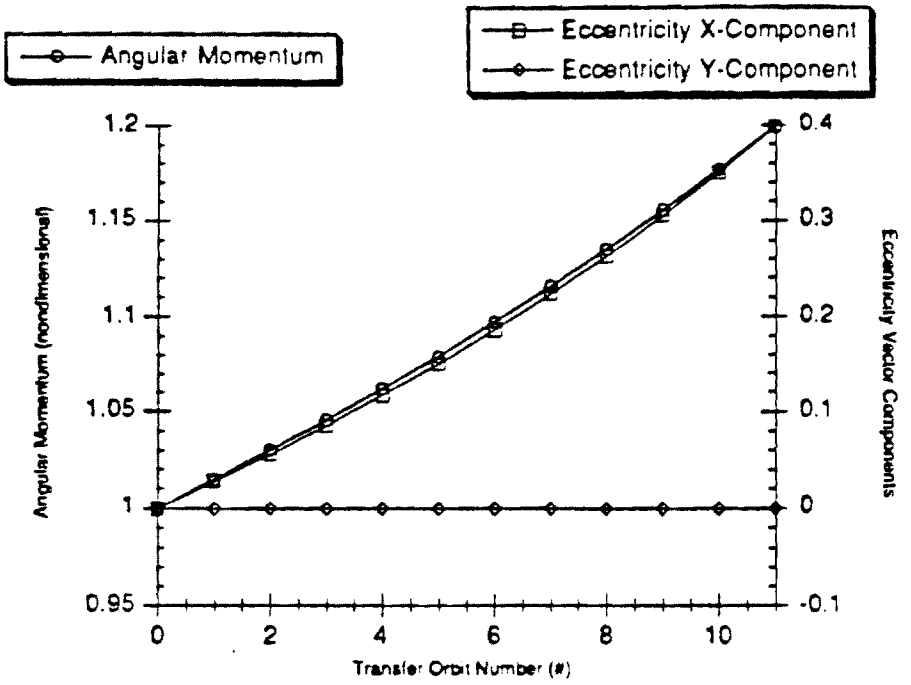


Figure 3.1 Orbital Elements of Each Transfer Orbit of Eleven Burn Solution

Figure 3.2 shows the angular position of the initial orbit exit point and final orbit entry point of each versus the index enumerating which transfer orbit the burn ends at. The symmetry of this plot is somewhat surprising. Even though each transfer orbit has its apse roughly aligned with the x-axis, each pair of angular positions are not reflected about the x-axis. The trend over time is almost exactly opposite between the two positions, but note that the values are not quite the negatives of each other. Also, it is clear that each burn of this transfer are perigee burns; each occurring around perigee.

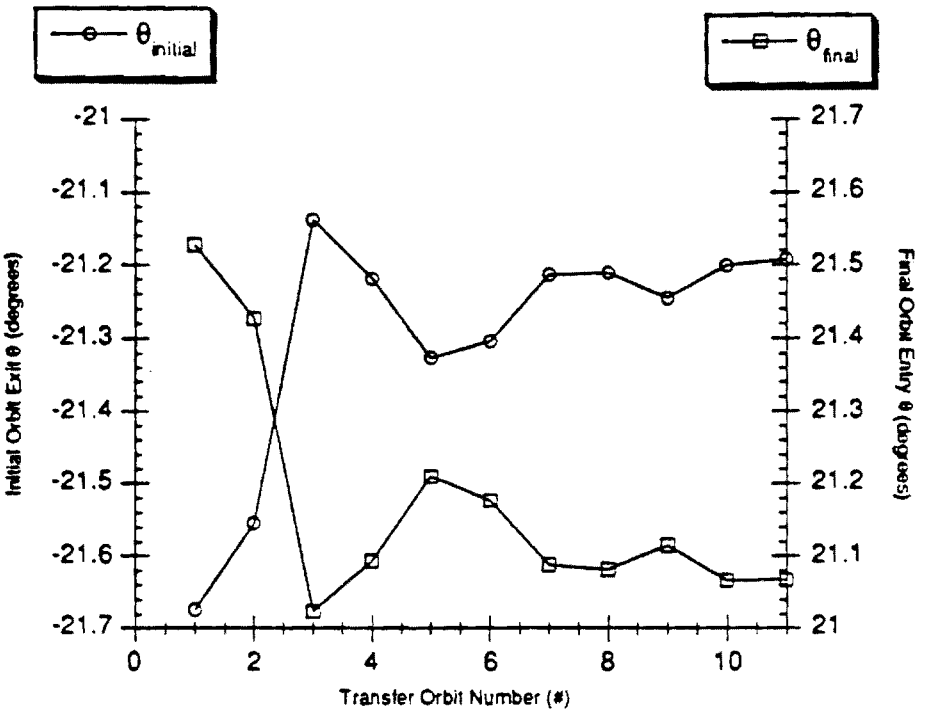


Figure 3.2 Orbit Transfer Terminal Points Indexed by Ending Orbit

Another interesting trend is found in Fig. 3.3, showing the burn length versus the same index as before. The burn length decreases monotonically with each successive burn, but does not decrease linearly. One can, of course, observe a relationship in the trend of burn length and angular positions from Figure 3.2. Both plots have a sharp change at the third burn which holds till the fourth burn and then returns to follow the trend from the first two. The irregular trend for this burn is attributed to the high tolerance given for the convergence criteria.

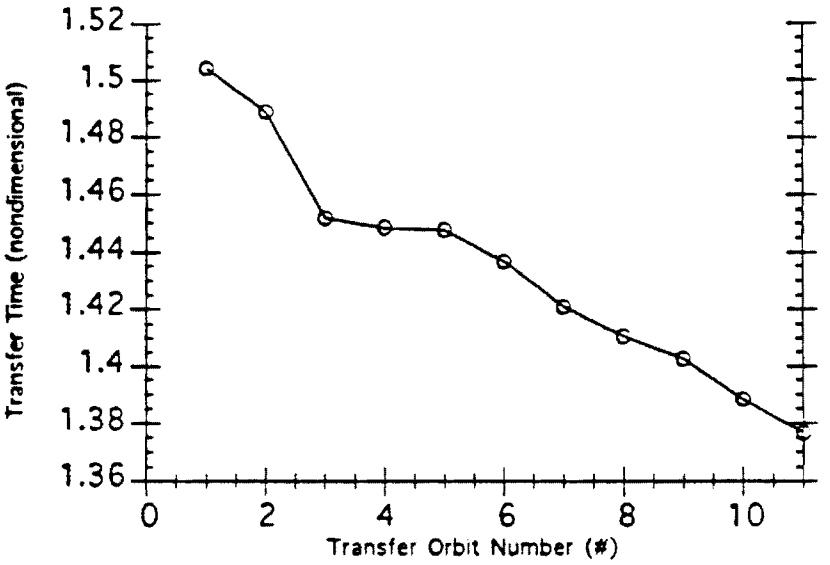


Figure 3.3 Transfer Time Indexed by Ending Orbit for the Eleven Burn Solution

III.3. The Modified Patched Method (MPM)

The Relaxed Patched Method is tailored to the orbit transfer optimization problem through known relations concerning the behavior of states and costates at different points along the trajectory. The concept central to these relations is that each burn of a multiple-burn orbit transfer qualifies as an optimal transfer between its own local terminal orbits. This method uses an algorithm similar to shooting methods.

This method puts forth an algorithm for computing problem constraints given the values of the problem variables. The number of variables and constraints are equal. Also, the method can be used with any multi-dimensional root-finding algorithm. The discussion below describes the variables and computation of the constraints for a two-burn trajectory.

In the following description of the variables and constraints, the vector $\lambda = \left[\lambda_r^T \quad \lambda_v^T\right]^T$ is used instead of the more common $\left[\lambda_r^T \quad \lambda_v^T \quad \lambda_m\right]^T$ so that λ_m can be discussed separately.

The arc between points #1 and #2 is assumed to be an arc of maximum thrust. Referring to Fig. 3.4, the variables at #1 are the initial true anomaly, θ_1; the first burn length, t_{f1}; and, the vector of constant Lagrange multipliers for the start of the first burn, ν_1. The only constraint associated with point #1 is for ν_1 to have unity magnitude.

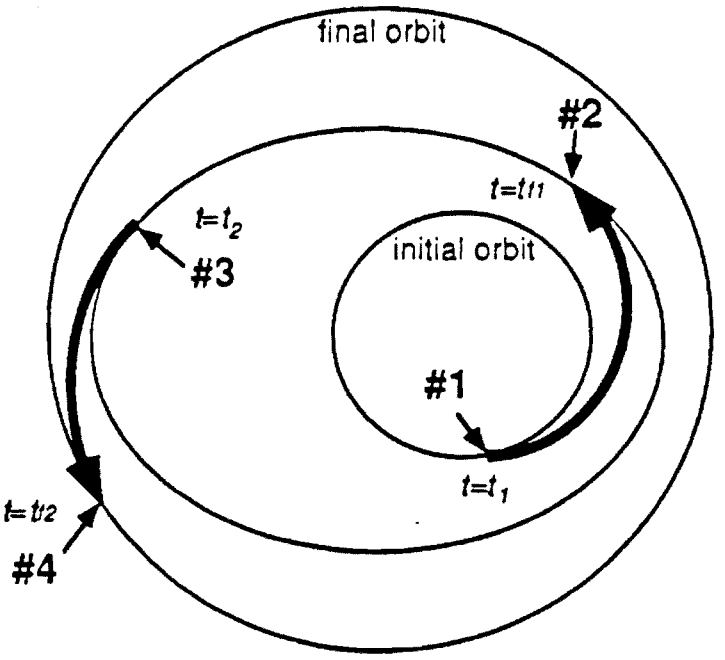


Figure 3.4 Diagram Illustrating the Layout of a Two-Burn Transfer

Knowing the true anomaly, θ, and the rest of the orbital elements, α, state, $\mathbf{x}(t)$ may be calculated with the function $\tilde{\mathbf{x}}(\theta;\alpha)$. Therefore, the Lagrange multipliers, $\lambda(t_1)$, and the state, $\mathbf{x}(t_1)$, at the initial orbit exit point may be computed using

$$\mathbf{x}(t_1) = \tilde{\mathbf{x}}(\theta_1;\alpha_0) \tag{3.38}$$

$$\lambda(t_1) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_1))\right]^T \nu_1 \tag{3.39}$$

Where $\psi(\mathbf{x})$ is a function that calculates the orbital elements α given the state $\mathbf{x}$. The Lagrange multipliers, $\lambda(t_{f1})$, and final state of the first burn, $\mathbf{x}(t_{f1})$, are calculated by numerical integration of the Euler-Lagrange and state differential equations.

The vector variables α_1 and ν_2 are associated with point #2. These are used to evaluate the constraints at point #2 as

$$\psi(\mathbf{x}(t_{f1})) = \alpha_1 \tag{3.40}$$

$$\lambda(t_{f1}) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_{f1}))\right]^T \nu_2 \tag{3.41}$$

The trajectory between points #2 and #3 is assumed to be an arc of null thrust. The variables θ_2, the initial true anomaly for the second burn, and t_{f2}, the second burn length, are associated with point #3. With these values, the Lagrange multipliers and the state may be calculated, much as before, with

$$\mathbf{x}(t_2) = \bar{\mathbf{x}}(\theta_2; \alpha_1) \tag{3.42}$$

$$\lambda(t_2) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_2))\right]^T \nu_2 \tag{3.43}$$

Using the integration results from the first burn and Eq. (3.43), the following constraint is evaluated at point #3

$$\left|\lambda_v(t_{f1})\right| = \left|\lambda_v(t_2)\right| \tag{3.44}$$

The arc between points #3 and #4 is assumed to be of maximum thrust. The variables θ_2, α_1, and ν_2, specified at points #2 and #3 enable the calculation of the Lagrange multipliers, $\lambda(t_{f2})$, and final state, $\mathbf{x}(t_{f2})$, in the same manner as the previous burn - numerically integrating from t_2 to t_{f2} with the initial conditions Eqs. (3.42) and (3.43).

The two-burn trajectory ends at point #4. The constant Lagrange multiplier vector ν_3 is associated with this point. The constraints evaluated at point #4 are

$$\psi\left(\mathbf{x}\left(t_{f2}\right)\right)=\alpha_2 \tag{3.45}$$

$$\lambda\left(t_{f2}\right)=\left[\frac{\partial\psi}{\partial\mathbf{x}}\left(\mathbf{x}\left(t_{f2}\right)\right)\right]^{\mathrm{T}}\nu_3 \tag{3.46}$$

These constraints complete the system.

With the discussion of the formulation for a two-burn trajectory concluded, the formulation for a more general problem is clear. For an N-burn trajectory with α_0, α_N, m_o, T, g_o, and I_{sp} specified, the variables are

$$\left\{\alpha_i|i=1,\ldots N-1\right\},\left\{\theta_i,t_{fi}|i=1,\ldots N\right\},\left\{\nu_i|i=1,\ldots N+1\right\} \tag{3.47}$$

By use of which, the following quantities are calculated

$$\mathbf{x}(t_i)=\bar{\mathbf{x}}(\theta_i;\alpha_{i-1})\ ;\ i=1\ldots N \tag{3.48}$$

$$\lambda(t_i)=\left[\frac{\partial\psi}{\partial\mathbf{x}}\left(\mathbf{x}(t_i)\right)\right]^{\mathrm{T}}\nu_i;\ i=1,\ldots N \tag{3.49}$$

$$\left\{\begin{array}{l} \mathbf{x}(t_{fi}) = \mathbf{x}(t_i) + \int_{t_i}^{t_{fi}} \mathbf{f}(\mathbf{x}(t)) + \dfrac{T}{m(t)}\mathbf{v}(t)dt \qquad (3.50) \\ \lambda(t_{fi}) = \lambda(t_i) + \int_{t_i}^{t_{fi}} \left[-\dfrac{\partial}{\partial \mathbf{x}}\mathbf{f}(\mathbf{x}(t))\right]^{\mathrm{T}} \lambda(t)dt \qquad (3.51) \\ \text{where } \mathbf{v}(t) = \dfrac{\lambda_v(t)}{|\lambda_v(t)|} \qquad (3.52) \\ \text{and } m(t) = m_o - \dfrac{T}{g_o I_{sp}}\left(t - t_{f1} + \sum_{j=1}^{i}(t_{fj} - t_j)\right),\ t \in [t_i, t_{fi}] \qquad (3.53) \end{array}\right\} i = 1,\ldots N$$

The constraints that must be then evaluated and satisfied are

$$|\nu_1| = 1 \qquad (3.54)$$

$$\psi(\mathbf{x}(t_{fi})) = \alpha_i;\ i = 1,\ldots N \qquad (3.55)$$

$$\lambda(t_{fi}) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_{fi}))\right]^{\mathrm{T}} \nu_{i+1};\ i = 1,\ldots N \qquad (3.56)$$

$$|\lambda_v(t_{fi})| = |\lambda_v(t_{i+1})|\ ;\ i = 1,\ldots N-1 \qquad (3.57)$$

This gives a total of $2N(M+1)$ variables and the same number of constraints, where M is the number of orbital elements. For nonplanar transfers M=5 but for planar transfers, it is more efficient to rotate the coordinate system so that M=3.

In summary, the Modified Patched Method executes the following procedure for the ith burn, i=1...N, of an N-burn transfer. Given the current iterates θ_i, α_{i-1}, and ν_i, (note, however, that α_o is not an iterate but a specified constant) calculate $\mathbf{x}(t_i)$ and $\lambda(t_i)$ with Eqs (3.48)-(3.49). If i=1, evaluate the scaling constraint, Eq. (3.54). Given t_{fi}, and the calculated initial values $\mathbf{x}(t_i)$, $\lambda(t_i)$, compute $\mathbf{x}(t_{fi})$, $\lambda(t_{fi})$ with Eqs (3.50)-(3.53). Evaluate the burn terminal point constraints, Eqs (3.55)-(3.56). If $i<N$, evaluate the switching function constraint, Eq. (3.57), where $\lambda_v(t_{i+1})$ is calculated with (3.49) knowing ν_{i+1}.

When implementing MPM on a computer, the angular variable θ_i should be replaced by the variables l_{1i}, l_{2i} and the constraint $l_{1i}^2 + l_{2i}^2 = 1$. This common substitution removes the periodic redundancy that may confuse a numerical method.

Completion of the iterative process updating the variables in (3.47) to satisfy the conditions in Eqs. (3.54)-(3.56) allows the final condition of the Modified Patched Method to be checked. Briefly, this checks the switching law:

$$\begin{aligned} \frac{|\lambda_v(t)|}{m(t)} - \frac{\lambda_m(t)}{g_o I_{sp}} > 0 \;,\; T = T_{max} \\ \frac{|\lambda_v(t)|}{m(t)} - \frac{\lambda_m(t)}{g_o I_{sp}} < 0 \;,\; T = 0 \end{aligned} \tag{3.58}$$

This condition is, in fact, borrowed directly from the application of Pontryagin's Maximum Principle. When all conditions are satisfied, it may be claimed that an extremal solution has been obtained.

The relationship between the Patched Method and MPM is primarily in the use of Eqs. (3.49) and (3.56), which perform basically the same function as Eqs. (3.27), (3.28), and (3.30) from the Patched Method. However, MPM also includes a technique apparently first employed by Brown, et. al.[21] which removes one Lagrange multiplier (λ_m) and significantly affects the way the switching conditions are handled. This technique is present here as the use of Equation (3.57).

III.3.1. Equivalency of MPM Conditions and Necessary Conditions

This subsection is concerned with proving the equivalency between necessary conditions and the Modified Patched Method conditions. From the standpoint of showing mathematical equivalence, some combinations of variables and constraints in MPM are

unnecessary. Essentially, guessing intermediate orbital elements can be replaced by requiring the state to be continuous between burns.

Definition: For optimal control problem (P), the conditions (III) are

$$|\nu_1| > 0 \tag{3.59}$$

$$\lambda(t_{f_i}) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_{f_i}))\right]^T \nu_{i+1};\ i = 1,\ldots N \tag{3.60}$$

$$\lambda(t_i) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t_i))\right]^T \nu_i;\ i = 1,\ldots N \tag{3.61}$$

$$\lambda(t_{f_i})^T \mathbf{g}(y(t_{f_i}), \mathbf{v}(t_{f_i})) = \lambda(t_{i+1})^T \mathbf{g}(y(t_{i+1}), \mathbf{v}(t_{i+1}))\ ;\ i = 1,\ldots N-1 \tag{3.62}$$

$$\left\{\begin{array}{c} \mathbf{x}(t_{f_i}) = \mathbf{x}(t_i) + \int_{t_i}^{t_{f_i}} [\mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(y(t), \mathbf{v}(t)) u_{max}]\, dt \\ \lambda(t_{f_i}) = \lambda(t_i) + \int_{t_i}^{t_{f_i}} \left[-\frac{\partial}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}(t))\right]^T \lambda(t)\, dt \\ \text{where } \lambda(t)^T \left[\frac{\partial}{\partial \mathbf{v}} \mathbf{g}(y(t), \mathbf{v}(t))\right] = 0 \\ \text{and } y(t) = y_o + c u_{max}\left(t - t_{f1} + \sum_{j=1}^{i} (t_{fj} - t_j)\right),\ t \in [t_i, t_{f_i}] \end{array}\right\} i = 1,\ldots N \tag{3.63}$$

$$\left\{\begin{array}{c} \mathbf{x}(t_{i+1}) = \mathbf{x}(t_{f_i}) + \int_{t_{f_i}}^{t_{i+1}} \mathbf{f}(\mathbf{x}(t))\, dt \\ y(t) = y(t_{i+1}) = y(t_{f_i}) \\ u(t) = 0,\ \ t \in [t_{f_i}, t_{i+1}] \end{array}\right\}\ i = 1,\ldots N-1 \tag{3.64}$$

$$\left|\frac{\lambda(t_{f_N})^T \mathbf{g}(y(t_{f_N}), \mathbf{v}(t_{f_N}))}{c}\right| > 0 \tag{3.65}$$

where $t_1 = t_o$ is assigned and the value for t_f is seen to be t_{fN}.

Theorem III.2: If and only if

$$\left\{\mathbf{x}(t), y(t), \mathbf{v}(t), u(t), \hat{\lambda}(t) \middle| t \in \left[t_o, t_f\right]\right\}, \hat{\mathbf{v}}_o, \hat{\mathbf{v}}_f, t_f, \left\{t_{s_i} \middle| i = 1, \ldots 2(N-1)\right\} \tag{3.66}$$

satisfies $\{I\}$ then

$$\left\{\mathbf{x}(t), y(t), \mathbf{v}(t), u(t), \lambda(t) \middle| t \in \left[t_1, t_{fN}\right]\right\}, \left\{t_i, t_{fi} \middle| i = 1, \ldots N\right\}, \left\{\mathbf{v}_i \middle| i = 1, \ldots N+1\right\} \tag{3.67}$$

satisfies $\{III\}$, assuming that the constraints and assumptions from $\{P\}$ are satisfied.

Proof:

Both sufficiency and necessity will be proven by assuming satisfaction of one set of conditions and then constructing the solution to the other. From here on, assume that the constraints and assumptions from $\{P\}$ are satisfied. The "if" part will be proven after the "only if" part. To prove the "only if" part, it will be useful to follow time in reverse from $t=t_f$ to the initial time, $t=t_o$.

Assume that (3.67) satisfies $\{III\}$. Define a scaling factor $\gamma \in R$,

$$\gamma = \frac{-c}{\lambda\left(t_{fN}\right)^{\dagger} \mathbf{g}\left(y\left(t_{fN}\right), \mathbf{v}\left(t_{fN}\right)\right)} \tag{3.68}$$

Equation (3.65) ensures that the γ exists as a finite real number. Define $\hat{\mathbf{v}}_f = \gamma \mathbf{v}_{N+1}$, $\hat{\lambda}_x\left(t_f\right) = \gamma \lambda\left(t_{fN}\right)$, and recall that $t_f = t_{fN}$. Note that this construction makes satisfaction of (3.60) with $i=N$ imply satisfaction of (3.18). Now, define $\hat{\lambda}_y\left(t_f\right) = 1$ which satisfies (3.20); this makes the switching function in the form of Eq. (3.21) vanish for $t=t_f$.

It is obvious that when assumption (i) holds, Eq. (3.18) is satisfied, and Eq. (3.21) vanishes for $t=t_f$ then Eq. (3.14) is satisfied at $t=t_f$. Now, extend the construction so that $\hat{\lambda}_x(t) = \gamma \lambda(t)$, $t \in \left[t_N, t_{fN}\right]$ and Eq. (3.16) is satisfied. Note that this and Eqs. (3.63) imply

that all Euler-Lagrange differential equations, Eqs. (3.15)-(3.17), are satisfied in this interval. Therefore, the Hamiltonian is constant in the interval and is hence equal to zero at $t=t_N$. Now, with the Hamiltonian zero, assumption (i) and Eq. (3.61) with $i=N$ implies that the switching function vanishes again at $t=t_N$. Define $t_{s2(N-1)}=t_N$. Since by (3.64) and (3.63), the bang-bang control, $u(t)$, switches from u_{max} to zero at $t=t_N$, the Hamiltonian will be continuous across this switching point and, therefore, zero.

Lemma III.2 with $\Gamma(\mathbf{x}(t))=\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t))$ for $t \in \left[t_{fN-1}, t_N\right]$, Eq. (3.64), and assumption (vi) implies satisfaction of Eqs. (3.15) and (3.17) in this interval. Extend the construction so that $\hat{\lambda}_y(t)=\hat{\lambda}_y(t_N)=\hat{\lambda}_y\left(t_{fN-1}\right)$ in the interval, thereby satisfying Eq. (3.16). Having this construction, knowing that the switching function vanishes at $t=t_N$, that $u(t)=0$ is assigned in this interval by (3.64), satisfaction of Eq. (3.62) implies that the switching function vanishes at $t=t_{fN-1}$. In order to imply satisfaction of Eq. (3.14) at the end of this interval, it must be recognized that again, the bang-bang control switches values at $t=t_{fN-1}$. Define $t_{s(2N-1)}=t_{fN-1}$.

The arguments in the preceding two paragraphs may be repeated until the initial time, t_o is reached. Recall that $t_1=t_o$. Define $\hat{\nu}_o=-\gamma \nu_1$ and recall that previous definitions require $\hat{\lambda}_x(t_o)=\gamma \hat{\lambda}_x(t_1)$; these definitions imply satisfaction of (3.19). The proof of the "only if" part is complete.

For the "if" part of the theorem, assume that (3.66) satisfies $\{I\}$. Define $\lambda(t)=\hat{\lambda}_x(t)$, $t \in \left[t_o, t_f\right]$ and recall that $t_f=t_{fN}$ and $t_1=t_o$. Define $\nu_1=-\hat{\nu}_o$ and $\nu_i=\hat{\nu}_f$. Given assumption (i), it is immediately obvious that all conditions in $\{III\}$ except Eqs. (3.59), (3.62), (3.65), (3.61) with $i\neq 1$, and (3.60) with $i\neq N$. Note that (3.61) and (3.60) each apply at a switching point and when $u=u_{max}$. Furthermore, Eq. (3.14) specifies that the Hamiltonian is zero throughout the trajectory. Therefore, by Lemma III.1, Eqs (3.14),

(3.21), and assumptions (v) and (vii) there exists a different value ν for each switching point such that Eqs. (3.60) and (3.61) hold; however, Lemma III.1 does not guarantee that the value of ν at one end of the kth u=0 arc (i=k-1 in (3.60)) equals the value of ν at the other end (i=k in (3.61)). But, Lemma III.2 with $\Gamma(\mathbf{x}(t)) = \frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t))$ and assumption (vi) implies that $\hat{\lambda}_x(t) = \left[\frac{\partial \psi}{\partial \mathbf{x}}(\mathbf{x}(t))\right]^T \nu$ solves (3.15) when u=0. Therefore, the value of ν at one end of a u=0 arc must equal the value of ν at the other end of the u=0 arc.

