A Discrete-Time Differential Dynamic Programming Algorithm with Application to Optimal Orbit Transfer

STANLEY B. GERSHWIN* AND DAVID H. JACOBSON†
Harvard University, Cambridge, Mass.

Recently, the notion of Differential Dynamic Programming has been used to obtain new second-order algorithms for solving nonlinear optimal control problems. (Unlike conventional Dynamic Programming, the Principle of Optimality is applied in the neighborhood of a nominal, nonoptimal, 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 nonlinear 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.

Notation

THE scalar product of column vectors a and b, where

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

is a^Tb or b^Ta and is equal to

$$\sum_{i=1}^{n} a_i b_i$$

The derivative of a scalar V by a column vector x is a row vector, and is written as

$$V_x = \partial V/\partial x = [\partial V/\partial x_1, \ldots, \partial V/\partial x_n]$$

The second derivative of a scalar V by vectors x and k is a matrix

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

Received September 15, 1969; revision received February 16, 1970. The authors wish to thank A. E. Bryson Jr. for his valuable suggestions, and 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 accessibility during the preparation of this paper. 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 NASA under Grant NGR 22-007-068.

* Graduate Student, Division of Engineering and Applied Physics.

†Assistant Professor, Division of Engineering and Applied Physics.

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

A second-order Taylor expansion is written as

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

I. Introduction

Jacobson[1-3] 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[4-11] 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.[4,5,8,11] The algorithm can be used to solve continuous time problems that are approximated by difference equations.

A nonlinear numerical example is presented and comparisons are drawn with McReynolds[4,5] and others[7,12] who have solved this problem previously, using other methods. The experience gained in the numerical computation has suggested extensions to the continuous algorithms in Refs. 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

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

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

‡ Although global variations in control are used, only small changes are allowed in the trajectory. Thus the method is not one for finding global minima. However, convergence to a local minimum is usually faster than that obtainable using existing "weak variation algorithms."

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

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

where the solution must satisfy the (vector) equality constraint

$$\theta(x_N) = 0 \tag{3}$$

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

The minimized value of $\hat{W}$ is referred to as $\hat{V}$. Defining

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

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

The minimized value of $W(x_o,k,t_o)$ is referred to as $V(x_o, k,t_o)$. More generally, if

$$W(x_j,k,t_j) = \sum_{i=j}^{N-1} L(x_i,u_i,t_i) + F(x_N) + k^T\theta(x_N)$$

where $x_{j+1}, \ldots, x_N, u_j, \ldots, u_{N-1}$ satisfy (1), then

$$V(x_j,k,t_i) = \min_{u_i, i \geq j} W(x_j,k,t_j)$$

and V is called the *optimal return function.*

The *nominal trajectory* $\bar{x}_o = x_o, \bar{x}_1, \ldots, \bar{x}_N$ is the solution to (1) with $u_i = \bar{u}_i$, $i = 0, \ldots, N-1$. The *nominal return function* $\bar{W}(\bar{x}_i,\bar{k},t_i)$ is evaluated with $k = \bar{k}$ along the nominal trajectory.

2. Outline of the Solution

The optimal return function V satisfies Bellman's "Principle of Optimality,"[12] which in this case is

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

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

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

$$x_i = \bar{x}_i + \delta x_i,\ x_{i+1} = \bar{x}_{i+1} + \delta x_{i+1},\ k = \bar{k} + \delta k$$

and Eq. (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 Eq. (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 Eq. (6) with $\delta x_i = 0$ and $\delta k = 0$.**

b) Expand about u_i^* with δx_i and δk nonzero, and minimize with respect to δu_i. This gives 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 are sufficiently small so that all Taylor expansions can be terminated at second-order terms.

3. Solution

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

$$V(\bar{x}_i + \delta x_i,\bar{k} + \delta k,t_i) = V(\bar{x}_i,\bar{k},t_i) + (\partial/\partial x)V(\bar{x}_i,\bar{k},t_i)\delta x_i + (\partial/\partial k)V(\bar{x}_i,\bar{k},t_i)\delta k + \tfrac{1}{2}\delta x_i{}^T(\partial^2/\partial x^2)V(\bar{x}_i,\bar{k},t_i)\delta x_i + \delta x_i{}^T(\partial^2/\partial x\partial k)V(\bar{x}_i,\bar{k},t_i)\delta k + \tfrac{1}{2}\delta k^T(\partial^2/\partial k^2)V(\bar{x}_i,\bar{k},t_i)\delta k + \ldots \tag{7}$$

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 $\bar{W}(\bar{x}_i,\bar{k},t_i)$, the value of the return function calculated along the nominal trajectory, starting from t_i.

Symbolically,

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

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

However,

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

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 Eq. (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 $\bar{W}(\bar{x}_i, \bar{k},t_i)$, we define

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

To simplify notation, let

$$\bar{W}(\bar{x}_i,\bar{k},t_i) = \bar{W}^i,\ V(\bar{x}_i,\bar{k},t_i) = V^i$$
$$a(\bar{x}_i,\bar{k},t_i) = a^i,\ (\partial/\partial x)V(\bar{x}_i,\bar{k},t_i) = V_x{}^i, \text{ etc.}$$

Then

$$a^i = V^i - \bar{W}^i \tag{10'}$$

and using Eq. (10) in Eq. (7), we obtain

$$V(\bar{x}_i + \delta x_i,\bar{k} + \delta k,t_i) = a^i + \bar{W}^i + V_x{}^i\delta x_i + V_k{}^i\delta k + \tfrac{1}{2}\delta x^{iT}V_{xx}{}^i\delta x_i + \delta x_i{}^TV_{xk}{}^i\delta k + \tfrac{1}{2}\delta k^TV_{kk}{}^i\delta k + \ldots \tag{11}$$

Similarly, expanding the quantity to be minimized in Eq. (6) about $\bar{x}_i$, $\bar{k}$, $\bar{x}_{i+1}$,†† we obtain

$$L^i + L_x{}^i\delta x_i + \tfrac{1}{2}\delta x_i{}^TL_{xx}{}^i\delta x_i + a^{i+1} + \bar{W}^{i+1} + V_x{}^{i+1}\delta x_{i+1} + V_k{}^{i+1}\delta k + \tfrac{1}{2}\delta x_{i+1}{}^TV_{xx}{}^{i+1}\delta x_{i+1} + \delta x_{i+1}{}^TV_{xk}{}^{i+1}\delta k + \tfrac{1}{2}\delta k^TV_{kk}{}^{i+1}\delta k + \ldots \tag{12}$$

where, as above, $a^{i+1} + \bar{W}^{i+1} = V^{i+1}$

§ It is well known that Eq. (4) need only have a stationary point with respect to u, for fixed k for the original problem to have a solution. Here, attention is restricted to that class of problems where Eq. (4) has a minimum with respect to u, for fixed k. Problems not belonging to this class may be handled by other techniques; for example, the penalty function method of Kelley.[12]

¶ In some problems, typically continuous time, bang-bang with terminal constraints, $\partial V(x_i,k,t_i)/\partial x$ and $\partial^2(x_i,k,t_i)/\partial x^2$ are not continuous throughout the state space. Reference 3 extends Differential Dynamic Programming to that class of problems.

