An Automatic Controller Tuning Algorithm

Michael A. Christodoulou

A project report submitted to the Faculty of Engineering, University of the Witwatersrand, Johannesburg, in partial fulfillment of the requirements for the degree of Master of Science in Engineering.

Johannesburg 1991

DECLARATION

I declare that this project report is my own, unaided work. It is being submitted for the Degree of Master of Science in Engineering at the University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in any other University.

[illegible]

Michael Christodoulou

The 06 day of February 1991

To my wife Brenda

ACKNOWLEDGMENTS

I am grateful to Professor I. M. MacLeod for his encouraging support and interest throughout the project.

I would like to thank the Technical Development Department of A.E.C.I. Modderfontein for their financial support for the duration of the project.

ABSTRACT

The report describes the design of an algorithm which can be used for automatic controller tuning purposes. It uses an on-line parameter estimator and a pole assignment design method. The resulting control law is formulated to approximate a proportional-integral (PI) industrial controller. The development of the algorithm is based on the delta-operator. Some implementation aspects such as covariance resetting, deadzone, and signal conditioning are also discussed. Robust stability and performance are two issues that govern the design approach. Additionally transient and steady state system response criteria are utilized from the time and frequency domains. The design work is substantiated with the use of simulation and real plant tests.

Contents

List of Figures

Chapter 1

Introduction

1.1 Background

The proportional-integral (PI) feedback controller is unquestionably the most commonly used controller in the chemical process industry. Its robust performance in a wide range of applications is the main reason behind its popularity. However, the large number of PI feedback control loops in a chemical plant makes it impractical to tune (or adjust) the parameters of every controller accurately. Frequent re-adjustment of controller parameters as process operating conditions change is also impractical.

Several manual methods have been developed in the past for the tuning of controllers. The most well known is the Ziegler and Nichols[1,2] method developed in the early 1940's. The closed-loop cycling method obtains the parameters using both the plant oscillation period and the observed system gain that causes the system to cycle. The open-loop method is an extension of the closed-loop cycling method. It determines the required parameters from the open-loop response of the system to a step input. Also, Shinskey [3], Cohen and Coon [4], and Hazebroek and Van Der Waerden[5] have developed similar methods that gained wide acceptance. Attaining the controller parameters using these methods can be a cumbersome and time consuming exercise. Additionally, they often cause the system response to be rather oscillatory. Therefore, it seems there is a need for a robust and reliable algorithm which can be utilized to automatically tune PI feedback controllers either on a continuous basis or upon a request initiated by the operator.

The idea of automatic controller tuning has been investigated by several authors such as Kalman[6], Astrom [7], Peterka[8], Clarke and Gawthrop [9], and Borison [10]. The recent hardware and software developments in the computer field and the price drop in computer equipment resulted in the widespread application of computers in the process industry. The availability and robustness of microprocessor based controllers especially the programmable logic controller (PLC), also, strengthen the industry's move towards automation. These reasons could possibly explain the recent increase in research funds and efforts in the fields of system identification and control.

This report considers the development of a generalized robust system which produces reliable identification of plant parameters as well as reliable and robust controller parameters. The aim of the study is for the algorithm to eventually be implemented in a plant supervisory computer system so that controller parameters can be automatically down-loaded into the programmable logic controllers (PLCs) which perform the front-end PI feedback control. The concept is illustrated in Figure 1.1.

[illegible]

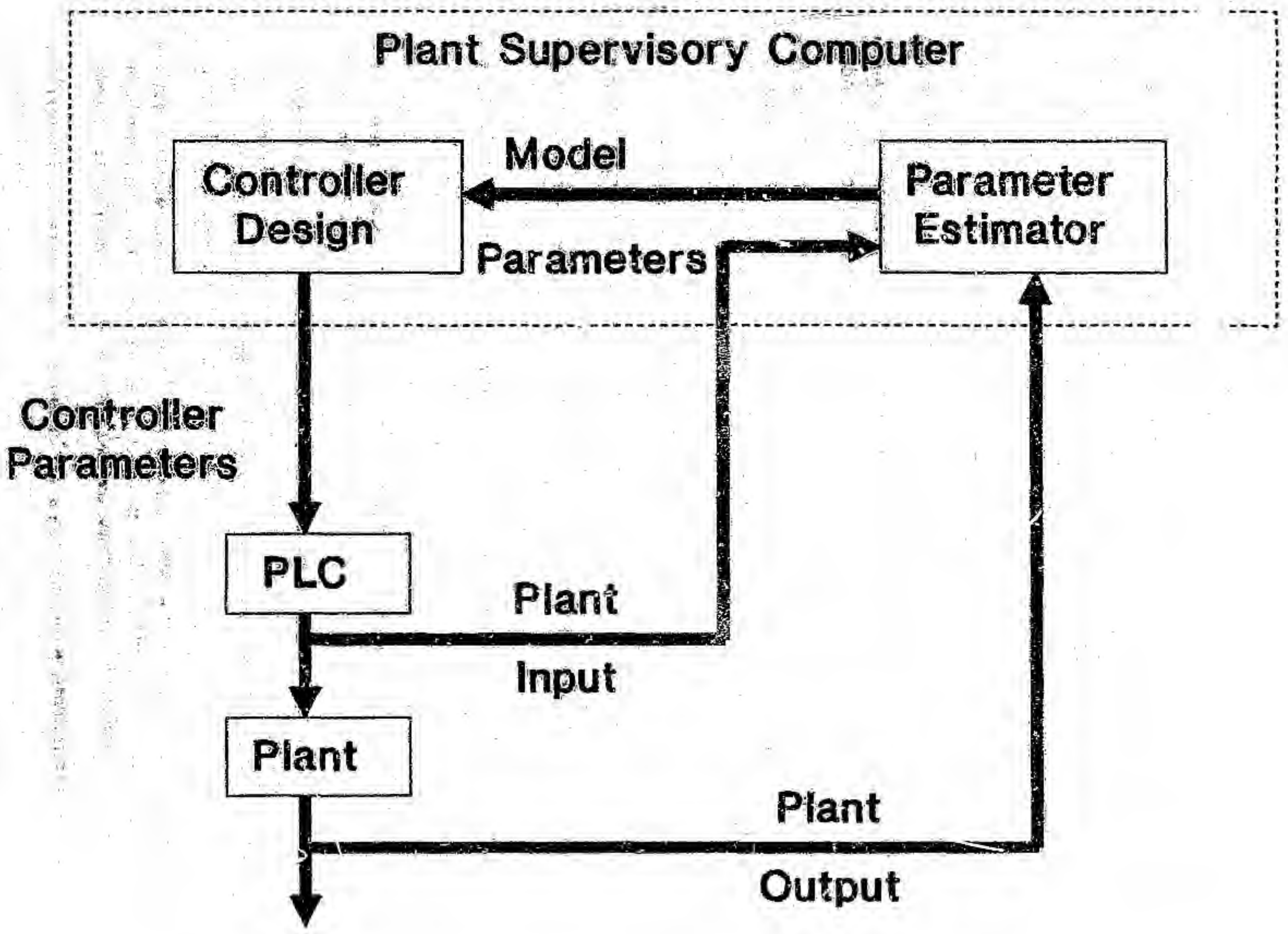


Figure 1.1: Block Diagram of the Computer Control System

1.2 Objective and Overview of the Report

1.2.1 Objective of the Research

The objective of the research is to design a system that can be used to perform automatic controller tuning. The approach consists of a recursive parameter estimation algorithm and a controller design algorithm. The controller design algorithm is based on the closed-loop pole assignment method and when the process is modeled by a first-order system, the pole assignment approximates the standard PI feedback controller. The method differs from others because the overall design is formulated using the delta-operator.

Some of the design's basic characteristics are the following.

- The delta-operator and its associated transform.
- Signal conditioning for proper conditioning of the measured signals to improve the estimator performance.
- A generalized process model that includes both measured and unmeasured disturbances.
- A rule-based mechanism for establishing the location of the closed-loop poles.

1.2.2 Overview of the Report

The report is organized in the following way.

- Chapter 2 discusses current techniques used for estimating unknown parameters of deterministic systems.
- Chapter 3 discusses some modern methods for designing controllers for the control of linear deterministic systems.
- Chapter 4 examines the concept of the new shift-operator namely the delta-operator. It, also, shows how the delta-operator can be related to the standard continuous and discrete time operators.
- Chapter 5 presents the structures of the parameter estimator algorithm and the pole assignment algorithm used for the design of the controller parameters. It, also, presents the method for formulating the pole assignment to approximate a standard proportional-integral controller.
- Chapter 6 discusses some implementation aspects of the design such as signal conditioning and robustness. It, also, presents the methods of translating the performance requirements into the desired location for the closed-loop poles.
- Chapter 7 presents the properties of the algorithm using simulation and real plant tests.
- Chapter 8 concludes the report and presents the main findings and recommendations of the study.
- Appendix A presents the Bierman U-D covariance factorization method which is modified to include the deadzone and forgetting factors

- Appendix B presents ‿ ›lternative method for expressing a PI controller in a delta-operator format.

- Appendix C presents the method of approximating a pole placement controller to a PID controller.

- Appendix D presents the fortran source code.

Chapter 2

Principles of Parameter Estimation

2.1 Introduction

The chapter gives a brief description of some current approaches to parameter estimation. Also, the chapter presents one of the typical difference equation formulas used most often to express a process model.

Parameter estimation is the procedure used to deduce the values of an unknown system by observing the nature of the response of the system under appropriate and controlled conditions. A survey of parameter estimation methods for linear systems based on sampled signals is given by Young [11].

Parameter estimation schemes can be separated into two general categories:

1. Off-Line schemes
2. On-Line schemes

The difference between off-line and on-line algorithms is that the off-line algorithms estimate the model parameters by processing plant input and output data in batches. The processed data have been stored previously on storage facilities and are always historical. The on-line algorithms estimate the model parameters by continuously processing historical and current process data.

The on-line techniques are mostly preferred by the researchers because they form the basis for the formulation of adaptive control philosophy. Finally, among all the on-line methods, the recursive least squares is the most widely used method due to its robustness and flexibility.

2.2 Linear Parametric Process Model

A process model format which can be used for parameter estimation purposes is the Deterministic Autoregressive Moving Averages (DARMA) Model given by:

$$y(k)+a_1\,y(k-1)+\cdots+a_n\,y(k-n) = b_0\,u(k-d)+b_1\,u(k-d-1)+\cdots+b_m\,u(k-d-m) \tag{2.1}$$

where: $k = \frac{t}{T_s} = 0,1,2,\ldots$ is the discrete time, T_s is the sampling time, d is the time delay, y and u are the plant output and input, respectively, and n and m give the order of the model.

Using the backward shift operator, the model can be represented by:

$$A(q^{-1})\,y(k) = B(q^{-1})\,u(k) \tag{2.2}$$

where:

$$A(q^{-1}) = 1 + a_1\,q^{-1} + a_2\,q^{-2} + \cdots + a_n\,q^{-n}$$
$$B(q^{-1}) = (b_0 + b_1\,q^{-1} + b_2\,q^{-2} + \cdots + b_m\,q^{-m})q^{-d}$$

The DARMA Model can, also, be expressed as:

$$y(k) = \phi(k-1)\,\theta(k-1) \tag{2.3}$$

where:
$y(k)$ is the system output at time k
$\phi(k-1)$ is the system's output and input history vector
$\theta(k-1)$ is the parameter vector

The model parameters are the unknown factors and need to be identified. The above model forms the basis for the parameter estimation schemes and it is particularly convenient for the subsequent development of the autotuning algorithm.

2.3 Projection Algorithms

2.3.1 Gradient Algorithm

The gradient algorithm or projection algorithm is expressed as:

$$\theta(k) = \theta(k-1) + \frac{\alpha\phi(k-1)}{c + \phi(k-1)^T \phi(k-1)}[y(k) - \phi(k-1)^T \theta(k-1)] \qquad (2.4)$$

The term $y(k) - \phi(k-1)^T \theta(k-1)$ can be expressed as:

$$e(k) = y(k) - \phi(k-1)^T \theta(k-1)$$

where $e(k)$ is called the modeling error.

It is shown by Goodwin and Sin [12] that the algorithm minimizes the cost function,

$$J = \frac{1}{2}\, |\theta(k) - \theta(k-1)|^2 \qquad (2.5)$$

The $c > 0$ is added to the regression vector to ensure that division by zero does not take place. The choice of the gain variable α is usually between 0 and 1. In some filtering literature, the sane algorithm is called the normalized least-mean-squares algorithm (NLSA). The algorithm has a very slow convergence and although, it always converges, it does not guarantee convergence to the true plant parameters.

2.3.2 Orthogonalized Gradient Algorithm

An improvement to the basic gradient algorithm is the orthogonalized gradient algorithm. It is given by:

$$\theta(k) = \theta(k-1) + \frac{P(k-2)\,\phi(k-1)}{c + \phi(k-1)^T P(k-2)\,\phi(k-1)}[y(k) - \phi(k-1)^T \theta(k-1)] \qquad (2.6)$$

$$P(k-1) = P(k-2) - \frac{P(k-2)\,\phi(k-1)\,\phi(k-1)^T P(k-2)}{c + \phi(k-1)^T P(k-2)\,\phi(k-1)} \qquad (2.7)$$

with the initial estimation vector $\theta(0)$ given, and $P(-1) = I$.

The $P(k-1)$ matrix is a projection operator which ensures the orthogonality of the algorithm. To avoid division by zero, the constant $c > 0$ is added in the denominator.

2.4 Least Squares Algorithm

It is by far the most popular algorithm and as Goodwin and Sin [13] show is based on the minimization of the quadratic cost function

$$J_N(\theta) = \frac{1}{2}\sum_{k=1}^{N}\left(y(k) - \phi(k-1)^T\theta\right)^2 + \frac{1}{2}(\theta - \theta(0))^T P_0^{-1}(\theta - \theta(0)) \tag{2.8}$$

The algorithm is written as follows:

$$\theta(k) = \theta(k-1) + \frac{P(k-2)\,\phi(k-1)}{1 + \phi(k-1)^T P(k-2)\,\phi(k-1)}\left[y(k) - \phi(k-1)^T\theta(k-1)\right] \tag{2.9}$$

$$P(k-1) = P(k-2) - \frac{P(k-2)\phi(k-1)\phi(k-1)^T P(k-2)}{1 + \phi(k-1)^T P(k-2)\phi(k-1)} \tag{2.10}$$

with $\theta(0)$ given and $P(-1)$ any positive definite matrix P_0.

The least squares algorithm is almost identical to the orthogonalized projection algorithm. The covariance matrix P is an indication of the parameter convergence and a measure the estimation error $e(k)$. Usually the P_0 diagonal coefficient in P is a large number due to the poor confidence in the starting parameter vector $\theta(0)$. Only when $\theta(0)$ is a reliable estimate is P_0 given smaller values to enable faster convergence.

The algorithm is a fast converging one initially, but, when the P matrix diagonal elements begin to get smaller due to the convergence of the estimates, the algorithm becomes more and more insensitive. Remedies that deal with the problem, modify the covariance matrix to sustain the agility of the algorithm. Some variations to the original least square algorithm that deal with the algorithm "falling asleep"[14] and with estimation in the presence of bounded noise are presented next.

2.4.1 Least Squares with Covariance Resetting

This scheme is based on the resetting of the covariance matrix P at various intervals. During the time intervals $k_1, k_2, k_3, \ldots$ the covariance matrix is reset to:

$$P(k_i - 1) = N_i I \tag{2.11}$$

where N_i is a constant ranging between $0 < N_i < \infty$. Otherwise, the normal least squares update is used.

2.4.2 Least Squares with Exponential Data Weighting

The least squares algorithm can be modified by enabling memory to purge itself from redundant data. The weighted least squares method achieves that by minimizing the weighted cost function.

$$J = \sum_{k=1}^{N} \lambda^{N-k}\,|e(k)|^2 \tag{2.12}$$

where:

$$e(k) = y(k) - \phi(k-1)^T\theta(k)$$

and the forgetting factor λ is a constant ranging between 0 and 1.

The modified algorithm appears as:

$$\theta(k) = \theta(k-1) + \frac{P(k-2)\,\phi(k-1)}{\lambda(k-1) + \phi(k-1)^T P(k-2)\,\phi(k-1)}.e(k) \tag{2.13}$$

$$P(k-1) = \frac{1}{\lambda(k-1)}\left[P(k-2) - \frac{P(k-2)\,\phi(k-1)\,\phi(k-1)^T P(k-2)}{\lambda(k-1) + \phi(k-1)^T P(k-2)\,\phi(k-1)}\right] \tag{2.14}$$

For most cases, Isermann [15] states that the range of the forgetting factor is between $0.95 \leq \lambda \leq 0.995$.

The number of historical samples N that are significant to the estimate of the parameters can be approximated by:

$$N \approx \frac{1}{1-\lambda} \tag{2.15}$$

The performance of the algorithm improves according to Soderstrom, Ljung, and Gustavsson [16] due to the increase in the weight,

$$\lambda(k) = \lambda_0\,\lambda(k-1) + (1-\lambda_0) \tag{2.16}$$

which combined with λ^{N-k} gives

$$\lambda(k) = \lambda_0\,\lambda(k-1) + \lambda(1-\lambda_0) \tag{2.17}$$

with $\lambda_0 < 1$ and $\lambda(0) < 1$.

The time dependent forgetting factor discards data during initial estimation at time increments. When λ reaches unity, the normal recursive least squares algorithms emerges.

2.4.3 Least Squares with Deadzone

The robustness of the least squares algorithm when operating in the presence of bounded noise can be improved with the use of a deadzone in the parameter update equation. Additionally, the introduction of deadzone in the algorithm helps to control "bursting", where due to lack of persistent excitation the gain increases quickly resulting in erroneous parameter estimates.

A recursive least squares algorithm with dead zone is defined as:

$$\theta(k) = \theta(k-1) + \frac{a(k-1)\,P(k-2)\,\phi(k-1)}{1 + a(k-1)\,\phi(k-1)^T P(k-2)\,\phi(k-1)}\left[y(k) - \hat{y}(k)\right] \tag{2.18}$$

where:

$$\hat{y}(k) = \phi(k-1)^T\,\theta(k-1)$$

is the estimated plant output and

$$P(k-1) = P(k-2) - \frac{a(k-1)\,P(k-2)\,\phi(k-1)\,\phi(k-1)^T P(k-2)}{1 + a(k-1)\,\phi(k-1)^T P(k-2)\,\phi(k-1)} \tag{2.19}$$

with the initial parameter vector $\theta(0)$ given and $P(-1)$ any positive definite matrix P_0.

The factor $a(k-1)$ is defined as:

$$a(k-1) = \begin{cases} 1, & \text{if} \frac{e(k)^2}{1+\phi(k-1)^T P(k-2)\phi(k-1)} > \Delta^2 > 0; \\ 0, & \text{otherwise.} \end{cases}$$

where:

Δ is a constant error limit and

$$e(k) = y(k) - \hat{y}(k)$$

is the estimation error.

2.5 Summary

In this chapter, a number of on-line parameter estimation techniques have been described. The formulation of all the algorithms was based on the DARMA model. Although the methods were presented for deterministic systems, they also produce good results when used for stochastic estimation purposes. The recursive least squares algorithm being the most commonly used one was presented in more detail. Also, presented were some important modifications to the basic least squares which improve its performance and robustness for specific problem areas.

Chapter 3

Control Principles

3.1 Introduction

The chapter presents some current control design strategies based on *discrete-time* theory. In general terms, the techniques can be separated into two main categories. The first is the minimum prediction error controller category. The second is the closed-loop pole assignment category.

The minimum prediction error method designs controllers which generate an output at the present instant of time which forces the future output of the system to obtain some predefined value. The close-loop pole assignment method assigns the closed-loop poles to some desired locations trying to accommodate some predefined design specifications.

The control schemes discussed here address the linear deterministic systems case.

3.2 Linear Deterministic System Model in Predictor Form

A single input single output (SISO) linear deterministic system can be represented by the following deterministic autoregressive moving averages (DARMA) model:

$$A(q^{-1})\,y(k) = B(q^{-1})\,u(k) \tag{3.1}$$

where:

$$\begin{aligned} A(q^{-1}) &= 1 + a_1\,q^{-1} + a_2\,q^{-2} + \cdots + a_n\,q^{-n} \\ B(q^{-1}) &= (b_0 + b_1\,q^{-1} + \cdots + b_m q^{-m})\,q^{-d} \\ &= q^{-d}\,B^{'}(q^{-1}) \end{aligned}$$

and d is the time delay.

The aim is to use the past and present values available to predict a future value at a time d.

According to the *Prediction Equality*, there exist the unique polynomials.

$$F(q^{-1}) = 1 + f_1\,q^{-1} + \cdots + f_{d-1}\,q^{-d+1} \tag{3.2}$$

$$G(q^{-1}) = g_0 + g_1\,q^{-1} + \cdots + g_{n-1}\,q^{-n+1} \tag{3.3}$$

that satisfy

$$1 = F(q^{-1})A(q^{-1}) + q^{-d}\,G(q^{-1}) \tag{3.4}$$

The Equation 3.4 can be expressed as

$$\frac{1}{A(q^{-1})} = F(q^{-1}) + q^{-d}\,\frac{G(q^{-1})}{A(q^{-1})} \tag{3.5}$$

The original DARMA model can be expressed in the following predictor form:

$$y(k+d) = \alpha(q^{-1})\,y(k) + \beta(q^{-1})\,u(k) \tag{3.6}$$

where:

$$\alpha(q^{-1}) = \alpha_0 + \alpha_1\,q^{-1} + \cdots + \alpha_{n-1}\,q^{-(n-1)}$$

$$\beta(q^{-1}) = \beta_0 + \beta_1\,q^{-1} + \cdots + \beta_{m+d-1}\,q^{-(m+d-1)}$$

and

$$\alpha(q^{-1}) = G(q^{-1})$$

$$\beta(q^{-1}) = F(q^{-1})B^{'}(q^{-1})$$

The above equation is the manner in which the DARMA model is expressed in predictor form. It forms the basis for the design of prediction controllers.

3.3 Minimum Prediction Error Controllers

This section discusses techniques for designing controllers based on the d-step-ahead prediction form.

3.3.1 One-Step-Ahead Control

The one-step-ahead control algorithm forces the plant output $y(k)$ to be equal to the setpoint $y^*(k+d)$ in one step (in one sampling period). The controller that matches the $y(k)$ to the $y^*(k+d)$ at time $k+d$ has the form

$$\beta(q^{-1})\,u(k) = y^*(k+d) - \alpha(q^{-1})\,y(k) \tag{3.7}$$

The closed-loop system is expressed as

$$\begin{aligned} y(k) &= y^*(k) \quad k \geq d && (3.8)\\ B(q^{-1})\,u(k) &= A(q^{-1})\,y^*(k) && (3.9)\end{aligned}$$

The one-step-ahead control law minimizes the quadratic cost function

$$J(k+d) = \frac{1}{2}|e(k)|^2 \tag{3.10}$$

with

$$e(k) = y(k+d) - y^*(k+d) \tag{3.11}$$

A stability requirement is that the zeros of $B(q^{-1})$, which are the poles of the closed-loop system, are inside the unit circle. A concern when using the one-step-ahead algorithm is that it generates a large signal to drive the plant output to match the setpoint, which often is not desirable.

3.3.2 Weighted One-Step-Ahead Control

The weighted one-step-ahead control is given by

$$u(k) = \frac{\beta_0[y^*(k+d) - \alpha(q^{-1})\,y(k) - \beta'(q^{-1})\,u(k-1)]}{\beta_0^2 + \lambda} \tag{3.12}$$

with

$$\begin{aligned} \beta'(q^{-1}) &= q[\beta(q^{-1}) - \beta_0]\\ &= \beta_1 + \beta_2 q^{-1} + \cdots + \beta_{m+d-1}\, q^{-(m+d-2)}\end{aligned}$$

The closed-loop system response is given by

$$\begin{aligned} P(q^{-1})y(k+d) &= B'(q^{-1})y^*(k+d) && (3.13)\\ P(q^{-1})u(k) &= A(q^{-1})y^*(k+d) && (3.14)\end{aligned}$$

with

$$P(q^{-1}) = B'(q^{-1}) + \frac{\lambda}{\beta_0} A(q^{-1})$$

The control law minimizes the cost function

$$J(k+d) = \left\{ \frac{1}{2}(y(k+d) - y^*(k+d))^2 + \frac{\lambda}{2} u(k)^2 \right\} \tag{3.15}$$

The modification restricts the controller signal strength and assists in avoiding system oscillations in between samples. When $\lambda = 0$, the algorithm reduces to the basic one-step-ahead algorithm.

3.3.3 Model Reference Control

The system again is represented by the DARMA model

$$A(q^{-1})\, y(k) = B(q^{-1})\, u(k) \tag{3.16}$$

The requirement is that the system output $y(k)$ follows the setpoint $y^*(k)$ generated by a reference model which in term is driven by a reference input $r(k)$.

The reference model is represented by

$$E(q^{-1})\, y^*(k) = q^{-d'}\, H(q^{-1})\, r(t)$$

with

$$\begin{aligned} H(q^{-1}) &= 1 + h_1 q^{-1} + \cdots + h_l q^{-l} && (3.17) \\ E(q^{-1}) &= 1 + e_1 q^{-1} + \cdots + e_l q^{-l} && (3.18) \end{aligned}$$

A requirement is that the roots of $E(q^{-1})$ are inside the unit circle.

The aim of the algorithm is to try and make the process output $y(k)$ identical to the reference model output $y^*(k)$, so that

$$y(k) = y^*(k)$$

and

$$E(q^{-1})\, y(k) = q^{-d}\, H(q^{-1})\, r(k) \tag{3.19}$$

An illustration of the control method is given in Figure 3.1. To design the control law, we follow a similar approach to the one-step-ahead control. First, the $E(q^{-1})y(k)$ is predicted and then set equal to $q^{-d}\, H(q^{-1})\, r(k)$ to generate Equation 3.19.

Using the generalized prediction equality

$$E(q^{-1}) = F(q^{-1})\, A(q^{-1}) + q^{-d}\, G(q^{-1}) \tag{3.20}$$

and multiplying the DARMA model by $F(q^{-1})$ we extract the predictor form

$$E(q^{-1})y(k+d) = \alpha(q^{-1})y(k) + \beta(q^{-1})u(k) \tag{3.21}$$

with:

$$\begin{aligned}\alpha(q^{-1}) &= G(q^{-1}) \\ \beta(q^{-1}) &= F(q^{-1})B'(q^{-1})\end{aligned}$$

and, also, the controller which is described by

$$\beta(q^{-1})u(k) = H(q^{-1})r(k) - \alpha(q^{-1})y(k) \tag{3.22}$$

The model reference control can be seen as being a generalization of the one-step-ahead control philosophy aiming at limiting the controller driving signal.

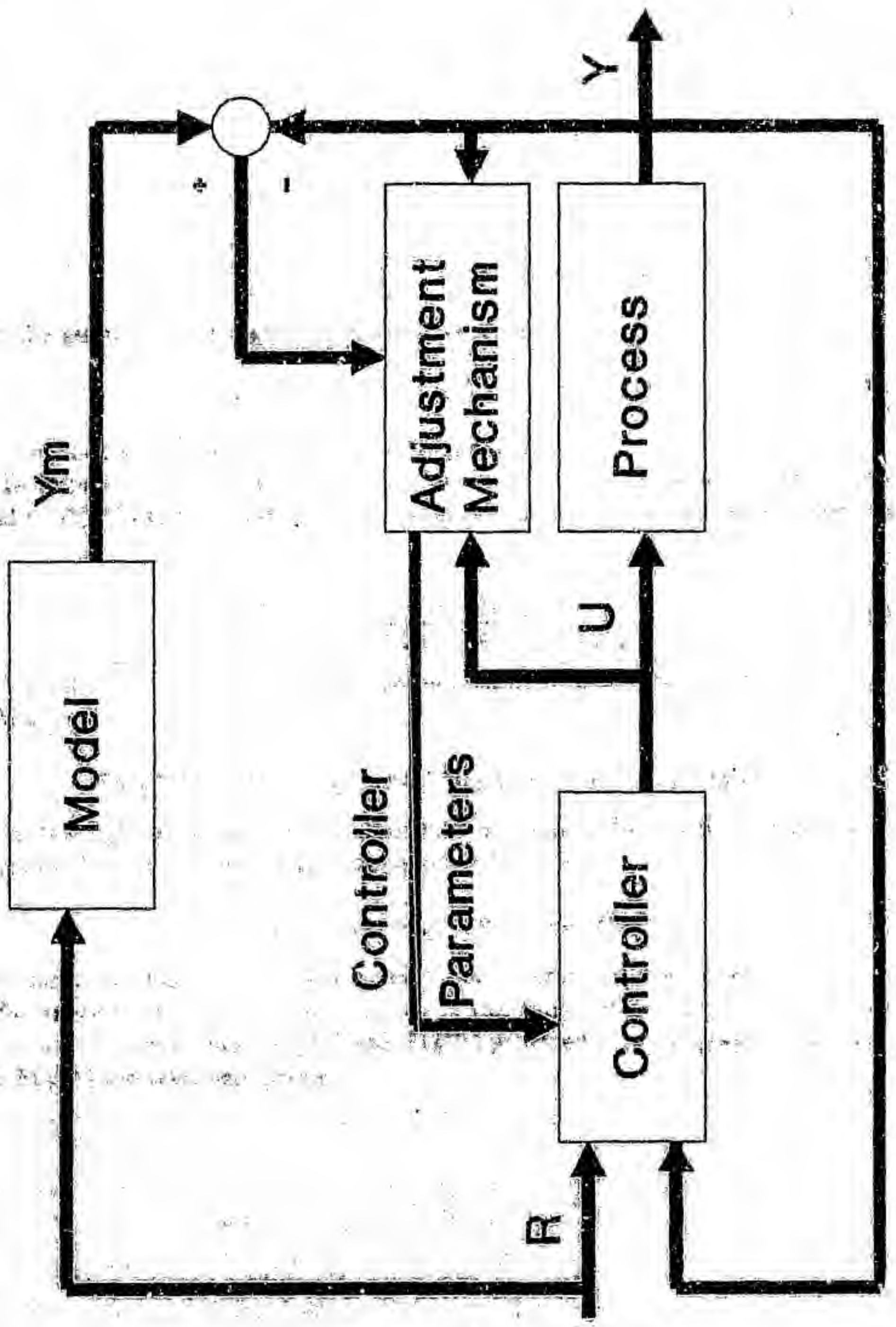


Figure 3.1: Model Reference Control System

3.4 Closed-Loop Pole Assignment Controllers

Pole assignment is rather a simple direct design method. The philosophy behind the algorithm is to generate a feedback control law so that the closed loop system has the desired properties specified by the designer.

The system model is described by a DARMA model

$$A(q^{-1})y(k) = B(q^{-1})u(k) \tag{3.23}$$

The general feedback structure is given by

$$\frac{y(k)}{y^*(k)} = \frac{M(q^{-1})B(q^{-1})}{L(q^{-1})A(q^{-1}) + P(q^{-1})B(q^{-1})} \tag{3.24}$$

with the general pole assignment control law given by

$$L(q)u(k) = -P(q^{-1})y(k) + M(q^{-1})y^*(k) \tag{3.25}$$

The feedback structure is illustrated in Figure 3.2.

To improve the system's sensitivity properties, the polynomial $M(q^{-1})$ is set equal to $P(q^{-1})$ thus, resulting in the usual closed-loop form with the transfer function

$$\frac{y(k)}{y^*(k)} = \frac{P(q^{-1})B(q^{-1})}{L(q^{-1})A(q^{-1}) + P(q^{-1})B(q^{-1})} \tag{3.26}$$

as illustrated by Figure 3.3. It can also be presented in the following form as shown by MacLeod [17].

$$[A(q^{-1})L(q^{-1}) + B(q^{-1})P(q^{-1})]y(k) = P(q^{-1})B(q^{-1})y^*(k) \tag{3.27}$$

When the right-hand side of Equation 3.27 is replaced by the desired closed loop polynomial, $A^*(q^{-1})$, we get the Diophantine Equation.

$$L(q^{-1})A(q^{-1}) + P(q^{-1})B(q^{-1}) = A^*(q^{-1}) \tag{3.28}$$

This indicates that the closed-loop poles can be arbitrarily assigned to satisfy the performance requirements. The solution of the Diophintine Equation will produce the controller polynomials $L(q^{-1})$ and $P(q^{-1})$ provided that the polynomials $A(q^{-1})$ and $B(q^{-1})$ are relatively prime.

Figure 3.2: General Feedback Loop

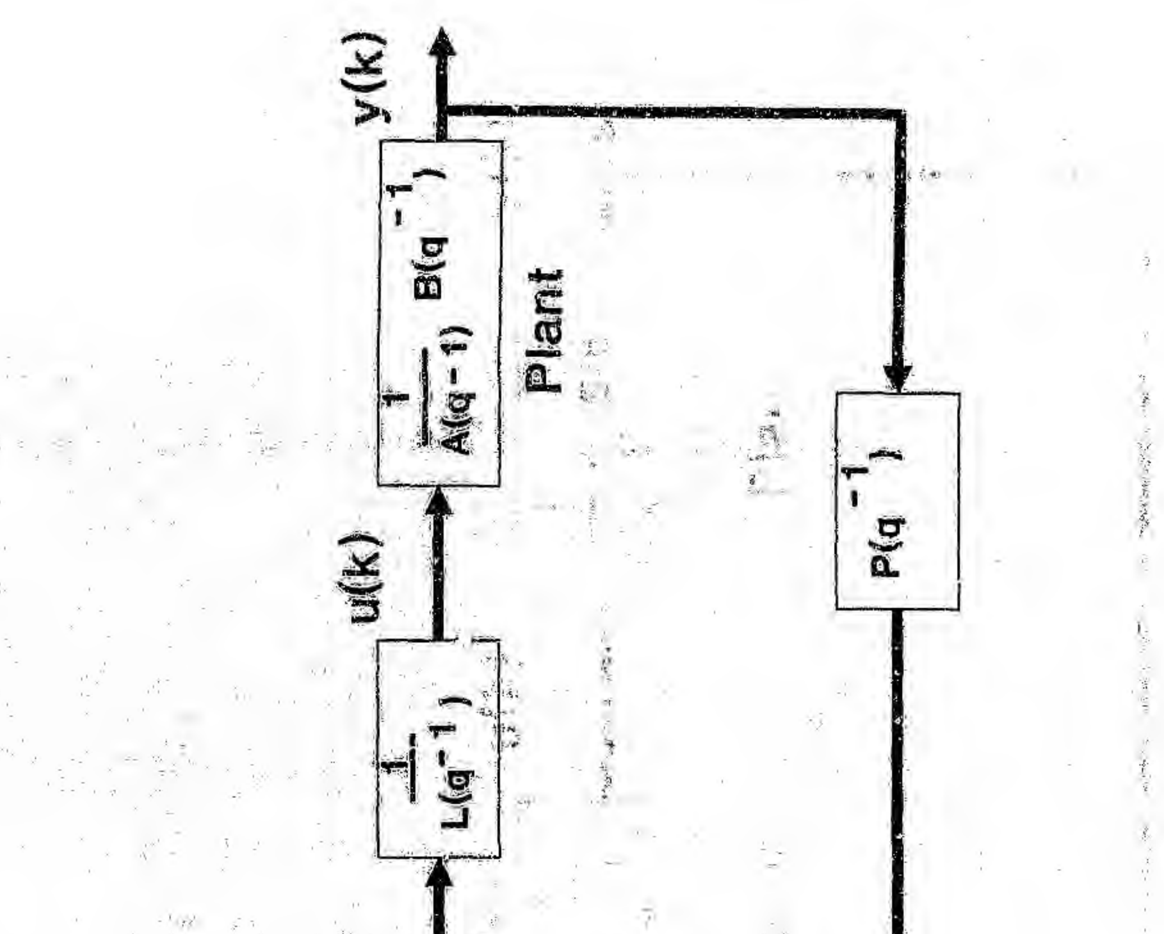


Figure 3.2: General Feedback Loop

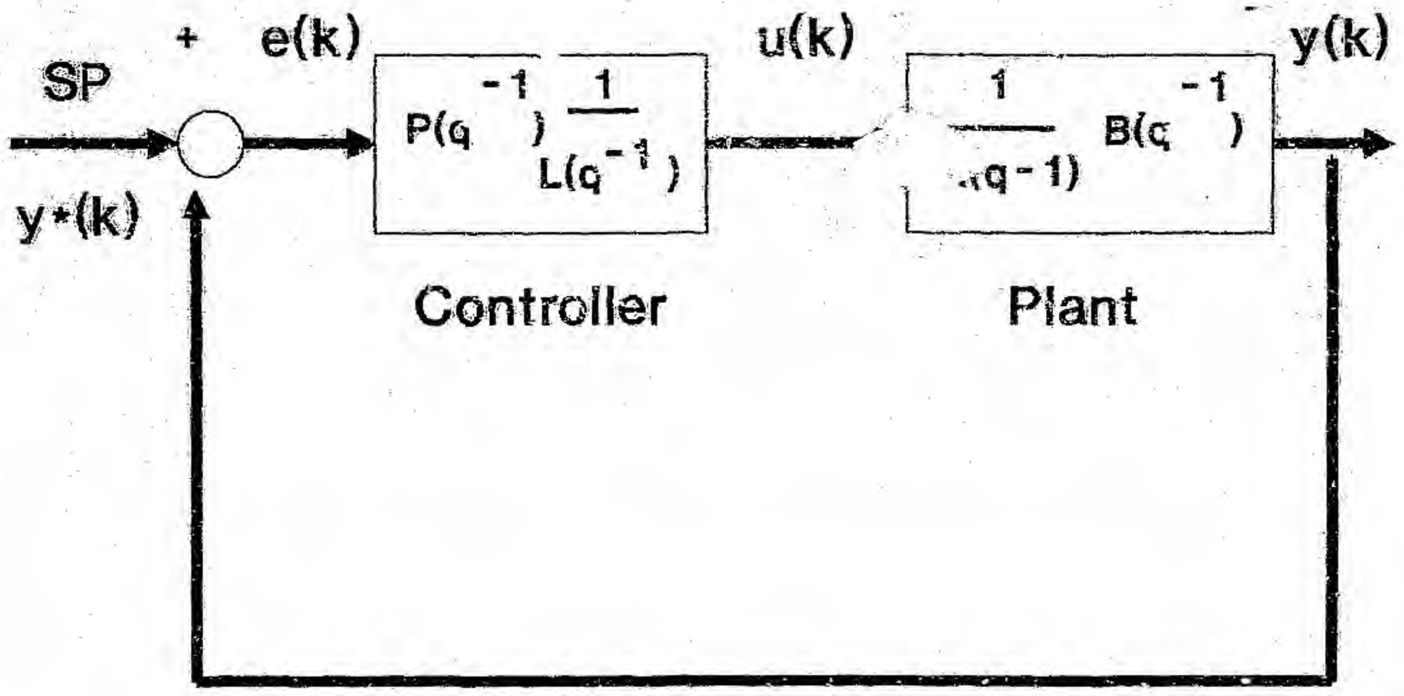

SP
+
e(k)
y*(k)
$P(q^{-1})\frac{1}{L(q^{-1})}$
Controller
u(k)
$\frac{1}{A(q^{-1})}B(q^{-1})$
Plant
y(k)

Figure 3.3: Closed-Loop Control

3.5 Summary

This chapter presented some controller design techniques for linear deterministic systems. Also, examined were some areas for which the described algorithm could be applied. It is clear that using the control schemes is not a straightforward procedure, and attention must be given to the constraints that govern each case. The pole placement is the most preferred technique of all due to its simplicity. Also, a point that should be made is that all the techniques presented in this chapter are special cases of the pole placement algorithm.

