USING PADE APPROXIMATION IN SYSTEM MODELLING AND SIMULATION WITH STATE-SPACE REPRESENTATION

by

Ulku (KOLAGASIOGLU) CEYLAN

B.S. in Computer and Control Engineering

Istanbul Technical University, 1987

Submitted to the Institute for Graduate Studies in Science and Engineering in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Engineering

Bogazici University

1992

ACKNOWLEDGEMENTS

This thesis benefitted from the advice and support of many people. Here firstly, I want to thank to my first advisor Dr. Serdar Biyiksiz for the research of the subject and to my second advisor Dr. Levent Akın for his valuable effort and contribution for the completion of this thesis. Especially, I want to thank to my advisor Levent Akın for his support for spending time on the programs and many other things.

Also I want greatfully thank to my husband for the patience and support he has shown during the long hours spent at home in preparing the materials. Without his support and understanding I would never have made it. My greatfully thanks are also extended to my Sister, my cousin , my mother and my mother-in-law for their valuable help.

I want also thank to my friends Serap Tutar, Birten Kurnaz, and Aylin Ergen for their help in typing of thesis.

Ulku (KOLAGASIOGLU) CEYLAN

ABSTRACT

Today, in a variety of application the statistical characteristics of a system response is important in order for analysis and model the systems.

In this study, we mainly made an investigation to the system analyzing and modelling methods. Especially, we considered Autoregressive Moving Average (ARMA) and Pade' approximation methods to find the modelled system transfer function coefficients. There are several algorithms to calculate these coefficients. In our study we used Modified Yule Walker Algorithm (MYWE) and AKAIKE algorithms for ARMA and a new Pade' algorithm developed by Biyiksiz for Pade' approximation.

When these three methods were simulated, it was seen that Pade' is mainly less sensitive to the coefficient quantization error and arithmetic round-off error accumulation introduced by finite word length. On the other hand it is not a good approximation for higher orders.It was seen that if the lower orders were used, Pade' approximation gave really good results compared to the MYWE and AKAIKE. But these ARMA models also are not guaranteed to give stable solutions for higher orders. In some cases for higher or lower order ARMA models produced good results especially for higher orders. But these orders should be choosen with one of the methodologies described for model order selection.

An extension of research was done to the state-space error sensitivity. When the mentioned errors were investigated for different representation types of the state-space approach, it was shown that Pade' algorithm was less sensitive to such errors especially for some of the representation types.

ÖZET

Bugün, pek çok uygulamada sistem modellemesi ve analizi için sistem cevabının istatistiksel karakteristiğini bulmak önem taşımaktadır.

Bu çalışmada temel olarak sistem modelleme ve analiz metodlarını araştırdık. Özellikle, Autoregressive Moving Average (ARMA) ve Pade' metodlarını sistem transfer fonksiyonu katsayılarını bulma yönünden inceledik. Transfer fonksiyonu katsayılarını bulmak için literatürde pek çok metod vardır. Bizim çalışmalarımızda ARMA metodunun Modified Yule Walker equations ve AKAIKE algoritmaları, ve Biyiksiz tarafından geliştirlen yeni bir Pade' algoritması incelendi.

Bu üç metod denendiğinde, Pade' algoritmasının bilgisayarda kullanılan sözcük uzunluğundan dolayı ortaya çıkan hatalara karşı daha az duyarlı olduğu görüldü. Öte yandan Pade' algoritması bazen orijinal sisteme göre kötü sonuçlar verdi. Düşük düzeyde üslü terim kullanıldığında Pade' algoritması diğer MYWE ve AKAIKE metodlarından genel olarak daha iyi sonuç verdi. ARMA ve AKAIKE'den ise genellikle yüksek düzeyde üslü terim kullanıldığında daha iyi sonuçlar alınabildi. Fakat bu düzey numaralarının da mutlaka geliştirilen düzey seçme algoritmalarından biri kullanılarak seçilmesi gerektiği gözlemlendi.

Ayrıca durum uzayı modellemelerine karşı her iki algoritmanın bulduğu transfer fonksiyonlarının duyarlılığını inceledik. Araştırmanın bu kısmında ise farklı durum uzayı gerçekleme tiplerini karşılaştırdık. Burada elde edilen sonuçlar Pade'nin hepsinde olmas bile bazı gerçekleme tiplerinde hataya daha az duyarlı bir transfer fonksiyonu üreteceğini gösterdi.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

ACF	:	Autocorrelation Function
AIC	:	Akaike Information Criterion
$a_1 \ldots a_p$	:	Prediction coefficients
AR	:	Autoregressive
AR(p)	:	Autoregressive Process of order p
ARMA	:	Autoregressive Moving Average
ARMA(p,q)	:	Autoregressive Moving Average process ith AR order p and MA order q
$\mathbf{A}(t_n)$	:	NxN dimensional state matrices
A(z)	:	AR filter system function
BT	:	Blackman-Tukey
$\mathbf{B}(t_n)$	:	NxM dimensional state matrices
B(z)	:	MA filter system function
CCF	:	Crosscorrelation Function
$\mathbf{C}(t_n)$	:	RxN dimensional state matrices
c_{xx}	:	Autocovariance function
c_{xy}	:	Cross-covariance function
DFT	:	Discrete Fourier Transform
$\mathbf{D}(t_n)$	:	RxM dimensional state matrices
$\mathcal{E}$	:	Expectation operator
e	:	Prediction error
f	:	Frequency
FFT	:	Fast Fourier Transform
FIR	:	Finite Impulse Response
FPE	:	Final Prediction Error
G(z)	:	System transfer function
H(z)	:	System transfer function
LS	:	Least Squares
LSMYWE	:	Least Squares Modified Yule-Walker Equations
MA	:	Moving Average
MA(q)	:	Moving Average process of order q
MLE	:	Maximum Likelihood Estimator
MSE	:	Mean Square Error

MYWE	:	Modified Yule-Walker Equations
μ_x	:	Mean value of x
PDF	:	Probablity Density Function
PEF	:	Prediction Error Filter
PER	:	Periodogram
PSD	:	Power Spectral Density
P_{xy}	:	Power spectral density (cross-PSD)
P_{xx}	:	Power spectral density (Auto-PSD)
ρ	:	Prediction error power
r_{xx}	:	Autocorrelation function
r_{xy}	:	Cross-correlation function
R_{xx}	:	Autocorrelation matrix
σ^2	:	Excitation white noise variance
SNR	:	Signal to Noise Ratio
u	:	Input white noise noise of time series model
x	:	Output of observed process
$\|\|x\|\|$	:	Norm of a vector
$\langle x,y \rangle$	:	Inner product between the vectorsx and y
$\mathbf{x}(t_n)$	:	N dimensional column state vector
$\mathbf{v}(t_n)$	:	M dimensional column state vector
WSS	:	Wide Sense Stationary
$\mathbf{y}(t_n)$	:	R dimensional column state vector

1. INTRODUCTION

Advances in technology in last few decades have caused a revolution in system design. Many functions are implemented more practically in digital form. So this development has revealed its effect also on the system modelling and identification. The primary concern of this study is the research work towards an alternate methodology for rational linear system modelling and identification.

In many applications, the underlying descriptive signals are inherently continuous-time in nature. If we are to employ the considerable powers of the digital computer for the processing of such signals, it is necessary to convert these signals into a format that is compatible with digital computation. Normally, in many practical situations, the given measurement can change at any instant of time. These signals are called continuous-time signals, to reflect to the dependence of signal on time. On the other hand, there exists an important class of processes in which the relevant signals can change value (or are defined) only at specific instants of time. This is usually done by sampling the input signal at uniformly spaced time intervals. Such a sequence is called a discrete-time signal.

Discrete time signals and their manipulation are inherently well-suited to digital computation and are used in describing the digital portions of a control system. Most often continuous time signals are involved in describing the plant and the interfaces between a controller and the plant its controls. Signals are further classified as being of continuous amplitude or discrete amplitude. Discrete amplitude (or quantized) signals can attain only discrete values, usually evenly spaced. For example, an 8-bit binary code can report

only 256 different values. Because of the complexity of dealing with quantized signals, digital control system design proceeds as if computer-generated signals were not of discrete-character.

Although, the study of continuous and discrete time signals is important and different in its own right, here we are concerned with investigating procedures where by a given signal U is changed (transformed) into another signal X in systematic manner. This information procedure is represented by the mathematical notation

$$X = T\,U \tag{1.1}$$

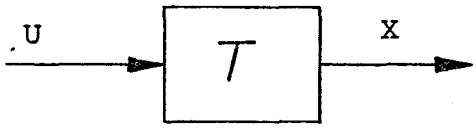


Figure 1.1. System input-output relation

where T represents some well-defined rule by which the signal U is changed into the signal X. In this representation U is interpreted as being the system input signal (or excitation) and X as a system corresponding output signal (or response). Such signals can be represented basically by two methods:

1. Definition of signal by means of a mathematical formula, that is, a closed form expression.
2. Displaying graphically the behaviour of signals.

In many cases, there may not exist a convenient formula by which a given signal can be described. One is then forced to use a graphical display in such situations or to represent the

signal implicity as the solution or output of some relation such as differential equations. Most of the time, it is very difficult to obtain the system response calculation in a closed form.

The Laplace transform method converts time- domain signal descriptions into functions of a complex variable. This complex domain description of a signal provides new insight into the analysis of signals and systems. In addition, The Laplace transform method often simplifies the calculations involved in obtaining system response signals. In working with transfer functions, linear differential equations describing system operations are transformed into algebraic relations, thus eliminating both the necessity of solving the differential equations using classical techniques and the tedium of convolution integration.

The Laplace transform of the continuous-time signal x(t) is

$$X(s) = \int_{-\infty}^{\infty} x(t)\, e^{-st}\, dt \tag{1.2}$$

designated by the symbol X(s) and is formally defined by the integration operation. The variable s that appears in this integrand exponential is generally complex-valued. It is often expressed in terms of its rectangular coordinates.

$$s = \sigma + j\omega \tag{1.3}$$

where σ=Re(s) and w=Im(s) are referred to as the real and imaginary components of s, respectively.

For more information one can refer to [1], [2] and [3] in references.

On the other hand, this type of conversion of control systems is applicable only for continuous time system. The use

of digital controllers revealed another type of research which is discrete-time processing of systems and signals. The z transform method is an important tool for analyzing linear, time-invariant, discrete-time systems. The z-transform plays the same role for discrete-time systems that the Laplace transform plays for continuous-time systems. In fact, the z-transform provides a bridge between continuous-and discrete-time signal processing because the Laplace transform $F^*(s)$ of an ideal impulse sampled signal $f^*(t)$ is related to the z-transform $F(z)$ of the discrete-time signal $f[nT]$ by the transformation $z = e^{sT}$. This transformation maps the left half plane in the complex s-plane into the unit complex z-plane. The interior of the unit circle,the unit circle,and the exterior of the unit circle in the z-plane have similar meaning for discrete-time signals as the left half s-plane, $j\omega$ axis, and right half s-plane for continuous-time signals.

Advancements in digital computer technology revealed the enormous potential of computers, and motivated extensive research to develop sophisticated discrete-time signal processing techniques. As a result of this advancement, the once purely theoretical methods can be applied in practice. In 1958 Blackman and Tukey [4] published classic articles describing how to estimate power spectra from a finite set of signal samples. Techniques were also developed for designing discrete-time filters as they are commonly called, to closely approximate specified frequency responses. In 1965 Cooley and Tukey published an article describing an algorithm, now known as the fast Fourier transform (FFT), for very efficiently computing Fourier Series at a set of uniformly spaced points [5]. The FFT changed the approach to digital power spectrum estimation and significantly reduced the computation time. It also made a frequency-domain approach to digital filtering competitive with the time-domain difference equation approach.

Estimation of the PSD, or simply the spectrum, of discretely sampled deterministic and stochastic processes is usually based on procedures employing the FFT. This approach to

spectral analysis is computationally efficient and produces reasonable results for a large class of signal processes. In spite of these advantages, there are several inherent performance limitations of the FFT approach. The most prominent limitation is that of frequency resolution, i.e., the ability to distinguish the spectral responses of two or more signals. The frequency resolution in Hertz is roughly the reciprocal of the time interval in seconds over which sample data is available. A second limitation is due to the implicit windowing of the data that occurs when processing with the FFT.

These two performance limitations of the FFT approach are particularly troublesome when analyzing short data records. Short data records occur frequently in practice because many measured processes are brief in duration or have slowly time-varying spectra that may be considered constant only for short record lengths. In radar applications, for example, only a few data samples are available from each received radar pulse. In sonar, the motion of targets results in a slowly time-varying spectral response due to Doppler effects.

In an attempt to alleviate the inherent limitations of the FFT approach, many alternative spectral estimation procedures have been proposed within the last decade. The apparent improvement in resolution provided by these techniques have fostered their popularity, even though classical FFT based spectral estimation has been shown to often provide better performance at very low signal-to-noise ratios. Even in those cases where improved spectral fidelity is achieved by use of an alternative spectral estimation procedure, the computational requirements of that alternative method may be significantly higher than the FFT processing required to compute periodogram. This makes some modern spectral estimators unattractive for some real-time implementations.

A summary of modellimg techniques are given in Chapter 2. Parametric modelling AR, MA, ARMA and Pade' methods are explained in Chapter 3. Chapter 4 summarizes the problems when

parametric modelling used. Finally, results are presented in Chapter 5 and conclusion is given in Chapter 6.

2. SPECTRAL ANALYSIS

2.1.SPECTRAL DENSITY BASICS

2.1.1. Random Process Characterization

A discrete random process x(n) is a sequence of random variables, real or complex, defined for every integer n. If the discrete time random process is wide sense stationary (WSS), it has a mean

$$\mathcal{E}[x[n]] = \mu_x \tag{2.1}$$

which does not depend on n and an autocorrelation function (ACF)

$$r_{xx}[k] = \mathcal{E}[x^*[n]\,x[n+k]] \tag{2.2}$$

which depends only on the lag between the two samples, not on their absolute positions. Also, the autocovariance function is defined as

$$C_{xx} = \mathcal{E}\left[(x^*[n] - \mu_x^*)\,(x[n+k] - \mu_x)\right] = r_{xx}[k] - |\mu_x|^2 \tag{2.3}$$

In a similar manner, two jointly WSS random process x[n] and y[n] have a cross-correlation function (CCF)

$$r_{xy} = \mathcal{E}[x^*(n)\,y(n+k)] \tag{2.4}$$

and a cross-covariance function

$$c_{xy}(k) = \mathcal{E}[(x^*(n)-\mu^*)\,(y(n+k)-\mu^y)] = r_{xy} - \mu^*_x\,\mu^y. \tag{2.5}$$

The autocorrelation matrix is defined as

$$\begin{bmatrix} r_{xx}[0] & r_{xx}[-1] & \cdots & r_{xx}[-(M-1)] \\ r_{xx}[1] & r_{xx}[0] & \cdots & r_{xx}[-(M-2)] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[M-1] & r_{xx}[M-2] & \cdots & r_{xx}[0] \end{bmatrix} \tag{2.6}$$

The z-transforms of the ACF and CCF, defined as

$$P_{xx}(z) = \sum_{k=-\infty}^{\infty} r_{xx}[k]\, z^{-k}$$
$$P_{xy}[z] = \sum_{k=-\infty}^{\infty} r_{xy}[k]\, z^{-k} \tag{2.7}$$

lead to the definition of the power spectral density. When evaluated on the unit circle $P_{xx}(z)$ and $P_{xy}(z)$ become auto-PSD, $P_{xx}(f)=P_{xx}(\exp[j2\pi f])$, and cross-PSD, $P_{xy}(f)=P_{xy}(\exp[j2\pi f])$, or

$$P_{xx}(f) = \sum_{k=-\infty}^{\infty} r_{xx}[k] \exp(-2\pi f k)$$
$$P_{xy}(f) = \sum_{k=-\infty}^{\infty} r_{xy}[k] \exp(-2\pi f k) \tag{2.8}$$

The relationship that the auto-PSD is the Fourier transform of the ACF as expressed by Eq.(2.8) is sometimes referred to as the Wiener-Khinchin [6] theorem. The auto-PSD describes the distribution in frequency of the power of x[n] and as such is real and nonnegative. The cross-PSD, on the other hand, is in general complex. The magnitude of the cross-PSD describes whether frequency components in x[n] are associated with large or small amplitudes at the same frequency in y[n], and the phase of the cross-PSD indicates the phase lag or lead of x[n] with the respect to y[n] for q given frequency component. Note that both spectral densities are periodic with period one. The

frequency interval $-\frac{1}{2} \leq f \leq \frac{1}{2}$ will be considered as the fundamental period. When there is no confusion, $P_{xx}(f)$ will be referred to simply as the power spectral density (PSD).

A process that is frequently encountered is discrete white noise. It is defined as having an ACF

$$r_{xx}(k) = \sigma_x^2 \delta(k) \tag{2.9}$$

where $\delta(k)$ is discrete impulse function. This says that each sample is uncorrelated with all the others. Using Eq.(2.8), PSD becomes

$$P_{xx}(f) = \sigma_x^2 \tag{2.10}$$

to be completely flat with frequency. Alternatively, white noise is composed of equipower contributions from all frequencies.

Denoting the system function by H(z)

$$H(z) = \sum_{n=-\infty}^{\infty} h[n] z^{-n} \tag{2.11}$$

the following relations for the PSD's follow

$$\begin{aligned} P_{xy}(z) &= H(z) P_{xx}(z) \\ P_{yx}(z) &= H^*(1/z^*) P_{xx}(z) \\ P_{yy}(z) &= H(z) H^*(1/z^*) P_{xx}(z) . \end{aligned} \tag{2.12}$$

if h[n] is real, $H^*(1/z^*) = H(1/z)$. The last relationship in Eq.(2.12) is particularly important in that it justifies the interpretation of $P_{xx}(f)$ as a PSD. Specifically, the expected power of the output process y[n] is $r_{yy}[0]$.

2.1.2 The Ergodicity Of The Autocorrelation Function

Estimation of the PSD of an arbitrary WSS random process requires one to estimate ACF. A difficulty arises in that the ACF is defined as the expectation of $x^*[n]x[n+k]$ obtained when averaged over an ensemble of realizations. In practice, however, only a segment of a single realization is available. Thus, it is imperative that a single realization of the random process or the infinite data set $x[n]$ for $-\infty<n<\infty$ be sufficient to determine the ACF. A random process which has this property is said to be autocorrelation ergodic. In general, a strictly ergodic process allows one to determine ensemble averages by replacing them with time averages. Hereafter, it will be assumed that the measured process is ergodic in the autocorrelation, so that a time average can replace an ensemble average.

2.2.SPECTRAL ANALYSIS HISTORY

The emergence of spectral estimayion is based on Fourier analysis, which typifies a nonparametric approach. In this approach no specific model is presupposed in formulating the estimation problem. The periodogram defines, in a sense, the frequency contents of a signal over a finite time interval. In general, the periodogram spectral estimate is obtained as the squared magnitude of the values from an DFT performed directly on the wide sense stationary time series observation. This information may, however, be fairly hidden due to the typically erratic behaviour of a periodogram as a function of w.

Traditional spectrum estimation, as currently implemented using the FFT, is characterized by many tradeoffs in an effort to produce statistically reliable spectral estimates. There are tradeoffs in windowing, time-domain

averaging, and frequency-domain averaging of sampled data obtained from random process in order to balance the need to reduce sidelobes, to perform effective ensemble averaging, and to ensure adequate spectral resolution [7], [8]. The spectrum analysis of a random process is in concept not obtained directly from the process x[t] itself, but is based on knowledge of the autocovariance function assuming a zero mean process as it is explained in section 2.1. In practice, one does not usually know the statistical autocovariance function. Thus an additional assumption often made is that the random process is ergodic in the first and second moments. This property permits the substitution of time averages for ensemble averages. For an ergodic process, the statistical autocovariance function may then be equated to

$$r_{xx}(\tau) = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} x(t+\tau)\, x^*(t)\, dt \tag{2.13}$$

with the use of above definitions

$$P(f) = \lim_{T\to\infty} \mathscr{E} \left[\left| \frac{1}{2T} \int_{-T}^{T} x(t) \exp(-j2pift)\, dt \right|^2 \right] . \tag{2.14}$$

The expectation operator is required since the ergodic property of $R_{xx}(\tau)$ does not necessarily imply that the Fourier transform of the process x(t) is also ergodic, this means that the limit in Eq.(2.14) without the expectation operation will not converge in any statistical sense.

Attempting to estimate P(f) with the finite data sets using Eq.(2.14) without taking into consideration the expectation operation and the limit operation, can lead to meaningless spectral estimates if no statistical averaging is performed.

2.3. CLASSICAL METHODS

Spectral estimation techniques based on Fourier transform operations are referred to as classical methods. Here two of them will be mentioned shortly. These are periodogram originally proposed by Schuster, and Blackman-Tukey spectral estimator [4]. The principal conclusion which result from the study of the classical methods is that the bias of the estimator can be reduced if we are willing to accept an increase in variance, and vice versa, but both types of errors can not be reduced simultaneously.

The periodogram definition relies on the PSD definition given by

$$P_{xx}(f) = \lim_{M\to\infty} \mathcal{E}\left[\frac{1}{2M+1}\left|\sum_{n=-M}^{M} x[n]\exp(-j2\pi fn)\right|^2\right] \tag{2.15}$$

By neglecting the expectation operator and using the available data { x[0], x[1],....,x[N-1]} the periodogram spectral estimator is defined as

$$P'_{PER}(f) = \frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]\exp(j2\pi fn)\right|^2 \tag{2.16}$$

It is shown that the periodogram is an inconsistent estimator in that even though the average value converts to the true value as the data record length becomes large, the variance is constant, as given by

$$var[P'_{PER}(f)] \approx P_{xx}^2(f). \tag{2.17}$$

To circumvent this problem the averaged periodogram as defined by

$$P'_{AVPER}(f) = \frac{1}{K}\sum_{m=0}^{K-1} P'^{(m)}_{PER}(f)$$

(2.15)

can be used. For this estimator the data record is segmented into non-overlapping blocks, which is then followed by an averaging of the periodograms for each block. The variance is then reduced by a factor approximately equal to the number of blocks averaged

$$var[P'_{AVPER}(f)] = \frac{1}{K} var[P'^{(m)}_{PER}(f)],$$

(2.19)

but the bias is increased. A compromise must then be made between bias and variance. The confidence interval for the average periodogram is given by

$$10 \log_{10} P'_{AVPER}(f) = \begin{cases} +10 \log_{10} \dfrac{2K}{\chi^2_{2K}(\frac{\alpha}{2})} \\ -10 \log_{10} \dfrac{\chi^2_{2K}(1-\alpha/2)}{2K} \end{cases}$$

(2.20)

The poorer estimates of the ACF at higher lags is a result of a fewer number of lag products averaged. One way to avoid this problem is to weight the ACF estimates at higher lags less or to use the spectral estimator. By using the properties of lag windows spectral estimator can be written as

$$P'_{BT}(f) = \sum_{k=-M}^{M} w(k)\, r'_{xx} \exp(-j2\pi fk).$$

(2.21)

This is called as Blackman-Tukey (BT) spectral estimator. This spectral estimator is sometimes called as weighted covariance estimator. Again a bias-variance trade-off is evident, with the mean being given by

$$\mathcal{E}\,[P'_{BT}(f)] \approx \int_{-1/2}^{1/2} W(f-\xi)\, P_{xx}(\xi)\, d\xi \qquad (2.22)$$

and the variance determined by

$$var\,[P'_{BT}(f)] \approx \frac{P_{xx}^2(f)}{N} \sum_{k=-M}^{M} W^2(k) \qquad (2.23)$$

The weighting of the ACF estimator will reduce the variance of the spectral estimator at the expense of increasing bias (unless the process is white noise for which the bias is zero for any lag window).

The performance of classical spectral estimates at a given frequency f may be characterized by the stability-time-bandwidth product inequality

$$\Delta S \Delta T \Delta J > 1 \qquad (2.24)$$

where ΔT is the time interval over which data has been measured, Δf is the resolution in Hertz, and ΔS is the stability factor, defined as ratio of the spectral estimate variance over the spectral estimate mean. In order to have a stable spectral estimate for a fixed data set of ΔT seconds duration, ΔS must be made small. However, Eq.(2.24) indicates this can only be achieved by giving up resolution (accepting a large value for Δf). Thus, spectral estimation involves a trade-off between statistical stability and resolution.

The conventional Blackman-Tukey and periodogram approaches to spectral estimation have the following advantages:

- Computationally efficient if only a few lags are needed (BT) or if the FFT is used (Periodogram),
- PSD estimate directly proportional to the power for sinusoid process,

- A good model for some applications (The model is a sum of harmonically-related sinusoids)

The disadvantages of the conventional Blackman-Tukey periodogram approach are

- Suppression of weak signal main-loop responses by strong signal sidelobes,
- Frequency resolution limited by the available data record duration, independent of the characteristics of the data or its SNR,
- Introduction of distortion in the spectrum due to sidelobe leakage
- Need for some sort of pseudo ensemble averaging to obtain statistically consistent periodogram spectra,
- The appearance of negative PSD volumes with the BT approach when some autocovariance sequence estimates are used.

For more information on classical modelling methods one can refer to the references [9], [10], and [11].

2.4. PARAMETRIC METHODS

Recent trends in the area of spectral estimation have been towards the development and use of parametric methods. Formulation of the problem in this approach is based on a "model" and the requirement to estimate the unknown parameters of the model given a finite set of the time series observation.

Thus, spectral estimation, in the context of modelling, becomes a three step procedure. The first step is to select a model. The second step is to estimate the parameters of the assumed model using the available data samples. The third step is to obtain the spectral estimate by substituting the estimated model parameters into the theoretical PSD implied by the model. One major motivation for the current interest in the

modelling approach to the spectral estimation is the apparent higher resolution achievable with these modern techniques over that achievable with the classical techniques explained in the previous part. The degree of improvement in resolution and spectral accuracy, if any, will be determined by the ability to fit an assumed model with a small number of parameters. Any inaccuracy in the model will result in a systematic or bias error in the spectral estimate.

The selection of a model and hence a spectral estimate is intimately related to the identification techniques employed in linear systems theory. One key feature of the modelling approach to spectral estimation that differentiates it from the general identification problem is that only the output process of the model is available for analysis; the input driving process is not assumed available as it is for general system identification. This restriction precludes the direct application of the myriad of system identification techniques to spectral estimation. On the other hand, based on the ability to estimate the input process, and both, some system identification techniques have been developed. One of these techniques is Pade' approximation which is the main subject of this thesis. Pade' approximation is also an important approximation method because it can be a solution to the problem of approximating a high order linear system by a lower order model. The exact analysis of most systems of high order is both tedious and costly. It is always desirable to replace such a high-order system by a system of lower order. On this subject there are various methods proposed, but one of the drawbacks of this algorithm is that the reduced order system may be unstable (stable), even if the higher-order is stable (unstable). The details of this approximation method will be given in Chapter 3.

If we choose to represent our model as a ratio of two polynomials, three separate categories can be distinguished. First, an autoregressive (AR), also known as all poles model, which is represented by the inverse of a rational polynomial.

Second, a moving average (MA), also known as an all zeros model, which is represented by a rational polynomial. third an autoregressive moving-average (ARMA), also known as both poles and zeros model, which is represented by a ratio of two rational polynomials. These modelling subjects with Pade' approximation is the main idea researched in this study, so the details of such modelling techniques and their approximation algorithms are given in detail in chapter 3.

2.5. NONPARAMETRIC METHODS

Apart from the above methods, there are some other nonparametric methods used for spectral estimation such as Maximum Likelihood spectral estimation [9], [10], [11], Pisarenko Harmonic decomposition [10], [11] and Music technique [9], [11]. Here, only Maximum Likelihood spectral estimation will be explained shortly.

2.5.1. MAXIMUM LIKELIHOOD SPECTRAL ESTIMATION (MLSE)

The maximum likelihood estimation (MLSE or Minimum Variance Spectral Estimation) falls into the category of a nonparametric technique in the sense that no model parameters are explicitly computed. The original concept was developed by Capon for frequency-wavenumber analysis [12]. A filter model analogy will be used to describe this method. The MLSE was originally developed for seismic array frequency-wavenumber analysis. In this method, one estimates the PSD by effectively measuring the power output of narrow-band filters. MLSE is actually a misnomer in that the spectral estimate is not necessarily a true maximum likelihood estimate of the PSD; it may more appropriately be termed the Capon spectral estimate after its inventor. The name MLSE will be retained here only for historic reasons. The difference between MLSE and

conventional BT/periodogram spectral estimation is that the shape of the narrow-band filters in MLSE are, in general, different for each frequency, where as they are fixed with the BT/periodogram procedures. The filters adapt to the process second order statistics for which a PSD estimate is sought. In response (FIR)types with p weights (taps),

$$A=[a_0 a_1 \ldots . a_{p-1}]^T \tag{2.25}$$

The coefficients are chosen so that at a frequency under consideration f_o the frequency response of filter is unity (i.e. an input sinusoid at that frequency would be undistorted at the filter output) and the variance of the output process is minimized. Thus the filter should adjust itself to reject components of the spectrum not near f_o so that the output power is due mainly to frequency components close to f_o. To obtain the filter, one minimizes the autput variance σ^2 given by Eq. (2.26) subject to the unity frequency response constraint 8 so that the sinusoid of frequency f_o is filtered without distortion). Where R_{xx} is the autocovariance matrix of R_{xx} is the autocovariance matrix of x_n, and E is the vector

$$\sigma^2=A^H R_{xx} A \tag{2.26}$$

$$E^H A=1 \tag{2.27}$$

$$E=[\,1\ \exp(j2\pi f_0 \Delta t) \ldots . \exp(j2\pi [p-1] f_0 \Delta t)\,]^T \tag{2.28}$$

The solution for the filter weights is easily shown to be

$$A_{OPT} = \frac{R_{xx}^{-1} E}{E^H R_{xx}^{-1} E} \tag{2.28}$$

and the minimum output variance is then

$$\sigma^2_{MIN} = \frac{\Delta t}{E^H R_{xx}^{-1} E} \tag{2.30}$$

It is seen that the frequency response of the optimum filter is unity at $f=f_o$ and that the filter characteristics change as a function of the underlying autocovariance function. Since the minimum output variance is due to frequency components near f_o, then $\sigma^2_{MIN}\Delta t$ can be interpreted as PSD estimate. Thus, the MLSE PSD is defined as

$$P_{ML}(f_o) = \frac{1}{E^H R_{xx}^{-1} E} \tag{2.31}$$

To compute the spectral estimate, one only needs an estimate of the autocovariance matrix.

The MLSE and AR PSD have been related analytically as follows. See also reference [11].

$$\frac{1}{P_{ML}(f)} = \frac{1}{p}\sum_{m=1}^{p} \frac{1}{P^{(m)}_{AR}(f)} \tag{2.32}$$

where $P'^{(m)}_{AR}(f)$ is the AR PSD for an mth order model and $P'_{ML}(f)$ is the MLSE PSD, both based upon a known autocovariance of order p.

Also a general tutorial summary of spectrum analysis techniques developed of discrete time series is published by Kay, S. M. and Marple, S. T. in reference [13].

3. PARAMETRIC MODELLING TECHNIQUES

Many discrete-time random processes encountered in practice are well approximated by a time series or rational function model. The output process of this class of models have power spectral densities that are totally described in terms of the model parameters and the variance of the white noise process. The parameters and white noise variance are obtained from the autocorrelation sequence through relationships. In this model an input driving sequence u[n] and the output sequence x[n] that is to model the data are related by the linear difference equation.

If the z domain transfer function of the system is considered, the output function is connected to the input

$$X(z) = H(z)\,U(z) \qquad \frac{X(z)}{U(z)} = H(z)$$

$$H(z) = \frac{a_0 z^0 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_n^{-n}}{b_0 z^0 + b_1 z^{-1} + b_2 z^{-2} + \ldots + b_n z^{-n}} \tag{3.1}$$

with cross multiplication,

$$X(z)\,(b_0 z^0 + b_1 z^{-1} + b_2 z^{-2} + \ldots + b_n z^{-n}) = U(z)\,(a_0 z^0 + a_1 z^{-1} + a_2 z^{-2} + \ldots + a_n z^{-n}) \tag{3.2}$$

by using the shifting operation of z domain, when converted to the discrete time expression

$$b_0 X[k] + b_1 X[k-1] + b_2 X[k-2] + \ldots + b_n X[k-n] = a_0 U[k] + a_1 U[k-1] + a_2 U[k-2] + \ldots + a_n U[k-n] \tag{3.3}$$

Then the output can be found :

$$x[n] = -\sum_{k=1}^{p} a[k]\,x[n-k] + \sum_{k=0}^{q} b[k]\,u[n-k]$$

$$x[n] = \sum_{k=0}^{\infty} h[k]\,u[n-k]$$

(3.4)

A time-series model that approximates many discrete-time deterministic and stochastic processes encountered in practice is represented by the filter linear difference equation of complex coefficients in which x[n] is the output sequence of a causal filter that models the observed data and u[n] is an input driving sequence.This most general linear model is termed as an Auto Regressive Moving Average (ARMA) model and is shown in Eq. (3.4). The assumption b[0]=1 can be made without loss of generality because input u[n] can always be scaled to account for any filter gain.

The system function H(z) between the input u[n] and the output x[n] for the ARMA process of eq. (3.4) is the rational function

$$H(z) = \frac{B(z)}{A(z)}$$

$$\textit{where } A(z) = \textit{z-transform of AR branch} = \sum_{k=0}^{p} a[k]\,z^{-k}$$

$$\textit{where } B(z) = \textit{z-transform of MA branch} = \sum_{k=0}^{q} b[k]\,z^{-k}$$

(3.5)

It is assumed that A(z) has all its zeros within the unit circle of z-plane. This guaranties that H(z) is a stable and causal filter. Without this assumption it can be shown that x[n] as given by Eq.(3.4) would not be a valid description of a WSS (wide sense stationary) random process.

It is well known that the z transform of the ACF at the output of linear filter, $P_{xx}(z)$, is related to that at the input, $P_{uu}(z)$, as follows:

$$P_{xx}(z) = H(z)\,H^*(1/z^*)\,P_{uu}(z) = \frac{B(z)\,B^*(1/z^*)}{A(z)\,A^*(1/z^*)}\,P_{uu}(z) \tag{3.6}$$

The input driving process u[n] is not generally available for purposes of spectral analysis. Many things could be assumed input driving process. It could be a unit impulse, an impulse train, or white noise. Here it will be assumed that the driving sequence is a white noise process of zero mean and variance σ^2, so that $P_{uu}(z)=\sigma^2$.

As it was mentioned above here we consider three types of rational parametric modelling approaches. These are autoregressive (AR), Moving average (MA) and Autoregressive Moving Average approaches.

3.1. AUTOREGRESSIVE PARAMETRIC MODELLING

The autoregressive (AR) spectral estimate has received the most attention in the technical literature of all the time-series models mentioned before. This interest is due to two reasons. First, autoregressive spectra tend to have sharp peaks,a feature often associated with high-resolution spectral estimates. Second, estimates of the AR parameters can be obtained as solutions to linear equations. The AR parameters and the autocorrelation sequence are related by a set of linear equations. Estimates of MA and ARMA parameters, however, require the solution of nonlinear equations.

The underlying assumption of AR process is the availability of the exact autocorrelation function of the random process as it was given by its above definition. In practice, the autocorrelation is usually not available, so one must make an AR spectral estimate based on the available data. There are several algorithmic techniques for producing AR

spectral estimates from data samples. These techniques actually make estimates of AR parameters, and from there the AR PSD function may be evaluated. These techniques are divided into two categories: algorithms for block data and algorithms for sequential data.

The block techniques may be succinctly described as fixed-time, recursive-in-order algorithms in the sense that they operate on a fixed block of time samples and recursively yield higher-order AR order parameter estimates based on lower order AR parameter estimates. This is an advantage in the situations where the appropriate AR model is not known and many different orders must be tried and compared in order to select a suitable order.

Conceptually, the simplest procedure to obtain an AR spectral estimate from data samples would be to produce estimates of autocorrelation sequence from the data using correlation formulas described in the previous chapter. These autocorrelation estimates would then be used in the Yule-Walker equations to yield the AR coefficients and from these the AR PSD function. There are also some other techniques that yield AR model parameters directly from the data without the need for autocorrelation estimates. Such as least mean square v.s. information on these can be found in references and names [9].

The block data techniques considered in this thesis fall into three categories.

The first one is obtained through the equivalent representation of either the autocorrelation sequence or the reflection coefficient sequence. The most common method to calculate the AR parameters that uses this method is Yule-Walker method which is used in this thesis to calculate AR parameter estimates. On the other hand, reflection coefficient sequence estimation is used by another method which is the Burg algorithm given in [6], [9].

An important category of AR parameter estimation method is based on a least squares linear prediction approach. Techniques in this category are further distinguished by the type of linear prediction used. They perform separate maximization of the forward and backward linear prediction squared errors [10], [13], [14].

Another class of techniques perform combined minimization of the forward and backward linear prediction squared errors among which is the modified covariance method [10], [14].

3.1.1 PROPERTIES OF AR PROCESS

3.1.1.1 Linear Prediction of AR Process

As all other modelling methodologies this technique is based on the problem of linear prediction which is to predict the unobserved sample x[n] based on the observed data set (x[n-1], x[n-2], x[n-3],....,x[n-p]). Assuming a predictor that is a linear combination of the past samples,

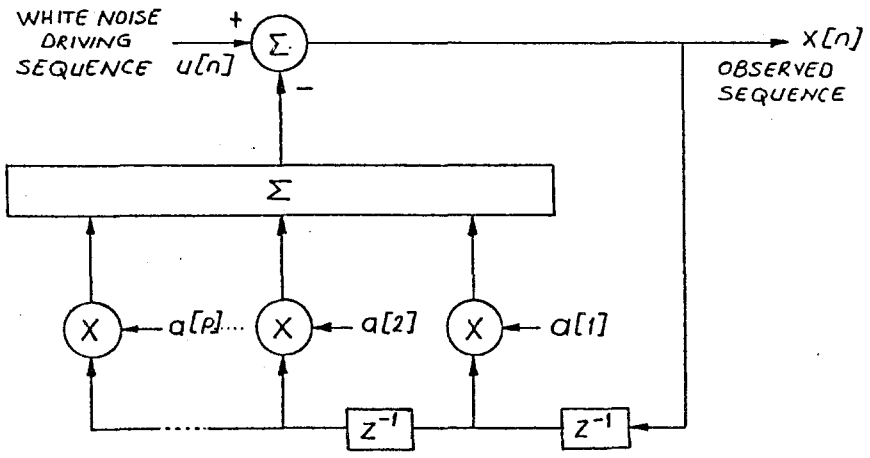


Figure 3.1.1. Autoregressive model of random process

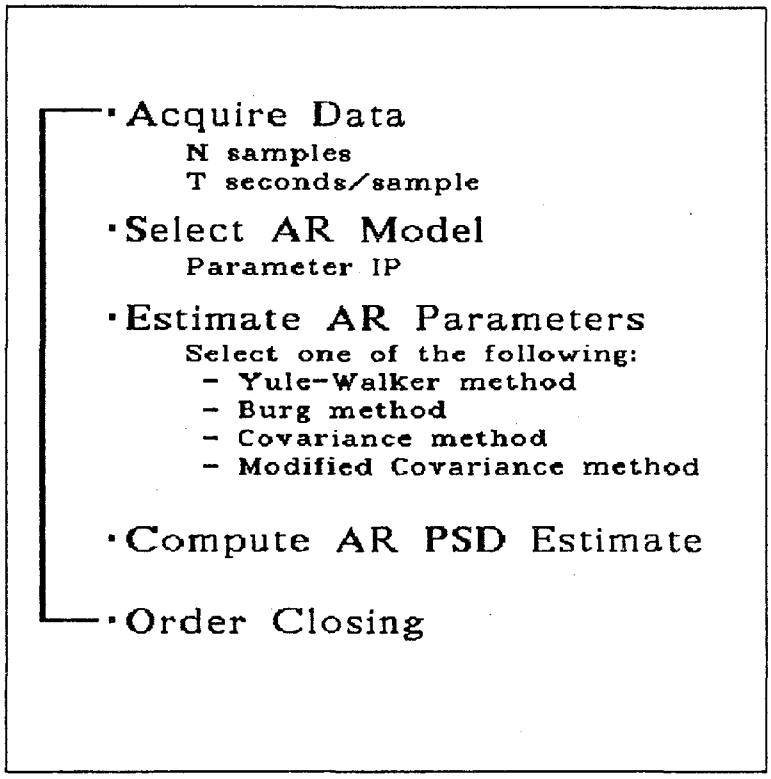


Figure 3.1.2. Summary of AR modelling technique

$$x'[n] = -\sum_{k=1}^{p} \alpha_k x[n-k] \tag{3.7}$$

the prediction coefficients $\{\alpha_1, \alpha_2, \ldots\ldots, \alpha_p\}$ are chosen to minimize the power of the prediction error e[n]:

$$\rho = \mathcal{E}[|e[n]|^2] = \mathcal{E}[|x[n] - \hat{x}[n]|^2] . \tag{3.8}$$

Although x[n] has specifically been chosen to be predicted, the optimal prediction coefficients are independent of the value of n. This is because x[n] is assumed to be WSS, so that the prediction coefficients, which will be a function of the ACF, are independent of n.Processing the ortogonality [6] principle to minimize ρ we have

$$r[k] = -\sum_{l=1}^{q} \alpha_l r_{xx}[k-l]$$

or

$$\mathcal{E}[x^*[n-k]\,[x(n)-x'(n)]] = 0 \qquad k=1,2,\ldots,p \quad (3.1.3.)$$

The minimum prediction error power is found by making use of Eq (3.1.4) to yield

$$\rho_{MIN} = \mathcal{E}\,[x^*(n)\,(x(n)-\hat{x}(n))]$$
$$\rho_{MIN} = r_{xx}(0) + \sum_{k=1}^{p} \alpha_k r_{xx}(-k)\,. \qquad (3.1.4)$$

The optimal linear prediction coefficients are just the AR parameters, and the resulting minimum prediction error power is just the excitation noise variance. This will only be true, however, if the order of the AR process and the order of the linear predictor are identical. The prediction error filter e[n] is

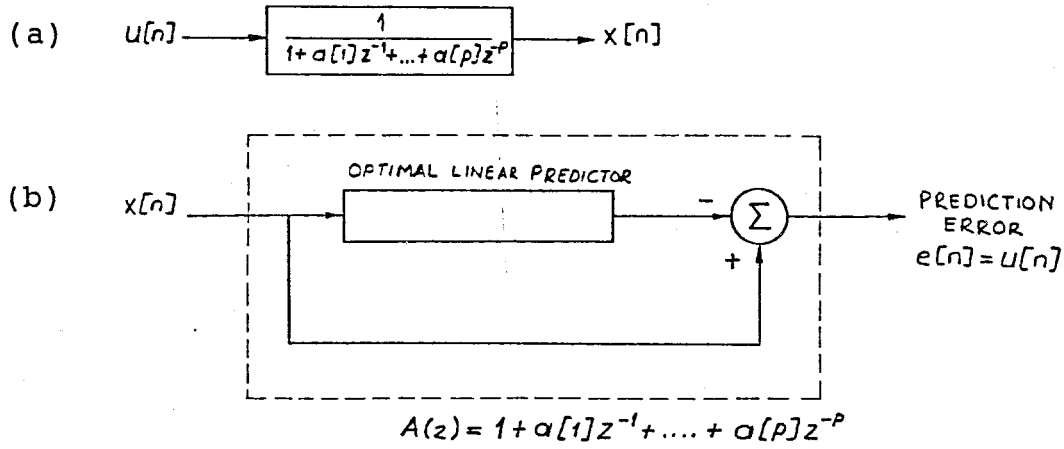


Figure 3.1.3. Filtering interpretation of linear prediction (a) AR model of order p. (b) Prediction error filter

3.1.1.2. Minimum - Phase Property of Prediction Error Filter

In defining the AR process it has been assumed that all the poles of 1/A(z) are inside the unit circle. This condition is necessary to ensure that x[n] is a WSS process. Indeed, if

$$e[n] = x[n] - x'[n] = x[n] - \left[-\sum_{k=1}^{p} \alpha_k x[n-k] \right]$$

$$= x[n] + \sum_{k=1}^{p} a[k]\, x[n-k]$$

$$= u[n] .$$

(3.1.5)

any pole is on or outside the unit circle, the variance of the x[n] will be infinite. On the other hand, if the AR parameters are obtained by solving the Yule-Walker equations, it is not obvious that the poles will be inside the unit circle. That the poles are guaranteed to be inside the unit circle follows from the observation that the optimal pth order linear prediction coefficients are identical to the AR parameters. With the latter results it is now shown that the solution of the Yule-Walker equations yields a stable all-pole filter 1/A(z) or a minimum phase A(z) if the autocorrelation sequence $\{r_{xx}[0], r_{xx}[1], \ldots, r_{xx}[p]\}$ is a valid one. By valid it is meant that the (p+1)x(p+1) autocorrelation matrix

$$R_{xx}^{(p+1)} = \begin{bmatrix} r_{xx}[0] & r_{xx}[-1] & \cdots & r_{xx}[-p] \\ r_{xx}[1] & r_{xx}[0] & \cdots & r_{xx}[-(p-1)] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[p] & r_{xx}[p-1] & \cdots & r_{xx}[0] \end{bmatrix}$$

(3.1.6)

is a positive semi-definite matrix.