** The min H approach of Kelley et al.[14] is similar to this step.

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

Expression (12) is an infinite series in δx_i, δx_{i+1}, and δk; however, it is clear that there is a relationship between δx_i and δx_{i+1} through Eq. (1). This relationship may be used to eliminate either δx_i or δx_{i+1} from (12), but to conform with Eq. (11), δx_{i+1} is removed. Using the following equations:

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

we obtain

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

or,

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

In Eq. (13), u_i is perfectly general. It is chosen later by the minimization operation of Eq. (6). Expanding Eq. (13) about $\bar{x}_i$, and defining

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

we obtain

$$\delta x_{i+1} = (f^i - \bar{f}^i) + f_x{}^i\delta x_i + \tfrac{1}{2}\delta x_i{}^T f_{xx}{}^i\delta x_i + \ldots \tag{14}$$

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

Substituting Eq. (14) into Eq. (12), we obtain

$$\begin{aligned} &L^i + a^{i+1} + \bar{W}^{i+1} + V_x{}^{i+1}(f^i - \bar{f}^i) + \\ &\tfrac{1}{2}(f^i - \bar{f}^i)^T V_{xx}{}^{i+1}(f^i - \bar{f}^i) + [L_x{}^i + V_x{}^{i+1}f_x{}^i + \\ &f_x{}^{iT}V_{xx}{}^{i+1}(f^i - \bar{f}^i)]\delta x_i + [V_k{}^{i+1} + (f^i - \bar{f}^i)^T V_{xk}{}^{i+1}]\delta k + \\ &\delta x_i{}^T f_x{}^i V_{xk}{}^{i+1}\delta k + \tfrac{1}{2}\delta k^T V_{kk}{}^{i+1}\delta k + \tfrac{1}{2}\delta x_i{}^T[L_{xx}{}^i + \\ &V_x{}^{i+1}f_{xx}{}^i + f_x{}^{iT}V_{xx}{}^{i+1}f_x{}^i + (f^i - \bar{f}^i)^T V_{xx}{}^{i+1}f_{xx}{}^i]\delta x_i + \ldots \end{aligned} \tag{15}$$

Clearly Eq. (5) has now been transformed to:

$$\text{right-hand side of Eq. (11)} = \min_{u_i}\{\text{expression (15)}\} \tag{16}$$

As suggested earlier, the minimization in Eq. (16) is performed in two stages.

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

$$\begin{aligned} &L^i + a^{i+1} + \bar{W}^{i+1} + V_x{}^{i+1}(f^i - \bar{f}^i) + \\ &\tfrac{1}{2}(f^i - \bar{f}^i)^T V_{xx}{}^{i+1}(f^i - \bar{f}^i) + \ldots \end{aligned} \tag{17}$$

[The terms not printed in expression (17) are of third and higher order in $(f^i - \bar{f}^i)$, and are assumed to be negligible.]

For convenience, we define

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

In Eq. (18), and for the rest of this paper, all functions of u_i, other than $\bar{f}^i$, are evaluated at $u_i{}^*$.

Note that

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

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

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

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

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

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

[The third term in Eq. (20) does not appear in the "weak variation" algorithms of Refs. 4, 5, 8, and 11].

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

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

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

The second stage of the minimization is accomplished when expression (21) is minimized with respect to δu_i. Taking the first derivative of expression (21) with respect to δu_i and setting it to zero, we obtain

$$\delta u_i = \beta_1\delta x_i + \beta_2\delta k \tag{22}$$

where

$$\beta_1 = -\Delta^{-1}[H_{ux}{}^i + f_u{}^{iT}V_{xx}{}^{i+1}f_x{}^i + (f^i - \bar{f}^i)^T V_{xx}{}^{i+1}f_{ux}{}^i] \tag{23}$$

$$\beta_2 = -\Delta^{-1}f_u{}^{iT}V_{xk}{}^{i+1} \tag{24}$$

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 Eq. (22) into expression (21), we obtain

$$\begin{aligned} &L^i + a^{i+1} + \bar{W}^{i+1} + V_x{}^{i+1}(f^i - \bar{f}^i) + \\ &\tfrac{1}{2}(f^i - \bar{f}^i)^T V_{xx}{}^{i+1}(f^i - \bar{f}^i) + [H_x{}^i + (f^i - \\ &\bar{f}^i)^T V_{xx}{}^{i+1}f_x{}^i]\delta x_i + [V_k{}^{i+1} + (f^i - \bar{f}^i)^T V_{xk}{}^{i+1}]\delta k + \\ &\delta x_i{}^T[f_x{}^{iT}V_{xk}{}^{i+1} - \beta_1{}^T\Delta\beta_2]\delta k + \tfrac{1}{2}\delta x_i{}^T[H_{xx}{}^i + \\ &f_x{}^{iT}V_{xx}{}^{i+1}f_x{}^i + (f^i - \bar{f}^i)V_{xx}{}^{i+1}f_{xx}{}^i - \beta_1{}^T\Delta\beta_1]\delta x_i + \\ &\tfrac{1}{2}\delta k^T[V_{kk}{}^{i+1} - \beta_2{}^T\Delta\beta_2]\delta k \end{aligned} \tag{25}$$

Expression (25) is the minimum of (15) with respect to u_i. Thus, from Eq. (16), expression (25) is equal to the right-hand side of Eq. (11), so that coefficients of like powers of δx_i and δk are equal.