Eq. (3.65) is implied by the switching function vanishing at $t=t_f$. Finally, it is obvious that the boundary value problem cannot be solved if $\hat{\lambda}_x(t) = 0$; therefore $|\hat{\nu}_o| > 0$, by assumption (v). That implies satisfaction of Eq (3.59). ■

III.3.2. MPM Example Solutions

The following examples satisfy all the conditions implied by the Euler-Lagrange equations and the Pontryagin Maximum Principle. All quantities have been nondimensionalized.

The first example solution is a 5-burn transfer reproducing a solution presented in a paper by Redding. Both the initial orbit and the final orbit are circular. However, there is an inclination of 28.5° between them. In this presentation of the solution, the initial orbit is equatorial and the final orbit is inclined 28.5°. The initial orbit radius is 1, the final orbit radius is 6.4. The initial nondimensional acceleration is 0.0517 and the nondimensional characteristic velocity is 0.567. Both the transfer computed by Redding and this solution calculated with the Modified Patched Method have final transfer orbits with e=0.723 and an inclination 26.5° away from that of the final orbit. Perigee burn durations for both range from 1.26 to 1.13. Both have a total transfer time of 60. Finally, it is worth noting that the solution presented here was computed without knowing the particulars of Redding's solution.

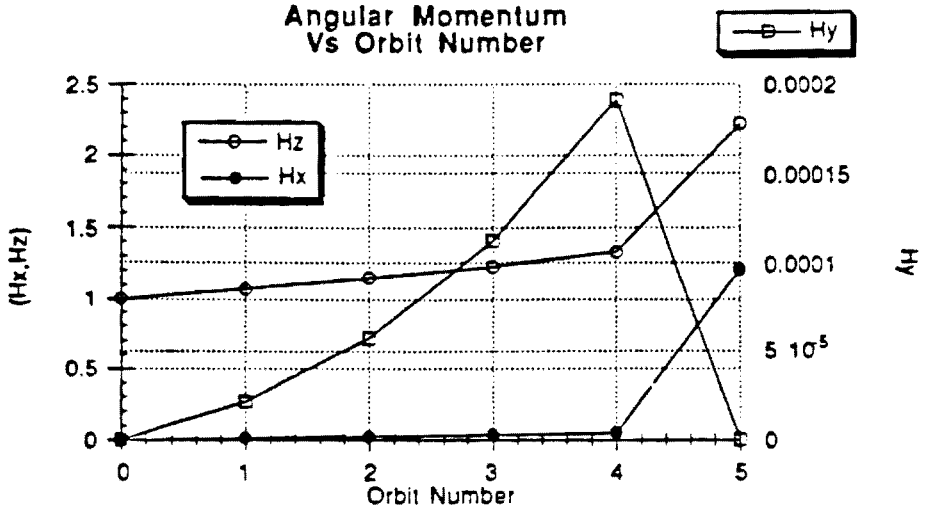


Figure 3.5 Components of the Angular Momentum Vector for each Transfer Orbit vs. Orbit Number of a 5-Burn Transfer with Plane Changes

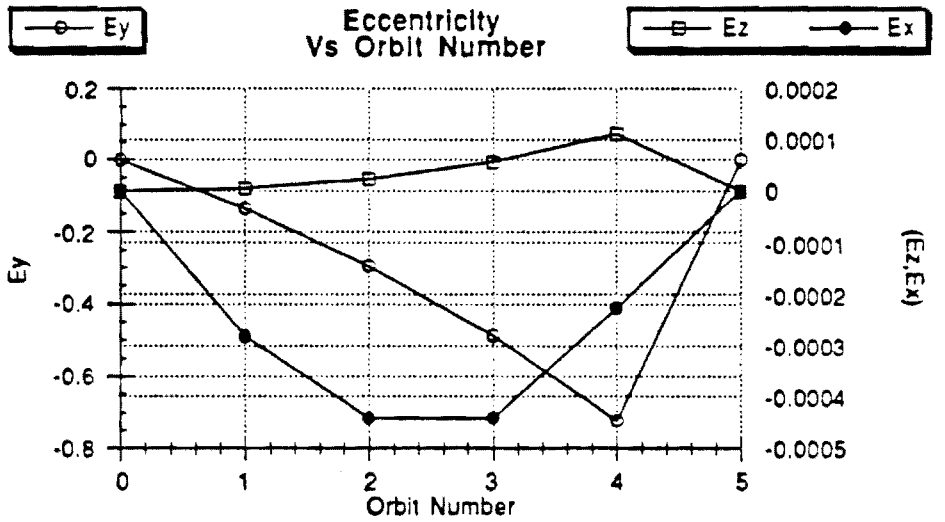


Figure 3.6 Components of the Eccentricity Vector Vs Orbit Number for each Transfer Orbit of a 5-Burn Transfer with Plane Changes

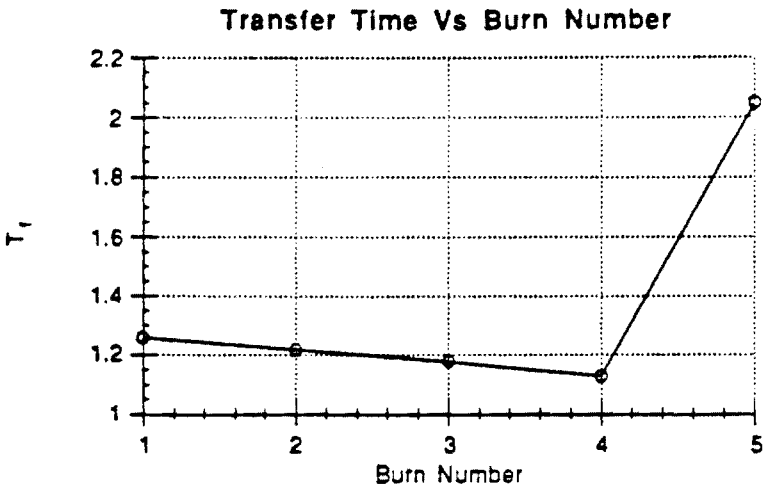


Figure 3.7 Transfer Time Vs Orbit Number for each Burn of a 5-Burn Transfer with Plane Changes

The second example is a 19-burn transfer. The initial nondimensional acceleration produced by the rocket motor (T/m_o) is 0.09698 and the initial nondimensional characteristic velocity ($g_o I_{sp}$) is 0.3929. The initial orbit is circular with a radius of 1, the final orbit has eccentricity of 0.73315 and a semimajor axis of 9.26. The total burn time for this trajectory is 26.84. Figures 3.8 — 3.9 show data in similar form for this transfer as Figures 3.5-3.7 for the previous transfer.

This 19-burn trajectory was extended to a 27-burn trajectory. This process involved the determination of transfers with 20, 22, 23, 24 burns, etc. It was found that adding burns one at a time was usually successful, two at a time slightly less successful, and so on. It was also interesting to see the decreasing improvement of the transfer's performance as plotted in Figure 3.10.

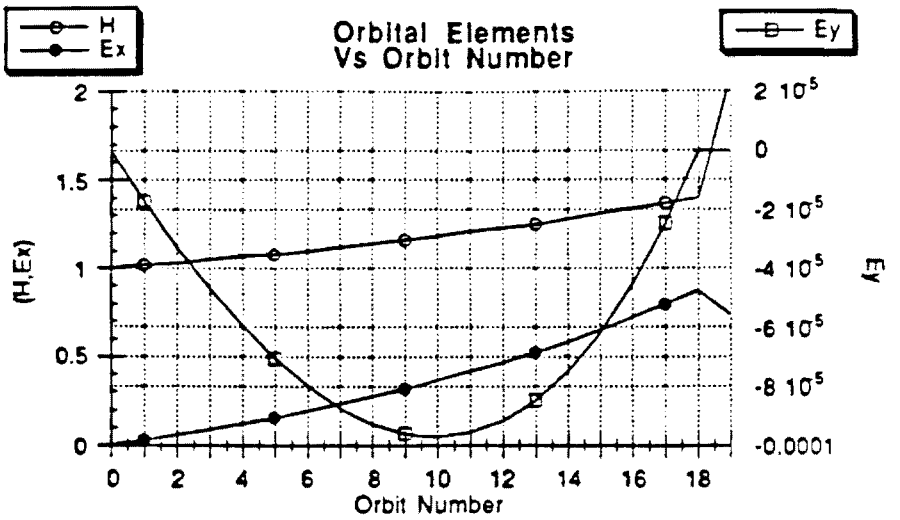


Figure 3.8 Orbital Elements for each Transfer Orbit Vs Orbit Number of a 19-Burn Transfer

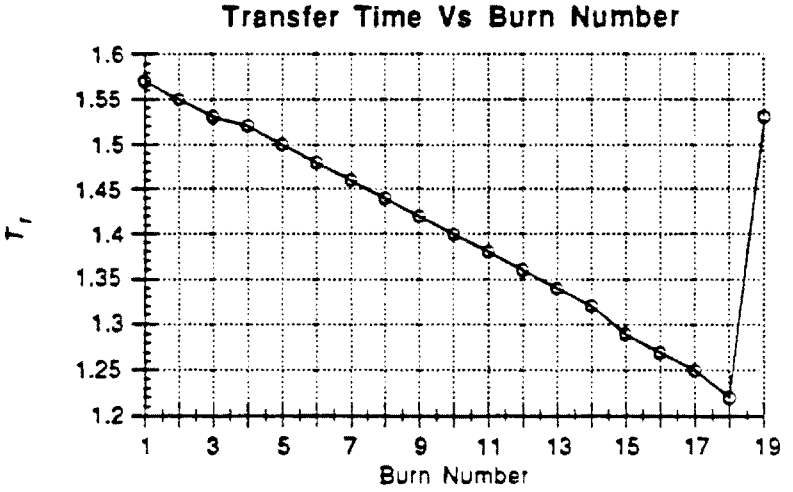


Figure 3.9 Transfer Time vs Orbit Number for each Burn of a 19-Burn Transfer

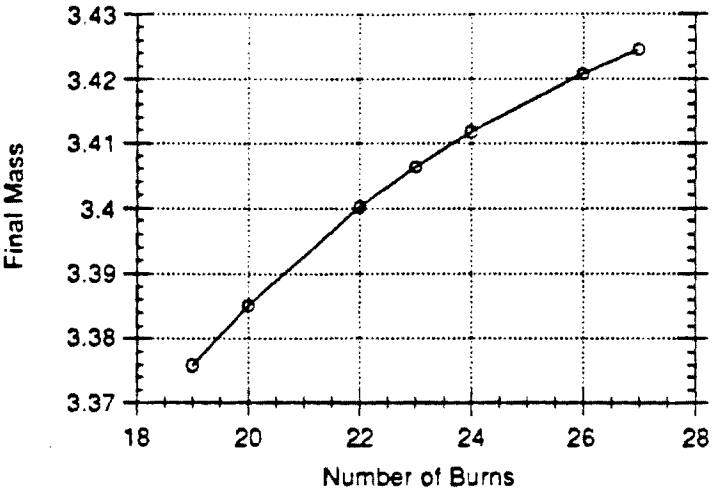


Figure 3.10 Performance Index, Final Mass, of the Extremal Trajectory vs Number of Burns Executed during the Trajectory

The third example is the aforementioned 27-burn trajectory. All parameters are identical between this transfer and the previous except the number of burns. The total burn time for this trajectory is 26.64. This is only a 0.7% decrease in transfer time for 42% more burns.

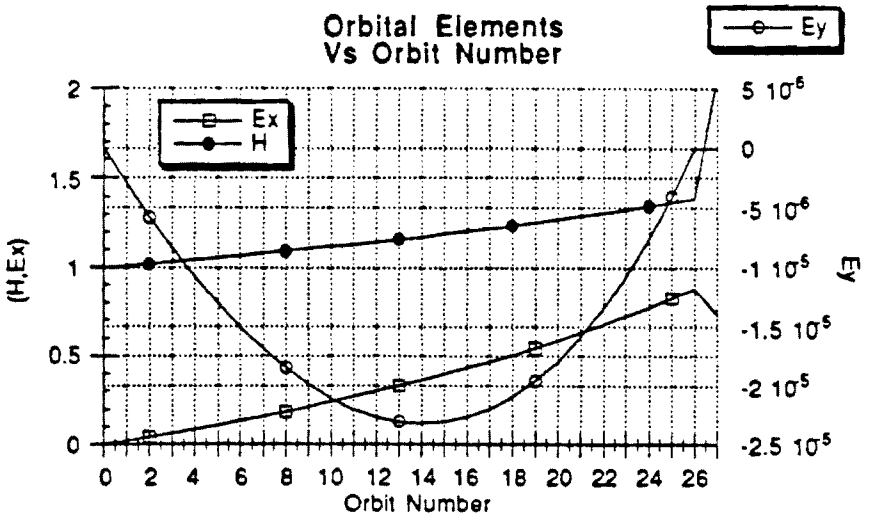


Figure 3.11 Orbital Elements for each Transfer Orbit Vs Orbit Number of a 27 Burn-Transfer

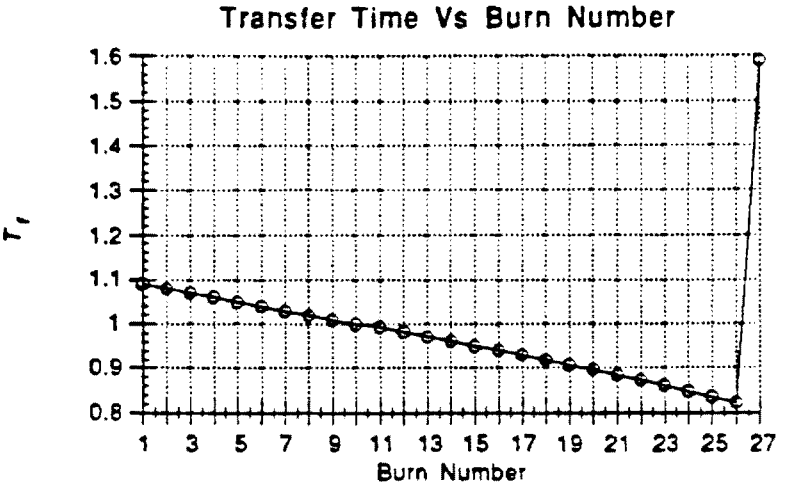


Figure 3.12 Transfer Time vs Orbit Number for each Burn of a 27-Burn Transfer

The fourth transfer is identical to the third except that the final orbit has an inclination of 63.4° This inclination angle was chosen because it is large and represents the inclination of the useful Molniya class of orbits. To obtain the solution, the planar transfer was used as the initial guess and the Modified Patched Method obtained the solution in 6 iterations. The following figures represent the transfer.

Each of these transfers show similar trends. An almost linear variation in the largest components of the angular momentum and eccentricity vectors and for the transfer time when plotted against the orbit or burn number. However, this trend is broken for the last burn. In each transfer, the last burn is an apogee burn and all previous burns are perigee burns. Each perigee burn steadily changes the angular momentum and eccentricity. The apogee burn then makes a last large change that brings the spacecraft to the final orbit. This last burn is also considerably longer than the burn before it. In the 5-burn case, Fig. (3.7) shows that the last burn is much longer than the first burn. In the 19-burn case, Fig. (3.9) shows the last burn almost just as long as the previous burn; in the 27-burn case, Fig. (3.15) indicates that it is considerably longer.

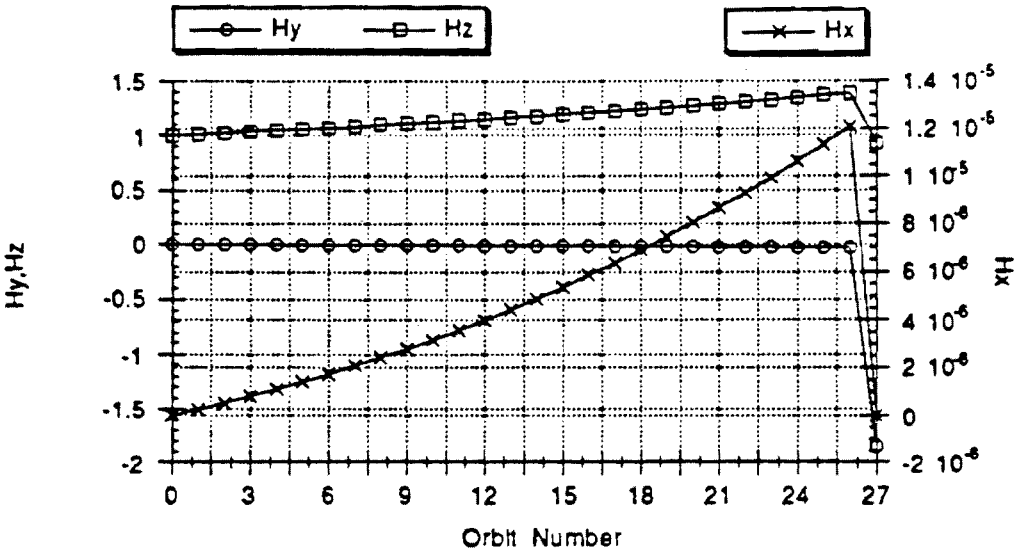


Figure 3.13 Components of the Angular Momentum Vector for each Transfer Orbit vs Orbit Number of a 27-Burn Transfer with a 63.4° Plane Change

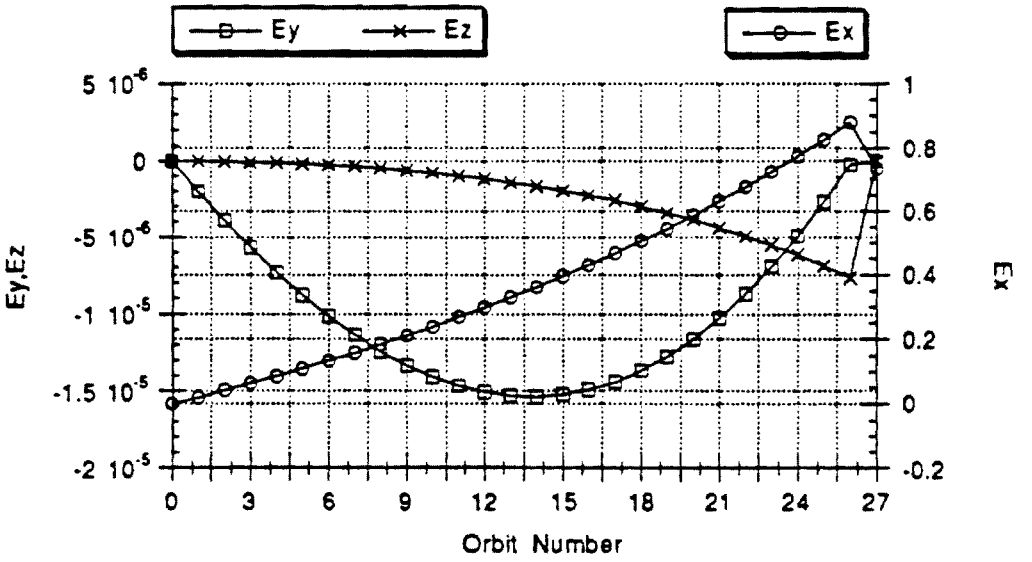


Figure 3.14 Components of the Eccentricity Vector vs Orbit Number for each Transfer Orbit of a 27-Burn Transfer with a 63.4° Plane Change

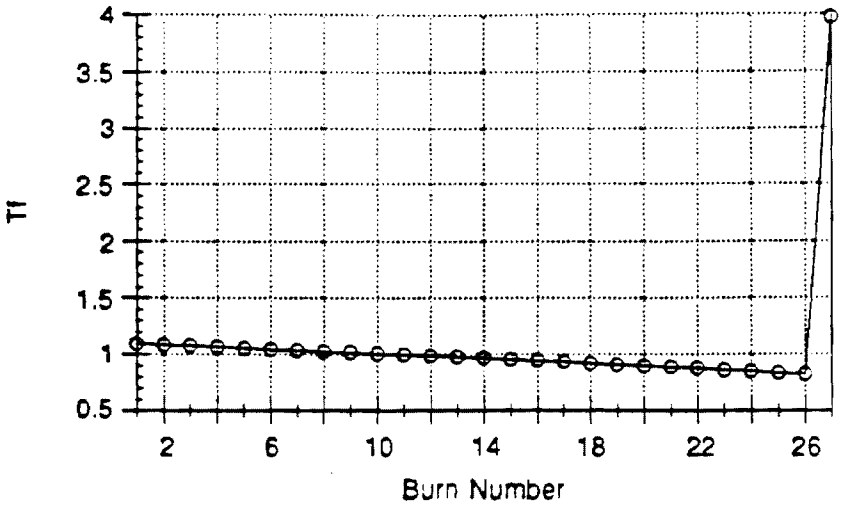


Figure 3.15 Transfer Time vs Orbit Number for each Burn of a 27-Burn Transfer with a 63.4° Plane Change

One feature that seems common to the large number of burns case and the small number of burns case is the use of the distant burn for inclination changes. Referring back to the nonplanar 3-burn transfer shown in Figs. 2.2-2.3, it is clear that the first burn is making most of the inclination change. Also, it is clear from the 27-burn transfer represented in Fig. (3.13) that the h_y component of the angular momentum vector, which indicates the inclination, has very little variation until the final burn takes its value from almost zero to almost -2. This same trend can be seen for the 5-burn transfer represented by Fig. 3.5; where the h_x component indicates inclination for this transfer.

III.4. Inclusion of Perturbation Terms

Neither the Patched Method nor MPM are equipped to produce exact solutions to fuel-optimal orbit transfer problems in the presence of orbit perturbations. Note that including orbit perturbations will cause assumption (i) from $\{P\}$ to be violated.

The tradeoff between making the ideal gravity assumption and obtaining solutions with much larger numbers of burns was deemed acceptable. It is hoped that the

techniques used in this tradeoff will find application in future research into the orbit transfer problem including perturbations.

However, BOUNDSCO was able to obtain a solution including orbit perturbations for the 5-burn transfer presented above in Figure 3.5. Perturbations are considered for this trajectory as opposed to the others, because BOUNDSCO iterations did not converge for the others, even after several trials including initial guesses that were slightly perturbed from the exact solution.

Figures 3.16-3.18 shows the changes in orbital elements and transfer time induced by the inclusion of atmospheric drag and oblateness effects. It is clear that the extremal trajectory includes a lengthened second burn which raises the energy of the second transfer orbit, thereby raising its altitude and decreasing the effect of drag. It is not so clear what decides that the longer burn will be the second and not the first. The nodal regression seems to manifest itself as a decreasing H_y component; it is interesting to note that, like inclination changes, the extremal transfer doesn't make the correction until the last burn. Turning attention to the burn lengths, note that the amount by which the first burn is shortened almost exactly counters the amount by which the next burn is lengthened. A similar trend shows itself for the third and fourth burns. The last burn is only slightly shorter, but not enough to indicate whether the total burn time is longer or shorter. In fact the final mass of the ideal gravity transfer was 3.762; for the transfer with perturbations it was 3.760. This is a performance loss of only 0.07%, a surprising result considering that the individual burn times change by as much as 1.6%.

