REDUCTION OF THE PARAMETER ESTIMATION TIME FOR AN ADAPTIVE CONTROL SYSTEM

A Thesis submitted for examination for

the Degree of Master of Engineering (Electrical)

Dept. of Electrical & Electronic Engineering

Faculty of Engineering

Victoria University of Technology (Footscray Campus)

Victoria

Australia

1994

By

Robert V.Ives

B.Sc.(Hons)

University of Nottingham

Nottingham

England

SYNOPSIS

The following work is concerned with the use of the Method of Least Squares in the parameter estimation of a discrete-time model of a system. In particular, the emphasis is upon both the initial convergence and accuracy of the estimates. The investigation is therefore pertinent to both the "cold-starting" of least squares estimators, and to systems in which "jump" changes in parameters occur, requiring resetting of the estimator.

The work was approached from an engineering viewpoint, with the requirement that the theory be applied to a real system. The real system selected was a positional servosystem, using a DC motor.

A number of least squares algorithms were compared for their suitability to such an application. The algorithms examined were:

1) A standard, non-recursive solution of the least squares equations by Lower-Upper Factorisation of the information matrix.

2) A standard, recursive solution, i.e. Recursive Least Squares, RLS.

3) Two reduced order solutions using a priori knowledge of the type number of the servosystem (LU Factorisation and RLS).

4) An Extended Least Squares Solution, using a recursive algorithm.

5) Several non-recursive solutions using instrumental variables.

The methods were initially examined using a software simulation of the servosystem. This simulation was based on a linear, second-order model. It was concluded that the preferred methods were the reduced-order solutions using a priori knowledge.

The following hypothesis was examined:

By raising the rate at which the signals are sampled, more information is provided to the estimator in any given period of time. Increasing the sampling rate should therefore result in a superior, real-time parameter estimator.

The simulation results indicated that the effect of sampling rate upon the quality of the estimate is different for the "Moving Average" coefficients, than for the "Auto-regressive" coefficients. For the noise model used in the simulations, an increase in sampling rate was found to improve the estimates of the "Auto-regressive" coefficients. Subsequent results, from measurements on the real system, showed that there is an upper sampling rate which should not be exceeded.

A real, positional servosystem was designed and constructed. Full details of this design are presented in the appendices of this thesis. The preferred algorithms, using a priori knowledge of the servosystem type number, were used to estimate the location of the unknown system pole. This pole was estimated using a number of different sampling rates. It was noted that the estimates deteriorated as the sampling rate was raised beyond a certain value. This was attributed to the decrease in signal to noise ratio with the increase in sampling rate.

All of the calculations and control were performed using a T800 Transputer. All programs were written in Occam2. Appendices E and G contain samples of the source code used for both the simulations and the estimation of the real system's pole location.

CONTENTS

SYNOPSIS i

CONTENTS iii

CHAPTER 1 INTRODUCTION 1

1-1 INTRODUCTION TO SYSTEM IDENTIFICATION AND PARAMETER ESTIMATION 1

The Uses of System Identification

1-2 THE PROBLEM INVESTIGATED IN THIS THESIS 2

1-3 METHODOLOGY 4

CHAPTER 2 LITERATURE REVIEW OF PARAMETER ESTIMATION USING THE METHOD OF LEAST SQUARES 6

CHAPTER 3 DESCRIPTION OF THE POSITIONAL SERVOSYSTEM 10

3-1 DESIGN AND CONSTRUCTION OF THE SERVOSYSTEM 10

The Drive Section of the Servosystem

The Measurement Section of the Servosystem

3-2 MODELLING OF THE DC PERMANENT MAGNET MOTOR . . 12

3-3 NON-LINEAR CHARACTERISTICS OF REAL DC MOTORS . . . 14

Non-Linear Torque-Armature Current Characteristic

Cogging

Temperature Variations

Demagnetisation of the Permanent Magnets

Non-Linear Friction-Angular Velocity Characteristic

Summary

3-4 REVIEW OF FRICTIONAL LOSSES 16

Sliding vs. Rolling Friction

Viscous Friction

Static Friction

Coulomb Friction

Solid Friction Models

Summary

3-5 MODELLING THE PULSE WIDTH MODULATOR CONTROLLED MOTOR - STEADY STATE BEHAVIOUR 21

Introduction

An Equivalent Circuit for the Armature Winding and its Supply

Derivation of the PWM Steady-State Model

3-6 MODELLING THE PULSE WIDTH MODULATOR CONTROLLED MOTOR - DYNAMIC BEHAVIOUR 26

Determination of the Component Values of the Equivalent Circuit of the Armature Winding

PSpice Simulations of the Equivalent Armature Circuit

A Disturbance Model of the Armature Current

Spectral Content of the Disturbance Signal as a Function of PWM Duty-Cycle

The Influence of the Worst Case Disturbance upon Motor Speed

Final Simple Model of the PWM Controlled Armature Current

CHAPTER 4 PARAMETER ESTIMATION OF A DISCRETE-TIME MODEL OF THE SERVOSYSTEM 38

4-1 DEVELOPMENT OF A DISCRETE-TIME, INPUT-OUTPUT MODEL OF THE SERVOSYSTEM 38

The Need for a Discrete-Time Model

Z Transform Notation

Determination of the Overall Discrete-Time Input-Output Model of the System

4-2 THE LEAST SQUARES METHOD OF PARAMETER

ESTIMATION 42

4-3 A NON-RECURSIVE SOLUTION OF THE LEAST SQUARES EQUATIONS 44

4-4 A RECURSIVE SOLUTION TO THE LEAST SQUARES EQUATIONS 46

The Need for Recursive Algorithms

Young's Approach to a Recursive Algorithm

CHAPTER 5 COMPARISON OF DIFFERENT SOLUTIONS OF THE LEAST SQUARES EQUATIONS 50

5-1 REVIEW OF SOME ALTERNATIVE METHODS OF SOLUTION . .50

Some Direct Methods of Solution

Cramer's Rule, Gaussian and Gauss-Jordan Elimination

LU Factorisation

Iterative Methods of Solution

5-2 BASES OF COMPARISON OF LU FACTORISATION AND RLS METHODS 52

Factors Affecting the Selection of the Preffered Method

The Selection of Computer Simulation as the Basis of the Comparison

Some Comments on the Simulation Programs

Computation Time

Rate of Convergence of Estimates

Bias of the Estimates

Sensitivity of the Estimators to noise

Richness of the Input Signal

Initial Values of the estimator Variables

Sampling Period of the Estimator

Stability of the Estimator

5-3 THE COMPARISON OF LU FACTORISATION AND RLS METHODS USING A STOCHASTIC INPUT SIGNAL AT A STANDARD SAMPLING RATE 59

Results of the Simulations

Computation Time

Rate of Convergence of Estimates

Bias of the Estimates

Sensitivity of the Estimators to Noise

Richness of the Input Signal

Sampling Period of the Estimator

Stability of the Estimator

Summary of Simulations Using a Stochastic Input Signal and Estimator with Sampling Period of 0.2 Seconds

5-4 THE COMPARISON OF LU FACTORISATION AND RLS METHODS USING A STOCHASTIC INPUT SIGNAL WITH AN INCREASED SAMPLING RATE 68

Results of the Simulations

Rate of Convergence of Estimates

Sensitivity of the Estimators to Noise

Richness of the Input Signal

Sampling Period of the Estimator

5-5 THE COMPARISON OF LU FACTORISATION AND RLS METHODS USING TEST INPUT SIGNALS 72

Description of Simulations Using Test Input Signals

Test Input Signal Definitions

Results of Simulations

Rate of Convergence of Estimates

Richness of the Input Signal

Sampling Period of the Estimator

Stability of the Estimator

5-6 CONCLUSIONS OF THE COMPARISON OF THE LU FACTORISATION AND RLS METHODS 87

CHAPTER 6 A REDUCED ORDER ESTIMATOR USING A PRIORI KNOWLEDGE OF THE SYSTEM TYPE NUMBER 88

6-1 USE OF A PRIORI KNOWLEDGE OF PLANT INTEGRAL ACTION TO REDUCE THE ORDER OF THE ESTIMATOR 88

Introduction

Derivation of the Reduced Order Estimator

6-2 THE COMPARISON OF THE STANDARD AND REDUCED ORDER RLS METHODS OF SOLUTION 89

Description of the Simulations upon which this Comparison is Made

Results of the Simulations Using the 3 & 4 Parameter RLS Methods

Relative Merits of the Three and Four Parameter RLS Methods

CHAPTER 7 AN INVESTIGATION OF TWO TECHNIQUES INTENDED TO REDUCE THE ESTIMATOR BIAS 98

7-1 INTRODUCTION 98

7-2 SOURCE OF ESTIMATE BIAS 98

7-3 ESTIMATE BIAS AS A CONSEQUENCE OF CORRELATED NOISE 99

7-4 EXTENDED LEAST SQUARES (ELS) 101

Introduction to the ELS Method

Computed Errors Based upon the Error in the One-Step Ahead Prediction

Computed Errors Based upon the Residual Sequence

Results of the Simulations Using the ELS Method

Relative Merits of the 3 Parameter RLS and 6 Parameter ELS Methods

7-5 INSTRUMENTAL VARIABLE TECHNIQUES 108

Introduction to Instrumental Variables

The Ordinary IV Method as a Means of Reducing Estimate Bias

Results of the Simulations Using Instrumental Variables

Relative Merits of the 3 Parameter RLS and 3 Parameter Instrumental Variable Methods

CHAPTER 8 THE APPLICATION OF THE THREE PARAMETER ESTIMATORS TO THE REAL SERVOSYSTEM 115

8-1 DESCRIPTION OF THE POSITIONAL SERVOSYSTEM 115

8-2 THE DETERMINATION OF THE LOCATION OF THE UNKNOWN SYSTEM POLE 115

Introduction

The Step Response of the Servosystem

8-3 USE OF THE THREE PARAMETER RLS ESTIMATOR WITHOUT MOTOR STOPPING OR REVERSAL 118

The Need to Avoid Motor Stopping or Reversal

The Test Input Signal

Results from the Three Parameter RLS Estimator without Motor Stopping or Reversal

Interpretation of the Results Using the RLS Estimator without Motor Stopping

or Reversal

Conclusion Drawn from the Above results

8-4 THE EFFECT OF INCREASED INERTIAL LOAD 125

Results from the Three Parameter RLS Estimator on the System with Turntable, and without Motor Stopping or Reversal

Interpretation of the Results with Increased Inertial Load

8-5 THE USE OF THE THREE PARAMETER RLS ESTIMATOR WITH MOTOR REVERSALS 130

The Test Input Signal Used to Obtain Motor Reversals

Results from the Three Parameter RLS Estimator with Motor Reversals

Interpretation of the Results Using the RLS Estimator with Motor Reversals

Conclusions Drawn from the Above Results

8-6 USE OF THE THREE PARAMETER LU FACTORISATION ESTIMATOR WITHOUT MOTOR STOPPING OR REVERSAL 137

The Reason for Reconsidering the LU Factorisation Algorithm

The Experimental Set-Up

Results from the Three Parameter LU Factorisation Estimator without Motor Stopping or Reversal

Interpretation of the Results Using the LU Factorisation Estimator without Motor Stopping or Reversal

Conclusions Drawn from the Above Results

8-7 THE THREE PARAMETER LU FACTORISATION METHOD WITH INFORMATION MATRIX RESETTING 144

Introduction

Results of Using Information Matrix Resetting

Interpretation of the Results for Information Matrix Resetting

Conclusions Concerning Information Matrix Resetting

CHAPTER 9 CONCLUSIONS 149

9-1 INTRODUCTION 149

9-2 INVESTIGATION OF INCREASED SAMPLING RATE AS A MEANS OF REDUCING THE PARAMETER ESTIMATION TIME 149

Conclusions Based upon Measurements on the Real System

Discussion Concerning the Simulations

Selection of Sampling Rate

9-3 USE OF INFORMATION MATRIX RESETTING TO IMPROVE THE NUMERICAL STABILITY OF PARAMETER ESTIMATORS . . . 151

The Problem of Numerical Instability

The Preferred Solution

9-4 COMPARISONS OF DIFFERENT LEAST SQUARES ALGORITHMS 154

9-5 OBSERVATIONS ON THE USE OF A TRANSPUTER IN REAL-TIME DIGITAL CONTROL 156

APPENDIX A DESCRIPTION OF THE TRANSPUTER SYSTEM 158

A-1 TRANSPUTER HARDWARE 158

A-2 TRANSPUTER SOFTWARE 158

Choice of Software Environment

Programming Language

A-3 DYNAMIC SWITCHING OF THE SYSTEM CONFIGURATION . 160

Explanation of 'Dynamic Switching'

The Need for Dynamic Switching

A-4 TIMING 162

APPENDIX B DESCRIPTION OF THE SERVOSYSTEM 163

B-1 DESCRIPTION OF THE COMPLETE POSITIONAL SERVOSYSTEM 163

B-2 THE PERMANENT MAGNET DC MOTOR 163

B-3 THE OPTICAL ENCODER 164

B-4 THE PLANETARY GEARHEAD 164

B-5 THE PULSE-WIDTH MODULATOR (PWM) UNIT AND POWER BRIDGE 172

B-6 THE INTERFACE CIRCUITRY 172

The Transputer Digital Interface Module

The Digital-to-Analogue Converter (DAC)

The Counter Circuit

The Counter Circuit Controller

Restrictions on the Counter Circuit Use

APPENDIX C NON-RECURSIVE SOLUTION OF THE LEAST SQUARES EQUATIONS 175

C-1 THE SEQUENCE OF EQUATIONS NECESSARY TO SOLVE A FOUR PARAMETER PROBLEM USING LU FACTORISATION 175

Introduction

Sequence of Equations for LU Factorisation Solution - Four Parameter Case

APPENDIX D PSPICE SIMULATIONS OF THE ARMATURE MODEL 178

APPENDIX E OCCAM2 SOURCE CODE OF THE SIMULATION PROGRAMS . 179

Source Code of the Method of Using A Priori Information to Reduce the Number of Normal Equations by One

Source Code for the Extended Least Squares Method

Source Code for the Solution Using Instrumental Variables

APPENDIX F THE LEAST SQUARES NOISE MODEL 192

APPENDIX G OCCAM2 SOURCE CODE OF THE PROGRAMS USED TO TEST THE REAL SERVOSYSTEM 194

Source Code Required to Perform Parameter Estimation of the Real Servosystem

Source Code Enabling Control of the B008 Motherboard Linkswitch

REFERENCES 205

ACKNOWLEDGMENTS 216

BIOGRAPHY 216

CHAPTER 1 INTRODUCTION

1-1 INTRODUCTION TO SYSTEM IDENTIFICATION AND PARAMETER ESTIMATION

EXPLANATION OF THE TERMS "SYSTEM IDENTIFICATION" AND "PARAMETER ESTIMATION"

System Identification may be viewed as the complete process by which an adequate mathematical model of a real system is obtained [1-9]. A model may be derived either by the application of physical laws, or by the experimental observation of the system's response to known input signals. Usually both approaches are required in order to develop an accurate model.

Once the structure of the model is known, it then becomes necessary to determine the numerical values of the various parameters of that model. This process is referred to as Parameter Estimation.

In the following work, the system is modelled by a linear difference equation. The justification of this model and the assumptions made in its derivation are given. Essentially this model, including its order, is derived from a consideration of the known components of the system, and the laws of physics that these components must obey.

It is well known that the coefficients of such a difference equation may be estimated from a knowledge of the system components. Unfortunately not all characteristics of the components may be known with sufficient accuracy. The process of determining the optimum numerical values of these coefficients is an example of Parameter Estimation. In the following work this process is achieved by using the model to predict the next output of the real system. The coefficients are estimated by adjusting them so as to minimise the summed square of the error in this prediction.

Parameter Estimation is thus a process that may be used as part of an overall exercise

known as System Identification.

THE USES OF SYSTEM IDENTIFICATION

In practice system identification is performed for one of two purposes.

The first is concerned with the understanding of the behaviour of a real system, as a necessary prerequisite to the successful control of that system. From Kalman's earliest self-optimising controller [1-1] all self-tuning controllers have required the inclusion of some means of identifying the controlled system. Indeed, this feature of a distinct system identifier may be used in the definition of both self-tuning controllers and regulators [1-2].

The second use of system identification in engineering is in fault detection. Numerous papers have been published on this aspect of system identification [e.g. 1-3,1-4,1-5]. Willsky [1-6] published a survey on the use of Kalman Filters, and related concepts, in the detection of system "failures".

1-2 THE PROBLEM INVESTIGATED IN THIS THESIS

The following work is concerned with the real-time parameter estimation of a discrete-time model of a servosystem. In particular, the parameters are to be estimated as rapidly as possible, necessitating the calculations being based upon relatively few measurements. A typical application of the system considered is the control of a turntable upon which a robotic arm has been mounted.

Such a system can be expected to exhibit behaviour predominantly dependent upon the ratio of frictional losses to inertia of the rotating turntable. The model proposed is of second order, with one pole determined by this ratio, and the other associated with the integral action of converting angular velocity into angular displacement.

If the end-effector of the robot arm is moved further out, away from the axis of rotation of the turntable, or if it picks up a load, then the inertia presented to the motor driving

the turntable will increase. From a system identification viewpoint, this would cause a jump in the parameter values exhibited by the system. The controller of the turntable drive should be aware of these changes, in order to compensate for them.

The controller may contain a parameter estimator. In which case, after the jump change in parameters this estimator should be reset, and a new set of estimates produced. The parameter estimator produces these new estimates from the sampled input and output signal sequences available after the jump change. The longer the length of these sequences the better (in terms of both increased information content and a consequent reduction in sensitivity to noise).

By increasing the sampling rate of these signals a larger number of signal pairs may be added to the sequences in any given, fixed time interval.

This thesis investigates whether or not such a simple technique can be used to improve the estimation process. The intention is to perform the new parameter estimation using only the relatively few, post-jump measurements, rather than a long, cumulative history of probably irrelevant sampled values.

This approach was inspired by a consideration of an ideal, noise-free system of known order. In such a deterministic case, a finite number of samples are required in order to exactly determine the values of the parameters. For a second-order, noise-free system, the difference equation relating the input and output signal sequences should have four coefficients. In the absence of noise, only four consecutive pairs of input-output signal samples should be required to determine the exact values of all four coefficients.

In a real system, all measured signals have associated noise. In addition no real, discrete-time system can be precisely modelled by a linear, low order difference equation [1-7].

From the outset this approach was recognised as having a number of problems, e.g.:

i) The small number of samples available to the estimator will make it sensitive to noise.

ii) The sensitivity of the estimated parameter values to the sampling period chosen.

1-3 METHODOLOGY

The structure of the model of the system is assumed to be known, a priori. The problem then is reduced to the determination of the numerical values of the parameters of that model.

The ideas and theory developed are aimed at one, real system. This system is a positional servosystem using a permanent magnet DC motor. The armature current of this motor is controlled by a pulse-width modulated (PWM) amplifier. Details of the designs and construction of the hardware are given in Appendix B.

The models used to describe the servosystem were all input-output models. There was no perceived advantage in using state-variable models. The work was restricted to techniques of parameter estimation based upon the Method of Least Squares.

A simple, linear model of the servosystem is proposed. The shortcomings of this model are discussed. Particular emphasis is placed upon justifying the model of the PWM unit, and upon the non-linear characteristics expected of a real DC motor. The simple model is based upon the known (e.g. datasheet) characteristics of the components of the servosystem. This model formed the basis of a software simulation of the servosystem. A software simulation was used so as to permit the testing of parameter estimation algorithms without problems resulting from either an inadequate process model or an unknown noise process.

This approach ensured that the true coefficients of the system were calculable, and hence available for comparison with the estimates. Testing the algorithms using either the real servosystem, or an analogue computer, would not have resulted in the availability of true coefficient values. The use of software simulation overcame the problems of both inadequate modelling and component drift.

The simulations were used to compare two parameter estimation algorithms at a time. The "better" of these two was then tested against a different algorithm, and this process was repeated until all of the algorithms to be considered had entered into this comparison.

Six algorithms were compared on the basis of their suitability to the specific task of real-time parameter estimation of a second-order, type one system. The algorithms considered were:

i) Four parameter, non-recursive solution using Lower-Upper (LU) Factorisation

ii) Four parameter solution using Recursive Least Squares (RLS) Method

iii) Reduced order (three parameter) solution using LU Factorisation

iv) Reduced order (three parameter) solution using RLS

v) Six parameter Recursive Extended Least Squares Method

vi) Reduced order (three parameter) solution using instrumental variables.

The preferred algorithms (iii and iv) were tested on the real system for a number of sampling rates. The other four algorithms having exhibited inferior performance for this application in the comparisons based upon simulation were not used on the real system.

The theoretically derived second-order model was assumed to be adequate to describe the real system behaviour at all times. There was no attempt to define a restricted, linear region of operation of the servosystem. Two different input test signals were used on the real system. The first caused the motor to rotate continuously in the one direction. This was done to avoid the non-linear, unmodelled behaviour expected in both reversing and stopping the motor. The model used did not consider such effects as backlash and deadband. The second test signal caused the motor to be driven alternately in either direction. This enabled some observations to be made on the effects of these unmodelled non-linearities upon the estimator performance.

All of the calculations, both in simulations and real-time, were performed using a T800 Transputer [1-8]. A full description of the transputer configuration is given in Appendix A. The hardware developed comprising a minimal, but powerful digital identifier and controller.

CHAPTER 2 LITERATURE REVIEW OF PARAMETER ESTIMATION USING THE METHOD OF LEAST SQUARES

The first self-tuning controllers [1-1,2-1,2-2] were primarily intended for the control of systems with unknown, time-invariant parameters. These designs involved the minimisation of a selected Performance Index. Kalman and Koepcke [2-3] attributed the idea that control systems should be designed in such a way as to minimise a performance index based upon the integral of a squared error signal to Wiener.

The original development of the Method of Least Squares was done by Legendre and Gauss, both publishing their work near the start of the nineteenth century [2-4]. These famous mathematicians independently developed this method to apply to batches of previously obtained astronomical measurements. The first recursive solution of the Least Squares Problem was published by Plackett [2-5]. This recursive approach was developed as a means of minimising the computation required if additional observations were to become available.

Kalman's early design of a self-optimising, discrete-time control system [1-1] used a performance index consisting of the mean squared error in the prediction of the system's output signal. Stromer [2-6] published a brief bibliography covering the early work on self-optimising control systems.

The simple performance index has been extended by various authors into more elaborate Cost Functions [e.g. 2-4]. Clarke and his co-workers [2-2,2-8] proposed cost functions that penalised control effort as well as output prediction error.

With the need to obtain real-time solutions to the Least Squares Equations on low cost computers, considerable attention has been paid to the numerical accuracy of the various algorithms available. A number of "Square Root" algorithms have been proposed to enable less well-conditioned problems to be solved [2-9,2-10,2-11]. These algorithms are considered to give a similar improvement in numerical accuracy as would the resort to double precision

arithmetic. They are especially valued for dealing with the problem of "round-off" errors in computers of short word-length.

Graupe and his co-authors considered a number of different algorithms based upon the Method of Least Squares [2-12]. Their paper classified the algorithms into the following groups:

i) Batch least squares methods (including covariance, autocorrelation and partial correlation methods)

ii) Sequential least squares methods (including the standard Recursive Least Squares Method [2-13], and sequential forms of the covariance and partial correlation methods)

iii) Lattice algorithms

iv) Square Root algorithms (including the use of Householder Transformations)

In their comparison the authors considered the convergence characteristics of each method. They concluded that "the convergence and convergence-rate properties of different least-squares algorithms are almost identical for 36 or 64 bit accuracy".[1]

Of all the methods considered by Graupe and his colleagues, they nominated the "Sequential Direct LS" approach, this is the same as that called Recursive Least Squares (RLS) in Chapter 4 of this thesis.

Wong and his co-authors [2-14] reported problems with numerical instability when using the RLS method within an industrial application. This group favoured an algorithm based upon the Recursive U-D Factorisation Method [2-11]. It should however be noted that this choice was in response to the problem of "bursting" of the estimates [2-14]. This problem is prevalent when the estimator has a large set of past samples available to it. The present work considers the case of the estimator having few samples, and hence "bursting" is not a concern

[1]The T800 Transputer used throughout the present work has a shorter wordlength, being a 32-bit machine.

in algorithm selection.

The application of self-tuning controllers to time-invariant, almost linear systems is now extensively covered in the literature. The problems of such designs and suggested solutions being well documented. e.g. biased estimates resulting from coloured noise [2-15], lack of persistency of excitation [2-16].

The self-tuning controllers can be easily modified to produce adaptive controllers suitable for systems with slowly time-varying parameters. Kalman, Swerling and Wittenmark [1-1,2-17,2-1] have all noted the ease by which this could be achieved.

In order to make a self-tuning controller adaptive, the most common modification is the introduction of an exponential weighting to discount previously measured values. This is achieved by scaling past data by a constant forgetting factor [2-18]. A number of authors have encountered problems when using a constant forgetting factor [e.g. 2-19,2-20]. Invariably the choice of this factor is empirical, ad hoc and crucial to the system performance and as a consequence, a number of schemes involving variable forgetting factors have been proposed [e.g. 2-18,2-22,2-23,2-24].

Gurubasavaraj [2-21] noted that an inappropriate choice of forgetting factor resulted in "numerical instabilities". Gurubasavaraj found it necessary to supplement the use of a constant forgetting factor with some resetting of the covariance matrix. Others have also resorted to resetting the information or covariance matrix of the estimator [2-25], or the use of a cascade of estimators [2-26].

The choice between forgetting factor or matrix resetting is largely determined by the rate at which the system parameters are able to change. The literature tends to consider two classes of parameter variation, commonly described as "drift" and "jump" changes. Goodwin and Teoh [2-27] provided a review of the literature on this problem up to 1983. They suggested an algorithm based upon covariance resetting, together with results that show far better performance than the simple use of a constant forgetting factor. Their algorithm, in its

simplest form, requires regular, periodic resetting of the covariance matrix. They suggest that in the case of "jumps" in parameter values, that there is some advantage in monitoring the prediction error as a basis for covariance resetting.

The choice of sampling period for parameter estimation of a real system must inevitably be related to the time constant of that system. Commonly the sampling period is based upon either the closed loop bandwidth of the system [e.g. 5-13,5-14,5-15], or some other time constant of the system [e.g. 5-16]. Astrom and Wittenmark [2-28] provide an introductory treatment of the selection of an appropriate sampling period. The parameter values are dependent upon the sampling period, and their sensitivity to error has been noted in both the selection of sampling period and choice of modeling operator [2-29].

Astrom [2-7] mentions that adaptive schemes derived from Stochastic Control Theory may be used to identify systems in which the parameters vary rapidly, but that these schemes are either difficult to implement or are impracticable.

CHAPTER 3 DESCRIPTION OF THE POSITIONAL SERVOSYSTEM

3-1 DESIGN AND CONSTRUCTION OF THE SERVOSYSTEM

The following brief description of the real servosystem is included as an introduction to the derivation and justification of a mathematical model of that system. A more comprehensive description of the design and construction is provided in Appendix B.

THE DRIVE SECTION OF THE SERVOSYSTEM

The servosystem uses an 18 volts, 40 watts, permanent magnet DC motor. This motor is controlled by its armature current. The armature winding is supplied from a bridge of power, VMOS transistors. The transistors act as switches, as shown in Figure 3-1.

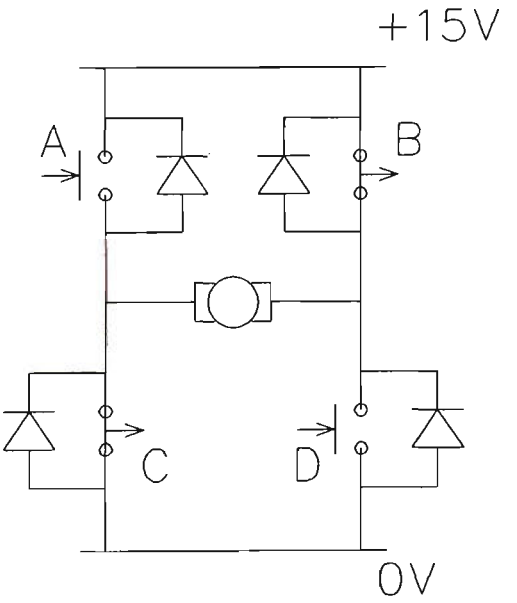

FIGURE 3-1 THE ARMATURE WINDING IN A BRIDGE OF VMOS TRANSISTOR SWITCHES

The transistors are alternately switched in opposite pairs. Initially, transistor switches A and D are open, with B and C closed. In this case the applied armature voltage will attempt to drive a positive conventional current through the winding from right to left.

Transistors B and C are subsequently turned-off also. For a brief period (nominally ten microseconds) all four switches are open, to avoid shorting out the +15 volt supply. After this short period, switches A and D are closed. The applied armature voltage now attempts to drive positive conventional current through the winding from left to right, however the total current is dependent upon the voltage induced in the armature winding.

The switching cycle is then completed by a further short period when all four switches are open. The complete cycle of four switching operations is repeated at a nominal, ultrasonic rate of 22KHz.

TABLE 3-1 SEQUENCE OF TRANSISTOR SWITCHING

Period Number	Switches A&D	Switches B&C
1	open	closed
2	open	open
3	closed	open
4	open	open

The armature current is controlled by varying the ratio of Period 1 to Period 3. This ratio is referred to here as the "duty-cycle" of the pulse width modulator (PWM) Unit. A duty-cycle of 50% indicates that Periods 1 and 3 are of equal duration, resulting in an average armature current of zero.

The design of the PWM Unit is based upon two comparators fed with an internally generated 22KHz. triangle wave and an input voltage, $v_i(t)$. The input signal, $v_i(t)$,is

produced by a digital-to-analogue converter (DAC) connected to an output port of the digital controller. Figure 3-2 shows the interconnection of the various circuit blocks of the Drive Section.

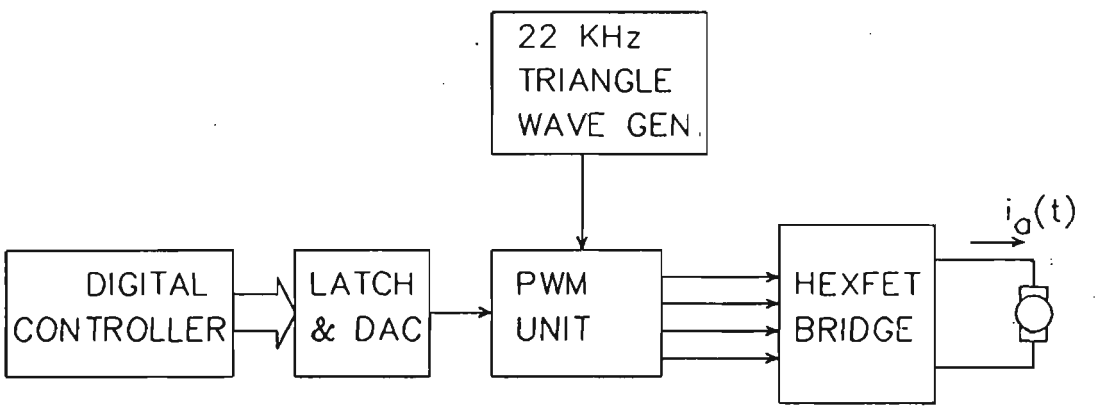

FIGURE 3-2 BLOCK DIAGRAM OF THE DRIVE CIRCUITS OF THE SERVOSYSTEM

THE MEASUREMENT SECTION OF THE SERVOSYSTEM

The controlled output variable of the system was the angular position of the output shaft of a gearbox driven by the motor armature shaft. Position measurements were made using an optical encoder connected to the armature shaft. Details of this transducer, and its associated circuitry are given in Appendix B.

3-2 MODELLING OF THE DC PERMANENT MAGNET MOTOR

For simplicity a very simple, linear model of the DC motor was selected. The majority of introductory textbooks in control engineering offer such models of DC motors [e.g. 3-1,3-2,3-3].

The selected model is summarised in Figure 3-3, with the following terms as defined:

$v_a(t)$, the applied armature voltage

$e_b(t)$, the back emf of the armature winding

$i_a(t)$, the armature current

$T(t)$, the torque developed by the armature

$\omega(t)$, the angular velocity of the armature

$\theta(t)$, the angular position of the armature

R_a , the resistance of the armature winding

L_a , the inductance of the armature winding

k_i , the motor torque constant

B , the viscous frictional coefficient for the rotating section of the machine

J , the rotational inertia of the machine

k_b , the back emf constant

Capital letters denote Laplace Transforms, i.e. $\mathcal{L}[i_a(t)] = I_a(s)$

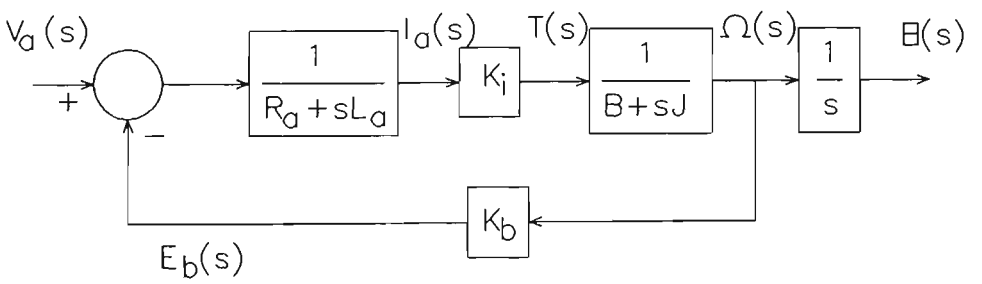

FIGURE 3-3 S-DOMAIN BLOCK DIAGRAM OF THE SIMPLE, LINEAR MODEL OF THE DC MOTOR

The above model incorporates a number of assumptions:

1) that the armature winding may be treated as a single winding comprising of a series connection of a lumped resistance, lumped inductance and a dependent voltage source. This voltage source modelling the back emf induced in the winding.

2) that the back emf is directly proportional to the angular velocity of the armature.

3) that the torque developed by the armature is directly proportional to the armature current, and that all of this developed torque is available to overcome frictional losses and to accelerate the rotor. i.e.

$$T(t) = B\omega(t) + J\frac{d}{dt}\omega(t) \qquad 3\text{-}1$$

There is no disturbance torque included in the model.

4) that the mechanical, frictional losses are viscous in nature. This assumption is reviewed in Section 3-3 below.

3-3 NON-LINEAR CHARACTERISTICS OF REAL DC MOTORS

Kuo and Tal [3-4] point to a number of inadequacies of the above model. These inadequacies are summarised below:

NON-LINEAR TORQUE-ARMATURE CURRENT CHARACTERISTIC

There are two sources of non-linearity in this characteristic.

i) The B-H curves of the materials of the motor's magnetic circuit are non-linear. The assumption of a "Torque Constant" is particularly suspect at high current levels, where magnetic saturation manifests itself as a reduction in the value of this "constant".

ii) The second source of non-linearity is associated with the angular position of the armature with respect to the stator. This effect is noticeable if the motor has an insufficient number of commutation points and manifests itself as a cyclical variation of torque It may be modelled as a cyclical torque disturbance.

COGGING

The reluctance of the magnetic circuit may vary with the angular position of the rotor. The rotor will then have preferred, stationary shaft-angle positions. At low speeds the rotor may be observed to "cog".

TEMPERATURE VARIATIONS

Arguably the most significant problem with temperature changes is the dependence of the relative permeability of the motor's magnetic materials upon temperature.

DEMAGNETISATION OF THE PERMANENT MAGNETS

Excessive currents may demagnetise the permanent magnet, resulting in an irreversible change in the characteristics of the motor.

NON-LINEAR FRICTION-ANGULAR VELOCITY CHARACTERISTIC

The viscous damping coefficient of the motor is not a constant. There are a number of mechanical and electromagnetic phenomena that contribute to the value of this coefficient. e.g. friction in the bearings and windage, and losses due to eddy currents and magnetic hysteresis. Not all of these components produce losses that are directly proportional to angular velocity.

SUMMARY

Of the above non-linear characteristics it was expected that the most significant would be the non-linearity of the frictional losses. The other non-ideal characteristics were considered to be either of a secondary nature or avoidable. An high quality motor was purchased, so as to minimise the significance of the above characteristics.

The assumption concerning the non-linearity of the frictional losses however, had the potential to discredit the whole of the model. Accordingly the following section provides a brief overview of the literature and available models describing frictional losses appropriate to DC motors.

3-4 REVIEW OF FRICTIONAL LOSSES

The literature concerning frictional losses in DC motors suggests that a general model, based on physical principles, is not available. Instead there exist a variety of empirical models, which tend to be specific to either certain modes of motor operation, or types of motor construction.

SLIDING VS. ROLLING FRICTION

Bowden and Tabor [3-20] provide an insight into why diverse models are able to co-exist. The underlying physical processes that manifest themselves as either "sliding friction" or "rolling friction" are entirely unrelated. Accordingly a motor constructed with ball-bearings would have a different speed-friction characteristic from one constructed using simple journal bush bearings.

VISCOUS FRICTION

The model of the motor proposed in Section 3-2 considered that the frictional component of shaft torque is directly proportional to the angular velocity of the shaft.

$$T_f(t) = B\frac{d}{dt}\theta(t) \qquad 3\text{-}2$$

where

$T_f(t)$, is the frictional component of shaft torque

$\theta(t)$, is the angular position of the shaft

B , is the Viscous Frictional Coefficient [3-4]

For the model of Section 3-2 the Viscous Frictional Coefficient is a constant.

A linear relationship between frictional torque and shaft speed implies a quadratic relationship between the frictional power loss and shaft speed. Puchstein [3-6] gives empirically based equations for determining the power loss in small DC motors, operated at constant speed. He suggests that the power loss due to bearing friction and windage is proportional to the armature speed raised to the power of 1.5. Kuo and Tal [3-5] state that not all components of the viscous damping effect are linearly related to angular velocity.

The distinction between the terms viscous frictional coefficient and viscous frictional constant is therefore important.

STATIC FRICTION

Static Friction is a component of friction associated with stationary surfaces only.

$$T_s(t) = \pm T_0|_{\dot{\theta}=0} \qquad 3\text{-}3$$

where

$T_s(t)$, is the torque due to Static Friction

T_0 , is an empirically obtained constant

$\dot{\theta}$, is the angular velocity of the shaft

Kuo & Tal [3-4] wrote that "once motion begins, the static frictional force vanishes, and other frictions take over".

Bowden & Tabor [3-20] differentiate between the "Coefficient of Static Friction" and the "Coefficient of Kinetic Friction". These authors give three reasons why the Coefficient of Static Friction should be the larger:

i) Stationary surfaces will probably be in better contact with one another

ii) Over a period there may be contact diffusion of atoms from the material of one surface to that of the other

iii) Any thin film, that would serve as a lubricant between the two surfaces, may break down.

COULOMB FRICTION

Classical models of Friction recognise a third type of friction, known as Coulomb Friction. The torque due to Coulomb Friction, $T_c(t)$, is given by:

$$T_c(t) = T_1 \cdot \frac{\left[\frac{d\theta}{dt}\right]}{\left|\frac{d\theta}{dt}\right|} \qquad 3\text{-}4$$

where

T_1 , is an empirically obtained constant.

Depping & Voits [3-7] considered the load torque presented to a DC motor driving a DC generator. They viewed the load torque as consisting of two components, one directly proportional to shaft speed, the other a speed independent constant. The difference between the load torque, and the torque developed by the motor being referred to as the "Acceleration Torque".

The physical justification of Coulomb Friction is to model the magnetic and mechanical hysteresis associated with reversals of direction of motor rotation [3-8]. The effect of Coulomb Friction in real systems is to reduce overshoot and oscillations, at the expense of steady-state error [3-4].

SOLID FRICTION MODELS

Some authors have noted that models of friction based upon Viscous, Coulomb and Static components, fail to describe the behaviour of real systems [3-9,3-10,3-13]. A more elaborate model, the Solid Friction Model, has been developed to overcome these failures. This new model has been subject to several developments and simplifications [3-9,3-10,3-12].

The Solid Friction Model is the result of finding a mathematical function that will fit an experimentally obtained curve. Unfortunately the resulting function is both unwieldly and non-linear:

$$\frac{dF(x)}{dt} = S.\sigma\left|1 - \frac{F}{F_c}.sgn\ \dot{x}\right|^{i}.\frac{dx}{dt} \qquad 3\text{-}5$$

where

F(x),	is the Frictional Force
x,	is the displacement
i, σ and F_c	are constants, chosen so as to enable the function to fit the required curve
S,	is a "Stability Factor" [3-11]

Dahl has used the Solid Friction Model to represent a system comprising of a DC motor, gearbox and inertial load [3-11]. This paper reveals the complexity involved in using the Solid Friction Model.

Walrath [3-13] used a simplified Solid Friction Model to describe the behaviour of a gimbal mounted tracking system. Walrath's model may be summarised by the equation:

$$T_f(t) + \tau.\frac{dT_f(t)}{dt} = (sgn\ \dot{\lambda}).T_c \qquad 3\text{-}6$$

where

$T_f(t)$, is the predicted frictional torque

T_c , is the constant rolling friction torque

$sgn\ \dot{\lambda}$, equals either +1 or -1, depending upon the sign of the relative gimbal velocity

τ , is a suitably chosen constant

SUMMARY

There exist two groups of models of friction appropriate to the positional servosystem considered in the present work.

The first group is based upon the "classical" components of friction, namely Viscous, Coulomb and Static. Various combinations of these components may be used, depending upon the operational mode of the motor considered. A combination of the Viscous and Coulomb components proving adequate to many authors [e.g. 3-7,3-14].

The second group of models are those derived from Dahl's Solid Friction Model.

The model of Section 3-2 uses a simple model consisting of viscous friction alone. Further, the Coefficient of Viscous Friction was taken to be a constant. This selection was made since it is so simple. The above review of friction indicates the level of confidence that can be placed on this choice. However it would not be sensible to choose a more complex model, before the arrival of experimental results makes the use of such a model imperative.

The above overview of friction suggests that different models are necessary for different modes of motor operation. i.e.

i) rotation continuously in one direction with only slight speed variations

ii) rotation continuously in one direction with a wide range of speeds

or iii) motion subject to either stopping or reversals of direction of rotation.

3-5 MODELLING THE PULSE WIDTH MODULATOR CONTROLLED MOTOR - STEADY STATE BEHAVIOUR

INTRODUCTION

The model of Section 3-2 showed the applied armature voltage as the input signal to the system. The real system has this voltage generated by a Pulse Width Modulator (PWM) controlled bridge of power transistors. This circuit is not linear, its output voltage being a variable duty-cycle train of rectangular pulses.

The following sections, 3-5 and 3-6, provide an analysis of the effects of this circuit on the armature current, and are a justification of the simple, linear model given as Equation 3-39.

AN EQUIVALENT CIRCUIT FOR THE ARMATURE WINDING AND ITS SUPPLY

The bridge of power transistors supplies a variable duty-cycle, fixed frequency, +/- 15 volts, rectangular wavetrain across the terminals of the armature winding. This wavetrain is modelled by two, synchronised, independent voltage sources, as shown in Figure 3-4. In practice there must be a deadtime during which both $v_1(t)$ and $v_2(t)$ are zero, however to simplify the following analysis the deadtime is assumed to be zero.

A factor k is introduced to describe the duty-cycle of the pulse width modulation. This factor may take any value in the range $0 \leq k \leq 1$. The relationship between $v_1(t)$, $v_2(t)$ and k is shown in Figure 3-5.

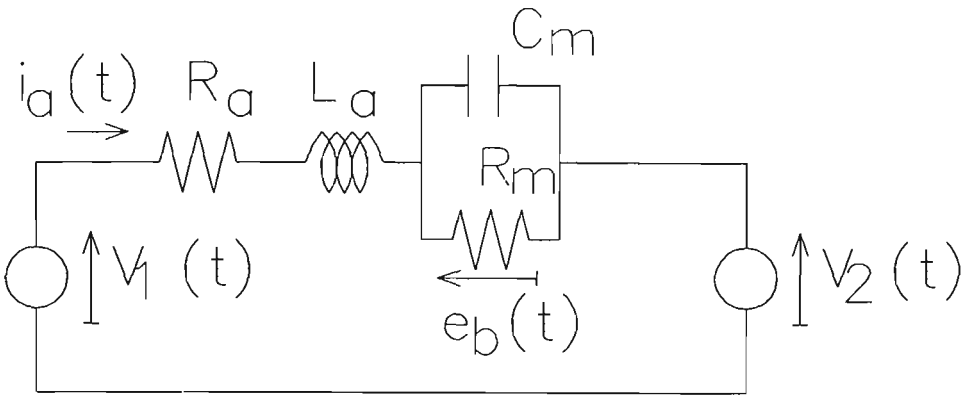

FIGURE 3-4 CIRCUIT DIAGRAM USED TO MODEL THE SWITCHING OF THE APPLIED ARMATURE VOLTAGE

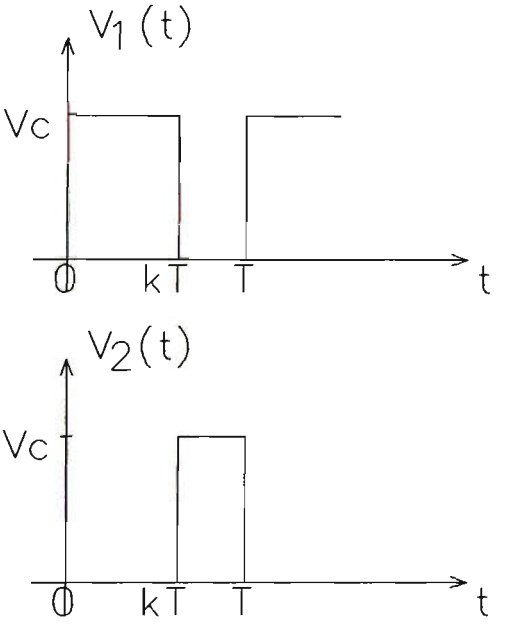

FIGURE 3-5 WAVEFORMS USED TO GENERATE THE APPLIED ARMATURE VOLTAGE

In Figure 3-4 the back-emf is considered to be the voltage developed across a parallel R-C circuit, rather than the more conventional use of either an independent or dependent voltage source. Next the use of the parallel R-C circuit is justified.

The model of Section 3-2 includes the following relationships:

$$T(t) = k_i i_a(t) \qquad 3\text{-}7$$

$$T(t) = B\omega(t) + J\frac{d}{dt}\omega(t) \qquad 3\text{-}8$$

$$e_b(t) = k_b\omega(t) \qquad 3\text{-}9$$

and therefore

$$i_a(t) = \frac{B}{k_i.k_b}e_b(t) + \frac{J}{k_i k_b}\frac{d}{dt}e_b(t) \qquad 3\text{-}10$$

Thus the back-emf may be modelled as the voltage developed across a parallel R-C circuit, as per Figure 3-4, where:

$$R_m = \frac{k_i k_b}{B} \quad Ohms \qquad 3\text{-}11$$

and

$$C_m = \frac{J}{k_i k_b} \quad Farads \qquad 3\text{-}12$$

The time constant of this circuit is equal to the Mechanical Time Constant of the motor.

DERIVATION OF THE PWM STEADY-STATE MODEL

The following sub-section of this work derives a steady-state relationship between the average armature current and the duty-cycle of the PWM Unit.

The following assumptions are made:

i) the PWM duty-cycle has been held at a constant value

ii) the load presented to the motor shaft is a constant torque.

As a consequence of these assumptions:

iii) variations in $\omega(t)$, and hence $e_b(t)$ are sufficiently small as to permit these variables to be treated as constants.

Assumption iii) relies on the Mechanical Time Constant of the motor being much greater than the time period of the PWM switching cycle. This assumption effectively ensures that the charge on the capacitor C_m remains constant throughout the period of the PWM switching cycle. The back-emf then becomes the voltage developed across the resistor R_m, with the capacitor replaced by an open-circuit,

i.e.

$$e_b(t) = R_m i_a(t) \qquad 3\text{-}13$$

The armature winding then exhibits a total resistance, R, where

$$R = R_a + R_m \qquad 3\text{-}14$$

Over the first part of the PWM cycle, $0<t<kT$, the armature current is given by:

$$i_a(t) = \frac{V_c}{R}.u(t) + \left[i_a(0) - \frac{V_c}{R}\right].\exp\left\{\frac{-R.t}{L}\right\} \qquad 3\text{-}15$$

where u(t) denotes a unit step.

Hence at the end of this first period the instantaneous value of the armature current will be given by

$$i_a(kT) = \frac{V_c}{R} + \left[i_a(0) - \frac{V_c}{R}\right].\exp\left\{\frac{-RkT}{L}\right\} \qquad 3\text{-}16$$

Over the second part of the PWM cycle ($kT < t < T$) the voltage $v_1(t)$ is zero, and $v_2(t)$ is now equal to $+V_c$ volts.

Define $\tau = t - kT$, in which case the second part of the cycle may be described by the range $0 < \tau < (1-k)T$. The initial current for this period may be described as

$$i_o = i_a(\tau)\big|_{\tau=0} = i_a(t)\big|_{t=kT} \qquad 3\text{-}17$$

and is as given above in Equation 3-16.

Over this second period it can be shown that:

$$i_a(\tau) = -\frac{V_c}{R} + \left[\frac{V_c}{R}+i_o\right].\exp\left\{\frac{-R\tau}{L}\right\} \qquad 3\text{-}18$$

Hence at the end of the PWM cycle it can be seen that:

$$i_a(t)\big|_{t=T} = i_a(\tau)\big|_{\tau=(1-k)T} = -\frac{V_c}{R} + \left[\frac{V_c}{R}+i_o\right].\exp\left\{\frac{-R(1-k)T}{L}\right\} \qquad 3\text{-}19$$

This equation is based on the assumption that the duty-cycle of the PWM is held constant, and accordingly:

$$i_a(t)\big|_{t=0} = i_a(t)\big|_{t=T} \qquad 3\text{-}20$$

Hence it can be shown that the initial current is given by:

$$i_a(t)\big|_{t=0} = \left\{\frac{V_c}{R}\right\}.\frac{1+2.\exp[\frac{-R(1-k)T}{L}]-\exp[\frac{-RT}{L}]}{1-\exp[\frac{-RT}{L}]} \qquad 3\text{-}21$$

The average value of the armature current over the period $0 \le t \le kT$, I_{AV1}, is given by:

$$I_{AV1} = \frac{1}{kT}\int_0^{kT} i_a(t).dt$$

$$= \frac{V_c}{R} + \frac{1}{kT}\left[i_a(t)\big|_{t=0}-\frac{V_c}{R}\right].\left[-\frac{L}{R}\right].\left[\exp(-\frac{kTR}{L})-1\right] \qquad 3\text{-}22$$

Similarly the average value of the armature current over the period $kT \le t \le T$, I_{AV2} is given by:

$$I_{AV2} = \frac{1}{(1-k)T}\int_0^{(1-k)T} i_a(\tau).d\tau$$

$$= -\frac{V_c}{R} + \frac{1}{(1-k)T}\left[i_a(t)\big|_{t=kT}+\frac{V_c}{R}\right].\left[-\frac{L}{R}\right].\left[\exp(\frac{(k-1)TR}{L})-1\right] \qquad 3\text{-}23$$

The average value of the armature current over the full switching cycle, I_{AV}, is given by:

$$I_{AV} = \frac{I_{AV1}.kT + I_{AV2}.(1-k)T}{T} \qquad 3\text{-}24$$

and hence

$$I_{AV} = \frac{V_c}{R}.(2k-1) \qquad 3\text{-}25$$

This equation shows the linear relationship between the duty-cycle k, and the average (over one cycle of the PWM switching period) armature current. The expression is valid subject to the assumptions stated.

3-6 MODELLING THE PULSE WIDTH MODULATOR CONTROLLED MOTOR - DYNAMIC BEHAVIOUR

In the foregoing analysis it was assumed that the duty cycle of the PWM drive was held constant and accordingly the dynamics of the motor were neglected. In this section the response to a change in duty-cycle will be considered. This response will be investigated by simulations using the software circuit analysis package "PSpice" [3-15,3-16]. The circuit simulated is that of Figure 3-4.

The simulation requires numerical values for the various components of the equivalent circuit. The following subsection justifies the values selected for use in the subsequent simulations.

DETERMINATION OF THE COMPONENT VALUES OF THE EQUIVALENT CIRCUIT OF THE ARMATURE WINDING

The component values were determined by use of data provided by the manufacturer of the selected motor [3-17]. It should be noted that this data is for the motor alone, without any other hardware fastened onto the rotor shaft. In the practical measurements made later in this work, the motor always had a gearbox and optical encoder mounted on this shaft. Further, for some of the real measurements, a turntable was added to increase the inertial load on the

motor. These fixtures will cause a difference to exist between the numerical values of Rotational Inertia, J, and Coefficient of Viscous Friction, B, calculated here, and those to be expected of the real system.

The main purpose of this section of the work is to justify the use of a simple, linear model of the PWM Unit. The validity of this justification is not lessened by the use of data for an unloaded motor.

The motor manufacturer provided the following data:

Rotor inertia, J = 63.5 mg.m.m

Armature resistance, R_a = 0.747 Ohms

Armature inductance, L_a = 0.23 mH

Torque constant, k_i = 36.5 mN.m/A

No load current, I_{a0} = 387 mA (18 Volts Supply)

No load speed, ω_0 = 4500 rpm (150π radians per second)

These figures enabled the no load back emf, E_{b0} to be found :

$$E_{b0} = V_a - I_{a0}.R_a = 17.7 \ Volts \qquad 3\text{-}26$$

The required value of the parallel resistance may be determined from the no-load information

$$R_m = \frac{E_{b0}}{I_{a0}} \approx 45.7 \ \Omega \qquad 3\text{-}27$$

Similarly the value of the Back-Emf Constant, k_b may be found

$$k_b = \frac{E_{b0}}{\omega_0} = \frac{17.7}{150.\pi} \approx 3.76.10^{-2} \qquad 3\text{-}28$$

Neglecting the mechanical and iron losses of the motor enables the motor input and output powers to be equated

$$e_b(t).i_a(t) = T(t).\omega(t) \qquad 3\text{-}29$$

This equation shows that the constants k_i and k_b have the same value. Many authors [e.g. 3-18,3-19] simplify their analyses by the use of this approximation. Non-zero values for the above losses requires the input power to be greater than the output power. This inequality results in $k_b > k_i$. The value of k_b obtained from Equation 3-28 is indeed slightly greater than the datasheet value of k_i.

The no-load data may be used to calculate the value of the Coefficient of Viscous Friction, B. Since the no-load values are steady state values, then the armature is not subject to acceleration. The Torque Equation , Equation 3-8, reduces to :-

$$T(t) = B.\omega(t) \qquad 3\text{-}30$$

and hence

$$B = k_i.\frac{I_{a0}}{\omega_0} \approx 3.10^{-5}\ Nm.secs \qquad 3\text{-}31$$

The values obtained for k_b and B, provide a second means of calculating the value of the equivalent parallel resistance.

$$R_m = \frac{k_i.k_b}{B} = \frac{36.5.10^{-3}.3.756.10^{-2}}{3.10^{-5}} = 45.7\Omega \qquad 3\text{-}32$$

which is consistent with the value obtained from Equation 3-27.

Further the equivalent parallel capacitance may be found

$$C_m = \frac{J}{k_i k_b} = \frac{6.35.10^{-5}}{36.5.10^{-3}.3.756.10^{-2}} = 46.3mF \qquad 3\text{-}33$$

PSPICE SIMULATIONS OF THE EQUIVALENT ARMATURE CIRCUIT

The PSpice Circuit Simulation and Analysis software was used to examine the behaviour of the armature model shown in Figure 3-4. All of the PSpice simulations use the component values determined above. A typical source code file is provided in Appendix D.

The first PSpice simulation examined the step response of the armature model. The input signal being a step in applied armature voltage from 0 volts to +15 volts. Figure 3-6 provides plots of the resulting armature current and back-emf.

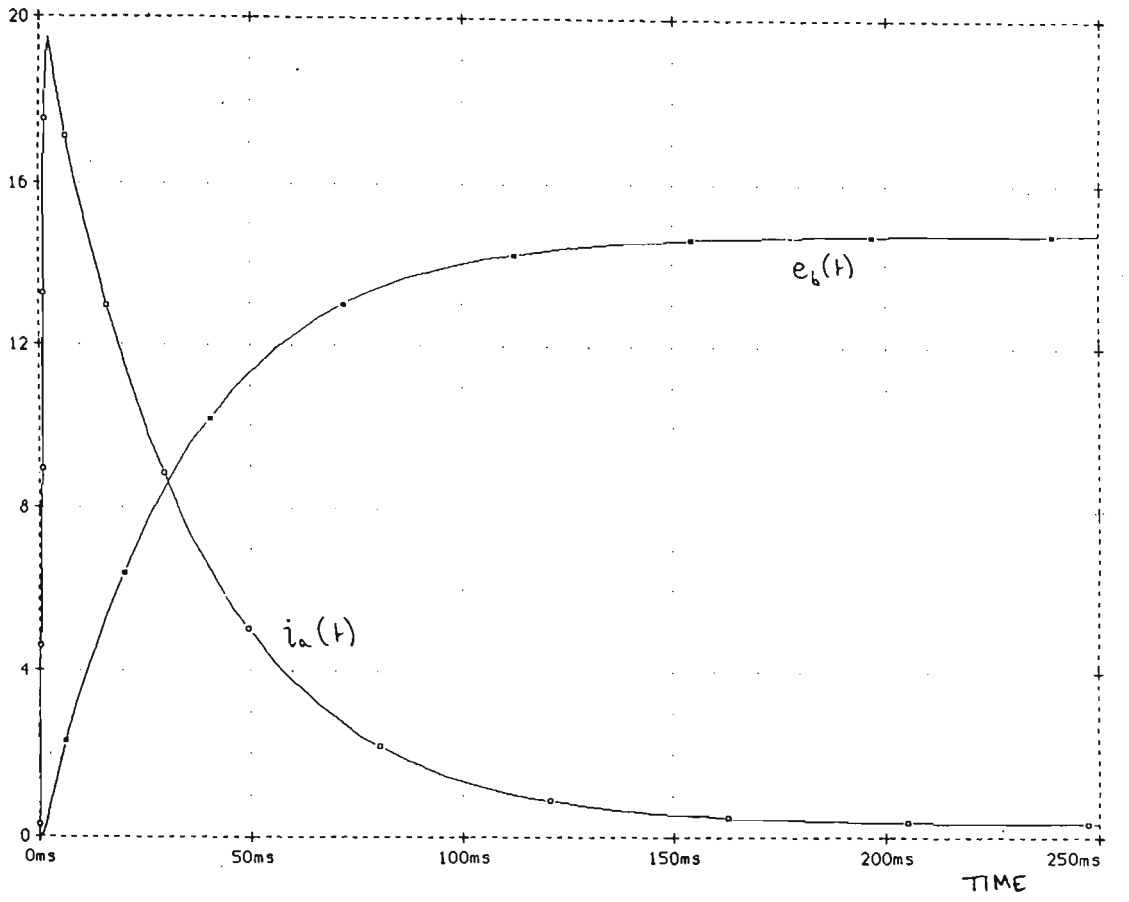

FIGURE 3-6 PSPICE SIMULATION OF THE ARMATURE CIRCUIT STEP RESPONSE

The mechanical time constant of the motor model was obtained from Figure 3-6 by measuring the time for the back-emf to change from 0% to 63% of its final value. This measurement suggests a time constant of approximately 36 milliseconds. This figure is in close agreement with that given by the motor manufacturer of 35.5 milliseconds [4-3].

Figure 3-7 shows the response of the armature model to the initial turn on of a 22kHz pulse width modulated voltage supply. The traces are for duty cycles of 50%, 60%, 70% and 90%. Figure 3-7 shows the initial 2 milliseconds of the transient, a period far shorter than the mechanical time constant of the motor. The traces therefore effectively show the build up of the starting current resulting from the PWM turning on, prior to the back-emf achieving its steady-state value.

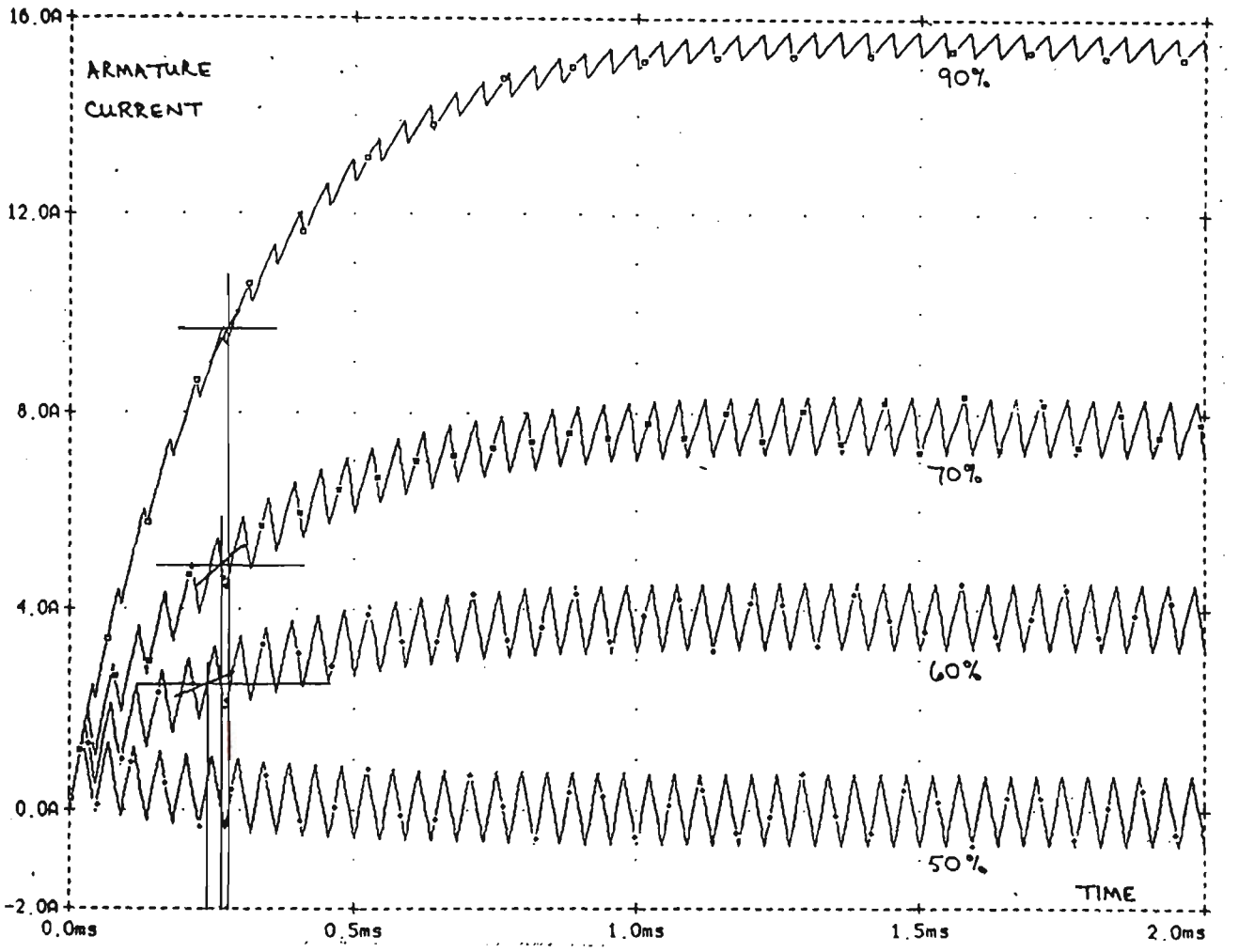

FIGURE 3-7 PSPICE SIMULATION OF THE ARMATURE TRANSIENT RESPONSE FOR THE START-UP OF THE PWM UNIT AT VARIOUS DUTY CYCLES

The traces of Figure 3-7 were used to measure the electrical time constant of the model of the armature winding. Each trace is treated as if it consists of a mean armature current, onto which a 22 kHz sawtooth disturbance has been superimposed. The electrical time constant is taken as the time taken for the mean current to reach 63% of its final value.

From Figure 3-7, three different values of the electrical time constant were obtained, corresponding to the traces for duty-cycles of 90%, 70% and 60%. The respective time constants being 0.28, 0.27 and 0.24 milliseconds. These values are slightly less than that expected using the motor datasheet [4-3]. The datasheet values indicate that the electrical time constant, τ_e, should have a value given by:

$$\tau_e = \frac{L_a}{R_a} \approx 0.31 \quad msecs \qquad 3\text{-}34$$

It is concluded that the average armature current changes at a maximum rate predominantly determined by the Electrical Time Constant of the armature winding.

Figure 3-8 is included to show that the simulations do subsequently show a fall in the armature current as the back-emf is developed. This graph shows the first 10 milliseconds of the transient for the 90% duty cycle.

A DISTURBANCE MODEL OF THE ARMATURE CURRENT

The simple, linear model of the PWM Unit that is presently being justified uses the above idea of the armature current being composed of an average value onto which a sawtooth disturbance waveform has been added. The disturbance waveform is an high frequency signal, formed from a 22 kHz fundamental and its harmonics. This disturbance signal is injected into a system which acts as a low pass filter. Consider the effect of the PWM duty cycle on the spectral content of this disturbance signal.

Analysis shows that the disturbance signal is most significant for a duty-cycle of 50%. It is subsequently shown that even for this worst-case duty-cycle the effect of the disturbance

signal is negligible.

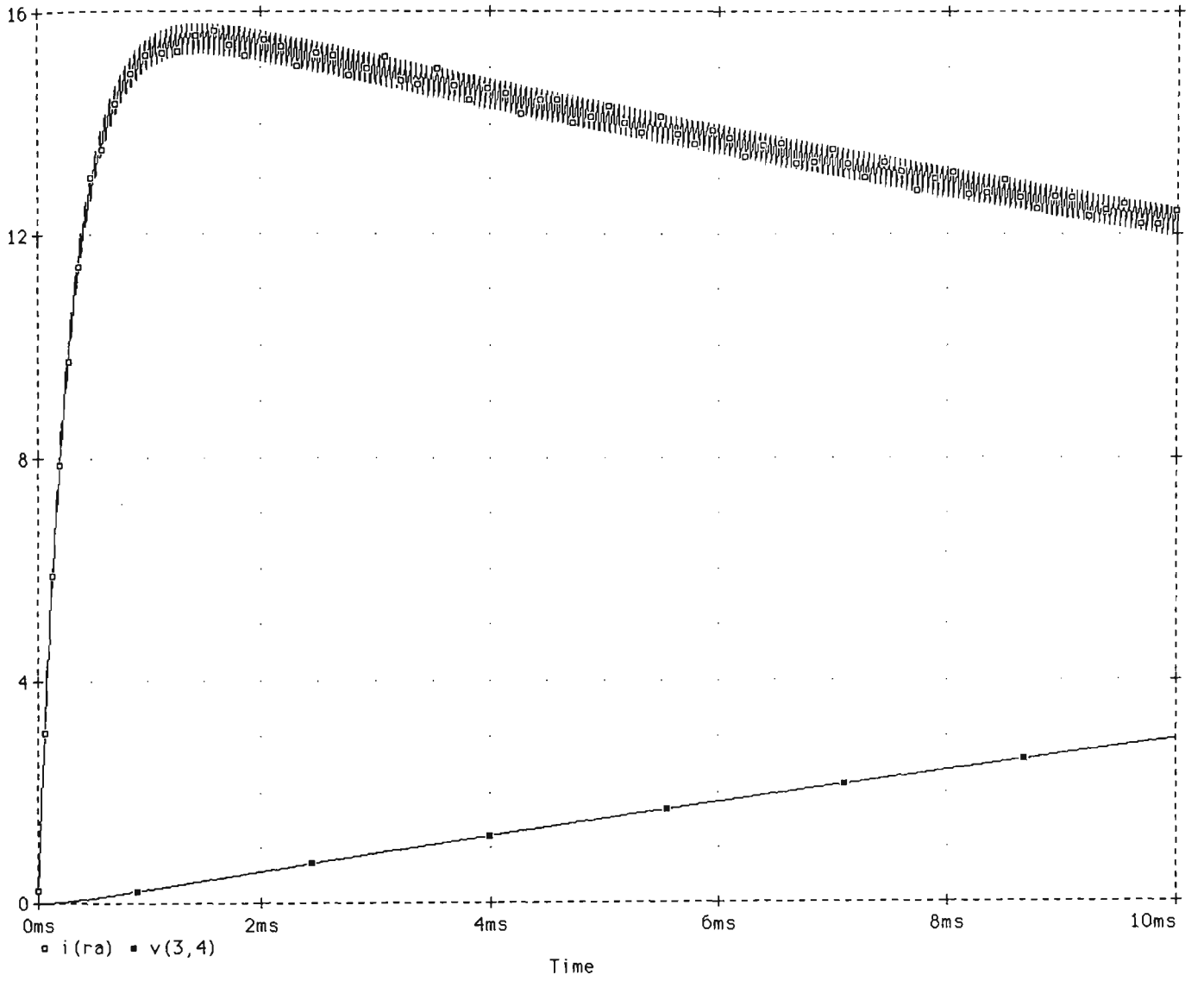

FIGURE 3-8 PSPICE SIMULATION OF THE ARMATURE TRANSIENT RESPONSE FOR PWM UNIT START-UP WITH A 90% DUTY CYCLE

Figure 3-9 shows a block diagram of the means of modelling the PWM Unit. The applied armature voltage $v_a(t)$, was considered to equal the mean value of the +/- 15 volt wavetrain from the PWM Unit. The 22 kHz switching frequency was modelled by an appropriately shaped, injected disturbance signal, $i_d(t)$.

The PWM Unit was thus modelled as a simple, gain block, k_d, where:

$$k_d = \frac{v_a(t)}{v_i(t)} \qquad 3\text{-}35$$

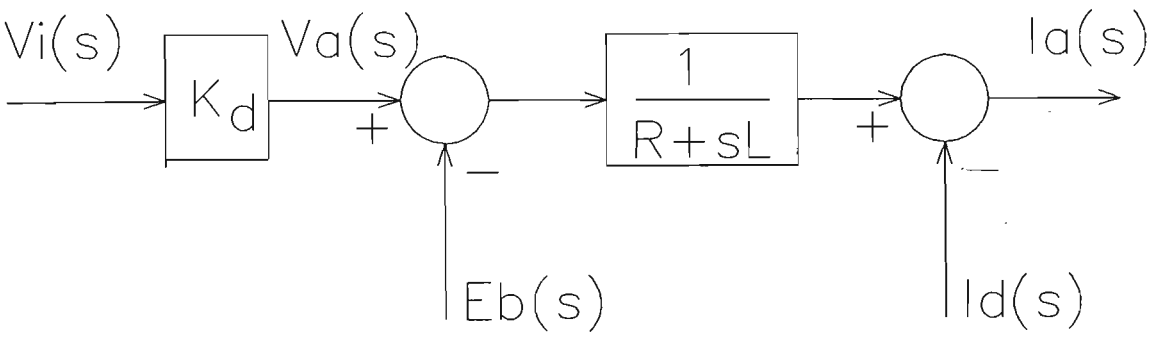

FIGURE 3-9 BLOCK DIAGRAM OF PWM UNIT MODEL WITH INJECTED ARMATURE DISTURBANCE CURRENT

SPECTRAL CONTENT OF THE DISTURBANCE SIGNAL AS A FUNCTION OF PWM DUTY-CYCLE

The duty-cycle of the PWM Unit affected both the amplitude and the waveshape of the disturbance signal. This section identifies the 50% duty-cycle as being that at which the disturbance signal was most significant. At this duty-cycle the amplitude of the disturbance signal was at its largest value, this is in agreement with the traces of Figure 3-7.

The same PSpice simulations were used to investigate the spectral content of the armature current at different PWM duty-cycles. The PSpice package permits the comparison of

the power in the fundamental and first eight harmonics of a periodic waveform.

These results were obtained from the transient simulations presented in Figure 3-7, for the last cycle of the disturbance signal within the period covered by the simulation. Accordingly it is acknowledged that the results are not true steady-state values, since these transient simulations terminate prior to the motor reaching its steady-state speed. The results are however, taken from a period well after the armature current has reached its maximum value.

The selection of this period as the basis of the Fourier Analysis was made so as to avoid the need for a number of protracted transient analyses. The results of the Fourier Analyses are presented in Table 3-2.

The servosystem acts as a low pass filter. Accordingly the higher frequency harmonics receive a greater attenuation than either the fundamental or low frequency harmonics. Table 3-2 shows, for the duty-cycles considered, that the fundamental carries most of the power of the disturbance signal. Table 3-2 also shows that as the duty-cycle is increased (i.e. moved further away from its base value of 50%) that the power at the fundamental frequency falls. These results are consistent with the earlier observation from Figure 3-7, that the disturbance signal has its largest amplitude for a duty-cycle of 50%. Accordingly it is claimed that the 50% duty-cycle is the worst case in terms of significance of the disturbance signal.

TABLE 3-2 EFFECT OF PWM DUTY-CYCLE UPON THE RELATIVE POWER OF THE SWITCHING FREQUENCY FUNDAMENTAL AND ITS HARMONICS

Frequency Component	Duty Cycle			
	50%	60%	70%	90%
22 kHz Fundamental	0.5984	0.5682	0.4817	0.1797
2nd Harmonic	0.0000	0.0874	0.1415	0.0850
3rd Harmonic	0.0651	0.0378	0.0209	0.0507
4th Harmonic	0.0000	0.0342	0.0206	0.0329
5th Harmonic	0.0219	0.0002	0.0217	0.0214
6th Harmonic	0.0000	0.0139	0.0089	0.0138
7th Harmonic	0.0106	0.0062	0.0030	0.0082
8th Harmonic	0.0000	0.0043	0.0069	0.0044
9th Harmonic	0.0056	0.0052	0.0044	0.0019

THE INFLUENCE OF THE WORST CASE DISTURBANCE SIGNAL UPON MOTOR SPEED

Figure 3-7 shows that for a 50% duty-cycle, the disturbance signal has a peak-to-peak value of approximately one Amp. The significance of this signal may be tested by replacing it by a sinewave disturbance signal of the same amplitude and periodic time. Consider then that the disturbance signal, $i_d(t)$, injected into the system is a 22 kHz sinewave with a crest value of 0.5 Amps.

To investigate the effect of this disturbance upon the motor speed, the input signal should be adjusted until the mean applied armature voltage, i.e. $v_a(t)$ of Figure 3-9, is zero.

The system may then be redrawn with the disturbance signal as the input signal, and the motor speed as the output, as in Figure 3-10.