Noting that

$$\bar{W}^i = \bar{W}^{i+1} + \bar{L}^i \tag{26}$$

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

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

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

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

$$V_{xk}{}^i = f_x{}^{iT}V_{xk}{}^{i+1} - \beta_1{}^T\Delta\beta_2 \tag{30}$$

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

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

The boundary conditions are applied at $i = N$, and are the same as in Refs. 1 and 2. They are found by expanding

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

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

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

$$a^N = 0 \tag{33}$$

and,

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

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

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

$$V_{kk}{}^N = 0 \tag{37}$$

$$V_{xx}{}^N = F_{xx}(\bar{x}_N) + \bar{k}^T\theta_{xx}(\bar{x}_N) \tag{38}$$

Thus if we "integrate" Eqs. (27–32) from $i = N - 1$ to 0 with Eqs. (33–38) as boundary conditions, then Eqs. (19) and (22) show how to calculate $u_i = u_i{}^* + \delta u_i$ to obtain optimal improvement in 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. Note that 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

1. Comparison and Discussion

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

$$f(x_i,u_i,t_i) = x_i + \Delta t\tilde{f}(x_i,u_i,t_1) \tag{39}$$

and

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

Clearly,

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

and

$$\lim_{\substack{\Delta t\to 0\\ N\to\infty}}\sum_{i=0}^{N-1} L(x_i,u_i,t_i) = \lim_{\substack{\Delta t\to 0\\ N\to\infty}}\sum_{i=0}^{N-1}\tilde{L}(x_i,u_i,t_i)\,\Delta t = \int_{t_0}^{tN}\tilde{L}\,[x(t),u(t),t]dt \tag{42}$$

if the discretization is done with care.

It is reasonable to expect that if 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[1,2] has solved that problem, and the statement and solution of the problem are reproduced in Appendix A.

Note that

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

so that

$$H^i = (\tilde{L}^i + V_x{}^{i+1}\tilde{f}^i)\Delta t + V_x{}^{i+1}\bar{x}_i = \tilde{H}^i\Delta t + V_x{}^{i+1}\bar{x}_i \tag{43}$$

Then, according to Eq. (20)

$$\Delta = \tilde{H}_{uu}{}^i\Delta t + (\Delta t)^2[\tilde{f}_u{}^iV_{xx}{}^{i+1}\tilde{f}_u{}^i + (\tilde{f}^i - \bar{\tilde{f}}^i)V_{xx}{}^{i+1}\tilde{f}_{uu}{}^i] \tag{44}$$

Now define

$$\tilde{\Delta} = \Delta/\Delta t \tag{45}$$

which we write as

$$\tilde{\Delta} = \tilde{H}_{uu}{}^i + A^i\Delta t \tag{46}$$

for clarity.

From Eqs. (23) and (45),

$$\beta_1 = -\tilde{\Delta}^{-1}(\tilde{H}_{ux}{}^i + \tilde{f}_u{}^{iT}V_{xx}{}^{i+1}) - \tilde{\Delta}^{-1}[\tilde{f}_u{}^{iT}V_{xx}{}^{i+1}\tilde{f}_x{}^{i+1} + (\tilde{f}^i - \bar{\tilde{f}}^i)^TV_{xx}{}^{i+1}f_{ux}{}^i]\Delta t \tag{47}$$

Similarly, from Eq. (24),

$$\beta_2 = -\tilde{\Delta}^{-1}\tilde{f}_u{}^{iT}V_{xk}{}^{i+1} \tag{48}$$

In the same manner, applying Eqs. (39, 40, 43, 45, 47, and 48) to (27–32), the following equations are obtained:

$$-(a^{i+1} - a^i)/\Delta t = \tilde{H}^i - \bar{\tilde{H}}^i + \tfrac{1}{2}(\tilde{f}^i - \bar{\tilde{f}}^i)^TV_{xx}{}^{i+1}(\tilde{f}^i - \bar{\tilde{f}}^i)\Delta t \tag{49}$$

$$-(V_x{}^{i+1} - V_x{}^i)/\Delta t = \tilde{H}_x{}^i + (\tilde{f}^i - \bar{\tilde{f}}^i)V_{xx}{}^{i+1} + (\tilde{f}^i - \bar{\tilde{f}}^i)^TV_{xx}{}^{i+1}\tilde{f}_x{}^i\Delta t \tag{50}$$

$$-(V_k{}^{i+1} - V_k{}^i)/\Delta t = (\tilde{f}^i - \bar{\tilde{f}}^i)^TV_{xk}{}^{i+1} \tag{51}$$

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

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

$$-(V_{xx}{}^{i+1} - V_{xx}{}^i)/\Delta t = \tilde{H}_{xx}{}^i + \tilde{f}_x{}^{iT}V_{xx}{}^{i+1} + V_{xx}{}^{i+1}\tilde{f}_x{}^i + \beta_1{}^T\tilde{\Delta}\beta_1 + \Delta t\,[\tilde{f}_x{}^{iT}V_{xx}{}^{i+1}\tilde{f}_x{}^i + (\tilde{f}^i - \bar{\tilde{f}}^i)^TV_{xx}{}^{i+1}\tilde{f}_{xx}{}^i] \tag{54}$$

Jacobson's equations for β_1, β_2, a, V_x, V_k, V_{xk}, V_{kk}, and V_{xx} are reproduced in Appendix A. Inspection will reveal agreement between those and Eqs. (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 Eqs. (39, 40, and 43), the discrete a, $\overline{W}$, 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 have been used previously to show the connection between the present discrete equations and the earlier[1,2] continuous equations. However, cases may exist where Eqs. (39) and (40) are useful numerical methods with which to solve a continuous problem.§§ Then, Eqs. (47–54) contain the full dependence on Δt, which involves terms of order Δt and higher. It may be worthwhile to retain high-order terms.[11] Also, Eqs. (47–54) indicate that some of the arguments of the right sides are evaluated at time $i + 1$, and others are evaluated at time i. A simple Euler discretization of the continuous time algorithm would evaluate all arguments at time $i + 1$.

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

2. Description of the Algorithm

The discrete algorithm is very similar to the continuous algorithm and is outlined in Fig. 1. 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.

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

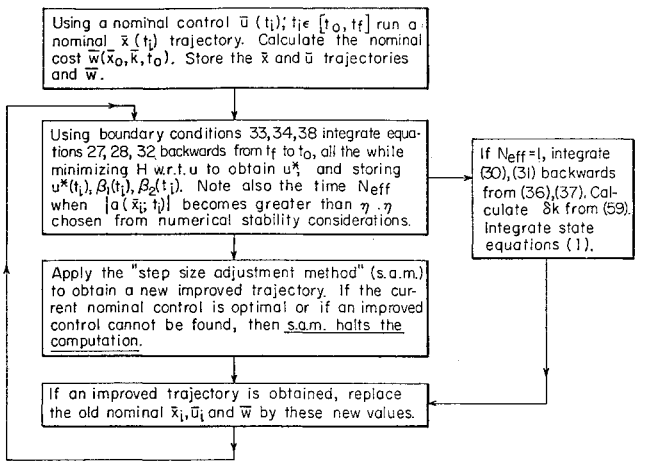

Fig. 1 Over-all computational procedure.