Because of the Yule-Walker equations yields the optimal one-step linear predictor for an AR(p) process, the solution minimizes

$$\rho = \mathcal{E}\left[|x[n] - x'[n])|^2 \right]$$

$$\rho = \mathcal{E}\left[\left| \sum_{k=0}^{p} \alpha_k x[n-k] \right|^2 \right]$$

(3.1.7)

where $a_o=1$. The minimum prediction error power ρ_{MIN} can be written as

$$\rho_{MIN} = \int_{-\frac{1}{2}}^{\frac{1}{2}} |A(\exp[j2\pi f])|^2 P_{xx}(f)\, df \qquad (3.1.8)$$

3.1.1.3. The Levinson Algorithm

The solution of the Yule-Walker equations for an AR(p) process was shown to produce the optimal one-step linear prediction coefficients. One can use any standard method to solve the set of linear equations. For instance, Gaussian elimination could be used but would require $O(p^3)$ operations. The Yule-Walker equations, however, are a special set of equations which can be solved in $O(p^2)$ operations by the Levinson algorithm. Although appearing at first to be just an efficient computational algorithm, it in fact reveals fundamental properties of AR process. The concepts of reflection coefficient representations and lattice filters all have their origins in the Levinson algorithm. To make these connections apparent, it becomes necessary to employ a vector space approach to optimal prediction

The Yule-Walker or Wiener-Hopf equations are now rederived using a vector space viewpoint. Let the linear vector space be composed of random variables with zero mean. The inner product is defined as

$$\langle x, y \rangle = \mathcal{E}(x^* y) \qquad (3.1.9)$$

so that the squared norm of a vector is

$$\|x\|^2 = \langle x, x \rangle = \mathcal{E}(|x|^2) = var(x)\,. \qquad (3.1.10)$$

The linear prediction problem is to find the optimal set of coefficients {a[1],a[2],...,a[p]} such that

$$x'[n] = -\sum_{k=1}^{p} a[k]\,x[n-k] \tag{3.1.11}$$

is the "best" predictor of x[n] given {x[n-1],x[n-2],...,x[n-p]}. In anticipation of the result that the linear prediction coefficients are equal to the AR(p) parameters, a[k] has been used to denote the prediction coefficients, the mean square error

$$\rho = \mathcal{E}\left(|x[n]-x'[n]|^2\right) \quad \|x[n]-x'[n]\|^2 \tag{3.1.12}$$

is minimized. By the orthogonality principle, the optimal predictor is found by requiring the error vector x'[n]-x [n] to be orthogonal to the supspace spanned by {x[n-1], x[n-2], ..., x[n-p]} or

$$\langle x[n-k], x[n]-x'[n]\rangle = 0 \qquad k=1,2,\ldots,p \tag{3.1.13}$$

By using Eq(3.1.11) in Eq(3.1.13) and standard properties of inner products, we obtain

$$\langle x[n-k], x[n]+\sum_{l=1}^{p} a[l]\,x[n-l]\rangle = 0 \tag{3.1.14}$$

$$\sum_{l=1}^{p} a[l]\,\langle x[n-k], x[n-l]\rangle = -\langle x[n-k], x[n]\rangle . \tag{3.1.15}$$

Evaluating the inner products as

$$\sum_{l=1}^{p} a[l]\,\mathcal{E}\left(x^*[n-k]\,x[n-l]\right) = -\,\mathcal{E}\left(x^*[n-k]\,x[n]\right) \tag{3.1.16}$$

result in

$$\sum_{l=1}^{p} a[l]\, r_{xx}[k-l] = -r_{xx}[k] \quad k=1,2,\ldots,p. \tag{3.1.17}$$

To find the minimum prediction error power,we begin with

$$\begin{aligned}\rho_{MIN} &= \langle x[n]-\hat{x}[n], x[n]-\hat{x}[n]\rangle \\ &= \langle x[n], x[n]-\hat{x}'[n]\rangle - \langle \hat{x}'[n], x[n]-\hat{x}'[n]\rangle\end{aligned} \tag{3.1.18}$$

But <x'[n], x[n]-x'[n]>=0 from Eq(3.1.13), then simply ρ_{MIN} can be obtained,

$$\rho_{MIN} = \langle x[n], x[n]\rangle + \sum_{k=1}^{p} a[k]\langle x[n], x[n-k]\rangle \tag{3.1.19}$$

$$\rho_{MIN} = r_{xx}[0] + \sum_{k=1}^{p} a[k]\, r_{xx}[-k]\,. \tag{3.1.20}$$

Eq (3.1.18) and Eq (3.1.19) are the Yule-Walker equations. The solution of Eq.(3.1.18) provides the optimal set of coefficients to predict x[n] as a linear combination of {x[n-1],x[n-2],...,x[n-p]} (i.e.,the optimal pth order linear predictor). If we wish to determine not only the pth order linear predictor but also the linear predictors of orders p - 1, p - 2,...,1, one possibility is to solve Eq.(3.1.19) for the various assumed model orders. The result will be sets of prediction coefficients {[a_1[1]}, {a_2[1], a_2[2]} ,..., {a_p[1] ,a_p[2] ,..., a_p[p]}, where a_j[i] is the ith coefficients of the jth order linear predictor. Clearly, a_p[i]=a[i] for i=1,2...,p. This procedure, although straightforward, proves to be computationally burdensome and is altogether unnecessary. An alternative approach is to recursively update the predictor of order k-1 to order k. This requires that we perform a Gram-Schmidt orthogonalization of the data {x[n-1],x[n-2],...x[n-p]} into orthogonal or uncorrelated random variables. To see how this is done, let x'_{k-1}[n] be the optimal (k-1)st order linear predictor of x[n] based on the previous k-1 samples or

$$x_{k-1}[n] = -\sum_{i=1}^{k-1} a_{k-1}[i]\,x[n-i]\,.$$

(3.1.21)

The subscript on x'[n] indicates the number of previous samples used in the prediction. Consider a first order linear predictor so that k-1=1. Then

$$x'_1[n] = -a_1[1]\,x[n-1]$$

(3.1.22)

and $a_1[1]$ is found by minimizing

$$\rho_1 = \|x[n] - x'_1[n]\|^2\,.$$

(3.1.23)

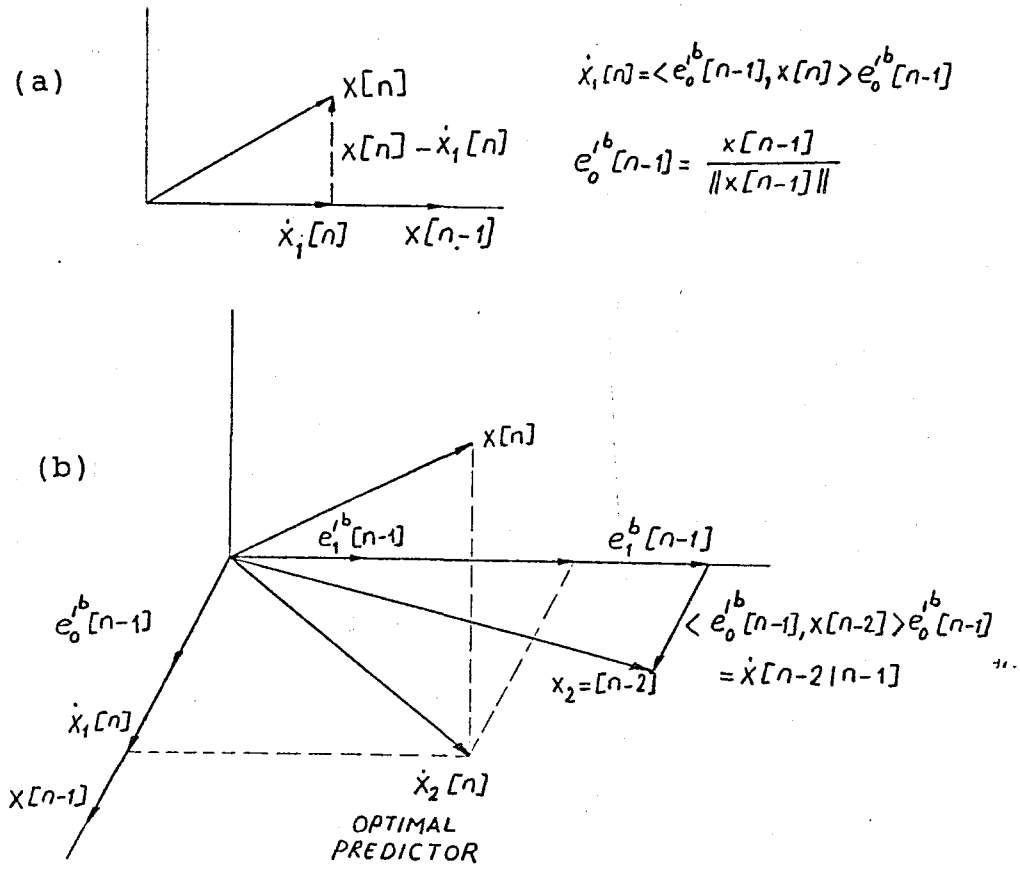


Figure 3.1.4. Vector space interpretation of linear prediction (a) First prediction (b) Second order predictor

The solution, depicted geometrically in Figure 3.1.4a, can be obtained using the orthogonality principle as

$$\langle x[n-1], x[n]-x'[n]\rangle=0 \tag{3.1.24}$$

which yields

$$a_1[1]=-\frac{\langle x[n-1], x[n]\rangle}{\langle x[n-1], x[n-1]\rangle} \tag{3.1.25}$$

so that

$$x'_1[n]=\frac{\langle x[n-1], x[n]\rangle}{\langle x[n-1], x[n-1]\rangle} \tag{3.1.26}$$

Now let

$$e'^{b}_{0}[n-1]=\frac{x[n-1]}{\|x[n-1]\|}. \tag{3.1.27}$$

$e_0'^{b}[n-1]$ is a zero-mean random variable with prime denoting that it is also unit variance. A quotation mark will henceforth denote a random variable that is "normalized" or has unit variance. The "0" subscript and "b" superscript are explained below. The optimal first order predictor then becomes, from Eq.(3.1.26).

$$x'_1[n]=\frac{\langle x[n-1], x[n]\rangle}{\|x[n-1]\|}\,\frac{x[n-1]}{\|x[n-1]\|}$$

$$x'_1[n]=\langle e'^{b}_{0}[n-1], x[n]\rangle e'^{b}_{0}[n-1] \tag{3.1.28}$$

It is seen that the optimal first order linear predictor is found by projecting x[n] along the x[n - 1] "direction," where the "unit vector" along the x[n - 1] direction is $e'^{b}_{0}[n-1]$.

Now consider a second order or updated linear predictor with k - 1=2: $x_2'[n]=-a_2[1]x[n-1]-a_2[2]x[n-2]$.

Referring to Figure 3.1.4b,we observe that x[n-2] is in general not orthogonal to x[n-1]. This means that x[n-2] is correlated with x[n-1], so that not all of the information

provided by x[n-2] about x[n] is new information. The optimal predictor $x_2'[n]$ can be decomposed into the sum of two vectors in orthogonal directions. One of the directions has already been specified by x[n-1]. The second direction will be that which is orthogonal to x[n-1]. The optimal second order predictor combines the first order predictor with the best prediction of x[n] based on the new information provided by x[n-2], or

$$x'_2[n] = x'_1[n] + \textit{best prediction of } x[n] \textit{ based on part of } x[n-2] \textit{ in new orthogonal direction} \tag{3.1.29}$$

To find the new information of part of x[n-2] orthogonal to x[n-1], recall that if we "predict" x[n-2] based on x[n-1], the error will be orthogonal to x[n-1]. Let x'[n-2|n-1] be the prediction of c[n-2] based on x[n-1] and let $e^b_1[n-1]$ be the error. Then,referring to Figure 3.1.4b, we have

$$\begin{aligned} e_1^b &= x[n-2] - x'[n-2|n-1] \\ &= x[n-2] - \langle e'^b_0[n-1], x[n-2] \rangle e'^b_0[n-1] . \end{aligned} \tag{3.1.30}$$

x'[n-2|n-1] is called the backward prediction since it is an estimate of x[n-2] based on the future sample x[n-1]. Also,$e^b_1[n-1]$ is called the backward prediction error of order 1. The subscript denotes the order of the prediction error or number of future samples used in the prediction. The b subscript has been added to distinguish it form the usual forward prediction error. The (k-1)st order forward prediction error, $x'[n]-x_{k-1}[n]$, will be denoted by $e^f_{k-1}[n]$. From Figure 3.1.4b, $e^b_1[n-1]$ is orthogonal to x[n-1] and so represents the new information in x[n-2] about x[n] not already provided by x[n-1]. For this reason $e^b_1[n-1]$ is sometimes referred to as the innovation.

If $e^b_1[n-1]$ is normalized.

$$e'^{b}_{1}[n-1] = \frac{e^{b}_{1}[n-1]}{\|e^{b}_{1}[n-1]\|} \tag{3.1.31}$$

then, from Eq.(3.1.29),

$$x'_{2}[n] = x'_{1}[n] + \langle e'^{b}_{0}[n-1], x[n] \rangle e'^{b}_{1}[n-1] . \tag{3.1.32}$$

The evaluation of the inner products will produce the equivalent from

$$x'_{2}[n] = -a_{2}[1]x[n-1] - a_{2}x[n-2] .$$

In general, if $x_{k}[n] = x'_{k-1}[n] + \langle e'^{b}_{k-1}[n-1], x[n] \rangle e'^{b}_{k-1}[n-1]$ (3.1.33)

where $e'^{b}_{k-1}[n-1]$ is the backward prediction error if x[n-k] is predicted on the basis of {x[n-(k-1),x[n-(k-2)],...,x[n-1]}. See Figure 3.1.5 for an illustration.

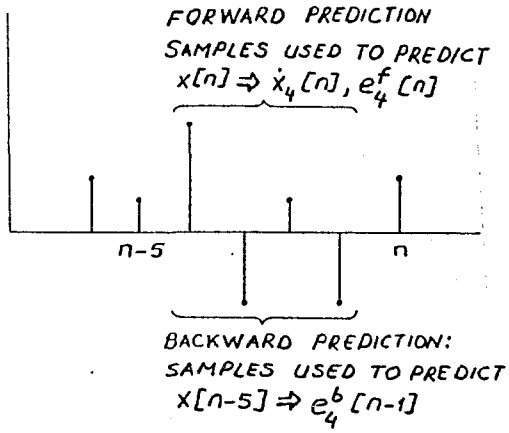


Figure 3.1.5. Illustration of forward and backward prediction

Note that $e^{-b}_{k-1}[n-1]$ has been defined so that the time index n-1 refers to the latest sample used in the prediction, not to the sample to be "predicted". Using Eg.(3.1.43), the kth order predictor becomes

$$x'_k[n] = -\sum_{i=1}^{k-1} a_{k-1}[i]\,x[n-i] + \frac{\langle e^b_{k-1}[n-1], x[n]\rangle}{\|e^b_{k-1}[n-1]\|^2}$$

(3.1.34)

Now let

$$k_k = -\frac{\langle e^b_{k-1}[n-1], x[n]\rangle}{\|e^b_{k-1}[n-1]\|^2}$$

(3.1.35)

Where k_k is termed the kth reflection coefficient. The backward prediction error may be written explicitly as

$$e^b_{k-1}[n-1] = x[n-k] - \left[-\sum_{i=0}^{k-2} b_{k-1}[i]\,x[n-1-i]\right]$$

(3.1.36)

where $b_{k-1}[i]$ are the optimal backward prediction coefficients. If we define $b_{k-1}[k-1]=1$, the backward prediction error becomes

$$e^b_{k-1}[n-1] = \sum_{i=0}^{k-1} b_{k-1}[i]\,x[n-1-i].$$

(3.1.37)

Substituting Eq.(3.1.35) and Eq.(3.1.37) into Eq.(3.1.34) result in

$$x'_k[n] = -\sum_{i=1}^{k-1} a_{k-1}[i]\,x[n-i] - k_k \sum_{i=0}^{k-1} b_{k-1}[i]\,x[n-1-i]$$

(3.1.38)

which must be identical to

$$x'_k[n] = -\sum_{i=1}^{k} a_k[i]\,x[n-i].$$

(3.1.39)

The optimal prediction coefficients for the kth order predictor will be the sum of the coefficients for the (k-1) st order predictor and a correction term due to $k_k b_{k-1}[i]$.

The relationship between the backward prediction, is based on k-1 future samples, and the forward prediction , is

based on the k-1 samples, an example of which is shown in Figure 3.1.5. Considering n=0 for siplicity, in forward prediction we predict x[0] based on {x[-1],x[-2], ..., x[-(k-1)]}, while in bacward prediction we predict x[-k] based on {x[-(k-1), x[-(k-2)], ...,x[-1]}. The two problems are nearly equivalent except for the reversal of time, so that it is not surprising that the optimal bacward prediction coefficients are the same as the optimal forward prediction coefficients except reversed in time and complex conjugated. This relationship is also apparent if we consider the coefficients of the forward and backward AR models it is now shown that

$$b_{k-1}[i] = a^*_{k-1}[k-1-i] \qquad i=0,1,\ldots,k-1 \tag{3.1.40}$$

The optimal backward predictor coefficients are used. An explicit form for the kth order prediction coefficients

$$\begin{aligned} x'_k[n] = -\sum_{i=1}^{k-1} a_{k-1}[i]\,x[n-i] - k_k\,(x[n-k] \\ + \sum_{i=0}^{k-2} a^*_{k-1}[k-1-i]\,x[n--1]\Big) \\ -\sum_{i=1}^{k-1} (a_{k-1}[i] + k_k a^*_{k-1}[k-i])\,x[n-i] - k_k x[n-k]\,. \end{aligned} \tag{3.1.41}$$

For details refer to [6]. Also, it may be expressed as

$$\hat{x}_k[n] = -\sum_{i=1}^{k} a_k[i]\,x[n-i] \tag{3.1.42}$$

and consequently, equating the two expressions yields

$$a_k[i] = \begin{cases} a_{k-1}[i] + k_k a^*_{k-1}[k-i] & i=1,2,\ldots,k-1 \\ k_k & i=k \end{cases} \tag{3.1.43}$$

Equations Eq.(3.1.43) are the model order update relations for the prediction coefficients for the kth order predictor. Note that the new coefficients are computed recursively based on the coefficients of the previous lower order predictor and the new reflection coefficient. Also, the $a_k[k]$ coefficient is just the reflection coefficient.

To complete the recursion of Eq.(3.1.43), we need to compute the reflection coefficient sequence. From Eq.(3.1.34),

$$k_k = -\frac{\langle e^b_{k-1}[n-1], x[n] \rangle}{\| e^b_{k-1}[n-1] \|^2}$$

(3.1.44)

Then using Eq.(3.1.37) ,Eq.(3.1.40) and the orthogonally principle. The reflection coefficient is found to be

$$k_k = -\frac{r_{xx}[k] + \sum_{i=1}^{k-1} a_{k-1}[i]\, r_{xx}[k-i]}{r_{xx}] + \sum_{i=1}^{k-1} a_{k-1}[i]\, r_{xx}[-i]}$$

(3.1.45)

Note that the reflection coefficients depend on the ACF as well as the lower order PEFs.
The Interpretation of k_k is as a correlation coefficient. The first step in making this correspondence is to realize that

$$\| e^b_{k-1}[n-1] \|^2$$

(3.1.46)

, which is the prediction error power for the (k-1)st order backward prediction is the same as that for the (k-1)st order forward prediction.

$$\| e^b_{k-1}[n-1] \|^2 = \| e^f_{k-1}[n-1] \|^2 = \| e^f_{k-1}[n] \|^2 = \rho_{k-1}.$$

(3.1.47)

This is the consequence of the hermitian symmetry property of the ACF of a WSS process.

From Eq.(3.1.35) and Eq.(3.1.47),

$$k_k = -\frac{\langle e^b_{k-1}[n-1], e^f_{k-1}[n]\rangle}{\|e^f_{k-1}[n]\| \; \|e^b_{k-1}[n-1]\|}$$

$$= -\frac{cov(e^b_{k-1}[n-1], e^f_{k-1}[n])}{\sqrt{var(e^f_{k-1}[n])}\;\sqrt{var(e^b_{k-1}[n-1])}} \qquad (3.1.48)$$

Where cov denotes the covariance. k_k is bounded by 1 in magnitude by the Cauchy-Schwartz inequality. The reflection coefficient is readily seen to be the negative of the correlation coefficient between the forward and backward prediction errors.

Finally, to complete the development of the Levinson algorithm, the simple recursive expression for p_k , the prediction error power for the kth order linear predictor,

$$\rho_k = (1 - |k_k|^2)\,\rho_{k-1} \qquad (3.1.49)$$

is derived. From Eq.(3.1.33) and Eq.(3.1.35) the kth order linear predictor may be written as

$$x'_k[n] = x'_{k-1}[n] - k_k e^b_{k-1}[n-1]. \qquad (3.1.50)$$

Adding -x[n] to both sides of this expression produces

$$e^f_k[n] = e^f_{k-1}[n] + k_k e^b_{k-1}[n-1]. \qquad (3.1.51)$$

Using standard properties of inner products and Eq.(3.1.47), we have

$$\begin{aligned}\rho_k &= \|e^f_k[n]\|^2 \\ &= \langle e^f_{k-1}[n] + k_k e^b_{k-1}[n-1], e^f_{k-1}[n] + k_k e^b_{k-1}[n-1]\rangle \\ &= \rho_{k-1} + k_k\langle e^f_{k-1}[n], e^b_{k-1}[n-1]\rangle \\ &\quad + k_k^*\langle e^b_{k-1}[n-1], e^f_{k-1}[n]\rangle - k_k\rho_{k-1}\end{aligned} \qquad (3.1.52)$$

But from Eq.(3.1.47) and Eq.(3.1.48)

$$\langle e^b_{k-1}[n-1], e^f_{k-1}[n]\rangle = -k_k\rho_{k-1} \qquad (3.1.53)$$

which upon substitution in the equation above result in Eq.(3.1.49). As expected, the prediction error power decreases as the order of the predictor increases (assuming that $k_k \neq 0$) and is nonnegative since $|k_k|<1$.

In summary , the Levinson algorithm recursively computes the parameter sets $\{a_1[1],\rho_1\}$, $\{a_2[1],a_2[2], p2\},\ldots.$, $\{a_p[1], a_p[2], \ldots.., a_p[p], \rho_p\}$. The final set at order p is the desired solution of the Yule-Walker equations. If x[n] is an AR (p) process, then $a_p[i]= a[i]$ for $i=1,2,\ldots,p$ and $p^p=\sigma^2$, as described in section 3.1.1. The recursive algorithm is initialized by

$$a_1[1] = -\frac{r_{xx}[1]}{r_{xx}[0]}$$
$$\rho_1 = (1 - |a_1[1]|^2)\, r_{xx}[0] \tag{3.1.54}$$

with the recursion for $k=2,3,\ldots,p$ given by

$$a_k[k] = -\frac{r_{xx}[k] + \sum_{l=1}^{k-1} a_{k-1}[l]\, r_{xx}[k-l]}{\rho_{k-1}} \tag{3.1.55}$$

$$a_k[i] = a_{k-1}[i] + a_k[k]\, a^*_{k-1}[k-i] \qquad i=1,2,\ldots,k-1 \tag{3.1.56}$$

$$\rho_k = (1-|a_k[k]|^2)\, \rho_{k-1} \tag{3.1.57}$$

The reflection coefficients are given by $k_k = a_k[k]$. The Levinson algorithm is summarized in Figure (3.1.6) The form of the algorithm given in Eq.(3.1.55)-(3.1.57) due Toeplitz set of equations, who refined the algorithm to take advantage of the special form of the right-hand-side vector. It is important to note that $\{a_j[1], a_j[2],\ldots,a_j[j],\rho_j\}$, as obtained from the Levinson algorithm is the same as would be obtained by solving Eq.(3.1.4) and Eq.(3.1.5) with p = j. The algorithm provides the AR parameters for all lower order AR model fits to the ACF as well as the desired model. This is a useful property when we

do not know a priori the correct model order. Using the Levinson recursion, successively higher order models can be generated until the modelling error ρ_k is reduced to a desired value. If the process is actually an AR(p) process, then $a_{p+1}[k]=a_p[k]$ for $k=1,2,\ldots,p$ and $a_{p+1}[p+1]=k_{p+1}=0$. In general, for an AR (p) process, $a_k[k]=k_k=0$ for $k>p$ and hence $\rho_k=\rho_p$ for $k>p$. This says that the variance of the excitation noise in the model is a constant for a model order equal to or greater than the true order. Hence the point at which ρ_k does not change would appear to be a good indicator of the correct model order.

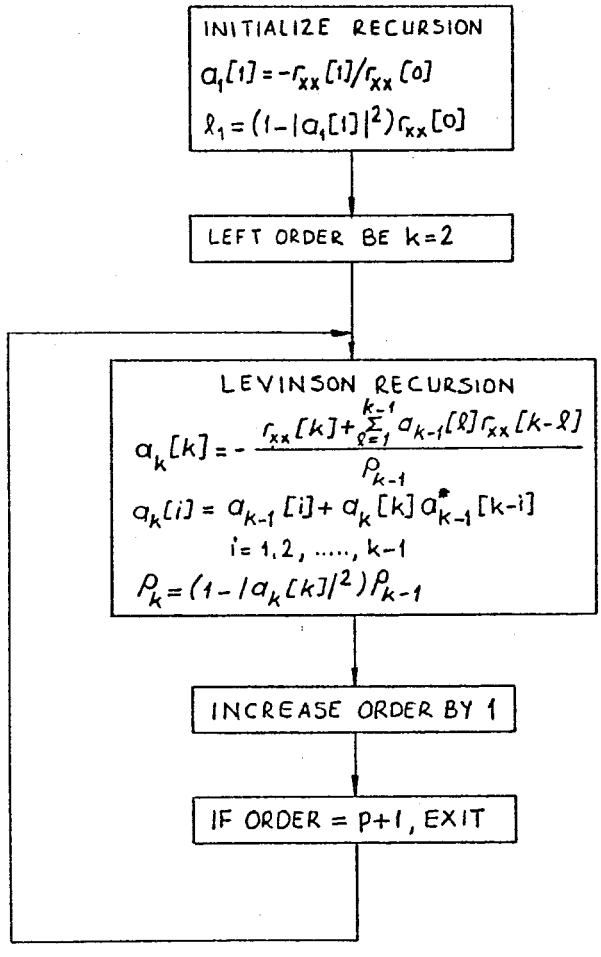


Figure 3.1.6. Summary of Levinson recursion

The property that

$$|a_k[k]| - |k_k| < 1 \tag{3.1.58}$$

leads to

$$\rho_{k+1} \leq \rho_k \tag{3.1.59}$$

$$\begin{aligned} a_k[k] &= -\frac{r_{xx}[k] + \sum_{l=1}^{k-1} a_{k-1}[l]\, r_{xx}[k-l]}{\rho_{k-1}} \\ a_k[i] &= a_{k-1}[i] + a_k[k]\, a^*_{k-1}[k-i] \qquad i = 1, 2, \ldots, k-1 \\ \rho_k &= (1 - |a_k[k]|^2)\, \rho_{k-1} \end{aligned} \tag{3.1.60}$$

which furthermore implies that ρ_k first attains its minimum value at the correct model order.

$$|k_k| = 1 \tag{3.1.61}$$

for some k, the recursion must terminate since $\rho_k = 0$. This case will only occur, however, if the process consists solely of k sinusoids.

3.1.2. AR PARAMETER AND PSD CALCULATION TECHNIQUES

There are various methods estimating AR parameters, here only a few of them which are related to our research will be expressed briefly.

3.1.2.1. Autocorrelation Method

As usual, it is assumed that the data { x[0], x[1] ...,x[N]} are observed. The AR parameters are estimated by minimizing an estimate of the prediction error power

$$\rho' = \frac{1}{N}\sum_{n=-\infty}^{\infty}\left| x[n] + \sum_{k=1}^{p} a[k]\,x[n-k]\right|^2$$

(3.1.62)

The samples of the x(n) process which are not observed (i.e., those not in the range $0 \le n \le N-1$ are set to zero in Eq.(3.1.62). The estimated prediction error power is minimized by differentiating Eq.(3.1.62). with respect to the real and imaginary parts of a[k]'s. This may be done by using the complex gradient to yield

$$\frac{1}{N}\sum_{n=-\infty}^{\infty}\left(x[n] + \sum_{k=1}^{p} a[k]\,x[n-k]\right) x^*[n-l] = 0 \qquad l = 1,2,\ldots,P$$

(3.1.63)

In matrix form this set of equations becomes

$$\begin{bmatrix} r_{xx}[0] & r_{xx}[-1] & \cdots & r_{xx}[-(p-1)] \\ r_{xx}[1] & r_{xx}[0] & \cdots & r_{xx}[-(p-2)] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[p-1] & r_{xx}[p-2] & \cdots & r_{xx}[0] \end{bmatrix} \begin{bmatrix} a'[1] \\ a'[2] \\ \vdots \\ a'[p] \end{bmatrix} = - \begin{bmatrix} r'_{xx}[1] \\ r'_{xx}[2] \\ \vdots \\ r'_{xx}[p] \end{bmatrix}$$

(3.1.64)

where

$$r'_{xx}(k) = \begin{cases} \frac{1}{N}\sum_{n=0}^{N-1-k} x^*[n]\,x[n+k] & \text{for } k = 0,1,\ldots,p \\ r'^{*}_{xx}[-k] & \text{for } k = -(p-1),-(p-2),\ldots,-1 \end{cases}$$

(3.1.65)

which is recognized as the biased ACF estimator. The matrix in Eq.(3.1.64) is hermitian ($r_{xx}'[-k] = \tilde{r}_{xx}[k]$) and Toeplitz, and furthermore can be shown to be positive definite. The alternative Yule-Walker method is due to the equivalence of the autocorrelation method to the use of the Yule-Walker equations with a biased ACF estimator. As such, the Levinson recursion may be used to solve the equations and the resulting estimated poles are guaranteed to be within the unit circle by the minimum-phase theorem.

The estimate of the white noise variance σ^2 is found as

ρ'_{MIN}, which is given

$$\sigma'^2 = \rho'_{MIN} = \frac{1}{\sum_{n=-\infty}^{\infty}} \left| x[n] + \sum_{k=1}^{p} a'[k]\, x[n-k] \right|^2$$

$$\sigma'^2 = \rho'_{MIN} = \frac{1}{N} \sum_{n=-\infty}^{\infty} \left[\left(x[n] + \sum_{k=1}^{p} a'(k)\, x[n-k] \right) x^*[n] + \left(x[n] + \sum_{k=1}^{p} a'[k]\, x[n-k] \right) \sum_{l=1}^{p} a'^*[l]\, x^*[n-l] \right]$$

(3.1.66)

From Eq.(3.1.64) the second term in the summation over n is zero, leading to the final result that

$$\sigma'^2 = r'_{xx}[0] + \sum_{k=1}^{p} a'[k]\, r'_{xx}[-k] \,.$$

(3.1.67)

$\sigma^{2'}$ may also be found in the last step of the Levinson recursion as the i'th order prediction power or in the alternative form as

$$\sigma'^2 = r'_{xx}[0] \prod_{i=1}^{p} (1 - |k'_i|^2)$$

(3.1.68)

where k_i' is the estimate of the i'th order reflection coefficient generated within the Levinson recursion,. The autocorrelation method given the above formulas are implemented in Pascal programs are given in the Appendix A of this thesis.

The autocorrelation method has been found to produce poorer resolution spectral estimates than the other estimators. For this reason it is not usually recommended for short data records. A variant of this approach is to use the unbiased autocorrelation estimator in the Yule-Walker equations. With this modification it may be shown that the autocorrelation matrix in Eq.(3.1.64) is no longer guaranteed to be positive definite.As a consequence, of this spectral estimators exhibit a large variance. The use of the unbiased ACF estimator is therefore not recommended.

3.1.2.2. Covariance Method

The covariance method was derived for real data as an approximate MLE. For complex data the analogous estimator may be found by minimizing the estimate of the prediction error power

$$\rho' = \frac{1}{N-p} \sum_{n=p}^{N-1} \left| x[n] + \sum_{k=1}^{p} a[k]\ x[n-k] \right|^2 . \tag{3.1.69}$$

Note that the only difference between the covariance method and the autocorrelation method is the range of summation in the prediction error power estimate. In the covariance method all the data points needed for computation have been observed. No zeroing data is necessary.

The minimizing of Eq (3.1.55) may be effected by applying complex gradient to yield the AR parameter estimates as the solution of the equations.

$$\begin{bmatrix} c_{xx}[1,1] & c_{xx}[1,2] & \cdots & c_{xx}[1,p] \\ c_{xx}[2,1] & c_{xx}[2,2] & \cdots & c_{xx}[2,p] \\ \vdots & \vdots & \ddots & \vdots \\ c_{xx}[p,1] & c_{xx}[p,2] & \cdots & c_{xx}[p,p] \end{bmatrix} \begin{bmatrix} a'[1] \\ a'[2] \\ \vdots \\ a'[p] \end{bmatrix} = - \begin{bmatrix} c_{xx}[1,0] \\ c_{xx}[2,0] \\ \vdots \\ c_{xx}[p,0] \end{bmatrix} \tag{3.1.70}$$

where

$$c_{xx}[j,k] = \frac{1}{N-p} \sum_{n=p}^{N-1} x^*[n-j]\ x[n-k] . \tag{3.1.71}$$

the white noise variance is estimated as

$$\sigma'^2 = \rho'_{MIN} = c_{xx}[0,0] + \sum_{k=1}^{p} a'[k]\ c_{xx}[0,k] . \tag{3.1.72}$$

The matrix in Eq.(3.1.70) is hermitian ($c_{xx}[k,j]=(c^{*}_{xx}[j,k])$ and positive semi-definite. It may be shown to be singular if the data consist of p-1 or fewer complex sinusoids. Any equations may be solved using the Cholesky decomposition, but the estimated poles using the covariance method are not guarantied to lie within the unit circle.

As implied from the definition, $c_{xx\epsilon}[j,k]$ is readily seen to be an estimate of $r_{xx}[j-k]$, although a different estimate than that encountered in the autocorrelation method. $c_{xx}[j,k]$ uses the sum of only N-p lag products to estimate the ACF for each lag even though more are available. As an example, in the estimation of $r_{xx}(0)$ the biased auto-correlation estimator of the autocorrelation method uses all N data points, while the covariance method uses only N-p data points in the summation. For large data records in which N>>p, these "end effects" are negligible and consequently, the autocorrelation and covariance methods will yield similar spectral estimates. A second contrasting feature is that for data consisting of pure sinusoids the covariance method may be used to perfectly extract the frequencies. This property is not shared by the autocorrelation method. Methods to estimate sinusoidal frequencies as described more fully are based on the covariance method. For detailed information one can refer to [6], [9], [14].

3.1.2.3. Modified Covariance Method

For an AR(p) process the optimal forward predictor is

$$x'[n] = -\sum_{k=1}^{p} a[k]\,x[n-k] \tag{3.1.73}$$

while the optimal predictor is

$$x'[n] = -\sum_{k=1}^{p} a^*[k]\, x[n+k] \tag{3.1.74}$$

Where the a[k]'s are the AR filter parameters. In either case the minimum prediction error power is just the white noise variance σ_w. The modified covariance method estimates the AR parameters by minimizing the average of estimated

$$\rho' = \frac{1}{2}(\rho'^{f} + \rho'^{b}) \tag{3.1.75}$$

where

$$\rho'^{f} = \frac{1}{N-p}\sum_{n=p}^{N-1}\left|x[n] + \sum_{k=1}^{p} a[k]\, x[n-k]\right|^2$$

$$\rho'^{b} = \frac{1}{N-p}\sum_{n=0}^{N-1-p}\left|x[n] + \sum_{k=1}^{p} a^*[k]\, x[n+k]\right|^2 \tag{3.1.76}$$

As in the covariance method the summations are only over the prediction errors that involve observed data samples. Note that an alternative way of viewing this estimator is to recognize that ρ'^{b} is the prediction error power estimated obtained by "flipping the data record" and complex conjugating and applying a forward predictor to this new data set. In this manner we obtain some extra data points and hence more prediction errors over which to average. Note that for any set of a[k]'s the forward and backward prediction error estimates will be slightly different due to the range of the summation.

To minimize prediction error power, we can differentiate the error power to the real and imaginary parts of a[k] for k=1,2,....p. By taking the advantage of the complex gradient relationship it yields

$$\frac{\partial \rho'}{\partial a[l]} = \frac{1}{N-p}\left[\sum_{n=p}^{N-1}\left(x[n]+\sum_{k=1}^{p} a[k]\,x[n-k]\right)x^*[n-l]\right.$$

$$\left.+\sum_{n=0}^{N-1-p}\left(x^*[n]+\sum_{k=1}^{p} a[k]\,x^*[n+k]\right)x[n+l]\right]$$

$$= 0 \qquad l = 1,2,,\ldots p$$

(3.1.77)

After some simplification this becomes

$$\sum_{k=1}^{p} a'[k]\left(\sum_{n=p}^{N-1} x[n-k]\,x^*[n-l]+\sum_{n=0}^{N-1-p} x^*[n+k]\,x[n+l]\right)$$

$$= -\left(\sum_{n=p}^{N-1} x[n]\,x^*[n-l]+\sum_{n=0}^{N-1-p} x^*[n]\,x[n+l]\right)$$

for $l = 1,2,\ldots p$. *Letting*

$$c_{xx}[j,k] = \frac{1}{2[N-p]}\left(\sum_{n=p}^{N-1} x^*[n-j]\,x[n-k]+\sum_{n=0}^{N-1-p} x[n+j]\,x^*[n+k]\right)$$

(3.1.78)

The equations for finding parameters can be written in identical matrix form as

$$\begin{bmatrix} c_{xx}[1,1] & c_{xx}[1,2] & \cdots & c_{xx}[1,p] \\ c_{xx}[2,1] & c_{xx}[2,2] & \cdots & c_{xx}[2,p] \\ \vdots & \vdots & \ddots & \vdots \\ c_{xx}[p,1] & c_{xx}[p,2] & \cdots & c_{xx}[p,p] \end{bmatrix} \begin{bmatrix} a'[1] \\ a'[2] \\ \vdots \\ a'[p] \end{bmatrix} = - \begin{bmatrix} c_{xx}[1,0] \\ c_{xx}[2,0] \\ \vdots \\ c_{xx}[p,0] \end{bmatrix}$$

(3.1.79)

The estimate of the white noise variance is

$$\sigma'^2 = \rho'_{MIN} = \frac{1}{2[N-p]}\left[\sum_{n=p}^{N-1}\left(x[n]+\sum_{k=1}^{p} a'[k]\,x[n-k]\right)x^*[n]\right.$$

$$\left.+\sum_{n=0}^{N-1-p}\left(x^*[n]+\sum_{k=1}^{p} a'[k]\,x^*[n+k]\right)x[n]\right]$$

(3.1.80)

finally,

$$\sigma'^2 = C_{xx}[0,0] + \sum_{k=1}^{p} a'[k]\, c_{xx}[0,k] \tag{3.1.81}$$

The modified covariance method appears to yield statistically stable spectral estimates with high resolution. For more information on this one can refer to [6], [9],[14].

3.1.2.4. Burg Method

In contrast to the autocorrelation, covariance, and modified covariance methods, which estimate the AR parameters directly, the Burg method estimates the reflection coefficients and then uses the Levinson recursion to obtain the AR parameter estimates. The reflection coefficient estimates are obtained by minimizing estimates of the reflection coefficients $\{k_1, k_2, \ldots . k_p\}$ are available, the AR parameters may be estimated as follows:

$$\begin{aligned} & r'_{xx}[0] = \frac{1}{N}\sum_{n=0}^{N-1} |x[n]|^2 \\ & a'[1] = k'_1 \\ & \rho'_1 = (1-|a'_1[1]|^2)\, r'_{xx}[0] . \end{aligned} \tag{3.1.82}$$

For k=2,3,...p,

$$a'_k[i] = \begin{cases} a'_{k-1}[i] + k'_k a'^{*}_{k-1}[k-i] & \text{for } i=1,2,\ldots k-1 \\ k'_k & \text{for } i=k \end{cases} \tag{3.1.83}$$

For detailed information one can refer to [6], [9], [12], [13].

3.1.3. Model Order Selection

The selection of the model order in AR spectral estimation is a critical one. Too low an order results in a

smoothed estimate while too large an order causes spurious peaks and general statistical instability. Many techniques have been derived by statistical analysis of real data. It is probable that these model order estimators may be applied directly to complex data; however, the extensions to complex data are not available.

For data observed from a pure AR process the model order estimators produce acceptable spectral estimates if the data record length is not extremely short [15]. It has been observed that for noise corrupted data the AR model order chosen is usually not sufficient to resolve spectral details. Of course, the true AR model for noisy data is of infinite order so that this result is not unexpected. It should also be emphasized that different estimates of the model order will be obtained if different AR parameter estimators are used in conjunction with the same model order estimator. No detailed studies are available which assess the spectral estimation performance of the various AR spectral estimators when the model order must be estimated in addition to the AR parameters. In comparing the strengths and weaknesses of the missing phose model order estimators, we should keep in mind that it is the quality of the spectral estimate, which is of importance. For example, an estimator that underestimates the true AR model order for broadband PSDs, which are smooth in appearance may well be preferable to one that indicates the true orders of the broadband AR process but which, when combined with an AR parameter estimator, gives rise to spurious peaks. This situation is possible if, for example, the data record is short.

Nearly all model order estimators are based on the estimated prediction error power. The estimated prediction error power is guaranteed to decrease or stay the same as the model order increases for all the AR parameter estimation methods described. Hence we cannot simply monitor the decrease in power as a means of determining model order but must also account for the increase in variance of a spectral estimate

based on an increasing number of parameters. Two methods proposed by Akaike adhere to this philosophy. The fist one, termed the final prediction error (FPE), estimates the model order as the value that minimizes

$$FPE[k] = \frac{N+k}{N-k}\rho'_k \tag{3.1.84}$$

where ρ'_k is the estimate of the white noise variance (prediction error power) for the kth order AR model. It is seen that whereas ρ_k decreases with k, the term (N + k)/(N - k) increases with k. The FPE is an estimate of the prediction error power when the prediction coefficients must be estimated from the data. The term (N +k)/(N - k) accounts for the increase in the variance of the prediction error power estimator due to the inaccuracies in the prediction coefficient estimates.

A second criterion, which appears to be in more general usage, is the Akaike information criterion (AIC). It is defined as

$$AIC[k] = N \ln \rho'_k + 2k. \tag{3.1.85}$$

As before, the order selected is the one that minimizes the AIC is an estimate of the Kullback-Leibler [15] distance between an assumed PDF and the true PDF of the data. The method is not limited to AR model order determination but may be used more generally for choosing a model among competing models. Consequently, the AIC is useful for MA and ARMA model order determination.The performance of the AIC and FPE is similar. For short data records the use of the AIC is recommended. For larger data records ($N \rightarrow \infty$) the two estimators will yield identical model order estimates since they are functionally related to each other.

3.2. MOVING AVERAGE (MA) MODELLING

MA models are appropriate for processes that have broad peaks or sharp nulls in their spectra. Since the MA is based on all-zero model of the data,it is not possible to use it to estimate PSDs with sharp peaks. Because the MA spectral estimator is not a high resolution spectral estimator for processes with narrowband spectral features, investigations of its properties have been somewhat limited.

In the following most general ARMA model, without loss of generality, we can assume that all a[k] coefficients except a[0]=1 for ARMA parameters,

$$x[n] = -\sum_{k=1}^{p} a[k]\,x[n-k] + \sum_{k=0}^{q} b[k]\,u[n-k] \tag{3.2.1}$$

then,

$$x[n] = \sum_{k=0}^{q} b[k]\,u[n-k] \tag{3.2.2}$$

and the process is strictly an MA process or order q, and

$$P_{MA}(f) = \sigma^2 |B(f)|^2 \tag{3.2.3}$$

This model is sometimes termed an all-zero model and is show in Figure (3.2.1).

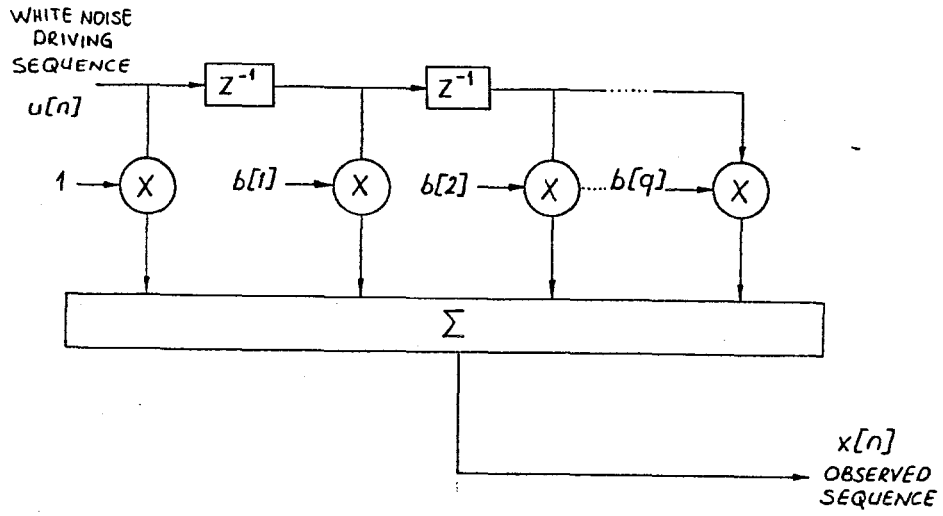


Figure 3.2.1. Moving Average model of random process

A flowchart of algorithm to estimate the MA parameters from a sample sequence is illustrated in Figure below.

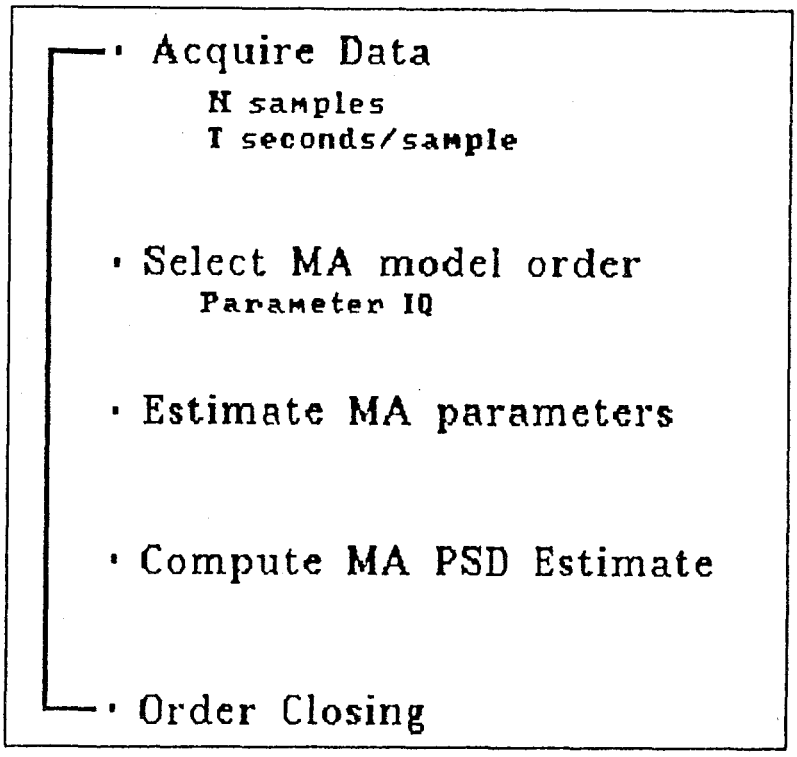


Figure 3.2.2. Summary of MA modelling technique

When it is assumed that x[n] is an MA(q) process, the problem is to estimate {b[1], b[2],....b[q],σ^2}. For reliable estimates of the MA parameters the MLE will be employed. Equivalently, we could obtain the MLE's for {r_{xx}[0]), r_{xx}[1],....., r_{xx}[q]} and calculate PSD using above formula. In the following part approximate MLE's for the MA parameters and the MA PSD are described. The algorithm, first converts the MA(q) process into an AR process and then uses the Yule-Walker equations to estimate the MA parameters.

3.2.1 MAXIMUM LIKELIHOOD ESTIMATION: DURBIN'S METHOD

Durbin's method is an approximate MLE. It is derived for real data, but the extension to complex data is straightforward and is given. The first step is to replace the MA(q) process by an approximate AR(L) process. An MA process

$$x[n] = \sum_{k=0}^{q} b[k]\, u[n-k] \tag{3.2.4}$$

is equivalent to the AR(∞) process.

$$x[n] = -\sum_{k=1}^{\infty} a[k]\, x[n-k] + u[n] \tag{3.2.5}$$

if a[k] is the impulse response of 1/B(z). This is immediately observed if we let

$$H(z) = B(z) = \frac{1}{A(z)} \tag{3.2.6}$$

so that

$$A(z) = \frac{1}{B(z)}. \tag{3.2.7}$$

If the impulse response of 1/B(z) has decayed to zero for an index greater than L, then an AR(L) process will be a good approximation to the MA(q) process. Now instead of considering the likelihood function for the data directly, we can use the likelihood function for the AR parameter estimates. This is because the usual AR parameter estimator is a sufficient statistic for the AR parameters for large data record. Let a', σ'^2 be the AR parameter estimates obtained by any of the methods of previous part (i.e.,any of the approximate MLE techniques) using an AR(L) model. For large data records $\Theta' = [\, a'^T \;\; \sigma'^2 \,]$ is distributed according to a multivariate Gaussian

PDF with mean

$$\mathcal{E}(\theta) = \theta = \begin{bmatrix} a \\ \sigma^2 \end{bmatrix} \tag{3.2.8}$$

and covariance matrix

$$C_{a,\sigma^2} = \begin{bmatrix} \frac{\sigma^2}{N} T_{xx}^{-1} & \theta \\ \theta^T & \frac{2\sigma^4}{N} \end{bmatrix} \tag{3.2.9}$$

where R_{xx} is the L x L autocorrelation matrix of the MA(q) or equivalent AR(L) process. The determinant of the covariance matrix is,

$$\det(C_{a,\sigma^2}) = \frac{2\sigma^4}{N}\left(\frac{\sigma^2}{N^L}\right)\det^{-1}(R_{xx}) \tag{3.2.10}$$

It may be shown that for large L,

$$\det(R_{xx}) \approx \sigma^{2L} \tag{3.2.11}$$

so that

$$\det(C_{a,\sigma^2}) = \frac{2\sigma^4}{N^{L+1}} \tag{3.2.12}$$

Thus the MLE is just σ'^2, which is the estimate obtained using the AR(L) model. Assuming that the autocorrelation method has been used for real data, autocorrelation function of AR is

$$\sigma'^2 = r'_{xx}[0] + \sum_{k=1}^{L} a'[k]\, r'_{xx}[k] \,. \tag{3.2.13}$$

The approximate MLE for the MA filter parameters is

$$b' = -R'^{-1}_{aa} r'_{aa} \qquad (3.2.14)$$

where

$$[R'_{aa}]_{ij} = \frac{1}{L+1} \sum_{n=0}^{L-|i-j|} a'[n]\, a'[n+|i-j|] \qquad i,j = 1,2,\ldots q$$

$$[r'_{xx}]_i = \frac{1}{L+1} \sum_{n=0}^{L-i} a'[n]\, a'[n+i] \qquad i = 1,2,\ldots\ldots q. \qquad (3.2.15)$$

Eq.(3.2.14) is Durbin's method for MA parameter estimation. The Levinson algorithm may be used to solve the equations for b. Because of the minimum phase property of the autocorrelation method the estimated zeros of B(z) will be inside the unit circle. Many variants of Durbin's method may be generated by replacing the autocorrelation method of AR parameter estimation by any of the techniques described in Autoregressive modelling part.