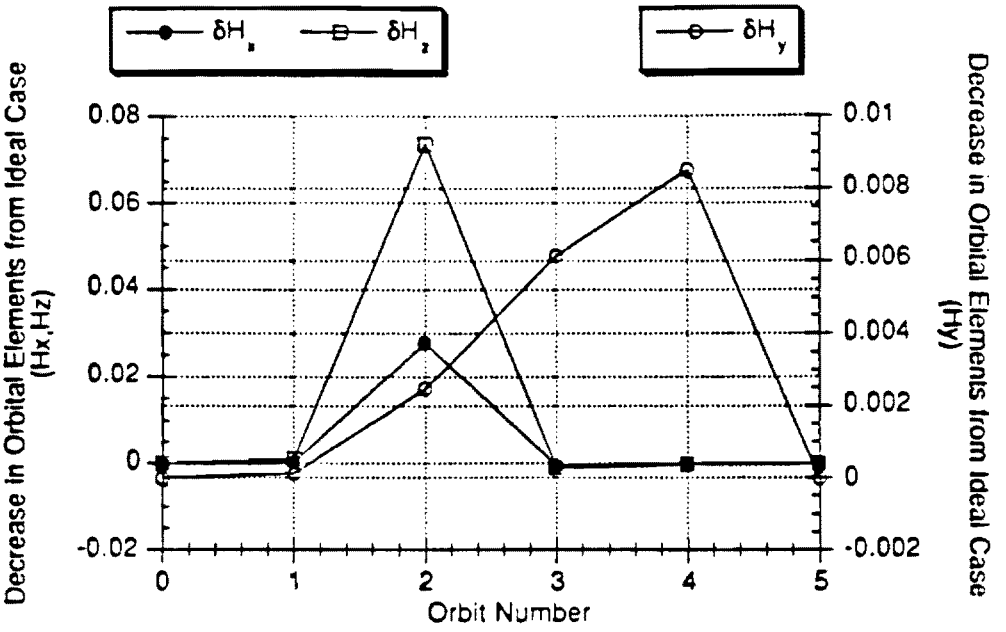


Figure 3.16 Decrease in the Components of the Angular Momentum Vector for the 5-Burn Transfer of Figure 3.5 Considering Orbit Perturbations

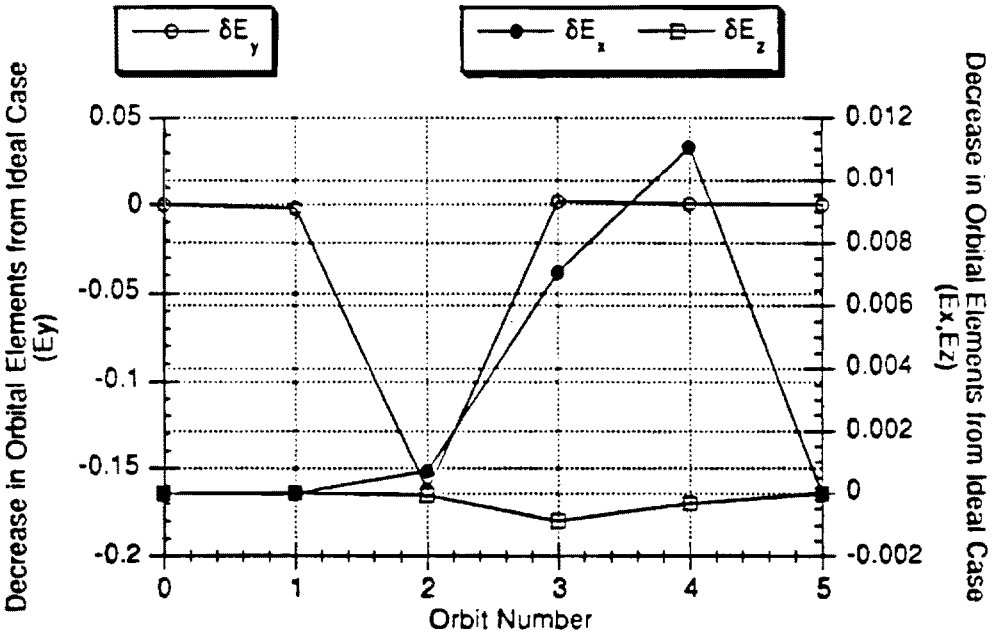


Figure 3.17 Decrease in the Components of the Eccentricity Vector for the 5-Burn Transfer of Figure 3.5 Considering Orbit Perturbations

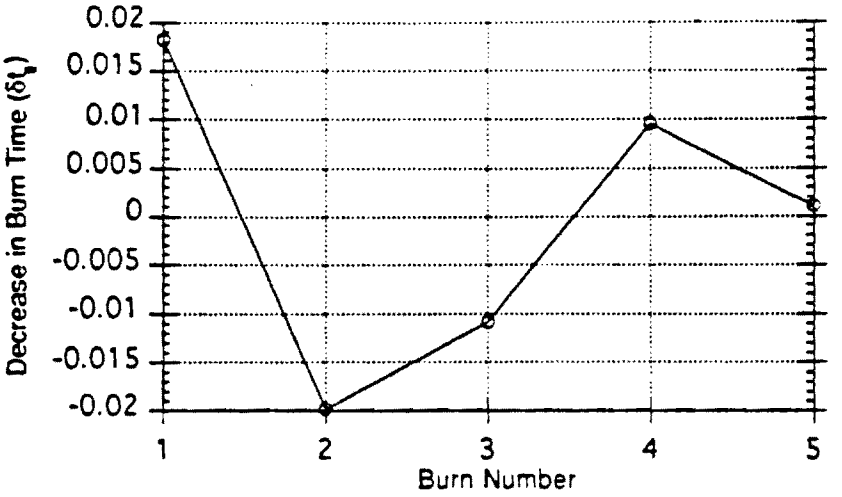


Figure 3.18 Decrease in the Burn Times for the 5-Burn Transfer of Figure 3.5 Considering Orbit Perturbations

III.5 Conclusions

In this section, two new methods for computing multiple-burn orbit transfers are presented. These methods, the Patched Method and the Modified Patched Method, have been developed specifically to fill an apparent gap in computational ability for fuel-optimal transfers with large numbers of burns. For this type of problem, both methods have out-performed BOUNDSCO and MBCM from the previous section.

The conditions upon which each of these methods are based on have been proven equivalent to necessary conditions. However, for both methods it is required that Pontryagin's Maximum Principle be checked after iterations have stopped.

The Patched Method, though slow, was very robust in obtaining solutions. Because of its use of a direct method, it was usually able to obtain the one-burn solutions between each pair of orbits. Also, the optimization of the transfer orbits usually proceeded well in the sense that each iteration would produce a better choice of orbital elements. However, the overall method tended to be quite slow because the cumulative

time required to compute the one-burn transfers in succession was quite long and increased with number of burns.

MPM computed solutions beyond the capability of any of the other methods investigated in this report. MPM was much quicker and slightly less robust, as would be expected of a method more akin to multiple-point shooting. Therefore, it is suggested that the Patched Method be used with a very low tolerance to obtain initial guesses for MPM.

Neither the Patched Method nor MPM is designed to handle orbit perturbations. However, the marked improvement in performance found with these configurations should be motivation enough for a future research effort to produce similar configurations that can handle orbit perturbations efficiently.

Also in this section, a new formulation for the solution of the Lagrange multipliers is presented. This formulation is valid over coast arcs where the Hamiltonian vanishes.

SECTION IV

GUIDANCE FOR OPTIMAL ORBIT TRANSFERS

IV.1. Introduction

The guidance scheme examined here is an implicit one which implements neighboring optimal feedback guidance. An implicit guidance system was chosen due to the fact that that type of guidance system often handles disturbances well[42]. Neighboring optimal feedback guidance was chosen because it has the advantage of being a feedback system, as opposed to open-loop guidance and it can be implemented very easily as with a gain-scheduling scheme. There also appears to be a lack of studies in the literature which examine this type of guidance scheme for this problem.

In this formulation, the initial orbit exit point is assumed to be perturbed from the nominal point but the other boundary condition, specifying the final orbit, is assumed unchanged. The goal is to use the controller to bring the trajectory to the final orbit at some point with minimal fuel.

In order for this guidance scheme to be implementable, the neighboring trajectory must exist; the sufficient conditions for a local extremal must be satisfied. The satisfaction of these conditions for the nominal solution will be shown. Following that, the guidance scheme will be investigated, including the use of a time-to-go indexing scheme.

---

[42]Naidu, D. Subbaram. Aeroassisted Orbital Transfer: Guidance and Control Strategies. New York: Springer-Verlag, 1994.

IV.2. Literature Review

Many researchers have used the first variation to compute extremal solutions to the fuel-optimal orbit transfer problem. However, few, if any, have made use of the conditions related to the second variation of the cost functional in computation. These provide sufficient conditions which, when met, declare an extremal solution as a locally weak optimal solution.

Once the second variation of the cost functional is verified so that it is known whether the sufficient conditions are met, the information obtained can then be used to implement a guidance scheme. Guidance schemes can typically be divided into two categories: implicit and explicit. Implicit guidance systems are characterized by the fact that the vehicle's motion must be precomputed on the ground and then compared to the actual motion. The equations which need to be solved are based upon the difference between these measured and precomputed values. The solutions to these equations are used in the vehicle's steering and velocity control. Explicit guidance systems are generalized by the fact that the vehicle's equations of motion are modeled and solved for by on-board computers during its motion. The solutions for the equations are solved continuously and are used to determine the difference between the vehicle's current motion and its destination. Commands are then generated to alleviate the anticipated error.

Guidance schemes have been presented in various papers.[43] A guidance scheme which is implemented using a linear tangent law is presented by Sinha, Shrivastave, Bhat,

---

[43]Chuang, C.-H., Goodson, T.D., Ledsinger, L.A., "The Second Variation and Neighboring Optimal Feedback Guidance for Multiple Burn Orbit Transfers," *Proceedings of the 1995 AIAA Conference on Guidance, Navigation, and Control*, Baltimore, Maryland, USA.

and Prabhu.[44] In a paper by Lu[45], a nonlinear guidance law is developed using two different strategies. One strategy uses optimal control theory to generate a new optimal trajectory onboard from the start, while the other uses flight-path-restoring-guidance to bring the trajectory back to the nominal. A guidance scheme that is developed using inverse methods for unthrusted, lift-modulated vehicles along an optimal space curve is presented by Hough.[46] Linearized guidance laws applicable to many different types of space missions are presented by Tempelman.[47] These guidance laws are based on fixed and free final time arrivals. Naidu[42] presents a neighboring optimal guidance scheme applicable to aeroassisted orbital transfers.

IV.3. Preliminary Considerations

Earlier, the optimal orbit transfer problem was given as a maximization problem. To conform to the convention used for the second variation[36] it is transformed to a minimization problem. For the minimization problem, the performance index can be made negative and considered a cost functional

$$J = -m(t_f) \tag{4.1}$$

As the necessary conditions are first-order conditions, they remain unchanged. However, Lawden's pointer vector theory is second-order and requires that the control be such that

$$\mathbf{e}_T = \frac{-\lambda_v}{|\lambda_v|} \tag{4.2}$$

---

[44]Sinha, S. K., S. K. Shrivastava, M. S. Bhat, and K. S. Prabhu. "Optimal Explicit Guidance for Three-Dimensional Launch Trajectory," *Acta Astronautica*. Vol. 9, 1989, pp. 115-125.

[45]Lu, P., "A General Nonlinear Guidance Law," *Proceedings of the the AIAA Guidance, Navigation, and Control Conference*, Scottsdale, Arizona, 1994.

[46]Hough, M. E., "Explicit Guidance Along an Optimal Space Curve," *Journal of Guidance, Control, and Dynamics*. Vol. 12, 1989, pp. 495-504.

[47]Tempelman, W., "Linear Guidance Laws for Space Missions," *Journal of Guidance, Control, and Dynamics*. Vol. 9, 1986, pp. 495-502.

Furthermore, Pontryagin's Minimum Principle requires that an extremal solution satisfy

$$\begin{aligned} H_S < 0, \quad T &= T_{max} \\ H_S > 0, \quad T &= 0 \end{aligned} \tag{4.3}$$

where

$$H_S = -\left(\frac{|\lambda_v|}{m} + \frac{\lambda_m}{g_o I_{sp}} \right) \tag{4.4}$$

If an extremal solution to the maximization problem is given as state time history $\mathbf{x}(t)$, Lagrange-multiplier time history $\lambda(t)$, and Lagrange multipliers ν, (associated with boundary conditions) then an extremal solution for the minimization problem with the cost function in Eq. (4.1) can be constructed as $\mathbf{x}(t)$, $(-1)*\lambda(t)$, and $(-1)*\nu$.

Additionally, it makes more sense in the planar guidance problem to consider the control as an angle θ, rather than individual components of a unit vector. This simplifies analysis because the control is now a scalar. Equation (4.2) now gives

$$tan(\theta) = -\frac{\lambda_v}{\lambda_u} \tag{4.5}$$

A practical approach to guidance is suggested by previous results in this report. If a multiple-burn transfer can be thought of as consisting of multiple optimal one-burn transfers, then it should be reasonable to examine a guidance scheme that attempts to match each of the intermediate transfer orbits of the multiple-burn transfer. In other words, use neighboring optimal feedback guidance for one burn at a time.

This is not suggested to be an optimal guidance scheme. By focusing on each burn with neighboring optimal feedback, but not considering the trajectory as a whole, this guidance scheme becomes a sub-optimal guidance scheme.

Each burn can be considered an extremal solution. These extremal solutions are considered to have a fixed initial point and free transfer time but the final point is only constrained in that it must lie on the final orbit. Recall, however, that in computing the multiple-burn transfer the initial point was not fixed; this condition is imposed for practical considerations. If the spacecraft is delivered to the correct orbit, and coasting to the nominal burn-on point has zero cost, then there is no reason to attempt to compute a new burn-on point. This reasoning holds for the beginning of each burn.

IV.4. The Second Variation for One-Burn Problems

Considering the second variation of the augmented cost functional, J, a new optimal control problem can be stated.[36] In this new problem, the state is $\delta\mathbf{x}$, the control δu, and the Lagrange-multipliers are $\delta\lambda$ and $d\nu$. The new problem is linear and can be solved using a sweepback method. For the problem considered here, $\mathbf{x}=[\mathbf{r}^T\ \mathbf{v}^T\ m]^T$ and $u=\theta$.

When the final time is free for optimization, the transversality condition must be satisfied by the nominal solution. The notation for this condition is

$$\Omega(\mathbf{x},\nu,t)\Big|_{t=t_f} = \left(\frac{dG}{dt}\right)_{t=t_f} = \left(\frac{\partial G}{\partial \mathbf{x}}\dot{\mathbf{x}}\right)_{t=t_f} = 0 \tag{4.6a}$$

where

$$G(\mathbf{x},\nu) = \phi(\mathbf{x}) + \nu^T\psi(\mathbf{x}) \tag{4.6b}$$

In general, neighboring optimal feedback guidance allows consideration of changes in boundary conditions. No such changes are considered, assuming that the destination orbit is fixed. Formulation will be made below for the free final time case.

The change in state and costate can be estimated with a linear time-varying dynamic system. This dynamic system is given below, where it is understood that matrix functions are evaluated with the nominal trajectory.

$$\frac{d}{dt}\delta\mathbf{x} = \mathbf{A}(t)\delta\mathbf{x} - \mathbf{B}(t)\delta\boldsymbol{\lambda} \tag{4.7}$$

$$\frac{d}{dt}\delta\boldsymbol{\lambda} = -\mathbf{C}(t)\delta\mathbf{x} - \mathbf{A}^{\mathrm{T}}(t)\delta\boldsymbol{\lambda} \tag{4.8}$$

where

$$\mathbf{A}(t) = \mathbf{f}_{\mathbf{x}} - \mathbf{f}_{\mathbf{u}}H_{\mathbf{uu}}^{-1}H_{\mathbf{ux}} \tag{4.9}$$

$$\mathbf{B}(t) = \mathbf{f}_{\mathbf{u}}H_{\mathbf{uu}}^{-1}\mathbf{f}_{\mathbf{u}}^{\mathrm{T}} \tag{4.10}$$

$$\mathbf{C}(t) = H_{\mathbf{xx}} - H_{\mathbf{xu}}H_{\mathbf{uu}}^{-1}H_{\mathbf{ux}} \tag{4.11}$$

Evaluating Eqs. (4.7)-(4.11) the recurring terms in the differential equations are:

$$\mathbf{f}_{\mathbf{x}} = \begin{bmatrix} 0 & 0 & -\left(\frac{\mu}{r^3}\right) - \frac{3\mu x^2}{r^5} & \frac{3\mu xy}{r^5} & 0 \\ 0 & 0 & \frac{3\mu xy}{r^5} & -\left(\frac{\mu}{r^3}\right) - \frac{3\mu y^2}{r^5} & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -\frac{T}{m}\cos(\theta) & -\frac{T}{m}\sin(\theta) & 0 \end{bmatrix}^{\mathrm{T}} \tag{4.12}$$

$$\mathbf{f}_{\theta} = \begin{bmatrix} 0 & 0 & -\frac{T}{m}\sin(\theta) & \frac{T}{m}\cos(\theta) & 0 \end{bmatrix}^{\mathrm{T}} \tag{4.13}$$

$$H_{xx} = \begin{bmatrix} \mathbf{Q} & 0 \\ 0 & 2\frac{T}{m^3}|\lambda_v| \end{bmatrix} \tag{4.14}$$

$$\mathbf{Q} = \frac{3\mu}{r^7}\begin{bmatrix} \{(3\lambda_x x + \lambda_v y)r^2 - 5(\lambda_v^T \mathbf{r})x^2\} & \{(\lambda_x y + \lambda_v x)r^2 - 5(\lambda_v^T \mathbf{r})xy\} \\ \{(\lambda_x y + \lambda_v x)r^2 - 5(\lambda_v^T \mathbf{r})xy\} & \{(3\lambda_v y + \lambda_x x)r^2 - 5(\lambda_v^T \mathbf{r})xy\} \end{bmatrix} \tag{4.15}$$

$$H_{uu} = \frac{T}{m}|\lambda_v| \tag{4.16}$$

$$H_{ux} = 0 \tag{4.17}$$

note that $\mathbf{r} = [x \quad y]^T$, $\mathbf{v} = [u \quad v]^T$, and $\lambda_v = [\lambda_u \quad \lambda_v]^T$ are taken as the nominal trajectory. Using the sweepback method for nonlinear terminal constraints the form for $\delta\lambda$ and $\delta\psi$ can be written as

$$\delta\lambda(t) = \bar{\mathbf{P}}(t)\delta\mathbf{x}(t) + \bar{\mathbf{S}}(t)d\nu \tag{4.18}$$

$$\delta\psi = \bar{\mathbf{S}}^T(t)\delta\mathbf{x}(t) + \bar{\mathbf{V}}(t)d\nu \tag{4.19}$$

which allows the solution for $d\nu$ to be written as

$$d\nu = \bar{\mathbf{V}}^{-1}(t_o)\left[\delta\psi - \bar{\mathbf{S}}^T(t_o)\delta\mathbf{x}(t_o)\right] \tag{4.20}$$

As mentioned above, $\delta\psi$=0 will be considered here. The matrices $\bar{\mathbf{P}}(t)$, $\bar{\mathbf{S}}(t)$, and $\bar{\mathbf{V}}(t)$, are computed using the following relations:

$$\bar{\mathbf{P}}(t) = \mathbf{P}(t) - \frac{\mathbf{m}(t)\mathbf{m}^T(t)}{\alpha(t)} \tag{4.21}$$

$$\bar{\mathbf{S}}(t) = \mathbf{S}(t) - \frac{\mathbf{m}(t)\mathbf{n}^T(t)}{\alpha(t)} \tag{4.22}$$

$$\bar{\mathbf{V}}(t) = \mathbf{V}(t) - \frac{\mathbf{n}(t)\mathbf{n}^T(t)}{\alpha(t)} \tag{4.23}$$

Now the matrices $\mathbf{P}(t)$, $\mathbf{S}(t)$, $\mathbf{V}(t)$, $\mathbf{m}(t)$, $\mathbf{n}(t)$, and the scalar function $\alpha(t)$ are computed from a dynamic system. The boundary condition equations for this system are given by:

$$\mathbf{P}(t_f) = \left[\phi_{xx} + \left(\nu^T \psi_x\right)_x\right]_{t=t_f} \tag{4.24}$$

$$\mathbf{S}(t_f) = \left[\psi_x^T\right]_{t=t_f} \tag{4.25}$$

$$\mathbf{V}(t_f) = 0 \tag{4.26}$$

where in the development for the orbital transfer these are:

$$\mathbf{P}(t_f) = \begin{bmatrix} a & b & d & e & 0 \\ b & c & f & g & 0 \\ d & f & h & i & 0 \\ e & g & i & j & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4.27}$$

$$a = \nu_2\mu\left[\frac{x}{r^3} - \frac{3x^3}{r^5} + \frac{2x}{r^3}\right] + \nu_3\mu\left[\frac{y}{r^3} - \frac{3x^2y}{r^5}\right] \tag{4.28a}$$

$$b = \nu_2\mu\left[\frac{y}{r^3} - \frac{3x^2y}{r^5}\right] + \nu_3\mu\left[\frac{x}{r^3} - \frac{3xy^2}{r^5}\right] \tag{4.28b}$$

$$c = \nu_3\mu\left[\frac{y}{r^3} - \frac{3y^3}{r^5} + \frac{2y}{r^3}\right] + \nu_2\mu\left[\frac{x}{r^3} - \frac{3xy^2}{r^5}\right] \tag{4.28c}$$

$$d = -\nu_3 v \tag{4.28d}$$

$$e = \nu_1 - \nu_2 u + 2\nu_3 v \tag{4.28e}$$

$$f = -\nu_1 - \nu_2 v + 2\nu_3 u \tag{4.28f}$$

$$g = -\nu_2 u \tag{4.28g}$$

$$h = 2\nu_3 y \tag{4.28h}$$

$$i = -\nu_3 x - \nu_2 y \tag{4.28i}$$

$$j = 2\nu_2 x \tag{4.28j}$$

and expression for Eq. (4.25) was previously given as Eq. (2.11).

Following from the assumptions expressed as Eqs. (4.18)-(4.19), the following nonlinear equations for **P**, **S**, and **V** must be integrated backwards. The results will be used to check the sufficient conditions governing a minimizing solution.

$$\dot{\mathbf{P}} = -\mathbf{PA} - \mathbf{A}^T\mathbf{P} + \mathbf{PBP} - \mathbf{C} \tag{4.29}$$

$$\dot{\mathbf{S}} = -\left(\mathbf{A}^T - \mathbf{PB}\right)\mathbf{S} \tag{4.30}$$

$$\dot{\mathbf{V}} = \mathbf{S}^T\mathbf{BS} \tag{4.31}$$

$$\dot{\mathbf{m}} = -\left(\mathbf{A}^T - \mathbf{PB}\right)\mathbf{m} \tag{4.32}$$

$$\dot{\mathbf{n}} = \mathbf{S}^T\mathbf{Bm} \tag{4.33}$$

$$\dot{\alpha} = \mathbf{m}^T\mathbf{Bm} \tag{4.34}$$

with the following boundary conditions applying

$$\mathbf{m}\left(t_f\right) = \left(\frac{d\Omega}{d\mathbf{x}}\right)^T_{t=t_f} \tag{4.35}$$

$$\mathbf{n}\left(t_f\right) = \left(\frac{d\psi}{dt}\right)_{t=t_f} \tag{4.36}$$

$$\alpha\left(t_f\right) = \left(\frac{d\Omega}{dt}\right)_{t=t_f} \tag{4.37}$$

The sufficient conditions for a minimizing solution can now be stated as follows:

convexity condition: $H_{\theta\theta}(t) > 0$ for $t_o \le t \le t_f$ (4.38)

normality condition:

$\overline{\mathbf{V}}^{-1}(t)$ exists for $t_o \le t < t_f$ (4.39a)

$\alpha^{-1}(t)$ exists for $t_o \le t < t_f$ (4.39b)

conjugate point condition: $\overline{\mathbf{P}}(t) - \overline{\mathbf{S}}(t)\overline{\mathbf{V}}^{-1}(t)\overline{\mathbf{S}}^T(t)$ finite for $t_o \le t < t_f$ (4.40)

The convexity condition is satisfied for any transfer satisfying Equation (4.5). This can be seen by noting that Eq. (4.16) is positive definite, irrespective of the time history for the Lagrange multipliers.

The eigenvalues of $\bar{V}$ are plotted in Figure 4.1. Figures 4.2-4.4 plot the elements of the conjugate point condition matrix. Figure 4.5 is a plot of $\alpha(t)$. Figure 4.1 shows that $\bar{V}$ is positive definite in the required interval. Figure 4.5 shows that $\alpha(t)$ is negative definite in the required interval. Since the normality condition requires that the inverse of $\bar{V}$ and $\alpha(t)$ exists in the interval, this solution is normal. Figures 4.2-4.4 show that the conjugate point condition is satisfied. The elements are bounded in the required interval and grow asymptotically at the final time; the curves in the figures have been truncated to show their variations prior to this asymptotic growth. Therefore, this solution satisfies the sufficient conditions for minimizing the cost functional with free transfer time.

It seems appropriate to first attempt the guidance scheme for a relatively uncomplicated transfer. Such a transfer was presented in Fig. 2.1 and discussed in subsection [II.2.4]. The transfer is planar; no plane changes occur. The guidance scheme considered here will be simulated for this trajectory.

IV.4.1. Neighboring Optimal Feedback Guidance

Conveniently, construction of a neighboring optimal feedback guidance law uses the same information as that required to check the second variation of the cost functional. As a result, much of the derivation required of guidance law has been stated already. The remaining discussion will describe how to form the feedback control law and adjust the characteristics of the bang-bang control in a feedback law.

The control, $\delta\theta$, for the fixed final time problem can be found using

$$\delta\theta(t) = -H_{uu}^{-1}\left[\left(f_u^T\overline{P}\right)\delta x + f_u^T\overline{S}d\nu\right] \qquad (4.41)$$
$$= -H_{uu}^{-1}\left[f_u^T\left(-\overline{S}\overline{V}^{-1}\overline{S}^T\right)\right]\delta x$$

and the change in the final time, dt_f, is:

$$dt_f = -\left[\left(\frac{m^T}{\alpha} - \frac{n^T}{\alpha}\overline{V}^{-1}\overline{S}^T\right)\right]\delta x \qquad (4.42)$$

Evaluating dt_f determines when the thrust will be turned off to complete the transfer.

