Department of AERONAUTICS and ASTRONAUTICS
STANFORD UNIVERSITY

T. E. BULLOCK

COMPUTATION OF OPTIMAL CONTROLS BY A METHOD BASED ON SECOND VARIATIONS

120

DECEMBER 1966

Prepared for the National Aeronautics and Space Administration under Grant NsG 133-61

SUDAAR NO. 297

Department of Aeronautics and Astronautics
Stanford University
Stanford, California

COMPUTATION OF OPTIMAL CONTROLS BY A METHOD BASED ON SECOND VARIATIONS

by

T. E. Bullock

SUDAAR NO. 297
December 1966

Prepared for the National Aeronautics and Space Administration
Under Grant NsG 133-61

ABSTRACT

This work is concerned with finding an efficient computational scheme for the solution to general optimal control problems with terminal constraints. It is initially assumed that the control variables are unbounded. The results are later extended to include a class of problems with bounded controls.

Previous work on problems of this type may be classified into two groups, methods which seek iterative solutions to the Euler-Lagrange equations and those which iteratively improve initial guesses for control functions. The solution presented is of the second type.

The approach begins by showing how the control problem may be converted into a sequence of simpler control problems which admit analytic solutions. These simplified problems, which have linear dynamics and quadratic performance criteria, are studied in detail and optimal feedback control laws are obtained for them. In addition, tests which are sufficient to show the optimality of the resulting control are given. This study is closely connected with the theory of the second variation in the calculus of variations.

The final solution, in the form of a computational technique, is found by combining the method for generating a sequence of simplified control problems and their solution together with a method for automatically adjusting several parameters necessary to insure convergence. The resulting algorithm requires very little computational heuristics in actual machine calculations. Since the method is second order, convergence is considerably improved over the usual gradient techniques. Former difficulties with other methods including small regions of

convergence and difficulties associated with conjugate points in the local accessory problem have been eliminated. The control law is generated in the form of a time function plus a linear time varying state variable feedback and may be used in a neighboring extremal guidance control scheme. Furthermore, tests are performed which are sufficient to show that the resulting control is optimal.

Several numerical examples are included to illustrate the application of the method in actual problem solution.

ACKNOWLEDGMENT

The author would like to express his appreciation to Professor G. F. Franklin for his excellent supervision of this research and to Professor J. V. Breakwell for his invaluable advice and uncanny ability to find the missing answer when all is lost. He is grateful to Professor V. R. Eshleman for a critical evaluation of the manuscript. A debt of gratitude is also owed to the many fellow graduate students who furnished many stimulating discussions. Two of the author's colleagues, P. H. Haley and D. R. McNeal, deserve special credit in this regard.

The entire staff of the Stanford Electronics Laboratory has been very helpful, far beyond the call of duty, in giving assistance and moral support. The financial support provided for by NASA under Research Grant NsG-133-61 is acknowledged with gratitude.

TABLE OF CONTENTS

Chapter		Page
I.	INTRODUCTION	1
	A. Motivation for the Study of this Problem	1
	B. Survey of Previous Work	3
	C. Outline of Results	8
II.	STATEMENT OF THE PROBLEM	11
	A. System Description	11
	B. Problem Statement	12
	C. Special Cases	14
	D. Control and State Space Constraints	19
III.	DIRECT METHODS OF SOLUTION	21
	A. Comparison of Direct and Indirect Formulations	21
	B. Gradient Type Methods	26
	C. Second-Order Methods	39
IV.	A DIRECT METHOD BASED ON SECOND VARIATIONS	43
	A. Derivation of the Method	43
	B. Extension to Bang-Bang Optimal Control	49
	C. Problems with Control Parameters	55
	D. Problems with Free Final Time	57
V.	THE SOLUTION OF THE AUXILIARY PROBLEM	60
	A. Problem Statement	60
	B. Case I - Problems with Free End Conditions	66
	C. Case II - Fixed Endpoint Problems	68
	D. Case III - General Linear End Constraints	70
	E. Sufficiency Conditions	77
VI.	THE COMPUTATIONAL METHOD	84
	A. Outline of the Computational Technique	84
	B. Extension to Other Types of Problems	96
	C. Properties of the Solution	111
	D. Suggestions for Coding	117
VII.	NUMERICAL EXAMPLES	124
	A. Linear Plant Quadratic Loss Example	125
	B. The Brachistochrone	130
	C. Quadratic Loss Van Der Pol with Free Endpoint	138
	D. Van Der Pol to a Line	148

TABLE OF CONTENTS (CON'D)

Chapter		Page
VIII.	CONCLUSIONS	154
	A. Summary of Results	154
	B. Suggestions for Future Research	155
APPENDIX A.	PROOFS OF THEOREMS OF CHAPTER III	158
APPENDIX B.	SAMPLE PROGRAM	164
APPENDIX C.	DETAILS OF NUMERICAL EXAMPLES	172
APPENDIX D.	PROPERTIES OF THE FUNDAMENTAL MATRIX FOR THE EULER-LAGRANGE EQUATIONS	177
REFERENCES		181

LIST OF ILLUSTRATIONS

Figure		Page
4.1	Effect of a Negative Shift in the Switching Time	52
6.1	Simplified Flow Chart for Computing Optimal Controls Using Second Variations	85
6.2	Summary of Results for the Accessory Problem	99
6.3	Summary of Results for the Accessory Problem when the Nominal Trajectory is an Extremal	114
7.1	Optimal Trajectories for $1/(s^2 + 1)$ Plant with Quadratic Loss $Q_1 = I$, $Q_2 = 1$, $Q_3 = 0$ and Free-End Conditions	127
7.2	Successive Control Iterations Using Steepest Descent, Example A	128
7.3	Adjoint Variables, Example A	129
7.4	Solution to Riccati Equation for Example A.	130
7.5	Brachistochrone, Example B	131
7.6	Trajectory Iterations by Second Variations for the Brachistochrone Problem, Example B	137
7.7	Control Iterations using Steepest Descent Initialized with $u(t) = 0$ for the Van Der Pol Problem, Example C	142
7.8	Control Iterations using Second Variations Initialized with $u(t) = 0$ for the Van Der Pol Problem, Example C	142
7.9	Control Iterations using Steepest Descent Initialized with $u(t) = 1$ for the Van Der Pol Problem, Example C	143
7.10	Control Iterations using Second Variations Initialized with $u(t) = 1$ for the Van Der Pol Problem, Example C	143
7.11	Optimal Trajectories for the Driven Van Der Pol Equation with an Integral Quadratic Loss Function	146
7.12	The Time Varying Feedback for the Neighboring Extremal Control Law, Example C	146
7.13	A Comparison of the Relative Rates of Convergence for Steepest Descent (SDVP) and Second Variations (2VVP)	148
7.14	Phase Plane Plot of Several Iterations for the Van Der Pol to a Line Problem, Example D	153
7.15	Neighboring Extremal Control Law for the Van Der Pol to a Line Problem, Example D	153

I. INTRODUCTION

A. MOTIVATION FOR THE STUDY OF THIS PROBLEM

The principal goal for the research effort presented here is the development of an efficient practical scheme for numerically solving optimal control problems. Although the modern approach to optimal control was begun a decade ago by a group of Russian mathematicians lead by L. S. Pontryagin [1956], [1962], its applications have been largely limited to simple problems which have analytic solutions. This situation was beginning to change when Breakwell [1959] proposed a computational scheme for numerically solving an optimal control problem with the aid of a computer. Since Breakwell's initial efforts, several other investigators have dealt with the problem of efficiently generating numerical solutions to problems of this nature.

In order to illustrate a typical problem which requires numerical solution, consider the following example. Suppose it is required to put a payload into orbit around the earth with a suitable boost vehicle. The direction of the vehicle is to be controlled by adjusting the direction of thrust of the engine. For a fixed engine design, how may the thrust direction be programmed so as to maximize the final altitude in orbit?

This problem is typical of optimal control problems. It requires the determination of a function of time, the control variable (the thrust direction), so that some functional is extremized (the final altitude), and that certain constraints are met. (The payload is placed in orbit.)

At this point, the problem presented may be interpreted as an optimal control problem with no analytic solution in the general case. The need for a solution to this problem other than as an academic exercise still requires demonstration. Perhaps the most important reason for the generation of numerical answers is to study the nature of optimal solutions. We may then use these solutions as guidelines for the design philosophy used for engineering solutions. Another important value of the numerical results might be to act as a standard to judge the performance of some suboptimal scheme which has been designed with hardware implementation in mind. The most obvious application of numerical solutions is to act as the actual control scheme for a vehicle. In practice, this idea is not too useful since the usual solutions are open loop. That is, the solutions generated only solve one example of a simplified model of the physical system with a fixed set of initial conditions and terminal constraints. The more desirable solution uses some measure of the system's state as feedback so that the control program may be altered to provide optimal performance for the calculated example and near optimal results for slightly different situations. By this technique, it is possible to construct a suboptimal control system which is similar to the more conventional linear feedback control system designed by classical methods. The construction of numerical controls which are of the form $u(t) = c'(t)\, x(t) + c_{n+1}(t)$* which are optimal for the given problem and demonstrate near optimal performance for similar problems is also considered in this study.

*For notation, we let u,c,x be vector or column matrices. c' is the transpose of the matrix c.

B. SURVEY OF PREVIOUS WORK

In order to put the present work in its proper perspective, a brief history of related research will be given. Although the field is relatively new, the widespread use of computers in control research, particularly in relation to aerospace problems, has accelerated the work so that many different computational techniques are now available. It is not the intention to cover all of the related work here, but to try to include the most significant ideas without discussing each method in detail.

The usual methods for solving variational problems in the Calculus of Variations lead to the reduction of the problem to one involving a set of differential equations. This set of differential equations may not be solved in a straightforward manner because the boundary conditions are specified on the boundary of some region R. In optimal control problems, R is usually an interval of values for the independent variable so that its boundary consists of two points. For such a problem, the set of differential equations and its boundary conditions are commonly called the Two-Point Boundary Value Problem, (TPBVP). The difficulties in solving the set of differential equations have led to a search for variational methods of a different kind, known as direct methods, which circumvent the problems associated with the differential equation solution. These methods, which are discussed in most modern books on the Calculus of Variations,* are based on finding a sequence of functions which give successively smaller values to the functional to be minimized.

*cf., Gelfand and Fomin [1963], Chapter 8.

Two of the classical direct methods are the Ritz method and the method of finite differences.

Following the distinction given in the Calculus of Variations, literature on computational techniques normally divides methods into two types, the direct methods and the indirect methods. Methods based on finding solutions to the derived TPBVP are generally referred to as indirect, and methods which directly construct minimizing sequences for the functional to be minimized are called direct. This dichotomization is often confusing, for frequently a method seems to have some of the characteristics of both techniques. For example, two-point boundary value problems of the kind normally associated with indirect methods are commonly encountered in second-order direct methods and furthermore construction of minimizing sequences, a technique that distinguishes direct methods, is often employed in solving the TPBVP in the indirect methods.

The basis for the indirect method involves finding the unknown boundary values at one point so that the resulting solution to a set of differential equations, the Euler-Lagrange equations, will satisfy the required boundary conditions at a second point. By regarding the boundary values at the second point as functions of the unknown boundary conditions at the first point, the problem becomes one of finding the values of the variables x_1, x_2 . . . which make several functions of these variables $f_1(x_1, x_2,\ldots,x_j),\ldots,f_k(x_1, x_2,\ldots,x_j)$ take on specified values. The approach used by Breakwell [1959] was to evaluate the functions $f_1, f_2,\ldots,f_k$ for several selected perturbed values of the variables $x_1, x_2,\ldots,x_j$, to fit a suitable polynomial approximation to each of the functions using the measured points, and to adjust the

variables $x_1, x_2, \ldots, x_j$ based on the polynomial approximation to the nonlinear functions. This technique essentially uses a form of numerical differentiation by means of finite differences. Other methods have been developed in which the required derivatives are computed analytically, thus hopefully avoiding the errors inherent in numerical differentiation. Although these methods differ in the details, they all effectively linearize the TPBVP, solve the linearized version by various techniques, and use the solution to adjust the boundary conditions for the nonlinear TPBVP. Some examples of this type of approach are found in Breakwell, Speyer, and Bryson [1963], Jazwinski [1964], and Payne [1965]. The chief characteristic of these methods is rapid convergence if they converge at all. The requirement of relatively good initial values of the parameters to be adjusted to insure convergence has led to the development of guides for choosing good initial guesses. These methods have been highly successful when the user is fairly resourceful in generating good initializing boundary values.

Direct methods are normally distinguished by the characteristic of not requiring good starting values to insure that an improved path may be found. The first methods, such as the Ritz method, attempt to minimize the functional by expressing the trajectory or the control, as an expansion in terms of a weighted sum of a suitable set of functions and finding the minimizing set of coefficients. Methods of this type have not been too popular in application to optimal control problems primarily due to the difficulties in finding a suitable set of basis functions and in determining the number of terms in the expansion to use except by experimentation. A second type of direct method is Bellman's dynamic programming [Bellman 1957], which is an efficient

sequential search scheme for determining optimal paths. The technique of dynamic programming is sufficiently different from the other methods so that further detailed discussion is beyond the intended scope of this study. Dynamic programming has the advantage of being simplified by state space and control constraints, of having the ability to include nonanalytic system descriptions, such as tabular data, and of generating entire families of optimal trajectories for problems with different initial and boundary conditions. Its primary disadvantage is the requirement for an excessive amount of computer memory, thus limiting its application to problems with a small number of state variables. Larson [1964] has presented a method for reducing the required memory for problems with a continuous independent variable which has the effect of increasing the range of problems for which computation by means of dynamic programming is feasible.

A significantly different type of direct method, known as the gradient method, was developed by Kelly [1960] and later by Bryson and Denham [1962]. The gradient methods have the ability to generate successively improved trajectories even with very poor starting values. However, they tend to converge slowly, particularly in the final stages of the iteration, and require the selection and subsequent adjustment of several convergence parameters. Several investigators have presented schemes for improving the convergence rate and for avoiding the selection of the somewhat arbitrary convergence parameters. (See, for example, Brown [1964], Rosenbaum [1963], and Stancil [1964].) Initial studies by Sinnott [1966] have indicated that the method of conjugate gradients in a function space shows considerable promise as a gradient-type method with improved speed of convergence.

The gradient method is essentially a first-order method since it is based on finding the first-order effects of the control on the functional to be minimized and the terminal constraints. In an attempt to accelerate the convergence of the gradient method, second- and higher-order direct methods were investigated by Merriam [1964], [1965]. Merriam's parameter expansion technique was developed for this purpose. A scheme with similar results was later given by Kelly, Kopp, and Moyer [1964]. Due to the similarity of the results obtained by Kelly, Kopp, and Moyer and the theory of the second variation in the Calculus of Variations, the direct second-order methods are often called methods based on second variations. These methods achieve the goal of improved rates of convergence at the expense of losing several of the desirable features of the gradient method. The primary difficulty is the necessity of again initializing the program with fairly good guesses of the control law. Also, Merriam's method provided no means for meeting the specified terminal conditions exactly. Merriam [1964] and Kelly, Kopp, and Moyer suggest that a gradient type method be employed until the convergence begins to slow and then be changed to a second-order method to accelerate the convergence. McReynolds and Bryson [1965] give a direct second-order method which includes a feedback solution to a linear TPBVP which must be solved as a part of the method.

Another type of method for computing optimal controls, known by various names as quasilinearization, differential approximation, or a generalized Newton-Raphson method, is, strictly speaking, an indirect method. However, it is considerably different from the other indirect methods. Conventional indirect methods solve the TPBVP by iteratively adjusting the unspecified boundary conditions. By quasilinearization, a set of functions is iteratively adjusted by solving a sequence of linear TPBVP's so that they converge to a solution of the nonlinear

TPBVP. A comparison of quasilinearization with some inefficient versions of the gradient and second variations techniques may be found in Kopp and McGill [1964] with numerical results in Moyer and Pinkham [1964]. Van Dine [1965] has combined quasilinearization with a finite difference scheme for eliminating the instability problems in solving the necessary linear TPBVP's. An application of Van Dine's technique to an aerospace control problem is found in Van Dine, Fimple, and Edelbaum [1965]. McGill [1965] has used penalty functions to extend the method of quasilinearization to problems with state inequality constraints. Kenneth [1965] has used a technique due to Valentine [1937] to include bounded control in a computing method based on quasilinearization. Although the general technique has very rapid convergence, it still has a limited region of convergence and requires sufficiently good initializing functions.

C. OUTLINE OF RESULTS

Merriam's work was the starting point for the research reported here. The result has been the development of a numerical method of the direct type which has the following characteristics:

1. The region of convergence is effectively as large as that of the usual gradient approach.
2. The convergence rate corresponds to that of gradient methods with feedback correction initially and to the rapid second-order methods as the minimum is approached.
3. Although a set of initial convergence type parameters must be specified as in the gradient methods, these parameters are

automatically adjusted by the program. A poor guess does not prevent convergence, but only slows it initially.

4. Adequate tests are performed without additional computation which are sufficient to show that the solution must be a minimizing curve. (Sufficiency test in the Calculus of Variations.)

5. The linear time-varying feedback coefficients for the so-called neighboring extremal control scheme are available without further calculations.

6. Terminal constraints are met "exactly," without the use of penalty functions.

The material to be presented is divided into eight chapters. Following the introduction in the first chapter and the problem statement and introductory material in the second chapter, the third chapter outlines, from a general point of view, the basic concepts involved in computing constrained and unconstrained extrema. Chapter IV uses the results of Chapter III to convert the computational problem into a sequence of linear control problems which have quadratic loss functionals. A feedback control solution to the linear plant, quadratic loss, control problem with general linear terminal conditions which guarantee that the solution obtained is optimal is also included. In Chapter VI, all of the previous results are combined to obtain the computational method. Several numerical examples are given in the following chapter as a demonstration of the value of the method in actual problem solution. Following the conclusions in Chapter VIII, a number of appendices are given as supplementary material which include a sample computer problem listing, some additional numerical details for the examples given in

Chapter VII, and a derivation of some useful properties of fundamental matrices for the Euler-Lagrange equations which are used in Chapter V.

II. STATEMENT OF THE PROBLEM

This chapter contains definitions of the notation to be used throughout this work and a precise statement of the mathematical control problem to be considered. In the last section, a number of special cases are enumerated for special study.

A. SYSTEM DESCRIPTION

The usual description of the system to be controlled is given by the vector differential equation

$$\dot{x}(t) = f[x(t), u(t)] \tag{2.1}$$

where $x(t)$ is an $(n \times 1)$ real vector of time functions hereafter called the state vector, $u(t)$ is an $(m \times 1)$ vector of functions called the control, $f(\cdot,\cdot)$ is an $(n \times 1)$ vector valued function of its arguments, and t is the independent variable usually identified with time.*

Although any dynamical system may be described by an equation of the type (2.1), it is perhaps necessary to note that for a general n^{th} order differential equation, this is not the case. For example, if the differential equation is given by

$$G(y, \dot{y}, \ddot{y}, \ldots, y^{(n)}, u) = 0 \tag{2.2}$$

where the y's are scalar time functions, there may not be an

*Problems in which the function f depends explicitly on the independent variable, t, may be considered by adding an additional state variable x_{n+1} which satisfies $\dot{x}_{n+1} = 1$, $x_{n+1}(0) = t_o$.

equivalent representation of the form (2.1). However, if (2.2) has a solution for $y^{(n)}$ as

$$y^{(n)} = y^{(n)}[y, \dot{y}, \ldots, y^{(n-1)}, u] \tag{2.3}$$

then there is no difficulty. It will be assumed that any system to be studied has a representation as in (2.1) which is called a state space representation.

The vector function u may contain a set of system parameters as well as a vector valued time function. For example, one control variable might be the staging time for a multistage rocket. By consideration of this more general class of controls, a wider range of problems may be studied without loss of generality.

B. PROBLEM STATEMENT

The control problem may now be stated as follows:

Control Problem: For the system described by (2.1) and the set of initial conditions

$$x(t_o) = x_o \qquad \text{(specified)} \tag{2.4}$$

find a vector control function $u \in U$, the class of admissible control functions, such that at some time $t_f > t_o$ the scalar payoff function $\varphi[x(t_f)]$ is minimized and the $(q \times 1)$ vector terminal constraints

$$\psi[x(t_f)] = 0 \tag{2.5}$$

are satisfied.

Given in this form, the control problem is identical to the problem of Mayer in the classical Calculus of Variations with a differential

subsidiary condition (the differential equation (2.1)). It is well known that problems in which the payoff function is of the Lagrange form

$$J = \int_{t_o}^{t_f} \ell(x, u)\, d\sigma \tag{2.6}$$

may be converted to the Mayer problem by defining an additional state variable x_{n+1} which satisfies

$$\dot{x}_{n+1} = \ell(x, u) \tag{2.7}$$

$$x_{n+1}(0) = 0$$

The payoff becomes

$$\phi[x(t_f)] = x_{n+1}(t_f) \tag{2.8}$$

In a similar manner, mixed problems of the Bolza form

$$J = \int_{t_o}^{t_f} \ell(x, u)\, d\sigma + \phi[x(t_f)] \tag{2.9}$$

may be written in the Mayer form without the integral cost function.

In order to insure that the solution to the problem may actually by computed, it is necessary to redefine the problem slightly. The actual question to be answered is "How may an optimal control be calculated?" For further practical reasons, only direct methods will be considered. This reasoning leads to a reformulation as follows: given a nominal control function $u(t)$ (and the corresponding trajectory),

construct a new control which is "better" in some sense. A more precise statement is given as the Computational Control Problem.

Computational Control Problem: For the system described by (2.1) and the initial conditions (2.3), let $x^o(t)$, $t \in [t_o, t_f]$ be the solution or trajectory for a given nominal control function $u^o(t)$. Find a vector control $u(t) \in U$, the class of admissible control functions, such that either the change in payoff $\Delta\varphi$ obeys

$$\Delta\varphi = \varphi[x(t_f^*)] - \varphi[x^o(t_f)] < 0 \tag{2.10}$$

and the terminal constraint functions satisfy $|\psi_i[x(t_f^*)]| \leq \epsilon_i$ if $|\psi_i[x^o(t_f)]| < \epsilon_i$ or, if $|\psi_i[x^o(t_f)]| < \epsilon_i$ is not satisfied, then

$$|\psi_i[x(t_f^*)]| < |\psi_i[x^o(t_f)]| \qquad i = 1,2,\ldots,q-1 \tag{2.11}$$

for suitably determined error bounds on the constraints ϵ_i, $i = 1,\ldots,q$. The terminal time t_f^* for the new trajectory $x(t)$, (obtained by solving (2.1) with initial conditions (2.3) and control $u(t)$) is determined from the stopping condition

$$\psi_q[x(t_f^*)] \triangleq \Omega[x(t_f^*)] = 0 \tag{2.12}$$

C. SPECIAL CASES

Although the problem statement given in the last section may be solved in general, there are several special cases which have the advantage of easier solutions. These simplifications may be made for more restrictive types of boundary conditions specified by the functions $\psi_i[x(t_f)]$, $i = 1,2,\ldots,q$.

The first simplification occurs when the stopping condition

$\Omega[x(t_f)]$ is of the form

$$\psi_q[x(t_f)] = \Omega[x(t_f)] = (t_f - b) = 0 \qquad (2.13)$$

for a specified constant b. With this restriction the problem is known as a Fixed Final Time* problem. Actually this special form for the stopping condition does not eliminate much of the formal difficulty except for some tedious algebra. However, the Free Final Time problem leads to programming complications in the actual computation. This is due to the necessity of storing time functions on the time interval $[t_o, t_f]$. That is, the time functions are stored in the form of a sequence of k sample points $f(t_i)$, for $i = 1,2,\ldots,k$. If the storage points are not uniformly spaced, it is necessary to store the set of storage times $\{t_i\}$. A considerable saving both in machine and programming time can be obtained by assuming that the number of points stored, k, and the set of storage times $\{t_i\}$ remain fixed from one iteration to the next. Of course, many problems of interest have the final time specified. Other problems may be converted to fixed interval problems by a change of the independent variable. For these reasons, the assumption of a fixed interval will usually be made for convenience with an indication of the modification for the more general case.

Several other problem simplifications can occur depending on the nature of the constraints $\psi_i[x(t_f)]$, $i = 1,2,\ldots,q-1$. In order to

*Although time is assumed to be the independent variable in the differential equation, of course this is not necessary. With this understanding, the independent variable will be called time to agree with common usage in the literature.

discuss these simplifications, it is necessary to consider the tangent plane to the constraint $\psi = 0$ given by

$$\sum_{j=1}^{n} \frac{\partial \psi_i[x(t_f)]}{\partial x_j} \Delta x_j = 0, \quad i = 1,\ldots,q-1 \tag{2.14}$$

Before discussing the simplifications, since summations of the form in (2.14) will appear frequently here and in later chapters, it is expedient to introduce a more compact notation at this point. The use of the usual matrix notation allows (2.14) to be written as

$$A\Delta x = 0 \tag{2.15}$$

Unfortunately, the notation for what the matrix A means in this case is not completely standard. The system adopted here will be to write the matrix A with a_{ij} representing the element of the i^{th} row and j^{th} column as

$$\psi_x = [a_{ij}] = \left[\frac{\partial \psi_i[x(t_f)]}{\partial x_j}\right] \tag{2.16}$$

This method has the distinct advantage of being the shortest possible without loss of too much of the important information. It has the disadvantage of being not completely standard and requiring more knowledge on the part of the reader. The chief point to remember is that ψ_x represents a matrix in which the i^{th} row is the collection of partial derivatives of the i^{th} row of the vector ψ.

Another frequently required expression is the matrix of second partial derivatives of a scalar quantity. These are the matrices whose elements are given by

$$b_{ij} = \frac{\partial^2 f}{\partial x_i \partial y_j} \tag{2.17}$$

$i = 1,2,\ldots,n_r$ (rows)

$j = 1,2,\ldots,n_c$ (columns)

where f is a scalar function of the vectors x $(n_r \times 1)$ and y $(n_c \times 1)$. The abbreviated notation for this matrix is

$$B = f_{xy} \tag{2.18}$$

It follows from this definition that

$$B' = f_{yn} \tag{2.19}$$

Now that the simplified notation has been introduced, the several special types of boundary conditions will be discussed. These special cases are distinguished by the dimension of the subspace described by the tangent plane to the terminal manifold $\psi[x(t_f)] = 0$. We assume the subspace

$$T : \{\Delta x | \psi_x[x(t_f)] \quad \Delta x = 0\} \tag{2.20}$$

has dimension r.

If $r = n$, the problem has a <u>free endpoint</u>. This situation provides the most straightforward solution. Since the methods of solution in this case are simplified, often problems with end constraints are converted to approximate free endpoint problems by the following technique. We define a new cost functional $\varphi_n[x(t_f)]$ related to the original cost $\varphi[x(t_f)]$ and the terminal constraint vector $\psi[x(t_f)]$ by

$$\varphi_n[x(t_f)] = \varphi[x(t_f)] + Mg\{\psi[x(t_f)]\} \tag{2.21}$$

where $g(\cdot)$ is a suitable penalty function of its vector argument and M is a positive number. In order for $g(\cdot)$ to be suitable, it should be a positive function which vanishes only when $\psi[x(t_f)] = 0$. In practice, this technique involves a careful choice of the form of the function $g(\cdot)$ and, more important, of the relative weighting given to errors in the end constraints as compared to changes in the original cost functional $\varphi[x(t_f)]$. If the value of the constant M is sufficiently large, the vector which minimizes φ_n will be close to the vector which minimizes φ and satisfies the constraints, $\psi = 0$. Unfortunately, very large values of M often lead to numerical difficulties in the computer solution and some compromise must be made. A complete study of the relative merits of the penalty function and exact methods for handling constraints is yet to be carried out. The exact method was chosen to be used in Chapter IV primarily to eliminate another arbitrary choice, that of the penalty function.

The second case, $r = 0$, is called the fixed endpoint problem. The most common example of this case is the problem in which all of the terminal states are required to have specified values. It will turn out that the fixed endpoint problem has sufficient structure so that a simplified computational scheme may be used as compared to the general case.

The third case is the most general one. When $0 < r < n$, the problem has partially specified end conditions. Actually the solution in this case includes the first two cases. It is never used for free or fixed endpoint problems since the computations for those cases require fewer equations.

D. CONTROL AND STATE SPACE CONSTRAINTS

For some problems, the set of admissible controls U, is closed. An example might be a rocket in which the control variable is the thrust level. Because the thrust level has a physical upper bound u_m, U is the set of all functions u defined on the interval $[t_o, t_f]$ for which $|u| \leq u_m$. In general the set U will depend on time and the state $x(t)$. Problems of this type are said to have Control Variable Constraints and demand special consideration.

Another type of constraint is obtained when there are physical limitations on the values of the state vector. To avoid an unrealistic solution to a rocket trajectory problem, we might require the solution to have positive altitude. Otherwise, the optimal solution might require an initial dive below the surface of the earth!! This type of constraint may be given in the form of r inequalities of the form

$$s_i[x(t), u(t), t] \leq 0 \qquad i = 1,\ldots,r \tag{2.22}$$

When these relations may be solved for the control u in terms of the state and time, they reduce to control variable constraints. Otherwise, the problem is said to have State Space Constraints.

The penalty function approach may also be applied to control variable constraints as well as to state space constraints. The only modification of the idea previously discussed is the addition of integral-type penalty functions to the cost of the form

$$k_i \int_{t_o}^{t_f} g_1\{s_i[x(\sigma), u(\sigma), \sigma]\}\, d\sigma \tag{2.23}$$

The function $g_1(\cdot)$ is chosen to be very large if any of the s_i's are positive. A similar integral penalty function for bounded controls may be used. The penalty function given by

$$\int_{t_o}^{t_f} |u(\sigma)|^k \, d\sigma \tag{2.24}$$

will cause $|u(t)| \leq 1$ for k large.

While the penalty function technique still may have its computational difficulties for this type of problem, it has increased attractiveness due to the additional complications of the "exact" method.

It is often possible to eliminate constraints by a clever change of variables. In problems for which $|u| \leq 1$, the control variable u may be replaced by the unbounded variable v by the transformation

$$u = \sin(v) \tag{2.25}$$

If the optimal u is "bang-bang" (i.e., the control is always on the boundary), it may perhaps be described by a time function switching between limits. The new unconstrained "control variable" may be defined as the set of numbers which specify the switching times.

III. DIRECT METHODS OF SOLUTION

The Computational Control Problem posed in the last chapter is by no means the only possible formulation. Further background for the problem solution will be developed in this chapter which will illustrate other approaches to the problem of computing optimal controls. The main part of this chapter will discuss the direct method for solving constrained and unconstrained minimization problems in a fairly general framework.

A. COMPARISON OF DIRECT AND INDIRECT FORMULATIONS

In order to compare the direct and indirect methods, it will first be instructive to consider a simple example. Suppose the minimum of a scalar function $f(x)$ depending on the vector $x \in R_n$ is desired. A necessary condition at the minimum is the set of n nonlinear equations

$$f_x(x) = 0 \tag{3.1}$$

It is usually rare that the set of equations (3.1) admits an easy solution. One might now propose some iterative technique for solving this set of equations. This is the <u>indirect</u> method for problem solution since the minimization is indirectly done through the solution to a set of nonlinear simultaneous equations. The usual iterative methods used to solve equations like (3.1) involve finding a relation

$$x_{n+1} = g(x_n) \tag{3.2}$$

which satisfies

$$\left|f_x(x_{n+1})\right| < \left|f_x(x_n)\right| \tag{3.3}$$

where the last equation represents a component by component inequality. Thus the indirect method typically converts a given minimization problem to a related but different minimization problem. The solution to the original problem is obtained indirectly by solving the related problem.

One might suspect that if a technique is available for generating a minimizing sequence $\{x_i\}$ for the components of $|f_x(x_n)|$, a similar technique could be used to minimize the original function $f(x)$ directly. This leads to the direct problem formulation.

By the direct method, a relationship of the form

$$x_{n+1} = h(x_n) \tag{3.4}$$

is used recursively to construct the minimizing sequence $\{x_n\}$ so that

$$f(x_{n+1}) < f(x_n) \tag{3.5}$$

In the control problem, the corresponding set of necessary conditions to (3.1) were given by L. S. Pontryagin [1956], [1962]. Pontryagin's Minimum Principle,* which gives the control law $u(t)$ in terms of the solution to a nonlinear two point boundary value problem, may be stated as follows.

Pontryagin's Minimum Principle

For the control problem, there exists a vector valued function $\lambda(t)$, not identically zero, and a set of numbers, $\nu_0 \geq 0$ and the vector ν, not all vanishing, which satisfy for $t \in [t_o, t_f]$

n State Equations $\dot{x} = H'_\lambda(\lambda, x, u)$

*Pontryagin stated his result as a Maximum Principle but it is common practice now to use the Minimum Form in the conditions. Only a change in sign is involved in the equations.

n Adjoint Equations $\dot{\lambda} = - H'_x(\lambda, x, u)$

n Initial Conditions $x(0) = x_0$

q Terminal Constraints $\psi[x(t_f)] = 0$

n Adjoint Boundary Conditions $\lambda'(t_f) = \nu_0 \phi_x[x(t_f)] - \nu'\psi_x[x(t_f)]$

m Control Equations $u = \min_{u \in U}^{-1}[H(\lambda, x, u)]$ (3.6)

Where the Hamiltonian $H(\lambda, x, u) = \lambda'(t)\, f(x, u)$ (3.7)

The notation $\min_{u \in U}^{-1}$ is used to denote a function $u \in U$ which minimizes H. If the final time t_f is not fixed, there is an additional relation

$$H[\lambda(t_f), x(t_f), u(t_f)] = \nu_0 \dot{\phi}[x(t_f)] - \nu'\dot{\psi}[x(t_f)] = 0 \ . \tag{3.8}$$

Also, if $f(x, u)$ is discontinuous at a set of points $t = t_i$ to be chosen in an optimal fashion, then

$$H(\lambda, x, u)\Big|_{t_i^-} = H(\lambda, x, u)\Big|_{t_i^+}$$

The set of relations (3.6), (3.7), and (3.8) are necessary for a solution to the optimal control problem.

In order to construct an indirect computational scheme from this set of necessary conditions, several assumptions will be made for simplicity. First, assuming $\nu_0 \neq 0$*, take $\nu_0 = 1$ without loss of generality. Next assume it is possible to find an explicit solution of the control equation which gives $u(t) = u[x(t), \lambda(t)]$. Substituting this equation into the adjoint and state equations reduces

*When $\nu_0 \neq 0$ the problem is said to be normal.

the necessary conditions to a set of $2n$ differential equations. As a final assumption, suppose the q terminal constraints and n adjoint boundary conditions can be solved to specify the terminal values of n of the variables $\lambda(t_f)$, $x(t_f)$. Call the n specified terminal variables y and the n remaining terminal variables z.

By integrating the state and adjoint equations backwards it is possible to compute the initial state $x(0)$ which is a function of the unknown terminal variables z. That is

$$x(0) = g(z) \tag{3.9}$$

The computational problem is now solved by specifying a method for finding the vector z so that the initial conditions $x(0)$ match the specified initial conditions x_0. The usual techniques involve defining a scalar error function which measures the distance between $x(0)$ and x_0. The problem then becomes a finite dimensional minimization problem of this error in n variables. Thus again the indirect method has converted the original minimization problem to another related minimization problem.

One obvious advantage of this approach is the large reduction in dimension. The original minimization problem required searching an infinite dimensional function space as compared with the auxiliary minimization problem in which the dimension of the parameter space is equal to the number of state variables. In addition to the conceptual advantage of this method, the theory for finite dimensional minimization is quite well developed and can be applied. Programs for this type of solution are relatively easy to write around a general purpose differential equation solving routine. Since the bulk of the calculation is

differential equation solution with little storage and logic needed, iterative analog or small hybrid-analog-digital computers may be used to implement the method.

At this point it is reasonable to question the value of the direct methods. Perhaps the primary consideration is one of convergence. The indirect methods require good guesses of the vector z in order to insure convergence. These guesses are frequently difficult to obtain from prior physical knowledge of the problem. On the other hand, the direct methods require an initial choice of the control function $u(t)$ which is usually obtainable from experience with the physical problem. When compared with the indirect methods, there are relatively no convergence difficulties in the direct methods due to bad initialization. A further practical matter concerns the value of the computational results. Since each iteration in a direct method improves the initial guess and generates a "better" trajectory, intermediate results are useful even if the process has not yet converged. Since the indirect methods only integrate extremals (i.e., solutions to the Euler-Lagrange equations), the results of each iteration do not give much useful information until the boundary conditions are almost met.

Probably the most severe difficulty with the indirect method is the numerical inaccuracies encountered when integrating the Euler-Lagrange equations. It may be shown (see for example Kipiniak [1961]) that in the case of linear equations with constant coefficients, the Euler-Lagrange equations have a set of characteristic roots which have the following property. If $\alpha + i\beta$ is a root, (i.e., there is a homogeneous solution of the form $e^{(\alpha+i\beta)t}$), then $-\alpha+i\beta$ is also a

root. When these equations are integrated either forward or backwards, for each damped stable mode, there is a corresponding unstable mode. For large time intervals these problems are extremely sensitive to small changes in the initial conditions and some form of double precision calculation may be required. Other investigators have noted difficulties with the indirect method for problems involving highly dissipative systems [Bryson, 1966]. However, for plants which are lightly damped, the two point boundary value problem is only slightly unstable. The difficulties with highly damped problems do not occur with the direct methods. This is because the system and adjoint equations are integrated separately in their "natural" directions. That is to say that the state equations are integrated forward and the adjoint equations are integrated backwards so that the linearized equations have the same set of eigenfunctions. The stability then depends only on the stability of the plant.

B. GRADIENT TYPE METHODS

In the remaining sections of this chapter, the discussion will be quite general so that it will be possible to connect the techniques used in functional minimization in an infinite dimensional space to those used in ordinary function minimization in E_n. The abstraction will actually simplify the statements to be made in most cases and the results will admit a wider interpretation. As a suitable reference for the mathematics used here, see Luenberger [1965], Lusternik and Sobolev [1961], or Kantorovich [1964].

