011482-F

CR-134169

FINAL REPORT

FOR

NASA JOHNSON SPACE CENTER

CONTRACT NAS 9-12872

TECHNIQUES FOR SHUTTLE TRAJECTORY OPTIMIZATION

by

E. R. Edge, C. J. Shieh, and W. F. Powers

Department of Aerospace Engineering
The University of Michigan
Ann Arbor, Michigan 48105

December 1973

FINAL REPORT

FOR

NASA JOHNSON SPACE CENTER

CONTRACT NAS 9-12872

TECHNIQUES FOR SHUTTLE TRAJECTORY OPTIMIZATION

by

E. R. Edge, C. J. Shieh, and W. F. Powers

Department of Aerospace Engineering
The University of Michigan
Ann Arbor, Michigan 48105

December 1973

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

This contract was concerned with the application of recently developed function-space Davidon-type techniques to the shuttle ascent trajectory optimization problem, and with an investigation of the recently developed PRAXIS algorithm for parameter optimization. The PRAXIS algorithm has been programmed into the NASA-JSC PEACE parameter optimization program, while the function-space algorithms are contained in a separate single program.

At the outset of this analysis the major deficiency of the function-space algorithms was their potential storage problems. Since most previous analyses of the methods were with relatively low-dimension problems, no storage problems were encountered. However, in shuttle trajectory optimization, storage is a problem and this problem was handled effectively. This point will be discussed further in Chapter 4.

In Chapter 2 the function-space algorithms are presented and discussed. The theory is presented in such a way that both parameter and function controls are handled naturally. Aerospace problems usually contain both types of controls.

In Chapter 3 the shuttle ascent model is presented along with the development of the particular optimization equations. In Chapter 4 the operation of the algorithm and typical simulations are presented.

In Chapter 5 variable final-time problem considerations are studied since some investigators have found the function-space accelerated-gradient methods to behave poorly on variable final-time problems. Simulations and heuristic reasoning indicate that the initial choice of t_f (in the iteration scheme), say $t_f^{(0)}$, is critical, and that a choice $t_f^{(0)} < t_f^*$ (optimal t_f) appears to improve the convergence rate considerably.

In Chapter 6 a modification of Powell's algorithm [20], developed by Brent[25], is presented and discussed. The algorithm is known as PRAXIS, and it is a parameter optimization scheme which does not require gradient information. A flow diagram of the algorithm is presented in Appendix D, and a listing is available with the NASA-JSC PEACE program. Finally, Chapter 7 presents the conclusions and recommendations for further study.

CHAPTER 2

THE ALGORITHMS

In the past few years numerous quasi-Newton type algorithms for the solution of parameter optimization problems have been extended from Euclidean spaces to infinite dimensional real Hilbert spaces. Just as in Euclidean space, the primary advantage in Hilbert space is the accelerated rate of convergence due to the building of second order information while requiring only function and gradient evaluations. Except for the conjugate gradient and gradient methods, existing function space methods cannot handle directly control variable inequality constraints. Thus applications to optimal control problems have primarily dealt with the classical Bolza problem. Since most realistic problems contain control variable inequality constraints it is desirable to be able to handle them directly in a computation scheme. In attempting to solve such problems a new function space algorithm has been generated and two existing quasi-Newton type algorithms have been modified to allow them to handle directly the bounded control problem. The modification of the algorithms was strongly influenced by the work of Pagurek and Woodside[1] on extending the conjugate gradient method to include bounded controls. The methods modified include Davidon[2] and Broyden[3] type algorithms.

2.1 The Algorithms

In this section the various algorithms will be formally stated for the problem of minimizing a real functional J(u), where u may be either finite- or infinite-dimensional. With u finite-dimensional, the formulas are applicable to the standard unconstrained parameter optimization problem. In the next section, the appropriate modifications for application to optimal control problems with bounded control variables will be presented.

In the listing below, each algorithm requires the specification of a starting vector, u_0. In addition, the Davidon and Broyden algorithms require the specification of a positive-definite, self-adjoint linear operator, H_0. Also, $\langle a,b\rangle$ and $a\rangle\langle b$ will be used to denote the inner and outer (dyadic) products, respectively, on the given Hilbert space. Note that if the space is n-dimensional Euclidean space, then $\langle a,b\rangle = a^T b$ and $a\rangle\langle b = a\, b^T$, where a, b are n-dimensional vectors. The inner and outer products for the optimal control problem will be defined in the next section.

Let g(u) denote the gradient of J, and define the update formula by

$$u_{i+1} = u_i + \alpha_i d_i, \tag{2.1}$$

where $d_i \equiv$ search direction vector and $\alpha_i \equiv$ scalar parameter defined by a one-dimensional search technique which minimizes J with respect to α.

I. Gradient Algorithm (G)

a. Calculate the search direction $d_i = -g(u_i)$.

b. Use Eq. (2.1) to calculate u_{i+1} and return to step a.

II. Conjugate Gradient Algorithm 1 (CG1)[5]

a. Calculate the search direction*

$$d_i = -g(u_i) + \beta_{i-1} d_{i-1} \tag{2.2}$$

where

$$\beta_{i-1} = \frac{\langle g_i, g_i \rangle}{\langle g_{i-1}, g_{i-1} \rangle} \tag{2.3}$$

b. Use Eq. (2.1) to calculate u_{i+1} and return to step a.

III. Davidon Algorithm (DAV)[2, 7]

a. Calculate the search direction

$$d_i = -H_i g(u_i) \tag{2.4}$$

b. Use Eq. (2.1) to calculate u_{i+1}

c. Calculate

$$s_i = u_{i+1} - u_i \tag{2.5}$$

$$y_i = g(u_{i+1}) - g(u_i) \tag{2.6}$$

d. Update H according to the following formula;

$$H_{i+1} = H_i + \frac{s_i \langle s_i}{\langle s_i, y_i \rangle} - \frac{H_i y_i \langle H_i y_i}{\langle y_i, H_i y_i \rangle} \tag{2.7}$$

e. Return to step a.

* On the first iterate (i = 0), define $d_i = -g(u_i)$.

IV. Broyden Algorithm (BRD)[3]

The same as DAV except for step d, where H is updated according to the formula

$$H_{i+1} = H_i + \left[1 + \frac{\langle y_i, H_i y_i \rangle}{\langle s_i, y_i \rangle}\right] \frac{s_i \times s_i}{\langle s_i, y_i \rangle} - \frac{s_i \times H_i y_i}{\langle s_i, y_i \rangle} - \frac{H_i y_i \times s_i}{\langle s_i, y_i \rangle} \quad (2.8)$$

2.2 Extension To Optimal Control Problems

As noted previously, in n-dimensional space the algorithms are used to minimize a scalar valued function J (u), where u is an n dimensional vector, the inner product is $< s, y> \equiv s^T y$, the dyadic operator is $s >< y \equiv sy^T$, and the H operator is an n x n matrix of scalars. Implementation of the algorithms on this type of problem is well documented in the literature. All of the algorithms described, with the exception of the (BRD) algorithm, have also been generalized to optimal control problems where g is the gradient of a functional. The primary difficulty in implementing the quasi-Newton type algorithms on optimal control problems lies in representing the infinite-dimensional H-operator.

In L_2 space the inner product is $< s, y> \equiv \int_{t_o}^{t_f} s^T y \, dt$ and the dyadic operator is $(s><y)\, u \equiv <y, u> s$ [8,9]. However, there simply is no convenient way to represent H. One way to overcome this difficulty is presented in Reference 4 by Lasdon, where it is observed that only $H_i g_i$ (not H_i itself) is needed to compute d_i. This is also true for the Broyden algorithm. To implement the Broyden algorithm, where g is the gradient of the functional, and u, s, and y are time functions, we proceed as follows:

i) H_o is taken to be any positive-definite, self-adjoint operator.

ii) Express H_i in Eq. (2.8) as a sum back to H_o. Operate on the resultant expression for H_i with g_i to obtain the following search direction:

$$d_i = -H_o g_i - \sum_{j=0}^{i-1} \left[\left(1 + \frac{\langle y_j, H_j y_j \rangle}{\langle s_j, y_j \rangle} \right) \frac{\langle s_j, g_i \rangle}{\langle s_j, y_j \rangle} \, s_j - \frac{\langle H_j y_j, g_i \rangle}{\langle s_j, y_j \rangle} \, s_j \right.$$

$$\left. - \frac{\langle s_j, g_i \rangle}{\langle s_j, y_j \rangle} \, H_j y_j \right] \qquad (2.9)$$

Equation (2.9) requires the computation of inner products of the functions $H_i y_i$, s_i, and y_i, and operating with H_o ($H_o = I$ being the simplest choice). The functions $(s_o, \ldots, s_{i-1})$ are available from past iterations. To compute the functions $H_i y_i$, we need only replace $-g_i$ by y_i in Eq.(2.9), i.e., H_i operating on y_i instead of $-g_i$. Then, for the case i-1:

$$H_{i-1} y_{i-1} = H_o y_{i-1} + \sum_{j=0}^{i-2} \left[\left(1 + \frac{\langle y_j, H_j y_j \rangle}{\langle s_j, y_j \rangle} \right) \frac{\langle s_j, y_{i-1} \rangle}{\langle s_j, y_j \rangle} \, s_j \right.$$

$$\left. - \frac{\langle H_j y_j, y_{i-1} \rangle}{\langle s_j, y_j \rangle} \, s_j - \frac{\langle s_j, y_{i-1} \rangle}{\langle s_j, y_j \rangle} \, H_j y_j \right] \qquad (2.10)$$

Thus $H_{i-1} y_{i-1}$ can be computed in a way requiring only inner products and operation with $H_o = I$, as was the case for $-H_i g_i$. Note that 2i + 4 time functions must be stored after the i iteration in order to compute the i+1 iterate, i.e.

$(s_o, \ldots, s_i)$	i+1 functions	
$(H_o y_o, \ldots, H_{i-1} y_{i-1})$	i functions	(2.11)
g_i, u_{i+1}, y_{i-1}	3 functions	

We shall now define the basic optimal control problem, and then discuss the problems of implementing the quasi-Newton algorithms. The interpretation of the above formulas and operations is more motivating in an optimal control setting.

The optimal control problem of interest is a Bolza problem with control constraints as follows:

$$\text{Minimize:} \quad J(u) = \phi(x_f) + \int_{t_o}^{t_f} L\,(t, x, u)\,dt \tag{2.12}$$

$$\text{Subject to:} \quad \dot{x} = f(t,\ x,\ y),\ x(t_o) = x_o\ (x \equiv \text{n-vector}) \tag{2.13}$$

$$|u_i| \leq c_i \quad (i = 1, \ldots, m) \tag{2.14}$$

t_o, t_f specified

Terminal conditions are included in the $\phi(x_f)$ - term and statevariable inequality Constraints are included in the $L(t,\ x,\ u)$ by the method of penalty functions.

A motivating way of viewing the quasi-Newton methods is as a class of algorithms between the first-order[10] and second-order[10, 11, 12, 13] optimal control gradient methods. The goal of a quasi-Newton algorithm is to build information about the second-variation operator without computing it explicitly, i.e., based upon gradient information only. Since only gradients and function evaluations are required for the quasi-Newton methods, we shall first outline the gradient method for optimal control problems, and then discuss the modifications for a quasi-Newton method.

In all of the algorithms, the following equations are required:

$$H = \lambda^T f(t, x, u) \tag{2.15}$$

$$\dot{\lambda} = -\frac{\partial H}{\partial x}, \quad \lambda(t_f) = \frac{\partial \phi}{\partial x_f} \tag{2.16}$$

$$g(u) = \frac{\partial H}{\partial u} \tag{2.17}$$

The function H above is the Hamiltonian, which is not to be confused with the operator H_i of the algorithms, and $g(u) = \partial H/\partial u$ is the function space gradient. The usual implementation of the standard gradient method is shown in Figure 1, where

$$u_{i+1}(t) = u_i(t) - \alpha_i H_{u_i}(t). \tag{2.18}$$

Note that the subscripts indicate the iterate number for the respective vectors; this allows less cumbersome writing of the quasi-Newton formulas.

The optimal control for the problem defined by Eqs. (2.14-16) will, in general, consist of a sequence of interior ($|u_i| < c_i$) and bounded ($|u_i| = c_i$) control component intervals. On each subarc the following conditions must be satisfied:

$$u_i = c_i \implies \frac{\partial H}{\partial u_i} \leq 0 \tag{2.19}$$

$$-c_i < u_i < c_i \implies \frac{\partial H}{\partial u_i} = 0 \tag{2.20}$$

$$u_i = -c_i \implies \frac{\partial H}{\partial u_i} \geq 0 \tag{2.21}$$

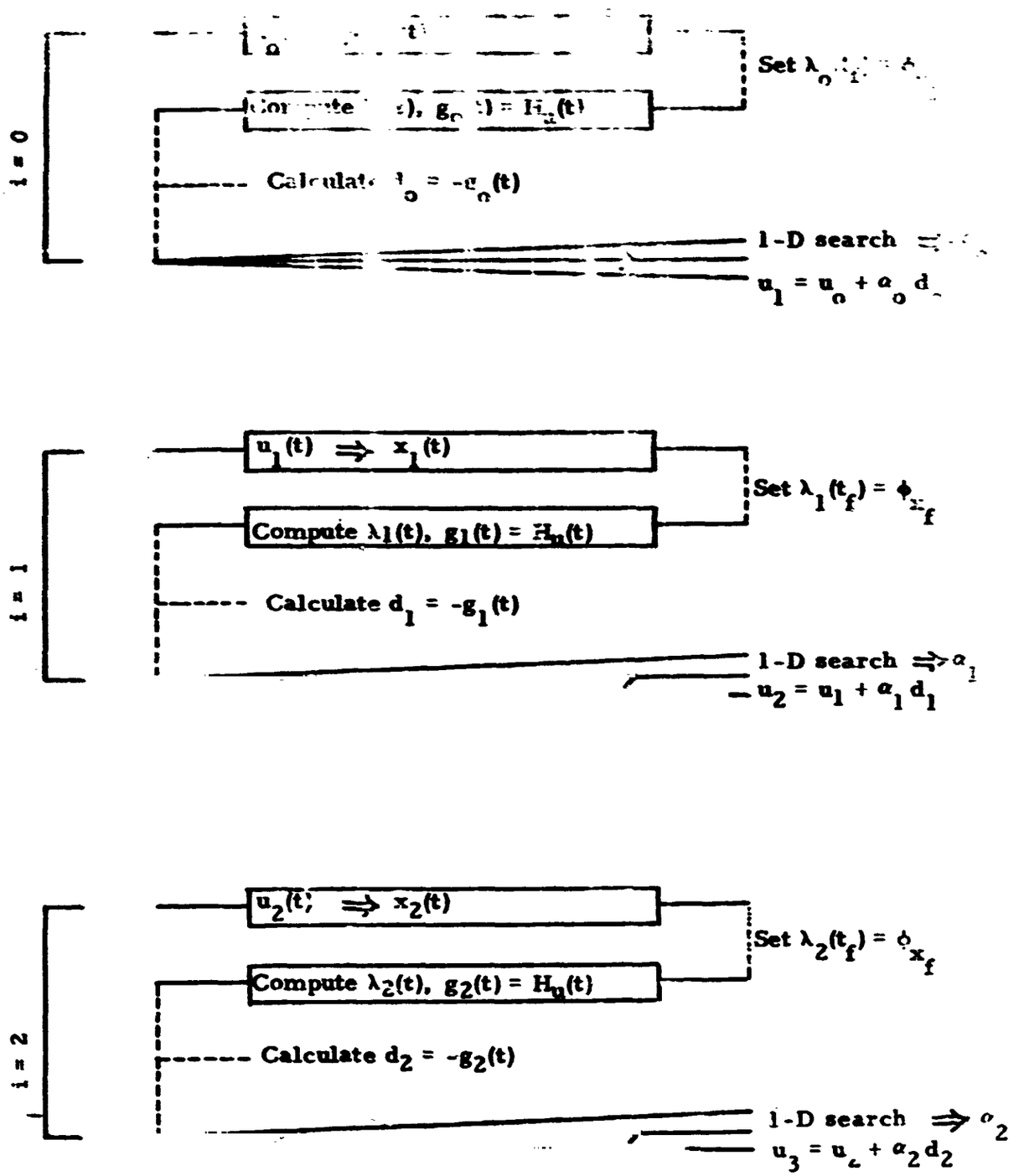


Figure 1. Flow of the Standard Gradient Method

We shall now discuss how bounded control variables are treated directly in the standard gradient method since the same basic idea is employed in the quasi-Newton methods.

As new controls are generated by varying α in Eq. (2.18), they may violate $|u_i| \leq c_i$. On these intervals u is truncated such that if $u_i > c_i$, u_i is set equal to c_i and if $u_i < -c_i$, u_i is set equal to $-c_i$. After truncation the cost associated with the given α is calculated. In this way the saturation region may change from iterate to iterate and costs are only computed for realizeable controls.

The implementation of the quasi-Newton type algorithm on unbounded control problems is shown in Figures 2 and 3. As in Figure 1 the subscripts indicate the iterate number, and Eq. (2.7) implies the Davidon algorithm and Eqs. (2.9) and (2.10) imply the Broyden algorithm.

Note that as the iteration proceeds, the number of functions stored increases. The computation time per iteration will also increase because of more inner product evaluations in the updating formulas for Hy and d. To overcome this difficulty the algorithm is restarted with a pure gradient step when i = q, where q is some predetermined interger. Pierson and Rajtora[14] demonstrated that the restart feature sometimes speeds convergence in addition to being a practical necessity. For certain problems, they found that for q small, say 3 or 4, that the convergence rate was enhanced.

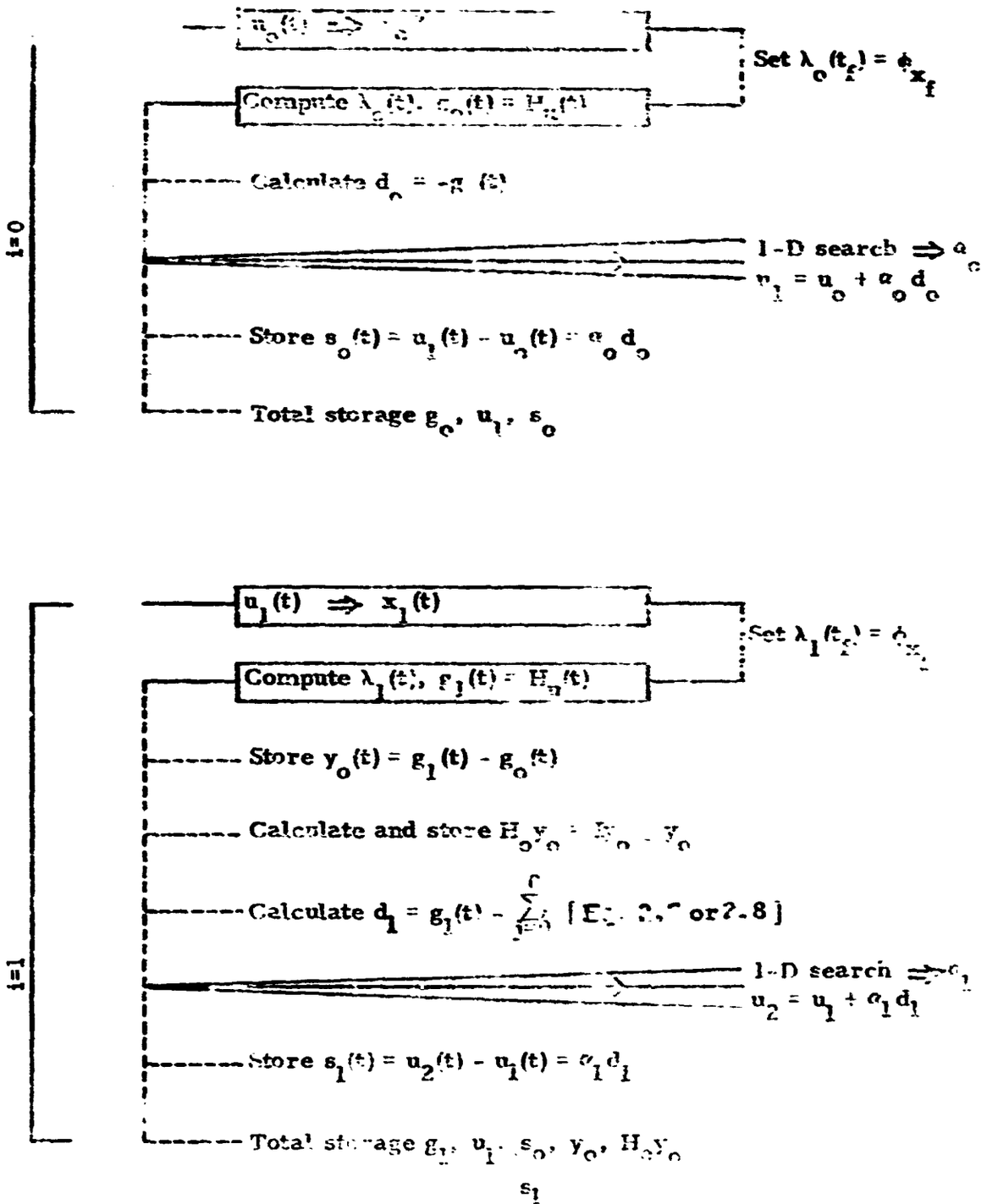


Figure 2. Flow of the Function Space Quasi-Newton Algorithms for $H_0 = I$ and $i = 0, 1$

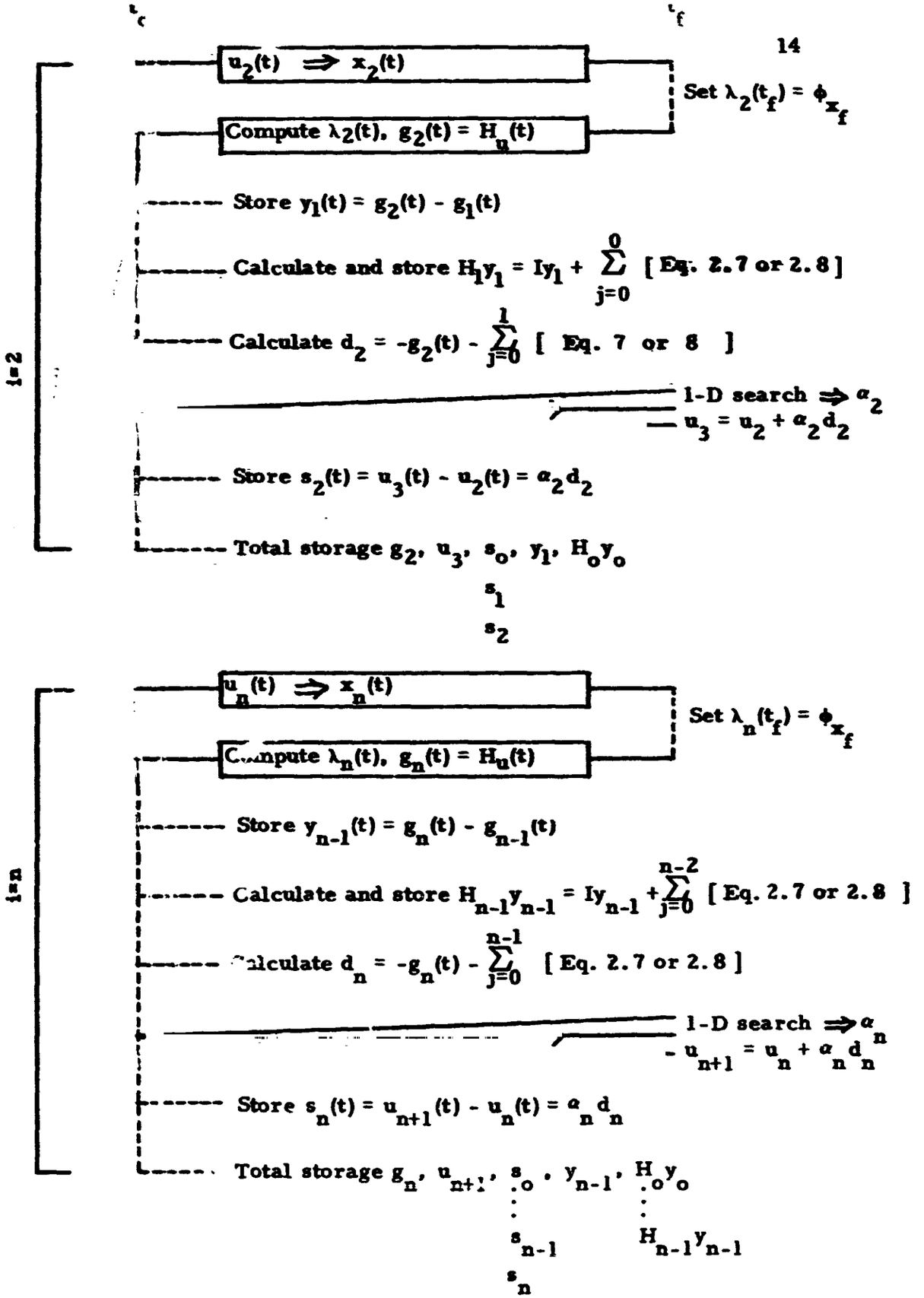


Figure 3. Flow of the Function Space Quasi-Newton Algorithms for $H_o = I$ and $i = 2, \ldots, n, \ldots$

2.3 Bounded Controls

To apply the quasi-Newton type algorithms to the bounded control problem a modification to the updating formula is required. In the interior portion of the control we wish to build second order information while second order information on the bounded portion of the control is of little use. Thus, the quasi-Newton formulas should concentrate on the interior controls, and a standard gradient formula can be used on the bounded portion of the control.

As with the gradient algorithm, as new controls are generated they are truncated before calculating the associated cost. A saturation function $w_i(t)$ identical to Pagurek and Woodside's[1] is defined. This saturation function is set equal to zero when the control is on the boundary and is set equal to unity on the interior. The saturation function is then used in the following way to compensate for our lack of freedom in choosing the control on the boundary. Instead of using q, y, and Hy in the formulas for calculating the search direction and updating Hy, we use wg, wy, and wHy. We know that $g = 0$ on the interior portion of the optimal control and this is where we wish to build second order information. On the region of saturation $g \neq 0$ (in general) and the y's (or Δg's) should not contribute to the inner products in the updating formulas. It is not necessary to apply the saturation function to s because on the saturation region, $s = \Delta u$ will already be zero.

The quasi-Newton algorithm for bounded control problems differs from the algorithm for unbounded control problems in the following ways,

i) As the 1-D search seeks the best α the associated controls are truncated before the associated cost is calculated.

ii) A saturation function $w_i(t)$ is generated after each iteration.

iii) w_g, w_y, and w_{Hy} are used in the updating formulas and in the calculation of the search direction.

2.4 Function and Parameter Controls

Some optimization problems are most naturally formulated using a combination of controls in R^n and L_2 spaces. The shuttle ascent problem developed in the following chapter is such a problem. While the approach for problems whose control space lies in either R^n or L_2^n is well developed, the theory is incomplete when a mixture of controls exists. For the general Balsa problem Eq. 2.12-14, the control is an. m vector of functions,

$$\begin{bmatrix} u_1(t) \\ \vdots \\ u_m(t) \end{bmatrix} = \bar{u} \in L_2^m [t_o, t_f]$$

and the first variation after appropriate adjoint function definitions is,

$$\delta J = \int_{t_o}^{t_f} H_u^T \, \delta u \, dt \qquad (2.22)$$

The gradient of J at the element $\bar{u}$, denoted by $\bar{g}$, is defined by the inner product relation,

$$\delta J = \frac{\partial}{\partial \epsilon} J [\bar{u}(t) + \epsilon h(t)] \Big|_{\epsilon=0} \epsilon = \langle g, h \rangle \epsilon = \langle g, \delta u \rangle \qquad (2.23)$$

$$\text{On } L_2^m [t_o, t_f] \qquad \langle v, z \rangle = \int_{t_o}^{t_f} v^T z \, dt \qquad (2.24)$$

Comparing Eqs. 2.22 and 2.23 we have,

$$\bar{g}(t) = H_u \qquad (2.25)$$

This is the usual function space gradient. It can be shown that in $L_2^m [t_o, t_f]$ the linear quadratic problem (L Q P),

$$\min \quad J = \frac{1}{2} \int_{t_o}^{t_f} [\, x^T P(t)\, x + u^T R(t)\, u]\, dt$$

$$\text{SUBJECT TO } \dot{x} = G(t)\, x + B(t)\, u \qquad (2.26)$$

$$x(t_o) = x_o \qquad x_o,\ t_o,\ t_f \text{ given}$$

in equivalent to the minimization of the unconstrained quadratic functional,

$$J = \frac{1}{2} \langle u,\ A\, u \rangle + \langle u,\ w \rangle + J_o \qquad (2.27)$$

where A, w, and J_o are appropriately defined and the inner product is defined by Eq. 2.24. It can also be shown that A is a strongly positive operator if $P(t) \geq 0$ and $R(t) > 0$ on $[\, t_o,\ t_f]$. Then, the quasi-Newton methods can be applied directly to the quadratic functional given in Eq. 2.27 for which convergence can be shown. Therefore an iterative solution to the optimal control problem of Eq. 2.26 can be generated.

In general, the nonquadratic functional is of interest. However, if the general Bolza problem can be approximated by a second-order expansion in the neighborhood of the minimizing control reasonable convergence may occur "near" the minimum.