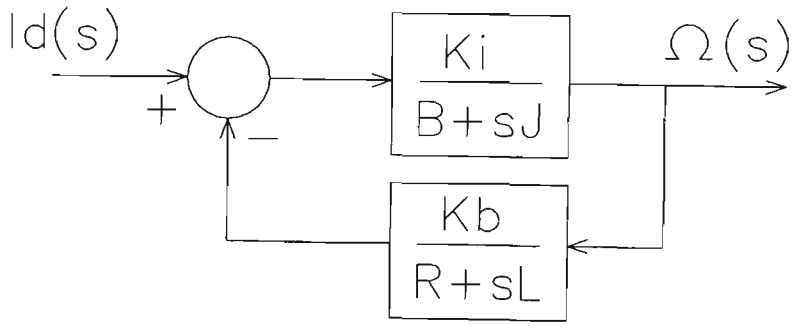

FIGURE 3-10 S-DOMAIN MODEL USED TO EXAMINE THE EFFECT OF PWM SWITCHING UPON THE MOTOR SPEED

The closed loop transfer function of this system is given by

$$\frac{\Omega(s)}{I_d(s)} = \frac{\frac{k_i}{J}[s+\frac{R}{L}]}{s^2+s[\frac{R}{L}+\frac{B}{J}]+[\frac{BR+k_i k_b}{LJ}]} \qquad 3\text{-}36$$

Using the values obtained above for all of these constants, this system may be shown to have a gain of -47.6dB for a 22kHz sinusoidal input current, $i_d(t)$. This represents in real terms a variation of 0.00416 radians per second per ampere, or approximately 0.04 rpm per ampere! The disturbance signal can therefore be neglected without introducing significant errors.

FINAL SIMPLE MODEL OF THE PWM CONTROLLED ARMATURE CURRENT

Accordingly the PWM circuit and armature winding will be modelled as shown in Figure 3-11.

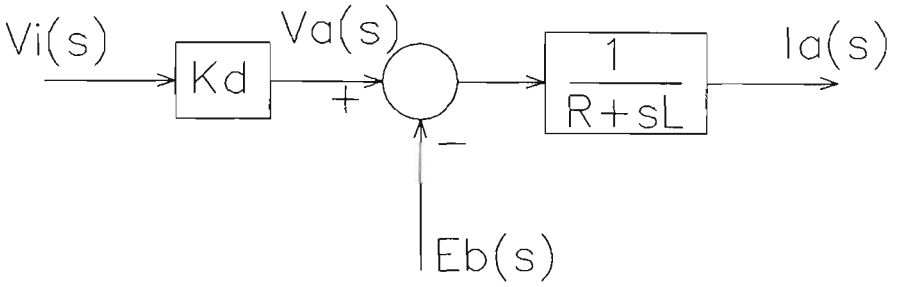

FIGURE 3-11 SIMPLIFIED S-DOMAIN MODEL OF THE PWM UNIT AND ARMATURE WINDING

With reference to Figure 3-3, the transfer function of the motor is taken as:

$$\frac{\Omega(s)}{V_a(s)} = \frac{\frac{k_i}{LJ}}{s^2+s[\frac{B}{J}+\frac{R}{L}]+[\frac{BR+k_ik_b}{LJ}]} \qquad 3\text{-}37$$

For the Maxon motor used on its own (i.e. without the gearbox) the poles of this transfer function are at 29.6 and 3218.4 radians per second. The low frequency pole is clearly dominant, and the above second order system may therefore be approximated to by a first order model with a time constant of 33.8 milliseconds.

$$\frac{\Omega(s)}{V_a(s)} = \frac{k_m}{s+29.6} = \frac{k_m}{s+p} \qquad 3\text{-}38$$

The overall transfer function of the position servomechanism , $G_p(s)$,will therefore be of the form:

$$G_p(s) = \frac{\Theta(s)}{V_i(s)} = \frac{k}{s(s+p)} \qquad 3\text{-}39$$

where

$$k = k_m.k_d \qquad 3\text{-}40$$

CHAPTER 4 PARAMETER ESTIMATION OF A DISCRETE-TIME MODEL OF THE SERVOSYSTEM

4-1 DEVELOPMENT OF A DISCRETE-TIME, INPUT-OUTPUT MODEL OF THE SERVOSYSTEM

THE NEED FOR A DISCRETE-TIME MODEL

The work so far has justified the modelling of the continuous time plant by Equation 3-39. This plant is to be controlled and monitored using a digital computer. The computer is used both to determine the input signal, $v_i(t)$, and to perform the parameter estimation of the servosystem. The plant input signal, $v_i(t)$ is a staircase waveform produced by a digital-to-analogue converter (DAC).

In this section a discrete-time model of the servosystem is developed. This model takes the sequence of values generated by the program as its input signal. This sequence is denoted as {u(kT)}. The value written out by the computer program at time t=kT being denoted by u(kT), where T is the periodic time with which the program writes to the output port. The output signal of the discrete-time model is a sequence of periodically sampled positions of the output shaft of the servosystem. The sampling of the continuous output signal $\theta(t)$, is timed so as to be synchronous with the generation of the sequence {u(kT)}. The sequence of sampled output values will be denoted by {y(kT)}, where

$$y(kT) = \theta(t)|_{t=kT} \qquad 4\text{-}1$$

The digital latch and DAC are modelled by a zero-order hold (ZOH). The ZOH is discussed in detail in most introductory textbooks on discrete-time control theory [e.g. 4-1,4-2,4-3].

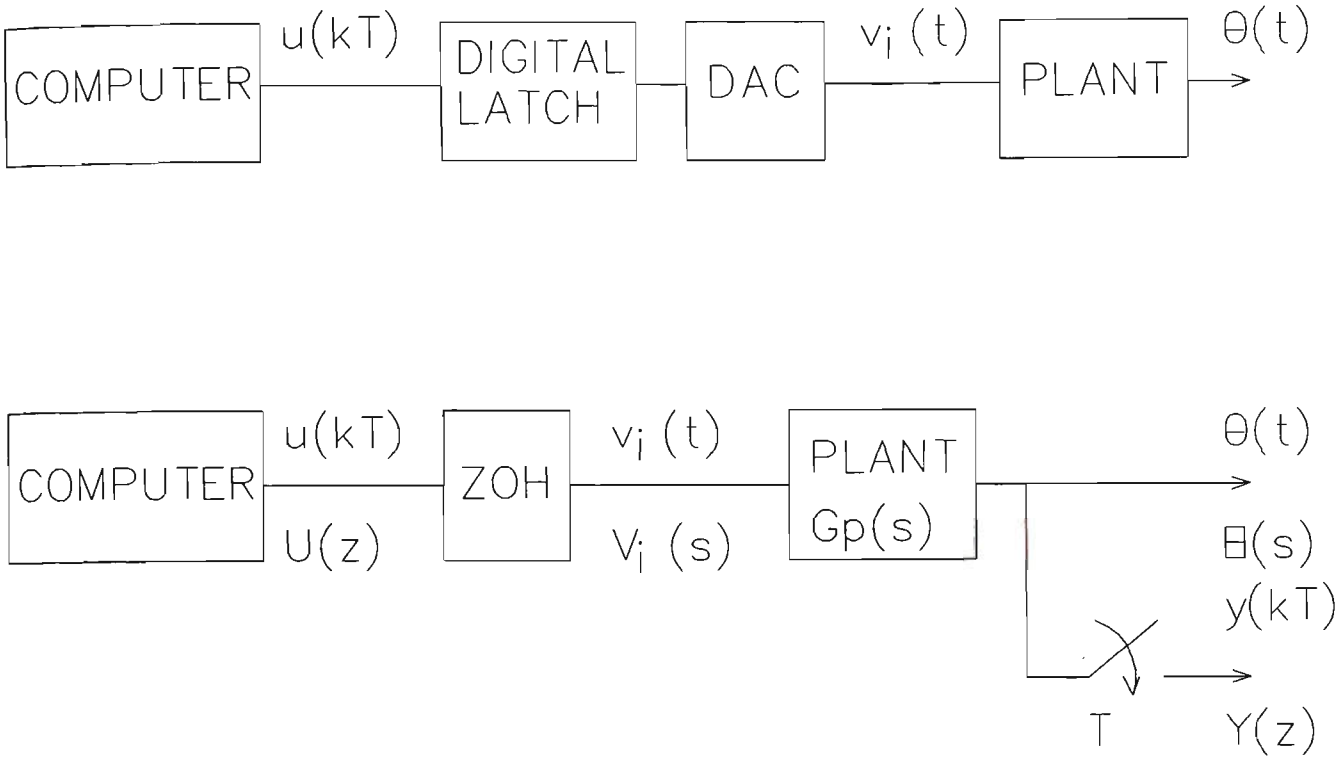

FIGURE 4-1 BLOCK DIAGRAMS OF THE COMPUTER CONTROLLED SERVOSYSTEM

Z-TRANSFORM NOTATION

The z-transform of any continuous time signal, e.g. a(t), will be denoted by A(z). The convention for representing the z-transform is that commonly found in the literature, and may be summarised by the following equations:

$$A(z) \neq A(s)\big|_{s=z} \qquad 4\text{-}2$$

where

$$A(s) = \mathcal{L}\{a(t)\}$$

$$\mathcal{Z}[a(t)] = A(z) \qquad 4\text{-}3$$

$$\mathfrak{Z}[a(t)] = \mathfrak{Z}[a(kT)] \qquad 4\text{-}4$$

$$\mathfrak{Z}[A(s)] = \mathfrak{Z}[a(t)] \qquad 4\text{-}5$$

DETERMINATION OF THE OVERALL DISCRETE TIME INPUT-OUTPUT MODEL OF THE SYSTEM

In the following subsection the overall system will be described by an Input-Output Model. Two related forms of this model are given:

i) the Discrete Transfer Function of the system

and ii) the Difference Equation relating elements of the sequences {u(kT)} and {y(kT)}.

Since the transfer function of the plant, Gp(s), is known, as per Equation 3-39, then the overall Discrete Transfer Function, G(z), may be determined, where:

$$G(z) = \frac{Y(z)}{U(z)} \qquad 4\text{-}6$$

G(z) may be found in terms of T (the periodic time of the discrete-time signals), and k and p (as defined for Equation 3-39).

Phillips and Harbor [4-1] provide an elegant method for determining the discrete transfer function , G(z). Their method considers the sequence {u(kT)} to be the same as that of a sampled unit step. The action of the ZOH is to recover the unit step, and hence

$$V_i(s) = \frac{1}{s} \qquad 4\text{-}7$$

The step response of the system, $\theta(t)$, may then be found from

$$\Theta(s) = V_i(s)G_p(s) = \frac{1}{s}G_p(s) \qquad 4\text{-}8$$

For the servosystem, using the model given in Equation 3-39, it can be shown that

$$Y(z) = \Theta(z) = \frac{kTz}{p(z^2-2z+1)} - \frac{kz}{p^2(z-1)} + \frac{kz}{p^2[z-\exp(-pT)]} \qquad 4\text{-}9$$

The discrete transfer function, G(z), may then be found from

$$G(z) = \frac{Y(z)}{U(z)} = \frac{\mathfrak{Z}[G_p(s).\frac{1}{s}]}{\mathfrak{Z}[\frac{1}{s}]} \qquad 4\text{-}10$$

Hence it can be shown that

$$G(z) = \frac{kT}{p(z-1)} - \frac{k}{p^2} + \frac{k(z-1)}{p^2(z-\alpha)} \qquad 4\text{-}11$$

where $\alpha = \exp(-pT)$

Alternatively this may be expressed in the form

$$\frac{Y(z)}{U(z)} = \frac{a_1.z+a_2}{z^2+b_1.z+b_2} \qquad 4\text{-}12$$

where

$$a_1 = \frac{k}{p^2}.[pT-1+\exp(-pT)] \qquad 4\text{-}13$$

$$a_2 = \frac{k}{p^2}.[1-\exp(-pT)-pT.\exp(-pT)] \qquad 4\text{-}14$$

$$b_1 = -1-\exp(-pT) \qquad 4\text{-}15$$

and

$$b_2 = \exp(-pT) \qquad 4\text{-}16$$

The difference equation relating the sequences {u(kT)} and {y(kT)} may be derived from Equation 4-12. The difference equation is:

$$y(kT) + b_1y[(k-1)T] + b_2y[(k-2)T] = a_1u[(k-1)T] + a_2u[(k-2)T] \qquad 4\text{-}17$$

This equation is known as an Auto-Regressive, Moving Average (ARMA) Model of the system. Coefficients b_1 and b_2 are referred to as the auto-regressive parameters of the model, a_1 and a_2 are the moving average parameters.

In the following work a more compact notation is adopted, whereby Equation 4-17 may be rewritten as:

$$y_0 + b_1y_1 + b_2y_2 = a_1u_1 + a_2u_2 \qquad 4\text{-}18$$

i.e. x_n denotes the sample of the sequence {x(kT)} at the time t, where t=(k-n)T.

4-2 THE LEAST SQUARES METHOD OF PARAMETER ESTIMATION

The Least Squares Method of Parameter Estimation considered in this thesis is a means by which the coefficients a_1, a_2, b_1 and b_2 may be determined using knowledge of the sequences {u(kT)} and {y(kT)}.

Equation 4-18 may be used as a One-Step Ahead Predictor of the output signal of the system. Using a "carat" to denote predicted values, Equation 4-18 may be rewritten as

$$\hat{y}_0 = a_1 u_1 + a_2 u_2 - b_1 y_1 - b_2 y_2 \qquad 4\text{-}19$$

If the values of the input and output sequences are known for the instants t = (k-1)T and t = (k-2)T, then Equation 4-19 may be used to calculate the subsequent output value y(kT). In practice there are two major problems:

i) the exact values of the coefficients are not known. Instead the best available estimates of these coefficients must be used. These will also be denoted by the use of a "carat", as in Equation 4-21 below.

ii) all of the measured signals include noise and measurement errors.

Equation 4-19 cannot therefore be expected to yield the correct value of y(kT). At time t = kT the true value of y(kT) becomes available for measurement. The difference between the measured and predicted values is the Error in Prediction, ϵ_0.

$$\epsilon_0 = y_0 - \hat{y}_0 \qquad 4\text{-}20$$

Alternatively the Error in Prediction may be expressed as

$$\epsilon_0 = y_0 + \hat{b}_1 y_1 + \hat{b}_2 y_2 - \hat{a}_1 u_1 - \hat{a}_2 u_2 \qquad 4\text{-}21$$

In the Method of Least Squares [4-4] the estimates of the coefficients are adjusted so as to minimise the Performance Index, J_n, where

$$J_n = \sum_{i=0}^{n} \epsilon_i^2 \qquad 4\text{-}22$$

The Performance Index is minimised by finding the solution of the following set of

simultaneous equations:

$$\frac{\partial J_n}{\partial \hat{a}_1} = 0, \quad \frac{\partial J_n}{\partial \hat{a}_2} = 0, \quad \frac{\partial J_n}{\partial \hat{b}_1} = 0, \quad \frac{\partial J_n}{\partial \hat{b}_2} = 0 \qquad 4\text{-}23$$

These equations are known as the "Least Square" or "Normal" Equations.

Using the notation defined at the start of this section, and omitting the limits of the summations, these partial derivatives may be expressed as

$$\frac{\partial J_n}{\partial \hat{a}_1} = 2\hat{a}_1 \sum u_1^2 - 2\sum u_1 y_0 - 2\hat{b}_1 \sum u_1 y_1 - 2\hat{b}_2 \sum u_1 y_2 + 2\hat{a}_2 \sum u_1 u_2 \qquad 4\text{-}24$$

$$\frac{\partial J_n}{\partial \hat{a}_2} = 2\hat{a}_2 \sum u_2^2 - 2\sum u_2 y_0 - 2\hat{b}_1 \sum u_2 y_1 - 2\hat{b}_2 \sum u_2 y_2 + 2\hat{a}_1 \sum u_1 u_2 \qquad 4\text{-}25$$

$$\frac{\partial J_n}{\partial \hat{b}_1} = 2\hat{b}_1 \sum y_1^2 + 2\sum y_0 y_1 + 2\hat{b}_2 \sum y_1 y_2 - 2\hat{a}_1 \sum u_1 y_1 - 2\hat{a}_2 \sum u_2 y_1 \qquad 4\text{-}26$$

$$\frac{\partial J_n}{\partial \hat{b}_2} = 2\hat{b}_2 \sum y_2^2 + 2\sum y_0 y_2 + 2\hat{b}_1 \sum y_1 y_2 - 2\hat{a}_1 \sum u_1 y_2 - 2\hat{a}_2 \sum u_2 y_2 \qquad 4\text{-}27$$

Hence the Least Squares Equations may be expressed as

$$\begin{bmatrix} \Sigma u_1 y_0 \\ \Sigma u_2 y_0 \\ -\Sigma y_0 y_1 \\ -\Sigma y_0 y_2 \end{bmatrix} = \begin{bmatrix} \Sigma u_1^2 & \Sigma u_1 u_2 & -\Sigma u_1 y_1 & -\Sigma u_1 y_2 \\ \Sigma u_1 u_2 & \Sigma u_2^2 & -\Sigma u_2 y_1 & -\Sigma u_2 y_2 \\ -\Sigma u_1 y_1 & -\Sigma u_2 y_1 & \Sigma y_1^2 & \Sigma y_1 y_2 \\ -\Sigma u_1 y_2 & -\Sigma u_2 y_2 & \Sigma y_1 y_2 & \Sigma y_2^2 \end{bmatrix} \cdot \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{b}_1 \\ \hat{b}_2 \end{bmatrix} \qquad 4\text{-}28$$

In this form the equations are basically the same as those used by Kalman [1-1] in his earlier work. Kalman however included a time "weighting function ", w(kT), in order to give greater weight to the more recent measurements. Thus Kalman replaced the matrix of correlation functions by one whose elements he referred to as " pseudo-correlation functions". This enabled his estimator to deal with a slowly varying system. Kalman solved the resulting

set of equations directly using the Gauss-Seidel Method [1-1].

4-3 A NON-RECURSIVE SOLUTION OF THE LEAST SQUARES EQUATIONS

The Least Squares Equations, given above as Equation 4-23, are a set of four simultaneous equations in four unknowns. The four unknowns are the coefficients or parameters of the system that are to be estimated. These simultaneous equations are presented in matrix form as Equation 4-28, which is now rewritten as Equation 4-29.

$$\underline{b} = \underline{R}.\underline{\theta} \tag{4-29}$$

where

$\underline{R}$ is a symmetrical, 4x4 matrix, known as the Information Matrix

$\underline{\theta}$ is the 4x1 Parameter Vector, whose elements are the coefficients to be estimated

$\underline{b}$ is a 4x1 column vector, given by:

$$\underline{b}^T = [\Sigma u_1 y_0 \;\; \Sigma u_2 y_0 \;\; -\Sigma y_0 y_1 \;\; -\Sigma y_0 y_2] \tag{4-30}$$

One method of solving Equation 4-29 is by Lower-Upper (LU) Factorisation [4-5] of the Information Matrix, i.e.

$$\underline{R} = \underline{L}.\underline{U} \tag{4-31}$$

Where the elements of the Lower Diagonal and Upper Diagonal Matrices are chosen such that:

$$\underline{R} = \begin{bmatrix} r_{00} & r_{01} & r_{02} & r_{03} \\ r_{10} & r_{11} & r_{12} & r_{13} \\ r_{20} & r_{21} & r_{22} & r_{23} \\ r_{30} & r_{31} & r_{32} & r_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{10} & 1 & 0 & 0 \\ l_{20} & l_{21} & 1 & 0 \\ l_{30} & l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{00} & u_{01} & u_{02} & u_{03} \\ 0 & u_{11} & u_{12} & u_{13} \\ 0 & 0 & u_{22} & u_{23} \\ 0 & 0 & 0 & u_{33} \end{bmatrix} \tag{4-32}$$

From Equation 4-29 the Least Squares Equations can then be rewritten as

$$\underline{L}.\underline{U}.\underline{\theta} = \underline{b} \tag{4-33}$$

Pre-multiplying by the inverse of $\underline{L}$ results in

$$\underline{L}^{-1}.\underline{L}.\underline{U}.\underline{\theta} = \underline{L}^{-1}.\underline{b} = \underline{c} \qquad 4\text{-}34$$

where

$$\underline{c} = \underline{U}.\underline{\theta} \qquad 4\text{-}35$$

and since

$$\underline{L}.\underline{L}^{-1}.\underline{b} = \underline{L}.\underline{c} \qquad 4\text{-}36$$

then

$$\underline{b} = \underline{L}.\underline{c} \qquad 4\text{-}37$$

This equation may be used to find the elements of the vector $\underline{c}$.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ l_{10} & 1 & 0 & 0 \\ l_{20} & l_{21} & 1 & 0 \\ l_{30} & l_{31} & l_{32} & 1 \end{bmatrix} \cdot \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} r_{04} \\ r_{14} \\ r_{24} \\ r_{34} \end{bmatrix} \qquad 4\text{-}38$$

Where the last column vector results from expressing the original least squares equations in the form of a partioned matrix $[\underline{R} : \underline{b}]$.

Once the elements of $\underline{c}$ have been calculated, then the equation :

$$\underline{U}.\underline{\theta} = \underline{c} \qquad 4\text{-}39$$

may be used to solve for the parameter estimates, $\underline{\theta}$, as per Equation 4-40.

$$\begin{bmatrix} u_{00} & u_{01} & u_{02} & u_{03} \\ 0 & u_{11} & u_{12} & u_{13} \\ 0 & 0 & u_{22} & u_{23} \\ 0 & 0 & 0 & u_{33} \end{bmatrix} \cdot \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} = \begin{bmatrix} r_{00} & r_{01} & r_{02} & r_{03} \\ 0 & u_{11} & u_{12} & u_{13} \\ 0 & 0 & u_{22} & u_{23} \\ 0 & 0 & 0 & u_{33} \end{bmatrix} \cdot \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \qquad 4\text{-}40$$

Appendix C contains a list of the sequence of equations required to use LU Factorisation to solve a set of four equations in four unknowns. Appendix E contains the source code of a program performing Parameter Estimation, using this method.

4-4 A RECURSIVE SOLUTION TO THE LEAST SQUARES EQUATIONS

THE NEED FOR RECURSIVE ALGORITHMS

In practical self-tuning controllers it is necessary for the parameter estimation to be performed in real-time. The time available for the required calculations will depend upon the sampling rate of the controller, which in turn will depend upon the speed of response of the system to be controlled. The required calculation effort will depend upon the complexity of the model used. For linear models the complexity is determined by the order of the model. The adoption of recursive solutions of the Least Squares Equations is one means of reducing the calculation time required.

It was found that, for the low order models used in this thesis, that recursive solutions do not offer a significant advantage in the reduction of calculation burden.

YOUNG'S APPROACH TO A RECURSIVE ALGORITHM

Young [4-6] shows how to develop a recursive solution of the Least Squares Equations. The advantage of his approach is that it requires only the simple manipulation of matrices, and avoids the need to invoke the Matrix Inversion Lemma [4-7].

Starting with the least squares equations written in the form:

$$\underline{R}_k.\underline{\theta}_k = \underline{b}_k \qquad 4\text{-}41$$

i.e. as per Equation 4-29, but now with a subscript indicating that these terms should have their elements updated to contain the signals and estimates available immediately after the instant t = kT seconds. It can be seen that

$$\underline{\theta}_k = \underline{R}_k^{-1}.\underline{b}_k = \underline{P}_k.\underline{b}_k \qquad 4\text{-}42$$

where the inverse of the Information Matrix, the Error Covariance Matrix is denoted by $\underline{P}_k$. The recursive solution requires a means whereby this "P" matrix may be updated at each sampling instant. This updating of the P-Matrix is considered next.

A data vector, $\underline{\Psi}_k$, is now introduced. This vector is defined such that:

$$\underline{\Psi}_k^T = [u_1 \; u_2 \; -y_1 \; -y_2] \tag{4-43}$$

It should be noted that all the elements of $\underline{\Psi}_k$ are known prior to the sampling time at t = kT.

The information matrix, $\underline{R}_k$, is updated using the equation:

$$\underline{R}_k = \underline{R}_{k-1} + \underline{\Psi}_k \cdot \underline{\Psi}_k^T \tag{4-44}$$

All of the elements of $\underline{R}_k$ are therefore available prior to t = kT.

The inverse of the P-matrix (i.e. the inverse of the Error Covariance Matrix) is updated by:

$$\underline{P}_k^{-1} = \underline{P}_{k-1}^{-1} + \underline{\Psi}_k \cdot \underline{\Psi}_k^T \tag{4-45}$$

By premultiplying this equation by $\underline{P}_k$, and then postmultiplying by $\underline{P}_{k-1}$ the following equation is obtained:

$$\underline{P}_{k-1} = \underline{P}_k + \underline{P}_k \cdot \underline{\Psi}_k \cdot \underline{\Psi}_k^T \cdot \underline{P}_{k-1} \tag{4-46}$$

Then by postmultiplying by $\underline{\Psi}_k$

$$\underline{P}_{k-1} \cdot \underline{\Psi}_k = \underline{P}_k \cdot \underline{\Psi}_k + \underline{P}_k \cdot \underline{\Psi}_k \cdot \underline{\Psi}_k^T \cdot \underline{P}_{k-1} \cdot \underline{\Psi}_k \tag{4-47}$$

The equations may be made less cumbersome by the definition of the vector, $\underline{W}_k$ and scalar D, where

$$\underline{W}_k = \underline{P}_{k-1} \cdot \underline{\Psi}_k \tag{4-48}$$

and

$$D = 1 + \underline{\Psi}_k^T \cdot \underline{P}_{k-1} \cdot \underline{\Psi}_k \tag{4-49}$$

As yet this derivation does not indicate how to determine the contents of $\underline{P}_{k-1}$, however, from Equations 4-43,4-48 and 4-49 it is apparent that sufficient measurements are

available at t = (k-1)T to enable both $\underline{W}_k$ and D to be calculated.

Combining equations 4-47 and 4-49 it can be seen that:

$$\underline{P}_{k-1}.\underline{\Psi}_k = \underline{P}_k.\underline{\Psi}_k.D \qquad 4\text{-}50$$

Postmultiplying by $D^{-1}.\underline{\Psi}_k^T.\underline{P}_{k-1}$, results in

$$\underline{P}_{k-1}.\underline{\Psi}_k.D^{-1}.\underline{\Psi}_k^T.\underline{P}_{k-1} = \underline{P}_k.\underline{\Psi}_k.\underline{\Psi}_k^T.\underline{P}_{k-1} \qquad 4\text{-}51$$

The right-hand side of Equation 4-51 may be expressed as the change required in the Covariance Matrix, as per Equation 4-46. Equation 4-51 may thus be rewritten as:

$$\underline{W}_k.D^{-1}.\underline{\Psi}_k^T.\underline{P}_{k-1} = \underline{P}_{k-1} - \underline{P}_k \qquad 4\text{-}52$$

Provided $\underline{P}_{k-1}$ is known this equation permits the P-matrix to be updated using information made available at time t = (k-1)T. i.e.

$$\underline{P}_k = \underline{P}_{k-1} - \underline{W}_k.D^{-1}.\underline{\Psi}_k^T.\underline{P}_{k-1} \qquad 4\text{-}53$$

The above derivation shows how the P-matrix may be updated. The expression for $\underline{P}_k$, as per Equation 4-53 may be substituted into Equation 4-42 in order to obtain the latest parameter estimates:

$$\underline{\theta}_k = \underline{P}_{k-1}.\underline{b}_k - \underline{W}_k.D^{-1}.\underline{\Psi}_k^T.\underline{P}_{k-1}.\underline{b}_k \qquad 4\text{-}54$$

where $\underline{b}_k$ may be updated by

$$\underline{b}_k = \underline{b}_{k-1} + \underline{\Psi}_k.y_0 \qquad 4\text{-}55$$

and hence

$$\underline{\theta}_k = \underline{P}_{k-1}.[\underline{b}_{k-1} + \underline{\Psi}_k.y_0] - \underline{W}_k.D^{-1}.\underline{\Psi}_k^T.\underline{P}_{k-1}.[\underline{b}_{k-1} + \underline{\Psi}_k.y_0] \qquad 4\text{-}56$$

This equation enables the estimates to be updated once y_0 has been measured at time t = kT. Next it will be shown how to make this update process recursive. By inspection of Equation 4-42 it can be seen that

$$\underline{\theta}_{k-1} = \underline{P}_{k-1}.\underline{b}_{k-1} \qquad 4\text{-}57$$

and hence Equation 4-56 may be rewritten as:

$$\underline{\theta}_k = \underline{\theta}_{k-1} + \underline{W}_k.D^{-1}[D.y_0 - \underline{\psi}_k^T.\underline{\theta}_{k-1} - \underline{\psi}_k^T.\underline{P}_{k-1}.\underline{\psi}_k.y_0] \qquad 4\text{-}58$$

Combining Equations 4-49 and 4-58 results in:

$$\underline{\theta}_k = \underline{\theta}_{k-1} + \underline{W}_k.D^{-1}.[y_0 - \underline{\psi}_k^T.\underline{\theta}_{k-1}] \qquad 4\text{-}59$$

Since the P-matrix remains symmetrical at all times, Equation 4-48 may be rewritten as

$$\underline{W}_k^T = \underline{\psi}_k^T.\underline{P}_{k-1} \qquad 4\text{-}60$$

Consequently the scalar D may be found from:

$$D = 1 + \underline{W}_k^T.\underline{\psi}_k \qquad 4\text{-}61$$

and from Equation 4-53 the P-matrix may be updated by:

$$\underline{P}_k = \underline{P}_{k-1} - D^{-1}.\underline{W}_k.\underline{W}_k^T \qquad 4\text{-}62$$

The RLS algorithm used to explore the performance of this method may be summarised as follows:

i) Calculate the Error in Prediction, $y_0 - \underline{\psi}_k^T.\underline{\theta}_{k-1}$

ii) Update $\underline{W}_k$, using Equation 4-48

iii) Calculate D using Equation 4-61

iv) Calculate $\underline{\theta}_k$, using Equation 4-59

v) Update $\underline{P}_k$, using Equation 4-62

To develop a practical RLS estimator based upon these equations it is necessary to initialise the values of both the P-matrix and the data and parameter vectors. The initial P-matrix used throughout this work consisted of an identity matrix of the required dimensions, multiplied by a scalar value of 10000. The elements of the data and parameter vectors were always initialised with zero values. In practice it would be sensible to initialise the parameter vector with the best possible 'guess' at the true values.

CHAPTER 5 COMPARISON OF DIFFERENT SOLUTIONS OF THE LEAST SQUARES EQUATIONS

5-1 REVIEW OF SOME ALTERNATIVE METHODS OF SOLUTION

In Section 4-2 it was shown that the Method of Least Squares requires the solution of a set of simultaneous equations. For the second order model (as per Equation 4-23), it is necessary to solve a set of four equations in four unknowns, i.e. one unknown for each parameter to be estimated.

Two methods of solution were selected, these solutions are outlined in Sections 4-3 and 4-4, and are known as Lower-Upper (LU) Factorisation and Recursive Least Squares (RLS) respectively. Their initial selection was based upon their low computation requirements, suggesting a short execution time suitable for high sampling rate, real-time parameter estimation. The selection of these two methods followed a brief literature review of alternative methods, and some programming to enable different direct methods of solution to be compared.

SOME DIRECT METHODS OF SOLUTION

In addition to LU Factorisation a number of other "direct" solutions of the Least Squares Equations were considered. These were:

Gaussian Elimination [4-5,5-1]

Gauss-Jordan Method [4-5,5-2]

Cramer's Rule [4-5]

Given a perfect computer these direct methods should all produce the same solution, in so much as they are merely different sequences of mathematical operations performed to solve the same set of Least Squares Equations. The initial selection of the preferred direct method did not consider the numerical properties of the different algorithms. The preferred, direct

method was therefore that with the shortest execution time.

CRAMER'S RULE, GAUSSIAN AND GAUSS-JORDAN ELIMINATION

Cramer's Rule is the most computationally tedious of the above direct methods [4-5,5-3], and hence was not considered further.

Gaussian Elimination permits a more rapid solution of a set of simultaneous equations than does the Gauss-Jordan Method [5-4,5-5].

LU FACTORISATION

A number of programs were written to compare the computation time required by the Gaussian Elimination, Gauss-Jordan, and LU Factorisation Methods. The comparisons were only made for the solution of four equations in four unknowns. The algorithm with the shortest execution time was that using LU Factorisation.

ITERATIVE METHODS OF SOLUTION

As an alternative to the "direct" methods, there are a number of iterative methods, e.g. Jacobi Iteration and the Gauss-Seidel Method [5-2,5-3]. These methods are useful for problems with a large number of equations, however for four equations in four unknowns Jacobi Iteration requires more computation time than does Gaussian Elimination [5-6]. Examination of Kalman's use of the Gauss-Seidel Method [1-1] showed this method to be no quicker than a direct solution.

5-2 BASES OF COMPARISON OF LU FACTORISATION AND RLS METHODS

FACTORS AFFECTING THE SELECTION OF THE PREFERRED METHOD

In comparing the two methods of solution it is necessary to identify the criteria by which their relative performances are to be judged. The following factors all have some influence upon the comparison:

i) the required computation (i.e. execution) times for each method

ii) the rates of convergence of the estimates

iii) the bias of the estimates (i.e. the difference between the true values of the parameters and the asymptotic values of the estimates)

iv) the sensitivity of the estimators to noise

v) the "richness" of the input signal (i.e. the ability of the selected input signal to provide adequate excitation of the plant for satisfactory system identification to be possible).

vi) the initial values given to the estimator variables, and especially the initial "guessed" values of the estimates.

vii) the sampling period of the estimator.

viii) the stability of the estimator.

THE SELECTION OF COMPUTER SIMULATION AS THE BASIS OF THE COMPARISON

There are three means whereby the different parameter estimators may be compared:

i) using a real system

ii) using an analogue computer to simulate a real system

iii) by a software simulation of a real system.

The first option, using a real system, has the major disadvantage that real systems have

non-linear behaviour. This option was therefore rejected, since any anomalies detected in the comparison may be attributable to the shortcomings of the mathematical model of the plant, rather than the Least Squares Algorithm being tested.

The second option, using an analogue computer, should result in the plant behaving linearly. However there still remains the problem of parameter uncertainty and drift as a consequence of the tolerances and drift of the components of the analogue circuitry.

It was therefore decided that the parameter estimation methods should be compared by means of software simulation. The programs used were coded in Occam2 [5-7,5-8]. Appendix E lists examples of the source code of the simulation programs. The simulations used an invariant, discrete-time system, such as that described by the ARMA model of Equation 4-18.

Selection of the following arbitary transfer function, $G_p(s)$, enabled the true parameters, $\underline{\theta}^T = [a_1\ a_2\ b_1\ b_2]$ to be calculated.

$$G_p(s) = \frac{0.5}{s.(s+1)} \qquad 5\text{-}1$$

The plant pole was selected so as to provide a normalised pole location of one radian per second. The true parameters were determined using Equations 4-13 to 4-16 inclusive. This is the major advantage of basing the comparison upon a computer simulation, since the estimates may then be compared with these "true" parameter values.

SOME COMMENTS ON THE SIMULATION PROGRAMS

The true parameters $\underline{\theta}^T$ were used to generate the next output signal of the simulated plant y_{0n}.

$$y_{0n} = a_1u_1 + a_2u_2 - b_1y_{1n} - b_2y_{2n} + n_0 \qquad 5\text{-}2$$

where n_0 is an element of the disturbance "noise" signal sequence {n(kT)}, superimposed onto the output signal sequence {y(kT)}. The output values have the added

subscript, n, to denote that these signals were contaminated by the previous elements of the sequence {n(kT)}.

The noise signal was produced from a random number generator. Use was made of a TDS library procedure [5-9] that used a pseudo-random binary sequence (PRBS) to generate a value in the range of zero to one. Subtraction of 0.5 from these values, gave a random number in the range of -0.5 to 0.5 (i.e. zero mean). This procedure resulted in the random sequence having a flat amplitude probability distribution over its range. Noise in real systems is expected to have a Normal (or Gaussian) Amplitude Distribution [5-10]. The simulation programs used the sum of twelve consecutive outputs of the above procedure to generate each element of {n(kT)}. This summation offered the following advantages:-

i) a better approximation to a Gaussian Amplitude Distribution

and ii) a greater range in the amplitude of the noise signal. A summation of twelve values results in a variance of 1.0 in this case.

The parameter estimators attempted to estimate $\underline{\theta}$ by the use of the estimated parameters $\hat{\underline{\theta}}^T = [\hat{a}_1\ \hat{a}_2\ \hat{b}_1\ \hat{b}_2]$, using the One-Step Ahead Predictor $\hat{y}_0$, such that:

$$\hat{y}_0 = \hat{a}_1 u_1 + \hat{a}_2 u_2 - \hat{b}_1 y_{1n} - \hat{b}_2 y_{2n} \qquad 5\text{-}3$$

It should be noted that in the simulations the noise sequence {n(kT)} was not only fed to the estimator, but was also coupled back via the Data Vector $\underline{\psi}_k$ into the simulated plant. The noise sequence thus acted as a disturbance signal superimposed on the output signal, rather than as measurement noise (i.e. random errors in measurement).

The simulations all used one thousand repeat runs for each combination of sampling rate, signal-to-noise Ratio (SNR), and input signal. Each run commenced with the estimates of the parameters set to zero (i.e. there was no assumption of prior knowledge of the parameter values). The different runs for a given set of sample rate, SNR, and input signal, differed only in the sections of the PRBS used in each.

At the end of each simulation of one thousand runs, for each sampling instant within the period covered by each run, there were one thousand estimates of each parameter produced by each estimator. The two different estimators being compared were provided with identical signal inputs.

The TDS Library does not provide routines to facilitate the plotting of graphical results [5-9]. The simulation programs were therefore designed to write their output to a DOS Text File. This file was then imported into a spreadsheet, which was used to generate the graphs used in the subsequent comparisons [5-17].

COMPUTATION TIME

The T800 Transputer enables parallel processes to be executed at one of two different priority levels. Processes executed with the high priority have access to a clock of one microsecond period. This clock was used to measure the execution times required by the various algorithms. The execution time of any block of code can thus be measured to a resolution of one microsecond.

RATE OF CONVERGENCE OF ESTIMATES

This characteristic was examined by plotting the decrease of the normalised Root Mean Square Errors (RMSE) of the estimates as the number of samples available to the estimator increased. The use of a simulation, and hence the availability of the true parameter values, enabled the RMSE to be determined, rather than merely the Standard Deviation of the estimates at each sampling instant. The RMSE is a preferred measure of estimate spread due to the expectation of bias in the estimates [4-4]. The results are presented in a normalised form, obtained by dividing the RMSE by the known true value of the relevant parameter.

As the number of samples available to an estimator increases, then the value of the estimate should reach some final value, the "Asymptotic Value" of the estimate [5-11]. The

difference between this Asymptotic Value and the true value of the parameter is the bias of the estimate. If the bias is zero, then the RMSE will equal the Standard Deviation of the estimate about the true value of the parameter.

The following points are made to aid in interpreting the results presented below as plots of percentage RMSE against time:

i) For a Normally Distributed Population it is well known [5-12] that 68.26% of the population will fall within one standard deviation from the population mean. Similarly 95.44% and 99.74% of the population will lie within two and three standard deviations from the mean, respectively.

ii) Hence an unbiassed estimate with a normalised percentage RMSE of X% would suggest the following confidence levels in that estimate:

68.26% probability of the estimate being within +/- X% of the true value

95.44% probability of the estimate being within +/- 2X% of the true value

99.74% probability of the estimate being within +/- 3X% of the true value.

Whether or not the estimators produce unbiassed estimates is the subject of the following:

BIAS OF THE ESTIMATES

Ideally the plots of Normalised RMSE against time should show the RMSE tending towards zero as the number of samples available to the estimator increases. i.e. after an adequately large number of samples the estimates should tend towards the true values of the parameters. In practice there is a non-zero, asymptotic value to which the RMSE converges. This non-zero value may result from two factors:

i) the estimates never settle exactly on the true value, but due to noise, fluctuate about the true value.

ii) bias in the estimates. i.e. the mean value of the estimates differs from the true

value of the parameter.

To detect any bias the mean of the estimates is compared with the true value. If the mean decreases monotonically towards the true value then the estimates are unbiassed. In Figures 5-5 and 5-6 plots are presented showing the difference between the mean of the estimate and the true value for each parameter. This difference or error was normalised by dividing the error by the true value of the parameter.

SENSITIVITY OF THE ESTIMATORS TO NOISE

The simulation programs were repeated for several different levels of added output noise. This was achieved by a simple, linear scaling of the n_0 term of Equation 5-2.

The Root Mean Square (RMS) value of the added noise was thus easily calculable. The resulting signal-to-noise Ratio (SNR) however was only determinable at the end of the last run of the simulation. The definition of SNR used here is given by:

$$SNR = 20\log_{10}\frac{RMS\ of\ noise\ free\ output}{RMS\ of\ added\ noise} \quad dB \qquad 5\text{-}4$$

In order to calculate the SNR the programs simulated two systems. The first of these systems was noise free, and was used solely to determine what the RMS value of the system output would be in the absence of noise.

RICHNESS OF THE INPUT SIGNAL

Two classes of input signal were used by the simulation programs:

i) A Stochastic Input Signal

The stochastic input signal was generated using one of the two random number generator procedures provided in the TDS Library [5-9]. Separate random number procedures were therefore available for generating both the noise sequence {n(kT)} and the stochastic input signal {u(kT)}. The use of separate PRBS procedures, using

different seed values and having different sequence lengths, ensured that there was no correlation between the stochastic input signal and the output noise.

ii) Test Input Signals

The simulations were run using a variety of test input signals, all of which were based upon rectangular pulses. These consisted of either single pulses or square wave pulse streams. Responses to a number of different pulse widths were considered.

INITIAL VALUES OF THE ESTIMATOR VARIABLES

In all of the simulation programs the estimates were initialised to have values of zero. There was no assumption of prior knowledge of the values of the unknown parameters.

In the case of the RLS Method, the P-Matrix was always initialised as an Identity Matrix scaled by a factor of 10000. No other scaling factors were investigated.

SAMPLING PERIOD OF THE ESTIMATOR

The literature of Adaptive Control Theory contains numerous "Rules of Thumb" for selecting an acceptable sampling rate. Many of these guidelines are based upon the closed loop bandwidth of the system [e.g. 5-13,5-14,5-15]. Alternative time constants of the system have been suggested as a basis for the selection of the sampling rate, e.g. the 95% Settling Time of the system [5-16].

In all simulations the plant was modelled with a real pole at $\omega_p = 1$ radian per second. Different sample periods were considered, selected so as to provide a range of ratios of sampling period to the time constant associated with this pole. The results presented here are for:

i) $T = 0.2$ seconds, considered to be a "standard" sampling rate

and ii) $T = 0.05$ seconds, considered to be an "high" sampling rate.

STABILITY OF THE ESTIMATOR

The stability of each estimator may be judged by examining the convergence of the estimates towards some asymptotic value. Loss of stability may not be noticed from the examination of the mean of the estimates, but should be clearly visible from the spread of the estimates.

5-3 THE COMPARISON OF LU FACTORISATION AND RLS METHODS USING A STOCHASTIC INPUT SIGNAL AT A STANDARD SAMPLING RATE

RESULTS OF THE SIMULATIONS

Figures 5-1 to 5-14 inclusive present some of the results of using a stochastic input signal to excite the system being identified. The results may be summarised as follows:

TABLE 5-1 SUMMARY OF PRESENTED RESULTS OF ESTIMATION OF ARMA COEFFICIENTS USING STOCHASTIC INPUT SIGNALS

FIGURE	PARAMETER	T (s)	Y-AXIS VARIABLE	No. of RUNS	RUN LENGTH	SNR (dB)
5-1	a1	0.2	%RMSE	1000	500T	47.4
5-2	a2	0.2	%RMSE	1000	500T	47.4
5-3	b1	0.2	%RMSE	1000	500T	47.4
5-4	b2	0.2	%RMSE	1000	500T	47.4
5-5	a1	0.2	%BIAS	1000	500T	47.4
5-6	b1	0.2	%BIAS	1000	500T	47.4
5-7	a1	0.2	%ERROR	1	500T	43.9
5-8	b1	0.2	%ERROR	1	500T	43.9
5-9	a1	0.2	%RMSE	1000	500T	53.4,45.4,39.4
5-10	b1	0.2	%RMSE	1000	500T	53.4,45.4,39.4

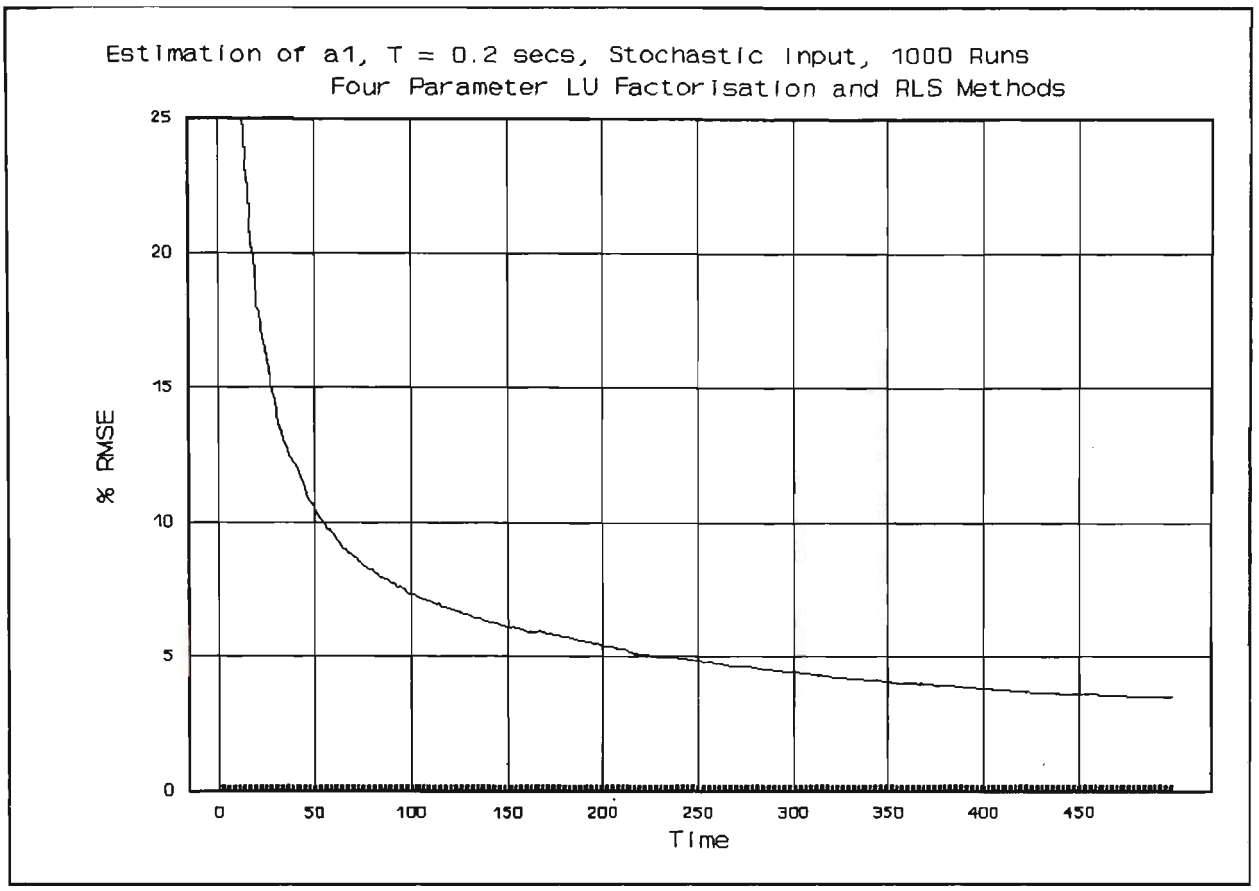

FIGURE 5-1 %RMSE IN ESTIMATION OF a1

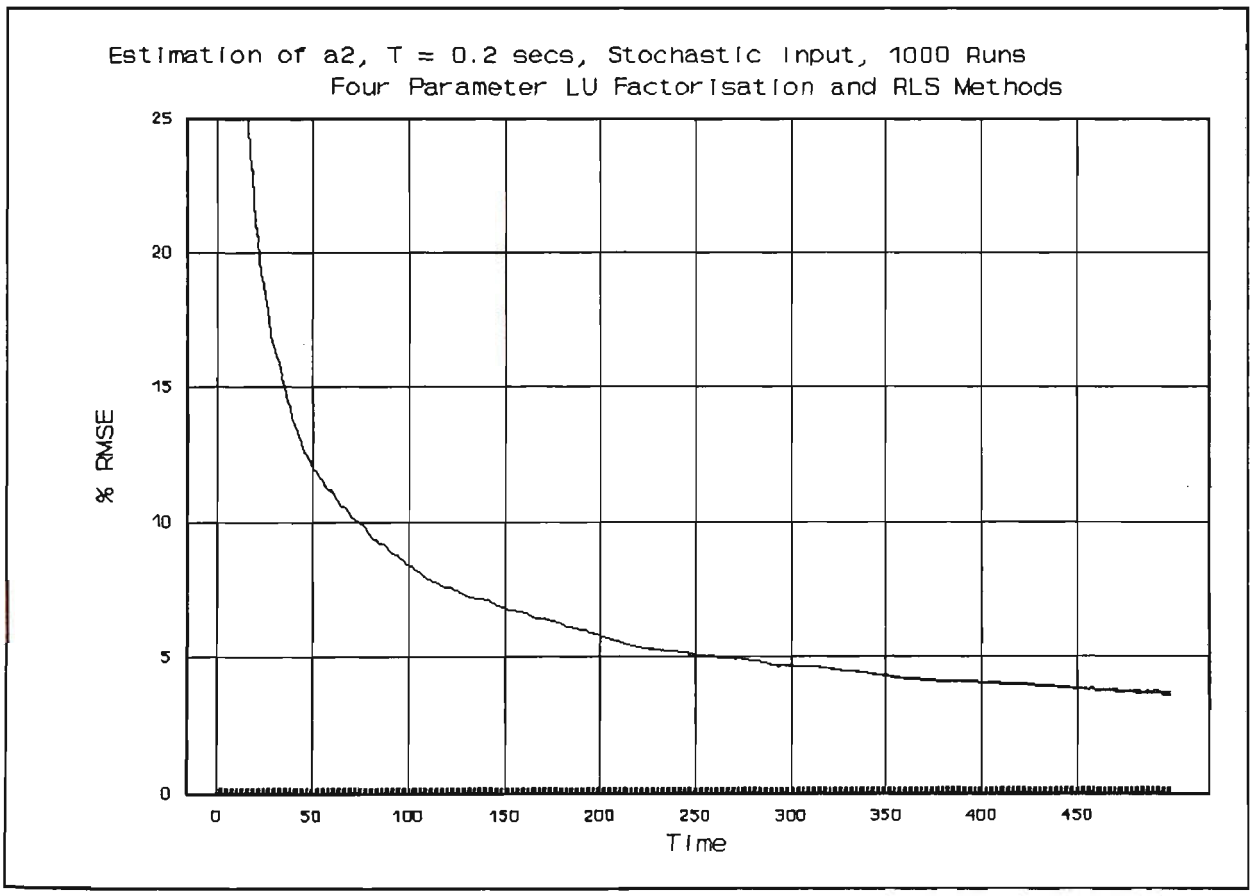

FIGURE 5-2 %RMSE IN ESTIMATION OF a2

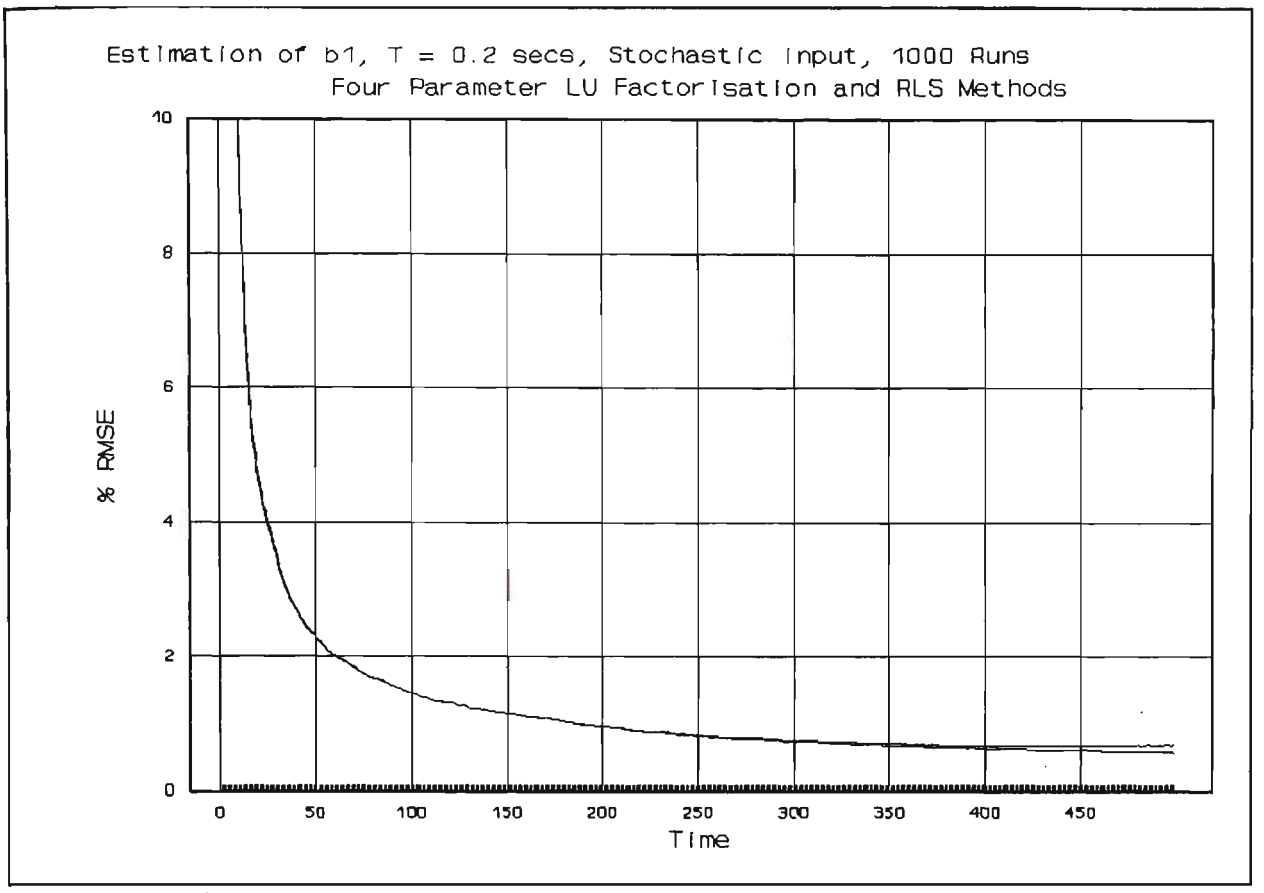

FIGURE 5-3 %RMSE IN ESTIMATION OF b1

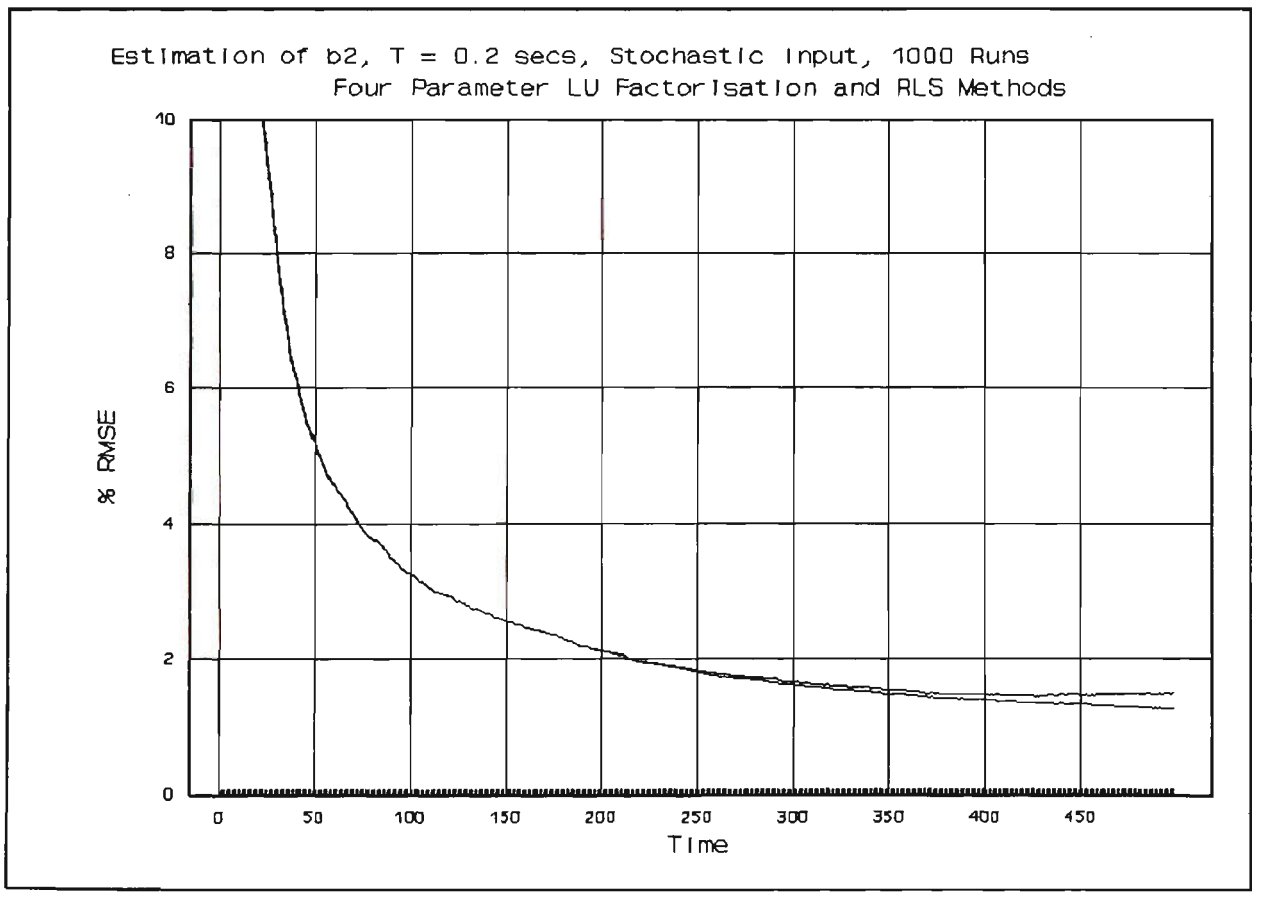

FIGURE 5-4 %RMSE IN ESTIMATION OF b2

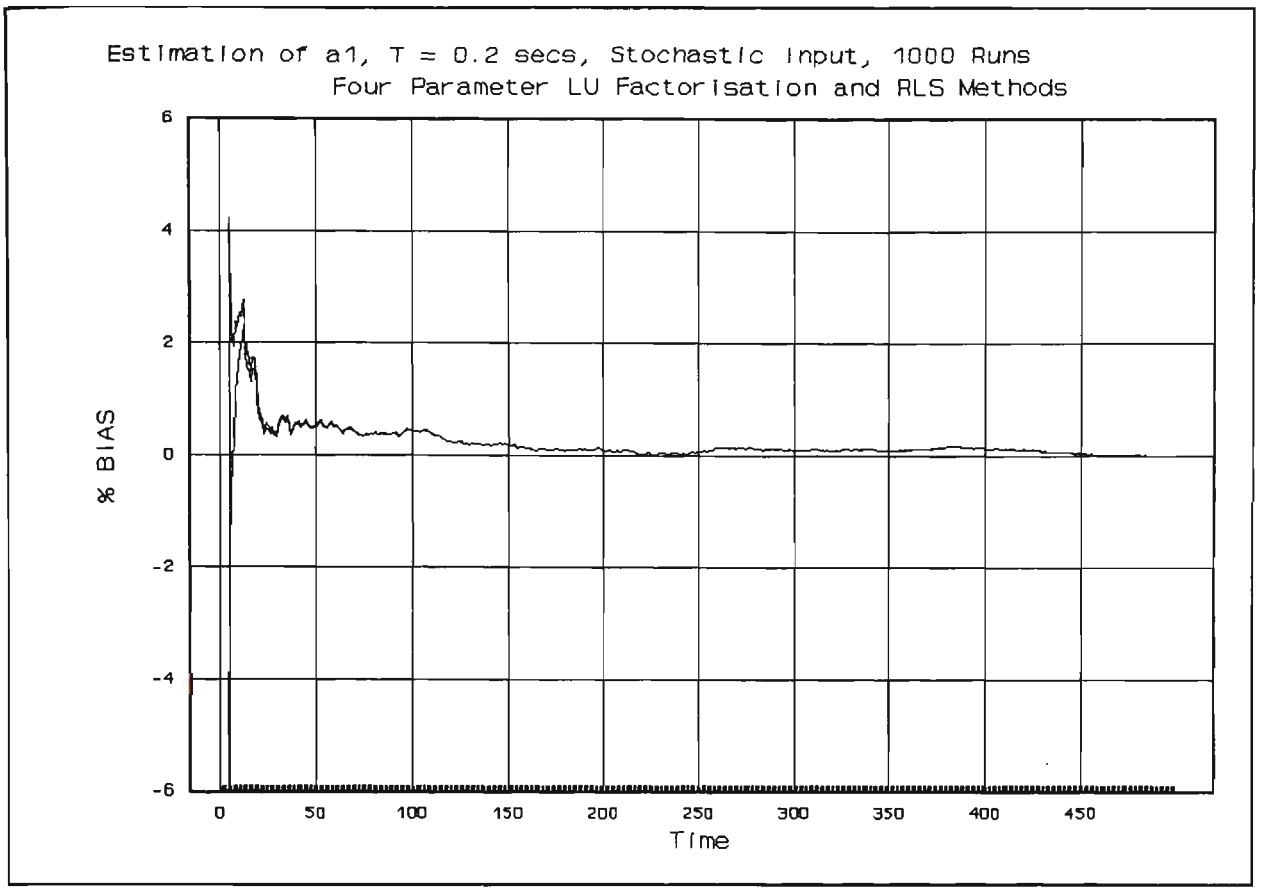

FIGURE 5-5 %BIAS IN ESTIMATION OF a1

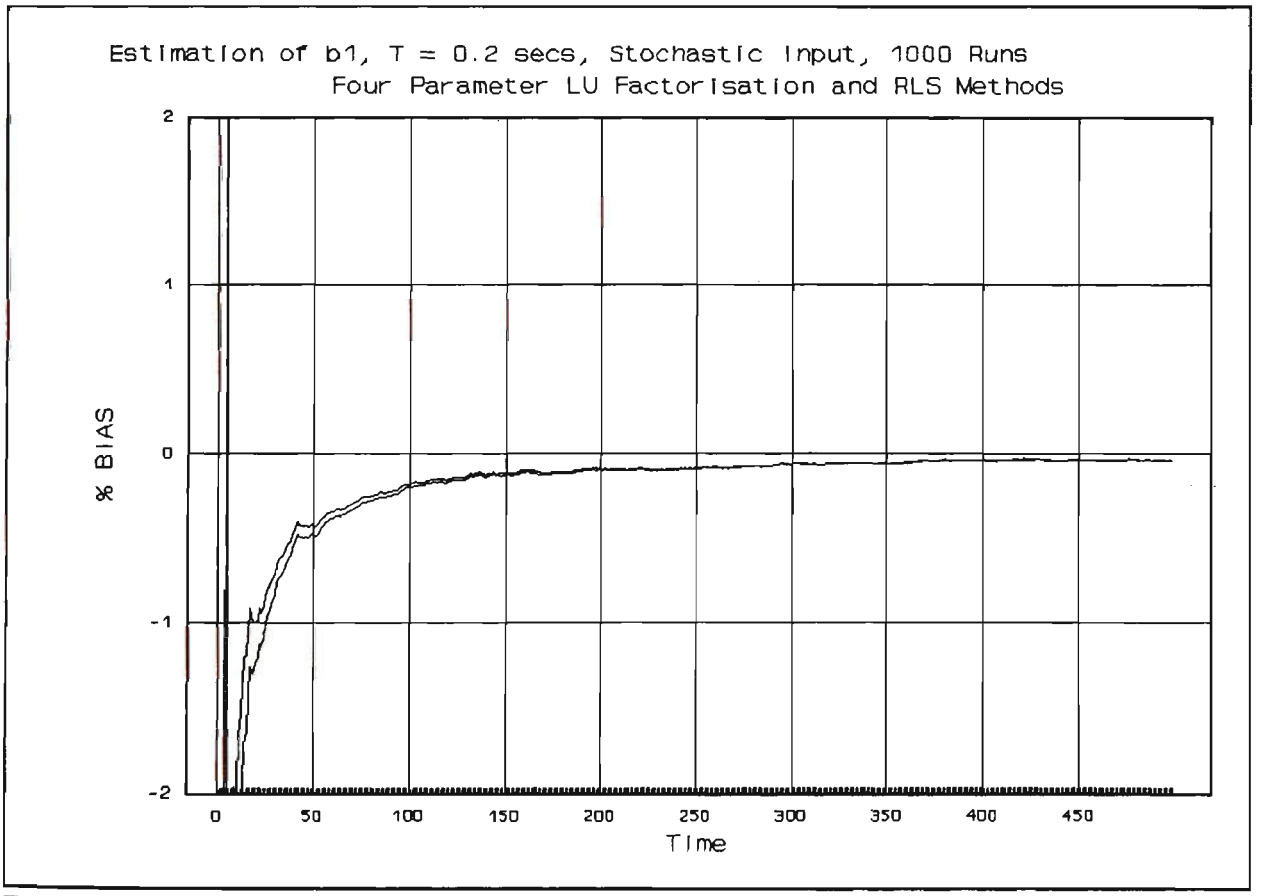

FIGURE 5-6 %BIAS IN ESTIMATION OF b1

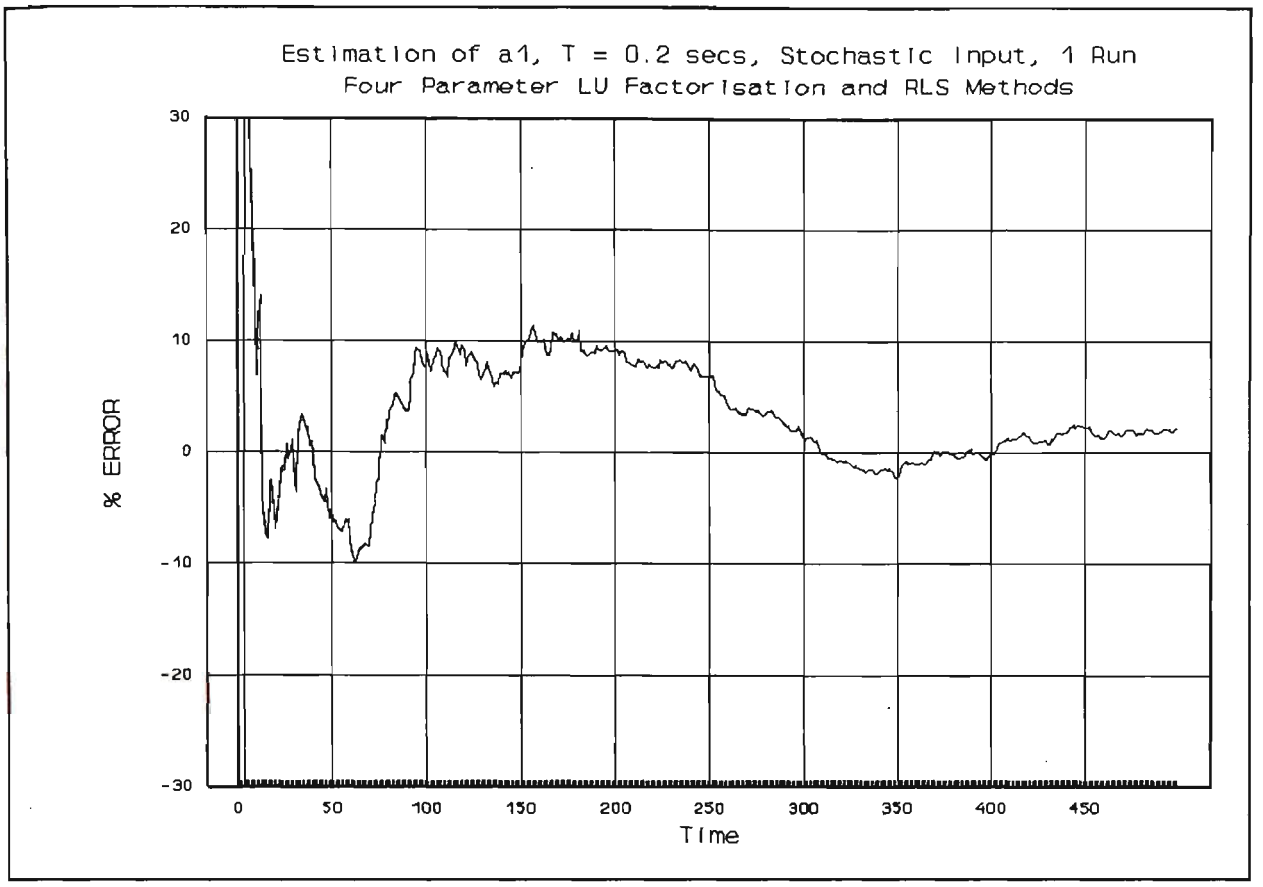

FIGURE 5-7 %ERROR IN ESTIMATION OF a1

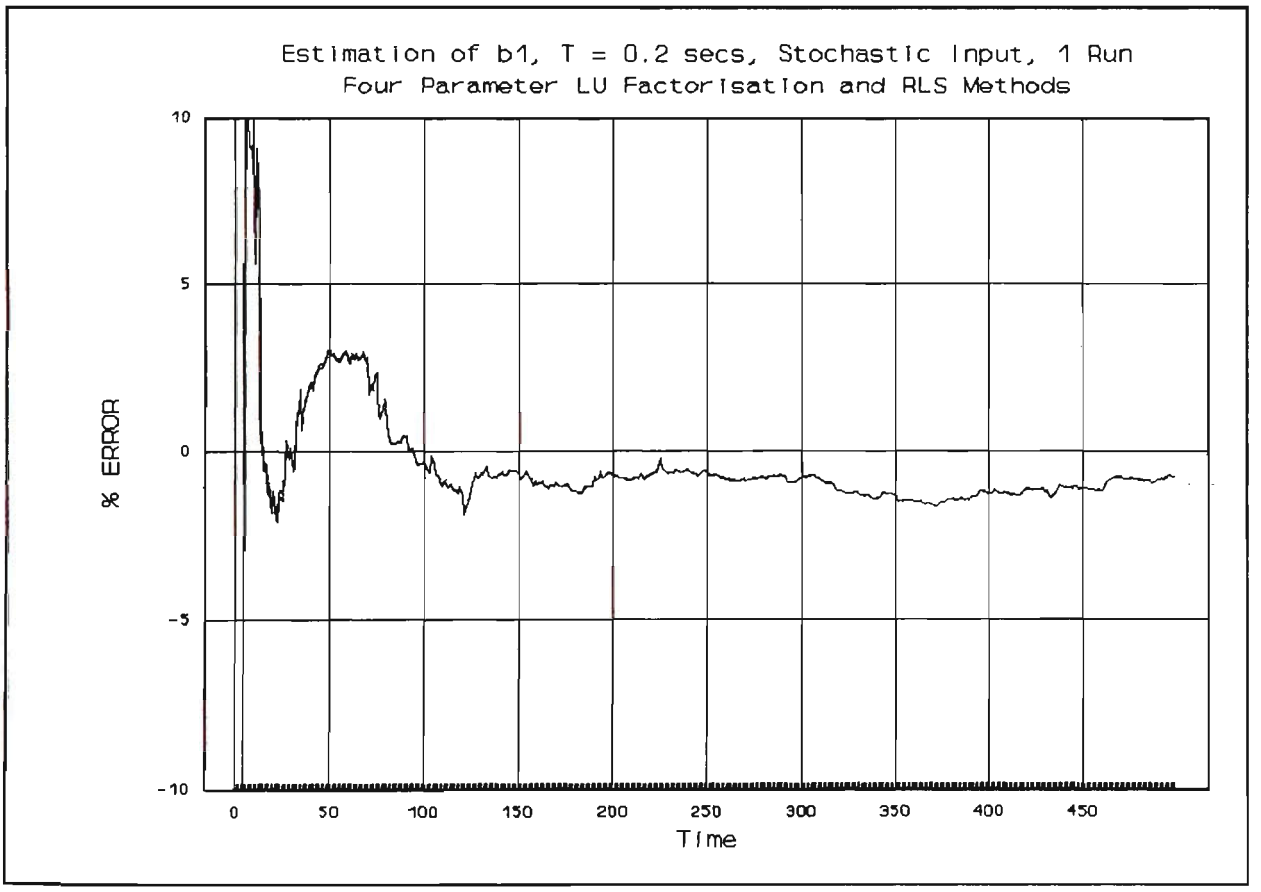

FIGURE 5-8 %ERROR IN ESTIMATION OF b1

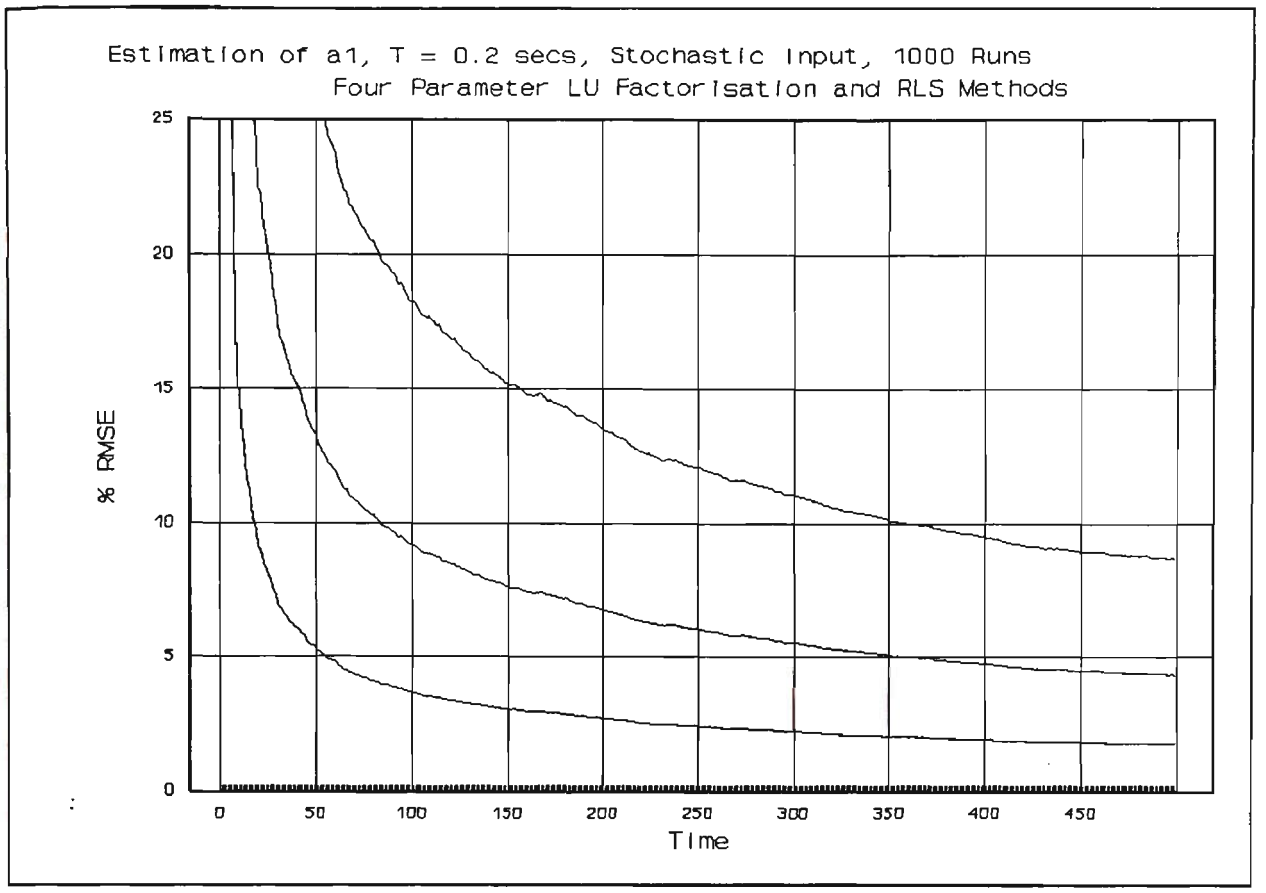

FIGURE 5-9 %RMSE IN ESTIMATION OF a1, SNR = 53.4,45.4 & 39.4 dB

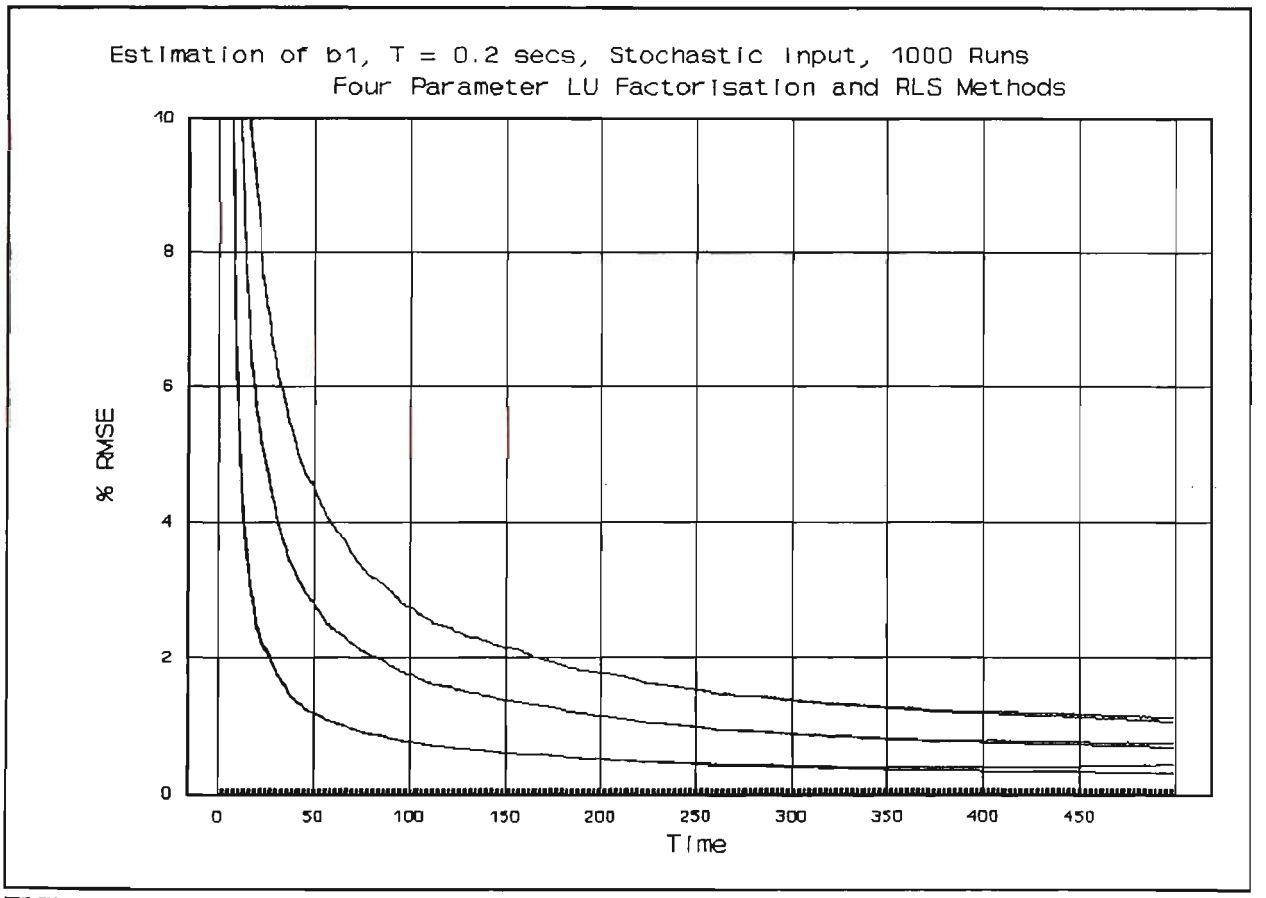

FIGURE 5-10 %RMSE IN ESTIMATION OF b1, SNR = 53.4,45.4 & 39.4 dB

COMPUTATION TIME

The program code required for solution using the RLS Method was measured to have an execution time of 93 microseconds. The corresponding time for the LU Factorisation Method was 112 microseconds.

