

INVESTIGATION OF METHODS FOR IDENTIFICATION
OF DISCRETE AND CONTINUOUS LINEAR SYSTEMS

by

David R. Hargreaves

A DISSERTATION
IN THE
FACULTY OF ENGINEERING

Presented in partial fulfilment of the requirements
for the Degree of
MASTER OF ENGINEERING

at

Sir George Williams University
Montreal, Canada

March, 1973



TABLE OF CONTENTS

	<u>page</u>
LIST OF IMPORTANT ABBREVIATIONS AND SYMBOLSviii

CHAPTER 1

INTRODUCTION

1.1 General Introduction to the Identification Problem	1
1.2 Techniques for Solving the Identification Problem	5
1.3 Previous Development of the Identification Problem	9
1.4 Scope of the Thesis	12

CHAPTER 2

STATEMENT OF THE PROBLEM

2.1 Discrete Time Systems	14
2.2 Continuous Time Systems	16

CHAPTER 3

DESCRIPTION OF SELECTED METHODS

3.1 Continuous System Model: Application of Kiefer-Wolfowitz Minimization	18
3.2 Computational Aspects of Methods Using Discrete Models	26
3.2.1 A stochastic approximation method using adaptive filtering	26
3.2.2 An adaptive recursive least-squares identification algorithm [28]	29
3.2.3 Fletcher-Powell Minimization [41]	31
3.2.4 Aström's technique for identification (Maximum Likelihood Method)	34

CHAPTER 4

TEST EXAMPLES

4.1	An Algorithm Using Kiefer-Wolfowitz Minimization Method	40
4.2	On-Line Stochastic Approximation Method (Scalar Gain)	59
4.2.1	Analog Simulation of Process to be Identified	60
4.2.2	Digital Computer Implementation	71
4.3	Implementation of Adaptive Recursive-Least- Square Algorithm (Matrix Gain)	75
4.4	Aström's Maximum Likelihood Technique	79
4.5	Algorithm Using Fletcher-Powell Minimization	83
4.6	Electric Power System Load Modelling	89

CHAPTER 5

CONCLUSIONS

5.1	Summary	103
5.2	Concluding Remarks	106
REFERENCES		109

APPENDIX

K.W.#4	A1
SGNA	A4
SGN	A7
TESPAN	A10
MXG	A14
IDEN1 (ASTRÖM)	A18
IDEN4 (FLETCHER/POWELL)	A29
PMXG	A32
GAUSSS	A37
RRANDU	A37

LIST OF FIGURES

<u>FIGURE</u>		<u>page</u>
1.1	Basic representation of the process parameter problem	2
1.2	Two basic identification schemes	4
3.1	Kiefer-Wolfowitz technique using hybrid computer	22
3.2	Gradient approximation by parameter perturbation	25
4.1	Sequence $u(t)$ and $e(t)$ generated by digital computer	42
4.2	Kiefer-Wolfowitz hybrid implementation ($J=\epsilon^2$)	45
4.3	Analog model used for Kiefer-Wolfowitz algorithm ($J=\epsilon^2$)	46
4.4	Estimate of "a" parameter only, b and c fixed at nominal value	48
4.5	Estimate of both "a" and "b", c fixed at nominal value	49
4.6	Kiefer-Wolfowitz hybrid implementation ($J=\int \epsilon^2 dt$)	53
4.7	Digital flowchart, Kiefer-Wolfowitz algorithm ($J=\int \epsilon^2 dt$)	54
4.8	Analog model and logic circuit used for Kiefer-Wolfowitz algorithm ($J=\int \epsilon^2 dt$)	56
4.9	Estimation of "a" and "b" parameters	58
4.10	Continuous representation of the discrete model	61
4.11	Hybrid implementation of Panuska's stochastic approximation method	62
4.12	Digital program flowchart: scalar gain	65
4.13	Analog computer circuit for use with scalar gain method	66

<u>FIGURE</u>		<u>page</u>
4.14	Parameter estimates using scalar gain stochastic algorithm	68
4.15	Definition of $u(T, \hat{T})$ in terms of actual and normal temperatures	92
4.16	x_{p3} estimate errors, all other parameters held at true values	99

LIST OF TABLES

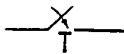
<u>TABLE</u>	<u>page</u>
1 Parameter Estimates for Second Order Model: Scalar Gain Algorithm	73
2 Parameter Estimates for Second Order Model; Scalar Gain Algorithm	74
3 Estimates of Parameters for Second Order Model (Matrix Gain)	78
4 Parameter Estimates After Each Iteration Using Aström's Maximum Likelihood Method	81
5 Final Parameter Estimates Resulting from Application of Aström's Technique	82
6 Parameter Estimates Resulting from Fletcher- Powell Minimization	86
7 Temperature Effect Input u for January 1972	96
8 Estimation of Parameters for Power Load Problem, Comparing Matrix Gain and Fletcher- Powell Techniques	97

LIST OF IMPORTANT ABBREVIATIONS AND SYMBOLS

a_i, b_i, c_i	true system parameters to be identified
$\hat{a}_i, \hat{b}_i, \hat{c}_i$	estimates of true parameter values
A/D	analog to digital signal converter
$A(z)$	polynomial in z of order k with coefficients a_1, a_2, \dots, a_k
$B(z)$	polynomial in z of order k with coefficients b_0, b_1, \dots, b_k
$C(z)$	polynomial in z of order k with coefficients c_1, c_2, \dots, c_k
C_i	carry-in signal to BCD counter
CLR	clear input for initiating BCD counter
CL	control line from 640 digital computer to 680 analog
\underline{d}	replacement parameter vector for $\underline{\theta}$ when converting to continuous system
$D(\%)$	per cent deviation of starting parameter values from true values
D/A	digital to analog signal converter
$e(t)$	random noise sequence
$E[\cdot]$	expected value notation
$f_1(nT)$	time domain function used for calculation of continuous transfer function between $u(t)$ and $y(t)$
$f_2(nT)$	used for calculation of transfer function between $e(t)$ and $y(t)$
g_i	gradient of residual, ϵ , with respect to parameter estimate x_i
$G_n(n, \text{VAR})$	scalar algorithm gain where n is the iteration number
H	matrix used in Fletcher-Powell algorithm, converging to Hessian matrix of cost function

HOLD	logic signal forcing the analog computer into the "hold" mode
IC	logic signal forcing the analog computer into the "initial condition" mode
I_n	information processing bloc
$J(\underline{\theta})$	cost function to be minimized with respect to $\underline{\theta}$
$\underline{J}_{\underline{\theta}}$	gradient (vector) of J with respect to $\underline{\theta}$
$J_{\theta\theta}$	matrix of partial derivatives of J with respect to $\underline{\theta}$
k_p	number of poles in the denominator of a function (ratio of two polynomials)
$K_1(s), K_2(s)$	transfer functions required for continuous system simulation with inputs $e(s)$, $u(s)$ and output $y(s)$
$L(\cdot)$	maximum likelihood function
$M(\underline{x})$	regression function of \underline{x} which has a maximum at unknown point $\underline{\theta}$
M_d	model bloc
N	number of data input/output samples used for identification
OP	logic signal forcing the analog computer into the "operate" mode
$p(\underline{x})$	density function of random variable \underline{x}
$P(x)$	distribution function of random variable \underline{x}
$q(t)$	power load model output (KWH)
$qq(t)$	modified power load output with known components removed (KWH)
$\underline{Q}(t)$	vector containing past k values of $q(t)$
r	scalar function used in calculation of step size, Fletcher-Powell minimization
$R(s), W(s)$	analog signals used when implementing Kiefer-Wolfowitz method on a hybrid computer
s	complex frequency (Laplacian notation)
$s(i)$	the ith component of step size vector using Fletcher-Powell minimization

SL	sense line from analog computer logic to the digital computer
t	independent variable time (computer seconds)
T	period between sampling of analog signals
T31,TIC31	"Track" and "Track Initial Condition" logic signals for analog amplifier 31
T_e, \hat{T}_e	actual weather temperature and normal predicted temperature
$u(t)$	known input to system under study
\underline{U}	step in parameter estimate vector to be used for Davidon cubic interpolation
v	the squared length of vector \underline{S}
\underline{V}	observation vector consisting of outputs, inputs and noise signals of system to be identified
$\hat{\underline{V}}_n$	estimated observation vector where the noise components are replaced by their estimates
$VAR(\epsilon)$	estimate of the variance of computed error ϵ
w	scalar function used in Fletcher-Powell minimization
\underline{x}	parameter estimation vector
\underline{x}_p	estimate of the periodic parameters for power load system modelling
\underline{x}_{PIN}	initial starting vector for \underline{x}_p
$\underline{X}(t)$	general state vector for state space representations
$y(t)$	output of system to be identified (also residual component of the power load for power system modelling)
$y_p(t)$	periodic component of power load (KWH)
$\dot{y}(t)$	derivative of y with respect to time t
z	operator when using discrete time notation; $z^{-1}y(t) = y(t-1)$
$Z[\cdot]$	Z-transform operator

$\delta y, \delta u$	difference values of $y(t)$ and $u(t)$ between values at time, t , and values at time, $t - 24$
ε	computed error signal corresponding to $e(t)$ when correct parameter estimates are obtained
λ	scalar corresponding to standard deviation of error signal $e(t)$
$\underline{\theta}$	true parameter vector
\in	notation for "belonging to a set"
$\gamma(n)$	algorithm gain sequence when implementing Kiefer-Wolfowitz minimization
$\alpha(n)$	scalar function of n used for perturbation of parameter estimates
σ_i	deviation of errors of parameter estimate x_i
$\Gamma(n)$	matrix converging to covariance matrix of parameter estimates as $n \rightarrow \infty$
ρ	scalar used in Fletcher-Powell minimization procedure
$\nabla \underline{J}(\underline{x})$	gradient vector of cost function J with respect to \underline{x}
Ω	rectangular set known to contain $\underline{\theta}$
ω^i	lower bound imposed on parameter estimate x^i
β^i	upper bound imposed on parameter estimate x^i
$\phi_i(t)$	the i th periodic input to power load system model
$\underline{\phi}(t)$	vector containing the periodic inputs $\phi_i(t)$
$\Theta(s), \Phi(s)$	transfer functions used for analog simulation to generate relationship between $u(s)$, $e(s)$ and $y(s)$
$\eta(t)$	matrix of sequence of vectors \underline{v}_i , $i=1, \dots, n$
$\Psi(t)$	matrix containing past k values of vector $\underline{\phi}(t)$
$\underline{\Delta}$	difference vector $(\underline{x}(i+1) - \underline{x}(i))$ when using Fletcher-Powell minimization
$\mathcal{L}[\cdot]$	Laplace transform operator
	digital sampler with sampling period T

ABSTRACT

The identification problem, as defined in this thesis, is the determination of unknown parameters of a dynamic system from noisy input-output observations when the system topology is assumed known. This thesis deals with the use of both continuous and discrete techniques for identification of linear systems. Since numerous proposed techniques are available, the scope of the study has been limited to the presentation of selected methods covering a variety of approaches to the problem.

Methods involving both analog/digital and all-digital computation are studied and applied to simulated system data. The chosen application problem is a second-order, single-input, single-output system. Comparisons are made with respect to relative ease of use, accuracy of estimates and required computing time. Proposals are made for possible improvements in the algorithms used.

Methods are also extended to electric power systems load modelling where large additional unknown periodic inputs are introduced. Estimation results based on simulated load data are presented for both a stochastic approximation technique and the maximum likelihood approach using the Fletcher-Powell method of minimization.

ACKNOWLEDGEMENTS

The author is deeply indebted to Dr. V. Panuska for his guidance and infinite patience throughout the course of this work.

Thanks are due to Ann Mylchreest for an excellent job of typing the following text.

Finally, for their moral support and understanding, the author wishes to thank his close friends, especially his mother, Kathleen Hargreaves.

CHAPTER 1

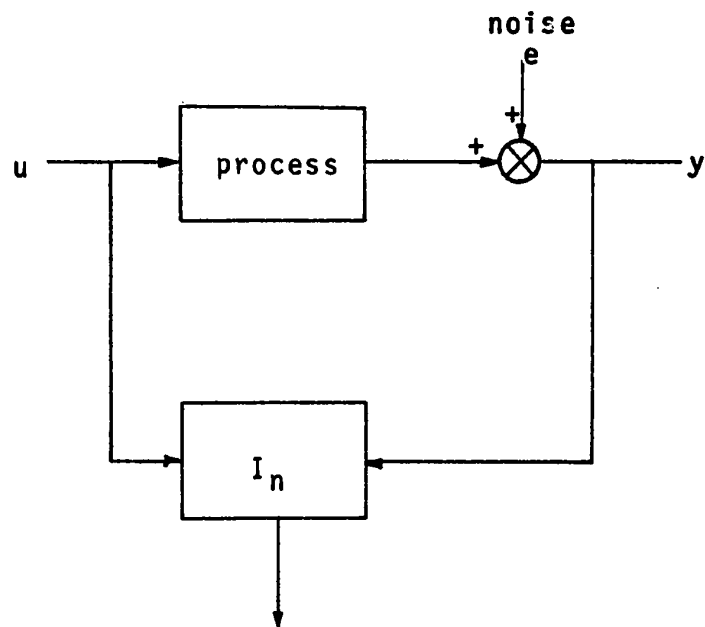
INTRODUCTION

1.1 General Introduction to the Identification Problem

Generally speaking, the goal of parameter estimation is to obtain knowledge about a physical system under normal operating conditions. This can be achieved by either using signals already present in the system, or by introducing test signals of a special class with a sufficient level. A simple representation of the problem is shown in Fig. 1.1, where the system is called a "process". The output y is contaminated with noise (for example, measurement errors). From measurements on u and y , knowledge about the system has to be derived.

The importance of modelling, identification and parameter estimation in automatic control is an accepted fact. One of the most pressing needs in control engineering is related to the modelling issue, namely, how accurate should the model be? how accurate should the model of uncertainties be? and how should the performance index be defined to consolidate design specifications and model inaccuracy? The key role of the control engineer is to understand the physics of the problem and to be able to translate it into accurate quantitative mathematical models.

In order to apply linear stochastic control theory, the process to be controlled should be described in terms of linear differential or difference equations driven by the input signals and disturbances. Samples of the process inputs and outputs can



Knowledge about process

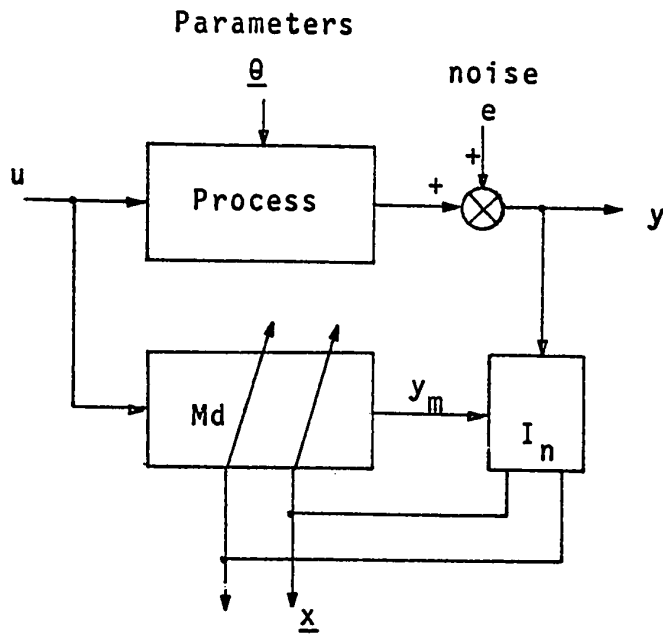
FIG. 1.1 Basic representation of the process parameter problem.

be used for determining such models.

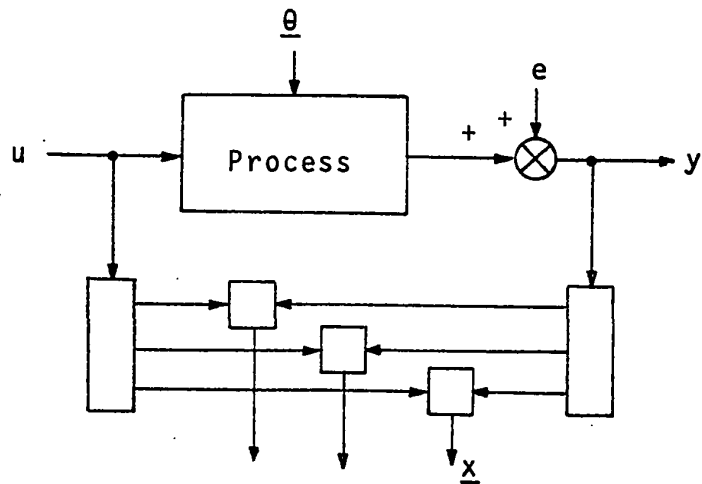
One might define general identification of a process as the determination of the topology of the process, considering it as a celebrated "black box", with the term "parameter estimation" referring to the determination of the parameter values of the process. However, from this point on in this study, the terms "parameter identification" and "parameter estimation" will be considered as synonymous.

In most engineering situations the black box approach is not a very realistic one. The experimenter, in many cases, has derived some a priori knowledge from physical insight into the process under consideration. This may give information on the topology of a conceptual model for that process, and perhaps even an approximate knowledge on the values of the coefficients (parameters) in that model. Thus one must consider the use of all a priori information about the process in order to obtain an optimum speed and best economical solution to the problem [15].

The scheme for solving the parameter estimation problem may use either a physical model or an explicit mathematical relation (such as classical least-squares). Figs. 1.2(a) and (b) indicate the difference between an "implicit" model and an "explicit" mathematical relation [16]. Vector $\underline{x}(t)$ in this case is defined as the estimate of the process parameters. The bloc denoted by " I_n " processes the available information. In Fig. 1.2(a) estimates (\underline{x}) for the parameters are determined by successive adjustments of the model. These



(a) Using a physical model.



(b) Using an explicit mathematical relationship.

FIG. 1.2 (Adapted from Ref. [16]). Two basic identification schemes.

adjustments are made using some quality criterion with respect to process and model correspondence.

Suppose, for example, the criterion in the scheme of Fig. 1.2(a) was the minimization of some function or functional of the error between the process output $y(t)$ and the model output $y_m(t)$. The goal of parameter estimation in this case would be defined as: adjustment of model parameters \underline{x} in such a way that the actual error ϵ is minimal in some pre-defined sense.

$$\epsilon = y - y_m \quad (1.1)$$

Certain properties [14] of the estimates \underline{x} are desired for evaluating the performance of the identification technique. These are that the estimate be unbiased, efficient and consistent. If these properties exist, then \underline{x} is considered to be a "good" estimate.

1.2 Techniques for Solving the Identification Problem

One can distinguish between different kinds of estimates on the basis of the availability and use of certain a priori information. The identification scheme provides numerical values of the parameter estimates in terms of the available a priori knowledge and measured variables.

- (a) Minimum Risk or Bayes Estimate [14]. In this situation much a priori information must be available; the probability density functions of the noise e and the para-

meter values $\underline{\theta}$, the probability density of the measurements y , and the cost of choosing the value \underline{x} for the estimate if the true value of the process parameters is $\underline{\theta}$ (see Fig. 1.2).

- (b) Maximum Likelihood Estimate [3]. The a priori knowledge required consists of the joint probability density functions of the samples $y(0), \dots, y(k\tau)$ in addition to the covariance matrix of the noise. For independent samples, each with a probability density function $p(y(i\tau); \underline{\theta})$, the likelihood function becomes:

$$L\{y(0), \dots, y(k\tau); \underline{\theta}\} = \prod_{i=1}^k p\{y(i\tau); \underline{\theta}\} \quad (1.2)$$

This method uses as an estimate \underline{x} of $\underline{\theta}$ that vector which makes L as large as possible.

If the noise sequence has a Gaussian (normal) distribution, the maximum likelihood solution can be shown [14] to be identical to the generalized least-squares solution.

- (c) Generalized Least Squares Estimation. If the covariance matrix of the noise \underline{e} is known a priori, i.e.,

$$N = E[\underline{e}\underline{e}^T] \quad \text{where} \quad \underline{e}^T = [e(0), e(1), \dots, e(k\tau)]$$

then the Generalized Least Squares estimate [8] is obtained by minimizing

$$J = \underline{\epsilon}^T N \underline{\epsilon} \quad (1.3)$$

For $N \neq I$ (identity matrix) the variance of this estimate is smaller than that of the classical least squares estimate [16].

- (d) Classical Least Squares Estimate. If no knowledge on the covariance of the noise is available, it is best to choose N as the identity matrix, and define the error criterion as

$$J = \underline{\epsilon}^T \underline{\epsilon} \quad (1.4)$$

From the minimization of J the so-called "normal equation" for this estimate can be found.

It can be shown [9] that for "coloured" noise or frequency-band-limited error samples, the estimates obtained by classical-least-squares are biased and remain biased even if the data increase without limit.

- (e) Optimum Linear Filtering [21]. Every problem in optimum linear filtering or prediction of random processes can be formulated as an exactly equivalent problem, yielding identical solutions, of estimating a vector of constant parameters by the method of least squares. As pointed out by Swerling [34], this can be demonstrated by choosing the appropriate quadratic function to be minimized.

- (f) Stochastic Approximation Methods. Consider, as an example,

the system shown in Fig. 1.2(a) where the error is defined in eqn. (1.1). A quadratic function of the error ϵ is chosen as cost function and it is desired to find a parameter estimate \underline{x} to minimize the cost J

$$J = E[\epsilon^2] \quad (1.5)$$

A popular adjustment policy [35] can be obtained by choosing

$$\underline{x}^{(i+1)} = \underline{x}^{(i)} + \gamma^{(i)} \underline{\nabla} J^{(i)} \quad (1.6)$$

where $\gamma^{(i)}$ = a factor governing the speed of convergence.

$\underline{\nabla} J^{(i)}$ = the gradient of J with respect to parameter estimate \underline{x} at the i th sequence of error samples.

The solution to this problem can be found by the deterministic algorithm (1.6) only in the cases where the probability density of the random variable y is known a priori so that J can be evaluated. If this is not the case, however, the optimal vector \underline{x}^* can be found by applying the gradient method to samples of ϵ^2 rather than to the expected value. This, then, is an example of one stochastic approximation method.

Another method would be to use a Newton-Raphson

algorithm in place of (1.6). In this case both the gradient and partial derivatives of J need to be estimated.

The crucial problem connected with equation (1.6) is the determination of the gradient. Several approaches [16] are available:

1. Use two models with parameters \underline{x} and $\underline{x} + \Delta \underline{x}$.
2. Use one model with measurements taken before and after a step change in \underline{x} .
3. Use parameter influence coefficients or parameter sensitivity functions (when applying the continuous version of equation (1.6)).

1.3 Previous Development of the Identification Problem

The first major analytical approach to the problem of eliminating unknown disturbances from measurements was developed by Gauss in the early 1800's in connection with the analysis of astronomical observations. He also showed the relevance of the famous Gaussian distribution to the characterization of measurement errors and in fact his procedures are in wide use today under the general title of Gauss Least-squares curve fitting. The next major development came in the 1940's when Wiener and Kolmogorov first discussed problems of linear least squares estimation for stochastic processes, but by entirely different methods, both deriving certain optimal filters for processing the measurements. The next major development in the analytical treatment of the problem occurred in the late 1950's when Kalman

and Bucy [22] reformulated the problem in recursive form using the state variable description of dynamic systems.

The basic idea of stochastic approximation was introduced by Robbins and Monro [30] who, in 1951, set up a scheme for finding the root of a regression function. Kiefer and Wolfowitz [23] extended this method to the problem of finding the extremum of a regression function where the function was the expected value of a random variable depending on several real value parameters.

Sakrison [31] considered the Kiefer-Wolfowitz procedure and the case where the random variable was an ergodic random process. A continuous version of the procedure was developed for this case. An advantage of this procedure lies in the fact that it may be mechanized with simple analog computation components.

Indeed, Saridis and Richer [44] mechanized Sakrison's method using an analog computer with digital sequencing logic. Two algorithms were described: the first using sensitivity functions to find the gradient of the error function, while the second method found the gradient using a modified Kiefer-Wolfowitz procedure. In both cases, the amount of analog equipment required became excessive when searching for more than two or three parameters.

Elliott and Swarder [13] studied the multidimensional Kiefer-Wolfowitz stochastic approximation algorithm and discussed a method of eliminating some of the restrictions required for convergence. In order to accomplish this, an appropriate transformation of the problem coordinate system was required to ensure

that the criterion of performance be almost equally "sensitive" to each component of the parameter vector. Although convergence was then less restricted by the system parameter sensitivities, it was obvious that the proposed algorithm was computationally expensive.

A further modification of the multidimensional Kiefer-Wolfowitz stochastic approximation algorithm was presented by Elliott and Swarder [12] using a "variable metric" technique. First an iterative method of evaluating the Hessian matrix of a regression function was proposed. This method was then used in conjunction with the Kiefer-Wolfowitz procedure to obtain a stochastic analog of the Newton-Raphson gradient search method. This algorithm is related to Davidon's [40] variable metric method for minimizing a function of several variables. Both of these methods by Elliott and Swarder have the limitation that computation time and computer memory increase rapidly with the dimension of the parameter vector.

A scalar-gain stochastic-approximation scheme involving simple implementation and significant reduction of computer time was described by Panuska [24]. Consistent parameter estimates were obtained from a stochastic approximation algorithm with an enlarged parameter space incorporating an adaptive filter. The "built in" adaptive filter is used to remove the bias in estimates caused by correlation between the noise and measured system outputs.

This scheme was later modified to recursive least squares form by Young [39] and by Panuska [28]. The proposed algorithm

is computationally more expensive than the corresponding scalar gain stochastic approximation formula [24], but converges much faster, and there are no problems with the choice of the gain constant.

Aström [3] has applied the maximum likelihood procedure to parameter identification of discrete-time systems from input/output samples. A Newton-Raphson algorithm was used for recursive parameter estimates to maximize the likelihood function. This algorithm was also used by Gustavsson [18] who presented a program package for identification by maximum likelihood method.

Fletcher and Powell [41] in developing a rapidly convergent method for minimization, eliminated the requirement for computation of second partial derivatives while retaining quadratic convergence properties. It was left up to the user to supply the gradient values.

Bekey and Malony [43] used the Fletcher-Powell method for least-squares estimation, implementing the scheme on a hybrid computer. The gradient vector was calculated on the analog computer using sensitivity equations.

Many other approaches to the identification problem have been taken, for example, Mehra [42] and Kailath [10]. However, it would be far beyond the scope of this thesis to investigate all possible techniques.

1.4 Scope of the Thesis

The main objective of the work presented here is to investigate a few methods for parameter identification of linear

systems, generally covering the approaches outlined in section 1.2. Comparisons are made with regard to such items as estimation accuracy, ease and cost of implementation. In most cases, a general (canonical) second-order model was used.

Chapter 2 presents the statement of the problem to be solved in detail for both discrete and continuous-time cases.

A detailed discussion of identification schemes selected for investigation is presented in Chapter 3. The results of these techniques as applied to a second-order system example are presented in Chapter 4.

Finally, as a practical application, the modelling of an electrical power system load is discussed. The identification techniques investigated in Chapter 3 are extended [11] for use with system models containing an additional period component. The extended methods are then applied to the problem of electric power system load modelling from input-output data. Results of model parameter estimation based on both simulated and real load data are presented.

CHAPTER 2

STATEMENT OF THE PROBLEM

2.1 Discrete Time Systems

Consider a discrete time single-input, single-output dynamical system whose input-output relation can be described by the equation

$$A(z^{-1})y_n = B(z^{-1})u_n + \sigma C(z^{-1})e_n \quad (2.1)$$

where $\{u_n\}$, $\{y_n\}$ are input and output sequences, $\{e_n\}$ is a sequence of independent random variables with zero mean and unit variance, σ is a positive constant. A , B , and C are polynomials of degree k in the backward shift operator z^{-1} defined by

$$z^{-1}y_n = y_{n-1} \quad (2.2)$$

$$A(z^{-1}) = 1 + a_1 z^{-1} + \dots + a_k z^{-k} \quad (2.3)$$

$$B(z^{-1}) = b_0 + b_1 z^{-1} + \dots + b_k z^{-k}$$

$$C(z^{-1}) = 1 + c_1 z^{-1} + \dots + c_k z^{-k}$$

The following assumptions are made:

- a) The functions $A(z^{-1})$ and $C(z^{-1})$ have all their zeros

inside the unit circle.

- b) There are no factors common to all three polynomials $A(z)$, $B(z)$ and $C(z)$.

These two assumptions imply that the homogeneous equation corresponding to (2.1) is asymptotically stable and that every state of the system is controllable either from u or e .

There is no loss in generality to assume that the leading coefficients of the polynomials $A(z)$ and $C(z)$ are unity. We cannot, however, make this assumption for the polynomial $B(z)$.

Note that $B(z^{-1})/A(z^{-1})$ has the interpretation of the pulse transfer function from the input $\{u_n\}$ to the output $\{y_n\}$. The initial conditions can be assumed zero.

The identification problem can now be formulated as follows.

PROBLEM

Given the input $\{u_n, n = 1, \dots, N\}$ and the observed output $\{y_n, n = 1, \dots, N\}$, find an estimate of the parameters of the model (2.1).

Special cases of this problem are well known:

- 1) $k = 0$; regression analysis.
- 2) $b_0 = b_1 = \dots = b_k = c_1 = c_2 = \dots = c_k = 0$; estimation of parameter in autoregressive processes.
- 3) $b_0 = b_1 = \dots = b_k = a_1 = a_2 = \dots = a_k = 0$; estimation of parameters in a moving average.

- 4) $b_0 = b_1 = \dots = b_k = 0$; parameter estimation of rational power spectra.
- 5) $c_1 = c_2 = \dots = c_k = 0$; least squares model building.
- 6) $a_i = c_i$, $i = 1, 2, \dots, k$; identification of noise-free process with measurement errors.

2.2 Continuous Time Systems

Consider a single-input, single-output system governed by

$$A(p)y(t) = B(p)u(t) + C(p)e(t) \quad (2.4)$$

where u is the input, $y(t)$ is the output, and $e(t)$ is a zero mean, stationary white noise with

$$E[e(t)e(\tau)] = \sigma^2 \delta(t-\tau) \quad (2.5)$$

$\delta(t-\tau)$ is the Dirac delta function.

$$\begin{aligned} \delta(t-\tau) &= 1, \quad \tau = t \\ \delta(t-\tau) &= 0, \quad \tau \neq t. \end{aligned}$$

The symbol p denotes a differentiation operator

$$px = dx/dt$$

and A , B and C are polynomials

$$A(p) = p^n + a_1 p^{n-1} + \dots + a_n \quad (2.6)$$

$$B(p) = b_0 p^n + b_1 p^{n-1} + \dots + b_n$$

$$C(p) = p^n + c_1 p^{n-1} + \dots + c_n$$

where $A(p)$ and $C(p)$ have zeros only in the left half-plane.

PROBLEM

Given input $u(t)$ and output $y(t)$ for $0 \leq t \leq T$, determine the parameters in eqns. (2.4) and (2.5).

CHAPTER 3

DESCRIPTION OF SELECTED METHODS

The parameter identification techniques presented here can be divided into two categories; continuous-time methods, and discrete-time methods.

Continuous methods shall be defined as those employing continuous-time models of the system under study. (For example, models constructed on analog computers). These models are constructed using updated estimates of the real system parameters, given the system input-output data. Updating of parameter estimates may be done by either continuous or discrete equations.

Discrete-time methods are those utilizing all-digital identification techniques, given the system input-output data.

3.1 Continuous System Model: Application of Kiefer-Wolfowitz Minimization

If $M(x)$ is a regression function with maximum at unknown point θ , where $M(x)$ itself is unknown to the experimenter, and the following conditions [23] are satisfied:

(a)

$$M(x) = \int_{-\infty}^{\infty} y \, dP(y/x) \quad (3.1)$$

where $P(y/x)$ is a family of distribution functions which depend on parameter x . $M(x)$ is thus defined as the expected value of random variable y given x .

$$(b) \quad \int_{-\infty}^{\infty} (y-M(x))^2 dP(y/x) \leq S < \infty \quad (3.2)$$

(c) $M(x)$ strictly increasing for $x < \theta$, and $M(x)$ strictly decreasing for $x > \theta$.