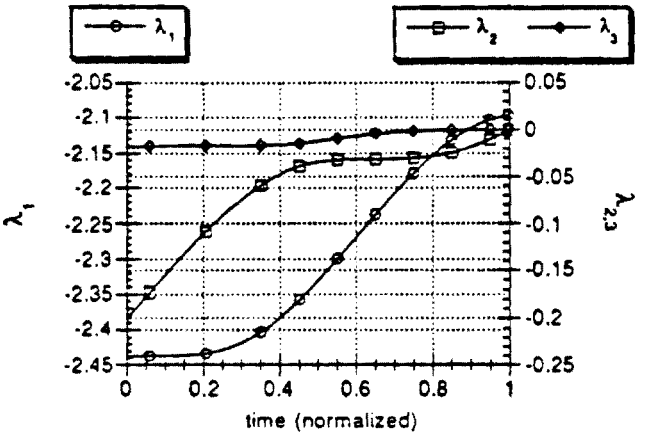


Figure 4.1 Plot of Eigenvalues of $\overline{V}(t)$ for Two-Burn Extremal, Last Burn

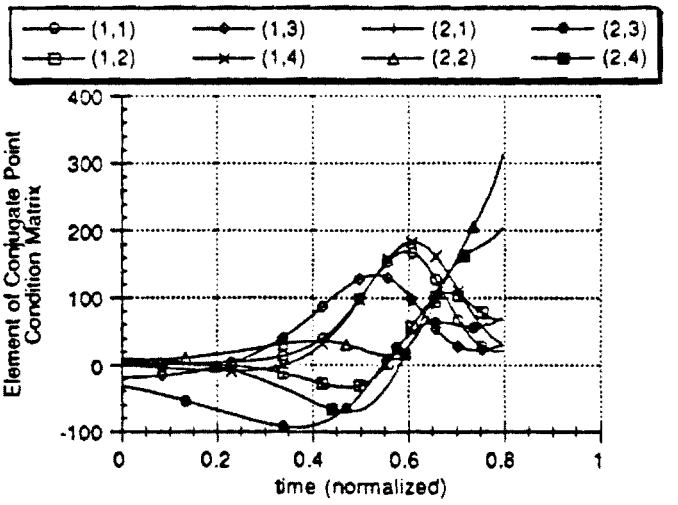


Figure 4.2 Plot of Elements of Conjugate Point Condition Matrix for Two-Burn Extremal, Last Burn

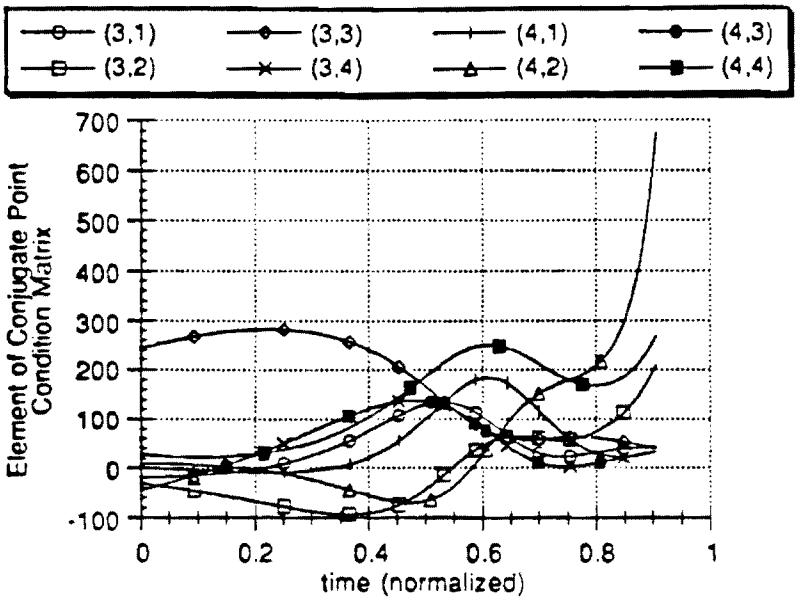


Figure 4.3 Plot of Elements of Conjugate Point Condition Matrix for Two Burn Extremal, Last Burn

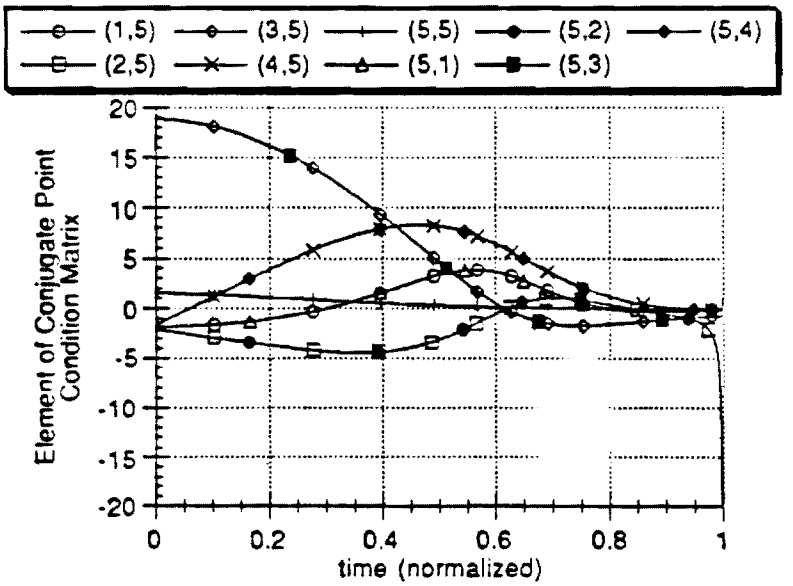


Figure 4.4 Plot of Elements of Conjugate Point Condition Matrix for Two Burn Extremal, Last Burn

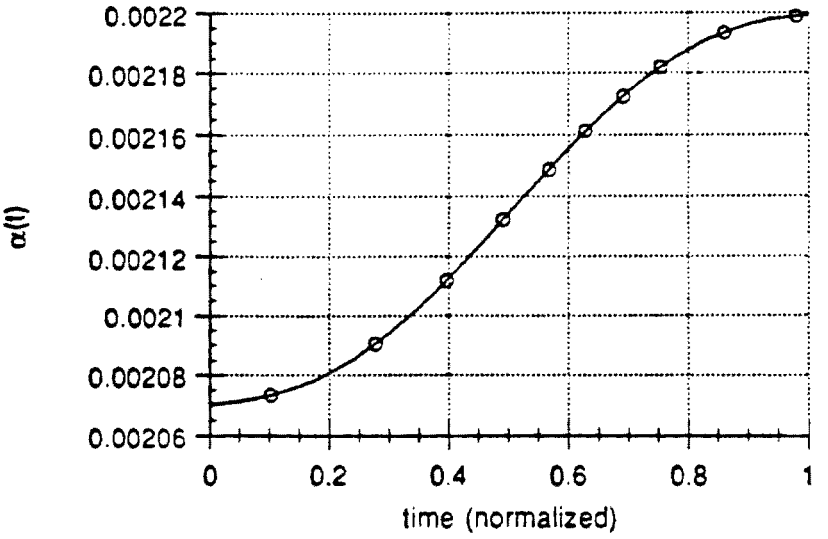


Figure 4.5 Plot of $\alpha(t)$ for Two Burn Extremal, Last Burn

This continuous feedback law has been constructed by estimating $d\nu$ at each instant of time instead of solving for $d\nu$ at the initial time and then using this value for all time

The feedback law depends on **P**, **S**, and **V** as functions of time. A particular advantage of neighboring optimal feedback is that the linearized TPBVP only has to be solved once. Afterwards, sampled values of the feedback gains may be stored. The feedback gains may then be computed for any time by interpolation between stored values. Use of this control should keep the spacecraft on a neighboring optimal solution and deliver it to the required orbit.

The block diagram for the feedback controller needed for neighboring optimal feedback guidance is shown in Figure 4.6.

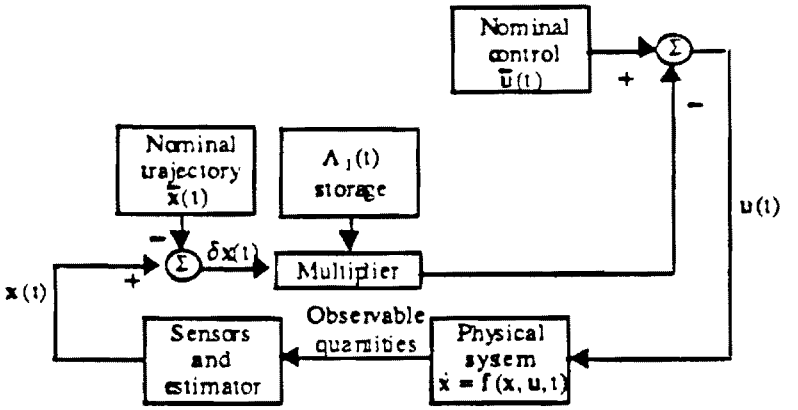


Figure 4.6 Diagram of Neighboring Optimal Feedback Controller Implementation

where $\Lambda_1(t)$ is the feedback gain from Eq. (4.41), computing $\delta\theta$.

IV.4.2. Simulation of the Guidance Algorithm

Justification for a feedback algorithm lies in Fig. 4.7 and Fig. 4.8. It can be noted that there is error in the variation of the states from the neighboring optimal trajectory when guidance is not used, Fig. 4.7, i.e., when the control correction is not used. However, Fig. 4.8 shows that a feedback law is needed because when implementing it, the errors in the variation of the states becomes much less, comparatively, than that using no guidance whatsoever. The neighboring optimal trajectory referenced in Figs. 4.7-4.8 was computed with BOUNDSCO.

IV.4.3. Time-To-Go Implementation

Since this problem is a free final-time problem, the possibility exists that the final-time will increase and the guidance algorithm will "run out of gains"; this is a familiar issue for neighboring optimal feedback guidance. The approach used in this study is based on discretizing the gains by N time nodes $\{t_1,...,t_i,...t_N\}$ where t_N is earlier than the nominal t_f. The gains at the nominal t_f will be infinite and impractical to store. Both the gains for calculating dt_f, via Eq. (4.42), and for $\delta\theta$, via Eq. (4.41), are then calculated at any time by linear interpolation between stored values.

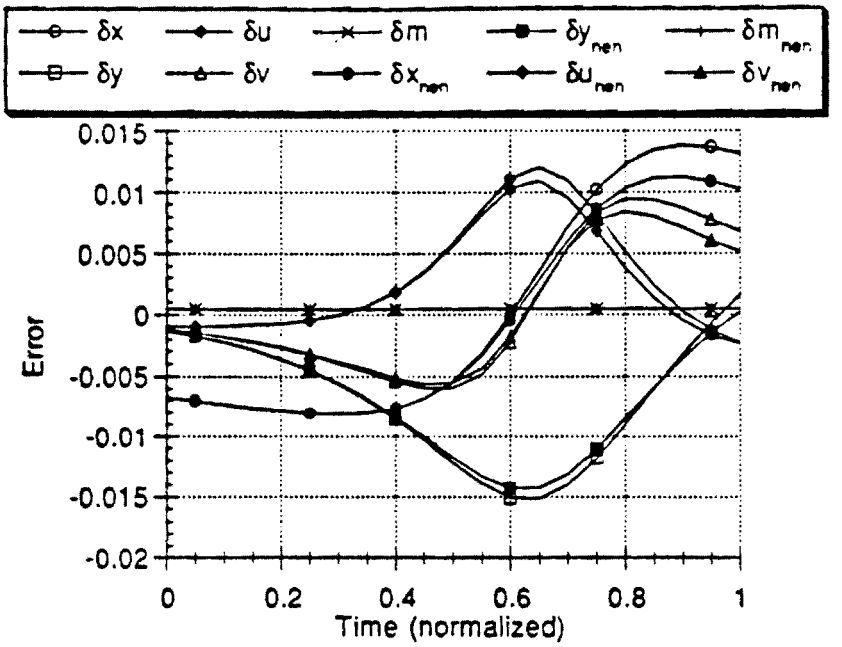


Figure 4.7 Plot of State Variation from the Nominal Trajectory vs. Time for Neighboring Optimal Trajectory ($()_{nen}$) and a Trajectory Without Guidance (no subscript)

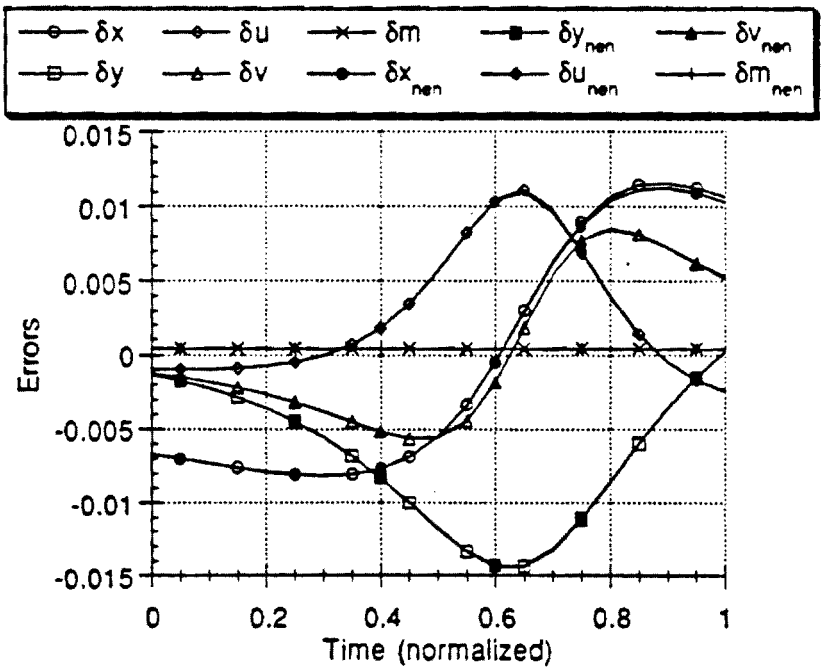


Figure 4.8 Plot of State Variation from the Nominal Trajectory vs. Time for Neighboring Optimal Trajectory ($()_{nen}$) and a Trajectory With Guidance (no subscript)

To consider time-to-go, the guidance must make active use of the dt_f estimation. Since both the nominal and the actual trajectories start at t_1, dt_{f1} can be initially calculated using the gains at that time. The length of the first guidance interval is then found by relating it to the estimated time-to-go.

$$\Delta t_{g1} = \frac{t_f + dt_{f1}}{t_f}(t_2 - t_1) \tag{4.43}$$

Then, at the end of the i-1th guidance interval, the gains at t_i are used to calculate dt_{fi}. Using this information, the length of the ith guidance interval can be computed as

$$\Delta t_{gi} = \frac{t_f + dt_{fi} - \sum_{j=1}^{i-1} \Delta t_{gj}}{t_f - t_i}(t_{i+1} - t_i) \tag{4.44}$$

This continues until Δt_i is computed as zero or a negative number or until i=N. When i=N, the Nth gain is used for the entire interval Δt_N. When this interval ends, the guidance scheme is finished.

The plots below compare guidance performance with and without this time-to-go formulation. The curves represent the time history of the boundary condition error, i.e. Eqs. (1.12) minus the desired orbital elements, evaluated continuously. Figure 4.9 makes continuous use of the gains but indexes these gains at the current actual time without calculating dt_f. For the perturbation simulated, the transfer time needs to increase and this first scheme must terminate prematurely. Figure 4.10 makes use the discretized gains and time-to-go formulation. This simulation also incorporates a practical saturation limit on the size of the gains. The improvement due to the time-to-go formulation is obvious when comparing these plots. Therefore, this is both a practical and superior

implementation of the continuous burn guidance considering the boundary condition error.

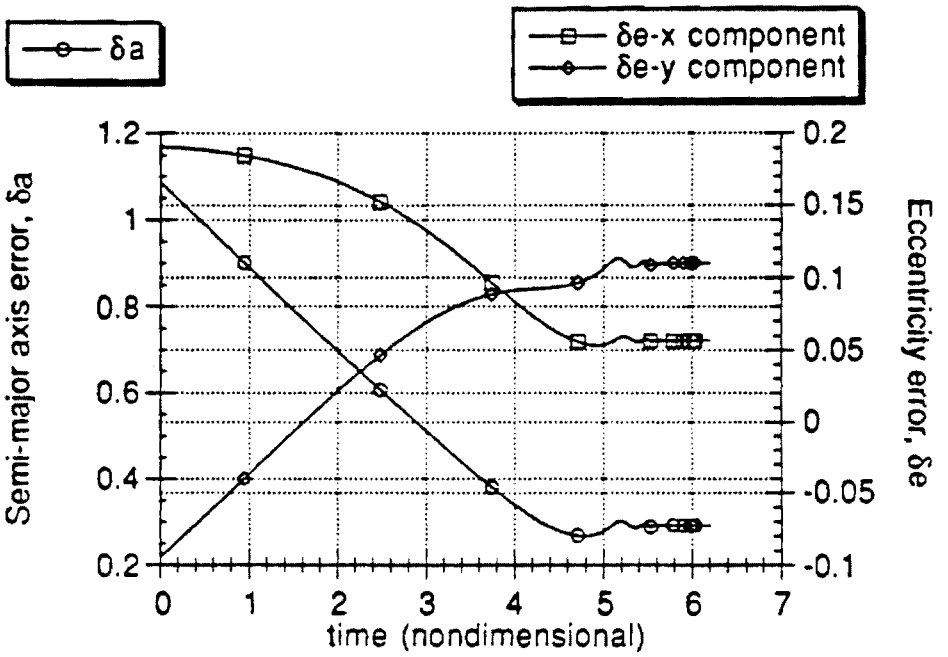


Figure 4.9 Plot of Boundary Condition Error for Continuous Guidance

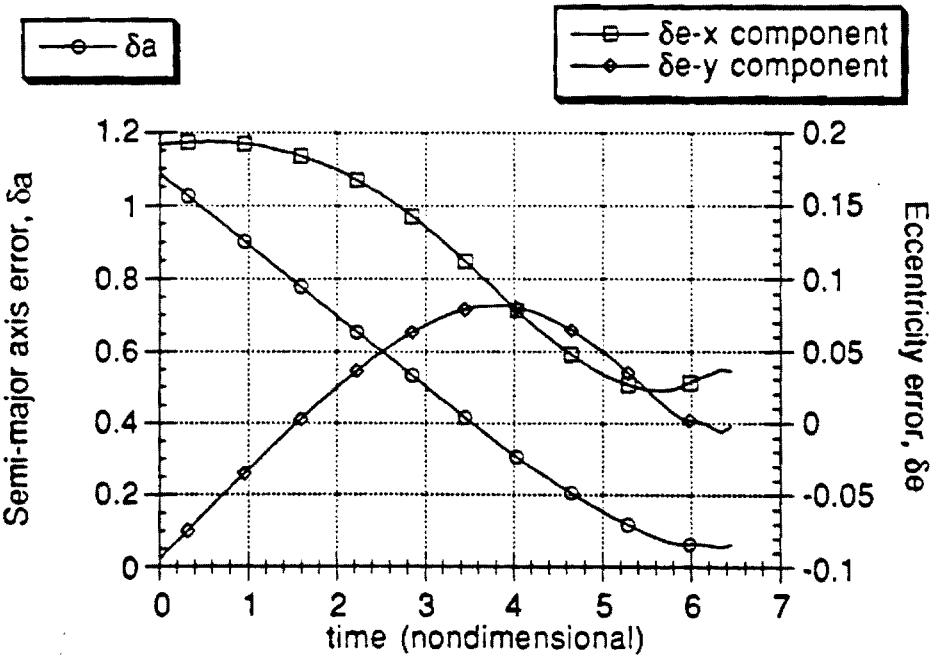


Figure 4.10 Plot of Boundary Condition Error for Discrete Guidance with Time-to-Go

IV.5. Multiple Burn Guidance

The guidance for multiple burns can also be discretized. For the two burn case, discretized guidance using time-to-go is used for the first burn. The guidance algorithm will place the spacecraft on the intermediate transfer orbit via the neighboring optimal trajectory. Since the cost on this coast arc is zero, the spacecraft can coast on this arc until it reaches the point at which the next burn is to start. Once the spacecraft reaches this point, discretized guidance using time-to-go can be used again for the second burn. The boundary conditions for the second burn should than be satisfied by the neighboring path. For multiple burns, this guidance scheme is extended in a straightforward manner.

The guidance scheme detailed above was used to recover the two burn transfer of Fig. 2.1 in the presence of an initial perturbation. Fig. 4.11 shows the boundary condition errors for the first burn given an initial perturbation of 10^{-3} in non-dimensionalized units. The boundary conditions are satisfied rather well for this burn. The resulting boundary condition errors for the second burn are shown in Figure 4.12. The boundary conditions are satisfied very well for this burn.

Figures 4.13 & 4.14 show the boundary condition errors during the second burn for a perturbation of the same magnitude as above in only the x position and the u velocity, respectively. Note that the error in the boundary conditions is slightly greater in Figure 4.14. This suggests that the trajectory is more sensitive to disturbances in the u velocity than in the x position.

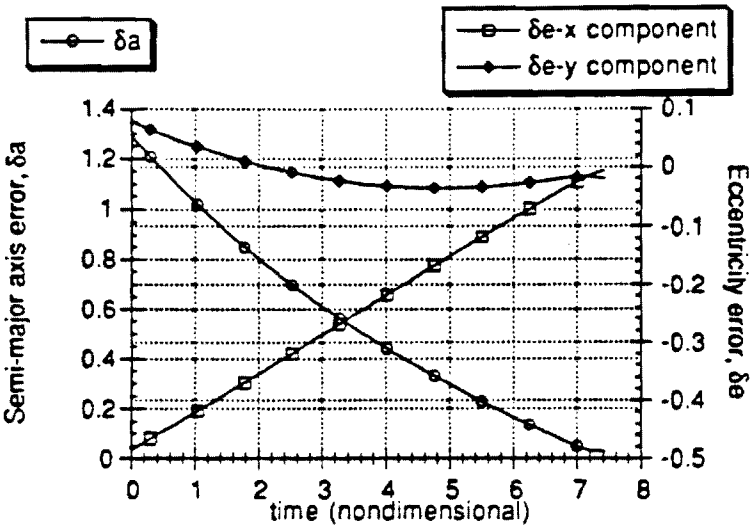


Figure 4.11 Plot of Boundary Condition Error for Discrete Guidance During the First Burn

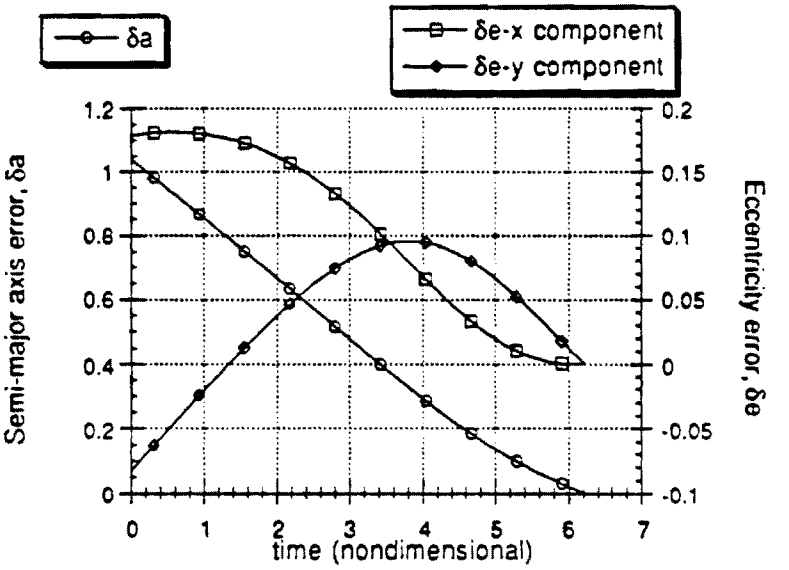


Figure 4.12 Plot of Boundary Condition Error for Discrete Guidance During the Second Burn (Continuation of Fig. 4.11)

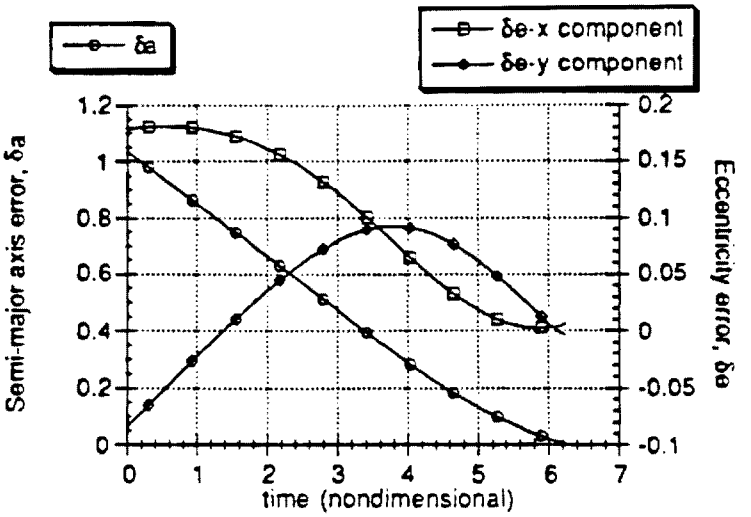


Figure 4.13 Plot of Boundary Condition Error for Discrete Guidance During the Second Burn for error in x at the initial time

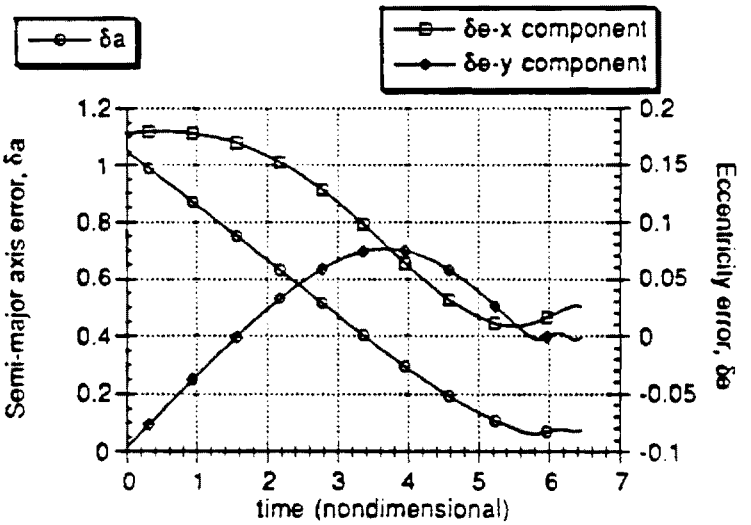


Figure 4.14 Plot of Boundary Condition Error for Discrete Guidance During the Second Burn for error in u

The resulting orbit transfer trajectory is shown in Figure 4.15. This plot corresponds to the boundary condition errors as shown in Figures 4.11 and 4.12.

