Computationally Fast Algorithms for ARMA Spectral Estimation

by

Koji Ogino

Dissertation submitted to the Graduate Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY
in
Electrical Engineering

APPROVED:

James A. Cadzow, Chairman

Hugh F. Van Landingham

Ioannis M. Besieris

C. Richard Johnson, Jr.

Lee W. Johnson

June, 1981

Blacksburg, Virginia

ACKNOWLEDGMENTS

The author expresses sincere appreciation to Dr. Hugh F. Van Landingham, Dr. Ioannis M. Besieris, Dr. C. Richard Johnson, Jr. and Dr. Lee W. Johnson for their thoughtful advice in preparation of this thesis. Special recognition is due to his committee chairman, Dr. James A. Cadzow for his motivation and patient guidance during the course of this research endeavor.

The financial support from the Electrical Engineering Department of Virginia Polytechnic Institute and State University is thankfully acknowledged.

He would also like to thank for her excellent typing assistance.

A special note of appreciation is due to my understanding wife, , for her patience and encouraging support.

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

The mathematical development of digital signal analysis has been an area of primary concern since the digital computers development over two decades ago. The analysis of the frequency characteristic of a signal is of particular interest in the field known as "time series analysis." Time series analysis encompasses such areas as statistics, economics, and communications. Most of the work in time series analysis has been carried out by statisticians. More recently, however, many advancements in the analysis of time series have been made in the field of signal processing based on power spectral estimation concepts and time domain analysis.

The need for power spectral estimates arises in a variety of contexts, including the measurement of noise spectra for the design of optimal linear filters, the detection of narrow-band signals in wide-band noise, and the estimation of parameters of a linear system by using a noisy excitation.

Current methods of spectral estimation can be broadly classified into two categories. One is the classical approach which includes the periodgram method, autocorrelation methods and its variants (Bartlett, 1953; Blackman and Tukey, 1958; Grenander and Rosenblatt, 1957; Jenkins and Watts, 1968; Koopmans, 1974). The second is

modern power spectral density estimations based on parameters modeling. This includes the maximum entropy method (Burg, 1967), one-step linear prediction (Parzen, 1969), and spectral estimation using ARMA model (Tretter and Steiglitz, 1967; Gutowski, Robinson and Treitel, 1978). In practical signal processing applications, classical approachs have been incorporated by many researchers and users. This is because classical methods are fairly easy to implement and can be computed efficiently by using the fast Fourier transform (Cooley and Tukey, 1965). However, the spectral estimates obtained by classical methods can provide unsatisfactory results when the data length is short. For example, variance of estimates is large and the resolution capability of noise embedded sinusoids is poor in such cases. To overcome these difficulties, the modern spectral estimation methods were developed. These methods provide better spectral performance than classical methods. For example, one of the widely used modern spectral methods referred to as the Maximum Entropy method (Burg, 1967) possesses better resolution capability than the classical periodgram approaches for short data lengths. The Maximum Entropy method is classified as an autoregressive (AR) model. The AR model is also known as an all-pole model which uses only a denominator polynomial of a rational model. In recognition of this constraint, a more general form, the autoregressive and moving average (ARMA) model which has numerator polynomials as well as denominator polynomial has been proposed. A variety of procedures has been developed for generating ARMA models. One of these methods is the so-called 'high

performance' ARMA method which was recently developed by Cadzow (1979). The 'high performance' ARMA method has provided excellent spectral estimation performance when compared with the Maximum Entropy and its variants. However, its computational efficiency is relatively burdensome.

Recently, attention has been directed towards developing 'fast' spectral estimation algorithms. These include the generalized Levinson's algorithm. As an example, it is possible to use this approach for estimating the autoregressive coefficients of a p-th order AR model with the number of required additions and multiplications being on the order of p^2 (i.e., $0(p^2)$). Recently, Morf developed the doubling algorithm which reduced the required computations to 0(p log p) by using the divide and conquer approach (Morf, 1980). More recently, recursive methods which have an ability to compute necessary parameters at the arrival of each new data point has been proposed (Lee and Morf, 1980). This algorithm does not require any matrix formulation and the computational requirements can be reduced to 0(p) to update the AR model parameters with each new data sample.

In this thesis, the development of fast algorithms for the high performance spectral estimation method is treated. To begin our development, in Chapter 2, the mathematical definition of power spectral density function is stated and two classical methods referred to as the periodgram and the autocorrelation method are discussed. The common weakness of these classical techniques are examined. In Chapter 3, a standard procedure of modern spectral estimation,

namely, the rational function model is discussed. Modern spectral linear estimators can be classified into three types of models: (i) AR (Autoregressive) model, (ii) MA (Moving Average) model, and (iii) ARMA (Autoregressive and Moving Average) model. It is widely known that the ARMA model is a desired form from a parameter parsimony viewpoint. In Chapter 4, the 'high performance' ARMA spectral estimation is described. Although this method gives excellent spectral performance, the computational requirements are relatively burdensome. To achieve a higher degree of computational efficiency, fast algorithms are developed in Chapter 5 and data modification methods are introduced. In Chapter 6, a recursive algorithm which requires $O(p)$ computations at the arrival of each new data sample is developed. Development of this algorithm is predicated on various projection operator decompositions.

Chapter 2

CONVENTIONAL SPECTRAL ESTIMATIONS

2.1 Introduction

The spectral density function is mathematically defined in Section 2.2. Conventional spectral estimation techniques have been developed based on the Fourier transform relationship between the power spectral density function and the autocorrelation sequence (Bartlett, 1953; Blackman and Tukey, 1958; Grenander and Rosenblatt, 1957; Jenkins and Watts, 1968; Koopmanns, 1974). For example Blackman and Tukey developed an autocorrelation method (Blackman and Tukey, 1958) which includes following steps:

(i) Estimate the autocorrelation sequence from the observed data;

(ii) Window the autocorrelation estimate;

(iii) Fourier transform of the windowed data record.

While various procedures are used in step (i) to estimate the autocorrelation function, the objective is usually to obtain a minimum bias and minimum variance estimate of the true autocorrelation sequence. In step (ii), windowing is used to reduce the bias and the variance of the power spectral estimate. However, the windowing process decreases the resolution of the power spectral estimate. This autocorrelation method demonstrates typical weaknesses of conventional spectral estimation approaches. Spectral estimation

performance had not been improved until the development of modern spectral estimation techniques.

2.2 Definition of Power Spectral Density

Let us consider a discrete time random sequence $\{x(n)\}$ with autocorrelation sequence $\{r_x(m)\}$ defined by

$$r_x(m) = E\left[x(n+m)\ \overset{*}{x}(n)\right] \tag{2.2.1}$$

where E and * denote the expected value and complex conjugate operation, respectively. We will denote the z-transform of $\{r_x(m)\}$ by

$$S_x(z) = \sum_{m=-\infty}^{\infty} r_x(m)\ z^{-m} \tag{2.2.2}$$

The associated power spectral density is then defined to be

$$S_x(\omega) = S_x(z)\Big|_{z=e^{j\omega}} = \sum_{m=-\infty}^{\infty} r_x(m)\ e^{-j\omega m} \tag{2.2.3}$$

Applying the inverse z-transform to eq. (2.2.2), we have

$$r_x(m) = \frac{1}{2\pi j} \oint_C S_x(z)\ z^{-m}\ \frac{dz}{z} \tag{2.2.4}$$

where C is a simple closed contour contained within the region of convergence for $S_x(z)$. If C is chosen to be the unit circle, by making the change of variable $z=e^{j\omega}$, we derive the discrete inverse Fourier transform relationship

$$r_x(m) = \frac{1}{2\pi} \int_{-\pi}^{\pi} S_x(\omega)\ e^{j\omega m}\ d\omega \qquad (2.2.5)$$

The variance of the random time series $\{x(n)\}$ is equal to $r_x(0)$ and can be expressed by

$$E\{|x(n)|^2\} = r_x(0) = \frac{1}{2\pi} \int_{-\pi}^{\pi} S_x(\omega)\ d\omega \qquad (2.2.6)$$

It follows that the average power in the incremental frequency band $\omega_0 \leq \omega \leq \omega_0 + d\omega$ (Tretter, 1976) is found to be

$$P_x(\omega_0) = S_x(\omega_0)\ \frac{d\omega}{2\pi} \qquad (2.2.7)$$

As shown in eq. (2.2.6), the time series variance is equal to the total power of the signal which is a scalar multiple of the area under the curve $S_x(\omega)$. Observing the relation between expressions (2.2.6) and (2.2.7), one can see that the integral over the incremental frequency band is proportional to the total power of the signal in that band. For these reasons the function $S_x(\omega)$ is called the power spectral density.

The frequency response of a linear shift-invariant system and the frequency domain representation of a discrete-time signal are essential concepts in digital signal processing. In this section we describe another interpretation of the power spectral density function using the theory of linear discrete-time systems for the case when the input is a random time series (Oppenheim and Schafer, 1975). Consider a stable linear shift-invariant system with unit-sample response $h(n)$. Let $\varepsilon(n)$ be a real input sequence that is a

sample sequence of a wide-sense stationary discrete-time random process. Then the output of the linear system is a sample function of a random process related to the input process by the linear transformation

$$x(n) = \sum_{k=-\infty}^{\infty} h(n - k)\ \varepsilon(k) \tag{2.2.8}$$

It can be shown that if the input is stationary, then so is the output. The input signal may be partially characterized by its mean and its autocorrelation function $r_{\varepsilon}(m)$, or we may also have additional information about first or higher order probability distributions. In characterizing the output random process $\{x(n)\}$, we desire similar information. For many applications, it is sufficient to characterize both the input and output in terms of simple averages, such as means, variances, and autocorrelations. Therefore, we shall derive input-output relationships between these quantities. Generally we consider zero mean processes and our analysis is restricted to the examination of the autocorrelation sequence. The autocorrelation function of the output process is readily shown to be given by

$$r_x(m) = \sum_{n=-\infty}^{\infty} r_{\varepsilon}(m - n) \sum_{k=-\infty}^{\infty} h(k)\ h^*(n + k) \tag{2.2.9}$$

To characterize the response of a linear time-invariant system to a discrete time input, we apply the z-transformation to expression (2.2.9) to yield

$$S_x(z) = H(z)\ \ H^*(z)\ S_{\varepsilon}(z) \tag{2.2.10}$$

where H(z) is the transfer function of the linear shift-invariant system. In terms of the power spectral density, (2.2.10) becomes

$$S_x(\omega) = |H(e^{j\omega})|^2 S_\varepsilon(\omega) \tag{2.2.11}$$

where the impulse response {h(k)} is taken to be a real sequence. If the input random process is a white noise with variance σ_ε^2, it follows that

$$S_x(\omega) = |H(e^{j\omega})|^2 \sigma_\varepsilon^2 \tag{2.2.12}$$

Relationship (2.2.12) is extensively used in analysis concerned with modern spectral estimation.

2.3 Discrete Fourier Transform Approach

As shown in Section 2.1, the power spectral density and autocorrelation functions are related by the discrete Fourier transform. Suppose that the sequence {x(n)} is a wide-sense stationary random time series and the complete knowledge of the associated autocorrelation $\{r_x(m)\}$ is given, the spectral density can be simply obtained by

$$S_x(\omega) = \sum_{m=-\infty}^{\infty} r_x(m)\, e^{-j\omega m} \tag{2.3.1}$$

In relevant signal processing applications, it is never feasible to measure an infinite number of autocorrelation sequence elements $\{r_x(m)\}$.

We will now begin to examine the problem of estimating $S_x(\omega)$ from a finite observation of the time series $\{x(n)\}$. This observation can be represented by a set of N contiguous samples

$$x(0),\ x(1),\ \ldots\ ,\ x(N-1) \tag{2.3.2}$$

About two decades ago, spectral estimates had been mostly accomplished by the periodgram and autocorrelation methods.

2.3.1 Periodgram Method

To include an additional degree of flexibility, suppose that the observed sequence is modified to form the auxiliary signal

$$f(n) = w(n)\ x(n) \qquad 0 \leq n \leq N-1 \tag{2.3.3}$$

where $w(n) = 0$ for $n < 0$ and $n \geq N$. The sequence $w(n)$ is frequently called a data window. The sample autocorrelation function for the modified observed sequence can be written as

$$r_f(n) = \frac{1}{N} \sum_{k=-\infty}^{\infty} f(k+n)\ f(k) \tag{2.3.4a}$$

$$= \frac{1}{N}\ f(n) * f(-n) \tag{2.3.4b}$$

where * denotes the operation of convolution. Denoting the z-transform of $r_f(n)$ and $f(n)$ by $R_f(z)$ and $F(z)$, respectively, the convolution and time reversal theorems yield the following relationship

$$R_f(z) = \frac{1}{N}\ F(z)\ F(z^{-1}) \tag{2.3.5}$$

Evaluating this expression at $z = e^{j\omega}$, we have

$$R_f(e^{j\omega}) = \frac{1}{N} \mid S_f(\omega) \mid^2 \tag{2.3.6}$$

The function $R_f(e^{j\omega})$ is known as the periodgram of $\{f(n)\}$. Two decades ago, the periodgram method became popular because $R_f(e^{j\omega})$ could be computed efficiently by using the fast Fourier transform (FFT see Cooley and Tukey, 1965).

2.3.2 Autocorrelation Method

When the true autocorrelation function $r_x(m)$ is unknown, it is desired to calculate an estimate of the autocorrelation function. The associated spectral estimate can then be obtained by taking a Fourier transform of this autocorrelation estimate (Blackman and Tuckey, 1959). Two common estimates

$$\hat{r}_x(m) = \frac{1}{N-m} \sum_{i=1}^{N-m} x(i)\, \overset{*}{x}(i+m) \qquad m = 0, \ldots, N-1 \tag{2.3.7}$$

and

$$\hat{\hat{r}}_x(m) = \frac{1}{N} \sum_{i=1}^{N-m} x(i)\, \overset{*}{x}(i+m) \qquad m = 0, \ldots, N-1 \tag{2.3.8}$$

are typically used for estimating the autocorrelation function. Applying the expected value operation on expression (2.3.7), we obtain

$$E\left[\hat{r}_x(m)\right] = \frac{1}{N-m} \sum_{i=1}^{N-m} E\left[x(i)\, \overset{*}{x}(i+m)\right] \tag{2.3.9a}$$

$$= r_x(m) \tag{2.3.9b}$$

The autocorrelation estimate $\hat{r}_x(m)$ is seen to be an unbiased estimate. On the other hand, one can similarly show that $\hat{\hat{r}}_x(m)$ is a biased estimate. Because $\hat{r}_x(m)$ is an unbiased estimate, it might be thought $\hat{r}_x(m)$ is the better estimate. For several reasons, however, $\hat{\hat{r}}_x(m)$ is sometimes preferable to $\hat{r}_x(m)$. First, the biased estimate does not violate a property of a valid autocorrelation functions, that is

$$r_x(0) \geq |r_x(m)| \tag{2.3.10}$$

while the unbiased estimate can violate this property. Second, the biased estimate produces a nonnegative spectral estimate, while the unbiased estimate may not (Burg, 1975). Third, the mean-square error for the biased estimate is less than that for the unbiased method (Jenkins and Watts, 1968). And finally, Parzen provides an argument in favor of the biased estimate by claiming that $\hat{\hat{r}}_x(m)$ has less variance than $\hat{r}_x(m)$ (Parzen, 1974).

Various procedures may also be used to estimate the autocorrelation function. The objective of these procedures is usually to obtain a minimum variance estimate of the true autocorrelation function. Similarly, the estimate of the autocorrelation function is windowed to reduce the bias and variance of the power spectral estimate, but increases its statistical stability. Various window functions have been used which are generally unrelated to the data or the random process being analyzed. Both the finite record length of the autocorrelation function estimate and the windowing process applied to the autocorrelation function decreases the resolution of the power

spectral estimate. An additional disadvantage of windowing is that unless one performs good windowing, excessive side lobes may be introduced in the power spectral estimate. Side lobes may be reduced by employing well designed windows but we then lose spectral resolution, particularly when the data record is short.

The autocorrelation method and its variants were developed to achieve better spectrum estimate performance in comparison to the periodgram method. As indicated above, however, the autocorrelation method has still several disadvantages. These disadvantages had not been overcome until the development of modern spectral estimation techniques.

Chapter 3

MODERN SPECTRAL ESTIMATION

3.1 Introduction

One of the most widely used models for spectral estimation is the rational model. The stochastic time series $\{x(n)\}$ is said to have a rational power spectrum if its power spectral density can be expressed in the form

$$S_x(\omega) = |H(e^{j\omega})|^2 \sigma^2 \tag{3.1.1}$$

where σ^2 is a positive constant and the characteristic rational function

$$H(e^{j\omega}) = \frac{B(e^{j\omega})}{A(e^{j\omega})} = \frac{b_0 + b_1 e^{-j\omega} + \ldots + b_q e^{-jq\omega}}{1 + a_1 e^{-j\omega} + \ldots + a_p e^{-jp\omega}} \tag{3.1.2}$$

is composed of the ratio of the polynomials $A(e^{j\omega})$ and $B(e^{j\omega})$ which may have real coefficients and the zeros of $A(e^{j\omega})$ are all contained within the unit circle. The rational power spectral density (3.1.1) is said to have order (p, q) and its zeroes and poles are seen to occur in reciprocal complex conjugate pairs.

A particularly convenient interpretation on how a stochastic time series with rational spectrum may arise follows directly from the characteristic rational function. This entails treating the characteristic rational function (3.1.2) as being the transfer function of a causal, time-invariant linear system. It then follows that this

system will be characterized by the recursive equation

$$x(n) = \sum_{i=0}^{q} b_i \, \varepsilon(n-i) - \sum_{i=1}^{p} a_i \, x(n-i) \qquad (3.1.3)$$

where the time series $\{\varepsilon(n)\}$ and $\{x(n)\}$ are taken to be the excitation and response signals, respectively. It has been shown in section 2.1 that when the excitation time series $\{\varepsilon(n)\}$ is a zero mean stationary white noise time series with variance σ^2, then the power spectral density of the response time series is given by relationship (3.1.1). Thus a stationary random time series with rational power spectral density can be interpreted as being the response of a stable causal, time-invariant linear system to a white noise excitation.

The general linear system (3.1.3) is commonly referred to as an autoregressive-moving average (ARMA) model in the spectral estimation literature. This ARMA model is said to be of order (p, q) and it gives rise to the rational spectrum (3.1.2) which possesses zeroes as well as poles. The ARMA model is the most general of rational spectrum models possible and its a_k and b_k coefficients uniquely characterize the spectrum.

In the spectral estimation literature, most of activity has been directed towards the special class of ARMA models known as auto-regressive (AR) models. An AR model is one in which the numerator polynomial $B(e^{j\omega})$ is equal to the constant b_0. As such, the AR model is also referred to as an all-pole model since its transfer function is specified by

$$H(e^{j\omega}) = \frac{b_0}{A(e^{j\omega})} \qquad (3.1.4)$$

This all-pole model is the one most often used in spectral estimation.

Another subclass of rational spectrum models which has received attention is the so-called moving average (MA) model as characterized by $A(e^{j\omega}) = 1$. The transfer function of a MA model is given by $B(e^{j\omega})$ and it is therefore also referred to as an all-zero model.

In summary, Table 3.1 shows the rational spectrum associated with each of these models.