(d) For $\{\alpha_n\}$, $\{\gamma_n\}$ infinite sequences of positive numbers;

$$\alpha_n \rightarrow 0 \quad (3.3a)$$

$$\sum \gamma_n = \infty \quad (3.3b)$$

$$\sum \alpha_n \gamma_n < \infty \quad (3.3c)$$

$$\sum \gamma_n^{-2} \alpha_n^{-2} < \infty \quad (3.3d)$$

Then the recursive scheme

$$x_{n+1} = x_n + \frac{\gamma_n}{\alpha_n} (y_{2n} - y_{2n-1}) \quad (3.4)$$

with x_1 arbitrary, results in x_n converging stochastically to θ (as $n \rightarrow \infty$), as shown by Kiefer and Wolfowitz [23]. Random variables y_{2n} , y_{2n-1} , have distribution functions

$$P(y/x_n + \alpha_n) \text{ and } P(y/x_n - \alpha_n)$$

where $P(y/x_n) \equiv P(y/x_n, x_{n-1}, \dots, x_2, x_1)$.

Consider the system described in Laplace form by:

$$A(s)y(s) = B(s)u(s) + C(s)e(s) \quad (3.5)$$

where: $y(s)$ the Laplacian output of the system
 $u(s)$ system input (driving function)
 $e(s)$ the Laplacian of zero mean stationary
white noise $e(t)$.

$$A(s) = s^n + a_1 s^{n-1} + \dots + a_n \quad (3.6)$$

$$B(s) = b_0 s^n + b_1 s^{n-1} + \dots + b_n$$

$$C(s) = s^n + c_1 s^{n-1} + \dots + c_n$$

$A(s)$ and $C(s)$ have zeros only in the left complex plane.
 s in this case signifies the Laplace operator for
differentiation with zero initial conditions.

The system represented in this form may easily be
simulated on the analog computer along with desired models using
estimated parameters. Since all the necessary dynamic filtering
is done by the analog models, only the Kiefer-Wolfowitz algorithm
need be programmed on the digital part of the computer.

The error function to be minimized is formed as follows:

Consider the model of eqn. (3.5) with input-output
values u, y .

$$\hat{A}(s)y(s) = \hat{B}(s)u(s) + \hat{C}(s)\epsilon(s) \quad (3.7)$$

where $\hat{A}(s)$, $\hat{B}(s)$ and $\hat{C}(s)$ include estimation of the true parameter and $\epsilon(s)$ is an estimate of $e(s)$.

Solving for $\epsilon(s)$,

$$\epsilon(s) = \hat{C}^{-1}(s)[\hat{A}(s)y(s) - \hat{B}(s)u(s)] \quad (3.8)$$

Since the expected value of e , $E[e(t)]$ is assumed to be zero, it is desirable to minimize the expected value of $[\epsilon(t)]^2$.

This may be used as the error function in the Kiefer-Wolfowitz algorithm:

$$x_i(n+1) = x_i(n) - \frac{\gamma(n)}{2\alpha(n)} [\epsilon^2(x_i(n)+\alpha(n)) - \epsilon^2(x_i(n)-\alpha(n))] \quad (3.9)$$

$x_i(n)$ is an estimate of the parameter $x_i \in \{a_1, a_2, \dots, a_n, b_0, b_1, \dots, b_n, c_1, c_2, \dots, c_n\}$. (Let θ be parameter vector $(a_1, a_2, \dots, c_n)^T$).

$\alpha(n)$ and $\gamma(n)$ are sequences of positive real numbers chosen to guarantee convergence by satisfying eqns. (3.3).

Fig. 3.1 shows a general hybrid computer implementation of this method. The transfer functions $F_1(s)$, $F_2(s)$, $F_3(s)$ and $F_4(s)$ are defined by:

$$\begin{aligned} F_1(s) &= \frac{B(s)}{A(s)} & F_2(s) &= \frac{C(s)}{A(s)} \\ F_3(s) &= \frac{\hat{B}(s)}{\hat{C}(s)} & F_4(s) &= \frac{\hat{A}(s)}{\hat{C}(s)} \end{aligned} \quad (3.10)$$

$J = \epsilon^2$

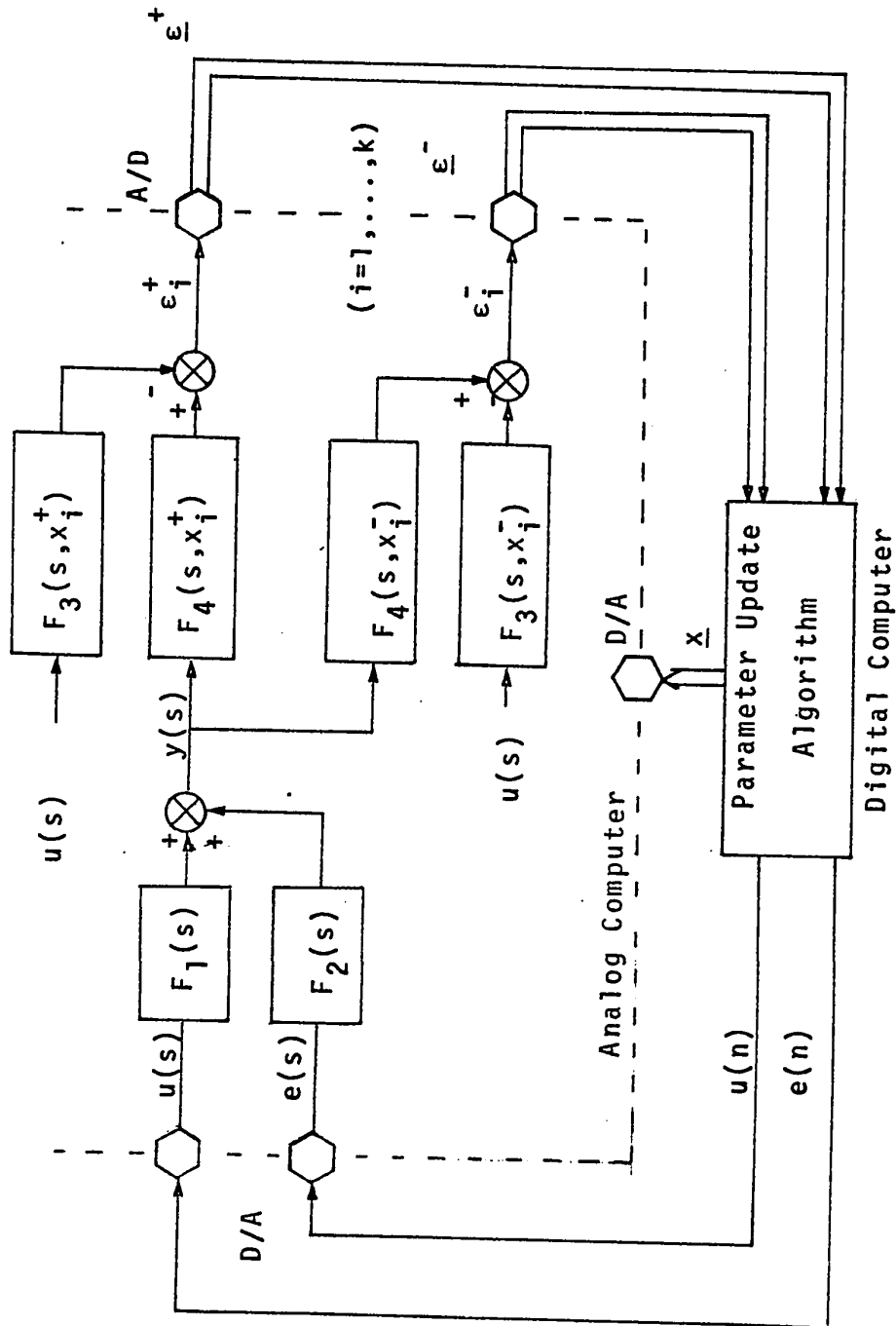


FIG. 3.1 Kiefer-Wolfowitz technique using hybrid computer.

The notation $F_3(x_i^+)$ means that the estimate of the i th element of the parameter vector θ has a perturbation given by $x_i^{(n)} = x_i^{(n)} + \alpha(n)$ and inserted into transfer function F_3 .

$u(t)$ and $e(t)$ are both generated in the digital section of the program. e is generated as a Gaussian disturbance with zero mean while u is a pseudo-random binary sequence of amplitude one. The sign of u is randomly chosen and satisfies a uniform distribution.

All parameter estimates are done on the digital computer and inserted into the F_3 and F_4 transfer functions in the analog section.

Note that $\epsilon(t)$ is not an instantaneous function of the parameters, due to transient effects present in dynamic systems, but rather it depends on the present and past history of both the system and the parameters. Thus, even if it were possible to obtain a nearly instantaneous adjustment of the parameters to their correct values, the criterion function $J = \epsilon^2$ would not instantaneously decrease to its minimum value unless all transients have been dissipated [1]. Since J depends on the entire time history of the parameters, it is no longer a function in the sense of ordinary calculus but rather a functional, and an instantaneous gradient cannot be mathematically defined. To circumvent this mathematical difficulty the parameters can be allowed to remain fixed during the computation of the gradient.

For example, if the criterion function was defined as

$$J = \int_0^T \epsilon^2 dt \quad (3.11)$$

the gradient of J would be calculated while the parameters are fixed during an interval of T seconds.

This approach is described in section 4.1 where a test example of the Kiefer-Wolfowitz method is shown.

A useful means of approximating the gradient of an unknown function is implied in the Kiefer-Wolfowitz algorithm.

As an example, consider the function $y(x)$ whose nature is unknown to the observer. The gradient at point x_n can be approximated by evaluating y at $(x_n + \alpha_n)$ and $(x_n - \alpha_n)$ and using the average slope, as indicated in Fig. 3.2.

The methods discussed later in this chapter require the calculation of the gradient of the function to be minimized. In some cases, the second partial derivatives are also required. This can be difficult when the nature of the function to be minimized is unknown to the experimenter.

When using analog computer techniques, it is possible to find the approximate gradients by the use of "sensitivity" equations [1]. However, it is necessary, in these cases, to assume that the rate of adjustment of the parameters will be sufficiently slow compared to the basic time constants of the system being identified. When a rapid rate of parameter adjustment is desired, the implementation of this method leads to serious stability problems. Furthermore, the use of sensitivity equations requires an excessive number of analog computing components.

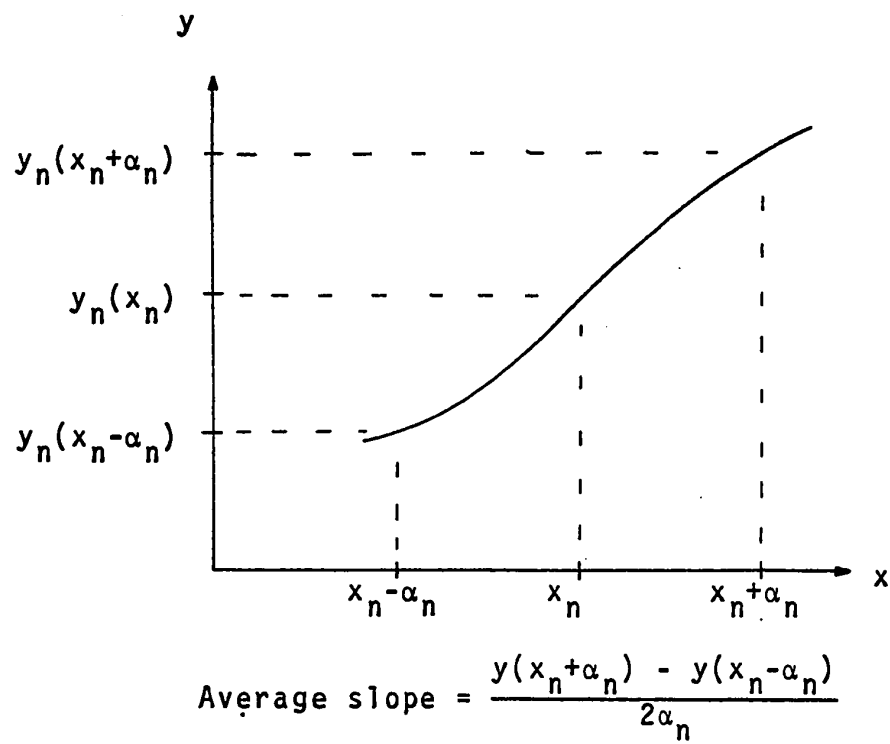


FIG. 3.2 Gradient approximation by parameter perturbation.

3.2 Computational Aspects of Methods Using Discrete Models

3.2.1 A stochastic approximation method using adaptive filtering

Panuska [24] developed a stochastic approximation algorithm with an enlarged parameter space incorporating an adaptive filter.

The system to be identified is governed by the linear difference equation:

$$A(z^{-1})y_n = B(z^{-1})u_n + \lambda C(z^{-1})e_n \quad (3.12)$$

where A , B , and C are polynomials of degree k in the backward shift operator z^{-1} defined by

$$z^{-1}y_n = y_{n-1} \quad (3.13)$$

$$A(z^{-1}) = 1 + a_1 z^{-1} + \dots + a_k z^{-k} \quad (3.14)$$

$$B(z^{-1}) = b_0 + b_1 z^{-1} + \dots + b_k z^{-k} \quad (3.15)$$

$$C(z^{-1}) = 1 + c_1 z^{-1} + \dots + c_k z^{-k} \quad (3.16)$$

$\{u_n\}$, $\{y_n\}$ are input-output sequences and $\{e_n\}$ is a sequence of independent random variables with zero mean and unit variance. λ is a positive constant.

Panuska showed that for the standard stochastic approximation technique, this model will produce biased results since the moving average produced by the $\lambda C(z^{-1})e_n$

terms in eqn. (3.12) is correlated with the system outputs y_n .

The bias of the standard schemes is removed by using an algorithm with a "built-in" adaptive filter, determined by the current estimates of the parameters of eqn. (3.12).

An enlarged parameter vector $\underline{\theta}$ was formed:

$$\underline{\theta}^T = (-a_1, \dots, -a_k, b_0, \dots, b_k, c_1, \dots, c_k) \quad (3.17)$$

with the n th estimate of the vector denoted by \underline{x}_n .

Then system (3.12) was rewritten as:

$$y_n = \underline{\theta}^T \underline{v}_n + \lambda e_n \quad (3.18)$$

where \underline{v}_n is the observation vector

$$\begin{aligned} \underline{v}_n^T = & (y_{n-1}, \dots, y_{n-k}, u_n, \dots, u_{n-k}, \\ & \lambda e_{n-1}, \dots, \lambda e_{n-k}) \end{aligned} \quad (3.19)$$

At the end of the n th step of estimation procedure the eqn. (3.18) reads

$$\lambda e_n = y_n - \underline{x}_n^T \hat{\underline{v}}_n \quad (3.20)$$

$\hat{\underline{v}}_n$ is defined by eqn. (3.19) as the computed observation vector with e_{n-1}, \dots, e_{n-k} being replaced by $\hat{e}_{n-1}, \dots, \hat{e}_{n-k}$ (outputs of the model).

The algorithm is then defined by

$$\underline{x}_{n+1} = \underline{x}_n + G_n(y_n - \underline{x}_n^T \hat{\underline{V}}_n) \hat{\underline{V}}_n \quad (3.21)$$

where $G_n = \frac{G}{n}$, G being a positive gain constant.

This method can be tested using an all-digital program by assuming that the system to be identified is governed by the discrete difference eqn. (3.12). All input-output observation would then be made from the digital solution of (3.12). However, since many practical processes are continuous, not discrete, it is desirable to achieve conditions as realistic as possible by modelling the system to be identified on the analog computer. Eqns. (3.20) and (3.21) would then be solved on the digital portion of the hybrid computer. Assuming that the analog model of the unknown system is realistic, the identification scheme becomes an on-line process.

In order to simulate the unknown system in the continuous form (analog computer), it is necessary to convert eqn. (3.12) from the Z-transform to the Laplace transform notation.

The resulting form would be

$$A'(s)y(s) = B'(s)u(s) + C'(s)e(s) \quad (3.22)$$

where the coefficients of polynomials $A'(s)$, $B'(s)$ and $C'(s)$ are not necessarily the same as those of the polynomials

$A(z)$, $B(z)$ and $C(z)$.

Since $B(z^{-1})/C(z^{-1})$ has in eqn. (3.12) the obvious interpretation as the pulse transfer function from input $\{u_n\}$ to output $\{y_n\}$, transforming $B(z^{-1})/A(z^{-1})$ and $C(z^{-1})/A(z^{-1})$ into Laplace form will yield the necessary continuous transfer functions.

Panuska [24] shows that convergence of the estimation algorithms in the mean square sense is obtained and both he and Sacks [36] discuss resulting parameter variances.

3.2.2 An adaptive recursive least-squares identification algorithm [28]

An adaptive recursive least-squares algorithm with "on-line" structure was proposed by Panuska [28] where, as in application of his scalar gain algorithm [24], the bias effect of correlated noise was eliminated by introducing the concept of an "enlarged" system parameter vector.

The general system equation as described by eqn. (2.1) can be written in vector notation:

$$y(n) = \underline{\theta}^T \underline{v}(n) + e(n) \quad (3.23)$$

where $\underline{\theta}$, \underline{v} are described by (3.17) and (3.19) respectively.

Restating the identification problem: Given a sequence of observed input-output pairs

$$\{u(n), y(n)\}, n = 1, 2, \dots,$$

find an estimate \underline{x} of the enlarged system parameter vector $\underline{\theta}$ in eqn. (3.23) and a variance estimate λ^2 .

The algorithm can then be obtained by a formal application of the least squares formula.

$$\begin{aligned} \underline{x}(n+1) = & [\underline{x}(n) + \Gamma(n)[\hat{\underline{V}}^T(n)\Gamma(n-1)\hat{\underline{V}}(n)+1]^{-1}[y(n) - \\ & -\underline{x}^T(n)\hat{\underline{V}}(n)]\hat{\underline{V}}(n)]\Omega \end{aligned} \quad (3.24)$$

$$\Gamma(n) = \Gamma(n-1) - \Gamma(n-1)\hat{\underline{V}}(n)[\hat{\underline{V}}(n)\Gamma(n-1)\hat{\underline{V}}(n)+1]^{-1}\hat{\underline{V}}^T(n)\Gamma(n-1) \quad (3.25)$$

Eqns. (3.24) and (3.25) in effect describe a Kalman filter [21] used for estimation of system parameters rather than system state.

Convergence of the algorithm in the mean square has been proven [21,28] as $n \rightarrow \infty$. Initial conditions are:

$$\underline{x}(1) = \text{arbitrary} \in \Omega$$

$$\Gamma(0) = \text{diag } [1,1,\dots,1]$$

$[\cdot]\Omega$ means truncation to a bounded closed rectangular set Ω known to contain $\underline{\theta}$.

$$\Omega \stackrel{\Delta}{=} \{\underline{x} : \omega_i < x_i < \beta_i\}$$

In eqn. (3.24), $\hat{\underline{V}}(n)$ denotes a vector obtained from

$\underline{v}(n)$ defined in (3.17) by replacing the errors $e(n-1), \dots, e(n-k)$, by "computed errors" $\epsilon(n-1), \dots, \epsilon(n-k)$, where

$$\epsilon(n) = y(n) - \underline{x}^T \hat{\underline{v}}(n) \quad (3.26)$$

with initial conditions

$$y(-k+1) = y(-k+2) = \dots = y(0) = 0$$

$$u(-k+1) = u(-k+2) = \dots = u(0) = 0$$

$$\epsilon(-k+1) = \epsilon(-k+2) = \dots = \epsilon(0) = 0$$

3.2.3 Fletcher-Powell Minimization [41]

Fletcher and Powell developed an iterative descent method for finding the local minimum of a function of several variables.

Let \underline{x} be the vector of adjustable model parameters. Define the criterion $J = \int_0^T \epsilon^2(\underline{x}, t) dt$ as in the previous section.

The Fletcher-Powell algorithm is an iterative procedure in which the $(i+1)$ st parameter vector is given by

$$\underline{x}^{i+1} = \underline{x}^i - \gamma^i H^i \nabla J(\underline{x}^i) \quad (3.27)$$

H is a positive definite symmetric matrix which tends to the inverted Hessian matrix for J at its minimum.

The iteration procedure can be described in the following steps.

1. Choose an initial value for H of $H' = \begin{vmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{vmatrix}$.
2. Set $\underline{S}^i = -H^i \nabla \underline{J}$. This establishes the direction of the step modification to \underline{x}^i .
3. Obtain γ^i such that $J(\underline{x}^i + \gamma^i \underline{S}^i)$ is a minimum with respect to ρ along $\underline{x}^i + \rho \underline{S}^i$ and $\gamma^i > 0$.
4. Set $\underline{\Delta}^i = \gamma^i \underline{S}^i$.
5. Set $\underline{x}^{i+1} = \underline{x}^i + \underline{\Delta}^i$.
6. Evaluate $J(\underline{x}^{i+1})$ and $\nabla \underline{J}(\underline{x}^{i+1})$.
7. Set $\underline{y}^i = \nabla \underline{J}(\underline{x}^{i+1}) - \nabla \underline{J}(\underline{x}^i)$.
8. Set $H^{i+1} = H^i + A^i + B^i$ (3.28)

$$\text{where matrix } A^i = \frac{\underline{\Delta}_i \underline{\Delta}_i^T}{\underline{\Delta}_i^T \underline{y}^i}$$

$$\text{and matrix } B^i = \frac{-H^i \underline{y}^i \underline{y}^{iT} H^i}{\underline{y}_i^T H^i \underline{y}_i}.$$

9. Set $i = i+1$ and repeat from step 2.

Stop the procedure when $(\underline{S}_i^T \underline{S}_i)^{1/2}$ is less than a prescribed amount, or when a specified number of iterations is exceeded.

The gradient $\nabla \underline{J}(\underline{x}^i)$ must be defined for each iteration by the user of the method.

Bekey and Maloney [43] have used sensitivity equations to compute $\nabla \underline{J}$ by analog computer and pointed out the dramatic speed advantage of the hybrid technique over all digital

solutions. However, a simple second order system was studied with no noise present in the system. As pointed out in section 3.1, sensitivity function methods require large amounts of analog computing equipment and can also present stability problems.

To estimate the parameter γ^i in step 3, Fletcher and Powell used an algorithm employing the cubic interpolation technique suggested by Davidon [40]. The initial step in the procedure is to choose a point \underline{u}^i on the vector $[\underline{x}^i + \rho \underline{s}^i]$ with $\rho > 0$.

Let J_x , ∇J_x , J_u , and ∇J_u denote the values of the function and gradient at points \underline{x}^i and \underline{u}^i . Then an estimate of γ^i can be found by interpolating cubically, using the function values J_x and J_u and the components of the gradient along \underline{s}^i .

This is given by,

$$\frac{\gamma^i}{\rho} = 1 - \frac{\nabla J_u^T \underline{s}^i + w - r}{\nabla J_u^T \underline{s}^i - \nabla J_x^T \underline{s}^i + 2w} \quad (3.29)$$

$$\text{where } w = (r^2 - \nabla J_x^T \underline{s}^i \nabla J_u^T \underline{s}^i)^{1/2} \quad (3.30)$$

$$\text{and } r = \frac{3}{\rho}(J_x - J_u) + \nabla J_x^T \underline{s}^i + \nabla J_u^T \underline{s}^i \quad (3.31)$$

A suitable choice of the point \underline{u}^i is given by selecting ρ from:

$$\rho = \text{minimum of } \left[1, \frac{-2(J_x - J_u)}{\nabla J_x^T \underline{s}^i} \right] \quad (3.32)$$

where J_0 is the predicted lower bound of $J(\underline{x})$.

If $J(\underline{x}^i + \gamma^i \underline{S}^i)$ is not less than both J_x and J_u , then use a smaller value of ρ and repeat the interpolation.

Davidon suggests one should ensure that the minimum is located between \underline{x}^i and \underline{u}^i by testing the sign of $\nabla J_u^T \underline{S}^i$ and comparing J_u and J_y before interpolating.

If $\nabla J_u^T \underline{S}^i$ is negative or, if $J_u < J_x$, then: before interpolation, increase the size of step \underline{S}^i by modifying H ,

$$H_{jk} + \frac{1}{v} S_j S_k \rightarrow H_{jk} \quad (3.33)$$

where v is the squared length of \underline{S} .

The process is then repeated starting from the new position.

3.2.4 Aström's technique for identification (Maximum Likelihood Method)

In the report by Gustavsson [18], a digital program package is used which produces a mathematical model of the process and its disturbances from given input/output samples. Identification is done by the Maximum Likelihood Method.

The problem can be stated as follows:

Given input/output samples $\{u(t), y(t); t=1, 2, \dots, N\}$ where $u(t)$ is the input signal and $y(t)$ is the output signal, find an estimate of the parameters of the system model

$$A(z^{-1})y(t) = B(z^{-1})u(t) + \lambda C(z^{-1})e(t) \quad (3.34)$$

Definitions for the various terms in (3.34) can be found in eqns. (3.13) and (3.14).

$A(z^{-1})$ and $C(z^{-1})$ have all their zeros inside the unit circle. Furthermore, there are no factors common to all the polynomials $A(z^{-1})$, $B(z^{-1})$ and $C(z^{-1})$.

It follows from (3.34) that the residuals $\{\varepsilon(t), t = 1, 2, \dots, N\}$ defined by

$$C(z^{-1})\varepsilon(t) = A(z^{-1})y(t) - B(z^{-1})u(t) \quad (3.35)$$

are independent and normal $(0, \lambda)$. The logarithm of the likelihood function becomes [45]

$$L = \frac{-1}{2\lambda^2} \sum_{t=1}^N \varepsilon^2(t) - N \log \lambda + \text{constant} \quad (3.36)$$

Maximizing this function is equivalent to minimizing the loss function

$$J(\underline{\theta}) = \frac{1}{2} \sum_{t=1}^N \varepsilon^2(t) \quad (3.37)$$

where $\underline{\theta}$ is the column vector $(a_1, \dots, a_k, b_0, \dots, b_k, c_1, \dots, c_k)$.

\underline{x} is found such that $J(\underline{x})$ is minimal.

To minimize the function, an iterative combined Gauss-Newton and Newton-Raphson algorithm is used.

$$\underline{x}^{n+1} = \underline{x}^n - \gamma [J_{\theta\theta}(\underline{x}^n)]^{-1} J_{\theta}(\underline{x}^n) \quad (3.38)$$

where J_0 = gradient vector of $J(\underline{x})$

$J_{\theta\theta}$ = matrix of second partial derivatives
of $J(\underline{x})$

γ = scaling factor.

Differentiating (3.37) gives

$$\frac{\partial J}{\partial x_i} = \sum_{t=1}^N \epsilon(t) \frac{\partial \epsilon(t)}{\partial x_i} \quad (3.39)$$

$$\frac{\partial^2 J}{\partial x_i \partial x_j} = \sum_{t=1}^N \frac{\partial \epsilon(t)}{\partial x_i} \frac{\partial \epsilon(t)}{\partial x_j} + \sum_{t=1}^N \epsilon(t) \frac{\partial^2 \epsilon(t)}{\partial x_i \partial x_j} \quad (3.40)$$

Eqn. (3.38) becomes a Gauss-Newton algorithm if only the first term of (3.40) is used.

Near the minimum, a Newton-Raphson procedure is used by considering both terms of the right-hand side of eqn. (3.40).

The minimizing algorithm becomes:

1. Put $\underline{x}^n = \underline{x}^0$ (starting value of \underline{x})
2. Evaluate $V_{\theta}(\underline{x}^n)$ and $V_{\theta\theta}(\underline{x}^n)$
3. Calculate \underline{x}^{n+1} and repeat from 2.

Aström [2,3] showed that for noise, $e(t)$, Gaussian zero mean, the estimates are asymptotically efficient. This means in practice that one cannot expect to find an estimator with a greater accuracy for long samples.

By choosing appropriate state variables for the computation, and by performing the computations recursively, the gradient of the residuals and the second partial derivatives of the residuals can be economically calculated.

Differentiating (3.35) gives

$$\frac{\partial}{\partial a_j} [C(z^{-1})\epsilon(t)] = z^{-j}y(t)$$

$$\therefore C(z^{-1}) \frac{\partial \epsilon(t)}{\partial a_j} = z^{-j}y(t) \quad (3.41a)$$

$$\text{similarly: } C(z^{-1}) \frac{\partial \epsilon}{\partial b_j} = -z^{-1}u(t) \quad (3.41b)$$

$$\frac{\partial}{\partial c_j} [C(z^{-1})\epsilon(t)] = 0$$

$$\therefore C(z^{-1}) \frac{\partial \epsilon}{\partial c_j} = -z^{-j}\epsilon(t) \quad (3.41c)$$

Differentiating (3.30c) once more:

$$\begin{aligned} C(z^{-1}) \frac{\partial^2 \epsilon}{\partial a_i \partial c_j} &= -z^{-i-j+1} \frac{\partial \epsilon(t)}{\partial a_1} \\ C(z^{-1}) \frac{\partial^2 \epsilon}{\partial b_i \partial c_j} &= -z^{-i-j+1} \frac{\partial \epsilon(t)}{\partial b_1} \\ C(z^{-1}) \frac{\partial^2 \epsilon}{\partial c_i \partial c_j} &= -2z^{-i-j+1} \frac{\partial \epsilon(t)}{\partial c_1} \end{aligned} \quad (3.42)$$

$$\text{Since } \frac{\partial \epsilon}{\partial a_i} = z^{-i+1} \frac{\partial \epsilon(t)}{\partial a_1} = \frac{\partial \epsilon(t-i+1)}{\partial a_1} \text{ for } i \leq t+1.$$

To compute the residuals from eqn. (3.35), a state variable representation is introduced:

$$x_1(t+1) = -c_1 x_1(t) + x_2(t) + a_1 y(t) - b_1 u(t) + y(t+1)$$

$$\begin{aligned}
 x_2(t+1) &= -c_2 x_1(t) + x_3(t) + a_2 y(t) - b_2 u(t) \\
 &\cdot \quad \cdot \\
 &\cdot \quad \cdot \\
 &\cdot \quad \cdot
 \end{aligned}
 \tag{3.43}$$

$$x_k(t+1) = -c_k x_1(t) + a_k y(t) - b_k u(t)$$

$$\text{and } \epsilon(t) \equiv (1, 0, \dots, 0) \underline{x}(t)$$

Eqns. (3.41) complete derivatives as follows, again using a state variable approach.

$$\frac{\partial \epsilon}{\partial a_1} = x_1(t)$$

$$\frac{\partial \epsilon}{\partial a_2} = x_1(t-1) = x_2(t) \text{ etc. (see Gustavsson [18])}.$$

For computation of the exact second partial derivatives of the loss function, the second partial derivatives of the residuals are needed, that is

$$\frac{\partial^2 \epsilon(t)}{\partial x_i \partial x_j} \quad i, j = 1, 2, \dots, n$$

From eqns. (3.42),

$$\frac{\partial \epsilon(t)}{\partial a_i \partial c_j} = \frac{\partial^2 \epsilon(t-i-j+2)}{\partial a_1 \partial c_1} \text{ etc.}$$

These relationships can be used to facilitate computation.

NOTE: The second partial derivatives are zero if no differentiating is made with respect to a "c"-parameter.

After parameter estimates have been obtained, the order of the model can be tested [3] to ensure that the assumed order is not less than the order of the system. A test of the residuals should indicate that $\{\epsilon(t), t=1, \dots, N\}$ form a series of independent normal variables. One simple test of independence would be to compute the covariance function $\frac{1}{N} \sum_{t=1}^N \epsilon(t) (t+\tau)$ for a few delays $\tau = 1, 2, 3, \dots$. A quick method is to count the sign changes, the number of which should be $\approx \frac{1}{2} N$ for a sequence of independent variables.

CHAPTER 4

TEST EXAMPLES

4.1 An Algorithm Using Kiefer-Wolfowitz Minimization Method

The following example problem illustrates the use of a continuous-time system model for parameter identification. The minimization technique proposed by Kiefer and Wolfowitz is implemented making use of hybrid computer techniques (EAI 690 hybrid computer). All system models are constructed on the analog computer. Limitations on the size of the example problem chosen were necessary due to the amount of analog and digital equipment available.

Thus, a simple first-order single-input example has been chosen.

Consider the first-order system written in Laplace form,

$$A(s)y(s) = B(s)u(s) + C(s)e(s) \quad (4.1)$$

$$\text{where } A(s) = s + a_1$$

$$B(s) = b_0s + b_1$$

$$C(s) = s + c_1$$

The parameters to be identified are:

$$a_1 = 1.0$$

$$b_1 = 6.0$$

$$b_0 = 0.0$$

$$c_1 = 2.0$$

Let $a = a_1$, $b = b_1$, $c = c_1$.