In the following, let H be a Hilbert space with inner product $\langle x, y \rangle$ where $x \in H$, $y \in H$. The norm of an element $x \in H$ will be

denoted by

$$\| x \| = \sqrt{< x, x >} \tag{3.10}$$

Consider the following problem which is related to the most simple form of the computational control problem. Given a point $x_o \in H$ and a functional $f(x) : H \to E_1$, find a point $x \in H$ so that

$$\Delta f(x_o, x) = f(x) - f(x_o) < 0 \tag{3.11}$$

Assume that there is a linear functional of $\delta x = x - x_o$, depending on x_o, denoted by $\varphi(x_o, \delta x)$ which allows $\Delta f(x_o, x)$ to be written as

$$\Delta f(x_o, x) = \varphi(x_o, \delta x) + o(\|\delta x\|)$$

for arbitrary points x_o and δx. The function $o(\|\delta x\|)$ (read "little o of δx") depends on x_o and satisfies

$$\frac{o(\|\delta x\|)}{\|\delta x\|} \to 0 \quad \text{as} \quad \|\delta x\| \to 0 \tag{3.12}$$

Then, by definition, the functional $f(x)$ has a strong or Frechet differential $\varphi(x_o, \delta x)$ at the point x_o.

If the Frechet differential exists, it is equivalent to the Gateau or weak differential $Df(x_o, \delta x)$ which is defined by the relation

$$Df(x_o, \delta x) = \lim_{\epsilon \to 0} \frac{f(x_o + \epsilon \delta x) - f(x_o)}{\epsilon} \tag{3.13}$$

The Frechet differential is usually called the (first) variation of the functional $f(x)$ in books on the Calculus of Variations. The form (3.13) is not always equivalent to the Frechet differential, but as

previously noted they are equal when the Frechet differential exists and is more useful for computation in most cases. In the following, the Frechet differential will be referred to as the first variation or variation of $f(x)$ and will be denoted by $\delta f(x_o, \delta x)$ or simply δf.

Since δf is a linear functional on a Hilbert space, it may be uniquely represented by the inner product of an element $y \in H$ and δx. Therefore (3.11) becomes

$$\Delta f(x_o, x) = < y, \delta x > + o(\|\delta x\|) \tag{3.14}$$

The function y, which will frequently be written as f_x, is known as the gradient of f at x_o.

The fundamental basis for nonlinear minimization by gradient techniques is given in the following proposition.

Proposition 3.1

There exists a constant $c_o > 0$ such that if $<y, \delta x>$ is minimized over all $\delta x \in H$ with $\|\delta x\| = c \leq c_o$ then $\Delta f < 0$. Furthermore the minimum occurs for $\delta x = - cy/\|y\|$.

Proof: Any $\delta x \in H$ may be written as $\delta x = \alpha y + z$ with $<y, z> = 0$. Then $< y, \delta x > = \alpha\|y\|^2$ and $\alpha^2\|y\|^2 + \|z\|^2 = c^2$. If $<y, \delta x>$ is minimized, α is as small as possible which implies $z = 0$. Then $\alpha^2 = c^2/\|y\|^2$ and $\delta x = - \frac{c}{\|y\|} y$. For any $c > 0$ we have $\Delta f = (\delta x, - \frac{\|y\|}{c} \delta x) + o(\|\delta x\|) = - \|y\| \|\delta x\| + o(\|\delta x\|)$. By the definition of $o(\|\delta x\|)$ there is a constant $c > 0$ for which $|o(\|\delta x\|)| < \|\delta x\| \|y\|$ if $\|\delta x\| \leq c$. Therefore $\Delta f < 0$.

The iterative technique usually referred to as steepest descent for unconstrained minimization of the functional f may now be stated as

$$x^{(n+1)} = x^{(n)} - ky(x^{(n)}) \tag{3.15}$$

where $k < 0$ is the scalar step size. As before, the function $y(x^{(n)})$ is the gradient of the functional f at the point $x^{(n)}$. By Proposition 3.1, the method has step by step convergence. That is, for the proper choice of the scalar constant k, the inequality

$$f(x^{(n+1)}) - f(x^{(n)}) = \Delta f(x^{(n+1)}, x^{(n)}) < 0 \tag{3.16}$$

is satisfied.

This method is a gradient method since the change in x is along the gradient. There are several schemes for computing the constant k in this equation. Perhaps the simplest heuristic technique is the halfing and doubling method. To start the method, an initial step is made. If (3.16) is satisfied, the cost functional is decreased and the step is successful. In this case, k is doubled for the next iteration. If (3.16) is not satisfied, k is halved and another step is made from the original point. Of course, there are many variations of this technique for experimentally determining k.

A different method is obtained by assuming the functional f is quadratic in x. That is, f may be written as

$$f(x) = \langle y, x \rangle + 1/2 \langle Qx, x \rangle \tag{3.17}$$

where Q is a self-adjoint linear operator from H to H. With this model for $f(x)$, it is possible to pick the best k to minimize Δf, (i.e., maximize $|\Delta f|$).

By the method of steepest descent

$$x^{(n+1)} = x^{(n)} - kp^{(n)} \tag{3.18}$$

where $p^{(n)}$ is the gradient at the point $x^{(n)}$ which is given by $(y + Qx^{(n)})$. The constant k is chosen to minimize Δf and is given by

$$k = \hat{k} = \frac{< p, p >}{< Qp, p >} \tag{3.19}$$

In practice an easier approach for some problems is to determine k by measuring Δf for two values in addition to the point $k = 0$ from the previous iteration. Since if f is quadratic in x, Δf is quadratic in k, and these three points may be used to fit a parabola in k and hence determine $\hat{k}$.

Most of the problems of interest have constraints and therefore require some modification of the unconstrained gradient technique. Before formulating the nonlinear theorem for the problem with constraints corresponding to Proposition 3.1, we shall consider the necessary conditions for the functional $f(x)$ to be an extremum with the set of constraints $g_i(x) = 0$ for $i = 1,\ldots,q$.

Suppose f and g_i are continuously differentiable in a neighborhood of the point $x = x_o$. Further assume that the constraints are linearly independent. That is, the gradients of the functionals g_i are linearly independent functions not all vanishing. An equivalent statement is that the equation $< g_{i,x}(x_o), h > = \alpha_i$, $i = 1,\ldots,q$ has a solution $h \in H$ for arbitrary α_i or that the $q \times q$ Gram matrix A given by

$$[a_{ij}] = [< g_{i,x}(x_o), g_{j,x}(x_o) >]$$

be nonsingular.*

In the unconstrained problem, the necessary condition for an extremum was that the gradient of f vanish. For the unconstrained problem, the extremum is achieved if it is not possible to increase the cost while moving along the constraint. This is equivalent to requiring the component of the gradient of the cost functional f in the tangent manifold of the constraint $g(x) = 0$ to be zero at the point $x = x_o$. The tangent manifold is the set of all elements $h \in H$ which satisfy $< g_{i,x}(x_o), h > = 0 \quad i = 1,\ldots,q$. The tangent manifold is a subspace of H and will be designated by T.

The gradient of f may be written uniquely as $f_x = u + v$ with $u \in T$ and $v \in S$, the orthogonal complement of T. In this representation, u is the projection of f_x onto the tangent manifold T. Taking the inner product of f_x with $g_{j,x}$ gives

$$< f_x, g_{j,x} > = < u, g_{j,x} > + < v, g_{j,x} > = < v, g_{j,x} >$$

$$j = 1,\ldots,q \tag{3.20}$$

since $u \in T$. By assumption the $g_{j,x}$ are linearly independent and thereby form a basis for S. v may be written as

$$v = \sum_{i=1}^{q} \alpha_i g_{i,x} \tag{3.21}$$

*This assumption for the control problem is related to the idea of controllability for the linearized problem in the sense of Kalman [Kalman, Ho, Narendra, 1963]. It is also related to normality in the classical calculus of variations as noted by Breakwell and Ho [1965]. The matrix $[a_{ij}]$ may also be recognized as the matrix $I_{\psi\psi}$ in Bryson and Denham [1962].

Using (3.20) and (3.21) we may solve for v as

$$v = \sum_{j=1}^{q} \sum_{i=1}^{q} [< g_{i,x}, g_{j,x} >]^{-1} < f_x, g_{i,x} > g_{j,x} \tag{3.22}$$

The necessary condition follows immediately

$$u = f_x - \sum_{j=1}^{q} \sum_{i=1}^{q} [< g_{i,x}, g_{j,x} >]^{-1} < f_x, g_{i,x} > g_{j,x} = 0 \tag{3.23}$$

If we note that the term

$$\sum_{i=1}^{q} [< g_{i,x}, g_{j,x} >]^{-1} < f_x, g_{i,x} > \qquad i = 1,2,\ldots,q$$

represents a vector λ_j, we can write (3.23) in the familiar form

$$f_x - \sum_{j=1}^{q} \lambda_j g_{j,x} = 0$$

or

$$f_x - \lambda' g_x = 0 \tag{3.24}$$

which is the well-known Lagrange multiplier rule where $g = 0$ is taken as a vector constraint.

A useful physical interpretation for the Lagrange multipliers may be obtained by considering a problem with slightly perturbed constraints. The modified problem consists of finding an extremum for $f(x)$ with the constraints

$$g_i(x) - \epsilon_i = 0, \quad i = 1,2,\ldots,q. \tag{3.25}$$

If x_o solves the problem with $\epsilon_i = 0$, then there is a solution to (3.25) for any ϵ_i suitably small by the linear independence of the constraints. Therefore, there is a solution, $x_o + h$, to the constrained minimization of $f(x)$ with the constraint (3.24). The corresponding change in the minimum value of $f(x)$ is

$$f(x_o + h) - f(x_o) = \langle f_x(x_o), h \rangle + o(\|h\|). \tag{3.26}$$

By the differentiability of the constraint,

$$g_i(x_o + h) - g_i(x_o) = \langle g_{i,x}(x_o), h \rangle + o(\|h\|) \tag{3.27}$$

Application of the multiplier rule to the original problem shows that there is a set of multipliers λ_i, not all zero which satisfy

$$f_x(x_o) - \sum_{i=1}^{q} \lambda_i g_{i,x}(x_o) = 0. \tag{3.28}$$

By multiplying (3.27) by λ_i, summing over i, and subtracting the result from (3.26), one obtains

$$f(x_o + h) - f(x_o) = \sum_{i=1}^{q} \lambda_i [g_i(x_o + h) - g_i(x_o)]$$

$$+ \langle [f_x(x_o) - \sum_{i=1}^{q} \lambda_i g_{i,x}(x_o)], h \rangle + o(\|h\|) \tag{3.29}$$

From the original assumptions that $g_i(x_o) = 0$ and $g_i(x_o + h) - \epsilon_i = 0$ together with (3.28), the last equation may be written as

$$f(x_o + h) - f(x_o) = \sum_{i=1}^{q} \lambda_i \epsilon_i + o(\|h\|) \tag{3.30}$$

Equation (3.30) is the basis for the sensitivity coefficient interpretation of the Lagrange multipliers. In other words, this result says that the constrained extreme value of $f(x)$ changes to first order by an amount $\lambda_j \epsilon_j$ when the j^{th} constraint is changed by a small amount ϵ_j.

In the following chapter, constraints which are differential equations will be considered. In this case, the constraint $g(x)$ may be of the form

$$g(x) = \ddot{x} + (1 - x^2)\,\dot{x} + x = 0(t)$$

so that the range of g is no longer simply a set of numbers, but it may be an entire time function. In the book by Liusternik and Sobolev [1961], the multiplier rule is extended to handle more general problems of this nature. This theorem will prove useful in future developments so that it will be stated here. For the proof, the reader is referred Liusternik and Sobolev [1961].

Preliminary to the theorem, a few additional definitions are necessary. Let the constraint function $g(x)$ be defined on a Banach space B with range in a Banach space C, $g(x) \in C$, $x \in B$. $f(x)$ is a functional defined on B. Again assume that the constraints are linearly independent or that the range of the operator defined by the

gradient of g, g_x, is the whole space C. Both g and f are assumed continuously differentiable in a neighborhood of x_o. In the theorem, let C^* denote the dual (or conjugate) space of C.

Proposition 3.2

If $f(x)$ has an extremum with $g(x) = 0$ at the point $x = x_o$, then there is a linear functional L defined on C, $L \in C^*$, such that the functional

$$F(x) = f(x) - L[g(x)]$$

has a local minimum at $x = x_o$, i.e., the Frechet differential of $F(x)$, $\varphi(x, h)$ satisfies

$$\varphi(x_o, h) = 0 \quad \text{for all} \quad h \in B.$$

The extension of this theorem to problems with inequality constraints $g(x) \geq 0$ has been studied as a generalization of the Kuhn-Tucker theorem by Hurwicz [1958]. Lack [1965] discusses the application of this theorem in deriving necessary conditions for the control problem. The Pontryagin Maximum Principle stated in Chapter II is actually another form of a Lagrange multiplier rule with inequality constraints (bounded control).

There are several versions of the gradient technique for computing constrained extrema for the control problem. In the absence of constraints, the solution is obtained by choosing δu to minimize the linear part of the change, $\Delta\varphi$, in the cost, plus an added quadratic functional chosen to restrict the step size. A constrained problem may be treated by requiring the change, δu, in the control to be chosen so as to satisfy specified changes, $\delta\psi$, in the constraints to first order in addition

to minimizing the linear part of $\Delta\varphi$ plus a quadratic term as in the unconstrained case. The following lemma gives one method for constructing the solution to the problem of minimizing a linear plus a quadratic functional with a linear equality constraint. The idea of the lemma is to first find the shortest (minimum norm) solution which satisfies the constraints and then to optimize in the tangent manifold so that the optimization process does not effect the constraints. The solution conveniently separates into two parts, the part necessary to meet the constraints, and the part which minimizes the cost.

Lemma 3.1 The solution to the problem of finding an element $x \in H$ which minimizes $< a, x > + 1/2\ W < x, x >$ with $< b, x > = \alpha$ where $a, b \in H$ and W and α are scalars, is given by

$$x = -\frac{1}{W} Pa + \hat{x}$$

where $\hat{x}$ is the minimum norm solution to $< b, x > = \alpha$ and P is a projection operator onto the nullspace of b, i.e.,

1. $< b, Px > = 0$ for every $x \in H$
2. $Pd = d$ for every d which satisfies $< b, d > = 0$

Proof: By the multiplier rule, the optimum x is given in terms of a constant λ as $x = -1/W(a - \lambda b)$. Since $< b, x > = \alpha$, then $< b, a > - \lambda < b, b > = - W\alpha$ or $\lambda = (< b, a > + W\alpha)/< b, b >$, so that

$$x = -\frac{1}{W}\left(a - \frac{<b, a>}{<b, b>} b\right) - \frac{\alpha b}{< b, b >}$$

Since the minimum norm solution, $\hat{x}$, to $< b, x > = \alpha$ is $\alpha b/< b, b >$,

the proof is completed by showing $Pa = (a - < b, a > b/< b, b >)$. To show property 1, $< b, Px > = < b, (x - b< b, x >/< b, b >)> = < b, x > - < b, x > = 0$. Property 2 follows from $Pd - (d - < b, d > b/< b, b >) = d$ since $< b, d > = 0$.

The solution x constructed in the lemma may be used to solve the nonlinear constrained minimization problem by a gradient technique. The basis for this approach is given in the following theorem.

Proposition 3.3

There exists a set of positive numbers W, k, and ϵ_i, $i = 1,2,\ldots q$, such that if h is an element in H which minimizes $< f_x(x_o), h > + \frac{1}{2}W < h, h >$ with $< g_{i,x}(x_o), h > = - k\, g_i(x_o)$ then if $|g_i(x_o)| > \epsilon_i$ then $|g_i(x_o + h)| < |g_i(x_o)|$ otherwise $|g_i(x_o + h)| < \epsilon_i$ and $f(x_o + h) < f(x_o)$.

Proof: See Appendix A.

By comparison of the results of the last theorem with the goal as defined in the statement of the Computational Control Problem in Chapter II, it may be seen that the problem is solved by specifying a method for finding the constants W, k, and ϵ_i. In effect, k controls the amount of improvement desired in the constraint g, ϵ_i sets a tolerance limit on the accuracy in meeting the contraints and W controls the step size. In order to maximize the convergence rate, it is desirable to pick $1/W$, k, and ϵ_i as large as possible without violating the requirements of Proposition 3.3. The method used for finding suitable values for W, k, and ϵ_i is discussed in Chapter VI.

Other versions of the construction of the element h in Proposition 3.3 may be used. One technique suggests calling for certain improvements,

δg_i, in each of the g_i's and δf in f. The element h is then chosen so that it is the minimum norm solution to the set of equations

$$\begin{aligned} &< g_{i,x}(x_o),\ h > = \delta g_i \\ &< f_x(x_o),\ h > = \delta f \end{aligned} \tag{3.31}$$

where δg_i and δf are again selected suitably small so that the requirements of Proposition 3.3 are satisfied. By the multiplier rule, the optimal h must furnish an extreme value for

$$< h,\ h > + \sum_{i=1}^{q} \lambda_i < g_{i,x},\ h > + \lambda_o < f_x,\ h > \tag{3.32}$$

where λ_o is chosen so that $< f_x,\ h > = \delta f$. The construction used in Proposition 3.3 requires h to furnish an extreme value for

$$1/2\ W < h,\ h > + \sum_{i=1}^{q} \nu_i < g_{i,x},\ h > + < f_x,\ h >$$

or

$$< h,\ h > + \sum_{i=1}^{q} \frac{2\nu_i}{W} < g_{i,x},\ h > + \frac{2}{W} < f_x,\ h > \tag{3.33}$$

so that the methods are equivalent with the identifications

$$\frac{2\nu_i}{W} = \lambda_i, \quad i = 1,2,\ldots,n \tag{3.34}$$

and

$$\frac{2}{W} = \lambda_o \tag{3.35}$$

In the second approach, the specification of δf determines λ_o and hence W. However, the method in the theorem requires the direct specification of W.

C. SECOND-ORDER METHODS

According to the multiplier rule given in the last section, the constrained minimization of $f(x)$ with $g_i(x) = 0$ may be reduced to the problem of finding the unconstrained minimum of an auxilliary functional $F(x,\lambda)$ defined by

$$F(x,\lambda) = f(x) + \lambda' g(x)$$

where for convenience the sum $\Sigma\lambda_i g_i(x)$ is written as $\lambda' g(x)$ with the appropriate definition of λ and g as vectors. It may be easily shown that a form of constrained gradient technique is obtained by applying the unconstrained gradient method (first order) to $F(x,\lambda)$. In this application a second-order method is used to minimize $F(x,\lambda)$, thus resulting in a second-order method which includes constraints.

In the unconstrained gradient method, $f(x)$ was minimized in a step by step fashion by choosing the change in x, h, to minimize the first order part of $f(x + h)$ with the constraint that the step size be small enough so that the higher order terms were negligible. In the second-order method, the functional to be minimized with the constraint $\|h\| = c$ is

$$f(x + h) - f(x) = < f_x, h > + \frac{1}{2} < f_{xx} h, h > + o(\|h\|^2)$$

in which f_{xx} is a linear self adjoint operator from H to H. The minimizing h is

$$h = - [f_{xx} + \nu I]^{-1} f_x$$

where I is the identity operator and $\nu/2$ is the Lagrange multiplier associated with the constraint $\|h\| = c$. Ordinarily, ν would be determined in terms of c from the constraint. An equivalent procedure would be to pick ν, instead of c, arbitrarily, and to minimize the expression

$$< f_x, h > + \frac{1}{2} < f_{xx} h, h > + \frac{1}{2} \nu < h, h > .$$

This method is formalized in the following theorem.

Proposition 3.4

There exists a constant ν sufficiently large such that if

$$< f_x, h > + \frac{1}{2} < f_{xx} h, h > + \frac{1}{2} \nu < h, h > \qquad (3.36)$$

is minimized over all $h \in H$, then

$$f(x_o + h) - f(x_o) < 0 .$$

Furthermore, the minimum occurs for

$$h = - [f_{xx} + \nu I]^{-1} f_x$$

Proof: See Appendix A.

A constrained minimization technique may be constructed by applying the above method to the functional $F(x,\lambda)$ as defined in the first part of this section. Expanding $F(x,\lambda)$ to second order as a function of x and λ gives

$$F(x + h, \lambda + \delta\lambda) = F(x, \lambda) + < F_x, h > + \delta\lambda' g(x) + \frac{1}{2} < F_{xx} h, h >$$

$$+ \delta\lambda' < g_x, h > + o(\|h\|^2)$$

The condition for $F(x + h, \lambda + \delta\lambda) - F(x, \lambda)$ to be an extremum for h and $\delta\lambda$ to second order is for h to furnish an extremum for F considered as a function of h alone and for

$$g_i(x) = - < g_{i,x} h >$$

which is the condition for the constraints to be met to first order. This idea leads to the following theorem.

Proposition 3.5

There exists a constant ν sufficiently large and a set of constraint tolerances ϵ_i, $i = 1,2,\ldots,q$ such that if

$$< F_x, h > + \frac{1}{2} < F_{xx} h, h > + \frac{1}{2} \nu < h, h >$$

is minimized over all $h \in H$, with

$$F(x, \lambda) = f(x) + \lambda' g(x)$$

and

$$g_i(x_o) = - < g_{i,x}(x_o), h >$$

then if $|g_i(x_o)| > \epsilon_i$, $|g_i(x_o + h)| < |g_i(x_o)|$ or if $|g_i(x_o)| < \epsilon_i$, $|g_i(x_o + h)| \leq \epsilon_i$ and $f(x_o + h) < f(x_o)$.

Proof: See Appendix A.

The results given have been only concerned with step by step convergence and do not include a consideration of the rate of convergence.

The second order method may be intuitively expected to converge faster than the gradient since it uses more information about the local behavior of the nonlinear functionals f and g_i. Kantorovitch [1964] gives some results concerning bounds on the convergence rate in the unconstrained case in terms of bounds on the operator f_{xx}. The resulting bounds have little use in practical computing schemes due to the difficulty in estimating a tight bound on f_{xx} and because actual results may be considerably better than the theoretical bounds. For these reasons, only an experimental investigation of the relative convergence rates has been considered in this report.

IV. A DIRECT METHOD BASED ON SECOND VARIATIONS

In the last chapter, the iterative constrained extremum problem was set up and solved in a general manner from an abstract point of view. The chief result was to reduce the complex nonlinear problem to a sequence of less difficult problems. The purpose of the present chapter is to apply these general results to our specific control problem. This will lead to the formulation of an easier control problem which can be handled analytically. The discussion of the resulting auxiliary problem is the topic of the following chapter, Chapter V.

A. DERIVATION OF THE METHOD

In order to avoid too many of the details of the general case, we shall first consider a more specialized problem. More specifically, we shall assume that the functions $f(x, u)$, $\psi[x(t_f)]$, $\phi[x(t_f)]$ all have continuous first and second partial derivatives and that the state or control variable constraints have been taken care of by suitably smooth penalty functions. Further, we assume that the final time, t_f, is fixed.

In order to apply the Lagrange multiplier technique developed in the last chapter to the problem of minimizing $\phi[x(t_f)]$ with $\psi[x(t_f)] = 0$ and $\dot{x} = f(x, u)$, suitable linear functionals to append the constraints must be constructed. The usual end constraint function $\psi[x(t_f)]$ has its range in E_q and hence the dual space is also E_q. We may then write the appropriate linear functional of ψ as required by the theorem as an inner product of an element of the dual space, represented by the vector $\omega \in E_q$, and ψ. This may be written as (ω, ψ) where $(\ ,\ )$

denotes the inner product in E_q or in the more conventional vector notation $\omega'\psi$. The constraint $\dot{x} - f(x, u) = 0$ may be handled by noting that its range is a set of n-dimensional time functions defined on $[t_o, t_f]$ with the inner product

$$< x, y > = \int_{t_o}^{t_f} \sum_{i=1}^{n} x_i(\sigma)\, y_i(\sigma)\, d\sigma$$

$$= \int_{t_o}^{t_f} x'(\sigma)\, y(\sigma)\, d\sigma \ .$$

A general linear functional defined on this space may be written as

$$f(x) = \int_{t_o}^{t_f} \lambda'(\sigma)\, x(\sigma)\, d\sigma$$

where $\lambda(t)$ is another n-vector time function defined on $[t_o, t_f]$.*

The control problem (as stated in Chapter II, Section B) is equivalent to finding the minimum of a new functional defined in the Lagrange multiplier rule. This functional may be written with the aid of the appropriate linear functionals defined above as

$$F(x, u, \lambda, \nu) = \phi[x(t_f)] - \nu'\psi[x(t_f)] + \int_{t_o}^{t_f} \lambda'(f - \dot{x})d\sigma \qquad (4.1)$$

*The definition of the function spaces has been made intentionally vague at this point to avoid unnecessary difficulties regarding the closure of the space. We shall tacitly assume that there is an appropriate Hilbert space which describes the functions of interest.

It is convenient to define the Hamiltonian function again as in (3.7) as $H = \lambda'(t)\, f(x, u)$. The functional F may then be written as

$$F(x, u, \lambda, \nu) = [x(t_f)] - \nu'\psi[x(t_f)] + \int_{t_o}^{t_f} (H - \lambda'\dot{x})\, d\sigma \,. \qquad (4.2)$$

In order to apply the results of the last chapter, it is necessary to compute several first and second Frechet differentials of the payoff φ and the constraints. However, we will not use this exact approach but use an equivalent one. By the multiplier rule, we seek the minimum of the new unconstrained functional F. By expanding F in a Taylor series in all of its variables to second order and finding the extremum of the result, we not only compute the required differentials, but the results in the last chapter are rederived for this specific problem.

For convenience, the expansion will be done in two parts. First, the function defined by

$$\varphi[x(t_f), \nu] = \phi[x(t_f)] - \nu'\psi[x(t_f)] \qquad (4.3)$$

will be expanded. To second order, φ is:

$$\varphi[x(t_f) + \delta x_f, \nu + \delta\nu] = \varphi + \varphi_x \delta x_f - \delta\nu'\psi + 1/2\, \delta x_f' \varphi_{xx} \delta x_f$$

$$- \delta\nu'\psi_x \delta x_f + o(\|\delta x_f\|^2) + o(\|\delta\nu\|^2) \qquad (4.4)$$

$$+ \nu'\delta a + 1/2\ \delta x_f' \varphi_{xx} \delta x_f - \delta\nu'(\psi_x \delta x_f - \delta a) + o(\|\delta x_f\|^2) +$$

$$+ o(\|\delta\nu\|^2) + o(\|\delta a\|^2) \qquad (4.4)$$

where all of the functions on the right are evaluated at the nominal point $x(t_f)$, ν and δx_f, $\delta\nu$ denote changes from the nominal point.

The technique for expanding the integral remaining in (4.2) is well known in classical calculus of variations and involves integration by parts. The result is

$$\int_{t_o}^{t_f} [H(x + \delta x,\ u + \delta u,\ \lambda + \delta\lambda) - (\lambda + \delta\lambda)'(\dot{x} + \delta\dot{x})]\ d\sigma$$

$$= \int_{t_o}^{t_f} [(H - \lambda'\dot{x}) + (H_x + \dot{\lambda}')\ \delta x + (H_\lambda - \dot{x}')\delta\lambda]\ d\sigma$$

$$+ \int_{t_o}^{t_f} [(H_u)\ \delta u + \delta\lambda'(H_{\lambda x}\delta x + H_{\lambda u}\delta u - \delta\dot{x})]d\sigma$$

$$+ 1/2 \int_{t_o}^{t_f} (\delta x'\delta u') \begin{pmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu} \end{pmatrix} \begin{pmatrix} \delta x \\ \delta u \end{pmatrix} d\sigma$$

$$- \lambda'(t_f)\delta x_f + \lambda'(t_o)\delta x(t_o) + o(\|\delta x\|^2) + o(\|\delta u\|^2) + o(\|\delta\lambda\|^2)\ . \qquad (4.5)$$

The nominal trajectory $x(t)$ is chosen to satisfy $\dot{x} = f(x, u)$ which requires

$$\dot{x} = H'_{\lambda} \ . \tag{4.6}$$

The adjoint variables λ may be chosen to satisfy the usual adjoint equation

$$\dot{\lambda} = - H'_{x} \ . \tag{4.7}$$

The change in x, δx satisfies the linear perturbation equation

$$\delta\dot{x} = f_x \delta x + f_u \delta u = H_{\lambda x} \delta x + H_{\lambda u} \delta u \tag{4.8}$$

and the boundary condition taken when $x(t_o)$ is specified

$$\delta x(t_o) = 0 \ . \tag{4.9}$$

F is made stationary with respect to δx_f by requiring

$$\lambda(t_f) = \varphi'_x \ . \tag{4.10}$$

Given ν and $x(t_f)$, (4.10) with (4.7) may be used to define λ. Equation (4.10) may also be used to compute ν if $x(t_f)$ and $\lambda(t_f)$ are known provided $x(t_f)$ and $\lambda(t_f)$ are such that a solution for ν exists.

Normally the numerical procedure calls for a full correction to ψ so we take $\psi_x \delta x_f = - \psi$. For a partial collection $\delta\psi$, the term $-\delta\nu'(\psi + \psi_x \delta x_f)$ in (4.4) becomes $-\delta\nu'(\psi + \delta\psi)$. Taking the definitions in (4.6) through (4.10) into account gives

$$\Delta F = F(x + \delta x, u + \delta u, \lambda + \delta\lambda, \nu + \delta\nu) - F(x, u, \lambda, \nu)$$

$$= \nu'\delta\psi + 1/2\ \delta x_f'\varphi_{xx}\delta x_f + \int_{t_o}^{t_f} H_u \delta u\ d\sigma$$

$$+ 1/2 \int_{t_o}^{t_f} (\delta x' \delta u') \begin{pmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu} \end{pmatrix} \begin{pmatrix} \delta x \\ \delta u \end{pmatrix} d\sigma$$

$$+ o(\|\delta u\|^2) \ . \qquad (4.11)$$

The error term is written in terms of δu alone since the other quantities $\delta\lambda$, $\delta\nu$, $\delta\psi$, and δx are related to δu by a bounded linear operator.

As shown in the last chapter, the Computational Control Problem is solved by finding δu to extremize $\hat{J}$ given by

$$\hat{J} = \nu'\delta\psi + 1/2\ \delta x_f'\varphi_{xx}\delta x_f + \int_{t_o}^{t_f} H_u \delta u\ d\sigma$$

$$+ 1/2 \int_{t_o}^{t_f} (\delta x' \delta u') \begin{pmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu} \end{pmatrix} \begin{pmatrix} \delta x \\ \delta u \end{pmatrix} d\sigma$$

$$+ 1/2 \int_{t_o}^{t_f} \delta u' W\ \delta u\ d\sigma \qquad (4.12)$$

for W chosen so that $\|W\|$ is suitably large. The weighting W is usually taken as a constant diagonal matrix cI with $c > 0$ $(c < 0)$ if $\hat{J}$ is to be minimized (maximized).

The Computational Control Problem is then solved by finding δu which extremizes (4.12) while satisfying certain constraints. The perturbed control δu and the perturbed state vector δx are related by the perturbation equation (4.8). The boundary conditions to be satisfied are

$$\delta x(t_o) = 0, \quad \psi_x[x(t_f)]\, \delta x(t_f) = \delta\psi \tag{4.13}$$

where $\delta\psi$ is usually specified as $-\psi$ in an effort to completely satisfy the terminal constraints. This subproblem has been studied in detail in the theory of optimal control and is generally called the linear quadratic loss problem. Before turning to the analytic solution of this problem, some of the assumptions made in the first part of this section will be removed.

B. EXTENSION TO BANG-BANG OPTIMAL CONTROL

There is a class of optimal control problems with bounded control, known as "Bang-Bang" problems, in which the Hamiltonian function assumes its extreme values for the control on the boundary. In these problems, the control may often be described in a simplified manner. For example, the control bound might be $|u(t)| \leq 1$. In this case, assuming the control is always $+1$ or -1, the control function may be described by its initial value and the sequence of switching times. By this technique, knowledge of the form of the optimal control from the Minimum Principle may be used to redefine the control variable so that the new "control," namely the initial control and the switching times, is possibly finite dimensional. Another valuable advantage of this

scheme is the fact that the new control may be considered to be unconstrained, thereby simplifying the computational effort.

In this section it will be assumed that all of the original "controls" consisted either of variable points of discontinuity of f, (staging times), or discontinuous controls which have been appropriately removed by a change of variables. As before, the investigation begins by expanding (4.2) in two parts. Equation (4.4) is still valid so that it is only necessary to compute the effect of a change in the points of discontinuity of f at $t = t_i$, $i = 1,2,\ldots,k$. It will become clear in the following derivation that it is sufficient to study only a single discontinuity at t_i without loss of generality. The expansion equivalent to (4.5) is obtained by a consideration of the difference of the integral of $H - \lambda'\dot{x}$ on the perturbed path as compared to the original path which may be written as

$$\int_{t_o}^{t_1+\delta t_i} \lambda'[f^-(x + \delta x) - \dot{x} - \delta\dot{x}]\, d\sigma + \int_{t_1+\delta t_i}^{t_f} \lambda'[f^+(x + \delta x) - \dot{x} - \delta\dot{x}]\, d\sigma$$

$$- \int_{t_o}^{t_1} \lambda'[f^-(x) - \dot{x}]d\sigma - \int_{t_1}^{t_f} \lambda'[f^+(x) - \dot{x}]\, d\sigma \qquad (4.14)$$

where f^- and f^+ denote the respective functional forms for f to the left and to the right of the discontinuity, and the shift δt_i has been taken positive. By adding and subtracting the integral

$$\int_{t_i}^{t_i+\delta t_i} \lambda' f^+\, d\sigma$$

the sum of the integrals in (4.14) becomes

$$\int_{t_o}^{t_i} \lambda'[f^-(x + \delta x) - f^-(x) - \delta\dot{x}]\, d\sigma + \int_{t_i}^{t_f} \lambda'[f^+(x + \delta x)$$

$$- f^+(x) - \delta\dot{x}]\, d\sigma - \int_{t_i}^{t_i+\delta t_i} \lambda'\delta\dot{x}\, d\sigma + \int_{t_i}^{t_i+\delta t_i} \lambda'[f^-(x + \delta x)$$

$$- f^+(x + \delta x)]\, d\sigma \ . \tag{4.15}$$

If δt_i is negative, then by adding and subtracting the integral

$$\int_{t_i+\delta t_i}^{t_i} \lambda' f^- d\sigma$$

equation (4.15) is again obtained.

Prior to evaluating (4.15), it will be convenient to define a type of forward difference operator $\mathfrak{D}_i$ by

$$\mathfrak{D}_i[g(t)] = g(t_i^+) - g(t_i^-) \tag{4.16}$$

In addition to the obvious linearity property of $\mathfrak{D}_i$, the following product rule will prove useful

$$\mathfrak{D}_i[f(t)g(t)] = f(t_i^+)\mathfrak{D}_i[g(t)] + \mathfrak{D}_i[f(t)]g(t_i^-)$$

$$= f(t_i^-)\mathfrak{D}_i[g(t)] + \mathfrak{D}_i[f(t)]g(t_i^+) \tag{4.17}$$

For convenience, the subscript i on the operator $\mathfrak{D}_i$ will be omitted where only one discontinuity is under consideration.

