A DISCRETE TIME, ON-LINE, IDENTIFICATION AND CONTROL ALGORITHM

A THESIS

Presented to

The Faculty of the Division of Graduate

Studies and Research

by

James M. Fowler, III

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

in the School of Electrical Engineering

Georgia Institute of Technology

April, 1973

A DISCRETE TIME, ON-LINE, IDENTIFICATION AND CONTROL ALGORITHM

Approved:

Chairman, Roger P. Webb

Jay H. Schlag

James E. Brown

Date approved by Chairman: May 23, 1973

ACKNOWLEDGMENTS

To Dr. Roger P. Webb, my thesis advisor, I would like to express my sincerest appreciation for his guidance and encouragement throughout the development of this dissertation. His aid has been invaluable. I also wish to thank Dr. James R. Rowland for helping me pick this thesis topic. Appreciation is also extended to Drs. Jay H. Schlag and James E. Brown for their services as members of my reading committee.

Finally, to my wife Marjorie go my love and deepest appreciation for her love, patience, understanding, and encouragement during my graduate study at Georgia Tech. I would also like to express to her my thanks for working beyond when she would have liked to have quit so that we could afford to continue school.

TABLE OF CONTENTS

TABLE OF CONTENTS (Concluded)

LIST OF TABLES

LIST OF ILLUSTRATIONS

SUMMARY

This dissertation presents a closed-loop, on-line, identification and control scheme for discrete time systems. The control system consists of an identification algorithm and a deterministic controller. The identification algorithm identifies unavailable plant states as well as unknown plant parameters that are either constants, or that vary slowly with time in some unknown manner. The latest identified values are used in recomputing the feedback control.

In this thesis, the application of the above scheme to the control of two important classes of problems is investigated; the linear state regulator system, and the perturbation control of nonlinear systems.

The state regulator problem was chosen not only because of its importance, but also because the calculation of the closed-loop control is well known. It is shown that, if all plant states are available, then the identification algorithm simplifies to a set of linear simultaneous equations which, except for singular or ill-conditioned solutions, converge to the correct answer in one iteration. Methods of handling the singular and ill-conditioned cases are also examined. If all the plant states are not available, then the identification algorithm does not simplify to a set of simultaneous equations and must converge in an iterative fashion to the correct solution. For this case, it is shown that there is a range of initial parameter and plant state estimates sufficient to ensure convergence; however, when convergence occurs, it

occurs in a quadratic manner.

The systems comprising the second class are those which employ a linear perturbation controller to control a nonlinear plant for small deviations from a nominal trajectory. The identification algorithm is used in this case to identify not only all unavailable plant states, but also all the unknown, time-varying, parameters of the linearized system coefficient matrix. The effectiveness of this perturbation control scheme is demonstrated for a practical example. The example chosen concerns the problem of controlling the startup of a thermal nuclear reactor.

CHAPTER I

INTRODUCTION

Introduction and Problem Description

The major contribution of this dissertation is the presentation and thorough evaluation of a closed-loop, on-line, identification and control scheme for discrete time systems.

Parameter identification is a major problem area in control system theory because mathematical models which adequately characterize the dynamic performance of physical systems are essential for most control optimization methods. In some instances there is insufficient a priori information about the system and the topology of the model must be determined from experimental tests, or defined by assuming some empirical form. In other cases, the functional form of the model, and perhaps some of the parameters, are known from theoretical analysis or previous tests. However, in almost all cases, complete specification of the model requires identification of the unknown parameters. The problem is made more difficult when on-line identification is desired, especially if the unknown parameters vary with time. Systems requiring such identification are those whose normal operation is either impossible or impractical to interrupt. For example, such systems include aircraft autopilots, adaptive chemical and industrial process controls, adaptive communication links, and others.

Further complications are encountered when attempts are made to optimize a closed-loop control law according to some cost or performance criterion. The added difficulty arises because the optimized feedback control law is usually a function not only of the unknown parameters, but also of some unavailable plant states. Thus, to find an optimum closed-loop control law, one must find that combination of parameter and state identification and control which satisfies some system performance index. Unfortunately, except for a small class of problems, the complexity of the optimum functional equations obtained which satisfies this combination renders most methods impractical. Consequently, most approaches are directed toward finding suboptimal solutions that can be implemented. These solutions are suboptimal in the sense that some degree of arbitrary separation of the identification and control aspects is involved. A number of these suboptimal solutions have been developed and, depending on the specific characteristics of the system involved, applied with varying degrees of success.

History of the Problem

Historically, the on-line identification and closed-loop control problem has been approached by one of three methods: estimation techniques, identification schemes, or adaptive control procedures. The choice of a particular method has largely been based on the amount of information available about the system and the peculiarities of that system.

The first approach uses estimation techniques to identify the unknown parameters. The identification process is then used in conjunction

with a deterministic controller. Since, as shown by the work of A. A. Fel'dbaun [1], complex results are obtained when trying to optimize both identification and control, work on this problem is generally directed to determining whether a separation principle holds or to finding some suboptimal solution.

The existence of a separation principle allows one to obtain the optimal overall system by cascading an optimum estimation algorithm with the optimum deterministic controller. Numerous articles [2,3,4] have been written which describe conditions for which the separation principle holds; however, in general, for nonlinear plants, very little is known about the structure of the optimal solution. For this reason, useful suboptimal solutions have been sought.

A variety of suboptimal solutions have been developed. For the case when a complete separation of identification and control is assumed, one of the more important techniques is that of least square estimation [5,6]. This scheme has been generalized in numerous articles [7,8] under the heading of "stochastic approximation." For nonlinear systems, another useful suboptimal technique is to expand the nonlinear system equations about some deterministic nominal trajectory [9,10,11]. There are a number of other less important suboptimal estimation techniques [12,13,14,15,16, 17] but each is limited to problems with specific characteristics.

The second class of approaches to the on-line identification and control problem involves extensions of some of the classical process identification schemes; i.e., a test signal is applied to the plant from which, by various methods, identification of the unknown parameters is

achieved [18,19]. Many methods apply an impulse, step, or sinusoidal test input to the system and analyze, by various methods, the corresponding output [20,21,22]. Others introduce either a random or pseudorandom noise input and then cross correlate this with the corresponding output [23,24,25].

Finally, an adaptive control approach is often used to solve the problem. Most adaptive control approaches result in suboptimal performance. Two types of adaptive schemes that relate to the proposed problem are the model reference and learning model approaches. In the model reference approach [26,27,28,29,30,31,32,33], the model is a representation of the desired system. A controller compares the output of the plant with the output of the reference model, and uses the difference to adjust controller parameters to force the plant response to closely match that of the reference model.

The learning model approach is more applicable to the proposed problem than any of the other techniques. This approach makes possible the design of a complete control system for a specific requirement without compromising the design of the controller for the range of parameter variations. In the learning model approach [26,34,35], the controller also compares the output of the plant with the output of a model. However, here the difference is used to adjust the model parameters to cause the model to behave as much like the process as possible. Parameters of the model are, therefore, descriptive of the process and thus may be used in an optimum controller design. To date, the theoretical justification for the design of systems based on this approach has not been sufficiently

well developed to permit extensive analysis related to performance characteristics and limitations. In addition, almost all previous work has been done for analog systems. Most attempts to convert this work to a digital format have led to complicated and inefficient designs [36,37], which do not make effective use of the characteristics of a digital system. Also, previous work has always employed a complete separation of the identification and control functions. Thus, a need exists for a new approach to the design of on-line, discrete-time, identifier-controller systems which make more efficient and optimum use of the digital nature of the problem.

Outline of the Thesis

The major emphasis in this research is not only the development of an identification algorithm, but also combining identification and control to strive to achieve an overall optimum closed-loop control law. Chapter II presents the basic mathematical problem formulation and also the derivation of a general identification algorithm. The proof of the existence of a solution to the identification problem and the classification of systems for which the algorithms developed in this thesis are applicable, are also presented in this chapter.

The combining of identification and control is developed in individual chapters depending upon the class of systems considered and whether only some or all of the plant states are available. Chapter III deals with state-regulator systems in which all of the plant states are available, whereas Chapter IV considers these systems when only some of the plant states are available. In Chapter V, a linear perturbation

control scheme employing the identification algorithm is developed. This scheme is then applied in Chapter VI to control the startup of a thermal nuclear reactor. Finally, Chapter VII summarizes the conclusions and presents recommendations for further work.

CHAPTER II

PROBLEM FORMULATION AND BASIC THEORY

Introduction

In this chapter a new approach to the closed-loop, on-line, identification and control problem described in Chapter I is developed. In the process of developing the basic mathematical formulation, a description of the class of systems for which the algorithms are applicable is also made. The basis of the approach is a general identification algorithm. This algorithm consists of a sequence of functions, where each member of the sequence is determined by split boundary conditions. Since the boundary conditions are split, it is not evident a priori that for nonlinear systems a unique solution actually exists. Thus, the problem of the existence and uniqueness of a solution must be proven.

Basic Problem Formulation

Figure 1 shows a general block diagram of the system considered. Besides the plant, the system is comprised of a parameter adjustable mathematical model, an identification algorithm, and a deterministic controller. The mathematical model is used very much like the learning model discussed in Chapter I. In converging to the proper solution, the parameters of the model are adjusted to cause the model to behave like the plant, so that the parameters of the model are descriptive of the

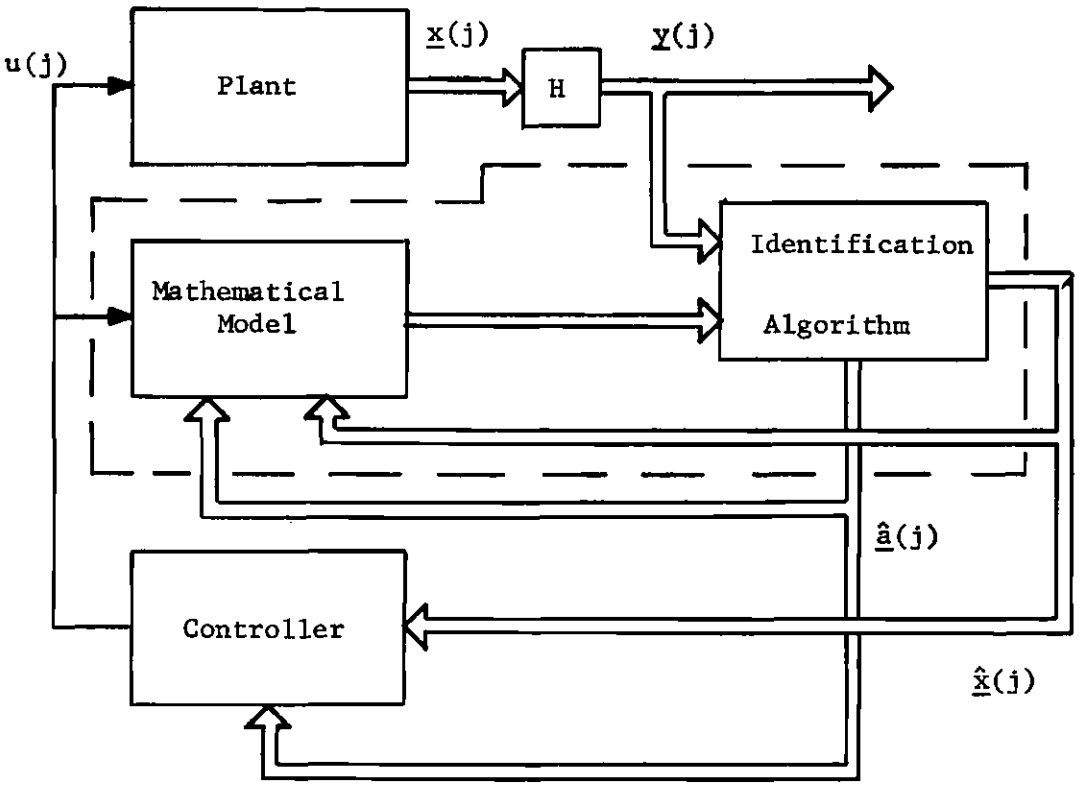

Figure 1. The General System Block Diagram

parameters of the plant. To determine the proper adjustments, the identification algorithm compares the outputs of the model with the outputs available from the plant. Based on this comparison, the algorithm updates its estimates of both the unknown plant states and parameter values. The updated estimates are then used not only to adjust the mathematical model but also to determine the optimum closed-loop control according to some prespecified performance criterion. Because the use of the mathematical model is so closely associated with the identification algorithm, it will be considered as part of the identification algorithm in all future references.

It is assumed that the plant can be described by a vector difference equation of the form

$$\underline{x}(j+1) = \underline{f}[\underline{x}(j),\underline{a}(j),u(j)], \tag{2.1}$$

where $\underline{x}(j)$ represents the n-dimensional state vector, $\underline{a}(j)$ an m-dimensional unknown parameter vector, and $u(j)$ a single control input. The dimensions n and m are assumed known. Equation (2.1) can be either linear or nonlinear but must satisfy the following conditions:

i) $\underline{f}[\underline{x}(j),\underline{a}(j),u(j)]$ is a continuous function of $\underline{x}(j)$, $\underline{a}(j)$, and $u(j)$.

ii) $\underline{f}_{\underline{x}(j)}[\underline{x}(j),\underline{a}(j),u(j)]$ and $\underline{f}_{\underline{a}(j)}[\underline{x}(j),\underline{a}(j),u(j)]$ both exist and are continuous.

iii) $\underline{f}_{\underline{x}(j)}[\underline{x}(j),\underline{a}(j),u(j)]$ satisfies a Lipschitz condition with respect to $\underline{x}(j)$; while $\underline{f}_{\underline{a}(j)}[\underline{x}(j),\underline{a}(j),u(j)]$ satisfies a Lipschitz condition with respect to $\underline{a}(j)$.

Mathematical models which describe most practical systems inherently satisfy the above conditions. Thus these conditions, which are necessary for the mathematical development of the algorithm, do not restrict the applicability of the procedure for most systems.

The only restriction imposed on the parameters of $\underline{a}(j)$ is that they are either unknown constants, or that they are slowly varying with time in an unknown manner.

For many systems, all of the plant states are not available as system outputs. To indicate this, an s-dimensional vector of output measurements is given by

$$\underline{y}(j) = H\,\underline{x}(j), \tag{2.2}$$

where H is an $s \times n$ constant matrix. Since all the plant states may not be available, it must be assumed that the system represented by (2.1) and (2.2) be observable, not only for the identification algorithm to converge to the desired unknown parameters, but also so that the unknown plant states can be computed in order to construct a feedback control.

Since, for practical systems, the control interval is never infinite, the systems considered are not required to be controllable. This arises because the contribution of the uncontrollable states to the performance functional is always finite provided the control interval is finite.

The problem can now be restated as follows: for the plant described by (2.1), with the output measurements given by (2.2), identify the unknown parameter vector $\underline{a}(j)$ and the unavailable plant states, and use the identified values to determine an optimum closed-loop control of the form

$u(j) = \gamma[\underline{x}(j),\underline{a}(j)]$ which minimizes some performance criterion.

There is a large class of systems for which a closed-loop control of the form expressed above can be found. Some of the most important members of this class are linear systems for which optimum closed-loop controls are found that minimize a quadratic performance criterion; i.e., the performance criterion is of the form

$$J = \frac{1}{2} \sum_{j=0}^{j_f-1} \{\underline{x}^T(j)Q\underline{x}(j) + u^2(j)\}, \tag{2.3}$$

where Q is an $n \times n$ matrix that must be at least positive semidefinite. Systems of this type are typically called state-regulators for they tend to keep the states near zero.

If, for a system, a closed-loop control of the form $u(j) = \gamma[\underline{x}(j),\underline{a}(j)]$ is not possible, a possible suboptimal alternative is to use a so-called perturbation control scheme. For this scheme an optimal open-loop control is found satisfying the desired performance criterion. Then a feedback control is developed about the optimal trajectory by minimizing a second cost function which is quadratic in deviation from the nominal trajectory and control, i.e., the deviation functional is of the form

$$J = \frac{1}{2} \sum_{j=0}^{j_f-1} \{\Delta x^T(j)Q\Delta x(j) + \Delta u^2(j)\}. \tag{2.4}$$

Both the state-regulator and perturbation control schemes will be investigated in this thesis.

Throughout this work, a scalar control, u(j), is used. The extension to a vector control, $\underline{u}(j)$, is straightforward, but results not only

in increased complexity in the identification and control algorithms, but also, as shown in the Appendix, an increased use of computer time needed for system simulations.

Identification Algorithm

The identification algorithm is based on a Taylor Series expansion of an equation of the form of equation (2.1). Consider the $(N+1)^{th}$ iteration of such an equation,

$$\underline{x}^{N+1}(j+1) = \underline{f}[\underline{x}^{N+1}(j), \underline{a}^{N+1}(j), u(j)]. \tag{2.5}$$

Expanding (2.5) in a Taylor Series about the N^{th} iteration,

$$\underline{x}^{N+1}(j+1) = \underline{f}[\underline{x}^{N}(j), \underline{a}^{N}(j), u(j)] + \left.\frac{\partial \underline{f}}{\partial \underline{x}}\right|_{\substack{\underline{x}=\underline{x}^{N}(j) \\ \underline{a}=\underline{a}^{N}(j)}} [\underline{x}^{N+1}(j) \tag{2.6}$$

$$- \underline{x}^{N}(j)] + \left.\frac{\partial \underline{f}}{\partial \underline{a}}\right|_{\substack{\underline{x}=\underline{x}^{N}(j) \\ \underline{a}=\underline{a}^{N}(j)}} [\underline{a}^{N+1}(j) - \underline{a}^{N}(j)],$$

where it is assumed that the control, u(j), is known over the identification interval. By considering (2.6) over just one identification cycle, the following equation can be adjoined,

$$\underline{a}^{N+1}(k+1) = \underline{a}^{N+1}(k), \tag{2.7}$$

where

$$k = j - NS(\ell-1). \tag{2.8}$$

The integer NS indicates the number of discrete time intervals per identification period (the calculation of which is shown later), and $\ell = 1,2,\ldots,$ indicates which identification interval is involved. A clearer understanding of the relationship between j, k, and ℓ can be obtained by studying Figure 2. This figure depicts three identification cycle periods for a system in which the integer constant NS = 2. Notice that the integer variable k is used to index the discrete time periods for each identification cycle.

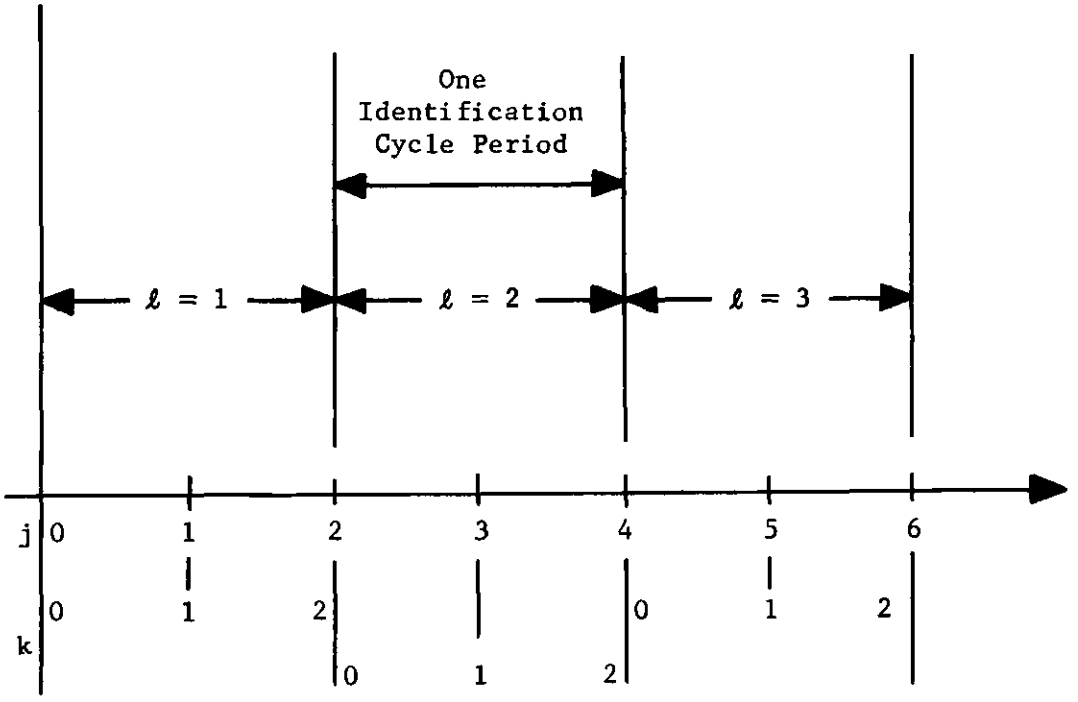

Figure 2. An Example Showing the System's Index Relationships

The set of equations (2.6), (2.7) can now be written in the form

$$\underline{z}^{N+1}(k+1) = D^{N}(k)\underline{z}^{N+1}(k) + \underline{v}^{N}(k), \tag{2.9}$$

where

$$\underline{z}^{N+1} = \begin{bmatrix} \underline{x}^{N+1} \\ \underline{a}^{N+1} \end{bmatrix}, \qquad D^N(k) = \begin{bmatrix} \left.\frac{\partial \underline{f}}{\partial \underline{x}}\right|_N & \left.\frac{\partial \underline{f}}{\partial \underline{a}}\right|_N \\ 0 & I \end{bmatrix},$$

and

$$\underline{v}^N(k) = \begin{bmatrix} \underline{f}[\underline{x}^N(k), \underline{a}^N(k), u(k)] - \left.\frac{\partial \underline{f}}{\partial \underline{a}}\right|_N \underline{a}^N(k) - \left.\frac{\partial \underline{f}}{\partial \underline{x}}\right|_N \underline{x}^N(k) \\ \underline{0} \end{bmatrix}.$$

The solution of equation (2.9), subject to the appropriate boundary conditions, solves the identification problem. The boundary conditions in this case consist of the available plant outputs. Consider one identification cycle in the process. Given as initial conditions are the plant states that are available and either an initial guess of the value of $\underline{a}$ or the value of $\underline{a}$ obtained from the previous cycle. The solution of (2.9) is given by (labeling the initial time as $k=k_0$)

$$\underline{z}^{N+1}(k) = \phi^{N+1}(k,k_0)\underline{z}^{N+1}(k_0) + \underline{p}^{N+1}(k), \tag{2.10}$$

where $\phi^{N+1}(k,k_0)$ represents the homogeneous solution and $\underline{p}^{N+1}(k)$ represents the particular solution. The homogeneous solution is the fundamental solution matrix of

$$\phi^{N+1}(k+1,k_0) = D^N(k)\phi^{N+1}(k,k_0) \tag{2.11}$$

$$\phi^{N+1}(k_0,k_0) = I \ ;$$

whereas, the particular solution is generated by the equation

$$\underline{p}^{N+1}(k+1) = D^{N}(k)\underline{p}^{N+1}(k) + \underline{v}^{N}(k) \tag{2.12}$$

$$\underline{p}^{N+1}(k_0) = \underline{0}.$$

The initial condition vector $\underline{z}^{N+1}(k_0)$ is obtained from the boundary conditions by solving

$$<\underline{C}_{ij}(k_i),\ [\phi^{N+1}(k_i,k_0)\underline{z}^{N+1}(k_0) + \underline{p}^{N+1}(k_i)]> = b_{ij}, \tag{2.13}$$

where $k_i = k_0,k_1,\ldots,k_f$; $ij = 1,2,3,\ldots,(n+m)$; b_{ij} denotes a boundary condition, which are the available plant state outputs; and $<,>$ denotes an inner product. The vector $\underline{C}_{ij}$ is used to select those elements of $[\phi^{N+1}(k_i,k_0)\underline{z}^{N+1}(k_0) + \underline{p}^{N+1}(k_i)]$ which represent the appropriate boundary conditions. For example, if only the plant state $x_{p_1}(k)$ was available as a boundary condition, then

$$\underline{C}_1^{T}(k_i) = \underbrace{[1\ 0\ 0\ \cdots\ 0]}_{n+m}.$$

The latest unknown parameter and unavailable plant state estimates obtained from solving (2.9) subject to the boundary conditions of (2.13), are used to update the value of the closed-loop control according to some performance index. The mathematical details of this updating are reserved for the next three chapters where they can be developed more thoroughly.

Existence and Uniqueness

No mention has been made concerning the important question of whether a unique solution actually exists when solving nonlinear plant equations. With the split boundary conditions depicted in equation (2.13), it is not immediately evident that there is such a solution. Therefore, in this section, a proof of the existence and uniqueness of a solution is given.