Using an analog model of eqn. (4.1) and solving for the estimated error ϵ , we get,

$$\epsilon(s) = \frac{s + \hat{a}}{s + \hat{c}} y(s) - \frac{\hat{b}}{s + \hat{c}} u(s) \quad (4.2)$$

Let parameter vector $\underline{x}^T = (a, b, c)$.

The problem now is to choose an error criterion J as a function of ϵ , then find \underline{x} which minimizes J using a Kiefer-Wolfowitz minimization algorithm.

The vector notation for eqn. (3.9) would be,

$$\underline{x}(n+1) = \underline{x}(n) - \frac{\gamma_n}{2\alpha_n} [\underline{J}^+ - \underline{J}^-] \quad (4.3)$$

where $\underline{J}^+ = [J(\hat{a}+\alpha(n), \hat{b}, \hat{c}), J(\hat{a}, \hat{b}+\alpha(n), \hat{c}), J(\hat{a}, \hat{b}, \hat{c}+\alpha(n))]^T$

$$\underline{J}^- = [J(\hat{a}-\alpha(n), \hat{b}, \hat{c}), J(\hat{a}, \hat{b}-\alpha(n), \hat{c}), J(\hat{a}, \hat{b}, \hat{c}-\alpha(n))]^T \quad (4.4)$$

The noise, e , is digitally generated as a series of numbers random in amplitude with a Gaussian distribution (0,0.5). That is, zero mean, variance of 0.5.

However, the input sequence, u , is a pseudo-random binary sequence whose sign is random with dependence on a uniform distributed random variable (also generated on the digital computer). Graphs illustrating the e, u sequences are shown in Fig. 4.1.

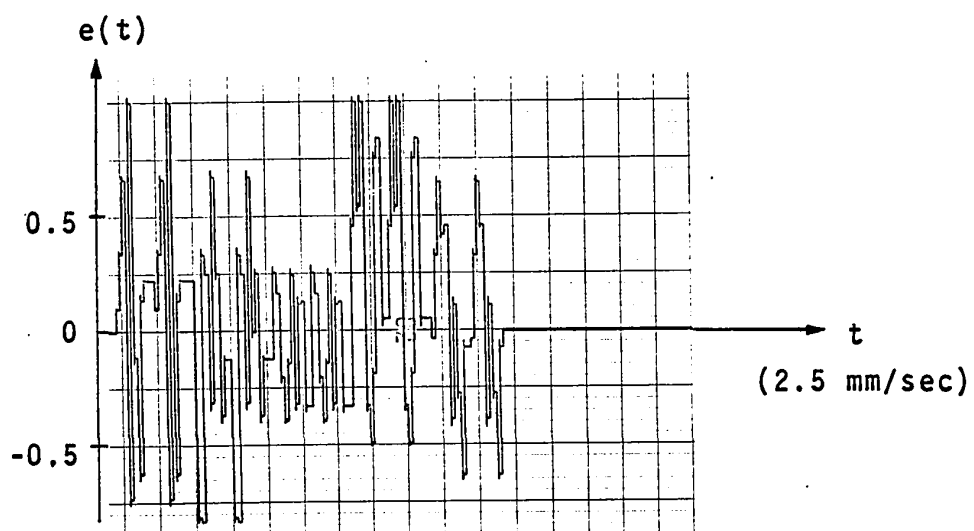
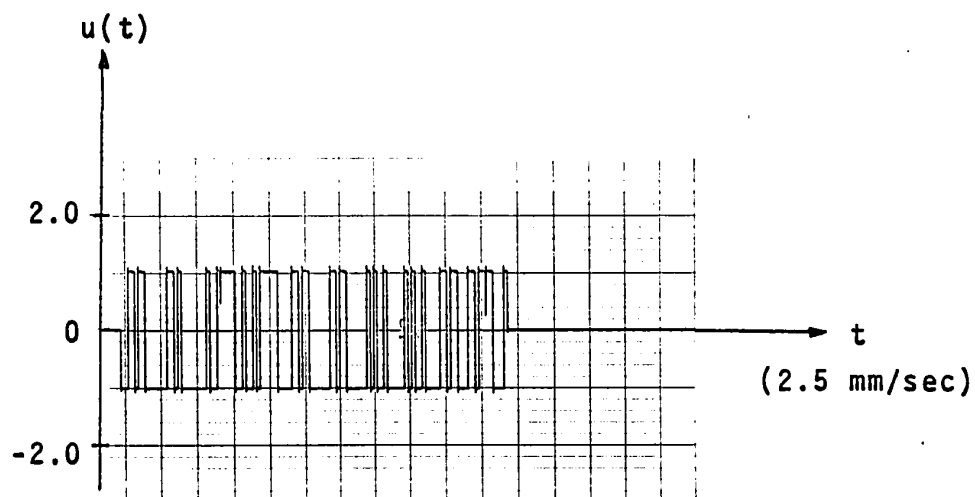


FIG. 4.1 Sequences $u(t)$ and $e(t)$ generated by digital computer.

To satisfy convergence theorems [23], sequences $\gamma(n)$ and $\alpha(n)$ must comply with restrictions (3.3). In this example α and γ are digitally generated from

$$\alpha(n) = \frac{E_1}{E_2 \cdot n^{PE} + E_3} \quad (4.5)$$

$$\gamma(n) = \frac{G_1}{G_2 \cdot n^{PG} + G_3}$$

where $E_1, E_2, E_3, PE, G_1, G_2, G_3$ and PG are constant for each identification process.

Two methods of evaluating the error, J , were considered:

a) $J = \epsilon^2$ (4.6)

In this case J is the instantaneous value of the squared error estimate ϵ .

Define

$$R(s) = \frac{y(s)}{s + \hat{c}} \quad (4.7)$$

$$W(s) = \frac{u(s)}{s + \hat{c}}$$

Substituting (4.7) into (4.2),

$$\epsilon(s) = sR(s) + \hat{a}R(s) - \hat{b}W(s) \quad (4.8)$$

Using eqn. (4.8), a significant reduction can be made on the number of analog computer components required to solve the problem. Only the terms $R(s)$, $sR(s)$ and $W(s)$ are generated on the analog computer. $\epsilon(s)$ is a linear combination of these terms

and can be calculated digitally, resulting in a saving on analog equipment.

To make an approximation of the gradient vector required for the Kiefer-Wolfowitz algorithm, define \underline{j}^+ and \underline{j}^- such that

$$\underline{j}^+ = \begin{vmatrix} sR(s) + (\hat{a} + \alpha(n))R(s) - \hat{b}W(s) \\ sR(s) + \hat{a}R(s) - (\hat{b} + \alpha(n))W(s) \\ sR^+(s) + \hat{a}R^+(s) - \hat{b}W^+(s) \end{vmatrix} \quad (4.9)$$

$$\underline{j}^- = \begin{vmatrix} sR(s) + (\hat{a} - \alpha(n))R(s) - \hat{b}W(s) \\ sR(s) + \hat{a}R(s) - (\hat{b} - \alpha(n))W(s) \\ sR^-(s) + \hat{a}R^-(s) - \hat{b}W^-(s) \end{vmatrix}$$

$$\text{where } R^+(s) = \frac{y(s)}{s + \hat{c} + \alpha(n)}$$

$$w^+(s) = \frac{u(s)}{s + \hat{c} + \alpha(n)}$$

$$R^-(s) = \frac{y(s)}{s + \hat{c} - \alpha(n)}$$

$$w^-(s) = \frac{u(s)}{s + \hat{c} - \alpha(n)}$$

The analog flow chart for the circuit required to calculate $W, R, sR, W^+, R^+, sR^+, W^-, R^-, sR^-$ is shown in Fig. 4.2.

These variables were digitally sampled at the same frequency ($\frac{1}{T}$) at which the inputs $e(t)$ and $u(t)$ were being updated. Thus \underline{j}^+ and \underline{j}^- are discrete variables rather than continuous, being evaluated every T seconds.

For the tests run, sequences $\alpha(n), \gamma(n)$ were chosen

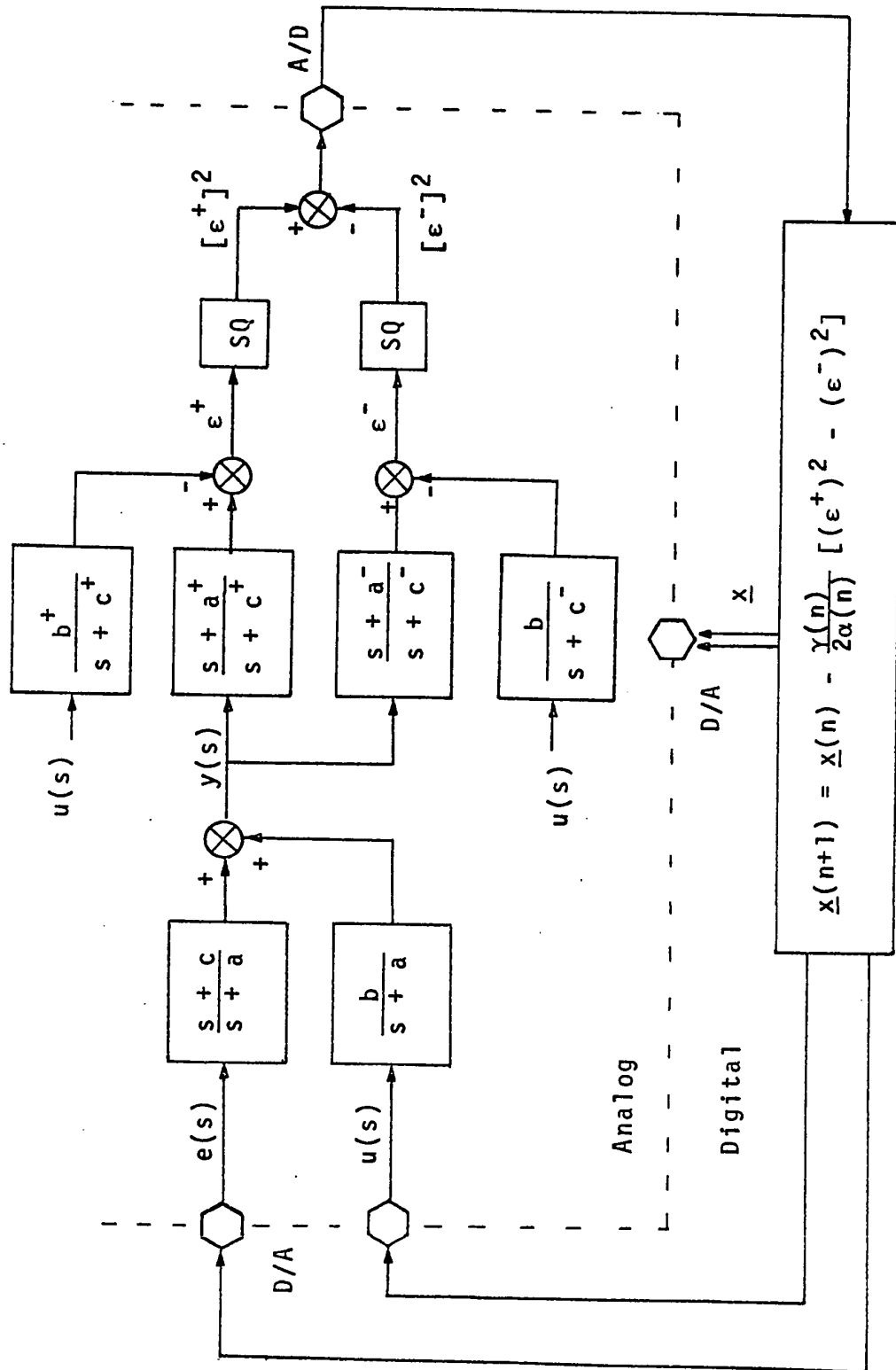


FIG. 4.2 Kiefer-Wolfowitz hybrid implementation ($J = \epsilon^2$).

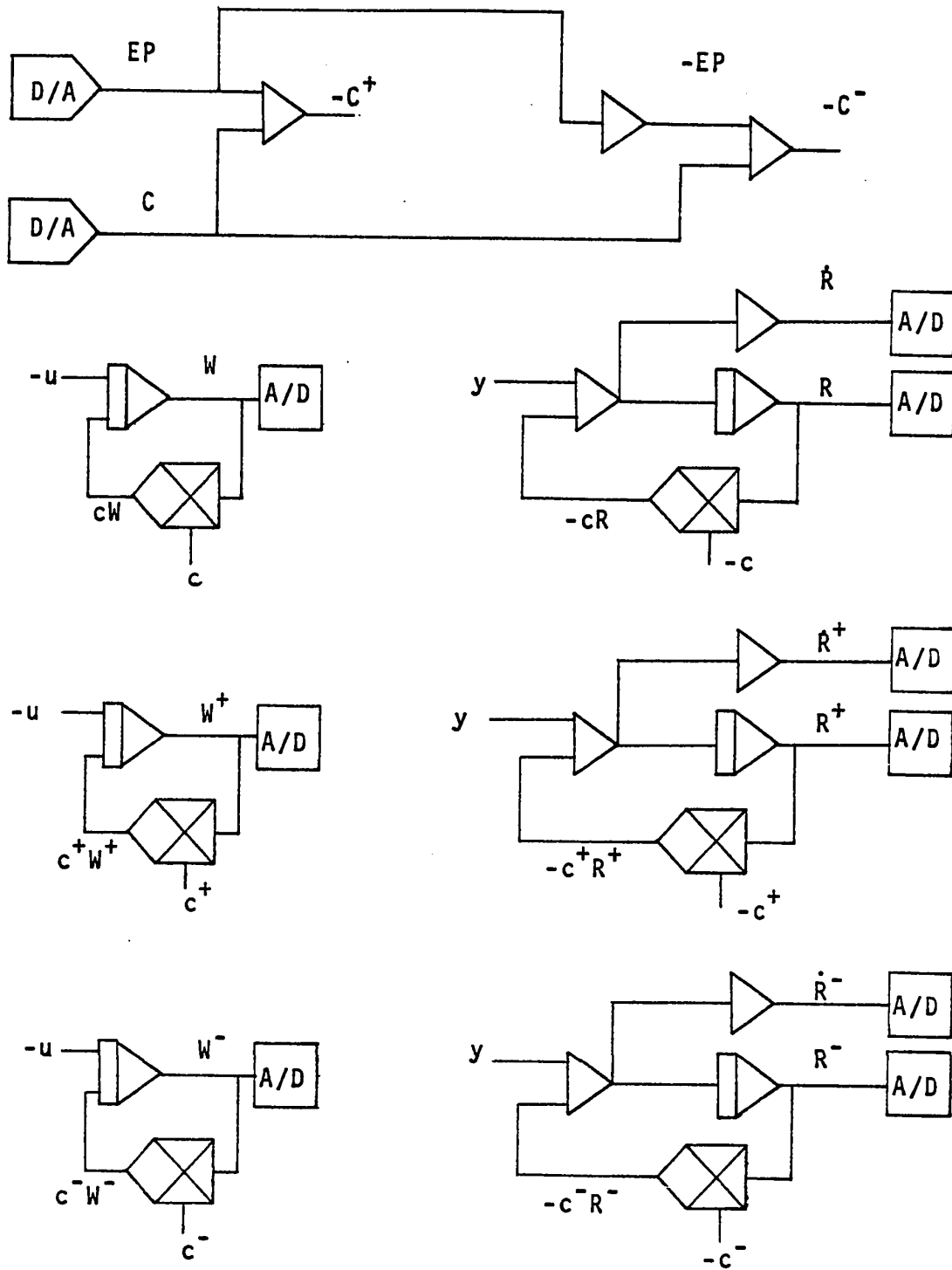


FIG. 4.3 Analog model used for Kiefer-Wolfowitz Algorithm ($J = \epsilon^2$) (amplitude scaling not shown).

according to eqn. (4.5) as

$$\alpha(n) = \frac{E_1}{n^{0.125}} \quad \gamma(n) = \frac{G_1}{n} \quad (4.10)$$

G_1 in this case was used as a gain control in the Kiefer-Wolfowitz algorithm (4.3).

Results of two estimates of "a" are shown in Fig. 4.4, one with noise source $e(t)$, removed from the system.

Fig. 4.5 shows estimates of a, b with the estimate of c set to a nominal value of 2.0. Here it was necessary to increase the algorithm gain (G_1) by a factor of 10 to obtain a reasonable identification on parameter b. However, this tended to decrease the stability of the estimation procedure for parameter a.

It was not possible to obtain a good estimate of the parameter c using this stochastic approximation method.

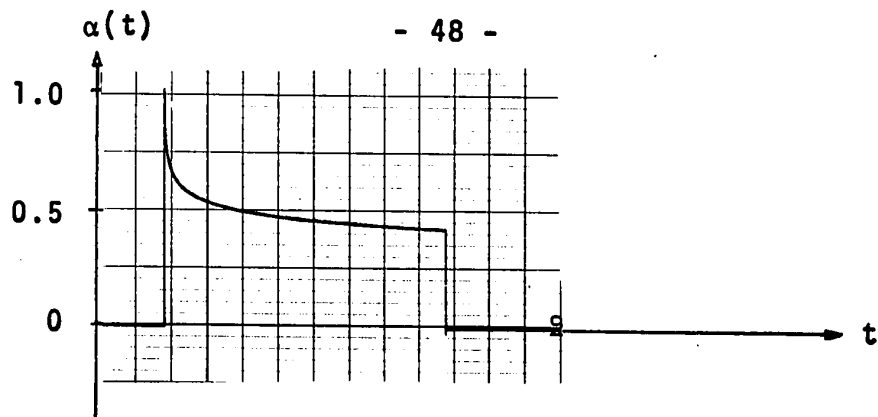
$$b) J = \int_0^{t_n} \hat{e}^2 dt$$

As pointed out in section 3.1, \hat{e} is not an instantaneous function of the parameter, but depends on the present and past history of both the system and the parameters. Thus J, as defined in eqn. (4.5), might introduce a certain amount of inaccuracy and lack of stability in the Kiefer-Wolfowitz algorithm.

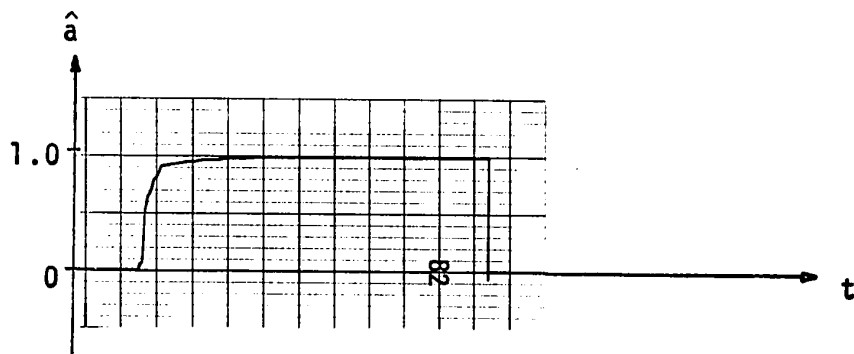
It was decided to use the more reasonable criterion function

$$J = \int_0^{t_s} \hat{e}^2 dt \quad (4.11)$$

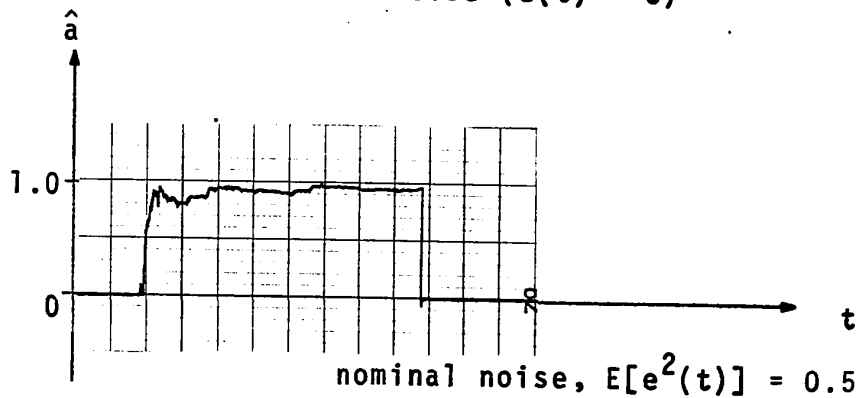
t = analog computer
time



→|200 seconds|←



no noise ($e(t) = 0$)

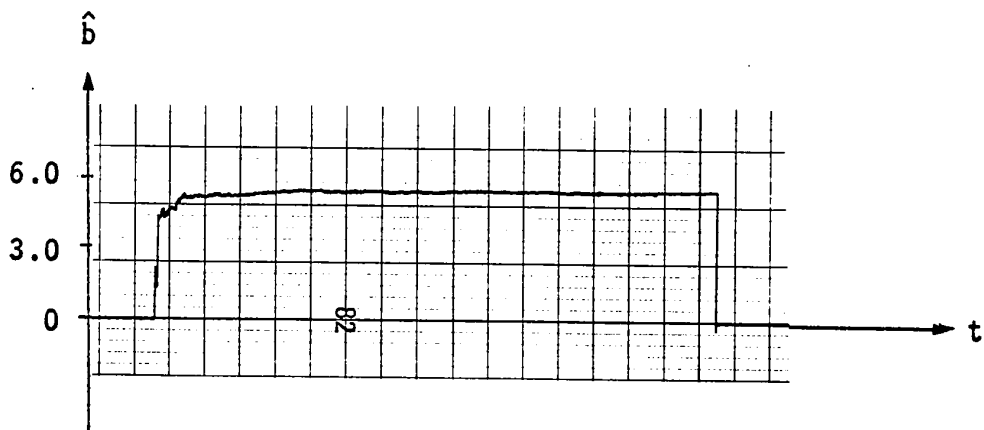
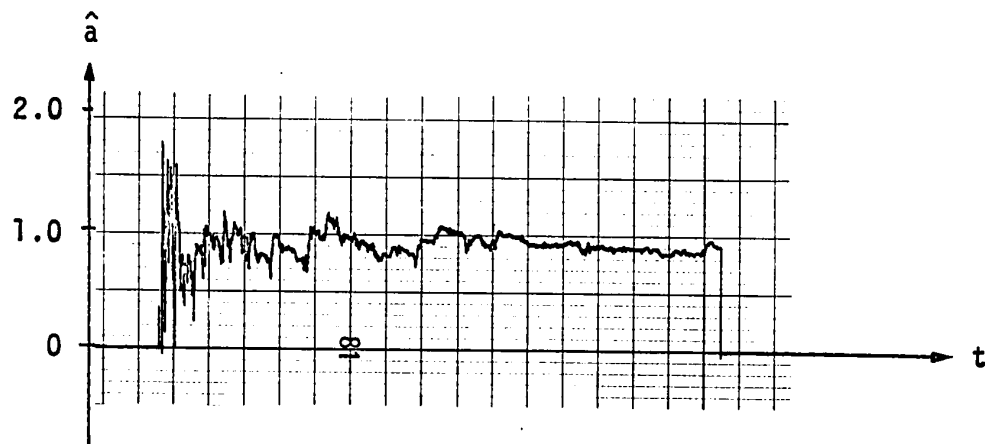


Time scale: 25 seconds/division

$E_1 = 10, G_1 = 10, N = 1000$

True values: $a = 1.0, b = 6.0, c = 2.0$

FIG. 4.4 Estimate of "a" parameter only, \hat{b} and \hat{c} fixed.



$$G_1 = 100, E_1 = 10, N = 2000$$

\hat{c} fixed at nominal value

Variance of noise = 0.5

True values: $a = 1.0, b = 6.0, c = 2.0$

Time scale: 25 seconds per division

FIG. 4.5 Estimate of both "a" and "b", \hat{c} fixed at nominal value.

In this case, t_s is the time interval over which J and its gradient are being evaluated. These values are then used for one iteration of eqn. (4.3).

As in the previous case, inputs to the system are the discrete random numbers

$$e_i, u_i$$

where $i = 1, 2, 3, \dots, N$

$$NT \gg t_s$$

e and u are generated by digital program every T seconds, the value of T being chosen for proper excitation of the system to be identified.

Note that the input sampling period T and the gradient evaluation interval t_s are independent. During the interval t_s there will occur t_s/T discrete random inputs. At completion of interval t_s , required values of the criterion function are samples and the analog re-initiated for the next iterative interval.

It is desirable to utilize the high-speed integration capabilities of the analog computer by requiring all calculations, excluding parameter update, to be performed on the 680 analog. However, a limited number of analog components were available, making a completely parallel analog calculation impossible for each iteration step.

To overcome this difficulty, a method of "time-sharing" analog equipment has been introduced. As shown in Figs. 4.6 and

and 4.8, one analog circuit evaluates J for each parameter estimation set (a^+, b^+, c^+) . At each iteration the desired vectors \underline{J}^+ and \underline{J}^- are obtained by integrating through the evaluation interval (t_s) six times, inserting the appropriate a^* , b^* , c^* each time according to eqn. (4.4). The FORTRAN digital program is listed in the appendix under the name K.W.#4.

Estimation results for the "a" parameter, holding b and c constant, are shown in Fig. 4.9. Parameter estimates for a and b , with c fixed, are shown in Fig. 4.9. The results are given for 200 iterations with the analog computer time-scaled for a speed-up factor 10. This is the fastest speed possible without exceeding digital program limits. Resulting time required for 200 iterations are quite large, 170 seconds in the case of the 2 parameter estimation (a,b) . This is approximately five times slower than the results obtained when using ϵ^2 as the error criterion.

In addition, only 10 input samples were taken for each integration interval when evaluating J . The sampling period was 0.01 seconds. Considering the random nature of the input signals, this was a rather small number of samples. A more reasonable selection might be 100 input samples per integration interval. Consider, as an example, a relatively large number of iterations (e.g. 1000) using 100 input samples per interval. For the two-parameter problem the computing time required would be approximately 1-1/3 hours. This type of computing time places severe restrictions on the experimentation with the method.

As in section (a), it was not possible to obtain estimates

for the parameter c . This may have been partly due to difficulties in analog amplitude scaling encountered during variations in the estimate of c .

It should be noted that bounds were placed on the parameter estimates due to amplitude scaling and stability considerations. The limits on "a" and "b" parameters were,

$$\hat{a}_{\max} = 9.0$$

$$\hat{a}_{\min} = 0.0$$

$$\hat{b}_{\max} = 9.0$$

$$\hat{b}_{\min} = 0.0$$

\hat{c} was held constant at 2.0.

The problems in choice of algorithm gain again appeared in this section, for the two-parameter case. Gain, $G_1 = 50$, seemed optimal for estimation of parameter "a", but this value was much too small for reasonable estimates of "b". The maximum gain possible, without causing estimates to be oversensitive, was $G_1 = 200$, a value still insufficiently large for successful results with parameter b. This indicates the desirability of introducing an "adaptive" gain based on the sensitivity of each parameter estimate.

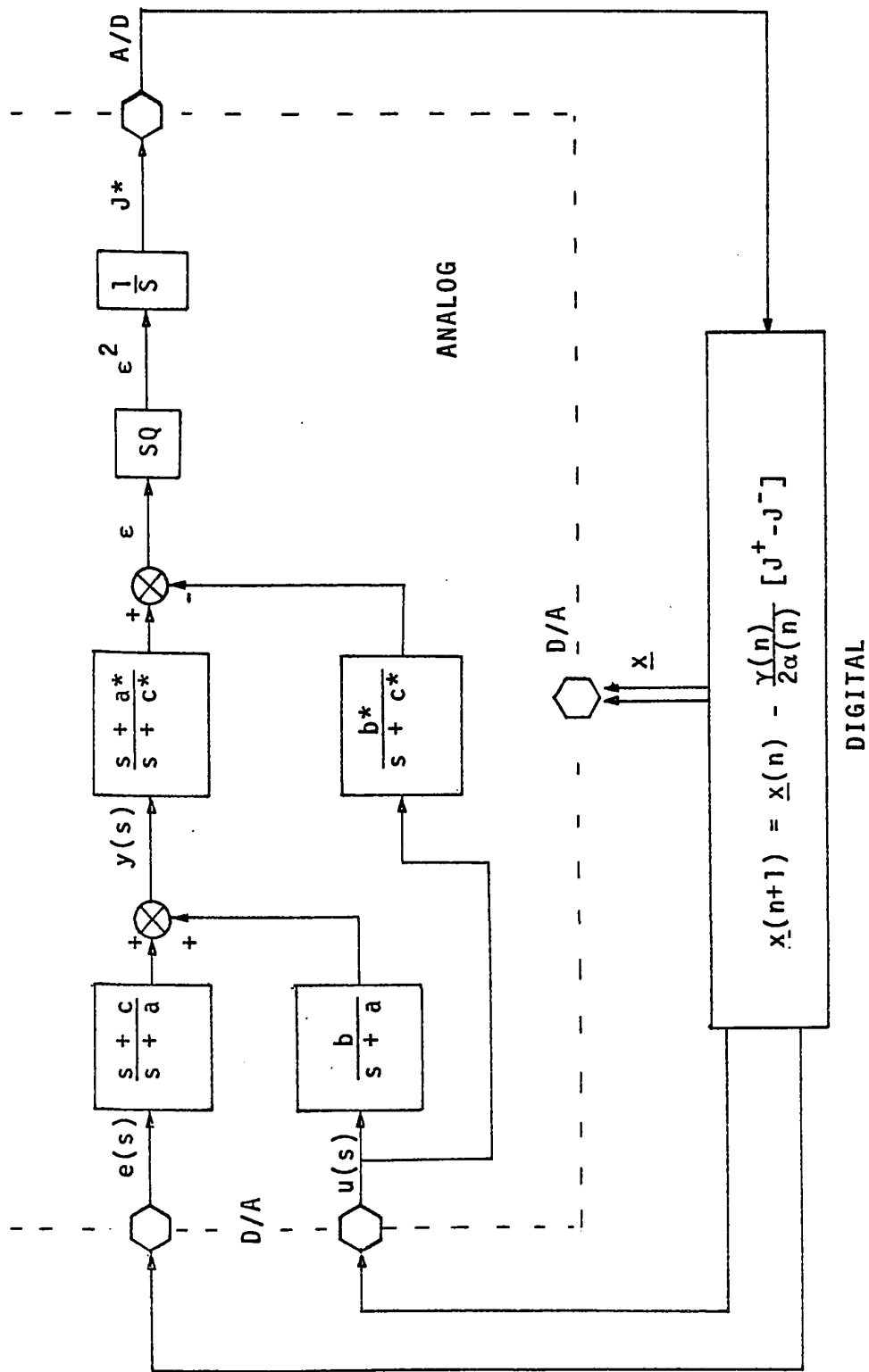


FIG. 4.6 Kiefer-Wolfowitz hybrid implementation: $J = \int \epsilon^2 dt$.

