University of Tennessee, Knoxville
TRACE: Tennessee Research and Creative Exchange

Masters Theses

Graduate School

3-1974

A discrete-time approach to process modeling and direct digital control.

Dennis K. Jones
University of Tennessee

Follow this and additional works at: https://trace.tennessee.edu/utk_gradthes

Recommended Citation

Jones, Dennis K., "A discrete-time approach to process modeling and direct digital control.. " Master's Thesis, University of Tennessee, 1974.
https://trace.tennessee.edu/utk_gradthes/5803

This Thesis is brought to you for free and open access by the Graduate School at TRACE: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Masters Theses by an authorized administrator of TRACE: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.

To the Graduate Council:

I am submitting herewith a thesis written by Dennis K. Jones entitled "A discrete-time approach to process modeling and direct digital control.." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Chemical Engineering.

Charles F. Moore, Major Professor

We have read this thesis and recommend its acceptance:

Accepted for the Council:
Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

March 15, 1974

To the Graduate Council:

I am submitting herewith a thesis written by Dennis K. Jones entitled "A Discrete-Time Approach to Process Modeling and Direct Digital Control." I recommend that it be accepted for 15 quarter hours of credit in partial fulfillment of the requirements for the degree of Master of Science, with a major in Chemical Engineering.

Charles F. Moore

Major Professor

We have read this thesis and recommend its acceptance:

E. E. Stansbury

H. W. Haw

Accepted for the Council:

Hilton A. Smith

Vice Chancellor for
Graduate Studies and Research

A DISCRETE-TIME APPROACH TO PROCESS MODELING AND DIRECT DIGITAL CONTROL

A Thesis

Presented to

The Graduate Council of

The University of Tennessee

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

---

by

Dennis K. Jones

June 1974

ACKNOWLEDGEMENTS

The author would like to acknowledge the following people, all of whom were instrumental in the completion of this thesis:

My parents, for their moral and financial support throughout my education.

My wife, Susie, for her companionship, encouragement, and patient understanding.

Dr. Charles F. Moore, my advisor, who initiated this project and without whose guidance its completion would have been impossible.

Dr. Homer F. Johnson, for arranging financial support in the form of a teaching assistantship.

Mrs. Joann Arms, for her excellent typing.

ABSTRACT

The purpose of this study was to investigate the advantages offered by a z-transform approach to direct digital control applications. A discrete-time modeling package was developed for modeling process input/output data using a general second-order pulse transfer function. Also, a z-transform controller presented in the literature was modified to yield a control algorithm which gives significant improvement over conventional DDC algorithms.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE PAGE

FIGURE PAGE

CHAPTER 1

INTRODUCTION

In the last decade, advances in computer technology have resulted in increased use of the digital computer to directly control industrial processes. The memory, logic, and computational capabilities of the computer, coupled with modern computer/process interface equipment, make possible the implementation of advanced control techniques and have served to establish the area of computer control as an important, and promising, segment of chemical process control. However, despite the computer's vast flexibility as compared to analog controllers, industrial applications of direct digital control (DDC) are usually little more than discrete versions of conventional analog control schemes. The design freedom afforded the control engineer has rarely been used to exploit the full potential of the digital computer at this first level of control.

At The University of Tennessee, a research program has been established in the Department of Chemical and Metallurgical Engineering to investigate DDC. The specific objective of the program, directed by Dr. C. F. Moore, is to develop improved DDC strategies and to experimentally evaluate these strategies using a PDP 15/35 digital computer interfaced to laboratory units designed to be representative of industrial control situations. The work described in this thesis is

part of this research program and deals with process modeling and controller development aspects of direct digital control loop design.

The DDC Loop Design Problem

Consider design of the DDC loop diagrammed in Figure 1(a). The continuous process output, x(t), is sampled to give a discrete-time signal, x_i. This is subtracted from a sampled set point signal, r_i, to produce a discrete error signal which can be used by the computer to calculate a control action, m_i, via some control algorithm. The discrete control action is then converted by a hold device to a continuous time signal, m(t), which is input to the process. Any disturbance, n(t), is considered to enter the process in addition with the control action.

The control loop design method of concern in this work is a three-step procedure consisting of:

(1) process modeling, in which a mathematical relation between u(t) and x(t) is found,

(2) controller specification, in which the form of the control algorithm is chosen, and

(3) controller tuning, in which values for the control algorithm parameters are determined such that the desired control loop performance is obtained. With conventional control algorithms, the advantage of process modeling is that the algorithm may be tuned off-line by simulation of the control loop, and modeling is not necessary if on-line tuning

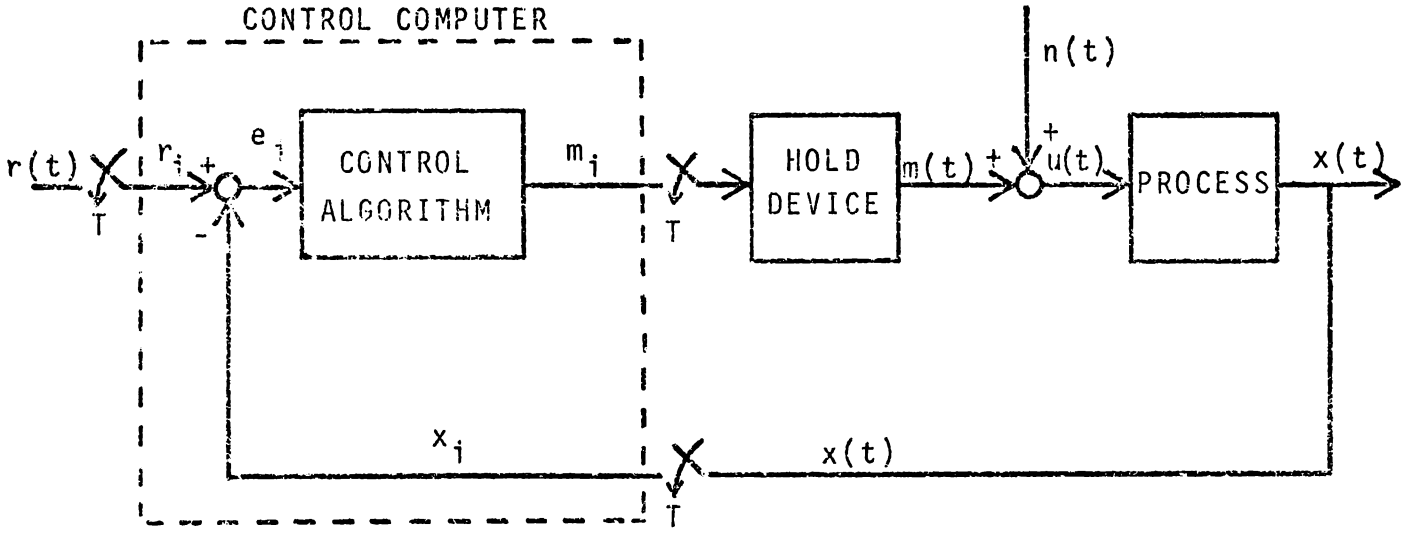


Figure 1(a),(b),(c). Direct Digital Control Loop

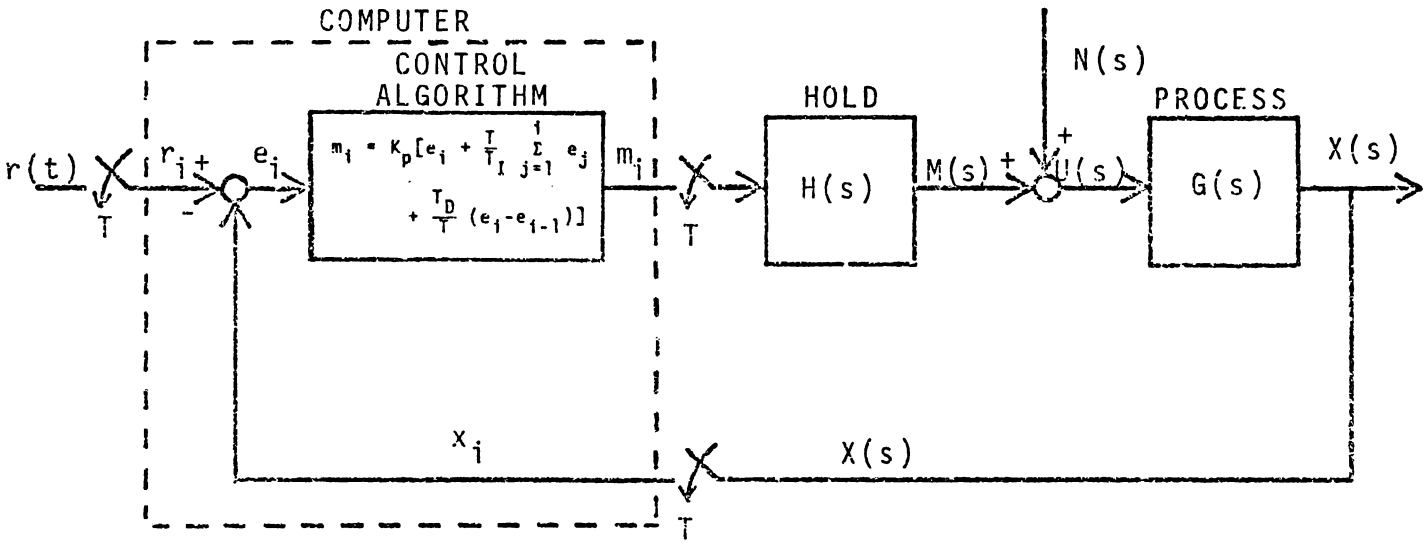


Figure 1(continued).

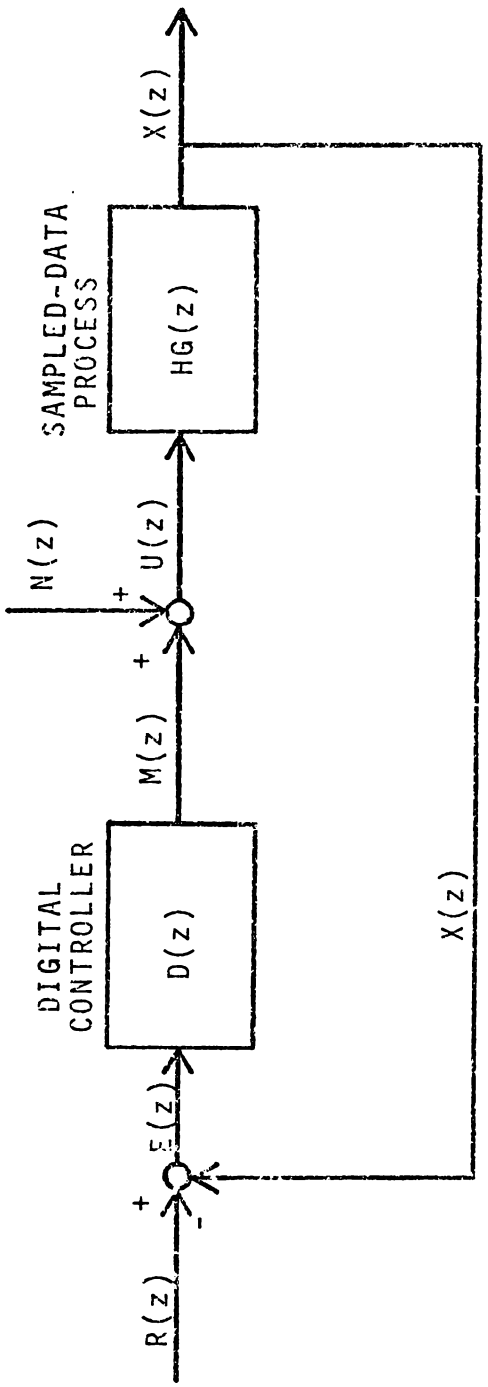


Figure 1(continued).

is practical. However, if advanced control techniques such as multivariable decoupling, feedforward control, or dead time compensation are to be implemented for improved control, some mathematical description of the process is required.

Conventional Design Approach

The conventional approach to digital control loop design in the chemical industries is to adapt continuous control techniques (see Figure 1(b)). For modeling purposes, the process dynamics are represented by a Laplace domain transfer function, and the most general model used to describe the process is an analytical solution to or a finite-difference representation of this transfer function. The fitting of process response data using the model is then performed to find values of the transfer function parameters which minimize a selected fit criterion (1,2).

As with process modeling, the general approach to controller specification for digital loops has relied heavily on continuous control concepts. The majority of digital control algorithms are simply numerical approximations of the analog one-, two-, and three-mode controllers. For example, the discrete PID controller is illustrated in Figure 1(b). It is well-known, however, that the performance of sampled-data control using discrete versions of analog modes decreases as sample time is increased. Thus, the engineer who employs these algorithms in digital systems must tend toward rapid sampling rates to achieve acceptable control which is,

nevertheless, limited to that attainable with continuous control.

A Discrete-Time Approach

Rather than considering the digital control loop as being comprised of both discrete and continuous elements, one can formulate the design problem entirely in terms of discrete-time components. If the samplers, the hold device, and the process are viewed collectively as a discrete-time system that reacts to a discrete input m_i to produce a discrete output x_i, z-transform methods can be used to derive a pulse transfer function, HG(z), relating the discrete inputs and outputs in the z-domain (3,4). Since disturbance design generally assumes that the load change is a step function beginning at a sample time, the disturbance may also be treated as a discrete-time function which enters the hold device with the controller output. Finally, the computer control algorithm can be considered a digital controller with a z-domain transfer function, D(z), and the control loop can be represented by the sampled-data control loop diagrammed in Figure 1(c). This discrete-time approach to direct digital control applications offers definite advantages over conventional techniques in the process modeling and controller specification phases of control loop design.

Inversion of the pulse transfer function gives a discrete-time model which is linear in all the model parameters except dead time. Thus, a least-squares fit of process

response data can conceivably be performed using a one-dimensional search for dead time with the remaining model parameters calculated by linear regression at each iteration. This constitutes a general, efficient modeling scheme for chemical process modeling.

The z-transform design of digital controllers is a well-developed subject of sampled-data control theory (4,5). Essentially, for a given pulse transfer function, block diagram algebra is used with Figure 1(c) to find the controller transfer function required to produce specified control loop characteristics. The relevance of such techniques to the control of chemical processes arises from the fact that controller design automatically includes compensation for any process dead time, a variable which is notoriously detrimental to the performance of conventional controllers.

Organization

The objective of this thesis was to investigate the advantages offered by a z-transform approach to process modeling and controller design. Chapter II describes a discrete-time modeling procedure for fitting process response data to a general second-order model; the listing of a Fortran IV modeling program based on these developments is given in Appendix A. Chapter III outlines the z-transform design of control algorithms and the development of an improved DDC algorithm; controller simulation results are presented, with a listing of the simulation program in

Appendix B. Final conclusions and recommendations are summarized in Chapter IV.

CHAPTER II

DISCRETE-TIME MODELING

Analytical process models are usually too complex to be of any practical use in chemical process control. The general approach in the chemical industries is to propose a simple model with adjustable parameters and to empirically determine the parameter values which, in some sense, "best" describe the process dynamics. One approximation which has been used for years is the first-order lag plus dead time model. The graphical methods presented by Ziegler and Nichols (6) and Miller (7) can be used when a continuous step response plot is available. For sampled process response data, an analytical solution of the first-order lag differential equation can be used for a least-squares fit of the data.

While the first-order model is attractive from the standpoint of simplicity, a much improved representation of the higher-order chemical process response may be obtained with a second-order model. Smith (3) simulated the continuous PID control of a fourth-order process and compared the set point and load change responses for an optimally tuned controller with those for controllers tuned using first- and second-order process models. The tuning criterion was ITAE. The possible improvement over the first-order tuned controller was nearly 50 percent for load changes and over 100 percent for set point changes, while the possible improvement

for the controller tuned using the second-order model was only 12 percent for load changes and less than one percent for set point changes. In a similar study, Chiu (8) simulated the temperature control of a jacketed backmix reactor and found that tuning PI and PID controllers using a second-order model gave controller parameters which were nearly optimal. Again, substantial improvements over first-order tuning were observed.

As with the first-order lag model, graphical methods are available for evaluating the parameters of a second-order lag plus dead time model from a step response plot. Sten's technique (9) is applicable only with an overdamped model, and the need for a graphical tangent can introduce considerable error. The method by Meyer (10) can be used for both overdamped and underdamped models but has the disadvantage that the dynamic parameters are based upon only two points.

A more general, objective approach which removes the restriction of a step input is to develop a finite-difference representation of the second-order differential equation. A least-squares fit of sampled process input/output data may then be performed using a one-dimensional search to find the model dead time with the remaining model parameters calculated by linear regression (1). This type of numerical procedure is also useful in adaptive control applications in which on-line model updating is required. However, because of the finite-difference approximations, such a model is strictly

limited to short sample times.

One common feature of these modeling methods is that the basis of the model is a Laplace domain transfer function, and the objective of the modeling is to find values for the transfer function gain, dead time, and time constants. For direct digital control applications, an approach more consistent with the discrete-time nature of the computer would be to consider the hold/process cascade as a discrete-time system and develop a modeling procedure with a z-transform pulse transfer function as its basis.

Discrete-Time Model Development

While almost all chemical engineers are aware of the Laplace transform and its usefulness in the analysis of continuous systems, a far smaller number are as familiar with the z-transform and its applications to discrete-time, or sampled-data, systems analysis. The z-transform of the discrete-time signal obtained by sampling the continuous signal f(t) with a sample time T can be denoted as F(z) and is defined as

$$F(z) = Z[f(kT)] = \sum_{k=0}^{\infty} f(kT)z^{-k}$$

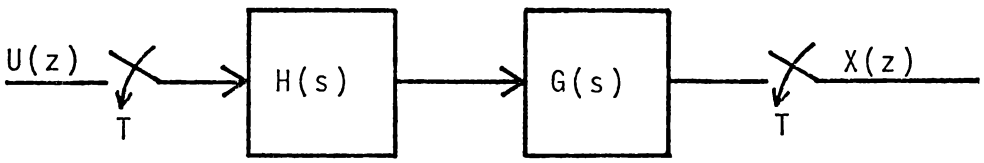
This equation results from representing the sampler output as the product of the continuous signal and a unit impulse train of period T. Taking the Laplace transform of this signal/impulse train representation, and making the change

of variable $z = e^{sT}$ yields the above relation (5). Therefore, the z-transform is essentially the Laplace transform of the sampler output. F(z) is often written as

$$F(z) = Z[F(s)]$$

which should be interpreted as the z-transform of the discrete-time signal obtained by sampling a continuous signal which has a Laplace transform F(s).

In terms of the figure below, the z-transform expression relating X(z), the sampled process output,



to the discrete-time input, U(z), is termed the pulse transfer function HG(z):

$$HG(z) = Z[H(s)G(s)]$$

Just as the Laplace transfer function for a continuous system is the Laplace transform of the system's impulse response, the pulse transfer function is the z-transform of the discrete-time system's response to a Kronecker delta input, $\delta(kT)$, defined by the relations

$$\delta(kT) = \begin{cases} 1 \text{ for } k=0 \\ 0 \text{ for } k\neq 0 \end{cases}$$

Tables listing the z-transform transfer function corresponding to various Laplace transfer functions are generally

included in any text that deals with sampled-data systems (3,4,5). These tables can be used to find the pulse transfer function for a given hold and process. For example, with a zero-order hold and a second-order lag plus dead time process,

$$H(s) = \frac{1-e^{-sT}}{s}$$

$$G(s) = \frac{Ke^{-\theta s}}{(t_1 s+1)(t_2 s+1)}$$

and

$$HG(z) = Z\left[\frac{(1-e^{-sT})Ke^{-\theta s}}{s(t_1 s+1)(t_2 s+1)}\right] \quad .$$

The z-transform of a time delay of m sample times is z^{-m}, and we have

$$HG(z) = z^{-N}(1-z^{-1})Z\left[\frac{K}{s(t_1 s+1)(t_2 s+1)}\right]$$

where $N = \theta/T$, assumed to be an integer. A partial fractions expansion of the expression in brackets yields

$$HG(z) = K\, z^{-N}(1-z^{-1})\left\{Z\left[\frac{1}{s}\right] + Z\left[\frac{t_1^2}{(t_2-t_1)(t_1 s+1)}\right] + Z\left[\frac{t_2^2}{(t_1-t_2)(t_2 s+1)}\right]\right\}$$

From z-transform tables,

$$Z\left[\frac{1}{s}\right] = \frac{1}{1-z^{-1}}$$

$$Z\left[\frac{t_1^2}{(t_2-t_1)(t_1 s+1)}\right] = \frac{t_1}{(t_2-t_1)(1-\alpha_1 z^{-1})}$$

and

$$Z\left[\frac{t_2^2}{(t_1-t_2)(t_2s+1)}\right] = \frac{t_2}{(t_1-t_2)(1-\alpha_2 z^{-1})}$$

where

$$\alpha_1 = e^{-T/t_1} \quad \text{and} \quad \alpha_2 = e^{-T/t_2} .$$

Substitution of these relations gives the pulse transfer function

$$HG(z) = \frac{z^{-N}(b_1 z^{-1}+b_2 z^{-2})}{1+a_1 z^{-1}+a_2 z^{-2}} \tag{2}$$

The coefficients a_1, a_2, b_1, and b_2 are functions of the process time constants, process gain, and the sample time. Cross-multiplication of equation (2) yields

$$[1+a_1 z^{-1}+a_2 z^{-2}]\, X(z) = z^{-N}\, [b_1 z^{-1}+b_2 z^{-2}]\, U(z)$$

and employing the right-shift property of the z-transform, which states that

$$Z^{-1}\, [z^{-m} X(z)] = x_{i-m}$$

we obtain the time-domain expression

$$x_i = -a_1 x_{i-1} - a_2 x_{i-2} + b_1 u_{i-N-1} + b_2 u_{i-N-2} \tag{3}$$

This equation is a second-order difference equation which relates the process output at sample i to past input and output samples.

Equation (3) is a very general expression in that it is valid for a number of first- and second-order processes of interest in chemical process modeling. In Table I are

TABLE I

DIFFERENCE EQUATION COEFFICIENTS

Process Transfer Function, G(s)	a_1	a_2	b_1	b_2
$\frac{Ke^{-\theta s}}{t_1 s+1}$	$-\alpha_1$	0	$K(1-\alpha_1)$	0
$\frac{Ke^{-\theta s}}{(t_1 s+1)^2}$	$-2\alpha_1$	α_1^2	$K[1-\alpha_1(1+\frac{T}{t_1})]$	$K[\alpha_1(\alpha_1-1+\frac{T}{t_1})]$
$\frac{Ke^{-\theta s}(t_3 s+1)}{(t_1 s+1)^2}$	$-2\alpha_1$	α_1^2	$K[1-\alpha_1(1+\frac{T}{t_1}-\frac{t_3 T}{t_1^2})]$	$K[\alpha_1(\alpha_1-1+\frac{T}{t_1}-\frac{t_3 T}{t_1^2})]$
$\frac{Ke^{-\theta s}}{(t_1 s+1)(t_2 s+1)}$	$-(\alpha_1+\alpha_2)$	$\alpha_1\alpha_2$	$K[1+\frac{t_1\alpha_1-t_2\alpha_2}{t_2-t_1}]$	$K[\alpha_1\alpha_2+\frac{t_1\alpha_2-t_2\alpha_1}{t_2-t_1}]$
$\frac{Ke^{-\theta s}(t_3 s+1)}{(t_1 s+1)(t_2 s+1)}$	$-(\alpha_1+\alpha_2)$	$\alpha_1\alpha_2$	$K[1+\frac{t_1\alpha_1-t_2\alpha_2+t_3(\alpha_2-\alpha_1)}{t_2-t_1}]$	$K[\alpha_1\alpha_2+\frac{t_1\alpha_2-t_2\alpha_1+t_3(\alpha_1-\alpha_2)}{t_2-t_1}]$

Note: $N = \theta/T$, $\alpha_1 = e^{-T/t_1}$, $\alpha_2 = e^{-T/t_2}$

$$y(I) = -a_1 y(I-1) - a_2 y(I-2) + b_1 u(I-1-N) + b_2 u(I-2-N)$$

presented the relations between the difference equation coefficients and the parameters of the corresponding Laplace transfer function. The process dead time is assumed to be an integer multiple of the sample time, and the hold device, as in almost all DDC applications, is a zero-order hold.