The first stage proceeds as follows. Equation (1) is "integrated" using initial conditions x_o and nominal control $\bar{u}_0, \ldots, \bar{u}_{N-1}$. Then Eqs. (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), Eq. (19) is solved for u_i^* and Eq. (22) is used to calculate $u_i = u_i^* + \delta u_i$, which is used as the new optimal control in Eq. (1). The cycle repeats. If necessary (see below for the descriptions of the tests to explain this necessity) the step-size adjustment routine is called. For a complete description of this routine, see Appendix B and Ref. 2.

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.

3. Determination of δk

Jacobson has shown (Ref. 1, Sec. 2.3) that, to second order, the proper value of δk is that which maximizes $V(\bar{x}_o, \bar{k} + \delta k, t_o)$.¶¶ We have

$$V(\bar{x}_0, \bar{k} + \delta k, t_0) = a^o + \bar{W}^o + V_{k^o}{}^T \delta k + \tfrac{1}{2}\delta k^T V_{kk^o} \delta k \quad (55)$$

so that the proper value of δk satisfies

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

or

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

(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 Eqs. (29) and (35). Thus, Eq. (57) becomes

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

Following Jacobson,[1] k is modified according to Eq. (58); Eq. (1) is then integrated forward with $u_i = u_i^* + \delta u_i$ chosen according to Eq. (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

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

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

4. New Criteria

It is essential that δx_i and δk be kept small. This ensures that δu_i is small, and thus that the second-order expansions of Eq. (6) 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 Sec. II, they should be reduced by Jacobson's techniques which apply to the discrete problem as well as to the continuous.

An additional criterion, required for fixed end point problems is described below (test 1). A criterion, alternative to that in Refs. 1 and 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.†

Test 1

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

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

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

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

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

Test 2

From Eqs. (4) and (9),

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

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

Thus

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

But, from Eq. (11),

$$\delta V^i = a^i + V_x{}^i \delta x_i + V_k{}^i \delta k + \tfrac{1}{2}\delta x_i{}^T V_{xx}{}^i \delta x_i + \delta x_i{}^T V_{xk}{}^i \delta k + \tfrac{1}{2}\delta k^T V_{kk}{}^i \delta k \quad (64)$$

Since Eqs. (63) and (64) are theoretically equal, their difference is a test of the size of δx_i and δk; this is because Eq. (63) is an exact expression, and Eq. (64) is an approximation, correct to second-order terms.

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

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

¶¶ McReynolds[4] and Bryson and Ho[15] have obtained similar conditions.

* Provided that the linearized system is controllable, and $\theta_x{}^T$ has full rank.

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

Table 1 100 time steps — final time = 3.32[a]

t	u	x_1	x_2	x_3	V_{x1}	V_{x2}	V_{x3}
0	0.4430	1	0	1	1.8890	0.94316	2.0604
0.166	0.5188	1.0008	0.0134	1.0201			
0.332	0.6073	1.0044	0.0353	1.0366	1.5777	0.94982	1.4229
0.498	0.7097	1.0121	0.0649	1.0478			
0.664	0.8269	1.0252	0.1011	1.0520	1.2963	0.85152	0.82311
0.830	0.9592	1.0446	0.1419	1.0479			
0.996	1.107	1.0711	0.1853	1.0349	1.0857	0.64554	0.34816
1.162	1.271	1.1047	0.2288	1.0129			
1.328	1.460	1.1455	0.2701	0.9823	0.96818	0.36222	0.055367
1.494	1.730	1.1929	0.3071	0.9433			
1.660	2.886	1.2459	0.3347	0.8924	0.93356	0.045050	−0.048480
1.826	4.493	1.3008	0.3157	0.8370			
1.992	4.765	1.3508	0.2786	0.8032	0.95639	−0.27469	0.0036793
2.158	4.913	1.3945	0.2390	0.7811			
2.324	5.023	1.4315	0.1991	0.7675	1.0117	−0.58597	0.17505
2.490	5.116	1.4619	0.1600	0.7611			
2.656	5.196	1.4860	0.1225	0.7609	1.1031	−0.88384	0.44689
2.822	5.269	1.5039	0.0872	0.7661			
2.988	5.335	1.5162	0.0546	0.7762	1.2105	−1.1608	0.81108
3.154	5.398	1.5233	0.0254	0.7908			
3.320	...	1.5257	0.0000	0.8096	1.3356	−1.4034	1.2647

[a] $V = 1.52572699$, $k_1 = -1.40339248$, $k_2 = 1.26501024$, $\theta_1 = 0.75 \times 10^{-6}$, $\theta_2 = 0.11 \times 10^{-5}$.

Thus,

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

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

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

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

$$\delta V^i - \delta V^o = a^i - a^o + V_x{}^i\delta x_i + \tfrac{1}{2}\delta x_i{}^T V_{xx}{}^i\delta x_i + \delta x_i{}^T V_{xk}{}^i\delta k + \tfrac{1}{2}\delta k^T V_{kk}{}^i\delta k - \tfrac{1}{2}\delta k^T V_{kk}{}^o\delta k \tag{68}$$

and, test 2 is performed by determining whether Eq. (66) agrees with Eq. (68) within a given tolerance. If the test is failed‡ then δk should be reduced, or, if δk is zero, δx_i should be reduced by the step-size adjustment method. This test is particularly simple to apply in cases where $L(x_i,u_i,t_i) \equiv 0$.

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

1. Statement of the Orbit Transfer Problem

An orbit transfer problem[4,5,7,12,16] is 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. Here, 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

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

$$\theta(x_N) = \begin{pmatrix} x_{2,N} \\ x_{3,N} - \dfrac{1}{(x_{1,N})_{1/2}} \end{pmatrix}$$

($\theta = 0$ is the condition for a state to be in a stable orbit.)

‡ 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.

Here,

$$\tilde{L}^i = 0, \; F(x_N) = x_{1,N}$$

so that

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

$$\tilde{f}^i = \begin{pmatrix} x_{2,i} \\ x_3{}^2{}_{,i}/x_{1,i} - 1/x_1{}^2{}_{,i} + A^i \sin u_i \\ -x_{2,i}x_{3,i}/x_{1,i} + A^i \cos u_i \end{pmatrix}$$

where

$$A^i = 0.1405/(1.0 - 0.07487t_i)$$

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

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

This statement of the problem was inserted into Eqs. (46–54) with terms of order higher than Δt dropped. The relevant equations are

$$\tilde{\Delta} = \tilde{H}_{uu}{}^i = V_x{}^{i+1}G_{uu}{}^i \tag{69}$$

$$\beta_1 = -\tilde{\Delta}^{-1}\tilde{f}_u{}^{iT}V_{xx}{}^{i+1} \tag{70}$$

$$\beta_2 = -\tilde{\Delta}^{-1}\tilde{f}_u{}^{iT}V_{xk}{}^{i+1} \tag{71}$$

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

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

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

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

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

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

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

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

The following necessary condition holds:

