Office of Naval Research

Contract N00014-67-A-0298-0006 NR-372-012

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
Grant NGR 22-007-068

A DISCRETE-TIME DIFFERENTIAL DYNAMIC PROGRAMMING ALGORITHM WITH APPLICATION TO OPTIMAL ORBIT TRANSFER

By

Stanley B. Gershwin & David H. Jacobson

August 1968

Technical Report No. 566

Division of Engineering and Applied Physics
Harvard University • Cambridge, Massachusetts

Office of Naval Research

Contract N00014-67-A-0298-0006

NR - 372 - 012

National Aeronautics and Space Administration

Grant NGR 22-007-068

A DISCRETE-TIME DIFFERENTIAL DYNAMIC PROGRAMMING ALGORITHM WITH APPLICATION TO OPTIMAL ORBIT TRANSFER

By

Stanley B. Gershwin and David H. Jacobson

Technical Report No. 566

August 1968

The research reported in this document was made possible through support extended the Division of Engineering and Applied Physics, Harvard University by the U. S. Army Research Office, the U. S. Air Force Office of Scientific Research and the U. S. Office of Naval Research under the Joint Services Electronics Program by Contracts N00014-67-A-0298-0006, 0005, and 0008 and by the National Aeronautics and Space Administration under Grant NGR 22-007-068.

Division of Engineering and Applied Physics

Harvard University Cambridge, Massachusetts

Acknowledgements

The authors wish to thank Professor A. E. Bryson, Jr. for his valuable suggestions, and Dr. R. G. Tobey of IBM for the use of the FORMAC system, which was helpful in manipulation of the rather complicated algebraic expressions that appear in this paper.

S. B. Gershwin wishes to thank D. H. Jacobson for his patience and accessability during the preparation of this report.

Page Intentionally Left Blank

A DISCRETE-TIME DIFFERENTIAL DYNAMIC PROGRAMMING ALGORITHM WITH APPLICATION TO OPTIMAL ORBIT TRANSFER

By

Stanley B. Gershwin and David H. Jacobson

Division of Engineering and Applied Physics

Harvard University Cambridge, Massachusetts

ABSTRACT

Recently, the notion of Differential Dynamic Programming has been used to obtain new second-order algorithms for solving non-linear optimal control problems. (Unlike conventional Dynamic Programming, the Principle of Optimality is applied in the neighborhood of a nominal, non-optimal, trajectory.) A novel feature of these algorithms is that they permit strong variations in the system trajectory.

In this paper, Differential Dynamic Programming is used to develop a second-order algorithm for solving discrete-time dynamic optimization problems with terminal constraints. This algorithm also utilizes strong variations and, as a result, has certain advantages over existing discrete-time methods.

A non-linear computed example is presented, and comparisons are made with the results of other researchers who have solved this problem.

The experience gained during the computation has suggested some extensions to an earlier, previously published Differential Dynamic Programming algorithm for continuous time problems. These extensions, and their implications are discussed.

Page Intentionally Left Blank

Notation

Vectors are columns; the scalar product of a and b, where

$$a = \begin{bmatrix} a_1 \\ \vdots \\ a_n \end{bmatrix} \qquad b = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

is $a^T b$ or $b^T a$ and is equal to $\sum_{i=1}^{n} a_i b_i$. The derivative of a scalar by a vector is a row, and is written:

$$V_x = \frac{\partial V}{\partial x} = [\frac{\partial V}{\partial x_1}, \ldots, \frac{\partial V}{\partial x_n}] \quad .$$

The second derivative of a scalar by vectors is a matrix:

$$V_{xk} = \frac{\partial^2 V}{\partial x \partial k} = \begin{bmatrix} \frac{\partial^2 V}{\partial x_1 \partial k_1} & \cdots & \frac{\partial^2 V}{\partial x_1 \partial k_m} \\ \vdots & & \vdots \\ \frac{\partial^2 V}{\partial x_n \partial k_1} & \cdots & \frac{\partial^2 V}{\partial x_n \partial k_m} \end{bmatrix}$$

where x is an n-vector and k is an m-vector.

Thus a second-order Taylor expansion will be written:

$$V(x + \delta x, k + \delta k) = V(x, k) + V_x \delta x + V_k \delta k + \frac{1}{2} \delta x^T V_{xx} \delta x + \delta x^T V_{xk} \delta k + \frac{1}{2} \delta k^T V_{kk} \delta k \quad .$$

I. Introduction

Jacobson [1], [2] has derived a second-order algorithm for solving continuous time optimal control problems using Differential Dynamic Programming. This algorithm differs from other second-order or second-variation algorithms, [4], [5], [6], [7], [9], [10], [11], [14] in that it is derived using global variations in control (strong variations in the trajectory).

In this paper a similar algorithm is developed for solving discrete-time dynamic optimization problems with terminal constraints. The new algorithm uses the notion of strong variations and hence, as in the case of the continuous time algorithm, has advantages over existing discrete-time algorithms [4], [5], [9], [14]. The algorithm can be used to solve continuous time problems that are approximated by difference equations.

A non-linear numerical example is presented and comparisons are drawn with McReynolds [4], [5] and others [7], [8], who have solved this problem previously, using other methods. The experience gained in the numerical computation has suggested extensions to the continuous algorithms in [1] and [2]. In particular, the 'step-size adjustment' technique is generalized by the introduction of additional criteria for ensuring that the 'trial new trajectory', at each iteration, is sufficiently close to the current nominal trajectory to guarantee an improvement in cost and/or terminal error.

II. Derivation of the Discrete Algorithm

II. 1. Statement of the General Problem

The problem to be solved is the following: if $x_o, \ldots, x_N$ are vector quantities which satisfy

(1) $$x_{i+1} = f(x_i, u_i, t_i)$$

and x_o is given, find the vectors $u_o, \ldots, u_{N-1}$ to minimize the scalar

(2) $$\hat{V} = \sum_{i=0}^{N-1} L(x_i, u_i, t_i) + F(x_N) \quad ,$$

where the solution must satisfy the (vector) equality constraint

(3) $$\theta(x_N) = 0$$

N and $t_o, \ldots, t_N$ are known quantities, and a nominal control $\bar{u}_o, \ldots, \bar{u}_{N-1}$ is given.

Defining

(4) $$V(x_o, k, t_o) = \hat{V} + k^T\theta \quad ,$$

the equivalent problem of finding $u_o, \ldots, u_{N-1}$ to minimize $V(x_o, k_o, t_o)$ † and k to satisfy (3) is solved in succeeding sections. A nominal value of k, $\bar{k}$, is assumed given.

II. 2. Outline of the Solution

The optimal return function V satisfies Bellman's "Principle of Optimality" [3], which in this case is:

(5) $$V(x_i, k, t_i) = \min_{u_i} [L(x_i, u_i, t_i) + V(x_{i+1}, k, t_{i+1})]$$

for $i = 0, \ldots, N-1$.

Regarded in terms of displacements δx_i, δx_{i+1}, and δk from the nominal trajectory,

† It is assumed that a minimum exists.

$$x_i = \bar{x}_i + \delta x_i$$

$$x_{i+1} = \bar{x}_{i+1} + \delta x_{i+1}$$

$$k = \bar{k} + \delta k$$

and (5) becomes

$$V(\bar{x}_i + \delta x_i, \bar{k} + \delta k, t_i) = \min_{u_i} [L(\bar{x}_i + \delta x_i, u_i, t_i) + V(\bar{x}_{i+1} + \delta x_{i+1}, \bar{k} + \delta k, t_{i+1})] \tag{6}$$

The algorithm is derived from equation (6) in the following sequence of steps:

1. Expand both sides in Taylor series about $\bar{x}_i$, $\bar{k}$ and $\bar{x}_{i+1}$ in δx_i, δk and δx_{i+1}.
2. Relate δx_{i+1} to δx_i.
3. Perform the indicated minimization with respect to u_i in two stages.
   A. Find u_i^* which minimizes the right side of (6) with $\delta x_i = 0$ and $\delta k = 0$.
   B. Expand about u_i^* in δu_i with δx_i and δk non-zero, and minimize with respect to δu_i. This will give δu_i as a function of δx_i and δk.
4. Equate coefficients of like powers of δx_i and δk to obtain difference equations in V_x^i, V_k^i, etc.

It is assumed that δx_i, δx_{i+1} and δk will be sufficiently small that all Taylor expansions can be terminated at second-order terms.

II. 3. Solution

Following the prescription of the previous section, the left side of (6), when expanded in a Taylor series, is,

$$(7)\qquad V(\bar{x}_i + \delta x_i, \bar{k} + \delta k, t_i) = V(\bar{x}_i, \bar{k}, t_i) + \frac{\partial}{\partial x} V(\bar{x}_i, \bar{k}, t_i)\delta x_i + \frac{\partial}{\partial k} V(\bar{x}_i, \bar{k}, t_i)\delta k$$

$$+ \frac{1}{2} \delta x_i^T \frac{\partial^2}{\partial x^2} V(\bar{x}_i, \bar{k}, t_i)\delta x_i$$

$$+ \delta x_i^T \frac{\partial^2}{\partial x \partial k} V(\bar{x}_i, \bar{k}, t_i)\delta k$$

$$+ \frac{1}{2} \delta k^T \frac{\partial^2}{\partial k^2} V(\bar{x}_i, \bar{k}, t_i)\delta k + \ldots$$

The reader should note that $V(\bar{x}_i, \bar{k}, t_i)$ is the minimal value of the return function obtainable with initial conditions at $\bar{x}_i$, t_i, and with $k = \bar{k}$. It is not the same as $\bar{V}(\bar{x}_i, \bar{k}, t_i)$, the value of the return function calculated along the nominal trajectory, starting from t_i.

Symbolically,

$$(8)\qquad V(\bar{x}_i, \bar{k}, t_i) = \min_{u_i, \ldots, u_{N-1}} \left[\sum_{j=i}^{N-1} L(x_j, u_j, t_j) + F(x_N) + \bar{k}^T \theta(x_N) \right]$$

where $x_{i+1}, \ldots, x_N$ satisfy (1), and $x_i = \bar{x}_i$.

However,

$$(9)\qquad \bar{V}(\bar{x}_i, \bar{k}, t_i) = \sum_{j=i}^{N-1} L(\bar{x}_j, \bar{u}_j, t_j) + F(\bar{x}_N) + \bar{k}^T \theta(\bar{x}_N)$$

where $\bar{u}_i, \ldots, \bar{u}_{N-1}$ is the nominal control sequence and thus, $\bar{x}_i, \ldots, \bar{x}_N$ is the nominal trajectory (which satisfies (1) with $u_j = \bar{u}_j$, $j = i, \ldots, N-1$).

Acknowledging the difference between $V(\bar{x}_i, \bar{k}, t_i)$ and $\bar{V}(\bar{x}_i, \bar{k}, t_i)$, define

$$(10)\qquad a(\bar{x}_i, \bar{k}, t_i) = V(\bar{x}_i, \bar{k}, t_i) - \bar{V}(\bar{x}_i, \bar{k}, t_i)$$

To simplify notation, let

$$\overline{V}(\overline{x}_i, \overline{k}, t_i) = \overline{V}^i$$

$$V(\overline{x}_i, \overline{k}, t_i) = V^i$$

$$a(\overline{x}_i, \overline{k}, t_i) = a^i$$

$$\frac{\partial}{\partial x} V(\overline{x}_i, \overline{k}, t_i) = V^i_x \quad , \qquad \text{etc.}$$

Then

(10') $$a^i = V^i - \overline{V}^i$$

and applying (10) to (7), obtain

(11) $$V(\overline{x}_i + \delta x_i, \overline{k} + \delta k, t_i) = a^i + \overline{V}^i + V^i_x \delta x_i + V^i_k \delta k + \frac{1}{2} \delta x_i^T V^i_{xx} \delta x_i + \delta x_i^T V^i_{xk} \delta k + \frac{1}{2} \delta k^T V^i_{kk} \delta k + \dots$$

Similarly, expanding the quantity to be minimized in equation (6) about $\overline{x}_i$, $\overline{k}$, $\overline{x}_{i+1}$,†

(12) $$L^i + L^i_x \delta x_i + \frac{1}{2} \delta x_i L^i_{xx} \delta x_i + a^{i+1} + \overline{V}^{i+1} + V^{i+1}_x \delta x_{i+1} + V^{i+1}_k \delta k + \frac{1}{2} \delta x^T_{i+1} V^{i+1}_{xx} \delta x_{i+1} + \delta x^T_{i+1} V^{i+1}_{xk} \delta k + \delta k^T V^{i+1}_{kk} \delta k + \dots$$

where, as above, $a^{i+1} + \overline{V}^{i+1} = V^{i+1}$.

Expression (12) is an infinite series in δx_i, δx_{i+1} and δk. But it is clear that there is a relationship between δx_i and δx_{i+1} through equation (1). This relationship may be used to eliminate either δx_i or δx_{i+1} from (12), but to conform with equation (11), δx_{i+1} will be removed.

$$x_{i+1} = f(x_i, u_i, t_i)$$
$$\overline{x}_{i+1} = f(\overline{x}_i, \overline{u}_i, t_i)$$

† L and its derivatives are evaluated at $\overline{x}_i, u_i, t_i$. The control u_i is yet to be determined.

Thus, $\delta x_{i+1} = f(x_i, u_i, t_i) - f(\overline{x}_i, \overline{u}_i, t_i)$ or,

(13) $$\delta x_{i+1} = f(\overline{x}_i + \delta x_i, u_i, t_i) - f(\overline{x}_i, \overline{u}_i, t_i)$$

In equation (13), u_i is perfectly general. It will later be fixed by the minimization operation of equation (6).

Expanding (13) about $\overline{x}_i$, and defining

$$f^i = f(\overline{x}_i, u_i, t_i)$$

$$\overline{f}^i = f(\overline{x}_i, \overline{u}_i, t_i) \quad ,$$

obtain

(14) $$\delta x_{i+1} = (f^i - \overline{f}^i) + f^i_x \delta x_i + \frac{1}{2} \delta x_i^T f^i_{xx} \delta x_i + \ldots$$

where the derivatives of f^i are evaluated at $(\overline{x}_i, u_i, t_i)$.