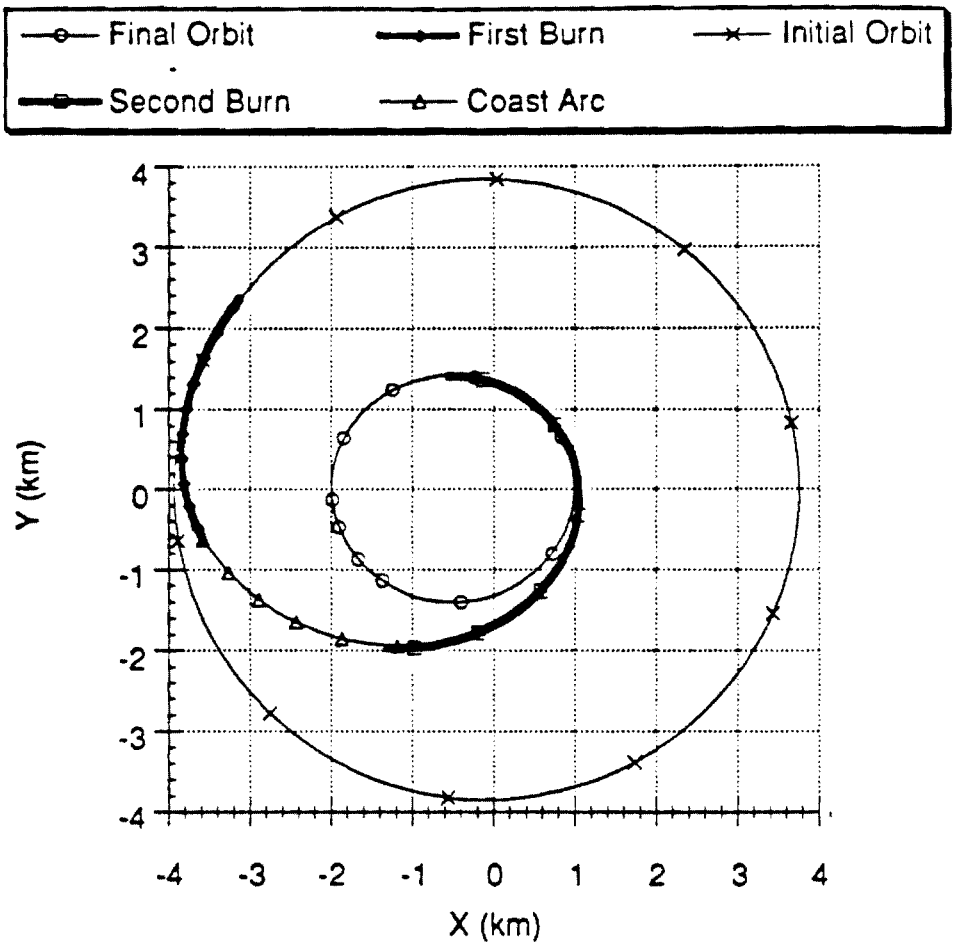


Figure 4.15 Plot of the Two Burn Orbit Transfer (from Fig. 2.1) with Initial Perturbation

IV.6. Conclusions

Extremal one burn trajectories have been shown to be weak locally optimal solutions using sufficient conditions. This does not prove that the multiple-burn transfer from which they were taken is itself a weak locally optimal solution, but it does allow the use of a new suboptimal guidance scheme.

This scheme was shown to reduce the terminal errors for small perturbations of the initial state. To increase the size of allowable perturbations, a time-to-go indexing

scheme was simulated. This time-to-go indexing did improve the performance of the guidance scheme.

The suboptimal multiple-burn guidance with time-to-go indexing was simulated for a planar transfer. The performance of this guidance scheme did not match expectations. The implication is that the region in which a linear control correction is a valid assumption was quite small. Actually, this is not a surprising conclusion since obtaining the nominal solutions is usually quite a challenge for iterative algorithms that attempt linear corrections for each iteration. If indeed this implication is correct, then a more sophisticated approach for neighboring feedback control is required.

SECTION V

CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER STUDY

V.1. Transfers with Small Numbers of Burns

It has been found that methods already present in the literature are capable of computing fuel-optimal orbit transfers with small numbers of burns. The methods investigated here were multiple-point shooting and modified shooting. However, a common way to attempt to increase the performance of a transfer is to increase the number of burns executed and, unfortunately, these methods are not very robust in that sense.

A new method has been introduced that is very useful for adding burns to fuel-optimal orbit transfers. The method is used in conjunction with homotopy and an iterative technique for computing transfers; the iterative technique must incorporate knowledge of the Lagrange multipliers. The method does require that the initial point, the final point, and the transfer time be free for optimization. It also assumes that the transfer is performed under the influence of ideal gravity. This assumption is required to obtain the switching function property that the method relies on.

It is recommended that this method be further developed such that orbit perturbations are taken into account. Since the switching function property in question no longer applies for this case, the task is challenging. Obviously, a fairly different approach must be taken. It is likely that requiring trajectories to begin and end with coast arcs will be necessary, since cost arcs will no longer be orbits. Perhaps then some

conditions may be identified under which the coast arcs could be extended to find optimal locations for the burns to be added.

V.3. Transfers with Large Numbers of Burns

The results of this research point to the Modified Patched Method as a practical way to compute fuel-optimal transfers with large numbers of burns. It does not appear that such a method existed previously in the literature, making MPM and theoretical results behind it the central contributions of this report.

An interesting spin-off of the theoretical development is a new formulation for the integration of the Lagrange multipliers over a time-optimal coast arc for the nonplanar case assuming ideal gravity. The formulation results from satisfaction of Lemma III.2. This particular formulation proved quite useful for MPM and may prove useful in future algorithms and future theoretical developments.

MPM does not allow for orbit perturbations. This restriction was a small price to pay for performance previously unobtained, *viz.* the ability to compute transfers with upwards of 27-burns and large inclination changes. Now that this performance has been obtained for the ideal gravity case, it is suggested that a future research effort should be able to produce a method with similar performance, or better, while taking orbit perturbations into account.

If an attempt is made to adapt MPM for orbit perturbations without recovering any properties lost, then MPM will degenerate into multiple-point shooting. This study has already concluded that multiple-point shooting does not perform well for large numbers of burns; therefore, some recovery of the properties from Theorem III.1 and/or Theorem III.2 must be made. Since the concept central to both the Patched Method and

MPM is the relationship of the optimal orbit transfer problem with the problem expressed by (3.1), it seems reasonable to expect some form of (3.1) to be recovered in the presence of orbit perturbations.

V.2. Multiple-Burn Guidance

A suboptimal multiple-burn guidance scheme was developed through this research and its performance investigated. The scheme may be described as "burn-by-burn" neighboring optimal feedback guidance with a time-to-go indexing scheme for each burn. The performance of this guidance scheme did not match expectations.

Since guidance has much practical importance, it is suggested that future research attempt to develop an improved guidance scheme. It is likely that this would involve techniques to improve neighboring optimal feedback or replacing this with some other one-burn guidance scheme. On the other hand, a future research effort might attempt to find an optimal guidance algorithm for the multiple-burn transfer as a whole. Since there is a strong relationship between the sufficient conditions for optimality and the computation of neighboring optimal feedback gains for the one-burn problem, a similar relationship might be expected for the multiple-burn problem. If an optimal multiple-burn guidance scheme is developed, it will likely lead to the development of sufficient conditions for the optimality of multiple-burn transfers.

Appendix

ORBPACK Users Manual

A Package of FORTRAN Programs to Construct Guesses and Solve Low- and Medium- Thrust Optimal Orbit Transfer Problems

Applied Control Laboratory
Georgia Institute of Technology

Table of Contents

I. Introduction 1
II. Orbit Transfer Problem Definition 2
    II.1. Parameters 2
    II.2. Scaling 3
III. Making Guesses for the Optimal Transfer 4
    III.1. GSHOOT Random Guess (Single Burn Only) 4
    III.2. PAT2D Sub-Optimal Transfer Guess (Multiple Burn Only) 6
IV. The Modified Patched Method (MPMM2D, MPMM3D) 7
    IV.1. Using MPMM2D to Compute Solutions 7
    IV.2. The Structure of the MPMM2D (MPMM3D) Code 9
V. The Patched Method in Two Dimensions (PAT2D) 11
    V.1. Using PAT2D to Compute Sub-Optimal/Extremal Solutions 11
    V.2. How PAT2D Works 11
VI. The Multiple Shooting Approach (BND3D) 15
    VI.1. Using BND3D to Compute Solutions 15
    VI.2. The BND3D Guess File Format 18
    VI.3. How BND3D Works 19
VII. The Minimizing Boundary Condition Method (MBCM3D) 21
    VII.1. Using MBCM3D to Compute Solutions 21
    VII.2. How MBCM3D Works 22
VIII. Tutorials 23
    VII.1. Planar Five Burn Transfer 23
    VII.2. Convert MPMM3D File to BND3D File, Run BND3D 28
    VII.3. Run BND3D with Homotopy 31
Appendix A GSHOOT's File Format 35
Appendix B The PAT2D and PAT3D File Formats 36

II.2. Scaling

It is very useful for numerical methods to work with numbers that are at or near the same order. This can be accomplished through nondimensionalizations. Such nondimensionalizations for the orbit transfer problem follow:

$$\hat{\mathbf{r}} = \mathbf{r}/r^{\star}$$

$$\hat{m} = m/m^{\star}$$

$$\hat{t} = t/\sqrt{r^{\star 3}/\mu}$$

and they require the following:

$$\hat{\mathbf{v}} = \mathbf{v}/\sqrt{\mu/r^{\star}}$$

$$\hat{t}_f = t_f/\sqrt{r^{\star 3}/\mu}$$

$$\hat{r}_o = r_o/r^{\star}$$

$$\left(\hat{\rho}_o \hat{S}\hat{C}_D\right) = \rho_o S C_D\left(r^{\star}/m^{\star}\right)$$

$$\hat{T} = \left(T/m^{\star}\right)/\left(\mu/r^{\star 2}\right)$$

$$\hat{\beta} = \beta r^{\star}$$

$$\left(\hat{g}_o \hat{I}_{sp}\right) = g_o I_{sp}\sqrt{r^{\star}/\mu}$$

$$\hat{R}_e = R_e/r^{\star}$$

Note that these nondimensionalizations result in dynamics with μ=1. The choices of $r^{\star}$ and $m^{\star}$ are completely arbitrary. A choice for $m^{\star}$ might be one such that the initial nondimensionalized mass is 1 or 10. A choice for $r^{\star}$ might be the radius of the planet or a number such that the initial semimajor axis, radius of perigee , or an "average" radius is 1.

III. Making Guesses for the Optimal Transfer

There are many different ways that one could conceive of to make guesses. The routines for making guesses, listed below, have been provided.

The tutorials in Chapter VIII demonstrate how to make guesses with these methods.

III.1. *GSHOOT* Random Guess (Single Burn Only)

The subroutine GSHOOT will randomly make guesses for the one-burn orbit transfer problem in two dimensions. Input for GSHOOT is a text file. Its output consists of two text files which represent data for direct and indirect methods.

How to use *GSHOOT*

GSHOOT requires a file, named "GINPUT," for input. A typical "GINPUT" file follows:

```
MU     = 1.00
GO     = 1.00
ISP    = 0.5673
THRUST = 0.5166
MO     = 10.0000
AO     = 1.00000
EO     = 0.000
WO     = 0.000
AF     = 1.285
EF     = 0.219
WF     = 0.000
TMAX   = 0.000
NGS    = 100
NIX    = 3
```

where MU (μ) is the gravitational constant, GO (g_o) is the gravitational acceleration of the earth at sea level, ISP (I_{sp}) is the motor's specific impulse, and Thrust is the motor's thrust level. MO (m_o) is the initial mass for the transfer. The next parameters specify the terminal orbits: AO (a_o) is the initial orbit's semimajor axis, EO (e_o) the initial orbit's eccentricity, and WO (ω_o) is the initial orbit's argument of perigee; AF (a_f), EF (e_f), and (ω_f) are the corresponding parameters for the final orbit. TMAX is the maximum burn time; if it is set to zero, then TMAX is assigned by GSHOOT to the amount of time required for the mass to vanish. NGS is how many guesses to make; half of these will be almost tangential thrusting with random initial true anomaly and the other half will have random initial direction and random initial true anomaly. For a detailed description of the file format, see Appendix A.

GSHOOT will create output files "DIRECT.DAT" and "INDIRECT.DAT" which can be used to construct a multiple burn guess in the PATCH2D file format. Both of these files have identical headers:

I. Introduction

ORBPACK is a collection of FORTRAN 77 programs for computing optimal orbit transfers. For the most part, these are all indirect methods; they are concerned with solving the Two Point Boundary Value Problem provided by optimal control theory.

None of these routines guarantee a globally optimum solution; only extremal solutions are claimed by convergence of iterations. With the exception of MBCM, solutions obtained with these methods must have their switching law checked. One must be sure that, in the computed solution, the thrust is on when the switching function is positive and the thrust is off when the switching function is negative. Furthermore, these methods assume that no intermediate thrust arcs will be found in the solution.

The charts below summarizes the programs in ORBPACK:

Solvers

Name	Method	Libraries	Suggested Use
BND3D	Multiple Shooting (BNDSCO)	BNDSCO	medium/low thrust; few burns
MBCM3D	Shooting w/ Minimizing Boundary Condition Method	VF02AD	medium/low thrust; few burns
PAT2D	Patched Method	BNDSCO; IMSL	medium/low thrust
MPMM2D, MPMM3D	Modified Patched Method	IMSL; ODEPACK	medium/low thrust; short burns

Accessories

Name	Use	Libraries
GSHOOT	random shooting for one-burn guesses	IMSL; ODEPACK
MPM2D3D	convert MPMM2D files to MPMM3D files	N/A
MP2BND	convert MPMM3D files to BND3D files	ODEPACK
BND2MBCM	convert BND3D files to MBCM files	N/A

All codes as supplied in ORBPACK solve multiple burn orbit transfers with free final time and free initial and final points. BND3D is already configured so to switch between free and fixed final time problems. MBCM3D can easily be reconfigured for such. PAT2D, MPMM2D, and MPMM3D have fixed configurations.

PAT2D, MPMM2D, and MPMM3D are also fixed to solve only problems where ideal gravity is assumed. BND3D and MBCM3D are configured to solve problems that include drag and oblateness effects. Finally, codes with the "2D" suffix are configured to solve planar transfers; the "3D" suffix indicates that the code is configured for nonplanar transfers.

II. Orbit Transfer Problem Definition

II.1. Parameters

All the programs in ORBPACK require the following orbit transfer parameters to be determined:

For the gravitating body:
- the gravitational constant for the central body (μ)

For the rocket motor:
- maximum thrust
- specific impulse (I_{sp})

For the terminal orbits, BND3D and MBCM3D require:
- semimajor axis
- eccentricity
- right ascension (degrees)
- argument of perigee (degrees)
- inclination (degrees)

For the terminal orbits, MPMM2D, MPMM3D, and PAT2D require:
- angular momentum vector (X, Y, Z components)
- eccentricity vector (X, Y components)

Each program also requires a value for Earth's acceleration at sea-level (g_0) in appropriate units; this number is only used in conjunction with the specific impulse to compute the fuel consumption.

BND3D and MBCM3D can account for oblateness and drag effects. For oblateness: R_e is the equatorial radius of the central body and J_2 is a constant describing the mass distribution of the central body; for Earth J_2=1082.61x10^{-6}. For drag: β is a constant from the atmosphere model describing air density variation in the prescribed altitude region, ρ_0 is the atmosphere density at the altitude r_0, S is the cross-sectional area of the craft, and C_D is the craft's drag coefficient.

The gravitational potential, including oblateness, is modeled as:

$$U = \frac{\mu m}{r} + \frac{1}{2} J_2 R_e^2 \frac{\mu m}{r^3}\left(1 - 3\cos^2(\theta)\right)$$

where r is the magnitude of the position vector $\mathbf{r}$. The drag force is modeled as:

$$\mathbf{F}_{drag} = \frac{1}{2}\rho_0 e^{-\beta(r-r_0)} S C_D v \mathbf{v}$$

where v is the magnitude of the velocity $\mathbf{v}$. Note that this form for the density variation indicates an isothermal region of the atmosphere.

```
T    =  0
GO   =  0
ISP  =  0
AO   =  0
EXO  =  0
EYO  =  0
AF   =  0
EXF  =  0
EYF  =  0
```

These output files contain the necessary information

If this output file represents a guess for any but the last burn, delete the last three of these lines (AF, EXF, EYF) when constructing the multiple-burn guess file. However, if this guess is for the last burn, keep the last three lines and delete lines six through eight (AO, EXO, EYO). If the guess is any but the first burn, then delete the first three lines (T, GO, ISP).

How *GSHOOT* works

GSHOOT makes a random guess by choosing the constant Lagrange multipliers (ν) as a random vector with unity magnitude. Since all the Lagrange multipliers may be scaled by an arbitrary constant, there is no loss of generality. The state vector is computed knowing the initial orbital elements and randomly choosing the initial true anomaly. Next, the vectors λ_r and λ_v are calculated for the initial time, using the following equation:

$$\begin{bmatrix}\lambda_r(t_o)\\ \lambda_v(t_o)\end{bmatrix} = \left[\frac{\partial\psi}{\partial\mathbf{x}}\left(\mathbf{x}(t_o)\right)\right]^T \nu \tag{3.1}$$

The initial value for λ_m is found by specifying that the switching function is zero at the initial time:

$$\lambda_m(t_o) = \left(g_o I_{sp}\right)\frac{\left|\lambda_v(t_o)\right|}{m(t_o)} \tag{3.2}$$

That the switching function is zero at the initial time is known to be true for the free transfer time and free terminal points problem. With the initial state and costate known, the initial value problem is integrated forward in time until either the desired final semimajor axis (AD) is reached, the current radius becomes small, the spacecraft enters a parabolic orbit, or the mass becomes small.

For guesses that are almost tangential, λ_v is chosen to be (+/-) $\mathbf{v}$ and λ_r is chosen to be (+/-) $(\mu/r^3)\mathbf{r}$. The positive sign usually produces orbit raising and the negative sign orbit lowering. Note that this initial guess for the costates zeros the Hamiltonian when the switching function is zero. Therefore, the ν_i's can be found by solving the least-squares problem of Eq. (3.1).

GSHOOT will try as many guesses as the user requests. The guess that best meets the required boundary conditions will be output.

III.2. PAT2D Sub-Optimal Transfer Guess (Multiple Burn Only)

PAT2D creates sub-optimal trajectories in the sense that the choice of intermediate transfer orbits has been fixed and each burn is an optimal one-burn orbit transfer. PAT2D iterates upon the choice of intermediate transfer orbits until it finds a choice that gives a local maximum in final mass The PAT2D program is described in detail in Chapter V.

Using PAT2D to Compute Guesses

PAT2D requires two files for input. The first file, "PATCH2D.TOLS," sets accuracy levels and limits the number of iterations (for more information on this file, see Chapter V). The second file, "PATCH2D.GUESS," supplies the guess information for both the choice of intermediate transfer orbits and the trajectories of the burn arcs between them. This latter file must be in the PAT2D format (for more information, see Appendix A and Chapter V).

The guess information from GSHOOT, or some other source, must be put into the PAT2D format. When run, the first thing that PAT2D will do is solve the one-burn problems defined by the intermediate transfer orbits. Often, the output from this step alone is a sufficiently good solution guess . This output is contained in the file "PATCH2D.INITIAL."

On the other hand, it is not uncommon for that output to be an insufficient guess. In this case, one approach is to allow PAT2D to iterate. At some point during the iteration, the user may take the file "PATCH2D.BEST" and use it as an initial solution guess. Alternatively, the user may set a rather loose stopping criterion for PAT2D and wait until this criterion is met. In this approach, the file "PATCH2D.SOL" will be the solution guess.

IV. The Modified Patched Method (MPMM2D, MPMM3D)

The subroutine MPM2D (MPM3D) is a realization of the Modified Patched Method in two (three) dimensions. The file "MPMM2D.f" (MPMM3D.f) contains an implementation of MPM2D (MPM3D) using IMSL's NEQNF to solve the nonlinear equations, its FORTRAN program name is MPMM2D (MPMM3D).

IV.1. Using MPMM2D to Compute Solutions

MPMM2D (MPMM3D) requires only one input file, which must follow the PAT2D (PAT3D) format (see Appendix A). This data file must be named "MPM2D.GUESS" ("MPM3D.GUESS")

The code "MPM2D3D.f" will convert an "MPM2D.GUESS" file into a "MPM3D.GUESS" file. In this code, no other input is required except "MPM2D.GUESS"

Data File (Input)

In "MPM2D.GUESS," ("MPM3D.GUESS") the tolerance setting (TOL) is the root-finding tolerance. The tolerance used in numerical integration is one-thousandth of this number. No information in the header is ignored.

For MPMM2D (MPMM3D), the option SEL may only be chosen as 1 or 2. These options indicate the data for the burn is given in the format for an indirect method. MPMM2D (MPMM3D) will treat both SEL=1 and SEL=2 identically.

MPMM2D (MPMM3D) only uses specific items from the PAT2D file format. The lines below are representative of the data for one burn in the PAT2D format. The underlined "#" symbols indicate which number items are important to MPM2D calculations.

MPMM2D

```
a     = #
ex    = #
ey    = #
NODE =  3
SEL   =  1
index,x,y,u,v,m,lx,ly,lu,lv,lm,tf,g1,g2,g3,g4,g5,g6
  1, #, #, *, *, #, *, *, *, *, #, #, *, *, *, #, #, #,
  2, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *,
  3, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *,
  4, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *,
```

MPMM3D

```
hx    = #
hy    = #
hz    = #
ex    = #
ey    = #
NODE =  3
SEL   =  1
INDEX,X,Y,Z,U,V,W,M,LX,LY,LZ,LU,LV,LW,LM,TF,G1,G2,G3,G4,G5,G6,G7,G8,G9,G10
  1, #, #, #, *, *, *, #, *, *, *, *, *, *, #, #, *, *, *, *, *, #, #, #, #
  2, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *
  3, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *
  4, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *, *
```

The important number items are "a," "ex," and "ey" with "x," "y," "m," "tf," "g4," "g5," and "g6" on the first line only. All other numbers are read by the program but not used. The "x" and "y" coordinates are used only to compute the true anomaly angle that the burn begins at. The only mass value remembered is the initial mass value. The mass costate is used to scale the constant Lagrange multipliers "g4," "g5," and "g6" in a manner consistent with patching the burns together; otherwise, it is not used.

MPM2D Iteration Info to Screen

Listed below is sample screen output from "MPMM2D.F"

```
  Cur. Norm     It#    Best Norm (at) #    Short Time    Bn#   Bst Wrst El.   El#
 ------------ ------ ------------ ------ ------------ ------ ------------ ------
 0.45051E+00      1  0.45051E+00      1  0.11289E+01      4  0.30952E+00     18
 0.45051E+00     45  0.45051E+00     15  0.11289E+01      4  0.30952E+00     18
 0.63552E-02     90  0.63550E-02     89  0.10551E+01      4  0.28471E-02     14
 0.48575E-02    135  0.48575E-02    105  0.10606E+01      4  0.24346E-02     14
 ---------
 Required # Function Evals = 172
 ------------------------------
  Total Burn Time = 6.514028424486
  Final Mass = 4.066434637841
  Shortest Burn Length = 1.126863878329
  Shortest Burn is #4
 ------------------------------
```

The first block of text is the *iteration table*. The column "Cur. Norm" shows the current 2-norm of the constraint errors in the absolute sense. The iteration, or number of times called, at which this value was computed is listed in column "It#." The lowest norm of constraint errors yet computed, next to the iteration number it was computed at is given under the "Best Norm (at) #" column. The length of the shortest burn at the current iteration is under "Short Time" and the burn with this length is indicated under the "Bn#" column. Finally, the largest absolute value of a constraint component for the best norm is listed under "Bst Wrst El." with "El#" listing which constraint component this is.

MPM3D Iteration Info to Screen

The iteration table from MPM3D is slightly different. It has the following header:

```
 CUR. NORM    IT#   BEST NORM (AT) #   WRST C. EL.    EL#  BST WRST EL.   EL#
 ------------ ----- ------------ ----- ------------ ----- ------------ -----
```

where "WRST C. EL." indicates the worst element of the current iteration constraint error vector.

For MPMM2D and MPMM3D, below the iteration table is the number of function calls required to reach an error level indicated by the tolerance. After this, some statistics of the solution are given. The "Total Burn Time" is the total amount of time the motor is on. The "Final Mass" is the mass of the spacecraft at the end of the transfer. The "Shortest Burn Length" is length in time of the quickest burn. Finally, the burn number for this quickest, or shortest, burn is listed.

Data File Output

The subroutine MPM2D (MPM3D), if desired, creates an output file that gives the status of iterations. The file is named "MPM2D.ISTAT" ("MPM3D.ISTAT"). This file is useful for computer systems that operate under a queuing system because such a system often does not show output to the screen until after execution is completed. However, such queuing systems usually allow files that are created and closed to appear in the users directory. Therefore, during execution under a queuing system, the user may list the contents of "MPM2D.ISTAT" ("MPM3D.ISTAT") and see current iteration information. The content of "MPM2D.ISTAT" ("MPM3D.ISTAT") is three lines long: the first two lines are the table headings from the iteration table, the third line is the current entry in the iteration table.

Both the main routine MPMM2D (MPMM3D) and MPM2D (MPM3D) contribute to a file named "MPM2D.REPORT" ("MPM3D.REPORT"). The first lines in this file gives feedback from MPMM2D (MPMM3D) while reading "MPM2D.GUESS" ("MPM3D.GUESS") so that any errors in that file may be easily identified.

The first eleven lines give the header parameters. At the beginning of each line, the text from "MPM2D.GUESS" ("MPM3D.GUESS") is given, then the number read from that line, and finally, in parentheses, the name of the variable which MPMM2D (MPMM3D) has assigned this number to. This same pattern is continued as MPMM2D (MPMM3D) reads the orbital elements of the transfer orbits.

The twelfth line and lines below are printed as each line of the input are read. Following this is a listing of the values of each variable used by MPM2D (MPM3D) for the first iteration; then a listing of the constraint values when given these variables.

