STRATEGIES FOR MULTIVARIABLE SELF-TUNING CONTROL

Timothy Hesketh
B.Sc. Hons (Electrical Engineering)
M.Sc. (Electrical Engineering)

A thesis presented in fulfilment of the requirements for the degree of Doctor of Philosophy in Industrial Management and Engineering at Massey University.

July, 1981

ABSTRACT

The research reported in this thesis develops strategies for applying self-tuning control theory to multivariable processes. Self-tuning control of scalar processes is now reasonably well established, and attention is being turned to the multivariable problem, with its attendant computational burden to overcome, and additional considerations such as interaction and decoupling. This thesis suggests methods for dealing with these problems, and implements the controllers on one of the readily available desk-top microcomputer systems, providing a cheap yet effective controller.

The research builds on the work of Clarke and Gawthrop [1975, 1979] on implicit controllers. The explicit schemes are derived from controllers developed by Wellstead and others [1979,(a) and (b)], using the work on multivariable controllers by Borisson [1979] and latterly Koivo [1980]. The full multivariable controller relies on standard techniques for pole placement reported by Wolovich [1974].

The proposed controllers remove interactions between loops, and other disturbances, and ensure that each loop output attains the set-point required for that loop. In particular, the explicit pole-placing controller requires a pole-placing calculation for each loop, and removes the interactions with a minimum-variance-like action, the resulting controller being modest in computational needs.

A solution is also proposed for the full multivariable pole-placing problem, which allows a comparison to be made between the full multivariable solution and the simpler controllers proposed. It is found that the full multivariable controller demands more computation, but that the resulting control may be no different from that achieved with the simpler controller.

The programs which are described are written in Fortran, in the form of subroutines which are designed for incorporation into existing control program suites. Alternatively, they may be run with a dedicated supervisory program to handle timing, input and output, display and storage of information. The subroutines are general in their application, and the information they need for any particular process may be established interactively using an offline program.

Simulation results are presented which confirm the robustness of the pole placing controller, and indicate that the proposed techniques may be used to stabilise and control a wide variety of processes; those which are non-minimum phase, certain non-linear or unstable processes. On-line control of a commercial heat exchanger process is reported, the process being similar to others which have been reported, thus providing a point of comparison of self-tuning control with other techniques. The process is multivariable. Good control is achieved, particularly in the face of perturbations to the process which result in changes to the parameters of the model describing the process behaviour, conditions under which some other controllers may not be suitable.

This research has contributed techniques which may be applied successfully to multivariable self-tuning control. Efficient programs have been written which implement the controllers on a microcomputer. Suggestions for future work include the development of a program generator which will allow more compact code to be developed, dedicated to a particular process, and which will execute more quickly. Strategies which better enable self-tuning controllers to deal with non-linear processes are also of interest.

ACKNOWLEDGEMENTS

I wish to thank Professor J. K. Scott for his support, and the facilities offered by the Department of Industrial Management and Engineering; the New Zealand Dairy Research Institute, for access to their plant during experimental phases of this project; Dr R.I. Chaplin, who made possible the use of ISIS and provided supervision during the latter stages of this research; and most of all, my wife, Beryl.

TABLE OF CONTENTS

NOTATION

Boldface letters are used to represent polynomials, polynomial matrices or matrices. A polynomial is an expression in the backward shift operator q^{-1}. Thus:

$$\mathbf{A}(q^{-1}) = a_0 + a_1 q^{-1} + a_2 q^{-2} + \ldots + a_{nA} q^{-nA}$$

n_A is defined to be the order of the polynomial **A**. Polynomials operate on process variables so that:

$$\mathbf{A}(q^{-1})y(t) = a_0 y(t) + a_1 q^{-1} y(t) + \ldots + a_{nA} q^{-nA} y(t)$$

$$= a_0 y(t) + a_1 y(t-1) + \ldots + a_{nA} y(t-n_A)$$

y is only defined at discrete time instants, y(t) being the (sampled) value of y at the present time. y(t-1), y(t-2) .. represent values of y at previous sampling instants. The sampling interval is not explicitly stated. The dependence of a polynomial on q^{-1} will be omitted for brevity unless ambiguity results.

A polynomial matrix may be represented equivalently by:

$$\mathbf{B} = \mathbf{B}_0 + \mathbf{B}_1 q^{-1} + \ldots + \mathbf{B}_{nB} q^{-nB}$$

where the B_i are m x p matrices, or by:

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \ldots & \mathbf{B}_{1p} \\ \ldots & \ldots & \ldots & \ldots \\ \mathbf{B}_{m1} & \ldots & \ldots & \mathbf{B}_{mp} \end{bmatrix}$$

where the $\mathbf{B}_{ij}$ are scalar polynomials of order n_B.

The scalar quantity A(1) is the value of polynomial **A** evaluated with $q^{-1}=1$.

Processes may be represented in Auto Regressive Moving Average (ARMA) form by:

$$\mathbf{A}y(t) = q^{-k}\mathbf{B}u(t) + q^{-k}\mathbf{D}v(t) + \mathbf{C}e(t) + d$$

A, **B**, **C** and **D** are polynomials (or polynomial matrices) representing a scalar (or a multivariable) process. The process time delay k is expressed as a whole number of sampling intervals. y(t) is the process output, u(t) the controlled input, v(t) a measured disturbance and e(t) a white noise sequence. d is an offset.

y(t|t-k) is the predicted value of y at time t, given information up to and including time t-k.

Principal symbols used in this thesis

k	time delay
i,j,l,m,n	indices
$\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}$	Process polynomials/polynomial matrices
$\mathbf{A1}, \mathbf{A2}, \mathbf{B1}, \mathbf{C1}$	
$\mathbf{A'}, \mathbf{B'}, \mathbf{D'}, \mathbf{A^{-}}, \mathbf{B^{-}}$	
$y(t)$	Process output
$w(t)$	Process set-point
$u(t)$, $u'(t)$	Controlled input
$v(t)$	Measured disturbance
$e(t)$, $e'(t)$, $e''(t)$	White noise sequences
d, d'	Offset values
$\mathbf{H}, \mathbf{G}, \mathbf{E}$	Controller polynomials $Hu(t) + Gy(T) + Ew(t) = 0$
$\mathbf{F}_c$	Multivariable pole placing polynomial matrix
$\mathbf{S1}, \mathbf{S2}$	Polynomials for servo control
$\mathbf{P}, \mathbf{P'}, \mathbf{Q}, \mathbf{Q'}, \mathbf{R}, \mathbf{S}$	General polynomials
$\mathbf{L}, \mathbf{M}, \mathbf{B1}, \mathbf{B2}$	
$\phi(t)$, $\phi_y(t)$, K	Scalar quantities
$\mathbf{X}, \mathbf{\Theta}, \mathbf{x}(t)$	General vectors
$\mathbf{A}_m, \mathbf{T}, \mathbf{T^{-}}$	Polynomials specifying poles for pole placement
$\mathbf{A}_0$	Observer polynomial

Chapter 1

LITERATURE SURVEY

1.0 INTRODUCTION

The self-tuning principle and the present form of the self-tuning regulator and controller may be attributed to Astrom and Wittenmark [1973]. The idea of a self-tuning control system was however proposed by Kalman [1958] and Peterka [1970].

1.1 SELF-TUNING REGULATOR BASED ON MINIMUM VARIANCE CONTROL

The self-tuning regulator was developed for systems with constant but unknown parameters, and combined a recursive parameter estimation scheme with a suitable control design procedure. Astrom and Wittenmark [1973] used a least squares estimator and a minimum variance control law. If the parameter estimates converged, and if weak assumptions were made about the system tc be controlled, it was shown that the control calculated recursively from the estimated biassed parameters would converge to the control law which could be calculated from a full knowledge of the system parameters. The feedback regulator was thus disturbance rejecting. The development of the self-tuning regulator [Astrom and Wittenmark, 1973], relied on an earlier exposition of stochastic control theory [Astrom, 1970]. This theory has been used to design regulators for the steady state control of industrial processes and establishes the notation and fundamental model structure which is used for developing the self-tuning procedures. It is instructive to examine Astrom's development of the self-tuning regulator.

Astrom and Wittenmark's [1973] development has established a pattern of analysis which is adhered to throughout the

literature. Essentially, results are derived first for systems with known parameters. These results are combined with an estimation scheme to give a self-tuning device, whose properties in the presence of noise have then to be examined. This basic idea has been called certainty equivalence, i.e. the use of a control law for known parameters, which are then replaced by their estimates [Egardt, 1978].

1.1.1 The CARMA model.

The notation has been described earlier. In particular the dependence of the polynomials on q^{-1} will be omitted for the sake of clarity except where this becomes ambiguous. The description given here relies on [Astrom, 1970], but also draws on [Astrom, 1977].

The single-input-single-output case will be considered. The controlled auto-regressive moving average model is so called because without the control signal the model is identical to the ARMA process which is commonly used in time series analysis [Box and Jenkins, 1970]. The model is linear, and using the superposition principle the process is characterised by two terms, the first describing the process dynamics in response to a control input u(t), while the second gives the response to e(t), a sequence of independent normal random variables.

$$y(t) = [\mathbf{B1/A1}]u(t-k) + [\mathbf{C1/A2}]e(t) \qquad ....(1.1)$$

The model can be simplified to that used by Astrom and Wittenmark [1973] by introducing $\mathbf{A} = \mathbf{A1A2}$, $\mathbf{B} = \mathbf{B1A2}$, and $\mathbf{C} = \mathbf{C1A1}$:

$$\mathbf{A}y(t) = \mathbf{B}u(t-k) + \mathbf{C}e(t) \qquad ....(1.2)$$

It is always possible to assume that the polynomial $\mathbf{C}$ has all its

zeros inside the unit circle [Astrom, 1977]. If a minimum variance control strategy is to be used, it is also necessary to assume that the system is minimum phase (i.e. that the zeros of **B** lie inside the unit circle). Further assumptions are that a bound can be given to the order of the system and that the time delay is known. The time delay is characterised by k which is the integral part of T/h where T is the true time delay and h is the sampling interval.

1.1.2 Optimal prediction and minimum variance control

Minimum variance control minimises the variance of the process output in the presence of process noise, achieving this by choosing the control variable so that the output of the k-step predictor coincides with the desired output value. The result is given in the separation theorem [Astrom, 1970]. The k-step optimal predictor minimises the variance of the prediction error in steady state. The analysis given below follows that of Astrom [1970] and puts:

$$[\mathbf{C}/\mathbf{A}] = \mathbf{F} + q^{-k}[\mathbf{G}/\mathbf{A}] \qquad \ldots.(1.3)$$

where the polynomial **G** is of order n -1 and **F** is of order k-1. Using (1.2) and (1.3):

$$\begin{aligned} y(t) &= \mathbf{F}e(t) + [\mathbf{BF}/\mathbf{C}]u(t-k) + [\mathbf{G}/\mathbf{C}]y(t-k) \\ &= \mathbf{F}e(t) + y(t|t-k) \end{aligned} \qquad \ldots.(1.4)$$

It is now possible to define a control law governing the behaviour of u(t), such that the variance of the output is as small as possible. This is given by:

$$\mathbf{BF}u(t) + \mathbf{G}y(t) = 0 \qquad \ldots.(1.5)$$

1.1.3 Self-tuning

The self-tuning regulator was designed based on a certainty-equivalence argument. The model of the process and its environment is estimated recursively and the control determined as if the estimated model was the true model. Consider the special case where $\mathbf{C}=1$. For this case, the parameters of the model (1.2) may be explicitly determined by a least squares algorithm, and a feedback control law applied. Astrom [1973] showed that it is in general not possible to estimate all the parameters of the model when the input is generated by a feedback control law and one of the parameters is fixed to overcome this possible difficulty.

It is now necessary to remove the assumption that $\mathbf{C}=1$, introducing a disturbance-rejecting controller. Use of the model equation (1.4) results in the system description:

$$y(t) = \mathbf{A}^{\bullet}y(t-k) + \mathbf{B}^{\bullet}u(t-k) + e'(t) \qquad ....(1.6)$$

where $\mathbf{A}^{\bullet}$ is of order n_A-1 and $\mathbf{B}^{\bullet}$ is of order n_B+k-1. The disturbance $e'(t)$ is a moving average of order k of the driving noise $e(t)$. The minimum variance strategy is then simply:

$$\mathbf{A}^{\bullet}y(t) + \mathbf{B}^{\bullet}u(t) = 0 \qquad ....(1.7)$$

The notation of Astrom and Wittenmark [1973] has been changed slightly to allow the use of the backward shift operator. A least squares method may still be used to identify the model, although k more parameters have to be estimated. If these parameter estimates converge then the corresponding regulator will converge to a minimum variance regulator. The theorems establishing this self-tuning property are given in [Astrom and Wittenmark, 1973].

1.1.4 Multivariable self-tuning regulators

The minimum variance self-tuning regulator described above has been extended to deal with a class of multivariable systems [Borisson, 1979]. The process is required to have as many inputs as outputs, to have a minimum phase property, and to have an impulse response starting with a non-singular matrix. The development is exactly the same as in the scalar case, with the change that polynomial matrices replace all the polynomials used so far, and the variables become vector quantities. Various matrix relationships have to be defined to overcome possible difficulties in the manipulation of the equations. The most important of these concern the k-step predictor. The process is given by:

$$\mathbf{A}y(t) = \mathbf{B}u(t-k) + \mathbf{C}e(t) \qquad \ldots.(1.8)$$

Introducing:

$$\mathbf{C} = \mathbf{AF} + q^{-k}\mathbf{G} \qquad \ldots.(1.9)$$

and also the matrices $\mathbf{F}$ and $\mathbf{G}$ given by:

$$\mathbf{F'G} = \mathbf{G'F} \qquad \ldots.(1.10)$$

then $\mathbf{C'}$ may be defined using an analogue of equation (1.9) i.e. $\mathbf{C'} = \mathbf{AF'} + q^{-k}\mathbf{G'}$ which with equation (1.10) gives $\mathbf{F'C} = \mathbf{C'F}$, and these, with equations (1.8)-(1.10) give, with manipulation:

$$y(t) = \mathbf{C'}^{-1}(\mathbf{F'B}u(t-k) + \mathbf{G'}y(t-k)) + \mathbf{F}e(t) \qquad \ldots.(1.11)$$

Equation (1.11) serves to define $y(t|t-k)$, and shows the noise to be a moving average of order k, uncorrelated with u and y at time t-k or earlier. In self-tuning, the algorithm estimates the parameters of the model:

$$y(t) = \mathbf{A}^{*}y(t-k) + \mathbf{B}^{*}u(t-k) + e'(t) \qquad ....(1.12)$$

and the control is computed from:

$$\mathbf{A}^{*}y(t) + \mathbf{B}^{*}u(t) = 0 \qquad ....(1.13)$$

Borisson [1979] points out that if $\mathbf{B}_0$ is non-singular, transforming the system equation (1.8) so that $u'(t)=\mathbf{B}_0u(t)$ will mean that each loop will have one dominant control signal which will influence it before other control signals from other loops. Each loop may be considered as a scalar case with feed-forward disturbances from other loops.

1.2 IMPLICIT OPTIMAL SELF-TUNING CONTROLLERS

This class of self-tuning controller was described by Clarke and Gawthrop [1975]. The possibility of optimally following set-point changes was introduced, and in particular a cost was placed on excessive control action, which is a problem associated with minimum variance controllers at certain sampling rates. A control costing, in addition to the cost of variation in the output variable, introduced concepts of optimal control. This extended the limited minimum control objective of Astrom and Wittenmark [1973], and at the same time dealt with problems of non-minimum phase systems, which were excluded from consideration under the earlier method. The necessary a-priori choice of one system parameter was also seen as detracting slightly from the self-tuning philosophy. Once again their controller dealt only with single-input-single-output systems. A later paper [Clarke and Gawthrop, 1979] summarised and expanded earlier work.

1.2.1 Prediction and control with known parameters

The process is described by the equation:

$$\mathbf{A}y(t) = \mathbf{B}u(t-k) + d' + \mathbf{C}e(t) \qquad \ldots.(1.14)$$

This is the same as equation (1.2) with the addition of the term d', which gives the output of the system in the absence of any inputs or noise. It is thus a true offset. Its introduction is of practical significance in the case where transducer offsets have to be dealt with. The cost function to be minimised is a quadratic one-step criterion:

$$I = E\{(\mathbf{P}y(t)-\mathbf{R}w(t-k))^2 + (\mathbf{Q}'u(t-k))^2\} \qquad \ldots.(1.15)$$

The control signal has to be calculated from a consideration of this cost function, which of course is impossible to calculate at the time. The control signal has to be the best that can be applied in an uncertain situation with the process subject to disturbances. As with the self-tuning regulator, the incorporation of a k-step predictor gives a disturbance rejecting controller, although in this case it is the cost function params that must be predicted. Uncertainty is due to uncertainty about the process output. Equations (1.3) and (1.4) describe the behaviour of the k-step predictor. The prediction error $\mathbf{F}e(t)$ is a moving average of order k, which is uncorrelated with y(t-k) and u(t-k) at or earlier than time t-k. So it is possible to write:

$$I = [\mathbf{P}y(t|t-k)-\mathbf{R}w(t-k)]^2 + [\mathbf{Q}'u(t-k)]^2 + E\{[\mathbf{P}e(t)]^2\} \qquad \ldots.(1.16)$$

The problem has thus been reduced to a deterministic optimisation problem which may be solved [Clarke and Gawthrop, 1975], by differentiation.

Following the above analysis, it becomes expedient to introduce an auxiliary function described by:

$$\phi(t) = \mathbf{P}y(t) + \mathbf{Q}u(t-k) - \mathbf{R}w(t-k) = \phi(t|t-k) + \underline{\xi}(t) \qquad \ldots.(1.17)$$

where $\mathbf{Q}=q_0'\mathbf{Q}^*/b_0$. The prediction of this function depends on the prediction of y:

$$\phi(t|t-k) = \mathbf{P}y(t|t-k) + \mathbf{Q}u(t-k) - \mathbf{R}w(t-k) \qquad \ldots.(1.18)$$

and it can be shown that the criterion is minimised if:

$$\phi(t|t-k) = 0 \qquad \ldots.(1.19)$$

It is possible to complete this analysis by substituting the predictor of (1.4) into (1.18), but it will prove more instructive to follow Allidina and Hughes [1979], and return to the process description (1.14):

$$\mathbf{P}y(t) = \{\mathbf{PB/A}\}q^{-k}u(t) + \{\mathbf{PC/A}\}e(t) + \{\mathbf{P/A}\}d' \qquad \ldots.(1.20)$$

The effect of the disturbance sequence $\{e\}$ has to be dealt with so this is separated into two sequences, one relating to future disturbances about which nothing can be done, and one relating to present and past disturbances.

$$\{\mathbf{PC/A}\} = \mathbf{F} + q^{-k}\{\mathbf{G/A}\} \qquad \ldots.(1.21)$$

Use of equation (1.21) will give a detuned form of the optimal predictor. Equations (1.17), (1.14), (1.19) and (1.21) are combined to give:

$$\phi(t) = \mathbf{F}e(t) + \{\mathbf{F/C}\}d' + \{\mathbf{BF/C}\}u(t-k) + \{\mathbf{G/C}\}y(t-k) + \mathbf{Q}u(t-k) - \mathbf{R}w(t-k) \qquad \ldots.(1.22)$$

Equation (1.19) gives the control signal which minimises the cost function. Writing d=F(1)d' :

$$u(t) = \{\mathbf{R}\mathbf{C}w(t) - \mathbf{G}y(t) - d\}/\{\mathbf{B}\mathbf{F} + \mathbf{C}\mathbf{Q}\} \qquad ....(1.23)$$

or:

$$u(t) = \{\mathbf{R}w(t) - \phi_y(t+k|t) - d\}/\mathbf{Q}$$

$$\phi_y(t+k|k) = \{\mathbf{B}\mathbf{F}/\mathbf{C}\}u(t-k) + \{\mathbf{G}/\mathbf{C}\}y(t-k) \qquad ....(1.24)$$

The representation of (1.24) was used by Clarke and Gawthrop [1979], who explored various possibilities arising from it, which highlight the effect that **Q** has in reducing the excursions of the control signal. They pointed out that if the form of (1.24) is used, **Q** may be varied on-line without changing any of the other controller parameters.

1.2.2 Self tuning

Equation (1.22) is an expression for the prediction of the cost function which incorporates the unknown process parameters and may be re-expressed as:

$$\mathbf{C}\phi(t+k|t) = \mathbf{G}y(t) + \mathbf{H}u(t) + \mathbf{E}w(t) + \text{offset} \qquad ....(1.25)$$

This gives an expression which must be solved at each step for the control signal u(t) (equation 1.16). The extension to the self-tuning case uses the certainty-equivalence principle. The unknown parameters are now **G**, **H**, and **E** which must now be recursively estimated. If **C**=1, (1.18) and (1.21) give:

$$\phi(t+k) = \mathbf{x}^T(t)\theta + e'(t+k) \qquad ....(1.26)$$

Where θ is a vector containing the unknown parameters, and $\mathbf{X}$ is a vector containing the known functions of time. $\phi(t+k)$ is known from equation (1.17). In the more general case when $\mathbf{C}$ is a polynomial:

$$\phi(t+k) = \mathbf{X}^T(t)\theta + e'(t+k) + (1-\mathbf{C})\phi(t+k|t) \qquad ....(1.27)$$

The control law sets $\phi(t+k|t)=0$ at each step, and assuming that the parameters converge, they will converge to the true values required to give the optimal control law.

Equation (1.25) makes obvious the requirement that $\mathbf{C}$ have all zeros inside the unit disc, as early incorrect values of $\phi(t+k|k)$ should decay if control is to converge.

Clarke and Gawthrop [1979] present an alternative procedure for self-tuning which has various useful merits. The $\phi_y(t+k|t)$ of equation (1.24) is estimated. The following equations establish the parameter estimation scheme:

$$\phi_y(t) = \mathbf{P}y(t)$$

$$\phi_y(t|t-k) = \mathbf{G}y(t-k) + \mathbf{BF}u(t-k) - \Sigma c_i \phi_y(t-i|t-k-i)$$

$$= \mathbf{X}^T(t-k)\theta \qquad ....(1.28)$$

While the ϕ_y for inclusion in $\mathbf{X}$ is not known, a value for it may be calculated at each step from the updated estimates and the previous $\mathbf{X}$ vector, and included as though the prediction error were white. Such an approach is common to many least squares procedures, (see for example Panuska [1980]). The control signal is calculated using equation (1.24).

It has been noted that equation (1.28) is a self-tuning predictor of the sequence $\emptyset_y$, and as such is divorced from the control law used to generate u(t), which may be linear or non-linear. This has a number of practical implications. Because there is no expectation placed on u(t) to ensure convergence, constraints may be placed on control signal amplitude, or least squares estimation may be started under manual control or using some other controller, and the self-tuning loop only closed when parameters have converged and it is apparent that satisfactory control may be achieved.

Gawthrop [1977] extends the cost function to include rational transfer function terms, where before polynomial terms were required, which enables in the first instance, prefiltering of the set point. More importantly however, it offers a method for providing model reference adaptive control, inviting comparisons with schemes proposed for example by Landau [1979]. A point which emerges here is the necessity to cancel system zeros with controller poles in MRAC schemes, precluding once again control of systems with zeros in the unstable region, or giving rise to unacceptable intersample behaviour.

A comparison with the Smith Predictor [Smith, 1959] is also presented by Gawthrop [1977] for the special case where $\mathbf{P}=\mathbf{R}=1$. The comparison is invited by the least squares prediction of the output performed by the k-step predictor. The use of a self-tuning controller to overcome the dead time removes some of the objections associated with the Smith Predictor, particularly its requirement for a parallel system model, which precludes its application to open loop unstable processes, and its sensitivity to system parameter drift. This last advantage of self-tuning systems is of course one of the major motivations for their use generally.

1.2.3 Closed loop behaviour

The certainty-equivalence approach makes it possible to examine the behaviour of the entire closed loop system when the controller is included in the loop, assuming that the parameters being estimated have converged to their final values. This technique may be used to analyse the response of the controlled system to disturbances and set point changes, and also for classical analysis of the closed loop transfer function for sensitivity results and stability margins. The closed loop transfer functions are derived by combining equations (1.23) and (1.14), ignoring the steady offset value:

$$\{\mathbf{BP} + \mathbf{AQ}\}y(t) = \mathbf{BR}w(t-k) + \{\mathbf{BF} + \mathbf{QC}\}e(t)$$
$$\{\mathbf{BP} + \mathbf{AQ}\}u(t) = \mathbf{A}w(t) - \mathbf{F}e(t) \qquad \ldots.(1.29)$$

Most systems to be controlled are continuous, and analysis of the underlying continuous system behaviour is important.

The closed loop transfer function can be used to ensure the correct steady state behaviour of the controlled system [Allidina and Hughes, 1979], [Hesketh, 1980]. If the steady state relationship between the set point and the process output is considered, and assuming the noise disturbance has zero mean (no offset) equation (1.29) can be re-written in terms of estimated parameters using the relationships $\mathbf{B}=(\mathbf{H}-\mathbf{QC})/\mathbf{F}$, $\mathbf{A}=(\mathbf{PC}-q^{-k}\mathbf{G})/\mathbf{F}$ to give:

$$(\mathbf{PH}-q^{-k}\mathbf{QG})y(t) = \mathbf{R}(\mathbf{H}-\mathbf{QC})w(t-k) + e'(t) \qquad \ldots.(1.30)$$

Using a standard result, the steady state relationship is obtained by evaluating (1.30) with q=1. Correct steady state set point attainment can be ensured by appropriate choice of R(1), and hence **R**. **R** can be adjusted on-line if care is taken to also adjust the associated estimated polynomial $\mathbf{E}=-\mathbf{RC}$ to maintain convergence of the estimated parameters.

1.2.4 Multivariable self-tuning controller

The methods of Borisson [1979] have been applied to the self-tuning controller to extend it to the multivariable case [Hesketh, 1980], [Koivo, 1980]. As in section 1.1.4, this extension can be made dependent on the re-stating of equations with polynomial matrices replacing the polynomials, and with vectors replacing scalars [Koivo, 1980]. In this way, the analysis follows almost exactly the original analysis of Clarke and Gawthrop (1975), and hence that of sections 1.2.1 and 1.2.2. The most notable requirement is the k-step predictor of section 1.1.4 (equations (1.9)-(1.11), [Borisson, 1979]). In the equivalent of (1.21) there is a requirement that **P** and **C** commute, and so **P** is limited to be a polynomial, not a matrix polynomial. In the final self-tuning form, the parameters to be identified are those of the equation:

$$\phi(t+k|t) = \mathbf{G}y(t) + \mathbf{H}u(t) + \mathbf{E}w(t) \qquad \ldots.(1.31)$$

This is the same as (1.25), the scalar version. As usual, these parameters are identified as though the noise were white, and the control signals computed from:

$$H_0 u(t) = -\{ \Sigma G_i y(t-i) + \Sigma H_i u(t-i) + \Sigma E_i w(t-i) + d' \} \qquad \ldots.(1.32)$$

In order to ensure set-point attainment, it is necessary to introduce integral behaviour into the controller structure [Koivo, 1980], which may be done by setting $\mathbf{Q} = (1-q^{-1})I$, or by using a difference structure. This latter technique has also been used for controllers based on more conventional estimation [Sandoz and Wong, 1978], where y(t) is replaced by y(t)-w(t), and

u(t) by u(t)-u(t-1). The multivariable extensions reported by Hesketh [1980] form part of the research reported in this thesis and will be discussed in Chapter 2.

1.3 IMPLICIT POLE ASSIGNMENT SELF-TUNING CONTROLLERS

Controllers of this kind have been dealt with by Allidina and Hughes [1980] and by Astrom and Wittenmark [1979]. Allidina and Hughes based their scheme on the implicit controller of Clarke and Gawthrop [1979]. Pole placement algorithms are dependent on the closed loop behaviour of the system, and their analysis depends heavily on the certainty equivalence principle once again.

1.3.1 Pole assignment based on the optimal self-tuning controller

Equation (1.30) gives the closed loop transfer function for the implicit controller. If $\mathbf{A}_m$ is a polynomial with the desired poles then **P** and **Q** have to be specified so that:

$$\mathbf{BP} + \mathbf{AQ} = \mathbf{A}_m \qquad \ldots.(1.33)$$

Equation (1.33) establishes requirements on orders of the polynomials:

$$n_P = n_H - 1, \quad n_Q = n_B - 1, \quad n_{Am} = n_A + n_B - 1 \qquad \ldots.(1.34)$$

As **A** and **B** are not known equation (1.33) cannot be solved directly but must be re-expressed in terms of estimated parameters. Multiplying (1.33) by **F** and using the fact that

$\mathbf{H}=\mathbf{BF}+\mathbf{QC}$, with equation (1.21) gives:

$$\mathbf{FA}_m = \mathbf{PH} - q^{-k}\mathbf{GQ} \qquad \ldots.(1.35)$$

Equation (1.35) is the relationship established by Allidina and Hughes [1980] which must be solved at each control step for the polynomials **P** and **Q**, this procedure introducing additional computational requirements into each step of the control loop. However, as the matrices involved in the solution of the simultaneous equations are in nearly upper triangular form, the computational burden can be tolerated. Steady state requirements are assured once again by calculating the steady state loop gain with q=1, and adjusting the gain as necessary.

1.3.2 Deterministic pole assigning controller

Implicit controllers for pole-zero placement have also been described by Astrom and Wittenmark [1979]. Their approach is more direct than that of Allidina and Hughes. A general form for a linear regulator is:

$$\mathbf{H}u(t) + \mathbf{G}y(t) + \mathbf{E}w(t) = 0 \qquad \ldots.(1.36)$$

Equation (1.36) of course is the same as equation (1.25). The problem is to find **H**, **G** and **E**. Astrom considered processes without any noise disturbances, i.e. e(t)=0, so the process to be considered is (with d=0 also):

$$\mathbf{A}y(t) = \mathbf{B}u(t-k) \qquad \ldots.(1.37)$$

Combining (1.36) and (1.37) gives the closed loop transfer function:

$$(\mathbf{HA} + q^{-k}\mathbf{BG})y(t) = -\mathbf{BE}w(t) \qquad \ldots.(1.38)$$

Zeros which are not desired in the final closed loop transfer function must be cancelled, and to this end **B** is factorised to give $\mathbf{B}=\mathbf{B}^{+}\mathbf{B}^{-}$ where $\mathbf{B}^{+}$ contains zeros corresponding to unstable modes which cannot be cancelled. Also it is necessary to introduce $\mathbf{A}_0$ to balance polynomial orders on both sides of the equation below (Astrom and Wittenmark interpret this as an observer polynomial, and it is interesting to compare this to **F** in equation (1.35)). For pole placement it is necessary to solve:

$$\mathbf{H'A} + q^{-k}\mathbf{B^{-}G} = \mathbf{A}_m\mathbf{A}_0 \qquad \ldots.(1.39)$$

where $\mathbf{H}=\mathbf{H'B^{-}}$. To develop an explicit algorithm, (1.39) is multiplied throughout by y(t), and combined with equation (1.37) to give:

$$\mathbf{A}_m\mathbf{A}_0y(t) = \mathbf{H'B}u(t-k) + \mathbf{B^{-}G}y(t-k) = \mathbf{B^{-}}(\mathbf{H}u(t-k) + \mathbf{G}y(t-k))$$

$$\ldots.(1.40)$$

Equation (1.40) is now in a form suitable for least squares identification and hence for self-tuning control. The presence of **B** does pose certain estimation problems, and of particular interest is the case where **B** =1 i.e. where all process zeros have been cancelled. This results in a particularly simple control scheme, which has limitations in that it is not suitable for non-minimum phase systems, and it does not deal with stochastic disturbances. Astrom showed that the controller will develop integral action to deal with constant offsets although these can be dealt with by direct estimation.

1.4 EXPLICIT SELF-TUNING CONTROLLERS WITH POLE ASSIGNMENT

Astrom and Wittenmark [1979] designated controllers as being "implicit" or "explicit": explicit schemes identify a process model, from which the control signal may be calculated, while implicit schemes identify parameters from which the control may be derived without a process model (usually the controller parameters themselves). All the schemes discussed before now have been implicit schemes. Explicit schemes lend themselves more directly to cases where the control signal is derived from considerations of the transfer function of the system.

Astrom [1980] presented closely allied implicit and explicit pole assigning control schemes for deterministic processes. The implicit case has been discussed in section 1.3.2.

1.4.1 Deterministic Pole-Assigning Controller

As before, the explicit schemes are applicable to processes described by (1.37) with a controller of the form of (1.36). If no process zeros are to be cancelled, (1.38) gives the closed loop transfer function, and appropriate poles are selected by choice of a characteristic equation $\mathbf{A}_m$. Then:

$$\mathbf{HA} + q^{-k}\mathbf{BG} = \mathbf{A}_m\mathbf{A}_0 \qquad \ldots.(1.41)$$

The above equation has to be solved for **H** and **G**, which constitute most of the controller parameters. **E** is selected so that $\mathbf{E}=K\mathbf{A}_0$, the K being a scaling factor to ensure the correct steady state gain for the process, calculated from the estimated parameters with q=1.

For the case where all the process zeros are cancelled, **H** is

replaced by $\mathbf{H}'$, containing the factors of $\mathbf{H}$ remaining after the cancellation, and the simultaneous equations to be solved are determined from:

$$\mathbf{AH}' + q^{-k}\mathbf{G} = \mathbf{A}_m\mathbf{A}_0 \qquad \ldots.(1.42)$$

Complete selection of closed loop zeros is possible, all the process zeros having been cancelled. $\mathbf{E}$ is chosen as $\mathbf{E}=\mathbf{B}_m\mathbf{A}_0$ and the correct gain will be ensured if $\mathbf{A}_0$ is normalised so that $A_0(1)=1$.

Astrom and Wittenmark [1980] have discussed certain properties of the control system given by equation (1.41), including the correctness of the parameter estimates, and local stability about the stationary solution. They also showed, by means of example, the degradation in control in the presence of a constant bias (i.e. non-zero d), and overcame this by using the method of Clarke and Gawthrop [1975], of estimating this offset, which could then be taken into consideration when computing the control signal.

The straightforward presentation of these algorithms provides an insight into the behaviour of such systems. However, factorisation techniques may be required for their solution, and it is more general to consider the behaviour of processes in the presence of noise disturbances. The extension to these cases has been given by Wellstead et al [1979,a,b,c].

1.4.2 Stochastic Pole-Assigning Regulator

Controllers of this type have been discussed by Wellstead, Edmunds, Prager and Zanker [1979], and by Wellstead, Prager and

Zanker [1979]. The first paper proved the self-tuning property for the general pole-zero assignment case, while the second justified an unusual model structure which uses leading zeros in polynomials to cater for time delays, and with self-tuning of the parameters is able to model systems with variable time delays. In both of these papers only the regulator was considered.

This regulator was developed to deal with problems of large amplitude control signals, and with difficulties encountered when gain limitations are placed on the control signals, although it should be noted that the formulation proposed by Clarke and Gawthrop [1979] deals with these cases. The ability of the pole placing algorithms to reduce gain, although with an increase in output variance, and to overcome problems encountered with conditional stability was demonstrated.

If the process model of (1.2) is considered, Wellstead et.al. showed that multiplying both sides of the equation by a suitably chosen polynomial $\mathbf{T}$ can yield the alternative structure:

$$(1+q^{-k}\mathbf{A}')y(t) = \mathbf{B}'u(t-k) + \mathbf{CT}e(t) \qquad ....(1.43)$$

(Throughout this section the notation of Wellstead et al has been changed to correspond with the notation already defined). The controller is described by:

$$\mathbf{H}u(t) + \mathbf{G}y(t) = 0 \qquad ....(1.44)$$

Combining (1.43) and (1.44) gives the closed loop description:

$$\{(1+q^{-k}\mathbf{A}')\mathbf{H}+q^{-k}\mathbf{B}'\mathbf{G}\}y(t) = \mathbf{HCT}e(t) \qquad ....(1.45)$$

The aim of pole/zero assignment is to force the closed loop system to be:

$$\mathbf{A}_m y(t) = \mathbf{B}_m e(t) \qquad \ldots.(1.46)$$

This determines the following set of simultaneous equations which must be solved for the controller parameters **H** and **G**:

$$\mathbf{B}_m\{(1+q^{-k}\mathbf{A}')\mathbf{H}+q^{-k}\mathbf{B}'\mathbf{G}\} = \mathbf{A}_m\mathbf{HCT} \qquad \ldots.(1.47)$$

Suitable choice of **H** and **G** can reduce equation (1.47) to the minimum variance regulator of Astrom [1973], but of more interest is the detuned minimum variance controller described by Wellstead et al [1979 a], in which case, the detuning or "tailoring" of the variance is achieved by using the following identity in the k-step predictor:

$$\{\mathbf{CP}/\mathbf{A}\} = \mathbf{F} + q^{-k}\{\mathbf{G}/\mathbf{A}\} \qquad \ldots.(1.48)$$

Where the detuning is achieved by introducing the polynomial **P**. Following a now-familiar analysis, (1.48) and (1.43) give finally:

$$\mathbf{P}y(t) = \mathbf{F}e(t) + \{\mathbf{BF}/\mathbf{C}\}u(t-k) + \{\mathbf{G}/\mathbf{C}\}y(t-k) \qquad \ldots.(1.49)$$

Equation (1.49) can now be seen to be a special case of the controller proposed by Clarke and Gawthrop [1979], with **Q**=**R**=0 (so making it minimum variance), and $\emptyset$=**P**y(t). This special case has been re-expressed in a form suitable for use in an explicit scheme. In particular, if **P** is chosen with the special structure:

$$\mathbf{P} = 1 + q^{-k}\mathbf{P}' \qquad \ldots.(1.50)$$

and assuming the noise is white (**C**=1):

$$y(t) = \mathbf{F}e(t) + \mathbf{BF}u(t-k) + (\mathbf{G}-\mathbf{P}')y(t-k) \qquad \ldots.(1.51)$$

Equation (1.51) makes very clear the explicit nature of the model to be identified, and shows also the modification of the system

poles by **P**. For the self-tuning case then, the model to be identified is:

$$y(t) = \mathbf{A}''y(t-k) + \mathbf{B}''u(t-k) + e'' \qquad ....(1.52)$$

which is the same as equation (1.6). As is normal in self-tuning the parameters are identified as though the noise were white, and the control signal is derived from:

$$(\mathbf{A}''-\mathbf{P}')y(t) + \mathbf{B}''u(t) = 0 \qquad ....(1.53)$$

where equation (1.53) now differs from equation (1.7) in the addition of the detuning polynomial $\mathbf{P}'$. Wellstead, Edmunds, Prager and Zanker [1979] prove that the self-tuning principle applies for the detuned minimum variance case. As can be seen from the above analysis, the control signal still depends on the inversion of **B**, and so this controller will not deal with non-minimum phase systems. For such systems, the complete pole placing algorithm is necessary. In the self-tuning situation the parameters of (1.52) are once again identified (using some least-squares algorithm), and **H** and **G** are calculated by solving:

$$(1+q^{-k}\mathbf{A}'')\mathbf{H} + q^{-k}\mathbf{B}''\mathbf{G} = \mathbf{A}_m\mathbf{T}'' \qquad ....(1.54)$$

Once determined, **H** and **G** are used in equation (1.44) to derive the control signal. The orders of the polynomials are $n_{A''}=n_A-1$, and $n_{B''}=n_B-1$. Once again adherence to the self-tuning principle has been proved .

1.4.3 Pole assigning regulator with varying time delays

The case where the time delay may be varying is dealt with by Wellstead, Prager and Zanker [1979], who introduced a novel model structure with zeros in the leading coefficients of the

polynomials to cater for the process delays. This structure is only allowable with certain algorithms, such as that used for pole assignment. The process is assumed to be described by (1.2) and the controller by (1.44), giving the closed loop description:

$$\{\mathbf{AH}+q^{-k}\mathbf{BG}\}y(t) = \mathbf{HC}e(t) \qquad ....(1.55)$$

The closed loop poles are shifted to the locations defined by $\mathbf{A}_m$ by requiring:

$$\mathbf{AH} + q^{-k}\mathbf{BG} = \mathbf{A}_m\mathbf{C} \qquad ....(1.56)$$

which may now be solved for **H** and **G**, which have orders $n_G=n_A-1$ and $n_H=n_B+k-1$. During self-tuning, the noise is assumed to be white (**C**=1), and the process modelled by:

$$\mathbf{A}'y(t) = \mathbf{B}'u(t) + e'(t) \qquad ....(1.57)$$

The order of $\mathbf{B}'$ has to be extended to accommodate k zero coefficients corresponding to the time delay. The equation to be solved for pole placement is:

$$\mathbf{HA}' + \mathbf{GB}' = \mathbf{A}_m \qquad ....(1.58)$$

The control signal defined by (1.44) is applied to the process. Variable time delays are dealt with by re-estimating the coefficients of $\mathbf{B}'$, which results in a conceptually and notationally simple structure, the price paid for this being the additional elements of $\mathbf{B}'$ which have to be estimated, and of **H** which have to be calculated.

1.4.4 Servo self-tuners

Clarke and Gawthrop [1979] have described a self-tuning

controller where the control signal applied to the process has a polynomial dependence on the reference input (equation (1.24)). Astrom [1980] has also described self-tuning controllers for deterministic systems (equation (1.36)). Possibly the most frequently used technique to ensure set point attainment has been to insert digital integrators into the control loop. Wellstead and Zanker [1979] have used the classical design technique of set-point feedforward to build on the pole assigning regulator described in section 1.4.2 in such a way as to decouple the regulator and servo sections. Their arrangement is best illustrated by means of a block diagram.

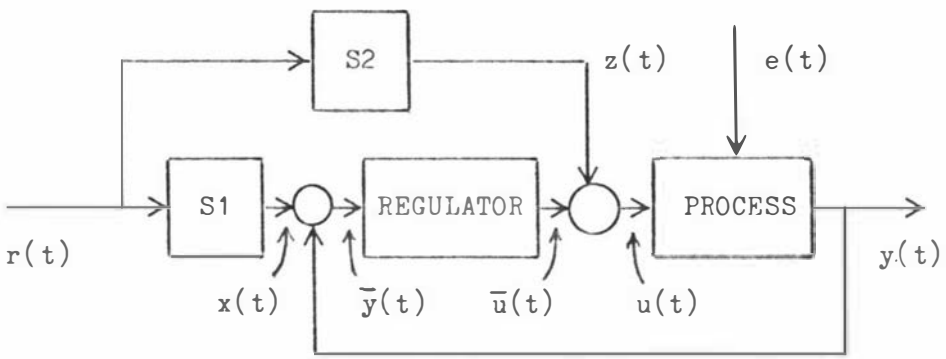


Figure 1.1 Servo control system

The above block diagram serves to define the two transfer functions **S1** and **S2** in addition to the named signals. The process is assumed to be described by equation (1.43), while the operation of the regulator is described by:

$$\bar{u}(t) = \{\mathbf{G/H}\}\bar{y}(t) \qquad \ldots.(1.59)$$

Using the block diagram, a transfer function for the closed loop system is easily obtained as:

$$\{(1+q^{-k}\mathbf{A}')+q^{-k}\mathbf{BG}\}y(t) = -\mathbf{HCT}e(t) + \mathbf{H}\{(1+q^{-k}\mathbf{A}')\mathbf{S1}-q^{-k}\mathbf{BS2}\}r(t) \qquad \ldots.(1.60)$$

The regulator is treated using the methods of section 1.4.2. The servo property is assured by setting $y(t)=e'(t)$, eliminating the effects of $r(t)$:

$$\mathbf{S1} = q^{-k}\mathbf{S} \qquad \text{and} \qquad \mathbf{S2} = \{(1+q^{-k}\mathbf{A}')/\mathbf{B}\}\mathbf{S} \qquad \ldots.(1.61)$$

S is a transfer function which defines the desired servo response. The extension to the self tuning case also relies on the regulator case. Equation (1.52) may be extended to include $x(t)$ and $z(t)$, once again using the block diagram to define the functions. The parameters to be identified are defined by the equation:

$$y(t) = \mathbf{A}''y(t-k) + \mathbf{B}''u(t-k) + \mathbf{L}x(t-k) + \mathbf{M}z(t-k) + e''(t) \qquad \ldots.(1.62)$$

Servo compensation is achieved by suitably selecting $z(t)$ and $x(t)$ at each time step:

$$z(t) = \{(1-q^{-k}\mathbf{L})/\mathbf{M}\}\mathbf{S}r(t)$$

$$x(t) = q^{-k}\mathbf{S}r(t) \qquad \ldots.(1.63)$$

H and **G** are calculated using equation (1.54) and are used in equation (1.59) to provide the regulator part of the control signal.

Wellstead and Zanker [1979,c] have reported that **L** and **M** converge to their unbiased values of **A'** and **B'** respectively, which, together with the biased parameter estimates of equation (1.62) provides a way of obtaining the true system parameters of **A**, **B** and **C**. It was argued that this provided an on-line maximum likelihood estimation algorithm. It can also be seen from equation (1.61) that difficulties may be encountered when inverting **M** (in (1.63)) due to its dependence on **B**. Wellstead and Zanker [1979,c] show by example how **M** may be included as a

factor in the numerator of **S** to avoid such difficulties.

1.5 APPLICATIONS

The practical viability of a particular technique becomes apparent once it has been applied in a variety of situations. The following section provides a review of some reported applications, concentrating particularly on techniques which may have some important influence on the research presented in this thesis.

1.5.1 Implementation

By virtue of their nature, self-tuning regulators require the use of a computer or microprocessor, and with the technological advances and cost of the latter device the use of a digital (DDC) algorithm can now frequently be justified. Of particular interest therefore is the implementation by Clarke and Gawthrop [1979], and colleagues, of a self-tuning controller using a microcomputer system. This was constructed with the following requirements in mind:

1. Portability;
2. Bulk-storage availability, for programs and data;
3. Standard instrumentation compatability (e.g. 4-20 ma current loops); and
4. Suitable high level language capability.

The hardware was a custom built microcomputer based on an Intel

SBC 80/10 single board computer, with 16K bytes of RAM memory, EPROM, two floppy disk drives and analogue interfaces.

The software consisted of a Control Basic interpreter, a disk monitor and machine language utility programs. Most of these software facilities are now provided in other commercially available microcomputer systems, except for the interrupt handling capability, (and in particular foreground/background operation within the same program), and the multi-tasking capability. The importance of the ability to modify programs on site, provided by the use of BASIC, was stressed.

Clarke and Gawthrop, [1979] have reported several applications of their controller. Set point attainment was usually ensured by the introduction of an integrator. Extreme non-linearities (pH control) could be handled, although satisfactory results required the inclusion of an inverse non-linearity and safeguards at the ends of the range. Examination of the underlying continuous process gave insights into the closed loop behaviour, confirming the choice in the pH control of a relatively long sample interval. It is worth noting that a longer sample interval will also require less accurate delay determination. Control of a batch chemical reactor also benefitted from offline estimation, for example during the start-up phase. A particular need was expressed for analysis of the behaviour of a linear self-tuning controller acting on a non-linear process, and linearising about the operating point.

However, in most cases the emphasis has not been on the application of any particular technology, but on the performance of the self-tuning algorithm. Most of the applications have used a process control minicomputer.

1.5.2 Testing the algorithms

Astrom and co-workers have reported applications of the minimum variance regulator with various plants. Astrom et al [1977] envisage three clearly defined ways in which the algorithm may be used. In the first, the algorithm is applied at the start when the dynamics of the system are unknown. Once the controller is operating satisfactorily, its use may be discontinued in favour of the more simple application of the controller so derived. However, the second use of the algorithm would be for periodic retuning of the controller, after which it is once again disconnected as before. In the third case, the controller is used continuously, to cope with continuously varying parameters.

The above paper gives brief descriptions of applications, reported more fully elsewhere, of the minimum variance version of the algorithm, applied to the control of a paper machine in one case, and of an ore crusher in the other. The second case is of interest because the process control computer was remote from the site, modem communication being used for the link-up.

A quadratic criterion admirably expresses the requirements of controllers which act as autopilots for steering tankers. Kallstrom et al [1979] report the use of self-tuning controllers using these criteria for such control. Controllers were reported which used a minimum variance criterion, and which solved a Riccati equation on-line so as to penalise excessive rudder movements in controlling heading error. A variable structure algorithm for controlling turning was also used. In a situation where full scale trials are so expensive, the use of simulations for prior evaluation of controller behaviour was shown to good advantage.

Ledwich [1979] also reports simulations of control of the excitation of a sychronous generator, the control scheme derived from considerations of convergence of the algorithm. In this

case, a minimum variance control law produces unacceptably large control signal amplitudes, while a one-step quadratic criterion has other unacceptable effects. The need to solve a Riccati equation to apply a full quadratic criterion is avoided by rescaling the gain of a controller using a minimum variance law.

The use of a quadratic criterion and the solution of the resulting Riccati equation on line is again reported by Buchholt and Kummel [1979] in the control of a highly non-linear pH-neutralisation process. The identified parameters, and input and output values are used to form a state-space model for which an optimal strategy is computed.

The work of Kershenbaum and Fortesque [1978] in controlling process plant, foretold the widespread use of variable forgetting factors to avoid the extreme sensitivity which results when an exponential forgetting factor is applied for long periods when little excitation is applied to the system, during which time hardly any information is available to the controller about the process dynamics. Whatever method is used, it is important to ensure that the covariance matrix remains bounded.

Keviczky et al [1978] have implemented a multivariable version of a minimum variance-like controller for blending of cement raw material. Their approach, using polynomial matrices, is similar to that of Borisson [1977], who applied multivariable self-tuning control to the header box of a paper machine. These methods have been taken up by Koivo [1980], whose paper was described earlier.