Accepting the desirability of the approach above, consider the class of optimal control problems whose control space is composed of elements in $L_2^m [\, t_o, t_f\,]$ and R^n.

$$\left[\begin{array}{c} u_1(t) \\ \vdots \\ u_m(t) \\ \hline c_1 \\ \vdots \\ c_n \end{array}\right] = \bar{u} \in L_2^m \times R^n \qquad (2.28)$$

Where $\{u_1(t), \_\_, u_m(t)\} \in L_2^m [t_o, t_f]$

$\{c_1, \_\_\_, c_n\} \in R^n$

The first variation becomes,

$$\delta J = \int_{t_o}^{t_f} H_u^T \delta u \, dt + \int_{t_o}^{t_f} H_c^T \delta c \, dt \qquad (2.$$

Since c_i ● constant, then δc ● dc and Eq. 2.27 becomes

$$\delta J = \int_{t_o}^{t_f} H_u^T \delta u \, dt + dc^T \int_{t_o}^{t_f} H_c \, dt \qquad (2.30$$

Proceeding as in $L_2^m [t_o, t_f]$ an inner product must be defined which will imply the gradient. Then to justify, in a sense, the application of the quasi-Newton algorithms to this class of problems a suitable optimal control problem similar to the L Q P must be developed which can be reduced to an equivalent unconstrained quadratic functional. The merit of the chosen inner product is measured by its usefulness in implementing the quasi-Newton methods.

A more straightforward approach to this class of problems is based on the observation that bounded elements in R^n can be thought of as constant functions in $L_2^n [t_o, t_f]$. Thus the control can be partitioned,

$$\bar{u} = \begin{bmatrix} u_1(t) \\ \vdots \\ u_p(t) \\ \hline u_{p+1} \\ \vdots \\ u_m \end{bmatrix} \qquad (2.$$

where u_i (i = 1, . . . ,p) ϵ L_2 [t_o, t_f] and u_i (i = p + 1, . . , m) are finite constant functions. Thus the admissible control space is a subspace S of L_2^m [t_o, t_f],

$$S = \left\{ \bar{u} \mid u_i \text{ (i = 1, . ., p) } \epsilon \; L_2 \text{ [} t_o, t_f \text{] ; (i = p + 1, ... , m) finite constant functions} \right\} \qquad (2.32)$$

The goal is to find a $\bar{u}$ ϵ S which minimizes the quadratic functional of Eq. 2.27. The advantage of this approach is that all the theory which has been developed for L_2^m [t_o, t_f] still applies. The only change is that $\bar{u}$ is restricted to lie in S.

The set S is a linear subspace of L_2^m [t_o, t_f]. The usefulness of the fact that S is a linear subspace of L_2^m [t_o, t_f] lies in the following property of Hilbert spaces and quasi-Newton algorithms.

Property : Let M be a linear subspace of a Hilbert space D. There exists a mapping P:D $\rightarrow$ M called a projection operator which is linear, selfadjoint, and indempotent. If H_o of the quasi-Newton algorithms is chosen to be P and $\bar{u}_o$ ϵ M , then $\bar{u}_i$ ϵ m for all i, and

$$\lim_{k \rightarrow \infty} || H_o g_k ||^2 = 0,$$

that is, the projection of the gradient onto M tends to zero which is the condition for convergence.

Proof: Reference 2.

If a projection operator $P = L_2^m[t_o, t_f] \rightarrow S$ can be found, a consistant method for handling combinations of function and constant type controls will result.

Property: Let $\bar{A} = \begin{bmatrix} A_f(t) \\ \hline A_c(t) \end{bmatrix}$ where $\bar{A} \in L_2^m[t_o, t_f]$

and $A_f(t) \in L_2^p[t_o, t_f]$, $A_c(t) \in L_2^{m-p}[t_o, t_f]$.

Define $P: L_2^m[t_o, t_f] \rightarrow S$ by

$$P\bar{A} = P\begin{bmatrix} A_f(t) \\ \hline A_c(t) \end{bmatrix} = \begin{bmatrix} A_f(t) \\ \hline \frac{1}{\Delta t}\int_{t_o}^{t_f} A_c(t)\,dt \end{bmatrix} \quad . \qquad \Delta t = t_f - t_o \tag{2.33}$$

Then, P is linear, self-adjoint, and idempotent.

Proof: i) Linearity - $P[\alpha \bar{A} + \beta \bar{B}] = P\begin{bmatrix} \alpha A_f + \beta B_f \\ \hline \alpha A_c + \beta B_c \end{bmatrix}$

$$= \begin{bmatrix} \alpha A_f + \beta B_f \\ \hline \frac{1}{\Delta t}\int_{t_o}^{t_f} (\alpha A_c + \beta B_c)\,dt \end{bmatrix}$$

$$= \alpha P\bar{A} + \beta P\bar{B}$$

ii) Self-Adjoint - Define P^* by $\langle \bar{A}, P\bar{B} \rangle = \langle P^*\bar{A}, \bar{B} \rangle$

$$\langle \bar{A}, P\bar{B} \rangle = \int_{t_o}^{t_f} \bar{A}^T P\bar{B}\,dt = \int_{t_o}^{t_f} [A_f\ A_c] P \begin{bmatrix} B_f \\ \hline B_c \end{bmatrix} dt$$

$$= \int_{t_o}^{t_f} [A_f\ A_c] \begin{bmatrix} B_f \\ \hline \frac{1}{\Delta t}\int_{t_o}^{t_f} B_c\,dt \end{bmatrix} dt$$

$$= \int_{t_o}^{t_f} A_f^T B_f \, dt + \left(\frac{1}{\Delta t}\int_{t_o}^{t_f} A_c^T dt\right)\left(\int_{t_o}^{t_f} B_c \, dt\right)$$

$$= \int_{t_o}^{t_f} \left[A_f \quad \frac{1}{\Delta t}\int_{t_o}^{t_f} A_c \, dt\right]^T \begin{bmatrix} B_f \\ B_c \end{bmatrix}$$

$$= \langle P\bar{A}, \bar{B}\rangle \Rightarrow P^* = P$$

iii) Idempotent ($P^2 = P$)

$$P\begin{bmatrix} A_f \\ \hline A_c \end{bmatrix} = \begin{bmatrix} A_f \\ \hline \frac{1}{\Delta t}\int_{t_o}^{t_f} A_c \, dt \end{bmatrix}$$

$$P^2\begin{bmatrix} A_f \\ \hline A_c \end{bmatrix} = P\begin{bmatrix} A_f \\ \hline \frac{1}{\Delta t}\int_{t_o}^{t_f} A_c \, dt \end{bmatrix} = \begin{bmatrix} A_f \\ \hline \frac{1}{\Delta t}\int_{t_o}^{t_f}\frac{1}{\Delta t}\int_{t_o}^{t_f} A_c \, dt \, d\Gamma \end{bmatrix}$$

$$= \begin{bmatrix} A_f \\ \hline \frac{1}{\Delta t}\int_{t_o}^{t_f} A_c \, dt \end{bmatrix} = P\begin{bmatrix} A_f \\ \hline A_c \end{bmatrix} \Rightarrow P^2 = P$$

The properties developed above imply how the first variation of Eq. (2.30) should be treated in the quasi-Newton algorithms. First, viewing $(u_1(t), \ldots u_m(t), c_1, \ldots, c_n)$ as an element of $L_2^{m+n}[t_o, t_f]$, the gradient is

$$g = \begin{bmatrix} H_u \\ \hline H_c \end{bmatrix} , \qquad (2.34)$$

and an admissible choice for H_o, say $\tilde{H}_o$, is the projection operator (2.33), which implies that the initial search direction is

$$\tilde{H}_o \hat{g}_o = \begin{bmatrix} H_u^{(0)} \\ \hline \frac{1}{t_f - t_o} \int_{t_o}^{t_f} H_c^{(0)} dt \end{bmatrix} \tag{2.35}$$

However, note that this is equivalent to assuming

$$g = \begin{bmatrix} H_u \\ \hline \frac{1}{t_f - t_o} \int_{t_o}^{t_f} H_c^{(0)} dt \end{bmatrix}, \tag{2.36}$$

with $(u_1(t), \ldots, u_m(t), c_1, \ldots, c_n) \in L_2^m|t_o, t_f| \times R^n$, and $H_o = I$ since

$$H_o g_o = \begin{bmatrix} H_u^{(0)} \\ \hline \frac{1}{t_f - t_o} \int_{t_o}^{t_f} H_c^{(0)} dt \end{bmatrix} = \tilde{H}_o \tilde{g}_o. \tag{2.37}$$

Furthermore, the choice of definition for the gradient (2.36) has the same convergence properties as the choice (2.34) since

$$|| \tilde{H}_o \hat{g}_o || \to 0$$

implies

$$|| g_o || = || g_{c_o} || + || H_{u_o} || \to 0 . \tag{2.38}$$

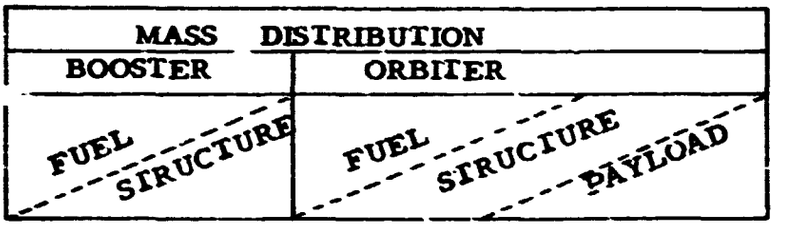
(2.35) will be utilized as the gradient expression in Chapter

CHAPTER 3

SPACE SHUTTLE ASCENT MODEL AND OPTIMIZATION

3.1 Vehicle and Mission Description

The vehicle and mission considered are taken from Reference 15. The goal is to determine a control history for the pressure-fed series burn booster/040 c orbiter which will yield maximum payload deliverable to a 50 x 100 nm. orbit inclined 28.5 degrees. The trajectory is constrained to 650 psf maximum dynamic pressure and 3.0 g maximum accleration. For trajectory purposes the mass of the vehicle can be broken down into five main subdivisions.



m_{f1} = fuel first stage = 3.50680×10^6 lb m.

m_{f2} = fuel second stage = 1.16415×10^6 lbm.

m_{s1} = structure first stage = 5.70850×10^5 lb m.

m_{s2} = structure second stage = 2.61300×10^5 lbm.

m_p = payload = quanity to be maximized

The trajectory is determined by two controls, the mass flow rate which implies the magnitude of the thrust and a thrust angle. The mass flow rate may vary from zero to maximum which implies thrust between 0 and 100%. The overall trajectory can be s[illegible] four "phases".

Each of the phases is characterized by the way in which the thrust angle is determined and by the coordinate system in which the equations of motion are being integrated.

FIRST STAGE			SECOND STAGE
COORDINATE SYSTEM SPHERICAL ROTATING APPENDIX A			POLAR
PHASE 1	PHASE 2	PHASE 3	PHASE 4
VERTICAL RISE	PITCH OVER	GRAVITY TURN	LINEAR TANGENT

The equations of motion for the first stage are integrated in a spherical coordinate system which rotates with the earth. This coordinate system was chosen because of the ease of representing initial conditions and aerodynamic forces. The general equations of motion are derived in Appendix A. Assuming the first stage engines are perfectly expanded to vacuum pressure we have,

$$\text{Thrust} = T = I_{sp_1} |\dot{m}| - P_{atm} A_{exit}$$

where $I_{sp_1} = 270.7$ sec.

$A_{exit} = 700 \text{ ft}^2$

$0 \leq |\dot{m}| \leq 3.01385 \times 10^4$ lbm/sec.

$\Longrightarrow \quad T_{max} = 8.15849 \times 10^6$ lbf

The first stage burn is further divided into three phases. They are,

i) Phase 1 - vertical rise for ten seconds

ii) Phase 2 - pitch over at a constant rate for ten seconds

iii) Phase 3 - gravity turn i.e., the thrust is parallel to the velocity. This phase terminates when all fuel is exhausted in the first stage.

Aerodynamic forces are on the order of 2% of the total forces acting on the vehicle after staging and drop off rapidly. Thus, aerodynamic forces are neglected during the second stage burn. This allows the equations of motion to be integrated in a polar coordinate system.

The equations of motion are stated in Section 3.4. This change of coordinate systems results in a new set of state variables. The equations relating the state variables before and after staging are derived in Appendix C. By integrating the equations in a polar coordinate system we have reduced the number of state variables, simplified the terminal boundary conditions, and simplified the adjoint equations. It is assumed that the second stage engines are perfectly expanded to vacuum pressure, thus,

$$\text{Thrust} = T = I_{sp_2} |\dot{m}|,$$

where $I_{sp_2} = 456.5$ sec.

$$0 \leq |\dot{m}| \leq 3.0887 \times 10^3 \text{ lbm/sec.}$$

$$\Longrightarrow T_{max} = 1.40999 \times 10^6 \text{ lbf.}$$

During second stage burn the thrust is orientated according to the linear tangent steering law, i.e.,

$$\tan \gamma = at + b, \text{ (a, b constants)} \qquad (3.1)$$

where γ is the angle between thrust vector and the local horizontal. The above discussion leads to the following overall problem:

i) Initial conditions - launch from KSC

ii) Terminal conditions - 50 x 100 nm orbit inclined 28.5 degrees with insertion at perigee.

iii) Controls and Unknown Parameters

1) Initial GLOW - a parameter.

2) Mass flow rate $\dot{m}(t)$ or, equivalently, thrust a function of time.

3) $\dot{\gamma}$ during ptich over - a parameter.

4) ψ during ptich over - a parameter control for out-of-plane thrust (explained in Section 3.2).

5) a and b - parameters used during linear tangent steering $\tan \gamma = a\ t + b$.

iv) Constraints

1) $Q_{max} \leq$ 650 psf

2) Acceleration $_{max} \leq$ 3.0 g's.

3.2 Thrust Forces

$$|\bar{T}| = |\dot{m}|\ I_{sp} - P_{atm}\ A_e$$

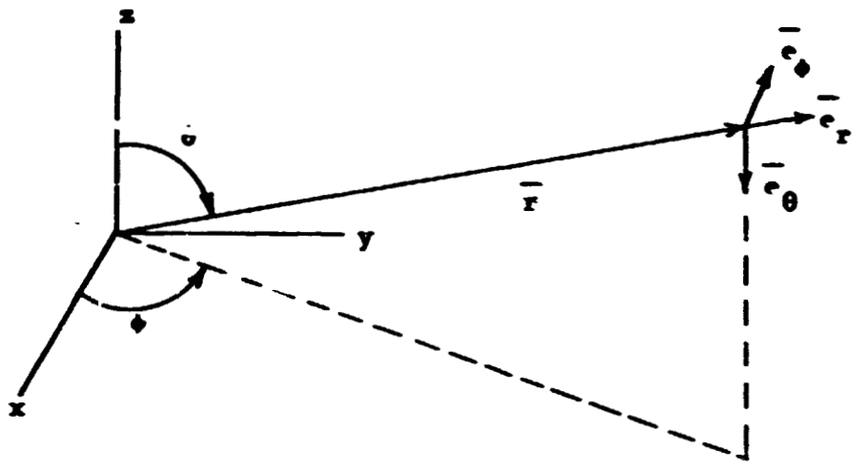


Figure 3.1 Coordinate System for First-Stage Computation

i) Vertical Rise: $\bar{T} = |T|\, \bar{e}_r$ (10 seconds)

ii) Pitch Over: Consider the trial of vectors $\bar{e}_r$, $\bar{e}_\theta$, $\bar{e}_\phi$.

The plane defined by $\bar{e}_\theta$ and $\bar{e}_\phi$ is the local horizontal plane.

The unit vector $\bar{e}_\phi$ points in the easterly direction for $\theta \neq 0$ or π.

After vertical rise, the vehicle pitches over and at the same time the plane of the orbit is determined by thrusting at some constant azimuth angle ψ. (See Fig. 3.2)

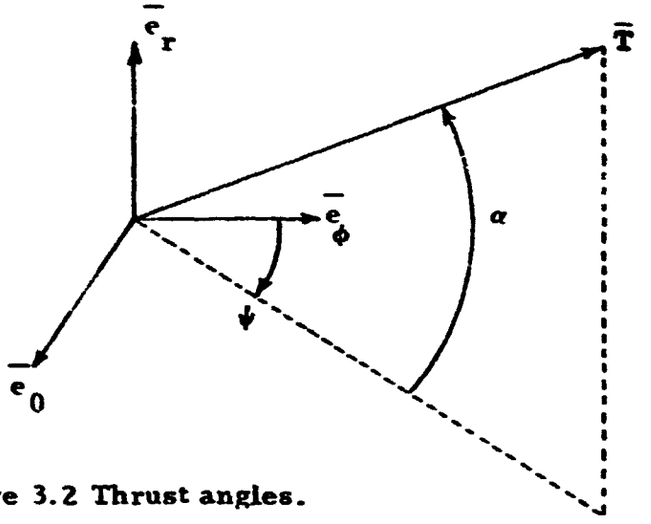


Figure 3.2 Thrust angles.

The initial thrust is in the vertical direction, i.e., $\gamma = \frac{\pi}{2}$. The vehicle then pitches over with $\dot{\gamma}$ = constant, which implies

$$\gamma = \frac{\pi}{2} - \dot{\gamma}\ (t - t_{vr}),\ (t_{vr} < t \leq 20)$$

where t_{vr} = time of pitch-over initiation. The angle ψ = Constant throughout vertical rire, and it is noted that ψ will not correspond to the final inclination. However, the final inclination will be very strongly influenced by ψ and, in fact, ψ will be the primary control which affects the final inclination angle. Thus,

$$\bar{T} = |\bar{T}| \left[\text{Sin } \gamma \, \bar{e}_r + \text{Cos } \gamma \, \text{Sin} \psi \, \bar{e}_\theta + \text{Cos } \gamma \, \text{Cos } \psi \, \bar{e}_\phi \right] \qquad (3.2)$$

$$= |\bar{T}| \left[\text{Cos } \dot{\gamma} \, (t - t_{vr}) \, \bar{e}_r + \text{Sin } \dot{\gamma} \, (t - t_{vr}) \left(\text{Sin } \psi \, \bar{e}_\theta + \text{Cos } \psi \, \bar{e}_\phi \right) \right]$$

iii) Gravity Turn: $\bar{T} \;||\; \bar{V} \quad \Rightarrow \bar{T} = |\bar{T}| \, \dfrac{\bar{V}}{|\bar{V}|}$ (3.3)

$$\bar{T} = |\bar{T}| \, \frac{u}{|\bar{V}|} \bar{e}_r + \frac{v}{|\bar{V}|} \bar{e}_\theta + \frac{w}{|\bar{V}|} \bar{e}_\phi \qquad (3.4)$$

iv) Linear Tangent: $\tan \gamma = a\,t + b$

$$\text{Sin } \gamma = \frac{\tan \gamma}{\sqrt{1 + \tan^2 \gamma}} \qquad (3.5)$$

$$(-\pi/2 \leq \gamma \leq \pi/2)$$

$$\text{Cos } \gamma = \frac{1}{\sqrt{\tan^2 \gamma + 1}} \qquad (3.6)$$

$$\bar{T} = |\bar{T}| \left[\text{Sin} \gamma \, \bar{e}_r + \text{Cos} \gamma \, \bar{e}_\theta \right]$$

3.3 Aerodynamic Forces

It is assumed that the vehicle is aligned with the local wind velocity, thus

$$\bar{A} = \text{Drag} = \bar{D} \quad - \bar{V}$$

$$|\bar{D}| = \frac{1}{2} \rho |\bar{V}|^2 A C_D$$

$C_D = C_D$ (altitude, mach number) $\qquad |\bar{V}| = \sqrt{u^2 + v^2 + w^2}$

$$\bar{A} = |\bar{A}| \frac{-\bar{V}}{|\bar{V}|}$$

$$\bar{A} = -\frac{1}{2} \rho A C_D (u^2 + v^2 + w^2) \left(\frac{u}{|\bar{V}|} \bar{e}_r + \frac{v}{|\bar{V}|} \bar{e}_\theta + \frac{w}{|\bar{V}|} \bar{e}_\phi \right)$$

$$= -\frac{1}{2} \rho A C_D \sqrt{u^2 + v^2 + w^2} \left(u\bar{e}_r + v\bar{e}_\theta + w\bar{e}_\phi \right) \qquad (3.8)$$

In all equations that follow the control notation will be,

u - mass flow rate magnitude

c_1 - GLOW (Gross Liftoff Weight)

c_2 - pitch-over rate during pitch-over phase

c_3 - a (linear tangent parameter)

c_4 - b (linear tangent parameter)

c_5 - out-of-plane thrust angle during pitch over.

3.4 Equations of Motion

(h) $$\dot{x}_1 = x_3$$

(θ) $$\dot{x}_2 = x_4 / \left(x_1 + R_o \right)$$

(u) $$\dot{x}_3 = \left(x_4^2 + x_5^2 \right) / \left(x_1 + R_o \right) - k / \left(x_1 + R_o \right)^2 + \left(x_1 + R_o \right) \Omega^2 \sin^2 x$$

$$+ 2\Omega x_5 \sin x_2 + \frac{F_1}{x_6}$$

(v) $$\dot{x}_4 = \frac{x_5^2}{\left(x_1 + R_o \right) \tan x_2} - \frac{x_3 x_4}{\left(x_1 + R_o \right)} + \left(x_1 + R_o \right) \Omega^2 \sin x_2 \cos x_2$$

$$+ 2 \Omega x_5 \cos x_2 + \frac{F_2}{x_6} \qquad (3.9)$$

(w) $$\dot{x}_5 = -\frac{x_3 x_5}{\left(x_1 + R_o \right)} - \frac{x_4 x_5}{\left(x_1 + R_o \right) \tan x_2} - 2\Omega x_4 \cos x_2$$

$$- 2 x_3 \Omega \sin x_2 + \frac{F_3}{x_6}$$

$$(\text{mass})\,\dot{x}_6 = -u$$

i) Vertical Rise: $F_1 = |\bar{T}| - Q\left(x_1, x_3\right) C_D\left(x_1, x_3\right)$

$$F_2 = 0$$

$$F_3 = 0$$

$$Q = \frac{1}{2}\rho\,(x_1)\,A\,x_3^{\,2}$$

$$|\bar{T}| = u I_{sp} - P\,(x_1)\,A_e \qquad (3.10)$$

ii) Pitch-Over: $F_1 = |\bar{T}|\,\text{Cos}\,C_2\left(t - t_{vr}\right)$

$$-Q\left(x_1, x_3, x_4, x_5\right) C_D\left(x_1, x_3, x_4, x_5\right) x_3$$

$$F_2 = |\bar{T}|\,\text{Sin}\,C_2\left(t - t_{vr}\right)\,\text{Sin}\,C_5$$

$$-Q\left(x_1, x_3, x_4, x_5\right) C_D\left(x_1, x_3, x_4, x_5\right) x_4 \qquad (3.11)$$

$$F_3 = |\bar{T}|\,\text{Sin}\,C_2\left(t - t_{vr}\right)\,\text{Cos}\,C_5$$

$$-Q\left(x_1, x_3, x_4, x_5\right) C_D\left(x_1, x_3, x_4, x_5\right) x_5$$

$$|\bar{T}| = uI_{sp} - PA_e$$

$$Q = \frac{1}{2} \rho (x_1) A \sqrt{x_3^2 + x_4^2 + x_5^2}$$

iii) Gravity Turn: $F_1 = |\bar{T}| \frac{x_3}{\sqrt{x_3^2 + x_4^2 + x_5^2}}$

$$- Q(x_1, x_3, x_4, x_5) C_D(x_1, x_3, x_4, x_5) x_3$$

$$F_2 = |\bar{T}| \frac{x_4}{\sqrt{x_3^2 + x_4^2 + x_5^2}}$$

$$- Q(x_1, x_3, x_4, x_5) C_D(x_1, x_3, x_4, x_5) x_4 \quad (3.12)$$

$$F_3 = |\bar{T}| \frac{x_5}{\sqrt{x_3^2 + x_4^2 + x_5^2}}$$

$$- Q(x_1, x_3, x_4, x_5) C_D(x_1, x_3, x_4, x_5) x_5$$

$$|\bar{T}| = uI_{sp} - P(x_1) A_e$$

$$Q = \frac{1}{2} \rho (x_1) A \sqrt{x_3^2 + x_4^2 + x_5^2}$$

iv) Linear Tangent $\quad \tilde{F}_1 = |\bar{T}| \sqrt{\frac{\Gamma^2}{\Gamma^2+1}}$

$$\tilde{F}_2 = |\bar{T}| \sqrt{\frac{1}{\Gamma^2+1}}$$

$$\text{Tan } \Gamma = C_3 t + C_4$$

$$|\vec{T}| = u I_{sp} \qquad (3.13)$$

Equations of Motion

$$\dot{\tilde{x}}_1 = \tilde{x}_2$$

$$\dot{\tilde{x}}_2 = \tilde{x}_3^2 / \left(\tilde{x}_1 + R_o \right) - k / \left(\tilde{x}_1 + R_o \right)^2 + \tilde{F}_1 / x_4$$

$$\dot{\tilde{x}}_3 = -\tilde{x}_2 \tilde{x}_3 / \left(\tilde{x}_1 + R_o \right) + \tilde{F}_2 / \tilde{x}_4$$

$$\dot{\tilde{x}}_4 = -u \qquad (3.14)$$

3.5 The First Variation

In order to apply optimization theory the problem must be stated in control notation or format. There are five parameter - type controls and one function- type control. Recall the Parameter Controls:

i) C_1 - GLOW

ii) C_2 - $\dot{\gamma}$

iii) C_3 - a

iv) C_4 - b

v) C_5 - ψ

Function Control: u (mass flow rate magnitude). The equations of motion for the system have already been derived (Appendix A) and may be symbolized by

$$\dot{x} = f(t, x, n) \qquad (0 \leq t < t_s)$$

$$\dot{\tilde{x}} = \tilde{f}(t, \tilde{x}, u) \qquad (t_s < t \leq t_f)$$

where, for convenience, u denotes the vector (C_1, ... , C_5 , u). The terminal boundary conditions will be handled by the method of quadratic penalty functions. The state variable inequality constraints will be handled by integral quadratic penalty functions. The performance index is,

$$
\begin{aligned}
J(u) = &- C_1 + P_1\left(\tilde{x}_1 (t_f) - x_{1f} \right)^2 \\
&+ P_2 \left(\tilde{x}_2 (t_f) - x_{2f} \right)^2 \\
&+ P_3 \left(\tilde{x}_3 (t_f) - x_{3f} \right)^2 \\
&+ P_4 \int_{to}^{t_s} (q-650)^2 \; U(q-650) \; dt \\
&+ P_5 \int_{to}^{t_s} (acc-3.05)^2 \; U(acc-3.05) \; dt \\
&+ P_6 \int_{t_s}^{t_f} (acc-3.00)^2 \; U(acc-3.00) \; dt \\
&+ P_7 \left(\cos\Phi(t_f) - \cos\Phi_f \right)^2
\end{aligned}
\qquad (3.15)
$$

Where Φ - inclination of orbit $\quad U(\eta) = \begin{cases} 1 & \eta > 0 \\ 0 & \eta \leq 0 \end{cases}$

q - dynamic pressure

acc - axial acceleration

P_i - penalty weighting factors

t_s - defined by fuel exhaustion 1st stage

t_f - defined by fuel exhaustion 2nd stage

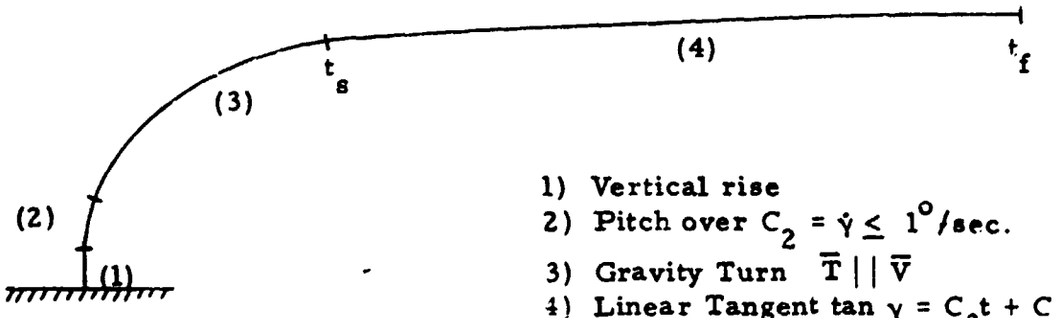


Figure 3. Phasing of Shuttle Ascent Trajectory Optimization.

