Office of Naval Research  
Contract N00014-67-A-0298-0006  
NR - 372 - 012  
National Aeronautics and Space Administration  
Grant NGR 22-007-068

NEW SECOND-ORDER AND FIRST-ORDER ALGORITHMS FOR  
DETERMINING OPTIMAL CONTROL: A DIFFERENTIAL DYNAMIC  
PROGRAMMING APPROACH

By  
D. H. Jacobson

Technical Report No. 551

Reproduction in whole or in part is permitted for  
any purpose of the United States Government.

February 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

Many of the results reported in this paper are contained in the author's Ph. D. thesis produced under the supervision of Dr. D. Q. Mayne at The Centre for Computing and Automation, Imperial College, University of London.
NEW SECOND-ORDER AND FIRST-ORDER ALGORITHMS
FOR DETERMINING OPTIMAL CONTROL: A DIFFERENTIAL
DYNAMIC PROGRAMMING APPROACH

By
D. H. Jacobson
Division of Engineering and Applied Physics
Harvard University Cambridge, Massachusetts

ABSTRACT

In this paper, the notion of Differential Dynamic Programming
is used to develop new second-order and first-order successive approximation methods for determining optimal control.

The unconstrained, non-linear control problem is first considered, and a second-order algorithm is developed which has wider application than existing second-variation and second-order algorithms. A new first-order algorithm emerges as a special case of the second-order one.

Control inequality constraints are introduced into the problem and a second-order algorithm is devised which is able to solve this constrained problem. It is believed that control constraints have not been handled, previously, in this way. Again, a first-order algorithm emerges as a special case.

The usefulness of the second-order algorithms is illustrated by the computer solution of three control problems.

The methods presented in this paper have been extended, by the author, to solve problems with terminal constraints and implicitly given final time. Details of these procedures are not given in this paper, but the relevant references are cited.
Notation

The following notation denotes the inner product of two n-dimensional vectors $x$ and $y$.

$$\langle x, y \rangle = \sum_{i=1}^{n} x_i y_i$$

A power series expansion, to second-order, of a scalar $V(x)$ about $\bar{x}$ ($x$ an n-vector) is represented in the following way:

$$V(\bar{x} + \delta x) = V(\bar{x}) + \langle V_x, \delta x \rangle + \frac{1}{2} \langle \delta x, V_{xx} \delta x \rangle$$

where $V_x = \frac{\partial V}{\partial x}$ evaluated at $\bar{x}$ and $V_{xx} = \frac{\partial^2 V}{\partial x^2}$ evaluated at $\bar{x}$.

$V_x$ is an n-dimensional column vector and $V_{xx}$ is an n x n symmetric matrix.

Further notational details are described in the text, when required.

The following abbreviations are used:

D. D. P. Differential Dynamic Programming

P. D. E. Partial Differential Equation

L. Q. P. Control problem with linear dynamics and quadratic performance criterion

r. h. s. right hand side

l. h. s. left hand side

w. r. t. with respect to

s. a. m. 'step size adjustment method'
1. Introduction

In recent years, much interest has centered on the problem of determining optimal control for dynamic systems described by non-linear ordinary differential equations of the following form:

\[ \dot{x} = f(x, u; t) ; \quad x(t_o) = x_o \]  \hspace{1cm} (1)

The criterion of optimality is the minimization of the performance index or so called 'cost functional'

\[ V(x_o; t_o) = \int_{t_o}^{t_f} L(x, u; t) \, dt + F(x(t_f); t_f) \]  \hspace{1cm} (2)

\( x(t) \) is an n-dimensional vector function of time describing the state of the dynamic system at any time \( t \in [t_o, t_f] \). (The symbol \( x \) is also taken to mean \( x(t) \).) \( u(t) \) (or \( u \)) is an m-dimensional vector function of time. At any time \( t \in [t_o, t_f] \), \( u \) represents the control variables available for manipulation.

\( L \) and \( F \) are non-linear scalar functions of their arguments.

\( f \) is an n-dimensional vector function of its arguments.

It describes the dynamic structure of the system.

The final time \( t_f \) is assumed to be given explicitly.

The notation \( f(x, u; t) \) should be understood in the following way:

'\( f \) is a function of \( x \), \( u \) and, maybe explicitly, of time. At a particular time \( t \in [t_o, t_f] \), \( f \) is a function of \( x \) and \( u \). The semi-colon is used to separate \( t \) from the other arguments. Some of these arguments may be time invariant parameters. (Some of the \( u \)'s may be control parameters, say.)

Similar remarks apply to \( V \), \( F \) and \( L \).

Sometimes it is required that \( x \) and \( u \) satisfy some or all of the following constraints.
where $g$ is a $p \leq m$-vector function of $u$ at time $t$.

$\psi$ is an $s \leq n$-vector function of $x$ at time $t_f$. All functions are assumed to be continuously differentiable in each argument up to any order required.

The object of the control problem is to choose $u(t)$; $t \in [t_0, t_f]$ such that (3) and (4) are satisfied and $V$, given by (2), is minimized.

Merriam [1], Mitter [2] and McReynolds and Bryson [3] are some who have developed second-variation type algorithms for successively improving a nominal, guessed control function. More recently Mayne [4] has developed a second-order algorithm using Dynamic Programming. (Differential Dynamic Programming) Jacobson [5], [6] and [7] has further developed the notion of D. D. P. and in [7], showed that the second-variation algorithms of Mitter and McReynolds and Bryson are only approximations to Mayne's second-order method.

All the above mentioned algorithms have the following drawbacks:

1) $H^{-1}_{uu}(\bar{x}, \bar{u}, V_x; t)$, the inverse second partial derivative matrix of the Hamiltonian w. r. t. $u$, evaluated along a nonimal trajectory $\bar{x}, \bar{u}$, must be positive-definite for $t \in [t_0, t_f]$. This restriction is very severe since it implies that $H$ must be strictly convex, globally, w. r. t. $u$. In many problems one finds that $H$ is strictly convex only in the neighborhood of its minimum w. r. t. $u$.

2) Inequality constraints on control variables cannot be handled directly. They have to be approximated by penalty functions.

3) Requirement 1) excludes the 'bang-bang' type of problem where $H_{uu} = 0$. 
It has been shown, [6], that if instead of allowing only small changes in control at each iteration, large (or global) changes in control are permitted, then difficulties 1), 2) and 3) can be overcome.

The purpose of this paper is to report new second-order and first-order algorithms, developed in [6], which do not suffer from drawbacks 1) and 2).

2. Differential Dynamic Programming

Mayne [4] introduced the notion of D. D. P. The treatment in this section is somewhat different, but was motivated by Mayne's paper.

It is well known that the optimal cost \( V^O(x; t) \) satisfies the following P. D. E.

\[
-\frac{\partial V^O(x; t)}{\partial t} = \min_{u} \left[ L(x, u; t) + \langle V^O_x(x; t), f(x, u; t) \rangle \right]
\]  \[(5)\]

Equation (5) is Bellman's P. D. E. for the optimal cost \( V^O \). (In this equation one should realise that \( x \) is an optimal quantity and so should be written with superscript \( ^O \). However, for simplicity, the superscript \( ^O \) appears on 'V quantities' only.)

It is assumed that \( V^O(x; t) \) is sufficiently smooth in \( x \) and \( t \) to allow the derivation of (5) which requires that the second partial derivatives of \( V^O \) w. r. t. \( x; t \) exist. [8]

Assume that the optimal control \( u^O(t); t \in [t_o, t_f] \) is unknown but that a nominal control \( \bar{u}(t); t \in [t_o, t_f] \) is available.

On application of the nominal control a nominal state trajectory \( \bar{x}(t); t \in [t_o, t_f] \) is produced by (1). The nominal cost \( \bar{V}(x_o; t_o) \) is calculated using (2).
Equations (1), (2) and (5) may be written in terms of the nominal trajectory by setting:

\[ x = \bar{x} + \delta x \]
\[ u = \bar{u} + \delta u \quad (6) \]

\( \delta x \) and \( \delta u \) are the state and control variables, respectively, measured w. r. t. the nominal quantities \( \bar{x}, \bar{u} \); they are not necessarily small quantities.

Equations (1), (2) and (5) become:

\[
\frac{d}{dt}(\bar{x} + \delta x) = f(\bar{x} + \delta x, \bar{u} + \delta u; t); \quad \bar{x}(t_o) + \delta x(t_o) = x_o \quad (7)
\]

\[
V(x_o; t_o) = \int_{t_o}^{t_f} L(\bar{x} + \delta x, \bar{u} + \delta u; t) dt + F(\bar{x}(t_f) + \delta x(t_f); t_f) \quad (8)
\]

\[
-\frac{\partial V^0}{\partial t}(\bar{x} + \delta x; t) = \min_{\delta u} \left[ L(\bar{x} + \delta x, \bar{u} + \delta u; t) + \left\langle V^0(\bar{x} + \delta x; t), f(\bar{x} + \delta x, \bar{u} + \delta u; t) \right\rangle \right] \quad (9)
\]

These equations are exactly equivalent to (1), (2) and (5) since no approximations have been introduced: the nominal trajectory has been made into a reference trajectory.