1.6 COMPARISON OF ALGORITHMS

There are numerous ways in which the situation in which self-tuning control is applied may be varied. These may be summarised as:

varying the identification method;
varying the control law;
using implicit or explicit schemes; or
varying the process which is being controlled.

Two studies have been reported which formally set out to compare the performance of self-tuning algorithms, [Kurz, Isermann and Schumann, 1980], [Jacobs and Saratchandran, 1980]. Simulations of the controlled process were used in both these studies. Kurz et al used an analog computer to simulate the process, and a digital minicomputer for the control. A noise generator ensured repeatability of the conditions for each process. Jacobs and Saratchandran used a totally digital simulation to obtain this repeatability for their Monte Carlo simulations.

The simulations of Jacobs and Saratchandran differed from those of Kurz et al in that they attempted to present a simulation study which could lead to conclusions having some general reliability. To this end, the parameter estimation scheme and the controllers were related to a state-space structure, with the parameters given as additional states, to be estimated with an extended Kalman filter. Separate estimation of states and parameters [Nelson and Stear, 1976] gave similar results. The controlled process was limited to first order, and a factorial design varying the three parameters in the model:

$$y(t) + ay(t-1) = bu(t-1) + e(t) + ce(t-1)$$

yielded eight classes of process, for $|a|<1$ or $|a|>1$, sgn(b) known or unknown, and $|c|>1$ or $|c|<1$. Kurz et al on the other hand chose eight different processes which exercised the ability of the controllers to perform in different situations. The processes possessed the following properties:

- low pass
- low pass, one zero outside unit circle

- damped oscillations
- damped oscillations with time delay
- all pass
- integral
- oscillating unstable
- monotone unstable

Recursive least squares and recursive maximum likelihood were combined with six different control schemes - two minimum variance, two deadbeat, one PID and one pole assignment, giving a total of twelve different adaptive schemes. One of the problems in comparing controllers designed to different criteria is how to make the comparison between them. Kurz et al [1980] used r.m.s. values of both the process input and process output.

Jacobs and Saratchandran [1980] compared schemes based on cost function minimisation, namely the optimal k-step ahead controller, and the controller of Clarke and Gawthrop [1975] which is a k-step ahead controller based on certainty equivalence. In these cases, the cost function provides a direct means of comparison. The costs are bounded below by the cost incurred when a controller is designed with full knowledge of the parameters, and bounded above by the cost of a controller where no adaptation is allowed to take place. The optimal controller is the "more optimal" of the two, so that where the results were distinguishable, it performed better. In other cases, the simpler-to-implement self-tuning controller is to be preferred. It is of interest that neither controller coped with processes characterised by $|a|>1$ AND sgn(b) uncertain.

Kurz et al [1980] also recommend different controllers for different situations. Adaptation was generally very quick, particularly for recursive least squares. The pole assignment controller, although imposing a relatively heavy overhead as regards computational requirements, is recommended where a wide influence on control behaviour is required.

Chapter 2
SELF-TUNING ALGORITHMS

2.0 INTRODUCTION

Chapter Two presents the self-tuning algorithms used in the research reported in this thesis. The multivariable controllers are presented in such a way that scalar processing techniques may be used for their implementation. Principal amongst these is a pole assigning algorithm, which may be used in full, or simplified slightly by assuming that the main control input to a particular loop may be taken to have an effect on that loop before disturbances from inputs to the other loops. The pole assigning controllers are explicit control schemes. An implicit scheme is also presented which extends a self-tuning controller with a one-step quadratic criterion to deal with multivariable systems.

2.1 POLE PLACING MULTIVARIABLE SELF-TUNING CONTROL

First the structure of the equations which are to be used will be established. A statement of the control problem will be followed by its solution for certain classes of processes, and the similarity between these solutions and those for the scalar case will be established. The scalar representation will be used to develop alternative results, and to point out relationships to other controllers.

2.1.1 Process Description

It is necessary to establish the structure of the equations representing the multivariable system. Following the lead of other researchers [Sandoz and Wong, 1978], the state space representation will be used as a basis for a suitable structure, and from this will be derived the equations to be used for the self-tuning controller. The state space equations are:

$$x(k+1) = \Phi x(k) + \Delta u(k) + \Gamma w(k)$$
$$y(k) = \Psi x(k) + \zeta(k) \qquad \ldots.(2.1)$$

where x(k) is the n-dimensional state, u(k) is the m-dimensional input, y(k) the p-dimensional measured output, w(k) is the r-dimensional state noise and $\zeta(k)$ the p-dimensional output noise. If these equations are given in phase canonical form, it is possible to eliminate the state vector and express them in terms of the system inputs and outputs [Nelson and Stear, 1976]. This results in the familiar Auto Regressive Moving Average (ARMA) representation. The paper of Nelson and Stear [1976] is important because it provides the link between the state space and the ARMA forms.

$$\mathbf{A}y(t) = q^{-k}\mathbf{B}u(t) + \mathbf{C}e(t) \qquad \ldots.(2.2)$$

A, **B** and **C** are polynomial matrices, capable of being represented in two ways, for example:

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_{11} & \ldots\ldots & \mathbf{B}_{1m} \\ & & \\ \mathbf{B}_{p1} & \ldots\ldots & \mathbf{B}_{pm} \end{bmatrix} \qquad \ldots.(2.3)$$

Here the $\mathbf{B}_{ij}$ are scalar polynomials. An alternative way of representing the same matrix is:

$$\mathbf{B} = \mathbf{B}_0 + \mathbf{B}_1 q^{-1} + \ldots\ldots + \mathbf{B}_n q^{-n} \qquad \ldots.(2.4)$$

The $\mathbf{B}_i$ are p x m coefficient matrices. **A** and **C** have the same form as this, consistent with maintaining the appropriate representation for the process, i.e.

$$\mathbf{A} = \mathbf{I} + q^{-1}\mathbf{A}_1 + \ldots\ldots$$

$$\mathbf{B} = \mathbf{B}_0 + q^{-1}\mathbf{B}_1 + \ldots\ldots$$

$$\mathbf{C} = \mathbf{I} + q^{-1}\mathbf{C}_1 + \ldots\ldots \qquad \ldots.(2.5)$$

The different inputs to different loops may have different delays associated with them, but all are grouped into a single delay k so that:

$$k = \min_{ij}\{k_{ij}\} \qquad \ldots.(2.6)$$

This may mean that some of the scalar polynomials in the input driving matrix may need to have leading zeros to cope with the additional delay associated with that input as described in Section 1.4.3.

In the controller development it is possible now to incorporate the k-step predictor for a multivariable system given by Borisson [1979] and Koivo [1980]. However, the work by Wellstead, Prager and Zanker [1979] and Astrom and Wittenmark [1979] suggests a more direct approach which will be followed here.

2.1.2 Multivariable Controller

The process to be controlled may incorporate measured disturbance terms which are fed forward, and measurement offsets. This results in the following representation:

$$\mathbf{A}y(t) = q^{-k}\mathbf{B}u(t) + q^{-k}\mathbf{D}v(t) + \mathbf{d} + \mathbf{C}e(t) \qquad \ldots.(2.7)$$

where **A**, **B**, **C** and **D** are polynomial matrices as described above, the v(t) are measured disturbance variables, and **d** is a diagonal matrix containing the offsets for each loop. It will further be assumed that there is one input to each loop, i.e. m=p.

The controller assumes a general form, and incorporates the set point w(t).

$$\mathbf{H}u(t) + \mathbf{G}y(t) - \mathbf{E}w(t) + \mathbf{F}v(t) = 0 \qquad \ldots.(2.8)$$

H, **G**, **E** and **F** are polynomial matrices. It is possible to select **E** so that it has polynomials only on the diagonal i.e. $\mathbf{E}=\text{diag}\{\mathbf{E}_{ii}\}$. The following analysis is given for the "i"th loop only, all loops being treated in the same way. Reducing the "i"th loop to scalar terms:

$$\mathbf{A}_{ii}y_i(t) + \mathbf{A}_{ij}y_j(t) + \ldots = q^{-k}\mathbf{B}_{ii}u_i(t) + q^{-k}\mathbf{B}_{ij}u_j(t) + ..$$

$$+ q^{-k}\mathbf{D}_{ii}v_i(t) + q^{-k}\mathbf{D}_{ij}v_j(t) + \ldots$$

$$+ d_i + \mathbf{C}_{ii}e_i(t) + \mathbf{C}_{ij}e_j(t) + \ldots$$

$$\ldots.(2.9)$$

In the same way the controller may be expressed using scalar polynomials as:

$$\mathbf{H}_{ii}u_i(t) = -\mathbf{H}_{ij}u_j(t) - \ldots - \mathbf{G}_{ii}y_i(t) - \mathbf{G}_{ij}y_j(t) - \ldots$$

$$+ \mathbf{E}_{ii}w_i(t) - \mathbf{F}_{ii}v_i(t) - \mathbf{F}_{ij}v_j(t) - \ldots \qquad \ldots.(2.10)$$

Combining these two equations gives the expression for the closed

loop behaviour of the process and the controller together:

$$(\mathbf{H}_{ii}\mathbf{A}_{ii}+q^{-k}\mathbf{B}_{ii}\mathbf{G}_{ii})y_i(t) + (\mathbf{H}_{ii}\mathbf{A}_{ij}+q^{-k}\mathbf{B}_{ii}\mathbf{G}_{ij})y_j(t) + \ldots$$

$$= q^{-k}(\mathbf{H}_{ii}\mathbf{B}_{ij}-\mathbf{B}_{ii}\mathbf{H}_{ij})u_j(t) + \ldots.$$

$$+ q^{-k}(\mathbf{H}_{ii}\mathbf{D}_{ii}-\mathbf{B}_{ii}\mathbf{F}_{ii})v_i(t) + q^{-k}(\mathbf{H}_{ii}\mathbf{D}_{ij}-\mathbf{B}_{ii}\mathbf{F}_{ij})v_j(t) + \ldots$$

$$+ q^{-k}\mathbf{B}_{ii}\mathbf{E}_{ii}w_i(t) + \mathbf{H}_{ii}d_i + \mathbf{H}_{ii}\mathbf{C}_{ii}e_i(t) + \ldots. \qquad \ldots.(2.11)$$

Equation (2.11) gives the requirements placed upon the various scalar polynomials. Before stating these explicitly however, it is instructive to examine a minimum variance controller with feedforward terms to see how these are handled.

2.1.3 Scalar Minimum Variance Regulation with Feedforward

The minimum variance controller is examined at this stage to determine how the feedforward terms may be dealt with. It is assumed that the process may be described by the linear difference equation:

$$\mathbf{A}^{*}y(t) = q^{-k}\mathbf{B}^{*}[u(t) + x(t)] + q^{-k}\mathbf{D}^{*}v(t) + d^{*} + \mathbf{C}e(t) \qquad \ldots.(2.12)$$

The process input (controlled variable) is the composite signal $[u(t) + x(t)]$. It is written in this way to explicitly define the

feedback portion u(t), and the feedforward portion x(t), of the signal. It is possible to loosely assign certain operations to each portion, although as will be seen, this assignment is only arbitrary. For the purposes of the present development, u(t) will be concerned with regulation and set-point attainment, while x(t) will be used for disturbance rejection and elimination of the offset.

A k-step prediction of the output which incorporates noise values far enough in the past to be determined, is formed using the identity:

$$\mathbf{C/A^*} = \mathbf{F} + q^{-k}\{\mathbf{G/A^*}\} \qquad \ldots.(2.13)$$

Combining this with the process description (equation (2.12)):

$$y(t) = q^{-k}\{\mathbf{B^*/A^*}\}[u(t) + x(t)] + q^{-k}\{\mathbf{D^*/A^*}\}v(t) + \{1/\mathbf{A^*}\}d^* + \mathbf{F}e(t) + q^{-k}\{\mathbf{G/A^*}\}e(t) \qquad \ldots.(2.14)$$

e(t) is unknown, but e(t-k) may be calculated from already-measured variables using equation (2.12). Eliminating e(t-k):

$$y(t) = q^{-k}\{\mathbf{B^*F/C}\}[u(t) + x(t)] + q^{-k}\{\mathbf{D^*F/C}\}v(t) + q^{-k}\{\mathbf{G/C}\}y(t) + \{\mathbf{F/C}\}d^* + \mathbf{F}e(t) \qquad \ldots.(2.15)$$

e(t) is uncorrelated with any of the other variables, so the cross products vanish when computing the variance $\mathbf{E}\{y(t)^2\}$. Minimum variance is achieved for the control signal:

$$[u(t) + x(t)] = -\{\mathbf{G/B^*F}\}y(t) - \{\mathbf{D^*/B^*}\}v(t) - \{1/\mathbf{B^*}\}d^* \qquad \ldots.(2.16)$$

Without the offset, this is the same as the result given by Astrom et al [1977]. Equation (2.16) serves to define u(t), which

is required for regulation, and x(t) which is the feedforward signal required for disturbance rejection of v(t). It is immaterial whether the offset d is dealt with by u(t) or x(t). Because it is constant (or only slowly varying), the term $\{\mathbf{1/B}\}d''$ can be replaced by $\{1/B(1)\}d$ where B(1) is **B** evaluated with q=1. For computational simplicity, d" will be considered with x(t). The feedforward signal is given by:

$$\mathbf{B}''x(t) + \mathbf{D}''v(t) + d'' = 0 \qquad \ldots.(2.17)$$

The minimum variance nature of this action is important, as x(t) is used to cancel disturbance effects before they can appreciably perturb the process. Equation (2.17) should be compared with the original process description, equation (2.12), and its similarity to part of that equation noted.

2.1.4 Pole-Placing Controller

The signal u(t) in equation (2.16) is used to eliminate as much of the noise as possible. The minimum variance controller performs this task at the expense of sometimes excessive control signal amplitudes. This may be overcome by appropriately selecting the poles of the closed loop transfer function to obtain a satisfactory compromise for the particular system. It is now possible to return to equation (2.11) to provide a multivariable pole-placing controller. It is necessary to eliminate the effect of the off-diagonal polynomials in the expression for the closed loop system behaviour, which will result in a diagonal-dominant decoupled system. The polynomials appearing in the diagonal terms are chosen according to a pole placing law. Specifically, $\mathbf{H}_{ii}$ and $\mathbf{G}_{ii}$ are chosen to satisfy:

$$\mathbf{H}_{ii}\mathbf{A}_{ii} + q^{-k}\mathbf{B}_{ii}\mathbf{G}_{ii} = \mathbf{C}_{ii}\mathbf{A}_{mi} \qquad \ldots.(2.18)$$

If it is to be possible to set the off-diagonal elements to zero, then:

$$\mathbf{H}_{ii}\mathbf{A}_{ij} + q^{-k}\mathbf{B}_{ii}\mathbf{G}_{ij} = 0$$

$$\mathbf{H}_{ii}\mathbf{B}_{ij} - \mathbf{B}_{ii}\mathbf{H}_{ij} = 0$$

$$\mathbf{H}_{ii}\mathbf{D}_{ii} - \mathbf{B}_{ii}\mathbf{F}_{ii} = 0$$

$$\mathbf{H}_{ii}\mathbf{D}_{ij} - \mathbf{B}_{ii}\mathbf{F}_{ij} = 0 \qquad \ldots.(2.19)$$

These conditions will be able to be met only if the following is true:

(i) The zeros of $\mathbf{B}_{ii}(z)$ are outside the unit disk

(ii) k=1

The first condition means that the polynomial will be invertible, a condition which must be met if the off-diagonal polynomials are to be calculated. If k>1 then a non-causal control action results in eliminating the cross-coupling effect of $y_j(t)$.

In general it is not possible to expect these conditions to be met, and some mechanism to deal with these situations must be found. Such a mechanism was outlined by Hesketh [1980]. It is possible to select:

$$\mathbf{G}_{ij} = -\mathbf{H}_{ii}A_{ij}(1)/B_{ii}(1)$$

$$\mathbf{H}_{ij} = \mathbf{H}_{ii}B_{ij}(1)/B_{ii}(1)$$

$$\mathbf{F}_{ii} = \mathbf{H}_{ii}D_{ii}(1)/B_{ii}(1)$$

$$\mathbf{F}_{ij} = \mathbf{H}_{ii}D_{ij}(1)/B_{ii}(1) \qquad \ldots.(2.20)$$

In general the above selection will result in imperfect decoupling of the loops, as the off-diagonal terms will be zero only in the steady state. As will be demonstrated with examples however, the resulting arrangement will be diagonally dominant, complete decoupling being achievable only for certain special cases. Attempting to decouple the loops as described above decreases the amount of computation necessary for the multivariable processes, as each loop is dealt with as though it were a separate scalar process.

Of more concern is the possible loss of the self-tuning property that may result from the approximations described above. While this property is undoubtedly lost at times, it will be shown in Chapter 4 that such loss is minimal, the controller being robust enough to deal with the approximations in all but the most severe situations. This is easily illustrated using simulations.

To ensure set-point attainment, the offset terms must be eliminated, and this robustness is achieved as in equation (2.16) by introducing an auxiliary control signal for each loop such that:

$$x_i(t) = -d_i/B_i(1) \qquad \ldots.(2.21)$$

The control signal applied to the loop is thus $[u_i(t)+x_i(t)]$.

2.1.5 Self Tuning On-Line

During self-tuning it is assumed that $\mathbf{C}=\mathbf{I}$, i.e. that the process noise is white. If the controller has the self-tuning property, then despite this assumption, the estimated model parameters will converge to values which give the correct closed loop behaviour (i.e. that determined from the off-line model), [Astrom, 1973].

A model incorporating a k-step predictor may be derived if the time delay is known a-priori. However, Wellstead et al [1979,b] proposed an alternative model structure in which leading zeros are incorporated into driving polynomials to cater for delays. Because these leading zeros are identified, variable delays can be dealt with. There is also a small computational advantage for subsequent pole placing calculations [Allidina and Hughes, 1980]. Under the assumption that $\mathbf{C}=\mathbf{I}$, the online model is:

$$\mathbf{A}'y(t) = q^{-1}\mathbf{B}'u(t) + q^{-1}\mathbf{D}'v(t) + d + e(t)$$

$$n_{A'}=n_A \qquad n_{B'}=n_B+k-1 \qquad n_{D'}=n_D+k-1$$

....(2.22)

A pole placing controller is calculated using

$$\mathbf{H}u(t) + \mathbf{G}y(t) - \mathbf{E}w(t) = 0 \qquad \ldots.(2.23)$$

where the $\mathbf{E}=\mathrm{diag}\{\mathbf{E}_{ii}\}$ polynomials are selected according to desired controller properties, and scaled so that correct set point attainment is assured, and $\mathbf{H}$ and $\mathbf{G}$ are derived from:

$$H_{ii}A'_{ii}+q^{-1}B'_{ii}G_{ii} = A_{mi}$$

$$H_{ij} = H_{ii}B'_{ij}(1)/B'_{ii}(1) \qquad j\neq i$$

$$G_{ij} = H_{ii}A'_{ij}(1)/B'_{ii}(1) \qquad j\neq i$$

$$F_{ij} = H_{ii}D_{ij}(1)/B_{ii}(1) \qquad \text{For all } j \qquad \ldots.(2.24)$$

At each interval, the parameters of equation (2.22) are estimated using the Kalman Filter U-D measurement update algorithm [Bierman, 1976]. Equations (2.23)-(2.24) are used to derive the control signals u(t).

2.1.6 Set Point Attainment

Set point attainment is ensured by removing the effect of the steady state offset, and by ensuring that the closed loop transfer function for each loop, which relates the set point to the output for that loop, has unity gain. The "decoupling" of the loops means that a change to the operating point of one loop will not effect the set point attained on another loop, at least once the transients have decayed. Equation (2.21) gives the steady state offset control:

$$u'_i(t) = u_i(t) - d_i/B_{ii}(1) \qquad \ldots.(2.25)$$

The closed loop transfer function for the "i"th loop is:

$$y_i(t) = q^{-1}\{B'_{ii}E_{ii}/[HA'_{ii}+q^{-1}B'_{ii}G_{ii}]\}w_i(t)$$

$$= q^{-1}\{B_{ii}E_{ii}/A_{mi}\}w_i(t) \qquad \ldots.(2.26)$$

The process zeros (B_{ii}) can be cancelled if they are included as factors in A_{mi}. However, this can lead to problems with non-minimum phase processes. Whether B_{ii} is cancelled or not, E_{ii} must be selected to ensure an overall gain of 1, i.e.:

$$E_{ii} = E^{\bullet}_{ii}A_{mi}(1)/[B'_{ii}(1)E'_{ii}(1)] \qquad \ldots.(2.27)$$

where $E^{\bullet}_{ii}$ is the unscaled polynomial. $E^{\bullet}_{ii}$ may be chosen in various ways depending on the desired response of y(t) to w(t). To retain the flexibility this gives, $E^{\bullet}_{ii}$ is chosen along with A_{mi} when the controller is specified. Two cases are particularly worthy of mention.

(i) $E1'_{ii} = A_{mi}$

In this case a servo controller results which has a behaviour very similar to that of the controller described by Wellstead and Zanker [1979,b].

(ii) $E2'_{ii} = (1 + q^{-1} + q^{-2} + \ldots)$

This ensures a gradual application of the control signal following a sudden set-point change. The conservative nature of the resulting control action probably justifies the adoption of this polynomial as the default selection. Note also that if a selection of $E'=E1^{\bullet}.E2^{\bullet}$ is made, a ramp response can be obtained

following a step change in w(t).

The controller may be less flexibly but more definitely endowed with servo properties by incorporating the reference signal as a feedforward term [Astrom et al, 1977; Wellstead and Zanker, 1979]. This has the disadvantage of increasing the number of parameters to be estimated during self-tuning.

2.1.7 Unmeasured Disturbances

Disturbance rejection has been ensured by incorporating disturbance terms into the controller equation (2.23). If $v_i(t)$ is unmeasured, then it must be estimated as though it were part of d_i, the constant offset. This will suffice if $v_i(t)$ changes only slowly or infrequently. The re-estimation of d_i results in behaviour akin to a slow integral action.

Recovery following a change in $v_i(t)$ (or d_i as it must now be considered) may be accelerated if some adjustment of d_i be made outside the least squares algorithm (Kalman Filter). It has been found that the most successful way of making the adjustment has been to use the error from set point $[y_i(t)-w_i(t-k)]$. The adjustment to d_i is made as:

$$d_i(t+1|t) = d_i(t) + a_i[y_i(t) - w_i(t-k)] \qquad ....(2.28)$$

Note the timing of the change, which is made before the parameter estimation, as a time update to the Kalman filter estimate, the updated estimate then being smoothed by the Kalman filter measurement update routine. The a_i are arbitrary parameters related to the required rate of response to disturbance changes, and may be considered as "reset" parameters. Default values of

0.01 are assigned to these parameters.

2.1.8 Restatement of the Multivariable Controller

Clarke and Gawthrop [1975], have suggested that the single loop controller may be extended to the multivariable case by considering that the inputs to, and the outputs from all interacting loops behave as though they were disturbances to the loop in question. Borisson [1977], describing the system using polynomial matrices, deals similarly with the problem, but assumes that disturbance effects from the inputs to other loops have a delayed action apon the loop in question, so as not to impede calculation of the control signal. Hesketh [1980] extends the Clarke/Gawthrop controller to deal with the multivariable case in the same way.

Equation (2.23) is used to calculate the control signals once **H** and **G** are known. Rewriting (2.23):

$$\begin{bmatrix} h_{110} & h_{120} & \cdots \\ \cdots & \cdots & \cdots \\ h_{n10} & \cdots & h_{nn0} \end{bmatrix} \begin{bmatrix} u_1 \\ \cdots \\ u_n \end{bmatrix}$$

$$= -[q^{-1}\mathbf{H}_1+q^{-2}\mathbf{H}_2+\ldots]u(t)+\mathbf{E}w(t)-\mathbf{G}y(t)-\mathbf{F}v(t) \quad \ldots.(2.29)$$

These simultaneous equations must be solved to give the control inputs to the two loops.

Borisson [1977] made the simplifying assumption that the effect of the input to one loop is delayed before it is experienced at another loop, implying that $h_{ij0}=0$, the leading zero in the polynomial delaying the application of the signal until the next sampling instant. The coefficient matrix of (2.29) then becomes diagonal. In the general case, equation (2.29) is solved by triangularisation using the same routine as is used to solve for **H** and **G**.

2.1.9 Non-Minimum Phase

The inversion of the polynomial **B** implicit in control schemes described above is not possible if **B** is non-minimum phase. The situation may be dealt with in one of two ways.

The simplest action is to leave the zeros uncancelled but to normalise the polynomial so that in the steady state its effect is multiplication by unity. B(1) must be normalised to unity, i.e. the **B** polynomial must be divided by B(1), as illustrated below.

Normal inversion

$$\mathbf{B}x(t) = 1$$

$$x(t) = 1/\mathbf{B} = (1-b_1q^{-1}x-\ldots)/b0 \qquad \ldots.(2.30)$$

No zero cancellation

$$\mathbf{B}x(t) = B(1)\mathbf{B}'x(t) = 1$$
$$\mathbf{B}'x(t) = 1/B(1) \qquad \ldots.(2.31)$$

B(1) is a scalar. $\mathbf{B}'x(t)$ has the same steady state value as x(t).

An alternative approach is to cancel only those factors of **B** which are non-minimum phase, but this requires factorisation of the polynomial. However, it is possible to break up the polynomial as follows:

$$\mathbf{B}x(t) = (\mathbf{B1}+q^{-1}\mathbf{B2})x(t) = 1$$

$$x(t) \sim 1/(\mathbf{B1}+\mathbf{B2}) \qquad \ldots.(2.32)$$

B can if necessary be broken down into a number of sequences each delayed by a different number of samples. There is a physical justification for this approach. A frequent cause of a non-minimum phase polynomial is a fractional process delay, resulting for example in a small value of b_0. If the polynomial is broken up as shown above, it has the effect of applying a signal to the process which while incorrect for disturbance cancellation in the current sampling instant will produce the correct effect after the next sampling instant. The additional delay associated with **B2** will result in increased variance, which may be quantified using methods described by Astrom [1970].

2.2 IMPLICIT MULTIVARIABLE CONTROLLER:

There are times when the requirements placed upon a controller are admirably specified by a cost function which weights both output error (from a set point), and input effort. The resulting optimality, combined with the appealing concept of implicit controller design, make the Clarke/Gawthrop controller an attractive proposition. A multivariable scheme based on their design has been proposed by Koivo [1980]. Previous work by Borisson [1979] had laid the groundwork for this development, and prompted a controller proposed by Hesketh [1980]. Clarke and

Gawthrop [1975] had indicated how multivariable extensions could be catered for.

This section will show how the scalar controller design can be used to obtain an implicit multivariable controller, following the methods of the previous section for explicit controllers.

The equation development bears a marked similarity to that followed in developing the equations for minimum variance control (equations (2.12) to (2.17)). The process is described as by (2.12) as:

$$\mathbf{A}y = q^{-k}\mathbf{B}u(t) + q^{-k}\mathbf{D}v(t) + d + \mathbf{C}e(t) \qquad ....(2.33)$$

Using (2.13), a prediction of y(t) given by (2.15) is obtained. This is modified by "detuning" as described by Wellstead et. al. [1979a]. Their detuning concept may be incorporated into the structure of the implied self tuning controller by writing:

$$\mathbf{P}y = q^{-k}\{(\mathbf{BF/C})u(t) + (\mathbf{DF/C})v(t) + (\mathbf{G/C})y(t)] + (\mathbf{F/C})d + \mathbf{F}e(t) \qquad ....(2.34)$$

2.2.1 Self-tuning:

Equation (2.34) defines the auxiliary function ϕ_y described by Clarke and Gawthrop [1975]. The parameters of this equation are identified in the self-tuning procedure as follows:

$$\mathbf{P}y = \phi_y = q^{-k}[\mathbf{H}u(t) + \mathbf{E}v(t) + \mathbf{G}y(t)] + d' - \Sigma c_i \phi_{yi}$$

Where: $d' = F(1)d$ $\mathbf{H} = \mathbf{BF}$ $\mathbf{E} = \mathbf{DF}$ ....(2.35)

2.2.2 Closed Loop

Clarke and Gawthrop [1979] show that the loop may be closed, and a one step quadratic criterion imposed by exerting a control signal given by:

$$\mathbf{Q}u(t) = \mathbf{R}w(t) - \emptyset_y - \mathbf{S}v(t) - Kd' \quad ....(2.36)$$

The detuned minimum variance control imposed by the introduction of **P** in equation (2.34) is thus extended with the **Q** weighting on u(t). The **R** polynomial is necessary to ensure correct set point attainment [Allidina and Hughes, 1979], while the **S** polynomial will ensure disturbance rejection [Hesketh, 1980], and the offset term is included for the same reason, with K to be determined.

Equations (2.35) and (2.36) are inserted into the system equation (2.33) to obtain the closed loop transfer functions relating the output from the loop to the set point, disturbances, offset and noise.

$$[\mathbf{Den}]\, y(t) = \mathbf{HR}\, q^{-k}w(t) + (\mathbf{EQ}-\mathbf{HS})q^{-k}v(t) + (\mathbf{Q}-\mathbf{HK}q^{-k})d$$

$$+ (\mathbf{F}/\mathbf{C})(\mathbf{QC}+\mathbf{H})e(t)$$

With $[\mathbf{Den}] = \mathbf{PH} + \mathbf{PQC}-q^{-k}\mathbf{GQ}$ ....(2.37)

The various transfer functions have to be taken one at a time, and for each, a decision made as to the desired effect of the input signal in question on the output.

2.2.3 Set Point Attainment

The transfer function relating y(t) to w(t) should be unity to ensure set point attainment for the loop. Thus:

$$H(1)\ R(1) = [Den](1) \qquad \ldots.(2.38)$$

Apart from the normalisation of **R**, this does not impose further restrictions upon the choice of **R**, and some of the comments made about the **E** polynomial for the pole placing controller (Section 2.1.6) would apply here.

2.2.4 Disturbance Rejection

Disturbance rejection will remove interactions between loops, and also remove the effects on a loop of measured disturbances. This is ensured by setting:

$$\mathbf{EQ} - \mathbf{HS} = 0 \qquad \ldots.(2.39)$$

S must be chosen (or at least scaled) appropriately, [Hesketh, 1980].

2.2.5 Offset Elimination:

The scalar quantity K must be chosen so that:

$$Q(1) - H(1)\ K = 0 \qquad \ldots.(2.40)$$

It is assumed that the offset term d changes only infrequently or slowly, so that estimation of d can be used to eliminate its effect. The value of d will include the steady state value of any disturbances which are not measured. The re-estimation of d following any change in its value can be accelerated as described

in section 2.1.7.

To maintain stability while the values of **R** and **S** are being changed, the changes are introduced gradually as for example:

$$\mathbf{R}_{new} = \mathbf{R}_{old}\{0.8+0.2[Den](1)/H(1)R(1)_{old}\} \qquad ....(2.41)$$

2.2.6 Other loop inputs as disturbances

Difficulty is likely to arise if the input to one loop is the disturbance for another, which is usually the case in multivariable systems. As a result, inputs have to be determined simultaneously, and until all are known, the disturbance effects that each will have on another loop cannot be assessed. Borisson [1979] suggested that the problem may be overcome by delaying the effect of any input on any other loop. This does degrade disturbance rejection, the amount of degradation experienced depending on the process being controlled. However, such a procedure enables the scalar representation to be retained.

2.2.7 Summary

The implicit controller identifies the parameters of equation (2.35). The controller is termed "implicit" because these parameters are associated with the controller, rather than with a process model. These identified parameters are then used to calculate the input to the process using equation (2.36). Some additional controller parameters, which are not identified, have to be specified by the operator for this equation (**Q**, **R**, **S**, K). Set point attainment, offset elimination, and disturbance rejection are ensured by suitably adjusting the values for **R**, **S** and K. The one step criterion:

$$J = E\{\phi^2\}$$

$$\phi = \mathbf{P}y + q^{-k}(\mathbf{Q}u - \mathbf{R}w + \mathbf{S}v)$$

is minimised by this controller.

2.3 FULL MULTIVARIABLE POLE-PLACING CONTROLLER

Equation (2.2) expresses in ARMA form the model of the process to be controlled, and the paper by Nelson and Stear [1976] relates the parameters of this model to the observable canonical state-space representation. The ARMA form is repeated for convenience as:

$$\mathbf{A}'y(t) = q^{-k}\mathbf{B}'u(t) + \mathbf{C}'e(t) \qquad ....(2.42)$$

A regulator for this process takes the form of:

$$\mathbf{H}u(t) + \mathbf{G}y(t) = 0 \qquad ....(2.43)$$

where **H** and **G** are polynomial matrices to be determined. For the single-input-single-output case discussed in section 1.4.3, **BH=HB** so that it is possible to derive a polynomial equation to be solved for the controller parameters. However, in the multivariable case these polynomial matrices do not commute, and it becomes necessary to define additional polynomial matrices so that for example $\mathbf{H}'\mathbf{B}=\mathbf{B}'\mathbf{H}$. Multivariable minimum variance controllers have been designed using a suitable artifice similar to that above, [Borisson, 1979; Koivo, 1980]. It is not fruitful to define $\mathbf{H}'$ and $\mathbf{B}'$ as above, and an alternative method may be employed to find **H** and **G**. The method will be outlined below.

2.3.1 Self-tuning case

The separation theorem [Astrom, 1970] asserts that the control law may be derived as for the noise-free case, and the self-tuning principle states that the control law calculated from the biassed parameter estimates will converge to the control law that would have been derived using a k-step predictor, with a full knowledge of the noise process. For the self-tuning case, let the parameters which are estimated (i.e. the biassed parameters), be the parameters of the model:

$$\mathbf{A}y(t) = q^{-1}\mathbf{B}u(t) + e(t) \qquad \ldots.(2.44)$$

The **B** polynomial has zero coefficient matrices to cater for time delays, while e(t) is a white noise sequence. This may be re-formulated for ease of manipulation when there are an equal number (m) of inputs and outputs:

$$\begin{bmatrix} .. \\ u(t-2) \\ u(t-1) \\ .. \\ y(t-1) \\ y(t) \end{bmatrix} = \begin{bmatrix} ..... & ..... \\ .... 0\ \ I & .... 0\ \ 0 \\ .... 0\ \ 0 & .... 0\ \ 0 \\ ..... & .. 0\ \ I\ \ 0 \\ .... 0\ \ 0 & .... 0\ \ I \\ .... \mathbf{B}_2\ \ \mathbf{B}_1 & .... \mathbf{A}_2\ \ \mathbf{A}_1 \end{bmatrix} \begin{bmatrix} .. \\ u(t-3) \\ u(t-2) \\ .. \\ y(t-2) \\ y(t-1) \end{bmatrix} + \begin{bmatrix} . \\ 0 \\ I \\ . \\ 0 \\ \mathbf{B}_0 \end{bmatrix} u(t-1) + e(t)$$

$$\text{i.e. } x(t) = \mathbf{D}\,x(t-1) + \mathbf{E}\,u(t-1) + e(t) \qquad \ldots.(2.45$$

The pole-placement design procedures of Wolovich [1974] may be applied to the pseudo-state-space form of equation (2.45). The requirement placed on **D** and **E**, is that this pair of matrices be controllable, as this representation must be transformed to a controllable companion form for the analysis that follows. The

controllability matrix is

$$\mathbf{J} = [\mathbf{E}, \mathbf{DE}, \mathbf{D^2E}, \ldots] \qquad \ldots.(2.46)$$

which must be of full rank if the pair $(\mathbf{D},\mathbf{E})$ is controllable. If $\mathbf{J}$ is full rank it is possible to form:

$$\mathbf{Q} = [\mathbf{Q}^*, \mathbf{Q}^*\mathbf{D}, \mathbf{Q}^*\mathbf{D}^2, \ldots]^T \qquad \ldots.(2.47)$$

where $\mathbf{Q}^*$ is composed of the last m rows of $\mathbf{J}^{-1}$. Following the analysis of Wolovich [1974, p78-79] for scalar processes, and the subsequent extension to multivariable processes, it is apparent that the transformation $x'=\mathbf{Q}x$ yields:

$$x'(t) = \mathbf{QDQ}^{-1}x'(t-1) + \mathbf{QE}u(t-1) \qquad \ldots.(2.48)$$

where

$$\mathbf{QE} = [0,0, \ldots\ldots ,I]^T$$

$$\mathbf{QDQ}^{-1} = \begin{bmatrix} 0 & I & 0 & \ldots\ldots\ldots\ldots\ldots & \\ 0 & 0 & I & .. & \\ & & & & \\ & & & & \\ \ldots\ldots\ldots\ldots\ldots & & & 0 & I & 0 \\ \ldots\ldots\ldots\ldots\ldots & & & 0 & 0 & I \\ \ldots. & 0 & 0 & \ldots. & A_2' & A_1' \end{bmatrix}$$

$\ldots.(2.49)$

(2.48) and (2.49) make it clear that a controller of the form

$$u(t-1) = \mathbf{F}x'(t-1) \qquad \ldots.(2.50)$$

may be used to modify the parameters of A_1', A_2' ... so that the resulting polynomial matrix has the same parameters as some model polynomial matrix with suitable poles, i.e. we can require

$$x'(t) = [\mathbf{QDQ}^{-1}+\mathbf{QEF}]x'(t-1) = \mathbf{A}_m x'(t-1) \qquad \ldots.(2.51a)$$

which because of the form of **QE** reduces to the following when the last m rows of (2.51a) are chosen:

$$[\ \ldots\ 0\ 0\ \ldots\ A_2'\ A_1'] + \mathbf{F}_c = [\ \ldots\ 0\ 0\ \ldots\ A_{m2}\ A_{m1}]$$

$$\ldots.(2.51b)$$

The m input signals may be calculated from

$$u(t-1) = \mathbf{F}_c x'(t-1) = \mathbf{F}_c \mathbf{Q} x(t-1) \qquad \ldots.(2.52)$$

x(t-1) is defined in equation (2.45) and it is apparent that (2.52) is now of the same form as (2.43), i.e. it defines polynomials **H** and **G** which are required by the pole placing controller. The full pole placing algorithm may now be given.

1. Estimate the parameters of equations (2.44) and (2.45). These parameters are estimated using the same estimation procedure as that described for the previous multivariable algorithms.

2. Form the **Q** matrix using equations (2.46) and (2.47).

3. Transform the **D** matrix as in equation (2.49).

4. Calculate $\mathbf{F}_c$ using equation (2.51).

5. Apply the control defined by equations (2.52) and (2.43).

The full sparse matrices described by equations (2.45) and (2.49) need never be formed in their entirety, but their structure defines a set of operations which must be performed. Two matrix inversions are required, but one at least - that of **J** defined by equation (2.46) - is of a relatively sparse matrix, and not too arduous.

The resulting structure of the matrices of equation (2.49) is of interest, particularly for a scalar process where one of the matrix inversions may be avoided, the algorithm reducing to the same complexity as that described in section 1.4.3. The resulting structure of equation (2.49) is illustrated in Appendix A, where some numerical examples of the pole-placing calculations are also presented.

2.3.2 Set point attainment

The regulator described above will not in general ensure set-point attainment if the set-point is non-zero, and will not remove steady-state offsets or the effects of load changes to unmeasured disturbance variables. It is assumed that the set-points will be held steady between changes, and that offsets and other disturbances are slowly varying. The analysis is further simplified by the fact that the model polynomial matrix is usually a diagonal matrix, so that interactions between loops may be minimised by the regulator. Constant offsets and unmeasured disturbances are incorporated into an offset term for each loop, all the offsets being gathered into a vector d. This vector is

estimated by the Kalman Filter measurement update routine, but an integral-like action may be introduced by a time-update of d between measurement updates. This time update takes the form:

$$d(t|t-1) = \mathbf{W}[y(t)-w(t-k)] + d(t|t) \qquad ....(2.53)$$

$\mathbf{W}$ is a diagonal weighting matrix which determines the "reset" rate of the integral action, and w(t) is the vector of set-points.

Knowledge of the set-points and offsets enables the process control input to be calculated as follows:

$$u_c(t) = u(t) + B^{-1}(1)[A_m(1)w(t) - d] \qquad ....(2.54)$$

where u(t) is the control action required by the pole-placing regulator, and the remaining terms ensure offset elimination and unity gain of the steady-state transfer function relating y(t) and w(t).

Chapter 3
METHOD

3.0 INTRODUCTION

The tangible product of this research is a suite of programs for control of a process using self-tuning methods. Having been written for research purposes, the programs are necessarily flexible and easy to change, although less efficient than a dedicated program would be. A high level language (1966 ANSI Standard FORTRAN) has been used, and all the control routines have been written as subroutines, care being taken to use only "Basic Fortran" statements which would make them as portable as possible from machine to machine. A main program, which calls the control subroutines, is assumed to exist, and to incorporate all the machine-dependent functions such as handling of disk files, timing operations, and all input and output (communication with both process and operator). This makes it possible to merely add these subroutines to many existing control programs for any control loops which require them. In cases where there was no suitable control program in existence, it has been necessary to write one.

Section 3.1 describes the programs which have been written, while the computer systems on which these programs have been implemented are dealt with in section 3.2. Finally section 3.3 introduces the processes which have been controlled in the course of this research.

3.1 PROGRAMS

The programs were originally written for an implicit self-tuning control scheme, and the same structure has been used since for explicit pole placing algorithms, and other DDC schemes. It has thus been found to be very general in its application. The

programs for the explicit case will be used to introduce the structure of the routines, and then it will be shown how this structure may be modified to incorporate other controllers.

3.1.1 Explicit pole-placing algorithms

A main subroutine establishes the COMMON areas for the data structures, and a calling sequence for other subroutines in the program suite. Data areas in the main subroutine have been set aside for the following:

(i) Process variables.

Process variables are stored in a single one-dimensional array, each variable comprising the present value, and several values from previous sample intervals, i.e. a variable x(t) is stored as a sub-array [x(t),x(t-1),x(t-2)...]. The position of each sub-array within the main array is identified by a variable pointer. Current variable values, to be passed between the control subroutines and the main program, are extracted and held in separate arrays, which may be integer arrays in anticipation of their use with A/D and D/A circuitry. Variable pointers are held in a separate integer array.

(ii) Model parameters.

Up to n model parameters for each of m loops are held in an n X m array. The covariance matrix for the identification routine is decomposed using the UDU' factorisation, and for each loop the U and D covariances are stored as a single one dimensional array, the overall array for m loops having m columns. Arrays are also required for the model polynomials for each loop, which establish the pole positions, and the observer polynomials which operate on the set points.

(iii) Parameters which control program execution.

(iv) Work area.

This area is set aside for operations associated with matrix inversion for pole placement.

All other variables, such as loop counters and temporary working variables, are local to the subroutine in which they are used.

Functional overview of programs

A main subroutine exercises overall control, and a logic flow description of this subroutine is given below.

```
Subroutine EXSTC
If this is the first call to EXSTC then
     begin
          Read the following parameters from disk file:
          - counts for certain operations
          - model and observer polynomials
          - variable pointers (determining process structure)
          initialise variables (e.g. covariances)
     end
else
     begin
          establish set-points (predetermined sequences, or
                                entered by operator)
          for each loop
               shift all variables (time update), and enter
               new variable values (from calling program)
          for each loop
               perform UDU' measurement update parameter estimation
          for each loop
```

```
                perform such control calculations as can be done
                for loops separately
            complete control calculations for multivariable process
            prepare variable arrays for return to calling program
        end
end.
```

Description of subroutines to perform above functions

As described above, each subroutine is called repeatedly for each of the process loops. At each call, the pointers to the parameters and variables are changed to accord with the particular loop under consideration. The more important subroutines are listed below, with a brief description of the function of each one.

(i) Miscellaneous functions.

Certain standard operations used by many of the other subroutines are defined as subroutine or function calls. These operations include polynomial evaluation and multiplication, and the movement of polynomial values from one polynomial to another. The movement of values is important because it is used in the time updating of loop variables. Rather than change the numerous pointers, the entire variable space is moved with respect to the pointers. For time critical applications, when using a Z-80 based microcomputer, the operation may be performed in a small assembler subroutine which makes use of the Z-80 block transfer machine code instruction. Apart from time updating, block transfers are used to exchange pointer arrays for different loops, and parameter arrays for the identification and control routines.

(ii) Establishing set points.

Various routines have been used, the choice of each depending on the application of the controller at the particular time. For experimental purposes, the set points may be changed according to some predetermined rule (e.g. a square wave, random numbers), or they may be read from a sequential disk file at appropriate time instants. In other situations, the set points may be changed in response to a demand from the process operator.

(iii) Parameter estimation.

Parameter estimation routines establish the correct vectors and covariance matrices. The parameters are estimated using a measurement update routine based on a UDU' factorisation (Bierman, 1976). "Covariance windup" is avoided by using a variable forgetting factor [Fortesque et al, 1978]. The actual formula used was:

$$\text{FORGET} = 1 - A*PHI*PHI/\text{Mean}(PHI*PHI) \qquad \ldots.(3.1)$$

A is a constant parameter, and PHI is the Kalman filter innovation. In an actual operating situation, the trace of the covariance matrix is checked from time to time to ensure that it does not become too large, and if necessary the matrix is scaled to reduce the trace to an acceptable value.

(iv) Controller parameter calculation.

The equation:

$$\mathbf{AH} + q^{-1}\mathbf{BG} = \mathbf{A}_m \qquad \ldots.(3.2)$$

has to be solved for the parameters of the **H** and **G** polynomials. This equation is re-expressed as a set of simultaneous equations giving:

$$\begin{bmatrix} 1 & 0 & \cdot & 0 & 0 & \cdot \\ a_1 & 1 & \cdot & b_0 & 0 & \cdot \\ a_2 & a_1 & \cdot & b_1 & b_0 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} h_0 \\ h_1 \\ \cdot \\ g_0 \\ \cdot \end{bmatrix} = \begin{bmatrix} a_{m0} \\ a_{m1} \\ \cdot \\ \cdot \\ \cdot \end{bmatrix} \quad \ldots.(3.3)$$

These equations are solved by triangularisation of the parameter matrix. The resulting parameters are then used by the control routine to establish the control input for the loop in question using the methods described in Section 2.1.

(v) Display and return of values to the main program.

The display and recording of process behaviour is ideally performed in the main program, but it is possible to incorporate a display routine as part of the self-tuning controller, particularly if display of model parameter variation is desired so that convergence may be examined. Usually, a simplified display is produced during a run, and a detailed analysis is made possible by producing graphs off-line from a disk file where the loop variables and parameters have been logged. Any variables that have to be returned to the main program have to be scaled and placed in the appropriate data areas at this point.

Offline parameter entry

The self-tuning controller still requires the prior choice of a number of parameters which establish the bounds of its operation. These parameters are stored in a disk file for use by the control

routines. This disk file may be set up using any standard text editor program, but it is more convenient to follow a question and answer process to create the required file. This interactive procedure relies on an offline program. Figure 3.1 shows an example of the dialogue followed by this routine.

```
NO. LOOPS, NO. STEPS, STEP LOOP CLOSED :
INITIAL R MATRIX (DIAG) :

LOOP 1
NO. DISTURBS,MODELORDER,PROCESSORDER,DELAY :
<Y> = INTEGRAL, ELSE <RET> :
 NA NB NC ND NG NH
 2  1  2  1  1  2 ..RETYPE ELSE <RET> :
OBSERVER POLYNOMIAL PARAMETERS :
MODEL PARAMETERS :
POINTERS TO Y,U,W,ZE,ZA ARE 1  11  21  31  41
FEEDFORWARD POINTERS START AT 101
ENTER POINTERS FOR 2 DISTURBANCES :
NO. STEPS BETWEEN SAMPLES :

LOOP 2
etc.
```

FIGURE 3.1 Offline Dialogue to define controller
(N.B. The <:> is a prompt for operator input)

3.1.2 Programs for Implicit Self-tuning.

The programs for the implicit self-tuning controller use exactly the same structure as those described in section 3.1.1, and many

of the routines are the same. The main changes are in the re-definition of the data sets, and the determination of the controller parameters and the control signal itself. Rather than describe the entire set of routines, this section will highlight the differences from the pole-placing routines of section 3.1.1.

Data storage areas

The polynomials **P**,**Q**,**R** and **S** which define the cost function weightings (equation 2.36) have to be defined, in place of the model polynomial $\mathbf{A}_m$ and the observer polynomial **E** required for pole placement. All other data areas are unchanged, although the parameters to be estimated will be those used for cost function prediction, rather than the parameters of the process model.

Program logic flow description

The main subroutine is identical to that used for explicit self-tuning, and which is described in section 3.1.1. Only the data areas have to be redefined.

The parameter estimation routines require different vectors for parameters and variables. The UDU' measurement update routine estimates the parameters of **THETA**, where:

$$\mathbf{P}y(t) = \mathbf{THETA}^*\mathbf{X}$$

$$\mathbf{X} = [u(t-k),\ldots,y(t-k),\ldots,\phi_y(t-1),\ldots,v(t-k),\ldots]'$$

Following the estimation of the parameters, the parameters of the weighting polynomials **P**,**Q**,**R** and **S** have to be adjusted as described in section 2.2. An additional subroutine is defined to

perform this adjusting operation.

The controller subroutine(s) are simpler for this implicit algorithm than for the explicit pole-placing algorithm. The control subroutine performs the operation defined by equation (2.41), calculating u(t).

The remaining operations for return of the correct values to the calling program, and for display and data logging, are the same for both implicit and explicit controllers.

The same structure has formed the basis for programs used for identification and control with other algorithms. Any unnecessary subroutines are omitted, and those defining key functions are changed. The common basic structure means that the programs have met the requirement of flexibility required for research purposes.

3.2 COMPUTER SYSTEMS

During the course of the research, the programs were run on two different computer systems. The simulations were performed on a PDP11/34 minicomputer, and on-line laboratory experiments used a Cromemco microcomputer. The use of a microcomputer is of interest as it emphasises the suitability of these low cost devices for such control.