Consider adjoining the plant equation, equation (2.1), and an equation representing the actual variations of the unknown parameters over an identification cycle; i.e., let

$$\underline{z}(k+1) = \underline{g}[\underline{z}(k),u(k)], \tag{2.14}$$

where

$$\underline{z}(k) = \begin{bmatrix} \underline{x}(k) \\ \\ \underline{a}(k) \end{bmatrix} .$$

Assume that $\underline{a}(k)$ is continuous. From previous assumptions, it is obvious that

1) $\underline{g}[\underline{z}(k),u(k)]$ is a continuous function of $\underline{z}(k)$ and $u(k)$.

2) $\underline{g}_{\underline{z}(k)}[\underline{z}(k),u(k)]$ exists and is continuous.

3) $\underline{g}[\underline{z}(k),u(k)]$ satisfies a Lipschitz condition with respect to $\underline{z}(k)$; i.e., there exists a number M such that

$$\| \underline{g}[\underline{z}_1(k),u(k)] - \underline{g}[\underline{z}_2(k),u(k)] \| \leq M \| \underline{z}_1(k) - \underline{z}_2(k) \| ,$$

where $\| \cdot \|$ denotes a norm.

The proof of the existence of a unique solution is based on the use of the principle of contraction mappings. This principle gives a

sufficient condition that the nonlinear operator equation

$$\underline{z} = T\underline{z} \tag{2.15}$$

has a unique solution. In what follows, no attempts shall be made to be any more general than is necessary.

The vector $\underline{z}$ belongs to a set of vectors comprising a complete normed linear space S. An operator T mapping the normed space S into itself is said to be a contraction mapping on the space if there is a number α, $0 < \alpha < 1$, such that for all $\underline{z}$, $\underline{\theta} \in S$

$$\| T\underline{\theta} - T\underline{z} \| \leq \alpha \| \underline{\theta} - \underline{z} \|. \tag{2.16}$$

Obviously, every contraction mapping is continuous, for if $\underline{z}^N \to \underline{z}$ (meaning $\|\underline{z}^N - \underline{z}\| \to 0$), then equation (2.16) implies $\|T\underline{z}^N - T\underline{z}\| \to 0$ as $N \to +\infty$; i.e., $T\underline{z}^N \to T\underline{z}$.

Theorem I: The Principle of Contraction Mappings

Every contraction mapping T defined on a complete normed linear space S has one and only one fixed point (i.e., $\underline{z} = T\underline{z}$ has exactly one solution).*

The solution of equation (2.14) can be put in the form

$$\underline{z}(k) = \sum_{r=0}^{k-1} \underline{h}[\underline{z}(r)]. \tag{2.17}$$

Thus, $T\underline{z}$ in this case is defined by the right side of equation (2.17).

*The proof can be found on page 43 of a book by Kolmogorov and Fomin [38].

If T is a contraction mapping on the space S, then Theorem I provides the required proof of existence and uniqueness.

It is thus necessary to determine when T is a contraction mapping on S. First, consider the equation

$$T\underline{\theta}(k) - T\underline{v}(k) = \sum_{r=0}^{k-1} \{\underline{h}[\underline{\theta}(r)] - \underline{h}[\underline{v}(r)]\}. \tag{2.19}$$

Taking the norm of both sides, the following equation results:

$$\| T\underline{\theta}(k) - T\underline{v}(k) \| \leq k \| \underline{h}[\underline{\theta}(k)] - \underline{h}[\underline{v}(k)] \|. \tag{2.20}$$

Then, if the Lipschitz condition is used,

$$\|T\underline{\theta}(k) - T\underline{v}(k)\| \leq k M \| \underline{\theta}(k) - \underline{v}(k)\|. \tag{2.21}$$

Equation (2.21) is the same form as equation (2.16) provided $\alpha = k M$. If $k M < 1$, the mapping is contracting; hence, the existence of a unique answer is proved. In a later chapter, the convergence of the linearized identification algorithm, equation (2.9), to the unique solution of (2.14) will be proven using the above results.

CHAPTER III

THE STATE REGULATOR SYSTEM WITH ALL PLANT STATES AVAILABLE

Introduction

This chapter examines the identification and control of state regulator systems in which all of the plant states are available as system outputs. It will be shown that, for this case, the identification algorithm simplifies to a set of simultaneous algebraic equations which, except for singular or ill-conditioned solutions, converge to the correct answer in one iteration. The singular and ill-conditioned cases are also examined and two methods of handling these problems are discussed. Computer simulated examples are included to verify the analytical work.

Mathematical Development

Let the linear system be represented by

$$\underline{x}(j+1) = A(\underline{a})\underline{x}(j) + \underline{b}u(j) \tag{3.1}$$

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

where $A(\underline{a})$ is an $n \times n$ matrix containing, as some of the elements, the unknown parameters, and $\underline{b}$ is an n-dimensional vector. Since all of the plant states are available as outputs, the output measurement vector becomes

$$\underline{y}(j) = \underline{x}(j). \tag{3.2}$$

For the quadratic cost functional

$$J = \frac{1}{2} \sum_{j=0}^{j_f-1} \{\underline{x}^T(j)Q\underline{x}(j) + u^2(j)\}, \tag{3.3}$$

the derivation of the optimum closed-loop control can be found in many sources [39,40]. The resulting control is

$$u(j) = \underline{b}^T[A^{-1}(\underline{a})]^T[P(j) - Q]\ \underline{x}(j), \tag{3.4}$$

where

$$P(j) = Q + A^T(\underline{a})P(j+1)A(\underline{a}) - \frac{A^T(\underline{a})P(j+1)\underline{b}\underline{b}^T P(j+1)A(\underline{a})}{[1 + \underline{b}^T P(j+1)\underline{b}]}, \tag{3.5}$$

with $P(j_f) = [0]$. Equation (3.5), called the Riccati equation, is solved backward in time from j_f, with each iteration being stored for use in equation (3.4) at the appropriate time. Using (3.4) in (3.1), the system trajectories are represented by the equation

$$\underline{x}(j+1) = \{A(\underline{a}) - \underline{b}\underline{b}^T[A^{-1}(\underline{a})]^T[P(j) - Q]\}\ \underline{x}(j). \tag{3.6}$$

Note that the Riccati equation (equation 3.5), the control (equation 3.4), and the system trajectories (equation 3.6) are all a function of the unknown parameter vector $\underline{a}$. In view of this fact, a description of the system operation is as follows. First, the unknown parameter vector $\underline{a}$ is identified. Using this latest estimate of $\underline{a}$, the Riccati equation is reiterated backward in time to yield an updated Riccati gain.

This gain is then used to recompute the optimum closed-loop control. The system trajectories can next be computed from (3.6). In this manner, the control used is always based on the latest estimates of the unknown parameters.

It was stated in the introduction that the identification algorithm simplifies to a set of simultaneous algebraic equations. These equations are derived in the following manner. Consider the $[\underline{x}^{N+1}(j) - \underline{x}^{N}(j)]$ term in equation (2.6). Since all of the plant states are available at each discrete time interval, this term is equal to zero. Then, by adjoining the unknown parameter vector to the plant states over an identification cycle as was done in Chapter II, equations (2.6), (2.7) can be rewritten as

$$\underline{q}^{N+1}(k+1) = F^{N}(k)\underline{q}^{N+1}(k) + \underline{w}^{N}(k), \tag{3.7}$$

where

$$\underline{q}^{N+1} = \begin{bmatrix} \underline{x}^{N+1} \\ \underline{a}^{N+1} \end{bmatrix}, \qquad F^{N}(k) = \left[\begin{array}{c|c} 0 & \left.\dfrac{\partial \underline{f}}{\partial \underline{a}}\right| \\ \hline 0 & I \end{array}\right]_{N},$$

and

$$\underline{w}^{N}(k) = \begin{bmatrix} \underline{f}[\underline{x}^{N}(k), \underline{a}^{N}(k)] - \left.\dfrac{\partial \underline{f}}{\partial \underline{a}}\right|_{N} \underline{a}^{N}(k) \\ \underline{0} \end{bmatrix}.$$

As before, the solution of (3.7), subject to the appropriate boundary conditions, solves the identification problem. The solution is given by an equation similar to (2.10), and is

$$\underline{q}^{N+1}(k) = \phi^{N+1}(k,k_0)\underline{q}^{N+1}(k_0) + \underline{p}^{N+1}(k), \tag{3.8}$$

where again $\phi^{N+1}(k,k_0)$ represents the homogeneous solution and $\underline{p}^{N+1}(k)$ represents the particular solution.

For this case, identification can be accomplished by considering only one discrete time interval. In considering such an interval, the homogeneous solution equation, (2.11), simplifies to

$$\phi^{N+1}(k_0+1,k_0) = F^N(k_0)\phi^{N+1}(k_0,k_0) = F^N(k_0), \tag{3.9}$$

and the particular solution equation, (2.12), also simplifies to

$$\underline{p}^{N+1}(k_0+1) = F^N(k_0)\underline{p}^{N+1}(k_0) + \underline{w}^N(k_0) = \underline{w}^N(k_0). \tag{3.10}$$

Using (3.9) and (3.10) in (3.8), the solution becomes

$$\underline{q}^{N+1}(k_0+1) = \phi^{N+1}(k_0+1,k_0)\underline{q}^{N+1}(k_0) + \underline{p}^{N+1}(k_0+1),$$

or

$$\underline{q}^{N+1}(k_0+1) = F^N(k_0)\underline{q}^{N+1}(k_0) + \underline{w}^N(k_0). \tag{3.11}$$

The term $\underline{q}^{N+1}(k_0)$ above is obtained from the boundary conditions by solving

$$< \underline{c}_{hj}(i), [\phi^{N+1}(i,k_0)\underline{q}^{N+1}(k_0) + \underline{p}^{N+1}(i)] > = b_{hj}(i), \tag{3.12}$$

where $i = k_0, k_0+1$; $hj = 1,2,\ldots,n$; and $b_{hj}(i)$ denotes a boundary condition. For $i = k_0$, equation (3.12) yields no new information, since the boundary conditions at k_0 have already been incorporated into the equations by setting $\underline{x}^{N+1}(k_0) = \underline{x}^{N}(k_0) = \underline{x}_p(k_0)$, where $\underline{x}_p(k_0)$ are the output plant states at k_0. However, for $i = k_0+1$, equation (3.12) yields a very valuable result; namely, that the $[\cdot]$ term is the same as the argument of difference equation (3.11). Thus, with $b_{hj}(k_0+1)$ equal to the plant states at time (k_0+1), the vector $\underline{C}_{hj}(k_0+1)$ chosen to yield those equations of (3.12) which are equal to the plant outputs, and with the definitions of $F^N(k_0)$, $\underline{q}^{N+1}(k_0)$, and $\underline{w}^N(k_0)$ previously given, equation (3.12) with $i = k_0+1$ simplifies to

$$\left.\frac{\partial \underline{f}}{\partial \underline{a}}\right|_{\substack{\underline{x}=\underline{x}^N(k) \\ \underline{a}=\underline{a}^N(k)}} \underline{a}^{N+1}(k_0) + \underline{f}[\underline{x}^N(k_0), \underline{a}^N(k_0)] - \left.\frac{\partial \underline{f}}{\partial \underline{a}}\right|_{\substack{\underline{x}=\underline{x}^N(k) \\ \underline{a}=\underline{a}^N(k)}} \underline{a}^N(k_0) = \underline{x}_p(k_0+1),$$

or

$$\left.\frac{\partial \underline{f}}{\partial \underline{a}}\right|_{\substack{\underline{x}=\underline{x}^N(k) \\ \underline{a}=\underline{a}^N(k)}} \underline{a}^{N+1}(k_0) = \underline{x}_p(k_0+1) + \left.\frac{\partial \underline{f}}{\partial \underline{a}}\right|_{\substack{\underline{x}=\underline{x}^N(k) \\ \underline{a}=\underline{a}^N(k)}} \underline{a}^N(k_0) - \underline{f}[\underline{x}^N(k_0), \underline{a}^N(k_0)] . \qquad (3.13)$$

Equation (3.13) is of the form

$$G^N(k)\underline{a}^{N+1}(k) = \underline{d}^N(k) \qquad (3.14)$$

which, provided $G^N(k)$ is nonsingular, yields for the unknown parameter vector

$$\underline{a}^{N+1}(k) = [G^N(k)]^{-1}\underline{d}^N(k). \qquad (3.15)$$

Since $\underline{a}$ is an m-dimension vector while $\underline{x}$ is an n-dimension vector, the ability to solve (3.15) for $\underline{a}^{N+1}(k)$ depends not only on whether the rank of $G^N(k)$ is m, but also on the dimension of the plant states. If $n = m$ and the rank of $G^N(k)$ is m, $G^N(k)$ will be nonsingular and no difficulty will be encountered in solving equation (3.15). If $n > m$, the system of equations represented by equation (3.14) will, in general, contain (n-m) dependent equations. This system must be reduced to minimal order (rank = m) in order for equation (3.15) to be solved. This is accomplished by considering just the first m independent equations available. If $n < m$, the system of equations represented by (3.14) will not contain enough independent equations to be solved. However, since $\underline{a}^{N+1}(k+1) = \underline{a}^{N+1}(k)$ and $\underline{x}^N(k_0+1) = \underline{x}^{N+1}(k_0+1) = \underline{x}_p(k_0+1)$, additional independent equations are made available by adjoining the equations obtained from the next discrete time period. Additional equations must be generated in this manner until m independent equations have been obtained, thus allowing equation (3.15) to be solved in a straightforward manner.

To provide some insight into the amount of computational time required to implement the identification and control scheme, and hence a ballpark estimate of the allowable system bandwidth, the analysis of the Appendix is presented. In the Appendix, the computation time required to implement one identification cycle is obtained as an explicit function of the dimensions of the system's state, unknown parameter, and control vectors. As an example, a third-order system is considered with three unknown parameters and a second-order control. It is shown that a computation time of approximately 0.08 sec is required for the implementation—

a figure compatible with the on-line implementation claim. This also suggests that, for this example, the system Nyquist rate is approximately 6.25 radians/sec. When only a scalar control is used, the computation time is cut to 0.047 sec.

Convergence Analysis

In this section, it will be demonstrated that, except for singular or ill-conditioned solution, the identification algorithm converges to the correct solution in one iteration. For a general system configuration, with unknown parameters possible in any or all elements of the system matrix, a general convergence proof is not possible. However, for a specific configuration with a priori assigned locations of unknown parameters, a convergence proof can be given. The following development exemplifies establishment of convergence for a specific system.

The system chosen features the case where $n < m$, so that additional equations must be adjoined from the next discrete time period in order to solve the identification algorithm. The true plant parameters and states are indicated with a subscript P; whereas the superscript indicates the iteration number. Standard matrix element row-column notation is also employed. Consider the following second order system model that contains, as unknown parameters, the elements $a_{11}^{N}(k)$, $a_{12}^{N}(k)$, and $a_{21}^{N}(k)$,

$$x_1^{N}(k+1) = a_{11}^{N}(k)x_{P_1}(k) + a_{12}^{N}(k)x_{P_2}(k) + b_1u(k) \tag{3.16}$$

$$x_2^{N}(k+1) = a_{21}^{N}(k)x_{P_1}(k) + a_{P_{22}}(k)x_{P_2}(k) + b_2u(k).$$

Since m = 3, one more equation must be obtained from the next time period and is, since $\underline{a}^N(k+1) = \underline{a}^N(k)$,

$$x_1^N(k+2) = a_{11}^N(k)x_{P_1}(k+1) + a_{12}^N(k)x_{P_2}(k+1) + b_1u(k+1). \qquad (3.17)$$

Using (3.16) and (3.17) in the identification algorithm (3.13), the following is obtained

$$\begin{bmatrix} x_{P_1}(k_0) & x_{P_2}(k_0) & 0 \\ 0 & 0 & x_{P_1}(k_0) \\ x_{P_1}(k_0+1) & x_{P_2}(k_0+1) & 0 \end{bmatrix} \begin{bmatrix} a_{11}^1(k_0) \\ a_{12}^1(k_0) \\ a_{21}^1(k_0) \end{bmatrix} \qquad (3.18)$$

$$= \begin{bmatrix} a_{P_{11}}(k_0)x_{P_1}(k_0) + a_{P_{12}}(k_0)x_{P_2}(k_0) + b_1u(k_0) \\ a_{P_{21}}(k_0)x_{P_1}(k_0) + a_{P_{22}}(k_0)x_{P_2}(k_0) + b_2u(k_0) \\ a_{P_{11}}(k_0)x_{P_1}(k_0+1) + a_{P_{12}}(k_0)x_{P_2}(k_0+1) + b_1u(k_0+1) \end{bmatrix}$$

$$+ \begin{bmatrix} x_{P_1}(k_0) & x_{P_2}(k_0) & 0 \\ 0 & 0 & x_{P_1}(k_0) \\ x_{P_1}(k_0+1) & x_{P_2}(k_0+1) & \end{bmatrix} \begin{bmatrix} a_{11}^0(k_0) \\ a_{12}^0(k_0) \\ a_{21}^0(k_0) \end{bmatrix}$$

$$- \begin{bmatrix} a_{11}^0(k_0)x_{P_1}(k_0) + a_{12}^0(k_0)x_{P_2}(k_0) + b_1u(k_0) \\ a_{21}^0(k_0)x_{P_1}(k_0) + a_{22}^0(k_0)x_{P_2}(k_0) + b_2u(k_0) \\ a_{11}^0(k_0)x_{P_1}(k_0+1) + a_{12}^0(k_0)x_{P_2}(k_0+1) + b_1u(k_0+1) \end{bmatrix},$$

where

$$\left.\frac{\partial \underline{f}}{\partial \underline{a}}\right|_{\substack{\underline{x}=\underline{x}^N(k) \\ \underline{a}=\underline{a}^N(k)}} = \begin{bmatrix} x_{P_1}(k) & x_{P_2}(k) & 0 \\ 0 & 0 & x_{P_1}(k) \\ x_{P_1}(k+1) & x_{P_2}(k+1) & 0 \end{bmatrix}.$$

By combining terms on the right hand side of the above equation, and pre-multiplying both sides by $[\partial \underline{f}/\partial \underline{a}]^{-1}$, equation (3.18) becomes

$$\begin{bmatrix} a_{11}^{1}(k_0) \\ a_{12}^{1}(k_0) \\ a_{21}^{1}(k_0) \end{bmatrix} = \begin{bmatrix} a_{11}^{0}(k_0) \\ a_{12}^{0}(k_0) \\ a_{21}^{0}(k_0) \end{bmatrix} + \tag{3.19}$$

$$\frac{1}{x_{P_1}(k_0)x_{P_2}(k_0)x_{P_1}(k_0+1) - [x_{P_1}(k_0)]^2 x_{P_2}(k_0+1)} \cdot$$

$$\begin{bmatrix} -x_{P_1}(k_0)x_{P_2}(k_0+1) & 0 & x_{P_1}(k_0)x_{P_2}(k_0) \\ x_{P_1}(k_0)x_{P_2}(k_0+1) & 0 & -[x_{P_1}(k_0)]^2 \\ 0 & \begin{matrix} -x_{P_1}(k)x_{P_2}(k_0+1) \\ +x_{P_2}(k_0)x_{P_1}(k_0+1) \end{matrix} & 0 \end{bmatrix} \cdot$$

$$\begin{bmatrix} \{a_{P_{11}}(k_0) - a_{11}^{0}(k_0)\}x_{P_1}(k_0) + \{a_{P_{12}}(k_0) - a_{12}^{0}(k_0)\}x_{P_2}(k_0) \\ \{a_{P_{21}}(k_0) - a_{21}^{0}(k_0)\}x_{P_1}(k_0) \\ \{a_{P_{11}}(k_0) - a_{11}^{0}(k_0)\}x_{P_1}(k_0+1) + \{a_{P_{12}}(k_0) - a_{12}^{0}(k_0)\}x_{P_2}(k_0+1) \end{bmatrix}.$$

Performing the indicated multiplication and combining terms yields the desired results; namely,

$$\begin{bmatrix} a_{11}{}^{1}(k_0) \\ a_{12}{}^{1}(k_0) \\ a_{21}{}^{1}(k_0) \end{bmatrix} = \begin{bmatrix} a_{P_{11}}(k_0) \\ a_{P_{12}}(k_0) \\ a_{P_{21}}(k_0) \end{bmatrix}. \tag{3.20}$$

It should be noted that the same procedure--i.e., writing the model equations as in equation (3.16), finding $\partial \underline{f}/\partial \underline{a}$, then using both in the identification equation (3.13)--can be used to show one step convergence for other specific system configurations.

Singular and Ill-Conditioned Systems

Analysis of the example used in the last section shows that, if the term

$$x_{P_1}(k)x_{P_2}(k)x_{P_1}(k+1) - [x_{P_1}(k)]^2 x_{P_2}(k+1) \tag{3.21}$$

(which is the value of the determinant of $\partial \underline{f}/\partial \underline{a}$) is equal to zero, then the matrix $\partial \underline{f}/\partial \underline{a}$, hence $G^N(k)$ in (3.14), is singular and equation (3.14) cannot be solved to find $\underline{a}^{N+1}(k)$.

If the value of (3.21) is near zero, the matrix is considered to be ill-conditioned. Ill-conditioning occurs when the rows or columns of (3.14) are nearly dependent. Trouble is encountered when working with ill-conditioned equations in that small errors in the value of the elements in the coefficient matrix $G^N(k)$, or in the elements of the vector

$\underline{d}^N(k)$, can cause large errors in the solution vector. These small errors can be introduced in a number of ways. For example, the value of the elements of the coefficient matrix are usually known only to a certain degree of accuracy, and this tolerance may not be small enough to prevent large errors in the solution. Even if known exactly, many numbers cannot be correctly represented when stored in a computer. Other sources of error include roundoff errors produced in the process of forming equation (3.14) and inaccuracies and noise encountered in the measuring of the plant states.

Since difficulties are encountered in applying the identification algorithm to both singular and ill-conditioned sets of equations, consideration must be given to methods that circumvent these types of problems. Two such methods are discussed. The method implemented throughout this thesis is based on an approach in which an attempt is made to solve the set of equations represented by (3.14) using a Gauss-Jordan elimination scheme with maximal pivoting by columns. However, before the complete solution is found, the value of the determinant of G is calculated. Since the Gauss-Jordan scheme diagonalizes the left hand side of equation (3.14), the value of the determinant is merely the product of the transformed diagonal g elements.

If the matrix G is singular (which implies that at least two equations of (3.14) are linearly dependent) then the transformed diagonal g elements of all but one of the dependent equations will be zero. Likewise, if the matrix G is nearly singular (ill-conditioned) the transformed diagonal g elements of all but one of the nearly dependent equations will be close to zero. A value "close to zero" is somewhat arbitrary. A

general rule of thumb is the following: if the value of the coefficient elements can be measured accurately to ζ decimal places, then $\pm 10^{-\zeta}$ is considered sufficiently close to zero [41]. A value of the matrix determinant of less than $\pm 10^{-4}$ is used in this thesis to indicate an ill-conditioned matrix. Thus, by checking the value of the transformed diagonal g elements, a determination is made not only whether the matrix is singular or ill-conditioned, but also which equations are causing the trouble. Once detected, the dependent or nearly dependent equations are removed from the set (3.14). Since $\underline{a}^{N+1}(k+1) = \underline{a}^{N+1}(k)$, new equations from the next discrete time period are added to take the place of those removed. The determinant of the new set is then tested as before. This process is continued until a well-conditioned matrix $G^N(k)$ is found.

This method is not without some drawbacks. For one, adding equations by enforcing $\underline{a}^{N+1}(k+1) = \underline{a}^{N+1}(k)$ increases the time required for one identification period. This, in effect, lowers the allowable time variations of those unknown parameters that are to be identified. In addition, there is a small but finite probability that a set of well-conditioned equations cannot be found over the entire length of time that the system is running. In this case identification of the unknown parameters would never be accomplished. However, in all the simulations used for this thesis, this condition was never encountered.

A second method that can be used to circumvent the singular or ill-conditioned matrix problem is to use a generalized inverse to obtain $[G^N(k)]^{-1}$. There have been a number of excellent articles written [42, 43,44] which discuss the use of a generalized inverse in solving linear sets of equations, so only a few brief comments will be made here concerning its use.

Basically, the generalized inverse G^+ will provide a least-squares fit solution to (3.14) when the rank of G is not m. In other words, whether G is rectangular, ill-conditioned, or singular, $\underline{a} = G^+\underline{d}$ will have minimum norm among those $\underline{a}$ that minimize $||G\underline{a} - \underline{d}||^2$. Furthermore, $G^+ = G^{-1}$ when the rank is m. If the designer can tolerate a least-squares solution, this method might be preferable over the other because it would allow a greater time variation in the unknown parameters to be identified.