Assume now that the optimal cost is smooth enough to allow for a power series expansion in \( \delta x \) about \( \bar{x} \).

\[
V^0(\bar{x} + \delta x; t) = V^0(\bar{x}; t) + \left\langle V^0_{x}(\bar{x}; t), \delta x \right\rangle + \frac{1}{2} \left\langle \delta x, V^0_{xx}(\bar{x}; t) \delta x \right\rangle + \text{higher-order terms} \quad (10)
\]

The optimal cost \( V^0(\bar{x}; t) = V(\bar{x}; t) + a^0(\bar{x}; t) \) \( (11) \)

where \( a^0(\bar{x}; t) \) is defined as the difference between the optimal cost \( V^0(\bar{x}; t) \) obtained by using the optimal controls \( u^o(\tau) = \bar{u}(\tau) + \delta u^o(\tau) \)
and the nominal cost $\bar{V}(\tilde{x}; t)$ obtained using the nominal controls $u(\tau); \tau \in [t, t_f]$

From (10) and (11):

$$V^0(\tilde{x} + \delta x; t) = \bar{V}(\tilde{x}; t) + a^0 + \langle V^0_x, \delta x \rangle + \frac{1}{2} \langle \delta x, V^0_{xx} \delta x \rangle + \text{higher-order terms in } \delta x$$  \hspace{1cm} (12)

Substituting (12) into (9) one obtains:

$$\frac{-\partial \bar{V}}{\partial t} - \frac{\partial x}{\partial t} - \langle \frac{\partial V^0_x}{\partial t}, \delta x \rangle - \frac{1}{2} \langle \delta x, \frac{\partial V^0_{xx}}{\partial t} \delta x \rangle + \text{higher-order terms}$$

$$= \min_{\delta u} \left[ L(\tilde{x} + \delta x, \tilde{u} + \delta u; t) + \langle V^0_x + V^0_{xx}, \delta x \rangle + \text{higher-order terms}, f(\tilde{x} + \delta x, \tilde{u} + \delta u; t) \right]$$  \hspace{1cm} (13)

Equation (13) is generally impossible to solve, as it stands, owing to the possibly infinite computing time, and storage requirements for the parameters of the power series expansion. However, one can truncate the power series provided that it is ensured that the truncated terms are negligible. In order to do this, the size of $\delta x$ must, somehow, be limited. That is, the trajectory $\bar{x}(t) + \delta x(t); t \in [t_o, t_f]$ must be kept in the neighborhood of the nominal trajectory.

Assume that $\delta x$ is kept small. (Methods of ensuring this are discussed later. At this point it is sufficient to note that, because $\delta x(t_o) = 0$, the $\delta x$'s produced in the interval $[t_o, t_f]$ are caused only by $\delta u$ acting through Equation (7).) Assume further, that the $\delta x$'s produced are small enough such that an expansion up to quadratic terms only, in $\delta x$, is sufficient to represent $V^0$ adequately in the neighborhood of the nominal trajectory. Equation (13) becomes:
\[- \frac{\partial V}{\partial t} + \nabla \cdot \delta x = \left< \frac{\partial V}{\partial t}, \delta x \right> + \frac{1}{2} \left< \delta x, \frac{\partial^2 V}{\partial x^2} \delta x \right> =
\]

\[\min_{\delta u} \left\{ L(\bar{x} + \delta x, \bar{u} + \delta u; t) + \left< V_x + \frac{\partial V}{\partial x} \delta x, f(\bar{x} + \delta x, \bar{u} + \delta u, t) \right> \right\}\]

(14)

where

\[V(\bar{x} + \delta x; t) = \bar{V}(\bar{x}; t) + a + \left< V_x, \delta x \right> + \frac{1}{2} \left< \delta x, V_{xx} \delta x \right> \]

and

\[V_x(\bar{x} + \delta x; t) = V_x(\bar{x}; t) + V_{xx} \delta x \]

(15)

Notice that expanding \(V\) to second-order only in \(\delta x\) produces, on differentiating, an expression for \(V_x(\bar{x} + \delta x; t)\) which is accurate only to first-order in \(\delta x\).

In subsequent sections of this paper, an algorithm shall be considered 'second-order' if, given the a priori expansions (15) for \(V\) and \(V_x\), all second-order terms arising on the r.h.s. of (14) during the derivations, are accounted for.

The superscript \(^0\) on \(V\) in (14) and (15) has been dropped for the following reason:

Modelling the cost surface, locally, by a second-order expansion is made possible by keeping \(\delta x\) small. So the cost described by the truncated series (15) is optimal, subject to the proviso that the \(\delta u\)'s are chosen in such a way that the \(\delta x\)'s remain small. It is therefore, not the truly optimal cost given that any size of \(\delta x\) is allowed.

\(V\), given by (15), is the optimal cost \(V^0\) when starting in state \(\bar{x} + \delta x\) at time \(t\) if:
either 1) The nominal trajectory is sufficiently close to the optimal one, i.e. if \( \bar{u}(\tau) \) is close to \( u^0(\tau) \); \( \tau \in [t, t_f] \) then the minimizing \( \delta u \)'s will be small and, from (7), the resulting \( \delta x \)'s will be small so that the size of the \( \delta x \)'s need not be restrained artificially. The expansion of \( V \) up to second-order will thus be adequate to describe the true optimal cost \( V^0 \) in the neighborhood of the nominal trajectory. \( V^0 \) will also be described adequately.

or 2) The problem is L. Q. P. [9].

Equation (14) can be used to develop methods for determining the optimal control \( u^0(t); t \in [t_0, t_f] \) by successively improving the current nominal trajectory \( \bar{u}(t); t \in [t_0, t_f] \). See [5], [6] and [7].


Before proceeding with the derivation of the algorithm, a brief note describing the approach used, is in order.

To overcome drawback 1) of Section 1, \( H(\bar{x}, u, V_x; t) \) is actually minimized w. r. t. \( u \). This minimizing \( u \) is denoted by \( u^* \). All quantities, including \( H^{-1}_{uu}(\bar{x}, u, V_x; t) \), are then evaluated at \( u^* \). Because \( u^* \) minimizes \( H \), the requirement that \( H^{-1}_{uu}(\bar{x}, u^*, V_x; t) \) be positive-definite is not nearly as restrictive as requiring \( H \) to be globally, strictly convex in \( u \). Further, \( H_u(\bar{x}, u^*, V_x; t) = 0 \), which is a well known condition of optimality.
Variations $\delta x$ about $\bar{x}$ are then introduced and a linear relationship found between $\delta u$ and $\delta x$, which maintains the necessary condition of optimality, $H_u (\bar{x} + \delta x, u^* + \delta u, V_x + V_{xx} \delta x; t) = 0$, for $\delta x$ sufficiently small.

With the above two points in mind, the derivation of the second-order algorithm will be understood readily.

The full deviation of the algorithm follows:

At any time $t \in [t_0, t_f]$, Equation (14) is valid locally w. r. t. $\delta x$, but globally w. r. t. $\delta u$. Consider Equation (14) at time $t$ with $\delta x(t)$ set equal to zero; its r. h. s. becomes:

$$\min_{\delta u} \left[ L(\bar{x}, \bar{u} + \delta u; t) + \langle V_x, f(\bar{x}, \bar{u} + \delta u; t) \rangle \right]$$

(16)

Instead of using a second-order prediction of the minimizing $\delta u$, as is done in [1], [2], [3] and [4], let us completely minimize the contents of the square brackets in Equation (16) w. r. t. $\delta u$; this may be done analytically or, if necessary, numerically.

Let the minimizing control be $u^* = \bar{u} + \delta u^*$. Expression (16) becomes:

$$L(\bar{x}, u^*; t) + \langle V_x, f(\bar{x}, u^*; t) \rangle$$

(17)

Now consider variations $\delta x$ about $\bar{x}$, i.e. re-introduce $\delta x$.

In order to maintain minimality of the r. h. s. of (14) the min must $\delta u$ be re-introduced; however, $\delta u$ is now measured with reference to $u^*$.

$$-\frac{\partial V}{\partial t} - \frac{\partial u^*}{\partial t} - \langle \frac{\partial V_x}{\partial t}, \delta x \rangle - \frac{1}{2} \langle \delta x, \frac{\partial^2 V}{\partial t^2} \delta x \rangle =$$

$$\min_{\delta u} \left[ L(\bar{x} + \delta x, u^* + \delta u; t) + \langle V_x + V_{xx} \delta x, f(\bar{x} + \delta x, u^* + \delta u; t) \rangle \right]$$

(18)
Of course, by allowing these large (global) changes in control, large $\delta x$'s will be introduced via Equation (7). The $\delta x$'s must, in some way, be restrained in size in order that the second-order expansion for $V$ be valid. This point is discussed later.