3.2.1 MINICOMPUTER IMPLEMENTATION

The hardware comprised a PDP 11/34 minicomputer with 64K words of memory, and RL01 disk drives. The EIS (Extended Instruction Set)

was available, but floating point operations were implemented in software. A Tektronix 4010 terminal with a storage screen was available for graphics output, and the graphs were recorded on a hard-copy unit associated with this terminal.

The operating system is RSX 11M, and programs were written in standard FORTRAN. ISIS, a simulation language for continuous systems, was used for some simulations.

3.2.2 MICROCOMPUTER IMPLEMENTATION

Hardware

The arrangement of the microcomputer hardware is shown in Figure 3.2. The microcomputer was a Cromemco System 2, based on a Z-80 microprocessor, with 48K bytes of random access memory, two 5.25 inch diskette drives, and a standard visual display unit without graphics. A low cost dot matrix printer was attached to the system, and graphical output was performed offline onto a pen recorder using D/A outputs.

Connection to the process used a Cromemco interface board which provides:

- 7 analog inputs (8 bits, +-2.54 volts)
- 7 analog output (8 bits, +-2.54 volts)
- 8 bit digital input
- 8 bit digital output

The analog outputs were converted to current loop signals for transmission to the process. Instrumentation amplifiers and low-pass analog filters were used on the low level signals from the process (e.g. temperature measurements using copper-constantan

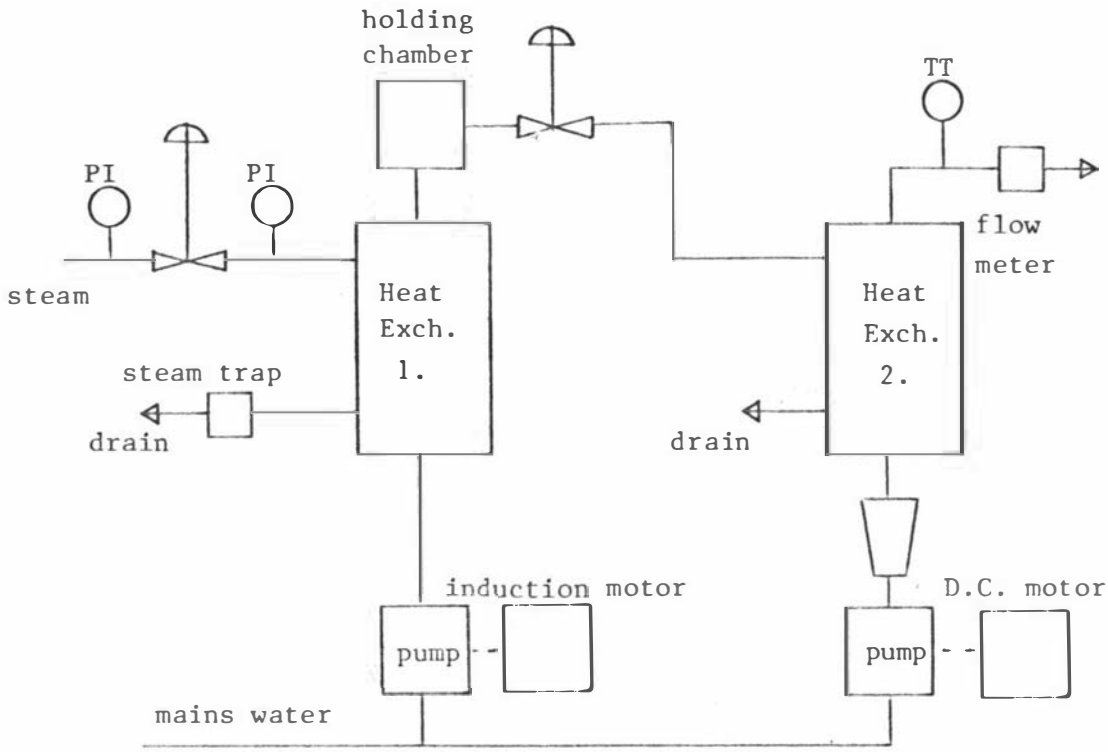


FIGURE 3.4 Heat Exchanger System

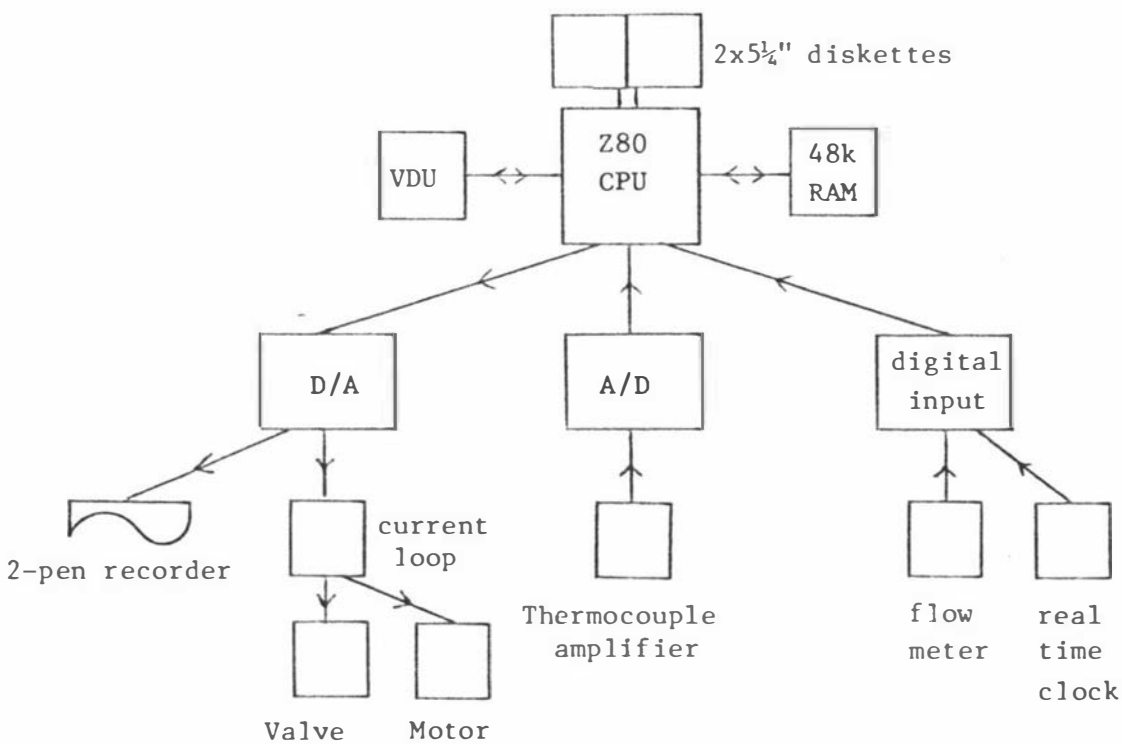


FIGURE 3.2 Microcomputer System

thermocouples).

The flow rate measurement produced digital impulses of varying frequency. These impulses were sensed directly on one bit of the digital input port, and counted during a fixed time interval. Each reading returned to the process was the average of five repeated measures.

Real time operation relied on impulses from a variable frequency oscillator, which were sensed on another bit of the digital port.

All the control loops incorporated smooth nonlinearities, and no attempt was made to linearise these.

Software

The software relied on the CDOS operating system [Cromemco, 1978], which is a superset of the CP/M operating system [Digital Research, 1978] and which provides ANSI Standard FORTRAN X3.9-1966.

Processing Requirements

The number of control loops which can be implemented on a single CPU is limited by the time taken to process each loop, and the storage requirements. The processing time for the self-tuning controller was 100-400 milliseconds/loop depending on order. Storage requirements for programs and data areas are given below in Figure 3.3.

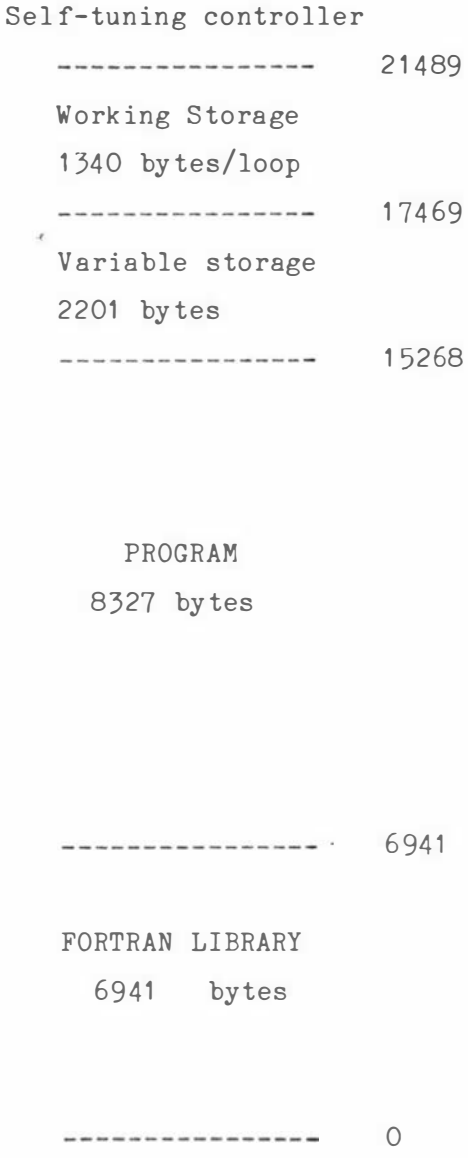


Fig. 3.3 Memory Requirements

3.3 PROCESSES CONTROLLED

For simulation purposes, the ISIS simulation language was made to represent the process to be controlled. For some simulations, ISIS was replaced by a discrete simulation program, written especially for this purpose. The microcomputer implementation of

the controller was tested in the laboratory using a commercial heat exchanger.

3.3.1 Simulations

The manner in which the control subroutines are written permits any other routine to represent itself as the process to be controlled. The use of ISIS [1980], a continuous simulation language, offers the advantages that the underlying continuous nature of many processes may be accurately simulated, and that the full power of a carefully developed simulation language may be used to investigate the behaviour of the processes being controlled. Processes may be endowed with non-linearities or time varying parameters, and ISIS provides good graphical facilities.

RSX 11/M provides intertask communication directives. These directives have been used to create a hand-shaking protocol for passing parameters between tasks. In this case one of the tasks is ISIS, simulating the process to be controlled. The second task comprises a small program which calls the control subroutines described earlier. ISIS is an interactive language, providing FORTRAN-like facilities in addition to the simulation language functions. An additional language function has to be defined, written in FORTRAN (the host language) and linked into the main ISIS task for the purpose of communicating with the control task. ISIS and the controller task execute concurrently.

For discrete simulations, the program which communicates with ISIS is replaced by a simple discrete simulation program. In this case, the display and reporting functions provided in the control subroutines are used. During a simulation run, model parameters and process variables are stored in a disk file for later examination and graphing.

3.3.2 Heat exchanger

A heat exchanger combination was used to provide a simple yet representative example of a real process. This process is very similar to that used by Davison [1980] to evaluate the performance of multivariable tuning methods. The choice of this process was motivated by the use at the New Zealand Dairy Research Institute of a similar heat exchanger to control temperature and viscosity of milk concentrate being passed from an evaporator to a spray drier.

Figure 3.4 is a schematic diagram of the heat exchanger combination which is to be controlled. The purpose of the control is to produce an output flow of water from heat exchanger 2, at a prescribed flow rate and temperature. The 4-20 ma current loop control signals regulate a pneumatically controlled steam valve to manipulate the steam flow to heat exchanger 1, and a thyristor controller driving a D.C. motor and pump which varies the flowrate of the output water from heat exchanger 2.

Heat exchanger 1 is a shell-and-tube heat exchanger. A thermodynamic steam trap is installed on the condensate line from the exchanger. Mains water is delivered to the exchanger by a pump driven by an induction motor and once heated this water passes through a holding chamber and a valve which controls the flow to heat exchanger 2, which is a parallel plate exchanger. After passing through heat exchanger 2, the heating water is delivered to the drain. Mains supply water to be heated is delivered to heat exchanger 2 by a D.C. motor controlled pump, whose speed is regulated by a thyristor control unit. Measurements are made of the outlet flow rate and temperature of the outlet water.

A water removal system ensures that the steam is saturated, although there is reason to doubt the efficiency of its

operation. The hold up vessel and pipe lengths produce time delays in both loops of the system. There is no linearisation of the effect of the control signals, and the fact that the steam process is a total condensation system implies that the dynamic behaviour of the steam process will be nonlinear. Hysteresis of the steam control valve is a major source of non-linearity. The differential equations describing the temperature loop will be "stiff" because of the diffusion effects. Considerable perturbations are introduced by periodic operation of the steam trap, and there are other minor sources of non-linearity or disturbance.

Normal Operating Conditions

Steam supply pressure	200 kPa
Output water flow from heat exchanger 1	15 litres/minute
Output water flow from heat exchanger 2	0.5-5 litres/minute
Temperature of cold water supply	~20 deg.C.
Output water temperature, heat exchanger 2	20-70 deg.C.

The system is perturbed from its normal operating condition by altering the output water flow rate from heat exchanger 1 to 10 litres/minute. This operation has the effect of altering both the time delay, and the parameters of the process model, and provides the facility for testing the controllers' capacity to deal with process changes.

Chapter 4
RESULTS

4.0 INTRODUCTION

Programs have been written which implement three different controller schemes, as developed in Chapter 2. These controllers will be referred to as follows:

#1. Implicit algorithm. This is described in section 2.1.

#2. Explicit pole placing algorithm. This is described in section 2.2. This includes the assumption that $\mathbf{B}_0$ and $\mathbf{H}_0$ are diagonal matrices.

#3. Explicit pole placing algorithm. This is described in section 2.2 and removes the assumption of #2 above that $\mathbf{B}_0$ and $\mathbf{H}_0$ are diagonal matrices.

This chapter presents simulations which establish the behaviour of particularly controller #3, both for single loop and multivariable systems, and for processes with measured disturbance terms, the effects of which must be eliminated by the control action. The differences between controllers #2 and #3 for multivariable systems are demonstrated. Some results are presented for controller #1, other results for similar controllers being available elsewhere. Suitable strategies for specifying the desired controller behaviour will become apparent. Finally, the application of these controllers to a commercial heat exchanger will be described.

4.1 SIMULATIONS WITH IMPLICIT SELF-TUNING CONTROLLER

The implicit self-tuning controller implements the control scheme of Clarke and Gawthrop [1979], incorporating approximations suggested by Hesketh [1980] to extend the controller to deal with multivariable processes, by decoupling the loops and ensuring steady state set-point attainment.

The controller has been described in Section 2.2. The following simulations shown in Figures 4.1 and 4.2 illustrate the effects of interactions, disturbances and set point errors.

Disturbance rejection is illustrated by simulating the scalar process:

$$\mathbf{A}y(t) = q^{-1}\mathbf{B}u(t) + q^{-1}\mathbf{D}v(t) + \mathbf{C}e(t)$$

with:

$$\mathbf{A} = 1-0.5q^{-1} \quad \mathbf{B} = 1+1.5q^{-1} \quad \mathbf{C} = 1-0.2q^{-1} \quad \mathbf{D} = 0.3+0.5q^{-1}$$

Set point w(t) and disturbance v(t) are varied as shown in Figure 4.1 e(t) is a uniformly distributed random noise sequence on the interval (-0.05, 0.05). The adjustments to **R** and **E** for set point error and to **S** for disturbance rejection are not made until step 200. Consequently, a comparison may be made between the behaviour of the system with and without these adjustments. The initial controller parameters are chosen as:

$$\mathbf{P} = 0.4 \quad \mathbf{Q} = 0.6 \quad \mathbf{R} = 0.4 \quad \mathbf{S} = 0.1$$

Following step 200 there is some convergence to new parameter values as a result of the adjustment to **S**, and then the benefits of the disturbance rejection become apparent, as evidenced by the changes to the control input u(t) in response to changes to the disturbance v(t).

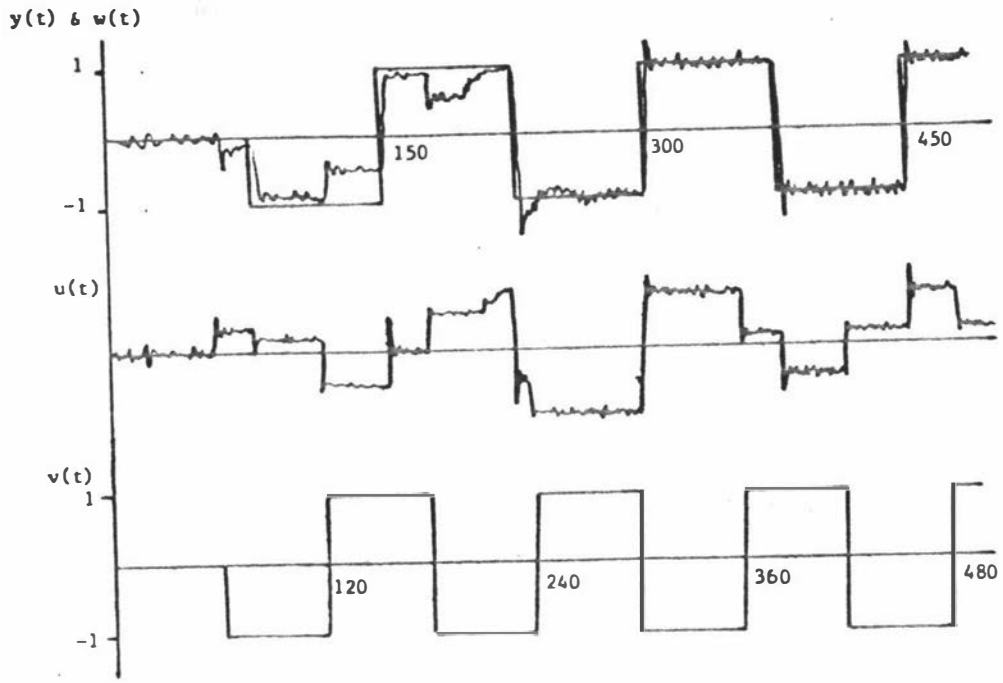


FIGURE 4.1

Implicit Controller

Disturbance effects and set-point attainment

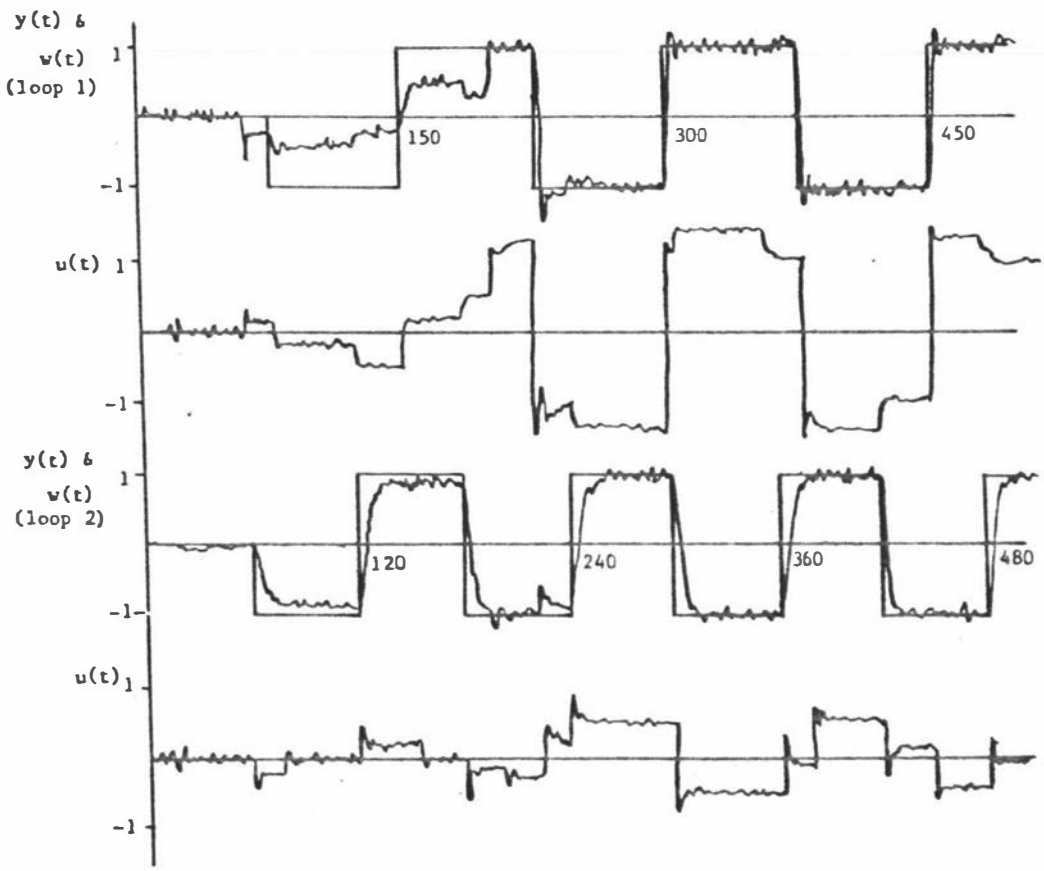
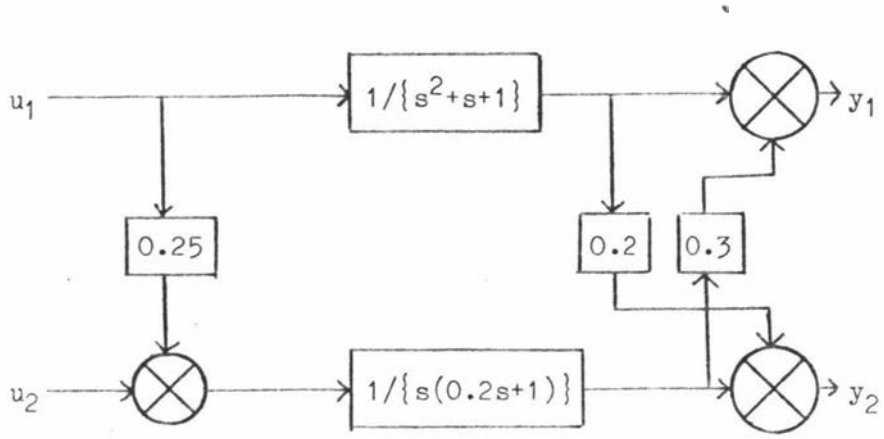


FIGURE 4.2

Implicit Controller

Interaction effects and set-point attainment

The disturbance rejection illustrated above may be used in the case of a multivariable process to effectively decouple the two loops. The following simulation will demonstrate this. The process to be simulated may represented in block-diagram form:



For the purpose of modelling the process shown in this block diagram, u_1 and y_1 are treated as disturbances to loop 2, and y_2 is treated as a disturbance to loop 1. The controller parameters for both loops are chosen as:

P=0.3 **Q**=0.7 **R**=0.3 **S1**=0.1 **S2**=0.1

Elimination of set-point error is ensured by adjustments to **R** and **E**, while adjustments to **S1** and **S2** are used for removing the effects of interactions between the loops. To enable comparison between the performances of the controller with and without these adjustments, no adjustment is made until step 200. The results of this simulation are presented in Figure 4.2. It is apparent that neither loop will attain the correct set-point value unless the adjustments to the polynomials are made, in the absence of any integral behaviour. The interaction between loops (particularly the effect of loop 2 on loop 1) is also apparent.

At time step 200, following the initial adjustment of the

polynomials, there is rapid convergence resulting in good set-point following from that point on. The removal of the interaction effects is apparent from the changes in the control inputs to the two loops.

An alternative way of ensuring set-point attainment is to include a factor $(1-q^{-1})$ in the **Q** polynomial. However, incorporation of an integrator in this way, or by operating on incremental control variables rather than their absolute values degrades the transient performance of the controller [Morris, Fenton and Nazer, 1977].

4.2 SIMULATIONS USING EXLICIT POLE PLACING CONTROLLER

In Chapter 3 it is described how an interactive offline program is used to establish a disk file defining the parameters of the controller, and the parameters which describe certain aspects of the process which are to be assumed by the controller. The specification of the controller for each process may be completely defined by describing the parameters which are placed in the disk file. Apart from the counters and pointers which establish the significant events during the run and the variable combinations for each loop, the following are the critical parameters which have to be specified for the controller:

1. Initial error variances. These are set to an arbitrary large value for the start of the run (for rapid initial convergence in the absence of good initial parameter estimates), and adjusted by the Kalman Filter routine to appropriate values during the run. Different initial values could be selected to enable investigation of the effect of different initial values.

2. The order of each polynomial for each loop. Default values are suggested, based on process order.

3. The maximum delay for each loop.

4. Whether the controller is to have integral properties or not.

5. Observer polynomial ($\mathbf{A}_0$) and model polynomial ($\mathbf{A}_m$) for each loop. $\mathbf{A}_m$ defines the pole placements.

6. Sampling rate for each loop.

4.2.1 Single loop processes

Kurz, Isermann and Schumann (1980) described several single loop processes which they used for comparison of various parameter adaptive control schemes. Some of these processes have been simulated using either the continuous simulation language ISIS, or a discrete simulation program, the control inputs to the processes being determined by the self-tuning controller running as a concurrent task.

The output of the process, which is the controlled variable, is the sum of the actual process output, and the output of a noise model. This accords with the process model used by Astrom (1970, page 173). The noise model used for all processes is:

$$E(s)/V(s) = (s^2+1)/(s^2+s+1) \qquad \ldots.(4.1)$$

V(s) is a sequence of steps the amplitude of each being determined by a random number generator, and changed at each sampling interval. $E\{e(t)e^T(t)\}=0.0133$.

In addition to the noise, some of the processes were subjected to an offset. This is indicated where appropriate. OFF=0.3.

Simulation summary

[A] Low pass behaviour, non-minimum phase (with 2 sec. sampling).

Transfer function	$1/(1+2.5s)(1+3.75s)(1+5s)$
Simulation model	$Y'''=0.0213(U+OFF-Y)-0.24Y'$ $-0.8667Y''$
Controller	$A_m=1-0.8q^{-1}$
	$A_0=1$
	Sampling interval 2 secs
	Delay k=1

[B] Low pass behaviour

Transfer function	$1/(1+2.5s)(1+3.75s)$
Simulation model	$Y''=0.1067(U+OFF-Y)-0.6667Y'$
Controller	$A_m=1-0.8q^{-1}$
	$A_0=1$
	Sampling interval 2 secs
	Delay k=1

[C] Damped oscillating behaviour

Transfer function	$(1+2s)/(1+3s)(25s^2+5s+1)$
Simulation model	E'''=0.01333(U+OFF-E)-0.10667E' -0.53333E''
	Y=E+2E'
Controller	(a) & (c) $A_m=1-0.8q^{-1}$
	(b) $A_m=1-0.9q^{-1}$
	Sampling interval (a) & (b) 3 secs
	Sampling interval (c) 6 secs
	Delay k=1

[D] Process [C] with time delay

Transfer function	$e^{-9s}G_c(s)$
Simulation model	As before with U=DELAY(9,U1)
Controller	$A_m=1-0.8q^{-1}$
	$A_O=1+q^{-1}+q^{-2}$
	Sampling interval 3 secs
	Delay k=4

[E] All pass behaviour (discrete model).

Transfer function	(1-4s)/(1+4s)(1+10s)
Simulation model	E''=0.025(U+OFF-E)-0.35E'
	Y=E-4E'
Controller	$\mathbf{A}_m=1-0.8q^{-1}$
	$\mathbf{A}_0=1$
	Sampling interval 2 secs
	Delay k=1

[F] Integral behaviour

Transfer function	1/s(1+5s)
Simulation model	Y''=0.2(U-Y')
Controller	$\mathbf{A}_m=1-0.8q^{-1}$
	$\mathbf{A}_0=1$
	Sampling interval 1 sec
	Delay k=1
Noise	$E\{e(t)e^T(t)\}=0.0033$

[G] Unstable behaviour

Transfer function	$1/(1+5s)(1-2s)$
Simulation model	$Y''=0.1(E-U)+0.3E'$
Controller	$A_m=1-0.8q^{-1}$
	$A_0=1$
	Sampling interval 1 sec
	Delay k=1

The same first order model polynomial is used for all processes, which consequently display the required exponential rise following a set-point change. Where the process order is greater than 1, zeros are appended to the polynomial to obtain enough simultaneous equations to solve for all variables. For these simulations, A_0 was not varied to obtain any other responses.

The results of these simulations are presented in Figs. 4.3-4.8 for processes [A] to [F] respectively. Some significant points are worthy of comment.

Processes [A] and [B] present no difficulty to the controller, which is designed to handle non-minimum phase systems. Process [C], at the sampling rate of 3 secs selected by Kurz, Isermann and Schumann [1980], requires a rather vigorous controller action, which is not alleviated by changing the model polynomial. However, changing the sampling rate to 6 secs has a marked effect with very little controller action necessary. Retaining the 3 sec sampling rate, and introducing a pure time delay [D] has no appreciable effect on the controlled process, apart from increasing the variability as expected [Astrom, 1970]. Process [D] is chosen to illustrate the parameter convergence, as shown in Figure 4.6 a-e. Process [E] with all-pass behaviour is readily

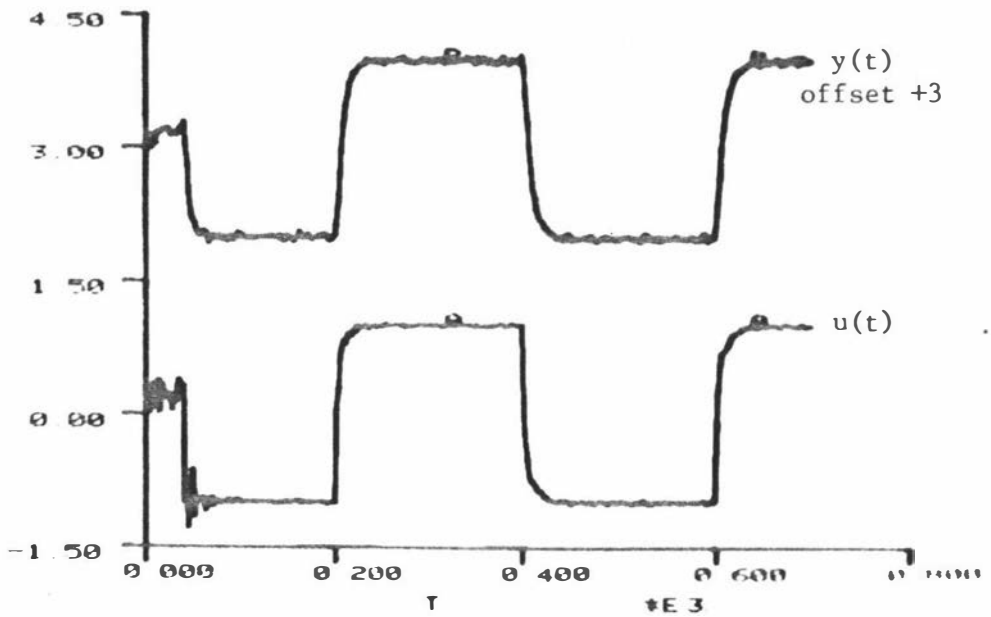


FIGURE 4.3

Process (A) Low Pass Non-Minimum Phase Process

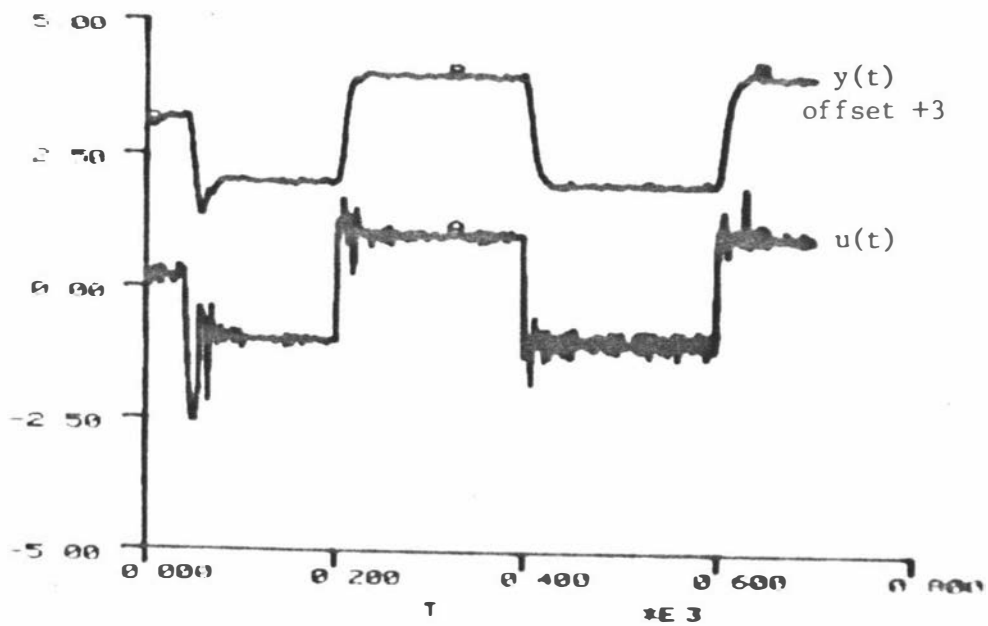


FIGURE 4.4

Process (B) Lowpass process.

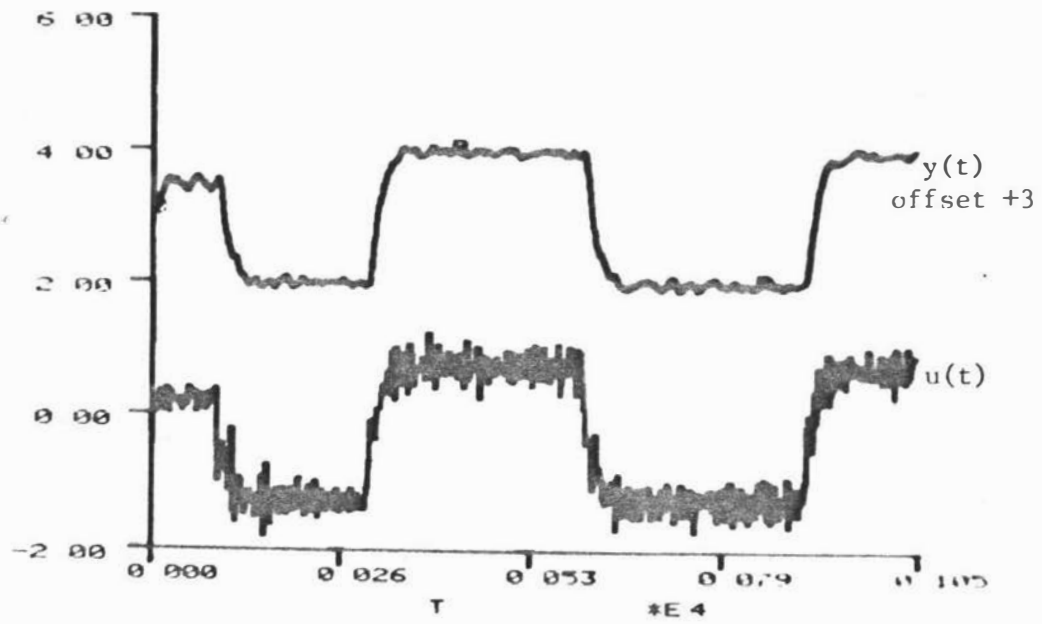


FIGURE 4.5(a)

Process (C) Underdamped process

3 second sampling interval, $A_m = 1-0.8q^{-1}$

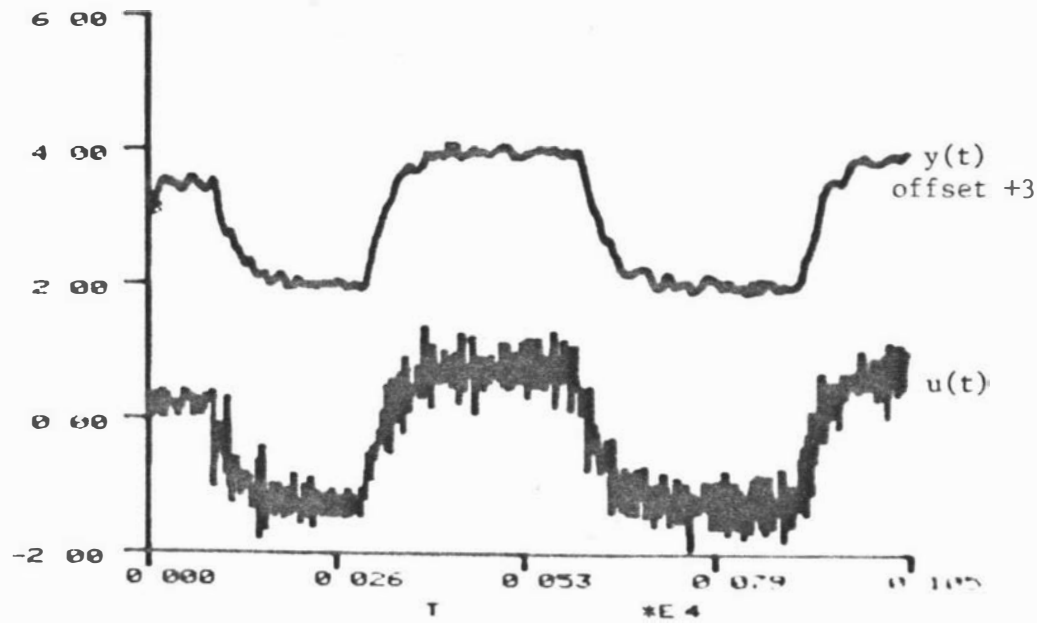


FIGURE 4.5 (b)

Process (C) Underdamped process

3 second sampling interval, $A_m = 1-0.9q^{-1}$

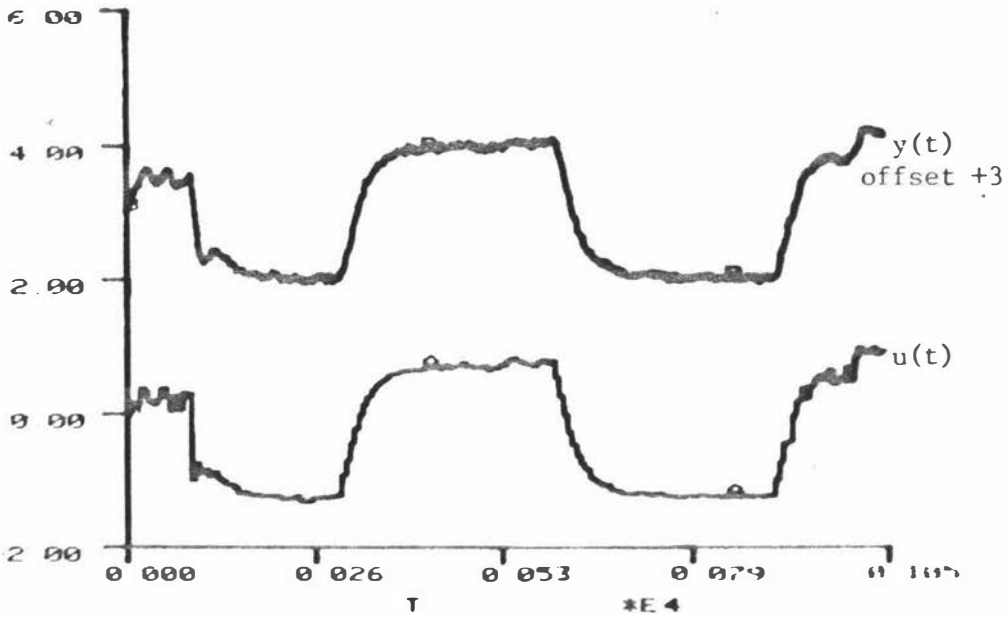


FIGURE 4.5 (c)

Process (C) Underdamped process
6 second sampling interval, $A_m = 1 - 0.8q^{-1}$

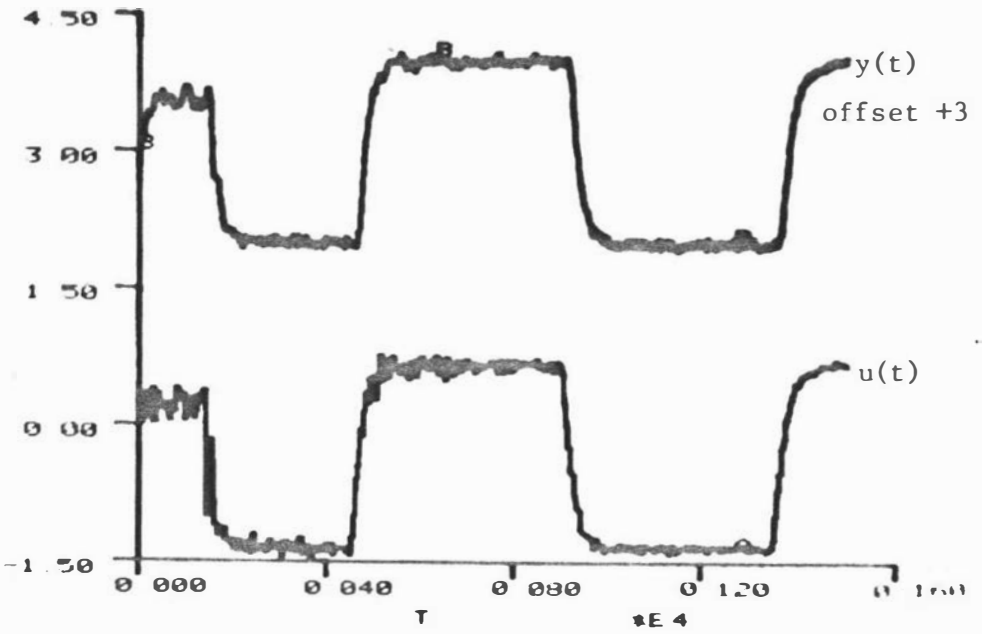


FIGURE 4.6 (a)

Process (D) Underdamped process with time delay

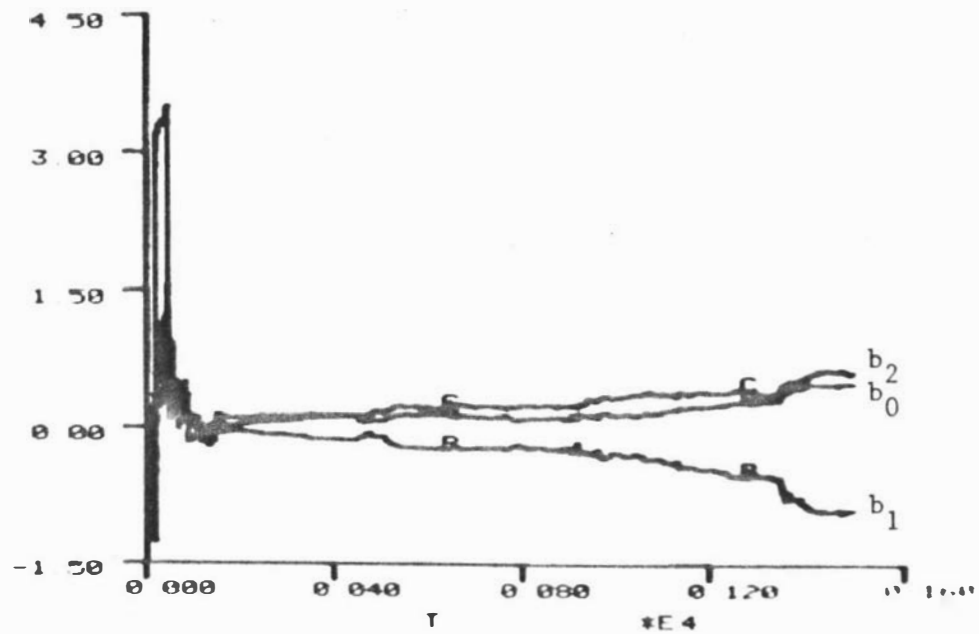


FIGURE 4.6 (b)

Process (D) Estimated parameters.