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

Table 2 400 time steps — final time = 3.32[a]

t	u	x_1	x_2	x_3	V_{x1}	V_{x2}	V_{x3}
0	0.4332	1	0	1	1.8803	0.93239	2.0340
0.166	0.5072	1.0010	0.0139	1.0200	1.7254	0.94700	1.7201
0.332	0.5937	1.0049	0.0361	1.0362	1.5729	0.93843	1.4045
0.498	0.6936	1.0132	0.0659	1.0470	1.4273	0.90323	1.0982
0.664	0.8080	1.0269	0.1020	1.0507	1.2942	0.83985	0.81261
0.830	0.9371	1.0469	0.1425	1.0464	1.1790	0.74911	0.55832
0.996	1.081	1.0740	0.1855	1.0332	1.0857	0.63411	0.34405
1.162	1.241	1.1082	0.2285	1.0114	1.0160	0.49963	0.17548
1.328	1.426	1.1493	0.2693	0.9811	0.96936	0.35134	0.054908
1.494	1.683	1.1970	0.3059	0.9428	0.94344	0.19473	−0.018536
1.660	2.645	1.2502	0.3335	0.8927	0.93488	0.034410	−0.048200
1.826	4.437	1.3048	0.3149	0.8375	0.94036	−0.12632	−0.039267
1.992	4.732	1.3542	0.2780	0.8039	0.95720	−0.28580	0.0028600
2.158	4.885	1.3972	0.2386	0.7818	0.98321	−0.44323	0.074425
2.324	4.999	1.4337	0.1988	0.7682	1.0168	−0.59804	0.17286
2.490	5.093	1.4636	0.1598	0.7617	1.0569	−0.74962	0.29642
2.656	5.176	1.4872	0.1224	0.7613	1.1029	−0.89717	0.44398
2.882	5.250	1.5047	0.0871	0.7664	1.1541	−1.0396	0.61486
2.988	5.318	1.5165	0.0546	0.7765	1.2102	−1.1755	0.80871
3.154	5.382	1.5232	0.0254	0.7910	1.2709	−1.3029	1.0253
3.320	...	1.5254	0.0000	0.8097	1.3356	−1.4194	1.2646

[a] $V = 1.52537493$, $k_1 = -1.41936325$, $k_2 = 1.26460750$, $\theta_1 = -0.33 \times 10^{-6}$, $\theta_2 = 0.37 \times 10^{-7}$.

whence,

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

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

$$\tilde{H}^i(\bar{x}_i + \delta x_i, u_i, V_x{}^{i+1} + \delta x_{i+1}{}^T V_{xx}{}^{i+1} + \delta k^T V_{kx}{}^{i+1}) = (V_x{}^{i+1} + \delta x_{i+1}{}^T V_{xx}{}^{i+1} + \delta k^T V_{kx}{}^{i+1})[\tilde{F}(\bar{x}_i + \delta x_i) + G^i(u_i)] \tag{79}$$

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

$$\delta x_{i+1} = \delta x_i + \Delta t\{[G^i(u_i) - G^i(\bar{u}_i)] + \tilde{F}_x(\bar{x}_i)\delta x_i + \tfrac{1}{2}\delta x_i{}^T \tilde{F}_{xx}(\bar{x}_i)\delta x_i\} \tag{80}$$

However, this is of higher order than the degree of approximation, and it is satisfactory to replace δx_{i+1} in Eq. (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 is given in Sec. V.

2. Comparison with Successive Sweep Method

Our algorithm converges somewhat faster than McReynolds' Successive Sweep Method[4-6] on this problem, starting from the same initial nominal solution. This may be because the two techniques differ primarily in the minimization,§ and the $f^i - \tilde{f}^i$ and $H^i - \tilde{H}^i$ terms which are present here, but are absent from the Successive Sweep Method. However, close to the optimal, these terms are small, so that our minimization yields results which are close to McReynolds'. 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 reach the vicinity of the optimal in fewer iterations than the Successive Sweep Method, and once there, takes just as many additional iterations to converge.

Note that this routine does not evaluate H_{uu} (or Δ) until after a minimization has been performed, so that H_{uu} is 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

1. Discussion of the Trajectories in Tables 1–4

Tables 1–3 list optimal trajectories calculated for the orbit-transfer problem obtained by means of the algorithm described previously.

The value of 3.32 was used for t_N in order to compare results with those described in Refs. 4 and 5. The other value, 3.3194, was determined in Ref. 16, where the authors solved a minimum time problem. Their 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 Ref. 16. (The normalized values of $V_x{}^o$ agree with $\lambda(t_o)$ given in Ref. 16 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.

Table 4 contains a trajectory which maximizes W 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 W is far from the maximum W obtained in Tables 1–3, and θ is not zero. Thus the free-end point problem, with k_1 and k_2 set to their optimal values, has at least two local maxima; 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 $|\theta|$, 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 W is attained and $\theta = 0$.)

On the average, the program took approximately 3 sec per iteration for the 100-step program and 12 sec per iteration

§ Which becomes a maximization in this problem.

¶ $-|H_{uu}{}^i + B^i|$ is used in place of $H_{uu}{}^i$ where B^i is chosen to go to zero as the nominal is approached.

Table 3 400 time steps — final time = 3.3194[a]

t	u	x_1	x_2	x_3	V_{x1}	V_{x2}	V_{x3}
0	0.4333	1	0	1	1.8800	0.93244	2.0334
0.166	0.5074	1.0010	0.0139	1.0199	1.7252	0.94699	1.7196
0.332	0.5938	1.0049	0.0361	1.0362	1.5727	0.93838	1.4041
0.498	0.6937	1.0131	0.0658	1.0470	1.4272	0.90314	1.0979
0.664	0.8080	1.0268	0.1019	1.0507	1.2941	0.83974	0.81232
0.830	0.9372	1.0469	0.1425	1.0464	1.1790	0.74899	0.55811
0.996	1.081	1.0739	0.1854	1.0332	1.0856	0.63399	0.34391
1.162	1.242	1.1081	0.2284	1.0114	1.0160	0.49953	0.17540
1.328	1.426	1.1493	0.2692	0.9812	0.96934	0.35127	0.054860
1.494	1.683	1.1969	0.3059	0.9429	0.94343	0.19468	−0.018561
1.660	2.645	1.2501	0.3334	0.8927	0.93487	0.034386	−0.048213
1.826	4.437	1.3046	0.3148	0.8375	0.94035	−0.12631	−0.039278
1.992	4.732	1.3540	0.2779	0.8040	0.95720	−0.28576	0.0028443
2.158	4.885	1.3971	0.2386	0.7819	0.98321	−0.44317	0.074400
2.324	4.999	1.4335	0.1988	0.7682	1.0168	−0.59796	0.17282
2.490	5.093	1.4634	0.1598	0.7617	1.0569	−0.74951	0.29636
2.656	5.176	1.4870	0.1223	0.7614	1.1029	−0.89703	0.44390
2.821	5.250	1.5045	0.0871	0.7665	1.1541	−1.0394	0.61476
2.987	5.318	1.5163	0.0546	0.7765	1.2103	−1.1753	0.80858
3.153	5.382	1.5230	0.0254	0.7911	1.2709	−1.3026	1.0252
3.319	...	1.5252	0.0000	0.8097	1.3357	−1.4191	1.2644