This difference operator may be used to write the perturbation equation corresponding to (4.8) in a simple form. In this case, δx satisfies the differential equation

$$\delta\dot{x} = f_x \delta x, \quad \delta x(0) = 0, \quad \psi_x \delta x(t_f) = \delta\psi \tag{4.18}$$

except at the points of discontinuity $t = t_i$. The discontinuity in δx is obtained by extrapolation of the effect of the change in t_i to the time of the old discontinuity in f at t_i. The idea is illustrated in Fig. 4.1 where the effect of a negative shift δt_i in the switching time on the j^{th} state variable y is shown. The actual difference between the trajectories is

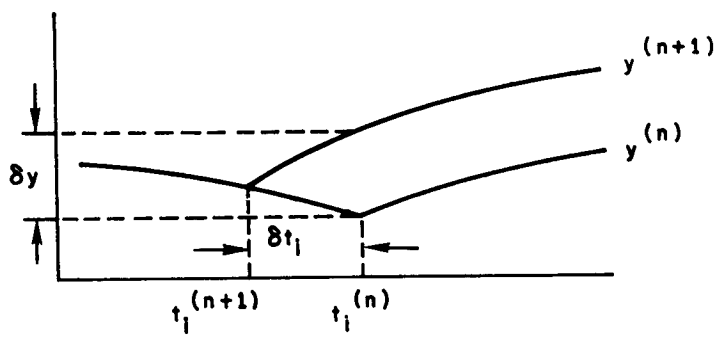

FIG. 4.1 EFFECT OF A NEGATIVE SHIFT IN THE SWITCHING TIME

continuous. The effect may be taken into account by considering δx to be discontinuous by

$$\Delta(\delta x) = -\Delta(f)\delta t_i - \Delta(f_x \delta x)\delta t_i - \frac{1}{2}\Delta(\dot{f})\delta t_i^2 \tag{4.19}$$

Returning to the evaluation of the integrals in (4.15), the third integral may be evaluated either by a careful limiting process or

by interpreting $\delta\dot{x}$ as a symbolic derivative of δx.[†] The result is

$$-\int_{t_i^-}^{t_i^+} \lambda'\delta\dot{x}\, d\sigma = -\frac{1}{2}[\lambda(t_i^+) + \lambda(t_i^-)]'\mathcal{S}(\delta x) \tag{4.20}$$

The fourth integral in (4.15) may be evaluated by a Taylor series expansion to obtain

$$\int_{t_i}^{t_i+\delta t_i} \lambda'[f^-(x+\delta x) - f^+(x+\delta x)]\, d\sigma$$

$$= -\lambda'(t_i^*)[\mathcal{S}(f) + \mathcal{S}(f_x\delta x)]\,\delta t_i - \frac{1}{2}\frac{d}{dt}[\lambda'\mathcal{S}(f)]\Big|_{t=t_i^*}\delta t_i^2 \tag{4.21}$$

where $t_i^* = t_i^+$ if $\delta t_i > 0$ and t_i^- if $\delta t_i < 0$. The last term may be simplified by carrying out the differentiation to show that

$$\frac{d}{dt}[\lambda'\mathcal{S}(f)]\Big|_{t=t_i^*} = \lambda'(t_i^*)\mathcal{S}(\dot{f}) \tag{4.22}$$

So that (4.21) becomes

$$\int_{t_i}^{t_i+\delta t_i} \lambda'[f^-(x+\delta x) - f^+(x+\delta x)]\, d\sigma = -\lambda'(t_i^*)\mathcal{S}(\delta x) \tag{4.23}$$

The first two integrals in (4.15) may be treated as in the last section by integrating the term $-\lambda'\delta\dot{x}$ by parts. After the integration

[†]For a description of symbolic differentiation of discontinuous functions see, for example, Friedman [1956], chapter 3.

and the subsequent elimination of terms [by requiring λ to satisfy the adjoint equation (4.7) on the intervals $[t_o, t_i^-)$, $(t_i^+, t_f]$, and to satisfy the boundary conditions (4.10)], the remaining part of the first two integrals is

$$\frac{1}{2}\int_{t_o}^{t_f} \delta x' H_{xx} \delta x d\sigma + \mathcal{S}(\lambda')\delta x(t_i^-) + \lambda'(t_i^+)\mathcal{S}(\delta x) \ . \tag{4.24}$$

The expression for the change, $\Delta\varphi$, in the payoff φ, may be obtained to second order by combining (4.20), (4.23), and (4.24) with the remaining terms of (4.4). The result is

$$\begin{aligned}\Delta\varphi = -\delta\nu'(\psi+\delta\psi) + \frac{1}{2}\delta x_f' \varphi_{xx} \delta x_f + \frac{1}{2}\int_{t_o}^{t_f} \delta x' H_{xx} \delta x d\sigma \\ + \mathcal{S}(\lambda')\delta x(t_i^-) + \lambda'(t_i^+)\mathcal{S}(\delta x) - \lambda'(t_i^*)\mathcal{S}(\delta x) \\ - \frac{1}{2}[\lambda'(t_i^+) + \lambda'(t_i^-)]\mathcal{S}(\delta x) \ .\end{aligned} \tag{4.25}$$

If $\Delta\varphi$ is to be stationary with respect to arbitrary changes in $\delta x(t_i^-)$, then the coefficient $\mathcal{S}(\lambda')$ of $\delta x(t_i^-)$ must vanish. The adjoint variable $\lambda(t)$ is therefore chosen continuous and (4.25) becomes

$$\Delta\varphi = \nu'\delta\psi + \frac{1}{2}\delta x_f' \varphi_{xx} \delta x_f + \frac{1}{2}\int_{t_o}^{t_f} \delta x' H_{xx} \delta x d\sigma - \lambda'\mathcal{S}(\delta x) \ . \tag{4.26}$$

The accessory problem may now be formulated. The control, in this case δt_i, $i = 1,2,\ldots k$, is to be found which minimizes the cost

$$\hat{J} = -\delta\nu'(\psi+\delta\psi)+ \frac{1}{2}\,\delta x_f'\,\varphi_{xx}\delta x_f + \frac{1}{2}\int_{t_o}^{t_f}\delta x' H_{xx}\delta x\,d\sigma + \sum_{i=1}^{k}\{\mathcal{D}_i(H)\delta t_i$$

$$+ \mathcal{D}_i(H_x\delta x)\delta t_i + \frac{1}{2}\,\lambda'\mathcal{D}_i(\dot{f})\delta t_i^2 + \frac{1}{2}\,W_i\delta t_i^2\} \qquad (4.27)$$

with the constraint that δx satisfies the differential equation

$$\delta\dot{x} = f_x\delta x,\ \delta x(0) = 0,\quad \psi_x\delta x(t_f) = \delta\psi \qquad (4.28)$$

except at the points $t = t_i$ where δx is discontinuous. The amount of discontinuity is

$$\mathcal{D}(\delta x) = -\mathcal{D}(f)\delta t_i - \mathcal{D}(f_x\delta x)\delta t_i - \frac{1}{2}\,\mathcal{D}(\dot{f})\delta t_i^2 \quad . \qquad (4.29)$$

This completes the corresponding subproblem specification for problems which have a bang-bang optimal control law. Although the method of steepest descent for such problems will not be discussed further here, it is clear that the technique presented for evaluating the functional gradient may be used to apply the method of Bryson and Denham [1962] to such problems. This idea has been successfully used in a different form by Vachino [1966] and Hales [1966] in developing a steepest descent algorithm applicable when some of the controls are of the on-off type.

C. PROBLEMS WITH CONTROL PARAMETERS

Often it is desired to optimize a system with respect to a set of plant parameters. For example, in a rocket steering problem, a control parameter might be the time at which staging occurs. In this case, the

right-hand side of the system equation $\dot{x} = f(x, u, t_s)$ is discontinuous with respect to the staging time t_s. The method developed in the last section may then be applied. Suppose that the mass of the vehicle and the initial orientation are also to be determined in an optimal manner. The right-hand side of the differential equation may be continuously differentiable with respect to some of the parameters, as the mass. This is actually a special case of the control function $u(t)$ as discussed in Section A. The specialization of the results of Section A to parameters of this type is the first topic of this section. The second topic is the optimization of initial conditions such as the unknown initial orientation of the rocket.

The system equation will be written as $\dot{x} = f[x(t), \alpha]$ where α is a $p \times 1$ vector of parameters to be determined. The function $f[x(t), \alpha]$ is taken as twice continuously differentiable on $t \in [t_o, t_f]$ for any $x(t)$.

Assuming α does not depend on time, the results of Section A are modified by replacing $\delta u(t)$ with $\delta\alpha$ and taking $\delta\alpha$ outside of the integrals. Equation (4.12) becomes

$$\Delta F = -\delta\nu'(\psi+\delta\psi) + \frac{1}{2}\delta x_f'\varphi_{xx}\delta x_f + \left[\int_{t_o}^{t_f} H_\alpha + \delta x' H_{x\alpha}\, d\sigma\right]\delta\alpha$$

$$+ \frac{1}{2}\int_{t_o}^{t_f} \delta x' H_{xx}\delta x\, d\sigma + \frac{1}{2}\delta\alpha'\left[\int_{t_o}^{t_f} (H_{\alpha\alpha} + W)\, d\sigma\right]\delta\alpha\ . \qquad (4.30)$$

The only additional change needed is in the perturbation equation which becomes

$$\delta\dot{x} = f_x \delta x + f_\alpha \delta\alpha \tag{4.31}$$

with boundary conditions given by (4.13). This completes the modifications necessary to include control parameters of this type in the theory.

In Section A, the initial conditions were assumed given and therefore $\delta x(t_o) = 0$ as in (4.9). If some of the components of $x(t_o)$ are not specified, the corresponding components of $\delta x(t_o) = \delta x_o$ are not zero. In this case ΔF in equation (4.11) will have an added term $-\lambda_o' \delta x_o$.

Since ΔF is linear in δx_o, a gradient technique for adjusting δx_o is suggested. This is done by the usual method of adding a term quadratic in δx_o of the form $1/2\ \delta x_o' V \delta x_o$ to ΔF. The positive definite matrix V is then adjusted for convergence.

D. PROBLEMS WITH FREE FINAL TIME

If the final time is not specified, the derivation of Section A no longer holds. The final state will now vary due to a change in the value of the state at time $t = t_f^{(n)}$ <u>and</u> due to a change in the final time. That is, the total change in the terminal state dx_f is given by

$$\begin{aligned} dx_f = x^{(n+1)}(t_f^{(n+1)}) - x^{(n)}(t_f^{(n)}) &= x^{(n+1)}(t_f^{(n)}) - x^{(n)}(t_f^{(n)}) \\ &+ [\,\dot{x}^{(n+1)}(t_f^{(n)}) - \dot{x}^{(n)}(t_f^{(n)}) + \dot{x}^{(n)}(t_f^{(n)})](t_f^{(n+1)} - t_f^{(n)}) \\ &+ 1/2\ \ddot{x}^{(n)}(t_f^{(n)})(t_f^{(n+1)} - t_f^{(n)})^2 + o(\|t_f^{(n+1)} - t_f^{(n)}\|^2) \\ &+ o(\|\, x_f^{(n+1)} - x_f^{(n)}\|^2) \quad . \end{aligned} \tag{4.32}$$

The usual more compact form is

$$dx_f = \delta x_f + \dot{x}_f \delta t_f + \delta \dot{x}_f \delta t_f + 1/2\ \ddot{x}_f (\delta t_f)^2 + \ldots \qquad (4.33)$$

which is more convenient but less illuminating. The substitution of dx_f from this expression for δx_f in (4.4) gives the correct expansion of φ when the final time is not specified.

$$\Delta\varphi = \delta\nu'(\psi - a) + \nu'\delta\psi - \delta\nu'(\psi_x dx_f - \delta\psi)$$

$$+ \varphi_x dx_f + 1/2\ dx_f' \varphi_{xx} dx_f + \ldots \ . \qquad (4.34)$$

The other effect of a change in the final time is to generate some additional terms in the expansion of the integral in (4.5). Evaluating the effect of a change in the upper limit of integration leads to

$$\int_{t_o}^{t_f+\delta t_f} [H(x + \delta x,\ u + \delta u,\ \lambda + \delta\lambda) - (\lambda + \delta\lambda)'(\dot{x} + \delta\dot{x})]\ d\sigma$$

$$= \int_{t_o}^{t_f} [\quad]\ d\sigma + [\quad]\Big|_{t=t_f} \delta t_f + 1/2\ \frac{d}{dt}[\quad]\Big|_{t=t_f} \delta t_f^2 + \ldots \qquad (4.35)$$

where $[\quad]$ denotes the integrand of the first integral. The integral $\int_{t_o}^{t_f} [\quad]\ d\sigma$ may now be expanded as before. It will be convenient to use the equations (4.6-4.10) to simplify the result together with the relation

$$\delta\psi = \psi_x dx_f \ . \qquad (4.36)$$

After some manipulation, (4.34) and (4.35) may be combined to give

$$\Delta F = \{(4.11) \text{ with } \delta x_f \text{ replaced by } dx_f\} + H\delta t_f$$

$$+ H_x \delta x_f \delta t_f + H_u \delta u_f \delta t_f - \lambda_f' \dot{\delta x}_f \delta t_f + 1/2\, H_u \dot{u} \delta t_f^2 - 1/2\, \dot{\lambda}_f' \dot{x}_f \delta t_f^2$$

$$- 1/2\, \lambda_f' \ddot{x}_f \delta t_f^2 + \varphi_x \dot{\delta x}_f \delta t_f + 1/2\, \varphi_x \ddot{x}_f \delta t_f^2$$

$$= \{(4.11) \text{ with } \delta x_f \text{ replaced by } dx_f\} + H\delta t_f$$

$$+ H_u \delta u_f \delta t_f + 1/2\, H_u \dot{u} \delta t_f^2 + H_x \delta x_f \delta t_f + 1/2\, H_x \dot{x} \delta t_f^2 \quad . \tag{4.37}$$

The Computational Control Problem may now be solved by finding the solution to the following accessory problem. We must find a control δu and a time δt_f which gives an extreme value for

$$\hat{J} = -\delta\nu'(\psi+\delta\psi) + 1/2\, dx_f'\, \varphi_{xx} dx_f + H_u \delta u_f \delta t_f + 1/2\, H_u \dot{u} \delta t_f^2 + H\delta t_f$$

$$+ H_x \delta x_f \delta t_f + 1/2\, H_x \dot{x} \delta t_f^2 + \int_{t_o}^{t_f} H_u \delta u \, d\sigma + 1/2 \int_{t_o}^{t_f} \delta u' W \delta u \, d\sigma$$

$$+ \frac{1}{2} \int_{t_o}^{t_f} (\delta x' \delta u') \begin{pmatrix} H_{xx} & H_{xu} \\ H_{ux} & H_{uu} \end{pmatrix} \begin{pmatrix} \delta x \\ \delta u \end{pmatrix} d\sigma \tag{4.38}$$

while satisfying the constraints

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

$$\delta x(0) = 0 \qquad \delta\psi = \psi_x\Big|_{t=t_f} dx_f \quad . \tag{4.39}$$

The matrix W is again suitably chosen for convergence as in Section A.

V. THE SOLUTION OF THE AUXILIARY PROBLEM

The central result of the previous chapters was to reduce the Computational Control Problem to a sequence of simplified problems. These simplified problems are similar to the auxiliary or accessory problems studied in the calculus of variations and the name will be retained here. A feedback solution for three special cases of the accessory problem will be obtained, followed by a discussion of conditions sufficient to insure a true extremum.

A. PROBLEM STATEMENT

Rather than follow along with the notation of the last chapter, a simplified problem formulation is used here--the identification of terms corresponding to the actual auxiliary problem will be made in a later chapter. With the new notation, this chapter is self contained.

The system equations to be considered are

$$\dot{q} = Bq + Dw \tag{5.1}$$

where the usual state variable x and control variable u have been replaced by q and w to avoid confusion with the state and control variables for the original problem. The variables q and w were written δx and δu in the previous chapter. In this formulation, q is $n \times 1$, B is $n \times n$, w is $m \times 1$, and D is $n \times m$. B and D are not necessarily constant.

The problem is to extremize the cost criterion given by

$$J = \frac{1}{2} q'(t_f) Q_3 q(t_f) + \int_{t_o}^{t_f} e'w \, d\sigma + \frac{1}{2} \int_{t_o}^{t_f} (q'w') \begin{pmatrix} R & S' \\ S & Q_2 \end{pmatrix} \begin{pmatrix} q \\ w \end{pmatrix} d\sigma \tag{5.2}$$

with the constraint (5.1) and the boundary conditions

$$Aq(t_f) = a$$

$$q(t_o) = 0 \ . \tag{5.3}$$

The matrix A is a $r \times n$ constant matrix which is full rank. The matrices $R(n \times n)$, $Q_2(m \times m)$, $Q_3(n \times n)$, $S(m \times n)$ and the $n \times 1$ vector e may depend on time. Without loss of generality, R, Q_2, and Q_3 are assumed symmetric.

It is well known that the condition $Q_2 \geq 0$ is necessary for this problem. If $Q_2 < 0$ on some interval, J may be made arbitrarily small by a control w which is a large amplitude sinusoid. If the frequency of this added control is high enough, the state will not be changed so that the boundary conditions are also unchanged. This situation is clearly not allowed since the necessary condition for a minimum, that the part of the payoff J which may be controlled be positive definite, is violated. A stronger condition is assumed here, $Q_2 > 0$, which is known as the Strengthened Legendre Condition.

It may not be possible to find a control which generates a trajectory $x(t)$ which satisfies (5.3). To avoid this possible difficulty, it will be assumed that it is possible to reach all points which satisfy $Aq(t_f) = a$ for any a. This is equivalent to requiring the system to be output controllable in the sense that, for any desired output y, there is a control w which produces a trajectory $q(t)$ for which $Aq(t_f) = y$. This condition is not as strong as complete controllability which is usually given for this problem.

The problem may be simplified by completing the square on the quadratic form inside the integral in (5.2). That is

$$q'Rq + q'S'w + w'Sq + w'Q_2w$$

may be written as

$$q'[R - S'Q_2^{-1}S]\, q + (w' + q'S'Q_2^{-1})\, Q_2(w + Q_2^{-1}Sq) \ .$$

This fact may be used to redefine the problem slightly and obtain a more convenient form. Define

$$v = w + Q_2^{-1}Sq$$

$$Q_1 = R - S'Q_2^{-1}S$$

$$F = B - DQ_2^{-1}S$$

$$g = - S'Q_2^{-1}e$$

Equation (5.1) becomes

$$\dot{q} = Fq + Dv \tag{5.4}$$

Equation (5.2) may now be written

$$J = 1/2\ q'(t_f)\ Q_3q(t_f) + \int_{t_o}^{t_f} [e'v + g'q]\ d\sigma$$

$$+ 1/2 \int_{t_o}^{t_f} [q'Q_1q + v'Q_2v]\ d\sigma \tag{5.5}$$

The necessary conditions given in (3.6) may now be directly applied. We define the Hamiltonian in terms of the $n \times 1$ vector function $p(t)$ as

$$\mathcal{H} = p'Fq + p'Dv + e'v + g'q + 1/2\ q'Q_1q + 1/2\ v'Q_2v \tag{5.6}$$

The optimal control, v^*, which minimizes $\mathcal{H}$ is found to be

$$v^* = - Q_2^{-1}[e + D'p] \tag{5.7}$$

The equations usually referred to as the Euler-Lagrange equations are found from (5.6)

$$\dot{q} = \mathcal{H}_p'$$

$$\dot{p} = - \mathcal{H}_q' \ , \tag{5.8}$$

where we substitute (5.7) for v. This leads to $2n$ linear nonhomogeneous equation

$$\dot{q} = Fq - DQ_2^{-1}D'p - DQ_2^{-1}e$$

$$\dot{p} = - Q_1q - F'p - g \ . \tag{5.9}$$

The boundary conditions are given in (5.3) and the added condition

$$p(t_f) = Q_3q(t_f) - A'\mu \tag{5.10}$$

where μ is an $r \times 1$ constant vector of Lagrange multipliers (corresponding to $\delta\nu$ of the last chapter).

The set of equations (5.3), (5.9) and (5.10) give the solution to the problem in terms of a two-point boundary value problem, that is, a differential equation with boundary conditions specified at two points, $t = t_o$ and $t = t_f$. If such a solution exists, the optimal control given in (5.7) may be computed as a time function. However, a more useful form for the optimal control is in a feedback form. That is, the control $v^*(t)$ should be given as a function of the present state $q(t)$. This feedback control should have the property that it gives the optimal control for any initial condition so that it is self compensating for errors in the initial conditions. This feature of the feedback control is not shared by the "open loop" control which is optimal only for the given initial conditions.

A feedback control law of this type may be achieved if it is possible to find a relation giving $p(t)$ as a function of $q(t)$. The question of the existence of such a relationship which gives a unique $p(t)$ for each $q(t)$ for every $t \in [t_o, t_f)$ is still open even if the two-point boundary value problem has a solution. This is an important point which will be discussed further in the latter part of this chapter where it will be shown that the existence of a unique feedback control is both necessary and sufficient to insure that the conditions in (3.6) and the Strengthened Legendre Condition give a true minimum to J.

As an initial step in solving the two-point boundary value problem, the general solution to the linear differential equation will be studied. This general solution may be written as

$$\begin{pmatrix} q(t) \\ p(t) \end{pmatrix} = \Phi(t, \tau) \begin{pmatrix} q(\tau) \\ p(\tau) \end{pmatrix} + \begin{pmatrix} q_p(t, \tau) \\ p_p(t, \tau) \end{pmatrix} \qquad (5.11)$$

where $\Phi(t, \tau)$ is a fundamental matrix* solution to (5.9) and

$$\begin{pmatrix} q_p(t, \tau) \\ p_p(t, \tau) \end{pmatrix}$$

is a particular solution with $q(\tau) = p(\tau) = 0$.

Taking $t = t_f$ and $\tau = t_o$ in (5.11) gives $2n$ equations in the variables $q(t_o)$, $p(t_o)$, $q(t_f)$, and $p(t_f)$. If the total set of $2n + n + n + r$ equations (5.11), (5.10), and (5.3) may be solved for the $4n + r$ variables $q(t_o)$, $p(t_o)$, $q(t_f)$, $p(t_f)$ and μ, the two-point boundary value problem is solved.

The solution for a feedback control, often called the synthesis problem, remains to be solved. If t is replaced by t_f and τ is replaced by t in (5.11), the resulting equation may possibly be solved with equation (5.10) and the first equation in (5.3) to eliminate μ, $p(t_f)$, and $q(t_f)$. This would produce a relation between $p(t)$ and $q(t)$ of the general form

$$M'(t)\, p(t) = N'(t)\, q(t) + b(t) \tag{5.12}$$

where $M(t)$ and $N(t)$ are $n \times n$ matrices and $b(t)$ is an $n \times 1$ vector.

This formal procedure which has been described requires the calculation of $\Phi(t_f, t)$ or $2n$ linearly independent solutions to (5.9). In the

*It will be assumed that the reader is familiar with this and other elementary properties of differential equations. For an excellent treatment of the subject, see Chapter III of the book by Coddington and Levinson [1955].

following sections simplified methods will be derived which involve at most n solutions to (5.9). The motivation for these derivations is a search for a relation of the form given in (5.12).

Having obtained a relation of the form (5.12), the feedback control is found by solving for $p(t)$ in terms of $q(t)$ and substituting the result into the control equation (5.7). Although it will always be possible to find such a relation (5.12), the solution for a unique $p(t)$ in terms of $q(t)$ may not exist. As previously mentioned, this question will be discussed in the later sections of this chapter.

B. CASE I - PROBLEMS WITH FREE END CONDITIONS

The results for problems with free end conditions are quite well known. In this case, a simpler form of (5.12) is obtained in the following. The approach used here will be to assume a special form for (5.12) with undetermined coefficients and to then find coefficients which satisfy the required conditions (5.9), (5.10), and (5.3). For this case $A = 0$ so that the boundary conditions at $t = t_f$ become

$$q(t_f) \sim \text{free}$$

$$p(t_f) = Q_3 q_f \ . \tag{5.13}$$

Assume that there is a nonsingular transformation $P(t)$ which relates $p(t)$ to $q(t)$ by

$$p = Pq + b \ . \tag{5.14}$$

The boundary conditions in (5.13) may be satisfied for all q_f if

$$P(t_f) = Q_3 \tag{5.15}$$

and

$$b(t_f) = 0 \ . \tag{5.16}$$

The additional requirement is that p and q satisfy the Euler-Lagrange equations (5.9). If (5.14) is differentiated with respect to time and (5.9) and (5.14) are used to eliminate the variables p, $\dot{p}$ and $\dot{q}$, the result is

$$(\dot{P} + PF + F'P + Q_1 - PDQ_2^{-1}D'P)\, q = \dot{b} + g + F'b - PDQ_2^{-1}D'b \ . \tag{5.17}$$

Since this relation must hold for all $q(t)$, P and b satisfy the differential equations

$$-\dot{P} = + PF + F'P + Q_1 - PDQ_2^{-1}D'P \tag{5.18a}$$

and

$$\dot{b} = (-F' + PDQ_2^{-1}D')\, b - g \ . \tag{5.18b}$$

Since (5.18a) is symmetric and $P(t_f)$ given by (5.15) is also symmetric, the matrix $P(t)$ is symmetric.

The optimal control is given by

$$v^* = -Q_2^{-1}D'Pq - Q_2^{-1}(D'b + e) \tag{5.19a}$$

or

$$w^* = -Q_2^{-1}(D'Pq + Sq + D'b + e) \tag{5.19b}$$

provided that the solution to (5.18a) exists in the interval from t to

t_f. Equation (5.18a) is a matrix Riccati equation which has the property known as finite escape time. That is, the solutions on finite time intervals may become unbounded. If P becomes unbounded for any $t \in (t_o, t_f)$, (5.19) no longer gives the optimal control. In fact it will be shown later that if the P matrix defined here is not bounded, then not only are there difficulties in obtaining the solution by this method, but any solution to the Euler equations obtained by other means does not minimize J.

C. CASE II - FIXED ENDPOINT PROBLEMS

Problems with completely specified end conditions, often called terminal control problems, have not been studied as actively as the free endpoint problem. Perhaps the reason for this neglect is that the optimal feedback control is not physically realizable. Due to the somewhat artifical requirement that the end conditions be met exactly, the feedback gains increase without bound to compensate for possible terminal errors as the final time is approached. In practice the optimal control is approximated with arbitrarily small error by bounded feedback gains. These difficulties do not influence the mathematical solution which is somewhat similar to the free endpoint solution.

The form for (5.12) assumed here is

$$q = Rp + b \, . \tag{5.20}$$

Again this assumption is verified by finding the matrix R and the vector b such that the boundary conditions ($q(t_f)$ specified) and the Euler-Lagrange equations (5.9) hold.

Following Section B, we differentiate (5.20) and use (5.9) and (5.20) to eliminate q, $\dot{q}$, and $\dot{p}$. The result is

$$(\dot{R} - RQ_1R - RF' + DQ_2^{-1}D' - FR)\, p = \dot{b} - RQ_1b + Rg - DQ_2^{-1}e \qquad (5.21)$$

which must hold for arbitrary $p(t)$. It follows that

$$\dot{R} = RF' + FR + RQ_1R - DQ_2^{-1}D' \qquad (5.22)$$

and

$$\dot{b} = (F + RQ_1)\, b + Rg - DQ_2^{-1}e\ . \qquad (5.23)$$

Since $q(t_f)$ is specified, (5.20) must hold at $t = t_f$ for arbitrary q. This may be satisfied by the choice of boundary conditions for R and b as

$$R(t_f) = 0$$

$$b(t_f) = q(t_f)\ . \qquad (5.24)$$

By the symmetry of (5.22) if R is symmetric, $\dot{R}$ is symmetric. Therefore, the solution $R(t)$ with the symmetric boundary condition $R(t_f) = 0$ will also be symmetric.

The feedback optimal control law is

$$v^* = - Q_2^{-1}D'R^{-1}(q - b) - Q_2^{-1}e$$

or

$$w^* = - Q_2^{-1}[Sq + e + D'R^{-1}(q - b)] \qquad (5.25)$$

which has the property that the coefficient of $(q - b)$ gets arbitrarily large near $t = t_f$ since $R(t_f)$ is singular there. As $t \to t_f$, $b(t) \to q(t_f)$ so that a large control is called for if $q(t)$ is not approaching $q(t_f)$.

The equation for R is again of the Riccati type which may exhibit finite escape time. Provided that R is bounded, the solution obtained for the assumed relation (5.20) holds. Also R must be nonsingular except at $t = t_f$ in order for (5.25) to hold.

D. CASE III - GENERAL LINEAR END CONSTRAINTS

At the beginning of this chapter, the boundary conditions were given in (5.3) as

$$Aq(t_f) = a \ . \tag{5.26}$$

In the past two sections, special results were obtained when the rank of A was either 0 or n. The general case, to be discussed now, deals with $0 \leq \text{Rank } A \leq n$. The relation assumed to exist between $q(t)$ and $p(t)$ is given in (5.18),

$$M'(t)\, p(t) = N'(t)\, q(t) + b(t) \ . \tag{5.27}$$

As before, the differential equations are obtained by differentiation of (5.27) and substitution of the Euler-Lagrange equations to eliminate $\dot{p}(t)$ and $\dot{q}(t)$. The result is

$$(\dot{M}' - M'F' + N'DQ_2^{-1}D')\, p = (\dot{N}' + N'F + M'Q_1)\, q$$

$$+ (\dot{b} - N'DQ_2^{-1}e - M'g) \ . \tag{5.28}$$

Note that (5.27) cannot be used to eliminate either $p(t)$ or $q(t)$ as before unless $M(t)$ or $N(t)$ is nonsingular for $t \in [t_o, t_f]$. In fact, if $M(t)$ is nonsingular, (5.27) may be written

$$p = [(M')^{-1}(N')]\, q + (M')^{-1} b \tag{5.29}$$

which reduces to Case I with the identification

$$P(t) = [M'(t)]^{-1} N'(t) \ . \tag{5.30}$$

By the same reasoning, if $N(t)$ is nonsingular, (5.27) may be solved for $q(t)$ and the resulting identification with Case II is

$$R(t) = [N'(t)]^{-1} M'(t) \ . \tag{5.31}$$

In the general case, $N(t)$ and $M(t)$ may both be singular somewhere in the time interval of interest so that a simplified form for (5.27) is not possible. A sufficient condition for (5.28) to hold for all $p(t)$ and $q(t)$ is that the coefficients of $p(t)$ and $q(t)$ vanish. Therefore, the vector $b(t)$ satisfies

$$\dot{b} = N' D Q_2^{-1} e + M' g \ . \tag{5.32}$$

The equations for $\dot{M}'$ and $\dot{N}'$ obtained by setting the coefficients of $p(t)$ and $q(t)$ equal to zero may be written in a convenient partitioned matrix form:

$$\begin{pmatrix} \dot{M} \\ \dot{N} \end{pmatrix} = \begin{pmatrix} F & -D Q_2^{-1} D' \\ -Q_1 & -F' \end{pmatrix} \begin{pmatrix} M \\ N \end{pmatrix} \ . \tag{5.33}$$

This set of equations is immediately recognized as the homogeneous part of the Euler-Lagrange equations (5.9).

The remaining task is to find a suitable set of boundary conditions for M, N, and b. A more involved procedure is necessary in this case since the boundary conditions do not specify either $p(t_f)$ or $q(t_f)$ completely. The set of conditions on $p(t_f)$ and $q(t_f)$ is

$$p(t_f) = Q_3 q(t_f) - A'\mu \tag{5.34}$$

and

$$Aq(t_f) = a \ . \tag{5.35}$$

In the following, it will be shown that there are n linearly independent vectors $[q'(t_f)\ p'(t_f)]'$ which satisfy (5.34) and (5.35) for arbitrarily selected μ and that this set of vectors may be used to construct boundary conditions for M and N.

Theorem 5.1 If the following assumptions hold

A1. A is full rank $r \leq n$

A2. The $r \times 1$ vector $a \in$ range of A

A3. $Q_3 = P'Q_3P$ where P is a projection operator onto the nullspace of A,

then there are n linearly independent solutions $[q'(t_f)\ p'(t_f)]'$ for arbitrary μ.

The assumptions A1 and A2 have been used throughout this work. A3 assures that the terminal cost is appropriate in that only the unconstrained part of the terminal state contributes to the cost. The first

step in a constructive proof is to find all of the vectors $q(t_f)$ which satisfy (5.35). The number of linearly independent solutions $q(t_f)$ to (5.35) is $n - q$, the dimension of the nullspace of A. Taking $\mu = 0$, Eq. (5.34) may be used with the $n - q$ solution to (5.35) to find $n - q$ linearly independent vectors $[q'(t_f)\ p'(t_f)]'$ which satisfy both (5.34) and (5.35). A set of r additional vectors may be generated by successively setting $\mu' = (1,0,\ldots), (0,1,0,\ldots)$, etc. in 5.34 with $q(t_f)$ taken from the set of $n - q$ solutionsto (5.35). These vectors span an r dimensional space since rank $A' = r$. By A3, $Q_3 q(t_f)$ is in the nullspace of A which is perpendicular to $A'\mu$ for all μ. Therefore, the r additional vectors do not lie in the space spanned by the first $n - r$ vectors. This completes the construction of n linearly independent solutions $[q'(t_f)\ p'(t_f)]'$ to (5.34) and (5.35).

In the following, this set of solutions to the boundary conditions (5.34), (5.35) with $a = 0$ will be used to define the $n \times n$ matrices $M(t_f)$ and $N(t_f)$ as follows

$$M(t_f) = [q_1(t_f)\ q_2(t_f),\ldots,q_n(t_f)]$$
$$N(t_f) = [p_1(t_f)\ p_2(t_f),\ldots,p_n(t_f)] \tag{5.36}$$

where $[q_i'(t_f)\ p_i'(t_f)]'$ is the i^{th} linearly independent solution. These matrices have been named $M(t_f)$ and $N(t_f)$ in anticipation of the proof that they will provide suitable boundary conditions for the matrices $M(t)$ and $N(t)$ previously discussed. Preliminary to this proof, some interesting properties of M and N will be obtained.

Property 1 $M'(t_f)\, N(t_f) = N'(t_f)\, M(t_f)$

Proof: Any q which satisfies (5.35) with $a = 0$ and (5.34) may be written uniquely as $M\zeta_1$, for ζ_1 an $n \times 1$ vector. The corresponding p is $N\zeta_1$. From (5.35) $AM\zeta_1 = 0$ for all ζ_1 or $M\zeta_1 \in$ nullspace of A. From (5.34) $p - Q_3 q = (N - Q_3 M)\, \zeta_2$ is in the range of A' which is perpendicular to the nullspace of A. Hence,

$$\zeta_1' M' (N - Q_3 M)\, \zeta_2 = \zeta_1' [M'N - M'Q_3 M]\, \zeta_2 = 0$$

for all ζ_1 and ζ_2. It follows that $M'(t_f)\, N(t_f)$ is symmetric.

Another useful property of the matrices $M(t)$ and $N(t)$ may be proved with the aid of several properties of the transition or fundamental matrix for the Euler-Lagrange equations. These properties are derived in Appendix D. They enable one to show that the symmetry property of $M'(t)\, N(t)$ for $t = t_f$ holds for all $t < t_f$.

Property 2 $M'(t)\, N(t) = N'(t)\, M(t)$ for all $t < t_f$ if $M'(t_f)\, N(t_f) = N'(t_f)\, M(t_f)$.

The algebraic proof of Property 2 is also given in Appendix D.

Using the matrices $M(t)$ and $N(t)$ any solution to the nonhomogeneous Euler-Lagrange equations (5.9) may be written as

$$\begin{pmatrix} q(t) \\ p(t) \end{pmatrix} = \begin{pmatrix} M(t) \\ N(t) \end{pmatrix} \zeta + \begin{pmatrix} q_p(t) \\ p_p(t) \end{pmatrix} \tag{5.37}$$

where $[q_p'(t)\ p_p'(t)]'$ is a particular solution. The boundary conditions for $[q_p'(t)\ p_p'(t)]'$ can be given, for example, by the minimum norm solution to

$$Aq_p(t_f) = a$$

and

$$p_p(t_f) = Q_3 q_p(t_f) \ .$$

With the parameter ζ, (5.37) describes the well-known n-parameter family of extremals emanating from the terminal manifold.

Premultiplying (5.37) by $[N'(t) - M'(t)]$ gives

$$N'(t)\, q(t) - M'(t)\, p(t) = [N'(t)\, M(t) - M'(t)\, N(t)]\, \zeta + [N'(t)\, q_p(t) - M'(t)\, p_p(t)]. \qquad (5.38)$$

Since $N'(t)\, M(t)$ is symmetric, the coefficient of ζ is zero. The result establishes the following main theorem.

Theorem 5.2 Any solution of the Euler-Lagrange equations (5.9) which satisfies the boundary conditions at $t = t_f$ given by (5.3) satisfies

$$M'(t)\, p(t) = N'(t)\, q(t) + b(t) \qquad (5.39)$$

Where $2n \times n$ matrix $[M'(t)\ N'(t)]'$ satisfies

$$\begin{pmatrix} \dot{M}(t) \\ \dot{N}(t) \end{pmatrix} = \begin{pmatrix} F & -DQ_2^{-1}D' \\ -Q_1 & -F' \end{pmatrix} \begin{pmatrix} M(t) \\ N(t) \end{pmatrix} \qquad (5.40)$$

with the boundary conditions (5.36), and the $n \times 1$ vector $b(t)$ solves

$$\dot{b}(t) = N'DQ_2^{-1}e + M'g \qquad (5.41)$$

with the boundary conditions

$$b(t_f) = -N'(t_f)A'(AA')^{-1}a \ . \qquad (5.42)$$

The only part of this theorem which has not been previously derived is the boundary conditions on $b(t_f)$. They are obtained from (5.38) by identifying $M'(t)\, p_p(t) - N'(t)\, q_p(t)$ with $b(t)$ and substituting $t = t_f$. The term $M'(t_f)\, p_p(t_f) = M'(t_f)\, Q_3 q_p(t_f) = 0$ by assumption A3.

It is clear that the matrices $M(t)$, $N(t)$, and the vector $b(t)$ in (5.39) are not unique. For example, (5.39) still holds if M, N, and b are each multiplied by a nonsingular possibly time-varying matrix. Furthermore, the general boundary conditions specified by (5.36) do not give unique values to $M(t_f)$ and $N(t_f)$. For numerical calculations, it is necessary to describe a specific set of initial conditions for $M(t_f)$ and $N(t_f)$ which are easy to obtain. For this purpose, $M(t_f)$ and $N(t_f)$ will be taken as the partitioned matrices

$$M(t_f) = [B \quad 0] \qquad (5.43)$$

and

$$N(t_f) = [Q_3 B \quad A'] \ . \qquad (5.44)$$

The columns of the $(n - r) \times n$ matrix B form a basis for the nullspace of A. If it is necessary to compute B numerically, it may be obtained by finding the $(n - r)$ eigenvectors with zero eignevalues for the $n \times n$ symmetric matrix $A'A$. In typical problems, this procedure is often unnecessary because the nullspace of A may be determined by inspection.

The optimal control is found from (5.29) and (5.7) as

$$v^* = - Q_2^{-1}[e + D'(M')^{-1}(N'q + b)]$$

so that the original control variable w^* is

$$w^* = - Q_2^{-1}[e + D'(M')^{-1}(N'q + b) + Sq] \qquad (5.45)$$

E. SUFFICIENCY CONDITIONS

In the last three sections, solutions to the necessary conditions for the optimal control problem posed in the first part of this chapter were obtained. The purpose of this section is to determine when these solutions to the necessary conditions are in fact optimal, that is, when they furnish a minimum value for the cost function and meet terminal constraints. It will turn out that this question is related to the existence of solutions to some of the matrix differential equations which was assumed in the last three sections.

In the following, it will be necessary to make several assumptions:

A1, A2, and A3 as in theorem 5.1

A4. $Q_2 > 0$, the strengthened Legendre Condition

A5. The system is completely controllable on any sub-interval of $[t_o, t_f]$.

As previously mentioned, condition A5 may be relaxed somewhat. However, the strong condition A5 will be used here.