To use equation (3) as a least-squares model, the actual output x_i is replaced by the model output, $\hat{x}_i$. For the general case in which the process output is non-zero for a zero input, an offset term D must be added, and the model equation becomes

$$\hat{x}_i = -a_1 x_{i-1} - a_2 x_{i-2} + b_1 u_{i-N-1} + b_2 u_{i-N-2} + D. \quad (4)$$

The model estimate of the process offset, x_o, can be calculated from the model parameters using the relation

$$x_o = \frac{D}{1+a_1+a_2} \quad .$$

If process response data are to be fit with a first-order model ($a_2=b_2=0$), then this discrete model offers no advantages. In fact, an analytical solution of the first-order lag differential equation gives the same model. However, in fitting a more descriptive second-order model to process data, the model of equation (4) offers several definite advantages. Since the pulse transfer function is an exact relation, the model is not limited to short sample times as are models derived using finite-differences. This is an important consideration if the model is to be used in

conjunction with a control algorithm that will allow increased sample times. Also, the same second-order model equation can be used whether the process exhibits overdamped, underdamped, or inverse response characteristics. Thus, one general equation can be used to model several types of systems with no restrictions on the type of process data. The model may be applied to closed-loop operating data as well as to open-loop response data. Finally, the linearity of the model in all parameters except dead time allows linear regression to be used with a search for dead time. This is, of course, an important computational advantage.

Kalman (11) suggested the use of this model, without dead time or the offset term, as the basis of an adaptive control system which could follow the changes in process parameters by on-line modeling. Dahlin (12) employed a first-order version of the model in the identification of paper machine dynamics. However, in the modeling of chemical processes, this model has been generally ignored. Apparently, one reason for this arises from the conventional approach of modeling in terms of Laplace transfer function parameters and from the fact that, at first glance, solving for the gain and two time constants of the important second-order lag, given the four discrete-time model parameters, would not appear to yield a unique solution. For example, Gallier and Otto (2) outlined the possible use of the model in an on-line updating of second-order lag parameters for adaptive DDC of chemical processes. Instead of taking advantage of the linear

regression calculation of the model coefficients, they chose to perform a numerical search for the time constants, gain, and dead time which minimized the least-squares fit criterion.

Consider relating the discrete model coefficients to the second-order lag parameters as follows. From Table I, we have the four equations shown below and three unknowns--t_1, t_2, and K.

$$a_1 = -(\alpha_1+\alpha_2)$$

$$a_2 = \alpha_1\alpha_2$$

$$b_1 = K\left[1 + \frac{t_1\alpha_1 - t_2\alpha_2}{t_2 - t_1}\right]$$

$$b_2 = K\left[\alpha_1\alpha_2 + \frac{t_1\alpha_2 - t_2\alpha_1}{t_2 - t_1}\right]$$

Another relation may be obtained from equation (3) giving the process gain as

$$K = \frac{b_1+b_2}{1+a_1+a_2} \quad .$$

But adding together the above expressions for b_1 and b_2 gives the same gain equation and, therefore, there are actually only three independent equations--those for a_1 and a_2 and the process gain relation. These three equations are easily solved to give t_1, t_2, and K. Therefore, if values for the second-order lag transfer function parameters are to be the objective of a modeling effort, the discrete-time model can still be used and the continuous parameters back-calculated from the model coefficients.

Formulation of the Regression

With the model equation given by equation (4), the objective is to find those values of a_1, a_2, b_1, b_2, D, and N which give the best fit of the process response data. For a least-squares regression, the fit criterion E is the sum of the squared residuals, or fit errors. That is,

$$E = \sum_{i=N+3}^{M} (x_i - \hat{x}_i)^2 \qquad (5)$$

where M is the number of data points. Note that the need for greater-than-zero subscripts on the u values requires that the error summation begin with i=N+3. The total number, m, of model output values considered in the fit criterion is then

$$m = M-N-2$$

which, for a fixed number of data points, decreases with increasing N.

Substitution of the model equation gives

$$E = \sum_{i=N+3}^{M} (x_i + a_1 x_{i-1} + a_2 x_{i-2} - b_1 u_{i-N-1} - b_2 u_{i-N-2} - D)^2 \qquad (6)$$

The best values for the model coefficients are those which minimize E. Thus, we require

$$\frac{\partial E}{\partial a_1} = 0 = 2 \sum_{i=N+3}^{M} x_{i-1}(x_i + a_1 x_{i-1} + a_2 x_{i-2} - b_1 u_{i-N-1} - b_2 u_{i-N-2} - D)$$

$$\frac{\partial E}{\partial a_2} = 0 = 2 \sum_{i=N+3}^{M} x_{i-2}(x_i+a_1x_{i-1}+a_2x_{i-2}-b_1u_{i-N-1}-b_2u_{i-N-2}-D)$$

$$\frac{\partial E}{\partial b_1} = 0 = -2 \sum_{i=N+3}^{M} u_{i-N-1}(x_i+a_1x_{i-1}+a_2x_{i-2}-b_1u_{i-N-1} -b_2u_{i-N-2}-D)$$

$$\frac{\partial E}{\partial b_2} = 0 = -2 \sum_{i=N+3}^{M} u_{i-N-2}(x_i+a_1x_{i-1}+a_2x_{i-2}-b_1u_{i-N-1} -b_2u_{i-N-2}-D)$$

$$\frac{\partial E}{\partial D} = 0 = -2 \sum_{i=N+3}^{M} (x_i+a_1x_{i-1}+a_2x_{i-2}-b_1u_{i-N-1}-b_2u_{i-N-2}-D)$$

which upon rearrangement gives

$$\begin{aligned}
&-a_1 \sum_{i=N+3}^{M} x_{i-1}^2 - a_2 \sum x_{i-2}x_{i-1} + b_1 \sum u_{i-N-1}x_{i-1} + b_2 \sum u_{i-N-2}x_{i-1} \\
&\qquad + D \sum x_{i-1} = \sum x_i x_{i-1} \\
&-a_1 \sum_{i=N+3}^{M} x_{i-1}x_{i-2} - a_2 \sum x_{i-2}^2 + b_1 \sum u_{i-N-1}x_{i-2} + b_2 \sum u_{i-N-2}x_{i-2} \\
&\qquad + D \sum x_{i-2} = \sum x_i x_{i-2} \qquad (7) \\
&-a_1 \sum_{i=N+3}^{M} x_{i-1}u_{i-N-1} - a_2 \sum x_{i-2}u_{i-N-1} + b_1 \sum u_{i-N-1}^2 + b_2 \sum u_{i-N-2}u_{i-N-1} \\
&\qquad + D \sum u_{i-N-1} = \sum x_i u_{i-N-1}
\end{aligned}$$

$$-a_1 \sum_{i=N+3}^{M} x_{i-1}u_{i-N-2} - a_2 \Sigma\, x_{i-2}u_{i-N-2} + b_1 \Sigma\, u_{i-N-1}u_{i-N-2}$$

$$+ b_2 \Sigma\, u_{i-N-2}^2 + D \Sigma\, u_{i-N-2} = \Sigma\, x_i u_{i-N-2}$$

$$-a_1 \sum_{i=N+3}^{M} x_{i-1} - a_2 \Sigma\, x_{i-2} + b_1 \Sigma\, u_{i-N-1} + b_2 \Sigma\, u_{i-N-2} + Dm = \Sigma\, x_i$$

All unlabeled summations are, of course, implied to sum from $i=N+3$ to M.

These equations are termed the normal equations for the regression, and the solution of this set of equations for a specific value of N gives the model coefficients which minimize the fit criterion for that particular choice of dead time. Therefore, a least-squares fit of the data can be performed by a search on N with the best values of the model coefficients for each iteration determined by solution of equations (7). That value of N which results in the least sum of squared residuals, as calculated by equation (5), and the corresponding values of the model coefficients give a least-squares fit of the experimental data.

Note the similarity of terms in the normal equations. For example, with step response data, one would not expect the elements of the first two equations to be very different; in other words, the $\Sigma\, x_{i-1}^2$ term would not be much different from the $\Sigma\, x_{i-1}x_{i-2}$ term, etc. Likewise, the third and fourth equations would not be expected to differ drastically. In such cases, the first and second equations and the third

and fourth equations would be nearly linearly dependent, and the numerical problems associated with such ill-conditioned systems could be expected to arise. In computer solutions of the equations, this was found to be the case, with changes in the sixth significant figure of the summation terms producing changes in the second significant figure of the solutions. Also, double precision computations were required to obtain correct solutions.

There are a couple of factors which must be considered in choosing the search technique to be used in finding the optimum dead time.

Note that the magnitude of the fit criterion of equation (5) is influenced by the number of residuals as well as the magnitude of those residuals. As was pointed out, the number of residuals decreases as the model dead time is increased; thus, the fit criterion could conceivably be minimized by a poor fit which utilizes a large estimate of dead time. Since the objective of the modeling procedure is to minimize the fit criterion by minimizing only the magnitude of the residuals, it follows that the search should be limited to relatively low values of dead time. In addition, minimizing the mean squared error would tend to offset the effect of the number of errors considered.

Also, large dead time estimates in the modeling of step response data can lead to results which are physically meaningless and useless for control purposes. Consider the sampled response of a second-order system to a unit step

input which enters at sample k. The response is, of course, governed by equation (3). However, for $i \geq k+N+2$, the discrete inputs, u_{i-N-1} and u_{i-N-2}, are constant and equal to one, and the response is given as

$$x_i = -a_1 x_{i-1} - a_2 x_{i-2} + D' \tag{8}$$

where

$$D' = b_1 + b_2 + D. \tag{9}$$

If the data are modeled using an estimate of dead time, N', such that

$$N' \geq N + k - 1 \tag{10}$$

Then all the data points considered in the error calculations can be represented by equation (8), and the least-squares fit gives an input-independent model which fits the data as well as equation (3). In fact, since the N' of equation (10) is larger than the actual dead time, the number of residuals summed in the fit error for N' would be less than those for N, resulting in a lower value of the fit criterion for N'.

A definite example vividly illustrates the above. The discrete model parameters shown below were used to generate unit step response data.

$$\begin{aligned} N &= 5 & D &= 0.01228 \\ a_1 &= -1.84034 & b_1 &= 0.003154 \\ a_2 &= 0.84648 & b_2 &= 0.002983 \end{aligned} \tag{11}$$

These parameters correspond to sampling the continuous process

$$G(s) = \frac{e^{-5s}}{(10s+1)(15s+1)}$$

with a sample time of one and an offset of 2.0. The input and calculated process output are shown in Figure 2. Note that the step input entered at the eighth sample. For estimates of N, equations (7) were solved on the IBM 360 computer by pivotal condensation (13) to find the least-squares model coefficients for that dead time estimate, and the least-squares fit criterion and mean squared error were then calculated. The regression equations were sufficiently ill-conditioned to require double precision calculations in their solution. Figure figure 3 is a plot of the fit error as a function of the model dead time. Beginning at a model dead time of zero, the fit error decreases to a definite minimum at the process dead time of 5. The error then rises but drops drastically again at a model dead time of 12, as expected from equation (10), and continues to decrease for higher values of N'. The fits for dead times of 12 and higher were characterized by zero values for b_1 and b_2, while the bias estimate obeyed equation (9). Figure 4 shows that the mean squared error behaved in the same general manner, and minimizing this parameter would offer no improvement.

This procedure was repeated with simulated set point change response data for the process of equation (11) under discrete PI control. Input and output data points are shown

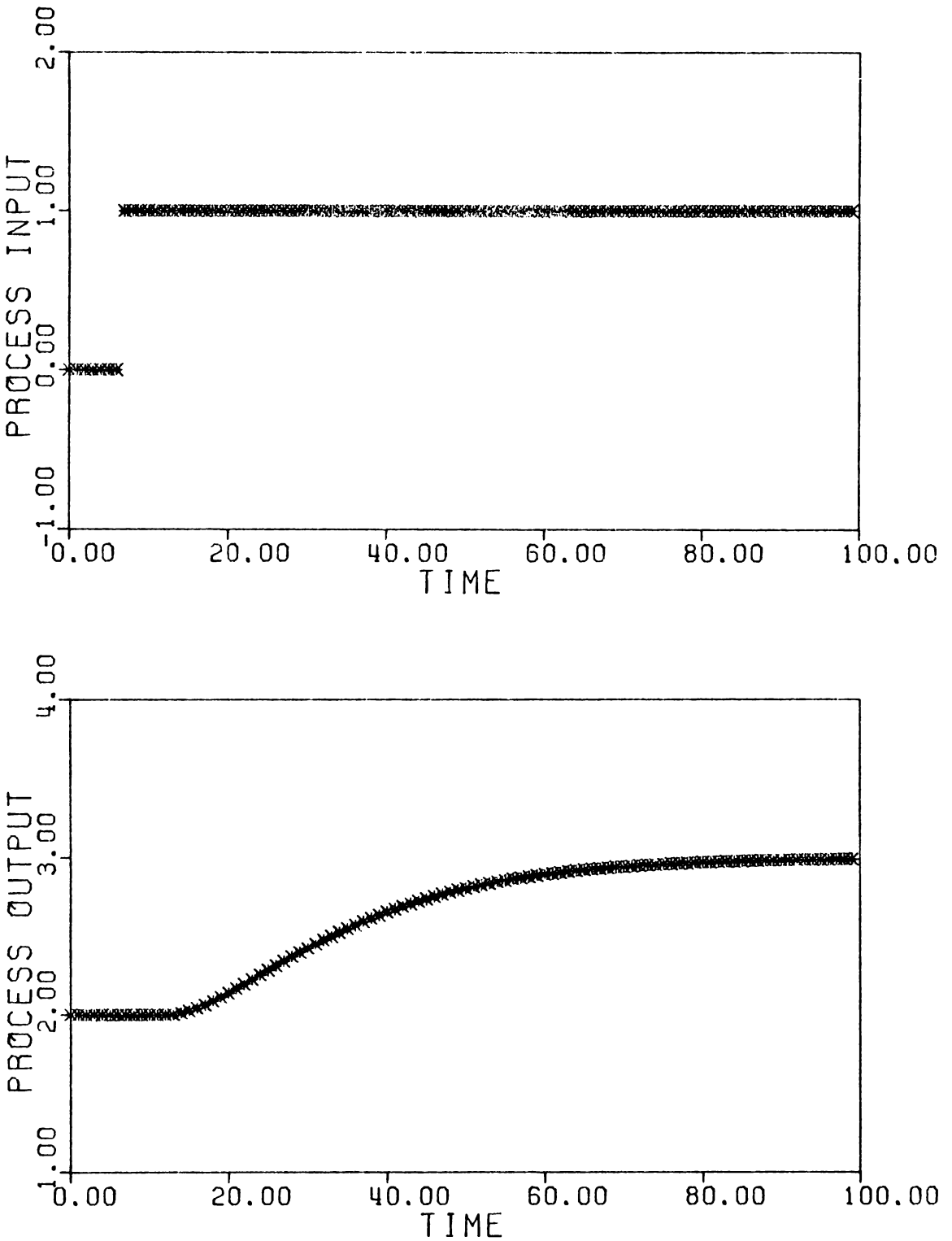


Figure 2. Step Response Data for Simulated Second-Order Process

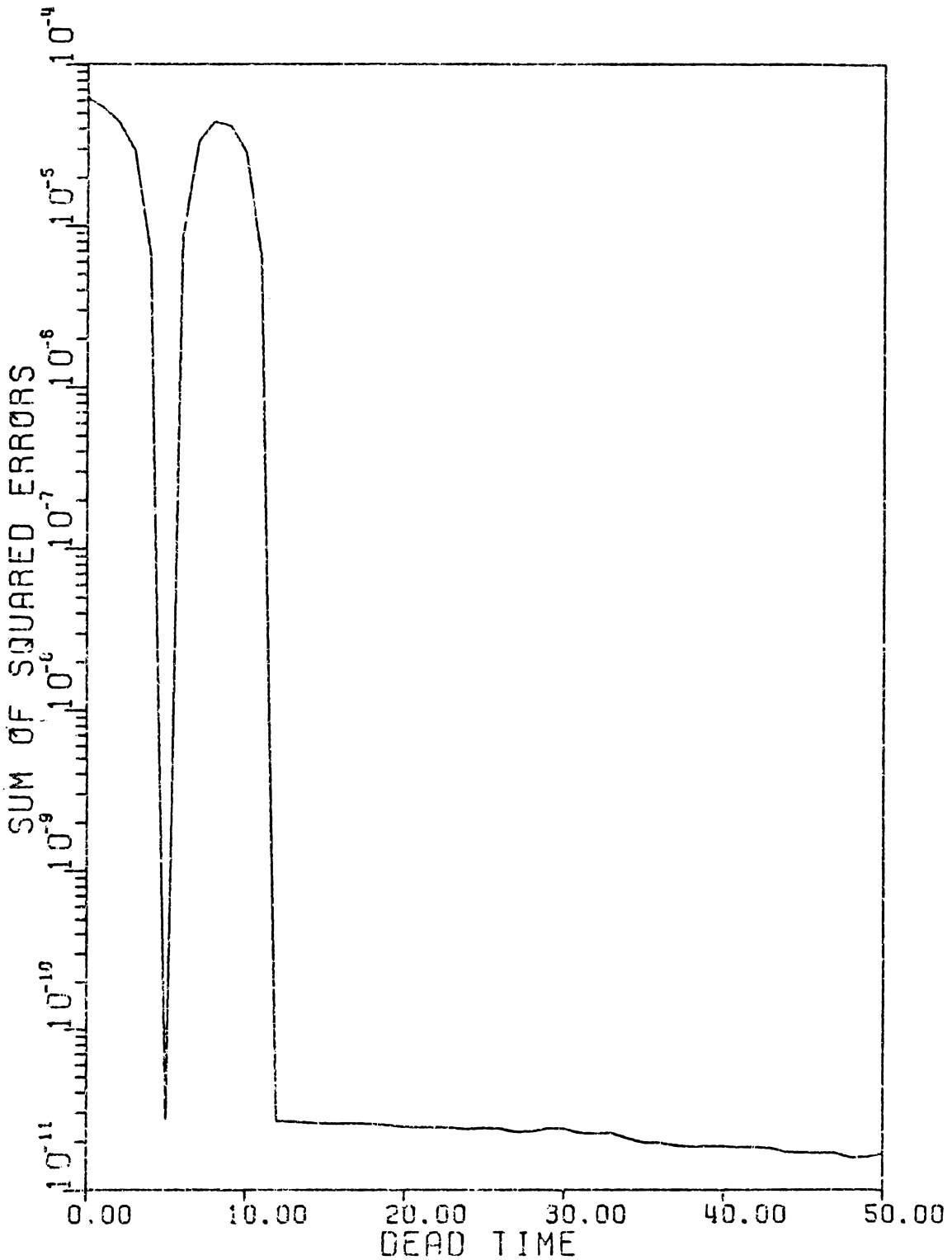


Figure 3. Least-Squares Fit Criterion as a Function of Model Dead Time for Simulated Step Response Data

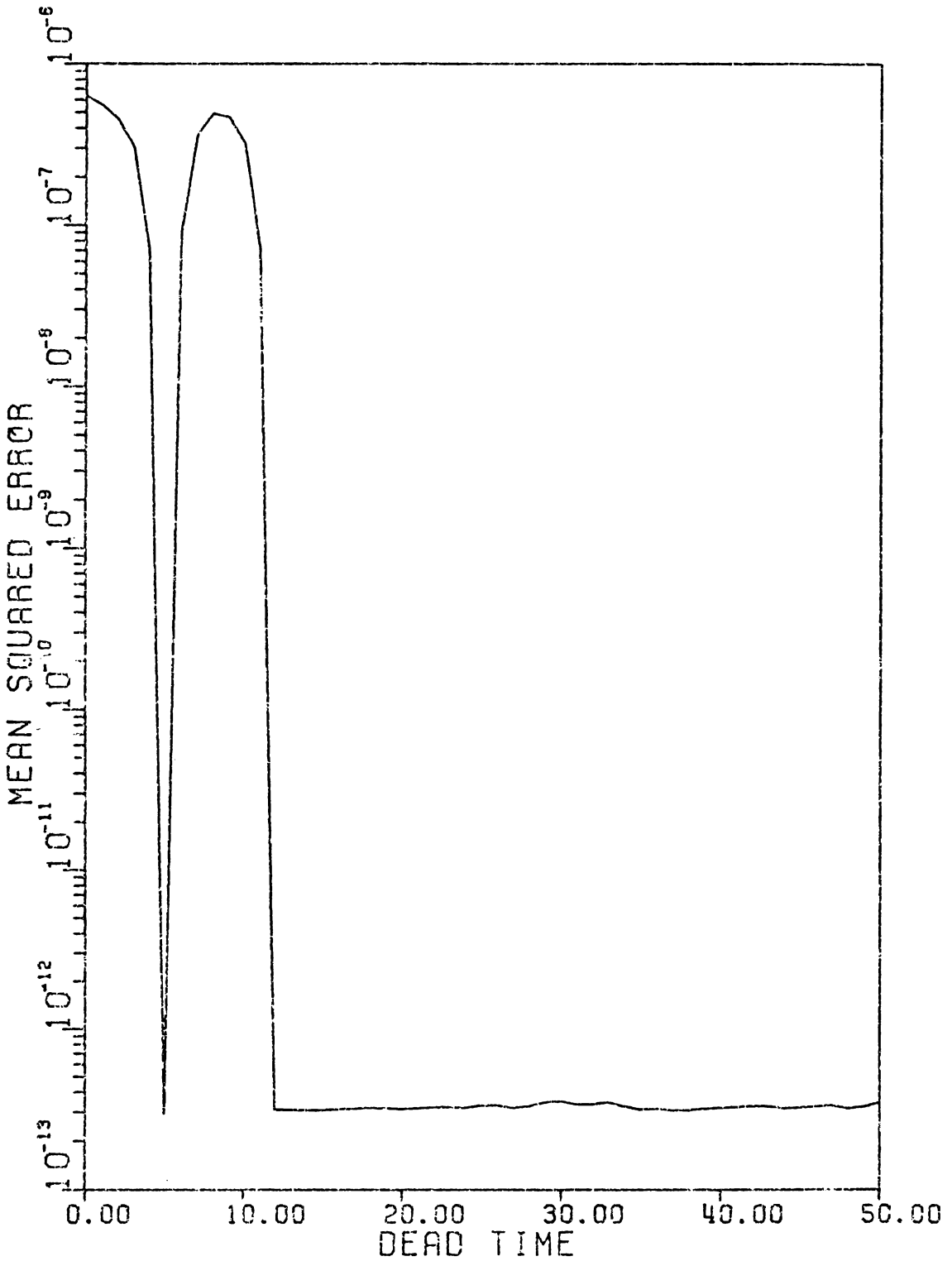


Figure 4. Mean Squared Error as a Function of Model Dead Time for Simulated Step Response Data

in Figure 5. It was found that for model dead times greater than 11, the least squares regression treated the overshoot response as the step response of an underdamped system, and the fit error behaved in the same manner as that of the step response data, with an input-independent model ($b_1=b_2=0$).

In light of the above, it was concluded that the best dead time search procedure for the least-squares modeling is to begin with an estimate of zero and increment N' by one until there is no improvement in the fit criterion. This ensures that a low value for the fit error does not arise because only a few data points are considered in its calculation. Such a cautious search also prevents the dead time estimate from becoming so large that process response data can be described by an input-independent model. Note that the maximum allowable dead time estimate is N'=M-7 since for model dead times greater than this, less than 5 data points are fit and the linear regression has no unique solution.