Chapter 4

Discrete Control Using Delta-Operators

4.1 Introduction

The chapter presents the incremental difference operator or delta-operator introduced by Goodwin et al[18].It offers a number of advantages which can be gained by using the delta-operator as an alternative to the discrete shift-operator.

The delta-operator concept is not something new. It was used in the past in applications such as motivating the z-transforms, improving digital filter behavior, and to improve finite word length characteristics regarding controller design, roundoff noise and coefficient representation as recently presented by Middleton and Goodwin[19].The delta-operator can, also, be used to resolve problems that arise from the sampling of continuous systems and converting continuous transfer functions to their discrete counterparts.

4.2 The Delta-Operator

The delta operator is defined by:

$$\delta = \frac{q-1}{\Delta} \tag{4.1}$$

where q is the usual forward shift-operator and Δ is the sampling interval.

Equation 4.1 represents δ as a Euler approximation to the derivative-operator $D \equiv \frac{d}{dt}$ which as indicated by MacLeod[20] is proven by:

$$\begin{aligned} D\,x(t) &\simeq \frac{x(k+\Delta)-x(k)}{\Delta} \\ &= \frac{q-1}{\Delta}x(k) \\ &= \delta\,x(k) \end{aligned} \tag{4.2}$$

The above relationship shows that assuming fast sampling, the derivative operator can be approximated by the delta-operator for purposes of translating analog designs.

The delta-operator can be represented as:

$$\delta^{-1} = \frac{\Delta}{q-1} = \frac{\Delta q^{-1}}{1-q^{-1}} \tag{4.3}$$

The above formula establishes that the δ-operator can substitute the backward shift-operator q^{-1} for plant model representation. Analyzing the expression below, we have

$$\begin{aligned} y_{out}(k) &= \delta^{-1}u_{in}(k) \\ &= \frac{\Delta q^{-1}}{1-q^{-1}}u_{in}(k) \end{aligned}$$

and

$$\begin{aligned} (1-q^{-1})y_{out}(k) &= \Delta q^{-1}u_{in}(k) \quad \text{so} \\ y_{out}(k) - q^{-1}y_{out}(k) &= \Delta q^{-1}u_{in}(k) \end{aligned}$$

finally

$$y_{out}(k) = y_{out}(k-1) + \Delta u_{in}(k-1) \tag{4.4}$$

which indicates that the δ^{-1} operator can be used to build a discrete integrator block illustrated in Figure 4.1.

An important point to be made is that the replacement of the shift-operator with the δ-operator preserves the degree of the polynomial in q and the degree of its associated transfer function.

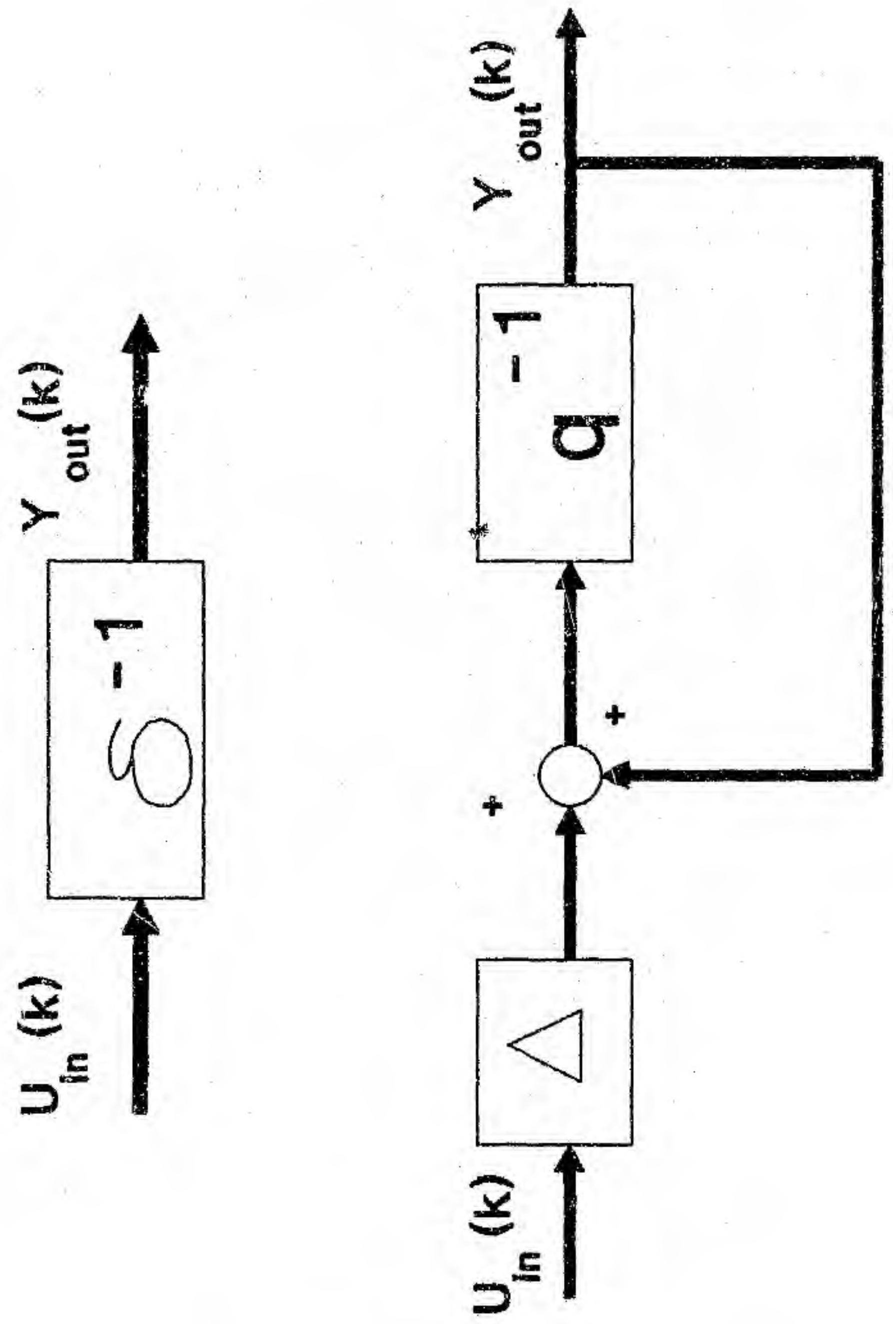


Figure 4.1: The Discrete Integrator

4.3 The Delta-Operator Transform Domain

The transformation of the δ-operator is defined as the γ- transform. A distinction between continuous and discrete models regarding stability is that for the former, the system poles must be in the left plane of the S domain where the later requires that the poles are inside the unit circle. In the γ-transform domain, this problem is overcome. The stability region of the γ-transform is the inside of a circle with center $-\frac{1}{\Delta}$ and radius $\frac{1}{\Delta}$. It is obvious that as the sampling interval decreases or the sampling rate increases, the stability region increases and approaches the continuous stability region as illustrated in Figure 4.2.

The definition of the γ-transform is given by:

$$F_\gamma(\gamma) = \Delta F_z(1+\Delta\gamma) = \Delta\sum_{i=0}^{\infty} f(k\Delta)(1+\Delta\gamma)^{-k} \tag{4.5}$$

where F_z is the z-transform and z is replaced by $1+\Delta\gamma$ and it represents the transform of the δ-operator models where $q = 1+\Delta\delta$.

It has been proven that the γ-transform converges to the s-transform as the sampling rate increases. This is shown by:

$$\lim_{\Delta\to 0}\{F_\gamma(\gamma)\} = F_s|_{s=\gamma} = \int_0^\infty f(t)e^{-\gamma t}dt \tag{4.6}$$

where the integral is a Riemann integral.

The standard formula for replacing a q model with a δ-model is:

$$A(\delta) = \frac{1}{\Delta^n}A'(\Delta\delta+1) \tag{4.7}$$

where:

$$A' = A(q) = q^n + a_{n-1}q^{n-1} + \cdots + a_0$$

and n the order of the polynomial $A(q)$.

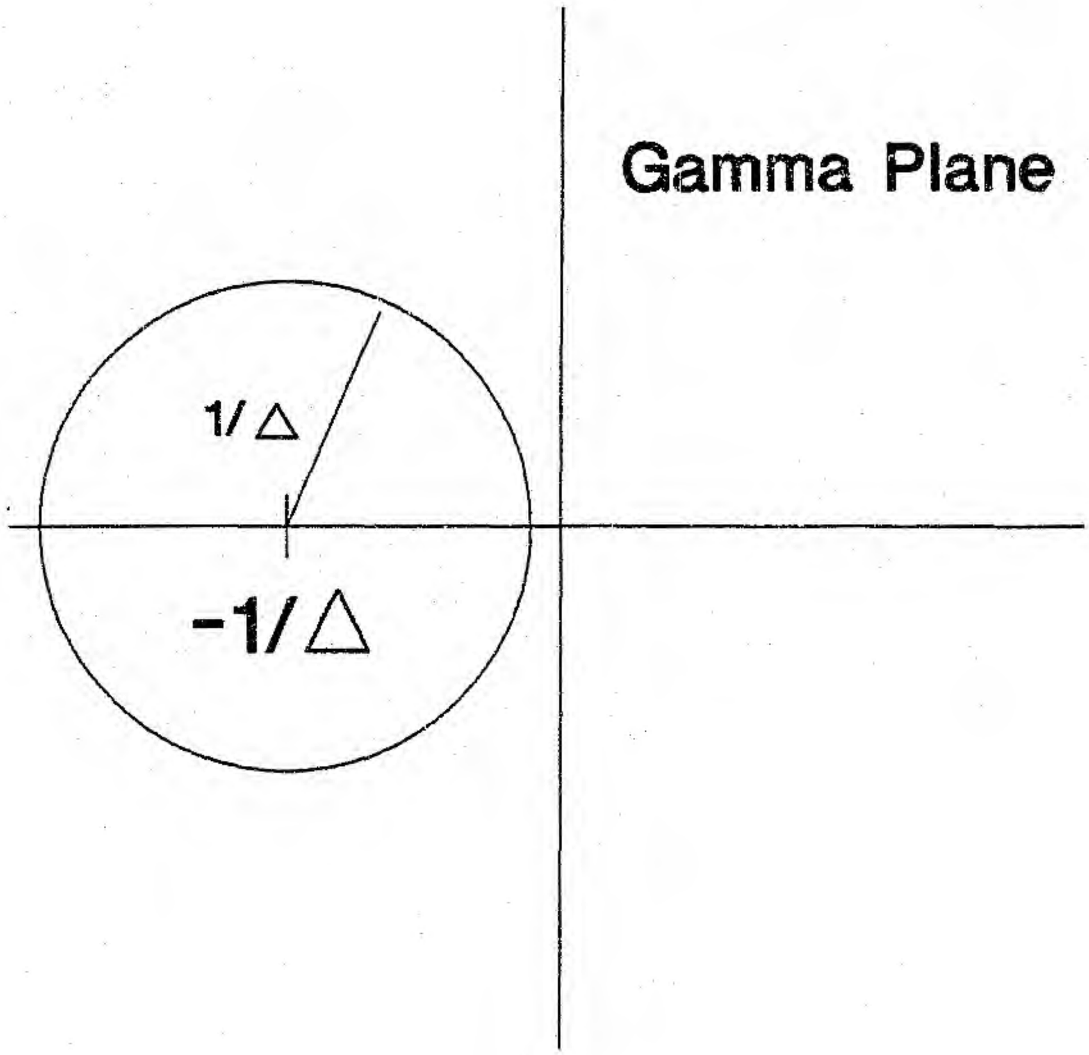


Figure 4.2: Stability Region of the γ-Transform

4.4 Summary

This chapter described the δ-operator and its associated γ-transform It discussed the principles behind the δ-operator and the advantage it has over the shift-operator and its z-transform in various applications. An important issue is that the replacement of a system model expressed in a shift-operator by δ-operator model does not affect the order and relative degree of the model. Also, presented was the rapprochement between discrete and continuous control where the γ-transform is the catalyst.

Chapter 5

Design of the Regulator Tuning System

5.1 Introduction

This chapter presents the approach followed for the design of the robust controller tuner. The system is comprised of two parts. The first is the robust parameter estimator and the second, the controller design. Separating the system into these two distinct parts results in a so-called explicit algorithm.

The recursive least squares algorithm was identified as a suitable identification method for determining the process parameters. It was used for developing the structure of the parameter identification algorithm. Modifications were performed on the basic least squares scheme to improve its performance.

The second step of the explicit algorithm is to use the parameters estimated from the first step to determine the regulator parameters. The design method for the second step must be a suitable and robust method. The closed-loop pole placement algorithm described by Aström and Wittenmark[21] has been shown to fulfill these requirements. Due to its robustness and simplicity, the pole placement algorithm was identified as the most suitable controller design algorithm for determining the regulator parameters.

The complete structure of the regulator tuner is formulated based on the δ-operator.

5.2 Parameter Estimator

The process model given in terms of the δ-operator is described by the DARMA model

$$A(\delta)\, y(k) = B(\delta)\, u(k) + F(\delta)\, z(k) + d(k) + \xi(k) \tag{5.1}$$

The expression includes both deterministic disturbance and random disturbance and modeling errors.

- $z(k)$ = measurable disturbance
- $d(k)$ = deterministic disturbance
- $\xi(k)$ = random disturbance/modeling error

The polynomial degrees are:

$$\begin{aligned} deg\, A(\delta) &= n \\ deg\, B(\delta) &= m \leq n-1 \\ deg\, F(\delta) &= r \leq n \end{aligned}$$

Polynomials $A(\delta)$, $B(\delta)$, and $F(\delta)$ are monic described as

$$\begin{aligned} A(\delta) &= \delta^n + a_{n-1}\,\delta^{n-1} + \cdots + a_1\,\delta + a_0 \\ B(\delta) &= \delta^m + b_{m-1}\,\delta^{m-1} + \cdots + b_1\,\delta + b_0 \\ F(\delta) &= \delta^r + f_{r-1}\,\delta^{r-1} + \cdots + f_1\,\delta + f_0 \end{aligned}$$

Inclusion of the measurable disturbance term $z(k)$ in the model, allows the development of a feedforward model which can be used for determining feedforward regulators.

The unmeasurable deterministic disturbance signal $d(k)$ can be modeled as:

$$D(\delta)\, d(k) = 0 \tag{5.2}$$

where $D(\delta)$ is a monic polynomial with non-repeated roots on the stability boundary. It is a nulling polynomial since it eliminates the deterministic signal.

For a constant disturbance signal having the form

$$d(k+1) = d(k)$$

MacLeod[22] has shown that

$$\begin{aligned} d(k+1) &= d(k) \\ d(k+1) - d(k) &= 0 \\ (q-1)\, d(k) &= 0 \end{aligned}$$

It shows that the nulling polynomial in q-operator is

$$D(q) = q - 1 \tag{5.3}$$

and D expressed in δ-operator form becomes

$$D(\delta) = \delta \tag{5.4}$$

where:

$$\delta = \frac{q-1}{\Delta}$$

Along the same lines, it is shown that for a sinusoidal disturbance signal

$$d(k) = A\sin(w_k k + \phi) \tag{5.5}$$

the D polynomial in δ-operator is

$$D(\delta) = \delta^2 + \frac{2w\delta}{\Delta} + \frac{2w^2}{\Delta^2} \tag{5.6}$$

where:

$$w = 1 - \cos w_k$$

Operating on both sides of Equation 5.1 by $D(\delta)$, the random deterministic disturbance $d(k)$ is eliminated. The expression becomes

$$A(\delta)\,D(\delta)\,y(k) = B(\delta)\,D(\delta)\,u(k) + F(\delta)\,D(\delta)\,z(k) + D(\delta)\,\xi(k) \tag{5.7}$$

Since $D(\delta)$ has roots on the stability boundary, the noise/error modeling term $\xi(k)$ could be amplified at high frequencies. The phenomenon can be avoided by introducing a stable polynomial $D'(\delta)$ "close" to the $D(\delta)$ polynomial.

Assuming that the dete·ministic disturbance $d(k)$ is a constant (d. c. offset disturbance) then the nulling polynomial is

$$D(\delta) = \delta$$

The stable polynomial, $D'(\delta)$, then is

$$D'(\delta) = \delta + \varepsilon \tag{5.8}$$

where ε is some small positive number.

The function $\frac{D}{D'}$ removes the deterministic disturbance and does not distort the spectrum. It is obvious that the function $\frac{D}{D'}$ is a high-.ıss filter. By approximating $\delta \simeq jw$, the filter has the frequency transfer function

$$\frac{D(jw)}{D'(jw)} = \frac{jw}{jw + \varepsilon} \tag{5.9}$$

with corner frequency the small number ε.

To avoid having high frequencies present which are above the Nyquist frequency, a low-pass filter is also introduced. It ensures that the band limited model is excited only by frequencies which are necessary to give a good process model. The structure of the filter is described in the next chapter.

The "filtered model" is described by

$$A(\delta)\,y_f(k) = B(\delta)\,u_f(k) + F(\delta)\,z_f(k) + n_f\,(k) \tag{5.10}$$

where y_f, u_t, z_f, n_f are the filtered process measurements.

The DARMA model can be represented in the regression format:

$$\delta^n\,y_f(k) = \phi(k-1)^T\,\theta(k) + n_f(k) \tag{5.11}$$

where

$$\begin{aligned}\phi^T &= [\delta^{n-1} y_f(k), \cdots y_f(k), \delta^m u_f(k) \cdots u_f(k), \delta^r z_f(k) \cdots z_f(k)] & (5.12)\\ \theta &= [-a_{n-1}, \cdots - a_0, b_m, \cdots b_0, f_r \cdots f_0] & (5.13)\end{aligned}$$

This is compatible with the recursive least squares algorithm described in Chapter 2.

To enhance the robustness of the estimator, the algorithm was modified to include both the deadzone $a(k)$ and exponential weight $\lambda(k)$ factors.

The modified recursive least squares algorithm expressed in δ-operator format is given below.

$$\theta(k) = \theta(k-1) + a(k-1) \cdot \frac{P(k-2)\,\phi(k-1)}{\lambda(k-1) + \phi(k-1)^T P(k-2)\,\phi(k-1)} . e(k) \quad (5.14)$$

$$P(k-1) = \frac{1}{\lambda(k-1)} \left[P(k-2) - a(k-1) . \frac{P(k-2)\,\phi(k-1)\,\phi(k-1)^T P(k-2)}{\lambda(k-1) + \phi(k-1)^T P(k-2)\,\phi(k-1)}\right] \quad (5.15)$$

with

$$e(k) = y_f(k) - \hat{y}_f(k) \quad (5.16)$$

with $\hat{y}_f(k)$ the estimated plant output.

Finally, to improve the numerical sensitivity of the algorithm the Bierman[23] UDU^T factorization technique is used. The technique is presented in Appendix A modified to include the deadzone and exponential forgetting factors.

5.3 Controller Design

The design of the controller emerges from the use of the pole placement algorithm described in Chapter Two and the application of the internal model principle. The common error driven control system including an output disturbance $d(k)$, as illustrated in Figure 5.1, is the principal block diagram the design is based on.

The setpoint $y^*(k)$ is defined as a sequence that can be generated through a linear finite dimensional system described by

$$S(\delta)\, y^*(k) = 0 \tag{5.17}$$

where $S(\delta)$ is a monic polynomial with its roots lying on the stability boundary.

The nulling polynomial, $D(\delta)$, is used to negate the output disturbance, $d(k)$, as it was described in the previous section.

The feedback control law used in developing the design is given by

$$L(\delta)\, u(k) = M(\delta)\, y^*(k) - P(\delta)\, y(k) \tag{5.18}$$

where

$$\begin{aligned} L(\delta) &= \delta^{\pi} + l_{\pi-1}\, \delta^{\pi-1} + \cdots + l_1\, \delta + l_0 \\ P(\delta) &= \delta^{\rho} + p_{\rho-1}\, \delta^{\rho-1} + \cdots + p_1\, \delta + p_0 \end{aligned}$$

Applying the Internal Model Principle[24], we assign

$$\begin{aligned} M(\delta) &= P(\delta) \\ L'(\delta) &= L(\delta)\, D(\delta)\, S(\delta) \end{aligned}$$

thus modifying the controller poles to include the reference model poles $S(\delta)$ and the disturbance poles $D(\delta)$. The control law becomes

$$L(\delta)\, D(\delta)\, S(\delta)\, u(k) = P(\delta)\, [y^*(k) - y(k)] \tag{5.19}$$

or in a more compact form

$$L'(\delta)\, u(k) = P(\delta)\, e(k) \tag{5.20}$$

where $e(k) = y^*(k) - y(k)$ is the feedback error signal.

In a not strictly manner, the modified closed-loop illustrated in Figure 5.2 can be represented by the transfer function

$$\frac{y}{y^*} = G(\delta) = \frac{PB}{L'A + PB} \tag{5.21}$$

The characteristic equation of the closed-loop transfer function is the closed-loop pole assignment equation

$$L(\delta)\, D(\delta)\, S(\delta)\, A(\delta) + P(\delta)\, B(\delta) = A^*(\delta) \tag{5.22}$$

The $A^*(\delta)$ term is defined as the closed-loop characteristic polynomial which defines the desired location of the closed-loop poles.

The coefficients of $L(\delta)$ and $P(\delta)$ are determined by solving the Equation 5.22.

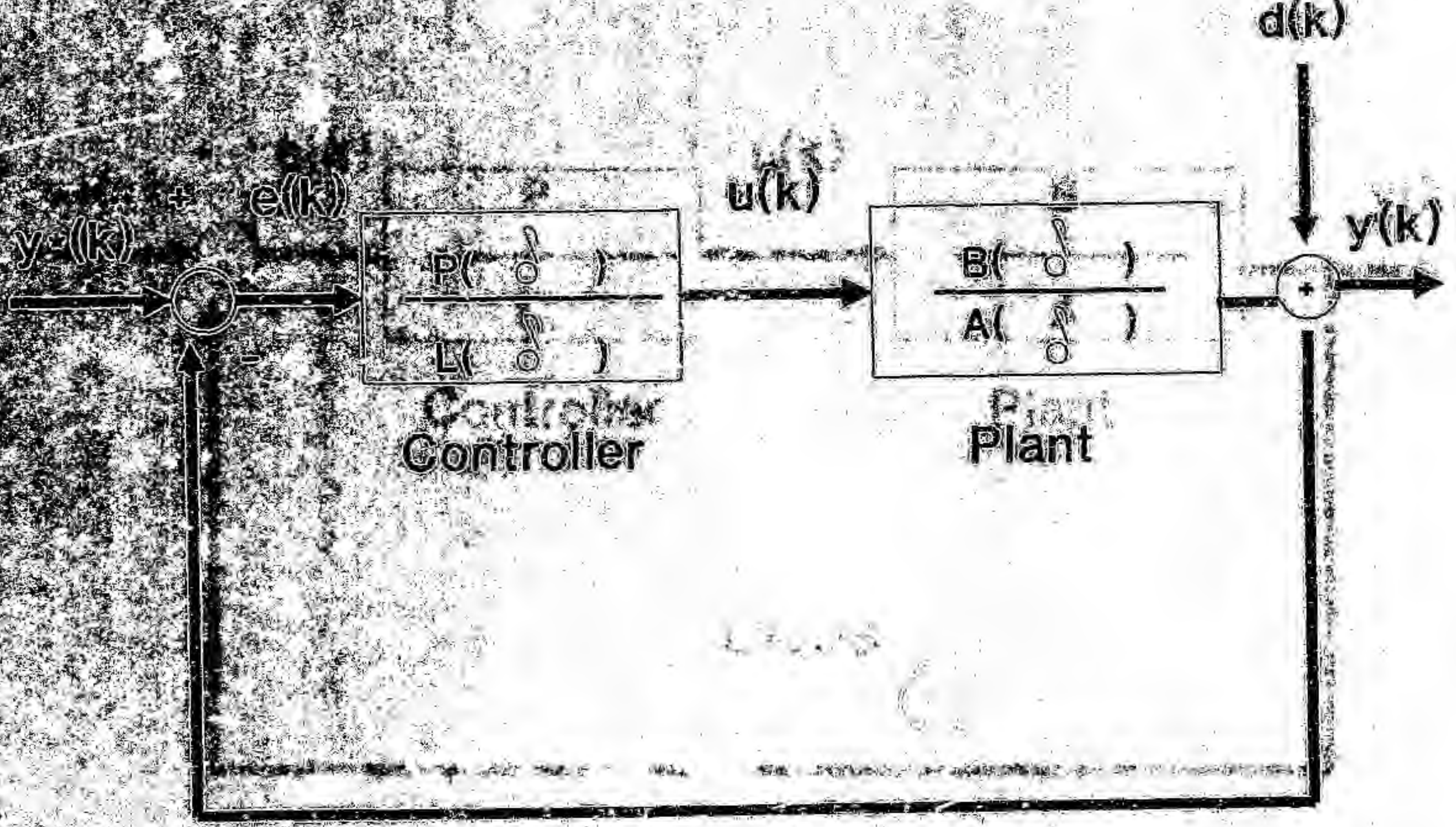


Figure 5.1: Closed-loop System with Output Disturbance

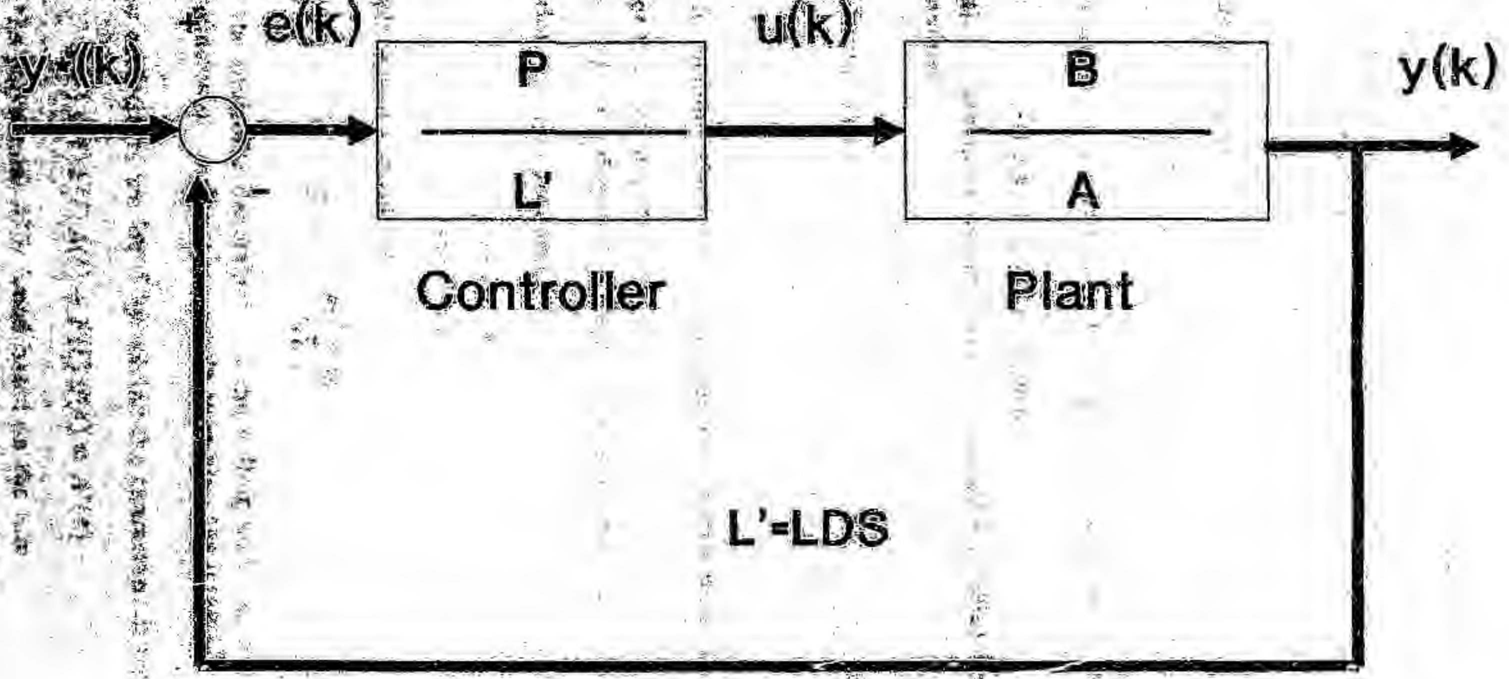


Figure 5.2: The Closed Loop System with D and S Included

5.3.1 Controller Design for First-Order Plant

A wide range of chemical process plants can adequately be modeled by first-order models. When the pole placement algorithm developed previously is worked out based on a first-order model, the algorithm develops as follows.

The control law is represented by

$$L'(\delta)\,u(k) = P(\delta)\,e(k) \tag{5.23}$$

or in a transfer function format

$$\frac{u(k)}{e(k)} = G_c = \frac{P}{L'} = \frac{P}{LDS} \tag{5.24}$$

The process is represented as a first-order model given by the linear deterministic equation

$$\delta\,y(k) + a_0\,y(k) = b_0\,u(k) \tag{5.25}$$

The deterministic disturbance is assumed to be a constant d.c. offset disturbance nulled by the polynomial

$$D(\delta) = \delta$$

The setpoint reference model is

$$S = 1$$

A general rule for defining the degree of the controller polynomials L and P is

$$\begin{aligned} degree\,L &= degree\,A - 1 \\ degree\,P &= degree\,A + degree\,D + degree\,S - 1 \end{aligned}$$

In this application, the rule produces

$$\begin{aligned} degree\,L = 0 \\ degree\,P = 1 \end{aligned}$$

which results the placement control law

$$G_c(\delta) = \frac{p_1\delta + p_0}{\delta(l_0)} \tag{5.26}$$

The term δ in the controller denominator is the nulling polynomial $D(\delta)$. It is obvious that the inclusion of the disturbance pole generated the integral term in the controller.

The control law coefficients are found by solving the characteristic equation

$$L(\delta)\,D(\delta)\,S(\delta)\,A(\delta) + P(\delta)\,B(\delta) = A^*(\delta) \tag{5.27}$$

The degree of the polynomial A^* is found based on the rule

$$degreeA^* = 2degree\,A + degree\,D + degree\,S - 1$$

Here the degree of A^* is $A^* = 2$ which results to

$$A^* = \delta^2 + a_1^*\delta + a_0^* \tag{5.28}$$

The Diophantine Equation 5.27 can be expressed as

$$(\delta + a_0)(\delta)(l_0) + (p_1\delta + p_0)(b_0) = \delta^2 + a_1^*\delta + a_0^* \tag{5.29}$$

By rearranging Equation 5.29, we get

$$l_0\delta^2 + a_0 l_0\delta + p_1 b_0\delta + p_0 b_0 = \delta^2 + a_1^*\delta + a_0^* \tag{5.30}$$

Equating coefficients on either side of Equation 5.30 results

$$M_a. \begin{bmatrix} l_0 \\ p_1 \\ p_0 \end{bmatrix} = \begin{bmatrix} 1 \\ a_1^* \\ a_0^* \end{bmatrix}$$

where

$$M_a = \begin{bmatrix} 1 & 0 & 0 \\ a_0 & b_0 & 0 \\ 0 & 0 & b_0 \end{bmatrix}$$

Equation 5.30 can be solved sequentially for l_0, p_1, p_0 by

$$l_0 = 1 \tag{5.31}$$

$$p_1 = \frac{a_1^* - a_0}{b_0} \tag{5.32}$$

$$p_0 = \frac{a_0^*}{b_0} \tag{5.33}$$

The next step is to correlate the pole placement coefficients with those of a PI regulator.

5.4 Approximation to a PI Regulator

The ideal analog PI controller is given by

$$u(t) = K_p \left[e(t) + \frac{1}{T_i}\int e(t)dt\right] \tag{5.34}$$

where K_p is the proportional gain and T_i is the integral or reset time.

Equation 5.34 written in the Laplace domain becomes

$$U(s) = K_p\left(1 + \frac{1}{T_i s}\right)E(s) \tag{5.35}$$

or

$$U(s) = K_p\left(1 + \frac{K_i}{s}\right)E(s) \tag{5.36}$$

where $U(s)$ and $E(s)$ are the Laplace transforms of the controller output and error signal, respectively and K_i is the integral or reset gain and $K_i = \frac{1}{T_i}$.

A typical cascade PI closed-loop is illustrated in Figure 5.3.

The PI regulator operates on the actuating signal and produces one that is proportional to both the magnitude and the integral of this signal. The integral part continues to increase as long as an error signal is present. The locations of the pole and zero of the regulator are illustrated in Figure 5.4.

Examining the PI controller in the frequency domain, it becomes obvious that the regulator has a phase lag frequency response as illustrated in Figure 5.5.

At low frequencies the regulator amplifies the system gain thus reducing the system sensitivity. It reduces the steady state error and improves the stability of the loop.

The Equation 5.36 can be rearranged to:

$$\frac{U_s}{E_s} = G_{PI}(s) = \frac{K_p s + K_p K_i}{s} \tag{5.37}$$

Using the approximation $s = \delta$ (assuming fast sampling), the equation becomes

$$G_{PI}(\delta) = \frac{K_p \delta + K_p K_i}{\delta} \tag{5.38}$$

Comparing Equation 5.38 with Equation 5.26, we have:

$$G_{PI}(\delta) = C_c(\delta)$$

and

$$\frac{K_p\delta + K_p K_i}{\delta} = \frac{p_1\delta + p_0}{\delta} \tag{5.39}$$

It is obvious that the coefficients of the two control laws correspond to each other. The following relationships exist:

$$K_p = p_1 \tag{5.40}$$

$$K_i = \frac{p_0}{p_1} \tag{5.41}$$

or

$$T_i = \frac{1}{K_i} = \frac{p_1}{p_0} \tag{5.42}$$

The proportional and integral gain can be determined by the p_1 and p_0 coefficients. Figure 5.6 illustrates the general block diagram for the generation of the PI parameter.

Another method of approximating the pole placement controller to a PI controller can be seen in Appendix B.

It is apparent that we can use pole placement techniques to tune the PI controller. It is, also, apparent that the closed loop polynomial A^* is of great importance to the algorithm. Correct selection of A^* is essential. The A^* can be generated based on the time and frequency design requirements, i.e. damping ratio , phase margin, etc. The selection of a suitable A^* is given in the next chapter. ditionally, Appendix C presents the formulation of the pole placement for a second-order model and its approximation to a PID controller.