The problem reduces to:

$$\text{Min } J(u) = \phi\,(x_o, x_s, \tilde{x}_f) + \int_o^{10-} L_1\,(t,x,u)\,dt + \int_{10+}^{20-} L_2(t,x,u)\,dt + \int_{20+}^{t_s^-} L_3(t,x,u)\,dt$$

$$+ \int_{t_s^+}^{t_f} L_4(t,\tilde{x},u)\,dt \qquad (3.16)$$

Subject to: $\dot{x} = f(t,x,u)$ $(t_o \le t < t_s)$

and $\dot{\tilde{x}} = \tilde{f}(t,\tilde{x},u)$ $(t_s < t \le t_f)$

to = 0 sec	10 sec	20 sec	t_s^- free	t_s^+ free	t_f free
x_1 - x_5 fixed	free	free	x_1 - x_5 free	$\tilde{x}(t_s^+) = g(x(t_s^-))$	$\tilde{x}_1$ - $\tilde{x}_3$ free
x_6 free			$\psi_s(x_6) = 0$		$\psi_f(\tilde{x}_4) = 0$
			Mass of Fuel 1st stage depleted defines t_s	Trans-formation Equations App. C	Mass of fuel 2nd stage depleted defines t_f

Define

$$H = L + \lambda^T f \text{ (on each subarc)}$$

The first variation is

$$\delta J = \phi_{xo}^T dx_o + \phi_{xs}^T dx_s + \phi_{\tilde{x}f}^T d\tilde{x}_f$$

$$+ \int_0^{0-} [H_x^T \delta x + H_u^T \delta u - \lambda^T \delta \dot{x}]\,dt$$

$$+ \int_{10+}^{20-} [H_x^T \delta x + H_u^T \delta u - \lambda^T \delta \dot{x}]\,dt$$

$$+ [H(t_s^-) - \lambda^T(t_s^-)\,\dot{x}(t_s^-)]$$

$$+\int_{20^+}^{t_s^-}\left[H_x^T\delta x+H_u^T\delta u-\lambda^T\delta\dot{x}\right]dt$$

$$-\left[\tilde{H}(t_s^+)-\tilde{\lambda}^T(t_s^+)\dot{\tilde{x}}(t_s^+)\right]\quad dt_s$$

$$+\left[\tilde{H}(t_f)-\tilde{\lambda}^T(t_f)\dot{\tilde{x}}(t_f)\right]\quad dt_f$$

$$+\int_{t_s}^{t_f}\left[\tilde{H}_x^T\delta\tilde{x}+\tilde{H}_u^T\delta u-\tilde{\lambda}^T\delta\dot{\tilde{x}}\right]dt \qquad (3.17)$$

Then, integrating $-\lambda^T\delta\dot{x}$ by parts

$$\int_{ta}^{t_b}-\lambda^T\delta\dot{x}\,dt=\lambda^T(t_a)\,\delta x(t_a)-\lambda^T(t_b)\,\delta x(t_b)+\int_{ta}^{t_b}\dot{\lambda}^T\delta x\,dt$$

and substituting into Eq. (3.17)

$$\delta J=\phi_{xo}{}^T dx_o+\phi_{xs}{}^T dx+\phi_{\tilde{x}f}{}^T d\tilde{x}_f$$

$$+\lambda^T(o)\,\delta x(0)-\lambda^T(10^-)\,\delta x(10^-)+\int_0^{10}\left[(H_x+\dot{\lambda})^T\delta x+H_u^T\delta u\right]dt$$

$$+\lambda^T(10^+)\,\delta x(10^+)-\lambda^T(20^-)\,\delta x(20^-)+\int_{10}^{20}\left[(H_x+\dot{\lambda})^T\delta x+H_u^T\delta u\right]dt$$

$$+\left[H(t_s^-)-\lambda^T(t_s^-)\,\dot{x}(t_s^-)\right]dt_s$$

$$+\lambda^T(20^+)\,\delta x(20^+)-\lambda^T(t_s^-)\,\delta x(t_s^-)+\int_{20}^{t_s}\left[Hx+\dot{\lambda})^T\delta x+H_u^T\delta u\right]dt$$

(3.18)

$$-\left[\tilde{H}(t_s^+)-\tilde{\lambda}^T(t_s^+)\dot{\tilde{x}}(t_s^+)\right]\quad dt_s$$

$$+\left[\tilde{H}(t_f)-\tilde{\lambda}^T(t_f)\dot{\tilde{x}}(t_f)\right]\quad dt_f$$

$$+\tilde{\lambda}^T(t_s^+)\,\delta\tilde{x}(t_s^+)-\tilde{\lambda}^T(t_f)\,\delta\tilde{x}(t_f)+\int_{t_s}^{t_f}\left[(\tilde{H}+\dot{\tilde{\lambda}})^T\delta\tilde{x}+\tilde{H}_u^T\delta u\right]dt$$

At t_s and t_f we need to substitute in the relationship between the variation in x and the differtial in x

$$dx = \delta x + \dot{x}\, dt$$

$$\delta x = dx - \dot{x}\, dt \qquad (3.19)$$

After substitution and reduction, the following expression is obtained

$$dJ = \left[\lambda_{x_0} + \lambda(t_0)\right]^T dx_0$$

$$- \left[\lambda_{x_s} - \lambda(t_s^-)\right]^T dx_s + \tilde{\lambda}^T(t_s^+)\, d\tilde{x}_s$$

$$- \left[\lambda_{x_f} - \lambda(t_f^-)\right]^T dx_f$$

$$+ \left[H(t_s^-) - \tilde{H}(t_s^+)\right] dt_s$$

$$- \tilde{H}(t_f^-)\, dt_f \qquad (3.20)$$

$$+ \int_{t_0}^{t_1^-} \left[H_x^T \delta x + H_u^T \delta u\right] dt$$

$$+ \int_{t_1^+}^{t_2^-} \left[H_x^T \delta x + H_u^T \delta u\right] dt$$

$$+ \int_{t_2^+}^{t_s^-} \left[H_x^T \delta x + H_u^T \delta u\right] dt$$

$$+ \int_{t_s^+}^{t_f^-} \left[\tilde{H}_{\tilde{x}}^T \delta \tilde{x} + \tilde{H}_u^T \delta u\right] dt$$

The differentials dx_s and $d\tilde{x}_s$ are related by the differential of the transformation $\tilde{x}(t_s) = \left[x(t_s^-)\right]$, i.e.,

$$d\tilde{x}_s = \left[\quad\right]$$

into Eq. (

$$\left[\phi_{x_s} - \lambda (t_s^-) \right]^T \quad dx_s + \tilde{\lambda}^T (t_s^+) \, d\tilde{x}_s = 0$$

$$\left[\phi_{x_s} - \lambda (t_s^-) \right]^T \quad dx_s + \lambda^T (t_s^+) \frac{\partial g}{\partial x}\Big|_{t_s^-} \quad dx_s = 0$$

$$\left[\phi_{x_s}^T - \lambda^T (t_s^-) + \tilde{\lambda}^T (t_s^+) \frac{\partial g}{\partial x}\Big|_{t_s^-} \right] dx_s = 0 \qquad (3.21)$$

Then, since dx_s is arbitrary

$$\lambda (t_s^-) = \tilde{\lambda} (t_s^+) \frac{\partial g}{\partial x}\Big|_{t_s^-} + \phi_{x_s} \qquad (3.22)$$

Finally,

$$\delta J = \left[\phi_{x_{60}} + \lambda_6 (t_o) - \lambda_6 (t_s^-) + \tilde{\lambda}_4 (t_s^+) - \tilde{\lambda}_4 (t_f) \right] dm_o$$
$$+ \int_0^{10-} H_u^T \delta u \, dt + \int_{10+}^{20-} H_u^T \delta u \, dt + \int_{20+}^{t_s^-} H_u^T \delta u \, dt + \int_{t_s^+}^{t_f} \tilde{H}_u^T \delta u \, dt \qquad (3.23)$$

After defining the various adjoint differential equations and boundary conditions by:

$$\dot{\lambda} = -\frac{\partial H}{\partial x} \text{ on } (to, 10), (10, 20), (20, t_s)$$

$$\dot{\tilde{\lambda}} = -\frac{\partial \tilde{H}}{\partial \tilde{x}} \text{ on } (t_s, t_f)$$

$$\tilde{\lambda}_i (t_f) = \phi_{\tilde{x}_{if}} \qquad i = 1, 2, 3 \qquad (3.24)$$

$$\tilde{H} (t_f) = 0 \implies \text{equation for } \tilde{\lambda}_4 (t_f)$$

$$\lambda (t_s^-) = \tilde{\lambda} (t_s^+) \frac{\partial g}{\partial x}\Big|_{t_s^-} + \phi_{x_s} \implies \lambda_i (t_s^+) \quad (i = 1, \ldots, 5)$$

$$H (t_s^-) = \tilde{H} (t_s^+) \implies \lambda_6 (t_s^-)$$

Assuming expansion about a nonoptional initial control estimate, the quantities δu and dm_o must be chosen to cause $\delta J < 0$. The particula-

choice is governed by the various algorithms of Chapter 2.

3.6 Adjoint Equations

In this section the particular terms necessary for the definition of the adjoint equations will be developed. First, consider the partial derivatives of the force expressions with respect to the state variables.

A. $C_D = C_D(M)$ with $M = M(x_1, x_3, x_4, x_5) = \dfrac{\sqrt{x_3^2 + x_4^2 + x_5^2}}{a(x_1)}$

$$\frac{\partial C_D}{\partial x_1} = \frac{\partial C_D}{\partial M}\frac{\partial M}{\partial x_1} = -\frac{\partial C_D}{\partial M}\frac{\sqrt{x_3^2 + x_4^2 + x_5^2}}{a^2(x_1)}\frac{\partial a}{\partial x_1} \qquad (3.25)$$

$$\frac{\partial C_D}{\partial x_i} = \frac{\partial C_D}{\partial M}\frac{\partial M}{\partial x_i} = \frac{\partial C_D}{\partial M}\frac{x_i}{\sqrt{x_3^2 + x_4^2 + x_5^2}}\frac{1}{a(x_1)} \quad i = 3, 4, 5$$

B. $Q = Q(x_1, x_3, x_4, x_5)$

i) Vertical Rise: $Q = \frac{1}{2}\rho(x_1) A x_3^2$

$$\frac{\partial Q}{\partial x_1} = \frac{A}{2} x_3^2 \frac{\partial \rho}{\partial x_1}$$

$$\frac{\partial Q}{\partial x_2} = \frac{\partial Q}{\partial x_4} = \frac{\partial Q}{\partial x_5} = \frac{\partial Q}{\partial x_6} = 0 \qquad (3.$$

$$\frac{\partial Q}{\partial x_3} = \rho A x_3 \quad \frac{\partial Q}{\partial x_i} = 0 \quad i = 4, 5, 6$$

ii) Pitch-Over and Gravity-Turn: $Q = \frac{1}{2}\rho(x_1) A \sqrt{x_3^2 + x_4^2 + x_5^2}$

$$\frac{\partial Q}{\partial x_1} = \frac{A}{2}\sqrt{x_3^2 + x_4^2 + x_5^2}\frac{\partial \rho}{\partial x_1}$$

$$\frac{\partial Q}{\partial x_2} = 0$$

$$\frac{\partial Q}{\partial x_i} = \frac{1}{2} \rho A \frac{x_i}{\sqrt{x_3^2 + x_4^2 + x_5^2}} \qquad i = 3, 4, 5$$

$$\frac{\partial Q}{\partial x_6} = 0$$

C. $|\vec{T}| = \text{fcn}\,(x_1)$

$$\frac{\partial |\bar{T}|}{\partial x_1} = -\frac{\partial p}{\partial x_1} \quad A \tag{3.28}$$

Combining the developments above:

Vertical Rise:

$$\frac{\partial F_1}{\partial x_1} = \frac{\partial T}{\partial x_1} - \frac{\partial Q}{\partial x_1} C_D - Q \frac{\partial C_D}{\partial x_1}$$

$$\frac{\partial F_1}{\partial x_2} = 0$$

$$\frac{\partial F_1}{\partial x_3} = -\frac{\partial Q}{\partial x_3} C_D - Q \frac{\partial C_D}{\partial x_3}$$

$$\frac{\partial F_1}{\partial x_i} = 0 \quad i = 4, 5, 6 \tag{3.29}$$

$$\frac{\partial F_2}{\partial x_i} = 0 \quad i = 1, \ldots, 6$$

$$\frac{\partial F_3}{\partial x_i} = 0 \quad i = 1, \ldots, 6$$

Pitch Over:

$$\frac{\partial F_1}{\partial x_1} = \frac{\partial T}{\partial x_1} \text{ Cos } C_2 (t-t_{vr}) - \left[\frac{\partial Q}{\partial x_1} C_D + Q \frac{\partial C_D}{\partial x_1}\right] x_3$$

$$\frac{\partial F_1}{\partial x_2} = 0$$

$$\frac{\partial F_1}{\partial x_3} = -Q C_D - \left[\frac{\partial Q}{\partial x_3} C_D + Q \frac{\partial C_D}{\partial x_3}\right] x_3$$

$$\frac{\partial F_1}{\partial x_4} = -\left[\frac{\partial Q}{\partial x_4} C_D + Q \frac{\partial C_D}{\partial x_4}\right] x_3 \qquad (3.30)$$

$$\frac{\partial F_1}{\partial x_5} = -\left[\frac{\partial Q}{\partial x_5} C_D + Q \frac{\partial C_D}{\partial x_5}\right] x_3$$

$$\frac{\partial F_1}{\partial x_6} = 0$$

$$\frac{\partial F_2}{\partial x_1} = \frac{\partial T}{\partial x_1} \text{ Sin } C_2 (t-t_{vr}) \text{ Sin } C_5 - \left[\frac{\partial Q}{\partial x_1} C_D + Q \frac{\partial C_D}{\partial x_1}\right] x_4$$

$$\frac{\partial F_2}{\partial x_2} = 0$$

$$\frac{\partial F_2}{\partial x_3} = -\left[\frac{\partial Q}{\partial x_3} C_D + Q \frac{\partial C_D}{\partial x_3}\right] x_4 \qquad (3.31)$$

$$\frac{\partial F_2}{\partial x_4} = -Q C_D - \left[\frac{\partial Q}{\partial x_4} C_D + Q \frac{\partial C_D}{\partial x_4}\right] x_4$$

$$\frac{\partial F_2}{\partial x_5} = -\left[\frac{\partial Q}{\partial x_5} C_D + Q \frac{\partial C_D}{\partial x_5}\right] x_4$$

$$\frac{\partial F_2}{\partial x_6} = 0$$

$$\frac{\partial F_3}{\partial x_1} = \frac{\partial T}{\partial x_1} \quad \text{Sin } C_2 \ (t-t_{vr}) \text{ Cos } C_5 - \left[\frac{\partial Q}{\partial x_1} C_D + Q \frac{\partial C_D}{\partial x_1}\right] x_5$$

$$\frac{\partial F_3}{\partial x_2} = 0$$

$$\frac{\partial F_3}{\partial x_3} = -\left[\frac{\partial Q}{\partial x_3} C_D + Q \frac{\partial C_D}{\partial x_3}\right] x_5$$

$$\frac{\partial F_3}{\partial x_4} = -\left[\frac{\partial Q}{\partial x_4} C_D + Q \frac{\partial C_D}{\partial x_4}\right] x_5 \qquad (3.32)$$

$$\frac{\partial F_3}{\partial x_5} = -Q C_D - \left[\frac{\partial Q}{\partial x_5} C_D + Q \frac{\partial C_D}{\partial x_5}\right] x_5$$

$$\frac{\partial F_3}{\partial x_6} = 0$$

Gravity Turn:

$$\frac{\partial F_1}{\partial x_1} = \frac{\partial T}{\partial x_1} \frac{x_3}{\sqrt{x_3^2 + x_4^2 + x_5^2}} - \left[\frac{\partial Q}{\partial x_1} C_D + Q \frac{\partial C_D}{\partial x_1}\right] x_3$$

$$\frac{\partial F_1}{\partial x_2} = 0$$

$$\frac{\partial F_1}{\partial x_3} = \frac{|\vec{T}|\left(x_4^2 + x_5^2\right)}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - Q C_D - \left[\frac{\partial Q}{\partial x_3} C_D + Q \frac{\partial C_D}{\partial x_3}\right] x_3$$

$$\frac{\partial F_1}{\partial x_4} = -\frac{|\vec{T}| x_3 x_4}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - \left[\frac{\partial Q}{\partial x_4} C_D + Q \frac{\partial C_D}{\partial x_4}\right] x_3 \qquad (3.33)$$

$$\frac{\partial F_1}{\partial x_5} = -\frac{|\vec{T}| x_3 x_5}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - \left[\frac{\partial Q}{\partial x_5} C_D + Q \frac{\partial C_D}{\partial x_5}\right] x_3$$

$$\frac{\partial F_1}{\partial x_6} = 0$$

$$\frac{\partial F_2}{\partial x_1} = \frac{\partial T}{\partial x_1} \frac{x_4}{\sqrt{x_3^2 + x_4^2 + x_5^2}} - \left[\frac{\partial \Omega}{\partial x_1} C_D + \Omega \frac{\partial C_D}{\partial x_1}\right] x_4$$

$$\frac{\partial F_2}{\partial x_2} = 0$$

$$\frac{\partial F_2}{\partial x_3} = -\frac{|\vec{T}| x_3 x_4}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - \left[\frac{\partial \Omega}{\partial x_3} C_D + \Omega \frac{\partial C_D}{\partial x_3}\right] x_4 \qquad (3.34)$$

$$\frac{\partial F_2}{\partial x_4} = \frac{|\vec{T}| \left(x_3^2 + x_5^2\right)}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - \Omega C_D - \left[\frac{\partial \Omega}{\partial x_4} C_D + \Omega \frac{\partial C_D}{\partial x_4}\right] x_4$$

$$\frac{\partial F_2}{\partial x_5} = -\frac{|\vec{T}| x_4 x_5}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - \left[\frac{\partial \Omega}{\partial x_5} C_D + \Omega \frac{\partial C_D}{\partial x_5}\right] x_4$$

$$\frac{\partial F_2}{\partial x_6} = 0$$

$$\frac{\partial F_3}{\partial x_1} = \frac{\partial T}{\partial x_1} \sqrt{\frac{x_5}{x_3^2 + x_4^2 + x_5^2}} - \left[\frac{\partial \Omega}{\partial x_1} C_D + \Omega \frac{\partial C_D}{\partial x_1}\right] x_5$$

$$\frac{\partial F_3}{\partial x_2} = 0$$

$$\frac{\partial F_3}{\partial x_3} = -\frac{|\vec{T}| x_3 x_5}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - \left[\frac{\partial \Omega}{\partial x_3} C_D + \Omega \frac{\partial C_D}{\partial x_3}\right] x_5 \qquad (3.35)$$

$$\frac{\partial F_3}{\partial x_4} = -\frac{|\vec{T}| x_4 x_5}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - \left[\frac{\partial \Omega}{\partial x_4} C_D + \Omega \frac{\partial C_D}{\partial x_4}\right] x_5$$

$$\frac{\partial F_3}{\partial x_5} = -\frac{|\vec{T}| \left(x_3^2 + x_4^2\right)}{\left(x_3^2 + x_4^2 + x_5^2\right)^{3/2}} - \Omega C_D - \left[\frac{\partial \Omega}{\partial x_5} C_D + \Omega \frac{\partial C_D}{\partial x_5}\right] x_5$$

$$\frac{\partial F_3}{\partial x_6} = 0$$

Linear Tangent

$$\frac{\partial \tilde{F}_i}{\partial \tilde{x}_j} = 0 \qquad i = 1,2 \quad j = 1,\ldots,4 \tag{3.36}$$

Second, consider the portions of the adjoint equations due to the performance index without the equations of motion adjoined (i.e., due to the Lagrangian terms, L).

A. Vertical - Rise, Pitch - Over, and Gravity Turn

$$L = \left[q - AA(25)\right]^2 AA(37) + \left[ACC - AA(26)\right]^2 AA(38)$$

where

$$AA(25) = q \max$$

$$AA(37) = \begin{cases} P_4 & [q - AA(25)] > 0 \\ 0 & [q - AA(25)] \leq 0 \end{cases}$$

$$AA(26) = 3.05$$

$$AA(38) = \begin{cases} P_5 & [ACC - AA(26)] > 0 \\ 0 & [ACC - AA(26)] \leq 0 \end{cases} \tag{3.37}$$

$$q = \frac{1}{2}\left(x_3^2 + x_4^2 + x_5^2\right)$$

$$ACC = \left[u I_{sp} - P(x_1) A_e - q A C_D\right] / x_6$$

$$\frac{\partial L}{\partial x} = 2\left[q - AA(25)\right] \frac{\partial q}{\partial x_i} AA(37) + 2\left[ACC - AA(26)\right] \frac{\partial ACC}{\partial x_i} AA(38) \text{ for } (i=1,$$

$$\frac{\partial q}{\partial x_1} = \frac{1}{2} \frac{\partial \rho}{\partial x_i}\left(x_3^2 + x_4^2 + x_5^2\right)$$

$$\frac{\partial q}{\partial x_2} = 0$$

$$\frac{\partial q}{\partial x_3} = \rho x_3 \qquad \frac{\partial ACC}{\partial x_i} = \left[\frac{\partial}{\partial x_i}\right.$$

$$\frac{\partial q}{\partial x_4} = \rho x_4 \qquad \frac{\partial ACC}{\partial x_i} =$$

$\frac{\partial q}{\partial x_3}$ Cont.

$$\frac{\partial ACC}{\partial x_3} = -\left[Aq \frac{\partial C_D}{\partial x_3} + A C_D \frac{\partial q}{\partial x_3}\right] / x_6$$

$$\frac{\partial ACC}{\partial x_4} = -\left[Aq \frac{\partial C_D}{\partial x_4} + A C_D \frac{\partial q}{\partial x_4}\right] / x_6 \qquad (3.$$

$$\frac{\partial ACC}{\partial x_5} = -\left[Aq \frac{\partial C_D}{\partial x_5} + A C_D \frac{\partial q}{\partial x_5}\right] / x_6$$

$$\frac{\partial ACC}{\partial x_6} = -\frac{1}{x_6^2}\left[u I_{sp} - PA_e - q AC_D\right] = -ACC/x_6$$

$$\frac{\partial L}{\partial x_6} = -2\ [\ ACC\text{-}AA(26)\ ]\ ACC\ AA(38)\ /\ x_6$$

B. Linear Tangent

$$L = [\ ACC\text{-}AA(27)]^2\ AA(39)$$

$$ACC = [\ U I_{sp}]\ /\tilde{x}_4$$

$$\frac{\partial ACC}{\partial \tilde{x}_4} = -(u I_{sp})/\tilde{x}_4^2 = -ACC/\tilde{x}_4 \qquad (3.$$

$$\frac{\partial L}{\partial \tilde{x}_i} = 0 \qquad = 1, 2, 3$$

$$\frac{\partial L}{\partial \tilde{x}_4} = 2\ [\ ACC\text{-}AA(27)]\ \frac{\partial ACC}{\partial \tilde{x}_4}\ AA(39)$$

$$= -2\ [\ ACC\text{-}AA(27)]\ AA(39)\ ACC\ /\ \tilde{x}_4$$

Adjoint Equations for Vertical Rise, Pitch Over, and Gravity Turn

$$-\dot{\lambda}_1 = \frac{\partial L}{\partial x_1} + \lambda_2 \left[-x_4 / (x_1 + R_o)^2 \right]$$

$$+ \lambda_3 \left[-(x_4^2 + x_5^2)/(x_1 + R_o)^2 + 2k/(x_1 + R_o)^3 + \Omega^2 \sin^2 x_2 + \frac{\partial F_1}{\partial x_1} / x_6 \right]$$

$$+ \lambda_4 \left[-x_5^2 / \left[(x_1 + R_o)^2 \tan x_2 \right] + x_3 x_4 / (x_1 + R_o)^2 + \Omega^2 \sin x_2 \cos x_2 \right.$$

$$\left. + \frac{\partial F_2}{\partial x_1} / x_6 \right] \qquad (3.41)$$

$$+ \lambda_5 \left[x_3 x_5 / (x_1 + R_o)^2 + x_4 x_5 / \left[(x_1 + R_o)^2 \tan x_2 \right] + \frac{\partial F_3}{\partial x_1} / x_6 \right]$$

$$-\dot{\lambda}_2 = \frac{\partial L}{\partial x_2} + \lambda_3 \left[2 (x_1 + R_o) \Omega^2 \sin x_2 \cos x_2 + 2 \Omega x_5 \cos x_2 \right]$$

$$+ \lambda_4 \left[-(x_5^2 \csc^2 x_2 / (x_1 + R_o) + (x_1 + R_o) \Omega^2 (\cos^2 x_2 - \sin^2 x_2) - 2 \Omega x_5 \sin x_2 \right]$$

$$+ \lambda_5 \left[x_4 x_5 \csc^2 x^2 / (x_1 + R_o) + 2 \Omega x_4 \sin x_2 - 2 x_3 \Omega \cos x_2 \right] \qquad (3.42)$$

$$-\dot{\lambda}_3 = \frac{\partial L}{\partial x_3} + \lambda_1 + \lambda_3 \left[\frac{\partial F_1}{\partial x_3} / x_6 \right]$$

$$+ \lambda_4 \left[-x_4 / (x_1 + R_o) + \frac{\partial F_2}{\partial x_3} / x_6 \right]$$

$$+ \lambda_5 \left[-x_5 / (x_1 + R_o) - 2 \Omega \sin x_2 + \frac{\partial F_3}{\partial x_3} / x_6 \right] \qquad (3.43)$$

$$-\dot{\lambda}_4 = \frac{\partial L}{\partial x_4} + \lambda_2 (x_1 + R_o)^{-1}$$

$$+ \lambda_3 \left[(2 x_4 / (x_1 + R_o) + \frac{\partial F_1}{\partial x_4} / x_6 \right]$$

$$+ \lambda_4 \left[-x_3 / (x_1 + R_o) + \frac{\partial F_2}{\partial x_4} / x_6 \right]$$

$$+ \lambda_5 \left[-x_5 / \left[(x_1 + R_o) \tan x_2 \right] - 2 \Omega \cos x_2 + \frac{\partial F_3}{\partial x_4} / x_6 \right] \qquad (3.44)$$

$$-\dot{\lambda}_5 = \frac{\partial F}{\partial x_5} + \lambda_3 \left[2x_5/(x_1 + R_o) + 2\Omega \operatorname{Sin} x_2 + \frac{\partial F_1}{\partial x_5} / x_6 \right]$$

$$+ \lambda_4 \left[2x_5/[(x_1+R_o)\tan x_2] + 2\Omega \operatorname{Cos} x_2 + \frac{\partial F_2}{\partial x_5} / x_6 \right]$$

$$+ \lambda_5 \left[-x_3/(x_1 + R_o) - x_4/[(x_1+R_o)\tan x_2] + \frac{\partial F_3}{\partial x_5}/x_6 \right] \quad (3.45)$$

$$-\dot{\lambda}_6 = \frac{\partial L}{\partial x_6} - \lambda_3 F_1/x_6^2$$

$$- \lambda_4 F_2/x_6^2$$

$$- \lambda_5 F_3/x_6^2 \quad (3.46)$$

Adjoint Equations for Linear Tangent Phase

$$-\dot{\tilde{\lambda}}_1 = \frac{\partial L}{\partial \tilde{x}_1} + \tilde{\lambda}_2 \left[-\tilde{x}_3^2/(\tilde{x}_1 + R_o)^2 + 2k/(\tilde{x}_1 + R_o)^3 \right]$$

$$+ \tilde{\lambda}_3 \left[\tilde{x}_2 \tilde{x}_3/(\tilde{x}_1 + R_o)^2 \right] \quad (3.47)$$

$$-\dot{\tilde{\lambda}}_2 = \frac{\partial L}{\partial \tilde{x}_2} + \tilde{\lambda}_1 + \tilde{\lambda}_3 \left[-\tilde{x}_3/(\tilde{x}_1 + R_o) \right] \quad (3.48)$$

$$-\dot{\tilde{\lambda}}_3 = \frac{\partial L}{\partial \tilde{x}_3} + \tilde{\lambda}_2 \left[2\tilde{x}_3/(\tilde{x}_1 + R_o) \right]$$

$$+ \tilde{\lambda}_3 \left[-\tilde{x}_2/(\tilde{x}_1 + R_o) \right] \quad (3.49)$$

$$-\dot{\tilde{\lambda}}_4 = \frac{\partial L}{\partial \tilde{x}_4} - \tilde{\lambda}_2 \tilde{F}_1/\tilde{x}_4^2 - \tilde{\lambda}_3 \tilde{F}_2/\tilde{x}_4^2 \quad (3.50)$$

3.7 Gradients

The gradients with respect to the control function, u(t), and control parameters $c_1, \ldots, c_5$ are developed below.

u (t): $H_u = 2$ [ACC-AA(26)] I_{sp} AA(38)/W

$$+ \lambda_3 \frac{\partial F_1}{\partial u} / x_6$$

$$+ \lambda_4 \frac{\partial F_2}{\partial u} / x_6$$

(Vertical rise, Pitch-Over, Gravity Turn)

$$\frac{\partial \tilde{F}_3}{\partial u} \Big/ \tilde{x}_5$$

(3.54)

$$\tilde{H}_u = 2\,[\ \text{ACC-AA(27)}\ ] \qquad \text{[ACC-AA(27)]}$$

$$+\tilde{\lambda}_2 \frac{\partial \tilde{F}_1}{\partial u} \Big/ \tilde{x}_4$$

$$+\tilde{\lambda}_3 \frac{\partial \tilde{F}_2}{\partial u} \Big/ \tilde{x}_4 \qquad \text{(Linear Tangent)}$$

$$-\tilde{\lambda}_4$$

(3.56)

The components for these gradients are as follows:

(i) Vertical Rise:

$$\frac{\partial F_1}{\partial u} = I_{sp}$$

$$\frac{\partial F_2}{\partial u} = 0$$

$$\frac{\partial F_3}{\partial u} = 0$$

(3.57)

(ii) Pitch-Over:

$$\frac{\partial F_1}{\partial u} = I_{sp} \text{ Cos } c_2 (t-t_{vr})$$

$$\frac{\partial F_2}{\partial u} = I_{sp} \text{ Sin } c_2 (t-t_{vr}) \text{ Sin } c_5$$

$$\frac{\partial F_3}{\partial u} = I_{sp} \text{ Sin } c_2 (t-t_{vr}) \text{ Cos } c_5$$

(3.58)

(iii) Gravity Turn:

$$\frac{\partial F_1}{\partial u} = \frac{I_{sp}\, x_3}{\sqrt{x_3^2 + x_4^2 + x_5^2}}$$

$$\frac{\partial F_2}{\partial u} = \frac{I_{sp}\, x_4}{\sqrt{x_3^2 + x_4^2 + x_5^2}}$$

$$\frac{\partial F_3}{\partial u} = \frac{I_{sp}\, x_5}{\sqrt{x_3^2 + x_4^2 + x_5^2}}$$

(3.59)

(iv) Linear Tangent:

$$\frac{\partial \tilde{F}_1}{\partial u} = I_{sp} \frac{\Gamma}{\sqrt{\Gamma^2 + 1}}$$

$$\frac{\partial \tilde{F}_2}{\partial u} = \frac{I_{sp}}{\sqrt{\Gamma^2 + 1}} \tag{3.56}$$

$\underline{c_1}$: $$H_{c_1} = -\left[\phi\, x_{60} + \lambda_6 (t_o) - \lambda_6 (t_s^-) + \tilde{\lambda}_4 (t_s^+) - \tilde{\lambda}_4 (t_f) \right] \tag{3.57}$$

$\underline{c_2}$: $$H_{c_2} = \lambda_3 \frac{\partial F_1}{\partial c_2} / x_6 + \lambda_4 \frac{\partial F_2}{\partial c_2} / x_6 + \lambda_5 \frac{\partial F_3}{\partial c_2} / x_6$$

$$\frac{\partial F_1}{\partial c_2} = -|\vec{T}| \left[\text{Sin } c_2 (t - t_{vr}) \right] (t - t_{vr})$$

$$\frac{\partial F_2}{\partial c_2} = |\vec{T}| \left[\text{Cos } c_2 (t - t_{vr}) \right] (t - t_{vr}) \text{ Sin } c_5$$

$$\frac{\partial F_3}{\partial c_2} = |\vec{T}| \left[\text{Cos } c_2 (t - t_{vr}) \right] (t - t_{vr}) \text{ Cos } c_5$$

$$|\vec{T}| = u I_{sp} - P(x_1) A_e \tag{3.58}$$

$\underline{c_3}$: $$\tilde{H}_{c_3} = (\tilde{\lambda}_2 / \tilde{x}_4) \frac{\partial \tilde{F}_1}{\partial c_3} + (\tilde{\lambda}_3 / \tilde{x}_4) \frac{\partial \tilde{F}_2}{\partial c_3}$$

$$\frac{\partial \tilde{F}_1}{\partial c_3} = |\vec{T}| \frac{t}{(\Gamma^2 + 1)^{3/2}}$$

$$\frac{\partial \tilde{F}_2}{\partial c_3} = |\vec{T}| \left[\frac{-\Gamma t}{(\Gamma^2 + 1)^{3/2}} \right] \tag{3.59}$$

$\underline{c_4}$: $$\tilde{H}_{c_4} = \tilde{\lambda}_2 \frac{\partial \tilde{F}_1}{\partial c_4} / \tilde{x}_4 + \tilde{\lambda}_3 \frac{\partial F_2}{\partial c_4} / \tilde{x}_4$$

$$\frac{\partial \tilde{F}_1}{\partial c_4} = |T| / (\Gamma^2 + 1)^{3/2}$$

$$\frac{\partial \tilde{F}_2}{\partial c_4} = -|\vec{T}| \, \Gamma / (\Gamma^2 + 1)^{3/2} \tag{3.60}$$

$\underline{c_5}$: $$H_{c_5} = \frac{\lambda_3}{x_6}\frac{\partial F_1}{\partial c_5} + \frac{\lambda_4}{x_6}\frac{\partial F_2}{\partial c_5} + \frac{\lambda_5}{x_6}\frac{\partial F_3}{\partial c_5}$$

$$\frac{\partial F_1}{\partial c_5} = 0$$

$$\frac{\partial F_2}{\partial c_5} = |\vec{T}| \text{ Sin } c_2 (t-t_{vr}) \text{ Cos } c_5$$

$$\frac{\partial F_3}{\partial c_5} = -|\vec{T}| \text{ Sin } c_2 (t-t_{vr}) \text{ Sin } c_5 \qquad (3.61)$$

3.8 Adjoint Function Boundary Conditions

$\underline{t_f}$: $$\lambda_{1f} = 2\left[\hat{x}_1 (t_f) - x_{1f}\right] P_1$$

$$\lambda_{2f} = 2\left[\hat{x}_2 (t_f) - x_{2f}\right] P_2$$

$$\lambda_{3f} = 2\left[\tilde{x}_3 (t_f) - x_{3f}\right] P_3$$

$$\tilde{H}(t_f) = 0 \Rightarrow \tilde{\lambda}_{4f} \qquad (3.62)$$

$\underline{t_s}$: $$\left[\lambda_1 \ldots . \lambda_5\right]_{t_s} = \left[\hat{\lambda}_1 \hat{\lambda}_2 \hat{\lambda}_3\right]_{t_s} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ B_{31} & B_{32} & B_{33} & B_{34} & B_{35} \end{bmatrix} + \phi_x^T (t_s)$$

where $B_{ij} = \frac{\partial g_i}{\partial x_j}$

$$H(t_s^-) = \tilde{H}(t_s^+) \Rightarrow \lambda_6 (t_s) \qquad (3.63)$$

In order to calculate $\lambda^T (t_s) B_{3_j}$ ($j = 1, \ldots, 5$) and $\phi_x (t_s)$ are required. First, consider the equations for B_{3_j} ($j = 1, \ldots 5$).