3.2 Moving Average Model

Many conventional methods of spectral estimation are classified as MA models. For example, the periodgram and correlation methods which have been discussed in Section 2.3 can be described in terms of a MA model. Generally, little attention has been focused on MA models. Welch has introduced (Welch, 1967), however, a MA model technique which is particularly applicable to the direct computation of a power spectrum estimate that uses the FFT. In this technique, the data record is first sectioned into K = N/M segments of M samples each as defined by

$$x^{(i)}(n) = x(n + iM - M) \qquad 0 \leq n \leq M-1,\ 1 \leq i \leq K \qquad (3.2.1)$$

A window w(n) is next applied directly to the data segments before computation of the periodgram. Then, the K modified periodgrams as specified by

Model	Spectrum
MA	$\|B(e^{j\omega})\|^2$
AR	$\frac{\|b_0\|^2}{\|A(e^{j\omega})\|^2}$
ARMA	$\|\frac{B(e^{j\omega})}{A(e^{j\omega})}\|^2$

$$A(e^{j\omega}) = \sum_{k=0}^{p} a_k e^{-jk\omega} \quad , \quad a_0 = 1$$

$$B(e^{j\omega}) = \sum_{k=0}^{q} b_k e^{-jk\omega}$$

Table 3.1 Rational Spectrum Models

$$J_M^{(i)}(\omega) = \frac{1}{MU} \left| \sum_{n=0}^{M-1} x^{(i)}(n)\, w(n)\, e^{-j\omega n} \right|^2 \qquad i = 1, 2, \ldots, K \tag{3.2.2}$$

are computed, where

$$U = \frac{1}{M} \sum_{n=0}^{M-1} w^2(n) \tag{3.2.3}$$

and the final spectrum estimate is defined as

$$B_x(\omega) = \frac{1}{K} \sum_{i=1}^{K} J_M^{(i)}(\omega) \tag{3.2.4}$$

By taking average of periodgrams of each data segment, the desired smoothed periodgram is obtained. In using this segmentation, the variance of the spectrum is reduced. The price paid for this reduction, however, is a loss in frequency resolution and an increased bias of the estimate.

3.3 Autoregressive (AR) Model

In the last decade, much attention has been focused on the analysis of AR models. Two major spectrum estimation methods for AR models, referred as one-step linear prediction and the maximum entropy method (MEM) appeared in the literature of mathematical statistics (Parzen, 1969) and geosciences (Burg, 1967; Lacoss, 1971; Ulrych, 1972). Although these two methods take different approaches, it has been shown that they give the same spectral estimate (A van den Bos, 1971).

3.3.1 One-Step Linear Prediction

In the application of one-step linear prediction, one seeks to characterize the spectral density of a time series based upon a finite set of time observation

$$x(1),\ x(2),\ \ldots\ ,\ x(N) \tag{3.3.1.1}$$

As described in Section 3.1, the AR model is structured by

$$x(n) + a_1\, x(n-1) + \ldots + a_p\, x(n-p) = \varepsilon(n) \tag{3.3.1.2}$$

in which $\varepsilon(n)$ is a white noise time series with zero mean and variance σ_ε^2. The objective of spectral estimation will be that of modeling an underlying time series $\{x(n)\}$ with the AR model structure (3.3.1.2) in which the a_k coefficients are estimated from the given finite set of observations (3.3.1.1). This is readily achieved by applying the well known method of one-step linear prediction.

A p-th order one-step linear prediction, by definition, estimates the value of a random time series using a linear combination of the most recent p samples. Namely, the sample x(n) is estimated by means of the relationship

$$\hat{x}(n) = -\sum_{k=1}^{p} a_k\, x(n-k) \tag{3.3.1.3}$$

The difference between this predicted value and the observed value x(n) over the observation interval is called the prediction error and is specified by

$$e(n) = x(n) - \hat{x}(n) \qquad p < n \leq N \tag{3.3.1.4}$$

or

$$e(n) = x(n) + \sum_{k=1}^{p} a_k\, x(n-k) \qquad p < n \leq N \tag{3.3.1.5}$$

Writing these error expressions in matrix form yields

$$\underline{e} = \underline{x} + X\,\underline{a} \tag{3.3.1.6}$$

where $\underline{a}$, $\underline{e}$, and $\underline{x}$ are $p \times 1$, $(N-p) \times 1$, and $(N-p) \times 1$ column vectors, respectively, given by

$$\underline{a} = [a_1, \ldots, a_p]^T \tag{3.3.1.7a}$$

$$\underline{e} = [e(p+1), e(p+2), \ldots, e(N)]^T \tag{3.3.1.7b}$$

$$\underline{X} = [x(p+1), x(p+2), \ldots, x(N)]^T \tag{3.3.1.7c}$$

and X is an $(N-p) \times p$ matrix specified by

$$X = \begin{bmatrix} x(p) & x(p+1) & \cdots & x(N-1) \\ x(p-1) & x(p) & \cdots & x(N-2) \\ \vdots & \vdots & & \vdots \\ x(1) & x(2) & \cdots & x(N-p) \end{bmatrix}^T \tag{3.3.1.7d}$$

where the superscript T denotes the transpose operation.

The a_k coefficients are to be now selected so as to cause each of the prediction error terms e(n) to be close to zero. This selection process will give rise to the so-called optimal one-step predictor. To achieve the required objective of setting the e(n) to be near zero, one typically appeals to the least squares method which

minimizes a squared error criterion of the form

$$f(\underline{a}) = \underline{e}^T W \underline{e} \qquad (3.3.1.8)$$

where W is an (N - p) x (N - p) nonnegative definite square matrix. The minimization of this quadratic functional with respect to the column vector $\underline{a}$ is straightforwardly carried out and results in

$$X^T W X \underline{a}^o = X^T W \underline{x} \qquad (3.3.1.9)$$

It can be shown that the resulting power spectral density estimate of the time series {x(n)} is then given by

$$S_x(\omega) = \frac{\sigma_\varepsilon^2}{|1 + a_1^o e^{-j\omega} + a_2^o e^{-j2\omega} + \ldots + a_p^o e^{-pj\omega}|^2} \qquad (3.3.1.10)$$

where the a_k^o coefficients are obtained upon solving relationship (3.3.1.9).

3.3.2 Maximum Entropy Method (MEM)

The MEM is a result of Burg's attempt (Burg, 1967) to derive a procedure for increasing spectral resolution when only a small number of samples or estimates of autocorrelation function are available. As mentioned in Section 2.3.2, in the autocorrelation method one first estimates the autocorrelation function, append zeroes to increase the length of the estimated autocorrelation, and then applies the Fourier transform. In contrast, the MEM suggests that the estimated autocorrelation function should be extrapolated beyond the data

limited range. The principle used for this extrapolation process is that the spectral estimate must be the most random or have the maximum entropy of any power spectrum which is consistent with the sample values of the estimated autocorrelation.

In the analysis of MEM, it is assumed that we possess a partial autocorrelation sequence $\{r(0), r(\pm 1), \ldots, r(\pm M)\}$ which is a subset of a infinite extent autocorrelation function $\{r(0), r(\pm 1), \ldots\ldots\}$. It is desired that we produce from this partial autocorrelation sequence a spectral representation

$$S_r(\omega) = \sum_{n=-\infty}^{\infty} r(k)\, e^{-j\omega n} \qquad (3.3.2.1)$$

which is a Fourier transform of the autocorrelation function of infinite length. For some spectral density function $S_f(\omega)$, we may associate a time series $\{f(n)\}$ by means of inverse Fourier transform

$$f(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} S_f(\omega)\, e^{j\omega n}\, d\omega \quad \text{for } n = 0, \pm 1, \ldots \qquad (3.3.2.2)$$

so that

$$r(n) = f(n) \qquad \text{for } n = 0, \pm 1, \ldots, \pm M \qquad (3.3.2.3)$$

This expression does not provide us with a unique expression for the spectrum $S_r(\omega)$. To overcome this difficulty, Burg developed a new spectral estimator called the maximum entropy method (Burg, 1967). The entropy associated with power spectrum density $S_r(\omega)$ is defined to be

$$H = \int_{-\pi}^{\pi} \log [S_r(\omega)] \, d\omega \qquad (3.3.2.4)$$

Maximizing the entropy with respect to the unknown r(n) for $|n| > M$ with the constraint

$$r(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} S_r(\omega) \, e^{j\omega n} \, d\omega \quad \text{for } |n| > M \qquad (3.3.2.5)$$

results in the maximum entropy spectral estimate. This estimate expresses maximum uncertainty with respect to the unknown information that is consistent with the known information. The problem of estimating $S_r(\omega)$ becomes a calculus of variations problem. The solution procedure which begins with the introduction of a Lagrange multiplier for each of the constraint equations is not difficult and results in the spectral estimate (Burg, 1967)

$$\hat{S}_r(\omega) = \frac{P_M}{|1 + a_1^o \, e^{-j\omega} + \ldots + a_M^o \, e^{-jM\omega}|^2} \qquad (3.3.2.6)$$

where optimum selection of a_k coefficients a_k^o ($k = 1, \ldots, M$) are obtained by solving the following matrix system of equations

$$\begin{bmatrix} r(0) & r(1) & \ldots & r(M) \\ r(1) & r(0) & \ldots & r(M-1) \\ \vdots & \vdots & & \vdots \\ r(M) & r(M-1) & \ldots & r(0) \end{bmatrix} \begin{bmatrix} 1 \\ a_1 \\ \vdots \\ a_M \end{bmatrix} = \begin{bmatrix} P_M \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (3.3.2.7)$$

Equation (3.3.2.7) can be solved efficiently by using Levinson's Algorithm which requires $0(M^2)$ computations (Levinson, 1947).

3.4 ARMA Model

A variety of procedures have been developed for generating ARMA spectral models. These include the whitening filter approach which is typically iterative in nature, generally slow in convergence, and, usually requires an excessively large number of time series' observations to be effective (Tretter and Steiglitz, 1967; Gutowski, Robinson and Treitel, 1978). More desirable closed form procedures which overcome these deficiencies have been offered. These include the so-called Box-Jenkins method and its variants (Box and Jenkins, 1976; Kaveh, 1979; Kinkel, Perl, Scharf and Stubberud, 1979), and, more recently, Cadzow has developed a "high performance" method (Cadzow, 1981). In this section, three ARMA methods, namely, the Whitening method, Gutowski ARMA method and Box-Jenkins method are briefly discussed.

3.4.1 Whitening Method

If we assume that the Gaussian random series $\{x(n)\}$ is given, the method of maximum likelihood (Haykin, 1979) can be used to estimate the coefficients of rational spectrum in the following way. Suppose the time sequence $\{x(n)\}$ is passed through a transfer function $A(e^{j\omega})/B(e^{j\omega})$ to give the output sequence $\{\varepsilon(n)\}$. The spectrum of $\{\varepsilon(n)\}$ is given by

$$S_\varepsilon(\omega) = \left| \frac{A(e^{j\omega})}{B(e^{j\omega})} \right|^2 S_x(\omega) \qquad (3.4.1.1)$$

If one could choose the coefficients of $A(e^{j\omega})$ and $B(e^{j\omega})$ so that $S_\varepsilon(\omega) = \sigma_\varepsilon^2$, the spectral density of $\{x(n)\}$ would be given by

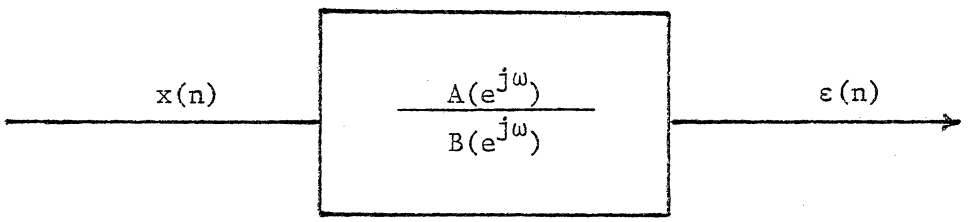
$$S_x(\omega) = \left| \frac{B(e^{j\omega})}{A(e^{j\omega})} \right|^2 \sigma_\varepsilon^2 \qquad (3.4.1.2)$$

In this case, $\{\varepsilon(n)\}$ is a white Gaussian process. The maximum likelihood parameter estimation is equivalent to finding the minimum of a function of several variables (Tretter and Steiglitz, 1967). This is called the minimum residual criterion and, intuitively, one attempts to "whiten" $\{x(n)\}$ as much as possible. The whitening process is suggestively depicted in Fig. 3.4.1.1.

Because of the rational spectrum model's structure, the minimum residual criterion leads to nonlinear equations which cannot be solved explicitly. This suggests the using of an iterative technique to optimize the denominator and numerator coefficients. Many such techniques are available, ranging from steepest descent to the Newton-Raphson algorithm.

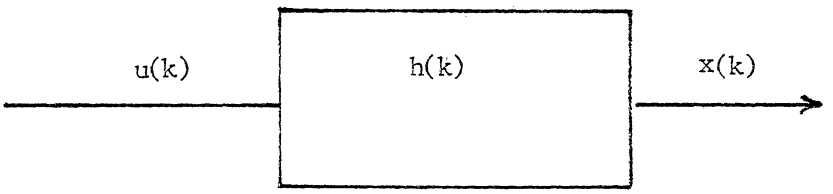
3.4.2 Gutowski ARMA Method

This section discusses the theoretical motivation for the ARMA modeling technique described by Gutowski (Gutowski, Robinson, Treitel, 1978). Consider the discrete time linear system shown in Fig. 3.4.2.1 with input u(k), output x(k), and



$$S_x(\omega) = \left| \frac{B(e^{j\omega})}{A(e^{j\omega})} \right|^2 \sigma_\varepsilon^2$$

Fig. 3.4.1.1 Spectrum Estimation by Whitening Approach



$$X(z) = H(z)\ U(z)$$

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

Fig. 3.4.2.1 Time Invariant Linear System

impulse response h(k). If the transfer function H(z) is assumed to be a rational function of z, then it may be written as

$$H(z) = \frac{B(z)}{A(z)} \qquad (3.4.2.1)$$

where A(z) and B(z) are polynomials of z of order p and q, respectively. This assumption in turn implies that the output is described by

$$X(z) = \frac{B(z)}{A(z)} \; U(z) \qquad (3.4.2.2)$$

where X(z) and U(z) denote z-transform of {x(k)} and {u(k)}, respectively. Gutowski's ARMA method assumes that u(k) is equal to the Kronecker delta function and it therefore follows that

$$\frac{B(z)}{A(z)} = X(z) \qquad (3.4.2.3)$$

Gutowski's method uses Equation (3.4.2.3) in an iterative procedure to estimate A(z) and B(z) from the data sequence {x(k)}. Each iteration may be described in terms of the following three equations:

$$A(z)\, X(z) = B(z) \qquad (3.4.2.4)$$

$$C(z) = \frac{1}{A(z)} \qquad (3.4.2.5)$$

$$C(z)\, B(z) = X(z) \qquad (3.4.2.6)$$

The basic iterative technique may be seen by using equation (3.4.2.4) through (3.4.2.6) and assuming that one starts with a reasonably good estimate of B(z). At k-th iteration, the following steps are required.

Step 1 Compute $A^{(k)}(z)$ with $X^{(k)}(z)$ input and $B^{(k)}(z)$ as desired output.

Step 2 Compute $C^{(k)}(z)$ by synthetic division of the value 1 by $A^{(k)}(z)$.

Step 3 Compute $B^{(k)}(z)$ with $C^{(k)}(z)$ as input and $X^{(k)}(z)$ as desired output.

After each iteration, if $A^{(k)}(z)$ and $B^{(k)}(z)$ are better than the previous iteration, then the fit will improve. At the completion of m-th iterations, the ARMA spectral estimate is given by

$$S_x(\omega) = \left| \frac{B^{(m)}(e^{j\omega})}{A^{(m)}(e^{j\omega})} \right|^2 \qquad (3.4.2.7)$$

The above procedure is repeated until convergence occurs. The minimum delay characteristics of $A^m(z)$ is guaranteed by the fact that the inverse is computed using a Toeplitz formulation. This is the strong point of this algorithm.

3.4.3 Box-Jenkins Method

The ARMA model with order (p, q) can be characterized by the following recursive relationship

$$x(n) = -\sum_{k=1}^{p} a_k \, x(n-k) + \sum_{k=0}^{q} b_k \, \varepsilon(n-k) \qquad (3.4.3.1)$$

$$n = p + 1, \ldots, +\infty$$

where $\{\varepsilon(k)\}$ is a white noise with variance σ_ε^2. The autocorrelation function of the mixed process may be derived by multiplying each

side of (3.4.3.1) by $x^*(n-m)$ and taking expectations to yield

$$r_x(m) = \sum_{k=1}^{p} a_k r_x(m-k) + \sum_{k=0}^{q} b_k r_{x\varepsilon}(m-k) \qquad (3.4.3.2)$$

where $r_x(n)$ and $r_{x\varepsilon}(n)$ denote the autocorrelation of the sequence $\{x(k)\}$ and cross covariance function between $\{x(k)\}$ and $\{\varepsilon(k)\}$, respectively. Since $x(n-k)$ depends only on inputs which have occurred up to time $n-k$, it then follows that

$$r_{x\varepsilon}(n) = 0 \qquad n > 0 \qquad (3.4.3.3a)$$

$$r_{x\varepsilon}(n) \neq 0 \qquad n \leq 0 \qquad (3.4.3.3b)$$

We see that (3.4.3.2) implies

$$r_x(n) = -\sum_{k=1}^{p} a_k r_x(n-k) \qquad \text{for } n \geq q+1 \qquad (3.4.3.4)$$

and yields the following matrix system of equations

$$\begin{bmatrix} r_x(q) & \cdots & r_x(q-p+1) \\ \vdots & & \vdots \\ r_x(q+p-1) & \cdots & r_x(q) \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} r_x(q+1) \\ \vdots \\ r_x(q+p) \end{bmatrix} \qquad (3.4.3.5)$$

The a_k coefficients will be obtained by solving the equation (3.4.3.5). The numerator dynamics of the ARMA model is characterized by c_k coefficients (Kaveh, 1979) which can be expressed as

$$c_k = c_{-k} = \sum_{i=0}^{p} \sum_{j=0}^{p} \alpha_j \alpha_j r(|i-j-k|) \qquad (3.4.3.6)$$

$$k = 0, 1, \ldots, q$$

where $\alpha_0 = 1$ and $\alpha_i = -a_i$ for $i = 1, \ldots, p$. The ARMA spectrum representation is then found to be

$$S_x(\omega) = \frac{\sum_{k=-q}^{q} c_k e^{-j\omega k}}{\left| 1 + \sum_{k=1}^{p} a_k e^{-j\omega k} \right|^2} \qquad (3.4.3.7)$$

Chapter 4

HIGH PERFORMANCE ARMA MODEL

4.1 Introduction

It is widely recognized that an ARMA spectral model is generally the most effective linear rational model from a parameter parsimony viewpoint (see Section 3.1). In recognition of this fact, a variety of procedures have been developed for generating ARMA models (Steiglitz, 1977; Box and Jenkins, 1976; Kaveh, 1979; Kinkel, Perl, Scharf and Stubberud, 1979). Some of these methods were discussed in Section 3.4. As indicated in Section 3.4, it is recognized that these methods share certain deficiencies. To overcome these deficiencies, the 'high performance' ARMA method was developed (Cadzow, 1979, 1980, a,b). It provides an excellent spectral estimation performance when compared with other spectral estimation methods. In this chapter, the 'high performance' method is described and numbers of numerical examples are provided. This chapter is basically identical to references (Cadzow, 1979, 1980 a, b). The development of this method is based upon some fundamental concept governing ARMA time series which will be discussed in next section.