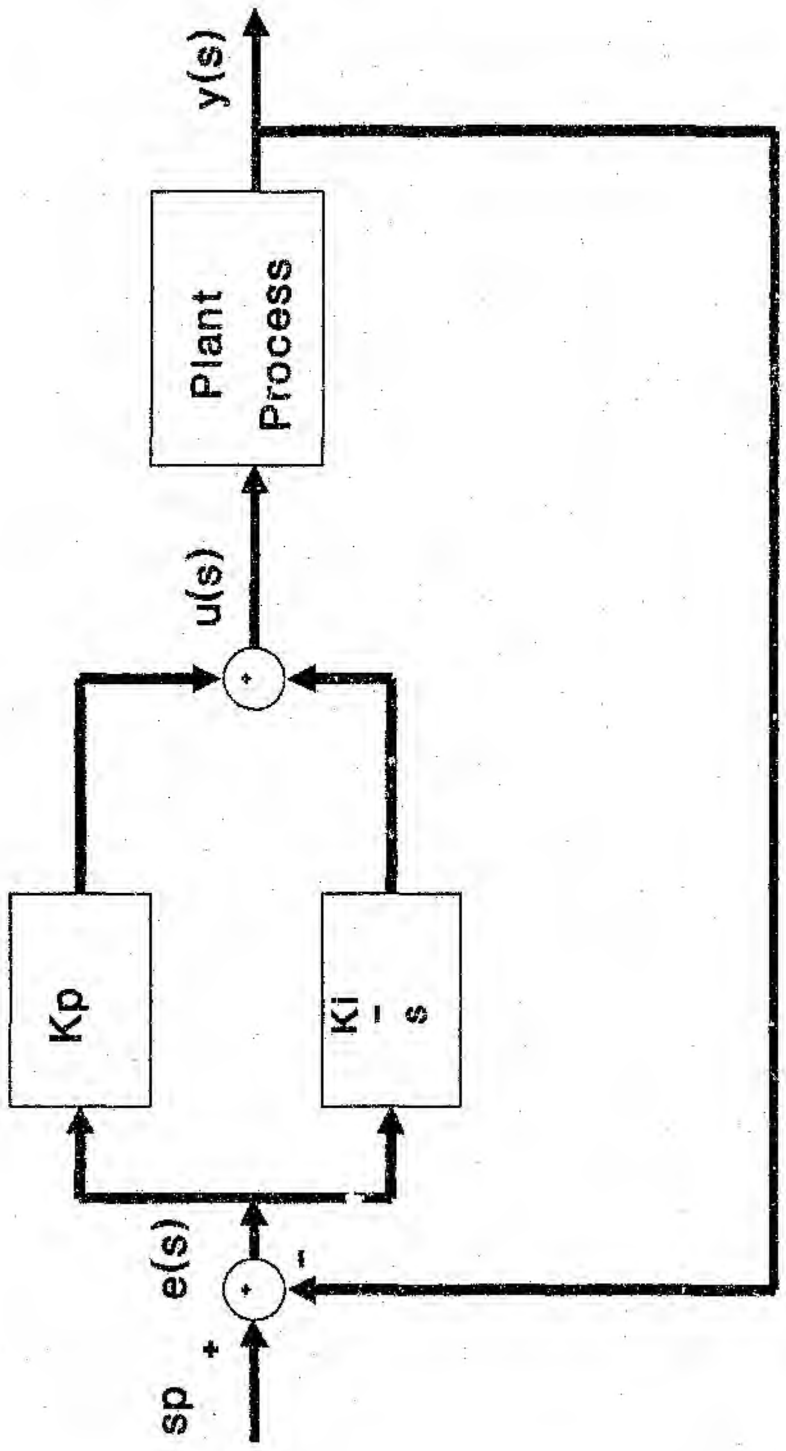


Figure 5.3: Typical PI Control Loop

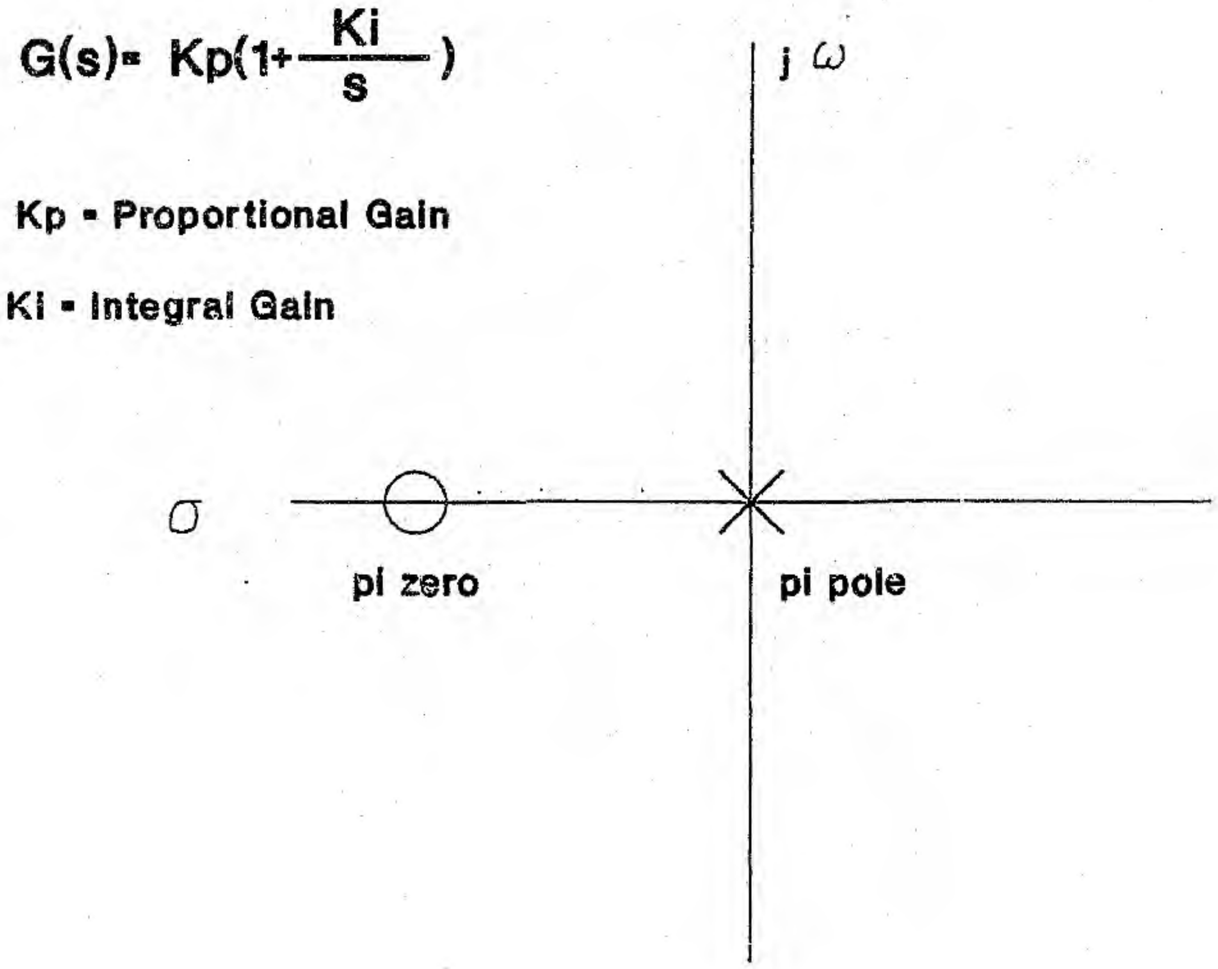


Figure 5.4: Pole-zero Locations of the PI Regulator

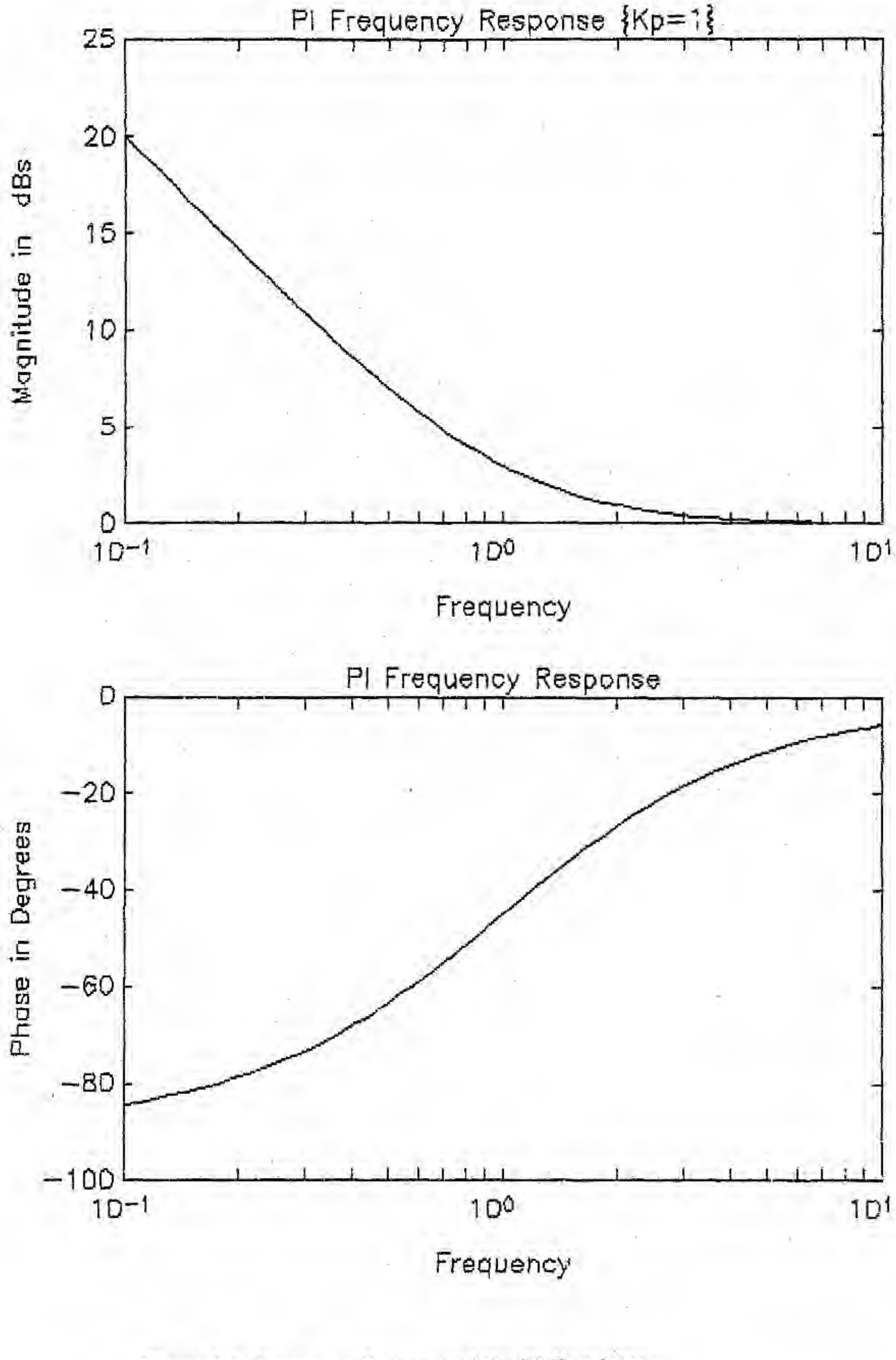


Figure 5.5: Frequency Response of the PI Regulator

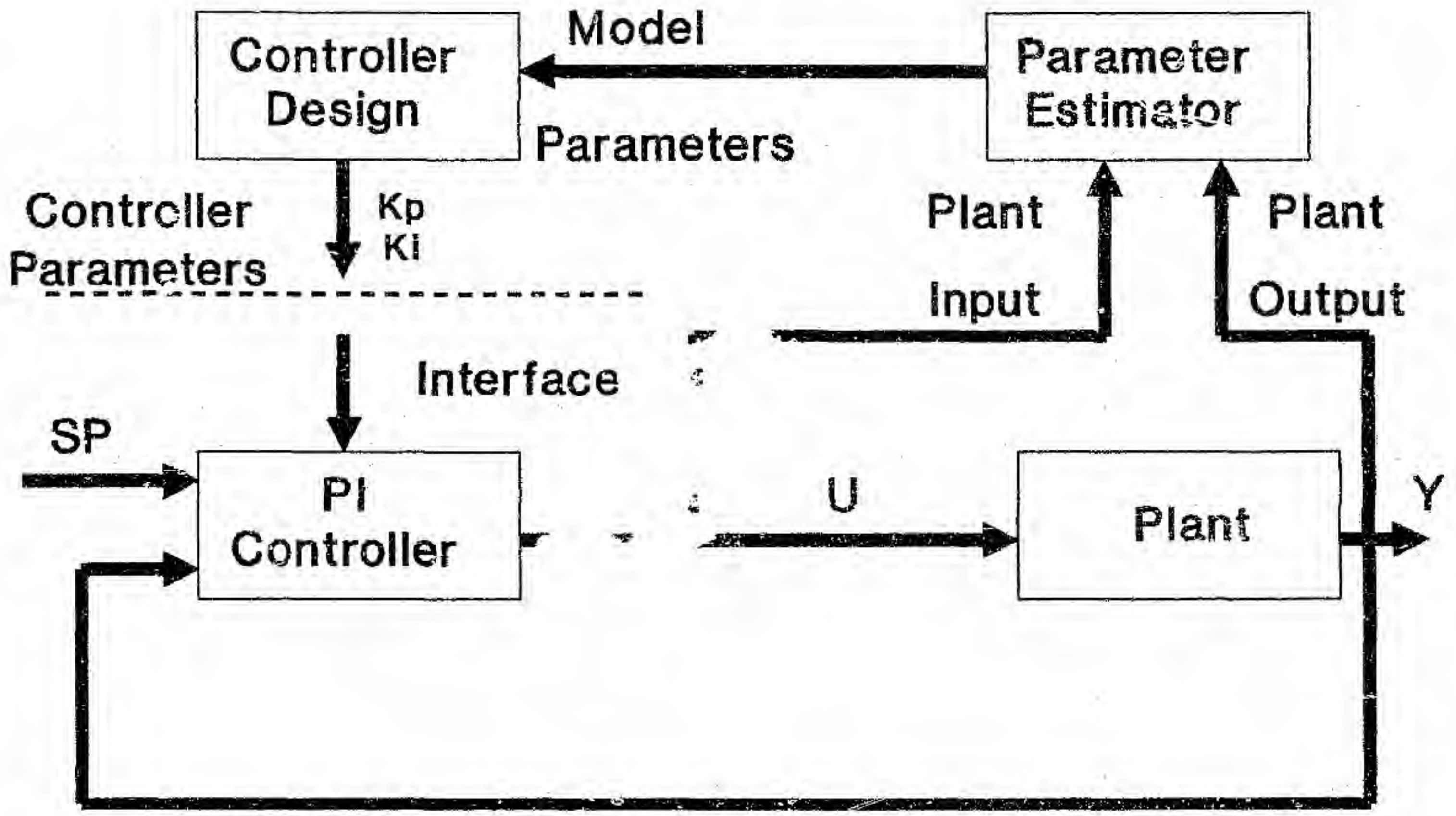


Figure 5.6: Block Diagram of the Tuning System

5.5 Summary

This chapter has presented the design philosophy that forms the basis for the controller tuning system. The algorithm was constructed using the δ-operator.

The estimator agrees with the recursive least squares method modified to handle covariance resetting and bounded noise. The controller was designed utilizing the pole placement and the internal model principle. The controller coefficients are found solving the Diophantine Equation. The inclusion of the disturbance nulling polynomial generated an integral part in the control law and solving the control law for a first-order plant model enabled the approximation of the pole placement controller to a PI controller. Also, some discussion was made on the importance of selecting a suitable closed loop polynomial A^*.

Chapter 6

Implementation Aspects

6.1 Introduction

The chapter discusses some practical issues that can influence the implementation of the estimator and controller algorithms. More specifically issues such as the choice of the sampling rate, signal conditioning, and selection of the correct startup parameters i.e. deadzone and forgetting factor, come under examination.

The performance of the estimator is heavily dependent on the implementation of appropriate filtering techniques. Filtering is necessary to enable the removal of high frequency noise, and d.c. values which can influence the ability of the estimator to converge to the real parameter values.

A very important issue in achieving optimum performance of the system is the selection of the closed-loop polynomial A^*. The A^* is established using a set of rules that convert the design specifications to a set of closed-loop pole locations.

6.2 Sample Rate Selection

The selection of the best sample rate for a digital control system depends on many factors. The absolute lower limit to the sample rate is based on the sampling theorem. The sampling theorem states that to reconstruct a band limited signal from samples of that signal, one must sample at twice the rate of the highest frequency contained in the signal. The theorem can be expressed as:

$$f_c \geq 2f_b \tag{6.1}$$

where:

f_c	is the sampling frequency
f_b	is the plant bandwidth

Quite often in practice, the Nyquist criterion given by Equation 6.1 can be insufficient in terms of system responses and system sensitivity thus, the need to sample faster arises. Franklin and Powell[25] state that faster sampling reduces delay between input change and system response and smoothes the system output response applied to a plant through a zero-order hold. Also, Goodwin and Sin[26] recommend that the sampling period be an even multiple of the time delay when the plant delay is a priori.

6.3 Signal Conditioning

This section describes the design method of the high-pass filter (hpf) and low-pass filter (lpf) which are implemented for proper conditioning of the measurement signals.

6.3.1 The Low-Pass Filter

The filter is implemented so that the noise above filter breakpoint is attenuated. A design requirement is to select the filter cutoff frequency, w_c, in such a way as to encompass the plant bandwidth. Another requirement is to provide enough signal attenuation so that the noise when aliased does not affect the estimators performance. The two requirements can be formulated as:

$$w_c \leq \frac{1}{2} w_s \tag{6.2}$$

$$\frac{w_c}{w_b} = 1 \tag{6.3}$$

where:

w_s	is the sampling frequency
w_b	is the plant bandwidth
w_c	is the lpf cutoff frequency

The discrete filter implemented in the algorithm is the realization of an analog filter with the transfer function.

$$H_{lpf}(s) = \frac{w_r^2}{s^2 + 2as + w_r^2} \tag{6.4}$$

The magnitude of the filter is given by:

$$A(w) = \frac{w_r^2}{\sqrt{(w_r^2 - w^2)^2 + 4a^2 w}} \tag{6.5}$$

The d. c. magnitude of the filter is:

$$A_{dc} = \frac{w_r^2}{w_r^2} = 1$$

which is a constraint that, if not satisfied, could as stated by Bergensen[27] render the internal estimator variables numerically sensitive to finite-word length errors.

Setting the real part a of the filter's complex roots equal to imaginary part b, the filter is restricted to a no-overshoot response.

The magnitude of the filter at the cutoff frequency is:

$$A(w_c) = 0.707\,A(0)$$

and

$$w_c = w_r$$

The analog filter has the following characteristics.

$$a = \beta = \frac{w_c}{\sqrt{2}} \tag{6.6}$$

$$w_c = w_r \tag{6.7}$$

The discrete filter is constructed using the bilinear transformation.

$$s = \frac{2}{T} \cdot \frac{1 - z^{-1}}{1 + z^{-1}} \tag{6.8}$$

thus we have

$$H(s) = \frac{w_r}{s^2 + 2as + w_r^2} \tag{6.9}$$

which results the discrete low pass filter

$$H(z) = \frac{\gamma_2 z^2 + \gamma_1 z_1 + \gamma_0}{z^2 + e_1 z + e_0} \tag{6.10}$$

where:

$$\begin{aligned} \mu v1 &= w_r^2 T^2 \\ \mu v2 &= 4aT \end{aligned}$$

and

$$\begin{aligned} e_2 &= 4\mu v2 + \mu v1 \\ e_1 &= 2\mu v1 - 8/e_2 \\ e_0 &= 4 - \mu v2 + \mu v1/e_2 \end{aligned}$$

$$\begin{aligned} \gamma_2 &= \mu v1/e_2 \\ \gamma_1 &= 2\mu v1/e_2 \\ \gamma_0 &= \mu v1/e_2 \end{aligned}$$

A block diagram of the discrete low-pass filter is illustrated in Figure 6.1. Also, illustrated in Figure 6.2 and Figure 6.3 are the frequency responses of both the analog and digital filters. The repetition of the digital filter response is due the fact that the filter magnitude and phase are periodic functions of w with period f_s, where f_s is the Nyquist frequency. To determine the resulting response, it suffices therefore to consider the behavior of the digital filter in the fundamental interval $(-f_s, f_s)$ only.

The cutoff frequency of the filter is defined by the designer during the actual application of the algorithm. The system, also, makes the necessary checks ensuring that the requirements dictated by Equations 6.2 and 6.3 are adhered to.

6.3.2 The High-Pass Filter

In Chapter Five, during the development of the robust estimator the high-pass filter (hpf)

$$G_{hpf}(\delta) = \frac{\delta}{\delta + \varepsilon} \tag{6.11}$$

was generated. The filter is used by the algorithm to eliminate the d.c. values during the parameter identification state.

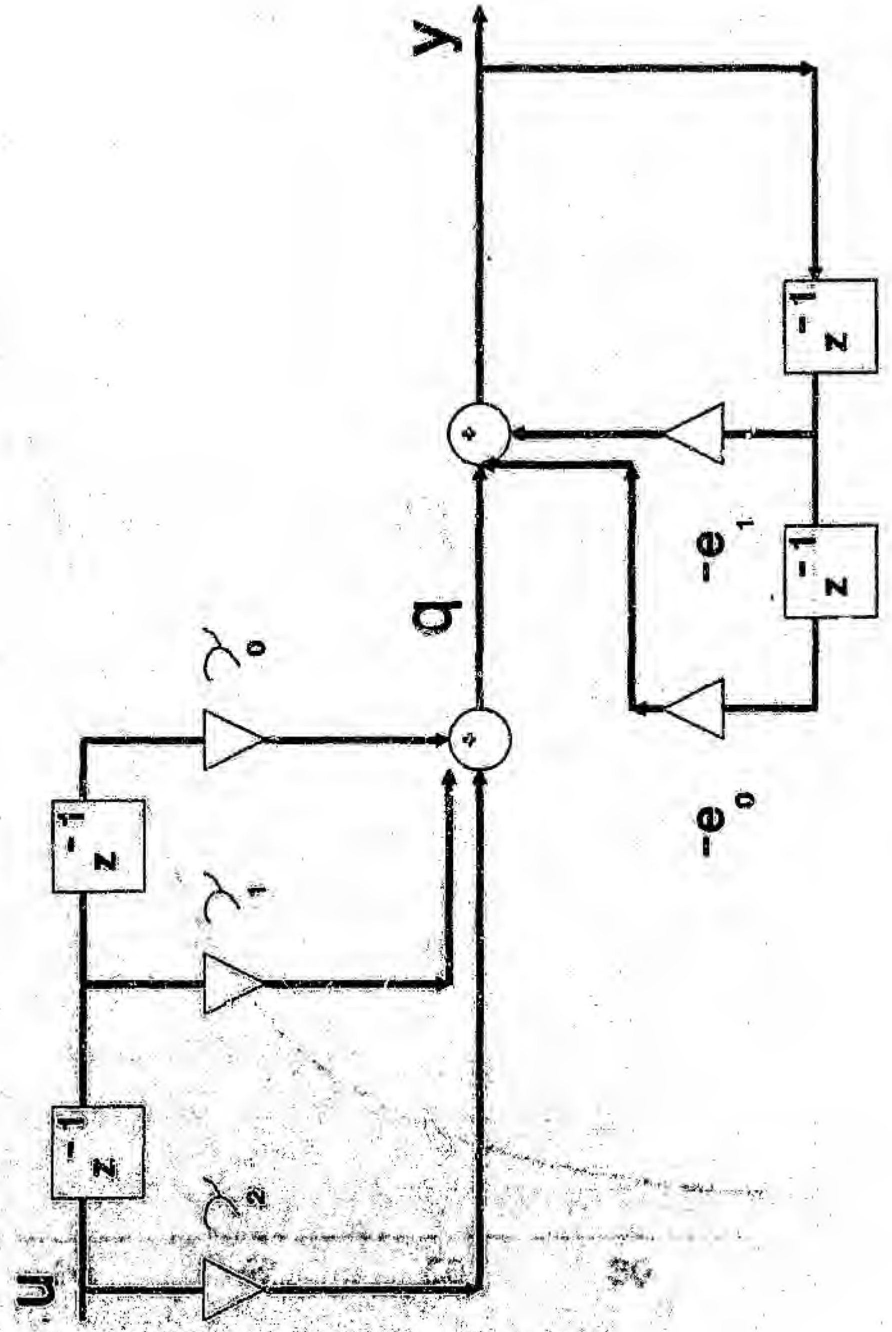


Figure 6.1: Discrete Low-Pass Filter

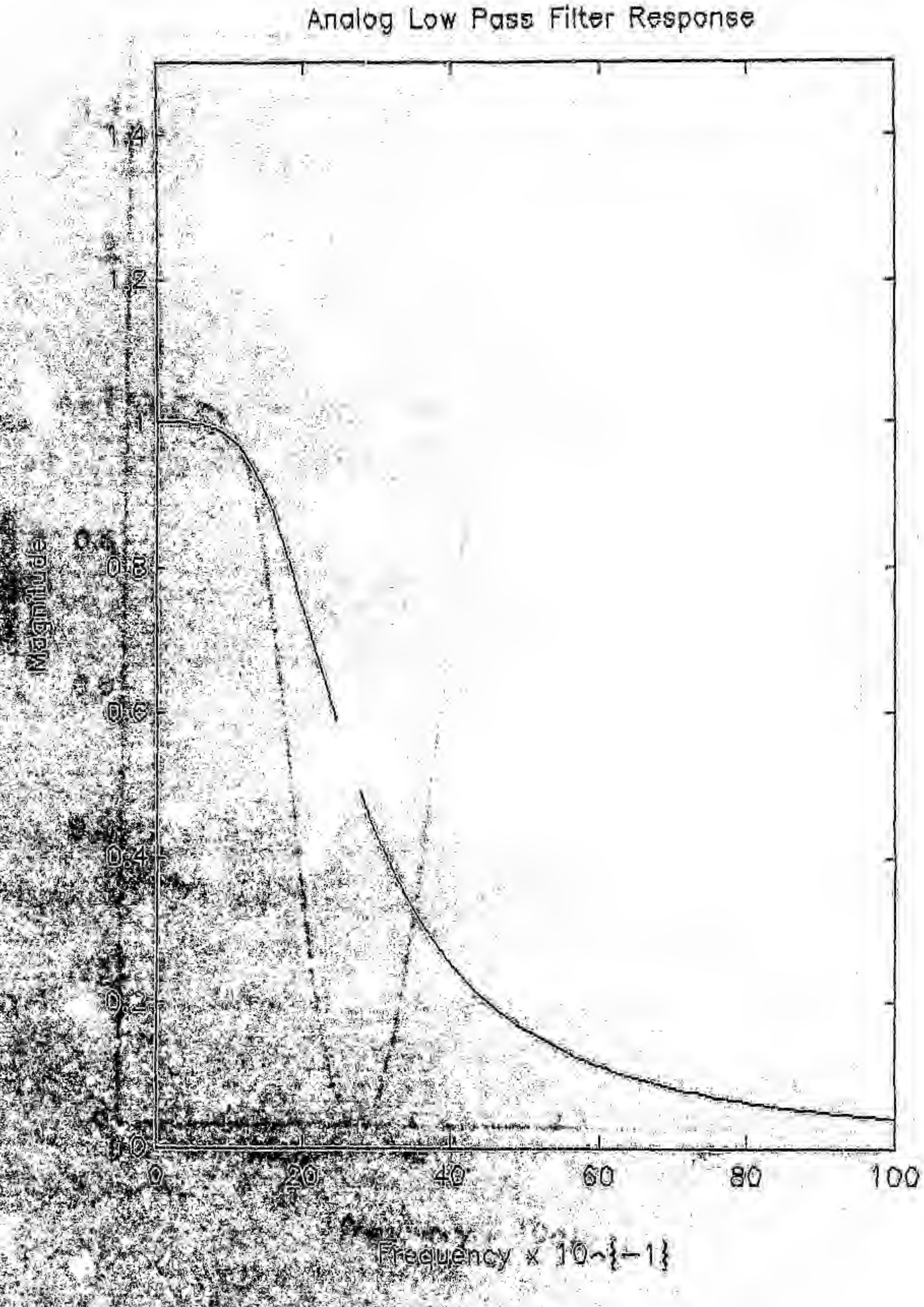


Figure 6.2: Frequency Response of the Analog LPF

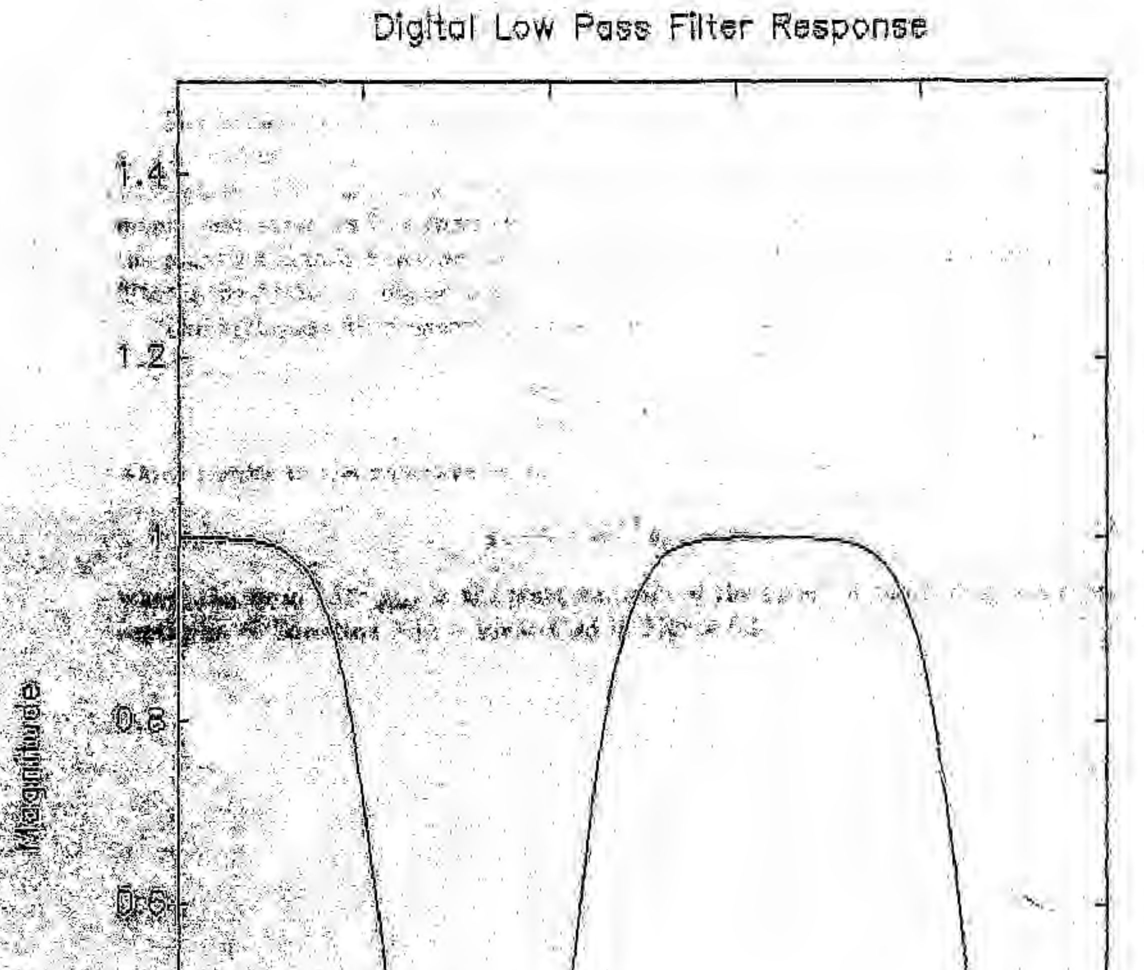


Figure 6.3: Frequency Response of the Digital LPF

The frequency response of the filter depends on the value of the corner frequency ε. The choice of ε is not very critical although the following constraints should be adhered to. The value of ε must be greater than the very slow frequency measurements and must be less than the frequency range of the plant so as not to alter the plant's dynamic behavior. A general indication of the frequency response of the filter is illustrated in Figure 6.4.

The high-pass filter equation can also be written as:

$$\frac{y_{out}}{y_{in}} = \frac{1}{1+\varepsilon\delta^{-1}} \quad . \tag{6.12}$$

which results to the recursive form:

$$y_{out} = -\varepsilon\delta^{-1}\, y_{out} + y_{in} \tag{6.13}$$

where the term $\varepsilon\delta^{-1}\, y_{out}$ is the state variable of the filter. A block diagram representation of Equation 6.12 is illustrated in Figure 6.5

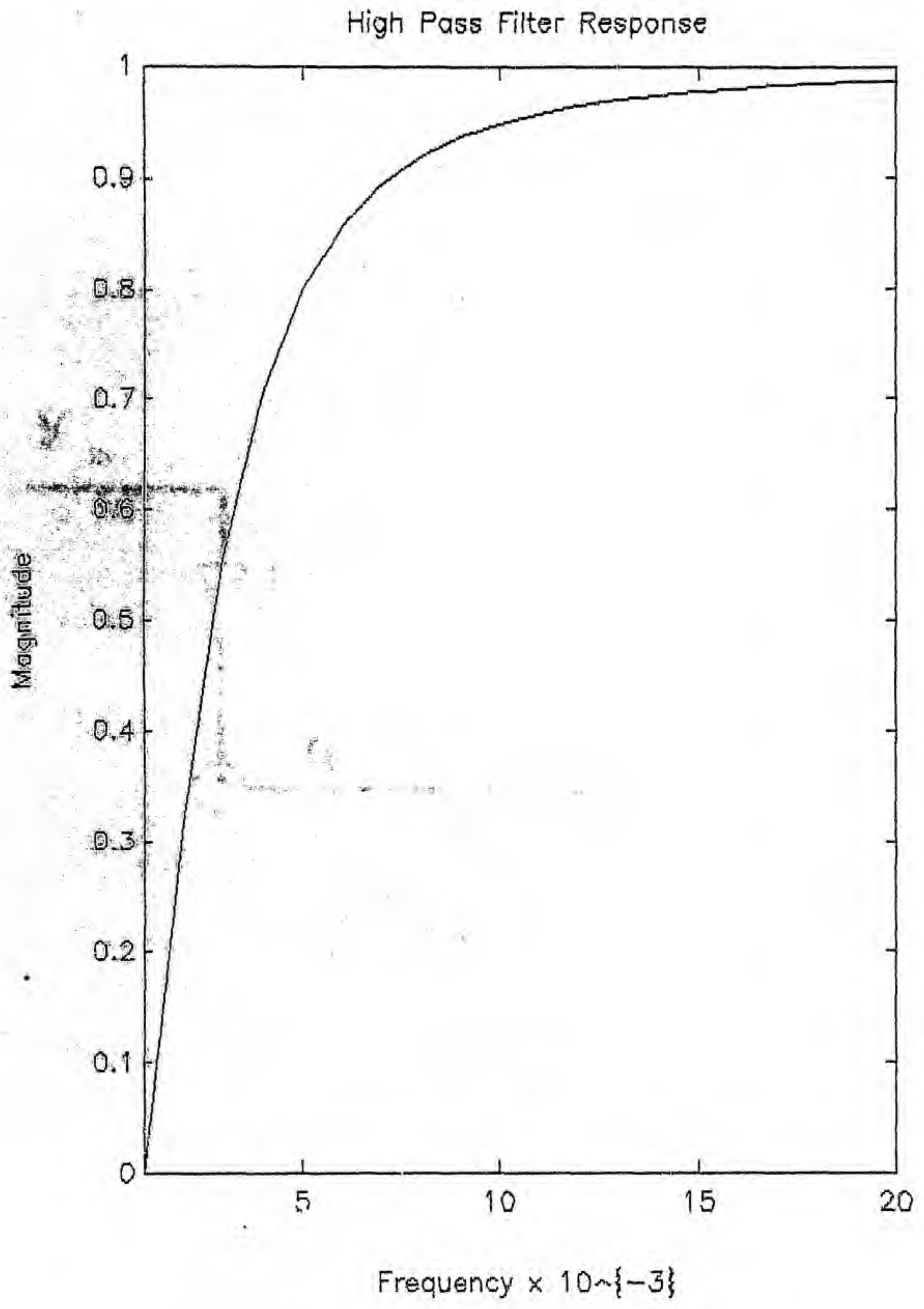


Figure 6.4: Frequency Response of the HPF

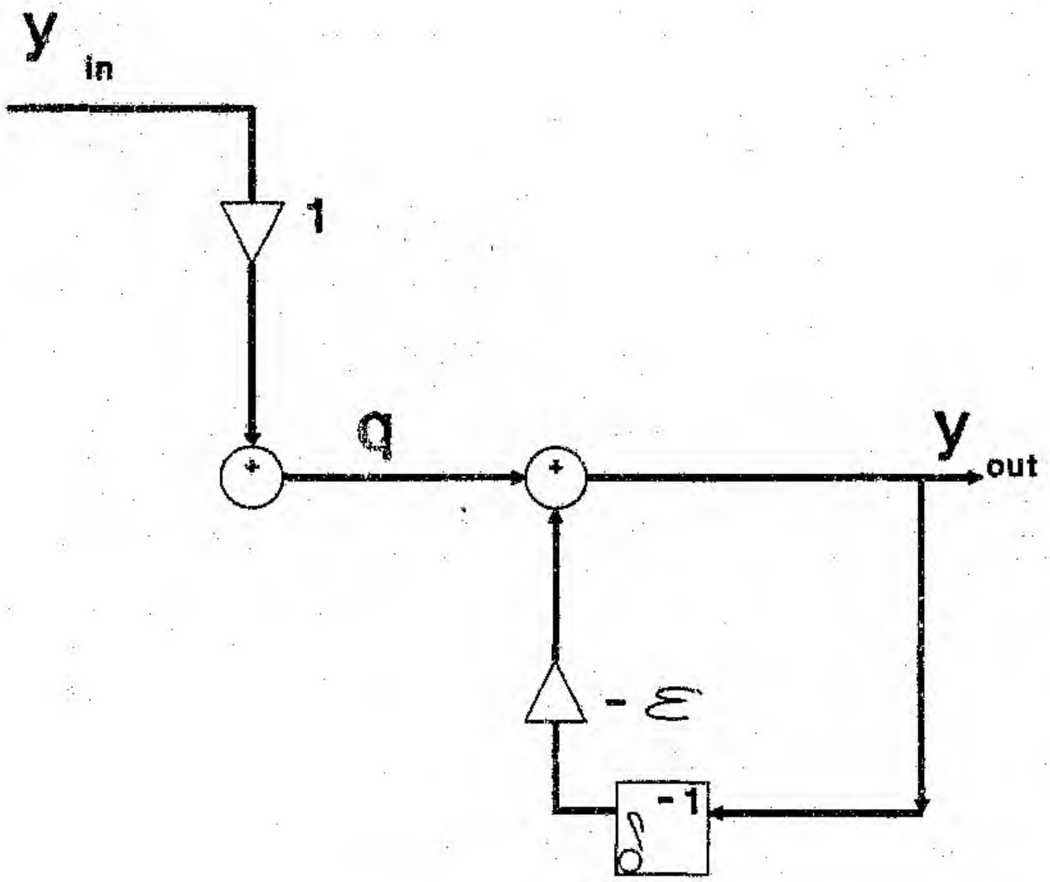


Figure 6.5: Block Diagram of High-Pass Filter

6.4 Exponential Discounting and Deadzone

6.4.1 Exponential Discounting

A simple way of ensuring good parameter variation tracking is to use exponential discounting to discard old data. Astrom et.al[28] have presented good practical results with application of the forgetting factor λ.