RATE OF CONVERGENCE OF ESTIMATES

Figures 5-1 to 5-4 inclusive all show the decline in the percentage RMSE with the increase in samples to the estimator. Over the period covered (i.e. the first 500 sampling periods) the percentage RMSE falls with a reducing rate of decline.

The period covered is not adequate to show whether or not these curves tend to a value of zero, or to some finite asymptotic value. In terms of the problem considered in this thesis, namely real-time parameter estimation based on relatively few samples, the final percentage RMSE asymptotic value is not in itself of interest. The source of the RMSE is important, whether it is primarily due to a bias in the estimates, or due to spreading of the estimates by noise. Knowledge of the significance of these two contributions to the RMSE is desirable as a prerequisite to attempts to reduce the estimate errors.

BIAS OF THE ESTIMATES

Figures 5-5 and 5-6 show the normalised, percentage bias in the estimates of parameters a1 and b1. These plots were derived from the same simulations used to obtain Figures 5-1 to 5-4 inclusive.

From Figures 5-5 and 5-6 it is not clear whether or not both estimates will finally achieve a percentage bias value of zero. However, these Figures do indicate that the bias is less than the final values of percentage RMSE in Figures 5-1 and 5-3. Accordingly the main contributor to the final percentage RMSE values of the simulations is shown to be the spreading of the estimates by noise.

Figures 5-7 and 5-8 show the normalised percentage error in the estimates of a1 and b1 respectively. These Figures use the data from one run of the full simulation of one thousand runs and are included to show typical fluctuations in the estimates as the number of samples increases. For the SNR considered, i.e. 43.9 dB, it can be seen that even after the 300th sample period that both estimates are capable of variations of over one percent.

Figures 5-6 and 5-8 indicate that there may be bias in the estimation of b1. The results presented are inadequate to draw a firm conclusion on this.

The percentage bias and percentage error plots are not presented for the estimates of a2 and b2. In all the tests conducted it was found that the Moving Average (MA) Parameters, a1 and a2, always behaved in a similar manner, with a1 always more accurately estimated than a2. Likewise, the Auto-Regressive (AR) Parameters, b1 and b2, always behaved in a similar manner to one another, with b1 always more accurately estimated than b2.

SENSITIVITY OF THE ESTIMATORS TO NOISE

Figures 5-9 and 5-10 are included to illustrate the effect of the SNR upon the accuracy of the estimates. A decrease in the SNR resulting in a decrease in the accuracy of the estimates. Neither method offered an appreciable advantage over the other in dealing with the added noise. However in Figure 5-10 it should be noted that the RLS Method tends towards a smaller asymptotic value than does the LU Factorisation Method.

Figures 5-1 to 5-8 inclusive were all based on simulations using the one scaling factor in the generation of the noise sequence {n(kT)}. The different SNR figures, i.e. 47.4 and 43.9 dB, are thus indicative of the range of different SNR values for each run within any one thousand runs of the simulation.

RICHNESS OF THE INPUT SIGNAL

The stochastic input signal in the above simulations was changed at the sampling

instants of the estimators. Thus the spectral content of this signal, and hence its richness, was dependent upon the chosen sampling rate of the estimators.

SAMPLING PERIOD OF THE ESTIMATOR

Due to the dependence of the input signal upon the sampling rate these simulations were not suitable to examine the effects of the ratio of system time constant to estimator sampling period.

STABILITY OF THE ESTIMATOR

Figures 5-3, 5-4 and 5-10 all show splitting of the traces of RMSE vs sample time. In all cases this splitting is observed towards the end of the simulation, and in all cases the RLS Method produced the lower trace.

This splitting was subsequently found to be a manifestation of a problem of instability with the LU Factorisation Method as implemented for these simulations. This problem is discussed in Section 5-5, where further results more clearly illustrate the nature of this problem.

SUMMARY OF SIMULATIONS USING A STOCHASTIC INPUT SIGNAL AND ESTIMATOR WITH SAMPLING PERIOD OF 0.2 SECONDS

From the results considered so far, the RLS Method offers two distinct advantages:

i) An execution time of 93 microseconds, compared to 112 microseconds required by the LU Factorisation Method.

ii) Superior long term convergence of the estimates of the Auto-Regressive Parameters, b1 and b2.

Other observations based upon these results are:

i) The plots suggest that for rapid, real-time parameter estimation there is little to choose

between the quality of the estimates produced by the two methods. Whilst the RLS Method initially produces grossly inaccurate estimates, the LU Factorisation Method has to remain dormant until the Information Matrix is of Full Rank. By the time that the LU Factorisation Method is available, the RLS Method is producing estimates of comparable accuracy.

ii) The Moving Average (MA) Estimates, $\hat{a}_1$ and $\hat{a}_2$, evolve in a similar manner to one another. The Auto-Regressive (AR) Estimates, $\hat{b}_1$ and $\hat{b}_2$, evolve in a similar manner to one another.

iii) The coefficients of the more recent signals (i.e. a1 and b1) are more accurately estimated than the corresponding coefficients of the earlier signals (i.e. a2 and b2 respectively).

iv) The estimates of the AR coefficients are more accurate than those of the MA coefficients.

v) There is no significant bias detected in the estimation of a1. The results do suggest a small bias in the estimation of b1.

vi) For the SNR and test length (i.e. 500 sample periods) considered, the errors in the estimates are predominantly attributable to a "large" standard deviation of the estimates, rather than bias.

5-4 THE COMPARISON OF LU FACTORISATION AND RLS METHODS USING A STOCHASTIC INPUT SIGNAL WITH AN INCREASED SAMPLING RATE

RESULTS OF SIMULATIONS

Simulations using a sample period of 0.05 seconds produced different results and conclusions. Some typical results are presented in Figures 5-11 to 5-14 inclusive.

TABLE 5-2 SUMMARY OF RESULTS PRESENTED OF ESTIMATION OF ARMA COEFFICIENTS USING AN INCREASED SAMPLING RATE

FIGURE	COEFF.	T (s)	Y-AXIS VARIABLE	No. of RUNS	RUN LENGTH	SNR (dB)
5-11	a1	0.05	%RMSE	1000	50T	42.0,27.9
5-12	a2	0.05	%RMSE	1000	50T	42.0,27.9
5-13	b1	0.05	%RMSE	1000	50T	42.0,27.9,14.0
5-14	b2	0.05	%RMSE	1000	50T	42.0,27.9,14.0

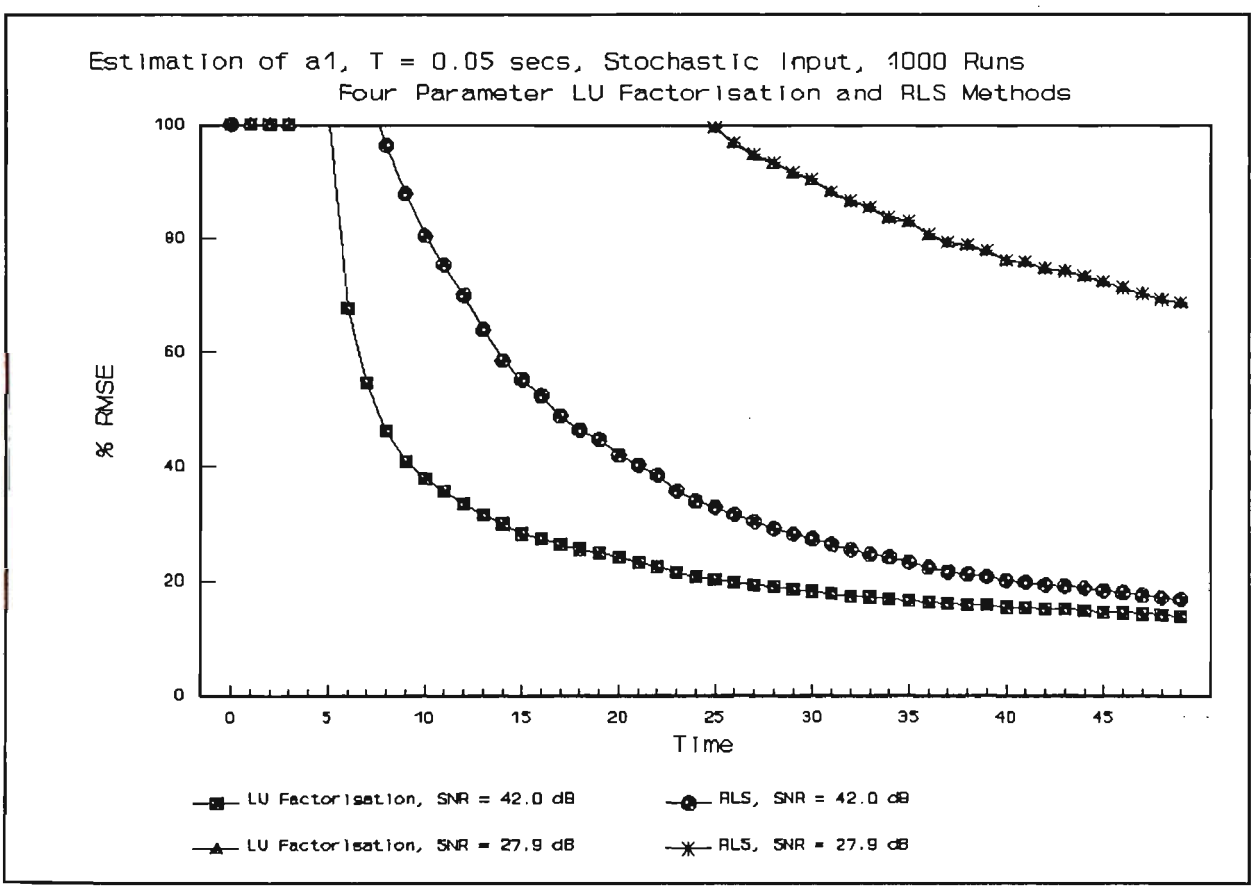

FIGURE 5-11 %RMSE IN ESTIMATION OF a1

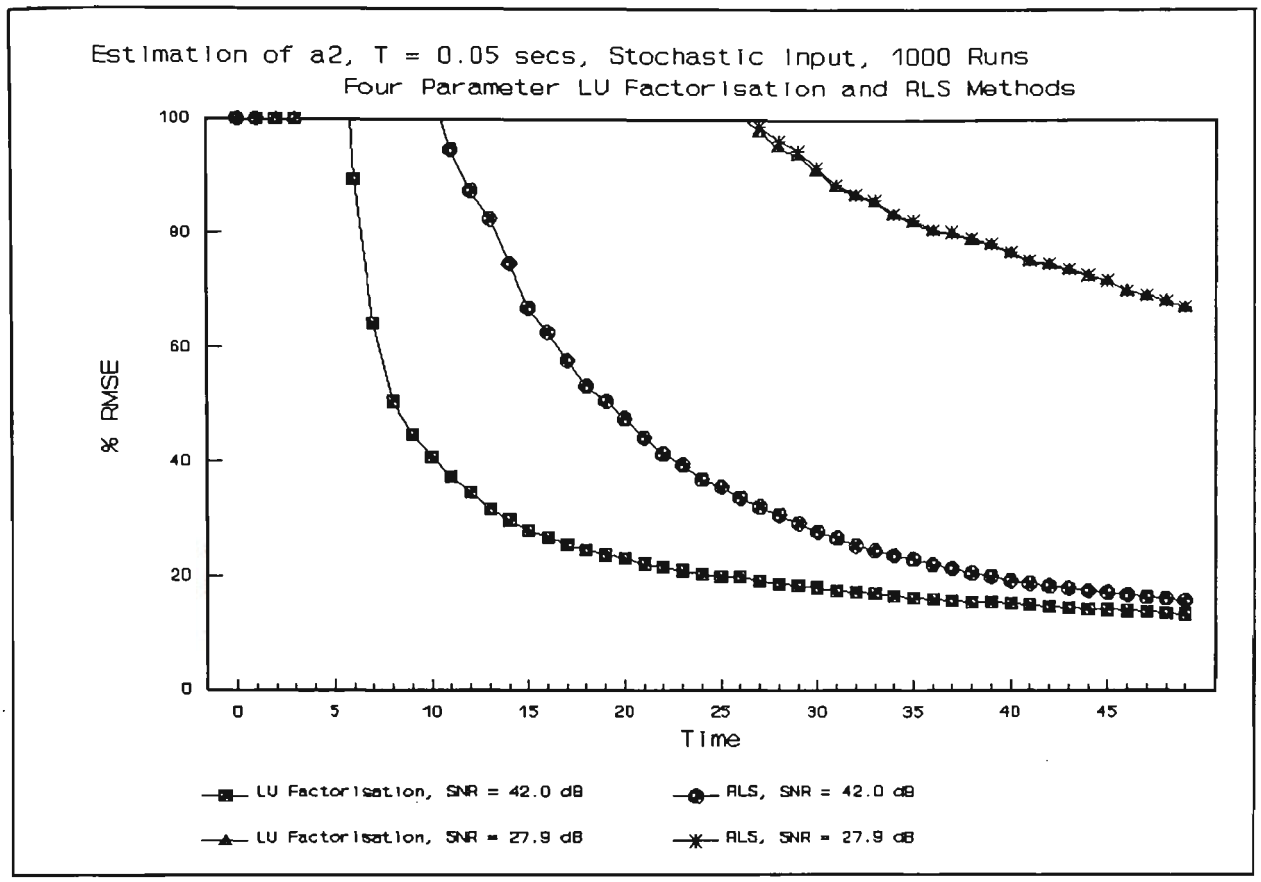

FIGURE 5-12 %RMSE IN ESTIMATION OF a2

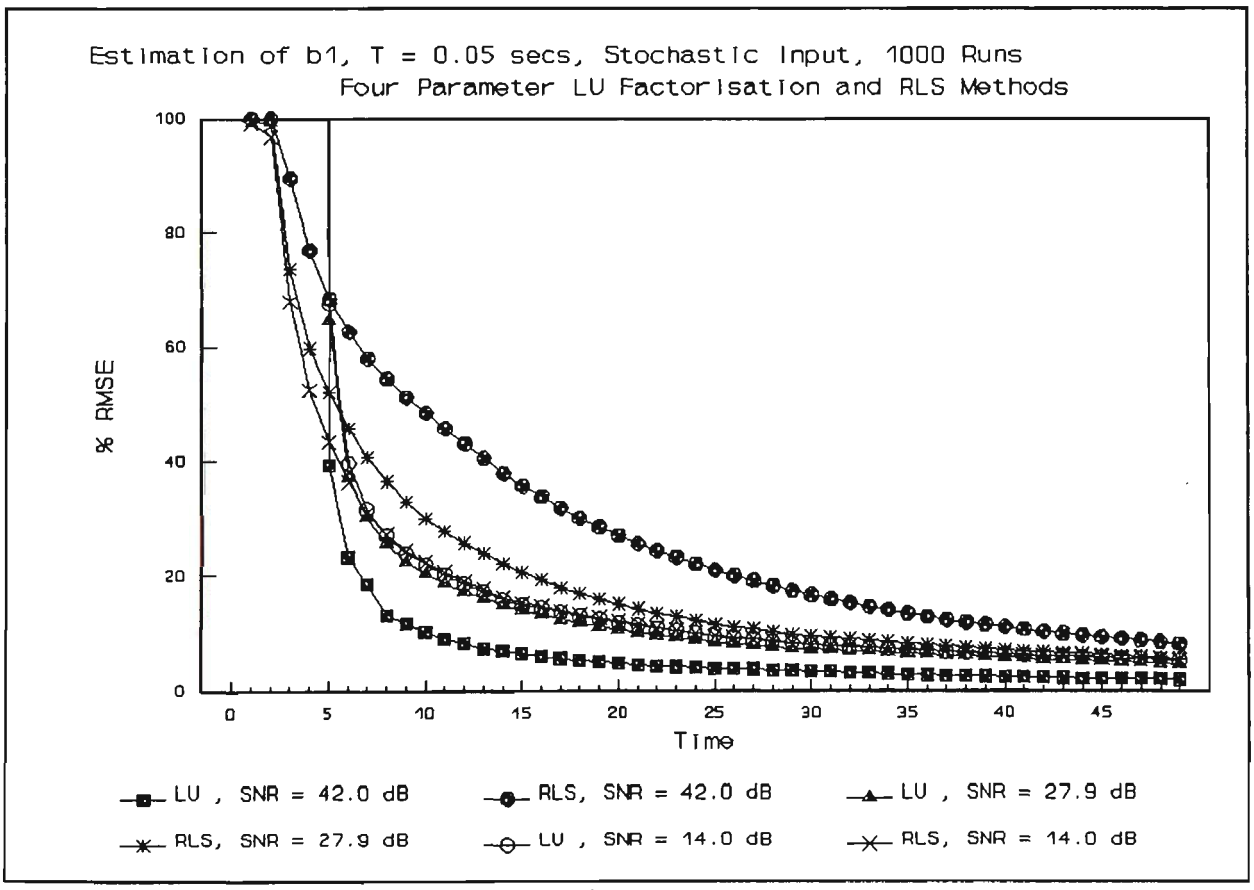

FIGURE 5-13 %RMSE IN ESTIMATION OF b1

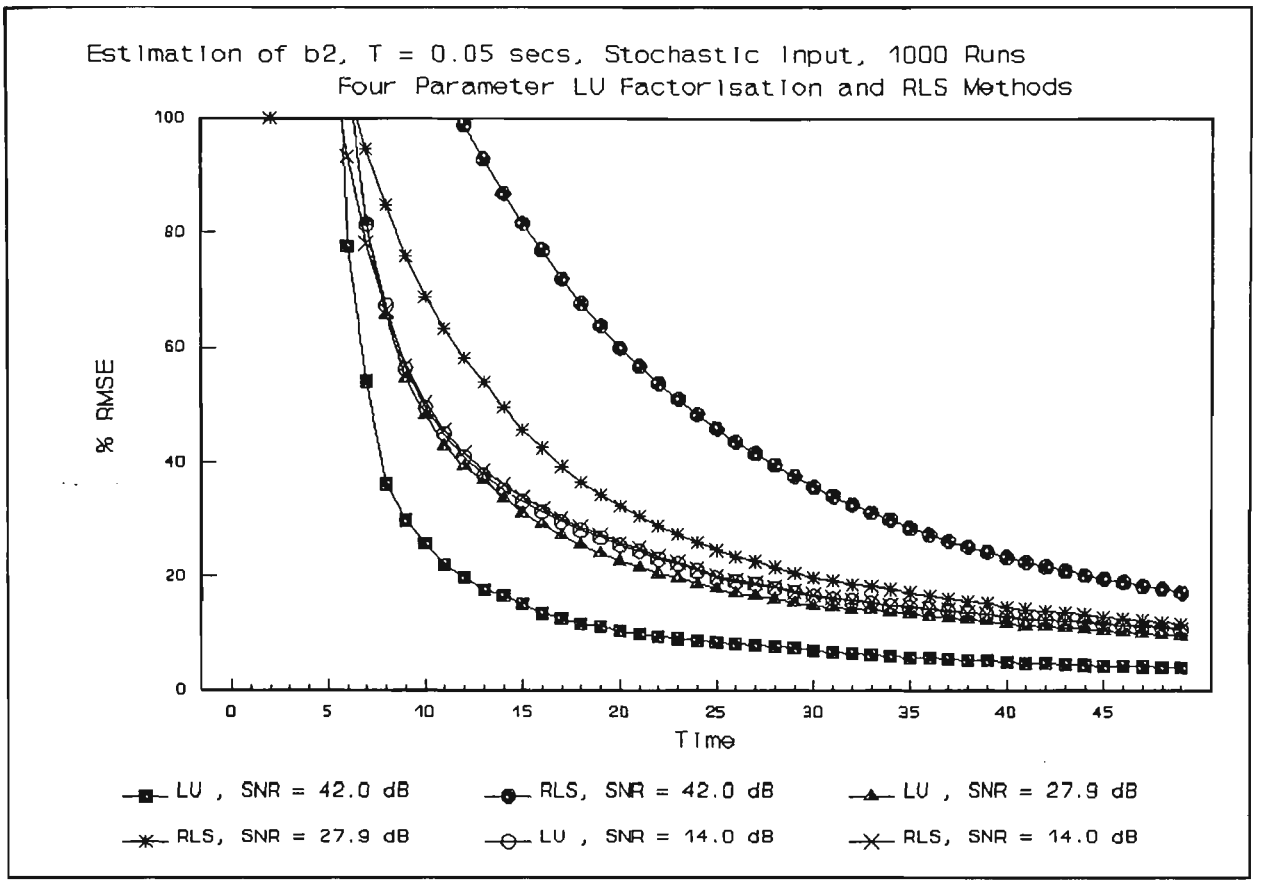

FIGURE 5-14 %RMSE IN ESTIMATION OF b2

RATE OF CONVERGENCE OF ESTIMATES

i) In estimating the MA Coefficients (a1 and a2) the LU Factorisation Method provided more rapid convergence than the RLS Method, for the case of an SNR of 42.0 dB.

ii) In estimating the AR Coefficients the LU Factorisation Method again provided more rapid convergence than the RLS Method.

iii) In estimating the AR Coefficients the RLS Method provided more rapid convergence as the SNR was reduced.

The following comments are an interpretation of these results:

SENSITIVITY OF THE ESTIMATORS TO NOISE

The improvement of the estimates of b1 and b2 as the noise is increased is a consequence of the means by which the output, noise, disturbance signal was modelled. The

sequence {n(kT)} was not only presented to the estimator, but also fed back into the simulated process through the regressor vector ψ. The noise sequence was thus able to excite the Auto-Regressive Section of the process model. It is this increased excitation that resulted in a decrease in SNR resulting in an improvement in the accuracy of the AR estimates.

RICHNESS OF THE INPUT SIGNAL

The use of a short sample period, T = 0.05 seconds, resulted in a poor excitation signal for the system. The stochastic input signal being of zero mean, and changing at every sampling instant, failed to drive the system output far from zero. Simulations using sample periods of 0.2 and 0.5 seconds produced larger variations in the output signal of the system. It is concluded that the LU Factorisation Method gives superior performance (i.e. more rapid convergence) when presented with a small number of samples of such poor signals. No attempt was made to improve the performance of the RLS Method by modification of the initial scaling of the Error Covariance Matrix.

SAMPLING PERIOD OF THE ESTIMATOR

The superior performance of the LU Factorisation Method with a short sample period is significant. As noted earlier, an important component of this thesis is to consider the effects of increasing the sampling rate as a means of improving both the real-time convergence and accuracy of the estimates.

5-5 THE COMPARISON OF LU FACTORISATION AND RLS METHODS USING TEST INPUT SIGNALS

DESCRIPTION OF SIMULATIONS USING TEST INPUT SIGNALS

Sections 5-3 and 5-4 dealt with simulations using stochastic input signals. This section presents results from simulations using test input signals based upon rectangular pulses.These

simulations used the same sampling periods (including T = 0.2 and 0.05 seconds) and same plant transfer function (as per Equation 5-1) as those with a stochastic input signal.

The main advantages of using a test input signal are:

i) to enable an investigation of the effects of the richness of the input signal to be made. This was done by using single pulses of different width, and by using multiple pulses as the test signal.

ii) to enable a simple investigation of the effects of the chosen sampling rate.[1]

The simulations using test input signals had the following restrictions:

i) The pulses of the different test input signals were to have transitions that were synchronised with all three estimator sample clocks. For the estimator sampling rates considered here this restriction was met by ensuring that all pulses had widths equal to an integer number of seconds.

ii) The ARMA Process was operated at the same rate as the estimator with the shortest period (i.e. 0.05 seconds). The ARMA Process was the same as that described by Equation 5-2.

All of the estimators were therefore presented with two signal sequences from the

---

[1]In the earlier simulations, using stochastic input signals, the input signal was changed at the same rate as the sampling rate of the estimator. This resulted in the spectral content of the input signal being dependent upon the selected sample period of the estimator. To examine the effects of chosen sampling rate using a stochastic input signal would require two 'noise' sequences. The first would have the same periodic time as the sampling period of the estimator, and would model the "measurement" noise, and as such would only be fed into the estimator itself.

The second noise sequence, of different periodic time, must be independent of the estimator sampling rate. This second sequence would model noise and disturbances within the ARMA Process. Accordingly the elements of this sequence would be fed into both the estimator and the ARMA Process itself. This second sequence entering the ARMA Process via the Regressor Vector.

Such an experiment would be complex due to a number of problems, e.g.

i) the relative scaling of the amplitudes of the two noise sequences
ii) the definition of SNR
iii) the selection of a ratio of estimator sampling period to the periodic time of the ARMA Process.

ARMA Process. These sequences represented the input and output signals {u(kT)} and $\{y_n(kT)\}$, with T = 0.05 seconds. The noise sequence, {n(kT)}, thus had a period of 0.05 seconds in these simulations.

The estimators with a sample period of 0.05 seconds used all elements of the signal sequences. The estimators with a sample period of 0.2 seconds used every fourth element of each sequence. The estimators with a sample period of 0.5 seconds used every tenth pair of elements.

The coefficients of the ARMA Process, Equation 5-2, may be calculated for a given periodic time from Equations 4-13 to 4-16 inclusive, and Equation 5-1. The results of such calculations are presented in Table 5-3 below:

TABLE 5-3 SHOWING THE VALUES OF THE COEFFICIENTS AT THE DIFFERENT SAMPLING RATES

T (secs)	a1	a2	b1	b2
0.5	0.0533	0.0451	-1.6065	0.6065
0.2	0.0094	0.0088	-1.8187	0.8187
0.05	0.0006	0.0006	-1.9512	0.9512

Clearly the sensitivity of the prediction $\hat{y}_0$ (as per Equation 5-3) to the different coefficients will change as T is changed. To circumvent this difficulty the comparison is made upon a consideration of the estimate of the finite pole location of the plant of Equation 5-1. This plant was chosen to have this unknown pole located at 1 radian per second.

The selection of the pole location as the parameter to be estimated offers a number of

advantages:

i) the mathematical model suggests that the value of p should be independent of the sampling rate of the estimator.

ii) the value of p is highly significant in terms of the behaviour of the servosystem.

iii) the use of a single parameter (i.e. p) as the basis of comparisons clarifies the task of presenting the results.

The estimate of the pole location, $\hat{p}$, may be found using the estimate of the parameter b_2, as per Equation 4-16, i.e.

$$\hat{p} = -\frac{1}{T}.\ln \hat{b}_2 \qquad 5\text{-}5$$

A further comparison was made based upon the magnitude of the error in the One-Step Ahead Prediction made by each estimator. It should be remembered that the Least Squares Estimators adjust the model coefficients so as to minimise the sum of the square of this Prediction Error.

TEST INPUT SIGNAL DEFINITIONS

The test input signals used were either single square wave pulses, or repeated square waves, all switched between levels of +5.0 and +0.01. The single pulses took the high value for the duration of the pulse. The square waves had their first half-cycle at the high value, with transitions once every second.

RESULTS OF SIMULATIONS

Figures 5-15 to 5-29 inclusive present the results of the simulations using a variety of test input signals and estimator sampling rates.

Figures 5-15 to 5-24 inclusive show the process by which the plant pole at 1 radian per

second has its location estimated. Each trace is based upon one thousand runs of the simulation, with each run using a different section of the noise sequence {n(kT)}. All of the one thousand runs of each simulation used the same scaling factor in determining the amplitude of this noise sequence. The traces shown in Figures 5-15 to 5-24 are:

i) the Mean of the estimate of the pole location

ii) the envelope of two standard deviations of these estimates either side of that mean.

In all of the simulations using test input signals, it should be noted that the SNR is dependent not only upon the scaling factor of the noise sequence {n(kT)}, but also upon the chosen input signal. Accordingly the description of the significance of noise is not adequately described by the SNR value alone. In all of the simulations presented in Figures 5-15 to 5-29 the RMS value of the noise sequence was held constant at 0.01.

TABLE 5-4 PRESENTED RESULTS OF ESTIMATION OF UNKNOWN POLE (MEAN +/- 2 STANDARD DEVIATIONS) USING TEST INPUT SIGNALS

FIGURE	ESTIMATOR T (secs)	SNR (dB)	RMS NOISE	INPUT SIGNAL	METHOD
5-15	0.2	47.8	0.01	1 sec pulse	LU
5-16	0.2	47.8	0.01	1 sec pulse	RLS
5-17	0.2	61.1	0.01	5 sec pulse	LU
5-18	0.2	61.1	0.01	5 sec pulse	RLS
5-19	0.2	64.9	0.01	2 sec sq wave	LU
5-20	0.2	64.9	0.01	2 sec sq wave	RLS
5-21	0.5	64.9	0.01	2 sec sq wave	LU
5-22	0.5	64.9	0.01	2 sec sq wave	RLS
5-23	0.05	64.9	0.01	2 sec sq wave	LU
5-24	0.05	64.9	0.01	2 sec sq wave	RLS

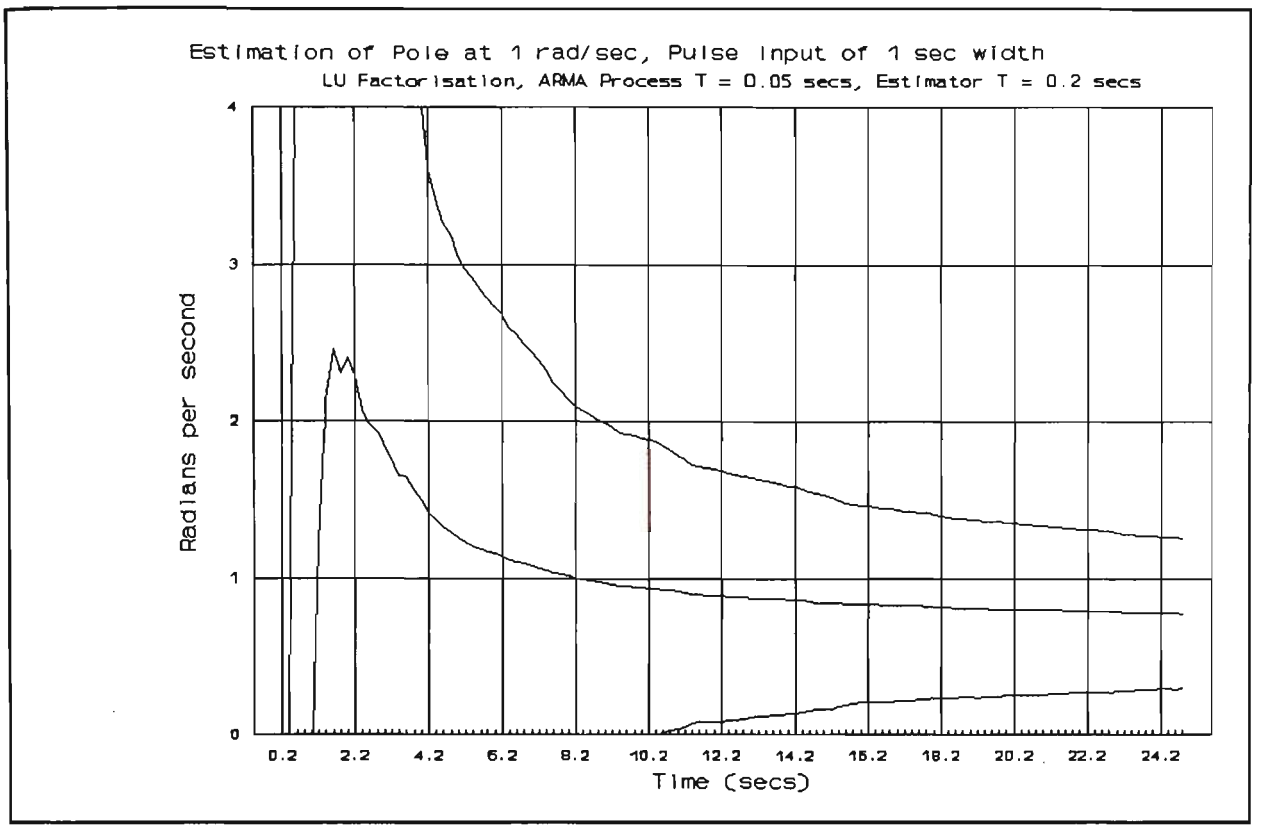

FIGURE 5-15 ESTIMATION OF POLE LOCATION USING LU FACTORISATION AND A SINGLE,NARROW PULSE INPUT SIGNAL

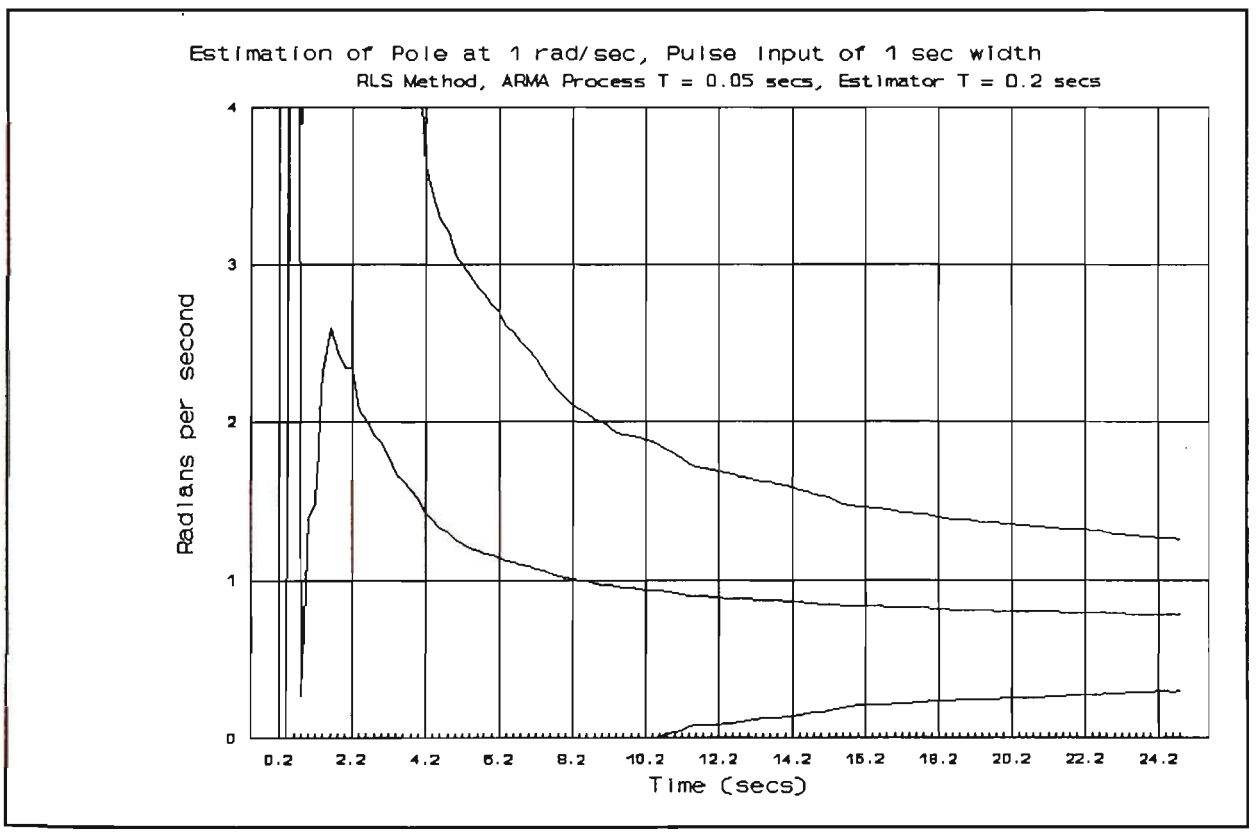

FIGURE 5-16 ESTIMATION OF POLE LOCATION USING RLS METHOD AND A SINGLE, NARROW PULSE INPUT SIGNAL

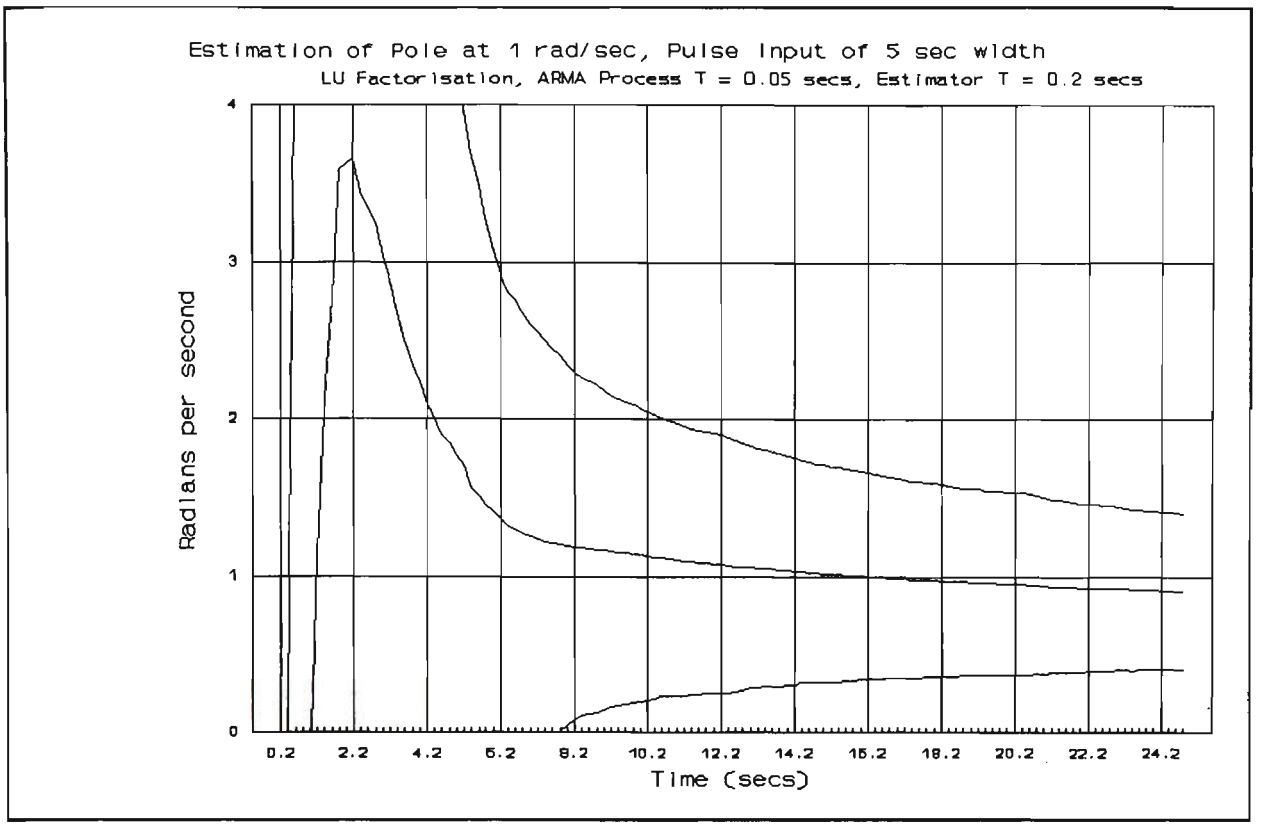

FIGURE 5-17 ESTIMATION OF POLE LOCATION USING LU FACTORISATION AND A SINGLE, WIDE PULSE INPUT SIGNAL

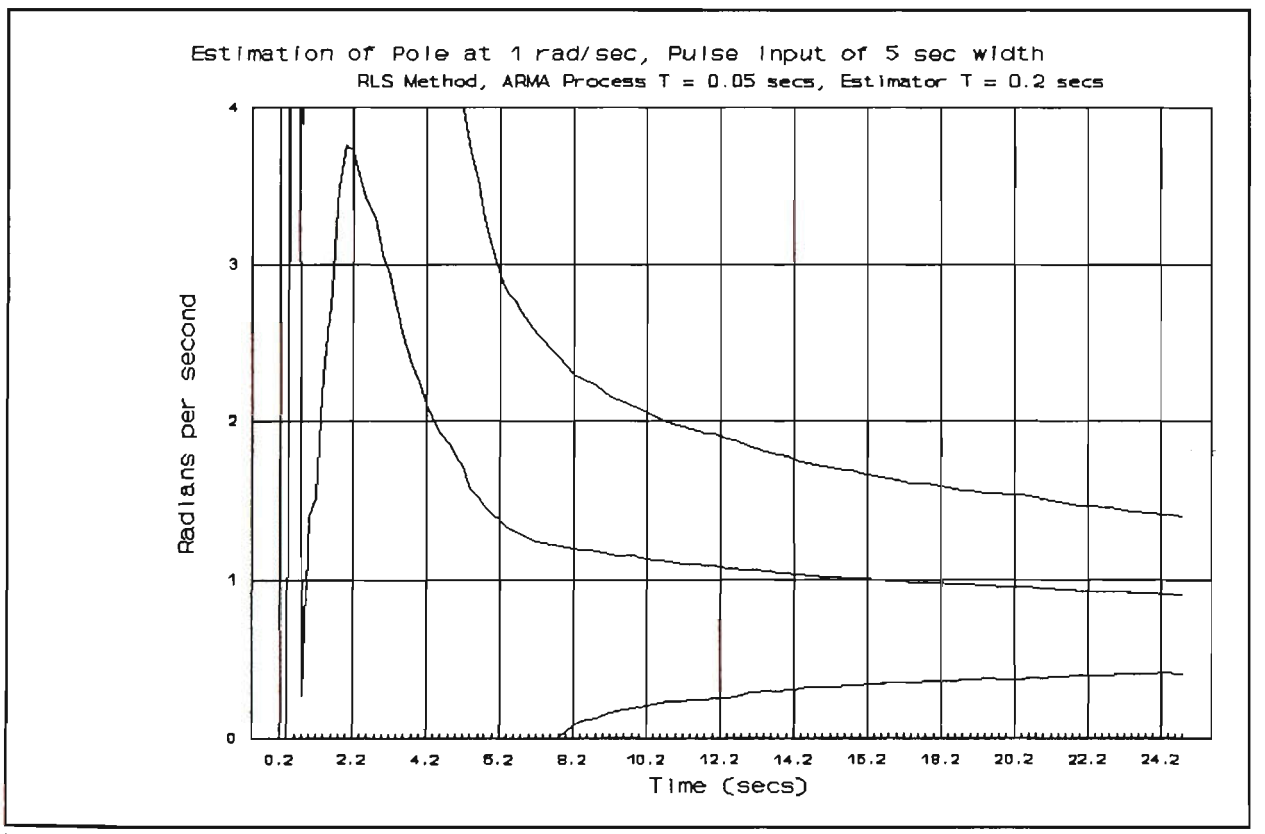

FIGURE 5-18 ESTIMATION OF POLE LOCATION USING RLS METHOD AND A SINGLE, WIDE PULSE INPUT SIGNAL

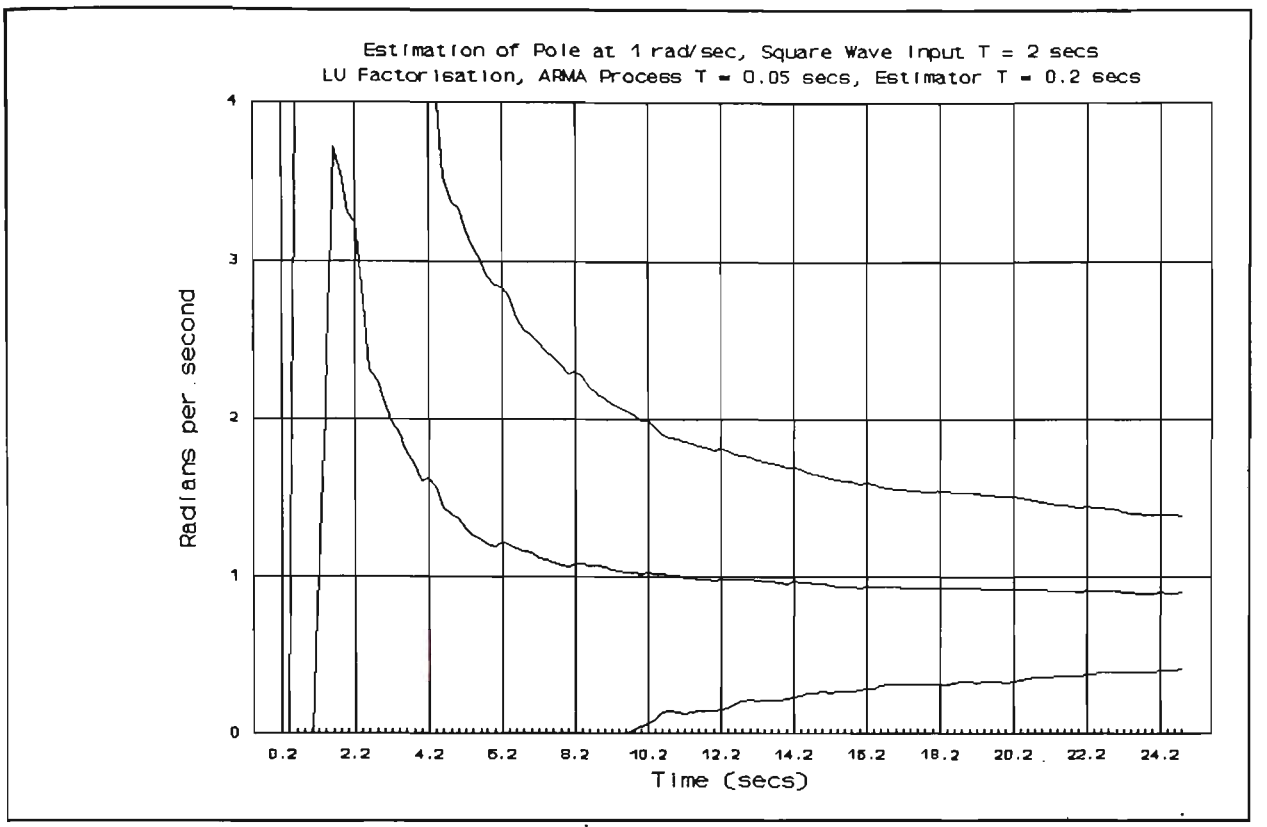

FIGURE 5-19 ESTIMATION OF POLE LOCATION USING LU FACTORISATION AND A SQUARE WAVE INPUT SIGNAL

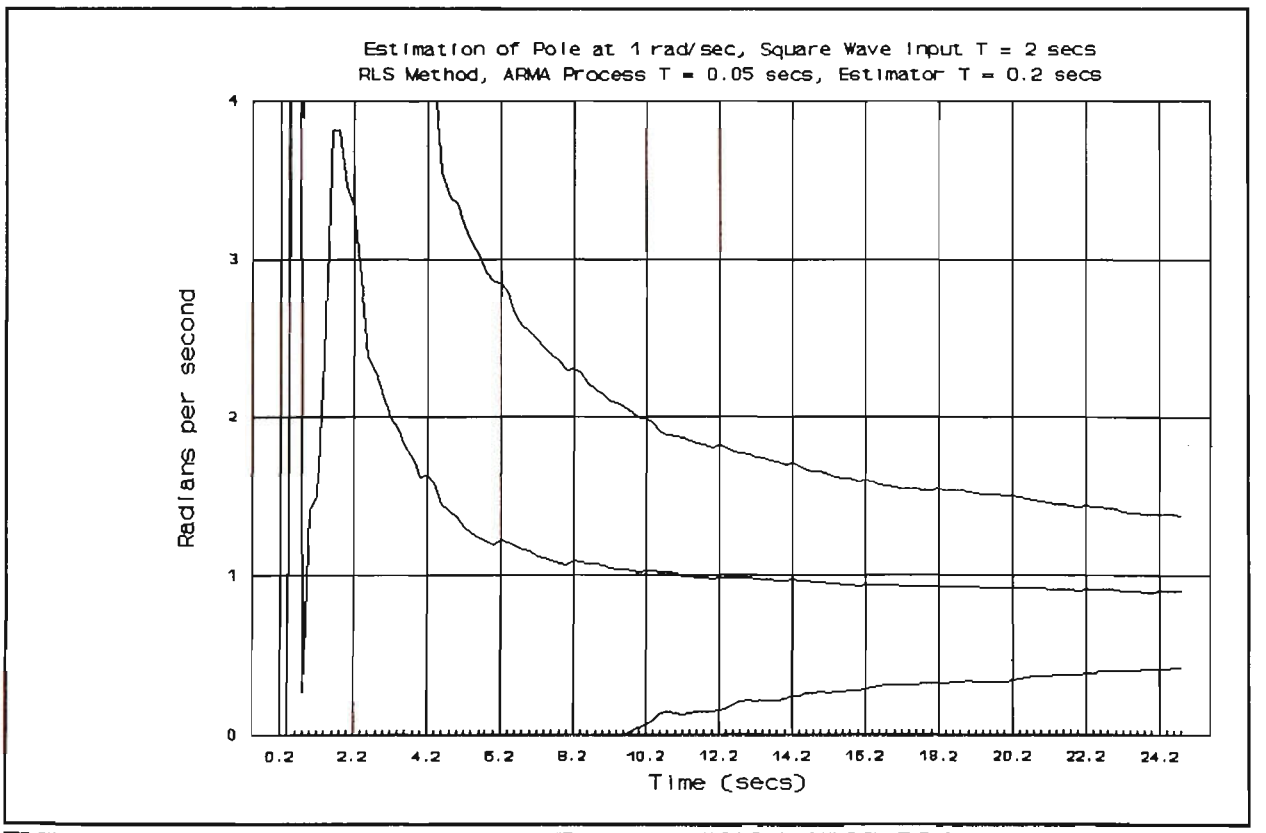

FIGURE 5-20 ESTIMATION OF POLE LOCATION USING RLS METHOD AND A SQUARE WAVE INPUT SIGNAL

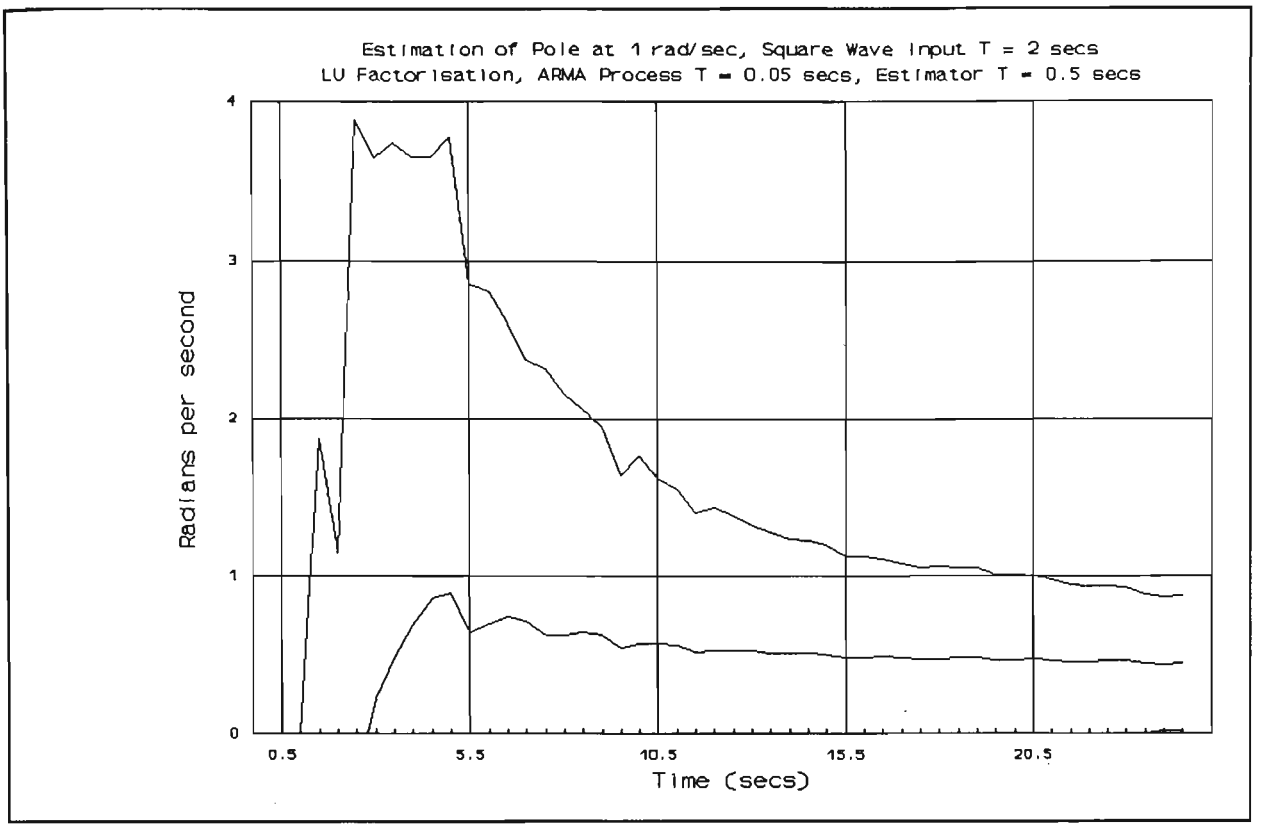

FIGURE 5-21 ESTIMATION OF POLE LOCATION USING LU FACTORISATION WITH A SLOW SAMPLING RATE

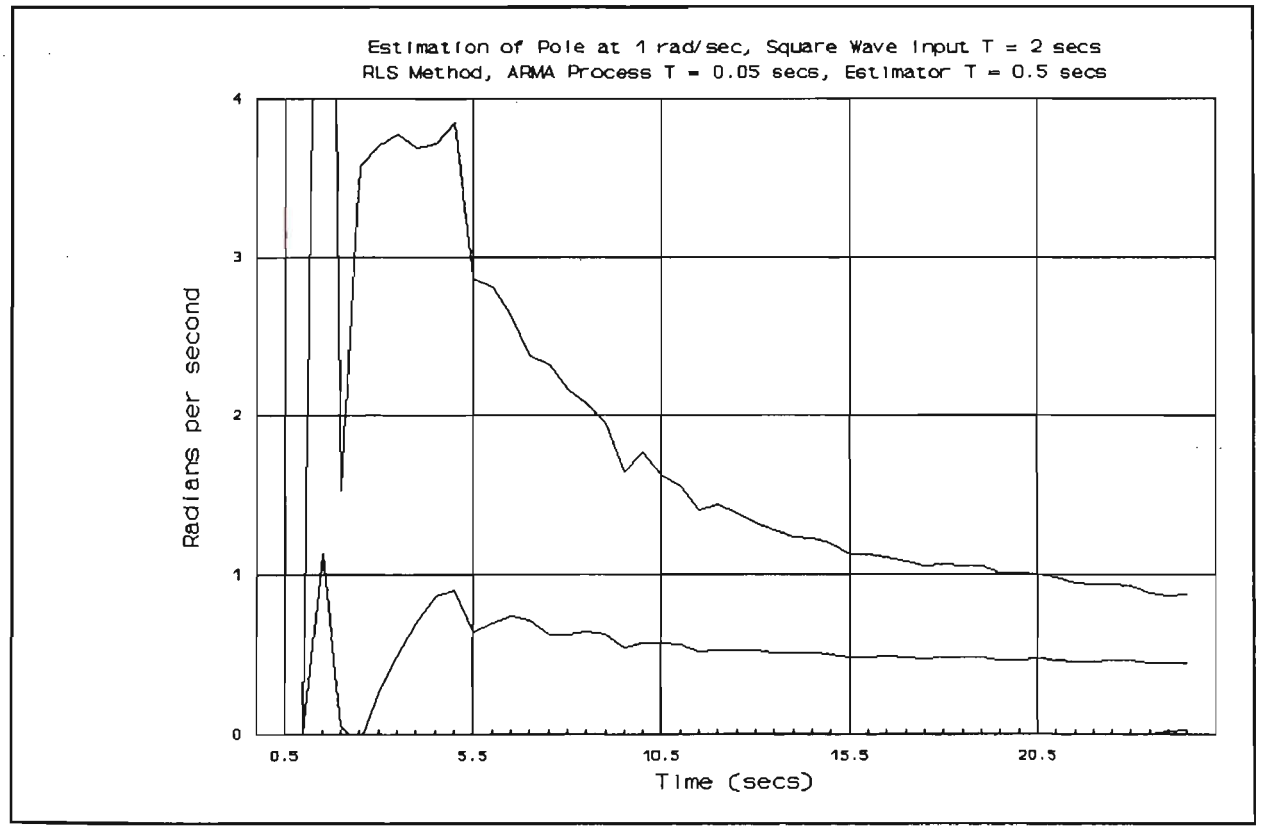

FIGURE 5-22 ESTIMATION OF POLE LOCATION USING RLS METHOD WITH A SLOW SAMPLING RATE

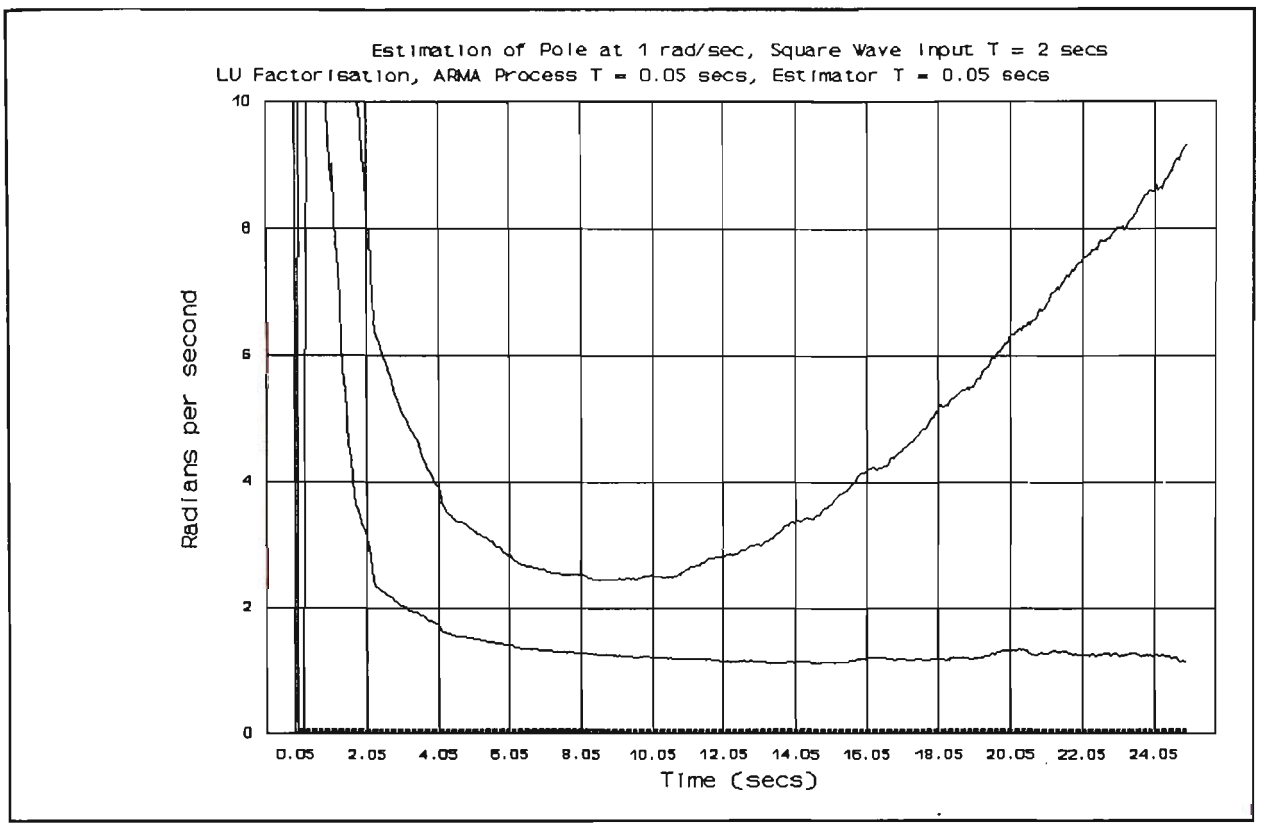

FIGURE 5-23 ESTIMATION OF POLE LOCATION USING LU FACTORISATION WITH A HIGH SAMPLING RATE

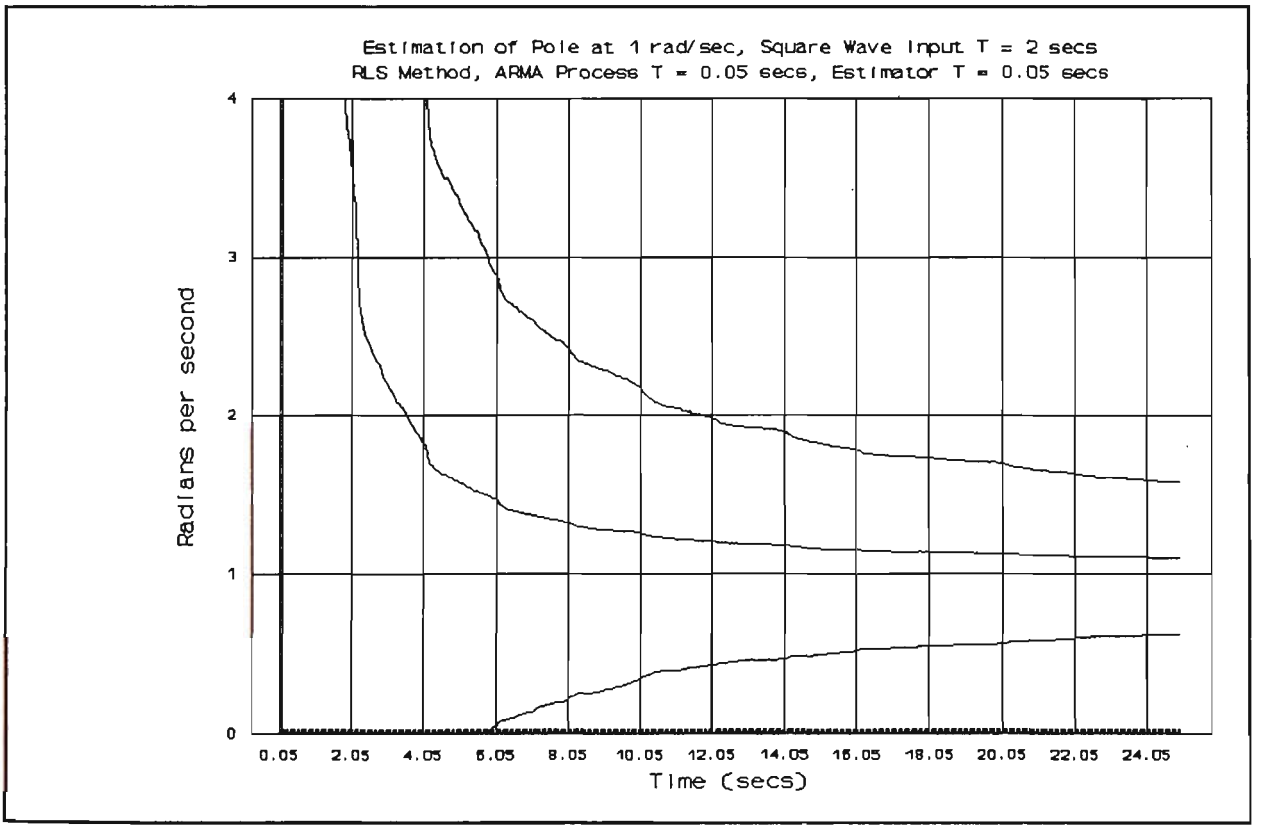

FIGURE 5-24 ESTIMATION OF POLE LOCATION USING RLS METHOD WITH A HIGH SAMPLING RATE

RATE OF CONVERGENCE OF ESTIMATES

The Rate of Convergence of the estimates may be measured by considering the narrowing of the envelope produced by the +/- two Standard Deviations traces. To assist in this, the mean value of the estimate at time t = 24.2 seconds is taken to be the 'Final' value of the estimate. The Rates of Convergence may then be compared by measuring the time at which the "+ 2 Standard Deviation" trace falls to a value double that of the 'Final' value. This time is described as "Convergence Time" in Table 5-5.

The last three results listed in Table 5-5 (based on the results presented in Figures 5-20, 5-22 and 5-24) indicate that as the estimator sample rate is increased both the Rate of Convergence of the estimates, and the accuracy of the 'Final' Value are improved. This result was subsequently attributed to the modelling of the noise process used in these simulations.

TABLE 5-5 SUMMARY OF FINAL VALUES AND CONVERGENCE RATES FOR THE DIFFERENT SIMULATIONS USING THE RLS METHOD

FIGURE	ESTIMATOR PERIOD	INPUT SIGNAL	'FINAL' VALUE	CONVERGENCE TIME
5-16	0.2	1s pulse	0.77	14.4 secs
5-18	0.2	5s pulse	0.90	13.2 secs
5-20	0.2	2s sq wave	0.90	11.8 secs
5-22	0.5	2s sq wave	0.43	23.8 secs
5-24	0.05	2s sq wave	1.07	10.0 secs

RICHNESS OF THE INPUT SIGNAL

The first three results listed in Table 5-5 (based on the results presented in Figures 5-16, 5-18 and 5-20) indicate that the choice of the test input signal affects both the accuracy of

the 'Final' Value and the Convergence Rate of the estimates. The square wave train, with its multiple transitions being the preferred signal of the three considered.

Figures 5-25, 5-26 and 5-27 illustrate how transitions in the input signal affect the error in the "One-Step Ahead Prediction". Whilst a small prediction error is desirable, suggesting as it does that the estimator has produced accurate estimates of the unknown parameters, it will be noted that the preferred input signal generates the largest perturbations in the prediction error. This is reasonable on two counts:

i) It is this error that enables the estimator to improve the accuracy of its estimates.

ii) The single pulse inputs provide a more steady output response from the system than that resulting from the square wave input signal. The more steady the output response, the more easily it may be predicted.

TABLE 5-6 SUMMARY OF PRESENTED RESULTS OF MEAN VALUE OF ERROR IN ONE-STEP AHEAD PREDICTION

FIGURE	ESTIMATOR T (secs)	SNR (dB)	RMS NOISE	INPUT SIGNAL	METHOD
5-25	0.2	47.8	0.01	1 sec pulse	RLS
5-26	0.2	61.1	0.01	5 sec pulse	RLS
5-27	0.2	64.9	0.01	2 sec sq wave	RLS
5-28	0.05	64.9	0.01	2 sec sq wave	LU
5-29	0.05	64.9	0.01	2 sec sq wave	RLS

Figures 5-25 to 5-29 inclusive show the magnitude (i.e. modulus) of the error in the One-Step Ahead Prediction of the system output. Again each trace indicates the mean of one

thousand runs of that simulation.

SAMPLING PERIOD OF THE ESTIMATOR

As noted above, an increase in the sampling rate of the estimator may result in an improvement in both the accuracy and the Convergence Rate of the estimates. However caution is required to ensure that the Least Squares problem remains numerically well-conditioned.

STABILITY OF THE ESTIMATOR

Figure 5-23 clearly illustrates a simulation in which the LU Factorisation Method becomes unstable. This observation is reinforced by the corresponding trace of Prediction Error presented in Figure 5-28. It should be noted that the RLS Method, using identical signal sequences performed well (Figures 5-24 and 5-29).

Further simulations revealed that the stability of the LU Factorisation Method was dependent upon both the selected input signal and upon the run length of the simulation.

This loss of stability by the LU Factorisation Method also explains the "splitting" of the traces noted in Figures 5-3,5-4 and 5-10 of Section 5-3. This "splitting" being a consequence of the onset of loss of stability as the contents of the Information Matrix increased.

In Figure 5-10 it can be seen that the magnitude of the splitting increased as the SNR was reduced. The loss of stability is attributed to rounding errors causing the Least Squares Equations on which the Information Matrix is based degenerating towards a singular set of equations. Some protection against this problem may be achieved by the use of "pivoting" [5-18]. However this would add yet further to the computation time required by the LU Factorisation Method.