Define: $\bar{x}_1 = x_1 + R_o$

$$g_3 = \sqrt{x_4^2 + (x_5 + x_1 \Omega \sin x_2)^2} = \left[x_4^2 + (\quad)^2\right]^{\frac{1}{2}}$$

$$B_{31} = \frac{\partial g_3}{\partial x_1} = \frac{1}{2}\left[\quad\right]^{\frac{1}{2}} \; 2\,(\quad)\;\Omega \sin x_2$$

$$B_{32} = \frac{1}{2}\left[\quad\right]^{-\frac{1}{2}} \; 2\,(\quad)\;\bar{x}_1 \Omega \cos x_2$$

$$B_{33} = 0$$

$$B_{34} = \frac{1}{2}\left[\quad\right]^{-\frac{1}{2}} 2x_4$$

$$B_{35} = \frac{1}{2}\left[\quad\right]^{-\frac{1}{2}} 2\,(\quad) \qquad (3.64)$$

Finally, consider the contribution due to $\phi_{x_2}(t_s^-)$, **where**

$$\phi_{\kappa}(t_s^-) = \left[\cos\left[\Phi(t_s^-)\right] - \cos \Phi_{t_s^-}\right] P_7$$

$$= \left[\frac{\left(x_5 + \bar{x}_1 \Omega \sin x_2\right) \sin x_2}{\left[x_4^2 + \left(x_5 + \bar{x}_1 \Omega \sin x_2\right)^2\right]^{\frac{1}{2}}} - \cos \Phi_{t_s^-}\right] P_7 = \eta^2 P_7 \qquad (3.65)$$

Then,

$$\frac{\partial \phi}{\partial x_i} = 2\eta \frac{\partial \eta}{\partial x_i} P_7 \qquad (3.66)$$

Define $\bar{x}_1 = R_o + x_1$, . **Then**

$$\eta = \frac{\left(x_5 + \bar{x}_1 \Omega \sin x_2\right) \sin x_2}{\left[x_4^2 + \left(x_5 + \bar{x}_1 \Omega \sin x_2\right)^2\right]^{\frac{1}{2}}} - \cos \Phi_{t_s^-}$$

$$\frac{\partial \eta}{\partial x_1} = \frac{\Omega \sin^2 x_2}{\left[\quad\right]^{\frac{1}{2}}} - \frac{(\quad) \sin x_2}{\left[\quad\right]^{3/2}} \; 2\,(\quad)\;\Omega \sin x_2$$

$$\frac{\partial \eta}{\partial x_2} = \frac{x_5 \cos x_2 + 2\bar{x}_1 \Omega \sin x_2 \cos x_2}{\left[\quad\right]^{1/2}}$$

$$- \frac{1}{2} \frac{(\quad) \sin x_2}{\left[\quad\right]^{3/2}} \; 2\,(\quad)\;\bar{x}_1 \Omega \cos x_2$$

$$\frac{\partial \eta}{\partial x_3} = 0 \qquad (3.67)$$

$$\frac{\partial \eta}{\partial x_4} = -\frac{1}{2} \left[\qquad \right]^{3/2} \left(\qquad \right) \sin x_2 \; 2 x_4$$

$$\frac{\partial \eta}{\partial x_5} = \frac{\sin x_2}{\left[\quad \right]^{1/2}} - \frac{1}{2} \left[\qquad \right]^{3/2} \left(\qquad \right) \sin x_2 \; 2 \left(\qquad \right)$$

$$\frac{\partial \eta}{\partial x_6} = 0$$

COMPUTER IMPLEMENTATION

The computer implementation of quasi-Newton algorithms for the solution of Bolza problems was discussed in Chapter 2. In Chapter 3 the shuttle ascent optimization problem was formulated and was not precisely in the Bolza form. In particular the performance index is composed of four integrals all of which have different equations of motion. Further, the staging and final times are not known but implied by state variable terminal constraints, $\psi(x_f) = 0$. Because terminal boundary conditions and state variable inequality constraints are handled by penalty methods it has been found that a considerable savings in computer time can be achieved by real time human interaction with the executing program by way of a CRT display terminal. Problems associated with the storage requirements of the quasi-Newton algorithms also had to be solved. One important element of all the algorithms described in Chapter 2 is the one dimensional search (1-D search). A large percentage of CPU time is consumed in the 1-D search. Its accuracy and efficiency have a great effect on the success of the algorithms. These various practical considerations will be discussed in this chapter along with a full description of the program itself.

4.1 Flow Diagram

On the following page is a flow diagram which is used to assist trajectory optimization program. The sequence of events is similiar to those presented in Chapter 3 with the addition of a CRT display terminal for human operator interaction with the program while it is executing. The main iteration loop consists of the forward integration, backward integration, calculation of search direction, 1-D search, and convergence check. These operations are done repeatedly for a given set of penalty coefficients until an "acceptable" degree of convergence is obtained. At this point the human operator evaluates the final control and associated trajectory graphicaly on the CRT display. The penalty coefficient values are appropriately changed by the operator and the iteration loop is re-entered. This process is repeated until the operator is satisfied that a further increase of penalty coefficients will not yield a "better" control and execution is terminated. The final control and associated trajectory are then plotted by Calcomp for future analysis.

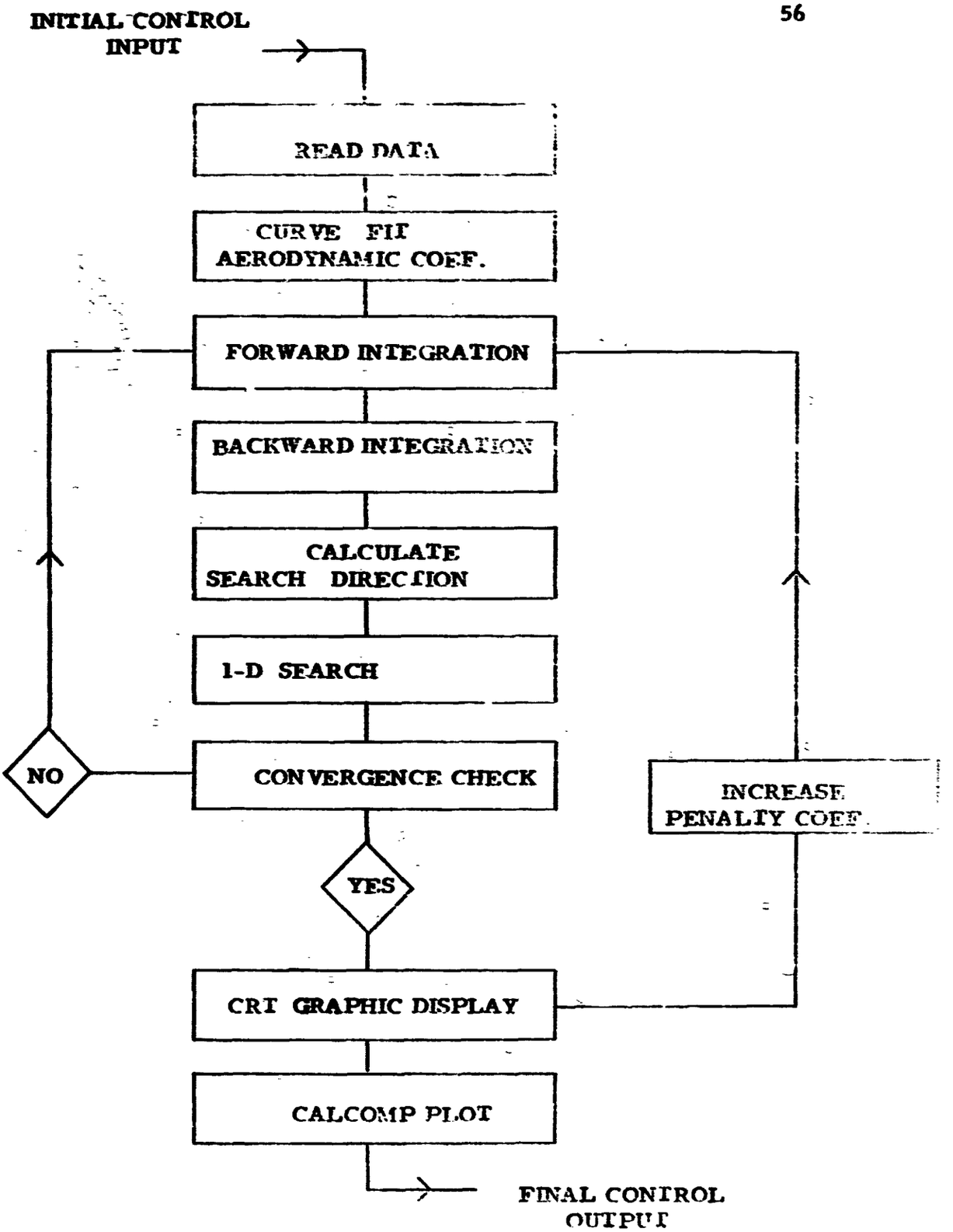


Figure 4.1 **Flow Diagram**

4.2 Staging and Final Time

The shuttle ascent trajectory optimization problem is formulated in such a way that t_s and t_f the staging and final times are free. The cutoff condition in both stages is determined by propellent exhaustion. Recall that

$$\text{Thrust} = I_{sp}\, u$$

where u(t), the magnitude of the mass flow rate of propellent, is one of the controls to be optimized. All direct numerical methods require a first guess for u(t). This guess is usually stored pointwise at certain known times. Consider the boost phase of the trajectory. Assume u(t) is stored at n equally spaced storage locations and is piecewise linear between storage locations.

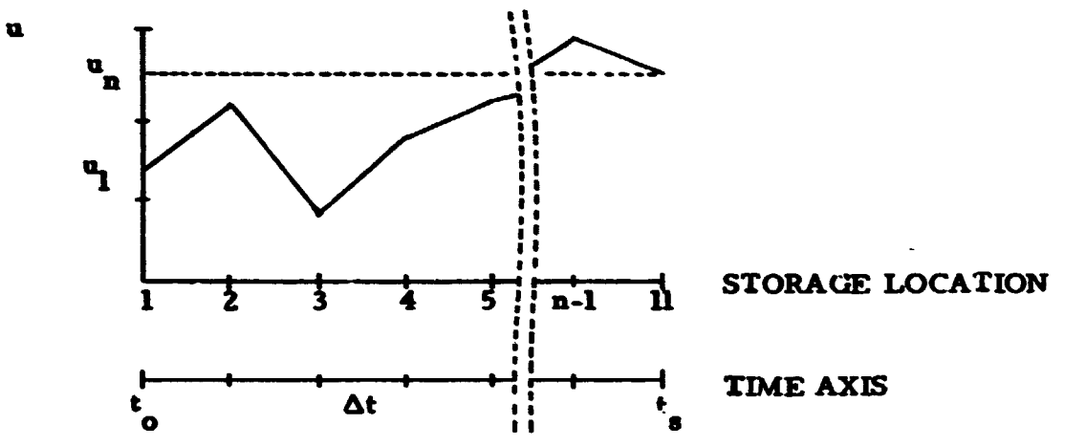


Figure 4.2 Control Storage

u_i (i = 1, n) and n are known values.

The mass of propellent m_f is known and must equal the area under the u(t) curve,

$$m_f = \left[\frac{u_1}{2} + u_2 + \ldots + u_{n-1} + \frac{u_n}{2}\right] \Delta t$$

Thus

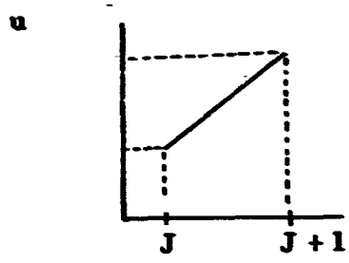
$$\Delta t = \frac{m_f}{\left[\frac{u_1}{2} + u_2 + \ldots u_{n-1} + \frac{u_n}{2}\right]}$$

Since Δt is known, t_s can be computed,

$$t_s = (n-1)\,\Delta t \quad \text{(assume } t_o = 0\text{)}$$

The same procedure is used to calculate t_f.

Now consider the interval between storage locations J and J + 1 which corresponds to the time interval t_J to t_{J+1}.



$$\Delta t = t_{J+1} - t_J$$

Figure 4.3 Single Control Segment

On this interval, $u(t) = 2a\,T + b$ where $T = t - t_J$ and

$$b = u_J \qquad 2a = \frac{u_{J+1} - u_J}{\Delta t}$$

Since $u = -\dot{m}$, the mass can be obtained as a function of T by integration,

$$m(T) = m_J - \int_o^T (2as + b)\,ds$$

$$= A T^2 + B T + C$$

Where,

$A = -a$

$B = -b$

$C = m_J$ (mass at start of interval)

Because of the assumed form for u (t) it is possible to calculate t_s, t_f, Δt, and mass (t) analytically. This avoids the problem of guessing

a t_f and Δt and integrating the whole system of equations while checking for fuel exhaustion. It also avoids the need for extending or contracting the control guess if the mass of propellent is not zero at the guessed t_f.

4.3 Storage Problems With Quasi-Newton Algorithms

In Section 2.2 it was shown that 2 i + 4 time functions must be stored after the i^{th} iterate in order to compute the i + 1 st search direction. Each of these functions is stored as a n- vector of numbers which correspond to the function values at n equally spaced points on $[t_o, t_f]$. Thus (2 i + 4) n floating point numbers must be stored after the i^{th} iterate. The computation per iterate also increases because of the increased number of inner product evaluations. Thus it is a practical necessity to restart the algorithms to a pure gradient step every q^{th} iterate. It has been found that $3 < q < 8$ is a good choice. The value of n must be large enough so that a "good" representation of the functions is obtained. For the shuttle optimization problem the time interval is approximately 500 seconds and n was chosen to be 500. Thus storage must be allocated for (2q + 4) n = (2 x 8 + 4) 500 = 10,000 double precision floating point numbers. Additional storage must be allocated for other variables used in the program and for the object program which is generated from a fortran source deck of 3800 statements.

During the initial testing of the program on the University of Michigan IBM 360/67 virtual memory computer all storage was done in fast memory. However it was found that core storage was exceeded when the program was first run on the JSC's Univac 1108 computer.

To overcome this difficulty the 10,000 double precision floating point numbers needed for the quasi-Newton algorithms were placed on drum storage. This reduced the amount of core storage required allowing the program to fit on the 1108. Upon running the modified program on the IBM computer a considerable savings was realized in reduced virtual memory charges. It was also found that no significant increase in the amount of CPU time was incurred. There are two reasons for this,

i) a very small percent of CPU time is spent calculating the search direction. Most of the CPU time is spent integrating the equations of motion. On each iterate a forward integration and a backward integration are required to determine the gradient and a number of cost evaluations also requiring forward integrations are performed by the 1-D search.

ii) the updating equation for $H_i y_i$ and the equation for d_i are summations which require inner products of the stored functions in the same sequence as they were generated and stored. Assume $H_{i-1}\ y_{i-1}$ and d_i are to be calculated. $H_o y_o$ through $H_{i-2}\ y_{i-2}$ are stored in a file,

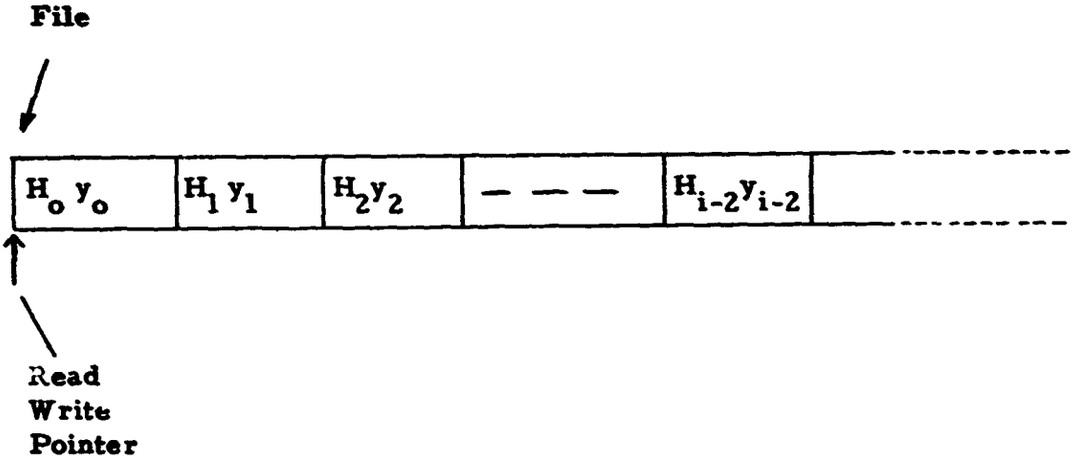


Figure 4.4 File Storage Diagram.

At the end of the last iteration the file has been rewound. The updating equations for $H_{i-1}\, y_{i-1}$ will read $H_o y_o$, $H_1 y_1$, . . . , $H_{i-2}\, y_{i-2}$ in order, calculate $H_{i-1}\, y_{i-1}$, then write $H_{i-1}\, y_{i-1}$ onto the file and rewind. Concurrently the equation for d_i has been using the Hy functions. The files in which Hy and s are stored need only be rewound once on a given iteration and no forward or back spacing is required. Even if tape were to be used as the storage medium, instead of fast core storage, the increase in computer time would be small. When drum storage is used the increase in computer time is insignificant. Thus there is no need to restart to a gradient step because of limited storage.

As mentioned previously the computation time per iterate increases due to the increasing number of inner product evaluations which must be made. The inner product is a quadrature.

$$\langle u, v \rangle = \int_{t_o}^{t_f} u^T v \, dt$$

where u and v are stored pointwise. If it is assumed that the stored functions are linear between storage locations the evaluation of the inner product reduces to a summation. Consider the interval t_1 to t_2,

Let $T = t - t_1$ and $\Delta t = t_2 - t_1$ then on $[t_1, t_2]$

$$u(T) = a\,T + b \quad \text{and} \quad v(T) = \alpha\,T + \beta$$

where

$$a = \frac{u_2 - u_1}{\Delta t} \qquad \alpha = \frac{v_2 - v_1}{\Delta t}$$

$$b = u_1 \qquad \beta = v_1$$

The inner product of the functions between t_1 and t_2 is;

$$\langle u, v \rangle_{t_1, t_2} = \int_0^{\Delta t} (aT + b)(\alpha T + \beta)\, dt$$

$$= \int_0^{\Delta t} [a\alpha\, T^2 + (\alpha\beta + ab)\, T + b\beta]\, dt$$

$$= \frac{a\alpha}{3} \Delta t^3 + \frac{a\beta + \alpha b}{2} \Delta t^2 + b\beta\, \Delta t$$

and the total inner product is

$$\langle u, v \rangle_{t_o, t_f} = \sum_{i=0}^{n-1} \langle u, v \rangle_{t_i, t_{i+1}}$$

It was found that this method of evaluating inner products is considerably faster then higher order quadrature formulas and that convergence rates of the algorithms do not suffer.

4.4 One Dimensional Search (1-D Search)

On each iteration a search direction d_i is generated, and then a new control is calculated,

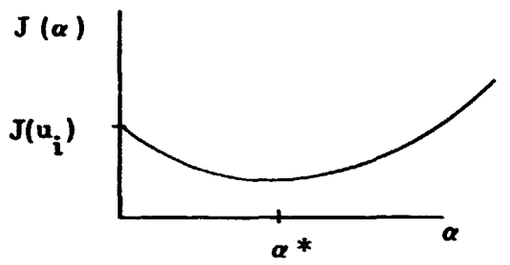
$$u_{i+1} = u_i + \alpha_i d_i$$

The goal of the 1-D search is to find a scalar parameter α_i which yields the greatest cost decrease. At such a value it is necessary that

$$\frac{\partial}{\partial \alpha} J(u_i + \alpha d_i) = 0,$$

which is an important element in the convergence proofs of the quasi-Newton algorithms for the linear quadratic problem.

A large fraction of CPU time is spent within the 1-D search and its accuracy and efficiency greatly influence the convergence rate. For small α, $\frac{\partial J}{\partial \alpha} < 0$ thus we can expect a functional relationship with the following form,



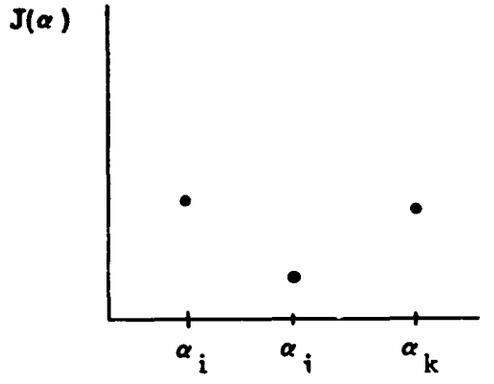
for $\alpha = 0$ $J = J(u_i)$

$J(u_{i+1})$ will occur at

$\alpha = \alpha^*$

Figure 4.5 Cost vs α

As α increases J will decrease until the higher order terms in the expansion dominate and J begins to increase again. The 1-D search attempts to find α^*. The performance index is evaluated at $\alpha = 0$ and $\alpha = \alpha_1$ where α_1 is a guess for α^*. α is then increased or decreased until the minimum is bracketed, that is three points are found such that,



$\alpha_i < \alpha_j < \alpha_k$

$J(\alpha_i) > J(\alpha_j) < J(\alpha_k)$

The function $J(\alpha)$ is approximated by a quadratic curve,

$$J(\alpha) = a\,\alpha^2 + b\,\alpha + c$$

where a, b, and c are determined by fitting the quadratic curve through the three data points. The minimum of the quadratic curve is given by,

$$\frac{\partial J}{\partial \alpha} = \partial a\alpha + b = 0$$

$$\alpha = \frac{-b}{2a}$$

The performance index is evaluated at α' and if,

$$J(\alpha') < J(\alpha_j)$$

the control generated by α is chosen as the local minimizing element. If

$$J(\alpha') > J(\alpha_j)$$

a new quadratic curve fit is performed with α', α_j and,

$$\alpha_i \text{ if } \alpha' > \alpha_j$$

or

$$\alpha_k \text{ if } \alpha' < \alpha_j$$

This process is repeated until a minimum is found.

4.5 CRT Graphic Display

The space shuttle ascent trajectory optimization problem developed in Chapter 3 is a Bolza problem with the addition of state variable inequality constraints,

$$\text{acceleration} \leq ACC_{max}$$

Dynamic Pressure $\leq Q_{max}$

and terminal boundary conditions,

50 x 100 nm orbit

Inclined 28.5^{o} to equator

Entered at perigee

where

ACC_{max} = 3.05 g's boost phase

ACC_{max} = 3.0 g's orbiter phase

Q_{max} = 650 psf.

This optimization problem is replaced by an unconstrained optimization problem where the terminal boundary conditions and state variable inequality constraints are enforced by the method of penalty functions. The new unconstrained optimization problem has seven independent penalty coefficients, and the performance index is

$$J = -W_o + P_1 [\text{ ERROR IN FINAL RADIUS}]^2$$

$$+ P_2 [\text{ ERROR IN FINAL RADIAL VELOCITY }]^2$$

$$+ P_3 [\text{ ERROR IN FINAL TANGENTIAL VELOCITY}]^2$$

$$+ P_4 \int_{t_o}^{t_s} (q(t) - q_{max})^2 U(q(t)-q_{max})\, dt$$

$$+ P_5 \int_{t_o}^{t_s} (acc(t)-acc_{max})^2 U(acc(t) - acc_{max})\, dt$$

$$+ P_6 \int_{t_s}^{t_f} (acc(t) -acc_{max})^2 U(acc(t) -acc_{max})\, dt$$

$$+ P_7 [\text{ ERROR IN FINAL INCLINATION}]$$

$$U(\eta) = \begin{cases} 0 & \eta < 0 \\ 1 & \eta \geq 0 \end{cases}$$

Here P_i (i = 1, 2, 3, 7) are penalty coefficients associated with the terminal boundary conditions and P_i (i = 4, 5, 6) are penalty coefficients associated with the state variable inequality constraints. For a given set of penalty coefficients a particular unconstrained optimization problem is defined. The solution to the original constrained optimization problem is approximated by a sequence of solutions to the unconstrained problem generated by letting P_i (i = 1, ... , 7) $\rightarrow \infty$. As P_i (i = 1, 2, 3, 7) are increased the solutions generated will more closely satisfy the requirements of a 50 x 100 nm orbit inclined 28.5° to the equator entered at perigee. Likewise as P_i (i = 4, 5. 6) are increased the state variable inequality constraints on dynamic pressure and acceleration are more strictly enforced. The ultimate goal is to find the control history which yields the maximum liftoff weight and satisfies all seven of the constraints. As expected, in practice as one penalty coefficient is increased the error associated with it will decrease while the errors associated with the other coefficients will increase. Thus by improving the trajectory in one respect it is possible to lose something somewhere else. Sensitivity to changesin the different penalty coefficients also varies. As the penalty coefficients become larger the overall problem will become increasingly sensitive to changes in the control and numerical instability will eventually result. The way in which the penalty coefficients are

increased will stronglyinfluence the overall convergence rate of the algorithms. The main drawback to the method of penalty functions is that the penalty coefficients must be increased in a problem dependent way. Even for simple example problems which require little computer time for a trajectory integration and which have only one or two penalty coefficients, the choice of these coefficients and the way in which they are increased is critical for rapid convergence. Because of the complexity and relatively long computer time required for a trajectory integration of the shuttle ascent optimization problem a better method than trial and error is required for choosing penalty coefficient values.

By using time sharing computers and CRT display terminals the problem of choosing penalty coefficient values can be very efficiently solved by human operator interaction with the executing program. At the end of each iteration execution is terminated and control transfered to a CRT display terminal. Because of time sharing this interruption of the executing program is very inexpensive. At the request of the human operator important information is then graphically displayed on the CRT. The information is evaluated and a decision on changes of the penalty coefficients is reached. This information is communicated to the computer and execution proceeds. By placing a human operator in the program iteration cycle convergence times are reduced, the computer is used more efficiently, and the operator quickly builds an intuitive feel for the physical problem being solved.