The choice of the forgetting factor λ for the parameter estimation depends on the speed with which the parameters vary, the model order, and the type of disturbances that are present. For constant processes or very slow varying processes $\lambda = 0.99$ is recommended. For slow varying processes and stochastic disturbances, $0.85 \leq \lambda \leq 0.90$ is an appropriate choice. The λ factor influences the memory length N of the estimator and thus the number of samples that are used by it. The relationship between N and λ is given by:

$$N = \frac{1}{1-\lambda} \tag{6.14}$$

For instance, when:

$$\begin{aligned} \lambda &= 0.99 \quad N = 100.0 \\ \lambda &= 0.85 \quad N = 6.666 \\ \lambda &= 0.90 \quad N = 10.00 \end{aligned}$$

Also, effective is the application of the variable forgetting factor

$$\lambda(k) = \lambda_0 \lambda(k-1) + (1-\lambda_0) \tag{6.15}$$

which imposes exponential data weighting for a transient period. Typical starting values for the algorithm are:

$$\begin{aligned} \lambda(0) &= 0.95 \\ \lambda_0 &= 0.99 \end{aligned}$$

Included in the tuning algorithm, is the option of using the fixed or the variable λ factor.

6.4.2 Deadzone

The lack of excitation for a long time can affect the gain $P(k)$ of the parameter update equation and cause an output burst. The estimator performance can be enhanced and avoid an estimator wind-up with the use of the deadzone method. The aim of the deadzone is to allow the update of the parameter vector only when the data from the plant contains useful information. It achieves this by "switching off" the parameter update equation when the estimation error $\varepsilon(k)$ is smaller than some specific threshold. Egardt[29] and Hägglund[30] have designed very good techniques based on this idea. The two principal deadzone methods used to turn off the estimator are presented next.

Constant Threshold Deadzone

In this case, the deadzone threshold is a priori factor. The deadzone $a(k)$ depending on the comparison between the estimation error and the noise size takes the fixed values:

$$a(k) = \begin{cases} 0, & \text{if } |e(k)| \leq 2\Delta; \\ 1, & \text{if } |e(k)| > 2\Delta \end{cases}$$

where $e(k)$ is the error between the actual plant output $y(k)$ and the estimated output $\hat{y}(k)$. The deadzone simply turns off the parameter vector update when the error $e(k) \leq 2\Delta$ and $\eta(k)$ is a bounded noise sequence such that $\Delta \geq |\eta(k)|$.

Variable Threshold Deadzone

This method implements the idea of using varying threshold, adjusting to the magnitude changes of the plant measurements. It is based on the reasoning that the estimation error, due to the bounded noise, is directly related to the magnitude of the plant data used by the estimator.

The deadzone $a(k)$ is defined as:

$$a(k) = \begin{cases} 0, & \text{if } |e(k)| < \beta m(k); \\ a \cdot f(e(k), \beta m(k))/e(k) & \text{otherwise.} \end{cases}$$

The function $m(k)$ is given by

$$m(k) = \sigma_0 m(k-1) + (1-\sigma_0)\varepsilon_0 + \varepsilon_1 |u(k-1)| + \varepsilon_2 |y(k-1)| \qquad (6.16)$$

The constants are $\varepsilon_0, \varepsilon_1, \varepsilon_2 \geq 0$ and $\sigma_0 \epsilon (0,1)$.

A constraint is that $m(k)$ must be greater or equal to the bounded noise sequence $\eta(k)$ at all time thus,

$$m(k) \geq \eta(k) \quad k \geq 0$$

with $m(0) = m_0; m_0 \geq 0$.

The function β is defined as

$$\beta = \sqrt{\varepsilon_4 - \frac{1}{1-a}}$$

with $\varepsilon_4 \geq 0$ and $a \epsilon (0,1)$.

The tuning system has been designed to allow the user the option of utilizing either the fixed or the variable deadzone. Establishing the values for the variable deadzone constants is a difficult process. Often the conditions the estimator operates under do not warrant the complexity associated with the relative deadzone thus the fixed deadzone is used.

6.5 Selection of the Closed-Loop Polynomial

The behavior of the process must agree with some desired closed-loop performance specifications. It is necessary to appropriately choose the polynomial A^* so that the design specification can be achieved and translated into a set of closed-loop pole locations.

When the process is represented by a first-order model, the closed-loop polynomial is given by:

$$A^*(\delta) = \delta^2 + a_1^*\delta + a_0^* \tag{6.17}$$

Approximating $\delta = s$ enables $A^*(\delta)$ to be identified with the second-order continuous-time system

$$A^*(\delta) = \delta^2 + 2\zeta w_\eta \delta + w_\eta^* \tag{6.18}$$

with ζ and w_η the damping ratio and natural frequency respectively. Correlation of the coefficients of the two equations results to:

$$\begin{aligned} a_1^* &= 2\zeta w_\eta \\ a_0^* &= w_\eta^2 \end{aligned}$$

thus relating the A^* polynomial coefficients to the transient performance specifications for the system.

The A^* coefficients are generated using a set of rules[31] which are built into the algorithm. The rules address both the time and frequency domain areas. These are as follows.

a. Time Domain

$$P.O. = 100e^{-\zeta\pi/\sqrt{1-\zeta^2}} \tag{6.19}$$

$$T_s = \frac{N_\tau}{\zeta w_n} \tag{6.20}$$

where P.O. is the percent overshoot of the system, T_s is the required settling time, and N_τ is the number of time constants τ within which the system is expected to settle.

The user defines the P.O. and T_s and the algorithm identifies the corresponding damping ratio, ζ, and natural frequency, w_η. A normal range for ζ is $0.450 \le \zeta \le 0.707$.

b. Frequency domain

$$\zeta = 0.01\phi_{pm} \tag{6.21}$$

$$w_r = w_n\sqrt{1-2\zeta^2} \tag{6.22}$$

$$M_{pw} = (2\zeta\sqrt{1-\zeta^2})^{-1} \tag{6.23}$$

$$w_{3db} = w_n(\sqrt{\zeta^2+1}+\zeta) \tag{6.24}$$

where ϕ_{pm} is the phase margin, w_r is the resonant frequency, M_{pw} the peak magnitude, and w_{3db} the $3db$ frequency. The above equations hold when $\zeta \le 0.707$. When $\zeta = 0$ then $w_r = w_n$. With the above rules the desired system is examined for relative stability in the frequency domain. The phase margin is established from the damping ratio. If the resulting ϕ_{pm} does not meet the given specifications the user can define the required ϕ_{pm}. The algorithm produces the closed-loop system parameters by examining both the frequency and time domains and enabling the user to make the appropriate modifications if necessary. Also, the algorithm ensures that the bandwidth of the compensated system does not violate the sampling theorem constraints by ensuring that the system bandwidth does not overbound the low-pass filter cutoff frequency w_c. The rules are described in the following pseudo-code.

```
BEGIN
DEFINE THE PERCENT OVERSHOOT
DEFINE THE SETTLING TIME
CALCULATE THE DAMPING RATIO
CALCULATE THE PHASE MARGIN
IF PHASE MARGIN NOT WITHIN LIMITS THEN
    DEFINE THE PHASE MARGIN
    CALCULATE THE PERCENT OVERSHOOT
    IF PERCENT OVERSHOOT NOT WITHIN LIMITS THEN
        GO TO BEGIN
    ENDIF
ENDIF
CALCULATE THE RESONANT FREQUENCY
CALCULATE THE PEAK MAGNITUDE
CALCULATE THE 3DB FREQUENCY
IF 3DB FREQUENCY IS .GT. THAN THE LPF WC FREQUENCY THEN
    GO TO BEGIN
ENDIF
```

6.6 Summary

This chapter presented some implementation issues which are essential for ensuring the robustness and good performance of the algorithm. It formulated a methodology for choosing the correct sampling rate, for implementing the correct signal conditioning methods, and for choosing the appropriate initial parameter values such as the deadzone and forgetting factor.

The close-loop characteristic polynomial A^* is constructed using continuous-time design specifications such as the damping ratio, settling time, natural frequency, and phase margin. These variables get used by set of rules that convert them to a set of close-loop poles that form the A^*.

Chapter 7

Experimental Results

7.1 Introduction

The design work is substantiated with the use of simulation and real plant tests. This chapter describes the tests that were performed to evaluate the algorithm with regard to:

a. Performance of the parameter estimator in terms of convergence, parameter tracking error minimization, and data discounting.

b. Controller robustness, adherence to design specifications, closed-loop performance to reference signal changes, and system stability.

Some comparisons are made with results obtained from using the Matlab ARX estimation routine. Finally, the performance of the variable deadzone is examined against that of the fixed deadzone and some results are presented.

7.2 Simulation

The continuous-time system used was a first-order system described by

$$\frac{dy(t)}{dt} + y(t) = u(t)$$

The transfer function of the system is:

$$\frac{Y(s)}{U(s)} = G(s) = \frac{1.0}{s + 1.0}$$

with

$$a_0 = 1.0$$
$$b_0 = 1.0$$

The algorithm was started with the following fixed parameters.

$w_c = 3.0$ LPF cutoff frequency

$\Delta = 0.1$ Sampling time

$\lambda = 0.97$ Exponential forgetting factor

$P(0) = 100.0$ Initial value for covariance matrix

$\theta(0) = 0.0$ Initial value for parameter vector

$\varepsilon = 0.0008$ HPF corner frequency

The open-loop system was excited by a rich input signal to ensure good convergence performance. The algorithm succeeded in producing a very good estimated output $\hat{y}(k)$ in comparison with the actual plant output $y(k)$ as it is shown in Figure 7.1. The estimation error $e(k)$ and deadzone $a(k)$ are shown in Figure 7.2. The error is quite large at the initial stages of the estimation process. The fixed deadzone $a(k)$ takes the value of 1.0 when the estimation error is greater than the estimation error limit $e_{\lim}$. The deadzone behavior follows the error $e(k)$ very closely. The parameter updating occurred mostly during the start-up period. This can, also, be seen in Figure 7.3. The parameters converge rapidly during the initial stages with only sprradic fluctuations during the later stages.

The algorithm produced the estimated parameters.

$$a_0 = 1.009796$$

$$b_0 = 1.012243$$

which represent a δ-operator model:

$$\delta y(k) + 1.009796\, y(k) = 1.012243\, u(k)$$

The estimated model approximates the continuous time first order plant very closely. The estimated parameters are very near to the true system parameters. This is an indication that the δ-operator concept under fast sampling conditions produces good models of continuous time systems. It also indicates that the $\delta = s$ approximation is valid under fast sampling conditions.

It is understood that the very close tracking of the plant output by the estimator and the very good convergence of the estimated parameters to the real ones is influenced by the fact that the plant order is the same as the plant order the algorithm is designed to cater for, namely, to produce first-order estimation models.

The open-loop system has a damping factor of $\zeta = 1$ and $w_n = 1$. The specifications for the closed loop system were a) percent overshoot $PO = 10\%$, and b) settling time $Ts = 4$ seconds. This resulted in

$$\zeta = 0.5912$$

$$w_n = 1.6916$$

$$w_{3dB} = 2.9651$$

$$\phi_{pm} = 59^o$$

The closed loop polynomial A^* was given by:

$$A^* = \delta^2 + 2.000\,\delta + 2.862$$

having the roots

$$s_1 = -1 + 1.36455\,j$$

$$s_2 = -1 - 1.36455\,j$$

The roots of $A^*(\delta)$ are depicted on a s-plane diagram shown in Figure 7.4.

The controller design parameters generated by the algorithm were:

$$\begin{aligned} K_p &= 1.0000 \\ K_i &= 2.8615 \quad \text{or} \\ T_i &= 0.3495 \end{aligned}$$

The closed loop response to a step input can be seen in Figure 7.7. The settling time of the system is less than four seconds and the overshoot is ten percent. It is clear that the compensated system meets the design specifications. Also, Figure 7.5 shows the response of the compensated system when the reference signal is a series of steps. The system is stable and the output follows the input signal smoothly. The algorithm generated PI parameter which produced very good tracking of the setpoint.

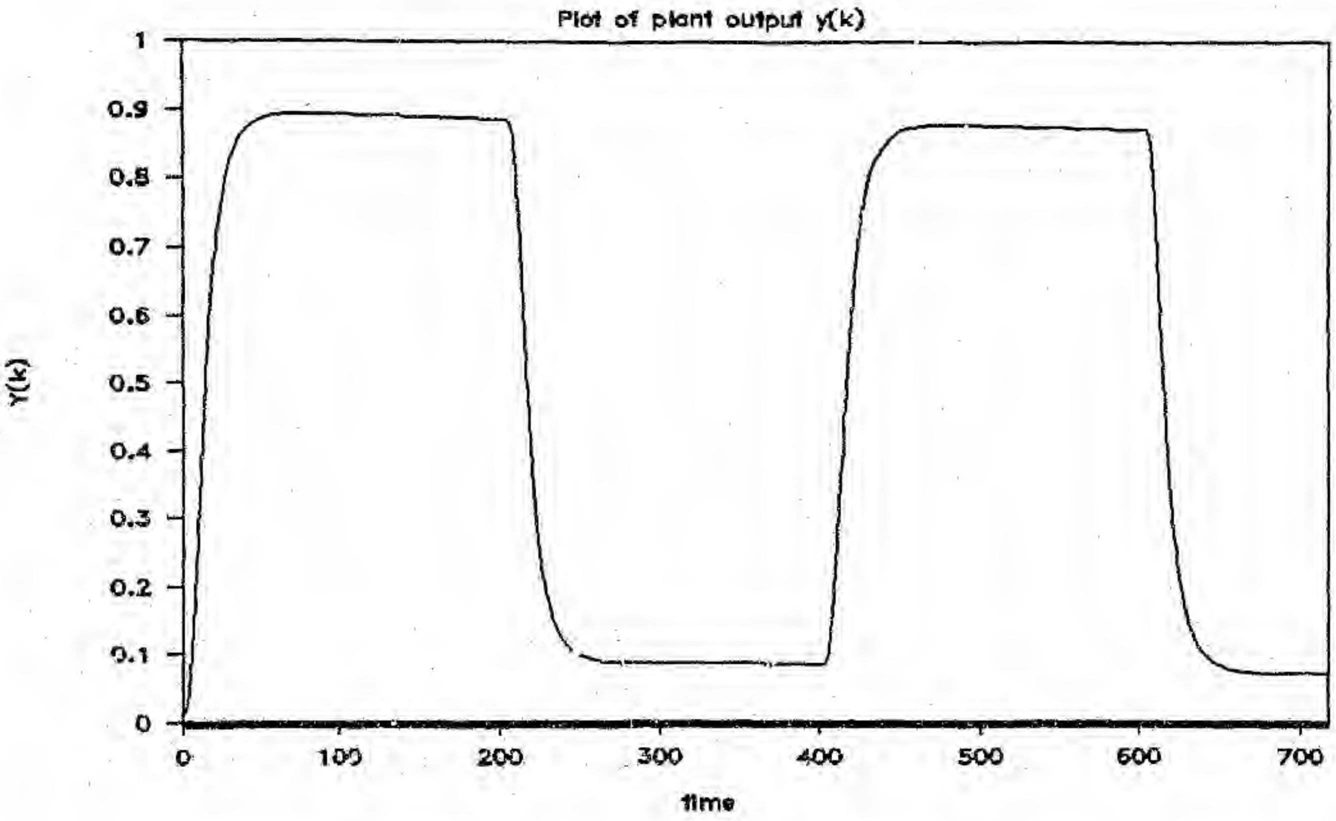
Next, the percent overshoot requirement was reduced drastically, but the settling time requirement was kept the same. The design specifications were $PO = 0.3\%$ and $T_s = 4$ seconds. This allowed the system almost no overshoot and still required a four second limit for reaching steady state. The controller parameters produced were:

$$\begin{aligned} K_p &= 2.000 \\ K_i &= 1.4308 \quad \text{or} \\ T_i &= 0.6989 \end{aligned}$$

Figure 7.6 illustrates the response of the compensated system to a series of steps. The system again tracked the setpoint in a satisfactory manner.

7.2.1 Summary

The algorithm was used to model a first order continuous-time system. The estimator produced parameters that were very close to the true plant parameters, reinforcing the idea that under fast sampling conditions, the delta-model can approximate the continuous-model well. The controller design produced PI parameters that produce stable closed-loop system responses and good setpoint tracking.



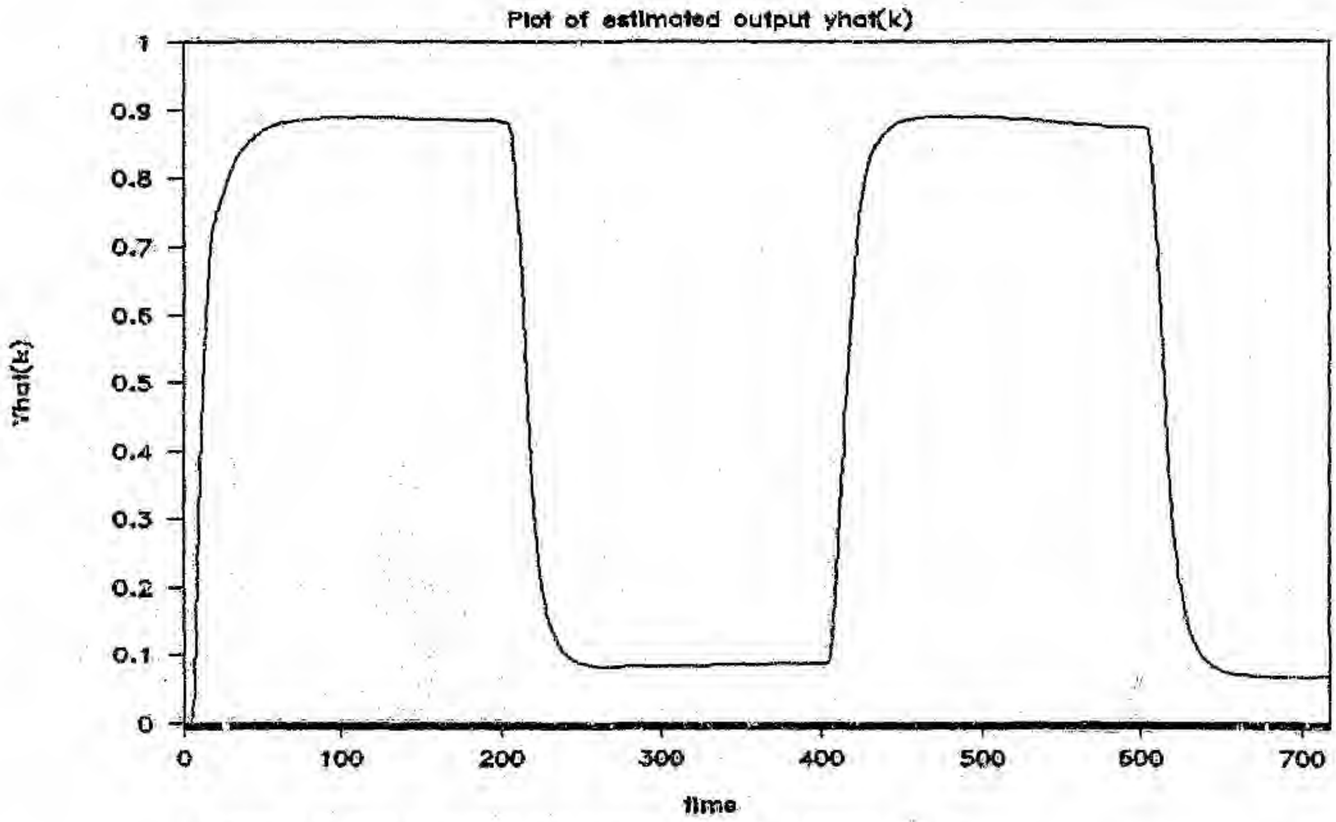
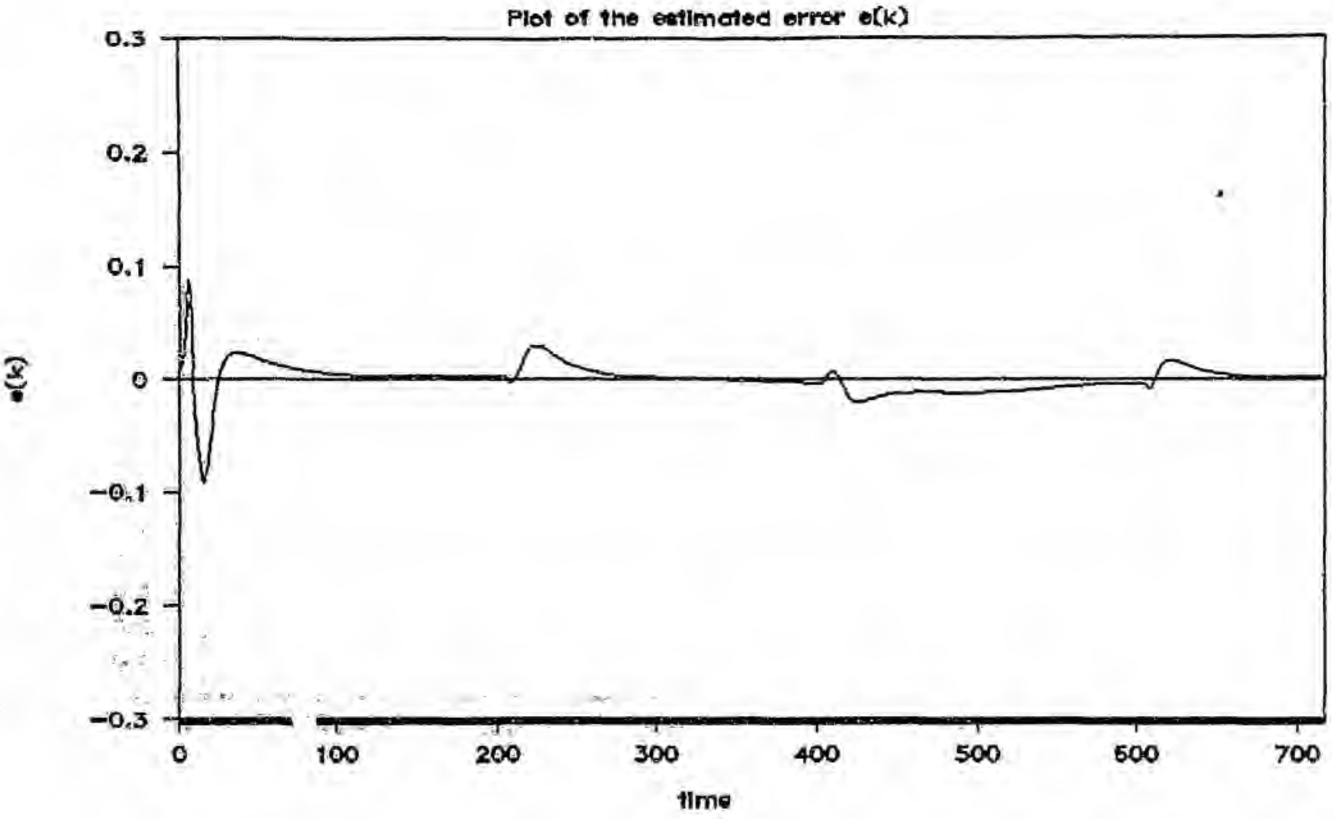


Figure 7.1: Plot of Actual and Estimated Plant Output Values



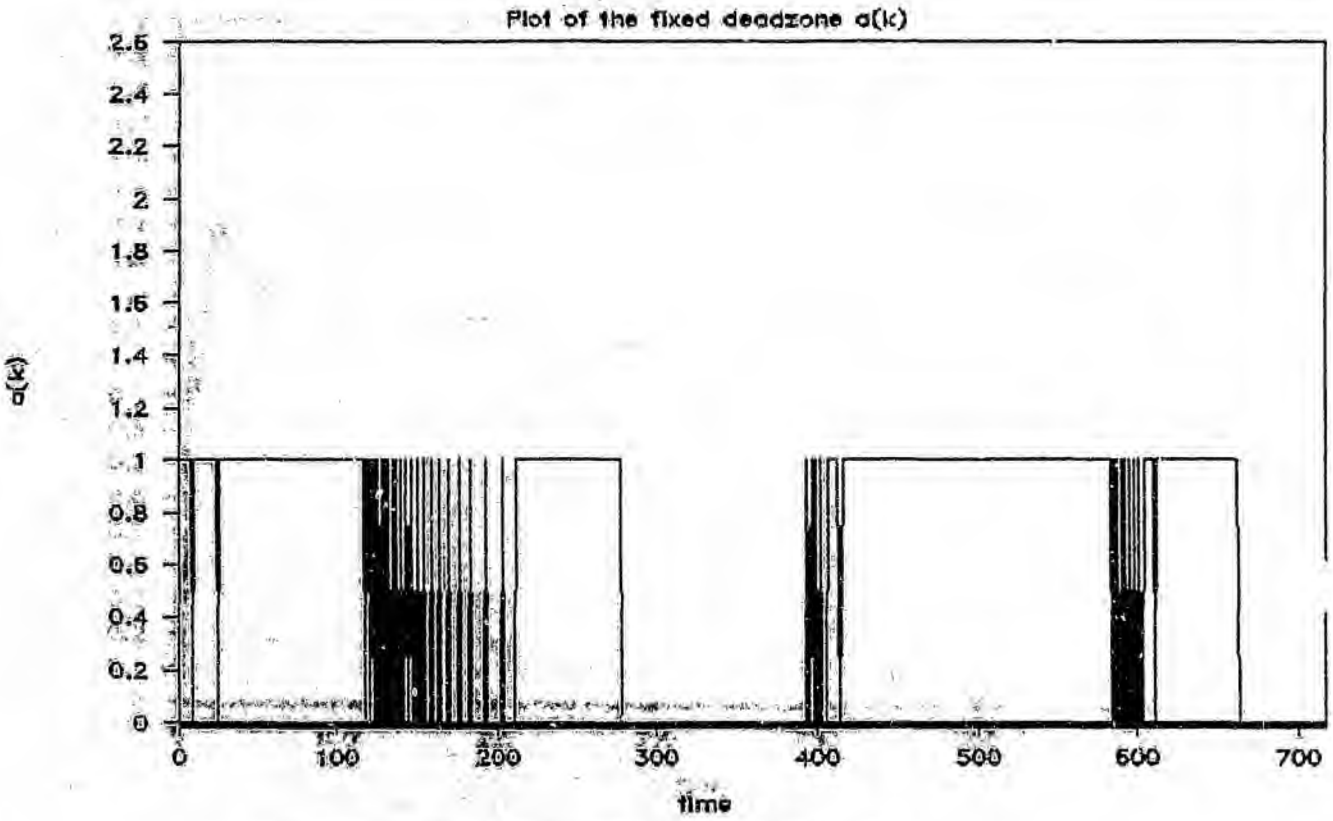
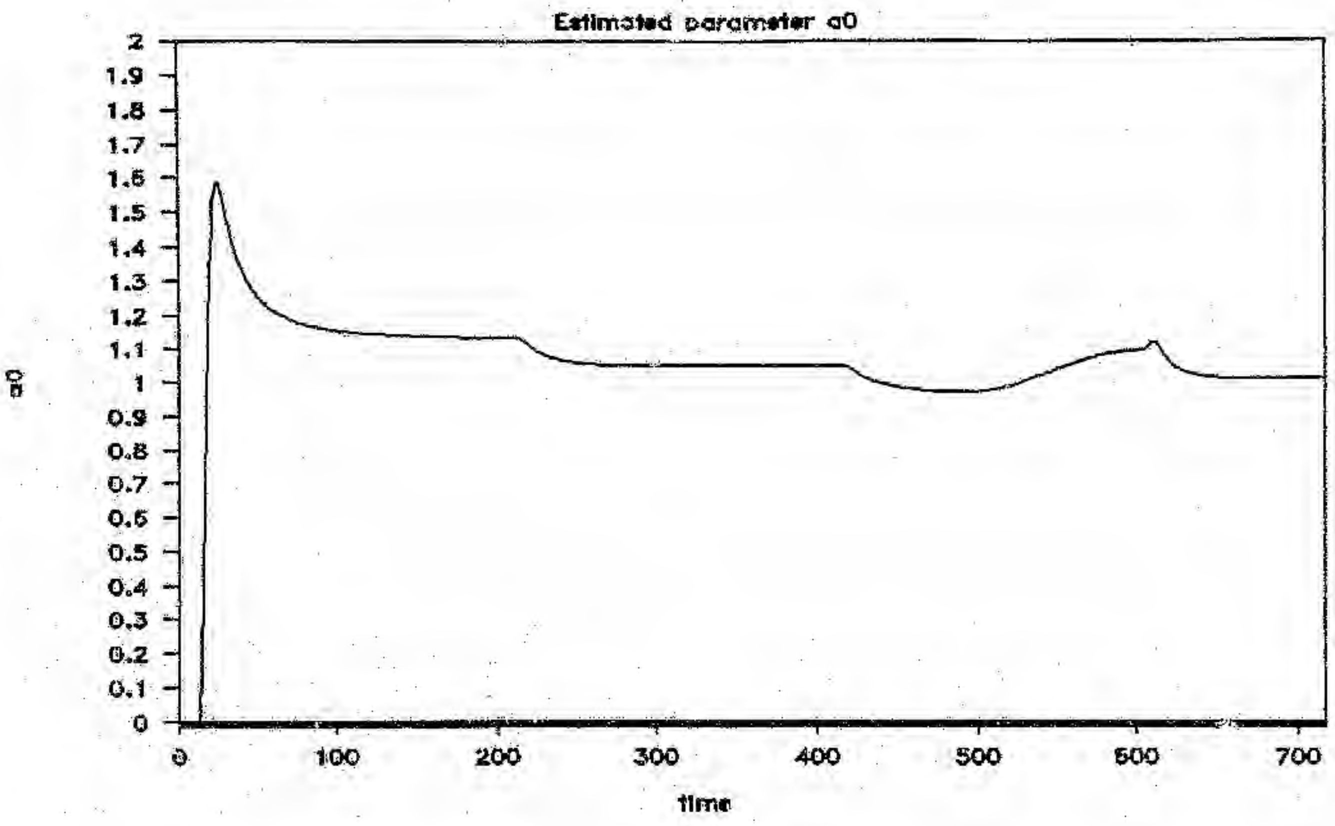


Figure 7.2: Plot of Estimated Error and Deadzone



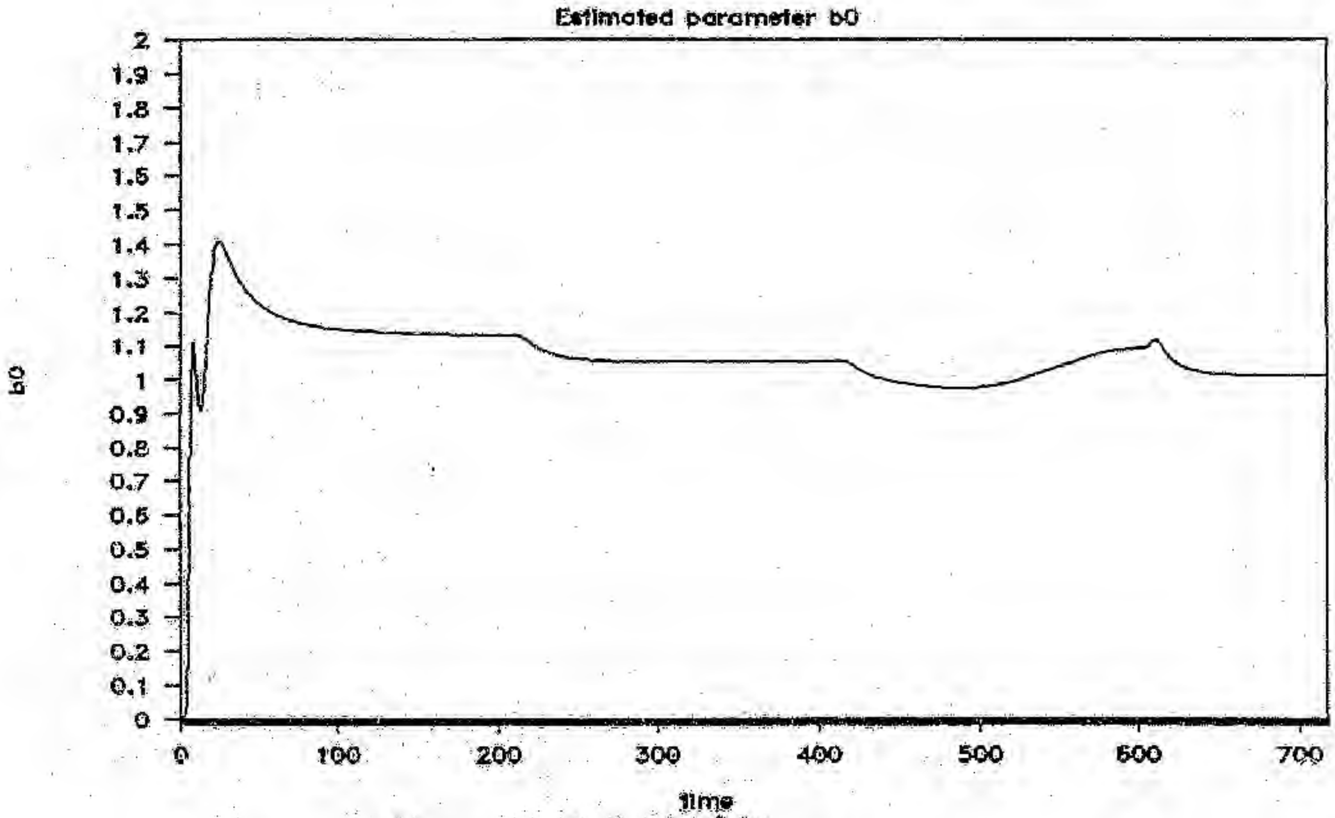


Figure 7.3: Plot of Estimated Parameters a0 and b0

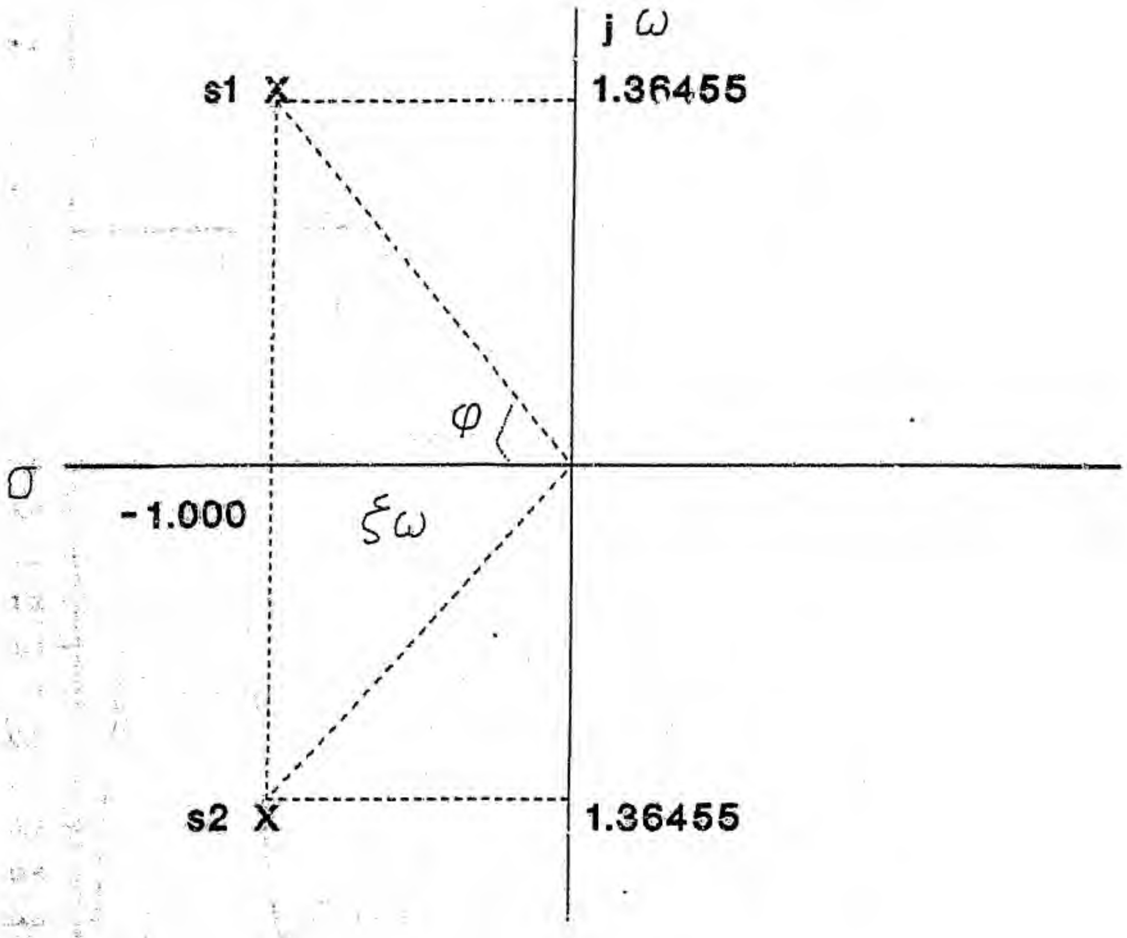
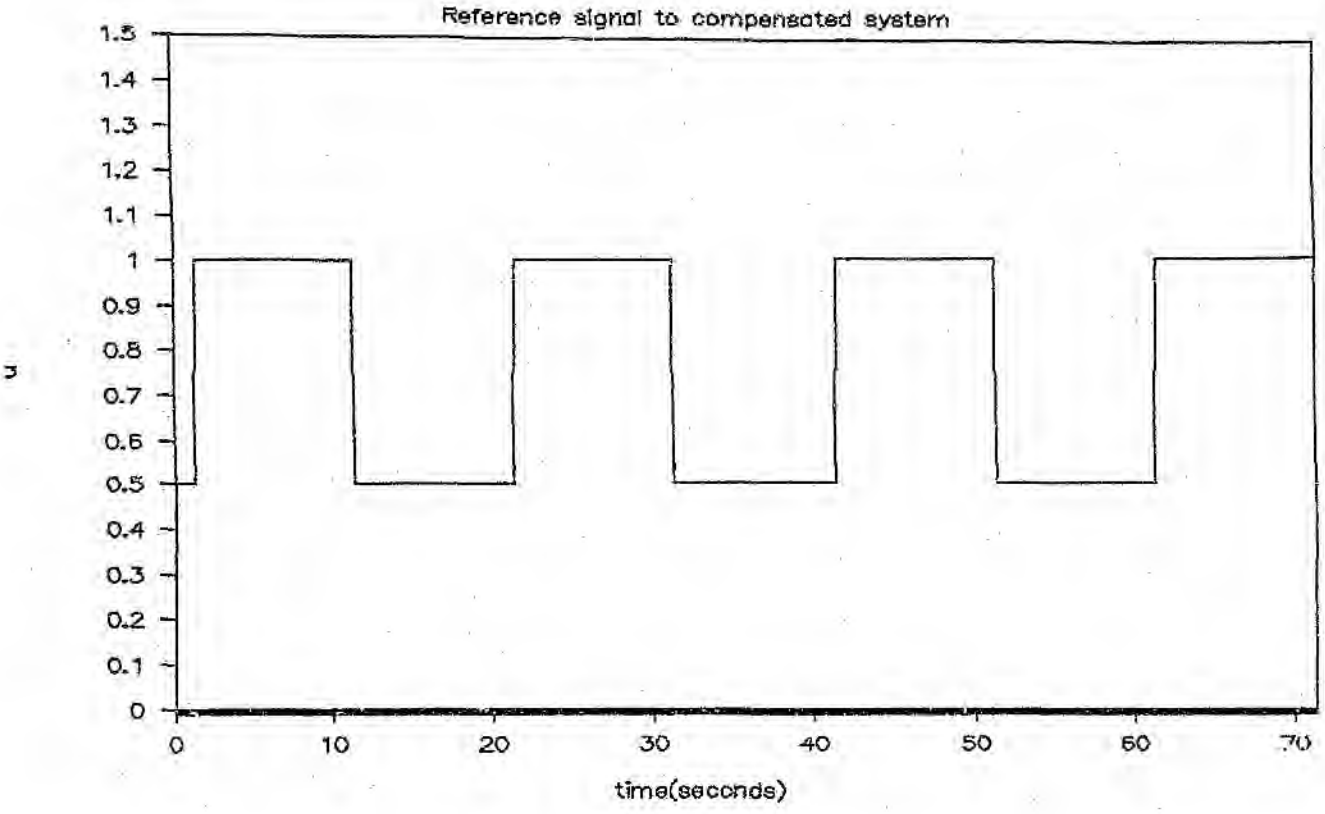