[a] $V = 1.52516085$, $k_1 = -1.41910912$, $k_2 = 1.26441935$, $\theta_1 = -0.10 \times 10^{-5}$, $\theta_2 = -0.26 \times 10^{-6}$.

for the 400-step program on an IBM 7094 computer. For this purpose, the "number of iterations" is defined as the number of times Eqs. (27, 28, and 32) were integrated. Thus an iteration includes at least one but possibly as many as 9 integrations of the state Eqs. (1). Also, an iteration may include the integration of Eqs. (30) and (31) and calculation of δk by Eq. (59). In the earlier versions of the program, where test 2 was absent, iteration times averaged as much as 6 sec 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 $0 \leqslant t \leqslant 1.66$; $\bar{u}(t) = 5.7124$ for $1.66 < t < 3.32$. Convergence to $|\theta_i(x_N)| < 10^{-6}$ $(i = 1,2)$ required 15 iterations.

The nominal used for Tables 2 and 3 is the optimal control computed in Ref. 5. (This 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.

Starting conditions for Table 4 are the nominal control used in Table 1, and the values of $\bar{k}_1$ and $\bar{k}_2$ used in Table 3. "Convergence" required 6 iterations.

2. Use 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.

Two programs were compared; one used only test 1, the other used both tests. The comparison indicates 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 were used.) In fact, the same values of δk were ultimately accepted by the two pro-

Table 4 100 time steps — final time = 3.32[a]

t	u	x_1	x_2	x_3	V_{x1}	V_{x2}	V_{x3}
0	1.147	1	0	1	1.5399	3.8503	1.5607
0.166	1.458	1.0015	0.0233	1.0055			
0.332	1.802	1.0071	0.0489	0.9993	0.065221	3.4865	−1.1789
0.498	2.172	1.0167	0.0706	0.9812			
0.664	2.423	1.0294	0.0826	0.9529	−1.2528	2.6667	−3.5848
0.830	2.674	1.0433	0.0833	0.9200			
0.996	2.819	1.0565	0.0722	0.8850	−2.0365	1.7928	−5.1481
1.162	2.917	1.0672	0.0508	0.8525			
1.328	2.990	1.0738	0.0203	0.8221	−2.4128	1.0467	−5.9689
1.494	3.048	1.0747	−0.0184	0.7956			
1.660	3.098	1.0686	−0.0650	0.7737	−2.6385	0.39794	−6.2592
1.826	3.143	1.0543	−0.1194	0.7572			
1.992	3.186	1.0305	−0.1818	0.7472	−2.7998	−0.14885	−6.0700
2.158	3.229	0.9957	−0.2532	0.7452			
2.324	3.275	0.9484	−0.3349	0.7536	−2.9268	−0.59906	−5.3983
2.490	3.331	0.8867	−0.4294	0.7765			
2.656	3.406	0.8083	−0.5404	0.8211	−2.9898	−0.97790	−4.1832
2.822	3.528	0.7100	−0.6738	0.9021			
2.988	3.778	0.5878	−0.8374	1.0534	−2.6070	−1.3288	−2.2274
3.154	4.473	0.4361	−1.0316	1.3731			
3.320	...	0.2559	−1.0621	2.2275	5.8682	−1.3996	1.2600

[a] $V = 2.05800182$, $k_1 = -1.3996310$, $k_2 = 1.2600310$, $\theta_1 = -1.0620840$, $\theta_2 = 0.25048833$.

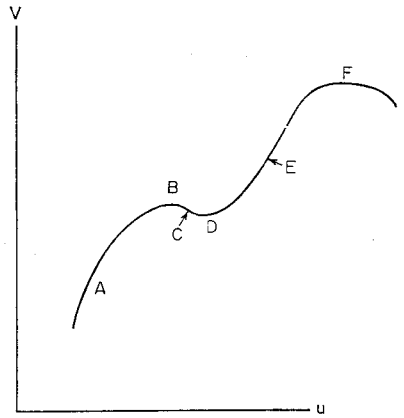

Fig. 2 Multimodal V-function.

grams, 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.

Note that this redundancy is helpful. Test 2 can be invoked often in the forward integration phare, whereas 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 sec. With only test 1, a 100-step iteration took, on the average, more than 6 sec. (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 became small. Tests 1 and 2, which involve differences of large quantities, became less reliable—in fact, excessively conservative. In order to overcome this difficulty, test 1 was disabled when $\delta V^o = V^o - \overline{W}^o$ was less, in absolute value, than $10^{-6}\overline{W}^o$, and test 2 was disabled when the absolute value of

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

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

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 Fig. 2.)

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

$$V(u) \approx V(\bar{u}) + V'(\bar{u})(u - \bar{u}) + \tfrac{1}{2}V''(\bar{u})(u - \bar{u})^2 \quad (81)$$

The value of u that maximizes this is given by

$$0 = V'(\bar{u}) + V''(\bar{u})(u - \bar{u})$$

or

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

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

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

which is positive for $V''(u) < 0$.

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 Eq. (82) will produce a value of u in the neighborhood of B. A value near E will eventually converge to F. Thus, Eq. (82) should be replaced by

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

Then, Eq. (83) becomes

$$V(u) - V(\bar{u}) = (\tfrac{1}{2}\epsilon^2 - \epsilon)V'(\bar{u})^2/V''(\bar{u}) \quad (85)$$

Thus an improvement is guaranteed at every stage if Eq. (84) is used with proper choice of ϵ, and 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, Eqs. (84) and (85) are not useable.

The aforementioned "Newton-Raphson" algorithm is not perfectly analogous to the algorithm for discrete-time dynamic optimization algorithms, but some comparisons can 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 $\theta_1 = \theta_2 = 0$ (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 Refs. 5, 6, 8, and 11), 3) test 2.