In summary, Durbin's algorithm for the estimation of the MA parameters of an MA(q) process proceeds as follows:

1. Using the data {x[0]),x[1],....x[n - 1]}, fit a large order AR model using the autocorrelation method. For an AR model order of L, where $q < L < N$, the white noise variance estimator σ'^2 is given by Eq.(3.2.13)

2. Using the AR parameter estimates obtained from step 1 as the data (i.e.,{1,a[1],a[2],.....,a[L]}], use the autocorrelation method with an order of q to find {b[1],b[2],....,b[q]} as given by Eq.(3.2.14).

For complex data the same steps apply if we use the complex AR parameter estimators. For more information one can refer to [6], [9], [13], [14].

3.2.2 MODEL ORDER SELECTION

Before describing several model order estimators, it should be mentioned that the prediction error power which formed the basis for the AR model order estimators cannot be applied to MA process. This is because it decreases monotonically with the order of the linear predictor. No theoretical minimum of the prediction error power of a linear predictor occurs for an order equal to the MA model order. Equivalently, the reflection coefficient sequence is not zero after a certain index but is generally composed of a sum of damped exponentials.

Several techniques for MA model order determination are now described. None of the techniques have been thoroughly tested so that a comparison of their relative merits is not available. For an MA process is defined as

$$AIC(i) = N \ln \sigma'^2_i + 2i \tag{3.2.16}$$

where i is the assumed MA model order and σ'^2_i is the MLE of the white noise variance based on an ith order model. For possible model orders the AIC is computed and the model order yielding the minimum is chosen. If Durbin's algorithm is used to estimate the MA parameters, then all the lower order MA models are available. σ'^2 can be found by filtering the data with an estimate of the ith order inverse MA filter 1/B(z), which is guaranteed to be stable, and estimating the power at the output. A second approach which relies on the statistical properties of Durbin's method is to examine Q_{MIN} versus i. It can be shown that if the MA(i) model is correct, then

$$Q_{MIN} \sim \chi^2_{L-i}. \tag{3.2.17}$$

Hence, if Q_{MIN} is computed versus i, the appropriateness of each model can be tested by comparing Q_{MIN} to a threshold. A large value of Q_{MIN} indicates that the model order is probably incorrect. Assume that it is desired to test the appropriateness of various model orders at a 95% significance level. Thresholds for the orders are computed from

$$Pr\{\chi^2_{L-i} > \alpha_i\} = 0.005 \tag{3.2.18}$$

If Q_{MIN} for a given i falls below the computed threshold α_i, that value of i can be considered as a candidate for the correct model order. If several values of i produce Q_{MIN} which fall below the threshold, it is not clear which model order should be chosen. It is also possible that all values of i may produce Q_{MIN} that exceed their respective thresholds. This type of test may not produce a good indication of model order. It should be noted that the Q_{MIN} are readily available from the Levinson solution of Eq.(3.2.14). Specifically, for an ith order model

$$Q_{MIN} = (L+1)\left(\hat{f}_{aa}[0] + \sum_{k=1}^{i} b_i[k]\, \hat{f}_{aa}[0k]\right) - 1 \tag{3.2.19}$$

where the $b_i[k]$'s are the MA filter parameter estimates obtained from the Levinson recursion and

$$r'_{aa}[k] = \frac{1}{L+1} \sum_{n=0}^{L-k} a'[n]\, a'[n+k] \,. \tag{3.2.20}$$

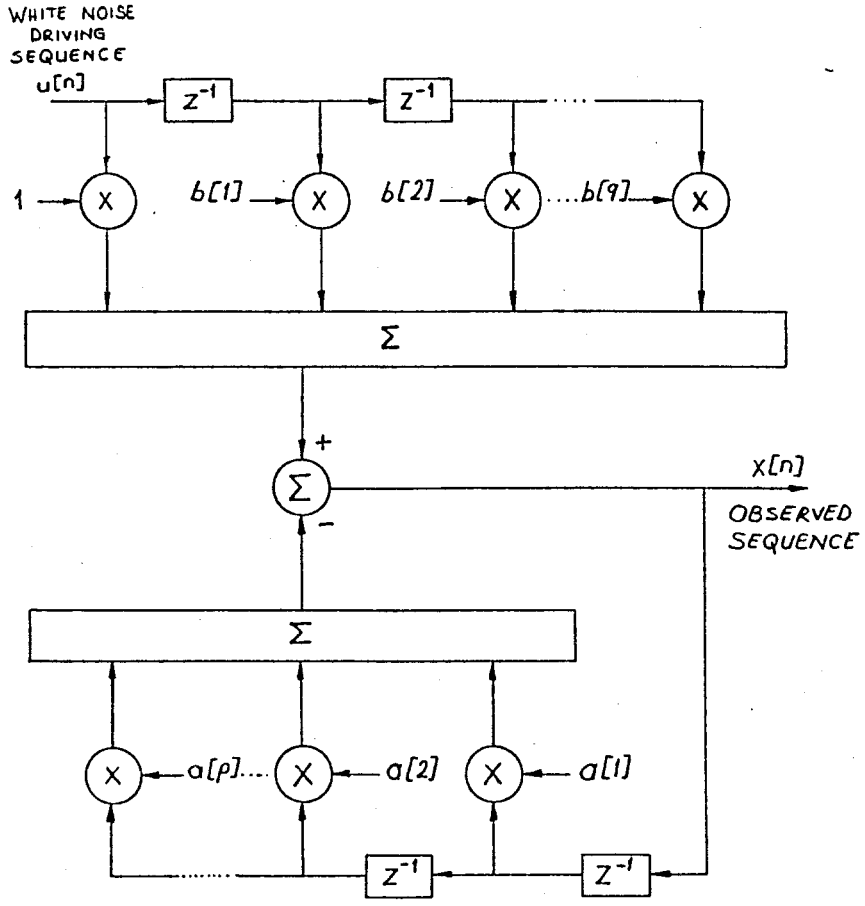
A third model order selection method tests the adequacy of an MA(i) model by testing whether the ACF samples for k > i are zero. Let

$$r''_{xx} = \left[r'_{xx}[i+1] \;\; r'_{xx}[i+2] \;\; \cdots \;\; r'_{xx}[i+M]\right]^T \tag{3.2.21}$$

Then the C_i is the covariance matrix for $\hat{r}_{xx}$ based on the assumption that x[n] is an MA(i) process. The threshold value is chosen to ensure with high probability that if the MA(i) is correct, the test will indicate this. It is not known how M should be chosen nor how the correct model among competing MA models may be chosen.

3.3. AUTOREGRESSIVE MOVING AVERAGE (ARMA) MODELLING

The autoregressive Moving Average (ARMA) model has more degrees of freedom than that the Autoregressive model (AR). Unlike the extensive repertoire of linear algorithms available to produce AR parameters and PSD estimators, there have been few algorithms produced for ARMA parameters and PSD estimators. This is due to primarily to the nonlinear nature required of algorithms that must simultaneously estimate the MA and AR parameters of the ARMA model. The nonlinear equations demonstrate the difficulty of estimating the ARMA parameters, even when the autocorrelation sequence is exactly known. Iterative optimization techniques based on maximum likelihood estimation (MLE) and related concepts are often used to solve nonlinear techniques [6], [9].



(a)

- Acquire Data
  - N samples
  - T seconds/sample
- Select AR and MA model orders
  - Parameters IP and IQ
- Estimate ARMA parameters
  - -Modified Yule Walker
  - -AKAIKE
  - -Least squares MYWE
- Compute ARMA PSD Estimate
- Order Closing

(b)

Figure 3.3.1. (a) ARMA model of random process
(b) Summary of ARMA modelling technique

These methods generally estimate the AR and then MA parameters separately, rather than jointly, as required for optimal parameter estimation. The AR parameters typically estimated first, independently of the MA parameters, by some variant of the Modified Yule-Walker equations. The MA parameters are then estimated assuming the AR parameters are known or have been previously estimated.

3.3.1 MAXIMUM LIKELIHOOD ESTIMATION

The MLE of the ARMA PSD is

$$P'_{ARMA}(f) = \frac{\sigma'^2 |1+b'[1]\exp(-j2\pi f) + \cdots + b'[q]\exp(-j2\pi fq)|^2}{|1+a'[1]\exp(-j2\pi f) + \cdots + a'[p]\exp(-j2\pi fp)|^2} \qquad (3.3.1)$$

where {a'[1], a'[2],....,a'[p], b'[1], b'[2],...,b'[q],σ^2 } are the MLEs of the ARMA parameters. This follows from the invariance principle. To obtain these MLEs we must maximize the likelihood function p(x[0], x[1],..., x[n-1]; a[1], a[2],...., a[p], b[1], b[2],...,b[q]) over the unknown parameters. This maximization will involve solving a set of highly nonlinear equations, even with several simplifying assumptions.

To derive the exact likelihood function is somewhat involved and lends little insight into a partial estimation procedure. An approximate likelihood function will be derived based on the following assumptions :

1. The data are real and Gaussian.
2. The data record N is large.
3. The poles and zeros are not close to unit circle.

The basic approach to determining an expression for the approximate likelihood function is to use an AR(∞) model of the ARMA process as described in previous section. Then the likelihood function already derived for an AR process can be applied to an ARMA process. Let the ARMA process be modeled as an AR(∞) process with filter coefficients {c[1],c[2],...}. A

finite model order approximation [i.e., an AR(L) model] will be a good approximation to the infinite order model if c[i] ≈ 0 for i > L. This requirements is equivalent to requiring that the impulse response of the filter with system function 1/B(z) be approximately zero for i > L. The approximate likelihood (actually, the conditional likelihood) function could be written as

$$p(\boldsymbol{x}|x[0],\ldots,x[L-1];\boldsymbol{a},\boldsymbol{b},\sigma^2) = \frac{1}{(2\pi\sigma^2)^{(N-L)/2}}\left[-\frac{1}{2\sigma^2}\sum_{n=L}^{N-1}\left(x[n]+\sum_{j=1}^{L}c[j]x[n-j]\right)^2\right] \tag{3.3.2}$$

where x = [x[0] x[1] .... x[N - 1]]^T. a,b are the vectors of the AR and MA filter coefficients, respectively, which depend on the c[j]'s. Note that for Eq.(3.3.2) to apply it was required that N be large and that the poles not be close to the unit circle. Now to maximize the likelihood function over a, b we must minimize

$$S_2(\boldsymbol{a},\boldsymbol{b}) = \sum_{n=1}^{N-1}\left(x[n]+\sum_{j=1}^{L}c[j]x[n-j]\right)^2 \tag{3.3.3}$$

S_2 is highly nonlinear in b but a quadratic function of a. As an example, consider an ARMA (1, 1) process. Then,

$$c[j] = (a[1]-b[1])(-b[1])^{j-1} \qquad j\geq 1 \tag{3.3.4}$$

so that,

$$S_2(\boldsymbol{a},\boldsymbol{b}) = \sum_{n=L}^{N-1}\left[x[n]+\sum_{j=1}^{L}(a[1]-b[1])(-b[1])^{j-1}x[n-j]\right]^2 \tag{3.3.5}$$

Because S_2 is quadratic in a, differentiation with respect to a and substitution of that unique value of a into S_2 will reduce S_2 to only a function of b. The resultant S_2 will be nonlinear in b and hence differentiation will produce a set of equations which if solved may only produce a local minimum. It is also possible to differentiate S_2 with respect to a and b

and solve the resulting nonlinear equations using a Newton - Raphson approach. This method is discussed as the Akaike estimator.

Assuming that S_e can be minimized to produce a',b' then the estimate of σ^2 is

$$\hat{\sigma}^2 = \frac{1}{N}\sum_{n=L}^{N-1}\left(x[n] + \sum_{j=1}^{L} \hat{c}[j]\,x[n-j]\right)^2 \qquad (3.3.6)$$

where the c'[j]'s are found as the impulse response of the filter with system function A'(z)/B'(z). It is seen that unfortunately in the ARMA case no simple set of Yule-Walker type equations result for the MLE. The use of the modified Yule - Walker equations for the estimation of the AR parameters as discussed before bears no resemblance to the MLE and hence cannot be expected to yield good estimates.

3.3.2. AKAIKE METHOD

The approximate MLE of the parameters of a real ARMA process can be determined as the minimum of a highly nonlinear function. In this section a Newton-Raphson iteration is employed to minimize this function. This approach, which was originally proposed by Akaike [16], is like all nonlinear optimization schemes, iterative in nature and therefore not guaranteed to converge. If convergence does occur, the minimum found may not be the global minimum. It is important to begin the iteration with an estimate that is close to the true parameter value, so that hopefully the global minimum will be found. For large data records local minima are not a problem [17], in that the log-likelihood function is approximately

quadratic in the ARMA parameters and therefore characterized by a single minimum. The approximate MLE of the ARMA filter parameters is obtained as the values that minimize

$$Q(a,b) = \frac{1}{N} S_2(a,b) = \int_{-\frac{1}{2}}^{\frac{1}{2}} I(f) \frac{|A(f)|^2}{|B(f)|^2} df \tag{3.3.7}$$

Akaike proposed using a Newton-Raphson iteration to find a zero of

$$[(\partial Q/\partial a)^T \quad (\partial Q/\partial b)^T]^T, \text{ or}$$

$$\begin{bmatrix} a_{k+1} \\ b_{k+1} \end{bmatrix} = \begin{bmatrix} a_{k+1} \\ b_{k+1} \end{bmatrix} - H^{-1}(a_k, b_k) \begin{bmatrix} \frac{\partial Q}{\partial a} \\ \frac{\partial Q}{\partial b} \end{bmatrix} \Bigg|_{a=a_k, b=b_k} \tag{3.3.8}$$

a_k, b_k are the kth iterates of the AR and MA filter parameter vectors, respectively. H(a,b) is the Hessian of Q, which is defined as

$$H(a,b) = \begin{bmatrix} \frac{\partial^2 Q}{\partial a \partial a^T} & \frac{\partial^2 Q}{\partial a \partial b^T} \\ \frac{\partial^2 Q}{\partial b \partial a^T} & \frac{\partial^2 Q}{\partial b \partial b^T} \end{bmatrix} = \begin{bmatrix} p \times q & p \times q \\ q \times p & q \times p \end{bmatrix} \tag{3.3.9}$$

the required partial derivatives are approximately where

$$r'_{yy}[k] = \frac{1}{N} \sum_{k=0}^{N-|k|-1} y[n]\, y[n+|k|]$$

$$r'_{zz}[k] = \frac{1}{N} \sum_{k=0}^{N-|k|-1} z[n]\, z[n+|k|] \tag{3.3.10}$$

$$\frac{\partial Q}{\partial a[k]} = -2\sum_{i=0}^{p} a[i]\, r'_{yy}[k-i] \qquad k=1,2,\ldots,p$$

$$\frac{\partial Q}{\partial b[l]} = -2\sum_{i=0}^{q} b[i]\, r'_{zz}[l-i] \qquad l=1,2,\ldots,q$$

$$\frac{\partial^2 Q}{\partial a[k]\,\partial a[l]} = 2r'_{yy}[k-l] \qquad \begin{matrix} k=1,2,\ldots,p \\ l=1,2,\ldots,p \end{matrix}$$

$$\frac{\partial^2 Q}{\partial b[k]\,\partial b[l]} = 2r'_{zz}[k-l] \qquad \begin{matrix} k=1,2,\ldots,q \\ l=1,2,\ldots,q \end{matrix}$$

$$\frac{\partial^2 Q}{\partial a[k]\,\partial b[l]} = 2r'_{yz}[k-l] \qquad \begin{matrix} k=1,2,\ldots,p \\ l=1,2,\ldots,q \end{matrix}$$

(3.3.11)

and

$$r'_{yz}[k] = \begin{cases} \frac{1}{N}\sum_{n=0}^{N-k-1} y[n]\, z[n+k] & \text{for } k \geq 0 \\ \frac{1}{N}\sum_{n=0}^{N-1} y[n]\, z[n+k] & \text{for } k < 0 \end{cases}$$

(3.3.12)

The sequences y[n], z[n] are defined as

$$y[n] = z^{-1}\left\{\frac{H(z)}{B(z)}\right\}$$

$$z[n] = z^{-1}\left\{\frac{H(z)A(z)}{B^2(z)}\right\}$$

(3.3.13)

where

$$H(z) = \sum_{n=0}^{N-1} x[n]\, z^{-n}$$

(3.3.14)

It is of interest to observe that the y[n], z[n] sequences are estimates of the processes arise in the derivation of the CR bound. Of course, this is not purely coincidence but can be shown to be a property of MLEs. Since the y[n], z[n] sequences are generated as the outputs of recursive filters, the initial conditions need to be specified. Akaike's approach sets these initial conditions equal to zero

on the premise that for large data records any transient introduced will be negligible. Clearly, this will not be the case when the zeros are near the unit circle since then the impulse response will be long.

The Akaike estimator may not yield minimum-phase filter estimates during the course of the iteration. If any iterate of the MA parameters causes B(z) to have a zero outside the unit circle, then due to the instability of 1/B(z), the y[n], z[n] sequences will grow large. We must therefore monitor the stability of the 1/B(z) filter. An approach to this problem would be to replace any zero outside the unit circle, say z_i by its conjugate reciprocal or $1/z_i^*$. However, a non-minimum phase filter would appear to be a deficiency of the algorithm, so that any ad hoc measure might lead to questionable results. Note that without the minimum-phase constraint it is possible to drive Q(a, b) to zero by making B(z) arbitrarily large. Assuming that B'(z) is minimum-phase, an alternative means of computing σ'^2, rather than to use Q(a',b'), is to use the approximately equivalently expression

$$\sigma'^2 = \frac{1}{N}\sum_{n=0}^{N-1} u'^2[n] \tag{3.3.15}$$

where

$$u'[n] = z^{-1}\left\{\frac{H(z)A'(z)}{B'(z)}\right\} \tag{3.3.16}$$

and the initial conditions of the recursive filter are arbitrarily assumed to be zero. In computing the new iterate of the ARMA parameters as per Eq.(3.3.8), we can avoid the inversion of the Hessian by rewriting the equations as

$$H(a_k, b_k)\begin{bmatrix} a_{k+1} \\ b_{k+1} \end{bmatrix} = H(a_k, b_k)\begin{bmatrix} A_k \\ b_k \end{bmatrix} - \begin{bmatrix} \frac{\partial Q}{\partial a} \\ \frac{\partial Q}{\partial b} \end{bmatrix} \tag{3.3.17}$$

and solving a set of simultaneous linear equations for new iterate. The Hessian is assured to be invertible since it is positive definite. A Cholesky decomposition can be used to solve Eq.(3.3.17). As mentioned previously, for good results it is necessary to provide a good set of initial estimates of the ARMA filter parameters. Any of the techniques described in this sections AR modelling estimates can be used for this purpose.

3.3.3. MODIFIED YULE-WALKER EQUATIONS

The ARMA estimation methods described in this section and the next are ad hoc in nature. They have arisen from the difficulties associated with the highly nonlinear MLE. Unlike the iterative techniques these methods are direct, relying on the modified Yule-Walker equations, but suboptimal. They do have the advantage that they are computationally simple.

Since these relationships hold when the ACF is known exactly, a reasonable approach is to replace the theoretical ACF samples by estimates and then solve the equations for the AR filter parameters. The MA parameters are subsequently found in a separate step. This leads to the following estimator for the AR filter parameters:

$$\begin{bmatrix} r_{xx}[0] & r_{xx}[-1] & \cdots & r_{xx}[-(M-1)] \\ r_{xx}[1] & r_{xx}[0] & \cdots & r_{xx}[-(M-2)] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[M-1] & r_{xx}[M-2] & \cdots & r_{xx}[0] \end{bmatrix} \begin{bmatrix} a'[1] \\ a'[2] \\ \vdots \\ a'[p] \end{bmatrix} = - \begin{bmatrix} r'_{xx}[q+1] \\ r'_{xx}[q+2] \\ \vdots \\ r'_{xx}[q+p] \end{bmatrix} \tag{3.3.18}$$

The ACF estimator may be either the biased or unbiased estimator. In general, a' will not be minimum-phase. Note that the matrix is Toeplitz since the elements along any NW to SE diagonal are the same, although not hermitian. Also, the matrix is not guaranteed to be nonsingular. Once the AR parameters have been estimated and x[n] filtered by A'(z) to produce an

approximate MA process, any of the methods of section 3.2 may be used to estimate the MA parameters.

The MYWE can be solved in an efficient manner using an extension of the Levinson recursion. The extension is implicit in the work of Trench [20], who showed how to invert a nonhermitian Toeplitz matrix.Recursive algorithm is initialized by

$$a_1[1] = -\frac{r_{xx}[q+1]}{r_{xx}[q]}$$
$$b_1[1] = -\frac{r_{xx}[q-1]}{r_{xx}[q]}$$
$$\rho_1 = (1-a_1[1]\,b_1[1])\,r_{xx}[q] \tag{3.3.19}$$

with the recursion for k = 2,3,....,p given by

$$a_k[k] = -\frac{r_{xx}[q+k] + \sum_{l=1}^{k-1} a_{k-1}[l]\,r_{xx}[q+k-l]}{\rho_{k-1}} \tag{3.3.20}$$

$$a_k[i] = a_{k-1}[i] + a_k[k]\,b_{k-1}[k-i] \qquad i = 1,2,....,k-1 \tag{3.3.21}$$

If k = p, exit; if not, continue.

$$b_k[k] = -\frac{r_{xx}[q-k] + \sum_{l=1}^{k-1} b_{k-1}[l]\,r_{xx}[q-k-l]}{\rho_{k-1}} \tag{3.3.22}$$

$$b_k[i] = b_{k-1}[i] + b_k[k]\,a_{k-1}[k-i] \qquad i=1,2,.....,k-1 \tag{3.3.23}$$

$$\rho_k = (1-a_k[k]\,b_k[k])\,\rho_{k-1} \tag{3.3.24}$$

The solution is a[k] = a_p[k], k=1,2,....,p. It is interesting to note that if q = 0 so that the MYWE reduce to the Yule-Walker equations, the algorithm reduces to the Levinson recursion. In this case, b_k[i] = a^*_k[i], making the computation of b_1[1] and the recursion Eq.(3.3.22) and Eq.(3.3.23) redundant. Upon examination of Eq.(3.3.20) and Eq(3.3.21), it is apparent that for the solution to exist it is required that $\rho_i \neq 0$ for i = 0,1,...,p - 1, where $\rho_0 = r_{xx}[q]$. This is also obvious if we note that [20]

$$det\ (R'_{xx}) = \prod_{i=0}^{p-1} P_i \tag{3.3.25}$$

The statistics of the AR filter parameter estimator obtained from the MYWE have been derived for large data records and for real Gaussian data by Gersh [18]. He has shown that the estimator is asymptotically (as N→∞) unbiased or $\mathcal{E}$[a']=a and that the covariance matrix is

$$c_a = \mathcal{E}[(a'-\mathcal{E}(a'))(a'-\mathcal{E}(a'))^T] = \frac{\sigma^2}{N-p-q} R'^{-1}_{xx} \mathcal{R} R'^{-T}_{xx} \tag{3.3.26}$$

where R'_{xx} is given in Eq.(3.3.18) with the ACF estimates replaced by their true values and

$$\mathcal{R} = \sum_{k=-p}^{q} \left(1 - \frac{|k|}{N-p-q}\right) c_k\, R_{xx}[k] \tag{3.3.27}$$

and

$$c_k = \sum_{i=0}^{q-|k|} b[i]\, b[i+|k|] \tag{3.3.28}$$

$$R_{xx}[k] = \begin{bmatrix} r_{xx}[k] & r_{xx}[k-1] & \cdots & r_{xx}[k-p+1] \\ r_{xx}[k+1] & r_{xx}[k] & \cdots & r_{xx}[k-p] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[k+p-1] & r_{xx}[k+p-2] & \cdots & r_{xx}[k] \end{bmatrix} \tag{3.3.29}$$

Note that in the pure AR case in which q=0, it follows that

$$\begin{aligned} \mathfrak{R} &= c_0 R_{xx}[0] = R_{xx} \\ \hat{R}_{xx} &= R_{xx} \end{aligned} \tag{3.3.30}$$

so that Eq.(3.3.26) becomes

$$C_a = \frac{\sigma^2}{N-p} R_{xx}^{-1} \approx \frac{\sigma^2}{N} R_{xx}^{-1} \tag{3.3.31}$$

Also, it can be shown that for pure AR process the use of MYWE, which involves higher order samples of the ACF, produces poorer estimates than those obtained using the Yule Walker equations [8].

The performance of the MYWE approach varies greatly. For some processes the estimates will be quite accurate, while for others they will be very poor.

The statistical properties of the spectral estimator based on the MYWE have been derived by Sakai and Tokumaru [19]. The results indicate that large variabilities are to be expected for frequencies where actual PSD is small.

3.3.4. LEAST SQUARES MODIFIED YULE-WALKER EQUATIONS

In an attempt to reduce the variance of the MYWE estimator has suggested utilizing more of the available

equations. Later, Cadzow [20] applied this idea to the spectral estimation problem. Since for an ARMA(p,q) process

$$r_{xx}[k] = -\sum_{l=1}^{p} a[l]\, r_{xx}[k-l] \qquad k \geq q+1$$

(3.3.32)

the choices of the p equations corresponding to k= q+1, q+2,...,q+p in Eq.(3.3.19) is an arbitrary one. It can be shown that there is information in the ACF at higher order samples . To use this information, assume that the highest sample of the ACF that can be accurately estimated is $r_{xx}[M]$, and consider the following theoretical equations:

$$\begin{bmatrix} r_{xx}[q+1] \\ r_{xx}[q+2] \\ \vdots \\ r_{xx}[M] \end{bmatrix} = - \begin{bmatrix} r_{xx}(q) & r_{xx}[q-1] & \cdots & r_{xx}[q-p+1]) \\ r_{xx}[q+1] & r_{xx}[q] & \cdots & r_{xx}[q-p+2]) \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[M-1] & r_{xx}[M-2] & \cdots & r_{xx}[M-p] \end{bmatrix} \begin{bmatrix} a[1] \\ a[2] \\ \vdots \\ a[p] \end{bmatrix}$$

$$or$$

$$r = -Ra$$

(3.3.33)

R is of dimension (M - q) x p. Assuming the theoretical ACF is replaced by an estimate the equations will no longer be satisfied. To account for estimation errors in the ACF the equations should be expressed as

$$r' = -R'a + e$$

(3.3.34)

where r', R' correspond to the estimators of r, R and the error term e is zero if r'= r, R' = R. It is recommended that the unbiased ACF estimator be used in Eq.(3.3.34) since then the average equation error is zero, or

$$\mathcal{E}(e) = \mathcal{E}(r) + \mathcal{E}(R')\, a = r + Ra = 0 .$$

(3.3.35)

The form of Eq.(3.3.34) immediately suggests the use of a least squares (LS) estimator. LS estimator of the AR parameters is

$$a' = -(R'^H R')^{-1} R'^H r' \tag{3.3.36}$$

where

$$R' = \begin{bmatrix} r'_{xx}[q] & r'_{xx}[q-1] & \cdots & r'_{xx}[q-p+1] \\ r'_{xx}[q+1] & r'_{xx}[q] & \cdots & r_{xx}[q-p+2] \\ \vdots & \vdots & \ddots & \vdots \\ r'_{xx}[M-1] & r'_{xx}[M-2] & \cdots & r'_{xx}[M-p] \end{bmatrix} \tag{3.3.37}$$

$$r' = \left[r'_{xx}[q+1] \;\; r'_{xx}[q+2] \;\; \cdots \;\; r'_{xx}[M] \right]^T \tag{3.3.38}$$

This estimator of the AR parameters of an ARMA process is sometimes referred to as an equation error modelling approach. Henceforth it will be termed the last squares modified Yule-Walker equation (LSMYWE) estimator. It should be emphasized that no optimality properties of the LS approach apply to this problem. This is because R'is not a constant matrix nor does have the statistical properties necessary to claim optimality. Because R'"R' is usually positive definite it is invertible by typical routines such as the Cholesky decomposition or myriad of other techniques developed for LS problems. In general, a'will not be minimum-phase. The set of equations given by Eq.(3.3.33)

$$r'_{xx} = -\sum_{k=1}^{p} a[k]\, r'_{xx}[n-k] + e[n] \qquad n \geq q+1 \tag{3.3.39}$$

are similar in structure to the AR time series model

$$x[n] = -\sum_{k=1}^{p} a[k]\, x[n-k] + u[n] \tag{3.3.40}$$

The LSMYWE estimator can therefore be interpreted as the implementation of the covariance method or linear prediction applied to the "data" sequence $\{r'_{xx}[q - p + 1], r'_{xx}[q-p+2], \ldots, r'_{xx}[M]\}$. This suggests application of other AR techniques to ARMA estimators.

3.3.5. MODEL ORDER SELECTION

For an ARMA time series the reflection coefficient sequence is infinite in extent so that the prediction error power is always decreasing. This is in contrast to an AR time series, in which the prediction error power first reaches its minimum at the correct model order. Hence model order determination approaches based on the linear prediction error power cannot be used for an ARMA process. Some methods that have been proposed for ARMA model order estimation are now described. The AIC as described in section (3.1.3) for AR model order determination and in section (3.2.2) for MA model order selection can also be used for the real ARMA case if we define [15]

$$AIC(i,j) = N \ln \sigma'^2_{ij} + 2(i+j) \tag{3.3.41}$$

where is the assumed AR model order, j is the assumed MA model order, and σ'^2_{ij} is the MLE of σ^2 obtained under the assumption that x[n] is an ARMA(I,j) process. As usual, the AIC is computed for all model orders of interest and the orders that minimize it are chosen. Another approach is to filter x[n] with the estimated inverse filter A'(z)/B'(z) to generate an estimate u'[n] of the white noise process. If the correct order has been chosen, u'[n] will be approximately white noise and hence the estimated ACF should be approximately zero for all lags except the zeroth one. It can be shown that if an ARMA(i,j) model is correct, then for a real process

$$Q = N\sum_{k=1}^{M}\left(\frac{r'_{uu}[k]}{r'_{uu}[0]}\right)^2 \qquad (3.3.42)$$

is distributed according to a χ^2_{M-1-J} random variable $r'_{xx}[k]$ is the biased estimator of the ACF of $u'[n]$ given by

$$r'_{uu}[k] = \frac{1}{N}\sum_{n=0}^{N-1-k} u'[n]\,u'[n+k] \qquad (3.3.43)$$

M should be the effective impulse response length of the filter with system function B(z)/A(z). If the model is incorrect, Q will be large. We might compute Q over several possible model orders and discard models that had inflated Q's. If all the models but one had inflated values, then by the process of elimination the remaining model could be chosen. Otherwise, further tests would be necessary. Finally, a model order selection rule based on the modified Yule - Walker equations has been proposed for AR model order determination of an ARMA process by Chow [21]. If we examine the i x i matrix R'_{xx}, where

$$R_{xx} = \begin{bmatrix} r_{xx}[q+1] & r_{xx}[q] & \cdots & r_{xx}[q-i+2] \\ r_{xx}[q+2] & r_{xx}[q+1] & \cdots & r_{xx}[q-i-3] \\ \vdots & \vdots & \ddots & \vdots \\ r_{xx}[q+i] & r_{xx}[q+i-1] & \cdots & r_{xx}[q+1] \end{bmatrix} \qquad (i\ X\ i) \qquad (3.3.44)$$

for an assumed model order of i, then for i > p, the true AR model order, the matrix will be singular. This follows from the modified Yule - Walker equations

$$\sum_{l=1}^{p} a[l]\,r_{xx}[k-l] = -r_{xx}[k] \qquad k \geq q+1 \qquad (3.3.45)$$

which imply that the columns of R'_{xx} will be linearly dependent. As an example, if i = p + 1, then $R'_{xx}=[r'_{q+1} \quad r'_{q} \quad \ldots \quad r'_{q+1-p}]$, where $r'_{k} = [r_{xx}[k] \quad r_{xx}[k+1] \ldots r_{xx}[k+p]]^{T}$. The columns r'_{i} are linearly dependent since

$$\sum_{i=0}^{p} a[i]\, r_{q+1-i}=0 \tag{3.3.46}$$

which follows from the modified Yule - Walker equations. We can monitor det (R'_{xx}) until it becomes sufficiently small for some i. Note that we need to know q or at least be able to assume that q is not larger than some value q_{max}. In the latter case q_{max} is used in Eq.(3.3.44) and the actual value of q can be determined by filtering x[n] by A'(z) once p has been determined and the AR parameters estimated.

3.4. INPUT-OUTPUT IDENTIFICATION APPROACHES

A class of suboptimal ARMA estimation algorithms have been proposed which rely on estimation of the driving white noise u[n]. If u[n] were known, we would have knowledge of the input as well as the output. Then the many estimators developed for system identification which require only the solution of linear equations could be used.

Specifically, if we examine the autocorrelation function, it becomes clear that the nonlinear nature of the Yule-Walker equations is due to the unknown cross-correlation between the input and output. If however, we knew u[n], the ARMA parameters could be estimated as the solution of a set of linear equations. Pade' approximation is one of these methods developed. It has an important place in the literature because of the fact that systems which have higher order transfer function can be realized with Pade' approximation. The detailed information on Pade' approximation is given below.

3.4.1. PADE' APPROXIMATION METHOD

A number of methods for the reduction in order of high-order systems have been proposed, based on expanding the system transfer function G(s) into a continued fraction and truncating it to get the reduced-order transfer function R(s) [22], [23], [24]. Others have proposed a method of reduction based on the fitting of the time-moments of the system and its reduced model. Shamash [24] has shown that for the case of rational transfer functions, the continued fraction methods were a special case of the time-moments method, which is equivalent to the Pade approximation method. The continued fraction and time-

moments techniques have a number of very useful advantages, such as computational simplicity, the fitting of the time-moments, and the steady-state values of the output of system and model are the same for inputs of the form $a_i t^i$. However, they do have a very serious disadvantage and that is the fact that the reduced-order model may be unstable even though the original high-order system is stable. Shamash [25] introduced a method of reduction based on the retention of poles of the high-order system in the reduced model, and the concept of Pade approximation about more than one point. The method preserves stability in the sense that the model is stable if the system is stable. A novel method of reduction based on the Routh stability criterion which was used to compute the denominator of R(s) was introduced. The numerator of R(s) is computed by expanding the numerator of the system transfer function into a sum and product of continued fractions, which are then truncated. The method was applied the reduction of single-input/single-output systems.

3.4.2. THEORY OF THE PADE' APPROXIMATION

A pade approximation is the ratio of two polynomials constructed from the coefficients of the Taylor series expansion of a function. Since it provides an approximation to the function throughout the whole complex plane, the study of Pade approximants is one of the mathematical approximation theory. It has wide applicability to those areas of knowledge that involve analytic techniques.

Let f be a formal power series. Pade' approximants are rational functions whose expansion in ascending powers of the variable coincides with f as far as possible, that is, up to sum the degrees of the numerator and denominator. The numerator and the denominator of a Pade approximant are completely determined by this condition.

Let $F(s) = P_m(s)/ Q_n(s)$, where $P_m(s)$ and $Q_n(s)$ are polynomials with real coefficients and nominal degrees m and n, respectively (that is, the actual degree may be lower). Then [m/n] is a full Pade approximant of F(s) if the power expansion of [m/n] is identical to that of F(s) up to and including terms of order s^{m+n}. In this case, we define F(s) = [m/n]. If the matching of terms is of lower than s^{m+n}, then it is a partial Pade approximant.

The relation between the coefficients of the Taylor series expansion of a function and the values of the function is both a profound mathematical question and an important practical one. It is basic to the study of mathematical analysis, and to the practical calculation of mathematical models of nature throughout much of physical and biological science. If the Taylor series expansion converges absolutely, then it uniquely defines the value of a function which is differentiable an arbitrary number of times. Conversely, if a function is differentiable an arbitrary number of times, it uniquely defines the Taylor series expansion. Practically, we are approximating the function by longer and longer polynomials. This approach, however, has undesirable limitations for practical calculations.

If we try to explain the Pade approximant by mathematical formulas; suppose that we are given a power series $\Sigma c_i z^i$ representing a function f(z), so that

$$f(z) = \sum_{i=0}^{\infty} c_i z^i \qquad (3.4.1)$$

A Pade approximant is a rational fraction

$$[L/M] = \frac{a_0 + a_1 z + \ldots + a_L z^L}{b_0 + b_1 z^1 \ldots + b_M z^M} \qquad (3.4.2)$$

which has a Maclaurin expansion which agrees with autocorrelation function as far as possible. Notice that in Eq.(3.4.2) there are L + 1 numerator coefficients and M + 1 denominator coefficients. There is a more or less irrelevant common factor between them, and for definiteness we take $b_0=1$. This choice turns out to be an essential part of the precise definition, and Eq.(3.4.2) is our conventional notation with this choice for b_0. So there are L+1 independent numerator coefficients and M independent denominator coefficients, making L+M+1 unknown coefficients in all. This number suggests that normally the [L/M] ought to fit the power series Eq.(3.4.1) through the orders 1, z, z^2,...z^{L+m}. In the notation of formal power series,

$$f(z)-\sum_{i=0}^{\infty} c_i z^i=\frac{a_0+a_1 z+\ldots+a_l z^L}{b_0+b_1 z+\ldots+b_M z^M}+O(z^{L+M+1}) \tag{3.4.3}$$

Cross multiplying Eq.(3.4.3), we find that

$$(b_0+b_1 z+\ldots+b_M z^M)(c_0+c_1 z+\ldots) - a_0+a_1 z+\ldots+a_L z^L+O(z^{L+m+1}) \tag{3.4.4}$$

Equating the coefficients of $z^{L+1}, z^{L+2}, \ldots, z^{L+m}$, we find

$$\begin{aligned} &b_M c_{L-M+1}+b_{M-1}c^{L-M+2}+\cdots+b_0 c_{L+1}=0,\\ &b_M c_{L-M+2}+b_{M-1}c_{L-M+3}+\cdots+b_0 c_{L+2}=0,\\ &\qquad\vdots\\ &b_M c_L \quad +b_{M-1}c_{L-1}\ +\cdots+b_0 c_{L+M}=0. \end{aligned} \tag{3.4.5}$$

If j < 0, we define $c_j = 0$ for consistency. Since $b_0 = 1$, Eq. (3.4.5) become a set of M linear equations for the M unknown denominator coefficients

$$\begin{bmatrix} c_{L_M+1} & c_{L-M+2} & c_{L-M+3} & \cdots & c_L \\ c_{L-M+2} & c_{L-M+3} & c_{HXM+4} & \cdots & c_{L+1} \\ c_{L-M+3} & c_{L-M+4} & c_{L-M+5} & \cdots & c_{L+2} \\ \vdots & \vdots & & & \vdots \\ c_L & c_{L+1} & c_{L+2} & \cdots & c_{L+M+1} \end{bmatrix} \begin{bmatrix} b_M \\ b_{M-1} \\ b_{M-2} \\ \vdots \\ b_1 \end{bmatrix} = - \begin{bmatrix} c_{L+1} \\ c_{L+2} \\ c_{L+3} \\ \vdots \\ c_{L+M} \end{bmatrix} \tag{3.4.6}$$

from which the b_i, may be found. The numerator coefficients, a_0, $a_1,\ldots,a_L$, follow immediately from Eq.(3.4.4) by equating the coefficients as $1, z, z^2,\ldots,z^L$.

$$\begin{aligned} a_0 &= c_0, \\ a_1 &= c_1 + b_1 c_0, \\ a_2 &= c_2 + b_1 c_1 + b_2 c_0, \\ &\vdots \\ a_L &= c_L + \sum_{i=1}^{\min(L,M)} b_i c_{L-i}. \end{aligned} \tag{3.4.7}$$

Thus Eq.(3.4.6) and Eq.(3.4.7) normally determine the Pade numerator and denominator and are called the Pade equations; we have constructed an [L/M] Pade approximant which agrees with $\Sigma c_i z^i$ through order z^{L+M}. Because the starting point of these manipulations is the given power series, we do not ever need to know about the existence of any function f(z) with $\Sigma c_i z^i$ as its Maclaurin series as in Eq.(3.4.1). Of course, we expect that a well-chosen sequence of Pade approximants will normally approximate a function f(z) with the Maclaurin expansion $\Sigma c_i z^i$, but it is important to distinguish between problems of convergence of Pade' approximants and problem of construction of Pade' approximants. Given the power series, Eq.(3.4.7) shown how the Pade' approximants are constructed.

Every power series has a circle of convergence $|z| = R$. If $|z| < R$, the series converges, and if $|z| > R$, it does not. If $R=\infty$ the power series represents an analytic function (Function analytic everywhere is called entire) and the series may be summed directly for any value of z to yield the function f(z). If R = 0, the power series is undoubtedly formal. It contains information about f(z), but just how this information is to be used is not immediately clear. However, if a sequence of Pade' approximants of the formal power series converges to a function g(z) for $z \in D$, then we may reasonably conclude that g(z) is a function with the given power series. If the given power series converges to the same function for $|z| < R$ with $0 < R < \infty$, then a sequence of Pade approximants may converge for $z \in D$ where D is a domain larger than $|z| < R$. We will then have extended our

domain of convergence. This is frequently a practical approach to what amounts to analytic continuation. The method of expansion and reexpansion due to Weierstrass is more suited to principle than practice.

There is one feature of the calculation of Pade approximations to be emphasized at the start-these calculations require more numerical accuracy than one might at first expect. The Pade approximant exploits the differences of the coefficients to do its long-range extrapolation, and so the differences must all be accurate. For more information one can refer to [26], [27].

3.4.3. APPLICATION OF PADE' APPROXIMATION TO SYSTEM IDENTICIATION

Although ARMA methodologies are sophisticated and through forms of system identification and modeling; one always hopes for a more simplified approach. The aim of our work has been this goal. ARMA modelling normally ends up to be a higher ordered rational transfer function model from which the linear system identification is based upon [14]. The question arises, is there a reduced order model based on or related to an ARMA which will yield as good if not better results? If so, then the reduced order model will replace the ARMA model to estimate further a prior output forecasting from the identified system. Pade' approximations are such a rational form of modelling to which this end is met.

In employing the Pade' approach, one has to assume two possible approaches. These are:

(1). a. achieve ARMA model

b. reduce to series expansion,

c. locate dominate poles of series to assure stability,

d. calculate Pade' table coefficients;

(2). a. determine system impulse

b. locate dominate poles of response series

c. calculate Pade'.

Thus far, the prior work on which more research work is currently based upon is approach (2) [25]. In approach (1), there is an added step to achieve ARMA, whereas in prior work only order reduction was desired for a known rational model [25].

One of the methodologies is employed by Shamash this approach, Shamash tries to fit a reduced order model by allowing retention of dominant poles [25].

Lets have a look at this methodology briefly. Also this methodology is expressed in Figure (3.4.1).

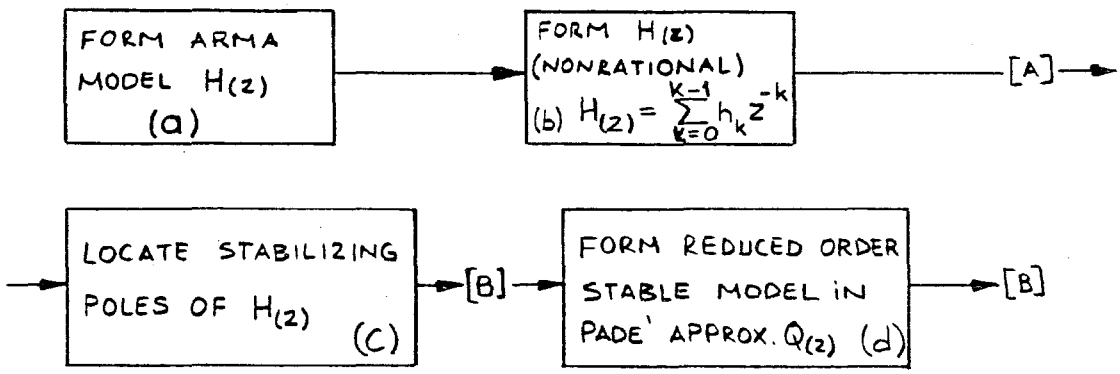


Figure 3.4.1. Pade' approximation Shamash approach

3.4.3.1. Pade' Approximation and Dominant Mode Reduction (First Approach-Shamash Approach):

Consider the following high order system transfer function

$$G(s) = \frac{d_0+d_1 s+d_2 s^2+\cdots+d_{n-1}s^{n-1}}{(s+s_1)(s+s_2)\cdots(s+s_n)}$$

$$= \frac{d_0+d_1 s+d_2 s^2+\cdots+d_{n-1}s^{n-1}}{e_0+e_1 s+e_2 s^2+\cdots+e_n s^n}$$

(3.4.8)

G(s) can be expended into a power series about s=0 of the form

$$G(s) = c_0+c_1 s+c_2 s^2+\cdots$$

(3.4.9)

where

$$\left.\begin{aligned} c_0 &= \frac{d_0}{e_0} \\ c_k &= \frac{1}{c_0}\left[d_k - \sum_{j=1}^{k} e_j c_{k-j}\right], \quad \forall k>0 \end{aligned}\right\}$$

(3.4.10)

with

$$d_k = 0, k>n-1$$

(3.4.11)

The e_i are directly proportional to the time-moments of the system, assuming the system is asymptotically stable, and throughout this section we will refer to them as the time-moments [28].

Assume that a reduced model R(s), of order k, is required which retains the pole at s = - s_1, say. Let

$$R(s) = \frac{a_0+a_1 s+a_2 s^2+\cdots+a_{k-1}s^{k-1}}{b_0+b_1 s+b_2 s^2+b_{k-1}s^{k-1}+s^k}$$

(3.4.12)

The orders of the numerator of R(s) and G(s) have been assumed to be one less than the denominators to simplify the notation. Then for R(s) to be a Pade approximant of G(s) we have Eq.(3.4.7).

$$\begin{aligned} a_0 &= b_0 c_0 \\ a_1 &= b_0 + b_1 c_0 \\ &\cdots \\ 0 &= b_0 c_{2k-2} + b_1 c_{2k-1} + \cdots + c_{k-1} \\ 0 &= b_0 c_{2k-1} + b_1 c_{2k-2} + \cdots + c_k \end{aligned}$$

(3.4.13)

But since R(s) is to have a pole at s = $-s_1$, then using the concept of Pade approximation about more than one point, the last Eq.(3.4.13)is replaced by the following equation

$$0 = b_0 - b_1 s_1 + b_2 s_1^2 - \cdots + (-1)^k s_1^k$$

(3.4.14)

Hence these equations can be solved for the coefficients b_i, a_i (i=0,...,k-1) of Eq.(3.4.13).

Now suppose that the reduced order model R(s) is required to retain the k dominant poles (the k poles nearest the origin) of the high-order system. Further suppose that the k dominant poles are known. R(s) can then be written as

$$R(s) = \frac{a_0 + a_1 s + \cdots + a_{k-1} s^{k-1}}{(s+s_1)(s+s_2)+\cdots(s+s_k)}$$

$$= \frac{a_0 + a_1 s + \cdots + a_{k-1} s^{k-1}}{b_0 + b_1 s + \cdots + b_{k-1} s^{k-1} + s^k}$$

(3.4.15)

where the b_i(i=0,1,...,k-1) may be computed in terms of $s_1, \ldots, s_k$.