4.2 Fundamental Concepts

The stationary random time series $\{x_k\}$ whose power spectrum is of a rational form may be modeled as the response of the causal ARMA

system of order (p, q)

$$x_k + \sum_{i=1}^{p} a_i x_{k-i} = \sum_{i=0}^{q} b_i \varepsilon_{k-i} \qquad (4.2.1)$$

where the time series $\{\varepsilon_k\}$ is taken to be a zero mean white noise excitation signal. The autocorrelation description of this system is obtained by first multiplying each side of expression (4.2.1) by the entity x^*_{k-m} and then taking the expected value. This results in the well known Yule-Walker equations as specified by

$$r_x(m) + \sum_{i=1}^{p} a_i r_x(m - i) = 0 \qquad \text{for } m > q \qquad (4.2.2)$$

The Yule-Walker equations (4.2.2) will serve as the basis for estimating the ARMA model's denominator coefficients (i.e., a_k coefficients).

4.3 Denominator Coefficient Selection

In this section, a novel procedure for estimating an ARMA model's denominator coefficients shall be presented (Cadzow, 1979, 1980 a). This development is begun by first evaluating the model equation (4.2.1) over the integer set $p + 1 \leq k \leq n$ to obtain the time series relationships

$$\begin{bmatrix} x_{p+1} \\ x_{p+2} \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} x_p & x_{p-1} & \cdots & x_1 \\ x_{p+1} & x_p & \cdots & x_2 \\ \vdots & \vdots & & \vdots \\ x_{n-1} & x_{n-2} & \cdots & x_{n-p} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} =$$

$$= \begin{bmatrix} \epsilon_{p+1} & \epsilon_{p} & \cdots & \epsilon_{p-q+1} \\ \epsilon_{p+2} & \epsilon_{p+1} & \cdots & \epsilon_{p-q+2} \\ \vdots & \vdots & & \vdots \\ \epsilon_{n} & \epsilon_{n-1} & \cdots & \epsilon_{n-q} \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_q \end{bmatrix} \qquad (4.3.1)$$

It will be compactly written in the matrix format

$$\underline{x} + X\,\underline{a} = \in \underline{b} \qquad (4.3.2)$$

where $\underline{x}$, $\underline{a}$ and $\underline{b}$ is $(n-p)x1$, $px1$ and $(n-p)x1$ column vector, respectively. The symbols X and $\in$ denote $(n-p)xp$ and $(n-p)\ x\ (q+1)$ Toeplitz type matrices, respectively. The entries of these vectors and matrices are directly obtained from expression (4.3.1).

It is now desired to utilize relationship (4.3.1) in conjunction with the Yule-Walker equations (4.2.2) to effect a procedure for estimating the ARMA model's autoregressive coefficients. As we will see, this objective is attained by first introducing the following $(n-p)xt$ Toeplitz type matrix

$$Y = \begin{bmatrix} x_{p-q} & x_{p-q-1} & \cdots & x_{p-q-t+1} \\ x_{p-q+1} & x_{p-q} & \cdots & x_{p-q-t+2} \\ \vdots & \vdots & & \vdots \\ x_{n-q-1} & x_{n-q-2} & \cdots & x_{n-q-t} \end{bmatrix} \qquad (4.3.3)$$

where the convention is adopted of setting to zero any matrix entry x_k for which k lies outside the observation set $1 \leq k \leq n$. The integer t which specifies the number of columns of this matrix will also be found to correspond to the number of Yule-Walker equations that are being approximated (i.e., relationship (4.2.2) for $q < m \leq q + t$). It thus follows that this integer parameter must be selected to at least equal p (i.e., $t \geq p$) so as to assure a well defined set of equations for the p autoregressive coefficients.

The above mentioned Yule-Walker equation approximation is achieved by premultiplying each side of relationship (4.3.2) by the complex conjugate transpose of matrix Y as denoted by $Y^{\dagger}$ to yield

$$Y^{\dagger} \underline{x} + Y^{\dagger} X \underline{a} = Y^{\dagger} \in \underline{a} \tag{4.3.4}$$

To demonstrate that this system of equations yields a logical choice for the Yule-Walker equation approximations, let us now take the expected value of each of its sides. This is found to result in

$$(n - m) \{r_x(m) + \sum_{k=1}^{p} a_k r_x(m - k)\} = 0 \tag{4.3.5}$$

$$\text{for} \quad q < m \leq q + t$$

Thus, the system of linear equations (4.3.4) is seen to provide an unbiased estimate of the underlying Yule-Walker equations. It is to be noted that the right hand side term has zero expected value due to the fact that the expected value of the matrix $Y^{\dagger} \in$ is the null matrix. This is a direct consequence of the ARMA model's causality and the whiteness of the excitation process which results in $E \{x_n^* \varepsilon_k\} \equiv 0$

for all $n < k$.

With these thoughts in mind, a logical procedure for selecting the ARMA model's autoregressive coefficients is suggested. Namely, they will be selected so as to cause the left hand side of relationship (4.3.4) to be close to its expected value which is the zero vector (i.e., $E\{Y^{\dagger} \in \underline{b}\} = \underline{\theta}$). If this selection procedure is adopted, an approximation of the Yule-Walker equations which in some sense is "most consistent" with the given time series observations is at hand. A computationally tractable measure of the closeness to which the left side of relationship (4.3.4) is to the zero vector is provided by the following quadratic functional

$$f(\underline{a}) = [Y^{\dagger}\ \underline{x} + Y^{\dagger}\ X\ \underline{a}]^{\dagger} \Lambda\ [Y^{\dagger}\ \underline{x} + Y^{\dagger}\ X\ \underline{a}] \tag{4.3.6}$$

in which Λ is a $t \times t$ positive-semidefinite diagonal matrix whose diagonal elements are chosen to possibly weight differently various elements of the error vector $Y^{\dagger}\ \underline{x} + Y^{\dagger}\ X\ \underline{a}$. It is a simple matter to show that a minimizing autoregressive coefficient vector must satisfy the consistent system of p linear equations

$$X^{\dagger}\ Y\ \Lambda\ Y^{\dagger}\ X\ \underline{a} = -X^{\dagger}\ Y\ \Lambda\ Y^{\dagger}\ \underline{x} \tag{4.3.7}$$

in the p autoregressive coefficient unknowns. One then solves this system of p equations for the most data consistent set of autoregressive coefficient estimates.

4.4 Numerator Dynamics

A variety of procedures exists for determining the numerator dynamics of an ARMA time series once the AR coefficients have been estimated. In this section, two procedures which have been found to be particularly effective shall be described. Each makes use of the governing ARMA relationship that models the underlying time series.

4.4.1 Yule-Walker Equation Method (Cadzow, 1979)

In this approach to estimating the numerator dynamics, we first introduce the so-called causal image of a time series autocorrelation sequence as specified by

$$r_x^+(n) = -\tfrac{1}{2}\, r_x(0)\, \delta(n) + r_x(n)\, u(n) \qquad (4.4.1.1)$$

in which $\delta(n)$ and $u(n)$ designate the unit-sample and unit-step sequences, respectively. Making use of the complex conjugate symmetrical property of stationary autocorrelation sequences, it then follows that the autocorrelation sequence can be uniquely recovered from its causal image according to the simple relationship

$$r_x(n) = r_x^+(n) + r_x^+(-n)^* \qquad (4.4.1.2)$$

Upon taking the discrete-Fourier transform of this relationship, it follows that the time series spectral density is given by

$$\begin{aligned} S_x(\omega) &= S_x^+(\omega) + [S_x^+(\omega)]^* \\ &= 2\, R_e\, [S_x^+(\omega)] \end{aligned} \qquad (4.4.1.3)$$

where $S_x^+(\omega)$ denotes the discrete Fourier transform of the causal image $r_x^+(n)$. According to relationship (4.4.1.3), one may attain a spectral density estimate by estimating $S_x^+(\omega)$. This will be the approach taken in this section.

An estimation of the Yule-Walker equations (4.2.2) which govern the ARMA model time series indicates that the causal image sequence will generate the auxiliary $\{c_k\}$ sequence according to

$$c_m = r_x^+(m) + \sum_{k=1}^{p} a_k \, r_x^+(m - k) \qquad (4.4.1.4)$$

$$m = 0, 1, \ldots, s \quad \text{for} \quad s = \max(q,p)$$

It is to be noted that the $\{c_k\}$ sequence will be identically zero outside the time range $0 \leq k \leq s$. Upon taking the discrete Fourier transform of relationship (4.4.1.4), we have $S_x^+(\omega)$ in the form

$$S_x^+(\omega) = \frac{c_0 + c_1 \, e^{-j\omega} + \ldots + c_s \, e^{-js\omega}}{1 + a_1 \, e^{-j\omega} + \ldots + a_p \, e^{-jp\omega}} \qquad (4.4.1.5)$$

If this expression is substituted into relationship (4.4.1.3), the required formulation of the spectral density estimate is completed.

4.4.2 Smoothed Periodgram Method (Cadzow, 1980 b)

In the smoothed periodgram method, one first generates the auxiliary "residual" time series elements according to the relationship

$$e(k) = x(k) + \sum_{i=1}^{p} a_i \, x(k - i) \qquad p + 1 \leq k \leq n \qquad (4.4.2.1)$$

in which the ARMA model's a_k coefficients as generated by relationship

(4.3.7) is utilized. Upon examination of relationship (4.2.1) and under the condition that the time series being characterized is an ARMA model of order p with the calculated a_k coefficients, it follows that the residual time series will have a moving average spectral density as given by

$$S_e(\omega) = \left| \sum_{k=0}^{q} b_k \, e^{-jk\omega} \right|^2 \qquad (4.4.2.2)$$

This observation in conjunction with the ARMA model representation then provides the vehicle for estimating the underlying time series spectral density, that is

$$S_x(\omega) = S_e(\omega) \; / \; \left| \sum_{k=0}^{p} a_k \, e^{-jk\omega} \right|^2 \quad , \; a_0 = 1 \qquad (4.4.2.3)$$

With this in mind, the final step of the spectral estimation procedure requires fitting a q-th order moving average (MA) model to the residual time series segment (4.4.2.1) to effect an estimate of $S_e(\omega)$.

The approach to be presented for obtaining the $q+1^{st}$ order MA model is an adaption of the well-known method of Welch for obtaining smoothed periodgrams (Welch, 1967). In essence, one first segments the calculated residual elements (4.4.2.1) into L segments each of length q + 1 according to

$$e_i(k) = w(k) \; e(k + 1 + p + id) \qquad (4.4.2.4)$$

$$0 \leq k \leq q$$

$$0 \leq i \leq L - 1$$

where w(n) is a data window and "d" is a positive integer which specifies the time shift between adjacent segments. These individual segments are seen to overlap for a shift selection of $d \leq q$. Furthermore, in order to include only the observed time series elements, the relevant parameter must be selected so that $p + q + (L - 1)d < n$. Finally, the q + 1 order periodogram of each of the L segments (4.4.2.4) is taken, and, these periodograms are in turn averaged to obtain the desired smoothed q + 1 order MA estimate given by

$$\hat{S}_e(\omega) = \frac{1}{L} \sum_{i=0}^{L-1} \{\frac{1}{q+1} \left| \sum_{k=0}^{q} w(k)\, e(k + 1 + p + id) e^{-j\omega k} \right|^2 \} \quad (4.4.2.5)$$

where the data window is normalized according to $\Sigma\, w^2(k) = 1$.

In using this smoothing procedure, the variance of the estimate $\hat{S}_e(\omega)$ is reduced. The price paid for this reduction, however, is a loss in frequency resolution and an increased bias of the estimate. Fortunately, the basic resolution capability of this and other ARMA model procedures is primarily influenced by the autoregressive coefficient selection. If one is mainly interested in resolution performance, an examination of the ARMA models' pole locations then need be investigated.

4.5 Numerical Examples

In this section, the classical problem of detecting the presence of sinusoids in additive noise is considered. In particular, we will investigate the specific case in which the time series observations

are generated according to

$$x(n) = A_1 \cos(\pi f_1 n) + A_2 \cos(\pi f_2 n) + w(n) \qquad (4.5.1)$$

$$1 \leq n \leq N$$

where w(n) is a white Gaussian time series with variance one. This particular problem serves as an excellent vehicle for measuring a spectral estimator's performance relative to: (i) detecting the presence of sinusoids in a strong noisy background, and (ii) resolving two sinusoids whose frequencies f_1 and f_2 are nearly equal. The individual sinusoidal signal-to-noise ratios (SNR) for the above signal are given by 20 log $(A_k/\sqrt{2})$ for k = 1,2. In order to consider the effectiveness of the high performance ARMA spectral estimator in different noise environments, we shall consider two cases. These cases have been examined in reference (Sullivan, etc., 1978) where the performance of many modern spectral estimators are empirically compared.

CASE I: $A_1 = \sqrt{20}$, $f_1 = 0.4$

$A_2 = \sqrt{2}$, $f_2 = 0.426$

In this example, we have two closely spaced (in frequency) sinusoids for which the stronger sinusoid has a SNR of 10 dB while the weaker sinusoid has a SNR of 0 dB. For this relatively low SNR case, the ability of a spectral estimator to resolve closely spaced sinusoids and identify their frequencies will be tested. Upon generating sequence (4.5.1) with the postulated parameters for a data length of N = 1024, spectral estimates were obtained using a 12-th order model

with the high performance ARMA method (diagonal element of the matrix Λ is $(N-m)^2$), maximum entropy method, and the Box-Jenkins method incorporating biased autocorrelation estimates. In addition, a standard periodgram spectral estimate was obtained using the same data. The resultant spectral estimates are displayed in Fig. 4.5.1 where a number of observations can be made

(i) The indirect ARMA spectral estimate provides excellent results with two sharp peaks at $\hat{f}_1 = 0.400$ and $\hat{f}_2 = 0.427$, and with the spectrum near 0 dB (the noise level) for most other frequencies.

(ii) The maximum entropy and Box-Jenkins methods were unable to resolve the two sinusoids in the prevailing low SNR environment.

(iii) Although the periodgram is able to resolve the two sinusoids, the well-known random fluctuation behavior which characterizes the periodgram method is in evidence.

This example nicely demonstrates the potential capability of the high performance ARMA spectral estimation method relative to existing procedures.

In many practical problems, one does not have available exceedingly long data lengths upon which to make a spectral estimate. To demonstrate the ability of the high performance ARMA spectral estimator to perform in such situations, the first 64 samples of the data sequence in the above example were used to generate a spectral estimate. The

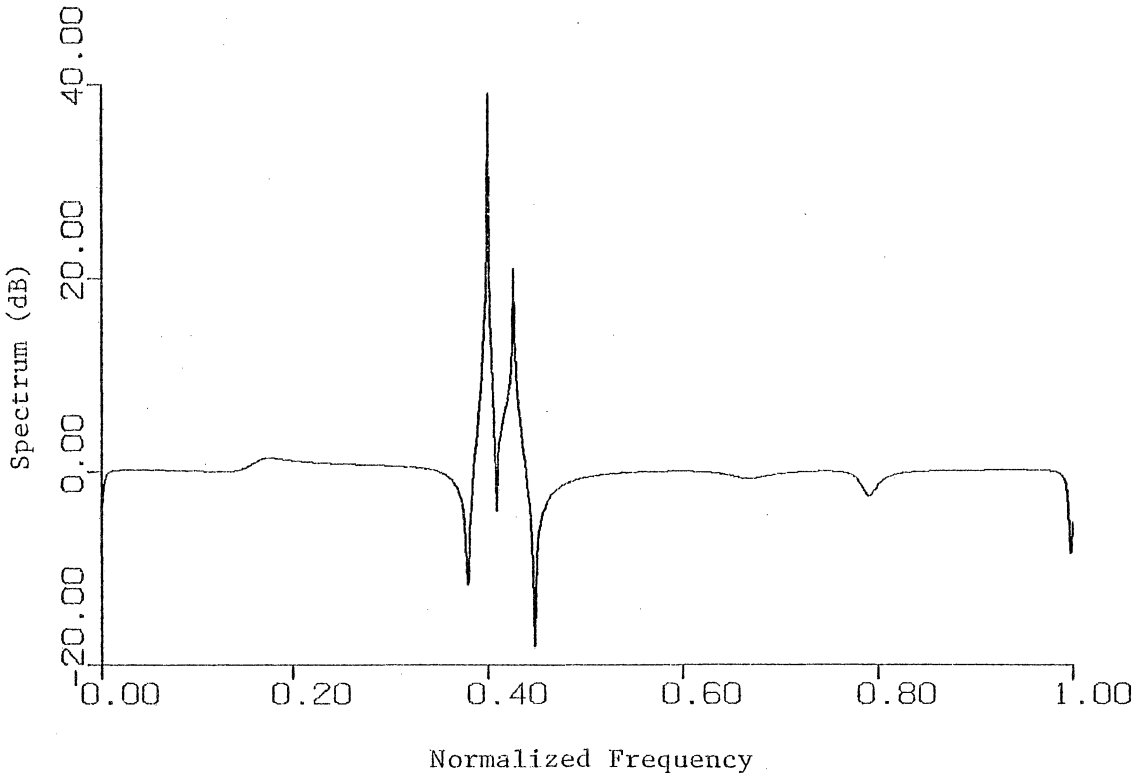


Fig. 4.5.1(a) High Performance ARMA Method (P = 12, N = 1024)

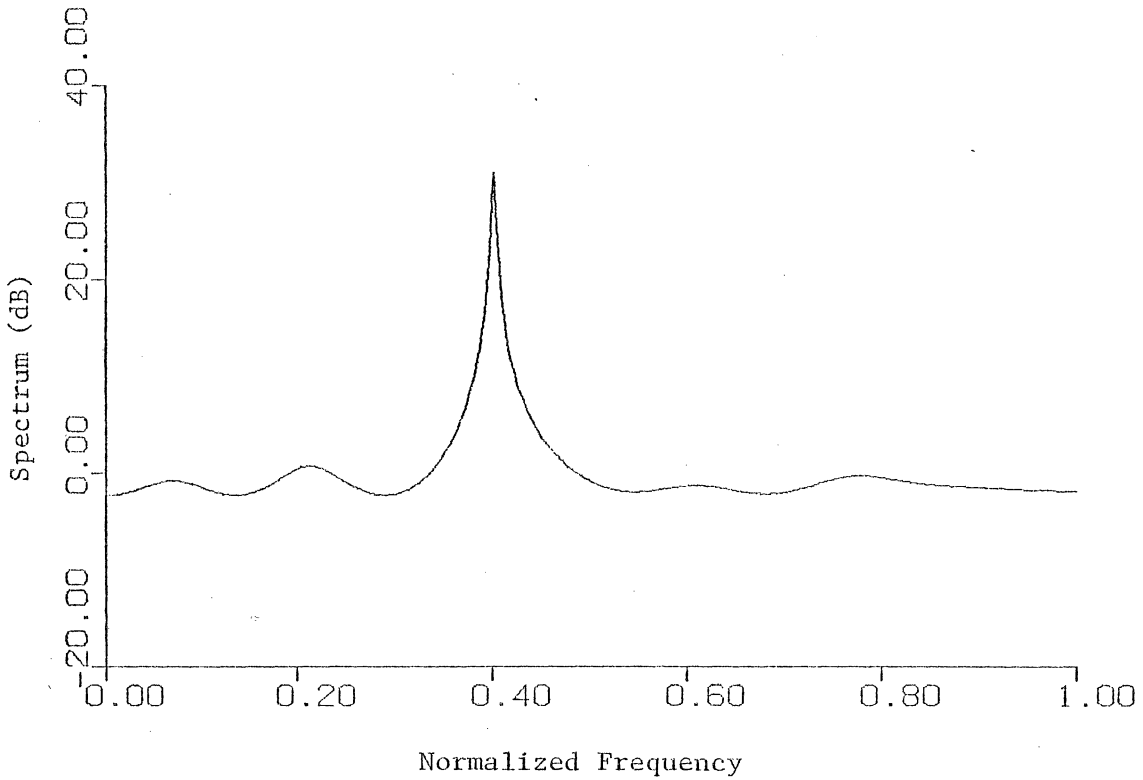


Fig. 4.5.1(b) Maximum Entropy Method (P = 15, N = 1024)