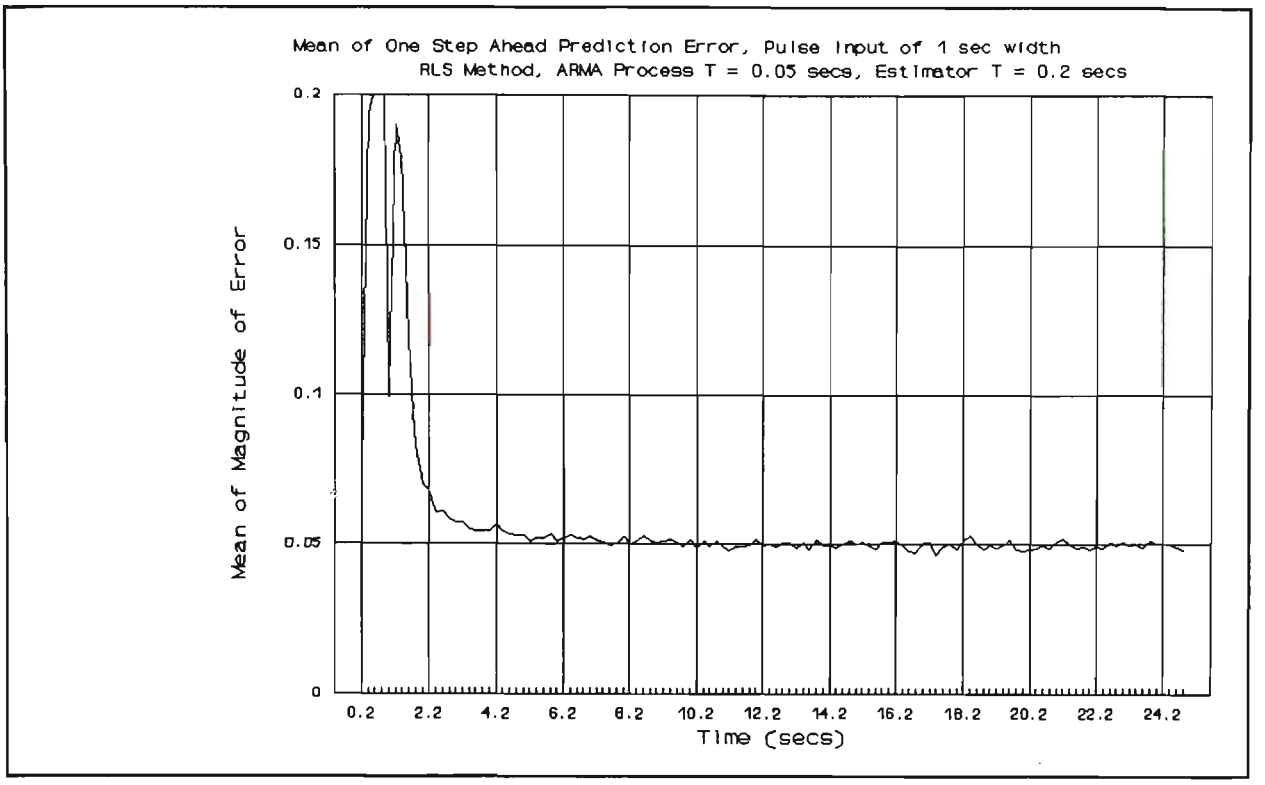

FIGURE 5-25 ERROR IN ONE-STEP AHEAD PREDICTION USING A SINGLE, NARROW PULSE INPUT SIGNAL

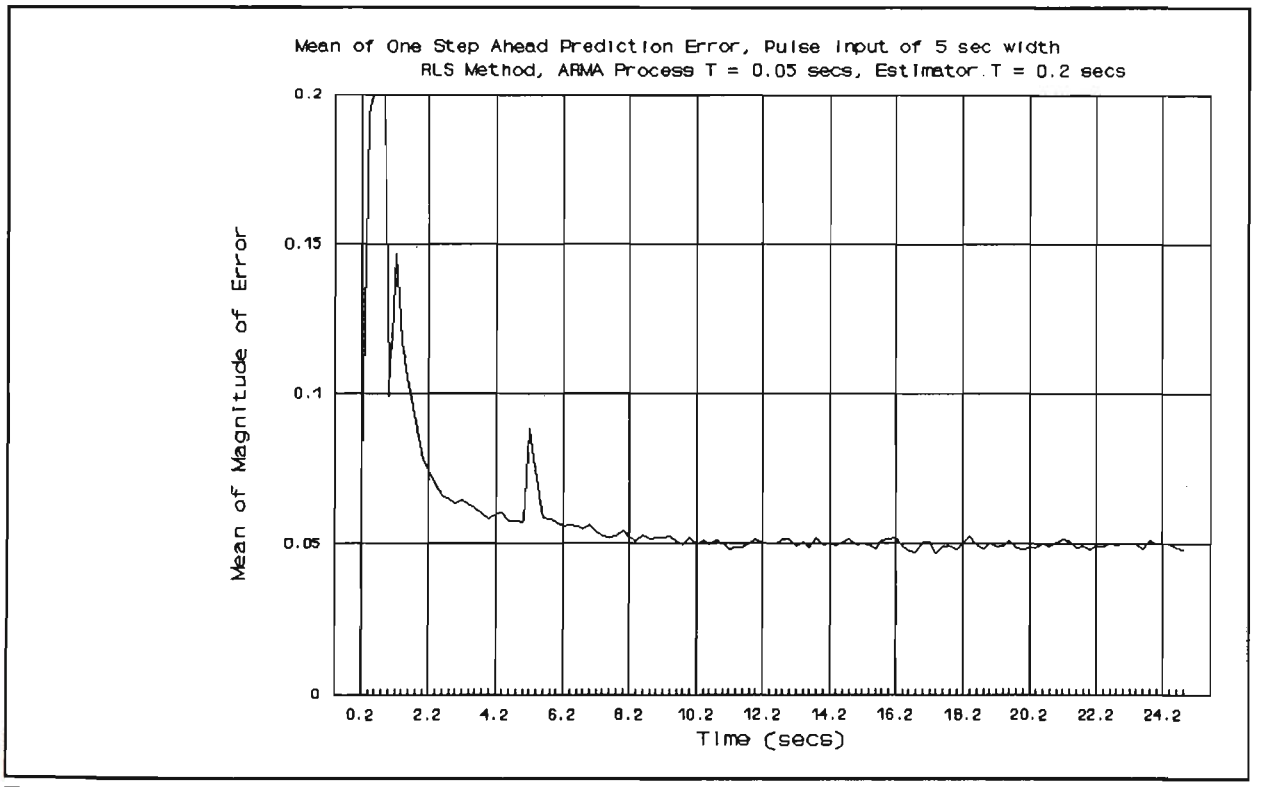

FIGURE 5-26 ERROR IN ONE-STEP AHEAD PREDICTION USING A SINGLE, WIDE PULSE INPUT SIGNAL

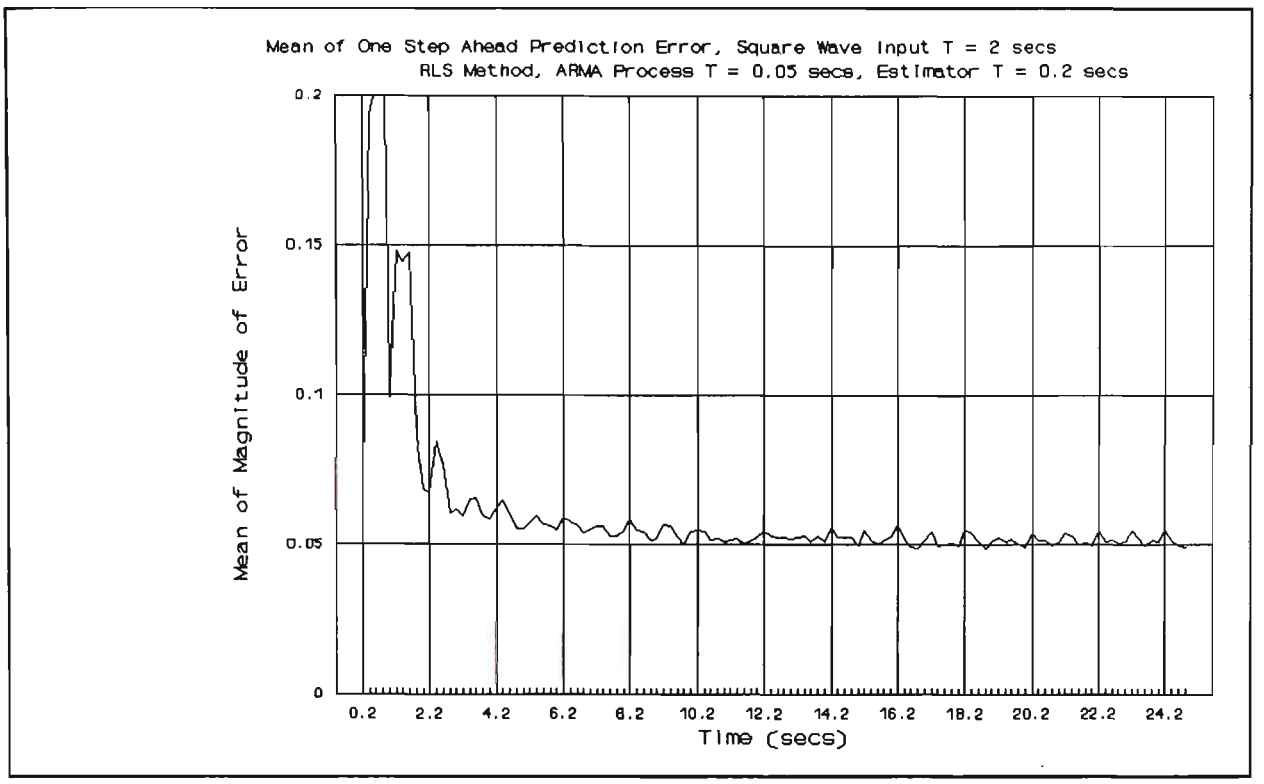

FIGURE 5-27 ERROR IN ONE-STEP AHEAD PREDICTION USING A SQUARE WAVE INPUT SIGNAL

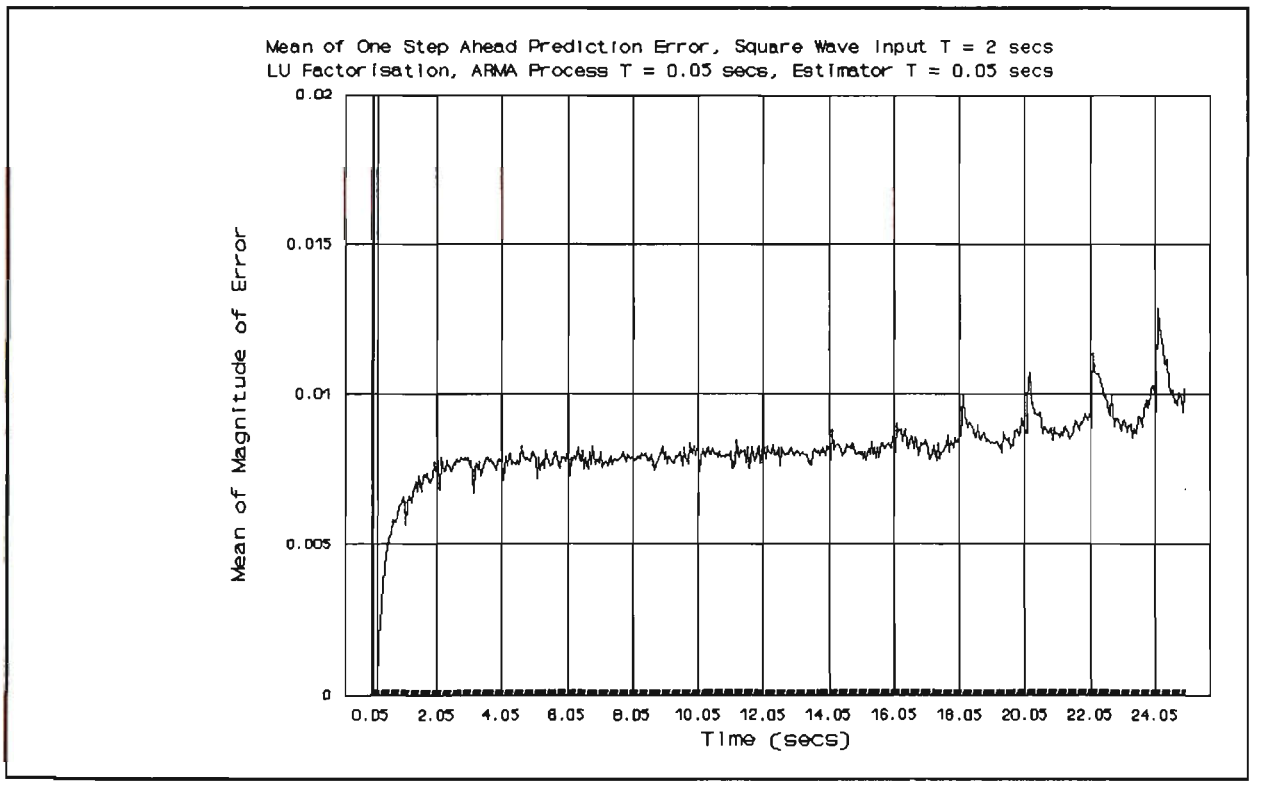

FIGURE 5-28 ERROR IN ONE-STEP AHEAD PREDICTION USING LU FACTORISATION WITH A HIGH SAMPLING RATE

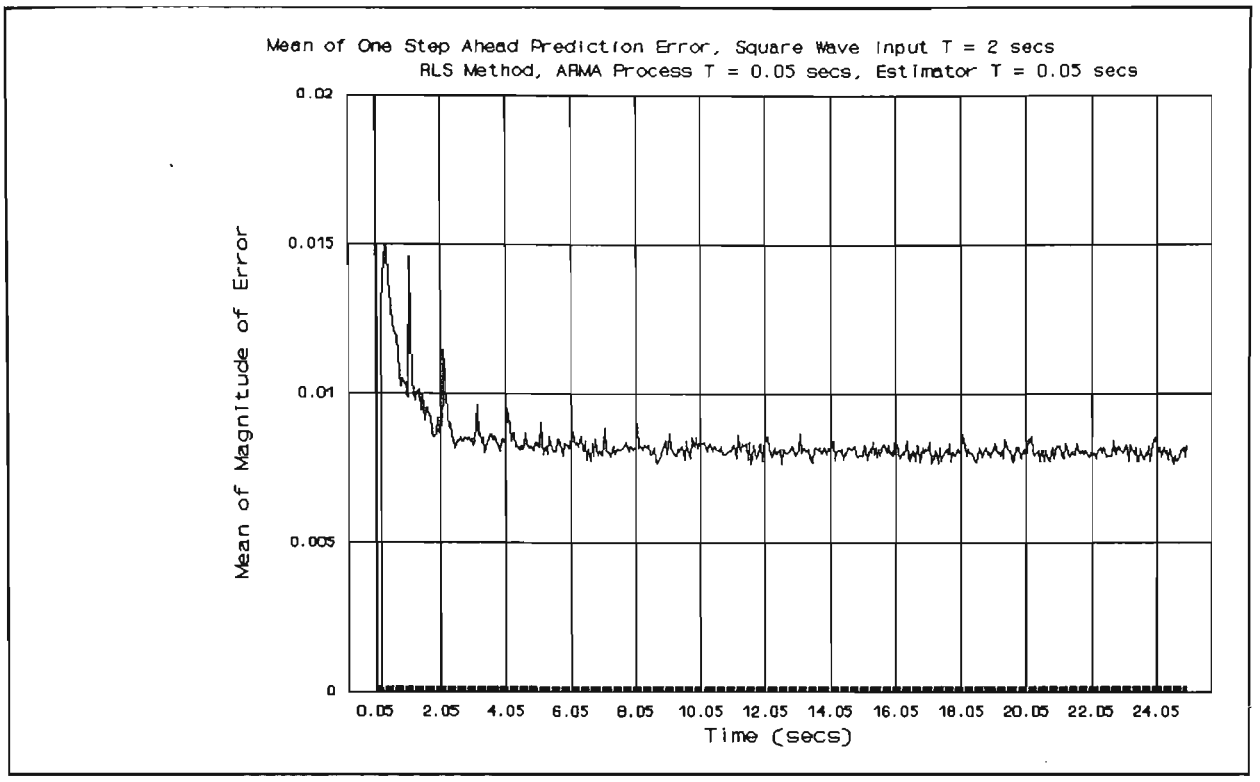

FIGURE 5-29 ERROR IN ONE-STEP AHEAD PREDICTION USING RLS METHOD WITH A HIGH SAMPLING RATE

5-6 CONCLUSIONS OF THE COMPARISON OF THE LU FACTORISATION AND RLS METHODS

The RLS Method is to be preferred over that using LU Factorisation. The main advantages of the former being its greater stability and its shorter computation time.

The LU Factorisation Method may be made more stable by the use of pivoting, yet this still would not guarantee stability [5-18].

The LU Factorisation Method did sometimes exhibit the advantage of more rapid initial convergence (compare Figures 5-11 to 5-14). This advantage over the RLS Method may be lost by increasing the value of the scaling factor used to initialise the Error Covariance Matrix used by the RLS Method [5-19].

As a consequence of the use of a "Least Squares Noise Model" (see Appendix F) in the simulations, the results suggest that a reduction of sampling period can be used to reduce the time required to bring the estimates within some chosen band of accuracy (see Table 5-5).

CHAPTER 6 A REDUCED ORDER ESTIMATOR USING A PRIORI KNOWLEDGE OF THE SYSTEM TYPE NUMBER

6-1 USE OF A PRIORI KNOWLEDGE OF PLANT INTEGRAL ACTION TO REDUCE THE ORDER OF THE ESTIMATOR

INTRODUCTION

So far in this work only the model structure was assumed to be known a priori. This means that the orders of the AR and MA sections of the ARMA Model were both known prior to the estimation process. This assumption was made to avoid the need to implement a full system identification procedure, and reduced the problem to simple parameter estimation.

The positional servomechanism considered was known to contain an integrator, and hence to have a pole at s = 0. In this chapter, this knowledge will be used to develop a reduced order estimator.

It is well known that the performance of a parameter estimator may be improved by the incorporation of a priori information [6-1,6-2,6-3,6-4]. Further, it is known that an ARMA model of a process of Type One requires one less coefficient than the ARMA model of a process of the same order, but of Type Zero [6-5]. Using this information it is possible to describe the least squares problem using one less least squares equation.

DERIVATION OF THE REDUCED ORDER ESTIMATOR

The ARMA model of the system is as given by Equations 4-17 or 4-18. A consequence of the integral action of the plant is that two of the coefficients of these equations b_1 and b_2, are dependent. This is clearly shown in Equations 4-15 and 4-16.

The reduced order estimator is developed by rewritting the Least Squares Equations in terms of the first differences of the output signal. These first differences are defined by:

$$\delta y_0 = y_0 - y_1 \qquad 6\text{-}1$$

$$\delta y_1 = y_1 - y_2 \qquad 6\text{-}2$$

Equation 4-18 may then be rewritten as:

$$y_0 = a_1 u_1 + a_2 u_2 + y_1 + b_2 \delta y_1 \qquad 6\text{-}3$$

Following the same argument given in Section 4-2, this equation may be used as a One-Step Ahead Predictor at time t = (k-1)T, to estimate the next output value y_0.

The error in estimation, ϵ_0, is now given by:

$$\epsilon_0 = y_0 - y_1 - \hat{a}_1 u_1 - \hat{a}_2 u_2 - \hat{b}_2 \delta y_1 \qquad 6\text{-}4$$

c.f. Equation 4-21.

This results in a set of Least Squares Equations of the form:

$$\begin{bmatrix} \Sigma u_1^2 & \Sigma u_1 u_2 & \Sigma u_1 \delta y_1 \\ \Sigma u_1 u_2 & \Sigma u_2^2 & \Sigma u_2 \delta y_1 \\ \Sigma u_1 \delta y_1 & \Sigma u_2 \delta y_1 & \Sigma \delta y_1^2 \end{bmatrix} \begin{bmatrix} \hat{a}_1 \\ \hat{a}_2 \\ \hat{b}_2 \end{bmatrix} = \begin{bmatrix} \Sigma u_1 \delta y_0 \\ \Sigma u_2 \delta y_0 \\ \Sigma \delta y_1 \delta y_0 \end{bmatrix} \qquad 6\text{-}5$$

c.f. Equation 4-28.

It can be shown that such a reduction in the number of Least Squares Equations can be achieved for any linear system of Type One or greater [6-7].

The above approach to simplification of the estimator is based upon applying the linear constraint on the coefficients before attempting to estimate those coefficient values. An alternative and equivalent approach would involve filtering the signal values [6-6]. The taking of the first difference of the sampled output signal is equivalent to feeding this signal through a differentiator acting as a filter to eliminate the dynamics of the known part of the plant, i.e. the integrator.

6-2 THE COMPARISON OF THE STANDARD AND REDUCED ORDER RLS METHODS OF SOLUTION

DESCRIPTION OF THE SIMULATIONS UPON WHICH THIS COMPARISON IS MADE

These simulations are based upon test input signals of the type described in Section

5-5. However to avoid duplicating results presented in that section, the following test input signals were used:

i) A square wave of period 2 seconds, switching between +5.0 and -5.0 input units (c.f. levels of +5.0 and +0.01 in Section 5-5). The first half-cycle of this waveform having the value of +5.0.

ii) A single pulse of width 10 seconds and amplitude of +5.0. This pulse falling to an "off" value of +0.01 (i.e. a wider pulse than those of Section 5-5).

RESULTS OF THE SIMULATIONS USING THE 3 & 4 PARAMETER RLS METHODS

The simulations whose results are presented below may be summarised as follows:

TABLE 6-1 SUMMARY OF RESULTS PRESENTED TO COMPARE THE 3 & 4 PARAMETER RLS METHODS, USING TEST INPUT SIGNALS

FIGURE	ESTIMATOR T (secs)	SNR (dB)	RMS NOISE	INPUT SIGNAL	METHOD
6-1	0.05	42.5	0.01	+/-5 sq wave	3RLS
6-2	0.05	42.5	0.01	+/-5 sq wave	4RLS
6-3	0.2	42.5	0.01	+/-5 sq wave	3RLS
6-4	0.2	42.5	0.01	+/-5 sq wave	4RLS
6-5	0.05	28.5	0.05	+/-5 sq wave	3RLS
6-6	0.05	28.5	0.05	+/-5 sq wave	4RLS
6-7	0.05	66.4	0.01	10 sec pls	3RLS
6-8	0.05	66.4	0.01	10 sec pls	4RLS
6-9	0.2	66.4	0.01	10 sec pls	3RLS
6-10	0.2	66.4	0.01	10 sec pls	4RLS

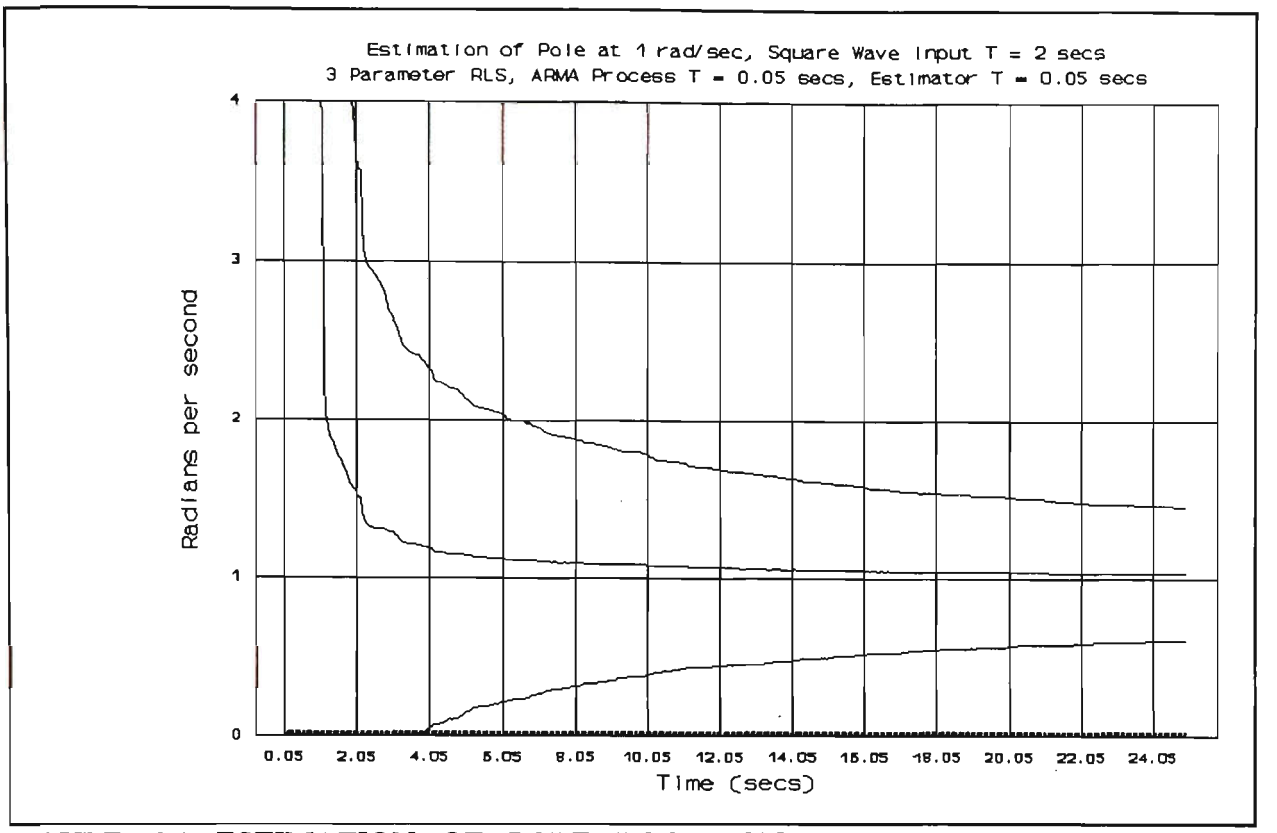

FIGURE 6-1 ESTIMATION OF POLE LOCATION USING 3 PARAMETER RLS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL

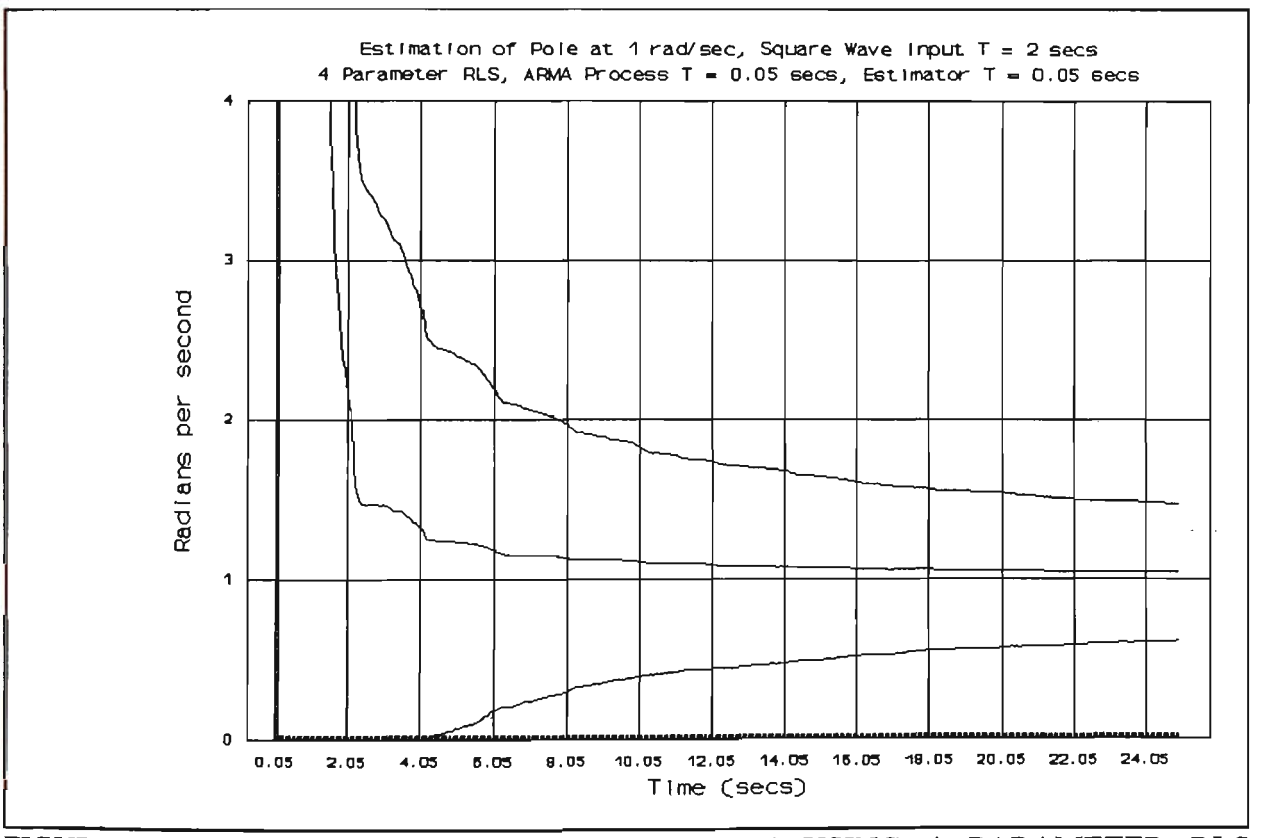

FIGURE 6-2 ESTIMATION OF POLE LOCATION USING 4 PARAMETER RLS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL

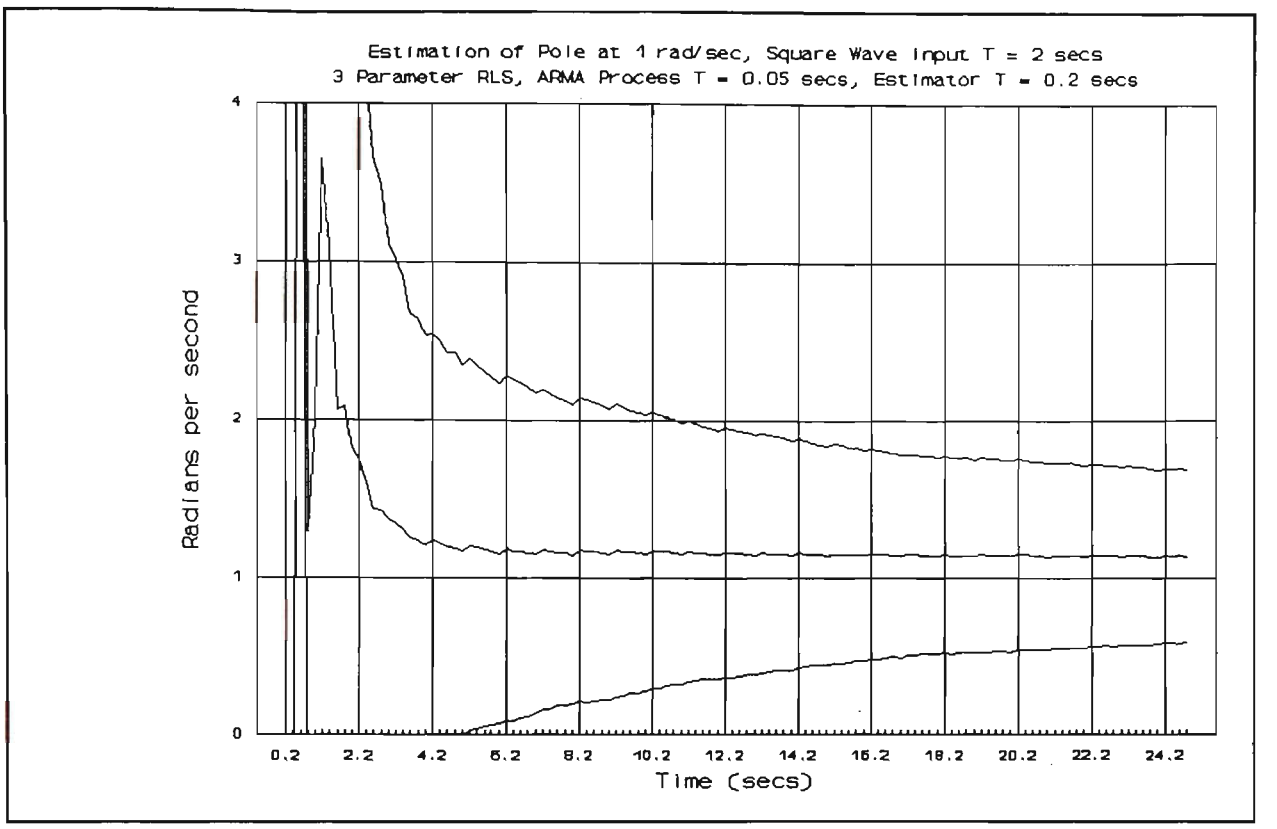

FIGURE 6-3 ESTIMATION OF POLE LOCATION USING 3 PARAMETER RLS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL WITH A LOWER SAMPLING RATE

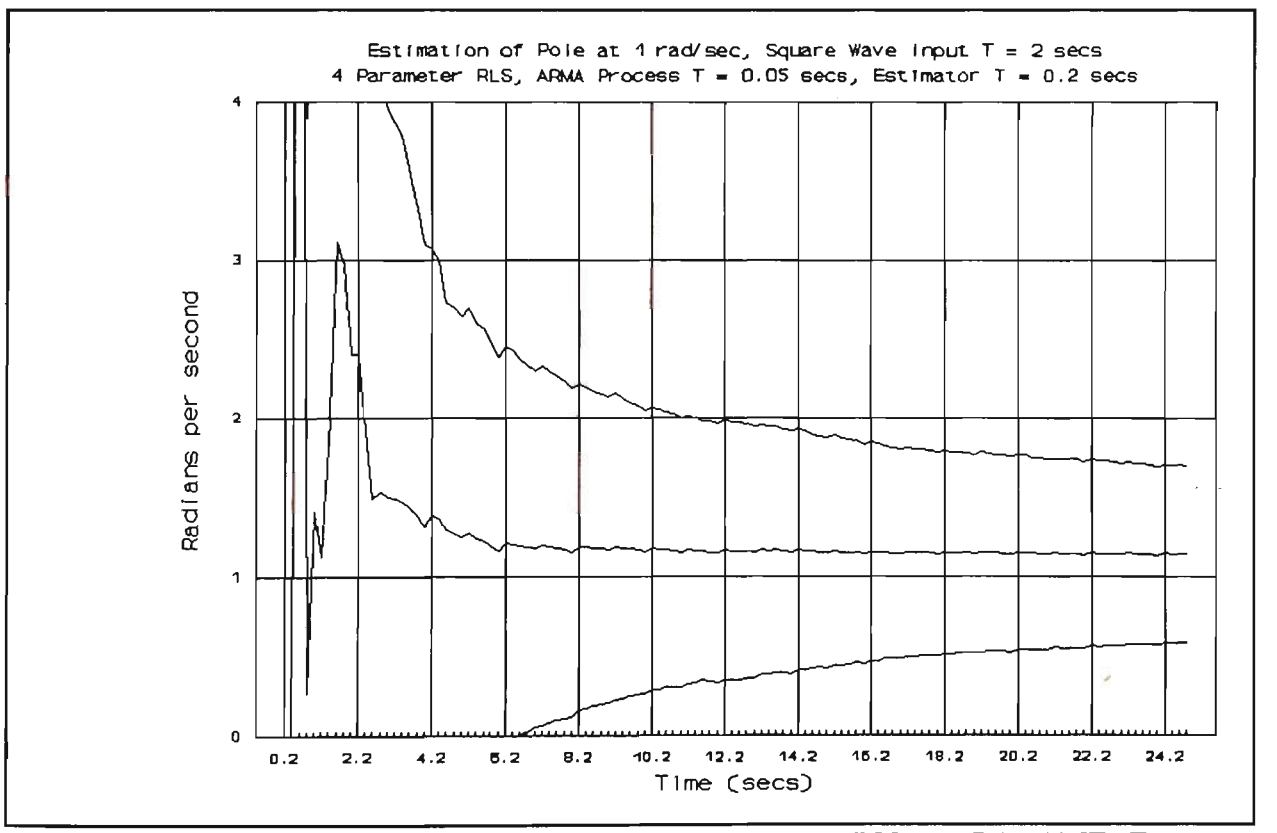

FIGURE 6-4 ESTIMATION OF POLE LOCATION USING 4 PARAMETER RLS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL WITH A LOWER SAMPLING RATE

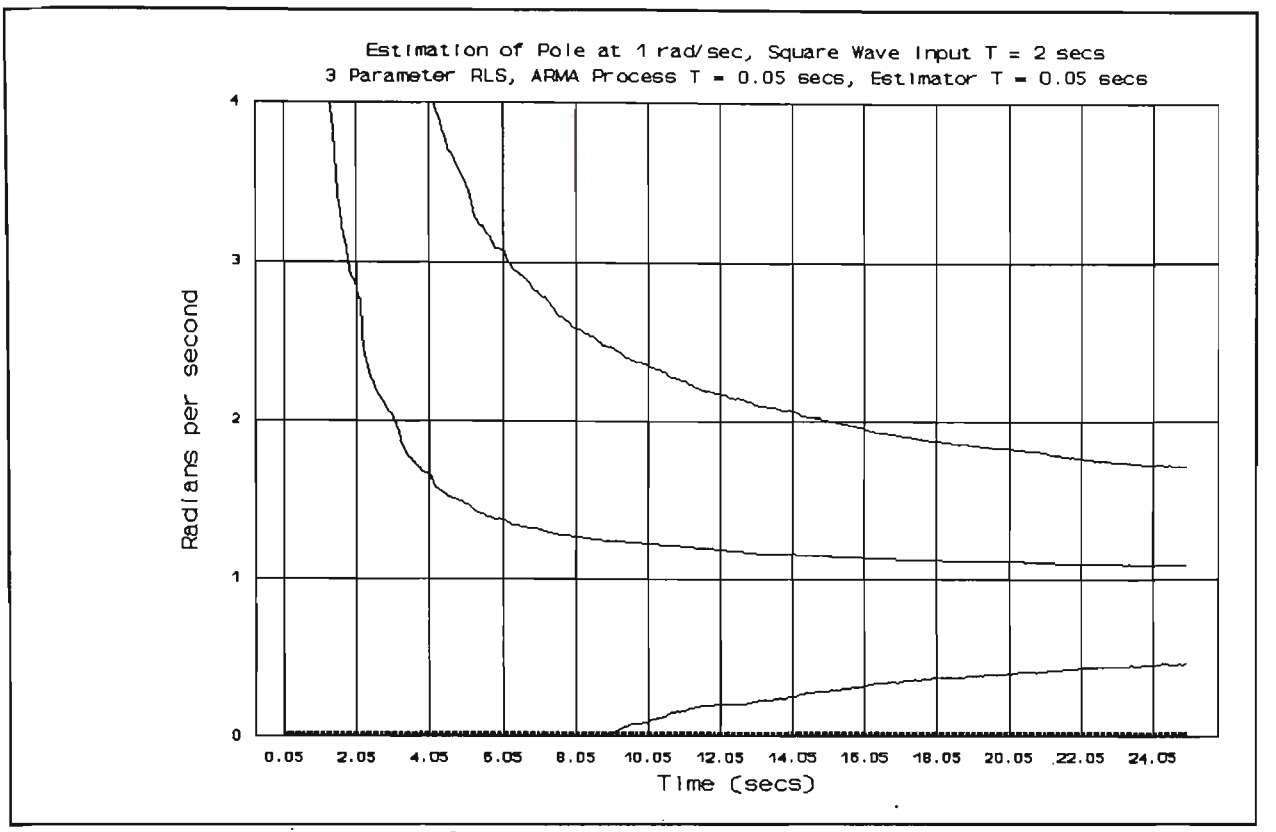

FIGURE 6-5 ESTIMATION OF POLE LOCATION USING 3 PARAMETER RLS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL WITH INCREASED NOISE

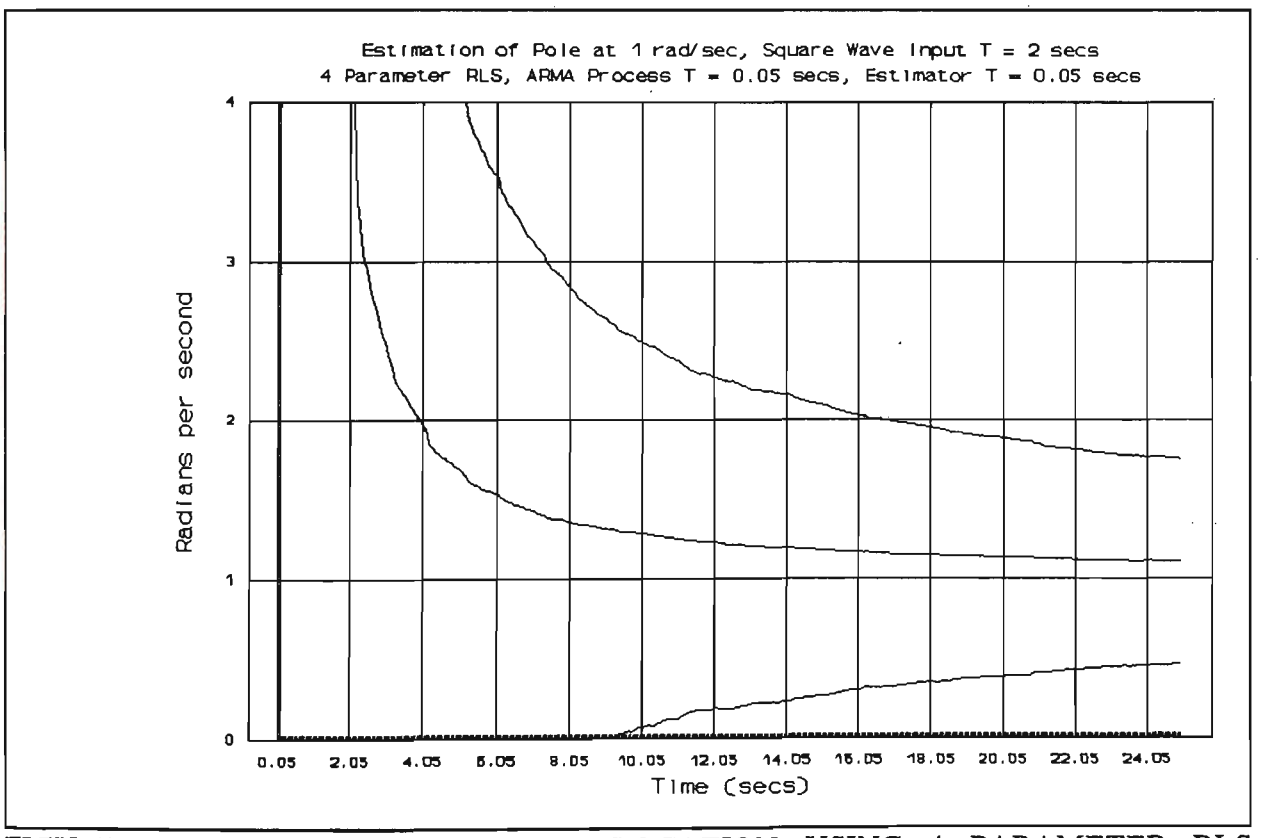

FIGURE 6-6 ESTIMATION OF POLE LOCATION USING 4 PARAMETER RLS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL WITH INCREASED NOISE

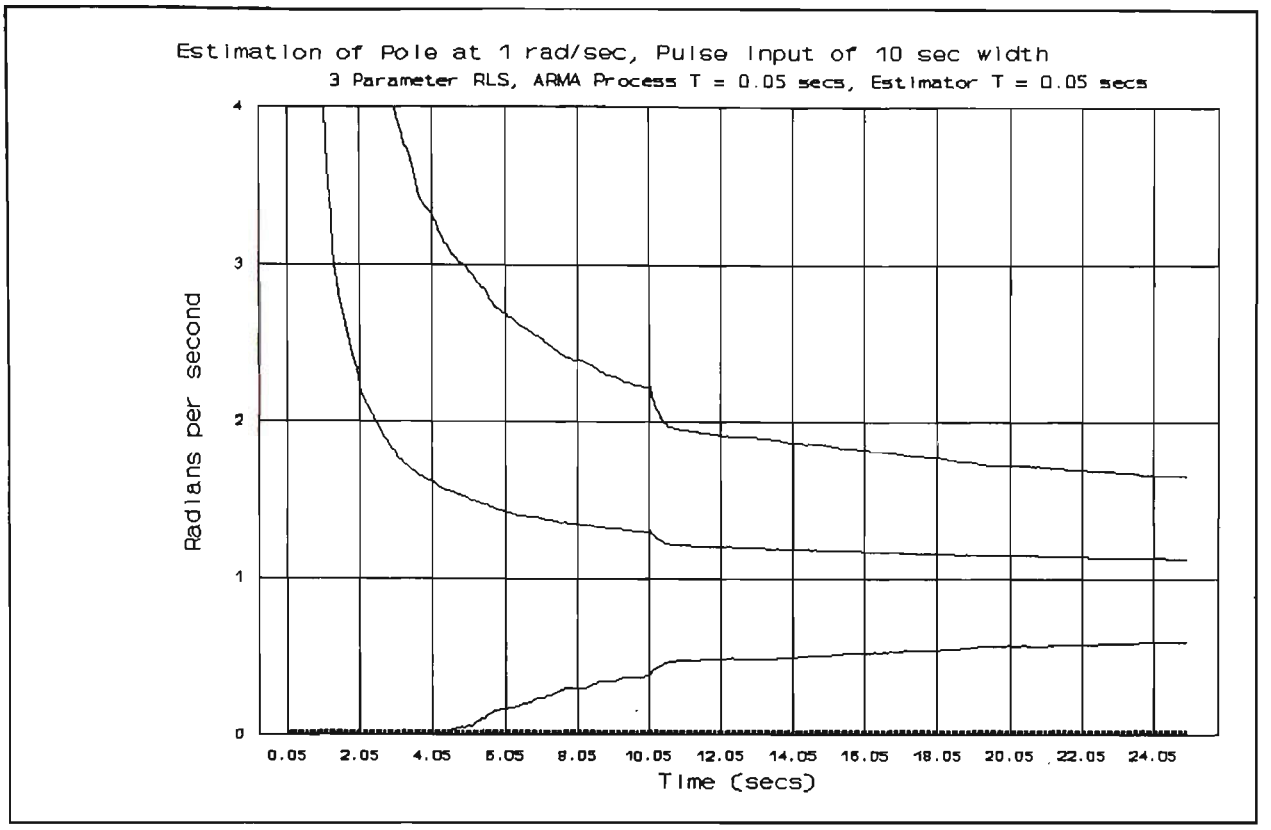

FIGURE 6-7 ESTIMATION OF POLE LOCATION USING 3 PARAMETER RLS METHOD AND A SINGLE, WIDE PULSE INPUT SIGNAL

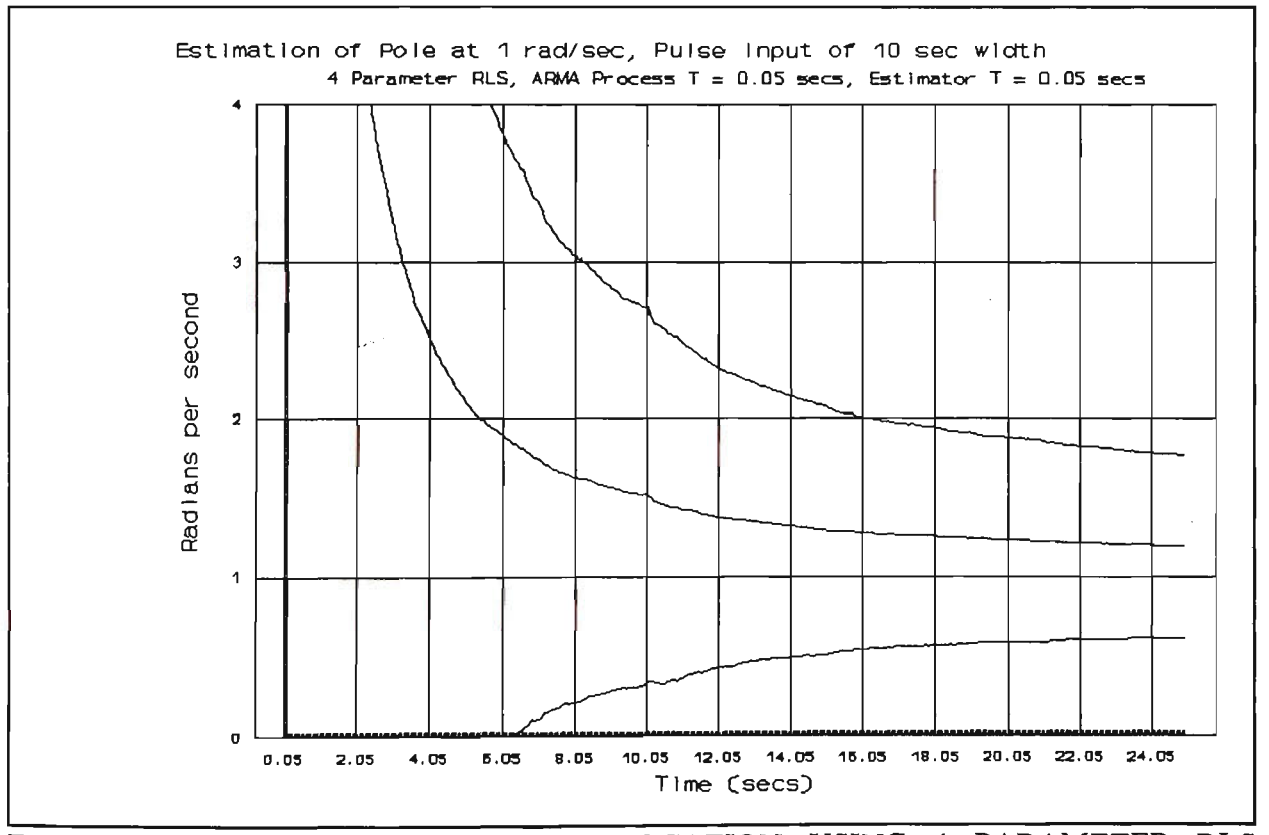

FIGURE 6-8 ESTIMATION OF POLE LOCATION USING 4 PARAMETER RLS METHOD AND A SINGLE, WIDE INPUT PULSE

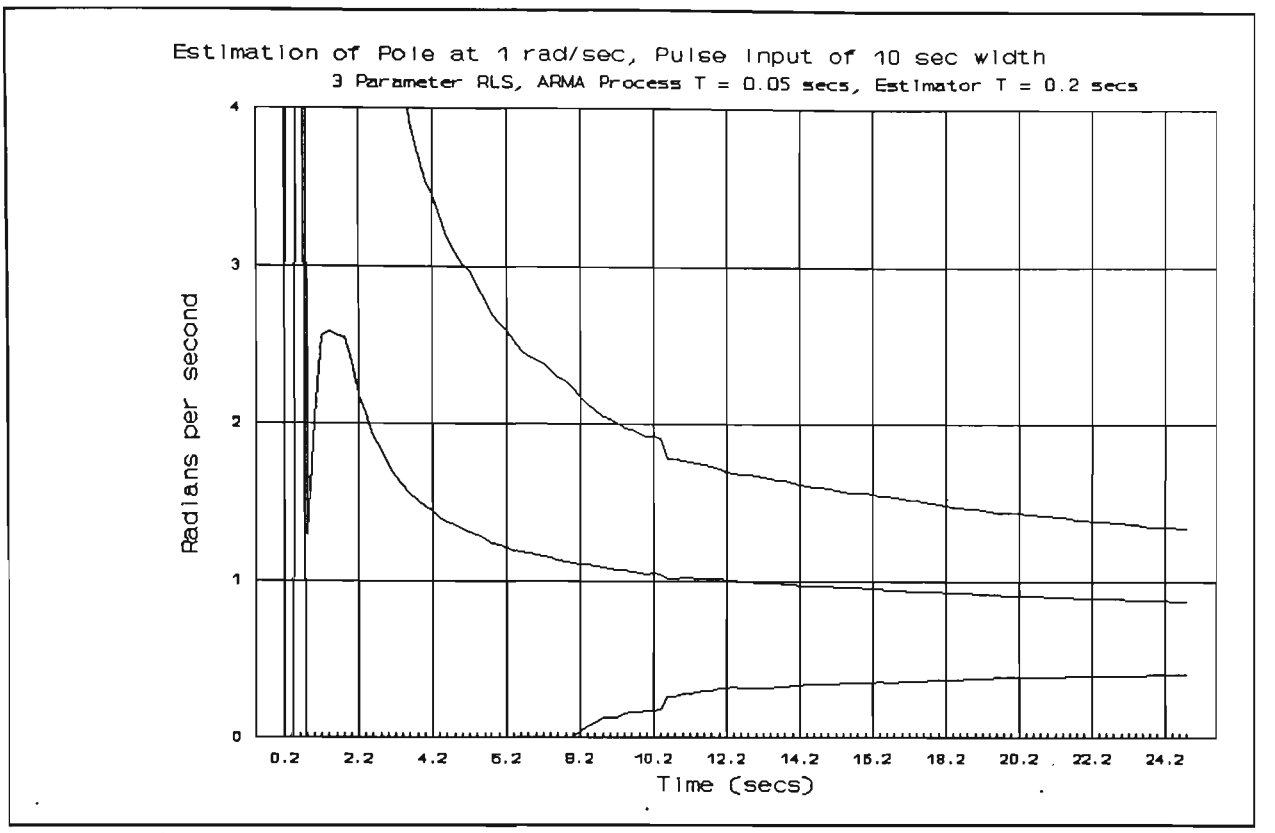

FIGURE 6-9 ESTIMATION OF POLE LOCATION USING 3 PARAMETER RLS METHOD AND A SINGLE, WIDE PULSE INPUT SIGNAL WITH A LOWER SAMPLING RATE

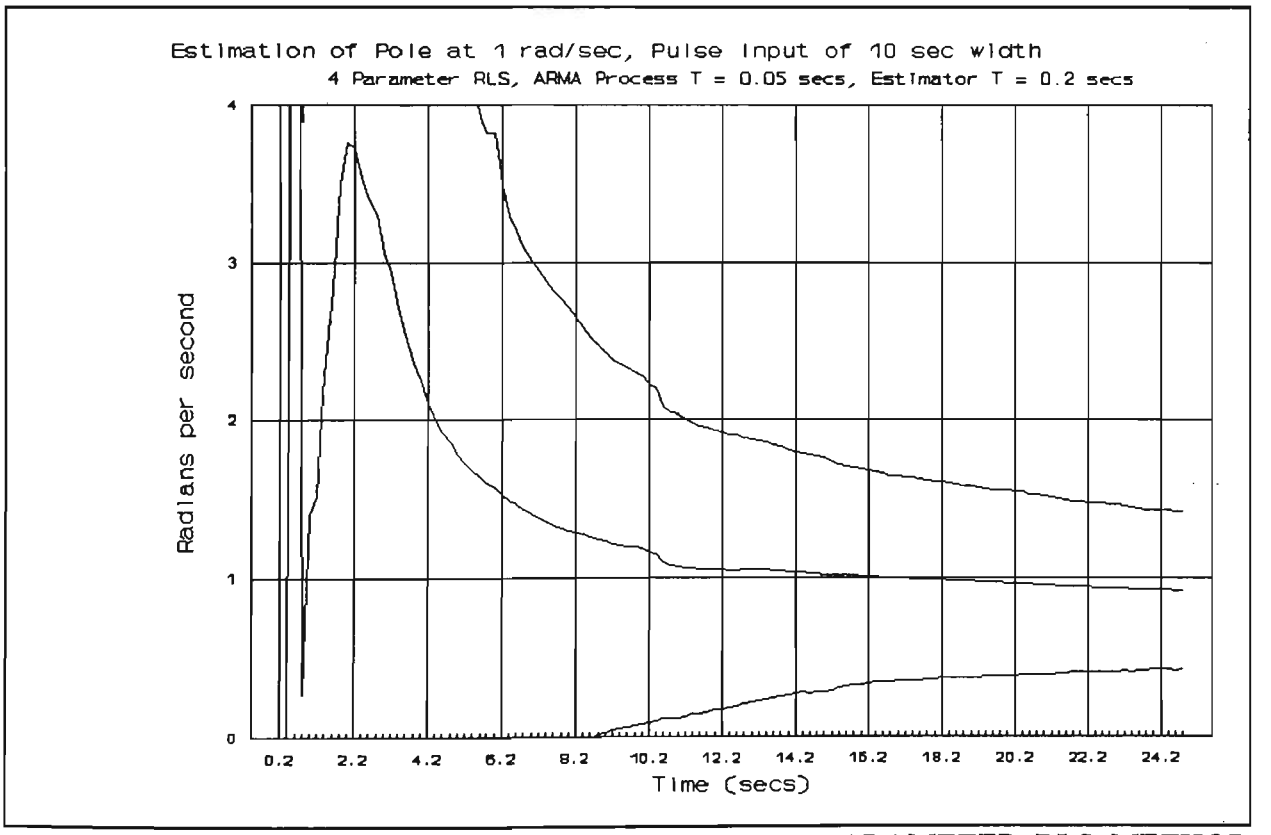

FIGURE 6-10 ESTIMATION OF POLE LOC. USING 4 PARAMETER RLS METHOD AND A SINGLE, WIDE PULSE INPUT SIGNAL WITH A LOW SAMPLING RATE

TABLE 6-2 SUMMARY OF FINAL VALUES AND CONVERGENCE RATES FOR THE 3 & 4 PARAMETER RLS SIMULATIONS

FIGURE/METHOD	ESTIMATOR PERIOD	INPUT SIGNAL	'FINAL' VALUE	CONVERGENCE TIME
6-1/ 3RLS	0.05	+/-5 sq wave	1.03	6.7
6-2/ 4RLS	0.05	+/-5 sq wave	1.03	7.8
6-3/ 3RLS	0.2	+/-5 sq wave	1.13	6.2
6-4/ 4RLS	0.2	+/-5 sq wave	1.13	7.6
6-5/ 3RLS	0.05	+/-5 sq wave	1.1	11.5
6-6/ 4RLS	0.05	+/-5 sq wave	1.1	13.0
6-7/ 3RLS	0.05	10s pulse	1.13	9.1
6-8/ 4RLS	0.05	10s pulse	1.2	11.6
6-9/ 3RLS	0.2	10s pulse	0.88	10.7
6-10/4RLS	0.2	10s pulse	0.92	13.7

RELATIVE MERITS OF THE THREE AND FOUR PARAMETER RLS METHODS

For the +/-5 square wave input signal, the 3 and 4 Parameter RLS Methods produced the same "Final Value". However in all of these simulations (Figures 6-1 to 6-6 inclusive) the 3 Parameter Estimator achieved a more rapid rate of convergence.

For the less-rich input signal of a single, ten second wide pulse, the 3 Parameter Estimator again showed better convergence. However the last pair of results (Figures 6-9 and 6-10) show that the 4 Parameter Estimator has the more accurate "Final Value". Inspection of the 'mean' traces on these figures reveals that these traces still have a noticeable gradient at the end of the simulation time. The greater the value of this "Final Gradient" the less valid the approximation of the so-called "Final Value" to the asymptotic value of the estimate.

Reduced order estimators using LU Factorisation were also tested. The computation time required by each method is summarised in Table 6-3.

TABLE 6-3 COMPUTATION TIME REQUIRED FOR THE 3 & 4 PARAMETER ESTIMATORS

METHOD	COMPUTATION TIME (microseconds)
4 Parameter LU Factorisation	112
4 Parameter RLS	93
3 Parameter LU Factorisation	66
3 Parameter RLS	61

In conclusion the Three Parameter RLS Method is clearly preferable to the Four Parameter RLS Method. It requires less computation time and delivers a more rapid rate of convergence of its estimates.

CHAPTER 7 AN INVESTIGATION OF TWO TECHNIQUES INTENDED TO REDUCE THE ESTIMATOR BIAS

7-1 INTRODUCTION

From the above simulations it appears that the mean estimate of the pole location tends towards an asymptotic value different from the known true pole location. This is clearly a failing of the estimator. In Section 7-2 some sources of the bias in the estimation process are considered. Based upon the understanding of these sources, two modifications to the basic, least squares estimation process are considered:

i) Extended Least Squares (Panuska's Method) in Section 7-4

ii) The use of Instrumental Variables in Section 7-5.

These two modified methods are compared with the Three-Parameter RLS Estimator of Chapter 6. These comparisons are again based upon computer simulations using test input signals.

7-2 SOURCES OF ESTIMATE BIAS

In practical system identification one of the main sources of bias is the excessive simplicity of the model of the system [7-1]. In the simulations used throughout this thesis the model order and structure upon which the estimator is based is the same as that of the process used to generate the output signal sequence. This knowledge that the process is no more complex than the model is a major advantage gained by basing the comparisons upon computer simulation.

In the absence of noise, the Least Squares Methods can return correct estimates of the unknown parameters within a small, finite number of sample periods. It is therefore obvious that the estimate bias is a consequence of noise corruption of the signals fed into the estimator. The requirements for non-biased estimates from Least Squares Estimators are well known [7-2]

and include:

i) Knowledge of the process order and deadtime.

ii) The input signal, and any DC component of that signal must be known.

iii) The input signal must be adequately rich.

iv) The equation error (i.e. the difference between the one-step ahead prediction of the output signal and the actual value subsequently measured) must be uncorrelated with the elements of the data (i.e. regressor) vector $\underline{\psi}$.

v) The expected value of the equation error should be zero.

vi) Noise superimposed on the output signal must be stationary.

i) and ii) present no problems in the above simulations. The simulated system and the estimator model having the same structure. Deadtime is not considered in this work. The input signal is exactly known to the estimator in all of the simulations.

iii) the effects of the "richness" of the input signal are briefly considered by examining the performance of each estimator for a number of different test input signals.

iv) As will be shown in Section 7-3 and Appendix F, the equation error is not independent of the elements of the data vector. The problem of estimate bias resulting from this dependence is the main topic of this chapter.

v) Since the above simulations do not provide all of the necessary conditions for unbiased least squares estimates, it would be unreasonable to expect the equation error to have an Expected Value of zero.

vi) The one PRBS, with a constant scaling factor was used throughout each simulation to generate the noise sequence $\{n(kT)\}$.

7-3 ESTIMATE BIAS AS A CONSEQUENCE OF CORRELATED NOISE

There are numerous models that describe the effects of noise upon a discrete-time

process. The above simulations treated the noise in the manner of an "Equation Error Model" [7-3]. This is equivalent to the "LS Model" of noise [7-7], as justified in Appendix F.

In practical system identification the sources of noise may be far more complex than the simple added noise sequence of the above simulations. Hence in practical work it may well be necessary to provide a non-constrained, general noise model. Such a general model should be able to mould itself to at least part of any hidden structure of the noise process of the real system. For the above simulations the noise process is well understood, and accordingly a general noise model is not necessary.

The noise process in the simulations is that of the Least Square Noise Model, in which the elements of the output noise sequence, {n(kT)}, are passed down the Data (Regressor) Vector with their associated output values. Each element of the noise sequence thus not only corrupts its associated, contemporaneous output sequence element, but also all the subsequent elements of the output signal sequence. In this manner the noise becomes correlated with the output signal sequence, this process is examined in greater detail in Appendix F. Having been shifted down the Data Vector, $\underline{\psi}$, the noise is able to enter the Information Matrix (or conversely the Error Covariance Matrix) of the Least Squares Estimator.

For simplicity, consider the effect of this noise on the elements of the Information Matrix of a non-recursive estimator. Clearly any noise corruption of either y_1 or δy_1 will result in a positive error being added to the corresponding Σy_1^2 or $\Sigma \delta y_1^2$ terms of the Information Matrix.

Further, since the noise corruption of y_1 will include a component due to the noise corruption of y_2, then the terms such as $\Sigma y_1 y_2$ in the Information Matrix (or $\Sigma \delta y_1 \delta y_0$ in the case of the reduced estimator) will also suffer from a cumulative error due to noise. The cumulative build up of these error terms in the Information Matrix result in the Least Squares Estimator producing biassed estimates.

On the basis of this understanding several attempts were made to reduce the magnitude of the bias.

The parameters of the chosen noise model are a subset of those of the process model and it is tempting to believe that simple corrections may be made to the corruptible elements of the Information Matrix. Such an approach is attractive, since it does not require an increase in the number of parameters to be estimated.

Unfortunately the prediction error has two sources. One is the added output noise sequence, the other is the errors in the estimates of the parameters. As a consequence of this it is not possible to directly obtain the values of the noise sequence from the prediction error sequence. Thus a simple modification to the Information Matrix is not possible.

System Identification literature contains several techniques for dealing with the problem of correlated residuals [2-15]. Of these standard techniques two were tested to see whether or not they were useful in the reduction of the bias problem. The first of the techniques investigated is that known as either "Extended Least Squares" or "Panuska's Method" [7-11]. The second of the techniques investigated concerned the modification of the Information Matrix by the use of various sets of instrumental variables.

7-4 EXTENDED LEAST SQUARES (ELS).

INTRODUCTION TO THE ELS METHOD

This method requires that the data and parameter vectors be extended to accommodate extra error signal terms.

In Appendix F it is shown that the output signal with noise y_{0n}, is equal to the sum of a virtual, noise-free output y_0, and an additional error term e_0.

i.e. Equation F-8, rewritten here as:

$$y_{0n} = y_0 + e_0 \tag{7-1}$$

where

$$e_0 = -b_1 e_1 - b_2 e_2 + n_0 \tag{7-2}$$

Substituting for y_0 gives:

$$y_{0n} = a_1 u_1 + a_2 u_2 - b_1 y_1 - b_2 y_2 - b_1 e_1 - b_2 e_2 + n_0 \tag{7-3}$$

n_0 is the most recent term of the noise sequence {n(kT)}, and hence is not predictable. The Extended Least Squares Method does however attempt to model the other 'error terms'by extending the data vector:

$$\underline{\psi_k}^T = [\ u_1\ u_2\ -y_1\ -y_2\ e_1\ e_2\ ] \tag{7-4}$$

The Parameter Vector must also be extended to match the Data Vector

$$\underline{\theta}^T = [\ a_1\ a_2\ b_1\ b_2\ c_1\ c_2\ ] \tag{7-5}$$

It should be noted that the method does not make the constraints $b_1 = c_1$ and $b_2 = c_2$ suggested by Equation 7-3.

Panuska [7-11] proposed that the sequence {e(kT)} should be a sequence of "computed errors". There are two obvious sources for such a sequence:

i) The Error in the One-Step Ahead Prediction of the output sequence

ii) The Residual Sequence [7-5].

The means by which the elements of these two "computed error" sequences may be obtained are discussed next.

COMPUTED ERRORS BASED UPON THE ERROR IN THE ONE-STEP AHEAD PREDICTION

Using the extended vectors of Equations 7-4 and 7-5, the next predicted output signal $\hat{y}_0$, is given by

$$\hat{y}_0 = \hat{a}_1(k-1)u_1 + \hat{a}_2(k-1)u_2 - \hat{b}_1(k-1)y_1 - \hat{b}_2(k-1)y_2 + \hat{c}_1(k-1)\hat{e}_1 + \hat{c}_2(k-1)\hat{e}_2 \tag{7-6}$$

where $\hat{e}_1$ and $\hat{e}_2$ are delayed versions of the chosen computed error, and $\hat{a}_1(k-1)$ denotes the estimate of a_1 using information available up to and including t = (k-1)T.

When the next output value, y_0, is subsequently measured the Prediction Error, ϵ_0 may be found

$$\epsilon_0 = y_0 - \hat{y}_0 \qquad 7\text{-}7$$

This prediction error is then one possible source of the computed error terms, i.e.

$$e_0 = \epsilon_0 \qquad 7\text{-}8$$

COMPUTED ERRORS BASED UPON THE RESIDUAL SEQUENCE

After y_0 has been measured the parameter estimates may be updated. These posterior estimates may then be used to recalculate (or filter) the expected output value. This filtered value is here denoted by $\hat{y}_{0f}$, and hence

$$\hat{y}_{0f} = \hat{a}_1(k)u_1+\hat{a}_2(k)u_2-\hat{b}_1(k)y_1-\hat{b}_2(k)y_2+\hat{c}_1(k)\hat{e}_1+\hat{c}_2(k)\hat{e}_2 \qquad 7\text{-}9$$

The Residual Error, $\hat{\epsilon}_0$, is defined such that

$$\hat{\epsilon}_0 = y_0 - \hat{y}_{0f} \qquad 7\text{-}10$$

and is the second possible source of computed error terms, i.e.

$$e_0 = \hat{\epsilon}_0 \qquad 7\text{-}11$$

The Least Squares Noise Model suggests that the coefficients b_1 and b_2 should equal c_1 and c_2 respectively. Several attempts were made to force this constraint upon the estimator, none of which resulted in an improved algorithm.