Then if R(s) is to approximate G(s), in the Pade sense, about s=0, then the a_i(i=0,1,...,k-1) may be determined using the first k equations of (3.4.13)

So far it has been assumed that the dominant poles of the system are known, which in most cases is not necessarily true. This where Koening's theorem, and its generalization are of great use, since by using them we can determine the number of dominant poles and their locations vary easily.

THEOREM 3.1

Let

$$f(s)=c_0+c_1s+c_2s^2+\cdots, \quad c_i \ real \wedge \ c_0\neq 0 \tag{3.4.16}$$

be meromorphic for |s| < R , and in this disc let it have just one simple pole s=r. If

$$|r|<\sigma R<R \tag{3.4.17}$$

then

$$\frac{c_\nu}{c_{\nu+1}}=r+O(\sigma^{\nu+1}) \tag{3.4.18}$$

THEOREM 3.2

Let f(s), given in Eq.(3.4.16), be meromorphic for |s| < R, and let it have exactly p poles $r=_1$, $r_2,\ldots,r_p$, not necessarily distinct in this disc. Let

$$0<|r_1|\leq|r_2|\leq|r_3|\leq\cdots\leq|r_p|<\sigma R<R \tag{3.4.19}$$

and let

$$\Psi(s)=(1-r_1^{-1}s)\,(1-r_2^{-1}s)\cdots(1-r_p^{-1}s)$$

$$=1+a_1s+a_2s^2+\cdots+a_ps^s \tag{3.4.20}$$

Finally, let the dominator of the [v,p] Pade' approximant be

$$K_v(s) = 1 + \alpha_1^{(v)} s + \alpha_2^{(v)} + \cdots + \alpha_p^{(v)} s^p \tag{3.4.21}$$

Then,

$$\alpha_1^{(v)} = \alpha_{1_1} O(\sigma^v) \tag{3.4.22}$$

$$K_v(s) = \Psi(s) + O(\sigma^v) \tag{3.4.23}$$

For the case when p=1, theorem p=1, theorem 2 reduces to theorem 1.

Thus to reduce a high-order transfer function, it is first expanded into a power series, then theorems 1 and 2 are used to determine the numerator dynamics of the reduced order transfer function. The amount of computation involved in using this method is the same as that required for ordinary Pade' approximation except perhaps more coefficients of the series Eq. (3.4.11) may have to be computed.

It should be noted that common poles and zeros in G(s) are automatically cancelled when using this method of reduction, and have no effect on the reduced model.

In this case when the system is described in state-vector form,

$$x' = Ax + Bu$$
$$y = Cx + Du \tag{3.4.24}$$

The system transfer function is given by

$$\begin{aligned} G(s) &= C(sI-A)^{-1}B+D \\ &= (CA^{-1}B+D)+CA^{-2}Bs+CA^{-3}B^2+\cdots \\ &= c_0+c_1s+c_s^2+c_3s^3+\cdots \end{aligned} \tag{3.4.25}$$

where

$$\begin{aligned} c_0 &= (CA_{-1}B+D) \\ c_i &= CA_{-(i+1)}B, \qquad \forall i>0 \end{aligned} \tag{3.4.26}$$

Hence the reduction algorithm may be applied to expansion Eq.(3.4.25), where the coefficients are determined using Eq.(3.4.26).

If the system being modelled is unstable, then it is important that the reduced model should be unstable as well. Hence the unstable model of G(s) must be retained in the reduced model. Koenig's theorem, and its generalization, may be used to compute the unstable modes as follows:

Given G(s), the following transformation is made

$$s=\frac{z-1}{z+1} \tag{3.4.27}$$

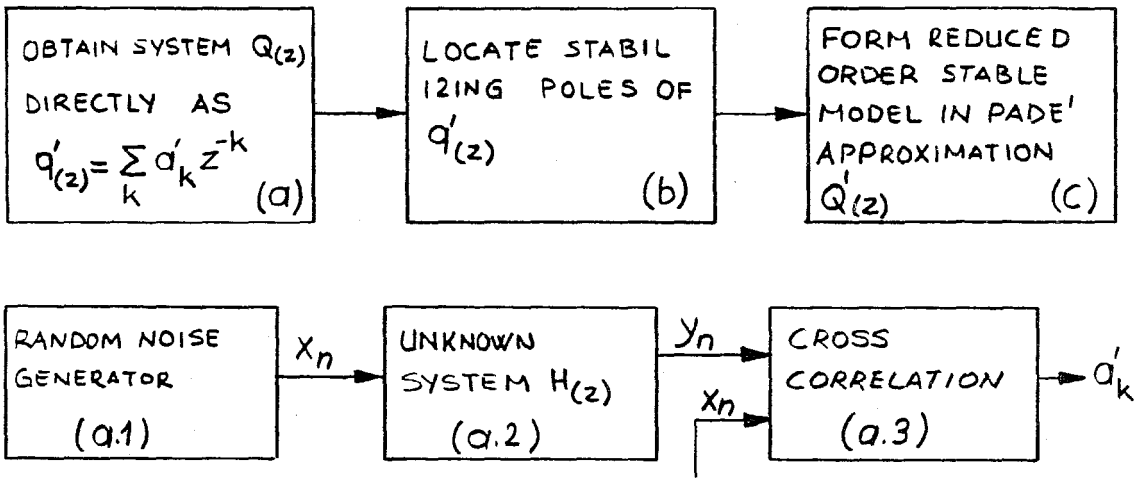
to get G(z). The unstable poles of G(z) in the form

$$G(z) = d_0+d_1z^{-1}+d_2z^{-2}+\cdots \tag{3.4.28}$$

Then applying Theorems 1 and 2 Eq.(3.4.28) we get all the large magnitude poles of G(z), which in this case will be the poles outside the unit circle. Having computed the unstable poles, the coefficients of R(s) are computed as before [25], [29].

3.4.3.2. Pade' Approximation Without Dominant Mode Reduction (Second Approach- Biyiksiz Approach)

This approach is based on research by Biyiksiz [30], where upon direct Pade' approximation of a reduced order model Q(z) is utilized in the system identification. Fig. (3.4.2) depicts this approach



$$a'_k = 1/\sigma_x^2 \, R_{yx}(k)$$

$$R'_{yx}(k) = 1/N \sum_{n=0}^{N-1-k} y_{n+k} \, x_n \; ; \quad k = 0, 1, \ldots, N/2$$

Figure 3.4.2. Pade' approximation Biyiksiz approach

Step (a) in Figure (3.4.1) is elaborated in Figure (3.4.2). The computation in Figure (3.4.2) has shown less computational effort than method 1, step (a).

The Pade' approximation Q(z) approach in its basics implies the following

$$q'(z)-\sum_{k=0}^{N-1} q'_k z^{-k} \approx Q'(z) - A'(z)/B'(z)$$

$$-\frac{\sum_i a'_i z^{-i}}{\sum_j b'_j z^{-j}} \qquad I+J \geq N$$

(3.4.29)

and q'(z) - Q'(z) ≈ 0 , from which the coefficients a'$_i$, b'$_j$ of Q'(z) are determined from

$$\begin{aligned}
&a'_0 - b'_0 q'_0 \\
&a'_1 - b'_0 q'_1 + b'_1 q'_1 \\
&\vdots \\
&0 - b'_0 q'_{2k-2} + b'_1 q'_{2k-3} + \cdots + q'_{k-1} \\
&0 - b'_0 q'_{2k-1} + b'_1 q'_{2k-2} + \cdots + q_k
\end{aligned}$$

(3.4.30)

When a Q'(z) is realized, the following has been observed:

1. When q'(z) is stable, Q'(z) need not be stable or

2. When q'(z) is stable Q'(z) may be stable and thus (1) and (2) yield untrue, reduced order, models of q(z). Case (1) is the more important real world problem. As a guideline it has been observed that unlike achieved ARMA model a Pade' approximation result in an unstable model even if the real system is stable or vice versa. This is especially true if the initial part of the system step $x_n=h_n+u_n$ response has a large overshoot. To overcome this, Shamash has implemented some rather novel methods based on Koenig's theorem, and its generalization[25], [29],[31],[32]. Basically, the two ways of approaching stability are (1) locate stabilizing poles at z = 0 or z = ∞ of q'(z) or (2) locate k dominate poles. When (1) is applied a trial and error approach results. When (2) is applied a longer computation results but stabilization is guaranteed. Biased on Konig's theorem, and its generalization, an idea of the k dominate poles may be achieved [25]. Thus

$$\lim_{k\to\infty}\frac{q'_k}{q'_{k+1}}=\alpha_k\to z_1,$$

$$\lim_{k\to\infty}\frac{(\alpha_k-\alpha_{k+1})}{(\alpha_{k+1}-\alpha_{k+2})}=\beta_k\to z_2$$

$$\lim_{k\to\infty}\frac{(\beta_{k+1}-\beta_{k+2})}{(\beta_{k+2}-\beta_{k+3})}=\delta_k\to z_3$$

(3.4.31)

for p poles assumed in reduced model $Q(z) = A'_{J-1}(z)/B'_{J-1}(z)$. As an alternative to limiting approach to determine z_p, one may assume a reduced model form as

$$Q'[v,J-1]=Q'_{J-1}(z)=\frac{A'_v(z)}{B'_{v_{J-1}}(z)}$$

$$=\frac{A'_v(z)}{\sum_{j=0}^{J-1}b_jz^{-j}}=\frac{A'_v(z)}{1+\sum_{j=1}^{J-1}b'_jz^{-j}}$$

(3.4.32)

$v = 0,1 \ldots$ and for successive iterations of v, b'_j coefficients may be evaluated, in conjunction with q'_k terms, by convergence to b'_j values as $v \to \infty$.

(c) Once b'_j terms are known, one now has

$$Q'_{I,J}(z)=\frac{\sum_i a'_iz^{-i}}{\sum_j b'_jz^{-j}}=q'(z)$$

(3.4.33)

from which a'_i terms may be evaluated. This then completes step (c). Before concluding here, it should be mentioned that possible errors introduced by the procedure in Fig (3.4.2) may be reduced as N is increased or use of elaborate noise generator algorithms for best white noise simulation.

3.4.3.3 Simualtion Algorithm to Achieve Pade' model

The following algorithm in Fig.(3.4.3) was utilized to simulate the Pade' modelling and identification process. This was based on an ARMA model for a system of higher order versus an assumed lower order Pade' model. Fig. (3.4.3) summaries the Pade' approximation simulation algorithm used in the programs written.

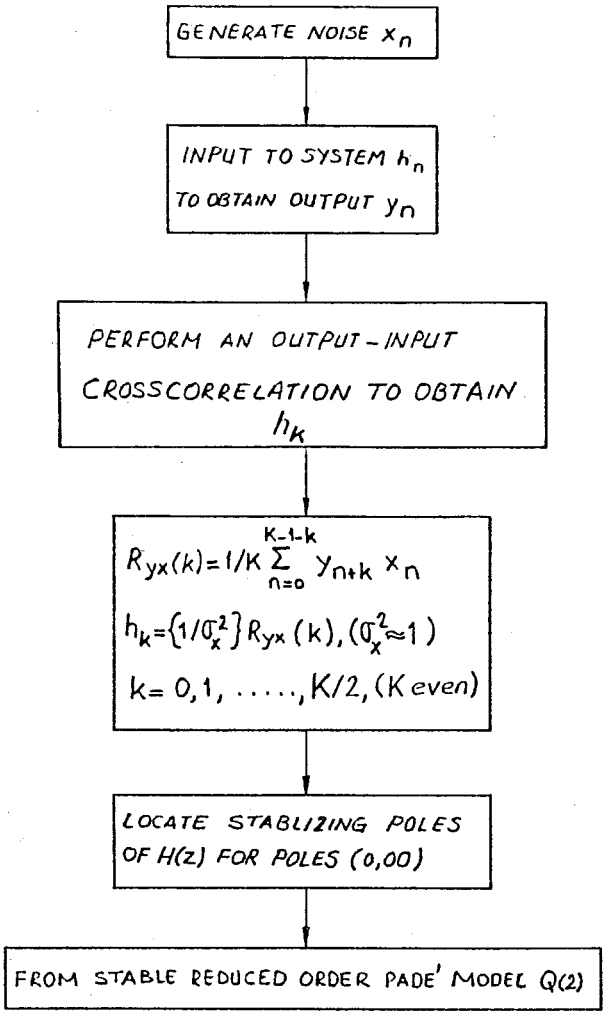


Figure 3.4.3. Summary of Pade' approximation

Figure (3.4.3) thus establishes the a forementioned simulation algorithm.

Pade' Simulation:

Basically the simulation takes on the foolowing approach:

(1) assume a high order ARMA system model,

(2) achieve the system impulse response h_k through nonparametric crosscorrelation,

(3) form the Pade' model Q(z),

(4) compare the true system impulse h_k to $q_k = Z^{-1}[Q(Z)]$.

(5) form an opinion for the goodness of the Pade' model verses the ARMA model based on (4).

3.5. STABILITY OF DISCRETE - TIME SYSTEMS

Stability can be defined in a variety of ways. We will use the following definition : A system is stable if and only if its output is bounded for every bounded input.

This definition is particularly suited to linear systems. For linear systems it is not necessary to test for a bounded output with every bounded input. It is only necessary to examine the pulse response of the system. The condition that the pulse response must satisfy for a time-invariant system is presented in the following theorem.

THEOREM 3.3.

A linear, time-invariant, discrete-time system with pulse response g[nT] is stable if and only if

$$\sum_{-\infty}^{\infty} |g[nT]| < \infty \qquad (3.5.1)$$

Proof:

Let us assume that g[nT] satisfies Eq.(3.5.1) and that f[nT] is any input signal with property that $|f[nT]| < L < \infty$ for all n. Then the output is

$$y[nT] = \sum_{k=-\infty}^{\infty} g[kT]\, f[nT-kT] \qquad (3.5.1)$$

so that

$$|y[nT]| \le \sum_{k=-\infty}^{\infty} |g[kT]||f[nT-kT]| \le L\sum_{k=-\infty}^{\infty} |g[kT]| \tag{3.5.3}$$

Therefore, if g[nT] is absolutely summable, then any bounded input causes a bounded output.

On the other hand, let us assume that g[nT] is not absolutely summable. We can choose f[nT] as

$$f[nT] = sign(g[rT-nT]) \tag{3.5.4}$$

where r is an integer and

$$sign(x) = \begin{cases} 1 & for \quad x>0 \\ 0 & for \quad x=0 \\ -1 & for \quad x<0 \end{cases} \tag{3.5.5}$$

Obviously, |f(nT)| ≤ 1. With this choice

$$y[rT] = \sum_{k=-\infty}^{\infty} g[kT]\, sign(g[kT]) = \sum_{k=-\infty}^{\infty} |g[kT]| = \infty \tag{3.5.6}$$

Consequently, the system is stable only if g(nT) is absolutely summable.

For causal systems with rational pulse transfer functions our definition of stability leads to the following frequently used criterion.

COROLARLY 3.4.

A causal system with a rational pulse transfer function G(z) is stable if and only if all the poles of G(z) are inside unit circle.

Proof:

If all the poles of G(z) are inside the unit circle, then the region of convergence for

$$Z\{g[nT]\}=\sum_{n=0}^{\infty} g[nT]\,z^{-n} \tag{3.5.7}$$

includes the unit circle. Therefore, this series converges absolutely for $|z| = 1$ so that

$$\sum_{n=0}^{\infty}|g[nT]|<\infty \tag{3.5.8}$$

and that system is stable according to Theorem 3.3.

On the other hand, if G(z) has any poles on or outside the unit circle, then the unit circle is not is the region of convergence. In this case, the series diverges for same z_o with $|z_o| = 1$ so that

$$\infty=\sum_{n=0}^{\infty} g[nT]\,z_0^{-n}\le\sum_{n=0}^{\infty}|g[nT]| \tag{3.5.9}$$

and the system is not stable according to the above theorem.

There are various methods developed for determining the locations of the poles of a rational pulse transfer function relative to the unit circle, such as the modified Schur-Cohn test the Nyquist criterion and root locus method.In this study, the satisfaction of the above condition is checked via a pascal program which finds the poles of the transfer function found by ARMA or PADE. This program is given in Appendix A.

For more information one can refer to [33], [34], [35].

3.6. STATE SPACE MODELLING

Systems that can be described by difference or differential equations are types of dynamical systems. For a dynamical system a set of variables called the state of the system can be found that contains all the information about the past behaviour of the system necessary to calculate its future state and a output given its present and future output. So state-space representation means representing an nth order, linear, difference or differential equation by a first-order, linear,matrix difference or differential equation describing the evolution of an n-dimensional state vector and an equation relating the present output to the present state and input. These equations are called as the state equation and output equation or sometimes simply as a state space representation. Different structures for realizing n-th order, linear difference or differential equations are examined from the state-space point of view.

The mathematical time-domain models used to describe sampled-data systems are almost always finite-order difference equations and differential equations whose solution exists and unique. The behaviour of these systems for $t \geq t_o$ can be uniquely determined if an appropriate set of initial conditions at time t_o is specified.

A system whose input v(t) and output y(t) are related by the constant-coefficient, linear, differantial equation

$$\frac{d^N}{dt^N} y(t) + b_1 \frac{d^{N-1}}{dt^{N-1}} y[t] + \ldots\ldots + b_N y(t) - a_0 \frac{d^N}{dt_N} v(t) + a_1 \frac{d^{N-1}}{dt^{N-1}} v(t) + \ldots\ldots + a_N v(t) \qquad (3.6.1)$$

has the transfer function

$$G(s) = \frac{Y(s)}{V(s)} = \frac{a_0 s^N + a_1 s^{N-1} + \ldots + a_N}{s^N + b_1 s^{N-1} + \ldots + b_N} \tag{3.6.2}$$

The transfer function G(z) given by Eq.(3.1.43) can be put in the form

$$G(z) = \frac{a_0 z^N + a_1 z^{N-1} + \ldots + a_N}{z^N + b_1 z^{N-1} + \ldots + b_N} \tag{3.6.3}$$

Therefore, each of the structures which can be applied for the transfer function G(s) when z is replaced by s. Each delay element labeled z^{-1} becomes an element labeled s^{-1} which is an integrator. The outputs of the integrators can be chosen as the state variables. To maintain the correspondence between the continuous and discrete-time systems, we will choose the state variables so that $x_k(t)$ corresponds to $x_k[nT]$. The input to the kth delay element $x_k[nT+T]$ becomes the input to the kth integrator and so must be relabeled x_k' This transformation is shown for the type 1 direct form realization in Figure (3.6.2). Clearly, the state and output equations for the continuous-time structure obtained by the simple transformation described in the previous paragraph can be determined from the equations for the original discrete-time structure simply by replacing v[nT] by v(t), y[nT] by y(t), x(t), and x[nT+T]) by x'(t).

By a linear, finite-dimensional, discrete-time, dynamical system we will mean a system with input $v(t_n)$, output $y(t_n)$, and state $x(t_n)$ having a state equation of the form

$$\boldsymbol{x}(t_{n+1}) = \boldsymbol{A}(t_n)\boldsymbol{x}(t_n) + \boldsymbol{B}(t_n)\boldsymbol{v}(t_n) \tag{3.6.4}$$

and an output equation of the form

$$\mathbf{Y}(t_n) = \mathbf{C}(t_n)\mathbf{x}(t_n) + \mathbf{D}(t_n)\mathbf{v}(t_n) \tag{3.6.5}$$

where

$\mathbf{x}(t_n)$ is an N-dimensional column vector
$\mathbf{v}(t_n)$ is an m-dimensional column vector
$\mathbf{y}(t_n)$ is an r-dimensional column vector
$\mathbf{A}(t_n)$ is an N X N nonsingular matrHx
$\mathbf{B}(t_n)$ is an N X m matrix
$\mathbf{C}(t_n)$ is an r X N matrix
$\mathbf{D}(t_n)$ is an r X m matrix

and

$$t_{n_2} > t_{n_1} \qquad for \qquad n_2 > n_1 \tag{3.6.6}$$

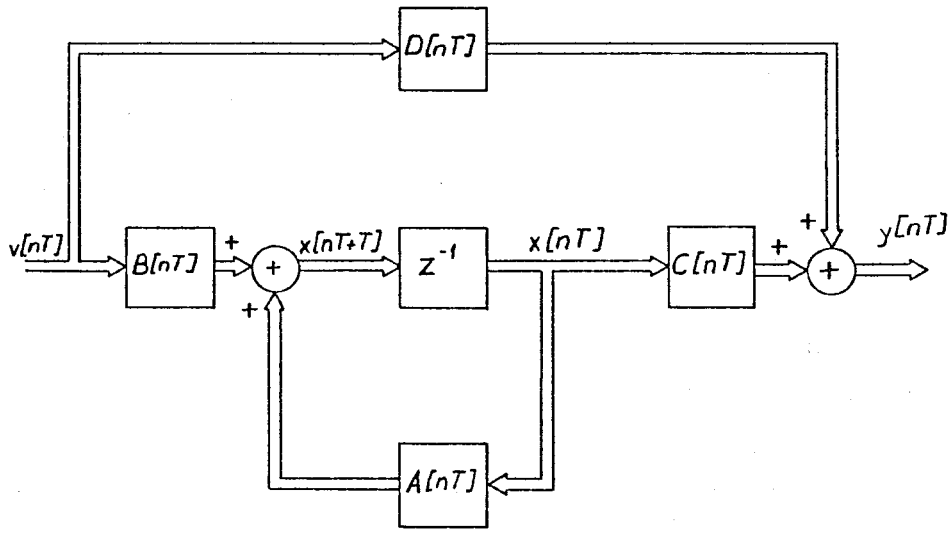


Figure 3.6.1. Pictorial representation of the state and output equations for a uniformly sampled, linear, discret-time sytems

We will use uniform sampling and let t_n=nT. In this case, the state and output equations can be represented pictorially by the block diagram in Figure (3.6.1).

When t_n=nT and **A**, **B**, **C**, and **D** are constant matrices in

Eq. (3.6.4) and Eq.(3.6.5) we say that system is a time-invariant, linear, discrete-time system. In this case the state equation becomes

$$\boldsymbol{x}(nT+T) = \boldsymbol{A}\boldsymbol{x}(nT) + \boldsymbol{B}\boldsymbol{v}(nT) \tag{3.6.7}$$

and the output equation becomes

$$\boldsymbol{Y}(nT) = \boldsymbol{C}\boldsymbol{x}(nT) + \boldsymbol{D}\boldsymbol{v}(nT) \tag{3.6.8}$$

A closed form for the state transition matrix and solution of the state and output equations can be obtained by Z-transform methods. We will define the one-sided Z-transform of an r x s matrix function f[nT] as the r x s matrix.

$$\boldsymbol{F}(z) = \sum_{n=0}^{\infty} \boldsymbol{f}(nT)\, z^{-n} \tag{3.6.9}$$

The elements of F(z) are just the transforms of the corresponding elements of f[nT]. Taking the transform of both sides of the state equation (3.6.7) gives

$$z\boldsymbol{X}(z) - z\boldsymbol{x}(0) = \boldsymbol{A}\boldsymbol{X}(z) + \boldsymbol{B}\boldsymbol{V}(z) \tag{3.6.10}$$

so that

$$\boldsymbol{X}(z) = (z\boldsymbol{I}-\boldsymbol{A})^{-1} z\boldsymbol{x}(0) + (z\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{B}\boldsymbol{V}(z) \tag{3.6.11}$$

From Eq.(3.6.8) we see that

$$\boldsymbol{Y}(z) = \boldsymbol{C}\boldsymbol{X}(z) + \boldsymbol{D}\boldsymbol{V}(z) \tag{3.6.12}$$

In many applications, and in all system identification methods one is primarily interested in the pulse transfer functions between the input and outputs of a system. Letting x(0)=0 and substituating Eq.(3.6.11) into Eq.(3.6.12), we find that

$$\boldsymbol{Y}(z) = [\boldsymbol{C}(z\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{B}+\boldsymbol{D}\,]\,\boldsymbol{v}(z) \tag{3.6.13}$$

The matrix

$$\boldsymbol{G}(z) = \boldsymbol{C}(z\boldsymbol{I}-\boldsymbol{A})^{-1}\boldsymbol{B}+\boldsymbol{D} \tag{3.6.14}$$

is known as the pulse transfer function matrix for the system since its ijth elements is the transfer function between the ith output and jth input For systems with a single input and single outputs, **G**(z) reduces to the ordinary scalar transfer function Y(z)/V(z).

3.6.1. STATE SPACE REPRESENTATIONS FOR CONSTANT-COEFFICIENT, LINEAR, DIFFERENCE EQUATIONS

In this section we again examine structures for realizing a system that has the pulse transfer function

$$G(z) = \frac{\sum_{k=0}^{N} a_k z^{-k}}{1+\sum_{k=1}^{N} b_k z^{-k}} \tag{3.6.15}$$

or difference equation

$$x[nT] = \sum_{k=0}^{N} a_k u[nT-T] - \sum_{k=1}^{N} b_k x[nT-T] \tag{3.6.16}$$

which is given before in Eq.(3.1.38) relating its input and output. By assigning state variables to the outputs of the delay elements in the block diagrams for the structures, we derive different state space representations for the difference Eq.(3.6.16). A reason for studying a variety of realizations is to find those that are insensitive to coefficient truncation, finite word length arithmetic, and other deviations from the

ideal in actual hardware. These problems are discussed later.

3.6.1.1. Type 1 Direct Form Realization

Type 1 direct form realization in Figure (3.6.2) has its input and output related by Eq.(3.6.16). As shown in Figure (3.6.1), we will choose the state variables $x_1[nT],\ldots,x_N[nT]$ as the outputs of the delay elements. From the block diagram we see that

$$\begin{aligned} x_1[nT+T] &= x_2[nT] \\ x_2[nT+T] &= x_3[nT] \\ &\vdots \\ x_{N-1}[nT+T] &= x_N[nT] \\ x_N[nT+T] &= -b_N x_1[nT] - b_{N-1}x_2[nT] - \cdots - b_1 x_N[nT] + v[nT] \end{aligned} \tag{3.6.17}$$

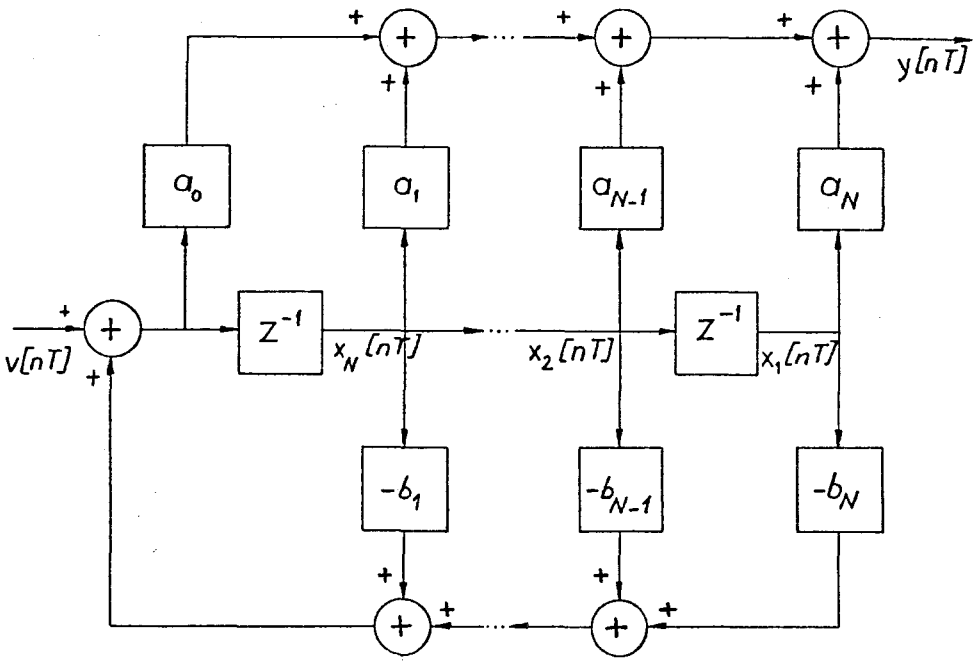


Figure 3.6.2. Type 1 direct form realization of G(z)

and

$$y[nT] = a_N x_1[nT] + a_{N-1} x_2[nT] + \cdots + a_1 x_N[nT] + a_0 [x[nT] - b_N x_1[nT] - b_{N-1} x_2[nT] - \cdots m - b_1 x_n[nT]]$$

(3.6.18)

or

$$y[nT] = (a_N - a_0 b_N) x_1[nT] + a_{N-1} - a_0 b_{N-1}) x_2[nT] + \cdots + (a_1 - a_0 b_1) x_N[nT] + a_0 v[nT]$$

(3.6.19)

Putting Eq.(3.6.17) and Eq.(3.6.18) into matrix form, we see that this structure can be described by the state equation

$$\begin{bmatrix} x_1[nT+T] \\ x_2[nT+T] \\ \vdots \\ x_{N-1}[NT+T] \\ \dot{x}_N[nT+T] \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 \cdots & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ -b_N & -b_{N-1} & \cdots & & -b_1 \end{bmatrix} \begin{bmatrix} x_1[T] \\ x_2[T] \\ \vdots \\ x_{N-1}[T] \\ x_N[T] \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} v[nT]$$

(3.6.20)

and output equation

$$y[t] = [a_N - a_0 b_N \;\; aN-1 - a_0 b_{N-1} \;\; \cdots \;\; a_1 - a_0 b_1] \begin{bmatrix} x_1[nT] \\ x_2[nT] \\ \cdot \\ \cdot \\ \cdot \\ x_N[nT] \end{bmatrix} + a_0 v[t]$$

(3.6.21)

These equations have the desired forms of Eq.(3.6.7) and Eq.(3.6.8).

3.6.1.2 Type 2 Direct Form Realization

The structure called the type 2 direct form realization shown in Figure (3.6.3) has its input and output related by the difference Eq.(3.6.17). If we choose the state variables as shown in Figure (3.6.3), then it follows that

$$y[nT] = x_N[nT] + a_0 v[nT] \tag{3.6.22}$$

and

$$\begin{aligned}
x_1[nT+T] &= -b_N y[nT] + a_N v[nT] = -b_N x_N[nT] + (a_N - a_0 b_N) v[nT] \\
x_2[nT+T] &= x_1[nT] - b_{N-1} x_N[nT] + (a_{N-1} - a_0 b_{N-1}) v[nT] \\
&\vdots \\
x_{N-1}[nT+T] &= x_{N-2}[nT] - b_2 x_N[nT] + (a_2 - a_0 b_2) v[nT] \\
x_N[nT+T] &= x_{N-1}[nT] - b_1 x_N[nT] + a_1 - a_0 b_1) v[nT]
\end{aligned} \tag{3.6.23}$$

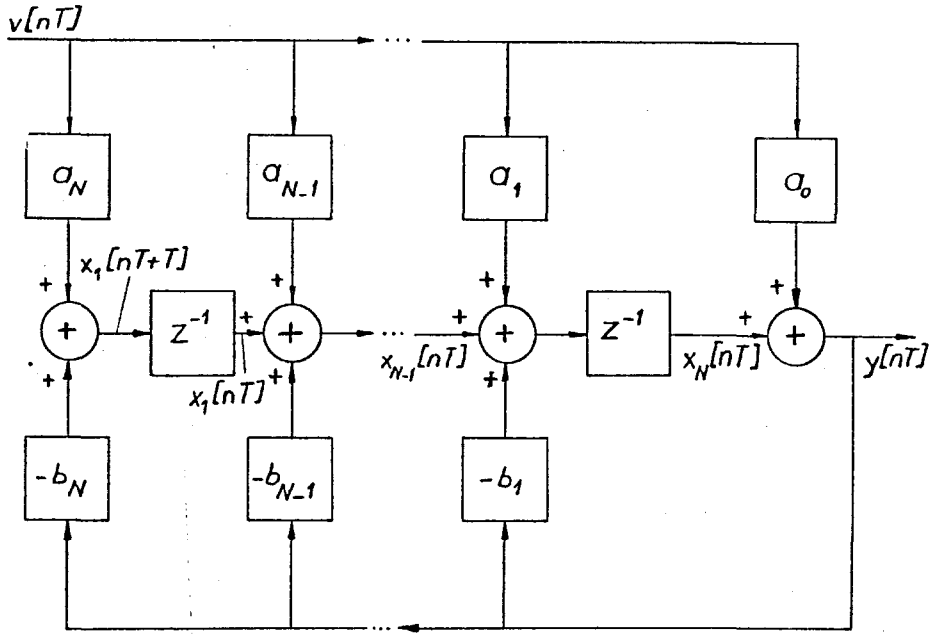


Figure 3.6.3. Type 2 direct form realisation of G(z)

Putting Eq.(3.6.22) and Eq.(3.6.23) into matrix form, we see that the type 2 direct form realization is described by the state equation

$$\begin{bmatrix} x_1[nT+T] \\ x_2[nT+T] \\ \vdots \\ x_{N-1}[NT+T] \\ x_N[nT+T] \end{bmatrix} = \begin{bmatrix} 0 & 0 & \cdots & & 0 & -b_N \\ 1 & 0 & \cdots & & 0 & -b_{N-1} \\ \vdots & \vdots & & & & \vdots \\ 0 & 0 & \cdots & 0 & 1 & -b_2 \\ 0 & 0 & \cdots & 0 & 1 & -b_1 \end{bmatrix} \begin{bmatrix} x_1[T] \\ x_2[T] \\ \vdots \\ x_{N-1}[T] \\ x_N[T] \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} v[nT]$$

(3.6.24)

and output equation

$$y[t] = [0 \ \cdots \ 0 \ 1] \begin{bmatrix} x_1[nT] \\ \vdots \\ x_N[nT] \end{bmatrix} + a_0 v[t]$$

(3.6.25)

3.6.1.3 Standard Form Realization

Another structure that can be used to realize the difference Eq.(3.6.4) is shown in Figure (3.6.5). This is sometimes called the standart form realization. The procedure to choose the parameteres $\alpha_0, \ldots \alpha_N$ and $\beta_1, \ldots, \beta_N$ to obtain the proper input-output relationship. Figure (3.6.4) is as follows from the block diagram it is clear that

$$\begin{aligned} x_1[nT+T] &= x_2[nT] + \alpha_1 v[nT] \\ &\vdots \\ x_k[nT+T] &= x_{k+1} + \alpha_k v[nT] \qquad \text{for } 1 \le k \le N-1 \\ &\vdots \\ x_N[nT+T] &= -\beta_N x_1[nT] - \beta_{N-1} x_2[nT] - \cdots - \beta_1 x_N[nT] + \alpha_N v[nT] \end{aligned}$$

(3.6.26)

and

$$y[nT] = x_1[nT] + \alpha_0 v[nT]$$

(3.6.27)

From Eq.(3.6.27) we see that

$$y[nT+T] = x_1[nT+T] + \alpha_0 v[nT+T] \tag{3.6.28}$$

and using the expression for $x_1[nT+T]$ from Eq.(3.6.26) that

$$y[nT+T] = x_2(nT) + \alpha_1 v(nT) + \alpha_0 v(nT+T) \tag{3.6.29}$$

Similarly

$$\begin{aligned} y[nT+2T] &= x_3[nT] + \alpha_2 v[nT] + \alpha_1 v[nT+T] + \alpha_0 v[nT+2T] \\ &\vdots \\ y[nT+(N-1)T] &= x_N[nT] + \alpha_{N-1} v[nT] + \alpha_{N-2} v[nT+T] + \cdots \\ &\quad + \alpha_0 v[nT+(N-1)T] \end{aligned} \tag{3.6.30}$$

and

$$\begin{aligned} y[nT+T] = -&\beta_N x_1[nT] - \beta_{N-1} x_2[nT] - \cdots - \beta_1 x_N[nT] \\ &+ \alpha_N v[nT] + \cdots + \alpha_0 v[nT+NT] \end{aligned} \tag{3.6.31}$$

Replacing n by n+N, the desired difference Eq.(3.6.16) becomes

$$\begin{aligned} y[nT+T] = -&b_1 y[nT+[N-1]T] - b_2 y[nT+(N-2)T - \cdots - b_N y[nT] \\ &+ a_0 v[nT+T] + \cdots + a_M v[nT] \end{aligned} \tag{3.6.32}$$

Substituting the expressions for y[nT],....y[nT+(N - 1)T] given by Eq.(3.6.27), Eq.(3.6.29), and Eq.(3.6.30) into Eq.(3.6.32) we get

$$\begin{aligned} y[nT+T] = -&b_1\{x_N[nT] + \alpha_{N-1} v[nT] + \alpha_{N-2} v[nT+T] + \cdots \\ &\quad + \alpha_0 v[nT+(N-1)T]\} \\ &- b_2\{x_{N-1}(nT) + \alpha_{N-1} v[nT] + \cdots + \alpha_0 v[nT+(N-2)T]\} \\ &\vdots \\ &- b_N\{x[nT] + \alpha_0 v[nT]\} \\ &+ a_0 v[nT+T] + \cdots + a_N v[nT] \end{aligned} \tag{3.6.33}$$

Equating the coefficients of $x_1[nT],\ldots,x_N[nT]$ and $v[nT],\ldots,v[nT+NT]$ in Eq.(3.6.31) and Eq.(3.6.33), we see that the standart from realization parameters must be

$$\beta_k = b_k \qquad for \qquad k=1,\ldots,N \tag{3.6.34}$$

and

$$\begin{aligned}
\alpha_0 &= a_0 \\
\alpha_1 &= a_1 - b_1\alpha_0 \\
\alpha_2 &= a_2 - b_2\alpha_0 - b_1\alpha_1 \\
&\vdots \\
\alpha_N &= a_N - b_N\alpha_0 - b_{N-1}\alpha_1 - \cdots - b_1\alpha_{N-1}
\end{aligned} \tag{3.6.35}$$

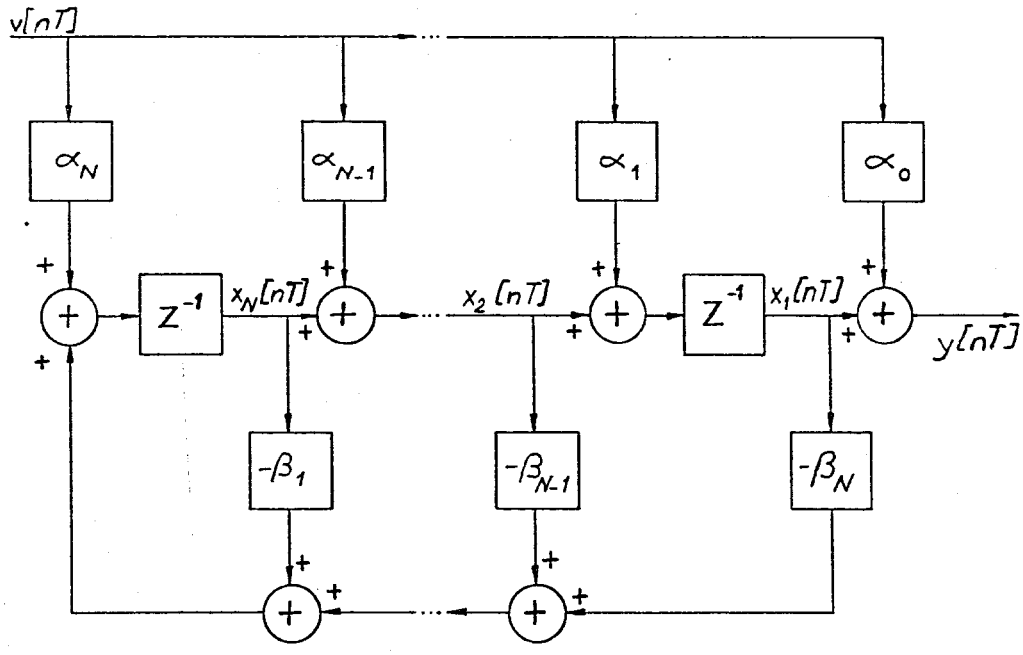


Figure 3.6.4. Standard form realization of G(z)

The set of equations in Eq.(3.6.35) is equivalent to

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ b_1 & 1 & 0 & 0 & \cdots & 0 \\ b_2 & b1 & 0 & 0 & \cdots & 0 \\ \vdots & & & & & \vdots \\ b_N & b_{N-1} & b_{N-2} & b_{N-3} & \cdots & b_1 \quad 1 \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix} \tag{3.6.36}$$

so that

$$\begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix} - \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ b_1 & 1 & 0 & 0 & \cdots & 0 \\ b_2 & b_1 & 1 & 0 & \cdots & 0 \\ \vdots & & & & & \vdots \\ b_N & b_{N-1} & b_{N-2} & b_{N-3} & \cdots & b_1 \quad 1 \end{bmatrix}^{-1} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_N \end{bmatrix} \tag{3.6.37}$$

Eq.(3.6.35) provide a convenient iterative solution to Eq.(3.6.37). Putting Eq.(3.6.26) and Eq.(3.6.27) into matrix form, we find that the standart form realization has the state equation

$$\begin{bmatrix} x_1[nT+T] \\ x_2[nT+T] \\ \vdots \\ x_{N-1}[NT+T] \\ x_N[nT+T] \end{bmatrix} - \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & & & & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ -b_N & -b_{N-1} & \cdots & -b_2 & -b_1 \end{bmatrix} \begin{bmatrix} x_1() \\ x_2(T) \\ \vdots \\ x_{N-1}(T) \\ x_N(T) \end{bmatrix} + \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{N-1} \\ \alpha_N \end{bmatrix} v[nT] \tag{3.6.38}$$

and output equation

$$y(nT) - [1 \ 0 \ \cdots \ 0] \begin{bmatrix} x_1(nT) \\ \vdots \\ x_N(nT) \end{bmatrix} + \alpha_0 v(t) \tag{3.6.39}$$

3.6.1.4 Parallel Type Representation

Another state space representation can be obtained by the partial fraction technique. This method results in a parallel form structure. First, let us assume that G(z) has N simple poles located at $p_1,\ldots,p_N$. Then G(z) can be expressed as

$$G(z) = d_0 + \sum_{k=1}^{N} \frac{d_k}{z-p_k} \tag{3.6.40}$$

where

$$d_0 = \lim_{z\to\infty} G(z) = a_0 \tag{3.6.41}$$

and

$$d_k = \lim_{z\to p_k} (z-p_k)\,G(z) \qquad for \qquad k=1,\ldots,N \tag{3.6.42}$$

Therefore

$$Y(z) = G(z)\,V(z) = a_0 V(z) + \sum_{k=1}^{N} d_k \frac{V(z)}{z-p_k} \tag{3.6.43}$$

Letting

$$X_k(z) = \frac{V(z)}{z-p_k} \qquad for \quad k=1,\ldots,N \tag{3.6.44}$$

Y(z) becomes

$$Y(z) = a_0 V(z) + \sum_{k=1}^{N} d_k X_k(z) \tag{3.6.45}$$

The time-domain equivalents of (3.6.44) and (3.6.45) are

$$x_k(nT+T) = p_k x_k(nT) + v(nT) \qquad for \quad k=1,\ldots,N \tag{3.6.46}$$

and

$$y(nT) = \sum_{k=1}^{N} d_k x_k(nT) + a_0 v(nT) \tag{3.6.47}$$

Putting Eq.(3.6.46) and Eq.(3.6.47) into matrix form, we find that the difference Eq.(3.6.16) can be represented by the state equation

$$\begin{bmatrix} x_1[nT+T] \\ x_2[nT+T] \\ \vdots \\ x_N[nT+T] \end{bmatrix} = \begin{bmatrix} p_1 & 0 & 0 & \cdots & 0 \\ 0 & p_2 & 0 & \cdots & 0 \\ \vdots & \vdots & & & \\ 0 & 0 & 0 & \cdots & p_N \end{bmatrix} \begin{bmatrix} x_1() \\ x_2(T) \\ \vdots \\ x_N(T) \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} v[nT]$$

$$Hx \tag{3.6.48}$$

and output equation

$$y[nT] = [d_1 \;\; d_2 \;\; \cdots \;\; d_N] \begin{bmatrix} x_1[nT] \\ x_2[nT] \\ \vdots \\ x_N[nT] \end{bmatrix} + a_0 v[t] \tag{3.6.49}$$

This is known as the normal form representation of (3.6.16). In this representation the "A" matrix is diagonal so that the state variables are uncoupled. The partial fraction technique can still be used if G(z) has some poles that are not simple. To illustrate the method, let us assume that G(z) has a poles of order r at p_1 and simple poles at $p_{r+1},\ldots,p_N$. Then G(z) can be expressed as

$$G(z) = d_0 + \sum_{k=1}^{r} \frac{d_k}{(z-p_1)^{r-k+1}} + \sum_{k=r+1}^{N} \frac{d_k}{(z-p_k)} \tag{3.6.50}$$

here

$$d_0 = \lim_{z\to\infty} G(z) = a_0 \tag{3.6.51}$$

and

$$d_k = \begin{cases} \lim_{z\to p_1} \frac{1}{(k-1)!} \frac{d^{k-1}}{dz^{k-1}} & for \quad 1 \le k \le r \\ \lim_{z\to p_k} (z-p_k)\, G(z) & for \quad r+1 \le k \le N \end{cases} \tag{3.6.52}$$

Therefore

$$Y(z) = G(z)\, V(z) = a_0 V(z) + \sum_{k=1}^{r} d_k \frac{V(z)}{(z-p_1)^{r-k+1}} + \sum_{k=r+1} d_k \frac{V(z)}{z-p_k} \tag{3.6.53}$$

Letting

$$X_k(z) = \frac{V(z)}{(z-p_1)^{r-k+1}} \qquad for \quad 1 \le k \le r \tag{3.6.54}$$

and

$$X_k(z) = \frac{V(z)}{z-p_k} \qquad for \quad r+1 \le k \le N \tag{3.6.55}$$

Y(z) becomes

$$Y(z) = a_0 V(z) + \sum_{k=1}^{N} d_k X_k(z) \tag{3.6.56}$$

Notice that

$$X_r(z) = \frac{V(z)}{z-p_1} \tag{3.6.57}$$

and

$$X_k(z) = \frac{X_{k+1}(z)}{z-p_1} \qquad for \quad 1 \le k \le r-1$$

(3.6.58)

The time domain equvialents of Eq.(3.6.55), Eq.(3.6.56), Eq.(3.6.57), and Eq.(3.6.58) are

$$x_k[nT+T] = \begin{cases} p_1 x_k[nT] + x_{k+1}[nT] & for\ 1 \le k \le r-1 \\ p_1 x_r[nT] + v[nT] & for\ k=r \\ p_k x_k[nT] + v[nT] & for\ r+1 \le k \le N \end{cases}$$

(3.6.59)

and

$$y[nT] = \sum_{k=1}^{N} d_k x_k[nT] + a_0 v[nT]$$

(3.6.60)

or

$$\begin{bmatrix} x_1[nT+T] \\ x_2[nT+T] \\ \vdots \\ x_{N-1}[nT+T] \\ x_N[nT+T] \\ \hline x_{r+1}[nT+T] \\ x_{r+2}[nT+T] \\ \vdots \\ x_{N-1}[nT+T] \\ x_N[nT+T] \end{bmatrix} = \left[\begin{array}{ccccccc:ccccc} p_1 & 1 & 0 & \cdots & & 0 & & & & & \\ 0 & p_1 & 1 & 0 & \cdots & 0 & & & & & \\ \vdots & & & & & & & & 0 & & \\ 0 & \cdots & & & 0 & p_1 & 1 & & & & \\ 0 & \cdots & & & & 0 & p_1 & & & & \\ \hline & & & & & & & p_{r+1} & 0 & \cdots & 0 \\ & & & & & & & 0 & p_{r+2} & 0 & \cdots \\ & & & 0 & & & & \vdots & & & \\ & & & & & & & 0 & \cdots & 0 & p_{N-1} \\ & & & & & & & 0 & \cdots & & 0 \end{array}\right] \begin{bmatrix} x_1[nT] \\ x_2[nT] \\ \vdots \\ x_{N-1}[nT] \\ x_N[nT] \\ \hline x_{r+1}[nT] \\ x_{r+2}[nT] \\ \vdots \\ x_{N-1}[nT] \\ x_N[nT] \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ \hline 1 \\ 1 \\ \cdots \\ 1 \\ 1 \end{bmatrix} v[n]$$

(3.6.61)

$$y[nT] = [d_1 \ \cdots \ d_N] \begin{bmatrix} x_1[nT] \\ \vdots \\ x_N[nT] \end{bmatrix} + a_0 v[nT]$$

(3.6.62)

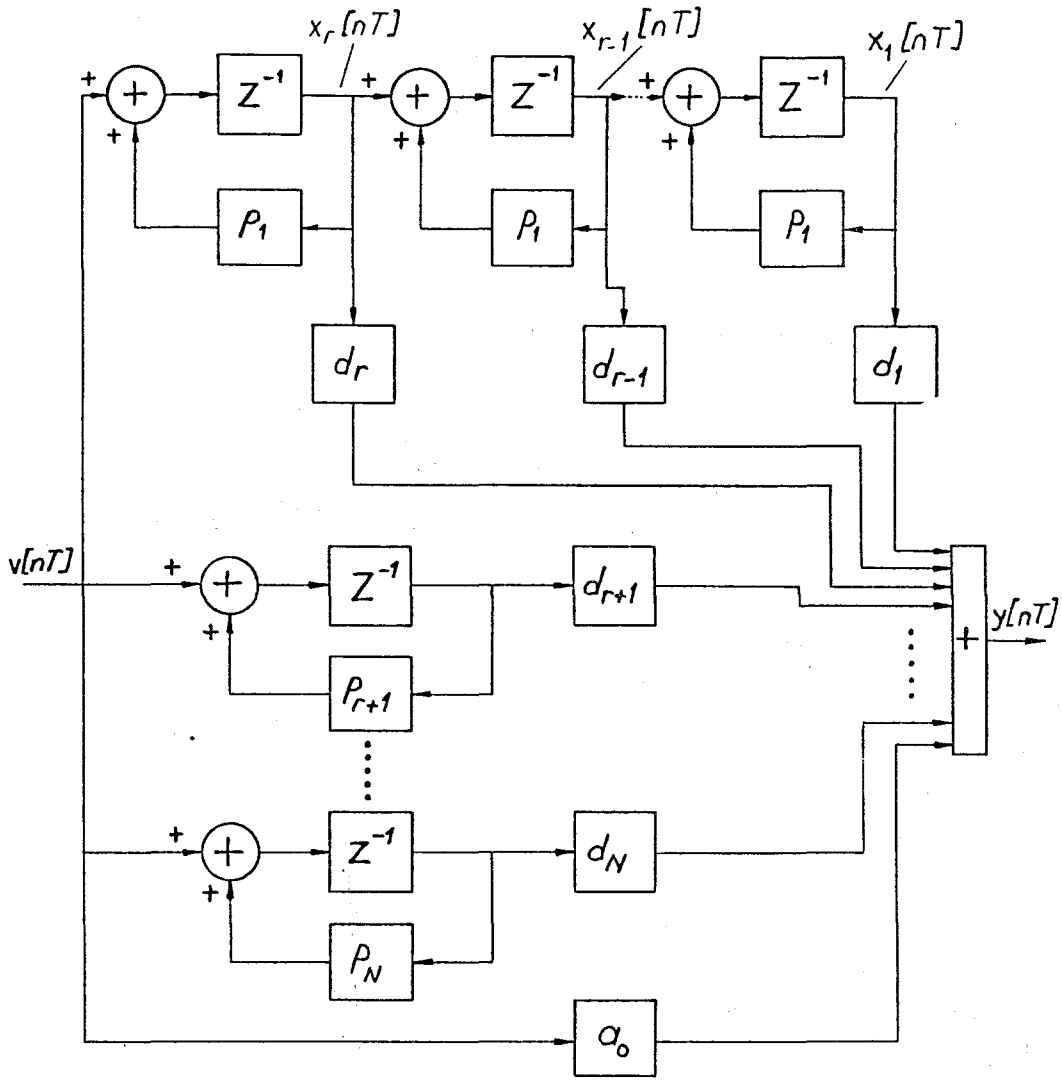


Figure 3.6.5. Parallel form realization of transfer function G(z)

The r x r block in the upper left-hand corner of the "A" matrix is called a Jordan block. The block diagram corresponding to this system realization is shown in Figure (3.6.6). This structure is called a parallel form realization for obvious reasons.