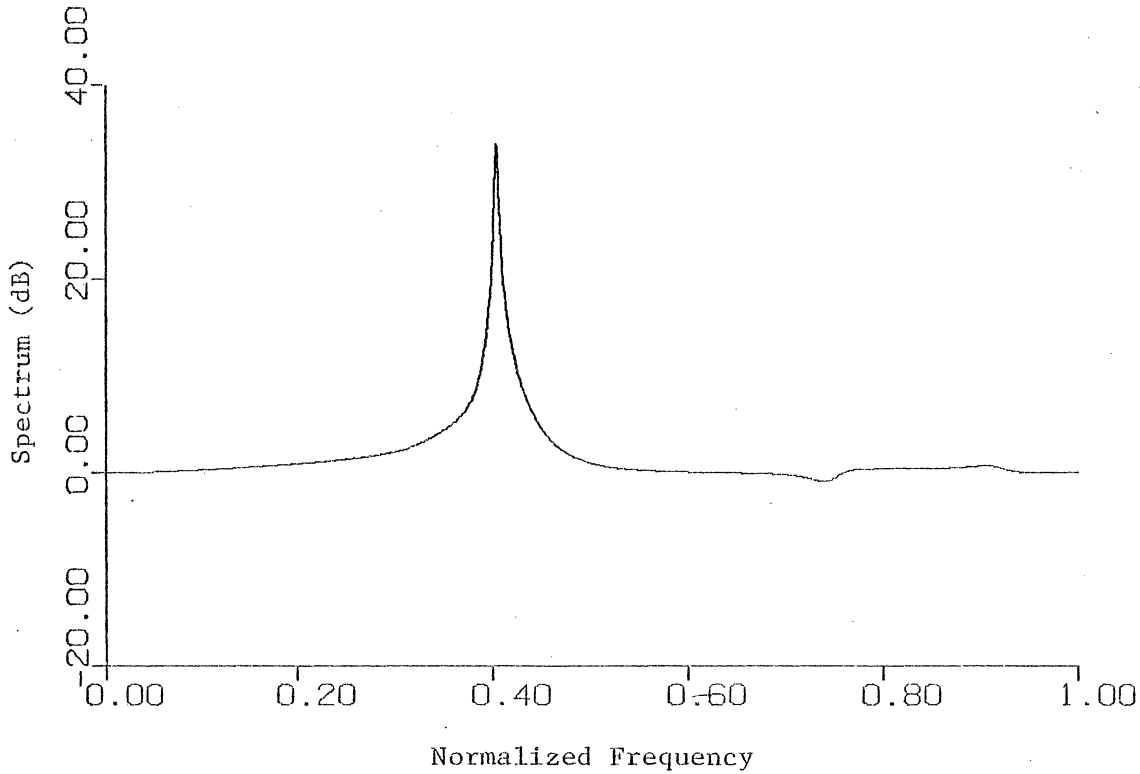


Fig. 4.5.1(c) Modified Box-Jenkins Method (P = 15, N = 1024)

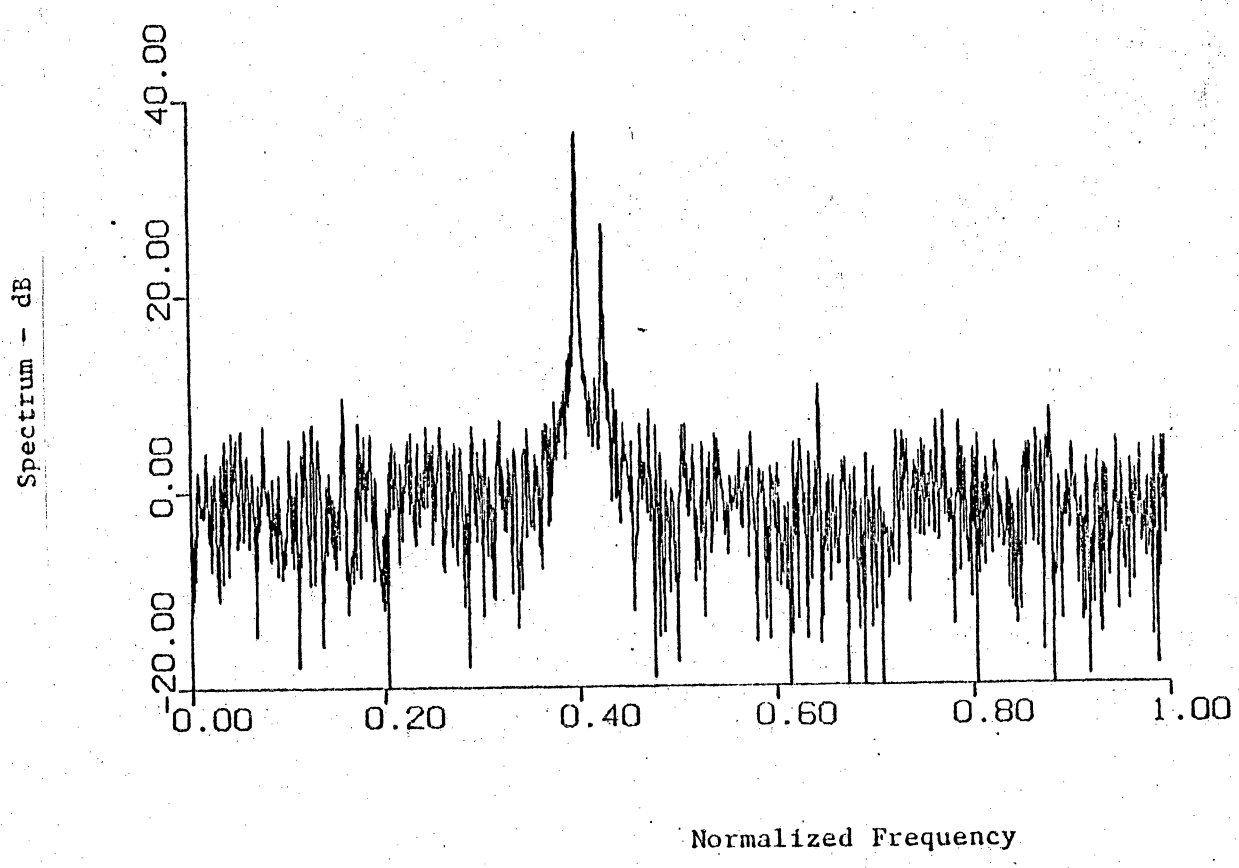


Fig. 4.5.1(d) Periodgram (N = 1024)

resultant 15-th order high performance ARMA spectral estimate obtained is shown in Fig. 4.5.2 where the ability to resolve the two closely spaced sinusoids is again evident. The sinusoid's frequency estimate $\hat{f}_1 = 0.399$ and $\hat{f}_2 = 0.423$ are also of good quality in this low SNR environment.

CASE II: $A_1 = \sqrt{2}$, $f_1 = 0.32812$

$A_2 = \sqrt{2}$, $f_2 = 0.5$

We are now examining the ability of the ARMA spectral estimator to detect sinusoids in a low SNR environment. For a selection of $N = 64$, $w(n) = (N - n)^2$ and $p = 5$, the resultant ARMA spectral estimation is displayed in Fig. 4.5.3(a). Clearly, one is able to detect the presence of the two sinusoids, and, the frequency estimate $\hat{f} = 0.3202$ and $\hat{f}_2 = 0.5012$ are of good quality considering the prevailing SNR environment. A 15-th order maximum entropy spectral estimator was then found to generate the spectral estimate displayed in Fig. 4.5.3(b). Although the two sinusoids were properly detected, a number of false peaks are in evidence.

Next, we treat the time series recently considered by Bruzzone and Kaveh (1980). Specifically, their ARMA time series is characterized by

$$x_k = x_k^1 + x_k^2 + 0.5\ \varepsilon_k \qquad (4.5.2a)$$

where the time series x_k^1 and x_k^2 are autoregressive process generated by

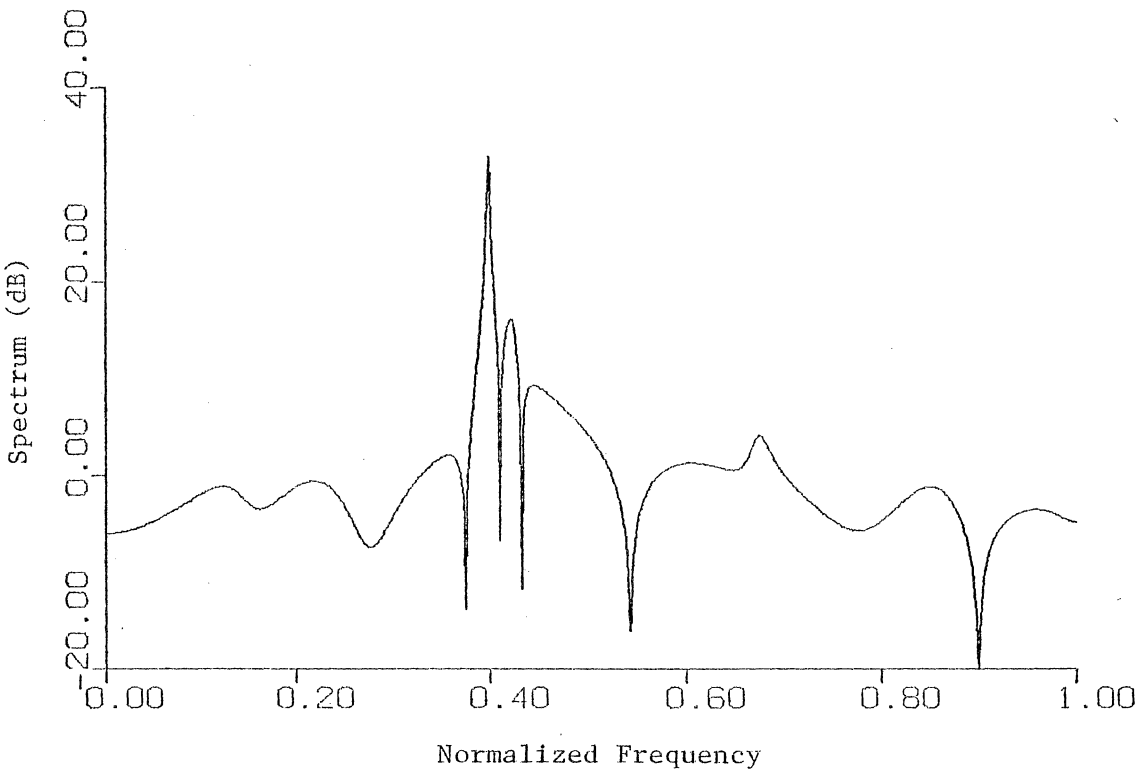


Fig. 4.5.2 High Performance ARMA Method (P = 15, N = 64)

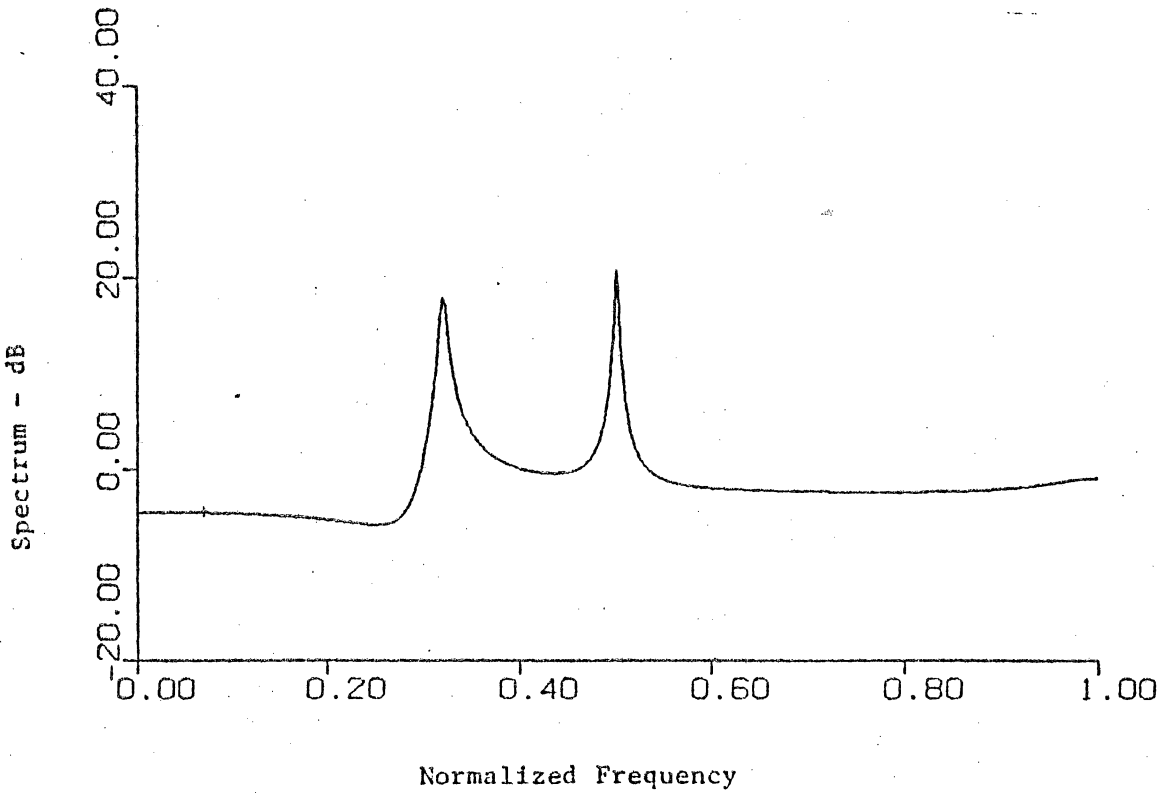


Fig. 4.5.3(a) High Performance ARMA Method (P = 5, N = 64)

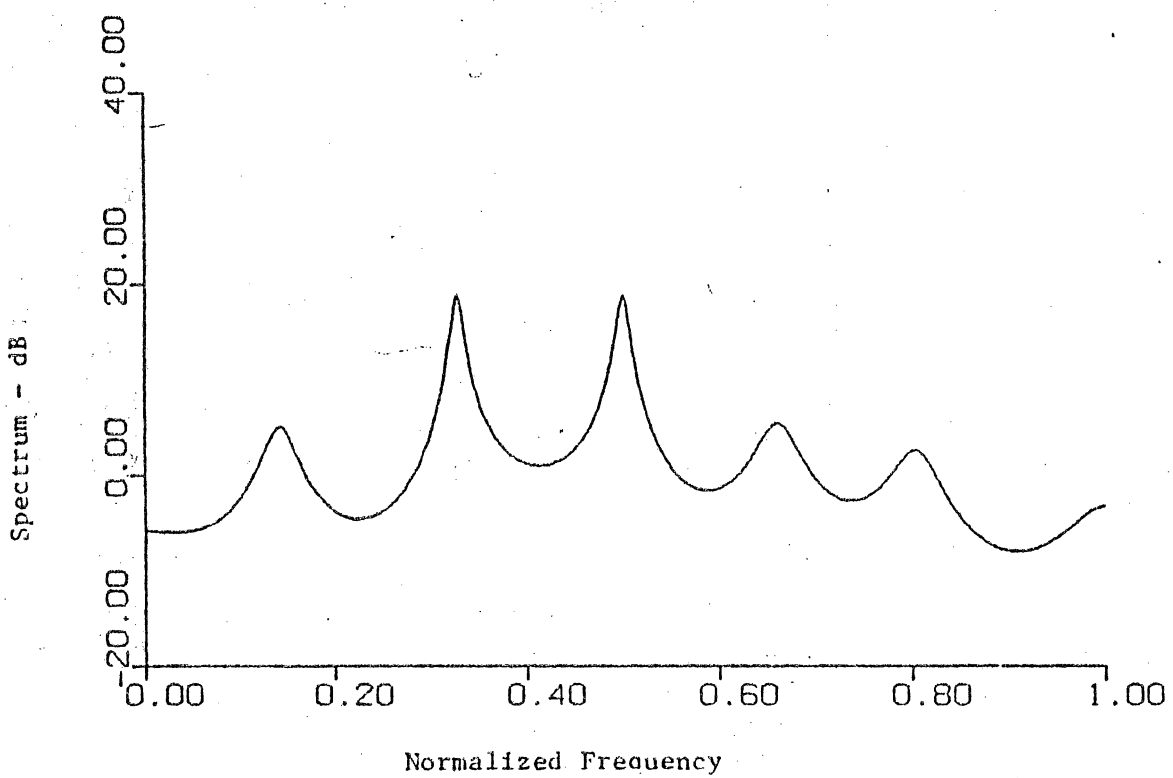


Fig. 4.5.3(b) Maximum Entropy Method (P = 15, N = 64)

$$x_k^1 = 0.4\, x_{k-1}^1 - 0.93\, x_{k-2}^1 + \varepsilon_k^1 \qquad (4.5.2b)$$

$$x_k^2 = -0.5\, x_{k-1}^2 - 0.93\, x_{k-2}^2 + \varepsilon_k^2 \qquad (4.5.2c)$$

in which the ε_k, ε_k^1 and ε_k^2 are uncorrelated Gaussian random variables with zero mean and variance 1. The spectral density of the above time series (4.5.2a) is given by

$$S_x(\omega) = \left|1 - 0.4\, e^{-j\omega} + 0.93\, e^{-j2\omega}\right|^{-2} + \left|1 + 0.5\, e^{-j\omega} + 0.93\, e^{-j2\omega}\right|^{-2} + 0.25 \qquad (4.5.2d)$$

Using this time series (4.5.2a), twenty different independent sampled sequences each of length 64 were generated. These twenty observation sets were used to test various spectral estimation methods. In Fig. 4.5.4, twenty superimposed plots of the ARMA model spectral estimates of order (4.4) as obtained by using the Box-Jenkins method, the high performance method with t = 4, 8 and 20 are shown. For comparison purposes, the ideal spectrum is also plotted. Comparing the two top most plots, the high performance method with the minimal value of t = 4 was found to yield a marginally better spectral estimate than the Box-Jenkins method. In the lower two plots, one can observe that the high performance spectral estimates improve significantly as t is increased. Next, twenty different samples sequence of length 200 were generated according to time series (4.5.2a). With this longer data length, it was anticipated that an improvement in spectral estimation performance would result. As shown in Fig. 4.5.5, a marked improvement

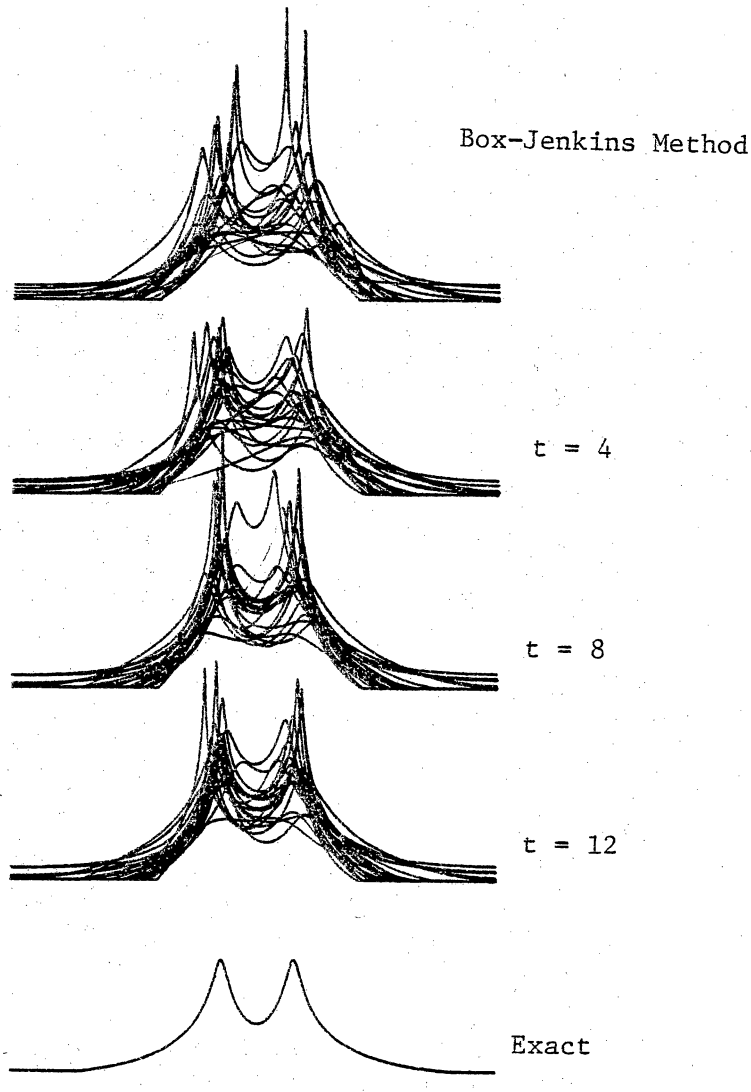


Fig. 4.5.4 ARMA Spectral Estimates of Order (4,4), Data Length of 64, and, $\lambda = 0.95$.