RESULTS OF THE SIMULATIONS USING THE ELS METHOD

The simulations whose results are presented below may be summarised as follows:

TABLE 7-1 SUMMARY OF RESULTS PRESENTED USING THE 6 PARAMETER RECURSIVE EXTENDED LEAST SQUARES METHOD

FIGURE	ESTIMATOR T	SNR (dB)	RMS NOISE	INPUT SIGNAL
7-1	0.05	64.9	0.01	+/-5 sq wave
7-2	0.2	64.9	0.01	+/-5 sq wave
7-3	0.05	50.9	0.05	+/-5 sq wave
7-4	0.05	66.4	0.01	10 sec pls
7-5	0.2	66.4	0.01	10 sec pls

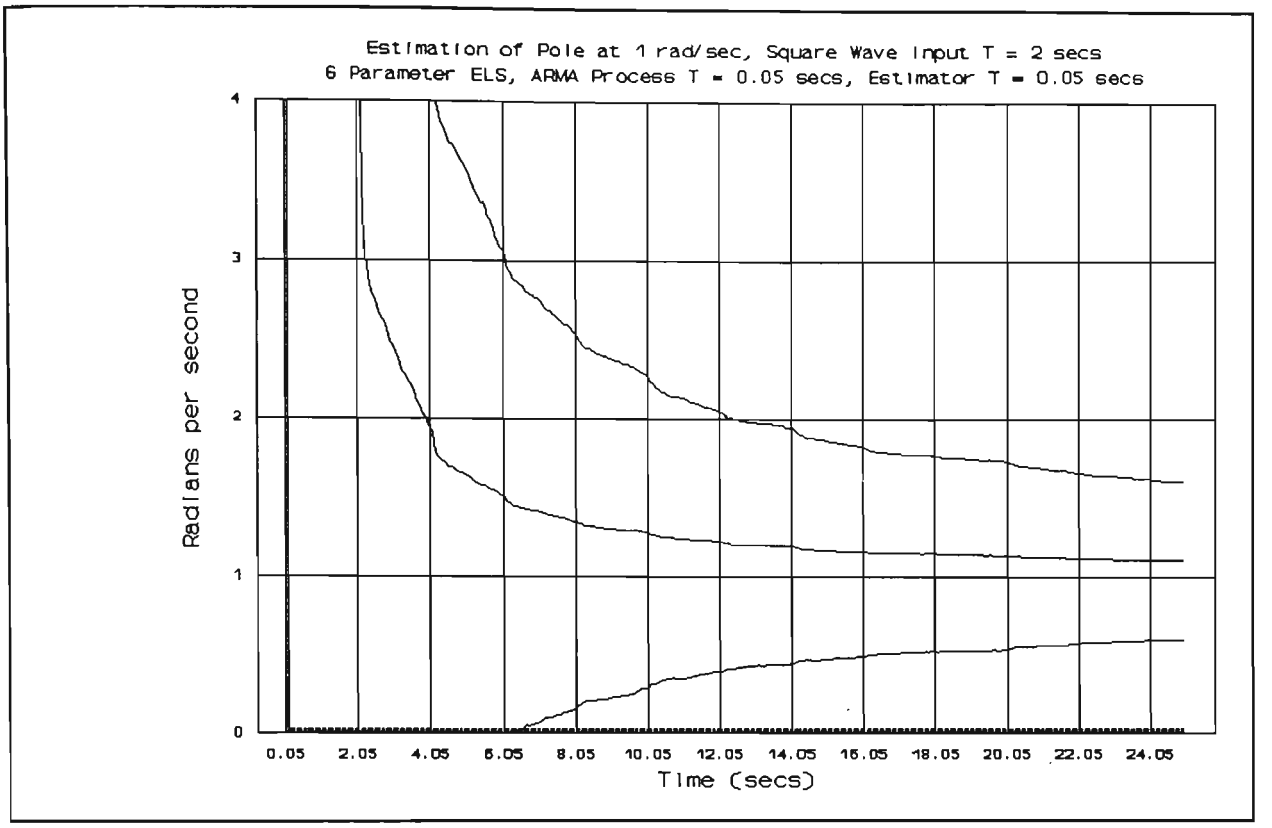

FIGURE 7-1 ESTIMATION OF POLE LOCATION USING 6 PARAMETER ELS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL

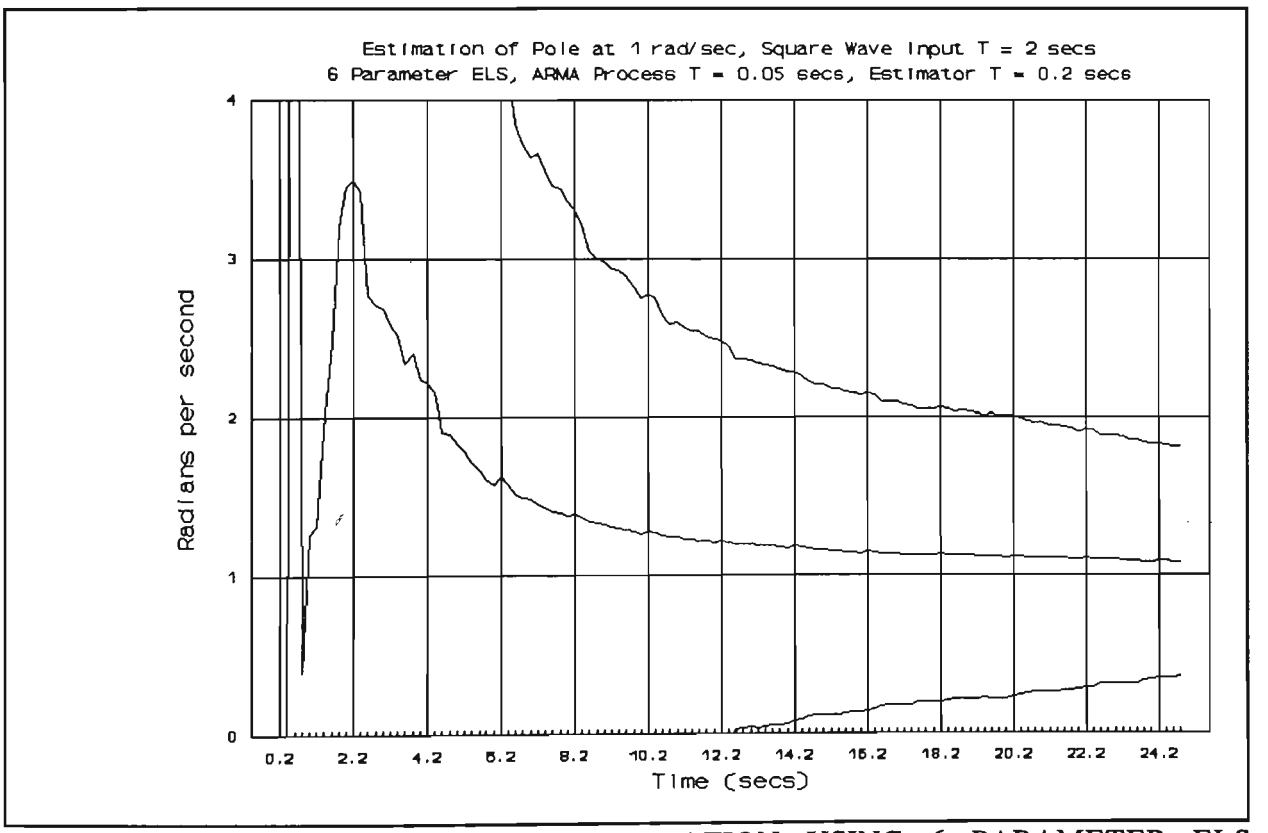

FIGURE 7-2 ESTIMATION OF POLE LOCATION USING 6 PARAMETER ELS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL WITH A LOWER SAMPLING RATE

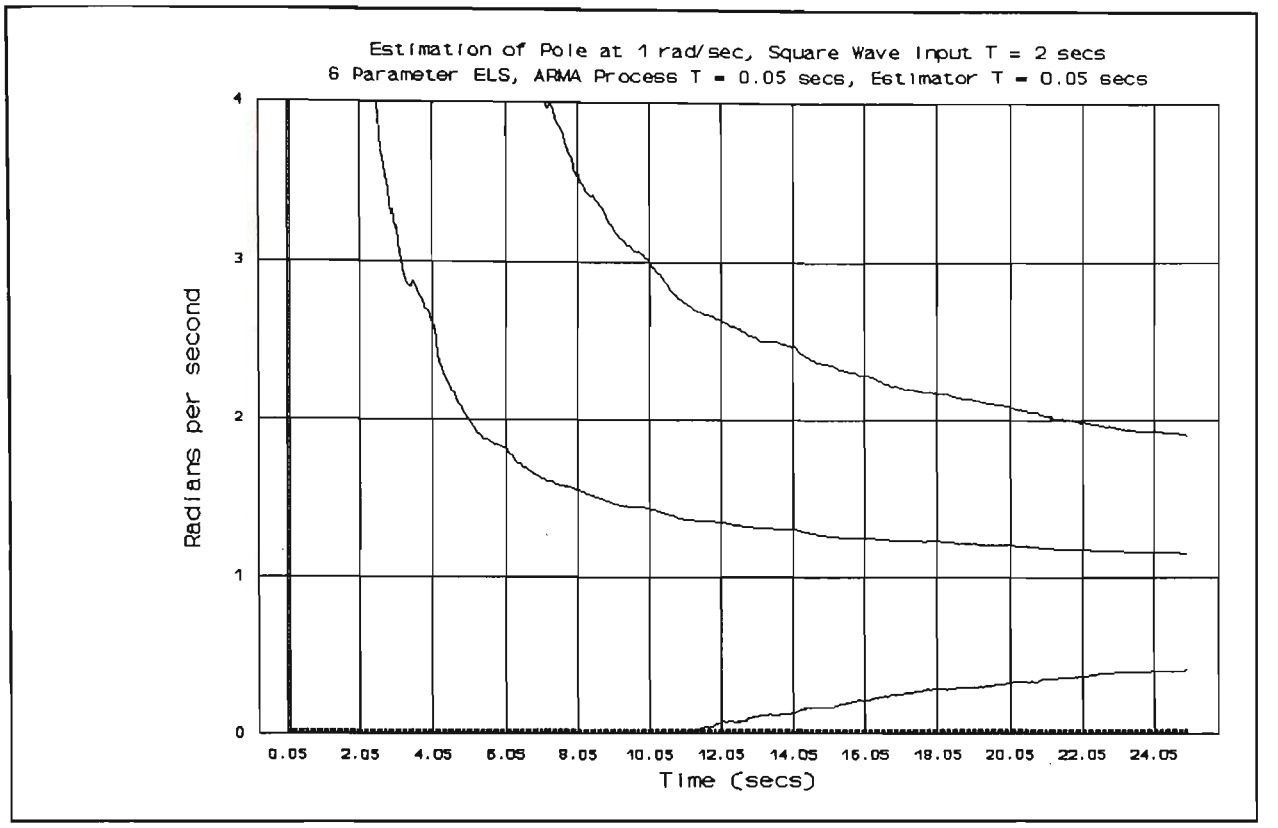

FIGURE 7-3 ESTIMATION OF POLE LOCATION USING 6 PARAMETER ELS METHOD AND A +/-5 SQUARE WAVE INPUT SIGNAL WITH INCREASED NOISE

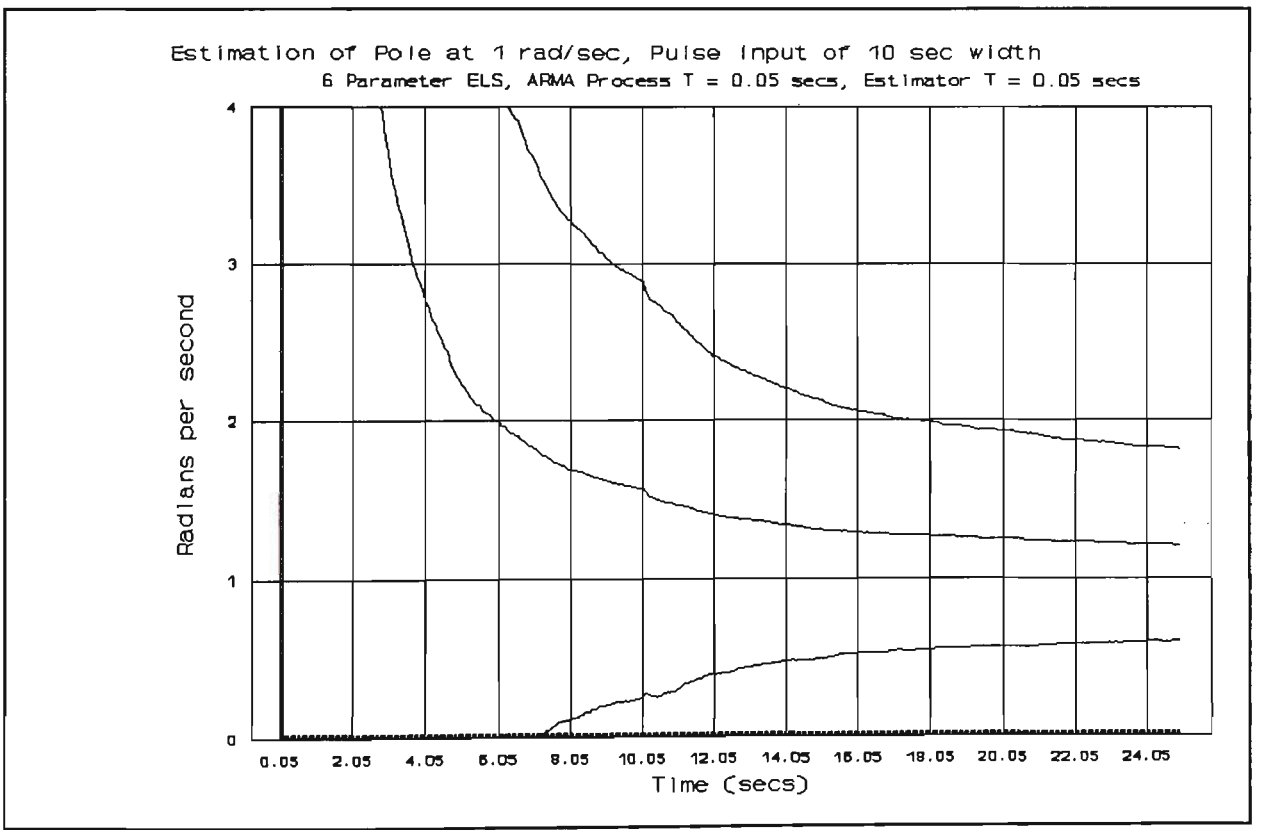

FIGURE 7-4 ESTIMATION OF POLE LOCATION USING 6 PARAMETER ELS METHOD AND A SINGLE, WIDE PULSE INPUT SIGNAL

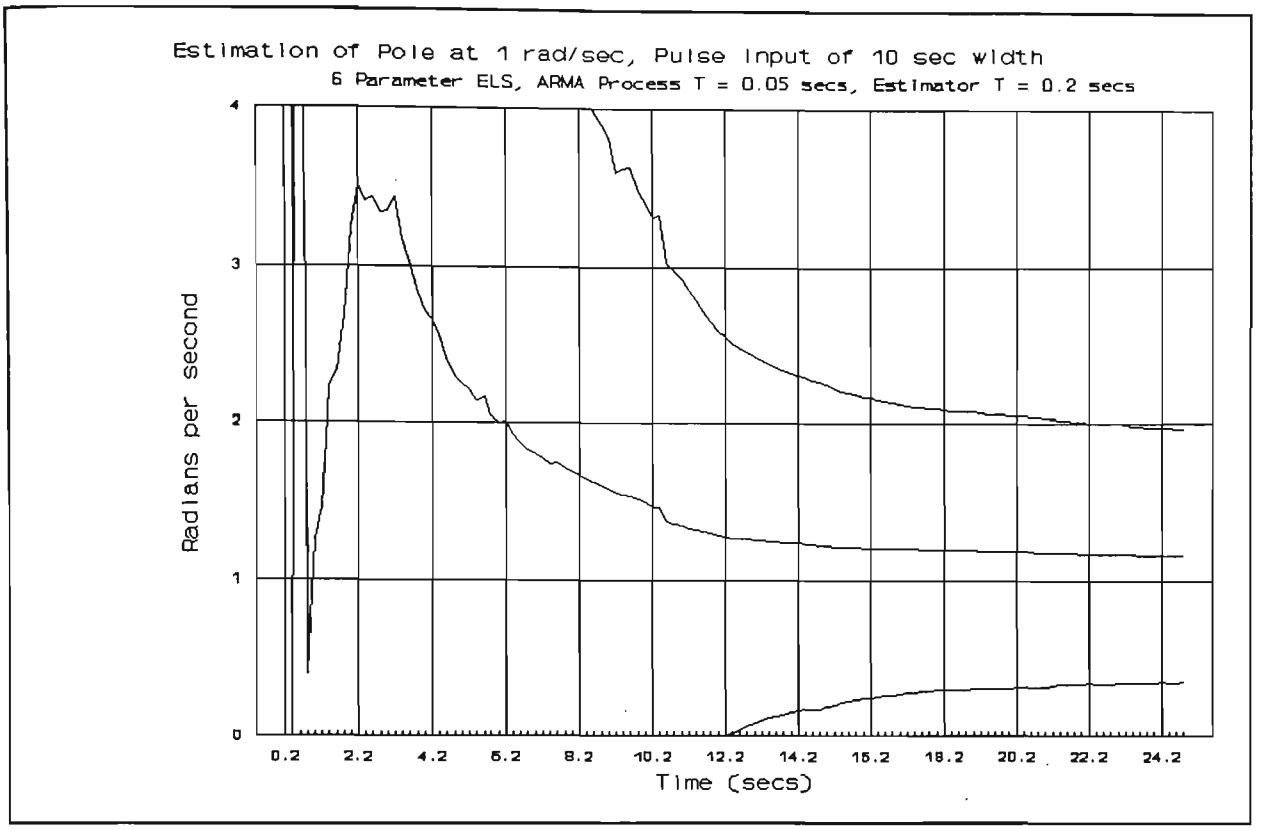

FIGURE 7-5 ESTIMATION OF POLE LOCATION USING 6 PARAMETER ELS METHOD AND A SINGLE, WIDE PULSE INPUT SIGNAL WITH A LOWER SAMPLING RATE

RELATIVE MERITS OF THE THREE PARAMETER RLS AND SIX PARAMETER ELS METHODS

In all of the cases presented, and all of the other simulations performed, the 3 Parameter RLS Method produced a more rapid convergence of the estimates, as shown in the "Convergence Time" column of Table 7-2.

In the majority of cases, the 3 Parameter RLS Method produced a more accurate 'Final Value' than did the 6 Parameter ELS Method.

The 6 Parameter ELS Method required a computation time of 196 microseconds, compared with the 61 microseconds required by the 3 Parameter RLS Method.

Over the period covered by the simulations, the 6 Parameter ELS Method failed to deliver any significant advantage over the basic 3 Parameter RLS Method in terms of a reduction in the estimate bias. This is consistent with the earlier comments of Section 5-3

concerning the 'Bias of the Estimates'. In Section 5-3 it was noted that the bias of the estimates, if present, was not the major contributor to the RMSE of the estimates. It would therefore be expected that a method designed to reduce bias would fail to deliver a significant improvement in the estimation process.

TABLE 7-2 SUMMARY OF FINAL VALUES AND CONVERGENCE RATES FOR THE 3-PARAMETER RLS AND 6-PARAMETER ELS SIMULATIONS

FIGURE/METHOD	ESTIMATOR PERIOD	INPUT SIGNAL	'FINAL' VALUE	CONVERGENCE TIME
7-1/6ELS	0.05	+/-5 sq wv	1.10	10.15
6-1/3RLS	0.05	+/-5 sq wv	1.03	6.7
7-2/6ELS	0.2	+/-5 sq wv	1.08	15.28
6-3/3RLS	0.2	+/-5 sq wv	1.13	6.2
7-3/6ELS	0.05	+/-5 sq wv	1.17	15.05
6-5/3RLS	0.05	+/-5 sq wv	1.10	11.5
7-4/6ELS	0.05	10 sec pls	1.20	12.05
6-7/3RLS	0.05	10 sec pls	1.13	9.1
7-5/6ELS	0.2	10 sec pls	1.17	13.97
6-9/3RLS	0.2	10 sec pls	0.88	10.7

7-5 INSTRUMENTAL VARIABLE TECHNIQUES

INTRODUCTION TO INSTRUMENTAL VARIABLES

The second approach to reducing the bias in the estimates involves the use of instrumental variables [7-9, 7-13] .

The use of instrumental variables was initially criticised for its failure to consider a

noise model [2-15]. By 1976 the Refined Instrumental Variable Method had been developed to include an ARMA noise model [7-9]. For the simulations considered above, the introduction of such a general noise model would be inappropriate.

A further criticism of the use of instrumental variables concerns their use in parameter-adaptive control systems (i.e. a combination of a parameter estimator and a controller design to obtain a closed loop system). In such applications the instrumental variable methods require a perturbation signal in the loop, or will produce biased estimates [7-10]. This criticism is not a problem for the work considered in this thesis, but it is mentioned as a possible problem in applications of instrumental variable methods.

THE ORDINARY IV METHOD AS A MEANS OF REDUCING ESTIMATE BIAS

Equation 4-28 may be rewritten in the form :

$$\left(\sum_{k=1}^{n} \underline{\Psi}_k \cdot \underline{\Psi}_k^{\,T}\right) \cdot \underline{\theta}_n = \sum_{k=1}^{n} \underline{\Psi}_k \cdot y_k \qquad 7\text{-}12$$

where $\underline{\Psi}_k^{\,T}$ is defined by Equation 4-43, and is the signal vector. The parameter vector $\underline{\theta}$, is defined such that:

$$\underline{\theta}_n^{\,T} = [\hat{a}_1 \ \hat{a}_2 \ \hat{b}_1 \ \hat{b}_2] \qquad 7\text{-}13$$

where the 'carat' denotes an estimate of that coefficient.

With the Ordinary IV Method [7-9] Equation 7-12 is replaced by:

$$\left(\sum_{k=1}^{n} \hat{\underline{\Psi}}_k \cdot \underline{\Psi}_k^{\,T}\right) \cdot \underline{\theta}_n = \sum_{k=1}^{n} \hat{\underline{\Psi}}_k \cdot y_k \qquad 7\text{-}14$$

where the "IV vector", $\hat{\underline{\Psi}}_k$, is given by:

$$\hat{\underline{\Psi}}_k = [u_1 \ u_2 \ -\hat{y}_1 \ -\hat{y}_2] \qquad 7\text{-}15$$

This vector contains instrumental variables in place of the output values of the signal vector. There are numerous techniques for generating instrumental variables [7-6, 7-9]. Two of

these techniques were tested. The first of these generated the instrumental variables by applying the known input sequence into an adaptive filter. The second approach was to use a delayed version of the output sequence as the source of instrumental variables [7-12].

In all cases the estimator was initially run without the introduction of the instrumental variablesso as to retain the rapid convergence of the basic least squares solution. Further, for the technique involving an adaptive filter, this initial non-use of instrumental variables allowed the efficient initialisation of the coefficient values of the filter.

In the subsequent work the Ordinary IV Method will be implemented on the "reduced" information matrix derived in Chapter Six.

Equation 6-5 may be rewritten in the form of Equation 7-12, in which case:

$$\underline{\Psi}_k^T = [u_1 \; u_2 \; \delta y_1] \tag{7-16}$$

and

$$\underline{\theta}_n^T = [\hat{a}_1 \; \hat{a}_2 \; \hat{b}_2] \tag{7-17}$$

Hence it is necessary to generate an "IV vector" of the form:

$$\underline{\hat{\Psi}}_k^T = [u_1 \; u_2 \; \delta \hat{y}_1] \tag{7-18}$$

where

$$\delta \hat{y}_1 = \hat{y}_1 - \hat{y}_2 \tag{7-19}$$

The modified least squares equations can thus be expressed as:

$$\begin{bmatrix} \Sigma u_1^2 & \Sigma u_1 u_2 & \Sigma u_1 \delta y_1 \\ \Sigma u_1 u_2 & \Sigma u_2^2 & \Sigma u_2 \delta y_1 \\ \Sigma u_1 \delta \hat{y}_1 & \Sigma u_2 \delta \hat{y}_1 & \Sigma \delta y_1 \delta \hat{y}_1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ b_2 \end{bmatrix} = \begin{bmatrix} \Sigma u_1 \delta y_0 \\ \Sigma u_2 \delta y_0 \\ \Sigma \delta \hat{y}_1 \delta y_0 \end{bmatrix} \tag{7-20}$$

RESULTS OF THE SIMULATIONS USING INSTRUMENTAL VARIABLES

Simulations were performed using a number of different techniques for generating the instrumental variables. The results of these simulations revealed some significant problems in the choice of the instrumental variables. In all cases the estimators used instrumental variables

within an Information Matrix, as the basis of a solution using LU Factorisation.

The first attempts at an estimator using instrumental variables generated these variables by an adaptive filter in the form of an Auxilliary Adaptive Model of the unknown system. This approach has two major problems:

i) the initial convergence of the estimates was slowed. The adaptive filter necessarily having to use previously estimated values of the parameters.

ii) the estimators proved to be highly sensitive to both the values of the initial parameters of the adaptive filter, and to the timing of the input sequence applied to it. The consequence of an unfortunate choice in either of these resulted in a systematic 'bursting' of short duration but large errors in the estimates. The bursts appeared as narrow, large amplitude spikes on an otherwise well-behaved RMSE vs. Time characteristic.

Attempts to improve the performance of the estimators by low-pass filtering of the parameters of the adaptive, auxilliary model [7-8] were not succesful. The 'bursting' phenomenon raises serious doubts about the stability of estimators using instrumental variables derived from an adaptive filter.

Greater success was achieved by using 'Delayed Signal' sequences as the source of the instrumental variables. The simulation result presented in Figure 7-6 below, used a delayed version of the $\{\delta y(kT)\}$ sequence.

It was found that the rate of convergence of the estimates was improved by initially running the estimator without the use of instrumental variables. Instrumental variables were then allowed to enter the Information Matrix once the estimates approached their asymptotic value.

From the simulations it was found that, over the period of the simulation, the 3 Parameter RLS Method consistently outperformed all of the methods using instrumental variables.

Only one result from the simulations using instrumental variables is presented here, as Figure 7-6. This simulation uses the same combination of test input signal, sampling rate and process model that was used to produce the results of Figure 5-24. The estimator in Figure 5-24 was the basic 4 Parameter LU Factorisation Method, and in the simulation it failed to maintain stability. Using LU Factorisation, but with only 3 parameters and instrumental variables, the new estimator maintained its stability throughout the duration of the simulation.

The simulations whose results are presented below may be summarised as follows:

TABLE 7-3 SUMMARY OF RESULTS PRESENTED USING AN INSTRUMENTAL VARIABLE PARAMETER ESTIMATOR

FIGURE/METHOD	ESTIMATOR PERIOD, T	SNR (dB)	RMS NOISE	INPUT SIGNAL
7-6/3IV(LU)	0.05	64.9	0.01	+5/+0.01 sq wv
7-7/3RLS	0.05	64.9	0.01	+5/+0.01 sq wv

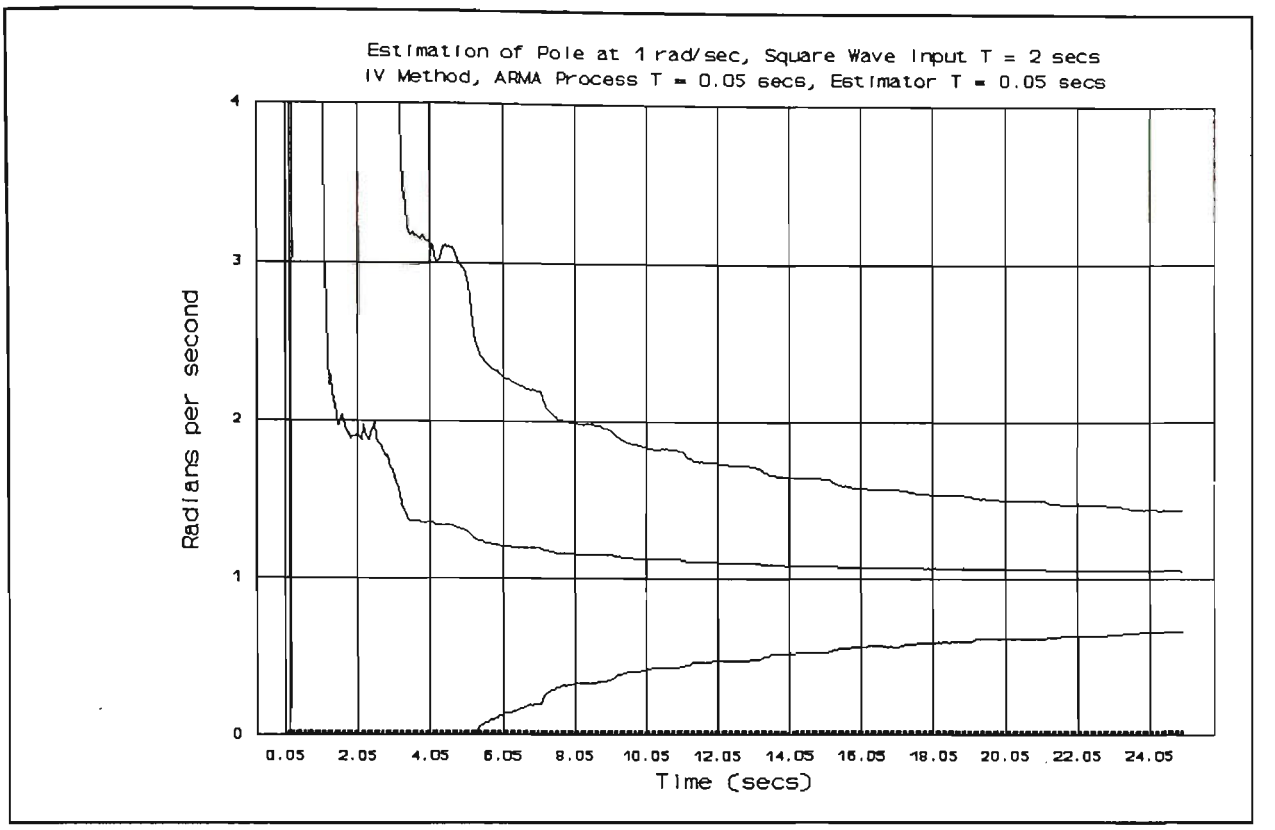

FIGURE 7-6 ESTIMATION OF POLE LOCATION USING 3 PARAMETER LU FACTORISATION METHOD WITH INSTRUMENTAL VARIABLES

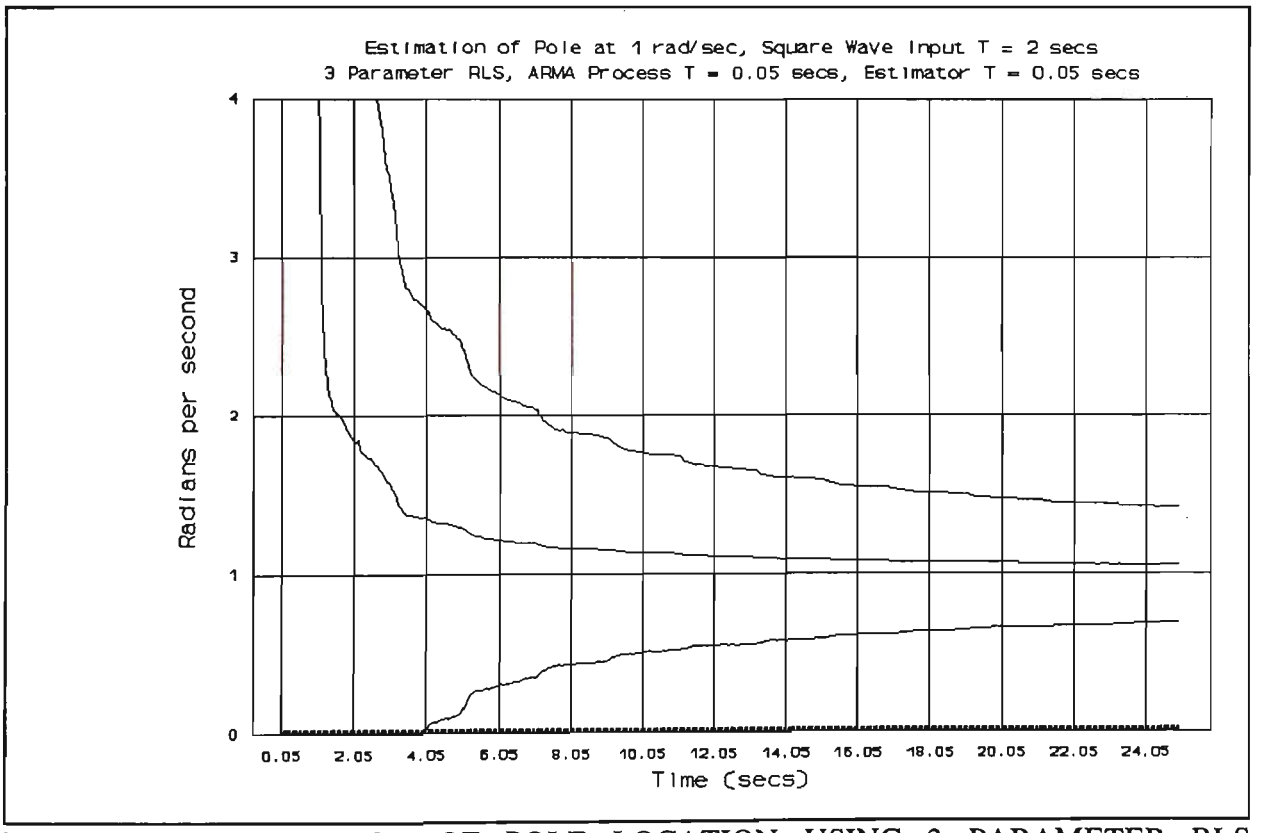

FIGURE 7-7 ESTIMATION OF POLE LOCATION USING 3 PARAMETER RLS METHOD

In Table 7-4 a comparison is made of the performance of the various estimators tested using the combination of input signal, sampling rate and process that resulted in the instability of the 4 Parameter LU Factorisation Method.

i.e. a square wave input signal of 2 seconds periodic time and amplitudes of +5 and +0.01. The ARMA process and estimator both using a clock of 0.05 seconds. An RMS noise value of 0.01, giving an SNR of 64.9 dB.

TABLE 7-4 COMPARISON OF FINAL VALUES AND CONVERGENCE TIMES OF SEVERAL METHODS UNDER THE SAME TEST CONDITIONS

FIGURE/METHOD	'FINAL' VALUE	CONVERGENCE TIME
5-24/4LU	Unstable	Unstable
5-25/4RLS	1.07	10.0
7-6/3IV(LU)	1.05	7.20
7-7/3RLS	1.05	6.05

RELATIVE MERITS OF THE THREE PARAMETER RLS AND THREE PARAMETER INSTRUMENTAL VARIABLE METHODS

In all cases the 3 Parameter RLS Method outperformed all varieties of method that used instrumental variables. The 3 Parameter Instrumental Variable Estimator, using LU Factorisation had a required computation time of 73 microseconds, compared with the 61 microseconds required by the 3 Parameter RLS Method.

Further, the observation of the phenomenon described as 'bursting' suggests that considerable care should be taken when implementing instrumental variable methods in which the instrumental variables are derived from an adaptive filter.

CHAPTER 8 THE APPLICATION OF THE THREE PARAMETER ESTIMATORS TO THE REAL POSITIONAL SERVOSYSTEM

8-1 DESCRIPTION OF THE POSITIONAL SERVOSYSTEM

The positional servosystem used consisted of the following components:

i) a 40W, permanent-magnet DC motor.

ii) a two-channel, incremental optical encoder, permanently coupled to the armature shaft of the motor.

iii) a 71:1 planetary gearhead, permanently coupled to the armature shaft of the motor.

iv) a detachable turntable mounted on the output shaft of the gearbox.

v) a pulse width modulation unit, used to control the armature current of the motor.

vi) interface circuitry to enable the above components to be controlled and monitored by a Transputer System.

The complete servosystem is described in Appendix B, with a comprehensive description of each of the above components. Figure B-1 of Appendix B is a block diagram, indicating the interconnection of the above components. The Transputer System is described in Appendix A.

8-2 THE DETERMINATION OF THE LOCATION OF THE UNKNOWN SYSTEM POLE

INTRODUCTION

The first order model of the motor, described by Equation 3-38, was used as the basis for the following work. In particular it was decided to estimate the location of the unknown pole, p.

In Section 3-6, based upon theoretical considerations and data sheet values, the unknown pole was predicted to be at the location of 29.6 radians per second. It should be noted that the data-sheet values of rotor inertia, J, and the coefficient of viscous friction, B, were for the motor alone. In all of the practical work the motor was permanently coupled to both the gearbox and the optical encoder.

The simulations of Chapters 5, 6 and 7, indicated the superiority of the 3 Parameter methods over the others considered. Accordingly these methods were used on the real servosystem. Experiments were performed both with and without the turntable, which having a mass of 1.3 kilograms and a diameter of 0.45 metres, provided a convenient means of changing the inertial load on the system.

THE STEP RESPONSE OF THE SERVOSYSTEM

Prior to performing the parameter estimation experiments, two Speed Step Response Tests were performed on the servosystem. The first test was conducted without the turntable, the second with the turntable fitted. The results of these tests are presented in Figures 8-1 and 8-2.

The responses were obtained by applying a 'zero' to 'positive maximum' output voltage step from the DAC circuit. The change in position of the armature shaft was measured at a sampling rate of 200 samples per second (i.e. T = 5 milliseconds). The ordinate axes of both Figures 8-1 and 8-2 are labelled in terms of radians of the output shaft, rather than of the armature shaft itself.[1]

[1]Comments are made in Section B-6 of Appendix B concerning the measurement of shaft angle as performed in these Step Response Tests. These comments concern some practical restrictions resulting from the hardware design.

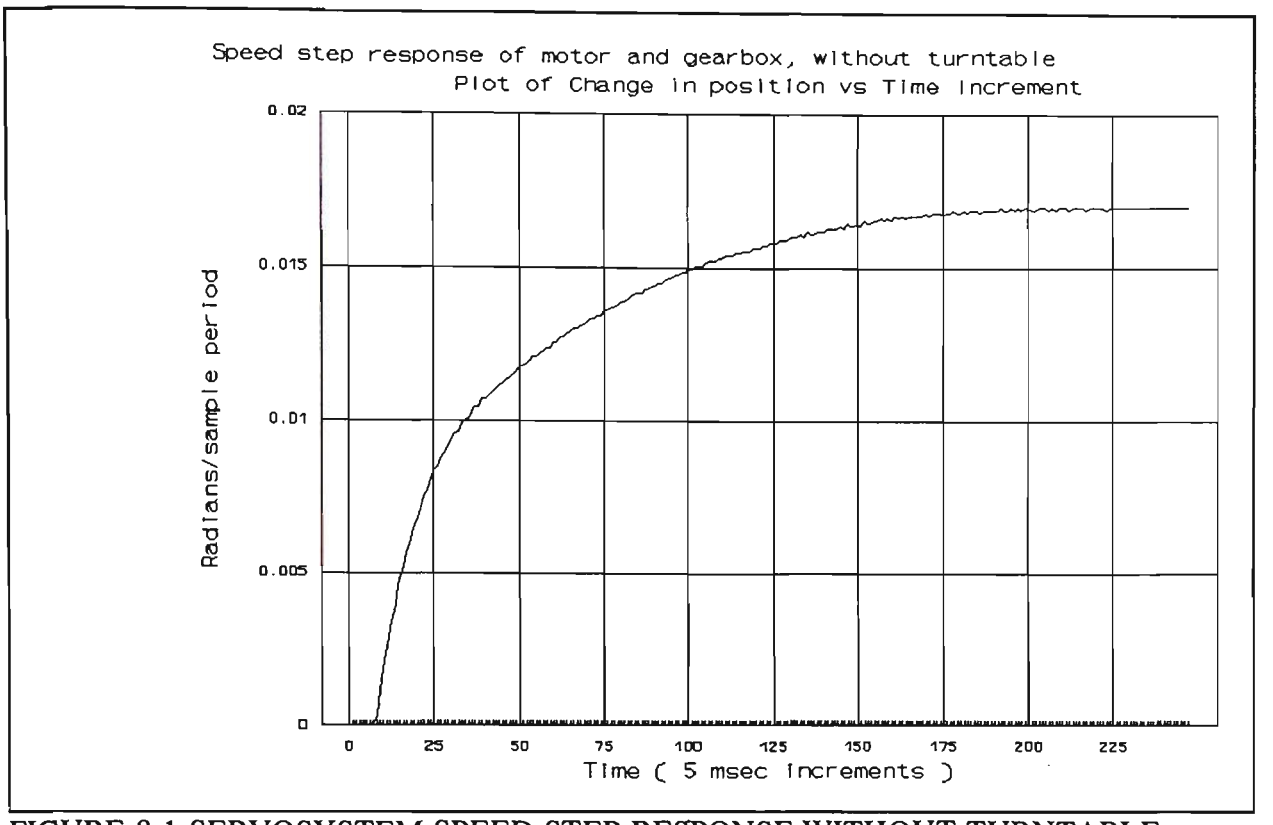

FIGURE 8-1 SERVOSYSTEM SPEED STEP-RESPONSE WITHOUT TURNTABLE

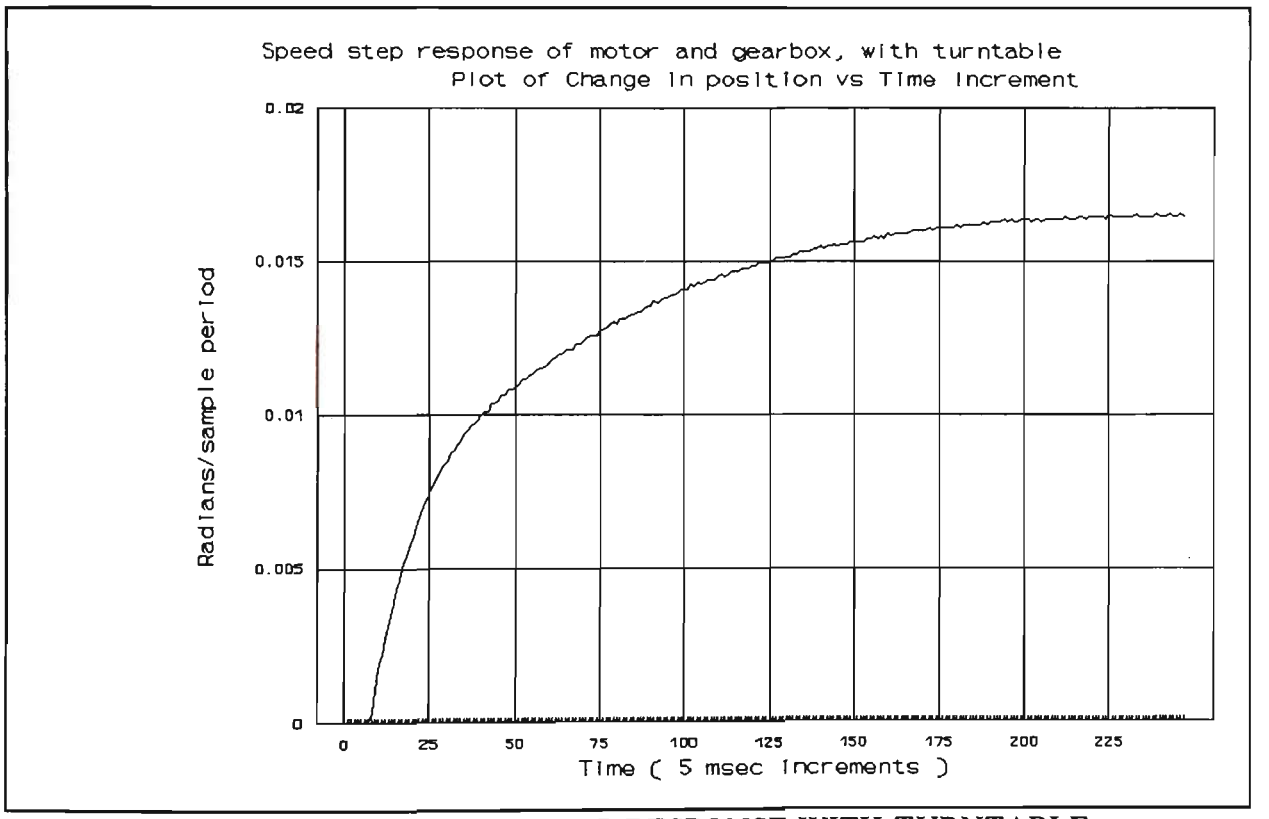

FIGURE 8-2 SERVOSYSTEM SPEED STEP RESPONSE WITH TURNTABLE

Measurements taken from the results presented as Figures 8-1 and 8-2 indicate the following system time constants:

Without turntable, τ = 165 milliseconds

With turntable, τ = 185 milliseconds

Accepting the validity of the first order model of Equation 3-38, then the above figures indicate poles located at 6.1 and 5.4 radians per second respectively.

8-3 USE OF THE THREE PARAMETER RLS ESTIMATOR WITHOUT MOTOR STOPPING OR REVERSAL

THE NEED TO AVOID MOTOR STOPPING OR REVERSAL

As noted in Chapter 3, there are many deficiencies in the simple linear model adopted to describe the servosystem. The most significant failings of the model are due to:

i) neglect of backlash and deadzone behaviours of the system

ii) the assumption of a Constant of Viscous Friction, B.

To minimise the effects of backlash and deadzone the first experiments were designed to ensure that the motor rotated continuously, in one direction, throughout the duration of the test run.

THE TEST INPUT SIGNAL

The experiments consisted of twenty consecutive runs, each of four seconds duration. Each run of four seconds consisted of two complete cycles of a square wave. The run starting with the DAC switching to its 'maximum positive' output voltage. In the second half-cycle of this input signal the DAC delivered a small, positive voltage, sufficient to ensure that the motor continued to rotate in the same direction.

At the start of each of the twenty runs both the position counter and the variables of the parameter estimator were initialised. The estimates of the coefficients of the ARMA Process being reset to zero. The Error Covariance Matrix being reset as an Identity Matrix scaled by a factor of 10000.

Within each run the pole location was calculated at each sampling instant of the estimator.

RESULTS FROM THE THREE PARAMETER RLS ESTIMATOR WITHOUT MOTOR STOPPING OR REVERSAL

Figures 8-3 to 8-8 show the results of using the 3 Parameter RLS Estimator with the above test input signal. Each figure shows the results from twenty consecutive runs of the estimator. The centre trace in each figure shows the mean estimate of the pole location. This trace is enclosed by traces showing the +/- two standard deviations spread in estimates, from those twenty runs. It should be stressed that this spread is calculated directly from the population of twenty estimates, and is not derived from the contents of the Error Covariance Matrix.

The sample period of the estimator was reduced from 100 msecs. to 2 msecs. as detailed below in Table 8-1. The 'Final Mean Estimate' is the mean estimate of the pole location at the end of each run, i.e. the mean of the twenty estimates of pole location at t = 4.0 secs.. The 'Mean at t=(1.0-T)' is included to indicate the response of the estimator to the single leading edge of the input signal. This mean is most useful in revealing the convergence rate achieved by each estimator.

TABLE 8-1 RESULTS FOR 3 PARAMETER RLS ESTIMATOR, WITHOUT MOTOR STOPPING OR REVERSAL, WITHOUT THE TURNTABLE

ESTIMATOR PERIOD (msecs)	FINAL MEAN ESTIMATE (Rad/sec)	MEAN @ t=(1.0-T) (Rad/sec)	FIGURE
100	6.74	5.12	8-3
50	6.97	6.10	8-4
20	7.33	7.05	8-5
10	7.80	8.61	8-6
5	9.29	14.29	8-7
2	20.01	52.48	8-8

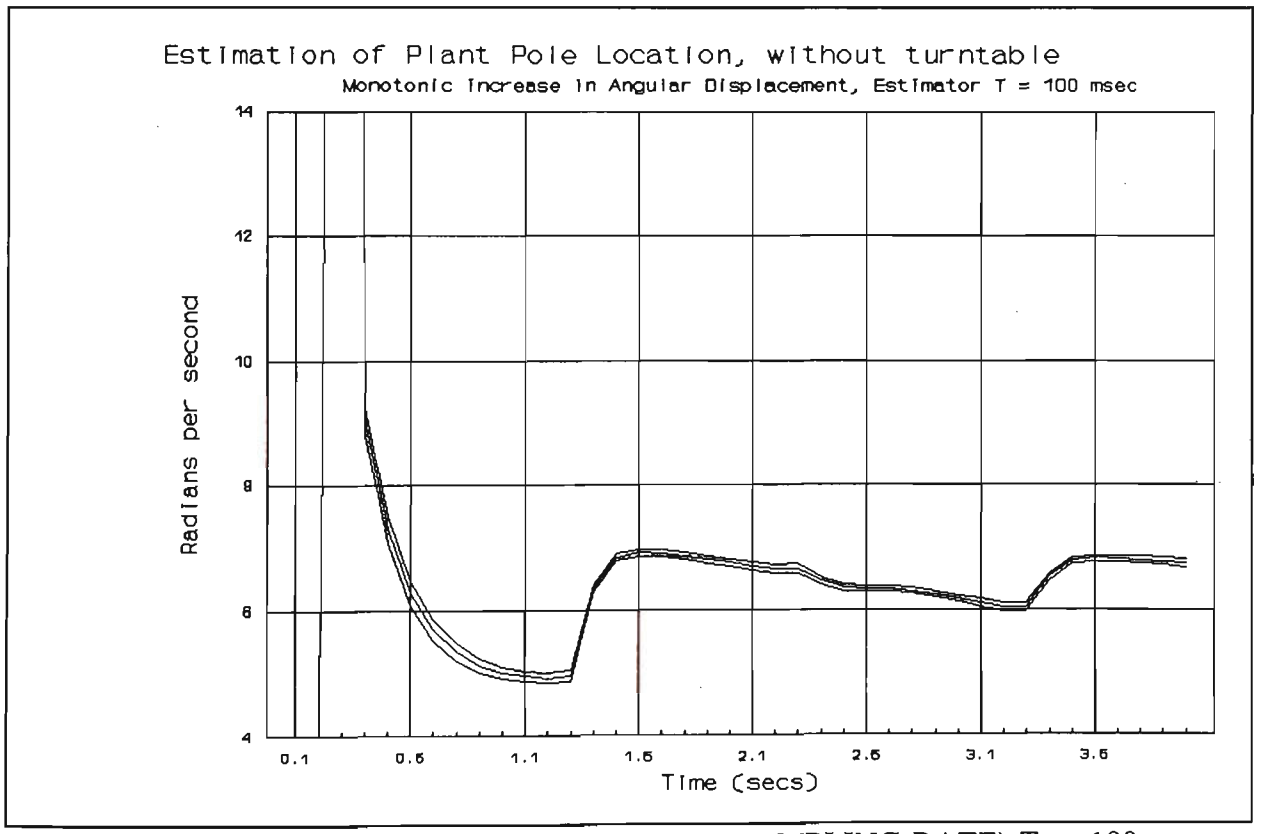

FIGURE 8-3 THREE PARAMETER RLS (LOWEST SAMPLING RATE) T = 100ms

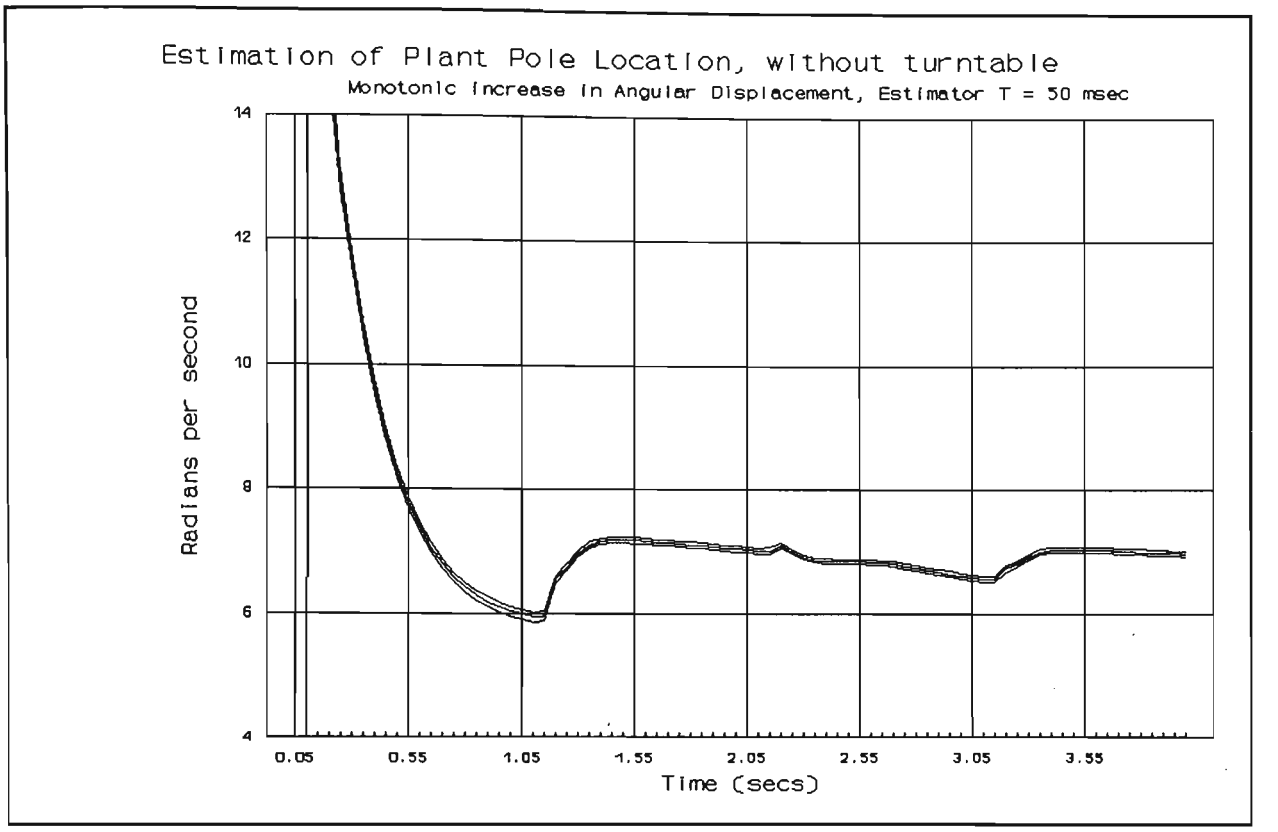

FIGURE 8-4 THREE PARAMETER RLS T = 50 ms

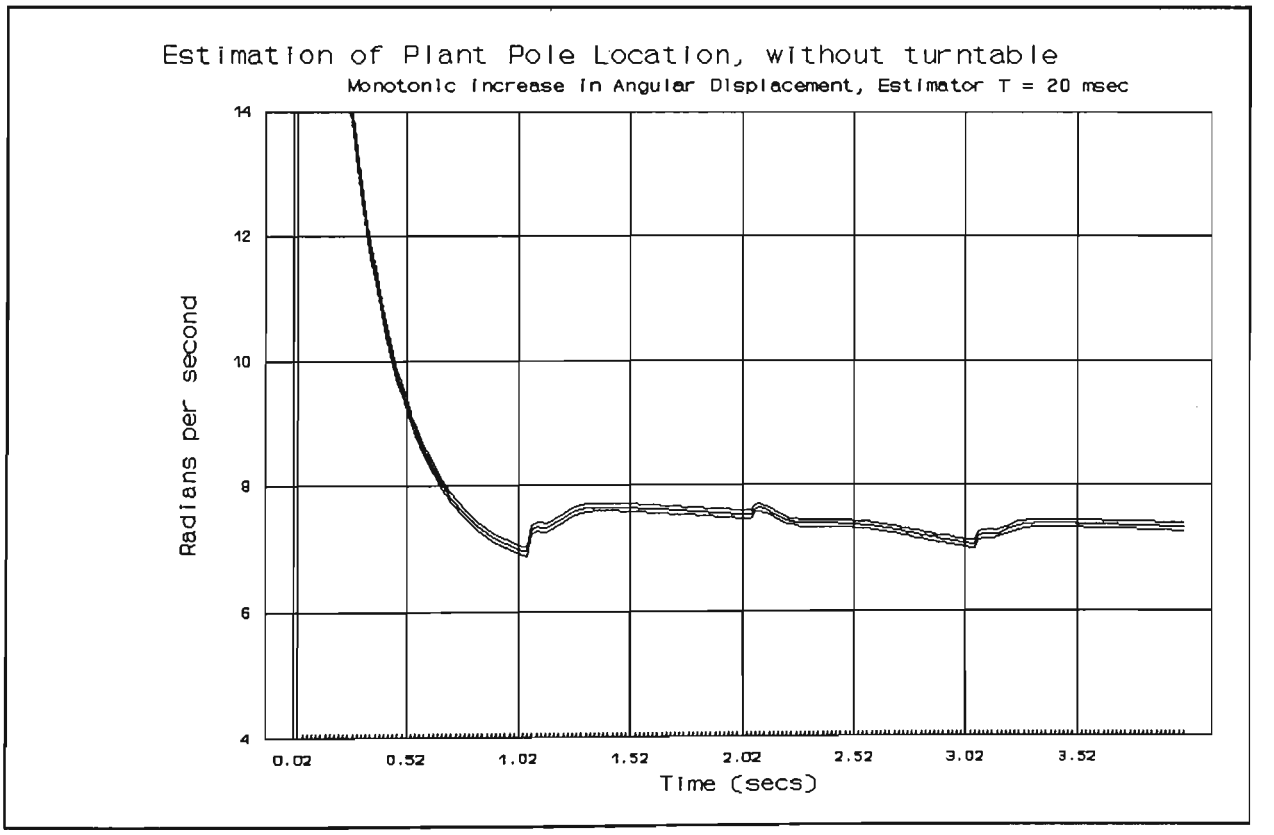

FIGURE 8-5 THREE PARAMETER RLS T = 20 ms

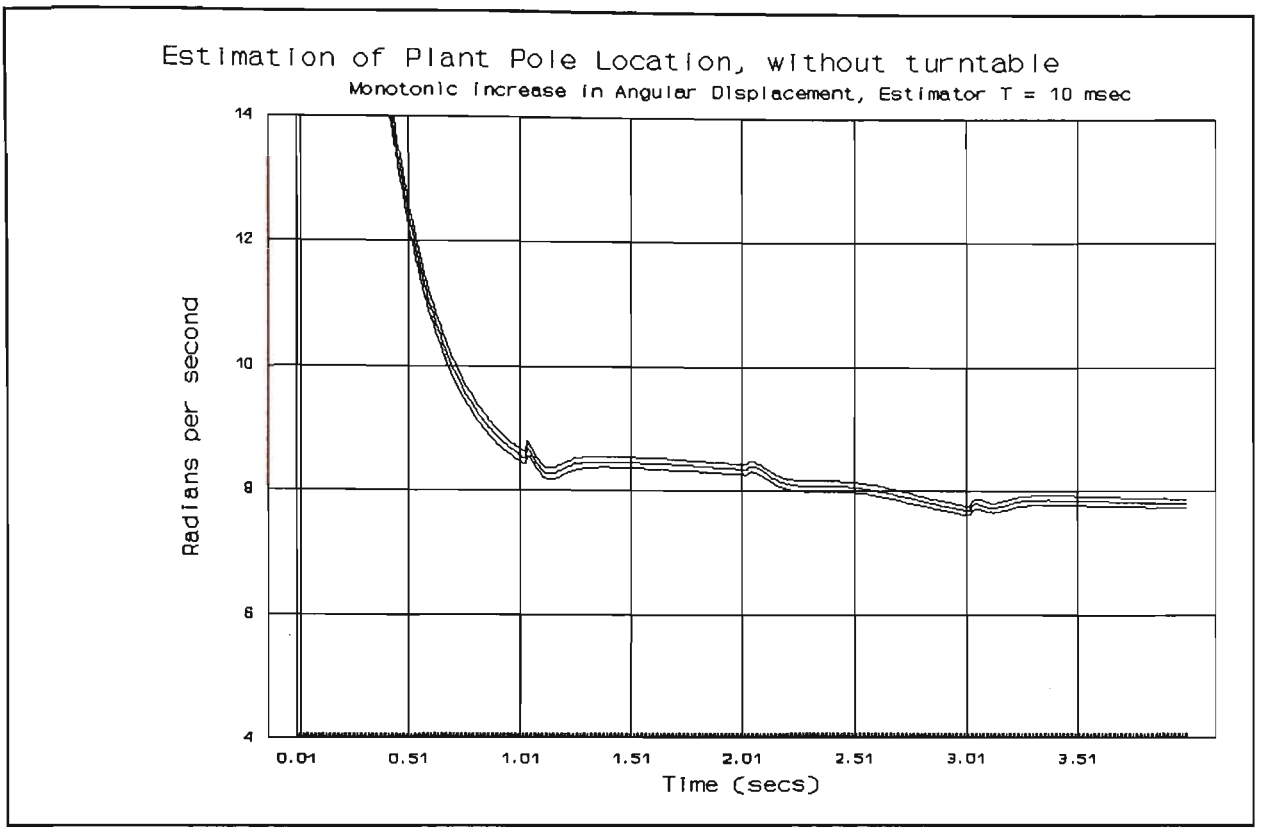

FIGURE 8-6 THREE PARAMETER RLS T = 10 ms

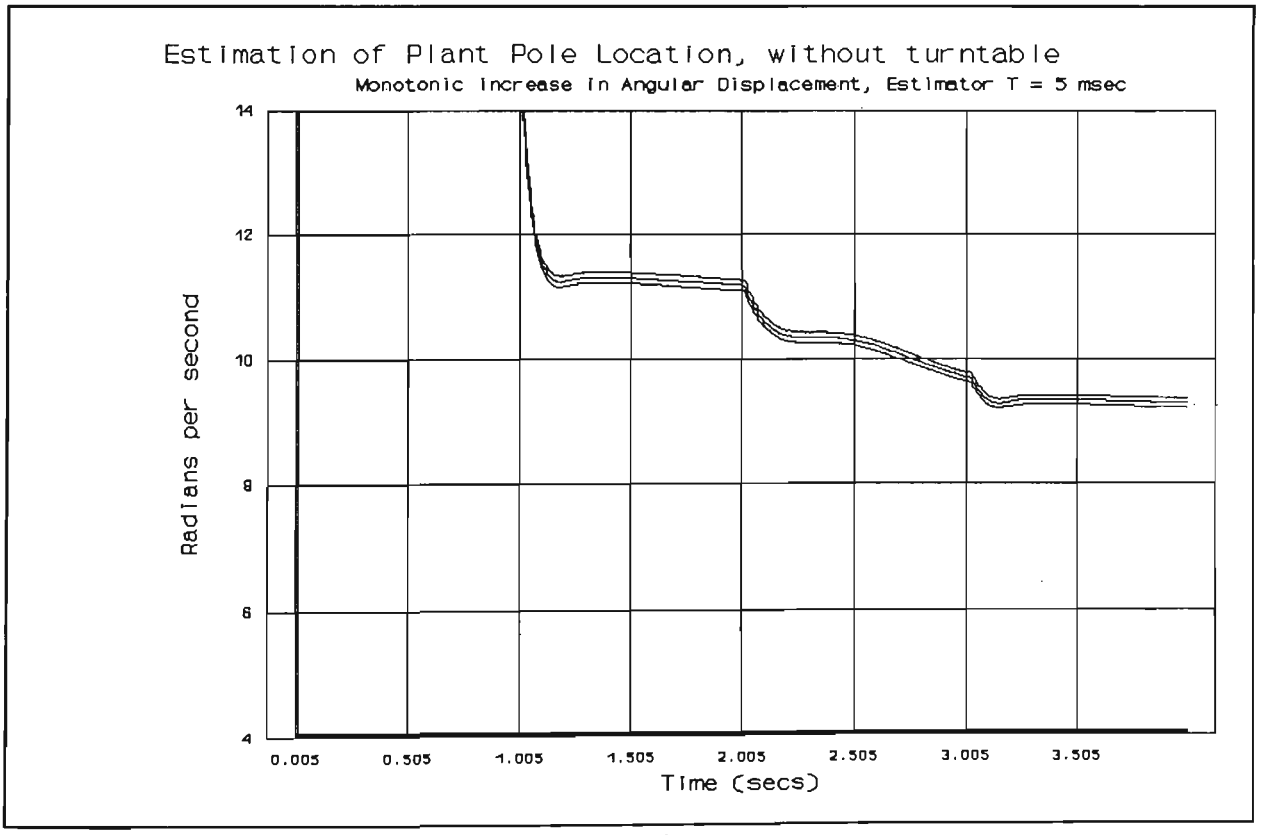

FIGURE 8-7 THREE PARAMETER RLS T = 5 ms

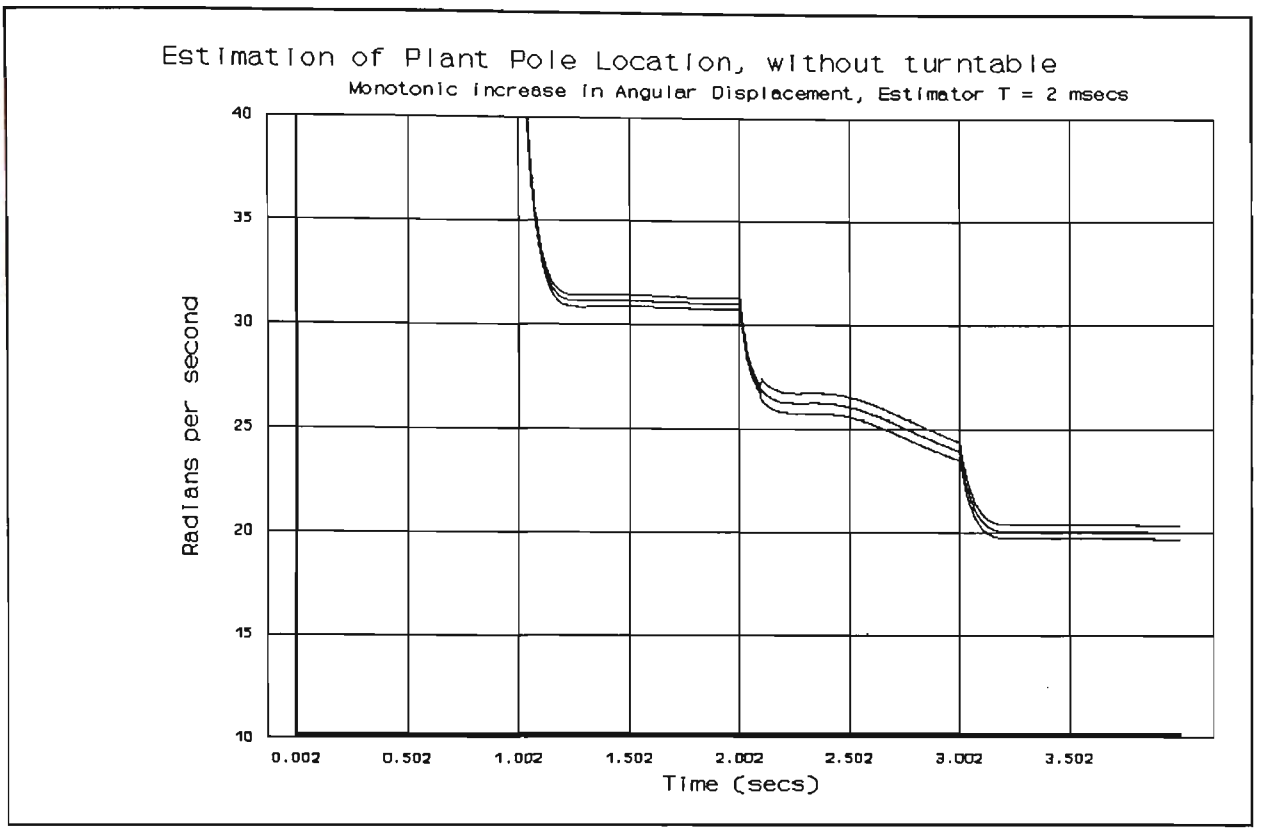

FIGURE 8-8 THREE PARAMETER RLS (HIGHEST SAMPLING RATE) T = 2 ms

INTERPRETATION OF THE RESULTS USING THE RLS ESTIMATOR WITHOUT MOTOR STOPPING OR REVERSAL

i) As the sampling rate increases the 'Final Estimate' (i.e. the estimate at t = 4.0 seconds) also increases. This is clearly shown in Table 8-1.

The Speed Step Response of Figure 8-1 suggested a pole location of 6.1 radians per second. Accepting this value as correct, an increase in sampling rate resulted in a deterioration in the value of the 'Final Estimate'. Further this deterioration became significant at the higher sampling rates.

ii) From the start of each run, the pole estimates decrease monotonically until the effects of the second transition of the input signal at t = 1.0 seconds influence the estimator. In Table 8-1 the column headed 'Mean @ t=(1.0-T)' enables the effect of

the sampling rate upon this transient to be considered. From this column, and the corresponding traces of Figures 8-7 and 8-8 it will be noted that the high sampling rate estimators fail to achieve a rapid convergence towards the expected pole location (6.1 radians/s) during this first second of operation. It is this reduced convergence rate that results in the less accurate estimates at the end of the four seconds of each run. The low sampling rate estimators make better use of the first transition of the input signal at $t = 0.0$ seconds.

iii) The rate of fall of the estimates towards the expected value of 6.1 radians per second is greatest during the periods of high motor velocity (approximately $0 < t < 1$ and $2 < t < 3$ seconds). The estimator is unable to make significant improvements to the estimate during the periods of low motor velocity.

iv) The transitions of the input signal result in discontinuities in the estimate vs. time traces. The amplitudes of these discontinuities will be dependent upon both the magnitude of the prediction error and the adaptive gain remaining with the RLS estimator. The effect of sampling rate upon the rate of fall of adaptive gain in the RLS algorithm is considered in Section 8-5, and Figures 8-21 and 8-22 in particular.

v) The envelope defined by the +/- two standard deviations traces is narrow compared with the amplitudes of the fluctuations in the values of the estimates. Assuming a Normal Distribution of estimates, a spread of +/- two standard deviations should include 95% of those estimates. The estimator can thus be seen to produce consistent but incorrect estimates.

vi) There is a tendency for the width of the +/- two standard deviations spread to increase as the sampling rate is increased. This may be attributed to an increase in sampling rate causing a reduction in the signal-to-noise ratio of the measured shaft position. These measurements have an error independent of the sampling rate, this error being fixed by the resolution of the optical encoder and its associated counter

circuit. The magnitude of the change in shaft-angle between consecutive sampling instants should obviously fall with increased sampling rate.

During some preliminary experiments there was a problem due to interference from the VMOS Gate Drive Circuits coupling into the input of the PWM Unit. This interference resulted in a significant increase in the +/- two standard deviations spread. All of the results presented here are from experiments where this interference was minimised by modification to the layout of the various connecting cables and circuit boards.

CONCLUSION DRAWN FROM THE ABOVE RESULTS

For these experiments increasing the estimator sampling rate did not improve the performance of the estimator. Quite the opposite, both the accuracy and convergence rate of the estimates deteriorated with increased estimator sampling rate. This conclusion being appropriate to the input signal and estimator used in obtaining these results.

From iii) above it is concluded that the RLS estimator fails to achieve adequate convergence when presented with "small" signals.

8-4 THE EFFECT OF INCREASED INERTIAL LOAD

The experiments described in Section 8-3 were repeated with an increased inertial load. This increase was achieved by connecting the turntable to the output shaft of the gearbox. The Speed Step Response measurements, as presented in Figures 8-1 and 8-2, indicate that the finite system pole should shift from 6.1 radians per second (without turntable), to 5.4 radians per second (with turntable). All other aspects of the experiment remained unchanged from those described in Section 8-3.

RESULTS FROM THE THREE PARAMETER RLS ESTIMATOR ON THE SYSTEM WITH TURNTABLE, AND WITHOUT MOTOR STOPPING OR REVERSAL

Figures 8-9 to 8-14 show results (with turntable) equivalent to those of Figures 8-3 to 8-8 (without turntable) respectively. The effect of the turntable upon the 'Final Mean Estimate' (i.e. the estimate at t = 4.0 seconds) is summarised below in the Table 8-2.