In Figure 6 is shown a flow diagram of this least-squares modeling scheme. The main program serves to input the process response data and execute the dead time search. For a given model dead time, the subroutine ASUBI sets up the normal equations for the linear regression, solves the equations for the model coefficients, and computes the fit error. Because the linear regression must be performed for each dead time iteration, the bulk of the program computations occur in the linear regression subroutine. It is desirable, therefore, that the calculations required to form

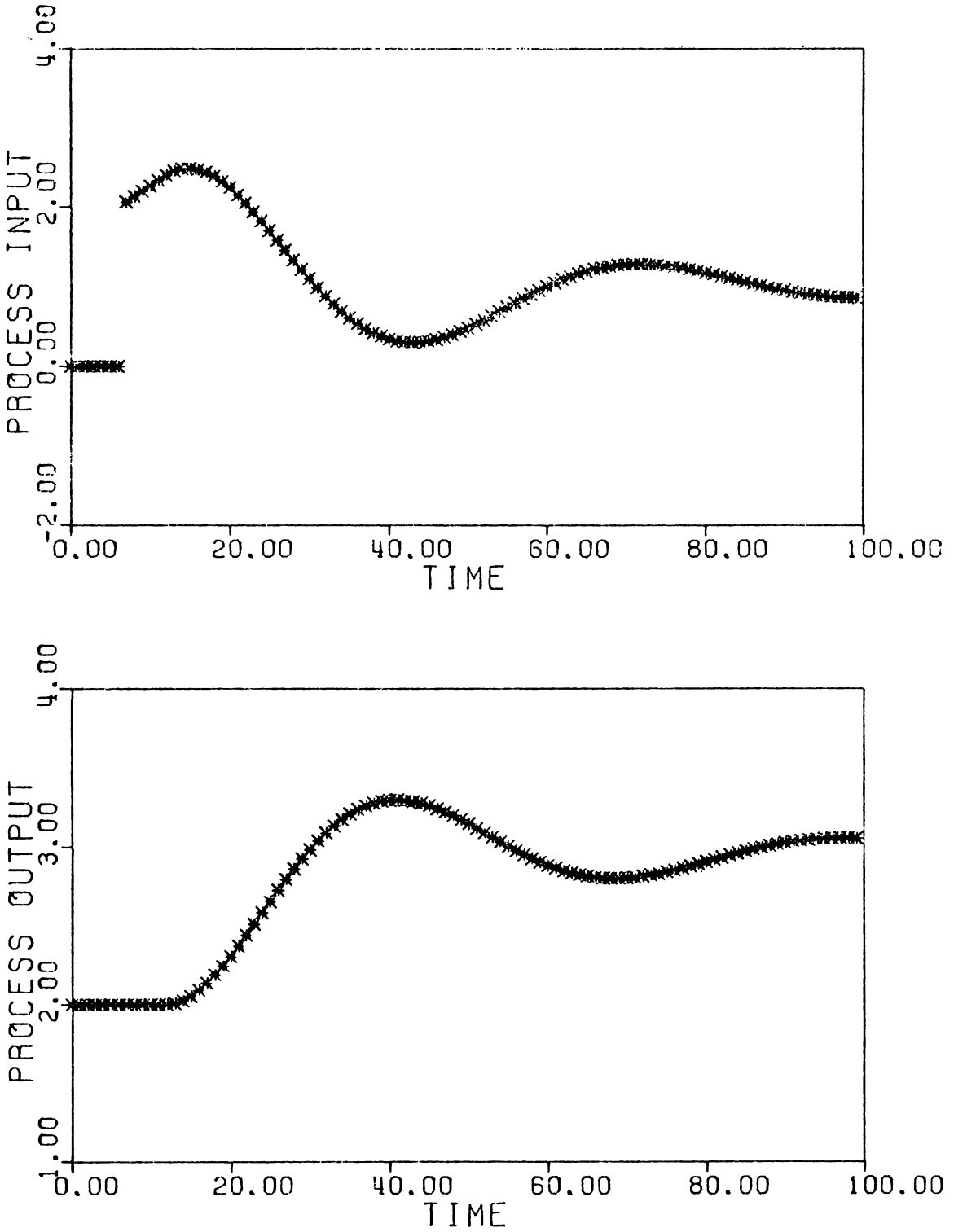
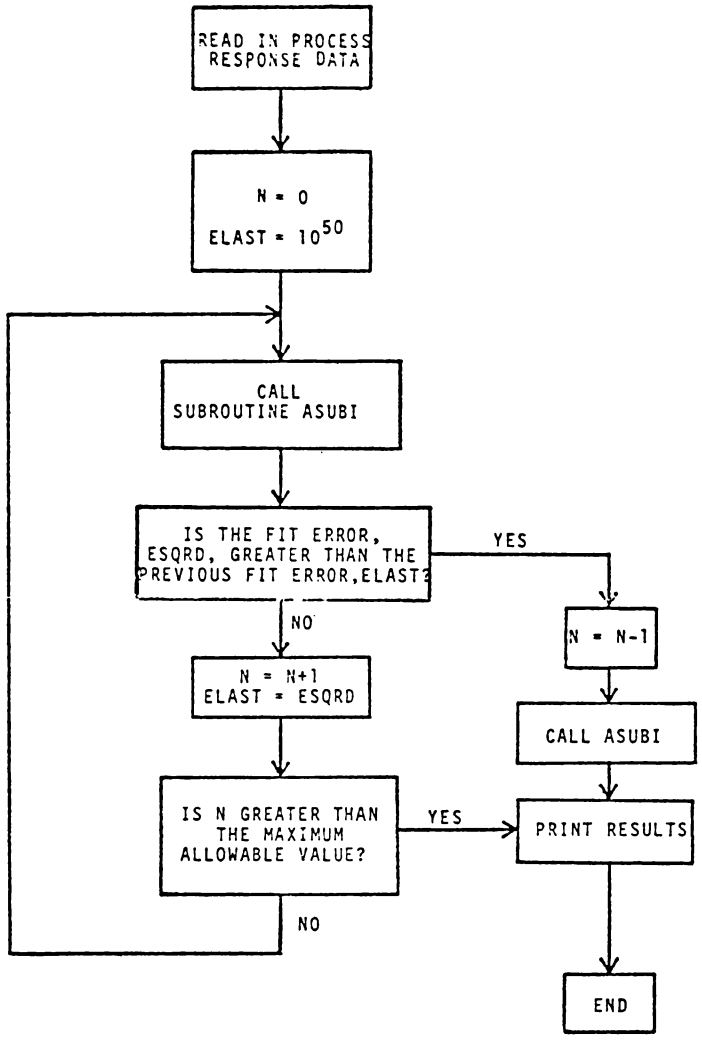
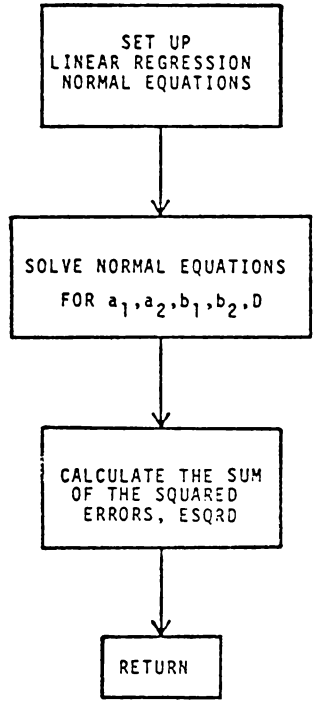


Figure 5. Simulated Set Point Response Data



(a) Main Program



(b) Linear Regression Subroutine, ASUBI

Figure 6. Flow Diagram of Least-Squares Modeling Scheme

the summation terms for the linear regression be a minimum.

Reduction of Computations

The linear regression equations (7) show that 29 summation terms must be evaluated for each iteration of dead time. The computation time required to form these summations can be considerably reduced by noting several relationships among the terms.

The first obvious simplification results from noting that 10 of the summation terms appear twice in the equations. Therefore, only the evaluation of 19 summations is actually necessary. There also exist the following relations among these 19 terms.

$$\sum_{i=N+3}^{M} x_{i-1} = x_{N+2} - x_M + \sum_{i=N+3}^{M} x_i$$

$$\sum_{i=N+3}^{M} x_{i-2} = x_{N+1} - x_{M-1} + \sum_{i=N+3}^{M} x_{i-1}$$

$$\sum_{i=N+3}^{M} x_{i-2}^2 = x_{N+1}^2 - x_{M-1}^2 + \sum_{i=N+3}^{M} x_{i-1}^2$$

$$\sum_{i=N+3}^{M} x_i x_{i-1} = x_M x_{M-1} - x_{N+2} x_{N+1} + \sum_{i=N+3}^{M} x_{i-1} x_{i-2}$$

$$\sum_{i=N+3}^{M} u_{i-N-2} = u_1 - u_{M-N-1} + \sum_{i=N+3}^{M} u_{i-N-1}$$

$$\sum_{i=N+3}^{M} u_{i-N-2}^2 = u_1^2 - u_{M-N-1}^2 + \sum_{i=N+3}^{M} u_{i-N-1}^2$$

$$\sum_{i=N+3}^{M} x_i u_{i-N-1} = x_M u_{M-N-1} - u_1 x_{N+2} + \sum_{i=N+3}^{M} u_{i-N-2} x_{i-1}$$

$$\sum_{i=N+3}^{M} u_{i-N-2} x_{i-2} = u_1 x_{N+1} - u_{M-N-1} x_{M-1} + \sum_{i=N+3}^{M} u_{i-N-1} x_{i-1}$$

These relationships show that 8 of the 19 summation terms necessary for the linear regression may be found by simple additions to other terms.

The computation time necessary to set up the regression equations can be further reduced since for two values of N, N_1 and N_2, such that $N_2 = N_1+1$, the following relations are valid.

$$\sum_{i=N_2+3}^{M} x_{i-1}^2 = \sum_{i=N_1+3}^{M} x_{i-1}^2 - x_{N_1+2}^2$$

$$\sum_{i=N_2+3}^{M} x_{i-2} x_{i-1} = \sum_{i=N_1+3}^{M} x_{i-2} x_{i-1} - x_{N_1+1} x_{N_1+2}$$

$$\sum_{i=N_2+3}^{M} u_{i-N_2-1}x_{i-1} = \sum_{i=N_1+3}^{M} u_{i-N_1-2}x_{i-1} - u_1 x_{N_1+2}$$

$$\sum_{i=N_2+3}^{M} u_{i-N_2-1}x_{i-2} = \sum_{i=N_1+3}^{M} u_{i-N_1-2}x_{i-2} - u_1 x_{N_1+1}$$

$$\sum_{i=N_2+3}^{M} u^2_{i-N_2-1} = \sum_{i=N_1+3}^{M} u^2_{i-N_1-1} - u^2_{M-N_1-1}$$

$$\sum_{i=N_2+3}^{M} u_{i-N_2-2}x_{i-1} = \sum_{i=N_1+3}^{M} u_{i-N_1-2}x_{i} - u_{M-N_1-2}x_{M}$$

$$\sum_{i=N_2+3}^{M} u_{i-N_2-2}u_{i-N_2-1} = \sum_{i=N_1+3}^{M} u_{i-N_1-2}u_{i-N_1-1} - u_{M-N_1-2}u_{M-N_1-1}$$

$$\sum_{i=N_2+3}^{M} u_{i-N_2-1} = \sum_{i=N_1+3}^{M} u_{i-N_1-1} - u_{M-N_1-1}$$

$$\sum_{i=N_2+3}^{M} x_i x_{i-2} = \sum_{i=N_1+3}^{M} x_i x_{i-2} - x_{N_1+3}x_{N_1+1}$$

$$\sum_{i=N_2+3}^{M} x_i = \sum_{i=N_1+3}^{M} x_i - x_{N_1+3}$$

Thus, for two successive iterations of dead time, 10 of the 11 summations required for the second iteration can be efficiently calculated from those required for the preceeding iteration. This leaves only one summation term to be completely evaluated, once the dead time search is initiated.

A reduction in the time required to evaluate the fit error may also be realized. Recall the fit criterion of equation (6):

$$E = \sum_{i=N+3}^{M} (x_i + a_1 x_{i-1} + a_2 x_{i-2} - b_1 u_{i-N-1} - b_2 u_{i-N-2} - D)^2$$

Expansion of the summation argument gives

$$E = SXX - 2SXU - 2SD + SUU \tag{12}$$

where

$$SXX = \sum_{i=N+3}^{M} x_i^2 + 2a_1 \sum x_i x_{i-1} + 2a_2 \sum x_i x_{i-2} + a_1^2 \sum x_{i-1}^2$$

$$+ 2a_1 a_2 \sum x_{i-1} x_{i-2} + a_2^2 \sum x_{i-2}^2$$

$$SXU = b_1 \sum x_i u_{i-N-1} + b_2 \sum x_i u_{i-N-2} + a_1 b_1 \sum x_{i-1} u_{i-N-1}$$

$$+ a_1 b_2 \sum x_{i-1} u_{i-N-2} + a_2 b_1 \sum x_{i-2} u_{i-N-1} + a_2 b_2 \sum x_{i-2} u_{i-N-2}$$

$$SD = D \Sigma x_i + a_1 D \Sigma x_{i-1} + a_2 D \Sigma x_{i-2} - b_1 D \Sigma u_{i-N-1} - b_2 D \Sigma u_{i-N-2} - m\frac{D^2}{2}$$

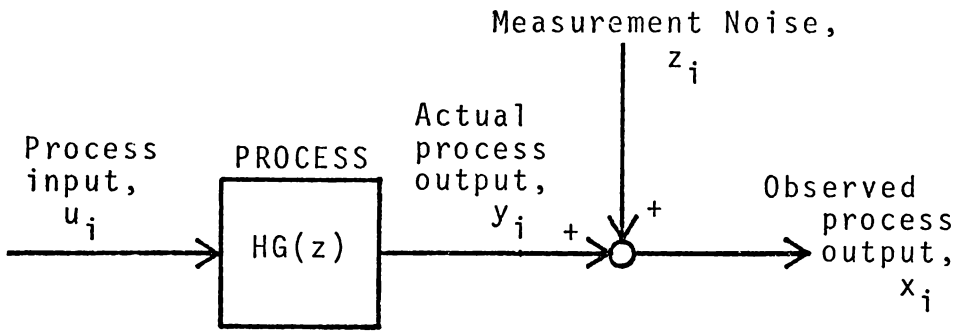
$$SUU = b_1^2 \Sigma u_{i-N-1}^2 + 2b_1 b_2 \Sigma u_{i-N-1} u_{i-N-2} + b_2^2 \Sigma u_{i-N-2}^2$$

Since equation (12) involves only terms which would be available from the linear regression calculations, it offers a savings in computations when compared with the individual residual evaluations of equation (6).

The least-squares modeling scheme diagrammed in Figure 6 was programmed in Fortran IV for execution on the IBM 360 computer. Because standard precision computations were found to be inadequate in solving the normal equations for the simulated response data, the program was written in double precision. The relationships permitting the simplified calculation of the normal equation summation terms and the fit criterion were, of course, incorporated in the linear regression subroutine. The numerical method chosen for solution of the normal equations was pivotal condensation (13), which is the standard Gaussian elimination method with partial pivoting. When an estimate of the process offset is available and has been subtracted from the process output data, a linear regression estimate of D is not necessary, although the normal equations still apply with all terms involving D equal to zero. This option was included in the program.

Effects of Measurement Noise

Consider the important case diagrammed below in which the actual process output is contaminated with noise. The measured process output can be visualized as the sum of the actual output and a random signal. In this situation,



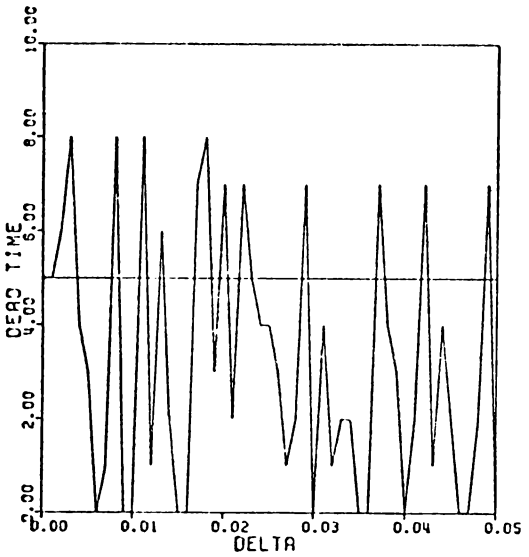
the independent variables, x_{i-1} and x_{i-2}, of the discrete-time model contain measurement errors, and the basic least-squares assumption of measurement errors only in the dependent variable is violated. Therefore, a least-squares modeling of the process using the observed process response data does not assure unbiased estimates of the process parameters (14). In view of this, a brief evaluation of the noise sensitivity of the least-squares modeling scheme was undertaken.

The IBM subroutine RANDU (15) was used to generate random numbers which were added to the output data of the simulated step response (see equation (11) and Figure 2). The random numbers were uniformly distributed between $+\delta$ and

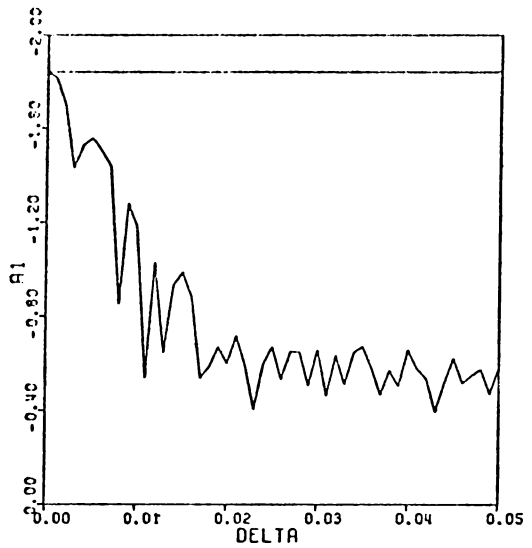
- δ. For values of δ up to 0.05, the noisy response data were modeled with the least-squares modeling program to determine the effect of increasing noise levels on the process parameter estimates. Figures 7 and 8 graphically summarize the results obtained. A horizontal line across each graph indicates the actual value of the particular process parameter.

Even for the extremely low noise levels of this study, it is evident the least-squares modeling performed poorly. Dead time estimates behaved erratically, deviating significantly from the actual process dead time. Least-squares estimates of a_1 and a_2 differed considerably from the process values with the deviation increasing with increasing noise levels. Although the definition of a_2 for a second-order lag process is the positive product of two exponentials, negative estimates of a_2 were obtained. Estimates of the process gain were consistently high with a maximum deviation from the process value of about 35%, while the values obtained for offset were generally better with a maximum deviation near 10%.

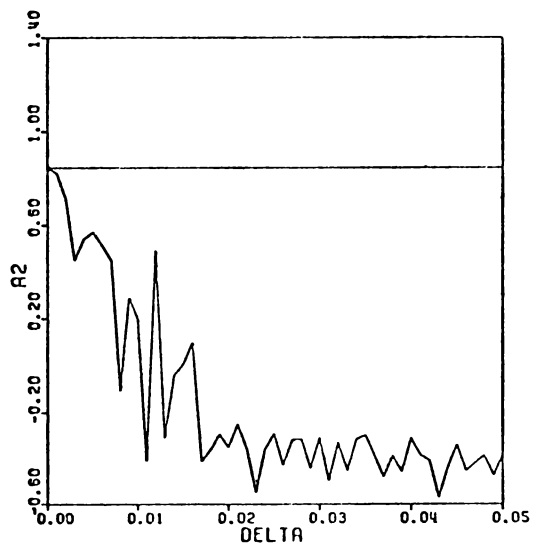
Statistically, the poor performance of the least-squares fit of noisy data is attributable to measurement errors in the independent variables of the dynamic model. Numerically, the biased parameter estimates are due to the ill-conditioned nature of the linear regression normal equations. Although measurement noise introduces small changes



(a) Dead Time Estimates

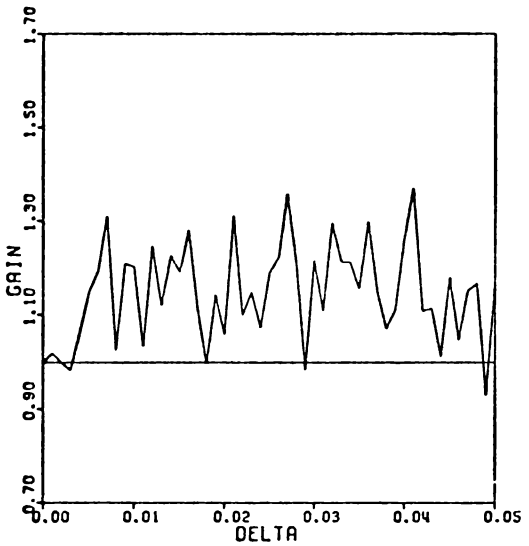


(b) Estimates of a_1

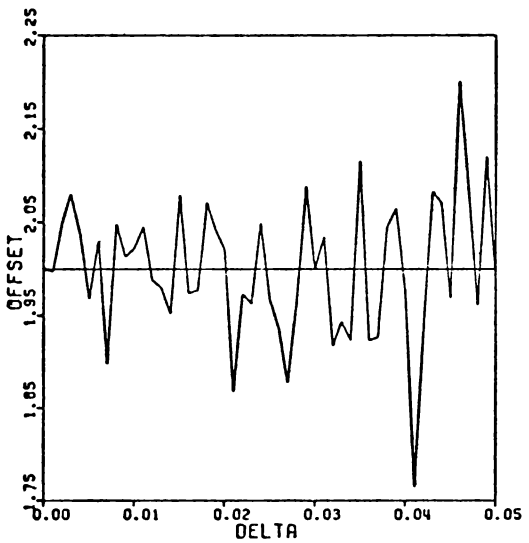


(c) Estimates of a_2

Figure 7. Least-Squares Estimates of the Dynamic Parameters for Simulated Step Response Data with Measurement Noise



(a) Estimates of Process Gain



(b) Estimates of Offset

Figure 8. Least-Squares Estimates of the Steady-State Parameters for Simulated Step Response Data with Measurement Noise

in the summation terms of the normal equations, large changes in the least-squares coefficients result because the equations are ill-conditioned.

Another form of the second-order model which is not so sensitive to measurement noise is the so-called "free-running" model (4):

$$\hat{x}_i = -a_1\hat{x}_{i-1} - a_2\hat{x}_{i-2} + b_1u_{i-N-1} + b_2u_{i-N-2} + D \qquad (15)$$

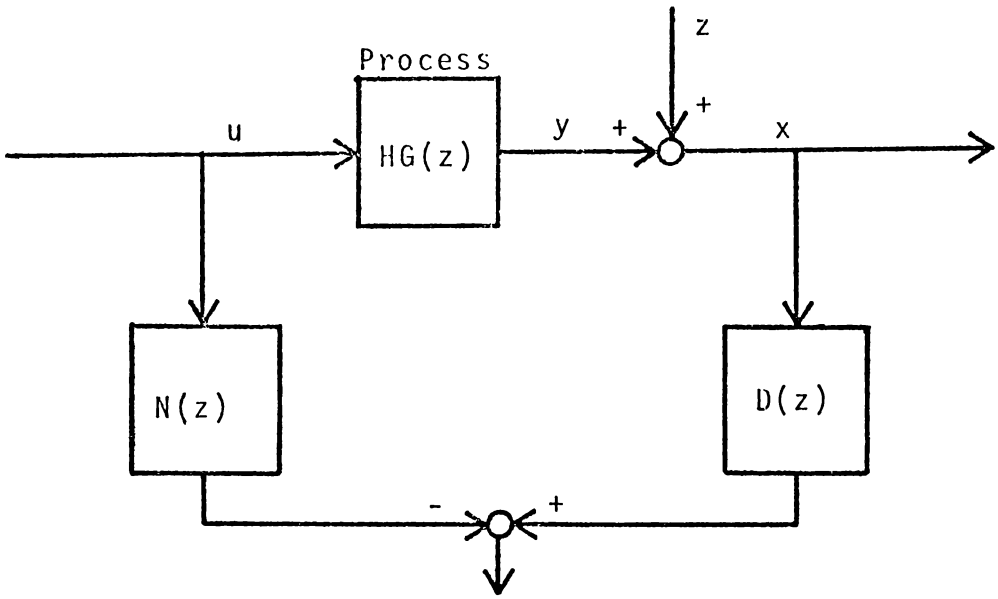
In this equation, the model output is not a function of the observed process output but of past model estimates of the process output. Thus, measurement noise is not involved in the calculation of the model output. Figure 9 shows block diagram representations of the model errors associated with the least-squares and free-running models. N(z) and D(z) are the numerator and denominator, respectively, of the model pulse transfer function. For the second-order model,

$$N(z) = z^{-N}(b_1z^{-1} + b_2z^{-2})$$

and

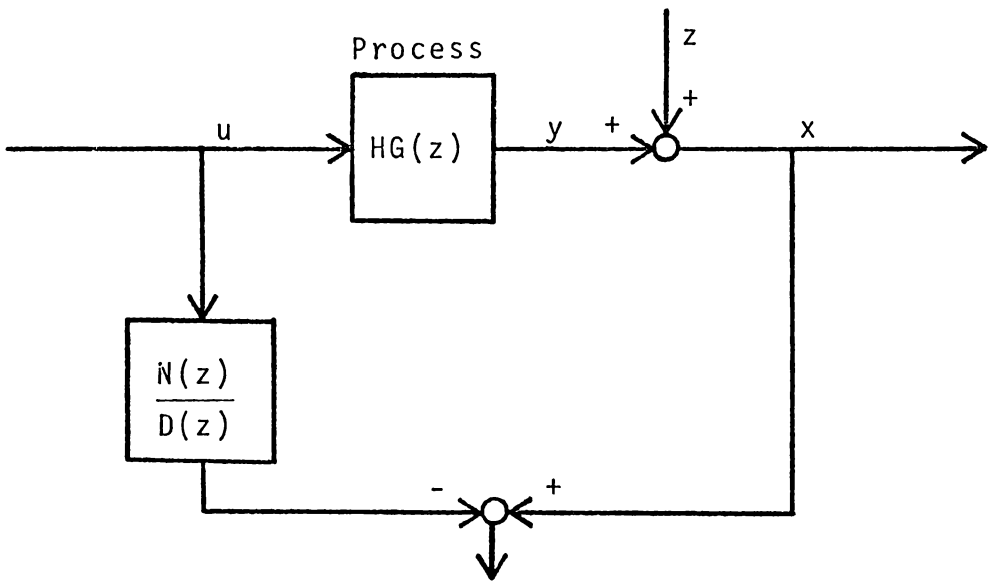
$$D(z) = 1 + a_1z^{-1} + a_2z^{-2} \quad .$$

The modeling of process data using equation (15) is a non-linear regression problem. However, Steiglitz and McBride (16) have shown that the minimization of the free-running model error may be accomplished by iteratively minimizing the least-squares model error. A diagram of their iterative scheme is presented in Figure 10.