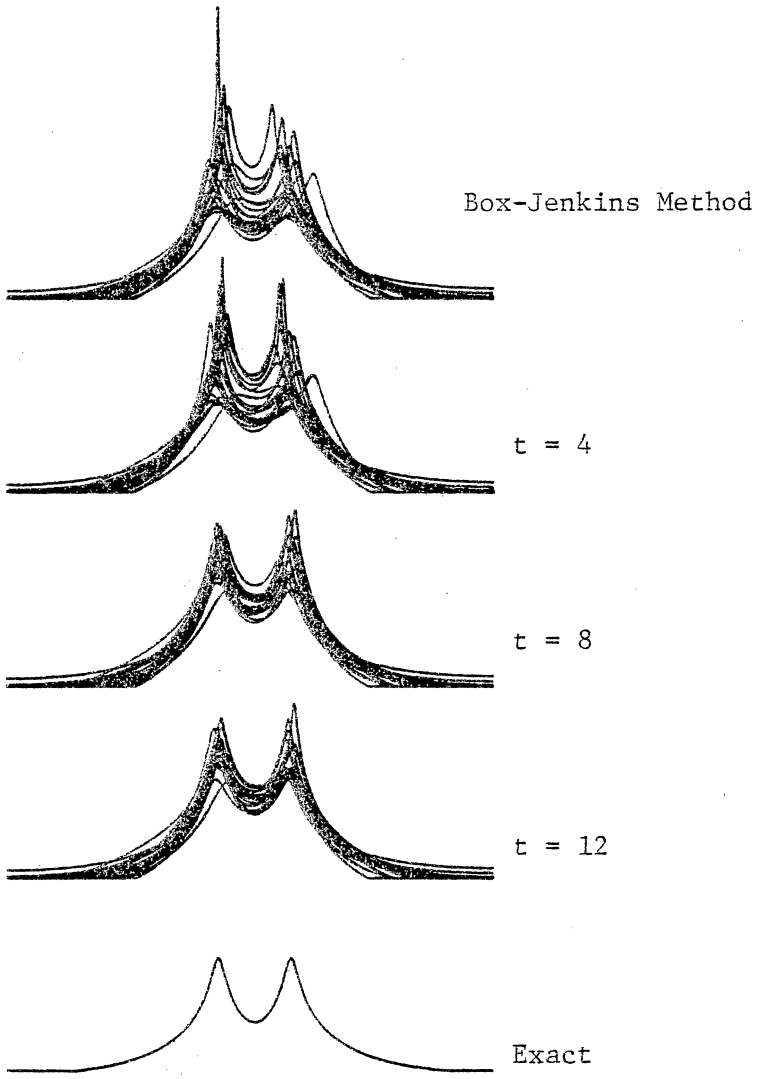


Fig. 4.5.5 ARMA Spectral Estimates of Order (4,4), Data Length of 200, and, $\lambda = 0.95$.

is evident, where the ARMA model spectral estimates of order (4.4) are shown for the Box-Jenkins method and the high performance method for selections of t = 4, 8, and 20.

It is also possible to use the high performance ARMA method for synthesizing digital filters. To illustrate the approach that is taken, let us consider the specific case of designing a low-pass filter of normalized cutoff frequency f_c. One may readily show that the impulse response of an idealized version of this low pass filter is given by $\sin(\pi f_c n)/\pi n$. With this in mind, one then applies the herein developed ARMA procedure to the specific sequence

$$x(n) = \sin[\pi f_c(n - 0.5\,N)]/\pi(n - 0.5\,N) \quad 1 \leq n \leq N \quad (4.5.3)$$

The resultant ARMA model obtained in this manner will have attenuation characteristics of the desired low-pass filter. To illustrate this, a 15-th order ARMA spectral estimate of this sequence was made for $f_c = 0.2$, $N = 128$ and $w(n) = (N-n)$. The resultant filter's magnitude characteristics are displayed in Fig. 4.5.6 where the low-pass characteristics are in evidence.

4.6 Summary

The "high performance" ARMA model spectral estimation has been described. This estimation approach provided an excellent spectral estimation performance when compared with such contemporary procedures as the maximum entropy and Box-Jenkins Methods. The above mentioned

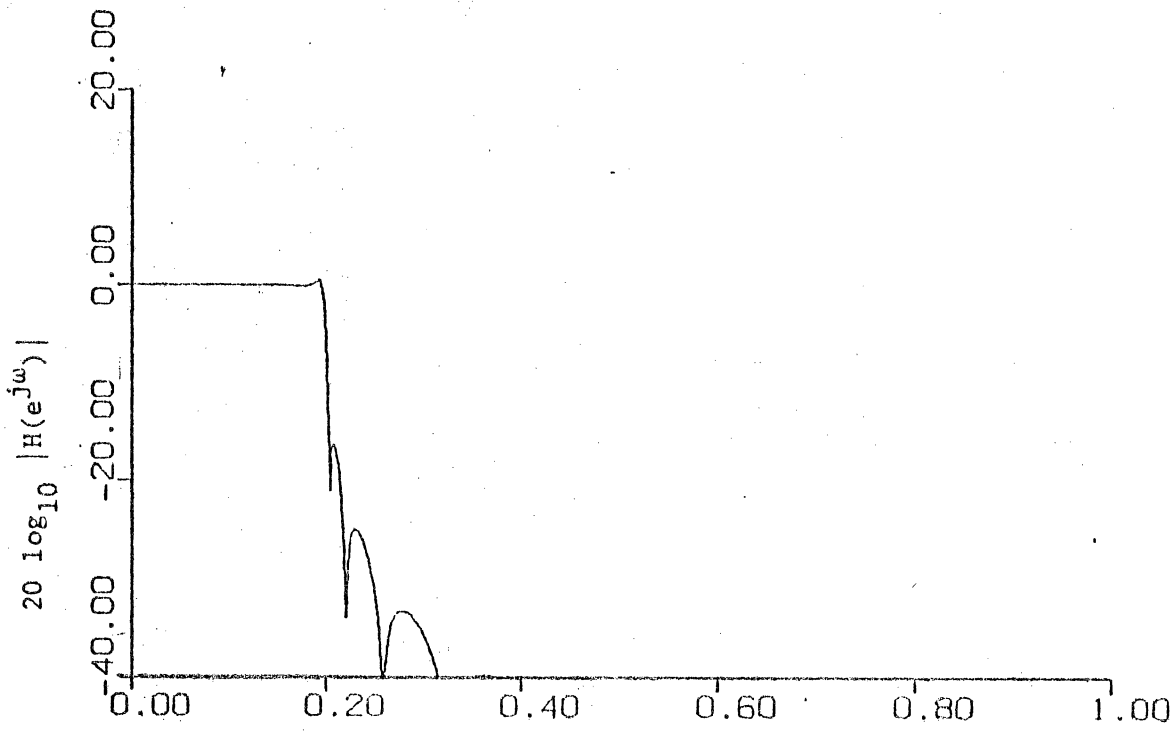


Fig. 4.5.6 Low-Pass Digital Filter Design by High Performance ARMA Method (P = 15)

"high performance" ARMA spectrum estimation will be developed further in Chapter 5 to achieve computational efficiency.

Chapter 5

COMPUTATIONALLY EFFICIENT ARMA SPECTRAL ESTIMATION

5.1 Introduction

Recently, much attention has been focused on developing spectral estimation algorithms. Unfortunately, direct application of the linear prediction method as described in Section 3.3.1 results in an excessive computational requirement, since it is necessary to solve a pxp matrix system of equations which generally requires $0(p^3)$ computations. For this reason, a number of computationally fast algorithms have been developed to overcome this difficulty. These include the Levinson's algorithm (Levinson, 1947). The Levinson algorithm is found to be dependent on the Toeplitz structure of the matrix characterizing the system of equations. With this very restrictive constraint in mind, Kailath, etc. developed the concept of the displacement rank so as to yield efficient solutions for non Toeplitz system of equations. The displacement rank measures how "close" to Toeplitz a given square matrix is (Kailath, etc., 1979). If a given matrix T is Toeplitz, then its structure is characterized by the following property

$$T = [t_{i,j}] = [t_{i+m,\ j+m}] \tag{5.1.1}$$

where $t_{i,j}$ denotes the (i,j)-th element of the pxp Toeplitz matrix T and m is a scalar integer ($1 \leq i+m,\ j+m \leq p$). That is, the elements

of the matrix T are identical along the diagonal and subdiagonal directions. In recognition of this key property of a Toeplitz structure, the displacement rank of the pxp matrix A is defined by

$$\alpha(A) = \min\{\alpha_+ (A), \alpha_- (A)\} \tag{5.1.2a}$$

where

$$\begin{aligned} \alpha_+ (A) &= \text{rank}\ \{A - S\ A\ S^T\} \\ \alpha_- (A) &= \text{rank}\ \{A - S^T\ A\ S\} \end{aligned} \tag{5.1.2b}$$

in which $\alpha_+ (A)$ and $\alpha_- (A)$ are called the positive and negative displacement ranks of matrix A, respectively, and S denotes the pxp down shift matrix defined by

$$S = \begin{bmatrix} 0 & & & \\ 1 & \ddots & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix} \tag{5.1.3}$$

It can be straightforwardly shown that the displacement rank of a Toeplitz matrix T is 2, that is

$$\alpha(T) = \alpha_+ (T) = \alpha_- (T) = 2 \tag{5.1.4}$$

If a given matrix A has a displacement rank α, then it has been shown that the inversion of A may be accomplished with the number of required computations being $0(\alpha p^2)$ (Friedlander, etc., 1979).

Based on these concepts, a number of computationally efficient algorithms for AR spectral models have been developed (Friedlander, etc., 1978, 1979; Morf, etc., 1977; Morf and Lee, 1978; Lee and Morf, 1980; Morf and Kailath, 1975; Mullis and Roberts, 1976; Morf, 1980;

Bitmead and Anderson, 1979). Some of these methods are classified by Morf, etc. (Morf, etc., 1977).

In this chapter, fast algorithms which are applicable to the "high performance" ARMA method (see Chapter 4) are developed. To achieve the fast algorithm solution capability, it will be necessary to restrict the number of Yule-Walker approximation to be p (i.e., t = p). Unfortunately, the restriction t = p will generally result in an associated decrease in spectral estimation performance. Thus, in obtaining a computationally fast algorithmic solution procedure for the a_k coefficients, an accompanying sacrifice in spectral estimation performance is the price being paid. One must therefore carefully consider the tradeoff for any given application. Fortunately, the degradation in performance is not great for many relevant applications in which the data length n adequately exceeds the ARMA model order parameters p and q.

The achievement of fast algorithms requires data modifications which will be discussed in Section 5.2. In Sections 5.3 and 5.4, algorithms which requires $0(p^2)$ and $0(p \log p)$ multiplications, respectively are discussed. An algorithm which requires 0(p) computations is developed in Chapter 6.

5.2 Data Modification

In this section, we will discuss three types of data modifications referred to as the pre-modification, post-modification and pre- and

post-modification methods (Cadzow and Ogino, 1981). These are modifications of the 'high performance' ARMA spectral estimation methods as discussed in Section 4.3 in which t is restricted to be p. It will be recalled that in this unmodified case one must solve the matrix system of equations (4.3.4). Without loss of generality, this matrix system of equations may be represented as

$$Y^{\dagger} X \underline{a} = Y^{\dagger} \underline{x} \tag{5.2.1}$$

where Y and X are (n - p) x p Toeplitz matrices, while $\underline{x}$ and $\underline{a}$ are (n - p) x 1 and p x 1 column vectors, respectively defined by

$$Y = \begin{bmatrix} x_{p-q}, & x_{p-q+1}, \cdots, x_{n-q-1} \\ \vdots & \vdots \\ x_{1-q} & x_{2-q}, \cdots, x_{n-q-p} \end{bmatrix}^{T} \tag{5.2.1a}$$

$$X = \begin{bmatrix} x_{p} & x_{p+1} \cdots x_{n-1} \\ \vdots & \vdots \\ x_{1} & x_{2} \cdots x_{n-p} \end{bmatrix}^{T} \tag{5.2.1b}$$

$$\underline{x} = [x_{p+1}, x_{p+2}, \ldots, x_{n}]^{T} \tag{5.2.1c}$$

$$\underline{a} = [a_{1}, \ldots, a_{p}]^{T} \tag{5.2.1d}$$

where the entries of the matrices X, Y and column vector $\underline{x}$ can be determined from expression (4.3.4). The entries of the column vector

$\underline{a}$ in expression (5.2.1d) denote the denominator AR coefficients to be found. It can be shown that the displacement rank of the matrix $Y^{\dagger}X$ is 4. As suggested in Section 5.1, it is possible to find an algorithmic solution procedure which requires $0(4p^2)$ computations. In fact, in Section 5.3, a generalized Levinson's algorithm will be developed.

It is possible to realize significant computational savings in the 'high performance' ARMA spectral estimation procedure. This improvement will entail a slight modification in the vector $\underline{x}$ and matrices X and Y. Although the suggested modifications will typically result in biased estimates of the Yule-Walker equations, it is shown that when the data length n adequately exceeds the order parameter p and q then these estimates are virtually unbiased (Cadzow and Ogino, 1981).

With the above high performance spectral estimation method representation serving as a basis, we shall now consider the aforementioned modifications required to achieve computationally efficient algorithmic solution procedures.

5.2.1 Pre-modification Method

In expressions (5.2.1a) and (5.2.1b), the addition of lower triangular matrices to the top of matrices X and Y yields the Toeplitz matrices

$$Y_1 = \begin{bmatrix} x_{1-q} & \cdots & x_{p-q} & \cdots & x_{n-q-1} \\ & \ddots & \vdots & & \vdots \\ & & x_{1-q} & \cdots & x_{n-q-p} \end{bmatrix}^T \qquad (5.2.1.1a)$$

$$X_1 = \begin{bmatrix} x_1 & \cdots & x_p & \cdots & x_{n-1} \\ & \ddots & \vdots & & \vdots \\ & & x_1 & \cdots & x_{n-p} \end{bmatrix}^T \qquad (5.2.1.1b)$$

with Y_1 and X_1 each being $(n - 1) \times p$ matrices. While maintaining the structure of expression (4.3.1), the vector $\underline{x}$ will be modified to

$$\underline{x}_1 = [x_2 \cdots x_{p+1} \cdots x_n]^T \qquad (5.2.1.1c)$$

Substitution of expressions Y_1, X_1 and $\underline{x}_1$ in place of Y, X and $\underline{x}$, respectively, yields

$$Y_1^\dagger X_1 \underline{a} = Y_1^\dagger \underline{x}_1 \qquad (5.2.2)$$

It can be shown that the displacement rank of the matrix $Y_1^\dagger X_1$ is 3 (Cadzow and Ogino, 1981). It is possible to find a generalized Levinson algorithm which requires $0(3p^2)$ computations to invert $Y_1^\dagger X_1$. More importantly, because of this specific structure, an algorithm which requires $0(p)$ computations has been developed and will be discussed in Chapter 6.

5.2.2 Post-modification Method

Following a similar procedure as employed in Section 5.2.1, the addition of an upper triangular matrix to the main body of the matrices specified by (5.2.1.a) and (5.2.1b) yields the Toeplitz matrices

$$Y_2 = \begin{bmatrix} x_{p-q} & \cdots & x_{n-q-1} \\ \vdots & & \vdots \\ x_{1-q} & \cdots & x_{n-q-p} \cdots x_{n-q-1} \end{bmatrix}^T \qquad (5.2.1.3a)$$

$$X_2 = \begin{bmatrix} x_{p} & \cdots & x_{n-1} \\ \vdots & & \vdots \\ x_{1} & \cdots & x_{n-p} \cdots x_{n-1} \end{bmatrix}^T \qquad (5.2.1.3b)$$

where X_2 and Y_2 are each $(n-1) \times p$ Toeplitz matrices. In a similar manner, the column vector $\underline{x}_2$ is defined by

$$\underline{x}_2 = \left[x_{p+1}, \ldots, x_n, \underbrace{0 \ldots 0}_{p_{zeros}} \right]^T \qquad (5.2.1.3c)$$

The displacement rank of the matrix $Y_2^\dagger X_2$ is found to be (Cadzow and Ogino, 1981)

$$\alpha_+ (Y_2^\dagger X_2) = \alpha_- (Y_2^\dagger X_2) = 3 \qquad (5.2.1.4)$$

thereby offering a generalized Levinson solution procedure requiring a computational complexity of $0(3p^2)$ for solving the system of equations

$$Y_2^{\dagger} X_2 \underline{a} = Y_2^{\dagger} \underline{x} \qquad (5.2.1.5)$$

A more computationally efficient algorithm associated with the post-modification will be developed herein. It is shown that the number of computations is reduced to $0(p \log p)$ if $p = q$ where p and q are the order of denominator and numerator coefficients of the ARMA model, respectively.

5.2.3 Pre- and Post-Modification Method

The combination of the previously discussed two modification methods yields the pre- and post-modification method. The matrices and vector are modified in the following manner

$$Y_3 = \begin{bmatrix} x_{1-q} & \cdots & x_{p-q} & \cdots & x_{n-1-q} & & \\ & \ddots & & \ddots & & \ddots & \\ & & x_{1-q} & \cdots & x_{n-p-q} & \cdots & x_{n-1-q} \end{bmatrix}^T \qquad (5.2.3.1a)$$

$$X_3 = \begin{bmatrix} x_1 & \cdots & x_p & \cdots & x_{n-1} & & \\ & \ddots & & \ddots & & \ddots & \\ & & x_1 & \cdots & x_{n-p} & \cdots & x_{n-1} \end{bmatrix}^T \qquad (5.2.3.1b)$$

$$\underline{x}_3 = [x_2 \; . \; . \; . \; x_{p+1} \; . \; . \; . \; x_n \quad \underbrace{0 \; . \; . \; . \; . \; 0}_{p_{zeros}}] \qquad (5.2.3.1c)$$

where Y_3 and X_3 denote $(n + p - 1)$ x p matrices and $\underline{x}_3$ is a $(n + p - 1)$ x 1 column vector respectively.