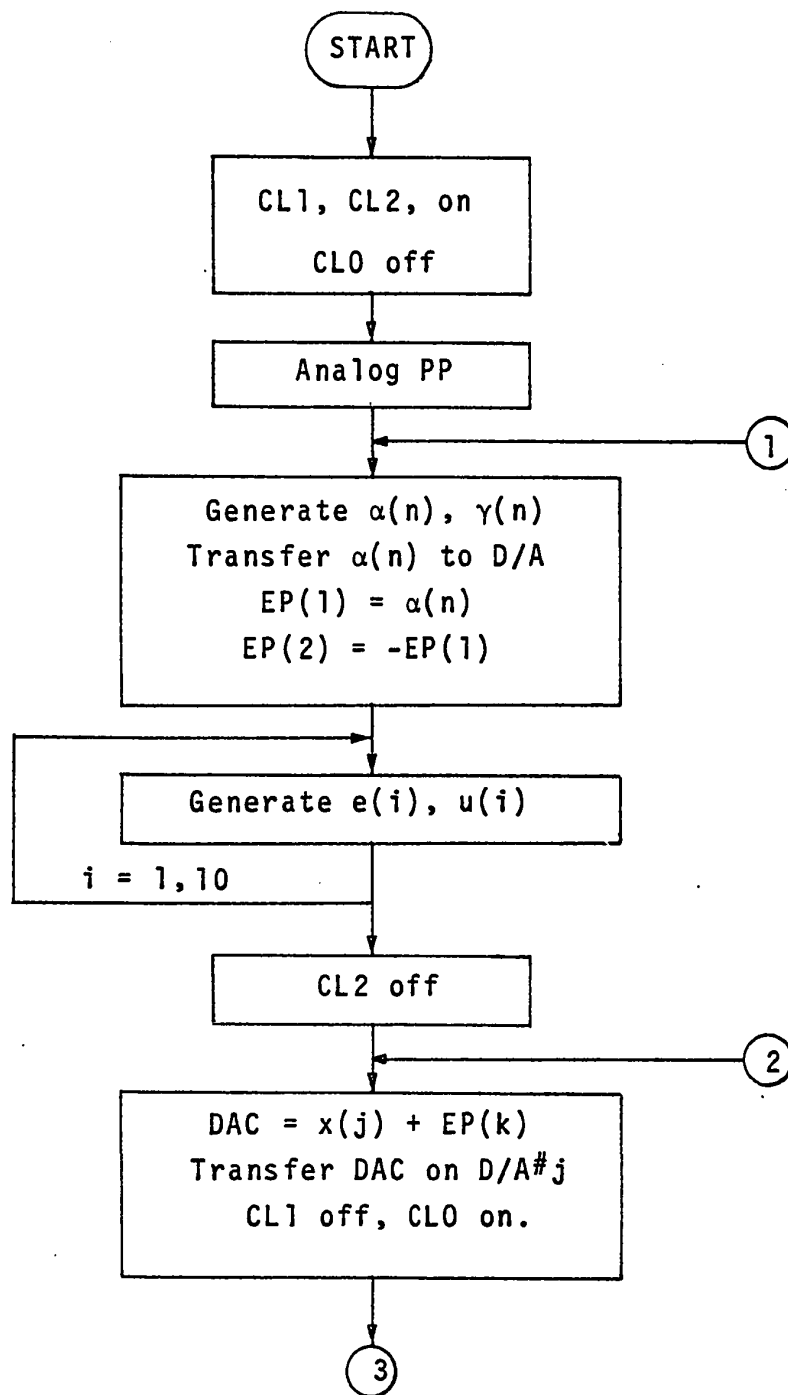


FIG. 4.7a Digital flowchart, Kiefer-Wolfowitz
algorithm: $J = \int e^2 dt.$

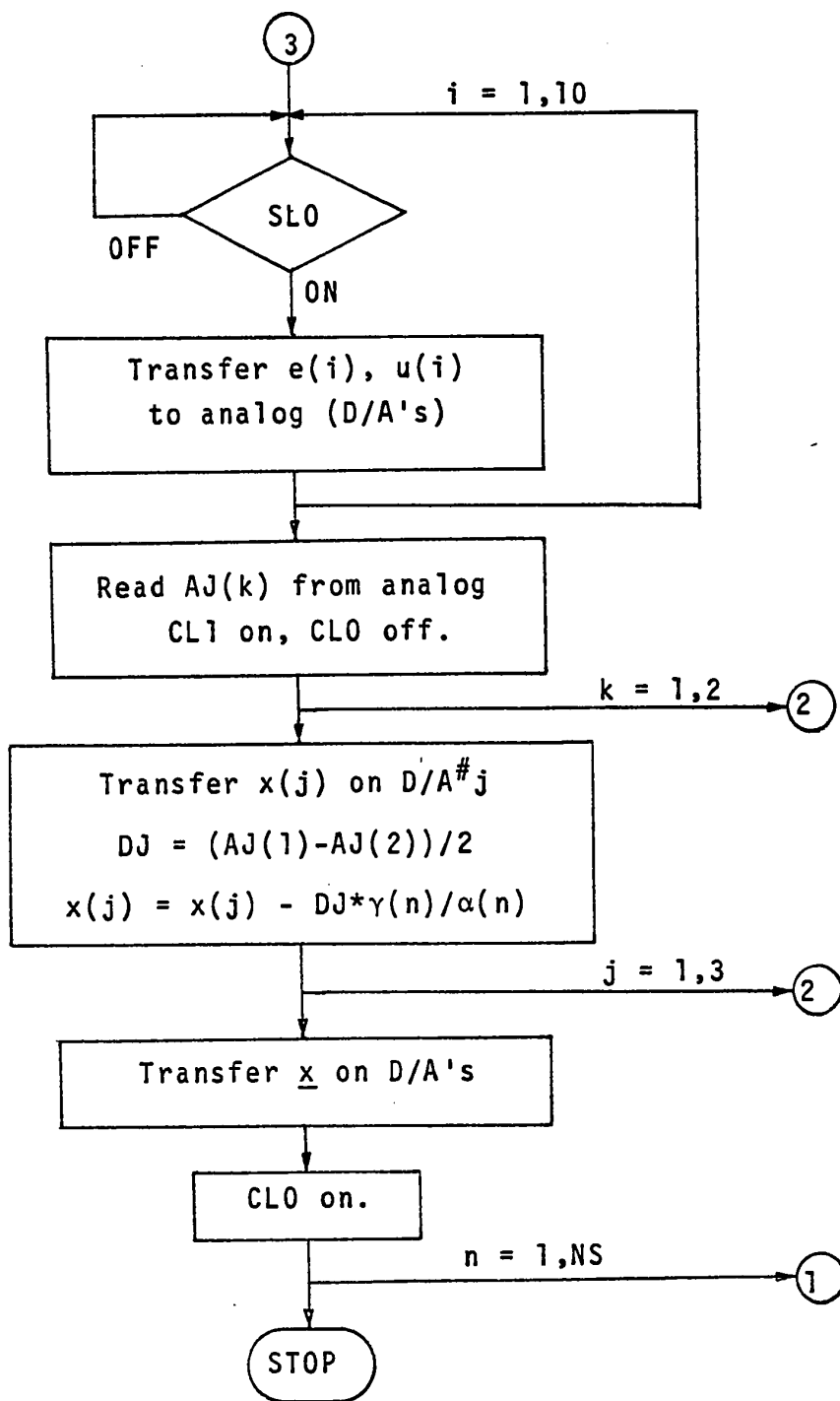


FIG. 4.7b Digital flowchart, Kiefer-Wolfowitz

algorithm: $J = \int e^2 dt.$

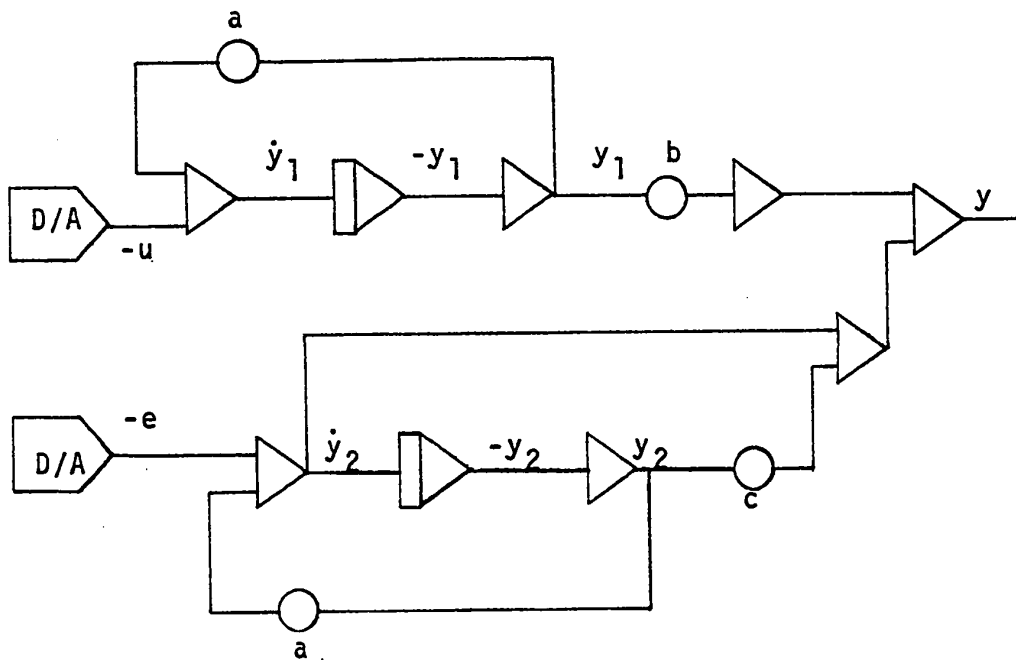


FIG. 4.8a Analog model and logic circuit used for Kiefer-Wolfowitz Implementation: $J = \int \epsilon^2 dt.$

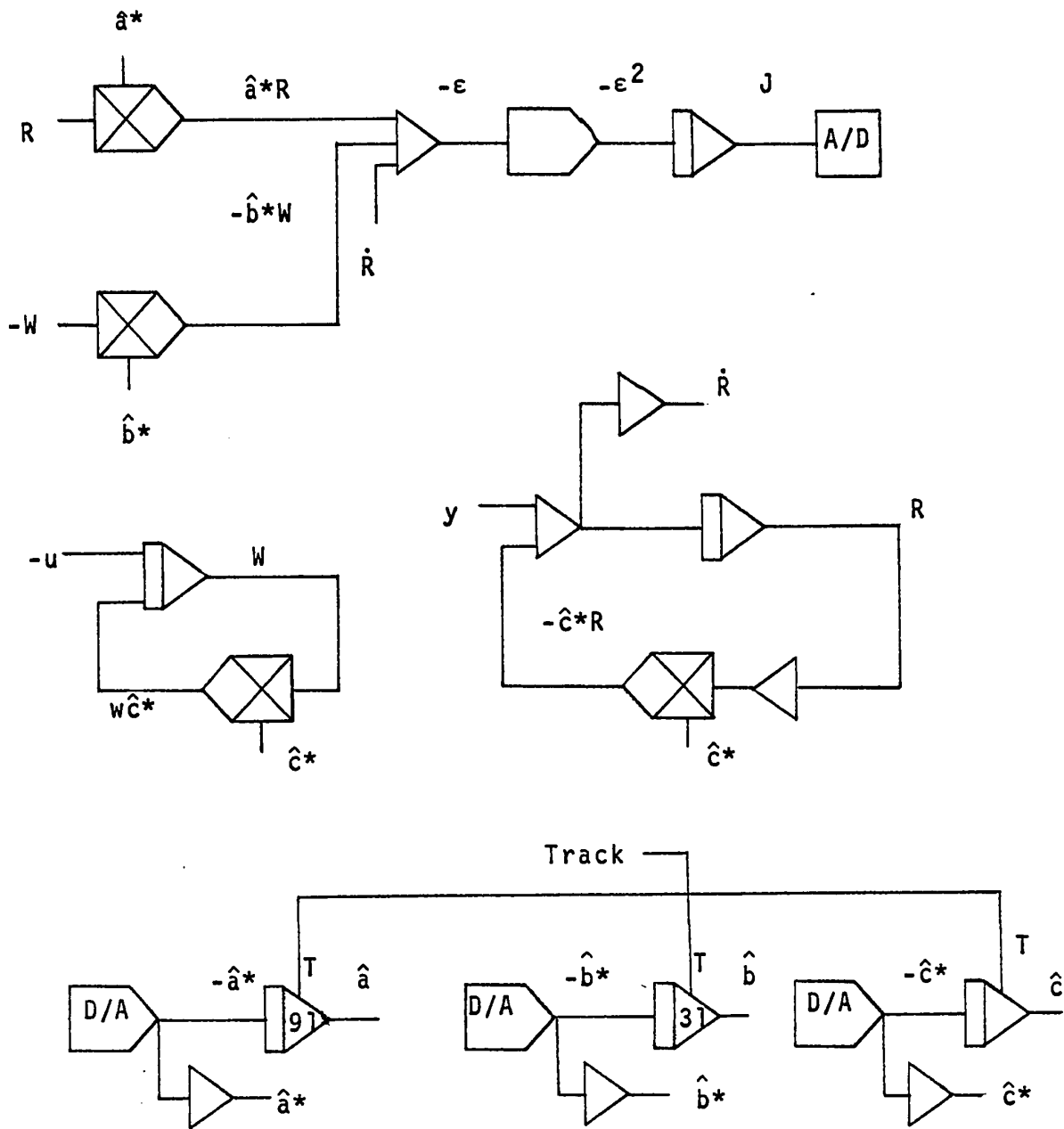
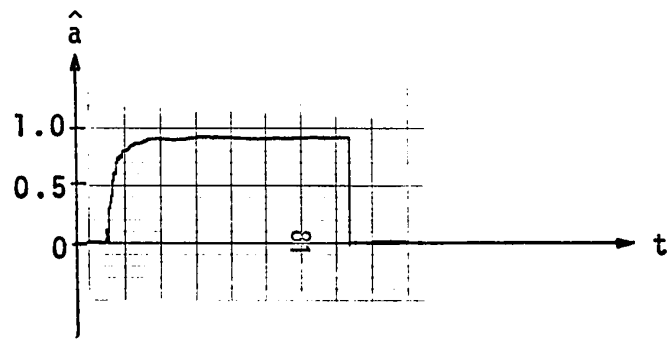


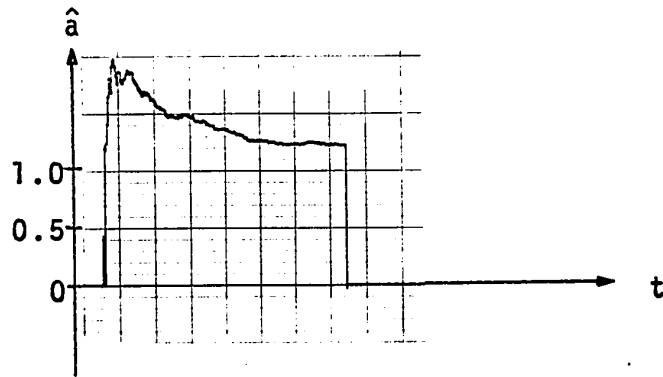
FIG. 4.8b Analog model and logic circuit for Kiefer-Wolfowitz implementation: $J = \int \epsilon^2 dt$.



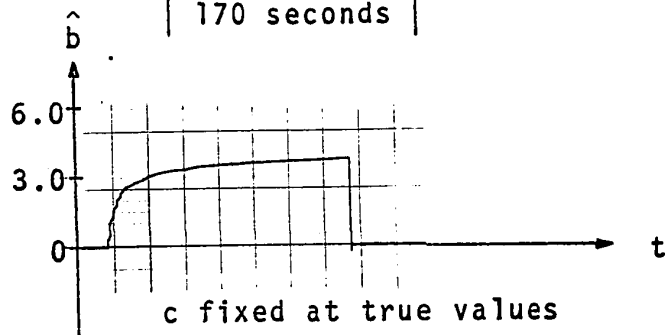
b, c fixed at true values

$$G_1 = 50$$

No. of iterations, $N = 200$



170 seconds



c fixed at true values

True values: $a = 1.0, b = 6.0$

$$N = 200, G_1 = 200$$

FIG. 4.9 Estimation of "a" and "b" parameters.

4.2 On-Line Stochastic Approximation Method (Scalar Gain)

Consider the following example process to be identified:

$$\begin{aligned} y_n - 1.5y_{n-1} + 0.7y_{n-2} = 1.0u_{n-1} + 0.5u_{n-2} + e_n - 1.0e_{n-1} + \\ + 0.2e_{n-2} \end{aligned} \quad (4.12a)$$

where e_n is a sequence of independent random variables with zero mean and unit variance.

Comparing with eqns. (3.12) and (3.14)

$$\begin{aligned} (1+a_1z^{-1} + a_2z^{-2})y_n = (b_0+b_1z^{-1} + b_2z^{-2})u_n + \\ + (1+c_1z^{-1} + c_2z^{-2})e_n \end{aligned} \quad (4.12b)$$

with $a_1 = -1.5$

$a_2 = 0.7$

$b_0 = 0.0$

$b_1 = 1.0$

$b_2 = 0.5$

$c_1 = -1.0$

$c_2 = 0.2$

The algorithm for the estimate of the parameter vector $\underline{\theta}$ is

$$\underline{x}(n+1) = \underline{x}(n) + G(n)[y(n) - \underline{x}(n)^T \hat{\underline{V}}(n)] \hat{\underline{V}}(n) \quad (4.13)$$

with $\underline{x}(n)$ and $\hat{\underline{V}}(n)$ defined by eqns. (3.17) and (3.19).

The problem, then, is to identify the parameter vector $\underline{\theta}$, choosing an appropriate gain sequence G_n .

4.2.1 Analog Simulation of Process to be Identified

Since most practical processes are continuous in nature, it is of interest to implement the algorithm by on-line sampling of a continuous real-time process. In this case, the process can be simulated by analog computer in real time, if eqn. (4.12) is converted from a discrete to a continuous process.

Eqn. (4.12) can be rewritten in Z-transform notation as

$$y(z) = \frac{B(z)}{A(z)} u(z) + \frac{C(z)}{A(z)} e(z) \quad (4.14)$$

where $A(z) = z^2 - 1.5z + 0.7$

$B(z) = z + 0.5$

$C(z) = z^2 - 1.0z + 0.2$

Transforming $B(z)/A(z)$ and $C(z)/A(z)$ into Laplace form will yield the necessary continuous transfer functions.

The equivalent transfer function in the analog system involves a zero-order sample and hold network at the input to the transfer function, while the output would also be sampled. This configuration is shown in Fig. 4.10.

Note that $u^*(s)$ is defined as the instantaneous sampled value of $u(s)$ and that,

$$Z[u^*(s)] = u(z)$$

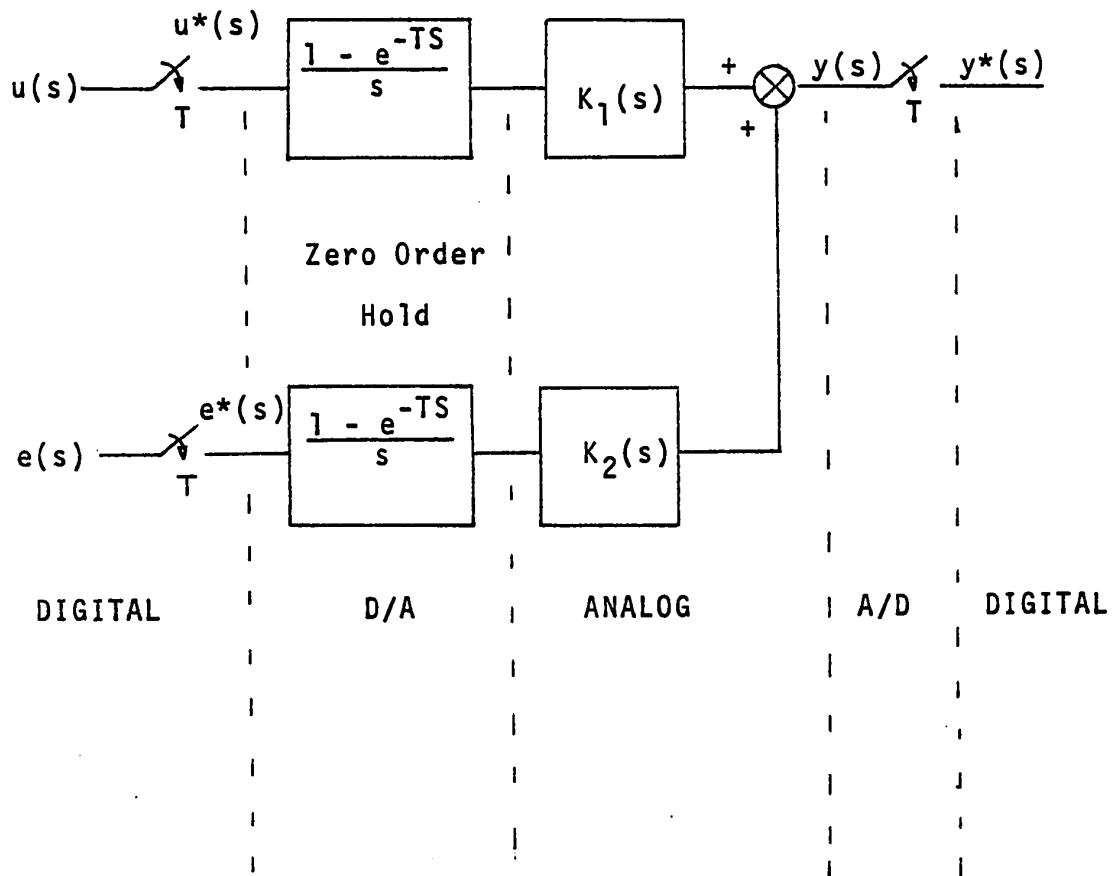


FIG. 4.10 Continuous representation of the discrete model.

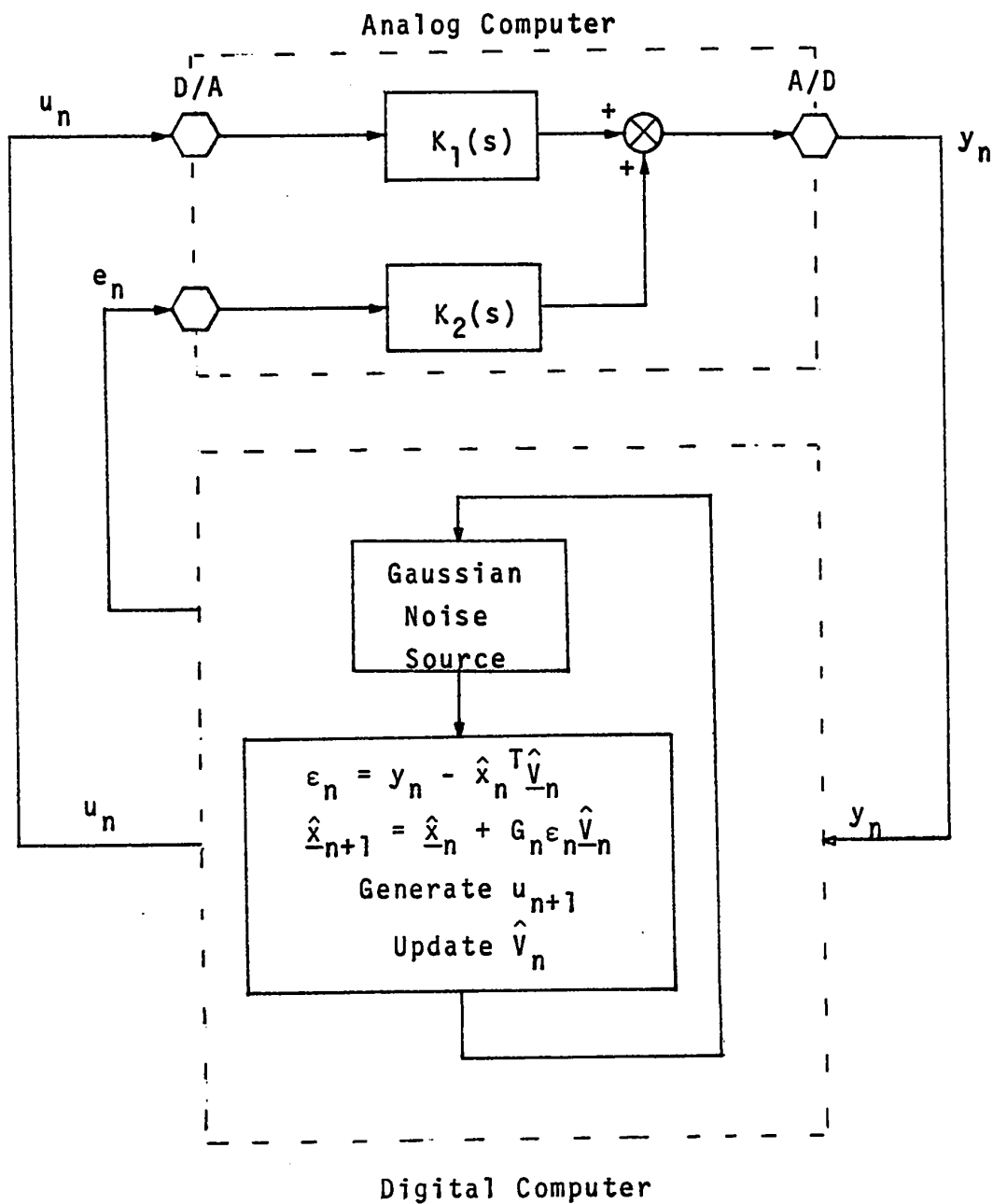


FIG. 4.11 Hybrid implementation of Panuska's stochastic approximation method.

$$Z[y^*(s)] = y(z)$$

$$Z[e^*(s)] = e(z)$$

where $z[\cdot]$ is the z-transform operator and T is the sampling period in seconds.

The problem before implementing this method is, given the discrete description of the system to be identified, find the transfer functions $K_1(s)$ and $K_2(s)$.

From Fig. 4.11,

$$\frac{K_1(s)}{s} (1 - e^{-Ts}) = \mathcal{L}\left[\frac{B}{A}(z)\right] \quad (4.15)$$

$$e^{-Ts} = z^{-1}$$

$$\therefore 1 - e^{-Ts} = 1 - z^{-1}$$

Interchanging the Laplacian and Z-transform operators in eqn. (4.15)

$$\frac{K_1(s)}{s} = z^{-1} \left[\frac{z}{z-1} \cdot \frac{B(z)}{A(z)} \right] \quad (4.16)$$

similarly $\frac{K_2(s)}{s} = z^{-1} \left[\frac{z}{z-1} \cdot \frac{C(z)}{A(z)} \right]$

Thus the parameters in the analog system may be calculated, given the discrete parameters.

Using residue theory,

$$f_1(nT) = \sum_{i=1}^{k_p} \text{Res} \frac{z^{nB}(z)}{(z-1)A(z)} \quad (4.17)$$

$$f_2(nT) = \sum_{i=1}^{k_p} \text{Res} \frac{z^n C(z)}{(z-1)A(z)}$$

evaluated at the k_p poles where $(z-1) \cdot A(z) = 0$.

The time domain functions $f_1(t)$ and $f_2(t)$ can be transformed using Laplace transform tables to give

$$\begin{aligned} K_1(s) = SF_1(s) &= \frac{d_3 s + d_4}{s^2 + d_1 s + d_2} \\ K_2(s) = SF_2(s) &= \frac{s^2 + d_5 s + d_6}{s^2 + d_1 s + d_2} \end{aligned} \quad (4.18)$$

$$\begin{aligned} \text{where } d_1 &= .3567/T \\ d_2 &= .2427/T^2 \\ d_3 &= .254/T \\ d_4 &= 1.812/T^2 \\ d_5 &= .9758/T \\ d_6 &= .2427/T^2 \end{aligned}$$

and T is the sampling period.

Analog simulation of functions K_1 , K_2 is shown in Fig. 4.13.

In order to facilitate analog programming, the following transfer functions have been simulated.

$$\frac{\Theta(s)}{u(s)} = \frac{\Phi(s)}{e(s)} = \frac{1}{s^2 + d_1 s + d_2} \quad (4.19a)$$

$$\frac{y_u(s)}{\Theta(s)} = d_3 s + d_4 \quad (4.19b)$$

$$\frac{y_e(s)}{\Phi(s)} = s^2 + d_5 s + d_6 \quad (4.19c)$$

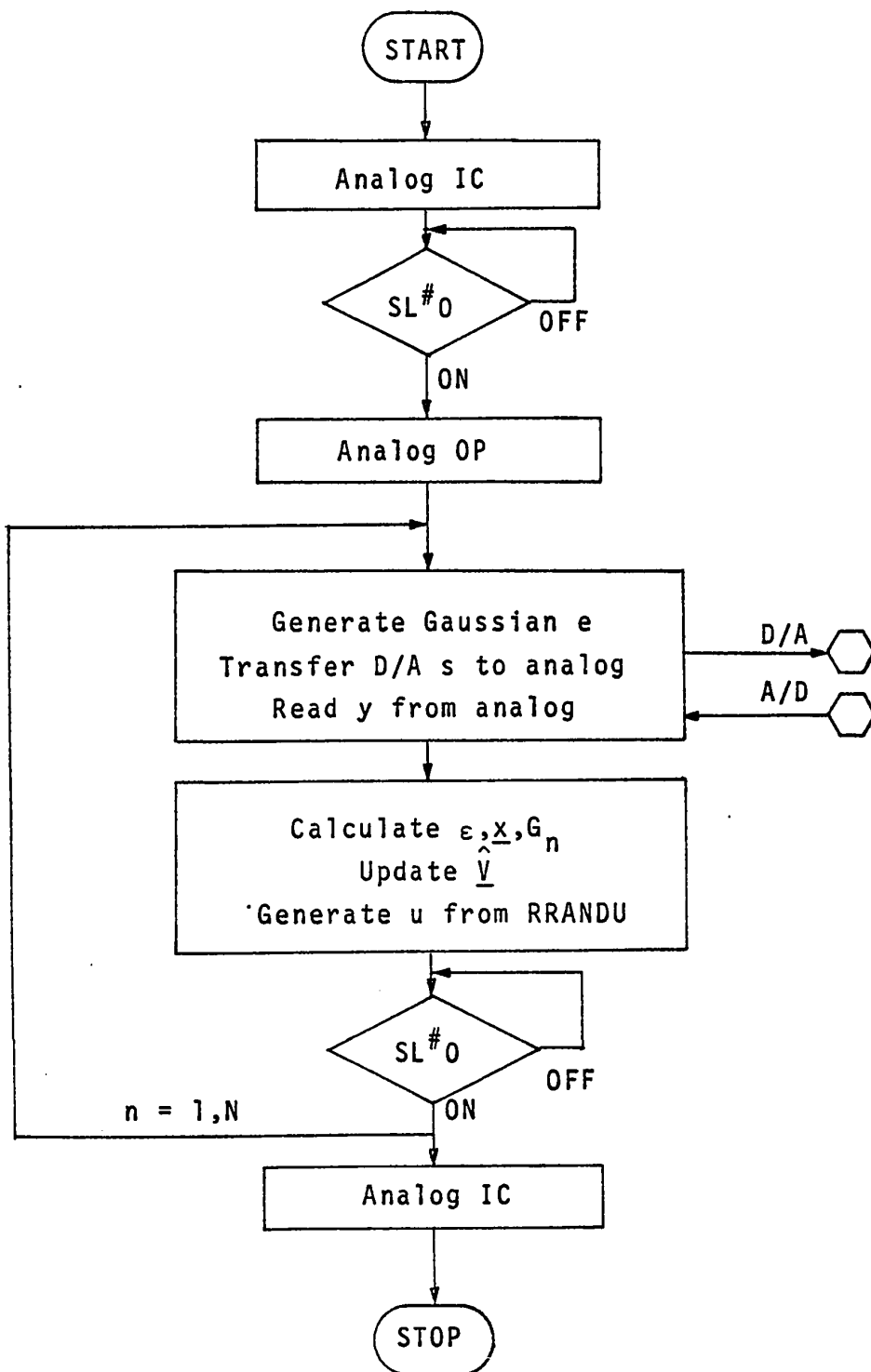


FIG. 4.12 Digital program flowchart: scalar gain method.