Model error, $e=x-\hat{x}$

(a) Least-Squares Model Error



Model error, $e=x-\hat{x}$

(b) Free-Running Model Error

Figure 9. Model Error for Least-Squares and Free-Running Models

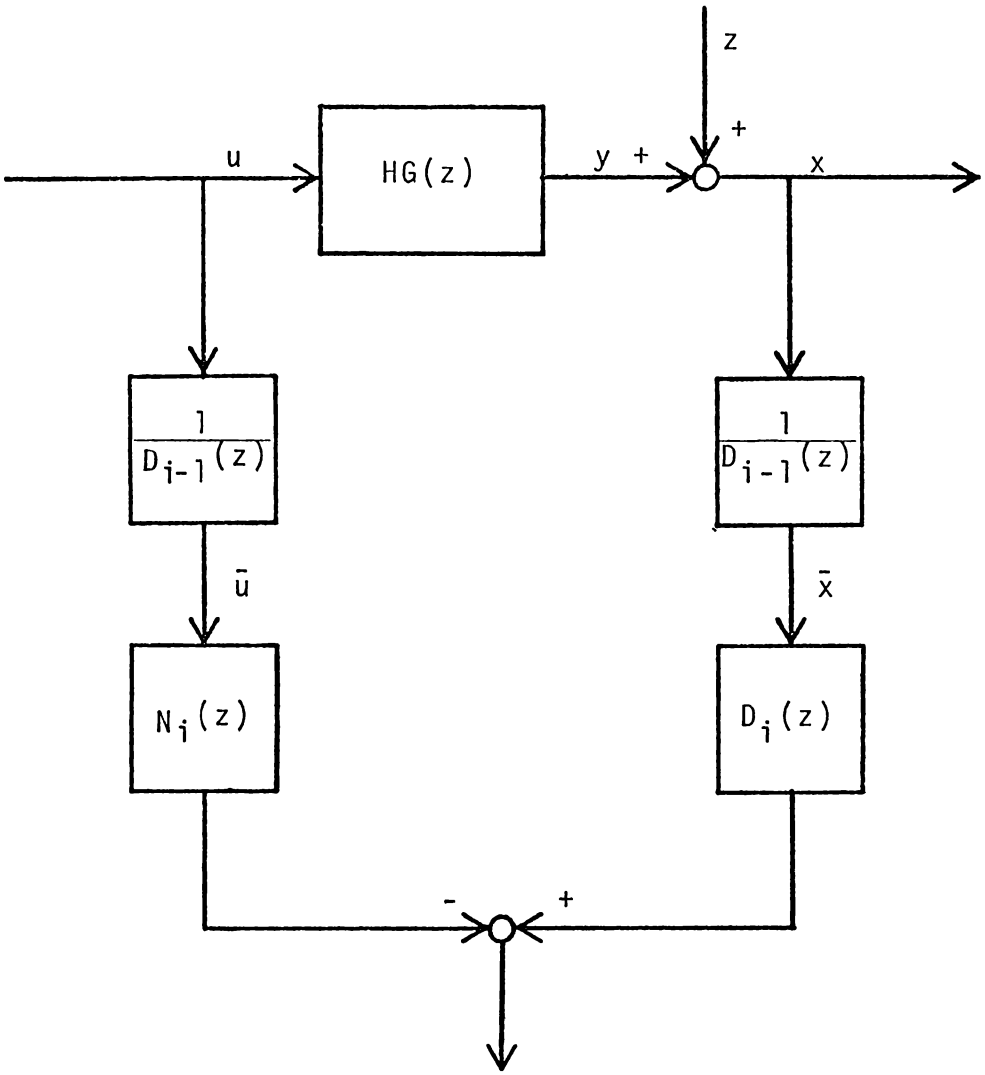


Figure 10. Model Error for Steiglitz-McBride Iterative Modeling Scheme

Each iteration consists simply of filtering the original process input/output data and performing a least-squares fit of this filtered data. The digital filter used to produce the filtered data, $\bar{u}$ and $\bar{x}$, is the inverse of the least-squares estimate of D(z) obtained in the preceeding iteration. For a normalized filter (unity gain),

$$\frac{\bar{X}(z)}{X(z)} = \frac{1+a_1+a_2}{1+a_1z^{-1}+a_2z^{-2}}$$

and

$$\bar{x}_i = K_f x_i - a_1\bar{x}_{i-1} - a_2\bar{x}_{i-2}$$

where $K_f = 1+a_1+a_2$, with an analogous expression for $\bar{u}_i$. If the minimization of the least-squares model error is obtained at each iteration and the coefficients of D(z) converge, that is, $D_i(z) = D_{i-1}(z)$ after a number of iterations, then the block diagram of Figure 10 becomes that of Figure 9(b) and the free-running model error is minimized. Although no proof of convergence was offered, Steiglitz and McBride reported convergence within 10-20 iterations in simulations of more than 50 different systems with signal-to-noise ratios as low as 0.6. In every case, there was significant improvement over the least-squares parameter estimates.

In Figure 11 is presented a flow diagram of the iterative modeling technique. A computer program was written to implement this scheme using the least-squares program developed earlier to perform the least-squares regression on

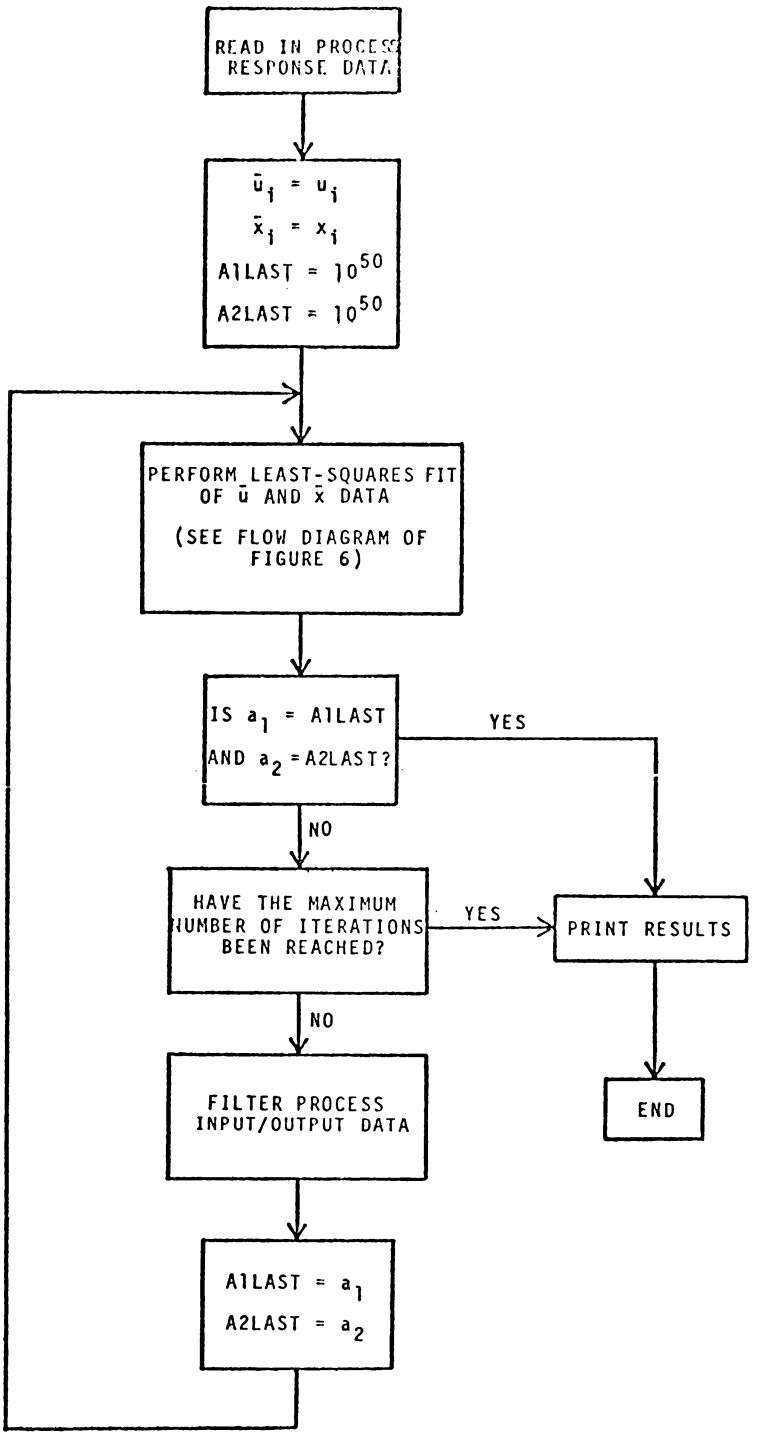


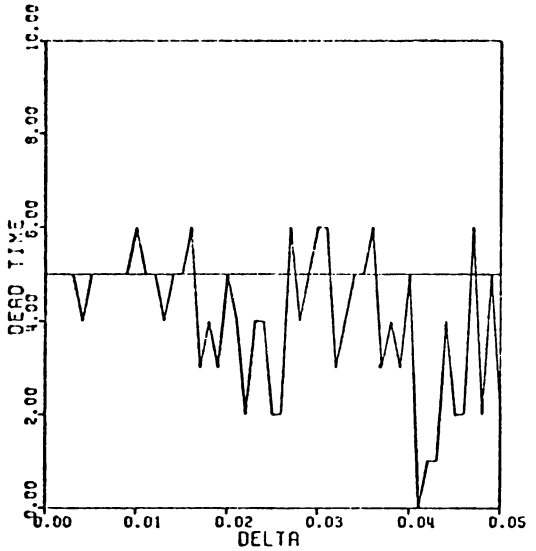
Figure 11. Flow Diagram for Iterative Modeling Scheme

the filtered data. The convergence criterion chosen was no difference in the sixth significant figure between successive estimates of a_1 and a_2, and the maximum number of iterations was fixed at 25. This modeling program was used to model the same noisy step response data used in studying the noise sensitivity of the least-squares modeling scheme. The results are presented in Figures 12 and 13.

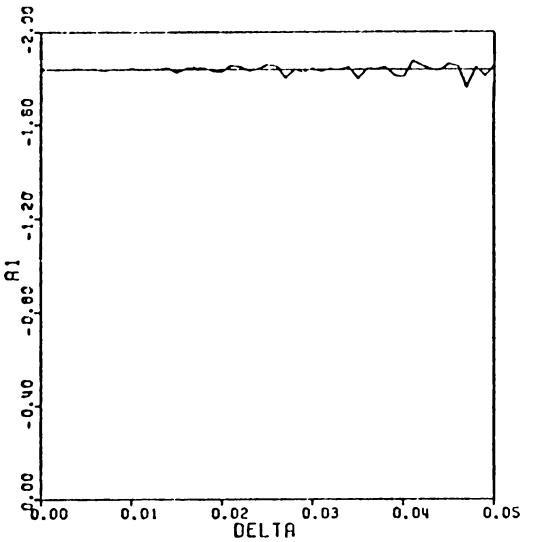
These results show that the iterative modeling program is effective in improving the least-squares parameter estimates. Although the model dead time varied with the noise level, the estimates were considerably better than those obtained from the least-squares regression alone. Model estimates of a_1, a_2, gain, and offset were greatly improved with no significant deviations from the process values.

Discrete-Time Modeling Program

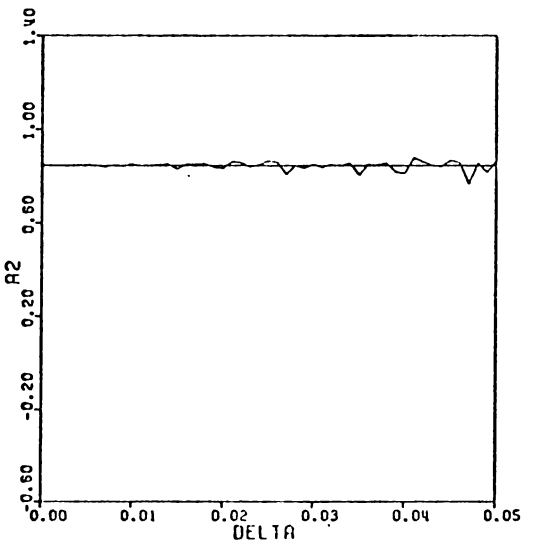
The iterative modeling program was modified slightly for use as a discrete-time modeling package to fit process input/output data with the model of equation (15). To provide a general data input feature, provisions were made for a user-supplied subroutine INPUT which reads in the process data to be modeled. Noise tests of the modeling scheme showed that the model dead time was constant after three or four iterations. Therefore, the program was modified to hold the dead time estimate constant after five iterations, eliminating the dead time search at that point. Except for these minor revisions, the program flow chart is the same as that



(a) Dead Time Estimates

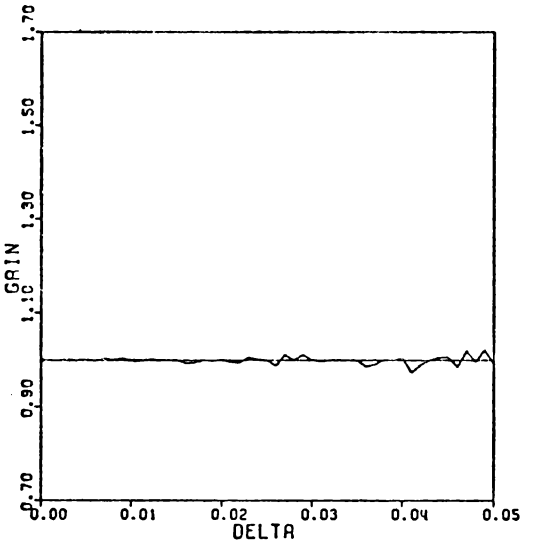


(b) Estimates of a_1

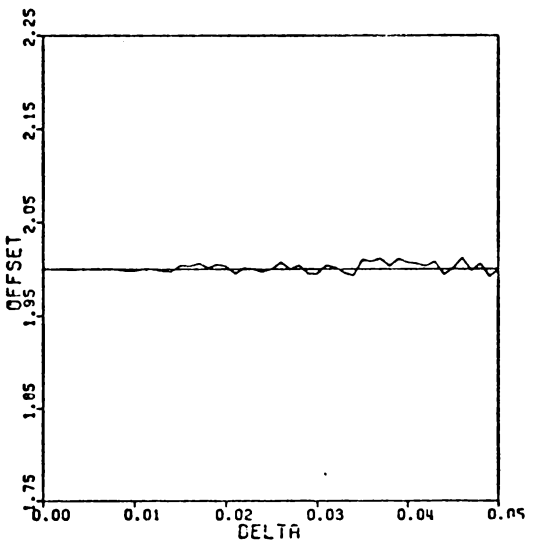


(c) Estimates of a_2

Figure 12. Iterative Modeling Estimates of the Dynamic Parameters for Simulated Step Response Data with Measurement Noise



(a) Estimates of Process Gain



(b) Estimates of Offset

Figure 13. Iterative Modeling Estimates of the Steady-State Parameters for Simulated Step Response Data with Measurement Noise

shown in Figure 11.

The program output consists of the number of iterations required for convergence, the values of the model parameters and the sum of the squared errors. Also, a printout of the original process data is given together with the output predicted by the model and the corresponding model error.

Appendix A contains a complete listing of the main program MODEL, subroutine ASUBI, and an example of the user-supplied subprogram INPUT. Definitions of key program variables, descriptions of the program flow, and instructions for program use are documented within the program by comment statements.

CHAPTER III

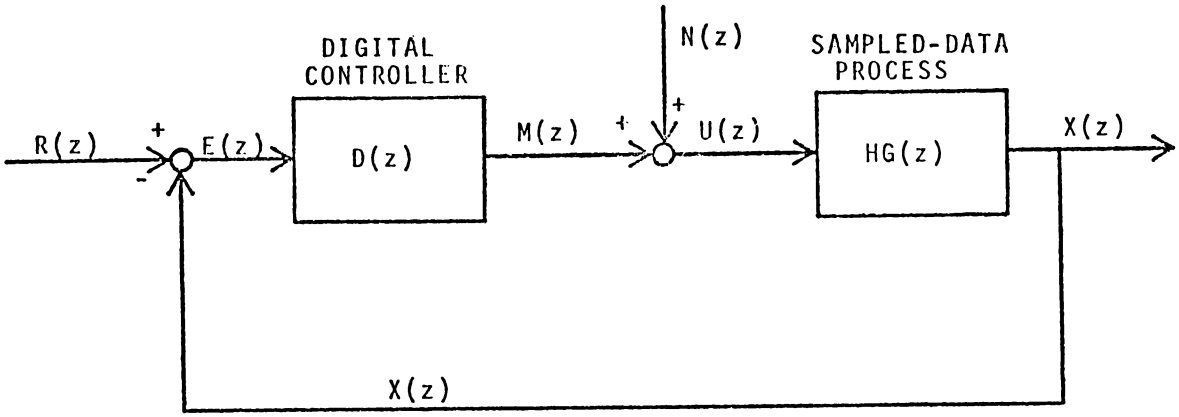
CONTROLLER DEVELOPMENT AND SIMULATION RESULTS

Because of the widespread use of the analog control modes in continuous control of chemical processes, it was natural that the first control algorithms developed for direct digital control were extensions of these modes. This approach to controller specification still prevails in the majority of DDC applications, with the most common controller algorithms being the discrete PI and PID algorithms obtained by representing the integral and derivative modes by a summation and a finite difference, respectively. However, the performance of these algorithms is limited to that attainable with their continuous counterparts (17), and the real promise and economic justification of direct digital control lies in the development of improved control strategies which are either impossible or too expensive to implement using conventional analog equipment. The mathematics of z-transforms is a potentially valuable tool for use in this development.

Z-Transform Controller Design

The z-transform design of digital controllers is a commonly included topic in texts dealing with sampled-data control theory (4,5). The basis of the technique may be developed by a simple analysis of the sampled-data control

loop below. As shown in Table II, there are eight variables



involved, and block diagram algebra gives four independent relations among these variables. This results in four degrees of freedom for the system, which means that once any four of the control loop variables are specified, the remaining four variables are automatically fixed. Therefore, desired control loop characteristics may be used to define four of the system variables, and the controller transfer function required to produce these operating characteristics can be determined.

As an example of this approach to controller design, consider the following. Suppose there is a controller, $D_0(z)$, which gives an acceptable set point change response when controlling a process, $HG_0(z)$, containing no dead time. If applied to the same process with dead time added, the controller may not produce an acceptable set point change response, because the control action, M(z), calculated for

TABLE II

SAMPLED-DATA CONTROL LOOP VARIABLES

Variables	Independent Relationships
R(z), set point	E(z) = R(z) - X(z)
E(z), error	
D(z), controller	M(z) = D(z)E(z)
M(z), controller output	
N(z), disturbance	U(z) = M(z) + N(z)
U(z), process input	
HG(z), process pulse transfer function	X(z) = HG(z)U(z)
X(z), process output	
Total = 8	Total = 4
Degrees of freedom = 4	

the dead time process will, most likely, differ from the control action, $M_0(z)$, for the case of no dead time. This is, of course, the situation encountered with discrete versions of the three-mode analog controllers. What is needed for acceptable control of the dead time process is a controller D(z) which, for a set point change, outputs the control action $M_0(z)$. The process response is then the response for the no-dead-time case delayed one dead time.

For the case of no dead time, four of the control loop variables must be specified before $M_0(z)$ can be found. $D_0(z)$ and the process transfer function, $HG_0(z)$, are fixed and for set point design, R(z) is specified with N(z)=0. This gives the control action as

$$M_0(z) = \frac{D_0(z)R(z)}{1+D_0(z)HG_0(z)} \quad .$$

For the same system with dead time, R(z) and N(z) remain unchanged, the process pulse transfer function is given as

$$HG(z) = z^{-N}HG_0(z)$$

and

$$M(z) = M_0(z) \quad .$$

The controller is then

$$D(z) = \frac{M(z)}{E(z)} = \frac{M_0(z)}{R(z)-HG(z)M_0(z)}$$

and substitution of the expressions for HG(z) and $M_0(z)$

yields

$$D(z) = \frac{D_0(z)}{1+D_0(z)HG_0(z)[1-z^{-N}]} \quad . \tag{16}$$

With this as the controller for the dead time process, the control loop transfer function is

$$\frac{X(z)}{R(z)} = \frac{D_0(z)HG_0(z)z^{-N}}{1+D_0(z)HG_0(z)}$$

which indicates that if the process is known exactly, the dead time is effectively removed from the feedback loop.

The controller of equation (16) gives the discrete equivalent of a control strategy proposed by Smith (18) for dead time compensation. Originally developed for continuous control, the Smith predictor uses a simulation of the dead time process to cancel the actual process output, and for perfect modeling the control action is based on the output of a minimum-phase (no dead time) process model. Direct digital control versions of Smith's scheme have been studied (19,20,21) with significant improvement over conventional algorithms. Equation (16) gives the predictor in a single, closed-form expression, eliminating the need for explicit simulations of the process.

The digital controller generally presented in discussions of z-transform controller design is the deadbeat controller. The deadbeat design procedure is to specify that the set point change response reach the new set point

in a minimum number of sample times and remain at the set point for all subsequent samples. The controller transfer function giving this performance can be determined after fixing the type of set point change (step,ramp,etc.) and the process pulse transfer function. The disturbance is, of course, zero. Kuo (4) points out that deadbeat response design has several disadvantages. First, while the design assures minimum-time response at the sampling instants, there is no assurance that the process output is constant between samples. Application of the deadbeat controller could conceivably produce an unstable set point change response that satisfied the design criteria simply by coinciding with the set point at each sample. Second, although the deadbeat control system is optimal for the specific input used for design, unsatisfactory performance may be obtained for other inputs. This difficulty is of no major consequence for set point change design in the process industries, since the set point change is consistently a step input. However, the response of systems designed for a step in disturbance might suffer because of this point. Finally, deadbeat design is basically a pole-zero cancellation, and Kuo indicates the results are highly sensitive to modeling errors. This disadvantage certainly affects the applicability of the method to digital control of chemical processes, the dynamics of which are seldom known precisely.

Mosler (22) reported the design of deadbeat controllers based on a first-order lag model for both set point and load

compensation, and applied the algorithms to digital control of a laboratory-scale temperature control system. Performance comparable to that of continuous PID control was obtained by adding a P-D transmitter to cancel one of the poles of the second-order process. Lane (23) also designed a first-order deadbeat controller for set point changes and presented a method for on-line tuning of the algorithm. Simulations using a first-order lag plus dead time process with an analog dead time approximation gave responses which were close to the ideal deadbeat response. However, the algorithm produced an excessively oscillating control action, a phenomenon known as controller ringing.

In another case, Dahlin (24) designed digital controllers based on a first-order lag plus dead time closed-loop response for a set point change. The response dead time was that of the process, and the lag time constant was used as a tuning parameter. Controller ringing was exhibited in simulations of the controller, but Dahlin showed that the useful portion of the control action could be retained by simply eliminating the controller poles causing ringing with appropriate gain adjustment to preserve the original controller steady-state gain. Chiu (25) applied the Dahlin design method in the simulated DDC of a jacketed backmix reactor using both first- and second-order process models. Eliminating the ringing poles of the controllers obtained gave a discrete PI controller for the first-order model and

a discrete PID controller for the second-order design.