3.6.1.5. Cascade Form Realization

The cascade form realization is frequently used in practice. This structure results when G(z) is expressed as the product of low-order rational factors and is realized as a cascade of sections corresponding to these factors. This structure is particularly appropriate when G(z) has zeros on or near the unit circle. To illustrate one form of cascade realization and the corresponding state space representation, let us assume that a_o is not zero in Figure (3.6.2).

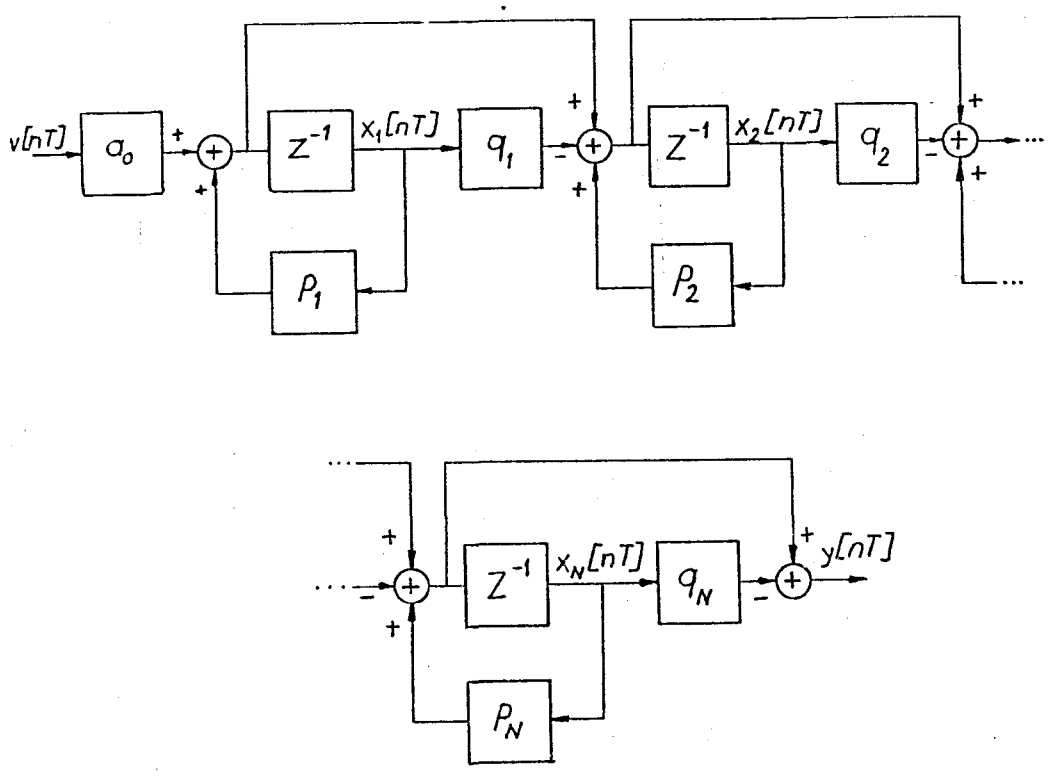


Figure 3.6.6. Cascade form realization of transfer function G(z)

Then G(z) can be factored and written as

$$G(z) = a_0 \prod_{k=1}^{N} \frac{z-q_k}{z-p_k}$$

(3.6.63)

When each first-order rational factor of Eq.(3.6.63) is realized by a type 1 direct form section and the output and input addres of adjacent sections are combined, we obtain the structure shown in Figure (3.6.6). Choosing the state variables as shown in this figure we find that

$$x_1[nT+T] = p_1 x_1[nT] + a_0 v[nT]$$

(3.6.64)

$$x_k[nT+T] = p_k x_k[nT] - q_{k-1}\, x_{k-1}[nT] + x_{k-1}[nT+T] \quad for \quad 2 \le k \le N$$

(3.6.65)

and

$$y(nT) = x_N(nT+T) - q_N x_N(nT)$$

(3.6.66)

Starting with k=2, using Eq.(3.6.64), and recursively evaluating Eq.(3.6.65) we also find that

$$x_k[nT+T] = p_k x_k[nT] + \sum_{r=1}^{k-1} (p_r - q_r)\, x_r[nT] + a_0 v[nT] \quad for 2 \le k \le N$$

(3.6.67)

and

$$y[nT] = \sum_{k=1}^{N} (p_k - q_k)\, x_k[nT] + a_0 v[nT]$$

(3.6.68)

or equivalently

$$\begin{bmatrix} x_1[nT+T] \\ x_2[nT+T] \\ \vdots \\ x_{N-1}[nT+T] \\ x_N[nT+T] \end{bmatrix} = \begin{bmatrix} p_1 & 0 & 0 & \cdots & 0 \\ p_1-q_1 & p_2 & 0 & & 0 \\ \vdots & & & & \vdots \\ p_1-q_1 & p_2-q_2 & \cdots & p_{N-1} & 0 \\ p_1 q_1 & p_2-q_2 & \cdots & p_{N-1}-q_{N-1} & p_N \end{bmatrix} \begin{bmatrix} x_1[nT] \\ x_2[nT] \\ \vdots \\ x_{N-1}[nT] \\ x_N[nT] \end{bmatrix} + \begin{bmatrix} a_0 \\ a_0 \\ \vdots \\ a_0 \\ a_0 \end{bmatrix} v[nT]$$

(3.6.69)

and

$$y[nT] = [p_1 - q_1 \cdots p_N - q_N] \begin{bmatrix} x_1[nT] \\ \vdots \\ x_N[nT] \end{bmatrix} + a_0 v[nT] \tag{3.6.70}$$

In practice, the state variables would most likely be calculated recursively by Eq.(3.6.65) rather than directly from Eq. (3.6.68). This corresponds to calculating the outputs of the addres in Figure (3.6.6) sequentially from left to right.

If G(z) has any complex poles or zeros, the cascade form realization shown in Figure (3.6.6) requires complex arithmetic. The parallel form realization shown in Figure (3.6.5) also requires complex arithmetic when G(z) has complex poles. The need for complex arithmetic is frequently eliminated by combining complex conjugate terms into low-order sections with real coefficients. These sections are then implemented as direct or standart form realizations. Various other structures have been suggested for realizing rational pulse transfer functions. In particular, there has been recent interest in realizations using various types of ladder structures. These will not be discussed further here. There are actually an infinite number of realizations for G(z). Some have basically different structures while others differ simply by scale factors. In general, an input-output relationship does not uniquely describe the internal structure of a system. If a realization is described by the equation

$$\boldsymbol{x}[nT+T] = \boldsymbol{A}\boldsymbol{x}[nT] + \boldsymbol{B}\boldsymbol{v}[nT] \tag{3.6.71}$$

and

$$\boldsymbol{y}[nT] = \boldsymbol{C}\boldsymbol{x}[nT] + \boldsymbol{D}\boldsymbol{v}[nT] \tag{3.6.72}$$

then for any NxN nonsingular matrix F the transformation

$$\boldsymbol{x}[nT] = \boldsymbol{F}\boldsymbol{x}'[nT] \tag{3.6.73}$$

results in a new realization described by the equations

$$\boldsymbol{x}'[nT+T] = \boldsymbol{A}'\boldsymbol{x}'[nT] + \boldsymbol{B}'\boldsymbol{v}[nT] \tag{3.6.74}$$

and

$$\boldsymbol{y}[nT] = \boldsymbol{C}'\boldsymbol{x}'[nT] + \boldsymbol{D}'\boldsymbol{v}[nT] \tag{3.6.75}$$

where

$$\begin{aligned} \boldsymbol{A}' &= \boldsymbol{F}^{-1}\boldsymbol{A}\boldsymbol{F} \\ \boldsymbol{B}' &= \boldsymbol{F}^{-1}\boldsymbol{B} \\ \boldsymbol{C}' &= \boldsymbol{C}\boldsymbol{F} \end{aligned} \tag{3.6.76}$$

and

$$\boldsymbol{D}' = \boldsymbol{D} \tag{3.6.77}$$

For more information one can refer to [14], [33].

4. PROBLEMS OF PARAMETRIC MODELLING

In this Chapter, some of the practical problems that must be taken into account when a pulse transfer function is actually implemented digitally will be investigated. These problems are all a result of the fact that numbers must be quantized and represented as finite bit binary words in digital machines. Because all digital technology operates with only a finite number of bits. The quantization process is an irreversible nonlinear operation. The effects of the quantization process can be operated in three categories.

1. Quantization errors are initially introduced when the analog input signal is sampled and converted into a sequence of binary numbers. This is called as input quantization. This effect can be modelled simply by adding noise to the ideal samples.

2. When the coefficients are quantized for implementation, the resulting filter must be checked to insure that its frequency response is still acceptable. Some small changes in the coefficients of a polynomial can cause large changes in the location of its roots when the roots are clustered near the unit circle. The changes are larger for higher order polynomials. This effect is particularly important in recursive filters since their frequency responses and stability are very sensitive to the position of poles near the unit circle.

3. A third type of quantization error is introduced by the rounding of products or sums of products to the original machine word length. This is known as finite word length arithmetic round-off errors.All these errors can be result in an unstable system response which is explained in section 3.5.

All these errors can result in an unstable system response by causing the poles going outside of the unit circle which is explained in section 3.5.

4.1. INPUT QUANTIZATION ERRORS

The process of approximating a sample of a continuous-time signal by a finite digit binary number is known as analog-to-digital conversion. The binary number generated by an analog-to-digital converter (ADC) is almost always in a fixed point format. The two's complement format is frequently chosen since subtraction can be performed by adding the two's complement of the subtrahend to the minuend eliminating the need for a separate subtracter. The nominal two's complement representation of any number n with $-A< x < A$

$$x/A=-b_0+\sum_{n-1}^{\infty} b_n 2^{-n} \tag{4.1.1}$$

where b_n can have only the values 0 or 1. For positive x b_o = 0 and for negative n b_o = 1. Therefore, b_o is called as the sign bit.

In an actual digital machine only a finite number of bits can be used to represent any number. If the numbers are represented by using two's complement format, by simply truncating the series in Eq.(4.1.1), we can obtain the K+1 bit approximation

$$[x]_t=A\left(-b_0+\sum_{n-1}^{k} b_n 2^{-n}\right) \tag{4.1.2}$$

which can be represented by the binary word $(b_0, b_1, \ldots, b_K)$ in the machine. Any number of this form must be an integral multiple of $q=A2^{-K}$

The quantity q is called the quantization step size. The relationship between $[x]_t$, and x is illustrated in Figure (4.1.1). It can be seen that the truncation error, $e_t = x - [x]_t$, must lie in the semi-open interval [0,q). Since the truncation error has a positive bias that can accumulate in a sequence of arithmetic operations, truncation is usually avoided.

Rounding x to the nearest integral multiple of q is a better method of approximating x by a K + 1 bit binary number. The rounded number can be represented as

$$[x]_r = A\left(-b_0 + \sum_{n=1}^{K} b_n 2^{-n} + b_{K+1}2^{-K}\right) \qquad (4.1.3)$$

The relationship between $[x]_r$ and x is shown in Figure (4.1.1). For $A(1-2^{-K-1})=A-q/2 \le x < A$, $b_0 = 0$ and $b_1 = b_2 = \ldots = b_{K+1} = 1$. In this case, the last term $b_{K+1}2^{-K}$ on the right-hand side of Eq.(4.1.3) will cause an overflow into the sign bit if no overflow detection is used. The overflow causes a jump to the value -A as shown on the bottom right of Figure (4.1.2). An advantage of using two's complement arithmetic is that if the total sum of a set normalized numbers is in the range [-1,1), then, even though partial sums overflow or underflow, the correct total sum will be obtained. Therefore, overflow and underflow detection is commonly omitted. The round-off error $e_r=x-[x]_r$ is confined to the interval [-q/2,q/2) except in the small overflow region.

In both round-off and truncation, numbers are quantized to a set of uniformly spaced levels. In some special applications like pulse code modulation voice transmission,

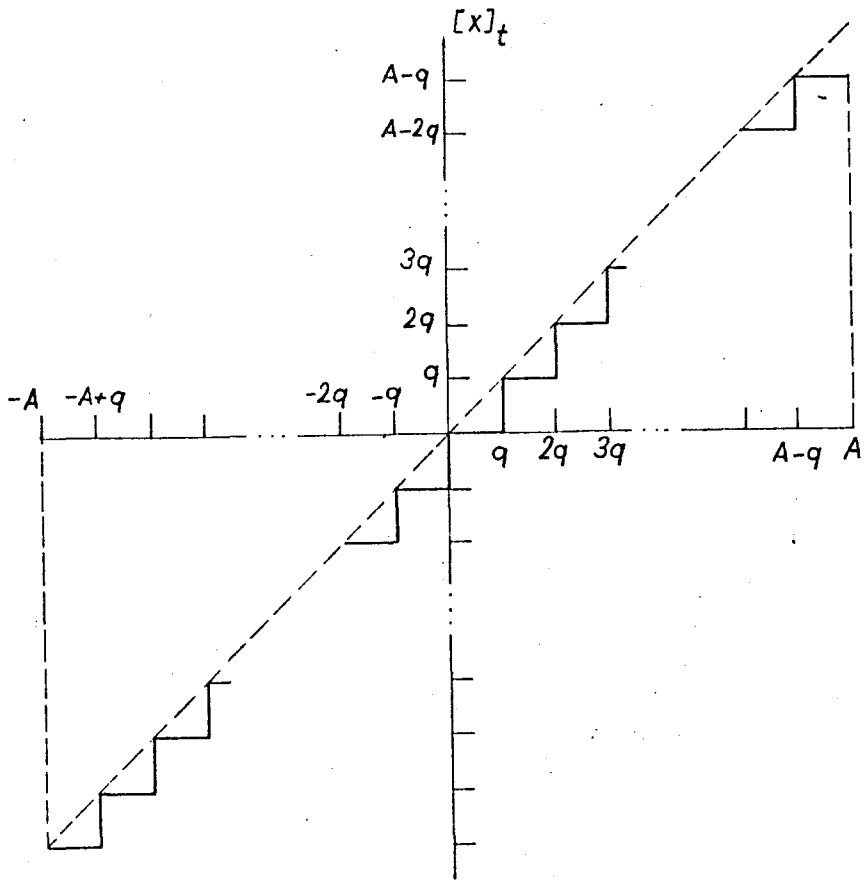


Figure 4.1.1. Truncation of two's complement numbers

signals are quantized to nonuniformly spaced levels to more accurately represent the signal amplitudes that occur most frequently.

Nonuniform quantization can be achieved by first passing the signal through an instantaneous nonlinearity and then into a uniform quantizer.The instantaneous nonlinearity is often called a compander. In the remainder of this chapter we assume that uniform quantization is used.

Let us now assume that the input to the quantizer (i.e., analog-to-digital converter) is a random variable X with the probability density function $f_x(x)$. In addition, let us assume that X can be quantized to any integral multiple of the

quantization step size q so that overflow and saturation do not occur. Then it follows that the probability density function for the round-off error is

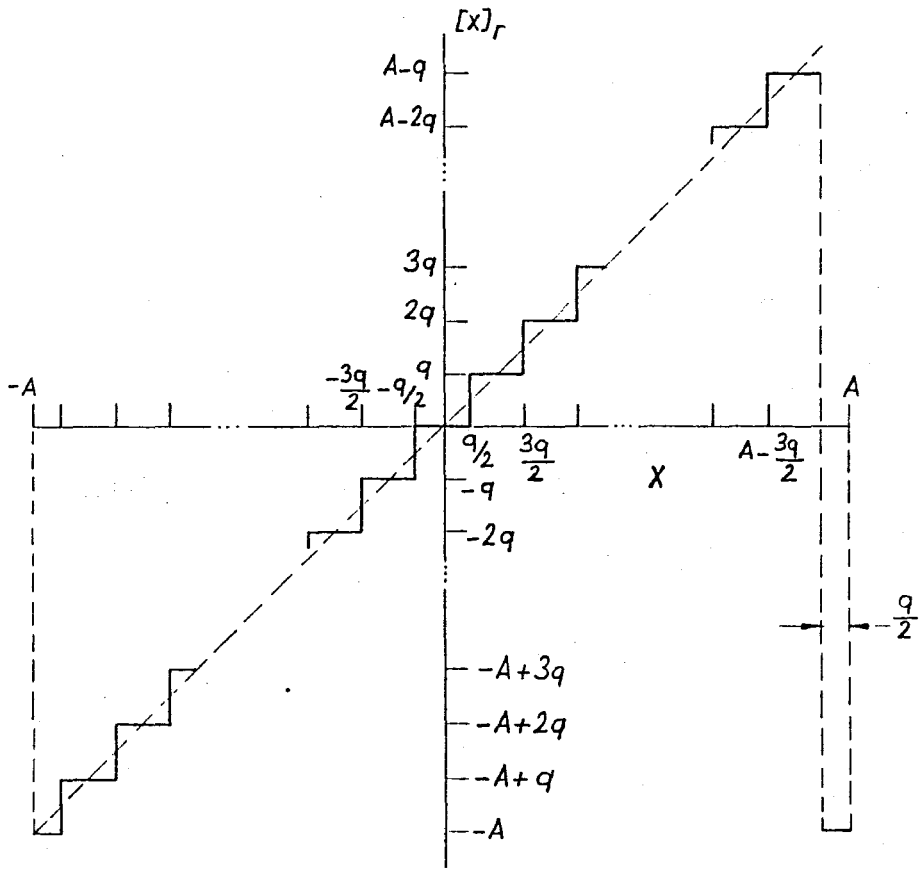


Figure 4.1.2. Rounding of two's complement numbers

$$f_E(e) = \begin{cases} \sum_{k=-\infty}^{\infty} f_X[e+nq] & \text{for} \quad -q/2 \le e < q/2 \\ 0 & \text{elsewhere} \end{cases} \tag{4.1.4}$$

If N is an integer and X is uniformly distributed over [-Nq,Nq] then we find from Eq.(4.1.4) that the round-off error is uniformly distributed over [-Q/2,Q/2], that is

$$f_E(e) = \begin{cases} 1/q & \text{for} \quad -q/2 \le e < q/2 \\ 0 & \text{elsewhere} \end{cases} \tag{4.1.5}$$

In this case, the round-off error has zero mean and variance $q^2/12$. If $f_x(x)$ is moderately broad relative to q, then the round- off error is still almost uniformly distributed over [-q/2,q/2). We can argue similarly that the truncation error is almost uniformly distributed over [0,q). In the remainder of this chapter we will always assume that numbers are quantized by rounding.

If a signal x(t) is sampled and quantized, then

$$[x[nT]]_r = x[nT] + e[nT] \tag{4.1.6}$$

where e[nT] is the round-off error sequence. Theoretical analyses and numerous simulations have shown that, when the probability density function for x(t) is moderately broad relative to q and the frequency spectrum of x(t) is sufficiently broad so that a number of quantization levels are normally crossed from sample to sample, e[nT] can be closely approximated by a white noise sequence uncorrelated with x[nT] and uniformly distributed over [-q/2,q/2) [34], [36]. With these assumption, e[nT] has zero mean, variance $q^2/12$, and the sampled power spectral density

$$S_{ee}(z) = \frac{q^2}{12} \tag{4.1.7}$$

In summary, the effect of analog-to-digital conversion can usually be modeled by simply adding a zero mean white noise sequence of variance $q^2/12$ to the original unquantized discrete-time signal.

The steady-state output component due to e[nT] is a zero-mean wide-sense-stationary (WSS) sequence with power spectral density given by

$$H(z)\,H\left(\frac{1}{z}\right)\frac{q^2}{12} \tag{4.1.8}$$

where H(z) is the transfer function of the filter. Here the effect on the output of coefficient inaccuracy and round off accumulation has been ignored, since their effect on the response to e[n] is much smaller than that due to the response to x[n].

To relate q to the word length of the digital filter, scaling of the input may need to be considered. For example, if the input has been scaled such that $|x_n| \leq 1$ and quantization is at the input of a fixed-point filter with a t-bit quantizer, then $q=2^{-t+1}$. Scaling is usually not important in floating-point filters. When it is used, the input signal spectrum and q^2 are scaled by the same factor.

The mean-squared value of the error at the output due to input quantization can be obtained by integrating the power spectral density given by Eq.(4.1.8). It is equated to

$$\frac{1}{2\pi j}\oint H(z)\,H\left(\frac{1}{x}\right)\frac{q^2}{12}\,\frac{dz}{z} \tag{4.1.9}$$

and can be evaluated, either numerically or algebraically, by a computer program or a table [34], [37].

One can also bound the output component due to input quantization. It is easily seen that the output due to e[n] is bounded in absolute value by $\Sigma_n|h_n|q/2$, where h_n is the impulse response of the filter. Although this bound can be approached with a particular input sequence, it is extremely unlikely for the e[n] to take on these values.

4.2. THE EFFECT OF COEFFICIENT QUANTIZATION

Here, we will try indirectly to investigate the effect of coefficient quantization on the frequency response of a digital filter by examining its effect on the location of the poles and zeros of the filter.

Suppose that the pulse transfer function of the desired filter has the form

$$G(z) = \frac{A(z)}{B(z)} \tag{4.2.1}$$

where

$$A(z) = \sum_{k=0}^{M} a_k z^{-k} \tag{4.2.2}$$

and

$$B(z) = 1 + \sum_{k=1}^{N} b_k z^{-k} = \prod_{k=1}^{N} (1 - p_k z^{-1}) \tag{4.2.3}$$

If the filter is realized using one of the direct forms discussed in Chapter 3, then the denominator coefficients $1, b_1, \ldots, b_N$ will appear directly in the required difference equations. To obtain a rough estimate of the accuracy with which these coefficients must be represented to maintain stability, let us assume that G(z) is a narrow-band low-pass filter. Then the poles of G(z), $p_1, \ldots, p_N$, will be clustered inside the unit circle close to the point z = 1. Therefore, we can write that

$$p_k = 1 + e_k \quad \text{with } |e_k| < 1 \quad \text{for } k = 1, \ldots, N \tag{4.2.4}$$

If a single coefficient b_r is changed to $b'_r = b_r + \delta$, then the new denominator will be

$$B'(z) = 1 + \sum_{k=1}^{N} b_k z^{-k} + \delta z^{-r} = B(z) + \delta z^{-r} \tag{4.2.5}$$

As δ is increased in magnitude, roots of B'(z) will eventually move outside the unit circle. In general, the roots will cross the unit circle at different points. It is particularly easy to check for roots crossing at z = 1. From Eq.(4.2.6) we can see that if

$$\delta = -B(1) = -\left(1+\sum_{k=1}^{N} b_k\right) = -\prod_{k=1}^{N} (1-p_k) \tag{4.2.6}$$

then B'(z) will have a zero at z = 1. Substituting Eq.(4.2.4) into Eq.(4.2.6), we find that

$$|\delta| = \prod |e_n| \leq 1 \tag{4.2.7}$$

Thus only a small coefficient perturbation is required to cause instability. From Eq.(4.2.7), we can see that the accuracy requirements are more severe when the filter order N is large. Similar results can be obtained for other common types of filters that have poles clustered near points on the unit circle.

In addition to maintaining stability, we must insure that the poles and zeros of the implemented filter are sufficiently close to those of the desired filter so that its frequency response is acceptable. Changes in the zeros of B(z) for incremental changes in its coefficients can be examined by using the total differential rule

$$dp_i = \sum_{n=1}^{N} \left.\frac{\partial p_i}{\partial b_n}\right|_{z=p_i} db_n \tag{4.2.8}$$

The partial derivatives can be calculated from the polynomial and factored forms for B(z) in Eq.(4.2.3) using the rule

$$\frac{\partial p_i}{\partial b_n} \quad \frac{\partial B/\partial b_n}{\partial B/\partial p_i}$$

(4.2.9)

When B(z) has only first-order zeros, we find from Eq.(4.2.8) and Eq.(4.2.9) that

$$dp_i = -\sum_{n=1}^{N} \frac{p_i^{1-n}}{\sum_{\substack{k=1 \\ k \neq i}}^{N} (1-p_k p_i^{-1})\, db_n}$$

(4.2.10)

If B(z) has a tightly clustered set of zeros and p_k and p_i are in this set, then $p_k p_i^{-1}$ is close to 1 so that the product in Eq.(4.2.10) will be small and its reciprocal large. In this cases small changes in the coefficients of B(z) will cause large changes in its zeros. This effect becomes more pronounced as the number of zeros in the cluster increases. The same argument applies to the numerator A(z). However,the frequency response of a filter is significantly more sensitive to changes in poles near the unit circle than to changes in zeros.

Changes in the zeros of B(z) when a single coefficient is varied can also be examined by using the root locus method. The right-hand side of Eq.(4.2.5) can be considered to be the characteristic polynomial for a single loop negative feedback system with the open loop gain $\delta z^{-1}/B(z)$.

The accuracy requirements become greater as the filter poles cluster closer together. For low-pass filters the poles cluster near z=1, and for high-pass filters they cluster near z=-1. For band-pass filters they cluster near $z= e^{\pm j\omega_0 T}$ where ω_0 is the center frequency of the filter. If the bandwidth of the filter is W, then a measure of the tightness of the clustering is W/ω_s. The clustering becomes tighter as this ratio decreases. If the filter bandwidth W is held fixed and the sampling rate ω_s is increased, then we see that the poles become more tightly clustered and the accuracy requirements increase.

The results of this section can be assumed up by saying that a direct form implementation of a practical recursive digital filter of order greater than two should usually be avoided. Sometimes even for a third-order filter the accuracy requirements for a direct form realization can be significant. A solution to the problem is to realize the filter by paralleling or cascading first-and second-order sections. The cascade form is most often chosen so that the zeros as well as the poles can be directly controlled.

4.2.1. COEFFICIENT QUANTIZATION ERROR CALCULATION FORMULAS FOR ARMA

Auto correlation coefficients errors:

$$r_{yy}[k] - \left(\frac{1}{N}\right)\left(\sum_j y[k]\,y[k+j]\right)$$

$$r_{yy}[k] - \frac{1}{N}\left(\sum_j \left(y[k] + e_{y[k+j]}\right)\left(y[k+j] + e_{y[k]}\right)\right)$$

$$r_{yy}[k] - r'_{yy}[k] + \frac{1}{N}\left(\sum_j \left(y[k+j]\,e_{y[k]} + e_{y[k+j]}\,y[k] + e_{y[k+j]}\,e_{y[k]}\right)\right) \tag{4.2.10}$$

Error introduced at transfer function coefficients because of the error introduced at auto correlation and transfer function recursion coefficients:

$$a_1[1] - -\frac{r_{yy}[q+1]}{r_{yy}[q]}$$

$$a'_1[1] = -\frac{r_{yy}[q+1] + e_{r_{yy}[q+1]}}{yy[q] + e_{r_{yy}[q]}}$$

$$a'_1[1] - a_1[1] + \frac{r_{yy}[q+1]\,e_{r_{yy}[q]} - e_{r_{yy}[q+1]}\,r_{yy}[q]}{r_{yy}[q]\left(r_{yy}[q] + e_{r_{yy}[q]}\right)} \tag{4.2.11}$$

$$b_1[1] = -\frac{r_{yy}[q-1]}{r_{yy}[q]}$$

$$b'_1[1] = -\frac{r_{yy}[q-1] + \epsilon_{r_{yy}[q-1]}}{r_{yy}[q] + \epsilon_{r_{yy}[q]}}$$

$$b'_1[1] = b_1[1] + \frac{r_{yy}[q-1]\,\epsilon_{r_{yy}[q]} - \epsilon_{r_{yy}[q-1]}\,r_{yy}[q]}{r_{yy}[q]\left(r_{yy}[q] + \epsilon_{r_{yy}[q]}\right)}$$

(4.2.12)

$$\rho_1 = (1 - a_1[1]\,b_1[1])\,r_{yy}[q]$$

$$\rho'_1 = (1 - (a_1[1] + \epsilon_{a_1[1]})\,(b_1[1] + \epsilon_{b_1[1]})\,(r_{yy}[q] + \epsilon_{r_{yy}[q]})$$

$$\rho'_1 = \rho_1 + \left(1 - a_1[1]\,b_1[1]\,\epsilon_{r_{yy}[q]}\right) - \left(\epsilon_{b_1[1]}\,a_1[1] + \epsilon a_1[1]\,b_{[1]}\right)\left(r_{yy}[q] + \epsilon_{r_{yy}[q]}\right)$$

(4.2.13)

$$a_k[k] = -\frac{r_{yy}[q+k] + \sum_{l=1}^{k-1} a_{k-1}[l]\,r_{yy}[q+k-l]}{k-1}$$

$$a'_k[k] = -\frac{\left(r_{yy}[q+k]\,\epsilon_{r_{yy}[q+k]}\right) + \sum_{l=1}^{k-1} (a_{k-1}[l] + \epsilon_{a_{k-1}[l]})\,(r_{yy}[q+k-l] + \epsilon_{r_{yy}[q+k-l]})}{\rho_{k-1} + \epsilon_{\rho_{k-1}}}$$

(4.2.1)

$$\rho_k = (1 - a_k[k]\,b_k[k])\,\rho_{k-1}$$

$$\rho'_k = \left(1 - \left[a_k[k] + \epsilon_{a_k[k]}\right]\left[b_k[k] + \epsilon_{b_k[k]}\right]\right)\left(\rho_{k-1} + \epsilon_{\rho_{k-1}}\right)$$

$$\rho'_k = \rho_k + (1 - a_k[k]\,b_k[k])\,\epsilon_{\rho_{k-1}} - (\epsilon_{a_k[k]}\,b_k[k] + \epsilon_{b_k[k]}\,a_k[k] + \epsilon_{a_k[k]}\epsilon_{b_k[k]})\,(\rho_{k-1} + \epsilon_{\rho_{k-1}})$$

(4.2.16)

4.3. FIXED POINT FINITE WORD LENGTH ARITHMETIC EFFECTS

Suppose that in a particular digital computer numbers are stored in a fixed point format using words of K+1 bits including the sign bit. When two of these numbers are added, the sum can be represented by K+1 bits except when an overflow occurs. When two of the numbers are multiplied using a fixed point algorithm, the full accuracy product contains 2K+1 bits. The typical operation performed in implementing a digital filter is a sum of products. The sum can be carried out using the full 2K+1 bit products rounded to a fewer number of bits. The total sum must then be rounded to K+1 bits for storage. This process is known as finite word length arithmetic.

The accuracy of the stored total sum depends on the number of bits retained in the product for addition as well as the number of bits used for storage. Suppose that products are rounded to less than 2K+1 but more than K+1 bits and that the resulting numbers correspond to multiples of the arithmetic quantization step size q_a. Let us assume that the stored numbers correspond to multiples of the storage quantization step size q_s. Suppose that we wish to calculate the sum of products

$$S=\sum_{n=1}^{N} a_n b_n=\sum_{n=1}^{N} c_n \tag{4.3.1}$$

where a_n and b_n are Kth bit numbers. The rounded products can be written as

$$[c_n]_r=c_n+e_n \tag{4.3.2}$$

where $|e_n|\leq q_a/2$. Thus, the computed sum can be written as

$$S_1 = \sum_{n=1}^{N} [c_n]_r = S + \sum_{n=1}^{N} e_n$$

(4.3.3)

We will assume that numbers have been scaled so that the probability of overflow is negligible. The computed sum rounded to K+1 bits for storage can be written as

$$S_2 = S_1 + v = S + \sum_{n=1}^{N} e_n + v$$

(4.3.4)

where $|v| < q_s/2$. From eq.(4.3.4) we can see that

$$|S_2 - S| \leq Nq_s/2 + q_s/2$$

(4.3.5)

When the full accuracy 2K+1 bit products are used, $e_n=0$ for n=1,...,N so that

$$|S_2 - S| \leq q_s/2$$

(4.3.6)

If the products are rounded to the storage accuracy of K+1 bits before addition, the resuHxing sum has K+1 bits and can be stored directly, so v=0 and

$$|S_2 - S| \leq Nq_s/2$$

(4.3.7)

The bounds given by Eq.(4.3.5), (4.3.6), and (4.3.7) are achievable worst case bounds. When N is greater than or equal to two, the composite bound decreases from $Nq_s/2$ to $q_s/2$ as the number of bits retained in products increases from K+1 to 2K+1.

A less conservative estimate of the noise introduced by finite word length arithmetic can be obtained by an approximate statistical approach. When products are rounded to more than K+1 and less than 2K+1 bits in such a way that the quantization errors $e_1,\ldots,e_N$ and v in Eq.(4.3.4) can take on 16 or more values, simulations have verified that these errors can be adequately modeled as zero mean, uncorrelated random variables

with e_n uniformly distributed over $(-q_a/2, q_a/2)$ and v uniformly distributed over $(-q_s/2, q_s/2)$. Under these assumptions, the variance of e_n is $q_a^2/12$, the variance of v is $q_s^2/12$, and it follows from Eq.(4.3.4) that the total quantization noise variance is

$$E\{(S_2-s^2)\} = Nq_a^2/12 + q_s^2/12 \tag{4.3.8}$$

When the full accuracy 2K+1 bit products are used, $e_1=\ldots=e_n$, so that

$$E\{(S_2-S)^2\} = Nq_s^2/12 \tag{4.3.9}$$

If products are rounded to the storage accuracy of K+1 bits, then v=0 and

$$E\{(S_2-S)^2\} = q_s^2/12 \tag{4.3.10}$$

4.3.1 NOISE IN THE OUTPUT OF A RECURSION FILTER CAUSED BY FIXED POINT FINITE WORD LENGTH ARITHMETIC

The noise introduced be finite word length arithmetic can be analyzed by replacing each rounded term by its original value plus an error term limited in magnitude to half the quantization step size. In this section we will use the approximate statistical approach discussed in section 4.3 and assume that the different rounding errors are zero mean, uncorrelated random variables each having variance $q^2/12$ where q is appropriate quantization step size.

The output of a finite tap nonrecursive filter is a weighted sum of inputs. Therefore, the error in the calculation of the present output introduced by finite word length arithmetic does not propagate into the calculation of future outputs. The resulting output noise can be characterized by the

appropriate equation in section 4.3.

The output of a recursive filter is a weighted sum of present and past inputs and past outputs. In this case, the rounding errors propagate into the calculation of successive outputs. To illustrate this effect, let us assume that the pulse transfer function given in Eq.(4.2.1) is implemented using a type 0 direct form realization. If x(nT) is the filter input and y(nT) is its output, then the ideal input output relation is

$$y(nT) = \sum_{k=0}^{M} a_k x(nT-kT) - \sum_{b_k}^{N} y(nT-kT) \tag{4.3.11}$$

We will assume that a_k, b_k, and x(nT) have already been quantized to the required word lengths and that these effect can be analyzed separately. We will assume that overflows do not occur. To simplify the analysis slightly, we will assume that products and the total sum are both rounded to multiples of q. Then the computed and stored output $y_1(nT)$ is

$$y_1(nT) = \sum_{k=0}^{M} [a_k x(nT-kT)]_r - \sum_{k=1}^{N} [b_k y_1(nT-kT)]_r \tag{4.3.12}$$

The rounded products can be written as

$$[a_k x(nT-kT)]_r = a_k x(nT-kT) + e_k(nT) \tag{4.3.13}$$

and

$$[b_k y_1(nT-kT)]_r = b_k y_1(nT-kT) + f_k(nT) \tag{4.3.14}$$

Therefore,

$$y_1(nT) = \sum_{k=0}^{M} a_k x(nT-kT) - \sum_{k=1}^{N} b_k y_1(nT-kT) + e(nT) \tag{4.3.15}$$

where

$$e(nT) = \sum_{k=0}^{M} e_k(nT) - \sum_{k=1}^{N} f_k(nT) \tag{4.3.16}$$

The filter with the roundoff errors is illustrated in Fig. (4.3.1). Assuming that the roundoff errors are zero mean, uncorrelated random variables each with variance $q^2/12$, we find that

$$E\{e^2(nT)\} = \frac{(M+N+1)\,q^2}{12} \tag{4.3.17}$$

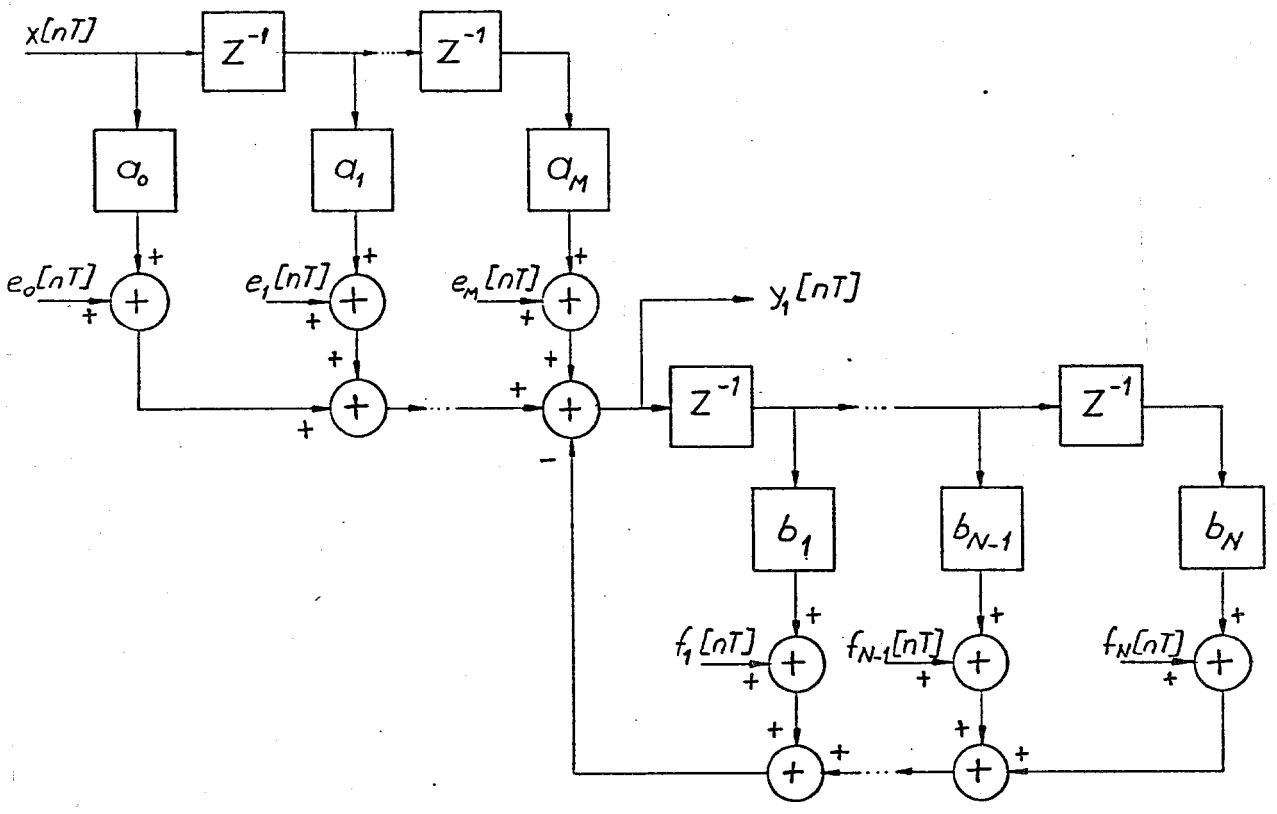


Figure 4.3.1. Noise in a type 0 direct form recursive filter realization caused by fixed point finite word length arithmetic

Taking the Z-transform of Eq.(4.3.15) yields

$$Y_1(z) = Y(z) + V(z) \tag{4.3.18}$$

where

$$Y(z) = X(z)A(z)/B(z) \tag{4.3.19}$$

and

$$V(z) = E(z)/B(z) \tag{4.3.20}$$

Thus the computed output $y_1[nT]$ is the sum of the desired output y[nT] and a noise signal v[nT]. Assuming that e[nT] is a white noise sequence, then by using the average power spectral density formula, the output noise power is

$$E\{v^2(nT)\} = \frac{q^2}{12}(M+N+1)\frac{1}{2\pi j}\oint \frac{1}{B(z)B(z^{-1})}\frac{dz}{z} \tag{4.3.21}$$

where the unit circle can be taken as the contour of integration [38], [39], [40].

The output noise power in parallel and cascade form realizations can be determined using the same approach. The output noise power study is given for parallel and cascade form and direct form realizations is given the below part.

4.3.2. FIXED POINT FILTERS

We shall consider first the direct form of realization and then use the result to treat parallel and cascade forms.

Direct Form: It is seen before that the actual output sequence y[n] is given by

$$y[n] = \sum_{k=0}^{M} (b_k)_t x[n-k] - \sum_{k=1}^{I} (a_k)_t y[n-k] + e_n \tag{4.3.22}$$

where $(a_k)_t$ and $(b_k)_t$ are t-bit fixed-point representations of the coefficients a_k and b_k and ε_n denotes the roundoff error in the calculation of y[n]. From the section 4.2 (error calculation for approximation), we have $(a_k)_t=a_k+\alpha_k$ and $(b_k)_t=b_k+\beta_k$ where α_k and β_k are the coefficient errors.

The error of the nth sample of the output is given by the difference between the actual output y[n] and the ideal output w[n] we have

$$e_n=-\sum_{k=1}^{L} a_k e_{n-k}+u_n \tag{4.3.23}$$

where

$$u_n=\sum_{k=0}^{M} \beta_k x[n-k]-\sum_{k=1}^{L} \alpha_k w[n-k]+e_n \tag{4.3.24}$$

Suppose x[n] is zero mean and WSS. with aoutocorrelation function $R_{xx}(n)$ and power spectral density $S_{xx}(z)$. Then w[n] is zero mean and WSS. with power spectral density $S_{ww}(z)$ given by

$$S_{ww}(z)=H(z)H\left(\frac{1}{z}\right)S_{xx}(z) \tag{4.3.25}$$

It can be shown that u[n] is also zero mean and WSS with the autocorrelation function given by

$$S_{uu}(z)=[B(z)-H(z)A(z)]\left[B\left(\frac{1}{z}\right)A\left(\frac{1}{z}\right)\right]S_{xx}(z)+\sigma^2(\mu+\nu) \tag{4.3.26}$$

where

$$A(z)=\sum_{k=1}^{L} \alpha_k z^{-k} \quad and \quad B(z)=\sum_{k=0}^{M} \beta_k z^{-k} \tag{4.3.27}$$

$\sigma^2=2^{-2t}/3$ is the variance of a random variable uniformly distributed in the interval $(-2^{-t}, 2^{-t})$ and μ and ν are, respectively, the number of b_k and a_k that are neither 1 nor 0. For simplicity, they may be taken to be (M+1) and L, respectively. The error ε_n is zero mean and WSS with

$$S_{ee}(z) = \frac{1}{D(z)D(1/z)} S_{uu(z)}$$

$$- \frac{C(z)C(1/z)}{D(z)D(1/z)} S_{xx}(z) + \frac{(\mu+\nu)\sigma^2}{D(z)D(1/z)}$$

(4.3.28)

where D(z) is the denominator of the transfer function given by

$$D(z) = 1 + \sum_{k=1}^{L} a_k z^{-k}$$

(4.3.29)

and

$$C(z) = B(z) - H(z)A(z)$$

(4.3.30)

The mean-squared value of e_n is then

$$E\{e_n^2\} = \frac{1}{2\pi j} \oint S_{ee}(z) \frac{dz}{z}$$

(4.3.31)

Suppose there is no coefficient rounding error; then the first term in Eq.(4.3.26) and Eq. (4.3.28) is absent and the result is as to be expected. Suppose there in no round off error; then the second term of Eq. (4.3.28) is absent. Thus we see that the error at the output of the filter consists of two components; one is due to roundoff accumulation and the other to the rounding of the coefficients to t bits. The component due to roundoff accumulation is uncorrelated with both the input x[n] and the ideal output w[n]. From Eq.(4.3.23), and Eq.(4.3.24) we can arrive at the block diagram shown in Figure (4.3.2), which will facilitate our discussion of the parallel and cascade realization forms. It is interesting to note that Eq.(4.3.28) can be written down almost by inspection of Figure (4.3.2)

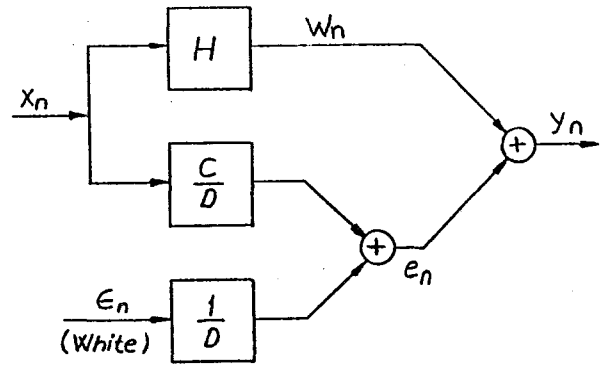


Figure 4.3.2. Round-off error accumulation representation in a filter

Parallel Form: For the parallel form of filter realization, H(z) is written as

$$H(z)=\sum_{i=1}^{K} H_i(z) \tag{4.3.32}$$

where

$$H_i(z)=\frac{N_i(z)}{D_i(z)}=\frac{b_{0i}+b_{1i}/z}{1+a_{1i}/z+a_{2i}/z^2} \tag{4.3.33}$$

Eq. (4.3.33) includes the possibility of a real pole or constant by setting $a_{2i}=b_{1i}=0$ or $a_{1i}=a_{2i}=b_{1i}=0$. The parallel form of implementation is shown in Figure (4.3.3) where K intermediate outputs $w_i[n]$ i=1,2,...,K, are calculated from x[n] and then summed to form the final output w[n].