It can be shown that the matrix $Y_3^{\dagger} X_3$ is a Toeplitz matrix. A conventional approach for solving the Toeplitz system of equations

$$Y_3^{\dagger} X_3 \underline{a} = Y_3^{\dagger} \underline{x}_3 \qquad (5.2.4)$$

was developed by Levinson (Levinson, 1947), which requires $0(p^2)$ computations. More recently much effort has been conducted in developing more efficient AR algorithms whose computational requirement is 0(p log p). Gustavson, etc., presented their algorithms which were based on the use of Padé approximates and the rational Hermite approximation (Gustavson and Yun, 1979). Morf developed the so-called doubling algorithm which requires 0(p log p) (Morf, 1980). Bitmead and Anderson also independently found a doubling algorithm (Bitmead and Anderson, 1979). In Section 5.4, an application of the doubling algorithm to the ARMA model is developed.

5.3 Generalized Levinson's Approach for the ARMA Model: The Unmodified Method

In this section, an algorithm which can be applied to the direct approach (i.e., no modification) will be developed. Without loss of generality, the m x m matrix $R_{1,m}^{n}$ will be defined by

$$R^{n}_{1,m} = [Y^{n}_{1,m}]^{T} X^{n}_{1,m} \tag{5.3.1a}$$

where $X^{n}_{1,m}$ denotes $(n - m + 1)$ x m matrix defined by

$$X^{n}_{1,m} = \begin{bmatrix} x_m & x_{m+1} & \cdots & x_n \\ \vdots & \vdots & & \vdots \\ x_1 & x_2 & \cdots & x_{n-m+1} \end{bmatrix}^{T} \tag{5.3.1b}$$

with the subscript m designating the number of columns of matrix $X^{n}_{1,m}$, 1 is the smallest and n the largest index of the observation data to form the matrix $X^{n}_{1,m}$. In a similar manner, matrix $Y^{n}_{1,m}$ is obtained by

$$Y^{n}_{1,m} = \begin{bmatrix} y_m & y_{m+1} & \cdots & y_n \\ \vdots & \vdots & & \vdots \\ y_1 & y_2 & \cdots & y_{n-m+1} \end{bmatrix}^{T} \tag{5.3.1c}$$

where the entries of the matrix $Y^{n}_{1,m}$ are given by

$$y_i = x^{*}_{i-q} \qquad \text{for} \quad i = 1, \ldots, n \tag{5.3.1d}$$

This particular representation has been chosen so that in the development of the generalized Levinson's algorithm for an ARMA model, notational complexity can be eased. It then follows that the matrix expressed by (5.3.1a) has the following shift invariance structures which characterizes the near Toeplitz structure of the matrix $R^{n}_{1,m}$, that is

$$R^n_{1,m} = R^{n-1}_{1,m} + \underline{y}^n_m \, (\underline{x}^n_m)^T \tag{5.3.2a}$$

$$= R^n_{2,m} + \underline{y}^m_m \, (\underline{x}^m_m)^T \tag{5.3.2b}$$

$$R^n_{1,m+1} = \begin{bmatrix} & (\underline{w}^n_{m+1})^T & \\ \hdashline x & & \\ \vdots & R^{n-1}_{1,m} & \\ x & & \end{bmatrix} \tag{5.3.2c}$$

$$= \begin{bmatrix} & & x \\ & R^n_{2,m} & \vdots \\ & & x \\ \hdashline (z^n_{m+1})^T & & \end{bmatrix} \tag{5.3.2d}$$

where $\underline{y}^n_m$, $\underline{x}^n_m$, $\underline{y}^n_m$ and $\underline{x}^m_m$ are m x 1 column vectors defined by

$$\underline{y}^n_m = [y_n, \ldots, y_{n-m+1}]^T \tag{5.3.2e}$$

$$\underline{x}^n_m = [x_n, \ldots, x_{n-m+1}]^T \tag{5.3.2f}$$

$$\underline{y}^m_m = [y_m, \ldots, y_1]^T \tag{5.3.2g}$$

$$\underline{x}^m_m = [x_m, \ldots, x_1]^T \tag{5.3.2h}$$

while $(z^n_{m+1})^T$ and $(\underline{w}^n_{m+1})^T$ denote the first and last rows of the matrix $R^n_{1,m+1}$ respectively, and the m x m matrices $R^n_{2,m}$ is defined by

$$R^n_{2,m} = [Y^n_{2,m}]^T \, X^n_{2,m} \tag{5.3.3}$$

From a structural viewpoint, relationship (5.3.2a) is called time update, since the matrix $R_{l,m}^{n}$ is explicitly defined as a sum of two matrices: $R_{l,m}^{n-1}$ which includes all of the past observations up to previous time index n-1 and $\underline{y}_m^n(\underline{x}_m^n)^T$ which includes the most recent data. In expression (5.3.2c), $R_{l,m}^{n-1}$ is seen to be a submatrix of $R_{l,m+1}^{n}$. It then follows that relationship (5.3.2c) is called order and time update.

A computationally efficient algorithm will be obtained by using various combinations of the above shift invariance structures. This fast algorithmic procedure for finding the solution is similar to Levinson's algorithm (Levinson, 1947). The overall solution is updated from the solution of a lower order to that of higher order system of equations (order update) and from the solution of previous time instance to that of present time (time update). To develop this algorithm, we apply an induction hypothesis. Suppose at order m and time n, we have the relationship

$$R_{l,m}^{n}\,[\underline{a}_{l,m}^{n}\;\;\underline{b}_{l,m}^{n}\;\;\underline{d}_{l,m}^{n}] = \begin{bmatrix} \zeta_{l,m}^{\varepsilon,n} & 0 & y_m \\ 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & 0 & \cdot \\ \cdot & & \\ 0 & \zeta_{l,m}^{r,n} & y_1 \end{bmatrix} \qquad (5.3.4a)$$

where $\underline{a}_{l,m}^{n}$, $\underline{b}_{l,m}^{n}$, and $\underline{d}_{l,m}^{n}$ are m x 1 column vectors defined by

$$\underline{a}_{l,m}^{n} = [1,\; a_{l,m}^{n}(1),\; \ldots,\; a_{l,m}^{n}(m-1)]^T \qquad (5.3.4b)$$

$$\underline{b}_{l,m}^{n} = [b_{l,m}^{n}(n-1),\; b_{l,m}^{n}(n-2),\; \ldots,\; b_{l,m}^{n}(1),\; 1]^T \qquad (5.3.4c)$$

and

$$\underline{d}^{n}_{1,m} = [d^{n}_{1,m}(1),\ d^{n}_{1,m}(2),\ \ldots,\ d^{n}_{1,m}(m)]^{T} \tag{5.3.4d}$$

Specifically, $\underline{a}^{n}_{1,m}$ and $\underline{b}^{n}_{1,m}$ are called the forward and the backward AR coefficient vectors, respectively. In the development of the computationally efficient algorithm, the auxiliary vectors $\underline{d}^{n}_{1,m}$ are needed to cancel the end effects due to the non-Toeplitz nature of matrix $R^{n}_{1,m}$. At the previous time index n-1, we have the relationship

$$R^{n-1}_{1,m}\,[a^{n-1}_{1,m} \quad \underline{b}^{n-1}_{1,m} \quad \underline{d}^{n-1}_{1,m}] = \begin{bmatrix} \zeta^{\varepsilon,n-1}_{1,m} & 0 & y_m \\ 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & 0 & \cdot \\ \cdot & & \cdot \\ 0 & \zeta^{r,n-1}_{1,m} & y_1 \end{bmatrix} \tag{5.3.4e}$$

Based on the relationship (5.3.4a) and (5.3.4e), we will develop a recursive solution procedure for the vectors $\underline{a}^{n}_{1,m+1}$, $b^{n}_{1,m+1}$ and $\underline{d}^{n}_{1,m+1}$ as a function of n. Applying the shift invariance structure (5.3.2b) to (5.3.4a) yields the following expressions

$$R^{n}_{2,m}\ \underline{a}^{n}_{1,m} = \zeta^{\varepsilon,n}_{1,m}\,\underline{e}_1 - \varepsilon_m\,\underline{y}^{m} \tag{5.3.5a}$$

$$R^{n}_{2,m}\,\underline{d}^{n}_{1,m} = (1 - f_m)\,\underline{y}_m \tag{5.3.5b}$$

where ε_m and f_m are scalars defined by

$$\varepsilon_m = (\underline{x}^{m})^{T}\,\underline{a}^{n}_{1,m} \tag{5.3.6a}$$

$$f_m = (\underline{x}^{m})^{T}\,\underline{d}^{n}_{1,m} \tag{5.3.6b}$$

and $\underline{e}_1$ is the m x 1 unit basis vector expressed by

$$\underline{e}_1 = [1,\ 0\ .\ .\ .\ 0]^T \tag{5.3.6c}$$

Expressions (5.3.5a) and (5.3.5b) lead to the following relationship

$$R^m_{2,m}\ \ \underline{a}^n_{2,m} = \zeta^n_{2,m}\ \underline{e}_1 \tag{5.3.7a}$$

where $\underline{a}^n_{2,m}$ and $\zeta^n_{2,m}$ are a m x 1 column vector and a scalar respectively, defined by

$$\underline{a}^n_{2,m} = [\underline{a}^n_{1,m} + \frac{\varepsilon_m}{1 - f_m}\ \underline{d}^n_{1,m}\ ]/\{1 + \frac{\varepsilon_m}{1 - f_m}\ d^n_{1,m}(1)\} \tag{5.3.7b}$$

$$\zeta^{\varepsilon,n}_{2,m} = \zeta^{\varepsilon,n}_{1,m} \ /\ \{1 + \frac{\varepsilon_m}{1 - f_m}\ d^n_{1,m}(1)\} \tag{5.3.7c}$$

where $d^n_{1,m}(1)$ denotes the first entry of the vector $\underline{d}^n_{1,m}$ (see eq. (5.3.4d)). Expressions (5.3.2c) and (5.3.2d) lead to the relationships

$$R^n_{1,m+1}\ \underline{a}^n_{1,m+1} = \zeta^{\varepsilon,n}_{1,m+1}\ \underline{e}_1 \tag{5.3.8a}$$

$$R^n_{1,m+1}\ b^n_{1,m+1} = \zeta^{r,n}_{1,m+1}\ \underline{e}_{m+1} \tag{5.3.8b}$$

where $\underline{e}_{m+1}$ is the (m+1)x1 unit bases vector defined by

$$\underline{e}_{m+1} = [0,\ .\ .\ .\ ,\ 1]^T \tag{5.3.8c}$$

In expressions (5.3.8a) and (5.3.8b), $\zeta^{\varepsilon,n}_{1,m+1}$ and $\zeta^{r,n}_{1,m+1}$ are scalars defined by

$$\zeta^{\varepsilon,n}_{1,m+1} = \zeta^{\varepsilon,n}_{2,m} - \frac{\alpha_m \beta_m}{\zeta^{r,n-1}_{1,m}} \tag{5.3.9a}$$

$$\zeta^{r,n}_{1,m+1} = \zeta^{r,n-1}_{1,m} - \frac{\alpha_m \beta_m}{\zeta^{\varepsilon,n}_{2,m}} \tag{5.3.9b}$$

in which α_m and β_m are scalars specified by

$$\alpha_m = (\underline{z}^n_{m+1})^T \begin{bmatrix} \underline{a}^n_{2,m} \\ 0 \end{bmatrix} \tag{5.3.10a}$$

$$\beta_m = (\underline{w}^n_{m+1})^T \begin{bmatrix} 0 \\ \underline{b}^{n-1}_{1,m} \end{bmatrix} \tag{5.3.10b}$$

The m x 1 column vectors $\underline{a}^n_{1,m+1}$ and $\underline{b}^n_{1,m+1}$ in expressions (5.3.8a) and (5.3.8b), respectively, are defined by

$$\underline{a}^n_{1,m+1} = \begin{bmatrix} \underline{a}^m_{2,m} \\ 0 \end{bmatrix} - \frac{\alpha_m}{\zeta^{r,n-1}_{1,m}} \begin{bmatrix} 0 \\ \underline{b}^{n-1}_{1,m} \end{bmatrix} \tag{5.3.11a}$$

$$\underline{b}^n_{1,m+1} = \begin{bmatrix} 0 \\ \underline{b}^{n-1}_{1,m} \end{bmatrix} - \frac{\beta_m}{\zeta^{\varepsilon,n}_{2,m}} \begin{bmatrix} \underline{a}^n_{2,m} \\ 0 \end{bmatrix} \tag{5.3.11b}$$

Expressions (5.3.11a) and (5.3.11b) are seen to be very similar to Levinson's algorithm (Levinson, 1947). In fact, one can show that these two expressions can be converted to Levinson's algorithm, if the pre- and post-modification method is applied on the matrix $R^n_{1,m}$.

Next, we will verify the relationship which updates the vector $\underline{d}^n_{1,m}$. The m x 1 column vector $\underline{d}^n_{1,m+1}$ is found to be

$$\underline{d}^{n}_{1,m+1} = \begin{bmatrix} 0 \\ \underline{d}^{n}_{1,m} \end{bmatrix} + \frac{y_{m+1} - \gamma_m}{\zeta^{\varepsilon,n}_{1,m+1}} \underline{a}^{n}_{1,m+1} \tag{5.3.12a}$$

where γ_m is a scalar given by

$$\gamma_m = (\underline{w}^{n}_{m+1})^T \begin{bmatrix} 0 \\ \underline{d}^{n}_{1,m} \end{bmatrix} \tag{5.3.12b}$$

It can be straightforwardly shown that

$$R^{n}_{1,m+1} \underline{d}^{n}_{1,m+1} = \underline{y}^{m+1}_{m+1} \tag{5.3.13}$$

Finally, combining expressions (5.3.11a), (5.3.11b), and (5.3.13), the following relationship is obtained.

$$R^{n}_{1,m+1} \left[\underline{a}^{n}_{1,m+1} \quad \underline{b}^{n}_{1,m+1} \quad \underline{d}^{n}_{1,m+1}\right] = \begin{bmatrix} \zeta^{\varepsilon,n}_{1,m+1} & 0 & y_{m+1} \\ 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & 0 & \cdot \\ \cdot & & \cdot \\ 0 & \zeta^{r}_{1,m+1} & y_1 \end{bmatrix} \tag{5.3.14}$$

In the above development, the generalized Levinson's algorithm for ARMA model is verified based on the induction hypothesis. The number of computation of the algorithm is readily found to be $0(3p^2)$ where p designates the number of denominator coefficients of the ARMA model.

We will now detail steps of the computations required in this recursive algorithm. The algorithm starts with the initialization procedure at n = q+1 and order m = 1. The solution $\underline{a}^{N}_{p+1}$ of the matrix

equation (5.3.4a) with m = p+1 is obtained by recursively updating $\underline{a}_m^n$ from m = 1 to p + 1 (order update) and from n = q+1 to N (time update). Meanwhile, auxiliary vectors $\underline{d}_m^n$ is also recursively updated. The above algorithm can be presented as follows

Step 1: Initialization for time update (n = q+1)

$$[\underline{w}_{1,1}^1] = [\underline{z}_{1,1}^1]^T = \zeta_{1,1}^{r,1} = \zeta_{1,1}^{\varepsilon,1} = y_{q+1}\, x_{q+1}$$

$$\underline{a}_{1,1}^1 = \underline{b}_{1,1}^1 = 1\ ,\ d_{1,1}^1 = y_1/\zeta_{1,1}^{\varepsilon,1}$$

Step 2: n = n+1

Step 3: Initialization for order update

$$\underline{w}_m^n = \underline{w}_m^{n-1} + y_n\, x_{n+1-m} \qquad m = 1, \ldots, M$$

$$\underline{z}_m^n = \underline{z}_m^{n-1} + y_{n+1-M}\, x_{n+1-m} \qquad m = 1, \ldots, M$$

where M = min (p+1, n-q) $\qquad$ m = 1, ... , M

$$\zeta_{1,m}^{\varepsilon,n} = \zeta_{1,m}^{r,m} = \underline{z}_m^n\,(1)$$

where $\underline{z}_m^n(1)$ denotes the first element of column vector $\underline{z}_m^n$

$$a_{1,1}^n = b_{1,1}^n = 1 \qquad d_{1,1}^n = y_1/\zeta_{1,m}^{\varepsilon,n}$$

Step 4: Compute recursively from m = 1 to M where M = min (p, n-q-1)

$$\varepsilon_m = (\underline{x}_m^m)^T\, \underline{a}_{1,m}^n,\ f_m = (\underline{x}_m^m)^T\, \underline{d}_{1,m}^n$$

$$\underline{a}^{n}_{2,m} = [\underline{a}^{n}_{1,m} + \frac{\varepsilon_m}{1 - f_m}\,\underline{d}^{n}_{1,m}]/\{1 + \frac{\varepsilon_m}{1 - f_m}\,\underline{d}^{n}_{1,m}(1)\}$$

$$\zeta^{\varepsilon,n}_{2,m} = \zeta^{\varepsilon,n}_{1,m}/\{1 + \frac{\varepsilon_m}{1 - f_m}\,\underline{d}^{n}_{1,m}(1)\}$$

$$\alpha_m = (\underline{z}^{n}_{m+1})^T \begin{bmatrix} \underline{a}^{n}_{2,m} \\ 0 \end{bmatrix} \qquad \beta_m = (\underline{w}^{n}_{m+1})^T \begin{bmatrix} 0 \\ \underline{b}^{n-1}_{1,m} \end{bmatrix}$$

· Update forward and backward solutions

$$\underline{a}^{n}_{1,m+1} = \begin{bmatrix} \underline{a}^{n}_{2,m} \\ 0 \end{bmatrix} - \frac{\alpha_m}{\zeta^{r,n-1}_{1,m}} \begin{bmatrix} 0 \\ \underline{b}^{n-1}_{1,m} \end{bmatrix}$$

$$\underline{b}^{n}_{1,m+1} = \begin{bmatrix} 0 \\ \underline{b}^{n-1}_{1,m} \end{bmatrix} - \frac{\beta_m}{\zeta^{\varepsilon,n}_{2,m}} \begin{bmatrix} \underline{a}^{n}_{2,n} \\ 0 \end{bmatrix}$$

$$\zeta^{\varepsilon,n}_{1,m+1} = \zeta^{\varepsilon,n}_{2,m} - \frac{\alpha_m \beta_m}{\zeta^{r,n-1}_{1,m}}$$

$$\zeta^{r,n}_{1,m+1} = \zeta^{r,n-1}_{1,m} - \frac{\alpha_m \beta_m}{\zeta^{\varepsilon,n}_{2,m}}$$

· Compute auxiliary vector $\underline{d}^{n}_{1,m}$

$$\gamma_m = (\underline{w}^{n}_{m+1})^T \begin{bmatrix} 0 \\ \underline{d}^{n}_{1,m} \end{bmatrix}$$

$$\underline{d}^{n}_{1,m+1} = \begin{bmatrix} 0 \\ \underline{d}^{n}_{1,m} \end{bmatrix} + \frac{y_{m+1} - \gamma_m}{\zeta^{\varepsilon,n}_{1,m+1}}\,\underline{a}^{n}_{1,m+1}$$

Step 5: If n < N go to Step 2

Step 6: End of algorithm

In above, N is taken to be the last index of vector $\underline{x}_N$.

5.4 ARMA Doubling Algorithm: The Pre- and Post-Method

As described in section 5.2, one of the data modification methods referred to as the pre- and post-modification method leads to the following set of equations

$$A_p \underline{a} = \underline{b}_p \tag{5.4.1}$$

where A_p is a pxp Toeplitz matrix and $\underline{b}_p$ is a pxl column vector given by

$$A_p = Y_3^T X_3 \tag{5.4.2}$$

$$\underline{b}_p = Y_3^T \underline{x}_3$$

where matrices Y_3, X_3 and column vector $\underline{x}_3$ are previously defined in expressions (5.2.3.1a), (5.2.3.1b) and (5.2.3.1c), respectively. The displacement rank of the matrix A_p is readily shown to be 2. This being the case, it is possible to apply the doubling algorithm (Morf, 1980; Bitmead and Anderson, 1979).