In Section D, it was shown that the n-parameter family of solutions to the Euler-Lagrange equations (5.9) and the boundary conditions (5.34) and (5.35) may be written as

$$\begin{pmatrix} q(t) \\ p(t) \end{pmatrix} = \begin{pmatrix} M(t) \\ N(t) \end{pmatrix} \zeta + \begin{pmatrix} q_p(t) \\ p_p(t) \end{pmatrix}$$

The solutions $q(t)$ are known as extremals. If there is a unique extremal passing through every point of a region $R \subset E_n$, the region R is said to be covered by a <u>field of extremals</u>. Two extremals which have corresponding parameters ζ_1 and ζ_2 arbitrarily close are said to be neighboring extremals because a measure of their separation $[q_1(t) - q_2(t)]'[q_1(t) - q_2(t)]$ is bounded by $K\|\zeta_1 - \zeta_2\|$ for some $K < 0$, where $\|M(t)\| < K$ for all t belonging to a finite interval. If at some time t^* two neighboring extremals cross, the point $q_1(t^*) = q_2(t^*)$ cannot belong to the region R which is covered by a field containing $q_1(t)$. This situation is made more precise in the definition of a conjugate point.†

<u>Definition</u> If two neighboring extremals $q_1(t)$ and $q_2(t)$ cross at $t = t^*$, i.e., $q_1(t^*) = q_2(t^*)$, then the extremal $q_1(t)$ (or $q_2(t)$) is said to have a <u>conjugate point</u> at $t = t^*$.

If there are two vectors ζ_1, ζ_2 which satisfy, for $i = 1,2$ $q_i(t) = M(t)\,\zeta_i + q_p(t)$ and for which $q_1(t^*) = q_2(t^*)$ then there is a conjugate point at $t = t^*$. In such a case, $M(t^*)$ must be singular and any ζ of the form $\zeta_1 + \zeta_o$ will also produce an extremal passing through $q(t^*) = q_1(t^*)$ if $M(t^*)\,\zeta_o = 0$. This shows that there are an infinite number of extremals passing through a conjugate point and leads to the following equivalent definition:

---

†The exact definition of a conjugate point is not completely standardized in the literature. The situation is further confused by definitions which include statements as "the point $t = t^*$ is conjugate to the point $t = t_f$" since there are three points to contend with, the "point" $t = t^*$, the "point" $q(t^*)$, and the "conjugate point." The above definition only mentions one point as such. This definition will be connected to other possible definitions in the following pages.

Alternate Definition If $\det M(t) = 0$ for $t = t^* < t_f$, then there is a conjugate point at $t = t^*$.

The relationship between the matrix $M(t)$ and the solutions to the Riccati equation $P(t)$ (Case I, Section B) and $R(t)$ (Case II, Section C) given in (5.30) and (5.31) may be used to connect the idea of conjugate points to the existence of solutions to the differential equations (5.18) and (5.22). The results, stated in the form of two lemmas, may also be used as possible conjugate point definitions.

Lemma 1 For the free endpoint problem (Case I), the matrix Riccati equation

$$-\dot{P} = PF + F'P + Q_1 - PDQ_2^{-1}D'P$$

with $P(t_f) = Q_3$, has a solution on $[t_o, t_f]$ if and only if there are no conjugate points in $[t_o, t_f]$.

Proof: For the free endpoint problem, the appropriate boundary conditions for $M(t)$ and $N(t)$ are $M(t_f) = I$, $N(t_f) = Q_3$. Since $\det M(t_f) \neq 0$ by the continuity of the solution to the differential equation for $M(t)$, there is an interval $(\epsilon, t_f]$ over which $\det M(t) \neq 0$. By direct substitution $P(t)$ and $[M'(t)]^{-1} N'(t) = N(t) M^{-1}(t)$ satisfy the same differential equations. Since also $P(t_f) = Q_3 = N(t_f) M^{-1}(t_f)$, $P(t) = N(t) M^{-1}(t)$ on $(\epsilon, t_f]$ by the uniqueness of the solution to the differential equations. In order for $P(t)$ to become unbounded, then $M(t)$ must be singular since $N(t)$ satisfies a linear homogeneous differential equation and cannot become unbounded in finite time. Conversely, if there is a conjugate point, $\det M(t) \to 0$ and hence $P(t)$ becomes unbounded.

Lemma 2 For the fixed endpoint problem (Case II), the matrix Riccati equation

$$\dot{R} = RF' + FR + RQ_1R - DQ_2^{-1}D'$$

with $R(t_f) = 0$, has a nonsingular solution on $[t_o, t_f)$ if and only if there are no conjugate points in the interval $[t_o, t_f)$.

Proof: If R is nonsingular on $[t_o, t_f)$ then by direct substitution it may be shown that $R^{-1}(t)$ satisfies the same differential equation as $P(t)$. If $R^{-1}(t_1) = P(t_1)$ at some $t = t_1$, then $R^{-1}(t) = P(t)$ for all $t \in [t_o, t_f)$. $P(t)$ is bounded since $R(t)$ is nonsingular, hence there are no conjugate points in $[t_o, t_f)$. Now it is assumed that there are no conjugate points in $[t_o, t_f)$. Then $M(t)$ is nonsingular and $N(t)\,M^{-1}(t) = P(t)$ is bounded. Therefore $R(t)$ is nonsingular.

The construction of a feedback optimal control in Sections B, C, and D required that either $P(t)$, $R^{-1}(t)$, or $M^{-1}(t)$ exist for $t \in [t_o, t_f)$ for each of the three cases. It is clear that if there are no conjugate points, then it is possible to construct a unique feedback control. In fact, it can be shown that if there is a conjugate point along an extremal, no optimal feedback control law exists.

If it can be determined that a solution to the necessary conditions has no conjugate points, then the following theorem may be used to establish optimality.

Theorem 5.3: If a solution $[q(t), p(t)]$ to the Euler-Lagrange equations exists which satisfies the boundary conditions and

1. A1, A2, and A3 of Theorem 5.1

2. A4 and A5 of this section

3. There are no conjugate points in $[t_o, t_f)$

then $[q(t), p(t)]$ furnishes a minimum for the cost functional J.

Proof: Assume there is another control $\mu = -Q_2^{-1}D'p + \delta u$, $\delta u \neq 0$, which results in a trajectory $\hat{q}$ which satisfies the boundary conditions. The difference trajectory, $\delta q = \hat{q} - q$, is a solution to $\delta\dot{q} = F\,\delta q + D\,\delta u$. The difference in the cost on the original trajectory, q, and on the trajectory $\hat{q}$ is

$$2\Delta J = \int_{t_o}^{t_f} [q'Q_1\delta q + \delta q'Q_1 q + \delta q'Q_1\delta q - p'D\delta u$$

$$- \delta u'D'p + \delta u'Q_2\delta u]d\sigma \qquad (5.46)$$

The expression for J may be written in a more convenient form by adding the integral

$$\int_{t_o}^{t_f} \frac{d}{dt}\,[(p' + \delta q'A(t))\delta q]d\sigma = 0 \qquad (5.47)$$

where $A(t)$ is an arbitrary $n \times n$ matrix. Since $\delta q(t_o) = \delta q(t_f) = 0$, the integral in (5.47) is equal to zero. Carrying out the indicated differentiation and substituting the differential equations for δq and p, the results are

$$\int_{t_o}^{t_f} \{-qQ_1\delta q + p'D\delta u + \delta q'A(t)DQ_2^{-1}D'q - \delta q'F'A'(t)\delta q]d\sigma$$

(continued)

$$+ \int_{t_o}^{t_f} \{q'DQ_2^{-1}D'A'(t)\delta q + \delta q'\dot{A}(t)\delta q - \delta q'A(t)F\delta q\}d\sigma \ . \qquad (5.48)$$

Since there are no conjugate points in $[t_o, t_f)$, the matrix Riccati equation has a solution there by Lemma 1. If $A(t) = P(t)$, a solution to the Riccati equation, then (5.48) becomes

$$\int_{t_o}^{t_f} \{-qQ_1\delta q + \delta q'PD'Q_2^{-1}D\delta q - \delta q'PDQ_2^{-1}D'q - q'DQ_2^{-1}D'P\delta q$$

$$- \delta qQ_1\delta q + p'D\delta u\}d\sigma \ . \qquad (5.49)$$

Adding (5.49) to (5.46),

$$2\Delta J = \int_{t_o}^{t_f} \{\delta q'PD'Q_2^{-1}D\delta q - \delta q'PDQ_2^{-1}D'q - qDQ_2^{-1}D'P\delta q$$

$$+ \delta u'Q_2\delta u\}d\sigma \qquad (5.50)$$

which may be written

$$2\Delta J = \int_{t_o}^{t_f} (\delta u + Q_2^{-1}DP\delta q)'Q_2^{-1}(\delta u + Q_2^{-1}DP\delta q)d\sigma \geq 0 \ . \qquad (5.51)$$

The above expression is non-negative since Q_2^{-1} is nonsingular. If the equality holds, then $\delta u = -Q_2^{-1}DP\delta q$ so that δq must satisfy a linear homogeneous differential equation. Since $\delta q(t_o) = 0$, $\delta q(t) \equiv 0$. Therefore for any control $u \neq -Q_2^{-1}D'p$, $\Delta J > 0$.

In the paper by Breakwell and Ho [1965], it is shown that the conjugate point condition is also necessary for the control problem. That is, if an extremal is also a minimizing trajectory, then there must be no conjugate points in $[t_o, t_f)$. This result is well known for the classical Bolza problem (see Bliss [1946] Chapter 9, or Gelfand and Fomin [1963] Chapter 5). In fact, for problems which are normal (in the sense of Bliss) and for which the Hamiltonian has a unique minimizing function $u(t)$ for each t (called nonsingular in the control literature), then the control $u(t)$ may be eliminated and the results of the classical calculus of variations may be applied to the control problem.

By the foregoing, computational methods based on second variations which do not test for conjugate points cannot be guaranteed to succeed. On the other hand, any method which generates a feedback control similar to the one developed in this chapter automatically tests for conjugate points.

VI. THE COMPUTATIONAL METHOD

In Chapter III, the so-called Computational Control Problem, that of finding a "better" control, was reduced to one of examining the expanded version of the cost functional to quadratic terms. Chapter IV showed how this expansion could be carried out for several special problems of interest and further reduced the problem to one of studying a special form of control problem, the linear quadratic loss problem. The next chapter, V, was concerned with finding optimal feedback controls for general linear plant quadratic loss problems. The purpose of the present chapter is to combine all of these previous results into a useful computational algorithm. The properties of the solutions and some of the details of the programs developed by the author for machine solution will be discussed in the last sections of this chapter.

A. OUTLINE OF THE COMPUTATIONAL TECHNIQUE

For the purpose of presenting an introductory overall picture of the type of calculations necessary, a simplified flow diagram of the procedure is given in Fig. 6.1. After describing how the procedure is carried out, the justification for the method will be given. In the first problem to be discussed, it will be assumed that the final time t_f is specified, general end conditions are given (Case III of Chapter V), the initial conditions on the state $x(o) = x_o$ are given, the functions $\Phi[x(t_f)]$, $\psi[x(t_f)]$ and $f(x, u)$ are twice continuously differentiable in all of their arguments, and that an unconstrained control function $u(t)$ is

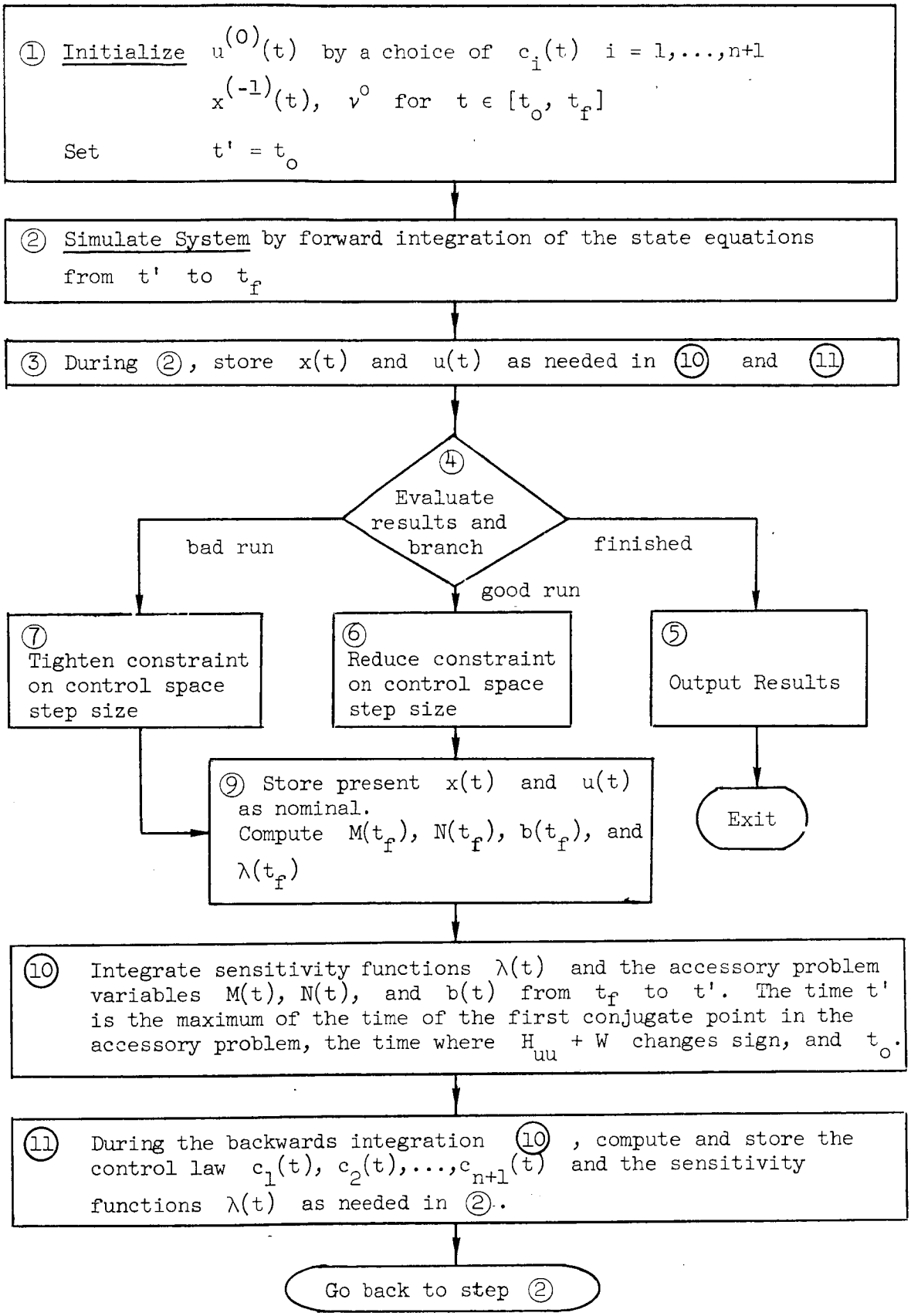

FIG. 6.1 SIMPLIFIED FLOW CHART FOR COMPUTING OPTIMAL CONTROLS USING SECOND VARIATIONS

to be found which gives an extreme value to the payoff $\phi[x(t_f)]$ while satisfying the terminal constraints $\psi[x(t_f)]$.

To start the computation, it is necessary to guess an initial control $u^{(0)}(t)$ (see ① in Fig. 6.1). This guess may be given as a function of time only or as a function of time plus a possibly time-varying linear combination of the states as feedback. That is, the user starts the program with values for the time functions $c_i(t)$ $i = 1,\ldots,n+1$. The nominal trajectory is then computed from the control law

$$u(t) = c_1(t)\, x_1(t) + c_2(t)\, x_2(t) + \ldots + c_n(t)\, x_n(t) + c_{n+1}(t) \tag{6.1}$$

and the state equation

$$\dot{x} = f(x, u),\ x(0) = x_o\ . \tag{6.2}$$

Of course, the convergence is improved by a fortunately good initial guess of the control. However, step by step improvement may be obtained with very poor initial guesses, and good starting controls, which are required by other numerical methods, are not necessary to insure convergence.

For the first iteration, it is necessary to guess starting values for the $q \times 1$ vector of Lagrange multipliers ν. The choice of good numerical values is aided by the physical interpretation of the ν's as sensitivities as in Chapter III. For this problem, the i^{th} component of ν, ν_i, is the sensitivity of the extreme value obtained for the payoff $\phi[x(t_f)]$ due to a small change in the value of the i^{th}

component of $\psi[x(t_f)]$. In other words, the extreme value of $\phi[x(t_f)]$ will change by the amount $\nu_i \epsilon_i$ when $\psi_i[x(t_f)]$ is changed by ϵ_i.

At the end of each simulation of the system in step ②, the results are evaluated by comparing the values for the payoff and the terminal constraints with the values from the previous iteration. If the payoff did not improve while the constraints remained within tolerances or if the constraint errors did not decrease and were too large, it is concluded that the change in control was too great. Therefore, the constraint on the step size given by

$$\|\delta u\| = \int_{t_o}^{t_f} \delta u' W \delta u \, dt \leq c \tag{6.3}$$

is made tighter for the next iteration. On the other hand, if the payoff and the constraints are both improved or remain unchanged, the iteration is considered successful. If the number of successful iterations is equal to the maximum number specified by the user as input data, or if the method has converged as indicated by no change in either the payoff or the constraints with $\|\delta u\|$ effectively unconstrained, the program outputs all of the results necessary for properly restarting the program and reads in the data for a new problem. At the end of a successful iteration which does not cause an exit, the states and the control are stored as the new nominal and the constraint on $\|\delta u\|$ is reduced for the next iteration.

The purpose of the backwards integration is to solve the accessory problem discussed in Chapter V and to find the sensitivity functions

$\lambda(t)$. The solution to the backwards integration is used to generate a correction to the control u and to the terminal constraint sensitivities ν.

The equations for the backwards integration are obtained by identifying the solution to the general problem found in Chapter V, Section A, with the linear quadratic loss problem derived in Chapter IV. This correspondence is

$$
\begin{aligned}
Q_3 &\leftarrow \varphi_{xx}(x_f) \\
Q_2 &\leftarrow H_{uu} + W \\
Q_1 &\leftarrow H_{xx} - H_{xu}(H_{uu} + W)^{-1}H_{ux} \\
F &\leftarrow f_x - f_u(H_{uu} + W)^{-1}H_{ux} \\
e &\leftarrow H'_u \\
g &\leftarrow -H_{xu}(H_{uu} + W)^{-1}H'_u \\
D &\leftarrow f_u \\
A &\leftarrow \psi_x \\
a &\leftarrow \delta\psi \\
p &\leftarrow \delta\lambda \\
q &\leftarrow \delta x \\
v &\leftarrow \delta u - (H_{uu} + W)^{-1}H_{ux}\delta x \ .
\end{aligned}
\tag{6.4}
$$

With these appropriate substitutions the equations for the backwards integration and boundary conditions are obtained from the last chapter.

The boundary condition on $\lambda(t)$ at $t = t_f$ is given by

$$\lambda(t_f) = \phi_x'[x(t_f)] - \psi_x'[x(t_f)]\,\nu \ . \tag{6.5}$$

Consequently, both $x(t_f)$ and ν must be updated before finding the new value of $\lambda(t_f)$. At first, it may seem reasonable to try to find $\delta\lambda(t_f)$ from the relation

$$M'(t_f)\,\delta\lambda(t_f) = N'(t_f)\,\delta x(t_f) + b(t_f) \tag{6.6}$$

where $M(t)$, $N(t)$, and $b(t)$ are from the previous iteration and $\delta x(t_f) = x^{(n)}(t_f) - x^{(n-1)}(t_f)$.† Then the new $\lambda^{(n+1)}(t_f)$ is calculated from

$$\lambda^{(n+1)}(t_f) = \lambda^{(n)}(t_f) + \delta\lambda(t_f) \ . \tag{6.7}$$

However, this is not possible since $M(t_f)$ is singular at $t = t_f$. In fact since $M'(t_f)\,\psi_x' = 0$, (6.7) cannot even by used to find $\delta\nu$ by taking $\delta\lambda = \varphi_{xx}\delta x - \psi_x'\delta\nu$. As an alternative to solving (6.6) for $\delta\lambda(t)$ at $t = t_f$, it may be solved at several points $t = t_f - \epsilon$, $t_f - 2\epsilon$, $t_f - 3\epsilon$ near $t = t_f$ and the result extrapolated to the end. This method has also met with little success in practice probably due to the difficulty of solving (6.6) when $M(t)$ is almost singular. More reliable results have been obtained by solving for $\delta\lambda(t)$ at some time $t = t'$ where $M(t)$ has become suitably well conditioned. The result is then extrapolated to $t = t_f$ by integrating the differential

†The superscript on $x^{(n)}(t_f)$ is used to denote the values of $x(t_f)$ from the n^{th} iteration.

equation for $\delta\lambda(t)$ which is obtained from (5.9) with the substitutions (6.4) as

$$\dot{\delta\lambda} = [-f'_x + H_{xu}(W + H_{uu})^{-1}f'_u]\ \delta\lambda$$

$$[H_{xu}(W + H_{uu})^{-1}H_{ux} - H_{xx}][x^{(n+1)}(t) - x^{(n)}(t)]$$

$$+ H_{xu}(W + H_{uu})^{-1}H'_u \ . \qquad (6.8)$$

where the partials of f and H are evaluated along the n^{th} trajectory.

In order to insure that the resulting $\lambda(t_f)$ satisfies (6.5), it is necessary to remove the part of $\lambda^{(n)}(t_f) + \delta\lambda(t_f) - \phi'_x[x^{(n+1)}(t_f)]$ which is perpendicular to ψ'_x. That is, after computing $\delta\lambda(t_f)$ from (6.8), $\nu^{(n+1)}$ is found as the least square solution to

$$\psi'_x[x^{(n)}(t_f)]\ \nu^{(n+1)} = \phi'_x[x^{(n)}(t_f)] - \lambda^{(n)}(t_f) - \delta\lambda(t_f) \ .$$

The solution is

$$\nu^{(n+1)} = [\psi_x\psi'_x]^{-1}\ \psi_x[\phi'_x - \lambda(t_f) - \delta\lambda(t_f)] \ . \qquad (6.9)$$

The boundary conditions on $\lambda(t)$ at $t = t_f$ may then be determined from (6.5).

The equations for $M(t_f)$, $N(t_f)$ and $b(t_f)$ are obtained by translating (5.42) and (5.43) with the "dictionary," (6.4). The results are

$$M(t_f) = [B \quad 0]$$

(continued)

$$N(t_f) = [Q_3 B \quad \psi_x']$$

$$b(t_f) = -N'(t_f)\,[\psi_x \psi_x']^{-1}\,\psi_x' \delta\psi \ . \tag{6.10}$$

The columns of the $n \times n - q$ matrix B are the $n - q$ linearly independent vectors which are perpendicular to the rows of ψ_x. The determination of B has been discussed in the last section of Chapter V. Normally $Q_3 = \varphi_{xx}$, but in some cases it may require some modification which is discussed later in this section.

From equations (5.40) and (5.41) together with (6.4), the differential equations for $M(t)$, $N(t)$, and $b(t)$ are obtained as

$$\begin{pmatrix} \dot{M} \\ \\ \dot{N} \end{pmatrix} = \begin{pmatrix} [f_x - f_u(W + H_{uu})^{-1}H_{ux}] & [-f_u(H_{uu} + W)^{-1}f_u'] \\ \\ [H_{xu}(H_{uu} + W)^{-1}H_{ux} - H_{xx}] & [H_{xu}(W + H_{uu})^{-1}f_u' - f_x'] \end{pmatrix} \begin{pmatrix} M \\ \\ N \end{pmatrix} \tag{6.11}$$

and

$$\dot{b} = N'f_u(H_{uu} + W)^{-1}H_u' - M'H_{xu}(H_{uu} + W)^{-1}\,H_u' \ . \tag{6.12}$$

On the next iteration, the new control is obtained by adding the correction δu to the old control $u(t)$. The expression for δu, which corresponds to w of Chapter V, is found by combining (5.45) and (6.4).

$$\delta u = -(H_{uu} + W)^{-1}[H_u' + H_{ux}\delta x + f_u'(M')^{-1}(N'\delta x + b)] \ . \tag{6.13}$$

The new control becomes

$$u^{n+1}(t) = u^n(t) - (H_{uu} + W)^{-1}[H_u' - H_{ux}x^{(n)} + f_u'(M')^{-1}\,(b - N'x^{(n)})]$$

(continued)

$$-(H_{uu} + W)^{-1}(H_{ux} + f'_u(M')^{-1}N')\, x^{n+1}(t) \tag{6.14}$$

which may be written in the form

$$u^{n+1}(t) = c_1(t)x_1(t) + c_2(t)x_2(t) + \ldots + c_n(t)x_n(t) + c_{n+1}(t) \tag{6.15}$$

with the definitions

$$(c_1 c_2 \ldots c_n)' = -(H_{uu} + W)^{-1}(H_{ux} + f'_u(M')^{-1}N') \tag{6.16}$$

and

$$c_{n+1} = u^n(t) - (H_{uu} + W)^{-1}[H'_u - H_{ux}x^{(n)} + f'_u(M')^{-1}(b - N'x^{(n)})]\,. \tag{6.17}$$

The partial derivatives of H and f are again to be evaluated along the nominal (old) trajectory. This is obvious if the c's are evaluated and stored during the backward integration since the new trajectory is not yet available. However, some confusion might arise if the c's were calculated during the forward integration which may also be done although it requires more storage.

The reverse time integration is continued until t_o is reached, or the determinant of $H_{uu} + W$ changes sign, or the determinant of M changes sign, whichever occurs first. If t_o is not reached, the starting time for the next forward integration, t', is set slightly to the right of the exit time in the backward integration. The test of the sign of the determinant of $H_{uu} + W$ insures that one of the assumptions made in Chapter V, the Strengthened Legendre Condition, is satisfied on the interval $(t' + \epsilon, t_f]$ if ϵ is at least one numerical integration

step size. This test requires no additional program effort since the inverse of $H_{uu} + W$ is already required. The test on the determinant of M checks for no conjugate points in (t', t_f), a necessary condition for the control computed in (6.13) to be optimal for the accessory problem. This test is also almost automatic since the determinant of M is computed in the calculations necessary for finding c_i, $i = 1,\ldots,n+1$ from (6.16) and (6.17).

Having computed the new control law by the coefficients $c_i(t)$, $i = 1,\ldots,n+1, t \in [t' + \epsilon, t_f]$, the resulting control is evaluated by returning to step ② in Fig. 6.1 and integrating forward from $t = t'$. The initial conditions for the states at t' are obtained from the stored values of the last trajectory $x^{(n)}(t)$ at $t = t'$. The process is then continued until the result of the test at ④ produces an exit to ⑤.

Due to the near singularity of $M'(t)$ which prevented an accurate determination of $\delta\lambda(t)$ from (6.6) for t near t_f, there are corresponding problems in computing the feedback coefficients $c_1, c_2, \ldots, c_n(t)$ as the terminal time is approached. Further difficulties are caused by the very large feedback gains which lead to instabilities in the numerical integration of the state equations. Good results have been obtained by changing to a type of open-loop corrections in an interval $[\tau, t_f]$. It is convenient to take the interval $[\tau, t_f]$ the same as the interval chosen for the integration of the differential equation for $\delta\lambda(t)$ in determining $\delta\lambda(t_f)$. The open-loop control correction is then computed from

$$\delta u = -(H_{uu} + W)^{-1}[H_u' + H_{ux}(x^{(n+1)} - x^{(n)}) + f_u'\delta\lambda(t)] \quad .$$

In order to show the validity of the technique, it is only necessary to collect together some of the previous results and verify that the computational method satisfies the necessary assumptions.

The Computational Control Problem was reduced, in Chapter III, to a consideration of the expansion of the functional $F(x, u, \lambda, \nu)$ to second-order terms only. By taking $\|\delta u\|$ and $\|\delta\psi\|$ sufficiently small, the quantities $\|\delta x\|$, $\|\delta\lambda\|$, and $\|\delta\nu\|$ are also small so that the higher-order terms in the expansion may be neglected. This condition is insured in the program by increasing $\|W\|$, which is equivalent to tightening the constraint on $\|\delta u\|$, until a particular iteration is successful.

The next assumption, in Chapter III, concerned normality. In assuming normality, $\nu_o \neq 0$; therefore it was set equal to unity. This operation may be viewed in another way as the result of dividing each ν_i through by ν_o so that as an abnormal solution is approached each of the ν_i's (which are actually ν_i/ν_o) become very large. The effect will be to produce a control which concentrates on the end constraints and ignores the payoff. No experience of applying this computational method to problems which are abnormal is available at this time. However, the relative sizes of the ν_i's $i = 1,\ldots,n$ as the extremum is approached give a crude numerical test for abnormality.

The final step in the proof that the computational scheme as described has step by step convergence is to show that the solution to the accessory problem actually furnishes a minimizing curve. This may be done by showing that each member of the set of sufficient conditions in Theorem 5.3 is satisfied. These conditions for the problem under consideration here are

A1'. ψ_x is full rank $q \leq n$

A2'. There is a vector δx which satisfies $\psi_x \delta x = \delta\psi$

A3'. $\varphi_{xx} = P'(\varphi_{xx})\, P$ where P is a projection onto the nullspace of ψ_x

A4'. $(H_{uu} + W) > 0$

A5'. The system in the accessory problem is completely controllable on any subinterval of $[t', t_f]$

A6'. There are no conjugate points in $[t', t_f]$.

If the problem has been properly formulated and $\hat{x}(t)$ is an optimal trajectory, $\psi_x[\hat{x}(t_f)]$ will have full rank. Otherwise, one or more of the constraints is redundant and has no effect on the problem solution. However, $\psi_x[x(t_f)]$ may not be full rank if $x(t_f)$ is not optimal even if the constraints are linearly independent for $x(t_f) = \hat{x}(t_f)$. Since the program requires the inversion of $\psi_x\psi_x'$, a test for the rank of ψ_x is automatically made. Although it is unlikely that $\psi_x\psi_x'$ will ever appear singular in practice due to the inevitable numerical errors in the inversion, this situation can be remedied by temporarily dropping the redundant constraints. This can be accomplished in principle by extracting a basis for the range of ψ_x and using this in place of ψ_x. If one column of ψ_x is a multiple of another, it may simply be removed. As a last resort, a Gram-Schmidt procedure (see e.g., Shilov [1961] Chapter 8) might be used to reduce ψ_x to a matrix of full rank. Of course, it may be possible to determine from the functional form of $\psi_x[x(t_f)]$ that it is full rank for all $x(t_f)$ and avoid the test all together. In any event it is possible to redefine the problem so that

$\psi_x[x(t_f)]$ is full rank and hopefully this will not be necessary. A1' is therefore satisfied.

A2' is then automatically satisfied if the original (unmodified) ψ_x was full rank. If it was necessary to construct a basis for ψ_x as given above, $\delta\psi$ must perhaps also be modified so that it is in the range of the new ψ_x. This can be done if necessary since $\delta\psi$ is specified independently by the user although it is usually taken equal to $-\psi$.

Conditions A3' and A4' may be satisfied by construction. If A3' does not hold, φ_{xx} may be replaced by $P'\varphi_{xx}P$ in the accessory problem so that then A3' will be satisfied. For any bounded H_{uu}, there is a W suitably large which satisfies A4'.

A5' and A6' are forced to hold by the choice of t'. The program determines $t' - \epsilon$ as the maximum of the time where the determinant of M (or P or R in cases I and II) changes sign, the time when the determinant of $H_{uu} + W$ changes sign, and the initial time t_o for the original problem. Thus on the interval over which the accessory problem is solved, $H_{uu} + W > 0$ and $\det(M) \neq 0$. These additional conditions are sufficient to show that A5' and A6' hold.

B. EXTENSION TO OTHER TYPES OF PROBLEMS

The computational method of the last section may be modified so that it is applicable to the several different types of problems as discussed in Chapters IV and V. Second-order techniques for handling problems with free endpoints, completely specified endpoints, free terminal time, control parameters, and variable switching times will now be considered.

The only reason for deriving special methods for problems with free end conditions (Case I) or completely specified end conditions (Case II) is to obtain more efficient computation since the general case still applies. The saving in computation is quite substantial particularly in the free endpoint case. The general case requires the integration of the differential equations for M, N, and b, a total of $2n^2 + n$ equations, and the inversion of an $n \times n$ matrix at every integration step. In comparison, Case II requires the integration of $1/2\ n(n + 1) + n$ equations and an $n \times n$ matrix inversion at each integration step. Case I is even easier to compute as it requires $1/2\ n(n + 1) + n$ equations to be integrated and no matrix inversions are needed in computing the control.

The equations for Case I and Case II may be obtained by reinterpreting the results given in Chapter V with the aid of (6.4). The results are summarized in Fig. 6.2. In addition to computing P and b (or R and b) and their boundary conditions, the test for conjugate points, the calculation of $\delta\lambda(t_f)$, and the calculation of δu must also be changed for Case I or Case II. The conjugate point test is made by checking for a change in the sign of the determinant of M, or R, or by checking to see if the norm of P becomes too large. In Case I, there is no need to compute $\delta\lambda(t_f)$ since $\lambda(t_f)$ is known to be equal to zero. For Case II $\delta\lambda(t)$ at some point t' near t_f may be obtained by solving $R(t)\ \delta 2(t) = \delta x(t) + b(t)$ and extrapolating the result to $t = t_f$ by solving the differential equation for $\delta\lambda(t)$ as before. The method for finding δu in each case is given in Fig. 6.2.

It is natural to question why problems with end constraints (Cases II and III) appear to be so much more difficult in terms of computation than

problems with unspecified end conditions (Case I). In the last chapter, M, N, R, and P were shown to be related by

$$[M'(t)]^{-1}N'(t) = R^{-1}(t) = P(t) \ . \tag{6.18}$$

This relationship is the key to the difficulty. For problems of Case II $R(t_f)$ is singular, and for problems of Case III $M(t_f)$ is singular so that there is no suitable boundary condition for $P(t)$ at $t = t_f$ in either case. However, at any other time t', which is not a conjugate point, $P(t')$ may be found if either M and N or R is known. Accordingly, at such a point $P(t')$ and $b(t')$ of Case I may be found from $R(t')$ and $b(t') = b_{II}(t')$ for Case II by

$$\begin{aligned} P(t') &= R^{-1}(t') \\ b(t') &= R^{-1}(t')\, b_{II}(t,) \ . \end{aligned} \tag{6.19}$$

Similarly, the variables of Case I and Case III are related by

$$\begin{aligned} P(t') &= [M'(t')]^{-1}N'(t') \\ b(t') &= [M'(t)]^{-1}b_{III}(t') \ . \end{aligned} \tag{6.20}$$

In order to avoid the added computations in Case II and III, in principle one would pick t' very near to t_f, use (6.19) or (6.20) to find $P(t')$ and $b(t')$, and then work the problem over the remaining interval from t' back to t_o as if it were Case I. In practice t' should be determined far enough away from $t = t_f$ so that $R(t')$ [or $M'(t')$] becomes well conditioned enabling accurate numerical results in (6.19) or (6.20). The advantage of this modification for a

Definitions

$$F = f_x - f_u(H_{uu} + W)^{-1} H_{ux}$$

$$Q = f_u(H_{uu} + W)^{-1} f_u'$$

$$S = H_{xx} - H_{xu}(H_{uu} + W)^{-1} H_{ux}$$

$$c = - f_u(H_{uu} + W)^{-1} H_u'$$

$$d = H_{ux}(H_{uu} + W)^{-1} H_u'$$

B is any $n \times n - q$ full rank solution to $\psi_x B = 0$

Case I (Free Endpoint) $\delta\lambda = P\delta x + b$

$$\dot{P} = -F'P - PF - S + PQP$$

$$\dot{b} = -(F' - PQ)\, b + d - Pc$$

$$P(t_f) = \varphi_{xx}, \qquad b(t_f) = 0$$

$$\delta u = -(H_{uu} + W)^{-1}[H_u' + H_{ux}\delta x + f_u(P\delta x + b)]$$

Case II (Fixed Endpoint) $R\delta\lambda = \delta x + b$

$$\dot{R} = FR + RF' + RSR - Q$$

$$\dot{b} = (F + RS)\, b + c - Rd$$

$$R(t_f) = 0, \quad b(t_f) = - \delta x(t_f)$$

$$\delta u = -(H_{uu} + W)^{-1}[H_u' + H_{ux}\delta x + f_u'R^{-1}(\delta x + b)]$$

Case III (General) $M'\delta\lambda = N'\delta x + b$

$$\dot{M} = FM - QN$$

$$\dot{N} = -SM - F'N$$

$$\dot{b} = M'd - N'c$$

$$M(t_f) = [B \quad 0]$$

$$N(t_f) = [\varphi_{xx}B \quad \psi_x']$$

$$b'(t_f) = [0 \quad \delta\psi']$$

$$\delta u = -(H_{uu} + W)^{-1}[H_u' + H_{ux}\delta x + f_u'(M')^{-1}(N\delta x + b)]$$

FIG. 6.2 SUMMARY OF RESULTS FOR THE ACCESSORY PROBLEM

Case III problem with four state variables is that 18 first-order differential equations are solved from t' to t_o as compared to 40 first-order differential equations and the inversion of a 4×4 matrix at each integration step.

An additional simplification occurs when the problem is in the Mayer form. In this case, it may be easily demonstrated by differentiation and substitution of the equations for $\dot{M}$, $\dot{\lambda}$, and $\dot{b}$, that the expression $M'(t)\ \lambda(t) + b(t)$ is constant so that

$$M'(t)\ \lambda(t) + b(t) = M'(t_f)\ \lambda(t_f) + b(t_f)\ . \tag{6.21}$$

There is therefore no need to integrate the equations for the n components of $b(t)$ in this case since $b(t)$ may be determined from $M(t)$ and $\lambda(t)$ from (6.21).