The Kalman Controller

Like the deadbeat controller design, the z-transform controller design presented by Kalman (26) is based upon a minimum-time set point change response. Using a second-order pulse transfer function for the process, the design differs in that the form of the controller output is also fixed. Although Kalman neglected process dead time, it is a simple matter to include this important variable in the design. As shown in Figure 14(a), the second-order process output is to reach the set point in N+2 sample times. This gives

$$X(z) = x_1 z^{-N-1} + z^{-N-2} + z^{-N-3} + \ldots$$

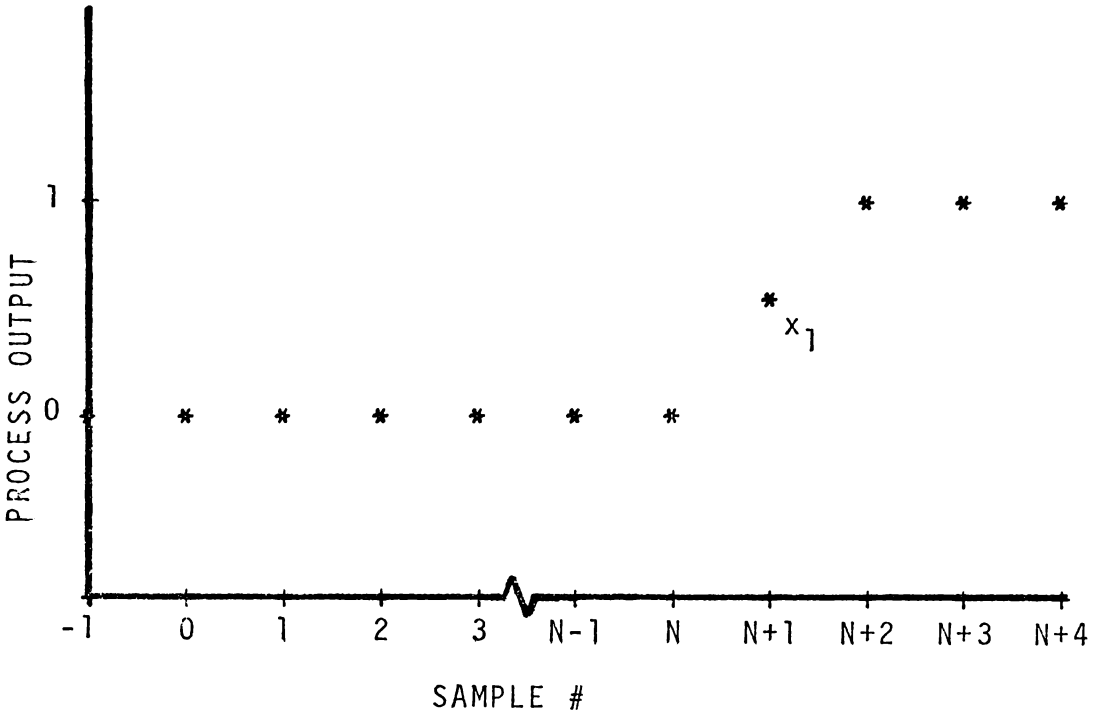
In forcing the process to the new set point, the controller operates in bang-bang fashion as in Figure 14(b). Thus,

$$M(z) = m_o + m_1 z^{-1} + m_f z^{-2} + m_f z^{-3} + \ldots$$

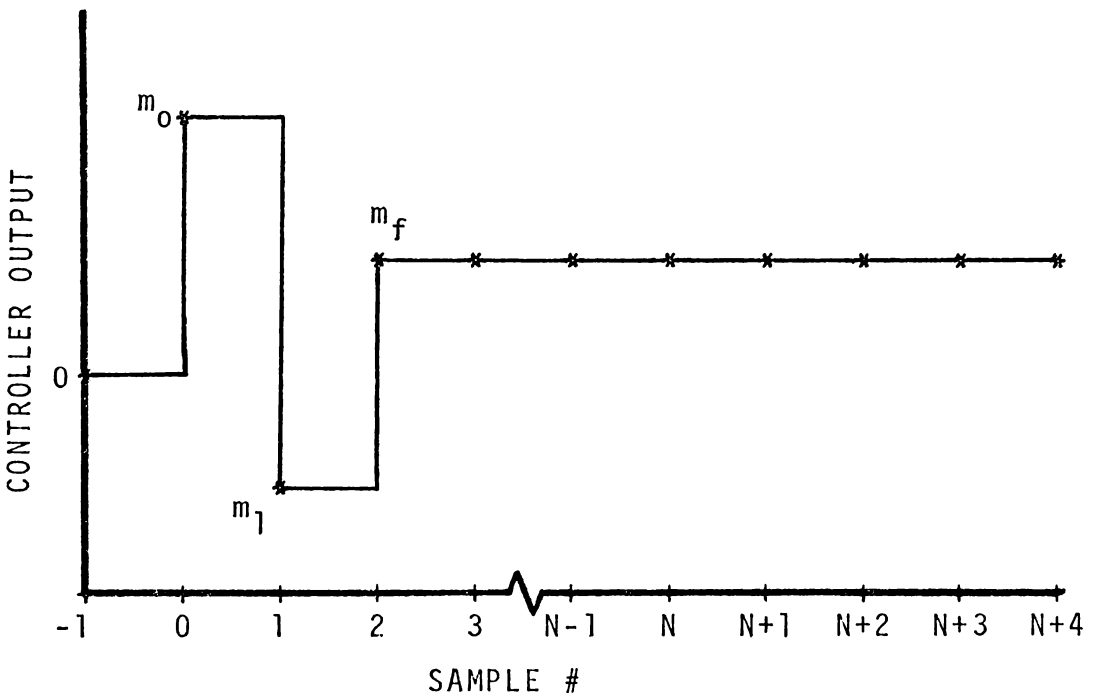
Note that for perfect modeling of the process, controller ringing is eliminated since the control action is limited to three switches. For a unit step change in set point,

$$R(z) = \frac{1}{1-z^{-1}}$$

$$\frac{X(z)}{R(z)} = x_1 z^{-N-1} + [1-x_1] z^{-N-2} = p_1 z^{-N-1} + p_2 z^{-N-2} = P(z) \qquad (17)$$



(a) Process Output



(b) Controller Output

Figure 14. Process Response and Control Action for Kalman Controller

and

$$\frac{M(z)}{R(z)} = [1-z^{-1}][m_0+m_1z^{-1}+m_fz^{-2}+m_fz^{-3}+\ . \ . \ .]$$

$$= m_0+(m_1-m_0)z^{-1}+(m_f-m_1)z^{-2}$$

$$= q_0+q_1z^{-1}+q_2z^{-2} = Q(z) \qquad (18)$$

From the definition of the process pulse transfer function and equations (17) and (18), it follows that

$$HG(z) = \frac{X(z)}{M(z)} = \frac{P(z)}{Q(z)}$$

and the second-order pulse transfer function gives

$$\frac{P(z)}{Q(z)} = \frac{z^{-N}[b_1z^{-1}+b_2z^{-2}]}{1+a_1z^{-1}+a_2z^{-2}}$$

$$= \frac{\frac{b_1}{b_1+b_2}z^{-N-1}+\frac{b_2}{b_1+b_2}z^{-N-2}}{\frac{1}{b_1+b_2}+\frac{a_1}{b_1+b_2}z^{-1}+\frac{a_2}{b_1+b_2}z^{-2}}$$

Division of the numerator and denominator of HG(z) by b_1+b_2 is necessary to ensure that $p_1+p_2=1$ as required by equation (17).

The controller transfer function is given as

$$D(z) = \frac{X(z)/R(z)}{HG(z)[1-X(z)/R(z)]}$$

and substitution of the above results yields

$$D(z) = \frac{Q(z)}{P(z)} \quad \frac{P(z)}{1-P(z)} = \frac{Q(z)}{1-P(z)}$$

$$= \frac{\frac{1}{b_1+b_2}\left[1+a_1z^{-1}+a_2z^{-2}\right]}{1-\frac{b_1}{b_1+b_2}z^{-N-1}-\frac{b_2}{b_1+b_2}z^{-N-2}}$$

Inverting this expression gives the digital control algorithm

$$m_i = K_c[e_i+a_1e_{i-1}+a_2e_{i-2}+b_1m_{i-N-1}+b_2m_{i-N-2}]$$

where $K_c = \frac{1}{b_1+b_2}$.

Inspection of this control algorithm shows that the Kalman controller output can be viewed as the sum of two control actions. If the error terms are expanded, the algorithm can be written as

$$m_i = K_c[r_i+a_1r_{i-1}+a_2r_{i-2}-x_i-a_1x_{i-1}-a_2x_{i-2}$$
$$+b_1m_{i-N-1}+b_2m_{i-N-2}] \quad .$$

From the least-squares model equation,

$$\hat{x}_i = -a_1x_{i-1}-a_2x_{i-2}+b_1m_{i-N-1}+b_2m_{i-N-2} \qquad (19)$$

and

$$m_i = K_c[r_i+a_1r_{i-1}+a_2r_{i-2}]+K_c[\hat{x}_i-x_i] \quad .$$

Therefore, the Kalman control action is the sum of the output

from an optimal open-loop controller, $M_r(z)$, and the output of a model-error controller, $M_m(z)$:

$$M(z) = M_r(z) + M_m(z)$$

where $$M_r(z) = K_c[1+a_1z^{-1}+a_2z^{-2}]R(z)$$

$$M_m(z) = K_cE_m(z)$$

$$E_m(z) = \hat{X}(z) - X(z)$$

A schematic representation of this expansion of the Kalman controller is shown in Figure 15.

The open-loop controller responds only to set point changes and supplies the same control action regardless of the process dead time, a required property for a good dead time compensator. The model-error controller is essentially a load estimator which attributes any difference between the actual and predicted process outputs to the presence of a disturbance acting on the process and subtracts an estimate of the load from the open-loop controller output in an attempt to cancel the disturbance. Thus, $M_m(z)$ provides compensation for load changes and supplements the open-loop control action for set point changes when model errors exist.

For perfect modeling, the open-loop controller may not be optimal for set point changes if the control action is constrained. Figure 16 illustrates the effects of controller constraints upon the process response. The process in the unit step set point change simulation was the second-order

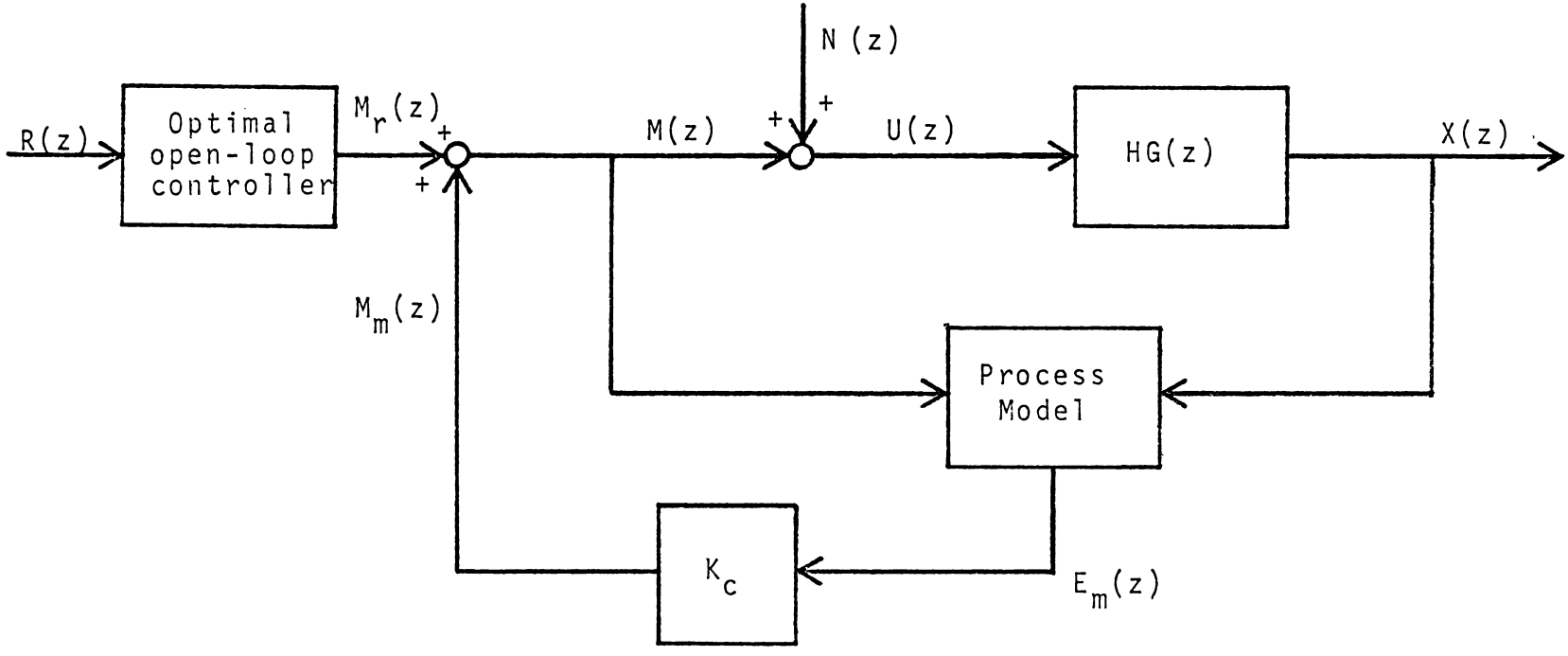


Figure 15. Block Diagram of Kalman Controller

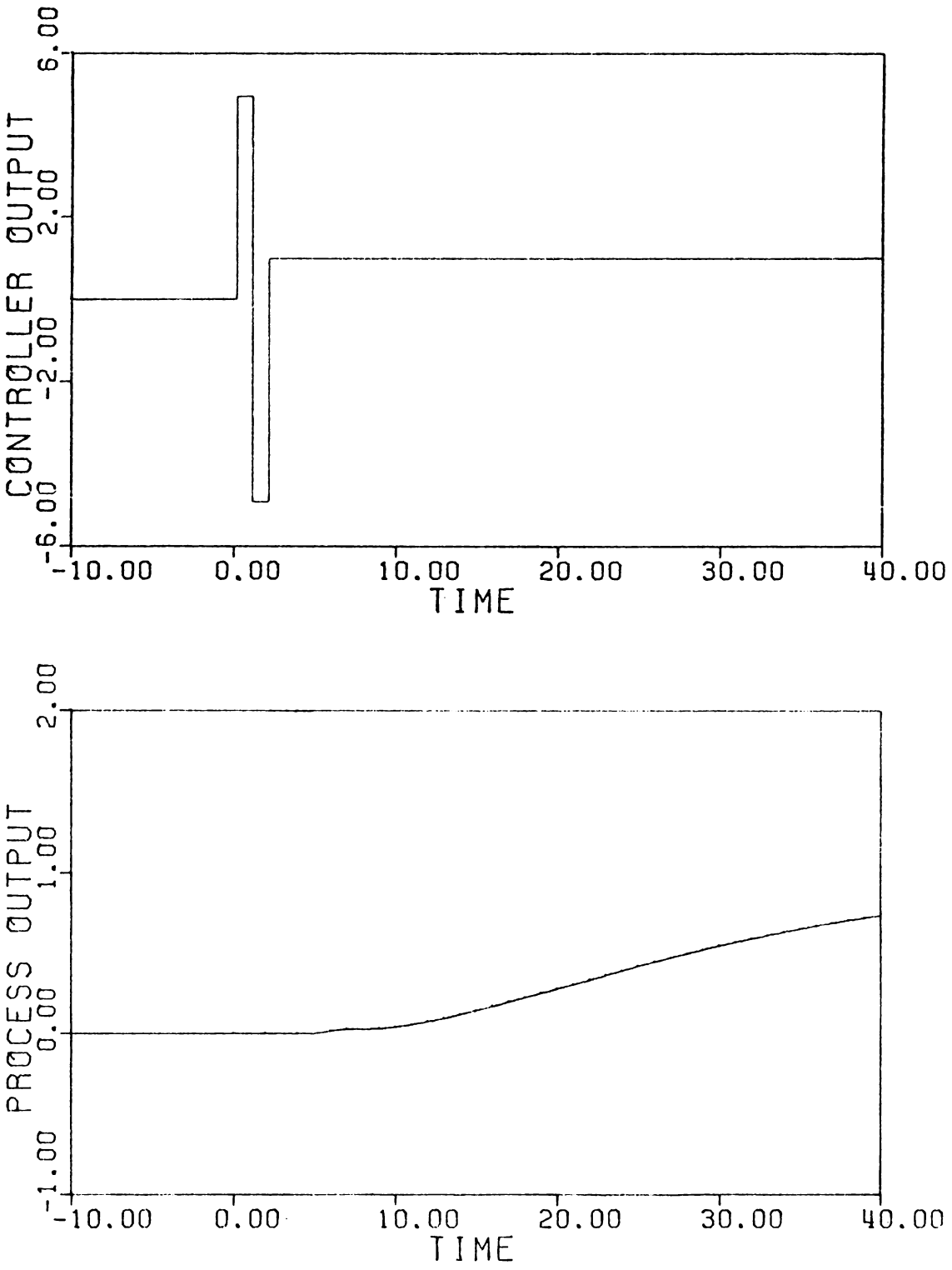


Figure 16. Set Point Change Response for Second-Order Process with Constrained Kalman Controller

lag plus dead time process of equation (11) in Chapter II. Since the process was a true second-order process, there was no modeling error involved giving a model-error control action of zero. The bang-bang control actions dictated by the open-loop controller were greater than the controller constraints of $\pm$ 5. Since the open-loop controller receives no information concerning the process output, the control action remains at its final value after two sample times regardless of the process response. The result is a very sluggish response.

Modification of the Kalman Controller

The basic structure of the Kalman controller is a sound one. An open-loop controller is used to give a minimum-time set point change response for the ideal case of perfect modeling and unlimited control action. This open-loop control action is supplemented by a second control action, based on model error, which serves to eliminate any offset due to imperfect modeling and provides load compensation when disturbances are present. However, for the case of controller constraints, the open-loop controller may give an unacceptable response, and modification of it is necessary.

A constraint-aware controller which preserves the optimal nature of the Kalman controller may be derived as follows. At sample i, suppose we calculate the control action m_i which is necessary to bring the process output to the set point, r_i, in two sample times. For no dead time and

perfect modeling, there are three equations which describe the process output at the next three samples:

$$x_{i+1} = -a_1 x_i - a_2 x_{i-1} + b_1 m_i + b_2 m_{i-1}$$

$$x_{i+2} = -a_1 x_{i+1} - a_2 x_i + b_1 m_{i+1} + b_2 m_i$$

$$x_{i+3} = -a_1 x_{i+2} - a_2 x_{i+1} + b_1 m_{i+2} + b_2 m_{i+1}$$

At sample i, x_i, x_{i-1}, and m_{i-1} are known, and if the process reaches the set point in two sample times, we must have

$$x_{i+2} = x_{i+3} = r_i$$

$$m_{i+2} = r_i/K \quad .$$

The three equations can then be solved for x_{i+1}, m_i, and m_{i+1}. Using two equations to eliminate x_{i+1} and m_{i+1} and solving for m_i gives

$$m_i = C_1 r_i + C_2 x_i + C_3 x_{i-1} + C_4 m_{i-1} \tag{20}$$

where $$C_1 = \frac{1}{b_1 + b_2}$$

$$C_2 = \frac{a_2 b_2 + a_1 a_2 b_1 - a_1^2 b_2}{C_o}$$

$$C_3 = \frac{a_2^2 b_1 - a_1 a_2 b_2}{C_o}$$

$$C_4 = \frac{a_1 b_2 - a_2 b_1 b_2}{C_o}$$

$$C_0 = b_2^2 - a_1 b_1 b_2 + a_2 b_1^2$$

Evaluating this control algorithm for a step change in set point shows that for no controller constraints, the algorithm is equivalent to the Kalman controller. This fact can be used to obtain simpler expressions for C_3 and C_4:

$$C_3 = C_1[1-C_4][a_1+a_2]-C_2-C_1C_4$$

$$C_4 = a_1 - C_2 b_1$$

However, unlike the Kalman controller, if the control action calculated by the algorithm has not been input to the process due to controller constraints, this is reflected in the feedback of the process output, and the controller continues in its attempt to force the process to the set point.

To use the algorithm of equation (20) as the open-loop controller in the Kalman scheme, the process output can be replaced by a minimum-phase process model output, x^o, to yield

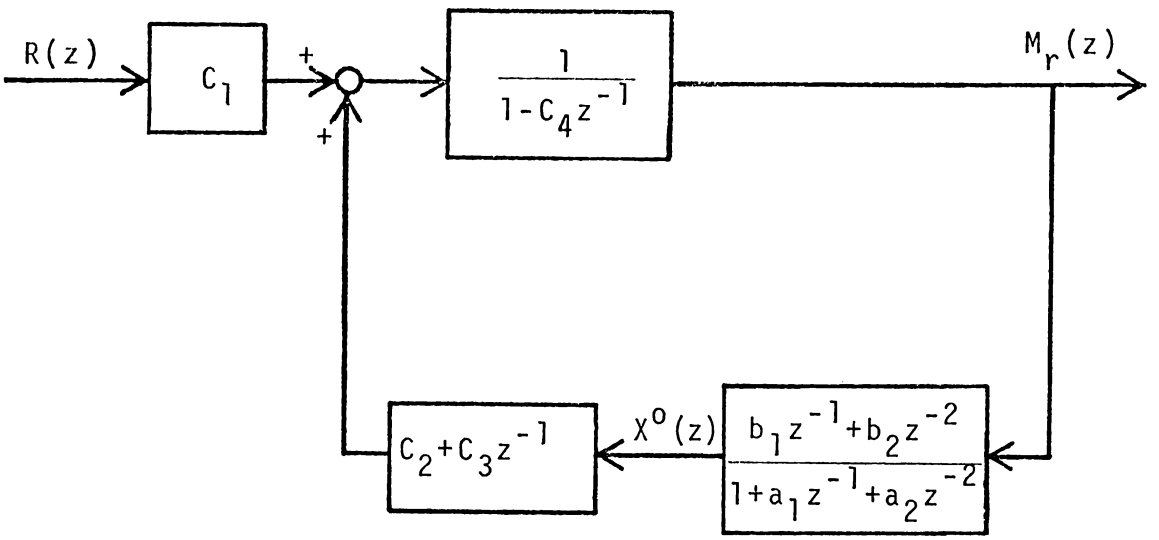
$$m_{r_i} = C_1 r_i + C_2 x_i^o + C_3 x_{i-1}^o + C_4 m_{r_{i-1}} \qquad (21)$$

where

$$x_i^o = -a_1 x_{i-1}^o - a_2 x_{i-2}^o + b_1 m_{r_{i-1}} + b_2 m_{r_{i-2}} \quad .$$

This gives an open-loop control action which is independent of the process dead time. Of course, the constrained values of the controller output are used in these equations rather

than the calculated values. Below is a z-transform block diagram of the controller. This controller was used as the



open-loop controller in the Kalman scheme of Figure 15, and a simulated set point change for the second-order process cited earlier was carried out. Figure 17 gives the results of the simulation and illustrates the improvement over the original Kalman controller for the case of controller constraints.

To investigate the use of the modified Kalman controller for control of higher-order processes, the process below was simulated using the IBM Continuous System Modeling

$$G(s) = \frac{e^{-1.0s}}{(s+1)^5} \tag{22}$$

Program (27) to obtain step response data for sample times of 0.1 and 1.0. The simulated response data was then

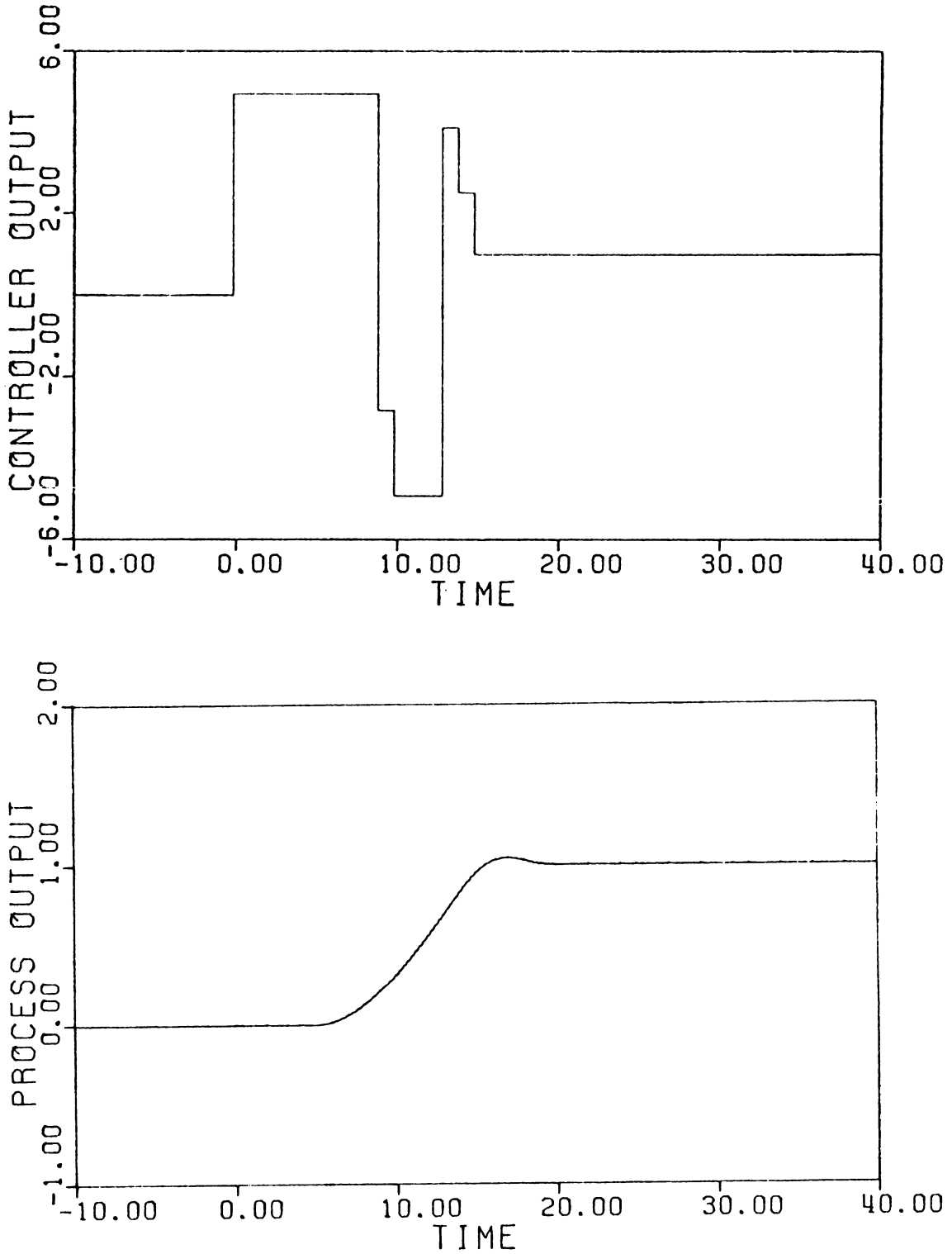


Figure 17. Set Point Change Response for Second-Order Process with Kalman Controller Modified for Controller Constraints

modeled using the discrete-time modeling package in Appendix A to give the second-order model parameters shown in Table III. A CSMP program was also written to simulate direct digital control of the fifth-order process, and set point change and load change responses were obtained. Controller constraints of $\pm$ 5 were used.