This suggests that the two methods may be combined to provide a solution for which there is a trade-off of accuracy in identifying the unknown parameters versus the allowable time variations of these parameters. Equations can be obtained over an identification interval whose maximum length is dictated by parameter variation considerations. If sufficient independent equations can be obtained, then $[G^N(k)]^{-1}$ can be found and the exact values of $\underline{a}$ determined. However, if G is still singular or ill-conditioned, then there are more data (equations) for which a least-squares solution can be obtained, hence a more accurate answer. Of course, the trade-off must be weighed by the designer for specific system specifications.

Numerical Examples

In this section two of a number of examples simulated to verify the analytical results are presented. One simulates the case for $n = m$. The other simulates the case $m > n$, so that additional equations must be obtained in order to solve the identification algorithm. In addition, for the latter example, the initial conditions and parameter values are chosen so that the initial G matrix is singular. This tests the algor-

ithm's ability to handle singular matrices.

Example I:

This example simulates the case where n = m = 2. Consider the following model equation

$$\begin{bmatrix} x_1(j+1) \\ x_2(j+1) \end{bmatrix} = \begin{bmatrix} a_{11} & 1 \\ 0 & a_{22} \end{bmatrix} \begin{bmatrix} x_1(j) \\ x_2(j) \end{bmatrix} + \begin{bmatrix} 1.0 \\ 0.5 \end{bmatrix} u(j), \tag{3.22}$$

$$\begin{bmatrix} x_{P_1}(0) \\ x_{P_2}(0) \end{bmatrix} = \begin{bmatrix} 2.0 \\ 1.0 \end{bmatrix},$$

where the constant parameters a_{11} and a_{22} are unknown. The true plant parameters are $a_{P_{11}} = 0.8$ and $a_{P_{22}} = 0.5$. A closed-loop control is desired for the above model which minimizes the following cost functional

$$J = \frac{1}{2} \sum_{j=0}^{j_f-1} \{\underline{x}^T(j) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \underline{x}(j) + u^2(j)\}. \tag{3.23}$$

Initial estimates of the unknown parameters are assumed to be $a_{11}^0 = 0.9$ and $a_{22}^0 = 0.3$. These estimates are used to calculate the Riccati gain, equation (3.5), which in turn is used to find u(0). Once the control is known, the identification algorithm is used to solve for a_{11}^1 and a_{22}^1. As expected, the identification algorithm yields

$$a_{11}^1 = a_{P_{11}} = 0.8$$

$$a_{22}^1 = a_{P_{22}} = 0.5$$

in just one iteration. The Riccati equation is then reiterated with the correct parameter values from which the optimum control is found for the duration of the system run time.

Example II:

For this example n = 2 and m = 3. The model equations are

$$\begin{bmatrix} x_1(j+1) \\ x_2(j+1) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & 0.5 \end{bmatrix} \begin{bmatrix} x_1(j) \\ x_2(j) \end{bmatrix} + \begin{bmatrix} 1.0 \\ 0.5 \end{bmatrix} u(j), \tag{3.24}$$

$$\begin{bmatrix} x_{P_1}(0) \\ x_{P_2}(0) \end{bmatrix} = \begin{bmatrix} 2.0 \\ 1.0 \end{bmatrix},$$

where now the constant parameters a_{11}, a_{12}, and a_{21} are unknown. Let $a_{P_{11}} = 0.8$, $a_{P_{12}} = 1.0$, and $a_{P_{21}} = 0.4$. From equation (3.19), it can be ascertained that the initial G matrix will be singular. Therefore, this example tests the algorithm's ability to handle singular (and ill-conditioned) matrices. Again, a closed-loop control is desired to minimize the cost functional (3.23). Initial unknown parameter estimates are assumed to be $a_{11}^0 = 0.9$, $a_{12}^0 = 1.2$, and $a_{21}^0 = 0.3$. These estimates are used to find the Riccati gain and to find not only u(0) but also u(1). This is necessary because two discrete time periods are required to obtain sufficient equations to solve for G^{-1}. As expected, a singular condition is encountered in attempting to solve the identification algorithm as initially structured. The algorithm then replaces the boundary condition $x_{P_1}(2)$ with $x_{P_2}(2)$ and yields

$$a_{11}^{1} = a_{P_{11}} = 0.8$$
$$a_{12}^{1} = a_{P_{12}} = 1.0$$
$$a_{21}^{1} = a_{P_{21}} = 0.4$$

in one iteration. The Riccati equation is then reiterated, the closed-loop control found, and the identification algorithm resolved to detect possible parameter changes in the duration of the system run. For these examples constant parameter values are assumed. Later examples test the algorithm's capability of tracking varying parameters.

Conclusions

It has been shown that, for the linear regulator system in which all of the plant states are available, the identification algorithm of Chapter II simplifies to a set of simultaneous equations. Except for singular or ill-conditioned cases, this set of equations converges to the correct answer in one iteration. Two methods of handling the singular or ill-conditioned problem are discussed with their associated advantages and disadvantages. Computer simulated examples have been worked to verify the analytical results.

CHAPTER IV

THE STATE REGULATOR SYSTEM WITH INCOMPLETE PLANT STATE KNOWLEDGE

Introduction

In the previous chapter the state regulator problem was examined for those systems for which all of the plant states are available as outputs. This chapter explores this problem when only some of the plant states are available. Conditions will be derived that ensure convergence of the identification algorithm to the proper solution. It will also be shown that, when convergence occurs, it occurs in a quadratic manner. As in Chapter III, computer simulated examples are included to verify the analytical results.

Mathematical Development

Let the linear system be represented again by equation (3.1). Now, however, since all of the plant states are not available as outputs, the output measurement vector must be written as

$$\underline{y}(j) = H\underline{x}(j). \tag{4.1}$$

The desire is still to derive an optimum closed-loop control which will minimize the quadratic cost functional of equation (3.3). A control u(j) of the form of equation (3.4) is the solution, except now estimates of the unavailable plant states must be used in $\underline{x}(j)$. This implies that the identification algorithm must identify not only the unknown parameters,

but also the unavailable plant states.

Unfortunately, for this case the identification algorithm usually does not simplify to a set of linear algebraic equations as it does when all the plant states are available. The reason is because a sufficient number of boundary conditions is not available to solve the identification algorithm by considering just one discrete time interval. Thus, the method of solution outlined in Chapter II must be followed.

Since all of the plant states are not available as outputs, the question of system observability arises. For a system to be observable, it must be possible to determine the state of an unforced system from the knowledge of the output of the system over some time interval. Since a knowledge of the states of the system is necessary if a closed-loop control is desired and also if identification is to be accomplished, the systems considered in this thesis must be observable. However, since the control interval for any realistic system is never infinite, the systems considered in this thesis do not have to be controllable.

With the aid of the functional flow diagram of Figure 3, a general description of the system operation will now be given. Initially, estimates of the unknown parameters and unavailable plant states are used by the controller to compute the closed-loop control. The system is controlled in this manner while data, consisting of the available plant outputs, are collected to determine boundary conditions for the identification algorithm. Once sufficient boundary conditions are accumulated, the identification algorithm is iterated until it converges to the correct solution of the multiple point boundary-value identification problem.

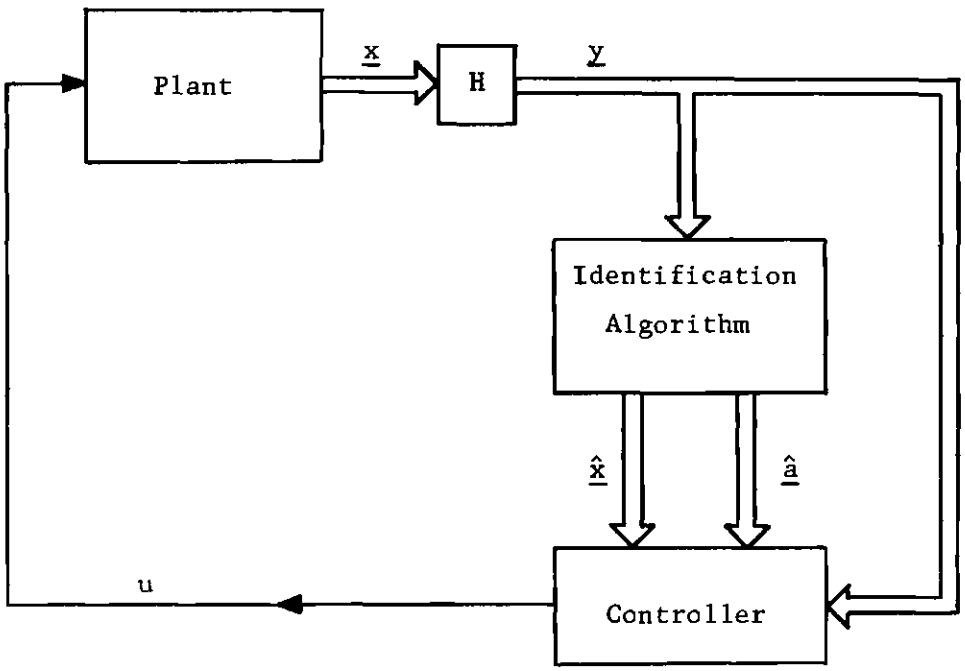

Figure 3. Functional Flow Diagram

The newly identified parameter and plant state values are then used by the controller to generate an updated optimum closed-loop control for the next identification period. This process is successively repeated in order to track time varying parameters. In this manner, the system control is always based on the latest identified parameter and plant state values.

A more detailed discussion will now be given concerning how the identification algorithm is implemented for the computer simulated examples used in this thesis. Central to the discussion will be the flow diagram of Figure 4. This diagram shows the implementation of the identification algorithm for the state regulator problem in which all of the plant states are not available as system outputs. The description which follows uses the nomenclature of the figure and the identification algorithm equations expressed in Chapter II.

Basically, the identification algorithm represents a transformation of the nonlinear multiple point boundary-value identification problem into a more readily solvable linear, nonstationary boundary-value problem. This results from expanding equation (2.5) in a Taylor Series expansion and ignoring the higher order terms. In fact, from equation (2.10), the solution for $\underline{z}^{N+1}(k)$ is just an initial condition equation which is easily solved once $\underline{z}^{N+1}(k_0)$ is known. The vector $\underline{z}^{N+1}(k_0)$ is obtained from solving the boundary condition equation (2.13). This equation is of the form

$$W^{N+1}\underline{z}^{N+1}(k_0) = \underline{b}, \tag{4.2}$$

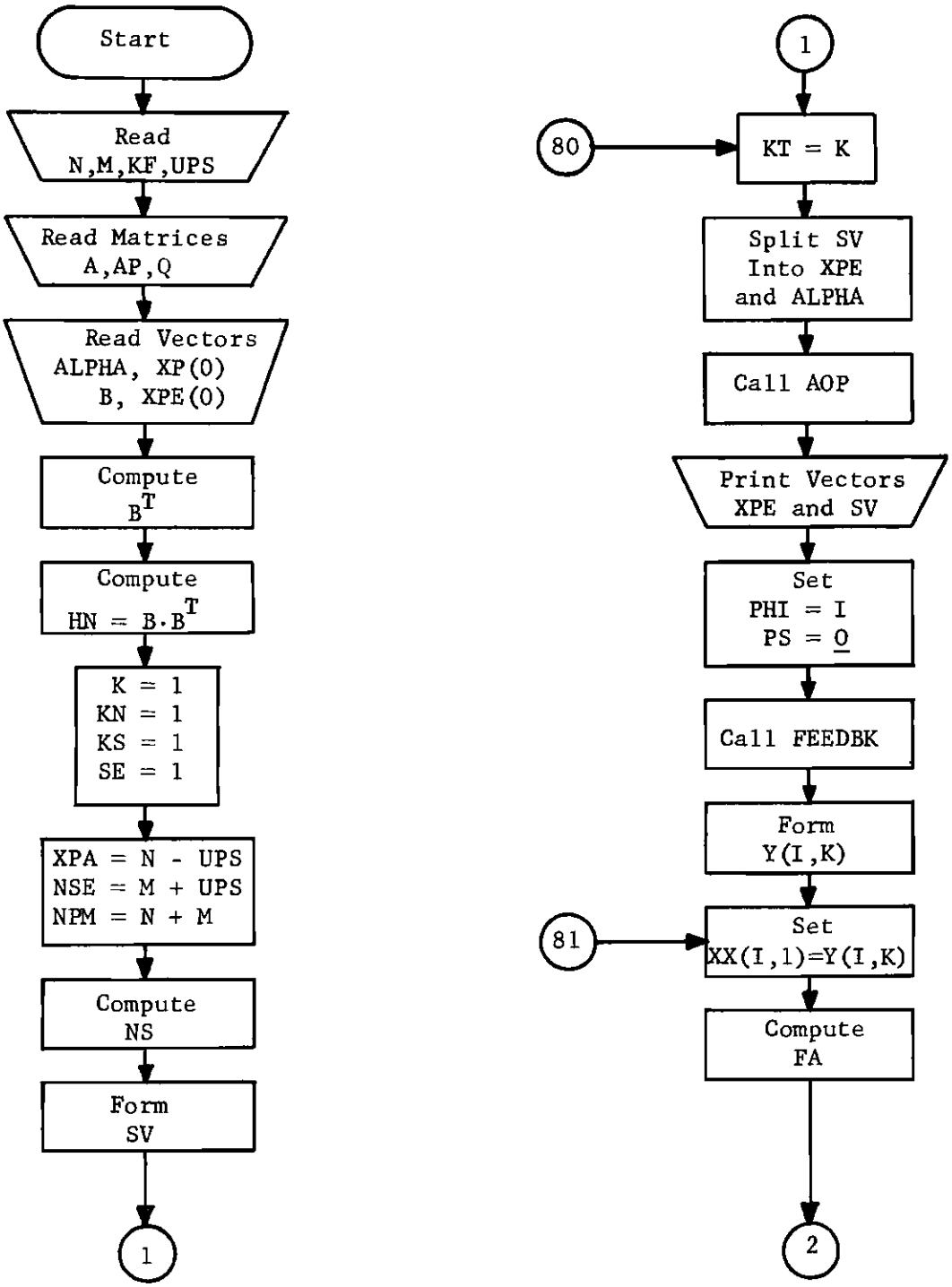

Figure 4. Flow Diagram of the Identification Algorithm's Implementation

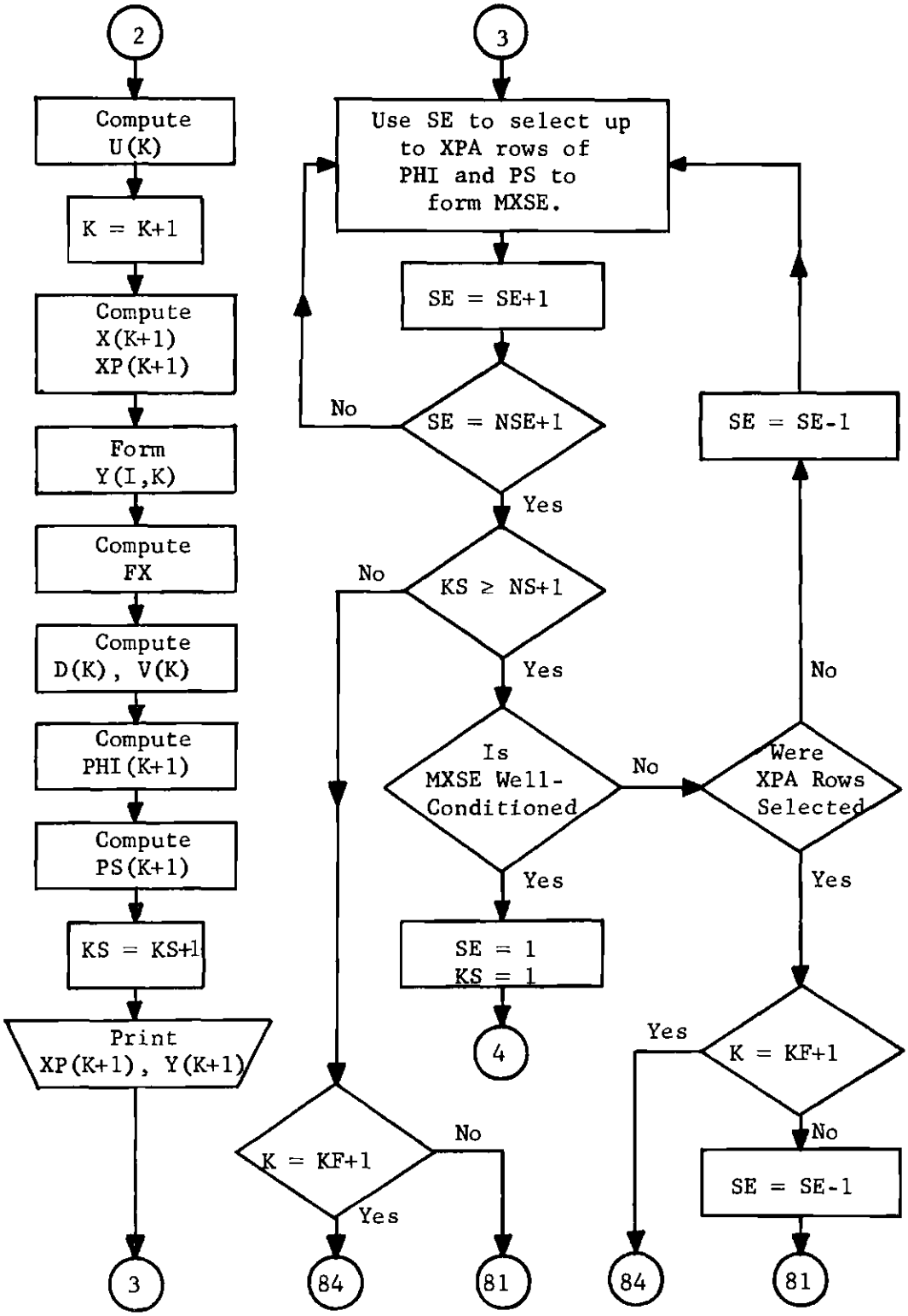

Figure 4. (Continued)

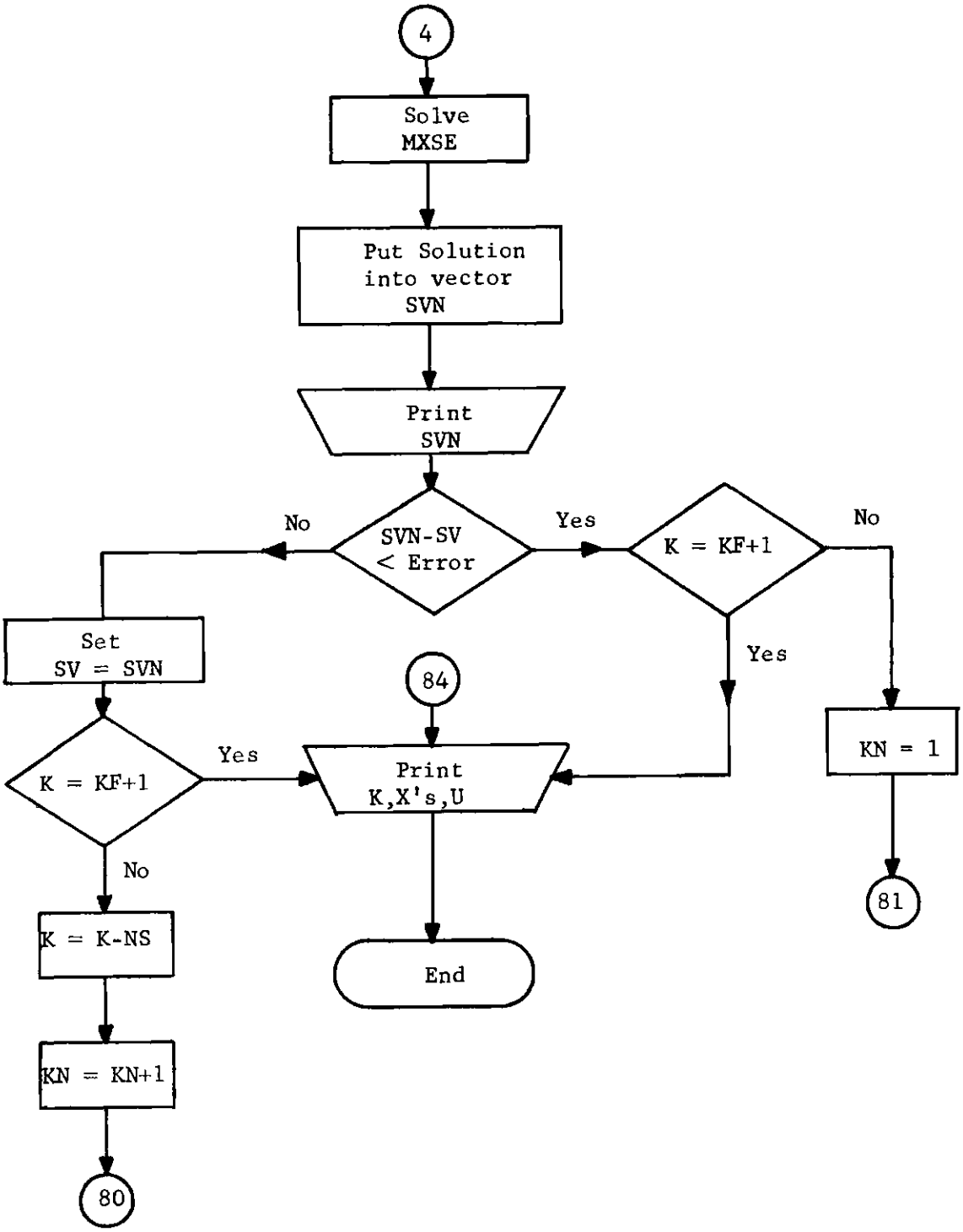

Figure 4. (Continued)

Flow Diagram Nomenclature

N	the n-dimension state vector
M	the m-dimension unknown parameter vector
KF	final system time
UPS	number of unavailable plant states
AP	true plant A matrix
ALPHA	the unknown parameter vector $\underline{a}$
XP(0)	known plant states at time zero
B	the input vector $\underline{b}$
XPE(0)	estimated plant states at time zero
KN	integer variable representing the iteration number for each identification cycle
KS	integer variable indicating each discrete period of the identification cycle
SE	integer variable indicating the number of simultaneous equations collected
XPA	integer constant representing the number of plant states available
NSE	integer constant representing the number of simultaneous equations needed
NPM	sum of n plus m
NS	integer constant representing the number of discrete time periods per identification cycle
SV	solution vector at the N^{th} iteration, composed of ALPHA and XPE
SVN	solution vector at the $(N+1)^{th}$ iteration
AOP	subroutine which updates the ALPHA estimates in A and computes A^T, A^{-1}, $[A^{-1}]^T$

Figure 4. (Continued)

PHI	homogeneous solution $\phi^N(k,k_0)$
PS	particular solution $\underline{p}^N(k)$
FEEDBK	subroutine to calculate the Riccati gain
Y(I,K)	stores X(I) over time
FA	equals $\partial \underline{f}/\partial \underline{a}_N$
FX	equals $\partial \underline{f}/\partial \underline{x}_N$
X(K)	model state variables
XP(K)	plant state variables
D(K),V(K)	same terms as defined in (2.9)
MXSE	matrix representing the simultaneous equation of (2.13)
Error	real number representing maximum difference in the solution vectors allowable to indicate convergence

Figure 4. (Concluded)

which, provided W^{N+1} is of proper rank, can be solved as discussed in Chapter III to give

$$\underline{z}^{N+1}(k_0) = [W^{N+1}]^{-1}\underline{b}. \tag{4.3}$$

The formulation and solution of equation (4.2) are thus the crux of the implementation of the identification algorithm. This equation contains NSE (representing the m-dimensional unknown parameter vector plus the ups-dimensional unavailable plant state vector) linearly independent simultaneous equations. Since only up to XPA (plant states available) boundary conditions are available per each discrete time period, the minimum length of one identification cycle is

$$NS = \left\lceil \frac{NSE}{XPA} \right\rceil \tag{4.4}$$

discrete time periods, where NS means the number of steps and $[\cdot]$ means the next highest integer. As pointed out in Chapter III, the matrix W^{N+1} might be singular or ill-conditioned, so additional equations must be obtained--possibly requiring additional discrete time periods to be added which lengthens the identification cycle.

To obtain the set of simultaneous equations only those values of $\phi^{N+1}(k,k_0)$ and $\underline{p}^{N+1}(k)$ necessary to compute $\underline{z}^{N+1}(k_0)$ from (2.13) are retained in memory until the evaluation is completed. Once $\underline{z}^{N+1}(k_0)$ is calculated, the $(N+1)^{th}$ trajectory, $\underline{z}^{N+1}(k)$, is generated from equation (2.9). This trajectory will satisfy the boundary conditions but will not in general satisfy (2.1), since a linearization assumption is used in deriving (2.9). However, $\underline{z}^{N+1}(k)$ is used for evaluating $D^N(k)$ and $\underline{v}^N(k)$

[hence $\phi^{N+1}(k,k_0)$ and $\underline{p}^{N+1}(k)$] for the next iteration.