TABLE 8-2 COMPARISON OF FINAL ESTIMATES FOR THE SYSTEM WITH AND WITHOUT THE TURNTABLE, USING THE THREE PARAMETER RLS ESTIMATOR

ESTIMATOR PERIOD (msecs)	FINAL MEAN WITHOUT T/T (Rad/sec)	FINAL MEAN WITH T/T (Rad/sec)
100	6.74	6.29
50	6.97	6.37
20	7.33	6.60
10	7.80	7.07
5	9.29	8.64
2	20.01	18.4
Speed Step Response	6.1	5.4

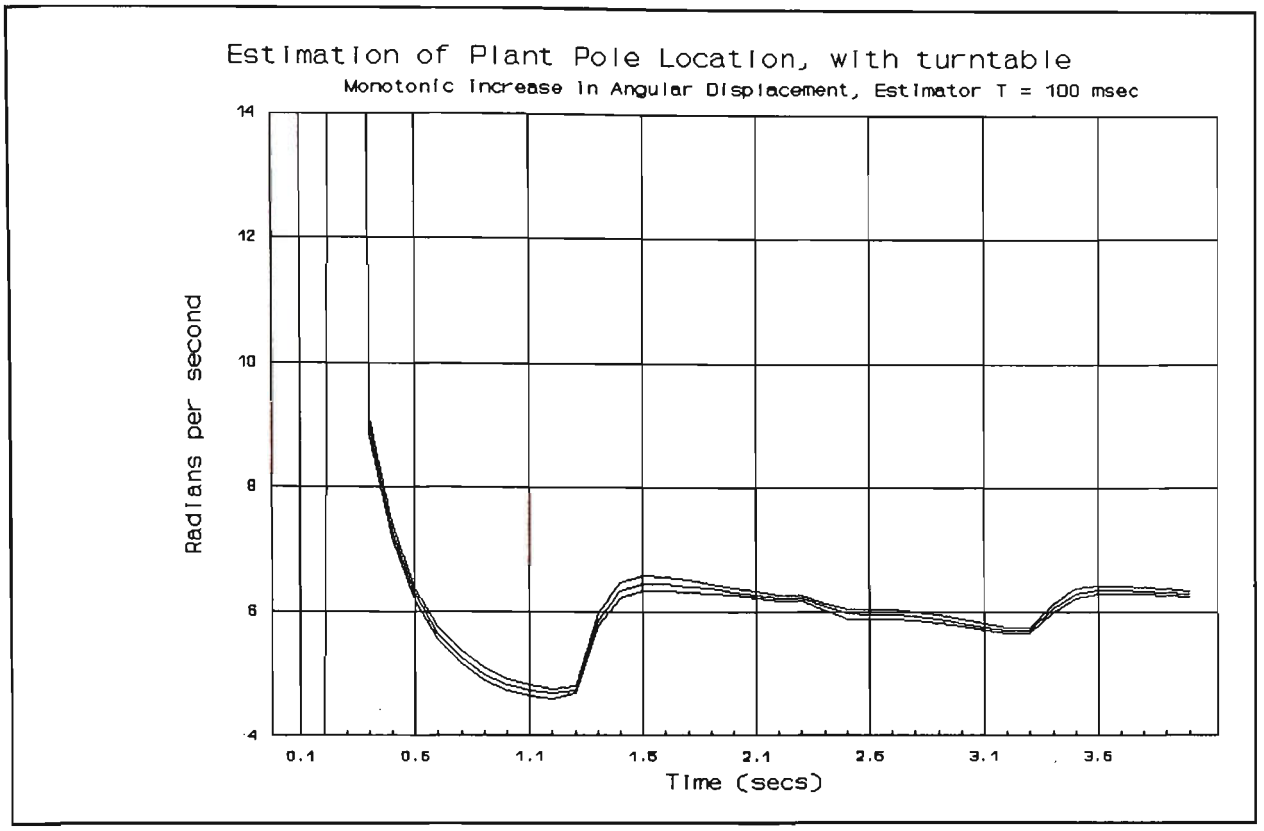

FIGURE 8-9 THREE PARAMETER RLS (LOWEST SAMPLING RATE) T = 100 ms, WITH TURNTABLE

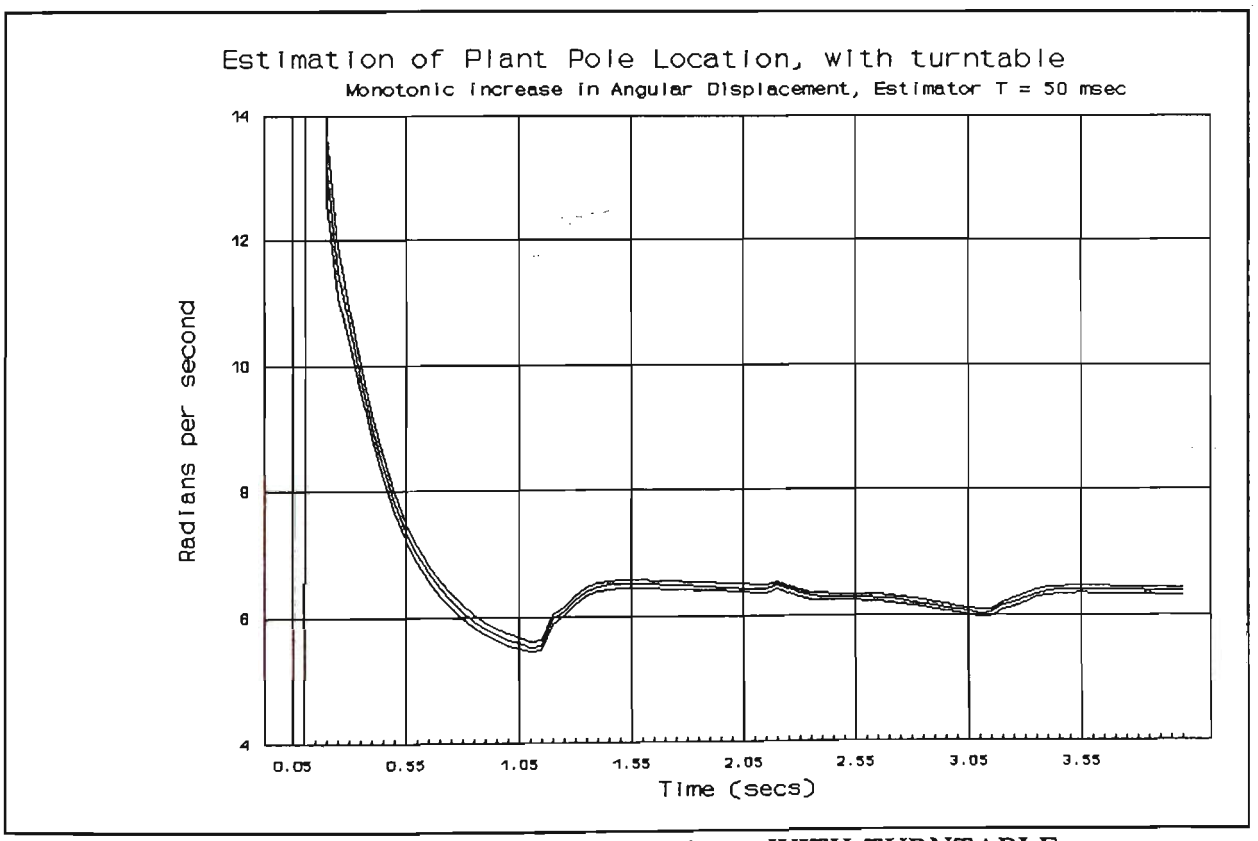

FIGURE 8-10 THREE PARAMETER RLS T = 50 ms, WITH TURNTABLE

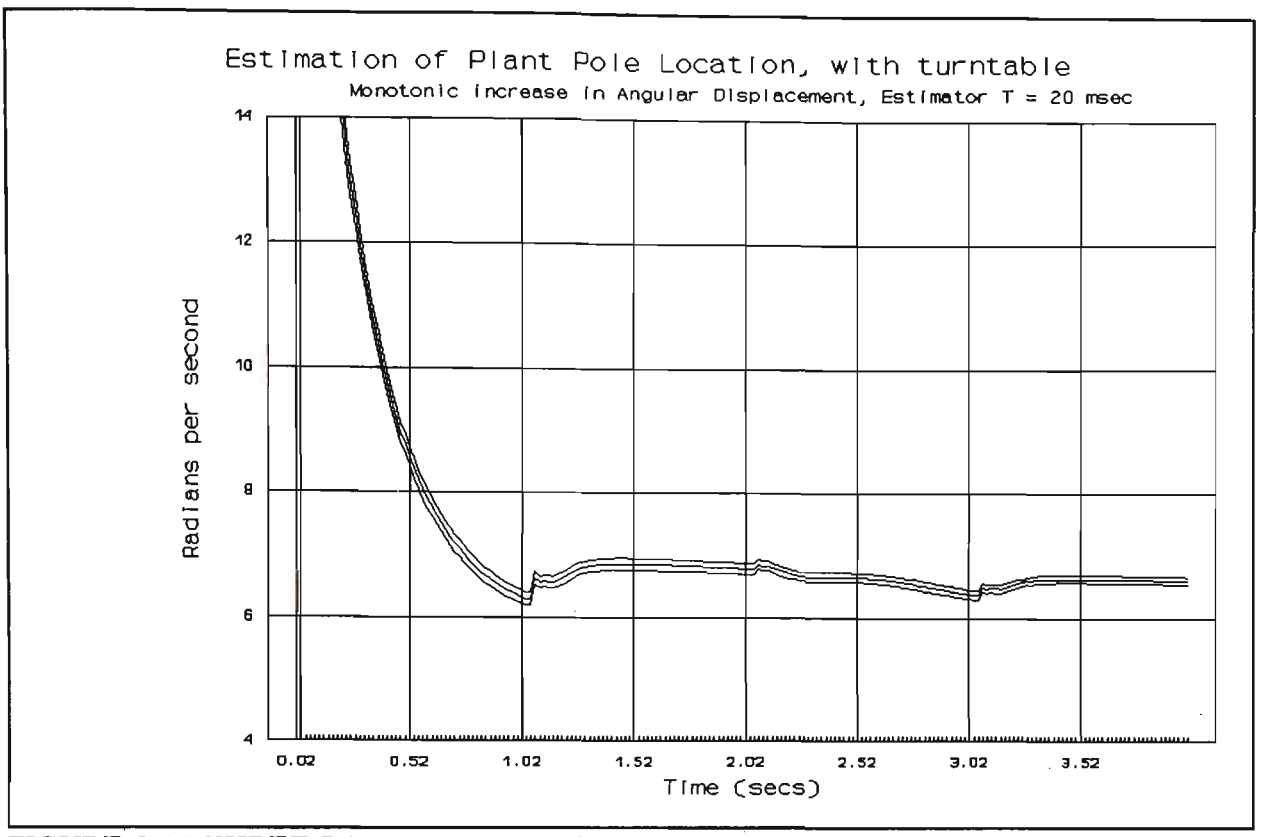

FIGURE 8-11 THREE PARAMETER RLS T = 20 ms, WITH TURNTABLE

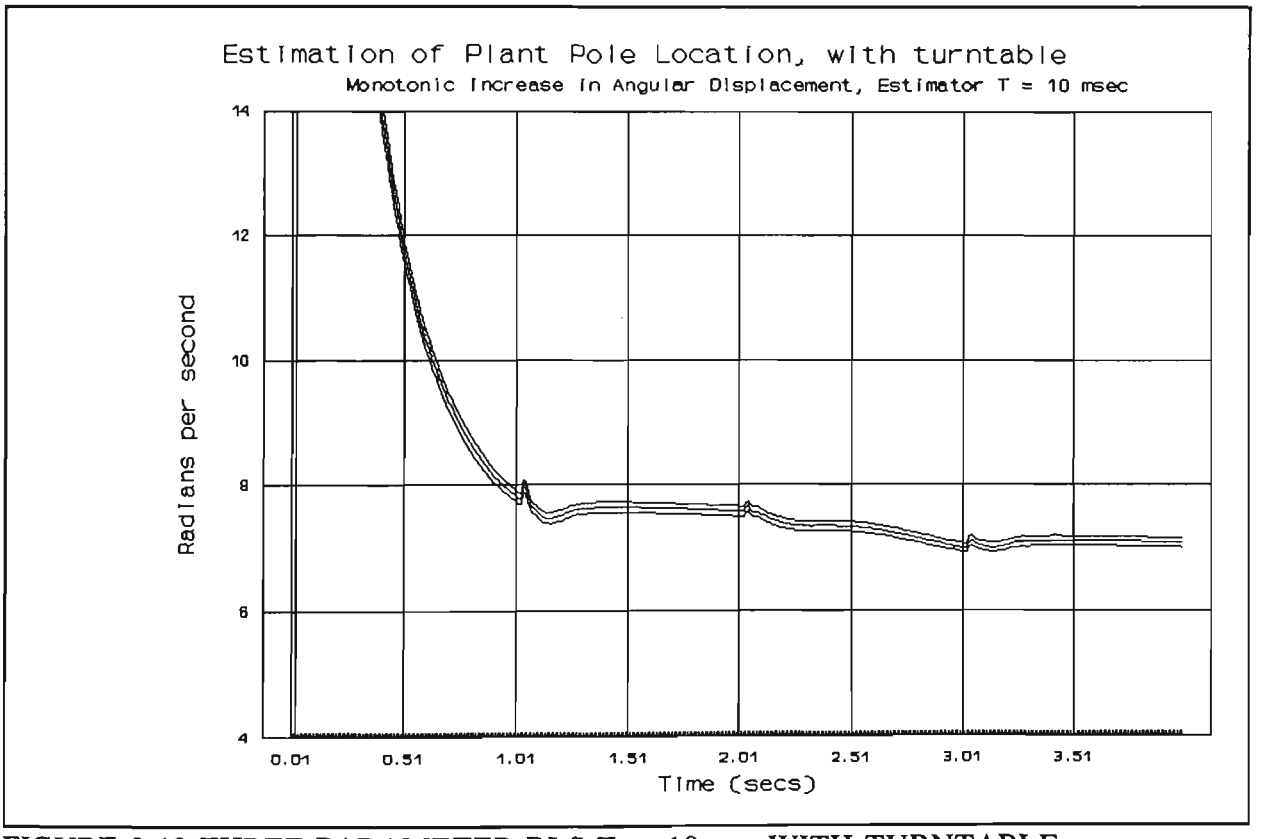

FIGURE 8-12 THREE PARAMETER RLS T = 10 ms, WITH TURNTABLE

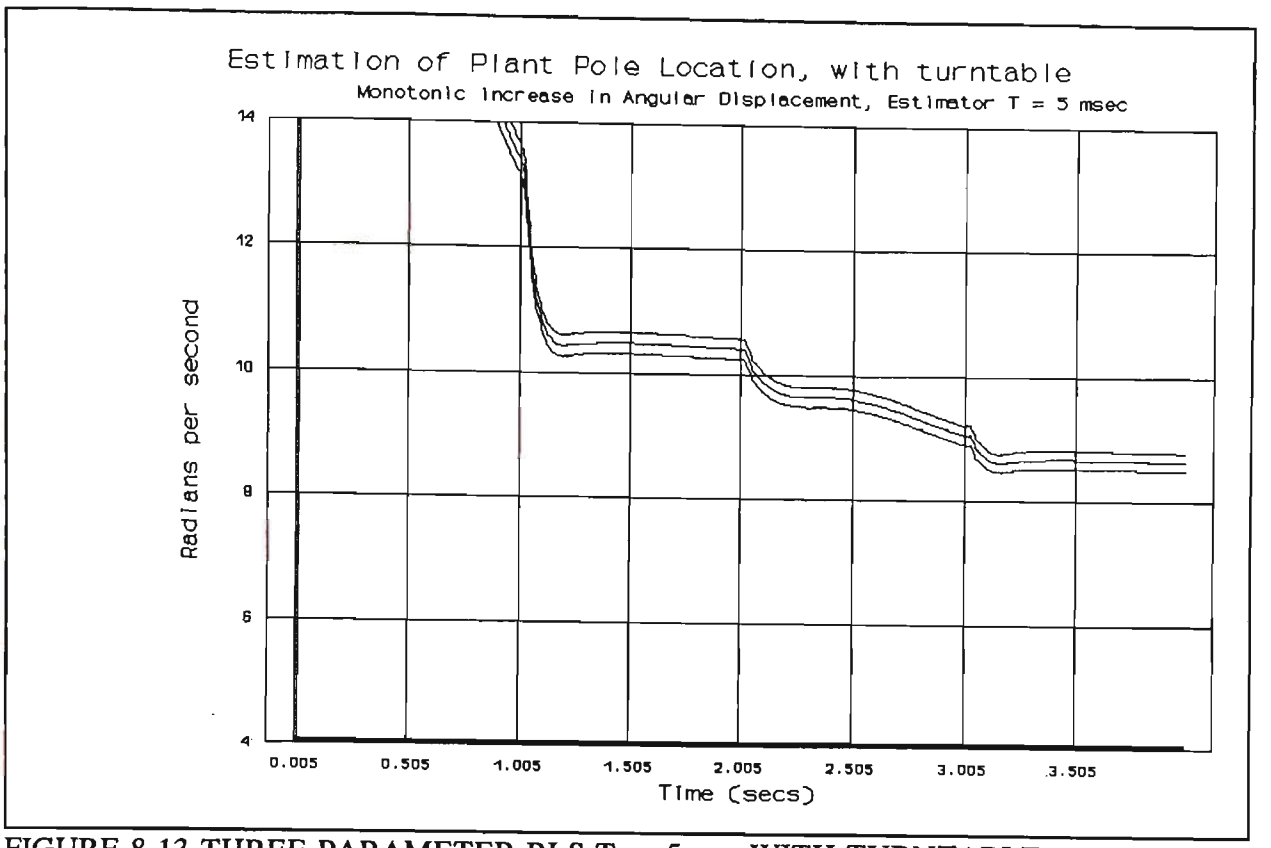

FIGURE 8-13 THREE PARAMETER RLS T = 5 ms, WITH TURNTABLE

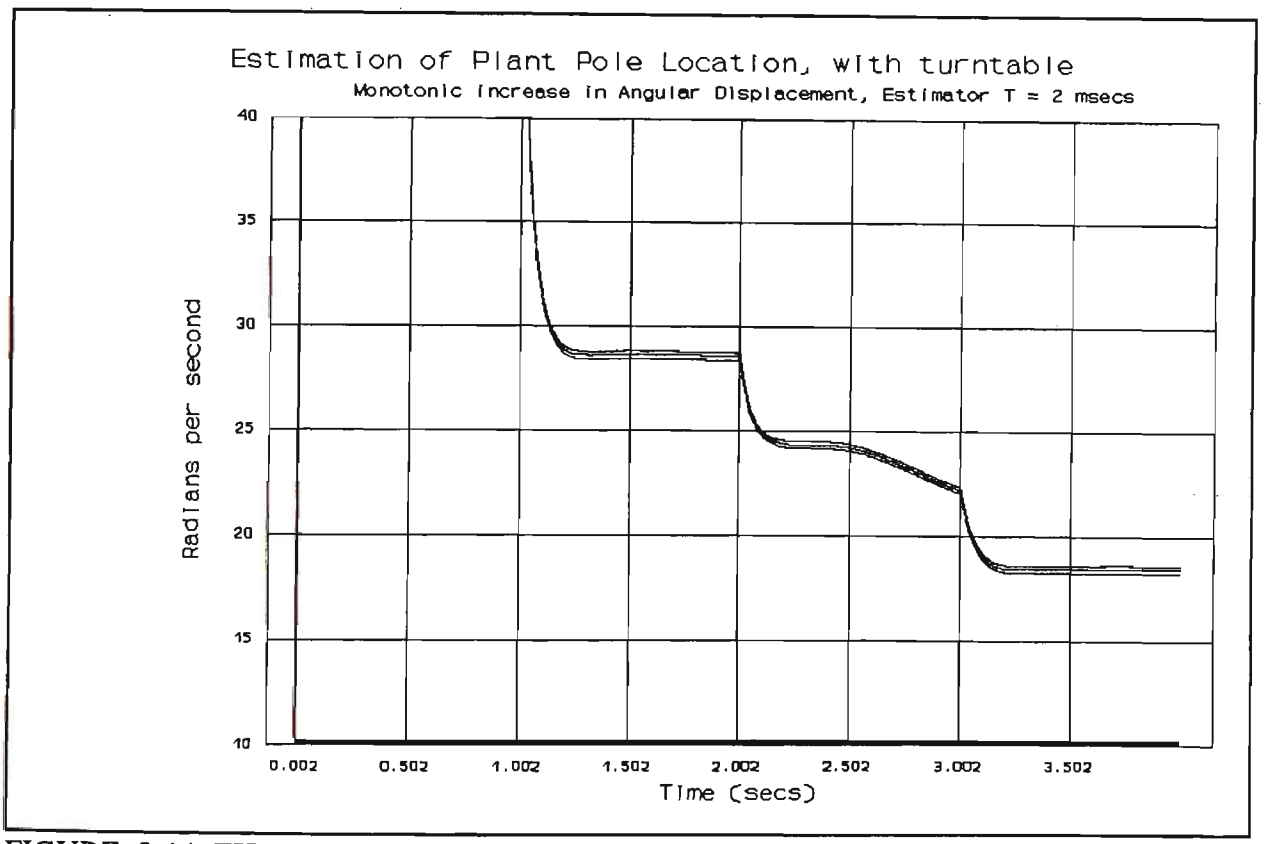

FIGURE 8-14 THREE PARAMETER RLS (HIGHEST SAMPLING RATE) T = 2 ms, WITH TURNTABLE

INTERPRETATION OF THE RESULTS WITH INCREASED INERTIAL LOAD

The results closely resemble those of Section 8-3, the addition of the inertial load having the effect of moving the traces of estimate vs. time downwards. This is consistent with the results obtained from the Speed Step Response measurements presented in Section 8-2.

8-5 THE USE OF THE THREE PARAMETER RLS ESTIMATOR WITH MOTOR REVERSALS

THE TEST INPUT SIGNAL USED TO OBTAIN MOTOR REVERSALS

The test input signal again consisted of twenty consecutive runs each of four seconds duration. This time however the DAC was switched between its 'maximum positive' and its 'maximum negative' output voltage.

RESULTS FROM THE THREE PARAMETER RLS ESTIMATOR WITH MOTOR REVERSALS

Figures 8-15 to 8-20 show results (without turntable) equivalent to those of Figures 8-3 to 8-8 (without turntable) respectively.

TABLE 8-3 RESULTS FOR 3 PARAMETER RLS ESTIMATOR, WITH MOTOR REVERSALS, WITHOUT THE TURNTABLE

ESTIMATOR PERIOD (msecs)	FINAL MEAN ESTIMATE (rad/sec)	MEAN @ t=(1.0-T) (rad/sec)
100	9.49	4.97
50	9.38	5.84
20	9.26	6.84
10	9.35	8.39
5	9.76	13.76
2	14.03	55.79

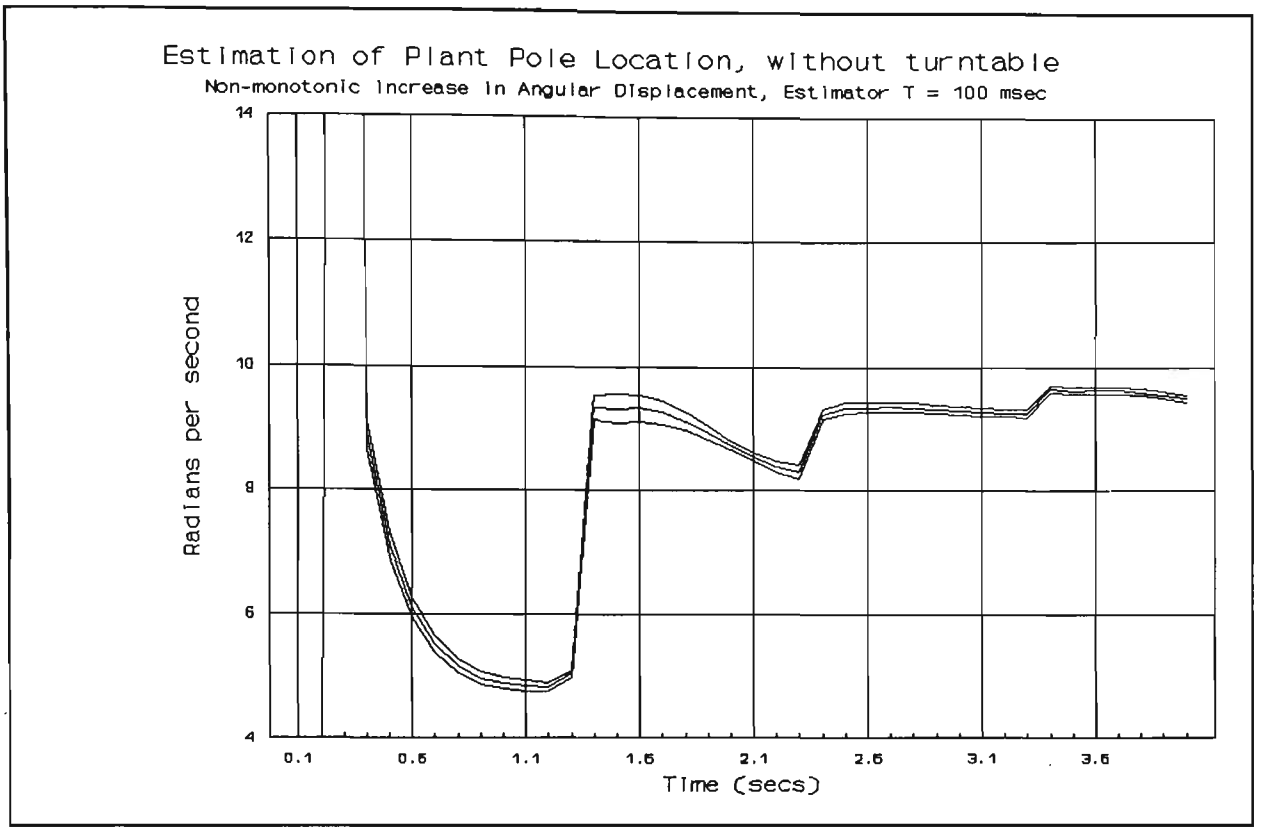

FIGURE 8-15 THREE PARAMETER RLS (LOWEST SAMPLING RATE) T = 100 ms, WITH REVERSALS

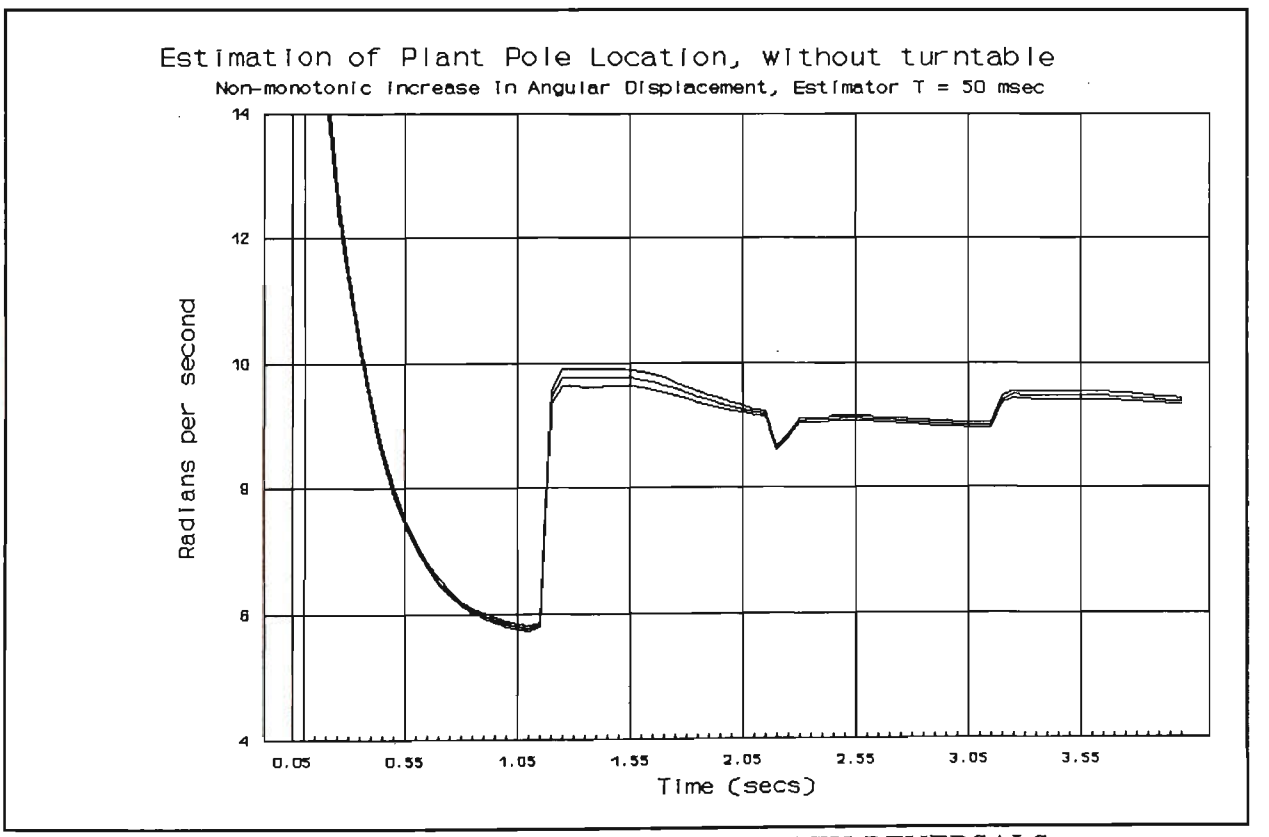

FIGURE 8-16 THREE PARAMETER RLS T = 50 ms, WITH REVERSALS

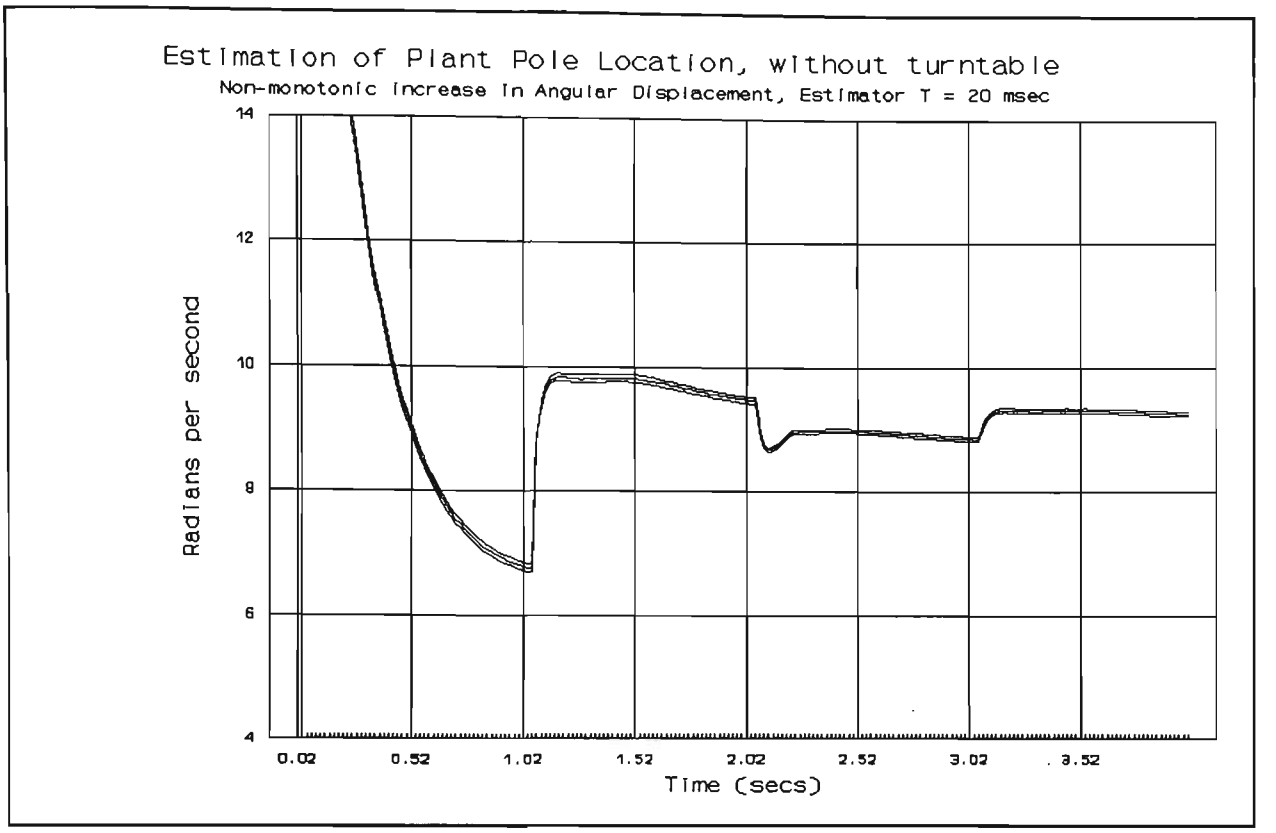

FIGURE 8-17 THREE PARAMETER RLS T = 20 ms, WITH REVERSALS

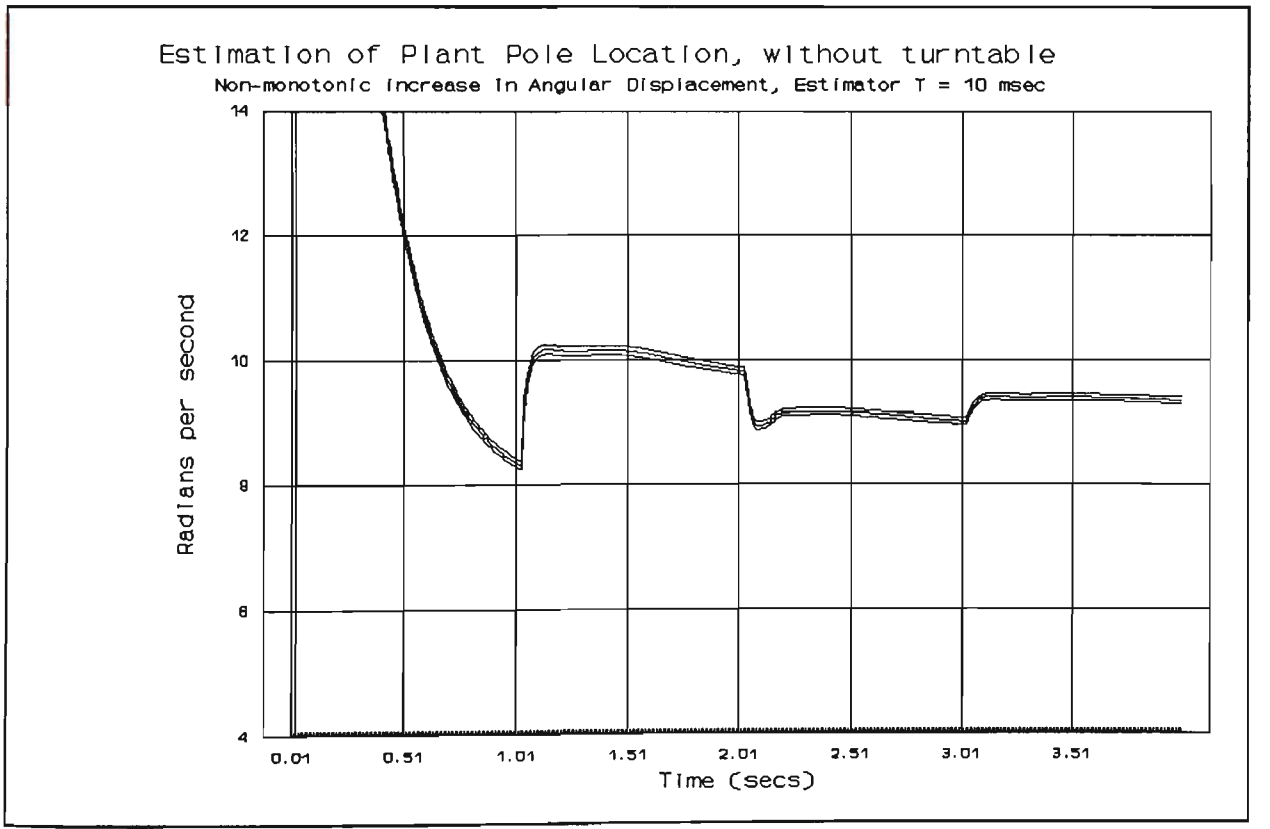

FIGURE 8-18 THREE PARAMETER RLS T = 10 ms, WITH REVERSALS

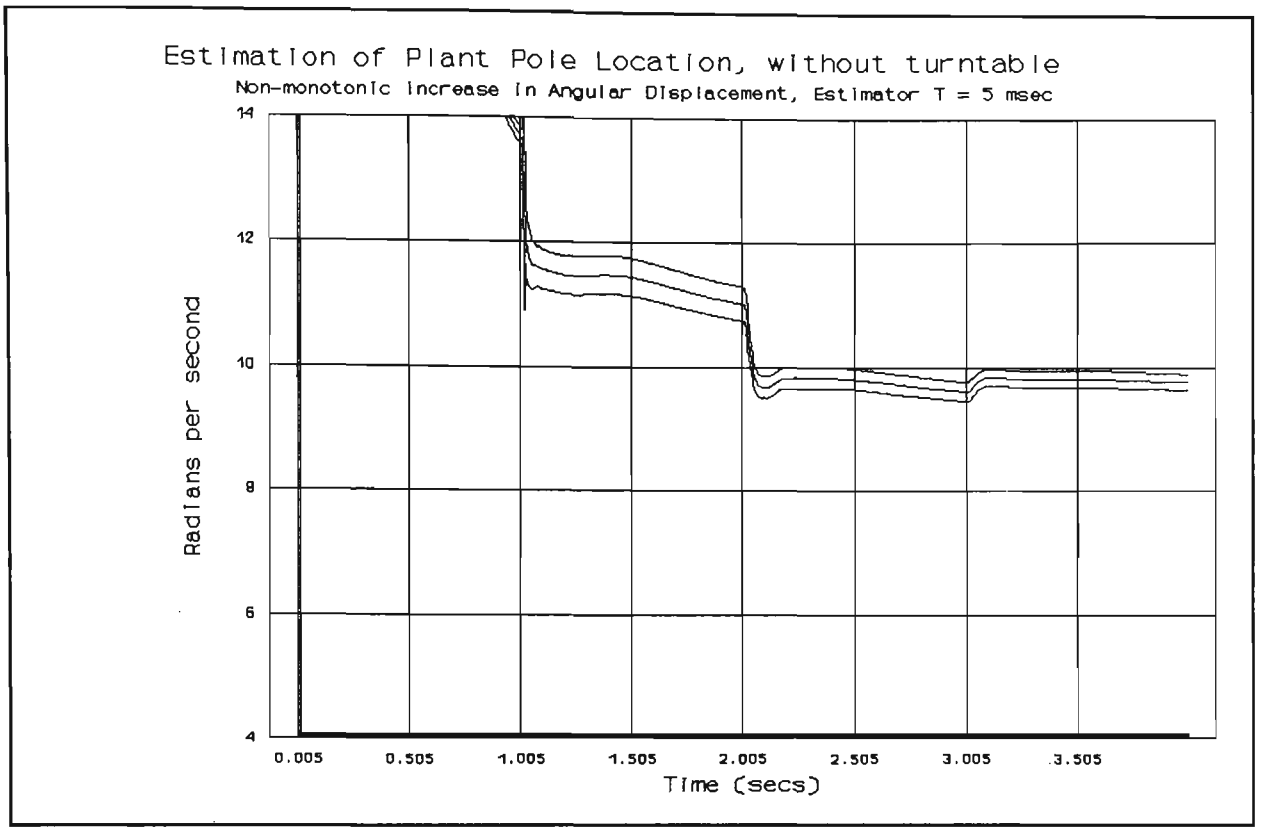

FIGURE 8-19 THREE PARAMETER RLS T = 5 ms, WITH REVERSALS

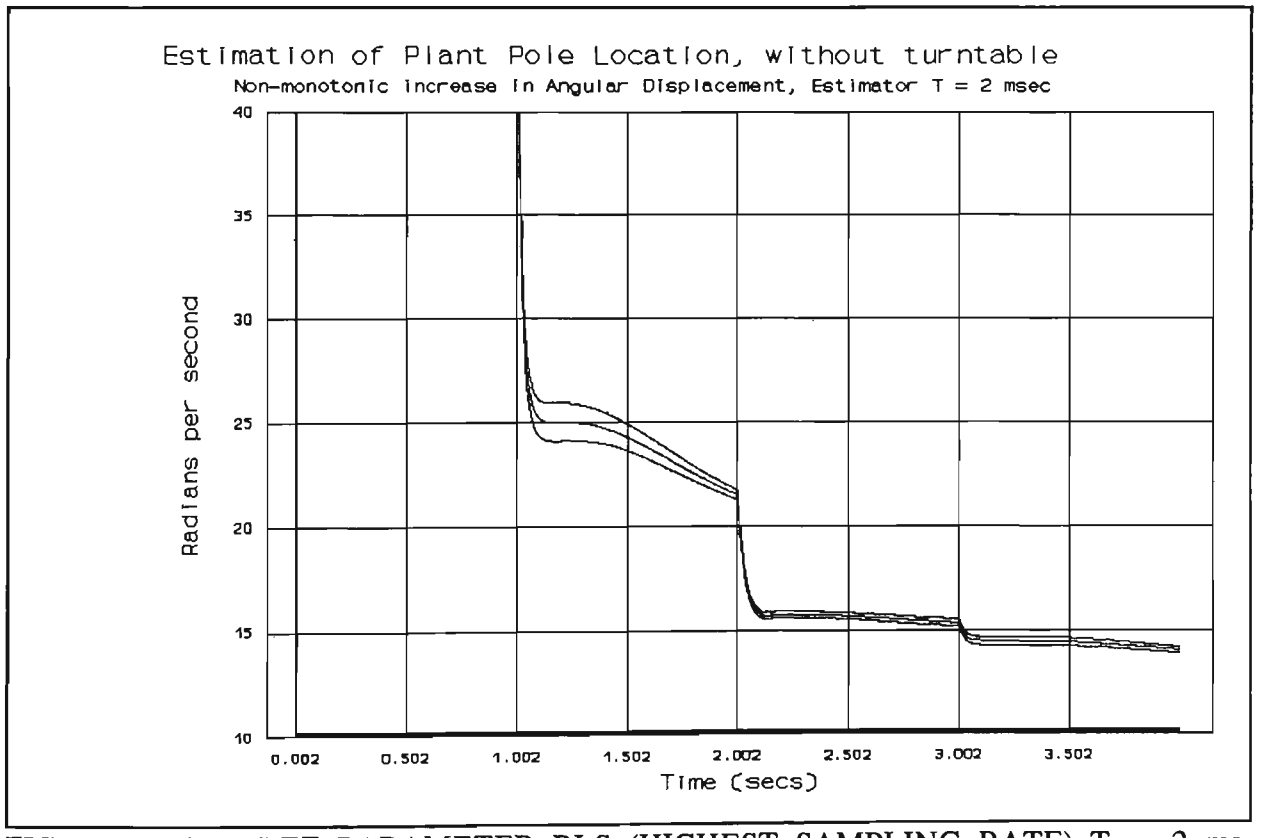

FIGURE 8-20 THREE PARAMETER RLS (HIGHEST SAMPLING RATE) T = 2 ms, WITH REVERSALS

INTERPRETATION OF THE RESULTS USING THE RLS ESTIMATOR WITH MOTOR REVERSALS

i) The linear model of the servomechanism would suggest that the 'Final Mean Estimates' of Table 8-3 (with reversals in motor direction) should be the same as those of Table 8-1 (without reversals), both sets of results being for the system without the turntable.

These tables reveal that the use of an input signal that causes the motor to reverse direction results in the estimate of the pole location rising. This increased value of the estimate is further from that suggested by the Speed Step Response of Section 8-2.

It should be noted that the assumptions on which the linear model of the servomechanism was based are less valid for a motor that either stops or reverses direction. The linear model fails to include the non-linearities associated with backlash and deadband, and assumes that the Coefficient of Viscous Friction is a constant.

ii) The two test input signals used in Sections 8-3 and 8-5 were the same up to the transition at t = 1.0 seconds. Accordingly the columns headed 'Mean @ t=(1.0-T)' in Tables 8-1 and 8-3 should provide the same values. The differences between these corresponding values give an indication of the repeatability expected of these experimental results.

iii) The transitions of the input signal again cause discontinuities in the estimate vs. time traces. These discontinuities are larger in Section 8-5 (with motor reversals), than those of Section 8-3 (without motor reversals). The non-modelled behaviour of the real system clearly provides increased excitation to the estimator, which unfortunately does not result in improved estimates.

iv) The rise in the value of the 'Final Mean Estimate' with increased sampling rate is not so significant in Section 8-5 (with motor reversals) as it was in Section 8-3

(without motor reversals). In Section 8-3 the 'Final Mean Estimate' rose from 6.74 radians per second (T = 100 msecs) to 20.01 radians per second (T = 2 msecs). These values correspond to a change in pole position from 9.49 radians per second to 14.03 radians per second, when the input signal causes the motor to switch from its extreme speed in one direction, to the extreme in the other direction.

The input signal in this section, Section 8-5, therefore provides greater excitation to the system. It is argued here that this increase in excitation makes the estimator accuracy less sensitive to an increase in sampling rate.

v) In Figure 8-20 it is seen that the high sampling rate estimator again has an unacceptably slow convergence towards its asymptotic value. An hypothesis to account for this slow convergence was tested, the results of this test are presented below in Figures 8-21 and 8-22.

It was hypothesised that the high sampling rate estimator might suffer from being 'swamped' by many 'low-information' content signals at the start of its operation. Such 'swamping' might result in the adaptive gain of the estimator falling too rapidly, prior to the estimator receiving signals permitting a more accurate estimate of pole location, p.

Figures 8-21 and 8-22 show the fall of adaptive gain for the RLS estimator with sampling rates of 10 and 200 samples per second respectively.

The adaptive gain of the RLS estimator may be interpreted as $\underline{W}_k.D^{-1}$ (as per Equation 4-59), where $\underline{W}_k$ is here a vector of three elements. The estimate of pole location is calculated as per Equation 5-5, using $\hat{b}_2$, the third element of the Parameter Vector $\underline{\theta}_k$.

i.e.

$$\hat{b}_2 = \theta_3 \qquad 8\text{-}1$$

The third element of $\underline{W}_k$, i.e. W_3, is the element used to determine the adaptive gain plotted in Figures 8-21 and 8-22. i.e.

$$Adaptive\ Gain = W_3.D^{-1} \qquad 8\text{-}2$$

Examination of Figures 8-21 and 8-22 shows that the adaptive gain of the high sampling rate estimator did not fall faster than that of the low sampling rate estimator. The hypothesis concerning "swamping" is thus disproven.

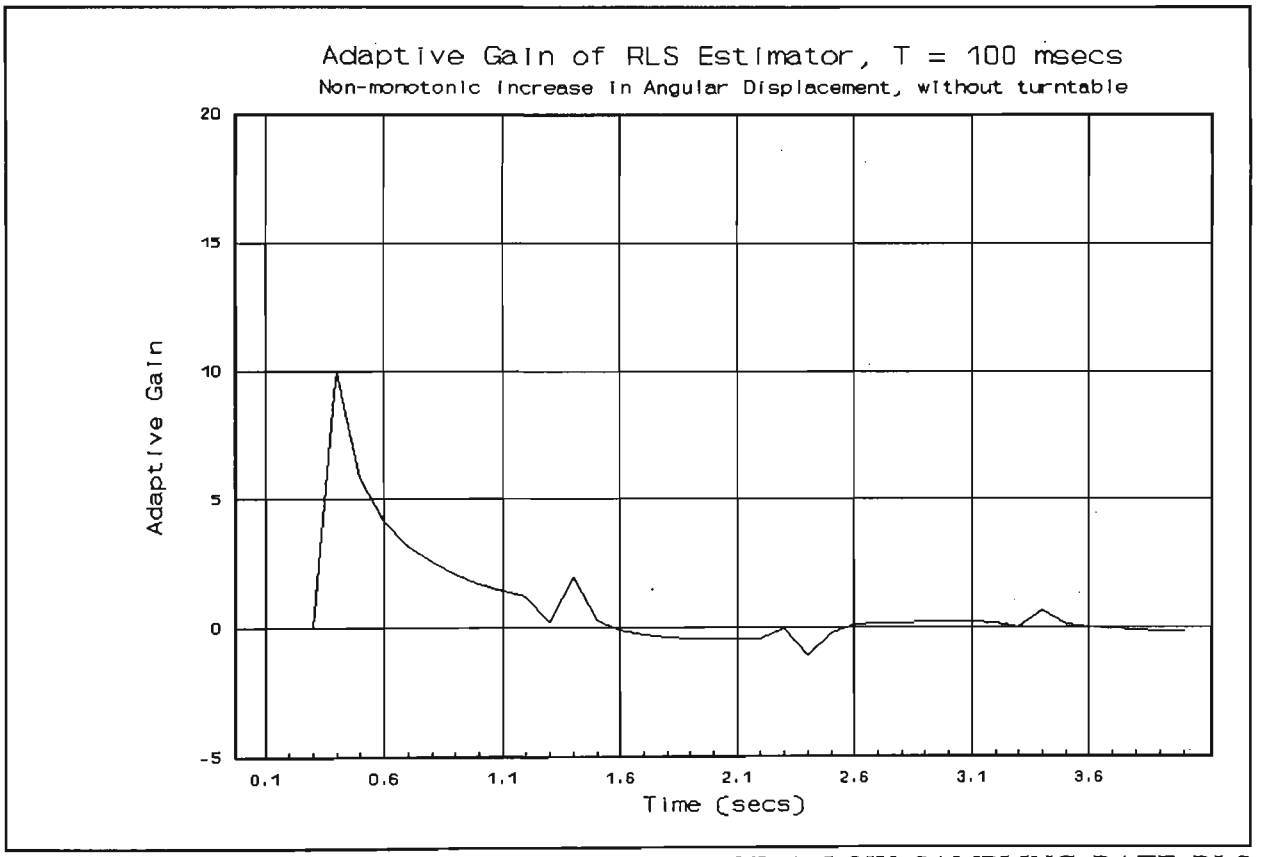

FIGURE 8-21 DECREASE OF ADAPTIVE GAIN OF A LOW SAMPLING RATE RLS ESTIMATOR

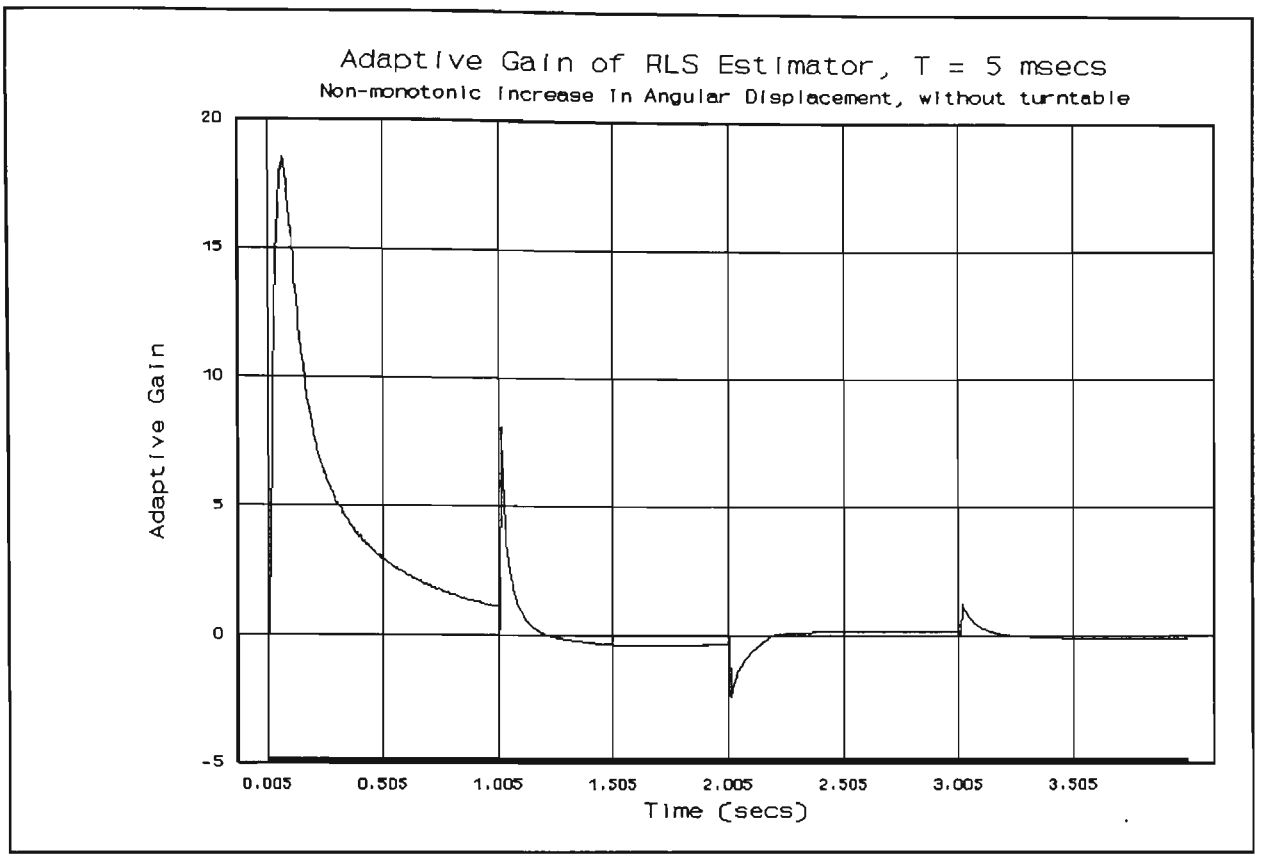

FIGURE 8-22 DECREASE IN ADAPTIVE GAIN OF AN HIGH SAMPLING RATE RLS ESTIMATOR

CONCLUSION DRAWN FROM THE ABOVE RESULTS

In this section (as compared with Section 8-3) the estimator is presented with an improved test input signal, which provides more excitation to the system. However this signal causes the motor to operate over a wider range, that includes non-linearities. As a consequence the model of the plant is less valid. This inadequacy of the plant model results in inferior estimates.

8-6 USE OF THE THREE PARAMETER LU FACTORISATION ESTIMATOR WITHOUT MOTOR STOPPING OR REVERSAL

THE REASON FOR RECONSIDERING THE LU FACTORISATION ALGORITHM

The RLS algorithm was adopted as the preferred method following the work detailed in Section 5-5. The computer simulations of that section revealed that the LU Factorisation

algorithm was prone to loss of its numerical stability.

In Section 5-4 it was noted that the LU Factorisation Method outperformed the RLS algorithm in some high sampling rate applications. Provided the LU Factorisation retains its stability, it is superior to the RLS Method. Further, the stability of the LU Factorisation Method may be enhanced by:

i) Pivotting (at a penalty of increased computation time)

or ii) Information Matrix resetting offers the advantage of inhibiting the drift of the Least Squares Equations towards a singular condition.

The results of Sections 8-3 to 8-5 inclusive show the RLS algorithm estimates becoming less accurate with increased sampling rate. These results, and the earlier simulations of Section 5-4 indicated that the RLS algorithm is inferior to the LU Factorisation Method when presented with small amplitude signals.

THE EXPERIMENTAL SET-UP

The work of Section 8-3 was repeated using a different estimator. The Three Parameter RLS algorithm of Section 8-3 being replaced by a Three Parameter LU Factorisation algorithm. Neither pivotting nor Information Matrix resetting were used, these refinements being unnecessary due to the estimator retaining its numerical stability throughout the duration of each four second run.

At the start of each run the Information Matrix and the parameter estimates were all initialised with values of zero.

RESULTS FROM THE THREE PARAMETER LU FACTORISATION ESTIMATOR WITHOUT MOTOR STOPPING OR REVERSAL

Figures 8-23 to 8-28 show the result of using the Three Parameter LU Factorisation Estimator, without the turntable. Figures 8-23 to 8-28 compare directly with Figures 8-3 to 8-8 (which resulted from the use of the 3 Parameter RLS Estimator) respectively.

Table 8-4 compares the estimates produced by these two estimators.

TABLE 8-4 COMPARISON OF FINAL ESTIMATES FOR THE SYSTEM USING THE RLS AND LU FACTORISATION ESTIMATORS

ESTIMATOR PERIOD (msecs)	FINAL MEAN 3RLS (Rad/sec)	FINAL MEAN 3LU (Rad/sec)
100	6.74	6.63
50	6.97	6.80
20	7.33	7.11
10	7.80	7.22
5	9.29	7.21
2	20.01	7.86

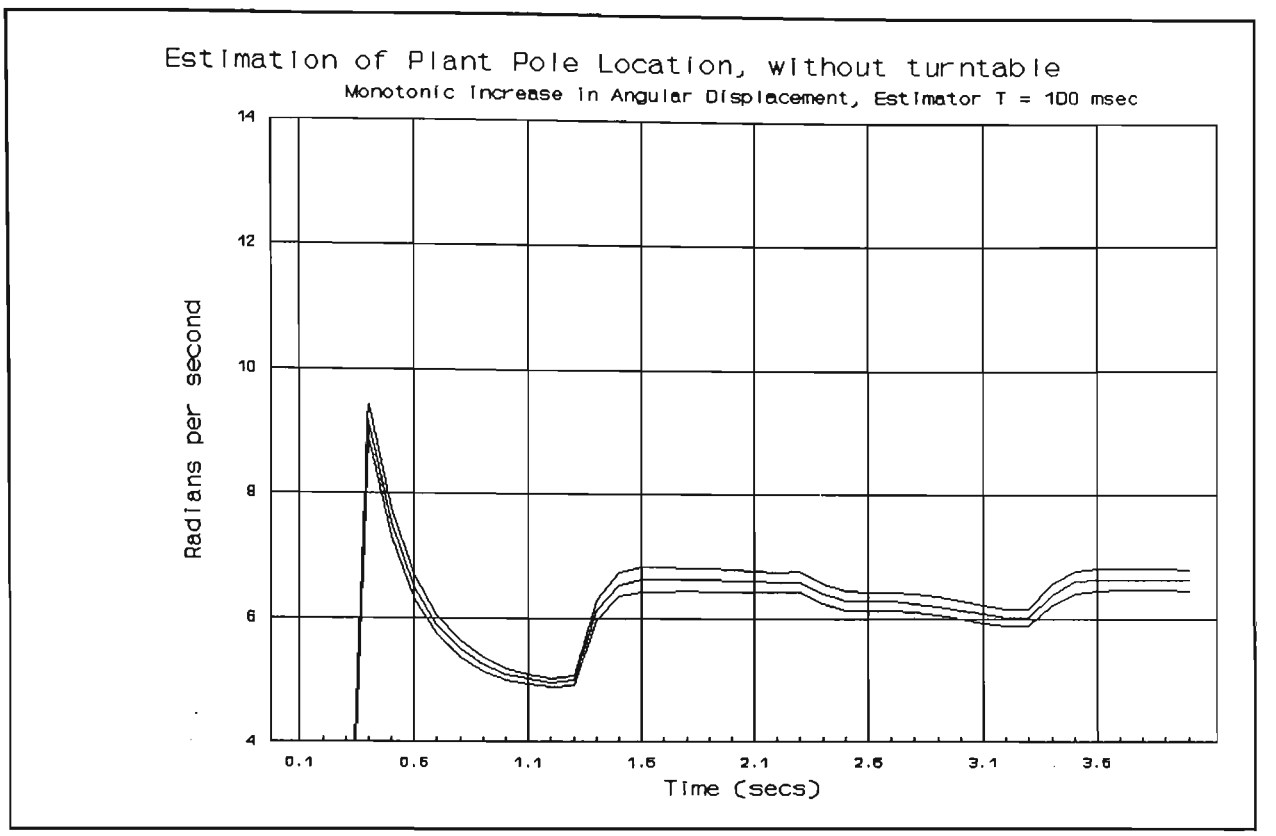

FIGURE 8-23 THREE PARAMETER LU FACTORISATION (LOWEST SAMPLING RATE) T = 100 ms

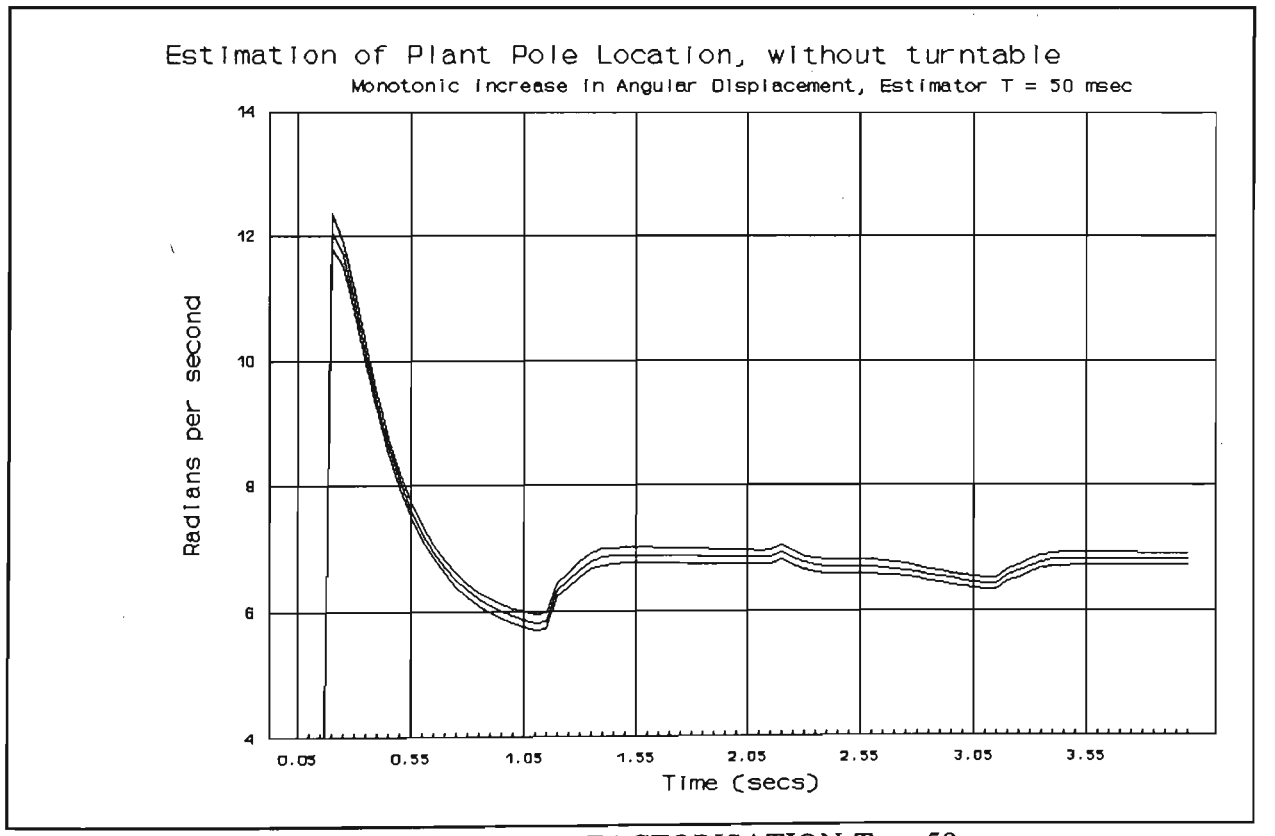

FIGURE 8-24 THREE PARAMETER LU FACTORISATION T = 50 ms

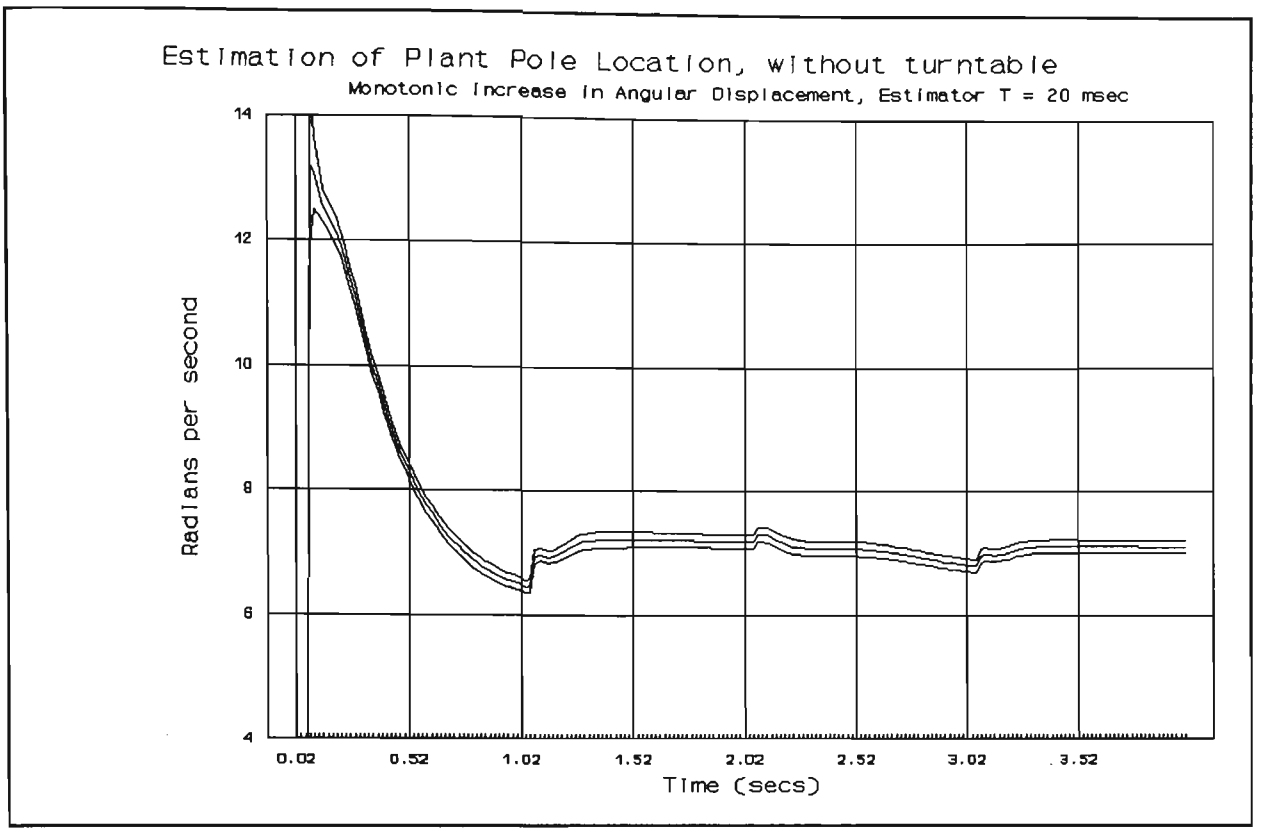

FIGURE 8-25 THREE PARAMETER LU FACTORISATION T = 20 ms

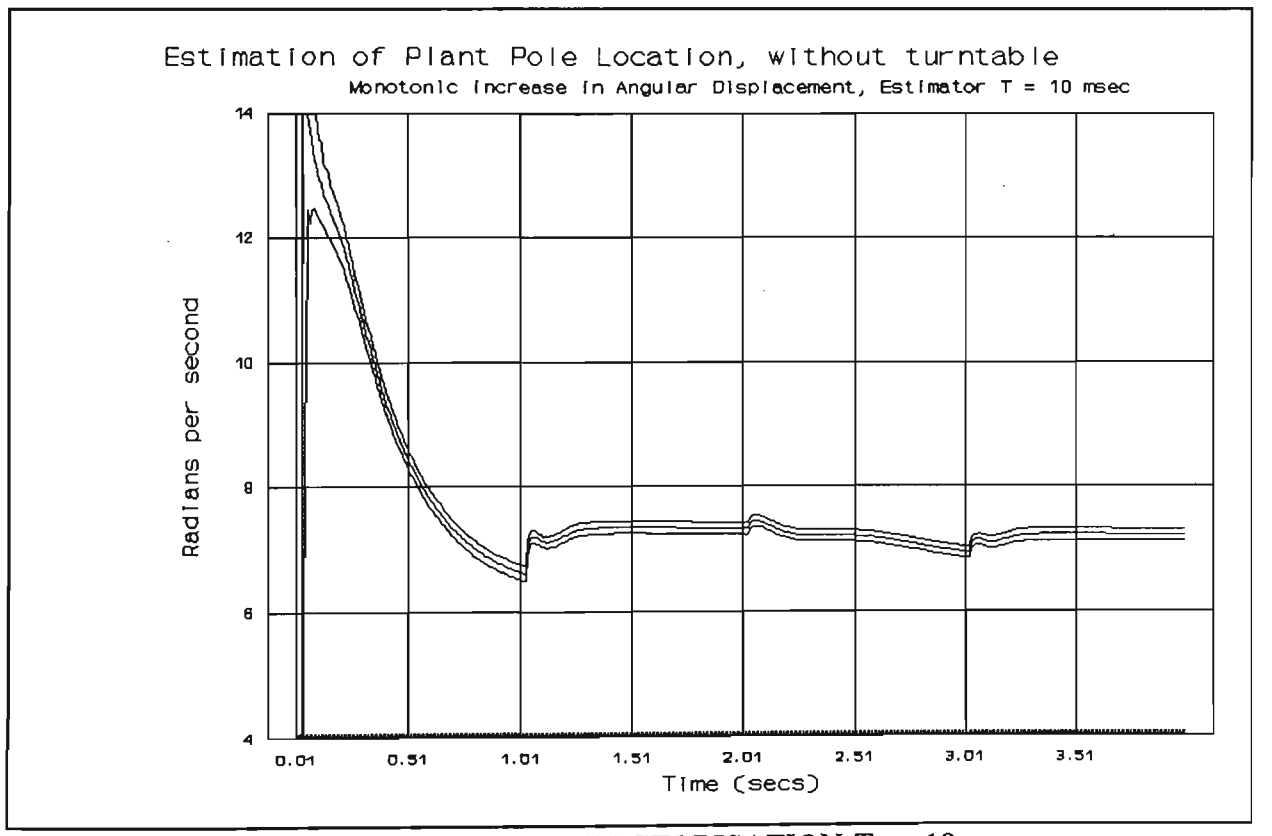

FIGURE 8-26 THREE PARAMETER LU FACTORISATION T = 10 ms

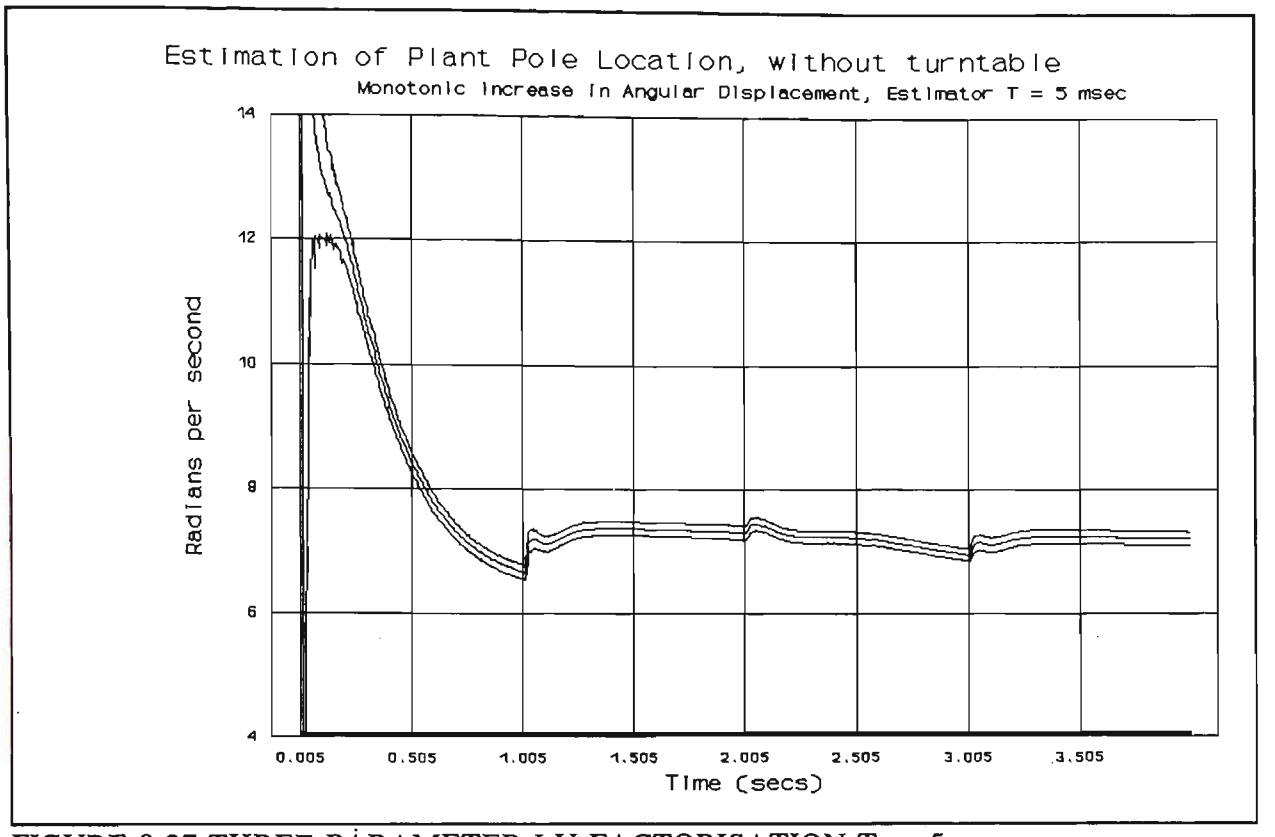

FIGURE 8-27 THREE PARAMETER LU FACTORISATION T = 5 ms

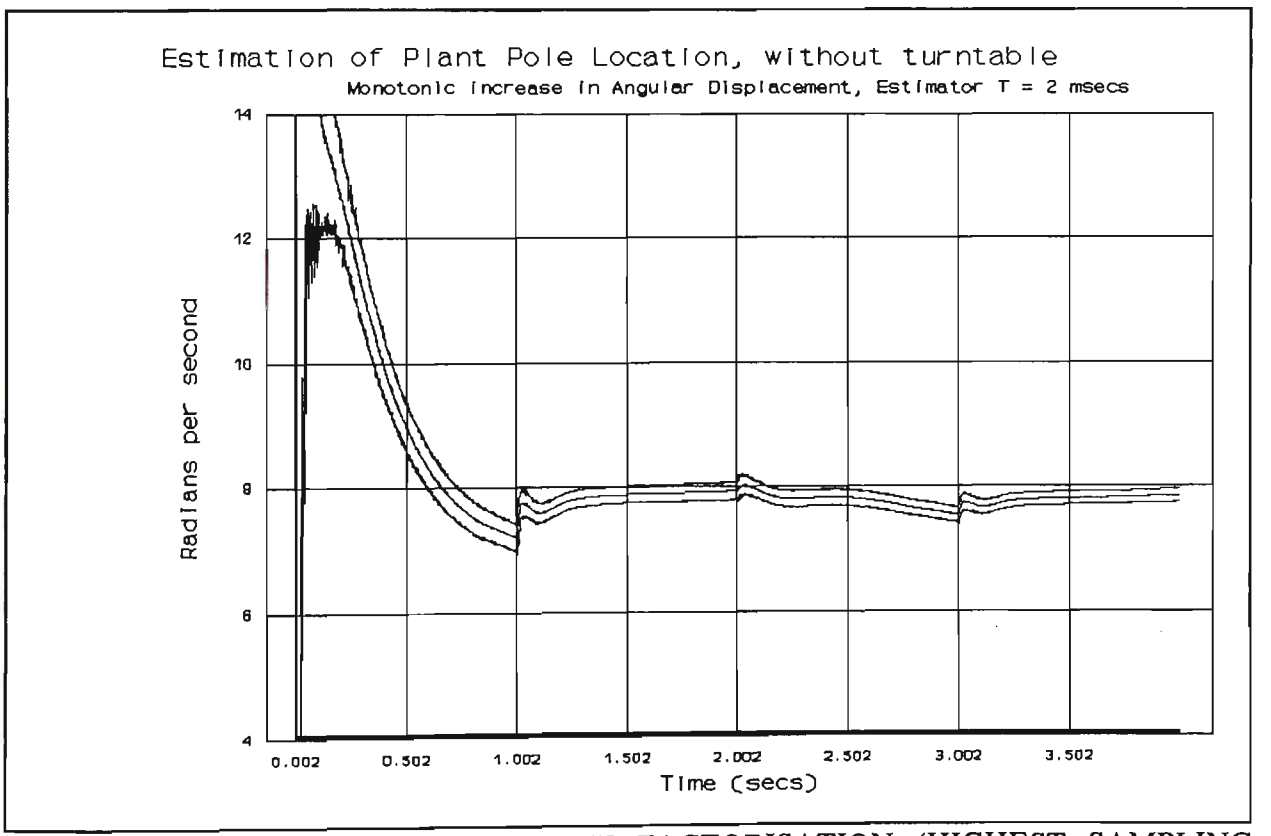

FIGURE 8-28 THREE PARAMETER LU FACTORISATION (HIGHEST SAMPLING RATE) T = 2 ms

INTERPRETATION OF THE RESULTS USING THE LU FACTORISATION ESTIMATOR WITHOUT MOTOR STOPPING OR REVERSAL

i) The 'Final Mean' estimate produced by the LU Factorisation Method shows an increase with increased sampling rate. The LU Factorisation Method offers the advantage of being less sensitive to the selected sampling rate than the RLS algorithm.

ii) In all cases the LU Factorisation Method produced 'Final Mean' estimates closer to the value obtained from the Speed Step-Response of Section 8-2. The Speed Step-Response suggesting a finite pole located at 6.1 radians per second.

iii) Notwithstanding the above comments, the LU Factorisation Estimator returned less accurate estimates as the sampling rate increased.

iv) In each case the effect of a transition in the input signal resulted in a reversal in the trend of the estimates of pole location. After the falling edges of the input signal (i.e. during the low motor speed periods of $1 < t < 2$ and $3 < t < 4$) there was a tendency for the estimated value to rise. Conversely, during the high motor speed periods (i.e. $0 < t < 1$ and $2 < t < 3$ seconds) the estimates tended towards lower values.

An explanation of this effect is that the estimator used an excessively simple model of the plant. In minimising the chosen performance index the estimator must produce different values for the estimate for the different motor speeds. This explanation is reinforced by the results to be presented in Section 8-7, in which the Information Matrix was reset with each change in the input signal.

v) The 'discontinuities' in the traces of estimate vs. time resulting from transitions in the input signal decreased as time increased. As the contents of the Information Matrix grew the estimator tended to average out the preferred value of p. The 'Final Value' (at $t = 4.0$ seconds) is a compromise between the two preferred values for high and low speed of rotation.

CONCLUSIONS DRAWN FROM THE ABOVE RESULTS

i) The LU Factorisation Method outperformed the RLS Method, especially at increased sampling rates.

ii) The raising of the sampling rate did not result in more accurate estimates, nor in an improved convergence rate of the estimates. The results obtained indicate that raising the sampling rate of the estimator led to a deterioration in the quality of the estimates.

8-7 THE THREE PARAMETER LU FACTORISATION METHOD WITH INFORMATION MATRIX RESETTING

INTRODUCTION

The results of Section 8-6 suggest that the estimator attempted to return two different values for the pole location, p. These values corresponding to the two motor speeds resulting from the square wave, test input signals used. In all of the above experiments using the real system, the estimator produced an estimate that was a compromise or average of the two different values. To confirm this interpretation of the results, the experiment of Section 8-5 was repeated using LU Factorisation with Information Matrix resetting.

At the start of each run (t = 0.0 seconds) the Information Matrix and all parameter estimates were initialised with values of zero. At each subsequent transition of the input signal (t = 1.0, 2.0 and 3.0 seconds) the Information Matrix was reset to zero, the parameter estimates were left unchanged. Only at the start of each run (i.e. immediately after t = 0.0 seconds) was it necessary to wait for the data vector to fill, to enable the Least Squares Equations to become soluble.

RESULTS OF USING INFORMATION MATRIX RESETTING

Figures 8-29 to 8-34 show the results obtained using Information Matrix resetting for

the experiment conducted:

i) without the turntable fitted

and ii) with the DAC switching between its 'Maximum Positive' and 'Maximum Negative' output voltages (i.e. with reversals of direction of rotation).

INTERPRETATION OF THE RESULTS FOR INFORMATION MATRIX RESETTING

i) The results confirm that the Least Squares estimator attempted to fit two values of p to the servosystem, corresponding to the two values of motor speed used. This reveals the inadequacy of the simple, linear model used to describe the servosystem.

ii) At the higher sampling rates the +/- two standard deviation spread was observed to have increased in width. This is especially evident in Figure 8-34, for the highest sampling rate of 500 samples per second. The crude resetting of the Information Matrix to zero made the estimator very sensitive to noise in the signal sequences.

iii) Consider the initial convergence of the estimates through the periods $0.0 < t < 1.0$ seconds. As the sample rate was increased the value of the estimate at $t = 1.0$ seconds also increased. This is an indication that the increase in sampling rate resulted in a deterioration in the estimate convergence rate.

CONCLUSIONS CONCERNING INFORMATION MATRIX RESETTING

i) Increasing the sampling rate failed to improve the performance of the estimator.

ii) The simple linear model used to describe the real servosystem was inadequate. As a consequence the estimator attempted to switch the value of the estimate of pole location with each change in input signal, so as to minimise the estimator's Performance Index.