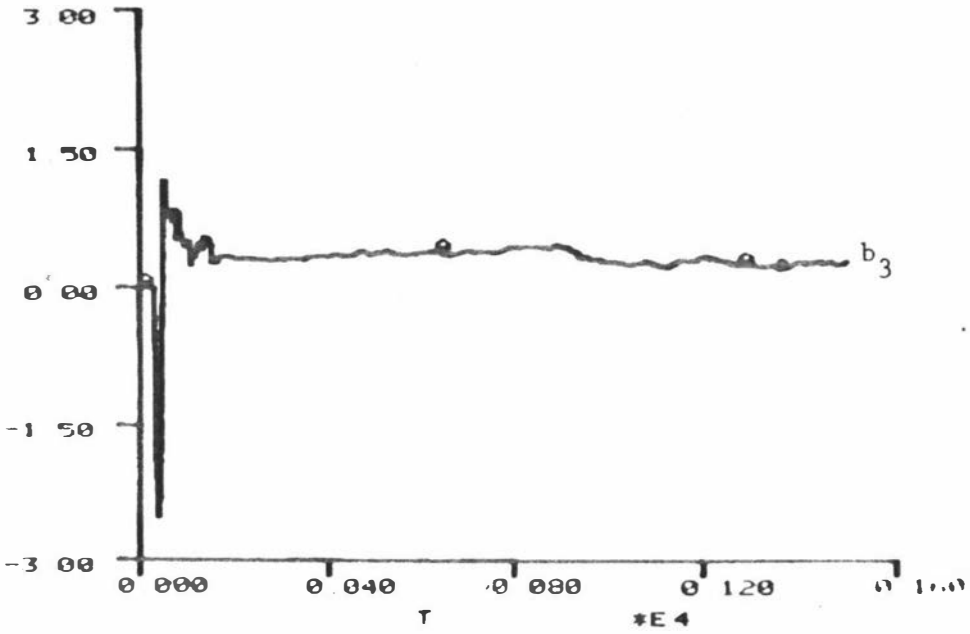


FIGURE 4.6 (c)

Process (D) Estimated parameters

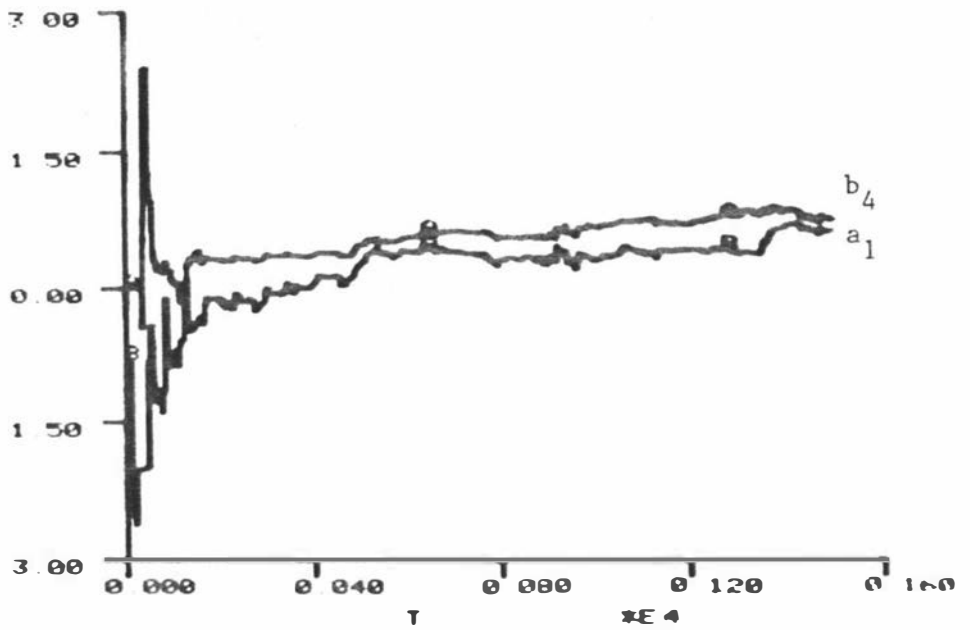


FIGURE 4.6 (d)

Process (D) Estimated parameters

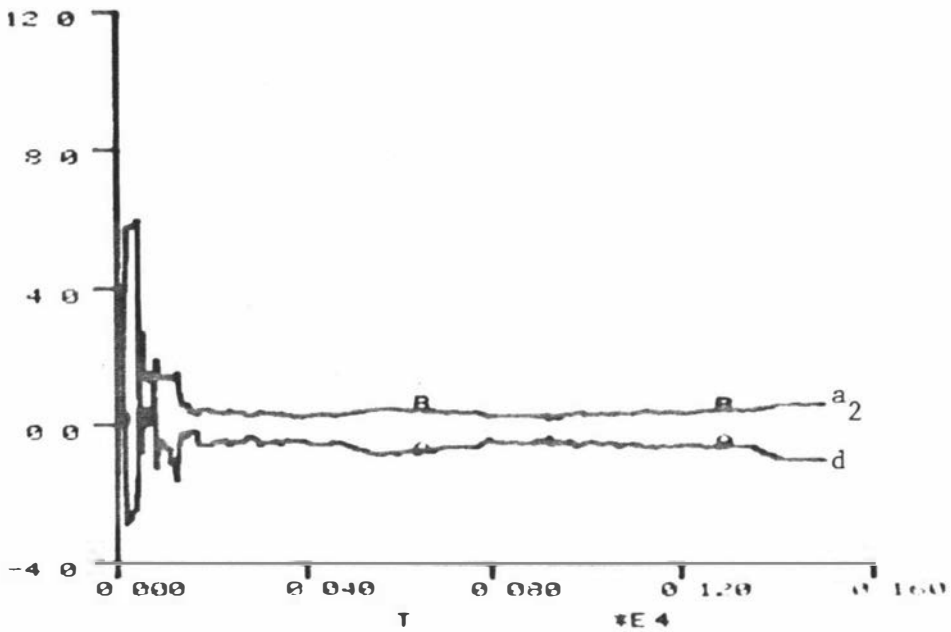


FIGURE 4.6 (e)

Process (D) Estimated parameters

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-3	9

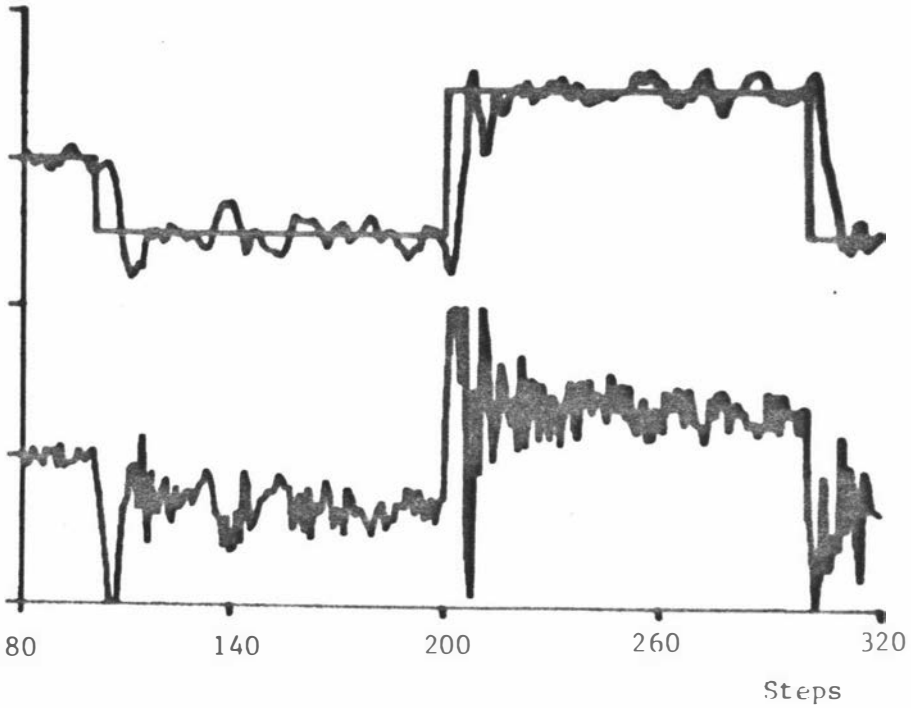


FIGURE 4.7 (a)

Process (E) All-pass behaviour
Sampling interval 3 seconds, $A_m = 1-0.8q^{-1}$

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-3	9

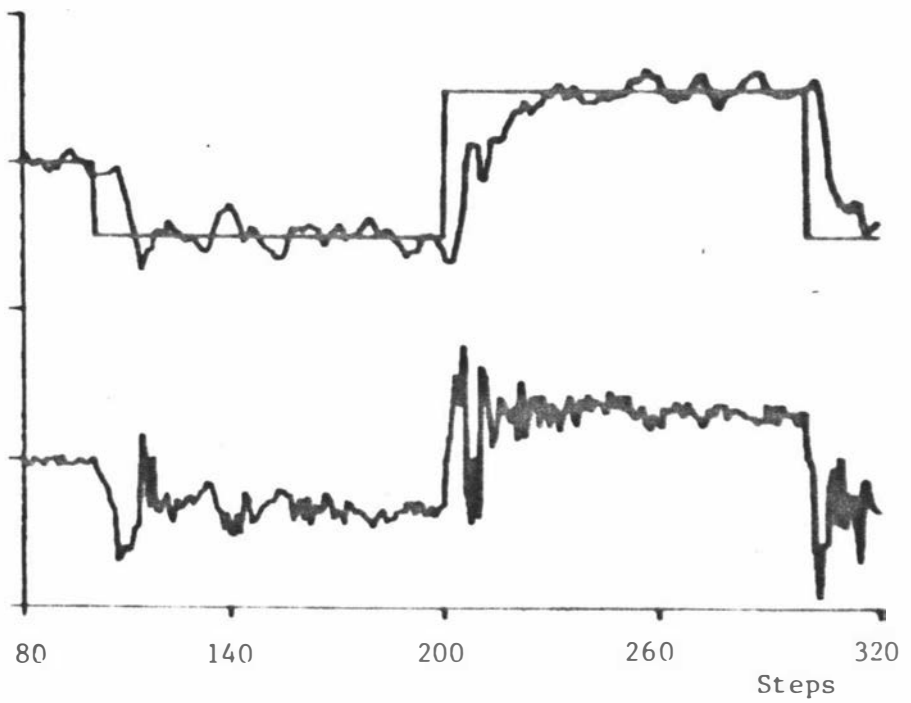


FIGURE 4.7 (b)

Process (E) All-pass behaviour
Sampling interval 3 seconds, $A_m = 1-0.9q^{-1}$

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-3	9

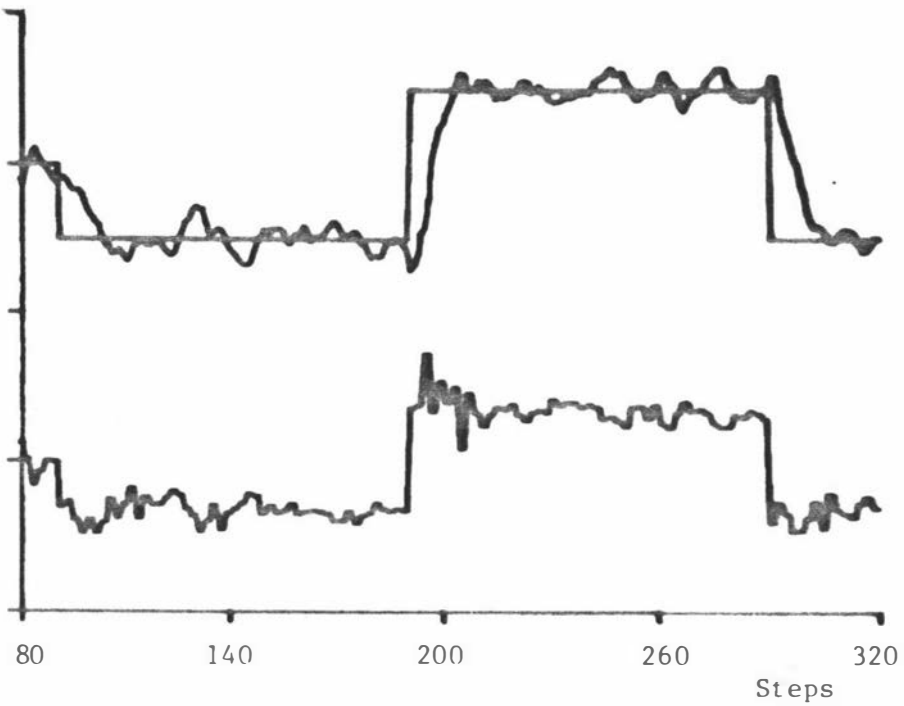


FIGURE 4.7(c)

Process (C) All-pass behaviour
Sampling interval 6 seconds, $A_m = 1 - 0.64q^{-1}$

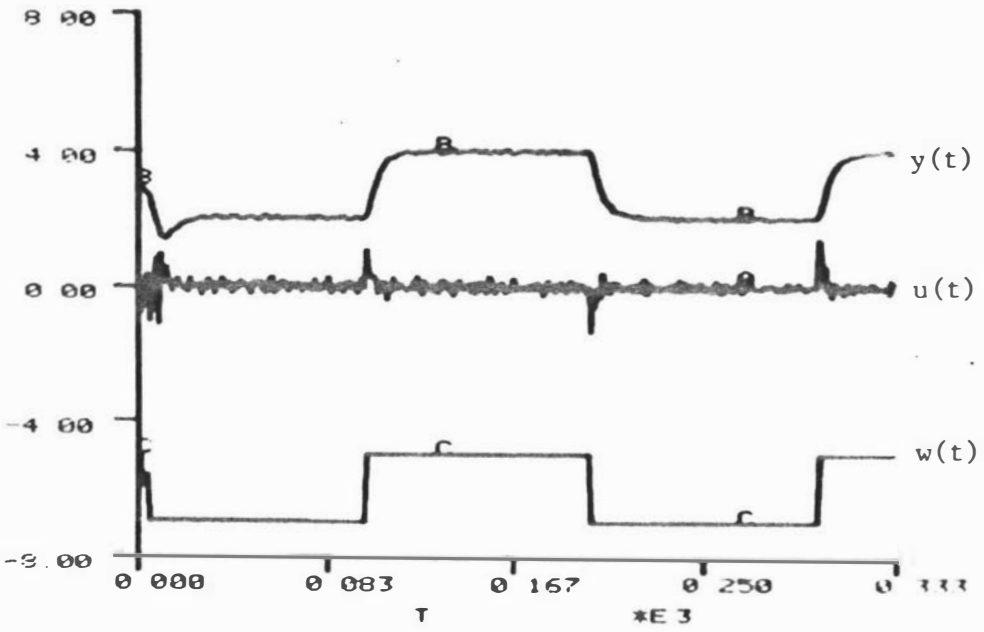


FIGURE 4.8(a)

Process (F) Integral behaviour

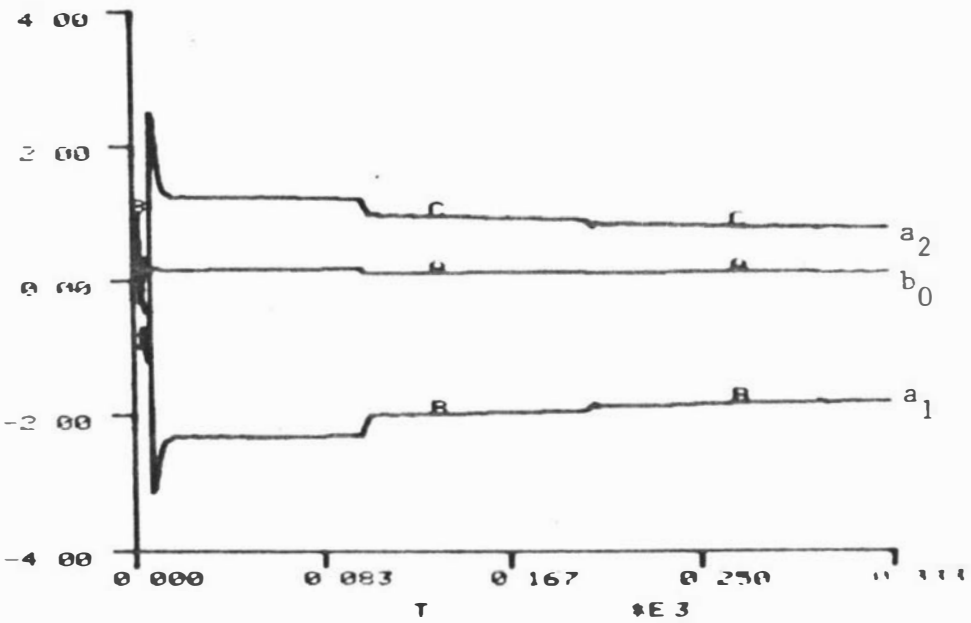


FIGURE 4.8 (b)

Process (F) Integral behaviour
Parameters of $(1+a_1q^{-1}+a_2q^{-2})y(t) = q^{-1}(b_0+b_1q^{-1})u(t)$

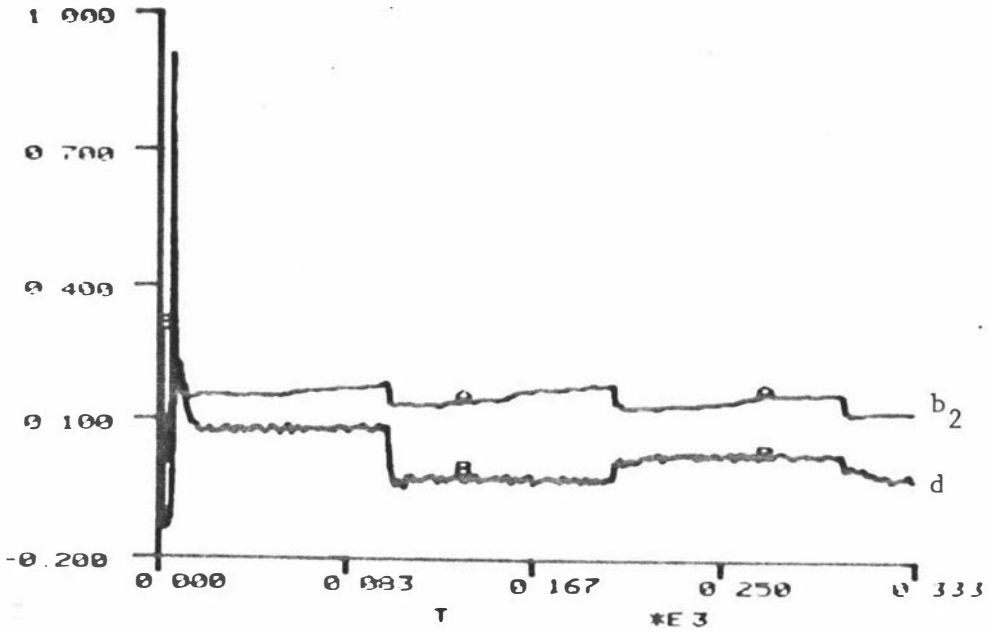


FIGURE 4.8(c)

Process (F) Parameters of model:

$$(1+a_1q^{-1}+a_2q^{-2})y(t) = q^{-1}(b_0+b_1q^{-1})u(t) + d + e(t)$$

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-3	7

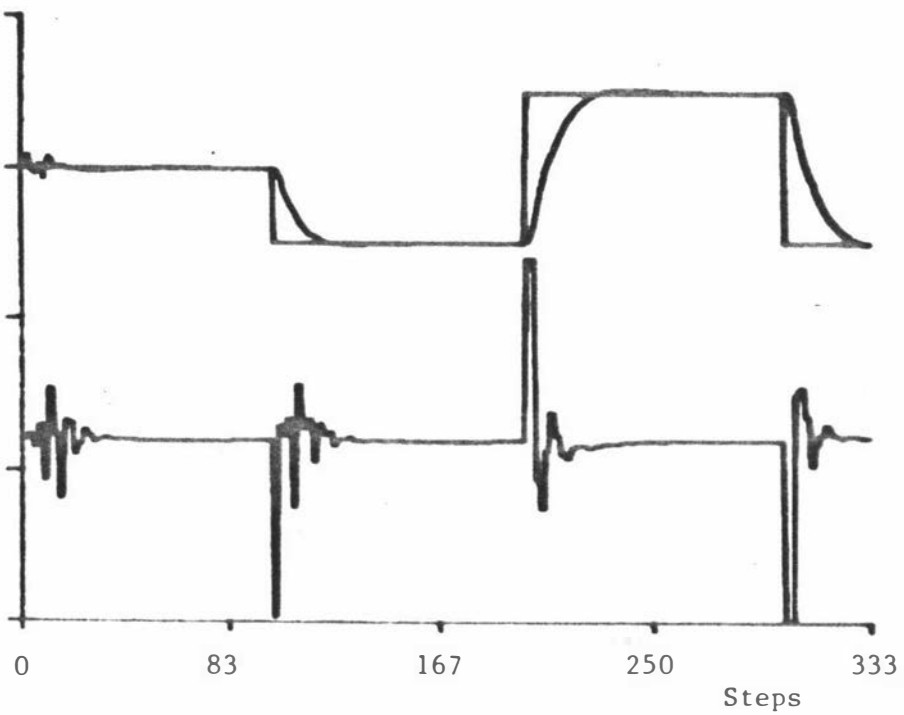


FIGURE 4.8 (d)

Process (F) No noise, saturating control signal

controlled, and again the beneficial effects of increasing the sampling time can be demonstrated. In this case, changing the model polynomial will also reduce the variations of the control signal, because of this process's unusual response to sudden changes in the control signal.

Processes with integral behaviour, like [F], and with unstable characteristics like [G] pose a problem if the self-tuner is starting without any prior information about the process parameters, as control is poor until a satisfactory parameter set is established, during which time the controlled variable may diverge from the set point. The integral behaviour is readily identified, and such processes cause difficulty only when an unknown offset term is present at the start. The unstable process poses a little more difficulty at the beginning, but once the initial transients are past, the controller performs quite adequately (Figure 4.9). Controllers #2 and #3 are thus able to cope with a variety of processes, providing good control.

4.2.2 Choice of observer and model polynomials

Wellstead and Zanker [1979] have shown how the pole placing self-tuning controller may be extended to incorporate servo properties at the price of increasing the number of parameters to be identified. If it may be assumed that the poles of the process are correctly placed by the controller, the observer polynomial $\mathbf{A}_0$ may be used to affect the transient response of the controlled system, as suggested in Section 2.1.7. Changing this polynomial is a simpler (and less definite) approach than that of Wellstead and Zanker, but as will be shown, one which can adequately effect control. A process simulated by Wellstead and Zanker [1979] may be used to demonstrate the effects of changing this polynomial.

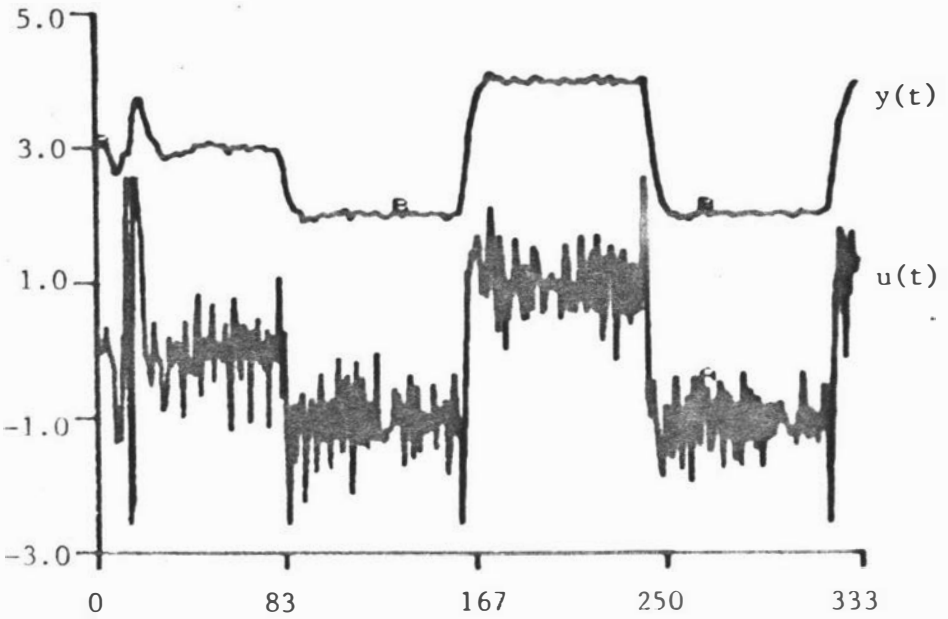


FIGURE 4.9 (a)

Process (G) Unstable

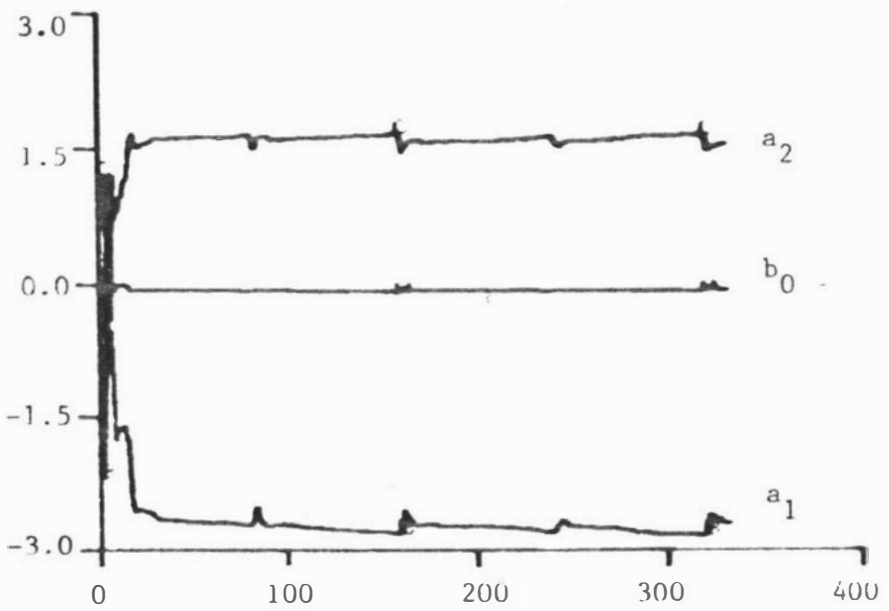


FIGURE 4.9 (b)

Process (G) Unstable

Parameters of $(1+a_1q^{-1}+a_2q^{-2})y=q^{-1}(b_0+b_1q^{-1})u+d+e(t)$

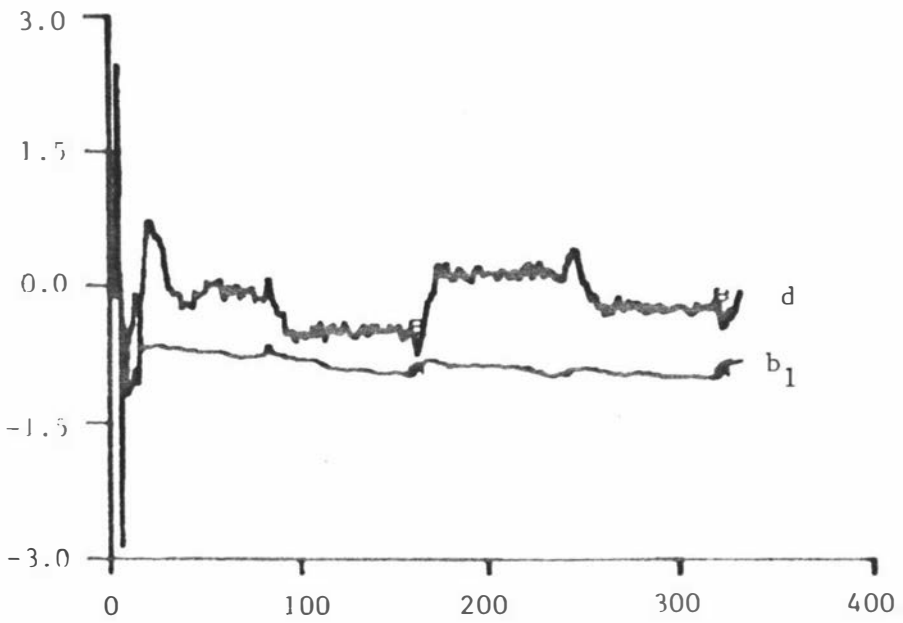


FIGURE 4.9 (c)

Process (G) Unstable

Parameters of $(1+a_1q^{-1}+a_2q^{-2})y = q^{-1}(b_0+b_1q^{-1})u + d + e(t)$

Transfer function	$25\exp(-0.016s)/s$
Discrete process	$(1-q^{-1})y(t)=q^{-2}(1+1.5q^{-1})u(t)$
	$+(1-0.5q^{-1})e(t)$
Sampling rate	100 ms
Delay	k=2
Noise	$E\{e(t)e^{T}(t)\}=0.0133$

Simulations are performed for a variety of combinations of the observer and model polynomials. However, before comparisons may be made between different runs, difficulties in making quantitative comparisons between the performances of different controllers must be overcome. Because a degree of model-following is called for, the penalty function should be calculated based on excursions from the desired trajectory and consequently, the trajectory is taken to be:

$$\mathbf{A}_m y_d(t) = q^{-k}\mathbf{A}_0 w(t) \qquad \ldots.(4.2)$$

where w(t) is the required set point for the loop. Comparisons between systems can be made over a specified time interval during which time identical changes to the set point value are made for each system. Two measures are presented:

$$MSE(y) = (y(t)-y_d(t))^2/nsteps$$

$$MSE(u) = (u(t)-u(t-1))^2/nsteps \qquad \ldots.(4.3)$$

The first is the mean square error of the output from the desired trajectory, calculated from equation (4.2). The second measure is

expected to have a non-zero value, and can only be used for comparison of the control effort required by different controllers on the same process.

Performance summary

Figure 4.10 a-e shows the responses of the process with a representative group of controllers. A complete performance summary is given in Table 4.2. Each cell contains the two measures, MSE(y) and MSE(u), scaled upward by a factor of x1000.

TABLE 4.2
Effect of different observer and model polynomials
MSE(y) and MSE(u) values

	Model polynomial $\mathbf{A}_m$			
	$1-0.8q^{-1}$		1	
Observer polynomial $\mathbf{A}_0$				
$\mathbf{A}_m$	83.54	2.263	52.36	13.72
$\mathbf{A}_m(1+q^{-1})$	78.50	2.727	39.92	12.02
$\mathbf{A}_m(1+q^{-1}\ldots q^{-6})$	47.80	0.695	19.96	4.944
1	32.58	0.457	52.36	13.72
$(1+q^{-1})$	32.65	0.478	39.92	12.02

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-0.5	1.5

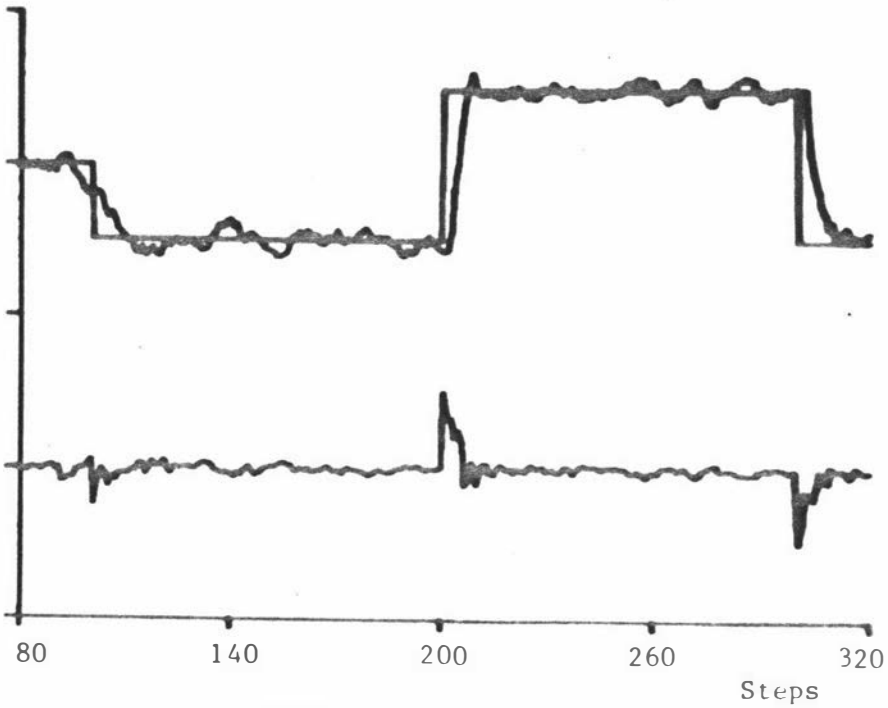


FIGURE 4.10 (a)

Process: $(1-q^{-1})y = q^{-2}(1+1.5q^{-1})u + (1-0.5q^{-1})e$

$A_m = A_0 = 1-0.8q^{-1}$

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-0.5	1.5

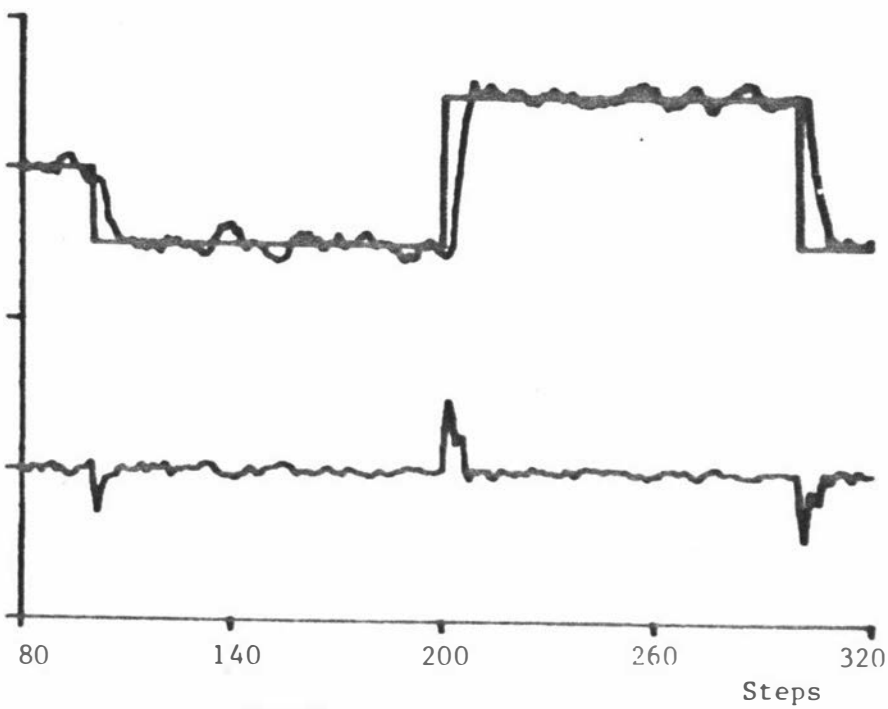


FIGURE 4.10 (b)

$A_m = 1-0.8q^{-1}$ $A_0 = 1 + 0.2q^{-1} - 0.8q^{-2}$

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-0.5	1.5

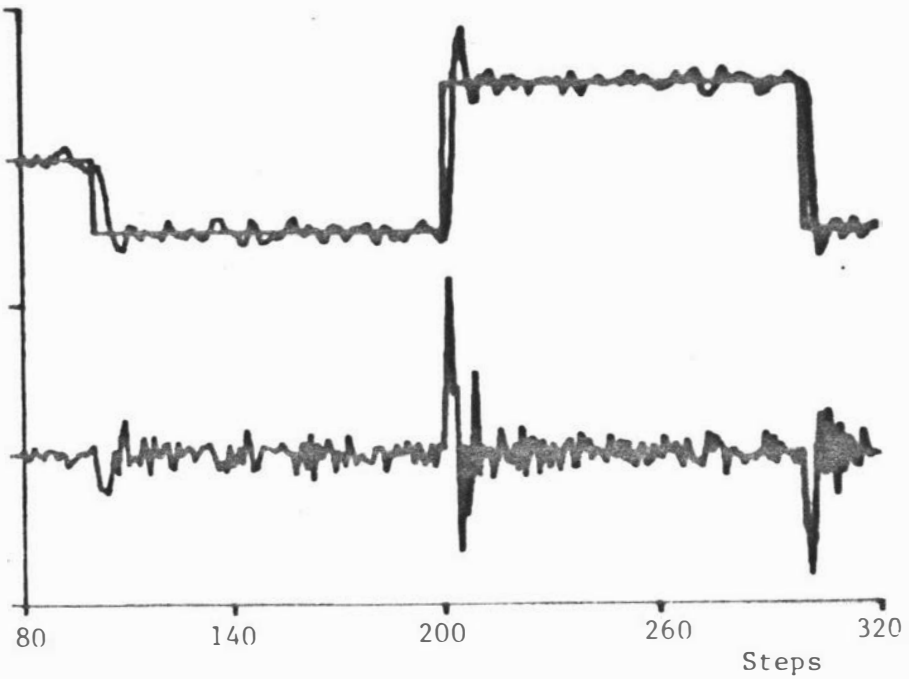


FIGURE 4.10 (c)

$A_m = 1 \qquad A_0 = 1 + q^{-1}$

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-0.5	1.5

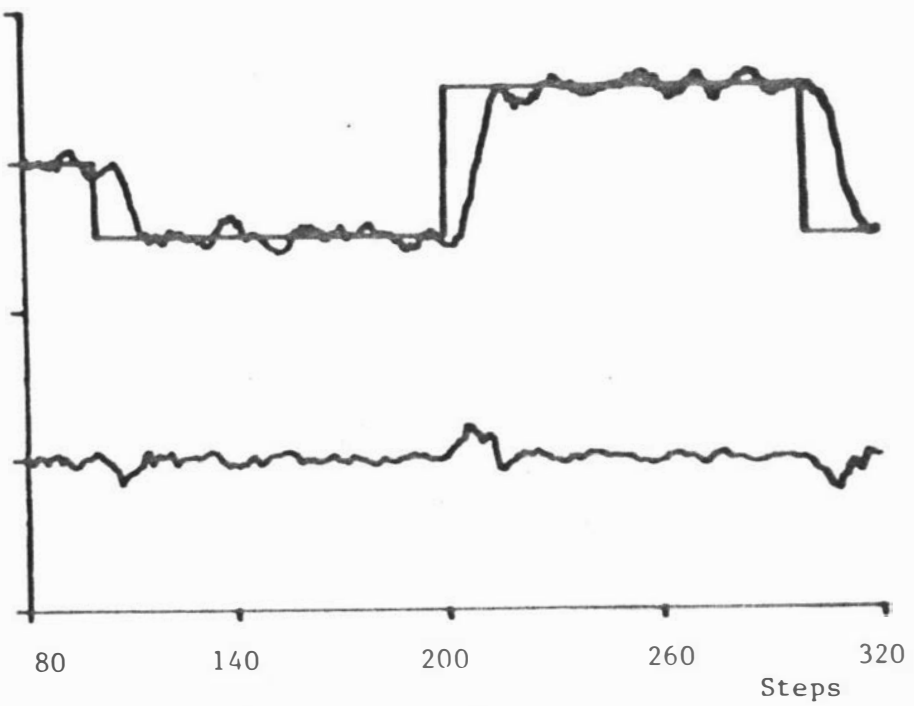


FIGURE 4.10 (d)

$A_m = 1-0.8q^{-1} \qquad A_0 = 1 + q^{-1} + \ldots + q^{-6}$

Var	Min	Max
y(t)	-6	2
w(t)	-6	2
u(t)	-0.4	1.2

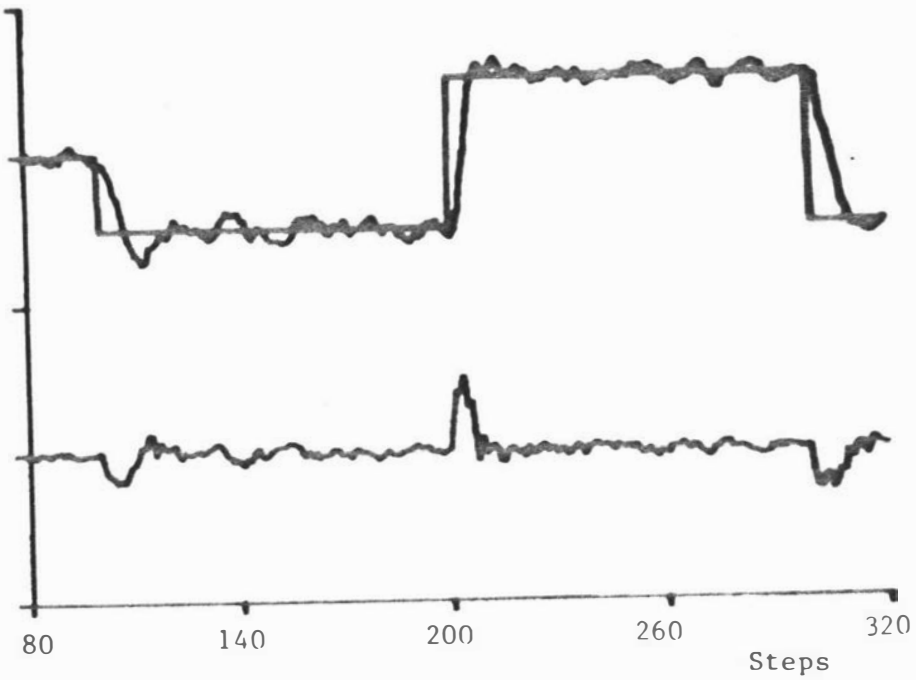


FIGURE 4.10 (e)

$A_m = 1-0.8q^{-1}$ $A_0 = 1$

In summary, the transient behaviour of the controlled process is influenced by the choice of the model and observer polynomials in a manner described by equation (4.2). In this way, factors of the denominator and numerator may be cancelled, to ensure close set-point following (the servo-mechanism problem), or appropriate factors may be left uncancelled in either the numerator (to affect transient response), or in the denominator (to affect process dynamics).

Disturbance rejection

Disturbance rejection relies on the feedforward of measured disturbance terms so that control action may be taken to counteract their effect, a capacity which is important in developing a multivariable controller, as it is the mechanism whereby interactions between loops may be removed. This allows decoupling of the loops, each then responding to its own set-point or command input.

Processes [C], [D] and [F], described previously, are modified to include disturbance terms.

[C']

Discrete model $(1-1.654q^{-1}+1.022q^{-2}-0.2019q^{-3})y(t)$

$=q^{-1}(0.1098+0.0792q^{-1}-0.0229q^{-2})u(t)$

$+q^{-1}(0.1+0.05q^{-1}-0.01q^{-2})v(t)$

Model polynomial $1-0.8q^{-1}$

Observer polynomial 1

[D']

Discrete model $(1-1.654q^{-1}+1.022q^{-2}-0.2019q^{-3})y(t)$

$=q^{-3}(0.1098+0.0792q^{-1}-0.0229q^{-2})u(t)$

$+q^{-3}(0.1+0.05q^{-1}-0.01q^{-2})v(t)$

Model Polynomial $1-0.8q^{-1}$

Observer polynomial 1

[F']

Discrete model $(1-1.9418q^{-1}+0.9418q^{-2})y(t)$

$=q^{-1}(0.0088+0.0086q^{-1})u(t)$

$+q^{-1}(0.0005+0.0004q^{-1})v(t)$

Model polynomial $1-0.8q^{-1}$

Observer polynomial 1

For all models, a noise term $(1-0.5q^{-1})e(t)$ is added, e(t) being a random noise sequence. For models [C'] and [D'], $E\{e(t)e^{T}(t)\}=0.0133$. For model [F'] the noise variance used was 0.0033 because of the low gain of the process, this value keeping the excursions of the control input within reasonable bounds. The results of the simulations are presented in Figures 4.11 - 4.13. The effects of the disturbance term in each case may be assessed by comparing the value of MSE(y) and MSE(u) for each of the processes, with and without the disturbance term. (All values have been multiplied x1000).

TABLE 4.3
Effect of disturbance rejection
MSE(y) and MSE(u) values

	With disturbance		Without disturbance	
	MSE(y)	MSE(u)	MSE(y)	MSE(u)
[C']	44.38	32.53	45.57	14.56
[D']	68.29	36.50	61.79	16.33
[F']	40.38	1000.0	36.61	758.5

The vigourous control effort is required by process [F'] because of the low gain of the process (small sampling interval). In all cases, the MSE(y) value is changed little by the addition of the disturbance, indicating good disturbance rejection by the controller. That this satisfactory behaviour is due to controller action is evidenced by the increased values of the MSE(u) terms.

The controller for the system [F'] also shows that it is possible to place a hard limit on the excursions of the control signal with the explicit form of the self-tuning controller. The effect of limiting u(t) is to induce a more gradual initial movement of the controlled variable following a set-point change, which in turn would be reflected in an increase of the MSE(y) term, although MSE(u) would obviously be decreased. Such a hard limit can have adverse effects if applied when disturbance elimination is required.

Var	Min	Max
y(t)	-9	3
w(t)	-9	3
u(t)	-6	6
v(t)	-2	10

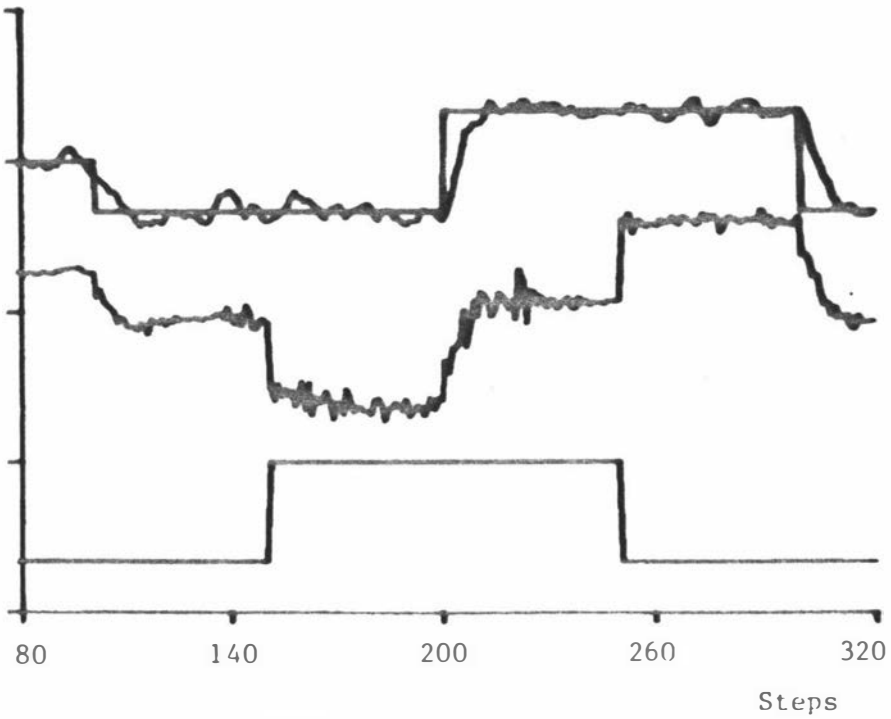


FIGURE 4.11

Process (C') Underdamped process

Disturbance Rejection

Var	Min	Max
y(t)	-9	3
w(t)	-9	3
u(t)	-6	6
v(t)	-2	10

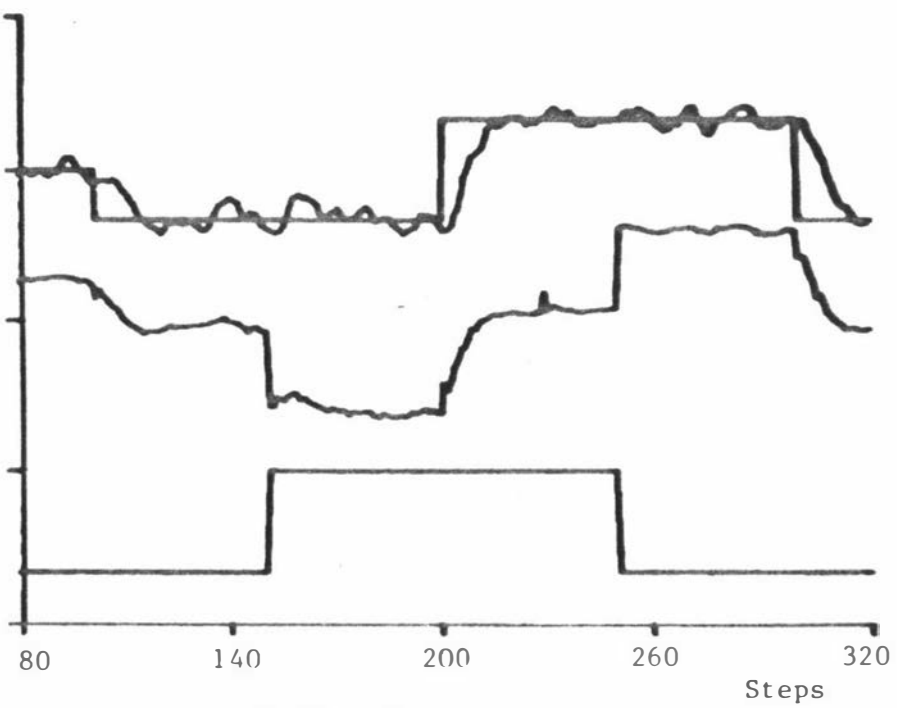


FIGURE 4.12

Process (D')

Underdamped process with time delay

Disturbance Rejection

Var	Min	Max
y(t)	-9	3
w(t)	-9	3
u(t)	-12	12
v(t)	-3	9

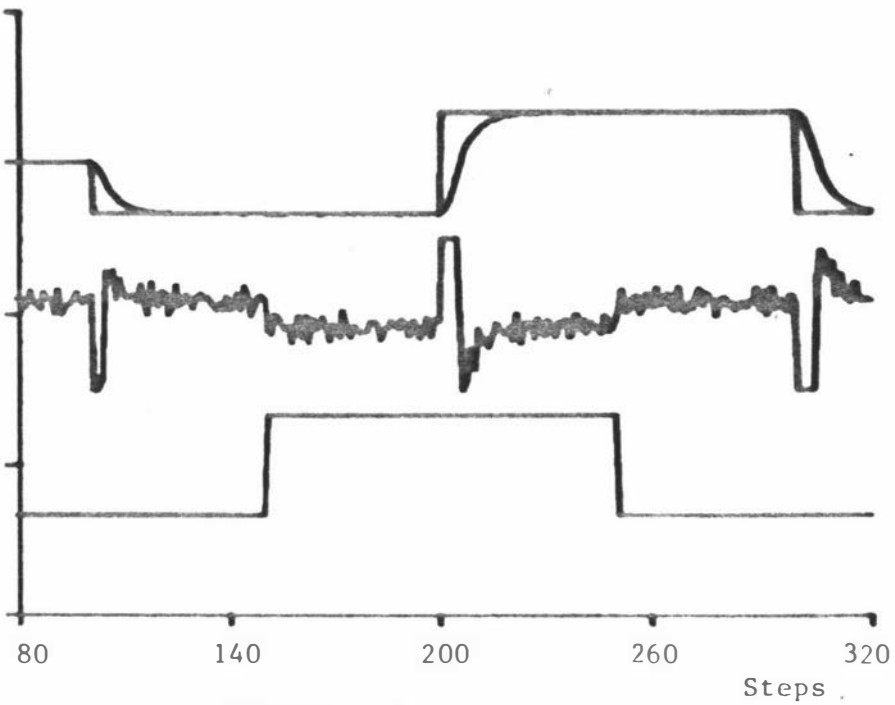


FIGURE 4.13
Process (F') Integral behaviour
Disturbance Rejection

4.2.3 Multivariable Processes

Borisson [1979] and Koivo [1980] have used simulations of multivariable systems to illustrate the behaviour of their multivariable controllers. The same processes have been chosen to illustrate the behaviour of the pole-placing self-tuning controller, and simulations of additional processes will be used to illustrate different features of these controllers.

The simulations which will be described use the same control subroutines as the online control system. The ranges and limits placed on variables have been determined to accord with the use of analogue circuitry. Where necessary, values of noise variances have been adjusted to suitable levels.

[Process 1]

The first process is a model of the head box of a paper machine, and is taken from Borisson [1979].

$$\mathbf{A}y(t) = q^{-1}\mathbf{B}ut(t) + e(t)$$

where

$$\mathbf{A} = \begin{bmatrix} 1-0.99101q^{-1} & 8.80512E-3q^{-1} \\ -0.80610q^{-1} & 1-0.77089q^{-1} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0.89889 & -4.59329e-3 \\ 19.390 & 0.88052 \end{bmatrix}$$

The noise conditions selected were $E\{e(t)e^{T}(t)\}$=diag{3.3E-3,3.3E-

3}. The model and observer polynomials were varied for different simulations. Figures 4.14 - 4.17 present the results of the simulations, while the relative performances of the different controllers may be summarised in the following table, where the entries are the values of MSE(y) and MSE(u) for each of the two loops. For any simulation the same polynomials have been used on both loops.

TABLE 4.4

Control of head-box model

	Model polynomial $\mathbf{A}_m$			
	$[1-0.8q^{-1}]$		[1]	
$\mathbf{A}_0$ Polynomial	MSE(y)	MSE(u)	MSE(y)	MSE(u)
$[1-0.8q^{-1}]$	21.95	15.97	67.33	4008
	29.78	19570	*****	*****
[1]	14.88	4.062	8.992	83.71
	4.062	1078	18.17	39083

All values have been multiplied x1000. The very large values are due to the excessive spikes in the control signal at set-point changes. The values ***** were so large as to be meaningless, and the controller specification for which they occurred would generally not be used. The control of this Process [1] poses no particular difficulties.