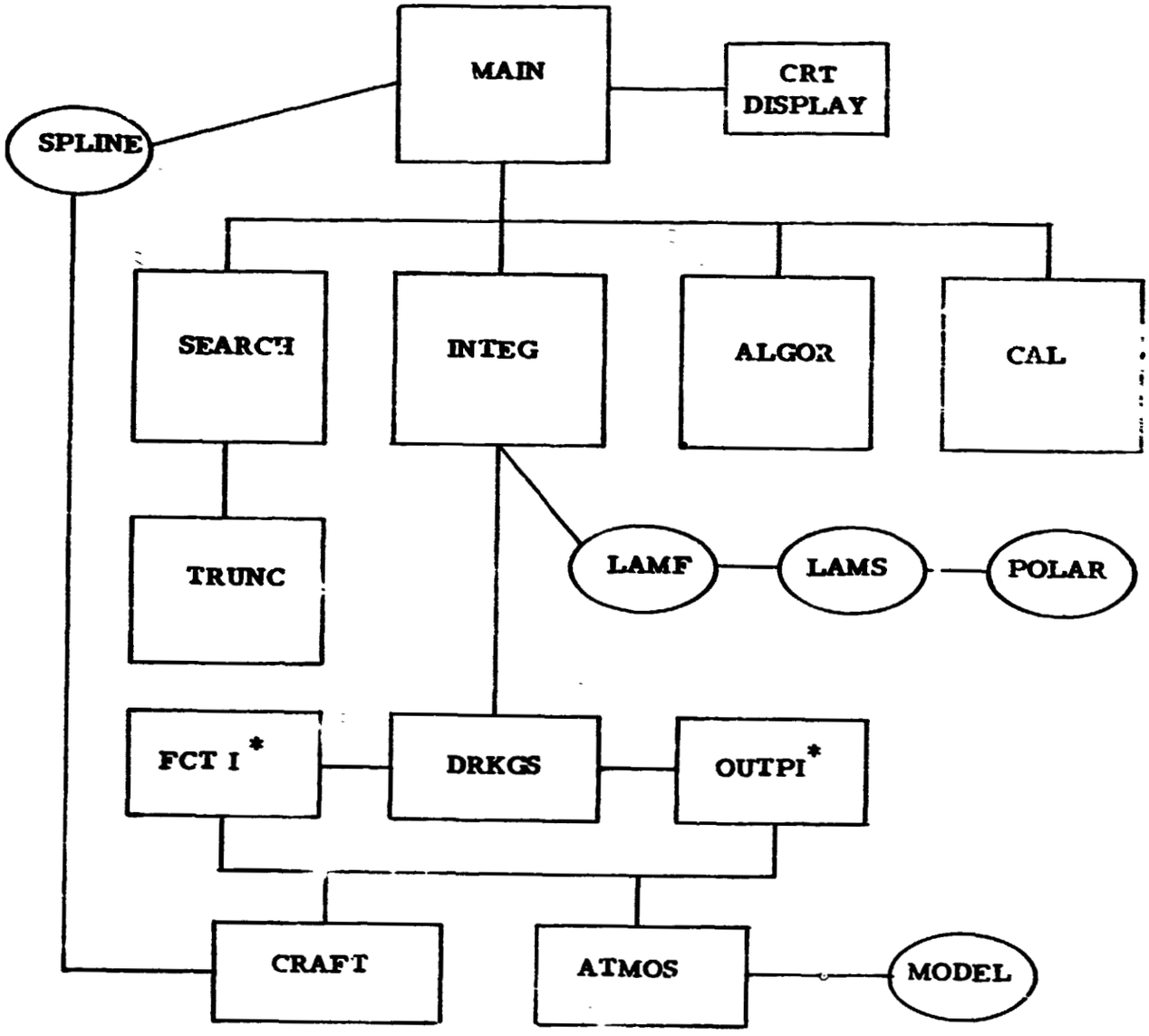
For the shuttle ascent optimization problem it is helpful to graphically display dynamic pressure, acceleration, and u as functions of time along with terminal miss values. The best convergence rate was achieved by first increasing P_i (i = 1, 2, 3, 7) yielding a trajectory which comes "close" to the desired terminal boundary conditions. Then P_i (i = 4, 5, 6) are increased to enforce the state variable inequality constraints while simultaneously increasing P_i (i = 1, 2, 3, 7) so that all intermediate trajectories remain "close" to the terminal boundary conditions.

The ability to interact with the executing program can be useful in other ways. The interrelationship of adjoint, state variable, search direction, and gradient time histories can be conveniently analyzed using the CRT display. In conclusion the ability to communicate with the executing program is a valuable tool for analysis of optimization programs.

4.6 Subroutine Description

The computer program consists of nineteen subroutines controlled by the main control program. Figure 4.6 presents a subroutine map which illustrates the relationship between subroutines. In this section the function of each subroutine will be explained.

MAIN - reads input parameters, calls SPLINE to obtain curve fit of aerodynamic coefficients, controls forward, backward, and cost integrations, calls CAL to determine constant gradients, calls SEARCH which contains the 1-D Search, also contains logic for interaction with CRT display terminal.



* there are three FCT's and three OUTP's I = 1, 2, 3

Figure 4.6 Subroutine Map

SPLINE - a subroutine which interpolates by piecewise cubic splines aerodynamic coefficients such as C_D which is given in tabular form as a function of Mach number.

INTEG - contains the logic to determine the mass distribution, staging time, final time, and calls DRKGS for forward, backward, and cost integrations.

ALGOR - contains the various algorithms which require a gradient g(t) as the input and produce d(t) the search direction as the output.

CAL - performs the quadrature which calculates the constant gradients.

SEARCH - contains the 1-D search, i.e., determines α which minimizes the performance index, see Section 4.4.

TRUNC - performs the truncation of new controls generated by varying $\bar{\alpha}$ in SEARCH.

LAMF - calculates the value of the adjoint variables at t_f.

LAMS - calculates the jump in adjoint variables at t_s.

POLAR - calculates the jump in state variable at t_s and calculates the inclination of the orbit.

DRKGS - a double precision fourth order variable step size Runge-Kutta integration subroutine contained in the IBM SSP package.

FCT - computes the right hand side of the system of equations to be integrated.

OUTP - an output subroutine used by DRKGS

CRAFT - calls spline to determine C_D and $\frac{\partial C_D}{\partial m}$.

ATMOS - calls MODEL to determine density (ρ), pressure (p), and speed of sound (a); also calculates

$$\frac{\partial \rho}{\partial h} \qquad \frac{\partial p}{\partial h} \qquad \frac{\partial a}{\partial h}$$

where h is the altitude.

MODEL - contains the atmospheric model; see Appendix B.

4.7 Numerical Results

The final control history and associated trajectory which will be presented in this section are the result of computer runs made at both JSC and at the University of Michigan. The initial control guess was:

C_1 = payload mass = 80,000 lbm.

$C_2 = \dot{\gamma} = 0.689241^{\circ}$/sec.

$C_3 = a = -0.431410 \times 10^{-3}$

$C_4 = b = 0.365070$

$C_5 = \psi = -19.0049^{\circ}$

$u(t) = 98\%$

This resulted in a trajectory with the following terminal miss values,

$\Delta R = -180,000$ ft.

$\Delta U = -200$ fps.

$\Delta V = 602.1$ fps.

Inclination = 26.23°

The staging time was 118.73 sec. with a final time of 503.3 sec. The state variable inequality constraints were also violated. Q reached a peak value of 819 psf at 65.2 sec. while the maximum acceleration during first stage was 3.8 g's and during second stage was 3.9 g's.

With the above initial control the program was run on JSC's computer for 12.75 minutes. The resulting final control became the initial control for subsequent runs made on the University of Michigan computer. An additional 45.7 minutes of computer time were expended on the University of Michigan computer for a total run time of 58.4 minutes. The penalty coefficients were;

	INITIAL	FINAL
P_1	10^6	10^{16}
P_2	10^9	10^{19}
P_3	10^9	10^{19}
P_4	10^6	10^{12}
P_5	10^0	10^3
P_6	10^{10}	10^{12}
P_7	10^9	10^{15}

The final control is;

$C_1 = 101,300$ lbm

$C_2 = 0.631857\ ^{o}$/sec.

$$C_3 = -.478541 \times 10^{-3}$$

$$C_4 = 0.366590$$

$$C_5 = -8.5^{\circ}$$

u (t) - Figure 4.7

On the converged trajectory, the staging time is 121.1 sec. and the final time is 504.0 sec. Figure 4.8 shows the angle $\dot{\gamma}$ above the local horizontal at which the thrust is orientated. Figures 4.9 and 4.10 indicate that the state variable inequality constraints are being enforced. Figure 4.11 shows the time history of altitude vs time. The terminal miss values are,

$$\Delta R = -4,700 \text{ ft.}$$

$$\Delta U = 1.2 \text{ fps}$$

$$\Delta V = 5.2 \text{ fps}$$

$$\text{Inclination} = 28.8^{\circ}$$

These values could be improved by decreasing the integration stepsize.

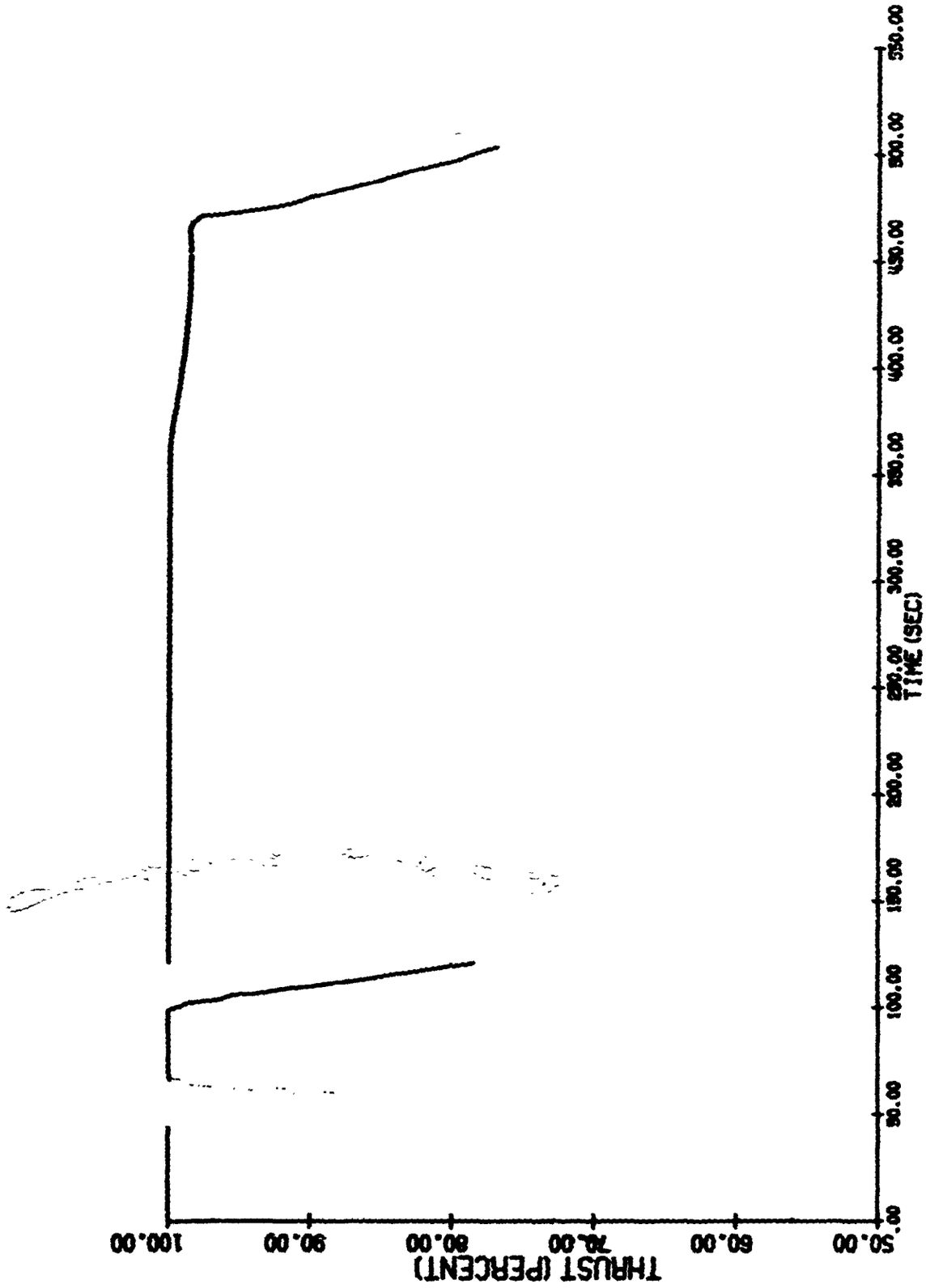

THRUST (PERCENT)
50.00
60.00
70.00
80.00
90.00
100.00
TIME (SEC)
.00
50.00
100.00
150.00
200.00
250.00
300.00
350.00
400.00
450.00
500.00
550.00

Figure 4.7

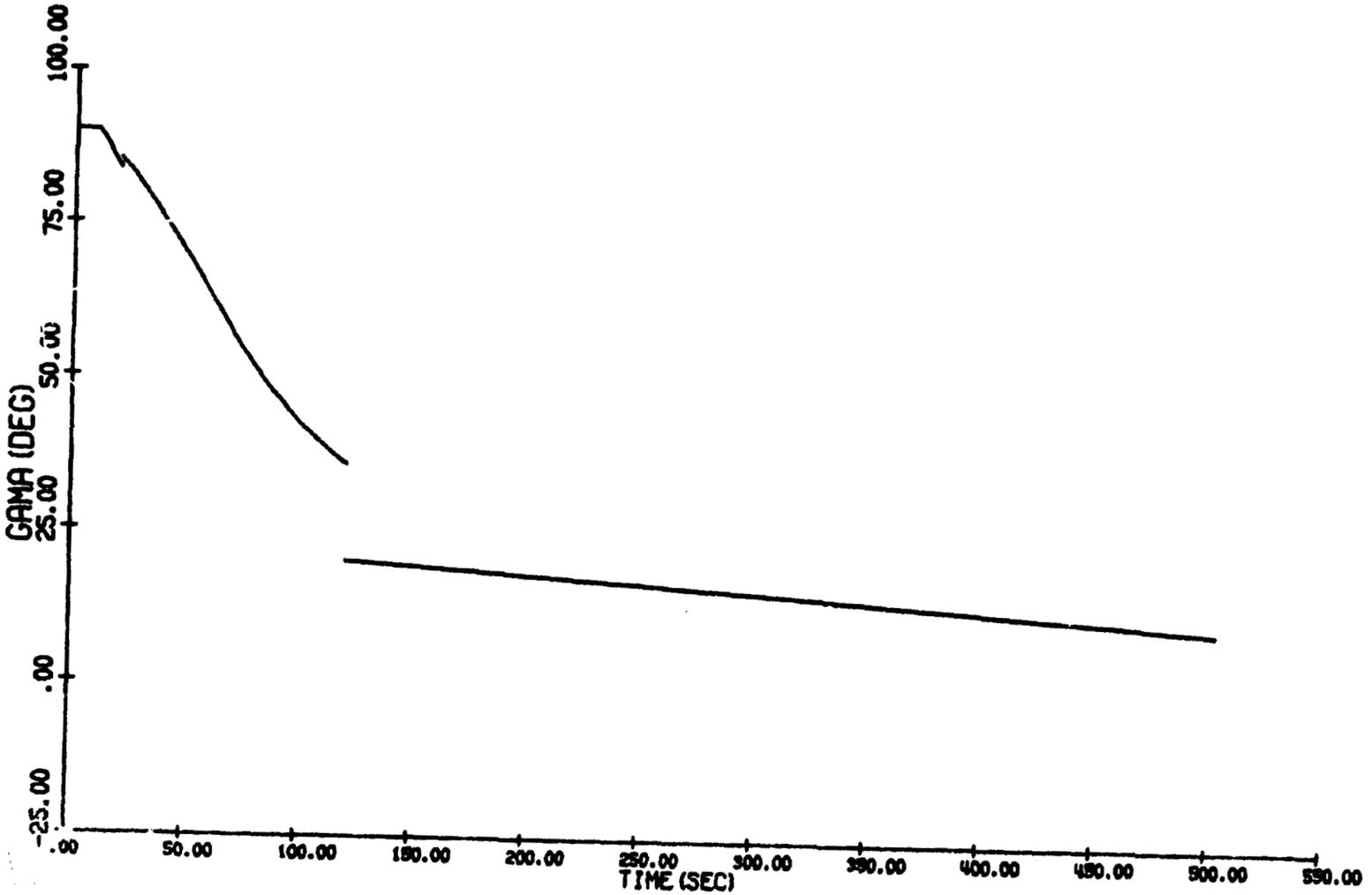

GAMA (DEG)
100.00
75.00
50.00
25.00
.00
-25.00
.00
50.00
100.00
150.00
200.00
250.00
300.00
350.00
400.00
450.00
500.00
550.00
TIME (SEC)

Figure 4.8

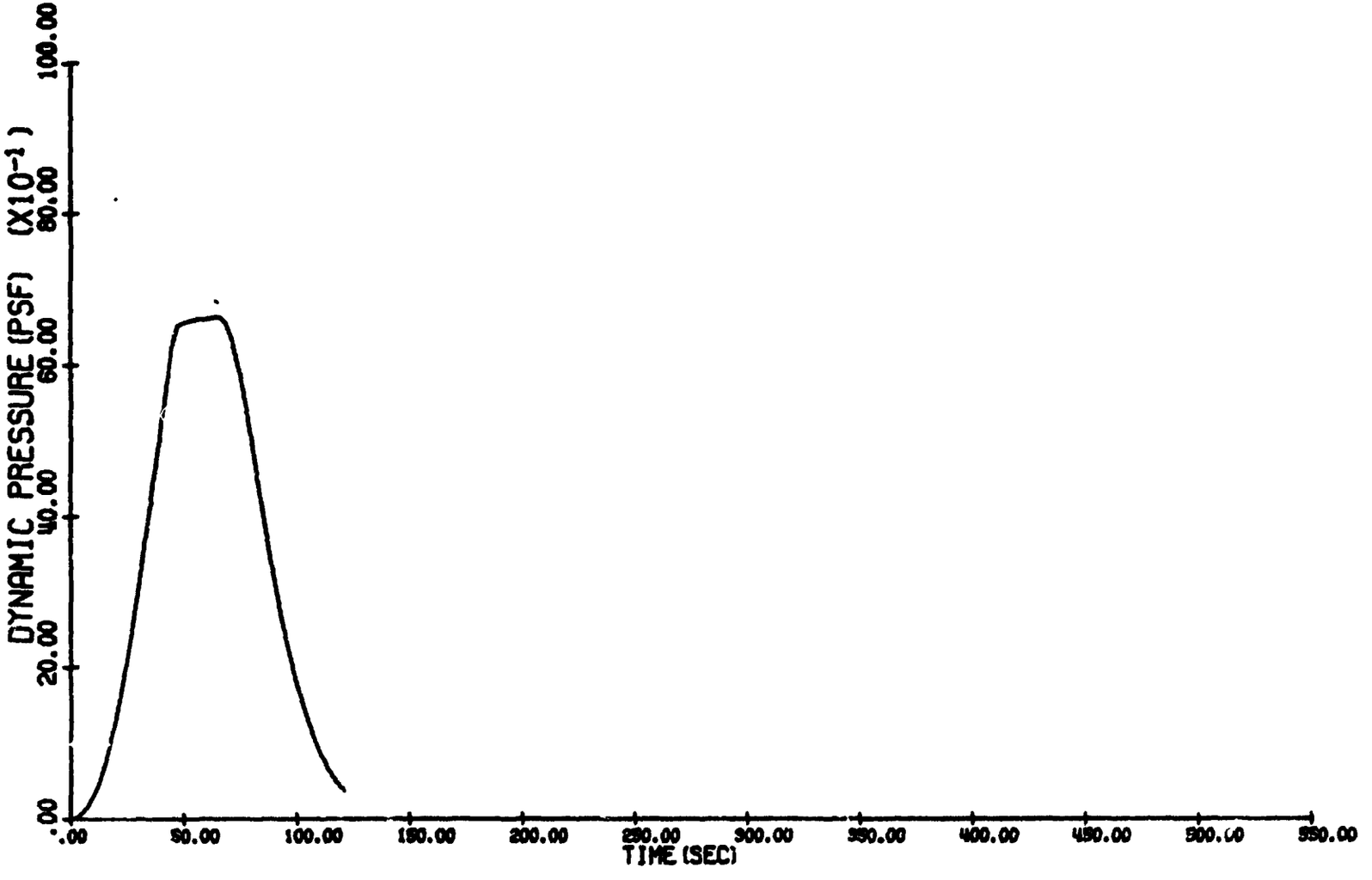

DYNAMIC PRESSURE (PSF) (X10⁻¹)
100.00
80.00
60.00
40.00
20.00
.00
.00
50.00
100.00
150.00
200.00
250.00
300.00
350.00
400.00
450.00
500.00
550.00
TIME (SEC)

Figure 4.9

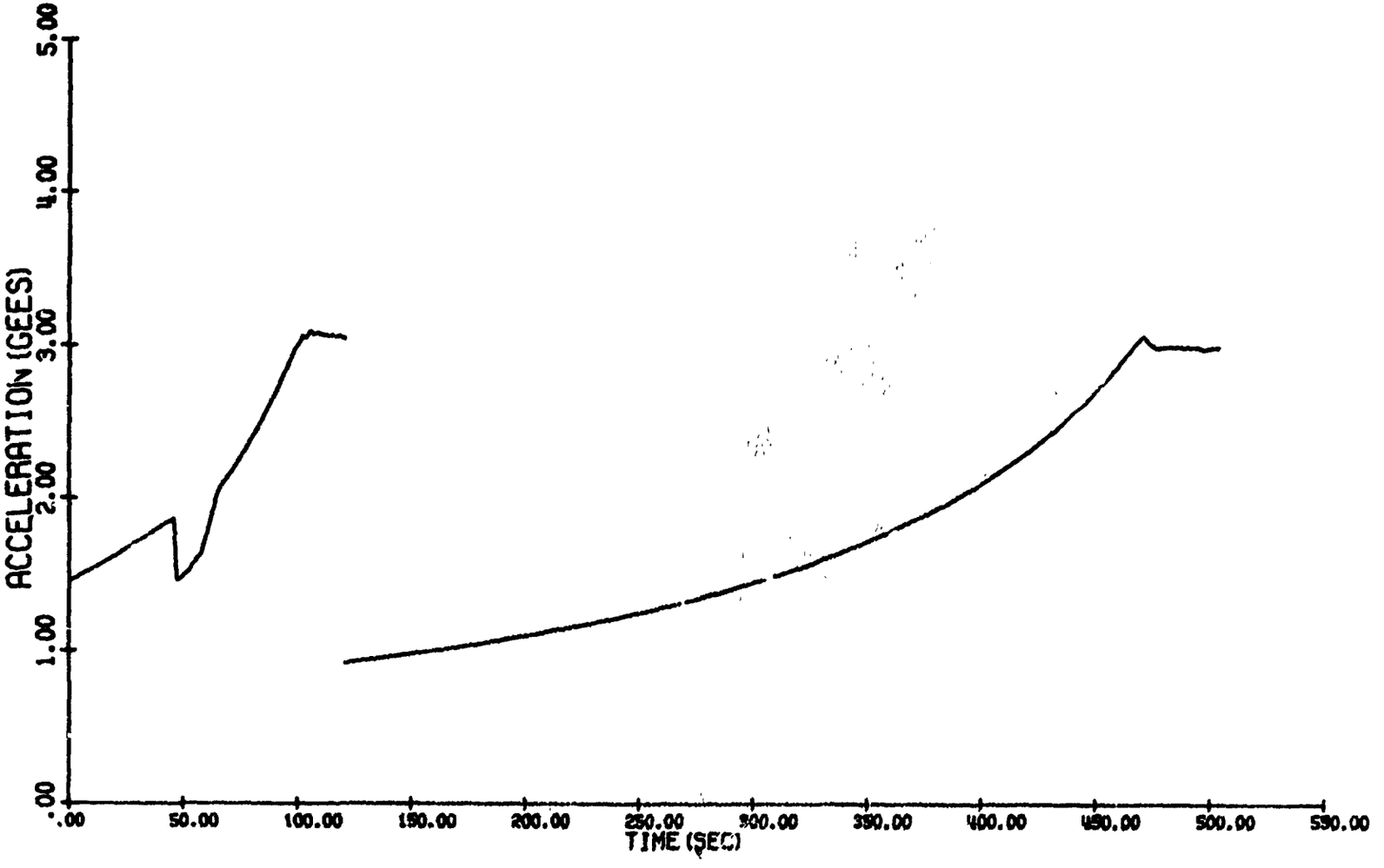

ACCELERATION (GEES)
5.00
4.00
3.00
2.00
1.00
.00
.00
50.00
100.00
150.00
200.00
250.00
300.00
350.00
400.00
450.00
500.00
550.00
TIME (SEC)

Figure 4.10

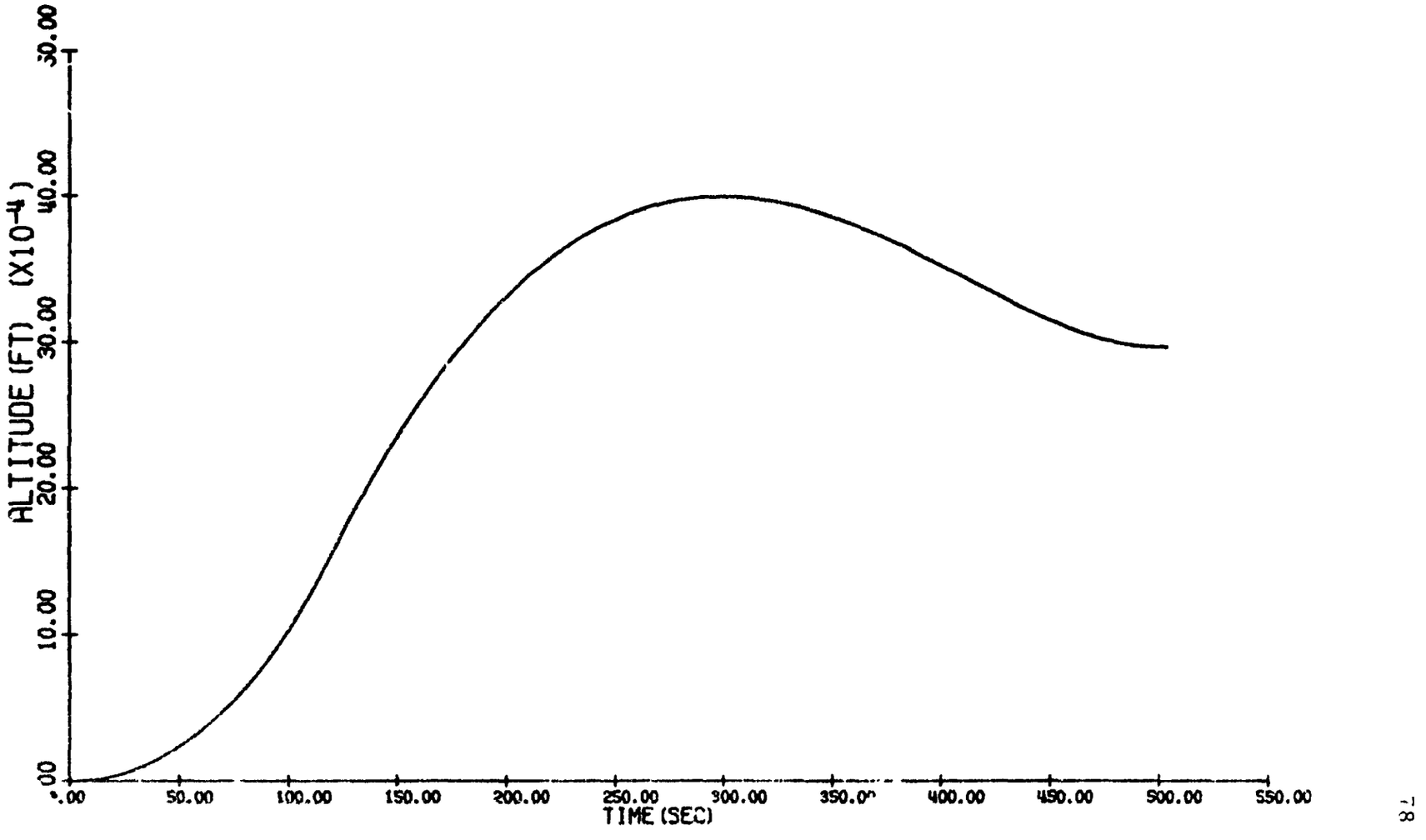

ALTITUDE (FT) (X10⁻⁴)
.00
10.00
20.00
30.00
40.00
50.00
.00
50.00
100.00
150.00
200.00
250.00
300.00
350.00
400.00
450.00
500.00
550.00
TIME (SEC)

Figure 4.11

CHAPTER 5

TERMINAL CONSTRAINTS AND VARIABLE FINAL TIME CONSIDERATIONS

When a function space gradient-type technique is employed, one must continually confront the problems associated with terminal constraints and variable final time. Usually for flexibility and ease of programming a penalty function type of approach is preferred to a projected gradient approach. In this chapter we shall compare the standard penalty function approach to the Hestenes multiplier method[18] on a single problem. The results for this particular problem indicate that the multiplier method does not improve the performance enough to merit the additional programming and complexity.

The major result of this chapter is concerned with a simple procedure which apparently improves considerably the rate of convergence of gradient-type methods when penalty functions are employed on variable final time problems. The result is simply that the initial estimate of t_f, say $t_f^{(0)}$, should be less than the optimal t_f value, say t_f^*. In fact, it appears that a physically unreasonable choice of $t_f^{(0)}$ which guarantees $t_f^{(0)} < t_f^*$ is superior to a physically reasonable initial trajectory with $t_f^{(0)} > t_f^*$, with respect to rate of convergence. Although this property has yet to be proved mathematically, it appears to be heuristically justifiable, and all of our numerical simulations confirm the trend. Finally, it will be shown that a recently proposed method for treating variable final time problems by Tripathi and Narendra[16] is

essentially the well-known method developed by Long [17] in 1965.