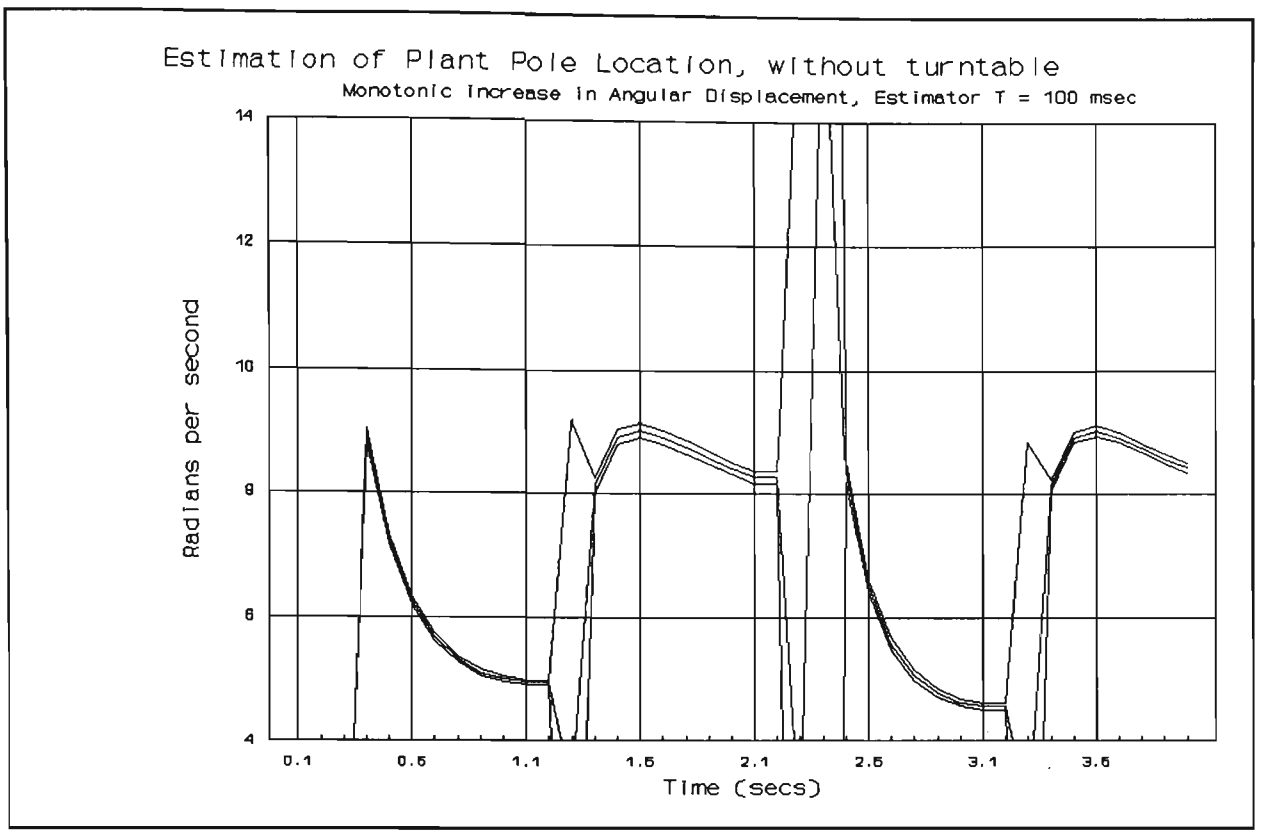

FIGURE 8-29 THREE PARAMETER LU FACTORISATION (LOWEST SAMPLING RATE) T = 100 ms, WITH INFORMATION MATRIX RESETTING

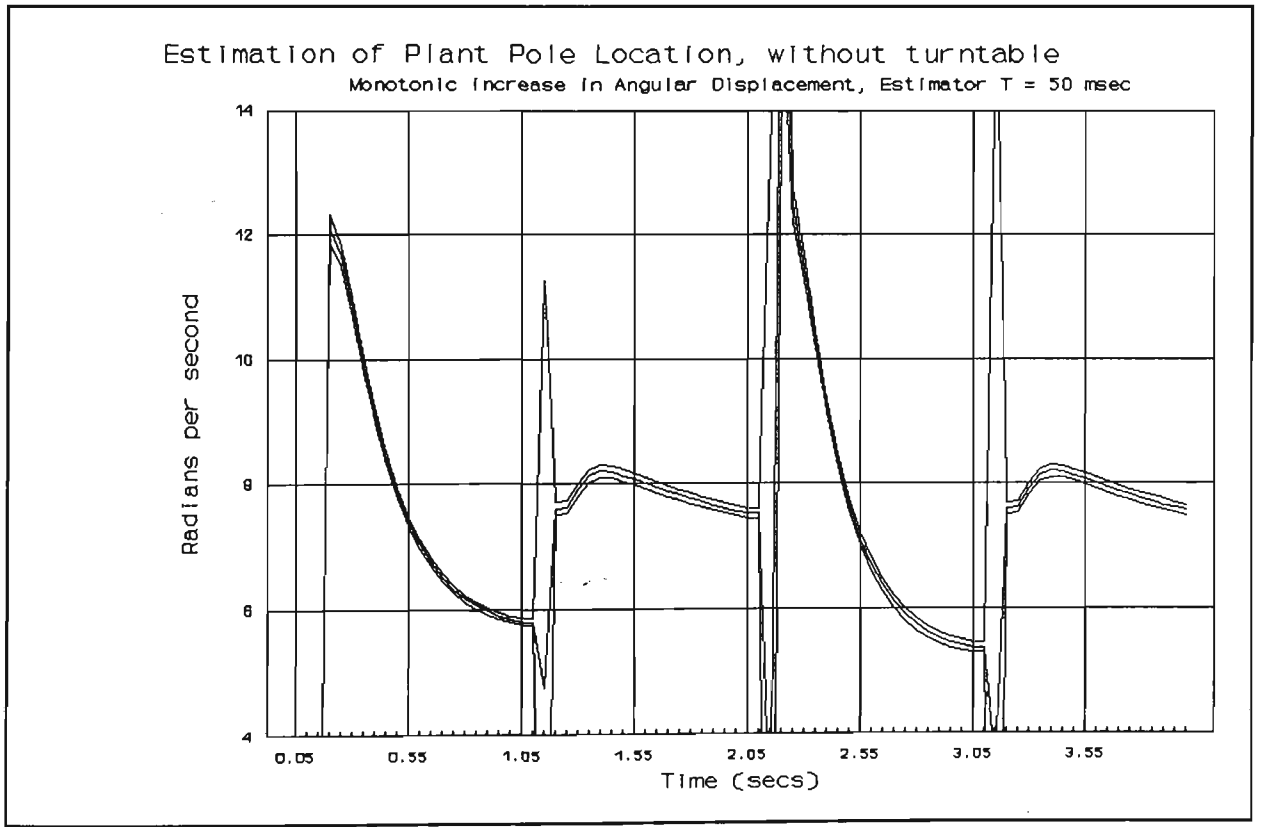

FIGURE 8-30 THREE PARAMETER LU FACTORISATION T = 50 ms, WITH INFORMATION MATRIX RESETTING

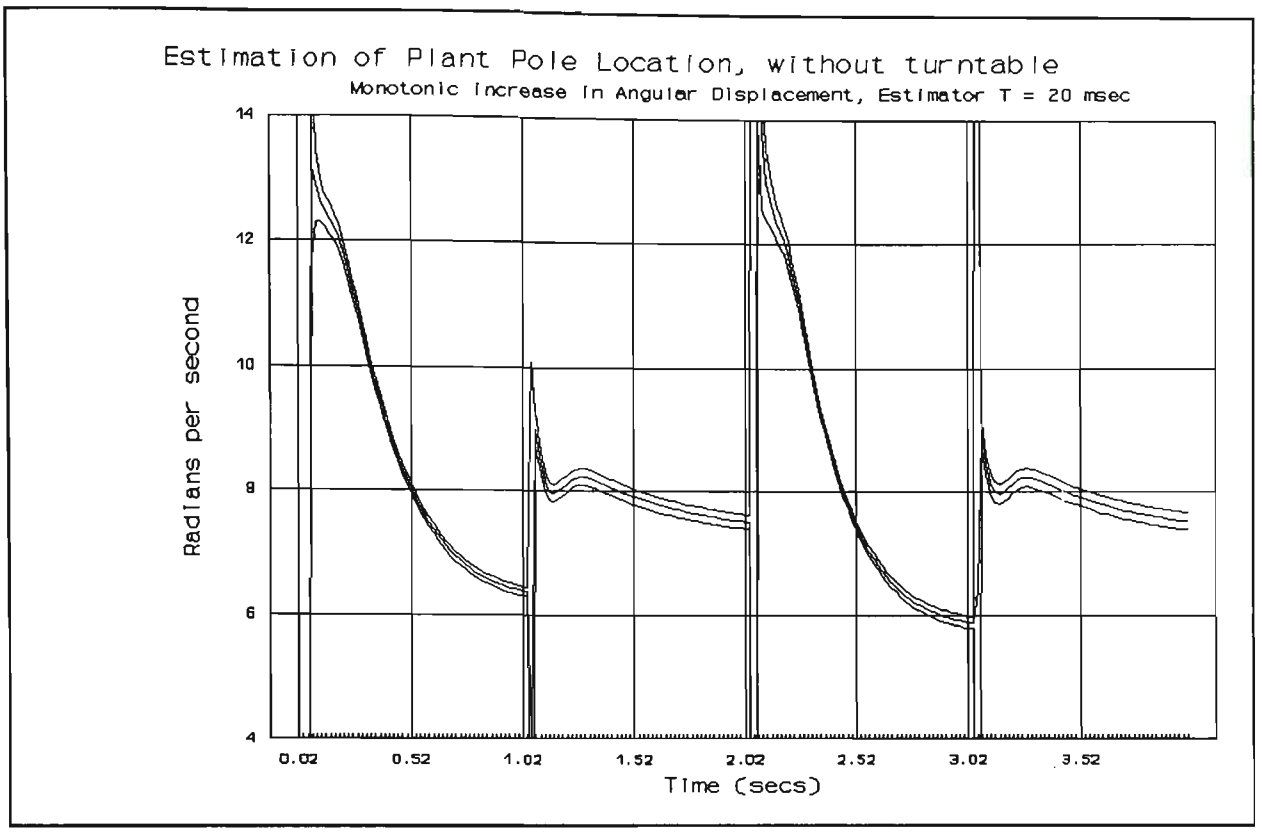

FIGURE 8-31 THREE PARAMETER LU FACTORISATION T = 20 ms, WITH INFORMATION MATRIX RESETTING

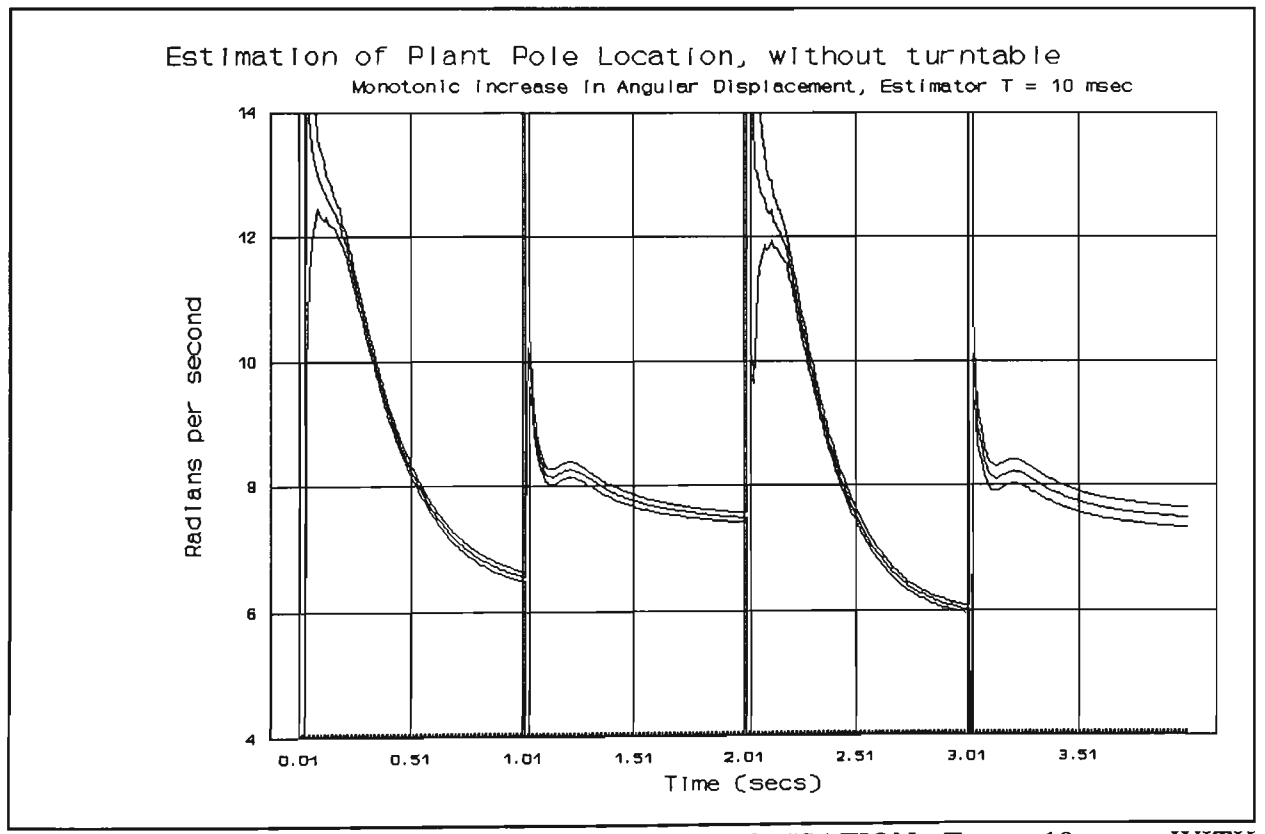

FIGURE 8-32 THREE PARAMETER LU FACTORISATION T = 10 ms, WITH INFORMATION MATRIX RESETTING

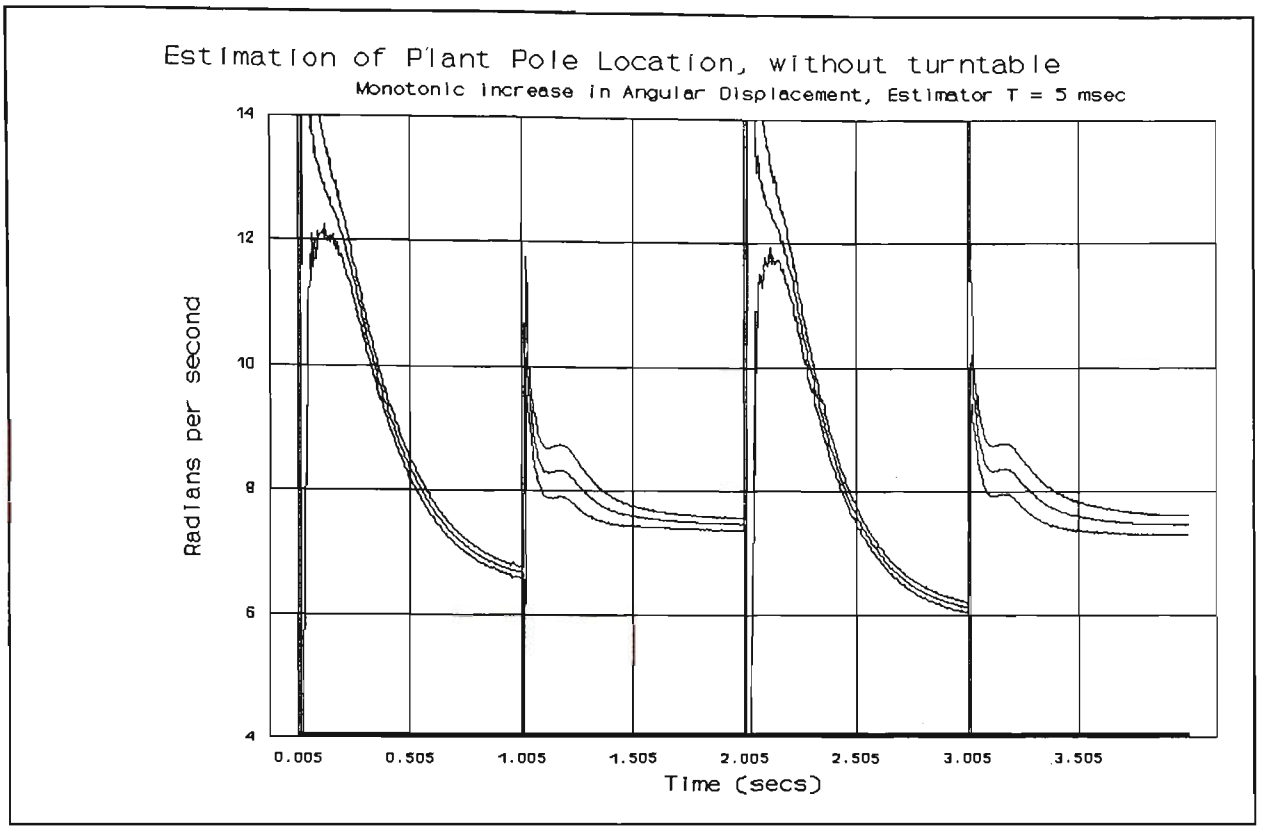

FIGURE 8-33 THREE PARAMETER LU FACTORISATION T = 5 ms, WITH INFORMATION MATRIX RESETTING

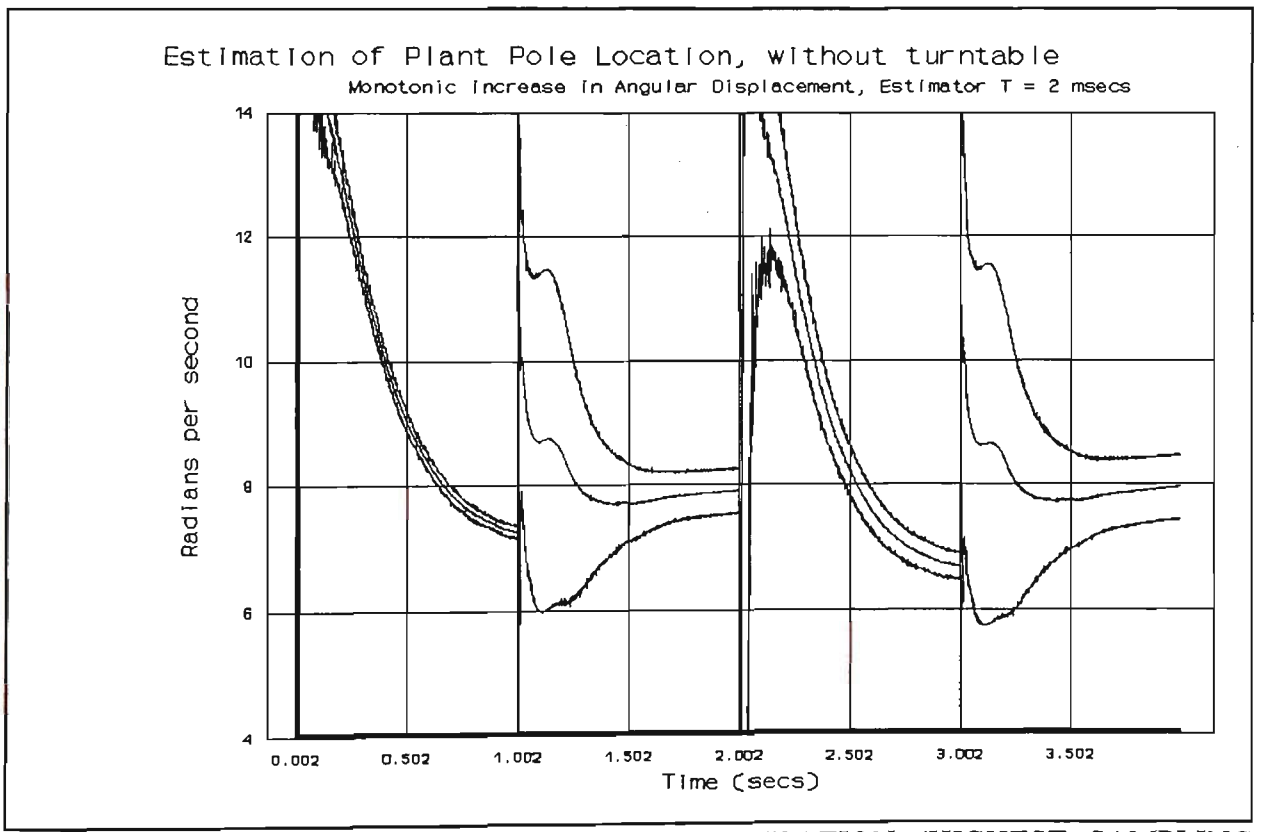

FIGURE 8-34 THREE PARAMETER LU FACTORISATION (HIGHEST SAMPLING RATE) T = 2 ms, WITH INFORMATION MATRIX RESETTING

CHAPTER 9 CONCLUSIONS

9-1 INTRODUCTION

The work described in this thesis originated from a need to perform parameter estimation as rapidly as possible. One method of reducing the parameter estimation time involved increasing the sampling rate of the estimator. This increased the number of signal-pair samples available to the estimator in any given interval of time. Section 9-2 comments upon the failure of this approach.

Section 9-3 provides conclusions on the choice of algorithm for solving the least squares equations. With hindsight the initial selection of methods of solution was naive. The work revealed the importance of the problems of numerical instability in the solution of sets of least squares equations.

Section 9-4 presents further conclusions concerning the algorithms considered. The more complex algorithms (Extended Least Squares and those using instrumental variables) were found to offer no advantages for this work. The greatest success was achieved by integrating prior knowledge of the integral action of the plant into the model used by the estimator.

Finally in Section 9-5 some observations are made on the application of the transputer in this work.

9-2 INVESTIGATION OF INCREASED SAMPLING RATE AS A MEANS OF REDUCING THE PARAMETER ESTIMATION TIME

CONCLUSIONS BASED UPON MEASUREMENTS ON THE REAL SYSTEM

i) Increasing the sampling rate of the estimators resulted in a deterioration in the quality of the estimates. This was true for the estimators using both LU Factorisation and RLS algorithms.

ii) The LU Factorisation Method outperformed the RLS Method, especially at the higher sampling rates. The LU Factorisation Method suffering less from the decrease in output signal amplitude, as the sample rate was increased.

iii) Increasing the sampling rate effectively decreased the signal to noise ratio of the output signal sequence. The measurement error of the output sequence was determined by the resolution of the optical encoder and its associated counter. This error was thus independent of the sampling rate. The output signal, being the number of counts between sampling instants, decreased as the sampling rate increased.

iv) The main limitation on estimate convergence of an estimator during its start-up period is the paucity of the information content of the signals available to it. Increasing the sampling rate does not, of itself, offer a means of circumventing this restriction. Increasing both the sampling rate and the level of signal excitation may result in a reduction in the settling time of the estimates.

DISCUSSION CONCERNING THE SIMULATIONS

The results presented in Table 5-6 suggested that an increase in the estimator sampling rate would result in an improvement in both the accuracy and rate of convergence of the estimate of pole location p.

This improvement is something of an artifice of the simulations. The ARMA process of the simulations ran with a periodic time of T = 0.05 seconds. The noise sequence added to the output sequence was folded back into the ARMA process by its passage through the data vector of the process. This noise sequence thus served as a supplementary source of excitation of the autoregressive (AR) section of the process.

The highest sampling rate estimator of the simulations also operated with a sampling period of T = 0.05 seconds and thus sampled all elements of the noise corrupted, output sequence of the ARMA process. The other estimators (with T = 0.2 s and T = 0.5 s) only

accessing every fourth or every tenth element of this sequence, therefore had a more restricted view of the disturbance excitation of the AR section of the process.

In a real system, the problem is one of measurement noise rather than an additive disturbance signal so an increase in the estimator sampling rate would not therefore result in an increased excitation of the AR section of the process.

SELECTION OF SAMPLING RATE

The strategy of increasing the sampling rate is one whereby the intended time-window used for parameter estimation is shortened. The narrowing of this window necessarily reduces the quality of the signals available to the estimator because of:

i) a reduction in the number of transitions of the input signal

and ii) a reduction in the quantity of output signal variation.

The results presented in this thesis indicate that the estimator time-window should not be made too short, with respect to the time constant of the process. Improvements in microprocessor technology will permit increased sampling rates, by the reduction of required time for computation. The above comments show that, rather than use this spare computational capacity to increase sampling rates, a more profitable avenue would involve the use of either a more elaborate system model, or the incorporation of some supervisory, expert system.

9-3 USE OF INFORMATION MATRIX RESETTING TO IMPROVE THE NUMERICAL STABILITY OF PARAMETER ESTIMATORS

THE PROBLEM OF NUMERICAL INSTABILITY

All least squares estimation algorithms are prone to loss of numerical stability. For a particular least squares algorithm the following factors will all influence the maintenance of stability:

i) input signal

ii) noise levels

iii) sampling rate

iv) duration of the estimation process

There are a variety of techniques that may be used to enhance the stability of the solution of the least squares equations:

i) Pivotting

It is well known that the direct methods (e.g. Gaussian Elimination, Gauss-Jordan Method) require the use of pivotting [9-1]. Experience gained in this work confirms the advice given by Press et al [9-2] that the LU Factorisation Method also requires pivotting to enhance its stability.

ii) Square Root Methods and Double Precision Arithmetic

At the expense of increased computation time the onset of instability may be delayed by the use of either square root methods or double precision arithmetic.

iii) Singular Value Decomposition

Press et al [9-3] recommend the use of Singular Value Decomposition for the solution of sets of least squares equations.

The LU Factorisation Method used in this work was not selected on the basis of its numerical stability, and this was found, on occasions, to be a problem. Superficially LU Factorisation resembles Cholesky Decomposition, since the information matrix of the estimator is always symmetrical [9-4].

i.e. for the LU Factorisation Method

$$\underline{R} = \underline{L}.\underline{U} \qquad 9\text{-}1$$

where the elements of the main diagonal of $\underline{L}$ are all unity. Whereas for Cholesky Decomposition :

$$\underline{R} = \underline{L}.\underline{L}^T \qquad 9\text{-}2$$

For LU Factorisation the first row, first column element of $\underline{U}$ is equal to that of $\underline{R}$, i.e.

$$r_{11} = u_{11} \qquad 9\text{-}3$$

For Cholesky Decomposition the corresponding equation is:

$$r_{11} = l_{11}.l_{11} \qquad 9\text{-}4$$

The Cholesky Decomposition should therefore possess superior numerical properties, but does require the need to use a "Square Root" procedure.

Bierman [9-5] proposed a "Square Root Free Cholesky Decomposition"

$$\underline{R} = \underline{L}.\underline{D}.\underline{L}^T \qquad 9\text{-}5$$

Press et al [9-4] comment that pivotting is not necessary when using Cholesky Decomposition. Bierman [9-6] comments that matrix triangular factorisations are "generally unaffected" by the use of pivotting.

The one clear point in all of this is that there is a significant probability of not obtaining a correct solution of the least squares equations. None of the above methods, with or without pivotting, guarantee numerical stability.

The RLS Method proved to be more robust throughout both the simulations and practical experiments. The literature does however contain references to numerical problems with this method [e.g. 2-14,2-16]. Unfortunately, in the work considered here, the RLS algorithms proved less useful due to the slower convergence of their estimates towards asymptotic values.

THE PREFERRED SOLUTION

For the work considered in this thesis, the solution of the least squares equations using an information matrix is preferable to the use of an RLS algorithm. Table 8-4 presented results which indicate that the RLS algorithm was both less accurate in its predictions and less robust to errors resulting from an increased sampling rate. It is necessary however to ensure that the

chosen solution maintains its numerical stability.

Frequent resetting of the information matrix should ensure that the least squares equations are not permitted to drift towards a singular condition. Such resetting should also bestow adaptibility upon the estimator.

The resetting strategy used in Section 8-7 was very primitive on two counts. Firstly a very crude trigger for information matrix resetting was employed. Secondly the matrix was reset to zero, making the subsequent estimates extremely sensitive to noise. However this strategy was adequate to display the inadequacies of the mathematical model, as explained in Section 8-7.

The following factors should all be considered in determining a suitable instant for information matrix resetting, e.g.

i) the magnitude of the change in input signal value

ii) the magnitude of the prediction error

iii) the 'richness' of the input signal in a particular period (i.e. whether or not good estimates are achievable from a given section of the input signal sequence)

iv) a knowledge of the non-linearities in the system.

The trigger and degree of resetting may well be best supervised by some form of expert system.

9-4 COMPARISONS OF DIFFERENT LEAST SQUARES ALGORITHMS

A total of six different Least Squares algorithms were considered in this work:

i) a standard, four parameter, LU Factorisation algorithm

ii) a standard, four parameter, RLS algorithm

iii) a reduced-order, three parameter, LU Factorisation algorithm

iv) a reduced-order, three parameter, RLS algorithm

v) a reduced-order, three parameter, LU Factorisation algorithm utilising

instrumental variables

vi) a Recursive Extended Least Squares algorithm with six parameters.

A number of conclusions were made based on the experience gained in considering these different methods:

i) Recursive algorithms usually have shorter execution times than the non-recursive, direct solutions, this being especially true as the number of unknown parameters increases. However, for simple models of four (or less) parameters there is little advantage in using a recursive method as a means of reducing the required time for solving the least squares problem.

ii) Recursive Least Squares is more numerically robust than the direct methods considered in this work.

iii) LU Factorisation algorithms outperform RLS algorithms when presented with signals from a weakly excited system.

iv) The Extended Least Squares (ELS) and Instrumental Variables (IV) Methods offered no advantages over the standard algorithms in the work considered here. Both ELS and IV methods are primarily designed to tackle the problem of estimate bias due to correlated residuals. During the initial period of the estimation process these correlated residuals have had little opportunity to build up within the information matrix (or conversely to influence the covariance matrix of a recursive method).

v) A number of different methods of generating instrumental variables were considered. Most success was achieved by deriving the instrumental variables from a delayed version of the output sequence.

vi) Estimators using instrumental variables derived from an adaptive model of the unknown process proved to be numerically ill-behaved. This behaviour raises doubts as to their performance and stability. Having concluded that IV methods were not a

probable source of superior estimates, the cause of these numerical problems was not investigated.

vii) Experiments revealed that IV methods were best operated initially without the use of instrumental variables. The instrumental variables were incorporated into the solution of the estimation-problem (no longer the least squares equations !) after some empirically chosen delay.

viii) In terms of both reduced computation time and superior estimate accuracy and convergence rate the three parameter, reduced order estimators proved the most successful. The reduced order estimators achieved these advantages by using prior knowledge, namely that the servosystem was of Type One (i.e. contained integral action).

9-5 OBSERVATIONS ON THE USE OF A TRANSPUTER IN REAL-TIME DIGITAL CONTROL

The transputer design offers a number of very useful features for applications in control engineering:

i) It forces a modular approach to the design which eases the subsequent expansion or enhancement of the control system. Such development may involve the addition of an extra processor to enable more computation to be performed in a given time interval. Alternatively further processes (i.e. tasks) may be added to an existing transputer controlled system, with simple interfaces between the new and old parts. The interfaces between processes are clearly defined by the link interconnections and their associated protocols.

ii) The provision of an easily accessible one microsecond clock enables very accurate timing of events and execution times.

iii) The folding Editor of the TDS software gives good support for sound, top-down designs.

iv) The Occam2 programming language is legible and disciplined, resulting in efficient real-time operation of the transputer.

v) The documentation supporting both software and hardware produced by Inmos has proved to be complete and intelligible.

The main problems encountered were:

i) The restriction of four hardware links on each processor.

ii) The need for the links to be operated in an electrically quiet environment. The simple message acknowledgment of the links is easily upset by a noise spike. In real applications the processes should include safeguards against process deadlocks as a result of such disruptions of inter-process communications.

APPENDIX A DESCRIPTION OF THE TRANSPUTER SYSTEM

A-1 TRANSPUTER HARDWARE

The transputer system used throughout this work was manufactured by Inmos Limited and consisted of the following components:

i) B008 Motherboard, enabling the transputer system to operate within an IBM XT desktop computer [A-1].

ii) A transputer module (TRAM) with a T800-17MHz transputer, 2 MBytes of DRAM and 32 KBytes of static RAM.

iii) Two ADC20 analogue-to-digital TRAMs [A-3].

iv) A digital input-output interface TRAM. The design of this module is described in Appendix B, Section B-6.

Figure A-1 shows the interconnections between the components of the transputer system and their interface to the real servosystem. The "Links" shown are the links of the various transputer modules [A-4]. The numbers on the arrows at each TRAM give the memory offset at which each link is positioned [A-5]. The numbers on the arrows of the C004 Linkswitch indicate the "link numbers" of the C004 integrated circuit [A-6]. These link numbers are dependent upon the printed circuit board layout of the B008 motherboard.

The "Slots" are the locations on the B008 motherboard, where each TRAM may be connected [A-7].

A-2 TRANSPUTER SOFTWARE

CHOICE OF SOFTWARE ENVIRONMENT

Initially the intention was to use the analogue-to-digital converters already available in the Control Laboratory of the Victoria University of Technology. These converters were of type DT2801, manufactured by 'Data Translation' [A-9]. Their use required the first programs

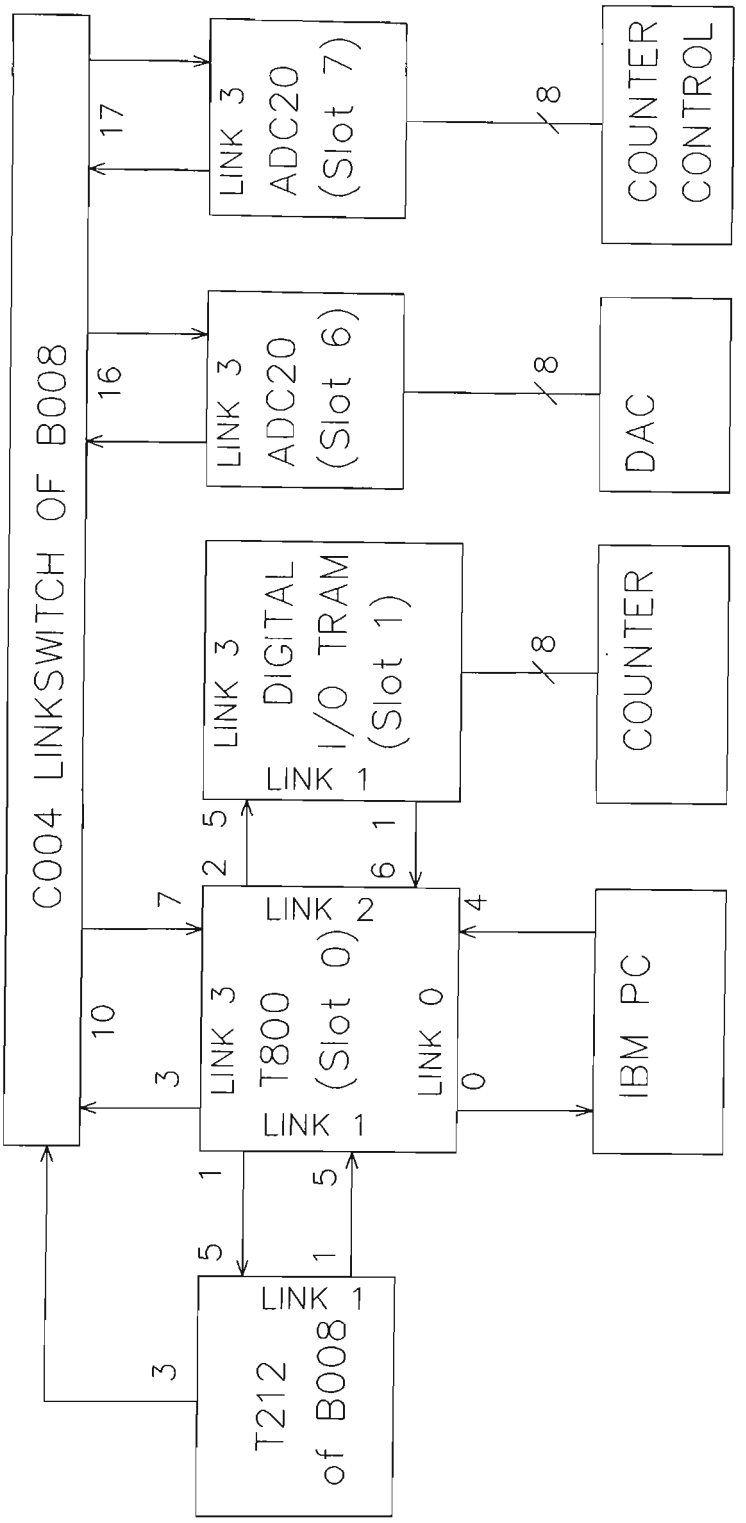

FIGURE A-1 THE INTERCONNECTION OF COMPONENTS OF THE TRANSPUTER SYSTEM AND THE SERVOMECHANISM

to be written using the Inmos Alien File Server, "Afserver" [A-2]. This server included library routines whereby the transputer could access the ports of the IBM PC.

The DT2801 cards required an indeterminate, nominal 'trigger' delay of one millisecond. Apart from this period being unacceptably long (i.e. greater than the computation time of the parameter estimators used !), the indeterminancy resulted in synchronisation problems between the DT2801 card and the time-slicing of the T800 transputer.

This problem was solved by the purchase of two ADC20 TRAMS, providing input-output capability to the transputer system itself [A-3]. This purchase also enabled the programs to be written using the file server of the Transputer Development System (TDS) [A-2]. All of the work presented in this thesis makes use of the TDS and its libraries.

PROGRAMMING LANGUAGE

All programs were written in Occam2 [5-7,5-8]. Samples of the source code are given in Appendices E and G.

A-3 DYNAMIC SWITCHING OF THE SYSTEM CONFIGURATION

EXPLANATION OF 'DYNAMIC SWITCHING'

The transputer design deals with parallel processing by mapping the various processes (tasks) onto the available processors. The processes communicate via "channels", the names of channels and their protocols are fixed by declarations within each program. Processes that communicate with processes on a different processor must have their channel (a software connection) placed onto a transputer "link" (a serial, hardware connection). A major problem with the T800 generation and earlier transputers is their restriction to only four links per processor.

On the B008 motherboard this constraint is alleviated by the provision of a programmable linkswitch, the C004 integrated circuit [A-1]. The B008 motherboard can

support up to ten TRAMs, all of which have at least one of their links connected to the linkswitch.

Inmos provide a software package, the Module Motherboard Software (MMS2), intended to permit the configuration of the links between the TRAMs on the motherboard, prior to their operation of the application software [A-8]. Rather than use this software, the programs have been written so as to permit the application software itself to control the linkswitch. The application program is thus able to reconfigure the hardware as it executes. The hazard in doing this is the risk of breaking a link mid-way through an information exchange.

THE NEED FOR DYNAMIC SWITCHING

Dynamic switching was unavoidable in this work due to financial restrictions. Funds were only available for the purchase of one transputer TRAM. This TRAM had to be placed in Slot 0 of the motherboard, in order to act as the Root Transputer of the system (i.e. to provide an interface to the host IBM XT).

In Slot 0 the transputer has only one link connected to the C004 linkswitch. This is the only hardware connection available that can be connected to the ADC20 interface TRAMs. The ADC20 TRAMs are constructed with only one link (Link 3). Thus for the T800 to communicate with two ADC20 TRAMs it was necessary for the program to be able to switch the links between the TRAMs.

This switching was achieved by loading a short program into the on-chip RAM of the T212 transputer, supplied on the B008 motherboard as a 'linkswitch controller'. The source code of this program is given in Appendix G, under the heading "Source Code Enabling Control of the B008 Motherboard Link Switch".

The availability of only one transputer for all calculations and control led to the tasks being written as a single, sequential process. The programs therefore avoid the hazard of

breaking a currently active link.

A-4 TIMING

The T800 transputer makes available two clocks, with periods of 64 microseconds and one microsecond. The T800 allows parallel processes to have two levels of priority. The low level priority processes having access to the slower clock, whereas the high level priority processes have access to the one microsecond clock. If no priority is specified, the default clock available is the 64 microsecond clock.

As noted above the programs were written as single, sequential processes. However to make the one microsecond clock available, it was necessary to declare this single process as an high priority parallel process. This required the addition of a 'dummy' low priority parallel process, consisting of a single 'SKIP' operation.

APPENDIX B DESCRIPTION OF THE SERVOSYSTEM

B-1 DESCRIPTION OF THE COMPLETE POSITIONAL SERVOSYSTEM

The positional servosystem consists of the following components:

i) A 40 Watts, permanent magnet DC servomotor.

ii) A two-channel, incremental optical encoder, providing a resolution of 1000 counts per armature shaft revolution.

iii) A 71:1 planetary gearhead.

iv) A detachable turntable, mounted on the output shaft of the gearbox.

v) A Pulse Width Modulator (PWM) Unit, used to switch a bridge of VMOS power transistors. This circuitry controlled the armature current of the motor.

vi) Circuitry to interface the above hardware to the Transputer System. This consists of two main sections. Firstly a Digital-to-Analogue Converter (DAC), whereby the Transputer could control the duty cycle of the PWM Unit. Secondly a Counter Circuit, whereby the Transputer could monitor the angular change in position of the armature shaft.

Figure B-1 is a block diagram showing the interconnection of these components, and their connections to the components of the Transputer System.

B-2 THE PERMANENT MAGNET DC MOTOR

The DC motor was manufactured by Interelectric AG of Switzerland, under the trade name of "Maxon", catalogue number 2260-811-51.216.200. The manufacturer provides comprehensive data on this range of motors [3-17], including the following details :-

Assigned power rating	40W
Nominal voltage	18V
No load speed	4500 rpm

Stall torque	880 mNm
Speed/torque gradient	5.35 rpm/mNm
No load current	387 mA
Starting current	24.1A
Terminal resistance	0.747 ohms
Terminal inductance	0.23 mH
Max. permissible speed	5000 rpm
Max. continuous current	3.31A
Max. power output @ 18V	98.9W
Max. efficiency	73.0%
Torque constant	36.5 mNm/A
Mechanical time constant	35.5 ms
Rotor inertia	635 gcm^2
Number of commutator segments	26

B-3 THE OPTICAL ENCODER

The armature of the motor was directly coupled to a Hewlett-Packard HEDS 6010 Digital Optical Encoder. This encoder giving a resolution of 1000 pulses per revolution of the armature shaft [B-2].

B-4 THE PLANETARY GEARHEAD

The motor was fitted with a planetary gearhead, giving a reduction of 71:1. The total no load backlash of the gearhead is specified as no more than 2.7 degrees [3-17].

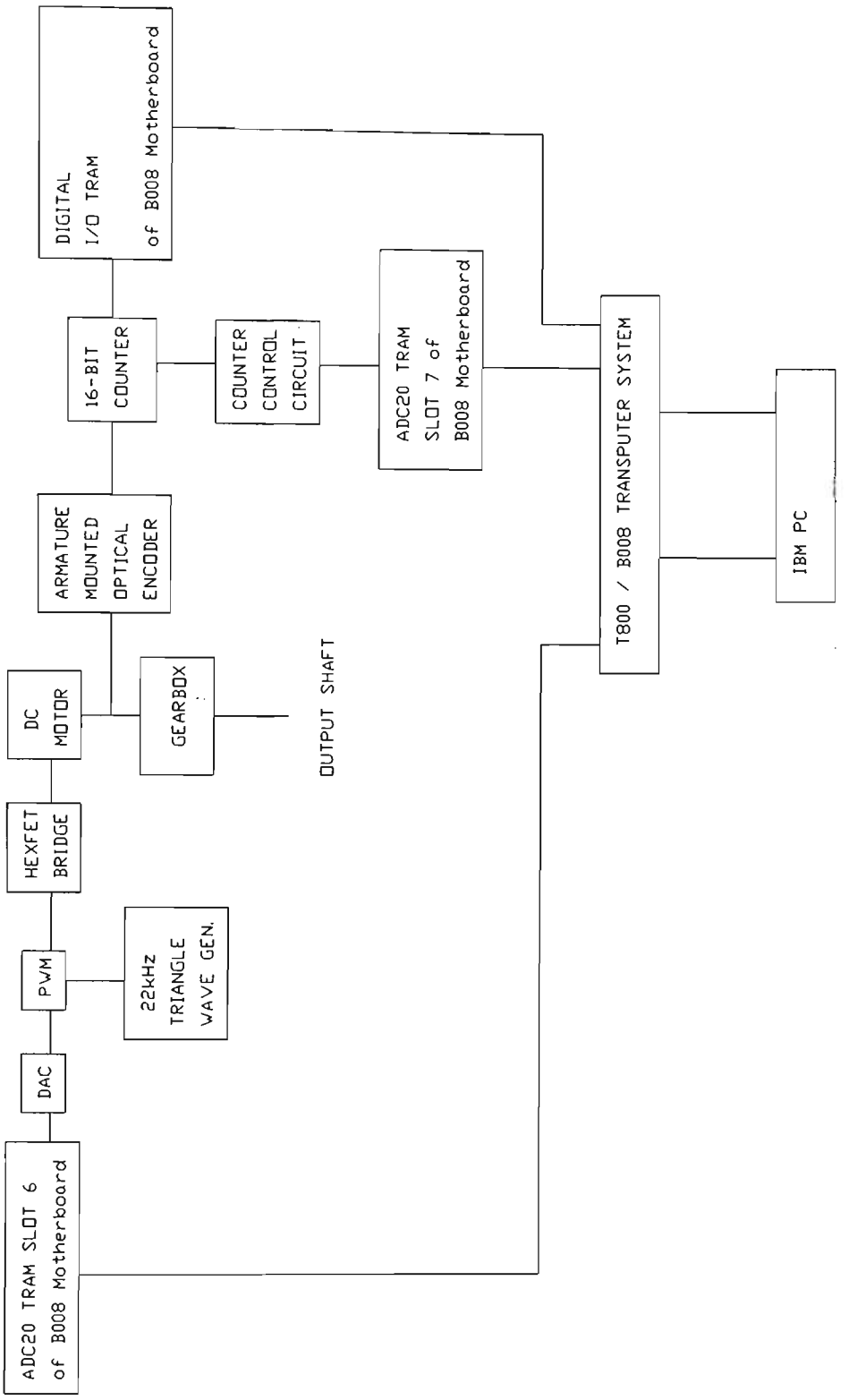

FIGURE B-1 BLOCK DIAGRAM OF THE COMPONENTS OF THE SERVOSYSTEM AND THEIR CONNECTIONS TO THE TRANSPUTER SYSTEM

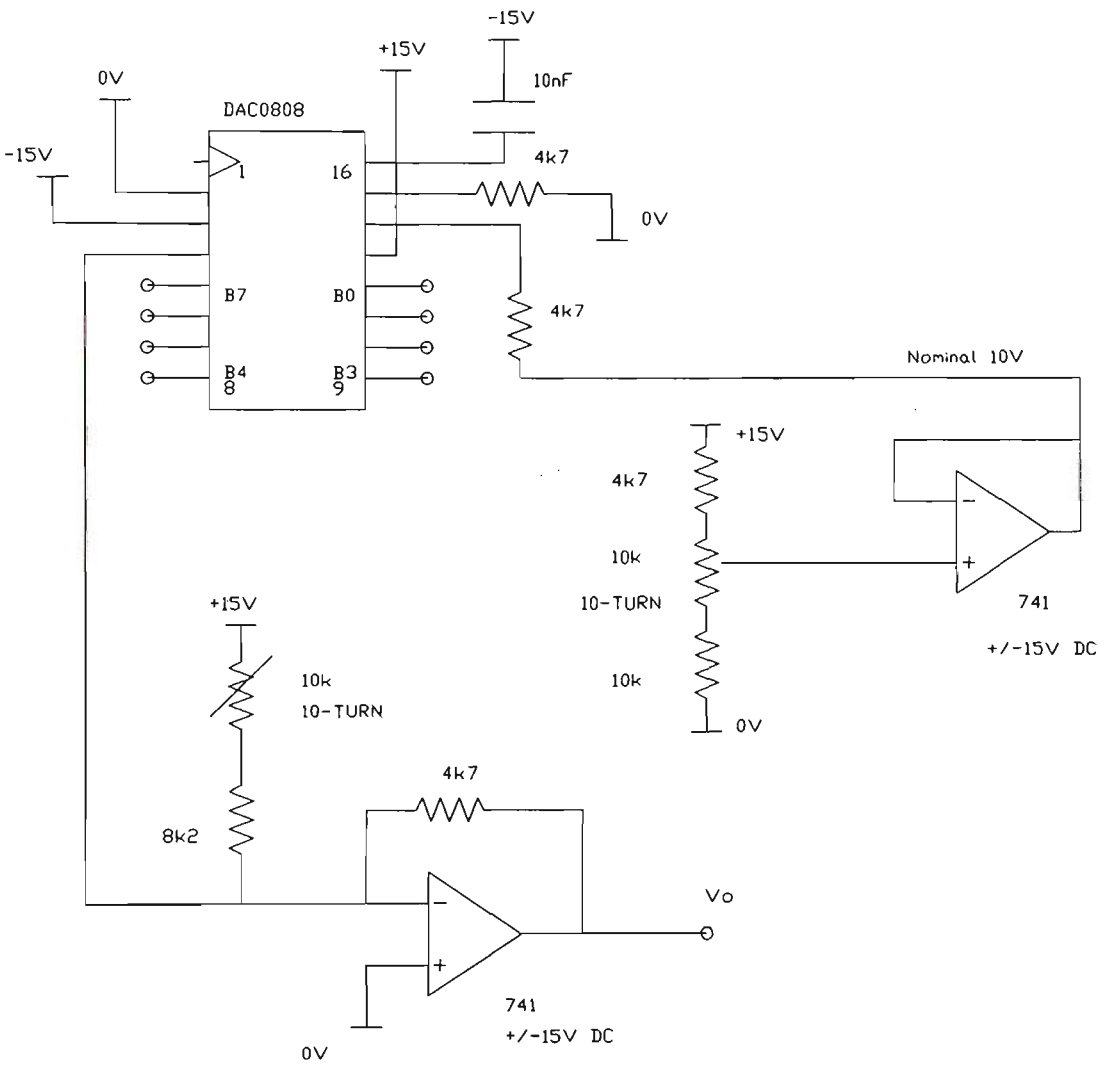

FIGURE B-2 CIRCUIT DIAGRAM OF THE DIGITAL TO ANALOGUE CONVERTER CIRCUIT

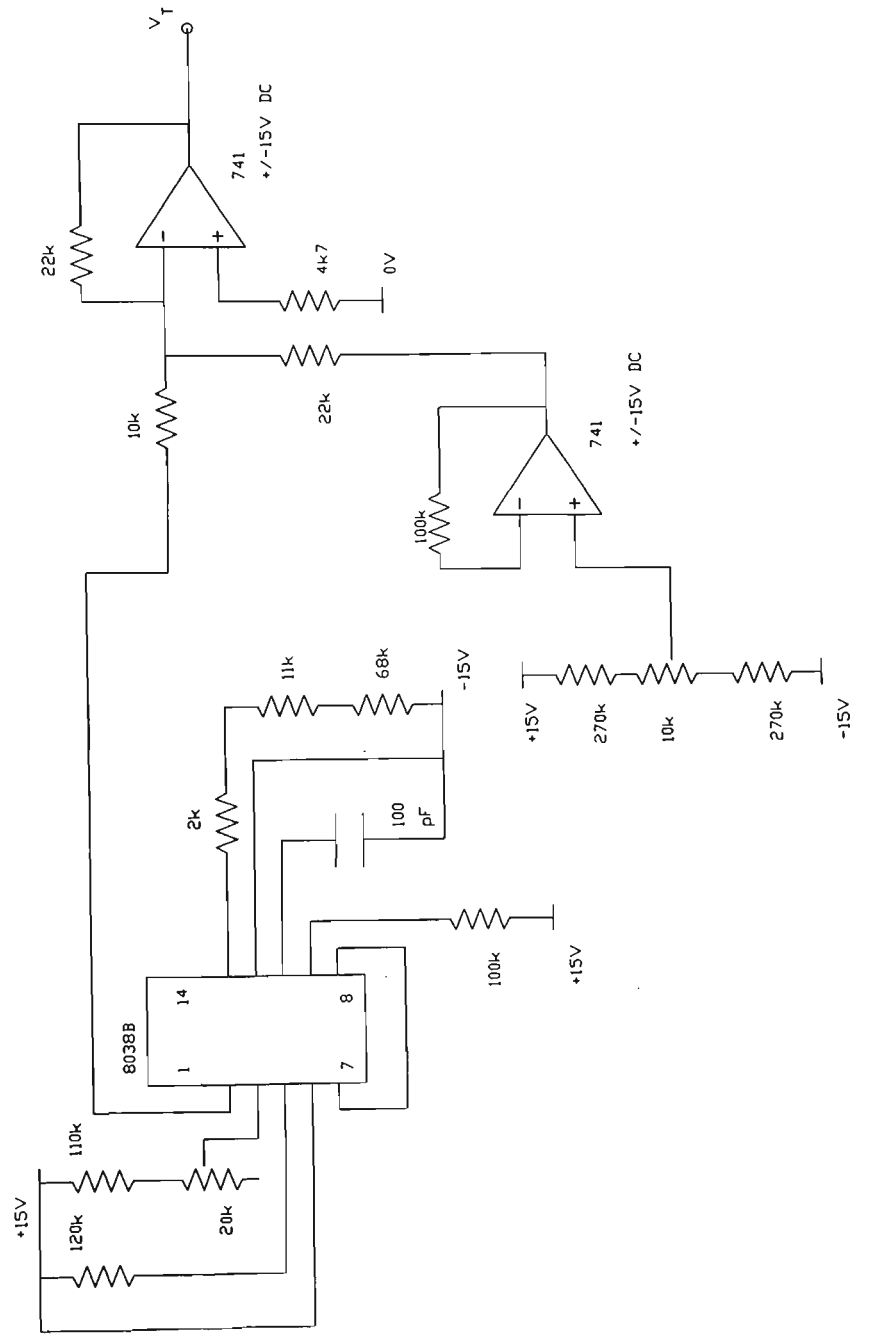

FIGURE B-3 CIRCUIT DIAGRAM OF THE TRIANGLE WAVE GENERATOR OF THE PWM UNIT

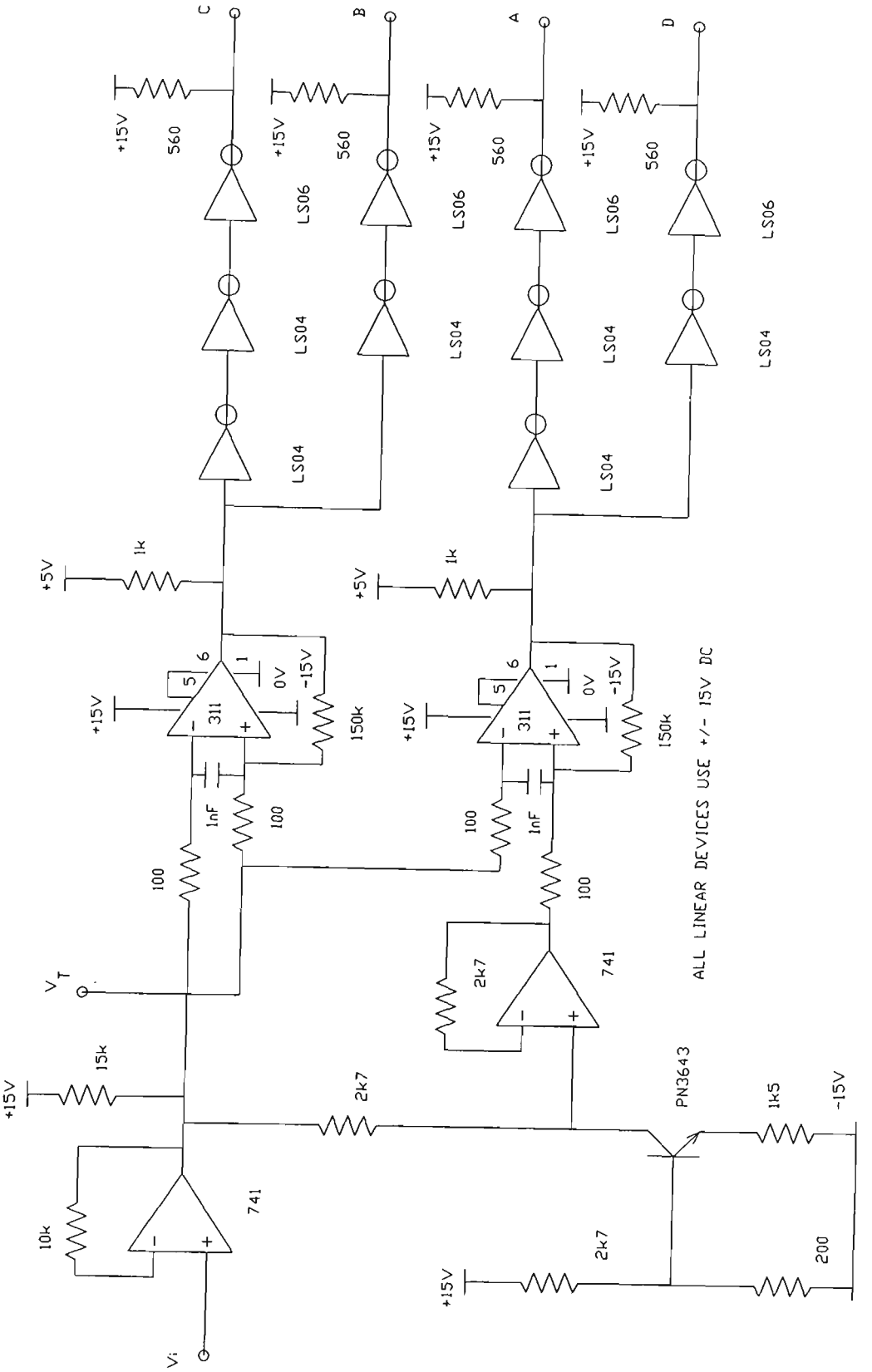

FIGURE B-4 CIRCUIT DIAGRAM OF THE REST OF THE PWM UNIT

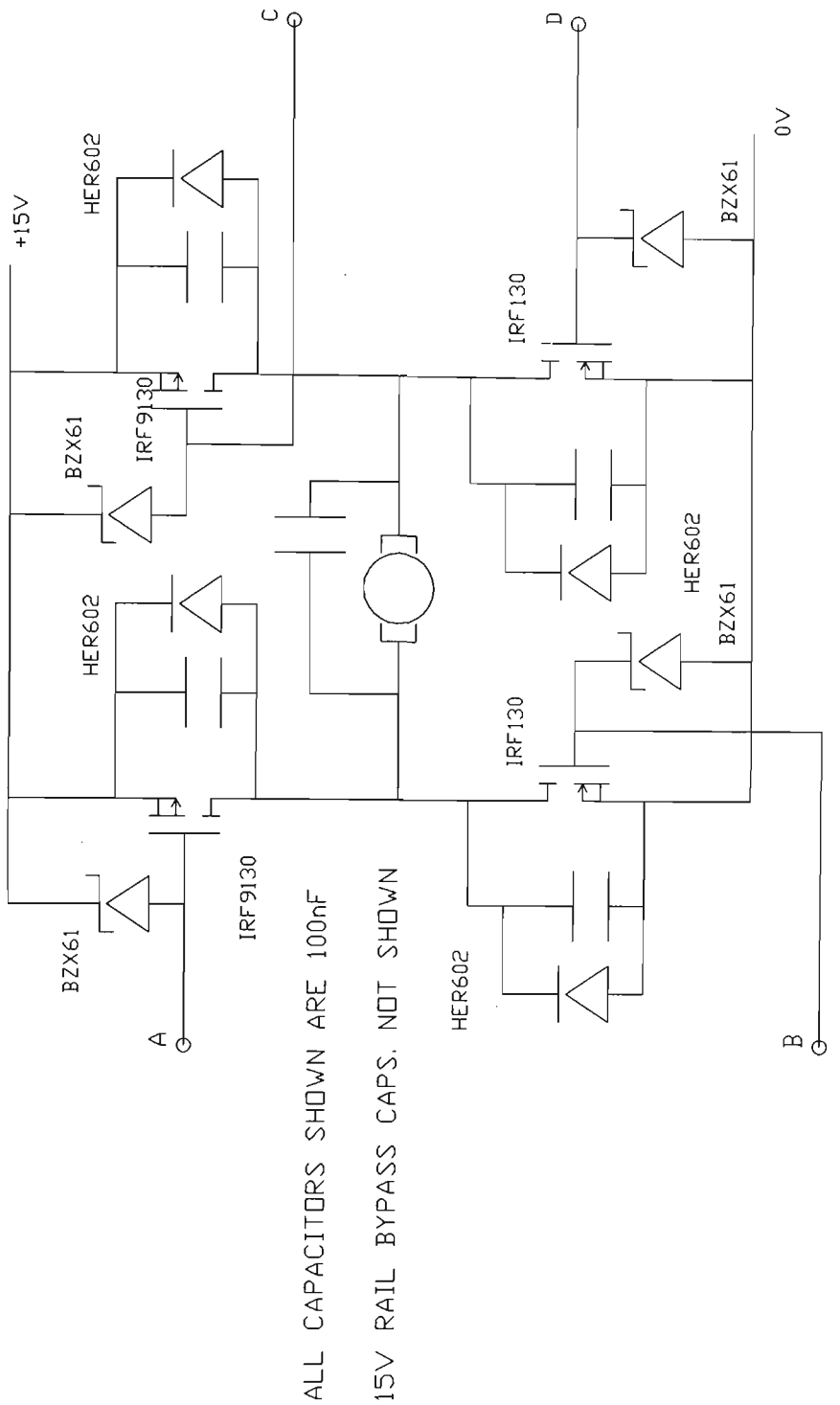

FIGURE B-5 CIRCUIT DIAGRAM OF THE BRIDGE OF POWER VMOS TRANSISTORS

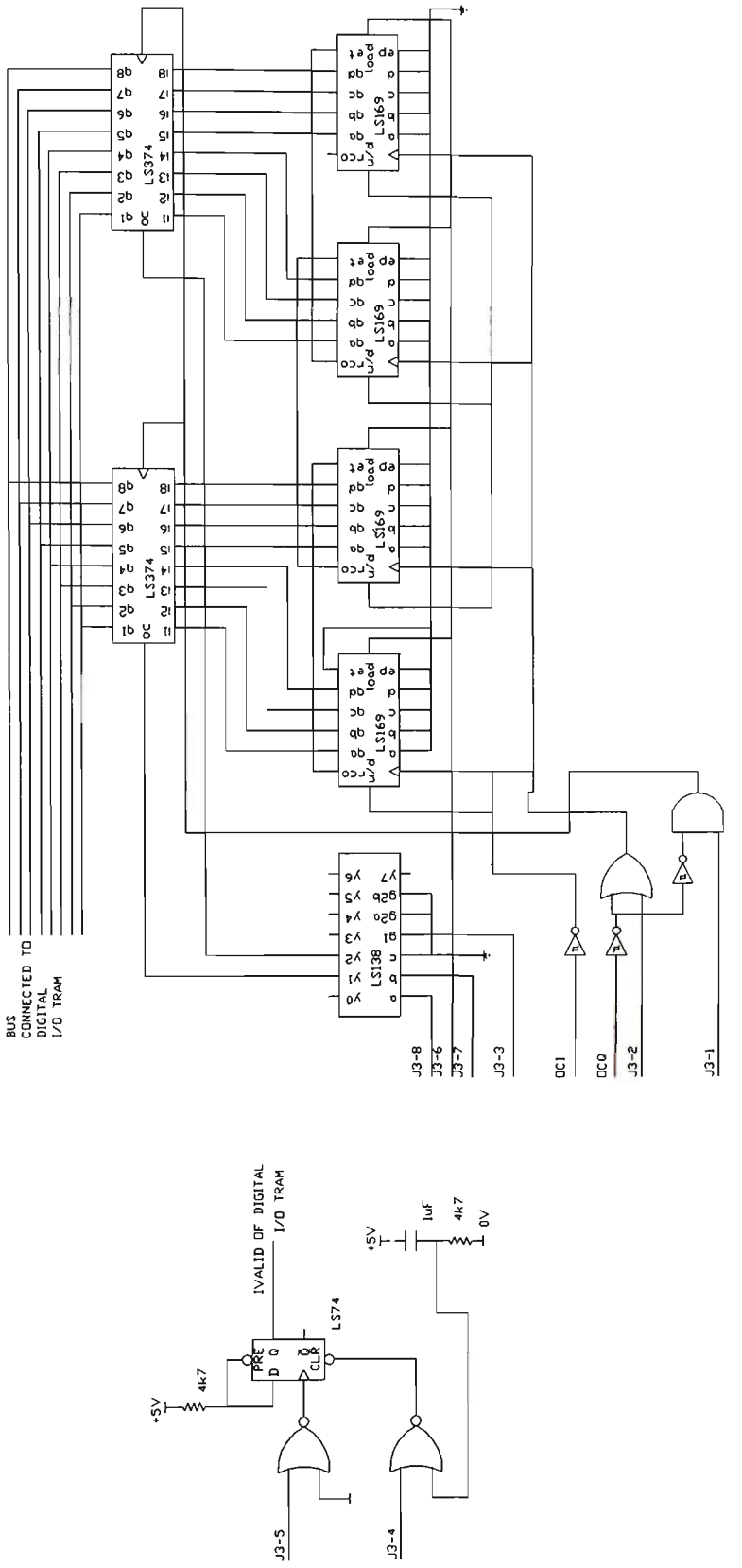

FIGURE B-6 CIRCUIT DIAGRAM OF THE COUNTER CIRCUIT AND ITS CONTROLLER

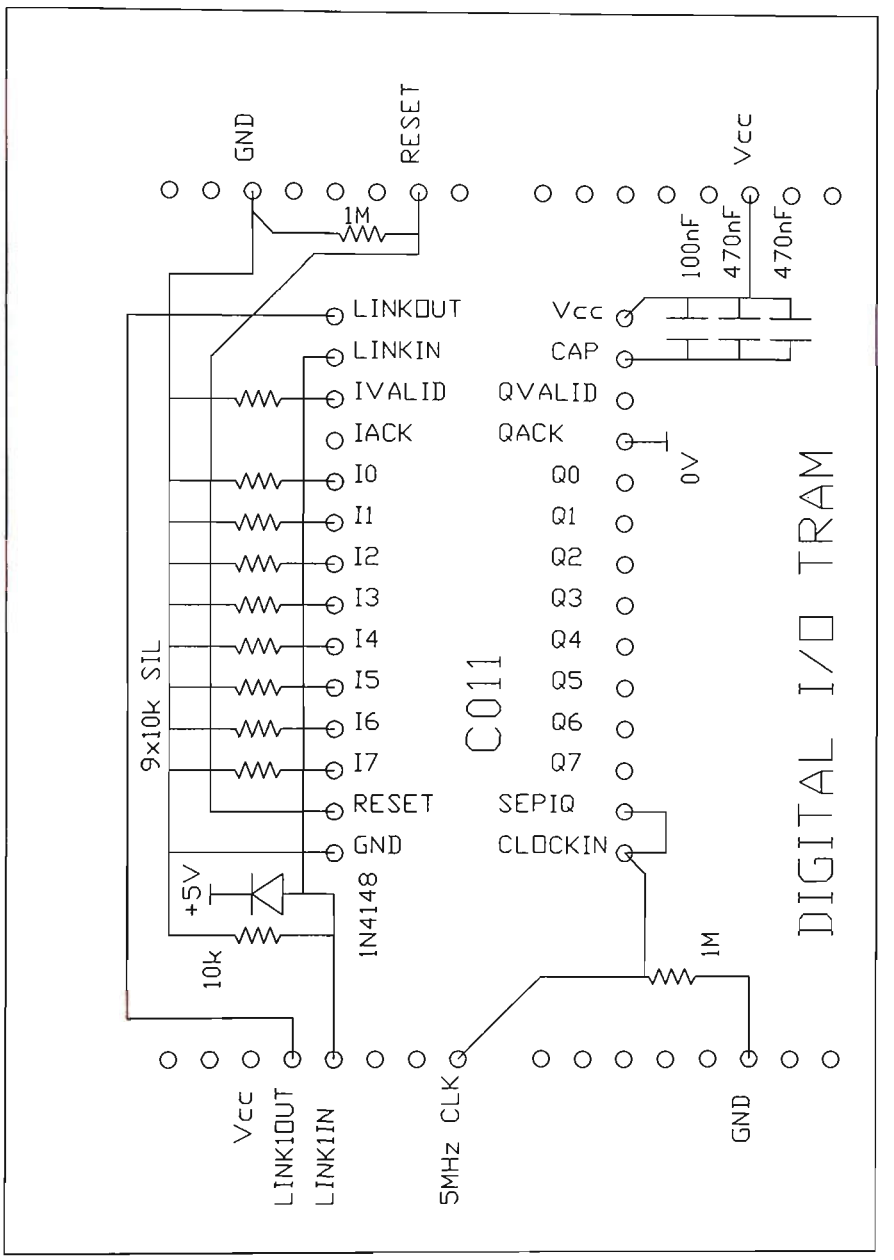

FIGURE B-7 CIRCUIT DIAGRAM AND LAYOUT OF THE TRANSPUTER DIGITAL INTERFACE MODULE

B-5 THE PULSE-WIDTH MODULATOR (PWM) UNIT AND POWER BRIDGE

Figures B-3 and B-4 show the circuitry of the PWM Unit. The Power Bridge circuit is shown in Figure B-5.

The PWM Unit contains a triangle wave generator (Figure B-3), based upon an 8038 waveform generator integrated circuit [B-1]. This circuit produces an ultrasonic (nominal 22kHz) triangle waveform, V_T used as an input to the two comparators of the PWM Unit (Figure B-4).

The other comparator inputs are derived from the output of a DAC, voltage V_i (Figure B-4). A deadband in the switching of the power transistors is provided by a DC Level Shifter circuit. This circuit consists of a $2.7 K\Omega$ resistor, and a current sink.

The outputs of the comparators are used to generate the gate drive signals of the VMOS transistors of the power bridge. The gate drive signals are available at the terminals labelled A,B,C and D of Figure B-4.

The VMOS bridge itself was supplied from a commercial DC Power Supply, with a current limit of up to 5 Amps used to protect both the power bridge and the DC motor. The other components of the bridge circuit (Figure B-5) are diodes required to protect the power transistors, and capacitors included to reduce interference problems.

B-6 THE INTERFACE CIRCUITRY

THE TRANSPUTER DIGITAL INTERFACE MODULE

Figure B-7 shows the circuit and layout of a digital input module, designed and built to provide an interface for reading from the 16-bit position counter. The module uses the C011 Link Adaptor integrated circuit [B-3]. A "Size 2" TRAM [B-4] was designed to enable this adaptor to be mounted on the B008 motherboard. The design enables the contents of the 16-bit counter to be read as two bytes.

THE DIGITAL-TO-ANALOGUE CONVERTER (DAC)

A simple 8-bit DAC circuit was used (Figure B-2). This circuit enables the transputer to generate the voltage V_i, used as an input to the PWM Unit. This voltage controls the duty-cycle of the PWM Unit.

THE COUNTER CIRCUIT

Figure B-6 shows the 16-bit counter circuit, used to monitor the position of the armature shaft.

The signals "OCI" and "OCQ" are the quadrature square waves generated by the optical encoder mounted on the armature shaft. Signal "OCI" is used to control the 'up-down' connection of the counter. Signal "OCQ" is used to clock the counter circuit.

The inputs prefixed by "J3" are connected to the terminals labelled "J3" of the ADC20 Module in slot 7 of the B008 motherboard [B-5].

The counter circuit design includes a strobed latch to enable the separation of the counting process from that of reading the counter. The reading process does not interrupt the counting process.

THE COUNTER CIRCUIT CONTROLLER

The control of the counter circuit is primarily performed by the software, resulting in a relatively simple hardware design. The source code listing of Appendix G contains sequences of code that generate the necessary bit patterns for:

i) initialisation of the counter

ii) latching the counter contents and reading the contents as two bytes of data.

and iii) performing the necessary handshaking with the C011 Link Adaptor of the Digital Transputer Interface Module.

RESTRICTIONS ON THE COUNTER CIRCUIT USE

The optical encoder, having a resolution of 1000 counts per revolution, and being fitted to the armature shaft, produced a count of 71000 for each revolution of the output shaft. The 16 bit counter used to record the output of the optical encoder had a maximum count value of 65536. It was therefore essential that the program monitoring the counter correctly interpreted the overflows and underflows of the counter.

The program can only be certain of a correct interpretation by sampling the counter at an adequately high rate. The counter cannot be allowed to change its count by more than one half of its total count range between consecutive sampling instants. Thus, for a 16 bit counter, the 'distance' between consecutive position measurements must not exceed 32768, equivalent to 32.768 rotations of the armature shaft. This value determines the lowest permissible sampling rate of the system.

Using the no-load speed of the motor as the maximum possible speed, results in a maximum permissible sampling period of 0.4369 seconds.

The highest sampling frequency is determined by the calculation time of the selected algorithm, plus the time required to acquire the signal samples and write out the next output signal.

The designs described above require 157 microseconds to

i) write out a new value via the DAC

and ii) read two bytes from the position counter.

A Three Parameter Estimator can be designed to estimate the pole location with an execution time of less than 163 microseconds. These figures suggest that a period of at least 320 microseconds must elapse between samples from the position counter for real-time estimation of the unknown pole location.

APPENDIX C NON-RECURSIVE SOLUTION OF THE LEAST SQUARES EQUATIONS

C-1 THE SEQUENCE OF EQUATIONS NECESSARY TO SOLVE A FOUR PARAMETER PROBLEM USING LU FACTORISATION

INTRODUCTION

The use of LU Factorisation for the solution of a set of simultaneous equations was outlined in Section 4-3. The following listing shows the sequence of equations necessary to solve a set of four equations in four unknowns, using this method. The listing is provided as a bridge between Section 4-3 and the Occam2 Source Code listing of Appendix E. The use of such a sequence, rather than matrix manipulation, resulted in reduced computation time.