Without loss of generality, we now assume that $p = 2^k$ for some integer k. The matrix A_p can be partitioned into 4 matrices whose sizes are $2^{(k-1)} \times 2^{(k-1)}$. Each $2^{(k-1)} \times 2^{(k-1)}$ matrix is then also

partitioned into $2^{(k-2)} \times 2^{(k-2)}$ matrices. This procedure is called the doubling or halving procedure. In this procedure, we can express a $2\ell \times 2\ell$ matrix $A_{2\ell}$ in terms of $\ell \times \ell$ submatrices B_ℓ, C_ℓ, D_ℓ and E_ℓ in the following manner

$$A_{2\ell} = \begin{bmatrix} B_\ell & C_\ell \\ D_\ell & E_\ell \end{bmatrix} \tag{5.4.4}$$

and its inverse is found to be the form

$$A_{2\ell}^{-1} = \begin{bmatrix} S_\ell & T_\ell \\ U_\ell & V_\ell \end{bmatrix} \tag{5.4.5}$$

where S_ℓ, T_ℓ, U_ℓ and V_ℓ are $\ell \times \ell$ square matrices given by

$$S_\ell = B_\ell^{-1} + B_\ell^{-1} C_\ell V_\ell D_\ell B_\ell^{-1} \tag{5.4.6a}$$

$$T_\ell = -S_\ell C_\ell E_\ell^{-1} \tag{5.4.6b}$$

$$U_\ell = -E_\ell^{-1} D_\ell S_\ell \tag{5.4.6c}$$

$$V_\ell = E_\ell^{-1} + E_\ell^{-1} D_\ell S_\ell C_\ell E_\ell^{-1} \tag{5.4.6d}$$

Relationships (5.4.6a) - (5.4.6d) are straightforwardly derived from the Schur complements theorem (Aho, etc., 1974). From the above relationships, we can obtain $A_{2\ell}^{-1}$ from B_ℓ^{-1} and E_ℓ^{-1}. The solution of the equation (5.4.1) requires $O(2\ c(m))$ computations where $c(m)$ is the number of operations required to multiply a vector times a triangular Toeplitz matrix.

The number of computation c(m) is obtained in a following manner. By definition (Kailath, etc., 1979), $A_{2\ell}$ can be decomposed in the form

$$A_{2\ell} = \sum_{i=1}^{2} L_{2\ell}^{i} \, U_{2\ell}^{i} \tag{5.4.7}$$

where $L_{2\ell}^{i}$ and $U_{2\ell}^{i}$ are lower and upper triangular Toeplitz matrices, respectively, which can be obtained recursively (Bitmead and B. Anderson, 1979). The matrices $L_{2\ell}^{i}$ and $U_{2\ell}^{i}$ are expressed by

$$L_{2\ell}^{i} = \begin{bmatrix} L_{\ell}^{i}(1,1) & \bigcirc \\ L_{\ell}^{i}(2,1) & L_{\ell}^{i}(2,2) \end{bmatrix} \tag{5.4.8a}$$

$$U_{2\ell}^{i} = \begin{bmatrix} U_{\ell}^{i}(1,1) & U_{\ell}^{i}(1,2) \\ \bigcirc & U_{\ell}^{i}(2,2) \end{bmatrix} \tag{5.4.8b}$$

where $L_{\ell}^{i}(1,1)$ and $L_{\ell}^{i}(2,2)$ are ℓ x ℓ lower triangular Toeplitz matrices, $U_{\ell}^{i}(1,1)$ and $U_{\ell}^{i}(2,2)$ are ℓ x ℓ upper triangular Toeplitz matrices, and $L_{\ell}^{i}(2,1)$ and $U_{\ell}^{i}(1,2)$ are ℓ x ℓ full Toeplitz matrices. Substitution of expressions (5.4.8a) and (5.4.8b) into (5.4.7) yields the partitions of the matrix $A_{2\ell}$ in expressions (5.4.4) to be

$$B_{\ell} = \sum_{i=1}^{2} L_{\ell}^{i}(1,1) \;\; U_{\ell}^{i}(1,1) \tag{5.4.9a}$$

$$C_{\ell} = \sum_{i=1}^{2} L_{\ell}^{i}(1,1) \, [U_{\ell}^{i}(1,2)]_{L} + \sum_{i=1}^{2} L_{\ell}^{i}(1,1) \, [U_{\ell}^{i}(1,2)]_{U} \tag{5.4.9b}$$

$$D_{\ell} = \sum_{i=1}^{2} [L_{\ell}^{i}(2,1)]_{L} \, U_{\ell}^{i}(1,1) + \sum_{i=1}^{2} [L_{\ell}^{i}(2,1)]_{U} \, U_{\ell}^{i}(1,1) \tag{5.4.9c}$$

$$E_\ell = \sum_{i=1}^{2} L_\ell^i(2,1)\, U_\ell^i(1,2) + \sum_{i=1}^{2} L_\ell^i(2,2)\, U_\ell^i(2,2) \qquad (5.4.9d)$$

where the following relationships are implicitly used

$$U_\ell^i(1,2) = [U_\ell^i(1,2)]_L + [U_\ell^i(1,2)]_U \qquad (5.4.10a)$$

$$L_\ell^i(2,1) = [L_\ell^i(2,1)]_L + [L_\ell^i(2,1)]_U \qquad (5.4.10b)$$

in which $[U_\ell^i(1,2)]_L$ and $[L_\ell^i(2,1)]_L$ denote ℓ x ℓ lower triangular matrices, and $[U_\ell^i(1,2)]_U$ and $[L_\ell^i(2,1)]_U$ denote ℓ x ℓ upper triangular matrices. The partitions given by equations (5.4.9a) - (5.4.9d) are expressed in terms of lower triangular and upper triangular matrices. It turns out that the use of above relationships reduces the computational complexity c(m) to be O(p log p) (Morf, 1980).

The algorithm which makes use of the doubling method can be found in (Morf, 1980). Morf described the algorithm by introducing high computer language which necessitates frequent subroutine calls. On the other hand, the step-wise description of the halving method is presented in (Bitmead and Anderson, 1979). Implementation of the halving algorithm is relatively complex and a rather large value of p is required before the computationally complexity O(p log p) is approached.

5.5 Numerical Example

In this section, the spectral performance of the pre- and the post-modified methods are compared with the unmodified method. As

a test example, we treat the time series (4.5.2a). Using this time series (4.5.2a), twenty different independent sampled sequences each of length 64 were generated.

The modification methods were applied to these twenty different sampled sequences of length 64 to obtain ARMA model spectral estimates or order (4,4). The resultant spectra are shown in Fig. 5.5.1. It is apparent that only a small degradation in spectral estimation performance has been shown by the modified method. It might be conjectured that the implementation of the fast algorithms will not much degrade spectral performance in many practical examples.

5.6 Summary

In this chapter, computationally efficient ARMA spectral estimation algorithms have been developed. These algorithms are predicated on the utilization of data modification methods. Specifically, two algorithms referred as the generalized Levinson's algorithm and the doubling algorithm were developed for obtaining AR coefficients of ARMA model. These algorithms have a computational complexity of $0(p^2)$ and $0(p \log p)$, respectively.

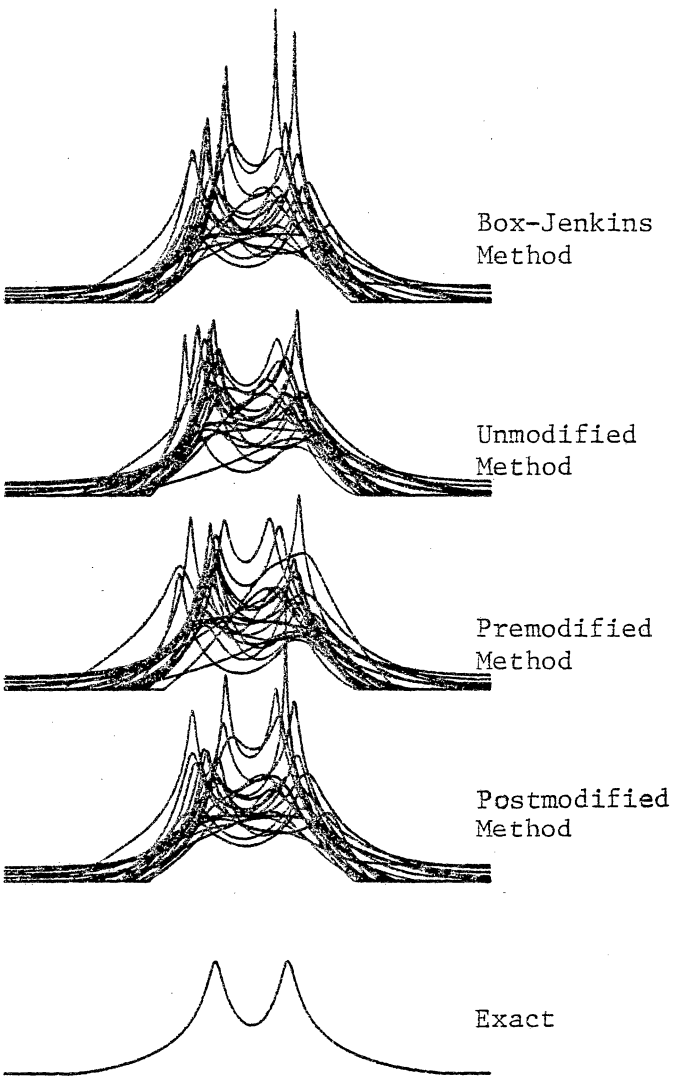


Fig. 5.5.1 ARMA Spectral Estimates of Order (4,4), Data Length 64, and, $\lambda = 0.95$.

Chapter 6

A RECURSIVE ARMA SPECTRAL ESTIMATOR: THE PREMODIFIED METHOD

6.1 Introduction

A recursive ARMA spectral estimation procedure is developed in this section. It is recursive in the sense that as a new element of the time series is observed, the parameters of a spectral estimation model are algorithmically updated. The recursive algorithm requires $0(p)$ computations to update the model's parameters for each new data point. The development of this algorithm is predicated on utilization of certain projection operators. In Section (6.2), a vector space is formulated by making use of the given observation data. The method of linear predictions will give rise to projection operators which decompose relevant vector spaces into subspaces spanned by the prediction error vector and the observation vectors. Linear prediction methods used in this chapter include forward prediction, backward prediction and delayed backward prediction. Each of these methods is associated with its own projection operator. The decomposition of these projection operators is discussed in Section (6.4). The order update and time update recursions, as described in Sections (6.5) and (6.6) play a central role in the overall recursive algorithm. Finally in Section (6.7), a recursive algorithm is outlined.

6.2 Vector Space Formulation

In this section, the given spectral estimation problem will be cast into a convenient vector space setting. It will be assumed that the following observations of the time series $\{x(n)\}$

$$x_1, x_2, \ldots, x_N \tag{6.2.1}$$

are given. This in turn will give rise to the associated column data vector

$$\underline{x}_N = [x_1, x_2, \ldots, x_N]^T \tag{6.2.2}$$

It is convenient to form an auxiliary column vector $\underline{y}_N$ specified by

$$\underline{y}_N = S^q \underline{x}_N \tag{6.2.3a}$$

$$= [0 \ldots 0 \; x_1 \ldots x_{N-q}]^T \tag{6.2.3b}$$

where S denotes the NxN down shift matrix (see eq. (5.1.3)) and q is the numerator order of the ARMA model. The vectors $\underline{x}_N$ and $\underline{y}_N$ lie in the product space

$$R^N = R \times R \times \ldots \times R \tag{6.2.4}$$

We next construct the subspace which is spanned by the set of vectors $S^i \underline{x}_N, \ldots, S^m \underline{x}_N$. This subspace will be suggestively denoted by

$$M \underline{x}_{N[i,m]} = \{S^i \underline{x}_N, S^{i+1} \underline{x}_N, \ldots, S^m \underline{x}_N\} \tag{6.2.5}$$

where the first integer i may take on any value in the set {0, 1, ... , m}. As will be described in Section (6.3), the recursive algorithm is derived for particular selections of indices i and m. Similarly, for the vector $\underline{y}_N$ contained in the product space R^N, the associated subspace $M\ \underline{y}_{N[i,m]}$ is defined by

$$M\ \underline{y}_{N[i,m]} = \{S^i\ \underline{y}_N,\ S^{i+1}\ \underline{y}_N,\ \ldots\ ,\ S^m\ \underline{y}_N\} \tag{6.2.6}$$

where the first integer i may take on any value in the set {0, 1, ... , m}. Next, we let $P\ \underline{x}_{N[i,m]}$ designate the projection operator on the subspace $M\ \underline{x}_{N[i,m]}$ along the subspace orthogonal to $M\ \underline{y}_{N[i,m]}$ (this orthogonal subspace will be denoted by $M^{\perp}\underline{y}_{N[i,m]}$). This projection operator which depends on $\underline{x}_N$ and $\underline{y}_N$ can be shown to have the form

$$P\ \underline{x}_{N[i,m]} \triangleq A\ \underline{x}_{N[i,m]}\ [A^{\dagger}\underline{y}_{N[i,m]}\ A\ \underline{x}_{N[i,m]}]^{-1}\ \ A^{\dagger}\underline{y}_{N[i,m]} \tag{6.2.7}$$

where $A\ \underline{x}_{N[i,m]}$ and $A\ \underline{y}_{N[i,m]}$ are the N x(m-i+1) matrices composed of the following ordered set of column vectors

$$A\ \underline{x}_{N[i,m]} = [S^i\ \underline{x}_N\ \ S^{i+1}\ \underline{x}_N\ \ \ldots\ \ S^m\ \underline{x}_N] \tag{6.2.8}$$

$$A\ \underline{y}_{N[i,m]} = [S^i\ \underline{y}_N\ \ S^{i+1}\ \underline{y}_N\ \ \ldots\ \ S^m\ \underline{y}_N] \tag{6.2.9}$$

The projection characteristics of operator (6.2.7) are depicted in Fig. 6.2.1. It will be convenient to introduce a projection operator on the complement of subspace $M\ \underline{x}_{N[i,m]}$. This operator is defined by

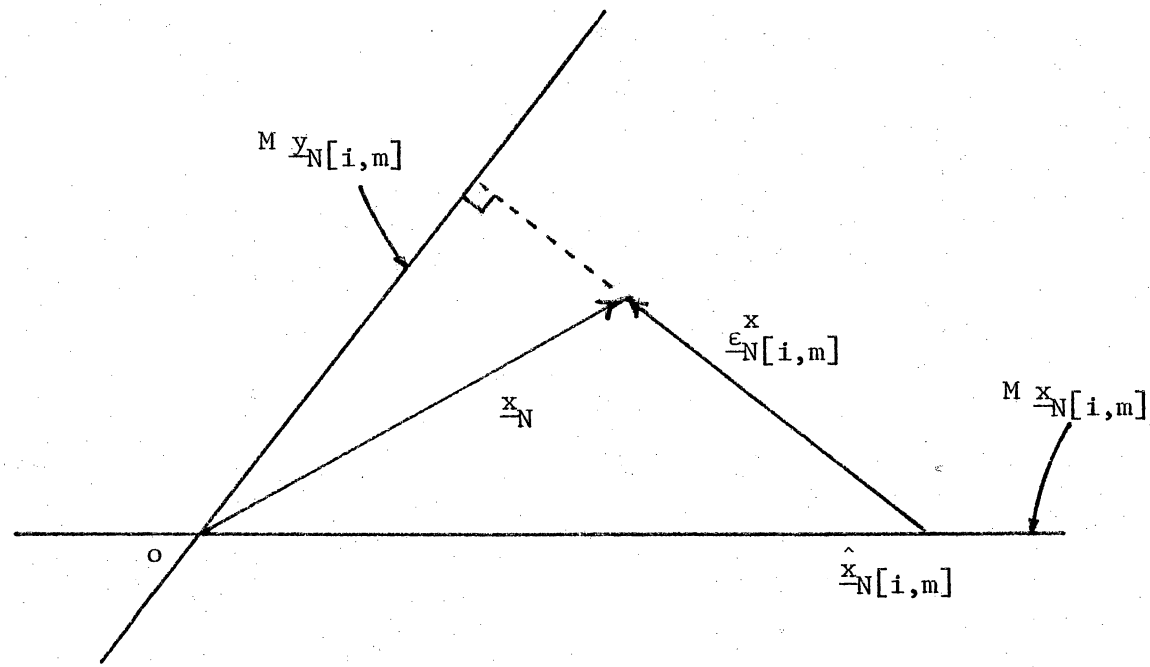


Fig. 6.2.1 Projection Operators on M $\underline{x}_{N[i,m]}$ along $M^{\perp}$ $\underline{y}_{N[i,m]}$

$$P^{c}\,\underline{x}_{N[i,m]} = I - P\,\underline{x}_{N[i,m]} \tag{6.2.10}$$

where I is the NxN identity matrix. In a similar fashion, the projection operator on the subspace $M\,\underline{y}_{N[i,m]}$ along the subspace $M^{\perp}\,\underline{x}_{N[i,m]}$ is specified by

$$P\,\underline{y}_{N[i,m]} \triangleq A\,\underline{y}_{N[i,m]}\,[A^{\dagger}\underline{x}_{N[i,m]}\,A\,\underline{y}_{N[i,m]}]^{-1}\,A^{\dagger}\underline{x}_{N[i,m]} \tag{6.2.11a}$$

It is to be noted that the following projection operator identity holds as is apparent from expressions (6.2.7) and (6.2.11a).

$$P\,\underline{y}_{N[i,m]} = P^{\dagger}\underline{x}_{N[i,m]} \tag{6.2.11b}$$

The complement of the projection operator (6.2.11b) is formally given by

$$P^{c}\underline{y}_{N[i,m]} = I - P\,\underline{y}_{N[i,m]} \tag{6.2.12}$$

A particular estimate $\hat{\underline{x}}_{N[i,m]}$ of the vector $\underline{x}_N$ can be specified as the projection of $\underline{x}_N$ on the subspace $M\,\underline{x}_{N[i,m]}$, that is

$$\hat{\underline{x}}_{N[i,m]} = P\,\underline{x}_{N[i,m]}\,\underline{x}_N \tag{6.2.13}$$

The error vector relative to estimate $\hat{\underline{x}}_{N[i,m]}$ and $\underline{x}_N$ is then given by

$$\underline{\varepsilon}^{x}_{N[i,m]} = \underline{x}_N - \hat{\underline{x}}_{N[i,m]} \tag{6.2.14a}$$

$$= P^{c}\underline{x}_{N[i,m]}\,\underline{x}_N \tag{6.2.14b}$$

which is expressed as a projection of the vector $\underline{x}_N$ on the complement

subspace of $M\,\underline{x}_{N[i,m]}$. It can be straightforwardly shown that

$$\underline{\varepsilon}^{x}_{N[i,m]} \perp M\,\underline{y}_{N[i,m]} \tag{6.2.15}$$

where $\perp$ denotes orthogonality, that is, the error vector $\underline{\varepsilon}^{x}_{N[i,m]}$ is orthogonal to the subspace $M\,\underline{y}_{N[i,m]}$. The vector space formulation described in this section is suggestively depicted in Fig. 6.2.1.

6.3 Linear Prediction and Projection Operator

In this section we will define three methods of linear predictions, namely, forward prediction, backward prediction, and delayed backward prediction. These methods will play a central role in the algorithmic solution procedure to be developed.

