SENSITIVITY THEORY APPLICATION TO THE NUMERICAL SOLUTIONS OF THE GENERAL OPTIMAL CONTROL PROBLEM

by

İbrahim Eksin

B.Sc. in Electrical Eng., Boğaziçi University, 1976

M.Sc. in Electrical Eng., Boğaziçi University, 1979

Submitted to the Institute for Graduate Studies in Science and Engineering in partial fulfillment of the requirements for the degree of

Doctor

of

Philosophy

in

Electrical Engineering

BOĞAZİÇİ UNIVERSITY

1983

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii

ABSTRACT iv

ÖZETÇE v

I. INTRODUCTION 1

1. BACKGROUND OF THE PROBLEM AND OUTLINE OF THE THESIS 1

II. A SURVEY ON THE NUMERICAL SOLUTIONS OF TWO-POINT BOUNDARY-VALUE PROBLEMS 11

1. INITIAL-VALUE PROBLEMS 11

2. TWO-POINT BOUNDARY-VALUE PROBLEMS 14

3. NUMERICAL METHODS 18

A. SHOOTING (INITIAL-VALUE) METHODS 18

a. CONTINUATION 20

b. PARALLEL-OR MULTIPLE-SHOOTING 21

c. INVARIANT-IMBEDDING 21

B. FINITE DIFFERENCE METHODS 25

C. FUNCTION SPACE APPROXIMATION METHODS 27

4. TPBVP IN OPTIMAL CONTROL THEORY 28

III. SENSITIVITY APPROACH TO TWO-POINT BOUNDARY-VALUE PROBLEMS 34

1. A BRIEF HISTORICAL REVIEW TO SENSITIVITY THEORY 34

2. BASIC CONCEPTS AND DEFINITIONS IN SYSTEM THEORY 35

3. MOTIVATION FOR SENSITIVITY APPROACH TO TPBVP 39

4. THE TRAJECTORY SENSITIVITY FUNCTION OF CONTINUOUS SYSTEMS 40

5. SENSITIVITY APPROACH TO THE SOLUTION OF TWO-POINT BOUNDARY-VALUE PROBLEMS 42

IV. NUMERICAL RESULTS AND SIDE-PRODUCTS OF SENSITIVITY APPROACH TO THE SOLUTION OF TPBVP 46

1. NUMERICAL RESULTS 46

2. ONE-STEP CONVERGENCE PROOF FOR LINEAR TPBVP 48

3. AN EFFICIENT ALGORITHM FOR FINDING THE SUFFICIENT NUMBER OF TERMS IN NUMERICAL EVALUATION OF FUNCTIONS BY POWER SERIES 52

4. A NEW METHOD FOR NUMERICAL SOLUTION FOR LINEAR STIFF SYSTEMS 58

5. SOME EQUIVALENCE PROPERTIES AND MODIFICATIONS IN SOLVING LINEAR AND NONLINEAR TPBVP USING SENSITIVITY APPROACH 60

6. CONCLUSIONS 61

V. DIRECT SENSITIVITY APPROACH TO THE GENERAL OPTIMAL CONTROL THEORY 63

1. PROBLEM FORMULATION AND HAMILTON-JACOBI-BELLMAN EQUATION 63

2. PERFORMANCE INDEX SENSITIVITY 66

3. DIRECT SENSITIVITY APPROACH 70

VI. APRIORI POLYNOMIAL APPROXIMATION METHODS VIA DIRECT SENSITIVITY APPROACH FOR THE GENERAL OPTIMAL CONTROL PROBLEMS 74

1. BASIC IDEA OF THE METHOD 74

2. NUMERICAL RESULTS 76
3. ANOTHER ALTERNATIVE FOR BACKWARD SEQUENTIAL OPTIMIZATION PROCEDURE 81
VII. APOSTERIORI POLYNOMIAL FITTING METHOD VIA DIRECT SENSITIVITY APPROACH FOR THE GENERAL OPTIMAL CONTROL PROBLEM 85
1. BASIC IDEA OF THE METHOD 85
2. NUMERICAL RESULTS 90
VIII. NUMERICAL SOLUTION OF THE INFINITE HORIZON PROBLEMS IN OPTIMAL CONTROL THEORY 96
1. BASIC IDEA OF THE METHOD 96
2. NUMERICAL RESULTS 100
IX. CONCLUSIONS, GENERAL OVERVIEW AND COMPARISON OF THE METHODS DEVELOPED IN THIS STUDY 103
REFERENCES 105

ACKNOWLEDGEMENTS

It is my pleasant duty to express sincere gratitude to Dr. M. Akif Eyler since he is the person who suggested the main subject of this study and gave many ideas throughout its development. I am equally indepted to Doç. Dr. Yorgo İstefanopulos for his constructive and fruitful suggestions and invaluable encouragements. I also wish to express my appreciation to Doç. Dr. Bülent Sankur and Prof. Dr. Atilla Aşkar for their valuable comments. Finally, my special thanks are due to my wife who not only typed but also proofread the entire thesis.

SENSITIVITY THEORY APPLICATION TO THE NUMERICAL SOLUTIONS OF THE GENERAL OPTIMAL CONTROL PROBLEM

This study proposes various efficient numerical methods for the general optimal control problem. The basic feature of the methods developed here is that they somehow exploit the ideas and the concepts of the sensitivity theory.

First, a new method for solving TPBVP, which is met in seeking an open-loop solution for optimal control problems, is developed. This method can be briefly expressed as an iterative procedure which is based on trajectory sensitivities with respect to initial conditions.

In the second part of the study, various numerical methods are developed for the closed-loop solutions of general optimal control p problems using performance index sensitivity functions with respect to controller parameters. These new methods may be treated in two categories :

1) Apriori polynomial approximation methods

   Here the basic assumption is that controller parameter function is assumed to be formed by a polynomial function.

2) Aposteriori polynomial approximation method

   In this method a sequence of subproblems are created using some intrinsic properties of the previous method. The values of the results of the subproblems are then used in the formation of the optimum controller parameter function.

GENEL ENİYİ DENETİM SORUNUNUN SAYISAL ÇÖZÜMLERİNE DUYARLILIK KURAMININ UYGULANMASI

Bu çalışma Genel Eniyi Denetim Sorununun sayısal çözümleri ile ilgili olarak çeşitli özgün ve etkin yöntemler önermektedir. Burada geliştirilen yöntemlerin ortak özelliği hepsinin de bir şekilde duyarlılık kuramındaki kavram ve düşünceleri kullanmış olmalarıdır.

İlk olarak Eniyi Denetim Problemlerinde açık-döngü çözüm arandığında karşılaşılan iki-nokta sınır-değer sorunları için yörünge duyarlılık matrislerini kullanan ardışık yeni bir sayısal çözüm yöntemi geliştirilmiştir.

Çalışmanın ikinci kısmında ise kapalı-döngü, durum geri-beslemeli eniyi denetim sorunları için denetimci parametrelerine göre davranış ölçütünün duyarlılık vektörü kullanılarak yeni sayısal yöntemler önerilmiş ve çeşitli örnek problemlerle denenmiştir. Bu yöntemler iki gruba ayrılabilir :

1) Önsel polinom yaklaşıklaması yöntemleri

   Buradaki temel varsayım denetimci parametre fonksiyonunun bir polinom fonksiyon olmasıdır.

2) Sonsal polinom yaklaşıklaması yöntemi

   Bu yöntemde önsel polinom yaklaşıklaması yönteminin bazı özelliklerinden esinlenerek bir altproblemler dizisi yaratılmıştır. Bunların sonuç değerlerine ise bir polinom yaklaşık olarak oturtulmuş ve denetimcinin parametreleri bulunmuştur.

1

INTRODUCTION

1.1. BACKGROUND OF THE PROBLEM AND OUTLINE OF THE THESIS

No control engineer can be content with simply formulating or analyzing a control problem. He must ultimately be concerned with the problem of designing systems according to the given specifications. In early days, trial-and-error methods were the basis for most decisions in system design. However, today it is no longer a trial-and-error effort, rather a precise science involving applied mathemathics and high speed computers.

There exist mainly two approaches to the system design. One is the classical (frequency domain) and the other is the modern (time domain) approaches. In the classical approach to system design, one utilizes such frequency domain techniques as root locus and Bode diagrams to determine systems with acceptable performance. In contrast, the modern approach is formulated almost exclusively in the time domain. In addition, the modern approach demands not only acceptable but optimal performance.

In order to talk of optimal performance it is obviously necessary to specify some method for determining the quality of the performance of a system. In the modern approach, this

is often done by means of a integral performance index of the following form:

$$PI = \int_{t_i}^{t_f} l(\underline{x}, \underline{u}, t)\, dt$$

where $\underline{x}$ is the state vector and $\underline{u}$ is the control function.

One then says that a system is optimal over the time interval t_i to t_f if the value of the performance index is minimum (or maximum in some cases). It should be noted that the minimization (or maximization) of the above performance index is done over the control vector $\underline{u}$ and subject to the system equation constraint defined by a set of differential equations of the form:

$$\dot{\underline{x}} = \frac{d\underline{x}}{dt} = \underline{f}\,(\underline{x}, \underline{u}; t)$$

Eventhough the optimal control problem thus stated informally as above may seem very simple, it presents various difficulties in both formulation and solution steps. For instance, one of the basic problems in the formulation step is the translation of system specifications often in such subjective terms as "good rise time with reasonable overshoot", into the form of a performance index. While another problem of the same step may arise in the derivation of system equations from a given physical process. However, this study will not cover this kind of formulation problems.

In a gross sense, this study will cover the problems of the solution of the optimal control problem. More speci-

fically, it will propose some new methods for the solution of the optimal control problem. Furthermore, it will try to give a new insight and/or a point of view via application of sensitivity ideas in the solution step of the problem.

In the optimal control problem stated above the type of the control function is not specified. Actually, there exist only two types of control functions. The first type is the open-loop control which utilizes the measurement of the initial state to compute and generate the control signals as functions of the initial state and time. The second one is the closed-loop (or feedback) control which utilizes continuous or rapidly sampled state measurements to compute the control signal as a function of the present state and terminal state and time.

Open-loop control is used when one or more of the state variables cannot be measured during the control interval but when an initial measurement is available. However, closed-loop control can be used when all of the state variables are known. Open-loop control requires fairly exact knowledge of the system parameters and therefore system dynamics whereas closed-loop control requires less accurate system knowledge, since the effect of the control signals on system state is monitored.

Depending on the type of the control function, either calculus of variations or dynamic programming approaches have been extensively used in the mathemathical formulation and solution of the optimal control problems. If an open-loop

control is required then the calculus of variations approach ends with the well known Two Point Boundary Value Problem. In the case of closed-loop (feedback) control dynamic programming approach provides us with the partial differential equation which is known as Hamilton-Jacobi-Bellman equation.

Stating more precisely, the calculus of variations was used to derive a set of necessary conditions that must be satisfied by an optimal control and its associated state-costate trajectory. The two-point nature of the resultant boundary conditions presents a serious computational problem. Moreover, the resultant two-point boundary value problem is generally a nonlinear one which makes the problem a bit more cumbersome. There exist several elegant computational schemes developed to solve linear and/or nonlinear two-point boundary value problems. However, the basic existence and uniqueness theory for nonlinear boundary value problems is not as developed as for initial-value problems or linear boundary value problems. Therefore still the convergence to the exact solution in every problem for various computational methods remain to be in doubt.

In this study, first an overview of the various computational methods for solving the two-point boundary value problem will be given from a mathemathical point of view. Next, the two-point boundary value problem will be considered from the optimal control side. Various methods developed upto now in this discipline are discussed.

In the third chapter a new method for the solution of the two-point boundary-value problem is presented. This new method which we have called sensitivity approach provides an insight for the logic behind the solution of the problem. As a by-product of this approach a new method is developed for the solution of stiff linear differential equations. Still as another by-product a method for the determination of the sufficient number of terms in the power-series expansion of any function is presented. Theoretical and numerical aspects of the method are also discussed in this section.

Next chapter is devoted to the various numerical results obtained by the new method and comparison of it with the other methods. In this chapter alternate way of solving nonlinear two-point boundary-value problem is considered using the same approach. The nonlinear problem is considered to be alinear one around the known and the guessed (or unknown) boundary values and the linear problem is solved using the same approach until the unknown boundary value is within acceptable tolerance limits. Lastly, a conclusion section completes this chapter.

However, even if we do solve the two-point boundary value problem we still do not have an acceptable solution, since only an open-loop solution for a specific set of initial and terminal states has been found. In other words, optimal control $\underline{u}^{o}$ is known only as a function of time and not as a feedback control law depending on system state. If either the initial state or terminal state is changed, or if any

disturbance acts on the system, the control $\underline{u}^{o}(t)$ is no longer optimal.

Dynamic programming is used as an alternate approach to eliminate the above difficulty. This alternate approach removes the necessity for solving a two-point boundary-value problem and yields a closed-loop solution in the form of an optimal control law $\underline{u}^{o}(x,t)$. However, as one might expect, this closed-loop approach also has its own problems. Chief among these is the necessity of solving a nonlinear partial differential equation known as Hamilton-Jacobi-Bellman equation. In fact, the solution of this equation is so difficult that it has been accomplished only for a few special cases.

Therefore, an alternate approach of attacking the same closed-loop optimal control problem must have been introduced. Here, at this point we have again tried to exploit the sensitivity idea in order to obtain a way of formulation of the problem. Basically, we have assumed that optimal control vector consists of some linear combination of state values. That is, the optimal control vector is assumed to be in the following form :

$$\underline{u}^{o}(\underline{x},t) = \underline{k}^{T}(t)\,\underline{x}(t)$$

Since the state vector $\underline{x}(t)$ is assumed to be known for all times in order to have an optimal control law, the only thing which remains to be determined is the coefficient or the gain

vector $\underline{k}(t)$. Therefore, a sensitivity analysis on this vector would yield us some valuable measures in the way of obtaining a closed-loop solution to the optimal control problem.

In the fifth chapter, the basic idea behind this new approach is introduced. Eventhough the idea behind it may seem simple, the approach which we have called direct sensitivity approach, provides one a vast amount of various attacking opportunities for the same problem. Theoretical and numerical aspects of the approach are also discussed in this chapter.

In the next chapter, two new methods for determining the coefficient (or the gain) vector are developed using the same direct sensitivity approach. These two new methods presented in this chapter can be named as "apriori polynomial fitting methods", since the coefficient vector is assumed to be formed by polynomials. Various numerical examples solved using the new apriori polynomial fitting methods are reported. It has also shown that the open-loop solution for the general optimal control problem can be obtained using the same methods presented in this section, and some examples related to this class of optimal control problems are solved and reported. This chapter concludes with a flow-chart of the apriori polynomial fitting methods using the direct sensitivity approach.

Chapter seven is completely devoted to a special method which again uses the direct sensitivity approach.

However, in contrast to apriori polynomial fitting methods of the previous chapter, this method first finds out some data points for the coefficient vector function by using the basic idea behind the direct sensitivity approach, and then tries to fit a polynomial for these points. Therefore, this special method is named as "aposteriori polynomial fitting method". This aposteriori polynomial fitting method is also able to solve the general optimal control problem for both open-loop and closed-loop cases as apriori polynomial fitting method. Various optimal control problems with linear or nonlinear system dynamics and quadratic or nonquadratic cost functionals are solved using the aposteriori polynomial fitting method and the results are reported in this chapter. Next, a flow-chart of the method is given. This chapter concludes with the comparison of the apriori and aposteriori polynomial fitting methods.

This study, by no means, tries to justify the superfluousness of the dynamic programming or the development of Hamilton-Jacobi approach; rather, it proposes a few alternate approaches for the treatment of the same optimal control problem. It develops more convenient or computationally easy methods for the optimal control problem with nonlinear system dynamics and/or nonquadratic performance index. For the problem defined by a linear system dynamics

$$\dot{\underline{x}} = \underline{A}\underline{x} + \underline{B}\underline{u}$$

and with a quadratic performance index

$$PI = \int_0^{\infty} (\underline{x}^T \underline{\underline{Q}} \underline{x} + \underline{u}^T \underline{\underline{P}} \underline{u}) \, dt$$

where Q is symmetric positive semidefinite and P is symmetric positive definite matrices, the resultant Hamilton-Jacobi-Bellman equation can be solved in a reasonably simple manner. Therefore, this fact alone justifies the dynamic programming or the development of the Hamilton-Jacobi approach.

When we try to solve the optimal control problem with infinite horizon defined as above using the methods developed in this study we are faced with the problem of choosing sufficient final time in order to be able to do integration since the final time is designated to be infinity in the problem. Therefore, a method or a rule of thumb of determining the sufficient final time for the infinite horizon problem of the optimal control theory must be searched.

In chapter eight a method for determining the sufficient final time in case of infinite horizon is developed again using the sensitivity idea and some properties of the cost functionals for stable control problems. Several infinite horizon optimal control problems are solved using the new method compared with the results of Hamilton-Jacobi approach. This chapter again ends up with a flow-chart of the method.

Finally, the last chapter presents an overview and a comparative discussion of the methods and the approaches

developed in this research. This chapter also discusses some possible extensions and defines further areas of work.

II

A SURVEY ON THE NUMERICAL SOLUTIONS OF TWO-POINT BOUNDARY-VALUE PROBLEMS

2.1. INITIAL-VALUE PROBLEMS

Among the various techniques available for the analytical and numerical solution of boundary value problems for differential equations there is a number of methods which attack the given problem by solving instead certain related initial-value problems. In fact, most of the universally applicable numerical methods for solving two-point boundary-value problems somehow employ initial-value techniques. Therefore, the theory of boundary value (especially, two-point boundary-value) problems relies rather heavily on initial-value problems.

The theory of ordinary differential equations subject to initial conditions (i.e., initial value problems) is one of the most extensively developed branches of mathemathical analysis. Theorems on existence and uniqueness of solutions related to this topic are widely available in the literature. Here we will very briefly review some basic definitions and theorems on this topic.

Since every nth-order ordinary differential equation can be replaced by an equivalent system of n first-order equations, the attention can be confined to first-order

systems of the form

$$\dot{\underline{x}} = \frac{d\underline{x}}{dt} = \underline{f}\ (\underline{x};t) \qquad \text{.....} \quad \text{Eq.(2.1.1)}$$

Here $\underline{x} \equiv (x_1, x_2, \ldots x_n)^T$ is an n-dimensional column vector with the dependent variables $x_k(t)$ as components; then $\underline{x}(t)$ is a vector-valued function; $\underline{f}(\underline{x};t)$ is vector-valued with components $f_k(x_1, x_2, \ldots x_n;t)$, which are functions of the n+1 variables $(\underline{x};t)$. An initial-value problem for the above system is obtained by prescribing at some point, say t=a, the value of $\underline{x}$, say

$$\underline{x}\ (a) = \underline{\alpha} \qquad \text{.....} \quad \text{Eq.(2.1.2)}$$

The existence, uniqueness and continuity properties of the solutions of such problems depend on the continuity and/or smoothness properties of the function $\underline{f}$ in a neighborhood of the initial point (_;a). As a measure of distance between two points in n-space the maximum norm

$$\| \underline{x}-\underline{y} \| = \max_{1 \leq k \leq n} |x_k - y_k| \qquad \text{.....} \quad \text{Eq.(2.1.3)}$$

or the Euclidean norm

$$\| x-y \|_2 = \left[(x_1-y_1)^2 + \ldots + ((x_n-y_n)^2 \right]^{\frac{1}{2}} \qquad \text{Eq.(2.1.4)}$$

can be employed equally well. One of the basic results

can now be stated as follows.