Substituting (14) into (12), obtain

(15) $$L^i + a^{i+1} + \overline{V}^{i+1} + V^{i+1}_x (f^i - \overline{f}^i) + \frac{1}{2}(f^i - \overline{f}^i)^T V^{i+1}_{xx}(f^i - \overline{f}^i)$$
$$+ [L^i_x + V^{i+1}_x f^i_x + f^{i\,T}_x V^{i+1}_{xx}(f^i - \overline{f}^i)]\delta x_i$$
$$+ [V^{i+1}_k + (f^i - \overline{f}^i)^T V^{i+1}_{xk}]\delta k$$
$$+ \delta x_i^T f^i_x V^{i+1}_{xk} \delta k$$
$$+ \frac{1}{2} \delta k^T V^{i+1}_{kk} \delta k$$
$$+ \frac{1}{2} \delta x_i^T [L^i_{xx} + V^{i+1}_x f^i_{xx} + f^{i\,T}_x V^{i+1}_{xx} f^i_x + (f^i - \overline{f}^i)^T V^{i+1}_{xx} f^i_{xx}]\delta x_i + \ldots$$

Recall that equation (5) has now been transformed to

(16) "r. h. s. of equation (11) = $\min_{u_i}$ {expression (15)}"

As suggested earlier, the minimization in (16) may be performed in two stages.

First u_i^* is found, which minimizes (15) with $\delta x_i = 0$ and $\delta k = 0$, i.e., u_i^* minimizes

$$L^i + a^{i+1} + \overline{V}^{i+1} + V_x^{i+1}(f^i - \overline{f}^i) + \frac{1}{2}(f^i - \overline{f}^i)^T V_{xx}^{i+1}(f^i - \overline{f}^i) + \ldots \tag{17}$$

(The terms not printed in (17) are of third and higher order in $(f^i - \overline{f}^i)$, and thus are assumed negligible.)

For convenience, define

$$H^i = H(\overline{x}_i, u_i^*, \overline{k}, t_i) = L^i + V_x^{i+1} f^i \tag{18}$$

In (18), and for the rest of this paper, all functions of u_i are evaluated at u_i^*.

Note that

$$H_x^i = L_x^i + V_x^{i+1} f_x^i$$

$$H_{xx}^i = L_{xx}^i + V_x^{i+1} f_{xx}^i \quad , \quad \text{etc.}$$

Since (17) is at a minimum when evaluated at u_i^*, its first derivative with respect to u_i must be zero;

$$H_u^i + (f^i - \overline{f}^i)^T V_{xx}^{i+1} f_u^i = 0 \tag{19}$$

In addition, the second derivative of (17) (to be defined as Δ) must be positive definite at $u_i = u_i^*$;

$$\Delta = H_{uu}^i + f_u^{i\,T} V_{xx}^{i+1} f_u^i + (f^i - \overline{f}^i)^T V_{xx}^{i+1} f_{uu}^i > 0 \tag{20}$$

(The third term in (20) does not appear in the 'weak variation' algorithms of [4], [5], [9], [14]).

Expanding (15) about u_i^*, with $u_i = u_i^* + \delta u_i$, the following is obtained, using (19) and (20).

$$
\begin{aligned}
(21)\qquad & L^i + a^{i+1} + \overline{V}^{i+1} + V_x^{i+1}(f^i - \overline{f}^i) + \frac{1}{2}(f^i - \overline{f}^i)^T V_{xx}^{i+1}(f^i - \overline{f}^i) \\
& + [H_x^i + f_x^{i\,T} V_{xx}^{i+1}(f^i - \overline{f}^i)]\delta x_i \\
& + [V_k^{i+1} + (f^i - \overline{f}^i)^T V_{xk}^{i+1}]\delta k \\
& + \delta x_i^T f_x^{i\,T} V_{xk}^{i+1}\delta k \\
& + \delta u_i^T f_u^{i\,T} V_{xk}^{i+1}\delta k \\
& + \delta x_i^T [H_{xu}^i + f_x^{i\,T} V_{xx}^{i+1} f_u^i + (f^i - \overline{f}^i)^T V_{xx}^{i+1} f_{xu}^i]\delta u_i \\
& + \frac{1}{2}\delta x_i^T [H_{xx}^i + f_x^{i\,T} V_{xx}^{i+1} f_x^i + (f^i - \overline{f}^i)^T V_{xx}^{i+1} f_{xx}^i]\delta x_i \\
& + \frac{1}{2}\delta k^T V_{kk}^{i+1}\delta k \\
& + \frac{1}{2}\delta u_i^T \Delta \delta u_i
\end{aligned}
$$

Terms of order $(\delta x_i)^3$, $(\delta u_i)^3$, $(\delta k)^3$ or greater have been ignored in (21).†

The second stage of the minimization is accomplished when (21) is minimized with respect to δu_i.

Taking the first derivative of (21) with respect to δu_i and setting it to zero, obtain

† It is assumed that δx_i, δu_i and δk are small enough to justify this truncation.

(22) $$\delta u_i = \beta_1 \delta x_i + \beta_2 \delta k$$

where

(23) $$\beta_1 = -\Delta^{-1}[H^i_{ux} + f^{i\,T}_u V^{i+1}_{xx} f^i_x + (f^i - \overline{f}^i)^T V^{i+1}_{xx} f^i_{ux}]$$

(24) $$\beta_2 = -\Delta^{-1} f^{i\,T}_u V^{i+1}_{xk}$$

Equation (22) is a linear feedback perturbation control law. It is sufficient to consider δu_i to be linear in δx_i and δk because on substituting an expression of higher order than (22) into (21), terms of higher order than quadratic would appear.

On substituting (22) into (21), the result is

(25) $$\begin{aligned} & L^i + a^{i+1} + \overline{V}^{i+1} + V^{i+1}_x (f^i - \overline{f}^i) + \frac{1}{2}(f^i - \overline{f}^i)^T V^{i+1}_{xx} (f^i - \overline{f}^i) \\ & + [H^i_x + (f^i - \overline{f}^i)^T V^{i+1}_{xx} f^i_x] \delta x_i \\ & + [V^{i+1}_k + (f^i - \overline{f}^i)^T V^{i+1}_{xk}] \delta k \\ & + \delta x_i^T [f^{i\,T}_x V^{i+1}_{xk} - \beta_1^T \Delta \beta_2] \delta k \\ & + \frac{1}{2} \delta x_i^T [H^i_{xx} + f^{i\,T}_x V^{i+1}_{xx} f^i_x + (f^i - \overline{f}^i) V^{i+1}_{xx} f^i_{xx} - \beta_1^T \Delta \beta_1] \delta x_i \\ & + \frac{1}{2} \delta k^T [V^{i+1}_{kk} - \beta_2^T \Delta \beta_2] \delta k \end{aligned}$$

Expression (25) is the minimum of (15) with respect to u_i. Thus, expression (25) is equal to the r.h.s. of equation (11), by (16). Therefore, coefficients of like powers of δx_i and δk must be equal.

Noting that

(26) $$\overline{V}^i = \overline{V}^{i+1} + \overline{L}^i ,$$

equating (11) and (25) produces the following difference equations, valid for i = 0, . . . , N-1.

(27) $$a^i = a^{i+1} + H^i - \overline{H}^i + \frac{1}{2}(f^i - \overline{f}^i)^T V_{xx}^{i+1}(f^i - \overline{f}^i)$$

(28) $$V_x^i = H_x^i + (f^i - \overline{f}^i)^T V_{xx}^{i+1} f_x^i$$

(29) $$V_k^i = V_k^{i+1} + (f^i - \overline{f}^i)^T V_{xk}^{i+1}$$

(30) $$V_{xk}^i = f_x^{i\,T} V_{xk}^{i+1} - \beta_1^T \Delta \beta_2$$

(31) $$V_{kk}^i = V_{kk}^{i+1} - \beta_2^T \Delta \beta_2$$

(32) $$V_{xx}^i = H_{xx}^i + f_x^{i\,T} V_{xx}^{i+1} f_x^i + (f^i - \overline{f}^i) V_{xx}^{i+1} f_{xx}^i - \beta_1^T \Delta \beta_1$$

The boundary conditions are applied at i = N, and are the same as in [1]. They are found by expanding

$$V(\overline{x}_N + \delta x_N, \overline{k} + \delta k, t_N) = F(\overline{x}_N + \delta x_N) + (\overline{k} + \delta k)^T \theta(\overline{x}_N + \delta x_N)$$

to second-order in a Taylor series in δx_N and δk. Because this is the last time step, $\overline{V}^N = V^N$. Thus,

(33) $$a^N = 0$$

and, from the expansion,

(34) $$V_x^N = F_x(\overline{x}_N) + \overline{k}^T \theta_x(\overline{x}_N)$$

(35) $$V_k^N = \theta^T(\overline{x}_N)$$

(36) $$V_{xk}^N = \theta_x^T(\overline{x}_N)$$

(37) $$V_{kk}^N = 0$$

(38) $$V^N_{xx} = F_{xx}(\bar{x}_N) + \bar{k}^T \theta_{xx}(\bar{x}_N)$$

Thus, if we "integrate" equations (27)-(32) from i = N-1 to 0 with equations (33)-(38) as boundary conditions, then equations (19) and (22) show how to calculate $u_i = u_i^* + \delta u_i$ to get optimal improvement on performance index $V(x_o, k, t_o)$.

These results are only meaningful if the second-order truncations of the Taylor series above are good approximations of the full expansions. Thus δx_i, δx_{i+1}, δk, and δu_i must be small. There is no restriction on $\Delta u_i = u_i^* - \bar{u}_i$ except that $f^i - \bar{f}^i = f(\bar{x}_i, u_i^*, t_i) - f(\bar{x}_i, \bar{u}_i, t_i)$ must be sufficiently small to guarantee the smallness of δx_{i+1}.

III. Comparison with and Extensions of Jacobson's Results

III. 1. Comparison and Discussion

The case in which the discrete problem is an Euler discretization of a continuous problem is of interest. In that case,

(39) $$f(x_i, u_i, t_i) = x_i + \Delta t \tilde{f}(x_i, u_i, t_i)$$

and

(40) $$L(x_i, u_i, t_i) = \tilde{L}(x_i, u_i, t_i) \Delta t$$

Clearly,

(41) $$\dot{x}(t_i) = \lim_{\Delta t \to 0} \frac{x(t_i + \Delta t) - x(t_i)}{\Delta t} = \lim_{\Delta t \to 0} \frac{x_{i+1} - x_i}{\Delta t} = f(x_i, u_i, t_i)$$

and

(42) $$\lim_{\substack{\Delta t \to 0 \\ N \to \infty}} \sum_{i=0}^{N-1} L(x_i, u_i, t_i) = \lim_{\substack{\Delta t \to 0 \\ N \to \infty}} \sum_{i=0}^{N-1} \tilde{L}(x_i, u_i, t_i) \Delta t = \int_{t_o}^{t_N} \tilde{L}(x(t), u(t), t) dt$$

if the discretization is done with care.

It is reasonable to expect that if the transformations (39) and (40) are applied to the results of the previous section and the limit is taken as $\Delta t \to 0$, equations should be obtained which solve the analogous continuous problem.

Jacobson [1] has solved that problem, and the statement of the problem, as well as the solution are reproduced below, in Appendix A.

Note that

$$H^i = \tilde{L}^i \Delta t + V_x^{i+1}(\bar{x}_i + \Delta t \tilde{f}^i)$$

where the same abbreviated notation as in the last section is used.

Thus

$$(45) \qquad H^i = (\tilde{L}^i + V_x^{i+1}\tilde{f}^i)\Delta t + V_x^{i+1}\bar{x}_i = \tilde{H}^i \Delta t + V_x^{i+1}\bar{x}_i$$

Then, according to (20)

$$(44) \qquad \Delta = \tilde{H}^i_{uu}\Delta t + (\Delta t)^2[\tilde{f}^i_u V^{i+1}_{xx}\tilde{f}^i_u + (\tilde{f}^i - \bar{f}^i)V^{i+1}_{xx}\tilde{f}^i_{uu}]$$

Define

$$(45) \qquad \tilde{\Delta} = \Delta/\Delta t \quad ,$$

which will be written

$$(46) \qquad \tilde{\Delta} = \tilde{H}^i_{uu} + A^i \Delta t$$

for clarity.

From (23) and (45),

$$(47) \qquad \beta_1 = -\tilde{\Delta}^{-1}(\tilde{H}^i_{ux} + \tilde{f}^{i\,T}_u V^{i+1}_{xx}) - \tilde{\Delta}^{-1}(\tilde{f}^{i\,T}_u V^{i+1}_{xx}\tilde{f}^{i+1}_x + (\tilde{f}^i - \bar{f}^i)^T V^{i+1}_{xx} f^i_{ux})\Delta t$$

Similarly, from (24),

$$(48) \qquad \beta_2 = -\tilde{\Delta}^{-1}\tilde{f}^{i\,T}_u V^{i+1}_{xk}$$

In the same manner, applying (39), (40), (43), (45), (47), and (48) to (27)-(32), the following are simply obtained.

(49) $$-\frac{a^{i+1} - a^i}{\Delta t} = \tilde{H}^i - \overline{\tilde{H}}^i + \frac{1}{2}(\tilde{f}^i - \overline{\tilde{f}}^i)^T V_{xx}^{i+1}(\tilde{f}^i - \overline{\tilde{f}}^i)\Delta t$$

(50) $$-\frac{V_x^{i+1} - V_x^i}{\Delta t} = \tilde{H}_x^i + (\tilde{f}^i - \overline{\tilde{f}}^i)V_{xx}^{i+1} + (\tilde{f}^i - \overline{\tilde{f}}^i)^T V_{xx}^{i+1}\tilde{f}_x^i \Delta t$$

(51) $$-\frac{V_k^{i+1} - V_k^i}{\Delta t} = (\tilde{f}^i - \overline{\tilde{f}}^i)^T V_{xk}^{i+1}$$

(52) $$-\frac{V_{xk}^{i+1} - V_{xk}^i}{\Delta t} = \tilde{f}_x^{i\,T} V_{xk}^{i+1} - \beta_1^T \tilde{\Delta}\beta_2$$

(53) $$-\frac{V_{kk}^{i+1} - V_{kk}^i}{\Delta t} = -\beta_2^T \tilde{\Delta}\beta_2$$

(54) $$-\frac{V_{xx}^{i+1} - V_{xx}^i}{\Delta t} = \tilde{H}_{xx}^i + \tilde{f}_x^{i\,T} V_{xx}^{i+1} + V_{xx}^{i+1}\tilde{f}_x^i + \beta_1^T \tilde{\Delta}\beta_1$$
$$+ \Delta t[\tilde{f}_x^{i\,T} V_{xx}^{i+1}\tilde{f}_x^i + (\tilde{f}^i - \overline{\tilde{f}}^i)^T V_{xx}^{i+1}\tilde{f}_{xx}^i]$$

Jacobson's [1] equations for β_1, β_2, a, V_x, V_k, V_{xk}, V_{kk}, and V_{xx} are reproduced below in Appendix A. Inspection will reveal agreement between those and (47)-(54) as $\Delta t \to 0$.