By using this iterative procedure, the sequence $\underline{z}^N(k)$, under proper conditions, converges to the correct solution for this multiple point boundary value problem. Convergence is checked by examining the rate of change of the initial condition $\underline{z}^N(k_0)$. If the difference between $z_i^{N+1}(k_0)$ and $z_i^N(k_0)$ is less than ERROR (10^{-4} for the simulations in this thesis) for all i, then the process is considered to have converged. In the next section it will be shown that, if this procedure converges, then the convergence occurs in a quadratic manner. (Quadratic convergence is defined to exist when the error of any one iteration is proportional to the square of the error of the previous iteration.) It will also be shown that there is a range of initial estimates of the unknown parameter and state variable values for which convergence is ensured.

Convergence Analysis

In Chapter II it was proven that there exists a unique solution to the problem of solving nonlinear plant equations subject to split boundary conditions. In this section it will be shown that the identification algorithm, equation (2.9), under proper conditions, converges to this unique solution. It will also be shown that, when convergence occurs, it occurs in a quadratic manner. This analysis is presented in this chapter because, even though the state regulator equations are linear in $\underline{x}$, the identification problem is nonlinear. The nonlinearity arises because unknowns of the form $a_{ij}^N x_j^N$ appear in the model equations.

First, consider the two equations comprising the identification algorithm [equations (2.5) and (2.7)]

$$\underline{x}^{N+1}(k+1) = \underline{f}[\underline{x}^{N+1}(k), \underline{a}^{N+1}(k), u(k)], \tag{4.5}$$

and

$$\underline{a}^{N+1}(k+1) = \underline{a}^{N+1}(k). \tag{4.6}$$

As was done in Chapter II, let

$$\underline{z}^{N+1} = \begin{bmatrix} \underline{x}^{N+1} \\ \underline{a}^{N+1} \end{bmatrix}. \tag{4.7}$$

Then equations (4.5) and (4.6) can be combined into the following equation

$$\underline{z}^{N+1}(k+1) = \underline{g}[\underline{z}^{N+1}(k), u(k)]. \tag{4.8}$$

Expanding (4.8) in a Taylor Series about the N^{th} iteration, assuming the control u(k) is known, the following is obtained

$$\underline{z}^{N+1}(k+1) = \underline{g}[\underline{z}^{N}(k), u(k)] + J_{\underline{z}^{N}(k)} \cdot [\underline{z}^{N+1}(k) - \underline{z}^{N}(k)], \tag{4.9}$$

where $J_{\underline{z}^{N}(k)}$ is the Jacobian matrix defined by

$$J_{\underline{z}^{N}(k)} = \frac{\partial \underline{g}[\underline{z}^{N}(k)]}{\partial \underline{z}^{N}(k)}. \tag{4.10}$$

It is obvious that equation (4.9) represents another way of expressing the identification algorithm of equation (2.9). Consider the N^{th} iteration of (4.9),

$$\underline{z}^{N}(k+1) = \underline{g}[\underline{z}^{N-1}(k),u(k)] + J_{\underline{z}^{N-1}(k)} \cdot [\underline{z}^{N}(k) - \underline{z}^{N-1}(k)]. \qquad (4.11)$$

Subtracting (4.11) from (4.9),

$$\underline{z}^{N+1}(k+1) - \underline{z}^{N}(k+1) = \underline{g}[\underline{z}^{N}(k),u(k)] - \underline{g}[\underline{z}^{N-1}(k),u(k)] \qquad (4.12)$$

$$- J_{\underline{z}^{N-1}(k)} \cdot [\underline{z}^{N}(k) - \underline{z}^{N-1}(k)] + J_{\underline{z}^{N}(k)} \cdot [\underline{z}^{N+1}(k) - \underline{z}^{N}(k)].$$

Now, because of the restrictions i) - iii) listed in Chapter II that the plant must satisfy, Taylor's formula with remainder can be applied to part of (4.12) as*

$$\underline{g}[\underline{z}^{N}(k),u(k)] - \underline{g}[\underline{z}^{N-1}(k),u(k)] - J_{\underline{z}^{N-1}(k)} \cdot [\underline{z}^{N}(k) - \underline{z}^{N-1}(k)] \qquad (4.13)$$

$$= \frac{1}{2} \sum_{i=1}^{n+m} \underline{e}_i [\underline{z}^{N}(k) - \underline{z}^{N-1}(k)]^{T} \left[\frac{\partial^2 g_i}{\partial \underline{z}^2} \right]_{\underline{\delta}} [\underline{z}^{N}(k) - \underline{z}^{N-1}(k)],$$

where the $\underline{e}_i$ are the natural basis vector (for example, $\underline{e}_2^{T} = [0\ 1\ 0\ \ldots\ 0]$), and $\underline{\delta}$ lies between $\underline{z}^{N}(k)$ and $\underline{z}^{N-1}(k)$. Using (4.13) in (4.12), the following equation is obtained

$$\underline{z}^{N+1}(k+1) - \underline{z}^{N}(k+1) = J_{\underline{z}^{N}(k)} \cdot [\underline{z}^{N+1}(k) - \underline{z}^{N}(k)] \qquad (4.14)$$

$$+ \frac{1}{2} \sum_{i=1}^{n+m} \underline{e}_i [\underline{z}^{N}(k) - \underline{z}^{N-1}(k)]^{T} \left[\frac{\partial^2 g_i}{\partial \underline{z}^2} \right]_{\underline{\delta}} [\underline{z}^{N}(k) - \underline{z}^{N-1}(k)].$$

*This formula can be found in Athans [45].

One form of solution of (4.14) is

$$\underline{z}^{N+1}(k) - \underline{z}^{N}(k) = \sum_{r=0}^{k-1} \Big\{ J_{\underline{z}^{N}(r)} \cdot [\underline{z}^{N+1}(r) - \underline{z}^{N}(r)] \tag{4.15}$$

$$+ \frac{1}{2} \sum_{i=1}^{n+m} \underline{e}_i [\underline{z}^{N}(r) - \underline{z}^{N-1}(r)]^T \left[\frac{\partial^2 g_i}{\partial \underline{z}^2} \right]_{\underline{\delta}} [\underline{z}^{N}(r) - \underline{z}^{N-1}(r)] \Big\}.$$

The next step is to take the norm of both sides of equation (4.15). Define the norm in the following manner

$$\| \underline{z}^{N+1}(k) - \underline{z}^{N}(k) \| = \max_{k} [\underline{z}^{N+1}(k) - \underline{z}^{N}(k)]^T [\underline{z}^{N+1}(k) - \underline{z}^{N}(k)]. \tag{4.16}$$

Using this definition, the following notation is used

$$\| \underline{z}^{N+1}(k) - \underline{z}^{N}(k) \|_{R_i} = [\underline{z}^{N+1}(k) - \underline{z}^{N}(k)]^T R_i [\underline{z}^{N+1}(k) - \underline{z}^{N}(k)]. \tag{4.17}$$

The norm of both sides of (4.15) then becomes

$$\| \underline{z}^{N+1}(k) - \underline{z}^{N}(k) \| \leq k \Big\{ \| \underline{z}^{N+1}(k) - \underline{z}^{N}(k) \|_{J^T_{\underline{z}^{N}(k)} J_{\underline{z}^{N}(k)}} \tag{4.18}$$

$$+ \frac{1}{2} \left[\left(\| \underline{z}^{N}(k) - \underline{z}^{N-1}(k) \|_{R_1} \right)^2 + \left(\| \underline{z}^{N}(k) - \underline{z}^{N-1}(k) \|_{R_2} \right)^2 \right.$$

$$\left. + \cdots + \left(\| \underline{z}^{N}(k) - \underline{z}^{N-1}(k) \|_{R_{n+m}} \right)^2 \right] \Big\},$$

where

$$R_i = \left[\frac{\partial^2 g_i}{\partial \underline{z}^2}\right]_{\underline{\delta}}.$$

The matrix $\|\underline{z}^N(k) - \underline{z}^{N-1}(k)\|_{R_i}$ is just a scalar constant times $\|\underline{z}^N(k) - \underline{z}^{N-1}(k)\|$, or

$$\|\underline{z}^N(k) - \underline{z}^{N-1}(k)\|_{R_i} = \psi_i \|\underline{z}^N(k) - \underline{z}^{N-1}(k)\|. \tag{4.19}$$

Taking the largest ψ_i for the R_i above, and letting $\|\underline{z}^{N+1}(k) - \underline{z}^N(k)\|_{J^T_{\underline{z}^N(k)} J_{\underline{z}^N(k)}} = \psi \|\underline{z}^{N+1}(k) - \underline{z}^N(k)\|$, equation (4.18) becomes

$$\|\underline{z}^{N+1}(k) - \underline{z}^N(k)\| \le k \{\psi \|\underline{z}^{N+1}(k) - \underline{z}^N(k)\| \tag{4.20}$$

$$+ \frac{1}{2}(n+m)\psi_i^2 \|\underline{z}^N(k) - \underline{z}^{N-1}(k)\|^2\}.$$

Solving (4.20) for $\|\underline{z}^{N+1}(k) - \underline{z}^N(k)\|$,

$$\|\underline{z}^{N+1}(k) - \underline{z}^N(k)\| \le \frac{\frac{1}{2}(n+m)\, k\psi_i^2}{1+k\psi} \|\underline{z}^N(k) - \underline{z}^{N-1}(k)\|^2 \tag{4.21}$$

$$= K \|\underline{z}^N(k) - \underline{z}^{N-1}(k)\|^2.$$

Equation (4.21) shows that convergence is quadratic if there is convergence at all. This is because the error at the $(N+1)^{th}$ iteration is proportional to the square of the error of the N^{th} iteration. Rewrite equa-

tion (4.21) as

$$\|\underline{z}^{N+1}(k) - \underline{z}^{N}(k)\| \leq K \|\underline{z}^{N}(k) - \underline{z}^{N-1}(k)\|^2$$

$$\|\underline{z}^{N}(k) - \underline{z}^{N-1}(k)\| \leq K \|\underline{z}^{N-1}(k) - \underline{z}^{N-2}(k)\|^2$$

$$\vdots$$

$$\|\underline{z}^{2}(k) - \underline{z}^{1}(k)\| \leq K \|z^{1}(k) - z^{0}(k)\|^2.$$

By the use of simple substitutions,

$$\|\underline{z}^{N+1}(k) - \underline{z}^{N}(k)\| \leq K \|\underline{z}^{N}(k) - \underline{z}^{N-1}(k)\|^2$$

$$\leq K(K \|\underline{z}^{N-1}(k) - \underline{z}^{N-2}(k)\|^2)^2 \leq \ldots,$$

the following inequality can be obtained:

$$\|\underline{z}^{N+1}(k) - \underline{z}^{N}(k)\| \leq K[K^{2^N-2}\|\underline{z}^{1}(k) - \underline{z}^{0}(k)\|^{2^N}] \tag{4.22}$$

$$= [K \|\underline{z}^{1}(k) - \underline{z}^{0}(k)\|]^{2^N} \Big/ K.$$

If the quantity $[K \|z^{1}(k) - z^{0}(k)\|] < 1$, the right hand side of (4.22) will approach zero as N increases. Consequently, $\underline{z}^{N}(k)$ will approach a function $\underline{z}(k)$ and (4.9) is reduced to

$$\underline{z}(k+1) = \underline{g}[\underline{z}(k), u(k)], \tag{4.23}$$

the solution of which satisfies the original equation (2.14).

Observe from equation (4.21) that, for a given value of k (k here is representative of the number of discrete time intervals per identification cycle), an initial approximation $\underline{z}^0(k)$ can theoretically be chosen so that $[K \|\underline{z}^1(k) - \underline{z}^0(k)\|]$ is less than one and convergence is assured.

Numerical Examples

In order to verify the analytical results presented in this chapter, two computer simulated examples are presented. The first example illustrates the fact that the system does not have to be controllable in order to apply the identification algorithm. In the second, initial condition estimates are varied to show that there is a definite range of initial values for which the identification algorithm will converge to the correct solution.

Example III:

Consider the following model equations

$$\begin{bmatrix} x_1(j+1) \\ x_2^N(j+1) \end{bmatrix} = \begin{bmatrix} a_{11} & 1 \\ 0 & a_{22} \end{bmatrix} \begin{bmatrix} x_1(j) \\ x_2^N(j) \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u(j), \qquad (4.24)$$

$$y(j) = x_1(j), \qquad x_{P_1}(0) = 2.0,$$

where the parameters a_{11}, a_{22}, and the state $x_2(j)$ are unknowns. This set of equations represents an uncontrollable system. The desire is to identify these unknown values and use them to calculate a closed-loop control which minimizes the following cost functional

$$J = \frac{1}{2} \sum_{j=0}^{10} \{\underline{x}^T(j) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \underline{x}(j) + u^2(j)\}. \qquad (4.25)$$

From the discussion of this chapter, it is expected that identification can be accomplished.

Let the true values of the unknowns be $a_{P_{11}} = 0.8$, $a_{P_{22}} = 0.5$, and $x_{P_2}(0) = 1.0$. Using initial estimates of $a_{11}^0 = 0.9$, $a_{22}^0 = 0.4$, and $x_2^0(0) = 1.4$, Table 1 shows the iterative values of the unknown quantities. As expected, the unknown quantities converge to the correct values in just a few iterations.

Table 1. Iterative Values of the Unknown Quantities of Example III

Iteration	a_{11}^N	a_{22}^N	$x_2^N(0)$
Initial	0.90000	0.40000	1.40000
1	0.93939	0.98254	0.72121
2	0.79444	0.29952	1.01110
3	0.78902	0.444179	1.02194
4	0.79981	0.49816	1.00037
5	0.79999	0.49999	1.00000
Real Value	0.80000	0.50000	1.00000

Example IV:

In the convergence analysis section, it was shown that there is a definite range of initial estimates of the unknown quantities for which the identification algorithm will converge. To illustrate this fact,

consider the following observable and controllable model

$$\begin{bmatrix} x_1(j+1) \\ x_2^N(j+1) \end{bmatrix} = \begin{bmatrix} a_{11} & 1 \\ 0 & a_{22} \end{bmatrix} \begin{bmatrix} x_1(j) \\ x_2^N(j) \end{bmatrix} + \begin{bmatrix} 1.0 \\ 0.5 \end{bmatrix} u(j), \qquad (4.26)$$

$$y(j) = x_1(j), \qquad x_{P_1}(0) = 2.0.$$

The cost functional of equation (4.25) is again used. Table 2 shows the iterative values of the unknown quantities for two sets of initial estimates. In Table 2a, when initial estimates of $a_{11}^0 = 0.9$, $a_{22}^0 = 0.4$, and $x_2^0(0) = 1.4$ are used, it is seen that the unknowns converge quadratically to the correct values of $a_{P_{11}} = 0.8$, $a_{P_{22}} = 0.5$, and $x_{P_2}(0) = 1.0$. However, in Table 2b, where the initial estimates are $a_{11}^0 = 1.0$, $a_{22}^0 = 0.3$, and $x_2^0(0) = 1.4$ convergence does not occur. Apparently, in Table 2b, the increase in the initial estimates of a_{11} and a_{22} away from the true values is enough so that the range of initial estimates necessary to ensure convergence is exceeded. Unfortunately, for an on-line system, there is no rigorous mathematical procedure for generating initial estimates that ensure convergence. The best estimates can probably be obtained from an in-depth knowledge of the system and its parameters. However, if the system can be run on a trial basis, or if the system can be accurately simulated, then there exist numerous techniques of obtaining better initial estimates.

Table 2. Iterative Values of the Unknown Quantities of Example IV

Iteration	a_{11}^N	a_{22}^N	$x_2^N(0)$
Initial	0.90000	0.40000	1.40000
1	0.86810	0.47401	0.86379
2	0.74798	0.51485	1.10403
3	0.79360	0.50190	1.01279
4	0.79986	0.50003	1.00026
5	0.80000	0.49999	0.99999
Real Value	0.80000	0.50000	1.00000

(a)

Iteration	a_{11}^N	a_{22}^N	$x_2^N(0)$
Initial	1.00000	0.30000	1.40000
1	0.93136	0.43902	0.73726
2	1.17463	0.39279	0.25073
3	1.23192	0.28413	0.35215
4	1.31059	0.20376	0.51629
5	1.19242	0.29325	0.40357
Real Value	0.80000	0.50000	1.00000

(b)

Conclusions

In this chapter the implementation of the identification algorithm of Chapter II to linear regulator systems in which only some of the plant states are available has been examined. These systems must be observable, but need not be controllable. Analytically it has been shown that there is a range of initial unknown parameter and state variable estimates for which the identification algorithm converges. This range is a function, among other things, of the order of the system, the number of unknown parameters that must be identified, and the length of the identification period required, and is fixed once the system model equations are established. It was also shown that, when convergence occurs, it occurs in a quadratic manner. Computer simulated examples have been included which verified the analytical results.

CHAPTER V

APPLICATION OF THE IDENTIFICATION ALGORITHM TO A LINEAR PERTURBATION CONTROL SCHEME

Introduction

For many systems a closed-loop control of the form $u(j) = \gamma[\underline{x}(j),\underline{a}(j)]$ cannot be found. Yet, some form of feedback is usually desirable. One solution to this problem is the use of a perturbation control scheme. For this scheme an optimal (nominal) open-loop control is found which minimizes some desired performance criterion. A feedback control is then developed about the optimal trajectory by minimizing a second cost function which is quadratic in deviation from the nominal trajectory and control. This chapter examines the use of the perturbation control scheme in conjunction with the identification algorithm of Chapter II. Only the mathematical development will be given in this chapter, while a practical example utilizing this scheme is presented in Chapter VI.

Mathematical Development

There are many systems for which the designer knows the desired nominal trajectory he would like the system to follow. For example, the trajectory might result from physical constraints imposed by the problem, such as the critical re-entry path followed by spacecraft vehicles. Or, more commonly, the trajectory is a result of some optimal control problem

in which nominal parameter values have been assumed. To be more specific, if nominal parameter (and unknown state) values are assumed, equation (2.1) can be rewritten as

$$\underline{x}(j+1) = \underline{f}[\underline{x}(j),u(j)]. \tag{5.1}$$

The optimal control problem is then to find a nominal input, $u_0(j)$, such that the following cost functional is minimized

$$J_1 = \frac{1}{2} \sum_{j=0}^{j_f-1} L\{\underline{x}(j),u(j)\}. \tag{5.2}$$

In either case, there are systems in which the designer desired to minimize the perturbations about the nominal trajectory. One method of accomplishing this minimization is to use a linear perturbation control scheme.

There are several excellent articles written [28,45] which describe the linear perturbation control scheme. Basically, this scheme is just a feedback control, developed about the nominal trajectory by minimizing a second cost functional, J_2, which is quadratic in deviation from the nominal trajectory and control. The nominal trajectory and control are related by

$$x_0(j+1) = \underline{f}[\underline{x}_0(j),u_0(j)]. \tag{5.3}$$

By considering equations (5.1) and (5.3), the deviation of the state and control variables about the nominal trajectories are expressed as

$$\Delta\underline{x}(j) = \underline{x}(j) - \underline{x}_0(j) \tag{5.4}$$

and

$$\Delta u(j) = u(j) - u_0(j). \tag{5.5}$$

The linearized perturbation equation can thus be represented as

$$\Delta\underline{x}(j+1) = A_0(j)\Delta\underline{x}(j) + \underline{b}_0(j)\Delta u(j), \tag{5.6}$$

where

$$A_0(j) = \left.\frac{\partial \underline{f}}{\partial \underline{x}}\right|_{\substack{\underline{x}_0(j) \\ u_0(j)}} \tag{5.7}$$

is an $n \times n$ time-varying matrix which is obtained by evaluating the elements of the Jacobian matrix $\partial\underline{f}/\partial\underline{x}$ along the known (precomputed) time functions $\underline{x}_0(j)$ and $u_0(j)$, and

$$\underline{b}_0(j) = \left.\frac{\partial \underline{f}}{\partial u}\right|_{\substack{\underline{x}_0(j) \\ u_0(j)}} \tag{5.8}$$

is an n-dimensional time-varying vector which is obtained by evaluating $\partial\underline{f}/\partial u$ along the same nominal trajectory and control. The perturbation control problem is to find a $\Delta u(j)$, using the constraint equation (5.6), such that the quadratic cost functional

$$J = \frac{1}{2}\sum_{j=0}^{j_f-1} [\Delta\underline{x}^T(j)Q_0\Delta\underline{x}(j) + \Delta u^2(j)] \tag{5.9}$$

is minimized. This problem is similar to the problem considered in Chapter III, so the solution has the same form as equation (3.4), and is

$$\Delta u(j) = - \underline{b}_0^T(j)[A_0^{-1}(j)]^T[P_0(j) - Q_0]\,\Delta\underline{x}(j), \tag{5.10}$$

where

$$P_0(j) = Q_0 + A_0^T(j)P_0(j+1)A_0(j) - \frac{A_0^T(j)P_0(j+1)\underline{b}_0\underline{b}_0^TP_0(j+1)A_0(j)}{1 + \underline{b}_0^TP_0(j+1)\underline{b}_0}, \tag{5.11}$$

with $P_0(j_f) = [0]$.

The design of the closed-loop controller is complete when $A_0(j)$ and $\underline{b}_0(j)$ are evaluated about the nominal trajectories, Q_0 is selected, and the associated Riccati equation, equation (5.11), is solved. The nominal trajectory, control, and time-varying gains are stored and used, on-line, to complete the perturbation controller design.

The main practical disadvantage of the above scheme is that, for many systems, all of the plant states are not available as system outputs, so $\Delta\underline{x}(j)$ of equation (5.4) cannot be calculated. Thus, some type of state estimation scheme is required for these systems in order for the perturbation feedback control to be implemented.

If the system contains unknown parameters that must be identified, then additional difficulties will be encountered in applying most perturbation control schemes. The trouble occurs because the time-varying matrix, $A_0(j)$, and the time-varying vector, $\underline{b}_0(j)$, will, in most instances, contain the unknown parameters as elements. Most perturbation control schemes circumvent this problem by evaluating $A_0(j)$ and $\underline{b}_0(j)$ using only

the nominally assumed parameter values. This scheme is clearly suboptimal if the actual parameter values differ from the nominally assumed ones. The optimum linear perturbation control scheme would track the varying parameter values, and use the latest estimates in evaluating, on-line, $A_0(j)$ and $\underline{b}_0(j)$. The updated values of $A_0(j)$ and $\underline{b}_0(j)$ would then be used to recompute the feedback control.

Thus, the parameter and state identification algorithm developed in Chapter II may be used in conjunction with the perturbation control. Figure 5 shows a block diagram of the composite system. The parameter and state identification portion of the system functions as described in Chapter IV. The updated parameter values, $\hat{\underline{a}}(j)$, are fed to the controller where they are used to reevaluate $A_0(j)$ and $\underline{b}_0(j)$ about the nominal trajectories. Then $A_0(j)$ and $\underline{b}_0(j)$ are used in recomputing the Riccati equation, and hence, the time-varying feedback gain

$$\Gamma(j) = - \underline{b}_0(j)[A_0^{-1}(j)]^T[P_0(j) - Q_0] \tag{5.12}$$

that will be used over the next identification interval. Thus, the feedback gain used over each identification cycle is computed from the latest identified parameter values.