Suppose the actual coefficients for the ith branch are $(b_{0i})_c, (b_{1i})_c, (a_{1i})_c$, and $(a_{2i})_c$ are related to the ideal coefficients by $(b_{0i})_c=b_{0i}+\beta_{0i}$, $(b_{1i})_c=b_{1i}+\beta_{1i}$, $(a_{1i})_c=a_{1i}+\alpha_{1i}$, and $(a_{2i})_c=a_{2i}+\alpha_{2i}$. Let $y_i[n]$ be the actual output of the ith branch and e_{ni} the error:

$$e_{ni} = y_i[n]-w_i[n] \tag{4.3.34}$$

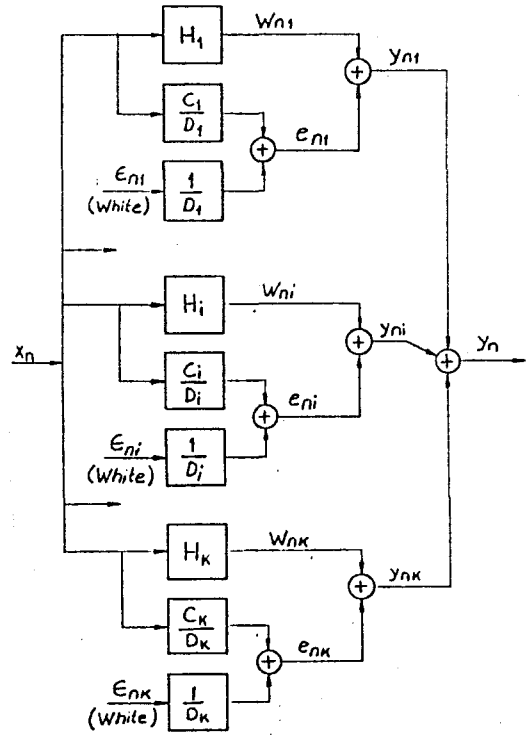


Figure 4.3.3. Round-off error accumulation for parallel form

By using Figure (4.3.2) we can draw a block diagram as shown in Figure (4.3.3), from which one quickly arrives at an expression for the power spectral density of the output error e_n:

$$S_{ee}(z) = S_{zz}(z)\left[\sum_{i=1}^{K}\frac{C_i(z)}{D_i(z)}\right]\left[\sum_{i=1}^{K}\frac{C_i(1/z)}{D_i(1/z)}\right]+\sigma^2\sum_{i=1}^{K}\frac{\mu_i+\nu_i}{D_i(z)D_i(1/z)} \qquad (4.3.35)$$

where

$$\begin{aligned} C_i(z) &= B_i(z) - H_i(z)A_i(z) \\ B_i(z) &= \beta_{0i} + \beta_{1i}z^{-1} \\ A_i(z) &= \alpha_{1i}z^{-1} + \alpha_{2i}z^{-2} \end{aligned} \qquad (4.3.36)$$

and ν_i may both be taken as 2. The mean-squared value of e_n can be computed by using Eq.(4.3.31) and Eq.(4.3.35).

Cascade Form: To realize the digital filter in the cascade form, H(z) is written as

$$H(z) = C\prod_{i=1}^{K}H_i(z) \qquad (4.3.37)$$

where c is a constant which shall be taken as 1 for simplicity, and

$$H_i(z) = \frac{N_i(z)}{D_i(z)} = \frac{1+b_{1i}/z+b_{2i}/z^2}{1+a_{1i}/z+a_{2i}/z^2}$$

(4.3.38)

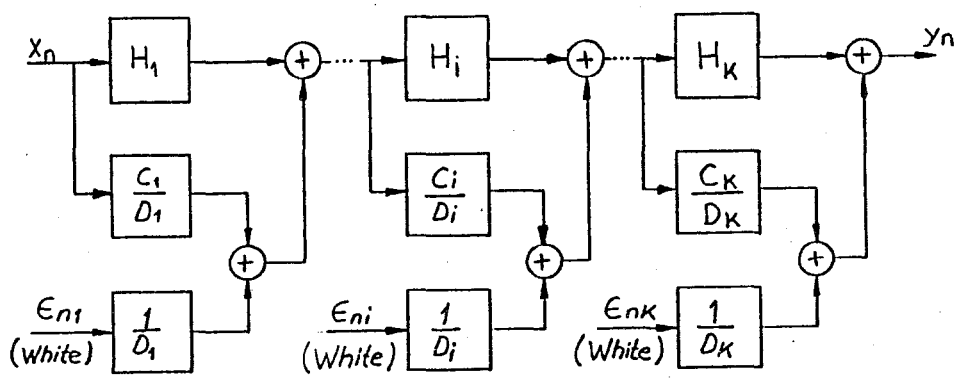


Figure 4.3.4. Round-off error accumulation for cascade form

Notice that the numerator $N_i(z)$ different from that in Eq. (4.3.33). Suppose $(b_{1i})_c$, $(b_{2i})_c$, $(a_{1i})_c$, and $(a_{2i})_c$ are the actual coefficients. Again we have $(b_{1i})_c = b_{1i}+\beta_{1i}$, $(b_{2i})_c = b+_{2i}+\beta_{2i}$, $(a_{1i})_c = a_{1i}+\alpha_{1i}$, and $(a_{2i})_c = a_{2i}+\alpha_{2i}$. By using Figure (4.3.2) , arrive at the block diagram shown in Figure (4.3.4) where

$$C_i(z) = B_i(z) - H_i(z)A_i(z)$$

$$B_i(z) = \beta_{1i}z^{-1}+\beta_{2i}z^{-2}$$

$$A_i(z) = \alpha_{1i}z^{-1}+\alpha_{2i}z^{-2}$$

(4.3.39)

The power spectral density of the actual output y[n] can be determined easily from Figure (4.3.4) by neglecting terms involving fourth or higher powers of sigma. From the expression so obtained, the power spectral density of the ideal output w[n] is subtracted. The remaining part is the power spectral density of the error e_n. The result is

$$S_{ee}(z) = S_{xx}(z) \sum_{i=1}^{K} \frac{C_i(z)\, C_i(1/z)}{D_i(z)\, D_i(1/z)} \prod_{\substack{j=1 \\ j \neq i}}^{K} H_j(z)\, H_j(1/z) +$$

$$+\sigma^2 \left[\frac{\mu_K + \nu_K}{D_K(z)\, D_K(1/z)} + \sum_{i=1}^{K} -1 \frac{\mu_i + \nu_i}{D_i(z)\, D_i(1/z)} \prod_{j=i+1}^{K} H_j(z)\, H_j(1/z) \right] \tag{4.3.40}$$

Both μ_i and ν_i can be taken as 2. The mean-squared value of e_n can be computed by using Eq.(4.3.31) and Eq.(4.3.40).

V. RESULTS and DISCUSSION

V.1. RESEARCH METHODOLOGY

In this study, the given system is considered as an unknown box, and by using the below denoted parametric modelling techniques transfer function in the z domain the response is got and the obtained transfer functions are compared with the original one. White noise input and some other necessary input sequences such as impulse and step inputs are used as driving input at necessary conditions. For modelling cases, Pade' algorithm and ARMA Modified Yule-Walker (MYWE) ARMA algorithm was chosen. The obtained input-output couples were used in the calculation of the transfer function for these algorithms. Two programs were written in Pascal programming language. One is used to find ARMA coefficients by using MYWE, the second one is for Pade' algorithm. These programs are given in Appendix A. For ARMA case, when the results were get, it was seen that Modified Yule-Walker algorithm were not producing the results as good as expected, sometimes it was producing unstable system responses. Another methodology was chosen for ARMA which was AKAIKE algorithm. This algorithm is constructed on the results obtained from MYWE algorithm because AKAIKE algorithm requires an initial estimate of the coefficients and then calculates more accurate coefficients. It was expected that the results would give higher reliability and accuracy than the first algorithms results. The last program for AKAIKE algorithm is also given in Appendix A.

The second concern after finding the transfer function was the stability of the obtained transfer

function. In order to investigate this problem, a program was written. As it is explained in the previous chapters, there are several algorithms to understand whether the system is stable or not. Mainly all of these algorithms depends on the location of the transfer function roots. The program written finds the location of the transfer function roots and decides if the obtained transfer function is stable.

Comparison of the obtained results for all of the algorithms have been done by setting some comparison rules. These are:

a. The response of the system when $t\rightarrow\infty$ for various input sequences
b. Stability of obtained transfer function
c. The values of coefficients
d. The sensitivity of transfer function towards the various error types and the sensitivity of modelling approach to the input quantization, coefficient quantization and round-off error accumulation.

5.2. PRACTICAL RESULTS

For the comparison of these methods two test cases are used. Here in sequence the results obtained will be given.

5.2.1. FIRST TEST CASE:

The first test case has the following transfer function:

$$Real\ H(z) = \frac{0.632z-0.05014}{z^2-0.785z+0.3618}$$

$$= \frac{0.6532z^{-1}-0.05014z^{-2}}{1-0.785z^{-1}+0.3618z^{-2}} \qquad (5.1)$$

COMPARISON OF METHODS FOR TEST CASE 1

a. System response: Response of the simulated trnasfer functions given in Table 5.1. for impulse and step input are shown in Figures (5.1), (5.2), (5.3), (5.4), (5.5), (5.6), (5.7) and (5.8).

Table 5.1 Coefficients of the simulated algorithms for nominator=2 and denominator=2

Coefficients	REAL	PADE'	ARMA MYWE	AKAIKE
a0	0	0.00164	-0.0632	-0.2403
a1	0.632	0.60273	0.2061	0.0143
a2	-0.05014	-0.10930	0.135	0.0256
b0	1	1	1	1
b1	-0.785	-0.85725	-0.70923	-0.917
b2	0.3618	0.35025	0.45068	0.406

Table 5.2. Location of poles for simulated methods

	POLE 1	POLE 2
REAL	0.3925 + j 0.4557	0.3925 - j 0.4557
PADE'	0.4286 + j 0.4081	0.4286 - j 0.4081
ARMA MYWE	0.3516 + j 0.5721	0.3546 - j 0.5721
AKAIKE	0.4589 + j 0.4422	0.4589 - j 0.4422

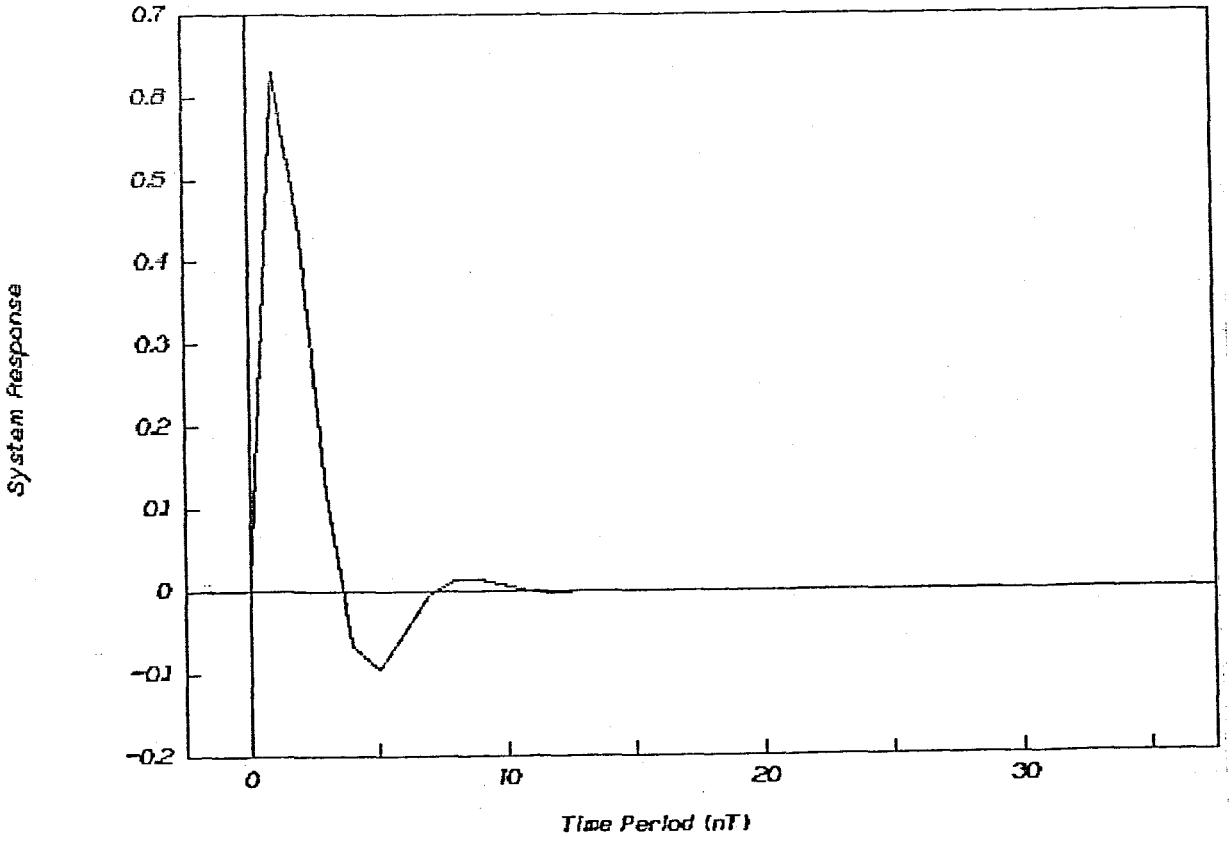


Figure 5.1. Impulse response of Real transfer functions of the first test case

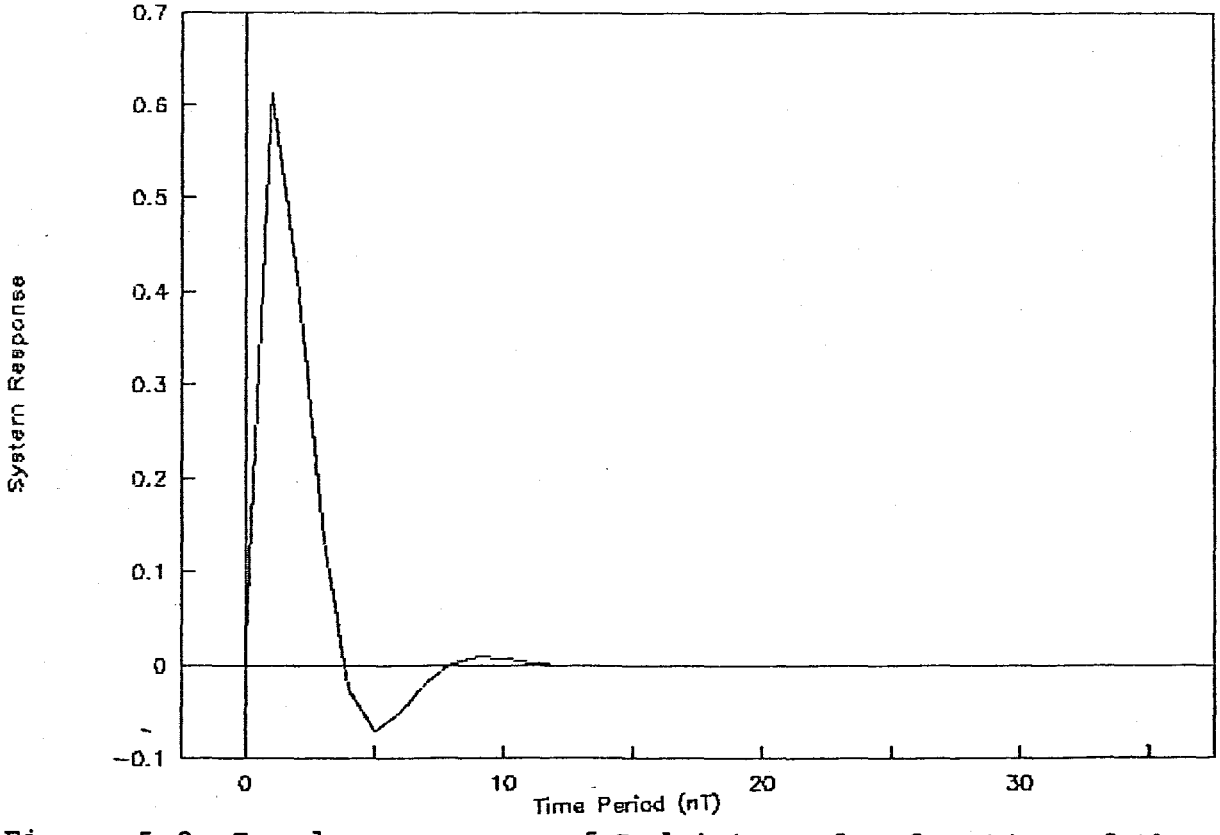


Figure 5.2. Impulse response of Pade' transfer function of the first test case

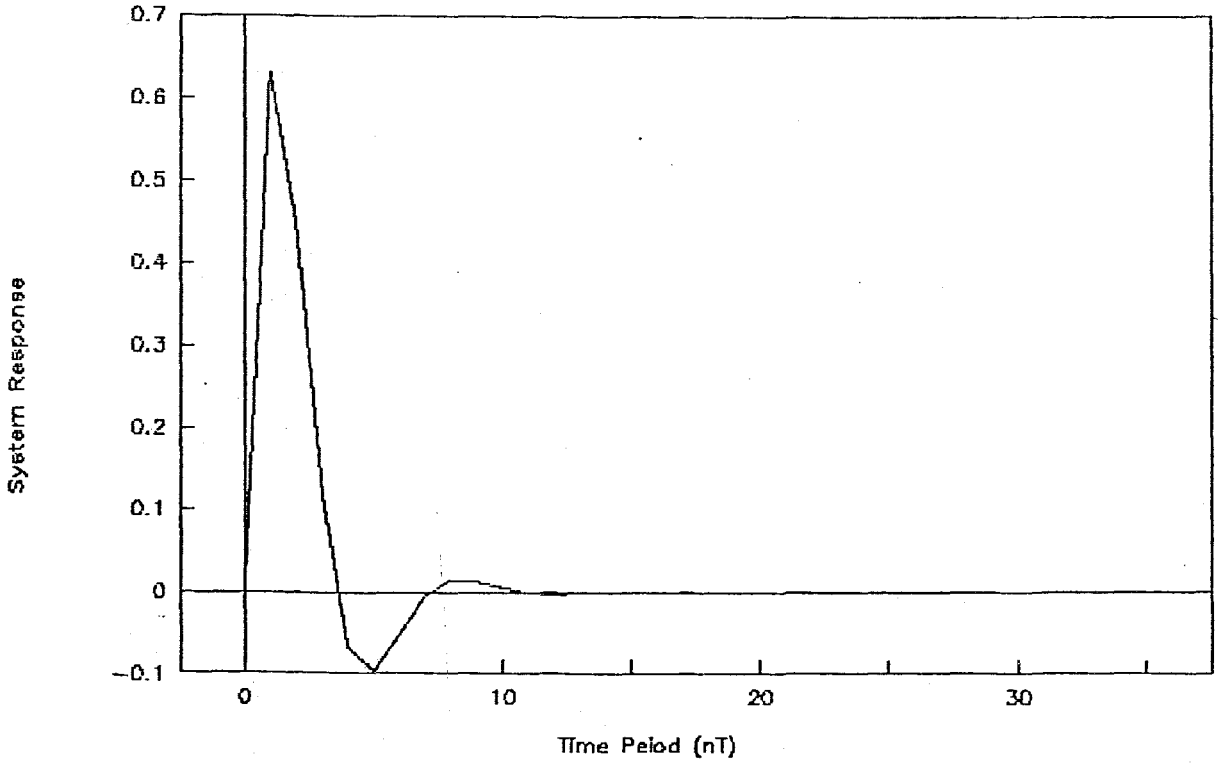


Figure 5.3. Impulse response of ARMA MYWE transfer function for the first test case

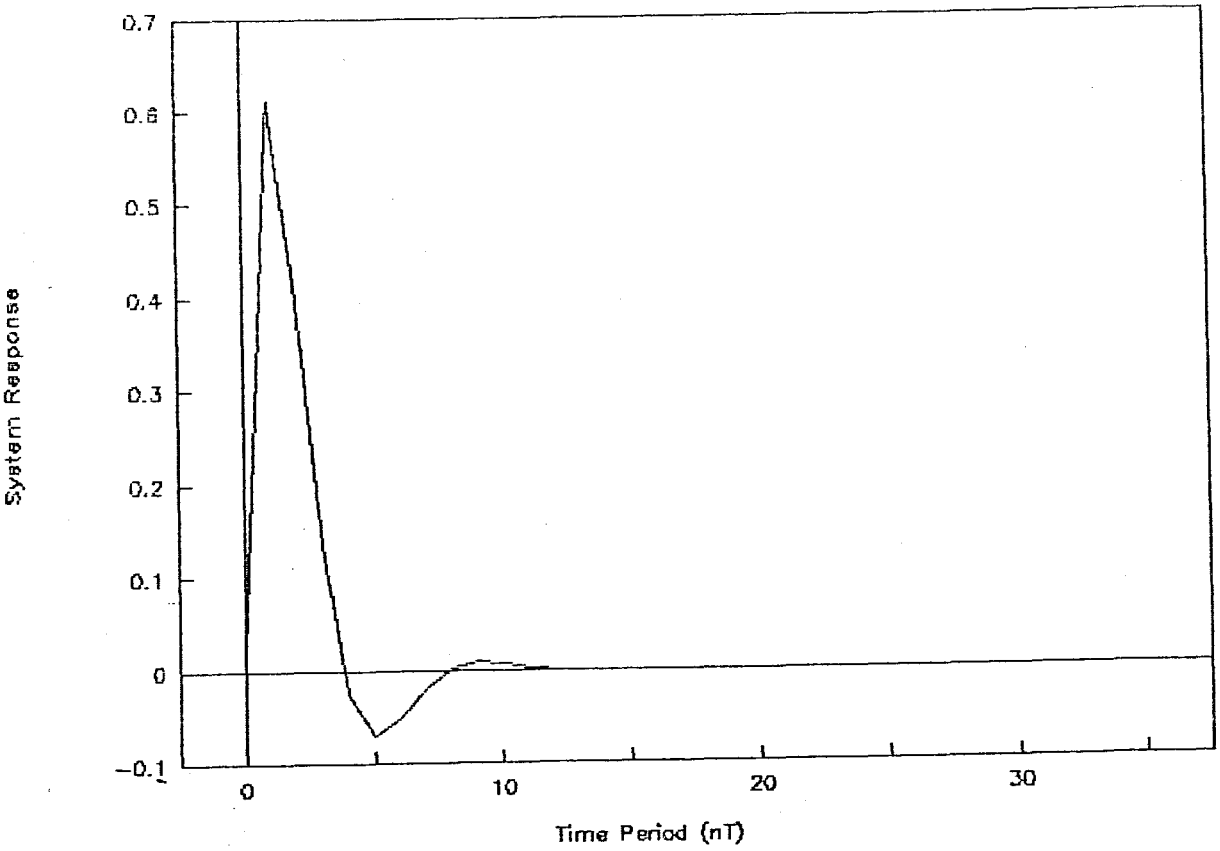


Figure 5.4. Impulse response of ARMA AKAIKE transfer function for the first test case

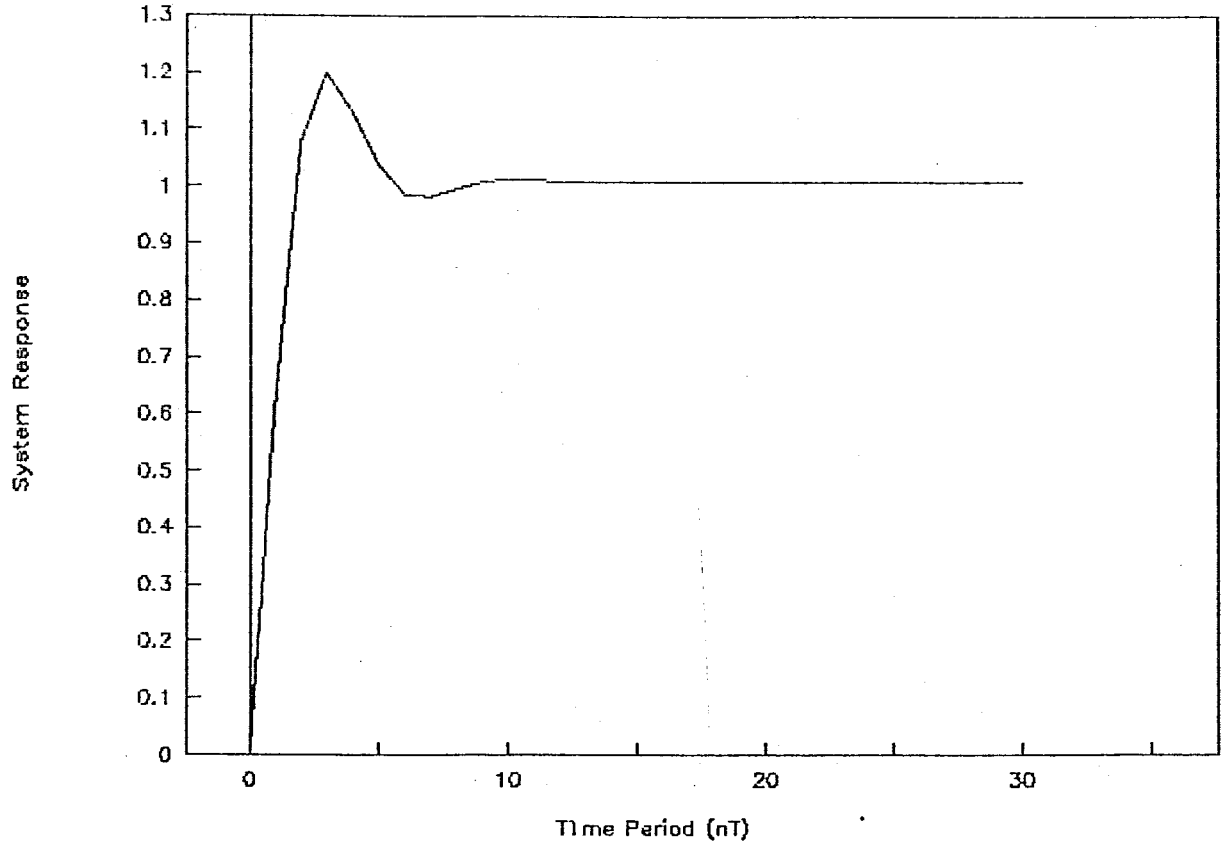


Figure 5.5. Step response of Real transfer function for the first test case

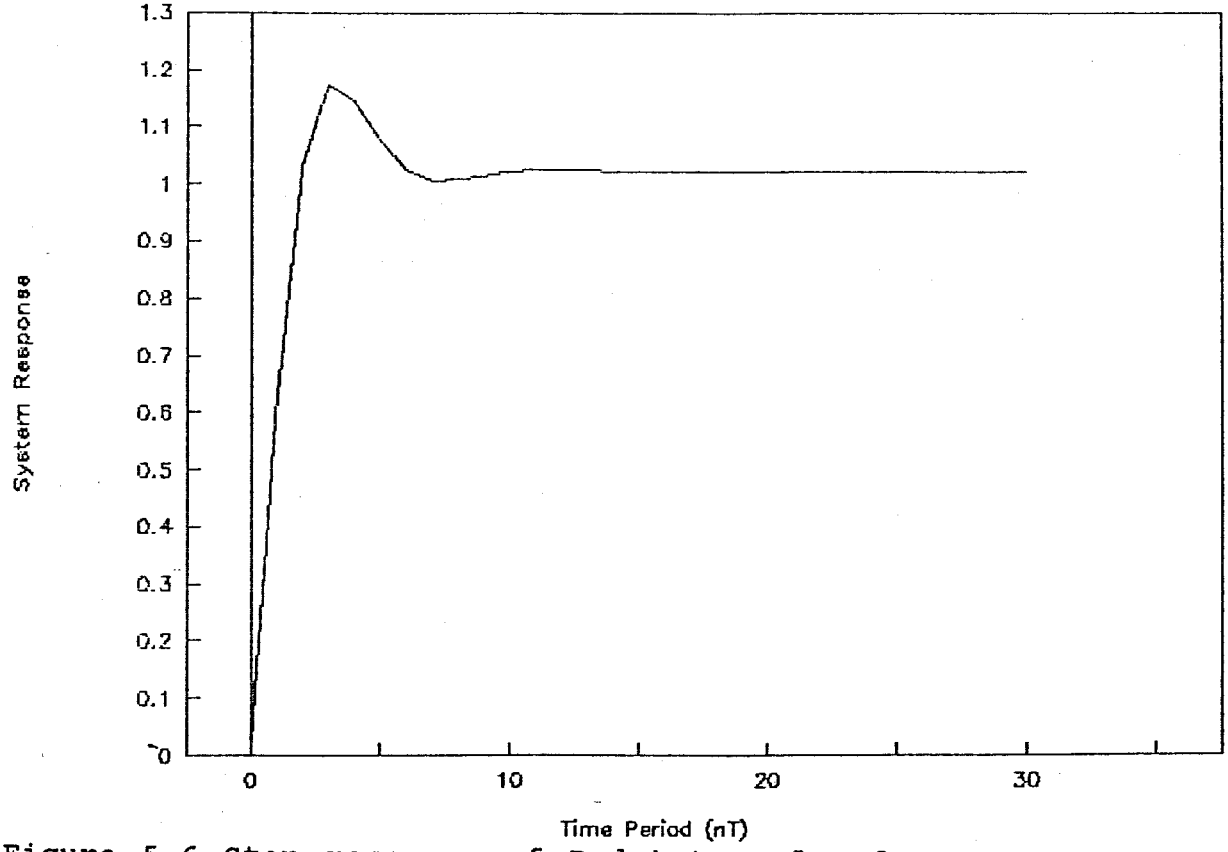


Figure 5.6 Step response of Pade' transfer function for the first test case

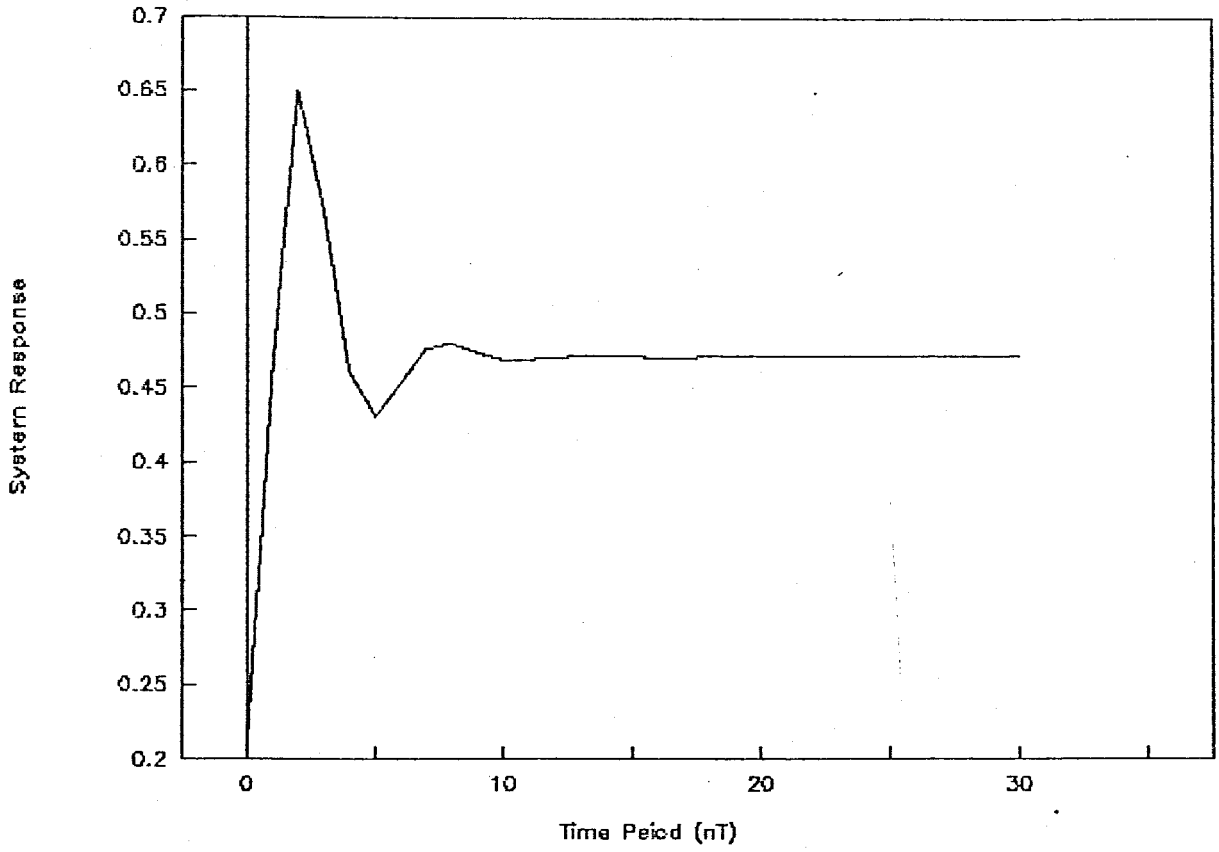


Figure 5.7. Step response of ARMA MYWE transfer function for the first test case

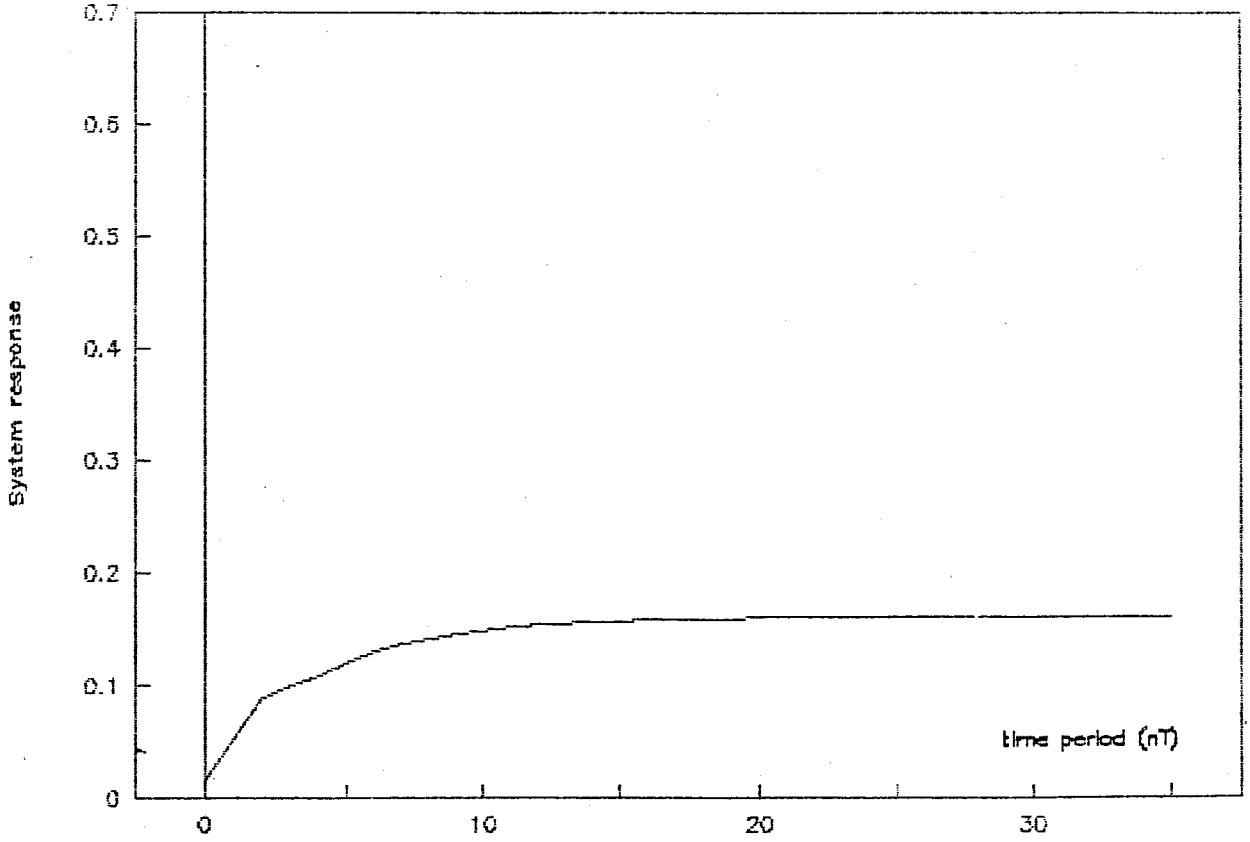


Figure 5.8. Step response of ARMA AKAIKE transfer function for the first test case

Table 5.3. Steady state response of simulated methods

	REAL	PADE'	ARMA MYWE	AKAIKE
x[∞] =	1.00877	1.02231	0.46012	0.161078

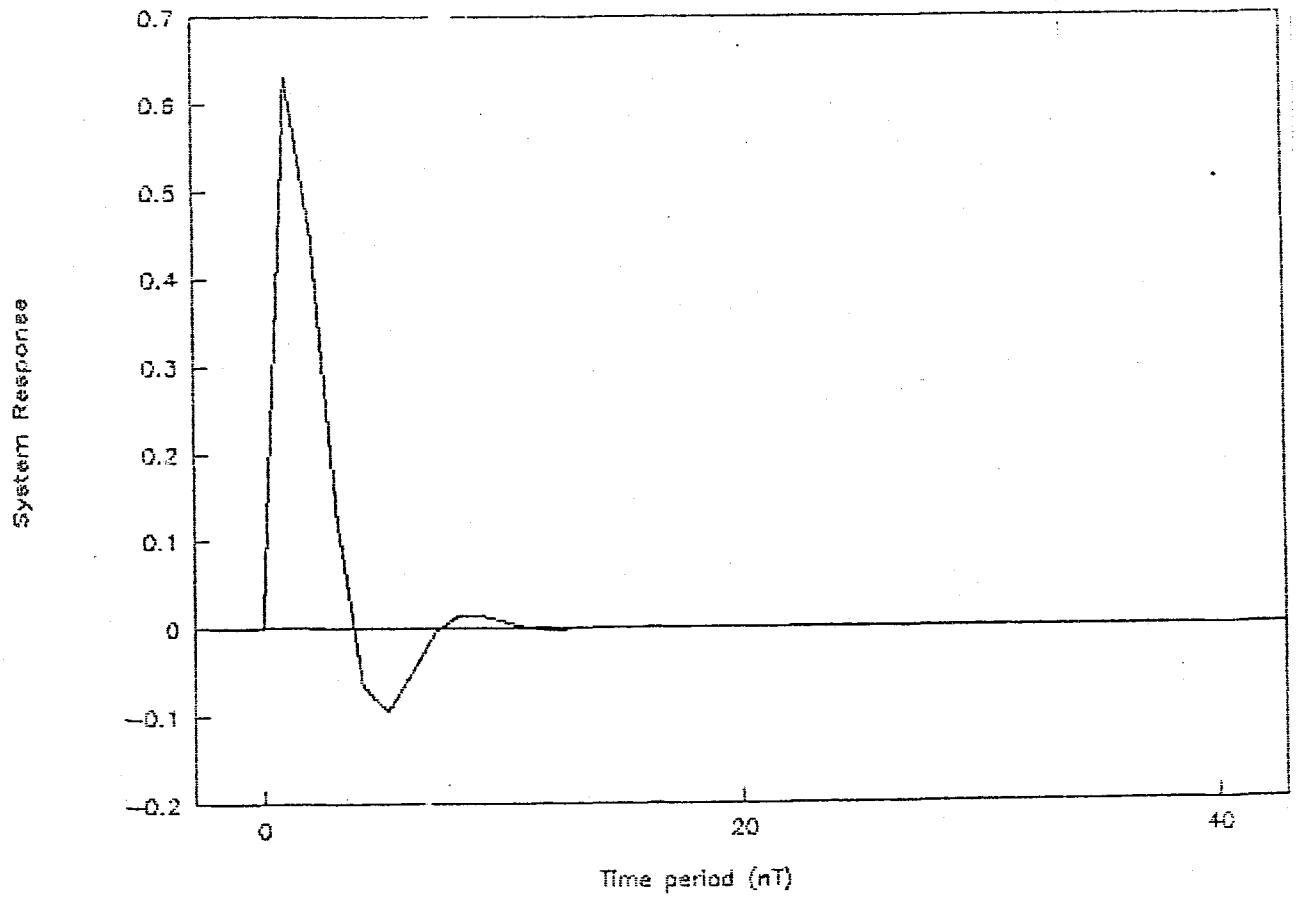


Figure 5.9. Impulse response of higher order ARMA MYWE algorithm for test case 1

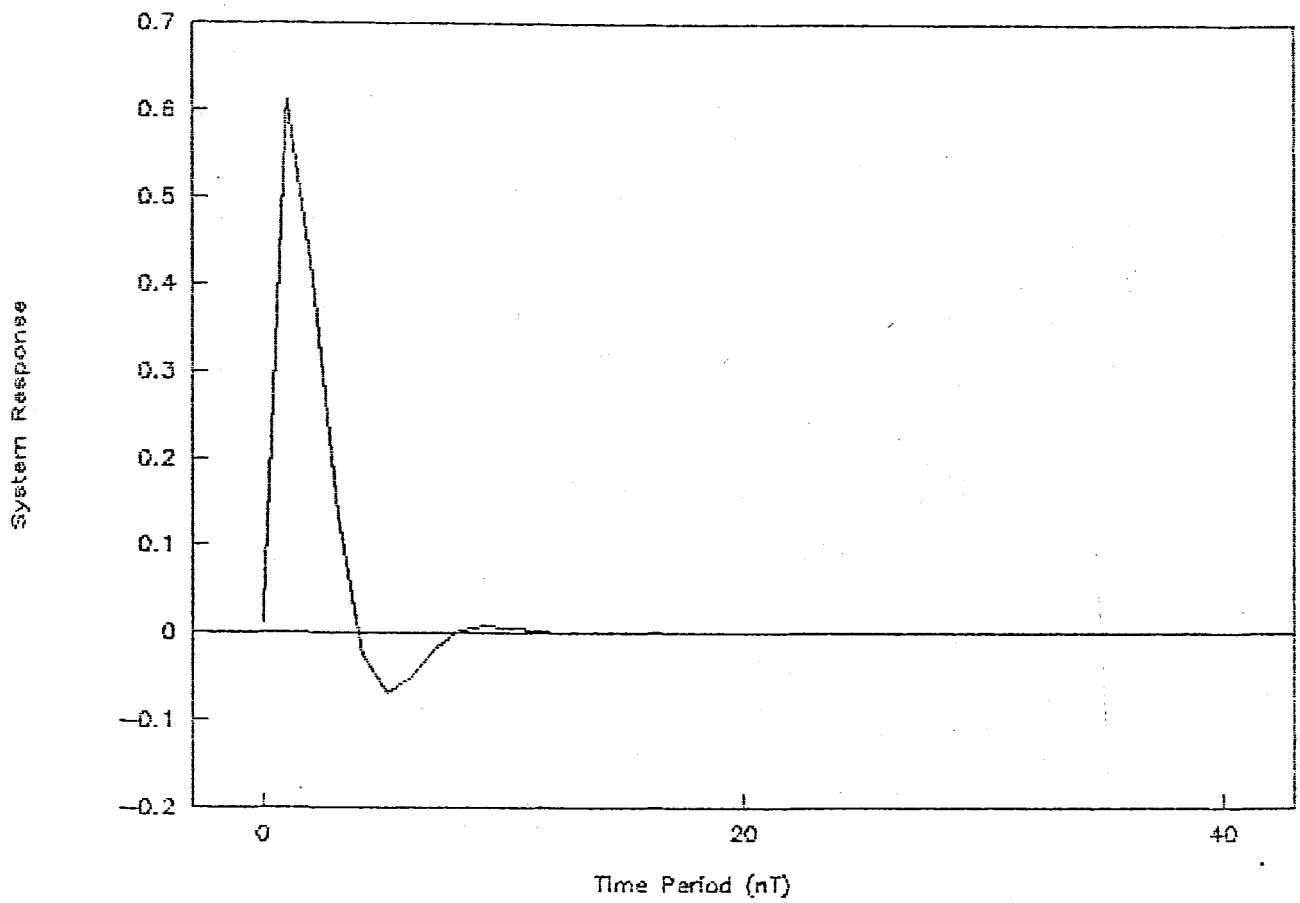


Figure 5.10. Impulse response of higher order ARMA AKAIKE algorithm for test case 1

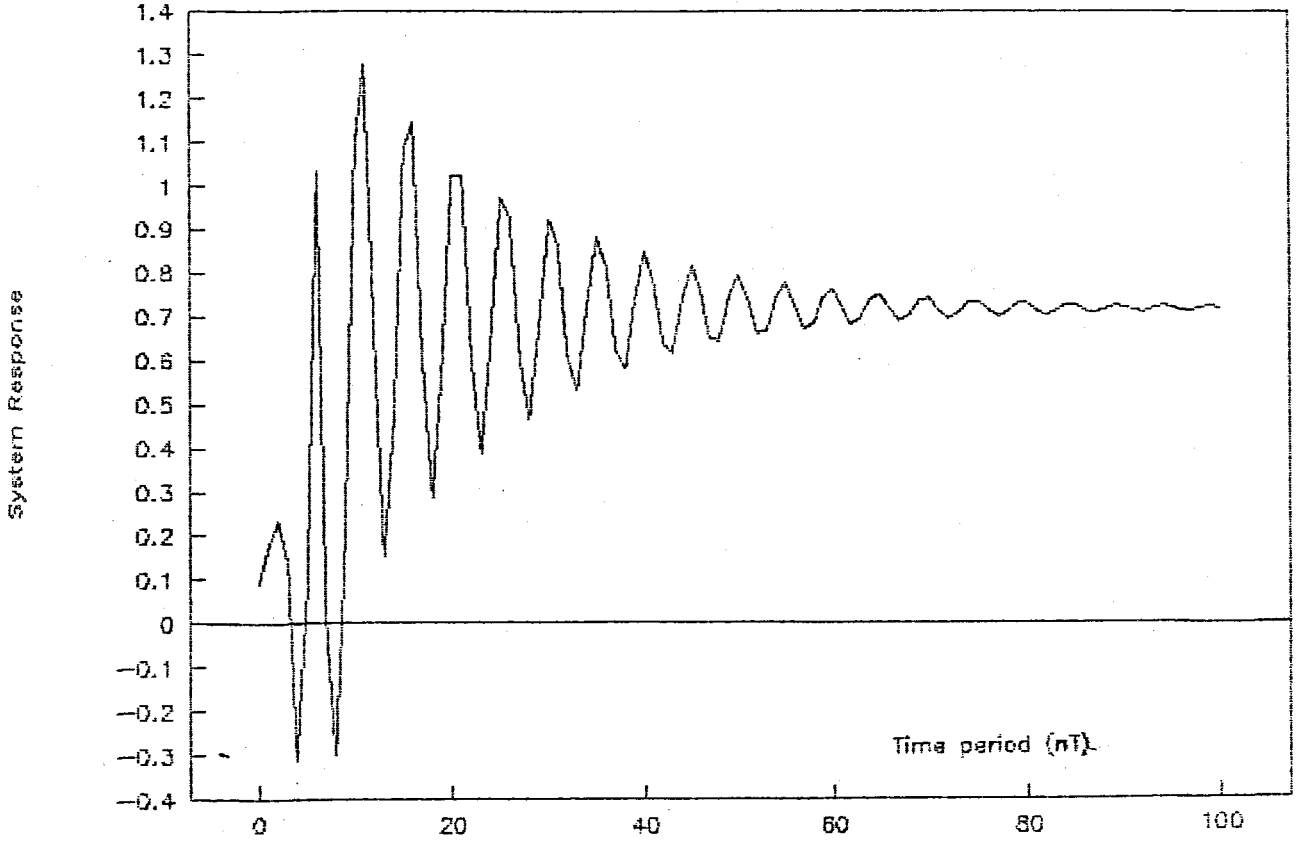


Figure 5.12. Step response of higher order ARMA MYWE algorithm for test case 1

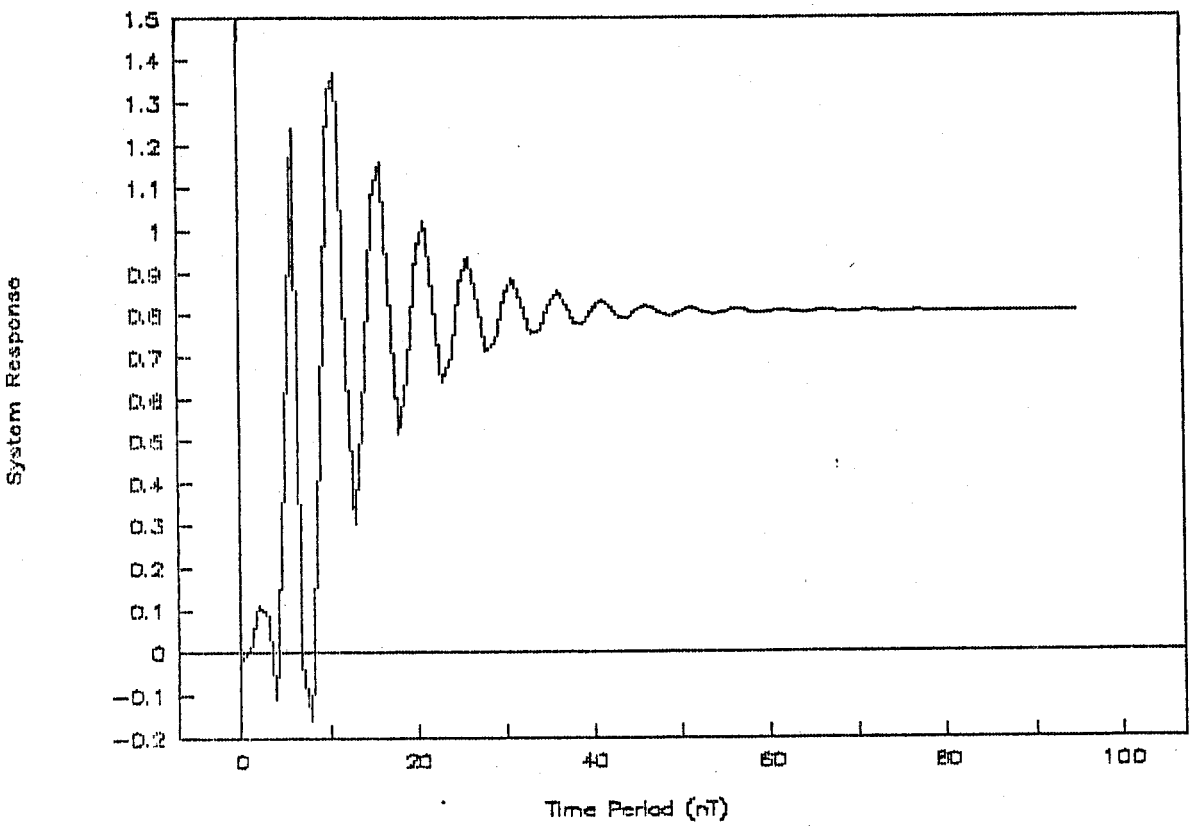


Figure 5.12. Step response of higher order ARMA AKAIKE algorithm for test case 1

As it is seen from the tables both ARMA methods MYWE and AKAIKE did not give expected results according to the real transfer function. On the other hand, that much differences real and calculated results are most probably due to the improper selection of order. If we use any of the mentioned algorithms explained section 3.3.4 a better approximation to the real transfer function can be found. Below a higher order approximation of ARMA process is given for both MYWE and AKAIKE.