It should be noted that although the discrete f, L, and H are related to their respective continuous counterparts through (39), (40), and (43), the discrete a, $\overline{V}$, derivatives of V, β_1, and β_2 directly approximate the continuous quantities. As $\Delta t \to 0$, the discrete and continuous versions of the latter quantities approach one another.

Equations (39) and (40) and the transformations that resulted from them were used to show the connection between the present discrete equations and the earlier [1] continuous equations. However, cases may exist where (39) and (40) are useful numerical methods with which to solve a continuous problem.† Then, (47)-(54) contain

† Continuous-time problems which are particularly sensitive to u may require a large number of small time steps when the algorithms of [1], [2] are used. Then, since Δt is small, sufficient integration accuracy may be obtained from an Euler scheme. See [2, page 17].

the full dependence on Δt, which involves terms of order Δt and higher. It may be worth while to retain high order terms [14].

Also, (47)-(54) indicate that some of the arguments of the right sides are to be evaluated at time i+1, and others must be evaluated at time i. A simple Euler discretization of the continuous time algorithm [1], [2] would evaluate all arguments at time i+1.

It may be possible to obtain more useful versions of (47)-(54) by replacing (39) and (40), the Euler discretizations of f and L, by a more sophisticated, accurate scheme.

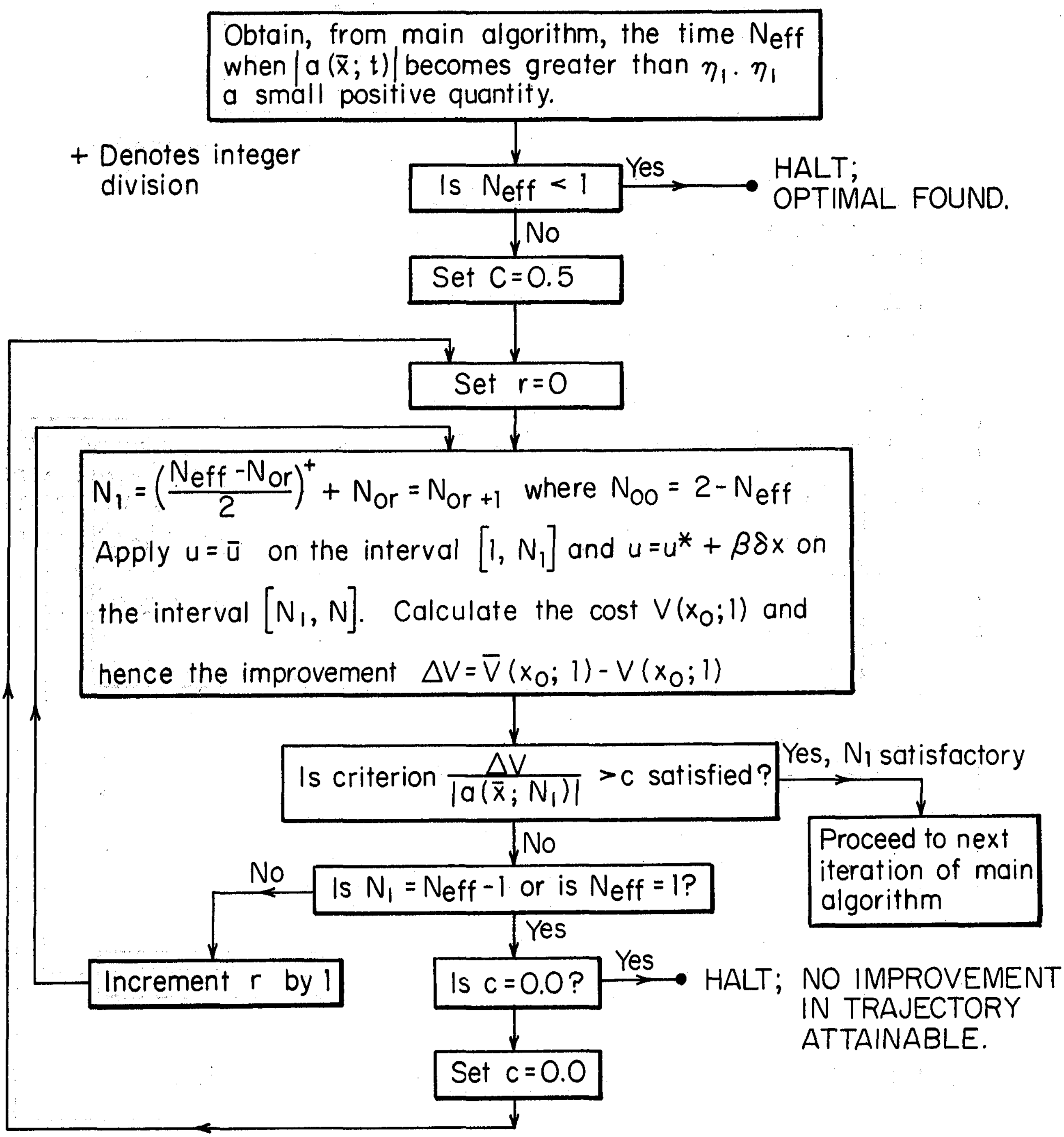
III. 2. Description of the Algorithm

The discrete algorithm is very similar to the continuous algorithm [1, section 4.8], and is outlined in Flow Chart II.

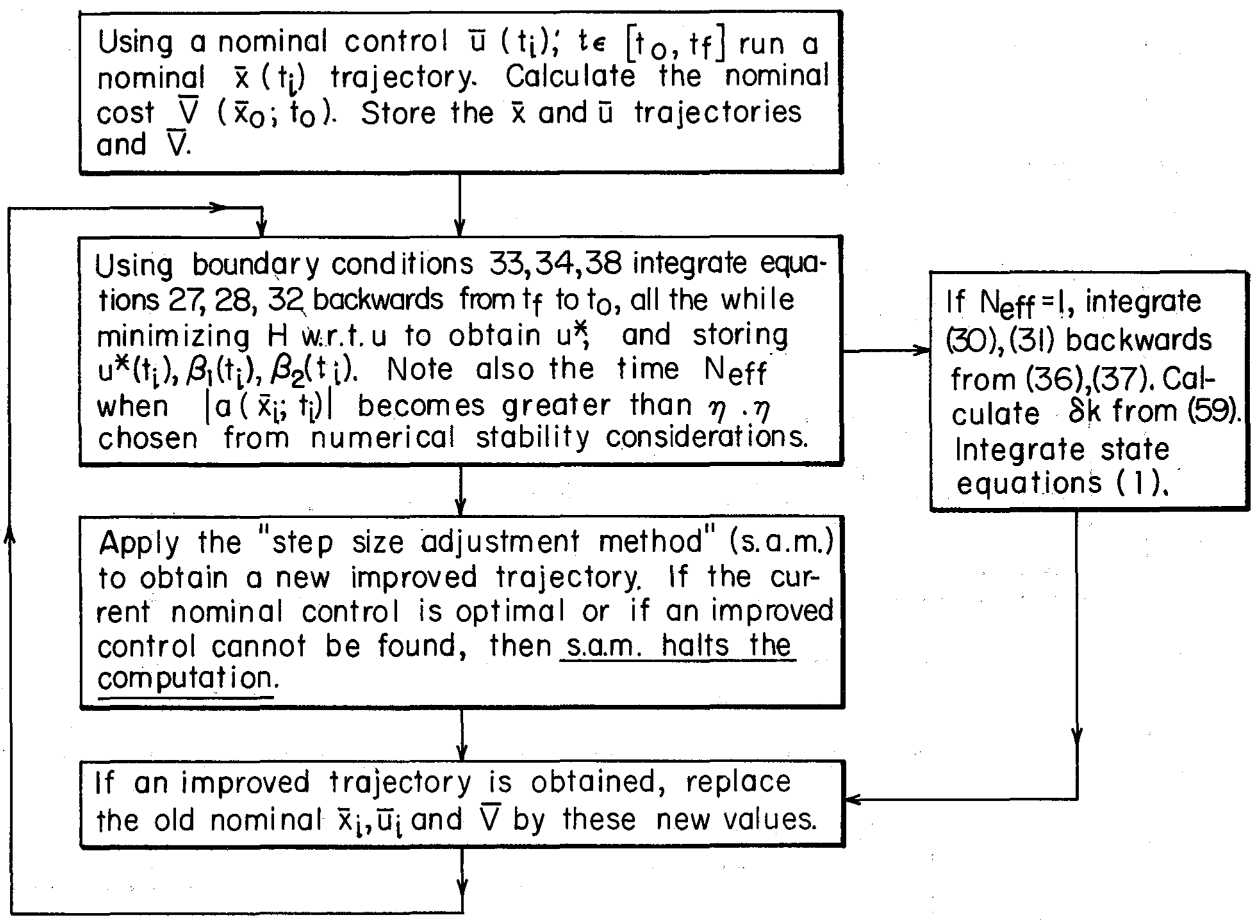
The algorithm is a successive approximation process, and each approximation has two stages. In the first stage, k is kept constant, and optimization takes place with respect to u_i, without regard to the value of θ. In the second, δk is calculated to reduce θ in absolute value.

The first stage proceeds as follows. Equation (1) is "integrated" using initial conditions x_o and nominal control $\bar{u}_o, \ldots, \bar{u}_{N-1}$. Then equations (27), (28), and (32) are integrated back from i = N, with boundary conditions (33), (34), and (38).

If a^o is not close to zero, then, by definition (10), the nominal control is not close to optimal for the current value of $\bar{k}$. To improve the trajectory (i.e., to get closer to the optimal and reduce a^o), (19) is solved for u_i^* and (22) is used to calculate $u_i = u_i^* + \delta u_i$, which is used as the new optimal control in (1). The cycle repeats. If necessary (see below for the descriptions of the tests to explain this



FLOWCHART I: "STEP SIZE ADJUSTMENT METHOD".



FLOW CHART II: THE OVERALL COMPUTATIONAL PROCEDURE necessity) the step-size adjustment routine is called. (This routine will not be discussed here, but, for completeness, it appears schematically in Flow Chart I. It is described in the references, [1], and [2, section 4].)

If a^o is close to zero, and θ is also close to zero, the problem is solved.

If a^o is close to zero but θ is not, the algorithm enters its second stage: k is modified (according to the formula of the next section) to reduce each component of θ in absolute value.

III. 3. Determination of δk

δk is found in the following manner. Jacobson has shown [1, section 4.6] that, to second-order, the proper value of δk is that which maximizes $V(\bar{x}_o, \bar{k} + \delta k, t_o)$.† But

(55) $$V(\bar{x}_o, \bar{k} + \delta k, t_o) = a^o + \bar{V}^o + V_k^{o^T} \delta k + \frac{1}{2}\delta k^T V_{kk}^o \delta k$$

Therefore the proper value of δk satisfies

(56) $$V_k^{o^T} + V_{kk}^o \delta k = 0$$

or

(57) $$\delta k^T = -V_{kk}^{o^{-1}} V_k^{o^T}$$

(Jacobson shows that V_{kk}^o is negative definite,‡ so that $V_{kk}^{o^{-1}}$ exists.)

Since, in the present algorithm, δk is only evaluated when $f^i - \bar{f}^i = 0$ (because $a^o = 0$), $V_k^o = \theta^T(\bar{x}_N)$ from equations (29) and (35). Then, (57) becomes

(58) $$\delta k = -V_{kk}^{o^{-1}} \theta(\bar{x}_N)$$

---

† McReynolds [4] and Bryson and Ho [13] have obtained similar conditions.

‡ Provided that the linearised system is controllable, and θ_x^T has full rank.

Following [1], k is modified according to (58); (1) is then integrated forward with $u_i = u_i^* + \delta u_i$ chosen according to (19) and (22). If the resultant value of $\theta(x_N)$ is not smaller in absolute value (component-wise) than $\theta(\bar{x}_N)$, choose

(59) $$\delta k = -\epsilon V_{kk}^{o^{-1}} \theta(\bar{x}_N)$$

where $0 < \epsilon < 1$, and reduce ϵ until $\theta(x_N)$ is reduced and a^o is near zero.

III. 4. New Criteria

It is essential that δx_i and δk be kept small. This ensures that δu_i will be small, and thus the second-order expansions of (6) will remain valid. If δx_i and δk are found to be too large, i.e., if they invalidate the truncations of the Taylor series in section II, means for reducing them are presented in Jacobson's algorithms [1, section 4.2.1], [1, section 4.8], [2, section 4]. These techniques apply to the discrete problem as well as to the continuous.

There are criteria in [1] and [2] for deciding whether to reduce δx_i and δk or not. However, an addition criterion, required for fixed end point problems is described below (Test 1).

A criterion, alternative to that in [1], [2] is also given. This criterion (Test 2) is useful in cases where it is desirable to keep the 'new trajectory' in the immediate neighborhood of the nominal.†

<u>Test 1</u>

Although δk is chosen according to (59) (where ϵ is such that $\theta(x_N)$ is reduced), it may lie outside the range of validity of the expansion (11) (when truncated at second-order terms).

---

† Such may be the case when the trajectory must be prevented from "jumping" to another near by local minimum. In the following section, an example is discussed in detail where this was found to be necessary.

At $i = 0$, (11) coincides with (55). Since both sides of (55) may be independently measured (i.e., choose δk and evaluate the left-hand side. Then integrate (1) as described above and evaluate the right-hand side, $V(\bar{x}_o, \bar{k} + \delta k, t_o))$, (55) may be considered to be a test of δk.

If δk is given by (59), then (55) predicts that

$$V(\bar{x}_o, \bar{k} + \delta k, t_o) - \bar{V}^o = a^o - (\epsilon - \frac{1}{2}\epsilon^2)\theta^T(\bar{x}_N) {V^o_{kk}}^{-1} \theta(\bar{x}_N) \tag{60}$$

If (60) does not predict the change in V to within a given tolerance, then ϵ should be reduced until it does.