VI. Conclusion

A new discrete algorithm has been derived which is analogous to the continuous algorithms of Refs. 1 and 2. Extensions to the latter (test 1 and test 2) have been developed to ensure that the new iteration is in the neighborhood of the current nominal. The algorithm has been used to solve a nonlinear, optimal orbit transfer problem. This problem has been attempted, and solved, in various forms, by a number of investigators using different computational methods. The numerical results obtained in this paper agree most closely with those of Ref. 16.

A more detailed description of the numerical solution of the optimal orbit transfer problem is given in Ref. 17. Tables of the second-partial-derivatives of V are given, and the optimal feedback gains are listed.

Appendix A: Continuous Results from Jacobson

The following is a statement and solution of the continuous-time optimal control problem solved in Refs. 1 and 2. The notation has been modified to conform to that of this paper. Thus some expressions involving derivatives

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

have been transposed, and $\sim$ has been placed over certain symbols to coincide with Sec. III.1.

Given that

$$\dot{x} = \tilde{f}(x,u,t); \quad x(t_0) = x_0 \tag{A1}$$

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

$$\hat{W}(x_0,t_0) = \int_{t_0}^{t_f} \tilde{L}(x,u,t)dt + F[x(t_f)] \tag{A2}$$

while satisfying

$$\theta[x(t_f)] = 0 \tag{A3}$$

The constraints (A3) are adjoined to the cost functional (A2):

$$W(x_0,k,t_0) = \hat{W} + k^T\theta[x(t_f)] \tag{A4}$$

The relevant equations are

$$\beta_1 = -\tilde{H}_{uu}^{-1}(\tilde{H}_{ux} + \tilde{f}_u{}^T V_{xx}) \tag{A5}$$

$$\beta_2 = -\tilde{H}_{uu}^{-1}\tilde{f}_u{}^T V_{xk} \tag{A6}$$

$$-\dot{a} = \tilde{H} - \bar{\tilde{H}} \tag{A7}$$

$$-\dot{V}_x = \tilde{H}_x + (\tilde{f} - \bar{\tilde{f}})V_{xx} \tag{A8}$$

$$-\dot{V}_k = (\tilde{f} - \bar{\tilde{f}})V_{xk} \tag{A9}$$

$$-\dot{V}_{xk} = (\tilde{f}_x{}^T + \beta_1{}^T\tilde{f}_u{}^T)V_{xk} \tag{A10}$$

$$-\dot{V}_{kk} = -V_{xk}{}^T\tilde{f}_u\tilde{H}_{uu}{}^{-1}\tilde{f}u^T V_{xk} \tag{A11}$$

$$-\dot{V}_{xx} = \tilde{H}_{xx} + \tilde{f}_x{}^T V_{xx} + V_{xx}\tilde{f}_x - (\tilde{H}_{ux} + f_u{}^T V_{xx})^T\tilde{H}_{uu}{}^{-1}(H_{ux} + f_u{}^T V_{xx}) \tag{A12}$$

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 for Eqs. (A7–A12) are the same as Eqs. (33–38).

Appendix B: Step-Size Adjustment Method

In the first stage of the algorithm (Sec. III.2) k is kept constant. Then Eq. (22), the linear feedback law, becomes

$$\delta u_i = \beta_i\delta x_i \tag{B1}$$

and the perturbed state Eq. (1) becomes

$$\bar{x}_{i+1} + \delta x_{i+1} = f(\bar{x}_i + \delta x_i, u_i{}^* + \beta_1\delta x_i\, t_i) \tag{B2}$$

with

$$\bar{x}(t_0) + \delta x(t_0) = x_0$$

Since $\delta x(t_o) = 0$, δx_i produced by Eq. (B2) is due to the driving action of $\delta u_i{}^* = u^*{}_i - \bar{u}_i$.

The size of δx_i can be restrained by altering the time interval over which Eq. (B2) is evaluated. Consider the time interval $[t_{N_1},t_N]$ where $0 \leq N_1 < N$. Assume one runs along the nominal trajectory $\bar{x}_i$ from t_o to t_{N_1}. At time $t_i = t_{N_1}$, $x(t_{N_1}) = \bar{x}(x_{N_1})$ since the path of the nominal trajectory has been followed from t_o to t_{N_1} (i.e., δx_i; $i\epsilon[0,N_1]$ is zero). Now consider integrating (B2) over the interval $[t_{N_1},t_N]$. If $[t_{N_1},t_N]$ is small, then δx_i produced by (B2) in this interval will be small, even for large $\delta u_i{}^*$, since there are very few steps over which to sum (B2) with initial conditions

$$\delta x(t_{N_1}) = 0$$

By making N_1 near N, one can generally force δx_i to be sufficiently small.

This description is summarized in the following statement. There exists a time t_{N_1}, sufficiently close to t_N, in the range $t_o \leq t_{N_1} < t_N$, such that if the nominal trajectory is followed from t_o to t_{N_1} and then (B2) is integrated from t_{N_1} to t_N, δx_i, in the interval $[t_{N_1},t_N]$ will be small enough for the second-order expansions of V, L, and f to be valid.

The following questions must be answered. 1) How does one decide if δx_i is "small enough?" 2) How does one chose N_1 such that δx_i is "small enough?"

The answers are: 1) From Eq. (27),

$$|a^{N_1}| = \left|\sum_{i=N_1}^{N-1} H^i - \bar{H}^i + \tfrac{1}{2}(f^i - \bar{f}^i)^T V_{xx}(f^i - \bar{f}^i)\right|$$

is the predicted improvement in cost when starting at the point $\bar{x}_{N_1}$ and using $u_i = u_i{}^* + \beta_1\,\delta x_i$.

Assume for the moment that $N_1 = 0$. Integrate Eq. (B2) and calculate the cost W. The actual improvement in cost is

$$\Delta V = \bar{W}(\bar{x}_0,\bar{k},t_0) - W(\bar{x}_0,\bar{k},t_0) \tag{B3}$$

If this actual improvement in cost is "near" to the predicted value $|a^{N_1}|$ then δx_i, produced by the new control acting through Eq. (B2), is considered "small enough."

It is convenient in practice to define "near" in the following way. If the following inequality is satisfied, ΔV is considered to be "near" $|a^{N_1}|$:

$$\Delta V/|a^{N_1}| > c; \quad c \geq 0 \tag{B4}$$

In practice c is set as 0.5. There are no hard and fast rules for setting c. Certainly it should be greater than or equal to zero since a negative ΔV is inadmissible. c should not be greater than unity since one should not expect improvement in cost greater than predicted, if the expansions for V, L and f are valid. Moreover, c should be somewhat less than unity so that any decisions based on Eq. (B4) are not influenced by round-off errors in the computations.

A more precise criterion which is relevant here is test 2, described in Sec. III.4. Test 2 was used in addition to Eq. (B4) in the program that produced Tables 1–4.