The extension of the computing method to problems with free terminal time requires considering the terms involving δt_f in (4.41). The part of $\hat{J}$ which depends on δt_f is

$$(1/2\ \dot{x}'\varphi_{xx}\dot{x} + 1/2\ H_u\dot{u} + 1/2\ H_x\dot{x})\ \delta^2 t_f + (H + x'\varphi_{xx}\dot{x}$$

$$+ H_u\delta u + H_x\delta u)\ \delta t_f$$

where the terms in parenthesis are evaluated on the n^{th} iteration at $t = t_f^{(n)}$. The quadratic form is extremized by setting

$$\delta t_f^* = \frac{(x'\varphi_{xx}\dot{x} + H_u\delta u + H_x\delta x + H)}{(\dot{x}'\varphi_{xx}\dot{x} + H_u\dot{u} + H_x\dot{x})} \tag{6.22}$$

so that on the $(n+1)^{st}$ iteration the final time, which cannot be computed until $t = t_f^{(n)}$, the new final time $t_f^{(n+1)}$ is given by

$$t_f^{(n+1)} = t_f^{(n)} + \delta t_f^* \,. \tag{6.23}$$

If $\delta t_f^* > 0$ so that the new interval $[t_o, t_f^{(n+1)}]$ is larger than before, δu is set equal to zero on $(t_f^{(n)}, t_f^{(n+1)}]$. For any δt_f^*, δu is computed in the normal manner over the interval $[t_o, t_f^{(n)}]$. Some difficulties may arise if (6.22) specifies a very large δt_f since a constraint $\|\delta u\|$, which will reduce $\delta u(t_f)$ and $\delta x(t_f)$, does not change the other terms. Consequently, it may be necessary to restrict δt_f by an artifical bound if δt_f^* from (6.22) is very large.

Problems with control parameters, although formulated in a similar manner to the problems with continuous control functions (c.f., Chapter IV Sections A and C), must be solved in a quite different manner. The reason for the difference is that since the parameters are constants, they cannot be adjusted along the trajectory as functions of δx. This eliminates the usual feedback approach which has been used for the other problems considered earlier. Following Section C of Chapter IV, $\delta\alpha$ is chosen so that the cost functional

$$\hat{J} = -\delta\nu'(\delta\psi + \psi) + 1/2\,\delta x_f'\varphi_{xx}\delta x_f + \left[\int_{t_o}^{t_f} H_\alpha + \delta x' H_{x\alpha}\, d\sigma\right] \delta\alpha$$

$$+ 1/2 \int_{t_o}^{t_f} \delta x' H_{xx} \delta x \, d\sigma + 1/2\, \delta\alpha' \left[\int_{t_o}^{t_f} (H_{\alpha\alpha} + W)\, d\sigma\right] \delta\alpha \tag{6.24}$$

is minimized while satisfying the constraints

$$\delta\dot{x} = f_x \delta x + f_\alpha \delta\alpha \tag{6.25}$$

and

$$\delta x(0) = 0 \qquad \psi_x \delta x(t_f) = \delta\psi \ . \tag{6.26}$$

The solution for this problem is quite straightforward. First, equation (6.25) is solved m times with the $m \times 1$ control vector $\delta\alpha$ set equal to $(1, 0,\ldots,0)'$, $(0, 1, 0,\ldots,0)'$, etc. That is, the $n \times m$ matrix solution $X(t)$ is found for the equation

$$\dot{X}(t) = f_x X(t) + f_\alpha K, \quad X(0) = 0 \tag{6.27}$$

where $k_{ij} = \delta_{ij}$.

By linearity, any solution to (6.25) for a particular $\delta\alpha$ is

$$\delta x = X\delta\alpha \tag{6.28}$$

so that X is the sensitivity of the solution to changes in α. After eliminating δx from (6.24) with (6.28) and (6.26), the problem, which is now strictly algebraic, becomes one of finding the constant vector $\delta\alpha$ which minimizes the quadratic form

$$\hat{J} = -\delta\nu'(\psi + \delta\psi) + 1/2\ \delta\alpha' Q \delta\alpha + a'\delta\alpha \tag{6.29}$$

where

$$Q = X'(t_f)\varphi_{xx}X(t_f) + \int_{t_o}^{t_f} [X'(t)\ H_{xx}X(t) + H_{\alpha\alpha} + W]\ d\sigma$$

$$a = \int_{t_o}^{t_f} (H'_\alpha + H_{\alpha x}X(t))\ d\sigma \ . \tag{6.30}$$

At the same time $\delta\alpha$ must satisfy the linear equation

$$B\delta\alpha = \psi_x X(t_f)\ \delta\alpha = \delta\psi \ . \tag{6.31}$$

By the methods of Chapter III, the optimal $\delta\alpha$ may be computed in terms of the projection operator $P = [B'(BB')^{-1}B - I]$ which projects any $m \times 1$ vector onto the nullspace of B. The minimizing vector $\delta\hat{\alpha}$ is given by

$$\delta\hat{\alpha} = B'(BB')^{-1}\ \delta\psi + Py \tag{6.32}$$

where the $m \times 1$ vector y is the minimum norm solution to

$$P'QPy = -P'a \ .$$

The adjustment of the terminal constraint sensitivities remains to be found. With the interpretation of $\delta\nu$ as the sensitivity of the optimal cost to changes in the constraints, $-\delta\nu'$ is the coefficient of $\delta\psi$ in the second two terms of $\hat{J}$ from (6.29) with (6.32) substituted for $\delta\alpha$. This results in

$$\delta\nu = -(BB')^{-1}B(QPy + a) \tag{6.33}$$

which completes the set of equations necessary to optimize sequentially a set of control parameters α.

The last special problem to be discussed concerns optimization with respect to the points of discontinuity of $f(x, t_1, t_2,\ldots,t_k)$. This is probably the most important one of the special extensions discussed as it includes the very interesting bang-bang control problems by a transformation of variables. Following Section B of Chapter IV, the accessory problem requires the minimization of

$$\hat{J} = -\delta\nu'(\psi+\delta\psi) + 1/2\,\delta x_f'\varphi_{xx}\delta x_f + 1/2\int_{t_o}^{t_f}(\delta x' H_{xx}\delta x)\,d\sigma$$

$$+\sum_{i=1}^{k}\left\{ \mathcal{D}_i(H)\delta t_i + \mathcal{D}_i(H_x\delta x)\delta t_i + 1/2\lambda'\mathcal{D}_i(\dot{f})\delta t_i^2 + 1/2 W_i \delta t_i^2 \right\}. \quad (6.34)$$

The variation in the state, δx, satisfies the differential equation and the boundary conditions

$$\delta\dot{x} = f_x\delta x$$

$$\delta x(0) = 0$$

$$\psi_x\delta x(t_f) = \delta\psi \quad (6.35)$$

on the intervals $t \in [t_o, t_1), (t_1, t_2), \ldots, (t_k, t_f]$. δx is discontinuous at $t = t_i$. The amount of discontinuity is

$$\mathcal{D}_i(\delta x) = -\mathcal{D}_i(f)\,\delta t_i - \mathcal{D}(f_x\delta x)\delta t_i - 1/2\mathcal{D}(\dot{f})\delta t_i^2 \quad . \quad (6.36)$$

Since δx does not satisfy a differential equation (6.35) on the whole interval $[t_o, t_f]$, the former derivation for the sensitivity functions is no longer valid. The differential equation constraint (6.35) may be taken into account in the usual manner by appending the following identically zero term to (6.34),

$$C = \int_{t_o}^{t_1^-}\delta\lambda'(f_x\delta x - \delta\dot{x})\,d\sigma + \int_{t_1^+}^{t_2^-} + \ldots + \int_{t_k^+}^{t_f}\delta\lambda'(f_x\delta x - \delta\dot{x})\,d\sigma = 0. \quad (6.37)$$

The integration by parts involves no tricks since all of the "bad points" $t = t_i$ are not interior points in intervals of integration. A typical term results in

$$\int_{t_i^+}^{t_{i+1}^-} \delta\lambda'(f_x\delta x - \delta\dot{x})\, d\sigma = \int_{t_i^+}^{t_{i+1}^-} \delta x'(f_x'\delta\lambda + \delta\dot{\lambda})\, d\sigma - \delta x'\delta\lambda\Big|_{t_i^+}^{t_{i+1}^-} . \tag{6.38}$$

Summing terms,

$$C = \sum_{i=0}^{k} \int_{t_i^+}^{t_{i+1}^-} \delta x'(f_x'\delta\lambda + \delta\dot{\lambda})\, d\sigma - \delta x'\delta\lambda\Big|_{t_o}^{t_f} - \sum_{i=1}^{k} \mathscr{D}_i(\delta x'\delta\lambda) . \tag{6.39}$$

The last sum may be combined with (6.36) which gives the discontinuity in δx at $t = t_i$. A representative term becomes

$$\begin{aligned} \mathscr{D}_i(\delta x'\delta\lambda) &= \delta x'(t_i^+)\,\delta\lambda(t_i^+) - \delta x'(t_i^-)\,\delta\lambda(t_i^-) \\ &= \delta x'(t_i^-)\mathscr{D}_i[\delta z(t)] - \delta\lambda'(t_i^+)\mathscr{D}_i[f(x, u)]\,\delta t_i . \end{aligned} \tag{6.40}$$

Equations (6.39) and (6.40) may be combined with (6.34) to obtain

$$\begin{aligned} \hat{J} = &-\delta\nu'(\psi+\delta\psi) + 1/2\ \delta x_f'\varphi_{xx}\delta x_f - \delta x_f'\delta\lambda_f \\ &+ \int_{t_o}^{t_f} [1/2\ \delta x'H_{xx}\delta x + \delta x'(f_x'\delta\lambda + \delta\dot{\lambda})]\, d\sigma \end{aligned}$$

(continued)

$$- \sum_{i=1}^{k} \delta x'(t_i^-)\mathscr{D}_i[\delta\lambda(t) - H_x'\delta t_i] + \sum_{i=1}^{k} [\mathscr{D}_i(H) + \delta\lambda'(t_i)\mathscr{D}_i f(x, u)]\delta t_i$$

$$+ 1/2 \sum_{i=1}^{k} [W_i + \lambda'(t_i)\mathscr{D}_i(\dot{f}) - 2H_x(t_i^+)\mathscr{D}_i f(x, u)](\delta t_i)^2 \quad . \qquad (6.41)$$

Following the usual calculus of variations argument, the necessary condition for an extremum requires that $\delta\hat{J} = 0$. In taking the variation of $\hat{J}$, variations in δx_f, $\delta x(t_i^-)$, δt_i, and $\delta x(t)$ are written as $\delta^2 x_f$, $\delta^2 x(t_i^-)$, $\delta^2 t_i$, and $\delta^2 x(t)$, corresponding to second variations in the variables of the original problem, x_f, $x(t_i^-)$, t_i, and $x(t)$. The result is

$$\delta\hat{J} = \delta^2\varphi = \delta^2 x_f'(\varphi_{xx}\delta x_f - \delta\lambda_f) + \int_{t_o}^{t_f} \delta^2 x'(H_{xx}\delta x + f_x'\delta\lambda + \delta\dot{\lambda})\, d\sigma$$

$$- \sum_{i=1}^{k} \delta^2 x'(t_i^-)\mathscr{D}_i[\delta\lambda(t) - H_x'\delta t_i] + \sum_{i=1}^{k} [\mathscr{D}_i(H)$$

$$+ \delta\lambda'(t_i^+)\mathscr{D}_i f(x, u) + \delta x'(t_i^-)\mathscr{D}_i H_x' + \lambda'(t_i)\mathscr{D}_i(\dot{f})\delta t_i$$

$$+ W_i\delta t_i - 2H_x(t_i^+)\mathscr{D}_i f(x, u)\, \delta t_i]\delta^2 t_i \quad . \qquad (6.42)$$

If $\delta\hat{J} = 0$ for arbitrary variations in $x(t)$ and t_i, the coefficients of $\delta^2 x_f$, $\delta^2 x(t)$, $\delta^2 x(t_i^-)$, and $\delta^2 t_i$ must all vanish. This leads to the necessary conditions for the accessory problem. The adjoint variable $\delta\lambda(t)$ for the accessory problem is chosen to satisfy the differential equation

$$\delta\dot{\lambda} = - f'_x\delta\lambda - H_{xx}\delta x \tag{6.43}$$

except at the points t_i, $i = 1, 2, \ldots, k$. At each of the points t_i, stationarity with respect to $\delta x(t_i^-)$ requires $\delta\lambda(t)$ to be chosen so that the quantity $\delta\lambda(t) + H'_x\delta t_i$ is continuous or that $\delta\lambda(t)$ is possibly discontinuous according to

$$\mathscr{D}_i\delta\lambda(t) = \mathscr{D}_i H'_x\delta t_i \ . \tag{6.44}$$

Equations (6.43), (6.44) and the end condition

$$\delta\lambda(t_f) = \varphi_{xx}\delta x(t_f) \tag{6.45}$$

completely specify the accessory adjoint variable $\delta\lambda(t)$.

The remaining term in (6.42) is set equal to zero if

$$\begin{aligned} &- [W_i + \lambda'(t_i)\mathscr{D}_i(\dot{f}) - 2H_x(t_i^+)\mathscr{D}_i f(x, u)]\ \delta\hat{t}_i \\ &= + \delta x'(t_i^-)\mathscr{D}_i H'_x + \mathscr{D}_i H + \delta\lambda'(t_i^+)\mathscr{D}_i f(x, u) \end{aligned} \tag{6.46}$$

which specifies the optimal shifts in the switching times if the coefficient of $\delta\hat{t}_i$ is not zero. Equation (6.46) may be written in terms of $\delta\lambda(t_i^-)$ instead of $\delta\lambda(t_i^+)$ by the use of (6.44) to obtain

$$\begin{aligned} &\{-\lambda'(t_i)\mathscr{D}_i(\dot{f}) - W_i + [H_x(t_i^+) + H_x(t_i^-)]\mathscr{D}_i f(x, u)\}\ \delta\hat{t}_i \\ &= \mathscr{D}_i H + \delta\lambda'(t_i^-)\mathscr{D}_i f(x, u) + \delta x'(t_i^-)\mathscr{D}_i H'_x \ . \end{aligned} \tag{6.47}$$

In order to achieve the goal of a feedback control, $\delta\lambda$ must be eliminated from the expression for $\delta\hat{t}_i$. As before, a relationship

enabling $\delta\lambda$ to be found from δx is desired. Having motivated the method in the previous chapter, a strictly algebraic approach will now be used. A relation between δx and $\delta\lambda$ of the form

$$M'(t)\,\delta\lambda(t) = N'(t)\,\delta x(t) + b \tag{6.48}$$

is assumed to hold except at the points t_i. By differentiation of (6.48) and the substitution of (6.35) and (6.43), it may be shown that (6.48) will hold for all $t \neq t_i$; $t_o \leq t \leq t_f$ if M and N satisfy

$$\begin{pmatrix} \dot{M} \\ \dot{N} \end{pmatrix} = \begin{pmatrix} f_x & 0 \\ -H_{xx} & -f'_x \end{pmatrix} \begin{pmatrix} M \\ N \end{pmatrix} . \tag{6.49}$$

On the interval $(t_k, t_f]$, the previous theory applies so that the set of boundary conditions for M, N, and b in (6.10) are also appropriate here. They are

$$\begin{aligned} M(t_f) &= [B \quad 0] \\ N(t_f) &= [Q_3 B \quad \psi'_x] \\ b(t_f) &= - N'(t_f)[\psi_x \psi'_x]^{-1} \psi'_x \delta\psi \end{aligned} \tag{6.50}$$

with the definitions of B and Q_3 as given in Section A of this chapter.

Since the Euler-Lagrange equations are homogeneous, the differential equation for b is $\dot{b} = 0$. b is therefore constant over each of the intervals $[t_o, t_1), (t_1, t_2), \ldots, (t_k, t_f]$.

It is reasonable to expect M, N, and b to be discontinuous at $t = t_i$ since δx and $\delta\lambda$ are not continuous there. A relationship

between the possible discontinuities in M, N, and b and the discontinuities in δx and $\delta\lambda$, which may be obtained from (6.48), is

$$\mathfrak{D}_i[M'(t)]\,\delta\lambda(t_i^+) + M'(t_i^-)\mathfrak{D}_i[\delta\lambda(t)]$$

$$= \mathfrak{D}_i[N'(t)]\,\delta x(t_i^-) + N'(t_i^+)\,\mathfrak{D}_i[\delta x(t)] + \mathfrak{D}_i[b(t)]\ . \qquad (6.51)$$

The idea to be used in finding $\mathfrak{D}_i M$, $\mathfrak{D}_i N$, and $\mathfrak{D}_i b$ from (6.51) is similar to the method used to obtain the differential equations for M, N, and b in Chapter V. If, by suitable manipulations, (6.51) may be written in a form $A(t_i)\,\delta x(t_i^-) + B(t_i)\,\delta\lambda(t_i^+) + C(t_i) = 0$ with $A(t)$, $B(t)$, and $C(t)$ not depending on δx or $\delta\lambda$, then a sufficient condition for the equality to hold for arbitrary $\delta x(t_i^-)$ and $\delta\lambda(t_i^+)$ is that $A(t_i) = B(t_i) = C(t_i) = 0$.

The terms in (6.51) involving $\mathfrak{D}_i\delta\lambda$ and $\mathfrak{D}_i\delta x$ may be eliminated by substituting (6.44) and the first-order part of (6.36) to obtain

$$\mathfrak{D}_i M'(t)\,\delta\lambda(t_i^+) - \mathfrak{D}_i N'(t)\,\delta x(t_i^-) - \mathfrak{D}_i b(t)$$

$$= [-M'(t_i^-)\,\mathfrak{D}_i H_x' - N'(t_i^+)\,\mathfrak{D}_i f]\,\delta t_i\ . \qquad (6.52)$$

By picking W_i large enough, the coefficient of $\delta\hat{t}_i$ in (6.46) is nonzero so that $\delta\hat{t}_i$ may be found by dividing through by its coefficient. The substitution of $\delta\hat{t}_i$ obtained in this way into (6.52) and the subsequent collection of terms gives

$$\{\mathfrak{D}_i M'(t) + \alpha_i[M'(t_i^-)\,\mathfrak{D}_i H_x' - N'(t_i^+)\,\mathfrak{D}_i f]\,D_i f'\}\,\delta\lambda(t_i^+)$$

$$= \{\mathfrak{D}_i N'(t) - \alpha_i[M'(t_i^-)\,\mathfrak{D}_i H_x' - N'(t_i^+)\,\mathfrak{D}_i f]\,\mathfrak{D}_i H_x\}\,\delta x(t_i^-)$$

(continued)

$$+ \mathcal{D}_i b - \alpha_i \mathcal{D}_i H[M'(t_i^-)\, \mathcal{D}_i H_x' - N'(t_i^+)\, \mathcal{D}_i f] \tag{6.53}$$

where

$$1/\alpha_i = - W_i - \lambda'(t_i)\, \mathcal{D}_i(\dot{f}) + 2H_x(t_i^+)\, \mathcal{D}_i f \ . \tag{6.54}$$

A sufficient condition for (6.53) to hold for all $\delta\lambda(t_i^+)$ and $\delta x(t_i^-)$ is that each of the terms in braces is zero. This leads to the conditions for the discontinuity in N,

$$\mathcal{D}_i N = \alpha_i [M'(t_i^-)\, \mathcal{D}_i H_x' - N'(t_i^+)\, \mathcal{D}_i f]\, \mathcal{D}_i H_x \tag{6.55}$$

and M

$$\mathcal{D}_i M(t) = - \alpha_i \mathcal{D}_i f[\mathcal{D}_i H_x M(t_i^-) - \mathcal{D}_i f' N(t_i^+)] \tag{6.56}$$

and b

$$\mathcal{D}_i b = + \alpha_i [\mathcal{D}_i H_x M(t_i^-) - \mathcal{D}_i f' N(t_i^+)] \mathcal{D}_i H \ . \tag{6.57}$$

The last three relations, together with the differential equations (6.49) and the boundary conditions (6.50), make it possible to compute the quantities $M(t)$, $N(t)$, and $b(t)$ by backward integration. In the forward integration, the shift in the switching times t_i is computed at each point $t = t_i$ from (6.46). A feedback form of correction may be obtained from (6.47) if $\delta\lambda(t_i^-)$ is found in terms of $\delta x(t_i^-)$ by the use of equation (6.48). Then the optimal shift in the switching times becomes

$$\delta\hat{t}_i = \beta_i [\mathcal{D}_i H + b' M^{-1} \mathcal{D}_i f] + \beta_i [\mathcal{D}_i (f')(M')^{-1}(N') + \mathcal{D}_i H_x](x^{(n+1)} - x^{(n)})\Big|_{t=t_i^-} \tag{6.58}$$

where

$$1/\beta_i = [H_x(t_i^+) + H_x(t_i^-)]\Delta_i f - W_i - \lambda'(t_i)\Delta_i(\dot{f}) \quad .$$

If $\delta\hat{t}_i$ as computed from (6.58) turns out to be negative, indicating that the nominal switch is too late, the correct new trajectory could be computed by backing up to the point $t = t_i + \delta\hat{t}_i$ and restarting the integration. An easier scheme for computation would be to allow x to be discontinuous by the discontinuity in δx given in (6.36), which has an effect approximating the effect of the shifted switching time, independent of the sign of $\delta\hat{t}_i$. For the next iteration, the times could be changed according to

$$t_i^{(n+1)} \leftarrow t_i^{(n)} + \delta\hat{t}_i \quad . \tag{6.59}$$

The adjustment of the ν's may be carried out as before by integrating the accessory adjoint, (6.43), over a small interval $[t', t_f]$ after initializing with $\delta\lambda(t')$ as found from (6.48).

C. PROPERTIES OF THE SOLUTION

The computational method has been shown to construct a sequence of successively better controls. In this section, several of the properties taken on by the solution as the sequence of controls converges will be discovered.

By the convergence of the control sequence, it is implied that $\delta u \rightarrow 0$ or

$$\delta u = -(H_{uu} + W)^{-1}[H_{ux}\delta x + f_u'(M')^{-1}(N'\delta x + k)] \rightarrow 0 \tag{6.60}$$

where*

$$k = M'(t_f)\,\lambda(t_f) + b(t_f) \ . \tag{6.61}$$

The original non-feedback form of Eq. (6.60) is

$$\delta u = -(H_{uu} + W)^{-1}[H_{ux}\delta x + H_{u\lambda}\delta\lambda + H'_u] \tag{6.62}$$

which may be recovered from (6.66) by using the relations

$$k = M'(t)\,\lambda(t) + b(t) \tag{6.63}$$

and

$$M'(t)\,\delta\lambda(t) = N'(t)\,\delta x(t) + b(t) \ . \tag{6.64}$$

The gradient of the Hamiltonian, H_u, evaluated along the $(n+1)^{st}$ trajectory may be expressed as

$$H_u^{(n+1)} = H_u^{(n)} + H_{ux}^{(n)}\,[x^{(n+1)} - x^{(n)}]$$

$$+ H_{u\lambda}^{(n)}[\lambda^{(n+1)} - \lambda^{(n)}] + H_{uu}^{(n)}\,[u^{(n+1)} - u^{(n)}]$$

$$+ o(\|\delta x\|^2) + o(\|\delta\lambda\|^2) + o(\|\delta u\|^2)$$

so that if $\delta u \to 0$, $W \to 0$, and $H_{uu} \neq 0$ then (6.62) implies

$$H_u^{(n+1)} \to 0 \ . \tag{6.65}$$

*Equations (6.61) and (6.63) hold for the Mayer problem only. It is assumed that the problem has been put into the Mayer form.

If $\delta u \to 0$ then $\delta x \to 0$ also and therefore $\delta\psi = 0$. The solution will therefore satisfy the constraint $\psi[x(t_f)] = 0$.

It has been previously shown that the method continues to generate successively better controls until no further progress is made. By the foregoing, it may be concluded that when $\delta u \to 0$, the solution satisfies all of the necessary conditions given in the Minimum Principle since the only conditions not originally satisfied by the construction of the computational technique were $H_u = 0$ and $\psi[x(t_f)] = 0$.

Throughout this study the linear quadratic loss problem solved in Chapter V has been called the "accessory" problem. To be more exact, this problem should be perhaps called the "pseudo accessory" problem to distinguish it from the accessory problem discussed in texts on the Calculus of Variations. The distinction is that the accessory problem arises when considering second variations about an extremal and that the "pseudo accessory" problem is obtained by studying second variations about <u>any</u> nominal trajectory. Since the method gives a solution which approaches a solution to the necessary conditions, the pseudo accessory problem approaches the true accessory problem. The equations for the true accessory problem are obtained from the equations in Fig. 6.2 by setting $H_u = 0$, $\delta x_f = 0$, $W = 0$, and $\delta\psi = 0$. Since c, d, and $b(t_f)$ are now zero in Fig. 6.2, b satisfies a homogeneous linear differential equation with zero terminal conditions and is therefore identically zero. The resulting equations for the accessory problem are summarized in Fig. 6.3. In the remainder of this section, the nominal trajectory will be assumed to satisfy all of the necessary conditions so that the equations in Fig. 6.3 describe the corresponding

Definitions

$$F = f_x - f_u H_{uu}^{-1} H_{ux}$$

$$Q = f_u H_{uu}^{-1} f_u'$$

$$S = H_{xx} - H_{xu} H_{uu}^{-1} H_{ux}$$

b is any $n \times n - q$ full rank solution to

$$\psi_x B = 0$$

Case I

(Free Endpoint) $\delta\lambda = P\delta x$

$$\dot{P} = -F'P - PF - S + PQP$$

$$P(t_f) = \varphi_{xx}$$

$$\delta u = -H_{uu}^{-1}(H_{ux}\delta x + f_u' P\, x)$$

Case II

(Fixed Endpoint) $R\delta\lambda = \delta x$

$$\dot{R} = FR + RF' + RSR - Q$$

$$R(t_f) = 0$$

$$\delta u = -H_{uu}^{-1}(H_{ux}\delta x + f_u' R^{-1}\delta x)$$

Case III

(General) $M'\delta\lambda = N'\delta x$

$$\dot{M} = FM - QN$$

$$\dot{N} = -SM - F'N$$

$$M(t_f) = [B \quad 0]$$

$$N(t_f) = [\varphi_{xx} B \quad \psi_x']$$

$$\delta u = -H_{uu}^{-1}[H_{ux}\delta x + f_u'(M')^{-1} N'\delta x]$$

FIG. 6.3 SUMMARY OF RESULTS FOR THE ACCESSORY PROBLEM WHEN THE NOMINAL TRAJECTORY IS AN EXTREMAL

accessory problem. At convergence, the solution to the accessory problem has two very important uses which are now to be presented.

One of the disadvantages in the application of optimal control to real problems is that a complete knowledge of the system equations and the initial conditions is required in order to generate a numerical answer. If some of the variables in the problem description are slightly in error, the numerical control is no longer optimal. Therefore, several methods have been devised for on line correction of the control when it is applied so that the resultant control is improved. These methods attempt to generate a new extremal from the old extremal in the event that the prescribed control $u(t)$ causes the trajectory to drift off of the originally computed optimal trajector due to unpredicted errors in the system equations, unforeseen extremal disturbances, or initial conditions. The Lambda-Matrix control scheme used by Bryson and Denham [1961] and the method of Rosenbaum [1963] are examples of this type of control correction. The same idea is called Neighboring Extremal Control in the paper by Breakwell, Bryson, and Speyer [1963]. In the following, it will be shown that the Neighboring Extremal Control Law is obtained as an automatic byproduct of the computational method based on second variations without additional calculations.

Optimal paths, or extremals, are constructed so that the cost does not change to first order for small changes in the control $u(t)$ or the state $x(t)$. Therefore, optimization schemes in the neighborhood of an extremal must consider second-order terms. In the neighboring optimal control scheme δu is chosen to optimize the second-order terms in the expansion of the functional $\varphi - \nu'\psi$ while maintaining $\psi[x(t_f)] = 0$

to first order. This is precisely the way in which the control was chosen in the computational method. In fact, since the correction to the control u was found as a function of δx by eliminating $\delta\lambda$, the coefficients $c_1(t)$, $c_2(t),\ldots,c_n(t)$, $c_{n+1}(t)$, computed with each iteration, give the correct neighboring extremal control law as

$$u(t) = c_1(t)\, x_1(t) + \ldots + c_n(t)\, x_n(t) + c_{n+1}(t) \ .$$

This control is optimal along extremals and has an error of order higher than $\|x(t) - y(t)\|$ along a nonoptimal trajectory $y(t)$ which is in the neighborhood of an optimal trajectory $x(t)$.

The accessory problem solution may be used to obtain another useful result, testing the conjugate point condition for the solution. In an earlier chapter, the absence of conjugate points was given as one of the sufficient conditions guaranteeing that the extremal was actually a minimizing curve for the pseudo accessory problem. There are similar results for the nonlinear problem which are given in the following theorem.

Theorem 6.1

If there exists a pair of vectors $[x(t), \lambda(t)]$ which satisfies the necessary conditions given in (3.6) and (3.7) (Pontryagin's Minimum Principle) and

1. H_{uu} is nonsingular for all $t \in [t_o, t_f]$ (Strengthened Legendre Condition)
2. There is an optimal δu for the accessory problem with the boundary condition $\psi_x \delta x = a$ for arbitrary a (output controllability)

3. There are no conjugate points in $[t_o, t_f]$ then the trajectory $x(t)$ is optimal in the sense that it provides a weak relative extremum for the payoff $\varphi[x(t_f)]$ while satisfying the constraints $\psi[x(t_f)] = 0$.

For a proof of this theorem, stated in a different form, see Bliss [1949], Chapter IV.* Condition 2 replaces Bliss' assumption of normal extremals. Since the accessory problem is quadratic, it is its own accessory problem, so that Condition 3 of the theorem may be interpreted as pertaining to conjugate points for the accessory problem or for the original nonlinear problem.

Since the conditions of Theorem 6.1 are also necessary for the computational method based on second variations to converge on the interval in $[t_o, t_f]$ in the sense that $\delta u \to 0$, $\lambda(t_f) \to$ a finite value, and $\|W\| \to 0$, it may be concluded that the numerical solution must furnish a local extremum for the payoff $\varphi[x(t_f)]$ while satisfying the constraints $\psi[x(t_f)] = 0$.

D. SUGGESTIONS FOR CODING

Comments concerning the mechanics of programming are usually not found in the literature on computational methods probably either because the authors did not perform the actual programming or because subjects of this nature do not make interesting reading for a general audience.

*Bliss' conjugate system $U_{ik}(x)$, $V_{ik}(x)$ $(k = 1,\ldots,n)$ of solutions to the accessory equations corresponds to the matrices $M(t)$ and $N(t)$ in in this report.

This section is included because the author was the programmer and some of the ideas may save the prospective programmer a great deal of wasted effort before he discovers the same thing for himself.

Some of the initial programs were written in FORTRAN II for the IBM 1620 and 7090. Later programs were written in a special form of ALGOL for the 7090, called SUBALGOL*, which is a compiler language developed at Stanford University. For reference, a sample listing of a SUBALGOL program is included in Appendix B. The sample program was used to obtain some of the numerical results given in Section D of the next chapter. Due to the way in which the language is constructed, readers with no prior experience with SUBALGOL, who are familiar with another compiler language, should experience little difficulty in reading the program. The sample program is strictly ad hoc, written for the purpose of investigating some of the properties of the method in obtaining numerical examples for a specific example. Because of this, it is suggested that the reader write his own program, using the listing to answer occasional questions rather than as a model program.

The heart of the program is the integration of differential equations so that it is worthwhile to devote some careful thought to the selection of the method to be used. Since most available library routines do not make provisions for some of the options desirable in this program such as storing the variables at prescribed intervals, testing several

*This language was derived from the Burroughs Algebraic Compiler (BALGOL), originally developed for the Burroughs 220 machine. SUBALGOL is the mnemonic name for Stanford University's version of the Burroughs Algebraic Compiler.

possible conditions for possible exits at each integration step, integrating backwards without changing variables, and integrating equations which depend on functions stored in tabular form, it is tempting to write a special differential equation solver incorporating the desired special features. This procedure, which was followed in the numerical work reported in the next chapter, is not recommended without first seriously considering modifying, if necessary, existing package routines for differential equation solution available at most computation facilities. The final version of the program used for the test of the method, entitled ADDUMS in the listing, is actually reasonably standard except for the features of backwards integration (the initial value of the independent variable is larger than the final) and the provisions for keeping track of the running index on the stored variables, which is, although convenient and efficient, really not necessary. In fact, most of the special features needed may be included as a part of the subroutine which furnishes the derivative of the dependent variable (Procedures BVDP and FVDP in the listing) since these programs must be written anyway. The type of numerical integration method used, based on the previous reasoning, is probably best determined by what is available. Procedure ADDUMS uses a fourth-order Runge-Kutta method for starting a fourth-order Adams-Bashforth predictor-corrector method as given in Hamming [1962]. Although a program using a fourth-order Runge-Kutta method, or any of the similar methods as Gill or Kutta-Merson, would have produced a somewhat simplified program and an ability for easily varying the integration step size, these methods were rejected in favor of the predictor-corrector method which requires two derivative evaluations at each integration step as compared with four derivative

evaluations for the R-K type methods. Primarily due to a desire for a simplified tabular function storage and interpolation scheme as discussed in the next paragraph, the integration step size was selected and fixed over predetermined intervals. As a check on the accuracy, a warning flag is printed by the integration routine if the relative error of the integration is too large.

As described in Section A of this chapter, both the forward and backward integration need variables, as time functions, which have been computed on previous integrations. Some means must then be provided for storing the functions at selected sample points and reconstructing the time functions from the stored values as required. The use of a fixed integration step size and storage grid helps to simplify the programming which may outweigh the fact that a variable integration step size and nonuniformly spaced sample points could save time and memory. For this program, both of these methods were discarded in favor of a fixed integration step size and storage of the variables at every integration step. If the memory is available, it is senseless to develop a more complicated storage-interpolation routine which will waste both running and programming time to conserve unrequired memory. If a fixed step size is unreasonable, interpolation may still be avoided by continuing to store at each integration step and using the same sequence of step size changes for each integration. In this way the storage points are held fixed. This method was successfully applied in reducing the integration step size over the final part of the trajectory in order to reduce the numerical errors in the terminal constraints. Since no storage shortage difficulties were experienced in programming the examples on the 7090, many extra unnecessary time functions were stored for convenience in

outputting the results for plotting. A considerable reduction in the total amount of memory used for data could have been achieved by outputting the results as they were computed, thus eliminating the need for much of the temporary storage.

The calculation of the inner products in the determination of the feedback coefficients was made with the aid of the program IPD18, a double precision routine coded originally in FAP. Furthermore, an iterative method for minimizing the sum of the squares of the residual errors was used in the required linear equation solution. These features were incorporated in some of the early programs in order to help to track down some small numerical errors. By the use of an open-loop control over the last part of the trajectory, the requirement for very accurate numerical linear equation solutions is not so important so that the use of double precision and iterative solution improvement may be replaced by a less sophisticated technique.

The evaluation of each run, step ④ in Fig. 6.1, is detailed in flow chart form in Fig. 6.4. To minimize the effects of computing inaccuracies or noise, both ϕ and ψ are modified before the tests are made. Tests of ϕ may be made only on the first few significant bits by first setting the remaining significant bits to zero. Since the desired value of ψ is zero, ψ is set equal to zero if it is below a desired error bound.

A final comment concerns the step size adjustment in steps ⑥ and ⑦. The theory specifies that if W is large enough, the iteration will be successful and that $W \to 0$ as the method converges to a solution for which $H_{uu} \neq 0$. In practice W is replaced by αW, $\alpha > 0$, where

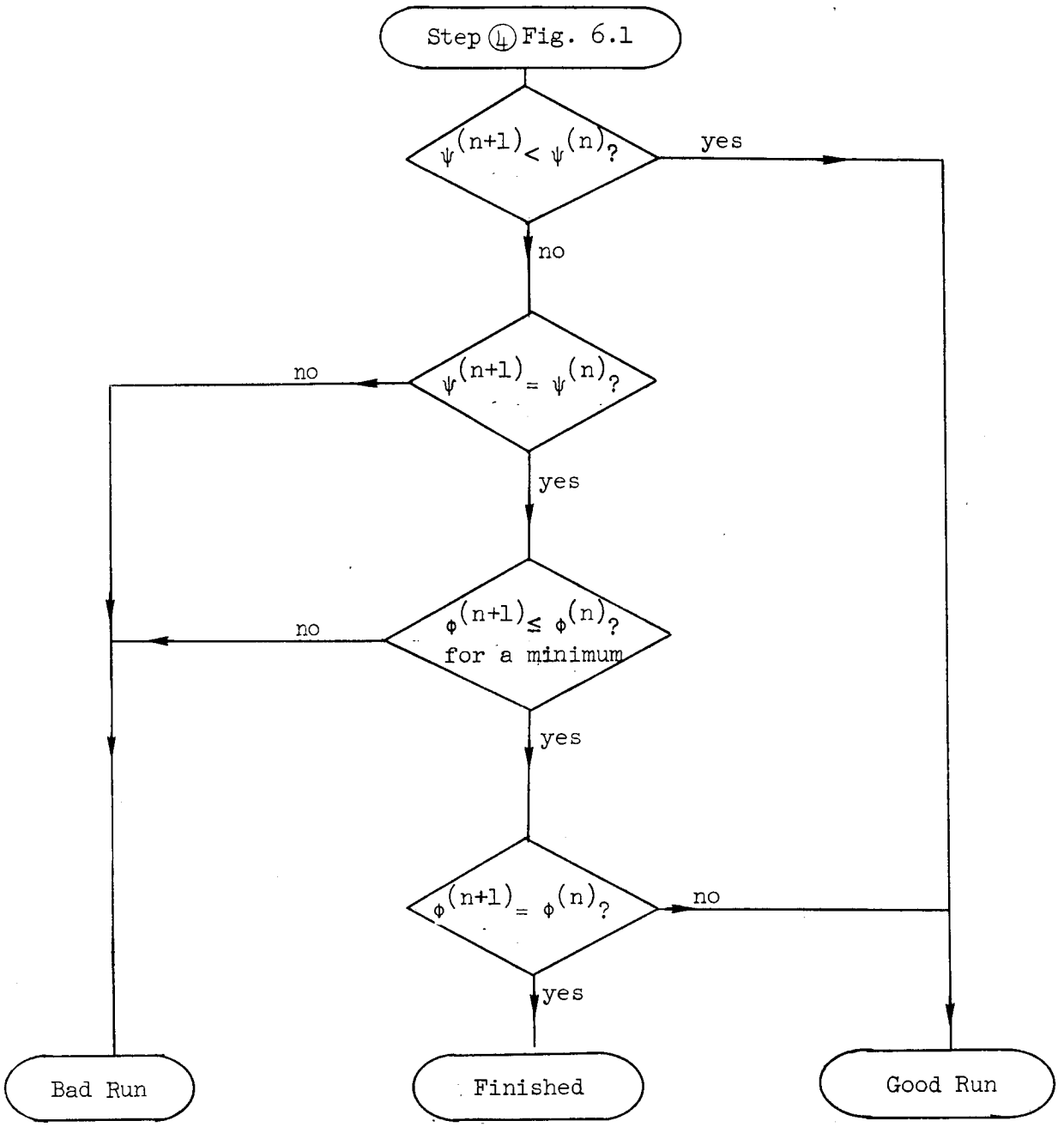

FIG. 6.4 DETAIL OF THE RUN EVALUATION

$\alpha = \alpha_g < 1$ for a successful run, step ⑥, and $\alpha = \alpha_b > 1$ for a bad run, step ⑦. In the numerical examples, the experimentally determined values $\alpha_b = 10$ and $\alpha_g = 0.5$ were found to produce a fairly efficient scheme for adjusting W.