Table 5.4. Steady state response of simulated methods ARMA MYWE and AKAIKE for orders (2,8)

	ARMA MYWE	AKAIKE
x[∞] =	0.71819	0.8011

Table 5.5. Coefficients of the simulated method ARMA MYWE and AKAIKE for orders (2,8)

	ARMA MYWE	AKAIKE
a0	-0.3196	-0.0820
a1	0.6359	0.6289
a2	0.1743	0.2647
b0	1	1
b1	-0.9723	-0.7598
b2	1.1305	0.9092
b3	-0.3892	-0.1358
b4	0.0872	-0.0197
b5	0.0046	0.0383
b6	0.0683	0.0719
b7	-0.0870	-0.0830

Table 5.6. Poles of the simulated methods ARMA MYWE and AKAIKE for orders (2,8)

	ARMA MYWE	AKAIKE
POLE 1	-0.4353 + j 0.3462	-0.4868 + j 0.3247
POLE 2	-0.4353 - j 0.3462	-0.4868 - j 0.3247
POLE 3	0.1671 + j 0.8718	0.1568 + j 0.8569
POLE 4	0.1671 - j 0.8718	0.1568 - j 0.8569
POLE 5	0.4493 + j 0.6119	0.4256 + j 0.6080
POLE 6	0.4493 - j 0.6119	0.4256 - j 0.6080
POLE 7	0.6107 + j 0.0000	0.6107 + j 0.0000

b. Stability of systems: In order to analyze the stability of obtained system transfer function for the above mentioned algorithms, We considered the above table and the location of poles on the z domain. From the table (5.7) it can be said that ARMA methodologies have more stable cases than PADE' because ARMA methodology uses higher orders and for this reason approximates the real transfer function easily. And most of the times PADE' produces stable cases for only very low orders. On the higher orders PADE', methodology easily produces unstable system responses, that can be understand by the location of poles.

Table 5.7. Simulation results obtained for methods PADE', ARMA MYWE and AKAIKE at different orders

I	J	PADE'	ARMA MYWE	ARMA AKAIKE
2	2	1.022316171	0.471377019	0.161077049
3	3	UNSTABLE	UNSTABLE	NOT APPL.
5	5	UNSTABLE	UNSTABLE	UNSTABLE
2	3	2.118566025	0.319512547	-0.59949731
3	4	UNSTABLE	UNSTABLE	UNSTABLE
7	8	UNSTABLE	UNSTABLE	UNSTABLE
6	7	UNSTABLE	UNSTABLE	UNSTABLE
8	3	NOT APPL.	UNSTABLE	UNSTABLE
7	3	UNSTABLE	UNSTABLE	UNSTABLE
10	5	NOT APPL.	UNSTABLE	UNSTABLE
4	4	UNSTABLE	UNSTABLE	UNSTABLE
2	7	UNSTABLE	0.718190263	0.808230014
3	8	UNSTABLE	UNSTABLE	UNSTABLE
4	7	UNSTABLE	UNSTABLE	0.974129734
2	8	UNSTABLE	UNSTABLE	UNSTABLE

To see the how the system response can be effected from the finite word length the system response for the different data representation bit lengths must be examined.

The Figure (5.5) shows the logarithmic difference between extended precision (80 bits), double precision (64 bits), real (48 bits) and single precision (32 bits) data representation type in Turbo Pascal on the system response for different methodologies.

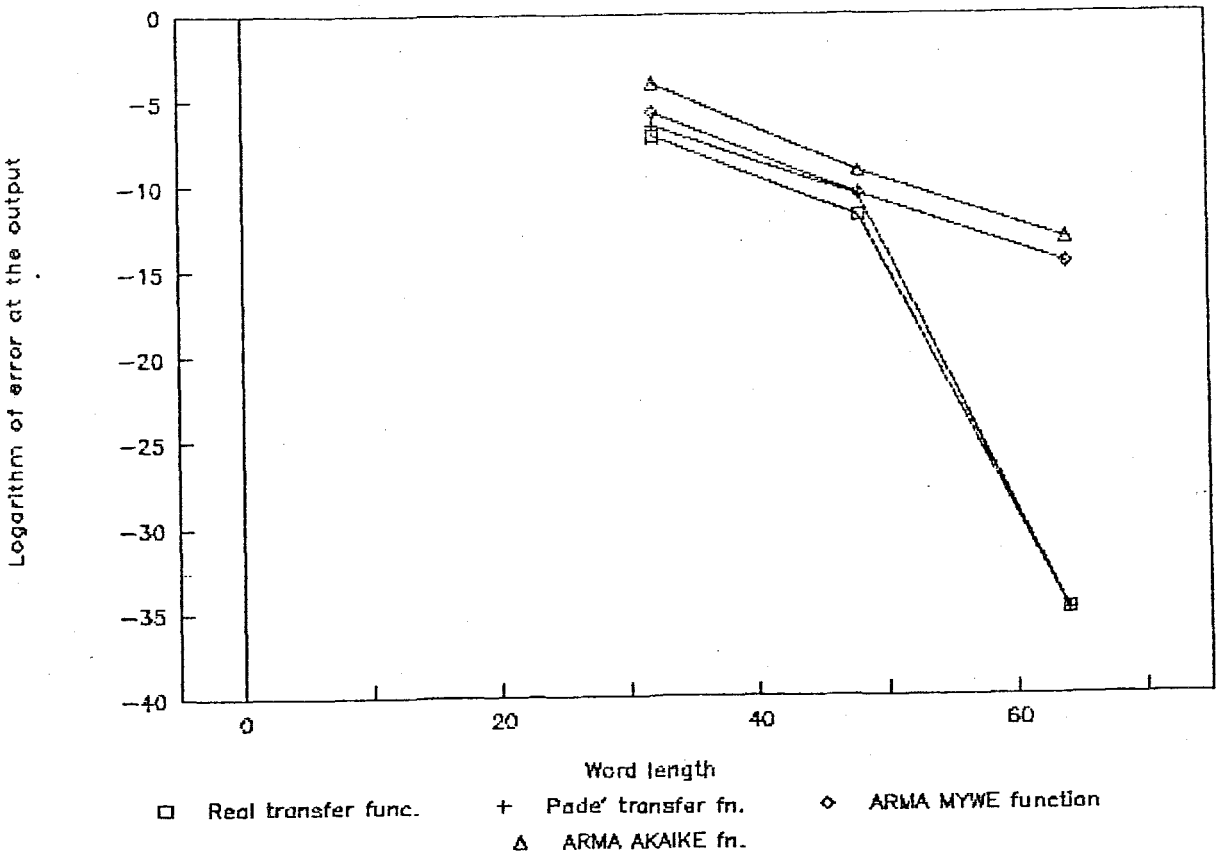


Figure 5.13 Logarithm of the differences of calculated output errors for data representation types

It is obvious that if we use single precision data representation system response is affected more. But on the other hand Pade' approximation is less affected for such data representation changes, after the real transfer function.

c. Coefficient values : One of our criteria was comparison of coefficients values. Here, in table (5.8) the coefficients found by simulated methods are given. This comparison is done at the same order level for all methods as taking the difference between the simulated and real coefficients. This difference is given in figure (5.14).

Table 5.8. Coefficients differences between real and simulated results

	REAL	PADE'	ARMA MYWE	AKAIKE
a0	0	0.01064	0.06327	0.24039
a1	0	0.02926	0.42589	0.61769
a2	0	0.0519	0.16356	0.07578
b0	0	0	0	0
b1	0	0.07223	0.07568	0.13298
b2	0	0.01559	0.08882	0.04441

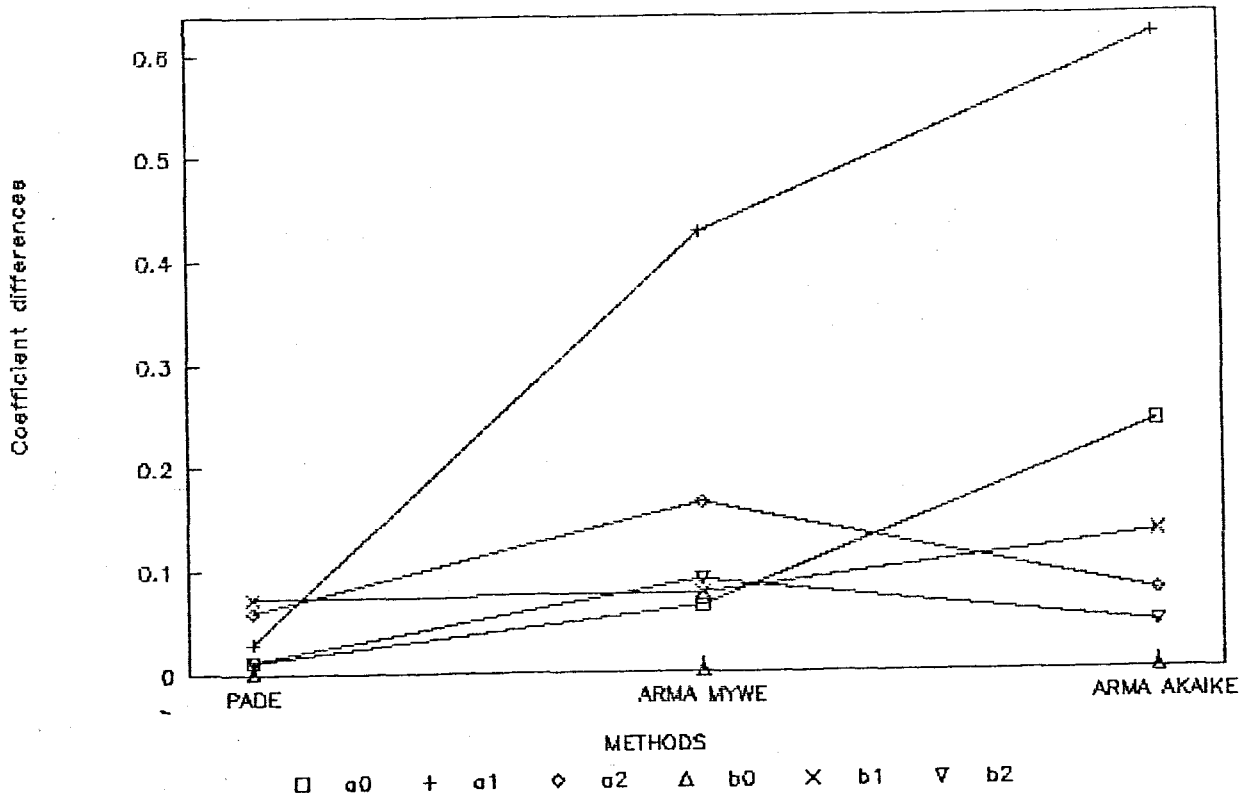


Figure 5.14. Coefficients error of simulated methods according to the extended data type

From figure (5.14) we can say that Pade' has better coefficient values than the other for the order (2,2). This means that, Pade' method has a better approximation to the real transfer function.

d. Sensitivity to error: Under this subject we will analyze mainly the effects of finite word length on the system. These effects are theoretically explained in chapter 4.

1. Input quantization error: This error is directly introduced by the quantization of the input signal. As it is explained in section 4.2, it can be modelled as $q^2/12$ by simply adding a zero mean white noise sequence of variance $q^2/12$ to the original unquantized discrete-time signal.

2. Effects of coefficient inaccuracy: As it is mentioned in section 4.2, under this effect the effect of coefficient quantization on the frequency response of a transfer function is considered by examining its effect on the location of the poles and zeros of the filter. The numerical algorithms to find zeros and poles of a transfer function (mainly roots of a polynomial) are based on some approximation techniques, so it is not a good idea to use the approximated roots for comparison. Here we compare the effect of finite word length on the obtained transfer function coefficients. This effect will be calculated by assuming that the solution found using the extended precision data type of pascal programming language is the true case. Other data types such as single, real and double are used to show the effect of short word length. See table (5.9).

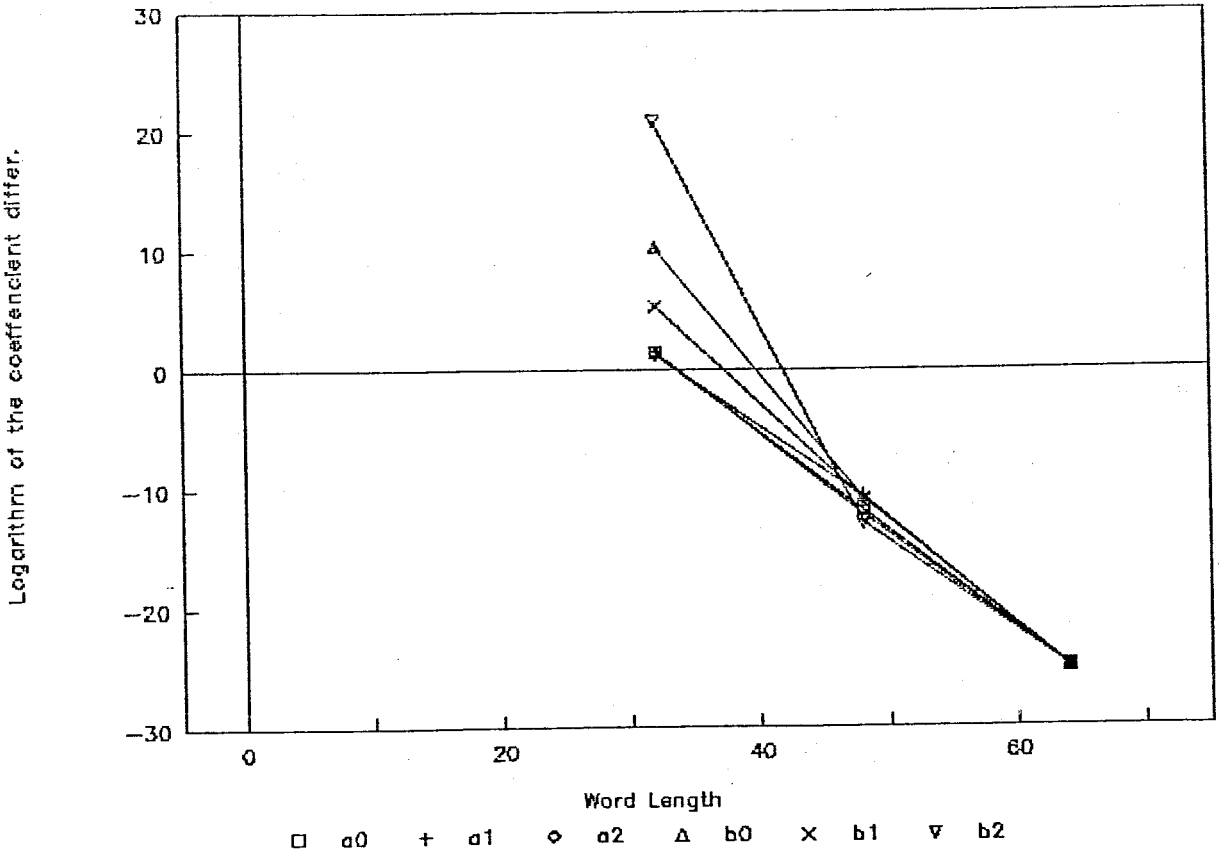


Figure 5.15 Real transfer function : Logarithm of coefficient errors according to the change of word length

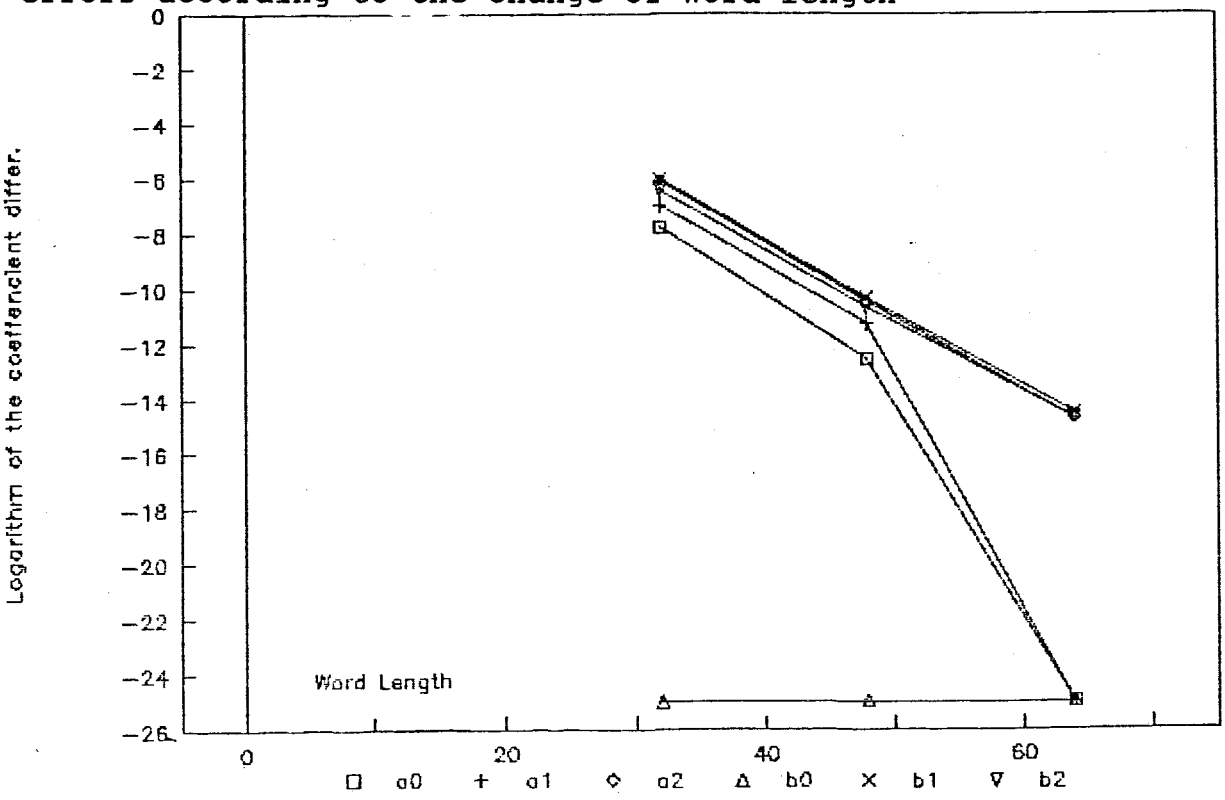


Figure 5.16 Pade transfer function : Logarithm of coefficient errors according to the change of word length

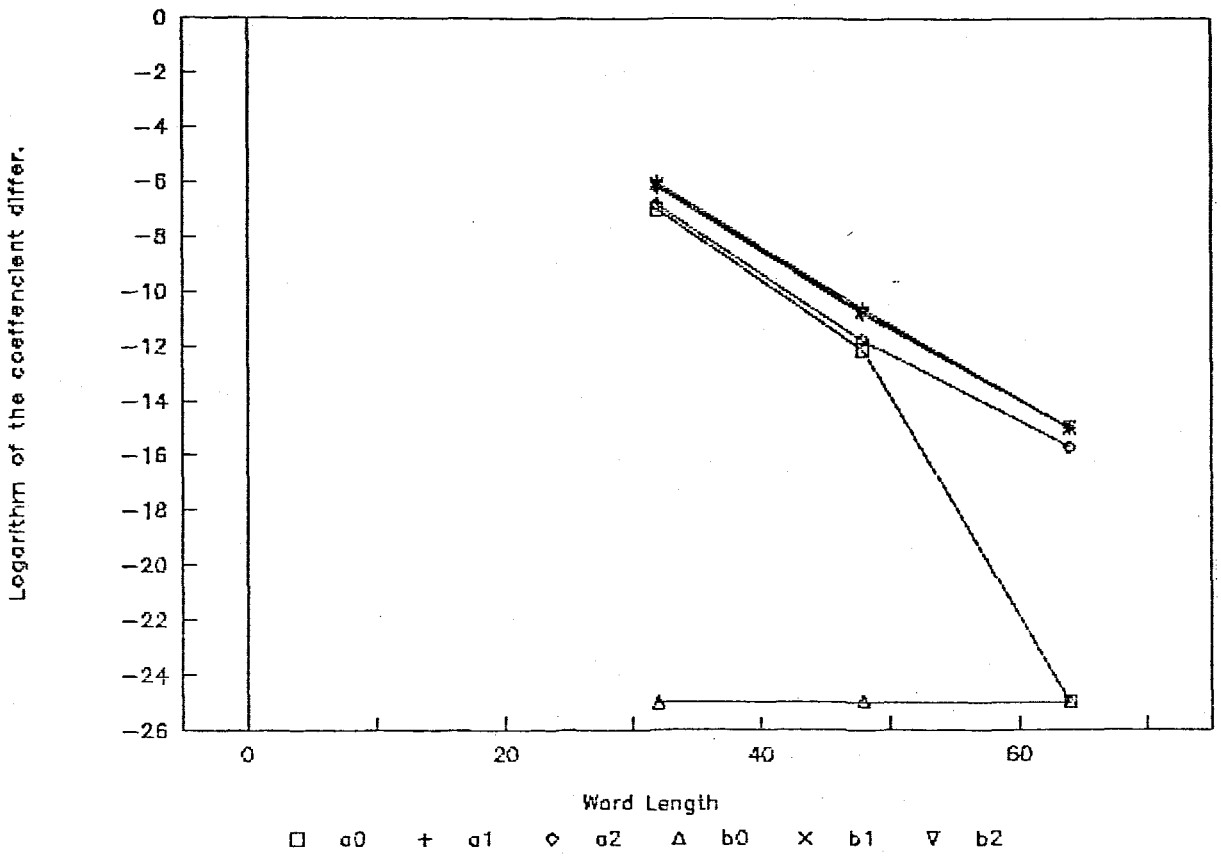


Figure 5.17 ARMA MYWE transfer function : Logarithm of coefficient errors according to the change of word length

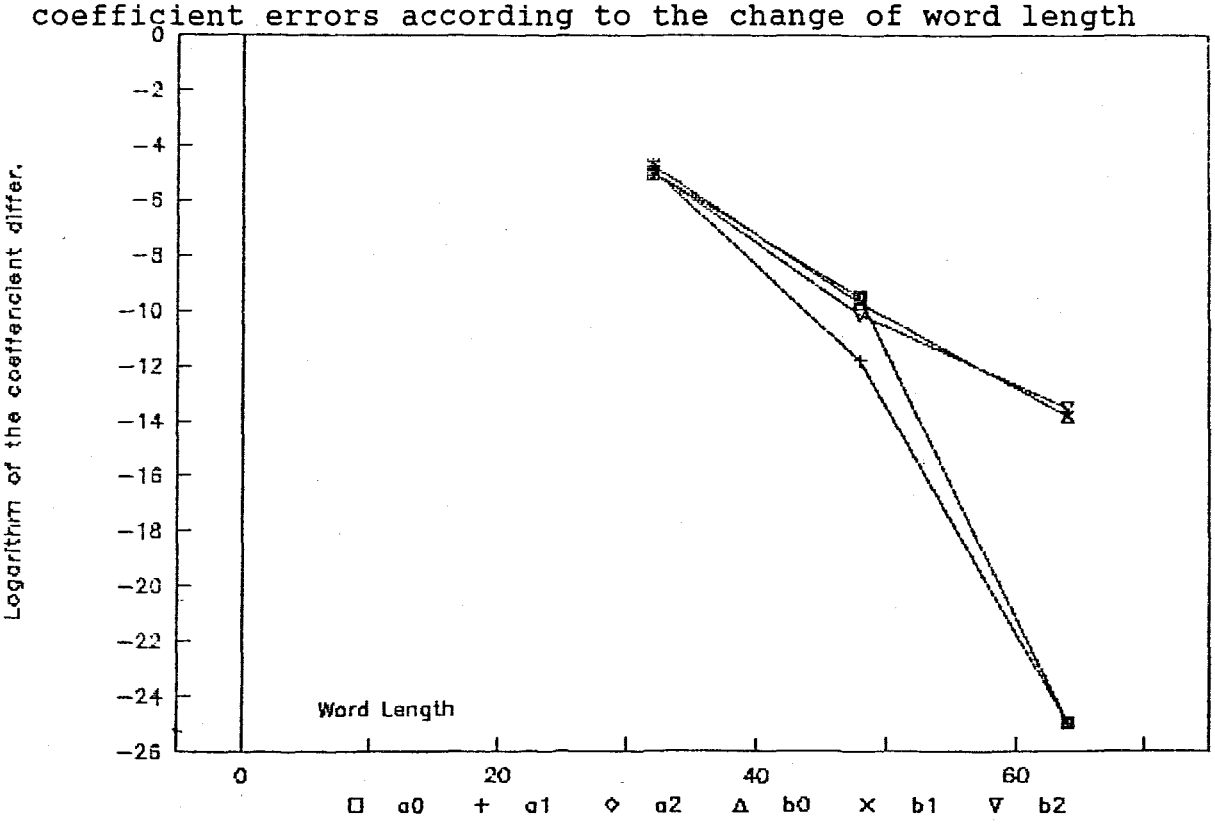


Figure 5.18 ARMA AKAIKE transfer function : Logarithm of coefficient errors according to the change of word length

From the figures it is seen that all methodologies approximately produces the same amount of error for the same order except the AKAIKE algorithm, because of the fact that it is based on the ARMA algorithm it is effected double from such truncation. It is obvious that Pade' is producing the obtained results in a lower order transfer function than ARMA so Pade' is effected less relatively to the ARMA if the orders level is considered.

3. The output of system is a weighted sum of present and past inputs and outputs. The rounding errors propagate into the calculation of successive outputs, according to the equation (4.3.21)

Here we will calculate this propagated error. Assuming that round-off errors are zero mean and uncorrolated random variables each with variance $q^2/12$, output noise power is

$$E\{v^2[nT)\} = \frac{q^2}{12}(M+N+1)\frac{1}{2\pi j}\oint \frac{1}{b(z)\,bz^{-1}}\frac{dz}{z} \qquad (5.2)$$

Where the unit circle is used as the contour of integration [34], [39], [42], [43].

Table 5.9. Calculated errors for test case 1 in state space direct form (all numbers should be multiplied by q^2)

I	J	PADE'	ARMA MYWE	ARMA AKAIKE
2	2	-0.918791778	-0.624480090	-0.881202293
2	3	-0.198390581	-0.041059074	-0.955662787
7	8	-	7.698239885	-
2	7	-	2.266102243	-0.070892356
4	7	-	-	-0.484446167

The round-off errors for test case 1 according to the methodologies are given in the table (5.10), the round-off errors are calculated only when stable transfer functions are obtained.

The second subject to consider error was the state-space representations sensitivity according to the various (especially cascade and parallel) representation types. As it is explained in section 4 especially, cascade and parallel state space representation types are less sensitive to the round-off errors. So, here we will investigate this on our Pade', ARMA, and AKAIKE modellings. These errors are calculated using residu theorem [44], [45].

Table 5.10. Output noise power of test case 1 for different state space representation types (all numbers should be multiplied with q^2)

Repr. type	REAL	PADE'	ARMA MYWE	ARMA AKAIKE
Direct Form	0.105711	0.918791	0.624480	0.8812
Parallel form	0.105	0.03222	0.0837	0.04403

The practical results shows the same thing with theoretical formulas. Pade' approximation technique is less sensitive to the round-off errors according to the ARMA and AKAIKE method for same order levels.

V.2.2 SECOND TEST CASE:

The second test case was have the following transfer function

$$Real\ H(z) = \frac{0.236z^2+0.36z-0.785}{z^3-0.3618z^2+0.9z-0.4596}$$

In this second case, for the lower orders Pade' approximation has not producing a good results. ARMA approximation is unstable but AKAIKE algorithm's result is better than other two. The transfer function of the second case, as it can easily be seen from all figures above, is have a high overshoot and it is reaching its steady-state value in a long time of period. This transfer function has a pole very close to the unit circle. So it is very easy to carry it by approximating outside of the unit circle like in ARMA MYWE case.

Table 5.11. Coefficients of transfer functions of test case 2 for simulated methods

	REAL	PADE'	ARMA MYWE	AKAIKE
a0	0	0.06318	-0.15873	0.15315
a1	0.236	0.12055	0.22340	-0.23252
a2	0.36	0.41355	0.08721	-0.15971
a3	-0.785	-0.75204	-0.06013	-0.12042
b0	1	1	1	1
b1	-0.3618	-0.32550	-0.30664	0.27452
b2	0.9	0.84945	0.94307	0.97841
b3	-0.4596	-0.41215	-0.12085	0.13653

Here, the step responses is given as a table for various orders. From this table, we can say that Pade' approximation is not successful for second case. Maybe, the reason of that is the transfer functions property.

COMPARISON OF METHODS FOR TEST CASE 2:

a. System response : When the impulse signal is applied to the systems the responses in figures (5.19), (5.20), (5.21) and (5.22) are obtained. And the step response is given in

figure (5.23), (5.24), (5.25) and (5.26).

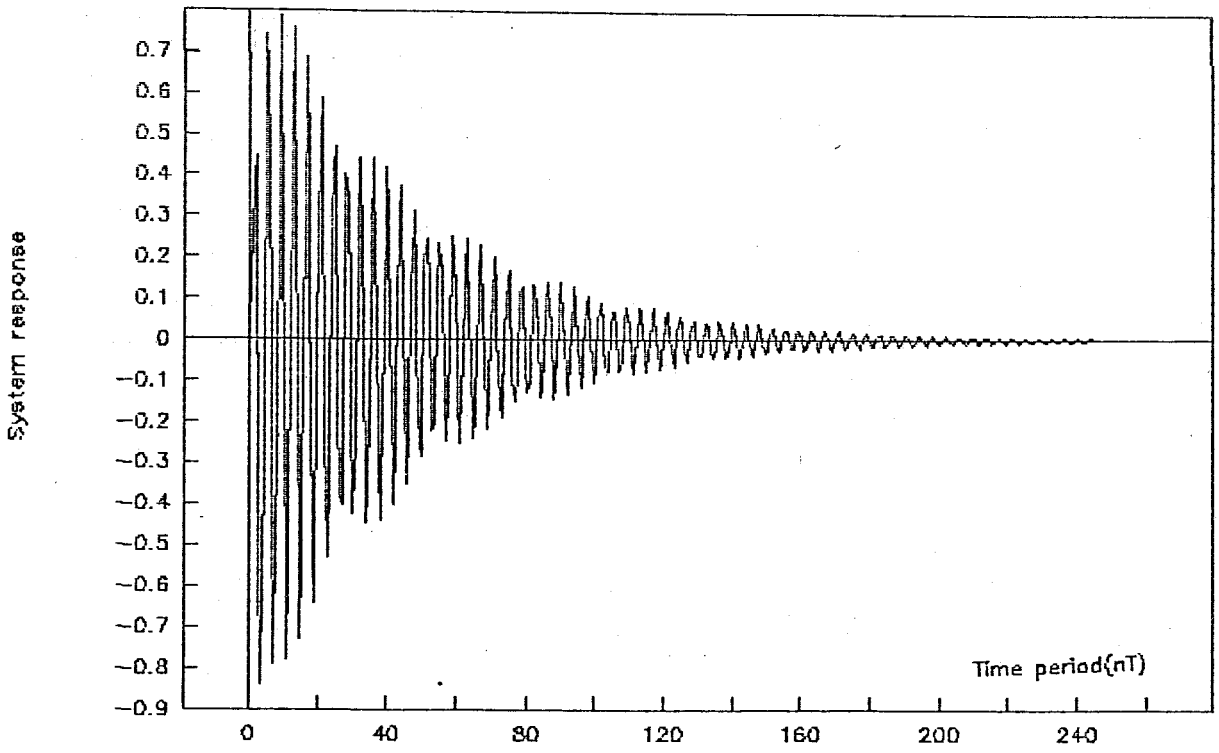


Figure 5.19. Impulse response of real transfer function for the test case 2

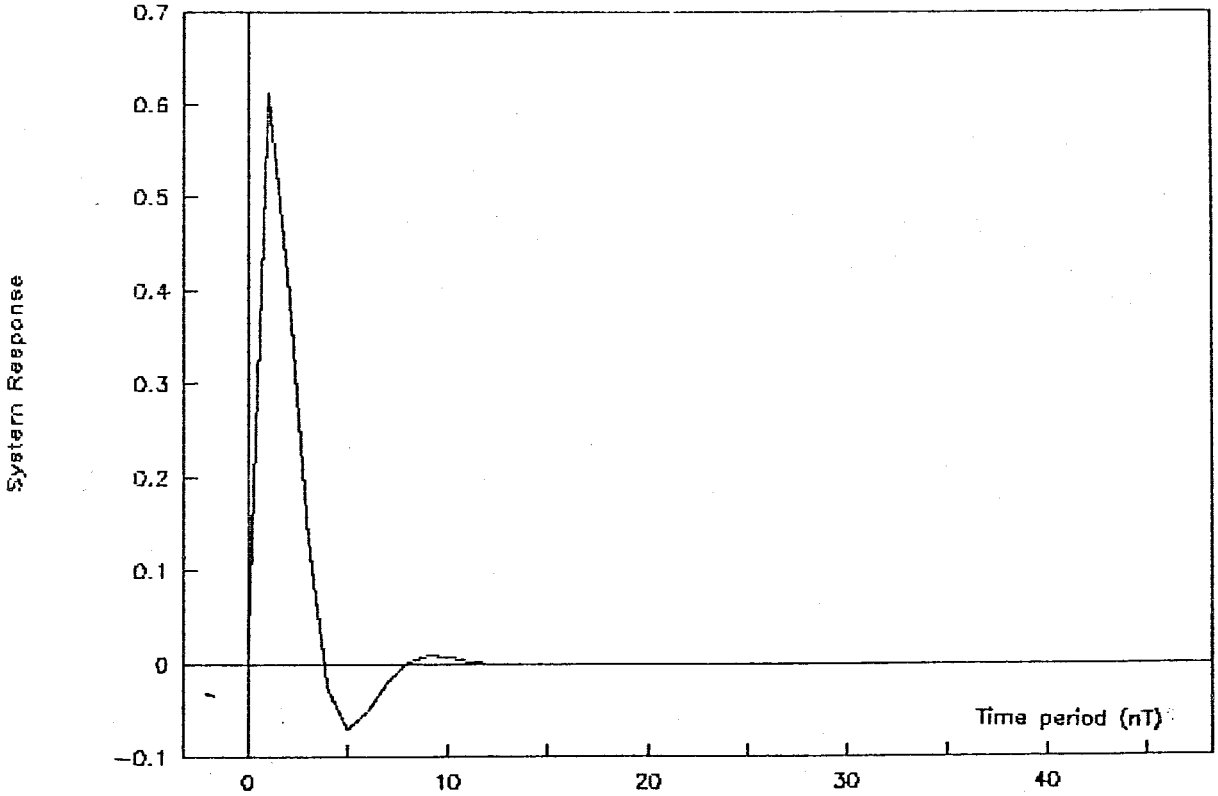


Figure 5.20. Impulse response of Pade' transfer function for the test case 2

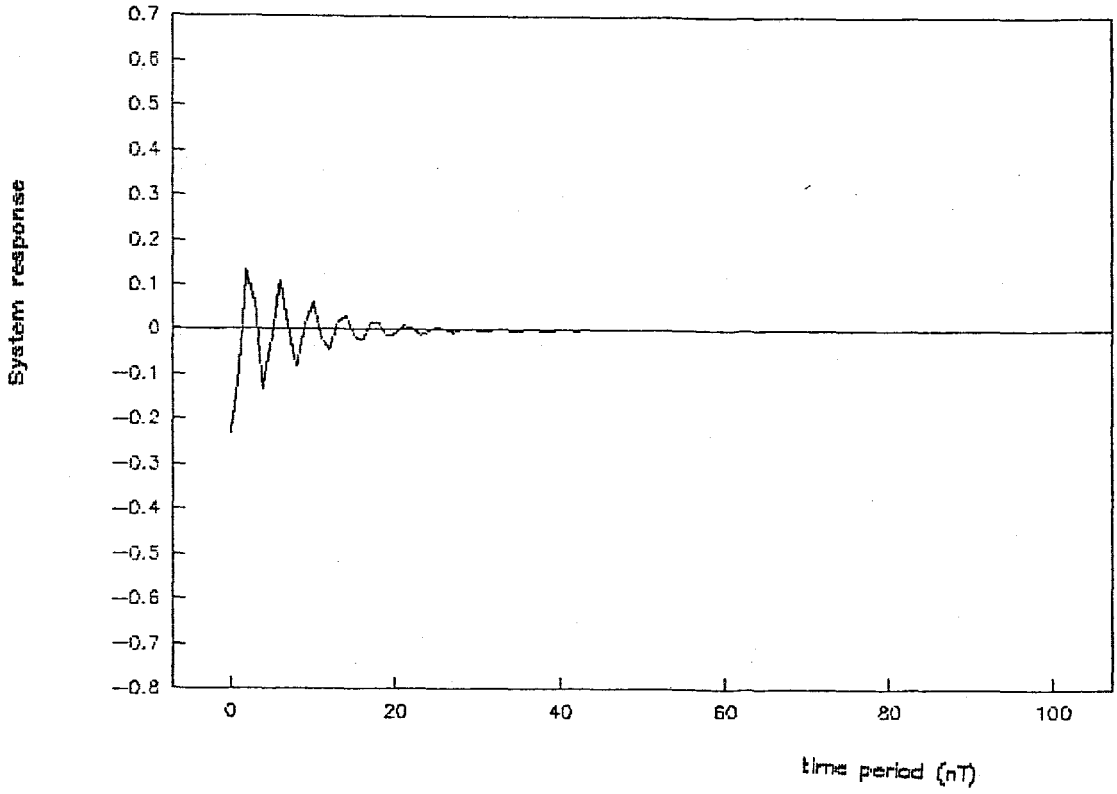


Figure 5.21. Impulse response of ARMA MYWE transfer function for the test case 2

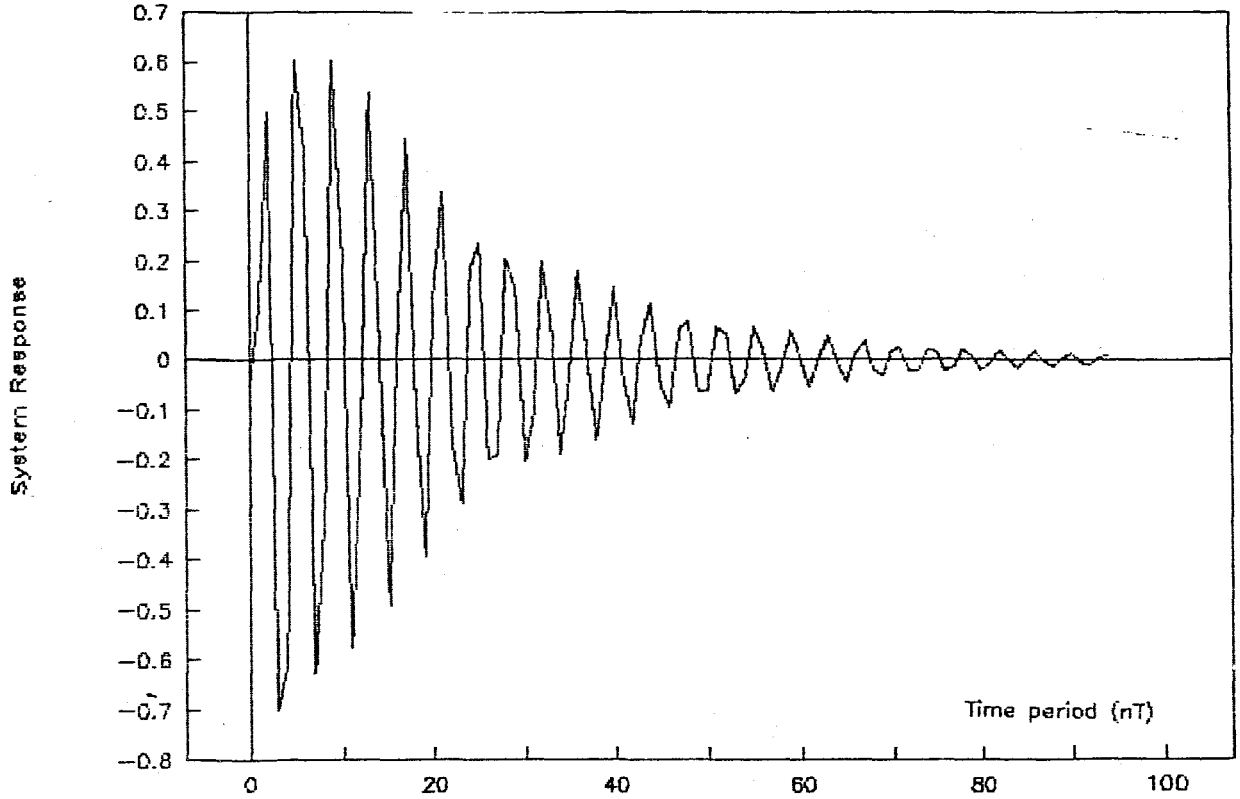


Figure 5.22 Impulse response of ARMA AKAIKE transfer function for the test case 2