THEOREM 2.1.1. : Let the function $\underline{f}(\underline{x};t)$ be continuous on the infinite strip

$$R : a \leq t \leq b \quad , \quad \|\underline{x}\| < \infty$$

and satisfy there a Lipschitz condition in $\underline{x}$ with constant K, uniformly in t; that is,

$$\|f(x;t) - f(y;t)\| \leq K \|\underline{x}-\underline{y}\| \quad \text{for all } (\underline{x};t) \text{ and } (\underline{y};t) \in R$$

Then

(a) the initial-value problem

$$\dot{\underline{x}} = \underline{f}(\underline{x};t) \qquad \underline{x}(a) = \underline{\alpha}$$

has a unique solution $\underline{x}=\underline{x}(\underline{\alpha};t)$ defined on the interval

$$[a,b] = \{t \mid a \leq t \leq b\} \quad ;$$

(b) this solution is Lipschitz-continuous in $\underline{\alpha}$, uniformly in t; in fact we have

$$\|\underline{x}(\underline{\alpha};t) - \underline{x}(\underline{\beta};t)\| \leq e^{K(t-a)} \cdot \|\underline{\alpha}-\underline{\beta}\| \quad \text{for all } (\underline{\alpha};t) \text{ and } (\underline{\beta};t) \in R$$

2.2. TWO-POINT BOUNDARY-VALUE PROBLEMS

A boundary-value problem for an ordinary differential equation (or system of equations) is obtained by requiring that the dependent variable (or variables) satisfy subsidiary conditions at two or more distinct points. By means of Theorem 2.1.1 we know that a unique solution of an nth-order equation is determined (for a very large class of equations) by specifying n conditions at one point (that is, for initial-value problems). However, with a total of n boundary conditions imposed at more than one point it is possible that a very smooth nth-order equation has many solutions or even no solution. Thus, as we may wxpect, the existence and uniqueness theory for boundary-value problems is considerably more complicated and less thoroughly developed than that for initial-value problems. When the boundary conditions are imposed at only two points, which is the usual case in many applications, a simple theory can be developed for many special classes of equations and systems of equations. This existence and uniqueness theory plays an important role in devising and analyzing numerical methods for solving boundary-value problems.

Therefore, some of the important aspects of the existence and uniqueness theory will be studied here with regard to a class of boundary-value problems in which the solution, $\underline{x}(t)$, of a second-order equation

$$\ddot{x} = \frac{d^2x}{dt^2} = f(x,\dot{x};t) \qquad \text{.....} \quad \text{Eq.(2.2.1a)}$$

is required to satisfy at two distinct points relations of the form

$$a_0 x(a) - a_1 \dot{x}(a) = \alpha \, |a_0| + |a_1| \neq 0$$
$$b_0 x(b) + b_1 \dot{x}(b) = \beta \, |b_0| + |b_1| \neq 0 \qquad \text{Eq.(2.2.1b)}$$

The solution is sought on the interval $[a,b]$.

A formal solution to the exact solution of this problem is obtained by considering a related initial-value problem, say

$$u = \frac{d^2 u}{dt^2} = f(u,\dot{u},t) \qquad \ldots\ldots\ldots \qquad \text{Eq.(2.2.2a)}$$

$$a_0 u(a) - a_1 \dot{u}(a) = \alpha \, , \; c_0 u(a) - c_1 \dot{u}(a) = s \qquad \text{Eq.(2.2.2b)}$$

The second initial condition is to be independent of the first. This is assured if $a_1 c_0 - a_0 c_1 \neq 0$. Without loss in generality it is required that c_0 and c_1 be chosen such that

$$a_1 c_0 - a_0 c_1 = 1 \qquad \ldots\ldots\ldots \qquad \text{Eq.(2.2.2c)}$$

With c_0 and c_1 fixed in this manner, the solution of Eq.(2.2.2) is denoted by

$$u = u\,(s;t)$$

to focus attention on its dependence on s. Evaluating the solution at t = b, a value of s is sought for which

$$\phi(s) \equiv b_0 u(s;b) + b_1 \dot{u}(s;b) - \beta = 0 \qquad \text{Eq.(2.2.3)}$$

With b and β fixed Equation (2.2.3) is, in general, a transcendental equation in s. If $s = s^*$ is a root of this equation, it is then expected that the function

$$x(t) \equiv u(s^*;t)$$

is a solution of the boundary-value problem.(2.2.1). This is true in many cases, and in fact all solutions of the problem (2.2.1) can frequently be determined in this way. To be precise, the following theorem can be stated.

THEOREM 2.2.1 : Let the function $f(u_1,u_2,t)$ be continuous on

$$R : a \leq t \leq b \quad , \quad u_1^2 + u_2^2 < \infty$$

and satisfy there a uniform Lipschitz condition in u_1 and u_2. Then the boundary-value problem (2.2.1) has as many solutions as there are distinct roots, $s = s^{(\nu)}$, of Equation (2.2.3). The solutions of (2.2.1) are

$$x(t) = x^{(\nu)}(t) \equiv u(s^{(\nu)};t)$$

that is, the solutions of the initial-value problem (2.2.2)

with initial data $s = s^{(\gamma)}$.

For the proof of the above theorem one may refer to Keller (Ref.4). By means of this theorem the problem of solving a boundary-value problem is "reduced" to that of finding the root, or roots, of an (in general, transcendental) equation. In fact, more general boundary-value problems than (2.2.1) can be reduced in this way to solving systems of (transcendental) equations.

Moreover, there is an important class of problems for which it can be assured that Equation (2.2.3) has a unique root. The existence and uniqueness theory for the corresponding boundary-value problems is then settled.

THEOREM 2.2.2 : Let the function $f(u_1,u_2,t)$ in Equation (2.2.1a) satisfy the hypothesis of Theorem (2.2.1) and have continuous derivatives on R which satisfy, for some positive constant M,

$$\frac{\partial f}{\partial u_1} > 0 \quad , \quad \left|\frac{\partial f}{\partial u_2}\right| \leq M$$

Let the coefficients in Equation (2.2.1b) satisfy

$$a_0 a_1 \geq 0 \quad , \quad b_0 b_1 \geq 0 \quad , \quad |a_0| + |b_0| \neq 0$$

Then the boundary-value problem (2.2.1) has a unique solution.

One may again refer to Keller (Ref.4) for the proof of the above theorem.

This much consideration of existence and uniqueness theory on boundary-value problems is sufficient for our purposes at this point.

2.3. NUMERICAL METHODS

There exist a wide variety of methods with overlapping or interacting ideas for solving boundary-value problems in the literature. Therefore, it is very difficult to make a precise classification of the existing methods. However, still all of the methods can be classified very broadly under three major headings some of which may contain subgroups (or subheadings). These three major headings are as follows :

A. SHOOTING METHODS (OR INITIAL-VALUE METHODS)

B. FINITE-DIFFERENCE METHODS

C. FUNCTION SPACE APPROXIMATION (OR PROJECTION) METHODS

Let us now have a brief overview of the above methods.

A. SHOOTING (OR INITIAL-VALUE) METHODS

Shooting methods are so natural and commonly used for treating boundary-value problems for ordinary differential equations that many papers employ them without an explicit statement of the fact. In the previous section we have seen that by means of Theorem (2.2.1) the problem of solving a boundary-value problem is reduced to that of finding the root, or roots, of an (in general, transcenden-

tal) equation. A very effective class of numerical methods, which we call shooting (or initial-value) methods, is based on this equivalence.

Let us consider a rather general nonlinear boundary-value problem

$$\dot{\underline{x}} = \frac{d\underline{x}}{dt} = \underline{f}(\underline{x},t) \qquad a \leq t \leq b \quad \ldots\ldots \quad \text{Eq.(2.3.1a)}$$

$$\underline{g}\ (\underline{x}(a),\ \underline{x}(b)) = \underline{0} \qquad \ldots\ldots\ldots\ldots \quad \text{Eq.(2.3.1b)}$$

The above boundary-value problem (2.3.1) is associated with the following initial-value problem.

$$\underline{u} = \frac{d\underline{u}}{dt} = \underline{f}\ (\underline{u},t) \qquad \ldots\ldots\ldots\ldots \quad \text{Eq.(2.3.2a)}$$

$$\underline{u}\ (a) = \underline{s} \qquad \ldots\ldots\ldots\ldots \quad \text{Eq.(2.3.2b)}$$

A solution $\underline{u} = \underline{u}(\underline{s},t)$ of the problem (2.3.2) is a solution of the problem (2.3.1) if $\underline{s}$ is a root of

$$\underline{\phi}(\underline{s}) = \underline{g}(\underline{s},\underline{u}(\underline{s},b)) = 0 \qquad \ldots\ldots\ldots \quad \text{Eq.(2.3.3)}$$

A fairly general theory of this process, using arbitrary stable accurate of $0(h^p)$ initial-value methods is developed in Keller (Ref.4). Eventhough, there may exist various root-finding schemes available in the literature, Newton's method or its alternatives or modifications of it is perhaps

the most commonly adopted scheme to be used on Equation (2.3.3).

Now, there remain three main topics that will be considered very briefly. First topic is related with the important question of how to pick the initial iterate, $\underline{s}^{(0)}$, and this automatically introduces the continuity or continuation studies. Next the standard question of unstable growth of the solution of the problem (2.3.2) leading to parallel- or multiple-shooting methods is considered. Finally invariant-imbedding which can also be considered as a special type of initial-value (or shooting) method will be discussed.

CONTINUATION

Selection of an appropriate initial iterate, $\underline{s}^{(0)}$, so that convergence to the desired root of Equation (2.3.3) occurs in whatever iteration scheme is being employed, is one of the basic open questions in shooting for nonlinear problems. A fairly general form of continuation consists in embedding the problem (2.3.1) in a family of problems

$$\dot{\underline{z}} = \frac{d\underline{z}}{dt} = \underline{f}\ (\underline{z}, t; \sigma) \qquad \text{..... Eq.(2.3.4a)}$$

$$\underline{g}\ (\underline{z}(a), \underline{z}(b); \sigma) = 0 \qquad \text{..... Eq.(2.3.4b)}$$

which for $\sigma = \sigma_F$ say, reduces to the problem (2.3.1). Further the problem (2.3.4) for $\sigma = \sigma_0$ has a known solution (or is "easily" solved). The idea is to compute

$\underline{z}(t; \sigma_F) \equiv \underline{x}(t)$ starting from the known solution $\underline{z}(t; \sigma_0)$ by continuation in the imbedding parameter, .

PARALLEL- OR MULTIPLE-SHOOTING

Parallel-shooting is employed for reducing the destabilizing effects of growing solutions of the initial-value problems. The basic idea in parallel-shooting is to partition the interval [a,b] into subintervals and to compute the solution over each subinterval (more or less) independently of the results in the other subintervals. Then simultaneously with attempting to satisfy the boundary conditions the relevant continuity conditions are imposed at each interval interface.

INVARIANT-IMBEDDING

Here the invariant imbedding method is considered to be a shooting-method because the resultant invariant imbedding equations represent an initial-value problem. The method of invariant imbedding which originated in 1957 with a series of papers by Bellman, Kalaba and Wing is actually an outgrowth of dynamic programming. Basically, the method involves generating a "family" of problems by means of a single parameter, where the basic properties of the system remain invariant under the generation of the family. The family then provides a means of advancing from one member, sometimes degenerate, to the solution of the original problem. In case of boundary-value problems the crucial parameter is taken to be the interval length.

We shall now present a derivation of the invariant imbedding equations. Let us consider the system of nonlinear ordinary differential equations

$$\dot{u} = F(u,v,t) \quad , \qquad \text{..... Eq.(2.3.5a)}$$

$$\dot{v} = G(u,v,t) \quad , \quad 0 < t < T \quad , \qquad \text{Eq.(2.3.5b)}$$

subject to the simple separated boundary conditions

$$u(0) = 0 \qquad \text{..... Eq.(2.3.5c)}$$

$$v(T) = c \qquad \text{..... Eq.(2.3.5d)}$$

For the sake of exposition, we assume that u and v are scalar functions. The multidimensional versions of the following results can be readily obtained.

By differentiating Equations (2.3.5a) - (2.3.5d) with respect to c, it is seen that

$$\dot{u}_c(t,c,T) = F_u u_c + F_v v_c \qquad \text{..... Eq.(2.3.6a)}$$

$$\dot{v}_c(t,c,T) = G_u u_c + G_v v_c \quad , \qquad \text{..... Eq.(2.3.6b)}$$

$$0 < t < T$$

$$u_c(0,c,T) = 0 \qquad \text{..... Eq.(2.3.6c)}$$

$$v_c(T,c,T) = 1 \qquad \text{..... Eq.(2.3.6d)}$$

Similarly, a differentiation in T yields

$$\dot{u}_T = F_u u_T + F_v v_T \qquad \text{..... Eq.(2.3.7a)}$$

$$\dot{v}_T = G_u u_T + G_v v_T \quad , \qquad \text{..... Eq.(2.3.7b)}$$
$$0 < t < T$$

$$u_T(0,c,T) = 0 \qquad \text{..... Eq.(2.3.7c)}$$

$$\dot{v}(T,c,T) + v_T(T,c,T) = 0 \qquad \text{..... Eq.(2.3.7d)}$$

To make use of these equations, note that from the differential equation (2.3.6b), when t =T we have

$$\begin{aligned}\dot{v}(T,c,T) &= G(u(T,c,T),v(T,c,T),T) \\ &= G(r(c,T),c,T) \qquad \text{.... Eq.(2.3.8)}\end{aligned}$$

where the notation

$$r(c,T) = u(T,c,T) \qquad \text{..... Eq.(2.3.9)}$$

has been introduced. Comparing Equations (2.3.6a) - (2.3.6d) with (2.3.7a) - (2.3.7d), and assuming a unique solution exists, it follows that

$$u_T(t,c,T) = -G(r(c,T),c,T)\, u_c(t,c,T) \qquad \text{Eq.(2.3.10a)}$$

$$v_T(t,c,T) = -G(r(c,T),c,T)\, v_c(t,c,T)$$
$$0 \leq t \leq T \quad , \quad |c| < \infty \qquad \text{Eq.(2.3.10b)}$$

Equations (2.3.10a) and (2.3.10b) are the desired partial differential equations for u and v. The initial conditions at T = t are

$$u(t,c,t) = r(c,t) \qquad \text{..... Eq.(2.3.10c)}$$

$$v(t,c,t) = c \qquad \text{..... Eq.(2.3.10d)}$$

It remains to consider the function r.

Differentiate Equation (2.3.9) with respect to T to obtain

$$r_T(c,T) = \dot{u}(T,c,T) + u_T(T,c,T) \qquad \text{..... Eq.(2.3.11)}$$

From Equations (2.3.5a) and (2.3.10a), we now see that

$$r_T(c,T) = F(r(c,T),c,T) - G(r(c,T),c,T)r_c(c,T) \qquad \text{Eq.(2.3.12a)}$$

This is the quasilinear first-order partial differential equation satisfied by r. From Equation (2.3.5c) we see that

$$r(c,0) = 0 \qquad \text{..... Eq.(2.3.12b)}$$

The equations for u, v and r, together with their initial conditions, constitute the initial value representation for the original nonlinear problem. Either some finite difference scheme must be developed for the solution of the above quasi-

linear first-order partial differential equation with an initial condition on r, or a method which replaces the nonlinear boundary-value problem by a sequence of linear boundary-value problems (which hopefully will converge to the solution of the nonlinear problem) must be introduced. The second choice (or the method) which is called quasilinearization has been used very effectively for solving certain important classes of nonlinear boundary-value problems, because the invariant imbedding equations derived for a linear boundary-value problem is much easier from computational point of view. These special equations are known as Riccati differential equations, and there exist many efficient methods developed for the soution of this kind of differential equations.

B. FINITE DIFFERENCE METHODS

In order to explain the basic idea behind these methods let us consider the general systems of n first-order equations subject to linear two-point boundary conditions

$$L\underline{x} \equiv \dot{\underline{x}} - \underline{f}(\underline{x},t) = 0 \qquad \text{..... Eq.(2.3.13a)}$$

$$\underline{\underline{A}}\underline{x}(a) + \underline{\underline{B}}\underline{x}(b) = \underline{\alpha} \qquad \text{..... Eq.(2.3.13b)}$$

In the present discussion a uniform net will be employed merely for notational convenience on [a,b] as

$$t_j = a+jh \ , \ j=0,1,\ldots N \ ; \ h= \frac{b-a}{N} \qquad \text{Eq.(2.3.14)}$$

The n-dimensional vectors $\underline{u}_j$ will denote approximations to the corresponding values of the solution $x(t_j)$ of Equation (2.3.13a) at the points of our net. One obvious system of difference equations for the determination of these approximations is

$$L_n \underline{u}_j = \frac{\underline{u}_j - \underline{u}_{j-1}}{h} - \underline{f}\left(\frac{\underline{u}_j + \underline{u}_{j-1}}{2}, t_{j-\frac{1}{2}}\right) = 0 \quad j=1,2,\ldots N \qquad \text{Eq.(2.3.15a)}$$

and the boundary conditions become

$$\underline{\underline{A}}\underline{u}_0 + \underline{\underline{B}}\underline{u}_N - \underline{\alpha} = \underline{0} \qquad \ldots\ldots\ldots \text{Eq.(2.3.15b)}$$

The scheme in Equation (2.3.15a) is known as the centered-difference method when used for the Equation (2.3.13a) subject to initial conditions. The nonlinear term in Equation (2.3.15a) might have been chosen as

$$\tfrac{1}{2} \quad f(u_j, t_j) + f(u_{j-1}, t_{j-1})$$

and the resulting scheme is called the modified Euler method.

The Equations (2.3.15), of N+1 sets of n equations each, are the difference equations whose solution is to approximate that of (2.3.13) on the net. We can write these difference equations in a more uniform and compact form. Let the n(N+1)-dimensional vector $\underline{U}$ be defined by

$$\underline{U} \equiv \begin{bmatrix} \underline{u}_0 \\ \underline{u}_1 \\ \vdots \\ u_N \end{bmatrix}$$

Then Equations (2.3.15) can be written as the system of n(N+1) equations

$$\underline{0}(\underline{U}) \equiv \begin{bmatrix} \underline{\underline{A}}\underline{u}_0 + \underline{\underline{B}}\underline{u}_N - \alpha \\ hL_h\underline{u}_1 \\ \vdots \\ hL_h\underline{u}_N \end{bmatrix} = \underline{0} \quad \text{.....} \quad \text{Eq.(2.3.16)}$$

We now see one basic difference, at least in point of view, between the initial-value methods and the finite-difference methods. In initial-value methods some unknowns, the initial values, are somehow determined recursively so as to be accurate approximations to solutions of the differential equations, and only when the last variables are computed are the boundary conditions employed. In finite-difference schemes no particular variables are preferred and the differential equations and boundary conditions are presumably treated simulataneously. In some iterative attempts at solving the system (2.3.16) one might proceed recursively guessing at $\underline{u}_0$, say, then solving the equations in (2.3.15a), exactly or approximately, in the order j=1,2,...N and finally checking (2.3.15b) to change the value of $\underline{u}_0$.

C. FUNCTION SPACE APPROXIMATION METHODS

These methods are expansion procedures for which the theoretical justification is considerably more difficult and less well developed. More specifically, the solution is approximated by a linear combination of linearly-independent functions in an appropriate function space. The coefficients

in the expansion are to be determined so that this combination minimizes some measure of the error in satisfying the boundary-value problem. There is tremendous variety in the choice of approximating functions and in the choice of "measure of error" in satisfying the problem.

Rayleigh-Ritz, Galerkin and Collocation are the most popular or known ones of the general function space approximation methods.