Figure 7.4: Root Locus of the System



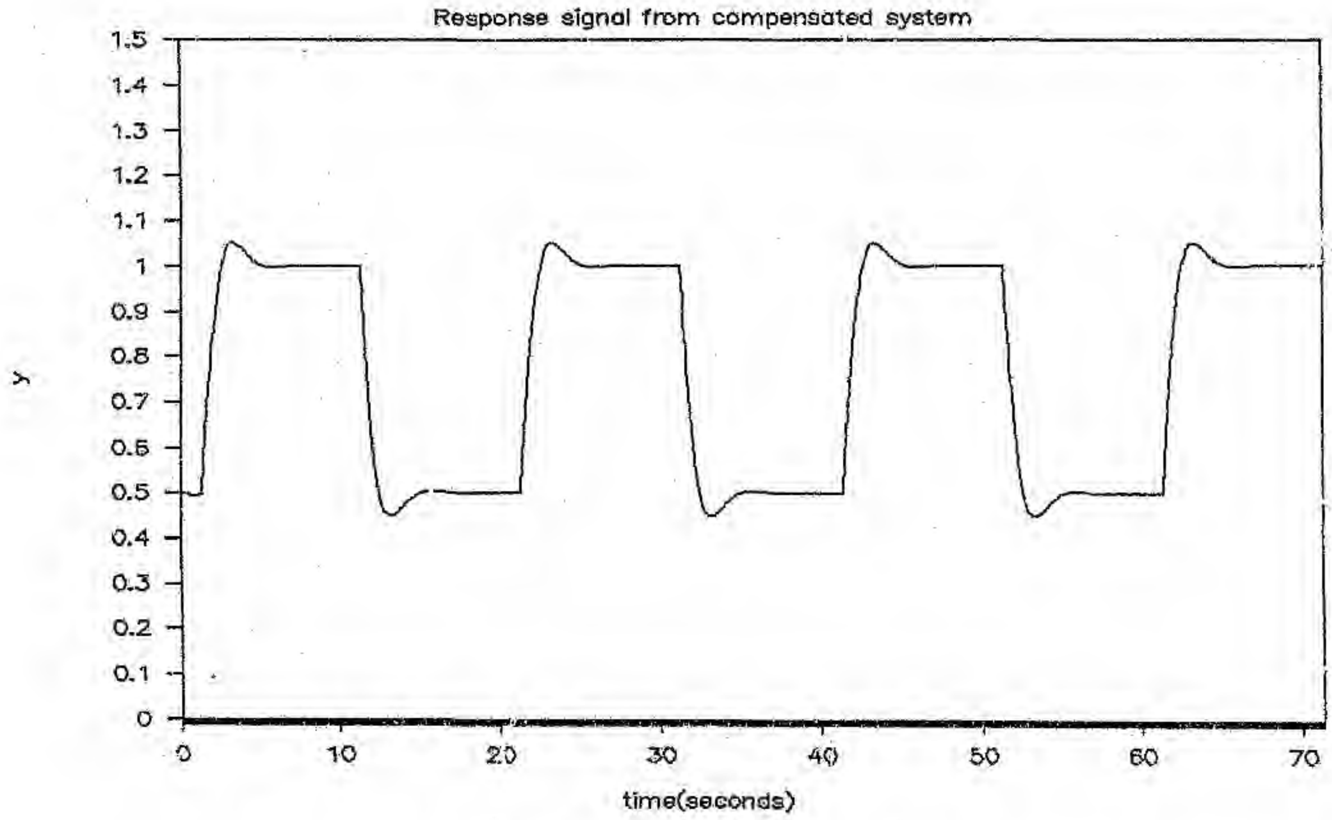
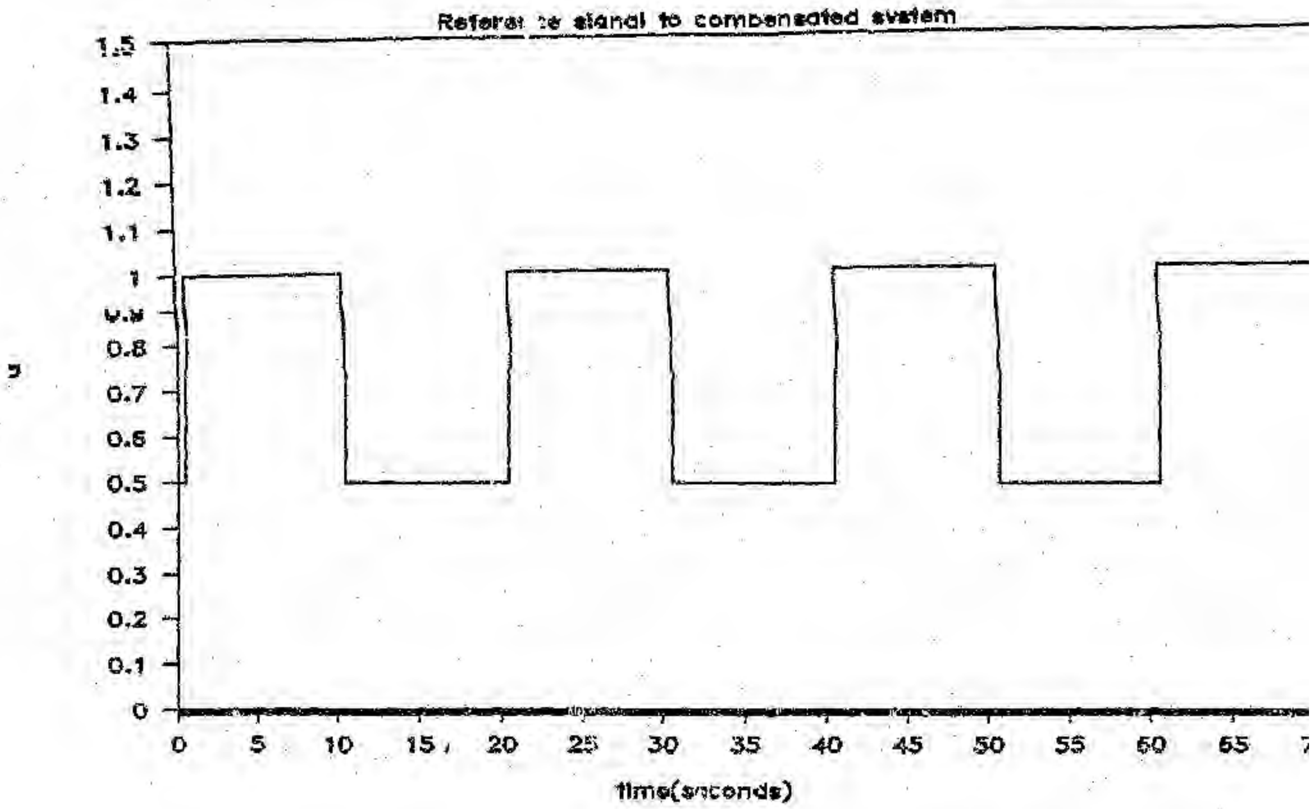


Figure 7.5: Response of the System to a Step Series



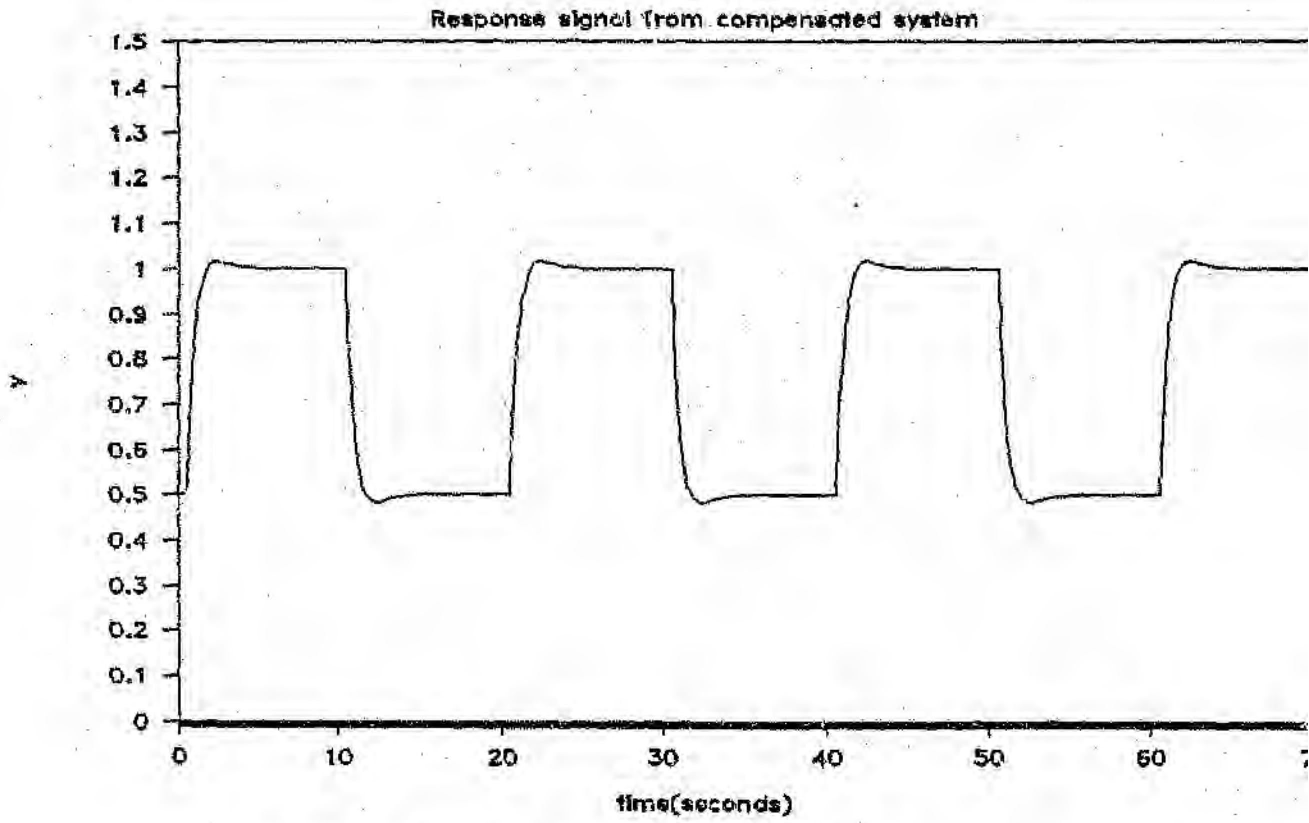


Figure 7.6: Response of the System to a Step Series

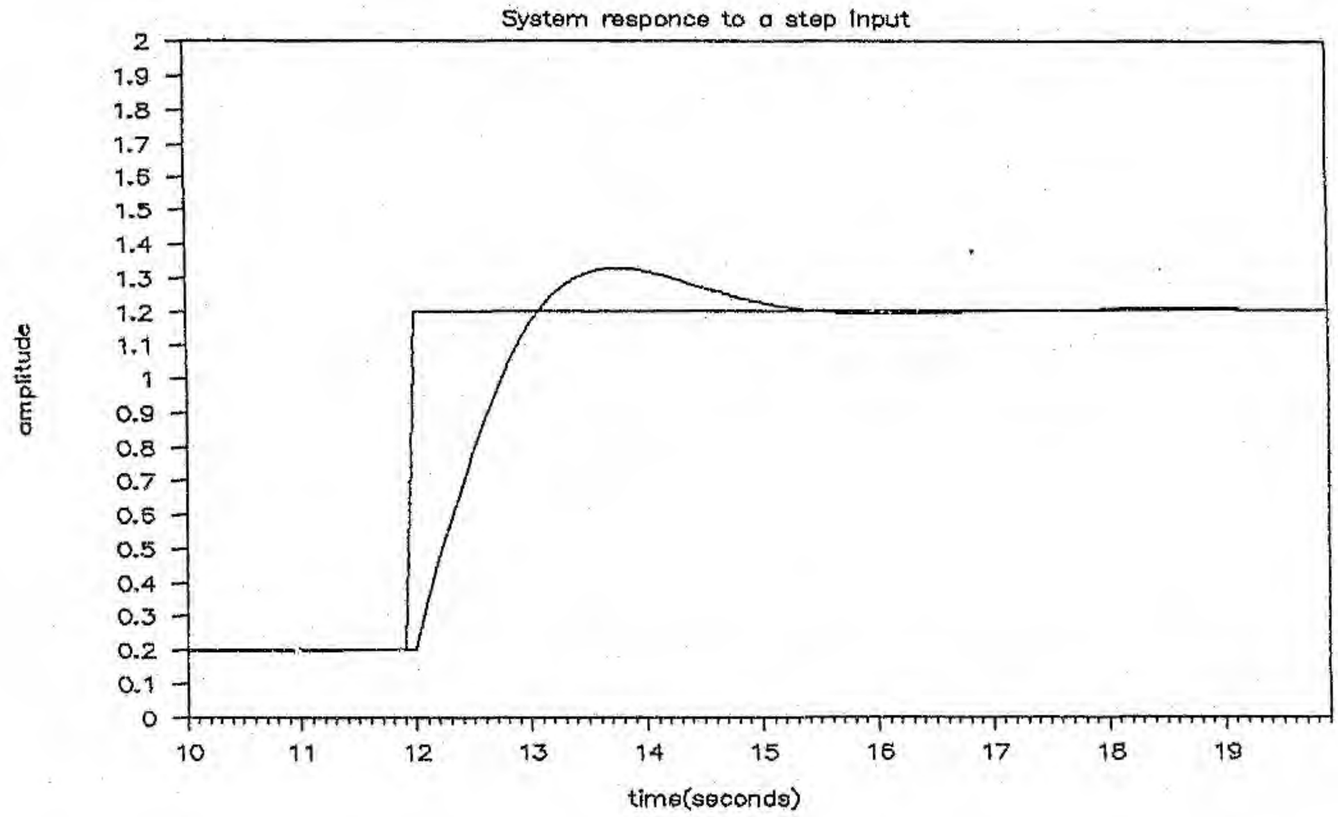


Figure 7.7: Response of the Compensated System to a Step Input

7.3 Simulation

The simulation used a second order system described by the differential equation.

$$\frac{d^2y(t)}{dt^2} + 0.5\frac{dy(t)}{dt} + 0.2y(t) = \frac{du(t)}{dt} + 1.5u(t) + \frac{d^2\eta(t)}{dt^2} + \frac{d\eta(t)}{dt} + \eta(t)$$

Using the Laplace transform, the system becomes

$$(s^2 + 0.5s + 0.2)\,y(s) = (s + 1.5)U(s) + (s^2 + s + 1)\,\eta(s)$$

with $\eta(s)$ the random disturbance signal.

The algorithm was executed with the following fixed parameters.

$w_c = 0.5$ LPF cutoff frequency

$\Delta = 1.0$ Sampling time

$\lambda = 0.97$ Exponential forgetting factor

$P(0) = 100.0$ Initial value for the covariance matrix

$\theta(0) = 0.0$ Initial value for parameter vector

$\varepsilon = 0.0008$ HPF corner frequency

$\alpha = 0.4$ Maximum value for variable deadzone

$\beta = 1.3$ Variable deadzone scaling factor

$\sigma = 0.98$ Variable deadzone pole

$\varepsilon_0 = 0.0002$ Threshold coefficient

$\varepsilon_1 = 0.0001$ Variable deadzone coefficient for input signal

$\varepsilon_2 = 0.00005$ Variable deadzone coefficient for output signal

The algorithm was used to estimate the first order δ model of the simulated plant. A series of very fast changing steps is used as the input signal to the process. The plant output $y(k)$ and the estimated output $\hat{y}(k)$ are shown in Figure 7.8 indicating good tracking ability by the algorithm. The low pass filter manages to partially negate the effect of the disturbance on the estimator. The estimation error shown in Figure 7.9 is substantially large as expected due to the effect of the disturbance $\eta(k)$ also shown in Figure 7.9.

The convergence of the estimated parameters is also influenced by the random disturbance. Although they converge to the right region quite rapidly, the parameter updating is sustained for much of the simulation run time. The plot of the estimated parameters is shown in Figure 7.10. The presence of $\eta(k)$ influences the converging ability of the algorithm.

The algorithm produced the estimated model.

$$\delta\, y(k) + 0.172378\, y(k) = 1.273907\, u(k)$$

with

$$a_0 = 0.172378$$

$$b_0 = 1.273907$$

The parameter estimates produced by the algorithm were compared against the estimates produced by a sophisticated commercial routine. The same data were used by the Matlab Arx estimation routine. The resulting first-order plant was:

$$qy(k) - 0.8225\, y(k) = 1.2693\, u(k)$$

Using the relationship,

$$q = \delta\Delta + 1$$

and for $\Delta = 1.0$, the q-model becomes

$$\delta\, y(k) + 0.1775\, y(k) = 1.2693\, u(k)$$

which is very similar to the δ-operator model given by the estimator. A plot of the response of the δ-operator model produced by the estimator and the response of the q-operator model given by the Matlab Arx routine is shown in Figure 7.11. As it was expected, they were very similar.

The performance of the estimator was also examined using the variable deadzone option. The variable deadzone $a(k)$ takes zero values only when the estimation $e(k)$ is less than $\beta\, m(k)$. The magnitude of $a(k)$ after the first sample is limited by the value of $\alpha = 0.4$.

The deadzone $\alpha(k)$ has remained mostly at the α region indicating that the error remained greater than the deadzone limits. The plot of the bmz and deadzone can be seen in Figure 7.12. The $bm(k)$ limit remained around the $bmz = 0.003$ region for the duration of the simulation. It is again clear that the disturbance influences the convergence of the estimator. It must be noted that since ε_1 is greater than ε_2 the deadzone error limit $bm(k)$ is influenced a lot more by the input $u(k)$ and obviously, the disturbance $\eta(k)$ than the output $y(k)$.

The estimated plant parameters given by the estimator when the variable deadzone was used were:

$$a_0 = 0.259151$$

$$b_0 = 1.898636$$

which are not as appropriate as the estimated parameters obtained with the fixed deadzone.

The results obtained with the use of the variable deadzone indicated that the fixed deadzone can produce even better results than the variable deadzone with less complexity and calculation time.

The specification for the closed-loop system were: a) percent overshoot $P.O. =$ 10% and b) settling time $T_s = 25$ seconds. This resulted in:

$$\begin{aligned} \zeta &= 0.5912 \\ w_n &= 0.2710 \\ w_{3dB} &= 0.4744 \\ \Psi_{pm} &= 59.0 \end{aligned}$$

The cutoff frequency of the low-pass filter is $w_c = 0.5$ rad/sec. It adheres to the first rule.

$$w_c \leq \frac{1}{2}\, w_s$$

with $w_s = 6.283$ rad/sec and also to the second rule

$$w_b \leq w_c$$

In this case, w_c is almost equal to the closed-loop bandwidth so that the random disturbance does not influence the convergence of the estimated parameters. The algorithm ensures that the designed closed-loop bandwidth does not exceed the LPF cutoff frequency.

The closed-loop polynomial A^* was given by

$$A^* = \delta^2 + 0.3204\,\delta + 0.0734$$

having the roots

$$s_1 = -0.1602 + 0.43704\,j$$

$$s_2 = -0.1602 - 0.43704\,j$$

The roots of $A^*(\delta)$ are illustrated in Figure 7.13.

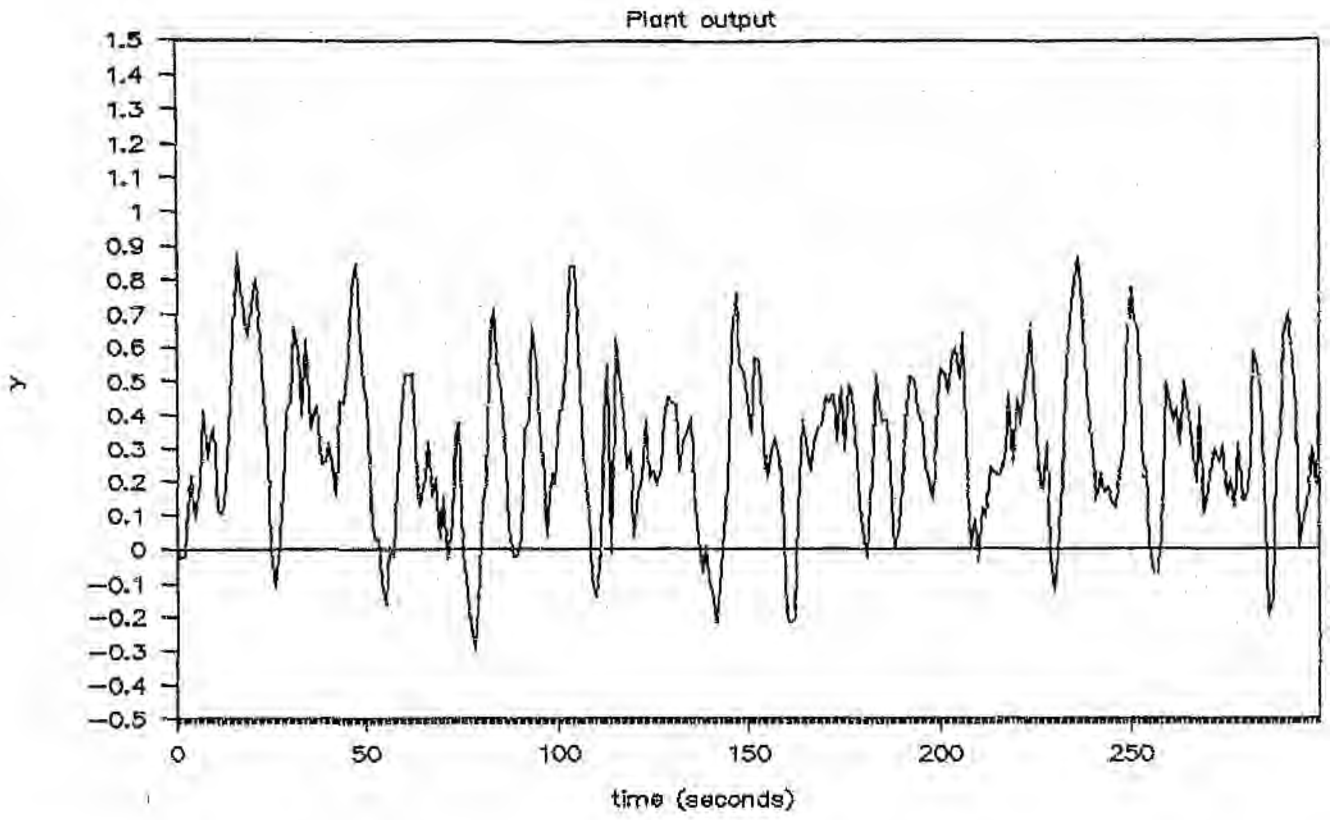
The controller parameters produced by the algorithm are given below as:

$$\begin{aligned} K_p &= 0.1159 \\ K_i &= 0.4963 \quad \text{or} \\ T_i &= 2.0150 \end{aligned}$$

The closed loop response to a step input is shown in Figure 7.14. The settling time of the compensated system is within the $T_s = 25$ seconds requirement.

7.3.1 Summary

The algorithm operated under severe conditions caused by the presence of the random disturbance. It achieved to produce a good first-order model for the plant under investigation. The use of a variable deadzone did not produce better results than the fixed deadzone. Also, the robustness of the algorithm was enhanced by the use of the low-pass filter. Finally, the controller design achieved to produce PI parameters that enabled the closed-loop system to meet the design specifications.



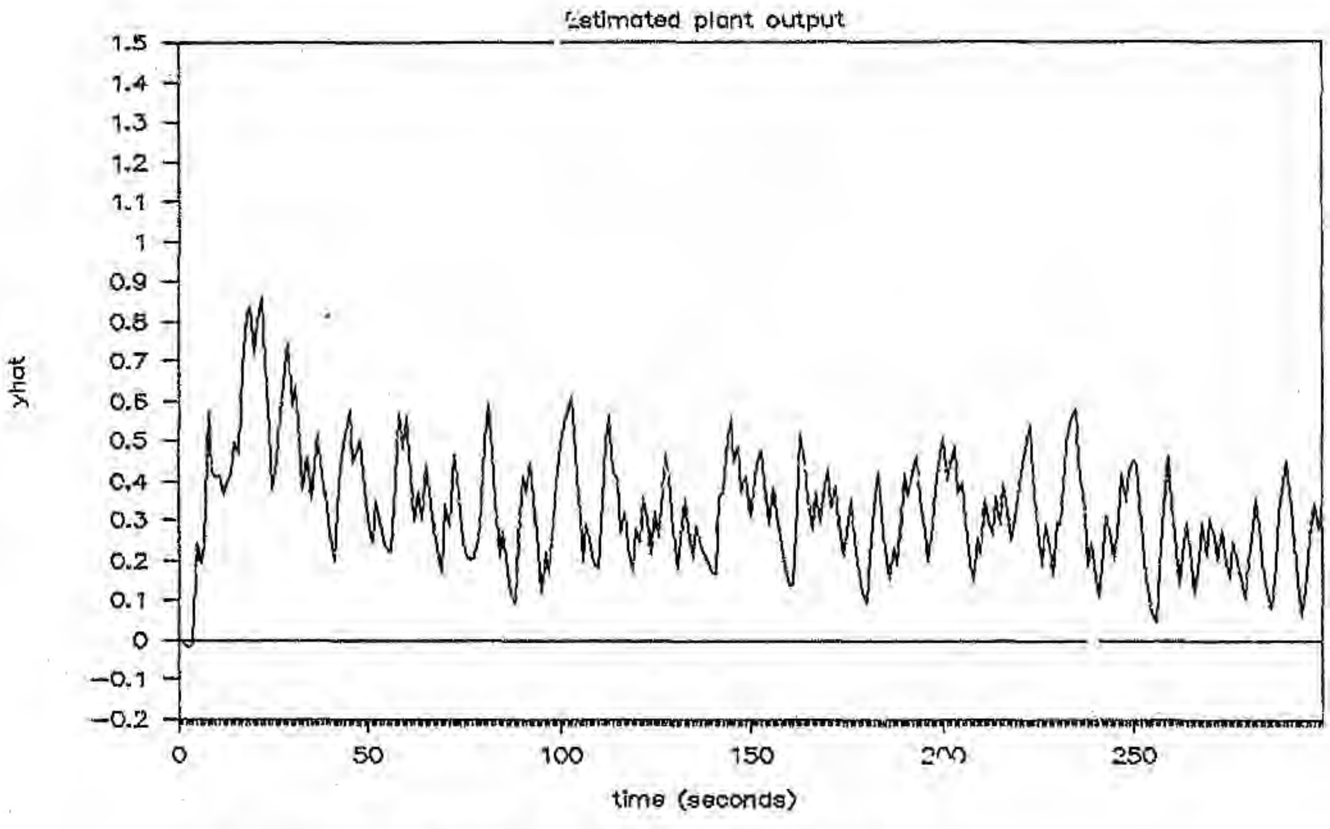
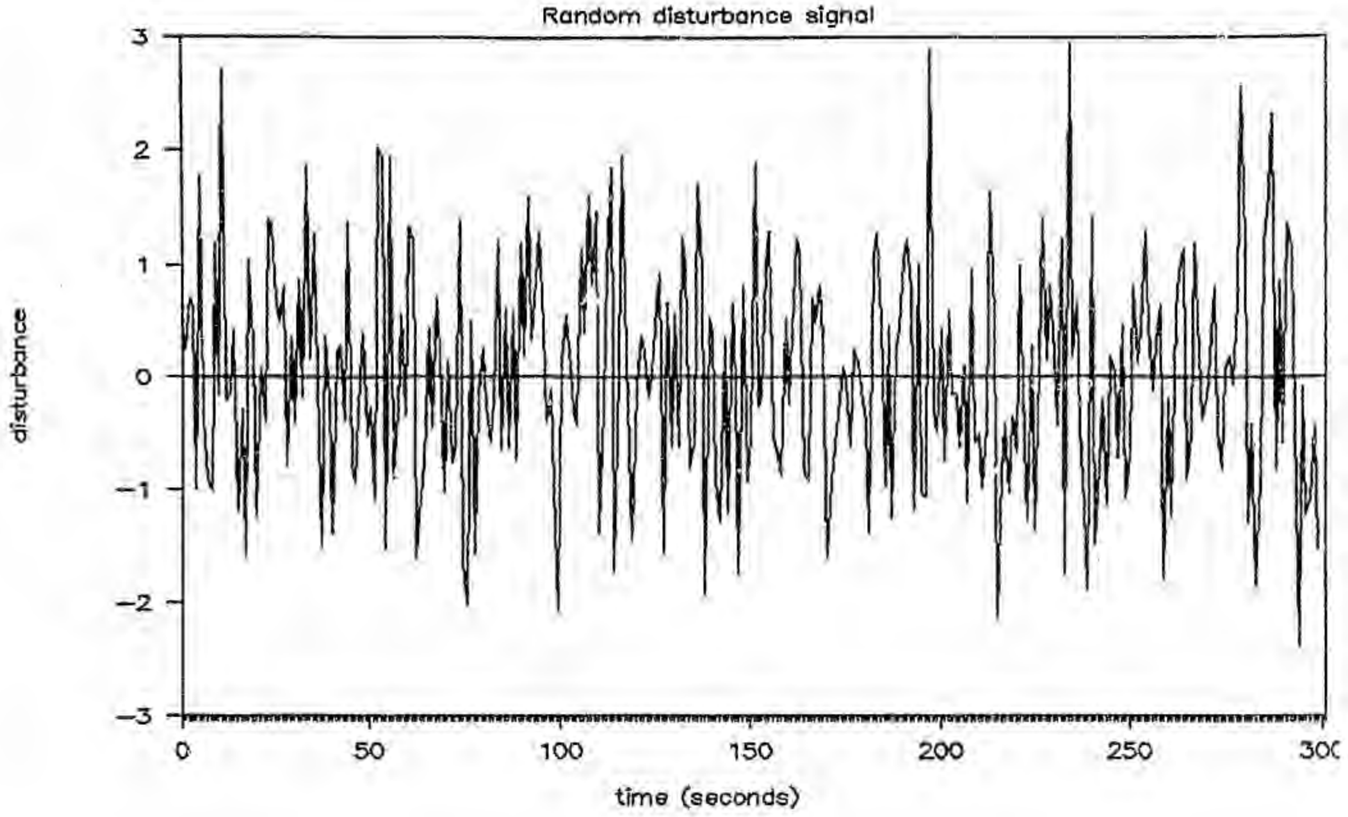