5.1 Variable Final Time Problems

Consider the performance index for a time-optimal control problem with fixed-endpoint type terminal conditions

$$J = C t_f + \sum_{i=1}^{n} P_i (x_i - x_{if})^2 + \sum_{i=1}^{n} C_i (x_i - x_{if}) \qquad (5.1)$$

where

P_i are penalty coefficients for terminal constraints; $P_i = 0$ if $x_i(t_f)$ is not specified.

C_i are multiplier constants for the multiplier method [18]; $C_i = 0$ if the penalty function method is used.

In the time-optimal control problem the algorithms require an initial estimate of t_f, say $t_f^{(0)}$. On future iterates, a procedure for updating t_f must be specified, and this will affect the rate of convergence. The following have been proposed in the literature.

1. If t_f is reduced, the pertinent functions are well-defined for all t. If t_f increases, the control is set equal to the value at $t_f^{(i)}$, which is t_f of the previous iterate, in the extended interval $[\, t_f^{(i)}, t_f^{(i+1)} \,]$.

   The program used in this chapter is based on this technique and it worked satisfactorily, at least for relatively simple problems.

2. If t_f increases, the various functions are suitably extrapolated over the new range.

3. The functions maintain the same form, only the time scale is

modified to take care of the changes in the interval $[t_o, t_f]$. Tripathi and Narendra [16] found this method to be satisfactory in practice.

4. Convert the problem to a fixed final time problem with the transformation below. (An additional parameter appears due to this transformation, and the method was proposed by R. Long[17]).

   A. Define $t = a\,s$

   where $\dot{x} = f(x, u, t)$: the equations of motion

   a = constant to be determined

   s = a new independent variable. $0 \leq s \leq 1$

   B. Let: $s = 0$ be the initial point.

   $s = 1$ be the final point $\Rightarrow t_f = a$.

   C. New equations of motion

$$( )' \equiv \frac{d( )}{ds}$$

$$x' = af \qquad (5.2)$$

$$a' = 0$$

Note: Since "a" is an unknown constant parameter, an initial estimate of "a" is needed and an updating scheme must be defined.

It will be shown that the method proposed by Tripathi and Narendra is essentially the same as the method by Long. The justification is as follows:

If $t_f^{(i+1)} = t_f^{(i)}$, then following Ref. 16, a function, say $k^i(t)$, should be updated by

$$k^{(i+1)}(t) = k^{(i)}(t/\alpha). \tag{5.3}$$

This method causes the function k (t) to be compressed or expanded as $t_f^{(i+1)}$ decreases or increases, respectively.

The method can be defined alternatively by introducing the independent variable $T = \alpha\, t$. Then, the function for the next iteration is

$$k^{(i+1)}(t) = k^{(i)}(t/\alpha). \quad (t_o = 0) \tag{5.4}$$

For example, if

$$k^{i}(t) = t^2 + t \qquad \forall\, t \in \left[0\ ,\ t_f\right],$$

then, the function for the (i + 1) -iterate will be

$$k^{i+1}(t) = k^{i}(t/\alpha) = \frac{t^2}{\alpha^2} + \frac{t}{\alpha} \qquad \forall\, t \in \left[\ 0\ ,\ \alpha\, t_f\right].$$

Thus, Tripathi and Narendra's method can be represented by the transformation $T = \alpha t$, if the relation $t_f^{\,i+1} = \alpha\, t_f^{\,i}$ holds.

In the application of Long's method, the value of the constant "a" has to be guessed initially to start the scheme. Assume that at the (i + 1) iteration.

$$a^{i+1} = \alpha\, a^{i} \tag{5.5}$$

This implies

$$t^{i+1} = a^{i+1} s \tag{5.6}$$

$$t^{i} = a^{i} s, \tag{5.7}$$

Substitution of (5. ato (5.6), and use of (5.7) implies

$$t^{i+1} = a^{i+1} s = \alpha\, a^{i} s = \alpha\, a^{i} \frac{t^{i}}{a^{i}} = \alpha\, t^{i},$$

or,

$$t^{i+1} = \alpha\, t^{i} \tag{5.8}$$

which is the transformation equation used by Tripathi and Narendra. Thus, method (3) and method (4) have similar basic characteristics.

In the next section method (1) above is employed, and numerical examples are presented to show that $t_f^{(0)} < t_f^*$ gives a more rapid rate of convergence than $t_f^{(0)} > t_f^*$.

5.2 Numerical Examples for Minimum Final Time Problems

Example 1. Zermelo's problem

$$\dot{x} = v \cos \theta$$

$$\dot{y} = v \sin \theta$$

$$\dot{\theta} = u \qquad (5.9)$$

where v = constant,

$$x(0) = x_o = 0\,,\ y(0) = y_o = 0\,,\ \theta(o) = \theta_o = 0$$

$$|u| \leq k,\ k \text{ maximum turning rate}$$

Determine the minimum time to reach the specified final states:

A. $x(t_f)$ = free, $y(t_f)$ = free, $\theta(t_f) = \theta_f$ (5.10)

E. $x(t_f) = x_f$, $y(t_f) = y_f$, $\theta(t_f)$ = free (5.11)

For these simple problems, analytical solutions can be obtained without difficulty.

Case A: It is easily shown that given v = 1, k = .50 and $\theta_f = 2\pi$, the control will be either u = + k or u = -k for the vehicle to reach the specified heading in minimum time.

The cost functional is

$$J = C\,t_f^2 + P_1 (x - x_f)^2 + P_2 (y-y_f)^2 + P_3 (\theta-\theta_f)^2$$

$$+ C_1 (x-x_f) + C_2 (y-y_f) + C_3 (\theta-\theta_f) \qquad (5.12)$$

where $P_1 = P_2 = 0$ and $C_1 = C_2 = C_3 = 0$ for the penalty function method. The optimal trajectory is a circle centered at (0,2) with radius two for $\theta_f = 2\pi = .5\, t_f^*$, $t_f^* = 12.56$ seconds, and

Initial Conditions	Terminal Conditions
$x_o = 0$	x_f = free
$y_o = 0$	y_f = free
$\theta_o = 0$	$\theta_f = 2\pi = 6.28$

The purpose of this problem is to show how the conjugate gradient method is affected by the initial final time estimate, $t_f^{(0)}$. Let $C = 1$, $C_3 = 0$, $P_3 = 100$, and the integration stepsize = $\Delta t = 0.2$ seconds.

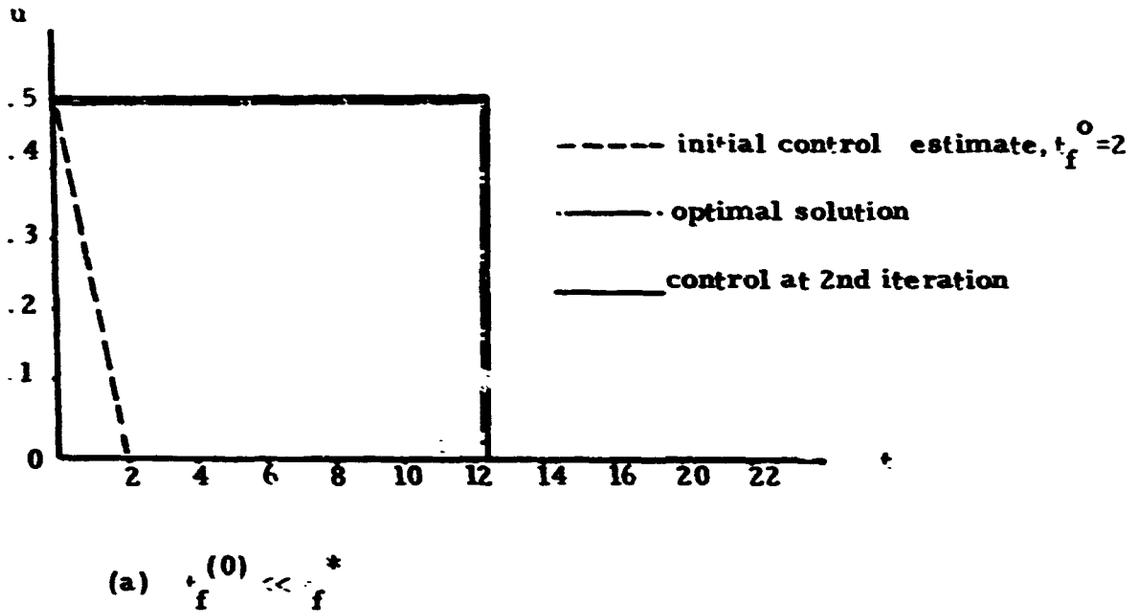
(i) Consider $t_f^{(0)} = 2$ seconds $\ll t_f^*$. The algorithm increases the final time to 12.65 seconds with $u^{(2)} = +.5$ in two iterations, (see Figure 5.1.a).

(ii) Consider $t_f^{(0)} = 19$ seconds $\gg t_f^*$. After six iterations, $t_f^{(6)} = 12.084$ seconds with $u = +.5$. (See Fig. 5.1.b.) After two iterations, $t_f^{(2)} = 18.96$. Thus, both cases converge rapidly, with the $t_f^{(0)} = 2 < t_f^*$ case having the fastest rate of convergence.

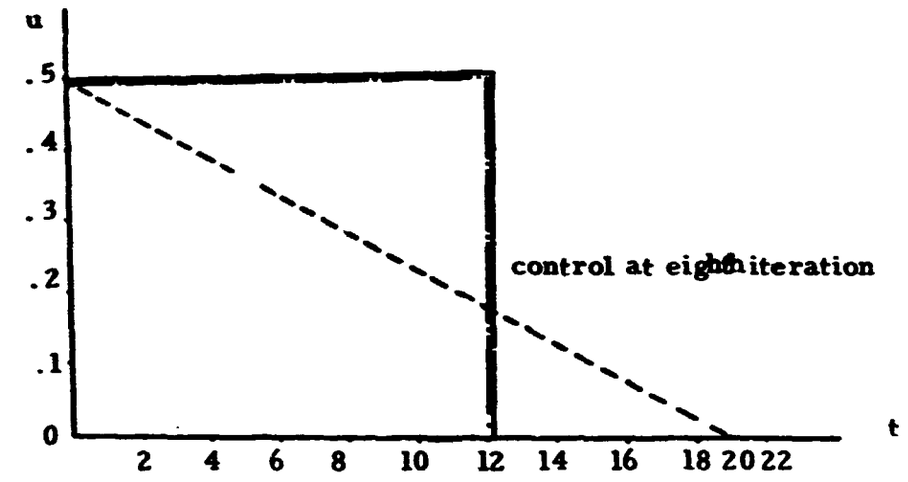
Case B: The exact solution for this case is as follows: To reach the specified position in minimum time, the vehicle will first turn at the maximum rate, and then switch to the singular arc $u = 0$ for straight line flight to the desired position, i.e.,

$$u = +k \qquad \forall\ t \in [t_o, t_1]$$

$$u = 0 \qquad \forall\ t \in [t_1, t_f^*]. \qquad (5.13)$$



(a) $t_f^{(0)} << t_f^*$



(b) $t_f^{(0)} >> t_f^*$

Figure 5.1 Control Profiles for Example 1. Case A.

The corresponding cost functional is (5.12) but with $C = 1$.

$P_1 = P_2 = 100$, $C_1 = C_2 = P_3 = C_3 = 0$ (penalty function method)

Initial Conditions	Terminal Conditions
$x_o = 0$	$x_f = 4$ miles
$y_o = 0$	$y_f = 3$ miles
$\theta_o = 0$	θ_f = free

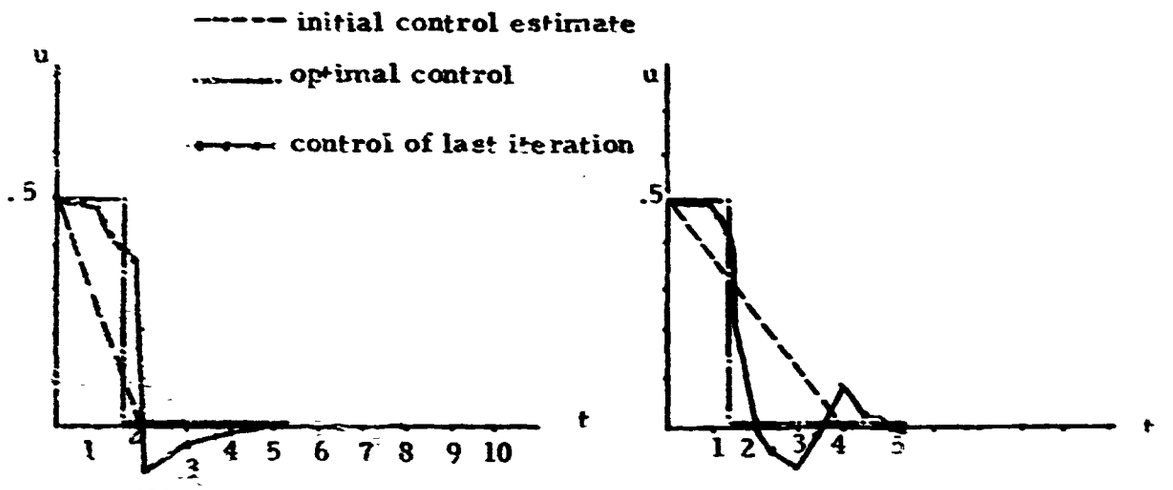
The optimal final time is $t_f^* = 5.058$ seconds. Four cases are considered in this example.

(i) $t_f^{(0)} = 2$ seconds $\ll t_f^*$. The final times goes to $t_f^{(1)} = 4.95$ seconds on the first iteration, and to $t_f^{(10)} = 5.042$ seconds in ten iterations. The position error after ten iterations is within one percent (see Figs. 5.2.a and 5.3).

(ii) $t_f^{(0)} = 4$ seconds. This guess is close to the true minimum. The program performs smoothly and the terminal position error is less than one percent (see Fig. 5.2.c).

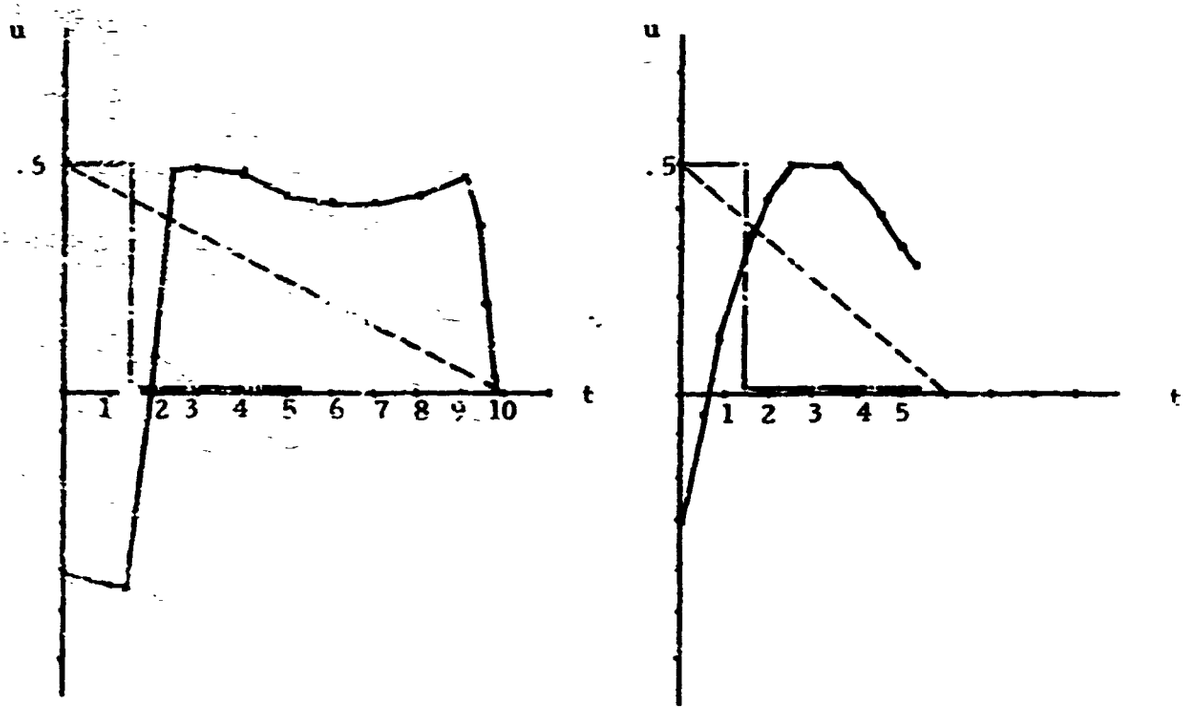
(iii) $t_f^{(0)} = 6$ seconds (slightly larger than the true minimum). Little improvement in final time, $t_f^{(12)} = 5.78$ seconds, after twelve iterations. The position error is about 2.5 percent, and the program terminated due to insignificant cost change. Another interesting aspect of this case is that the control profile converges to a profile far from the optimum. This implies that an initial guess with $t_f^{(0)} > t_f^*$ may have the tendency to converge (apparently) to nonoptimal solutions (see Figs. 5.2.d and 5.3).

(iv) $t_f^{(0)} = 10$ seconds $\gg t_f^*$. After eight iterations, the position error is less than .2 percent. However $t_f^{(8)} = 9.87$ and again



(a) $t_f^{(0)} = 2$ sec, $t_f^* = 5.10$ sec., $t_f^{(2)} = 5.048$ sec.

(c) $t_f^{(0)} = 4$ sec., $t_f^* = 5.10$ sec.
$t_f^{(20)} = 5.062$ sec.



(b) $t_f^{(0)} = 10$ sec., $t_f^* = 5.10$ sec.
$t_f^{(8)} = 9.869$ sec.

(d) $t_f^{(0)} = 6$ sec., $t_f^* = 5.10$ sec.,
$t_f^{(9)} = 5.20$ sec.

Figure 5.2 Control Profiles for Example 1. Case B

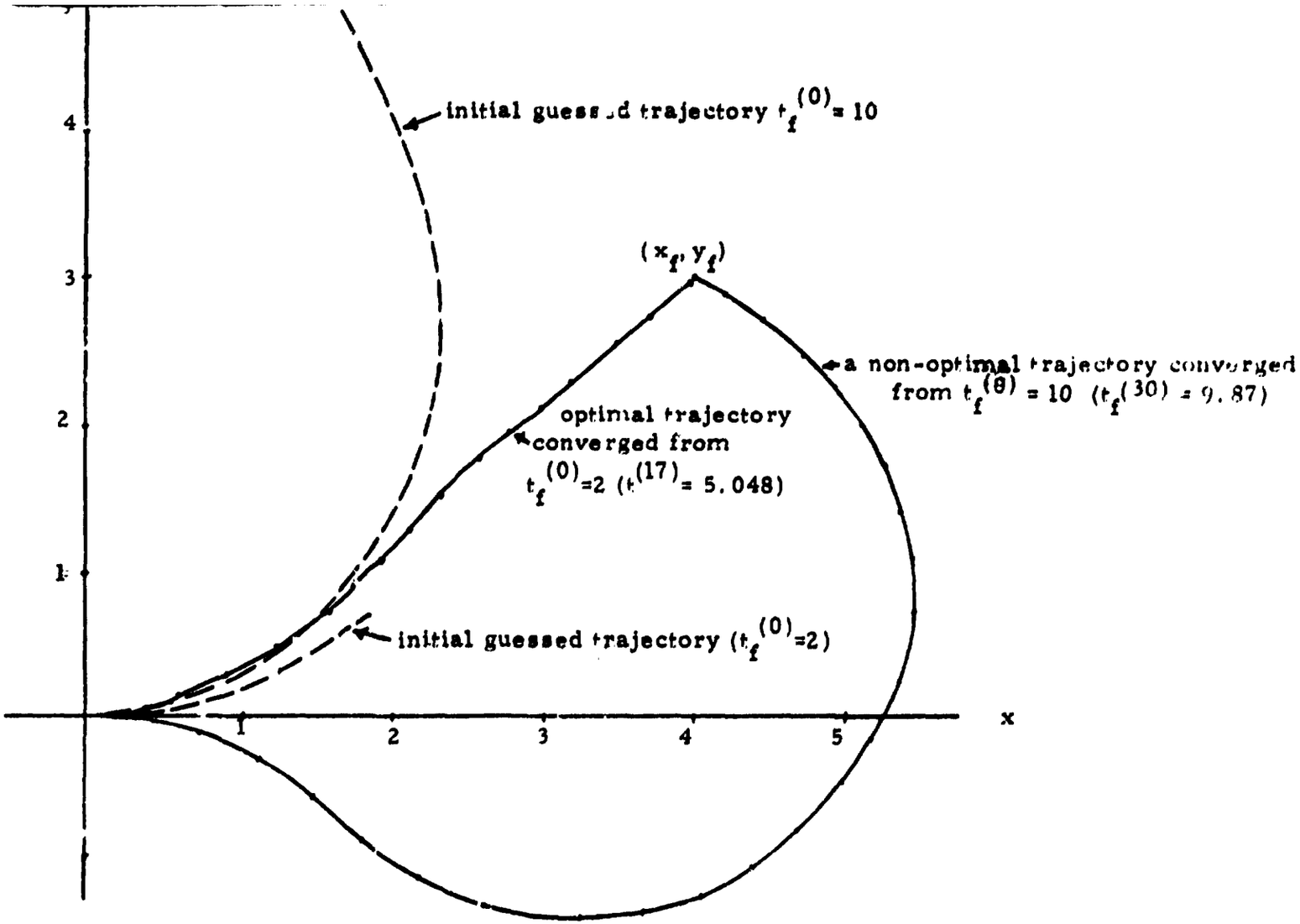


Figure 5.3 Trajectories for Example 1, Case B-b

the control profile moves in the wrong direction (see Figs. 5.2.b and 5.3).

Example 2.Flight in a Horizontal Plane

The equations of motion for a coordinated turn in a horizontal plane, with the thrust always aligned with the velocity, are

$$\frac{dx}{dt} = V \cos \beta$$

$$\frac{dy}{dt} = V \sin \beta$$

$$\frac{dV}{dt} = \frac{(T-D)}{m}$$

$$V\frac{d\beta}{dt} = \frac{L \sin \sigma}{m} \qquad (5.14)$$

$$\frac{dm}{dt} = \frac{-T}{c}$$

To maintain the aircraft in the horizontal plane, an algebraic constraint is imposed

$$L \cos \sigma = mg, \qquad (5.15)$$

and the parabalic drag polar is assumed, i.e.,

$$C_D = C_{DO} + K C_L^2, \qquad (5.16)$$

where C_{DO} and K are independent of the Mach number and the Reynolds number and

$$L = \frac{1}{2}\rho S C_L V^2$$

$$D = \frac{1}{2}\rho S C_D V^2. \qquad (5.17)$$

Two of the three functions T, β, σ may be identified as controls with the third determined by the constraint (5.15). In this example thrust magnitude and bank angle are controls which are all bounded, i.e.,

$$\sigma(t)_{min} \leq \sigma(t) \leq \sigma(t)_{max}$$

$$T_{min} \leq T \leq T_{max} \tag{5.18}$$

$\sigma(t)_{min} = -\sigma(t)_{max}$ for a symmetrical aircraft.

The cost functional to be minimized is

$$J = C t_f^2 + P_1 (x - x_f)^2 + P_2 (y-y_f)^2 + P_3 (V-V_f)^2 + P_4 (\beta - \beta_f)^2 + P_5 (m-m_f)^2$$

$$+ C_1 (x-x_f) + C_2 (y-y_f) + C_3 (V-V_f) + C_4 (\beta - \beta_f) + C_5 (m-m_f). \tag{5.19}$$

A relatively simple case is selected to show how the initial final time estimate will affect the performance.

Initial Conditions	Terminal Conditions
$x_o = 0$	$x_f = 6$ miles
$y_o = 0$	y_f = free
$V = 2.2$ Mach $\sim$ 2136.2 ft/sec	V_f = free
$\beta_o = 0$	β_f = free
$W_o = 861$ lbs	$W_f = 434$ lbs

Again the penalty function method is used, and $P_1 = 1000, P_2 = P_3 = P_4 = P_5 = 0$, $C_i = 0$, $i = 1, \ldots, 5$, $C = 1$.

The optimal solution for this case is $t_f = 5.75$ seconds. The thrust profile is the boost-coast type and the bank angle is zero for all time.

Three final time estimates are considered. (Figure 5)

1. $t_f^{(0)} = 4$ seconds, program forces the final time to the neighborhood of t_f^* in three iterations and obtains $t_f = 5.79$ on the fourteenth iteration, with the boundary condition error less than 0.2 percent (see Fig. 5.4.a).

2. $t_f^{(0)} = 6$ seconds. This is again an estimate close to optimal, but slightly longer than the true minimum. The control profile approached the boost-coast type and ended up with $t_f^{(14)} = 5.94$ in fourteen iterations. (see Fig. 5.4.b).

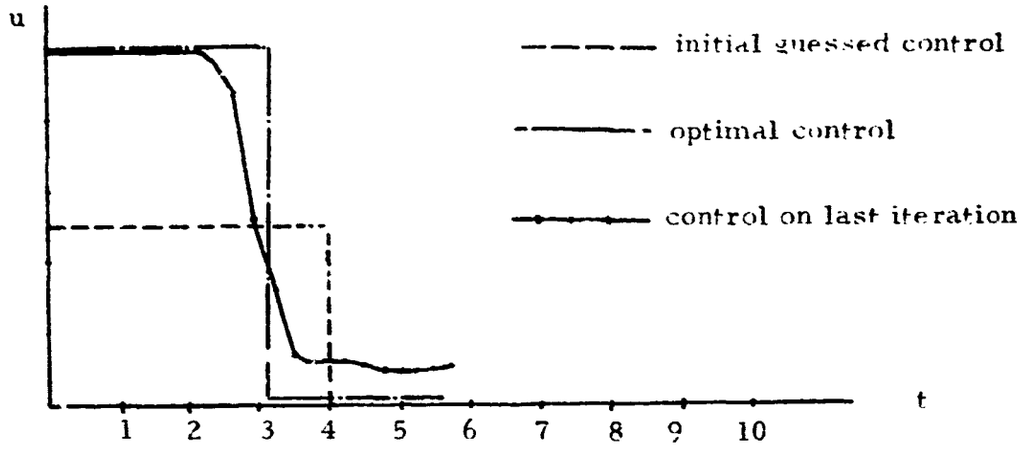
3. $t_f^{(0)} = 10$ seconds. After twenty-three iterations the terminal position error was less than .1 percent ($x_f = 6.00003$), but there was insignificant improvement in flight time, $t_f = 9.85$. At the thirtieth iteration, t_f started to improve to 8.76 which is still far from the true minimum time. The thrust control profile tends to the coast-boost type which is far from the optimal solution (see Fig. 5.4.c). Heuristic reasons for the behavior in the examples above are given in Section 5.3.

5.3 Method of Multipliers

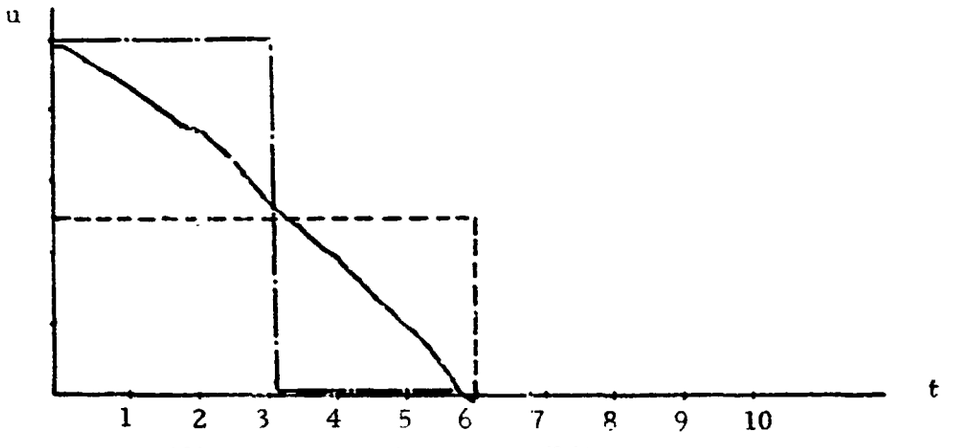
A brief comparison of the penalty function method and the modified multiplier method (M M - 2), Ref. 19, was undertaken in the study. Based upon the theory by Hestenes[18], M M - 2 should perform better than the penalty function method. Our experience has been that, with the conjugate-gradient algorithm, some improvement does occur. However, the improvement is not significant enough to justify the additional programming.