Test 2

From (4) and (9),

$$V^i = \sum_{j=i}^{N} L^j + F(x_N) + k^T \theta(x_N) \tag{61}$$

$$\bar{V}^i = \sum_{j=i}^{N} \bar{L}^j + F(\bar{x}_N) + \bar{k}^T \theta(\bar{x}_N) \tag{62}$$

Thus

$$\delta V^i = V^i - \bar{V}^i = \sum_{j=i}^{N} \delta L^j + (F(x_N) - F(\bar{x}_N)) + (k^T \theta(x_N) - \bar{k}^T \theta(\bar{x}_N)) \tag{63}$$

But, from (11),

$$\delta V^i = a^i + V^i_x \delta x_i + V^i_k \delta k + \frac{1}{2} \delta x_i^T V^i_{xx} \delta x_i + \delta x_i^T V^i_{xk} \delta k + \frac{1}{2} \delta k^T V^i_{kk} \delta k \tag{64}$$

Since (63) and (64) must be equal, their proximity is a test on the size of δx_i and δk. This is because (63) is an exact expression, and (64) is an approximation dependent on δx_i and δk.

In order to use (63) and (64) as a step-by-step test of δx_i, their form should be modified. This is because (63) involves x_N, which is not yet available at step i of the forward integration. The modification is a simple one: from (63),

$$(65) \qquad \delta V^o = \sum_{j=0}^{N} \delta L^j + (F(x_N) - F(\bar{x}_N)) + (k^T\theta(x_N) - \bar{k}^T\theta(\bar{x}_N))$$

Thus,

$$(66) \qquad \delta V^i - \delta V^o = \sum_{j=0}^{i-1} \delta L^j$$

Similarly, $\delta V^i - \delta V^o$ may be calculated from (64).

$$(67) \qquad \delta V^i - \delta V^o = [a^i + V_x^i \delta x_i + V_k^i \delta k + \frac{1}{2}\delta x_i^T V_{xx}^i \delta x_i + \delta x_i^T V_{xk}^i \delta k + \frac{1}{2}\delta k^T V_{kk}^i \delta k] - [a^o + V_k^o \delta k + \frac{1}{2}\delta k^T V_{kk}^o \delta k]$$

The last equation may be simplified somewhat by noticing that $V_k^i = V_k^o$ whenever δk is evaluated. Thus

$$(68) \qquad \delta V^i - \delta V^o = a^i - a^o + V_x^i \delta x_i + \frac{1}{2}\delta x_i^T V_{kx}^i \delta x_i + \delta x_i^T V_{xk}^i \delta k + \frac{1}{2}\delta k^T V_{kk}^i \delta k - \frac{1}{2}\delta k^T V_{kk}^o \delta k$$

Then, test 2 is performed by determining whether (66) agrees with (68) within a given tolerance. If the test is failed† then δk should be reduced, or, if δk is not present, δx_i should be reduced by the step-size adjustment method.

This test is particularly simple to apply in cases where $L(x_i, u_i, t_i) \equiv 0$.

---

† Failure of the test at t_i $(0 < t_i < t_N)$ allows one to discontinue integration of this 'trial trajectory' at t_i instead of integrating all the way to t_N; this can save considerable computer time.

IV. Numerical Example - Comparison with McReynolds' Successive Sweep Method

IV. 1. Statement of the Orbit Transfer Problem

An orbit transfer problem [4], [5], [7], [8], [12] has been solved. In this problem, a control sequence must be found to maximize the radial distance of a rocket from the sun, with the terminal condition that the rocket be in a solar orbit.

x_i is a 3-vector, whose components represent radial distance (from the sun), radial velocity, and angular velocity, respectively, normalized so that the initial condition (in earth's orbit) is

$$x_o = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

$$\theta(x_N) = \begin{pmatrix} x_{2,N} \\ x_{3,N} - \dfrac{1}{\sqrt{x_{1,N}}} \end{pmatrix};$$ ($\theta = 0$ is the condition for a state to be in a stable orbit.)

$$\tilde{L}^i = 0.$$

$F(x_N) = x_{1,N}$. Thus,

$$V = x_{1,N} + k_1\theta_1 + k_2\theta_2$$

$$\tilde{f}^i = \begin{pmatrix} x_{2,i} \\ \dfrac{x_{3,i}^2}{x_{1,i}} - \dfrac{1}{x_{1,i}^2} + A^i \sin u_i \\ -\dfrac{x_{2,i}x_{3,i}}{x_{1,i}} + A^i \cos u_i \end{pmatrix}$$

where

$$A^i = \frac{.1405}{1.0 - .07487t_i} .$$

The time interval $[0, t_N]$ is given.

Note that $\tilde{f}^i(x_i, u_i) = \tilde{F}(x_i) + G^i(u_i)$. Thus $\tilde{H}^i = V_x^{i+1}(\tilde{F}(x_i) + G^i(u_i))$ and $\tilde{H}^i_{ux}$ and $\tilde{f}^i_{ux}$ vanish.

This statement of the problem was inserted into (46)-(54) with terms of order higher than Δt dropped. The equations become:

(69) $$\tilde{\Delta} = \tilde{H}^i_{uu} = V_x^{i+1} G^i_{uu}$$

(70) $$\beta_1 = -\tilde{\Delta}^{-1} \tilde{f}^{i\,T}_u V_{xx}^{i+1}$$

(71) $$\beta_2 = -\tilde{\Delta}^{-1} \tilde{f}^{i\,T}_u V_{xk}^{i+1}$$

(72) $$a^i = a^{i+1} + V_x^{i+1}(G^i(u_i^*) - G^i(\bar{u}_i))\Delta t$$

(73) $$V_x^i = V_x^{i+1} + (V_x^{i+1}\tilde{F}_x(\bar{x}_i) + (G^i(u_i^*) - G^i(\bar{u}_i))V_{xx}^{i+1})\Delta t$$

(74) $$V_k^i = V_k^{i+1} + (G^i(u_i^*) - G^i(\bar{u}_i))V_{xk}^{i+1}\Delta t$$

(75) $$V_{xk}^i = V_{xk}^{i+1} + (\tilde{F}_x(\bar{x}_i)V_{xk}^{i+1} - \beta_1^T\tilde{\Delta}\beta_2)\Delta t$$

(76) $$V_{kk}^i = V_{kk}^{i+1} - \beta_2^T\tilde{\Delta}\beta_2\Delta t$$

(77) $$V_{xx}^i = V_{xx}^{i+1} + \{V_x^{i+1}\tilde{F}_{xx}(\bar{x}_i) + \tilde{F}_x(\bar{x}_i^T)V_{xx}^{i+1} + V_{xx}^{i+1}\tilde{F}_x(\bar{x}_i) + \beta_1^T\tilde{\Delta}\beta_1\}\Delta t$$

where u_i^* was found by maximizing $\tilde{H}^i$ which was equivalent to maximizing $V_x^{i+1}G^i(u_i)$, which, in turn, was equivalent to finding the maximum of

$$V_{x,2}^{i+1}\sin u_i + V_{x,3}^{i+1}\cos u_i \quad .$$

Thus

$$V_{x,2}^{i+1} \cos u_i^* - V_{x,3}^{i+1} \sin u_i^* = 0$$

or,

$$(78) \qquad u_i^* = \arctan(V_{x,2}^{i+1}/V_{x,3}^{i+1})$$

Terms of higher order in Δt were dropped on the assumption that such terms were negligible in comparison with those of order Δt.

In the forward integration phases, $u_i = u_i^* + \delta u_i$ was computed directly by maximizing

$$(79) \qquad \tilde{H}^i(\bar{x}_i + \delta x_i, u_i, V_x^{i+1} + \delta x_{i+1}^T V_{xx}^{i+1} + \delta k^T V_{kx}^{i+1})$$

$$= (V_x^{i+1} + \delta x_{i+1}^T V_{xx}^{i+1} + \delta k^T V_{kx}^{i+1})(\tilde{F}(\bar{x}_i + \delta x_i) + G^i(u_i))$$

with respect to u_i. Note that δx_{i+1} should be replaced by (14), which becomes

$$(80) \qquad \delta x_{i+1} = \delta x_i + \Delta t[(G(u_i) - G(\bar{u}_i)) + F_x(\bar{x}_i)\delta x_i + \frac{1}{2}\delta x_i^T F_{xx}(\bar{x}_i)\delta x_i]$$

However, this is of higher order than the degree of approximation, and it is satisfactory to replace δx_{i+1} in (79) by δx_i.

The new criteria described in the previous section were experimentally applied. Test 1 appeared to be essential for the algorithm to converge. Without it, δk was often chosen too large. Test 2 was found to be helpful and time saving. A more detailed discussion will be found in section V.

IV. 2. Comparison with Successive Sweep Method

This algorithm converges somewhat faster than McReynolds' Successive Sweep Method [4], [5], [6] on this problem, starting from the same initial nominal. This may be because the two techniques differ primarily in the minimization‡ and $f^i - \overline{f}^i$ and $H^i - \overline{H}^i$ terms which are present here and absent from the successive sweep method. But, close to the optimal, those terms are small, and the minimization yields results which are close to McReynolds' method for choosing δu_i. Thus, close to the optimal, the algorithms are very nearly the same. Earlier in the computation, the terms are large, and the minimization permits the present routine to take larger steps. Thus, this routine is able to get to the vicinity of the nominal in fewer iterations than the Successive Sweep Method, and once there, to take just as many additional iterations to converge.

In addition, this routine does not evaluate H_{uu} (or Δ) until after a minimization has been performed. Thus H_{uu} is always negative (definite). McReynolds evaluates H_{uu} on the nominal trajectory, and so, he must either choose his initial nominal so that H_{uu} is negative, or he must invoke a device to partially overcome the difficulty.†

V. Numerical Results

V.I. Discussion of the Trajectories in Tables 1-4

Tables 1-3 contain optimal trajectories calculated for the problem of the previous section by means of the algorithm described above. (The computer program is presented in detail in Appendix B. See the section on the BETA subroutine for an explanation of β_1, β_2, β_3.)

The value of 3.32 was used for t_N in order to compare results with [4] and [5]. The other value, 3.3194 was determined in [12], where the authors solved a minimum time problem. Their problem

† $-|H^i_{uu} + B^i|$ is used in place of H^i_{uu} where B^i is chosen to go to zero as the nominal is approached. See [5, page 596].

‡ Which becomes a maximization in this problem.

was identical with the present problem, except that they specified $x_1(t_N) = 1.525$ (corresponding to the orbit of Mars) as a constraint and left t_N free. Our results agree most closely with those of [12]. (The normalized values of V_x^o agree with $\lambda(t_o)$ given in [12], to 3 figures.)

The rather large differences between the results of 100 time steps and of 400 steps indicate that 100 "Euler integration" steps are not really sufficient to model the continuous time dynamic system. It should be noted that the greatest discrepancies occur in the second-order quantities. But from (69)-(78), those quantities are the only ones whose exact equations have high order Δt terms near the nominal. (Near the nominal, $f^i - \bar{f}^i$ is small or zero.) This may account for the difference in values between our β_1, β_2, and β_3 and those given by McReynolds [5].

It is interesting to note that many different attempts have been made to solve this problem [4], [5], [7], [8], [12]. Our results agree most closely with those quoted in [12] and are more detailed than those previously published.

Table 4 contains a trajectory which maximizes V without regard to terminal constraints for nearly optimal values of k_1 and k_2. It is interesting to note that the maximum obtained for V is far from the maximum V obtained in Tables 1-3, and the θ's are not zero. Thus the free end point problem, with k_1 and k_2 set to their optimal values has at least <u>two local maxima</u>; the one maximum coincides with the point $\theta = 0$, while the other does not. (We have found that if, starting with this other maximum solution, and the optimal k's, the k's are changed successively to reduce $|\theta|$, using the algorithm, then the optimal solution to the problem is obtained. I.e. the k's are adjusted away from their 'optimal' values, but again return to these optimal values, at which stage the 'correct' minimum of V is attained and $\theta = 0$.)

On the average, the program took approximately 3 seconds per iteration for the 100 step program and 12 seconds per iteration for the 400 step program. For this purpose, the "number of iterations" is defined as the number of times the program went into BAKINT (see Appendix B) i.e., the number of times (27), (28), and (32) were integrated. Thus, an iteration includes at least one but possibly as many as 9 times through FORINT, the subprogram that integrates the state equations (1) forward. Also, an iteration may include DKCALC, the program to integrate (30) and (31) and calculate δk by (59).

In the earlier versions of the program, where Test 2 was absent, iteration times averaged as much as 6 seconds for 100 step trajectories. More details on this follow.

The nominal used to compute the trajectory in Table 1 was the nominal McReynolds used: $\bar{k}_1 = -1$; $\bar{k}_2 = 1$; $\bar{u}(t) = 1.57078$ for $0 \leq t \leq 1.66$; $\bar{u}(t) = 5.7124$ for $1.66 < t < 3.32$. Convergence to $|\theta_i(x_N)| < 10^{-6}$ $(i = 1, 2)$ required 15 iterations.

The control history of the nominal used for Tables 2 and 3 was the optimal trajectory computed in [5]. (It was linearly interpolated to 100 points, and then expanded to 400 points by repeating each value four times). For Table 2, $\bar{k}_1 = -1.41936541$, $\bar{k}_2 = 1.264609$, and convergence required 10 iterations. For Table 3, $\bar{k}_1 = -1.399631$, $\bar{k}_2 = 1.260031$ (optimal values from [4]), and 11 iterations were required.

Table 4 was started from a nominal consisting of the control history of Table 1's nominal and $\overline{k}_1$ and $\overline{k}_2$ the same as those of Table 3. It took 6 iterations to "converge."

V.2. Uses of Tests 1 and 2

With neither Test 1 nor Test 2 present the algorithm did not converge. Constraining each new trajectory by the requirement that Test 1 be satisfied was sufficient to ensure convergence. Because this constraint was usually effective - i.e., many values of δk were rejected - this problem appears to be very sensitive to changes in the multipliers k.

Pairs of runs were compared: of each, one had only Test 1; the other had both tests. The comparison indicated a certain redundancy between the two tests. A large number of trial δk's were rejected by both Test 1 (where that was the only test) and Test 2 (where both tests existed.) In fact, the same values of δk were ultimately accepted by the two programs, and the programs generally converged to the same optimal trajectory in the same number of steps.

However, the redundancy was not complete. There were δk's that were accepted by Test 2 and rejected by Test 1.

But the redundancy is helpful. Test 2 can be invoked often in the forward integration phase, while Test 1 can only be invoked after the forward integration phase is complete. Thus Test 2 can save execution time. This time appears to be quite significant: with both tests present, a 100 step iteration took about 3 seconds. With only Test 1, a 100 step iteration took - on the average - more than six seconds. (As pointed out in the footnote on page 20, the forward integration of the system equations can be terminated as soon as

Test 2 fails. However, Test 1 requires that the integration be performed up until t_N. This accounts for the 'time saving' when Test 2 is included.)