Figure 7.8: Plot of the Real and Estimated Plant Values



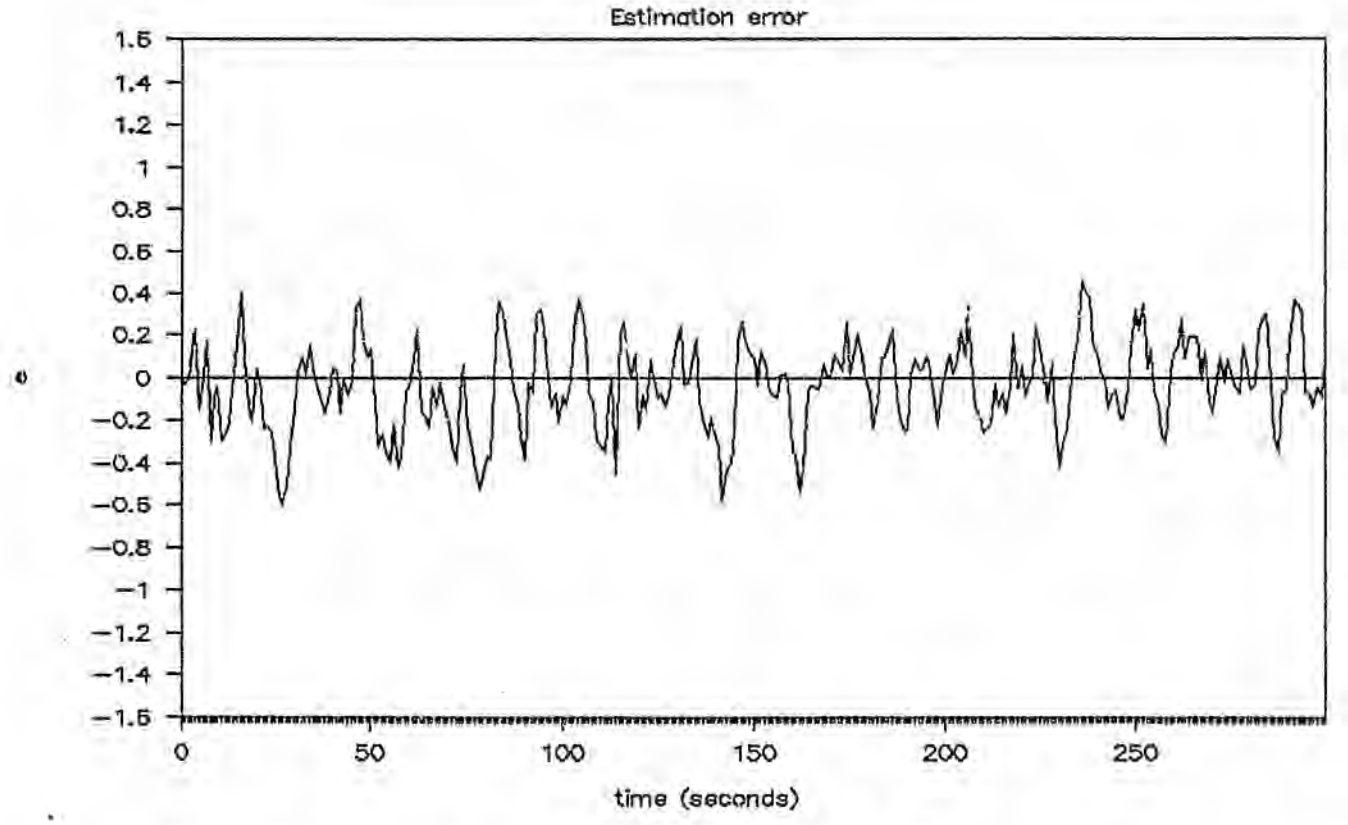
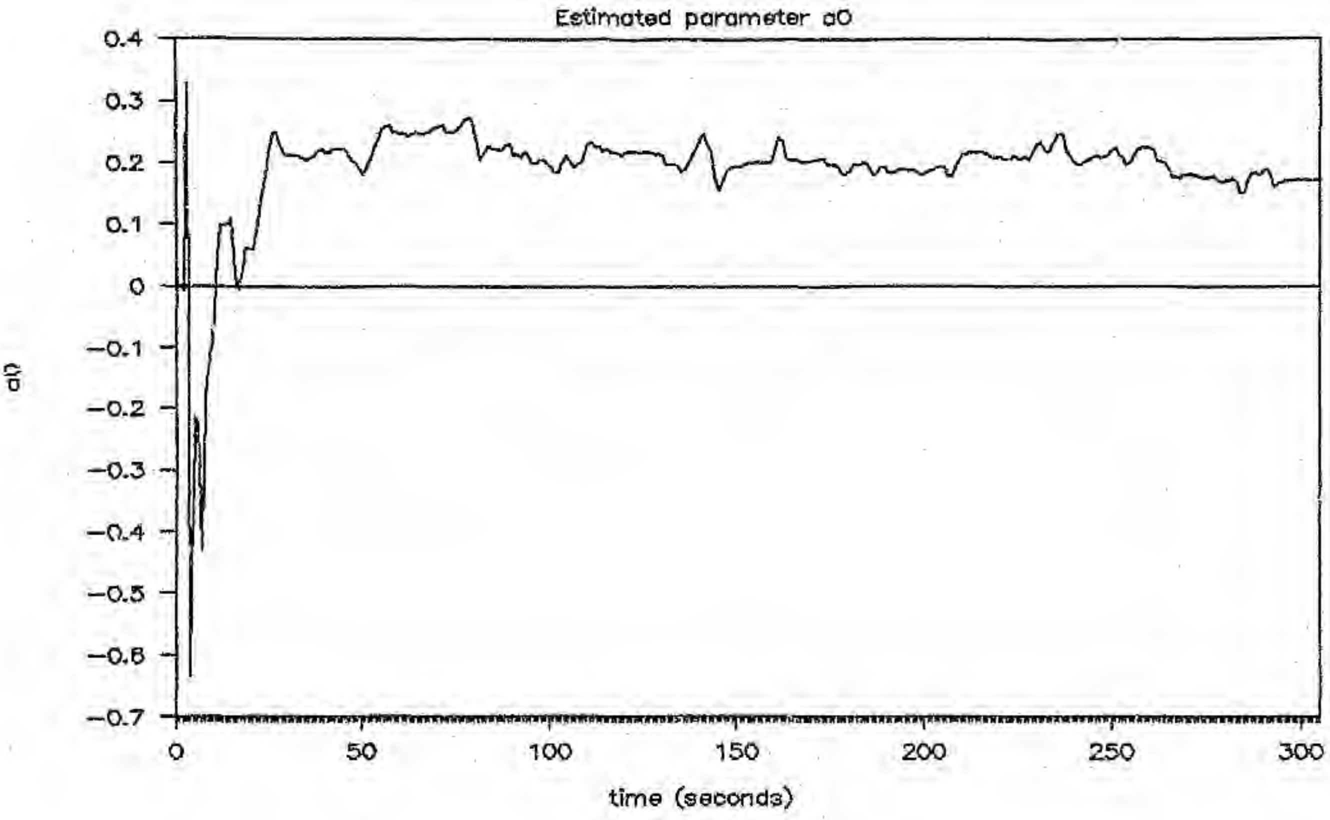


Figure 7.9: Plot of the Random Disturbance and Estimation Error



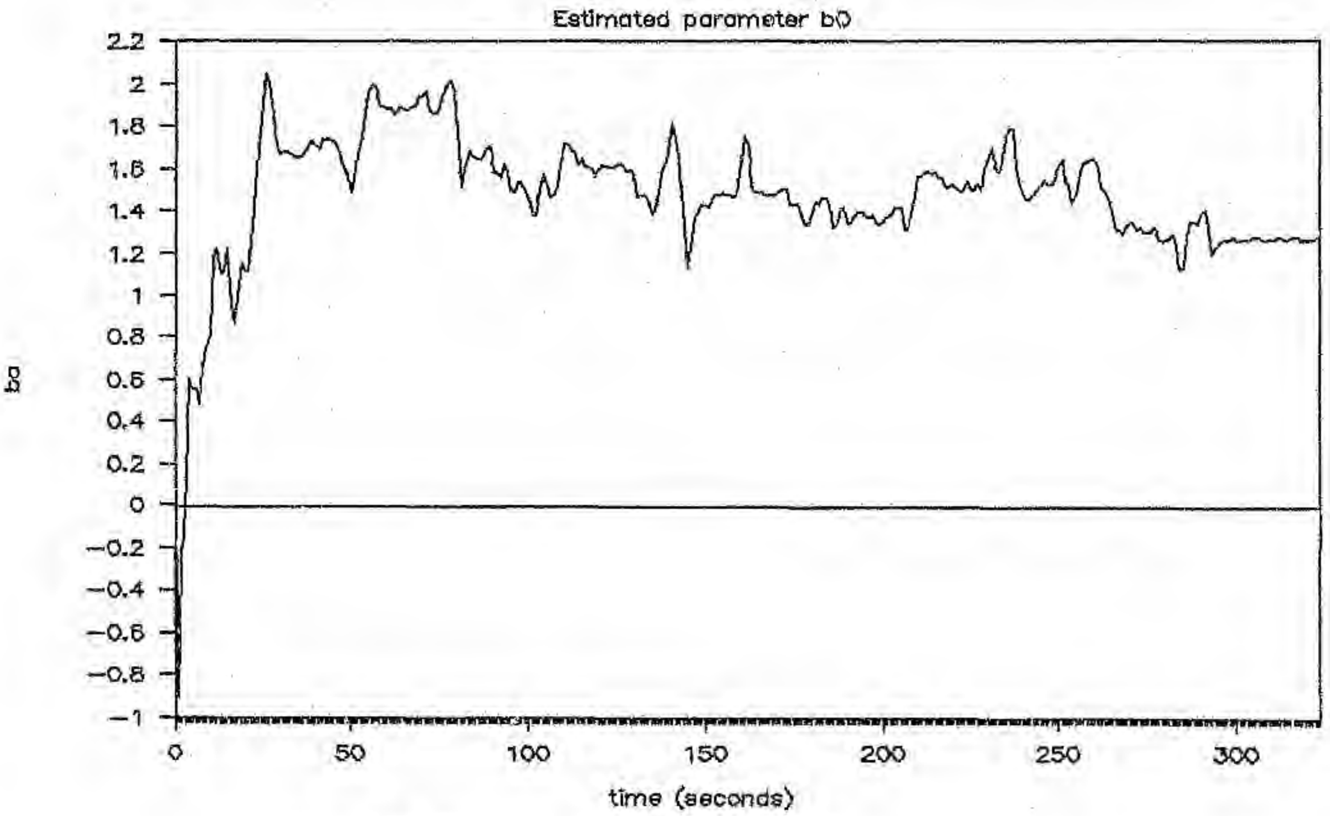
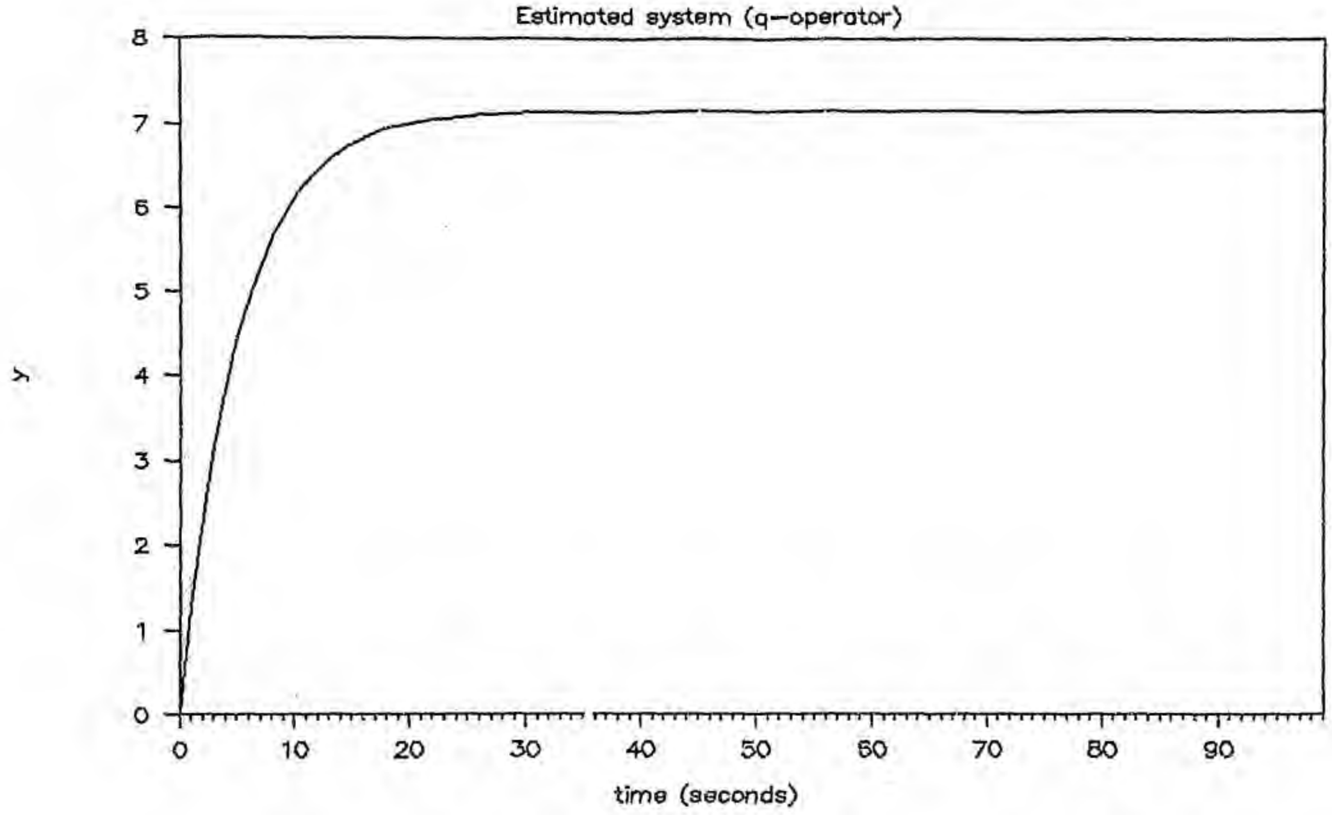


Figure 7.10: Plot of the Estimated Parameters



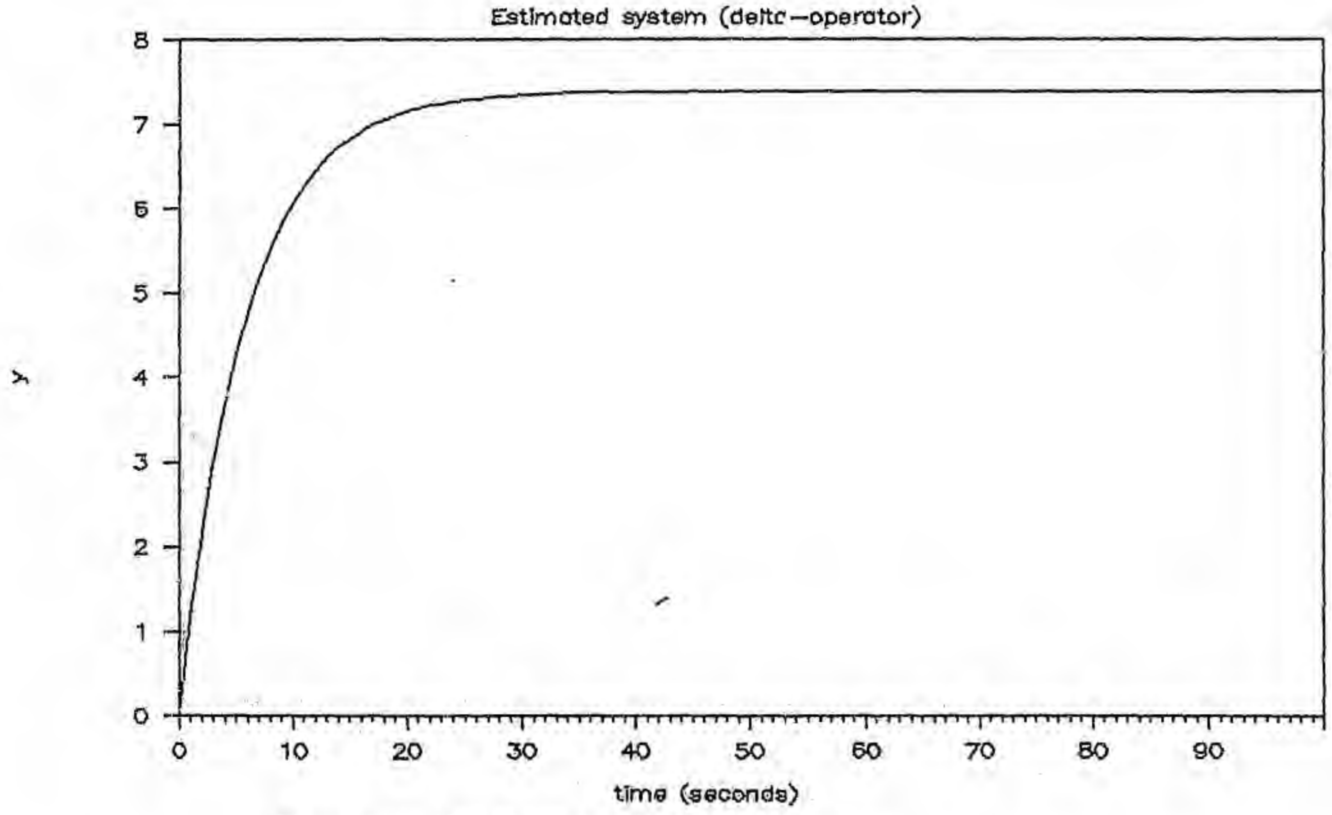
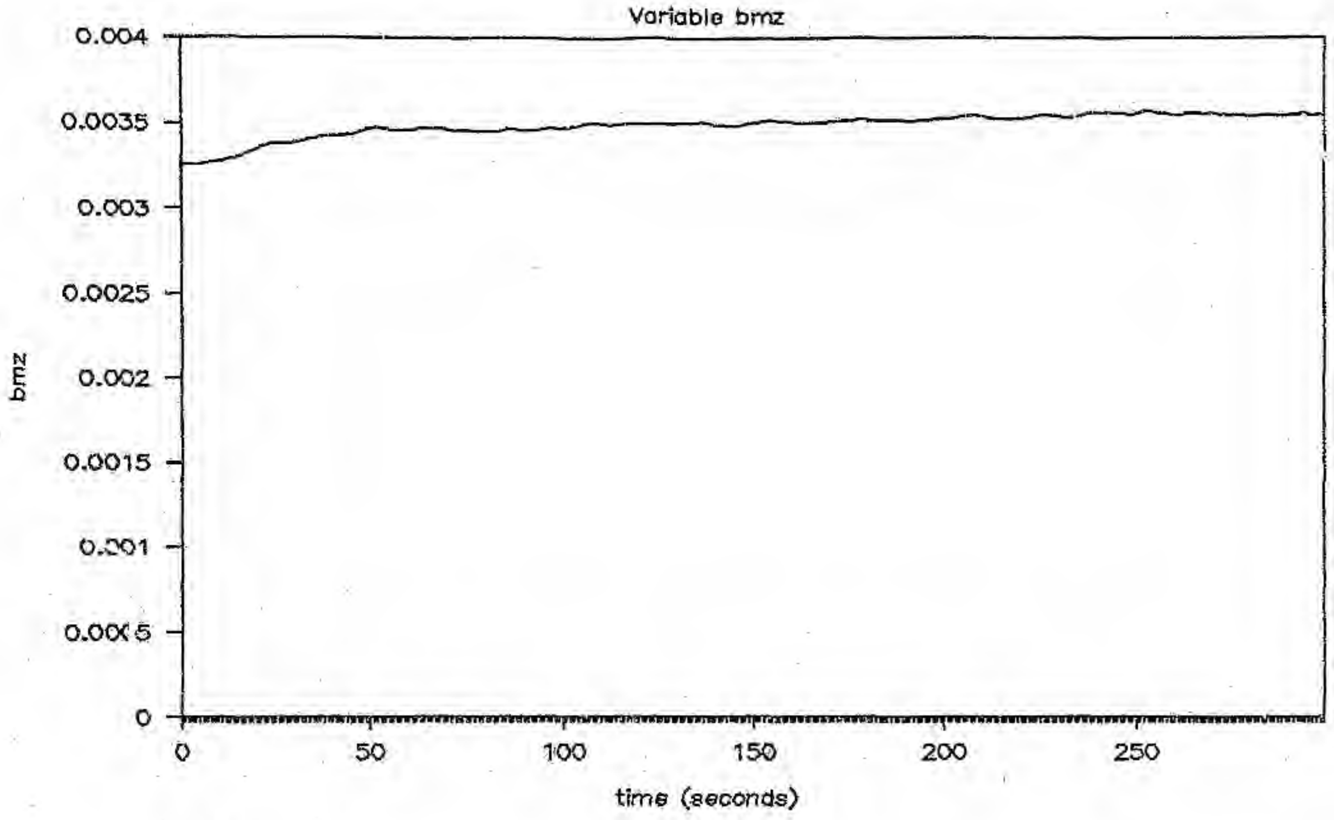


Figure 7.11: Plot of the Two Estimated Models



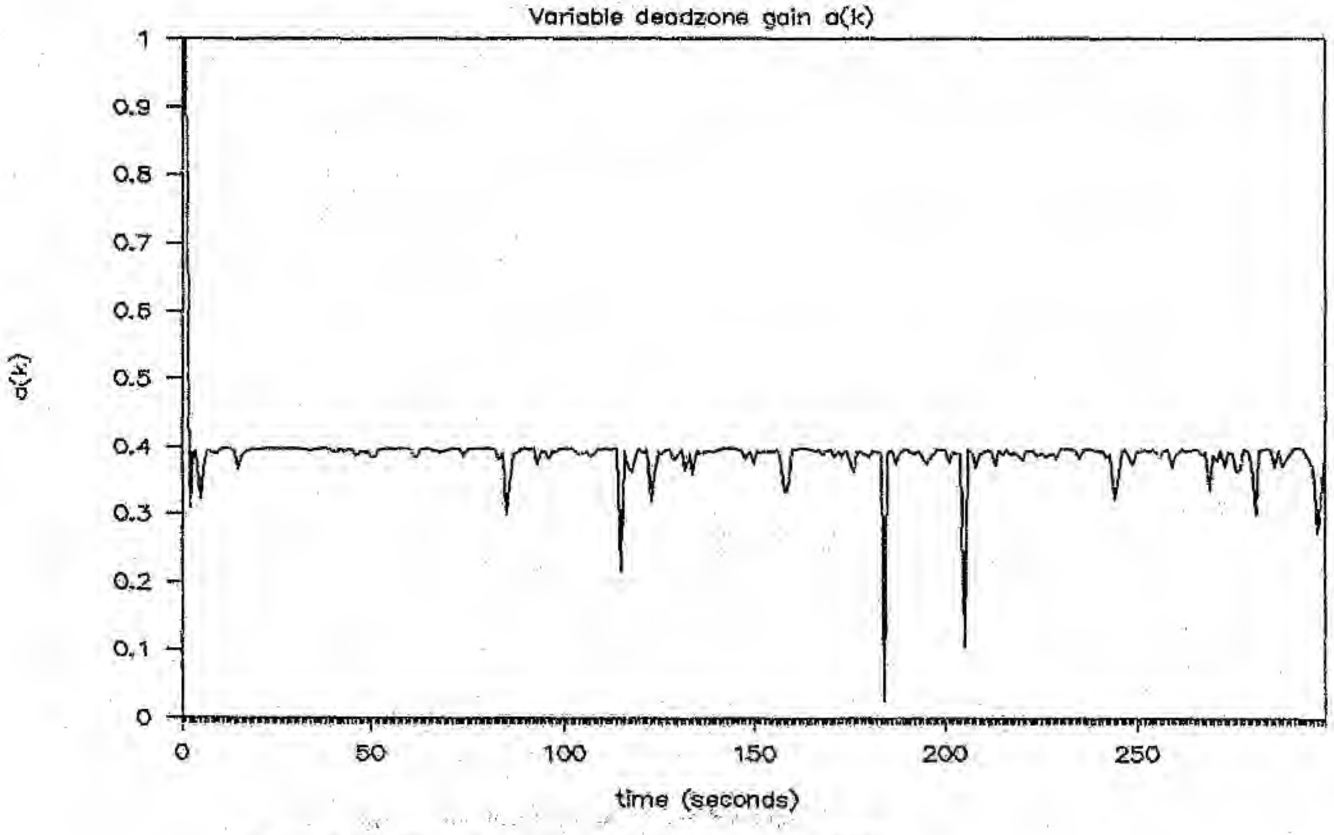


Figure 7.12: Plot of the Variable Deadzone and BMZ

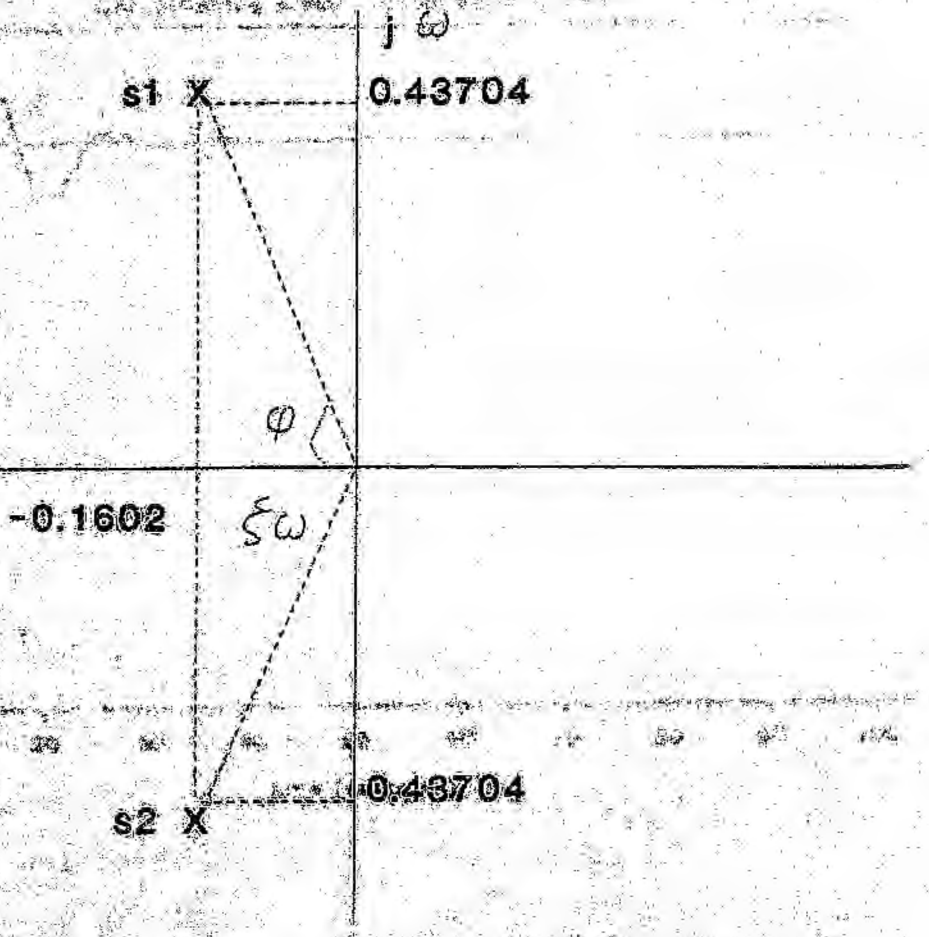


Figure 7.13: Root Locus of the System

7.4 Real Plant Test

The performance of the algorithm was examined using real-time plant data. The aim of the test was to observe the behavior and robustness of the estimator under plant operating conditions. The Raw Gas Compressors Section of the Ammonia Plant, located at the AECI Modderfontein Factory, was used for the estimator trial tests.

The two raw gas compressors deliver raw gas, produced by the Gasification Section, to the CO Conversion Section where the hydrogen needed for the production of liquid ammonia is extracted from the raw gas. The compressors are driven by two Delaval steam turbines. The speed of the compressors is controlled by two Compressors Control Corporation (CCC) Load Sharing Controllers. The 4-20 mA control signal varies the position of the steam inlet multiports thus controlling the steam flow into the turbine. The load sharing controllers are part of a complex control scheme that controls the load and, consequently, the speed of each compressor based on the level of the accumulated raw gas inside the Raw Gas Holder and, also, on the position of the operating point of each compressor in comparison to the surge curve of each machine. An illustration of the process is given in Figure 7.15.

The estimated model relates the load sharing control signal to the steam flow into the turbine. The plant data used by the estimator were taken by sampling the load sharing control signal (LYB 8123) and the B stream steam flow signal (FIB 8101). The signals were sampled every ten seconds. The loop was operating under closed loop conditions and the control signal was sufficiently act[illegible] was not necessary to inject any kind of special test signal.

The estimator was set up for the parameter identification test with the following parameters defined.

$w_c = 0.3$ LPF cutoff frequency

$\Delta = 10$ Sampling time

$\lambda = 0.97$ Exponential forgetting factor

$P(0) = 100$ Initial value for covariance matrix

$\hat{\theta}(0) = 0.0$ Initial value for parameter vector

$n = 0.0008$ HPF corner frequency

$\alpha = 0.4$ Maximum value for variable deadzone

$a = 1.3$ Variable deadzone scaling factor

$n_0 = 0.00025$ Threshold coefficient

$e = 0.98$ Variable deadzone pol[illegible]

$\varepsilon_1 = 0.0001$ Variable deadzone weight for input

$\varepsilon_2 = 0.00005$ Variable deadzone weight for output

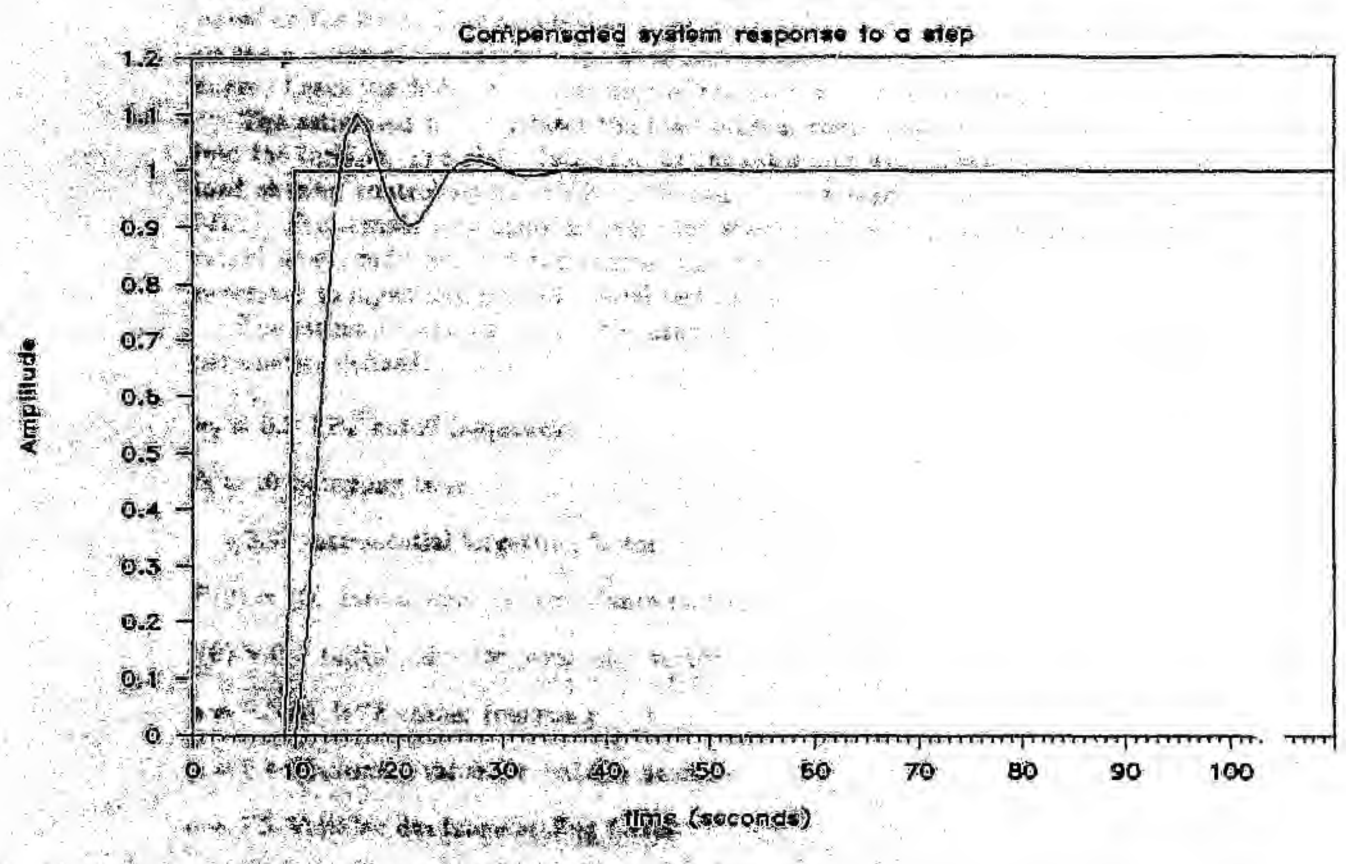


Figure 7.14: System Response to a Step Input

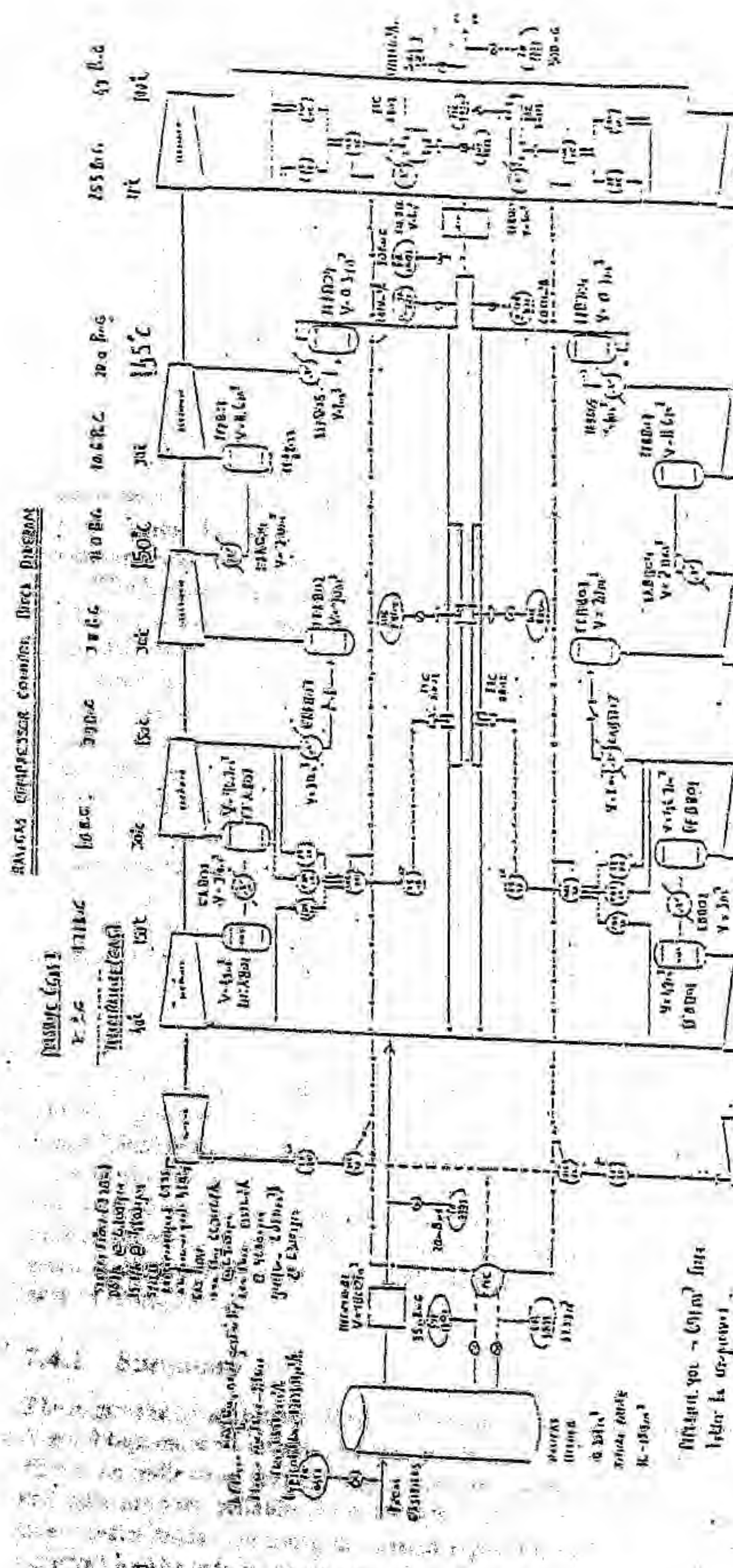


Figure 7.15: Block Diagram of the Process

With the control signal LYB 8123 as the plant input and FIB 8101 as the plant output, the estimator ran for a duration of 50 minutes. The estimated plant parameters were:

$$\begin{aligned} a_0 &= 0.029910 \\ b_0 &= 0.028978 \end{aligned}$$

The estimated plant model is given by

$$\delta\, y(k) + 0.029910\, y(k) = 0.028978\, u(k)$$

The plots of the plant output FIB 8101 and the estimated plant output @FIB 8101 are shown in Figure 7.16. Also, Figure 7.17 shows the plot of the input signal LYB 8123. A comparison between the FIB 8101 and @FIB 8101 plots shows good agreement between the measured plant output and the predicted output from the model using the same input. It must be noted that the plant output was affected by measurement noise and non-linearities that were not incorporated into the model.

The low-pass filter performed well in filtering out the high frequency noise affecting the measured signals. It is an indication that in this experiment, the $w_c = \frac{1}{2} w_s$ rule did provide a good restriction regarding the frequency range of the filter.

Heavy parameter updating occurred during the initial stages of the algorithm. The measurement noise present during the estimation influenced the speed with which the estimator converged to the real plant values. After the first 20 minutes, the estimator succeeded to converge to the correct parameter region and soon settled to the right values. A plot of the estimated parameters is given in Figure 7.18. A plot of the estimated error $e(k)$ is shown in Figure 7.17. The plot shows the error to be substantial during the initial stages of the estimation, but to minimize as the time progresses and eventually to reaches the zero value.

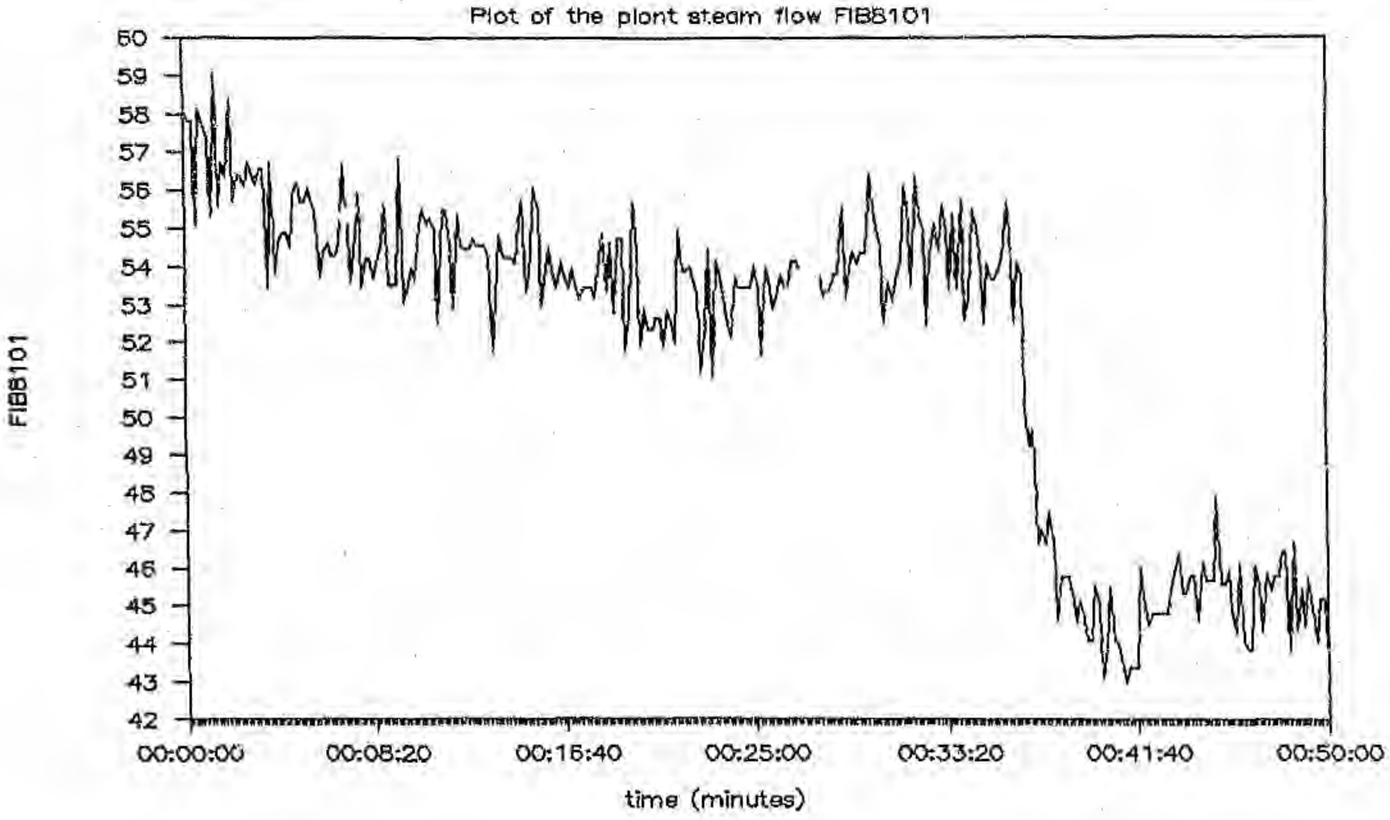
The same plant data were used by the algorithm to produce plant parameters using the variable deadzone. The algorithm produced the following parameters.

$$\begin{aligned} a_0 &= -0.100309 \\ b_0 &= -0.096794 \end{aligned}$$

It is clear that the algorithm using the variable deadzone option produced an unstable plant model. A plot of the variable deadzone and the variable error limit range (bmz) indicates that the algorithm never managed to produce estimated values close to the measured ones. The deadzone after an initial dip remained at a value $a(k) = 0.25$ and never achieved a zero value. A plot of the variable deadzone and bmz is shown in Figure 7.19. A plot of the variable deadzone estimation error is given in Figure 7.20 which also, shows the same results. It tracks the bmz without ever reaching zero.