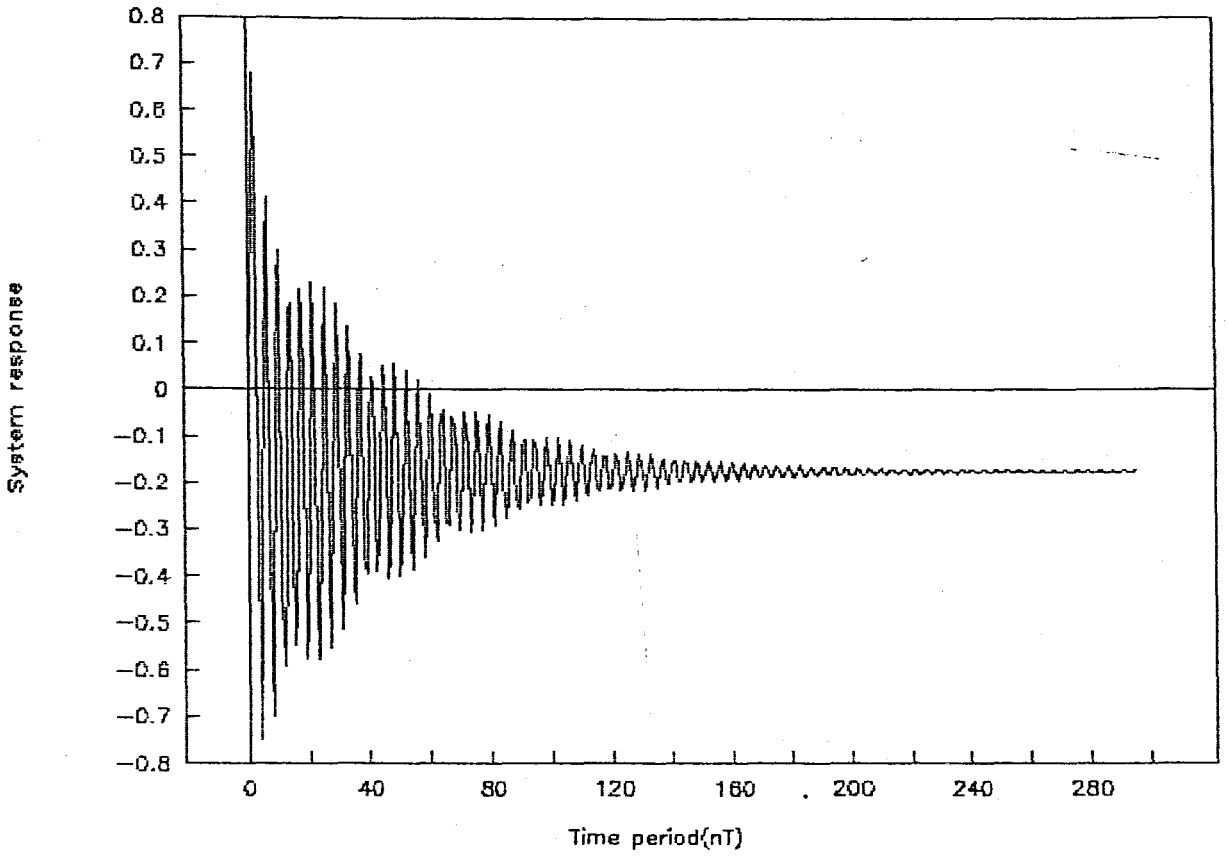


Figure 5.23 Step response of Real transfer function for the test case 2

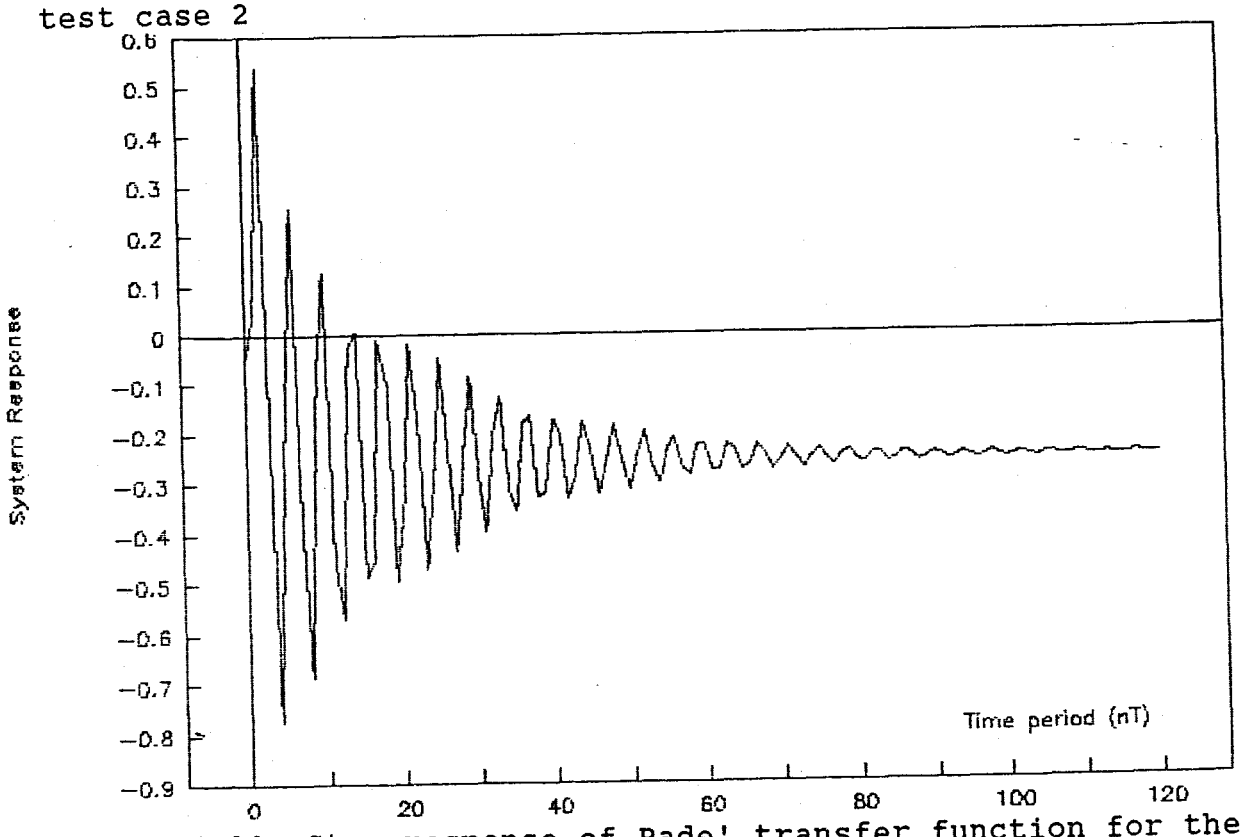


Figure 5.24. Step response of Pade' transfer function for the test case 2

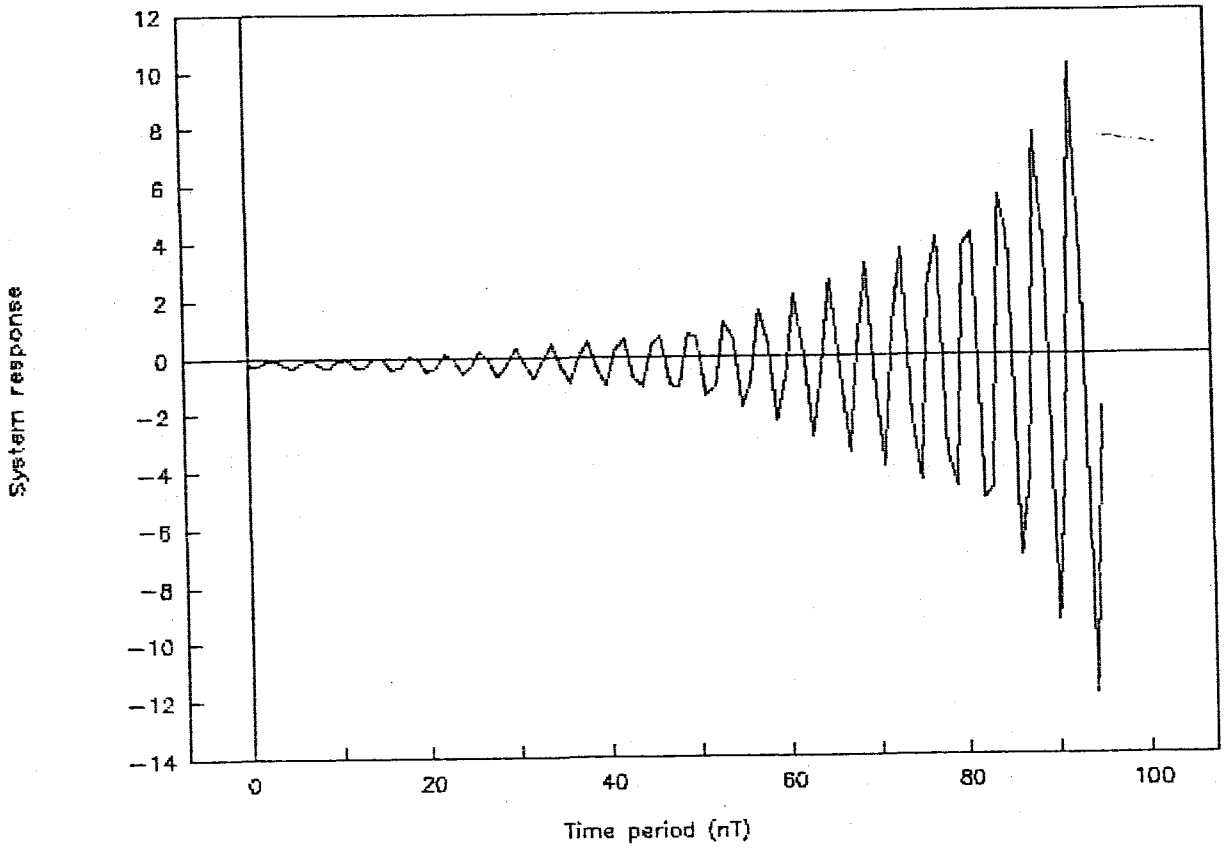


Figure 5.25. Step response of ARMA MYWE transfer function for the test case 2

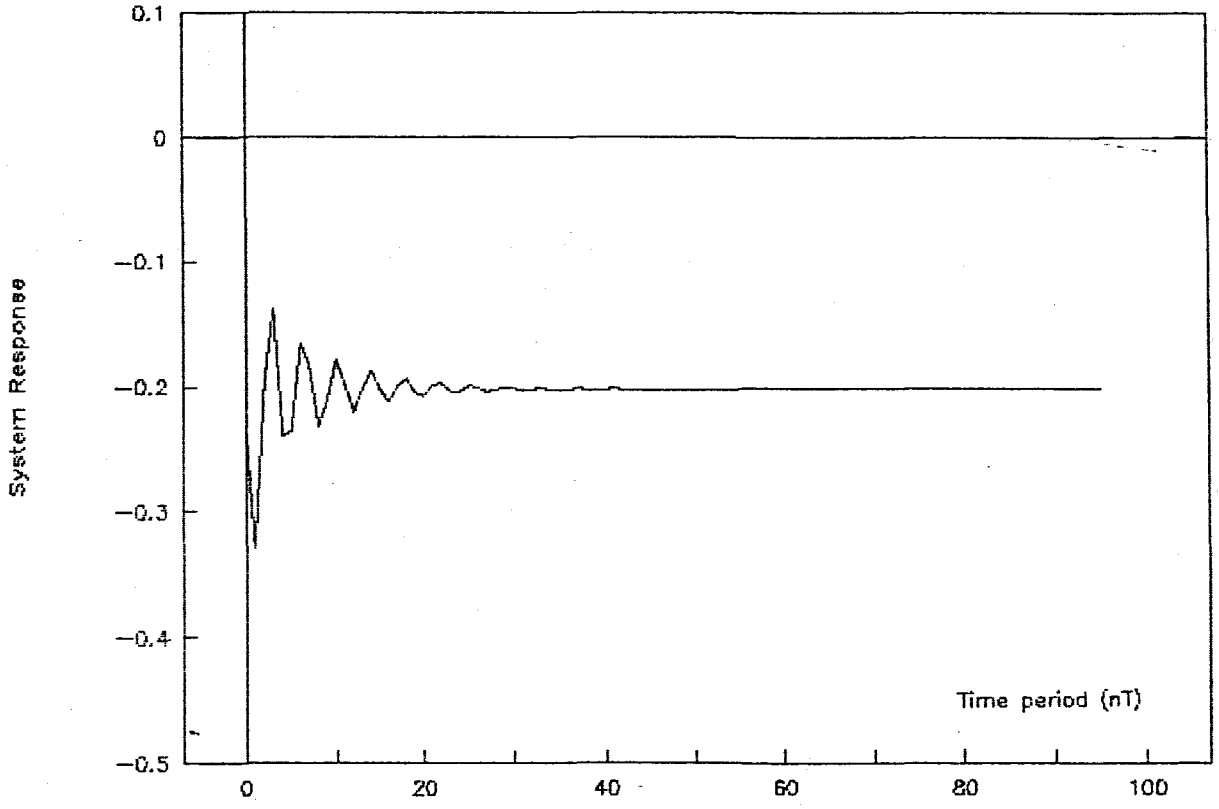


Figure 5.26 Step response of ARMA AKAIKE transfer function for the test case 2

Table 5.12. Steady state value of test case 2 for step input

	REAL	PADE'	ARMA MYWE	AKAIKE
x[∞]	-0.175228	-0.233081	0.4714	-0.19162

Table 5.13. Location of poles of test case 2 for simulated methods

	POLE 1	POLE 2	POLE 3
REAL	-0.0592+j0.9776	-0.0599-j0.9776	0.48029+j0.0000
PADE'	-0.0646-j0.9508	-0.0646+j0.9508	0.45486+j0.00000
ARMA MYWE	-0.0651-j1.0001	-0.0613-j1.0001	0.13335+j0.00000
AKAIKE	-0.6612+j0.9773	-0.6612-j0.9773	0.14228+j0.0000

Here, the step responses, poles and transfer function coefficients are give are given in tables (5.11), (5.12), and (5.13) for all simulated methods. From these table, we can say that Pade' approximation is not successful for the second case. This is due to the nature of the transfer function. But AKAIKE algorithm reaches some good results at very low orders. It can be said that Pade' approximation is not a good method for transfer functions which have poles close the unit circle.

Table 5.14. Simulation for various orders

I	J	PADE'	ARMA MYWE	ARMA AKAIKE
2	2	UNSTABLE	UNSTABLE	-0.185090566
3	3	-0.253085154	UNSTABLE	-0.201624638
5	5	-0.279122738	UNSTABLE	UNSTABLE
2	3	UNSTABLE	UNSTABLE	-0.400155232
3	4	UNSTABLE	UNSTABLE	-0.254319036
7	8	0.545654458	UNSTABLE	NOT APPL.
6	7	UNSTABLE	UNSTABLE	NOT APPL.
8	3	UNSTABLE	UNSTABLE	0.032225702
7	3	UNSTABLE	UNSTABLE	0.0091571770
10	5	UNSTABLE	UNSTABLE	UNSTABLE
4	4	UNSTABLE	-0.601602018	UNSTABLE
2	7	UNSTABLE	UNSTABLE	-0.385882972
3	8	UNSTABLE	UNSTABLE	UNSTABLE
4	7	UNSTABLE	UNSTABLE	UNSTABLE
2	8	UNSTABLE	UNSTABLE	UNSTABLE

b. Stability of systems: In order to analyze the stability of obtained system transfer function for the mentioned algorithms, we considered the table (5.12) and the location of poles on the z domain. From the table (5.12) it can be said that ARMA methodologies have more stable cases than PADE' and most of the times PADE' produces stable cases for only very low orders. On the higher orders of PADE' methodology easily produce unstable system responses, that can be understand by the location of poles.

To see the how the system response can be effected from the finite word length the system response for the different data representation bit lengths were investigated.

The figure (5.27) shows the logarithm of differences between extended and double, real and single data representation types on the system response for different methodologies

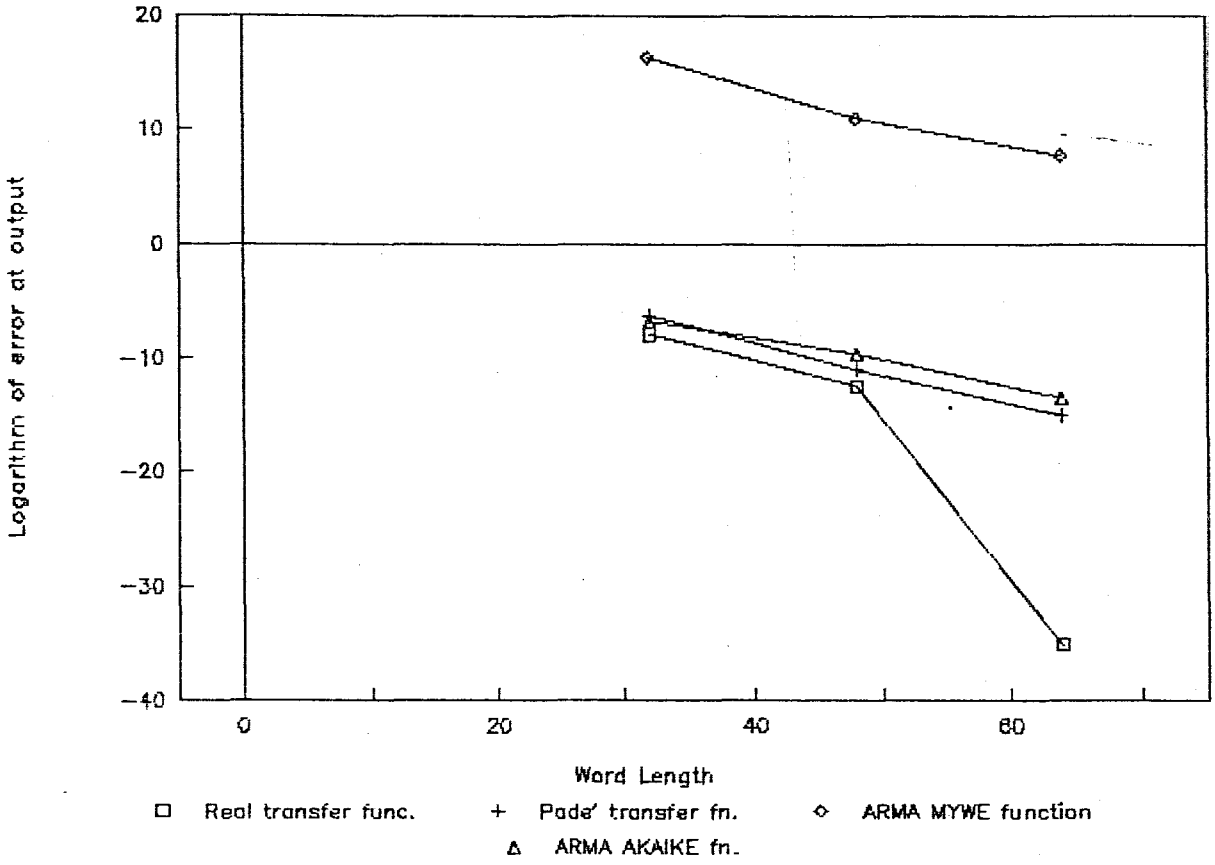


Figure 5.27. Logarithm differences of calculated output errors for the data representation types

It is obvious that if we use single data representation system response is effected more. But on the other hand Pade' approximation is less effected for such data representation changes.

c. Coefficient values : The coefficients differences from the real transfer function is given in Table (5.15) and in Figure (5.28) for test case 2.

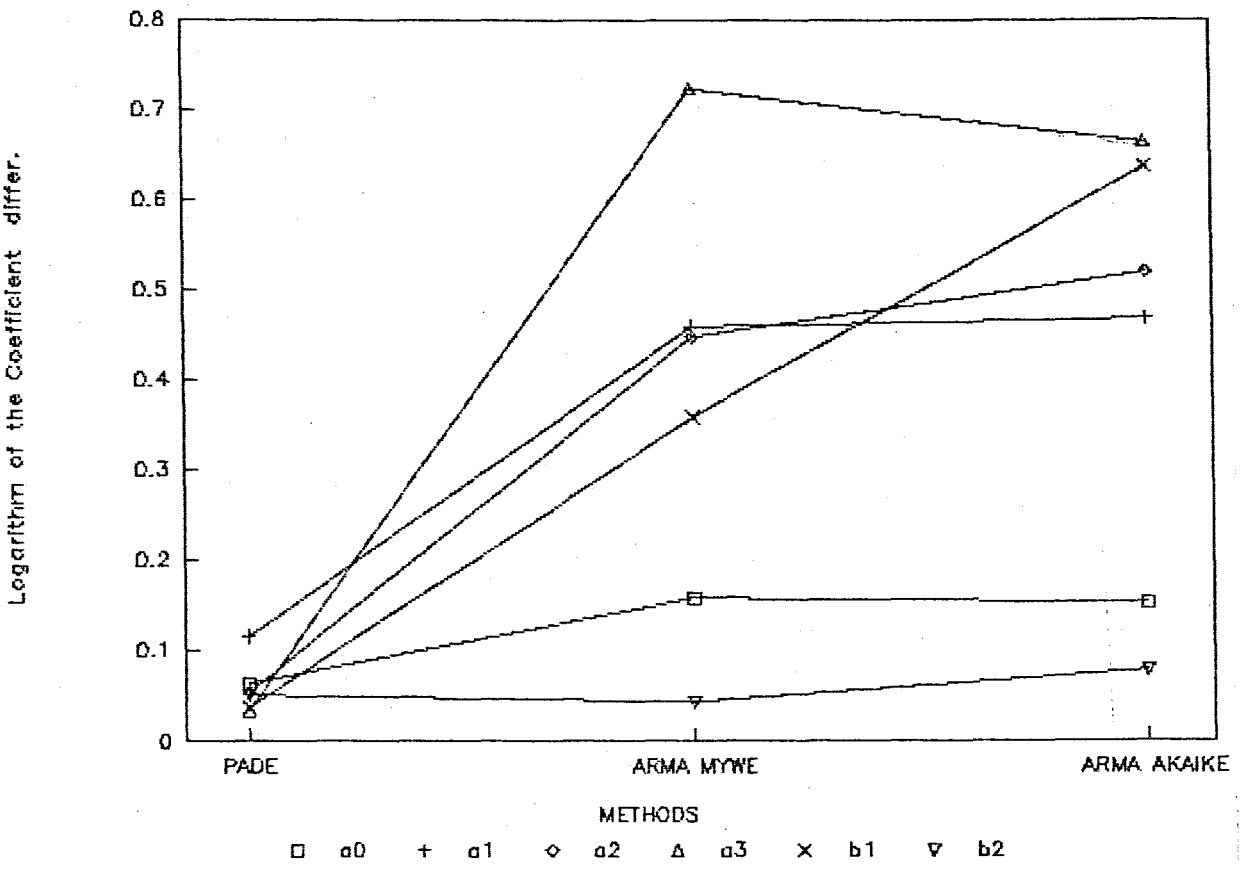


Figure 5.28 Coefficients error of simulated methods according to the extended data representation type

Table 5.15 Coefficients differences between real and simulated results

	REAL	PADE'	ARMA MYWE	AKAIKE
a0	0	0.06318	0.15873	0.15313
a1	0	0.11544	0.45940	0.46851
a2	0	0.05335	0.44721	0.51973
a3	0	0.03295	0.72362	0.66455
b0	0	0	0	0
b1	0	0.36298	0.35871	0.63632
b2	0	0.05054	0.04307	0.07841
b3	0	0.04644	0.33159	0.59602

In a similar manner, for the second case Pade' has better coefficient values approximation than the others for the order (3,3). So, Pade' approximation method simulates the real transfer function better.

d. Sensitivity to error:

1. Input quantization error: Again this error is modelled by simply adding a zero mean white noise sequence of variance $q^2/12$ to the original unquantized discrete-time signal.

2. Effects of coefficient inaccuracy: Here a table is given to show the effect of finite word length by using single, real and double data types as it is explained in the first case.

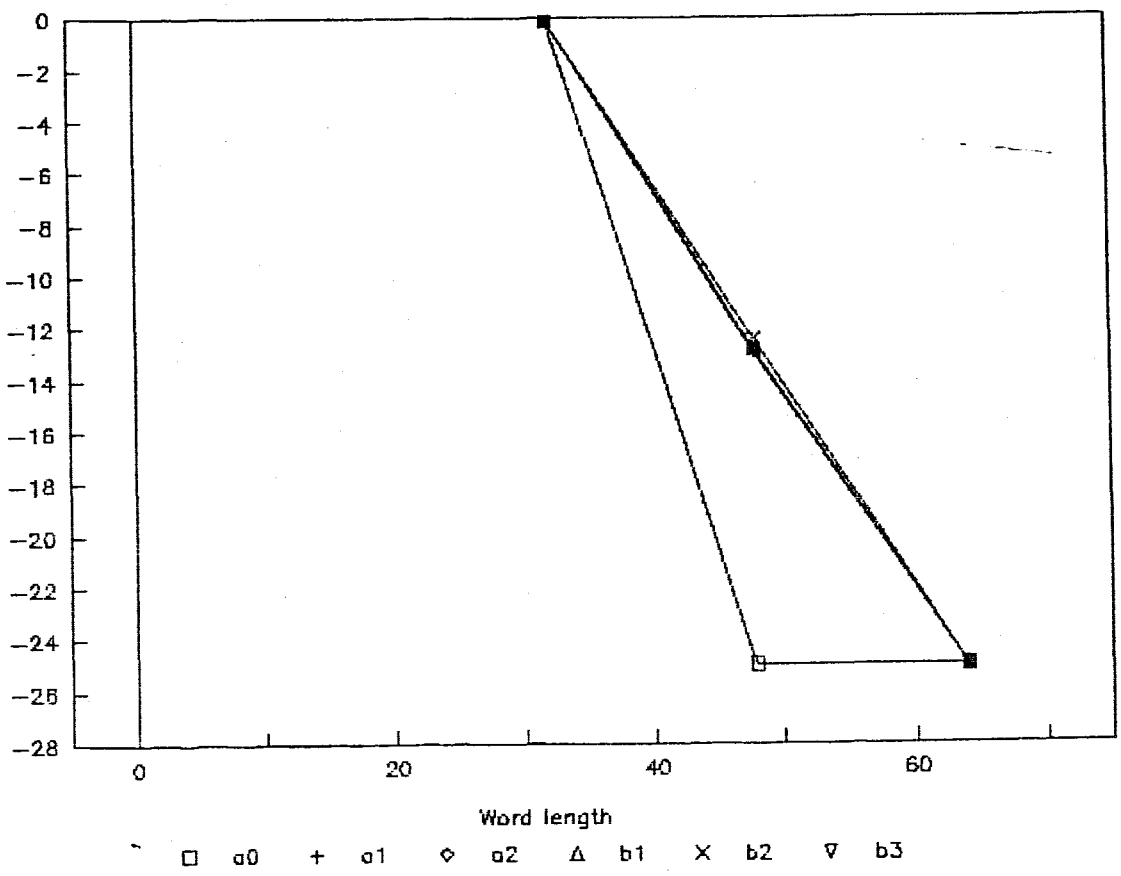


Figure 5.29 Real transfer function : Logarithm of coefficient errors according to the change of word length

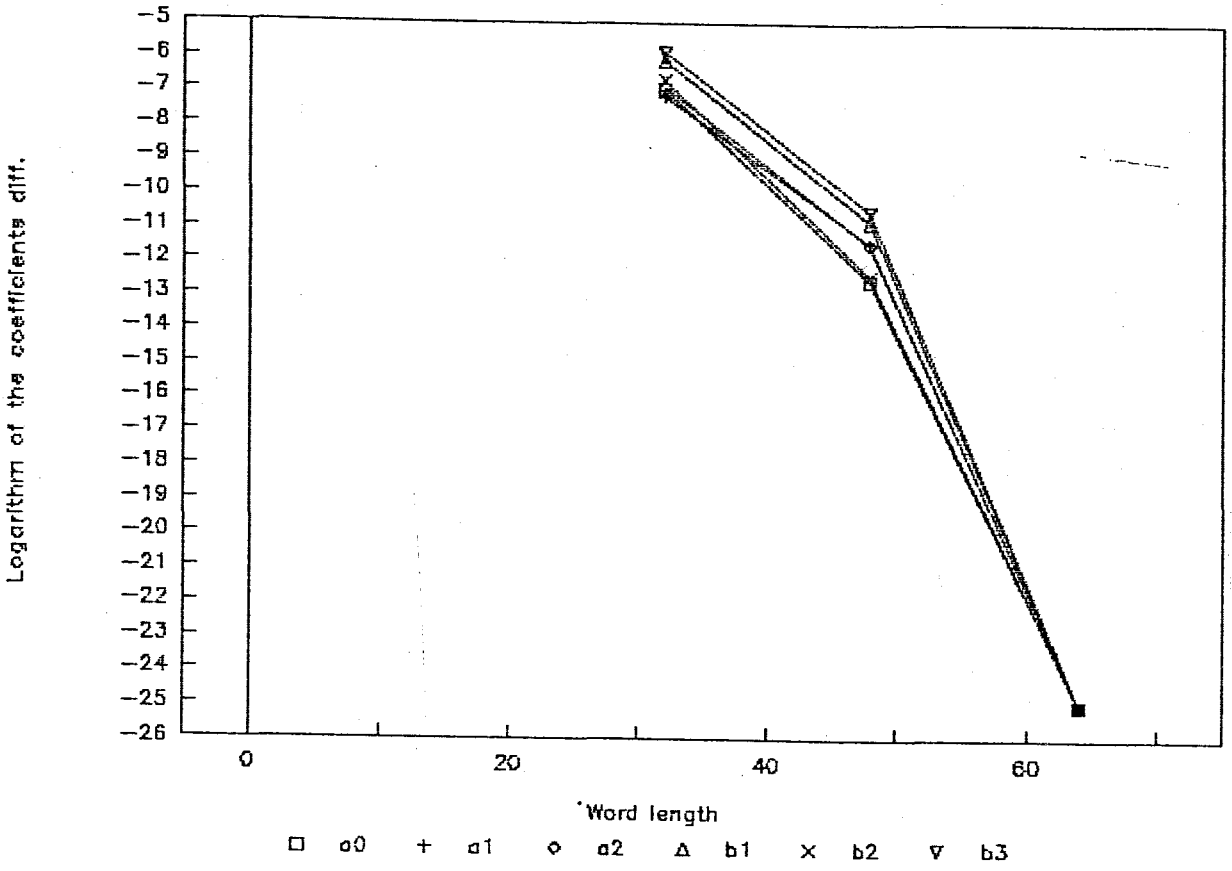


Figure 5.30. Pade' transfer function : Logarithm of coefficient errors according to the change of word length

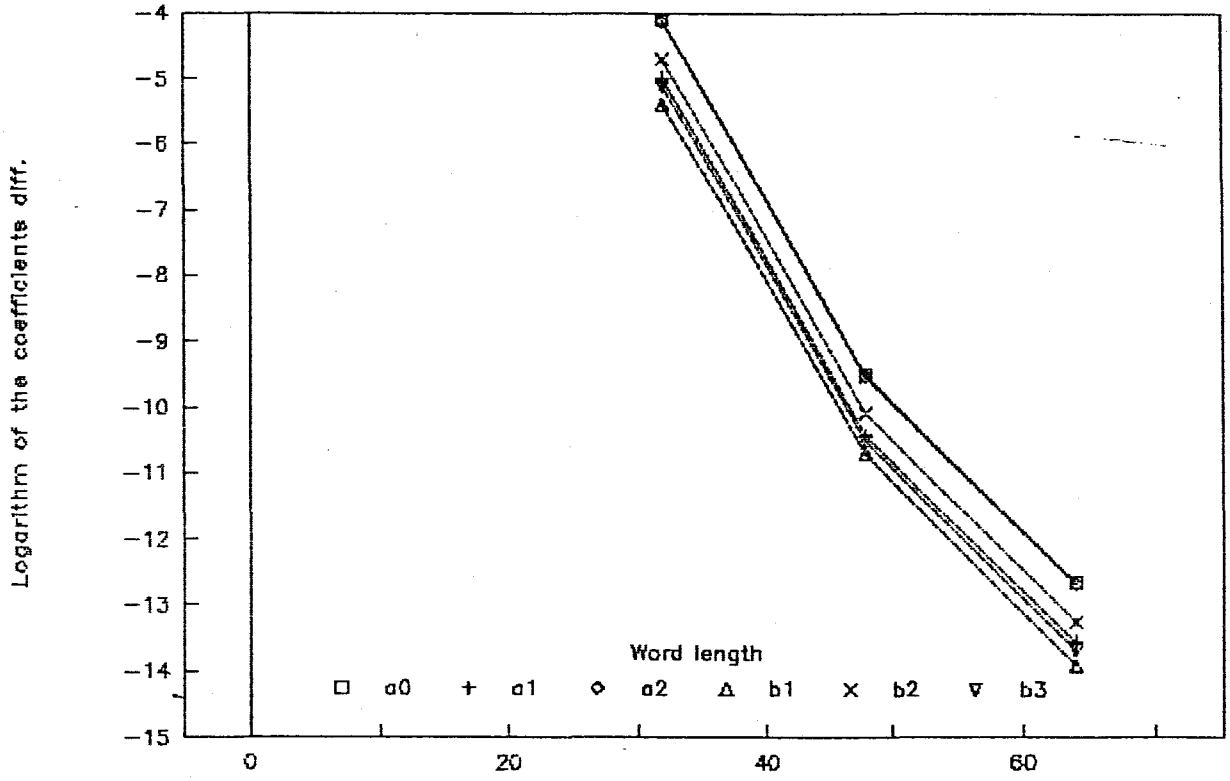


Figure 5.31. ARMA MYWE transfer function: Logarithm of coefficient errors according to the change of word length

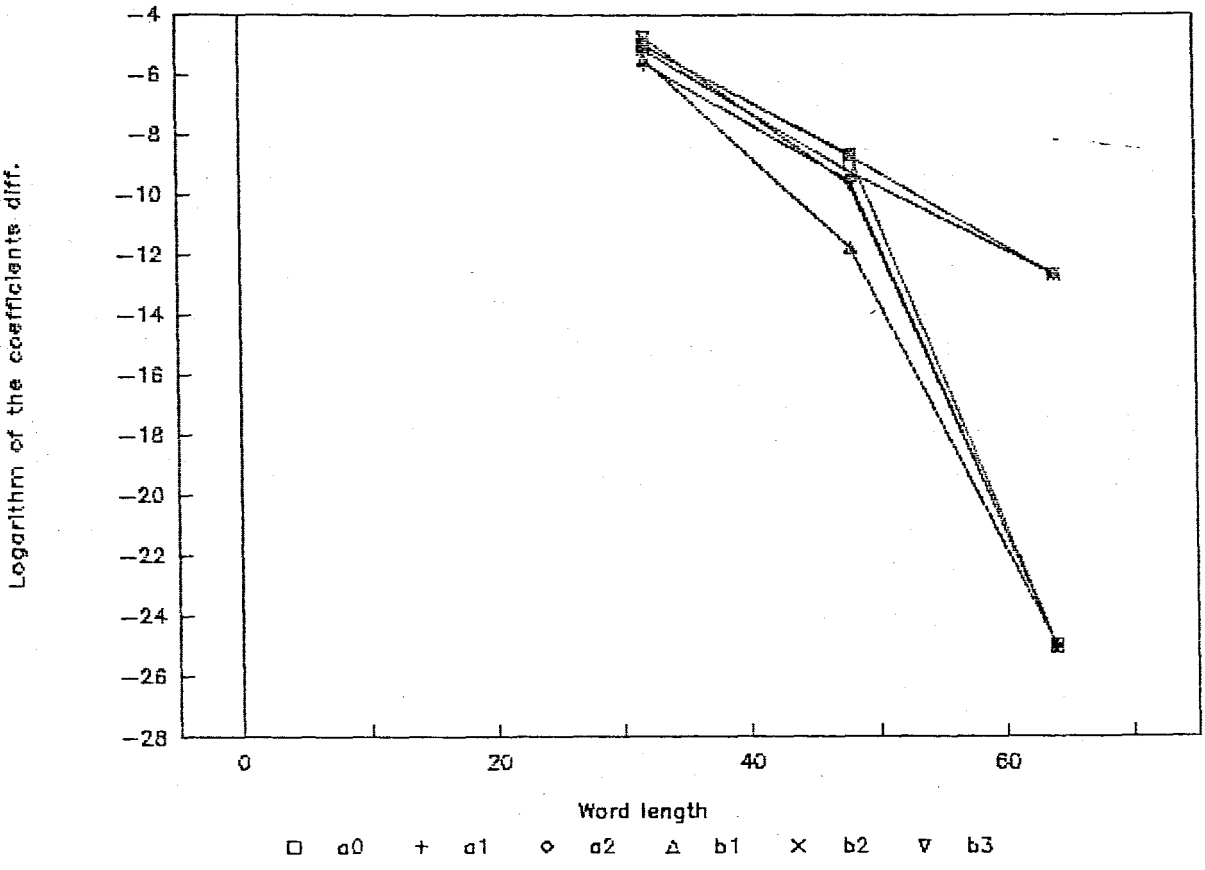


Figure 5.32. ARMA AKAIKE transfer function : Logarithm of coefficient errors according to the change of word length

The figures shows the same results with the results of test case one. It is obvious that Pade' is producing the results is a lower order transfer function than ARMA so Pade' is effected less relatively to the ARMA if the orders level is considered.

3. Round-off error accumulation: Here we will calculate the propagated error using the formula given in test case one with the equation number (5.2).

Table 5.16. Calculated errors for approximation types in State-space direct form (all numbers should be multiplied by q^2)

I	J	PADE'	ARMA MYWE	ARMA AKAIKE
2	2	-	-	-0.26419273
3	3	0.289673616	-	0.282533023
5	5	0.1821913131	-	-
2	3	-	0.478756531	0.246597057
3	4	-	-	0.330128270
8	3	-	-	0.283400141
7	3	-	-	0.320590893
4	4	-	0.244017160	-
2	7	-	-	0.490973875

The round-off errors are calculated only for stable transfer functions obtained. Secondly the error sensitivity for different state-space the state-space representations is given below only for parallel representation type.

Table 5.17. Output noise power of test case 2 for different state space representation types (all numbers should be multiplied with q^2)

Repr. type	REAL	PADE'	ARMA MYWE	ARMA AKAIKE
Direct Form	0.09641	0.120697	0.100007	0.117722
Parallel form	0.113	0.1152	0.13312	0.1598

Here, with these results we can same thing that Pade' approximation technique is less sensitive to the round-off errors according to the ARMA and AKAIKE method for same order levels.

6. CONCLUSION

This study had two major objectives: Firstly, to make a literature survey on a new approach of Pade' approximation in system modelling and secondly, to analyze the introduced approach from the point of view of sensitivity to input quantization error, coefficient quantization error and round-off accumulation error.

Several papers [29], [30], [32], [41] report the following drawback for Pade' approximation : "Pade' approximation can produce stable (unstable) system response even if the real system is unstable (stable)". Shamash [32] is especially concerned with the solution of the unstability problem and suggests a different way to provide stability. This approach of Shamash is based on fitting an ARMA model from data samples and then reducing it by Pade'. On the other hand, Biyiksiz's algorithm obtains the Pade' type reduced order transfer function directly from the data samples[30]. Obviously Biyiksiz's approach eliminates some of the steps, but this approach does not guarantee stability of obtained transfer function. Shamash's approach finds the dominant poles of higher order transfer function (generally ARMA type), then by applying Koenig's theorem and its generalization expanding it to power series, lastly fits a Pade' approximation to this power series. By this way it provides a stable (unstable) simulation.

In this study, in order to find out the advantages and disadvantages of Pade' approximation we mainly compared it with ARMA approximation techniques. The results of two test cases with ARMA is given in Chapter 5. Although we worked on several sample functions, here only two of them are documented, because they explain the behaviour of the approximation clearly. In general, it was observed that a stability problem of obtained transfer function always exists. It appears mostly

in Pade' cases. On the ARMA side, to obtain better approximation not only MYWE but also, AKAIKE method is considered.When the right order is selected ARMA approaches produced proper results, but for Pade' we could not find any research on model order selection. The lack of model order selection rules caused to choose the orders by examining all the orders. It also created some difficulties in the analsis of obtained results. From the results it is seen that when the right order is chosen, Pade' model reaches the real transfer function at a lower order than the others.

When the stability problem is eliminated, second concern was the sensitivity to error. This was investigated respectively at the orders that were assumed as the right order for Pade' and ARMA. Three types of error were examined: Input signal quantization, coefficients quantization, and arithmetic round-off error accumulation. In the examination of the results obtained from both methods, it is seen that Pade' approximation method is less sensitive to the finite word length effects. ARMA methods were relatively more sensitive to error compared to Pade'. Comparison is done on system response and coefficients values produced by Pade'. It is shown in Chapter 5 that coefficients produced by Pade' approximation is less effected by the change of the word length. The round-off error value for direct type representation was greater than ARMA transfer function direct representation error. It is seen that Pade' has higher sensitivity to round-off error accumulation for direct type state-space realization. When the parallel type of state-space approach is studied, we observed that Pade' method transfer function has also less error accumulation that of ARMA. This is especially true when the obtained transfer function response has the best approximation to the original function.

As a conclusion, Pade' approximation has some advantages and disadvantages which can be summarized as follows:

The advantages of Pade' approximation:

a. It is easy to use.
b. Computationally, it is simpler than other similar methods.
c. The Pade' model is of lower order so the multiplicative effects of coefficient quantization is minimized.
d. The Pade' model is relatively less sensitive to quantization noise caused by the effect in (c).
e. Normally globally effective Pade' models of second order may be achieved which lend themselves as ideal candidates for direct form state space realizations.
f. Pade' model greater than second order are of lower order than ARMA models achieved for the same system under consideration; therefore, even when the direct form is not feasible, the Pade' model will realize cascade or parallel forms with fewer second order selections, thus affording less complexity.

The disadvantages of Pade' approximation:

a. Because of the lack of model order selection rules, user should decide on the order on their own experience or examining any other method.
b. As a result of (a) it can become unstable (stable) although the original was system stable (unstable)
c. Especially, for the systems which has poles very close to the unit circle, special care should be payed. The reduction of poles can cause unstability, because these systems have large magnitude poles with negative real parts.

When Pade' approximation is used both of the advantages and disadvantages should be taken into account before applying it to the system at hand.

This work can be extended as follows:

1. A research should be done on order selection method of Pade' for Biyiksiz approach. Some criteria should be developed to choose the right order.

2. The study on state-space modelling technique is done only for parallel approach because of the lack of time. So, we suggest that a detailed research should be done on various state-space representation types to investigate error sensitivity.

APPENDIX A

In this appendix, all the programs written along the development of this study is given. But giving all these programs documented causes lots of pages printed, so all the programs is given with a 360 KB. 5 1/4' diskette in IBM PC compatiable format. Readers can use the program sources and compile them by using Turbo Pascal 5.0. A READ.ME file is placed on the diskette. This file explaines each source file and its function in study. To explaine the requirements of users two files named HARDWARE.TXT and SOFTWARE.TXT is given.

REFERENCES

[1] Spiegel M.R. Laplace donusumleri, Egitim Yayinlari, Istanbul, 1965.

[2] Dorf C. R., Modern Control Systems, Addison-Wesley, Massachusetts USA.,1974.

[3] Cadzow A.J., Hung F. L., Signals, systems, and transforms, Prentice Hall, New Jersey , 1985.

[4] Blackman R.B., Tukey J.W.,The Measurement Of Power Spectra From The Point Of Commucations Engineering, Dower Publications, New York,1958

[5] Cooley J.W., Tukey J.W., "An Algorithm For The Machine Calculation Of Complex Fourier Series", Mathematics Of Computation, Vol. 19,pp. 297-301,1965

[6] Steven M. K., Modern Spectral Estimation: Theory and applications, Prentice Hall, New Jersey, 1988.

[7] Harris F.J., "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform", Proceeding IEEE, Vol. 66, pp. 51-83, 1978

[8] Welch P.D., "The Use of Fast Fourier Transform for Estimation of Power Spectra: A Method Based on Time Averaging over Short, Modified Periodograms", IEEE Transactions on Audio Electroacoustics, Vol. 15, pp. 70-73, 1967

[9] Marple S.L., Digital Spectral Analysis with Applications, Prentice Hall, New Jersey, 1987.

[10] Marple S.L., "A tutomal Overview of Modern Spectral Estimation", Preceedings of IEEE, Vol. 83, pp. 2152-2157, 1989

[11] Celasun I., "Comparative Evaluation Of Different

Techniques For Robust Spectral Estimation", M.S. Thessis, Bosphorus University, Istanbul-Turkey, 1990

[12] Capon J.,"High Resolution Frequency-Wavenumber Spectrum Analysis", Proceeding IEEE, Vol. 57, pp. 1408-1418,1969

[13] Kay M.S., Member, IEEE, Marple L.S., Jr., "Spectrum Analysis-A Modern Perspective", Proceeding Of The IEEE, Vol. 69, pp. 1380-1418,1981

[14] Ljung L., System Identification,Prentice Hall, New Jersey, 1987.

[15] Akaike H.,"A New Look At The Statistical Model Identification", IEEE Transactions On Automatic Control,Vol. AC19, pp.716-723, 1974

[16] Akaike H.,"Maximum Likelyhood Identification Of Gaussian Autoregressive Moving Average Models", Biometrika, Vol. 60,pp. 255-265,1973

[17] Astrom K.J., Soderstrom T.,"Uniqueness Of The Maximum Likelihood Estimates Of The Parameters Of An ARMA Model", IEEE Transactions On Automatic Control,Vol. AC19,pp. 769-773,1974

[18] Gersch W., "Estimation of the Autoregressive Parameters of a Mixed Autoregressive Moving-Average Time Series", IEEE Transactions Automatic Control, Vol. AC15, pp. 583-588, 1970

[19] Sakai H.,Tokumaru H., "Statistical Analysis Of A Spectral Estimator For ARMA Processes", IEEE Transactions On Automatic Control,Vol. AC25,pp. 122-124, 1980

[20] Cadzow J.," Spectral Estimation: An Overdeterminet Rational Model Equation Aproach", Proceeding IEEE,Vol. 70,pp. 907-939, 1982

[21] Chow J.C.,"On The Estimation Of The Order f A Moving-Avarage Process", IEEE Transactions On Automatic Control, Vol.AC17,pp. 386-387, 1972

[22] Bosley M.J., Lees F.P., "Methods for the Reduction of High Order State Variable Models to Simplify Transfer Function Models", Conf. Digital Representation of Continuous Systems, Preprints, Budapest, 1971

[23] Chen C.H., Shieh L.S., "A Novel Approach to Linear Model Simplification", JACC Rec., pp. 454-461, 1968

[24] Shamash Y.,"Model Reduction Using The Routh Stability Criterion And The Pade' Approximation Technique",Int. J. Control,Vol. 21,pp.475-484,1975

[25] Shamash Y., "Linear System Reduction Using Pade' Approximation To Allow Retention Of Dominant Modes", International Journal Of Control, Vol. 21, pp. 257-272, 1975

[26] Baker G.A., Essentials of Pade' Approximants, Academic Press, 1975.

[27] Baker A. G., Morris P. G., Carruthers P. A., Encyclopedia of Mathematics and its Applications Part I:Pade' Approximants, Addison-Wesley, Massachusetts USA.,1980.

[28] Shamash Y.,IEEE Conference On Computer Aided Control System Designe, Cambridge University, pp. 220, 1973

[29] Shamash Y., "Application Of Koenig's Theorem To Model Reduction", IEEE Transactions On Circuits And Systems, Vol. 22, pp. 702-704, 1975

[30] Biyiksiz S.M., "Rational System Modeling And Identification", IEEE 33rd Midwest symposium On Circuits

And Systems, Calgary, Alberta, Canada, 1990

[31] Shamash Y.,"Viability Of Methods For Generating Stable Reduced Order Models", IEEE Transactions On Automatic Control,Vol. AC26, pp. 1285-1286, 1981

[32] Shamash Y.,"Stable Reduced-Order Models Using Pade'-Type Approximations", IEEE Transactions On Automatic Control, Vol. 20, pp. 615-616,1976

[33] Hostetter G.H., Digital Control System Design, Holt,Rinehart and Winston Inc., New York USA., 1988.

[34] Tretter S.A.,Introduction To Discrete-Time Signal Processing, John Wiley & Sons,USA.,1976

[35] Goodwin G.C.,Sin K.S.,Adaptive Fittering Prediction & Control,Prentice-Hall Inc.,New Jersey,1984

[36] Liu B., Member, IEEE, "Effect Of Finite Word Length On The Accuracy Of Digital Filters-A Review", IEEE Transactions On Circuit Theory, Vol. CT-18, pp. 670-677, 1971

[37] Mitra K.S., Fellow, IEEE, Hirano K., Member, IEEE, Sakaguchi H., "A Simple Method Of Computing The Input Quantization And Multiplication Roundoff Errors In A Digital Filter", IEEE Transactions On Acustics,Speech,Signal Processing, Vol. 22, pp. 326-329, 1974

[38] Charalambous C., Best M.J.,"Optimization Of Recursive Digital Filters With Finite Word Lengths", IEEE Transactions On Acoustics,Speech And Signal Processing, Vol. 22,pp. 424-431,1974

[39] Hwang S.Y.,Member,IEEE,"Roundoff Noise In State-Space Digital Filtering: A General Analysis", IEEE Transactions On Acoustics,Speech And Signal Processing,Vol. 24, pp.

256-262, 1976

[40] Lam J., "Convergence Of A Class Of Pade' Approximations For Delay Systems", International Journal Of Control, Vol. 52, pp. 988-1009, 1990

[41] Claasen T.A.C.M.,Mecklenbrauker,Peek J.B.H.,"Effects Of Quantization And Overflow In Recursive Digital Filters",IEEE Transactions On Acoustics,Speech And Signal Processing,Vol. 24, pp. 517-529,1976

[42] Thomas J.B., Matematik, Ayrim Yayinlari, Ankara, 1986.

[43] Kan P.F.E., Member, IEEE, Aggarwal J.K., "Error Analysis Of Digital Filter Employing Floating-Point Arithmetic", IEEE Transactions On Circuit Theory, Vol. CT-18, pp. 678-686, 1971

[44] Box G.E.P., Jenkins G.M., Time Series Analysis, Holden-Day Inc., San Fransisko USA., 1970.

[45] San N., Kompleks Fonksiyonlar Teorisi, Ege Universitesi Basimevi, Izmir, 1979.

[46] Smirnov V.I., Lebedev N.A., Functions of a Complex Variable, Iliffe Books Ltd., London, 1968.