2.4. TPBVP IN OPTIMAL CONTROL THEORY

In general, in optimization problems for dynamic systems, whether the system under consideration is continuous or multi-stage or single-state discrete, it is finally encountered with Two-Point Boundary-Value Problems (TPBVP). More specifically, in optimal control theory, one of the basic problems is to find a control function or a control-law which will minimize a certain performance index while satisfying the state equation constraints. Formally, the basic (and the most general) optimal control problem can be described by a set of differential equations of the form

$$\dot{\underline{x}} = \underline{f}\ (\underline{x},\underline{u},t) \qquad \text{..... Eq.(2.4.1)}$$

and a performance index defined as follows

$$PI = \varphi(\underline{x}(t_f),t_f) + \int_{t_i}^{t_f} 1(\underline{x}(t),\underline{u}(t),t)\ dt \qquad \text{Eq.(2.4.2)}$$

which is tried to be minimized by a certain control function $\underline{u}(t)$, or the optimal-control law $\underline{u}=\underline{k}(\underline{x}(t),t)$ which is required to be a member of a set U called the control region. U may be either open or closed and bounded or unbounded. In this formulation, initial state and time are fixed; that is, $\underline{x}(t_i)$ and t_i are given; the terminal time t_f may be fixed or free; and the terminal state $\underline{x}(t_f)$ may be fixed, completely free or specified by a set of relations of the form

$$\underline{g}_i(\underline{x}(t_f),t_f) = \underline{0} \qquad i=1,2,\ldots m \quad n \qquad \text{Eq.(2.4.3)}$$

The solution of this general optimal control problem using the Pontryagin's minimum principle and calculus of variations leads us to solve the set of 2n equations.

$$\dot{\underline{x}} = \frac{\partial H^0}{\partial \underline{p}} (\underline{x},\underline{p},t) \qquad \ldots\ldots\ldots \qquad \text{Eq.(2.4.4)}$$

$$\dot{\underline{p}} = \frac{\partial H^0}{\partial \underline{x}} (\underline{x},\underline{p},t) \qquad \ldots\ldots\ldots \qquad \text{Eq.(2.4.5)}$$

with the given initial and terminal boundary conditions and generalized boundary condition

$$\left[\frac{\partial \varphi}{\partial \underline{x}} (\underline{x},t) - \underline{p}\right]^T .d\underline{x} \Bigg|_{t_f} + \left[H^0(\underline{x},\underline{p},t) + \frac{\partial \varphi}{\partial t}(\underline{x},t)\right] .dt \Bigg|_{t_f} = 0$$

Eq.(2.4.6)

where

$$H^0(\underline{x},\underline{p},t) = H(\underline{x},\underline{u}^0,\underline{p},t) = \min_{u \in U} H(\underline{x},\underline{u},\underline{p},t) \qquad \text{Eq.(2.4.7)}$$

and

$$H(\underline{x},\underline{u},\underline{p},t) = l(\underline{x},\underline{u},t) + \underline{p}^T \cdot \underline{f}(\underline{x},\underline{u},t) \qquad \text{Eq.(2.4.8)}$$

Unless the system equations, the performance index and the constraints are quite simple, numerical methods are required to solve the two-point boundary-value problems. All numerical methods for the solution of such problems involve either dynamic programming or iterative procedures.

Dynamic programming, as applied to two-point boundary-value problems, can be described as a process of generating many solutions satisfying the specified boundary conditions as parameters. If the suitable or correct range of parameters is chosen, some of the solutions will pass through (or near) the desired boundary conditions at the other end. This method is not feasible for problems with two or three state variables even on the larger computers of our day.

There exist several different ways of treating the same problem with iterative procedures, and only three of these possible procedures have been extensively used. Almost all of the iterative procedures use "successive linearization" while only a small portion of methods use transformations for instance Riccati-type transformations.

Stating in words, the nonlinear TPBVP is to find

(a) the n state variables $\underline{x}(t)$

(b) the n costate functions $\underline{p}(t)$

(c) the m control functions $\underline{u}(t)$

and satisfy simulataneously

(1) the n system of differential equations (involving $\underline{x},\underline{u}$)

(2) the n costate differential equations (involving $\underline{p},\underline{x},\underline{u}$)

(3) the m optimality conditions (involving $\underline{p},\underline{x},\underline{u}$)

(4) the initial and final boundary conditions (involving $\underline{x},\underline{p}$)

One of the three iterative procedures so far studied is neighboring extremal method (or variation of extremals). In this method, a nominal solution is chosen which satisfies the three conditions (1) through (3) above; then this nominal condition is modified by succesive linearization so that the remaining boundary conditions are also satisfied. When using neighboring extremal methods and quasilinearization methods, we must solve a succession of linear two-point boundary-value problems. Such problems can be solved by (a) finding the transition matrix between unspecified boundary conditions at one end and specified boundary conditions at the other end, or by (b) "sweeping" the boundary conditions from one end point to the other end point, which involves solving a matrix Riccati equation.

The main difficulty with neighboring extremal methods is getting started; i.e., finding a first estimate of the unspecified conditions at one end that produces a solution reasonably close to the specified conditions at the other end. The reason for this peculiar difficulty is that extremal solutions are often very sensitive to small changes in the unspecified boundary conditions. This extraordinary sensitivity is a direct result of the nature of the costate function equations.

The second method is gradient method which is developed to surmount the "initial guess" difficulty associated with the

extremal methods. In these methods, the chosen nominal solution satisfies system equations and costate equations. These methods are characterized by iterative algorithms for improving estimates of the control function, u(t), so as to come closer to satisfying the optimality condition. The drawback of first-order gradient method is its poor convergence near the optimal solution region. The second-order gradient method has solved this problem, but may have starting difficulties since the nominal solution to be chosen has to be "convex".

The quasilinearization methods involve choosing nominal functions for $\underline{x}(t)$ and $\underline{p}(t)$ that satisfy as many of the boundary conditions as possible. The system equations and costate equations are linearized about the nominal and a succession of nonhomogeneous, linear two-point boundary-value problems are solved to modify the solution until it satisfies the system and costate equations to the desired accuracy. These methods are more attractive when compared to other methods. First it is often easier to guess nominal state variable histories than control variable histories. Second, these methods converge rapidly near the optimum solution.

Other than the methods mentioned above, there exist the differential dynamic programming approach which was first introduced into optimal control by Mayne and was later developed for continuous systems by Jacobson. This approach is a successive approximation technique, based on dynamic programmin rather than the calculus of variations, for determining optimal control of nonlinear systems. In each iteration the system

equations are integrated in forward time using the current nominal control; and costate equations, which yield the coefficients of a linear or quadratic expansion of the cost function in the neighborhood of the nominal x trajectory, are integrated in reverse time, thusyielding an improved control law. This control is applied to the system equations, producing a new and improved trajectory. By continued iteration, the procedure produces control functions that successively approximate the optimal control function.

III

SENSITIVITY APPROACH TO TWO-POINT BOUNDARY-VALUE PROBLEMS

3,1, A BRIEF HISTORICAL REVIEW TO SENSITIVITY THEORY

Sensitivity considerations have long been of concern in connection with dynamic systems. Historically, these sensitivity considerations have provided a fundamental motivation for the use of feedback and are largely responsible for its development into what is called modern control theory, implying the principles of optimization and adaptation. Therefore, it is quite natural that the basic concepts in this area were already given in the fundamental literature on feedback control systems thirty years ago. Bode was the first to establish the significance of sensitivity in the design of feedback control systems. He has introduced a proper sensitivity definition on the basis of frequency domain.

In its subsequent development it seemed that automatic control theory should include the study of sensitivity as an essential component. However, with few exceptions, the sensitivity problem was not even discussed in the academic texts on automatic control in the following decade. It was mainly the problem of accuracy in network-analyzers and analog computers that gave new impulses to the theory of sensitivity during the fifties. Many basic methods were also worked out in connection with the design of electric networks. Toward the end of this period the ideas of Bode were rediscovered in control engineeri

with the appearance of adaptive systems, more precisely, as a reaction to their appearance. Horowitz has developed the methods of frequency domain to a high extent and has applied them with great success to the design of low sensitivity conventional feedback control systems.

Beginning in the period 1958-1960, the number of publications in the time domain rose considerably due to the development of state space methods in control engineering and the availability of the digital computer.

3.2. BASIC CONCEPTS AND DEFINITIONS IN SYSTEM THEORY

Sensitivity theory can be interpreted as a section of a general system theory, taking into consideration parameter variations as inputs instead of signals. From a mathematical point of view, what we call a system is the explicitly or implicitly given relationship between the input signal $\underline{u}(t)$ and the output signal $\underline{y}(t)$. In general, $\underline{u}(t)$ and $\underline{y}(t)$ can be vectors. The character of this relationship is commonly called the structure of the system. For example, the structure of the system may be characterized by

(a) the order of a differential or difference equation,

(b) linearity or nonlinearity,

(c) the order of the numerator and denominator of a rational transfer function,

and

(d) the rationality or irrationality of the transfer function.

The quantitative properties of the system are characterized by the system parameters. Typical parameters are

(1) initial conditions,

(2) time-invariant or time-variant coefficients,

(3) natural frequencies, pulse frequencies,

(4) sampling periods, sampling instants,

(5) pulse width or magnitude, and

(6) dead times (or time delays).

Dynamic processes in a system, say, the change of the state or of the output variable with time, can be caused by

(1) the inlfuence of input signals,

(2) the change of parameters.

While studying the influence of input signals, the dynamics of the system are usually considered only as a function of the input signals, assuming that the relationship is qualitatively and quantitatively unchanged. This is the subject matter of conventional system theory.

While studying the influence of parameters, the dynamics of the system are considered as a function of changes in the parameters (or of the structure of the system, because the change of system parameters can also change the system structure). The dependence of the system dynamics on the parameters is called sensitivity. Strictly, parameter sensitivity can be defined as follows:

Definition 3.2.1.: Parameter sensitivity is the effect of parameter changes on the dynamics of a system, say the time response, the state, the transfer function, or any other quantity characterizing the system dynamics.

The mathematical problem to be solved in sensitivity theory is the calculation of the change in the system behavior

due to the parameter variations. Let the parameters of the system be represented by a vector $\underline{\alpha} = [\alpha_1 \; \alpha_2 \; \ldots \; \alpha_r]^T$. The mathematical model of a system relates the parameter vector $\underline{\alpha}$ to a quantity characterizing its dynamic behavior in some way. The characterizing quantity in case of a dynamic system will be the state vector $\underline{x}$.

Let us explain the basic idea of the sensitivity theory by means of this example. It is assumed that the mathematical model of the (possibly nonlinear) system is given by the general vector differential equation

$$\dot{\underline{x}} = \underline{f}(\underline{x}, \underline{\alpha}, t, \underline{u}) \; , \quad \underline{x}(t_0) = \underline{x}^0 \qquad \text{Eq.(3.2.1)}$$

where $\underline{x}$ represents the state vector with the initial state $\underline{x}(t_0)=\underline{x}^0$, and $\underline{u}$ represents the input vector. Among other things, this equation relates the state vector $\underline{x}$ to the parameter vector $\underline{\alpha}$. In terms of set theory, this relation can also be interpreted as a mapping $\underline{\alpha} \longrightarrow \underline{x}$.

Generally, in mathematics, a unique relationship between the parameter vector and the state vector is assumed. However, this is not possible in engineering practice. Here the parameter vector of the mathematical model means a nominal parameter vector that will be denoted by $\underline{\alpha}_0$ in the sequel, whereas the parameter vector of the actual system is $\underline{\alpha} = \underline{\alpha}_0 + \Delta\underline{\alpha}$, in the sequel called the actual parameter vector.

In order to study the influence of the parameter deviations $\Delta\underline{\alpha}$ on the behavior of the system, let us define

R_α as the subspace of the parameter variations $\Delta \underline{\alpha}$ around $\underline{\alpha}_0$, and

R_x as the corresponding subspace of the state vector.

By this definition the mapping $\underline{\alpha} \longrightarrow \underline{x}$ can be replaced by the mapping $R_\alpha \longrightarrow R_x$ as shown in Fig. 3.2.1.

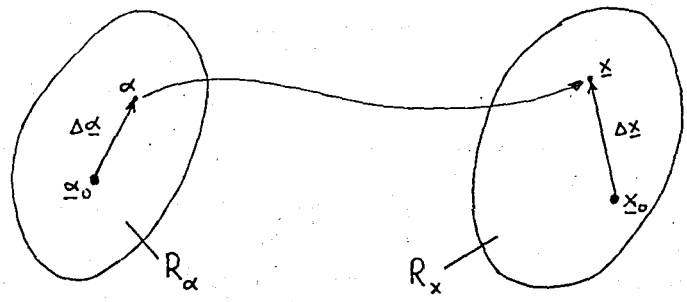

FIGURE 3.2.1. Mapping of the parameter space into the state space.

R_x is uniquely determined by Equation (3.2.1) if R_α is known. However, for a number of reasons, it is not reasonable to characterize the sensitivity in terms of Equation (3.2.1): first, since the direct solution of Equation (3.2.1) for all elements of R_α requires an infinite number of solutions and depends on the definition of R_α, and second, since the result for small parameter variations $\|\Delta \underline{\alpha}\| \ll \|\underline{\alpha}_0\|$ would be very inaccurate if approximations are applied for the evaluation if this equation. For example, this would be true in the case of numerical or analog computation.

Therefore, it is a common practice in sensitivity theory to define a so-called sensitivity function $\underline{\underline{S}}$ which, under certain continuity conditions, relates the elements of the set of the parameter deviations $\Delta \underline{\alpha}$ to the elements of the set of the parameter-induced errors of the system function $\Delta \underline{x}$ by the linear equation

$$\Delta \underline{x} \cong \underline{\underline{S}}(\underline{\alpha}_0)\,\Delta \underline{\alpha} \qquad \text{..... Eq.(3.2.2)}$$

Actually there are several ways to define quantities for the characterization of the parameter sensitivity of a system. One of these definitions which is frequently used in the sequel will be given below.

Let the behavior of the dynamic system be characterized by a quantity $\underline{\xi}(\underline{\alpha})$, called a system function, which, among other dependences, is a function of the parameter vector $\underline{\alpha} = [\underline{\alpha}_1 \;\; \underline{\alpha}_2 \;\ldots\; \underline{\alpha}_r]^T$. For example, $\underline{\xi}$ can represent any time domain or frequency domain property or a performance index.

Definition 3.2.2.: Absolute sensitivity function.

$$\underline{\underline{S}} \triangleq \left.\frac{\partial \underline{\xi}}{\partial \underline{\alpha}}\right|_{\alpha_0}$$

3.3. MOTIVATION FOR SENSITIVITY APPROACH TO TPBVP

The motivation for using sensitivity function (or a matrix) in TPBVP stemmed from the fact that almost all of the methods for solving this problem have the difficulty of "getting started" or "initial guess"; and the reason is the extraordinary sensitivity nature of the Euler-Lagrange equations. In order to eliminate this difficulty we thought that it was necessary to know the sensitivity of the dynamic system to the changes in the unspecified boundary condition (in our case initial costate function values). If we had known this sensitivit function of the system under consideration, then it would have

been possible to change the unspecified boundary conditions in a rational fashion, i.e., if the system or equivalently the differential equations describing it has high sensitivity, them the incremental addition to the unspecified boundary condition will have to be smaller or if the system has low sensitivity, then this addition will have to be larger; because high sensitivity means that a small change in the unspecified boundary conditions induces a large change at the other end, so we will have to take smaller steps.

3.4. THE TRAJECTORY SENSITIVITY FUNCTION OF CONTINUOUS SYSTEMS

The new method which will be proposed for the solution of general nonlinear two-point boundary-value problems will employ trajectory sensitivity function as a crucial tool in its development. Therefore, at this point, let us define this special sensitivity function with relation to continuous systems.

A continuous, possibly nonlinear system of nth-order can, in general, be described in the state space by a vector differential equation of the form

$$\underline{\dot{x}} = \underline{f}(\underline{x}, \underline{u}, \underline{\alpha}_0, t) \quad , \quad \underline{x}(t_0) = \underline{x}_0 \qquad \text{Eq.(3.4.1)}$$

Here $\underline{x}$ is an nx1 state vector, $\underline{f}$ an nx1 vector function, $\underline{u}$ an input vector, $\underline{\alpha}_0$ a nominal rx1 parameter vector, and $\underline{x}_0$ is the nx1 initial state vector. Equation (3.4.1) is called the nominal state equation.

Assuming that the parameter vector deviates from the nominal value $\underline{\alpha}_0$ by $\Delta\underline{\alpha}$, we have

$$\dot{\underline{x}} = \underline{f}(\underline{x}, \underline{u}, \underline{\alpha}, t) \qquad \underline{x}(t_0) = \underline{x}_0 \qquad \text{Eq.(3.4.2)}$$

This equation is called the actual state equation.

Now it is assumed that Equation (3.4.2) has a unique solution $\underline{x}=\underline{x}(\underline{\alpha},t)$ for all admissible initial conditions and parameter values. $\underline{x}$ is of course a function of $\underline{u}$, $\underline{x}_0$ and t_0 as well. However, this dependence is not needed for the following considerations and will, therefore, be dropped for ease of notation. Furthermore, the solution $\underline{x}$ is assumed to be a bounded continuous function in $\underline{\alpha}$ and t. It is known that this property is guaranteed if $\underline{f}$ is a bounded continuous function satisfying the Lipschitz condition (Refer to Theorem 2.1.1).

If the parameter takes on its nominal value $\underline{\alpha}_0$, the nominal solution $\underline{x}_n=x(\underline{\alpha}_0,t)$ is obtained. If, on the other hand, the actual solution is given by $\underline{x}=\underline{x}(\underline{\alpha},t)$, then the parameter-induced change of the state vector is defined as

$$\Delta\underline{x}(\underline{\alpha},t) \triangleq \underline{x}(\underline{\alpha},t) - \underline{x}(\underline{\alpha}_0,t) \qquad \text{Eq.(3.4.3)}$$

A first-order approximation of $\Delta\underline{x}$ can be written by use of a Taylor expansion in the form

$$\Delta\underline{x}(\underline{\alpha},t) = \left.\frac{\partial \underline{x}}{\partial \underline{\alpha}}\right|_{\underline{\alpha}_0} \cdot \Delta\underline{\alpha} + O(\Delta\alpha) \qquad \text{Eq.(3.4.4)}$$

This equation can be viewed as a definition of the parameter-

induced trajectory deviation. Now we can state the following important definition.

Definition 3.4.2.: Trajectory Sensitivity Matrix.

$$\underline{\underline{S}} \triangleq \left.\frac{\partial \underline{x}}{\partial \underline{\alpha}}\right|_{\underline{\alpha}_0} = \left[\begin{array}{ccc} \frac{\partial x_1}{\partial \alpha_1} & \cdots & \frac{\partial x_1}{\partial \alpha_r} \\ \vdots & & \vdots \\ \frac{\partial x_n}{\partial \alpha_1} & \cdots & \frac{\partial x_n}{\partial \alpha_r} \end{array}\right]_{\underline{\alpha}_0} \qquad \text{Eq.(3.4.5)}$$

3.5. SENSITIVITY APPROACH TO THE SOLUTION OF TWO-POINT BOUNDARY-VALUE PROBLEMS

In this section a new method for the numerical solution of two-point boundary-value problems will be presented. The method requires the assignment of arbitrary initial conditions for the variable that are specified at the final time. Then, the problem is solved iteratively, based on trajectory sensitivities with respect to the initial conditions.

Consider a two-point boundary-value problem for n-vector $\underline{x}(t)$ and m-vector $\underline{y}(t)$.

$$\dot{\underline{x}} = \underline{f}(\underline{x},\underline{y}) \quad , \quad \underline{y} = \underline{h}(\underline{x},\underline{y}) \qquad \text{Eq.(3.5.1)}$$

$$\underline{x}(0) = \underline{a} \quad , \quad \underline{y}(T) = \underline{b} \; . \qquad \text{Eq.(3.5.2)}$$

Equation (3.5.2) shows that n conditions are prescribed at the initial time t=0 and m conditions at the final time t=T. The

separation of boundary conditions in time makes this problem much more difficult to solve than a comparable initial-value problem.

Suppose an initial guess for $\underline{y}(0)$ is assigned as follows

$$\underline{x}(0) = \underline{a} \qquad \underline{y}(0) = \underline{c} \qquad \ldots\ldots\ldots \text{Eq.(3.5.3)}$$

This is an initial-value problem that can be integrated much more easily than the original problem. However, $\underline{y}(T)$ calculated using Equation (3.5.3) will in general be different from the desired boundary value $\underline{b}$. The aim of the iterations will be to make $\underline{y}(T)=\underline{b}$.