Define

$$H(x, u, V_x; t) = L(x, u; t) + \langle V_x, f(x, u; t) \rangle$$

(19)

The r. h. s. of (18) becomes:

$$\min_{\delta u} H(\bar{x} + \delta x, u^* + \delta u, V_x + V_{xx} \delta x; t)$$

(20)

Since $u^*$ minimizes $H(\bar{x}, u, V_x; t)$, the following necessary condition holds:

$$H_u(\bar{x}, u^*, V_x; t) = 0$$

(21)

Expanding (20) about $\bar{x}, u^*$ the following expression is obtained:

$$\min_{\delta u} \left[ H + \langle H_u, \delta u \rangle + \langle H_x, \delta x \rangle + \langle V_{xx} f, \delta x \rangle + \langle \delta u, \frac{\partial f}{\partial u} \rangle + \frac{1}{2} \langle \delta u, H_{uu} \delta u \rangle + \text{higher-order terms} \right]$$

(22)

All quantities in (22) are evaluated at $\bar{x}, u^*; t$.

From Equation (21), $H_u = 0$, so the terms involving $\delta u$ in (22) are:

$$\langle \delta u, (H_{ux} + \frac{\partial f}{\partial u} V_{xx}) \delta x \rangle + \frac{1}{2} \langle \delta u, H_{uu} \delta u \rangle + \text{higher-order terms}$$

(23)

If $\delta u$ is of the same order as $\delta x$ then these terms are quadratic in $\delta x$ + higher-order terms in $\delta x$. There is, therefore, no point in finding a relationship between $\delta u$ and $\delta x$ which is of order higher than linear, since terms higher than second-order in $\delta x$ are neglected. (Since the l. h. s. of (18) is expanded only to second-order in $\delta x$.) A relationship of the following form is therefore required:
\[ \delta u = \beta \delta x \] 

(24)

where \( \beta \) is chosen to minimize the contents of the square brackets in Expression (22).

A necessary condition for minimality is obtained by differentiating (22) w. r. t. \( \delta u \) and equating to zero.

\[
H_u + H_{uu} \delta u + (H_{ux} + f^T V_{xx}) \delta x + \text{higher-order terms} = 0
\]

(25)

Substituting (24) into (25):

\[
H_u + H_{uu} \beta \delta x + (H_{ux} + f^T V_{xx}) \delta x + \text{higher-order terms in } \delta x = 0
\]

(26)

From Equation (21), \( H_u = 0 \). For (26) to hold for \( \delta x \) sufficiently small, coefficients of the first-order terms may be equated to zero to yield:

\[
\beta = -H^{-1}_{uu}(H_{ux} + f^T V_{xx})
\]

(27)

Quantities in (27) are evaluated at \( x, u^*; t \).

This \( \beta \) is the optimal linear feedback controller which maintains the necessary condition of optimality \( H_u (x + \delta x, u^* + \delta u, V_x + V_{xx} \delta x; t) = 0 \) for \( \delta x \) sufficiently small.

Substituting (24) into (22) and neglecting terms of order higher than the second, one obtains:

\[
H + \langle H_x + V_{xx} f + \beta^T H_u, \delta x \rangle + \hat{\beta} \langle \delta x, (H_{xx} + f^T V_{xx} + V_{xx} f_x - \beta^T H_{uu} \beta) \delta x \rangle
\]

(28)

Expression (28) equals the l. h. s. of Equation (18). Since equality holds for all \( \delta x \) sufficiently small, the coefficients of like powers of \( \delta x \) may be equated to obtain:
\[-\frac{\partial \mathcal{V}}{\partial t} - \frac{\partial a}{\partial t} = H\]

\[-\frac{\partial \mathcal{V}}{\partial t} = H_x + \mathcal{V}_{xx} f + \beta^T H_u\]

\[-\frac{\partial \mathcal{V}}{\partial t} = H_{xx} + f^T \mathcal{V}_{xx} + \mathcal{V}_{xx} f_x - (H_u x + f_u^T \mathcal{V}_{xx}) H_u^{-1} (H_u x + f_u^T \mathcal{V}_{xx})\]

All quantities evaluated at $\bar{x}$, $u^*$; $t$.

$\mathcal{V}$, $a$, $\mathcal{V}_x$ and $\mathcal{V}_{xx}$ are all functions of $x$ and $t$ along the nominal $\bar{x}$ trajectory so:

\[
\frac{d}{dt} (\mathcal{V}(\bar{x}; t) + a(\bar{x}; t)) = \frac{\partial}{\partial t} (\mathcal{V} + a) + \langle \mathcal{V}_x, f(\bar{x}, \bar{u}; t) \rangle
\]

Also

\[
\dot{\mathcal{V}}_x = \frac{\partial \mathcal{V}}{\partial x} + \mathcal{V}_{xx} f(\bar{x}, \bar{u}; t)
\]
\[
\dot{\mathcal{V}}_{xx} = \frac{\partial \mathcal{V}_{xx}}{\partial t}
\]

since higher-order terms of $\mathcal{V}$ have been truncated.

Using Equations (30) in (29) and noting that $-\frac{\partial \mathcal{V}}{\partial t} = \mathcal{L}(\bar{x}, \bar{u}; t)$:

\[-\dot{a} = H - H(\bar{x}, \bar{u}, \mathcal{V}_x; t)\]

\[-\dot{\mathcal{V}}_x = H_x + \mathcal{V}_{xx} (f - f(\bar{x}, \bar{u}; t)) + \beta^T H_u^{\infty}\]

\[-\dot{\mathcal{V}}_{xx} = H_{xx} + f^T \mathcal{V}_{xx} + \mathcal{V}_{xx} f_x - (H_u x + f_u^T \mathcal{V}_{xx}) H_u^{-1} (H_u x + f_u^T \mathcal{V}_{xx})\]

(31)
Unless otherwise stated, all quantities are evaluated at \( \bar{x}, u^*; t \). (The symbolism \( H_u^o \) is used to indicate that \( H_u = 0 \).)

At \( t = t_f \), \( V(\bar{x}; t_f) = F(\bar{x}(t_f); t_f) \)

whence: \( a(t_f) = 0 \)

\[
V_x(t_f) = F_x(\bar{x}(t_f); t_f) \quad V_{xx}(t_f) = F_{xx}(\bar{x}(t_f); t_f) \tag{32}
\]

Equations (32) are boundary conditions for the differential Equations (31).

These equations are similar to those obtained by Mitter [2], McReynolds and Bryson [3] and Mayne [4]. An important difference is that the above equations are evaluated at \( \bar{x}, u^* \) and not \( \bar{x}, \bar{u} \).

The new control that is applied to the system is, of course

\[
\delta u = \bar{u} + \delta u^* + \beta \delta x = u^* + \beta \delta x \tag{33}
\]

The above theory assumes that the \( \delta x \)'s generated by (33) will be small enough to justify the second-order expansions used earlier.

If \( \delta x \) becomes too large, a scale factor \( \varepsilon \); \( 0 < \varepsilon \leq 1 \) cannot be placed in front of \( \delta u^* (\delta u^* = u^* - \bar{u}) \), as is done in [2], [3] and [4], since \( u^* = \bar{u} + \delta u^* \) is imbedded in the reverse differential Equations (31) which have already been integrated. Moreover, \( H \) is often non-convex w. r. t. \( u \), which precludes this type of linear interpolation between \( \bar{u} \) and \( \bar{u} + \delta u^* \).

(In the L. Q. P. problem there is no difficulty since immediate application of (33) yields the optimal solution.)

4. A New 'Step Size Adjustment Method.'

Substituting (33) into (7), the following equation is obtained:

\[
\frac{d}{dt} (\bar{x} + \delta x) = f(\bar{x} + \delta x, u^* + \beta \delta x; t) \tag{34}
\]
with \( \bar{x}(t_0) + \delta x(t_0) = x_0 \)

Since \( \delta x(t_0) = 0 \), the \( \delta x \)'s produced by (34) are due to the driving action of \( \delta u^* = u^* - \bar{u} \).

A way in which the size of the \( \delta x \)'s can be restrained is by altering the time interval over which Equation (34) is integrated.

Consider the time interval \([t_1, t_f]\) where \( t_o \leq t_1 < t_f \). Assume that one runs along the nominal trajectory \( \bar{x} \) from \( t_o \) to \( t_1 \). At time \( t = t_1 \), \( x(t_1) = \bar{x}(t_1) \) since the path of the nominal trajectory has been followed from \( t_o \) to \( t_1 \). (i.e. \( \delta x(t); t \in [t_o, t_1] \) is zero) Now consider integrating (34) over the time interval \([t_1, t_f]\). If \( t_1 < t_f \) and \([t_1, t_f]\) is small, then the \( \delta x \)'s produced by (34) in this interval will be small, even for large \( \delta u^* \), since there is very little time over which to integrate the differential equation:

\[
\frac{d}{dt} (\bar{x} + \delta x) = f(\bar{x} + \delta x, u^* + \beta \delta x; t); \quad x(t_1) + \delta x(t_1) = \bar{x}(t_1)
\]

By making \( t_1 \) near to \( t_f \) one can force the \( \delta x \)'s to be as small as one pleases.