7.4.1 Summary

The algorithm was tested using real plant data and produced very good results. A good agreement between the real plant and the estimated values was achieved. This is an indication that the algorithm can operate under adverse plant conditions and still produce reliable plant models. Although, due to plant operational and safety restrictions, the controller design algorithm was not tested. The good results achieved by the estimator indicate that the overall system can be used in the plant environment.



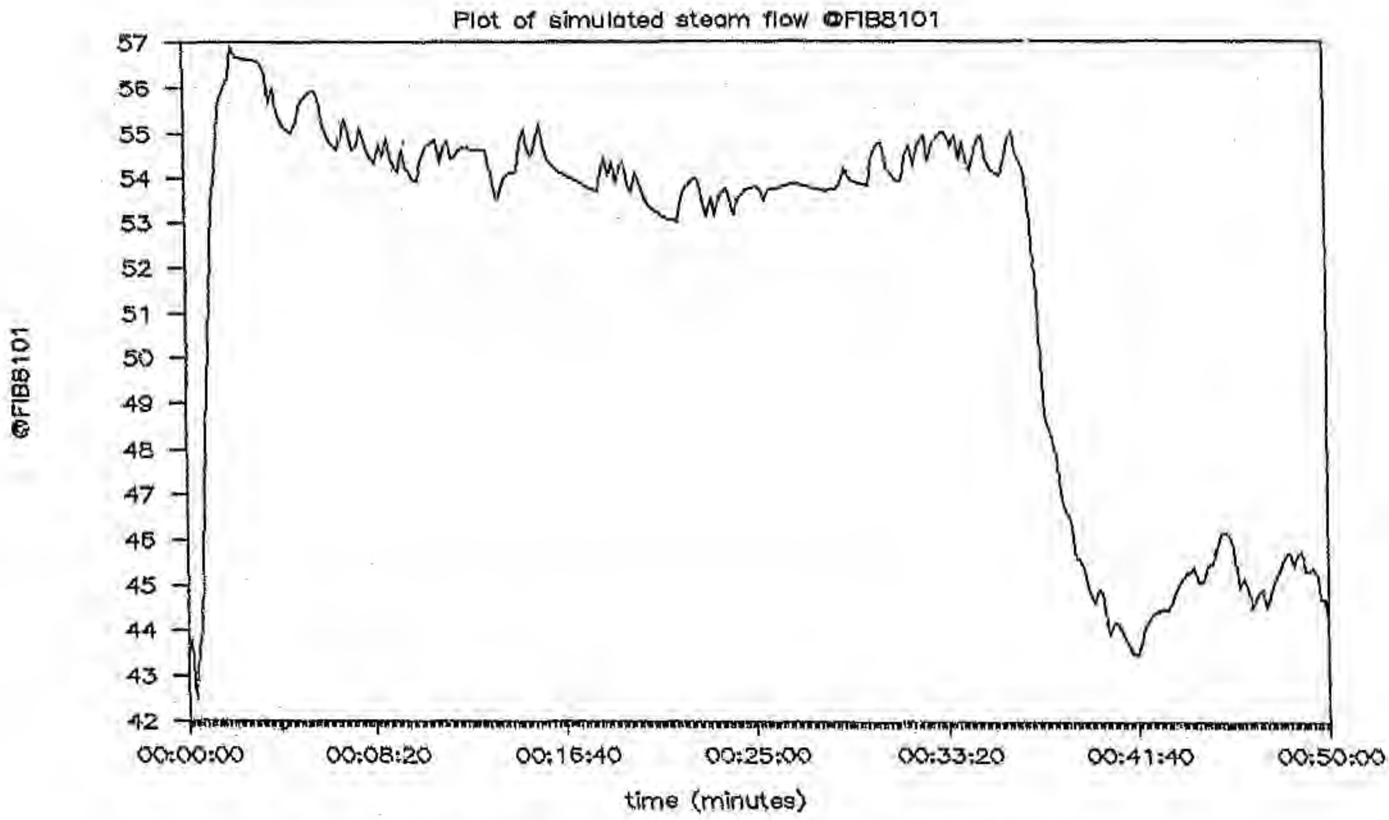
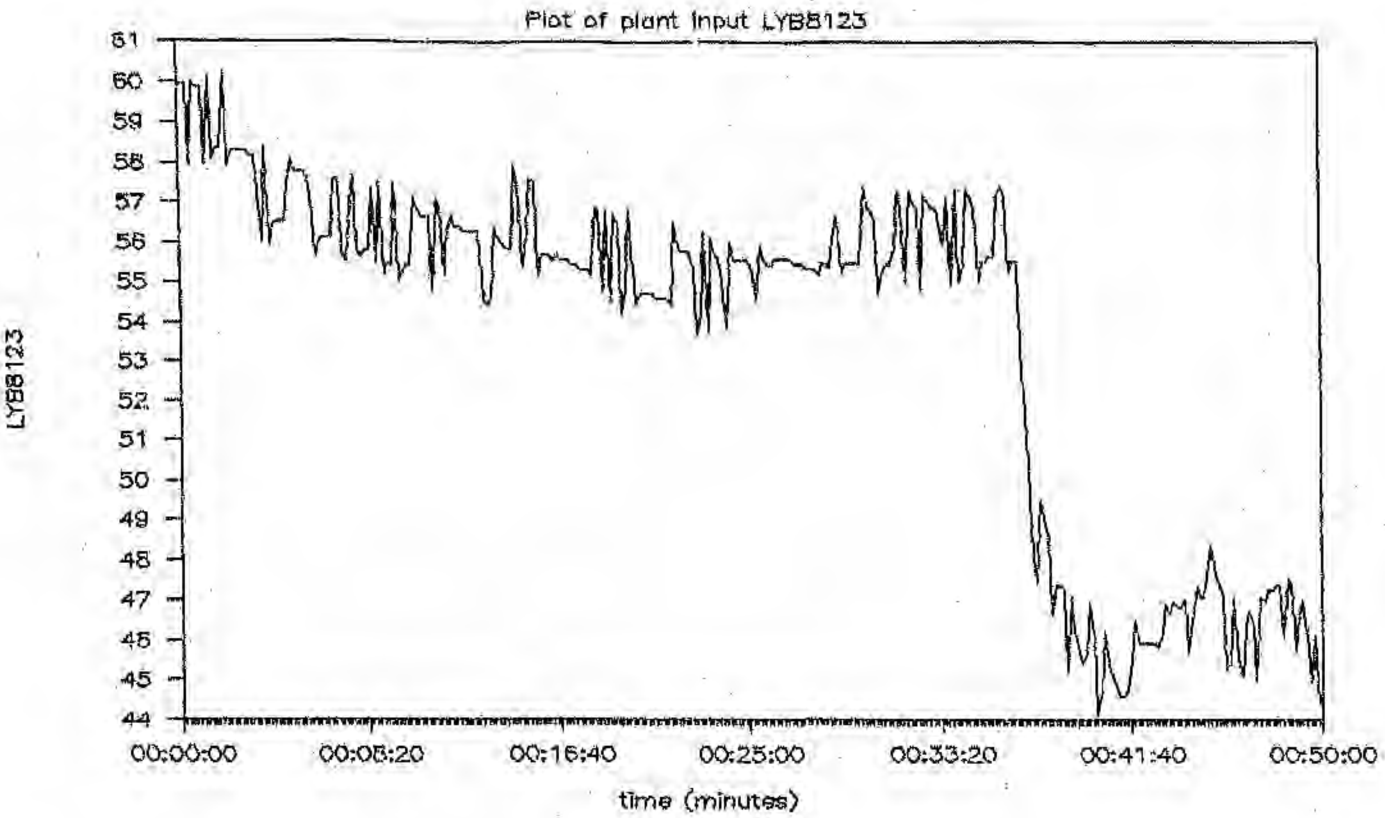


Figure 7.16: Plot of the Real and Estimated Plant Output



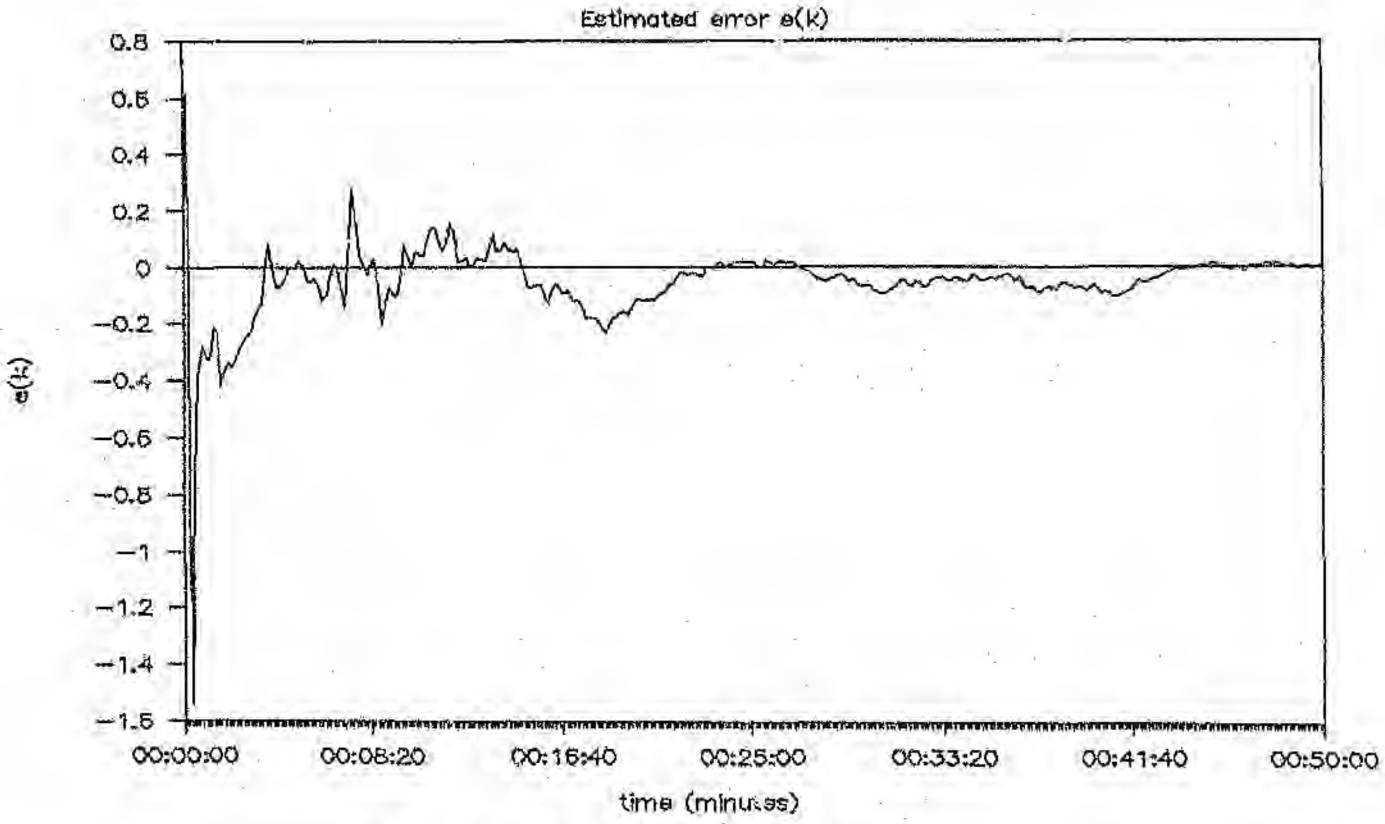
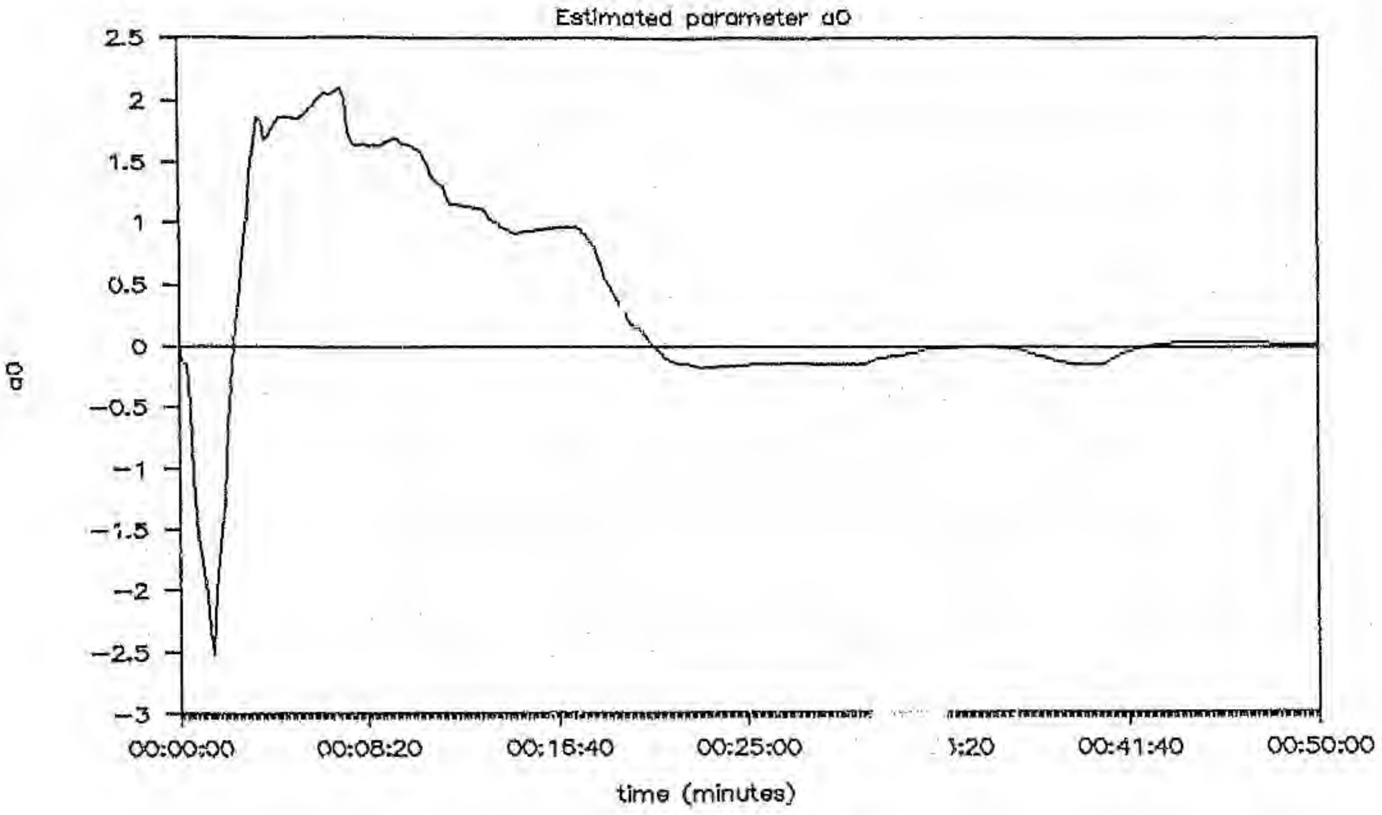


Figure 7.17: Plot of the Plant Input and the Estimation Error



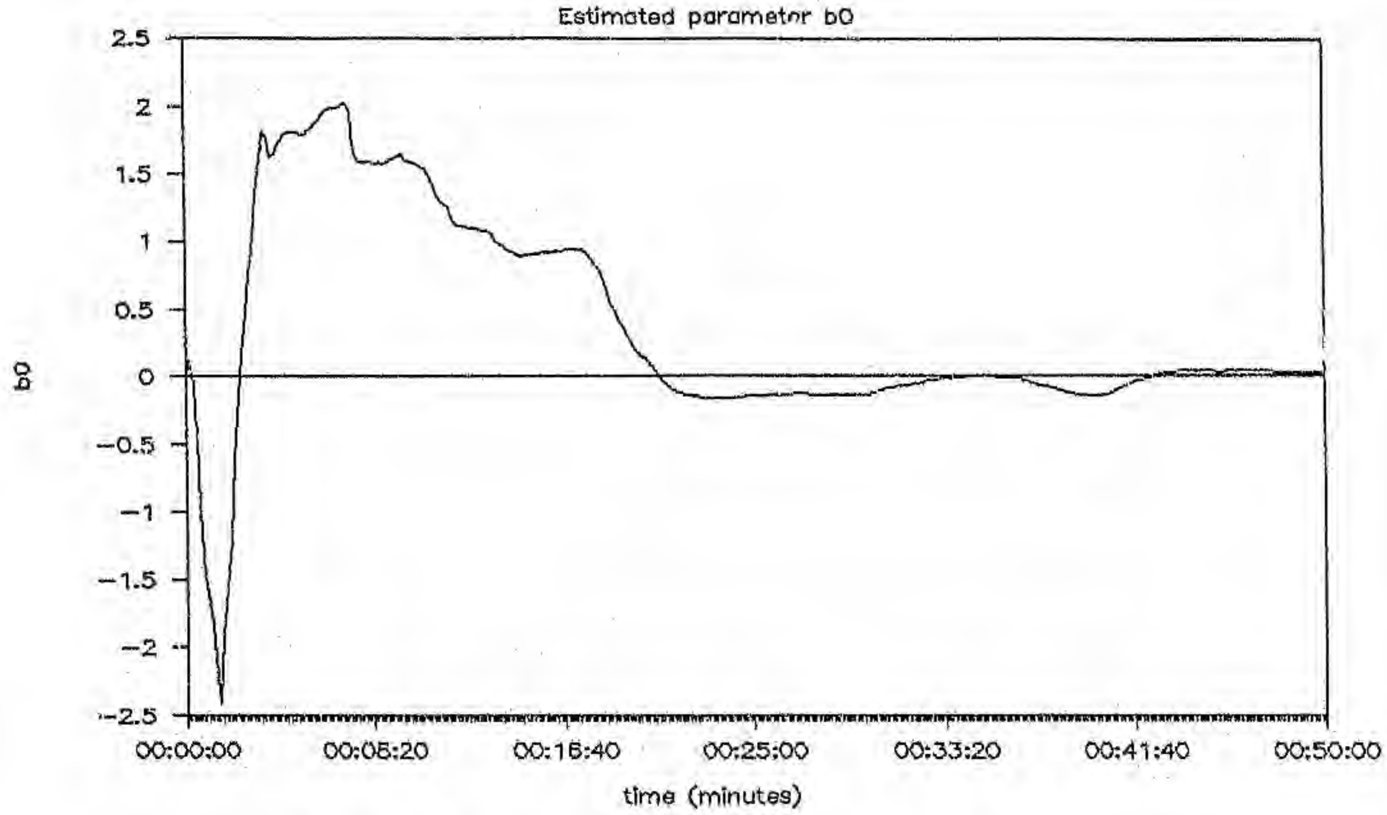
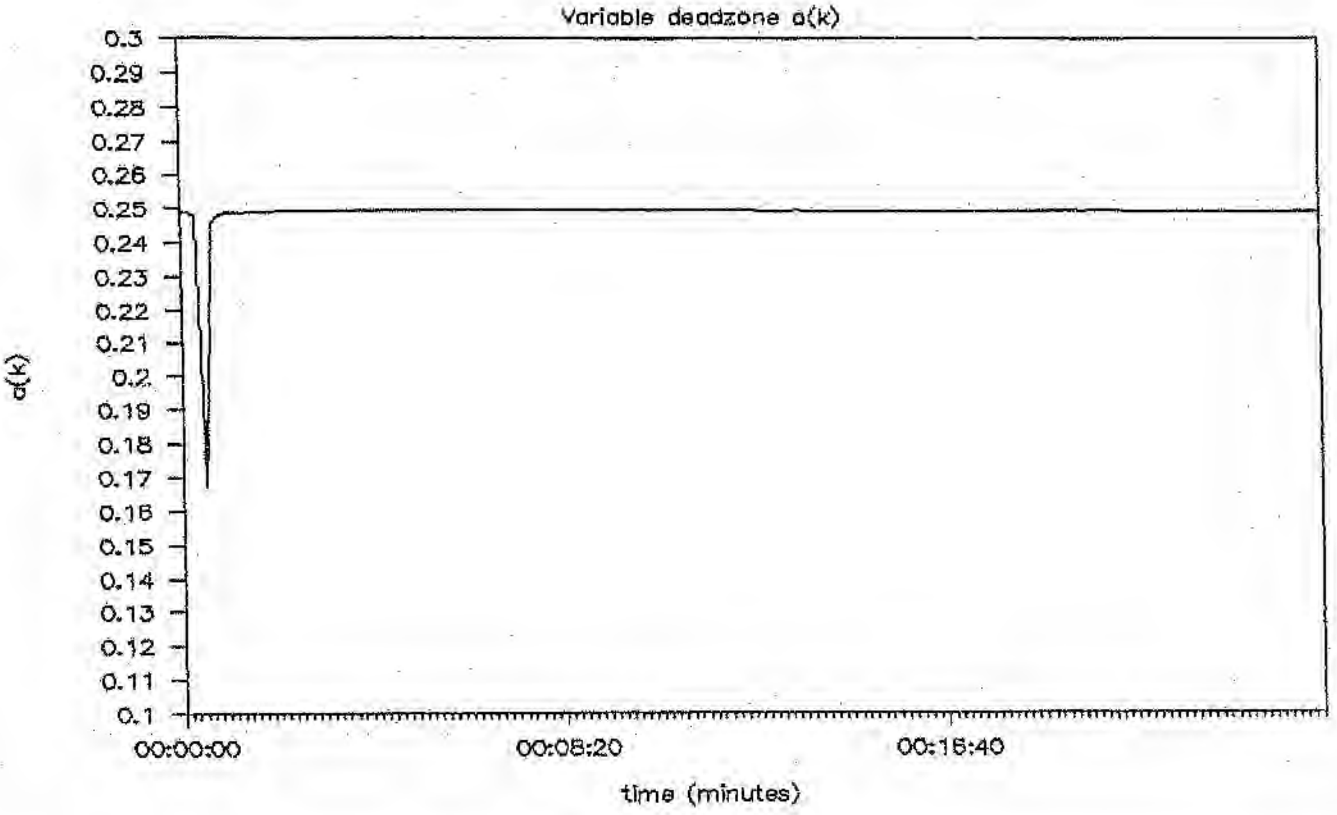


Figure 7.18: Plot of the Estimated Plant Parameters



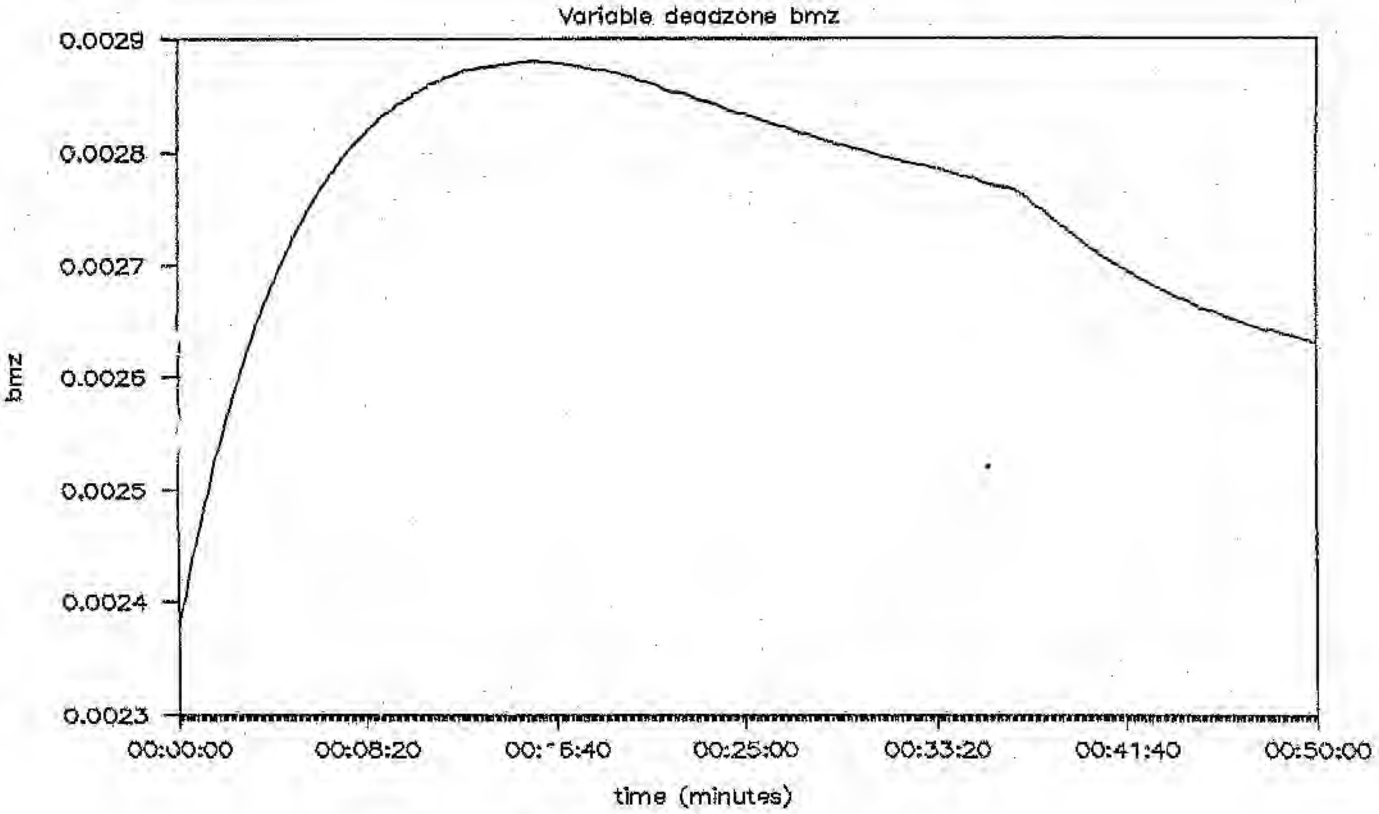


Figure 7.19: Plot of the Variable Deadzone and BMZ

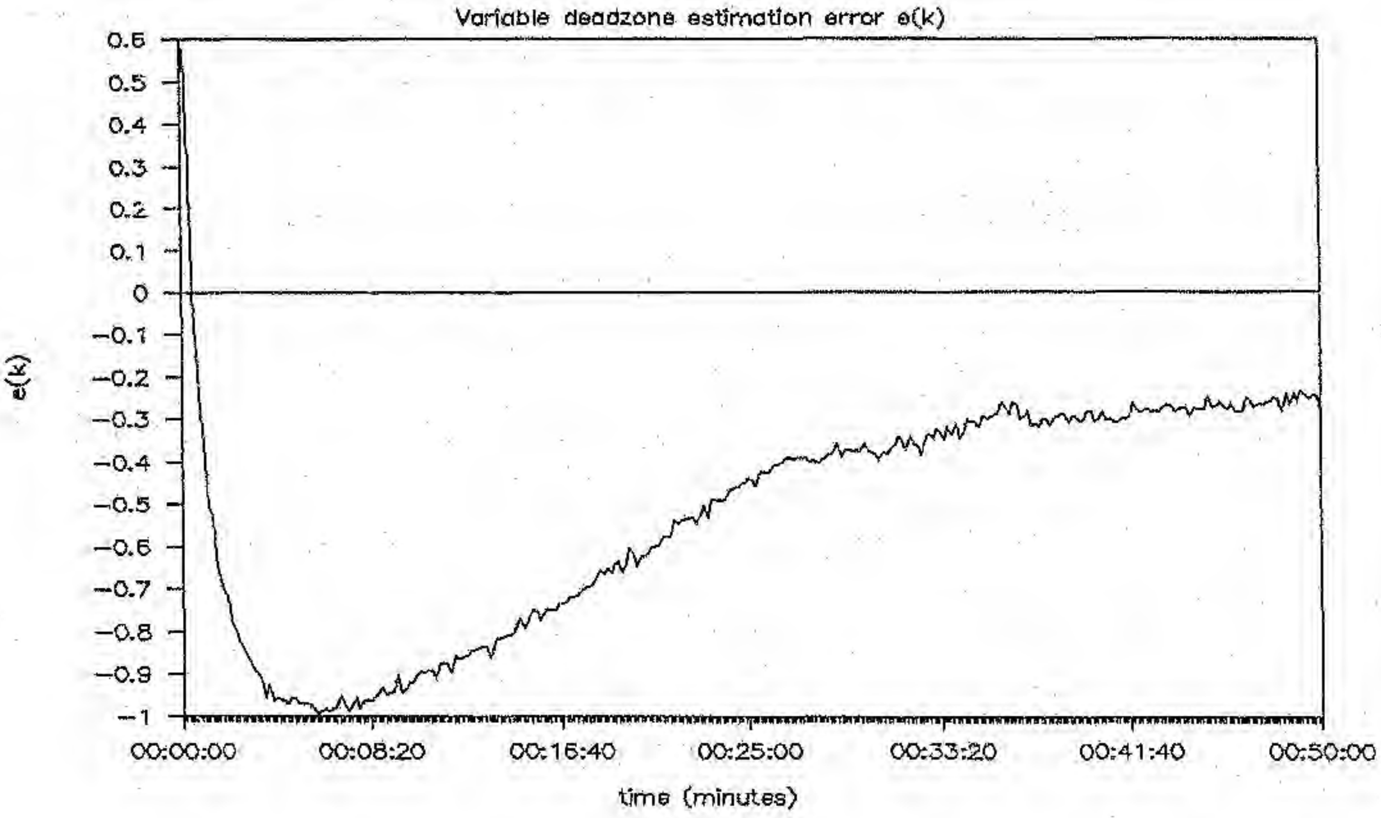


Figure 7.20: Plot of the Parameter Estimation Error

Chapter 8

Conclusion

8.1 General

The report has presented the philosophy and the design methodology of a robust automatic tuning algorithm. This chapter gives a summary of the developments of the previous chapters and discusses some of the most important findings emerging from the research. Also, it lists a number of recommendations that can improve the design and enable its use in a real plant environment.

A review of a number of on-line parameter estimation and control techniques has been presented. The δ-operator and its associated transform have also been discussed. The δ-operator was used to formulate the structure of the estimator and controller algorithms.

The robust estimator has been developed based on the general DARMA Model modified to accommodate random and unmeasurable disturbances. Techniques such as the deadzone, the exponential data weighting, the Bierman U-D covariance factorization have also, been used to enhance the estimator's performance.

The controller was designed based on the closed-loop pole assignment principle. The use of the δ-operator facilitated the use of continuous-time design specifications to be directly applied in establishing the appropriate closed-loop characteristic polynomial.

It is believed that this research is significant in that a number of new concepts have been successfully combined to produce a reliable algorithm. These are the following.

1. Use of the δ-operator and its associated transform to formulate the estimator and controller design.

2. It uses a generalized process model that includes measured and unmeasured disturbances. The estimator is developed based on this model and its robustness improved by including in the normal algorithm the covariance resetting and deadzone options. Use of the U-D factorization technique to improve the numerical performance of the algorithm. It, also, allows the user to vary the sampling time of the algorithm.

3. Use of a low-pass filter and a high-pass filter to improve the signal conditioning of the measured signals and ensure the algorithm only receives information that is vital to its functionality.

4. Use of a closed-loop pole placement method for the design of the controller. It translates the pole placement controller to an industrial PI controller and furnishes K_p and K_i parameters.

5. It uses continuous-time specifications to establish the location of the closed-loop poles.

8.2 Results of the Study

A set of simulations was used to substantiate the design work. The simulation of a first-order plant produced excellent estimation and control design results. Good results were also achieved from the simulation of a second-order plant with the addition of a random disturbance signal indicating the robustness of the design. The system produced good PI control parameters so that the specified closed-loop requirements were met. Reinforced was, also, the concept that under fast sampling conditions the δ-operator model tends to approach the continuous time model.

The estimator was tested under real plant conditions. The results emerging from the test were good. A good agreement was achieved between real plant output and estimated model output indicating that the algorithm can operate under noisy plant conditions. An observation arising from the behavior of the algorithm is that a good prior knowledge of the plant is desirable to avoid large initial estimation errors. Also, the choice of the start-up parameters has a direct effect on the speed of convergence of the estimator and, also, on its robustness. The achieved results are of great significance in that the overall performance of the estimator reinforced the possibility of using the complete system in the plant environment in the future.

8.3 Suggestions for Future Work

Additional work which may yield interesting results can be considered for the following [illegible].

1. Improve the method of establishing the closed-loop poles using continuous-time design criteria. A possibility is to use a knowledge based system which will provide the rules for the design of the control law.

2. Expand the system to design a second order pole placement control law which can be translated into an industrial PID controller. Also, modify the algorithm to accommodate higher order processes.

3. Expand the system to include a feedforward compensator design facility.

4. Perform additional real plant tests and test not only the estimator, but also the controller design.

A large number of chemical factories use computer systems for supervisory and control purposes. The availability of these computer systems enhance the opportunities of using the algorithm in a variety of industrial applications.

One option is to use the algorithm in a self-tuning mode. It could be used to estimate the process model and to generate the required PI and PID control parameters. The calculated parameters could then be downloaded to the PLC which are performing the actual plant control. Otherwise, the downloading of the controller parameters can be initiated by the process operator.

Another option is to use the algorithm in an adaptive controller mode where it monitors the process continuously estimating the process model and adjusting the control law to cater for a new region of operation.

Good results have been achieved by this work,however, additional theoretical and practical research is required to guarantee a good and reliable system performance for any circumstance which may arise in practice.

Appendix A

Bierman's U-D Covariance Factorization

The Bierman's U-D factorization method is adapted to include the deadzone ($b(k)$) and the forgetting factor ($r(k)$).

The original RLS algorithm is:

$$\begin{aligned}
\theta(k) &= \theta(k-1) + \frac{a(k-1)\,P(k-2)\,\phi(k-1)}{\lambda(k-1)+\phi(k-1)^T\,P(k-2)\,\phi(k-1)}.[y(k)-\phi^T(k-1)\,\theta(k-1)] \\
P(k-1) &= \frac{1}{\lambda(k-1)}\Big[P(k-2) - \frac{a(k-1)\,P(k-2)\,\phi(k-1)\,\phi(k-1)^T\,P(k-2)}{\lambda(k-1)+\phi(k-1)^T\,P(k-2)\,\phi(k-1)}\Big]
\end{aligned}$$

The RLS algorithm expressed in Bierman notation is:

$$\begin{aligned}
\hat{x} &= \tilde{x} + k(z - a^T\tilde{x}) \\
\alpha &= r + a^T\tilde{P}a, \quad k = b\tilde{P}a/\alpha \\
\hat{P} &= \frac{1}{r}(\tilde{P} - ka^T\tilde{P})
\end{aligned}$$

with r=forgetting factor and b=relative deadzone.