Suppose small changes in the m-vector $\underline{c}$ are introduced. The effect of such changes is predicted by trajectory sensitivities, which can be calculated while the initial-value problem is being integrated.

Define sensitivity matrices as follows

$$\underline{\underline{R}}(t) \triangleq \frac{\partial \underline{x}(t)}{\partial \underline{c}} = \begin{bmatrix} \dfrac{\partial x_1(t)}{\partial c_1} & \cdots & \dfrac{\partial x_1(t)}{\partial c_m} \\ \vdots & & \vdots \\ \dfrac{\partial x_n(t)}{\partial c_1} & \cdots & \dfrac{\partial x_n(t)}{\partial c_m} \end{bmatrix} \text{Eq.(3.5.4a)}$$

and

$$\underline{\underline{S}}(t) \triangleq \frac{\partial \underline{y}(t)}{\partial \underline{c}} = \begin{bmatrix} \dfrac{\partial y_1(t)}{\partial c_1} & \cdots & \dfrac{\partial y_1(t)}{\partial c_m} \\ \vdots & & \vdots \\ \dfrac{\partial y_m(t)}{\partial c_1} & \cdots & \dfrac{\partial y_m(t)}{\partial c_m} \end{bmatrix} \text{Eq.(3.5.4b)}$$

Then we obtain two sets of linear differential equations by taking the partial derivative of the Equations (3.5.1) and (3.5.3) with respect to $\underline{c}$

$$\frac{\partial \dot{\underline{x}}}{\partial \underline{c}} = \frac{\partial \underline{f}}{\partial \underline{x}} \frac{\partial \underline{x}}{\partial \underline{c}} + \frac{\partial \underline{f}}{\partial \underline{y}} \frac{\partial \underline{y}}{\partial \underline{c}} \quad , \quad \frac{\partial \underline{x}(0)}{\partial \underline{c}} = \underline{\underline{0}}$$

$$\frac{\partial \dot{\underline{y}}}{\partial \underline{c}} = \frac{\partial \underline{h}}{\partial \underline{x}} \frac{\partial \underline{x}}{\partial \underline{c}} + \frac{\partial \underline{h}}{\partial \underline{y}} \frac{\partial \underline{y}}{\partial \underline{c}} \quad , \quad \frac{\partial \underline{y}(0)}{\partial \underline{c}} = \underline{\underline{I}}$$

or in a more compact form, using Equations (3.5.4a) and (3.5.4b), we may write

$$\begin{bmatrix} \dot{\underline{\underline{R}}} \\ \dot{\underline{\underline{S}}} \end{bmatrix} = \begin{bmatrix} \frac{\partial \underline{f}}{\partial \underline{x}} & \frac{\partial \underline{f}}{\partial \underline{y}} \\ \frac{\partial \underline{h}}{\partial \underline{x}} & \frac{\partial \underline{h}}{\partial \underline{y}} \end{bmatrix} \begin{bmatrix} \underline{\underline{R}} \\ \underline{\underline{S}} \end{bmatrix}, \quad \begin{matrix} \underline{\underline{R}}(0) = \underline{\underline{0}} \\ \underline{\underline{S}}(0) = \underline{\underline{I}} \end{matrix} \qquad \text{Eq.(3.5.5)}$$

Thus, Equations (3.5.1), (3.5.3) and (3.5.5) constitute two sets of initial-value problems for $\underline{x}$, $\underline{y}$ and for R, S, which can be integrated concurrently. Only $\underline{y}(T)$ and S(T) will be used as a result of this integration. Here, S(T) predicts the change in $\underline{y}(T)$ as a result of changes in $\underline{c}$, by the defining Equation (3.5.4b). Therefore, if $\underline{y}(T)$ is not suffficiently close to the given vector $\underline{b}$, the vector $\underline{c}$ should be changed in a direction dictated by the sensitivity matrix S(T).

Suppose a finite change $\Delta\underline{c}$ on the initial guess $\underline{c}$ is introduced. The resulting change in $\underline{y}(T)$ can be approximated by

$$\Delta \underline{y}(T) = \left(\frac{\partial \underline{y}(T)}{\partial \underline{c}}\right) . \Delta \underline{c} + 0(\Delta \underline{c}) = \underline{\underline{S}}(T)\, \Delta \underline{c} \qquad \text{Eq.(3.5.6)}$$

However, we would like to make $\underline{y}(T)=\underline{b}$. This makes $\Delta\underline{y}(T)=\underline{b}-\underline{y}(T)$, and Equation (3.5.6) can be used to calculate $\Delta\underline{c}$ approximately as

$$\Delta\underline{c} = (\underline{\underline{S}}(T)^{-1}(\underline{b}-\underline{y}(T)) \qquad \ldots\ldots\ldots \text{ Eq.(3.5.7)}$$

provided that the sensitivity matrix is invertible.

Now, the method described above can be summarized as an iterative algorithm :

{ input : vectors $\underline{a}$ and $\underline{b}$ }
choose a norm in E^m and an accuracy δ
make an initial guess for $\underline{c}$
repeat
    solve the initial-value problems
    if $\|\underline{b}-\underline{y}(T)\| < \delta$ then solution found
                else update $\underline{c}$ using Eq.(3.5.7)
until solution found or too many iterations
{output : $\underline{x}(t)$ and $\underline{y}(t)$, for $0 \leq t \leq T$ }

If the original problem has no solution, this algorithm will of course fail. This failure will usually produce a non-invertible S(T). But the converse is not generally true. That is, if the algorithm happens to converge to a non-invertible S(T), it does not mean that the original problem has no solution. In fact, in such a case, the algorithm should be restarted using a different initial guess for $\underline{c}$.

IV

NUMERICAL RESULTS AND SIDE-PRODUCTS OF SENSITIVITY APPROACH TO THE SOLUTION OF TPBVP

4.1. NUMERICAL RESULTS

The method described above has been used in the solution of several linear and nonlinear two-point boundary-value problems. Three of the examples reported below represent linear cases, two with scalar variables, the other with vector variables. The last three examples involve equations with different nonlinearity features and they exhibit the generality of the method.

In all examples, except Example 5, an error tolerance of ε=0.0001 was allowed. The integrations were performed with a step-size of 0.01.

Example 1 :

The linear system

$$\dot{x} = -2\sqrt{2}x - 0.5y \qquad \dot{y} = -2x + 2\sqrt{2}y$$

$$x(0) = 2 \qquad y(1) = 0$$

was solved and $y(0)=0.685$ was found in 5 iterations. In this example, the Runge-Kutta 4 integration routine was employed.

Example 2 :

$$\dot{x} = -x - 0.5y \qquad \dot{y} = -2x + y$$

$$x(0) = 1 \qquad y(1) = 0$$

In this example $y(0)$ was found out to be 0.773 in 4 iterations with the same integration routine.

Example 3 :

Vector variables were considered in this example.

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix} \underline{x} + \begin{bmatrix} 0 & 0 \\ 0 & -0.5 \end{bmatrix} \underline{y}$$

$$\dot{\underline{y}} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \underline{x} + \begin{bmatrix} 0 & -2 \\ 1 & -3 \end{bmatrix} \underline{y}$$

$$\underline{x}(0) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad y(1) = \begin{bmatrix} 1 \\ -1 \end{bmatrix}$$

The $\underline{y}(0)$ vector was calculated to be $[-12.07 \ -12.73]$ in 4 iterations using the same Runge-Kutta 4 integration routine again. It should be noted that new method does not bother with any additional computational complexity and difficulty in extending to the vector variable cases. The number of iterations and computation time for vector variable cases are almost the same for scalar variable cases.

Example 4 :

Here the nonlinearity is in quatient form,

$\dot{x} = y/(1+y)$ $\qquad$ $\dot{y} = x$

$x(0) = 0$ $\qquad$ $y(1) = 1$

Starting with an initial guess of $y(0)=3$, the unspecified initial condition $y(0)$ was found out to be 0.772 in 4 iterations. The Runge-Kutta 4 integration routine was used in solving initial value problems.

Example 5 :

Another type of nonlinearity is considered in the following problem,

$$\dot{x} = y \qquad \dot{y} = -1-x-y^9$$

$$x(0) = 0 \qquad y(1) = -0.5$$

Starting with an initial guess $y(0)=0$, the algorithm produced $y(0)=0.652$ in 6 iterations using an error tolerance of 0.001. The iterated Euler routine was used for integration.

Example 6 :

A stronger nonlinearity is involved in the following problem,

$$\dot{x} = y \qquad y = |x+y|^{\frac{1}{2}}$$

$$x(0) = 0 \qquad y(1) = 1$$

In this problem, starting with a guess of $y(0)=2$, the algorithm found the value $y(0)=0.179$ in 11 iterations. For integration the Runge-Kutta 4 routine was used.

Actually several more problems, both linear and nonlinear, are solved successfully using the new method. However, the reported ones are sufficient to elaborate the significance of the method.

4.2. ONE-STEP CONVERGENCE PROOF FOR LINEAR TPBVP

It can be proven that when the original equations are linear, the algorithm converges to the solution always in one step regardless of the initial guess. The only requirement for this one-step convergence is the exact forward integration of the resulting initial-value problem. This feature was not observed in the linear examples reported, due to the fact that approximate numerical integration routines were employed.

Consider the following general linear two-point boundary-value problem.

$$\dot{\underline{x}} = \underline{\underline{A}}\underline{x} + \underline{\underline{B}}\underline{y} \qquad \dot{\underline{y}} = \underline{\underline{C}}\underline{x} + \underline{\underline{D}}\underline{y}$$

$$\underline{x}(0) = \underline{a} \qquad \underline{y}(T) = \underline{b}$$

or writing in a more compact matrix form, we have

$$\begin{bmatrix} \dot{\underline{x}} \\ \dot{\underline{y}} \end{bmatrix} = \begin{bmatrix} \underline{\underline{A}} & \underline{\underline{B}} \\ \underline{\underline{C}} & \underline{\underline{D}} \end{bmatrix} \begin{bmatrix} \underline{x} \\ \underline{y} \end{bmatrix} \qquad \begin{aligned} \underline{x}(0) &= \underline{a} \\ \underline{y}(T) &= \underline{b} \ . \end{aligned}$$

The sensitivity equations for the above problem are as follows:

$$\begin{bmatrix} \dot{\underline{\underline{R}}} \\ \dot{\underline{\underline{S}}} \end{bmatrix} = \begin{bmatrix} \underline{\underline{A}} & \underline{\underline{B}} \\ \underline{\underline{C}} & \underline{\underline{D}} \end{bmatrix} \begin{bmatrix} \underline{\underline{R}} \\ \underline{\underline{S}} \end{bmatrix} \qquad \begin{aligned} \underline{\underline{R}}(0) &= \underline{\underline{0}} \\ \underline{\underline{S}}(0) &= \underline{\underline{I}} \end{aligned}$$

where $\underline{\underline{I}}$ is the mxm identity matrix. Now, let the initial guess on $\underline{y}(0)$ be $\underline{c}$ vector.

Then we have two sets of linear initial-value problems for $\underline{x},\underline{y}$ and for $\underline{R},\underline{S}$.

The solution of these two sets of linear initial-value problems can be written analytically as follows

$$\begin{bmatrix} \underline{x}(t) \\ \underline{y}(t) \end{bmatrix} = e^{\underline{\underline{\Lambda}}t} \cdot \begin{bmatrix} \underline{a} \\ \underline{c} \end{bmatrix}$$

and

$$\begin{bmatrix} \underline{\underline{R}}(t) \\ \underline{\underline{S}}(t) \end{bmatrix} = e^{\underline{\underline{\Lambda}}t} \begin{bmatrix} \underline{\underline{0}} \\ \underline{\underline{I}} \end{bmatrix}$$

where

$$\underline{\underline{\Lambda}} = \begin{bmatrix} \underline{\underline{A}} & \underline{\underline{B}} \\ \underline{\underline{C}} & \underline{\underline{D}} \end{bmatrix}$$

let $$e^{\underline{\underline{A}}\Delta t} = \begin{bmatrix} \underline{\underline{P}}(t) & \underline{\underline{N}}(t) \\ \underline{\underline{Q}}(t) & \underline{\underline{M}}(t) \end{bmatrix}$$

then $$\begin{bmatrix} \underline{\underline{R}}(t) \\ \underline{\underline{S}}(t) \end{bmatrix} = e^{\underline{\underline{A}}\Delta t} \begin{bmatrix} \underline{\underline{0}} \\ \underline{\underline{I}} \end{bmatrix} = \begin{bmatrix} \underline{\underline{P}}(t) & \underline{\underline{N}}(t) \\ \underline{\underline{Q}}(t) & \underline{\underline{M}}(t) \end{bmatrix} \begin{bmatrix} \underline{\underline{0}} \\ \underline{\underline{I}} \end{bmatrix}$$

$$\begin{bmatrix} \underline{\underline{R}}(t) \\ \underline{\underline{S}}(t) \end{bmatrix} = \begin{bmatrix} N(t) \\ M(t) \end{bmatrix}$$

which implies that the solution for $\underline{x}(t)$ and $\underline{y}(t)$ can be written as follows

$$\underline{x}(t) = \underline{\underline{P}}(t)\ \underline{a} + \underline{\underline{R}}(t)\ \underline{c}$$

$$\underline{y}(t) = \underline{\underline{Q}}(t)\ \underline{a} + \underline{\underline{S}}(t)\ \underline{c}$$

and at the final time T we obtain

$$\underline{y}(T) = \underline{\underline{Q}}(T)\ \underline{a} + \underline{\underline{S}}(T)\ \underline{c}$$

The problem requires that $\underline{y}(T)=\underline{b}$. Therefore, the unspecified boundary value (which is guessed to be c vector) must satisfy the following condition at the final end.

$$\underline{\underline{Q}}(T)\ \underline{a} + \underline{\underline{S}}(T)\ \underline{c}_{exact} = \underline{b}$$

Solving for $\underline{c}$ vector, we get

$$\underline{c}_{exact} = (\underline{\underline{S}}(T))^{-1} \cdot (\underline{b} - \underline{\underline{Q}}(T) \cdot \underline{a})$$

However, we use

$$\Delta \underline{c} = (\underline{\underline{S}}(T))^{-1} (\underline{b}-\underline{y}(T))$$

as the iteration equation in the algorithm. Thus,

$$\underline{c}_{new} = \underline{c} + (\underline{\underline{S}}(T))^{-1} (\underline{b}-\underline{y}(T))$$

In the first iteration step $\underline{y}(T)$ has been calculated to be

$$\underline{y}(T) = \underline{\underline{Q}}(T)\ \underline{a} + \underline{\underline{S}}(T)\ \underline{c}$$

Substituting this value of $\underline{y}(T)$ into the iteration equation we get

$$\underline{c}_{new} = \underline{c} + (\underline{\underline{S}}(T))^{-1} (\underline{b}-\underline{\underline{Q}}(T)\underline{a}-\underline{\underline{S}}(T)\underline{c})$$

$$= \underline{c} + (\underline{\underline{S}}(T))^{-1}\ \underline{b} - (\underline{\underline{S}}(T))^{-1}\underline{\underline{Q}}(T)\underline{a} - (\underline{\underline{S}}(T))^{-1}\underline{\underline{S}}(T)\underline{c}$$

$$= \underline{c} + (\underline{\underline{S}}(T))^{-1} (\underline{b}-\underline{\underline{Q}}(T)\underline{a}) - \underline{c}$$

$$= (\underline{\underline{S}}(T))^{-1} (\underline{b}-\underline{\underline{Q}}(T)\underline{a})$$

$\therefore$ $\underline{c}_{exact} = \underline{c}_{new}$ after the first iteration.

4.3. AN EFFICIENT ALGORITHM FOR FINDING THE SUFFICIENT NUMBER OF TERMS IN NUMERICAL EVALUATION OF FUNCTIONS BY POWER SERIES

Suppose that we have a function f having derivatives of every order in an open interval about a point a. We call such functions infinitely differentiable in this interval. Then we can certainly form the following power series which is known as Taylor's series generated by f at a.

$$f(x) = \sum_{n=0}^{\infty} \frac{f^n(a)}{n!} (x-a)^n$$

We also know that Taylor's formula with remainder provides a finite expansion of the form

$$f_n(x) = \sum_{k=0}^{n} \frac{f^{(k)}(a)}{k!} (x-a)^k + E_n(x)$$

The finite sum is the Taylor polynomial of degree n generated by f at a, and $E_n(x)$ is the error made in approximating f by its Taylor polynomial. If we let $n \to \infty$ we see that the power series will converge to f(x) if and only if the error term tends to zero. A useful sufficient condition for the error term to tend to zero is stated in the following theorem.

THEOREM 4.3.1 : Assume f is infinitely differentiable in an open interval I and assume that there is a positive constant A such that

$$\left| f^{(n)}(x) \right| \leq A^n \quad \text{for } n=1,2,3,\ldots$$

and every x in I. Then the Taylor's series generated by f at a converges to f(x) for each x in I.

However, this theorem does not provide any means for finding the sufficient number of terms in the numerical evaluation of a function value at a given point by power series. Since it is not possible to let n go to infinity for convergence to the exact value in numerical calculation it is obligatory to devise an algorithm for finding this sufficient number n.

Here, at this point we have again tried to exploit sensitivity idea and Newton's method.

First assume that sufficient condition for convergence holds for the considered function f in the given interval; therefore, the error term is a smoothly decreasing function of iteration terms n after some fixed value of n.

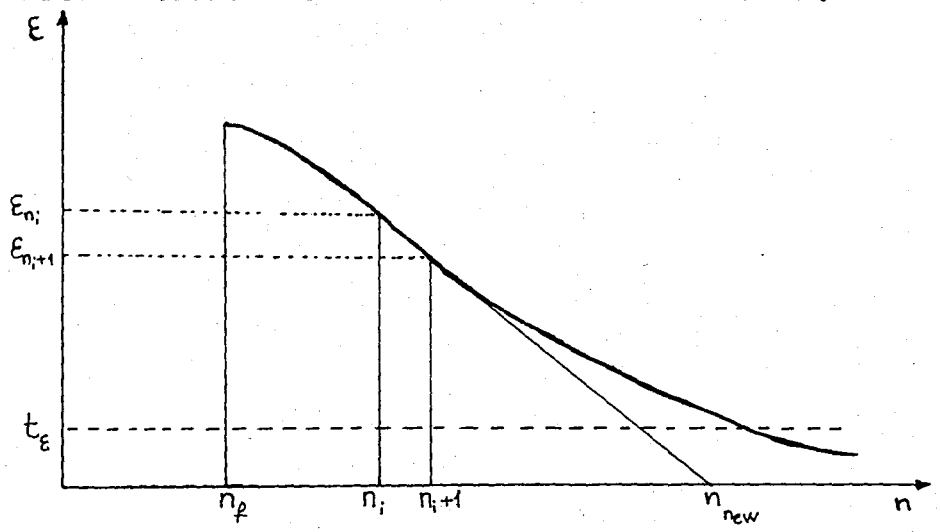

FIGURE 4.3.1. Graph of error term function

By definition error term at $n=n_i$ is given by

$$\varepsilon_{n_i} = f_{n_i+1} - f_{n_i}$$

Similarly,

$$\varepsilon_{n_i+1} = f_{n_i+2} - f_{n_i+1}$$

Then the sensitivity (or the gradient) function may be defined as

$$S_{n_i} = \frac{\varepsilon_{n_i+1} - \varepsilon_{n_i}}{n_i+1 - n_i} = \varepsilon_{n_i+1} - \varepsilon_{n_i}$$

Since our aim is to make the error term zero or below certain prescribed tolerance value t_ε, we may write the following recursion formula using the Newton's idea.

$$n_{new} = n_i + S^{-1}_{n_i}(0 - \varepsilon_{n_i})$$

$$n_{new} = n_i - S^{-1}_{n_i} \cdot \varepsilon_{n_i}$$

Eventhough, it is true that the error term will tend to decrease very smoothly (under sufficiency condition for convergence) after some fixed value of n, call it n_f, we do not know this "fixed" value n_f beforehand to set it as initial guess n_i in order to start the iteration. Here, we have two choices. First we may obtain n_f analytically. Consider the sensitivity (or gradient) function S_n

$$S_n = \varepsilon_{n+1} - \varepsilon_n = (f_{n+2} - f_{n+1}) - (f_{n+1} - f_n)$$

Set this function equal to zero and solve for n value in terms of x, $f^{(n+2)}(a)$, $f^{(n+1)}(a)$, $f^{(n)}(a)$. The "fixed" value n_f may

be set equal to one more of the value n obtained above. Second choice could be an algorithmic approach. We may not bother with the calculation of n_f; but instead we may increase our initial guess n_i by fixed amounts until we arrive to the smoothly decreasing portion of the error term function. This fixed amount may be increased geometrically or exponentially as the failure occurs in reaching the decreasing portion of error term function in order to speed up the algorithm.