The above description is summarised in the following statement:

There exists a time \( t_1 \), sufficiently close to \( t_f \), in the range \( t_o \leq t_1 < t_f \), such that if the nominal trajectory is followed from \( t_o \) to \( t_1 \) and then (35) is integrated from \( t_1 \) to \( t_f \), the \( \delta x \)'s produced by (35) in the interval \([t_1, t_f]\) will be 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 the \( \delta x \)'s are 'small enough'?

2) How does one chose a \( t_1 \) such that the \( \delta x \)'s produced are 'small enough'?
1) Recall that \( |a(\bar{x}, t_1)| = \int_{t_f}^{t_1} [H - H(\bar{x}, \bar{u}, V_x; t)] \, dt \) is the predicted improvement in cost when starting at the point \( \bar{x}(t_1); t_1 \) and using \( u(\tau) = u^*(\tau) + \delta x(\tau); \tau \in [t_1, t_f] \).

Assume for the moment that \( t_1 = t_o \), (i.e., consider the whole time interval \([t_o, t_f]\).) Integrate (35) and calculate the cost \( V \). The actual improvement in cost is

\[
\Delta V = \bar{V}(\bar{x}; t_o) - V(\bar{x}; t_o)
\]

If this actual improvement in cost is 'near' to the predicted value \( |\bar{a}(\bar{x}; t_1)| \), then the \( \delta x \)'s produced by the new control, acting through Equation (35), are considered 'small enough'.

It is convenient, in practice, to define 'near' in the following way:

If the following inequality is satisfied, \( \Delta V \) is considered to be 'near' \( |a(\bar{x}, t_1)| \)

\[
\frac{\Delta V}{|a(\bar{x}; t_1)|} > c \quad ; \quad c \geq 0
\]

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 \( \Delta V \) is inadmissible. \( c \) should not be greater than unity since one should not expect improvements 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 (37) are not influenced by round off errors in the computations.

2) If test (37) is passed with \( t_1 = t_o \) all is well, and the next iteration of the main algorithms may be begun with the knowledge that a reasonable reduction in cost of \( \Delta V \) has been made. If (37) is not satisfied then set
The above procedure is repeated with this $t_1$ and (37) is checked again (with the new $\Delta V$). If it is satisfied then the next iteration is begun. If not, then set

$$t_1 = \frac{t_f - t_o}{2} + t_o = t_{o1}$$

and repeat again.

Subdividing $[t_o, t_f]$ in this way, there will come a time $t_1$ when iteration (37) is satisfied.

In general

$$t_1 = \frac{t_f - t_{or}}{2} + t_{or} = t_{or1}$$

where $r = 0, 1, \ldots$ and $t_{oo} = 2t_o - t_f$.

Notice that the new nominal trajectory will sometimes have a corner at $t_1$ since $\bar{u}(t_1)$ may be different from $u^*(t_1)$. This introduces no difficulty provided that the numerical integration routine used is capable of handling differential equations with discontinuous right hand sides. An example of such a method is the Fourth-order Runge-Kutta routine.

It may happen that the nominal trajectory $\bar{x}(t)$ is optimal on an interval $[t_2, t_f]$; $t_2 \in [t_o, t_f]$, but is non-optimal on the interval $[t_o, t_f]$. If $t_1$ is being found in the manner outlined above, then a trial $t_1$ may fall in the interval $[t_2, t_f]$. The $\delta x$'s generated in the interval $[t_1, t_f]$ would then be zero - because $u^*(t) = \bar{u}(t); t \in [t_2, t_f]$ - and no reduction in cost would occur, even though the whole trajectory $\bar{x}(t); t \in [t_o, t_f]$ is non-optimal in $[t_2, t_f]$. One must ensure, therefore, that $t_1$ will never fall in $[t_2, t_f]$. This condition is ensured easily in the following way:
At \( t = t_f \), \( a(x; t) = 0 \). When integrating the backwards equations monitor \( |a(x; t)| \). Record the time \( t_{\text{eff}} \) when \( |a(x; t)| \) becomes different from zero. (Or in practice, when it becomes greater than a small, positive quantity, \( \eta \)). The trajectory between \( t_{\text{eff}} \) and \( t_f \) satisfies a necessary condition of optimality, viz:

\[
a(x; t) = 0 \quad t \in [t_{\text{eff}}, t_f]
\]  

(41)

If, on the forwards run, a time \( t_1 \neq t_0 \), needs to be found then the time interval \([t_0, t_{\text{eff}}]\) is subdivided as described earlier, and not \([t_0, t_f]\).

As the overall trajectory becomes more and more optimal, from iteration to iteration, \( t_{\text{eff}} \rightarrow t_o \). Finally, on an optimal trajectory, \( |a(x; t)| < \eta_i \), \( t \in [t_o, t_f] \) and \( t_{\text{eff}} = t_o \) and the computation is stopped.

When programming algorithms on a digital computer it is generally necessary to use a numerical integration routine to integrate the differential equations. This means that the interval \([t_0, t_f]\) is divided into \( N-1 \) time steps. (i.e. \( t \) going from 1 to \( N \))

The subdivision of \([t_0, t_{\text{eff}}]\) used for determining \( t_1 \) must be done w.r.t. this discretised scale, i.e. Now a time \( N_1 \) must be sought, \( N_1 \in [1, N_{\text{eff}}] \), where \( N_1 \) is given by

\[
N_1 = \frac{N_{\text{eff}} - N_{\text{or}}}{2} + N_{\text{or}} = N_{\text{or}} + 1
\]  

(42)

where \( N_{\text{or}} = 2 - N_{\text{eff}} \) and \( r = 0, 1, \ldots \). Integer division is used in (42).

\( r \) is increased until \( N_1 = N_{\text{eff}} - 1 \). If \( N_{\text{eff}} = 1 \) 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. The smallest possible non zero time interval is
It is clear that \( N \) must be large enough such that the \( \delta \)'s produced during this basic time interval are 'small enough'. This restriction is a practical one, brought about by the discrete time routine of the digital computation.

When \( \Delta V \) and \( |a(\bar{x}; N_1)| \) are small, but \( > \eta_1 \), the criterion (37) may be too severe with \( c = 0.5 \), owing to round off errors. i.e. there may come a stage where (37) remains unsatisfied even when \( N_1 = N_{\text{eff}} - 1 \). 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 (37) is unsatisfied, even when \( N_1 = N_{\text{eff}} - 1 \), then stop the computation since no further reduction in cost is possible. This implies that either optimality has been attained (in which case \( |a(\bar{x}; t_0)| < \eta_1 \) and so \( N_{\text{eff}} < 1 \)) or \( N \) is not large enough and hence \( t_f - t_0 \)

\( \frac{N - 1}{N_{\text{eff}} - 1} \) is too large a basic time interval. Usually, however, the \( N \) needed for accurate integration of the differential equations is large enough; the contrary has been encountered only in some problems which are near singular and hence extremely sensitive to changes in \( u \). In these cases it may prove desirable to use a very simple integration routine (Euler) and a large number of steps, \( N \).

A summary of the 'Step Size Adjustment Method' is given in Flow Chart I.

5. The Overall Computational Procedure.

The computational procedure is given in Flow Chart II.

The minimization of \( H \) w.r.t. \( u \), required when going backwards in time, may be done either analytically or by using one of the well known hill climbing techniques. [10]
Obtain, from main algorithm, the time $N_{eff}$ when $|a(x; t)|$ becomes greater than $\eta_1 \cdot \eta_1$, a small positive quantity.

Yes - HALT; + Denotes integer division

Is $N_{eff} < 1$?  
Yes -> HALT; OPTIMAL FOUND.
No ->
Set $C=0.5$

Set $r=0$

$N_1 = \left( \frac{N_{eff} - N_{or}}{2} \right)^+ + N_{or} = N_{or} + 1$ where $N_{oo} = 2 - N_{eff}$

Apply $u = \bar{u}$ on the interval $[1, N_1]$ and $u = u^* + \beta \delta x$ on the interval $[N_1, N]$. Calculate the cost $V(x_0; l)$ and hence the improvement $\Delta V = \bar{V}(x_0; l) - V(x_0; l)$

Is criterion $\frac{\Delta V}{|a(x; N_1)|} > c$ satisfied?  
Yes, $N_1$ satisfactory -> Proceed to next iteration of main algorithm
No ->
Is $N_1 = N_{eff} - 1$ or is $N_{eff} = 1$?
Yes -> HALT; NO IMPROVEMENT IN TRAJECTORY ATTAINABLE.
No ->
Increment $r$ by 1

Is $c = 0.0$?
Yes -> HALT; NO IMPROVEMENT IN TRAJECTORY ATTAINABLE.
No -> Set $c = 0.0$

FLOWCHART I: "STEP SIZE ADJUSTMENT METHOD".
Using a nominal control $\tilde{u}(t)$; $t \in [t_0, t_f]$ run a nominal $\tilde{x}(t)$ trajectory. Calculate the nominal cost $\tilde{\nabla}(\tilde{x}_0; t_0)$. Store the $\tilde{x}$ and $\tilde{u}$ trajectories and $\tilde{\nabla}$.