By factorizing $P(k)$ as,

$$P(k) = U(k)D(k)U^T(k)$$

where $U(k)$, and $D(k)$ are the upper triangular and diagonal terms of the initial RLS, we get:

$$\begin{aligned}
\hat{U}\hat{D}\hat{U}^T &= \frac{1}{r}\Big[\tilde{U}\tilde{D}\tilde{U}^T - \frac{b\tilde{U}vv^T\tilde{U}^T}{\alpha}\Big] \\
\hat{U}\hat{D}\hat{U}^T &= \frac{1}{r}\Big[\tilde{U}[\tilde{D} - b\alpha^{-1}vv^T]\tilde{U}\Big]
\end{aligned} \tag{A.1}$$

with

$$\begin{aligned}
f &= \tilde{U}^Ta = & \tilde{U}(k-2)\phi(k-1) \\
v &= \tilde{D}f = & D(k-2)\,U(k-2)\,\phi(k-1) \\
\alpha &= r + bvf = & r + b\phi^T(k-1)\,P(k-2)\,\phi(k-1)
\end{aligned}$$

with

$$\begin{aligned}
\hat{U} &= U(k-1), \tilde{U} = U(k-2) \\
\hat{D} &= D(k-1), \tilde{D} = D(k-2)
\end{aligned}$$

First assume that $\tilde{D} - b\alpha^{-1}vv^T = \tilde{U}\tilde{D}\tilde{U}^T$. For the assumption to be correct, replacing $\tilde{U}\tilde{D}\tilde{U}^T$ in Equation 1 should produce $\hat{U}\hat{D}\hat{U}^T$. By substitution, we get

$$\frac{1}{r}\left[\bar{U}[\tilde{U}\tilde{D}\tilde{U}^T]\bar{U}^T\right] = \bar{U}\tilde{U}\frac{\tilde{D}}{r}\tilde{U}^T\bar{U}^T$$

which results in

$$\hat{U} = \bar{U}\tilde{U}, \quad \tilde{U}^T\bar{U}^T and \quad \hat{D} = \frac{\tilde{D}}{r}$$

Thus we have that:

$$\tilde{U}\tilde{D}\tilde{U}^T = \tilde{D} - b\alpha^{-1}vv^T [Aggee - Turner Factorization]$$

From the above result, we see that the construction of the update U-D factors depend on the simple factorization

$$\tilde{U}\tilde{D}\tilde{U}^T = \tilde{D} - b\alpha^{-1}vv^T$$

Using the Agee-Turner factorization adjusted so as to coincide with out notation, we have:

$$\begin{aligned}
\tilde{d}_j &= \bar{d}_j + c_j v_j^2 \qquad j = n, \cdots 2, -1 && \text{(A.2)}\\
c_{j-1} &= c_j \bar{d}_j / \tilde{d}_j && \text{(A.3)}\\
\tilde{v}_{ij} &= c_j v_j v_i / \tilde{d}_j \qquad i = 1 \cdots j-1 && \text{(A.4)}
\end{aligned}$$

The above three equations are backward recursive for $j = n$ down to 2. The variable c_n is given by:

$$c_n = -\frac{b}{\alpha} \tag{A.5}$$

where b is the deadzone Rearranging Equation 2, we get:

$$\tilde{d}_j = \bar{d}_j + c_j(\bar{d}_j f_i^2) = \bar{d}_j(1 + c_j\bar{d}_j f_j)^2 \tag{A.6}$$

Rearranging Equation 3, we get:

$$\frac{1}{c_{j-1}} = \frac{1}{c_j}\frac{\tilde{d}_j}{\bar{d}_j} = \frac{1}{c_j} + \bar{d}_j f_j^2 \tag{A.7}$$

But since

$$\frac{1}{c_n} = -\frac{\alpha}{b} = -(r + \sum_{i=1}^{n} d_i f i^2)$$

then follows that:

$$\frac{1}{c_j} = -\frac{\alpha_j}{b} \tag{A.8}$$

By substituting Equations 7 and 8 into 6, we get:

$$\begin{aligned}
\tilde{d}_j &= \bar{d}_j(1 + c_j\bar{d}_j f_i^2)\\
\tilde{d}_j &= \bar{d}_j c_j(\frac{1}{c_j} + \bar{d}_j f_i^2)\\
\tilde{d}_j &= \bar{d}_j c_j(\frac{1}{c_{j-1}})
\end{aligned}$$

which combined with

$$r\hat{d}_j = \bar{d}_j$$

gives

$$r\hat{d}_j = \bar{d}_j \frac{c_j}{c_{j-1}}$$

and inserting

$$\frac{1}{c_j} = -\frac{\alpha_j}{b}$$

in the above equation gives the diagonal term D

$$\hat{d}_j = \frac{1}{r}[\bar{d}_j \frac{\alpha_{j-1}}{\alpha_j}] \quad (A.8)$$

Looking into the construction of the upper triangular vector U, we saw that:

$$\hat{U} = \bar{U}\tilde{U}$$

Also vector U can be expressed as:

$$\tilde{U} = I + [0, \lambda_2 v^{(1)}, \lambda_3 v^{(2)} \cdots \lambda_n v^{(n-1)}]$$

and combined with

$$\begin{aligned} \lambda_j &= \frac{v_j c_j}{d_j} \\ \lambda_j &= \frac{v_j c_j}{\bar{d}_j r} \\ \frac{1}{c_j} &= -\frac{\alpha_j}{b} \end{aligned}$$

we get λ as

$$\begin{aligned} \lambda_j &= \frac{-v_j b}{\alpha_j r \hat{d}_j} \\ &= \frac{-v_j b}{\alpha_j r \bar{d}_j \frac{\alpha_{j-1}}{\alpha_j r}} \\ \lambda_j &= -\frac{b v_j}{\bar{d}_j \alpha_{j-1}} \\ &= \frac{b f_j}{\alpha_{j-1}} \end{aligned}$$

From

$$K = \frac{\bar{U}\bar{D}\bar{U}^T a}{\alpha}$$

and

$$v = \tilde{D}f = \tilde{D}\tilde{U}^T\alpha$$

we get

$$K = \frac{\tilde{U}v}{\alpha}$$

The Kalman gain is:

$$K = \frac{k_{n+1}}{\alpha_n}$$

Now for $n = 1$, we get:

$$\hat{d}_1 = \frac{\tilde{d}_1 r}{\alpha_1}$$

Using the above proof the Bierman factorization consists of the following steps.

a. For $n = 1$

$$\begin{aligned} \alpha_1 &= r + v_1 f_1 \\ K_2 &= (v_1, 0 \cdots, 0) \\ \hat{d}_1 &= \tilde{d}_1 r/\alpha_1 \end{aligned}$$

b. For $n > 1, j = 2 \cdots n$

$$\begin{aligned} \alpha_j &= \alpha_{j-1} + v_j f_j \\ \hat{d}_j &= \tilde{d}_j\, \alpha_{j-1}/a_j \\ \lambda_j &= -f_j/\alpha_{j-1} \\ \hat{u}_j &= \tilde{u}_j + \lambda_j k_j \\ K_{j+1} &= k_j + v_j V_j \\ K &= K_{n+1}/\alpha_n \end{aligned}$$

Appendix B

Alternative Method of Expressing the PI Controller in δ-Operator Format

The PI controller is given by:

$$u(t) = K_p[e(t) + \frac{1}{Ti}\int e(t)dt]$$

If

$$u_1(t) = \frac{1}{Ti}\int e(t) \qquad \text{then}$$

$$\frac{uu_1(t)}{dt} = \frac{1}{Ti}e(t)$$

and using Euler's approximation, we get:

$$\frac{du_1(t)}{dt} \approx \frac{u(k+1) - u(k)}{\Delta} = \frac{q-1}{\Delta}u(k)$$

where q is the forward shift-operator.

The analog PI controller can be approximated by

$$u(k) = K_p[e(k) + k_i\frac{\Delta e(k)}{q-1}]$$

where

$$k_i = \frac{1}{Ti}$$

The transfer function of the discrete PI controller is

$$G_{PI}(q) = K_p + \frac{K_pK_i\Delta}{q-1} = \frac{Kp(q-1) + KpK_i\Delta}{q-2}$$

Using $q = \delta\Delta + 1$ results in:

$$G_{PI}(\delta) = \frac{K_p\delta + K_pK_i}{\delta}$$

which is the same result achieved by directly substituting $s = \delta$.

Appendix C

Formulation of the Pole Placement Controller as a PID Controller

Assuming a second order plant described by

$$\delta^2 y(k) + a_1 \delta y(k) + a_0 y(k) = b_1 \delta u(k) + b_0 u(k)$$

and

- degree A = 2
- degree B = 1
- degree S = 0
- degree D = 1
- $degreeL = degreeA - 1 = 1$
- $degreeP = degreeA + degreeD + degreeS - 1 = 2$

and degree $A^* = 2.degreeA + degreeD + degreeS - 1 = 4$.

The pole placement algorithm produces the control law.

$$G_c(\delta) = \frac{P_2\delta^2 + P_1\delta + P_0}{(l_1\delta + l_0)\delta}$$

The analog PID controller is given by

$$\begin{aligned} G(s) &= K_p(1 + \frac{1}{T_i S} + \frac{T_d s}{1 + \frac{T_d}{N} S}) \\ &= K_p(1 + \frac{1}{T_i S} + \frac{T_d N s}{N + T_d s}) \\ &= K_p(1 + \frac{1}{T_i S} + \frac{N s}{S + \frac{N}{T_d}}) \\ &= K_p(1 + \frac{K_i}{s} + \frac{N s}{S + K_d}) \end{aligned}$$

where :

$$\begin{aligned} K_i &= \frac{1}{T_i} \\ K_d &= \frac{N}{T_d} \end{aligned}$$

and $\frac{N}{T_d}$ is the derivative filter.

The transfer function can be written as:

$$\begin{aligned} G(s) &= \frac{K_p(s(s+K_d)+K_i(s+K_d)+Ns^2)}{s(s+K_d)} \\ &= K_p(s^2+K_d s+K_i s+K_i K_d+Ns^2) \\ &= \frac{K_p(N+1)s^2+K_p(K_d+K_i)s+K_i K_d K_p}{s(s+K_d)} \end{aligned}$$

By approximating, $s=\delta$ we get

$$\frac{K_p(N+1)s^2+K_p(K_d+K_i)s+K_i K_d K_p}{s(s+K_d)} = \frac{p_2\delta^2+p_1\delta+p_0}{\delta(l_1\delta+l_0)}$$

Appendix D

The Computer Program

The algorithm was implemented using Microsoft Fortran. The complete program is listed below.

```
C
C                      THE HEADER FOR THE AUTOTUNING PROGRAM
C
C
         WRITE(*,015)
 015     FORMAT(20X,'PRESS <RETURN> TO CONTINUE')
         READ(*,001)LA
 001     FORMAT(20X,I2)
         WRITE(*,002)
 002     FORMAT(20X,'THE MODEL REFERENCE δ TYPE CONTROLLER')
         WRITE(*,003)
 003     FORMAT(20X,'--------------------------------------')
         WRITE(*,004)
 004     FORMAT(20X,'THE SYSTEM READS THE PLANT I/O AND THEN')
         WRITE(*,005)
 005     FORMAT(20X,'GENERATES A PLANT MODEL. THE MODEL')
         WRITE(*,006)
 006     FORMAT(20X,'PARAMETERS ARE USED TO PRODUCE THE')
         WRITE(*,007)
 007     FORMAT(20X,'PI & PID CONTROLLER PARAMETERS')
         WRITE(*,003)
         WRITE(*,013)
 013     FORMAT(20X,' ')
         WRITE(*,008)
 008     FORMAT(20X,'NOTE:  YOU MUST SATISFY THE FOLLOWING RULES')
         WRITE(*,009)
 009     FORMAT(20X,'            a. ws ≥ 2wb             ')
         WRITE(*,010)
 010     FORMAT(20X,'            b. wc ≤ 1/2ws              ')
         WRITE(*,013)
         WRITE(*,013)
         WRITE(*,015)
         READ(*,001)IA
         END
```

```
C          AUTOTUNER FOR PI CONTROLLERS
C             USING THE DELTA OPERATOR
C
C             COMMENTS
C
C DELTA      = SAMPLING INTERVAL
C EPSHP      = OUTPUT STATE OF HPF FOR CONSTANT DISTURBANCE
C CFACTOR    = INITILIZATION VALUE FOR THE U-D DIAGONAL TERM
C WC         = CUT-OFF FREQUENCY OF THE SECOND ORDER FILTER
C Z          = DAMPING FACTOR FOR THE SECOND ORDER FILTER
C E1         = FIRST PARAMETER OF THE SECOND ORDER FILTER
C E0         = SECOND PARAMETER OF THE SECOND ORDER FILTER
C GAM2       = FIRST DENOMINATOR PARAMETER OF THE SECOND ORDER FILTER
C GAM1       = SECOND DENOMINATOR PARAMETER OF THE SECOND ORDER FILTER
C GAM0       = THIRD DENOMINATOR PARAMETER OF THE SECOND ORDER FILTER
C ERRLIM     = FIXED DEADZONE PARAMETER
C SIGMA      = ADJUSTABLE DEADZONE PARAMETER
C EPS0       = ADJUSTABLE DEADZONE PARAMETER
C EPS1       = ADJUSTABLE DEADZONE PARAMETER
C EPS2       = ADJUSTABLE DEADZONE PARAMETER
C NPAR       = NUMBER OF PARAMETERS
C NUVEC      = DIMENSION OF THE U VECTOR
C ALPHA      = ADJUSTABLE DEADZONE PARAMETER
C BETA       = ADJUSTABLE DEADZONE PARAMETER
C SIGMA1     = ADJUSTABLE DEADZONE PARAMETER
C DZLIMIT    = ERROR LIMIT FOR DEADZONE
C AZONE      = THE DEADZONE
C ALPHA      = THE VARIABLE DEADZONE GAIN LIMIT
C BETA       = THE VARIABLE DEADZONE SCALING FACTOR
C PHI        = THE DATA REGRESSION VECTOR
C THETA      = THE PARAMETER VECTOR
C K          = THE GAIN VECTOR
C ALPHAJ     = BIERMAN U-D VARIABLE
C YHAT       = THE PREDICTED OUTPUT VALUE
C YH         = THE PLANT OUTPUT AFTER THE HPF
C UH         = THE PLANT INPUT AFTER THE HPF
C Y          = THE PLANT OUTPUT AFTER THE LPF
C U          = THE PLANT INPUT AFTER THE LPF
C E          = THE PREDICTED ERROR
C LAMDA      = THE EXPONENTIAL FORGETTING FACTOR
C LAMDA0     = ADJUSTABLE FORGETTING FACTOR CONSTANT
C YSPAN      = THE RANGE OF THE PLANT OUTPUT SIGNAL
C USPAN      = THE RANGE OF THE PLANT INPUT SIGNAL
C END        = EOF
C
C
C+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C
      INTEGER N,FINI,HPF,LPF,AZONEFLAG,LAMDAFLAG,NN
      INTEGER DZFLAG,INTERATION
      REAL LAMDA0,ERRLIM,E,BMZ,MZ0,MZ
C
      REAL LPFF(4),LPFBAR(4),MU1,MU2
C
      REAL FI(4),THETA(4),K(4),DIAG(4),OFFDIAG(6),LAMDA,ALPHAJ,AZONE
      COMMON/AREA1/FI,THETA,K,DIAG,OFFDIAG,LAMDA,ALPHAJ,AZONE
C
      PARAMETER(SIGMA=0.98,EPS0=2.5E-3,EPS1=1.0E-3,EPS2=0.5E-3)
      PARAMETER(ALPHA=0.4,BETA=1.3,CFACTOR=100.0,PI=3.141592654)
C
C
C
      WRITE(*,001)
 001  FORMAT(21X,'THE OPERATOR IS TO DEFINE THE FOLLOWING VARIABLES')
```

```
      WRITE(*,002)
 002  FORMAT(21X,'=================================================')
      PRINT*,' '
      PRINT*,' '
      PRINT*,' '
      WRITE(*,003)
 003  FORMAT(21X,'PLEASE DEFINE NUMBER OF PARAMETERS:a)2,b)4')
      READ(*,004)N
 004  FORMAT(I2)
      WRITE(*,005)
 005  FORMAT(21X,'SET HPF FLAG:FLAG=1(HPF IS ON),FLAG=0(HPF IS OFF)')
      READ(*,006)HPF
 006  FORMAT(I1)
      WRITE(*,007)
 007  FORMAT(21X,'SET LPF FLAG:FLAG=1(LPF IS ON),FLAG=0(LPF IS OFF)')
      READ(*,008)LPF
 008  FORMAT(I1)
      WRITE(*,009)
 009  FORMAT(21X,'SET LAMDAFLAG:FLAG=1(CON.FORFAC)FLAG=0(VAR.FORFAC)')
      READ(*,010)LAMDAFLAG
 010  FORMAT(I1)
      WRITE(*,011)
 011  FORMAT(21X,'SET AZONEFLAG:FLAG=1(CON.DZONE),FLAG=0(VAR.DZONE)')
      READ(*,012)AZONEFLAG
 012  FORMAT(I1)
      WRITE(*,070)
 070  FORMAT(21X,'SET THE OUTPUT ZERO-BASE AND SPAN')
      READ(*,071)YZERO,YSPAN
 071  FORMAT(F6.3,F6.3)
      WRITE(*,072)
 072  FORMAT(21X,'SET THE INPUT ZERO-BASE AND SPAN')
      READ(*,073)UZERO,USPAN
 073  FORMAT(F6.3,F6.3)
      WRITE(*,074)
 074  FORMAT(21X,'SET THE SAMPLING RATE')
      READ(*,075)DELTA
 075  FORMAT(F6.3)
 688  WRITE(*,076)
 076  FORMAT(21X,'SET THE Wc CUTOFF OF THE LPF')
      READ(*,077)WC
 077  FORMAT(F7.4)
      WRITE(*,078)
 078  FORMAT(21X,'SET THE W1m RESTRICTION FREQUENCY')
      READ(*,077)WL
C
            PRINT*,WC,WL
C
C_______________________________________________________________________
C|                                                                     |
C|                           PART ONE:                                 |
C|                          INITIALIZER                                |
C|                                                                     |
C|                                                                     |
C|                                                                     |
C|_____________________________________________________________________|
C
C
C
      NN=(N*[illegible]))/2
      TIME=0.0
      INTERATION=0
      ALPHAJ=1.0
      IF(N.EQ.2)THEN
        A0=0.0
        B0=0.0
        YHAT=0.0
      ELSEIF(N.EQ.4)THEN
        A0=0.0
        A1=0.0
        B0=0.0
        B1=0.0
        YHAT=0.0
      ELSE
        PAUSE'NUMBER OF PARAMETERS MUST BE 2 OR 4'
        STOP
      ENDIF
C
C     INITIALIZATION OF THE RLS VARIABLES
C
      DO 14 I=1,N
             FI(I)=0.0
             K(I)=0.0
             THETA(I)=0.0
             DIAG(I)=CFACTOR
 14   CONTINUE
      DO 15 I=1,NN
           OFFDIAG(I)=0.0
 15   CONTINUE
C
C
      IF(HPF.EQ.1)THEN
        EPSH=0.0008
        XHPY=0.
        XHPU=0.
      ENDIF
C
C INITIALIZATION OF THE LPF VARIABLE
C
      WS=2.0*PI*(1.0/DELTA)
           PRINT*,WS
      IF(WC.GT.0.5*WS)THEN
        WRITE(*,080)
 80     FORMAT(21X,'THE WC IS GREATER THAN 1/2 WS RE-DEFINE WC')
        GO TO 688
      ENDIF
      IF(WC.GT.WL)THEN
        WRITE(*,81)
 81     FORMAT(21X,'THE WC IS GREATER THAN WL RE-DEFINE WC')
        GO TO 688
      ENDIF
      ALPF=WC/SQRT(2.0)
      MU1=(WC**2)*(DELTA**2)
      MU2=ALPF*4.0*DELTA
      E2=4.0+MU2+MU1
      E1=((2.0*MU1)-8.0)/E2
      E0=(4.0-MU2+MU1)/E2
      GAM2=MU1/E2
      GAM1=(2.0*MU1)/E2
      GAM0=MU1/E2
C


      DO 16 I=1,4
        LPFF(I)=0.0
        LPFBAR(I)=0.0
 15   CONTINUE
C
      IF(LAMDAFLAG.EQ.1)THEN
        LAMDA=0.970
      ELSEIF(LAMDAFLAG.EQ.0)THEN
        LAMDA=0.950
```

```
          LAMDA0=0.985
        ELSE
          PAUSE'LAMDA FLAG MUST BE EITHER 1 OR 0'
          STOP
        ENDIF
C
        IF(AZONEFLAG.EQ.1)THEN
          ERRLIM=0.003
          AZONE=1.0
        ELSEIF(AZONEFLAG.EQ.0)THEN
          SIGMA1=1.0-SIGMA
          MZ0=EPS0
          AZONE=1.0
        ELSE
          PAUSE'AZONE FLAG MUST BE EITHER 1 OR 0'
          STOP
        ENDIF
C
        OPEN(UNIT=1,FILE='PLANTY')
        OPEN(UNIT=2,FILE='PLANTU')
        OPEN(UNIT=3,FILE='PAR.DAT',STATUS='NEW')
        OPEN(UNIT=4,FILE='IO.DAT',STATUS='NEW')
        OPEN(UNIT=7,FILE='PHI.DAT',STATUS='NEW')
        OPEN(UNIT=8,FILE='THETA.DAT',STATUS='NEW')
C
C
C_______________________________________________________________________
C
C
C                           PART TWO:
C                 ESTIMATOR AND CONROLLER DESIGN
C
C
C
C_______________________________________________________________________
C
C
C
 500    INTERATION=INTERATION+1
        TIME=DELTA*INTERATION
C
        CALL GETOUT(YOUT,FINI)
        YXX=(YOUT-YZERO)/YSPAN
        CALL GETIN(UIN)
        UXX=(UIN-UZERO)/USPAN
C
        IF(HPF.EQ.1)THEN
          YH=YXX-EPSH*XHPY
          XHPY=XHPY+DELTA*YH
          UH=UXX-EPSH*XHPU
          XHPU=XHPU+DELTA*UH
        ELSEIF(HPF.EQ.0)THEN
          YH=YXX
          UH=UXX
        ELSE
          PAUSE'HPF FLAG MUST BE EITHER 1 OR 0'
          STOP
        ENDIF
C
        IF(LPF.EQ.1)THEN
          QY=GAM2*YH+GAM1*LPFBAR(1)+GAM0*LPFBAR(2)
          Y=-E1*LPFF(1)-E0*LPFF(2)+QY
          LPFF(2)=LPFF(1)
          LPFF(1)=Y
          LPFBAR(2)=LPFBAR(1)
          LPFBAR(1)=YH
          QU=GAM2*UH+GAM1*LPFBAR(3)+GAM0*LPFBAR(4)
          U=-E1*LPFF(3)-E0*LPFF(4)+QU
          LPFF(4)=LPFF(3)
          LPFF(3)=U
          LPFBAR(4)=LPFBAR(3)
          LPFBAR(3)=UH
        ELSEIF(LPF.EQ.0)THEN
          Y=YH
          U=UH
        ELSE
          PAUSE'HPF FLAG MUST BE EITHER 1 OR 0'
          STOP
        ENDIF
C
C
        IF(LAMDAFLAG.EQ.0)THEN
          LAMDA=LAMDA*LAMDA0+(1.0-LAMDA0)
        ENDIF
C
        IF(AZONEFLAG.EQ.0)THEN
          MZ=MZ0
          BMZ=BETA*MZ
          MZ0=SIGMA*MZ+SIGMA1*(E1.0+EPS1*ABS(UH)+EPS2*ABS(YXX))
        ENDIF
C
C       THE GENERATION OF THE PREDICTED ERROR
C
        YHAT=0.0
        DO 20 I=1,N
           YHAT=YHAT+(THETA(I)*FI(I))
 20     CONTINUE
C
        E=Y-YHAT
C
        IF(AZONEFLAG.EQ.1)THEN
          IF(ABS(E).GT.ERRLIM)THEN
            DZFLAG=1
            AZONE=1.0
          ELSE
            DZFLAG=0
            AZONE=0.0
          ENDIF
        ENDIF
C
        IF(AZONEFLAG.EQ.0)THEN
          IF(ABS(E).GT.BMZ)THEN
            DZFLAG=1
            AZONE=(ALPHA/E)*DEAD(-BMZ,BMZ,E)
          ELSE
            DZFLAG=0
          ENDIF
        ENDIF
C
        CALL LS(Y,U,N,DELTA,DZFLAG,E,A0,A1,B0,B1)
C
        WRITE(*,100)A1,A0,B1,B0
 100    FORMAT(1X,F10.6,1X,F10.6,1X,F10.6,1X,F10.6)
        WRITE(3,200)A1,A0,B1,B0,YH,UH
 200    FORMAT(F10.6,1X,F10.6,1X,F10.6,1X,F10.6,1X,F10.6,1X,F10.6)
        WRITE(4,200)Y,YHAT,E,K(1),K(2),K(3)
        WRITE(7,200)FI(1),FI(2),FI(3),FI(4),AZONE,BMZ
        WRITE(8,200)THETA(1),THETA(2),THETA(3),THETA(4)
 210    FORMAT(1X,F13.10,1X,F13.10,1X,F13.10,1X,F13.10)
C
        IF(FINI.EQ.0 .AND. TIME.LT.24000.0)THEN
            GO TO 500
        GO TO 300
      ENDIF
      CLOSE(3,STATUS='KEEP')
      CLOSE(4,STATUS='KEEP')
      CLOSE(7,STATUS='KEEP')
      CLOSE(8,STATUS='KEEP')
;
;
; GENERATION OF THE CONTROLLER PARAMETERS
;
      CALL CONTROL(A0,A1,B0,B1,N,WC)
;
;
;
      STOP
      END
;
;
;
; FUNCTION DEAD FOR VARIABLE DEADZONE VALUES
;
      FUNCTION DEAD(BMZL,BMZH,ERROR)
      REAL BMZL,BMZH,ERROR
      IF(ERROR.LE.BMZL)THEN
       DEAD=ERROR-BMZL
      ELSEIF(ERROR.GT.BMZH)THEN
       DEAD=ERROR-BMZH
      ELSE
       DEAD=0.0
      ENDIF
      RETURN
      END
;
;
;
      SUBROUTINE GETOUT(YOUT,FIN)
      REAL YOUT
      INTEGER FIN
      READ(1,400)YOUT
 400  FORMAT(F15.10)
      IF(.NOT. EOF(1))THEN
           FIN=0
      ELSE
           FIN=1
      ENDIF
      RETURN
      END
;
;
      SUBROUTINE GETIN(UIN)
      REAL UIN
      READ(2,400)UIN
 400  FORMAT(F15.10)
      RETURN
      END
;
;
;
      SUBROUTINE LS(Y,U,N,DELTA,DZFLAG,PERR,A0,A1,B0,B1)
      INTEGER DZFLAG,N
      REAL PERR,Y,U,A1,A0,B1,B0,DELTA
;
      REAL FI(4),THETA(4),K(4),DIAG(4),OFFDIAG(6),LAMDA,ALPHAJ,AZONE
      COMMON/AREA1/FI,THETA,K,DIAG,OFFDIAG,LAMDA,ALPHAJ,AZONE
      IF(DZFLAG.EQ.1)THEN
        FJ=FI(1)
        VJ=DIAG(1)*FJ
        K(1)=VJ
        ALPHAJ=1.0+VJ*FJ
        DIAG(1)=DIAG(1)/ALPHAJ/LAMDA
        IF(N.GT.1)THEN
          KF=0
          KU=0
          DO 20 J=2,N
            FJ=FI(J)
            DO 30 I=1,J-1
              KF=KF+1
              FJ=FJ+FI(I)*OFFDIAG(KF)
30          CONTINUE
            VJ=FJ*DIAG(J)
            K(J)=VJ
            AJLAST=ALPHAJ
            ALPHAJ=AJLAST+VJ*FJ
            DIAG(J)=DIAG(J)*AJLAST/ALPHAJ/LAMDA
            PJ=-(FJ*AZONE)/AJLAST
            DO 40 I=1,J-1
              KU=KU+1
              W=OFFDIAG(KU)+K(I)*PJ
              K(I)=K(I)+OFFDIAG(KU)*VJ
              OFFDIAG(KU)=W
40          CONTINUE
20          CONTINUE
        ENDIF
        ALPHAJ=ALPHAJ/AZONE
      ENDIF
      IF(DZFLAG.EQ.
        ALPHAJ=1.0
        DO 50 I=1,N
           K(I)=0.0
50      CONTINUE
      ENDIF
;
;
      DO 60 I=1,N
         THETA(I)=THETA(I)+PERR*K(I)/ALPHAJ
60    CONTINUE
      IF(N.EQ.2)THEN
        A0=-THETA(1)
        B0=THETA(2)
      ENDIF
      IF(N.EQ.4)THEN
        A1=-THETA(1)
        A0=-THETA(2)
        B1=THETA(3)
        B0=THETA(4)
      ENDIF
;
      IF(N.EQ.2)THEN
        FI(1)=FI(1)+DELTA*Y
      [illegible]
      [illegible]

        FI(2)=FI(2)+DELTA*U
      ENDIF
      IF(N.EQ.4)THEN
        FI(4)=FI(4)+DELTA*FI(3)
        FI(2)=FI(2)+DELTA*FI(1)
        FI(1)=FI(1)+DELTA*Y
        FI(3)=FI(3)+DELTA*U
      ENDIF
      RETURN
      END
C
C
C
C
      SUBROUTINE CONTROL(A0,A1,B0,B1,N,WC)
      REAL MPW,KP,KI
      PARAMETER(PI=3.141592654)
C
C
 100  WRITE(*,001)
 001  FORMAT(21X,'---------------------------------------------')
      PRINT *,' '
      PRINT *,' '
      WRITE(*,002)
 002  FORMAT(21X,'THE ESTIMATED PLANT PARAMETERS ARE:')
      PRINT *,' '
      IF(N.EQ.2)THEN
        WRITE(*,003)A0
 003    FORMAT(42X,'a0=',F10.6)
        WRITE(*,004)B0
 004    FORMAT(42Y,'b0=',F10.6)
      ENDIF
      IF(N.EQ.4)THEN
        WRITE(*,005)A0
 005    FORMAT(42X,'a0=',F10.6)
        WRITE(*,005)A1
        WRITE(*,005)B0
        WRITE(*,005)B1
      ENDIF
      WRITE(*,799)
 799  FORMAT(21X,'OK=1 ELSE 0')
      READ(*,800)KK
 800  FORMAT(I1)
      IF(KK.EQ.0)THEN
        READ(*,801)A0,A1,B0,B1
 801  FORMAT(F10.6,F10.6,F10.6,F10.6)
      PRINT*,A0,B0
      ENDIF
      PRINT *,' '
      PRINT*,' '
      WRITE(*,007)
 007  FORMAT(21X,'PRESS ANY KEY TO CONTINUE')
      READ(*,600)LA
 600  FORMAT(21X,I2)
      PRINT*,' '
      PRINT*,' '
      WRITE(*,700)
 700  FORMAT(21X,'THE CONTROL PARAMETERS MUST BE IN REAL NUMBERS')
      PRINT*,' '
      PRINT*,' '
      WRITE(*,008)
 008  FORMAT(21X,'DEFINE THE DESIRED PERCENT OVERSHOOT')
      READ(*,009)PO
 009  FORMAT(F6.3)
      WRITE(*,010)
 010  FORMAT(21X,'DEFINE THE DESIRED SETTLING TIME')
      READ(*,011)TS
 011  FORMAT(F6.3)
      PRINT*,' '
      PRINT*,' '
C
C
C CALCULATION OF THE DAMPING FACTOR
C
      B=((LOG(PO/100.0)*(-1.0))/PI)**2
[illegible]
```

```
      Z=SQRT(B/(B+1.0))
C
C
C CALCULATION OF THE PHASE MARGIN
C
      PM=(Z/0.01)
C
      IF(PM.LT.45.0 .OR. PM.GT.80.0)THEN
           WRITE(*,012)
 012       FORMAT(21X,'WHAT IS THE DESIRED PHASE MARGIN?')
           READ(*,013)PM
 013       FORMAT(F6.3)
           Z=PM*0.01
           PO=100.*EXP((-Z*PI)/SQRT(1.0-Z**2))
           WRITE(*,014)
 014       FORMAT(21X,'IS THE PERCENT OVERSHOOT ACCEPTABLE?')
           WRITE(*,601)PO
 601       FORMAT(21X,'PERCENT OVERSHOOT=',F6.3)
           WRITE(*,015)
 015       FORMAT(21X,'IF YES PRESS -1-, ELSE PRESS -0-')
           READ(*,016)NN
 016       FORMAT(I2)
           IF(NN.EQ.0)THEN
              WRITE(*,017)
 017          FORMAT(21X,'RE-DEFINE THE PERCENT OVERSHOOT')
              GO TO 100
           ENDIF
      ENDIF
C
C
C CALCULATION OF THE NATURAL FREQUENCY
C
      WN=4.0/(TS*Z)
C
C CALCULATION OF THE PEAK AMPLITUTE
C
      MPW=1.0/(2.0*Z*(SQRT(1.0-Z**2)))
C
C CALCULATION OF THE RESONANT FREQUENCY
C
      WR=WN*SQRT(1.0-(2.0*Z**2))
C
C CALCULATION OF THE CUTOFF FREQUENCY (3db)
C
      W3DB=WN*(SQRT((Z**2)+1.0)+Z)
C
C
      IF(W3DB.GT.WC)THEN
         WRITE(*,018)
 018     FORMAT(19X,'THE 3db Wc OF THE SYSTEM IS GREATER THAN FILTER
     Z                   CUTOFF FREQUENCY RE-DEFINE  THE SETTLING TIME')
         GO TO 100
      ENDIF
      PRINT*,' '
      PRINT*,' '
      WRITE(*,019)PO,TS,Z,PM
 019  FORMAT(1X,'PO=',F8.4,1X,'TS=',F3.1,1X,'Z=',F8.4,1X,'PM=',F8.4)
      WRITE(*,006)MPW,WN,WR,W3DB
 006 FORMAT(1X,'MPW=',F4.1,1X,'WN=',F8.4,1X,'WR=',F8.4,1X,'W3DB=',F8.4)
      PRINT*,' '
      PRINT*,' '
      PRINT*,' '
C
C
C CONSTRACTION OF THE ASTAR VECTOR
C
```

```
          IF(N.EQ.2)THEN
            A1STAR=2.0*Z*WN
            A0STAR=WN**2
C
            P1=(A1STAR-A0)/B0
            P0=A0STAR/B0
C
            KP=P1
            KI=P0/KP
C
            WRITE(*,020)KP,KI
 020        FORMAT(1X,'GAIN Kp=',F8.4,10X,'INTEGRAL GAIN Ki=',F8.4,)
          ENDIF
C
          RETURN
          END
```

[illegible]

References

[1] J. G. Ziegler and N. B. Nichols. Optimum settings for automatic controllers. *ASME Transactions*, 64:759–768, November 1942.

[2] J. G. Ziegler and N. B. Nichols. Process lags in automatic control circuits. *ASME Transactions*, 433–444, July 1943.

[3] F. G. Shinskey. *Process Control Systems*. McGraw-Hill, New York, second edition, 1979.

[4] G. H. Cohen and G. A. Coon. Theoretical consiaerations in retarded control. *Taylor Instruments Co.*, Bull. No. TDS-10A102.

[5] P. Hazebroek and B. L. Van Der Waerden. Theoretical considerations on the optimum adjustment of regulators. *ASME Transactions*, 309–315, April 1950.

[6] R. E. Kalman. Design of a self-optimizing system. *ASME Transactions*, 468–478, February 1958.

[7] K. J. Astrom. Theory and applications of adaptive control. *Automatica*, 471–486, September 1^{\circ}33.

[8] V. Peterka. Predictor-based self-tuning control. *Automatica*, 39–50, January 1984.

[9] D. W. Clarke and P. W. Gawthrop. Self-tuning control. *Proc. IEE*, 633–640, June 1979.

[10] U. Borison. Self-tuning regulators for a class of multivariable systems. *Automatica*, 209–215, March 1979.

[11] P. C. Young. Parameter estimation for continuous-time models-a survey. *Automatica*, 17,231, 1981.

[12] G. C. Goodwin and K. S. Sin. *Adaptive Filtering Prediction and Control*. Prentice-Hall Inc., 1984.

[13] G. C. Goodwin and K. S. Sin. *Adaptive Filtering Prediction and Control*. Prentice-Hall Inc., 1984.

[14] T. F. Edgar D. E. Seborg and S. L. Shah. Adaptive control strategies for process control: a survey. *AIChE*, 32, No. 6:881–913, 1986.

[15] R. Iserman. Parameter adaptive control algorithms-a tutorial. *Automatica*, 18:513–528, September 1982.

[16] L. Ljung T. Soderstrom and I. Gustavsson. A comparative study of recursive identification methods. *Dept. Of Automatic Control, Lund Institute of Technology*, Report 7427:, 1976.

[17] I. M. MacLeod. State space modeling and control. *Course Notes*, 1987.

[18] G. C. Goodwin D. Q. Maynes and R. H. Middleton. Rapprochement between continuous and discrete model reference adaptive control. *Automatica*, 22 No2:199–207, 1986.

[19] R. H. Middleton and G. C. Goodwin. Improved finite word length characteristics in digital control using delta operators. *IEEE Transactions on Automatic Control*, AC-31 No11:1015–1021, 1986.

[20] I. M. MacLeod. Delta operator. *Course Notes*, 1990.

[21] K. J. Astrom and B. Wittenmark. Self-tuning controllers based on pole-zero placement. *Proc. IEE*, 127:120–130, 1980.

[22] I. M. MacLeod. Msc thesis lectures. *Notes*, 1990.

[23] G. J. Bierman. *Factorization Methods for Discrete Sequential Lstimation*. Academic Press, 1977.

[24] G. C. Goodwin and K. S. Sin. *Adaptive Filtering Prediction and Control*. Prentice-Hall Inc., 1984.

[25] G. Franklin and J. Powell. *Digital Control of Dynamic Systems*. Addison - Wesley, 1981.

[26] G. C. Goodwin and K. S. Sin. *Adaptive Filtering Prediction and Control*. Prentice-Hall Inc., 1984.

[27] M. L. Bergensen. The implementation of a generalized robust adaptive controller. *MSc Thesis , Wits University, Witwatersrand, South Africa*, 1988.

[28] K. J. Astrom U. Borisson L. Ljung and B. Wittenmark. Theory and applications of self-tuning regulators. *Automatica*, 13:453, 1977.

[29] B. Egardt. Stability of adaptive controllers. *Lecture Notes in Control and Information Sciences*, 26, 1977.

[30] T. Hagglund. New estimation techniques for adaptive control. *PhD Thesis, Report TFRT-1025,Lund Institude of Technology,Sweden*, 1984.

[31] R. C. Dorf. *Modern Control System*. Addison-Wesley Publishing Company, 1980.