VII. NUMERICAL EXAMPLES

In order to evaluate the efficiency of the proposed computational method experimentally, several numerical examples are presented in this chapter. It should be emphasized that the actual machine computation is an essential part of this research. Although it may be possible to prove analytically that a method converges to a solution, a machine solution may not be feasible due to the numerical inaccuracies involved. The experimental results presented here give a demonstration that the method works in actual practice, at least for the examples chosen.

The choice of problems has been made to illustrate the various special cases previously discussed. The first example, a linear plant with a quadratic loss function and free-end conditions, compares the one-step convergence of the second-order method to the relatively slow convergence of a usual first-order gradient-type method. An example with a complete specification of the terminal states for a nonlinear plant is then given to show the special technique developed for problems with fixed-end conditions. An example of a nonlinear plant with free-end conditions and a quadratic loss function is presented to again compare the second and first order techniques on a simple nonlinear problem with no analytic solution. The last example presented illustrates the method as applied to a problem with partially specified terminal states. This final example represents the most general type of boundary condition and the corresponding method developed in the chapter on the solution of two-point boundary value problems is applied.

In an effort to improve the readability of this chapter, some of the program details have been summarized in Appendix C for reference.

A. LINEAR PLANT QUADRATIC LOSS EXAMPLE

The first example to be studied is a driven harmonic oscillator described by the following set of linear differential equations

$$\dot{x}_1 = x_2$$
$$\dot{x}_2 = - x_1 + u \tag{7.1}$$

The cost function is the integral of the sum of the squares of the states and the control given by

$$J = 1/2 \int_0^{10} (x_1^2 + x_2^2 + u^2)\, d\sigma \tag{7.2}$$

The initial conditions are taken as $x_1(0) = 1$, $x_2(0) = 0$ and the final state is unspecified.

This problem was solved by the usual method of steepest descent with the program titled LQL and with the method based on second variations in program 2MV. Both methods require reverse time solutions of the adjoint equations

$$\dot{\lambda}_1 = \lambda_2 - x_1$$
$$\dot{\lambda}_2 = - \lambda_1 - x_2$$
$$\lambda_1(10) = \lambda_2(10) = 0 \tag{7.3}$$

Program 2MV also required solutions to the additional set of equations

$$\dot{p}_{11} = 2p_{12} + p_{12}^2 - 1$$

$$\dot{p}_{12} = p_{22} - p_{11} + p_{12}p_{22}$$

$$\dot{p}_{22} = - p_{12} - 1 + p_{22}^2$$

$$\dot{b}_1 = b_2 + p_{12}(b_2 + u + \lambda_2)$$

$$\dot{b}_2 = - b_1 + p_{22}(b_2 + u + \lambda_2)$$

$$p_{11}(10) = p_{12}(10) = p_{22}(10) = b_1(10) = b_2(10) = 0 \qquad (7.4)$$

In (7.4) the p's are the components of the symmetric P matrix and the b's are the components of the b vector.

The algorithm for updating the control in this problem in LQL is

$$u^{(n+1)} = u^{(n)} - \epsilon H_u = u^{(n)} - \epsilon (\lambda_2 + u^{(n)}) \qquad (7.5)$$

In order to give the best possible advantage to the program using the usual steepest descent approach, LQL, the step size ϵ was optimized at each step. The exact step size is determined at each point for the present problem. The step size optimization routine involves two extra integrations of the state equations at each step and results in an additional cost reduction which probably does not justify use in general programs. However, its use here eliminated all guessing from the method and possible unfair comparisons due to poor guesses of the step size.

In the program using second variations, 2MV, the control is updated by

$$u^{(n+1)} = u^{(n)} - (H_{uu} + W)^{-1}(H'_u + H_{ux}\delta x + f'_u P\delta x + f'_u b)$$

$$= -\lambda_2 - b_2 + p_{12}(x_1^{(n)} - x_1^{(n+1)}) + p_{22}(x_2^{(n)} - x_2^{(n+1)}) \quad (7.6)$$

The final optimal trajectories obtained from the second variations program are shown in Fig. 7.1 where the state variables x_1 and x_2, the cost J, and the control u are all plotted as functions of time. As expected, the second variations program coverged in one step.

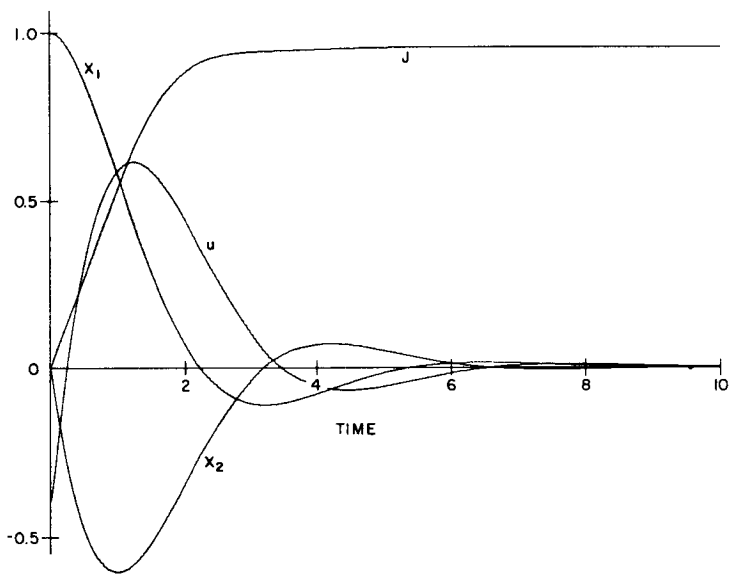

FIG. 7.1. OPTIMAL TRAJECTORIES FOR $1/(s^2 + 1)$ PLANT WITH QUADRATIC LOSS $Q_1 = I$, $Q_2 = 1$, $Q_3 = 0$ AND FREE-END CONDITIONS

The results of the steepest descent program are shown in Fig. 7.2. Starting with $u = 0$, 14 successive iterations on the control are shown. At the end of the 14th iteration, the cost was 0.95667 as compared to the optimal cost of 0.95613.

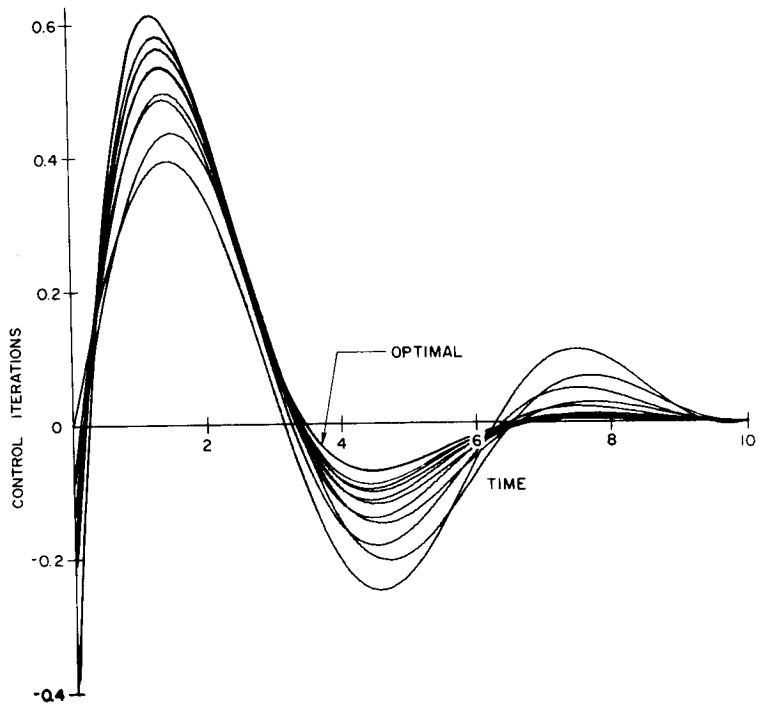

FIG. 7.2. SUCCESSIVE CONTROL ITERATIONS USING STEEPEST DESCENT, EXAMPLE A

The advantage of the second-order method is clear not only from the total number of iterations required for this problem, but also from the total time for computation. The time per iteration is not quite doubled by the second-order method.

This problem also illustrates some of the difficulties associated with the indirect method. Consider the adjoint variables shown in Fig. 7.3. Since the final adjoints are required to be zero, the quantities to be determined are the final state variables. From the plots, the optimal final states are picked near zero so that both the states and adjoints remain near zero for the interval between 10 and 5, and then rise to fairly large values in the remaining interval. From personal experience, this problem is almost impossible to work by the indirect

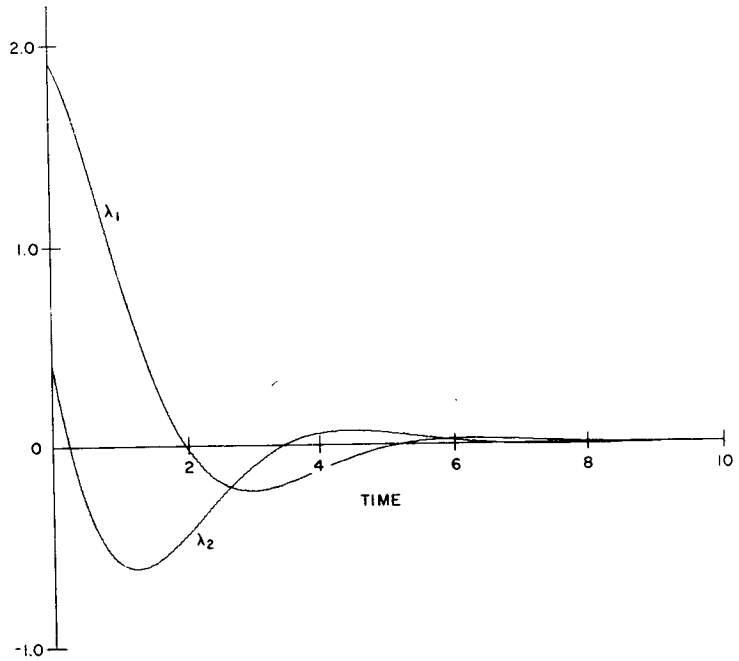

FIG. 7.3. ADJOINT VARIABLES, EXAMPLE A.

method (i.e., adjusting the final states) on an analog computer due to the large sensitivity of the initial states to changes in the final states which are near zero. However, the problem with $t_f = 5$ is reasonably easy. This is due to the exponential growth of the sensitivity with t_f for this problem. Most of the successful examples worked by the indirect method either have small values of the final time or have lightly damped plants. Both of these situations lead to reasonable sensitivities so that a solution is feasible.

The final set of curves given for this example, Fig. 7.4, shows the solution to the matrix Riccati equation. The optimal control for this problem is given in feedback form by $u = - p_{12}x_1 - p_{22}x_2$ so that this plot also shows the magnitude of the optimal feedback gains. For this example, the feedback control is a global optimal. That is, this feedback control law is optimal for this problem for any initial state.

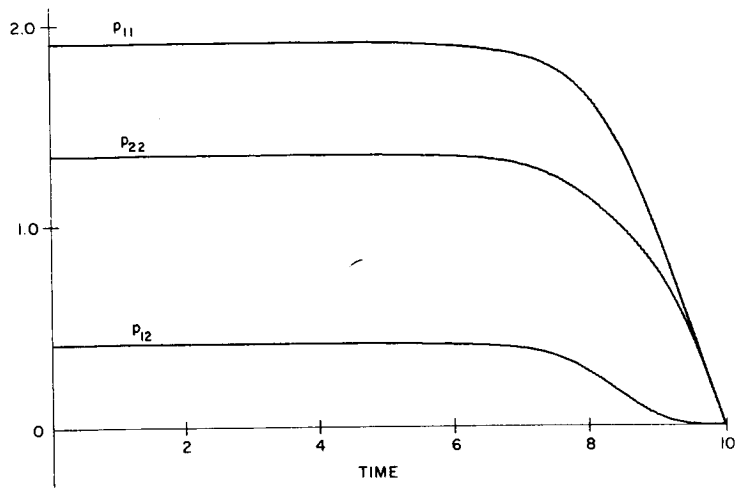

FIG. 7.4. SOLUTION TO RICCATI EQUATION FOR EXAMPLE A. The Optimal Feedback Control $u = - p_{12}x_1 - p_{22}x_2$

B. THE BRACHISTOCHRONE

The classical brachistochrone problem was chosen to illustrate a nonlinear problem with fixed-end points. This problem has the advantage of an analytic solution for direct comparison of results. Jazwinski [1964] has reported that the ordinary gradient method has very slow convergence for this problem. It seemed reasonable to see if the second-order technique could be employed to speed convergence.

Starting at the point $(0, 0)$, a particle slides down a frictionless wire under the influence of gravity until it reaches the point (ξ_f, η_f).

At the point $(0, 0)$ the particle is assumed to have the velocity obtained by a free-fall one unit distance or $\sqrt{2g}$. The problem is to find the shape of the guiding wire which minimizes the time of transition.

The velocity of the particle is

$$\frac{ds}{d\tau} = \sqrt{1 + \eta'^2} \quad \frac{d\xi}{d\tau} \tag{7.7}$$

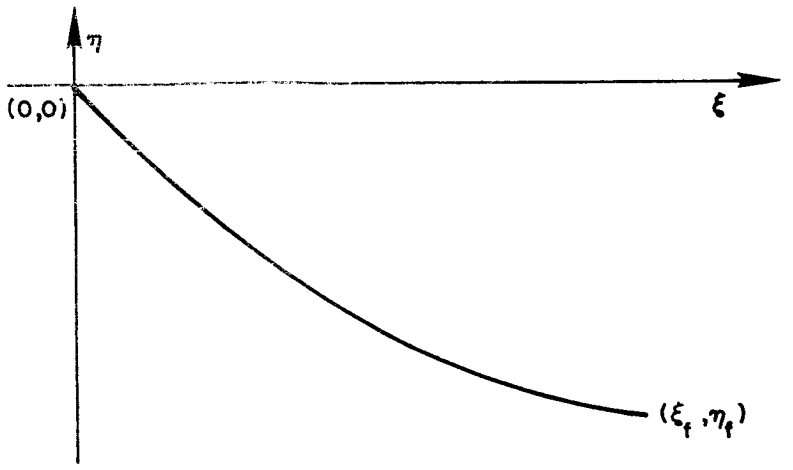

FIG. 7.5. BRACHISTOCHRONE, EXAMPLE B

where η' denotes the derivative of η with respect to ξ. The transition time is

$$T = \int_0^{\xi_f} \frac{\sqrt{1 + \eta'^2}}{v} \, d\xi$$

$$= \frac{1}{\sqrt{2g}} \int_0^{\xi_f} \frac{\sqrt{1 + \eta'^2}}{\sqrt{1 - \eta}} \, d\xi \qquad (7.8)$$

This problem may be expressed in control problem notation by identifying $-\eta'$ with u and $-\eta$ with the state x. With these definitions the reformulated problem has a cost function to be extremized given by

$$J = \int_0^{\alpha_f} \frac{\sqrt{1 + u^2}}{\sqrt{1 + x}} \, d\alpha \qquad (7.9)$$

The independent variable has been denoted by α instead of t, as is the usual convention, in order to avoid confusion with the time variable τ in the original problem. The state equation is

$$\dot{x} = u \tag{7.10}$$

with the boundary conditions

$$x(0) = 0 \qquad x(\alpha_f) = x_f \tag{7.11}$$

The Hamiltonian for this problem is given by

$$H = \frac{\sqrt{1 + u^2}}{\sqrt{1 + x}} + \lambda u \tag{7.12}$$

Along an extremal, the optimal control u^* minimizes H. Therefore

$$H_u^* = \frac{u^*}{\sqrt{1 + x}\ \sqrt{1 + u^{*2}}} + \lambda = 0 \tag{7.13}$$

Since the Hamiltonian does not contain α explicitly, it is a constant of motion along extremals. Substituting (7.10) and (7.13) into (7.12) yields after manipulation,

$$(1 + x)(1 + \dot{x}^2) = c \tag{7.14}$$

where c is a constant to be determined by the boundary conditions.

The set of solutions to this differential equation may be written in parametric form with parameter v as

$$\xi(v) = r(1 - \cos(v)) + 1$$

$$\eta(v) = r(v - \sin(v)) + k \tag{7.15}$$

which describes a family of cycloids. The former constant c has been absorbed in the new constants r and k which are picked to satisfy the boundary conditions (7.11). The initial and final values of the parameter v are also chosen so that the boundary conditions are satisfied. This leads to a set of four simultaneous transcendental equations in the unknowns v_o, v_1, r, and k.

$$0 = r(1 - \cos(v_o)) + 1$$

$$0 = r(v_o - \sin(v_o)) + k$$

$$\xi_f = r(1 - \cos(v_1)) + 1$$

$$\eta_f = r(v_1 - \sin(v_1)) + k \qquad (7.16)$$

In order to solve (7.16), a numerical technique must be used. An IBM 1620 program was written to carry out a solution by a form of Newton's method. The solution for $\xi_f = 1.0$ and $\eta_f = -0.5$ is $v_o = -1.8087562$, $v_1 = -2.5936165$, $r = -0.8092445$, and $k = -0.6772854$.

It may be easily shown that the minimum transit time is given by

$$T = \sqrt{-2r}\,(v_1 - v_o)/\sqrt{2g} \qquad (7.17)$$

For this particular terminal condition, the minimum time is computed as

$$T = 0.99849827/\sqrt{2g}\ .$$

The optimal trajectory is now completely specified by the constants computed above and is given in parametric form in (7.15). However, for comparison with the trajectories generated by the second variations, it

is convenient to have the value of η for a set of evenly spaced values of ξ. The set of corresponding values of η was found by another 1620 program using iteration on the parametric equations.

The preliminary calculations to set up the direct method based on second variations begins by computing the required partial derivatives of the Hamiltonian.

$$H_u = \lambda + \frac{u}{\sqrt{1 + x}\ \sqrt{1 + u^2}}$$

$$H_x = \frac{-\sqrt{1 + u^2}}{2(1 + x)^{3/2}}$$

$$H_{uu} = \frac{1}{\sqrt{1 + x}\ (1 + u^2)^{3/2}}$$

$$H_{ux} = \frac{-u}{2\sqrt{1 + u^2}\ (1 + x)^{3/2}}$$

$$H_{xx} = \frac{3\sqrt{1 + u^2}}{4(1 + x)^{5/2}} \qquad (7.18)$$

The adjoint equation is

$$\dot{\lambda} = -H_x = \frac{c}{2d^3} \qquad (7.19)$$

where c and d are defined by

$$c = \sqrt{1 + u^2}$$

$$d = \sqrt{1 + x} \qquad (7.20)$$

The remaining equations for R and b, as defined in Chapter V, are readily obtained by substituting (7.18) into the defining equations

$$\dot{R} = [f_x - f_u[H_{uu} + W]^{-1}H_{ux}]\, R + R[f'_x - H_{xu}[H_{uu} + W]^{-1}f'_u]$$

$$+ R[H_{xx} - H_{xu}[H_{uu} + W]^{-1}H_{ux}]\, R - f_u[H_{uu} + W]^{-1}f'_u$$

$$\dot{b} = (f_x - f_u[H_{uu} + W]^{-1}H_{ux} + R[H_{xx} - H_{xu}[H_{uu} + W]^{-1}H_{ux}])\, b$$

$$- f_u[H_{uu} + W]^{-1}H'_u - RH_{xu}[H_{uu} + W]^{-1}H'_u \qquad (7.21)$$

The substitution and simplification for this example give the following equations for the scalars R and b,

$$\dot{R} = \left[\frac{c}{4d^5}\left(3 - \frac{u^2}{1+Wdc^3}\right)\right] R^2 - \frac{uc^2}{d^2(1+Wdc^3)} - \frac{dc^3}{(1+Wdc^3)}$$

$$\dot{b} = \left[\frac{c}{4d^5}\left(3 - \frac{u^2}{1+Wdc^3}\right)\right] Rb - \frac{uc^2}{2d^2(1+Wdc^3)}\, b - \left(\lambda + \frac{u}{cd}\right)$$

$$\cdot \left(\frac{c^3d - R\,\frac{uc^2}{2d^2}}{(1+Wdc^3)}\right) \qquad (7.22)$$

The boundary conditions are

$$R(\alpha_f) = 0$$

$$b(\alpha_f) = \Delta x_f \text{ desired} \qquad = -x_f + 0.5 \text{ nominally} . \qquad (7.23)$$

The control for the $(n+1)^{st}$ iteration is given by

$$u^{(n+1)} = u^{(n)} - [H_{uu} + W]^{-1}[H_u + H_{ux}(x^{(n+1)} - x^{(n)})$$

$$+ f_u' R^{-1}(x^{(n+1)} - x^{(n)} - b)] \quad . \tag{7.24}$$

Substituting the expressions for this example results in the equation for the control

$$u^{(n+1)} = u^{(n)} - \frac{c^3 d}{1+Wc^3 d}\left[\lambda + \frac{u}{cd} - \left(\frac{u}{2cd^3} + \frac{1}{R}\right) x^{(n)} - \frac{b}{R}\right]$$

$$- \frac{c^3 d}{1+Wc^3 d}\left(\frac{u}{2cd^3} + \frac{1}{R}\right) x^{(n+1)} \tag{7.25}$$

The terminal boundary condition for λ is initially assigned an arbitrary value and then updated at each iteration by solving the equation

$$x^{(n+1)}(t) - x^{(n)}(t) - b(t) = R(t)[\lambda^{(n+1)}(t) - \lambda^{(n)}(t)] \tag{7.26}$$

at the final time for $\lambda^{(n+1)}$. However, $R = 0$ at the end point. The method used in the program involved solving for $\delta\lambda = \lambda^{(n+1)} - \lambda^{(n)}$ at several points near $t = t_f$ and then extrapolating the result to the end by fitting a polynomial through the computed points.

The machine results are shown in Fig. 7.6 which is a plot of the iterations on the trajectory. The initial guess was $u = 0$ which corresponds to a horizontal path. The first iteration reduced the cost and met the end conditions to within machine accuracy. The high degree

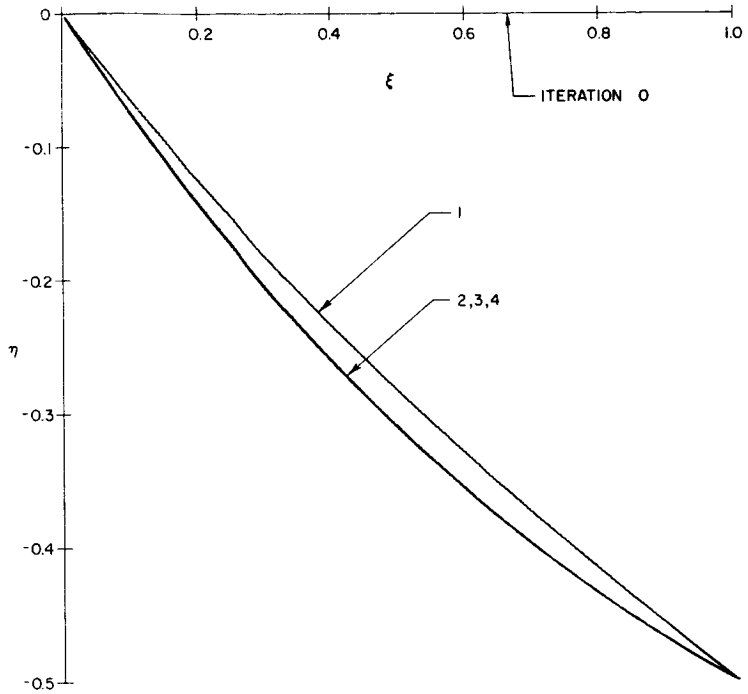

FIG. 7.6. TRAJECTORY ITERATIONS BY SECOND VARIATIONS FOR THE BRACHISTOCHRONE PROBLEM, EXAMPLE B

of success, which may seem surprising at first glance, may be attributed to two major causes. Theoretically, the accuracy is to be expected since the corrections are in fact exact for errors in a linear terminal constraint with a linear state equation. However, one might suspect this will not be the case in practice due to integration errors. These errors are compensated for by the feedback control which helps to force the errors in the terminal constraint to zero.

The method converged to within the accuracy of the numerical integration in only two steps. The plot shows that further iterations coincide with the second. The cost continued to decrease slightly after the second iteration, with variations in the eighth significant figure only.

As originally noted, this problem was chosen because of the poor convergence of the normal gradient method as reported by Jazwinski [1964]. The version of the problem worked here is due to McReynolds [1966]. The difference in the problem worked by Jazwinski and McReynolds is only in the numerical value of the terminal conditions. Jazwinski used $\xi_f = 5$, $\eta_f = -7$ and McReynolds used $\xi_f = 1$, $\eta_f = -0.5$. A quick check revealed that the change in terminal conditionsdid not change the convergence rate with the method based on second variations. Sinnott [1966] recently checked the problem with the gradient method and found it to be quite effective, converging in 3 or 4 steps to an acceptable answer for both choices of terminal conditions. This does not agree with the work of Jazwinski, who reported that his program terminated after 13 iterations and that the resulting trajectory did not satisfy the Euler equations well.

C. QUADRATIC LOSS VAN DER POL WITH FREE ENDPOINT

The problem chosen for this section is found on pages 267-270 of C. W. Merriam's book [1964] on optimization techniques. In discussing this problem, Merriam states for a particular control initialization that the application of "... the method based on second variations results in complete failure." The difficulty encountered here is due to the existence of a conjugate point in the accessory problem. The application of the theory developed in Chapters V and VI to circumvent these difficulties is illustrated in this numerical example.

The driven second-order nonlinear oscillator studied by Van Der Pol may be written in state space form as

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -x_1 + a(1 - x_1^2)\, x_2 + u \tag{7.27}$$

where the driving function $u(t)$ has been added as a control. The parameter a, which determines the degree of nonlinear behavior of the solutions, is taken as 1. This causes the free oscillations to be a rough sawtooth waveform. The initial conditions given are $x_1(0) = 1$ and $x_2(0) = 0$, which is a point inside the stable limit cycle.

The cost function to be minimized in this problem is

$$J = 1/2 \int_0^5 (x_1^2 + x_2^2 + u^2)\, dt \tag{7.28}$$

and the end condition is left free.

The first example is similar to the present one, in fact, the linear problem is a linearization of the nonlinear problem about the point $x_1 = 0$, $x_2 = 0$.

The first step in setting up the iterative technique is to define the Hamiltonian H as

$$H = \lambda_1 x_2 - \lambda_2 x_1 + \lambda_2(1 - x_1^2)\, x_2 + \lambda_2 u + 1/2(x_1^2 + x_2^2 + u^2) \tag{7.29}$$

As before, the required partials of H are evaluated and substituted into the equations necessary. The program used to generate the steepest descent solutions, titled SDVP, and the program based on second variations, titled 2VVP, both required solutions to the adjoint equations

$$\dot{\lambda} = -H_x' = -f_x'\lambda$$

which become

$$\dot{\lambda}_1 = (1 + 2x_1x_2)\,\lambda_2 - x_1$$

$$\dot{\lambda}_2 = -\lambda_1 + (x_1^2 - 1)\,\lambda_2 - x_2 \,. \tag{7.30}$$

Program 2VVP also required the $n \times n$ symmetric matrix P which satisfies

$$\dot{P} = -f_x'P - Pf_x - H_{xx} + Pf_u(H_{uu} + W)^{-1}f_u'P$$

where W determines the constraint on the control space step size. For this problem the components of P solve the following set of scalar differential equations

$$\dot{p}_{11} = 2(1 + 2x_1x_2)\,p_{12} + 2\lambda_2x_2 - 1 + \frac{p_{12}^2}{1+W}$$

$$\dot{p}_{12} = -p_{11} + (x_1^2 - 1)\,p_{12} + (1 + 2x_1x_2)\,p_{22} + 2\lambda_2x_1 + \frac{p_{12}p_{22}}{1+W}$$

$$\dot{p}_{22} = -2p_{12} + 2(x_1^2 - 1)\,p_{22} - 1 + \frac{p_{22}^2}{1+W} \,. \tag{7.31}$$

The additional n vector b satisfies

$$\dot{b} = -f_x'b + Pf_u(H_{uu} + W)^{-1}f_u'b + f_u(H_{uu} + W)^{-1}H_u \,.$$

For this problem the equations for the components of b are

$$\dot{b}_1 = (1 + 2x_1x_2)\, b_2 + p_{12}(b_2 + \lambda_2 + u)/(1 + W)$$

$$\dot{b}_2 = -b_1 + (x_1^2 - 1)\, b_2 + p_{22}(b_2 + \lambda_2 + u)/(1 + W) \ . \qquad (7.32)$$

Since the end conditions are not specified for the state variables in this problem, the terminal adjoint variables are zero for both programs. For the same reason the final b variables are also zero. The final P matrix is zero because there is no terminal cost function. The total set of boundary conditions at the terminal time is

$$b(t_f) = 0,\ P(t_f) = 0,\ \lambda(t_f) = 0 \ . \qquad (7.33)$$

In SDVP, the steepest descent algorithm for updating the control is

$$u^{(n+1)} = u^{(n)} - \epsilon\, H_u = u^{(n)} - \epsilon(\lambda_2 + u^{(n)}) \ . \qquad (7.34)$$

The program SDVP was initialized with two different starting values for the control function $u(t) = 0$ and $u(t) = 1$ in order to investigate the effect on the convergence. No particular difficulties were encountered with either guess. However, the $u = 0$ guess produced a lower cost after 18 iterations, although the cost on the first iteration was higher than for $u = 1$. For a comparison, the successive iterations on the control function are plotted in Fig. 7.7 for $u^{(o)} = 0$ and in Fig. 7.9 for $u^{(o)} = 1$. After 18 iterations, the costs were 1.450 and 1.565 for the runs initialized with $u^{(o)} = 0$ and $u^{(o)} = 1$ respectively. These figures are to be compared with the optimal cost of 1.433508 as obtained by second variations.

Program 2VVP was also initialized with several starting control functions. Since the change in the shape of the control function is

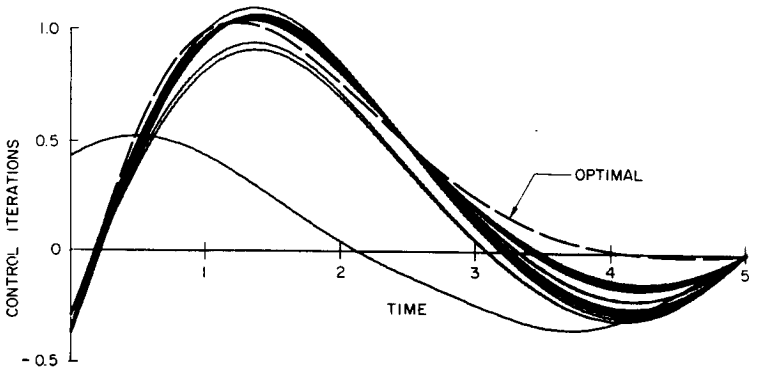

FIG. 7.7. CONTROL ITERATIONS USING STEEPEST DESCENT INITIALIZED WITH $u(t) = 0$ FOR THE VAN DER POL PROBLEM, EXAMPLE C

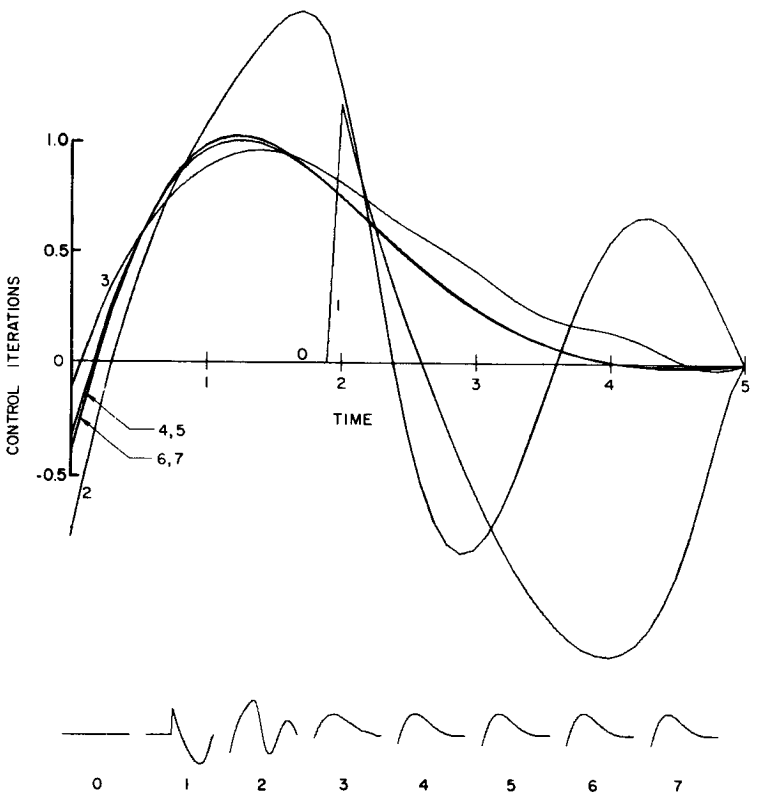

FIG. 7.8. CONTROL ITERATIONS USING SECOND VARIATIONS INITIALIZED WITH $u(t) = 0$ FOR THE VAN DER POL PROBLEM, EXAMPLE C (The sequence of small numbered plots may be used to help distinguish each iteration in the larger plot.)

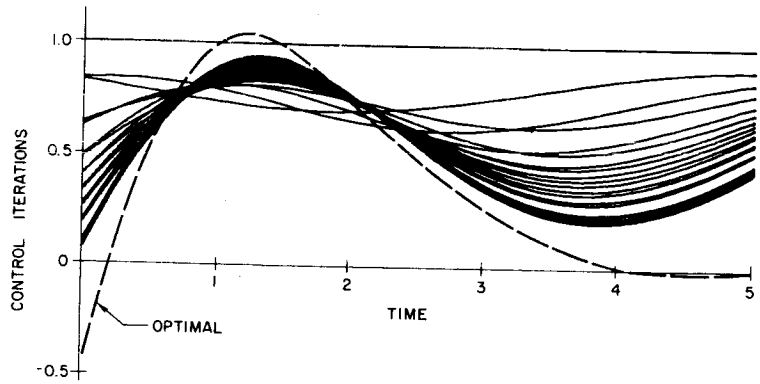

FIG. 7.9. CONTROL ITERATIONS USING STEEPEST DESCENT INITIALIZED WITH $u(t) = 1$ FOR THE VAN DER POL PROBLEM, EXAMPLE C

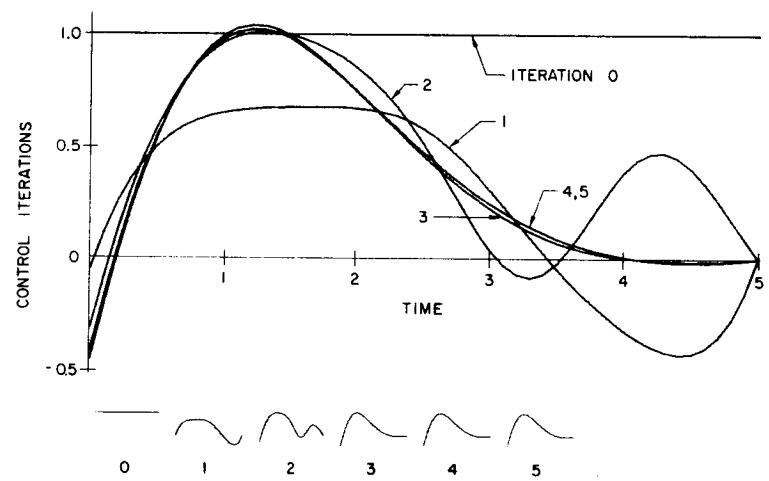

FIG. 7.10. CONTROL ITERATIONS USING SECOND VARIATIONS INITIALIZED WITH $u(t) = 1$ FOR THE VAN DER POL PROBLEM, EXAMPLE C

quite large from one iteration to the next, some additional information is helpful to distinguish the various curves. The iterations on u for $u^{(o)} = 0$ are shown in Fig. 7.8 and for $u^{(o)} = 1$ in Fig. 7.10. At the bottom of each figure, a sequence of small numbered plots shows the general trend of each iteration. These small figures may be used to help trace out each corresponding curve in the large plot which shows all iterations superimposed.