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

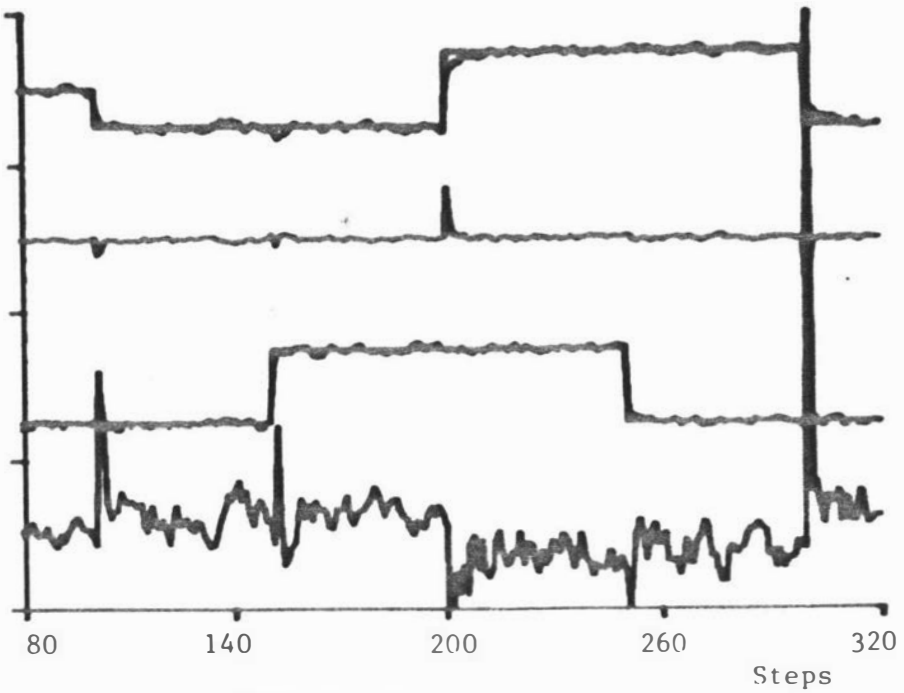


FIGURE 4.14

Head box model

$A_m = A_0 = 1=0.8q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

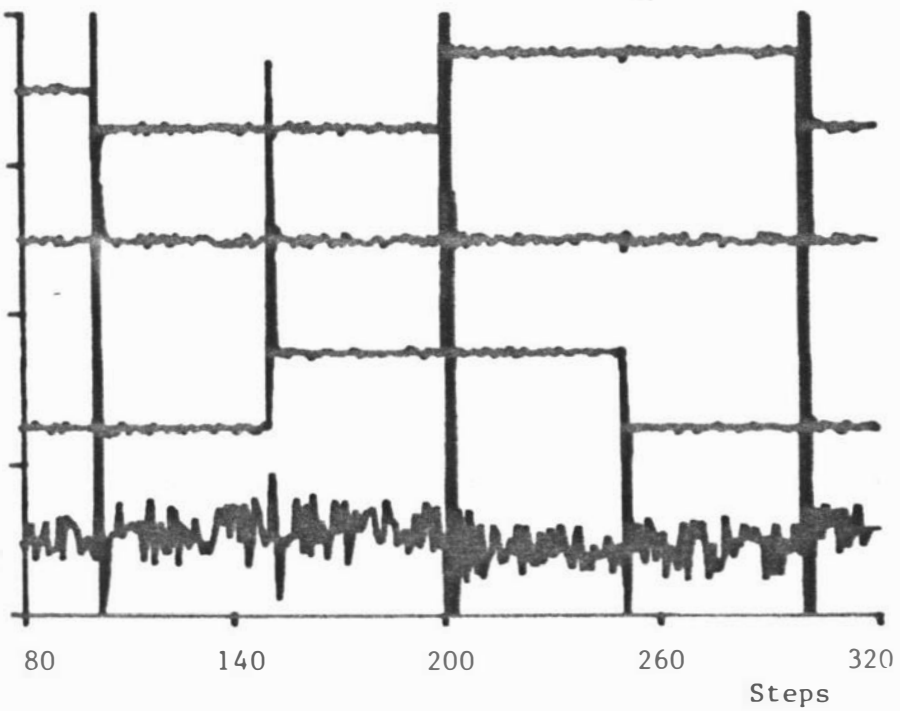


FIGURE 4.15

Head box model

$A_m = 1 \qquad A_0 = 1-0.8q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-4	28

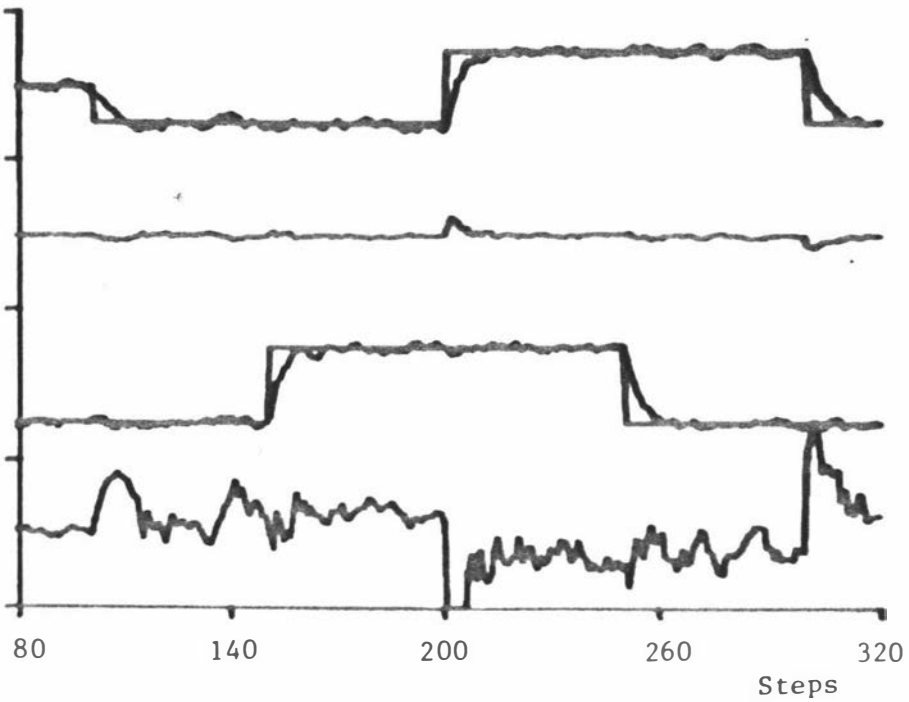


FIGURE 4.16
Head Box Model
$A_m = 1-0.8q^{-1}$ $A_0 = 1$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-4	28

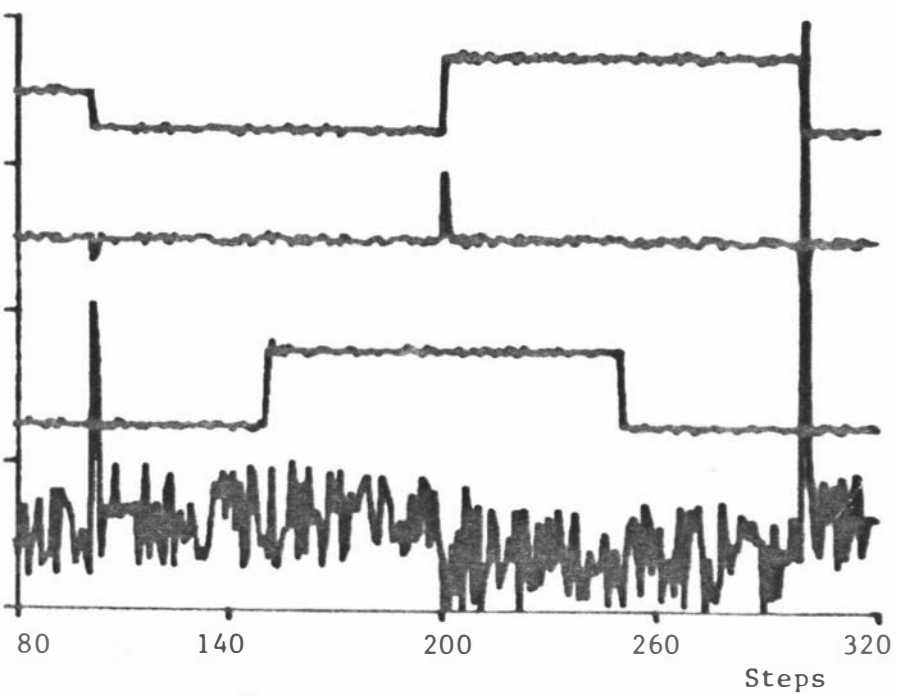


FIGURE 4.17
Head Box Model
$A_m = A_0 = 1$

[Process 2]

[Process 2] is a non-minimum phase system, introduced by Koivo [1980].

$$\mathbf{A}y(t) = q^{-1}\mathbf{B}u(t) + e(t)$$

$$\mathbf{A} = \begin{bmatrix} 1-0.9q^{-1} & 0.5q^{-1} \\ 0.5q^{-1} & 1-0.2q^{-1} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 0.2+q^{-1} & 1 \\ 0.25 & 0.2+q^{-1} \end{bmatrix}$$

The noise is chosen to satisfy $E\{e(t)e^T(t)\}$ = diag{8E-4,8E-4}.

The open loop poles are q_1 = 1.163 and q_2 = -0.060. Owing to the value of the first, the open loop system is unstable. The system is also non-minimum phase as both the zeros (-1.429 and 3.333) are outside the unit circle.

That this is an intractable system to control is evidenced by the difficulty experienced by Koivo [1980] in maintaining the value of y_2 at a set point of 0. Careful placement of the closed loop system poles is necessary to ensure stability. Results are presented in Figure 4.18 for the following controller:

Loop 1: $\mathbf{A}_0=1$ $\mathbf{A}_m=1-0.9q^{-1}$

Loop 2: $\mathbf{A}_0=1$ $\mathbf{A}_m=1-0.8q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

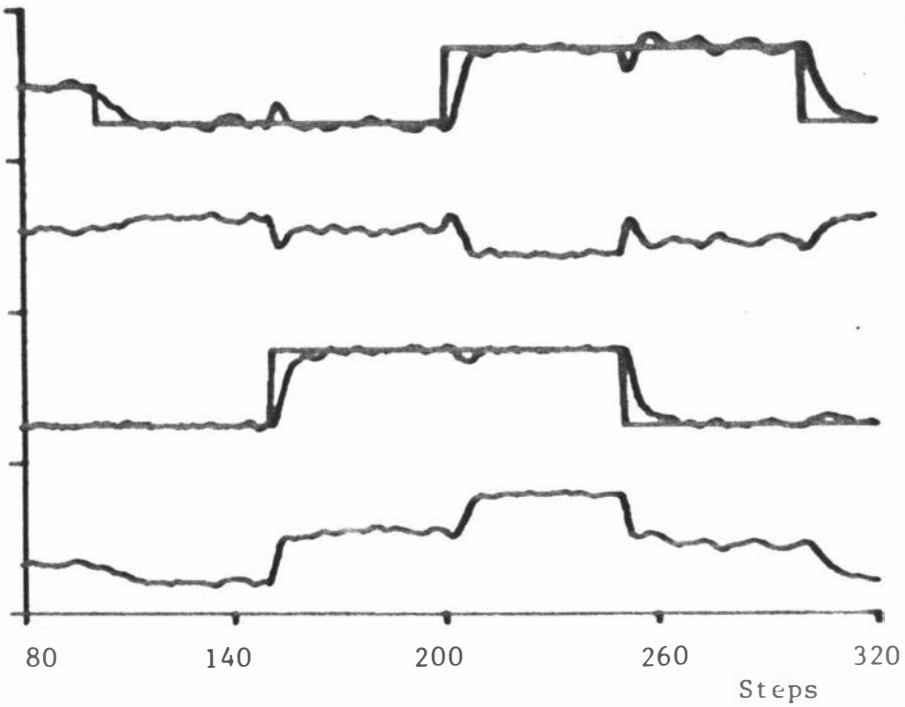


FIGURE 4.18

Unstable non-minimum phase process

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

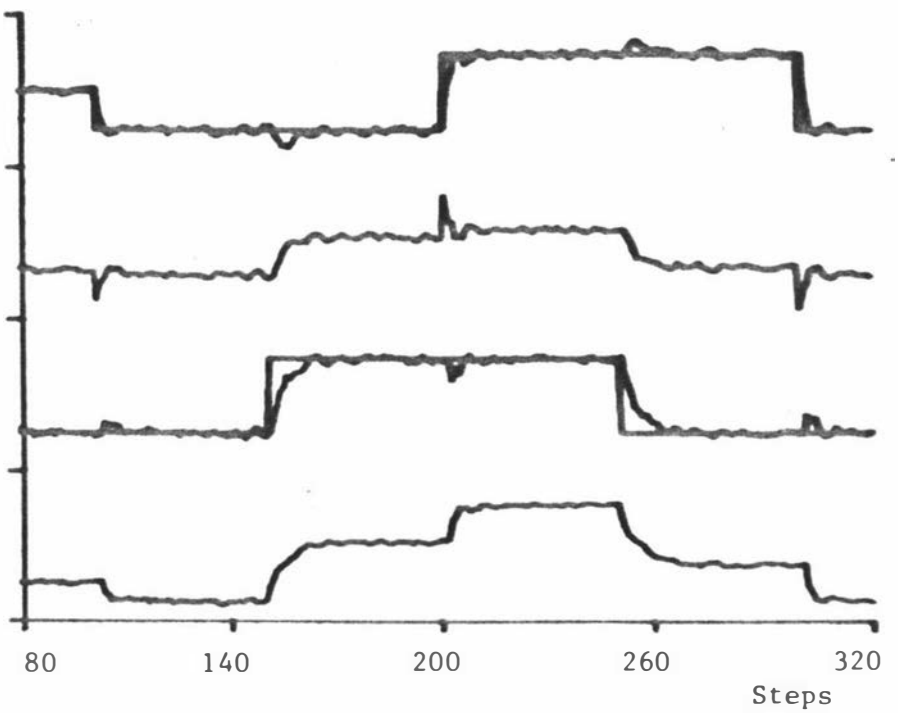


FIGURE 4.19

Non-minimum phase process with time delay

The system is able to follow all the set-point commands, and the standard of control achieved is quite satisfactory. The ability of the controller to stabilise unstable processes, demonstrated for the single loop case, is now extended to multivariable processes.

[Process 3]

Borisson [1979] introduced the process described below, which is very similar to process [2] described above, but incorporates a time delay.

$$\mathbf{A}y(t) = q^{-2}\mathbf{B}u(t) + \mathbf{C}e(t)$$

$$\mathbf{A} = \begin{bmatrix} 1-0.9q^{-1} & 0.5q^{-1} \\ 0.5q^{-1} & 1-0.2q^{-1} \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 1-0.2q^{-1} & -0.4q^{-1} \\ 0.2q^{-1} & 1-0.8q^{-1} \end{bmatrix}$$

$$\text{with } E\{e(t)e^{T}(t)\} = \begin{bmatrix} 3.3E\text{-}3 & 3.3E\text{-}3 \\ 3.3E\text{-}3 & 6.6E\text{-}3 \end{bmatrix}$$

The open loop zeros are -0.060 and 1.163 and this process is consequently unstable. The following controller is used:

Loop 1: $A_0=1$ $A_m=1-0.5q^{-1}$

Loop 2: $A_0=1$ $A_m=1-0.5q^{-1}$

This process is easier to control than Process 2, but the delay in the action of the input variables now means that any disturbances which take effect before a change in control can counteract them cannot be eliminated. The y interactions provide such disturbances, and their effect can be seen in Figure 4.19 which presents the results of the simulation.

4.2.4 Processes with changing parameters

The self-tuning controller is particularly important for the control of processes where the parameters of the process change with time, which may be due for example to changes in raw material being processed, or to some change in the operation of the plant, or to non-linearities where the linearised process model changes with the set-point. In this section, the same basic multivariable process is adapted to simulate process changes and the behaviour of the controlled process is examined under conditions of:

(a) A sudden change to one of the model parameters.
(b) A sudden time delay change.
(c) A non-linearity.

Process:

$$\mathbf{A}y(t) = q^{-2}\mathbf{B}u(t) + \mathbf{C}e(t)$$

$$A = \begin{bmatrix} 1-.772q^{-1}+.391q^{-2} & -0.070q^{-1}-0.115q^{-2} \\ 0.047q^{-1}+0.077q^{-2} & 1-1.021q^{-1}-0.016q^{-2} \end{bmatrix}$$

$$B = \begin{bmatrix} 0.400+0.256q^{-1} & 0.240+0.058q^{-1} \\ 0.268+0.096q^{-1} & 0.801+0.192q^{-1} \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 1-0.5q^{-1} & 0 \\ 0 & 1-0.2q^{-1} \end{bmatrix}$$

Noise: $E\{e(t)e^{T}(t)\} = \mathrm{diag}\{3.3E\text{-}3, 3.3E\text{-}3\}$.

Controller: $\mathbf{A}_m = 1-0.8q^{-1}$ and $\mathbf{A}_0 = 1$ for both loops.

The self-tuning algorithm identifies 12 parameters for each loop, comprising the polynomials $\mathbf{A}_{nn}$, (2 parameters), $\mathbf{B}_{nn}$, $\mathbf{A}_{nm}$, $\mathbf{B}_{nm}$ (3 parameters each, allowing for the delay), and the offset d. A variable forgetting factor is used, which varies between $0.95<f<1.00$. Allowing a greater range of forgetting factors would increase the convergence rate, but necessitate other checks on the condition of the covariance matrix where a noisy process

(with errors which decrease the forgetting factor) is held at a fixed set point (with little new information about process dynamics).

Process changes:

(a) Changes to model parameters

The $\mathbf{B}_{22}$ polynomial is changed to $0.4-0.2q^{-1}$ at time step 225.

(b) Changes to time delay

The time delay for loop 1 is set at k=1 for the first 225 steps, and then changed to k=2 for the remainder of the run.

(c) Nonlinearity

(i) $y_1(t) = y_1(t) + 0.2y_1(t-2)u_1(t-2)$

(ii) $y_1(t) = y_1(t) + 0.2y_2(t-2)u_1(t-2)$

Both cases are simulated.

Simulation results:

These may be summarised in a table where the entries are the pair of values [MSE(y),MSE(u)] for each of the two loops. The values have been multiplied x1000.

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

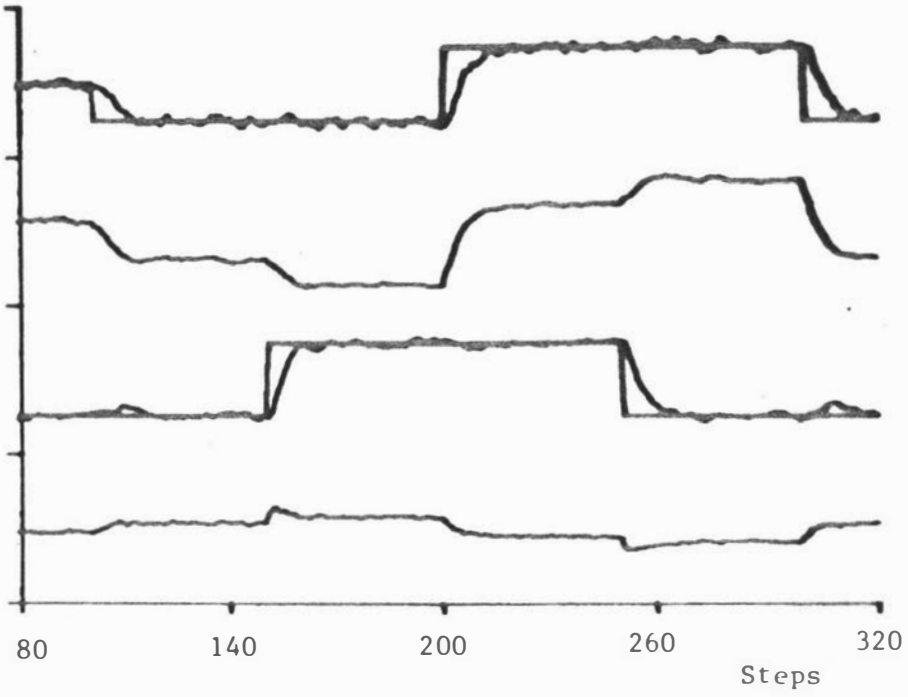


FIGURE 4.20

Multivariable process

No Parameter changes

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

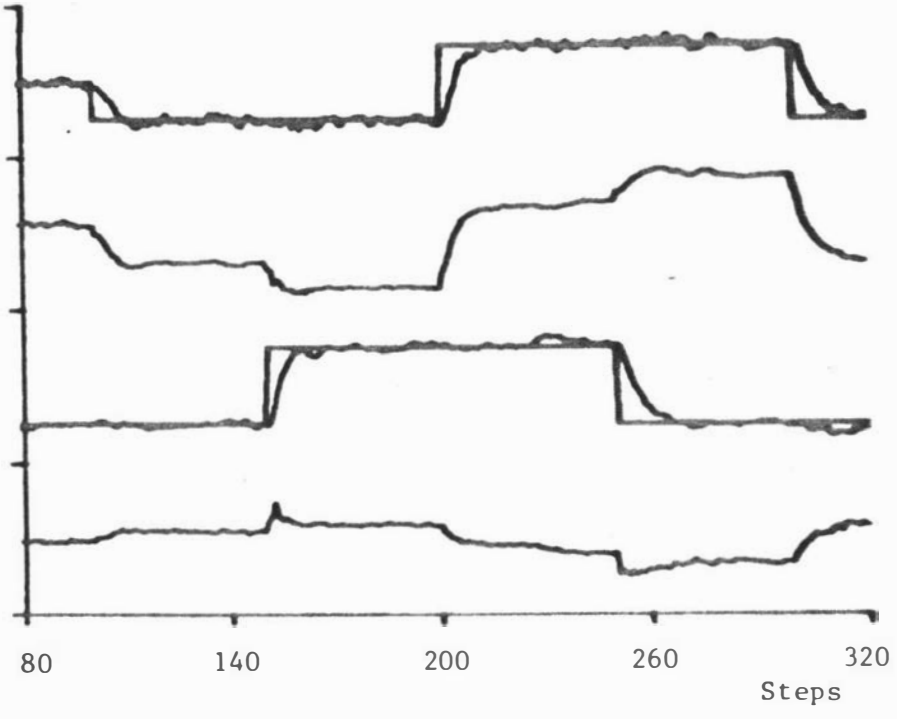


FIGURE 4.21 (a)

Effect of parameter change at step 225

Var	Min	Max
1	-14	2
2	-12	4
3	-10	6
4	-8	8
5	-6	10
6	-4	12

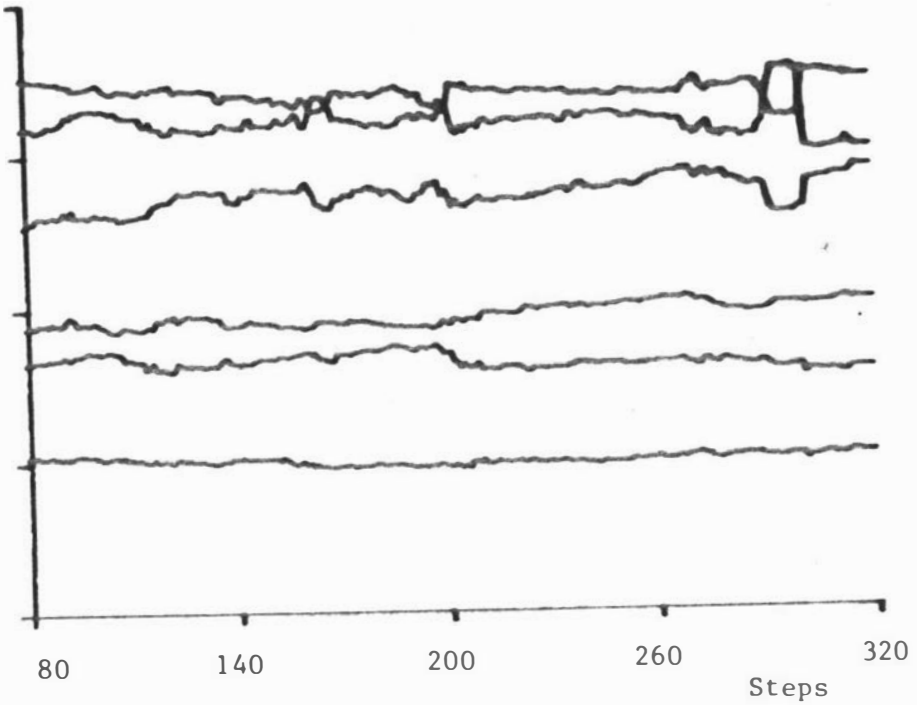


FIGURE 4.21 (b)

Parameter Estimates

Parameter change at step 225

Loop 1 $(B_{11}(3), A_{11}(2), d_1)$

Var	Min	Max
1	-14	2
2	-12	4
3	-10	6
4	-8	8
5	-6	10
6	-4	12

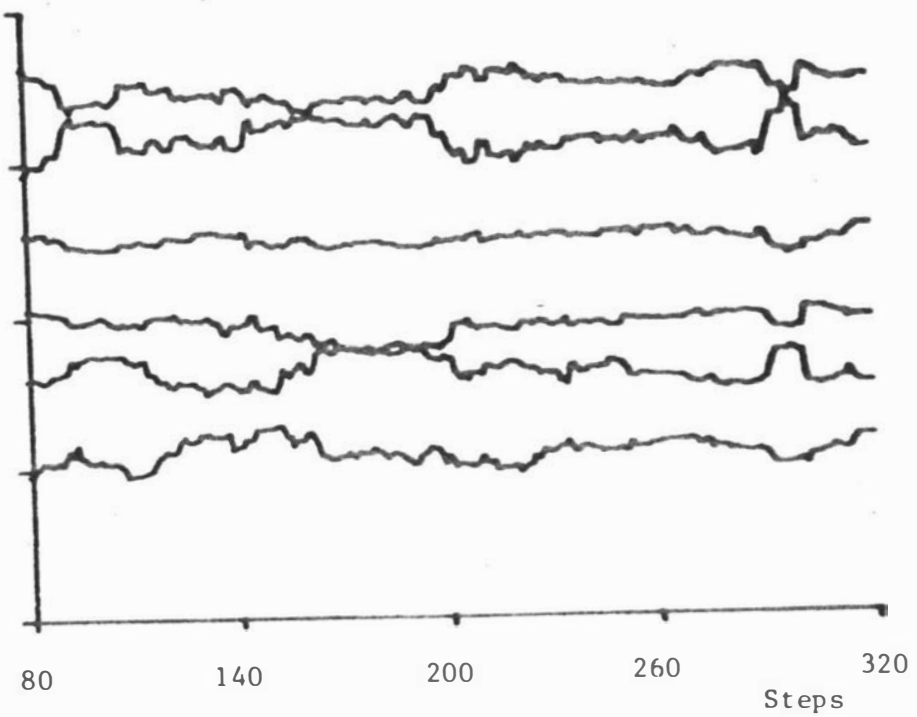


FIGURE 4.21 (c)

Parameter Estimates

Parameter change at step 225

Loop 1 $(A_{12}(3), B_{12}(3))$

Var	Min	Max
1	-14	2
2	-12	4
3	-10	6
4	-8	8
5	-6	10
6	-4	12

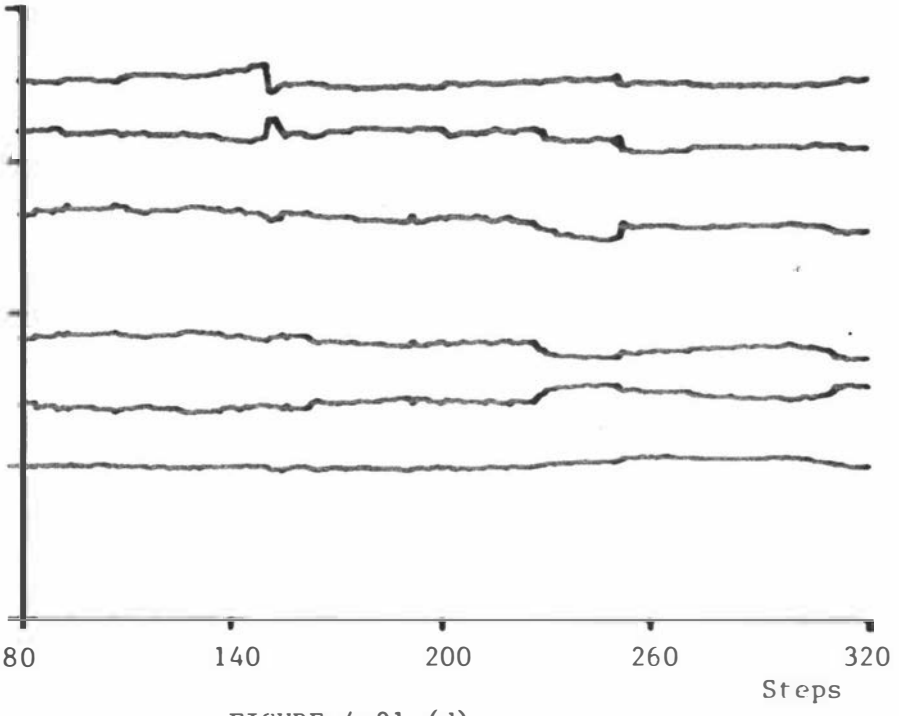


FIGURE 4.21 (d)

Parameter Estimates

Parameter change at step 225

Loop 2 $(B_{22}(3), A_{22}(2), d)$

Var	Min	Max
1	-14	2
2	-12	4
3	-10	6
4	-8	8
5	-6	10
6	-4	12

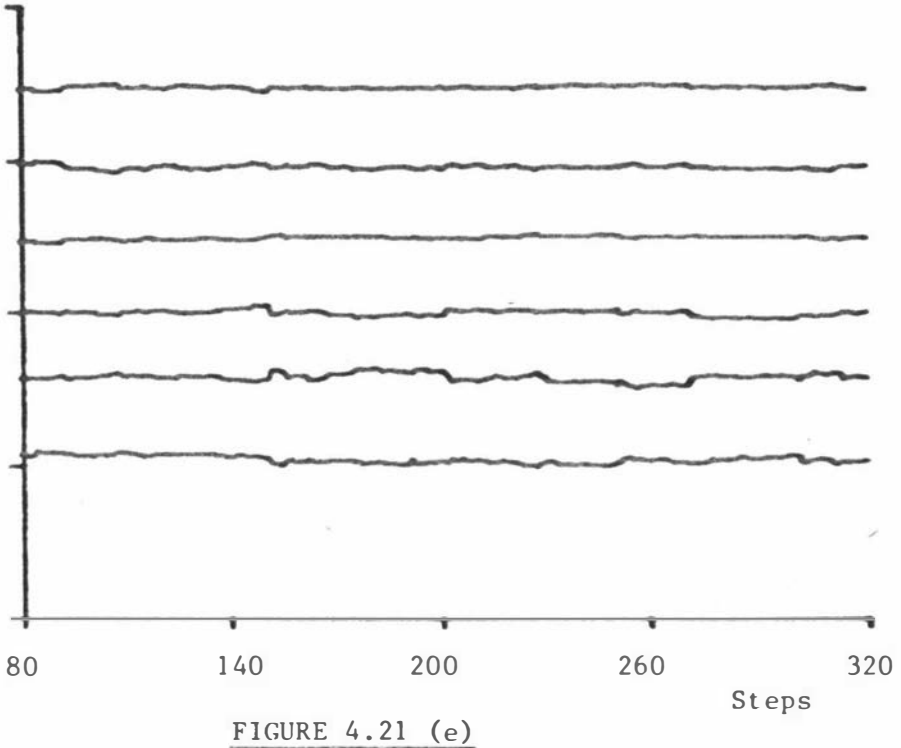


FIGURE 4.21 (e)

Parameter Estimates

Parameter change at step 225

Loop 2 $(A_{21}(3), B_{21}(3))$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

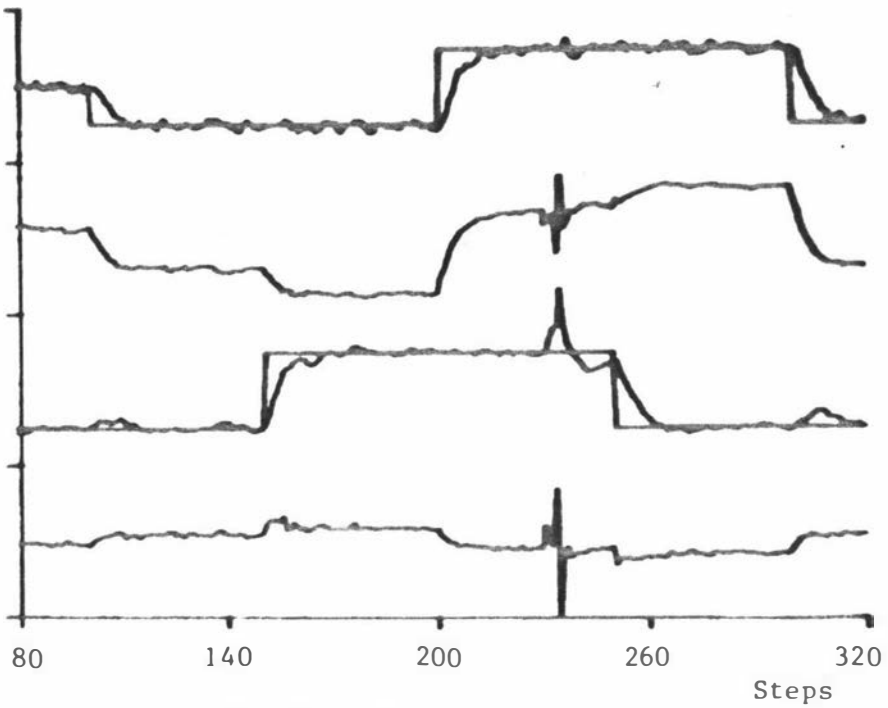


FIGURE 4.22 (a)

Effect of delay change at step 225

Var	Min	Max
1	-14	2
2	-12	4
3	-10	6
4	-8	8
5	-6	10
6	-4	12

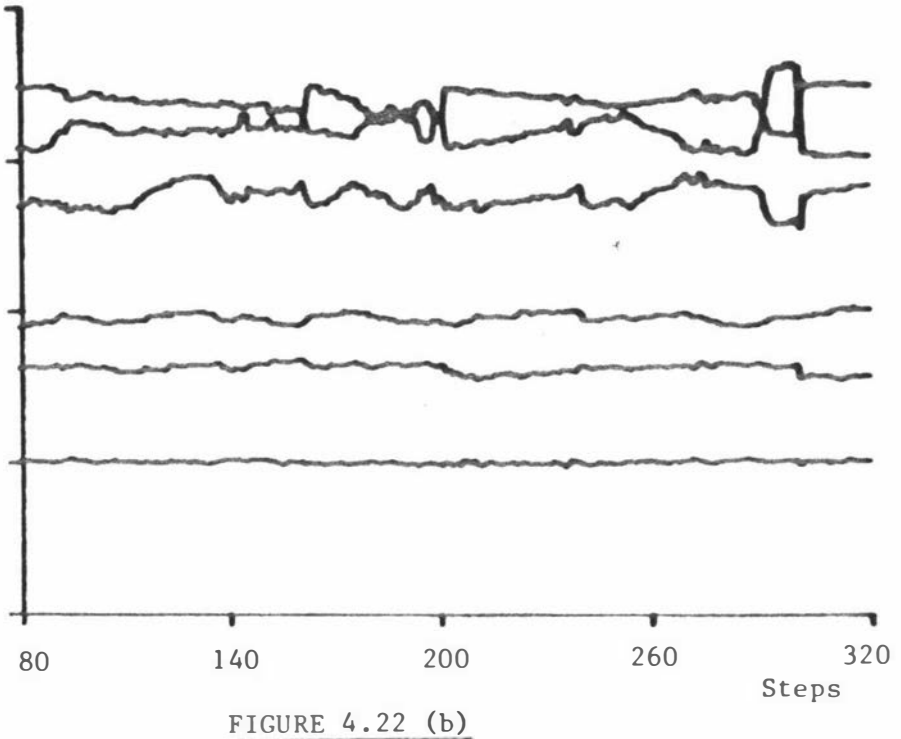


FIGURE 4.22 (b)

Parameter Estimates

Effect of delay change step 225

Loop 1 $(B_{11}(3), A_{11}(3), d_1)$

Var	Min	Max
1	-14	2
2	-12	4
3	-10	6
4	-8	8
5	-6	10
6	-4	12

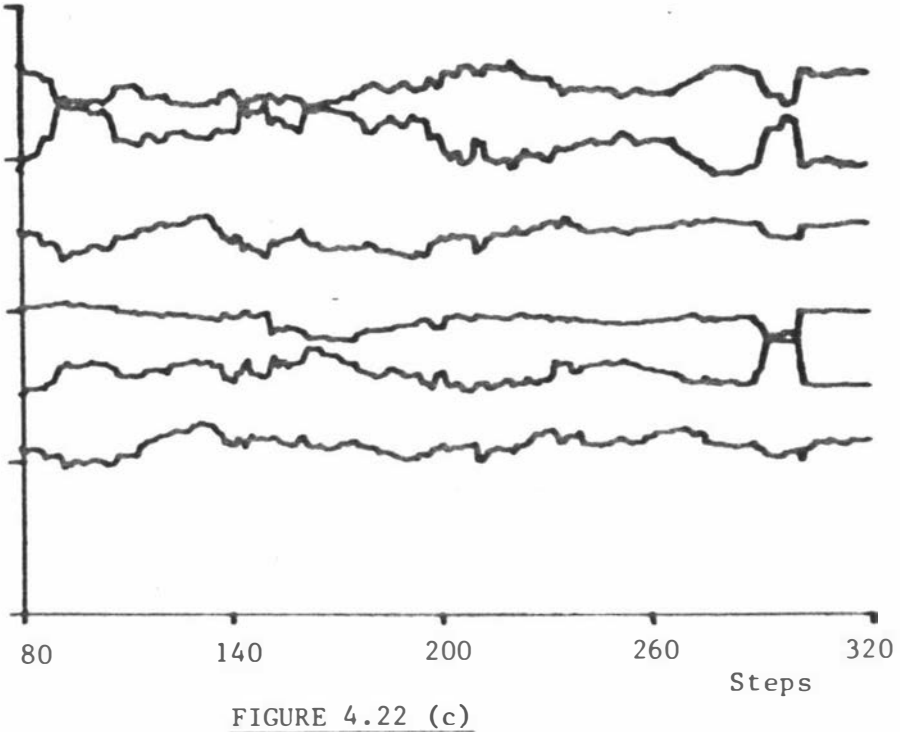


FIGURE 4.22 (c)

Parameter Estimates

Effect of delay change at step 225

Loop 1 $(A_{12}(3), B_{12}(3))$

Var	Min	Max
1	-14	2
2	-12	4
3	-10	6
4	-8	8
5	-6	10
6	-4	12

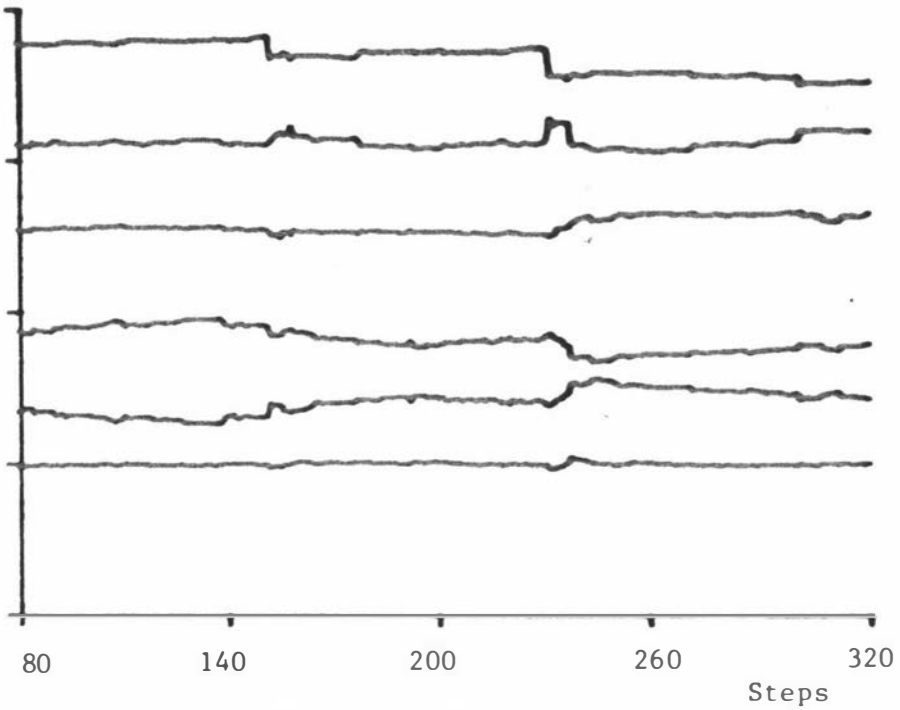


FIGURE 4.22 (d)

Parameter Estimates

Effect of delay change at step 225

Loop 2 $(B_{22}(3), A_{22}(3), d_2)$

Var	Min	Max
1	-14	2
2	-12	4
3	-10	6
4	-8	8
5	-6	10
6	-4	12

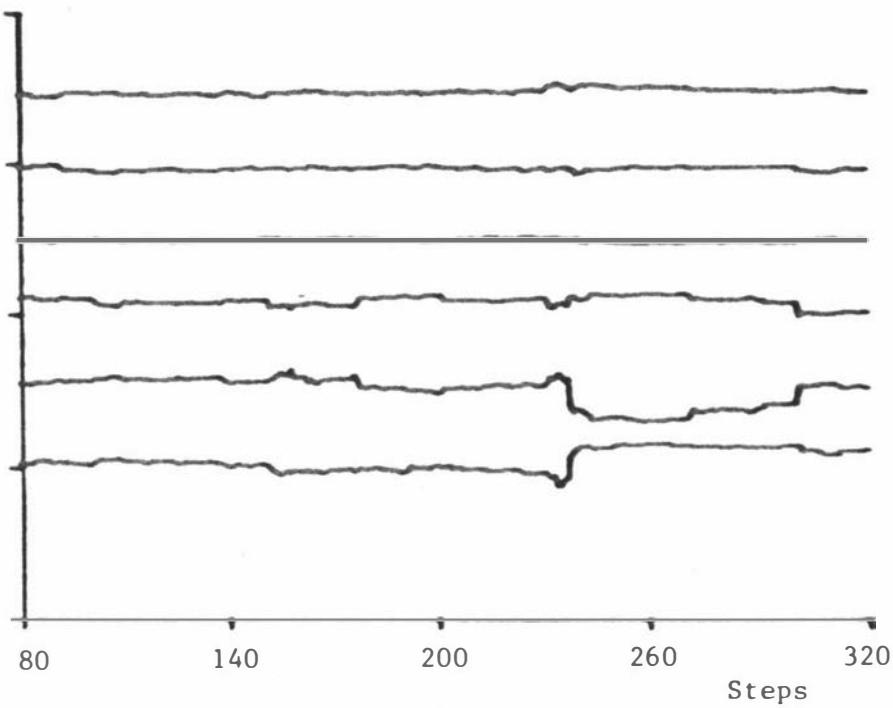


FIGURE 4.22 (e)

Parameter Estimates

Effect of delay change at step 225

Loop 2 $(A_{21}(3), B_{21}(3))$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

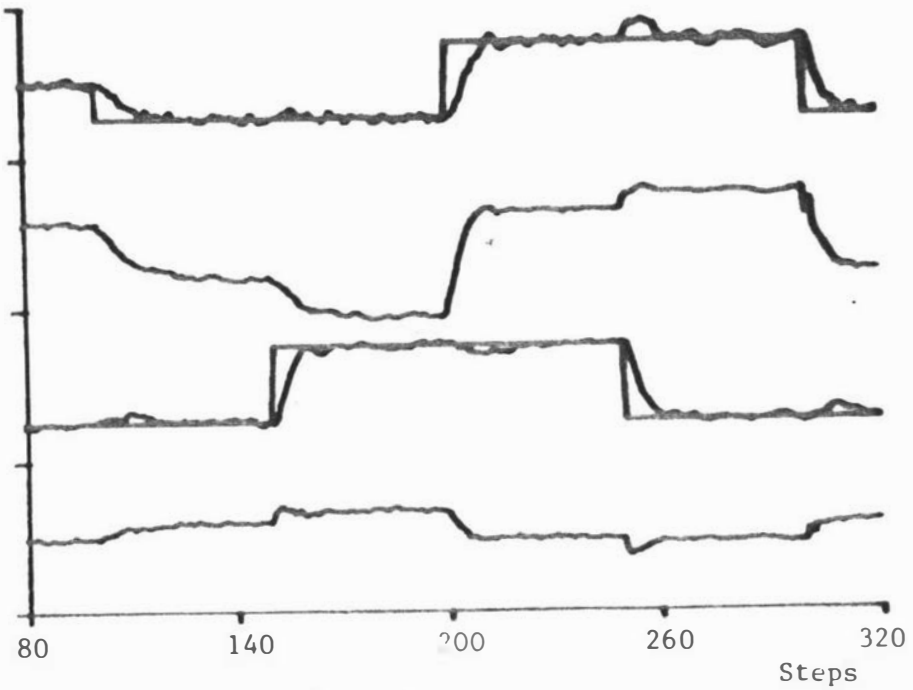


FIGURE 4.23

Effect of non-linearity (i)

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

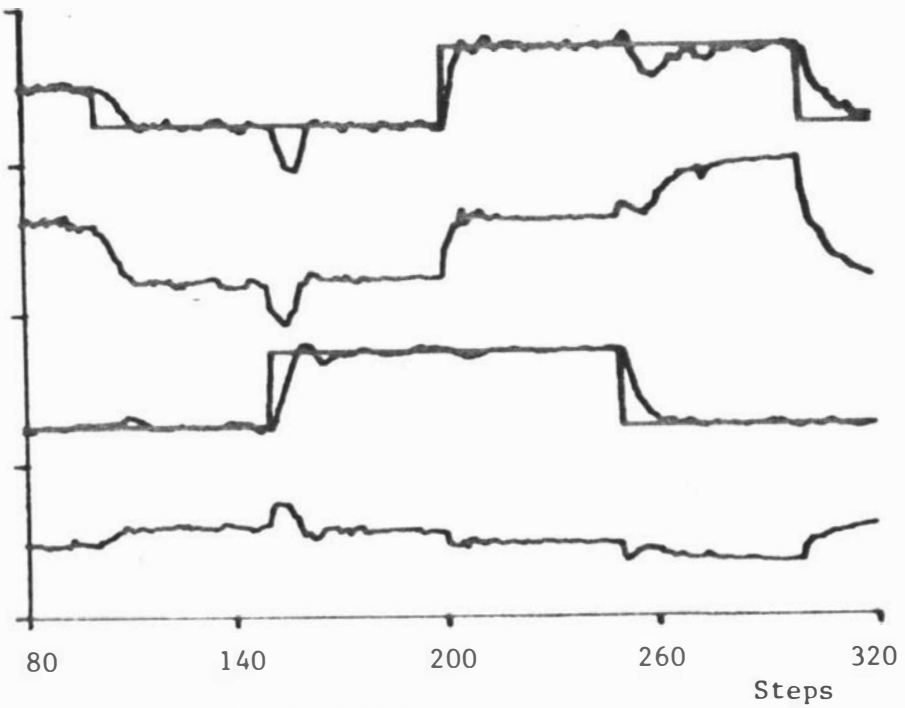


FIGURE 4.24

Effect of non-linearity (ii)

TABLE 4.5
Effect of process changes

	Loop 1 MSE(y) MSE(u)	Loop 2 MSE(y) MSE(u)
No changes	[19.72,7.863]	[13.24,2.251]
Parameter changes	[18.37,10.91]	[28.00,6.886]
Time delay change	[18.31,45.41]	[57.59,97.03]
Non-linearity (i)	[34.75,14.84]	[17.49,3.768]
Non-linearity (ii)	[90.61,26.61]	[22.66,9.587]

Figures 4.20 - 4.24 depict graphically the process responses. For the no-change case, (Figure 4.20), the interaction between the loops is evident in the behaviour of the control inputs necessary to maintain the correct set-points. Disturbance rejection is good, but noticeable due to the delayed response to an input change, and the fact that some interactions are not delayed.

Figure 4.21 shows the effect of changing the gain of loop 2 at time step 225. Re-estimation of the parameters is rapid following the change with the variable forgetting factor operating.

A sudden delay change of an integer number of time steps is a severe test of a controller, and the effects of this are shown in Figure 4.22. Control is lost for 4 or 5 time steps, following which the controller overcorrects for the perturbation, and then recovers as the re-estimation of parameters takes effect. A gradual time delay change is a more usual situation, which would be manifested as progressive model parameter changes. The non-minimum phase characteristics which may accompany such changes are no problem because either the controller relies on a pole

placing design, or for the interactions, the possibility of non-minimum phase polynomials has been allowed for (Chapter 2).

Two non-linearities of different severity are used in obtaining the results for Figures 4.23 and 4.24. The controller linearises its model about the operating point of the process, and this model must change as the set-points are altered. The degraded behaviour before the parameters are re-estimated is evidenced by the errors which occur at set-point changes. A cautious controller, producing slow response to a set point change, has an advantage in these situations, allowing more time for parameter re-estimation. Suitable choice of the observer polynomial provides the necessary caution, which is valuable in real applications.

4.2.5 Differences between pole placing controllers

In Section 2.1.9 it was shown how the calculation of the control inputs to the process involves the inversion of the $\mathbf{H}_0$ matrix, and how using the simplifying assumption, due to Borisson [1979] that $\mathbf{B}_0$ is a diagonal matrix, (i.e. disturbances due to inputs to other loops have a delayed effect), $\mathbf{H}_0$ becomes diagonal. Controller #2 assumes that $\mathbf{H}_0$ is diagonal, while controller #3 removes this assumption, at the price of the additional computation to invert $\mathbf{H}_0$.

Simulations are presented to illustrate the difference in the performance of the two controllers #2 and #3, using the process of section 4.2.4. These results are shown graphically in Figures 4.25 - 4.29, and Table 4.6 shows the values for MSE(y) and MSE(u) for each of the two loops for the different controller specifications. The table entries have been multiplied x1000.