SEQUENCE OF EQUATIONS FOR LU FACTORISATION SOLUTION - FOUR PARAMETER CASE

$$l_{10}=\frac{r_{10}}{r_{00}} \qquad \text{C-1}$$

$$l_{20}=\frac{r_{20}}{r_{00}} \qquad \text{C-2}$$

$$l_{30}=\frac{r_{30}}{r_{00}} \qquad \text{C-3}$$

$$u_{11}=r_{11}-l_{10}.r_{01} \qquad \text{C-4}$$

$$l_{21}=\frac{r_{21}-l_{20}.r_{01}}{u_{11}} \qquad \text{C-5}$$

$$l_{31}=\frac{r_{31}-l_{30}.r_{01}}{u_{11}} \qquad \text{C-6}$$

$$u_{12}=r_{12}-l_{10}.r_{02} \qquad \text{C-7}$$

$$u_{22}=r_{22}-l_{21}.u_{12}-l_{20}.r_{02} \qquad \text{C-8}$$

$$l_{32}=\frac{r_{32}-l_{31}.u_{12}-l_{30}.r_{02}}{u_{22}} \qquad \text{C-9}$$

$$u_{13}=r_{13}-l_{10}.r_{03} \qquad \text{C-10}$$

$$u_{23}=r_{23}-l_{21}.u_{13}-l_{20}.r_{03} \qquad \text{C-11}$$

$$u_{33}=r_{33}-l_{32}.u_{23}-l_{31}.u_{13}-l_{30}.r_{03} \qquad \text{C-12}$$

$$c_0=r_{04} \qquad \text{C-13}$$

$$c_1=r_{14}-l_{10}.r_{04} \qquad \text{C-14}$$

$$c_2=r_{24}-l_{21}.c_1-l_{20}.r_{04} \qquad \text{C-15}$$

$$c_3=r_{34}-l_{32}.c_2-l_{31}.c_1-l_{30}.r_{04} \qquad \text{C-16}$$

$$\theta_3=\frac{c_3}{u_{33}} \qquad \text{C-17}$$

$$\theta_2 = \frac{c_2 - u_{23}.\theta_3}{u_{22}} \qquad \text{C-18}$$

$$\theta_1 = \frac{c_1 - u_{13}.\theta_3 - u12.\theta_2}{u_{11}} \qquad \text{C-19}$$

$$\theta_0 = \frac{c_0 - r_{03}.\theta_3 - r_{02}.\theta_2 - r_{01}.\theta_1}{r_{00}} \qquad \text{C-20}$$

APPENDIX D PSPICE SIMULATIONS OF THE ARMATURE MODEL

This Appendix contains further information on the PSpice simulations of Section 3-6. The source code is of a simulation with a 90% PWM duty cycle. The pulse voltage sources of this listing model the action of the PWM Unit and its VMOS Bridge.

```
ARMATURE-PWM UNIT STEP RESPONSE
.width out = 80
vx 1 2 pulse(0 15 0 0 0 10 10.2)
vy 4 2 pulse(0 15 9 0 0 0.1 10.2)
ra 1 0 0.747
la 0 3 0.23m
rm 3 4 45.7
c 3 4 46.32m
.tran 1e-3 0.5
.options itl5=0
.probe
.end
```

APPENDIX E OCCAM2 SOURCE CODE OF THE SIMULATION PROGRAMS

The following source code listings in Occam2 are of the programs used to perform the simulations described in this thesis. The programs make extensive use of the library procedures of the "Transputer Development System", a programming environment most commonly referred to as TDS. [A-2]

The following code gives the full program used to compare the Lower-Upper Factorisation and the Recursive Least Squares Methods of solution of a set of four normal equations, in four unknown parameters.

```
-- FOUR PARAMETER SOLUTION BY LU FACTORISATION & RLS
#USE interf
#USE userio
#USE snglmath
#USE dblmath
-- DECLARATIONS OF COMMON VARIABLES
INT t.start.i,t.start.ii,t.end.i,t.end.ii,td.i,td.ii:
INT kbdip,ichar,conv.no,run.no,cc,rr,n,r,c:
INT32 i.ran.s:
INT64 i.ran.d:
REAL32 k,a,sample.period.secs,sq.y.error,sq.y0,sq.y0.n,sq.u0,conv.n,run.n:
REAL32 y.error,y.error.scale,true.a1,true.a2,true.b1,true.b2:
REAL32 u0,u1,u2,y0,y0.n,y1,y1.n,y2,y2.n:
REAL32 h,snr,rms.u0,rms.y0,rms.y0.n,rms.error:
REAL32 rand.s:
REAL64 rand.d:
TIMER clock:
CHAN OF ANY to.file:
[63]BYTE filename:
INT result,name.len:
VAL dos.filename IS "estvals.dat":
--DECLARATIONS OF VARIABLES FOR SOLN. BY LU & RLS METHODS
REAL32 u11,u12,u13,u22,u23,u33,l10,l20,l21,l30,l31,l32,c1,c2,c3:
[4]REAL32 Ti:
[4][5]REAL32 R,R.temp:
[101]REAL32 Ti.sq0,Ti.sq1,Ti.sq2,Ti.sq3:
REAL32 error,D,error.over.D:
[4]REAL32 W,Tii:
[4][4]REAL32 P:
[101]REAL32 Tii.sq0,Tii.sq1,Tii.sq2,Tii.sq3:
SEQ          --START OF PROCESSES
  PRI PAR
```

```
SEQ
  ichar := 0(INT)     --INITIALISATION OF MISC. VARIABLES
  i.ran.s := 1(INT32)
  i.ran.d := 3(INT64)
  k := 0.5(REAL32)
  a := 1.0(REAL32)
  sample.period.secs := 0.2(REAL32)
  sq.y.error := 0.0(REAL32)
  sq.y0 := 0.0(REAL32)
  sq.y0.n := 0.0(REAL32)
  sq.u0 := 0.0(REAL32)
  conv.no := 50(INT)
  run.no := 1000(INT)
  conv.n := REAL32 ROUND (conv.no)
  run.n := REAL32 ROUND (run.no)
  -- INITIALISATION OF VARIABLES REQ. FOR VARIANCE CALCULATION
  cc := #00
  WHILE cc < conv.no
    SEQ
      Ti.sq0[cc] := 0.0(REAL32)
      Ti.sq1[cc] := 0.0(REAL32)
      Ti.sq2[cc] := 0.0(REAL32)
      Ti.sq3[cc] := 0.0(REAL32)
      Tii.sq0[cc] := 0.0(REAL32)
      Tii.sq1[cc] := 0.0(REAL32)
      Tii.sq2[cc] := 0.0(REAL32)
      Tii.sq3[cc] := 0.0(REAL32)
      cc := cc + 1(INT)
  cc := 0(INT)
  newline(screen)    --REQUEST LEVEL OF NOISE REQUIRED
  write.full.string(screen,"Enter the required y.error.scale   ")
  read.real32(keyboard,y.error.scale,ichar)
  -- CALCULATION OF VALUES TO BE SUBSEQUENTLY ESTIMATED
  true.b2 := EXP((-a)*sample.period.secs)
  true.b1 := (-1.0(REAL32)) - true.b2
  true.a1 := (k/(a*a))*(((a*sample.period.secs) - 1.0(REAL32)) + true.b2)
  true.a2 := (k/(a*a))*((1.0(REAL32)-true.b2)-
                                ((a*sample.period.secs)*true.b2))
  -- LOOP OF 1000 RUNS TO EXERCISE BOTH ESTIMATORS
  SEQ
    rr := 0(INT)
    WHILE rr < run.no
      SEQ
        -- INITIALISATION OF SIGNAL VECTOR FOR RUN rr
        u2 := 0.0(REAL32)
        u1 := 0.0(REAL32)
        y2 := 0.0(REAL32)   -- ie noise free outputs req for SNR calc.
        y1 := 0.0(REAL32)
        y1.n := 0.0(REAL32) -- ie ouputs of system,
        y2.n := 0.0(REAL32) --          inc noise generated components
        cc := 0(INT)
```

```
-- INITIALISATION OF VARIABLES FOR LU & RLS SOLUTIONS
SEQ n = 0 FOR 4
  Ti[n] := 0.0(REAL32)    -- vector of parameter estimates, theta
SEQ
  SEQ r = 0 FOR 4
    SEQ c = 0 FOR 5
      R[r][c] := 0.0(REAL32)
-- VARIABLES USED IN RLS SOLUTION
SEQ
  P[0][0] := 10000.0(REAL32)
  P[1][1] := 10000.0(REAL32)
  P[2][2] := 10000.0(REAL32)
  P[3][3] := 10000.0(REAL32)
  P[0][1] := 0.0(REAL32)
  P[0][2] := 0.0(REAL32)
  P[0][3] := 0.0(REAL32)
  P[1][0] := 0.0(REAL32)
  P[1][2] := 0.0(REAL32)
  P[1][3] := 0.0(REAL32)
  P[2][0] := 0.0(REAL32)
  P[2][1] := 0.0(REAL32)
  P[2][3] := 0.0(REAL32)
  P[3][0] := 0.0(REAL32)
  P[3][1] := 0.0(REAL32)
  P[3][2] := 0.0(REAL32)
  SEQ n = 0 FOR 4
    Tii[n] := 0.0(REAL32) -- vector of parameter estimates, theta
WHILE cc < conv.no
  SEQ                -- GENERATE NEXT INPUT SIGNAL
    rand.d,i.ran.d := DRAN(i.ran.d)
    u0 := REAL32 ROUND ((rand.d - 0.5(REAL64))*10.0(REAL64))
    -- GENERATE ADDITIONAL NOISE ON O/P SIGNAL
    y.error := 0.0(REAL32)
    SEQ n = 0 FOR 12
      SEQ
        rand.s,i.ran.s := RAN(i.ran.s)
        y.error := y.error + (rand.s - 0.5(REAL32))
    y.error := y.error * y.error.scale
    y0.n:=(((true.a1*u1)+(true.a2*u2)) - ((true.b1*y1.n)+
                                  (true.b2*y2.n)))+y.error
    y0 :=  ((true.a1*u1)+(true.a2*u2)) - ((true.b1*y1) +
                                        (true.b2*y2))
    -- this noise free o/p required to calculate SNR
    -- CALCULATIONS REQ. FOR DETERMINATION OF RMS VALUES
    sq.y.error := sq.y.error + (y.error*y.error)
    sq.y0 := sq.y0 + (y0*y0)
    sq.y0.n := sq.y0.n + (y0.n*y0.n)
    sq.u0 := sq.u0 + (u0*u0)
    SEQ      -- 4 PARAMETER SOLUTION BY LU FACTORISATION
      clock ? t.start.i
      R[0][0] := R[0][0] + (u1*u1)   --UPDATE MATRIX R
```

```
R[0][1] := R[0][1] + (u1*u2)
R[0][2] := R[0][2] - (u1*y1.n)
R[0][3] := R[0][3] - (u1*y2.n)
R[0][4] := R[0][4] + (u1*y0.n)
R[1][0] := R[0][1]
R[1][1] := R[1][1] + (u2*u2)
R[1][2] := R[1][2] - (u2*y1.n)
R[1][3] := R[1][3] - (u2*y2.n)
R[1][4] := R[1][4] + (u2*y0.n)
R[2][0] := R[0][2]
R[2][1] := R[1][2]
R[2][2] := R[2][2] + (y1.n*y1.n)
R[2][3] := R[2][3] + (y1.n*y2.n)
R[2][4] := R[2][4] - (y1.n*y0.n)
R[3][0] := R[0][3]
R[3][1] := R[1][3]
R[3][2] := R[2][3]
R[3][3] := R[3][3] + (y2.n*y2.n)
R[3][4] := R[3][4] - (y2.n*y0.n)
IF
  cc > #03     -- SOLVE CURRENT MATRIX R
    SEQ
      R.temp := R
      l10 := R.temp[1][0]/R.temp[0][0]
      l20 := R.temp[2][0]/R.temp[0][0]
      l30 := R.temp[3][0]/R.temp[0][0]
      u11 := R.temp[1][1] - (R.temp[0][1]*l10)
      IF
        u11 = 0.0(REAL32)
          SKIP
        TRUE
          SEQ
            l21 := (R.temp[2][1] - (R.temp[0][1]*l20))/u11
            l31 := (R.temp[3][1] - (R.temp[0][1]*l30))/u11
            u12 := R.temp[1][2] - (R.temp[0][2]*l10)
            u22 := R.temp[2][2] - ((u12*l21) + (R.temp[0][2]*l20))
            IF
              u22 = 0.0(REAL32)
                SKIP
              TRUE
                SEQ
                  l32 := (R.temp[3][2] - ((l31*u12) +
                                          (l30*R.temp[0][2])))/u22
                  u13 := R.temp[1][3] - (l10*R.temp[0][3])
                  u23 := R.temp[2][3] - ((l21*u13) +
                                         (l20*R.temp[0][3]))
                  u33 := R.temp[3][3] - ((l32*u23) +
                             ((l31*u13) + (l30*R.temp[0][3])))
                  IF
                    u33 = 0.0(REAL32)
                      SKIP
```

```
                TRUE
                  SEQ
                    c1 := R.temp[1][4] - (l10*R.temp[0][4])
                    c2 := R.temp[2][4] - ((l21*c1) +
                                          (l20*R.temp[0][4]))
                    c3 := R.temp[3][4] - ((l32*c2) +
                          ((l31*c1) + (l30*R.temp[0][4])))
                    Ti[3] := c3/u33
                    Ti[2] := (c2 - (u23*Ti[3]))/u22
                    Ti[1] := (c1 - ((u13*Ti[3]) + (u12*Ti[2])))/u11
                    Ti[0] := (R.temp[0][4]-((R.temp[0][3]*
                                Ti[3])+((R.temp[0][2]*
                                Ti[2])+(R.temp[0][1]*
                                Ti[1]))))/R.temp[0][0]

    TRUE
      SKIP
  clock ? t.end.i
SEQ       -- 4 PARAMETER SOLUTION BY RLS
  clock ? t.start.ii
  -- CALCULATE ERROR IN PREDICTION
  error := (y0.n-((Tii[0]*u1)+(Tii[1]*u2)))+((Tii[2]*y1.n)+
                                    (Tii[3]*y2.n))
  -- UPDATE VECTOR W
  W[0] := ((P[0][0]*u1) + (P[0][1]*u2)) - ((P[0][2]*y1.n) +
                                    (P[0][3]*y2.n))
  W[1] := ((P[1][0]*u1) + (P[1][1]*u2)) - ((P[1][2]*y1.n) +
                                    (P[1][3]*y2.n))
  W[2] := ((P[2][0]*u1) + (P[2][1]*u2)) - ((P[2][2]*y1.n) +
                                    (P[2][3]*y2.n))
  W[3] := ((P[3][0]*u1) + (P[3][1]*u2)) - ((P[3][2]*y1.n) +
                                    (P[3][3]*y2.n))
  -- CALCULATE D
  D := (((W[0]*u1)+(W[1]*u2))+1.0(REAL32))-((W[2]*y1.n)+
                                    (W[3]*y2.n))
  IF
    D = 0.0(REAL32)
      SKIP
    TRUE
      SEQ
        -- CALCULATE ESTIMATES (VECTOR THETA)
        error.over.D := error/D
        Tii[0] := Tii[0] + (W[0]*error.over.D)
        Tii[1] := Tii[1] + (W[1]*error.over.D)
        Tii[2] := Tii[2] + (W[2]*error.over.D)
        Tii[3] := Tii[3] + (W[3]*error.over.D)
        -- UPDATE THE P MATRIX
        P[0][0] := P[0][0] - ((W[0]*W[0])/D)
        P[0][1] := P[0][1] - ((W[0]*W[1])/D)
        P[0][2] := P[0][2] - ((W[0]*W[2])/D)
        P[0][3] := P[0][3] - ((W[0]*W[3])/D)
```

```
                P[1][0] := P[0][1]
                P[1][1] := P[1][1] - ((W[1]*W[1])/D)
                P[1][2] := P[1][2] - ((W[1]*W[2])/D)
                P[1][3] := P[1][3] - ((W[1]*W[3])/D)
                P[2][0] := P[0][2]
                P[2][1] := P[1][2]
                P[2][2] := P[2][2] - ((W[2]*W[2])/D)
                P[2][3] := P[2][3] - ((W[2]*W[3])/D)
                P[3][0] := P[0][3]
                P[3][1] := P[1][3]
                P[3][2] := P[2][3]
                P[3][3] := P[3][3] - ((W[3]*W[3])/D)
            clock ? t.end.ii
          -- UPDATE VARIABLES REQ. FOR CALCULATION OF VARIANCES
          Ti.sq0[cc]:=Ti.sq0[cc]+((Ti[0]-true.a1)*(Ti[0]-true.a1))
          Ti.sq1[cc]:=Ti.sq1[cc]+((Ti[1]-true.a2)*(Ti[1]-true.a2))
          Ti.sq2[cc]:=Ti.sq2[cc]+((Ti[2]-true.b1)*(Ti[2]-true.b1))
          Ti.sq3[cc]:=Ti.sq3[cc]+((Ti[3]-true.b2)*(Ti[3]-true.b2))
          Tii.sq0[cc]:=Tii.sq0[cc]+((Tii[0]-true.a1)*(Tii[0]-true.a1))
          Tii.sq1[cc]:=Tii.sq1[cc]+((Tii[1]-true.a2)*(Tii[1]-true.a2))
          Tii.sq2[cc]:=Tii.sq2[cc]+((Tii[2]-true.b1)*(Tii[2]-true.b1))
          Tii.sq3[cc]:=Tii.sq3[cc]+((Tii[3]-true.b2)*(Tii[3]-true.b2))
          -- UPDATE THE DATA ( REGRESSOR ) VECTOR, PSI
          u2 := u1
          u1 := u0
          y2.n := y1.n
          y2 := y1
          y1.n := y0.n
          y1 := y0
          cc := cc + 1(INT)              --END OF INNER LOOP
      rr := rr + 1(INT)                  --END OF OUTER LOOP
rms.error := sq.y.error/(conv.n*run.n) --CALCULATE RMS VALUES & SNR
rms.error := POWER(rms.error,0.5(REAL32))
rms.y0 := sq.y0/(conv.n*run.n)
rms.y0.n := sq.y0.n/(conv.n*run.n)
rms.u0 := sq.u0/(conv.n*run.n)
rms.y0 := POWER(rms.y0,0.5(REAL32))
rms.y0.n := POWER(rms.y0.n,0.5(REAL32))
rms.u0 := POWER(rms.u0,0.5(REAL32))
snr := 20.0(REAL32) * ( ALOG10(rms.y0/rms.error))
newline(screen)      -- WRITE OUT VALUES FROM RUNS TO SCREEN
write.full.string(screen,"    y.error.scale =  ")
write.real32(screen,y.error.scale,4,4)
newline(screen)
newline(screen)
write.full.string(screen,"    RMS y.error =  ")
write.real32(screen,rms.error,4,4)
write.full.string(screen,"   RMS y0 =  ")
write.real32(screen,rms.y0,4,4)
newline(screen)
write.full.string(screen," RMS u0 =  ")
```

```
write.real32(screen,rms.u0,4,4)
write.full.string(screen,"  SNR  =  ")
write.real32(screen,snr,4,4)
write.full.string(screen," RMS y0.n =  ")
write.real32(screen,rms.y0.n,4,4)
newline(screen)
td.i := t.end.i MINUS t.start.i
td.ii := t.end.ii MINUS t.start.ii
newline(screen)
write.full.string(screen,"Calculation time in usecs for  LU method  = ")
write.int(screen,td.i,8)
newline(screen)
write.full.string(screen,"Calculation time in usecs for RLS method  = ")
write.int(screen,td.ii,8)
newline(screen)
write.full.string(screen,"conv.no =  ")
write.int(screen,conv.no,8)
write.full.string(screen,"        run.no =  ")
write.int(screen,run.no,8)
cc := 0(INT)    --CALCULATE MSE FROM CUMULATIVE VALUES
WHILE cc < conv.no
  SEQ
    h := 0.5(REAL32)
    Ti.sq0[cc] := ((POWER((Ti.sq0[cc]/run.n),h))/true.a1)*100.0(REAL32)
    Ti.sq1[cc] := ((POWER((Ti.sq1[cc]/run.n),h))/true.a2)*100.0(REAL32)
    Ti.sq2[cc] := ((POWER((Ti.sq2[cc]/run.n),h))/true.b1)*100.0(REAL32)
    Ti.sq3[cc] := ((POWER((Ti.sq3[cc]/run.n),h))/true.b2)*100.0(REAL32)
    Ti.sq2[cc] := -(Ti.sq2[cc])       -- note, true.b1 is a -ve number
    Tii.sq0[cc] :=((POWER((Tii.sq0[cc]/run.n),h))/true.a1)*100.0(REAL32)
    Tii.sq1[cc] :=((POWER((Tii.sq1[cc]/run.n),h))/true.a2)*100.0(REAL32)
    Tii.sq2[cc] :=((POWER((Tii.sq2[cc]/run.n),h))/true.b1)*100.0(REAL32)
    Tii.sq3[cc] :=((POWER((Tii.sq3[cc]/run.n),h))/true.b2)*100.0(REAL32)
    Tii.sq2[cc] := -(Tii.sq2[cc])     -- note, true.b1 is a -ve number
    cc := cc + 1(INT)
newline(screen)       -- WRITE TO SCREEN TRUE VALUES & THEIR ESTIMATES
write.full.string(screen,"Expected coeffs        ")
write.real32(screen,true.a1,4,4)
write.full.string(screen,"  ")
write.real32(screen,true.a2,4,4)
write.full.string(screen,"  ")
write.real32(screen,true.b1,4,4)
write.full.string(screen,"  ")
write.real32(screen,true.b2,4,4)
newline(screen)
write.full.string(screen,"Estimates by LU method    ")
write.real32(screen,Ti[0],4,4)
write.full.string(screen,"  ")
write.real32(screen,Ti[1],4,4)
write.full.string(screen,"  ")
write.real32(screen,Ti[2],4,4)
write.full.string(screen,"  ")
```

```
    write.real32(screen,Ti[3],4,4)
    newline(screen)
    write.full.string(screen,"Estimates by RLS method   ")
    write.real32(screen,Tii[0],4,4)
    write.full.string(screen,"   ")
    write.real32(screen,Tii[1],4,4)
    write.full.string(screen,"   ")
    write.real32(screen,Tii[2],4,4)
    write.full.string(screen,"   ")
    write.real32(screen,Tii[3],4,4)
    newline(screen)
    -- FREEZE SCREEN TO ENABLE DUMPING TO PRINTER
    newline(screen)
    write.full.string(screen,"Strike any key to terminate")
    keyboard ? kbdip
  SKIP      -- dummy low priority process
-- OUTPUT RMSEs TO DOS TEXT FILE FOR BOTH METHODS
PAR
  SEQ
    name.len := SIZE dos.filename
    [filename FROM 0 FOR name.len] := dos.filename
    scrstream.to.server(to.file,from.filer,to.filer,name.len,
                                        filename,result)
  SEQ
    cc := 0(INT)
    WHILE cc < conv.no
      SEQ
        write.real32(to.file,Ti.sq0[cc],10,6)
        write.full.string(to.file," ")
        write.real32(to.file,Ti.sq1[cc],10,6)
        write.full.string(to.file," ")
        write.real32(to.file,Ti.sq2[cc],10,6)
        write.full.string(to.file," ")
        write.real32(to.file,Ti.sq3[cc],10,6)
        write.full.string(to.file," ")
        write.real32(to.file,Tii.sq0[cc],10,6)
        write.full.string(to.file," ")
        write.real32(to.file,Tii.sq1[cc],10,6)
        write.full.string(to.file," ")
        write.real32(to.file,Tii.sq2[cc],10,6)
        write.full.string(to.file," ")
        write.real32(to.file,Tii.sq3[cc],10,6)
        write.full.string(to.file," ")
        write.int(to.file,cc,6)
        write.full.string(to.file," ")
        newline(to.file)
        cc := cc + 1(INT)
    write.endstream(to.file)
```

SOURCE CODE OF THE METHOD OF USING A PRIORI INFORMATION TO REDUCE THE NUMBER OF NORMAL EQUATIONS BY ONE

The following code shows an alternative method of parameter estimation appropriate to a second order, type one system. The method utilises the a priori knowledge of a plant pole at s = 0. The variable declarations and initialisation required for this method are not listed. An RLS version of the Three Parameter Estimator is given in the source code listing of Appendix G. The following uses LU Factorisation:

```
SEQ
  clock ? t.start.c
  delta.y1 := y1.c - y2.c
  delta.y0 := y0.n - y1.c
  Rc[0][0] := Rc[0][0] + (u1.c*u1.c)
  Rc[0][1] := Rc[0][1] + (u1.c*u2.c)
  Rc[0][2] := Rc[0][2] + (u1.c*delta.y1)
  Rc[0][3] := Rc[0][3] + (u1.c*delta.y0)
  Rc[1][0] := Rc[0][1]
  Rc[1][1] := Rc[1][1] + (u2.c*u2.c)
  Rc[1][2] := Rc[1][2] + (u2.c*delta.y1)
  Rc[1][3] := Rc[1][3] + (u2.c*delta.y0)
  Rc[2][0] := Rc[0][2]
  Rc[2][1] := Rc[1][2]
  Rc[2][2] := Rc[2][2] + (delta.y1*delta.y1)
  Rc[2][3] := Rc[2][3] + (delta.y1*delta.y0)
  IF
    cc > #02
      SEQ
        R.temp := Rc
        l10 := R.temp[1][0]/R.temp[0][0]
        l20 := R.temp[2][0]/R.temp[0][0]
        u11 := R.temp[1][1] - (R.temp[0][1]*l10)
        IF
          u11 = 0.0(REAL32)
            SKIP
          TRUE
            SEQ
              l21 := (R.temp[2][1] - (R.temp[0][1]*l20))/u11
              u12 := R.temp[1][2] - (R.temp[0][2]*l10)
              u22 := R.temp[2][2] - ((u12*l21) + (R.temp[0][2]*l20))
              IF
                u22 = 0.0(REAL32)
                  SKIP
                TRUE
                  SEQ
```

```
            c1 := R.temp[1][3] - (l10*R.temp[0][3])
            c2 := R.temp[2][3] - ((l20*R.temp[0][3]) + (l21*c1))
            Tc[2] := c2/u22
            Tc[1] := (c1 - (u12*Tc[2]))/u11
            Tc[0] := (R.temp[0][3]-((R.temp[0][1]*Tc[1])+
                        (R.temp[0][2]*Tc[2])))/R.temp[0][0]
  TRUE
    SKIP
u2.c := u1.c
u1.c := u0
y2.c := y1.c
y1.c := y0.n
clock ? t.end.c
```

SOURCE CODE FOR THE EXTENDED LEAST SQUARES METHOD

The following source code is for the solution by the method referred to as ELS or Panuska's Method. The variable declarations and initialisation are not given. Since this method increases the number of parameters to be estimated to six a recursive form of the solution is given:

```
-- 6 PARAMETER RECURSIVE EXTENDED LEAST SQUARES SOLUTION
SEQ
  clock ? t.start.p
  -- CALCULATE PREDICTION ERROR
  error := ((y0.n-((Tp[0]*u1.p)+(Tp[1]*u2.p)))+((Tp[2]*y1.p)+
           (Tp[3]*y2.p)))-((Tp[4]*e1)+(Tp[5]*e2))
  -- UPDATE VECTOR W
  W[0] := (((P[0][0]*u1.p) + (P[0][1]*u2.p)) - ((P[0][2]*y1.p) +
          (P[0][3]*y2.p))) + ((P[0][4]*e1)+(P[0][5]*e2))
  W[1] := (((P[1][0]*u1.p) + (P[1][1]*u2.p)) - ((P[1][2]*y1.p) +
          (P[1][3]*y2.p))) + ((P[1][4]*e1)+(P[1][5]*e2))
  W[2] := (((P[2][0]*u1.p) + (P[2][1]*u2.p)) - ((P[2][2]*y1.p) +
          (P[2][3]*y2.p))) + ((P[2][4]*e1)+(P[2][5]*e2))
  W[3] := (((P[3][0]*u1.p) + (P[3][1]*u2.p)) - ((P[3][2]*y1.p) +
          (P[3][3]*y2.p))) + ((P[3][4]*e1)+(P[3][5]*e2))
  W[4] := (((P[4][0]*u1.p) + (P[4][1]*u2.p)) - ((P[4][2]*y1.p) +
          (P[4][3]*y2.p))) + ((P[4][4]*e1)+(P[4][5]*e2))
  W[5] := (((P[5][0]*u1.p) + (P[5][1]*u2.p)) - ((P[5][2]*y1.p) +
          (P[5][3]*y2.p))) + ((P[5][4]*e1)+(P[5][5]*e2))
  -- CALCULATE D
  D := ((((W[0]*u1.p)+(W[1]*u2.p))+1.0(REAL32))-((W[2]*y1.p)+
          (W[3]*y2.p))) + ((W[4]*e1)+(W[5]*e2))
  IF
    D = 0.0(REAL32)
      SKIP
```

```
  TRUE
    SEQ
      -- CALCULATE ESTIMATES (VECTOR THETA)
      error.over.D := error/D
      Tp[0] := Tp[0] + (W[0]*error.over.D)
      Tp[1] := Tp[1] + (W[1]*error.over.D)
      Tp[2] := Tp[2] + (W[2]*error.over.D)
      Tp[3] := Tp[3] + (W[3]*error.over.D)
      Tp[4] := Tp[4] + (W[4]*error.over.D)
      Tp[5] := Tp[5] + (W[5]*error.over.D)
      -- CALCULATE ERROR IN PREDICTION
      error := ((y0.n-((Tp[0]*u1.p)+(Tp[1]*u2.p)))+((Tp[2]*y1.p)+
               (Tp[3]*y2.p)))-((Tp[4]*e1)+(Tp[5]*e2))
      -- UPDATE THE P MATRIX
      P[0][0] := P[0][0] - ((W[0]*W[0])/D)
      P[0][1] := P[0][1] - ((W[0]*W[1])/D)
      P[0][2] := P[0][2] - ((W[0]*W[2])/D)
      P[0][3] := P[0][3] - ((W[0]*W[3])/D)
      P[0][4] := P[0][4] - ((W[0]*W[4])/D)
      P[0][5] := P[0][5] - ((W[0]*W[5])/D)
      P[1][0] := P[0][1]
      P[1][1] := P[1][1] - ((W[1]*W[1])/D)
      P[1][2] := P[1][2] - ((W[1]*W[2])/D)
      P[1][3] := P[1][3] - ((W[1]*W[3])/D)
      P[1][4] := P[1][4] - ((W[1]*W[4])/D)
      P[1][5] := P[1][5] - ((W[1]*W[5])/D)
      P[2][0] := P[0][2]
      P[2][1] := P[1][2]
      P[2][2] := P[2][2] - ((W[2]*W[2])/D)
      P[2][3] := P[2][3] - ((W[2]*W[3])/D)
      P[2][4] := P[2][4] - ((W[2]*W[4])/D)
      P[2][5] := P[2][5] - ((W[2]*W[5])/D)
      P[3][0] := P[0][3]
      P[3][1] := P[1][3]
      P[3][2] := P[2][3]
      P[3][3] := P[3][3] - ((W[3]*W[3])/D)
      P[3][4] := P[3][4] - ((W[3]*W[4])/D)
      P[3][5] := P[3][5] - ((W[3]*W[5])/D)
      P[4][0] := P[0][4]
      P[4][1] := P[1][4]
      P[4][2] := P[2][4]
      P[4][3] := P[3][4]
      P[4][4] := P[4][4] - ((W[4]*W[4])/D)
      P[4][5] := P[4][5] - ((W[4]*W[5])/D)
      P[5][0] := P[0][5]
      P[5][1] := P[1][5]
      P[5][2] := P[2][5]
      P[5][3] := P[3][5]
      P[5][4] := P[4][5]
      P[5][5] := P[5][5] - ((W[5]*W[5])/D)
u2.p := u1.p
```

```
u1.p := u0
y2.p := y1.p
y1.p := y0.n
e2 := e1
e1 := error
clock ? t.end.p
```

SOURCE CODE FOR THE SOLUTION USING INSTRUMENTAL VARIABLES

The following source code is for a method that uses instrumental variables. The instrumental variables are derived from a delayed output signal sequence.

```
-- INSTRUMENTAL VARIABLE SOLUTION
SEQ
  clock ? t.start.e
  delta.y1.e := y1.e-y2.e
  delta.y0.e := y0.n-y1.e
  -- UPDATE MATRIX Re
  IF
    cc < #10
      delta.y1.hat := delta.y1.e
    TRUE
      delta.y1.hat := delta.d5
  Re[0][0] := Re[0][0] + (u1.e*u1.e)
  Re[0][1] := Re[0][1] + (u1.e*u2.e)
  Re[0][2] := Re[0][2] + (u1.e*delta.y1.e)
  Re[0][3] := Re[0][3] + (u1.e*delta.y0.e)
  Re[1][0] := Re[0][1]
  Re[1][1] := Re[1][1] + (u2.e*u2.e)
  Re[1][2] := Re[1][2] + (u2.e*delta.y1.e)
  Re[1][3] := Re[1][3] + (u2.e*delta.y0.e)
  Re[2][0] := Re[2][0] + (u1.e*delta.y1.hat)
  Re[2][1] := Re[2][1] + (u2.e*delta.y1.hat)
  Re[2][2] := Re[2][2] + (delta.y1.hat*delta.y1.e)
  Re[2][3] := Re[2][3] + (delta.y1.hat*delta.y0.e)
  IF
    cc > #02
      -- SOLVE MATRIX Re
      SEQ
        R.temp := Re
        l10 := R.temp[1][0]/R.temp[0][0]
        l20 := R.temp[2][0]/R.temp[0][0]
        u11 := R.temp[1][1] - (R.temp[0][1]*l10)
        IF
          u11 = 0.0(REAL32)
            SKIP
          TRUE
```

```
        SEQ
          l21 := (R.temp[2][1] - (R.temp[0][1]*l20))/u11
          u12 := R.temp[1][2] - (R.temp[0][2]*l10)
          u22 := R.temp[2][2] - ((u12*l21) + (R.temp[0][2]*l20))
          IF
            u22 = 0.0(REAL32)
              SKIP
            TRUE
              SEQ
                c1 := R.temp[1][3] - (l10*R.temp[0][3])
                c2 := R.temp[2][3] - ((l20*R.temp[0][3]) + (l21*c1))
                Te[2] := c2/u22
                Te[1] := (c1 - (u12*Te[2]))/u11
                Te[0] := (R.temp[0][3]-((R.temp[0][1]*Te[1])+
                                (R.temp[0][2]*Te[2])))/R.temp[0][0]
    TRUE
      SKIP
delta.d5 := delta.d4
delta.d4 := delta.d3
delta.d3 := delta.d2
delta.d2 := delta.d1
delta.d1 := delta.y1.e
u2.e := u1.e
u1.e := u0
y2.e := y1.e
y1.e := y0.n
clock ? t.end.e
```

APPENDIX F THE LEAST SQUARES NOISE MODEL

The following work justifies the use of the "Least Squares Noise Model" to describe the process resulting from the simulation programs of Chapter 5.

Consider a noise free system that may be described by Equation 4-18. The output sequence generated by this system would be such as to obey the following equations :-

$$y_2 = a_1u_3+a_2u_4-b_1y_3-b_2y_4 \qquad \text{F-1}$$

$$y_1 = a_1u_2+a_2u_3-b_1y_2-b_2y_3 \qquad \text{F-2}$$

$$y_0 = a_1u_1+a_2u_2-b_1y_1-b_2y_2 \qquad \text{F-3}$$

Next consider how added output noise would change the above evolution of the output sequence. Consider that the first noise to contaminate the output sequence is v_2, added to the output term y_2. This noise contaminated output term will be denoted as y_{2n}, in which case

$$y_{2n} = y_2 + v_2 \qquad \text{F-4}$$

Alternatively a more general notation would be

$$y_{2n} = y_2 + e_2 \qquad \text{F-5}$$

where e_2 represents the error due to all previous added output noise terms. Since the first noise is v_2 it is clear that in this case

$$e_2 = v_2 \qquad \text{F-6}$$

Similarly the subsequent output value will be contaminated by further additional noise, v_1. This output value, including noise components, is denoted by y_{1n}, such that

$$\begin{aligned} y_{1n} &= a_1u_2+a_2u_3-b_1y_{2n}-b_2y_3+v_1 \\ &= y_1-b_1v_2+v_1 = y_1+e_1 \end{aligned} \qquad \text{F-7}$$

where e_1 denotes the difference between y_{1n} and y_1 due to all past, added output noise terms.

Similarly it can be seen that

$$y_{0n} = a_1u_1+a_2u_2-b_1y_{1n}-b_2y_{2n}+v_0$$

$$= y_0 + e_0 \qquad \text{F-8}$$

where

$$e_0 = -b_1e_1-b_2e_2+v_0 \qquad \text{F-9}$$

This discrete-time, difference equation suggests a noise model described by the transfer function:

$$\frac{E(z)}{V(z)} = \frac{1}{1+b_1z^{-1}+b_2z^{-2}} = \frac{1}{B(z)} \qquad \text{F-10}$$

This is consistent with the "LS-model" of noise given by Isermann [F-1].

The above suggests that the following process-and-noise model is adequate. This model is restricted to the case where the only colouring of the white noise input sequence is due to the passage of this sequence through the regressor vector.

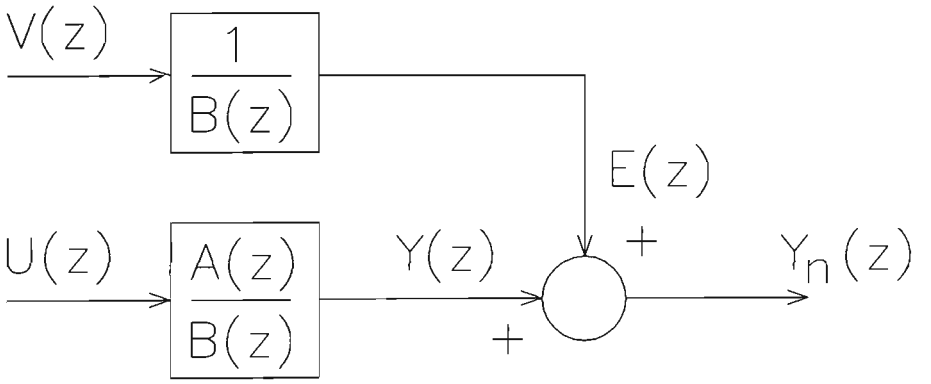

FIGURE F-1 THE LEAST SQUARES PROCESS-AND-NOISE MODEL

APPENDIX G OCCAM2 SOURCE CODE OF THE PROGRAMS USED TO TEST THE REAL SERVOSYSTEM

SOURCE CODE REQUIRED TO PERFORM PARAMETER ESTIMATION OF THE REAL SERVOSYSTEM

The following source code is run on the T800 Root Transputer in Slot 0 of the B008 Motherboard. It is typical of the programs used in this work, this particular program containing an estimator using the Three Parameter RLS Method discussed in Chapter 6.

```
#USE interf                    --TDS LIBRARIES USED
#USE userio
#USE snglmath
BYTE dac.val,ls.byte,ms.byte:  --DECLARATIONS OF VARIABLES
INT16 ls.word,ms.word:
REAL32 dac.volts,dac.volts.high,dac.volts.low:
CHAN OF INT16 to.t212,from.t212:
CHAN OF BYTE to.ad:
CHAN OF BYTE from.ad:
CHAN OF BYTE from.tram2:
PLACE to.t212 AT 1:
PLACE from.t212 AT 5:
PLACE to.ad AT 3:
PLACE from.ad AT 7:
PLACE from.tram2 AT 6:
INT16 t212.reply:
REAL32 pole,error3,D3,error.over.D3:
REAL32 u1,u2,delta.y0,delta.y1:
[3]REAL32 W3,Ti:
[3][3]REAL32 P3:
[4001]REAL32 temp1,temp2,pole.mean,pole.sd,pole.sum,pole.sq.sum:
CHAN OF ANY to.file:
[63]BYTE filename:
INT result,name.len:
VAL dos.filename IS "a:p.dat":
REAL32 run.n,conv.n,sample.period.secs,run.secs,conv.secs:
REAL32 delta.radians:
INT16 new.position,old.position,delta.position:
INT n,r,c,kbdip,ichar,run.no,rr,conv.no,cc,ip.cc:
INT16 position.ls,position.ms:
INT time,t.start,sample.period.int:
INT t.end.calc,t.end.pos.read,t.end,td1,td2,td3:
BYTE tram2.byte:
TIMER clock:
```

```
SEQ                                  --START OF MAIN PROGRAM
 PRI PAR
  SEQ                                --MAKE THE MOTOR STATIONARY
   to.t212 ! 3(INT16)                --connect to ADC20 in slot 6
   from.t212 ? t212.reply
   IF
    t212.reply = 6(INT16)    -- ie successful connection of adc in slot 6
     SKIP
    TRUE
     SEQ
      newline(screen)
      write.full.string(screen,"UNsuccessful connection of adc in slot 6   ")
   to.ad ! #7F(BYTE)                 -- ie 0 volts o/p via dac for 50% duty cycle
   from.ad ? ls.byte
   from.ad ? ms.byte
   ichar := 0(INT)                   --REQUEST KEYBOARD I/P OF ESTIMATOR PERIOD
   newline(screen)
   write.full.string(screen,"Enter the required sample period   ")
   read.real32(keyboard,sample.period.secs,ichar)
   sample.period.int := INT ROUND ( sample.period.secs/1.0E-6(REAL32))
   run.secs := 4.0(REAL32)  --INITIALISATION OF VARIABLES
   conv.n := run.secs/sample.period.secs
   run.no := 20(INT)
   conv.no := INT ROUND (conv.n)   --number of sample periods/4 sec. run
   run.n := REAL32 ROUND (run.no)
   newline(screen)                   --CALIBRATION OF DAC
   write.full.string(screen,"Type y if you wish to measure DAC output")
   newline(screen)
   write.full.string(screen,"Type n to skip, and use default values")
   keyboard ? kbdip
   IF
    (kbdip = (INT'y')) OR (kbdip = (INT 'Y'))
     SEQ
      newline(screen)
      write.full.string(screen,"Take first reading of dac output")
      to.ad ! #6B(BYTE)      -- ie dac.volts.low
      from.ad ? ls.byte
      from.ad ? ms.byte
      newline(screen)
      write.full.string(screen,"Enter the voltage measured   ")
      read.real32(keyboard,dac.volts.low,ichar)
      newline(screen)
      write.full.string(screen,"Take second reading of dac output")
      to.ad ! #00(BYTE)      -- ie dac.volts.high
      from.ad ? ls.byte
      from.ad ? ms.byte
      newline(screen)
      write.full.string(screen,"Enter the voltage measured   ")
      read.real32(keyboard,dac.volts.high,ichar)
      to.ad ! #7F(BYTE)      -- ie 0 volts for 50% duty-cycle
      from.ad ? ls.byte
```

```
      from.ad ? ms.byte
    TRUE                        -- ie accept following default values
      SEQ
        dac.volts.low := 0.73(REAL32)
        dac.volts.high := 4.33(REAL32)
cc := #00                       --INITIALISATION OF CUMULATIVE VARIABLES
WHILE cc < conv.no
  SEQ
    pole.sum[cc] := 0.0(REAL32)
    pole.sq.sum[cc] := 0.0(REAL32)
    pole.mean[cc] := 0.0(REAL32)
    pole.sd[cc] := 0.0(REAL32)
    temp1[cc] := 0.0(REAL32)
    temp2[cc] := 0.0(REAL32)
    cc := cc + 1(INT)
cc := 0(INT)
-- START OF MULTIPLE RUNS OF PARAMETER ESTIMATION
SEQ
  rr := 0(INT)
  WHILE rr < run.no      --counter of outer loop of run number
    SEQ
      SEQ               --INITIALISATION OF RLS ESTIMATOR VARIABLES for run rr
        P3[0][0] := 10000.0(REAL32)
        P3[1][1] := 10000.0(REAL32)
        P3[2][2] := 10000.0(REAL32)
        P3[0][1] := 0.0(REAL32)
        P3[0][2] := 0.0(REAL32)
        P3[1][0] := 0.0(REAL32)
        P3[1][2] := 0.0(REAL32)
        P3[2][0] := 0.0(REAL32)
        P3[2][1] := 0.0(REAL32)
        SEQ n = 0 FOR 3
          Ti[n] := 0.0(REAL32) -- vector of parameter estimates, theta
      pole := 0.0(REAL32)
      delta.y1 := 0.0(REAL32)
      u1 := 0.0(REAL32)
      u2 := 0.0(REAL32)
      -- CODE TO ZERO THE POSITION COUNTER
      to.t212 ! 2(INT16)     -- connect adc20 in slot 7 to access counter cct.
      from.t212 ? t212.reply
      IF
        t212.reply = 5(INT16) -- ie successful connection of adc in slot 7
          SKIP
        TRUE
          SEQ
            newline(screen)
            write.full.string(screen,"UNsuccessful connection of adc in slot 7   ")
      -- prepare to parallel load counter with all zeros
      to.ad ! #9D(BYTE)    -- clear Ivalid,prepare ll'l load, select ls latch
      from.ad ? ls.byte
      from.ad ? ms.byte
```

```
to.t212 ! 3(INT16)      -- connect adc20 in slot 6 to enable dac access
from.t212 ? t212.reply
IF
  t212.reply = 6(INT16) -- ie successful connection of adc in slot 6
    SKIP
  TRUE
    SEQ
      newline(screen)
      write.full.string(screen,"UNsuccessful connection of adc in slot 6   ")
-- run motor to generate pulses for parallel loading of counter cct.
to.ad ! #FF(BYTE)       -- ie +5 volts o/p via dac
from.ad ? ls.byte
from.ad ? ms.byte
clock ? time
clock ? AFTER time PLUS 500000  -- ie 0.5 secs delay
dac.val := #7F(BYTE)
to.ad ! dac.val         -- return dac o/p to 0 volts
from.ad ? ls.byte
from.ad ? ms.byte
clock ? time            -- delay to ensure motor stationary
clock ? AFTER time PLUS 500000  -- ie 0.5 secs delay
to.t212 ! 2(INT16)      -- reconnect adc20 in slot 7 for counter access
from.t212 ? t212.reply
IF
  t212.reply = 5(INT16) -- ie successful connection of adc in slot 7
    SKIP
  TRUE
    SEQ
      newline(screen)
      write.full.string(screen,"UNsuccessful connection of adc in slot 7   ")
-- restore counter circuit
to.ad ! #B5(BYTE)       -- enable Ivalid f/f, inhibit ll'l loading
from.ad ? ls.byte
from.ad ? ms.byte
new.position := #0000(INT16)  -- necessary to initialise old.position
-- READING OF CONTENTS OF POSITION COUNTER
to.ad ! #B4(BYTE)       -- freeze latch contents
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #A4(BYTE)       -- Ivalid goes high
from.tram2 ? tram2.byte
from.ad ? ls.byte
from.ad ? ms.byte
position.ms := (INT16 tram2.byte)
to.ad ! #AC(BYTE)       -- Ivalid goes low
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #7C(BYTE)       -- Prepare to read ls byte
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #74(BYTE)       -- Prepare Ivalid f/f
```

```
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #64(BYTE)     -- Ivalid goes high
from.tram2 ? tram2.byte
from.ad ? ls.byte
from.ad ? ms.byte
position.ls := (INT16 tram2.byte)
position.ls := position.ls/\#00FF(INT16)
position.ms := position.ms<<(8)
old.position := new.position
new.position := INT16 (position.ms + position.ls)
delta.position := new.position MINUS old.position
-- TEST COUNTER FOR UNDER/OVERFLOW
IF
  delta.position > #7FFF(INT16)
    SEQ
      delta.position := #FFFF(INT16) - delta.position
      delta.radians :=-((REAL32 ROUND delta.position)*8.84956E-5(REAL32))
  TRUE
    delta.radians := (REAL32 ROUND delta.position)*8.84956E-5(REAL32)
to.ad ! #AD(BYTE)    -- unfreeze latches,Ivalid low,select ms byte
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #BD(BYTE)    -- prepare f/f clock
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #B5(BYTE)     -- prepare Ivalid f/f
from.ad ? ls.byte
from.ad ? ms.byte
dac.val := #00(BYTE) -- ready for entering loop
cc := 0(INT)
-- CONNECT ADC20 SLOT 6 FOR DAC O/P AT START OF NEXT RUN
to.t212 ! 3(INT16)
from.t212 ? t212.reply
IF
  t212.reply = 6(INT16) -- ie successful connection of adc in slot 6
    SKIP
  TRUE
    SEQ
      newline(screen)
      write.full.string(screen,"UNsuccessful connection of adc in slot 6   ")
-- START OF RUN NUMBER rr OF THE PARAMETER ESTIMATOR
WHILE cc < conv.no  --counter of sample periods within run rr
  SEQ
    clock ? t.start
    -- WRITE OUT VIA THE DAC TO THE PWM UNIT
    to.ad ! dac.val       -- output via dac
    from.ad ? ls.byte
    from.ad ? ms.byte
    -- READ CONTENTS OF POSITION COUNTER
    to.t212 ! 2(INT16)  -- connect adc20 slot 7 for access to counter cct.
```

```
from.t212 ? t212.reply
IF
  t212.reply = 5(INT16) -- ie successful connection of adc in slot 7
    SKIP
  TRUE
    SEQ
      newline(screen)
      write.full.string(screen,"UNsuccessful connection of adc in slot 7   ")
to.ad ! #B4(BYTE)     -- freeze latch contents
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #A4(BYTE)     -- Ivalid goes high
from.tram2 ? tram2.byte
from.ad ? ls.byte
from.ad ? ms.byte
position.ms := (INT16 tram2.byte)
to.ad ! #AC(BYTE)     -- Ivalid goes low
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #7C(BYTE)     -- Prepare to read ls byte
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #74(BYTE)     -- Prepare Ivalid f/f
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #64(BYTE)     -- Ivalid goes high
from.tram2 ? tram2.byte
from.ad ? ls.byte
from.ad ? ms.byte
position.ls := (INT16 tram2.byte)
to.ad ! #AD(BYTE)     -- unfreeze latches,Ivalid low,select ms byte
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #BD(BYTE)     -- prepare f/f clock
from.ad ? ls.byte
from.ad ? ms.byte
to.ad ! #B5(BYTE)     -- prepare Ivalid f/f
from.ad ? ls.byte
from.ad ? ms.byte
clock ? t.end.pos.read  -- end of period taken in signal capture
-- PROCESSING OF SIGNALS
IF
  dac.val = #7F(BYTE)
    dac.volts := 0.0(REAL32)
  dac.val = #00(BYTE)
    dac.volts := dac.volts.high
  dac.val = #6B(BYTE)
    dac.volts := dac.volts.low   --value for monotonic increase in shaft angle
position.ls := position.ls/\#00FF(INT16)
position.ms := position.ms<<(8)
old.position := new.position
```

```
new.position := INT16 (position.ms + position.ls)
delta.position := new.position MINUS old.position
IF
  delta.position > #7FFF(INT16)
    SEQ
      delta.position := #FFFF(INT16) - delta.position
      delta.radians :=-((REAL32 ROUND delta.position)*8.84956E-5(REAL32))
  TRUE
    delta.radians := (REAL32 ROUND delta.position)*8.84956E-5(REAL32)
-- CODE FOR THE 3 PARAMETER RLS ESTIMATOR
SEQ
  delta.y0 := delta.radians
  -- CALCULATE ERROR IN PREDICTION
  error3 := (delta.y0-((Ti[0]*u1)+(u2*Ti[1])))-(Ti[2]*delta.y1)
  -- UPDATE VECTOR W
  W3[0] := (P3[0][0]*u1) + ((P3[0][1]*u2) + (P3[0][2]*delta.y1))
  W3[1] := (P3[1][0]*u1) + ((P3[1][1]*u2) + (P3[1][2]*delta.y1))
  W3[2] := (P3[2][0]*u1) + ((P3[2][1]*u2) + (P3[2][2]*delta.y1))
  -- CALCULATE D
  D3 := (1.0(REAL32)+(W3[0]*u1))+((W3[1]*u2)+(W3[2]*delta.y1))
  IF
    D3 = 0.0(REAL32)
      SKIP
    TRUE
      SEQ
        -- CALCULATE ESTIMATES (VECTOR THETA)
        error.over.D3 := error3/D3
        Ti[0] := Ti[0] + (W3[0]*error.over.D3)
        Ti[1] := Ti[1] + (W3[1]*error.over.D3)
        Ti[2] := Ti[2] + (W3[2]*error.over.D3)
        -- CALCULATE ESTIMATE OF POLE LOCATION
        pole := 0.0(REAL32)
        IF
          Ti[2] > 0.000001(REAL32)    --protection against run time errors
            SEQ
              pole := (-(ALOG(Ti[2])))/sample.period.secs
              pole.sum[cc] := pole.sum[cc] + pole
              pole.sq.sum[cc] := pole.sq.sum[cc] + (pole * pole)
          TRUE
            SKIP
        -- UPDATE THE P MATRIX
        P3[0][0] := P3[0][0] - ((W3[0]*W3[0])/D3)
        P3[0][1] := P3[0][1] - ((W3[0]*W3[1])/D3)
        P3[0][2] := P3[0][2] - ((W3[0]*W3[2])/D3)
        P3[1][0] := P3[0][1]
        P3[1][1] := P3[1][1] - ((W3[1]*W3[1])/D3)
        P3[1][2] := P3[1][2] - ((W3[1]*W3[2])/D3)
        P3[2][0] := P3[0][2]
        P3[2][1] := P3[1][2]
        P3[2][2] := P3[2][2] - ((W3[2]*W3[2])/D3)
        -- UPDATE ESTIMATOR DATA VECTOR
```

```
            u2 := u1
            u1 := dac.volts
            delta.y1 := delta.y0
          -- CALCULATE VALUE OF NEXT DAC OUTPUT
          conv.secs := sample.period.secs*(REAL32 ROUND (cc+1(INT)))
          IF
            conv.secs > 3.0(REAL32)
              dac.val := #6B(BYTE)       -- ie nominal -0.82 volts
            conv.secs > 2.0(REAL32)
              dac.val := #00(BYTE)       -- ie nominal -5 volts
            conv.secs > 1.0(REAL32)
              dac.val := #6B(BYTE)
            TRUE
              dac.val := #00(BYTE)
          -- CONNECT ADC20 IN SLOT 6 FOR DAC ACCESS
          to.t212 ! 3(INT16)
          from.t212 ? t212.reply
          IF
            t212.reply = 6(INT16)     -- ie successful connection of adc in slot 6
              SKIP
            TRUE
              SEQ
                newline(screen)
                write.full.string(screen,"UNsuccessful connection of adc in slot 6   ")
          cc := cc + 1(INT)
          clock ? t.end.calc     -- end of period required for all calculations
          --WAIT UNTIL NEXT ESTIMATOR SAMPLING INSTANT IS DUE
          clock ? AFTER t.start PLUS (sample.period.int - 2(INT))  -- DELAY
          clock ? t.end
      rr := rr + 1(INT)
-- OUTPUT 0 VOLTS VIA DAC TO TIDY UP AFTER LAST RUN
to.ad ! #7F(BYTE)        -- ie 0 volts o/p via dac
from.ad ? ls.byte
from.ad ? ms.byte
-- OUTPUT TIMING INFORMATION TO THE SCREEN
td1 := t.end.calc MINUS t.start
td2 := t.end.pos.read MINUS t.start
td3 := t.end MINUS t.start
newline(screen)
write.full.string(screen,"Time to end of position reading  = ")
write.int(screen,td2,4)
write.full.string(screen," usecs ")
newline(screen)
write.full.string(screen,"Time to end of calculations      = ")
write.int(screen,td1,4)
write.full.string(screen," usecs ")
newline(screen)
write.full.string(screen,"Time to end of cycle             = ")
write.int(screen,td3,4)
write.full.string(screen," usecs ")
newline(screen)
```

```
      write.full.string(screen,"Strike any key to terminate")
      keyboard ? kbdip
    SKIP     -- dummy low priority process
  -- CALCULATE & OUTPUT VALUES TO DOS TEXT FILE
  SEQ
    cc := 0(INT)
    WHILE cc < conv.no
      SEQ
        pole.mean[cc] := pole.sum[cc]/run.n
        pole.sd[cc] := pole.sq.sum[cc]/run.n
        pole.sd[cc] := pole.sd[cc]-(pole.mean[cc]*pole.mean[cc])
        pole.sd[cc] := POWER(pole.sd[cc],0.5(REAL32))
        temp1[cc] := pole.mean[cc] + (2.0(REAL32)*pole.sd[cc])
        temp2[cc] := pole.mean[cc] - (2.0(REAL32)*pole.sd[cc])
        cc := cc + 1(INT)
  PAR
    SEQ
      name.len := SIZE dos.filename
      [filename FROM 0 FOR name.len] := dos.filename
      scrstream.to.server(to.file,from.filer,to.filer,name.len,
                                          filename,result)
    SEQ
      cc := 0(INT)
      WHILE cc < conv.no
        SEQ
          conv.secs := sample.period.secs*(REAL32 ROUND (cc + 1(INT)))
          write.real32(to.file,pole.mean[cc],10,6)
          write.full.string(to.file," ")
          write.real32(to.file,temp1[cc],10,6)
          write.full.string(to.file," ")
          write.real32(to.file,temp2[cc],10,6)
          write.full.string(to.file," ")
          write.real32(to.file,conv.secs,10,6)
          write.full.string(to.file," ")
          write.int(to.file,cc,6)
          newline(to.file)
          cc := cc + 1(INT)
      write.endstream(to.file)
```

SOURCE CODE ENABLING CONTROL OF THE B008 MOTHERBOARD LINK SWITCH

The following source code is for the program loaded onto the T212 transputer used to control the Link switch of the B008 motherboard. The need for dynamic switching of the transputer module links is discussed in Section A-3 of Appendix A.

```
PROC t212.control.c004(CHAN OF INT16 from.t8, to.t8,CHAN OF BYTE to.c004)
  INT16 val.rec:
```

```
  VAL connect.link IS 1(BYTE):
  VAL setup.cmd IS 3(BYTE):
  VAL master.reset IS 4(BYTE):
  VAL slot0.link3 IS 10(BYTE):
  VAL slot7.link3 IS 17(BYTE):
  VAL slot8.link3 IS 18(BYTE):
  VAL slot6.link3 IS 16(BYTE):
  VAL slot1.link3 IS 11(BYTE):
  SEQ
    val.rec := 0(INT16)
    WHILE val.rec <> 9(INT16)
      SEQ
        from.t8 ? val.rec
        IF
          val.rec = 1(INT16)   -- master reset of c004
            SEQ
              to.c004 ! master.reset
              to.t8 ! 4(INT16)
          val.rec = 2(INT16)   -- connect to adc in slot 7
            SEQ
              to.c004 ! master.reset
              to.c004 ! connect.link
              to.c004 ! slot0.link3
              to.c004 ! slot7.link3
              to.c004 ! setup.cmd
              to.t8 ! 5(INT16)
          val.rec = 3(INT16)   -- connect to adc in slot 6
            SEQ
              to.c004 ! master.reset
              to.c004 ! connect.link
              to.c004 ! slot0.link3
              to.c004 ! slot6.link3
              to.c004 ! setup.cmd
              to.t8 ! 6(INT16)
          val.rec = 4(INT16)   -- connect to digital i/o in slot 1
            SEQ
              to.c004 ! master.reset
              to.c004 ! connect.link
              to.c004 ! slot0.link3
              to.c004 ! slot1.link3
              to.c004 ! setup.cmd
              to.t8 ! 9(INT16)
          val.rec = 9(INT16)   -- terminate PROG
            SEQ
              to.c004 ! master.reset
              to.t8 ! 7(INT16)
              STOP
          TRUE
            to.t8 ! 8(INT16)   -- signifies error in operation
:
CHAN OF INT16 from.root, to.root:
```

```
CHAN OF BYTE to.config:
PROCESSOR 0 T2
  PLACE from.root AT 5:
  PLACE to.root AT 1:
  PLACE to.config AT 3:
  t212.control.c004(from.root,to.root,to.config)
```

REFERENCES

1-1 Kalman, RE
Design of a Self-Optimising Control System
Transactions of the American Society of Mechanical Engineers
Vol 80, Feb 1958, pp 468-78

1-2 Astrom, KJ & Wittenmark, B
Adaptive Control
Addison-Wesley, Reading, Massachusetts, 1989, p 11

1-3 Geiger, G
On-line Fault Detection and Localization in Electrical DC Drives based on Process Parameter Estimation and Statistical Decision Methods
Control in Power Electronics and Electrical Drives: Proceedings of the Third IFAC Symposium, Lausanne, Switzerland, 12-14 September 1983. Editor R. Zwicky, Pergamon Press. pp 475-83

1-4 Tylee, JL
On-line Failure Detection in Nuclear Power Plant Instrumentation
IEEE Transactions on Automatic Control
Vol AC-28, March 1983, pp 406-15

1-5 Dailly, C
Fault Monitoring and Diagnosis
Computing and Control Engineering Journal
Vol 1, No 2, March 1990, pp 57-62

1-6 Willsky, AS
A Survey of Design Methods for Failure Detection in Dynamic Systems
Automatica
Vol 12, 1976, pp 601-11

1-7 Eykhoff, P
System Identification: Parameter and State Estimation
John Wiley, Chichester, 1979, pp 197-205

1-8 The Transputer Databook
Inmos
First Edition, Nov 1988

1-9 Astrom, KJ & Wittenmark, B
Computer Controlled Systems: Theory and Design
Prentice-Hall, 1984, Chapter 13

2-1 Wittenmark, B
A Self-Tuning Predictor
IEEE Transactions on Automatic Control
Vol AC-19, No 6, Dec 1974, pp 848-51

2-2 Clarke, DW & Gawthrop, PJ
Self-Tuning Controller
Proceedings of the Institution of Electrical Engineers
Vol 122, No 9, Sept 1975, pp 929-34

2-3 Kalman, RE & Koepcke, RW
Optimal Synthesis of Linear Sampling Control
Transactions of the American Society of Mechanical Engineers
Nov 1958, pp 1820-6

2-4 Sorenson, HW
Least-Squares Estimation: from Gauss to Kalman
IEEE Spectrum
Vol 7, July 1970, pp 63-8

2-5 Plackett, RL
Some Theorems in Least Squares
Biometrika
Vol 37, 1950, pp 149-571

2-6 Stromer, PR
Adaptive or Self-Optimizing Control Systems - A Bibliography
IRE Transactions on Automatic Control
Vol AC-7, May 1959, pp 65-8

2-7 Astrom, KJ
Adaptive Feedback Control
Proceedings of the IEEE
Vol 75, N0 2, Feb 1987, pp 185-217

2-8 Clarke, DW & Hasting-James, R
Design of Digital Controllers for Randomly Disturbed Systems
Proceedings of the Institution of Electrical Engineers: Control & Science
Vol 118, No 10, Oct 1971, pp 1503-6

2-9 Kaminski, PG, Bryson, AE & Schmidt, SF
Discrete Square Root Filtering: A Survey of Current Techniques
IEEE Transactions on Automatic Control
Vol AC-16, No 6, Dec 1971, pp 727-35

2-10 Peterka, V
A Square Root Filter for Real Time Multivariate Regression
Kybernetika
Vol 11, No 1, 1975, pp 53-67

2-11 Bierman, GJ
Factorization Methods for Discrete Sequential Estimation
Academic Press, New York 1977

2-12 Graupe, D, Jain, VK & Salahi, J
A Comparative Analysis of Various Least-Squares Identification Algorithms

Automatica
Vol 16, 1980, pp 663-81

2-13 Astrom, KJ & Wittenmark, B
Adaptive Control
Addison-Wesley Pub. Co., 1989, p 90

2-14 Wong, KY, Bayoumi, MM & Nuyan, S
Comparative Study of some Estimation Algorithms for the Self-Tuning Control of Time-Varying Systems
Control and Computers
Vol 14, No 1, 1986, pp 4-10

2-15 Astrom, KJ & Eykhoff, P
System Identification - A Survey
Automatica
Vol 7, 1971, pp 123-62

2-16 Anderson, BDO
Adaptive Systems, Lack of Persistency of Excitation, and Bursting Phenomena
Automatica
Vol 21, No 3, 1985, pp 247-58

2-17 Swerling, P
First Order Error Propagation in a Stagewise Smoothing Procedure for Satellite Observations
Journal of the Astronautical Sciences
Vol 6, Autumn 1959, pp 46-52

2-18 Fortescue, TR, Kershenbaum, LS & Ydstie, BE
Implementation of Self-Tuning Regulators with Variable Forgetting Factors
Automatica
Vol 17, No 6, 1981, pp831-5

2-19 Morris, AJ, Fenton, TP & Nazer, Y
Application of Self-Tuning Regulators to the Control of Chemical Processes
pp 447-55 of "Digital Computer Applications to Process Control". Editor Van Nauta Lemke. IFAC & North-Holland Publishing Company, 1977

2-20 Astrom, KJ & Wittenmark, B
Self-Tuning Controllers Based on Pole-Zero Placement
IEE Proceedings
Vol 127, Part D, No 3, May 1980, pp 120-30

2-21 Gurubasavaraj, KH
Implementation of a Self-Tuning Controller using Digital Signal Processor Chips
IEEE Control Systems Magazine
June 1989, pp 38-42

2-22 Cordero, AO & Mayne, DQ
Deterministic Convergence of a Self-Tuning Regulator with Variable Forgetting Factor

Proceedings of the IEE
Vol 128, Part D, No 1, Jan 1981, pp 19-23

2-23 Hagglund, T
Recursive Estimation of Slowly Time-Varying Parameters
Identification and System Parameter Estimation 1985: Proceedings of the Seventh IFAC/IFORS Symposium. Editors HA Barker & PC Young, Pergamon Press, 1985 ,pp 1137-42

2-24 Ydstie, BE
Adaptive Control and Estimation with Forgetting Factors
Identification and System Parameter Estimation 1985. Proceedings of the Seventh IFAC/IFORS Symposium in Two Volumes. Editors HA Barker & PC Young, Pergamon Press, 1985, pp 1761-6

2-25 Brickwedde, A
Microprocessor-based Adaptive Control for Electrical Drives
Proceedings of the IFAC Conference on Control in Power Electronics and Electrical Drives, Lausanne, Switzerland, 1983. pp 119-24

2-26 Moore, WE & Smair, S
The Use of Transputers in System Identification
Chapter 9 of "The Transputer in Australasia" .Editors T Bossomaier, T Hintz & J Hulskamp, ATOUG-3, Proceedings of the Third Australian Transputer and Occam User Group Conference, 28-29 June 1990, Sydney, Australia, IOS 1990, pp 69-76

2-27 Goodwin, GC & Teoh, EK
Adaptive Control of a Class of Linear Time Varying Systems
Proceedings of the IFAC Workshop on Adaptive Systems in Control and Signal Processing, San Francisco, USA, 1983, pp 1-6

2-28 Astrom, KJ & Wittenmark, B
Computer Controlled Systems: Theory and Design
Prentice-Hall, 1984

2-29 Goodwin, GC, Middleton, RH & Poor, HV
High-speed Digital Signal Processing and Control
Proceedings of the IEEE
Vol 80, Issue 2, Feb 1992, pp 240-59

3-1 Sinha, NK
Control Systems
CBS College Publishing, 1986, pp 13-5

3-2 Rowland, JR
Linear Control Systems
John Wiley & Sons, 1986, pp 37-9

3-3 Kuo, BC
Automatic Control Systems
Prentice-Hall, 1987, pp 171-6

3-4 Kuo, BC & Tal, J
DC Motors and Control Systems
SRL Publishing Co., Illinois

3-5 ibid p 95

3-6 Puchstein, AF
The Design of Small Direct-Current Motors
John Wiley, New York 1961

3-7 Depping, F & Voits, M
Automatic Selection of Control Algorithms for an Electrical Drive with Microcomputer-Based Speed Control
Control in Power Electronics and Electrical Drives: Proceedings of the Third IFAC Symposium, Lausanne, Switzerland, 12-14 September 1983. Editor R Zwicky, Pergamon Press, Oxford 1983

3-8 Butler, H, Honderd, G & Amerongen, J
Model Reference Adaptive Control of a Direct-Drive DC Motor
IEEE Control Systems Magazine
Jan 1989, pp 80-4

3-9 Dahl, PR
Solid Friction Damping of Space-craft Oscillations
AIAA Guidance and Control Conference, Boston, Mass., Aug 1975, Paper No 75-1104

3-10 Dahl, PR
Solid Friction Damping of Mechanical Vibrations
AIAA Journal
Vol 14, No 12, Dec 1976, pp 1675-82

3-11 Dahl, PR
Measurement of Solid Friction Parameters of Ball Bearings
Proceedings of the Sixth Annual Symposium on Incremental Motion Control Systems and Devices
Electrical Engineering Dept., University of Illinois. May 1977

3-12 King, BG & Martin, LB
Simulation of DC Torque Motor Magnetic Hysteresis and Cogging Effects
Proceedings of the Sixth Annual Symposium on Incremental Motion Control Systems and Devices
Electrical Engineering Dept., University of Illinois. May 1977

3-13 Walrath, CD
Adaptive Bearing Friction Compensation Based on Recent Knowledge of Dynamic Friction
Automatica
Vol 20, No 6, 1984, pp 717-27

3-14 Canudas De Wit, CA
Adaptive Control for Partially Known Systems: Theory and Applications
Studies in Automation and Control 7
Elsevier, Amsterdam, 1988, pp 135 & 141+

3-15 Tuinenga, PW
SPICE. A Guide to Circuit Simulation and Analysis Using PSpice
Prentice-Hall, Englewood Cliffs, 1988

3-16 PSpice Reference Manual
Microsim Corporation
Laguna Hills, California, 1987

3-17 Maxon Motor - Stock and Standard Program with Selection Guide
Interelectric AG, Sachsein, Switzerland, 1989

3-18 Saal, C, Margineanu, I & Ungar, R
Parameter Determination of the DC Drive Motors Using a Digital Automatic Device
Rev. Roum. Sci. Techn. - Electrotechn. et Energ.
Vol 31, No 2, 1986, pp 169-80

3-19 Daniels, AR
The Performance of Electrical Machines
McGraw-Hill, London, 1968, pp 243+

3-20 Bowden, FP & Tabor, D
Friction: An Introduction to Tribology
Heinemann, London 1973

4-1 Phillips, CL & Harbor, RD
Feedback Control Systems
Prentice-Hall International, 1988

4-2 Ogata, K
Discrete-Time Control Systems
Prentice-Hall International, 1987

4-3 Hostetter, GH, Savant, CJ & Stefani, RT
Design of Feedback Control Systems
Holt, Rinehart & Winston Inc., 2nd Edition, 1989

4-4 Mendenhall, W, Scheaffer, RL & Wackerly, DD
Mathematical Statistics with Applications
Duxbury Press, Boston, 3rd Edition, 1986

4-5 Strang, G
Linear Algebra and its Applications
Academic Press, new York, 2nd Edition, 1980

4-6 Young, PC
Applying Parameter Estimation to Dynamic Systems - Part 1
Control Engineering
Oct 1969, pp 119-25

4-7 Astrom, KJ & Wittenmark, B
Adaptive Control
Addison-Wesley Pub. Co., 1989, pp 64-6

5-1 Stecher, M
Linear Algebra
Harper-Row, New York 1988

5-2 Carnahan, B, Luther, HA & Wilkes, JO
Applied Numerical Methods
John Wiley & Sons Inc., New York, 1969

5-3 Mason, JC
Basic Matrix Methods
Butterworths, London, 1984

5-4 Rothenberg, RI
Linear Algebra with Computer Applications
Wiley, 1983

5-5 Hamming, RW
Numerical Methods for Scientists and Engineers
McGraw-Hill, New York, 2nd Edition, 1973

5-6 Steinberg, DI
Computational Matrix Algebra
McGraw-Hill, 1974

5-7 Burns, A
Programming in Occam2
Addison-Wesley, 1988

5-8 Pountain, D & May, D
A Tutorial Introduction to Occam Programming
BSP Professional Books, Oxford, 1987

5-9 Transputer Development System (Reference Manual)
Inmos
Prentice-Hall, New York, 1988

5-10 Kalman, RE
A New Approach to Linear Filtering and Prediction Problems
Transactions of the ASME, Journal of Basic Engineering
Series 82D, March 1960, pp 35-45

5-11 Ljung, L & Yuan, Z-D
Asymptotic Properties of Black-Box Identification of Transfer Functions
IEEE Transactions on Automatic Control
Vol AC-30, No 6, June 1985, pp 514-30

5-12 Devore, JL
Probability and Statistics for Engineering and the Sciences
Brooks/Cole Pub. Co., Monterey, California, 1982

5-13 Astrom, KJ & Wittenmark, B
Adaptive Control
Addison-Wesley, Reading, Massachusetts, 1989, p 421

5-14 Middleton, RH & Goodwin, GC
Digital Control and Estimation : A Unified Approach
Prentice-Hall, 1990, pp 456-61

5-15 Franklin, GF, Powell, JD & Workman, ML
Digital Control of Dynamic Systems
Addison-Wesley, 1990, pp 483-513

5-16 Isermann, R
Parameter Adaptive Control Algorithms - A Tutorial
Automatica
Vol 18, No 5, 1982, pp 513-28

5-17 Lotus 1-2-3 Release 3.1 Reference
Lotus Development Corp., Cambridge, MA, USA 1990

5-18 Press, WH, Flannery, BP, Teukolsky, SA & Vetterling WT
Numerical Recipes in Pascal: The Art of Scientific Computing
Cambridge University Press, Chapter 2, 1989

5-19 Young, PC
Recursive Approaches to Time Series Analysis
The Institute of Mathematics and its Applications
May/June 1974, pp 209-24

6-1 Eykhoff, P
System Identification: Parameter and State Estimation
John Wiley & Sons, Chichester, 1974, p 31

6-2 Bai, E-W & Sastry, SS
Discrete-Time Adaptive Control Utilizing Prior Information
IEEE Transactions on Automatic control
Vol AC-31, No 8, Aug 1986, pp 779-82

6-3 Canudas de Wit, CA
Adaptive Control of Partially Known Systems: Theory and Applications
Elsevier, Amsterdam, 1988

6-4 Wittenmark, B
Adaptive Stability Augmentation
Automatica
Vol 26, No 4, 1990, pp 699-707

6-5 Middleton, RH & Goodwin, GC
Digital Control and Estimation: A Unified Approach
Prentice-Hall, 1990, p 366

6-6 Clary, JP & Franklin, GF
Self-Tuning Control with A Priori Knowledge
Proceedings of the 23rd Conference on Decision and Control, Las Vegas, Nevada, 1984, pp 369-74

6-7 Ives, RV
Least Squares Parameter Estimation of Systems with Integral Action - A Tutorial
Submitted for publication in the 'Journal of Electrical and Electronic Engineering, Australia'. 1994

7-1 Wahlberg, B & Ljung, L
Design Variables for Bias Distribution in Transfer Function Estimation
IEEE Transactions on Automatic Control
Vol AC-31, No 2, Feb 1986, pp 134-44

7-2 Isermann, R
Parameter Adaptive Control Algorithms - A Tutorial
Automatica
Vol 18, No 5, 1982, pp 513-28

7-3 Ljung, L & Soderstrom, T
Theory and Practice of Recursive Identification
MIT Press, Cambridge, Mass., USA, 1986, pp 105-22

7-5 ibid pp 316-20

7-6 ibid pp 103-7

7-7 Isermann, R
Digital Control Systems
Springer Verlag, Berlin, 1981
pp 365-6

7-8 ibid pp 375-6

7-9 Young, PC
The Instrumental Variable Method: A Practical Approach to Identification and System Parameter Estimation
IFAC Identification and System Parameter Estimation 1985, York, UK, pp 1-15

7-10 Kurz, H, Isermann, R & Schumann, R
Experimental Comparison and Application of Various Parameter-Adaptive Control Algorithms
Automatica
Vol 16, 1980, pp 117-33

7-11 Panuska, V
An Adaptive Recursive-Least Squares Identification Algorithm
Proceedings of the 8th IEEE Symposium on Adaptive Control Processes
1969, pp 6-e-1 to 6-e-5

7-12 Singh, MG & Titli, A
Systems: Decomposition, Optimisation and Control
Pergamon Press, Oxford, 1978, pp 491-3

7-13 Kendall, MG & Stuart, A
The Advanced Theory of Statistics
Vol 2. Inference and Relationships
Hafner, New York, 2nd Edition, 1967, p 398

9-1 Press, WH, Teukolsky, SA, Vetterling, WT & Flannery, BP
Numerical Recipes in C
Cambridge University Press, 2nd Edition, 1992
pp 38-41

9-2 ibid pp 45-6

9-3 ibid pp 59+

9-4 ibid pp 96-8

9-5 Bierman, GJ
Factorisation Methods for Discrete Sequential Estimation
Academic Press, 1977
pp 37-43

9-6 ibid p43

A-1 IMS B008 User Guide and Reference Manual
Inmos
Jan 1988

A-2 Transputer Development System (Reference Manual)
Inmos
Prentice-Hall, New York 1988

A-3 ADC20 Analogue to Digital Converter Transputer Module: Installation and Users Manual
Revision Date: 30/5/89, Issue B
Author & Publisher unknown, supplied by "Hawk Electronics Pty. Ltd.", Picnic Point, NSW

A-4 The Transputer Databook
Inmos, 1st Edition, November 1988
pp 20+

A-5 ibid p 71

A-6 IMS B008 User Guide and Reference Manual
Inmos, Jan 1988
p 47

A-7 ibid p 2

A-8 MMS2 User Manual
(Module Motherboard Software User Manual)
Inmos

A-9 User Manual for DT2801 Series
Data Translation Inc.
Marlborough, Mass., USA, 1983

B-1 Exar Databook: Linear, Control, Interface, Custom
Exar Corporation, CA, USA, 1985

B-2 Hewlett Packard
28mm Diameter Two and Three Channel Incremental Optical Encoder Kit. HEDS-5000 Series. Technical Data.
March 1988, Hewlett Packard, Palo Alto, CA., USA

B-3 The Transputer Databook
Inmos
First Edition, November 1988
pp 389-412

B-4 Walker, P
Dual Inline Transputer Modules (TRAMs)
Inmos Technical Note 29
Inmos, UK

B-5 ADC20 Analogue to Digital Converter Transputer Module: Installation and User Manual
Revision Date: 30/5/89, Issue B
Author & Publisher not printed, supplied by "Hawk Electronics Pty. Ltd.", Picnic Point, NSW

F-1 Isermann,R
Digital Control Systems
Springer-Verlag, Berlin, 1981, pp 365-6

ACKNOWLEDGMENTS

I wish to thank my supervisors, Associate Professor Len Herron and Associate Professor Wally Evans, for their support and advice throughout the duration of this work.

In addition I thank Professor Walker and my colleagues in the Department of Electrical and Electronic Engineering for the provision of time and resources required by this exercise. In particular Mr. Alec Simcock and Mr. Wee Sit Lee, for their advice and encouragement, and Dr. Roman Malyniak for his time in reviewing this dissertation.

BIOGRAPHY

Robert Ives graduated from the University of Nottingham, England in 1975. Since June 1982 he has been a member of the academic staff of the Department of Electrical and Electronic Engineering at the Victoria University of Technology (formerly the Footscray Institute of Technology), Australia. Prior to that he was with the British Aerospace Dynamics Group, England.