The first striking difference between the iterations in the steepest descent and second variations is in the apparently large steps taken with 2VVP. Recall the definition of "close" functions required that the norm of the difference given by

$$\|\delta u\| = \int_0^5 (u^{(n)}(t) - u^{(n+1)}(t))^2 \, dt$$

be sufficiently small. In practice "sufficiently small" is determined so that the resulting control leads to improved cost and constraints. On the other hand, the method of steepest descent determines $\|\delta u\|$ by a different scheme. In this case δu is picked along the gradient H_u (i.e., it is a function proportional to the function H_u). Consider the resulting change in cost to be a function of $\|\delta u\|$. Then $\|\delta u\|$ is picked as the smallest value which gives a local minimum to the function giving the change in cost. This example illustrates the large differences in $\|\delta u\|$ which occur when the two different criteria for determining $\|\delta u\|$ are applied in the two methods.

Some of the control iterates may be seen to have sharp discontinuities. (The computer plotting fails to show the exact plot in these

regions.) The curves for this problem exhibit a step continuity when the accessory problem has a conjugate point. This is due to the method of solution. When a conjugate point occurs, the optimization in the smaller interval produces a nonzero δu only in the smaller interval. If this δu is not zero at the ends of the small interval, the next resulting control becomes discontinuous. For this problem the control is updated in the second variations program by

$$u^{(n+1)} = u^{(n)} - [H_{uu} + W]^{-1}(H_u + \delta x' P f_u + b' f_u)$$

$$= u^{(n)} - (\lambda_2 + u^{(n)} + \delta x_1 p_{11} + \delta x_2 p_{12} + b_2)/(1 + W)$$

$$u^{(n+1)} = [-\lambda_2 + (x_1^{(n)} - x_1^{(n+1)})\, p_{11} + (x_2^{(n)} - x_2^{(n+1)})\, p_{12} - b_2]/(1+W) \tag{7.35}$$

Since λ, x, b, and P are continuous, $u^{(n+1)}$ will also be continuous on the next iteration provided there are no conjugate points.

The optimal trajectories as computed by 2VVP are shown in Fig. 7.11. Although the nonlinear system equation differs considerably from the response of the linearized version discussed in Section A of this chapter, the controlled responses are quite similar. (Compare Fig. 7.11 and the first 5 seconds of Fig. 7.1.) The control law is shown in feedback form in Fig. 7.12. This neighboring extremal control is optimum for the given initial conditions and is correct to second order for changes in the state. These numerical results agree with previously published solutions by Merriam ([1964] pp. 266-267).

The method based on second variations has a clear advantage in this example. This is illustrated graphically in the Figs. 7.7-10 and

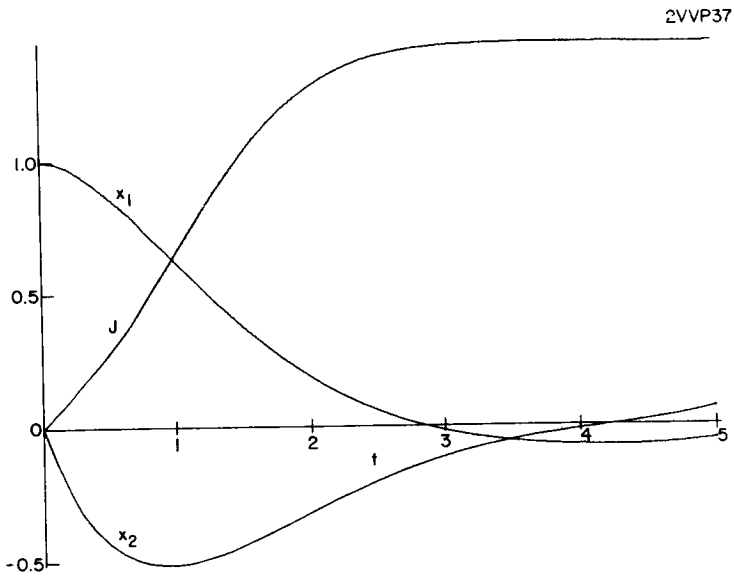

FIG. 7.11. OPTIMAL TRAJECTORIES FOR THE DRIVEN VAN DER POL EQUATION WITH AN INTEGRAL QUADRATIC LOSS FUNCTION

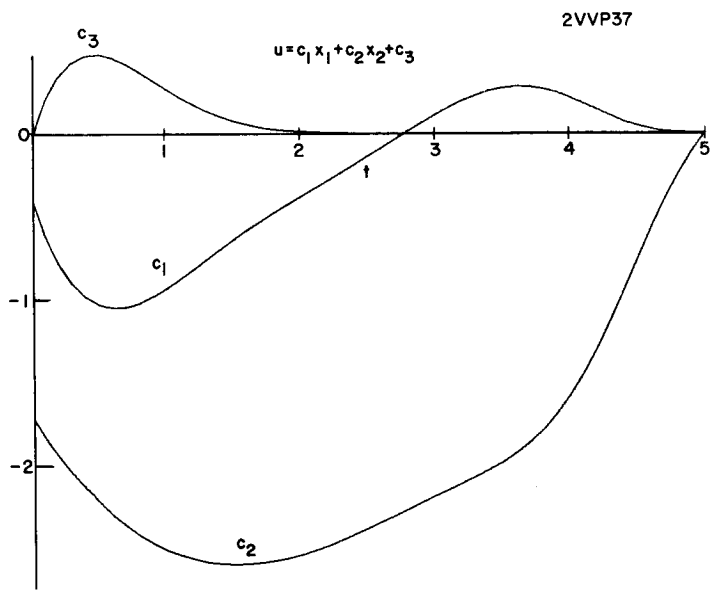

FIG. 7.12. THE TIME VARYING FEEDBACK FOR THE NEIGHBORING EXTREMAL CONTROL LAW, EXAMPLE C

numerically in the following data. For $u^{(o)} = 0$, SDVP obtained a cost which agreed with the optimal cost in only two significant figures after 18 iterations. For the same $u^{(o)} = 0$, 2VVP converged to 8 significant figures in the cost in 7 iterations and the cost was correct to 5 figures in only 5 iterations. For $u^{(o)} = 1$, SDVP took 19 iterations for a cost in error in the second significant figure. 2VVP converged to 8 figures in only 5 steps.

In the numerical results presented here, conjugate point difficulties were avoided by working the accessory problem in a smaller interval. This method proved successful in that it was able to eliminate the conjugate point in one step for both choices of the initializing control. The initial convergence rate was slowed due to this difficulty as expected. However, the rate of improvement was only slightly less than that of the steepest descent for the first few steps. It is doubtful that the frequently proposed scheme of using a steepest descent program for the first few iterations to initialize the second variations program would have much effect on the convergence rate at the added expense of writing an additional program.

The relative rates of convergence for the two methods are further compared in Fig. 7.13. This figure was made by plotting the logarithm of $J^{(n)} - J^*$ versus the iteration number where $J^{(n)}$ is the cost on the n^{th} iteration and J^* is the optimal cost. The effect of this scale is to show the errors in terms of the equivalent number of significant figures. The curves are given here for $u^{(o)} = 0$ only, since the results are similar for $u^{(o)} = 1$.

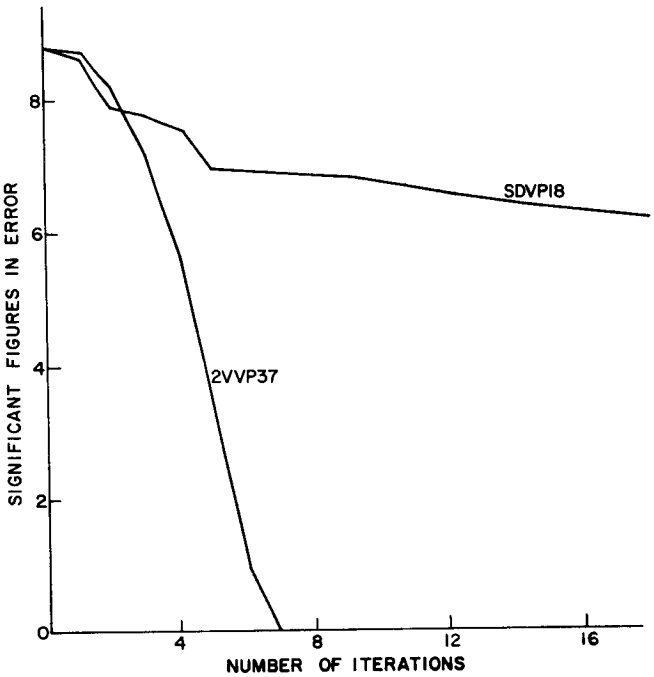

FIG. 7.13. A COMPARISON OF THE RELATIVE RATES OF CONVERGENCE FOR STEEPEST DESCENT (SDVP) AND SECOND VARIATIONS (2VVP)

D. VAN DER POL TO A LINE

This problem was chosen to illustrate the method as applied to a problem with partially specified, or Case III, type boundary conditions. The problem is the same as the problem specified in (7.27) and (7.28) of the last section, except for the boundary conditions. The initial conditions

$$x_1(0) = 1, \quad x_2(0) = 0 \tag{7.36}$$

are unchanged. The new terminal conditions require that

$$\psi[x(t_f)] = 1 - x_1(t_f) + x_2(t_f) = 0 \tag{7.37}$$

which represents a line in the state space.

The program written for this example used the method of second variations and required the solution of the adjoint equations, (7.30), the state and cost equations, (7.27) and (7.28), and the equations for M, N, and b given below. The differential equations are

$$\dot{m}_{1i} = m_{2i}$$

$$\dot{m}_{2i} = -(1 + 2x_1x_2)\, m_{1i} + (1 - x_1^2)\, m_{2i} - n_{2i}k$$

$$\dot{n}_{1i} = (2x_2\lambda_2 - 1)\, m_{1i} + 2x_1\lambda_2 m_{2i} + (1 + 2x_1x_2)\, n_{2i}$$

$$\dot{n}_{2i} = 2x_1\lambda_2 m_{1i} - m_{2i} - n_{1i} + (x_1^2 - 1)\, n_{2i}$$

$$\dot{b}_1 = n_{21}(\lambda_2 + u)\, k$$

$$\dot{b}_2 = n_{22}(\lambda_2 + u)\, k \qquad (7.38)$$

for $i = 1,2$. The constant k is equal to $1/(1 + W)$, where the constant W effectively controls the step size in control space and is adjusted by the method given in Chapter VI.

In order to find the end conditions for M and N, it is necessary to find the $n \times 1$ matrix B which is any nonzero solution to $\psi_x B = 0$. Since $\psi_x = (-1 \quad 1)$, by inspection $B = (1 \quad 1)'$. According to Fig. 6.2, the end conditions for M, N, and b are given by

$$M(t_f) = [B \quad 0] = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}$$

$$N(t_f) = [\varphi_{xx}B \quad \psi_x'] = \begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix}$$

$$b(t_f) = \begin{bmatrix} 0 \\ \delta\psi \end{bmatrix} = \begin{bmatrix} 0 \\ -\psi \end{bmatrix} \tag{7.39}$$

The control is computed from

$$u^{(n+1)} = (1 - k)\, u^{(n)} - k\,\{\lambda_2 + (0 \quad 1)(M')^{-1}[N(x^{(n+1)} - x^{(n)}) + b]\}$$

$$= (1 - k)\, u^{(n)} - k[\lambda_2 + (0 \quad 1)(M')^{-1}(b - Nx^{(n)})]$$

$$- k(0 \quad 1)(M')^{-1}Nx^{(n+1)} \tag{7.40}$$

which may also be written as

$$u^{(n+1)} = c_1(t)\, x_1(t) + c_2(t)\, x_2(t) + c_3(t) \tag{7.41}$$

with the coefficients $c_1(t)$, $c_2(t)$, and $c_3(t)$ given by

$$[c_1(t) \quad c_2(t)] = -k(0 \quad 1)(M')^{-1}N \tag{7.42}$$

and

$$c_3(t) = (1 - k)\, u^{(n)} - k[\lambda_2 + (0 \quad 1)(M')^{-1}(b - Nx^{(n)})] \,. \tag{7.43}$$

In the interval from $0.9\ t_f$ to t_f, the control is computed in terms of $\delta\lambda(t)$ by solving the differential equations

$$\dot{\delta\lambda}_1 = (2x_2\lambda_2 - 1)\,\delta x_1 + (2x_1\lambda_2)\,\delta x_2 + (1 + 2x_1x_2)\,\delta\lambda_2$$

$$\dot{\delta\lambda}_2 = (2x_1\lambda_2)\,\delta x_1 - \delta x_2 - \delta\lambda_1 + (x_1^2 - 1)\,\delta\lambda_2 \qquad (7.44)$$

where $\delta x(t) = x^{(n+1)}(t) - x^{(n)}(t)$. Equation (7.44) is integrated from $0.9\ t_f$ to t_f with the boundary conditions obtained from solving

$$M'(0.9\ t_f)\,\delta\lambda(0.9\ t_f) = N'(0.9\ t_f)\,\delta x(0.9\ t_f) + b(0.9\ t_f). \qquad (7.45)$$

The control is then found from

$$u^{(n+1)} = (1 - k)\,u^{(n)} - k[\lambda_2 + \delta\lambda_2]\ . \qquad (7.46]$$

The values of $\delta\lambda(t_f)$ obtained from the solution to (7.44) are used to find the correction to ν as shown in Chapter VI, equation (6.9).

The results of applying the computational method to the problem are shown in Fig. 7.14, which is a phase plane plot showing the trajectories for the first seven iterations. The initial trajectory, labeled iteration 0, resulted from the nominal control $u(t) = 0$. The nominal trajectory gave a cost of 7.478 and a terminal constraint error of 0.6313. After only seven iterations, the cost was reduced to 1.6857 with an error in the terminal constraint of -5×10^{-6}. A conjugate point was encountered on the second iteration so that the second and the third iterations are identical until the time of the conjugate point at $t = 3.95$ seconds.

The neighboring extremal control law for this problem is shown in Fig. 7.15. Although the feedback coefficients may be computed in the entire interval $[t_o, t_f)$, c_1 and c_2 were set to zero during the last tenth of the interval so that the final control is open loop.

Additional numerical results are given in Appendix C, Example D, which includes a table giving the values of J, ψ, and $\lambda_1(t_f)$ for each iteration. From the table, it may be observed that quite good results are obtained for the cost and the constraints even before the value of $\lambda_1(t_f)$ is correct to within two significant figures. This is because the control is not found by finding $u(t)$ in terms of $x(t)$ and $\lambda(t)$ directly, so that a fairly good value of $u(t)$ may be obtained even before the value of $\lambda(t_f)$ has converged.

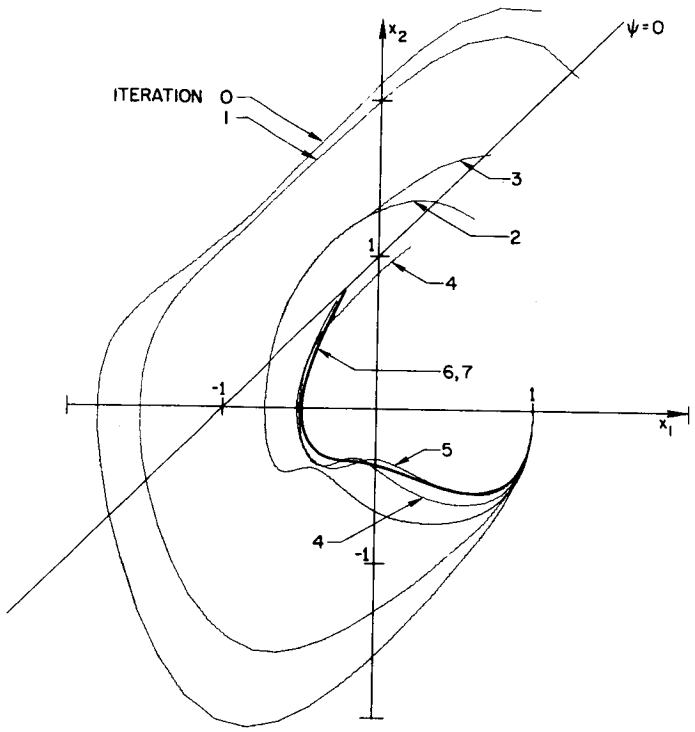

FIG. 7.14. PHASE PLANE PLOT OF SEVERAL ITERATIONS FOR THE VAN DER POL TO A LINE PROBLEM, EXAMPLE D

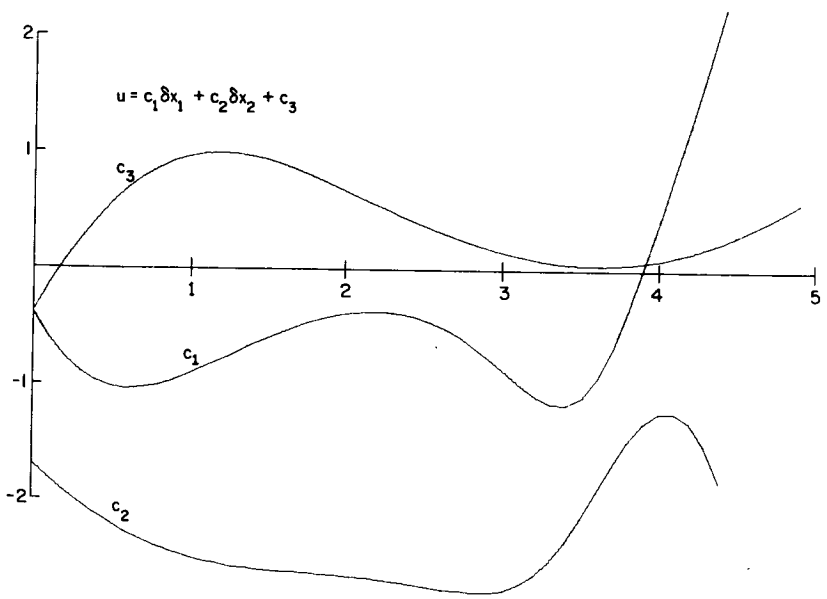

FIG. 7.15. NEIGHBORING EXTREMAL CONTROL LAW FOR THE VAN DER POL TO A LINE PROBLEM, EXAMPLE D

VIII. CONCLUSIONS

A. SUMMARY OF RESULTS

This investigation has been primarily concerned with the search for an efficient computational scheme for the solution of optimal control problems. The procedure which has been developed, while not a final solution to the problem, offers several advantages over previous methods. Some of the important features are:

1. The region of convergence is effectively as large as that of the usual gradient approach. This is a distinct advantage over most other second-order methods and eliminates the requirement for good initializing control time histories.

2. The convergence rate corresponds to that of the gradient or steepest-ascent methods initially and to the rapid second-order methods as the solution is approached.

3. Although a set of initial convergence type parameters must be specified as in the gradient methods, these parameters are automatically adjusted by the program. Poor initial guesses do not prevent convergence, but only slow it initially.

4. Adequate tests are performed without additional computation which are sufficient to show that the solution must be a minimizing curve.

5. The linear time-varying feedback coefficients for the so-called neighboring extremal control scheme are available without further calculations.

6. Terminal constraints are met "exactly," without the use of penalty functions.

7. Certain types of problems with adjustable points of discontinuity in the differential equation, known as "staging times," and "bang-bang" problems are included.

As a byproduct of the derivation of the computational method, a complete study of feedback solutions to the linear plant quadratic loss control problem with general linear end constraints is given in Chapter V. This discussion leads to an investigation of a set of sufficiency conditions for optimality for this problem.

Another result of this research which has value in itself is the work given in Chapters IV and VI on extending the method to bang-bang and related problems. In addition to the application to computing optimal trajectories, this result allows the construction of neighboring extremal solutions in a feedback fashion for this problem for the first time.

B. SUGGESTIONS FOR FUTURE RESEARCH

As is frequently the case with research, this study has perhaps uncovered more interesting problems than it has solved. The first general area for future work is the field of computational experience. It would be very instructive to try the method out on some large scale trajectory optimization problems such as a reentry calculation, in order to further test its usefulness. There is also a need for the development of a set of several standard test problems with known bad properties in order to compare the various techniques. Since it is doubtful that no single method is best for all problems, it would be very useful to be able to say something about what type of method should be used for a particular problem at hand. Another interesting point is the discrete

vs continuous optimization. Since the calculations are to be done on the digital computer and ultimately discretized, perhaps a complete discrete theory of optimization would lead to a more efficient scheme. There is virtually nothing in the current literature on a theory of errors in computing optimal controls, although it has been generally known for some time that certain problems are more difficult than others due to error propagation. Also, since most of the differential equations which must be solved in optimizing nonlinear problems are linear, more work could be done in developing special techniques for the integration of linear differential equations as well as the application of these methods to the computational technique presented here. A final computational topic would be a thorough investigation of the use of penalty functions as compared to the "exact" method for dealing with terminal constraints.

The second area is more theoretical than the first. In this development, possible singular as well as abnormal problems can arise quite naturally in the process of calculation even when the true solution may not possess any of these undesirable properties. Very little seems to be known concerning the optimization of near singular or near abnormal problems. Furthermore, problems with conjugate points can occur in the course of computation. With the exception of a few isolated papers, conjugate points are not discussed in the literature on control theory. Other areas of interest include an extension of the method to problems with state variable constraints and a consideration of sufficiency conditions for bang-bang problems. It is quite likely that the method developed for handling the bang-bang problems can be used to obtain a

complete theory of second variations for such problems and corresponding sufficiency theorems for local optimal controls.

APPENDIX A. PROOFS OF THEOREMS OF CHAPTER III

Proposition 3.3

There exists a set of positive numbers W, k, and ϵ_i, $i = 1,2,\ldots,q$, such that if h is an element in H which minimizes

$$< f_x(x_o),\ h > + 1/2W < h,\ h >$$

with

$$< g_{i,x}(x_o),\ h > = - kg_i(x_o), \quad \text{then}$$

1. for $|g_i(x_o)| > \epsilon_i$, $|g_i(x_o + h)| < |g_i(x_o)|$

2. for $|g_i(x_o)| \leq \epsilon_i$, $|g_i(x_o + h)| < \epsilon_i$

   and $f(x_o + h) < f(x_o)$

Proof: From Lemma 3.1, h is given by

$$h = - \frac{1}{W} Pf_x + k\hat{h}$$

where $\hat{h}$ is the minimum norm solution to $< g_{i,x},\ h > = - g_i$ $i = 1,2,\ldots q$ and P is a projection operator onto the nullspace of $< g_{i,x},\ >$. Since $\hat{h}$ is perpendicular to Pf_x, the norm of h satisfies

$$\|h\|^2 = \frac{1}{W^2} \|Pf_x\|^2 + k^2 \|\hat{h}\|^2 .$$

By assumption $\|f_x\|$ is bounded and $\|\hat{h}\|$ is bounded since the Gram matrix $< g_{i,x},\ g_{j,x} >$ is nonsingular. Hence there exist positive numbers M and N such that

$$\|h\|^2 < \frac{1}{W^2} M + k^2 N \quad .$$

If $|g_i(x_o)| > \epsilon_i$, then we must show that $|g_i(x_o + h)| < |g_i(x_o)|$. $g_i(x_o + h)$ may be written as

$$g_i(x_o + h) = g_i(x_o) + < g_{i,x}(x_o), h > + o(\|h\|)$$

$$= g_i(x_o) + k < g_{i,x}(x_o), \hat{h} > + o(\|h\|)$$

$$= (1 - k)g_i(x_o) + o(\|h\|) \quad .$$

If W is chosen so that $1/k = W$, then

$$g_i(x_o + h) - g_i(x_o) = - kg_i(x_o) + o(|k|) \quad .$$

By the definition of $o(|k|)$, there is a bound k_m on k such that if $o < k < k_m$ then

$$|kg_i(x_o)| > o(|k|)$$

and hence for k_m sufficiently small

$$0 < g_i(x_o + h) < g_i(x_o), \quad \text{if} \quad g_i(x_o) > 0$$

or

$$0 > g_i(x_o + h) > g_i(x_o), \quad \text{if} \quad g_i(x_o) < 0$$

It follows that

$$|g_i(x_o + h)| < |g_i(x_o)| \quad . \tag{A.1}$$

In the second case $|g_i(x_o)| < \epsilon_i$ and we must show $|g_i(x_o + h)| \leq \epsilon_i$ while $f(x_o + h) < f(x_o)$. $f(x_o + h)$ is given by

$$f(x_o + h) = f(x_o) + \langle f_x, h \rangle + o(\|h\|)$$

$$= f(x_o) - \frac{1}{W} \langle f_x, Pf_x \rangle + k \langle f_x, \hat{h} \rangle + o(\|h\|) .$$

Now choose $k = \alpha/W$ such that

$$\frac{1}{W} \langle f_x, Pf_x \rangle > k | \langle f_x, \hat{h} \rangle | .$$

There is a bound k_n such that if $k < k_n$ then

$$f(x_o + h) < f(x_o) . \tag{A.2}$$

If $g_i(x_o) \neq 0$, then the proof of the first part of the theorem holds and by choosing $k < \min (k_m, k_n)$ then (A.1) and (A.2) both hold.

If $g_i(x_o) = 0$, then we must show

$$|g_i(x_o + h)| < \epsilon_i . \tag{A.3}$$

In this case $g_i(x_o + h) = o(|k|)$. Choose k_o so that if $k < k_o$, then $|g_i(x_o + h)| < \epsilon_i$. Then if $k < \min (k_o, k_m)$, (A.1) and (A.2) both hold.

Proposition 3.4

There exists a constant ν sufficiently large such that if

$$\langle f_x, h \rangle + \frac{1}{2} \langle f_{xx} h, h \rangle + \frac{1}{2} \nu \langle h, h \rangle \tag{A.4}$$

is minimized over all $h \in H$, then

$$f(x_o + h) < f(x_o) \quad . \tag{A.5}$$

Furthermore, the minimum occurs for

$$h = -\frac{1}{\nu}[f_{xx}\frac{1}{\nu} + I]^{-1} f_x \quad .$$

Proof: The expression for the value of h which minimizes (A.4) follows directly from setting the gradient to zero and solving for h. Note that ν must be chosen sufficiently large so that there is a unique solution. The expression for h may also be written as

$$h = -\frac{1}{\nu} f_x + o(|\frac{1}{\nu}|) \quad .$$

The resulting change in f is

$$f(x_o + h) = f(x_o) + < f_x, h > + o(\|h\|)$$

$$= f(x_o) - \frac{1}{\nu} < f_x, f_x > + o(\frac{1}{\nu})$$

For $1/\nu$ small enough $\frac{1}{\nu} < f_x, f_x > > |o(\frac{1}{\nu})|$ so that

$$f(x_o + h) < f(x_o) \quad .$$

Proposition 3.5

There exists a set of constants ν and $1/k$ sufficiently large and a set of tolerances ϵ_i, $i = 1,2,\ldots q$ such that if

$$< F_x, h > + \frac{1}{2} < F_{xx} h, h > + \frac{1}{2} \nu < h, h > \tag{A.6}$$

is minimized over all $h \in H$ with

$$F(x,\lambda) = f(x) + \lambda' g(x)$$

and

$$g_i(x_o) = -\frac{1}{k} < g_{i,x}(x_o),\ h > \qquad (A.7)$$

then

1. if $|g_i(x_o)| > \epsilon_i$, $|g_i(x_o + h)| < |g_i(x_o)|$ or

2. if $|g_i(x_o)| < \epsilon_i$, $|g_i(x_o + h)| \leq \epsilon_i$ and $f(x_o + h) < f(x_o)$.

Proof: By an easy extension of Lemma 3.1, the h which minimizes (A.) while satisfying (A.7) is given by

$$h = -[F_{xx} + \nu I]^{-1} PF_x + k\hat{h}$$

where P is a projection operator onto the nullspace of $L = < g_{i,x}, \ >$, $i = 1,2,\ldots q$ and $\hat{h}$ is the minimum norm solution to $< g_{i,x}, h > = - g_i(x_o)$. Since

$$F_x = f_x + \sum_{i=1}^{q} \lambda_i g_{i,x},\quad PF_x = Pf_x + \sum_{i=1}^{q} \lambda_i Pg_{i,x} = Pf_x$$

so that

$$h = -[F_{xx} + \nu I]^{-1} Pf_x + k\hat{h}\ .$$

The expression may be further simplified as follows

$$h = -\frac{1}{\nu} Pf_x + k\hat{h} + o(\frac{1}{\nu}) .$$

The proof of the theorem then follows from Proposition 3.3.