Meanwhile, the state perturbation vector, $\Delta\underline{x}(j)$, is calculated by subtracting the stored nominal state vector, $\underline{x}_0(j)$, from the estimated state vector, $\hat{\underline{x}}(j)$. The perturbation control is then merely

$$\Delta u(j) = \Gamma(j)\Delta\underline{x}(j). \tag{5.13}$$

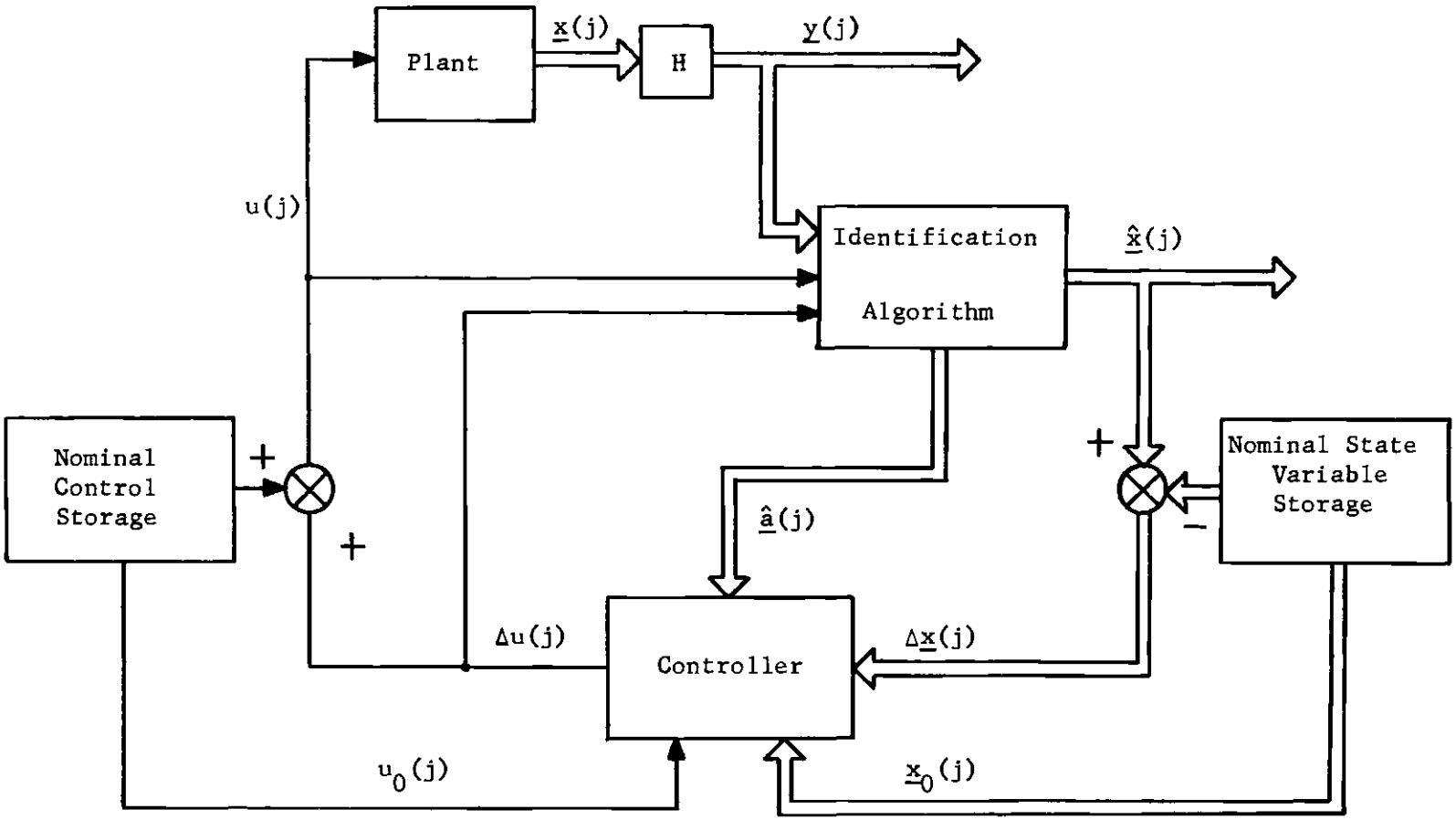

Figure 5. Combined Identifier-Linear Perturbation Control System Block Diagram

This control, added to the stored nominal control, $u_0(j)$, provides the updated input control, $u(j)$, to the plant. For each identification cycle the above process is repeated.

The linear perturbation control scheme described above is still not optimal. This occurs because the exact parameter and state values are only available at the end of each identification interval. In the interim, the mathematical model portion of the identification algorithm is used to provide state estimates based on the latest identified values. Although the proposed perturbation control scheme is suboptimal, it is obvious that it provides a better feedback controller than if just the nominally assumed parameter values are used.

The disadvantage of the proposed scheme is, of course, that the time-varying feedback gain must be recomputed for each identification cycle. Thus, the designer is faced with a trade-off between accuracy in tracking the nominal trajectory versus the amount of computing time available to obtain the desired accuracy.

Conclusions

In this chapter a linear perturbation control scheme has been developed for systems in which all of the plant states are not available as system outputs. A distinct feature of this scheme is that all unknown parameter values are continuously identified, with the latest values being used in the calculation of the optimal feedback control. A practical example utilizing this scheme is developed in the next chapter.

CHAPTER VI

PERTURBATION CONTROL OF THE STARTUP OF A THERMAL NUCLEAR REACTOR

Introduction

In this chapter the linear perturbation control scheme developed in the last chapter will be applied to the control of the startup of a thermal nuclear reactor which has no temperature feedback. The system is modeled by a set of kinetic equations in which disturbances due to coolant flow through the core is represented as a random parameter variation. The identification algorithm is used to track the value of these variations in order to effect an optimum linear perturbation control of the deviations about the nominal trajectories.

Description of a Thermal Nuclear Reactor*

A nuclear reactor depends for its operation on a continued chain-reaction involving fission of heavy nuclei and release of neutrons. Fission takes place as a result of neutron capture by the heavy nuclei. One product of the fission process is additional neutrons which take part in further capture--fission reactions, thus producing a chain-reaction. If conditions are such that each fission process leads, on the average, to one additional fission, the reactor is said to be critical. Super-

*This description was condensed from Schultz [46] and Masters and Sage [49].

critical and subcritical are defined accordingly. There are a number of factors which affect the reactivity (criticality) of an operating pile. Among these are the nature of the fuel, the geometry of the reactor, the presence of neutron-absorbing substances known as poisons, the presence of certain neutron-producing materials known as enriching materials, environmental conditions such as temperature and pressure, the presence of a moderating substance (added to slow the neutrons to a desirable energy), and the position of control rods or devices used for regulation and control of the reactor. Two distinct types of neutrons take part in a chain reaction. The vast majority of the neutrons produced (about 99 percent) are "prompt" neutrons produced at the instant of fission. These particles give rise to dynamic effects characterized by relatively short time constants. The other type of neutron involved in a chain reaction is the "delayed" neutron which is released from the fission products at some time after the fission process and gives rise to effects exhibiting relatively long time constants. Despite the small proportion of delayed neutrons produced, their effects are extremely important to the dynamic aspects of reactor control. This importance is due to the fact that the delay time of the particles which produce the delayed neutrons is relatively long, thus allowing these particles to accumulate. Furthermore, since the reactor is normally operated near critical, small changes in neutron flux density are important.

Problem Formulation

For the purposes of the following analysis, the principle measure of reactor criticality will be the reactivity, ρ. The reactivity is a

measure of the excess neutrons generated by the chain reaction and is a function of (among other factors) the position of the control rods. For this reason, ρ will be the input variable. Between the time that a fission reaction takes place and the time that a delayed neutron is produced, the delayed neutron can be considered to reside in an intermediate particle known as a precursor. The fission process produces, in addition to prompt neutrons, a number of precursors which, in turn, decay at a later time, to emit a delayed neutron. The delayed neutrons are found to be divided into a number of groups, each with its own decay constant and relative concentration. If the composite reactor is considered as an entity, it is possible to formulate a deterministic point-model for its dynamic characteristics. These kinetic equations can be written as follows:*

$$\frac{d\eta(t)}{dt} = \left[\frac{\rho(t) - \beta}{\Lambda}\right] \eta(t) + \sum_i \lambda_i \, c_i(t) \tag{6.1}$$

$$\frac{dc_i(t)}{dt} = \frac{\beta_i}{\Lambda} \eta(t) - \lambda_i \, c_i(t), \tag{6.2}$$

where

$\eta(t)$ = Neutron flux density ($1/cm^3$);

$c_i(t)$ = Precursor density ($1/cm^3$);

$\rho(t)$ = Reactivity (input variable);

β = Average fraction of precursors formed;

*A rigorous derivation of these equations can be found in Ash [47]. These equations are of academic interest only, but are used to illustrate the system features.

Λ = Effective neutron lifetime (secs);

λ_i = Decay constant for precursor $c_i(t)$.

The fraction of precursors formed is subtracted from the reactivity and the rate of production of delayed neutrons by precursor decay is added to the rate of change of neutron flux density. The above equations comprise a set of coupled, first order equations. Due to the fact that $\rho(t)$ enters as the product $\rho(t)\eta(t)$, these equations are nonlinear if any attempt at closed loop control is made. Typically, one studies the macroscopic viewpoint of a nuclear reactor so that the precursors are considered to be lumped into a single "average" group with density $c(t)$, decay constant λ, and relative fraction β. Also, Thie [48] has observed that thermal nuclear reactors have random disturbances in the reactor flux due to coolant flow through the core, and that these disturbances can have a peak value of five to ten percent of the average flux. Thus, the model for the reactor becomes

$$\frac{d\eta(t)}{dt} = \left[\frac{\rho(t) + \xi(t) - \beta}{\Lambda}\right] \eta(t) + \lambda c(t) \tag{6.3}$$

$$\frac{dc(t)}{dt} = \frac{\beta}{\Lambda} \eta(t) - \lambda c(t), \tag{6.4}$$

where $\xi(t)$ represents the random disturbances in the reactor flux. Using a first difference approximation for the derivatives, the digital model of the reactor dynamics becomes:

$$\eta(j+1) = \left[1 + \frac{T}{\Lambda} \{\rho(j) + \xi(j) - \beta\}\right] \eta(j) + T\lambda c(j) \tag{6.5}$$

$$c(j+1) = \frac{T\beta}{\Lambda}\,\eta(j) + (1-\lambda T)c(j), \tag{6.6}$$

where $\eta(j) = \eta(jT)$ and T is the sampling period.

The nominal model equations are obtained by considering a system with no random disturbance, and using typical parameter values for β, λ, and Λ. For U_{235} the typical parameter values used for β, λ, and Λ are

$$\beta = 0.0064$$

$$\lambda = 0.1 \text{ sec}$$

$$\Lambda = 0.001 \text{ sec} .$$

Considering these values, and a sampling period of 0.025 sec, equations (6.5) and (6.6) become

$$\eta(j+1) = [0.84 + 25.0\rho(j)]\eta(j) + 0.0025c(j) \tag{6.7}$$

$$c(j+1) = 0.16\eta(j) + 0.9975c(j). \tag{6.8}$$

For the startup problem, the desire is to find the control, $\rho(j)$, which will increase the output power (the neutron flux density) from an initial state to a terminal state in some optimum manner. The manner chosen is to minimize the energy required to drive the control rods. It can also be shown [28, p. 514] that this minimization tends to minimize the reactor period, which is a very desirable physical objective. To define the problem more specifically, for the equations (6.7) and (6.8),

let the initial states be $\eta_0(0) = 0.5$ kw and $c_0(0) = 32.0$ kw and find the reactivity, ρ_0, such that at $t_f = 1$ sec, $\eta_0(t_f) = 5.0$ kw, and the performance index

$$J_1 = 0.0125 \sum_{j=0}^{39} \rho_0^2(j) \tag{6.9}$$

is minimized.

As explained in Chapter V, this problem is a nonlinear two-point boundary value problem which can be solved by several off-line numerical iterative techniques. For this particular problem, quasilinearization is used to obtain the solution. The nominal control and trajectory results are shown in Figure 6. These curves represent the desired ideal response of the system.

Open-Loop Response

An investigation is made into the open-loop response of the system to the nominal control, ρ_0, when the random disturbance in the reactor flux due to coolant flow through the core is included. Thie [48] has determined that this disturbance is correlated, and has a typical auto-correlation function as shown in Figure 7. For this thesis, a simulation of this autocorrelation function is produced by passing gaussian white noise through a one-pole filter with a cutoff frequency chosen to produce an autocorrelation function of $R(\tau) = e^{-1.5\tau}$. How well this simulates the actual autocorrelation function is also shown in Figure 7.

Using this disturbance in equation (6.5), with a variance of 0.001,

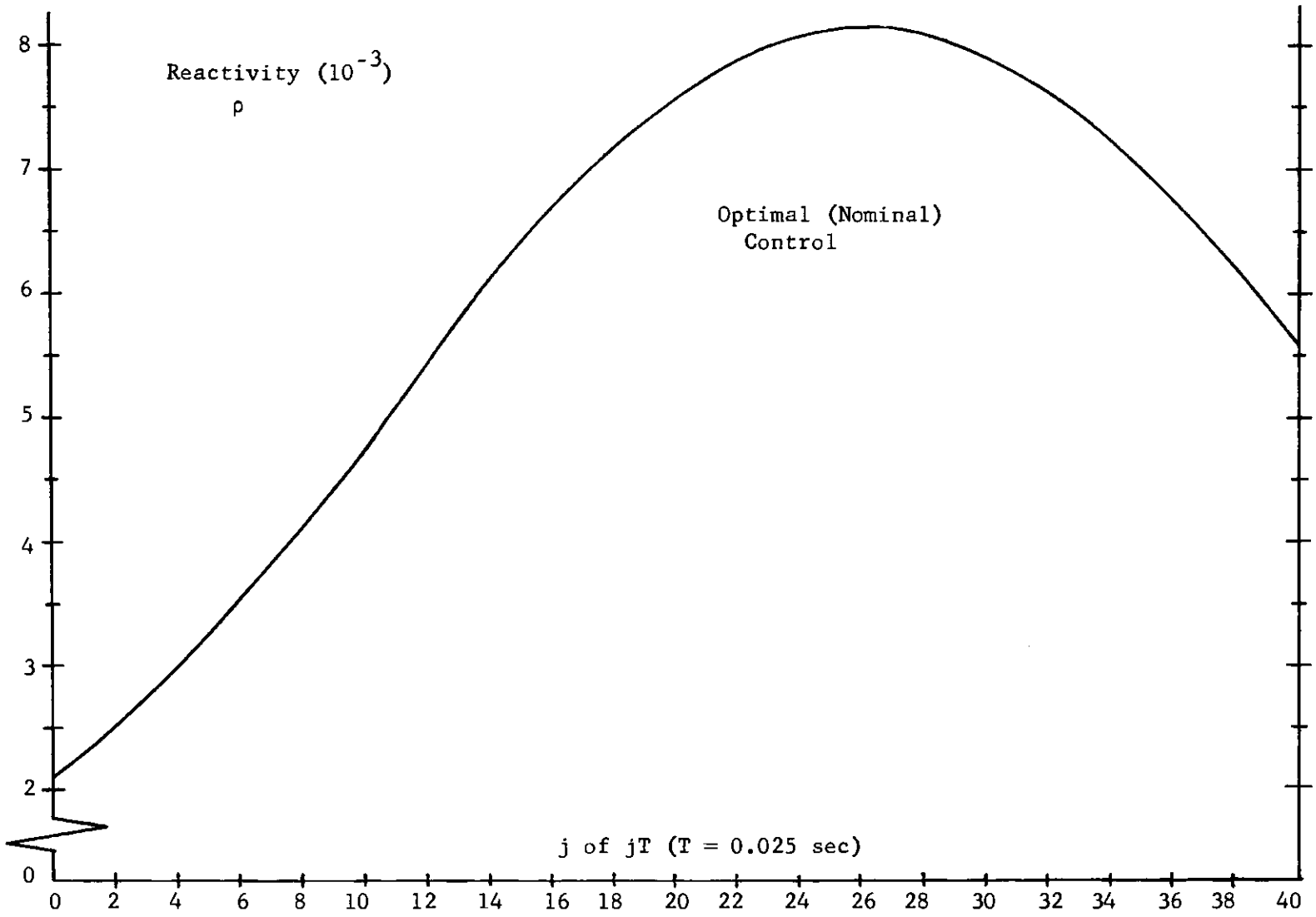

Figure 6 (a,b,c). Nominal Control and Trajectories

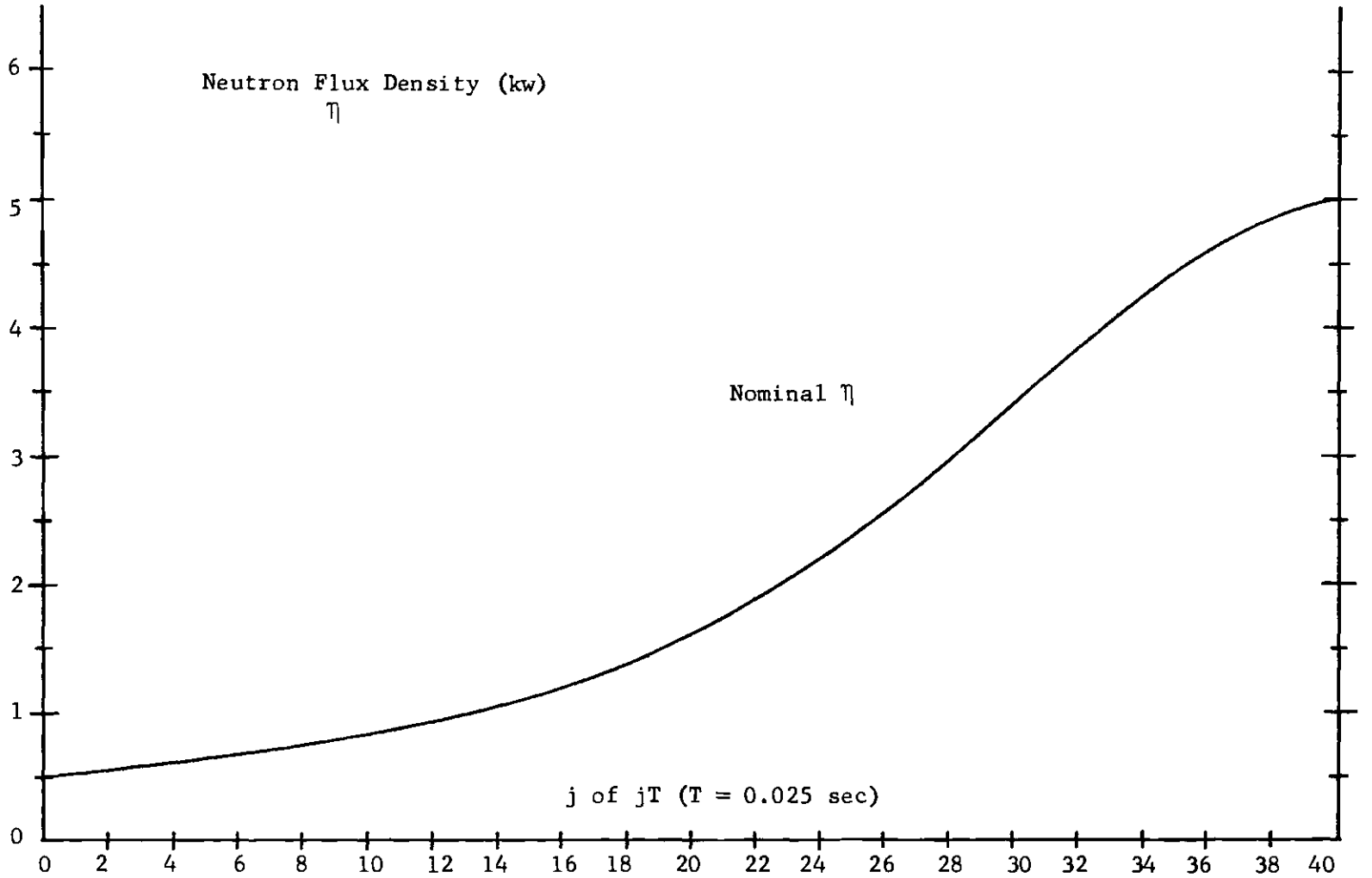
Neutron Flux Density (kw)
η
Nominal η
j of jT (T = 0.025 sec)
0
1
2
3
4
5
6
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40

Figure 6b.

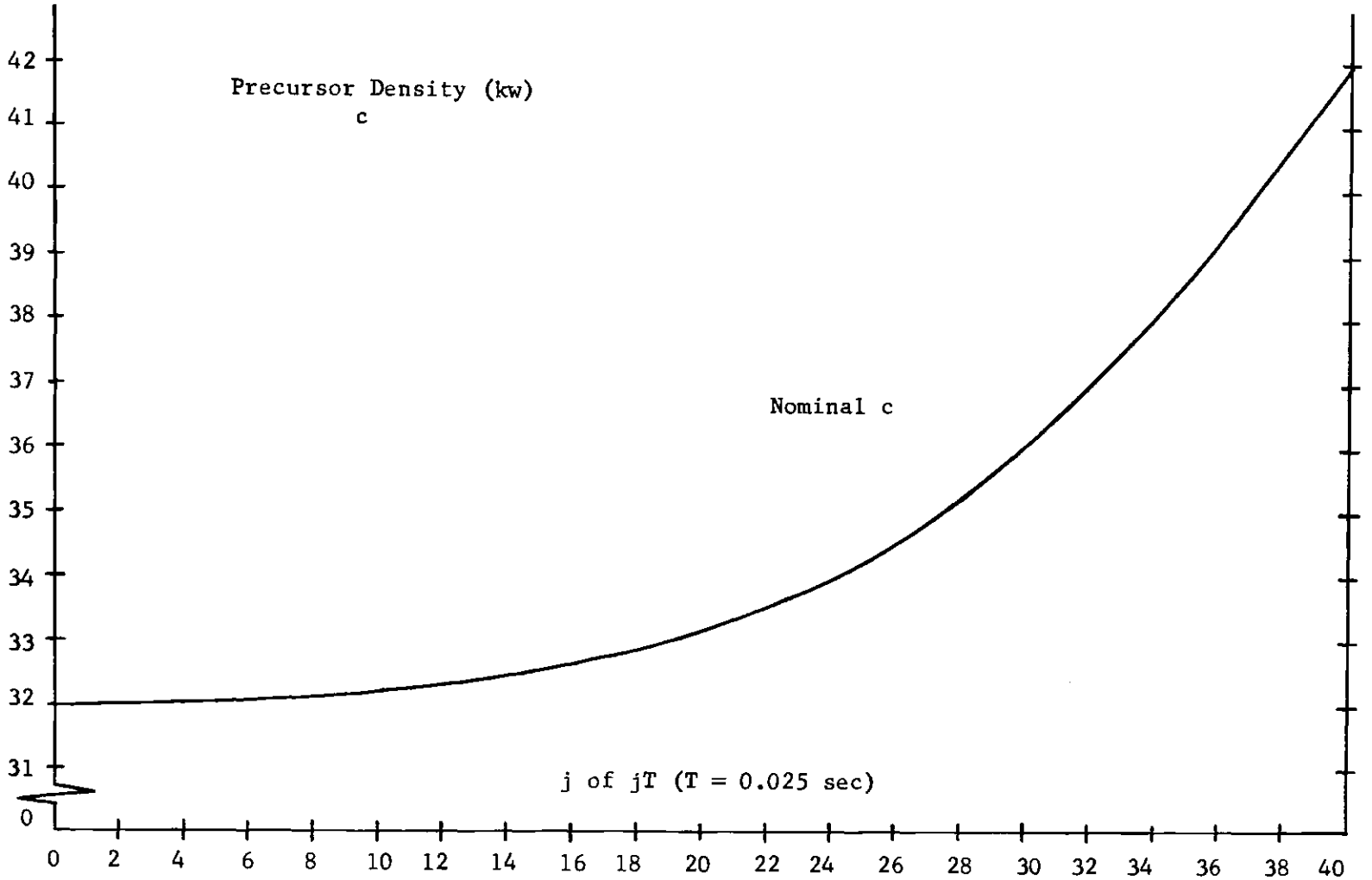
Precursor Density (kw)
c
Nominal c
j of jT (T = 0.025 sec)
42
41
40
39
38
37
36
35
34
33
32
31
0
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40

Figure 6c.