Using boundary conditions (32), integrate equations (31) backwards from $t_f$ to $t_0$, all the while minimizing $H$ w.r.t. $u$ to obtain $u^*$, and storing $u^*(t)$ and $\beta(t)$. Note also the time $N_{\text{eff}}$ when $|a(\tilde{x}; t)|$ becomes greater than $\eta_1$. $\eta_1$ chosen from numerical stability considerations.

Apply the "step size adjustment method" (s.a.m.) to obtain a new improved trajectory. If the current nominal control is optimal or if an improved control cannot be found, then s.a.m. halts the computation.

If an improved trajectory is obtained, replace the old nominal $\tilde{x}$, $\tilde{u}$ and $\tilde{\nabla}$ by these new values.

FLOW CHART II: THE OVERALL COMPUTATIONAL PROCEDURE

1) The procedure exhibits one step convergence on L, Q, P. problems [9].

2) In a neighborhood of the optimum, convergence is rapid for non-linear problems because the second-order expansions represent the functions \( V^0 \), \( L \) and \( f \) well, for small \( \delta x \) and \( \delta u \).

3) The algorithm is very much more powerful than the existing methods [1], [2], [3] and [4] for the following reason:

   \( H^{-1}_{uu}(\bar{x}, \bar{u}, V_x^*; t) \) is not required to be positive definite along non-optimal nominal trajectories. In this algorithm \( H(\bar{x}, u, V_x; t) \) is minimized w. r. t. \( u \) and so it is required only that \( H^{-1}_{uu}(\bar{x}, u, V_x; t) \) be positive-definite at the minimizing \( u = u^* \). i.e. \( H^{-1}_{uu}(\bar{x}, u, V_x; t) \) must be strictly convex only in the neighborhood of \( u^* \). This is a much less restrictive requirement and so the algorithm is capable of handling a larger class of non-linear problems than the second-order or second-variation methods.

4) In some problems, the solution of the Riccati equation becomes unbounded along some nominal trajectories though along optimal trajectories it always has a bounded solution. The new algorithm is able to compute optimal control for these problems whereas the existing methods are not. This is illustrated by example II to follow.

5) If the \( \delta x \)'s produced by the new control are too large, as measured by criterion (37), then the 'step size adjustment routine' must be used. If the problem is very non-linear the routine will have to be used a number of times in order to determine \( t_1 \), which will be close to \( t_f \). However, as \( t_1 \rightarrow t_f \), (35) is integrated over ever decreasing time intervals \([t_1, t_f]\). This is in contrast with methods [1], [2], [3] and [4] where,
in order to determine the scale factor \( \varepsilon \), the \( x \) equation has to be integrated over the whole time interval \([t_0, t_f]\). So, in non-linear problems it is likely that the new algorithm will use less computing time in determining \( t_f \) than existing methods in determining \( \varepsilon \).

6) The algorithm requires the integration of \( n \) differential equations less than \([2] \) and \([3] \). Mitter and McReynolds and Bryson integrate an \( n \)-vector differential equation, \( \dot{h} \), additional to their equation for \( \dot{\lambda} \).

7. A Computational Trick that Improves Convergence Rate.

In the algorithm the new control is computed using

\[
u(t) = u^*(t) + \beta(t) \delta x(t)
\]  

(43)

It can happen, in non-linear problems, that \( \beta(t) \delta x(t) \) becomes too large and so invalidates the local expansions in \( \delta u \). However, \( \delta x \) might still be small enough for

\[
V_x (x + \delta x; t) = V_x + V_{xx} \delta x
\]

to be valid.

The following alternative way can be used for computing \( u(t) \):

Instead of storing \( u^*(t), \beta(t) \) store \( V_x(t) \) and \( V_{xx}(t) \). Compute \( u(t) \) directly, by minimizing \( H(x + \delta x, u, V_x + V_{xx} \delta x; t) \) w.r.t. \( u \) either analytically or using \([10] \). In this way the radius of convergence of the algorithm may be increased.

8. Sufficient Conditions for a Reduction in Cost at Each Iteration.

In order that the cost decrease at each iteration, for \( \delta x \) sufficiently small, \( a(x; t) \) must be less than zero. Sufficient conditions for \( a(x; t_f) < 0 \) are that, for \( t \in [t_o, t_f] \):
1) \( H(\bar{x}, u^*, V_x; t) < H(\bar{x}, \bar{u}, V_x; t); \ u^* \neq \bar{u} \)

Since \( u^* \) minimizes \( H \), 1) is ensured if:

\[
H_u (\bar{x}, u^*, V_x; t) = 0
\]

\[
H^{-1}_{uu} (\bar{x}, u^*, V_x; t) \text{ is positive-definite}
\]

2) The solutions of Equations (31) be bounded.

Proof:

\[
a (\bar{x}; t) = \int_{t_f}^{t_1} [H(\bar{x}, u^*, V_x; t) - H(\bar{x}, \bar{u}, V_x; t)] \, dt
\]  \hspace{1cm} (44)

For \( a (\bar{x}; t_1) < 0 \) it is clearly sufficient that:

\[
H(\bar{x}, u^*, V_x; t) < H(\bar{x}, \bar{u}, V_x; t); \ u^* \neq \bar{u}
\]  \hspace{1cm} (45)

\( u^* \) is the control that minimizes \( H \), so (45) is true for \( u^* \neq \bar{u} \)

if \( H_u (\bar{x}, u^*, V_x; t) = 0 \)

and \( H^{-1}_{uu} (\bar{x}, u^*, V_x; t) \) is positive-definite

The quantities manipulated above must be bounded in magnitude for these conditions to be valid, so it is required that the solutions of Equations (31) be bounded.


Assume that \( V(\bar{x} + \delta x; t) \) is expanded to first-order only.

\[
V(\bar{x} + \delta x; t) = \bar{V} + a + \langle V_x, \delta x \rangle
\]  \hspace{1cm} (46)

The following set of equations are obtained easily

\[
-\dot{a} = H - H(\bar{x}, \bar{u}, V_x; t); \ a(t_f) = 0
\]

\[
-\dot{V}_x = H_x \quad ; \quad V_x(t_f) = F_x(\bar{x}(t_f); t_f)
\]  \hspace{1cm} (47)
The quantities are evaluated at $x, u^*; t$ unless otherwise stated. $u^*$ is the control that minimizes $H(x, u, V_x; t)$ w. r. t. $u$.

The new control that is applied is:

$$ u(t) = u^*(t); t \in [t_1, t_f] $$

The 'step size adjustment method' is used to keep $\delta x$ small enough.


1) The algorithm is fundamentally different from the gradient or first-variation method in that Equations (47) is integrated backwards along $x, u^*$ and the new control on the forwards run is given by (48).

2) The algorithm uses the 'step size adjustment routine' described earlier. Since $V$ is expanded to first-order it might be necessary to repeatedly use the 'step size adjustment routine' a number of times before an acceptable $t_1$ is found. However, the integration of the $\dot{x}$ equation is done over ever decreasing time intervals $[t_1, t_f]$ and so it is likely that the method will be faster than the gradient method where $\epsilon$ has to be chosen. Note also that $t_1 \in [t_0, t_f]$. In the gradient method $\epsilon$ is required to be non-zero but no upper bound is available for it. This usually makes the choice of $\epsilon$, tricky.

3) Consider the problem:

$$ \dot{x} = Ax + Bu $$

$$ \min V = \int_{t_0}^{t_f} L(u; t) dt + \langle c, x(t_f) \rangle $$

(49)

If $L$ is a convex function of $u$, it can be shown that the first-order algorithm solves this problem in one step since $V^0$ is linear in $x$. This
is not true of the gradient method where only a small change in control, $\delta u = - \epsilon H_u'$, is made at each iteration.

### 11. Computed Examples

The following computed examples serve to illustrate some of the advantages of the new second-order algorithm.

1) \[ \dot{x} = -2x + 10 \tanh u ; \quad x(t_o) = 5.0 \]

Choose $u(t); \quad t \in [0, 0.5]$ to minimize

\[ V(x_0; t_o) = \int_0^{t_f} (10x^2 + u^2) dt + 10x^2(t_f) \]

The problem, though simple, is a good illustrative one since along certain non-optimal trajectories $H_{uu}^{-1}(x, \bar{u}, V_x; t)$ is not positive.