5.4 Conclusions

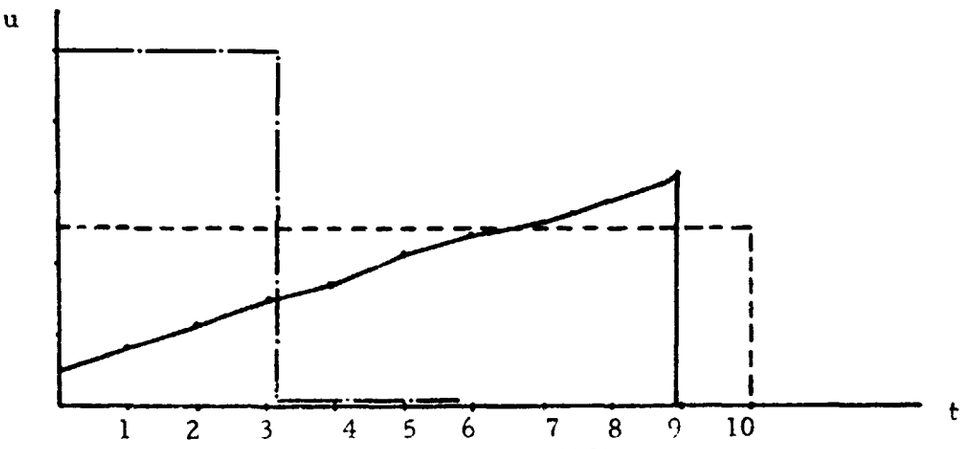
The examples in Section 5.2 demonstrate numerically that the choice $t_f^{(0)} \neq t_f^*$ appears to improve considerably the performance of gradient-type methods when penalty functions are employed. Although



(a) $t_f^{(0)} = 4$ sec, $t_f^* = 5.75$, $t_f^{(13)} = 5.79$ sec



(b) $t_f^{(0)} = 6$ sec., $t_f^* = 5.75$ $t_f^{(14)} = 5.946$ sec



(c) $t_f^{(0)} = 10$ sec., $t_f^* = 5.75$, $t_f^{(30)} = 5.77$ sec

Figure 5.4 Control profiles for Example 2.

a mathematical proof of this fact (which would involve a rate-of-convergence-type proof) has not been obtained to date, the following heuristic argument is offered in support of the possible generality of this observation.

Consider a time optimal control problem with terminal constraints, where t_f^* is the optimal final time. Suppose $t_f^{(0)} < t_f^*$. Then, it is impossible for the initial trajectory to meet the boundary conditions, and $t_f^{(1)}$ must be greater than $t_f^{(0)}$ to decrease the error on the terminal constraints. Thus, the optimal solution has the unique characteristic of being the closest trajectory to the initial iterate, with respect to final time, which satisfies the terminal constraints. On the other hand, if $t_f^{(0)} > t_f^*$, then it is probable that there exist infinitely many near-by solutions which satisfy the terminal constraints. Since with penalty functions terminal constraint satisfaction is a major part of the performance index, there exists the tendency to "lock-in" on the terminal conditions at $t_f > t_f^*$. That is, the optimal solution no longer possesses the unique property of being the closest trajectory which satisfies the boundary conditions. With regard to mathematical implications, the statements above imply that the minimum is "flatter" if $t_f^{(0)} > t_f^*$ than if $t_f^{(0)} < t_f^*$.

CHAPTER 6

THE PRAXIS ALGORITHM

In the previous chapters function space algorithms for minimization have been studied. In this chapter we shall consider a recently developed parameter optimization scheme which does not require the objective function to be differentiable. Such a scheme is of use in problems where it is difficult or even impossible to find the partial derivatives of the objective function directly.

We shall first discuss Powell's method[20] and the modifications due to Fletcher[23] and Brent[25]. Some specific properties which are closely related to convergence are presented along with an application of the method to a time-optimal control problem. Also, the subroutine of Appendix D has been built into the NASA-JSC PEACE parameter optimization program.

6.1 Powell's Algorithm

The basic concept of Powell's Algorithm is to minimize a scalar function of n variables, say $f(x^1, \ldots, x^n)$, by searching along n directions which span the space. Thus, for one iteration, the basic procedure is as follows:

Let x_J be the estimate of the vector x on the J^{th} iterate, and $u_1, \ldots, u_n$ be vectors which span the space (initially $[u_{ij}]$ is the nxn identity matrix). Then:

1. For i = 1, . . . , n, compute β_i to minimize $f(x_{J,\,i-1} + \beta_i u_i)$, where $x_{J,i} \stackrel{=}{=} x_{J,\,i-1} + \beta_i u_i$ and $x_{J,0} \stackrel{=}{=} x_J$.
2. For i = 1, . . . , n-1, replace u_i by u_{i+1}.
3. Replace u_n by $x_{J,n} - x_J$.
4. Compute β to minimize $f(x_J + \beta u_n)$, define $x_{J+1} \stackrel{=}{=} x_J + \beta u_n$, return to (1).

A simple graphical example will clarify the iteration procedure. Consider an ellipse in two-dimensional space (see Fig. 6.1). The algorithm starts at (x_o, y_o), and on the first iteration, the search directions u_1, u_2 are (1, 0) and (0, 1), which are along the x and y directions.

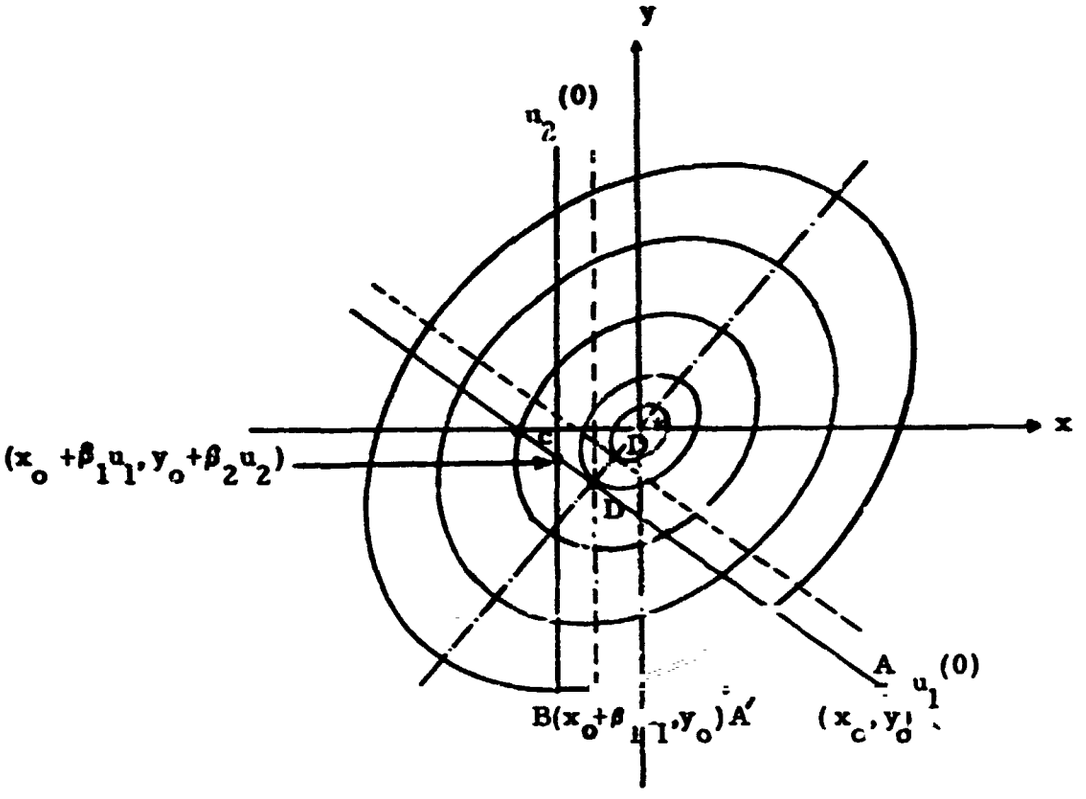


Fig. 6.1 Operation of Powell's Method.

Following Step 1, the algorithm searches along the $u_1^{(0)}$ direction, and B is the resultant point along the $u_1^{(0)}$ - axis. Then, the algorithm searches along the $u_2^{(0)}$ direction from the point $(x_o + \beta_1 u_1^{(0)}, y_o)$, and C is the resultant point. Steps 2 and 3 require the new search directions to be: $u_1^{(1)} = u_2^{(0)}$, $u_2^{(1)} = (\beta_1 u_1^{(0)}, \beta_2 u_2^{(0)})$, where $u_2^{(1)}$ is actually the direction along AC. Finally, Step 4 implies that point D is the value of the new estimate, i.e., $x_1 = x_o + \beta u_2^{(1)}$. The procedure is then repeated, and for this example, convergence will be obtained on the second iterate.

On the second iterate, the new direction D D* is conjugate to $u_2^{(1)}$ according to a theorem developed by Powell[20]. The minimum is obtained by performing an additional search along this direction.

This example demonstrates that the algorithm converges in a finite number of iterates for quadratic functions. As one might expect, the property of conjugacy plays an important role in this connection, and more details will be presented in the next section.

This basic procedure has the defect, as pointed out by Zangwill [22], that a poor guess of the initial position (e.g., point B in Fig. 6.1) might lead the algorithm to fail to find the minimum. Instead, the algorithm will converge to a minimum along the line $u_2^{(0)}$, which defines a proper subspace of the space R^2.

In order to overcome this difficulty, both Powell[20] and Zangwill [22] proposed methods to retain the linear independence. Numerical experiments in Ref. 25 show that Powell's modification is preferable

to Zangwill's.

6.2 The Roles of Conjugacy, Orthogonality and Independence

By definition, two vectors u_1 and u_2 are said to be conjugate with respect to the positive definite symmetric matrix A if

$$u_1^T A u_2 = 0.$$

A set of conjugate directions is a set in which the vectors are pairwise conjugate.

Consider a quadratic function $f(x, y) = x^2 + 4y^2$, with elliptical contours as shown in Figure 6.2. Writing the function in matrix form,

$$f(x,y) = [\ x, y\ ] \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad (6.3)$$

with the A matrix

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 4 \end{bmatrix} \qquad (6.4)$$

and the two unit vectors along the x and y axes

$$u_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad u_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad (6.5)$$

It is obvious that u_1 and u_2 are conjugate.

With Powell's method one can obtain the minimum by searching along each of the conjugate directions once as long as the space on which the function is defined is spanned by the conjugate vectors. From Figure 6.2, one can easily see that the minimum is obtained by searching along the x- and y- directions only once regardless of the initial guessed position.

Now consider the same function in a coordinate system rotated 45° from the original system:

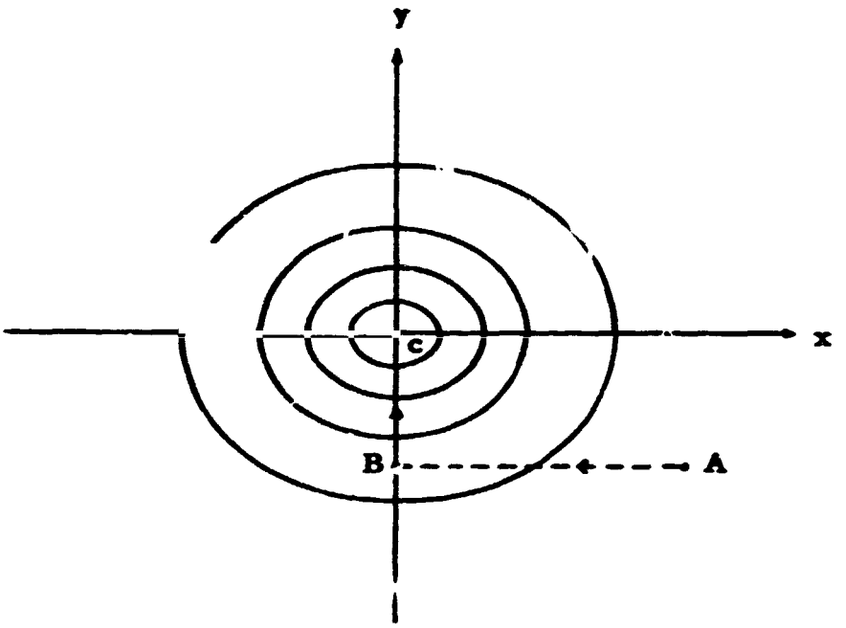


Figure 6.2 Case when initial search directions are principal axes.

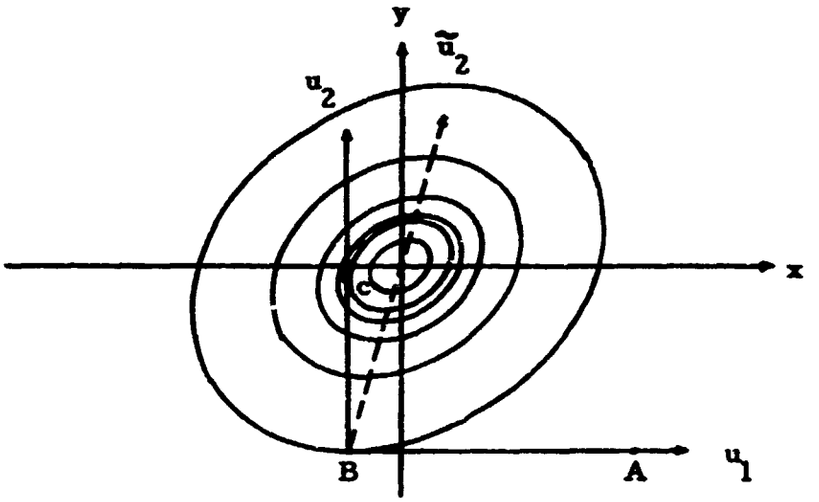


Figure 6.3. Convergence characteristics for nonconjugate and and conjugate search directions.

$$f(x,y) = [\,x,y\,]\begin{bmatrix} \frac{5}{2} & -\frac{3}{2} \\ -\frac{3}{2} & \frac{5}{2} \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \qquad (6.6)$$

From Figure 6.3, successive searches along the x and y axes will not reach the minimum due to the fact that the x and y axes are no longer conjugate. On the other hand, the minimum can always be reached by successive searches along two conjugate directions. For example, consider

$$u_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad u_2 = \begin{bmatrix} 3/5 \\ 1 \end{bmatrix} .$$

These vectors are conjugate since

$$u_1^T A u_2 = [\,1 \;\; 0\,]\begin{bmatrix} \frac{5}{2} & -\frac{3}{2} \\ -3/2 & \frac{5}{2} \end{bmatrix}\begin{bmatrix} \frac{3}{5} \\ 1 \end{bmatrix} = 0$$

and, as can be seen in Figure 6.3, the minimum is reached by successive searches along $u_1 = [\,1\,,\,0\,]$ and $u_2 = [\,\frac{3}{5}\,,\,1\,]$.

Since the principal axes of a quadratic function are orthogonal and also A-conjugate, one can always find the minimum by searching along the principal axes once only. The algorithm modification by Brent is essentially based on this concept, i.e., it is to find the principal axes of the function f (or its quadratic approximation) and to search along the principal axes to obtain the minimum.

6.3 PRAXIS - A Modification of Powell's Method

Because of the deficiencies of Powell's method, e.g., the loss of linear independence and conjugacy, Brent [25] developed a modified version called PRAXIS. The main modifications are:

1) A restart device is included to reset the search directions to a set of orthogonal, A-conjugate vectors after every n or n+ 1 iterations to insure the linear independence of the new search directions. These conjugate vectors are computed on the assumption that f is quadratic or is the quadratic approximation of the function to be minimized. If f is quadratic or if the quadratic approximation is good, then the new search directions are conjugate with respect to a matrix which is close to the Hessian matrix of f at the minimum. This resetting method will prevent the scheme from searching for a minimum in a subspace.

2) A random step is inserted to enable the scheme to search for another initial point in each iteration if the most recent linear search has failed to improve the current approximation to the minimum. With this step in the scheme, the trouble noted by Zangwill will be avoided.

For example, in Figure 6.1 if point B is chosen as the initial point, then Powell's basic procedure will find C as the minimum and stop, as noted by Zangwill. Powell's modified procedure will retain the old search vectors as the new search direction for the next iteration; hence one more iteration is needed to reach the minimum.

With the random step, the algorithm will replace point B by an arbitrary point in the space, say A' in Figure 6.1, after having failed to obtain an improvement in the direction of $u_1 = (1, 0)$. This rules out the possibility of linearly dependent search directions.

3) Discarding criterion. Powell's modification proposed that the search direction should be discarded and replaced by one which maximizes $| \det (V_1 \ldots . V_n) |$, where

$$V_i = (u_i^T A u_i)^{\frac{-1}{2}} u_i, \quad 1 \leq i \leq n \qquad (6.7)$$

A: nxn matrix related to the quadratic approximation

This discarding method may lead to the elimination of one of the mutually conjugate directions, in which case finite convergence for a quadratic function can no longer be assured.

In PRAXIS the criterion described below is employed. It is essentially Powell's criterion except an additional restriction is imposed to insure the finite convergence for a quadratic function property.

The discarding criterion for PRAXIS is as follows:

At K^{th} iteration with search direction $u_1, \ldots, u_n$ and $|\det (V_1, \ldots, V_n)|_o$

(a) For $i = 1, \ldots, n-k+1$, take u_i out of $u_1, \ldots, u_{n-k+1}$, and compute

$$V_J = [(x_n - x_o)^T A (x_n - x_o)]^{\frac{-1}{2}} (x_n - x_o)$$

(b) Compute $|\det (V_1, \ldots, V_{i-1}, V_{i+1}, \ldots, V_{n-1}, V_J)| = D_i$, $i = 1, \ldots, n-k+1$.

(c) If no D_i is larger than the value $|\det (V_1, \ldots V_n)|_o$, then no replacement for the search direction occurs. Otherwise go to (d).

(d) For $D_m = \text{Max}(D_i)$ and $D_m > |\det (V_1, \ldots, V_n)|_o$, renumber $u_1 = u_1, \ldots, u_{m-1} = u_{m-1}, u_m = u_{m+1}, \ldots, u_{n-1} = u_n$, and $u_n = x_n - x_o$.

Thus, at the K^{th} iteration, only one of $u_1, \ldots, u_{n-k+1}$ is permitted to be discarded.

4) The linear search in PRAXIS is similar to Powell s procedure. It reduces the number of function evaluations considerably. For example,

consider the linear search in the direction u, i.e., minimize

$$\psi(\lambda) = f(x_o + \lambda u), \tag{6.8}$$

At the first iteration three function evaluations are needed for a quadratic curve - fit, say $p(\lambda) = a\lambda^2 + b\lambda + c$. The seond derivativeof $P(\lambda)$, i.e., a, is saved because it can be used in the next iteration when this search direction is utilized again.[20] Then, the approximation for the second derivative of $p(\lambda)$ is always available if a linear search in the direction u has already been performed or if u resulted from a singular value decomposition, which is the step to find the principal axis vectors in PRAXIS. Thus, only two additional function values are needed for the three constants a, b, c, where $a = \psi''(0)$ after the first iteration.

6.4 Examples

Zermelo's problem is used to demonstrate the efficiency and reliability of the algorithm. Long's method[17] is used to transform this variable time problem into a fixed final time problem. The equations of motion are

$$\begin{aligned} \dot{x} &= V\cos\theta \\ \dot{y} &= V\sin\theta \\ \dot{\theta} &= u, \quad |u| \leq 0.5 \end{aligned} \tag{6.9}$$

and the performance index is

$$J = Ct_f^2 + P_1(x(t_f) - x_f)^2 + P_2(y(t_f) - y_f)^2 + P_3(\theta(t_f) - \theta_f)^2 \tag{6.10}$$

Performing Long's transformation, i.e., $t = as$, $s \in [0,1]$,

$$dt = a\,ds, \qquad t_f = a \text{ at } s = 1, \quad (\quad)' = \frac{d(\quad)}{ds}$$

Thus,

$$x' = a\, V \cos\theta$$

$$y' = a\, V \sin\theta$$

$$\theta' = au$$

$$a' = 0 \tag{6.11}$$

and

$$J = Ca^2 + P_1 [\, x(1) - x_f \,]^2 + P_2 [\, y(1) - y_f \,]^2 + P_3 [\, \theta(1) - \theta_f \,]^2 \tag{6.12}$$

To employ PRAXIS, the control u(s) must be discretized. Let $u(s) = u_1, \; s \in [\, 0, s_1)$; $u(s) = u_2, \; s \in [\, s_1, s_2 \,]$; . . . ; $u(s) = u_n, \; s \in [\, s_{n-1}, 1]$, and consider the cost J to be minimized as a function of the $n + 1$ variables a, $u_1, \ldots, u_n$, i.e.,

$$J = J [\, a, u_1, u_2, \ldots, u_n] \tag{6.13}$$

The problem was then attacked with PRAXIS for three different initial estimates for a, i. e., $t_f^{(0)}$. The results are summarized in Table 6.1.

Table 6.1. Parameters and Results with PRAXIS.

Constants: $C = 1$

$P_1 = P_2 = 10000,\ P_3 = 0$

$n = 10,$

$u_i^{(0)} = 0.2,\ i = 1, \ldots, 10$

$|u_i| \leq .5$

$x_f = 4.0,\ y_f = 5.0$

Optimal Control: $u_i = .5$ for $s \in [0, .3]$

$u_i = 0.0$ for $s \in [0.3, 1]$

CASE	$a^{(0)} = t_f^{(0)}$	final	a*	x(1)	y(1)	Figure
1	1	6.734	6.71	3.997	5.001	6.4.a
2	4	6.730	6.71	3.999	4.999	6.4.b
3	10	6.7408	6.71	3.999	4.999	6.4.c

The cost reduction versus number of linear searches for cases 1 and 3 are shown in Figure 6.5, while the various control profiles are shown in Figure 6.4. In addition, the cost for $t_f^{(0)} = 40$ is shown in Figure 6.5. This trial converged to a local minimum and no longer improved the cost. Note that the trials with $t_f^{(0)} = 1$ and $t_f^{(0)} = 10$ converged to the neighborhood of the minimum cost rapidly. However, the control profiles, in a sense, oscillate about the optimal control. Of course, this behavior could be improved by assuming a more representative parameterized control.

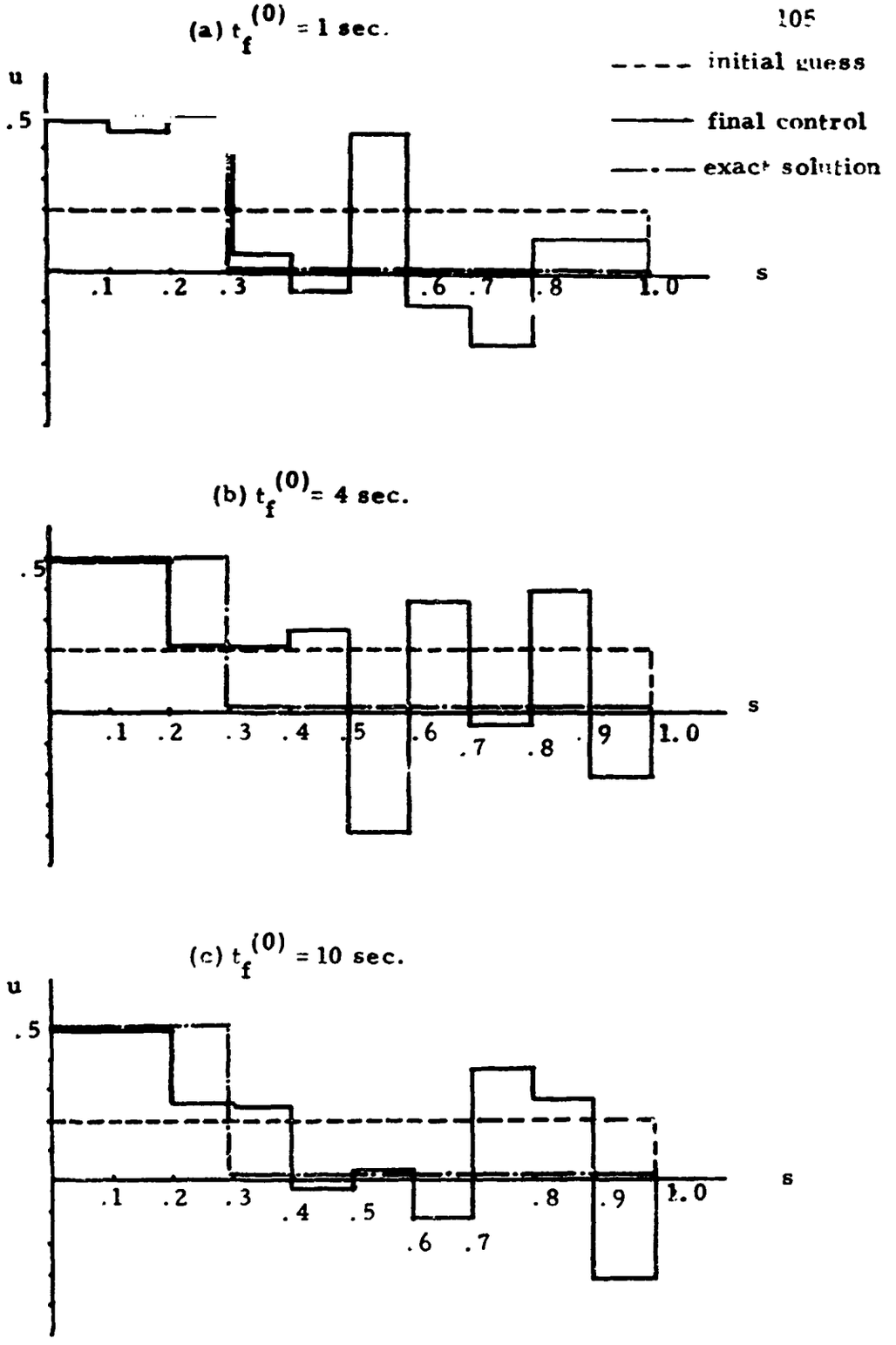


Figure 6.4. Control profiles using PRAXIS with various initial estimates of t_f.

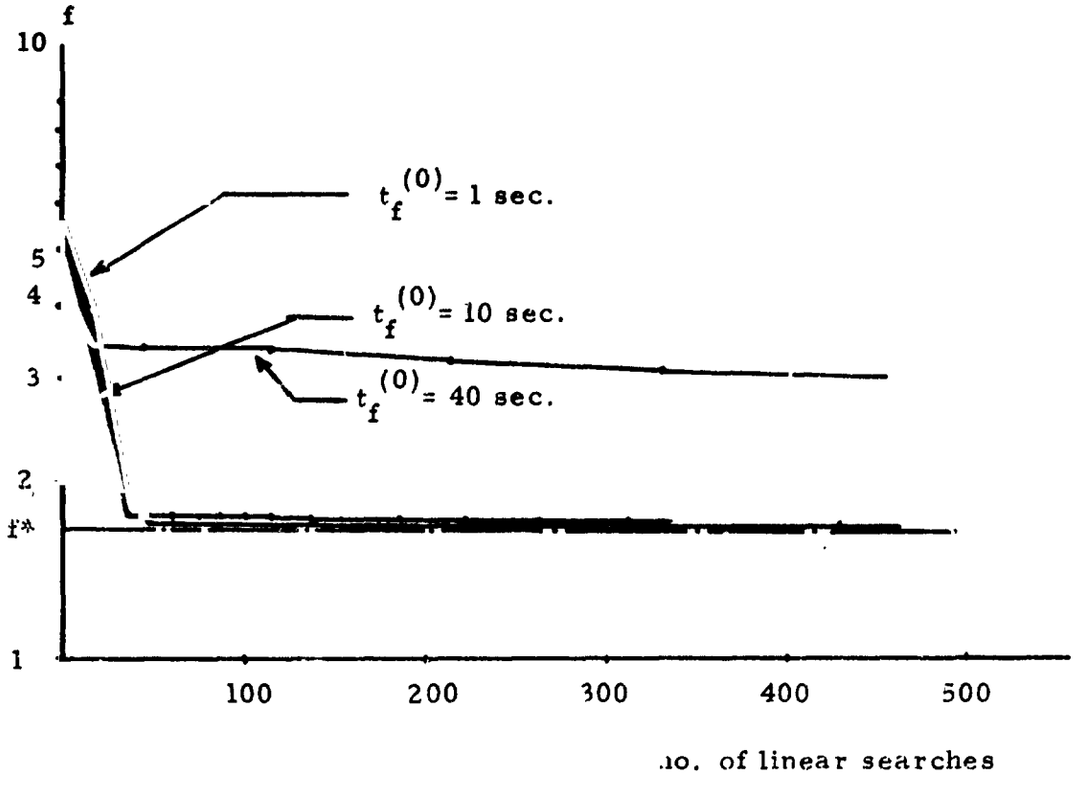


Figure 6.5 Cost versus number of linear searches with PRAXIS.

CHAPTER 7

CONCLUSIONS AND RECOMMENDATIONS

7.1 Summary

Computer programs for shuttle trajectory optimization have been developed and delivered to NASA-JSC. One of the programs contains the function-space gradient, conjugate-gradient, Davidon, and Broyden algorithms for the ascent problem. The PRAXIS parameter optimization scheme has been integrated into the NASA-JSC PEACE parameter optimization program.

7.2 Conclusions and Recommendations

1.) The function-space Broyden and Davidon methods performed appreciably better on the shuttle ascent problem than the gradient and conjugate-gradient algorithms, with Broyden slightly better than Davidon. Both control and state variable inequality constraints were included in the formulation with the control constraints handled directly while the state variable constraints were included with penalty functions.

2.) The storage problems associated with function-space Davidon-type techniques have been overcome. Although considerable storage is necessary for the computation of inner products, the storage need not be in fast memory. On the University of Michigan computer the storage problem is handled very easily by a disk file storage system. The programs for use on the NASA-JSC computer require modifications for drum storage.

3.) The University of Michigan computer is a time-sharing system with an interactive-graphics capability. This capability accelerates considerably the time required to converge a large-scale optimization problem. For example, standard operation with the NASA-JSC PEACE parameter optimization program (when a large number of parameters is involved) usually involves: making a single computer run daily, analysis of the result, adjustment of parameters (usually penalty coefficients), and resubmission of the program. This means that the analyst must stop-and-start on the same problem many times, and the process is a somewhat inefficient use of the analyst's time. With a time-shared, interactive graphics capability, the analyst can stay with the problem continuously for longer periods of time with the result being: less total computer time, less total human effort, more physical knowledge of the problem, and more rapid solution of the problem. Thus, it is recommended that MPAD consider the use of interactive-graphics terminals in the solution of large-scale trajectory optimization and mission analysis problems.

4.) Previous investigators have noted difficulties in solving variable final-time trajectory optimization problems with accelerated-gradient methods. In Chapter 5 heuristic arguments and simulations indicate that the initial estimate of t_f is critical, and $t_f^{(0)} < t_f^*$ improves the convergence rate considerably.

5.) Due to budget limitations, the PRAXIS algorithm could not be simulated on realistic shuttle trajectory optimization problems. The worth of this algorithm will be determined by NASA-JSC personnel.