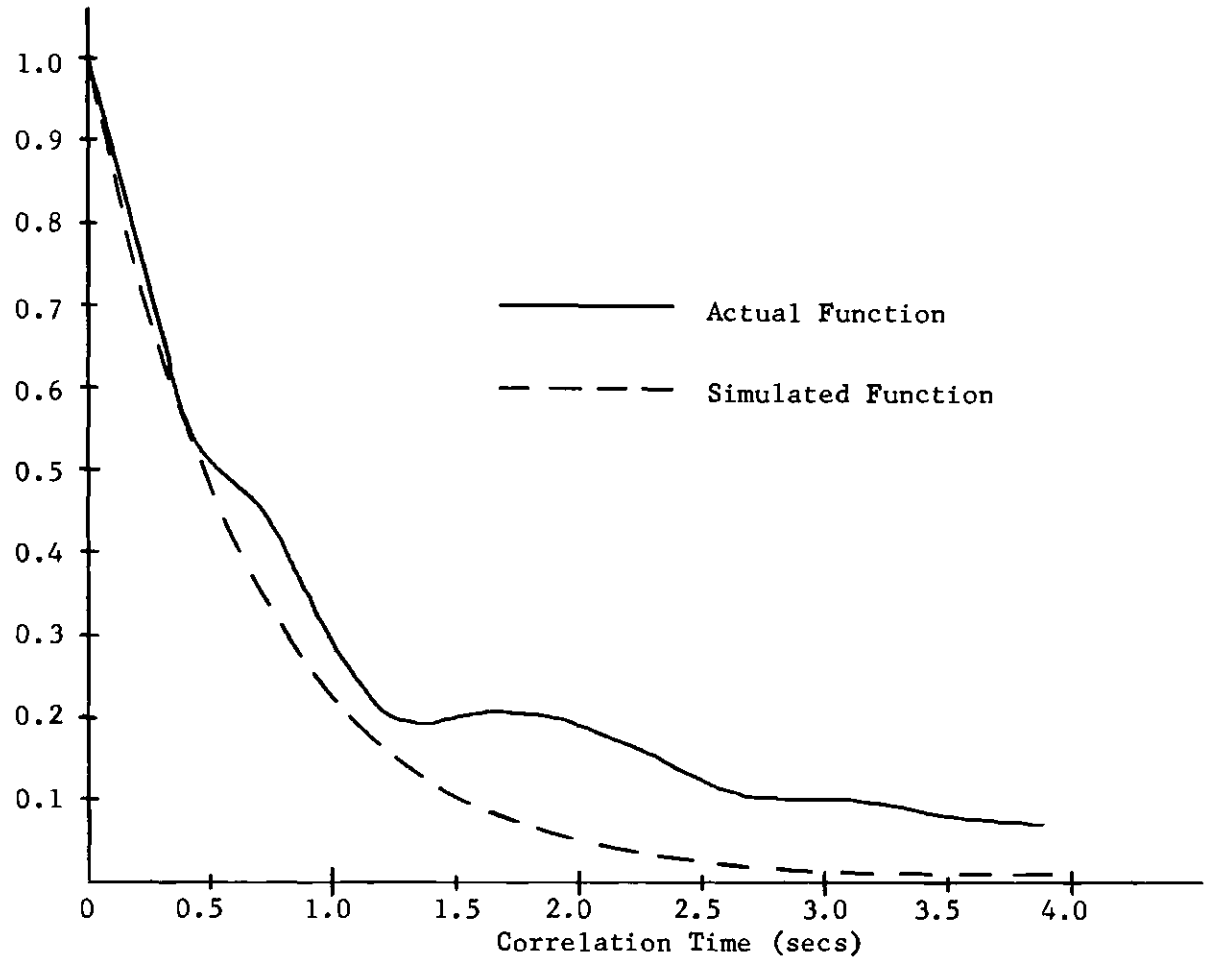

Figure 7. Random Disturbance Autocorrelation Function

the open-loop response of the system to the nominal control is again obtained. The results for a single time function are shown in Figure 8. Notice that the disturbance causes the trajectories to deviate as much as 12 percent from their nominals. To obtain a better picture of the extent with which the disturbance affects the open-loop response, a Monte Carlo simulation representing 100 time runs was performed. Figure 9 shows a plot of the ensemble mean and standard deviation of these runs. By inspection of the ensemble standard deviation, it is obvious that the random disturbance greatly affects the open-loop trajectories. In fact, for this application, the effect of the random disturbance is so great that the open-loop system is unacceptable.

Perturbation Control Response

From the curves shown in Figure 9, it is obvious that some form of feedback is required. In this section, the perturbation control scheme developed in Chapter V is used to provide this feedback. For the sake of clarity, the general perturbation control block diagram of Figure 5 is specialized for the nuclear reactor problem in Figure 10. Notice that the input to the nuclear system is the sum of the nominal control, the perturbation control, and the reactivity disturbance. However, as can be seen from equation (6.5), this disturbance can be modelled as a randomly varying parameter. Since the precursor concentration cannot be measured, $H = [1\ 0]$. Thus the identification algorithm must estimate the parameter $\xi(j)$ and the plant state $c(j)$.

The scheme is implemented as described in Chapter V. The perturbation controlled trajectories for a single time function are also shown

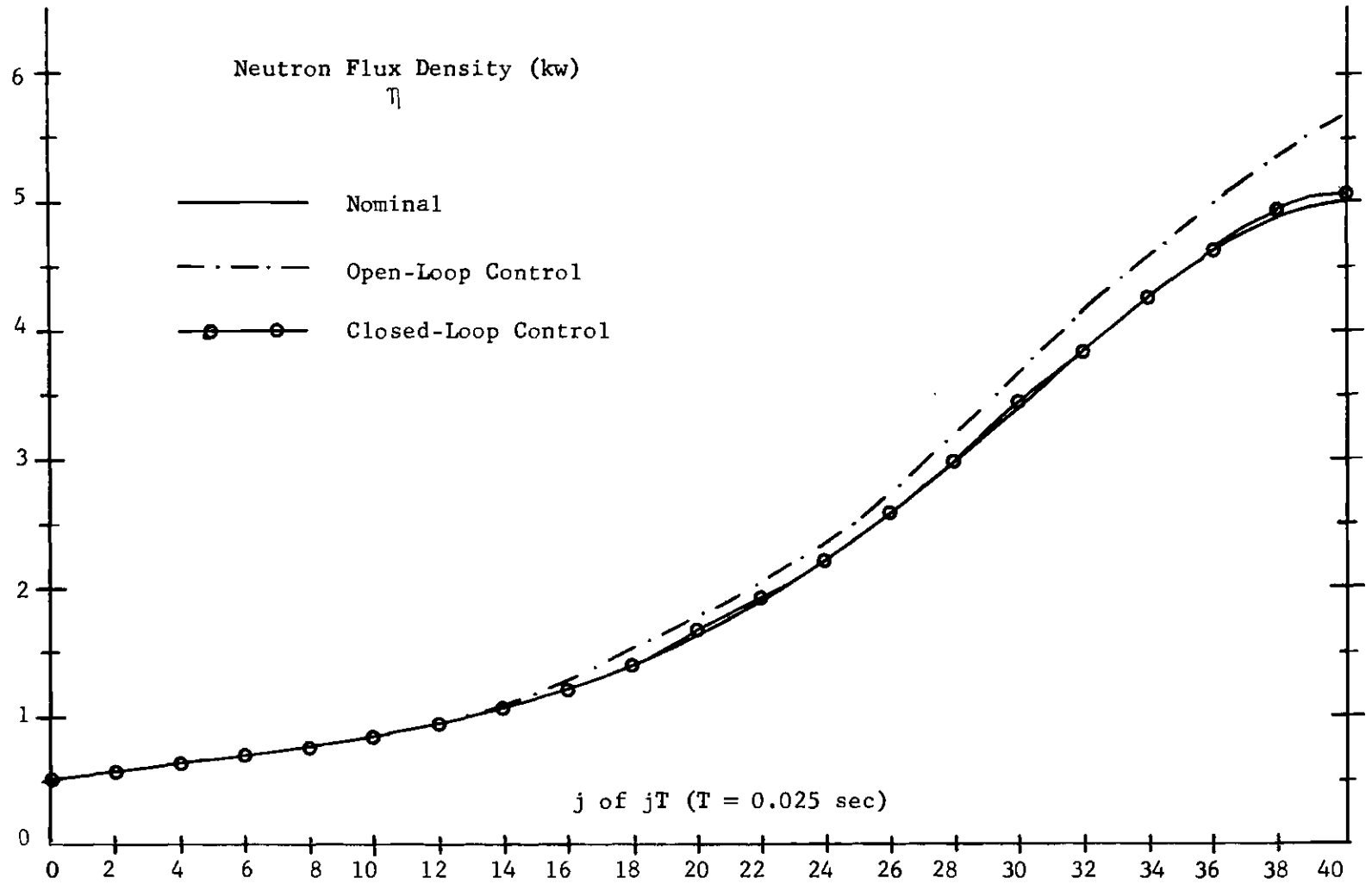

Figure 8(a,b). Comparison of the Open-Loop and Closed-Loop System Trajectories About the Nominal

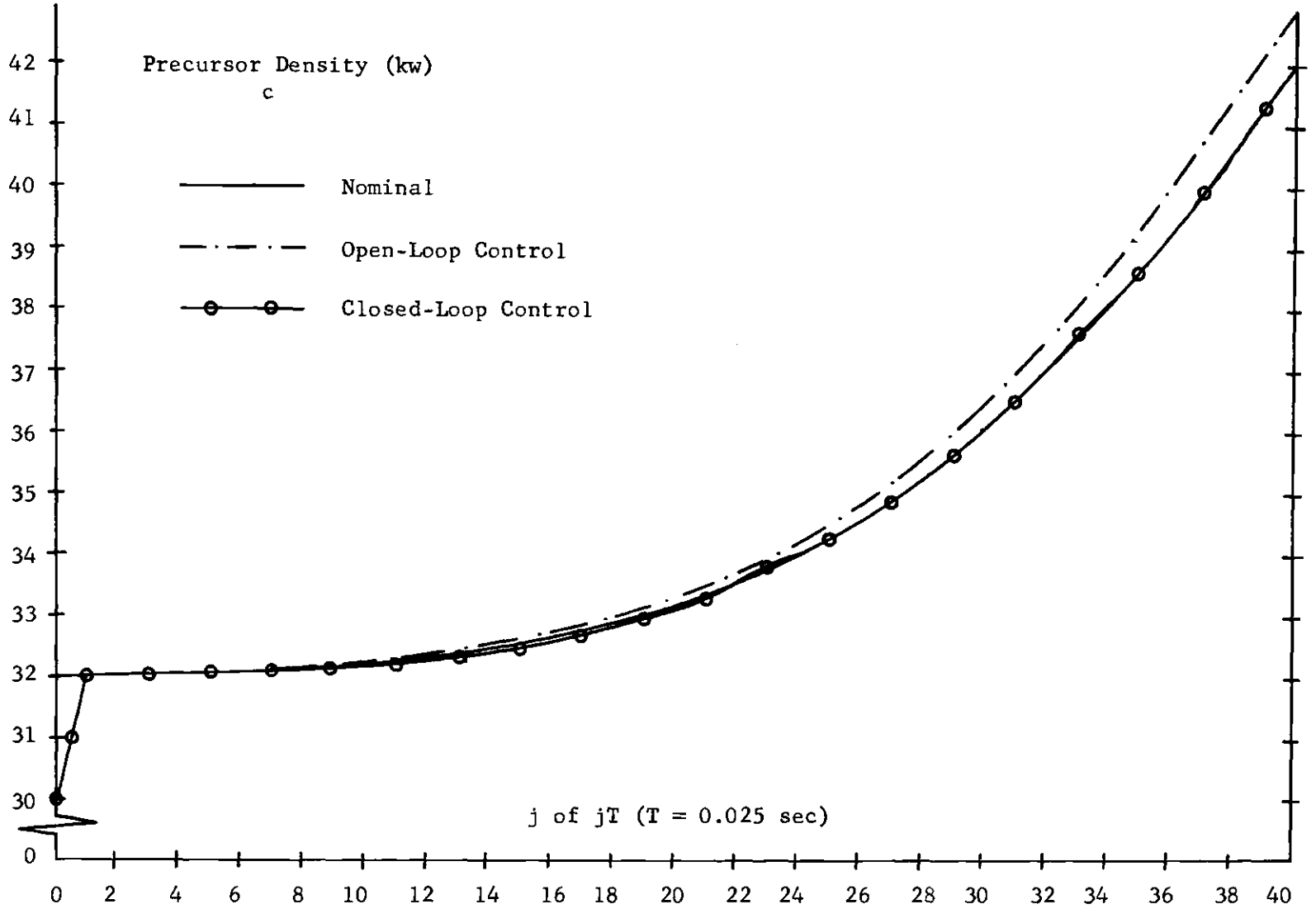
Precursor Density (kw)
c
Nominal
Open-Loop Control
Closed-Loop Control
j of jT (T = 0.025 sec)
42
41
40
39
38
37
36
35
34
33
32
31
30
0
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40

Figure 8b.

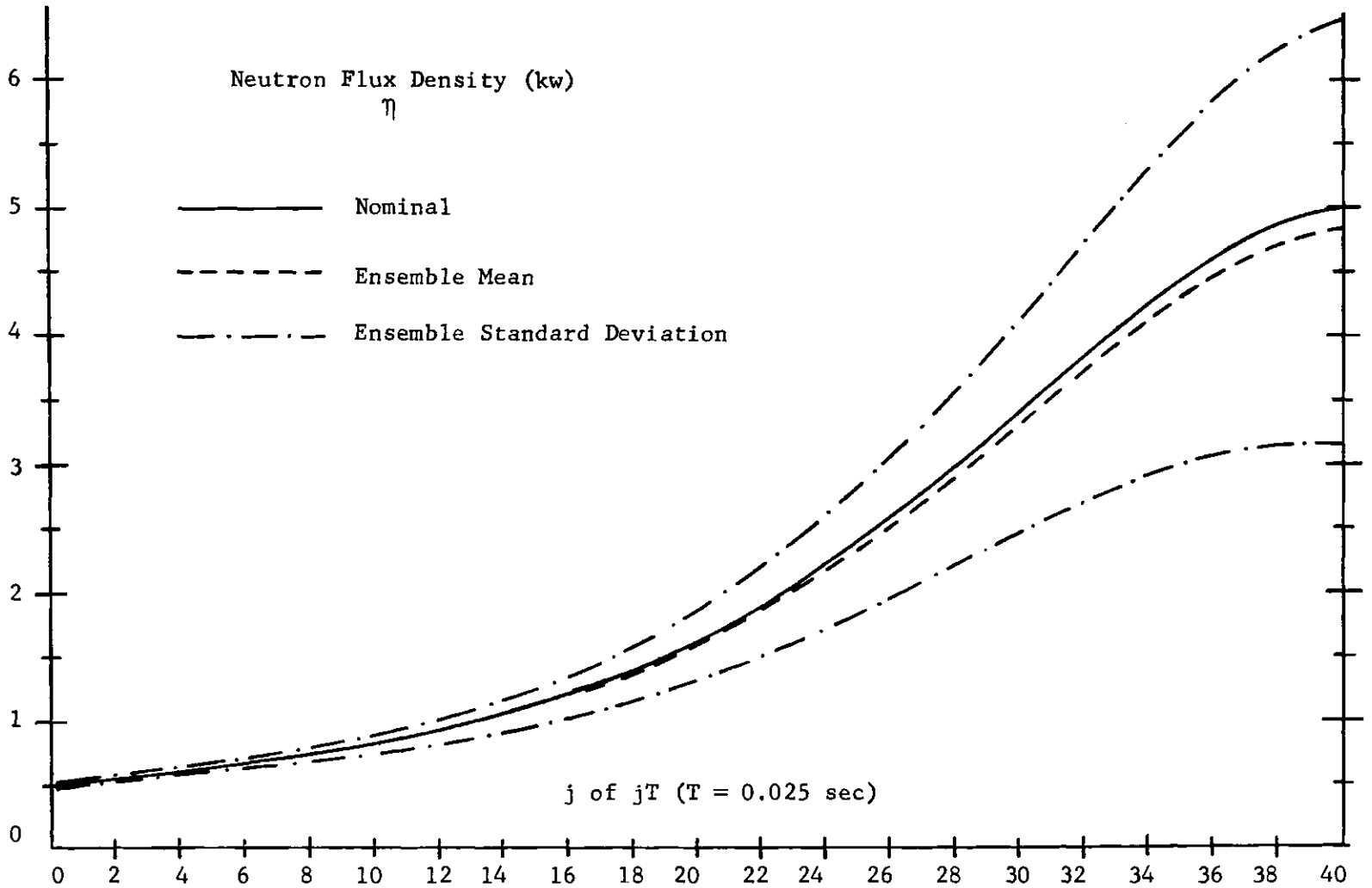

Figure 9(a,b). Open-Loop Monte Carlo Ensemble Mean and Standard Deviation

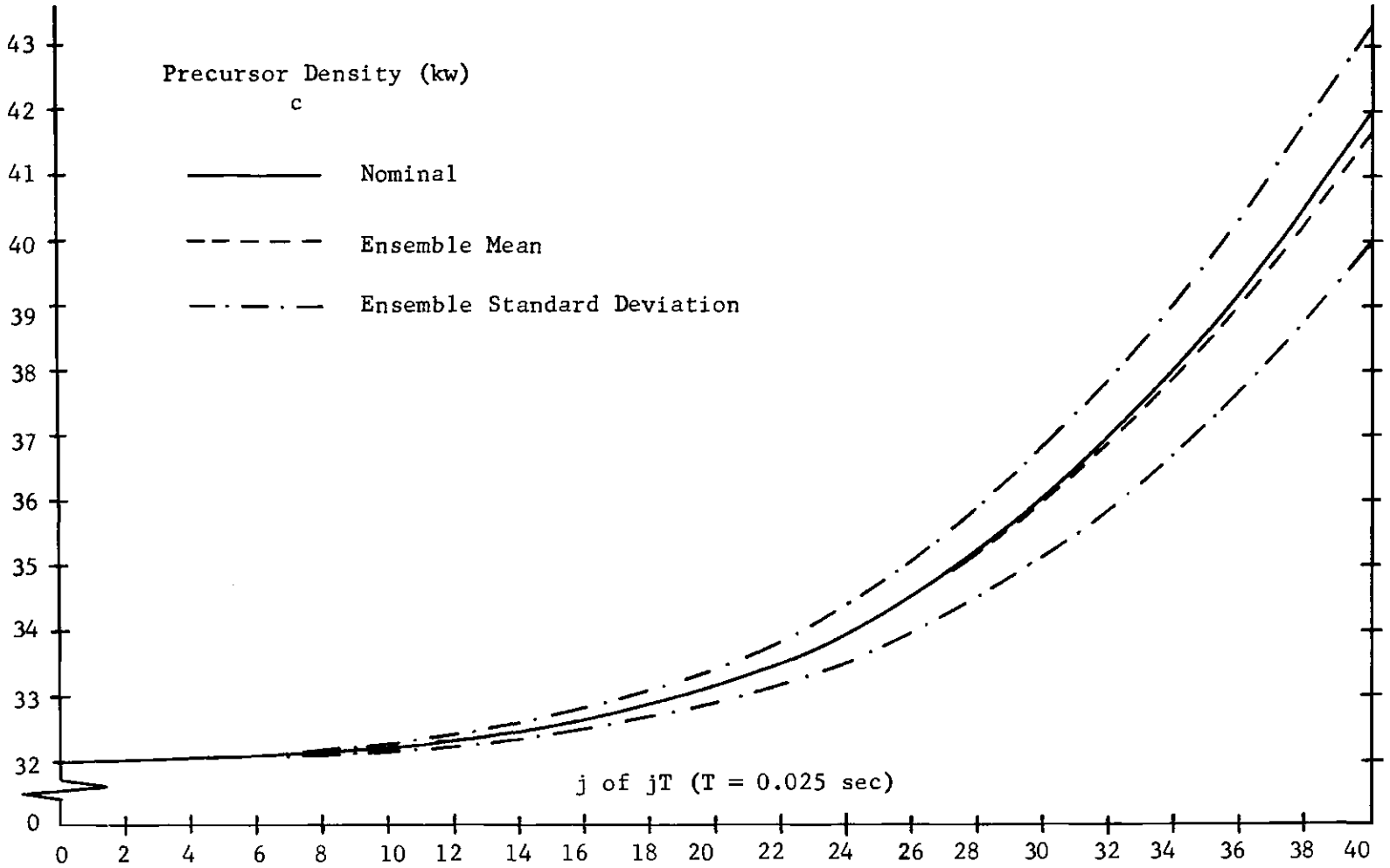
Precursor Density (kw)
c
Nominal
Ensemble Mean
Ensemble Standard Deviation
43
42
41
40
39
38
37
36
35
34
33
32
0
j of jT (T = 0.025 sec)
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40

Figure 9b.

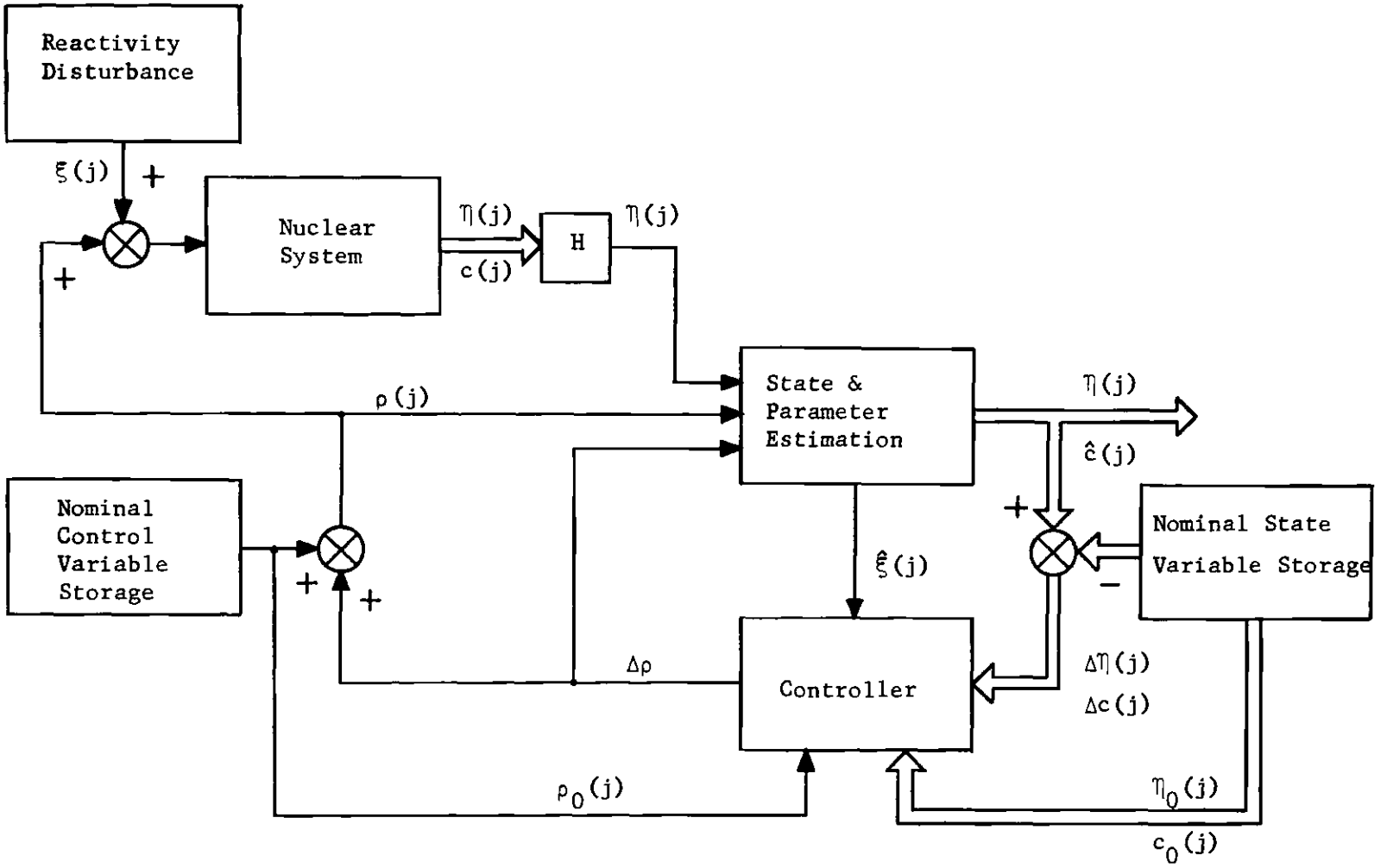

Figure 10. Block Diagram for the Perturbation Control of the Startup of a Nuclear Reactor

in Figure 8 so that they can be easily compared with the open-loop results. Notice that the closed-loop trajectories track the nominal trajectories extremely well. Another important feature to note is that even though an initial estimate of the precursor density is taken to be 30 kw instead of the actual 32 kw, the true value is identified in just one identification period. As was done with the open-loop response, a Monte Carlo simulation representing 100 runs was performed to evaluate, more thoroughly, the ability of the perturbation control scheme to track the nominal trajectory. Figure 11 shows a plot of the ensemble mean and standard deviation. It can be seen that the standard deviation represents a maximum deviation from the nominal of only two percent.

In order to demonstrate the effect continuous parameter updating has on the accuracy obtained by a linear perturbation controller, a Monte Carlo simulation representing 100 runs was again performed on the system, except now only the assumed nominal parameter values are used in evaluating $A_0(j)$ and $\underline{b}_0(j)$ about the nominal. There is still a need to use the state estimator portion of the identification algorithm to identify the unavailable precursor concentration state. Figure 12 shows a plot of the resulting ensemble mean and standard deviation. Notice that, without continuous parameter updating, the maximum deviation of the standard deviation curve from the nominal is approximately five percent. Thus, for this system, by continuously updating the parameter values, the accuracy with which the linear perturbation controller tracks the nominal trajectory is approximately 2 1/2 times greater than the accuracy obtained by using just the assumed nominal parameter values.