Now, let us summarize the first choice as an iterative algorithm.

choose an accuracy t_ε

make an initial guess $n > n_f$

repeat $n=n_{new}$

calculate ε_n , ε_{n+1} , S_n

if $\varepsilon_n < t_\varepsilon$ then solution found

else update n using $n_{new}=n-S_n^{-1} \cdot \varepsilon_n$

until solution is found

{output : n}

The ideas and algorithms developed in this section can easily be extended to the numerical evaluation of function of matrices by power series. However, it is necessary to make a few slight modifications. We know that a function of square matrix can, in general, be expressed by a power series as follo

$$f(\underline{\underline{A}}) = \sum_{k=0}^{\infty} c_k \underline{\underline{A}}^k$$

For instance, an important function in system and control theory is

$$e^{\underline{\underline{A}}T} = \sum_{k=0}^{\infty} \frac{(\underline{\underline{A}}T)^k}{k!}$$

Here, we are again faced with the question of "sufficient number of terms" to terminate the infinite power series in the numerical evaluation of a function of a matrix. Eventhough, there may exist some empirical relations such as

$$N = \min \left\{ 3 \left\| \underline{\underline{A}}T \right\| + 6 \ , \ 100 \right\}$$

where N is the sufficient number of terms, it is easy to show that this kind of empirical relations is, in general, not sufficient to cover all degenerate cases.

Now, the argument of the funciton, the error term and the sensitivity functions are all matrices; whereas the number of terms to be chosen or found out is still a scalar value. Therefore, a scalar value must be determined out of these matrix funcitons by employing some relevant and logical criteria in order to be able to draw a relation with the scalar value of the number of terms. This scalar value can be chosen to be

(a) the trace of the matrix

(b) the largest of the absolute eigenvalues of the matrix.

Then, an algorithm similar to the ones developed for scalar functions might be devised.

First let us consider a scalar example.

Example 1 : Let $f(x)=e^x$, and develop the recursion formula for this function

$$e^x = 1 + x + \frac{x^2}{2!} + \ldots + \frac{x^n}{n!} + \ldots$$

$$\varepsilon_n = \frac{x^{n+1}}{(n+1)!} \quad \text{and} \quad \varepsilon_{n+1} = \frac{x^{n+2}}{(n+2)!}$$

Therefore,

$$S_n = \varepsilon_{n+1} - \varepsilon_n = \frac{x^{n+2}}{(n+2)!} - \frac{x^{n+1}}{(n+1)!}$$

$$= \frac{x^{n+1}}{(n+1)!} \left(\frac{x}{n+2} - 1\right)$$

and

$$S_n^{-1} = \frac{(n+1)!}{x^{n+1}} \quad \frac{n+2}{x-n-2}$$

Then, the recursion formula

$$n_{new} = n - \frac{(n+1)!}{x^{n+1}} \quad \frac{n+2}{x-n-2} \quad \frac{x^{n+1}}{(n+1)!}$$

$$n_{new} = n - \frac{n+2}{x-n-2}$$

The most important quantity in the above recursion formula is the denominator (x-n-2). If the initial guess on n is chosen arbitrarily as it is done in the second algorithm developed in this section, then a special attention must be paid to the denominator (x-n-2). If this quantity is positive then the guess on n must be increased by certain amounts until the denominator becomes negative.

Remark 1 In case of calculating $e^{\underline{\underline{A}}}$, x might be the largest of the absolute eigenvalue of the matrix $\underline{\underline{A}}$.

Remark 2 If $x > M$ then overflow will occur due to the limits of the computing element. For instance, M is equal to 230 for Univac 1106 Computer used in this study.

If $x < 0$ then there is no risk of being out of range.

Remark 3 Similar recursion formulas may be derived for other known functions

Remark 4 One may stop the calculation when the incremental change in n value which is given by $-(n+2)/(x-n-2)$ is less than 1.5. This value of 1.5 is determined experimentally.

4.4. A NEW METHOD FOR NUMERICAL SOLUTION OF LINEAR STIFF SYSTEMS

A stiff ordinary differential equation (o.d.e) is one in which one component of the solution decays much faster than others. Many chemical engineering systems give rise to systems of stiff o.d.es.

Most realistic stiff systems do not have analytical solutions so that a numerical procedure must be used. Conventional methods such as Euler, explicit Runge-Kutta and Adams-Moulton are restricted to a very small step size in order that the solution be stable. This means that a great deal of computer time could be required.

Some of the more readily available methods for stiff equations include :

(a) Variable-order methods based on backward differentiation multistep formulas
(b) Methods based on the trapezoidal rule
(c) Implicit Runge-Kutta methods
(d) Methods based on the use of preliminary mathematical transformations to remove stiffness and the solution of the transformed problem by traditional techniques
(e) Methods eliminating those differential equation having

small time constants and solving them as algebraic equations instead.

Consider the following linear system

$$\dot{\underline{x}} = \underline{\underline{A}}\,\underline{x} \quad , \quad \underline{x}(0) = \underline{x}_0$$

where the eigenvalues of the matrix A are widely separated, i.e., a stiff system. We know that the solution of this system is given by

$$\underline{x}(t) = e^{\underline{\underline{A}}t} \cdot \underline{x}_0$$

Thus, the problem has now been transformed to the exact numerical calculation of the exponential function of a matrix. If the eigenvalues of the matrix A had not been widely separated any marching method would usually have been able to solve the above linear system.

In the previous section we have developed an efficient method for the numerical calculation of a funciton of a matrix by power series. Therefore, it would be very easy for us to calculate this special case of a function of a matrix using those results.

Example 1 :

A linear two-point boundary-value problem which gives rise to a stiff initial-value problem for any initial guess of the unspecified boundary-value is considered and the method mentioned above is used as the integration routine for solving those stiff initial-value problems.

$$\dot{x} = 10y \qquad \dot{y} = 10x$$

$$x(0) = 0 \qquad y(2) = 1$$

Starting with an initial guess of $y(0)=1$, the unspecified initial condition $y(0)$ was found out to be 0.412×10^{-8} in three iterations. An error tolerance of $\delta=0.0001$ was allowed.

4.5. SOME EQUIVALENCE PROPERTIES AND MODIFICATIONS IN SOLVING LINEAR AND NONLINEAR TPBVP USING SENSITIVITY APPROACH

There is an important property related to the case of linear TPBVP which one can exploit quite easily. This is the equivalence of the sensitivity matrix and the so-called transition matrix of the linear system of differential equations. This equivalence property removes the necessity of solving two-sets of initial-value problems which was necessary in the algorithm developed for solving the two-point boundary-value problem; because when one set of initial-value problem is solved, its solution will represent both the sensitivity matrix and the transition matrix. Thus, when the sensitivity matrix is then multiplied by the initial value vector of that iteration the vector values $\underline{x}(T)$ and $\underline{y}(T)$ would have been obtained. This is quite a saving from computational point of view.

There exist one more important property of sensitivity matrix which may be valuable in the treatment of nonlinear two-point boundary-value problems. Actually, the sensitivity matrix is a solution of the linearized version of the nonlinear system of equations.

Now it is possible to use both of the equivalence properties of the sensitivity function in the treatment of nonlinear two-point boundary-value problems. That is, first the nonlinear two-point boundary-value problem is assumed to be linearized by considering only the initial-value problem for R and the sensitivity matrix S. Next, this sensitivity matrix is again multiplied with the augmented initial condition vector of that iteration to obtain $\underline{x}(T)$ and $\underline{y}(T)$ values for the linearized version of the nonlinear problem. The iteration may continue until the solution is within some prescribed limits. Then one may return to the normal procedure of the algorithm.

4.6. CONCLUSIONS

In view of the variety of boundary-value problems that can be encountered, it is not possible to imagine one algorithm ever being able to solve all problems both accurately and efficiently. Even in the case of a nonlinear two-point boundary-value problem, it can be seen that some of the existing methods have "initial guess" difficulty, and some have "poor convergence" near the solution.

In this study, a new method is proposed for the numerical solution of the two-point boundary-value problem which uses trajectory sensitivities with respect to the initial conditions. Motivation for using sensitivities originated from the fact that the Euler-Lagrange equations which give rise to the two-point boundary-value problem in optimal control were often highly sensitive to changes in the unspecified boundary condi-

tions, and therefore, they were causing problems for the methods which use initial guess.

The generality and the power of the new method lie in the fact that the transition from the unspecified boundary conditions to the specified end is calculated quite efficiently and accurately for nonlinear and large systems.

The sensitivity matrix which is instrumental in the implementation of the method may in some problems become either zero or infinity. In such situations a remedy may be the re-assignment of the initial guess. However, it must be noted that we have never met with such critical situations eventhough many linear and nonlinear examples are solved using the method.

There is one more difficulty associated with the numerica solution of the differential equations forward in time after the initial guess is made. This difficulty is known as the "stiffness' in the differential equations and can be overcome by selecting appropriate integration routines. In fact, as a byproduct of this study, a method is developed which resolves this "stiffness" problem in case of linear differential equations.

Moreover, it has been proven that when the original equations are linear, the algorithm converges to the solution always in one step regardless of the initial guess. The only requirement for this one-step convergence is the exact forward integration of the resulting initial-value problem. This feature was not observed in the linear examples reported due to the fact that approximate integration routines were employed.

V

DIRECT SENSITIVITY APPROACH TO THE GENERAL OPTIMAL CONTROL PROBLEM

5.1. PROBLEM FORMULATION AND HAMILTON-JACOBI-BELLMAN EQUATION

It often happens in automatic control problem that one would like to know the optimal control function u(t) from a great many different initial points to a given terminal hypersurface, since we may not know where the system will start from or when it will start. To cover this situation we must calculate a "family" of optimal paths so that all of the possible initial points are on, or at least very close to, one of the calculated optimal paths. In the literature on the calculus of variations, such a family is called a "field of extremals".

In general, only one optimal path to the terminal hypersurface will pass through a given point $(\underline{x}(t),t)$, and a unique optimal control vector $\underline{u}^0(t)$ is then associated with each point. Hence, we may write

$$\underline{u}^0 = \underline{u}^0 (\underline{x},t) \qquad \text{..... Eq.(5.1.1)}$$

This is the optimal feedback control law; i.e., the control vector is given as a function of the present state $\underline{x}(t)$ and the present time t.

Associated with starting from a point $(\underline{x},t)$ and proceeding optimally to the terminal hypersurface, there is a

unique optimal value of the performance index, J^0. We may therefore regard J^0 as a function of the initial point, that is,

$$J^0 = J^0(x,t)$$

This is the "optimal return function".

One aspect of the classical Hamilton-Jacobi theory is concerned with finding the partial differential equation satisfied by the optimal return function $J^0(x,t)$. There is also a (vector) partial differential equation satisfied by the optimal control functions $u^0(x,t)$. Bellman has generalized the Hamilton-Jacobi theory to include multistage systems and combinatorial problems and he called this overall theory dynamic programming.

Now consider the general optimal control problem for an arbitrary initial point $(\underline{x},t)$. The performance index is

$$J = \varphi\left[\underline{x}(t_f),t_f\right] + \int_t^{t_f} l(\underline{x}(\tau),\underline{u}(\tau),\tau)\,d\tau \qquad \text{Eq.(5.1.3)}$$

and system equations are

$$\dot{\underline{x}} = \underline{f}(\underline{x},\underline{u},t) \qquad \text{..... Eq.(5.1.4)}$$

with the terminal boundary conditions

$$\psi\left[\underline{x}(t_f),t_f\right] = 0 \qquad \text{..... Eq.(5.1.5)}$$

The optimal return function, defined in Equation (5.1.2), is

given symbolically by

$$J^0(x,t) = \min_{\underline{u}(t)} \left\{ \varphi[\underline{x}(t_f), t_f] + \int_t^{t_f} l(\underline{x},\underline{u},\tau)\,d\tau \right\} \quad \text{Eq.(5.1.6)}$$

with the boundary condition

$$J^0(\underline{x},t) = \varphi(\underline{x},t) \text{ on the hypersurface } \psi(x,t)=0 \quad \text{Eq.(5.1.7)}$$

Assuming that $J^0(\underline{x},t)$ exists, is continuous, and possesses continuous first and second partial derivatives at all points of interest in the $\underline{x},t$ space, and applying the dynamic programming approach one finally obtains the following equation

$$-\frac{\partial J^0}{\partial t} = H^0\left(\underline{x}, \frac{\partial J^0}{\partial x}, t\right) , \quad \text{Eq.(5.1.8)}$$

where

$$H^0\left(\underline{x}, \frac{\partial J^0}{\partial \underline{x}}, t\right) = \min_{u(t)} H\left(\underline{x}, \frac{\partial J^0}{\partial \underline{x}}, \underline{u}, t\right) \quad \text{Eq.(5.1.9)}$$

and

$$H\left(\underline{x}, \frac{\partial J^0}{\partial \underline{x}}, \underline{u}, t\right) = l(\underline{x},\underline{u},t) + \left(\frac{\partial J^0}{\partial \underline{x}}\right)^T . \underline{f}(\underline{x},\underline{u},t) \quad \text{Eq.(5.1.10)}$$

Equation (5.1.8) is called the Hamilton-Jacobi-Bellman Equation; it is a first-order nonlinear partial differential equation and it must be solved with the boundary condition (5.1.5).

Numerical solution for the Hamilton-Jacobi-Bellman Equation is very difficult. In fact, it is only rarely feasible to solve this partial differential equation for a nonlinear

system of any practical significance; and hence, the development of "exact" explicit feedback control schemes for nonlinear systems is usually out of reach.

The Hamilton-Jacobi-Bellman Equation gives rise to some valuable outcomes only in the case of linear plant dynamics and quadratic performance criteria which are referred to as linear regulator problems in the optimal control theory. If the final time is specified and fixed in the linear regulator problem then the Hamilton-Jacobi-Bellman equation reduces to the well known "Matrix Riccati Differential Equation", and if the final time is infinite then one ends up with the "Algebraic Matrix Riccati Equation", both for which there exist various efficient numerical solution schemes in the literature.

5.2. PERFORMANCE INDEX SENSITIVITY

A variety of computational methods have been developed for solving general nonlinear optimal control problems. All of these methods are iterative techniques which somehow use successive approximations; and certain of them involve linearization of the nonlinear differential equations that are generated by first-order variational analysis.

Here a new approach will be presented which hopefully will give rise to various iterative methods and algorithms for solving the general optimal control problem stated in section 5.1. This approach is essentially based on the "performance index sensitivity". Therefore, before devoting ouselves to the different methods and the algorithms within this approach, let us briefly outline the basic properties of the performance-

index sensitivity.

Changes of the performance-index of an optimal control system can be caused either by parameter changes of the mathematical model of the given process (the plant) or by changes in the control law. For a feedback control system, changes in the control law are equivalent changes of the controller parameters. Since the optimization is achieved by minimizing the performance index J with respect to the control variable, it is evident that the performance index sensitivity J_{α_R} with respect to changes in the controller parameters α_R vanish as long as teh minimum is a relative one. On the other hand, the performance-index sensitivity J_{α_S} with respect to changes in the plant parameters α_S is not necessarily equal to zero. It can take on any real value. The same is true for simultaneous changes of the plant and controller parameters as may be encountered in ideal optimal controllers that provide optimal control for any set of actual parameters.

Let us now derive the necessary equation for the performance index sensitivity $J_{\alpha} = \left(\partial J / \partial \underline{\alpha} \right) \Big|_{\underline{\alpha}_0}$ of a continuous optimal system with respect to controller parameter variations.

Let the control system, for which an open-loop input or an optimal closed-loop control is to be designed, be given by the following general vector differential equation

$$\dot{\underline{x}}(t) = \underline{f}(\underline{x}(t), u(t), t) \quad , \quad \underline{x}(t_0) = \underline{x}_0 \qquad \text{Eq.(5.2.1)}$$

where $\underline{x}$ is a n-dimensional state vector, u is a one-dimensional

control vector.

Now the optimization problem consists in finding a control u(t) which minimizes a performance index of the form

$$J = \varphi(x(t_f), t_f) + \int_{t_0}^{t_f} l(\underline{x}, u, t)\,dt \qquad \text{Eq.(5.2.2)}$$

We have assumed that u(t) is, in particular, given by

$$u(t) = \underline{p}^T(t)\ \underline{x}(t) \qquad \text{.....} \qquad \text{Eq.(5.2.3)}$$

where $\underline{p}(t)$ is n-dimensional controller parameter vector. We might then obtain the performance index sensitivity equations by taking the partial derivative of the Equation (5.2.2) with respect to the parameter vector $\underline{p}$ as follows

$$J_{\underline{p}} = \frac{\partial J}{\partial \underline{p}} = \frac{\partial \varphi}{\partial \underline{x}}\bigg|_{t_f} \cdot \frac{\partial \underline{x}}{\partial \underline{p}}\bigg|_{t_f} + \int_{t_0}^{t_f} \left[\left(\frac{\partial l}{\partial \underline{x}}\right)\left(\frac{\partial \underline{x}}{\partial \underline{p}}\right) + \frac{\partial l}{\partial \underline{p}} \right] dt \qquad \text{Eq.(5.2.4)}$$

Here, $\partial\varphi/\partial\underline{x}$, $\partial l/\partial\underline{x}$, and $\partial l/\partial\underline{p}$ are defined as row vectors; whereas the term $(\partial\underline{x}/\partial\underline{p})$ is the trajectory sensitivity matrix S which is defined in section 3.4. Thus, Equation (5.2.4) can be written as

$$J_{\underline{p}} = \frac{\partial \varphi}{\partial \underline{x}}\bigg|_{t_f} \underline{\underline{S}}(t_f) + \int_{t_0}^{t_f} \left[\left(\frac{\partial l}{\partial \underline{x}}\right) \cdot \underline{\underline{S}} + \frac{\partial l}{\partial \underline{p}} \right] dt \qquad \text{Eq.(5.2.5)}$$

S can be determined from the trajectory sensitivity equation

$$\dot{\underline{\underline{S}}} = \left(\frac{\partial \underline{f}}{\partial \underline{x}}\right) \cdot \underline{\underline{S}} + \left(\frac{\partial \underline{f}}{\partial \underline{p}}\right) \quad , \quad \underline{\underline{S}}(0) = \underline{\underline{0}}$$

The solution for the trajectory sensitivity equation is given by

$$\underline{\underline{S}} = \int_{t_o}^{t} \underline{\underline{\phi}}(t,\tau)\ \left(\frac{\partial \underline{f}}{\partial \underline{p}}\right)\ d\tau \qquad \text{..... Eq.(5.2.7)}$$

$\underline{\underline{\phi}}(t,\tau)$ is the transition matrix obeying $\underline{\underline{\phi}}(\tau,\tau)=\underline{\underline{I}}$ with

$$\frac{\partial \underline{\underline{\phi}}}{\partial t} = \frac{\partial \underline{f}}{\partial \underline{x}}\ \underline{\underline{\phi}} \qquad \text{..... Eq.(5.2.8)}$$

Actually, we might determine the performance index sensitivity vector still in another way by simply applying the definitions. Let us suppose that the performance index is given by $J^{\underline{p}}$ for the parameter vector $\underline{p}$. Then, let a small change in $\underline{p}$ vector, say $\Delta\underline{p}$ amount, occurred. This change first induces a direct change in the state trajectory of the system and next induces directly and indirectly a change in the performance index. Supposing that the new performance index is given by $J^{\underline{p}+\Delta\underline{p}}$, the deviation in the performance index is given by

$$\Delta J = J^{\underline{p}+\Delta\underline{p}} - J^{\underline{p}} \qquad \text{..... Eq.(5.2.9)}$$

For infinitesimal parameter changes $\Delta\underline{p}=d\underline{p}$, the performance index deviation ΔJ can be written as

$$\Delta J = \left.\frac{\partial J}{\partial \underline{p}}\right|_{\underline{p}} \cdot \Delta\underline{p} \qquad \text{..... Eq.(5.2.10)}$$

Therefore, the performance index sensitivity can be determined directly by applying a perturbation in the parameter vector and noting the deviations. One crucial drawback of this method,

however, is that one would have to deal with nonlinear differential equations whereas the derived sensitivity equations are, in general, linear differential equations.