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

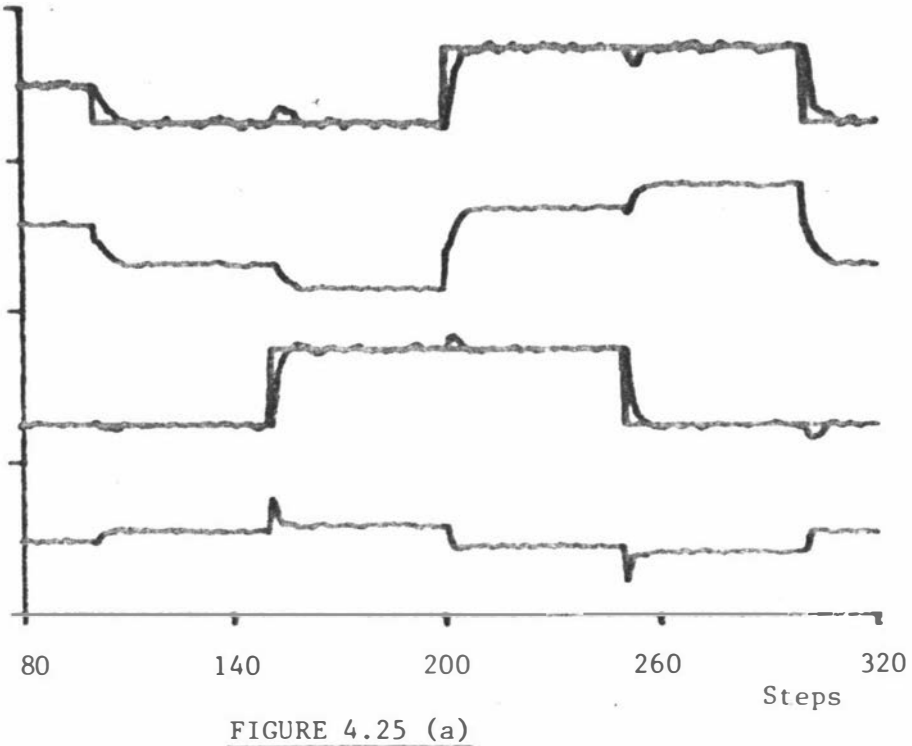


FIGURE 4.25 (a)

Controller comparison

Controller #2 $A_0 = 1$ $A_m = 1-0.5q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

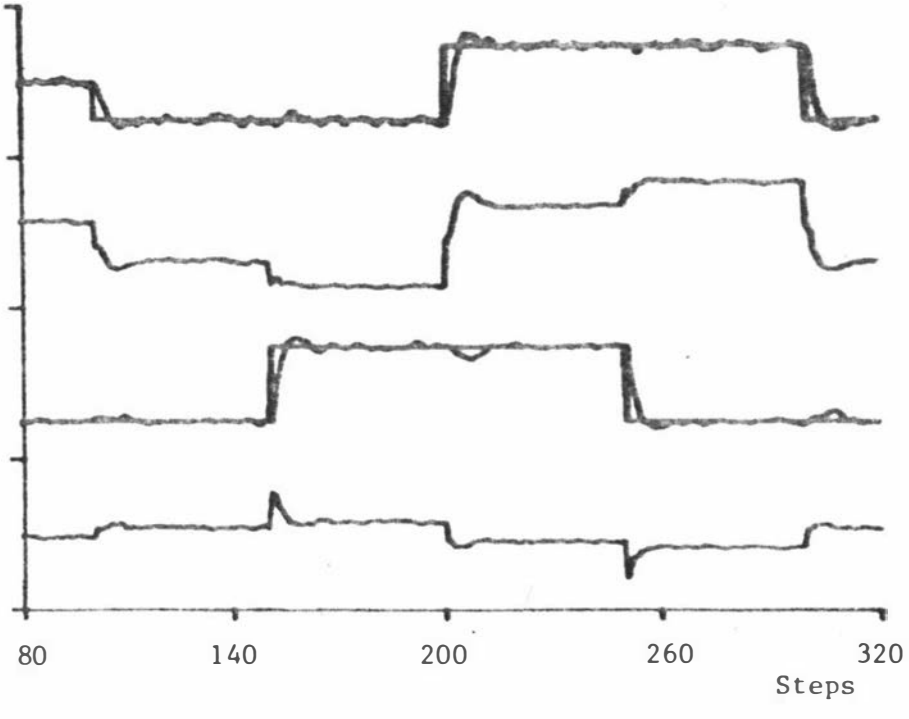


FIGURE 4.25 (b)

Controller comparison

Controller #3

$A_0=1$ $A_m=1-0.5q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

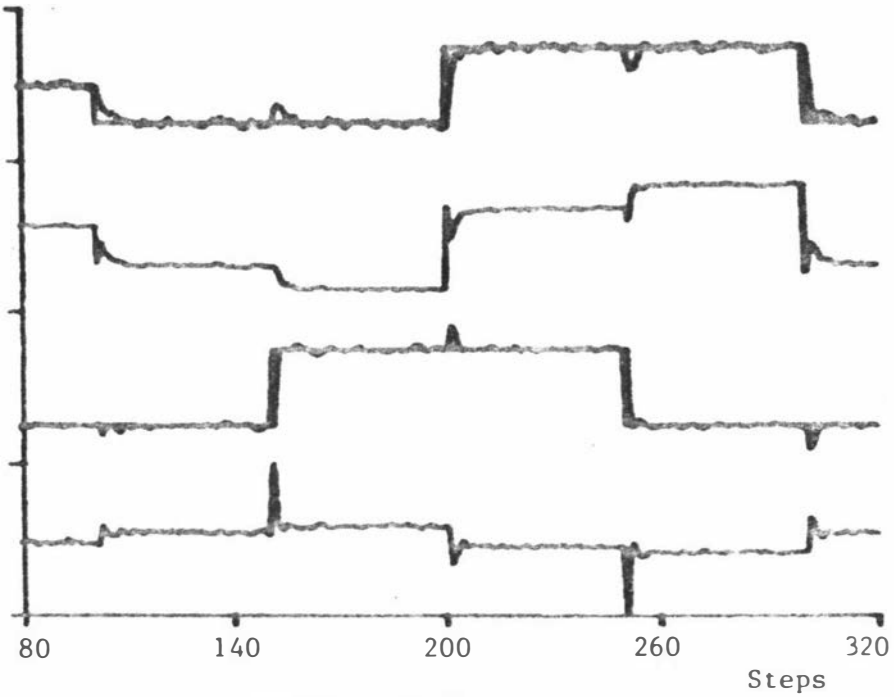


FIGURE 4.26 (a)

Controller comparison

Controller #2 $A_0 = A_m = 1-0.5q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

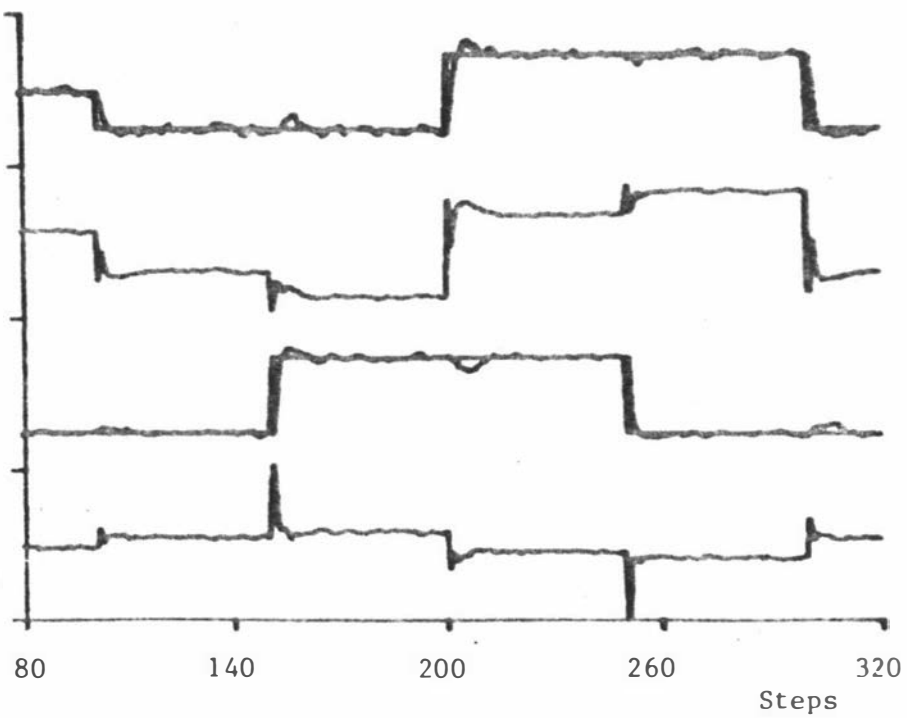


FIGURE 4.26 (b)

Controller comparison

Controller #3

$A_0 = A_m = 1-0.5q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

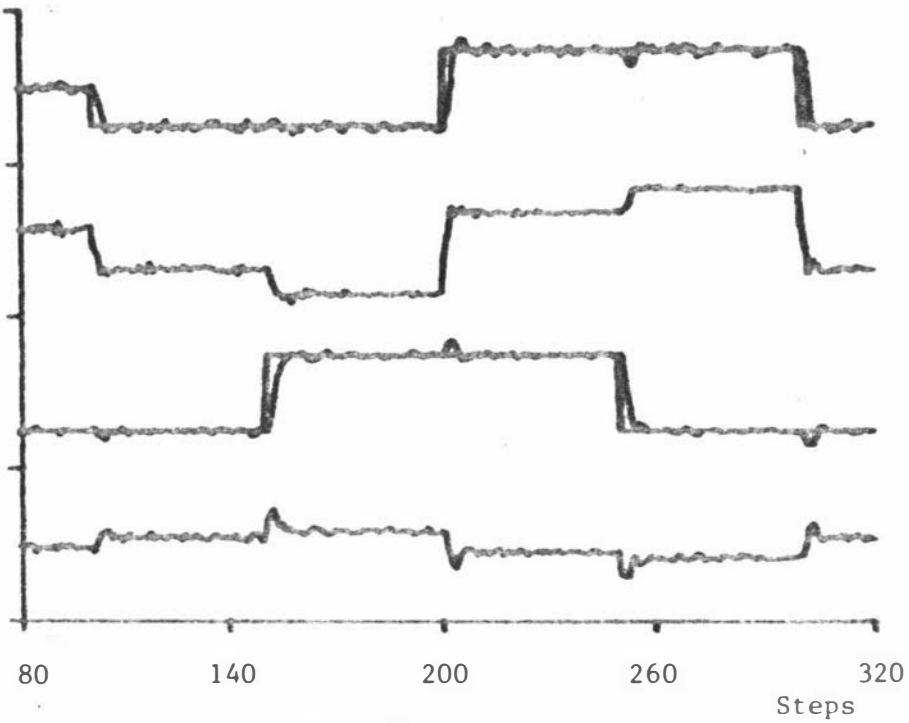


FIGURE 4.27 (a)

Controller comparison

Controller #2 $A_0 = 1+q^{-1}+q^{-2}$ $A_m = 1$

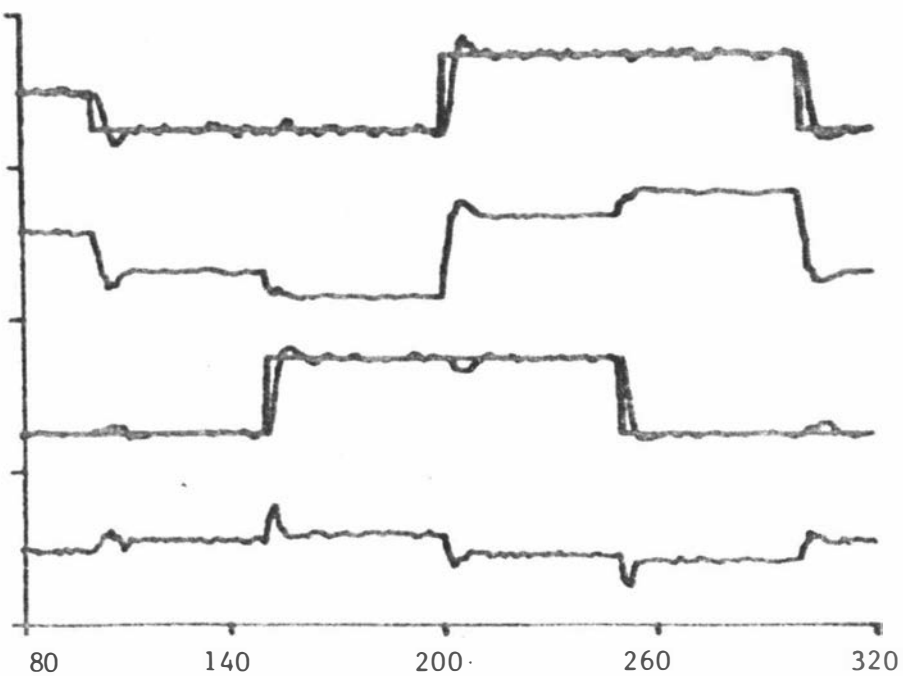
Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

FIGURE $.27 (b)

Controller comparison

Controller #3

$A_0=1+q^{-1}+q^{-2}$ $A_m=1$



Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

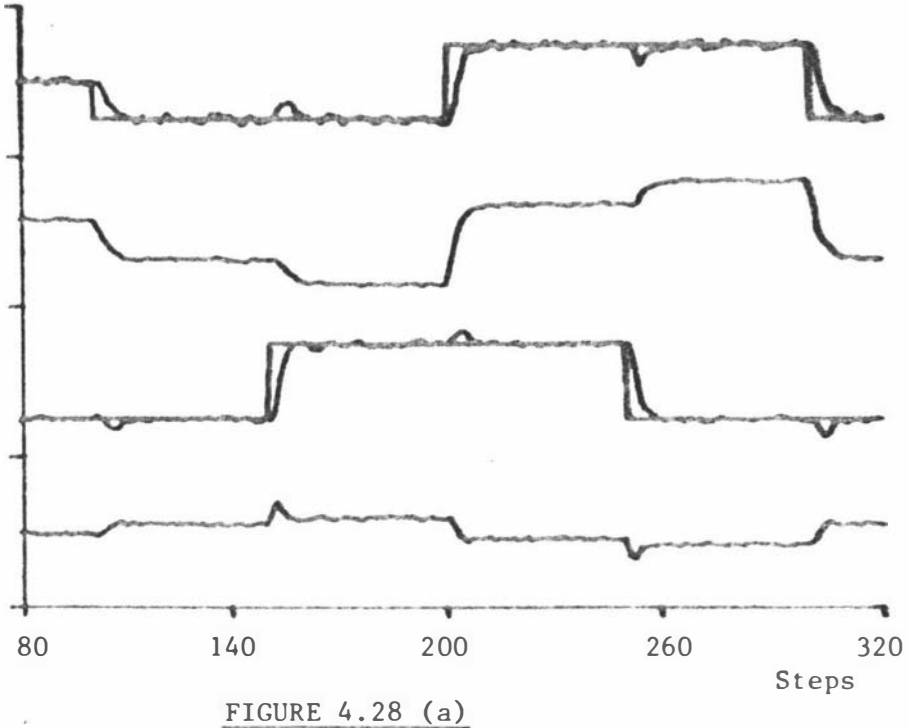


FIGURE 4.28 (a)

Controller comparison

Controller #2 $A_0 = 1+q^{-1}+q^{-2}$ $A_m = 1-0.5q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

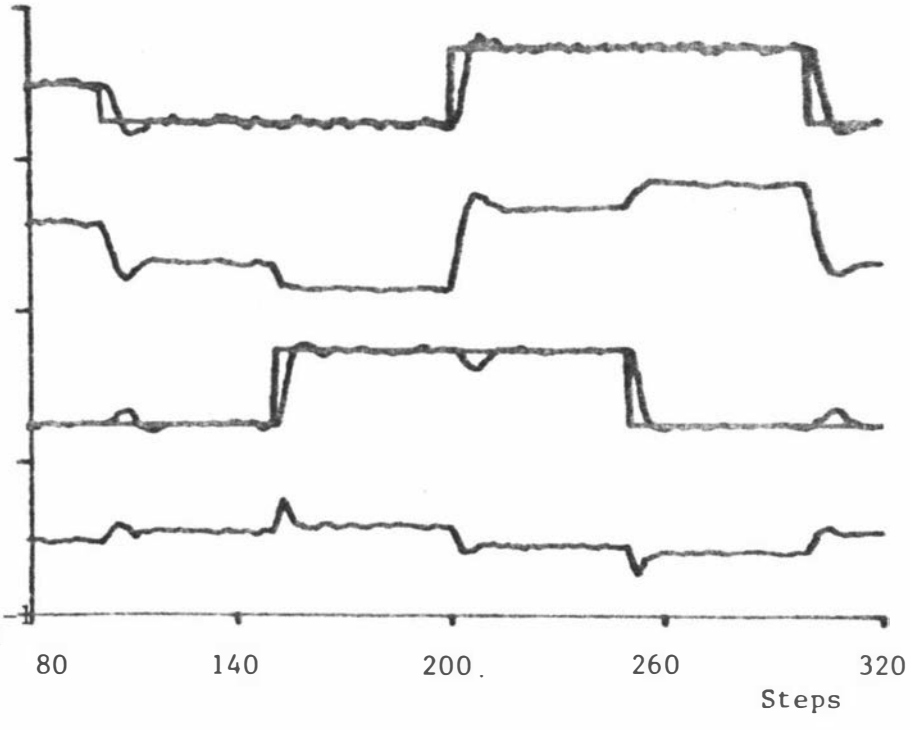


FIGURE 4.28 (b)

Controller comparison

Controller #3

$A_0=1+q^{-1}+q^{-2}$ $A_m=1-0.5q^{-1}$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

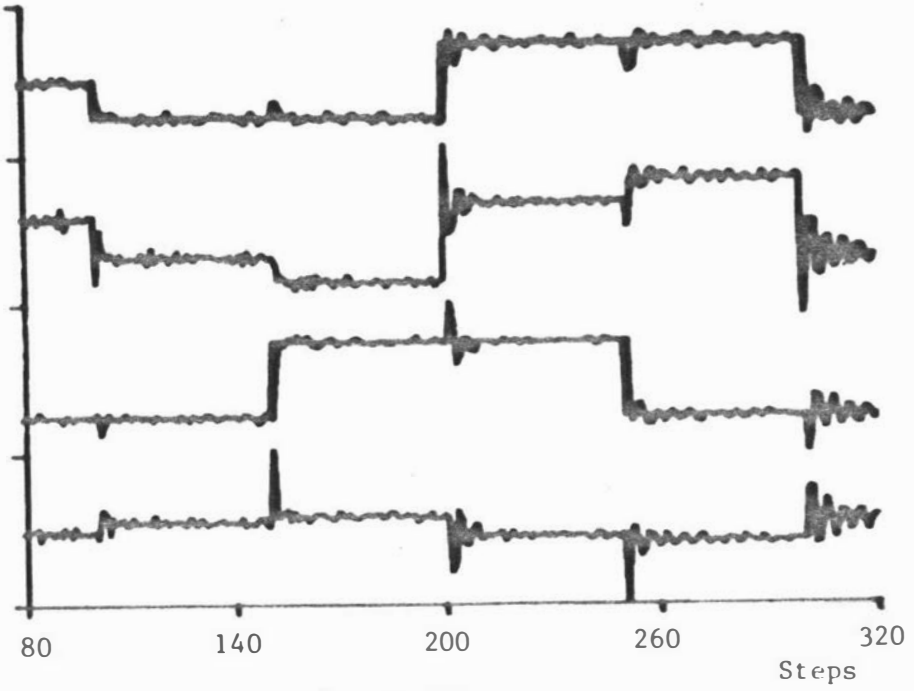


FIGURE 4.29 (a)

Controller Comparison

Controller #2 $A_m = 1$ $A_0 = 1$

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

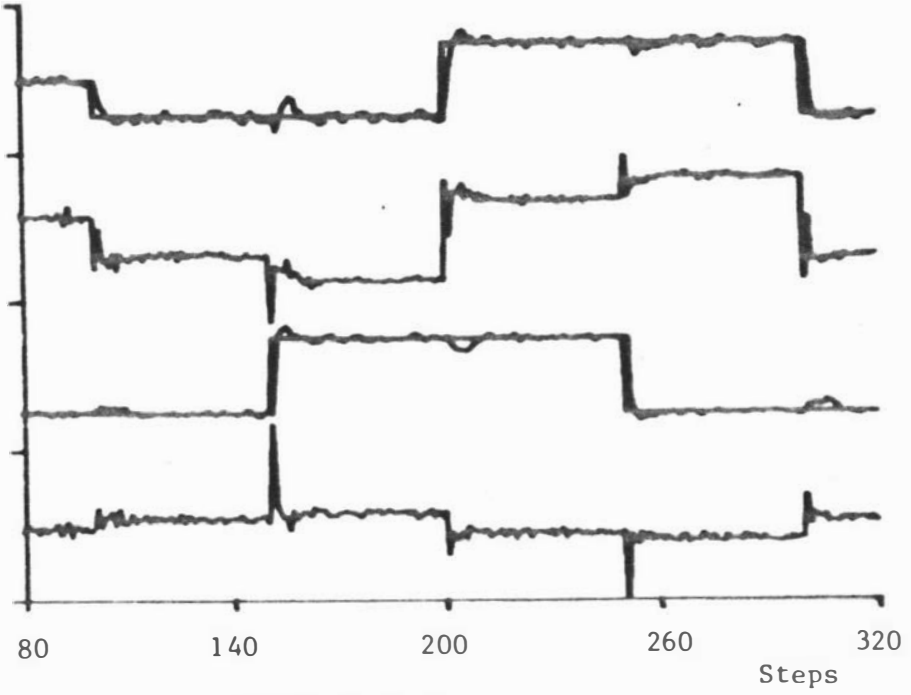


FIGURE 4.29 (b)

Controller Comparison

Controller #3 $A_m = 1$ $A_0 = 1$

TABLE 4.5
Comparison of controllers #2 and #3

Controller #2

A_0	Loop #	$A_m=1$		$A_m=1-0.5q^{-1}$	
		MSE(y)	MSE(u)	MSE(y)	MSE(u)
1	1	509.4	17137	34.83	238.2
	2	398.6	1520	14.32	35.41
A_m	1	-	-	49.73	292.5
	2	-	-	18.98	122.2
$1+q^{-1}+q^{-2}$	1	71.34	2219	33.29	230.6
	2	115.8	743.4	15.21	9.680

Controller #3

A_0	Loop #	$A_m=1$		$A_m=1-0.5q^{-1}$	
		MSE(y)	MSE(u)	MSE(y)	MSE(u)
1	1	55.11	328.8	57.87	41.08
	2	15.99	205.4	14.96	19.83
A_m	1	-	-	46.38	110.6
	2	-	-	18.22	95.31
$1+q^{-1}+q^{-2}$	1	26.42	27.50	24.76	16.14
	2	15.10	21.34	22.47	7.486

Once again the sensitivity of MSE(y) and MSE(u) to a few large excursions of y or u is apparent. The difference between the controllers is most clearly seen in the case where $\mathbf{A}_0=\mathbf{A}_m=1$. The difference is very marked in the graphical presentation (Figures 4.25-4.29). A critical change to the control input to a loop has to be modified in the next time step when the result of the control input change to the other loop is known. This sets up a sequence of changes which is fueled by either a set point change, or by attempts to eliminate noise effects. Suitable pole placement and the selection of an observer polynomial to give a conservative action produces a suitable controller, although obviously the behaviour of #3 is generally more acceptable.

4.2.6 The self-tuning property

The self-tuning principle [Astrom and Wittenmark, 1973] asserts that if the closed loop system converges, it will converge to that which would be obtained if the controller were designed with full a priori knowledge of the process parameters. In this case, if the delay k=1, the residual error sequence will converge to the system driving noise sequence. ([Astrom, 1970] deals with the case for k>1).

The comparison of the residual noise sequence and the driving noise sequence provides a test for self-tuning when both sequences can be determined, i.e. during simulations.

The choice of controller polynomials to decouple the loops involves certain approximations, which must jeopardise the self-tuning nature of the controller. In order to test whether the self-tuning property is retained, the process described below is simulated, and the driving noise and residual noise sequences compared.

Process simulated

$$\mathbf{A}y(t) = q^{-1}\mathbf{B}u(t) + \mathbf{C}e(t)$$

where

$$\mathbf{A} = \mathbf{I} + \begin{bmatrix} 0.786 & -0.030 \\ 0.000 & 0.755 \end{bmatrix} q^{-1} + \begin{bmatrix} -0.368 & 0.318 \\ -0.050 & 0.000 \end{bmatrix} q^{-2}$$

$$\mathbf{B} = \begin{bmatrix} 0.554 & 0.213 \\ 0.213 & 0.213 \end{bmatrix} q^{-1} + \begin{bmatrix} 0.323 & 0.081 \\ 0.081 & 0.081 \end{bmatrix} q^{-2}$$

$$\mathbf{C} = \mathbf{I} + \begin{bmatrix} -0.5 & 0.0 \\ 0.0 & -0.2 \end{bmatrix} q^{-1}$$

and the model polynomial is:

$$\mathbf{A}_m = \mathbf{I} + \begin{bmatrix} -0.3 & 0.0 \\ 0.0 & -0.3 \end{bmatrix} q^{-1}$$

Noise

e1(t) and e2(t) are both uniformly distributed random sequences on the interval $-0.25<e(t)<0.25$, i.e. with variance .0104.

Calculation of residuals

The residuals are calculated as y(t)-r(t), where r(t) is the required value of y(t), given by:

$$\mathbf{A}_m r(t) = \mathbf{E}w(t)$$

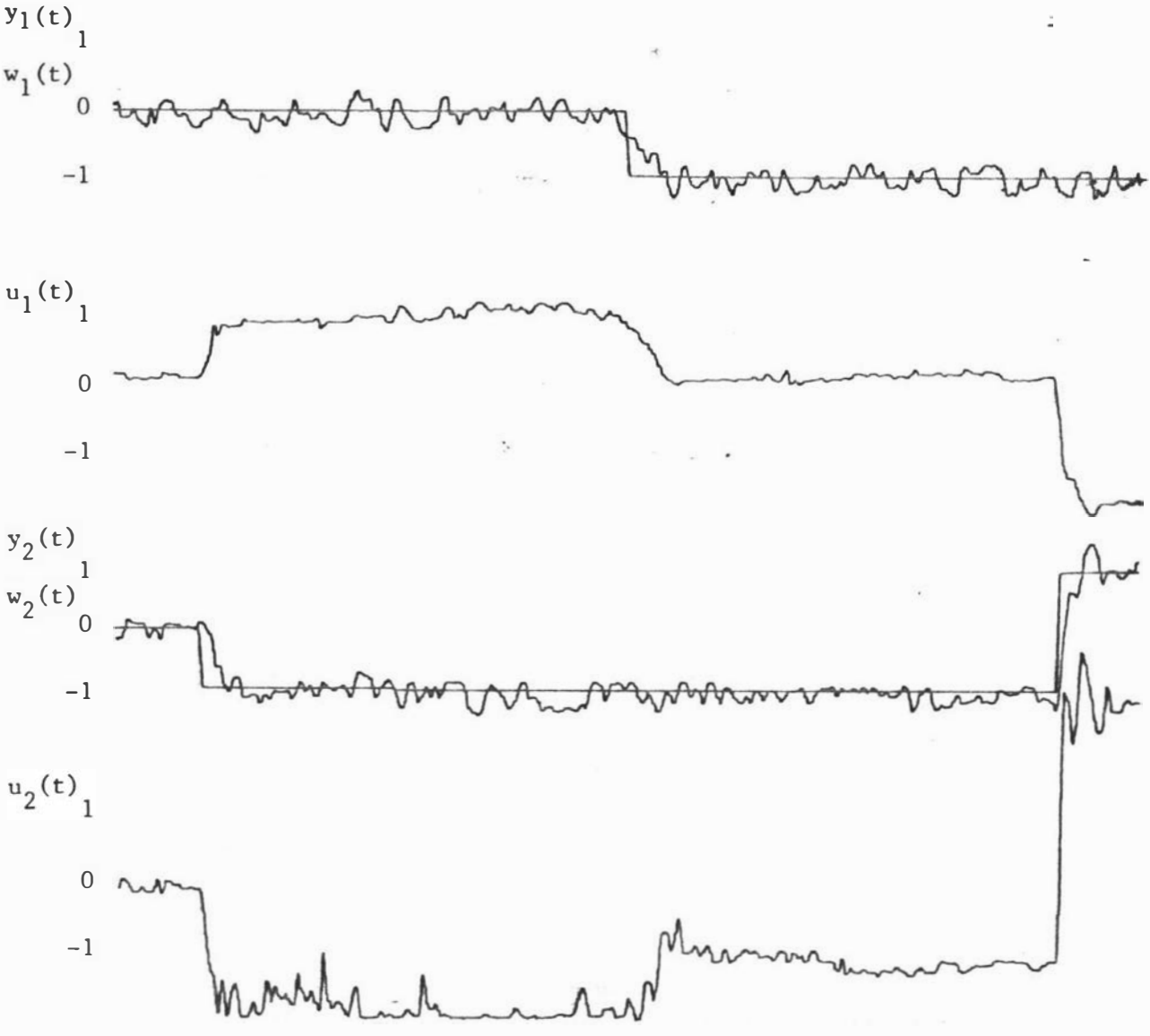


FIGURE 4.30

Check of self-tuning property

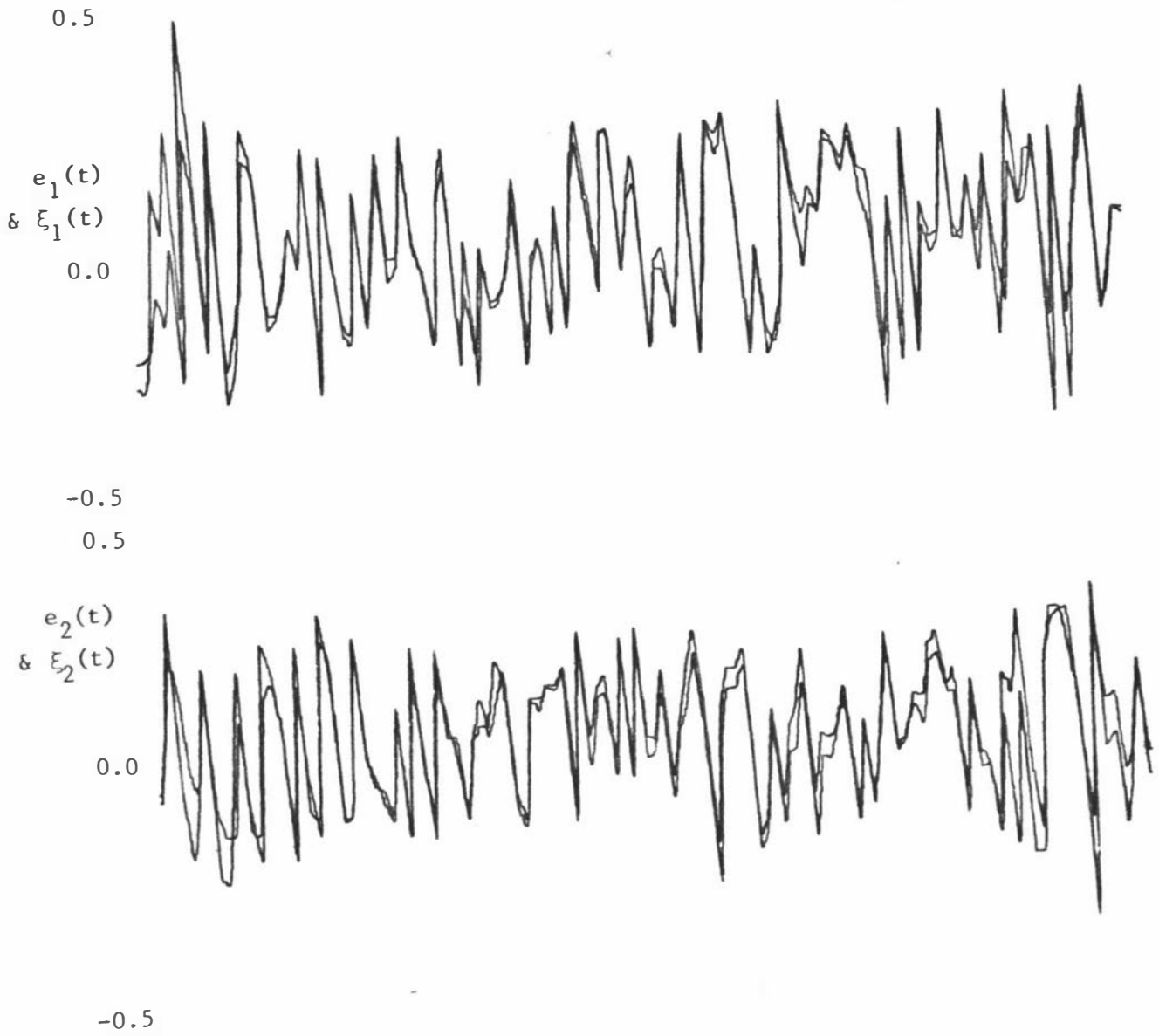


FIGURE 4.31

Check of self-tuning property

Noise and residual sequences superimposed

where w(t) is the set point, Am is the model polynomial and E the observer polynomial described in Chapter 2. The residuals will be correct in the steady state, but will not have the true values following set-point changes.

The results of the simulations are shown in Figures 4.30 and 4.31. The first shows the behaviour of the system being simulated. Note that the run includes set point changes, when it is to be expected that the noise and residuals will be different. Also note that only 320 steps are simulated, after which time it is unlikely that the parameters will have completely converged.

The noise and residuals are shown superimposed on the same axes in Figure 4.31. For much of the run, the noise and residuals for the two loops are indistinguishable, showing that the self-tuning principle is inviolate. As expected, the two traces diverge following set-point changes, but rapidly converge following this perturbation.

In all the simulations that have been described in this chapter, the residuals corresponding to those depicted in Figure 4.31 have been used as a measure for comparison of the controllers.

4.2.7 Full Pole-Placing Algorithm

A new full multivariable pole-placing controller has been described in section 2.3, and this controller may be used as a standard against which to compare the other algorithms which have been proposed, and for which results have been presented in this chapter.

The closed loop behaviour of a system incorporating the full multivariable pole placing controller has been examined by simulating the following process with the controller:

$$\mathbf{A}y(t) = q^{-k}\mathbf{B}u(t) + \mathbf{C}e(t)$$

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} -0.786 & 0.030 \\ 0.000 & -0.755 \end{bmatrix} q^{-1} + \begin{bmatrix} 0.368 & 0.300 \\ 0.000 & 0.050 \end{bmatrix} q^{-2}$$

$$\mathbf{B} = 0\, q^{-1} + \begin{bmatrix} 0.500 & 0.200 \\ 0.100 & 0.080 \end{bmatrix} q^{-2} + \begin{bmatrix} 0.300 & 0.080 \\ 0.040 & 0.200 \end{bmatrix} q^{-3}$$

$$\mathbf{C} = \begin{bmatrix} 1 - 0.5q^{-1} & 0 \\ 0 & 1 - 0.5q^{-1} \end{bmatrix}$$

B has roots given by q=5.127 or q=0.273, and consequently the process is non-minimum phase. The controller is required to place the closed-loop system poles at locations consistent with the model polynomial:

$$\mathbf{A}_m = \begin{bmatrix} 1 - 0.5q^{-1} & 0 \\ 0 & 1 - 0.5q^{-1} \end{bmatrix}$$

Set point attainment and elimination of offset error has been described in section 2.3. In order to make a comparison with controller #3, simulations for which have already been presented, the system is simulated with various noise levels. The results of the first simulation are presented in Figure 4.32, (a) and (b),

for the full multivariable controller and controller #3 respectively. White noise with a variance of 8.3E-5 is present on both loops. The results for the two controllers are very similar, the most noticeable difference being the interaction effects between the two loops. In the case of the full multivariable controller, both loops exhibit an interaction effect which persists until steady state conditions can be asserted, although this effect is small and quite acceptable, the controller being designed primarily as a regulator. Controller #3 on the other hand has been designed for rapid elimination of disturbance effects, and this is apparent in its behaviour. The transient behaviour of loop 1 is also marginally better for controller #3.

The variance of the noise is increased to 0.01 for both loops, and simulation results presented in Figure 4.33 (a) for the full multivariable controller, and Figure 4.33 (b) for controller #3. The difference in the interaction effects is not now so apparent, with no clear advantage to either controller. The vertical axis scale may be expanded to permit closer examination of the system behaviour, and Figure 4.34 (a) and (c) present the responses for loops 1 and 2 with the full multivariable controller, while Figure 4.34 (b) and (d) show the responses for loops 1 and 2 with controller #3. Only very minor differences are apparent at distinctive points on the graphs.

The process control inputs produced by controller #3 display no increase in activity over those of the full multivariable controller, even though controller #3 was designed with a minimum-variance-like control action to remove interactions. The interactions are not of a minor nature, as evidenced by the considerable controller effort required for their elimination.

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

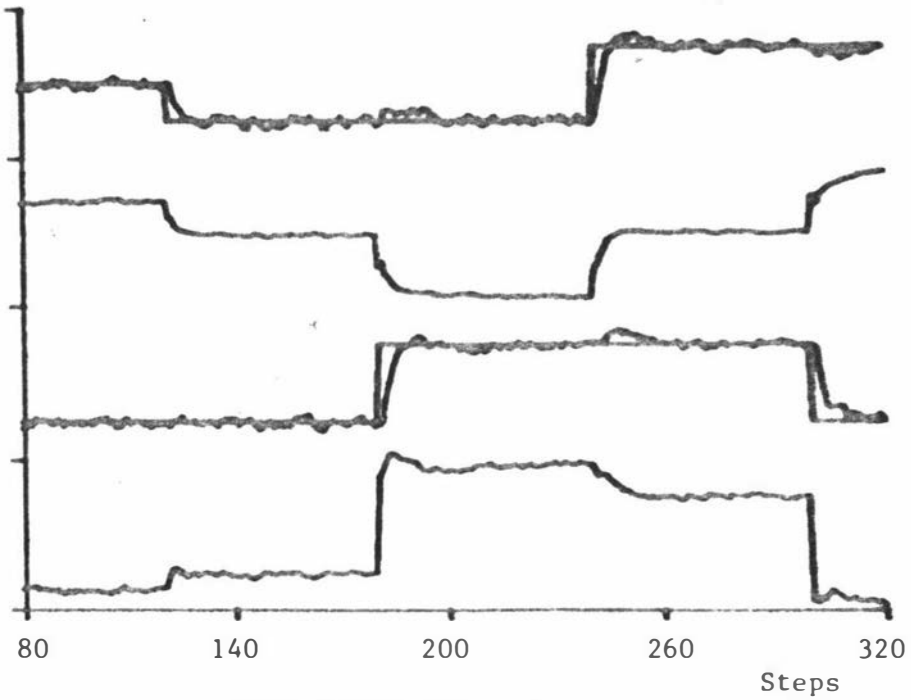


FIGURE 4.32 (a)

Full multivariable controller

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

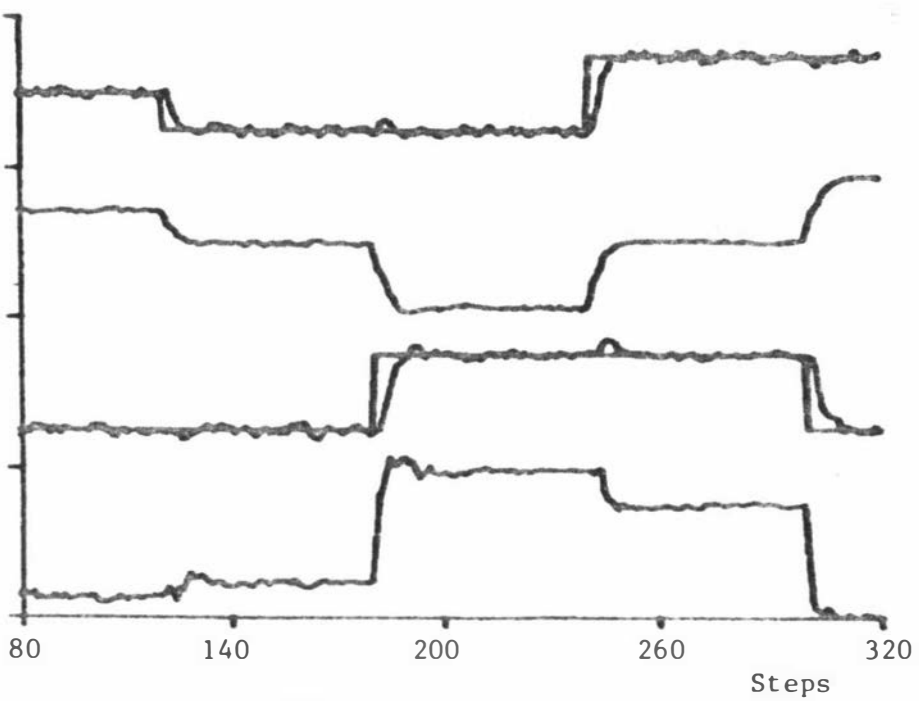


FIGURE 4.32 (b)

Controller #3.

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(D)$	-6	10
$u_2(t)$	-2	14

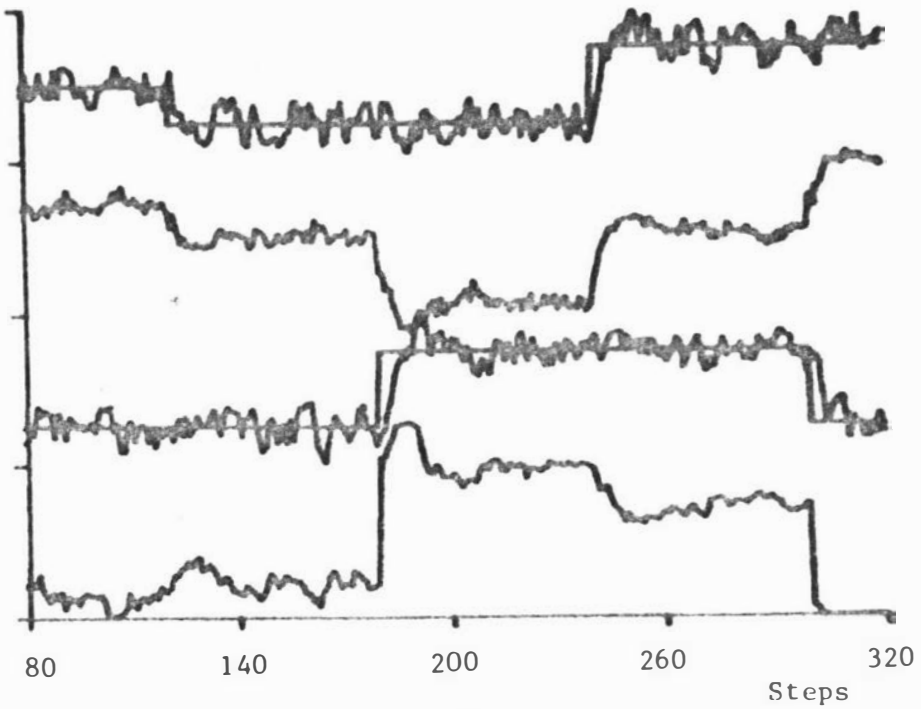


FIGURE 4.33 (a)

Full multivariable controller

Var	Min	Max
$y_1(t)$	-14	2
$w_1(t)$	-14	2
$u_1(t)$	-10	6
$y_2(t)$	-6	10
$w_2(t)$	-6	10
$u_2(t)$	-2	14

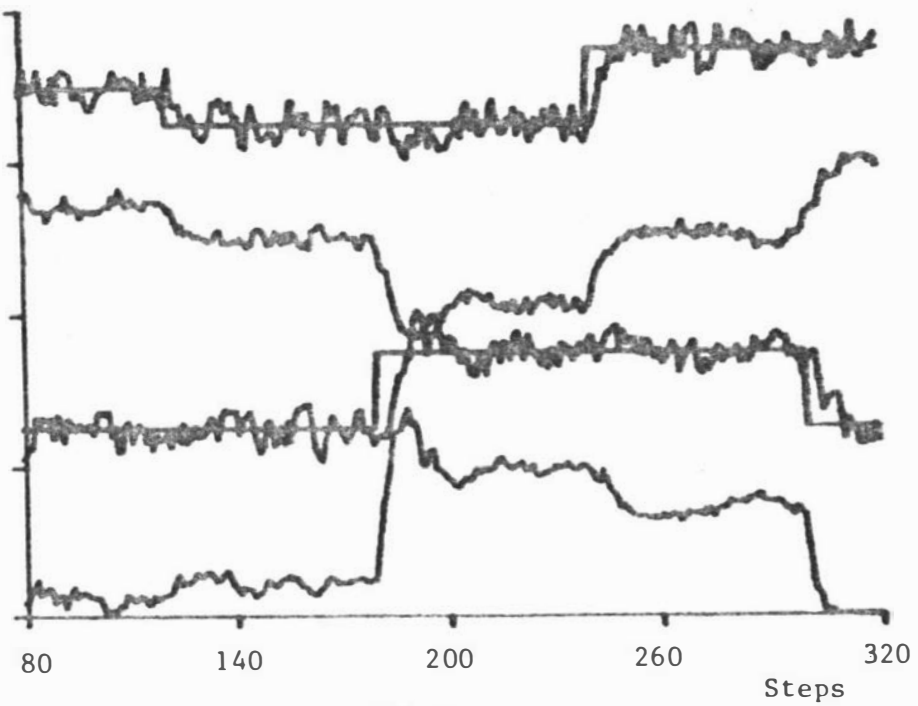


FIGURE 4.33 (b)

Controller #3

Var	Min	Max
$y_1(t)$	-2	2
$w_1(t)$	-2	2

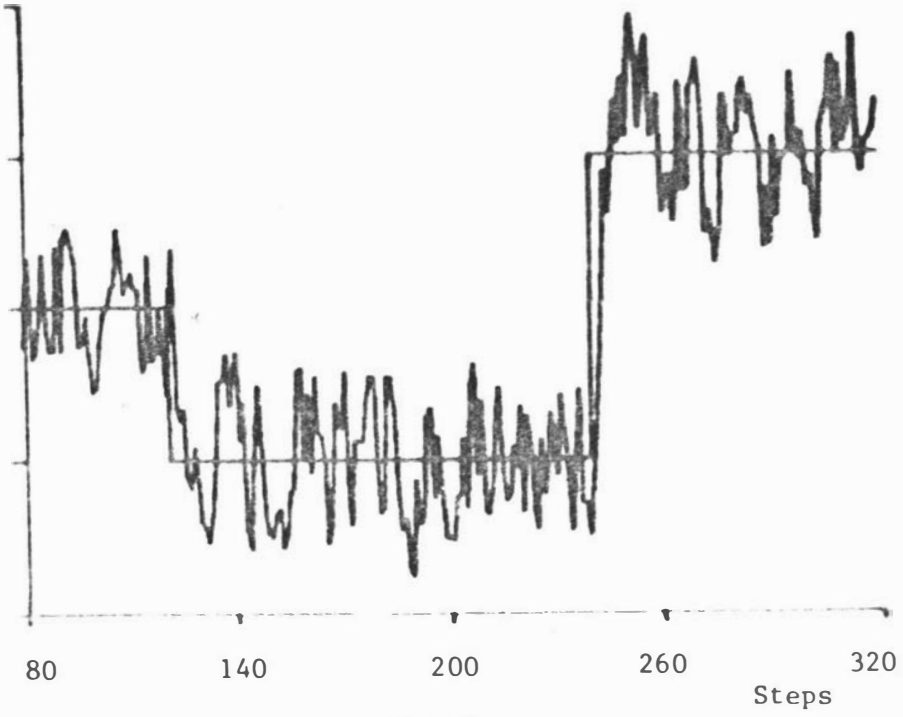


FIGURE 4.34 (a)

Full multivariable controller Loop 1

Var	Min	Max
$y_1(t)$	-2	2
$w_1(t)$	-2	2

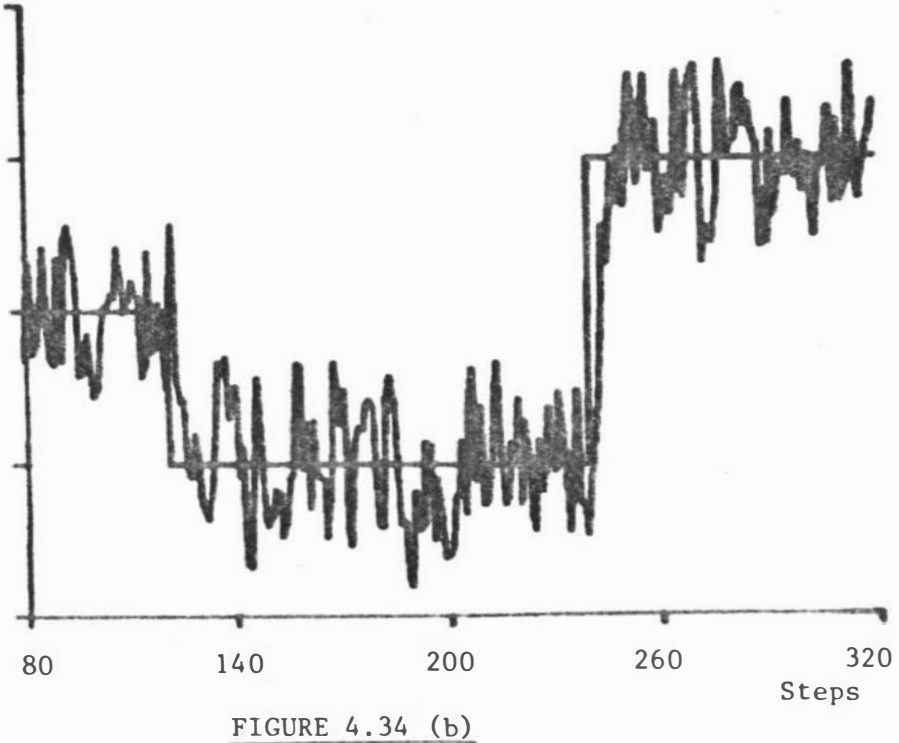


FIGURE 4.34 (b)

Controller #3 Loop 1

Var	Min	Max
$y_2(t)$	-2	2
$w_2(t)$	-2	2

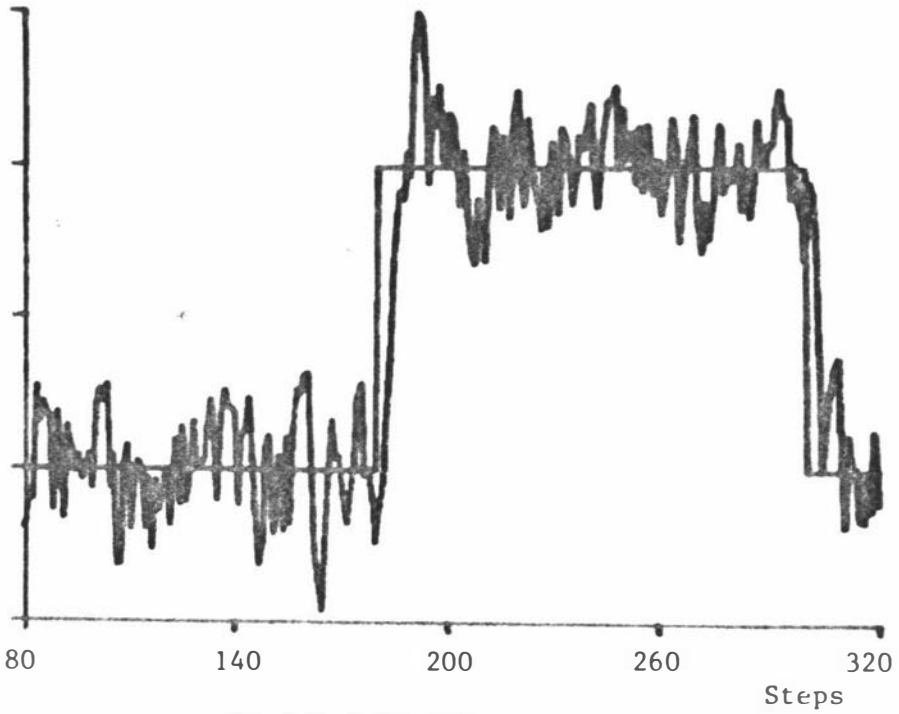


FIGURE 4.34 (c)

Full multivariable controller Loop 2

Var	Min	Max
$y_2(t)$	-2	2
$w_2(t)$	-2	2

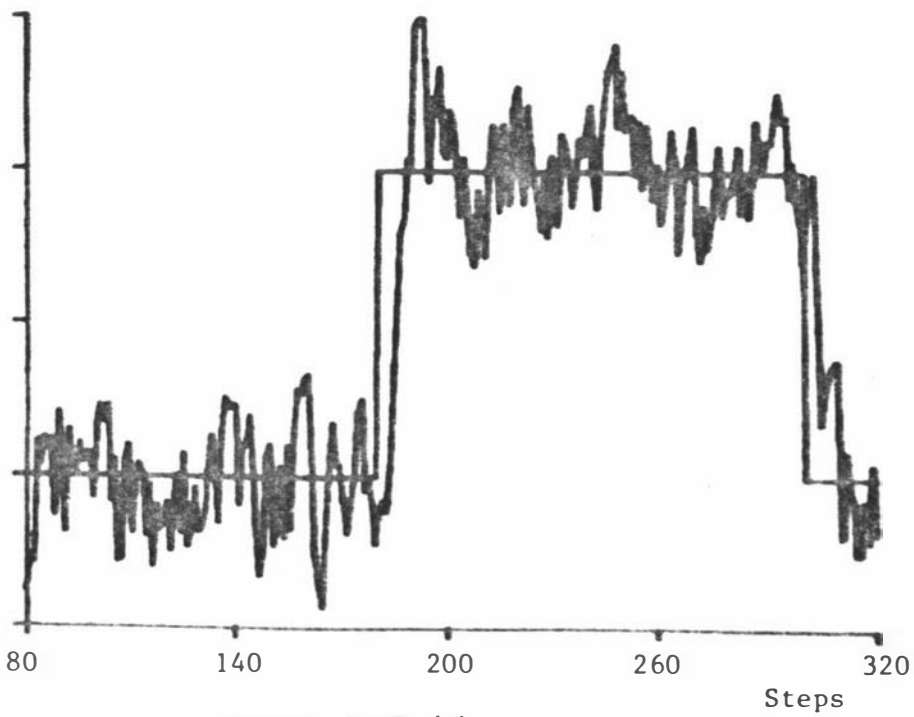


FIGURE 4.34 (d)

Controller #3 Loop 2

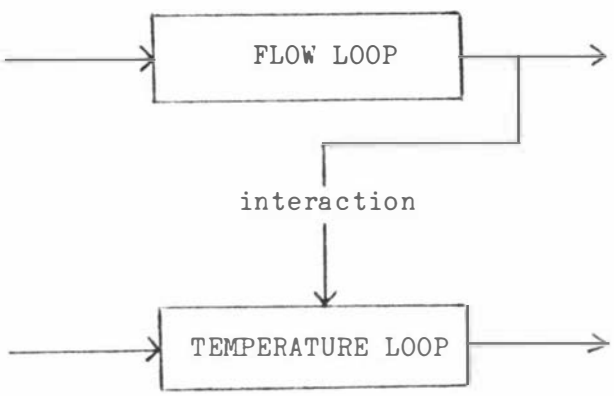
4.3 CONTROL OF HEAT EXCHANGER

The self-tuning controller has been applied to control of a commercial heat exchanger. The use of this process was motivated by the use at the Dairy Research Institute of a heat exchanger to control temperature and viscosity of milk concentrate being passed from an evaporator to a spray drier. Single loop PID controllers have been applied to the same problem, and the results contrasted with those for self-tuning. [Hesketh and Chaplin, 1981].