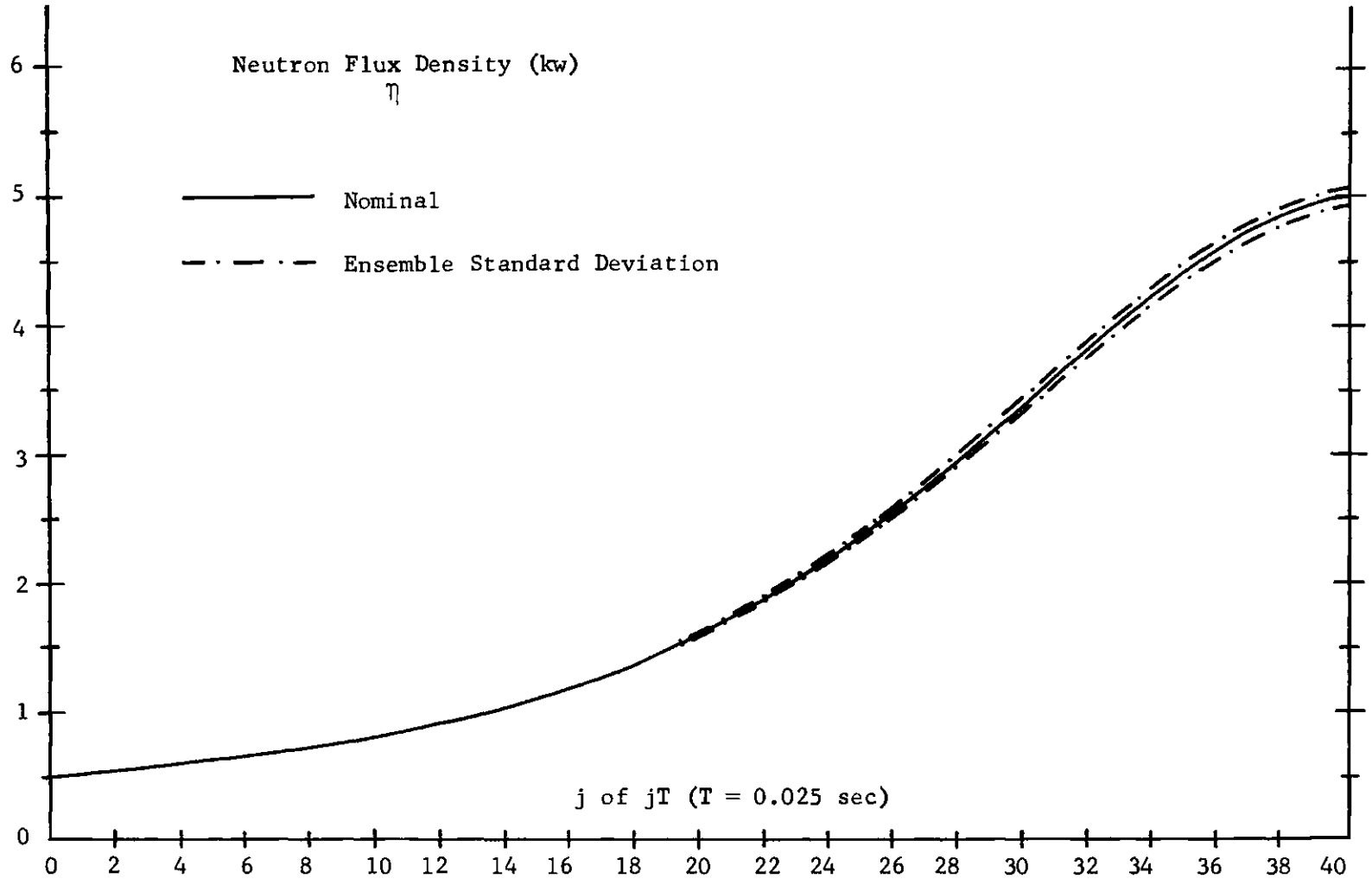

Figure 11(a,b). Monte Carlo Ensemble Standard Deviation Curves for a Perturbation Control Scheme Employing Continuous Parameter Updatings

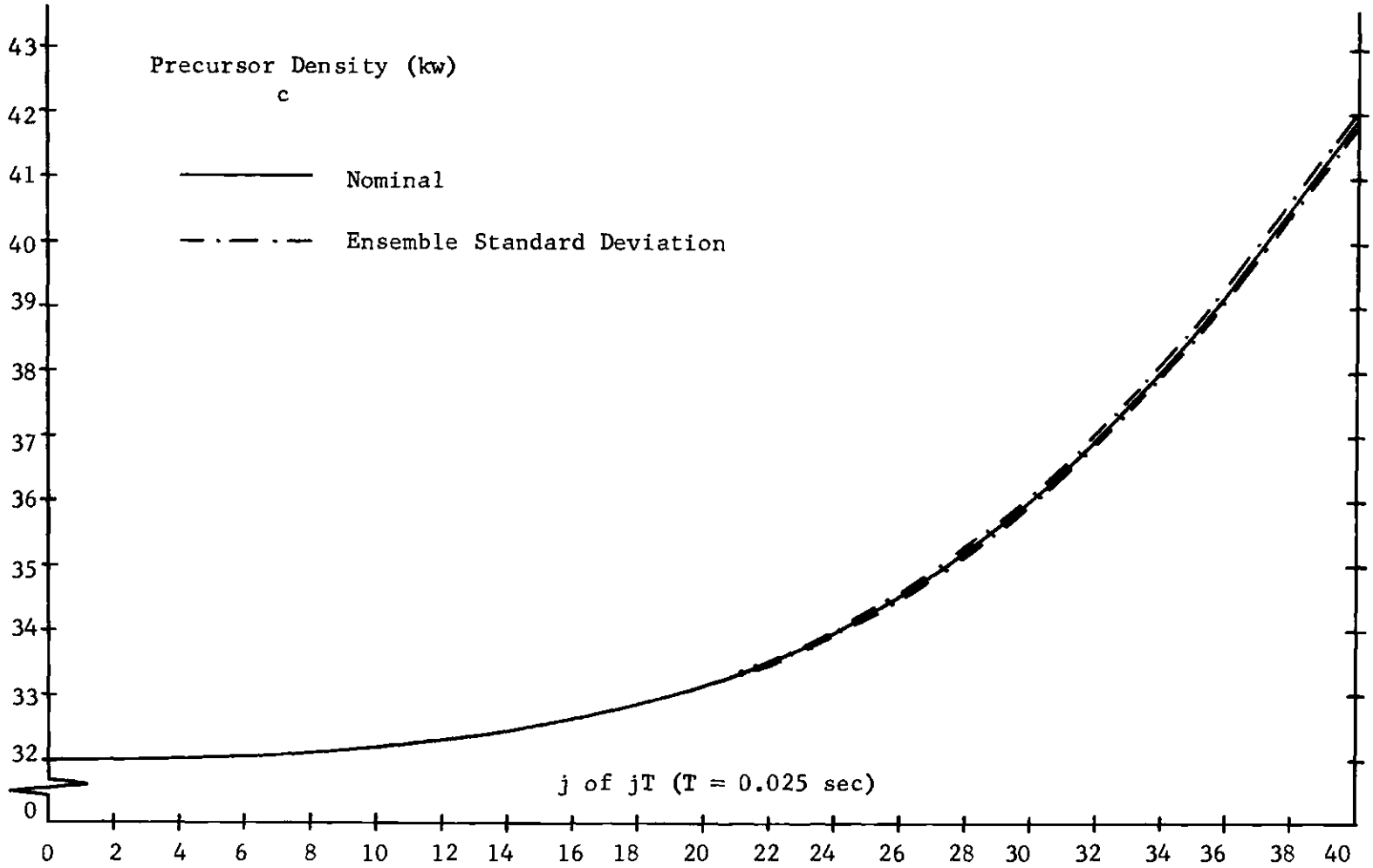
Precursor Density (kw)
c
Nominal
Ensemble Standard Deviation
43
42
41
40
39
38
37
36
35
34
33
32
0
j of jT (T = 0.025 sec)
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40

Figure 11b.

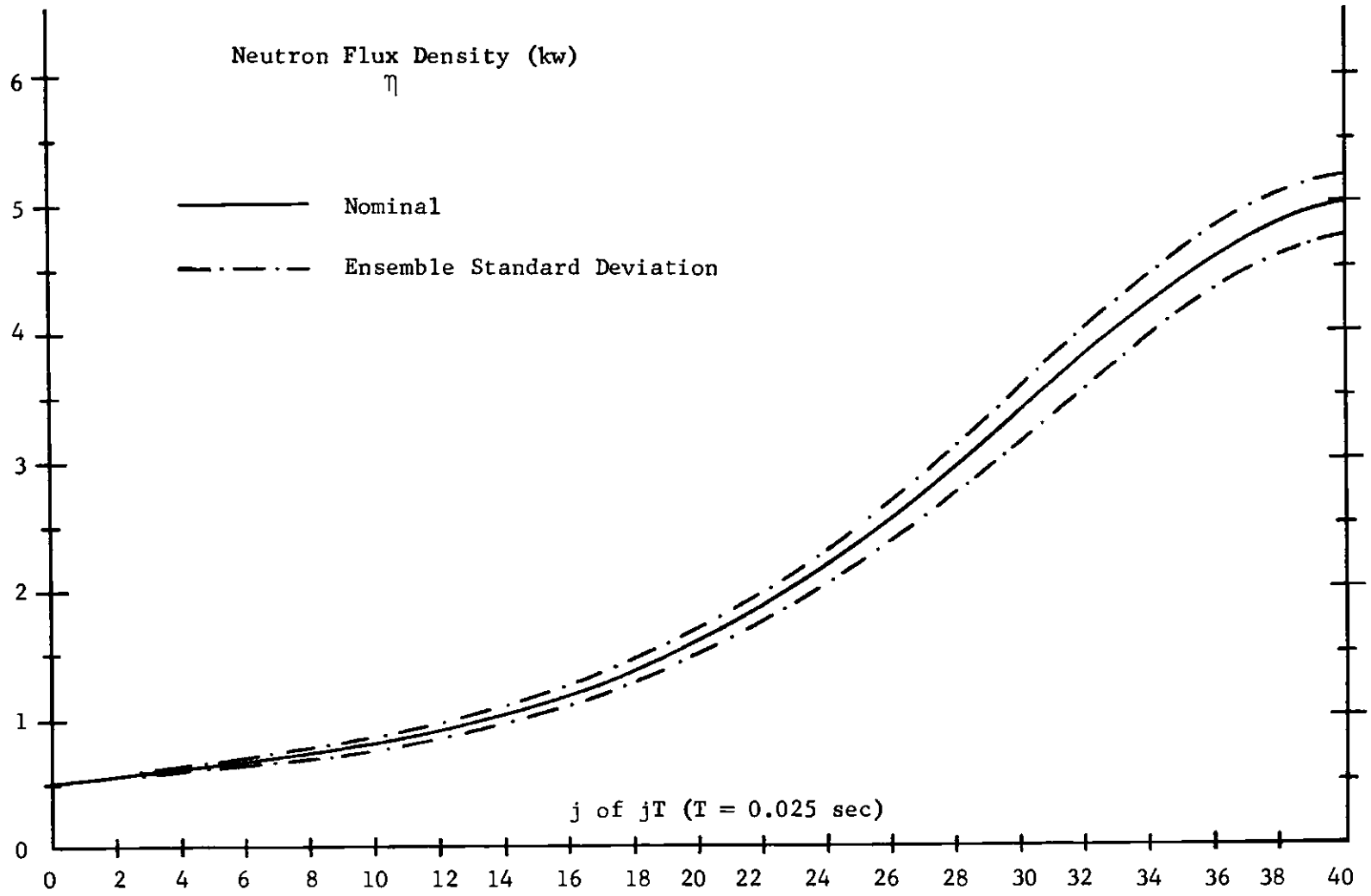

Figure 12(a,b). Monte Carlo Ensemble Standard Deviation Curves for a Perturbation Control Scheme Employing only Nominal Parameters

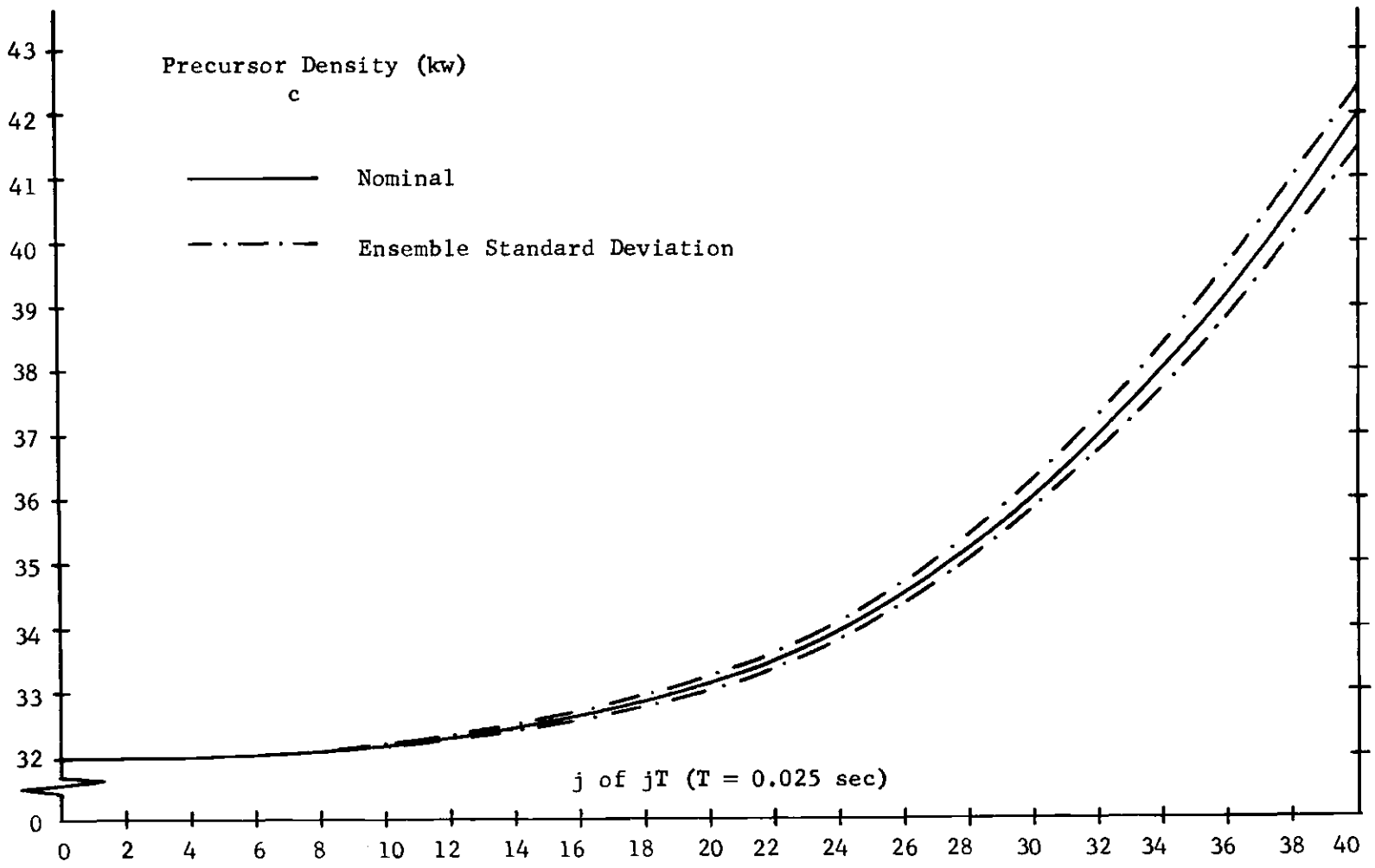
Precursor Density (kw)
c
Nominal
Ensemble Standard Deviation
43
42
41
40
39
38
37
36
35
34
33
32
0
j of jT (T = 0.025 sec)
0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40

Figure 12b.

To obtain some insight into the allowable parameter variations, the curves of Figure 13 are presented. The abscissas represent the time constant, $1/\mu$, of the correlation function, $R(\tau) = \sigma^2 e^{-\mu|\tau|}$, of the input disturbance, $\xi(j)$. Thus, going from right to left, the disturbance becomes more uncorrelated. The ordinate represents the magnitude of the gaussian white noise variance being fed to the correlation filter. The curves, representing three different noise ensembles, divide the graph into regions in which the identification algorithm converges to the correct solution and regions in which the algorithm diverges. As explained in Chapter IV, divergence occurs when the range on initial parameter estimates necessary to insure convergence for any identification cycle is exceeded. In obtaining these curves, once divergence occurs for any identification cycle, the complete system run is considered to be divergent. The graph can be interpreted in the following manner. The relatively flat portion of the curves in the lower left-hand region represents a threshold below which convergence occurs no matter how uncorrelated the disturbance. The reason the threshold exists is that no matter what the parameter values are at the beginning of each identification cycle, the change of these values from the previous cycle is always less than the range on the initial parameter estimates necessary to ensure convergence. As the noise becomes more correlated, a larger noise variance can be tolerated and still have the change in parameter values from one identification cycle to the next be within the range necessary for convergence. This explains why the curve has a positive slope. Notice that the slope of the curve changes at approximately one radian/sec. A plausible explanation for this behavior is based on the consideration that the closed-loop nuclear system, although

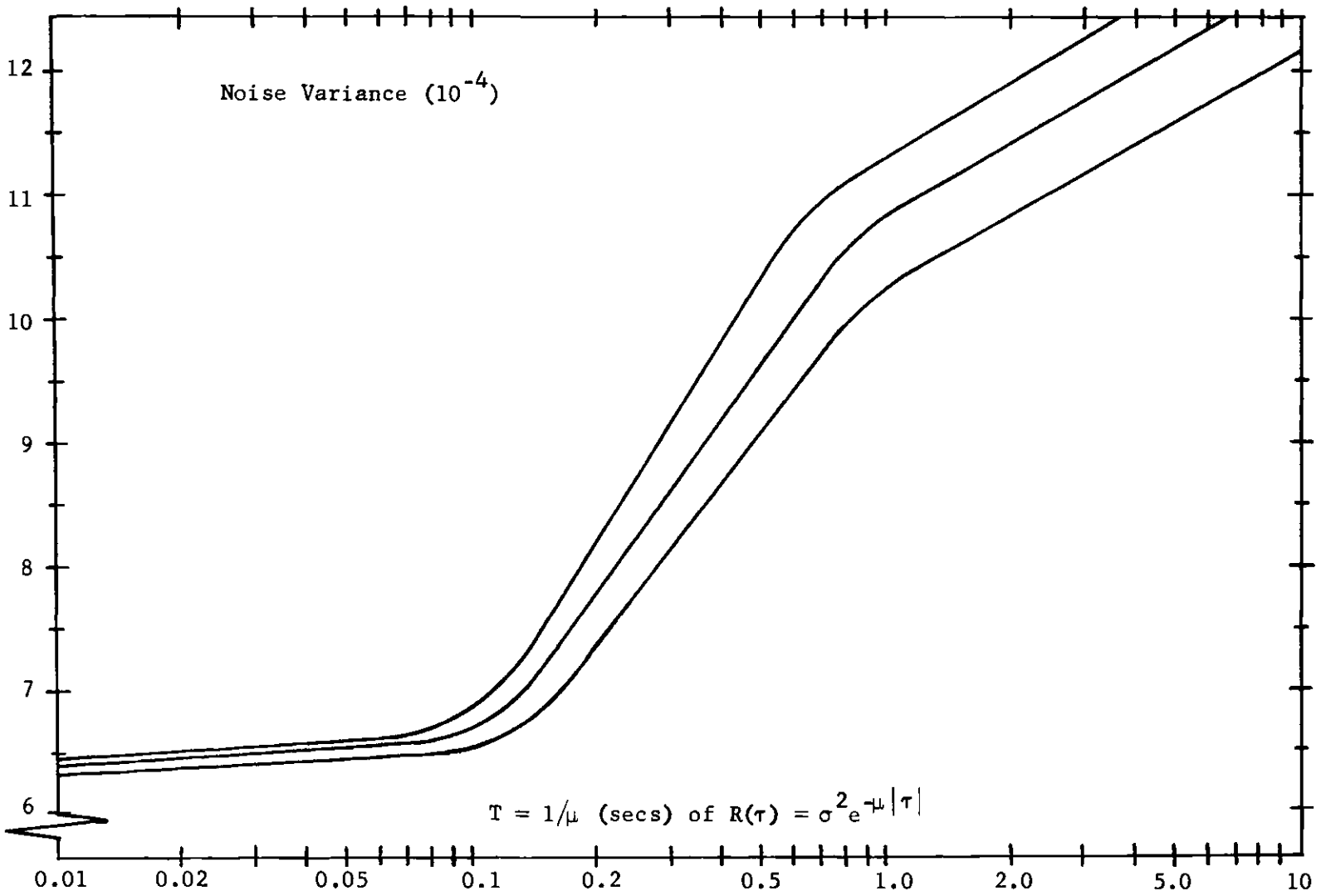

Figure 13. Noise Variance vs. Noise Correlation Necessary to Insure Convergence

time varying, has a bandwidth of approximately one radian/sec. Consequently, as the noise sequence becomes more correlated, more noise energy is concentrated in the lower frequencies, and hence within the bandwidth of the system. The more concentrated the noise energy appearing within the system bandwidth, the more likely that it will cause the identification algorithm to diverge. Thus, the rate of change of the variance must be reduced.

It would be desirable to have, for every correlation time, a histogram of the magnitudes of noise variances necessary to produce divergence for a number of noise ensembles. Unfortunately, the amount of computer time required to obtain such histograms is prohibitive.

Conclusions

This chapter has examined the effectiveness of the linear perturbation control scheme developed in Chapter V in controlling the startup of a thermal nuclear reactor. The reactor model equations consisted of a couple of first-order nonlinear equations, in which one of the plant states was not available as a system output, and a disturbance due to coolant flow through the core was represented as a random parameter variation.

First, by ignoring the disturbance, an optimum (nominal) open-loop control was found which minimized the energy required to drive the control rods while increasing the output power from 0.5 kw to 5 kw in one second. However, when the disturbance was included in a Monte Carlo system simulation, the resulting trajectories varied widely about this nominal.

To prevent such wide variations, the linear perturbation controller of Chapter V was employed. Besides identifying unavailable plant states, this controller tracks unknown parameter variations and uses the latest identified values in determining the optimum perturbation control. A Monte Carlo simulation of the nuclear system using this feedback controller resulted in an ensemble standard deviation curve which varied a maximum of two percent from the nominal. As a comparison, a Monte Carlo simulation of a perturbation control scheme employing only the assumed nominal parameter values resulted in a deviation of the ensemble standard deviation curve from the nominal of as much as five percent. Thus, a more accurate perturbation controller can be obtained if the unknown parameter values are continuously identified and used to update the perturbation control. The disadvantage, of course, is the increase in computer time necessary to accomplish this updating.

Finally, to obtain a feeling of the allowable parameter variations for the nuclear system, several noise ensemble curves were presented. These curves showed that if the change in the magnitude of the parameter values from one identification period to the next is sufficiently small, then rapidly time varying parameters can be accommodated. However, as the magnitude of the change in the parameter values increases, the time variation of the parameters must decrease.

CHAPTER VII

CONCLUSIONS AND RECOMMENDATIONS

Conclusions

This dissertation presents a closed-loop, on-line, parameter identification and control scheme for discrete time systems. Emphasis has been placed not only in developing the identification algorithm, but also coordinating the identification and control functions to strive to achieve an overall optimum closed-loop control law. The salient features of the scheme developed are that it:

1. Provides on-line identification.
2. Identifies not only unavailable plant states, but also unknown constant or slowly varying parameters.
3. Requires no external test signal.
4. Applies to both linear and nonlinear systems.
5. Always employs a control based on the latest parameter and plant state values.
6. Is digitally implemented.
7. Substantiated by theoretical analysis.

The general class of applicable systems is described in Chapter II. Basically, the plant must be described by a vector difference equation of the form

$$\underline{x}(j+1) = \underline{f}[\underline{x}(j),\underline{a}(j),u(j)] \ , \qquad (7.1)$$

where $\underline{x}(j)$ is the state vector, $\underline{a}(j)$ the unknown parameter vector, and $u(j)$ the control input. This equation can be either linear or nonlinear, but the function must be continuous for both $\underline{x}(j)$ and $\underline{a}(j)$, $\underline{f}_{\underline{x}(j)}$ and $\underline{f}_{\underline{a}(j)}$ must exist and be continuous, and $\underline{f}_{\underline{x}(j)}$ must satisfy a Lipschitz condition with respect to $\underline{x}(j)$, while $\underline{f}_{\underline{a}(j)}$ must satisfy a Lipschitz condition with respect to $\underline{a}(j)$. The system must also be observable, but need not be controllable. The only constraints placed on $\underline{a}(j)$ is that the parameters must either be unknown constants, or values that slowly vary with time.

The scheme functions as follows: for the plant described by equation (7.1), all available plant states are used by an identification algorithm in estimating the unavailable plant states and unknown parameters. These identified values are then used to determine an optimum closed-loop control of the form $u(j) = \gamma[\underline{x}(j),\underline{a}(j)]$. In this manner, the feedback control is always based on the latest identified values.