2) If test (B4) is passed with $N_1 = 0$, all is well, and the next iteration of the main algorithms may be begun with the knowledge that a reasonable reduction in cost of ΔV has been made. If (B4) is not satisfied, then set

$$N_1 = N/2 = N_{o1} \tag{B5}$$

The aforementioned procedure is repeated with this N_1 and Eq. (B4) is checked again (with the new ΔV). If it is satisfied, then the next iteration is begun. If not, then set

$$N_1 = (N - N_{o1})/2 + N_{o1} = N_{o2} \tag{B6}$$

and repeat again.

In general

$$N_1 = (N - N_{or})/2 + N_{or} = N_{or+1} \tag{B7}$$

where $r = 0,1 \ldots$ and $N_{oo} = -N$.

It may happen that the nominal trajectory $\bar{x}_i$ is optimal in an interval $[N_2,N]$; $N_2\epsilon[0,N]$, but is nonoptimal in the interval $[0,N]$. If N_1 is being found in the manner outlined above, then some N_{or} may fall in the interval $[N_2,N]$. Then δx_i in the interval $[N_1,N]$ would be zero because $u_i{}^* = \bar{u}_i$; $i\epsilon[N_2,N]$, so that no reduction ΔV in cost would occur, even though the whole trajectory $\bar{x}_i$ is nonoptimal in $[0,N]$. One must insure, therefore, that N_1 will never fall in $[N_2,N]$. This is accomplished easily in the following way.

From Eq. (33), $a^N = 0$. When integrating the backwards equations monitor $|a^i|$. Record the step N_{eff} when $|a^i|$ becomes different from zero. (Or in practice, when it becomes greater than a small, positive quantity, η.) The trajectory between N_{eff} and N satisfies a necessary condition of optimality,

viz.,

$$a^i = 0, i = N_{eff}, \ldots, N \tag{B8}$$

If, on the forwards run, a time $N_1 \neq 0$ must be found, then the time interval $[0,N_{eff}]$ is subdivided as described earlier, and not $[0,N]$. As the over-all trajectory becomes more and more optimal, from iteration to iteration, so $N_{eff} \to 0$. Finally, on an optimal trajectory, $|a^i| < \eta$; $i = 0, \ldots, N$ and $N_{eff} = 0$ and the computation is stopped.

Thus, Eq. (B7) becomes

$$N_1 = (N_{eff} - N_{or})/2 + N_{or} = N_{or+1} \tag{B9}$$

where $N_\infty = 2 - N_{eff}$ and $r = 0,1 \ldots$. Integer division is used in (B7) and (B9) and r is increased until $N_1 = N_{eff} - 1$. If $N_{eff} = 1$ then only $r = 0$ is used.

It should be appreciated that since there are a finite number $N - 1$ of discrete time steps, this subdivision can only be done a finite number of times. For problems like those of Section III.1, the smallest possible nonzero time interval is $\Delta t = (t_f - t_o)/(N - 1)$. It is clear that N must be large enough such that δx_i produced during this basic interval is "small enough." This restriction is a practical one, brought about by the discrete time nature of the digital computation.

When ΔV and $|a^{N_1}|$ are small, but $> \eta$, the criterion (B4) may be too severe with $c = 0.5$, owing to round-off errors. There may thus come a stage where (B4) remains unsatisfied even when $N_1 = N_{eff} - 1$. If this happens, set $c = 0.0$ and repeat the procedure for determining N_1. $c = 0$ is a much less stringent test because it asks only that $\Delta V > 0$. If once again (B4) is unsatisfied, even when $N_1 = N_{eff} - 1$, then stop the computation since no further reduction in cost is possible. This implies that either optimality has been attained (in which case $|a^o| < \eta$ and so $N_{eff} < 1$) or N is not large enough and hence Δt is too large a basic time interval. For additional refinements of the step-size adjustment method see Ref. 1, Chap. 4.

References

[1] Jacobson, D. H. and Mayne, D. Q., *Differential Dynamic Programming*, American Elsevier, New York, 1970.

[2] Jacobson, D. H., "New Second Order and First Order Algorithms for Determining Optimal Control: A Differential Dynamic Programming Approach," *Journal of Optimization Theory and Applications*, Vol. 2, No. 6, 1968, pp. 411–440.

[3] Jacobson, D. H., "Differential Dynamic Programming Algorithms for Solving Bang-Bang Control Problems," *IEEE Transactions on Automatic Control*, AC-13, Dec. 1968, pp. 661–675.

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

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

[6] McReynolds, S. R. and Bryson, A. E., Jr., "A Successive Sweep Method for Solving Optimal Programming Problems," *Joint Automatic Control Conference*, Vol. 6, 1965, pp. 551–555.

[7] Kelley, H. J., Kopp, R. E., and Moyer, H. G., "A Trajectory Optimization Technique Based Upon the Theory of the Second Variation," *Progress in Astronautics and Aeronautics*, Vol. 14, 1964, pp. 559–582.

[8] Mayne, D. Q., "A Second Order Gradient Method of Optimizing Non-Linear Discrete-Time Systems," *International Journal of Control*, Vol. 3, 1966, pp. 85–95.

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

[10] Bullock, T. E. and Franklin, G. F., "A Second-Order Feedback Method for Optimal Control Computations," *IEEE Transactions on Automatic Control*, AC-12, 1967, p. 666.

[11] Dreyfus, S. E., "The Numerical Solution of Non-Linear Optimal Control Problems," *Numerical Solutions of Non-Linear Differential Equations*, edited by D. Greenspan, Wiley, New York, 1966, pp. 97–113.

[12] Kelley, H. J., "Method of Gradients," *Optimization Techniques*, edited by G. Leitman, Academic Press, New York, 1961, pp. 205–254.

[13] Bellman, R. and Kalaba, R., *Dynamic Programing and Modern Control Theory*, Academic Press, New York, 1965.

[14] Kelley, H. J., Kopp, R. E., and Moyer, H. G., "Successive Approximation Techniques for Trajectory Optimization," *IAS Vehicle Systems Optimization Symposium*, Garden City, 1961, pp. 10–25.

[15] Bryson, A. E. and Ho, Y. C., *Applied Optimal Control*, Blaisdell, Waltham, Mass., 1969, Chaps. 5–7.

[16] Tapley, B. D. and Lewallen, J. M., "Comparison of Several Numerical Optimization Methods," *Journal of Optimization Theory and Applications*, Vol. 1, No. 1, July 1967, pp. 1–32.

[17] Gershwin, S. B. and Jacobson, D. H., "A Discrete Time Differential Dynamic Programming Algorithm with Application to Optimal Orbit Transfer," TR 566, Aug. 1968, Div. of Engineering and Applied Physics, Harvard Univ.