5.3. THE DIRECT SENSITIVITY APPROACH

The general approach which will be presented below could be named as "direct sensitivity approach", since all of the methods which will be developed using this approach are based upon the performance index sensitivities with respect to controller parameters. That is, one is dealing with the design of optimal control function u(t) "directly" by employing the performance index "sensitivities".

Bearing in mind the fact that the performance index sensitivities are identically equal to zero at the optimal value of the controller parameters, the following general iterative procedure might be considered (or designed) for solving the optimal control problems.

(a) Choose sufficient number of controller parameter vectors, say $\underline{p}_1, \underline{p}_2, \cdots \underline{p}_s$.

(b) Calculate the performance index sensitivities for each of the controller parameter vectors $\{\underline{p}_i\}_{i=1}^{s}$ using any one of the two methods discussed in the previous section. One may also calculate and store the value of the performance index for each controller parameter vectors.

(c) Form a quadratic functional, $\tilde{J}$, in the parameter space using the calculated values of step (b).

(d) Find a new controller parameter vector, say $\underline{p}^*$, which

minimizes the quadratic functional $\tilde{J}$ formed in step (c).

(e) Check the stopping criteria; and stop or continue accordingly.

(f) Put the new minimum controller parameter vector $\underline{p}^*$ in place of the controller parameter vector which either gives rise to the greatest performance index value or produces normwise highest performance index sensitivities, and go to step (b).

The sufficiency of the number of controller parameter vectors basically depends upon the dimension of the system and minimization routine chosen (or more precisely, the formation of the quadratic functional). For instance, 2 controller parameter vectors are sufficient for a second order system if the performance index values are used in the formation of the quadratic functional, whereas at least 3 controller parameter vectors are needed if only the performance index sensitivities are employed in the quadratic functional formation.

In step (c) the method basically tries to approximate the real or exact performance index functional by a quadratic one, call it $\tilde{J}$, using the values calculated in step (b). $\tilde{J}$ can be written as a function of the controller parameter vector $\underline{p}$ as follows

$$\tilde{J}(\underline{p}) = \tfrac{1}{2}\, \underline{p}^T \underline{Q}\, \underline{p} - \underline{p}^T b + c \qquad \ldots\ldots\ \text{Eq.(5.3.1)}$$

where Q is a nxn symmetric matrix to be determined. The minimum of this approximate performance index functional $\tilde{J}$ is given by

$$\underline{p}^* = \underline{\underline{Q}}^{-1}\,\underline{b} \qquad \text{..... Eq.(5.3.2)}$$

Using the performance index sensitivity vectors determined for each controller vector one may obtain the following relation for $\underline{b}$ vector

$$\underline{b} = \underline{\underline{Q}}\,\underline{p}_i - J_{\underline{p}_i} \qquad \forall i \in \{1,2,\ldots s\} \qquad \text{Eq.(5.3.3)}$$

Therefore, $\underline{p}^*$ which is the new minimum vector of the iteration, can be determined as follows

$$\underline{p}^* = \underline{\underline{Q}}^{-1}\underline{b} = Q^{-1}(Q\underline{p}_i - J_{\underline{p}_i})$$

$$= \underline{p}_i - \underline{\underline{Q}}^{-1}\,J_{\underline{p}_i} \quad \text{for any } i \in \{1,2,\ldots s\}$$

Eq.(5.3.4)

where $J_{\underline{p}_i}$ is the performance index sensitivity vector for the ith controller parameter vector.

As it is pointed out previously one may either use both performance index values and performance index sensitivity vectors together or employ only performance index sensitivity vectors in the formation of the quadratic functional (or more specifically, in determining the symmetric Q matrix). However, the sensitivity vectors are more significant and valuable compared to the performance index values; since the absolute performance index values do not mean much whereas the sensitivity vectors give a lot of information about the shape of the functional to be considered.

A single one or a combination of the following conditions might be considered as the stopping criteria of step (e).

(i) the difference of two successive performance index values is less than some given tolerance

(ii) the distance between any two recorded minimum controller parameter vectors is less than some given tolerance

(iii) a certain norm of the sensitivity vectors for each of the controller parameter vectors is less than some given tolerance.

The iterative procedure presented above is actually quite a general approach to the optimal control problems. Depending upon the choice of the functions for $\underline{p}(t)$ and the minimization routine, various effective algorithms may be developed out of this approach. In the following chapters a few of these alternatives will be considered.

VI

A PRIORI POLYNOMIAL APPROXIMATION METHODS VIA DIRECT SENSITIVITY APPROACH FOR THE GENERAL OPTIMAL CONTROL PROBLEMS

6.1. BASIC IDEA OF THE METHOD

As it has been pointed out in the previous chapter, different choices of functions for $\underline{p}(t)$ would give rise to different algorithms; eventhough the approach is still the same direct sensitivity approach. In the algorithms developed in this chapter, the functions which are to be chosen beforehand (apriori) for $\underline{p}(t)$ are polynomials. That is, symbolically, it is assumed that $\underline{p}(t)$ is given by

$$\underline{p}(t) = \begin{bmatrix} p_{01}+p_{11}t+ \cdots + p_{m1}t^m \\ p_{02}+p_{12}t+ \cdots + p_{m2}t^m \\ \vdots \quad \vdots \qquad\qquad \vdots \\ p_{0n}+p_{1n}t+ \cdots + p_{mn}t^m \end{bmatrix} \qquad \text{Eq.(6.1.1)}$$

In this manner, the infinite dimensional optimal control problem has been mapped to a finite dimensional mathematical programming problem. Then, this finite dimensional mathematical programming problem is solved "sequentially" using the general iterative optimization procedure presented in section 5.3. That is, it is first assumed that the controlle parameter vector function is given by the arbitrary vector

$$\underline{p}_{f_0}(t) = \begin{bmatrix} p_{01} \\ p_{02} \\ \vdots \\ p_{0n} \end{bmatrix} \qquad \text{..... Eq.(6.1.2)}$$

and the minimizing $\underline{p}_{f_0}(t)$ is found, say $\underline{p}^*_{f_0}$, using the general iterative optimization procedure. Then, the new controller vector function is formed as

$$\underline{p}_{f_1}(t) = \underline{p}^*_{f_0} + \begin{bmatrix} p_{11} \\ p_{12} \\ \vdots \\ p_{1n} \end{bmatrix} t \qquad \text{..... Eq.(6.1.3)}$$

Now, the general iterative optimization procedure is applied to the arbitrary controller parameter vector $[p_{11}\ p_{12} \cdots p_{1n}]^T$, and the optimal parameter vector, say $\underline{p}^*_{f_1}$, is found. Then, the optimal controller parameter vector function becomes

$$\underline{p}^*_{f_1}(t) = \underline{p}^*_{f_0} + \underline{p}^*_{f_1} t$$

This "sequential" procedure continues until one finally arrives at the minimizing controller parameter vector for $\underline{p}_{f_m}$, say $\underline{p}^*_{f_m}$. The optimal controller parameter vector function is then formed as

$$\underline{p}^*_f(t) = \underline{p}^*_{f_0} + \underline{p}^*_{f_1} t + \cdots + \underline{p}^*_{f_m} t^m \qquad \text{Eq.(6.1.4)}$$

This is a "forward" sequential optimization procedure. The reverse of the above sequential optimization procedure might

be designed. That is, the procedure might begin from the parameter vector $\underline{p}_{b_m}$ and terminate at the parameter vector $\underline{p}_{b_0}$. The optimal controller parameter vector function, in this case, is given by

$$\underline{p}_b^*(t) = \underline{p}_{b_m}^* t^m + \underline{p}_{b_{m-1}}^* t^{m-1} + ... + p_{b_0}^* \qquad \text{Eq.(6.1.5)}$$

This, then, might be called a "backward" sequential optimization procedure for the apriori polynomial approximation methods. Depending on which procedure is used, the optimal control-law is determined by either

$$u(t) = \underline{p}_f^*(t)^T \underline{x}(t) \qquad ..... \qquad \text{Eq.(6.1.6)}$$

or

$$u(t) = \underline{p}_b^*(t)^T \underline{x}(t) \qquad ..... \qquad \text{Eq.(6.1.7)}$$

6.2. NUMERICAL RESULTS

Eventhough the direct sensitivity approach is especially developed for generating the closed-loop control function of the general optimal control problems, it has been observed that the algorithms developed based on this approach has given quite satisfactory results also in the generation of the open-loop optimal control functions. Therefore, a few examples related to this class of problems is also reported below.

First let us consider a simple example whose analytic solution for the closed-loop control function is known.

Example 1 : A first-order linear system with quadratic performance index

Given

$$\dot{x} = u \quad ; \quad x, u \text{ scalar variables}$$

$$J = x^2(t_f) + \tfrac{1}{2} \int_0^1 u^2(t)\, dt$$

Find $u(x,t)$ to minimize J

The analytic solution is given by

$$u(x,t) = \frac{1}{t-3/2}\, x(t)$$

Therefore, the optimal controller parameter function which is, in this case, a scalar one, is

$$p^0(t) = \frac{1}{t-3/2}$$

which can be approximated by a polynomial as follows

$$p^0(t) \cong -\frac{2}{3} - \frac{4}{9}\, t - \frac{8}{27}\, t^2 - \ldots - \left(\frac{2}{3}\right)^n t^n$$

As n approaches to infinity the exact value of the function $p^0(t)$ will be obtained.

The following polynomial function is obtained for the controller parameter function when forward sequential optimization procedure is used.

$$p_f(t) = -0.976 - 0.279\, t - 0.204\, t^2$$

The value of the performance index functional for this function

is equal to 0.312.

When backward sequential optimization procedure is employed the following polynomial function is obtained for the controller parameter function.

$$p_b(t) = - 0.442 - 0.509\ t - 2.707\ t^2$$

The value of the performance index functional is equal to 0.342 for this function.

The figure (6.2.1) shows the graphs of the four functions, $p^0(t)$, $p_f(t)$, $p_b(t)$, $p_{av}(t)$. It can be seen from the graph that the function $p_f(t)$ first overestimates the optimal function $p^0(t)$ and then underestimates in the second half of the time interval. The same is true for the function $p_b(t)$ in a reverse order.

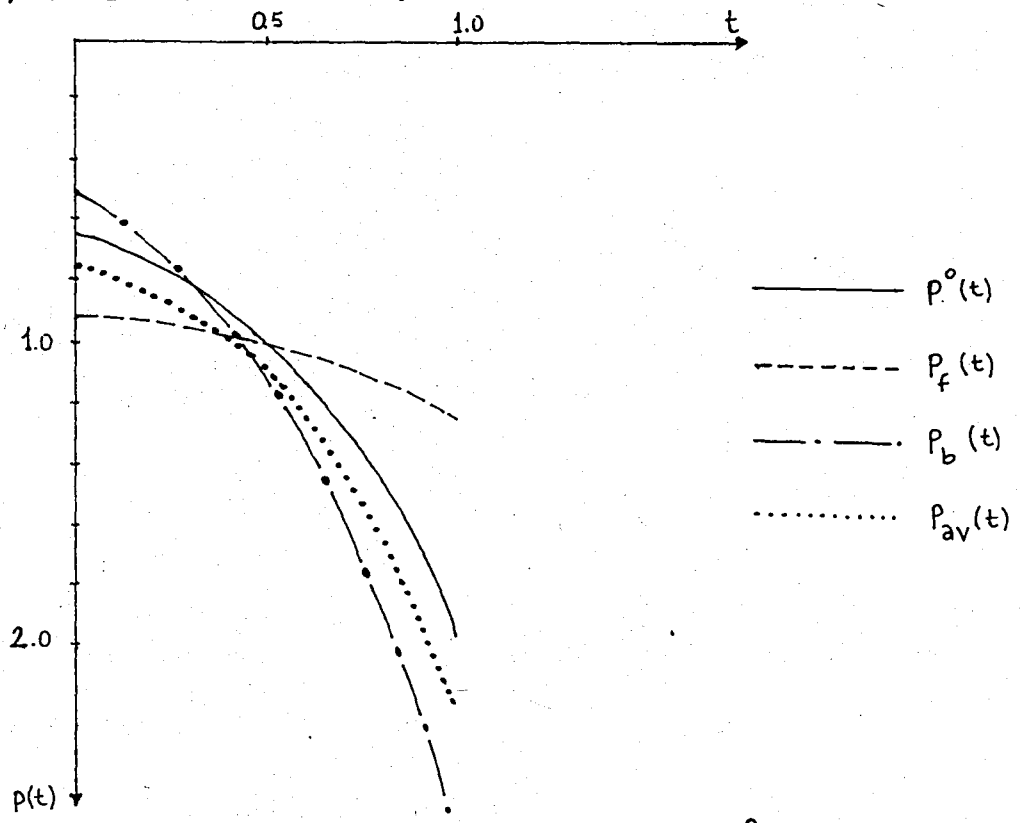

FIGURE 6.2.1. The graph of the functions $p^0(t), p_f(t), p_b(t), p_{av}(t$

Therefore, if we take the average of the coefficients of the same power of the functions $p_f(t)$ and $p_b(t)$ it is certain that one will usually obtain polynomials which approximate the optimal controller parameter function $p(t)$ much better than either one of the functions $p_f(t)$ and $p_b(t)$.

In this special case, the average polynomial function is given by

$$p_{av}(t) = - 0.709 - 0.394\ t - 1.455\ t^2$$

The next example considers the open-loop function of again a first-order linear system with quadratic performance index

Example 2 :

Given

$$\dot{x} = - x + u \qquad x(0) = 0$$

$$J = \tfrac{1}{2} \int_0^1 u^2(t)\ dt - x(1)$$

Find $u(t)$ to minimize J

The analytic solution is given by

$$u^0(t) = e^{t-1}$$

In forward sequential procedure the control function is obtained to be

$$u_f(t) = 0.637 + 0.161\ t + 0.081\ t^2$$

and the value of the performance index functional is found to be -0.208 for this function.

When backward sequential procedure is employed the control function found to be

$$u_b(t) = 0.135 + 0.108\ t + 1.349\ t^2$$

with -0.192 as the value of the performance index.

The average polynomial of $u_f(t)$ and $u_b(t)$ is given by

$$u_{av}(t) = 0.386 + 0.135\ t + 0.715\ t^2$$

The Figure (6.2.2) shows the graphs of the four functions $u^0(t)$, $u_f(t)$, $u_b(t)$, $u_{av}(t)$.

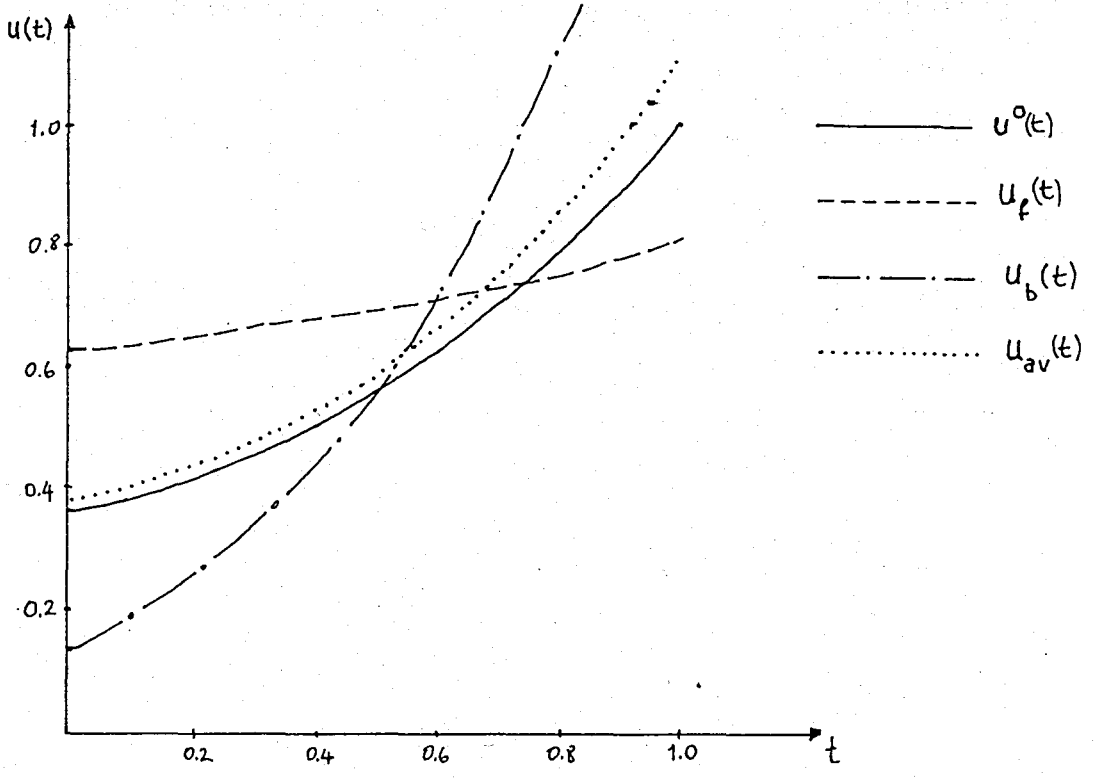

FIGURE 6.2.2. The graphs of the functions $u^0(t), u_f(t), u_b(t), u_{av}(t)$

6.3. ANOTHER ALTERNATIVE FOR BACKWARD SEQUENTIAL OPTIMIZATION PROCEDURE

Among the various alternatives of apriori polynomial approximation methods, we have employed one which makes use of "shifted polynomials". In this version the polynomial functions chosen for controller parameter vector function $\underline{p}(t)$ are shifted in time either to the right or the left by certain amounts. This "certain amount" can be safely chosen to be the interval length; that is, $t_f - t_0$. This version of backward sequential optimization procedure has been applied to various problems and the results have shown that shifted polynomial functions, in general, would approximate the optimal controller parameter function much better than the polynomials used in the previous section, assuming that (if) the "apriori" chosen polynomial function is shifted in the right direction.

Let us consider now a few examples which proves this claim.

Example 1 :

Given

$$\dot{x} = - x + u$$

$$J = \int_0^1 (x^2 + u^2)\, dt$$

Find $u(\underline{x},t)$ to minimize J

When the related scalar Riccati differential equation is solved one obtains the following optimal control-law

$$u(x,t) = - \frac{e^{-2(t-1)} - e^{2(t-1)}}{(2+1)e^{-2(t-1)} + (2-1)e^{2(t-1)}} \cdot x(t)$$

It has been assumed that controller parameter function is given by the following second-order shifted polynomial

$$p_b^s(t) = \sum_{k=2}^{0} p_k (t-1)^k$$

and the control function determined by the relation

$$u(x,t) = p_b^s(t) \cdot x(t)$$

Then the following controller function is obtained

$$p_b^s(t) = - 0.582 + 1.084\ t - 0.530\ t^2$$

with the value of performance index being 0.386.