Figure 18 shows the performance of the modified Kalman controller for a unit step set point change and 1.0 sample time. While the process output reaches its final value in about 12 samples, the control action does not settle until after another ten samples. For a sample time of 0.10, the response was nearly the same, but the excessive control action was more pronounced. After the process had settled at the set point, the controller output oscillated between 5 and -3 with a period of about 25 samples.

A unit step load change with a sample time of 1.0 gave the response shown in Figure 19. In this case, the model-error controller responds to slowly cancel the disturbance entering the process.

In an attempt to eliminate the excessive control action obtained for set point changes, the model error was filtered using the digital filter

$$\frac{\tilde{E}_m(z)}{E_m(z)} = \frac{K_f}{1+a_1 z^{-1}+a_2 z^{-2}}$$

where $K_f = 1+a_1+a_2$.

TABLE III

MODEL PARAMETERS FOR FIFTH-ORDER PROCESS

T	N	a_1	a_2	b_1	b_2
0.1	23	-1.9275	0.92957	0.00022	0.00178
1.0	2	-1.3447	0.48876	0.03806	0.10320

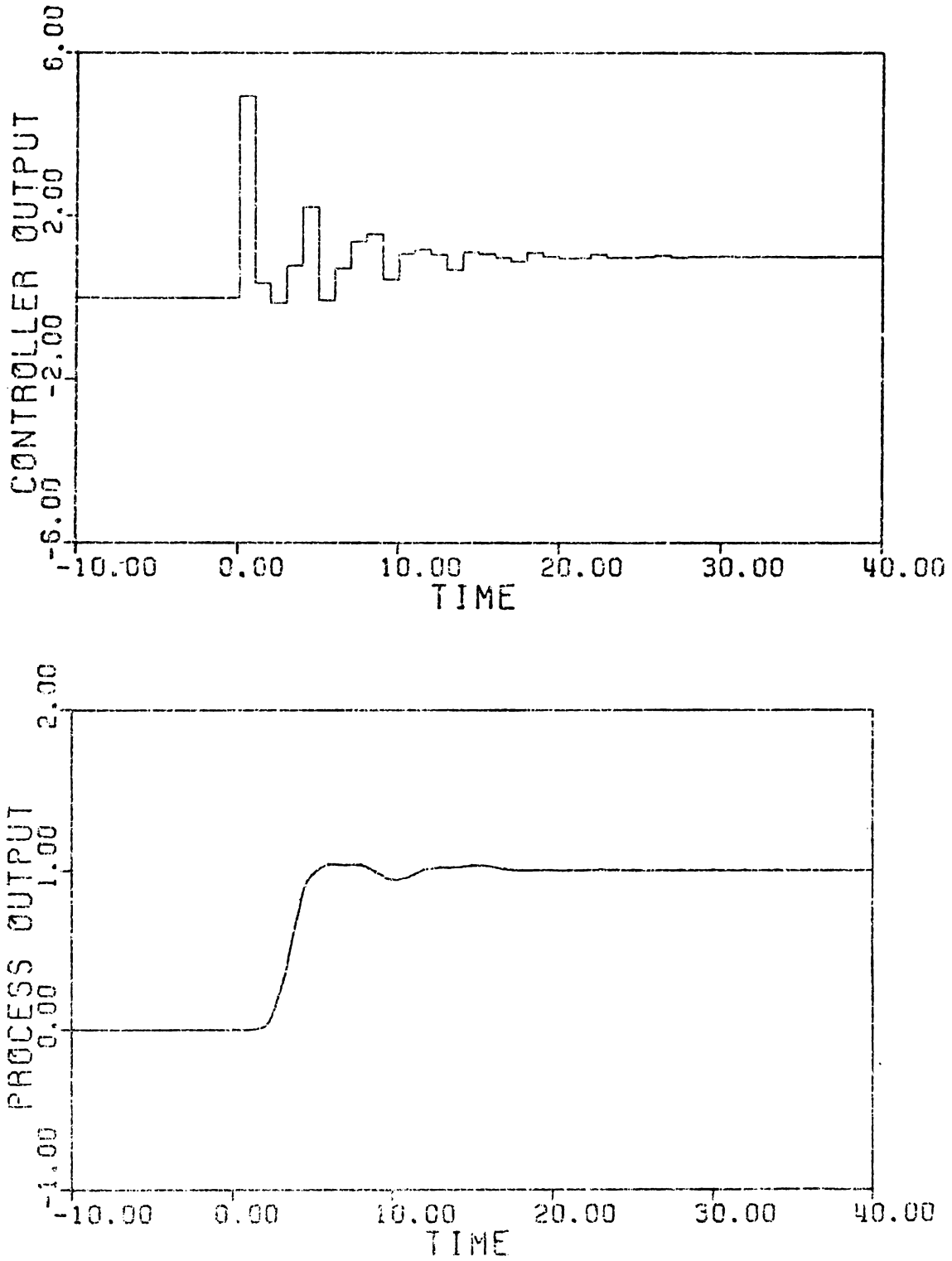


Figure 18. Set Point Change Response for Fifth-Order Process with Kalman Controller Modified for Controller Constraints

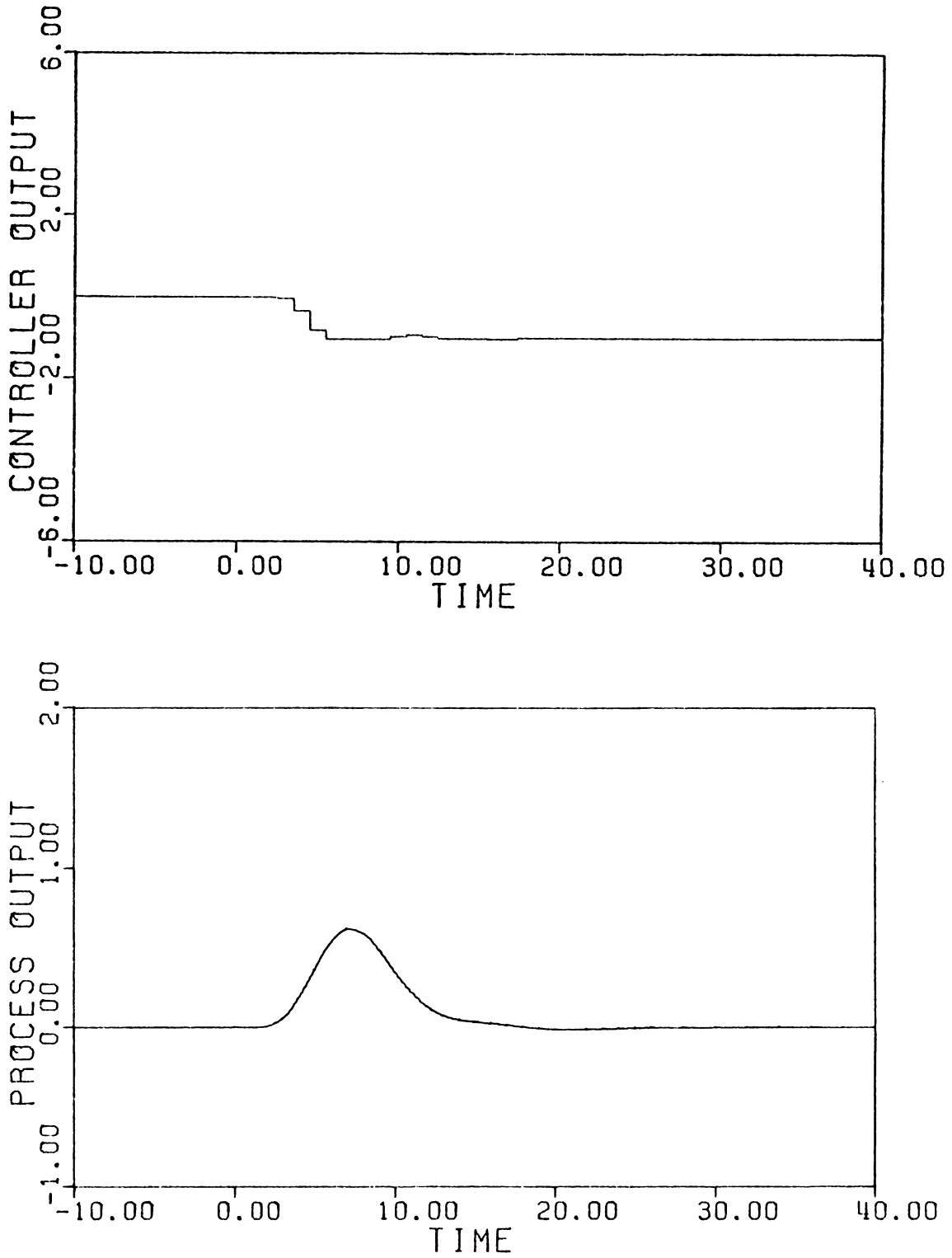


Figure 19. Load Change Response with Kalman Controller Modified for Controller Constraints

The model-error control action was then based on the filtered model error, $\bar{E}_m(z)$:

$$M_m(z) = K_c\bar{E}_m(z)$$

This approach gave the set point change and load change responses of Figures 20 and 21. A sample time of 1.0 was used.

While the process output for the set point change is essentially the same as that in Figure 18, the control action reflects a substantial improvement over that for the unfiltered case. For a sample time of 0.10, a more drastic improvement was observed with the controller oscillations completely eliminated by filtering the model error. However, Figure 21 shows that the improved control action for set point changes is obtained at the expense of a slower load change response. The second-order filter introduces a lag in the model-error controller response, resulting in a slower return to the set point than for the unfiltered case.

Since the model-error controller is essentially a load estimator, an alternative approach to the estimation of the disturbance was developed. Assuming that any difference between the predicted process output and the actual process output at sample i is due to a step disturbance which entered at sample i-N-2 will allow estimation of that step and, therefore, of the present load. For perfect modeling, the process output at sample i is

$$x_i = -a_1x_{i-1}-a_2x_{i-2}+b_1[m_{i-N-1}+n_{i-N-1}]+b_2[m_{i-N-2}+n_{i-N-2}]$$

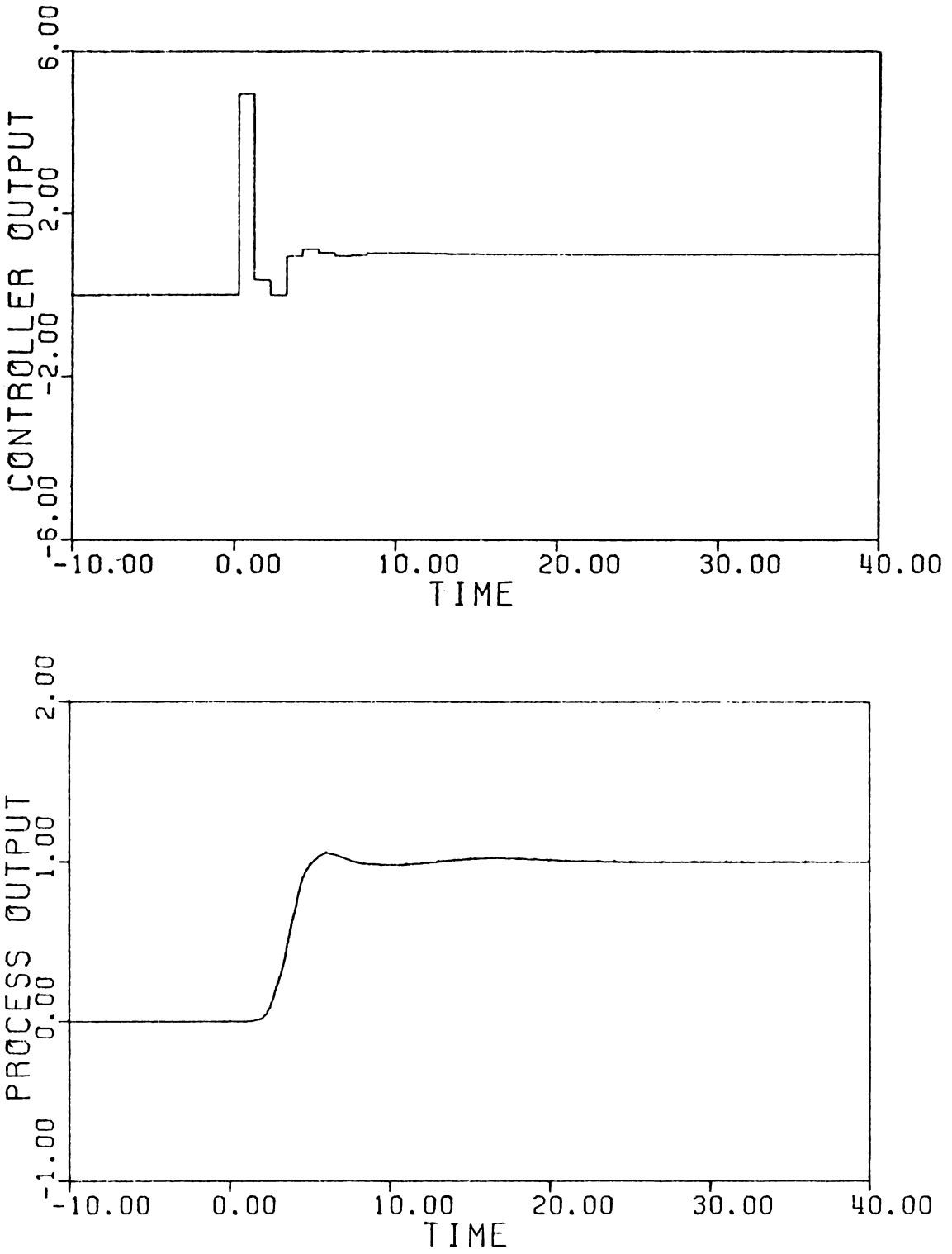


Figure 20. Set Point Change Response for Fifth-Order Process with Modified Kalman Controller for Filtered Model Error

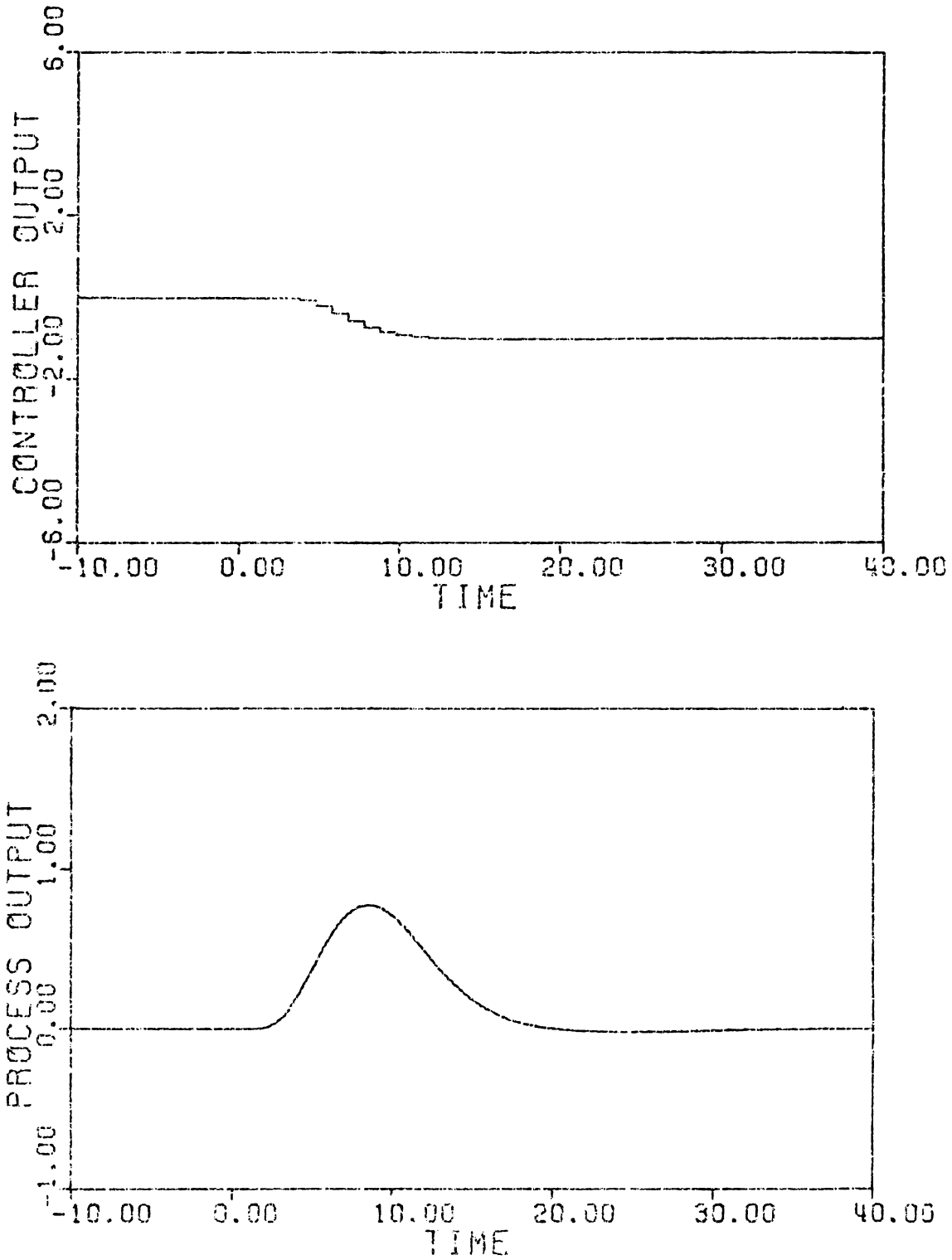


Figure 21 Load Change Response for Fifth-Order Process with Modified Kalman Controller for Filtered Model Error

which, due to the step load assumption, simplifies to

$$x_i = -a_1 x_{i-1} - a_2 x_{i-2} + b_1 m_{i-N-1} + b_2 m_{i-N-2} + [b_1 + b_2] n_i \qquad (23)$$

The predicted output is

$$\hat{x}_i = -a_1 x_{i-1} - a_2 x_{i-2} + b_1 m_{i-N-1} + b_2 m_{i-N-2} + [b_1 + b_2] \hat{n}_{i-1} \qquad (24)$$

where $\hat{n}_{i-1}$ is the load estimate for the preceeding sample, and subtracting equation (23) from this equation gives

$$\hat{x}_i - x_i = [b_1 + b_2][\hat{n}_{i-1} - n_i] \quad .$$

Rearrangement yields a recursive relation for updating the load estimate:

$$\hat{n}_i = \hat{n}_{i-1} - K_I[\hat{x}_i - x_i] \qquad (25)$$

While the analytical relation for K_I is

$$K_I = K_c = \frac{1}{b_1 + b_2}$$

the use of K_I as a tuning parameter gives an integral-mode estimation of the load. Moore (17) used this method of disturbance estimation in the development of his first-order analytical predictor for dead time compensation. For the modified Kalman controller, this approach is attractive in that K_I can be tuned such that the undesirable effects of the load estimator upon controller output are lessened for set point changes without sacrificing good disturbance regulation.

This load estimation method was used as the model-error controller in the Kalman controller scheme, and CSMP

simulations for various values of K_I were obtained. The load estimate given by equation (25) was subtracted from the open-loop control action calculated by equation (21). Note that for this approach, the predicted process output is obtained from equation (24) rather than from the disturbance-free model of equation (19). The process was the fifth-order process in equation (22), the sample time was 1.0, and controller constraints were again set at $\pm$ 5. To illustrate programming of the algorithm, a listing of the CSMP program used for set point change simulations is given in Appendix B. Simulation results are presented in Figures 22-25.

Figures 22, 23, and 24 show that for a unit step set point change, the process output is relatively insensitive to changes in K_I. Although calculated values for the integral absolute error (IAE) show slightly better performance for a K_I value equal to 0.5 K_c, the real difference is in the control action. A more desirable controller output is obtained for values of K_I less than the analytical value, with K_I = 0.1 K_c giving the smoothest control action of the three cases studied.

As would be expected, lower values for K_I gave slower load change responses. Figure 25 shows that lowering K_I reduces the rate at which the controller output reaches the value required to cancel the disturbance. The best performance is, of course, obtained with $K_I = K_c$. However, the response for K_I = 0.5 K_c is certainly acceptable, and in

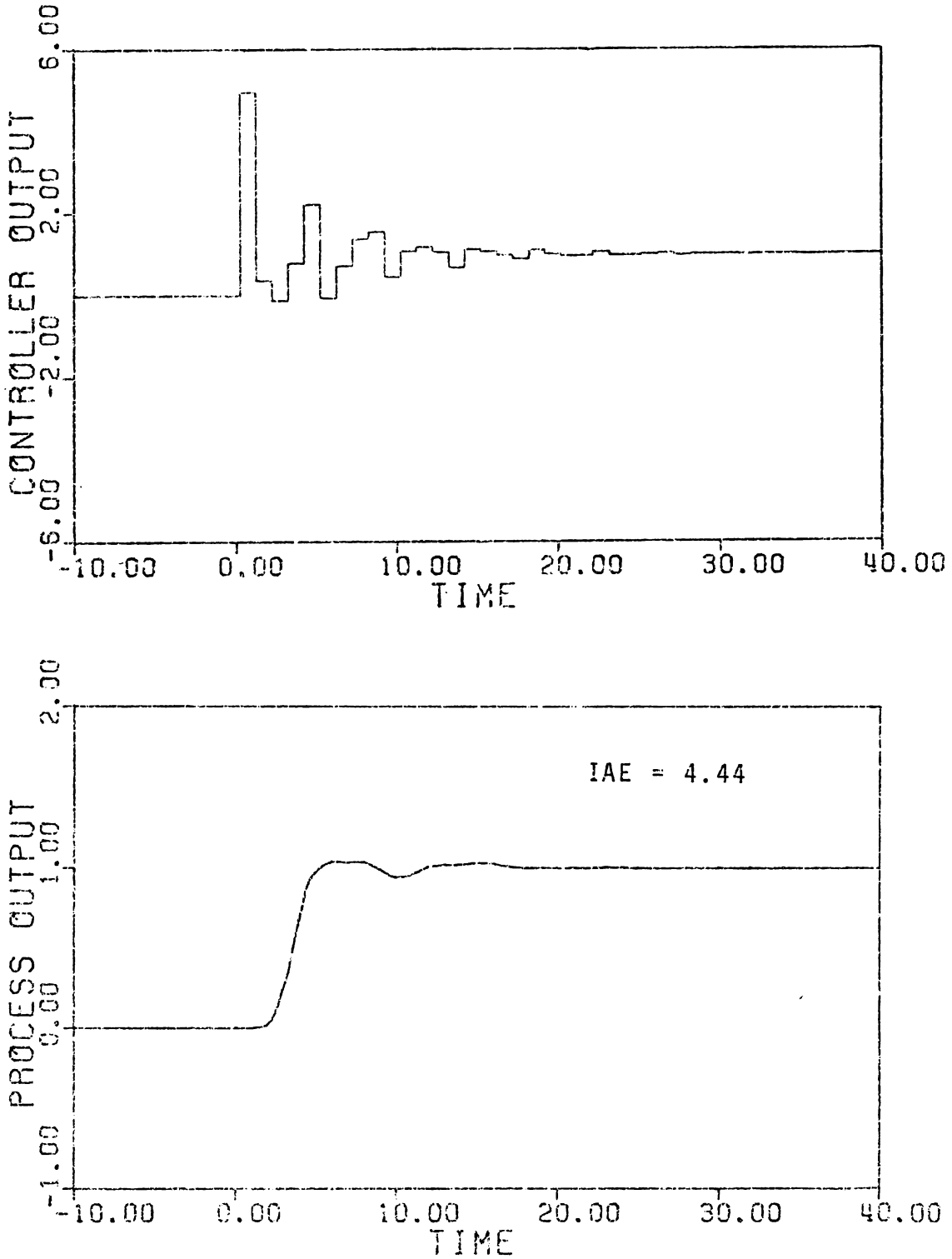


Figure 22. Set Point Change Response for Fifth-Order Process with Modified Kalman Controller for Integral Load Estimation ($K_I = K_c$)