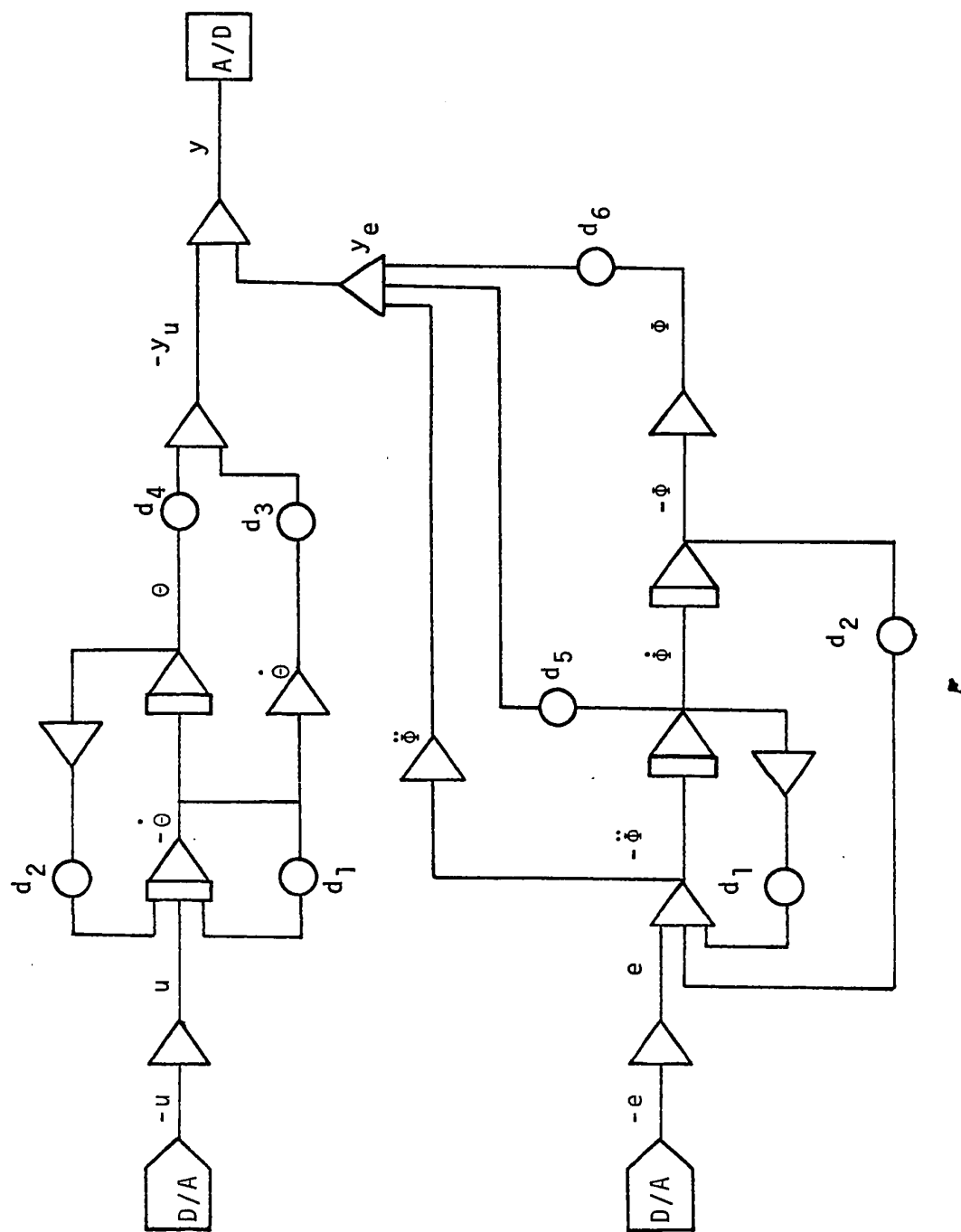


FIG. 4.13 Analog computer circuit for use with scalar gain algorithm.

$$y(s) = y_u(s) + y_e(s) = K_1(s)u(s) + K_2(s)e(s) \quad (4.19d)$$

A sampling period of 0.1 seconds was chosen to correspond to the maximum calculation speed of the digital program. $G_n = 1.0/n$, n is the current number of samples. Variations of parameter estimates over 200 samples are shown in Fig. 4.14.

Final estimates after 1000 samples yielded:

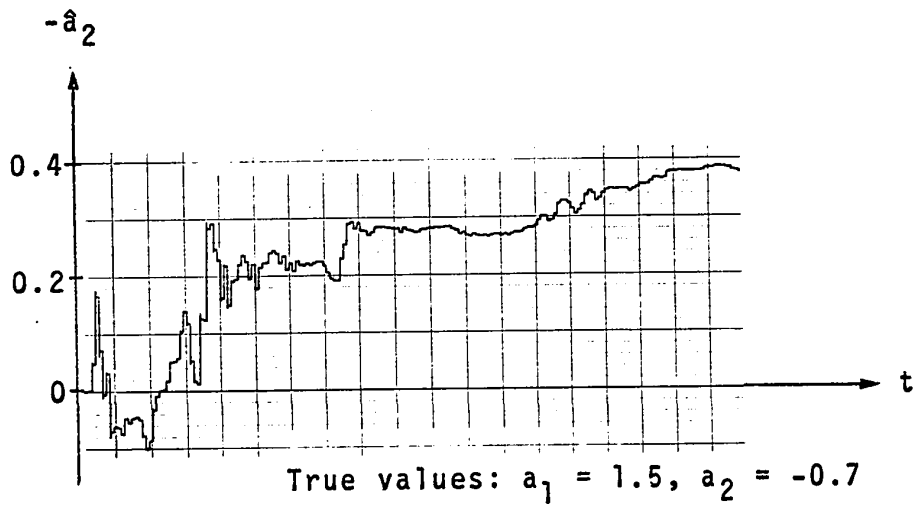
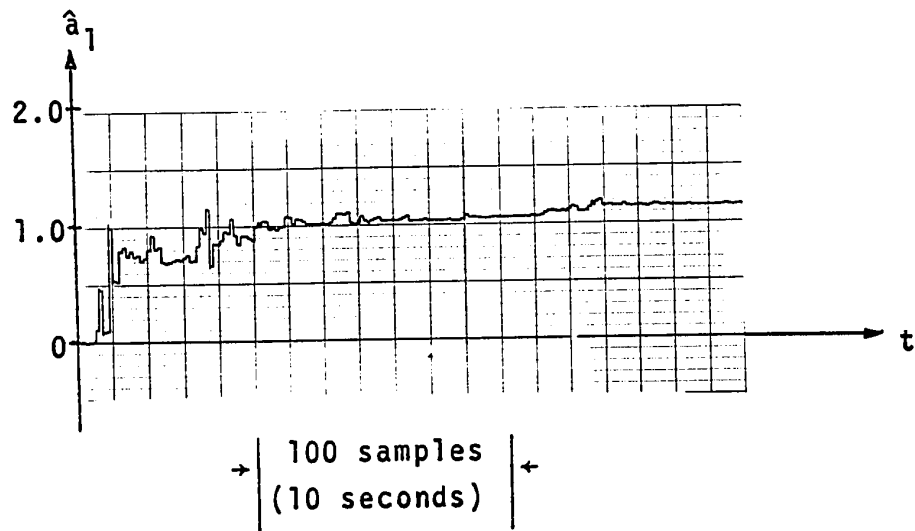
$$\begin{aligned}\hat{a}_1 &= 1.2260 \quad (1.5) \\ \hat{a}_2 &= -.4698 \quad (-0.7) \\ \hat{b}_0 &= .0502 \quad (0.0) \\ \hat{b}_1 &= .7984 \quad (1.0) \\ \hat{b}_2 &= .6623 \quad (0.5) \\ \hat{c}_1 &= -.1474 \quad (-1.0) \\ \hat{c}_2 &= .2554 \quad (0.2) \\ \hat{V}_e &= 1.02 \quad (1.0)\end{aligned}$$

where V_e is the variance of the noise, e . True values are in parentheses.

Increasing the number of samples did not significantly improve the parameter estimates. This is not surprising when consideration is given to the size of variable gain G_n as n increases, and to the restrictive 16 bit word size of the EAI 640 digital computer.

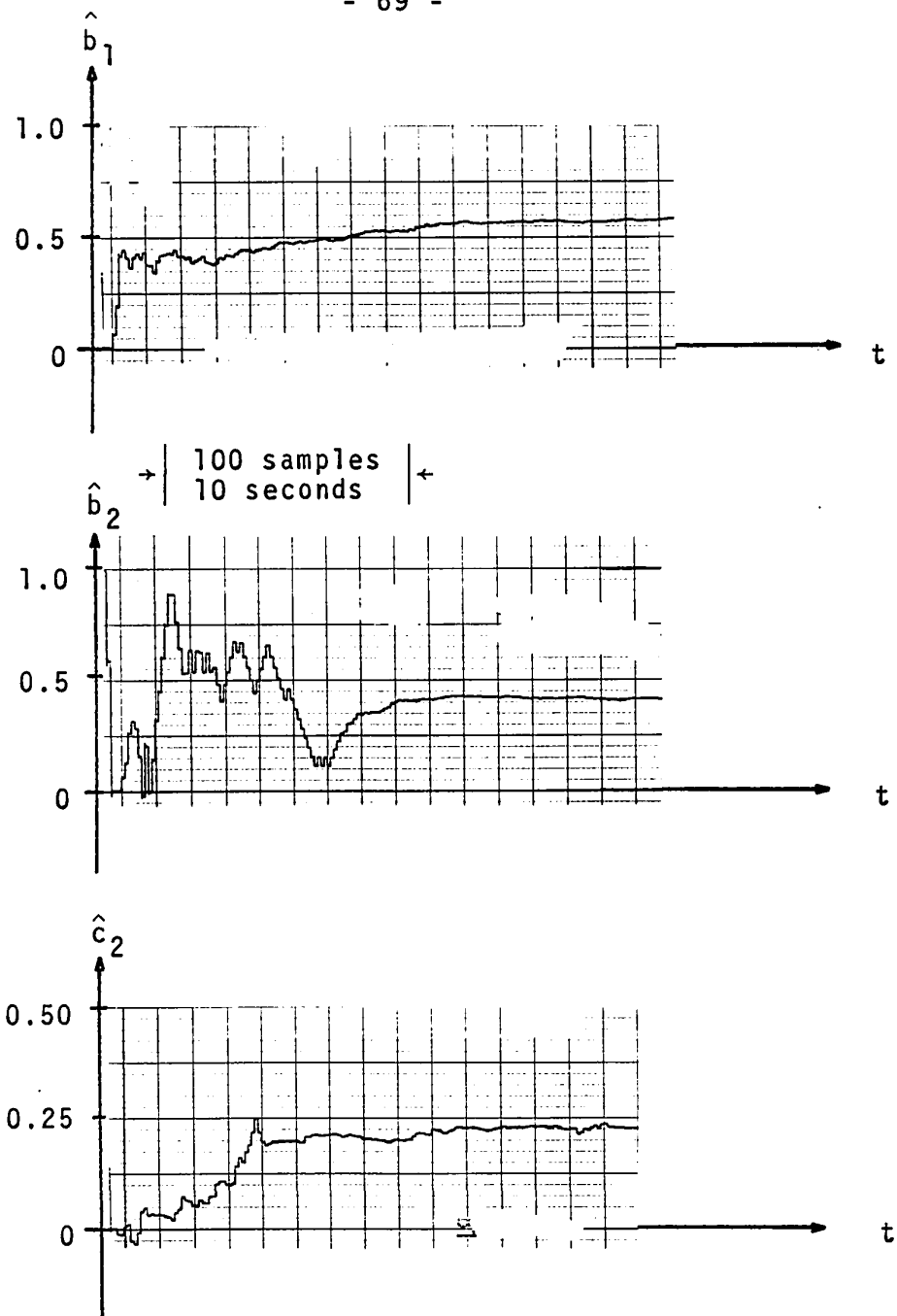
Gain G was increased but unstable oscillations in parameter estimates occurred over the first few samples.

An attempt was made to improve the algorithm accuracy of estimates by modifying gain sequence G_n such that



Time scale: 1 division per second

FIG. 4.14a Parameter estimates using scalar gain stochastic algorithm.



True values: $b_1 = 1.0$, $b_2 = 0.5$,
 $c_2 = 0.2$

Time scale: 1 division per second

FIG. 4.14b Parameter estimates using scalar gain stochastic algorithm.

$$G_n = \frac{1.0}{[A_n + 50(k_n - 1)]} \quad (4.20)$$

where A_n is set to the value 1.0 each time $n = k_n * \text{ANS}$.

Integer $k_n = 1, 2, 3, \dots$ such that $1 \leq A_n \leq \text{ANS}$,
ANS being a preselected number of samples.

The modified gain sequence resulted in drastic disturbances in parameter estimates at each "jump" in value of G_n . This was due to the particular value of noise disturbance e_n at time $n = k_n * \text{ANS}$.

The minimum solution time for 1000 samples was 100 seconds. The main limitation in computer speed for this problem is the digital program execution speed.

As can be observed in the estimation results, significant deviations from the true parameter values were evident. Part of the inaccuracy of estimates could be due to accuracy limits in analog computation (0.1%). The effect of computing delays between input and output sampling of the analog system was not determined. After the above considerations, plus the fact that more sophisticated gain algorithms could not be implemented due to an 8000 word core memory size restriction, it was decided to continue the remaining portion of this study on a digital-only program basis, using the more accurate 64-bit word size of a CDC 6400 computer.

4.2.2 Digital Computer Implementation

A FORTRAN program for testing the scalar gain stochastic algorithm (4.13) was written for a CDC 6400 computer. The program, called SCALAR GAIN, is listed in the appendix. Eqn. (4.12a) is again used as a second-order example.

The estimates in Table 1 were obtained with variations in gain sequence G_n . Estimates obtained after 1,000 samples required 2 seconds of computer CPU time.

As can be seen from Table 1, best results were obtained using a gain constant, G , of 1.0 for a large number of samples. Parameter estimates can be seen to converge asymptotically to the true parameter values. Increasing G to values greater than 1.0 increased the variance of the estimates. (However, the algorithm converges irrespective of the choice of gain constant). Another gain sequence was chosen such that:

$$G_n = \frac{1.0}{A_n + 5(k_n - 1)} \quad (4.21)$$

with A_n , k_n defined as for eqn. (4.20).

Variances of estimates for this gain sequence were quite large but the estimates tended to converge to true values over a large number of samples. Convergence was expected since

$$\sum_{n=1}^{\infty} G_n = \infty \quad \sum_{n=1}^{\infty} G_n^2 < \infty \quad (4.22)$$

satisfying the assumptions stated by Panuska [24].

Truncation in the parameter space can be introduced if some a priori knowledge is known about the parameters to be estimated. Truncation of the gradient is desirable, especially when estimate \underline{x}_n differs substantially from true value $\underline{\theta}$, to ensure steady, uniform corrections. For this example, bounds on the coefficient estimates were ± 5.0 . Truncation of the gradient was introduced by limiting the estimated error sequences to ± 10.0 .

Table 2 contains estimation results when using several runs through input-output records of fixed length. Values of estimates after the first pass were used as starting values for the second pass and so forth. Starting values for the first pass were zero, as were the model initial conditions.

Each set of results in Table 2 represents mean values and standard deviation of estimates computed by Monte Carlo method (from 20 different samples). These mean values of estimate samples tended to give more reasonable results than any one set of estimates. This is to be expected since desired estimates, \underline{x} , are random in nature with mean $\underline{\theta}$ and covariance matrix tending toward the lower bound given by the Cramer-Rao theory [2].

One set of results was obtained by defining the input sequence, u_n , to be a pseudo-random binary sequence (amplitude 1.0) rather than Gaussian $N(0,1)$. Very little difference was noted between estimates obtained using the two methods of input generation. However, CPU computer time was reduced by 15%

TABLE 1. Parameter Estimates for Second Order Model; Scalar Gain Algorithm
 Computing time for 10,000 samples: 20.5 seconds
 for 1,000 samples: 1.9 seconds.

Input Signal: Gaussian, (0,1)
 Noise: Gaussian, (0,1)
 Bounds: Coeff \pm 5.0
 Errors \pm 10.0

$\frac{G}{G} = \frac{G}{G}$	$G = \frac{A_n + 5K}{G}$	Length of record	a_1	a_2	b_1	b_2	c_1	c_2	var_e
			-1.500	0.700	1.000	0.500	-1.000	0.200	1.0
			\hat{a}_1	\hat{a}_2	\hat{b}_1	\hat{b}_2	\hat{c}_1	\hat{c}_2	\hat{var}_e
x	1.0	1000	-1.460	0.656	1.102	0.553	-0.885	0.159	6.96
x	1.0	5000	-1.484	0.689	1.043	0.545	-0.959	0.206	1.92
x	1.0	10000	-1.481	0.699	1.022	0.527	-0.973	0.217	1.48
x	2.0	1000	-1.602	0.749	1.051	0.379	-1.156	0.247	10.4
x	2.0	5000	-1.530	0.730	1.020	0.462	-1.061	0.136	2.86
x	2.0	10000	-1.516	0.723	0.996	0.462	-1.052	0.174	1.95
			-1.357	0.536	0.908	0.758	-0.618	0.127	15.7
x	1.0	1000	-1.441	0.704	0.979	0.492	-0.976	0.646	4.97
x	1.0	5000	-1.482	0.716	1.030	0.499	-1.063	0.134	3.05
x	1.0	10000							

TABLE 2. Parameter Estimates for Second Order Model; Scalar Gain Algorithm
Multiple Passes Through Fixed Record Length.

Input Signal: $u = N(0,1)$, Gaussian
 $u = \text{SRAN}$, Pseudorandom Binary Sequence

Length of record $N = 500$ δ - standard deviation of estimates

Bounds: Coeff ± 5.0
Errors ± 10.0

u = SRAN	u = N(0,1)	g	Number of passes	a ₁		a ₂		b ₁		b ₂		c ₁		c ₂		Computing time, secs.
				+δ		+δ		+δ		+δ		+δ		+δ		
				+δ		+δ		+δ		+δ		+δ		+δ		
				+δ		+δ		+δ		+δ		+δ		+δ		
	x	2.5	1	-1.308	.019	0.531	.009	0.920	.020	0.800	.069	-0.480	.037	0.031	.011	16.7
	x	2.5	5	-1.465	.001	0.652	.001	0.986	.002	0.573	.005	-0.845	.004	0.087	.003	36.0
	x	1.0	5	-1.385	.013	0.586	.011	1.002	.006	0.683	.077	-0.608	.038	0.047	.003	35.7
	x	5.0	5	-1.472	.002	0.661	.001	0.987	.001	0.552	.006	-0.845	.003	0.088	.003	35.0
x		5.0	5	-1.466	.001	0.673	.002	0.988	.003	0.549	.008	-0.837	.002	0.079	.003	29.6

when using the binary sequence since its sign was set according to a uniformly distributed (rather than Gaussian) random variable.

It was noted that, during the 5 passes through the data, the estimates showed a trend to change further even on the fifth and final pass. One reason for this is that repeated passes through the same data of finite length N are equivalent to the case when the original sequence of errors, e_n , is periodic with period N [38].

The digital program for Table 2 results can be found in the appendix under the name TESPAN.

4.3 Implementation of Adaptive Recursive-Least-Square Algorithm (Matrix Gain)

Restating the example problem to be solved:

Given the process described by (4.12a),

$$\begin{aligned} y_n - 1.5y_{n-1} + 0.7y_{n-2} = 1.0u_{n-1} + 0.5u_{n-2} + \\ + e_n - 1.0e_{n-1} + 0.2e_{n-2} \end{aligned} \quad (4.23)$$

where e_n is a sequence of independent random variables with zero mean and unit variance.

Given a sequence of observed input-output pairs, $\{u, y\}$, and using the identification scheme described by eqns. (3.24), (3.25) and (3.26), find an estimate \underline{x} of system parameter vector

θ

$$\underline{\theta}^T = (1.5, -0.7, 1.0, 0.5, -1.0, 0.2) \quad (4.24)$$

The problem was solved using the FORTRAN program "MATRIX GAIN" listed in the appendix. Results are listed in Table 3. For the example studied, the starting value for matrix Γ was set to the unit matrix. The sequence of inputs $\{u_n\}$ was defined as Gaussian distributed random variable with zero mean and unit variance as was noise sequence $\{e_n\}$, noting the required independence of the two random variables. Bounds on parameter estimates were set at ± 5.0 , while limits of ± 10.0 were chosen for the calculated error, EH .

Runs were made for cases where the starting values for parameter estimates were: (a) set to zero; (b) estimated from an initial least-squares estimate (linear regression) on a block of the first one hundred input-output data pairs.

The second case was made possible by observing the fact that the expected value of e_n is defined as zero, and approximating the system equation by the model:

$$\underline{z} = \eta^T \underline{x} \quad (4.25)$$

assuming no noise sequence, e_n , appearing in the model (for the first 100 sample only).

$$\underline{x}^T = (a_1, a_2, b_1, b_2) = (1.4, -0.7, 1.0, 0.5) \quad (4.26a)$$

$$\underline{z}^T = (z_1, z_2, \dots, z_n) \quad (4.26b)$$

$$\eta = (V_1, V_2, \dots, V_n) \quad (4.26c)$$

Observation vector \underline{V}_n was defined by

$$\underline{V}_n^T = (y_{n-1}, y_{n-2}, u_{n-1}, u_{n-2}) \quad (4.26d)$$

Then the initial estimate of parameters was

$$\underline{x} = [\eta \eta^T]^{-1} \eta \underline{z} \quad (4.27)$$

which gave starting values for estimates of a_1 , a_2 , b_1 and b_2 . Initial coefficient estimates for c_1 and c_2 were taken as zero.

During trial runs, it was noted that the value of the initial parameter estimates did not significantly affect the performance of the method, since, for each trial, the initial value of matrix Γ used in the algorithm was set to the unit matrix. Thus, several iterations were required for stabilization of Γ regardless of choice of initial parameter estimates. In theory [20], Γ tends to the covariance matrix of the parameter estimates as the number of iterations increases. This implies that an optimum choice of initial Γ (related to starting values of parameter estimates) can be derived. Time limitations precluded further study of this aspect of the problem.

Referring to Table 3, it can be seen that within 500 iterations, reasonable parameter estimates were obtained on the a and b parameters. The c parameters, more difficult to obtain because the corresponding elements in the \underline{V} (observation vector)

TABLE 3. Estimates of Parameters for Second Order Model (Matrix Gain)

True	1.500	-0.700	1.000	0.500	-1.000	0.200	1.00
Number of Samples	a_1	a_2	b_1	b_2	c_1	c_2	Variance of error
Initial Estimate*	1.272	-0.509	1.034	0.676	0.000	0.000	-
500	1.505	-0.692	1.071	0.385	-0.738	0.012	1.07
1000	1.512	-0.696	1.026	0.370	-0.872	0.026	1.06
2000	1.530	-0.719	1.003	0.420	-0.945	0.084	1.06
5000	1.521	-0.719	1.017	0.423	-0.984	0.115	1.04
10000	1.520	-0.716	0.997	0.462	-0.995	0.149	1.02

Computer Execution Time: N = 1000, 13.1 seconds

N = 10000, 131.7 seconds

*Initial Least Squares Estimate Assuming c_1 , c_2 zero.

can only be estimates of e_n , required a large number of iterations before reasonable estimates could be obtained.

During the first several hundred iterations, the variance of ϵ , the estimated error sequence, was much lower than values obtained when using the SCALAR GAIN algorithm described in section 4.2.2.

4.4 Aström's Maximum Likelihood Technique

Again the system described in (4.12a) is used as a test example. Aström's [3] method was implemented using eqns. (3.37) to (3.40) with residuals, $\epsilon(t)$, defined by (3.35) and with α taken as unity.

Runs were made for both cases where u was a pseudo-random binary signal with amplitude (± 1.0), sign being Gaussian dependent in nature, and where u was generated as a sequence of independent normal (0,1) random variables. Little difference was observed in results when interchanging the two methods of generating u .

The program package for these tests were set up the following way:

PRO - controls the iteration procedure according to eqn. (3.27)

VVIVZ - calculates $J(\theta)$, J_θ and $J_{\theta\theta}$ according to above formulas (3.39 to 3.42)

GJRV - inverts asymmetric matrices using a Gauss-Jordan technique.

These programs are listed in the appendix under the

the heading IDEN1. The user main program, (IDEN1), provides input-output data pairs and generates the necessary calls to PRO.

A typical call to PRO is as follows:

CALL PRO(NO,NI,NP,LI,L2,IT,AC1,AC2,IPRINT)

where the arguments are described in the program listing.

For this example, convergence parameters AC1 and AC2 were set to 10^{-5} and 10^{-2} respectively.

Some of the results are given in Tables 4 and 5 including the parameter estimates, together with estimates of their uncertainties, (σ_i) .

Notations: J loss function for 2nd order model.
ex exact second derivatives are used.
 σ_i standard deviation of i coefficient estimates.

The minimum values of the cost function, J, can be used to estimate the variance of the noise sequence $\{e_n\}$.

By definition,

$$J = \frac{1}{2} \sum_{i=1}^N e_i^2 \quad (4.28)$$

where N is the number of input-output pairs (data samples).

For 1000 data samples, J_{\min} was found to be 458.5.

$$E[e^2] \approx \frac{2}{N} J_{\min} = 0.922 \quad (4.29)$$

This value corresponds accurately to the computer calcu-

TABLE 4. Parameter Estimates After Each Iteration Using Aström's Maximum Likelihood Method

N = 1000 u = normal (0,1)

Step	a ₁	a ₂	b ₁	b ₂	c ₁	c ₂	J
0	0	0	0	0	0	0	10918
1	-1.24	0.46	0.96	0.77	0	0	810
2	-1.40	0.60	0.97	0.60	-0.40	-0.97	611
3	-1.482	0.678	0.980	0.509	-0.705	-0.756	503
4	-1.498	0.702	0.997	0.486	-0.986	0.154	459.95
5	-1.499	0.701	1.000	0.471	-1.011	0.195	458.49
6	-1.4990	0.7006	1.0009	0.4718	-1.0115	0.1941	458.49
7	-1.4990	0.7006	1.0008	0.4717	-1.0115	0.1944	458.49
8ex.	-1.4990	0.7006	1.0008	0.4717	-1.0115	0.1944	

σ_i 0.0090 0.0075 0.0301 0.0403 0.0332 0.0318

TABLE 5. Final Parameter Estimates Resulting from Application of Aström's Technique

No. of Samples	-1.50 a ₁	0.70 a ₂	1.00 b ₁	0.50 b ₂	-1.00 c ₁	0.20 c ₂	Computer time sec.
300	-1.502 ± .017	0.706 ± .014	1.005 ± .057	0.469 ± .078	-0.978 ± .057	0.092 ± .061	6.7
500	-1.486 ± .013	0.689 ± .011	1.015 ± .044	0.553 ± .059	-0.945 ± .045	0.156 ± .043	9.6
1000	-1.499 ± .009	0.701 ± .008	1.001 ± .030	0.472 ± .040	-1.012 ± .033	0.194 ± .032	20.8

lated standard deviation of errors, $\lambda = 0.967$.

From Table 4 it can be observed that the repeated use of approximate second derivatives was sufficient for obtaining maximum accuracy possible in the parameter estimates. Using exact derivatives near the completion of the run did not significantly modify the estimates although the gradient values became quite small, indicating closeness to the minimum point. It may have been possible to shorten the computing time required if exact second derivatives were used at an earlier iteration count.

As shown in Table 5, parameter estimates improved with increasing length of data record and the standard deviation of estimates was observed to be inversely proportional to N .

A study by Gustavson [18] using Aström's method employed stability testing routines for the C polynomial and required transformation routines to ensure that the data (to be used for identification) had zero mean. However, for the example shown here, these additional techniques were not necessary for satisfactory results. In fact, the C polynomial was quite well-behaved in all test cases.

4.5 Algorithm Using Fletcher-Powell Minimization

The example problem described earlier was solved using the Fletcher-Powell minimization technique described in section 3.2.3. Restating the process to be identified:

$$\begin{aligned} y_n - 1.5y_{n-1} + 0.7y_n = 1.0u_{n-1} + 0.5u_{n-2} + \\ + e_n + c_1e_{n-1} + c_2e_{n-2} \end{aligned} \quad (4.30)$$

with u and e defined previously.

Program IDEN4 was written to utilize the IBM minimizing routine FMFP. IDEN4, in addition to generating input output data pairs to be used for identification, supplied FMFP with required values of cost function being minimized and gradient of the cost function.

In addition to the previous example problem (4.30), the case was considered where noise in system output, y , was due to measurement errors only.

$$y_n - 1.5y_{n-1} + 0.7y_{n-2} = 1.0u_{n-1} + 0.5u_{n-2} + e_n \quad (4.31)$$

This is equivalent to setting the C polynomial in eqn. (3.12) to unity.

For identification of the system described in (4.30), the gradient calculations were set up as follows:

Consider the computed residuals $\{\epsilon(t), t = 1, 2, \dots, N\}$ defined from (3.12) as

$$C(z^{-1})\epsilon(t) = A(z^{-1})y(t) - B(z^{-1})u(t) \quad (4.32)$$

Since the function being minimized is defined as,

$$J = \frac{1}{N} \sum_{t=1}^N \epsilon^2(t) \quad (4.33)$$

it follows that the required gradient is

$$\frac{\partial J}{\partial x_i} = \frac{2}{N} \sum_{t=1}^N \epsilon(t) \frac{\partial \epsilon(t)}{\partial x_i} \quad (4.34)$$

where \underline{x} is the vector of parameter estimates.

Differentiating (4.32) gives,

$$\begin{aligned} c(z^{-1}) \frac{\partial \epsilon(t)}{\partial a_j} &= z^{-j} y(t) \\ c(z^{-1}) \frac{\partial \epsilon(t)}{\partial b_j} &= -z^{-j} u(t) \\ c(z^{-1}) \frac{\partial \epsilon(t)}{\partial c_j} &= -z^{-j} \epsilon(t) \end{aligned} \quad (4.35)$$

Using a state variable representation of (4.35), the following recursive equation can be easily implemented.

$$\begin{aligned} g_1(t+1) &= - \sum_{i=1}^2 c_i x_i(t) + y(t) \\ g_3(t+1) &= - \sum_{i=1}^2 c_i x_{2+i}(t) - u(t) \\ g_5(t+1) &= - \sum_{i=1}^2 c_i x_{4+i}(t) - \epsilon(t) \end{aligned} \quad (4.36)$$

$$g_2(t) = g_1(t-1), \quad g_4(t) = g_3(t-1), \quad g_6(t) = g_5(t-1)$$

Components of the gradient vector can be defined as,

$$\frac{\partial J}{\partial x_i}(t) = \epsilon(t) g_i(t) \quad (4.37)$$

Results are shown in Table 6 for 1000 input-output

TABLE 6. Parameter Estimates Resulting from Fletcher-Powell Minimization

Iterations required	D%	a ₁	a ₂	b ₁	b ₂	c ₁	c ₂	J	Computing time sec.
50	0	1.524	-0.712	0.979	0.482	-1.011	0.199	0.968	36.1
10	20	1.490	-0.694	1.081	0.472	-1.042	0.273	1.013	9.1
50	30	1.568	-0.751	1.019	0.273	-0.838	0.437	1.300	34.8
	True	1.500	-0.700	1.000	0.500	-1.000	0.200	1.000	

D is deviation of starting values from true values.

No. of data.samples, N = 1000.

data pairs for identification using both models (4.30).

It was found necessary to use starting values lying within a restricted range around the true parameter values, since the accuracy of estimates appeared to depend on the starting values. For parameter starting values deviating more than 30% from true values, very poor convergence was observed.

For starting values within approximately 30% of true values, the maximum specified 50 iterations were reached. Extending the iteration limit was not of interest since the increments in parameter estimations were extremely small near the end of the run.

For starting values within 20% of true values, the run was terminated after 10 iterations due to an overflow condition within subroutine FMFP, indicating a zero divisor. At this point, the parameter estimates appeared to be converging and had attained reasonable values.

Results obtained from these runs were obviously not as accurate as the parameter estimates shown in Table 4 (Aström's method). Both the Fletcher-Powell and Aström methods employed hill-climbing techniques for minimization. The Fletcher-Powell technique employs an iterative algorithm for calculation of matrix H where H tends to the inverse Hessian of the cost function at its minimum. No matrix inversion routines are required. However, the Aström method calculates the Hessian matrix on each estimation iteration. Matrix inverses are required, but computations are rather easy when estimating relatively few parameters.

Thus, accuracy of the Fletcher-Powell method depends

somewhat on the speed of convergence of H to the Hessian matrix.

It was noted that, when using starting values at 20% of the true parameters, the C polynomial became unstable during one of the iterations. For the runs presented in Table 6, no bounds were placed on the parameter estimates and no stability tests on the A and C polynomials were made. A possible improvement might be the detection of instability conditions with the option of modifying the iteration step size.

In addition to the results presented in Table 6, trial runs were made using approximate gradient equations. The gradient of the loss function was approximated as follows:

$$\begin{aligned}\frac{\partial J(t)}{\partial a_i} &= \frac{2}{N} \sum_{t=1}^N \epsilon(t)y(t-i) \\ \frac{\partial J(t)}{\partial b_i} &= - \frac{2}{N} \sum_{i=1}^N \epsilon(t)u(t-1) \\ \frac{\partial J(t)}{\partial c_i} &= - \frac{2}{N} \sum_{i=1}^N \epsilon(t)\epsilon(t-1)\end{aligned}\tag{4.38}$$

The difference between (4.38) and (4.37) is that part values of estimated error, ϵ , are not included. In other words, the C polynomial in eqn. (4.35) is set to unity. This approximation should be reasonable near the minimum J. Results obtained showed no improvement over those shown in Table 6.

Since it was required to choose starting values within some suitable range of true parameter values, it would seem appropriate to use a combination of the Fletcher-Powell method with other parameter identification methods. A method such as

the "matrix gain" technique in section 4.3 could be used initially to obtain reasonable starting values, followed by the powerful search technique of Fletcher-Powell. This work remains to be done.

4.6 Electric Power System Load Modelling

This section presents a practical application of parameter identification techniques for the development of a probability model for the load of an electric power system. Development of such a model is useful for short-term load forecasting.