Simulations were used to establish suitable controller parameters without the need for tedious and expensive runs on the actual plant, before applying the controller on-line.

4.3.1 Heat exchanger process

The process has been described in Section 3.3.2, but its description may be summarised for convenience. There are two loops to control:



The only major interaction is between flow and temperature. The inclusion of other interactions, requiring the identification of parameters close to zero, can degrade rather than improve the

performance. The presence of time delays means that it is immaterial whether the input to the flow loop, or the output from the flow loop is used to anticipate future changes in the temperature. There is an interaction of temperature on flow, due to viscosity changes, but this interaction is small, and the rapid response of the flow to input changes makes the flow loop an easy loop to control, the interaction effects being removed quite satisfactorily by the feedback controller.

4.3.2 Simulation of heat exchanger

Software is required for manual operation of the equipment for startup procedures, and to enable transducers to be tested and calibrated, and variable values scaled to ensure that the model is well conditioned. The self-tuning controller programs contain the necessary routines for least-squares identification, although these estimates will be biassed because the noise terms are ignored.

If the identification procedures are extended to include a noise model [Panuska, 1980], and the manual operation replaced by random perturbations, a model for the process may be identified. The rate of response of loop 1 required a 2 second sampling interval, but for loop 2 this was unacceptable because the time delay would have required too many sample intervals for its representation, and the process gain would have been ridiculously low, representing hardly any change in each sampling interval. The following simplified model for the heat exchanger was obtained:

Sampling interval = 2 seconds for loop 1, 4 seconds for loop 2.

$$\mathbf{A}y(t) = q^{-1}\mathbf{B}u(t) + e(t)$$

where the matrices may be partitioned to give:

$$\mathbf{A}_{11} = 1+0.1665q^{-1}$$

$$\mathbf{A}_{12} = 0$$

$$\mathbf{A}_{21} = -0.0256q^{-1}+0.847q^{-2}$$

$$\mathbf{A}_{22} = 1-1.5598q^{-1}+0.6056q^{-2}$$

$$\mathbf{B}_{11} = 0.4552-2.2159q^{-1}$$

$$\mathbf{B}_{22} = -0.0200+0.0064q^{-1}-0.0035q^{-2}-0.0084q^{-3}-0.0052q^{-4}$$

$$\mathbf{B}_{12} = \mathbf{B}_{21} = 0$$

Noise: $E\{e(t)e^{T}(t)\} = \text{diag}(0.0133,0.0133)$.

The form of $\mathbf{B}_{22}$ arises because these polynomials were identified with leading zeros to cater for time delays. However, the temperature loop responds reasonably quickly to changes in flow rate, and following a change to the steam valve, shows a small immediate response, with a larger effect after 20 seconds. Terms in the noise model were considered small enough to ignore for the simulation. Although the temperature loop had an apparent 20 second delay, the above model gave the best fit.

4.3.3 Simulation results

Figure 4.35 presents the simulation results for the following controller:

Loop 1: $\mathbf{A}_m = 1-0.8q^{-1}$ $\mathbf{A}_O = 1+q^{-1}$ Delay k=1

Loop 2: $\mathbf{A}_m = 1-0.3q^{-1}$ $\mathbf{A}_O = 1+\ldots+q^{-4}$ Delay k=4

Loop 1 requires two simulations steps for every one step of loop 2. A noticeable feature of the simulation is the vigourous controller action required by loop 2. It has been shown in previous simulations that this may be reduced by appropriate selection of the poles or by changing the sampling interval of the loop in question. Consequently, simulations were performed for different sampling intervals and model polynomials for loop 2. Table 4.6 summarises the performance, presenting values of MSE(y) and MSE(u) (x1000). As the performance of the loop 1 controller was satisfactory, the same controller was used for all runs. Figures 4.35 (g), (h) and (i) show the results of the remaining heat exchanger simulations.

Var	Min	Max
1	-14	2
3	-14	2
5	-10	6
2	-6	10
4	-6	10
6	-8	56

Loop 1: 2 secs
$A_m = 1 - 0.8q^{-1}$

Loop 2: 4 secs
$A_m = 1 - 0.3q^{-1}$

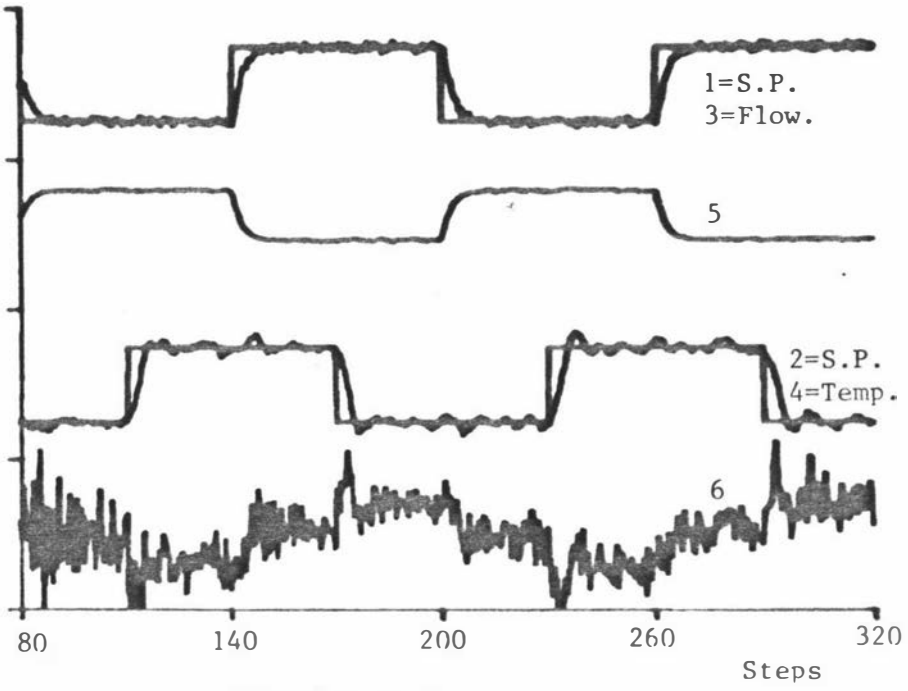


FIGURE 4.35 (a)

Heat exchanger simulation

Var	Min	Max
b_0	-14	2
b_1	-10	6
a_1	-6	10
d	-2	14

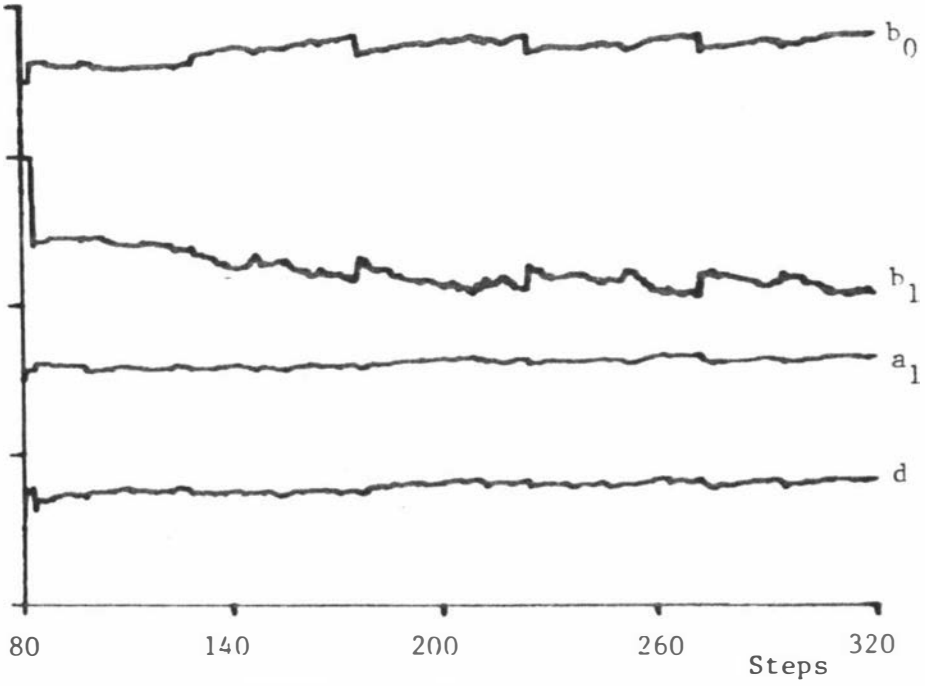


FIGURE 4.35 (b)

Parameter estimates Loop 1

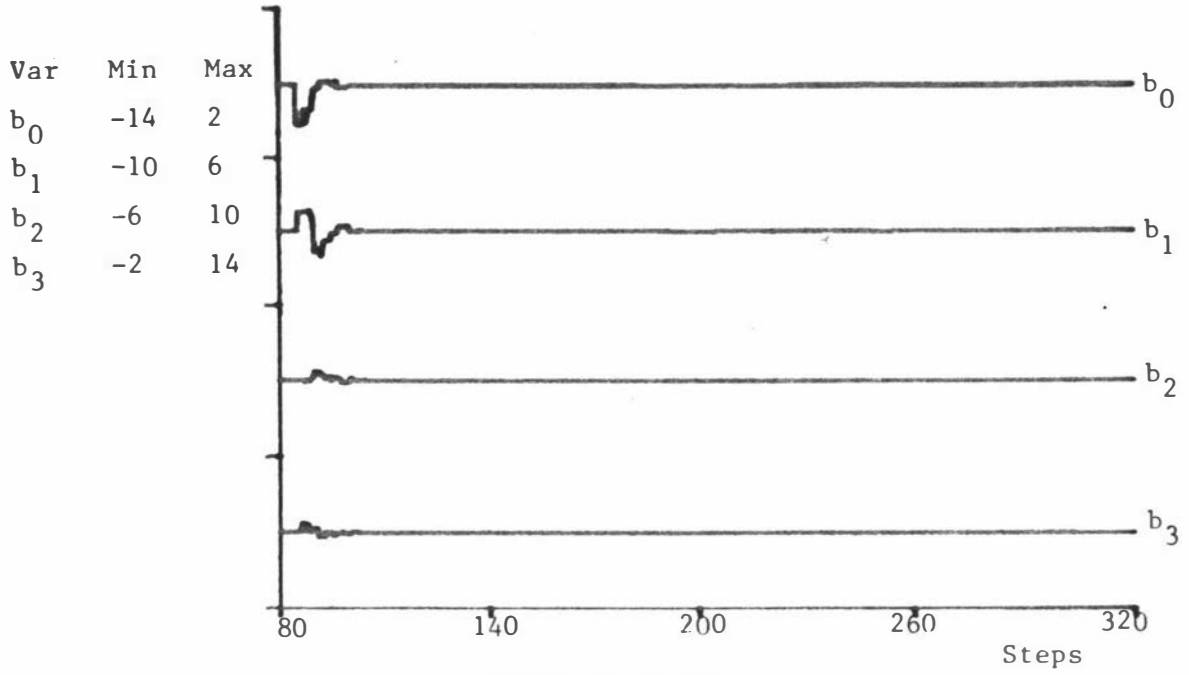


FIGURE 4.35 (c)

Parameter estimates Loop 2

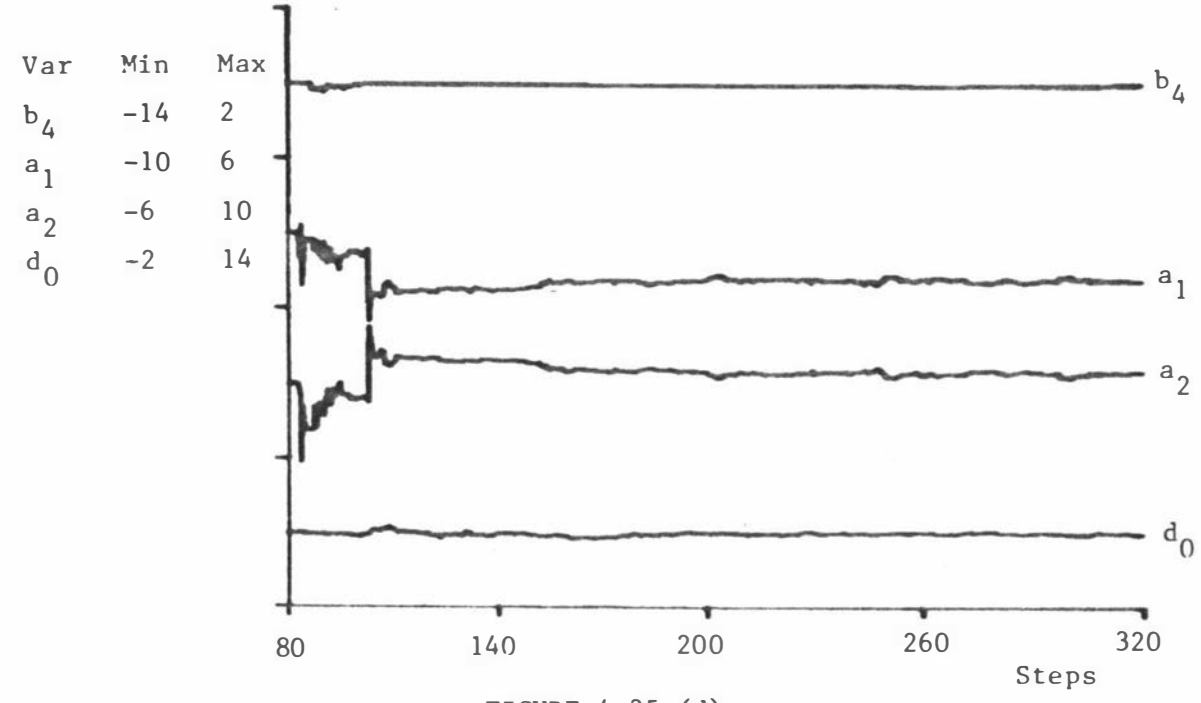


FIGURE 4.35 (d)

Parameter estimates Loop 2

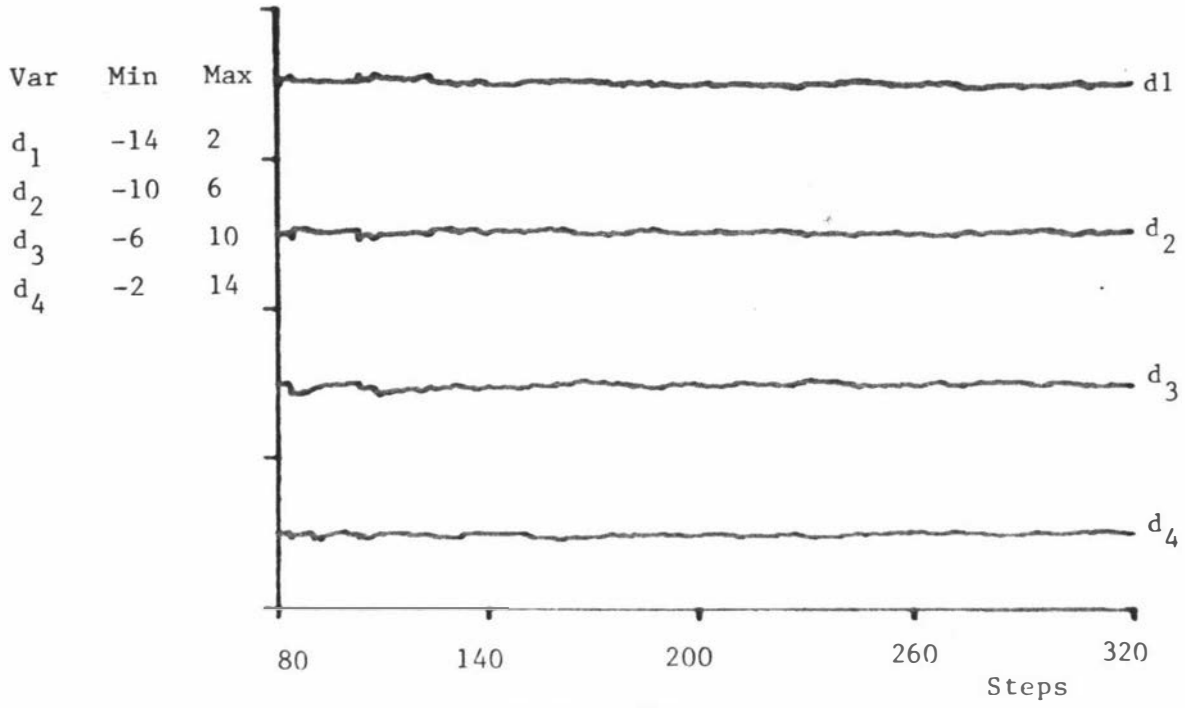


FIGURE 4.35(e)

Parameter estimates Loop 2

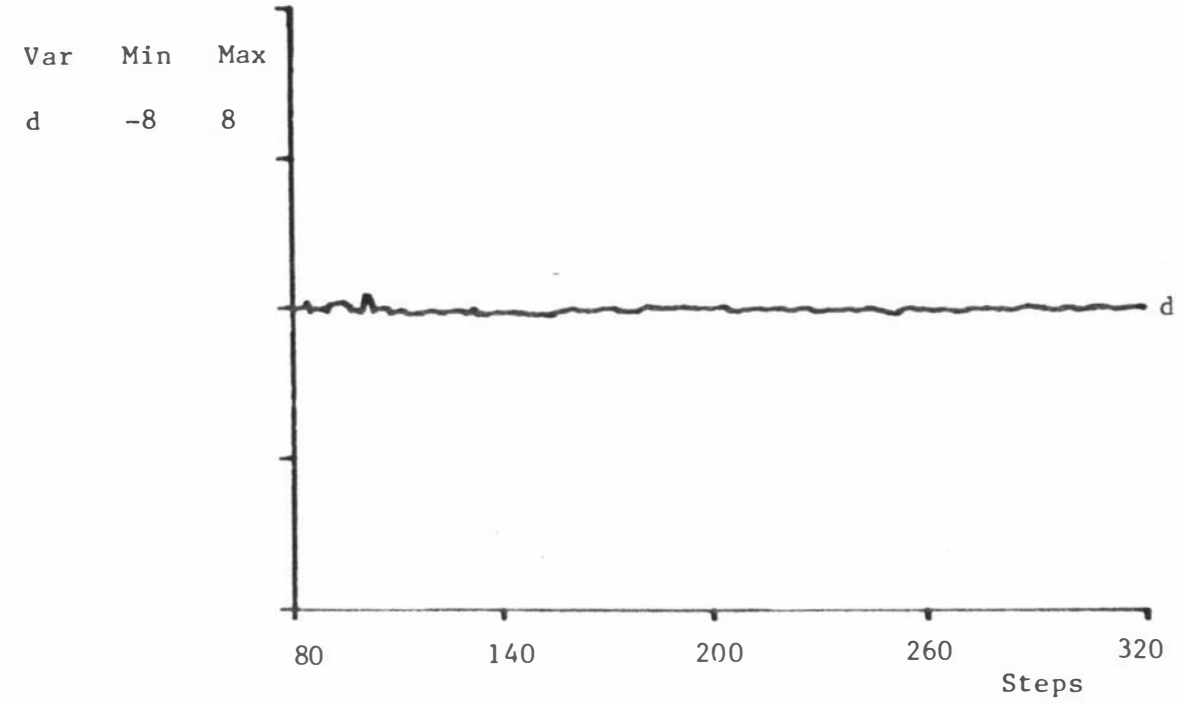


FIGURE 4.35(f)

Parameter estimates Loop 2

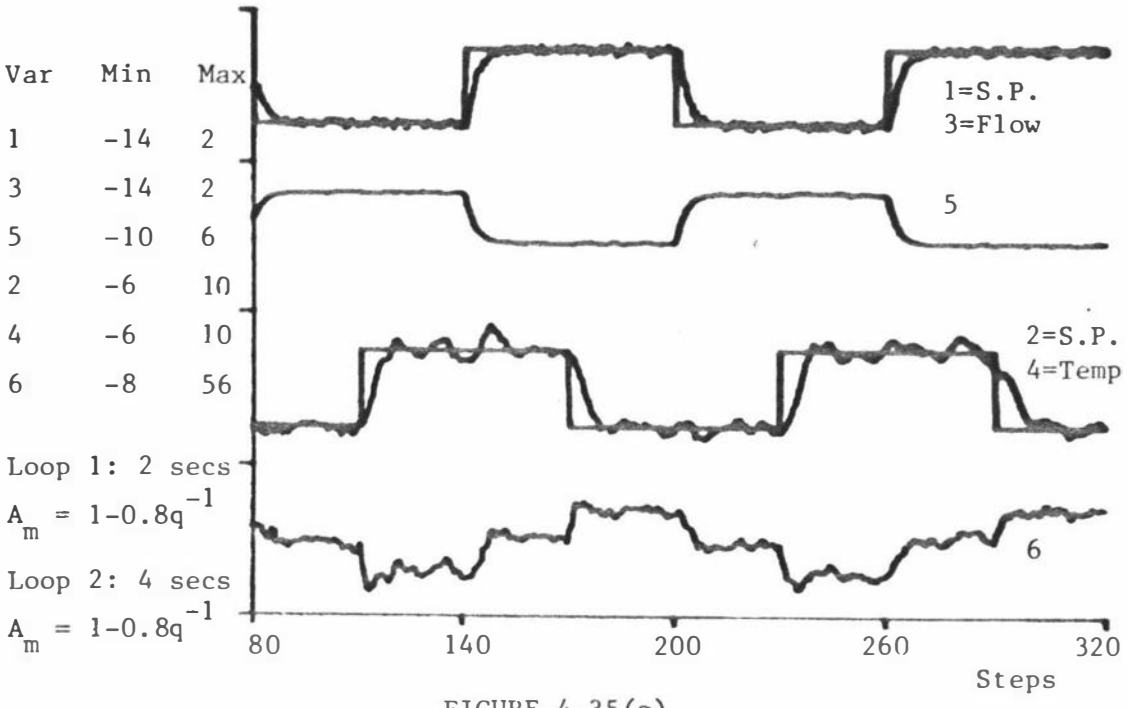


FIGURE 4.35(g)

Heat Exchanger Simulation

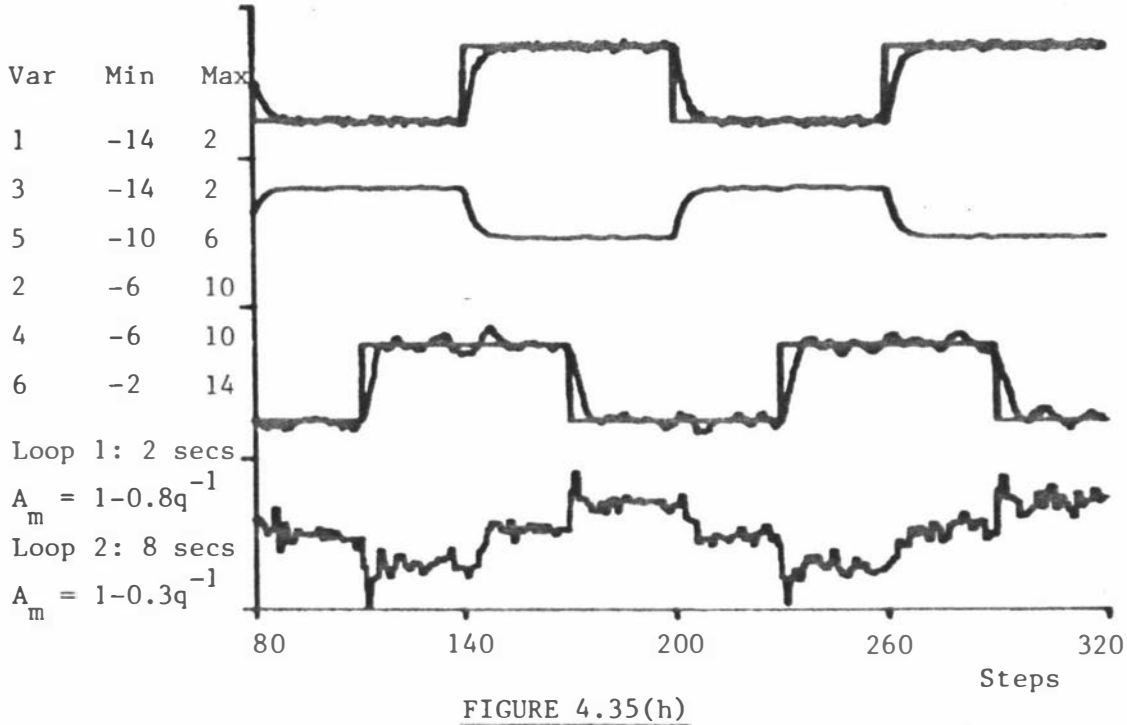


FIGURE 4.35(h)

Heat Exchanger Simulation

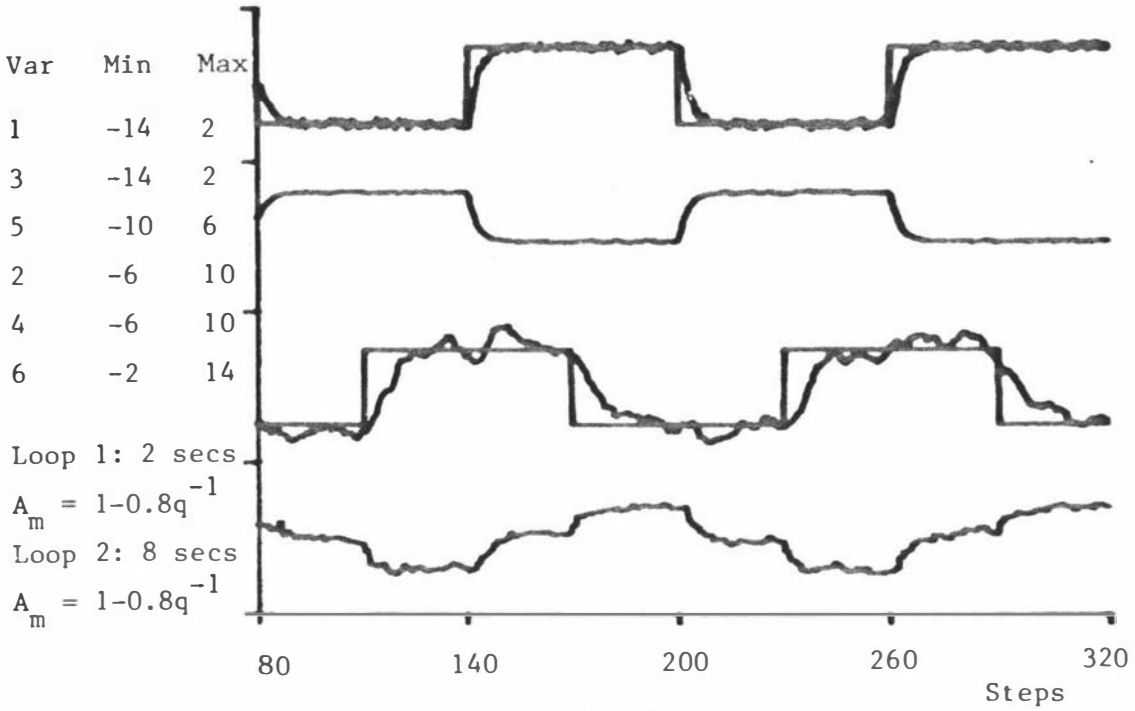


FIGURE 4.35(i)

Heat Exchanger Simulation

TABLE 4.6

Heat exchanger simulations

	control interval (4 secs)		control interval (8secs)	
	MSE(y)	MSE(u)	MSE(y)	MSE(u)
$Am=1-0.3q^{-1}$	8.004	1.387	7.545	1.927
	28.46	13883	16.73	1218
$Am=1-0.8q^{-1}$	12.91	3.559	8.335	1.354
	49.16	231.6	29.04	164.9

4.3.4 Results of on-line experiments

On-line experiments with the heat exchanger were conducted for two different situations.

(a) No perturbation to the process.

The flow rate of the heated water through the first heat exchanger was held constant at 15 litres/minute. The set points to the two loops were varied (flow rate and temperature of heated water from second heat exchanger). The loops are subject to significant interaction effects. The temperature loop is subject to disturbances from the action of the float trap which removes condensate from the first heat exchanger.

(b) Perturbing the process.

The flow rate of the heated water through the first heat exchanger was held constant at 15 litres/minute for the first 1400 seconds. The flow rate was then adjusted suddenly to 10 litres/minute, which changes the parameters of the process model

and also changes the time delay of the temperature control loop. The interaction effects and unmeasured disturbances are still present.

Summary of Controller Parameters.

(i) Self-tuning controller Loop 1 (flow rate)

Process model order	1
time delay	2 sample intervals
Observer polynomial (E)	$1+q^{-1}$
Desired model	$1-0.8q^{-1}$
Sampling interval	2 seconds
Interactions	0

(ii) Self-tuning controller Loop 2 (temperature)

Process model order	2
Time delay	4 sample intervals
Observer polynomial (E)	$1+q^{-1}+q^{-2}+q^{-3}+q^{-4}+q^{-5}+q^{-6}$
Desired model	$1-0.5q^{-1}$
Sampling interval	4 seconds
Interactions	1 (flow rate)

Notes:

The process model is obviously of reduced order (see for example Baur and Isermann [1977] who identified a third order model for a single similar heat exchanger). The time delay can be variable for the form of the model identified, the value given above merely placing an upper bound on the value used in the model. The observer polynomial is conservatively chosen to give a gradual

change following a set point alteration. The desired model is first order (less than the order of the process model). The zeroth order model would also be permissible, higher order coefficients being assumed zero as appropriate. Although this is a multivariable controller, different sampling rates on different loops are permitted.

TABLE 4.7

Performance summary

	MSE(y)	
Process	.00441	Loop 1
Perturbation	.03316	Loop 2
No Process	.00481	Loop 1
Perturbation	.04226	Loop 2

Graphical analysis

Figure 4.36 shows the behaviour of the system with a change to the process at 1400 seconds. Control is unaffected by the change. The price paid for this performance is the vigorous manipulation of the steam valve. See also Appendix B.

Figure 4.37 shows performance in the absence of any change to the process. The effects of changes to the flow rate are apparent, as are the disturbance effects arising from float trap operation. Note how the self-tuning controller reduces the effects of the interaction. Another much smaller interaction effect is also demonstrated - the effect of changing viscosity on the flow as the temperature is changed. It is not necessary to include this interaction in the model as the effect is small, and the controller deals with the fast responding loop easily.

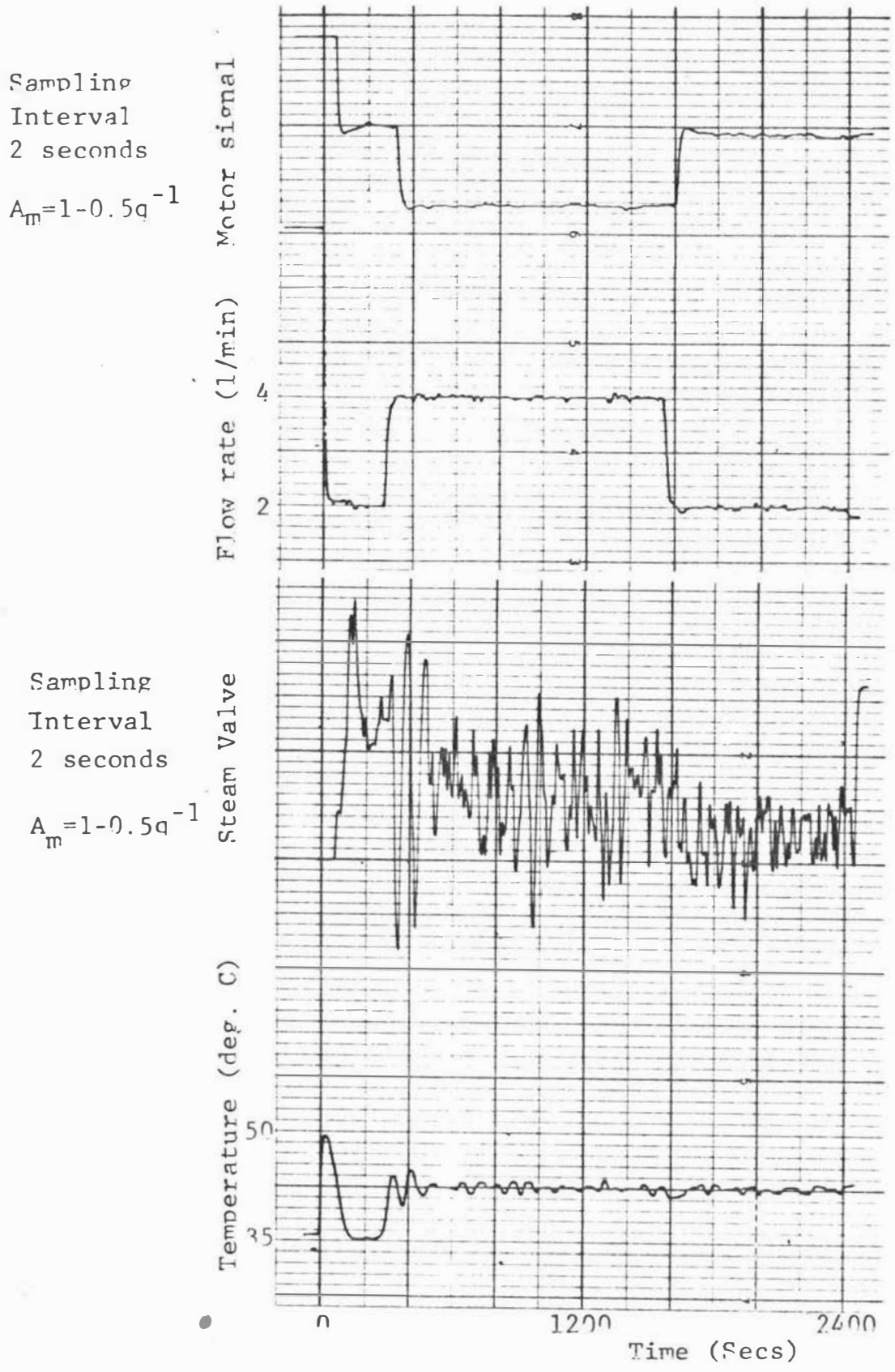


Figure 4.36

Heat Exchanger Control

Process perturbation at 1400 secs.

Sampling interval 2 seconds

$A_m = 1 - 0.8q^{-1}$

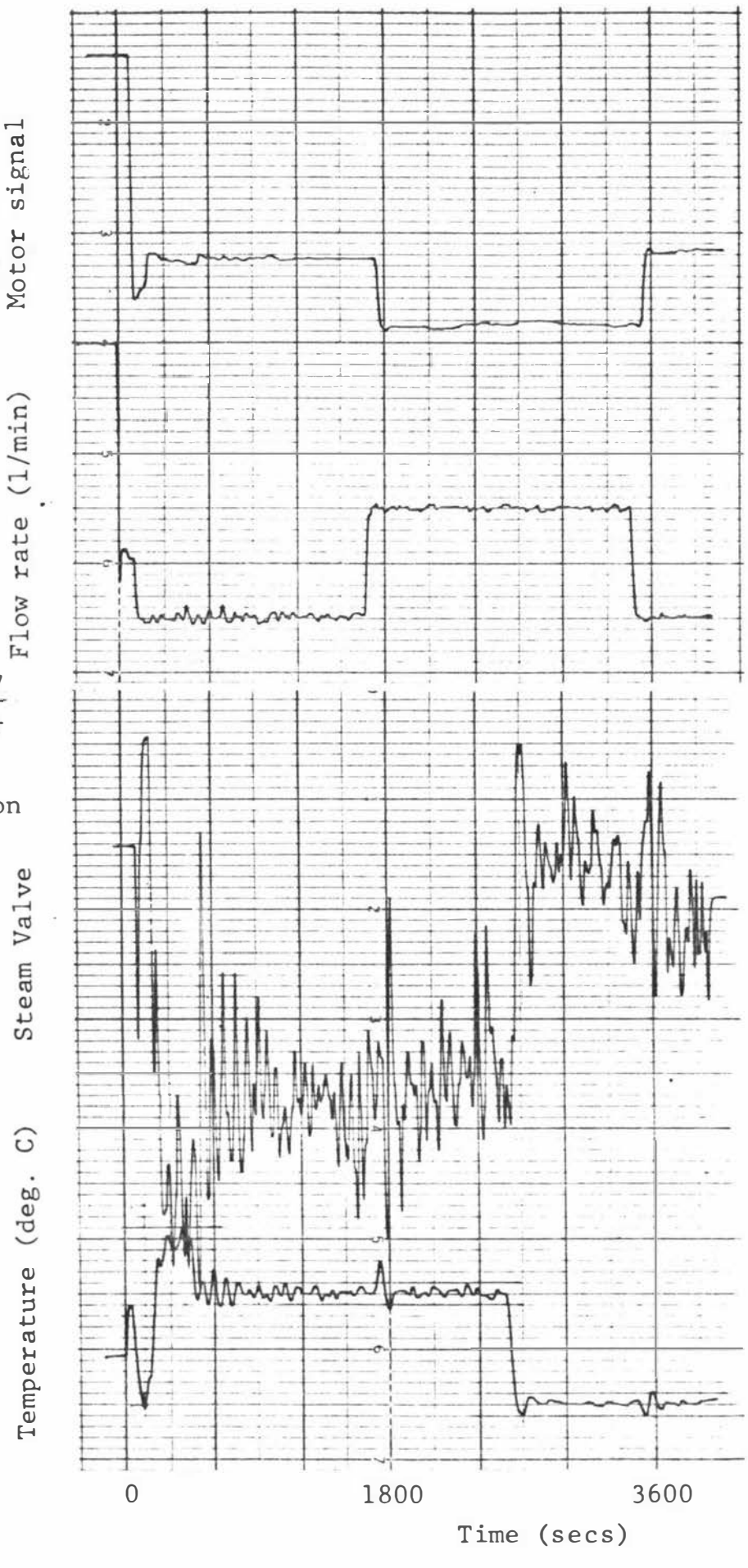


Figure 4.37
Heat Exchanger Control
No perturbation

In order to reduce the action of the control input to loop 2, changes were made to the control interval and to the model polynomial. Figures 4.38 - 4.40 depict graphically the results when these changes are made. All other controller parameters remain unchanged.

(i) Loop 2

Model order	2
Time delay	2 sample intervals
Observer polynomial	$1+\ldots+q^{-6}$
Desired model	$1-0.5q^{-1}$
sampling interval	20 seconds

(ii) Loop 2

Model order	2
Time delay	4 sample intervals
Observer polynomial	$1+\ldots+q^{-6}$
Desired Model	$1-0.84q^{-1}$
Sampling interval	4 seconds

(iii) Loop 2 PROCESS CHANGE AT TIME 1800

Model order	2
Time delay	3 sample intervals
Observer Polynomial	$1+\ldots+q^{-6}$
Desired model	$1-0.4q^{-1}$
Sampling interval	6 seconds

Sampling
Interval
2 seconds

$A_m = 1 - 0.8q^{-1}$

Sampling
interval
20 seconds

$A_m = 1 - 0.5q^{-1}$

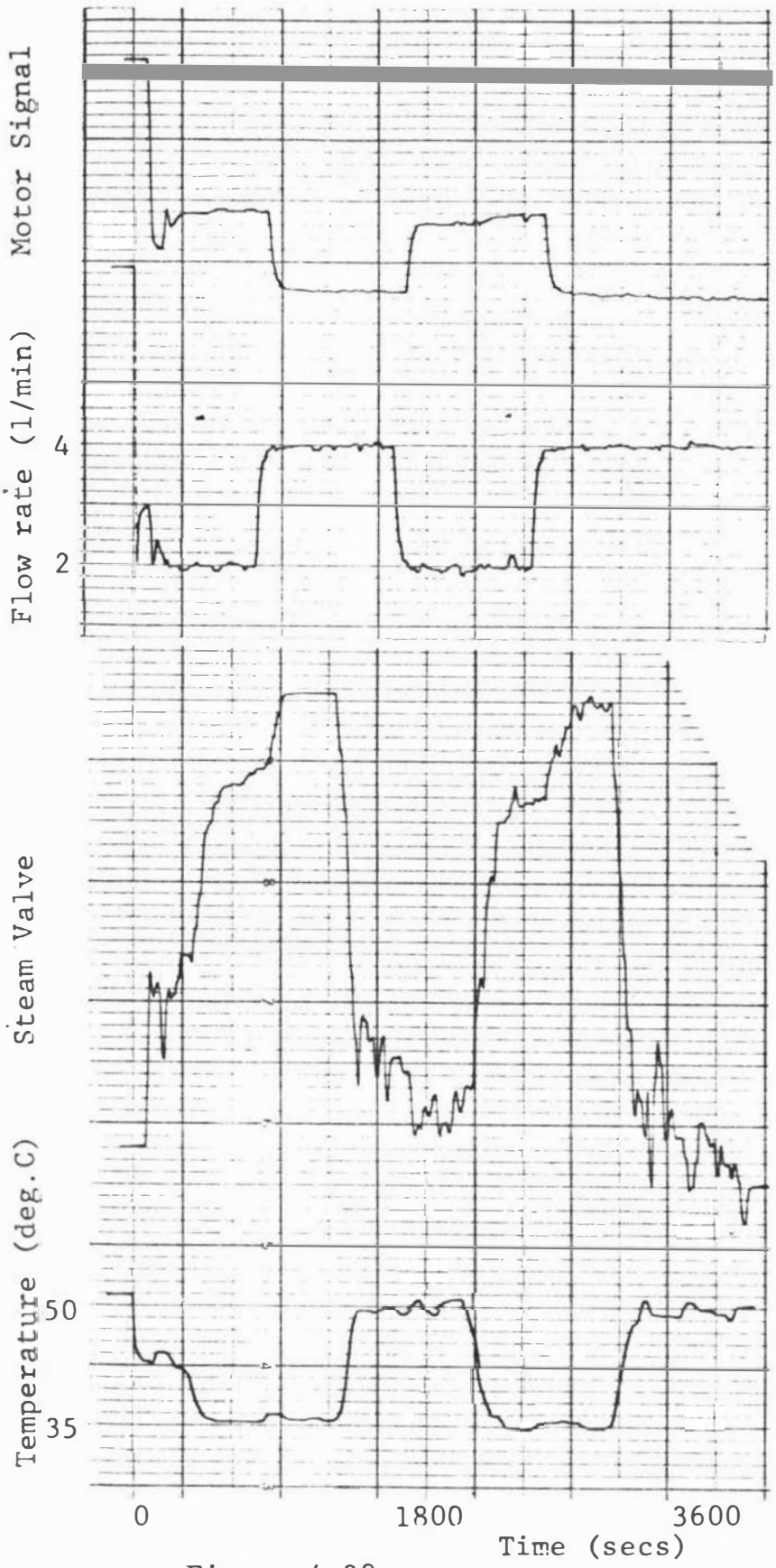


Figure 4.38

Heat Exchanger Control

Sampling interval 2 seconds

$A_m = 1 - .8q^{-1}$

Sampling interval 4 seconds

$A_m = 1 - .84q^{-1}$

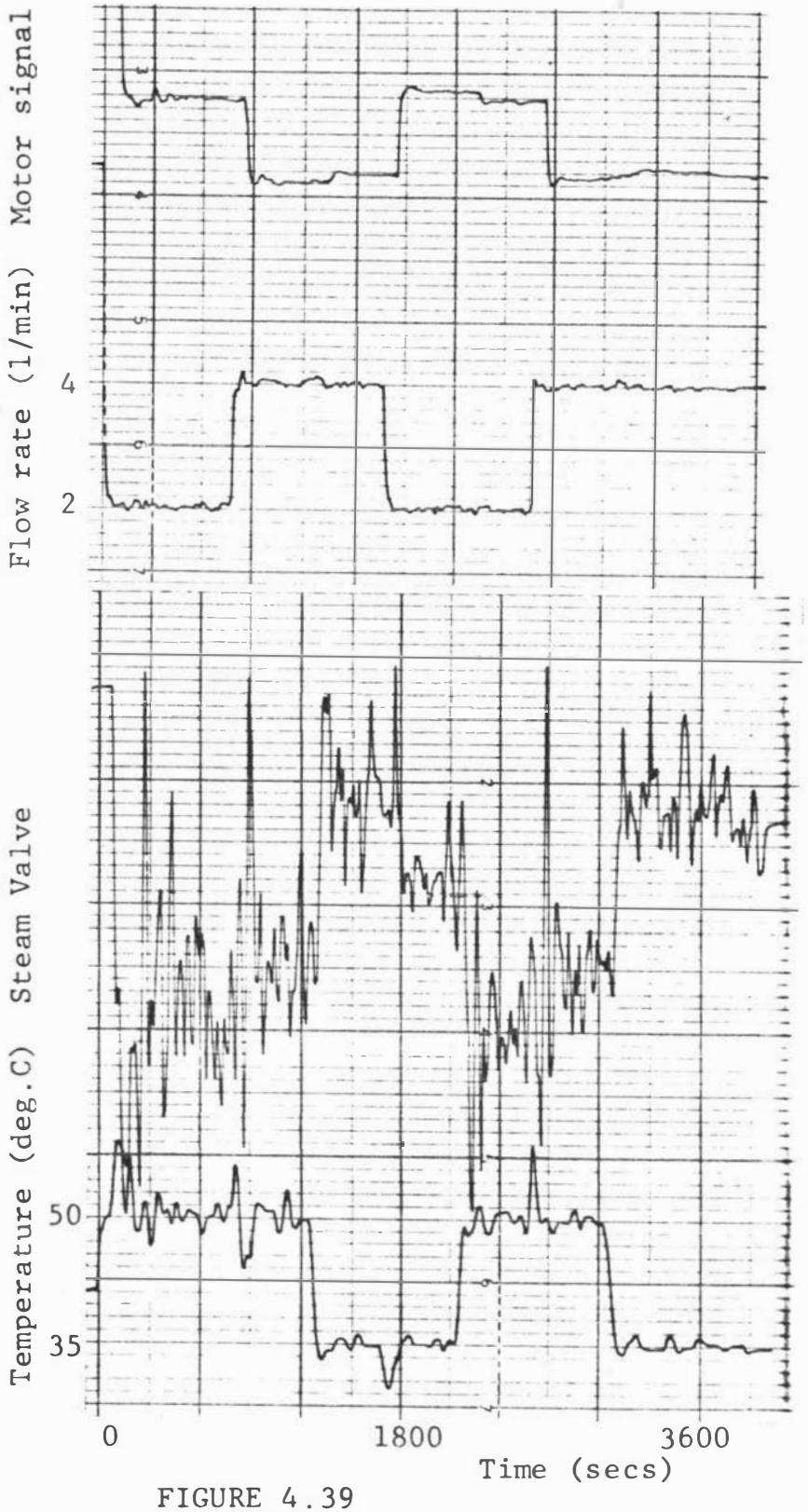


FIGURE 4.39

Heat Exchanger Control

Sampling interval 2 seconds

$A_m = 1 - 0.8q^{-1}$

Sampling interval 6 seconds

$A_m = 1 - 0.4q^{-1}$

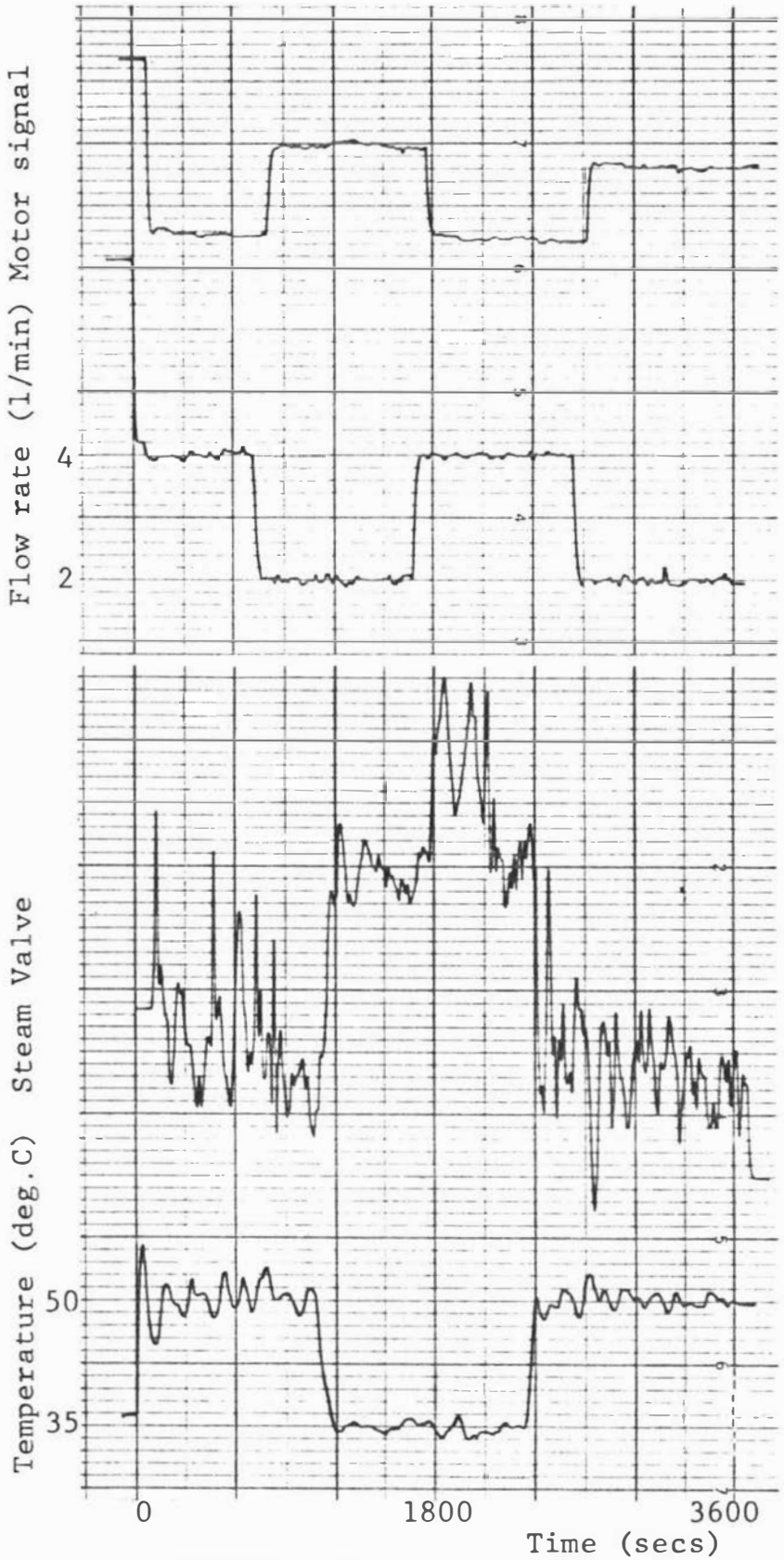


FIGURE 4.40

Heat Exchanger Control

Process perturbation at 1800 secs.