When the normal second-order polynomial is used in the same backward iterative optimization procedure, that is,

$$p_b(t) = \sum_{k=2}^{0} p_k t^k$$

then one obtains the following results

$$p_b(t) = - 0.227 - 0.280\ t \mp 0.502\ t^2$$

and the performance index value of 0.403.

Finally, let us state the resulting polynomial when the forward sequential optimization procedure is applied.

$$p_f(t) = -0.388 + 0.231\ t + 0.030\ t^2$$

The value of the performance index is 0.370.

The graphs of all of the related controller parameter functions are shown in Figure (6.3.1).

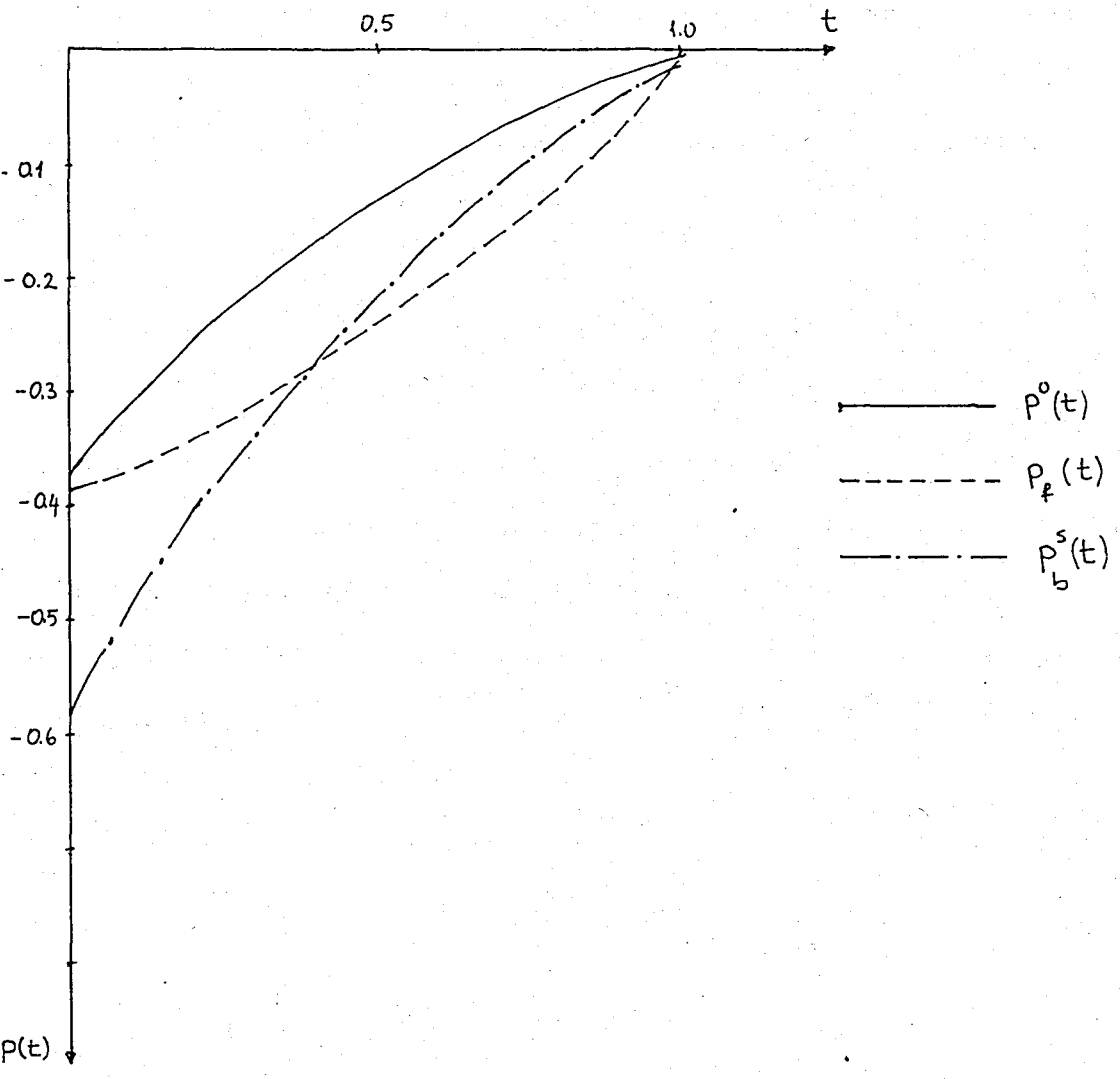

FIGURE 6.3.1. The graphs of the functions $p^o(t), p_f(t), p_b^s(t)$

Example 2 : The same problem as in Example 2 of the previous section.

The form of the shifted polynomial function is as follows :

$$p_b^s(t) = \sum_{k=2}^{0} p_k \, (t+1)^k \quad ,$$

and control function to be determined is directly given by the above polynomial function since only open-loop control function is sought for.

Then the controller parameter function (or the shifted polynomial) has been determined to be

$$p_b^s(t) = 0.280 + 0.437\ t + 0.266\ t^2$$

and the performance index value -0.220.

The graphs of the optimal open-loop control function and the approximating shifted polynomial function are shown in Figure (6.3.2) below in order to give one the opportunity of comparison.

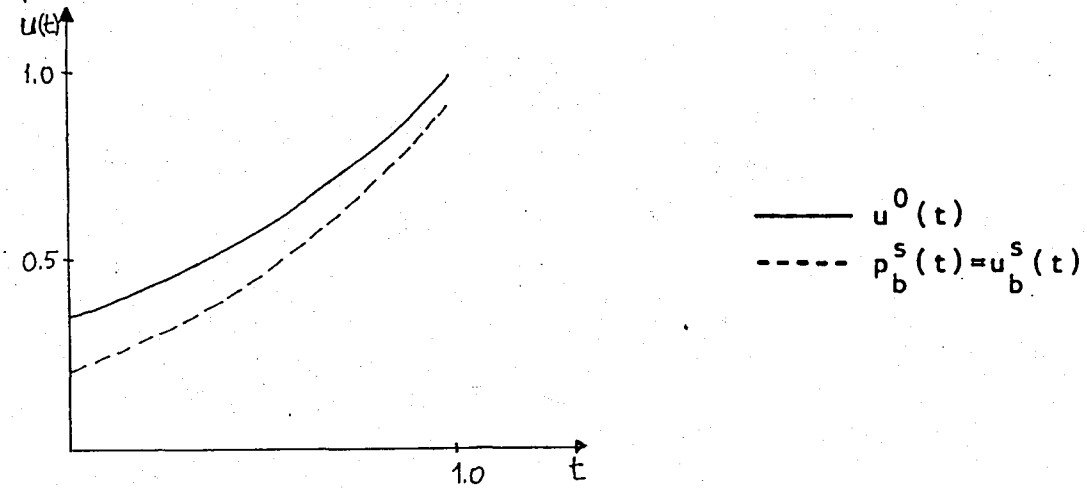

FIGURE 6.3.2. The graphs of the functions $u^0(t)$, $u_b^s(t)$.

Example 3 : A nonlinear system with quadratic performance index.

Given

$$\dot{x} = - x^3 + u \qquad x(0) = 1$$

$$J = \int_0^1 (x^2+u^2)\, dt$$

Find u(t) which minimizes J.

In this case, it is assumed that the optimal open-loop control function is given by the following second-order shifted polynomial

$$u_b^s(t) = \sum_{k=2}^{0} p_k\, (t-1)^k$$

When the backward sequential optimization procedure is employed the following polynomial function is found

$$u_b^s(t) = - 0.483 + 0.885\ t - 0.487\ t^2$$

with a performance index value of 0.497.

VII

APOSTERIORI POLYNOMIAL FITTING METHOD VIA DIRECT SENSITIVITY APPROACH FOR THE GENERAL OPTIMAL CONTROL PROBLEM

7.1. BASIC IDEA OF THE METHOD

In the development and the solution phases of the apriori polynomial approximation methods it has been observed that a large step in approximating the optimal controller parameter function is achieved at the first iteration of the so called general sequential optimization procedure. For instance, when forward sequential optimization procedure is employed, the optimum constant controller parameter function of the first iteration generally approximates the exact controller parameter function quite successfully with regard to performance index values. That is, the contributions of the polynomial functions found in the following iteration to the minimization of the performance index are ususally insignificant.

Moreover, the constant controller parameter function of the first iteration acts as an averager of the unknown optimal controller parameter function. For instance, if the optimal controller parameter function $p^*(t)$ (which, in this case, is assumed to be a scalar function to simplify the illustration) has the shape given in Figure (7.1.1), then the first step of the forward sequential optimization

procedure of the apriori polynomial approximation method would yield a constant function which approximately takes the average of the function $p^*(t)$ as illustrated in the same figure below.

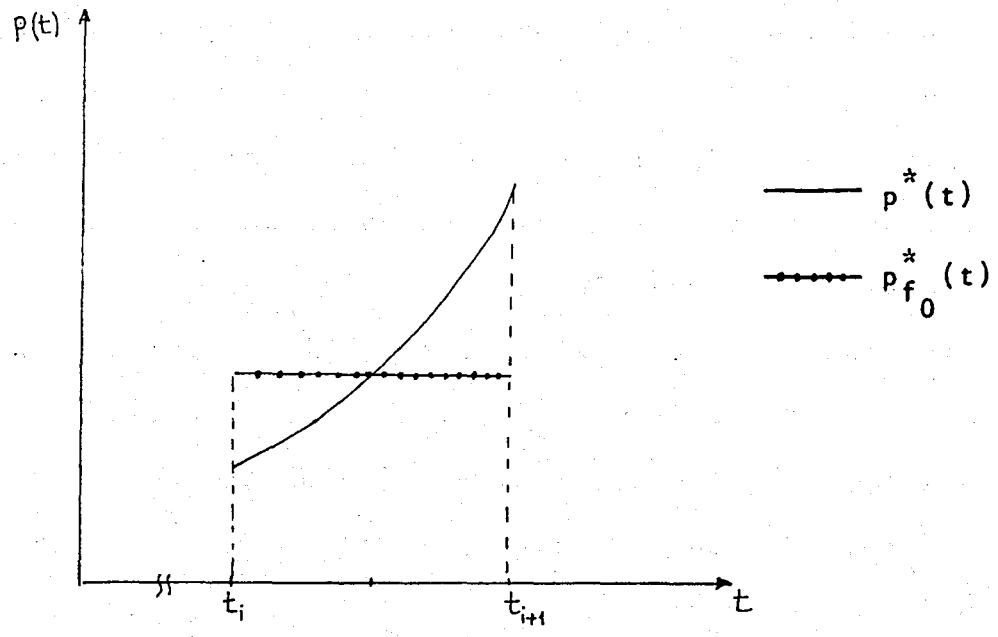

FIGURE 7.1.1. The graph of functions $p^*(t)$ and $p^*_{f_0}(t)$

As it can easily be deduced from the Figure (7.1.1), the smallest difference between the actual optimal controller parameter function $p^*(t)$ and the constant controller function generated by the first iteration of forward sequential optimization procedure would "generally" occur at the midpoint of the time interval $[t_i, t_{i+1}]$.

Therefore, if the time interval $[t_0, t_f]$ specified by the general optimal control problem is divided into some prescribed subintervals "properly" and the forward sequential optimization procedure is applied to these subintervals only for constant functions, then one would obtain some significant

data points on the parameter function space for construction of optimal controller parameter functions more precisely. However, the division of the time interval $[t_0, t_f]$ into subintervals should be made in a special manner because of the inherent properties of the problem under consideration. More precisely, we mean that the division should be done without changing the final time specified by the problem; because changing the specified final time would change the original optimal control problem. This special division of the time interval could be accomplished only if one remembers the fact that the optimal solution is valid for all initial states and times in the closed-loop optimal control problems. Therefore, one can make the division "properly", that is, without changing the original optimal control problem by "sliding" the initial time (and in return initial state) forward at certain amounts determined by the subinterval lengths.

In the light of the facts discussed above one might design a new method to find an approximating function for the optimal controller parameter function. This new method, which we have called "aposteriori polynomial fitting", might be summarized as follows :

(a) Choose a fixed number N which determines the number of subintervals and equivalently the number of significant data points to be obtained as a result of the following procedure.

(b) Set $k = 1$

(c) Find the sliding initial time t_{0_s} from the following equation

$$t_{0_s} = t_0 + \left(\frac{t_f - t_0}{N}\right)(k-1)$$

(d) Perform the forward sequential optimization procedure of the previous chapter using the new (slided) initial time t_{0_s} for only constant function and store the obtained optimum value, say $p_{f_0}^{(k)}$. It is assumed that the optimal controller parameter function passes through the point

$$\left(\frac{t_{0_s} + t_f}{2}, \; p_{f_0}^{(k)}\right)$$

(e) Set $k = k+1$

(f) Check : if $k \leq N$ then go to step (c)
else continue

(g) Pass a polynomial function of any specified order through the data points obtained in the above procedure using one of the fitting methods of numerical analysis; and stop .

There exist one important drawback of the above procedure as it is. The data points which are used in the construction of the approximating function for the optimal controller parameter function were cumulated on the second half of the whole interval $[t_0, t_f]$; that is, on $\left[\frac{t_0 + t_f}{2}, t_f\right]$, because of the intrinsic properties of

the method. Thus, the optimal controller parameter function would be approximated better on the second half of the interval $[t_0, t_f]$. However, this serious drawback of the method can be overcome by simply creating a "virtual" initial time, say t_v, which satisfies the following equation

$$\frac{t_f + t_v}{2} = t_0$$

which implies that

$$t_v = 2\, t_0 - t_f$$

In this way, it is assumed that the whole interval $[t_0, t_f]$ is covered uniformly. The step (c) of the above procedure would then become

(c) Find the sliding initial time t_{0_s} from the equation

$$t_{o_s} = t_v + \left(\frac{t_f - t_v}{N}\right)(k - 1)$$

The new method presented above is basically developed for determining the optimal closed-loop control function. However, it can be easily used in determining the optimal open-loop control function by making one slight modification in the procedure. More precisely, one would have to make a backward integration on the system equations from the specified

initial state to the new state at the virtual initial time since the optimal open-loop control function is dependent on the initial conditions.

7.2. NUMERICAL RESULTS

Several problems with linear or nonlinear system equations and quadratic or nonquadratic performance index criteria are treated for the optimal closed-loop or open-loop solutions using the new "aposteriori polynomial fitting" method developed above; and, quite satisfactory results are obtained in all of the examples. In order to be able to make a comparison of the "apriori" and "aposteriori" methods we have first considered the same examples discussed in the previous chapter. In all of the examples reported below the data points obtained are fitted employing the well-known least-squares approximation method; and virtual initial time and state is used in all of the examples, except in Example 3.

Example 1 :

Given

$$\dot{x} = u$$

$$J = x^2(t_f) + \tfrac{1}{2} \int_0^1 u^2(t)\, dt$$

Find u(x,t) which minimizes J.

(i) First the time interval $[t_v, t_f]$ is divided into

sixteen subintervals. A second-order polynomial is fitted for the resulting sixteen data points on the time interval $[t_0, t_f]$, and the following result is obtained

$$p_d^{(i)}(t) = - 0.688 - 0.004\ t - 1.157\ t^2$$

with a performance index value of 0.329.

(ii) When eight subintervals, and in return, eight data points are used, the second-order polynomial fitted is given by

$$p_d^{(ii)}(t) = - 0.714 + 0.192\ t - 1.413\ t^2$$

with the same performance index value of 0.329.

The two fitted polynomial functions and the optimal solution is shown in the Figure (7.2.1) to make a comparison.

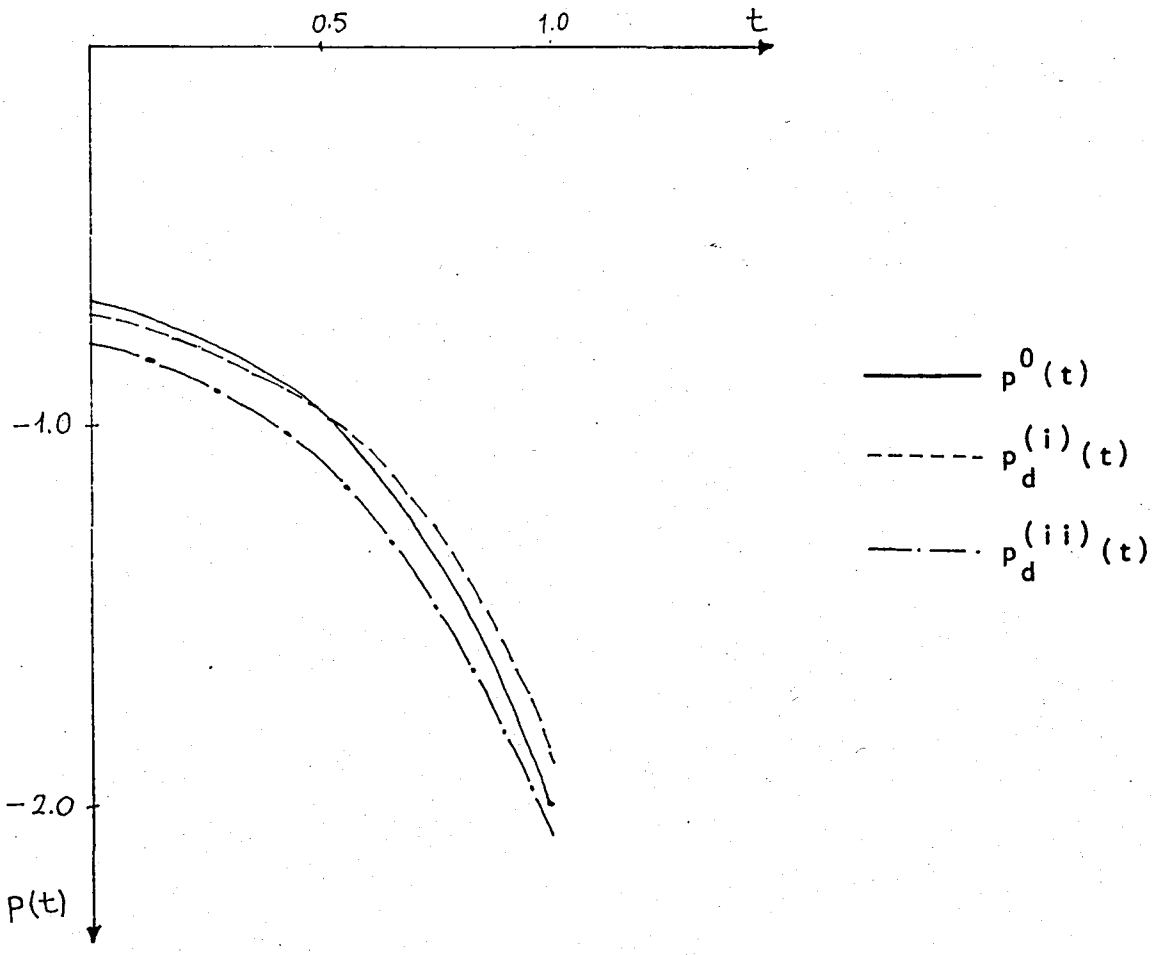

FIGURE 7.2.1. The graphs of the functions $p^0(t)$, $p_d^{(i)}(t)$, $p_d^{(ii)}(t$

Example 2 :

Given

$$\dot{x} = -x + u \qquad x(0) = 0$$

$$J = \tfrac{1}{2} \int_0^1 u^2(t)\, dt - x(1)$$

Find u(t) which minimizes J.

(i) The time interval $[-1, 1]$ is again divided into sixteen subintervals, and the following second-order polynomial function is fitted for the sixteen data points obtained on the time interval $[0, 1]$.

$$u_d^{(i)}(t) = 0.446 + 0.152\ t + 0.456\ t^2$$

The value of the performance index for this control function is equal to -0.220.

(ii) When eight subintervals and in return eight data points are employed, the second-order polynomial is obtained to be

$$u_d^{(ii)}(t) = 0.438 + 0.209\ t + 0.397\ t^2$$

with no significant difference in the performance index value.