A difficulty was encountered in using the tests. As the algorithm approached the optimal, steps and changes in parameters tended to grow rather small. Then all tests which involve differences of large quantities become less reliable - in fact, excessively conservative. Thus there should be some means of disabling the tests when δx_i or δk are sufficiently small.

Once the difficulty was recognized, Test 1 was disabled when $\delta V^o = V^o - \overline{V}^o$ was less, in absolute value, than $10^{-6}\overline{V}^o$. Test 2 was disabled when the absolute value of

$$a^o + V_k^o \delta k + \frac{1}{2} \delta k^T V_{kk}^o \delta k$$

was less than $10^{-6}\overline{V}^o$.

V. 3. Behavior of the Algorithm

The existence of the maximum in Table 4 may be illustrated by analogy with a static maximization of a function of a single variable. See figure 1.

In order to maximize V(u), one may approximate V with a second-order Taylor expansion in the neighborhood of $\overline{u}$, a nominal value.

(81) $$V(u) \approx V(\overline{u}) + V'(\overline{u})(u - \overline{u}) + \frac{1}{2} V''(\overline{u})(u - \overline{u})^2$$

The value of u that maximizes this is given by

$$0 = V'(\overline{u}) + \frac{1}{2} V''(\overline{u})(u - \overline{u})$$

or

(82) $$u = \bar{u} - \frac{V'(\bar{u})}{V''(\bar{u})}$$

Equation (81) may be used to predict the improvement in V using (82).

(83) $$V(u) - V(\bar{u}) = -\frac{1}{2}\frac{V'(\bar{u})^2}{V''(\bar{u})}$$

which is positive for $V''(u) < 0$. Then (83) may be used as a criterion for optimality: when (83) is zero, $\bar{u}$ is a maximum.

If $\bar{u}$ is at point A, and the local maximum at point B is the one desired (rather than the one at point F) some means must be employed to guarantee that (82) will produce a value of u in the neighborhood of B. A value near E will eventually converge to F. Thus (82) should be replaced by

(84) $$u = \bar{u} - \epsilon\frac{V'(\bar{u})}{V''(\bar{u})}$$

Then, (83) becomes

(85) $$V(u) - V(\bar{u}) = \left(\frac{1}{2}\epsilon^2 - \epsilon\right)\frac{V'(\bar{u})^2}{V''(\bar{u})}$$

Thus, an improvement may be guaranteed at every stage if (84) is used with proper choice of ϵ, if the initial nominal lies somewhere to the left of point D.

If the nominal is to the right of point E, the algorithm will tend to point F.

Points between C and E are problematical because $V''(u)$ is not negative-definite†. In neighborhoods of C and E, (84) and (85) are not useable.

† In the case of vector u, an increased cost <u>may</u> be obtained even if $V''(u)$ is non-negative-definite. In the scalar case this is not possible.

This is not perfectly analogous to the algorithm for discrete-time dynamic optimization algorithms, but some comparisons may be drawn. In the discrete dynamic case, u may be thought of as an N-vector (N = 100 or 400). Then V' is a vector, V" is a matrix, and ϵ represents the step-size adjustment method. Figure 1 may be thought of as a graph of V as a function of u for constant, near optimal k. Point B is the local maximum where $\theta_1 = \theta_2 = 0$, and is shown in tables 1-3. Point F is the maximum of table 4.

Behavior due to a point analogous to E has been observed. Iteration began at point A, for near optimal k. The next value of u was to the right of point D (because V calculated at that point was greater than that of Table 1. In this case, N = 100, $t_N = 3.32$). In successive iterations, V continued to increase, as did $|\theta_1|$ and $|\theta_2|$ because u was chosen to maximize H. However, it was impossible to drive a^o (analogous to (85)) below a certain value. After a few iterations, a^o began to increase. Finally, a^o jumped from a typical value of less than 10^{-3} to more than 300 in one iteration. At that iteration, elements of V_{xx} were of the order of 5000. This situation corresponds to a point near C or E where V_{uu} is near singular (the singularity manifests itself in the large values of V_{xx} and a^o).

Thus, in order to guarantee proper convergence iterations must be restricted to the neighborhood of the relative minimum desired. In the present algorithm, the restrictions are accomplished by:

1) The choice of a 'sufficiently good' nominal.
2) Minimization of H(u) (rather than the use of $\delta u = -H_{uu}^{-1} H_u$ as in [5], [6], [9] and [14]).
3) Test 2.

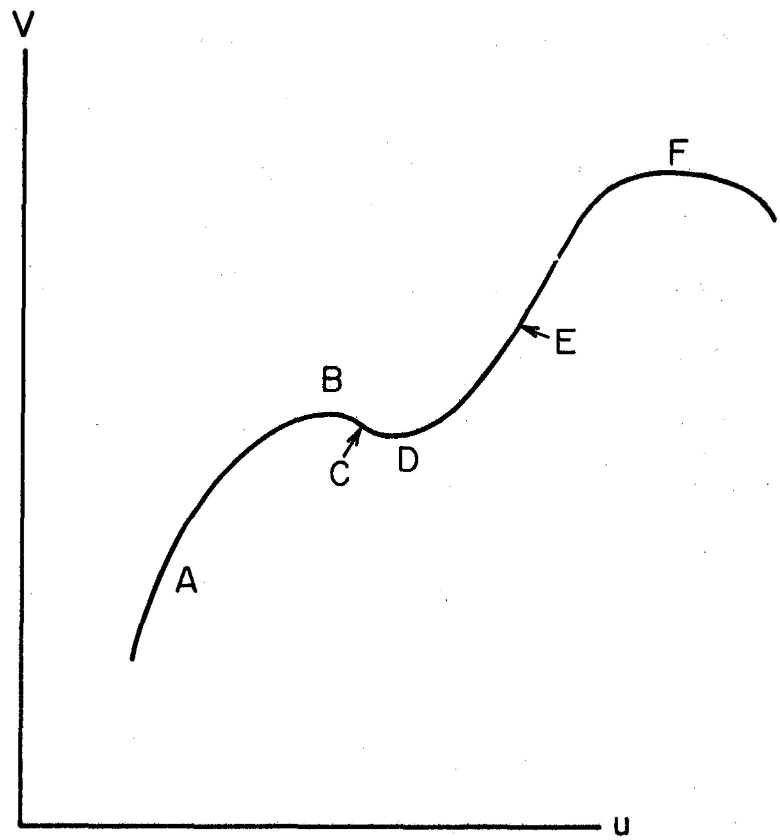


FIGURE 1

V.4. Numerical Values of Tolerances

ETA, the criterion of optimality of a^o, was set to 10^{-2} and good results were obtained. Late in the iteration process, a^o was always less and generally considerably less than this value, so that this constraint is rather ineffective. Earlier in the process, little is gained by requiring a^o to be extremely small, since that would require precise calculation of quantities which must change when k is changed by δk, and which are non-critical.

Satisfactory results were obtained with CK and TOL, the tolerances of Test 1 and Test 2, respectively, set to 20% and 30%. At less than 10%, it became impossible to take steps sufficiently small in δx_i to satisfy Test 2. (This was found with N = 100.)

Table 1

100	Time	Steps	—	Final	Time	=	3.32
t	u	x_1	x_2	x_3	V_{x_1}	V_{x_2}	V_{x_3}
0	.4430	1	0	1	1.8890	.94316	2.0604
.166	.5188	1.0008	.0134	1.0201			
.332	.6073	1.0044	.0353	1.0366	1.5777	.94982	1.4229
.498	.7097	1.0121	.0649	1.0478			
.664	.8269	1.0252	.1011	1.0520	1.2963	.85152	.82311
.830	.9592	1.0446	.1419	1.0479			
.996	1.107	1.0711	.1853	1.0349	1.0857	.64554	.34816
1.162	1.271	1.1047	.2288	1.0129			
1.328	1.460	1.1455	.2701	.9823	.96818	.36222	.055367
1.494	1.730	1.1929	.3071	.9433			
1.660	2.886	1.2459	.3347	.8924	.93356	.045050	-.048480
1.826	4.493	1.3008	.3157	.8370			
1.992	4.765	1.3508	.2786	.8032	.95639	-.27469	.0036793
2.158	4.913	1.3945	.2390	.7811			
2.324	5.023	1.4315	.1991	.7675	1.0117	-.58597	.17505
2.490	5.116	1.4619	.1600	.7611			
2.656	5.196	1.4860	.1225	.7609	1.1031	-.88384	.44689
2.822	5.269	1.5039	.0872	.7661			
2.988	5.335	1.5162	.0546	.7762	1.2105	-1.1608	.81108
3.154	5.398	1.5233	.0254	.7908			
3.320	——	1.5257	.0000	.8096	1.3356	-1.4034	1.2647

Optimal $V = 1.52572699$

$k_1 = -1.40339248$

$k_2 = 1.26501024$

$\theta_1 = .75 \times 10^{-6}$

$\theta_2 = .11 \times 10^{-6}$

Table 1

t	$V_{x_1x_1}$	$V_{x_1x_2}$	$V_{x_1x_3}$	$V_{x_2x_2}$	$V_{x_2x_3}$	$V_{x_3x_3}$
0	17.382	5.0412	25.681	2.7053	4.1297	36.371
.332	14.024	5.9021	19.323	2.5363	5.9353	26.238
.664	10.142	5.6729	12.489	2.6660	5.9028	15.733
.996	6.5844	4.3821	6.8543	2.3731	4.3792	7.5536
1.328	3.8740	2.6629	3.2567	1.5047	2.5309	2.7188
1.660	1.6309	.93085	1.2301	.40562	1.0976	.34899
1.992	1.0014	.43781	.83305	.17979	.74599	-.72441
2.324	.62113	.20735	.68267	.14331	.37114	-1.1303
2.656	.30776	.095264	.51340	.10731	.028400	-1.0641
2.988	10^{-3} × .20185	.037090	.29087	.040093	-.13086	-.61585
3.320	-.33000	0	0	0	0	0

t	$V_{x_1k_1}$	$V_{x_2k_1}$	$V_{x_3k_1}$	$V_{x_1k_2}$	$V_{x_2k_2}$	$V_{x_3k_2}$
0	11.763	1.8948	16.457	2.0921	.47188	2.6138
.332	10.134	3.2408	13.460	1.8111	.60603	2.0839
.664	7.9954	3.8989	9.8678	1.4737	.62691	1.5026
.996	5.7024	3.7125	6.3684	1.1258	.51215	.97654
1.328	3.5734	2.8099	3.5868	.80242	.28264	.59946
1.660	1.2686	1.3089	1.5331	.45836	-.023940	.38236
1.992	.68877	1.0293	1.0404	.36011	-.16333	.56995
2.324	.40892	.95075	.84940	.30271	-.19132	.74612
2.656	.23166	.95901	.61959	.27821	-.15414	.88625
2.988	.10279	.98678	.33387	.26806	-.084390	.97275
3.320	0	1	0	.26537	0	1

t	$V_{k_1k_1}$	$V_{k_1k_2}$	$V_{k_2k_2}$
0	6.2739	1.1584	.34241
.332	5.6858	1.0802	.33200
.664	4.9673	.97793	.31744
.996	4.0578	.83754	.29576
1.325	2.8729	.63669	.26168
1.660	.73108	.28510	.20310
1.992	.29045	.19060	.13001
2.324	.13863	.10495	.081612
2.656	.061401	.053471	.047200
2.988	.020738	.020602	.020546
3.320	0	0	0

Table 2

400 Time Steps —— Final Time = 3.32

t	u	x_1	x_2	x_3	V_{x_1}	V_{x_2}	V_{x_3}
0	.4332	1	0	1	1.8803	.93239	2.0340
.166	.5072	1.0010	.0139	1.0200	1.7254	.94700	1.7201
.332	.5937	1.0049	.0361	1.0362	1.5729	.93843	1.4045
.498	.6936	1.0132	.0659	1.0470	1.4273	.90323	1.0982
.664	.8080	1.0269	.1020	1.0507	1.2942	.83985	.81261
.830	.9371	1.0469	.1425	1.0464	1.1790	.74911	.55832
.996	1.081	1.0740	.1855	1.0332	1.0857	.63410	.34405
1.162	1.241	1.1082	.2285	1.0114	1.0160	.49963	.17548
1.328	1.426	1.1493	.2693	.9811	.96936	.35134	.054908
1.494	1.683	1.1970	.3059	.9428	.94344	.19473	-.018536
1.660	2.645	1.2502	.3335	.8927	.93488	.034410	-.048200
1.826	4.437	1.3048	.3149	.8375	.94036	-.12632	-.039267
1.992	4.732	1.3542	.2780	.8039	.95720	-.28580	.0028600
2.158	4.885	1.3972	.2386	.7818	.98321	-.44323	.074425
2.324	4.999	1.4337	.1988	.7682	1.0168	-.59804	.17286
2.490	5.093	1.4636	.1598	.7617	1.0569	-.74962	.29642
2.656	5.176	1.4872	.1224	.7613	1.1029	-.89717	.44398
2.882	5.250	1.5047	.0871	.7664	1.1541	-1.0396	.61486
2.988	5.318	1.5165	.0546	.7765	1.2102	-1.1755	.80871
3.154	5.382	1.5232	.0254	.7910	1.2709	-1.3029	1.0253
3.320	——	1.5254	.0000	.8097	1.3356	-1.4194	1.2646

Optimal $V = 1.52537493$

$k_1 = -1.41936325$

$k_2 = 1.26460750$

$\theta_1 = -.33 \times 10^{-6}$

$\theta_2 = .37 \times 10^{-7}$

Table 2

t	$V_{x_1x_1}$	$V_{x_1x_2}$	$V_{x_1x_3}$	$V_{x_2x_2}$	$V_{x_2x_3}$	$V_{x_3x_3}$
0	25.668	6.4714	36.811	2.9982	6.1215	51.271
.166	23.004	7.3407	32.083	3.0133	7.5962	43.827
.332	20.020	7.8261	26.940	3.1994	8.4188	35.869
.498	16.875	7.8472	21.711	3.4165	8.5199	27.975
.664	13.767	7.4030	16.748	3.5261	7.9529	20.696
.830	10.887	6.5734	12.357	3.4270	6.8843	14.457
.996	8.3712	5.4944	8.7324	3.0840	5.5498	9.4885
1.162	6.2725	4.3143	5.9299	2.5329	4.1859	5.8081
1.328	4.5578	3.1493	3.8821	1.8550	2.9672	3.2561
1.494	3.1074	2.0442	2.4311	1.1295	1.9686	1.5713
1.660	1.6814	.96700	1.2982	.42449	1.1349	.39922
1.826	1.2733	.66432	.95119	.26681	.91097	-.31000
1.992	1.0094	.44680	.85425	.17135	.75975	-.68761
2.158	.80110	.30614	.77199	.14147	.58102	-.95063
2.324	.62423	.21096	.69358	.13159	.39144	-1.0988
2.490	.46343	.14436	.61088	.12113	.20821	-1.1304
2.656	.30967	.096961	.51862	.10202	.050278	-1.0504
2.822	.15707	.062838	.41337	.073672	-.063771	-.87208
2.988	.0014626	.037691	.29278	.040783	-.11762	-.61715
3.154	-.16020	.017914	.15530	.012180	-.098794	-.31479
3.320	-.33005	0	0	0	0	0