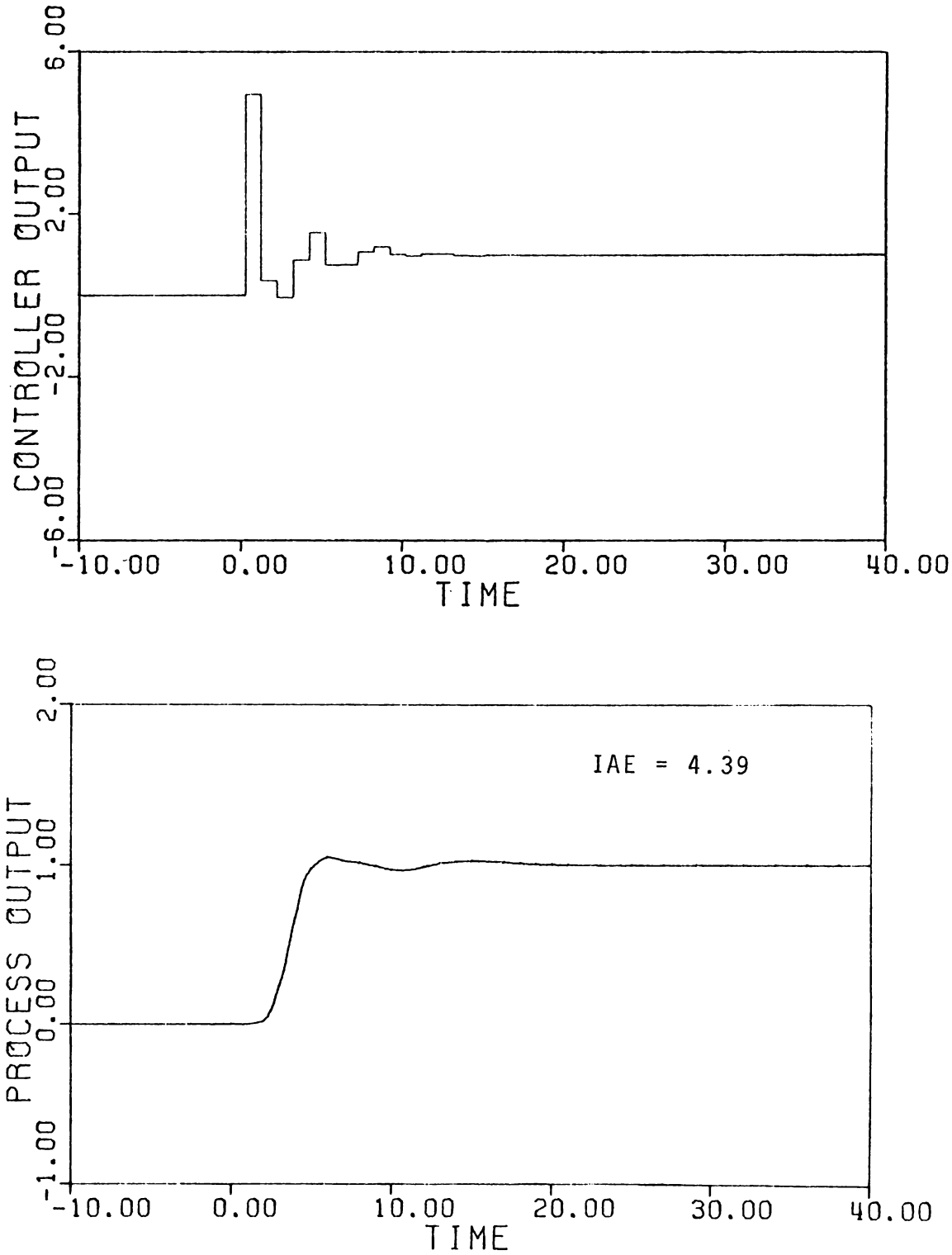


Figure 23. Set Point Change Response for Fifth-Order Process with Modified Kalman Controller for Integral Load Estimation ($K_I = 0.5K_c$)

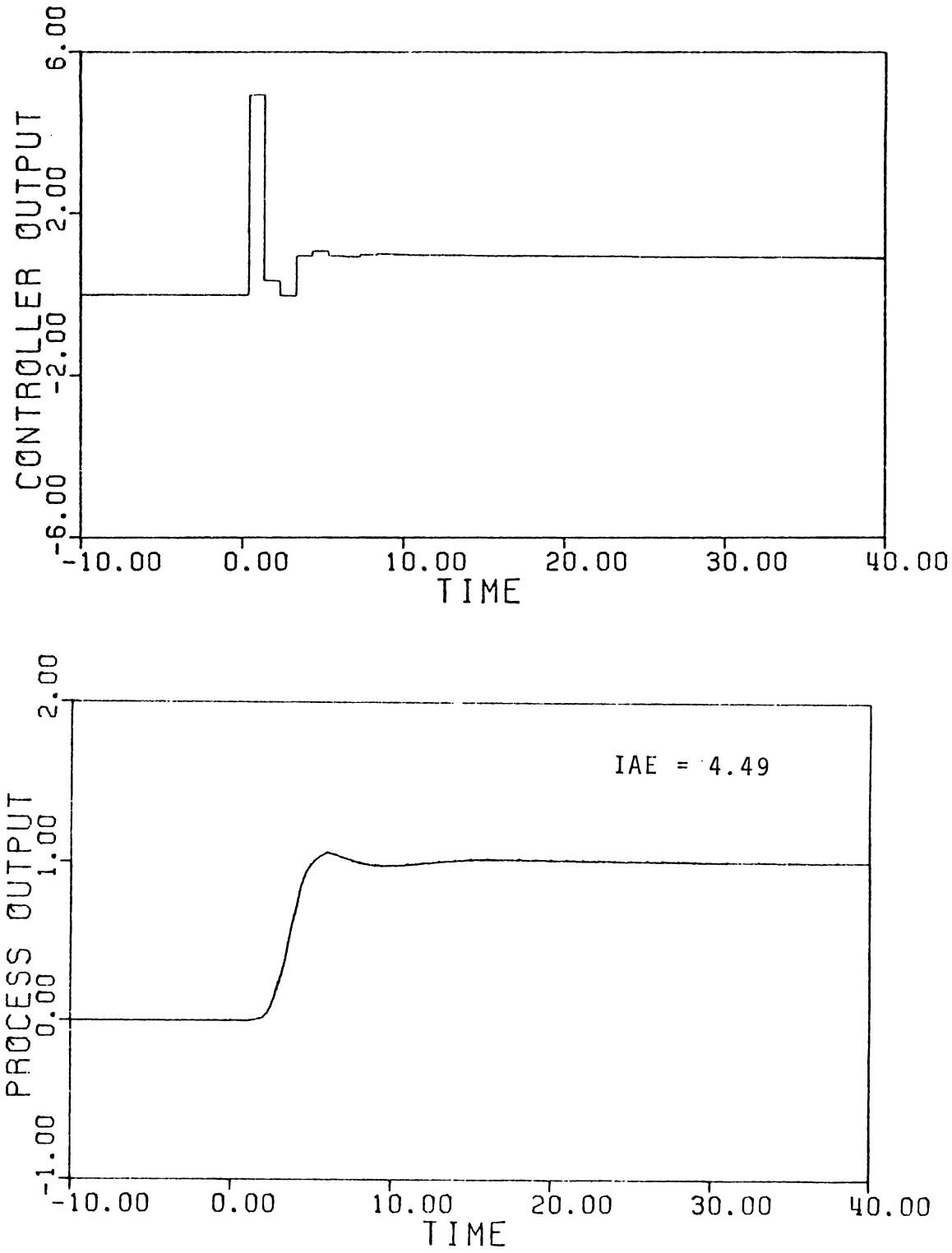


Figure 24. Set Point Change Response for Fifth-Order Process with Modified Kalman Controller for Integral Load Estimation ($K_I = 0.1K_c$)

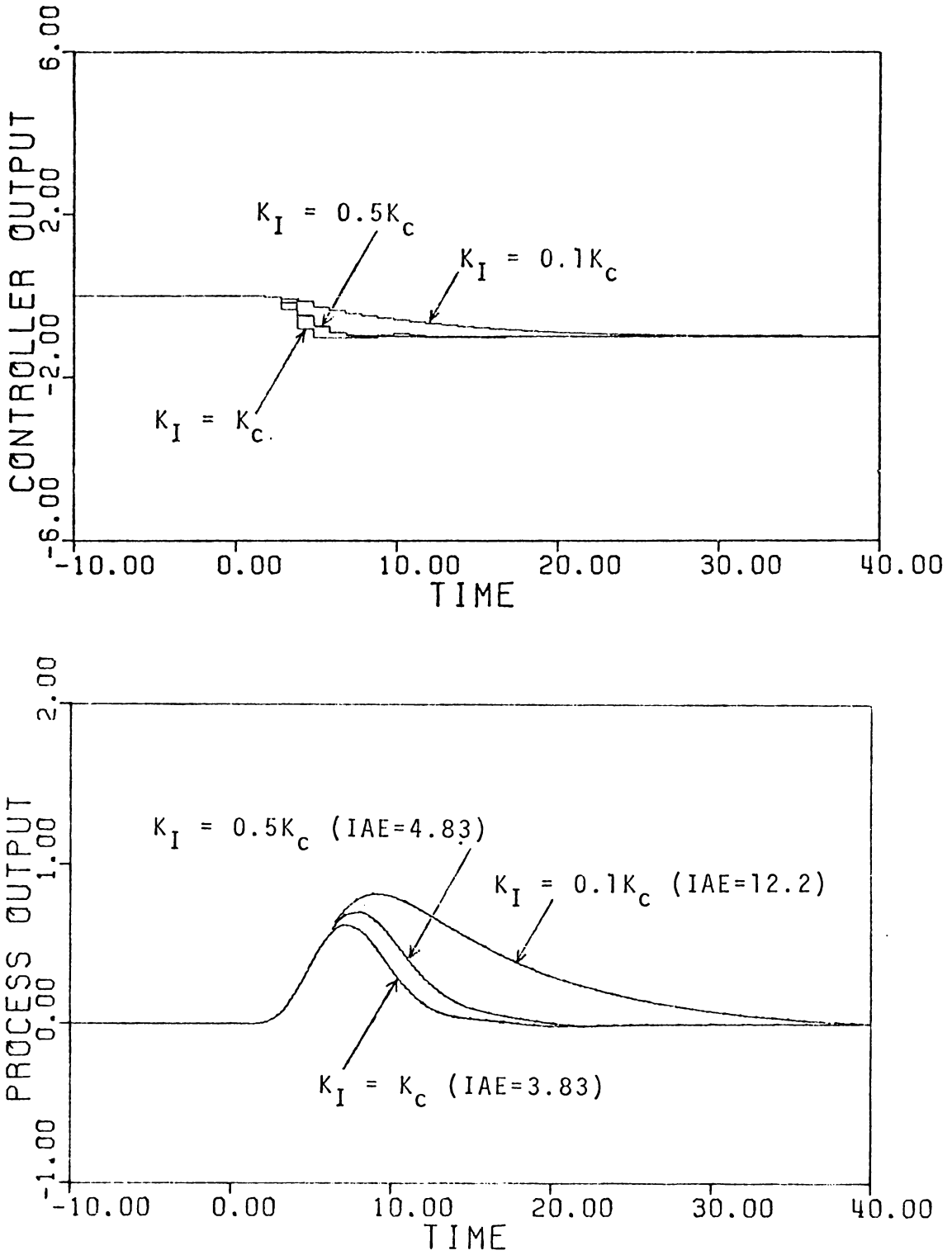


Figure 25. Load Change Response for Fifth-Order Process with Modified Kalman Controller for Integral Load Estimation

view of the corresponding set point change control action, this value gave the best overall results.

For comparison, simulated set point and load change responses were obtained using the discrete proportional-integral (PI) controller shown below.

$$m_i = K_p[e_i + \frac{T}{T_I} \sum_{j=1}^{i} e_j]$$

This is perhaps the most common algorithm in DDC applications today. The PI algorithm was tuned by simulating the digital control loop using the second-order discrete-time model of the fifth-order process. A Pattern search (1) was used to find those values of K_p and T_I which minimized the IAE integral criterion for unit step changes in set point and load.

Figure 26 shows the set point change response for the fifth-order process under discrete PI control. The process output exhibits a largerrise time and requires longer to settle out than that for the modified Kalman controller in Figure 23. In terms of the integral performance criterion, the modified Kalman controller with $K_I = 0.5\ K_c$ gave a 40% improvement over the PI controller.

Similarly, the load change response for PI control in Figure 27 was much slower than the response obtained with the modified Kalman controller. Comparison with the process response in Figure 25 for $K_I = 0.5\ K_c$ shows that while the Kalman controller returns the process to the set point with

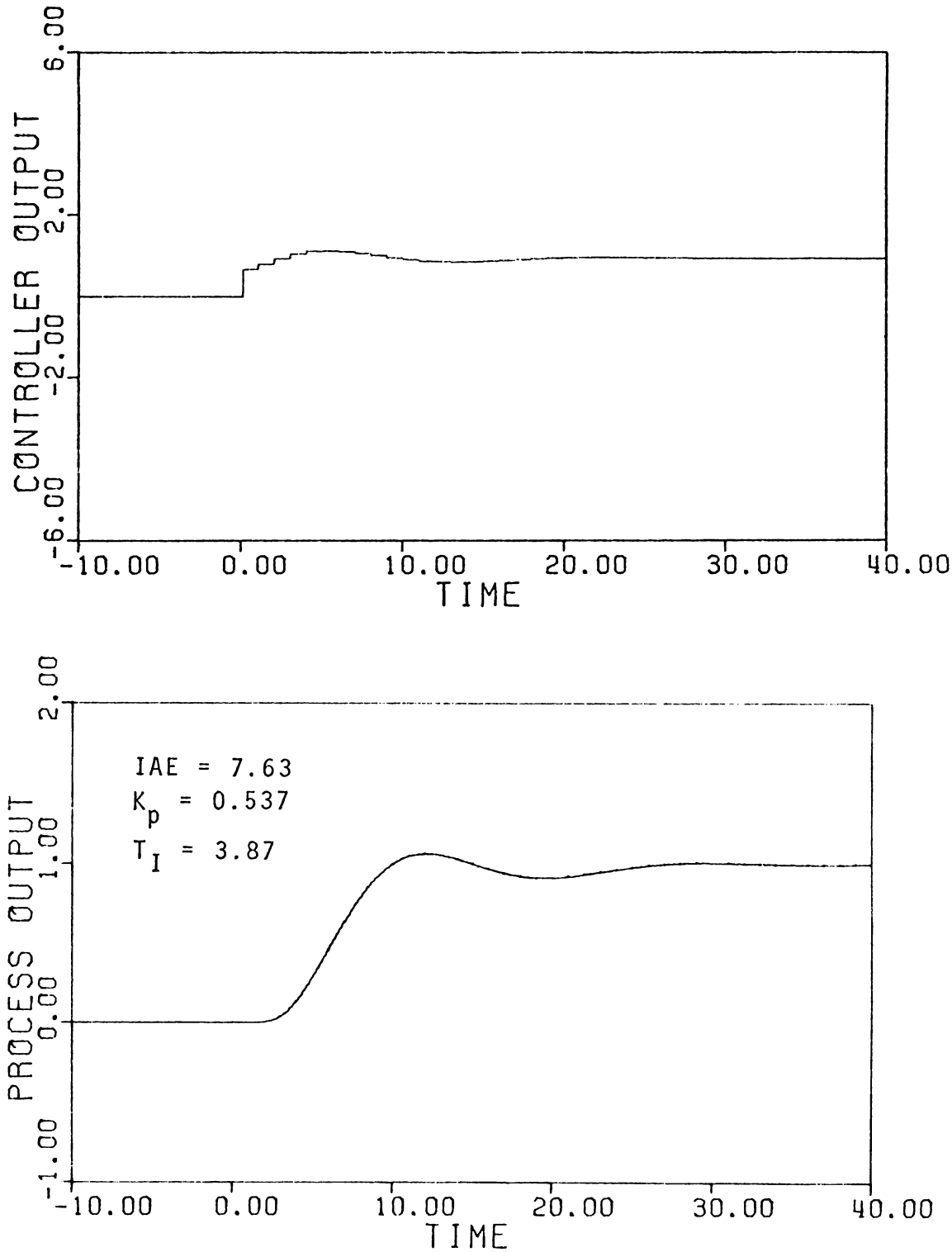


Figure 26. Set Point Change Response for Fifth-Order Process with Discrete PI Control

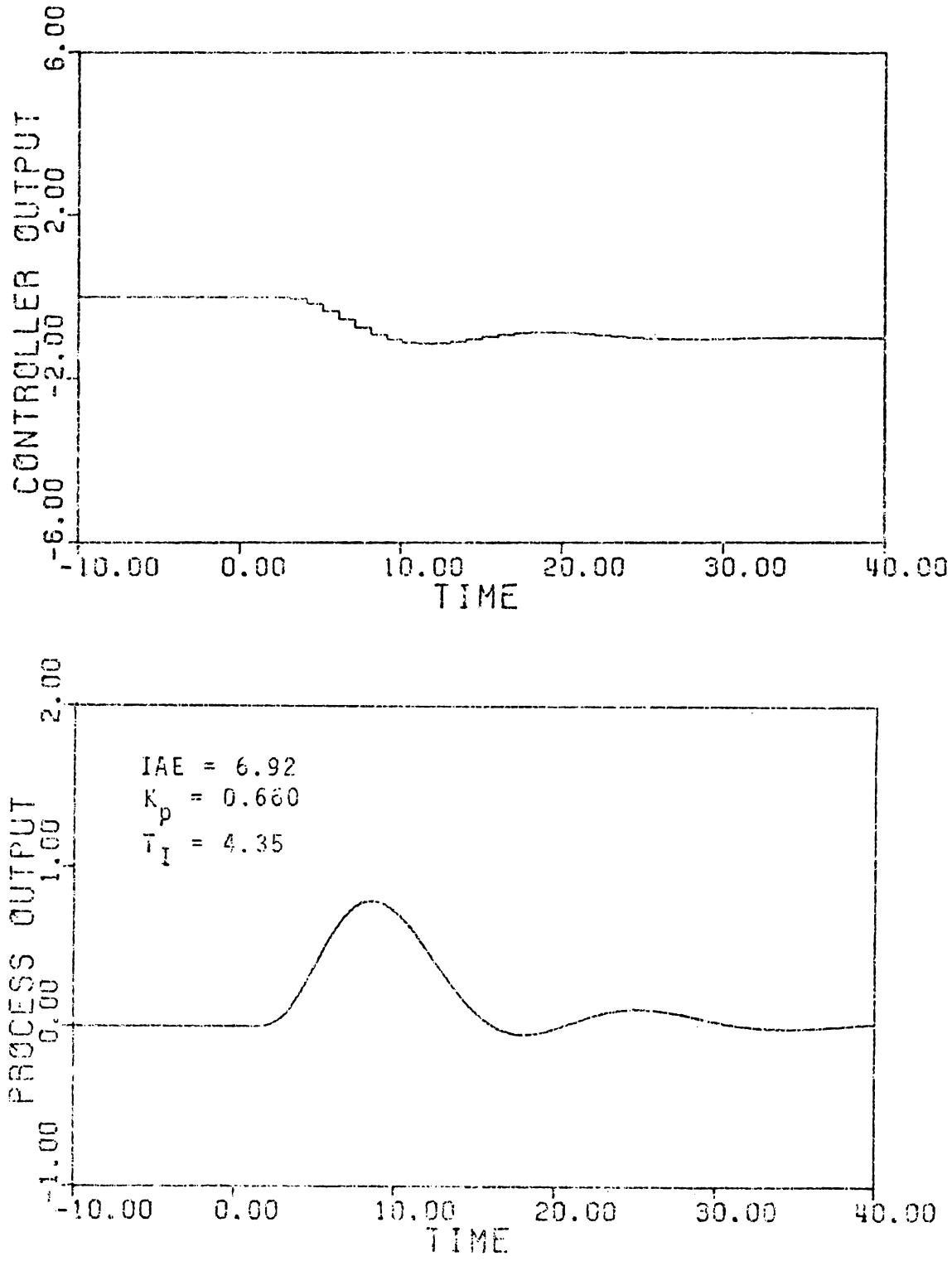
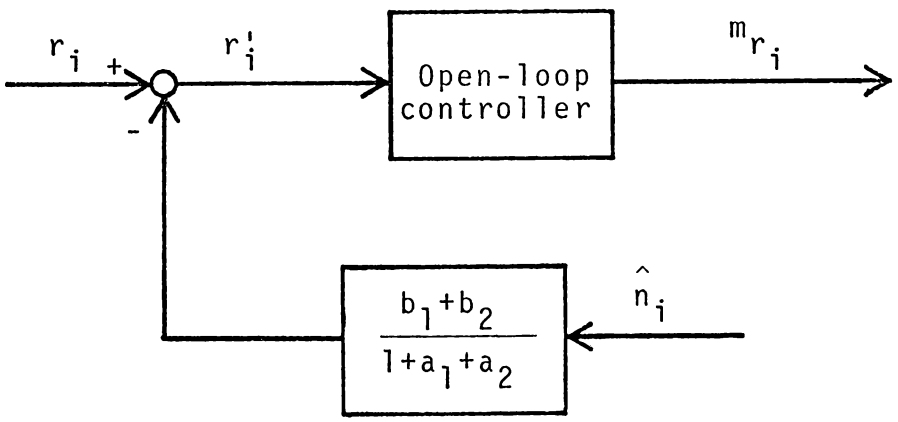


Figure 27. Load Change Response for Fifth-Order Process with Discrete PI Control

negligible overshoot, the process output for PI control is more oscillatory with an IAE about 1.5 times that for the Kalman controller.

A final attempt to further speed up the load response for the modified Kalman controller consisted of the approach diagrammed below:



To take advantage of the open-loop controller's quick response to a set point change, the load estimate of equation (25) was used to reset the set point of the controller. The load estimate was not subtracted from the open-loop control action as before, and the only control action sent to the process was $M_r(z)$. Simulations of this load estimate feedback scheme showed that because of the high sensitivity of the open-loop controller, a K_I value of 0.1 K_c was needed to maintain stability for both set point and load changes. The load change response corresponding to this value of K_I was no better than that for PI control.

CHAPTER IV

CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE WORK

Conclusions

The general conclusion resulting from this work is that a discrete-time approach to direct digital control offers definite advantages over the simple adaptation of continuous control techniques. This work has shown that a single discrete-time model is descriptive of many processes of interest in chemical process modeling. Also, the modification of a controller designed by z-transforms gave an algorithm which promises improved process control. Other conclusions are briefly summarized in the following paragraphs.

A least-squares fit of noise-contaminated process data appears to be inadequate even for extremely low noise levels. The measurement error introduced in independent variables of the model causes biased estimates of the process parameters. The iterative least-squares modeling scheme of Steiglitz and McBride (16) is effective in improving the process parameter estimates resulting from a single least-squares fit of process data and is easily incorporated in a computer program for off-line modeling.

The modified Kalman controller obtained by including controller constraints and revising the load estimation method of Kalman's z-transform controller (26) is a

potentially useful algorithm for direct digital control. Simulation results show that the modified Kalman controller gives significantly improved performance over the popular discrete PI algorithm.

Recommendations

In the modeling of simulated step response data, the process dead time was the least accurate of the parameter estimates. The use of more substantial test inputs might improve the dead time estimate, and the effect of alternative process inputs for open-loop testing (pulse,ramp,etc.) on the resulting model parameters is one area for future study.

Because the discrete-time model is applicable to operating data as well as open-loop response data, the development of an on-line modeling scheme for following changes in process parameters would be a logical extension of this work. The determinant or, more appropriately, the norm of the matrix formed by the linear regression summation terms could be used to judge when significant changes in the process input and output had occurred, and, consequently, if the normal equations were sufficiently independent to allow modeling of the collected data.

An investigation of the effects of modeling errors on the performance of the modified Kalman controller should be carried out. The sensitivity of the algorithm to imperfect knowledge of process parameters would be an important consideration in its application to control of an actual process.

Also, some consideration should be given to tuning the integral mode in the load estimation portion of the algorithm. One possible tuning criterion would be the integral of some function of the set point change controller output and the load change response.

LIST OF REFERENCES

LIST OF REFERENCES

1. Moore, C. F., "Selected Problems in the Design and Implementation of Direct Digital Control," Ph.D. Dissertation, Louisiana State University, 1969.