For this problem:

\[ H(x, u, V_x; t) = 10x^2 + u^2 + V_x(-0.2x + 10 \tanh u) \]

where

\[ H_u = 2u + 10V_x(1 - \tanh^2 u) \]

and

\[ H_{uu} = 2 - 20V_x \tanh u (1 - \tanh^2 u) \]

The new second-order algorithm and those of [2], [3] and [4] were programmed. A fourth-order Runge-Kutta routine was used for the integration. The interval $[0, 0.5]$ was divided into 100 steps.

From (53) it is clear that there is no guarantee that $H_{uu}^{-1}$ will be greater than zero for any nominal trajectory $\bar{x}, \bar{u}$. i.e. there is no guarantee that methods [2], [3] and [4] will be successful.

The new algorithm requires only that $H_{uu}^{-1}(\bar{x}, u^*, V_x; t) > 0$. At $u = u^*$ one has, since $H_u(\bar{x}, u^*, V_x; t) = 0$, that:
Using Equation (54) in (53):

\[ H_{uu}(\bar{x}, u^*, V_x; t) = 2 + 4u^* \tanh u^* \]  

\( u^* \tanh u^* \geq 0 \) for all \( u^* \) so, from (55), \( H_{uu}(\bar{x}, u^*, V_x; t) > 0 \) regardless of the nominal trajectory. The new algorithm should, therefore, not fail to solve this problem.

A nominal control \( u(t) = +1 \); \( t \in [0, 0.5] \) was chosen and an attempt was made to use methods [2], [3] and [4]. For this nominal control, \( H_{uu}(\bar{x}, \bar{u}, V_x; t) \) turned out to be negative for \( t \in [0, 0.5] \) and so the algorithms were unable to improve the trajectory.

Starting from the same nominal trajectory, the new algorithm was tried. By (55) above, \( H_{uu}(\bar{x}, u^*, V_x; t) \) remained positive on the interval [0, 5]. \( u^* \) was determined by quadratic prediction [11]). On the forward run, a reduction in cost was achieved and after two iterations, the optimal trajectory was reached. The cost was reduced from the nominal value of 886.0 to the optimal value of 41.6. The trajectory was considered optimal when \( |a(x_o; t_o)| \), the predicted reduction in cost, was less than 0.1. Figure 1 shows the nominal and optimal control functions.

This simple example illustrates the failure of methods [2], [3], [4] to find a solution to a control problem where the nominal trajectory is such that \( H_{uu}^{-1}(\bar{x}, \bar{u}, V_x; t) \) is non-positive-definite. The new algorithm where \( H(\bar{x}, u, V_x; t) \) is minimized w.r.t. \( u \) easily finds the optimal trajectory.
FIG. 1 THE SCALAR CONTROL PROBLEM: NOMINAL AND OPTIMAL CONTROL FUNCTIONS
It should be noted that when using the new algorithm, the 'computational trick' of minimizing $H(\mathbf{x} + \delta\mathbf{x}, u, V_{x} + V_{xx} \delta x; t)$ w. r. t. $u$ was used on the forward run. The same problem was tried using (43) to calculate the new control; four iterations were required to reach the optimum. This is owing to the fact that $H$ is very non-linear in $u$ and so, $u = u^* + \beta \delta u$ is valid only for very small variations $\delta u$ from $u^*$. An increased radius of convergence is thus obtained by choosing $u$ by $\min_u H(\mathbf{x} + \delta\mathbf{x}, u, V_{x} + V_{xx} \delta x; t)$.

2) The Rayleigh Equation

In this example, the solution of the Riccati equation becomes unbounded when integrating backwards from $t_f$ to $t_o$ along some nominal trajectories. It is demonstrated that the new second-order algorithm is still able to achieve a reduction in cost at each iteration and moreover, reaches the optimal trajectory after 9 iterations. The methods [2], [3] and [4] fail to solve this problem.

Consider the following control problem

\[
\dot{x}_1 = x_2 \quad ; \quad x_1(t_o) = -5.
\]
\[
\dot{x}_2 = -x_1 + 1.4x_2 - .14x_2^3 + 4u \quad ; \quad x_2(t_o) = -5. \tag{56}
\]

Find $u(t); t \in [0, 2.5]$ to minimize

\[
V = \int_0^{2.5} (x_1^2 + u^2) \, dt \tag{57}
\]

\[
H = x_1^2 + u^2 + V_{x_1} x_2 + V_{x_2} (-x_1 + 1.4x_2 - .14x_2^3 + 4u)
\]

\[
H_u = 2u + 4V_{x_2} \quad ; \quad \text{whence } u^* = -2V_{x_2}
\]

\[
H_{uu} = 2 > 0 \tag{56}
\]
An arbitrary nominal control of

\( \bar{u}(t) = -0.5 \); \( t \in [0, 2.5] \)

was chosen.

The fourth order Runge-Kutta routine was used for the integrations. One hundred integration steps were used. The methods [2], [3], [4] were tried firstly. It was found that during the backwards integration of the \( \dot{V}_x, \dot{V}_{xx} \) equations, their solutions became unbounded. The integration step-size was reduced by increasing the number of integration steps from 100 to 1000, but the same behavior persisted. This meant that the methods could not be used.

Since, for this problem, the \( \dot{V}_x \) equation is linear in \( V_x \), its solution can be unbounded only if \( V_{xx} \) becomes unbounded. i.e. if the solution of the Riccati equation becomes unbounded.

From [9], sufficient conditions for the boundedness of the Riccati solution are that:

\[
H_x = \begin{bmatrix}
2x_1 - V_{x_2} \\
V_{x_1} + 1.4V_{x_2} - .42V_{x_2}^2
\end{bmatrix}
\]

\[
H_{ux} = 0
\]

\[
H_{xx} = \begin{bmatrix}
2 & 0 \\
0 & -0.84V_{x_2}^2
\end{bmatrix}
\]
\[ H_{xx} - H_{ux}^T H_{uu}^{-1} H_{ux} \text{ is positive semi-definite} \]

\[ H_{uu}^{-1} \text{ is positive definite} \]

\[ F_{xx}(\bar{x}(t); t) \text{ is positive semi-definite} \]

The second and third conditions are satisfied since \( H_{uu} = 2 \), \( F_{xx} = 0 \). In addition \( H_{ux} = 0 \).

It turns out that \( H_{xx} \) given by (59) is not positive semi-definite all along the nominal trajectory. i.e. the above sufficiency conditions for the boundedness of the solution of the Riccati equation are not satisfied. This could account for the observed unboundedness of the solution.

The new algorithm was tried next. Once again the solutions of the backwards equations became unbounded. However, using the 'step size adjustment method' a \( t_b \) could be found; \( t_b < t_1 < t_f \) - \( t_b \) the time at which the solutions became unbounded. - such that an improvement in cost resulted. As the method progressed from iteration to iteration, so \( t_b \) became nearer and nearer \( t_o \). Finally the optimal trajectory was reached along which all the equations had bounded solutions. i.e. the new second-order algorithm was entirely successful.

Fig. 2 shows \( V_{xx}(t) \) for various iterations, illustrating how the time \( t_b \) moved backwards to \( t_o \) as the optimum trajectory was approached. Fig. 3 shows the cost \( V \) as a function of the iteration number.

The above example shows that the new algorithm is more powerful than existing one's in the sense that the boundedness of the solutions of the backwards equations is not required on the whole interval \([t_o, t_f]\).
FIG. 2 THE RAYLEIGH CONTROL PROBLEM: $V_{x_2}(t)$ FOR VARIOUS ITERATIONS.
FIG. 3 THE RAYLEIGH CONTROL PROBLEM: $V$ AS A FUNCTION OF ITERATION NUMBER
along non-optimal trajectories. It is required only that they have bounded solutions on the interval \([t_b, t_f]\); \(t_b < t_f\) and that \(x(t)\); \(t \in [t_b, t_f]\) be non-optimal unless \(t_b < t_o\).

If these conditions are fulfilled, then one can always find a \(t_1\), such that the cost decreases.

It should be clear from the above, that the conditions of Section 8 are 'over sufficient'.

12. **Control Inequality Constraints: A New Second-Order Algorithm.**

Consider the control problem where constraints of the form (3) are present, i.e.

\[
g(u; t) \leq 0
\]  

(61)

where \(g\) is a \(p \leq m\) vector function.

In the past, it has not been obvious how to develop second-order algorithms for solving this problem. The following is quoted from [12], page 61:

'While the steepest ascent method presents no difficulty when upper and/or lower limits are present on the forcing signals (e.g. \(|u| \leq 1\)), any iterative procedure based on expansions does not appear to be applicable in such cases.'

It is shown in this Section that constraints of the form (61) can be included in the second-order analysis.

The following assumption is made:

The optimal control function \(u^0(t)\) is continuous on the whole interval \([t_o, t_f]\). That is, if and when a control hits or leaves a constraint, it
does so without a sudden jump. This is illustrated in Fig. 4. Times $t_a$ and $t_i$ are time at which the constraint becomes active and inactive respectively.