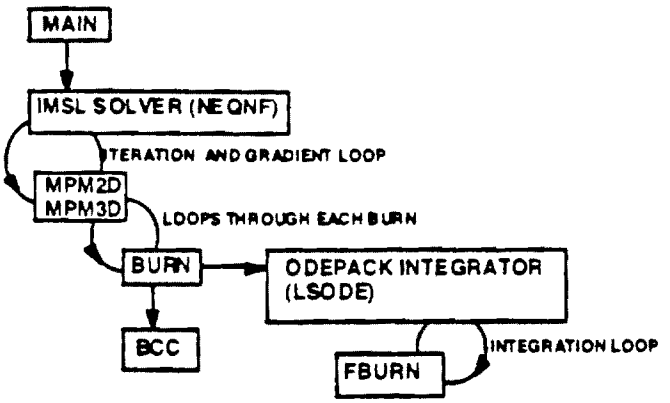
Next is the iteration table as printed to the screen. Following this, a total number of calls to MPM2D (MPM3D). Then a listing of variables and constraint evaluations for the solution. Finally, at the bottom of the file is the solution summary statistics just as printed to the screen.

The other file created by MPMM2D (MPMM3D) is "MPM2D.SOL" ("MPM3D.SOL"), the solution file. This file contains the solution to the orbit transfer problem in the PAT2D (PAT3D) format.

IV.2. The Structure of the MPMM2D (MPMM3D) Code

The structure of the MPMM2D (MPMM3D) program is generalized in the following diagram, not intended as a formal flow chart:

MPMM2D/MPMM3D Diagram



The main routine, calls the multidimensional nonlinear equation solver, IMSL's NEQNF, with the guess from "MPM2D.GUESS" ("MPM3D.GUESS") The solver calls MPM2D (MPM3D) iteratively to solve the problem and to numerically compute partial derivatives. This recurrent use of MPM2D (MPM3D) is illustrated in the diagram by a loop with an arrow on it, connecting the two blocks.

MPM2D (MPM3D) evaluates the MPM conditions given the variables. For each burn in the orbit transfer problem, variables are sent to BURN. This subroutine integrates each burn arc by calling LSODE and evaluates boundary conditions for that burn by calling BCC (BCC). The derivatives for integration, required for LSODE, are supplied by FBURN. FBURN is called repeatedly by LSODE during solution of each burn's initial value problem.

V. The Patched Method in Two Dimensions (PAT2D)

The subroutine FUNC is a realization of the Patched Method in two dimensions. The file "PAT2D.f" contains an implementation of FUNC with the conjugate gradient method. The conjugate gradient algorithm was taken from "Numerical Recipes" and is only slightly modified from what is presented there.

V.1. Using PAT2D to Compute Sub-Optimal/Extremal Solutions

PAT2D requires two input files for execution. These files specify iteration parameters ("PATCH2D.TOLS") and the initial solution guess ("PATCH2D.GUESS"). The "PATCH2D.GUESS" file must be in the PAT2D format (see Appendix A). The format for "PATCH2D.TOLS" is much simpler and demonstrated in the example below:

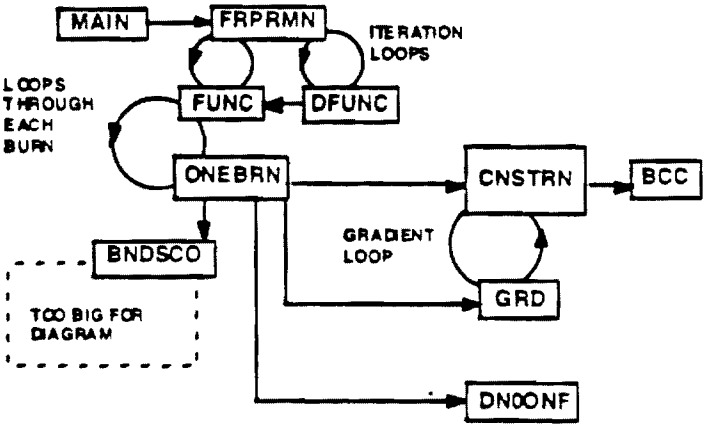
```
FTOL = 1.00000000000000000000E-06
LTOL = 1.00000000000000000000E-07
GTOL = 1.00000000000000000000E-03
TOL2 = 1.00000000000000000000E-05
ITMX = 200
MFUN = 200
MITN = 1000
ITMB = 15
```

The FORMAT edit descriptors for the first four lines, containing REAL values, are (1X,A6,D27.20) and likewise for the last four lines, containing INTEGER values, (1X,A6,I6). The value for FTOL specifies the function value stopping criterion, when the change in total burn time after a line search is less than FTOL the iteration stops. The value for LTOL is the line search tolerance. GTOL specifies how small the 2-norm of the gradient should be fore stopping. TOL2 is the tolerance for DCNLP one-burn solutions. ITMX is the maximum number of allowed conjugate gradient iterations. MFUN limits function calls and MITN limits the overall iteration count for DCNLP. ITNB limits the number of multiple-shooting iterations performed by BOUNDSCO.

V.2. How PAT2D Works

The diagram below shows the general structure of the code in the file "PAT2D.f."

PAT2D Diagram



The subroutine FUNC is the heart of PAT2D. This is the function that, given the choice of intermediate orbital elements, calculates the total burn time for the transfer. FRPRMN is the conjugate gradient routine, from "Numerical Recipes," that iteratively calls FUNC and DFUNC (gradient routine) to find the optimal choice of intermediate transfer orbits.

PAT2D has a two-loop structure; there is an inner loop (FUNC/ONEBRN) and an outer loop (FRPRMN). The outer loop successively changes the transfer orbits until a minimum is found in the total burn time (maximum of final mass). The inner loop solves the one burn trajectories between each transfer orbit. Solving this trajectories yields the burn time s for each intermediate transfer. These burn times are summed, giving the output of FUNC.

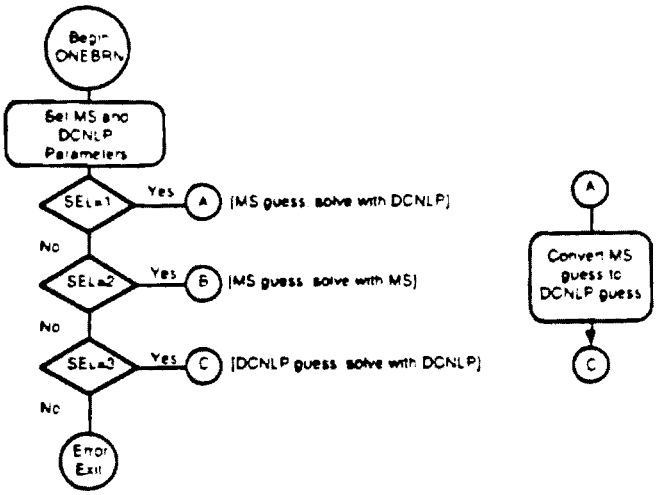
Note that each successful outer loop iteration produces a suboptimal transfer. This transfer satisfies all the conditions on the state but is not an extremal transfer.

The main routine loads the solution guess and calls FUNC once, before FRPRMN does. This is done because there is no assurance that the trajectory guesses in the PATCH2D.GUESS file will successfully produce a suboptimal solution. The output from this first call is named "PATCH2D.INITIAL" and is often a good guess for MPMM2D. However, if this is a poor guess, then a good strategy is to allow PAT2D several iterations to produce a transfer closer to the solution.

The inner loop iterations are a little complicated. This is the result of an attempt to make them robust. It is also designed so that each successful inner loop iteration produces a solution to the Two Point Boundary Value Problem (TPBVP) with BOUNDSCO, a multiple-point shooting algorithm (MS). However, it is widely known that direct methods often have a large region of convergence than indirect methods. Therefore, Direct Collocation with Nonlinear Programming (DCNLP) has also been implemented.

The following diagram shows how the ONEBRN subroutine interprets the user's selection as to what is the appropriate first action, use MS or DCNLP first?

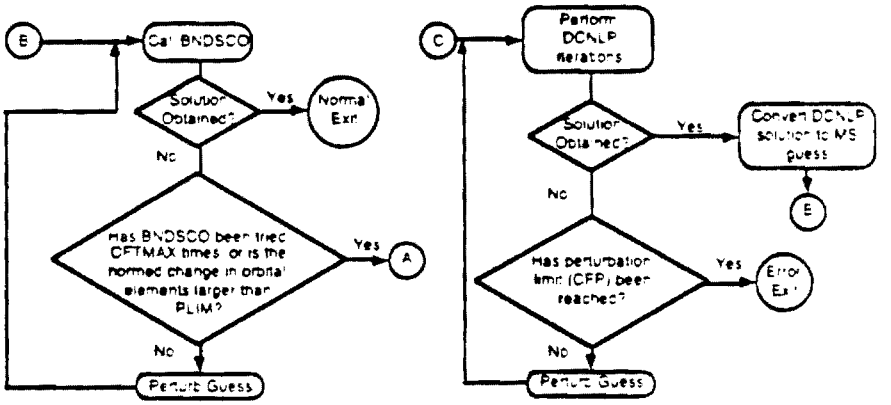
ONEBRN Flow Chart part 1 (abridged)



Note that a MS guess can be given for DCNLP in this structure. A DCNLP guess cannot be given for MS because a DCNLP *solution* is required in the conversion process from DCNLP information to MS information.

The next diagrams shows how MS (BNDSCO) and DCNLP (IMSL's DN0ONF) are incorporated:

ONEBRN Flow Chart part 2 (abridged)



Attempts with either method have a similar structure. If a failure in iterations occurs, the guess is perturbed and the method attempted again. After each failure, the perturbation size is increased. If MS fails too many times, control is handed over to DCNLP. However, if DCNLP fails too many times there is no backup and an error exit occurs.

After ONEBRN succeeds in computing a MS solution, the SEL parameter is set to 2 for that burn.

Output files:

- "PATCH2D.HIST" (iteration data)
- "PATCH2D.INITIAL" (first suboptimal sol) first optimal solution obtained, in patch2d format
- "PATCH2D.SOL" contains the extremal solution obtained to tolerance
- "PATCH2D.BURN" (iteration status) prints iteration status; file is useful when program is being run under a queuing system and screen output is withheld. Printed after a burn is solved.
- "PATCH2D.COST" (iteration status); file is useful when program is being run under a queuing sys and screen output is withheld. Printed after a complete transfer is solved.
- "PATCH2D.CURRENT" contains current suboptimal trajectory, unless it is the best.
- "PATCH2D.BEST" contains best suboptimal trajectory to date
- "PATCH2D.PERT" gives information as to the progress of solving the current burn.
- "FRPRMN.OUT" output from conjugate gradient routine, FRPRMN
- "FRPRMN.ITERATES" current output from FRPRMN, for info when using a queuing system

VI. The Multiple Shooting Approach (BND3D)

The BND3D program implements the modified multiple-point (MS) algorithm of BOUNDSCO (**Boundary** value problem solver with **S**witching **Co**nditions). BOUNDSCO makes use of Newton's method, a Broyden update, and Deuflhard's relaxation strategy. One should refer to the BOUNDSCO manual[1] for detailed information on BOUNDSCO. Note that BOUNDSCO does not make use of an analytical gradient.

BND3D also has a homotopy loop around BNDSCO. A homotopy variable U is defined such that, as the loop repeats, U will change from 1 to UMIN (The choice of UMIN is set by the user, but usually is chosen as 0). Certain parameters for the orbit transfer problem definition are included in the homotopy loop and vary as the value of U changes. A tutorial using homotopy is included in the Tutorials section.

The code MP2BND will convert MPMM3D input files into BND3D input files.

VI.1. Using BND3D to Compute Solutions

BND3D requires two input files: "BND3D.SCRIPT" which contains instructions and parameters, and another file (named by user) which contains the solution guess.

The format of the file "BND3D.SCRIPT" depends on how BND3D is to be used. This format is best described line-by-line. The character in the first column of each line is ignored.

The four different layouts of the "BND3D.SCRIPT" file are described below:

Normal Execution: Free Final Time, No Homotopy

- *Line 1*: (1X,A28) On this line, the name of the file containing the solution guess is specified. No more than 28 characters are allowed.
- *Line 2*: (1X,I6) Here, a "1" indicates that boundary condition errors should be displayed to the screen, in addition to the normal BNDSCO iteration output; a "0" indicates otherwise. Usually, one would place a "0" here; this output is usually only useful in finding errors in the input file.
- *Line 3*: (1X,I6) A **"1"** on this lines chooses the free final time option.
- *Line 4*: (1X,I6) A **"0"** deselects the homotopy option.
- *Line 5*: (1X,I6) A "1" on this line tells BNDSCO to insert nodes for the switching times in the output; a "0" says not to.

[1]Oberle, H.J, Grimm, W., "BNDSCO: A Program for the Numerical Solution of Optimal Control Problems," English Translation of DFVLR-Mitt. 85-05.

- *Line 6*: (A,D12.5) The value on this line sets the BNDSCO parameter FCMIN. FCMIN is the lower limit of the relaxation factor.
- *Line 7*: (A,D12.5) The value on this line sets the BNDSCO iteration tolerance.
- *Line 8*: (1X,I4) The maximum number of iterations.
- *Line 9*: (1X,A28) The name for the file containing the solution
- *Line 10*: (1X,I6) A "1" on this line requests detailed solution information ("BND3D.EXTRA" and the file named on the next line). A "0" indicates otherwise.
- *Line 11*: (1X,A28) The file name for additional information (if a "1" on the previous line).

Fixed Final Time; No Homotopy

- *Line 1*: (1X,A28) On this line, the name of the file containing the solution guess is specified. No more than 28 characters are allowed.
- *Line 2*: (1X,I6) Here, a "1" indicates that boundary condition errors should be displayed to the screen, in addition to the normal BNDSCO iteration output; a "0" indicates otherwise. Usually, one would place a "0" here; this output is usually only useful in finding errors in the input file.
- *Line 3*: (1X,I6) A **"0"** on this lines chooses the fixed final time option.
- *Line 4*: (A,D12.5) The value for the final time.
- *Line 5*: (1X,I6) A **"0"** deselects the homotopy option.
- *Line 6*: (1X,I6) A "1" on this line tells BNDSCO to insert nodes for the switching times in the output; a "0" says not to.
- *Line 7*: (A,D12.5) The value on this line sets the BNDSCO parameter FCMIN. FCMIN is the lower limit of the relaxation factor.
- *Line 8*: (A,D12.5) The value on this line sets the BNDSCO iteration tolerance.
- *Line 9*: (1X,I4) The maximum number of iterations.
- *Line 10*: (1X,A28) The name for the file containing the solution
- *Line 11*: (1X,I6) A "1" on this line requests detailed solution information ("BND3D.EXTRA" and the file named on the next line). A "0" indicates otherwise.
- *Line 12*: (1X,A28) The file name for additional information (if a "1" on the previous line).

Free Final Time, Homotopy Activated

- *Line 1*: (1X,A28) On this line, the name of the file containing the solution guess is specified. No more than 28 characters are allowed.
- *Line 2*: (1X,I6) Here, a "1" indicates that boundary condition errors should be displayed to the screen, in addition to the normal BNDSCO iteration output; a "0" indicates otherwise. Usually, one would place a "0" here; this output is usually only useful in finding errors in the input file.
- *Line 3*: (1X,I6) A **"1"** on this lines chooses the free final time option.

- *Line 4*: (1X,I6) A "1" selects the homotopy option.
- *Line 5*: (1X,I6) the suggested number of homotopy loops to perform
- *Line 6*: (*) Enter UMIN, the value of the homotopy variable to stop at. The homotopy variable, U, starts at 1 and ends at UMIN. Enter "0.0" here to attempt to achieve the values below.
- *Line 7*: (*) Enter the desired maximum thrust level
- *Line 8*: (*) Enter the desired specific impulse
- *Line 9*: (*) Enter the desired final orbit semimajor axis
- *Line 10*: (*) Enter the desired final orbit eccentricity
- *Line 11*: (*) Enter the desired final orbit argument of perigee
- *Line 12*: (*) Enter the desired initial orbit semimajor axis
- *Line 13*: (*) Enter the desired initial orbit eccentricity
- *Line 14*: (*) Enter the desired initial orbit argument of perigee
- *Line 15*. (*) Enter the desired initial orbit argument inclination
- *Line 16*: (1X,I6) A "1" on this line tells BNDSCO to insert nodes for the switching times in the output; a "0" says not to.
- *Line 17*: (A,D12.5) The value on this line sets the BNDSCO parameter FCMIN. FCMIN is the lower limit of the relaxation factor.
- *Line 18*: (A,D12.5) The value on this line sets the BNDSCO iteration tolerance.
- *Line 19*: (1X,I4) The maximum number of iterations.
- *Line 20*: (1X,A28) The name for the file containing the solution
- *Line 21*: (1X,I6) A "1" on this line requests detailed solution information ("BND3D.EXTRA" and the file named on the next line). A "0" indicates otherwise.
- *Line 22*: (1X,A28) The file name for additional information (if a "1" on the previous line).

Fixed Final Time, Homotopy Activated (in this case, the fixed final time is also achieved through the homotopy loop)

- *Line 1*: (1X,A28) On this line, the name of the file containing the solution guess is specified. No more than 28 characters are allowed.
- *Line 2*: (1X,I6) Here, a "1" indicates that boundary condition errors should be displayed to the screen, in addition to the normal BNDSCO iteration output; a "0" indicates otherwise. Usually, one would place a "0" here: this output is usually only useful in finding errors in the input file.
- *Line 3*: (1X,I6) A "0" on this lines chooses the fixed final time option.
- *Line 4*: (A,D12.5) The value for the final time.
- *Line 5*: (1X,I6) A "1" selects the homotopy option.
- *Line 6*: (1X,I6) the suggested number of homotopy loops to perform
- *Line 7*: (*) Enter UMIN, the value of the homotopy variable to stop at. The homotopy variable, U, starts at 1 and ends at UMIN. Enter "0.0" here to attempt to achieve the values below.
- *Line 8*: (*) Enter the desired maximum thrust level
- *Line 9*: (*) Enter the desired specific impulse
- *Line 10*: (*) Enter the desired final orbit semimajor axis

- *Line 11*: (*) Enter the desired final orbit eccentricity
- *Line 12*: (*) Enter the desired final orbit argument of perigee
- *Line 13*: (*) Enter the desired initial orbit semimajor axis
- *Line 14*: (*) Enter the desired initial orbit eccentricity
- *Line 15*: (*) Enter the desired initial orbit argument of perigee
- *Line 16*: (*) Enter the desired initial orbit argument inclination
- *Line 17*: (1X,I6) A "1" on this line tells BNDSCO to insert nodes for the switching times in the output; a "0" says not to.
- *Line 18*: (A,D12.5) The value on this line sets the BNDSCO parameter FCMIN. FCMIN is the lower limit of the relaxation factor.
- *Line 19*: (A,D12.5) The value on this line sets the BNDSCO iteration tolerance.
- *Line 20*: (1X,I4) The maximum number of iterations.
- *Line 21*: (1X,A28) The name for the file containing the solution
- *Line 22*: (1X,I6) A "1" on this line requests detailed solution information ("BND3D.EXTRA" and the file named on the next line). A "0" indicates otherwise.
- *Line 23*: (1X,A28) The file name for additional information (if a "1" on the previous line).

VI.2. The BND3D Guess File Format

The BND3D Guess file (named in "BND3D.SCRIPT") has a specific format. The first twenty lines specify orbit transfer parameters of type DOUBLE PRECISION and have FORMAT edit descriptors (1X,A9,F30.15). These parameters are as follows and in this order:

MU	gravitational constant of the central body (1.0 for no dimensions)
REQ	equatorial radius of the central body
J2	constant describing the mass distribution of the central body; for Earth J_2=1082.61x10^{-6}
GO	acceleration at sea-level
BETA	constant from the atmosphere model describing air density variation in the prescribed altitude region
RO	r_o, +REQ
ROU	atmosphere density at the altitude r_o
S	cross-sectional area of the craft
CD	drag coefficient
ISP	specific impulse
THRUST	maximum thrust
AI	initial semimajor axis
EI	initial eccentricity
OMEGAI	initial argument of perigee (degrees)
RAI	initial right ascension (degrees)
I-I	initial inclination (degrees)
AF	final semimajor axis
EF	final eccentricity
OMEGAF	final argument of perigee (degrees)
RAF	final right ascension (degrees)
I-F	final inclination (degrees)

The 21st line (1X,I5) gives the number of intervals (# nodes - 1).
The next line is a dummy string line (1X,A) that, on output, is used to provide a header for the data in the following lines (useful in plotting results).
The next (# nodes) lines gives the BND3D state at each node with edit descriptors (1X,F30.15,25(A2,F30.15)). The BND3D state is as follows:

```
 0  1  2  3  4  5  6  7   8    9   10   11   12   13   14   15
(T, X, Y, Z, U, V, W, M, L-X, L-Y, L-Z, L-U, L-V, L-W, L-M, TF

 ( FINAL ORBIT    )  ( INITIAL ORBIT  )
 16  17  18  19  20  21  22  23  24  25
 G1, G2, G3, G4, G5, G6, G7, G8, G9, G10)

   <X,Y,Z> IS POSITION                <L-X,L-Y,L-Z> IS LAMEDA-P
   <U,V,W> IS VELOCITY                <L-U,L-V,L-W> IS LAMEDA-V
   M IS MASS
   L-M IS LAMEDA-M
   T IS THE NORMALIZED TIME [0,1]
```

Where TF is the final time and G# are components of the constant Lagrange multipliers (ν); G1-G5 being ν for the final boundary conditions and G6-G10 being ν for the initial boundary conditions.

The nodes are entered in the reverse order, starting with the final node and ending with the initial node.

Following the node information is a line (1X,I5) for the number of switching points. It is suggested to use an even number of switching points - this indicates to BNDSCO that the first and last intervals are burn arcs.

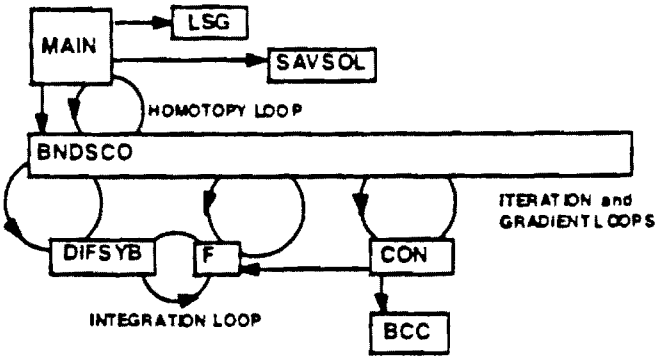
The next lines (1X,F30.15), one for each switching point, give the switching times in normalized time [0,1]. No lines after these are read.

VI.3. How BND3D Works

BND3D supplies the necessary routines (F and CON) to BNDSCO. "F" supplies the derivatives of the state and "CON" evaluates the boundary conditions. The routine "BCC" computes repeated formulas, "LSG" loads the solution guess, "SAVSOL" saves solution data in the same format as the guess data. The routine "DIFSYB" performs numerical integration.

The flow diagram below indicates the interdependence of the BND3D subroutines.

BND3D Flow Diagram



VII. The Minimizing Boundary Condition Method (MBCM3D)

The Minimizing Boundary Condition Method (MBCM) is a relaxed simple shooting algorithm. Instead of using a multidimensional nonlinear equation solver for the two point boundary value problem (TPBVP), it transforms the TPBVP into a nonlinear programming (NLP) problem.

As included in ORBPACK, MBCM3D uses the square of the Hamiltonian as the NLP cost function. All other boundary conditions are taken as NLP constraints.

VII.1. Using MBCM3D to Compute Solutions

MBCM3D requires one input file, MBCM3D.GUESS. This file has a very specific format. The first 47 lines of this file have the FORMAT edit descriptors (1X,A9,E30.15). They describe, in the following order:

MU	gravitational constant of the central body (1.0 for no dimensions)
REQ	equatorial radius of the central body
J2	constant describing the mass distribution of the central body; for Earth J_2=1082.61x10^{-6}
GO	acceleration at sea-level
BETA	constant from the atmosphere model describing air density variation in the prescribed altitude region
RO	r_o +REQ
ROU	atmosphere density at the altitude r_o
S	cross-sectional area of the craft
CD	drag coefficient
ISP	specific impulse
THRUST	maximum thrust
AI	initial semimajor axis
EI	initial eccentricity
OMEGAI	initial argument of perigee (degrees)
RAI	initial right ascension (degrees)
I-I	initial inclination (degrees)
AF	final semimajor axis
EF	final eccentricity
OMEGAF	final argument of perigee (degrees)
RAF	final right ascension (degrees)
I-F	final inclination (degrees)
[the next 14 lines give the initial state]	
TF	transfer time
[the next 10 lines give G1-G10]	
ACC	solution tolerance

Where G# are components of the constant Lagrange multipliers (ν); G1-G5 being ν for the final boundary conditions and G6-G10 being ν for the initial boundary conditions.

The last line of "MBCM3D.GUESS" (1X,A9,I10) gives the maximum number of iterations.

The code "BND2MBCM.f" will convert a BND3D guess file named "BND3D.GUESS" into a MBCM3D guess file ("MBCM3D.GUESS").

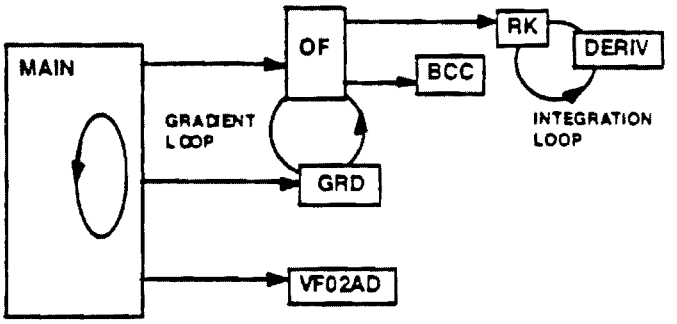
VII.2. How MBCM3D Works

MBCM3D uses VF02AD to solve the NLP problem. VF02AD uses reverse communication: the main routine calls OF to compute NLP cost and constraints given input; then GRD to compute gradients; then calls VF02AD to compute the new iterates. The main routine then uses these new iterates as input for OF and repeats the loop until VF02AD signals convergence.