2. Gallier, P. W. and R. E. Otto, "Self-Tuning Computer Adapts DDC Algorithms," Instrumentation Technology, Vol. 15, No. 2, February 1968, pp. 65-70.

3. Smith, C. L., Digital Computer Process Control, International Textbook Company, Scranton, Pennsylvania, 1972.

4. Kuo, B. C., Analysis and Synthesis of Sampled-Data Control Systems, Prentice-Hall, Englewood Cliffs, N.J., 1963.

5. Tou, J. T., Digital and Sampled-Data Control Systems, McGraw-Hill Book Company, New York, 1959.

6. Ziegler, J. G. and N. B. Nichols, "Optimum Settings for Automatic Controllers," Transactions ASME, Vol. 64, Nov. 1942, pp. 759-765.

7. Miller, J. A., A. M. Lopez, C. L. Smith, and P. W. Murrill, "A Comparison of Controller Tuning Techniques," Control Engineering, Vol. 14, No. 12, Dec. 1967, pp. 72-75.

8. Chiu, K. C., A. B. Corripio, and C. L. Smith, "Process Models for Controller Tuning," Instruments and Control Systems, Vol. 45, No. 1, Jan. 1972, pp. 84-88.

9. Sten, J. W., "Evaluating Second-Order Parameters," Instrumentation Technology, Vol. 17, No. 9, Sept. 1970, pp. 39-41.

10. Meyer, J. R., G. D. Whitehouse, C. L. Smith, and P. W. Murrill, "Simplifying Process Response Approximation," Instruments and Control Systems, Vol. 40, No. 12, Dec. 1967, pp. 76-79.

11. Kalman, R. E., "Design of a Self-Optimizing Control System," Transactions ASME, Vol. 80, No. 1, Feb. 1958, pp. 468-478.

12. Dahlin, E. B., "On-Line Identification of Process Dynamics," IBM Journal, Vol. 11, No. 4, July 1967, pp. 406-426.

13. Conte, S. D., Elementary Numerical Analysis, McGraw-Hill Book Company, New York, 1965.

14. Astrom, K. J. and P. Eykhoff, "System Identification--A Survey," Automatica, Vol. 7, No. 2, March 1971, pp. 123-162.

15. System/360 Scientific Subroutine Package Programmer's Manual, IBM Corporation, 1968, p. 77.

16. Steiglitz, K. and L. E. McBride, "A Technique for the Identification of Linear Systems," IEEE Transactions on Automatic Control, Vol. 10, No. 4, Oct. 1965, pp. 461-464.

17. Moore, C. F., C. L. Smith, and P. W. Murrill, "Improved Algorithm for Direct Digital Control," Instruments and Control Systems, Vol. 43, No. 1, Jan. 1970, pp. 70-74.

18. Smith, O. J. M., "Closer Control of Loops with Dead Time," Chemical Engineering Progress, Vol. 53, No. 5, May 1967, pp. 217-219.

19. Bakke, R. M., "Direct Digital Control with Self-Adjustment for Processes with Variable Dead Time and/or Multiple Delays," 20th Annual ISA Conference and Exhibit, Los Angeles, October 1965.

20. Corripio, A. B. and C. L. Smith, "Computer Simulation to Evaluate Control Strategies," Instruments and Control Systems, Vol. 44, No. 1, Jan. 1971, pp. 87-91.

21. Doss, J. E., "Direct Digital Control of Dead Time Processes," M.S. Thesis, University of Tennessee, 1973.

22. Mosler, H. A., L. B. Koppel, and C. R. Coughanowr, "Process Control by Digital Compensation," AIChE Journal, Vol. 13, No. 4, July 1967, pp. 768-778.

23. Lane, D. W., "Digital Control Algorithm for Slow Sampling Rates," Instrumentation Technology, Vol. 17, No. 7, July 1970, pp. 47-52.

24. Dahlin, E. B., "Designing and Tuning Digital Controllers," Instruments and Control Systems, Vol. 41, No. 6, June 1968, pp. 77-83.

25. Chiu, K. C., A. B. Corripio, and C. L. Smith, "Digital Control Algorithms, Part I," Instruments and Control Systems, Vol. 46, No. 10, Oct. 1973, pp. 57-59.

26. Kalman, R. E., Transactions AIEE, Vol. 73, No. 11, November 1954, pp. 245-247.

27. System/360 Continuous System Modeling Program User's Manual, IBM Corporation, 1969.

APPENDICES

APPENDIX A

DISCRETE-TIME MODELING PROGRAM

```
C*************************************************************************
C
C MODEL -- THIS IS THE MAIN PROGRAM OF A MODELING PACKAGE WHICH PERFORMS
C   A LEAST-SQUARES FIT OF PROCESS DATA USING THE DISCRETE-TIME MODEL
C       X(I) = -A1*X(I-1)-A2*X(I-2)+B1*U(I-N-1)+B2*U(I-N-2)+D.
C   THE METHOD USED IS THE ITERATIVE FILTERING SCHEME PRESENTED BY STEIGLITZ
C   AND MCBRIDE (IEEE TRANS. ON AUTO. CONTROL, OCT. 1965, PP. 461-464) WITH
C   A ONE-DIMENSIONAL SEARCH FOR DEAD TIME.  THE USER MUST SUPPLY A DOUBLE
C   PRECISION SUBPROGRAM INPUT OF THE FORM
C                     SUBROUTINE INPUT(UN,XN,NPTS)
C   WHICH READS IN THE PROCESS INPUT (UN) AND PROCESS OUTPUT (XN) DATA TO
C   BE MODELED.  NPTS IS THE NUMBER OF DATA POINTS.
C   THE FIRST DATA CARD SHOULD CONTAIN VALUES FOR THE VARIABLES NPTS (<500)
C   AND NP IN A (I3,I2) FORMAT.  NP EQUALS 0 IF NO ESTIMATE OF THE PROCESS
C   OFFSET IS DESIRED AND 1 IF THE OFFSET TERM D IS TO BE INCLUDED.  FOLLOWING
C   THIS CARD SHOULD BE THE PROCESS RESPONSE DATA IN THE FORMAT USED IN
C   SUBROUTINE INPUT.  NO OTHER DATA CARDS ARE NECESSARY.
C   THE PROGRAM OUTPUT CONSISTS OF THE NUMBER OF ITERATIONS REQUIRED (ITER),
C   VALUES OF THE MODEL PARAMETERS (A(1),A(2),...,A(5)), THE SUM OF THE SQUARED
C   ERRORS (ESQRD), AND A PRINT-OUT OF THE PROCESS INPUT, PROCESS OUTPUT,
C   MODEL OUTPUT, AND MODEL ERROR.
C
C*************************************************************************
      DOUBLE PRECISION UN(500),XN(500),U(500),X(500),A(5),ERROR,
     1       A1OLD,A2OLD,ESQR0,ESQRD,GAINF,XMIN1,XMIN2,XMOD,OFFSET
      COMMON UN,XN,U,X,NPTS
      IIN = 5
      IOUT = 6
      READ (IIN,1001) NPTS,NP
C
C   READ IN PROCESS INPUT/OUTPUT DATA.
C
      CALL INPUT(UN,XN,NPTS)
      DO 5 I=1,NPTS
      U(I) = UN(I)
      X(I) = XN(I)
    5 CONTINUE
      WRITE (IOUT,1003)
      NP = NP+4
      A1OLD = 1.0D+50
      A2OLD = 1.0D+50
      DO 40 ITER=1,25
      IF (ITER.GT.5) GO TO 25
C
C   PERFORM A LEAST-SQUARES FIT OF THE FILTERED DATA.
C
      CALL ASUBI(0,NP,1,A,ESQR0)
      IE = NPTS-7
      DO 10 N=1,IE
      CALL ASUBI(N,NP,2,A,ESQRD)
      IF (ESQRD.GT.ESQR0) GO TO 20
      ESQR0 = ESQRD
   10 CONTINUE
   20 N = N-1
   25 CALL ASUBI(N,NP,1,A,ESQRD)
C
C   TEST FOR CONVERGENCE OF A(1) AND A(2).
C
```

```
      IF (DABS(A1OLD-A(1)).LT.2.0D-05.AND.DABS(A2OLD-A(2)).LT.2.0D-06)
     1    GO TO 50
      A1OLD = A(1)
      A2OLD = A(2)
C
C   FILTER THE ORIGINAL PROCESS DATA.
C
      GAINF = 1.0+A(1)+A(2)
      X(1) = A(5)/GAINF
      U(2) = GAINF*UN(2)-A(1)*U(1)
      X(2) = GAINF*XN(2)-(A(1)+A(2))*X(1)
      DO 30 IJ=3,NPTS
      U(IJ) = GAINF*UN(IJ)-A(1)*U(IJ-1)-A(2)*U(IJ-2)
      X(IJ) = GAINF*XN(IJ)-A(1)*X(IJ-1)-A(2)*X(IJ-2)
   30 CONTINUE
   40 CONTINUE
      WRITE (IOUT,1012)
      GO TO 60
C
C   PRINT RESULTS.
C
   50 WRITE (IOUT,1011) ITER
   60 WRITE (IOUT,1004) N
      WRITE (IOUT,1005)
      WRITE (IOUT,1006) (A(I),I=1,5)
      WRITE (IOUT,1007)
      J = N+2
      WRITE (IOUT,1008) (UN(I),XN(I),I=1,J)
      J = J+1
      ESQRD = 0.0
      OFFSET = A(5)/(1.0+A(1)+A(2))
      XMIN1 = OFFSET
      XMIN2 = OFFSET
      DO 70 I=J,NPTS
      IN = I-N
      XMOD = -A(1)*XMIN1-A(2)*XMIN2+A(3)*UN(IN-1)
     1       +A(4)*UN(IN-2)+A(5)
      ERROR = XN(I)-XMOD
      ESQRD = ESQRD+ERROR**2
      WRITE (IOUT,1009) UN(I),XN(I),XMOD,ERROR
      XMIN2 = XMIN1
      XMIN1 = XMOD
   70 CONTINUE
      WRITE (IOUT,1010) NPTS,ESQRD
 1001 FORMAT (I3,I2)
 1003 FORMAT (1H1)
 1004 FORMAT (1H0,3HN =,I3)
 1005 FORMAT (1H0,6X,2HA1,14X,2HA2,14X,2HB1,14X,2HB2,14X,1HD)
 1006 FORMAT (1H ,D13.6,4D16.6)
 1007 FORMAT (1H0,27X,7HPROCESS,10X,5HMODEL,/,13X,5HINPUT,10X,6HOUTPUT,
     1        11X,6HOUTPUT,10X,5HERROR,/,6X,4(16H    ----------))
 1008 FORMAT (1H ,5X,2D16.6)
 1009 FORMAT (1H ,5X,4D16.6)
 1010 FORMAT (1H0,14HTOTAL POINTS =,I4,//,24H SUM OF SQUARED ERRORS =,
     1        D13.6)
 1011 FORMAT (1H ,12HITERATIONS =,I3)
 1012 FORMAT (1H ,14HNO CONVERGENCE)
      CALL EXIT
      END
```

```
      SUBROUTINE ASUBI(N,NP,ICALL,A,ESQRD)
C*****************************************************************************
C
C   FOR A GIVEN VALUE OF DEAD TIME, N, THIS SUBPROGRAM SOLVES THE LINEAR
C   REGRESSION NORMAL EQUATIONS FOR THE COEFFICIENTS A(1),...,A(5) OF THE
C   MODEL SHOWN IN MAIN PROGRAM MODEL.  SOLUTION OF THE SET OF SIMULTANEOUS
C   EQUATIONS IS BY PIVOTAL CONDENSATION.  ICALL IS A FLAG VARIABLE WHICH
C   INDICATES THE POSSIBILITY OF SHORTENED CALCULATION OF THE NORMAL EQUATION
C   ELEMENTS.  IF ICALL=2, THE DEAD TIME SEARCH HAS BEEN INITIATED, AND ALL
C   BUT ONE OF THE SUMMATION TERMS MAY BE CALCULATED FROM THOSE REQUIRED FOR
C   THE PRECEEDING DEAD TIME ITERATION.  OTHERWISE, ICALL=1.
C
C*****************************************************************************
      DOUBLE PRECISION UN(500),XN(500),U(500),X(500),A(5),AM(5,6),
     1       AUX(20),FNPRM,PIVOT,ASAVE,AMULT,XMOD,ESQRD,XMIN1,XMIN2
      DOUBLE PRECISION SXX,SXU,SD,SUU,OFFSET
      COMMON UN,XN,U,X,NPTS
      NPRM = NPTS-N-2
      FNPRM = FLOAT(NPRM)
      IB = N+3
C
C   COMPUTE NORMAL EQUATION ELEMENTS.
C
      GO TO (10,40,10),ICALL
   10 DO 20 I=1,20
      AUX(I) = 0.0
   20 CONTINUE
      DO 30 I=IB,NPTS
      IN = I-N
      AUX(1)  = AUX(1) +X(I-1)**2
      AUX(2)  = AUX(2) +X(I-2)*X(I-1)
      AUX(4)  = AUX(4) +U(IN-1)*X(I-1)
      AUX(5)  = AUX(5) +U(IN-1)*X(I-2)
      AUX(6)  = AUX(6) +U(IN-1)**2
      AUX(7)  = AUX(7) +U(IN-2)*X(I-1)
      AUX(9)  = AUX(9) +U(IN-2)*U(IN-1)
      AUX(13) = AUX(13)+U(IN-1)
      AUX(17) = AUX(17)+X(I)*X(I-2)
      AUX(19) = AUX(19)+X(I)*U(IN-2)
      AUX(20) = AUX(20)+X(I)
   30 CONTINUE
      GO TO 60
   40 IN = NPTS-N
      AUX(1)  = AUX(1) -X(N+1)**2
      AUX(2)  = AUX(2) -X(N)*X(N+1)
      AUX(4)  = AUX(7) -U(1)*X(N+1)
      AUX(5)  = AUX(8) -U(1)*X(N)
      AUX(6)  = AUX(6) -U(IN)**2
      AUX(7)  = AUX(19)-U(IN-1)*X(NPTS)
      AUX(9)  = AUX(9) -U(IN-1)*U(IN)
      AUX(13) = AUX(13)-U(IN)
      AUX(17) = AUX(17)-X(N+2)*X(N)
      AUX(20) = AUX(20)-X(N+2)
      AUX(19) = 0.0
      DO 50 I=IB,NPTS
      IN = I-N
      AUX(19) = AUX(19)+X(I)*U(IN-2)
   50 CONTINUE
   60 IN = NPTS-N
      AUX(3)  = AUX(1) +X(N+1)**2-X(NPTS-1)**2
      AUX(8)  = AUX(4) +U(1)*X(N+1)-U(IN-1)*X(NPTS-1)
      AUX(10) = AUX(6) +U(1)**2-U(IN-1)**2
      AUX(11) = AUX(20)+X(N+2)-X(NPTS)
      AUX(12) = AUX(11)+X(N+1)-X(NPTS-1)
      AUX(14) = AUX(13)+U(1)-U(IN-1)
```

```
      AUX(15) = FNPRM
      AUX(16) = AUX(2) +X(NPTS)*X(NPTS-1)-X(N+2)*X(N+1)
      AUX(18) = AUX(7) +X(NPTS)*U(IN-1)-U(1)*X(N+2)
      IF (NP.LT.5) GO TO 70
      GO TO 90
   70 DO 80 I=1,5
      AUX(I+10) = 0.0
   80 CONTINUE
      AUX(20) = 0.0
      A(5) = 0.0
   90 I = 1
      DO 100 IC=1,5
      AM(IC,6) = AUX(IC+15)/FNPRM
      DO 100 IR=1,IC
      AM(IR,IC) = AUX(I)/FNPRM
      AM(IC,IR) = AM(IR,IC)
      I = I+1
  100 CONTINUE
C
C SOLUTION OF THE NORMAL EQUATIONS...
C
      IE = NP-1
      DO 150 IELIM=1,IE
      IB = IELIM+1
C
C   LOCATE THE LARGEST COLUMN MEMBER FOR USE AS THE
C   PIVOT ELEMENT FOR THIS ELIMINATION STEP.
C
      PIVOT = AM(IELIM,IELIM)
      IROW = IELIM
      DO 110 IR=IB,NP
      IF (DABS(PIVOT).GT.DABS(AM(IR,IELIM))) GO TO 110
      PIVOT = AM(IR,IELIM)
      IROW = IR
  110 CONTINUE
C
C   INTERCHANGE ROWS, IF NECESSARY.
C
      IF (IROW.EQ.IELIM) GO TO 130
      DO 120 IC=IELIM,6
      ASAVE = AM(IELIM,IC)
      AM(IELIM,IC) = AM(IROW,IC)
      AM(IROW,IC) = ASAVE
  120 CONTINUE
C
C   PERFORM ELIMINATION.
C
  130 DO 140 IR=IB,NP
      AMULT = AM(IR,IELIM)/AM(IELIM,IELIM)
      DO 140 IC=IB,6
      AM(IR,IC) = AM(IR,IC)-AMULT*AM(IELIM,IC)
  140 CONTINUE
  150 CONTINUE
C
C   CALCULATE MODEL COEFFICIENTS BY BACK SUBSTITUTION.
C
      DO 160 IR=1,NP
      A(IR) = AM(IR,6)/AM(IR,IR)
  160 CONTINUE
      IN = NP+1
      DO 170 I=1,IE
      IR = NP-I
      DO 170 J=1,I
      IC = IN-J
```

```
      A(IR) = A(IR)-AM(IR,IC)/AM(IR,IR)*A(IC)
  170 CONTINUE
      A(1) = -A(1)
      A(2) = -A(2)
C
C   COMPUTE THE SUM OF THE SQUARED ERRORS.
C
      SXX = AUX(1)+X(NPTS)**2-X(N+2)**2+A(2)*A(2)*AUX(3)
     1  +2.0*A(2)*AUX(17)+A(1)*A(1)*AUX(1)+(2.0*A(1)*A(2)*AUX(2)
     2  +2.0*A(1)*AUX(16))
      SXU = (A(3)*AUX(18)+A(2)*A(4)*AUX(8))+(A(4)*AUX(19)
     1  +A(2)*A(3)*AUX(5))+(A(1)*A(3)*AUX(4)+A(1)*A(4)*AUX(7))
      SD = A(5)*(AUX(20)+A(2)*AUX(12)+A(1)*AUX(11)+(-A(3)*AUX(13)
     1  -A(4)*AUX(14)-0.5*A(5)*ENPRM))
      SUU = A(3)*A(3)*AUX(6)+A(4)*A(4)*AUX(10)+2.0*A(3)*A(4)*AUX(9)
      ESQRD = (SXX+SUU)-2.0*(SXU+SD)
      RETURN
      END
```

```
      SUBROUTINE INPUT(UN,XN,NPTS)
C*************************************************************************
C
C   THIS IS AN EXAMPLE OF THE USER-SUPPLIED SUBPROGRAM FOR READING IN PROCESS
C   INPUT/OUTPUT DATA FOR THE DISCRETE-TIME MODELING PACKAGE.
C
C*************************************************************************
      DOUBLE PRECISION UN(500),XN(500)
      IIN = 5
      DO 10 I=1,NPTS
      READ (IIN,1001) UN(I),XN(I)
   10 CONTINUE
 1001 FORMAT (2F10.6)
      RETURN
      END
```

APPENDIX B

CSMP SIMULATION PROGRAM

```
**************************************************************************************
*
* THIS IS A CSMP PROGRAM FOR SIMULATING DIRECT DIGITAL CONTROL OF A
* FIFTH-ORDER PROCESS USING THE MODIFIED KALMAN CONTROLLER.
*
* THE INITIAL SECTION INITIALIZES PAST VALUES OF CONTROL ACTION, MODEL
* OUTPUTS, AND THE LOAD ESTIMATE TO ZERO.  ALSO, MODEL PARAMETERS ARE
* DEFINED AND CONTROLLER CONSTANTS ARE CALCULATED.
*
* THE DYNAMIC SECTION CONTAINS THE CONTROL LOOP SIMULATION.  IT CONSISTS
* OF TWO PARTS: (1)THE DISCRETE-TIME CONTROLLER AND (2)THE CONTINUOUS
* PROCESS AND THE ZERO-ORDER HOLD.  AT EACH SAMPLE TIME, THE MINIMUM-
* PHASE MODEL OUTPUT XZM IS CALCULATED AND USED IN THE CALCULATION OF MR,
* THE OPEN-LOOP CONTROL ACTION.  THIS CONTROLLER OUTPUT IS THEN CONSTRAINED
* TO AN ABSOLUTE VALUE OF 5.0.  NEXT, THE NON-MINIMUM PHASE MODEL OUTPUT
* XMOD IS EVALUATED AND USED WITH THE SAMPLED PROCESS OUTPUT XZ TO DETERMINE
* THE LOAD ESTIMATE NEST.  THIS LOAD ESTIMATE IS SUBTRACTED FROM MR
* AND THE RESULT CONSTRAINED TO GIVE MZ(1), THE MODIFIED KALMAN CONTROLLER
* OUTPUT. PAST CONTROL ACTIONS ARE MAINTAINED IN A RUNNING STORAGE TABLE
* USING THE MZ VECTOR.
* MZ(1) IS INPUT TO A ZOH TO YIELD THE CONTINUOUS CONTROL ACTION TO THE
* PROCESS. XS IS THE CONTINUOUS PROCESS OUTPUT.
*
**************************************************************************************
/     DIMENSION MZ(30)
FIXED NM,IE,I,J
INITIAL
NOSORT
      NS = 0.0
      IAE = 0.0
      MR1 = 0.0
      MR2 = 0.0
      XZM = 0.0
      XZM1 = 0.0
      XZM2 = 0.0
      XZ = 0.0
      XZ1 = 0.0
      XZ2 = 0.0
      RZ = 0.0
      SAMPT = 1.00
      DLAYX = SAMPT*1.0
      NM = 2
      A1 = -0.134467E+1
      A2 = 0.488763E+0
      B1 = 0.380643E-1
      B2 = 0.103196E+0
      C1 = 1.0/(B1+B2)
      C2 = (A2*(B2+A1*B1)-A1*A1*B2)/(B2*(B2-A1*B1)+A2*B1*B1)
      C4 = A1-C2*B1
      C3 = C1*(1.0-C4)*(A1+A2)-C2-C1*C4
      KI = 0.5*C1
      NEST1 = 0.0
      IE = NM+3
      DO 20 I=1,IE
      MZ(I) = 0.0
   20 CONTINUE
```

```
DYNAMIC
NOSORT
      RS = STEP(0.0)
      T = IMPULS(0.0,SAMPT)
      IF (T.NE.1.0) GO TO 10
      IF (KEEP.NE.1) GO TO 10
      RZ = RS
      XZ = XS
      IAE = IAE+ABS(RZ-XZ)*SAMPT
      XZM = -A1*XZM1-A2*XZM2+B1*MR1+B2*MR2
      MR = C1*RZ+C2*XZM+C3*XZM1+C4*MR1
      IF (MR.GT.5.0) MR=5.0
      IF (MR.LT.-5.0) MR=-5.0
      XMOD = -A1*XZ1-A2*XZ2+B1*MZ(IE-1)+B2*MZ(IE)+(B1+B2)*NEST1
      NEST = NEST1-KI*(XMOD-XZ)
      EI = XMOD-XZ
      MZ(1) = MR-NEST
      IF (MZ(1).GE.5.0) MZ(1)=5.0
      IF (MZ(1).LE.-5.0) MZ(1)=-5.0
      DO 30 I=2,IE
      J = IE-I+2
      MZ(J) = MZ(J-1)
   30 CONTINUE
      MR2 = MR1
      MR1 = MR
      XZM2 = XZM1
      XZM1 = XZM
      XZ2 = XZ1
      XZ1 = XZ
      NEST1 = NEST
   10 CONTINUE
      MS = ZHOLD(T,MZ(1))
      INPUT = MS+NS
      OUT1 = REALPL(0.0,1.0,INPUT)
      OUT2 = REALPL(0.0,1.0,OUT1)
      OUT3 = REALPL(0.0,1.0,OUT2)
      OUT4 = REALPL(0.0,1.0,OUT3)
      XS1 = REALPL(0.0,1.0,OUT4)
      XS = DELAY(100,DLAYX,XS1)
TERMINAL
NOSORT
      WRITE (6,1001) IAE
 1001 FORMAT (1H ,E13.5)
TIMER FINTIM=40.0,OUTDEL=0.1,DELT=0.10
METHOD RKSFX
END
STOP
ENDJOB
```

VITA

Dennis K. Jones was born in Kingsport, Tennessee, on October 6, 1949. Attending elementary and high school in Kingsport, he graduated from Sullivan High School in 1967. The following September, he entered The University of Tennessee, and graduated with a Bachelor of Science in Chemical Engineering in August, 1972. Entering The University of Tennessee Graduate School that fall, he received the Master of Science in Chemical Engineering in 1974.

He is married to the former Ellen Sue Williams.