Table 2

t	$V_{x_1k_1}$	$V_{x_2k_1}$	$V_{x_3k_1}$	$V_{x_2k_1}$	$V_{x_2k_2}$	$V_{x_3k_2}$
0	16.324	2.7072	22.568	2.9965	.62088	3.8397
.166	15.077	3.6576	20.360	2.7752	.74548	3.4393
.332	13.635	4.3937	17.898	2.5261	.83261	3.0041
.498	12.044	4.8705	15.288	2.2562	.87461	2.5524
.664	10.369	5.0587	12.652	1.9757	.86650	2.1047
.830	8.6893	4.9548	10.119	1.6962	.80765	1.6828
.996	7.0750	4.5854	7.8073	1.4283	.70248	1.3065
1.162	5.5754	4.0023	5.8054	1.1785	.55971	.99064
1.328	4.1963	3.2638	4.1519	.94615	.38932	.74192
1.494	2.8623	2.3949	2.8156	.71731	.19653	.55547
1.660	1.3086	1.3438	1.5850	.47315	-.0064651	.41096
1.826	.90514	1.1591	1.1539	.42623	-.084704	.48855
1.992	.69021	1.0388	1.0480	.36451	-.15287	.57766
2.158	.53029	.97807	.94985	.32738	-.18133	.66436
2.324	.40854	.95379	.84945	.30386	-.18620	.74675
2.490	.31148	.95055	.74012	.28843	-.17506	.82111
2.656	.23115	.95886	.61829	.27827	-.15239	.88454
2.822	.16260	.97224	.48265	.27178	-.12132	.93504
2.988	.10257	.98591	.33340	.26792	-.084297	.97138
3.154	.048862	.99615	.17185	.26598	-.043302	.99302
3.320	0	1	0	.26540	0	1

Table 2

t	$V_{k_1k_1}$	$V_{k_1k_2}$	$V_{k_2k_2}$	β_1	β_2	β_3
0	8.8160	1.6610	.43881	1.6117	.92861	1.2998
.166	8.3022	1.5836	.42715	1.6380	.92017	1.3739
.332	7.7593	1.5002	.41432	1.6858	.94436	1.4882
.498	7.1776	1.4088	.39995	1.7499	1.0069	1.6473
.664	6.5494	1.3075	.38363	1.8270	1.1172	1.8637
.830	5.8690	1.1947	.36492	1.9211	1.2932	2.1662
.996	5.1333	1.0687	.34336	2.0551	1.5759	2.6231
1.162	4.3363	.92737	.31828	2.3043	2.0820	3.4117
1.328	3.4536	.76489	.28837	2.9216	3.2442	5.0950
1.494	2.3868	.56341	.25032	5.2092	7.7912	10.505
1.660	.77092	.29390	.20465	35.639	100.04	71.209
1.826	.41196	.26081	.17086	-9.1970	-21.079	-23.071
1.992	.29083	.19216	.13187	-14.033	-86.862	-25.854
2.158	.20165	.14280	.10454	-9.4342	-98.806	-14.020
2.324	.13913	.10584	.082684	-4.7097	-118.50	-12.13581
2.490	.094332	.076993	.064096	1.6346	-153.98	21.075
2.656	.061797	.054015	.047855	12.727	-221.03	60.962
2.822	.038102	.035595	.033524	37.541	-366.27	153.11
2.988	.020935	.020859	.020865	114.52	-781.63	442.30
3.154	.0086435	.0091655	.0097293	593.49	-3112.0	2247.1
3.320	0	0	0	∞	∞	∞

Table 3

400 Time Steps —— Final Time = 3.3194

t	u	x_1	x_2	x_3	V_{x_1}	V_{x_2}	V_{x_3}
0	.4333	1	0	1	1.8800	-.93244	2.0334
.166	.5074	1.0010	.0139	1.0199	1.7252	.94699	1.7196
.332	.5938	1.0049	.0361	1.0362	1.5727	.93838	1.4041
.498	.6937	1.0131	.0658	1.0470	1.4272	.90314	1.0979
.664	.8080	1.0268	.1019	1.0507	1.2941	.83974	.81232
.830	.9372	1.0469	.1425	1.0464	1.1790	.74899	.55811
.996	1.081	1.0739	.1854	1.0332	1.0856	.63399	.34391
1.162	1.242	1.1081	.2284	1.0114	1.0160	.49953	.17540
1.328	1.426	1.1493	.2692	.9812	.96934	.35127	.054860
1.494	1.683	1.1969	.3059	.9429	.94343	.19468	-.018561
1.660	2.645	1.2501	.3334	.8927	.93487	.034386	-.048213
1.826	4.437	1.3046	.3148	.8375	.94035	-.12631	-.039278
1.992	4.732	1.3540	.2779	.8040	.95720	-.28576	.0028443
2.158	4.885	1.3971	.2386	.7819	.98321	-.44317	.074400
2.324	4.999	1.4335	.1988	.7682	1.0168	-.59796	.17282
2.490	5.093	1.4634	.1598	.7617	1.0569	-.74951	.29636
2.656	5.176	1.4870	.1223	.7614	1.1029	-.89703	.44390
2.821	5.250	1.5045	.0871	.7665	1.1541	-1.0394	.61476
2.987	5.318	1.5163	.0546	.7765	1.2103	-1.1753	.80858
3.153	5.382	1.5230	.0254	.7911	1.2709	-1.3026	1.0252
3.319	——	1.5252	.0000	.8097	1.3357	-1.4191	1.2644

Optimal $V = 1.52516085$

$k_1 = -1.41910912$

$k_2 = 1.26441935$

$\theta_1 = -.10 \times 10^{-5}$

$\theta_2 = -.26 \times 10^{-6}$

Table 3

t	$V_{x_1x_1}$	$V_{x_1x_2}$	$V_{x_1x_3}$	$V_{x_2x_2}$	$V_{x_2x_3}$	$V_{x_3x_3}$
0	25.654	6.4705	36.789	2.9974	6.1205	51.239
.166	22.991	7.3385	32.064	3.0124	7.5935	43.799
.332	20.009	7.8231	26.924	3.1981	8.4148	35.846
.498	16.866	7.8437	21.698	3.4148	8.5154	27.958
.664	13.760	7.3995	16.739	3.5240	7.9484	20.684
.830	10.882	6.5703	12.351	3.4247	6.8805	14.449
.996	8.3684	5.4918	8.7288	3.0819	5.5469	9.4838
1.162	6.2708	4.3124	5.9278	2.5311	4.1839	5.8055
1.328	4.5568	3.1480	3.8811	1.8536	2.9660	3.2549
1.494	3.1069	2.0433	2.4306	1.1286	1.9680	1.5708
1.660	1.6813	.96672	1.2981	.42418	1.1346	.39905
1.826	1.2735	.66426	.95129	.26668	.91091	-.30994
1.992	1.0096	.44678	.85434	.17128	.75970	-.68749
2.158	.80125	.30613	.77206	.14142	.58099	-.95047
2.324	.62436	.21096	.69364	.13156	.39141	-1.0986
2.490	.46354	.14436	.61091	.12111	.20820	-1.1302
2.656	.30975	.096968	.51865	.10201	.050286	-1.0502
2.821	.15712	.062844	.41338	.073662	-.063752	-.87192
2.987	.0014710	.037694	.29279	.040778	-.11760	-.61704
3.153	-.16023	.017915	.15530	.012178	-.098777	-.31473
3.319	-.33013	0	0	0	0	0

Table 3

t	$V_{x_1k_1}$	$V_{x_2k_1}$	$V_{x_3k_1}$	$V_{x_1k_2}$	$V_{x_2k_2}$	$V_{x_3k_2}$
0	16.317	2.7075	22.557	2.9951	.62084	3.8376
.166	15.071	3.6572	20.351	2.7740	.74530	3.4374
.332	13.630	4.3926	17.890	2.5250	.83230	3.0025
.498	12.039	4.8689	15.281	2.2552	.87420	2.5510
.664	10.366	5.0568	12.647	1.9749	.86604	2.1036
.830	8.6864	4.9528	10.115	1.6956	.80718	1.6819
.996	7.0729	4.5835	7.8046	1.4279	.70203	1.3059
1.162	5.5739	4.0006	5.8036	1.1782	.55932	.99027
1.328	4.1953	3.2625	4.1508	.94593	.38900	.74171
1.494	2.8616	2.3940	2.8151	.71717	.19629	.55536
1.660	1.3084	1.3435	1.5848	.47314	-.0065624	.41096
1.826	.90522	1.1590	1.1540	.42626	-.084763	.48859
1.992	.69030	1.0388	1.0481	.36455	-.15291	.57770
2.158	.53037	.97804	.94994	.32743	-.18136	.66440
2.324	.40861	.95377	.84953	.30391	-.18622	.74678
2.490	.31154	.95054	.74018	.28848	-.17508	.82113
2.656	.24120	.95886	.61834	.27832	-.15240	.88456
2.821	.16264	.97224	.48268	.27183	-.12133	.93505
2.987	.10259	.98590	.33342	.26798	-.084302	.97139
3.153	.048875	.99615	.17186	.26604	-.043304	.99303
3.319	0	1	0	.26546	0	1

Table 3

t	$V_{k_1 k_1}$	$V_{k_1 k_2}$	$V_{k_2 k_2}$	β_1	β_2	β_3
0	8.8130	1.6604	.43870	1.6118	.92857	1.3000
.166	8.2995	1.5831	.42704	1.6381	.92017	1.3742
.332	7.7568	1.4997	.41422	1.6860	.94440	1.4886
.498	7.1754	1.4083	.39986	1.7501	1.0070	1.6478
.664	6.5474	1.3071	.38354	1.8273	1.1174	1.8643
.830	5.8673	1.1943	.36485	1.9215	1.2934	2.1669
.996	5.1318	1.0684	.34330	2.0557	1.5762	2.6241
1.162	4.3350	.92709	.31823	2.3051	2.0825	3.4131
1.328	3.4525	.76466	.28833	2.9229	3.2455	5.0974
1.494	2.3859	.56322	.25029	5.2124	7.7957	10.511
1.660	.77050	.29385	.20465	35.663	100.10	71.248
1.826	.41195	.26081	.17086	-9.1921	-21.042	-23.063
1.992	.29084	.19217	.13187	-14.038	-86.876	-25.861
2.158	.20165	.14280	.10454	-9.4381	-98.827	-14.025
2.324	.13913	.10584	.082681	-4.7120	-118.53	-.13744
2.490	.094332	.076991	.064092	1.6350	-154.02	21.079
2.656	.061797	.054013	.047852	12.732	-221.08	60.976
2.821	.038101	.035593	.033521	37.558	-366.35	153.15
2.987	.020934	.020858	.020863	114.57	-781.82	442.42
3.153	.0086431	.0091649	.0097283	593.80	-3112.9	2247.8
3.319	0	0	0	∞	∞	∞

Table 4

100 Time Steps —— Final Time = 3. 32

t	u	x_1	x_2	x_3	V_{x_1}	V_{x_2}	V_{x_3}
0	1. 147	1.	0	1	1. 5399	3. 8503	1. 5607
. 166	1. 458	1. 0015	. 0233	1. 0055			
. 332	1. 802	1. 0071	. 0489	. 9993	. 065221	3. 4865	-1. 1789
. 498	2. 172	1. 0167	. 0706	. 9812			
. 664	2. 423	1. 0294	. 0826	. 9529	-1. 2528	2. 6667	-3. 5848
. 830	2. 674	1. 0433	. 0833	. 9200			
. 996	2. 819	1. 0565	. 0722	. 8850	-2. 0365	1. 7928	-5. 1481
1. 162	2. 917	1. 0672	. 0508	. 8525			
1. 328	2. 990	1. 0738	. 0203	. 8221	-2. 4128	1. 0467	-5. 9689
1. 494	3. 048	1. 0747	-. 0184	. 7956			
1. 660	3. 098	1. 0686	-. 0650	. 7737	-2. 6385	. 39794	-6. 2592
1. 826	3. 143	1. 0543	-. 1194	. 7572			
1. 992	3. 186	1. 0305	-. 1818	. 7472	-2. 7998	-. 14885	-6. 0700
2. 158	3. 229	. 9957	-. 2532	. 7452			
2. 324	3. 275	. 9484	-. 3349	. 7536	-2. 9268	-. 59906	-5. 3983
2. 490	3. 331	. 8867	-. 4294	. 7765			
2. 656	3. 406	. 8083	-. 5404	. 8211	-2. 9898	-. 97790	-4. 1832
2. 822	3. 528	. 7100	-. 6738	. 9021			
2. 988	3. 778	. 5878	-. 8374	1. 0534	-2. 6070	-1. 3288	-2. 2274
3. 154	4. 473	. 4361	-1. 0316	1. 3731			
3. 320	——	. 2559	-1. 0621	2. 2275	5. 8682	-1. 3996	1. 2600

V 2. 05800182

k_1 = -1. 3996310

k_2 = 1. 2600310

θ_1 = -1. 0620840

θ_2 = 0. 25048833

Table 4

t	$V_{x_1x_1}$	$V_{x_1x_2}$	$V_{x_1x_3}$	$V_{x_2x_2}$	$V_{x_2x_3}$	$V_{x_3x_3}$
0	-24.414	-2.0022	-37.216	20.959	-7.9807	-43.107
	-36.687	-17.856	-47.037	7.6941	-30.388	-45.222
	-53.131	-41.090	-54.533	-13.587	-51.207	-37.624
	-61.956	-58.592	-47.827	-35.420	-58.469	-12.146
	-61.104	-65.932	-27.929	-49.859	-48.845	-21.572
	-51.538	-62.151	-1.0216	-52.137	-27.091	48.156
	-33.480	-48.075	24.962	-41.927	-3.0412	55.434
	-8.6663	-27.949	40.562	-24.679	12.112	41.086
	18.256	-9.1917	39.034	-9.0632	13.118	16.361
	39.504	1.1026	21.040	-.90940	4.8651	-.70196
	-28.541	0	0	0	0	0