The above assumption is not overly restrictive; many control inequality constrained problems exhibit this behavior. An exception is the bang-bang problem where $H_{uu} = 0$ and the constraint $g$ is of the form $|u| \leq 1$. Solutions to this problem will be described in a future paper.

The r. h. s. of the D. D. P. equation (14) is:

$$\min_{\delta u} \left[ H(x + \delta x, u + \delta u, v_x; t) + \langle v_{xx} \delta x, f(x + \delta x, u + \delta u; t) \rangle \right]$$

(62)

As in Section 3, consider the case where $\delta x(t) = 0$. The minimization w. r. t. $\delta u$ must be carried out subject to constraint (61).

Define the set

$$U = \{ u : g(u; t) \leq 0 \}$$

(63)

then (62) becomes:

$$\min_{\delta u} \left[ H(x, \bar{u} + \delta u, v_x; t) \right]$$

(64)

$\bar{u} + \delta u \in U$

Let $\hat{u}$ be the control that minimizes $H$ w. r. t. $u$ subject to $u \in U$.

Expression (64) becomes

$$H(\bar{x}, \hat{u}, v_x; t)$$

(65)

(The constrained minimization in (64) is a non-linear programming problem. A useful method of solving it appears to be that of Fiacco and McCormick [13]. In control problems it often happens that there
FIG. 4 ILLUSTRATION OF A CONTROL INEQUALITY CONSTRAINT
are only a few controls and a few simple constraints on u; the minimization is then performed easily.)

If the minimization is performed and it turns out that no constraints are active (i.e., strict inequality holds, in (61), for \( u = \hat{u} \), so \( \hat{u} = u^* \)) then the algorithm is the same as that of Section 3.

Assume that \( \hat{p} \) of the constraints are active. (0 < \( \hat{p} \leq p \)) refer to them as \( \hat{g}(u; t) \).

\[ \hat{g}(\hat{u}; t) = 0 \]

(66)

Now adjoin (66) to (65) using a vector language multiplier \( \lambda \) of dimension \( \hat{p} \).

i.e.

\[ J(\bar{x}, \hat{u}, V_x, \lambda, t) = H(\bar{x}, \hat{u}, V_x; t) + \langle \lambda, \hat{g}(\hat{u}; t) \rangle \]

(67)

Under certain assumptions on \( H \) and \( \hat{g} \), given in Section 14, the following equations are necessary for determining \( \lambda \) and \( \hat{u} \):

\[ \frac{\partial J}{\partial u} = H_u(\bar{x}, \hat{u}, V_x; t) + \hat{g}_u^T(\hat{u}; t)\lambda = 0 \]

(68)

\[ \frac{\partial J}{\partial \lambda} = \hat{g}(\hat{u}; t) = 0 \]

(69)

Assume now that small variations \( \delta x \) in \( x \) are introduced at time \( t \), and that all the constraints \( \hat{g}(\hat{u} + \delta u; t) \) remain active. (i.e. assume that constraints \( \hat{g} \) are well and truly active; \( |\lambda_i| > 0 \), i = 1, \ldots, \hat{p} \). This will be true at all times \( t \) except those at which a constraint is just becoming active or inactive. (In Fig. 4 such times are \( t_a \) and \( t_i \).) Ignore such boundary points for the moment.

Re-introducing \( \delta x \) into (67):
\[
\begin{align*}
\min_{\delta u} & \left[ H(\bar{x} + \delta x, \hat{u} + \delta u, V_x, t) + < V_{xx} \delta x, f(\bar{x} + \delta x, \hat{u} + \delta u; t) > \\
& + \langle \lambda + \delta \lambda, \hat{g}(\hat{u} + \delta u, t) > \right] \\
(70)
\end{align*}
\]

\(\delta \lambda\) is present also, on the re-introduction of \(\delta x\), since the constraints \(\hat{g}\) at \(\hat{u} + \delta u\) are assumed to remain active, and so \(\lambda\) must change to \(\lambda + \delta \lambda\) to ensure this.

The following necessary conditions, analogous to (68) and (69), are obtained:

\[
H_u (\bar{x} + \delta x, \hat{u} + \delta u, V_x, t) + f_u^T (\bar{x} + \delta x, \hat{u} + \delta u; t) V_{xx} \delta x
\]

\[
+ \hat{g}_u^T (\hat{u} + \delta u; t) (\lambda + \delta \lambda) = 0
\]

and \(\hat{g} (\hat{u} + \delta u; t) = 0\)  

(71)

Expanding to first-order about \(\bar{x}, \hat{u}\) and using (68) and (69), the following equations result:

\[
(H_{uu} + \lambda \hat{g}_{uu}) \delta u + \hat{g}_u^T \delta \lambda = - (H_{ux} + f_u^T V_{xx}) \delta x
\]

(72)

\[
\hat{g}_u \delta u = 0
\]

(73)

All quantities evaluated at \(\bar{x}, \hat{u}; t\). (Reasons for expanding (71) to first-order only are similar to those discussed in detail in Section 3)

From the above equations:

\[
\delta \lambda = - \left[ \hat{g}_u (H_{uu} + \lambda \hat{g}_{uu})^{-1} \hat{g}_u \right]^{-1} \hat{g}_u (H_{uu} + \lambda \hat{g}_{uu})^{-1} (H_{ux} + f_u^T V_{xx}) \delta x
\]

(74)
and
\[ \delta u = - (H_{uu} + \lambda \hat{g}_{uu})^{-1} \left\{ I_m - \hat{g}_u^T \left[ \hat{g}_u (H_{uu} + \lambda \hat{g}_{uu})^{-1} \hat{g}_u^T \right]^{-1} \hat{g}_u \right\} (H_{ux} + f_u^T V_{xx}) \delta x \]

(75)

where \( I_m \) is the unit matrix of dimension \( m \), the number of controls.

Expanding (70) to second-order, substituting in expressions (74) and (75) for \( \delta \lambda \) and \( \delta u \), and using equations (68) and (69), the following expression is obtained:

\[
H + \langle H_x + V_{xx} f, \delta x \rangle + \frac{1}{2} \langle \delta x, [H_{xx} + f_x^T V_{xx} + V_{xx} f - (H_{ux} + f_u^T V_{xx})^T Z^T (H_{uu} + \lambda \hat{g}_{uu})^{-1} Z (H_{ux} + f_u^T V_{xx})] \delta x \rangle
\]

(76)

where
\[
Z = I_m - \hat{g}_u^T \left[ \hat{g}_u (H_{uu} + \lambda \hat{g}_{uu})^{-1} \hat{g}_u^T \right]^{-1} \hat{g}_u (H_{uu} + \lambda \hat{g}_{uu})^{-1}
\]

(77)

Equating (76) to the r.h.s. of (14), the following equations are obtained in the manner described in Section 3:

\[
\begin{align*}
\dot{\alpha} &= H - H(\bar{x}, \bar{u}, V_{x}; t) \\
\dot{V}_x &= H_x + V_{xx} (f - f(\bar{x}, \bar{u}; t)) \\
\dot{V}_{xx} &= H_{xx} + f_x^T V_{xx} + V_{xx} f - (H_{ux} + f_u^T V_{xx})^T Z^T (H_{uu} + \lambda \hat{g}_{uu})^{-1} \\
&\quad \cdot Z (H_{ux} + f_u^T V_{xx}) \\
u &= \hat{u} + \hat{\beta} \delta x
\end{align*}
\]

(78)

where
\[
\hat{\beta} = -(H_{uu} + \lambda \hat{g}_{uu})^{-1} Z (H_{ux} + f_u^T V_{xx})
\]

(79)

and \( \hat{u} \) is chosen by \( \min \) \( H(\bar{x}, u, V_{x}; t) \) which yields also the \( \hat{g}(\hat{u}; t) \),

\[
u \quad u \in U
\]

\( \lambda \) is given by (68) and (69).
Unless otherwise stated, all quantities are evaluated at $x, \hat{u}, t$. Boundary conditions are the same as before, namely Equations (32).

At times when no constraints are active, the above equations reduce to those of Section 3. (i.e. $Z = I_m$)

In the above derivations, $H_{uu} + \lambda \hat{g}_{uu}$ has been assumed positive-definite. This condition ensures that $u^0(t)$ is continuous on the interval $[t_0, t_f]$, as required earlier, and also that $J$ (Equation (67)) has an unconstrained relative minimum w.r.t. $u$ at $u = \hat{u}$. Since $H_{uu} + \lambda \hat{g}_{uu}$ is evaluated at $\hat{u}$ and not the nominal $\bar{u}$, global, strict convexity w.r.t. $u$ of $H + \langle \lambda, \hat{g} \rangle$, is not required. Local, strict convexity at $\hat{u}$ is sufficient: many problems exhibit this property.