APPENDIX B

SAMPLE PROGRAM

```
**STEEPEST DESCENT 2ND VARIATION VAN DER POL PROBLEM TO A LINE                $
 COMMENT MAIN PROGRAM BEGINS HERE                                            $
 REAL EXTERNAL PROCEDURE IPD18()                                             $
 EXTERNAL PROCEDURE SMOOTH()$  REAL SMOOTH()                                 $
 EXTERNAL PROCEDURE STARTTIMER()$ EXTERNAL PROCEDURE TIMER()                 $
 REAL TIMER()                                                                $
      INTEGER I,M,MM,INT,NSETZ,J,NBITS,NTOL,POOPED                           $
       ARRAY TITLE(12),XZ(20),YZ(20),ZZ(20),A(4),B(2),G(2),E(2),YY(20)       $
      INTEGER ARRAY AX(6) = (@    X1@,@    X2@,@  COST@,@    C1@,
                                              @    C2@,@    C3@)             $
      INTEGER ARRAY AA(12) =(@   M11@,@   M12@,@   N11@,@   N12@,
           @    B1@,@    L1@,@   M21@,@   M22@,@   N21@,@   N22@,
           @    B2@,@    L2@)                                                $
 GLOBAL REAL ARRAY RAT(2),FEATHERS(2)    $ GLOBAL REAL DETM                  $
 GLOBAL REAL ARRAY X(6,20),AU( 501),ASTATE(5, 501),ASTOVE(5, 501)            $
 GLOBAL REAL ARRAY AADJ(12, 501), SAU( 501),CEE(3,501)                       $
 GLOBAL REAL T,TEMP,FKK,FU,UU,EPS,C1,C2,C3,CC                                $
 GLOBAL INTEGER NFUNCT, N, ITIME, ICU,ICUP,NSETT                             $
 GLOBAL BACKK, CONJUGATEPOINT, HADES                                         $
 PROCEDURE OUT(K)                                                            $
      BEGIN INTEGER K,I                                                      $
        IF (NFUNCT EQL 1)                                                    $
           BEGIN                                                             $
             IF K EQL 0 AND N EQL 3                                          $
                WRITE($$HEAD1)                                               $
             PRINTOUT(T,FOR I = (1,1,3) $ X(1,I),UU,C1,C2,C3)                $
             RETURN                                                          $
           END                                                               $
        IF (NFUNCT EQL 2)                                                    $
             BEGIN  IF (K EQL 0)                                             $
               WRITE($$HEAD2)                                                $
               WRITE($$ADJ1,F05)$  WRITE($$ADJ2,F06)                         $
             RETURN  END                                                     $
 FORMAT HEAD1(B7,*T*,B14,*X1*,B13,*X2*,B12,*COST*,
      B12,*U*,B14,*C1*,B13,*C2*,B13,*C3*,WO)                                 $
 FORMAT HEAD2(B7,*T*,B22,*M*,B29,*N*,B20,*B*,B14,*L*,B12,*DETM*,WO)          $
 OUTPUT ADJ1(T,(FOR I = (1,1,6)$X(1,I)),DETM)                                $
 OUTPUT ADJ2(FOR I = (7,1,12)$X(1,I) )                                       $
 FORMAT F05(8F15.8,W4), F06(B15,6F15.8,WO)                                   $
 END OUT()                                                                   $
 PROCEDURE ADDUMS( H , ITO, TZERO,   TMAX,  KK  $ XZ()  $ F() )             $
      BEGIN                                                                  $
        BOOLEAN TIRED$  TIRED = 0                                            $
        INTEGER IT,KK,EOA,ITO,INCR,ISET,K,I,NSTEPS                           $
        INTEGER COUNT$  COUNT = 0                                            $
        ARRAY C(6,20)                                                        $
```

```
  H = ABS( H ) $ IT = ITO  $ ITIME = KK                                    $
  D = TZERO - TMAX $NSTEPS = ENTIRE(ABS(D/H)+0.5)                          $
  EITHER IF (D LSS 0.0)  $ INCR = 1                                        $
  OTHERWISE$ ( H = - H    $  INCR = -1  )                                  $
     HH = 0.5.H $ D = H/24.0 $ EOA = KK +INCR.NSTEPS                       $
START..
     T = TZERO $ ISET = 0                                                  $
     FOR I =(1,1,N)                                                       $
       (X(1,I) = X(6,I) = XZ(I))                                          $
     F(1$X(2,))  $ F(0$X(5,))                                             $
     OUT(0)                                                               $
  206..
     FOR K = (1,1,N)                                                      $
           C(1,K) = X(5,K)HH                                              $
     FOR K=(1,1,N) $ X(1,K) = X(1,K) + C(1,K)                             $
     T = T+HH                                                             $
          F( 2 $ C(2, )  ) $ ITIME = ITIME+INCR $ F(2$C(3,))              $
     FOR K = (1,1,N) $ C(2,K) = 0.5(C(2,K) + C(3,K) )HH                   $
     FOR K=(1,1,N) $ X(1,K) = X(6,K) + C(2,K)                             $
          F( 2 $ C(3, )  ) $ ITIME = ITIME-INCR $ F(2$C(4,))              $
     FOR K = (1,1,N) $ C(3,K) =0.5(C(4,K) + C(3,K))H                      $
     T = T+HH          $ ITIME = ITIME + INCR                             $
     FOR K=(1,1,N) $ X(1,K) = X(6,K) + C(3,K)                             $
          F( 1 $ C(4, )  )                                                $
     FOR K=(1,1,N)                                                        $
     (X(6,K)=X(1,K)=X(6,K)+(C(1,K)+2.C(2,K)+C(3,K)+C(4,K)HH)/3.0)         $
     ISET = ISET+1                                                        $
     IT = IT - 1                                                          $
     IF(IT EQL 0) $(OUT(1) $ IT = ITO)                                    $
     SWITCH ISET,(TIME1,TIME2,TIME3)                                      $
  TIME1..  F(0$X(3,)) $ FOR K = (1,1,N)$  X(5,K) = X(3,K)                 $
     GO TO 206                                                            $
  TIME2..  F(0$X(4,)) $ FOR K = (1,1,N) $ X(5,K) = X(4,K)                 $
     GO TO 206                                                            $
  TIME3..  F(0$X(5,))                                                     $
     IF IT EQL 0 $ ( IT = ITO $ OUT(1) )                                  $
     IF TIRED  $ RETURN                                                   $
     IT = IT-1                                                            $
     ITIME = ITIME + INCR                                                 $
     T = T + H                                                            $
     FOR K = (1,1,N)                                                      $
       X(1,K)=X(6,K)+D(55.X(5,K)-59.X(4,K)+37.X(3,K)-9.X(2,K))            $
     FOR K = (1,1,N)        $     BEGIN                                   $
       X(2,K) = X(3,K)                                                    $
       X(3,K) = X(4,K)                                                    $
       X(4,K) = X(5,K)      $     END                                     $
```

```
          IF ITIME EQL EOA $          TIRED = 1                                  $
          F(1$X(5,))                                                             $
          FOR K = (1,1,N)                                                        $
             X(6,K) = X(6,K)+D(9.X(5,K)+19.X(4,K)-5.X(3,K)+X(2,K))               $
          FOR K = (1,1,N)$  BEGIN                                                $
            IF (ABS(X(6,K)-X(1,K)) GTR 00.001ABS(X(6,K)))                        $
               BEGIN WRITE($$LOW,ACCURACY)$ COUNT = COUNT + 1                    $
                     IF COUNT GTR 30 $  GO TO HADES                              $
               END                                                               $
            X(1,K) = X(6,K)$    END
            GO TO TIME3                                                          $
     FORMAT ACCURACY(*P - C TOO LARGE, P = *,F15.8,* C = *,F15.8,
             *FOR  X*,J,WO)                                                      $
     OUTPUT LOW(X(1,K), X(6,K),K)                                                $
END ADDUMS()
                  $
PROCEDURE  FVDP(BOOL$F())                                                        $
     BEGIN INTEGER BOOL,K                                                        $
          IF BOOL$  BEGIN                                                        $
            C1 = CEE(1,ITIME) $ C2 = CEE(2,ITIME)$C3 = CEE(3,ITIME) END$
COMMENT AT AN INTERMEDIATE STEP IN THE R-K STARTING INTEGRATION, THE
          INTEGER BOOL IS 2 AND WE KEEP DX=LAST VALUE                            $
          IF BOOL LEQ 1 $                                          BEGIN         $
            DX1 = X(1,1) - ASTOVE(1,ITIME)                                       $
            DX2 = X(1,2) - ASTOVE(2,ITIME)                         END           $
          EITHER IF N GTR 3 $                                      BEGIN         $
               UU = C3 - CC.X(1,5)                                               $
               IF ITIME EQL ICUP$ UU = C3 + C1.DX1 + C2.DX2                      $
          F(4) = (2.0 . EPS . AADJ(12,ITIME) . X(1,2) - 1.0 - FU) . DX1
                 +(2.0 . EPS . AADJ(12,ITIME) . X(1,1) ) . DX2
                 +(1.0 + 2.0.EPS.X(1,1).X(1,2))X(1,5)                            $
          F(5) = DX1(2.0.EPS.AADJ(12,ITIME).X(1,1)) - DX2(1.0 + FU)
                 -X(1,4) -X(1,5)EPS(1.0 - X(1,1)X(1,1) )                 END      $
          OTHERWISE        $       UU = C3 + C1.DX1 + C2.DX2                     $
          F(1) = X(1,2)                                                          $
          F(2) = -X(1,1) + EPS(1-X(1,1)*2).X(1,2) + UU                           $
          F(3) = 0.5(X(1,1)*2 + X(1,2)*2 + UU*2)                                 $
          IF NOT BOOL $                                            BEGIN         $
               SAU(ITIME) = UU    $    A = 0.0                                   $
               FOR K = (1,1,N) $                                   BEGIN         $
                   XX=ASTATE(K,ITIME)=X(1,K)$A=MAX(ABS(XX),A)      END           $
               IF A GTR 1.0**5          $  GO    BACKK              END           $
     RETURN END FVDP()                                                           $
PROCEDURE BVDP(BOOL$F())                                                         $
BEGIN BOOLEAN BOOL $ REAL ARRAY G(2),  E(2)  ,  A(4)  ,  B(2)                    $
```

```
  FEATHERS(1) = M22 = X(1,8) $ FEATHERS(2) = M21 = X(1,7)                    $
  RAT(1)=M11=X(1,1)  $  M12= X(1,2) $  RAT(2) = -M12                         $
  N11 = X(1,3) $  N12 = X(1,4)  $  N21 = X(1,9)  $  N22 = X(1,10)            $
  B(1)= X(1,5) $  B(2)= X(1,11)$  L1 =  X(1,6)$  L2 = X(1,12)               $
       X1 = ASTOVE(1,ITIME)$ X2 = ASTOVE(2,ITIME)                            $
EITHER IF  ( NOT BOOL ) AND ( ITIME NEQ ICU)    $                BEGIN       $
 COMMENT... THE FOLLOWING CHECKS DET(M)                                      $
            DETM  = IPD18(1,1,2,RAT(),FEATHERS())                            $
            EITHER IF ITIME EQL ICU-1$(OLDSGN=SIGN(DETM)$G(1)=G(2)=0)        $
            OTHERWISE                                                        $
                 IF SIGN(DETM) NEQ OLDSGN  $    GO TO CONJUGATEPOINT         $
COMMENT FIND THE FEEDBACK CONTROL ONLY IF NOT TOO NEAR TF                    $
          IF ITIME GTR ICUP                                    $      BEGIN  $
               CEE(3,ITIME)=AU(ITIME)(1-CC)-X(1,12)CC$GO POGO END            $
  COMMENT NOW SOLVE M.G = - FU BY LEAST SQUARE ITERATION                     $
  A(1) = - M22/DETM    $    A(2) =   M12/DETM                                $
  A(3) =   M21/DETM    $    A(4) = - M11/DETM                                 $
          FOR J = (1,1,2)                                        $ BEGIN     $
          E(1) = IPD18(1,1,2,X(1,),G())                                      $
          E(2) = IPD18(7,1,2,X(1,),G(),1.0,1.0)                              $
          G(1) = IPD18(1,1,2,A(),E(),G(1),1.0)                               $
          G(2) = IPD18(3,1,2,A(),E(),G(2),1.0)                   $ END       $
          CEE(1,ITIME) = IPD18(1,3,2,G(),X(1,))CC                            $
          CEE(2,ITIME) = IPD18(1,9,2,G(),X(1,))CC                            $
          CEE(3,ITIME) = IPD18(1,1,2,G(),B()) . CC + AU(ITIME)
                        -CC(AU(ITIME)+X(1,12))                       END     $
OTHERWISE $  IF NOT BOOL $ DETM = 0.0                                        $
POGO..     FOR I = (1,1,N) $ AADJ(I,ITIME) = X(1,I)                          $
       F(1) = DM11 = M21                                                     $
       F(2) = DM12 = M22                                                     $
       F(7) = DM21 = -(1+2.EPS.X1.X2)M11 + EPS.(1-X1*2)M21 - N21.CC          $
       F(8) = DM22 = -(1+2EPS.X1.X2)M12 + EPS(1-X1*2)M22 - N22.CC            $
       F(3) = DN11 = (2EPS.L2.X2-1 -FU)M11 + (2EPS.L2.X1)M21
                   + (1+2EPS.X1.X2)N21                                       $
       F(4) = DN12 = (2EPS.L2.X2 -1 -FU)M12 + (2EPS.L2.X1)M22
                   + (1+2EPS.X1.X2)N22                                       $
       F(9) = DN21 = M11(2EPS.L2.X1) - M21(1+FU) - N11 -EPS(1-X1*2)N21       $
       F(10)= DN22 = M12(2EPS.L2.X1) - M22(1+FU) - N12 -EPS(1-X1*2)N22       $
       F(5) = DB1  = (L2+AU(ITIME))N21.CC                                    $
       F(11)= DB2  = (L2+AU(ITIME))N22.CC                                    $
       F(6) = DL1  = (1+2EPS.X1.X2)L2 - X1                                   $
       F(12)= DL2  = -L1 - EPS(1-X1*2)L2 -X2                                 $
  RETURN   END   BVDP()                                                      $
```

```
SUBROUTINE SANCTUM                                                    $
  BEGIN                                                               $
        COMMENT  EXTRAPOLATE TO END POINT WHERE M IS SINGULAR         $
          FOR I = (1,1,3)                                             $
              CEE(I,ICU)=3CEE(I,ICU-1)-3CEE(I,ICU-2)+CEE(I,ICU-3)     $
         WRITE( $ $ BYRON , WINN )                                    $
RETURN END SANCTUM                                                    $

SUBROUTINE MISSION                                                    $
    BEGIN    IF MM  $  BEGIN                                          $
         FOR I =(1,1,N-1)$ WRITE($$  JOE,  SINNOTT )  $ NFUNCT = 2    $
         IF POOPED                $                          BEGIN   $
         FOR I = (1,1,3) $ WRITE($$  JOE,  SINNOTT )  $ NFUNCT = 1    $
         FOR I = 1,2,7,8,3,4,9,10,5,11,6,12$ WRITE($$JOEY,SINNOTT)    $
                                                          END END$
    OUTPUT JOEY(FOR J = (1,INT,ICU)$AADJ(I,J),AA(I))                  $
    OUTPUT JOE(IF NFUNCT EQL 1$
         (FOR J = (     1     , INT,ICU )$ASTATE(I,J),AX(I)),
               IF NFUNCT EQL 2                                        $
         (FOR J = (     1     , INT,ICU )$  CEE (I,J),AX(I+3)))      $
    FORMAT SINNOTT(6(8F9.3,P),3F9.3,B46,A6,P)                         $
RETURN END MISSION                                                    $

COMMENT THE REST OF THE MAIN PROGRAM STARTS HERE.....                 $

  HADES.. CARDREAD(FOR I = (1,1,12)$TITLE(I))                         $
     WRITE($$TITLL,HDG)                                               $
     WRITE($$TITLL,PT)                                                $
     WRITE($$TITLL,TITFO)                                             $
     READ($$OPTS)                                                     $
     WRITE($$OPT ,F53)                                                $
COMMENT... OPTS INCLUDES PRINT INTERVAL (M), ZERO FOR NO PUNCH (MM),
STEP SIZE (H), NUMBER ITERATIONS (NSETZ), ITERATION STEP SIZE (FKK),
         FINAL TIME (TEND), EPSILON (EPS)..  IF FKK IS NEGATIVE, IT
         IT USED TO CONTROL THE CONSTRAINT ON DX, OTHERWISE DU        $
     READ($$ALOTOFSTUFF)                                              $
     WRITE($$ALOTOFSTUFFOUT,MESS)                                     $
COMMENT... THE INTEGER NBITS ROUGHLY SETS THE ACCURACY OF PSI REQUIRED,
           AND THE SECOND NUMBER(NTOL) SETS THE LOWER LIMIT ON THE TOL$
     EITHER IF FKK GTR 0 $ FU = 0.0                                   $
     OTHERWISE  $    (FU = -FKK $ FKK = 0     )                      $
     STARTTIMER(TEMP)                                                 $
     NSETT = -1 $ POOPED = 0                                          $
     ICU = ENTIRE(   TEND/H + 0.5) + 1                                $
     ICUP = ENTIRE(0.9TEND/H+0.5) + 1                                 $
     INT = ICUP/45                                                    $
```

```
            OLDPSI = OLDCOST = 1.0**+20                                    $
     COMMENT READ IN INITAL FEEDBACK GAINS (G1), (G2) AND (LAMBDA2 TF)$
     CARDREAD(G1,G2,ZZ(12) )   $  ZZ(6) = - ZZ(12)                         $
     CARDREAD(FOR I = (1,1,ICU) $    CEE(3,I))                             $
     CARDREAD(FOR I = (1,1,2)$XZ(I))                                       $
     FOR J = (1,1,ICU) $ BEGIN  CEE (1,J) = G1 $ CEE (2,J) = G2            $
        FOR I = (1,1,3) $ ASTOVE(I,J) = 0.0 $                      END     $
     FOR I = 2,3,5,8,9          $   ZZ(I) = 0.0                            $
     FOR I = 1,7,10             $ ZZ(I) = 1.0                              $
     ZZ(4) = -1                                                            $
  LOOP..  N = 3 $ NFUNCT = 1                                               $
     ADDUMS(H,M,0,0.9TEND,1$XZ()$FVDP())                                   $
FORTH..  IF NSETT GEQ 0    $                                 BEGIN
COMMENT THIS SECTION FINDS DLAMBDA(0.9TF) FROM M@DLAM =  N@DX + B      $
     A(1) =  AADJ(3,ICUP)                $  A(2) =  AADJ(9,ICUP)           $
     A(3) =  AADJ(4,ICUP)                $  A(4) =  AADJ(10,ICUP)          $
     G(1) = X(1,1) - ASTOVE(1,ICUP)      $  G(2) = X(1,2) -ASTOVE(2,ICUP)$
     B(1) = IPD18(1,1,2,A(),G(),AADJ( 5,ICUP),1.0)                         $
     B(2) = IPD18(3,1,2,A(),G(),AADJ(11,ICUP),1.0)                         $
     RAT(1) = AADJ(8,ICUP)               $  RAT(2) = -AADJ(7,ICUP)         $
     DETM = IPD18(1,1,2,RAT(),AADJ(,ICUP)    )                            $
     YZ(1) = -AADJ(8,ICUP) / DETM        $  YZ(2) =  AADJ(7,ICUP) / DETM $
     YZ(3) =  AADJ(2,ICUP) / DETM        $  YZ(4) = -AADJ(1,ICUP) / DETM $
     A(1) =  AADJ(1,ICUP)                $  A(2) =  AADJ(7,ICUP)           $
     A(3) =  AADJ(2,ICUP)                $  A(4) =  AADJ(8,ICUP)           $
     G(1) = G(2) = 0.0                                                     $
     FOR J = (1,1,3)$                                          BEGIN       $
          E(1) = IPD18(1,1,2,A (),G(),-B(1),1.0)                           $
          E(2) = IPD18(3,1,2,A (),G(),-B(2),1.0)                           $
          G(1) = IPD18(1,1,2,YZ(),E(),G(1),1.0)                            $
          G(2) = IPD18(3,1,2,YZ(),E(),G(2),1.0)                END         $
     WRITE( $ $ SHA,ZAM   )                                                $
     X(1,4) = G(1) $ X(1,5) = G(2)  $  N = 5                   END         $
     FOR I = (1,1,N) $ YZ(I) = X(1,I)                                      $
     ADDUMS(   H,  M,0.9TEND,TEND,  ICUP   $YZ()$FVDP())                  $
     TEMP = TIMER(TEMP) $ WRITE($$LOT1,FORT) $ STARTTIMER(TEMP)            $
     DLAM1 = 0.5(X(1,4) - X(1,5)) $  N = 3                                 $
     PSI = -1.0 - X(1,1) + X(1,2)                                          $
     WRITE($$LSTAT,FOO) $ WRITE($$PPSI,FO1)                                $
COMMENT EVALUATE NEW ITERATION AND ADJUST CONVERGENCE FACTORS              $
     NEWCOST = SMOOTH(X(1,3)) $ NEWPSI = SMOOTH(ABS(PSI)+2*NBITS)          $
     IF NEWPSI GTR OLDPSI AND NEWCOST GTR OLDCOST                          $
          BEGIN WRITE ($ $ PSPOTOMATIC ) $ GO BACKTO END                   $
     IF NEWCOST EQL OLDCOST AND NEWPSI EQL OLDPSI                          $
          BEGIN                                                            $
            EITHER IF NBITS LEQ NTOL $(POOPED = 1 $WRITE($$HOTDOG) )  $
            OTHERWISE$( NBITS = NBITS - 4 $ WRITE( $ $ SCRU,NCH   )    )$
          END                                                              $
```

```
GOODRUN.. NSETT = NSETT + 1 $ WRITE( $ $ MAD,EIT )                          $
     IF NSETT EQL NSETZ $ POOPED = 1                                        $$
     FOR I = (1,1,ICU)$ AU(I) = SAU(I)                                      $$
     FOR I = (1,1,N) $ FOR J = (1,1,ICU) $ ASTOVE(I,J) = ASTATE(I,J)        $$
                       ENTER   MISSION                                      $$
     IF POOPED  $ BEGIN WRITE( $ $ LU,PUNU ) $ GO TO HADES END              $$
     OLDCOST = NEWCOST   $  OLDPSI = NEWPSI                                 $$
     FKK = 0.5.FKK  $ FU = 0.5FU                                            $$
COMMENT... UPDATE BOUNDARY VALUES                                           $$
     ZZ(6) = ZZ(6) + DLAM1  $ ZZ(12) = - ZZ(6) $ ZZ(11) = PSI               $$
                GO BOCK                                                     $$
  BACKK.. PRINTOUT(@BLEW UP  @ ,X(1,1),X(1,2),X(1,3),@AT T =@,T)            $$
  BACKTO..  FKK = 10FKK  $ FU = 10FU                                        $$
  BOCK..  WRITE($$BRAD,AFMAN)    $  CC = 1.0/(1.0 + FKK)$WRITE($$PAGE)      $$
     NFUNCT = 2$ N = 12                                                     $$
     ADDUMS(   H,  M,TEND,0.9TEND,ICU$ZZ()$BVDP())                         $$
     FOR I =(1,1,N) $ YZ(I) = X(1,I)                                        $$
     ADDUMS(H,M,0.9TEND,0,  ICUP   $YZ()$BVDP())                            $$
  9..TEMP = TIMER(TEMP) $ WRITE($$LOT1,FORT) $ STARTTIMER(TEMP)             $$
     WRITE($$HEAD3)$  OUT(0) $ WRITE($$PAGE)                                $$
                           ENTER SANCTUM     $   GO TO LOOP                 $$
CONJUGATEPOINT..  WRITE($$CON,JU)$  OUT(0)                                  $$
  TEMP = TIMER(TEMP) $ WRITE($$LOT1,FORT) $ STARTTIMER(TEMP)                $$
  WRITE($$PAGE)                                                             $$
  NFUNCT = 1$  N = 3                                                        $$
  J = MIN(ITIME + 5 + M, ICU - 1)                                           $$
  J = J - MOD(J,M) + 1                                                      $$
  IF J GEQ   ICUP    $ BEGIN                                                $$
       IF J GEQ ICU  $   (PRINTOUT(@CONJUGATE PT TOO CLOSE TO TF@)          $$
                         GO TO HADES       )                              $$
       ITIME =   ICUP       $   ENTER SANCTUM                             $$
       FOR I = (1,1,N) $X(1,I) = ASTOVE(I,ITIME) $GO FORTH  END             $$
  TZ= T + FLOAT(J - ITIME)H  $ ITIME = J                                    $$
  ENTER SANCTUM                                                             $$
  FOR I = (1,1,3)$ YZ(I) = ASTOVE(I,ITIME)                                  $$
     ADDUMS(H,M,TZ,0.9TEND,ITIME$YZ()$FVDP())                               $$
     GO FORTH $  COMMENT ********** END OF PROGRAM**********                $$
OUTPUT TITLL(FOR I = (1,1,12)$TITLE(I))                                     $$
FORMAT(TITF0(A72,W9) ,HDG(A72,W7))                                          $$
INPUT OPTS(M,MM,H,NSETZ,FKK,TEND,EPS)                                       $$
  OUTPUT OPT(M,MM,H,NSETZ,FKK,TEND,EPS)                                     $$
FORMAT F53(WO,*PRINT INTERVAL *,I5,*, MM *,J,*, H *,F8.2,*, NSETZ *,J,
  WO,*PRESENT PENALTY ON STEP SIZE...  *,F15.8,*   TEND  *,F15.8,
  * EPS = *,F15.8,WO)                                                       $
OUTPUT CON(T) , PPSI(PSI)                                                   $
FORMAT HOTDOG(*HOT DOG... COST IS UNCHANGED WITHIN SIX BITS,
      ...PROBLEM SOLVED...*,WO)                                             $
OUTPUT LSTAT(FOR I = (1,1,N)$ASTATE(I,ICU))   , BRAD(FKK+FU)                $
FORMAT FOO(WO,*THE FINAL STATES ARE..*,WO,2F18.8,WO,*WITH A COST
  OF  *,F15.8,WO) , FO1 (@PSI IS ...  @,F18.8,WO)                           $
FORMAT AFMAN(*PRESENT PENALTY ON STEP SIZE...  *,F15.8,WO)                  $
```

```
FORMAT PT(A72,P)                                                          $
OUTPUT LU(FOR I = (1,1,ICU)$AU(I))                                        $
FORMAT PUNU(5F15.8,P)                                                     $
FORMAT HEAD3(*THE INITIAL VALUES ARE...  *,WO)                            $
OUTPUT LOT1(FIX(1000TEMP))$FORMAT FORT(*ELAPSED TIME = *,J,*MSEC*,WO)     $
FORMAT JU(*AHA.. SUSPECT CONJUGATE POINT NEAR T = *,X5.2,* SECONDS *,
        WO,*  CURRENT VALUES ARE   *,WO)                                  $
FORMAT PSPOTOMATIC(*OH NUTS.. CONSTRAINTS NOT IMPROVED, TRY AGAIN WITH
 *,      *SMALLER STATE SPACE STEP*,WO)                                   $
FORMAT PAGE(W1)                                                           $
OUTPUT BYRON (FOR I = (1,1,3)$FOR J = ICU-3,ICU-2,ICU-1,ICU$CEE (I,J))$
FORMAT WINN (WO,@THE LAST 3 + EXTRAPOLATED VALUES OF C1,C2, + C3 WERE@,
          WO,3(B20,4F15.8,WO))                                            $
OUTPUT MAD(NSETT),SCRU(NBITS)                                             $
FORMAT EIT(*THIS RUN LOOKS GOOD. ITERATION NUMBER *,J,W4)                 $
FORMAT NCH (*TIGHTEN ERROR MARGIN ON PSI. NBITS = *,J,WO)                 $
FORMAT MESS(*PSI IS ADDED TO *,J,* BEFORE ANY TEST IS MADE*,WO
          * AT THE END PSI IS COMPARED TO *,J,WO)                         $
INPUT ALOTOFSTUFF ( NBITS , NTOL )                                        $
OUTPUT ALOTOFSTUFFOUT (2*NBITS,2*NTOL)                                    $
OUTPUT SHA (  G(1) , G(2) , E(1) , E(2)   )                               $
FORMAT ZAM (@NEW DELTA LAMBDA(0.9TF) = @,2F15.8,@ ERROR = @,2F15.8,WO)$

FINISH                                                                    $

  ***** BINARY DECKS FOR MACHINE LANGUAGE PROGRAMS IN HERE*****

2FINISH                                                                   $
```

APPENDIX C

DETAILS OF NUMERICAL EXAMPLES

Example A Linear Quadratic Loss Problem

Program Titles	LQL (steepest descent) 2MV (second variations)
System	$\dot{x}_1 = x_2$ $\dot{x}_2 = -x_1 + u$
Cost Function	$J = 1/2 \int_0^{10} (x_1^2 + x_2^2 + u^2)\, d\sigma$
Initial Conditions	$x_1(0) = 1, \quad x_2(0) = 0$
Terminal Conditions	$t_f = 10, \quad x_f$ free
Integration Step Size	0.01 (very conservative)
Trajectory Storage Interval	0.05, 201 points each

Results:

	LQL	2MV
Time/Iteration	12.1 sec.	11.6 sec.
Realistic[1] Time/Iteration	5.7 sec.	11.6 sec.
Cost after (N) Iterations	0.962250 (9)	0.956137 (1)
$x_1(10)$	0.006445	-0.002774
$x_2(10)$	-0.01492	+0.0006251
$\lambda_1(0)$	1.912	1.912
$\lambda_2(0)$	0.4140	0.4140
$p_{11}(0)$	-	1.912
$p_{12}(0)$	-	0.4142
$p_{22}(0)$	-	1.352
$b_1(0)$	-	-0.00041
$b_2(0)$	-	-0.000014

Notes: 1. Realistic time indicates the program time without the step size optimization loop.

Example B The Brachistochrone

Program Title	2VBRA (second variations)
System[1]	$\dot{x} = u$
Cost Function	$J = 1/2 \int_0^1 [(1 + u^2)/(1 + x)]^{1/2} \, d\sigma$
Initial Conditions	$x(0) = 0$
Terminal Conditions[2]	$x(1) = +0.5$
Integration Step Size	$0.01 \quad 0 \leq t \leq 0.9$
	$0.0001 \quad 0.9 \leq t \leq 1.0$
Trajectory Storage Interval	each integration step stored, 191 points per variable

Results:

Time/Iteration	2.4 sec.
Cost after 3 Iterations[3]	0.99849
$\lambda_1(1)$	0.21627
$\lambda_1(0)$	0.61365
$b(0)$	-0.00429
$R(0)$	0.95715
$x(1)$	0.50000

Notes:

1. Alternate choices of the state variables are possible. A different choice which leads to simplified equations is $x = \xi$, $u = d\xi/d\eta$.
2. The corresponding condition for the original problem variables is $\xi(1) = -0.5$.
3. In 3 iterations the trajectory, control, and cost all agreed with the optimal solution to within 5 figures, the accuracy justified by the integration errors.

Example C Free Van Der Pol

Program Titles	SDVP (steepest descent) 2VVP (second variations)
System	$\dot{x}_1 = x_2$ $\dot{x}_2 = -x_1 + x_2(1 - x_1^2) + u$
Cost Function	$J = 1/2 \int_0^5 (x_1^2 + x_2^2 + u^2)\, d\sigma$
Initial Conditions	$x_1(0) = 1, \quad x_2(0) = 0$
Terminal Conditions	$t_f = 5, \quad x_f$ free
Integration Step Size	0.025 for SDVP with $u^{(0)} = 0$ 0.1 for others
Trajectory Storage	each integration step stored 201 points for SDVP, $u^{(0)} = 0$ 51 points for others

Results:	SDVP[1]	2VVP[1]
Time/Iteration	1.7 sec.	0.7 sec.
Realistic Time/Iteration[2]	0.6 sec.	0.7 sec.
Cost after (N) Iterations	1.72403 (12)	1.43350 (7)
$x_1(5)$	0.0745010	-0.0519296
$x_2(5)$	-0.459410	+0.0662353
$\lambda_1(0)$	2.30185	2.43604
$\lambda_2(0)$	1.06942	0.412329
$p_{11}(0)$	-	1.01156
$p_{12}(0)$	-	0.413450
$p_{22}(0)$	-	1.72858
$b_1(0)$	-	0.00079
$b_2(0)$	-	-0.00107

Notes: 1. These results are for $u^{(0)} = 1$.

2. See Note 1, Example A.

Example D Van Der Pol to a Line

Program Title	VDPTL
System	$\dot{x}_1 = x_2$ $\dot{x}_2 = -x_1 + x_2(1 - x_1^2) + u$
Cost Function	$J = 1/2 \int_0^5 (x_1^2 + x_2^2 + u^2)\, d\sigma$
Initial Conditions	$x_1(0) = 1, \quad x_2(0) = 0$
Terminal Conditions	$\psi = 1 - x_1(t_f) + x_2(t_f) = 0$
Integration Step Size	0.025
Trajectory Storage	each integration step stored for a total of 201 points

Results:

Time/Iteration[1]	6.14 sec.
Cost after 7 Iterations	1.6857157
ψ after 7 Iterations	-4.97×10^{-6}
Cost after 10 Iterations	1.6857045
ψ after 10 Iterations	1.60×10^{-6}
x(5)	(-.22931 +.77068)
$\lambda(5)$	(.59248 -.59248)
$\lambda(0)$	(2.3766 .38855)
b(0)	(-0.0011 -.0015)

Notes: 1. This time is large due to a conservative (small) integration step size.

Example D (Cont.)

Iteration #	Cost	ψ	$\lambda_1(t_f)$
0	7.4780	.63131	0
1	6.2783	-.0519	-2.0267
2[1]	3.0891	-.3279	-7.5890
3	3.0011	-.00534	1.7636
4	1.9177	-.1172	-0.2283
5	1.6991	-.0184	0.7909
6	1.6871	+.00067	0.6002
7	1.6857	-.0000049	0.59309
8	1.6857	-.000000089	0.59249
9	1.6857	+.000001609	0.59249
10	1.6857	+.000001765	0.59248

Notes: 1. A conjugate point was encountered at t= 3.45 seconds.

APPENDIX D

PROPERTIES OF THE FUNDAMENTAL MATRIX FOR THE EULER-LAGRANGE EQUATIONS

Several properties of the transition matrix $\Phi(t, \tau)$ for the homogeneous Euler-Lagrange equations are necessary to derive Property 2 of Chapter 5, Section D. Since these properties are relatively unknown in the literature except in Kalman and Englar [1965], they will be derived as necessary before presenting the proof of Property 2.

The homogeneous form of the Euler-Lagrange equations to be studied here may be written as

$$\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} F & S \\ Q & -F' \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{D.1}$$

where x and y are $n \times 1$ vectors and F, S, and Q are $n \times n$ matrices with S and Q symmetric. The fundamental matrix $\Phi(t, \tau)$ will be written in partitioned form in terms of four $n \times n$ matrices as

$$\Phi(t, \tau) = \begin{pmatrix} \phi_{11}(t, \tau) & \phi_{12}(t, \tau) \\ \phi_{21}(t, \tau) & \phi_{22}(t, \tau) \end{pmatrix} \tag{D.2}$$

It will be convenient to define the $2n \times 2n$ matrix J in terms of the $n \times n$ identity matrix I_n by

$$J = \begin{pmatrix} 0 & I_n \\ -I_n & 0 \end{pmatrix} \tag{D.3}$$

Note that J satisfies the following identities

$$JJ' = I_{2n}$$

$$J' = -J \tag{D.4}$$

Another useful definition is the symplectic property of a matrix. A matrix A is said to be <u>symplectic</u> if it satisfies the relation

$$A^{-1} = J'A'J \tag{D.5}$$

In the following, it will be shown that the fundamental matrix $\Phi(t, \tau)$ corresponding to the Euler-Lagrange equations (D.1) is symplectic.

<u>Theorem</u> $\Phi(t, \tau)$ is symplectic.

<u>Proof</u>: Since Φ satisfies

$$\dot{\Phi} = \begin{pmatrix} F & S \\ Q & -F' \end{pmatrix} \Phi \qquad \Phi(\tau, \tau) = I$$

then from the identity $JJ' = I$,

$$\frac{d}{dt}(J'\Phi'J) = J'\Phi'JJ' \begin{pmatrix} F & S \\ Q & -F' \end{pmatrix}' J$$

or

$$\frac{d}{dt}(J'\Phi'J) = -(J'\Phi'J) \begin{pmatrix} F & S \\ Q & -F' \end{pmatrix}.$$

The proof is completed by showing that Φ^{-1} satisfies the same differential equation as $J'\Phi'J$ since $\Phi^{-1}(\tau, \tau) = J'\Phi'(\tau, \tau)\, J = I$. Differentiation of the identity $\Phi^{-1}\Phi = I$ with respect to time may be used to show

$$\frac{d}{dt}(\Phi^{-1}) = -\Phi^{-1}\frac{d}{dt}(\Phi)\,\Phi^{-1}$$

$$= -\Phi^{-1}\begin{pmatrix} F & S \\ Q & -F' \end{pmatrix}$$

which is the desired result.

From the identities $\Phi^{-1}\Phi = I$ and $\Phi\Phi^{-1} = I$ and the symplectic property of Φ, the following set of relations may be obtained:

$$\phi'_{11}\phi_{21} = (\phi'_{11}\phi_{21})' \tag{D.6a}$$

$$\phi_{11}\phi'_{12} = (\phi_{11}\phi'_{12})' \tag{D.6b}$$

$$\phi'_{22}\phi_{12} = (\phi'_{22}\phi_{12})' \tag{D.6c}$$

$$\phi_{22}\phi'_{21} = (\phi_{22}\phi'_{21})' \tag{D.6d}$$

and

$$\phi'_{22}\phi_{11} - \phi'_{12}\phi_{21} = I \tag{D.7a}$$

$$\phi_{22}\phi'_{11} - \phi_{21}\phi'_{12} = I \tag{D.7b}$$

The proof of Property 2 of the matrices $M(t)$ and $N(t)$ as given in Chapter 5, Section D, will now be given.

Property 2 $M'(t)\,N(t) = N'(t)\,M(t)$ for all t if $M'(t_f)\,N(t_f) = N'(t_f)\,M(t_f)$.

Proof: Since $\begin{pmatrix} M(t) \\ N(t) \end{pmatrix}$ are solutions to equation (D.1), they may be written in terms of their values at $t = t_f$ as

$$\begin{pmatrix} M(t) \\ \\ N(t) \end{pmatrix} = \Phi(t,\ t_f) \begin{pmatrix} M(t_f) \\ \\ N(t_f) \end{pmatrix}$$

or

$$M(t) = \phi_{11} M(t_f) + \phi_{12} N(t_f) \tag{D.8a}$$

$$N(t) = \phi_{21} M(t_f) + \phi_{22} N(t_f). \tag{D.8b}$$

For convenience $M(t)$ will be written simply as M, $N(t)$ as N, $M(t_f)$ as A, and $N(t_f)$ as B. Then from (D.8a) and (D.8b) one obtains

$$\begin{aligned} M'N - N'M = {} & A'(\phi_{11}'\phi_{21} - \phi_{21}'\phi_{11})\ A \\ & + B'(\phi_{12}'\phi_{22} - \phi_{22}'\phi_{12})\ B \\ & + A'(\phi_{11}'\phi_{22} - \phi_{21}'\phi_{12})\ B \\ & + B'(\phi_{12}'\phi_{21} - \phi_{22}'\phi_{11})\ B. \end{aligned}$$

Using (D.6) and (D.7), this reduces to

$$M'N - N'M = A'B - B'A,$$

which shows that $M'(t)\ N(t) = M'N$ is symmetric if and only if $A'B = M'(t_f)\ N(t_f)$ is symmetric.

REFERENCES

R. Bellman, Dynamic Programming, Princeton University Press, Princeton, New Jersey, 1957.

G. A. Bliss, Lectures on the Calculus of Variations, University of Chicago Press, Chicago, 1946.

J. V. Breakwell, "The Optimization of Trajectories," J. Soc. Indust. Appl. Math., 7, 2, Jun 1959.

J. V. Breakwell and A. E. Bryson, "Neighboring Optimum Terminal Control for Multivariable Nonlinear Systems," SIAM Symposium on Multivariable System Theory, Cambridge, Massachusetts, 1962.

J. V. Breakwell, and Y. C. Ho, "On the Conjugate Point Condition for the Control Problem," Int. J. Engng. Sci., 2, 1965.

J. V. Breakwell, J. L. Speyer, and A. E. Bryson, Jr., "Optimization and Control of Nonlinear Systems Using the Second Variation," J. Soc. Indust. Appl. Math. on Control, Ser. A, 1, 2, Feb 1963.

R. E. Brown, "Some Numerical Aspects of Steepest Descent Trajectory Optimization," presented at AIAA/ION Astrodynamics Guidance and Control Conference, UCLA, Los Angeles, California, Aug 1964.

A. E. Bryson, "Optimal Programming and Control," Proceedings of the IBM Scientific Computing Symposium on Control Theory and Applications, Yorktown Heights, New York, Oct 1964.

A. E. Bryson and W. F. Denham, "Multivariable Terminal Control To Minimize Mean Square Deviation from a Nominal Path," Proceedings of Symposium on Vehicle Systems Optimization, Inst. Aerospace Sciences, Nov 1961. (Also Raytheon Report BR-1333).

A. E. Bryson and W. F. Denham, "A Steepest Ascent Method for Solving Optimum Programming Problems," Jour. Appl. Mech., Series E, 29, 2, Jun 1962.

E. A. Coddington and N. Levinson, Theory of Ordinary Differential Equations, McGraw-Hill Book Co., New York, 1955.

B. Friedman, Principles and Techniques of Applied Mathematics, John Wiley and Sons, New York, 1956.

I. M. Gelfand and S. V. Fomin, Calculus of Variations, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1963.

K. Hales, "Minimum-Fuel Attitude Control of a Rigid Body in Orbit by an Extended Method of Steepest-Descent," Ph.D. Dissertation, Stanford University, Stanford, California, 1966.

R. W. Hamming, Numerical Methods for Scientists and Engineers, McGraw-Hill Book Co., Inc., New York, 1962.

L. Hurwicz, "Programming in Linear Spaces," in Studies in Linear and Nonlinear Programming, Arrow, Hurwicz, and Uzawa, eds, Stanford University Press, Stanford, California, 1958.

A. H. Jazwinski, "Optimal Trajectories and Linear Control of Nonlinear Systems," AIAA Journal, 2, 8, August 1964.

R. E. Kalman, Y. C. Ho, and K. S. Narendra, "Controllability of Linear Dynamical Systems," Contributions to Differential Equations, 1, 1963.

R. E. Kalman and T. S. Englar, "An Automatic Synthesis Program for Control and Optimization," final report on Contract NAS 2-1107, RIAS, Baltimore, 1965.

L. V. Kantorovich and G. P. Akilov, Functional Analysis in Normed Spaces, MacMillan Company, New York, 1965.

H. J. Kelly, "Gradient Theory of Optimal Flight Paths," ARS Journal, 30, 10, Oct 1960.

H. J. Kelly, R. E. Kopp, and H. G. Moyer, "A Trajectory Optimization Technique Based upon the Theory of the Second Variation," Progress in Astronautics and Aeronautics, 14, 1964.

P. Kenneth, and G. E. Taylor, "Solution of Variational Problems with Bounded Control Variables by means of the Generalized Newton-Raphson Method," Symposium on Recent Advances in Optimization Techniques, Carnegie Institute of Technology, Pittsburgh, Pennsylvania, April 21-23, 1965.

W. Kipiniak, Dynamic Optimization and Control, M.I.T. Press, Mass. Institute of Technology and John Wiley and Sons, Inc., New York, 1961.

R. E. Kopp and R. McGill, "Several Trajectory Optimization Techniques, Part I; Discussion," in Computing Methods in Optimization Problems, Academic Press, New York, 1964.

Geoffrey N. T. Lack, "Optimization Studies with Applications to Planning in the Electric Power Industry and Optimal Control Theory," Report CCS-5, Institute in Engineering-Economic Systems, Stanford University, Stanford, California, Aug 1965.

R. E. Larson, "Dynamic Programming with a Continuous Independent Variable," Ph.D. Thesis at Stanford Univ., Stanford, California, 1964.

D. G. Luenberger, Lecture notes for EE292h, Stanford University, spring, 1964.

L. A. Liusternik and V. J. Sobolev, Elements of Functional Analysis, Frederick Ungar Publishing Co., New York, 1961.

R. McGill, "Optimal Control, Inequality State Constraints, and the Generalized Newton-Raphson Algorithm," SIAM Journal, Series A: Control, 3, 2, 1965.

S. R. McReynolds and A. E. Bryson, "A Successive Sweep Method for Solving Optimal Programming Problems," Technical Report 463, Cruft Laboratory, Division of Engineering and Applied Physics, Harvard University, Cambridge, Mass., 1965. (Also Proceedings of Joint Automatic Control Conference, Rensselear Polytechnic Institute, Troy, New York, June 1965.)

S. R. McReynolds, "A Successive Sweep Method for Solving Optimal Programming Problems," Ph.D. Dissertation, Harvard University, Cambridge, Massachusetts, 1966.

C. W. Merriam, Optimization Theory and the Design of Feedback Control Systems, McGraw-Hill, Inc., New York, 1964.

C. W. Merriam, "An Algorithm for the Iterative Solution of a Class of Two Point Boundary Value Problems," Information and Control, 8, 2, Apr. 1965.

G. H. Moyer and G. Pinkham, "Several Trajectory Optimization Techniques, Part II: Application," in Computing Methods in Optimization Problems, Academic Press, New York, 1964.

J. A. Payne, "Computational Methods in Optimal Control Problems," Technical Report AFFDL-TR-65-50, Department of Engineering, U.C.L.A., Los Angeles, 1965.

L. S. Pontryagin, V. G. Boltyanskii, and R. V. Gamkrelidze, "On the Theory of Optimal Processes," Doklady Aka. Nauk S.S.S.R., 110, 1956.

L. S. Pontryagin, V. G. Boltyanskii, R. V. Gamkrelidze, and E. F. Mischenko, The Mathematical Theory of Optimal Processes, Interscience, New York, 1962.

R. Rosenbaum, "Convergence Technique for Steepest Descent Method of Trajectory Optimization," AIAA Journal, 1, 7, July 1963.

J. F. Sinnott, Private Communication, Stanford University, Stanford, California, 1966.

G. E. Shilov, An Introduction to the Theory of Linear Spaces, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1961.

R. T. Stancil, "A New Approach to Steepest Ascent Trajectory Optimization," AIAA Journal, 2, 8, Aug 1964.

R. F. Vachino, "Steepest Descent with Inequality Constraints," Journal of SIAM on Control, 4, 1, 1966.

P. A. Valentine, "The Problem of Lagrange with Differential Inequalities as Added Side Conditions," Ph.D. Dissertation, Department of Mathematics, University of Chicago, Chicago, Illinois, 1937.

C. P. Van Dine, "Application of Newton's Method to the Finite Difference Solution of Non-Linear Boundary Value Systems," Report UAR-D37, Research Laboratories, United Aircraft Corporation, Mar 1965.

C. P. Van Dine, W. R. Fimple, and T. N. Edelbaum, "Application of a Finite-Difference Newton-Raphson Algorithm to Problems of Low Thrust Trajectory Optimization," AIAA paper No. 65-698, AIAA/ION Astrodynamics Specialist Conference, Monterey, California, Sept 16-17, 1965.