Table 4

t	$V_{x_1k_1}$	$V_{x_2k_1}$	$V_{x_3k_1}$	$V_{x_1k_2}$	$V_{x_2k_2}$	$V_{x_3k_2}$
0	-.63978	-1.8778	-.73634	.28581	1.6463	.11388
.	-.058647	-1.8077	.47180	-.55295	1.3166	-1.2612
.	.22644	-1.6848	1.3543	-1.4659	.65052	-2.5802
.	.20785	-1.6316	1.9042	-2.2572	-.18701	-3.5000
.	.055229	-1.6358	2.3214	-2.9432	-1.0756	-3.9021
.	-.11039	-1.6351	2.7404	-3.4968	-1.8585	-3.6752
.	-.14652	-1.5086	3.2172	-3.7835	-2.3211	-2.7753
.	.10870	-1.1360	3.6343	-3.6113	-2.2884	-1.3815
.	.77093	-.50338	3.6627	-2.8160	-1.7731	.081057
.	1.8439	2.9461	2.8218	-1.1017	-.96980	1.1273
.	0	1	0	3.8635	0	1

Table 4

t	$V_{k_1k_1}$	$V_{k_1k_2}$	$V_{k_2k_2}$
0	.17173	.22694	.30997
.332	.17168	.22658	.30454
.664	.17005	.22270	.29477
.996	.16428	.21388	.28121
1.328	.15402	.19916	.26008
1.660	.13853	.17695	.22827
1.992	.11778	.14737	.18608
2.324	.093937	.11390	.13910
2.656	.070021	.081529	.095279
2.988	.043426	.048409	.053988
3.320	0	0	0

VI. Conclusion

A new discrete algorithm has been derived which is analogous to the continuous algorithm of [1] and [2]. Extensions to the latter (Test 1 and Test 2) have been developed to ensure that the new iterate is in the neighborhood of the current nominal.

The algorithm has been used to solve a non-linear, optimal orbit transfer problem. This problem has been attempted, and solved, in various forms, by a number of investigators using different computational methods.

The results obtained in this paper agree most closely with those of [12].

References

[1] D. H. Jacobson, Differential Dynamic Programming Methods for Determining Optimal Control of Non-Linear Systems, Ph. D. Thesis, Imperial College of Science and Technology, University of London, October, 1967.

[2] D. H. Jacobson, New Second Order and First Order Algorithms for Determining Optimal Control: A Differential Dynamic Programming Approach, Technical Report. 551, Division of Engineering and Applied Physics, Harvard University, February, 1968 also J. Opt. Theory Applns. 2, No. 6, 1968 to appear.

[3] R. Bellman and R. Kalaba, Dynamic Programming and Modern Control Theory, Academic Press, New York, 1965.

[4] S. R. McReynolds, A Successive Sweep Method for Solving Optimal Control Problems, Ph. D. Thesis, Harvard University, Cambridge, 1965.

[5] S. R. McReynolds, The Successive Sweep Method and Dynamic Programming, Journal of Mathematical Analysis and Applications, Vol. 19, No. 3, pp 565-598 September, 1967.

[6] S. R. McReynolds and A. E. Bryson, Jr, A Successive Sweep Method for Solving Optimal Programming Problems, J.A.C.C., 6, 1965, 551.

[7] H. J. Kelley, R. E. Kopp, H. G. Moyer, A Trajectory Optimization Technique Based Upon the Theory of the Second Variation, AIAA Astrodynamics Conference, New Haven, Conn., August, 1963.

[8] H. J. Kelley, Method of Gradients, Optimization Techniques, Ed. G. Leitman, Academic Press, 1961, Ch 6.

[9] D. Q. Mayne, A Second Order Gradient Method of Optimizing Non-Linear Discrete-Time Systems, Int. J. Control, 3, 1966, 85.

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

[11] T. E. Bullock and G. F. Franklin, A Second-Order Feedback Method for Optimal Control Computations, IEEE Trans/on Auto. Control. AC-12, 1967, p 666.

[12] B. D. Tapley and J. M. Lewallen, Comparison of Several Numerical Optimization Methods, Journal of Optimization Theory and Applications, Vol. 1, No. 1, July 1967.

[13] A. E. Bryson and Y. C. Ho, Optimization, Estimation and Control, Blaisdell Publishing Co, Waltham, Mass., 1968 to appear, Chs. 5-7.

[14] S. E. Dreyfus, The Numerical Solution of Non-Linear Optimal Control Problems, in Numerical Solutions of Non-Linear Differential Equations, D. Greenspan, Editor, John Wiley & Sons, 1966, pp 97-113.

Appendix A

Continuous Results from Jacobson

The following is a statement and solution of the continuous-time optimal control problem solved in [1]. The notation has been modified to conform to that of this paper. Thus some expression involving derivatives have been transposed, and $\sim$ has been placed over certain symbols to coincide with section III. 1, above.

Problem: given that

A-1 $$\dot{x} = \tilde{f}(x, u, t) \quad ; \quad x(t_o) = x_o$$

Find u(t), $t \in [t_o, t_f]$ to minimize

A-2 $$\hat{V}(x_o, t_o) = \int_{t_o}^{t_f} \tilde{L}(x, u, t)dt + F(x(t_f))$$

while satisfying

A-3 $$\theta(x(t_f)) = 0$$

The constraints (A-3) are adjoined to the cost functional (A-2):

A-4 $$V(x_o, t_o) = \hat{V} + k^T \theta(x(t_f))$$

The solution is:

A-5 $$\beta_1 = -\tilde{H}_{uu}^{-1}(\tilde{H}_{ux} + \tilde{f}_u^T V_{xx})$$

A-6 $$\beta_2 = -\tilde{H}_{uu}^{-1}\tilde{f}_u^T V_{xk}$$

A-7 $$-\dot{a} = \tilde{H} - \tilde{\tilde{H}}$$

A-8 $$-\dot{V}_x = \tilde{H}_x + (\tilde{f} - \tilde{\tilde{f}})V_{xx}$$

A-9 $$-\dot{V}_k = (\tilde{f} - \tilde{\tilde{f}})V_{xk}$$

A-10 $$-\dot{V}_{xk} = (\tilde{f}_x^T + \beta_1^T \tilde{f}_u^T) V_{xk}$$

A-11 $$-\dot{V}_{kk} = -V_{xk}^T \tilde{f}_u \tilde{H}_{uu}^{-1} \tilde{f}_u^T V_{xk}$$

A-12 $$-\dot{V}_{xx} = \tilde{H}_{xx} + \tilde{f}_x^T V_{xx} + V_{xx} \tilde{f}_x - (\tilde{H}_{ux} + f_u^T V_{xx})^T \tilde{H}_{uu}^{-1} (H_{ux} + f_u^T V_{xx})$$

where $\tilde{H} = \tilde{L} + V_x \tilde{f}$, and derivatives of H are taken with V_x constant, i.e.

$$\tilde{H}_x = \tilde{L}_x + V_x \tilde{f}_x$$

The boundary conditions of (A-7) through (A-12) are the same as equations (33)-(38) above.

Appendix B

The Computer Program

Implementation of the algorithm on the problem described in section three required the use of a computer. A program has been written for the IBM 7094 in FORTRAN IV, which consists of several subprograms.

1. MAIN

This program is described in Flow Chart II in general outline. This program coordinates the algorithm. It starts by setting initial quantities, and quantities which do not change throughout the computation. Included are input numbers, constant elements of $\tilde{f}_x$ and $\tilde{f}_{xx}$, and constant boundary conditions.

The routine FORINT is called, which integrates the state equations (1). On the first iteration, the initial nominal control history is used. Subsequently, u_i is calculated in FORINT. The performance index and terminal constraints are evaluated.

The calling of FORINT is part of the "step-size adjustment", as described in [1] and [2] and Flow Chart I.

Once a suitable trajectory is calculated, it is printed out and BAKINT is called to integrate the equations for a^i, V_x^i, and V_{xx}^i. If the absolute values of a^o and the terminal constraints are less than ETA, ETA1, and ETA2, respectively (which are input quantities), iteration ceases. The routine BETA is called, which calculates the optimal feedback vector β such that on a path slightly perturbed from the optimal, $\delta u = \beta^T \delta x$.

If a^o is not smaller than ETA in absolute value, the program transfers to the forward integrator to improve the nominal trajectory.

When the trajectory has been optimized for a given value of k, i.e., when a^o is driven to less than ETA, the routine DKCALC is called, which integrates the V^i_{kk} and V^i_{xk} equations, and calculates δk according to (59). The value of ϵ is originally 1., but if each component of θ is not decreased (by the introduction of δk) in absolute value, and if the change in performance index is not within a tolerance (an input quantity) of the value predicted by (60) (i.e., if Test 1 is failed), then ϵ is reduced by half and the forward integrator is called again to calculate θ and V. When the criteria are satisfied, $\bar{k}$ is replaced by $\bar{k} + \delta k$ and the program transfers to BAKINT.

2. FORINT

This routine integrates (1) forward. It calculates u_i by maximizing

$$H(\bar{x}_i + \delta x_i, u_i, k + \delta k, t_i) = V_x^{i+1}(\bar{x}_{i+1} + \delta x_{i+1}, k + \delta k) f(\bar{x}_i + \delta x_i, u_i, t_i) \ ,$$

which is equivalent to maximizing

$$E = C \sin u_i + D \cos u_i \quad , \qquad \text{where}$$

$$C = V_{x_2}^{i+1}(\bar{x}_{i+1} + \delta x_{i+1}, \bar{k} + \delta k)$$

$$D = V_{x_3}^{i+1}(\bar{x}_{i+1} + \delta x_{i+1}, \bar{k} + \delta k) \ .$$

C and D are calculated by expanding V_x^{i+1} in δx_{i+1} and δk. However, δx_i is used in place of δx_{i+1}. See section IV.1.

At the maximum of E,

$$u_i = \tan^{-1}(C/D) \ ,$$

but this also determines a minimum. The maximum is chosen simply by requiring that E be positive.

Test 2 is applied by determining whether (11) is constant (within a tolerance TOL) over time.† (It should be constant because L^i is zero.) Because this test is time consuming, it is done at rare intervals.

3. BAKINT

This routine calculates u_i^* according to (19), (in a similar fashion to that of calculating u_i in FORINT) and integrates (27), (28), and (32) with (33), (34), and (38) as boundary conditions. It prints out its results.

4. DKCALC

This integrates (31) and (32) with (36) and (37) as boundary conditions, and prints values of V^i_{xk}, V^i_{kk}. At t = 0, it calculates δk according to (58).

5. START

This short routine accepts input information. The input must include the maximum number of iterations, the number of time steps, the tolerances ETA, ETA1, ETA2, CK, and TOL, the initial value of $\bar{k}$, and the initial nominal control history.

6. BETA

The optimal perturbation feedback law for small deviations from an optimal trajectory is given by (22), which, in the present problem, may be approximated by,

$$\delta u_i = -H^{i^{-1}}_{uu} f^i_u [V^{i+1}_{xx} \delta x_i + V^{i+1}_{xk} \delta k] \ .$$

From (58), and since $V^i_k = \theta^T = 0$ on an optimal trajectory,

$$\delta k = -V^{i+1^{-1}}_{kk} V^{i+1}_{kx} \delta x_{i+1}$$

---

† or from (68), $\delta V_i - \delta V_o \approx 0$.

To first-order in Δt (in a problem which originates from a continuous problem), this may be written

$$\delta k = -{V_{kk}^{i+1}}^{-1} V_{kx}^{i+1} \delta x_i \quad .$$

See section IV. 1.

Thus,

$$\delta u_i = -{H_{uu}^{i}}^{-1} f_u^i [V_{xx}^{i+1} - V_{xk}^{i+1} {V_{kk}^{i+1}}^{-1} V_{kx}^{i+1}] \delta x_i$$

The coefficient of δx_i is calculated in BETA, and printed as β_1, β_2, β_3.

Joint Services Electronics Program

N00014-67-A-0298-0006, 0005, and 0008

Academy Library (DFSLB)
U. S. Air Force Academy
Colorado Springs, Colorado 80912

AEDC (ARO, INC)
Attn: Library/Documents
Arnold AFB, Tenn. 37389

Aeronautics Library
Graduate Aeronautical Laboratories
California Institute of Technology
1201 E. California Blvd.
Pasadena, California 91109

Aerospace Corporation
P. O. Box 95085
Los Angeles, Calif. 90045
Attn: Library Acquisitions Group

Airborne Instruments Laboratory
Deerpark, New York 11729

AFAL (AVTE/R. D. Larson)
Wright-Paterson AFB
Ohio 45433

AFCRL (CRMXLR)
ARCRL Research Library, Stop 29
L. G. Hanscom Field
Bedford, Mass. 01731

AFETR (ETLIG - 1)
STINFO Officer (for library)
Patrick AFB, Florida 32925

AFETR Technical Library
(ETV, MU-135)
Patrick AFB, Florida 32925

AFFTC (FRBPP-2)
Technical Library
Edwards AFB, Calif. 93523

APGC (PHBPS-12)
Eglin AFB
Florida 32542

ARL (ARIY)
Wright-Paterson AFB
Ohio 45433

AULST-9663
Maxwell AFB
Alabama 36112

Mr. Henry L. Bachman
Assistant Chief Engineer
Wheeler Laboratories
122 Cuttermill Road
Great Neck, N. Y. 11021

Bendix Pacific Division
11600 Sherman Way
North Hollywood, Calif. 91605

Colonel A. D. Blue
RTD (RTTL)
Bolling AFB
Washington, D. C. 20332

California Institute of Technology
Pasadena, California 91109
Attn: Documents Library

Carnegie Institute of Technology
Electrical Engineering Dept.
Pittsburg, Pa. 15213

Central Intelligence Agency
Attn: OCR/DD Publications
Washington, D. C. 20505

Chief of Naval Operations
OP-07
Washington, D. C. 20350 [2]

Chief of Naval Research
Department of the Navy
Washington, D. C. 20360
Attn: Code 427 [3]

Commandant
U. S. Army and General Staff College
Attn: Secretary
Fort Leavenworth, Kansas 66370