The model [29] assumes the power load is given by the sum of a periodic discrete time series with a period of 24 hours and a residual term. The latter is characterized by the output of a discrete time dynamical linear system driven by a white random process and a deterministic input, u , which is determined by a non-linear function of the actual and normal temperatures. The periodic component of the load depends on the time of day and the day of the week.

The hypothesis is made that the load, q , at any hour of the day is

$$q(t) = y_p(t) + y(t) \quad (4.39)$$

$y(t)$ is defined as the residual component (including measurement uncertainty and temperature effects) while y_p is the period component of the load. y_p is assumed a deterministic process so that its exact value is determinable from its model. The

periodic component describes that part of the load which goes through a 24 hour near-periodic cycle which rises in the morning, peaks at mid-morning, drops until late afternoon, rises again during the evening, and finally drops considerably at night.

The structure of the periodic component can be expressed as a time series,

$$y_p(t) = xp_0 + \sum_{i=1}^{n_p} \{xp_i \sin[2\pi i/24]t + xp(n_{p+i}) \cos[2\pi i/24]t\} \quad (4.40)$$

which can be written in vector form as

$$y_p(t) = \underline{\Phi}^T(t) \underline{x}_p \quad (4.41a)$$

where defining

$$\omega_0 = 2\pi/24 \quad (4.41b)$$

then,

$$\underline{\Phi}^T(t) = [1, \sin\omega_0 t, \dots, \sin n_p \omega_0 t, \cos\omega_0 t, \dots, \cos n_p \omega_0 t] \quad (4.41c)$$

$$\underline{x}_p^T = [xp_0, xp_1, xp_2, \dots, xp_{(2n_p)}] \quad (4.41d)$$

while t stands for the hour of the day.

Vector \underline{x}_p is assumed constant Monday through Friday over a span of three weeks. Any longer span could introduce considerable error since normal load consumption does vary over the seasons. Data for Saturdays and Sundays are not included (as would be expected).

ted) since plant shutdowns, increases in power for private homes, etc., cause redistribution of load consumption. Due to the short span of time over which the model can be determined, the maximum data record length possible is 288 data points.

The residual component of the load, y , is an uncertain process, time varying and correlated with itself as well as with certain weather effects described by u . The following relationship is proposed between temperature effects u and inherent uncertainty in load:

$$y(t) = \sum_{i=1}^n a_i y(t-i) + \sum_{j=0}^m b_j u(t-j) + e(t) \quad (4.42)$$

$e(t)$ is assumed a zero-mean white process.

For the effect of temperature on load, the input to the model will be defined by $u(T_e, \hat{T}_e)$, given by Fig. 4.15, taken from ref. [29], where T_e is the actual temperature and \hat{T}_e is the normal temperature.

The model tested was of the form,

$$q(t) = y_p(t) + y(t) \quad (4.43)$$

$$y_p(t) = 1500 + 100 \sin(2\pi t/24) + 100 \cos(2\pi t/24) \quad (4.44)$$

and

$$y(t) = 1.4 y(t-1) - 0.49 y(t-2) + 3 u(t) + y(t-1) + e(t) + u(t-1) \quad (4.45)$$

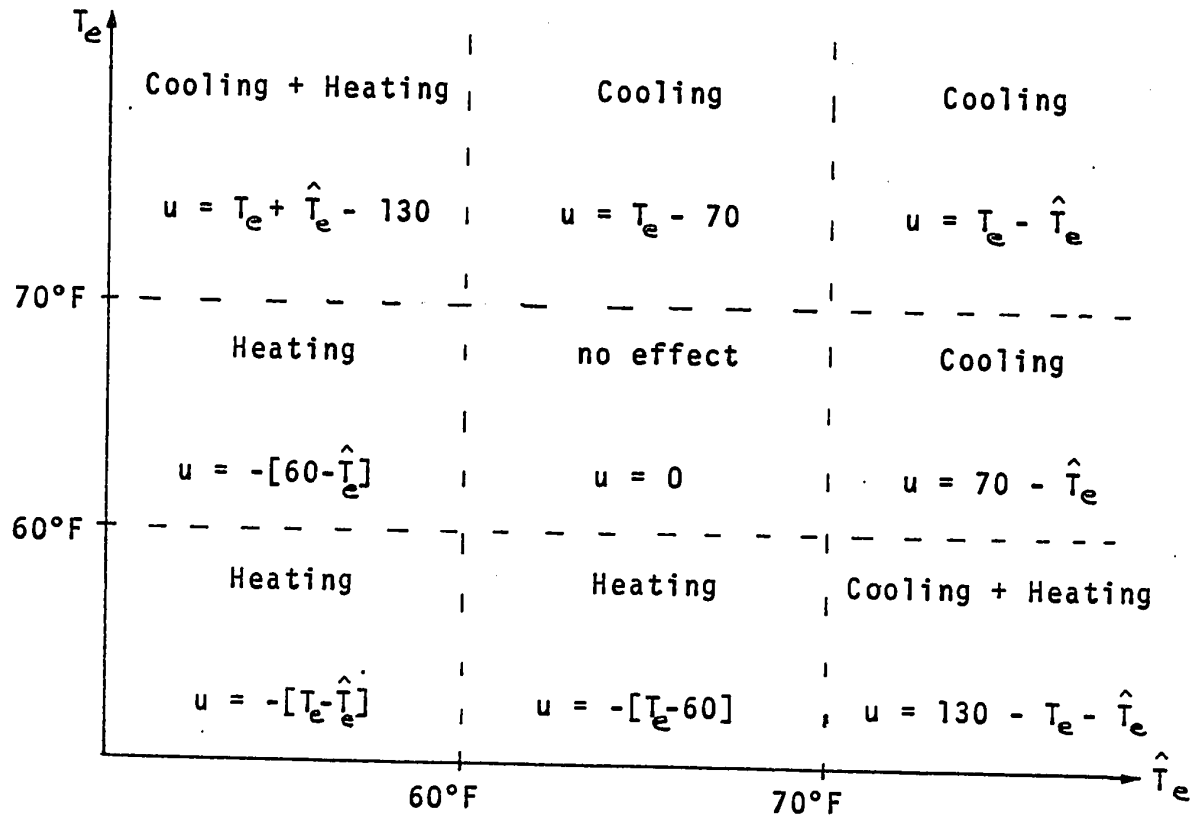


FIG. 4.15 Definition of $u(T, \hat{T})$ in terms of actual and normal temperatures.

where $e(t)$ is a white Gaussian process with mean value and variance given by,

$$E[e(t)] = 0 \quad (4.46)$$

$$E[e^2(t)] = 25 \quad (4.47)$$

The input u has been generated using weather data from the Dorval Weather Bureau and using Fig. 4.15. Results were obtained for the first three weeks of January, 1972. Values for u are listed in Table 7.

$q(t)$ is expressed in terms of KWH power.

The problem then is: given the measured output values $q(t)$ and temperature-dependent inputs $u(t)$, find the value of all the parameters describing eqns. (4.44) and (4.45).

Since the matrix gain technique described in section 3.2.2 showed promising results, for example problem (4.12), and can be used for real-time on-line applications, it was chosen for study in this section.

One of the difficulties encountered when setting up a model for (4.43) is that the values of the residual $y(t)$ are not directly available from measurements. However, this difficulty can be circumvented by using the relationship,

$$y(t) = q(t) - \underline{\Phi}^T(t) \underline{x}_p \quad (4.48)$$

Substitution into (4.37) yields

$$y(t) = \underline{a}^T Q(t) + \underline{b}^T U(t) + \underline{a}^T \Psi(t) \underline{x}_p + e(t) \quad (4.49)$$

$$\text{where } \underline{a} = [a_1, a_2, \dots, a_n]^T \quad (4.50a)$$

$$\underline{b} = [b_0, b_1, \dots, b_m]^T \quad (4.50b)$$

$$Q(t) = [q(t-1), q(t-2), \dots, q(t-n)]^T \quad (4.50c)$$

$$\underline{U}(t) = [u(t), u(t-1), \dots, u(t-n)]^T \quad (4.50d)$$

$$\Psi(t) = [\underline{\phi}(t-1), \underline{\phi}(t-2), \dots, \underline{\phi}(t-n)]^T \quad (4.50e)$$

In this manner $y(t)$ can be eliminated from eqn. (4.38).

At this point, the identification algorithm described by eqns. (3.24) to (3.26) can be applied. Rewriting eqn. (3.23), the model becomes [11]

$$q(t) = \underline{\theta}^T \underline{V}(t) + e(t) \quad (4.51)$$

For this example, observation vector $\underline{V}(t)$ is now defined as follows,

$$\underline{V}(t) = [y(t-1), y(t-2), u(t), u(t-1), \phi_1(t), \phi_2(t), \phi_3(t)]^T \quad (4.52)$$

with $y(t-1)$, $y(t-2)$ made available through eqn. (4.48).

It is important to use the most recent estimates of \underline{x}_p in eqn. (4.43). For example, when eliminating $y(t-2)$ from $\underline{V}(t)$, the

values of \underline{x}_p known at time t should be used to improve convergence of the estimation algorithm.

It can be seen that the parameter vector $\underline{\theta}$ in eqn. (4.51) will be unevenly "weighted" due to large differences in order of magnitude between the \underline{x}_p elements and those elements corresponding to the residual equations. Similarly an imbalance in weight of the matrix elements used in the identification algorithm can be expected. This could have a bearing on the accuracy of matrix calculations. However, the parameters to be estimated can be chosen to have similar orders of magnitude by making use of "a priori" knowledge of the \underline{x}_p parameters. In general, these coefficients of the periodic load components can be approximated using past information.

Approximate values for \underline{x}_p can also be obtained from a "one shot" least squares estimation of the \underline{x}_p vector only, regarding y as zero-mean residual noise.

Defining the starting values for \underline{x}_p as \underline{x}_{pIN} , a new set of output load measurement data can be generated from the following relationship,

$$qq(t) = q(t) - \underline{x}_{pIN}^T \cdot \Phi(t) \quad (4.53)$$

Now the periodic parameter to be identified becomes,

$$d\underline{x}_p = \underline{x}_p - \underline{x}_{pIN} \quad (4.54)$$

Results are listed in tabular form in Table 8 for the

TABLE 7. TEMPERATURE EFFECT INPUT U FOR JANUARY 1972.

-8.2	-9.5	-7.5	-2.8	-4.1	-6.2	-6.4	-4.4	-4.1	-2.7	-6.6
1.7	2.4	3.1	3.2	2.6	1.9	1.2	1.0	1.7	1.2	1.8
.2	.9	.8	1.6	1.4	2.4	3.5	6.3	5.2	7.7	9.7
12.1	12.8	12.6	10.7	14.2	14.4	14.8	16.4	17.2	20.8	21.4
23.7	21.5	21.4	20.1	18.8	18.8	19.0	20.7	20.4	19.0	14.1
13.4	11.1	8.8	5.8	4.4	3.5	6.9	6.2	8.2	9.8	8.4
6.8	5.6	5.4	5.2	3.8	2.7	-4.1	-4.7	-4.8	-4.4	-6.5
-5.5	-5.9	-6.2	-6.4	-7.8	-9.7	-11.2	-15.0	-14.9	-14.3	-11.7
-19.8	-20.2	-21.5	-23.0	-23.3	-22.5	-25.3	-23.5	-18.6	-20.9	-27.8
-25.8	-24.9	-24.1	-23.0	-22.4	-22.2	-21.7	-21.3	-20.6	-20.5	-20.2
-19.0	-20.4	-21.7	-22.0	-20.2	-20.5	-20.5	-20.4	-20.8	-22.1	-21.1
-21.1	-20.2	-19.6	-18.5	-18.0	-16.9	-17.4	-18.1	-18.3	-18.4	-20.1
-21.1	-24.7	-25.1	-26.6	-26.8	-28.1	-28.1	-29.5	-30.6	-29.9	-29.8
-28.6	-28.6	-28.8	-29.7	-30.2	-32.1	-32.6	-33.0	-28.4	-22.5	-22.1
-18.1	-17.6	-16.1	-11.5	-6.8	-5.0	-4.0	-2.2	-1.4	-.6	.7
1.2	2.1	.9	.1	.6	1.6	2.0	-.4	-.7	-1.9	-4.6
-17.2	-18.5	-19.1	-18.6	-17.8	-16.0	-14.0	-13.0	-16.1	-19.1	-18.7
-19.2	-16.3	-16.6	-18.4	-19.5	-24.5	-25.3	-22.4	-22.7	-24.1	-24.6
-26.3	-25.2	-26.9	-32.5	-33.8	-32.8	-31.9	-30.8	-30.8	-30.8	-24.2
-21.7	-19.8	-15.1	-12.8	-11.0	-10.9	-10.7	-9.9	-9.5	-7.7	-8.1
-4.5	-5.3	-5.8	-7.1	-8.4	-3.3	.6	1.0	1.9	1.7	4.2
4.5	4.0	4.7	5.0	5.4	4.0	5.1	6.8	7.2	6.8	6.2
4.7	3.9	3.4	2.0	1.8	6.7	6.6	4.0	7.0	9.9	2.2
14.5	14.1	14.0	12.2	13.1	13.4	15.6	14.3	14.7	14.5	17.9
										20.0

TABLE 8. Estimation of Parameters for Power Load Problem, Comparing Matrix Gain and Fletcher-Powell Techniques

No. of passes	Matrix Gain	Fletcher-Powell	a ₁	a ₂	b ₁	b ₂	XP1	XP2	XP3	VAR(e)	Computing time sec.
1	X		1.475	-0.563	2.626	1.125	1447.5	93.91	93.15	77.6	-
4	X		1.419	-0.509	2.939	1.008	1453.8	93.34	94.10	50.5	21
38	X		1.397	-0.489	3.002	1.042	1462.2	93.13	96.95	39.5	193
38	X		1.403	-0.493	3.004	0.991	1499.2*	93.14	96.92	27.5	193
129		X	1.407	-0.497	2.953	1.010	1501.4	92.62	95.70	23.0	110
True			1.400	-0.490	3.000	1.000	1500.0	100.0	100.0	25.0	
Starting			2.000	-1.000	2.000	2.000	1450.0	95.0	95.0	-	

*Additional adaptive gain introduced for XP1 corrections, KA =30,000 (eqn. 4.55).
Number of hours (data points) = 288.

computer program PMXG listed in the appendix. Parameter estimations are given for up to 38 passes through the simulated data generated by eqns. (4.43) to (4.45) with data record length of 288 samples. Estimates for the residual parameters were good even after only 4 passes through the data, but periodic parameter estimates indicated a rather slow convergence to true values. This implied that the introduction of additional adaptive gain components into the parameter updating algorithm (3.24) might be feasible. Experimental trials were run with the addition of this new gain applied to estimation of periodic parameter x_{p3} , with all other parameter estimates held at true values.

The form of additional gain, GN, was as follows,

$$GN = \frac{KA + n}{10VAR(EH) + n} \quad (4.55)$$

where n is the number of iterations, KA an arbitrary constant, $VAR(EH)$ is the estimated variance of the computed error.

Thus, the gain GN is reduced in size when large variances occur (i.e. when parameter estimates differ from true values by large amounts). As the number of iterations becomes large, the value of GN tends to unity, leading back to the original matrix gain algorithm.

As the number of iterations becomes very large ($n \rightarrow \infty$), gain GN will tend to unity. Thus the convergence properties of the algorithm (3.24) are maintained [28].

Fig. 4.16 indicates the effect of GN in speeding up

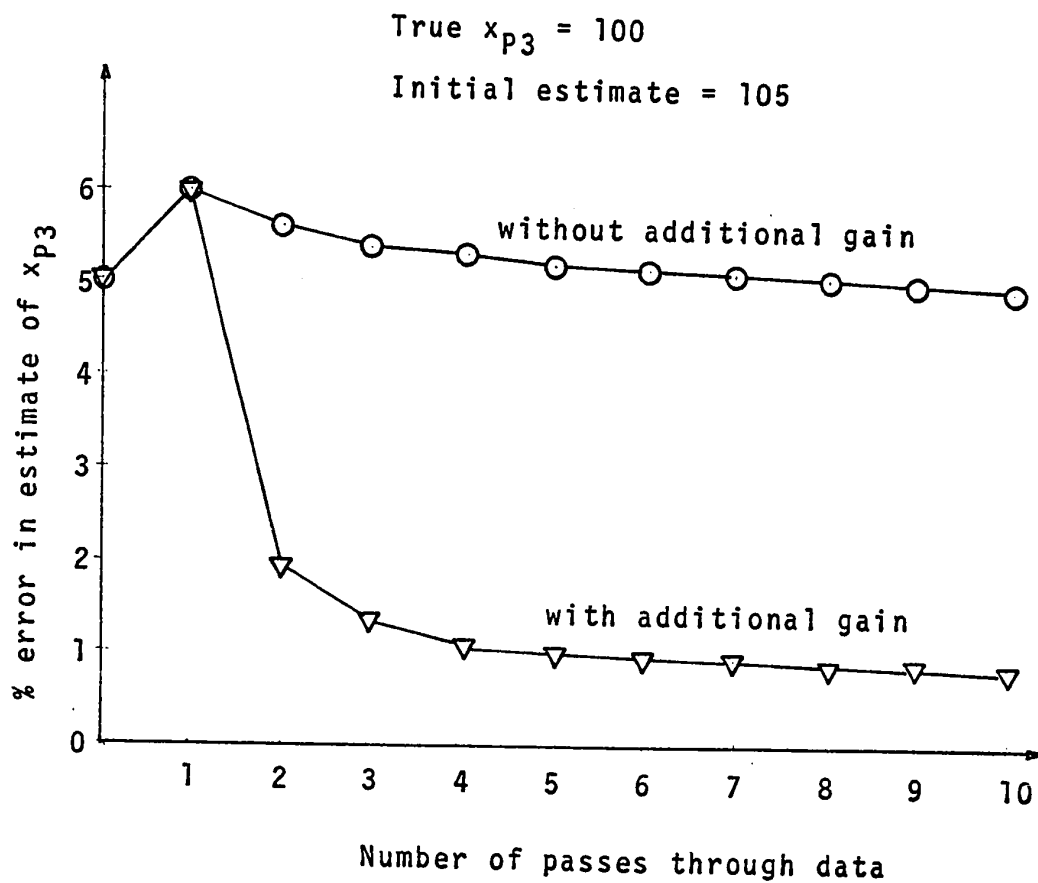


FIG. 4.16 x_{p3} estimate error, all other parameters held at true values.

the convergence of parameter estimate x_{p3} to its true value. The starting value for x_{p3} in this case was 5% higher than its true value.

Table 8 indicates the effect of introducing the gain GN into the parameter estimates for the x_p coefficients. Estimates for the residual parameters were unaffected, but x_{p1} showed a marked improvement.

Also shown in this table are comparisons with results obtained using a double precision Fletcher-Powell minimization routine (DFMP). Parameter estimates compared favourably between the two methods, with similar computing times required. However, the Fletcher-Powell program was much larger in size and did not have the real-time, on-line capabilities of the "Matrix-Gain" method.

For the problem studied in this section, it may be possible to use other identification techniques to obtain accurate estimates in a simpler manner. For instance, a method of separating the periodic and residual components of $z(t)$ is proposed as follows [33]:

Since the periodic terms have a period of 24 hours, the following relationship is valid,

$$q(t) - q(t-24) = \delta q(t) = \delta y(t) \quad (4.56)$$

from eqn. (4.37)

$$\delta y(t) = \sum_{i=1}^n a_i \delta y(t-i) + \sum_{j=0}^m b_j \delta u(t-j) + \delta e(t) \quad (4.57)$$

where,

$$\delta y(t) = y(t) - y(t-24)$$

$$\delta u(t) = u(t) - u(t-24) \quad (4.58)$$

$$\delta e(t) = e(t) - e(t-24)$$

The noise term $e(t)$ satisfies the original assumptions,

$$E[\delta e(t)] = 0 \quad (4.59)$$

However, the variance of the noise has, in effect, been doubled.

$$E[\delta e^2(t)] = E[e^2(t)] - 2E[e(t)]E[e(t-24)] + E[e^2(t-24)] \quad (4.60)$$

$$E[\delta e^2(t)] = 2E[e^2(t)]$$

$e^2(t)$ and $e^2(t-24)$ have been defined as independent.

Thus $\delta e(t)$ is an independent white noise sequence.

Since $y(t)$ can be obtained from measurement data, $q(t)$, the model described by (4.43) may easily be identified using a technique such as recursive least squares. Once the parameters a_i , b_i are known, determination of the x_{pi} coefficients is quite straightforward. Eqns. (4.49) and (4.43) can be expressed in terms of \underline{x}_p and the new observation vector formed by knowledge

of \underline{a} , \underline{b} , Q , u and ψ . After an \underline{x}_p estimate is obtained, an iteration procedure can be established by repeating the estimation of \underline{a} , \underline{b} and continuing the "two-part" estimation technique until parameter convergence is achieved.

It should be noted that the power load problem studied in this section can be solved using implementation of the techniques discussed here on a "mini" computer with suitable communication channels established with a large-scale computing system [26].

CHAPTER 5

CONCLUSIONS

5.1 Summary

Chapter 4 presents the experimental work done when using several identification methods for parameter identification of known systems. For ease in comparison of methods, results were obtained for simulated data only.

Section 4.1 discusses a relatively crude Kiefer-Wolfowitz approach to minimization using analog and digital combined capabilities. Size of the example problem was limited to a first order system due to equipment restrictions on the analog computer. Results were obtained for estimation of two parameters. Results indicated the necessity of a more sophisticated algorithm gain depending on the rate of change of parameter estimates.

Another proposed stochastic identification algorithm was tested on a second order system in section 4.2. Here, bounds were necessary on parameter estimates to ensure stability conditions. The procedures used in this section can be considered as "real-time on-line". This means that real data were sampled as it was made available by the simulated process, resulting in substantial savings in memory requirements. Only fair parameter estimation accuracy was achieved on a 16-bit word digital-analog hybrid computer. From this point on, a more powerful large general purpose computer, with 64-bit word capability, was utilized.

Results were generally successful, but good estimates of the noise coefficient were obtained only after either using a large number of data samples or recycling several times through a shorter data record. It was decided to use this example test problem to test and compare the remaining identification methods.

In section 4.3 the application of a matrix gain algorithm was discussed (equivalent to the standard Kalman filter applied to estimation of system parameters). Again this was an example of an on-line real-time application. The results obtained indicate faster convergence over the first few hundred samples, as compared to the previous scalar gain algorithms. As in the scalar gain algorithm of section 4.2, good estimates on the "c" parameters required a large number of iterations. For this method, computer calculation times become quite significant over a large number of iterations.

Aström [3] proposed the identification technique studied in section 4.4. This technique was not applied in the same on-line real-time sense as other methods presented up to this point, but a minimization procedure was applied, utilizing a set of observed input-output data obtained from a simulation of the example system. Excellent results were obtained, with no bounds being necessary on the parameter estimates. Computational times were reasonable (relatively short compared to the matrix gain algorithm discussed earlier). It was found that approximate second derivatives (for the Hessian matrix) were adequate.

In section 4.5 the identification of the example problem via the Fletcher-Powell minimization scheme is discussed where

simulated data are available. Estimation of the noise parameters, c_1 and c_2 , seemed to present some difficulty unless initial estimates were chosen to within 20 or 30 per cent of true values. This is not too serious a limitation when using the method since some "a priori" information about the parameters is usually known. Programming requirements for utilization of this method were not quite as tedious as for Aström's method, since only the cost function and gradient of the cost function were required from the programmer.

As a final example, a more difficult problem was studied where large periodic components appear in the output data used for identification. The proposed example problem was the modelling required for the load of a power system. The matrix gain algorithm of section 4.3 was chosen as a promising approach to this type of problem. This was chosen over Aström's method partly because of the complexity of supplying 2nd order partial derivatives of the system model, and partly because of the attractiveness of on-line real-time implementation. Results based on temperature records for a 3-week period compared favourably to those obtained when using the Fletcher-Powell minimization. It should be noted that the results were not obtained in a true "real-time" sense, since recycling through a fixed length input-output data record was necessary.

An effort was made to improve the estimates (specifically for the parameter estimates of the periodic components) by modifying the algorithm structure. The modification was made in such a way as to make the algorithm more sensitive to the rate

of estimate changes at some convenient point in the iteration procedure, but shifting back into the standard matrix gain algorithm as the number of iterations grew large. Some success was achieved with a few of the periodic parameters.

Other possible approaches to this example problem were discussed, including the separation of periodic and residual outputs by suitable transformation on output data.

5.2 Concluding Remarks

From the results of parameter identification methods studied here, it can be seen that many factors come into consideration when choosing identification procedures. Choice of one particular identification scheme over another depends on the a priori information available (for example information known about the true values of parameters and characteristics of the system noise), computing speed limitations, accuracy of the parameter estimates required, desirability of real-time on-line implementation, complexity of the programming required and computer memory available for use. Also to be considered are the type of system studied and the structure of the required model. Special characteristics of the model can somewhat determine the choice of method.

Consider, for example, the scalar gain algorithm proposed by Panuska [24] where employment of an enlarged parameter space avoids the problem of biased estimates caused by correlation between input noise sequence and system outputs. For the problems tested, this method showed reasonable results for

long term on-line sampling. The method is very simple to implement and the relatively small size of program makes the use of a small computer system feasible. However, for shorter record lengths, recycling through data is necessary to achieve reasonable results. Even then, the accuracy of estimates obtained using Aström's method (for the same example) is not achieved. Choice of suitable gain is required which can lead to experimentation before the method can be applied to a particular problem.

On the other hand, Aström's technique can produce excellent results for the canonical model discussed in section 4, even for relatively short record lengths. The success is not surprising since Aström [3] has shown that the maximum likelihood method produces estimates that are in general consistent, asymptotically normal and efficient for increasing sample length. But part of the success is due to the powerful search technique used for minimization, which depends on the availability of the Hessian matrix of the function being minimized. The requirement that the user supply this matrix of second partial derivatives can be a serious programming limitation when identifying systems using more complex models. Another consideration is the difficulty of applying the method on a real-time on-line basis.

The matrix gain technique described in section 4.3 was again easy to implement (as was the scalar gain technique) and indicated good performance in the on-line application to the particular example problem chosen. One promising area of study with respect to this method would be the improvement of the algo-

rithm convergence by matrix gain modification at suitable times during the iteration procedure.

The second method employing a powerful search technique (in addition to Aström's method) for function minimization was the Fletcher-Power procedure. It seems, from results obtained so far, that to use this method for parameter estimation of stochastic systems, estimation starting values suitably close to the true parameter values are required. This suggests that other more approximate techniques might be used in conjunction with the Fletcher-Powell method. In fact, a useful area of study might be the combining of several identification techniques for a particular problem (such as the power load estimation problem) switching from one method to another when certain pre-defined conditions are satisfied.

The problems studied in this thesis did not include discussion of identification of non-linear systems. However, the linear models presented here are valid for application to non-linear system identification when considering the small signal operation of a non-linear process about its normal operating point.

It seems highly desirable, in the field of parameter identification theory, that a general systematic modelling and parameter identification approach be formulated. This would be especially advantageous to control system designers. General rules and guidelines would be outlined, based on the methods studied here plus the utilization of other techniques presently available.

REFERENCES

1. Bekey, G.A. and Karplus, W.J., Hybrid Computation, Wiley and Sons, 1968.
2. Aström, K.J., "On the Achievable Accuracy in Identification Problems", in "Identification in Automatic Control Systems", IFAC Symposium, Prague, Czechoslovakia, Paper 1.8, 1967.
3. Aström, K.J. and Bohlin, T., "Numerical Identification of Linear Dynamical Systems from Normal Operating Records", in "Theory of Self-Adaptive Systems", P.H. Hammond (ed.), Plenum Press, 1966.
4. Aström, K.J., "Computer Control of a Paper Machine - an Application of Linear Stochastic Control Theory", IBM J. of Research and Development, Vol. II, no. 4, July 1967. pp. 389-405.
5. Athans, M., "The Role and Use of the Stochastic Linear-Quadratic-Gaussian Problem in Control Systems Design", IEEE Transactions in Automatic Control, Vol. AC-16, no. 6, Dec. 1971. pp. 529-552.
6. Athans, M. and Tse, E., "A Direct Derivation of the Optimal Linear Filter Using the Maximum Principle", IEEE Transactions on Automatic Control, Vol. AC-12, no. 6, Dec. 1967. pp. 690-698.
7. Bar-Shalom, Y., "Optimal Simultaneous State Estimation and Parameter Identification in Linear-Discrete-Time Systems", IEEE Transactions on Automatic Control, Vol. AC-17, no. 3, June 1972. pp. 308-319.
8. Briggs, P.A.N., Clarke, E.W. and Hammond, P.H., "Introduction to Statistical Identification Methods in Control Systems", Control, Vol. 12, no. 117, March 1968. pp. 233-238.
9. Clarke, D.W., "Generalized-Least-Squares Estimation of the Parameters of a Dynamic Model", IFAC Symposium, Prague, Czechoslovakia, Paper 3.17, June 1967.
10. Kailath, T., "An Innovations Approach to Least-Squares Estimation - Part I: Linear Filtering in Additive White Noise", IEEE Transactions on Automatic Control, Vol. AC-15, no. 2, Dec. 1968. pp. 646-652.
11. Hargreaves, D.R. and Panuska, V., "Modelling of Electric Power System Load by Stochastic Approximation", submitted to 1973 Canadian Conference on Automatic Control, University of New Brunswick, 1973.

12. Elliott, D.F. and Swoorder, D.D., "A Variable Metric Technique for Parameter Optimization", Automatica, Vol. 5, 1969. pp. 811-816.
13. Elliott, D.F. and Swoorder, D.D., "Application of a Simplified Multidimensional Stochastic Approximation Algorithm", Joint Automatic Control Conference, University of Colorado, 1969.
14. Eykhoff, P., "Process Parameter and State Estimation", IFAC Symposium, Prague, Czechoslovakia, June 1967.
15. Eykhoff, P., "Some Fundamental Aspects of Process-Parameter Estimation", IEEE Transactions on Automatic Control, Vol. AC-8, no. 5, Oct. 1963. pp. 347-357.
16. Eykhoff, P., van der Grinten, P.M.E.M., Kwakernaak, H. and Veltman, B.P.Th., "Systems Modelling and Identification", Third Congress of the International Federation of Automatic Control, June 20, 1966.
17. Gustavsson, I., "Parametric Identification of Time Series", Report 6803, Lund Institute of Technology, Division of Automatic Control, April 1968.
18. Gustavsson, I., "Parameter Identification on Multiple Input Single Output Linear Dynamic Systems", Report 6907, Lund Institute of Technology, Division of Automatic Control, July 1969.
19. Ho, Y.C. and Lee, R.C.K., "A Bayesian Approach to Problems in Stochastic Estimation and Control", IEEE Transactions on Automatic Control, Vol. AC-9, no. 5, Oct. 1964. pp. 333-339.
20. Meditch, J.S., Stochastic Optimal Linear Estimation and Control, McGraw-Hill, 1969.
21. Kalman, R.E. "A New Approach to Linear Filtering and Prediction Problems", ASME Transactions, Vol. 82, J. of Basic Engineering, March 1960. pp. 35-45.
22. Kalman, R.E. and Bucy, R.S., "New Results in Linear Filtering and Prediction Theory", ASME Transactions, Vol. 83, J. of Basic Engineering, March 1961. pp. 95-107.
23. Kiefer, J. and Wolfowitz, J., "Stochastic Estimation of the Maximum of a Regression Function", Annals of Mathematical Statistics, Vol. 23, 1952. PP. 462-466.
24. Panuska, V., "A Stochastic Approximation Method for Identification of Linear Systems Using Adaptive Filtering", Joint Automatic Control Conference, Ann Arbor, Michigan, 1968.