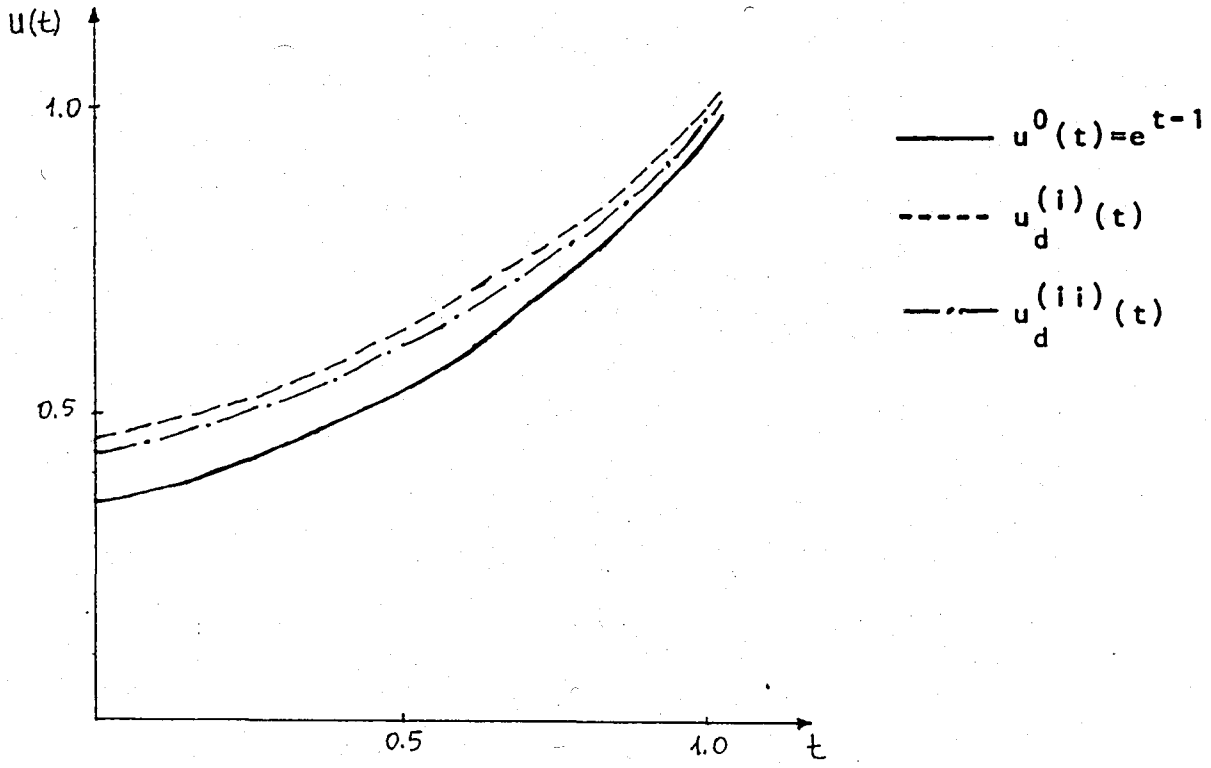

FIGURE 7.2.2. The graps of the functions $u^0(t)$, $u_d^{(i)}(t)$, $u_d^{(ii)}(t)$

Example 3 :

Given

$$\dot{x} = - x^3 + u \qquad x(0) = 1$$

$$J = \int_0^1 (x^2 + u^2)\, dt$$

Find u(t) which minimizes J.

In this case, the time interval $[0, 1]$ is divided into eight subintervals. A second-order polynomial is fitted for the resulting eight data points on the time interval $[0, 5, 1]$, and the following function is obtained

$$u_d(t) = - 0.266 - 0.007\ t + 0.275\ t^2$$

with a performance index value of 0.495.

In order to be able to make a comparison the same nonlinear problem is transformed to a two-point boundary-value problem and solved using the method developed in this study. The resulting two-point boundary-value problem can be stated as

$$\dot{x} = - x^3 - 0.5y \qquad \dot{y} = - 2x + 3yx^2$$

$$x(0) = 1 \qquad y(1) = 0$$

and the optimal control function $u^0(t)$ is given by

$$u^0(t) = -\tfrac{1}{2}\, y(t)$$

The graphs of the three functions $u_d(t)$, $u_b^s(t)$, and $u^0(t)$ is shown in the Figure (7.2.3), where $u_b^s(t)$ is the function obtained in Example 3 of section 6.3.

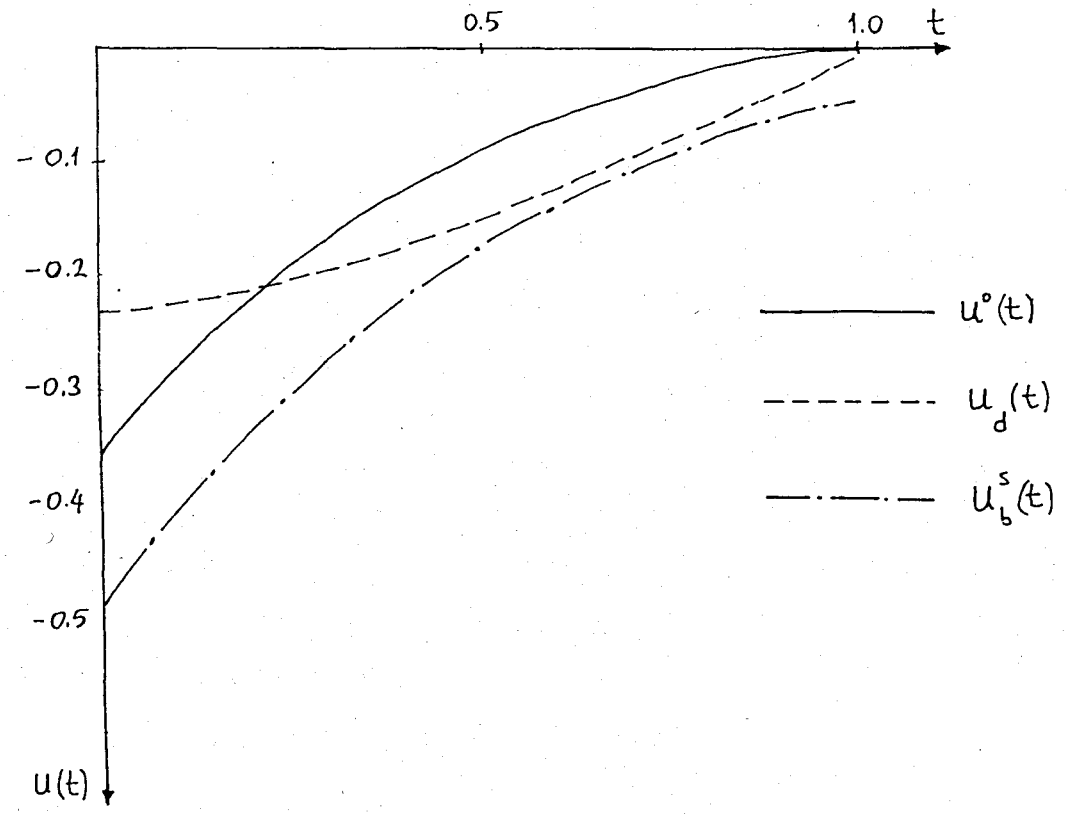

FIGURE 7.2.3. The graphs of the functions $u^0(t)$, $u_d(t)$ and $u_b^s(t)$.

VIII

NUMERICAL SOLUTION OF THE INFINITE HORIZON PROBLEMS IN OPTIMAL CONTROL THEORY

8.1. BASIC IDEA OF THE METHOD

For the problem defined by a linear system dynamics

$$\dot{\underline{x}} = \underline{\underline{A}}\,\underline{x} + \underline{\underline{B}}\,\underline{u}$$

and with a quadratic performance index

$$J = \int_0^\infty (\underline{x}^T\,\underline{\underline{Q}}\,\underline{x} + \underline{u}^T\,\underline{\underline{P}}\,\underline{u})\,dt$$

where Q is symmetric positive semidefinite and P is symmetric positive definite matrices, the Hamilton-Jacobi-Bellman equation, as it is pointed out before, results in the well-known Algebraic Matrix Riccati equation which can be solved in a reasonably simple manner. However, if there exists a simple nonlinearity in system dynamics or if the performance index deviates from the quadratic criteria then one is again faced with the first-order nonlinear partial differential equation of Hamilton-Jacobi-Bellman which is quite difficult to solve. In order to avoid solving this nonlinear partial differential equation one may like to turn back to the approximate solution generating methods presented in the previous

chapters of this study. However, one then faces the difficulty of choosing a sufficient final time value in the numerical evaluation of performance index J. If the system equations were linear then one might choose the sufficient final time value according to a criteria defined by

$$t_f = 10 \frac{1}{|\lambda_{min}|}$$

where λ_{min} is the minimum eigenvalue of the system

$$\dot{\underline{x}} = (\underline{\underline{A}} + \underline{\underline{B}}\ \underline{\underline{K}})\ \underline{x}$$

K is the gain matrix, which is a function of controller parameters. That is, even in the linear case one has to find the eigenvalues of the matrix $(\underline{\underline{A}} + \underline{\underline{B}}\ \underline{\underline{K}})$ at every iteration step, since the controller parameter values will change in the iterative algorithm employed.

For the problem defined by a nonlinear system dynamics

$$\dot{\underline{x}} = \underline{f}\ (\underline{x}, u, t)$$

and with a nonquadratic performance index

$$J = \int_0^\infty l(\underline{x}, u, t)\ dt$$

the calculation of the eigenvalues becomes much more cumbersome.

Here a new and effective algorithm is devised to be used in the calculation of sufficient final time in the case of infinite horizon problems of optimal control theory.

Assuming that the system under study is stabilizable and controllable then we know that J will "saturate" after some fixed time t, call it t_f in infinite horizon problems. A tentative graph for performance index function J versus time t for a stabilizable and controllable system is shown in Figure (8.1.1).

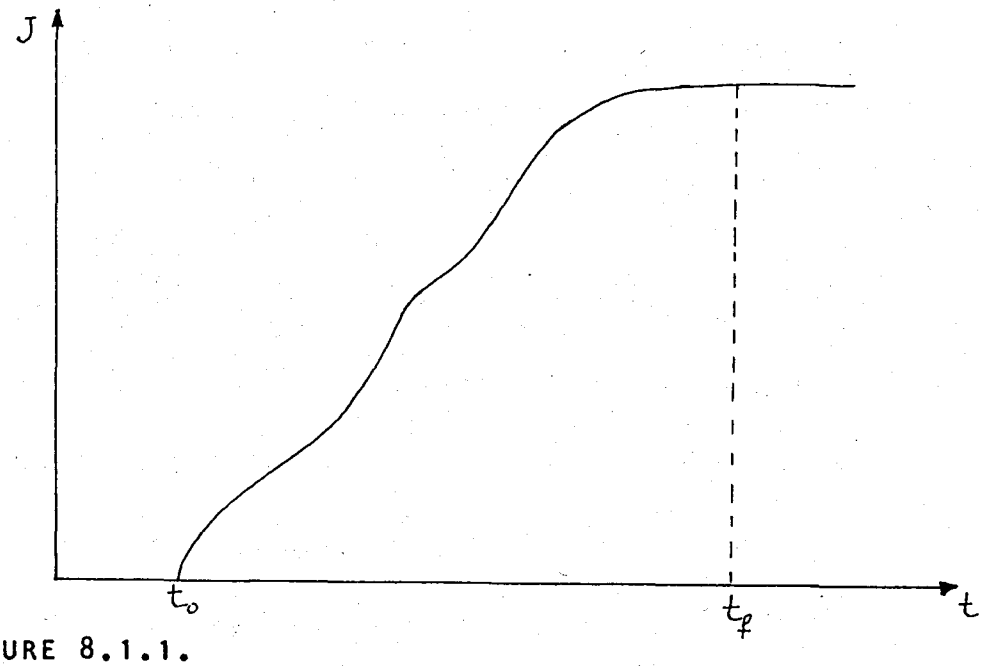

FIGURE 8.1.1.

Now suppose that the performance index J for infinite horizon problems is given by

$$J(t) = \int_0^t l(\underline{x},u,t)\ dt$$

Then, taking the derivative of the above performance index J with respect to t one obtains

$$\frac{dJ}{dt} = l\ (\underline{x},u,t)$$

The performance index function J must be behaving like a constant function around the saturation region; therefore, our aim becomes to find the time where

$$\frac{dJ}{dt} = 0$$

Using the facts developed above one may devise an effective algorithm which will yield the sufficient final time value assuming that one is dealing with a stabilizable and controllable system.

(a) Choose an accuracy δ, and a constant steplength A, a fixed number M for permitted maximum iterations.

(b) Make an initial guess for the sufficient final time t_f.

(c) Devise and initialize a counter $k \leftarrow 0$.

(d) Solve the system equations for the given parameter to obtain the state vector $\underline{x}$ and evaluate the cost function $l(\underline{x},u,t)$ at the final time t_f if $l(\underline{x},u,t)\Big|_{t_f} < \delta$ then sufficient final time found, else perturb t_f by Δt amount $t_{f_p} \leftarrow t_f + \Delta t$

(e) Obtain the new state vector $\underline{x}$ and evaluate the cost function $l^p(\underline{x},u,t)$ at the perturbed final time t_{f_p}.

(f) Form the function S defined by

$$S = \frac{I^p - I}{\Delta t}$$

If $S < 0$ then update t_f by $t_f \leftarrow t_f - S^{-1} I$ and go to step (c)

else $k \leftarrow k + 1$ if $k > M$ then "system unstable for this parameter" change the parameters

else update t_f by $t_f \leftarrow t_f + A$ and go to step (d)

8.2. NUMERICAL RESULTS

Eventhough several infinite horizon problems are solved quite successfully using the methods developed in this study, only two examples with known solutions are reported in order to give one the opportunity of comparison. The first example is a scalar linear system with quadratic performance index, and the other one is a two-dimensional linear system with again quadratic performance index.

Example 1 :

Given

$$\dot{x} = -2x + u$$

$$J = \int_0^{\infty} (x^2 + u^2)$$

Find u(x) to minimize J.

The optimal solution for u(x) is given by

$$u^0(x) = - (\sqrt{5}-2)\ x(t) = - 0.2361\ x(t)$$

The algorithms presented here have yielded the result

$$u(x) = - 0.2339\ x(t)$$

Here, it must be pointed out that Crude-Euler initial-value technique is employed for the solution of differential system equations and trapezoidal integration routine is used for the evaluation of performance index integral J. In the above example all the integrations were performed with a step size of 0.01, and the CPU time was 45 seconds.

Example 2 :

Given

$$\dot{\underline{x}} = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \underline{x} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} u$$

$$J = \int_0^{\infty} (x_1^2 + x_2^2 + 2u^2)\ dt$$

Find u(x) to minimize J.

The optimal solution for u(x) is given by

$$u^0(\underline{x}) = [-\ 0.602 \ -\ 0.333]\ \underline{x}$$

Our algorithm has yielded the following result for $u(\underline{x})$.

$$u(\underline{x}) = [-\ 0.620 \ -\ 0.410]\ \underline{x}$$

In this example, again Crude-Euler initial-value technique and trapezoidal integration routine are employed with a stepsize of 0.05, and the CPU time was 1 minute and 36 seconds. When the stepsize is decreased to 0.025 we have obtained

$$u(\underline{x}) = [-0.610 \ -\ 0.371]\ \underline{x}$$

with a CPU time of 2 minutes and 50 seconds. The improvement in the performance index value for the two different stepsizes is in order of 1 percents.

CONCLUSIONS, GENERAL OVERVIEW AND COMPARISON OF THE METHODS DEVELOPED IN THIS STUDY

First a new method for the numerical solution of the two-point boundary-value problem is proposed. This new method is based on trajectory sensitivities with respect to initial conditions. Since when the two-point boundary-value problem is solved only an open-loop solution for a specific set of initial conditions has been found in the general optimal control problems, an alternate approach which will yield a closed-loop solution is sought for other than the dynamic programming approach which results in Hamilton-Jacobi-Bellman Equation. Here, at this point, we have employed the basic properties of the performance index sensitivities with respect to controller parameter functions and developed a general iterative optimization procedure which, in turn, gives rise to various effective methods.

The first of the methods developed using the direct sensitivity approach was a quite general method called "apriori polynomial approximation methods", and a different version of the same method was produced. In the apriori polynomial approximation methods the basic assumption is that controller parameter function is formed by a polynomial function.

Several problems which have been solved using various versions of the apriori polynomial approximation methods have shown that "shifted" polynomial version would, in general,

yield much better results compared to normal polynomial approximations using either "forward" or "backward" sequential optimization procedure. The only problem with shifted polynomial version is that shifting direction for the polynomial is usually not known apriori by the designer. A sensitivity analysis on this shifting direction parameter could be employed and this version of the apriori polynomial approximation methods could be made almost perfect.

The aposteriori polynomial fitting method which is developed in Chapter 7 is superior than all the existing apriori polynomial approximation methods both in accuracy (or exactness) and computation time. It must be pointed at that no significant drawback of the aposteriori polynomial fitting method has been encountered in the various examples solved upto now.

REFERENCES

1. Frank, P.M. Introduction to System Sensitivity Theory, Academic Press, 1978.
2. Aziz, K.A. Numerical Solution of Boundary-Value Problems for Ordinary Differential Equations, Academic Press, 1975.
3. Meyer, G.H. Initial-Value Methods for Boundary-Value Problems, Academic Press, 1973.
4. Keller, H.B. Numerical Methods for TPBVP, Blaisdell, 1968.
5. Jacobson, Mayne. Differential Dynamic Programming, American Elsevier Pub. Comp., 1970.
6. Falb, De Jong. Some Successive Approximation Methods in Contro and Oscillation Theory, Academic Press, 1968.
7. Fox, L. Numerical Solution of TPBVP, Clarendon Press, 1957.
8. Scott, M.R. Invariant Imbedding, Addison-Wesley Pub. Comp., 1973.
9. Casti, J., Kalaba, R. Imbedding Methods in Applied Mathematics Addison-Wesley Pub. Comp., 1973.
10. Leondes, C.T. Advances in Control Systems, Vol.3-4, Academic Press, 1966.
11. Lapidus,L., Schiesser,W.E. Numerical Methods for Differential Systems, Academic Press, 1976.
12. Beltrami, E.J. An Algorithmic Approach to Nonlinear Analysis and Optimization, Academic Press, 1970.
13. Greenspan, D. Numerical Solutions of Nonlinear Differential Equations, John Wiley & Sons, Inc., 1967.
14. Bryson, A.E., Ho, Yu-Chi. Applied Optimal Control, John Wiley & Sons Inc., 1975.
15. Luenberger, D.G. Optimization by Vector Space Methods, John Wiley & Sons Inc., 1969.
16. Tabak, D., Kuo, B.C. Optimal Control by Mathematical Programming, Prentice-Hall Inc., 1971.

17. Kirk, D.E. Optimal Control Theory, Prentice-Hall Inc.,1970.

18. Bellman, R. Introduction to Matrix Analysis, McGraw Hill, 1970.

19. Cadzow, J.A., Martens, H.R. Discrete-Time and Computer Control Systems, Prentice-Hall Inc., 1970.

20. Schultx, D.G., Melsa, J.L. State Functions and Linear Control Systems, McGraw Hill, 1967.

21. Orava, P.J., Lautala, P.A. "Back and Forth Shooting Method for Solving Two-Point Boundary-Value Problems", Journal of Optimization Theory and Applications, V.18, 1976.

22. Dreyfus, S.E. "The Numerical Solution of Nonlinear Optimal Control Problems" in Ref.(13).

23. Macconi, M., Pasquali, A. "Numerical Solution of Unstable TPBVP by Quasilinearization and Orthonormalization", Journal of Optimization Theory and Applications, V.19, 1976.