Commander
Naval Air Development and
Material Center
Johnsville, Pennsylvania 18974

Commanding General
Frankford Arsenal
Attn: SMUFA-L6000 (Dr. Sidney Ross)
Philadelphia, Pa. 19137

Commandant
U. S. Army Air Defense School
Attn: Missile Sciences Div. C and S Dept.
P. O. Box 9390
Fort Bliss, Texas 79916

Commander
U. S. Naval Air Missile Test Center
Point Magu, California 93041

Commanding General
Attn: STEWS-WS-VT
White Sands Missile Range
New Mexico 88002 [2]

Commanding General
U. S. Army Electronics Command
Fort Monmouth, N. J. 07703
Attn: AMSEL-SC
RD-D
RD-G
RD-GF
RD-MAT
XL-D
XL-E
XL-C
XL-S
HL-D
HL-CT-R
HL-CT-P
HL-CT-L
HL-CT-O
HL-CT-I
HL-CT-A
NL-D
NL-A
NL-P
NL-R
NL-S
KL-D
KL-E
KL-S
KL-T
VL-D
WL-D

Commanding General
U. S. Army Material Command
Attn: AMCRD-RS-DE-E
Washington, D. C. 20315

Commanding General
U. S. Army Missile Command
Attn: Technical Library
Redstone Arsenal, Alabama 35809

Commanding Officer
Naval Avionics Facility
Indianapolis, Indiana 46241

Commanding Officer
U. S. Army Limited War Laboratory
Attn: Technical Director
Aberdeen Proving Ground
Aberdeen, Maryland 21005

Commanding Officer
U. S. Army Materials Research Agency
Watertown Arsenal
Watertown, Massachusetts 02172

Commanding Officer
U. S. Army Security Agency
Arlington Hall
Arlington, Virginia 22212

Commanding Officer and Director
U. S. Naval Underwater Sound Lab.
Fort Trumbull
New London, Conn. 06840

Defense Documentation Center
Attn: TISIA
Cameron Station, Bldg. 5
Alexandria, Virginia 22314 [20]

Det No. 6, OAR (LODAR)
Air Force Unit Post Office
Los Angeles, Calif. 90045

Director
Advanced Research Projects Agency
Department of Defense
Washington, D. C. 20301

Director for Materials Sciences
Advanced Research Projects Agency
Department of Defense
Washington, D. C. 20301

Director
Columbia Radiation Laboratory
Columbia University
538 West 120th Street
New York, New York 10027

Director
Coordinated Science Laboratory
University of Illinois
Urbana, Illinois 61803

Director
Electronics Research Laboratory
University of California
Berkeley, California 94720

Director
Electronic Sciences Laboratory
University Southern California
Los Angeles, California 90007

Director
Microwave Laboratory
Stanford University
Stanford, California 94305

Director - Inst. for Exploratory
Research
U. S. Army Electronics Command
Attn: Mr. Robert O. Parker
Executive Secretary, JSTAC
(AMSEL-XL-D)
Fort Monmouth, N. J. 07703

Director
National Security Agency
Fort George G. Meade
Maryland 20755
Attn: James T. Tippett

Director, Naval Research Laboratory
Technical Information Officer
Washington, D. C.
Attn: Code 2000 [8]

Director
Research Laboratory of Electronics
Massachusetts Institute of Technology
Cambridge, Mass. 02139

Director
Stanford Electronics Laboratories
Stanford University
Stanford, California 94305

Commanding Officer
Naval Ordnance Laboratory
Corona, California 91720

Commanding Officer
Naval Ordnance Laboratory
White Oak, Maryland 21502 [2]

Commanding Officer
Naval Ordnance Test Station
China Lake, Calif. 93555

Commanding Officer
Naval Training Device Center
Orlando, Florida 32811

Commanding Officer
Office of Naval Research Branch Office
1030 East Green Street
Pasadena, California

Commanding Officer
Office of Naval Research Branch Office
219 South Dearborn Street
Chicago, Illinois 60604

Commanding Officer
Office of Naval Research Branch Office
495 Summer Street
Boston, Massachusetts 02210

Commanding Officer
Office of Naval Research Branch Office
207 West 24th Street
New York, New York 10011

Commanding Officer
Office of Naval Research Branch Office
Box 39, Fleet Post Office
New York 09510 [2]

Commanding Officer
U. S. Army Electronics R & D Activity
White Sands Missile Range
New Mexico 88002

Commanding Officer
U. S. Army Engineer R & D Laboratory
Attn: STINFO Branch
Fort Belvoir, Virginia 22060

Commanding Officer
U. S. Army Research Office (Durham)
Attn: CRD-AA-IP (Richard O. Ulsh)
Box CM, Duke Station
Durham, North Carolina 27706

Commanding General
USASTRATCOM
Technical Information Center
Fort Huachuca, Arizona 85613

Commanding Officer
Harry Diamond
Attn: Dr. Berthold Altman (AMXDO-TI)
Connecticut Ave. & Van Ness St. NW
Washington, D. C. 20438

Commanding Officer
Human Engineering Laboratories
Aberdeen Proving Ground
Maryland 21005

Commanding Officer
U. S. Army Ballistics Research Lab.
Attn: V. W. Richards
Aberdeen Proving Ground
Maryland 21005

Director, USAF Project RAND
Via: Air Force Liaison Office
The RAND Corporation
1700 Main Street
Santa Monica, Calif. 90406
Attn: Library

Director
U. S. Army Engineer Geodesy,
Intelligence and Mapping
Research and Development Agency
Fort Belvoir, Virginia 22060

Director
U. S. Naval Observatory
Washington, D. C. 20390

Director, U. S. Naval Security Group
Attn: G43
3801 Nebraska Avenue
Washington, D. C. 20390

Division of Engineering and Applied
Physics
130 Pierce Hall
Harvard University
Cambridge, Massachusetts 02138

Professor A. A. Dougal, Director
Laboratories for Electronics and
Related Sciences Research
University of Texas
Austin, Texas 78712

ESD (ESTI)
L. G. Hanscom Field
Bedford, Mass. 01731 [2]

European Office of Aerospace Research
Shell Building
47 Rue Cantersteen
Brussels, Belgium [2]

Colonel Robert E. Fontana
Dept. of Electrical Engineering
Air Force Institute of Technology
Wright-Patterson AFB, Ohio 45433

General Electric Company
Research Laboratories
Schenectady, New York 12301

Professor Nicholas George
California Institute of Technology
Pasadena, California 91109

Goddard Space Flight Center
National Aeronautics and Space Admin.
Attn: Library, Documents Section
Code 252
Green Belt, Maryland 20771

Dr. John C. Hancock, Director
Electronic Systems Research Laboratory
Purdue University
Lafayette, Indiana 47907

Dr. H. Harrison, Code RRE
Chief, Electrophysics Branch
National Aeronautics and Space Admin.
Washington, D. C. 20546

Head, Technical Division
U. S. Naval Counter Intelligence
Support Center
Fairmont Building
4420 North Fairfax Drive
Arlington, Virginia 22203

Headquarters
Defense Communications Agency
The Pentagon
Washington, D. C. 20305

Dr. L. M. Hollinsworth
ARCRL (CRN)
L. G. Hanscom Field
Bedford, Massachusetts 01731

Hunt Library
Carnegie Institute of Technology
Schenely Park
Pittsburgh, Pa. 15213

The Johns Hopkins University
Applied Physics Laboratory
8621 Georgia Avenue
Silver Spring, Maryland 20910
Attn: Boris W. Kuvshinoff
Document Librarian

Lt. Col. Robert B. Kalisch
Chief, Electronics Division
Directorate of Engineering Sciences
Air Force Office of Scientific Research
Arlington, Virginia 22209 [5]

Colonel Kee
ARFSTE
Hqs. USAF
Room ID-429, The Pentagon
Washington, D. C. 20330

Dr. S. Benedict Levin, Director
Institute for Exploratory Research
U. S. Army Electronics Command
Fort Monmouth, New Jersey 07703

Los Alamos Scientific Laboratory
Attn: Reports Library
P. O. Box 1663
Los Alamos, New Mexico 87544

Librarian
U. S. Naval Electronics Laboratory
San Diego, California 95152 [2]

Lockheed Aircraft Corp.
P. O. Box 504
Sunnyvale, California 94088

Dr. I. R. Mirman
AFSC (SCT)
Andrews Air Force Base, Maryland

Lt. Col. Bernard S. Morgan
Frank J. Seiler Research Laboratory
U. S. Air Force Academy
Colorado Springs, Colorado 80912

Dr. G. J. Murphy
The Technological Institute
Northwestern University
Evanston, Illinois 60201

Mr. Peter Murray
Air Force Avionics Laboratory
Wright-Patterson AFB, Ohio 45433

NASA Lewis Research Center
Attn: Library
21000 Brookpark Road
Cleveland, Ohio 44135

NASA Scientific & Technical
Information Facility
Attn: Acquisitions Branch (S/AK/DL)
P. O. Box 33
College Park, Maryland 20740 [2]

National Science Foundation
Attn: Dr. John R. Lehmann
Division of Engineering
1800 G Street, NW
Washington, D. C. 20550

National Security Agency
Attn: R4 - James Tippet
Office of Research
Fort George C. Meade, Maryland 20755

Naval Air Systems Command
AIR 03
Washington, D. C. 20360 [2]

Naval Electronics Systems Command
ELFX 03
Falls Church, Virginia 22046 [2]

Naval Ordnance Systems Command
ORD 32
Washington, D. C. 20360 [2]

Naval Ordnance Systems Command
SHIP 035
Washington, D. C. 20360

Naval Ship Systems Command
SHIP 031
Washington, D. C. 20360

New York University
College of Engineering
New York, New York 10019

Dr. H. V. Noble
Air Force Avionics Laboratory
Wright-Patterson AFB, Ohio 45433

Office of Deputy Director
(Research and Information Rm. 3D1037)
Department of Defense
The Pentagon
Washington, D. C. 20301

Polytechnic Institute of Brooklyn
55 Johnson Street
Brooklyn, New York 11201
Attn: Mr. Jerome Fox
Research Coordination

RAD (EMLAL-1)
Griffiss AFB, New York 13442
Attn: Documents Library

Raytheon Company
Bedford, Mass. 01730
Attn: Librarian

Lt. Col. J. L. Reeves
AFSC (SCBB)
Andrews Air Force Base, Md. 20331

Dr. A. A. Dougal
Asst. Director of Research
Office of Defense Res. and Eng.
Department of Defense
Washington, D. C. 20301

Research Plans Office
U. S. Army Research Office
3045 Columbia Pike
Arlington, Virginia 22204

Dr. H. Robl, Deputy Chief Scientist
U. S. Army Research Office (Durham)
Durham, North Carolina 27706

Emil Schafer, Head
Electronics Properties Info. Center
Hughes Aircraft Company
Culver City, California 90230

School of Engineering Sciences
Arizona State University
Tempe, Arizona 85281

SAMSO (SMSDI-STINFO)
AF Unit Post Office
Los Angeles, California 90045

SSD (SSTRT/Lt. Starbuck)
AFUPO
Los Angeles, California 90045

Superintendent
U. S. Army Military Academy
West Point, New York 10996

Colonel A. Swan
Aerospace Medical Division
AMD (AMRXI)
Brooks AFB, Texas 78235

Syracuse University
Dept. of Electrical Engineering
Syracuse, New York 13210

University of California
Santa Barbara, California 93106
Attn: Library

University of Calif. at Los Angeles
Dept. of Engineering
Los Angeles, California 90024

University of Michigan
Electrical Engineering Dept.
Ann Arbor, Michigan 48104

U. S. Army Munitions Command
Attn: Technical Information Branch
Picatinney Arsenal
Dover, New Jersey 07801

U. S. Army Research Office
Attn: Physical Sciences Division
3045 Columbia Pike
Arlington, Virginia 22204

U. S. Atomic Energy Commission
Division of Technical Information Ext.
P. O. Box 62
Oak Ridge, Tenn. 37831

Dept. of Electrical Engineering
Texas Technological College
Lubbock, Texas 79409

U. S. Naval Weapons Laboratory
Dahlgren, Virginia 22448

Major Charles Waespy
Technical Division
Deputy for Technology
Space Systems Division, AFSC
Los Angeles, California 90045

The Walter Reed Institute of Research
Walter Reed Medical Center
Washington, D. C. 20012

AFSC (SCTR)
Andrews Air Force Base
Maryland 20331

Weapons Systems Test Division
Naval Air Test Center
Patuxtent River, Maryland 20670
Attn: Library

Weapons Systems Evaluation Group
Attn: Col. Daniel W. McElwee
Department of Defense
Washington, D. C. 20305

Yale University
Engineering Department
New Haven, Connecticut 06720

Mr. Charles F. Yost
Special Asst. to the Director of Research
NASA
Washington, D. C. 20546

Dr. Leo Young
Stanford Research Institute
Menlo Park, California 94025

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Division of Engineering and Applied Physics Harvard University Cambridge, Massachusetts	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

A DISCRETE-TIME DIFFERENTIAL DYNAMIC PROGRAMMING ALGORITHM WITH APPLICATION TO OPTIMAL ORBIT TRANSFER

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Interim technical report

5. AUTHOR(S) *(First name, middle initial, last name)*

Stanley B. Gershwin and David H. Jacobson

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
August 1968	60	14

8a. CONTRACT OR GRANT NO. N00014-67-A-0298-0006 and NASA Grant NGR 22-007-068 b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) Technical Report No. 566 9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Office of Naval Research

13. ABSTRACT

Recently, the notion of Differential Dynamic Programming has been used to obtain new second-order algorithms for solving non-linear optimal control problems. (Unlike conventional Dynamic Programming, the Principle of Optimality is applied in the neighborhood of a nominal, non-optimal, trajectory.) A novel feature of these algorithms is that they permit strong variations in the system trajectory.

In this paper, Differential Dynamic Programming is used to develop a second-order algorithm for solving discrete-time dynamic optimization problems with terminal constraints. This algorithm also utilizes strong variations and, as a result, has certain advantages over existing discrete-time methods.

A non-linear computed example is presented, and comparisons are made with the results of other researchers who have solved this problem.

The experience gained during the computation has suggested some extensions to an earlier, previously published Differential Dynamic Programming algorithm for continuous time problems. These extensions, and their implications are discussed.

Unclassified
Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Dynamic Programming Differential Dynamic Programming Orbit Transfer Optimization Algorithm						

DD FORM 1 NOV 65 1473 (BACK)

S/N 0101-807-6821

Unclassified
Security Classification

A-31409