REFERENCES

1. Pagurek, B., and Woodside, C.M., "The Conjugate Gradient Method for Optimal Control Problems with Bounded Control Variables," Automatica, Vol. 4, pp 337-349, 1968.

2. Horwitz, L.B., and Sarachik, P.E., "Davidon's Method in Hilbert Space," SIAM J. Appl. Math., Vol. 16, pp 676-695, 1968.

3. Broyden, C.G., "The Convergence of a Class of Double-Rank Minimization Algorithms: 2. The New Algorithm," J. of Inst. of Math. and Appl. Vol. 6, pp 222-231, 1970.

4. Lasdon, L.S., "Conjugate Direction Methods for Optimal Control", IEEE Trans. on Auto. Control, Vol. AC-15, pp 267-268, 1970.

5. Lasdon, L.S., Mitter, S.K., and Waren, A.D., "The Conjugate Gradient Method for Optimal Control Problems," IEEE Trans. on Auto. Control, Vol. AC-12, pp 132-138, 1967.

6. Klessig, R., and Polak, E., "Efficient Implementations of the Polak-Ribiere Conjugate Gradient Algorithm," SIAM J. Control, Vol. 10, pp 524-549, 1972.

7. Tripathi, S.S., and Narendra, K.S., "Optimization Using Conjugate Gradient Methods," IEEE Trans. on Auto Control, Vol. AC-15, pp 268-270, 1970.

8. Porter, W.A., Modern Foundations of Systems Engineering, Macmillan, New York, Chap. 3, 1966.

9. Friedman, B., Principles and Techniques of Applied Mathematics, J. Wiley and Sons, New York, 1966.

10. Bryson, A.E. Jr., and Ho, Y.C., Applied Optimal Control, Blaisdell, Waltham, Mass., Chap. 7, 1969.

11. Kelley, H.J., Uzzell, B.R., and McKay, S.S., "Rocket Trajectory Optimization by a Second-Order Numerical Technique," AIAA J., Vol. 7 pp 879-884, 1969.

12. Mitter, S.K., "Successive Approximation Methods for the Solution of Optimal Control Problems," Automatica, Vol. 3, pp 135-149, 1966.

13. Jacobson, D.H., and Mayne, D.Q., Differential Dynamic Programming, American Elsevier New York, 1970.

14. Pierson, B.L., and Rajtora, S.G., "Computational Experience with the Davidon Method Applied to Optimal Control Problems," IEEE Trans. on Sys. Sci. and Cyber., pp 240-242, 1970.

15. Lasz1o, W.O., and Sullivan, H.C., "Optimal Shuttle Ascent Trajectories Using the PEACE Program," NASA-MSC Memorandum FM94 (72-82), June 16, 1972.

16. Tripathi, S S., and Narendra, K.S., "Constrained Optimization Problems by the Multiplier Method," J. of Optimization Theory and Applications, Vol. 9, pp. 59.70, 1972.

17. Long, R.S., "Newton-Raphson Operator; Problems with Undetermined Endpoints," AIAA J. Vol. 3, pp. 1351-1352, 1965.

18. Hestenes, M.R., "Multiplier and Gradient Methods," J. of Optimization Theory and Applications, Vol. 4, pp. 303-320, 1969.

19. Miele, A., Moseley, P.E., Levy, A.V., and Coggins, G.M., "On the Method of Multipliers for Mathematical Programming Problems," J. of Optimization Theory and Applications, Vol. 10, pp. 1-32, 1972.

20. Powell, M.J.D., "An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives," Computer Journal, Vol. 7, pp. 155-162, 1964.

21. Powell, M.J.D., "A Method for Minimizing a Sum of Squares of Nonlinear Functions Without Calculating Derivatives," Computer Journal, Vol. 7, pp. 303-307, 1965.

22. Zangwill, W.I., "Minimizing a Function Without Calcuiating Derivatives," Computer Journal, Vol. 10, pp. 293-296, 1967.

23. Fletcher, R., "Function Minimization Without Evaluating Derivatives-A Review," Computer Journal, Vol. 8, pp. 33-41, 1965.

24. Powell, M.J.D., "A Survey of Numerical Methods for Unconstrained Optimization," SIAM Review, Vol. 12, pp. 79-97, 1970.

25. Brent, R.P., Algorithms for Minimization Without Derivatives, Prentice-Hall, Chapter 7, 1973.

26. Smith, O.E., and Weidner, D.K., "A Reference Atmosphere for Patrick AFB, Florida," (1963 Revision) NASA TM X-53139, NASA-MSFC, 19. .

27. Kamm, J.L., and Kim . I.J., "A Fast and Precise 1963 Patrick Atmosphere Analytical Model," TRW-Houston IOC 5521.5-13, December 2, 1970.

Appendix A

Dynamical Equations of Motion: First Stage

As is customary in trajectory optimization the vehicle is modeled as a point mass. It is further assumed that the thrust, aerodynamic, and gravitational forces act through the center of mass. By Newton's Second Law,

$$\sum \bar{F} = m \ddot{\bar{r}}, \tag{A.1}$$

where $\ddot{\bar{r}}$ is measured in an inertial coordinate system. Since numerical integration is desired in the first stage, consider Figure A.1. The acceleration of $\bar{r}$ is:

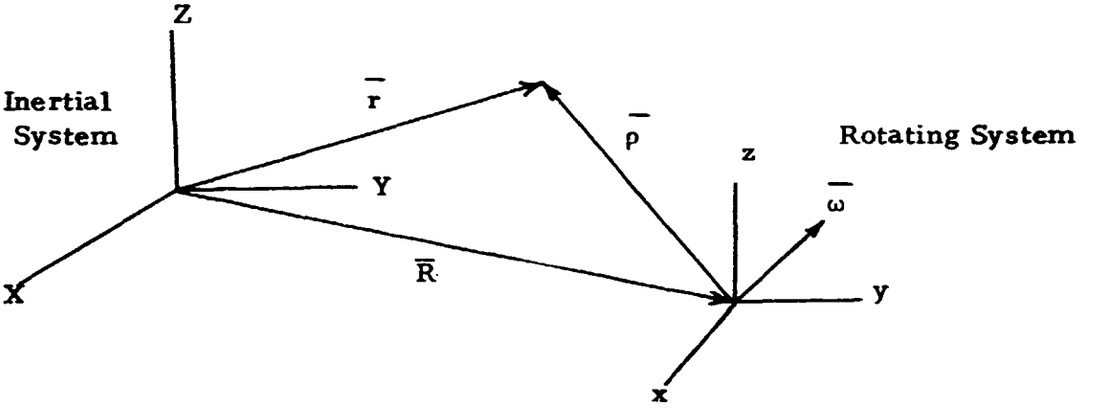


Figure A.1. Rotating Coordinate System Definitions

$$\ddot{\bar{r}} = \ddot{\bar{R}} + \dot{\bar{\omega}} \times \bar{\rho} + \bar{\omega} \times (\bar{\omega} \times \bar{\rho}) + \ddot{\bar{\rho}}_{ROT} + 2\bar{w} \times \dot{\bar{\rho}}_{ROT} \tag{A.2}$$

Consider two coordinate systems fixed at the center of the earth, one of which rotates with the earth and the other inertial. Then,

i) $\bar{R} = 0$, since both coordinate systems are fixed at the same point.

ii) $\bar{\omega}$ = constant $\Rightarrow \dot{\bar{\omega}} = 0$, since rototation of earth about its axis is constant.

iii) $\bar{r} = \bar{\rho}$; follows from $\bar{r} = \bar{R} + \bar{\rho}$ and i) $\bar{R} = 0$.

Thus the acceleration is

$$\ddot{\bar{r}} = \ddot{\bar{r}}_{rot} + \dot{\bar{\omega}} \times (\bar{\omega} \times \bar{r}) + 2\bar{\omega} \times \dot{\bar{r}}_{ROT} \qquad (A.3)$$

Now consider the two spherical coordinate systems centered at the center of the earth shown in Figure A.2. By definition

$$\bar{r} = r\bar{e}_r = R\bar{e}_R \,, \quad \bar{e}_r = \bar{e}_R \Longleftrightarrow r = R.$$

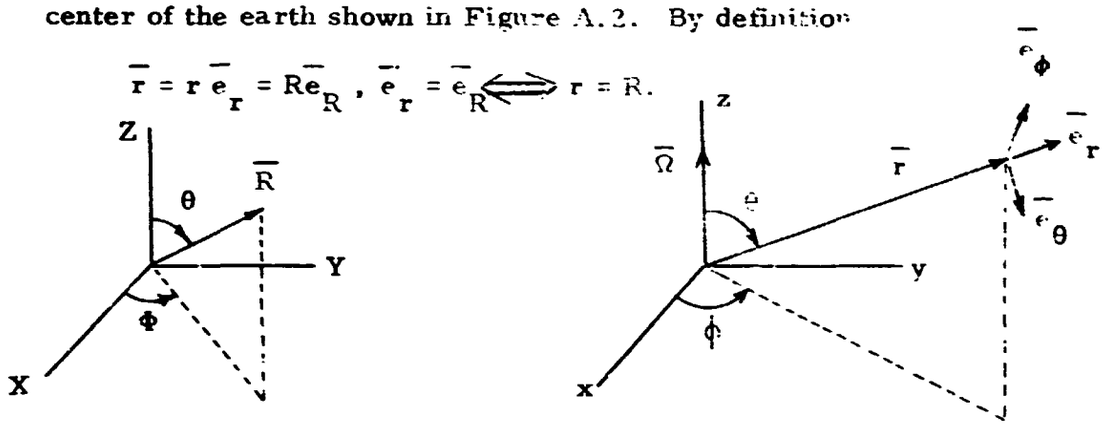


Figure A.2. First-Stage Coordinate System.

Since $\bar{\Omega}$ is along the z-axis,

$$\bar{\Omega} = \Omega[\text{Cos}\,\theta\ \bar{e}_r - \text{Sin}\,\theta\ \bar{e}_\theta],$$

which implies

$$\bar{\Omega} \times \bar{r} = \begin{bmatrix} \bar{e}_r & \bar{e}_\theta & \bar{e}_\phi \\ \Omega\,\text{Cos}\,\theta & -\Omega\,\text{Sin}\,\theta & 0 \\ r & 0 & 0 \end{bmatrix} = r\Omega\,\text{Sin}\,\theta\ \bar{e}_\phi$$

$$\bar{\Omega} \times (\bar{\Omega} \times \bar{r}) = \begin{bmatrix} \bar{e}_r & \bar{e}_\theta & \bar{e}_\phi \\ \Omega\,\text{Cos}\,\theta & -\Omega\,\text{Sin}\,\theta & 0 \\ 0 & 0 & r\Omega\,\text{Sin}\,\theta \end{bmatrix}$$

$$= -[\, r\Omega^2\,\text{Sin}^2\,\theta\,]\,\bar{e}_r - [\, r\Omega^2\,\text{Sin}\,\theta\,\text{Cos}\,\theta]\,\bar{e}_\theta$$

$$2\bar{\Omega} \times \dot{\bar{r}}_{ROT} = \begin{bmatrix} \bar{e}_r & \bar{e}_\theta & \bar{e}_\phi \\ 2\Omega \cos\theta & -2\Omega \sin\theta & 0 \\ \dot{r} & r\dot{\theta} & r\dot{\phi}\sin\theta \end{bmatrix}$$

$$= -[\, 2r\Omega\dot{\phi}\sin^2\theta \,]\,\bar{e}_r - [\, 2r\Omega\dot{\phi}\sin\theta\cos\theta \,]\,\bar{e}_\theta$$

$$+ [\, 2r\Omega\dot{\theta}\cos\theta + 2\dot{r}\,\Omega\sin\theta \,]\,\bar{e}_\phi$$

yet $\bar{T}$ = Thrust Force = $T_r \bar{e}_r + T_\theta \bar{e}_\theta + T_\phi \bar{e}_\phi$

$\bar{A}$ = Aerodynamic Force = $A_r \bar{e}_r + A_\theta \bar{e}_\theta + A_\phi \bar{e}_\phi$

$\bar{G}$ = Gravitational Force = $\dfrac{-m k}{r^2}\,\bar{e}_r$

Then, upon substitution into Eq. (A.1)

$$\ddot{r} - r\dot{\theta}^2 - r\dot{\phi}^2\sin^2\theta - r\Omega^2\sin^2\theta - 2r\Omega\dot{\phi}\sin^2\theta = \frac{1}{m}\left[\, T_r + A_r - m\frac{k}{r^2} \,\right] \qquad (A.4)$$

$$r\ddot{\theta} + 2\dot{r}\dot{\theta} - r\dot{\phi}^2\sin\theta\cos\theta - r\Omega^2\sin\theta\cos\theta - 2r\Omega\dot{\phi}\sin\theta\cos\theta = \frac{1}{m}\left[\, T_\theta + A_\theta \,\right] \qquad (A.5)$$

$$r\ddot{\phi}\sin\theta + 2\dot{r}\dot{\phi}\sin\theta + 2r\dot{\theta}\dot{\phi}\cos\theta + 2r\Omega\dot{\theta}\cos\theta + 2\dot{r}\,\Omega\sin\theta = \frac{1}{m}\left[\, T_\phi + A_\phi \,\right] \qquad (A.6)$$

Define:

$$\dot{r} = v_r$$

$$\dot{\theta} = v_\theta$$

$$\dot{\phi} = \frac{v_\phi}{r\sin\theta}$$

$\dot{m}$ = mass flow rate (A.7)

$F_r = A_r + T_r$

$F_\theta = A_\theta + T_\theta$

$F_\phi = A_\phi + T_\phi$

Since ϕ is an ignorable coordinate, the $\dot{\phi}$ - equation is neglected. Then, with the following state-control definitions

$$x_1 = r - R_o,\ x_2 = \theta,\ x_3 = u_r,\ x_4 = v_\theta,\ x_5 = v_\phi,\ x_6 = m,\ u = |\dot{m}|$$

the equations of motion are:

(r) $\dot{x}_1 = x_3$

(θ) $\dot{x}_2 = \dfrac{x_4}{(x_1 + R_o)}$

(u) $\dot{x}_3 = \dfrac{x_4^2 + x_5^2}{(x_1 + R_o)} \left[\dfrac{k}{(x_1+R_o)^2} + (x_1 + R_o)\Omega^2 \right] \sin x_2$

$+ 2\,\Omega x_5 \sin x_2 - \dfrac{F_r}{x_6}$

(v) $\dot{x}_4 = \dfrac{x_5^2}{(x_1+R_o)\tan x_2} - \dfrac{x_3\, x_4}{(x_1 + R_o)} + (x_1+R_o)\Omega^2 \sin x_2 \cos x_2$

$+ 2\,\Omega x_5 \cos x_2 + \dfrac{F_\theta}{x_6}$ (A.8)

(w) $\dot{x}_5 = -\dfrac{x_3\, x_5}{(x_1+R_o)} - \dfrac{x_4\, x_5}{(x_1 + R_o)\tan x_2} - 2\,\Omega x_4 \cos x_2$

$- 2\, x_3\, \Omega \sin x_2 + \dfrac{F_\phi}{x_6}$

(mass) $\dot{x}_6 = -u$

where x_1 = altitude above earth

$x_2 = \theta$

$x_3 = v_r$ -velocity in $\bar{e}_r$ direction

$x_4 = v_\theta$ - velocity in $\bar{e}_\theta$ direction

$x_5 = v_\phi$ - velocity in $\bar{e}_\phi$ direction

x_6 = mass of vehicle

$u = |\dot{m}|$ - mass flow rate

R_o = radius of earth

Ω - angular velocity of earth

k = gravitational constant of earth

Appendix B

Atmosphere and C_D Models

The 1963 Patrick Atmosphere model was used. Pressure and density ratios, and speed of sound data were obtained from Ref. 26 and curve fitted as functions of altitude according to the equations

$$\rho / \rho_{SL} = \exp (a_o + a_1 x + \ldots + a_{13} x^{13})$$

$$P / P_{SL} = \exp (b_o + b_1 x + \ldots + b_{13} x^{13})$$

$$a = \exp (c_o + c_1 x + \ldots + c_{13} x^{13})$$

$$x = \frac{\text{altitude (ft)} - 200,000}{100,000}$$

The coefficients a_i, b_i, c_i are given in Ref. 27

The data in Table B.1 were used to define the drag model. The drag force is given by

$$D = \frac{1}{2} \rho V^2 C_D A$$

For Mach numbers between those in the data table, interpolation by piecewise cubic splines was used.

Mach No. (M)	C_D
0	.028
0.2	.028
0.4	.028
0.5	.029
0.6	.032
0.8	.058
1.0	.110
1.1	.121
1.2	.123
1.5	.121
1.75	.115
2.0	.106
2.4	.088
3.0	.066
3.5	.055
4.0	.047
5.0	.039
6.0	.031
7.0	.025
8.0	.021
9.0	.019
10.0	.019

Table B.1 Drag Coefficient

Appendix C

Transformation Equations

At the end of first-stage burn we wish to determine both the inclination of the orbital plane and the initial conditions for the second-stage, non-rotating polar coordinate system. The plane of the orbit is determined at first-stage burnout because there is no out-of-plane thrusting in the second-stage. Consider the conditions at first-sta burnout:

$$\bar{r} = r\,\bar{e}_r$$

$$\bar{v} = \bar{v}_{ROT} + \bar{\Omega} \times \bar{r} = v_r\,\bar{e}_r + v_\theta\,\bar{e}_\theta + v_\phi\,\bar{e}_\phi + r\,\Omega \sin\vartheta\,\bar{e}_\phi$$

$$= v_r\,\bar{e}_r + v_\theta\,\bar{e}_\theta + (v_\phi + r\,\Omega \sin\theta)\,\bar{e}_\phi \qquad (C.1)$$

Consider the new polar system ($\tilde{r}$, $\tilde{\theta}$, $\tilde{v}_r$, $\tilde{v}_\theta$):

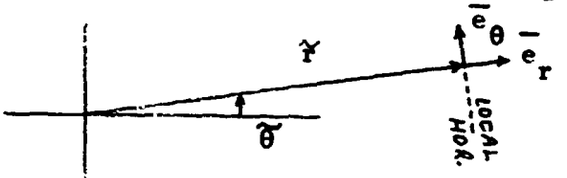


Figure C.1. Second-Stage Polar Coordinate System.

since $|\bar{V}| = \sqrt{v_r^2 + v_\theta^2 + (v_\phi + r\,\Omega \sin\theta)^2}$ and the radial components of the velocity are the same in each system, the velocity transformation is

$$\tilde{v}_r = v_r$$

$$\tilde{v}_\theta = \sqrt{v_\theta^2 + (v_\phi + r\,\Omega \sin\theta)^2} \qquad (C.2)$$

Thus, in state notation

$$\tilde{x}_1 = x_1$$

$$\tilde{x}_2 = x_3$$

$$\tilde{x}_3 = \sqrt{x_4^2 + [x_5 + (x_1 + R_o)\,\Omega \sin x_2]^2}, \qquad (C.3)$$

where $\tilde{x}_1 = \tilde{r} - R_o$, $\tilde{x}_2 = \tilde{v}_r$, $\tilde{x}_3 = \tilde{v}_\theta$. Note that the mass will change by the amount of structure discarded.

To obtain the inclination, consider the following unit vector which is perpendicular to the plane of the orbit:

$$\bar{N} = \bar{r} \times \bar{V} = \begin{bmatrix} e_r & e_\theta & e_\phi \\ r & 0 & 0 \\ v_r & v_\theta & (v_\phi + r\,\Omega \sin\theta) \end{bmatrix}$$

$$= r\,v_\theta\,\bar{e}_\phi - r\,(v_\phi + r\,\Omega \sin\theta)\,\bar{e}_\theta$$

$$\bar{e}_N = \frac{\bar{N}}{|\bar{N}|} = \frac{v_\theta}{\sqrt{v_\theta^2 + (v_\phi + r\,\Omega \sin\theta)^2}}\,\bar{e}_\phi - \frac{(v_\phi + r\,\Omega \sin\theta)}{\sqrt{v_\theta^2 + (v_\phi + r\Omega \sin\theta)^2}}\,\bar{e}_\theta$$

Let $\bar{k}$ = unit vector along the axis of rotation of the earth. Then,

$$\bar{k} = \cos\theta\,\bar{e}_r - \sin\theta\,\bar{e}_\theta$$

The relation between $\bar{e}_N$, $\bar{k}$, and the inclination, Φ, is shown in Figure C.2.

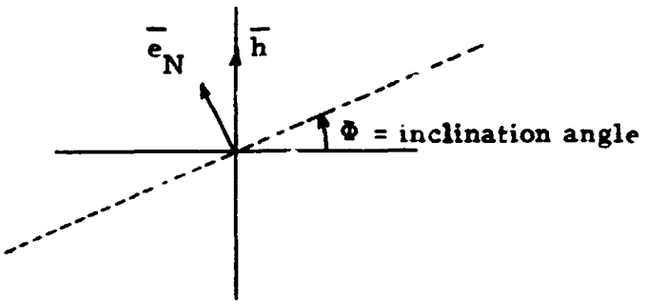


Figure C.2. Orbit Inclination Geometry.

Thus,

$$\cos\Phi = \bar{e}_N \cdot \bar{k} = \frac{(v_\phi + r\,\Omega \sin\theta)\sin\theta}{\sqrt{v_\theta^2 + (v_\phi + r\,\Omega \sin\theta)^2}},$$

or in state notation

$$\text{Cos } \Phi = \frac{\left[x_5 + (x_1+R_o)\,\Omega \text{ Sin } x_2 \right] \text{ Sin } x_2}{\left[x_4^2 + \left[x_5 + (x_1+R_o)\,\Omega \text{ Sin } x_2 \right]^2 \right]^{\frac{1}{2}}} \qquad (C.4)$$

Denote the transformation equations (C.3) by

$$\tilde{x} = g(x). \qquad (C.5)$$

The partial derivatives of g with respect to the x_i are defined by

i) $\tilde{x}_1 = g_1 = x_1$

$$\frac{\partial g_1}{\partial x_1} = 1 \qquad \frac{\partial g_1}{\partial x_i} = 0 \quad i = 2, 6$$

ii) $\tilde{x}_2 = g_2 = x_3$

$$\frac{\partial g_2}{\partial x_i} = 0 \quad i = 1, 2, 4, 5, 6 \qquad \frac{\partial g_2}{\partial x_3} = 1$$

iii) $\tilde{x}_3 = g_3 = \sqrt{x_4^2 + \left[x_5 + (x_1+R_e)\Omega \text{ Sin } x_2\right]^2}$

Let $x_5 + (x_1+R_o)\Omega \text{ Sin } x_2 = (\quad)$

$\left[x_4^2 + (\quad)^2 \right] = \left[\quad \right]$

Thus

$$\frac{\partial g_3}{\partial x_1} = \frac{1}{2} \left[\quad \right]^{\frac{-1}{2}} 2 (\quad) \Omega \text{ Sin } x_2$$

$$\frac{\partial g_3}{\partial x_2} = \frac{1}{2} \left[\quad \right]^{\frac{-1}{2}} 2 (\quad)(x_1 + R_o) \text{ Cos } x_2$$

$$\frac{\partial g_3}{\partial x_3} = 0$$

$$\frac{\partial g_3}{\partial x_4} = \frac{1}{2}[\qquad\qquad] - \frac{1}{2} \quad 2x_4$$

$$\frac{\partial g_3}{\partial x_5} = \frac{1}{2}[\qquad\qquad]\frac{1}{2} \quad 2 \; (\qquad\qquad)$$

Appendix D

User's Guide for PRAXIS

PRAXIS determines the local minimum of a scalar function which need not be differentiable. Double precision is necessary for all floating point variables. An EXTERNAL statement for the function to be minimized is needed in the program which calls PRAXIS. However, the gradient of the function is not required.

Usage of PRAXIS

CALL PRAXIS (TO, HO, N, IPRIN, X, F, FMIN)

Description of parameters:

F : Function to be minimized

MACHUP: A machine precision parameter furnished in the program; it is about 2.22×10^{-16} on the IBM 360.

TO: A tolerence for the stopping criterion; the program stops searching for the minimum if

$$|| x^{i} - x^{i+1} || \leq \text{MACHUP} \; || x^{i+1} || + \text{TO}$$

HO: Maximum step-size. To assure fast convergence, HO should be about the maximum distance from the initial guess to the minimum.

N: The number of dependent variables, i.e., the dimension of x (N should not be less than two).

IPRIN: An integer for controlling the printing of numerical results.

IPRIN = 0. Nothing is printed by PRAXIS.

IPRIN = 1. Value of F is printed after every N + 1 or N 2 linear minimizations. Final x is printed. If N $\leq$ 4, intermediate x is printed also

IPRIN = 2. The scale factors and the principal values of the approximating quadratic form are also printed.

IPRIN = 3. The values of x after every few linear minimizations are printed also.

IPRIN = 4. All available and relevent values are printed.

X : An N dimensional vector. Initial guess of minimum is placed here to start the program. Final estimate of X is returned to here.

F(X,N) : A REAL * 8 function to minimized. A declared EXTERNAL is necessary in the calling program.

FMIN : The final value of F obtained.

Output variables.

LMIN: Number of linear minimizations.

EVALS: Number of function evaluations.

MIN F: Function value at LMIN th linear minization

Example of use

```
IMPLICIT REAL * 8 (A-H, Ø -Z)
DIMENSION X (2)
EXTERNAL BANANA
TO = 1. D-5
```

```
      N = 2
      X(1) =-1.2 DO
      X (2) = 1. DO
      HO = 2.0
      IPRIN = 1
      CALL PRAXIS (TO, HO, N, IPRIN, X, BANANA, FMIN)
      PRINT FMIN
1     FORMAT (' FMIN =', D 25.15)
      END
C. . . Function to be minimized . . . . . .
      REAL FUNCTION BANANA (X, N)
      IMPLICIT REAL * 8 (A-H, ¢ -Z)
      DIMENSION X (N)
      BANANA = 100. DO * (X (2) - X (1) ** 2) ** 2 + (1. DO- X(1: ) ** 2
C . . NOTE, THERE ARE NO DERIVATIVES OF BANANA . . . . .
      RETURN
      END
```

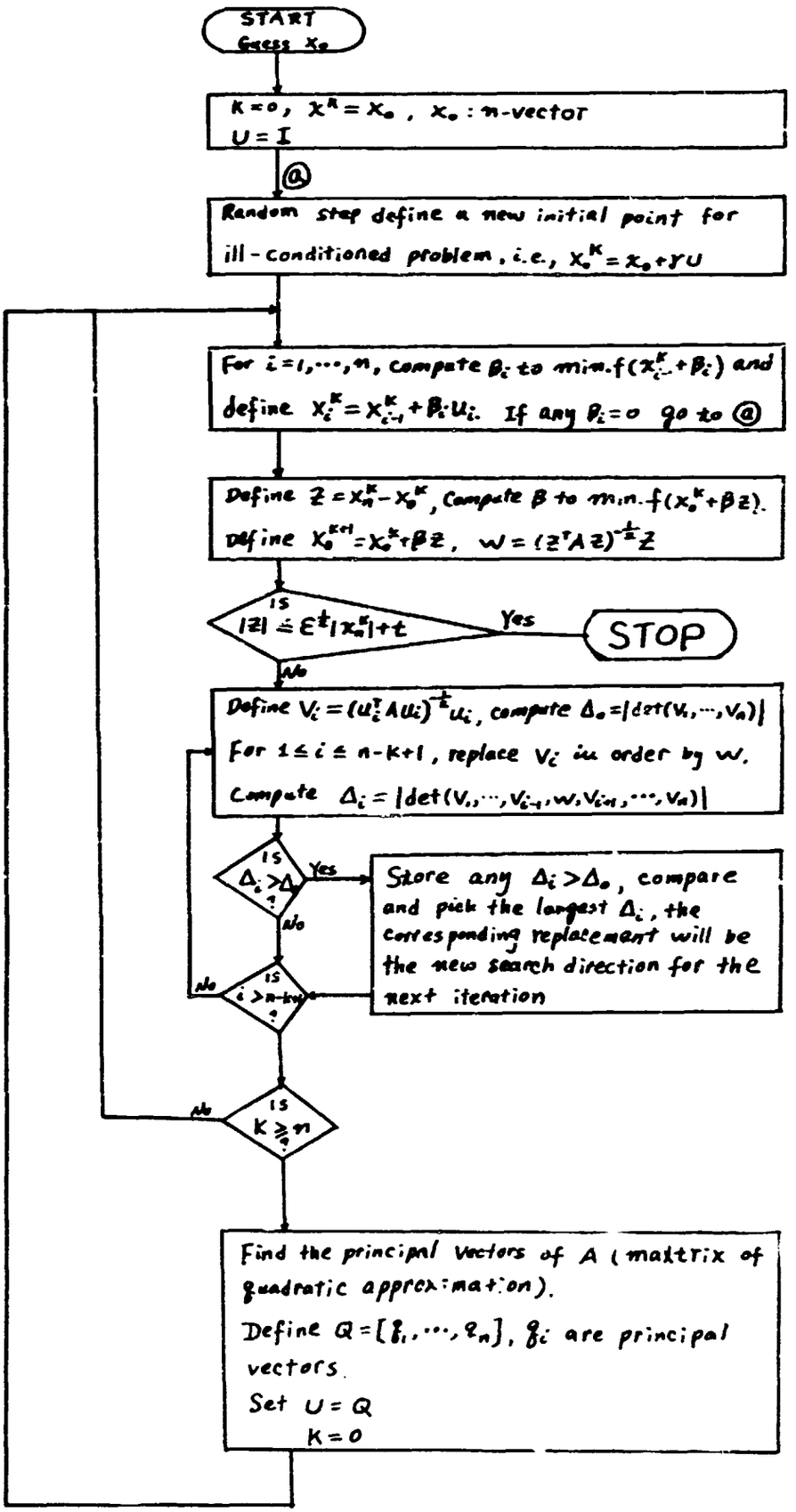


Figure D.1. Flow Diagram of PRAXIS.