25. Mehra, R.K., "Identification of Stochastic Linear Dynamic Systems", Proc. of the 1969 IEEE Symposium on Adaptive Processes (8th), Decision and Control, Pennsylvania State University, Nov. 1969. pp. 6.f.1-6.f.5.
26. Girgenti, J.C, Hargreaves, D.R. and Panuska, V., "Electric Load Modelling and Prediction Using a Data-Link to a Large-Scale Computer", Submitted to 1973 Canadian Conference on Automatic Control, University of New Brunswick, 1973.
27. Panuska, V., "An Adaptive Hybrid Model for System Identification", Symposium on System Science, Hawaii, 1970.
28. Panuska, V., "An Adaptive Recursive-Least-Squares Identification Algorithm", Proc. of the 1969 IEEE Symposium on Adaptive Processes (8th), Decision and Control, Pennsylvania State University, Nov. 1969. pp. 6.e.1-6.e.5.
29. Galiana, F.D., "An Application of System Identification and State Prediction to Electric Load Modelling and Forecasting", Doctoral Thesis, Massachusetts Institute of Technology, 1971.
30. Robbins, H. and Monro, S., "A Stochastic Approximation Method", Annals of Mathematical Statistics, Vol. 22, 1951. pp. 400-407.
31. Sakrison, D.J., "A Continuous Kiefer-Wolfowitz Procedure for Random Processes", Annals of Mathematical Statistics, Vol. 35, 1964. pp. 590-599.
32. Sakrison, D.J., "The Use of Stochastic Approximation to Solve the System Identification Problem", IEEE Transactions on Automatic Control, Vol. AC-12, no. 5, Oct. 1967. pp. 563-567.
33. Panuska, V., "Electric Power System Load Modelling by a Two-Stage Stochastic Approximation Procedure", In preparation.
34. Swerling, P., "Modern State Estimation Methods from the Viewpoint of Least Squares", IEEE Transactions on Automatic Control, Vol. AC-16, no. 6, Dec. 1971. pp. 707-719.
35. Tsytkin, Ya.Z., "Adaptation, Learning and Selflearning in Control Systems", Third Congress of the International Federation of Automatic Control, London, June 1966.
36. Sacks, J., "Asymptotic Distribution of Stochastic Approximation Procedures", Annals of Mathematical Statistics, Vol. 29, 1958. pp. 373-405.
37. Wolfowitz, J., "On the Stochastic Approximation Method of Robbins and Monro", Annals of Mathematical Statistics, Vol. 23, 1952. pp. 457-461.

38. Valis, J. and Gustavsson, I., "Some Computational Results Obtained by Panuska's Method of Stochastic Approximations for Identification of Discrete Time Systems", Report 6915, Lund Institute of Technology, Division of Automatic Control, June 1969.
39. Young, P.C., "Comments on One-Line Identification of Linear Dynamic Systems with Applications to Kalman Filtering", IEEE Transactions on Automatic Control, Vol. AC-17, no. 2, April 1972. pp. 269-270.
40. Davidon, W.C., "Variable Metric Method for Minimization", Argonne National Laboratory, ANL-5990, 1959.
41. Fletcher, R. and Powell, M.J.D., "A Rapidly Convergent Descent Method for Minimization", Computer J., Vol. 6, July 1963. pp. 163-168.
42. Mehra, R.K., "On the Identification of Variance and Adaptive Kalman Filtering", IEEE Transactions on Automatic Control, Vol. AC-15, no. 2, April 1970. pp. 175-184.
43. Bekey, G.A. and Maloney, J.C., "Parameter Identification with a Hybrid Computer; Implementation of the Fletcher-Powell Method", Proc. of Third Hawaii International Conference on System Science, January 1970. pp. 175-177.
44. Saridis, G.N. and Ricker, D.W., "Analog Methods for On-Line System Identification Using Noisy Measurements", Simulation, Vol. 11, no. 5, Nov. 1968. pp. 241-248.
45. Neville, A. and Kennedy, J.B., Basic Statistical Methods for Engineers, International Textbook Co., 1966.

APPENDIX

APPENDIX

PAGE 1 C ON-LINE IDENTIFICATION---K.W.#4-1.

C D.R.H JAN.16/72

C ONE ANALOG MODEL TO CALCULATE THE GRADIENT OF INT.(E**2)
C THREE PARAMETER MODEL ONLY (A,B,C)
C

```

    DIMENSION EA(100),UA(100),EP(2),AJ(2),X(3),XP(3),XN(3),ITX(3)
    DIMENSION DAC(6)
    LOGICAL LCL
    CALL QSHYIN(IERR,680)
    DO 5 I=1,6
5    DAC(I)=0.0
    CALL QWBDAR(DAC,0,6,IERR)
    CALL QSTDA
    CALL QWCLL(0,.FALSE.,IERR)
    CALL QWCLL(1,.TRUE.,IERR)
    CALL QWCLL(2,.TRUE.,IERR)
    ACCEPT 100, E1,E2,E3,G1,G2,G3
100  FORMAT(6F10.5)
    ACCEPT 105, AI,BI,CI,XP(1),XP(2),XP(3),XN(1),XN(2),XN(3)
105  FORMAT(9F8.5)
    ACCEPT 110, ITX(1),ITX(2),ITX(3),IXI,IYI
110  FORMAT(3I1,2I3)
    ACCEPT 115, PE,PG
115  FORMAT(2F10.5)
    CALL QSPF(IERR)
10  ACCEPT 120, NS,NT
120  FORMAT(I5,I3)
    N=0
    SC=0.5
    AMN=0.0
    SN=0.0
    IX=IXI
    IY=IYI
    X(1)=AI
    X(2)=BI
    X(3)=CI
    DO 12 L=1,3
12  DAC(L)=X(L)/10.
    CALL QWBDAR(DAC,3,3,IERR)
    CALL QSTDA
15  N=N+1
    SN=SN+1.
    CALL GAMMA(SN,E1,E2,E3,PE,EPS)
    CALL GAMMA(SN,G1,G2,G3,PG,GA)
    EP(1)=EPS
    EP(2)=-EPS
    AEP=EPS/10.
    CALL QWJDAR(AEP,2,IERR)
    DO 20 I=1,NT
    CALL RANGAU(IX,IY,SC,AMN,E,U)
    EA(I)=E
    UA(I)=U

```

```

20  CONTINUE
    CALL QWCLL(2, .FALSE., IERR)
    DO 25 J=1,3
    IF(ITX(J).LT.1) GO TO 25
    K=1
    NCH=J+2
30  DA=(X(J)+EP(K))/10.
    CALL QWJDAR(DA, NCH, IERR)
    CALL QRSLL(0, LCL, IERR)
6   CALL QRSLL(0, LCL, IERR)
    IF(.NOT.LCL) GO TO 6
    CALL QWCLL(1, .FALSE., IERR)
    CALL QWCLL(0, .TRUE., IERR)
    DO 35 I=1, NT
    DAC(I)=UA(I)
    DAC(2)=EA(I)
    CALL QWBDAR(DAC, 0, 2, IERR)
    CALL QSTDA
    CALL QRSLL(0, LCL, IERR)
4   CALL QRSLL(0, LCL, IERR)
    IF(.NOT.LCL) GO TO 4
35  CONTINUE
    CALL QRBADR(AJ(K), 0, 1, IERR)
    CALL QWCLL(0, .FALSE., IERR)
    CALL QWCLL(1, .TRUE., IERR)
    IF(K.GT.1) GO TO 40
    K=K+1
    GO TO 30
40  DJ=(AJ(1)-AJ(2))/2.
    CALL QWJDAR(X(J), NCH, IERR)
    X(J)=X(J)-DJ*GA/EP
    IF(X(J).GT.XP(J)) X(J)=XP(J)
    IF(X(J).LT.XN(J)) X(J)=XN(J)
25  CONTINUE
    DO 45 J=1,3
45  DAC(J)=X(J)/10.
    CALL QWBDAR(DAC, 3, 3, IERR)
    CALL QSTDA
    CALL QWCLL(2, .TRUE., IERR)
    IF(N.LT.NS) GO TO 15
    DO 50 I=1,6
50  DAC(I)=0.0
    CALL QWBDAR(DAC, 0, 6, IERR)
    CALL QSTDA
    TYPE 200, N
200  FORMAT(//10X, 4HN = ,15)
    TYPE 201, X(1), X(2), X(3)
201  FORMAT(10X, 3HA =, F7.4, 4X, 3HB =, F7.4, 4X, 3HC =, F7.4)
    PAUSE 1
    GO TO 10
    END

```

PAGE 1 C RANGAU

```
C D,R,H. AUG.12/71
C
C GENERATES RANDOM NO. E WITH NORMAL DISTRIBUTION,
C AND PSEUDO-RANDOM SEQUENCE U (AMPLITUDE 0.5)
C
SUBROUTINE RANGAU(IX,IY,SC,AMN,E,U)
CALL GAUSSS(IX,IY,SC,AMN,E)
CALL RRANDU(IX,IY,RN)
IF(RN-0.5) 1,2,2
1 U=-0.5
GO TO 3
2 U=0.5
3 CONTINUE
RETURN
END
```

JOB CORRECT

PAGE 1

```
SUBROUTINE GAMMA(SAN,SG1,SG2,SG3,SPG,SGA)
AGAM=SG2*(SAN**SPG)
SGA=SG1/(AGAM+SG3)
RETURN
END
```

JOB CORRECT

PAGE 1 C ON LINE IDENTIFICATION OF PROCESS DYNAMICS

C NS IS NUMBER OF TESTS

C

DIMENSION XH(7),VH(7),DAC(2),REC(9)

ACCEPT 4, NS,IFIRST,INUM,IADC

READ (0,6) J1,J2,SJ1,SJ2

ACCEPT 5,R,SC,G

K=2

KH=K+1

KHH=2*K+2

C KM IS DIMENSION OF PARAMETER VECTOR

KM=3*K+1

KML=KM-1

XLIM=3.

VLIM=10.

IX=1

S=1.

AMN=0.

C INITIALIZE

N=0

VAR=0.

V=0.

DO 13 I=1,KM

XH(I)=0.

13 VH(I)=0.

C

C SLECT CONSOLE CONFIGURATION

C

CALL QSHYIN (IERR,680)

C

C SELECT OP ANALOG MODE

C

CALL QSOP (IERR)

C BEGIN LOOP

100 N=N+1

CALL GAUSSS(IX,SC,AMN,E)

CALL SLSYNC (2,LADR)

DAC(1)=V

DAC(2)=E

DAC(3)=REC(J1)/SJ1

DAC(4)=REC(J2)/SJ2

CALL QWBDAR (DAC,IFIRST,INUM,IERR)

CALL QSTDA

CALL QSDLY(1)

CALL QRBADR (RY,IADC,1,IERR)

CALL SLSYNC(3,LADR)

Y=RY/R

```

C  NOW GO TO SA ALGORITHM
    EH=Y-PRO(XH,VH,KM)
    IF(ABS(EH).GT.VLIM) EH=SIGN(VLIM,EH)
    VAR=VAR+EH*EH
    AN=N
    GN=G/AN
    DO 10 IW=1,KM
    XH(IW)=XH(IW)+GN*EH*VH(IW)
    REC(IW)=XH(IW)
    IF(ABS(XH(IW)).GT.XLIM) XH(IW)=SIGN(XLIM,XH(IW))
10  CONTINUE
C  NOW UPDATE VH

```

PAGE 2 C ON LINE IDENTIFICATION OF PROCESS DYNAMICS

```

    DO 20 MM=1,KML
    IUP=KM-MM
20  VH(IUP+1)=VH(IUP)
C  INSERT CURRENT VALUES
    VH(1)=Y
    CALL GAUSSS(IX,S,AMN,YFL)
    IF(YFL) 60,70,70
60  V=-.5
    GO TO 80
70  V=0.5
30  VH(KM)=V
    VH(KM+1)=EH
    VART=VAR/AN
    REC(KM+1)=RY
    REC(KM+2)=EH
C
    IF (N-NS) 100,200,200
C
C  SELECT IC ANALOG MODE
C
200 CALL QSIC (IERR)
    TYPE 2,VART
    TYPE 3,(XH(1),I=1,KM)
1  FORMAT(2X,7HUSTART ,F8.4//)
    TYPE 4, N
4  FORMAT(I5,3I2)
2  FORMAT(2X,11HVARIANCE = ,E9.4)
3  FORMAT(2X,7(F8.4,2X))
5  FORMAT(3F5.1)
6  FORMAT(2I2,2F5.1)
    END

```

```
FUNCTION PRO(R,S,K)
  DIMENSION R(7),S(7)
  PO=0.
  DO 30 I=1,K
30 PO=PO+R(I)*S(I)
  PRO=PO
  RETURN
END
```

```
SUBROUTINE GAUSSS(IX,S,AM,V)
  A=0.0
  DO 50 I=1,12
  CALL RRANDU(IX,IY,Y)
  IX=IY
50 A=A+Y
  V=(A-6.0)*S+AM
  RETURN
END
```

```
0 IX
1 S
2 AM
3 V
65 A
30 .50
67 I
X RRANDU
70 IY
71 Y
```

```
SUBROUTINE SLSYNC(SENS,LADR)
  LOGICAL LADR
  CALL QRSLL(SENS,LADR,IERR)
1 CALL QRSLL(SENS,LADR,IERR)
  IF(.NOT.LADR) GO TO 1
  RETURN
END
```

PROGRAM

SGN

CDC 6600 FTN V3.0-P296 OPT=:

```
PROGRAM SGN(INPUT,OUTPUT)
DIMENSION T(7),V(7),XH(7),VH(7)
READ 100, NS,KNTEST
READ 101, (T(I),I=1,7)
```

```
5 K=2
  KH=3
  KHH=5
  KHH=6
  KM=6
0 KML=KM-1
  S=1.0
  SU=1.0
  G=1.0
  AM=0.0
5 VAR=0.0
  NTEST=KNTEST
  BEH=10.0
  BX=5.0
  DO 10 I=1,7
3 XH(I)=0.0
  V(I)=0.0
10 VH(I)=0.0
  PRINT 200
  PRINT 205, NS,S
5 PRINT 210, (T(I),I=1,KM)
  CALL GAUSSS(SU,AM,U)
  V(KH)=U
  VH(KH)=U
  DO 60 N=1,NS
3 CALL GAUSSS(S,AM,E)
  Y=PRO(T,V,KM) + E
  EH = Y - PRO(XH,VH,KM)
  EH=BOUND(EH,BEH)
  VAR = VAR + EH*EH
5 GN = G/N
  DO 20 I=1,KM
20 XH(I) = XH(I) + GN*EH*VH(I)
  DO 22 I=1,KM
22 XH(I)=BOUND(XH(I),BX)
  DO 25 KK=1,KML
  IUP = KM-KK
  V(IUP+1)=V(IUP)
25 VH(IUP+1)=VH(IUP)
  V(1)=Y
  VH(1)=Y
  CALL GAUSSS(SU,AM,U)
  V(KH)=U
  VH(KHH)=EH
  VH(KH)=U
  V(KHH)=E
  IF(N.LT.NTEST) GO TO 60
  VART=VAR/N
  PRINT 212, NTEST
  PRINT 215, VART
  PRINT 220, (XH(I),I=1,KM)
```

PROGRAM

SGN

CDC 6600 FTN V3.0-P296 OPT

```
      NTEST = KNTTEST+NTEST
      60  CONTINUE
      100  FORMAT(2I5)
      101  FORMAT(7F5.2)
60      200  FORMAT(1H1, //15X, #SCALAR GAIN STOCHASTIC ALGORITHM#, ///)
      205  FORMAT(10X, #NO. OF SAMPLES =#, I5, 5X, #S =#, F5.2, ///)
      210  FORMAT(10X, #TRUE PARAMETERS#, 5X, 6F7.2, ///)
      212  FORMAT(/10X, #TESTS =#, I5)
      215  FORMAT(10X, #VAR =#E10.3)
65      220  FORMAT(10X, #XH =#, 7(E10.3, 2X), //)
      END
```

FUNCTION PRO

CDC 6600 FTN V3.0-P296 OPT=

FUNCTION PRO(R,S,K)

DIMENSION R(1),S(1)

PO=0.0

DO 30 I=1,K

30 PO=PO+R(I)*S(I)

PRO=PO

RETURN

END

PROGRAM	TFSPAN	CDC 6600 FTN V3.0-P296 OP
		PROGRAM TESPAN(INPUT,OUTPUT) DIMENSION A(5),B(5),C(5),EA(5),EB(5),EC(5),RAC(20,20) DIMENSION U(1010),E(1010),Y(1010) DIMENSION LHAC(10),VHAC(10)
5		BE=10. BC=5. NN=500 NNN=NN+10 N=2 NE=2
10		PRINT 200 PRINT 201,NN READ 100,A(1),A(2),B(1),B(2),C(1),C(2) ICOUNT=5 ALF=5.0 AM=0.0
15		S=1.0 PRINT 202, ALF PRINT 100,A(1),A(2),B(1),B(2),C(1),C(2) DO 40 I=1,20 DO 10 I=11,NNN CALL UGEN(Z)
20		U(I)=Z CALL GAUSSS(S,AM,Z) F(1)=Z 10 CONTINUE DO 15 I=1,10 U(I)=0.0 Y(I)=0.0
25	10	F(I)=0.0 DO 25 K=11,NNN SM=0.0 DO 20 I=1,N 20 SM = SM + B(I)*U(K-I) + C(I)*E(K-I) - A(I)*Y(K-I)
30	15	Y(K)=E(K) + SM PRINT 203,I,REA DO 21 I=1,NE EA(I)=0.0 EB(I)=0.0 EC(I)=0.0
35	20	DO 40 KK=1,ICOUNT E(11)=0.0 DO 35 K1=2,NN K=K1+10 45 G = ALF/((KK-1)*NN+K1) SM=0.0
40	21	DO 30 I=1,NE 30 SM = SM + EA(I)*Y(K-I) - EB(I)*U(K-I) - EC(I)*E(K-I) F(K)=Y(K)+SM F(K)=ROUND(F(K),BE) G=G+E(K) DO 21 I=1,NE J=K-I FC(I)=ROUND(EC(I)+G+E(J),BC) EB(I)=ROUND(EB(I)+G*U(J),BC)
45	30	
50		
55		

PROGRAM	TFSPAN	CDC 6600 FTN V3.0-P296 OPT
	31	EA(I)=ROUND(EA(I)-G*Y(J),HC)
	35	CONTINUE
		PRINT 204, KK
		PRINT 205, ((EA(I), I=1, NE), (EB(I), I=1, NE), (EC(I), I=1, NE))
60	40	CONTINUE
		DO 41 I=1, NE
		I1=I+NE
		I2=I+2*NE
		HAC(IREA, I)=EA(I)
65		HAC(IREA, I1)=EB(I)
	41	HAC(IREA, I2)=EC(I)
	60	CONTINUE
		ND=2*NE
70		DO 41 I=1, ND
		SUM=0.
		DO 50 J=1, 20
	50	SUM=SUM+BAC(J, I)
	61	EBAC(I)=SUM/20.
75		PRINT 206, (EBAC(I), I=1, ND)
		DO 62 I=1, ND
		DO 62 J=1, 20
	62	HAC(J, I)=BAC(J, I)-EBAC(I)
		DO 63 I=1, ND
		SUM=0.0
80		DO 64 J=1, 20
	64	SUM=SUM + BAC(J, I)*BAC(J, I)
	63	VHAC(I)=SUM/20.
		PRINT 207, (VHAC(I), I=1, ND)
85	100	FORMAT(6F5.2)
	190	FORMAT(10X, #TRUE VALUES #, 6F7.2, //)
	200	FORMAT(1H1, ///, 15X, #TEST PANUSKA STOCHASTIC METHOD#, ///)
	201	FORMAT(10X, #NUMBER OF DATA PAIRS = #, 15, //)
	202	FORMAT(10X, #ALFA = #, F6.2, //)
	203	FORMAT(/10X, #IDENTIFICATION NO. #, I4/)
90	204	FORMAT(10X, #PASS NO. #I3,)
	205	FORMAT(10X, 6E12.4)
	206	FORMAT(/10X, #EXP = #, 9E12.4)
	207	FORMAT(/10X, #VAR = #9E12.4)
		END

SUBROUTINE UGEN

CDC 6600 FTN V3.0-P296

SUBROUTINE UGEN(U)

YR=RANF(0)

U=-1.0

IF(YR.GE.0.5) U=1.0

RETURN

END

5

FUNCTION	BOUND	CDC 6600 FTN V3.0-P296
		FUNCTION BOUND(X,B)
		IF(ABS(X).LE.B) GO TO 1
		IF(X) 2,1,3
5	1	BOUND=X
		RETURN
	2	BOUND=-B
		RETURN
	3	BOUND=B
		RETURN
10		END

PROGRAM MXG

CDC 6600 FTN V3.0-P29

```
PROGRAM MXG(INPUT,OUTPUT)
DIMENSION XH(7),VH(7), XLIM(7),XMIN(7)
DIMENSION GN(7,7),COR(7),VTH(7),RR(7,7),RRR(7,7)
COMMON VV(7),V(7,500),XIN(7),T(7)
```

```
5      C
      C      JAN. 5/73.
      C
      C      INITIALIZE
      C      K IS MODEL ORDER
10     C

      KIN=100
      KNTTEST=100
      NTEST = KNTTEST
15     NS=1000
      RM=1.0
      AMN=0.0

      S=1.0
      S1=1.0
20     KN=1
      KCYC=1
      NVAR=0
      K=2

      KB=2
25     KH=K+1
      KHH = K+KB+1
      KM = 2*K+KB
      KML=KM-1
      VLIM=10.

30     DO 10 I=1,KM
      DO 10 J=1,KM
      10  GN(I,J)=0.0
      DO 15 I=1,KM
      15  GN(I,I)=1.0
      DO 25 I=1,KM
35     DO 20 J=1,KIN
      20  V(I,J)=0.0
      VV(I)=0.0
      25  VH(I)=0.0
      PRINT 200
40     PRINT 205,S

      READ 100, (T(I),I=1,KM)
      READ 100, (XIN(I),I=1,KM)
      READ 105, (XLIM(I),I=1,KM)
      READ 105, (XMIN(I),I=1,KM)
45     CALL INIT(KM,KH,KHH,KIN,S,RM)
      DO 30 J=1,KM
      VH(J) = V(J,KIN)
      30  XH(J)=XIN(J)
      DO 35 M=KHH,KM
      35  VH(M)=0.0
      PRINT 210, (T(I),I=1,KM)
      VAR = 0.0

      CALL ARRAY(2,KM,KM,7,7,GN,GN)
40     N=N+1
55     KN=KN+1
```

PROGRAM MXG

CDC 6600 FTN V3.0-P296 OPT

```

        NVAR = NVAR+1
        CALL GAUSSS(S,AM,E)
        Y=PRO(T,VV,KM) + E
        EH = Y - PRO(XH,VH,KM)
60      IF(ABS(EH).GT.VLIM) EH=SIGN(VLIM,EH)
        VAR = VAR + EH*EH
        C
        C  UPDATE GAIN MATRIX
        C
65      CALL GMPRD(GN,VH,COR,KM,KM,1)
        R=PRO(VH,COR,KM)
        R=1./(R+RM)
        CALL GMTRA(VH,VHT,KM,1)
70      CALL GMPRD(COR,VHT,RR,KM,1,KM)
        CALL GMPRD(RR,GN,RRR,KM,KM,KM)
        KMKM=KM*KM
        DO 42 JW=1,KMKM
42      GN(JW) = GN(JW)-RRR(JW)*R
        C
75      C  NOW COMPUTE CORRECTION
        C
        CALL GMPRD(GN,VH,COR,KM,KM,1)
        DO 170 IH=1,KM
80      XH(IH)=XH(IH)+COR(IH)*EH*R
        XH(2)=-XH(2)
        XH(5)=-XH(5)
        IF(ABS(XH(IH)).GT.XLIM(IH)) XH(IH)=SIGN(XLIM(IH),XH(IH))
        IF(XH(IH).LT.XMIN(IH)) XH(IH)=XMIN(IH)
85      XH(2)=-XH(2)
        XH(5)=-XH(5)
        170 CONTINUE
        C
        C  UPDATE V, VH
        C
90      DO 45 MM=1,KML
        IUP = KM - MM
        VV(IUP+1) = VV(IUP)
45      VH(IUP+1) = VH(IUP)
        C
95      C  INSERT CURRENT COMPUTED VALUES
        C
        VV(1)=Y
        VH(1)=Y
        CALL GAUSSS(S1,AM,U)
        VV(KH)=U
        VH(KH)=U
        VV(KHH)=E
        VH(KHH)=EH
        IF(N-NTEST) 40,50,50
05      50  VART=VAR/NVAR
        PRINT 215, NTEST
        NTEST = NTEST +KNTTEST
        PRINT 220, VART
        PRINT 225, (XH(I),I=1,KM)
10      IF(N-NS) 40,60,60

```

PROGRAM MXG

CDC 6600 FIN V3.0-P29

```

        60 IF(KN-KCYC) 61,62,62
        61 N=0
           NTEST = KNTST
           PRINT 227, KN
115      GO TO 40
        62 CALL ARRAY(1,KM,KM,7,7,GN,GN)
           PRINT 210, (T(I),I=1,KM)
           PRINT 230
           DO 65 I=1,KM
120      PRINT 235, (GN(I,J),J=1,KM)
        65 CONTINUE
        100 FORMAT(7F5.2)
        105 FORMAT(7F5.2)
        125 200 FORMAT(1H1, //15X, #MATRIX GAIN ALGORITHM#, //)
           205 FORMAT(10X, #S = #, F5.2, //)
           210 FORMAT(//7X, #TRUE VAL#, 6E12.3/)
           215 FORMAT(//10X, #TEST = #, I5)
           220 FORMAT(10X, #VAR = #, E10.3)
           225 FORMAT(10X, #XH = #, 6E12.3)
130      230 FORMAT(//10X, #GN#)
           235 FORMAT(10X, 6E12.3)
           227 FORMAT(//10X, #CYCLE#, I3)
           END
```

SUBROUTINE INIT

CDC 6600 FTY V3.0-P29

SUBROUTINE INIT(KM,KH, KHH,KIN,S,RM)
 DIMENSION WL(10),WM(10),X(7),VT(500,7),RR(7,500)
 DIMENSION GN(7,7)
 DIMENSION Z(500)

5 COMMON VV(7),V(7,500),XIN(7),T(7)
 AM=0.0

NV=KHH-1
 KML=KM-1

10 S1=1.0
 JL=0

15 JL=JL+1
 CALL GAUSSSS(S1,AM,U)

V(KH,JL)=U
 VV(KH)=U
 15 CALL GAUSSSS(S,AM,E)
 Y = PRO(T,VV,KM) + E

Z(JL)=Y
 20 IF(JL.GE.KIN) GO TO 25
 DO 20 MM=1,KML
 J=KM-MM

VV(J+1)= VV(J)
 20 V(J+1,JL+1) = V(J,JL)

V(1,JL+1)= Y
 25 VV(1)=Y
 VV(KHH)=E
 GO TO 15

25 CALL ARRAY(2,NV,KIN,7,500,V,V)
 CALL GMTRA(V,VT,NV,KIN)

30 CALL GMPRD(V,VT,GN,NV,KIN,NV)
 CALL MINV(GN,NV,D,WL,WM)
 CALL GMPRD(GN,V,RR,NV,NV,KIN)
 CALL GMPRD(RR,Z,X,NV,KIN,1)
 DO 30 I=1,NV

35 30 XIN(I)=X(I)
 CALL ARRAY(1,NV,NV,7,7,GN,GN)
 CALL ARRAY(1,NV,KIN,7,500,V,V)
 PRINT 200

PRINT 201,KIN
 40 PRINT 202, (T(I),I=1,KM)
 PRINT 203, (XIN(I),I=1,KM)
 PRINT 210

DO 40 I=1,NV
 40 PRINT 204, (GN(I,K), K=1,NV)
 PRINT 205

45 200 FORMAT(/15X,#INITIAL LEAST SQUARES ESTIMATE#)
 201 FORMAT(/10X,#NO. OF DATA POINTS #,I4)

202 FORMAT(/10X,#TRUE VAL #,6E12.3)
 203 FORMAT(/10X,#X INITIAL#,6E12.3)

210 FORMAT(/10X,#GN#)
 50 204 FORMAT(10X,6E12.3)
 205 FORMAT(///15X,#BEGIN STOCH. ESTIMATION#,//)
 RETURN

END



PROGRAM MAIN1

CDC 6600 FTN V3.0-P296 OPT

```
PROGRAM MAIN1(INPUT,OUTPUT)
COMMON EE,V,Y,U(8),E(10),C(100),EC(100),V1(100),VCC(200),
1 ECC(200),V2( 60, 60),DAT(2500)
READ 100, NO,NI,NP
5 READ 101, IPRINT
  READ 101, N
  READ 110, AC1,AC2
  READ 115, KA4,KA5,KA6,KA9
10 READ 115, KB4,KB5,KB6,KB9
  READ 115, KC4,KC5,KC6,KC9
  CALL SYST1(NO,NI,NP,IPRINT)
  PRINT 115, KA4,KA5,KA6,KA9
  PRINT 115, KB4,KB5,KB6,KB9
15 PRINT 115, KC4,KC5,KC6,KC9
  PRINT 120, AC1,AC2
  CALL PRO(NO,NI,NP,KA4,KA5,KA6,AC1,AC2,KA9)
  CALL PRO(NO,NI,NP,KB4,KB5,KB6,AC1,AC2,KB9)
  CALL PRO(NO,NI,NP,KC4,KC5,KC6,AC1,AC2,KC9)
20 100 FORMAT(3I5)
    101 FORMAT(I2)
    110 FORMAT(2F10.2)
    115 FORMAT(4I5)
    120 FORMAT(/10X, #AC1,AC2 #,2E11.2,/)
  END
```

LINE SYST1

CDC 6600 FIN V3.0-P296 OPT=1 737

SUBROUTINE SYST1(NO,NI,NP,IPRINT)

INTEGER GMJ

COMMON EE,V,Y,U(8),E(10),C(100),EC(100),V1(100),VCC(200),

1 ECC(200),V2(60, 60),DAT(2500)

DIMENSION COEF(30),A(5),B(5),C1(5),UIN(1000),ER(2,1000),YG(1000)

READ 100, S

100 FORMAT(F5.1)

AM=0.

NQ=(NI+2)*NO

READ 102, (COEF(I),I=1,NQ)

102 FORMAT(6F5.2)

READ 103, ALAMBD

103 FORMAT(F5.2)

PRINT 210, NO,NI,NP

210 FORMAT(1H1,10X,2(I2,2X),I4,/))

PRINT 250, S

250 FORMAT(/10X,#S = #,F5.1,/))

PRINT 211

211 FORMAT(10X,4HCOEF)

DO 10 I=1,NO

A(I)=COEF(I)

NQ=NO+I

B(I)=COEF(NQ)

NR=2*NO+I

C(NR)=COEF(NR)

10 C1(I)=COEF(NR)

PRINT 212, A(1),A(2),B(1),B(2),C1(1),C1(2)

212 FORMAT(10X,6(F6.2,3X),//)

C

C

GENERATE U INPUT.