If there is only one control then, from (73), $\delta u = 0$ if $\hat{g}_{uu} \neq 0$ and $Z = \hat{\beta} = 0$ even if $H_{uu} + \lambda \hat{g}_{uu}$ is non-positive. Note that in this case the Riccati Equation degenerates into a linear matrix equation.

At boundaries where a constraint ceases to be active or inactive, $Z$ will change discontinuously. However, $\hat{u}$ is continuous. It follows, then, that only $\hat{V}_{xx}$ suffers a discontinuity.

From (79), note that when running forwards and generating the new trial trajectory, $u(t)$ will be discontinuous at times of boundary points of the $\hat{g}$ owing to the presence of discontinuity in $Z$. However, this discontinuity does not affect the cost to second-order. This is intuitively so, and is proved in [6] and [14]. The proof, though simple, is omitted here because of its length. The discontinuity in the forwards $u(t)$ can be overcome easily, if desired, by using the computational trick of Section 7. Since $V_x$ and $V_{xx}$ are continuous the $u(t)$ so generated will be continuous. On an optimal trajectory $u = \hat{u} = u^0$ which is continuous.
The computational procedure for this algorithm is the same as that described in Section 5 except that the minimization of $H$ w. r. t. $u$ is done for $u \in U$. This produces $\hat{u}$ and $\hat{g}$. Using (68) and (69) $\lambda$ is then calculated, which enables $Z$ and $\hat{\beta}$ to be calculated using (77) and (79). The nominal control $\bar{u}(t)$ is assumed to satisfy $g(\bar{u}; t) \leq 0$.


1) Control inequality constraints of the type $g(u; t) \leq 0$ can be handled provided the optimal control function is continuous. It is believed that the algorithm is the only second-order method available that can treat these problems.

2) The procedure does not exhibit one step convergence for the L. Q. P. problem with linear control constraints because the optimal cost $V^O(x; t)$ for this problem is not quadratic.

3) The requirement that $H_{uu}(\bar{x}, \hat{u}, V_x; t) + \lambda_{gu} \left[ \hat{g}_{uu}(\bar{u}; t) \right]$ be positive-definite is rather restrictive. In a future paper the control problem where this matrix is identically zero, will be treated.


In this and the next section it should be remembered that $\hat{g}_u(\bar{u}; t) = 0$ and $\bar{u} \in U$.

Sufficient conditions to guarantee $a(\bar{x}; t_f) < 0$ and hence a reduction in cost for $\delta x$'s sufficiently small are that for $t \in [t_0, t_f]$:

1) $H(\bar{x}, \hat{u}, V_x; t) < H(\bar{x}, \bar{u}, V_x; t)$; $\hat{u} \neq \bar{u}$

2) $\hat{g}_u(\hat{u}; t)$ has full rank $\hat{p}$ and $\left[ \hat{g}_{uu}(\hat{u}; t) ; H_u(\bar{x}, \hat{u}, V_x; t) \right]$ has rank $\hat{p}$.

3) $\left[ H_{uu}(\bar{x}, \hat{u}, V_x; t) + \lambda_{gu} \left[ \hat{g}_{uu}(\hat{u}; t) \right] \right]^{-1}$ be positive-definite.

4) The solutions of the differential equations be bounded.
Proof:

\[ a(\vec{x}; t_1) = \int_{t_f}^{t_1} \left[ H(\vec{x}, \hat{u}, V_x; t) - H(\vec{x}, \bar{u}, V_x; t) \right] dt \]

A sufficient condition for \( a(\vec{x}; t_1) < 0 \) is clearly

\[ H(\vec{x}, \hat{u}, V_x; t) < H(\vec{x}, \bar{u}, V_x; t) \; ; \; \hat{u} \neq \bar{u}, \hat{u} \text{ and } \bar{u} \in U \]

From linear equation theory [15], [16] necessary and sufficient conditions for \( \lambda \) to be determined from (68) are that:

\[ g_u^T(\hat{u}; t) \text{ has full rank } \hat{p} \]

\[ [g_u^T(\hat{u}; t); H_u(\vec{x}, \hat{u}, V_x; t)] \text{ has rank } \hat{p} \]

For \( H(\vec{x}, u, V_x; t) + \lambda, \hat{g}(u; t) \) to have an unconstrained relative minimum w.r.t. \( u \) at \( u = \hat{u} \), which ensures \( a(\vec{x}; t_1) < 0 \), a necessary condition is from [16], that:

\[ H_{uu}(\vec{x}, \hat{u}, V_x; t) + \lambda g_u^T(\hat{u}; t) \text{ be positive definite.} \]

Further this allows the calculation of \( \hat{\beta} \) and, together with \( g_u^T(\hat{u}; t) \) having rank \( \hat{p} \), allows the calculation of \( Z \). Also, the continuity of \( \hat{u}(t); t \in [t_o, t_f] \), is assured.

Again, the differential equations are required to have bounded solutions.

15. Control Inequality Constraints: A New First-Order Algorithm.

As in Section 9, a first-order algorithm emerges as a special case of the second-order one.

\[ -\dot{a} = H - H(\vec{x}, \bar{u}, V_x; t) \; ; \; a(t_f) = 0 \]

\[ -\dot{V}_x = H_x \; ; \; V_x(t_f) = F_x(\vec{x}(t_f); t_f) \]

(80)
All quantities evaluated at $\bar{x}, \hat{u}; t$ unless otherwise stated.

The new control that is applied is $u(t) = \hat{u}(t); t \in [t_f, t_f]$.

The implementation and characteristics of the algorithm are similar to those of Sections 9 and 10.


Consider again the Rayleigh Equation, Example 2 of Section 11.

The following control constraints are introduced:

$$|u| \leq 1$$  \hspace{1cm} (81)

Note that $H_{uu} = 2$ and the constraints $g(u; t)$ are of the form:

$$u - 1 \leq 0 \text{ if } u > 0$$

$$u + 1 \geq 0 \text{ if } u < 0$$  \hspace{1cm} (82)

It is clear, therefore, that $H_{uu} + \lambda g_{uu}$ is positive for all $u$. It is seen easily that when the constraint is inactive $Z = 1$ and when active $Z = 0$ and the Riccati equation becomes a linear one.

Starting from the same nominal trajectory as before, the new second-order algorithm found the optimal solution in 3 iterations. Fig. 5 shows the cost as a function of iteration number and Fig. 6 shows the control function for various iterations. Note the jumps in the control along non-optimal trajectories and observe that they disappear when the optimal trajectory is reached.

It should be noted that the Riccati equation now has a bounded solution on the whole interval $[t_0, t_f]$, even along non-optimal trajectories. This is because, when $Z = \hat{\lambda} = 0$ along the constraint, it becomes a linear equation which cannot have an unbounded solution. The time interval over which $Z = 1$, $\hat{\lambda} \neq 0$ is too small for the Riccati equation to produce an unbounded solution.
FIG. 5 THE CONSTRAINED, RAYLEIGH CONTROL PROBLEM: \( V \) AS A FUNCTION OF ITERATION NUMBER
FIG. 6 THE CONSTRAINED, RAYLEIGH CONTROL PROBLEM: CONTROL FUNCTIONS FOR VARIOUS ITERATIONS.
17. **Equality End-Point Constraints and Implicity Given Final Time** $t_f$

The algorithms described in this paper can be extended to solve this class of problem. Complete details of the derivations are to be found in [6] and [14]. These extensions are not described in this paper, since the Lagrange multiplier techniques used in [6] and [14] are well known [3], [17]. The algorithms do, however, require the integration of less differential equations than those of [3].

18. **Conclusion.**

It is believed that the D. D. P. algorithms described in this paper are contributions in the field of 'Numerical Techniques for Solving Optimal Control Problems' in that they are able to handle a larger class of problems than was heretofore possible using second-variation successive approximation methods.

A future paper will describe D. D. P. techniques for solving bang-bang problems. It is hoped to extend the D. D. P. approach to study problems with state variable inequality constraints, and singular problems.
REFERENCES


In this paper, the notion of Differential Dynamic Programming is used to develop new second-order and first-order successive approximation methods for determining optimal control.

The unconstrained, non-linear control problem is first considered, and a second-order algorithm is developed which has wider application than existing second-variation and second-order algorithms. A new first-order algorithm emerges as a special case of the second-order one.

Control inequality constraints are introduced into the problem and a second-order algorithm is devised which is able to solve this constrained problem. It is believed that control constraints have not been handled, previously, in this way. Again, a first-order algorithm emerges as a special case.

The usefulness of the second-order algorithms is illustrated by the computer solution of three control problems.

The methods presented in this paper have been extended, by the author, to solve problems with terminal constraints and implicitly given final time. Details of these procedures are not given in this paper, but the relevant references are cited.
<table>
<thead>
<tr>
<th>KEY WORDS</th>
<th>LINK A</th>
<th>LINK B</th>
<th>LINK C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Differential Dynamic Programming</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dynamic Programming</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Optimization</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Successive approximation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sweep methods</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Second-variation</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>