Comments

The model polynomial of loop 1 was chosen to give a relatively slow response on this loop so that the disturbance effect on loop 2 would be reduced over any single control interval.

No significant improvement to the system behaviour was found when the model order was increased. Consequently, the minimum order that gave good control was selected.

There is no pure time delay, and consequently the controller will behave well for a range of values. It is important to specify a time delay long enough, or there is a marked deterioration in the controller performance. However, because of the use of leading zeros in the polynomials, specifying too long a delay can make the model over-parameterised.

As a general policy, the desired model poles should be chosen so that the controller has a fast response, and can eliminate noise effects. The slower controller (Figure 4.39) was unable to respond quickly enough to reduce the variability.

A shorter sampling interval generally produces better control. However, making the sampling interval too short can greatly increase the controller effort, to a degree that might not be acceptable. This is due to a low process gain, or small values in the **B** polynomial. While the pole placing control law is chosen to reduce variability of u(t), a certain amount of controller effort is necessary to maintain the self-tuning property and ensure that the residual errors are not too large.

4.3.5 Online implicit control

The implicit controller described in Chapter 2 was also used for online experiments with the heat exchanger. The use of this controller highlighted the objections to the use of "optimal control" methods for such controllers. The choice of the weighting polynomials to produce a satisfactory system response is not a trivial task. For a non-linear and badly behaved process such as this, considerable experimentation is required before a satisfactory response may be achieved.

Figure 4.41 shows the behaviour of the heat exchanger for the following controller:

Loop 1

$$\mathbf{P} = 1-0.9q^{-1} \qquad \mathbf{Q} = 3-2.7q^{-1} \qquad \mathbf{R} = 0.1$$

Loop 2

$$\mathbf{P} = 2.8-1.4q^{-1} \qquad \mathbf{Q} = 2.5 \qquad \mathbf{R} = 0.1 \qquad \mathbf{S} = 0.1$$

The output from loop 1 (flow rate) interacts with loop 2, and is treated as a disturbance to this loop.

The control of flow rate is good, as expected, because this variable is easy to control. The response following a set-point change is governed by the polynomial $(1-0.9q^{-1})$. This polynomial is present in the closed loop transfer function as a "dominant pole". Of interest is the very clear change in the control input to the flow loop to counteract the interaction due to the viscosity change of the water as the temperature changes.

The control of the temperature loop is not very good, and some tendency to oscillation is apparent. Despite considerable experimentation, the temperature control could not be improved,

demonstrating the difficulty of choosing the appropriate weighting polynomials to achieve suitable system response. The results for this implicit controller contrast with those for the pole placing controller, which is more robust, and where the actual closed loop behaviour is specified. The importance of relating the weighting polynomials to some other control objective such as pole placement [Allidina and Hughes, 1980] becomes apparent, although this nullifies the computational advantages enjoyed by the implicit algorithm.

$P=1-0.9q^{-1}$
$Q=3-2.7q^{-1}$
$R=0.1$

$P=2.8-1.4q^{-1}$
$Q=2.5$
$R=0.1$
$S=0.1$

FIGURE 4.41

Heat Exchanger Control

Implicit Controller

Chapter 5
DISCUSSION & CONCLUSIONS

5.0 INTRODUCTION

The study of adaptive control systems has matured, and the results presented in Chapter 4 for implicit and explicit control schemes shows the suitability of self-tuning methods for control of multivariable systems.

The controller based on implicit identification of the controller parameters provides a system which is economical in terms of programming requirements and execution time, while the concept of identifying the controller parameters directly is conceptually attractive. However, the need to formulate the model so that it is expressed in terms of the controller parameters places some limitations on this method at present, while the explicit schemes offer greater freedom to choose the control law and formulate the algorithm.

Pole placing control algorithms are robust [Astrom, 1980], and attractive for certain processes because the controller parameters specified bear a direct relationship to the response which may be expected from the controlled system. Optimal or sub-optimal methods have the advantage of providing a criterion by which their performance may be evaluated. The one step minimisation of the cost function is sub-optimal, and some processes may benefit from a full solution of the Ricatti equation combined with an explicit identification process. The extra computational effort involved nullifies some of the attractiveness of the method.

5.1 Discussion of results

In Section 4.1 it is demonstrated that for one of the implicit control schemes which can at present be employed, suitable adjustment of the weighting polynomials of the cost function can result in good disturbance rejection, and that this may be used to achieve satisfactory control of a multivariable process using methods proposed in Chapter 2. The transient interactions may be greatly reduced, and generally the decoupling of the loops is good, while the steady state interactions may be eliminated without the use of an integrator. A similar technique has been used to ensure steady state set-point attainment [Allidina and Hughes, 1980].

The ability of the explicit pole placing controllers to control a variety of processes has been demonstrated in Section 4.2.1. In summary, unstable processes may be stabilised (provided the process response may be kept bounded while a set of model parameters is established initially), and processes with time delays, non-minimum phase or all pass properties are readily controlled.

For processes which are unstable or show integral behaviour, it would be necessary in practice to ensure the boundedness of the response by providing a good set of initial parameter estimates. This raises the question of sensitivity of the controller to changes in the parameters during the initial re-estimation. For some initial estimates, the process output remains bounded, while for others control is poor for several steps at this stage. This remains an open question.

The mechanism used for disturbance rejection in the pole placing scheme is motivated by that used for the implicit algorithm. If the process is non-minimum phase, then the **B** polynomial cannot be inverted as required, and some approximation is required. The zeros of this polynomial may be ignored, and only the steady

state effect considered, which leads to transient errors, which however may generally be small and short-lived. Alternatively, the single non-minimum phase input may be replaced by several minimum phase inputs with different delays, which degrade performance in a quantifiable way using techniques described in Astrom [1970]. Any deviation from the desired trajectory or set-point introduced by these approximations is corrected by the controller, such correction being influenced by the selected pole positions, so that the excessive controller action of minimum variance controllers may be avoided.

The simulation results presented in Figure 4.5 make it clear that the position of the selected poles do not necessarily influence the controller effort required to effect control. As a result of self-tuning, the residual errors converge to the system driving noise [Wellstead, Edmunds, Prager and Zanker, 1979], regardless of the pole positioning. If the process has very low gain, then violent control action will still be required. This can be alleviated by increasing the process gain, which is achieved for discrete processes by increasing the sampling interval.

The simulations of multivariable processes (Section 4.2.3) also make use of process models that have been reported elsewhere [Borisson, 1979; Koivo, 1980]. The model of the headbox of a paper machine shows that when a very strong interaction exists, as evidenced by the large off-diagonal term in the **B** matrix, ([Process 1], Section 4.2.3), vigourous control action may be necessary to maintain the self-tuning property as described above. In this case the simulations show that the vigourous action on loop 2 may be reduced by suitable placement of the poles of loop 1, reducing the excursions of $u_1(t)$ and hence reducing the interaction effect on loop 2. The simulations by Koivo [1980] show very large excursions of $u_2(t)$, and while no claim is made here regarding the appropriateness of the pole placing controller for this particular process, the beneficial

effects which may result from its use for this and other processes are apparent.

[Process 2] is non-minimum phase and unstable. The simulation of this process presented by Koivo [1980] shows that his $y_2(t)$ was not maintained at the set point of 0 following a change to the set point of loop 1, and no change is made to the set-point of loop 2. However, having simulated the process, it is obvious that it is a severe test of an adaptive controller. Figure 4.18 shows that complete control of both loops of this process is possible with the proposed pole-placing algorithm.

Changes to the parameters which describe a process may be accommodated by self-tuning controllers, although the degradation in the control during re-estimation of the parameters depends on the particular process being considered, as discussed above. The self-tuning controller is best suited to processes whose parameters may change, but not instantaneously. In the case of a non-linear process for example, a gradual change may be ensured by a slow set-point change, allowing some re-linearisation about each point.

The simplifying assumption that $\mathbf{B}_0$ is a diagonal matrix suggested by Borisson [1979] avoids the inversion of the $\mathbf{H}_0$ matrix when the control inputs are calculated, (section 2.1.9). This slight computational advantage has to be weighed against the degradation in control performance this produces for some processes [Chapter 4, Figures 4.25-4.29]. For some processes such as the heat exchanger used for the on-line tests of the controllers, the inputs to each loop may have very little effect on any other loop, and this simplification is valid. However, where a loop input may have a large interaction effect on another loop, this interaction has to be countered by adjustments to the control inputs at subsequent time intervals, each adjustment introducing a further perturbation of its own, with the result illustrated in Figure 4.29 (a), which should be compared with Figure 4.29 (b),

when the additional inversion was performed.

To verify that a controller has the self-tuning property, it is possible to compare the residual noise sequence and the driving noise sequence if a simulation is being performed, to check whether the two converge as the self-tuning principle asserts they should [Wellstead et al, 1979 (a) and (b)]. This is particularly necessary when some approximations have to be made in the implementation of the controller. It has been shown in Figure 4.31 (Section 4.2.6) that reasonable convergence of the two sequences is achieved by the proposed pole placing controller for the multivariable process considered, i.e. that these noise sequences converge for each loop. As has been described above, it is this adherence to the self-tuning property that may exact a penalty when a noisy process with low gain is being controlled.

A self-tuning controller has also been proposed based on full multivariable pole-placement (Section 2.3). The method draws on techniques for pole-placement given by Wolovich [1974] for continuous processes expressed in state-space form. It has been shown how the discrete polynomial matrices may be rearranged into a form which allows advantage to be taken of the same matrix properties which make possible the pole-placement for the continuous process models. For the scalar problem, this method yields results which are identical to those obtained from the polynomial approach [Appendix A].

While more computation is required for the full multivariable solution than for the other controllers which have been proposed, the computational burden is not necessarily prohibitive for process control, particularly if a controller were dedicated to a particular process enabling the elimination of many of the general features incorporated into the programs used for this research. Advantage may be taken of more powerful microprocessor systems. To efficiently tailor a controller for a process, a program generator can be of great assistance, and this need not

of course be limited to generating programs only for self-tuning. Such a program generator is the subject of continuing work. Interestingly, more compact code has been generated for the matrix than for the polynomial solution.

Very little difference has been demonstrated between the closed loop behaviour resulting from use of the full multivariable controller, and that resulting from use of the simplified controller. Even though for the simpler controller a minimum-variance-like technique was used to eliminate the effects of interactions between loops, there was no accompanying increase in the controller action required for the process that was simulated.

The system comprising two commercial heat exchangers provided the opportunity to demonstrate many of the properties described above for a realistic process. As with many of the simulations, similar processes have been used elsewhere [e.g. Davison, 1980], enabling comparisons to be drawn. The results presented for the experiments with the heat exchanger reinforce many of the points already discussed, in particular:

The value of simulations of the process has been demonstrated, even for the situation where the process was non-linear, and where a simplified linear model was used. The control strategies which had to be adopted for the control of the real and simulated processes were similar.

The process was non-linear, with diffusion effects as has been described in Chapter 3, and may be operated in such a way that the process model parameters and time delay change during use. The self-tuning controller proved to be particularly suited to control of the process under these conditions.

When the delay between samples was small, resulting in low

gain, the predicted vigo rous control action was evidenced as has been discussed above (see Figure 4.36). It is also evident that this excessive action may be reduced by increasing the sampling interval, as in Figure 4.38. It is shown in Figure 4.39 that selection of a pole to give a very damped response may not greatly decrease the controller action, but may introduce other undesirable effects due to the inability of the controller to rapidly correct other deviations from set point, due for example to interactions.

In controlling the heat exchanger process, great difficulty was experienced in the use of the implicit controller minimising a one step criterion. It is felt that the one-step criterion is probably inappropriate for this particular process, and that a controller based on a complete solution of the Riccati equation may be more successful though computationally less attractive.

5.2 Methods employed

The programs which have been used have been written with the aim of obtaining a compromise between fast execution and economical program size, to enable them to be efficiently implemented on a microcomputer. Of necessity, the structure has had to be made as flexible as possible to enable different ideas to be tested quickly and easily. That this has been achieved is evidenced by the use of this same structure for other control schemes ranging from three-term controllers to optimal controllers, and for parameter identification programs. In their present form the routines are also suitable for incorporating into other control program suites, such as the one described by Sandoz and Wong [1978].

Although the routines have all been written in Fortran, the code produced by the Cromemco CDOS Fortran compiler is suitable for storing in EPROM memory. The arrays and variables of course have to be stored in random access memory. The subroutines in this form are suitable for execution in a single-board microcomputer system (based on a Z-80 microprocessor). The price paid for this economy is the loss of the recording, graphing and interactive facilities offered by a system with disks and a graphics terminal. That the programs can realistically execute on one of the standard microcomputer systems so readily available, is felt to be an important practical result of this research.

So that the program size may be further reduced while the speed of execution is also improved, a program generator could be used to generate in-line code which would be dedicated to use on a particular process. The use of the generator would make this tailoring of the programs for each process feasible. This would eliminate any redundancy in array storage space, and the use of pointers and generalised program loops for dealing with the polynomials. Standard subroutines such as that for estimation of parameters could be re-coded in assembler language.

The use of a full simulation language (ISIS) for simulations with a controller is a powerful facility. The structure of the subroutines, written so that they could be incorporated into other controller program suites, facilitated the use of ISIS.

5.3 Relationship to other self-tuning control schemes

Self-tuning schemes have divided into those employing implicit methods, and those which explicitly identify parameters. Implicit self-tuners are usually based on a minimum variance control law [Astrom and Wittenmark, 1973], or on the control scheme proposed by Clarke and Gawthrop [1975, 1979] which overcomes the objection

to the minimum variance control that it cannot deal with non-minimum phase systems. Multivariable controllers for the minimum variance controller have been proposed by Borisson [1979] and used for control of cement raw material blending by Keviczky et al [1978]. Koivo [1980] has described a multivariable controller derived from that of Clarke and Gawthrop [1979].

The implicit controller which has been developed in this thesis uses the method of disturbance feed-forward to deal with multivariable plant, hinted at by Clarke and Gawthrop [1975] and Astrom et al [1977]. The difficulties associated with this method have been addressed, [Hesketh, 1980]. Experience with this controller suggests that for some processes, the problem of relating the weighting polynomials of the cost function to the closed loop behaviour which results, particularly with the one-step cost function minimisation, can mitigate against the acceptance of this controller for use with such processes. The work of Allidina and Hughes [1980] relating the cost function polynomials to the pole placement problem may be important in this regard.

Explicit identification of the process parameters has been used to extend the one-step minimisation used by Clarke and Gawthrop [1979] to a full solution of the Riccati equation [Buchholt and Kummel, 1979], or to a "more optimal" k-step design method [Jacobs and Saratchandran, 1980], with attendant increases to the computational requirements of such methods.

The pole-placing techniques of Wellstead et al [1979 a,b,c] also use explicit methods, providing a robust mechanism which deals adequately with non-minimum phase systems, and which may be extended to deal with multivariable processes, although at the cost of considerable computation. The burden of calculating the controller parameters is the chief fault of the pole-placing self-tuners, particularly when self-tuning is so suited in other ways to low-cost microprocessor implementation. This thesis has

presented a method for designing a pole-placing controller which follows the principle of using feedforward terms to deal with interactions between loops, and at the same time does not impose an excessive computational burden. The proposed controller provides a simplified "decoupling" of the loops, so that the output for each loop is related by a scalar transfer function to the set-point for that loop. Set point attainment is ensured by appropriate selection of an "observer polynomial" (in the terminology of Astrom [1980]), which is made possible by the decoupling described above. A full solution of the pole-placing problem has also been presented, and this allows a comparison of the simplified decoupling method with a full multivariable solution, which shows that there may be very little difference between the two.

APPENDIX A

Numerical examples of pole-placing controller design.

1. Scalar process, polynomial approach.

Consider the process described by the difference equation

$$(1-0.5q^{-1})y(t) = (0q^{-1}+1.0q^{-2}+1.5q^{-3})u(t) + e(t) \quad ....(A1)$$

i.e. a process which has a time delay k=2 sample steps, and which is non-minimum phase. Note that it is assumed that these values are biassed estimates from the least squares parameter estimation procedure. The required closed-loop response of the controller and process is defined by the model polynomial

$$\mathbf{A}_m = 1-0.8q^{-1} \quad ....(A2)$$

Equations (A3) and (A4) describe the regulator and the closed loop transfer function obtained by substituting (A3) into (A1).

$$\mathbf{H}u(t) + \mathbf{G}y(t) = 0 \quad ....(A3)$$

$$[(1-0.5q^{-1})\mathbf{H} + (0q^{-1}+1.0q^{-2}+1.5q^{-3})\mathbf{G}]y(t) = \mathbf{H}e(t) \quad ....(A4)$$

The characteristic equation should be equated to the model polynomial, and the resulting polynomial equation solved for the coefficients of **H** and **G** giving:

$$\begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 \\ -0.5 & 1.0 & 0.0 & 0.0 \\ 0.0 & -0.5 & 1.0 & 1.0 \\ 0.0 & 0.0 & -0.5 & 1.5 \end{bmatrix} \begin{bmatrix} h0 \\ h1 \\ h2 \\ g1 \end{bmatrix} = \begin{bmatrix} 1.0 \\ -0.8 \\ 0.0 \\ 0.0 \end{bmatrix} \quad ....(A5)$$

The solution is

$$[h0\ \ h1\ \ h2\ \ g0]^T = [1.0\ \ -0.3\ \ -0.112\ \ -0.037]^T \qquad \ldots.(A6)$$

2. Scalar process, matrix approach.

The process described by equation (A1) may be re-expressed in matrix form as:

$$\begin{bmatrix} u(t-2) \\ u(t-1) \\ y(t) \end{bmatrix} = \begin{bmatrix} 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 1.5 & 1.0 & 0.5 \end{bmatrix} \begin{bmatrix} u(t-3) \\ u(t-2) \\ y(t-1) \end{bmatrix} + \begin{bmatrix} 0.0 \\ 1.0 \\ 0.0 \end{bmatrix} u(t-1)$$

$$x(t) = \mathbf{D}\, x(t-1) + \mathbf{e}\, u(t-1) \qquad \ldots.(A7)$$

The observability matrix and its inverse are given by:

$$\mathbf{J} = [\mathbf{e}, \mathbf{De}, \mathbf{D}^2\mathbf{e}] = \begin{bmatrix} 0.0 & 1.0 & 0.0 \\ 1.0 & 0.0 & 0.0 \\ 0.0 & 1.0 & 2.0 \end{bmatrix}$$

$$\mathbf{J}^{-1} = \begin{bmatrix} 0.0 & 1.0 & 0.0 \\ 1.0 & 0.0 & 0.0 \\ -0.5 & 0.0 & 0.5 \end{bmatrix} \qquad \ldots.(A8)$$

This permits the formulation of the matrix **Q** used for the similarity transformation $x'=\mathbf{Q}x$.

$$\mathbf{Q} = \begin{bmatrix} \mathbf{q} \\ \mathbf{qD} \\ \mathbf{qD}^2 \end{bmatrix} = \begin{bmatrix} -0.500 & 0.000 & 0.500 \\ 0.750 & 0.000 & 0.250 \\ 0.375 & 1.000 & 0.125 \end{bmatrix} \qquad \ldots.(A9)$$

The transition and driving matrices of the transformed state equations are given by

$$\mathbf{QDQ}^{-1} = \begin{bmatrix} 0.0 & 1.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \\ 0.0 & 0.0 & 0.5 \end{bmatrix} \qquad \mathbf{Qe} = \begin{bmatrix} 0.0 \\ 0.0 \\ 1.0 \end{bmatrix} \qquad \ldots.(A10)$$

It is of interest to note the structure of these matrices; in particular, the last row has coefficients of 0 for the terms in u(t-2) and u(t-3) but THE COEFFICIENT OF y(t)IS UNCHANGED BY THE TRANSFORMATION, i.e. THE TRANSFORMATION WILL NOT CHANGE THE COEFFICIENTS OF THE PROCESS CHARACTERISTIC EQUATION.

The last row of the $\mathbf{QDQ}^{-1}$ matrix is required to be [0.0,0.0,0.8] to produce the pole required by the model polynomial (A2). Thus the controller is given by:

$$\mathbf{F}_c = [0.0\ 0.0\ 0.3]$$

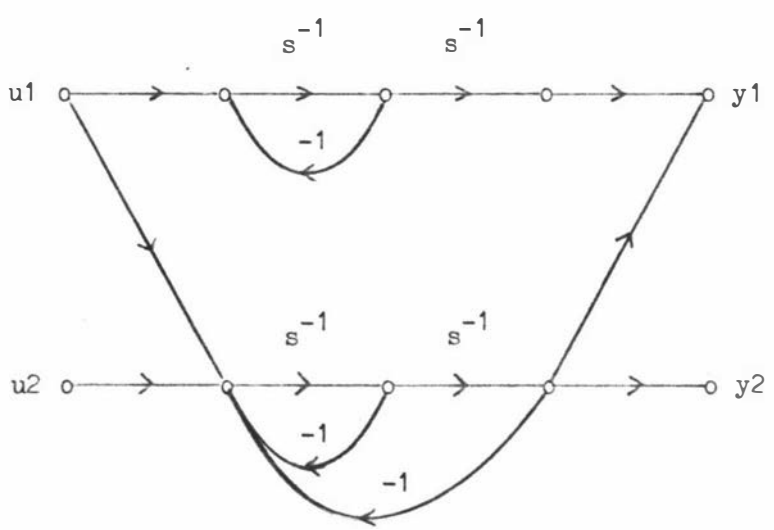
$$u(t-1) = \mathbf{F}_c x'(t-1) = \mathbf{F}_c \mathbf{Q} x(t-1)$$

$$\mathbf{F}_c \mathbf{Q} = [0.112\ 0.300\ 0.038] \qquad \ldots.(A11)$$

This defines exactly the same controller as that described by equation (A6), allowing for the re-ordered vector. It should be noted that, because of the structure of the matrices given in (A10), computation of $\mathbf{F}_c$ follows immediately the parameters of equations (A1) and (A2) are known. Only computation of $\mathbf{Q}$ is required to complete the controller given by (A11).

3. Multivariable controller.

A multivariable pole-placing controller will be designed for the process shown in the flow diagram below (omitting noise terms).



For a sampling interval of 1 second and a delay k=2 sampling

intervals this is described by the difference equations:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + \begin{bmatrix} -1.368 & 0.582 \\ 0.000 & 0.786 \end{bmatrix} q^{-1} + \begin{bmatrix} 0.368 & 0.000 \\ 0.000 & 0.368 \end{bmatrix} q^{-2}$$

$$\mathbf{B} = \begin{bmatrix} 0.708 & 0.340 \\ 0.340 & 0.340 \end{bmatrix} q^{-2} + \begin{bmatrix} 0.506 & 0.242 \\ 0.242 & 0.242 \end{bmatrix} q^{-3}$$

....(A12)

Following the same analysis as for the scalar case the following matrices may be calculated:

$$\mathbf{D} = \begin{bmatrix} 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 1.000 \\ 0.506 & 0.242 & 0.708 & 0.340 & -0.368 & 0.000 & 1.368 & -0.582 \\ 0.242 & 0.242 & 0.340 & 0.340 & 0.000 & -0.368 & 0.000 & 0.786 \end{bmatrix}$$

$$\mathbf{E} = \begin{bmatrix} 0.000 & 0.000 \\ 0.000 & 0.000 \\ 1.000 & 0.000 \\ 0.000 & 1.000 \\ 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.000 \end{bmatrix}$$

$$\mathbf{J} = \begin{bmatrix} 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.708 & 0.340 & 1.277 & 0.509 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.340 & 0.340 & 0.509 & 0.509 \\ 0.000 & 0.000 & 0.708 & 0.340 & 1.277 & 0.509 & 1.190 & 0.275 \\ 0.000 & 0.000 & 0.340 & 0.340 & 0.509 & 0.509 & 0.275 & 0.275 \end{bmatrix}$$

$$\mathbf{J}^{-1} = \begin{bmatrix} 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ -1.117 & 0.000 & 0.000 & 0.000 & -3.615 & 3.615 & 3.034 & -3.034 \\ 0.071 & -1.045 & 0.000 & 0.000 & 3.615 & -5.276 & -3.034 & 6.108 \\ 0.535 & 0.000 & 0.000 & 0.000 & 3.034 & -3.034 & -1.454 & 1.454 \\ 0.163 & 0.698 & 0.000 & 0.000 & -3.034 & 6.108 & 1.454 & -3.507 \end{bmatrix}$$

$$\mathbf{Q} = \begin{bmatrix} 0.535 & 0.000 & 0.000 & 0.000 & 3.034 & -3.034 & -1.454 & 1.454 \\ 0.163 & 0.698 & 0.000 & 0.000 & -3.034 & 6.108 & 1.454 & -3.507 \\ -0.384 & 0.000 & 0.000 & 0.000 & 0.535 & -0.535 & 1.045 & -1.045 \\ -0.113 & -0.497 & 0.000 & 0.000 & -0.535 & 1.291 & -1.045 & 2.505 \\ 0.276 & 0.000 & 0.000 & 0.000 & -0.385 & 0.385 & 1.965 & -1.965 \\ 0.000 & 0.353 & 0.000 & 0.000 & 0.385 & -0.922 & -1.965 & 3.868 \\ 0.519 & 0.000 & 1.000 & 0.000 & -0.723 & 0.723 & 2.303 & -2.303 \\ -0.000 & 0.461 & 0.000 & 1.000 & 0.723 & -1.423 & -2.303 & 3.262 \end{bmatrix}$$

$$\mathbf{Q}^{-1} = \begin{bmatrix} 0.368 & 0.000 & -1.368 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.368 & 0.479 & -0.786 & 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.368 & 0.000 & -1.368 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.163 & 0.368 & 0.462 & -0.786 & 0.000 & 1.000 \\ 0.506 & 0.242 & 0.708 & 0.340 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.242 & 0.242 & 0.340 & 0.340 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.470 & 0.242 & 0.735 & 0.340 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.206 & 0.242 & 0.368 & 0.340 & 0.000 & 0.000 \end{bmatrix}$$

$$QDQ^{-1} = \begin{bmatrix} 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 1.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 0.000 & 1.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & -0.368 & 0.000 & 1.366 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & -0.176 & -0.368 & -0.521 & 0.786 \end{bmatrix}$$

$$QE = \begin{bmatrix} 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.000 \\ 1.000 & 0.000 \\ 0.000 & 1.000 \end{bmatrix}$$

....(A13)

Examining QDQ^{-1} it is apparent that while the coefficients on the diagonals of the new characteristic polynomial coefficient matrices are unchanged from those of the original matrices, the off-diagonal coefficients do change as a result of the transformation, necessitating the computation of all the matrices in the multivariable case. Requiring

$$\mathbf{A}_m = \begin{bmatrix} 1-0.8q^{-1} & 0 \\ 0 & 1-0.8q^{-1} \end{bmatrix}$$

$$\mathbf{F}_c = \begin{bmatrix} 0.000 & 0.000 & 0.000 & 0.000 & 0.366 & 0.000 & -0.567 & 0.000 \\ 0.000 & 0.000 & 0.000 & 0.000 & 0.176 & 0.369 & 0.521 & 0.013 \end{bmatrix}$$

$$\mathbf{F}_c\mathbf{Q} = \begin{bmatrix} -0.193 & 0.000 & -0.567 & 0.000 & 0.269 & -0.269 & -0.587 & 0.587 \\ 0.319 & 0.136 & 0.521 & 0.013 & -0.293 & 0.086 & 0.791 & -0.076 \end{bmatrix}$$

....(A14)

This last matrix contains the parameters of the multivariable pole-placing controller.

APPENDIX B

Heat Exchanger Response with Three-Term Control

Hesketh and Chaplin (1981) have described the behaviour of the heat exchanger process reported in this thesis under control of both the self-tuning controller, and so as to provide a comparison, two three term controllers, one for each loop. The behaviour with the three term controllers is reproduced here to illustrate the severity of the disturbance which results when the flow through the first heat exchanger is changed from 15 litres/min to 10 litres/min. This emphasises the capability of the self-tuning controller.

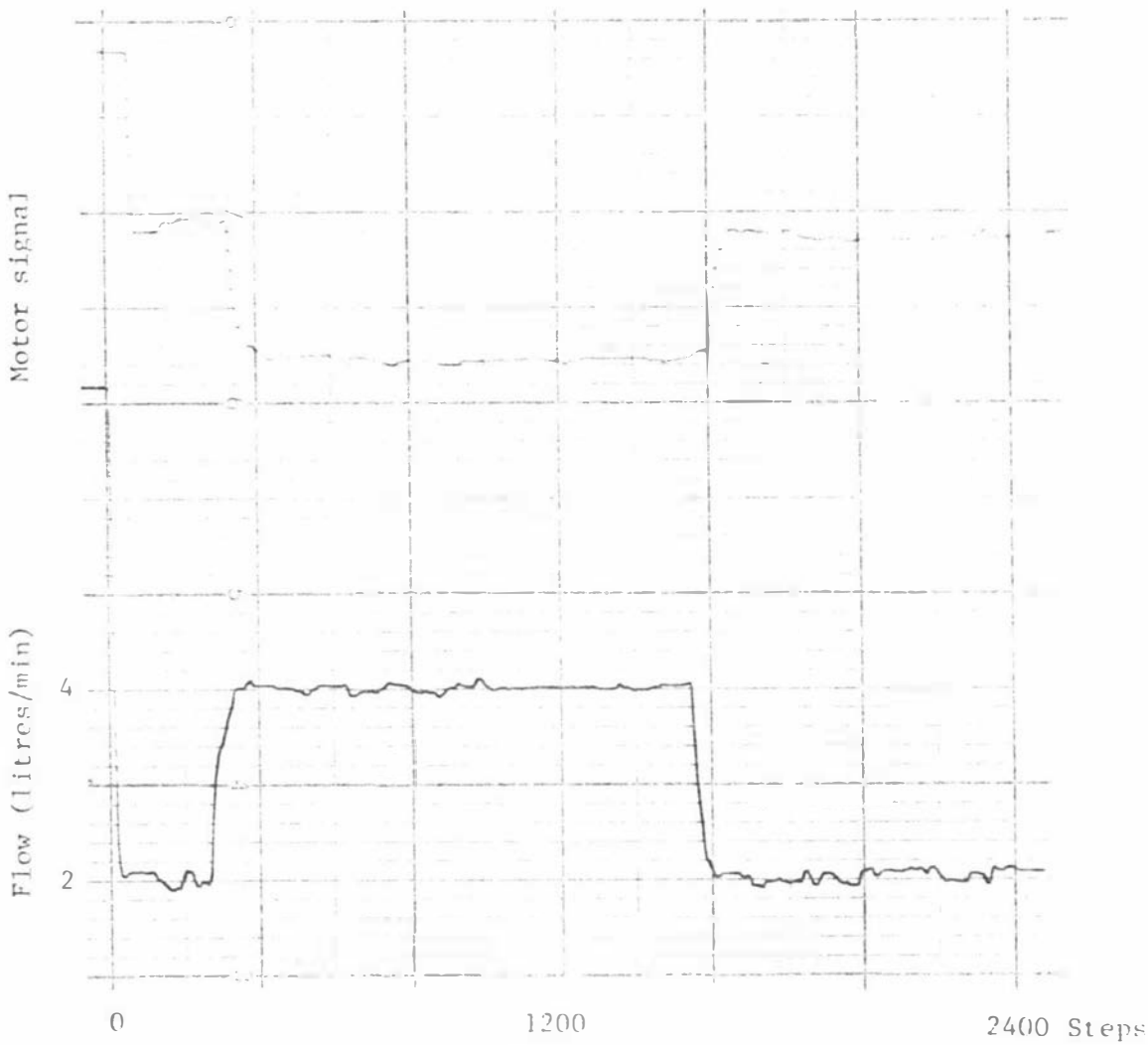


FIGURE B1

Heat exchanger loop 1 with three-term control

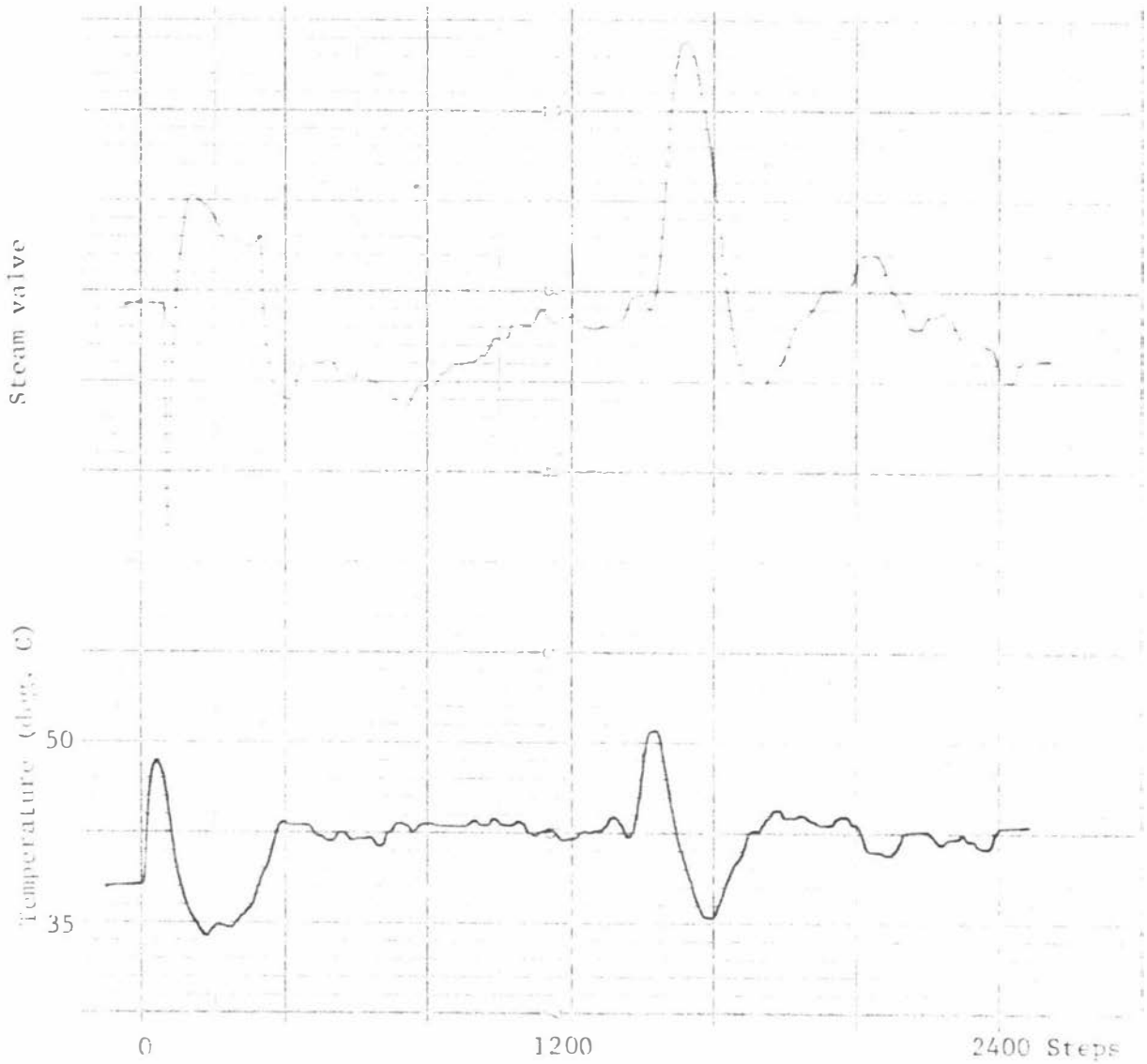


FIGURE B2

Heat exchanger loop 2 with three term control

REFERENCES

ALLIDINA, A.Y. and HUGHES, F.M., (1979), 'Self-tuning controller steady state error', Electronics Letters, 15, (12), 346-347.

ALLIDINA, A.Y. and HUGHES, F.M., (1980), 'Generalised self-tuning controller with pole assignment', IEE Proceedings, 127, Pt. D, No. 1, 13-18.

ASTROM, K.J., 'Introduction to Stochastic Control Theory', (1970), (London: Academic Press).

ASTROM, K.J., BORISSON, U., LJUNG, L. and WITTENMARK, B., (1977), 'Theory and applications of self-tuning regulators', Automatica, 13, 457-476.

ASTROM, K.J. and WITTENMARK, B., (1973), 'On self-tuning regulators', Automatica, 9, 185-199.

ASTROM, K.J. and WITTENMARK, B., (1980), 'Self-tuning controllers based on pole-zero assignment', IEE Proceedings, 127, Pt. D, No. 3, 120-130.

ASTROM, K.J., (1980), 'Robustness of a design method based on assignment of poles and zeros', IEEE Transactions on Automatic Control, AC-25, 3, 588-590.

BAUR, U. and ISERMANN, R., (1977), 'On-line identification of a heat exchanger with a process computer - a case study', Automatica, 13, 487-496.

BAYOUMI, M.M. and EL-BAGOURI, M.A., (1979), 'Comments on "Self-tuning adaptive control of raw-material blending"', Automatica, 15, 693-694.

BIERMAN, G.J., (1976), 'Measurement updating using the U-D Factorisation', Automatica, 12, 375-382.

BORISSON, U., (1979), 'Self-tuning regulators for a class of multivariable systems', Automatica, 15, 209-215.

BOX, G.E.P. and JENKINS, G.M., (1970), Time series analysis, forecasting and control', Holden Day.

BUCHHOLT, F. and KUMMEL, M., (1979), 'Self-tuning control of a pH-neutralisation process', Automatica, 15, 665-671.

CLARKE, D.W. and GAWTHROP, P.J., (1975), 'Self-tuning controller', Proceedings IEE, 122, 9, 929-934.

CLARKE, D.W. and GAWTHROP, P.J., (1979), 'Self-tuning control', Proceedings IEE, 126, 6, 633-640.

CLARKE, D.W. and GAWTHROP, P.J., (1979), 'Implementation and application of microprocessor based self-tuners', Preprints 5th IFAC Symposium on Identification and System Parameter Estimation, Darmstadt, 197-208.

DAVISON, E.J., TAYLOR, P.A. and WRIGHT, J.D., (1980), 'On the application of tuning regulators to control a commercial heat exchanger', IEEE Transactions on Automatic Control, AC-25, 3, 361-374.

EGARDT, B., (1978), 'Stability of model reference adaptive and self-tuning regulators', Report LUT FD2/(TFRT-1017)/1-163/1978, Lund Institute of Technology, Sweden.

EGARDT, B., (1979), 'Unification of some real-time adaptive control schemes', IEEE Transactions on Automatic Control, AC-24, 4, 588-591.

FEUER, A. and MORSE, A.S., (1978), 'Adaptive control of single-input, single-output linear systems', IEEE Transactions on Automatic Control, AC-23, 4, 557-569.

FORTESQUE, T.R., KERSHENBAUM, L.S., and YDSTIE, B.E., (1979), 'Implementation of self-tuning regulators with variable forgetting factors', Report, Department of Chemical Engineering and Chemical Technology, Imperial College, London.

GAWTHROP, P.J., (1977), 'Some interpretations of the self-tuning controller', Proceedings IEE, 124, 10, 889-894.

GAWTHROP, P.J., (1980), 'Hybrid self-tuning control', IEE Proceedings, 127, Pt. D, 5, 229-236.

GLATTFELDER, A.H., HUGUENIN, F. and SCHAUFELBERGER, W., (1980), 'Microcomputer based self-tuning and self-selecting controllers', Automatica, 16, 1-8.

GOODWIN, G.C. and RAMADGE, P.J., (1979), 'Design of restricted complexity adaptive regulators', IEEE Transactions on Automatic Control, AC-24, 4, 584-588.

GOODWIN, G.C., RAMADGE, P.J. and CAINES, P.E., (1980), 'Discrete time multivariable adaptive control', IEEE Transactions on Automatic Control, AC-25, 3, 449-456.

HESKETH, T., (1980), 'Disturbance rejection and multivariable self-tuning controllers', Electronics Letters, 16, 11, 404-406.

HESKETH, T. and CHAPLIN, R., (1981), 'Comparison of algorithms for microprocessor control', Proceedings Control '81, N.Z. Automatic Control and Instrumentation Society, 125-147.

JACOBS, O.L.R. and SARATCHANDRAN, P., (1980), 'Comparison of adaptive controllers', Automatica, 16, 89-97.

KALLSTROM, C.G., ASTROM, K.J., THORELL, N.E., ERIKSSON, J., STEN, L., (1979), 'Adaptive autopilots for tankers', Automatica, 15, 241-254.

KERSHENBAUM, L.S. and FORTESQUE, T.R., (1978), 'Implementation of on-line control in chemical process plants', Report, Department of Chemical Engineering and Chemical Technology, Imperial College, London.

KEVICZKY, L., HETTHESSY, J., HILGER M. and KOLOSTORI, J., (1978), 'Self-tuning adaptive control of cement raw material blending', Automatica, 14, 525-532.

KOIVO, H.N., (1980), 'A multivariable self-tuning controller', Automatica, 16, 351-366.

KURZ, H., ISERMANN, R. and SCHUMANN, R., (1980), 'Experimental comparison and application of various parameter-adaptive control algorithms', Automatica, 16, 117-133.

LANDAU, Y.D., (1979), 'Adaptive control: the model reference approach', Marcel Dekker Inc., New York.

LEDWICH, G., (1979), 'Adaptive excitation control', Proceedings IEE, 126, 3, 249-253.

LJUNG, L., (1977), 'On positive real transfer functions and the convergence of some recursive schemes', IEEE Transactions on Automatic Control, AC-22, 4, 539-575.

LJUNG, L., (1978), 'Convergence of an adaptive filter algorithm', International Journal of Control, 27, 5, 673-693.

MORRIS, A.J., FENTON, T.P. and NAZER, Y., (1977), 'Application of self-tuning regulators to the control of chemical processes', Digital Computer Applications to Process Control, van Naute Lemke (Editor), IFAC and North Holland Publishing Co.

NELSON, L.W. and STEAR, E., (1976), 'The simultaneous on-line estimation of parameters and states in linear systems', IEEE Transactions on Automatic Control, AC-21, 94-98.

PADILLA, R.A., PADILLA, C.S. and BINGULAC, S.P., (1978), 'Comments on "The simultaneous on-line estimation of parameters and states in linear systems"',IEEE Transactions on Automatic Control, AC-23, 1, 96-97.

PANUSKA, V., (1980), 'A new form of the extended Kalman filter for parameter estimation in linear systems with correlated noise', IEEE Transactions on Automatic Control, AC-25, 2, 229-235.

PENTTINEN, J. and KOIVO, H.N., (1980), 'Multivariable tuning regulators for unknown systems', Automatica, 16, 393-398.

PETERKA, V., (1970), 'Adaptive digital regulation of noisy systems', 2nd Prague IFAC Symposium on Identification and Process Parameter Estimation.

SANDOZ, D.J. and WONG, O., (1978), 'Design of hierarchical computer control systems for industrial plant', Parts 1 & 2, IEE Proceedings, 115,11, 1290-1298,1299-1306.

SMITH, O.J.M., (1959), 'A controller to overcome dead-time', Instrument Society of America Journal, 6, 28-33.

THORNTON, C.L. and BIERMAN, G.J., (1978), 'Filtering and error analysis via the UDU covariance factorisation', IEEE Transations on Automatic Control, AC-23, 901-907.

WELLSTEAD, P.E., EDMUNDS, J.M., PRAGER, D. and ZANKER, P., (1979), 'Self-tuning pole/zero assignment regulators', International Journal of Control, 30, 1, 1-26.

WELLSTEAD, P.E., PRAGER, D. and ZANKER, P., (1979), 'Pole Assignment self-tuning regulator', Proceedings IEE, 126, 8, 781-787.

WELLSTEAD, P.E. and ZANKER, P., (1979), 'Servo self-tuners', International Journal of Control, 30, 1, 27-36.

WITTENMARK, B., (1979), 'A two level estimator for time varying parameters', Automatica, 15, 85-89.

WOLOVICH, W.A., (1974), 'Linear multivariable systems', Springer Verlag, New York.

YOUNG, P.C., (1979), 'Self-adaptive Kalman Filter', Electronics Letters, 15, 12, 358-360.

KALMAN, R.E., (1958), 'Design of a self-optimising control system', Trans. ASME, 80, 468-478.