C

DO 20 I=1,NP

CALL GAUSSS(S,AM,RR)

IF(RR.GT.0.0) UIN(I)=1.0

IF(RR.LE.0.0) UIN(I)=-1.0

ER(1,I)=UIN(I)

20 CONTINUE

DO 25 K=1,NP

NQ=2*K-1

25 DAT(NQ)=UIN(K)

DAT(1)=0.0

DAT(3)=0.0

C

C

GENERATE NOISE INPUT

C

DO 30 J=3,NP

CALL GAUSSS(S,AM,RR)

30 ER(2,J)=RR

DO 45 I=1,NO

YG(I)=0.0

ER(2,I)=0.0

45 ER(1,I)=0.0

IF(IPRINT) 40,35,40

40 PRINT 200

PRINT 201, (ER(1,J),J=1,NP)

SUBROUTINE SYST1

CDC 6600 FTN V3.0-P29

```

      PRINT 202
      PRINT 201, (ER(2,J), J=1, NP)
200  FORMAT (/10X, 5HINPUT)
201  FORMAT (20F6.2)
60   202  FORMAT (/10X, 5HNOISE)
      C
      C  OBTAIN SYSTEM OUTPUT
      C
      35  N01=N0+1
65     DO 50 J=N01, NP
          AB=0.0
          BC=0.0
          CD=0.0
70     CD=CD+ER(2,J)
          DO 55 I=1, N0
          GMJ=J-I
          AB=AB-A(I)*YG(GMJ)
          BC=BC+B(I)*ER(1, GMJ)
75     55  CD=CD+C1(I)*ER(2, GMJ)
          CD=CD*ALAMBD
          50  YG(J)=AB+BC+CD
          DO 60 I=1, NP
          NQ=2*I
80     60  DAT(NQ)=YG(I)
          IF(IPRINT) 62, 63, 62
          62  PRINT 203
          PRINT 204, (YG(I), I=1, NP)
          203  FORMAT (/10X, 6HOUTPUT)
85     204  FORMAT (10E12.4)
          63  CONTINUE
          RETURN
          END
```

JBROUTINE PRO

CDC 6600 FTN V3.0-P296 OPT=1

```

      SUBROUTINE PRO(NO,NI,NP,L1,L2,IT,ACC1,ACC2,IPRINT)
      C
      C ROUTINE FOR IDENTIFICATION OF NI INPUTS ONE OUTPUT SYSTEM
      C NO= ORDER OF SYSTEM, MAX 10
      C NI= NUMBER OF INPUTS, MAX 8
      C NP= NUMBER OF MEASUREMENT POINTS
      C L1=-1 GIVES COMMON ESTIMATION FOR STARTING VALUES
      C L1=0 GIVES COMMON ESTIMATION
      C L1=1 GIVES LEAST SQUARE ESTIMATION FOR STARTING VALUES
      C L1=2 GIVES LEAST SQUARE ESTIMATION FROM SPEC. ERROR-COEFF.
      C L2= NUMBER OF ESTIMATIONS
      C L2=0 GIVES ESTIMATIONS UNTIL MAX.COEFF.CORR.LE.0.0001
      C IT=0 GIVES APPROXIMATIVE SECOND DERIVATES
      C IT=1 GIVES EXACT SECOND DERIVATES
      C
      C C = COEFF. VECTOR
      C (OUTPUT COEFF,INPUT1 COEFF, ...,INPUTNI COEFF,ERROR COEFF)
      C CC= COEFF. CORR. VECTOR
      C V = LOSS FUNCTION
      C V1= GRADIENT OF V
      C V2= SECOND DERIVATES OF V
      C ALFA= REDUCTION FACTOR FOR COEFF.-CORR.
      C USED WHEN THE LOSS FUNCTION IS GREATER THAN
      C THE PREVIOUS LOSS FUNCTION
      C
      C REQUIRE INPUT OUTPUT DATA IN THE ARRAY DAT,
      C INPUT1(1) IN DAT(1), ...,INPUTNI(1) IN DAT(NI),OUTPUT(1) IN
      C DAT(NI+1) AND SO ON
      C
      C SUBROUTINE REQUIRED
      C VV1V2
      C GJRV
      C
      COMMON EE,V,Y,U(8),E(10),C(100),EC(100),V1(100),VCC(200),
1 ECC(200),V2( 60, 60),DAT(2500)
      DIMENSION CC(100)
      C
      MO=NO*(NI+2)
      MM=MO-NO
      IF(L1-1) 5,3,1
      1 DO 2 I=1,MM
      2 C(I)=0.0
      GO TO 5
      3 DO 4 I=1,MO
      4 C(I)=0.0
      C COEFF. ZERO
      5 CONTINUE
      C
      C START LOOP L
      IF(L2.EQ.0) GO TO 1001
      DO 1000 L=1,L2
      1001 ALFA=1.0
      IF(L1.GE.1.OR.L1.LT.0) V=1.0E15
      VO=V
      40 CALL VV1V2(NO,NI,NP,IT)

```

SUBROUTINE PRO

CDC 6600 FYN V3.0-P29

```

        IF(V.LE.V0) GO TO 42
        TK1=0.0
        DO 41 I=1,M0
        CC(I)=0.5*CC(I)
60      IF (ABS(CC(I)-TK1)) 41,41,44
        44 TK1=ABS(CC(I))
        41 C(I)=C(I)-CC(I)
        ALFA=0.5*ALFA
        IF(IPRINT-1) 300,301,301
65      301 PRINT 111,V
        PRINT 108, ALFA
        PRINT 109, (C(I), I=1,M0)
        300 IF(TK1-ACC1) 1002,1002,45
        45 CONTINUE
        GO TO 40
70      42 CONTINUE
        IF(LI-1) 9,6,6
        6 M=MM+1
        DO 8 I=M,M0
75      V1(I)=0.0
        DO 7 J=1,M0
        V2(I,J)=0.0
        7 V2(J,I)=0.0
        8 V2(I,I)=1.0
80      C DERIVATES ZERO
        9 CONTINUE
        C PRINT V,LB(EE),V1,V2
        SPR=SQRT(2.0*V/NP)
        PRINT 100,V,SPR
85      IF(IPRINT-1) 302,303,303
        303 PRINT 101, (V1(I), I=1,M0)
        302 IF(IPRINT-1) 308,308,309
        309 IF(IT.EQ.0) PRINT 102
        IF(IT.EQ.1) PRINT 103
90      DO 10 I=1,M0
        10 PRINT 104, (V2(I,J), J=1,M0)
        308 V2M=0.0
        DO 11 I=1,M0
        DO 11 J=I,M0
95      11 V2M=AMAX1(ABS(V2(I,J)),V2M)
        C
        CALL GJRV(V2,M0,1.0E-08,IERR, 60)
        IF(IERR+1) 20,19,20
100     19 PRINT 120
        RETURN
        C PRINT V2-INVERS
        20 IF(IPRINT-1) 304,304,305
        305 PRINT 105
        DO 21 I=1,M0
105     21 PRINT 106, (V2(I,J), J=1,M0)
        304 V2IM=0.0
        DO 12 I=1,M0
        DO 12 J=I,M0
        12 V2IM=AMAX1(ABS(V2(I,J)),V2IM)
110     V2COND=M0*V2M*V2IM

```

SUBROUTINE PRO

CDC 6600 FTN V3.0-P296 OPT=1

```

      IF(IPRINT-1) 306,306,307
307 PRINT 107, V2COND
C   COMPUTE COEFF.-CORR. FROM NEWTON-RAPHSON
306 TK=0.0
      DO 24 I=1,M0
      CC(I)=0.0
      DO 22 J=1,M0
22  CC(I)=CC(I)-V2(I,J)*V1(J)
      IF(ABS(CC(I))-TK) 24,24,23
23  TK=ABS(CC(I))
24  CONTINUE
      DO 25 I=1,M0
25  C(I)=C(I)+CC(I)
C
C   PRINT COEFF. AND LB(COEFF.)
C
      PRINT 109, (C(I), I=1,M0)
      DO 28 I=1,M0
28  V2(I,I)=SQRT(ABS(SPR*SPR*V2(I,I)))
      IF(IPRINT-1) 310,311,311
311 PRINT 110, (V2(I,I), I=1,M0)
310 IF(L2) 1000,30,1000
30  IF(TK.LE.ACC2.AND.IT.EQ.0) GO TO 1003
      IF(TK-ACC1) 1002,1002,1001
1000 CONTINUE
      GO TO 1003
1002 PRINT 112,ACC1
1003 CONTINUE
C
100 FORMAT(/////////5X,15HLOSS FUNCTION =,E16.8/5X,
1 29HSTANDARD DEVIATION OF ERRORS=,E16.8)
101 FORMAT(/5X,6HGRAD V/(8E15.7))
102 FORMAT(/5X,30HAPPROXIMATIVE SECOND DERIVATES)
103 FORMAT(/5X,22HEXACT SECOND DERIVATES)
104 FORMAT(8E15.7)
105 FORMAT(/5X,6HINVERS)
106 FORMAT(8E15.7)
107 FORMAT(5X,8HV2COND.=,E16.8)
108 FORMAT(/5X,46HTHE PREVIOUS STEP HAS BEEN REDUCED WITH ALFA= ,
1 E16.8)
109 FORMAT(/5X,10HNEW COEFF./(8E15.7))
110 FORMAT(5X,28HSTANDARD DEVIATION OF COEFF./(8E15.7))
111 FORMAT(/5X,15HLOSS FUNCTION =,E16.8)
112 FORMAT(/5X,28HMAX.COEFF.CORR. IS LESS THAN,E16.8)
120 FORMAT(/5X,42HA PIVOT ELEMENT HAS BEEN LESS THAN 1.0E-08)
      RETURN
      END

```

SUBROUTINE VV1V2

CDC 6600 FTN V3.0-P296 OPT=1

SUBROUTINE VV1V2(NO,NI,NP,IT)

CVV1V2 FOR SUBROUTINE PRO

C

C VV1V2 COMPUTES LOSS FUNCTION V, GRADIENT OF V AND SECOND
C DERIVATES OF V, FOR SUBROUTINE PRO

C

C NO= ORDER OF SYSTEM, MAX 10

C NI= NUMBER OF INPUTS, MAX 8

C NP= NUMBER OF MEASUREMENT POINTS

C IT=0 GIVES APPROXIMATIVE SECOND DERIVATES

C IT=1 GIVES EXACT SECOND DERIVATES

C

C C = COEFF. VECTOR

C (OUTPUT COEFF, INPUT1 COEFF, ..., INPUTNI COEFF, ERROR COEFF)

C U = INPUT DATA VECTOR

C Y = OUTPUT DATA

C

C EE= ERROR

C E = STATE VECTOR OF ERROR

C EC= FIRST DERIVATES OF ERROR

C ECC= SECOND DERIVATES OF ERROR

C ED= HELP VECTOR FOR EC

C EDD= HELP VECTOR FOR ECC

C

C V = LOSS FUNCTION

C V1= GRADIENT OF V

C V2= SECOND DERIVATES OF V

C VCC= VECTOR WITH TERMS FOR EXACT V2

C

C REQUIRE INPUT OUTPUT DATA IN THE ARRAY DAT,

C INPUT1(1) IN DAT(1), ..., INPUTNI(1) IN DAT(NI), OUTPUT(1) IN

C

C DAT(NI+1) AND SO ON

C

C SUBROUTINE REQUIRED

C

C NONE

C

C DIMENSION ED(10),EDD(10)

C COMMON EE,V,Y,U(8),E(10),C(100),EC(100),V1(100),VCC(200),

1 ECC(200),V2(60, 60),DAT(2500)

C INTEGER GM8IQ,GM8IQ2,GMMEI,GMNO,GMNO,GMNOJ,GMMEJ,GMMEI

C INTEGER GMIP,GMIP,GMIV,GMIR,GMIPJ ,GMEJ

C

C MO=NO*(NI+2)

C MM=MO-NO

C M=2*MO

C MI=NI+2

C MJ=NI+1

C KK=NP-NO+1

C

C Y=0.0

C EE=0.0

C V=0.0

C DO 1 I=1,NI

1 U(I)=0.0

C DO 2 I=1,NO

2 E(1)=0.0

C E(2)=0.0

C DO 3 I=1,M

SUBROUTINE VV1V2

CDC 6600 FTN V3.0-P296

```

      VCC(I)=0.0
      3 ECC(I)=0.0
      DO 4 I=1,MO
      EC(I)=0.0
60      V1(I)=0.0
      DO 4 J=1,MO
      4 V2(I,J)=0.0
      C
      C      START LOOP K
      DO 1000 K=1,NP
65      DO 5 J=1,MI
      IQ=NO*(J-1)
      ED(J)=0.0
      EDD(J)=0.0
      70      DO 5 I=1,NO
      IP=MM+I
      GM8IQ=IQ+I
      GM8IQ2=2*IQ+I
      ED(J)=ED(J)-C(IP)*EC(GM8IQ)
      75      IF(IT.EQ.1) EDD(J)=EDD(J)-C(IP)*ECC(GM8IQ2)
      5 CONTINUE
      IF(IT-1) 11,9,11
      9 ME=M-1
      DO 7 I=1,ME
      80      GMMEI=ME-I
      7 ECC(GMMEI+2)=ECC(GMMEI+1)
      ME=NI+1
      DO 10 J=1,ME
      GMNO=NO*(J-1)
      85      10 ECC(2*GMNO+1)=EDD(J)-EC(GMNO+1)
      ECC(2*MM+1)=EDD(NI+2)-2*EC(MM+1)
      11 CONTINUE
      ME=MO-1
      DO 6 I=1,ME
      90      GMMO=MO-I
      6 EC(GMMO+1)=EC(GMMO)
      EC(1)=ED(1)+Y
      EC(MM+1)=ED(NI+2)-EE
      DO 8 J=1,NI
      95      GMNO=NO*J
      8 EC(GMNO+1)=ED(J+1)-U(J)
      C      FIRST AND SECOND DERIVATES OF ERROR COMPUTED
      C      COMPUT STATE VECTOR E
      EE=-C(MM+1)*E(1)+E(2)+C(1)*Y
      100      ME=NO-1
      DO 20 I=2,ME
      GMMI=MM+I
      20 E(I)=-C(GMMI)*E(1)+E(I+1)+C(I)*Y
      E(NO)=-C(MO)*E(1)+C(NO)*Y
      105      E(1)=EE
      DO 21 I=1,NO
      DO 21 J=1,NI
      GMNOJ=NO*J+I
      21 E(I)=E(I)-C(GMNOJ)*U(J)
      110      C      STATE VECTOR E COMPUTED

```

SUBROUTINE VV1V2

CDC 6600 FIN V3.0-P29

```

      C      STEP TO NEXT MEASUREMENT POINT
      ME=(NI+1)*(K-1)
      DO 22 J=1,NI
      GMEJ=ME+J
115      22 U(J)=DAT(GMEJ)
      GMMENI=ME+NI+1
      Y=DAT(GMMENI)
      E(1)=E(1)+Y
      EE=E(1)
120      C      ERROR COMPUTED
      V=V+EE*EE
      IF(V.GT.1.0E15) GO TO 55
      C      LOSS FUNCTION COMPUTED
125      DO 23 J=1,MI
      IP=NO*(J-1)
      DO 23 I=1,NO
      IQ=IP+I
      23 V1(IQ)=V1(IQ)+EE*EC(IQ)
      GRAD V COMPUTED
130      C
      C
      C      START COMPUTATION OF V2
      DO 31 IJ=1,6
      DO 31 J=IJ,6
135      31 V2(IJ,J)=V2(IJ,J)+EC(IJ)*EC(J)
      C
      C      APP.V2 COMPUTED
      200 CONTINUE
      IF(IT-1) 43,41,43
140      41 DO 42 I=1,M
      42 VCC(I)=VCC(I)+EE*ECC(I)
      C      TERMS VCC FOR EX. V2 COMPUTED
      43 CONTINUE
      C      END LOOP K
145      1000 CONTINUE
      C
      V=V/2.0
      IF(IT-1) 53,51,53
150      C      ADD TERMS VCC TO APP V2.
      51 DO 52 JJ=1,MI
      IP=NO*(JJ-1)
      IQ=2*IP-1
      IR=MO-NO
155      DO 52 J=1,NO
      DO 52 I=1,NO
      GMIP=IP+I
      GMIR=IR+J
      GMIQJ=IQ+I+J
160      52 V2(GMIP,GMIR)=V2(GMIP,GMIR)+VCC(GMIQJ)
      C      EXACT V2 COMPUTED
      53 CONTINUE
      DO 54 I=1,MO
      DO 54 J=I,MO
165      54 V2(J,I)=V2(I,J)
      55 CONTINUE

```

SUBROUTINE GJRV

CDC 6600 FTN V3.0-P296 OP

SUBROUTINE GJRV(A,N,EPS,IERR,IA)

C
C INVERTS ASYMMETRIC MATRICES, HAS EMERGENCY EXIT,
C REQUIRES N**2+4*N WORDS OF ARRAY STORAGE

5 C
C A IS THE NAME OF THE MATRIX TO BE INVERTED
C N IS THE ORDER OF A
C EPS IS A VALUE TO BE USED AS A TOLERANCE FOR
C ACCEPTANCE OF THE SINGULARITY OF A GIVEN MATRIX
10 C IERR IS AN INTEGER VARIABLE WHICH WILL CONTAIN ZERO

C UPON RETURN IF INVERSION IS COMPLETED OR -1 IF SOME
C PIVOT ELEMENT HAS AN ABSOLUTE VALUE LESS THAN EPS
C IA IS THE DIMENSION PARAMETER
C MAXIMUM ORDER OF A=100
15 C THE ORIGINAL MATRIX IS DESTROYED
C IF IERR IS RETURNED ==-1 THEN THE INVERSION HAS FAILED
C OTHERWISE THE RESULTING INVERSE IS PLACED IN A

C
C
C SUBROUTINE REQUIRED
20 C NONE
C DIMENSION A(60,60),B(100),C(100),IP(100),IQ(100)
C IERR=0

25 DO 140 K=1,N
PIVOT=0.0
DO 120 I=K,N
DO 2 J=K,N
IF(ABS(A(I,J))-ABS(PIVOT))2,2,1
1 PIVOT=A(I,J)

30 IP(K)=I
IQ(K)=J
2 CONTINUE
120 CONTINUE
IF(ABS(PIVOT)-EPS)100,100,3
3 IF(IP(K)-K) 4,6,4

35 4 DO 5 J=1,N
IPX=IP(K)
Z=A(IPX,J)
A(IPX,J)=A(K,J)
5 A(K,J)=Z
40 6 IF(IQ(K)-K) 7,9,7

7 DO 8 I=1,N
IPX=IQ(K)
Z=A(I,IPX)
A(I,IPX)=A(I,K)
45 8 A(I,K)=Z
9 DO 13 J=1,N

IF(J-K) 11,10,11
10 B(J)=1.0/PIVOT
C(J)=1.0
GO TO 12
11 B(J)=-A(K,J)/PIVOT
C(J)=A(J,K)

50 12 A(K,J)=0.0
A(J,K)=0.0
55 13 CONTINUE

SUBROUTINE GJRV

CDC 6600 FTN V3.0-P29

```

        DO 14 I=1,N
        DO 14 J=1,N
    14   A(I,J)=A(I,J)+C(I)*B(J)
    140  CONTINUE
60      DO 20 KP=1,N
        K=N+1-KP
        IF(IP(K)-K) 15,17,15
    15   DO 16 I=1,N
        IPX=IP(K)
65      Z=A(I,IPX)
        A(I,IPX)=A(I,K)
    16   A(I,K)=Z
    17   IF(IQ(K)-K) 18,20,18
    18   DO 19 J=1,N
        IPX=IQ(K)
70      Z=A(IPX,J)
        A(IPX,J)=A(K,J)
    19   A(K,J)=Z
    20   CONTINUE
75      GO TO 21
    100  IERR=-1
    21   RETURN
        END
```

AM IDEN4

CDC 6600 FTN V3.0-P296 OPT=1 73/0.

```
PROGRAM IDEN4(INPUT,OUTPUT)
COMMON U(1000),T(7),Y(1000),KH,KHH,NS
COMMON F(1000)
DIMENSION H(49),X(7),G(7)
EXTERNAL FUNCT
NS=1000
NV=6
KS=2
KB=2
KH=KS+1
KHH = KH + KB
AM=0.0
S=1.0
READ 100, (T(I),I=1,NV)
READ 100, (X(I),I=1,NV)
PRINT 200
PRINT 205, (T(I),I=1,NV)
PRINT 250, (X(I),I=1,NV)
CALL SYST(NV,S,AM)
EST=0.0
EPS=10.**( -15)
LIMIT=50
CALL FMFP(FUNCT,NV,X,F,G,EST,EPS,LIMIT,IER,H)
PRINT 210,IER
PRINT 215, (X(I),I=1,NV)
PRINT 220,F
PRINT 270, LIMIT
PRINT 280, (G(I),I=1,NV)
100 FORMAT(7F5.2)
200 FORMAT(1H1,15X, #FLETCHER,POWELL MINIMIZATION FOR IDENTIFICATION#/)
205 FORMAT(/10X, #TRUE #,7F7.2)
210 FORMAT(/10X, #IER = #,I3)
215 FORMAT(/10X, #X = #,7E13.5)
220 FORMAT(/10X, #FUNCTION = #E10.3)
250 FORMAT(/10X, # XIN #,7F7.2)
270 FORMAT(/10X, # KOUNT = #I3)
280 FORMAT(/10X, #GRAD = #,7E11.3)
END
```

SUBROUTINE SYST

CDC 6600 FTN V3.0-P296 OP

SUBROUTINE SYST(NV,S,AM)
COMMON U(1000),T(7),Y(1000),KH,KHH,NS
COMMON E(1000)
DIMENSION V(7)

5		KML=NV-1 YS=0.0 ES=0.0 SU=1.0 SUMA=0.0 SUMB=0.0
10		VARC=0.0 DO 5 I=1,NS CALL GAUSSS(S,AM,ES) SUMA=SUMA+ES SUMB=SUMB+ES*ES
15	5	E(I)=ES EMEAN=SUMA/NS VARE=SUMB/NS
20	200	PRINT 200, EMEAN, VARE FORMAT(//5X, #MEAN E BEFORE CORRECTION, VARE, #, 2E12.4) DO 7 I=1,NS E(I)=E(I)-EMEAN
25	7	VARC=VARC+E(I)*E(I) VARC=VARC/NS PRINT 210, VARC 210 FORMAT(5X, #VARE AFTER CORRECTION#, E12.4, //) DO 10 I=1, NV
30	10	V(I)=0.0 DO 50 J=1, NS CALL GAUSSS(SU,AM,YU) V(KH)=YU YS=PRO(T,V,NV)+E(J) DO 20 MM=1, KML IUP=NV-MM
35	20	V(IUP+1)=V(IUP) V(1)=YS V(KHH)=E(J) U(J)=YU Y(J)=YS
40	50	CONTINUE RETURN END

SUBROUTINE FUNCT

CDC 6600 FTN V3.0-P2

SUBROUTINE FUNCT(NV,X,VAL,GRAD)
COMMON U(1000),T(7),Y(1000),KH,KHH,NS
COMMON E(1000)
DIMENSION UD(1000),EC(20),ED(10)

5 DIMENSION X(1),GRAD(1),VH(7)

EH=0
VAR=0.0
DO 5 I=1,NV
VH(I)=0.0
10 EC(I)=0.0

5 GRAD(I) = 0.0
KML = NV-1
VH(KH) = U(1)
LU=NS-1
15 DO 3 KU=1,LU
3 UD(KU)=U(KU+1)

UD(1000)=U(1)

N=0
20 N=N+1
DO 25 J=1,3
IQ=2*(J-1)
ED(J)=0.0

DO 25 I=1,2
IP=4+I
GM8IQ=IQ+I
25 ED(J)=ED(J)-X(IP)*EC(GM8IQ)
CONTINUE
DO 26 I=1,5

GMM0=6-I
30 26 EC(GMM0+1)=EC(GMM0)
EC(1)=ED(1)-Y(N)
EC(3)=ED(2)-UD(N)
EC(5)=ED(3)-EH
EH = Y(N) - PRO(X,VH,NV)

35 VAR=VAR + EH*EH
IF(N.GE.NS) GO TO 50
DO 40 MM=1,KML
IUP = NV-MM

40 40 VH(IUP+1) = VH(IUP)
VH(1)=Y(N)

VH(KH)=U(N+1)
VH(KHH)=EH
DO 30 I=1,NV
30 GRAD(I)=GRAD(I)+EH*EC(I)
GO TO 20
50 CONTINUE

DO 55 I=1,NV
55 GRAD(I) = 2.*GRAD(I)/N
VAL = VAR/N
PRINT 200, (X(I),I=1,NV)
PRINT 210, (GRAD(I),I=1,NV)
PRINT 220, VAL

200 FORMAT(/#XF = #,7E12.4)
210 FORMAT(#GRADF = #,7E12.4)
55 220 FORMAT(#VALF = #,E12.4/)

IAM PMXG CDC 6600 FTN V3.0-P296 OPT=1 73/0

```

PROGRAM PMAG(INPUT,OUTPUT)
DIMENSION AH(7),VH(7),XTH(7),PHI(3),XPI(7)
DIMENSION RR(7,7),RRR(7,7),COR(7),VHT(7)
DIMENSION XLIM(7),XMIN(7)
DIMENSION I(7),VV(7),U(301),XIN(7),GN(7,7)
DIMENSION PSI(3,2)
DIMENSION G(7),AU(12)
C INITIALIZE
C K IS MODEL ORDER
  READ 161,KP1,KP2
  READ 162, KA,KB,NN
  READ 163, KADPT
  READ 164, KNTST,KCYC
  READ 165, S
  NTEST=KNTST
  U(289)=U(1)
  RM=1.0
  A=0.0
C
C MOD31.      JAN 21/73.
C
  MOD=31
  KN=1
  NS=2
  K=NS
  MS=2
  NP=1
  NP3=2*NP+1
  NV=NS+MS+NP3
  KNP=NP3
  KH=K+1
  KHH=NS+MS+1
  KM=NV
  KML=KM-1
  VLIM=20.
  DO 105 I=1,KM
  DO 105 J=1,KM
105  GN(I,J)=0.0
  DO 106 I=1,KM
  XPI(I)=0.0
106  GN(I,I)=1.0
  READ 5,NDAY
  NWK=3
  NDAS=NWK*NDAY
  NHOURS=NDAS*24
  NCARDS=2*NDAS
  DO 13 I=1,KM
  VV(I)=0.0
13  VH(I)=0.0
  PRINT 7, NDAS,NHOURS
  PRINT 107,MOD,S,RM
C
C TO READ INITIAL ESTIMATE OF PARAMETER VALUES
C
C FIRST NS ARE A VECTOR, SECOND MS ARE B VECTOR, THIRD NP3 ARE XP

```

PROGRAM	PMXG	CDC 6600 FTN V3.0-P296
	C	READ 16, (T(I),I=1,NV) READ 16, (XIN(I),I=1,NV) READ 12, (XLIM(I),I=1,KM)
60		READ 12, (XMIN(I),I=1,KM) PRINT 140 KKN=0 DO 84 J=1,NCARDS READ 17, (AU(I),I=1,12)
65		DO 90 JJ=1,12 KKN=KKN+1 90 U(KKN)=AU(JJ) PRINT 150, (AU(I),I=1,12)
70	84	CONTINUE PRINT 18, (T(I),I=1,KM) DO 21 J=KHH,KM
75	21	XPI(J)=XIN(J) DO 177 JKX=1,KM 177 XH(JKX)=XIN(JKX) PRINT 6, (XH(I),I=1,NV) DO 22 J=KHH,KM
	22	XH(J)=XH(J)-XPI(J) DO 81 I=1,KM 81 G(I)=1.0 120 VAR=0.0 N=0 NVAR=0 NC=N+1
85	C	C C GENERATE CONTROL U C ZY1=0.0 VH(KH)=U(NC) VV(KH)=U(NC)
90		CALL PHISUB(NP,NC,PHI) I=0 DO 71 J=KHH,KM I=I+1 PSI(I,2)=0.0 PSI(I,1)=PHI(I)
95		VV(J)=PHI(I) 71 VH(J)=PHI(I) CALL ARRAY(2,KM,KM,7,7,GN,GN)
100	C	C C MAIN LOOP C
	100	N=N+1 NVAR=NVAR+1 NC=N+1 CALL GAUSSS(S,A,E) Z = PRO(T,VV,KM)+E SUM=0.0 SUM1=0.0 DO 11 I=KHH,KM SUM1=SUM1+VV(I)*XPI(I)
105		
110		

PROGRAM	PMXG	
		CDC 6600 FTN V3.0-P296 OP1
	11	SUM=SUM+T(I)*VV(I) ZZ=Z-SUM1 YR=Z-SUM
	C	NOW GO TO SA ALGORITHM
115		EH=ZZ -PRU(XH,VH,KM) IF (ABS(EH).GT.VLIM) EH=SIGN(VLIM,EH) VAR=VAR+EH*EH
	C	UPDATE GAIN MATRIX
120		CALL GMPRD(GN,VH,COR,KM,KM,1) R=PRU(VH,COR,KM) R=1./(R+RM) CALL GMTRA(VH,VHT,KM,1) CALL GMPRD(COR,VHT,RR,KM,1,KM) CALL GMPRD(RR,GN,RRR,KM,KM,KM) KMKM=KM*KM DO 42 JW=1,KMKM
125	42	GN(JW) = GN(JW)-RRR(JW)*R
	C	
130	C	NOW COMPUTE CORRECTION
	C	
		CALL GMPRD(GN,VH,COR,KM,KM,1) IF (KN.LT.KADPT) GO TO 82 NN=NN+1 DO 83 JW=KP1,KP2 83 G(JW) = (KA+NN)/((KB*VAR/NVAR)+NN) 82 CONTINUE DO 170 IH=1,KM XH(IH)=XH(IH)+COR(IH)*EH*R*G(IH) XH(2)=-XH(2) IF (ABS(XH(IH)).GT.XLIM(IH)) XH(IH)=SIGN(XLIM(IH),XH(IH)) IF (XH(IH).LT.XMIN(IH)) XH(IH)=XMIN(IH) XH(2)=-XH(2) 170 CONTINUE
135		
140		
	C	NOW ENTER KNOWN PARAMETERS
145	C	NOW UPDATE V,VH
		DO 20 MM=1,KML IUP=KM-MM VV(IUP+1)=VV(IUP) VH(IUP+1)=VH(IUP)
150	20	
	C	INSERT CURRENT COMPUTED VALUES
		I=0 SUM2=0.0 SUM3=0.0 DO 10 J=KHH,KM I=I+1 SUM2=SUM2+PSI(I,2)*XH(J) 10 SUM3=SUM3+PSI(I,1)*XH(J) ZY2=ZZ VH(1)=ZY2-SUM3 VH(2)=ZY1-SUM2 VV(1)=YR VV(KH)=U(NC) VH(KH)=U(NC) ZY1=ZY2 65 CALL PHISUB(NP,NC,PHI)

PROGRAM	PMXG
	I=0
	DO 72 J=KHH,KM
	I=I+1
170	PSI(I,2)=PSI(I,1)
	PSI(I,1)=PHI(I)
	VV(J)=PHI(1)
72	VH(J)=PHI(I)
	IF(N=NTEST) 100,300,100
300	VART=VAR/NVAR
75	PRINT 1,NTEST
	NTEST=NTEST+KNTEST
	DO 23 J=1,KM
23	XTH(J)=XH(J)+XPI(J)
	PRINT 2,VART
80	PRINT 3,(XTH(I),I=1,KM)
130	CONTINUE
	IF(N=NHOURS) 100,200,200
200	CONTINUE
85	IF(KN.GE.KCYC) GO TO 337
	KPCYC=KN+1
	PRINT 344,KPCYC
	N=0
	KN=KN+1
90	NC=1
	NTEST=KNTEST
	GO TO 100
337	CONTINUE
	PRINT 18,(T(I),I=1,KM)
95	PRINT 333
	CALL ARRAY(1,KM,KM,7,7,GN,GN)
	DO 160 K=1,KM
1	FORMAT(/2X,5HNTEST,I10)
2	FORMAT(2X,10HVARIANCE=,E10.4)
3	FORMAT(2X,2HXH,11(F8.4,2X))
0	FORMAT(1H1,2X,7E12.4)
5	FORMAT(I3)
	FORMAT(/2X,2HXI,7(F8.4,2X)/)
7	FORMAT(1H1,5X,#NUMBER OF DAYS AND DATA POINTS =#,I5,*,#,I5/)
12	FORMAT(7F5.2)
16	FORMAT(8F10.5)
17	FORMAT(12F5.1)
18	FORMAT(/,2X,2HT,7(F8.4,2X))
107	FORMAT(5X,#MOD =#,I3,5X,#S =#,F4.1,5X,#RM =#,F5.1,/)
140	FORMAT(/10X,#U INPUT#)
150	FORMAT(10X,12F5.1)
160	PRINT 330,(GN(I,K),I=1,KM)
161	FORMAT(2I2)
162	FORMAT(3I5)
163	FORMAT(I5)
164	FORMAT(2I5)
165	FORMAT(F5.2)
330	FORMAT(/5X,7E10.3)
333	FORMAT(/5X,2HGN)
344	FORMAT(/5X,5HKCYC,I2)
	END

SUBROUTINE PHISUB

CDC 6600 FTN V3.0-P296

SUBROUTINE PHISUB(NP,IT,PHI)

C
C
C

THIS DEFINES PHI MATRIX AS FUNCTION OF HOUR OF DAY

5

DIMENSION PHI(15)

NP1=NP+1

NP2=NP+2

NP3=2*NP+1

PI=3.14159265359

10

C
C
C

TO DEFINE PHI

PHI(1)=10.

DO 10 I=2,NP1

15

10

PHI(I)=SIN(2.*PI*FLOAT(IT*(I-1))/24.)

DO 20 I=NP2,NP3

20

PHI(I)=COS(2.*PI*FLOAT(IT*(I-NP1))/24.)

RETURN

END

SUBROUTINE GAUSSS

CDC 6600 FTN V3.0-P29

SUBROUTINE GAUSSS(S,AM,V)

A=0.0

DO 50 I=1,12

CALL RRANDU(Y)

5

50

A=A+Y

V=(A-6.0)*S+AM

RETURN

END

SUBROUTINE RRANDU

CDC 6600 FTN V3.0-P29

SUBROUTINE RRANDU(YFL)

YFL=RANF(0)

RETURN

END

SUBROUTINE SYST

CDC 6600 FTN V3.0-P296

SUBROUTINE SYST(T,V,KM,S,Z)

DIMENSION T(7),V(7)

AM=0.0

CALL GAUSSS(S,AM,W)

5

Z=PRO(T,V,KM)+W

RETURN

END