The identification algorithm is based on a Taylor Series expansion of an equation of the form of equation (7.1). To this expansion is adjoined an equation which restricts the unknown parameter vector, $\underline{a}(j)$, to remain constant over the identification interval. The combined equations thus represent a set of first order, linear, difference equations, the solution of which, subject to the appropriate boundary conditions, solves the identification problem. The boundary conditions are composed of the available plant outputs. In general, the combined parameter and state identification problem is nonlinear. Since a linearized identification algorithm is being used, this algorithm must be iterated so that,

under proper conditions, it will converge uniformly to the correct solution. Chapter IV presents a proof that such an iterative, linearized algorithm can converge to the correct solution, and indicates that there is a range of initial unknown parameter and state variable estimates for which convergence will occur. It is also shown that when convergence occurs, it occurs in a quadratic manner.

As previously stated, the latest identified parameter and plant state values are used to update closed-loop controls of the form $u(j) = \gamma[\underline{x}(j),\underline{a}(j)]$. There are a large class of systems for which closed-loop controls of the form expressed above can be found. This thesis investigates, thoroughly, one of the most important constituents of this class--namely, linear systems for which optimum closed-loop controls are found that minimize a quadratic performance criterion (so-called state-regulator systems).

In Chapter III, state-regulator systems for which all of the plant states are available are considered. It is shown that, for this case, the identification algorithm simplifies to a set of simultaneous algebraic equations which, except for singular or ill-conditioned solutions, converge to the correct solution in one iteration. Two methods of handling the singular or ill-conditioned solutions are considered. The method implemented throughout the thesis detects, then removes, the dependent or nearly dependent equations from the set. Additional equations are then added from the boundary conditions available at each discrete time period until a well-conditioned set of equations are obtained. The disadvantage of this scheme is that the length of the identification interval is

increased, such that the allowable parameter variations that can be tracked are reduced. The second method is to use a generalized inverse on the data available. Here, of course, a least-squares solution is generated which might, or might not, be acceptable to the designer. The latest identified parameter values are then used in recomputing the well known Riccati equation, from which an updated feedback control is obtained. The limitations in applying the identification and control scheme in this case depend on several factors including the order of the system versus the number of unknown parameters, the accuracy with which the set of simultaneous equations can be solved, and the time available between discrete time periods so that the identification and control updating can be accomplished. Considering all, it is felt that up to a twentieth order system can be accommodated with 10-15 unknown parameters.

The state-regulator case in which only some of the plant states are available is considered in Chapter IV. Unfortunately, for this case, the identification algorithm usually does not simplify to a set of linear algebraic equations as it did above. Thus the identification algorithm must be employed as previously described. This case has the same limitations as the previous case, but in addition, a range of initial parameter and plant state estimates necessary to ensure convergence must be considered. It was shown that this range is a function, among other things, of the order of the system and the number of unknown parameters that must be identified. The larger the system and/or the greater the number of unknown parameter and plant state values that must be identified, the smaller the initial estimate tolerance allowed for each parameter and

plant state value. Considering the tolerance allowed for the second order system of Chapter IV with three unknown values, it is felt that only up to a tenth order system with a maximum of five unknown parameter and plant state values can be considered and still have a reasonable tolerance on the initial value estimates.

For those systems in which a closed-loop control of the form $u(j) = \gamma[\underline{x}(j),\underline{a}(j)]$ is not possible, this thesis has considered in Chapter V the use of the identification algorithm in conjunction with a linear perturbation controller. Besides identifying all unavailable plant states, the identification algorithm also tracks questionable parameter variations, with the latest values being used in the calculation of the perturbation control. Thus, the use of the identification algorithm with a linear perturbation controller is very similar to its use for state-regulator systems.

In Chapter VI, the effectiveness of the perturbation control scheme mentioned above is examined for the practical problem of controlling the startup of a thermal nuclear reactor. The reactor is modeled by two first-order nonlinear equations, in which one of the plant states is not available as a system output, and a disturbance due to coolant flow through the core is represented as a random parameter variation. By ignoring the disturbance, an optimum (nominal) open-loop control is found which minimizes the energy required to drive the control rods while increasing the output power from 0.5 kw to 5 kw in one second. However, when the disturbance is included in a Monte Carlo system simulation, the resulting trajectories vary widely about the nominal. The identification-linear perturbation controller is then employed to minimize these variations.

A Monte Carlo simulation using this feedback controller results in an ensemble standard deviation curve which varies a maximum of two percent from the nominal. As a comparison, a Monte Carlo simulation of a perturbation control scheme employing only assumed nominal parameter values, results in a deviation of the ensemble standard deviation curve from the nominal of as much as five percent. Thus, a more accurate perturbation controller can be obtained if the unknown parameter values are continuously identified and used to update the perturbation control. The disadvantage, of course, is the increase in computer time necessary to accomplish the updating.

Finally, to obtain a feeling of the allowable parameter variations for the nuclear system, several noise ensemble curves are presented. These curves show that if the change in the magnitude of the parameter values from one identification period to the next is sufficiently small, then rapidly time varying parameters can be accommodated. However, as the magnitude of the change increases, the time variation of the parameters must decrease.

Recommendations

There are several areas of study for extensions of the research presented in this dissertation. These extensions deal with increasing the applicability of the scheme developed.

One of the biggest problems in applying the identification algorithm is obtaining initial parameter and state estimates for each identification cycle that fall within the range of values necessary to insure

convergence. Methods, such as perhaps, differential approximation, should be sought that can provide, on-line, better initial estimates.

Secondly, methods can be applied which cut down the computation time needed in recomputing the closed-loop control. Such methods usually take advantage of the fact that the Riccati gain for a time invariant system approaches a constant value rapidly.

APPENDIX

COMPUTATION REQUIREMENTS FOR AN IDENTIFICATION CYCLE

In this appendix, the computational requirements necessary to implement one identification cycle for the systems considered in Chapter III are obtained. These systems are linear systems for which all the plant states are available as system outputs. As shown in the chapter, the identification algorithm converges to the correct solution in one iteration. The work of this appendix is directly applicable to systems for which convergence requires more than one iteration--the computation time being proportional to a multiple of the computation time required for one iteration. The computational requirements are obtained as explicit functions of the dimensions of the system's state vector (n-dimensional), unknown parameter vector (m-dimensional), and control vector (r-dimensional).

This appendix is based on an article written by Jerry M. Mendel [50]. In the article, Mendel defines the computation time required to implement a discrete Kalman filter in terms of a unit cycle time (T_u), and the dimensions of the system's state, measurement, and disturbance vectors. The work considers not only the number of multiplications and additions needed for the implementation, but also the amount of logic time required for properly controlling and sequencing the operations for an assumed computer configuration. The number of multiplications, the number of additions, and the logic time requirements of this appendix are

based solely on this article.

The equations implemented include the identification algorithm, equation (3.13); a Riccati equation based on a vector control, analogous to equation (3.5); the Riccati equation, equation (3.5); the feedback control equation, equation (3.4); and the state equation, equation (3.1). Tables 3-7 indicate the computation requirements for these equations. The total computational time requirement for a vector control (CT_v) is given by

$$CT_v = [(M+ \{j_f - j\}M_1 + M_2 + M_3) \cdot MUL + 2(A + \{j_f - j\}A_1 + A_2 + A_3)$$

$$+ (L + \{j_f - j\}L_1 + L_2 + L_3)] \cdot T_u ,$$

where $j_f - j$ is the number of discrete time periods remaining until the end of the system run, MUL is the execution time (in unit times) required for multiplication, and T_u denotes the basic unit time. As an example of the time required, let $n = m = 3$, $r = 2$, $j_f - j = 10$, MUL = 6, DIV = 12, and $T_u = 1\ \mu$sec. The total computation time required is then

$$CT_v = [(18 + 10 \cdot 135 + 81 + 27) \cdot 6 + 2(18 + 10 \cdot 117 + 72 + 24)$$

$$+ (4,370 + 10 \cdot 5,818 + 4,980 + 608)]\ 10^{-6} \text{ sec} ,$$

$$CT_v = [\ 1476 \cdot 6 + 2 \cdot 1284 + 68,138]\ 10^{-6} \text{ sec},$$

$$CT_v = 0.079562 \simeq 0.08 \text{ sec} .$$

Table 3. Computational Requirements for the Identification Algorithm

Defining Equation: $G^N(k)\underline{a}^{N+1}(k) = \underline{x}_p(k+1) + G^N(k)\underline{a}^N(k) - \underline{f}[\underline{x}^N(k),\underline{a}(k)]$*

Computations	Number of Multiplications	Number of Additions	Logic Time
$G^N(k)\underline{a}^N(k)$	m^2	$m^2 - m$	$10 + 6m^2 + 37m$
$G^N(k)\underline{a}^N(k) - \underline{f}[\underline{x}^N(k),\underline{a}(k)]$		m	$27 + 5m + MUL$
$\underline{x}_p(k+1) + G^N(k)\underline{a}^N(k) - \underline{f}$		m	$27 + 5m + MUL$
$[G^N(k)]^{-1}$	m^3	m^3	$10 + 41m^3 + 140m^2 + 92m + MMORE$**
$[G^N(k)]^{-1}[\underline{x}_p + G^N(k)\underline{a}^N(k) - \underline{f}]$	m^2	$m^2 - m$	$10 + 6m^2 + 37m$
Total	M = $m^3 + 2m^2$	A = $m^3 + 2m^2$	L = $84 + 41m^3 + 152m^2 + 142m + 2MUL + MMORE$

*Assumes $\underline{f}[\underline{x}^N(k),\underline{a}(k)]$ and $G^N(k)$ are precomputed.

**MMORE = $7.5m^4 + DIV(2m^3 + m) + MUL(0.5m^2 + 2.5m)$

Table 4. Computational Requirements for the Riccati Equation Based on a Vector Control

Defining Equation: $P(k) = Q + A^T P(k+1)[I + BR^{-1}B^T P(k+1)]^{-1} A$ *

Computations	Number of Multiplications	Number of Additions	Logic Time
$BR^{-1}B^T P(k+1)$	n^3	$n^3 - n^2$	$10 + 6n^3 + 21n^2 + 16n$
$I + BR^{-1}B^T P(k+1)$		n^2	$27 + 5n^2 +$ MUL
$[I + BR^{-1}B^T P(k+1)]^{-1} = X$	n^3		$10 + 41n^3 + 140n^2 + 92n +$ MORE**
XA	n^3	$n^3 - n^2$	$10 + 6n^3 + 21n^2 + 16n$
$A^T PXA$	$2n^3$	$2n^3 - 2n^2$	$20 + 12n^3 + 42n^2 + 32n$
$Q + A^T PXA$		n^2	$27 + 5^{n2} +$ MUL
Total	$M_1 = 5n^3$	$A_1 = 5n^3 - 2n^2$	$L_1 = 104 + 65n^3 + 234n^2 + 156n$ + 2 MUL + MORE

*Assumes $BR^{-1}B^T$ is precomputed.

**MORE = $7.5\ n^4$ + DIV$(2n^2 + n)$ + MUL$(0.5n^2 + 2.5n)$.

Table 5. Computational Requirements for Updating the Control

Defining Equation: $\underline{u}(k) = - B^T[A^{-1}]^T[P(k) - Q]\underline{x}(k)$ *

Computations	Number of Multiplications	Number of Additions	Logic Time
$P(k) - Q$		n^2	$27 + 5n^2 + MUL$
$[P(k) - Q]\underline{x}(k)$	n^2	$n^2 - n$	$10 + 6n^2 + 37n$
$[A^{-1}]^T$	n^3	n^3	$10 + 41n^3 + 140n^2 + 92n + MORE$**
$[A^{-1}]^T[P(k) - Q]\underline{x}(k)$	n^3	$n^3 - n^2$	$10 + 6n^3 + 21n^2 + 16n$
$- B^T[A^{-1}]^T[P(k) - Q]\underline{x}(k)$	n^2r	$n^2r - nr$	$10 + 6n^2r + 21nr + 16n$
Total	$M_2 = 2n^3 + n^2 + n^2r$	$A_2 = 2n^3 + n^2 - n + n^2r - nr$	$L_2 = 67 + 47n^3 + 172n^2 + 161n + 6n^2r + 21nr + MUL + MORE$

*Assumes $(- B^T)$ is precomputed.

**MORE = $7.5n^4 + DIV(2n^2 + n) + MUL(0.5n^2 + 2.5n)$.

Table 6. Computational Requirements for Updating the State-Vector

Defining Equation: $\underline{x}(k+1) = A\underline{x}(k) + B\underline{u}(k)$

Computations	Number of Multiplications	Number of Additions	Logic Time
$B\underline{u}(k)$	n^2r	$n^2r - n^2$	$10 + 6n^2r + 21n^2 + 16n$
$A\underline{x}(k)$	n^2	$n^2 - n$	$10 + 6n^2 + 37n$
$A\underline{x}(k) + B\underline{u}(k)$		n^2	$27 + 5n^2 + MUL$
Total	$M_3 = n^2r + n^2$	$A_3 = n^2r + n^2 - n$	$L_3 = 47 + 6n^2r + 32n^2 + 53n + MUL$

Table 7. Computational Requirements for the Riccati Equation Based on a Scalar Control

Defining Equation: $P(k) = Q + A^T P(k+1)A - \frac{A^T P(k+1)\underline{b}\underline{b}^T P(k+1)A}{1 + \underline{b}^T P(k+1)\underline{b}}$ *

Computations	Number of Multiplications	Number of Additions	Logic Time
$A^T P(k+1)\underline{b}\underline{b}^T P(k+1)A$	$5n^3$	$5n^3 - 5n^2$	$50 + 30n^3 + 105n^2 + 80n$
$\underline{b}^T P(k+1)\underline{b}$	$2n^2$	$2n^2 - 2n$	$20 + 12n^2 + 74n$
$1 + \underline{b}^T P(k+1)\underline{b}$		1	32 + MUL
$\frac{1}{1 + \underline{b}^T P(k+1)\underline{b}} = Y$			10 + DIV
$YA^T P(k+1)\underline{b}\underline{b}^T P(k+1)A = Z$	n^2		8n
$Q + A^T P(k+1)A + Z$		$2n^2$	$54 + 10n^2$ + 2 MUL
Total	$M_4 = 5n^3 + 3n^2$	$A_4 = 5n^3 - n^2 - 2n + 1$	$L_4 = 166 + 30n^3 + 107n^2 + 163n$ + 3 MUL + DIV

*Assumes $\underline{b}\underline{b}^T$ is precomputed.

When only a scalar control is required, computation time can be saved by using the Riccati equation of Table 7. The savings arise because there is no inverse to compute as there is for the Riccati equation of Table 4. The total computation time requirement (CT_s) is now given by

$$CT_s = [(M + M_2 + M_3 + \{j_f - j\}M_4) \cdot MUL$$

$$+ 2(A + A_2 + A_3 + \{j_f - j\}A_4)$$

$$+ (L + L_2 + L_3 + \{j_f - j\}L_4)] \cdot T_u \ .$$

Considering the same example as before with the exception of a scalar control, the total computation time required is now

$$CT_s = [(18 + 10 \cdot 162 + 72 + 18) \cdot 6 + 2(18 + 66 + 15 + 10 \cdot 121)$$

$$+ (4,370 + 4,863 + 544 + 10 \cdot 2,458)] \ 10^{-6} \text{ sec} \ ,$$

$$CT_s = [1728 \cdot 6 + 2 \cdot 1309 + 34,367] \ 10^{-6} \text{ sec} \ ,$$

$$CT_s = 0.047353 \text{ sec} \simeq 0.047 \text{ sec} \ .$$

The quantity $j_f - j = 10$ was chosen because seldom would a designer continue to compute the Riccati gain in the manner illustrated in the tables past $j_f - j = 10$. Instead, he would take advantage of the fact that the Riccati gain quickly reaches a steady-state value.

BIBLIOGRAPHY

1. A. A. Fel'dbaum, "Theory of Dual Control, I, II, III, IV," Automn. and Remote Control 21, pp. 1240-49; 1453-64 (1970); 22, pp. 3-16, 129-143 (1961).

2. P. D. Joseph, J. T. Tou, "On Linear Control Theory," A.I.E.E. Transactions on Applications and Industry 80, pp. 193-196, September (1961).

3. D. L. Alspach, H. W. Sorenson, "A Separation Principle for Non-gaussian Noise," International Journal of Control, (To Appear).

4. C. Striebel, "Sufficient Statistics in the Optimum Control of Stochastic Systems," Journal of Math. Anal. and Appl. 12, No. 3, pp. 576-592 (1965).

5. Y. C. Ho, R. C. K. Lee, "Identification of Linear Dynamic Systems," Information and Control 8, 93 (1965).

6. R. C. K. Lee, Optimal Estimation, Identification, and Control (Cambridge, Massachusetts: The M.I.T. Press, 1964).

7. G. N. Saridis, G. Stein, "Stochastic Approximation Algorithms for Linear Discrete--Time System Identification," I.E.E.E. Transactions on Automatic Control, AC-13, No. 5, pp. 515-523, October (1968).

8. A. E. Albert, L. A. Gardner, Jr., Stochastic Approximation and Nonlinear Regression (Cambridge, Massachusetts: The Riverside Press, 1967).

9. A. E. Bryson, Y. C. Ho, Applied Optimal Control (Waltham, Massachusetts: Ginn-Blaisdell, 1969).

10. H. J. Kushner, "Near Optimal Control in the Presence of Small Stochastic Perturbations," Proc. 5th Joint Automatic Control Conference (Stanford University, June 1964), pp. 392-396.

11. W. J. Culver, M. D. Mesarovic, "Dynamic Statistical Linearization," A.E.E.E. Transactions on Computing Devices, pp. 317-327, July (1963).

12. G. N. Saridis, G. Stein, "A Parameter--Adaptive Control Technique," Automatica 5, 731 (1969).

BIBLIOGRAPHY (Continued)

13. G. N. Saridis, R. T. Kitahara, "Comparison of Per-Interval and Overall Parameter--Adaptive Self-Organizing Control," I.E.E.E. Symposium on Adaptive Processes (1969).

14. J. B. Farison, R. E. Graham, R. C. Shelton, Jr., "Identification and Control of Linear Discrete Systems," I.E.E.E. Transactions on Automatic Control, AC-12, No. 4, pp. 438-442, August (1967).

15. G. L. Weegmann, J. B. Farison, "Identification--Adaptive Quadratic-Cost Linear System Control," I.E.E.E. Transactions on Automatic Control (Correspondence), pp. 395-397, June (1970).

16. W. J. Murphy, "Optimal Stochastic Control of Discrete Linear Systems with Unknown Gain," Proc. 9th Joint Automatic Control Conference (Ann Arbor, Michigan, June 1968), pp. 45-53.

17. Y. Bar-Shalom, R. Sivan, "The Optimal Control of Discrete Time Linear Systems with Random Parameters," Proc. 2nd Annual Princeton Conference on Information Sciences and Systems (Princeton, N.J., March 1968), pp. 292-297.

18. M. Cuenod, A. P. Sage, "Comparison of Some Methods Used for Process Identification," I.F.A.C. (Prague, June 1967), published in Automatica 4, pp. 235-268 (1968).

19. R. E. Nieman, D. G. Fisher, D. E. Seborg, "A Review of Process Identification and Parameter Estimation Techniques," Int. J. Control 13, No. 2, pp. 209-264 (1971).

20. M. Cuenod, A. E. Durling, An Introduction to Impulse Analysis (New York: Academic Press, 1967).

21. N. A. Anderson, "Step Analysis Method of Finding Time Constants," Instr. Control Systems, 130, November (1963).

22. V. Klein, "The Identification of Linear Systems by Means of Frequency Response," I.F.A.C. (London, 1966).

23. W. W. Lichtenberger, "A Technique of Linear System Identification Using Correlating Filters," I.R.F. Transactions on Automatic Control, AC-6, 183 (1961).

24. H. Wilson, "Cross-Correlation Analyzer Using Pseudo-Random Binary Signals," Electron. Engng. 41, No. 495, 66 (1969).

BIBLIOGRAPHY (Continued)

25. W. D. T. Davies, System Identification for Self-Adaptive Control (Bath, Great Britain: Pitman Press, 1970).

26. D. D. Donalson, F. H. Kishi, "Review of Adaptive Control System Theories and Techniques," in Modern Control System Theory, Ed. by C. T. Leondes (New York: McGraw-Hill, 1965).

27. Ibid.

28. A. P. Sage, Optimum Systems Control (Englewood Cliffs, N.J.: Prentice-Hall, 1968).

29. E. H. Lowe, III, "Design of Model Reference Adaptive Control Systems Using Liapunov Theory," Ph.D. Thesis, School of Electrical Engineering, Georgia Institute of Technology, 1969.

30. M. Margolis, C. T. Leondes, "A Parameter Tracking Servo for Adaptive Control Systems," I.R.E. Transactions on Automatic Control, AC-14, No. 2, 100, November (1959).

31. D. D. Donalson, C. T. Leondes, "A Model Reference Parameter Tracking Technique for Adaptive Control Systems," I.E.E.E. Transactions on Applications and Industry 82, 241, September (1963).

32. R. M. Dressler, "An Approach to Model--Reference Adaptive Control Systems," I.E.E.E. Transactions on Automatic Control, AC-12, No. 1, 75, February (1967).

33. P. V. Osburn, H. P. Whitaker, A. Kezer, "New Developments in the Design of Model Reference Adaptive Systems," IAS Report No. 61-39, January (1961).

34. M. Margolis, C. T. Leondes, "On the Theory of Adaptive Control Systems; the Learning Model Approach," I.F.A.C. (Moscow, 1960).

35. J. Rissanen, "On the Theory of Self-Adjusting Models," Automatica 1, 297 (1963).

36. P. F. Klubnikin, "The Realization of a Self-Adapting Control Programme in a System with a Digital Computer," I.F.A.C. (Basle, 1963).

37. R. M. Bakke, "Adaptive Direct Digital Control With Multi-Parameter Adjustment," Proc. 2nd I.F.A.C. Symposium on Theory of Self-Adaptive Control Systems (Teddington, England, September 14-17, 1965).

BIBLIOGRAPHY (Concluded)

38. A. N. Kolmogorov, S. V. Fomin, Elements of the Theory of Functions and Functional Analysis (Rochester, N.Y.: Graylock Press, 1957).

39. Sage, op. cit., pp. 130-131.

40. J. A. Cadzow, H. R. Martens, Discrete-Time and Computer Control Systems (Englewood Cliffs, N.J.: Prentice-Hall, 1970), pp. 302-05.

41. A. Ralston, A First Course in Numerical Analysis (New York: McGraw-Hill, 1965), pg. 398.

42. R. Penrose, "On Best Approximate Solutions of Linear Matrix Equations," Proceedings of the Cambridge Philosophical Society, Vol. 52, No. 1, 15 (1956).

43. J. B. Rosen, "Minimum and Basic Solutions to Singular Linear Systems," S.I.A.M. Journal on Applied Mathematics, Vol. 12, No. 1, pp. 156-162 (1964).

44. A. Klinger, "Approximate Pseudoinverse Solutions to Ill-Conditioned Linear Systems," Journal of Optimization Theory and Applications, Vol. 2, No. 2, pp. 117-124 (1968).

45. M. Athans, "The Role and Use of the Stochastic Linear-Quadratic-Gaussian Problem in Control System Design," I.E.E.E. Transactions on Automatic Control, AC-16, No. 6, 535, December (1971).

46. M. A. Schultz, Control of Nuclear Reactors and Power Plants (New York: McGraw-Hill, 1961).

47. M. Ash, Nuclear Reactor Kinetics (New York: McGraw-Hill, 1965).

48. J. A. Thie, "Statistical Analysis of Power-Reactor Noise," Nucleonics, Vol. 17, No. 10, 102, October (1959).

49. G. W. Masters, A. P. Sage, "Identification and Modeling of States and Parameters of Nuclear Reactor Systems," I.E.E.E. Transactions on Nuclear Science, February (1967).

50. J. M. Mendel, "Computational Requirements for a Discrete Kalman Filter," I.E.E.E. Transactions on Automatic Control, AC-16, No. 6, 748, December (1971).

VITA

James M. Fowler III was born in Marietta, Georgia on June 9, 1941. He entered the University of Florida in 1959 and received a B.E.E. degree in 1963. The ensuing five years were spent as a design engineer, then a project engineer, at Electro-Mechanical Research, Inc. in Sarasota, Florida. During this period he also received his M.S.E.E. degree from the University of Florida's extension at St. Petersburg, Florida. In 1968, he received a National Science Foundation Trainee Fellowship to work on a Ph.D. at Georgia Tech. In 1971, he was appointed a graduate teaching assistantship in the school of Electrical Engineering. He will complete the requirements for his Ph.D. in Electrical Engineering in the field of system identification and control in the spring of 1973.

In December, 1963, Mr. Fowler married the former Marjorie Lynn Hine of Chattanooga, Tennessee. He presently has one daughter, Lynley Allison.