OF evaluates the TPBVP as a NLP. The shooting problem is integrated with RK, a Runge-Kutta integration routine. Integration of the shooting problem is interrupted often to check the sign of the switching function. If a sign change is detected, the integration interval is adjusted until the exact switching point is located. During this process, OF keeps track of the sign of the switching function and appropriately adheres to the optimal switching law. This should ensure that the switching law is followed, however, it is always prudent to check the switching law after a solution is claimed.

The flow diagram below indicates the interdependence of the MBCM3D subroutines.

MBCM3D Flow Diagram



VIII. Tutorials

The following tutorials demonstrate some aspects of using ORBPACK that the user may commonly encounter.

VII.1. Planar Five Burn Transfer

This tutorial demonstrates the use of the supplied code in solving a planar transfer from a circular LEO to circular GEO. The initial radius is 6600 km, the final radius is 42241 km. The initial rocket motor thrust is 9.918 kN; its I_{sp} is 450 seconds. The initial mass is 20980 kg. A five burn solution is desired.

After nondimensionalization, these parameters are: initial mass=10, thrust=0.5166, go=1, Isp=0.5673, initial radius=1, final radius=6.4.

Based on the characteristics of these types of transfers, the following guess for the transfer orbits may seem reasonable:

a	e
1.285	0.2189
1.570	0.3584
1.856	0.4550
3.707	0.7262

All their apses are aligned and the final transfer orbit is similar to the Hohmann transfer orbit.

Use GSHOOT to Construct a Guess

The trajectory for each burn will now be guessed using GSHOOT. The "INDIRECT.DAT" files produced by GSHOOT will then be concatenated together to form an "MPM2D.guess" file. The first burn input file for GSHOOT ("GINPUT") is supplied as "Tutorials/2D 5burn/GSHOOT/burn 1/GINPUT" and listed below:

```
Mu     = 1.00
Go     = 1.00
Isp    = 0.5673
Thrust = 0.5166
Mo     = 10.0000
ao     = 1.00000
eo     = 0.000
wo     = 0.000
ad     = 1.2E5
ed     = 0.219
wd     = 0.000
TMAX   = 0.000
NGS    = 100
NIX    = 3
```

GSHOOT reports:

```
 Best constant Lagrange multipliers (initial)
   G... 0.15245E-00  0.98820E-00  0.14581E-01
 Best initial true anomaly
    vo= 0.53047E-01
 Best transfer time
    tf= 0.19312E-01
 Best relative errors (h,ex,ey,Hs)
   G... 0.18817E-08 -0.49820E-01  0.15555E-02  0.27599E-02
```

The resulting file has been supplied as "Tutorials/2D 5burn/GSHOOT/burn 1/INDIRECT.DAT" The second burn "GINPUT" is ["Tutorials/2D 5burn/GSHOOT/burn 2/"]:

```
Mu      = 1.00
Go      = 1.00
Isp     = 0.5673
Thrust  = 0.5166
Mo      = 10.0000
ao      = 1.285
eo      = 0.219
wo      = 0.000
ad      = 1.570
ed      = 0.3584
wd      = 0.000
TMAX    = 0.000
NGS     = 100
NIX     = 3
```

GSHOOT reports:

```
 Best constant Lagrange multipliers (initial)
   C... 0.70359E-00  0.17901E-00 -0.24402E-14
 Best initial true anomaly
    vo= 0.56451E+01
 Best transfer time
    tf= 0.11458E-01
 Best relative errors (h,ex,ey,Hs)
   G... 0.10846E-07 -0.82645E-02 -0.16307E-02  0.32135E-03
```

The resulting file has been supplied as "Tutorials/2D 5burn/GSHOOT/burn 2/INDIRECT.DAT" The third burn "" is ["Tutorials/2D 5burn/GSHOOT/burn 3/"]:

```
Mu      = 1.00
Go      = 1.00
Isp     = 0.5673
Thrust  = 0.5166
Mo      = 10.0000
a0      = 1.570
e0      = 0.3584
w0      = 0.000
ad      = 1.856
ed      = 0.4550
wd      = 0.000
TMAX    = 0.000
NGS     = 100
NIX     = 3
```

GSHOOT reports:

```
 Best constant Lagrange multipliers (initial)
   C... 0.54451E-00  0.26192E-00 -0.10330E-14
 Best initial true anomaly
    vo= 0.60064E+01
 Best transfer time
    tf= 0.79429E-00
 Best relative errors (h,ex,ey,Hs)
   G... 0.92974E-08  0.48454E-02  0.13268E-01 -0.35436E-02
```

The resulting file has been supplied as "Tutorials/2D 5burn/GSHOOT/burn3/" The fourth burn "" ["Tutorials/2D 5burn/GSHOOT/burn 4/"]:

```
mu      = 1.00
Go      = 1.00
Isp     = 0.5673
Thrust  = 0.5166
Mo      = 10.0000
ao      = 1.856
eo      = 0.4550
wo      = 0.000
ad      = 3.707
ed      = 0.7262
wd      = 0.000
TMAX    = 0.000
NGS     = 100
NIX     = 3
```

GSHOOT reports:

```
Best constant Lagrange multipliers (initial)
  C...  0.44612E+00  0.30915E+00  0.35928E-14
Best initial true anomaly
   vo= 0.53782E+01
Best transfer time
   tf= 0.18265E+01
Best relative errors (h,ex,ey,Hs)
  G...  0.58626E-08 -0.39904E-01  0.17988E-01 -0.36813E-02
```

The resulting file has been supplied as "Tutorials/2D 5burn/GSHOOT/burn 4/" The fifth burn "" ["Tutorials/2D 5burn/GSHOOT/burn 5/"]:

```
mu      = 1.00
Go      = 1.00
Isp     = 0.5673
Thrust  = 0.5166
Mo      = 10.0000
ac      = 3.707
ec      = 0.7262
wc      = 0.000
ad      = 6.400
ed      = 0.0000
wd      = 0.000
THAX    = 0.000
NGS     = 100
NIX     = 3
```

GSHOOT reports:

```
Best constant Lagrange multipliers (initial)
  C...  0.28015E+00 -0.71802E+00 -0.62715E-00
Best initial true anomaly
   vo= 0.30096E+01
Best transfer time
   tf= 0.32219E+01
Best relative errors (h,ex,ey,Hs)
  G...  0.26077E-11 -0.93204E-02 -0.25981E-01  0.53608E-01
```

The GSHOOT output has been supplied as "Tutorials/2D 5burn/GSHOOT/burn 5/"

The files easily concatenate. The resulting file has been supplied as "Tutorials/2D 5burn/GSHOOT/MPM2D.guess"

Attempt Computation of Solution with MPMM2D

At this point, we have a solution guess for the entire trajectory in the PATCH2D format. One option for obtaining the solution is to run MPMM2D with this input. However, one may get a iteration history like this:

MPMM2D Output

```
  Cur. Norm     It#    Best Norm (at) #    Short Time    Bn#  Bst Wrst El    El#
 ------------ ----- ------------ ----- ------------ ----- ------------ -----
 0.68735E+01     1  0.68735E+01     1  0.79429E-00     3  0.34045E+01     43
 0.68735E+01    45  0.68735E+01    45  0.79429E-00     3  0.34045E+01     43
 0.68735E+01    90  0.68735E+01    45  0.79429E-00     3  0.34045E+01     43
BCC: Possible conflict in orbit choice
     A=-2.617712643152
     E=2.335666952254
     W=2.556150238017
     [LOCATION #1]
BCC: Possible conflict in orbit choice
     A=-2.617712643152
     E=2.335666952254
     W=2.556150238017
     [LOCATION #1]
BURN  WARNING. BCC CLAIMS AN ERROR
      IN THE INITIAL POINT CALCULATION
      W1=5.684341886808E-14
      W2=1.858576979153
      W3=0.7367094236306
      [LOCATION #1]
BCC: Possible conflict in orbit choice
     A=4.117497825609
     E=1.458915419989
     W=-0.5075814176646
     [LOCATION #1]
     INCONSISTENT:
     A*(1e0-E**2).LT.0E0
STOP  (called by BCC )
CP: 20.155s,  Wallclock: 29.935s,  33.7% of 2-CPU Machine
HWM mem: 213617, HWM stack: 26810, Stack overflows: 0
```

Note that the current norm error started at 6.3735: though such a large error does not always induce failure of MPMM2D, it may.

If MPMM2D Fails, Use PATCH2D

In such a situation, the more robust PATCH2D is useful. Since the file format is identical, this is very convenient. PATCH2D does require one additional input file, for its inner loop tolerances. The file is called "PATCH2D.tols" and for this tutorial, it has been supplied as "Tutorials/2D 5burn/PATCH2D/PATCH2D.tols" and listed below:

```
FTOL = 1.000000000000000000000E-08
LTOL = 1.000000000000000000000E-07
GTOL = 1.000000000000000000000E-03
TOL2 = 1.000000000000000000000E-05
```

We have chosen a rather strict tolerance for "function improvement" convergence, a slightly less strict tolerance for "line search" convergence, a very loose tolerance for "gradient norm" convergence, and a rather loose convergence tolerance for DCNLP iterations.

It needs to be said that the drawback to PATCH2D is its speed. For this tutorial, PATCH2D was run. After renaming "MPM2D.GUESS" to "PATCH2D.GUESS" and running PATCH2D, we see the following iterations:

```
   Function TOL (FTOL) = 1.E-8
   Gradient TOL (GTOL) = 1.E-3
Line Search TOL (LTOL) = 1.E-7
Max # iterates (ITMAX) = 200

IT#  Cost Func     Improvement  Gradient     Criterion    Line  #eval
---- ------------ ------------ ------------ ------------ ---- ------
   0  0.66455E-01  0.00000E+00  0.47211E-02
   1  0.66455E-01  0.00000E+00  0.47211E-02  0.86837E-01    7    12
   2  0.66021E-01 -0.43418E-01  0.14439E-02  0.21908E-01    6    22
   3  0.65911E-01 -0.56372E-01  0.56984E-01  0.52307E-02    4    32
   4  0.65885E-01 -0.56988E-01  0.12130E-02  0.10740E-01    5    41
   5  0.65831E-01 -0.62356E-01  0.82250E-01  0.55562E-01    4    51
   6  0.65556E-01 -0.90139E-01  0.18560E-02  0.13395E-01    5    63
   7  0.65487E-01 -0.96836E-01  0.11952E-02  0.12299E-01    4    72
   8  0.65425E-01 -0.10299E-00  0.39034E-01  0.17844E-02    5    82
   9  0.65416E-01 -0.10388E-00  0.75207E-01  0.88954E-02    5    92
  10  0.65372E-01 -0.10833E-00  0.71069E-01  0.85288E-02    3   101
  11  0.65329E-01 -0.11259E-00  0.80968E-01  0.21341E-02    5   111
  12  0.65318E-01 -0.11366E-00  0.32664E-01  0.13259E-02    6   121
  13  0.65312E-01 -0.11432E-00  0.13498E-01  0.50050E-04    6   131
  14  0.65311E-01 -0.11435E-00  0.55807E-00  0.31066E-04    5   140
  15  0.65311E-01 -0.11436E-00  0.33032E-00  0.16464E-04    3   149
  16  0.65311E-01 -0.11437E-00  0.76677E-00  0.18621E-03    3   159
  17  0.65310E-01 -0.11446E-00  0.11374E-01  0.24295E-03    3   168
  18  0.65309E-01 -0.11456E-00  0.19965E-01  0.16953E-03    4   177
  19  0.65308E-01 -0.11466E-00  0.83243E-00  0.10038E-03    4   186
  20  0.65308E-01 -0.11471E-00  0.54440E-00  0.11258E-04    4   195
  21  0.65308E-01 -0.11471E-00  0.42573E-00  0.76726E-04    5   206
  22  0.65307E-01 -0.11475E-00  0.80416E-00  0.10644E-03    3   215
  23  0.65307E-01 -0.11481E-00  0.15569E-01  0.13474E-03    4   224
  24  0.65306E-01 -0.11487E-00  0.43370E-00  0.33022E-04    3   233
  25  0.65306E-01 -0.11489E-00  0.70424E-00  0.24930E-04    4   242
  26  0.65306E-01 -0.11490E-00  0.58308E-00  0.20997E-03    4   253
  27  0.65305E-01 -0.11501E-00  0.14167E-01  0.27205E-03    4   262
  28  0.65303E-01 -0.11514E-00  0.30933E-01  0.10666E-01    6   277
  29  0.65250E-01 -0.12048E-00  0.75971E-01  0.83111E-02    7   289
  30  0.65209E-01 -0.12463E-00  0.21314E-01  0.34787E-03    5   299
  31  0.65207E-01 -0.12481E-00  0.18041E-01  0.23515E-03    6   309
```

The PATCH2D code had been left to run overnight, about 12 hrs. It did not satisfy any convergence criterion by the 31st iteration, execution was terminated. The output file "PATCH2D.BEST" has been put into in the "Tutorial" folder as "Tutorials/2D 5burn/PATCH2D/PATCH2D.BEST"

Use PATCH2D Output for MPMM2D

Now, this file was renamed to "MPM2D.GUESS" and used for input to "MPMM2D." The iterations are listed below:

```
  CUR. NORM    IT#   BEST NORM (AT) #   SHORT TIME   EN#  BST WRST EL   EL#
------------ ----- ------------ ----- ------------ ----- ------------ -----
0.40240E-00      1  0.40240E-00      1  0.67665E-00     3  0.31525E-01    34
0.40240E-00     45  0.40240E-00     41  0.67665E-00     3  0.31525E-01    34
0.35362E-02     90  0.35361E-02     72  0.10959E-01     3  0.19249E-02    36
0.24411E-06    135  0.35880E-10    121  0.11288E-01     4  0.30566E-11    26
0.65414E-07    180  0.30394E-10    174  0.11288E-01     4  0.24710E-11    26
0.29068E-10    225  0.29068E-10    225  0.11288E-01     4  0.23204E-11    26
0.72452E-07    270  0.29068E-10    225  0.11288E-01     4  0.23204E-11    26
0.11927E-06    315  0.26968E-10    276  0.11288E-01     4  0.21352E-11    26
0.47058E-07    360  0.22244E-10    331  0.11288E-01     4  0.16456E-11    26
0.28231E-08    405  0.20320E-10    360  0.11288E-01     4  0.15220E-11    26
0.23782E-06    450  0.15318E-10    441  0.11288E-01     4  0.10881E-11    26
0.14141E-10    495  0.13615E-10    493  0.11288E-01     4  0.99760E-11    26
0.11785E-08    540  0.13615E-10    493  0.11288E-01     4  0.99760E-11    26
0.40513E-06    585  0.13339E-10    544  0.11288E-01     4  0.92668E-11    26
0.47061E-07    630  0.11624E-10    599  0.11288E-01     4  0.76525E-11    26
0.78723E-07    675  0.98522E-11    654  0.11288E-01     4  0.60751E-11    26

*** FATAL    ERROR 3 from NEQNF.  The iteration has not made good progress
***          The user may try a new initial guess.
```

Obviously, the solution was found; however, a shortcoming in the NEQNF solver did not allow it to claim convergence. This seems to be common among nonlinear equation solvers. An easy fix is to perturb the guess slightly. In this case, the eccentricity of the first transfer orbit was perturbed from

```
ex    = 0.14423753690672836260E-00
```

to

```
ex    = 0.16433753690672836260E-00
```

For this new guess, in the "Tutorial" folder as "Tutorials/2D 5burn/MPM2D.GUESS," the MPMM2D iterations are:

```
  CUR. NORM     IT#    BEST NORM (AT) #    SHORT TIME    BN#  BST WRST EL.   EL#
 ------------ ------ ------------ ------ ------------ ----- ------------ -----
 0.40418E-00      1  0.40418E-00      1  0.67665E-00     3  0.31525E-00     34
 0.40418E-00     45  0.40418E-00     41  0.67665E-00     3  0.31525E-00     34
 0.30687E-01     90  0.30687E-01     56  0.10688E-01     3  0.14737E-01     26
 0.46830E-07    135  0.21092E-10    122  0.11288E-01     4  0.13635E-10     35
 0.60908E-07    180  0.18042E-10    172  0.11288E-01     4  0.14477E-10     22
 0.30214E-06    225  0.17836E-10    220  0.11288E-01     4  0.14065E-10     22
---------
REQUIRED # FUNCTION EVALS = 268
------------------------------
 TOTAL BURN TIME = 6.512750674051
 FINAL MASS = 4.068387805015
 SHORTEST BURN LENGTH = 1.128831615888
 SHORTEST BURN IS #4
------------------------------
SOLUTION SAVED
```

The solution file is given in the "Tutorial" folder as "Tutorials/2D 5burn/MPM2D.SOL".

VII.2. Convert MPMM3D File to BND3D File, Run BND3D

This tutorial demonstrates how to use MP2BND to convert a MPMM3D file to a BND3D file.

The file "Tutorials/MPM to BND3D/MPM3D.GUESS" is a solution to an orbit transfer problem, as claimed by MPMM3D. The particular problem it solves is not relevant, but it will be clarified anyway. The header of this file follows:

```
TOL  = 0.10000000000000000000E-08
MU   = 0.10000000000000000000E+01
T    = 0.51658300000000068053E+00
Go   = 0.10000000000000000000E+01
Isp  = 0.56730999999999909278E+00
hxo  = 0.47715876030000003993E+00
hyo  = 0.00000000000000000000E+00
hzo  = 0.87881711269999840397E+00
exo  = 0.00000000000000000000E+00
eyo  = 0.00000000000000000000E+00
hxf  = 0.00000000000000000000E+00
hyf  = 0.00000000000000000000E+00
hzf  = 0.25298517739999937248E+01
exf  = 0.00000000000000000000E+00
eyf  = 0.00000000000000000000E+00
NORB =  5
```

The orbit transfer is, therefore, from LEO to GEO and circle to circle in 6 burns. Now, suppose we want to further investigate this problem with the more general BND3D code, so that oblateness and drag effects can be modeled.

Run MP2BND

The main task here is to simply run MP2BND. This code will create the file "BND3D.GUESS" which has been supplied as "Tutorials/MPM to BND3D/BND3D.GUESS."

Run BND3D to check It is prudent at this point to use BND3D to check MPMM3D's results In this tutorial, the following "BND3D.SCRIPT" file was used

```
BND3D.GUESS
0
1
0
0
1d-4
1d-10
100
BND3D.SOL
1
BND3D.REINT
```

which is supplied as "Tutorials/MPM to BND3D/BND3D.SCRIPT." This says the input file is "BND3D.GUESS," don't show B.C. errors to the screen, solve with free final time, don't include switching points as nodes in the output, FCMIN=1D-4, TOL=1D-10, use no more than 100 iterations, save solution as "BND3D.SOL," provide additional info and save this info in "BND3D.REINT." The output BND3D produces to the screen is listed below:

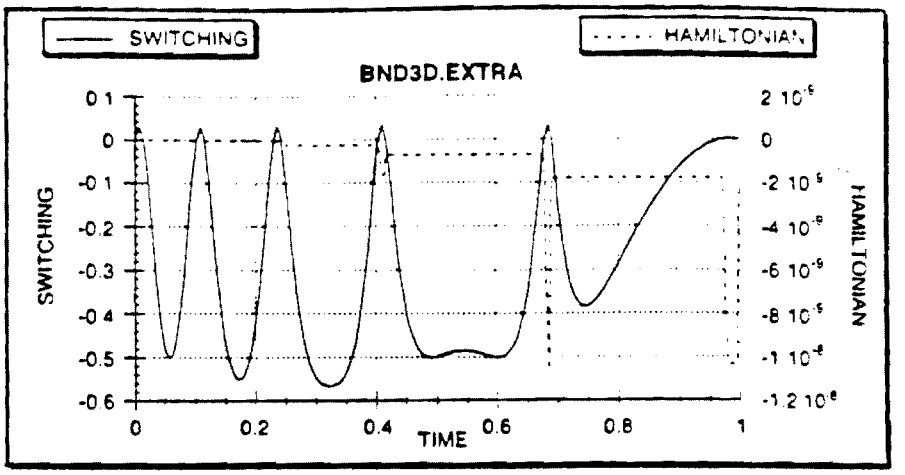
```
B.C.S ? 0
FREE FINAL TIME  1
HOMOTOPY: 0
MU=       1.0000000000000000
REQ=      0.0000000000000000000E+00
J2=       0.0000000000000000000E+00
GO=       1.0000000000000000
BETA=     0.0000000000000000000E+00
RO=       0.0000000000000000000E+00
FOU=      0.0000000000000000000E+00
S=        0.0000000000000000000E+00
CD=       0.0000000000000000000E+00
ISP=      0.5673099999999998962
THRUST=   0.5165830000000021014
AI=       1.0000000001053E792
EI=       0.0000000000000000000E+00
OMEGAI=   0.0000000000000000000E+00
RAI=      89.999999997066E8C
I-I=      28.500000000901081?
AF=       6.400014999641091026
EF=       0.0000000000000000000E+00
OMEGAF=   0.0000000000000000000E+00
RAF=      0.0000000000000000000E+00
I-F=      0.0000000000000000000E+00

NOTE: ANGLES MUST BE IN DEGREES
M= 44
*N= 25



 ..........................................................................
.....................................


   INITIAL DATA



   N=25      M=44      MS=10
   PRESCRIBED RELATIVE PRECISION    .10D-09
   MAXIMUM PERMITTED NUMBER OF ITERATIONS:100




 ..........................................................................
.....................................
  IT    ABS ERR    LEVEL1     LEVEL2     LEVEL3   RELAX  NEW   COND(M)    NORM M
```

```
 0    .14D-07    .11D-07    .11D-07    .76D-08          0    .38D+08    .93D-02
      .14D-07    .11D-07    .11D-07    .76D-08   .000
 1    .14D-07    .67D-07    .67D-07    .14D-07          1    .36D+08    .89D-02
      .14D-07    .67D-07    .67D-07    .14D-07   .001
 2    .14D-07    .67D-07    .67D-07    .14D-07          2    .36D+08    .89D-02
      .14D-07    .66D-07    .66D-07    .13D-07   .008
 3    .14D-07    .66D-07    .66D-07    .13D-07          3    .36D+08    .89D-02
      .11D-07    .51D-07    .51D-07    .10D-07   .121
 4 *  .11D-07    .51D-07    .51D-07    .10D-07          4    .36D+08    .89D-02
      .34D-15    .44D-15    .72D-12    .43D-10  1.000
 5    .34D-15    .44D-15    .73D-12    .44D-10          5    .36D+08    .89D-02
      .26D-15    .31D-15    .28D-14    .58D-12  1.000
 6    .26D-15    .31D-15    .33D-14    .43D-12          6    .20D+09    .89D-02
      .73D-15    .21D-14    .32D-14    .77D-11  1.000
 7    .73D-15    .21D-14    .21D-14    .73D-13          0    .37D+08    .89D-02
      .43D-15    .12D-14    .12D-14    .42D-13   .236
 8    .43D-15    .12D-14    .13D-14    .50D-13          1    .47D+09    .89D-02
      .47D-16    .18D-16    .24D-16    .76D-14  1.000
 9    .47D-16    .18D-16    .19D-16    .12D-14          2    .22D+09    .89D-02
      .22D-18    .81D-18    .11D-17    .14D-15  1.000
10    .22D-18    .81D-18    .11D-17    .16D-15          3    .20D+09    .89D-02
      .75D-20    .38D-19    .14D-18    .25D-16  1.000
11    .75D-20    .38D-19    .83D-19    .15D-16          4    .47D+08    .89D-02
      .32D-21    .66D-21    .55D-20    .87D-17  1.000
12    .32D-21    .66D-21    .35D-20    .33D-17          5    .45D+08    .89D-02
      .16D-21    .55D-21    .56D-20    .61D-17  1.000
13    .16D-21    .55D-21    .79D-20    .35D-16          0    .37D+08    .89D-02
      .48D-22    .17D-21    .26D-19    .10D-16   .449
14    .48D-22    .17D-21    .26D-19    .10D-16          0    .37D+08    .89D-02
      .36D-25    .58D-24    .50D-19    .22D-17  1.000
15    .36D-25    .58D-24    .51D-19    .23D-17          0    .37D+08    .89D-02
      .36D-25    .10D-24    .66D-21    .18D-17  1.000
16    .36D-25    .10D-24    .66D-21    .98D-18          1    .42D+08    .89D-02
      .18D-25    .20D-24    .36D-19    .19D-17  1.000
      .29D-25    .28D-24    .40D-19    .56D-17   .473
      .22D-25    .27D-24    .35D-19    .52D-17   .080
      .38D-25    .26D-24    .23D-19    .11D-17   .007
      .28D-25    .50D-25    .93D-20    .68D-18   .001
17    .28D-25    .50D-25    .58D-20    .17D-17          0    .37D+08    .89D-02
      .36D-25    .70D-24    .18D-19    .96D-18   .001
18    .36D-25    .70D-24    .17D-19    .95D-18          0    .37D+08    .89D-02
      .28D-25    .95D-25    .96D-21    .33D-18   .103
19    .28D-25    .95D-25    .27D-20    .16D-18          1    .75D+07    .89D-02
 ****************************************************************************
*************************************


   SOLUTION OBTAINED AFTER 20    ITERATION STEPS

   SOLUTION DATA

   SWITCHING POINTS



 ****************************************************************************
*************************************
NAME OF FILE FOR SOLUTION DATA: ->BND3D.SOL                    <-
```

It eventually computes the solution to its own criterion, however, it is clear that BND3D has verified the MPMM3D solution.

Useful Information in BND3D.EXTRA

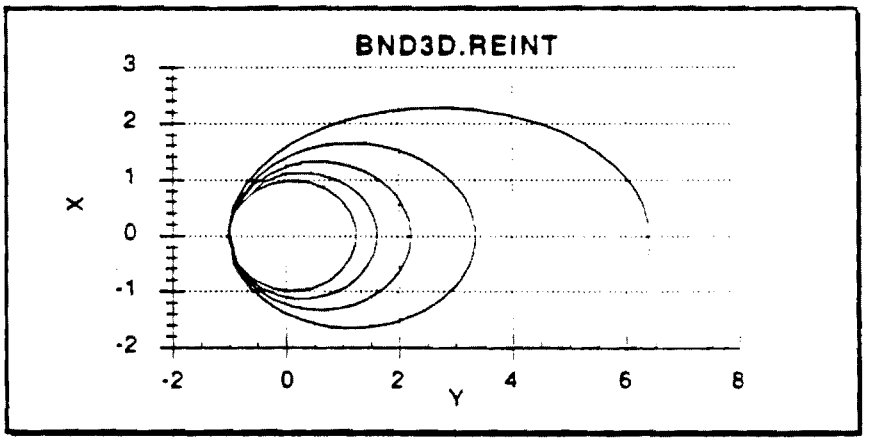
The information provided by BND3D.EXTRA is arguable essential. This file contains data for the switching function and Hamiltonian as functions of time. The plot below is a graphical representation of what BND3D.EXTRA provides



The Hamiltonian is almost zero, and very close to the tolerance. The jumps in the Hamiltonian at the switching points is a common numerical phenomenon. Also very important, note that this verifies the assumed switching structure: thrust on at the beginning, precisely ten switching points, and thrust on at the end. Finally, note the hump between the fourth and fifth burns, noting the location of such humps is often useful in deciding the location of an additional burn

Useful Information in BND3D.REINT

The file "BND3D.REINT" also supplies useful data in the form of a detailed trajectory. The complete state and costate is included. The plot below, a projection of the trajectory onto the x-y plane, was created using the raw data in the "BND3D.REINT" file.



Note that this plot is rotated 90° for clarity.

VII.3. Run BND3D with Homotopy

This tutorial begins with the solution file from the "Convert MPMM3D File to BND3D File, Run BND3D;" tutorial.

Suppose we try and accomplish this change in one step, by altering the "BND3D.GUESS" file. The script ("BND3D.SCRIPT") is, simply:

```
BND3D.GUESS
0
1
0
0
1d-4
1d-10
100
BND3D.SOL
1
BND3D.REINT
```

Here is the BND3D output to the screen:

```
B C S ? 0
FREE FINAL TIME: 1
HOMOTOPY: 0
MU=       1.0000000000000000
REQ=      0.0000000000000000000E+00
J2=       0.0000000000000000000E+00
GO=       1.0000000000000000
BETA=     0.0000000000000000000E+00
RO=       0.0000000000000000000E+00
ROU=      0.0000000000000000000E+00
S=        0.0000000000000000000E+00
CD=       0.0000000000000000000E+00
ISP=      0.567309999999998982
THRUST=   0.516583000000001014
A1=       1.0000000001C538792
E1=       0.0000000000000000000E+00
OMEGA1=   0.0000000000000000000E+00
RA1=      89.9999999997066880
I-1=      28.5000000009C10819
AF=       6.6001499984109104 3
EF=       0.0000000000000000000E+00
OMEGAF=   0.0000000000000000000E+00
RAF=      0.0000000000000000000E+00
I-F=      0.0000000000000000000E+00

NOTE: ANGLES MUST BE IN DEGREES
M= 44
*N= 25
```

```
*****************************************************************************
*****************************************

   INITIAL DATA


   N=25      M=44      MS=10
   PRESCRIBED RELATIVE PRECISION    .10D-09
   MAXIMUM PERMITTED NUMBER OF ITERATIONS100
```

```
*****************************************************************************
*****************************************
 IT    ABS.ERR.    LEVEL1      LEVEL2      LEVEL3    RELAX.  NEW   COND(M)     NORM(M)
  0    .15D-02     .15D-02     .22D-01     .11D+03           0    .39D+08     .93D+02
       .15D-02     .15D-02     .22D-01     .11D+03   .000
  1    .15D-02     .15D-02     .19D-01     .10D+03           0    .37D+08     .89D+02
       .15D-02     .15D-02     .19D-01     .10D+03   .003
  2    .15D-02     .15D-02     .48D-01     .11D+03           1    .67D+08     .89D+02
       .16D-02     .18D-02     .22D+00     .13D+03   .026
       .15D-02     .15D-02     .49D-01     .11D+03   .005
  3    .15D-02     .15D-02     .37D+00     .11D+03           0    .37D+08     .89D+02
       .15D-02     .15D-02     .41D+00     .11D+03   .017
  4    .15D-02     .15D-02     .90D+01     .46D+03           0    .36D+08     .89D+02
       .15D-02     .15D-02     .88D+01     .45D+03   .011
  5    .15D-02     .15D-02     .13D+02     .62D+03           0    .37D+08     .89D+02
       .21D-02     .26D-02     .18D+02     .82D+03   .056
       .14D-02     .15D-02     .13D+02     .61D+03   .013
  6    .14D-02     .15D-02     .18D+02     .81D+03           0    .37D+08     .89D+02
```

```
        .26D-02     .36D-02     .28D-02     .12D-04     073
        .14D-02     .15D-02     .17D-02     .79D-03     025
   7    .14D-02     .15D-02     .23D-02     .10D-04                 0     37D-05     89D-02
        .34D-02     .44D-02     .40D-02     .17D-04     .092
        .14D-02     .14D-02     .22D-02     .10D-04     .019
   8    .14D-02     .15D-02     .28D-02     .13D-04                 0     36D-05     89D-02
        .67D-02     .15D-01     .90D-02     .40D-04     .144
        .13D-02     .15D-02     .27D-02     .12D-04     .023
   9    .13D-02     .15D-02     .36D-02     .16D-04                 0     38D-05     89D-02
        .50D-02     .11D-01     .84D-02     .36D-04     .155
        .13D-02     .15D-02     .34D-02     .15D-04     .031
  10    .13D-02     .15D-02     .48D-02     .21D-04                 0     40D-05     89D-02
        .14D-02     .30D-02     .22D-02     .70D-04     .135
  11    .14D-02     .30D-02     .30D-03     .14D-05                 0     29D-05     91D-02
        .36D-02     .76D-02     .61D-03     .27D-05     .038
        .14D-02     .30D-02     .30D-03     .14D-05     .004
(many lines omitted for brevity)
  62    .15D-00     .15D-01     .29D-01     .24D-03                 0     62D-06     95D-02
        .14D-00     .15D-01     .26D-01     .20D-03     .089
  63    .14D-00     .15D-01     .26D-01     .17D-03                 1     58D-06     95D-02
        .52D-01     .26D-02     .10D-06     .23D-06    1.000
        .14D-00     .14D-01     .24D-01     .15D-03     .062
  64    .14D-00     .14D-01     .23D-01     .28D-03                 0     68D-06     95D-02
        .16D-00     .16D-01     .23D-01     .19D-03     .135
  65    .16D-00     .16D-01     .25D-01     .23D-03                 0     73D-06     96D-02
        .11D-01     .30D-02     .10D-03     .50D-05    1.000
        .15D-00     .15D-01     .22D-01     .18D-03     .101
  66    .15D-00     .15D-01     .24D-01     .18D-03                 1     92D-06     96D-02
        .96D-01     .25D-02     .47D-03     .58D-05    1.000
        .17D-00     .15D-01     .22D-01     .14D-03     .091
```

Execution was terminated early because BND3D was clearly stuck In this type of situation, where BND3D has difficulty, it is often useful to resort to homotopy.

BND3D has a homotopy loop and is utilized, for this tutorial, with the following script (supplied as "BND3D HOMOTOPY/BND3D.SCRIPT"):

```
BND3D.GUESS
0
1
1
10
0D0
0.516583D0
0.5673D0
6.6D0
0D0
0D0
1D0
0D0
0D0
28.5D0
0
1d-4
1d-7
100
BND3D.SOL
1
BND3D.REINT
```

To make convergence easier, the tolerance was reduced to 10^{-7}. Ten homotopy steps have been suggested and the final semimajor axis is requested to be 6.6.

The output to the screen is very long for a homotopy run, and is omitted from the tutorial, however, it may be found in the file "BND3D HOMOTOPY/screen output." One the other hand, the

"BND3D HOMOTOPY/BND3D.REPORT" file indicates how the homotopy progressed:

```
      IJ,        KP,          U,          DU
       0 ,        25 ,  .9000000D+00 , -.1000000D+00 ,
       1 ,        22 ,  .8000000D+00 , -.1000000D+00 ,
       2 ,        25 ,  .7000000D+00 , -.1000000D+00 ,
       3 ,        30 ,  .6000000D+00 , -.1000000D+00 ,
       4 ,        19 ,  .5000000D+00 , -.1000000D+00 ,
       5 ,        19 ,  .4000000D+00 , -.1000000D+00 ,
       6 ,        26 ,  .3000000D+00 , -.1000000D+00 ,
       7 ,        19 ,  .2000000D+00 , -.1000000D+00 ,
       8 ,        28 ,  .1000000D+00 , -.1000000D+00 ,
       9 ,        -4 ,  .1387779D-15 , -.1000000D+00 ,
      10 ,        17 ,  .7500000D-01 , -.2500000D-01 ,
      11 ,        14 ,  .5000000D-01 , -.2500000D-01 ,
      12 ,        23 ,  .2500000D-01 , -.2500000D-01 ,
      13 ,        17 ,  .1457168D-15 , -.2500000D-01 ,
```

This indicates that even though ten steps were suggest, thirteen were required. Iterations failed for the ninth step. BND3D then adjusted the step size (DU) to one-quarter and continued until completion.

VII.4. Using MBCM3D

The following sample input file has been supplied for MBCM3D ("Tutorials/MBCM3D/MBCM3D.GUESS"):

```
Mu=     ,              1.0000000000000000
Req=    ,              0.0000000000000000
J2=     ,              0.0000000000000000
Go=     ,              0.0098000000000000
Beta=   ,              0.0000000000000000
Ro=     ,              0.0000000000000000
Rou=    ,              0.0000000000000000
S=      ,              0.0000000000000000
Cd=     ,              0.0000000000000000
Isp=    ,            134.0000000000000000
Thrust=,               0.0300000000000000
Ai=     ,              3.8473050000000000
ei=     ,              0.0237770420000000
omegai=,               0.0000000000000000
RAi=    ,              0.0000000000000000
i-i=    ,              0.0000000000000000
Af=     ,              1.5000000000000000
ef=     ,              0.3333333333333333
omegaf=,               0.0000000000000000
RAf=    ,              0.0000000000000000
i-f=    ,              0.0000000000000000
X =     ,             -3.1176801908731560
Y =     ,              2.3750078935282890
Z =     ,              0.0000000000000000
U =     ,             -0.3091335043231690
V =     ,             -0.3934426605343490
W =     ,              0.0000000000000000
M =     ,              1.5270000000000000
Lam-X=  ,              0.0841506494807840
Lam-Y=  ,             -0.0700639150701650
Lam-Z=  ,              0.0000000000000000
Lam-U=  ,              0.5317586997542810
Lam-V=  ,              0.7377831735348990
Lam-W=  ,              0.0000000000000000
Lam-M=  ,              0.7821113170205860
TF =    ,             19.0531498613972200
G1 =    ,              0.0000000000000000
G2 =    ,              0.0000000000000000
G3 =    ,             -0.6560877956359570
G4 =    ,             -0.2359886514576700
G5 =    ,             -0.0006530929371980
G6 =    ,              0.0000000030000000
G7 =    ,              0.0000000000000000
G8 =    ,              0.2054327729109010
G9 =    ,             -0.0286054101410370
G10=    ,              0.0063519666993770
ACC=    ,              0.0000010010000000
ITM=    , 100
```

```
note  all angles are in degrees
```

The MBCM3D iterations, output to the screen (see file "Tutorials/MBCM3D/screen.output") follow:

```
    ITERATIONS =    1     CALLS OF VF02AD =    1

 X = -0.311768019087732E+01  0.237500789352B3E-01  0.000000000000000E+00
     -0.309133504322317E+00 -0.393443660534J5E-00  0.000000000000000E+00
      0.152700000000000E+01  0.841506494807E4E-01 -0.700639150701E5E-01
      0.000000000000000E+00  0 531758699754J6E-00  0 737783173534S1E-00
      0.000000000000000E+00  0.782111317020S9E-00  0.190531498663S7E-02
      0 000000000000000E+00  0.000000000000000E+00 -0.656087795631S6E-00
     -0.235988651457S7E+00 -0.453092937198O0E-03  0.000000000000000E+00
      0 000000000000000E+00  0.205432772910S0E+00 -0.286054101412O7E-01
      0.635196669937?0E-02

 F =  0.863344944743?3E-07

 C =  0.000000000000000E+00  0.000000000000000E+00 -0.328158533203?9E-02
      0.318039781237?1E-02  0.187645807330E3E-02 -0.293807918097B1E-01
      0.437516918980?1E-02  0.000000000000000E+00 -0.111158988604?1E-01
      0.844026729551?5E-03  0.000000000000000E+00 -0.160254768326?5E-01
      0.000000000000000E+00  0.000000000000000E+00 -0.742677297349?5E-
      0.225309999408?6E-03  0.173131539696?5E-03  0.000000000000000F-
     -0.227406697499?0E-04 -0.211773936?0?8E-06  0.000000000000000F-
     -0.575695534958?4E-05 -0.906235655406?2E-05  0.000000000000000E+

    ITERATIONS =    2     CALLS OF VF02AD =    2

 X = -0.310365022095?4E-01  0.239286421449?0E-01  0.255684466553?9E-
     -0.311363E3425?03E-00 -0.391753222637?7E-00 -0.425233626334?9E-
      0.152700000000000E+01  0.837653141956?0E-01 -0.705674822912?1E-
     -0.801980941063?0E-20  0.535965621552?6E-00  0.734455897938?6E-
     -0.464815336300?5E-20  0.782036182439?7E-00  0.190537649149?5E-
     -0.301281718970?2E-18  0.122320397866?5E-19 -0.656665258117?9E-
     -0.236206597572?0E-00 -0.205698050444?8E-03 -0.374259547933?7E-
      0.249443016286?6E-20  0.205540533137?9E+00 -0.284196726047?9E-
      0.646566681221?6E-02

 F =  0.249101071805?1E-11

 C = -0.580425834854?7E-20 -0.300641924651?4E-20  0.112788883566?5E-
     -0.868832300437?8E-03  0.627799353623?9E-03  0.506080430774?0E-
     -0.775665264914?4E-03  0.882470873594?6E-22  0.136607927672?4E-
     -0.656540833915?0E-03 -0.154936967040?0E-19  0.498717087?71?2E-
     -0.158742193226?6E-21 -0.211593653068?5E-19  0.638920883?5?2E-
     -0.285051090613?8E-04  0.299212571190?7E-04  0.000000000000?2E-
      0.383087677136?2E-06 -0.853257215993?1E-07  0.101918?0?0?110E-
      0.242368706082?3E-05  0.306748837886?6E-05  0.690507?8954?59E-

                              [lines omitted]

    ITERATIONS =    5     CALLS OF VF02AD =    5

 X = -0.311131662535?9E-01  0.238308940926?5E-01  0.695081098?7120E-
     -0.310097034142?0E-00 -0.392731270031?3E-00 -0.202845?82?6?9?E-
      0.152700000000000E+01  0.839490890444?9E-01 -0.703?0?59?7?55?E-
     -0.558237674945?4E-20  0.533486322697?1E-00  0.736340492551?9E-
     -0.799584338609?2E-20  0.782111329866?4E-00  0.190?37?8?666?9E-
     -0.757651206253?9E-20 -0.156952063594?8E-20 -0.656187?21381?1E-
     -0.235988547596?1E-00 -0.453174630730?1E-03 -0.628?63981065?8E-
     -0.797?133422306?4E-17  0.205432738536?2E-00 -0.286?5461858?7E-
      0.635192533216?2E-02

 F =  0.212217202906?2?E-25

 C = -0.581299987686?2E-20  0.894363524811?4E-21  0.207360869?0E33E-
     -0.779467939082?8E-07  0.330181236851?9E-06  0.200017?864?9?0E-
     -0.711327?8854?12E-07  0.672546662598?9E-20 -0.359?47?9656?16E-
     -0.496519123593?4E-07  0.460734029654?7E-20  0.109048812644?1E-
     -0.208848628841?8E-20 -0.847900411647?7E-20  0.831285262647?1E-
     -0.492936136357?1E-09  0.357737?2E32925E-09  0.710542?3576?10E-
      0.514033260401?5E-11 -0.178523862359?3E-11 -0.346340337?02?5E-
      0.330722116259?4E-10  0.211670680982?3E-10  0.398022?5503?91E-
```

THE PRINTING OF THE LAST ITERATION GIVES THE
VALUES THAT ARE RETURNED BY SUBROUTINE VF02AD

```
---SOLUTION CONVERGED---
X =    -0.311131662535?9E-01
     0.238308940926?5E-01
     0.699081098?7120E-20
    -0.310097034142?0E-00
    -0.392731270031?3E-00
    -0.202845?8?6?9?E-20
     0.152700000000000E+01
```

```
 0.8394908904444 8E-01
-0.7030059073959 5E-01
-0.5582376749457 4E-20
 0.5334863228974 1E+00
 0.7363404925512 9E+00
-0.7995843386093 2E-20
 0.7821113298657 4E+00
 0.1907378806634 9E+02
-0.7578912082537 9E-20
-0.1569520635946 8E-20
-0.6560876213819 1E+00
-0.2359885475967 1E+00
-0.4531746307311 E-03
-0.6283639510656 8E-17
-0.7977133422306 4E-17
 0.2054327385363 2E+00
-0.2860546185862 7E-01
 0.6351925332169 2E-02
```

Appendix A GSHOOT's File Format

The input file "" for "GSHOOT" has a specific file format. The file must consist of exactly 14 lines. The variables read from this file have a specific order: MU, GO, ISP, THRUST, MO, AO, EO, WO, AD, ED, WD, TMAX, NGS, and NIX. All variables are of the type REAL except the last two, NGS and NIX, which are of the type INTEGER. An example file is listed below.

```
Mu      = 1.00
Go      = 1.00
Isp     = 0.567
Thrust  = 0.516
Mo      = 10.0000
ao      = 1.00000
eo      = 0.000
wo      = 0.000
ad      = 1.285
ed      = 0.219
wd      = 0.000
TMAX    = 0.000
NGS     = 100
NIX     = 3
```

On each line intended to supply a REAL variable, the FORTRAN FORMAT layout is (1X,A9,F30.15); for INTEGER variables, this statement is (1X,A9,I10). Therefore, each line starts with a blank space followed by nine characters, all of which are ignored. Only the numerical data following is used.

Appendix B The PAT2D and PAT3D File Formats

The PAT2D file format is used by MPMM2D and PAT2D. The PAT3D file format is only used by MPMM3D. They are called the PAT formats because all of the information supplied by the PAT2D format is used by PAT2D; only some of the information is used by MPMM2D and MPMM3D. Exactly what information is used by MPMM2D and MPMM3D is described in Chapter IV.

The PAT2D format is represented below:

```
TOL   = #
MU    = #
T     = #
GO    = #
ISP   = #
AO    = #
EXO   = #
EYO   = #
AF    = #
EXF   = #
EYF   = #
a     = #
ex    = #
ey    = #
NORB  =  2
NODE  =  3
SEL   =  1
index,x,y,u,v,m,lx,ly,lu,lv,lm,tf,g1,g2,g3,g4,g5,gf
  1, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #,
  2, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #,
  3, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #,
  4, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #,
a     = #
ex    = #
ey    = #
NODE  =  3
SEL   =  2
index,x,y,u,v,m,lx,ly,lu,lv,lm,tf,g1,g2,g3,g4,g5,g6
  1, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #,
  2, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #,
  3, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #,
  4, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #, #,
a     = #
ex    = #
ey    = #
NODE  =  3
SEL   =  3
INDEX,X,Y,U,V,M,TF,L1,L2
  1, #, #, #, #, #, #, #, #,
  2, #, #, #, #, #, #, #, #,
  3, #, #, #, #, #, #, #, #,
  4, #, #, #, #, #, #, #, #,
```

where the symbol "#" is used in place of digits. The first eleven lines give constants for the orbit transfer problem in type REAL. These have a fixed order: TOL, MU, T, GO, ISP, AO, EXO, EYO, AF, EXF, and EYF. Their descriptions follow:

```
TOL ..........  THE SOLUTION TOLERANCE
MU ...........  THE GRAVITATIONAL CONSTANT FOR THE CENTRAL BODY
T ............  THE THRUST LEVEL OF THE ROCKET MOTOR
GO ...........  EARTH'S GRAVITATIONAL ACCELERATION AT SEA-LEVEL
                (ONLY USED FOR GET MOTOR FUEL CONSUMPTION)
ISP ..........  SPECIFIC IMPULSE OF ROCKET MOTOR
AO ...........  INITIAL ORBIT SEMIMAJOR AXIS
EXO ..........  INITIAL ORBIT X-COMPONENT ECCENTRICITY
EYO ..........  INITIAL ORBIT Y-COMPONENT ECCENTRICITY
AF ...........  FINAL ORBIT SEMIMAJOR AXIS
EXF ..........  FINAL ORBIT X-COMPONENT ECCENTRICITY
EYF ..........  FINAL ORBIT Y-COMPONENT ECCENTRICITY
```

Note that these apply to the transfer as a whole, esp. when referring to the initial and final orbits. The FORTRAN FORMAT edit descriptors for each of these first eleven lines is (1X,A6,E27.20).

The PAT3D format up to this point is identical except that HXO, HYO, HZO, EXO, EYO, HXF, HYF, HZF, EXF, EYF replace AO, EXO, EYO, AF, EXF, and EYF. Their descriptions follow:

```
HXO ..........  INITIAL ORBIT X-COMPONENT ANG. MOMENTUM
HYO ..........  INITIAL ORBIT Y-COMPONENT ANG. MOMENTUM
HZO ..........  INITIAL ORBIT Z-COMPONENT ANG. MOMENTUM
EXO ..........  INITIAL ORBIT X-COMPONENT ECCENTRICITY
EYO ..........  INITIAL ORBIT Y-COMPONENT ECCENTRICITY
HXF ..........  FINAL ORBIT X-COMPONENT ANG. MOMENTUM
HYF ..........  FINAL ORBIT Y-COMPONENT ANG. MOMENTUM
HZF ..........  FINAL ORBIT Z-COMPONENT ANG. MOMENTUM
EXF ..........  FINAL ORBIT X-COMPONENT ECCENTRICITY
EYF ..........  FINAL ORBIT Y-COMPONENT ECCENTRICITY
```

For both PAT2D and PAT3D formats, the next line indicates how many intermediate transfer orbits there are. The variable NORB takes on this value. The FORTRAN FORMAT edit descriptors for this line is (1X,A6,I3). This same layout is used for the next two lines, both also containing INTEGER data. These lines specify data for the first burn. NODE is how many nodes, not counting the first one, are to be used for this burn. Specifying a "3" for NODE indicates that four lines of data will describe the burn.

The line after NODE's is for SEL. The variable SEL indicates which method should be used. Note that in the PAT2D representation above, three different values are given for SEL. A "1" indicates that the data below is in a multiple-point shooting format but Direct Collocation with Nonlinear Programming (DCNLP) should be used in the first attempt to obtain a solution. A "2" also indicates that the data below is in a multiple-point shooting format but that multiple-point shooting should be used in the first attempt to obtain a solution A "3" indicates that the data below is in a DCNLP format and DCNLP should be used in the first attempt to obtain a solution. The following table summarizes:

SEL	Guess Format	Method to try First
1	Multiple Shooting	DCNLP
2	Multiple Shooting	Multiple Shooting
3	DCNLP	DCNLP

No matter what format the data lines will be in, the line following SEL's line has the FORMAT edit descriptors (1X,A). The contents of this line are ignored.

Note that since MPMM3D cannot accept SEL=3, in PAT2D only SEL=1 or SEL=2 is acceptable.

The next NODE+1 lines are the guess data for that burn. The FORMAT edit descriptors are (1X,I3,A1,50(D27.20,A1) irrespective of which guess format is intended. Considering only PAT2D, the multiple-point shooting format has 18 elements in each line. These

elements are in the following order: INDEX, X, Y, U, V, M, LX, LY, LU, LV, LM, TF, G1, G2, G3, G4, G5, G6. "INDEX" numbers each line; the first line represents the initial point for this burn and last line represents the final point for this burn. The lines for each burn are evenly spaced. "X, Y, U, V" are the Cartesian components of the 2D position and velocity vectors, respectively. "M" is the mass. "LX, LY, LU, LV, LM" are the values of the Lagrange multiplier functions, or costates, λ_r, λ_v, and λ_m, respectively. "TF" is the length of time the burn lasts. "G1, G2, G3" are the constant Lagrange multipliers, ν_f, associated with the final boundary conditions. "G4, G5, G6" are the constant Lagrange multipliers, ν_o, associated with the initial boundary conditions.

For PAT3D, the multiple-point shooting format has 26 elements in each line. These elements are in the following order: INDEX, X, Y, Z, U, V, W, M, LX, LY, LZ, LU, LV, LW, LM, TF, G1, G2, G3, G4, G5, G6, G7, G8, G9, G10. Their meanings are simple extensions of those from PAT2D.

The DCNLP format has 9 elements in each line. These elements are in the following order: INDEX, X, Y, U, V, M, TF, L1, L2. All of these are as described above, except "L1, L2" which are the Cartesian components in the inertial frame of the thrust direction unit vector.