6.3.1 Forward Prediction

The m-th order forward prediction is referred to as that specific procedure for estimating the column vector $\underline{x}_N$ and $\underline{y}_N$ by means of a linear combination of the set of m shifted vectors $\{S^1\underline{x}_N, S^2\underline{x}_N, \ldots, S^m\underline{x}_N\}$ and $\{S^1\underline{y}_N, S^2\underline{y}_N, \ldots, S^m\underline{y}_N\}$, respectively. Considering the projection operator defined in Section 6.2, the associated estimates $\hat{\underline{x}}_{N[1,m]}$ and $\hat{\underline{y}}_{N[1,m]}$ are seen to have the form

$$\hat{\underline{x}}_{N[1,m]} = P\,\underline{x}_{N[1,m]}\,\underline{x}_N \tag{6.3.1a}$$

$$\hat{\underline{y}}_{N[1,m]} = P\,\underline{y}_{N[1,m]}\,\underline{y}_N \tag{6.3.1b}$$

The difference between the estimate $\hat{\underline{x}}_N$ and the given vector $\underline{x}_N$ is

called the forward prediction error vector and is specified by

$$\underline{\varepsilon}^{x}_{N,m} = \underline{x}_N - \hat{\underline{x}}_N[1,m] \tag{6.3.2a}$$

while the forward prediction error vector of $\underline{y}_N$ is of the form

$$\underline{\varepsilon}^{y}_{N,m} = \underline{y}_N - \hat{\underline{y}}_N[1,m] \tag{6.3.2b}$$

Now these error vectors are each orthogonal to the subspaces $M\ \underline{y}_N[1,m]$ and $M\ \underline{x}_N[1,m]$, respectively. Use of complement projection operators defined by (6.2.10) and (6.2.12) yields

$$\underline{\varepsilon}^{x}_{N,m} = P^{c}_{\underline{x}_N[1,m]}\ \underline{x}_N \tag{6.3.3a}$$

$$\underline{\varepsilon}^{y}_{N,m} = P^{c}_{\underline{y}_N[1,m]}\ \underline{y}_N \tag{6.3.3b}$$

6.3.2 Backward Prediction

The m-th order backward prediction is that procedure of estimating the column vector $S^m\underline{x}_N$ and $S^m\underline{y}_N$ by a linear combination of the set of m shifted vectors $\{S^0\underline{x}_N, S^1\underline{x}_N, \ldots, S^{m-1}\underline{x}_N\}$ and $\{S^0\underline{y}_N, S^1\underline{y}_N, \ldots, S^{m-1}\underline{y}_N\}$, respectively. In the same manner as with forward prediction, by applying the projection operator, it can be shown that the backward estimate is given by

$$\hat{\hat{\underline{x}}}_N[0,m-1] = P_{\underline{x}_N[0,m-1]}\ S^m\underline{x}_N \tag{6.3.4a}$$

where the double caret notation designates backward prediction. The backward prediction error vector is then found to be

$$\underline{b}^{x}_{N,m} = P^{c}_{\underline{x}_N[0,m-1]}\, S^{m}\underline{x}_N \tag{6.3.4b}$$

6.3.3 Delayed Backward Prediction

The m-th order delayed backward prediction is similarly defined to be that procedure of estimating the column vector $S^{m+1}\underline{x}_N$ and $S^{m+1}\underline{y}_N$ by a linear combination of the sets of vectors $\{S^{1}\underline{x}_N, S^{2}\underline{x}_N, \ldots, S^{m}\underline{x}_N\}$ and $\{S^{1}\underline{y}_N, S^{2}\underline{y}_N, \ldots, S^{m}\underline{y}_N\}$, respectively. It can be shown that the delayed backward prediction is given by

$$\hat{\underline{x}}_N[1,m] = P_{\underline{x}_N[1,m]}\, S^{m+1}\underline{x}_N \tag{6.3.5a}$$

while the delayed backward prediction error is specified by

$$\underline{d}^{x}_{N,m} = P^{c}_{\underline{x}_N[1,m]}\, S^{m+1}\underline{x}_N \tag{6.3.5b}$$

A little thought will convince oneself that the projection operators $P_{\underline{x}_N[1,m]}$ can be expressed as

$$P_{\underline{x}_N[1,m]} = A_{\underline{x}_N[1,m]}\left[A^{\dagger}\underline{y}_N[1,m]\, A_{\underline{x}_N[1,m]}\right]^{-1} A^{\dagger}\underline{y}_N[1,m]$$

$$= \begin{bmatrix} 0\ 0\ \ldots\ 0 \\ \hline A\,\underline{x}_{N-1}[0,m-1] \end{bmatrix} \left[A^{\dagger}\underline{y}_{N-1}[0,m-1]\, A\,\underline{x}_{N-1}[0,m-1] \right]^{-1}$$

$$\times \begin{bmatrix} 0\ 0\ \ldots\ 0 \\ \hline A\,\underline{y}_{N-1}[0,m-1] \end{bmatrix}^{\dagger} \tag{6.3.6}$$

This formula is straightforwardly obtained by making use of the structure of matrices $A\,\underline{x}_{N[1,m]}$ and $A\,\underline{y}_{N[1,m]}$ defined by (6.2.8) and (6.2.9). The relationship between the backward error and the delayed backward error is then readily found to be

$$\underline{d}^{x}_{N,m} = [0 \ , \ \underline{b}^{x}_{N-1,m}]^{T} \tag{6.3.7a}$$

$$\underline{d}^{y}_{N,m} = [0 \ , \ \underline{b}^{y}_{N-1,m}]^{T} \tag{6.3.7b}$$

It then follows that the N-th delayed error is equal to the (N-1)-st backward error, that is

$$\underline{d}^{x}_{N,m}(N) = \underline{b}^{x}_{N-1,m}(N-1) \tag{6.3.8a}$$

$$\underline{d}^{y}_{N,m}(N) = \underline{b}^{y}_{N-1,m}(N-1) \tag{6.3.8b}$$

The relationship between forward, backward, and delayed backward is suggestively depicted in Fig. 6.3.1.

6.4 Decomposition of Projection Operators

The development of a computationally efficient algorithm is dependent on the decomposition of the above projection operators. This decomposition makes use of the specific matrix structure referred to as shift invariancy. A matrix which has a displacement rank 3 will possess this shift invariancy (see Chapter 5). In this section, the shift invariant structure is utilized to decompose projection operators. The formulae obtained in this section will be used for

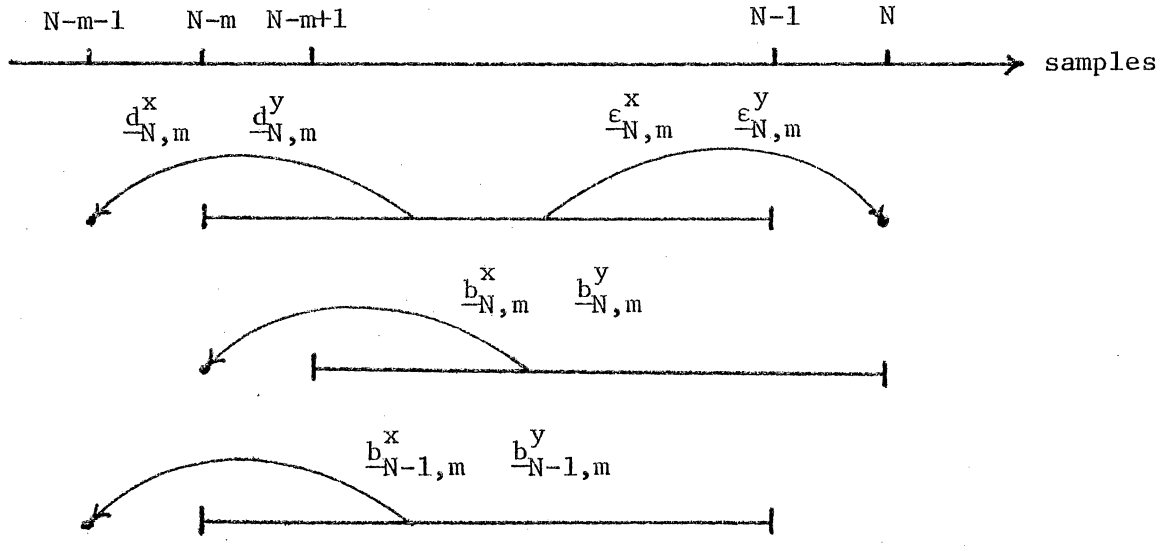


Fig. 6.3.1 Forward Backward and Delayed Backward Relationship

the development of order update recursions in section 6.5.

First, we will discuss the decomposition of the projection operator $P\ \underline{x}_N[0,m]$. This projection operator $P\ \underline{x}_N[0,m]$ may be expressed as

$$P\ \underline{x}_N[0,m] = A\ \underline{x}_N[0,m]\ [R\ \underline{x}_N[0,m]]^{-1}\ A^{\dagger} \underline{y}_N[0,m] \tag{6.4.1}$$

which is obtained by substituting i = 0 in expression (6.2.7). The matrix $R\ \underline{x}_N[0,m]$ is defined by

$$R\ \underline{x}_N[0,m] = A^{\dagger} \underline{y}_N[0,m]\ A\ \underline{x}_N[0,m] \tag{6.4.2}$$

Substitution of expressions (6.2.8) and (6.2.9) into (6.4.2) yields

$$R\ \underline{x}_N[0,m] = \left[\begin{array}{c|c} \underline{y}_N^{\dagger}\ \underline{x}_N & \underline{y}_N^{\dagger}\ A\ \underline{x}_N[1,m] \\ \hline A^{\dagger} \underline{y}_N[1,m]\ \underline{x}_N & R\ \underline{x}_N[1,m] \end{array}\right] \tag{6.4.3a}$$

where $R\ \underline{x}_N[1,m]$ is defined by substituting 1 in place of 0 in expression (6.4.2). If we denote the inverse of matrix $R\ \underline{x}_N[1,m]$ by $R^{-1} \underline{x}_N[1,m]$, it then follows that

$$R\ \underline{x}_N[0,m] \left[\begin{array}{c} 0 \ldots\ldots 0 \\ \cdot \\ \cdot\ R^{-1} \underline{x}_N[1,m] \\ \cdot \\ \cdot \\ 0 \end{array}\right] = \left[\begin{array}{c|c} 0 & \underline{y}_N^{\dagger}\ A\ \underline{x}_N[1,m]\ R^{-1} \underline{x}_N[1,m] \\ \hline 0 & \\ \cdot & \\ \cdot & \quad I \\ \cdot & \\ 0 & \end{array}\right] \tag{6.4.4}$$

where I denotes the m x m identify matrix. Upon examination of expression (6.4.3a) and (6.4.4), it can be readily shown that

$$\underline{u}_{m+1}^{\dagger} \; R \; \underline{x}_{N[0,m]} = \underline{e}_1^T \tag{6.4.5}$$

where $\underline{e}_1$ denotes the (m + 1) x 1 unit basis vector and $\underline{u}_{m+1}$ is a (m + 1) x 1 column vector given by*

$$\underline{u}_{m+1}^{\dagger} = [1, \; -\underline{y}_N^{\dagger} \; A \; \underline{x}_{N[1,m]} \; R^{-1} \underline{x}_{N[1,m]}]/f_{N,m}^{\varepsilon} \tag{6.4.6}$$

in which $f_{N,m}^{\varepsilon}$ is a scalar defined by

$$f_{N,m}^{\varepsilon} = \underline{y}^{\dagger} \; \underline{\varepsilon}_{N,m}^{x} = (\underline{\varepsilon}_{N,m}^{y})^{\dagger} \; \underline{\varepsilon}_{N,m}^{x} \tag{6.4.7}$$

In a similar fashion, let us define a matrix $R \; \underline{y}_{N[0,m]}$ by

$$R \; \underline{y}_{N[0,m]} = A^{\dagger} \underline{x}_{N[0,m]} \; A \; \underline{y}_{N[0,m]} \tag{6.4.8}$$

It then follows that

$$\underline{\nu}_{m+1}^{\dagger} \; R \; \underline{y}_{N[0,m]} = \underline{e}_1^T \tag{6.4.9}$$

where $\underline{\nu}_{m+1}$ is a column vector expressed by

$$\underline{\nu}_{m+1}^{\dagger} = [1, \; -\underline{x}_N^{\dagger} \; A \; \underline{y}_{N[1,m]} \; R^{-1} \underline{y}_{N[1,m]}]/ \; f_{N,m}^{\varepsilon} \; * \tag{6.4.9a}$$

Taking the complex conjugate vector transpose of expression (6.4.9), yields

---

*In general, $\underline{e}_k$ represents the standard unit basis vectors whose components are also zero except for its k-th which is one.

$$R\,\underline{x}_{N[0,m]}\,\underline{\nu}_{m+1} = \underline{e}_1 \tag{6.4.10}$$

The inverse of the matrix $R\,\underline{x}_{N[0,m]}$ is found to be

$$R^{-1}\underline{x}_{N[0,m]} = \begin{bmatrix} 0 & \cdots & 0 \\ \vdots & & \\ & R^{-1}\underline{x}_{N[1,m]} & \\ \vdots & & \\ 0 & & \end{bmatrix} + f^{\varepsilon}_{N,m}\,\underline{\nu}_{m+1}\,\underline{u}^{\dagger}_{m+1} \tag{6.4.11}$$

Substitution of expression (6.4.11) into (6.4.1) then leads to the following relationship.

$$P\,\underline{x}_{N[0,m]} = P\,\underline{x}_{N[1,m]} + A\,\underline{x}_{N[0,m]}\,f^{\varepsilon}_{N,m}\,\underline{\nu}_{m+1}\,\underline{u}^{\dagger}_{m+1}\,A^{\dagger}\underline{y}_{N[0,m]} \tag{6.4.12}$$

After a simple algebraic manipulation, the projection operator $P\,\underline{x}_{N[0,m]}$ is decomposed by the following relationships

$$P\,\underline{x}_{N[0,m]} = P\,\underline{x}_{N[1,m]} + P\,\underline{\varepsilon}^{x}_{N,m} \tag{6.4.13a}$$

$$= P\,\underline{x}_{N[1,m]} + (I - P\,\underline{x}_{N[1,m]})\,P\,\underline{\varepsilon}^{x}_{N,m} \tag{6.4.13b}$$

$$= P\,\underline{x}_{N[1,m]} + P\,\underline{\varepsilon}^{x}_{N,m}\,(I - P\,\underline{x}_{N[1,m]}) \tag{6.4.13c}$$

where it is readily shown that $P\,\underline{\varepsilon}^{x}_{N,m}$ is a projection operator onto the subspace spanned by $\underline{\varepsilon}^{x}_{N,m}$ along the subspace which is orthogonal to the subspace spanned by $\underline{\varepsilon}^{y}_{N,m}$ and is defined by

$$P\,\underline{\varepsilon}^{x}_{N,m} = \frac{1}{f^{\varepsilon}_{N,m}}\,\underline{\varepsilon}^{x}_{N,m}\,(\underline{\varepsilon}^{y}_{N,m})^{\dagger} \tag{6.4.14}$$

Furthermore, expression (6.4.13c) leads to the following relationship

$$I - P\ \underline{x}_N[0,m+1] = (I - P\ \underline{\varepsilon}^x_{N,m})(I - P\ \underline{x}_N[1,m]) \qquad (6.4.15)$$

The projection operator decomposition as expressed in (6.4.15) will be used to find a backward error recursion in the next section.

Next, we will decompose the projection operator $P\ \underline{x}_N[1,m+1]$ which is necessary to compute the forward prediction error. The projection operator $P\ \underline{x}_N[1,m+1]$ is given by

$$P\ \underline{x}_N[1,m+1] = A\ \underline{x}_N[1,m+1]\ [R\ \underline{x}_N[1,m+1]]^{-1}\ A^\dagger \underline{y}_N[1,m+1] \qquad (6.4.16)$$

which is obtained by substituting i = 1 in expression (6.2.7). The matrix $R\ \underline{x}_N[1,m+1]$ is defined by

$$R\ \underline{x}_N[1,m+1] = A^\dagger \underline{y}_N[1,m+1]\ A\ \underline{x}_N[1,m+1] \qquad (6.4.17)$$

Substitution of expressions (6.2.8) and (6.2.9) with i = 1 into (6.4.17) yields

$$R\ \underline{x}_N[1,m+1] = \left[\begin{array}{c|c} R\ \underline{x}_N[1,m] & A^\dagger \underline{y}_N[1,m]\ S^{m+1}\ \underline{x}_N \\ \hline (S^{m+1}\ \underline{y}_N)^\dagger\ A\ \underline{x}_N[1,m] & (S^{m+1}\ \underline{y}_N)^\dagger\ S^{m+1}\ \underline{x}_N \end{array}\right] \qquad (6.4.18)$$

It then follows

$$R\,\underline{x}_{N[1,m+1]}\begin{bmatrix} & & 0 \\ & R^{-1}\underline{x}_{N[1,m]} & \vdots \\ & & \\ 0 & \cdots\cdots & 0 \end{bmatrix} = \left[\begin{array}{ccc:c} & & & 0 \\ & & I & \vdots \\ & & & 0 \\ \hdashline (S^{m+1}\underline{y}_N)^{\dagger} A\,\underline{x}_{N[1,m]} R^{-1}\underline{x}_{N[1,m]} & & & 0 \end{array}\right] \tag{6.4.19}$$

Upon examination of (6.4.18) and (6.4.19), the following relationship can be derived

$$\underline{s}_{m+1}^{\dagger}\, R\, \underline{x}_{N[1,m+1]} = \underline{e}_{m+1}^{T} \tag{6.4.20}$$

where $\underline{e}_{m+1}$ is a unit basis vector whose $m+1^{st}$ element is 1 and $\underline{s}_{m+1}$ is a $(m+1) \times 1$ column vector defined by

$$\underline{s}_{m+1}^{\dagger} = [-(s^{m+1}\,\underline{y}_N)^{\dagger}\, A\, \underline{x}_{N[1,m]}\, R^{-1}\underline{x}_{N[1,m]},\ 1]/\ f_{N,m}^{d} \tag{6.4.21}$$

in which $f_{N,m}^{d}$ is a scalar defined by

$$f_{N,m}^{d} = (s^{m+1}\,\underline{y}_N)^{\dagger}\, \underline{d}_{N,m}^{x} = (\underline{d}_{N,m}^{y})^{\dagger}\, \underline{d}_{N,m}^{x} \tag{6.4.22a}$$

$$= (\underline{b}_{N-1,m}^{y})^{\dagger}\, \underline{b}_{N-1,m}^{x} = f_{N-1,m}^{r} \tag{6.4.22b}$$

Relationship (6.4.22b) is obtained from (6.3.7a) and (6.3.7b). After applying a similar analysis to the matrix $R\,\underline{y}_{N[1,m+1]}$ it can be shown that

$$\underline{t}_{m+1}^{\dagger}\, R\, \underline{y}_{N[1,m+1]} = \underline{e}_{m+1}^{T} \tag{6.4.23}$$

where $\underline{t}_{m+1}$ is a (m + 1) x 1 column vector expressed as

$$\underline{t}_{m+1} = [-(S^{m+1}\,\underline{x}_N)^{\dagger} A\, \underline{y}_{N[1,m]}\, R^{-1} \underline{y}_{N[1,m]},\ 1]/\ f^{d}_{N,m}{}^{*} \tag{6.4.24}$$

Applying the vector transpose operation to both sides of expression (6.4.23), we have

$$R\, \underline{x}_{N[1,m+1]}\, \underline{t}_{m+1} = \underline{e}_{m+1} \tag{6.4.25}$$

The inverse of the matrix $R\, \underline{x}_{N[1,m+1]}$ is readily found to be

$$R^{-1}\underline{x}_{N[1,m+1]} = \begin{bmatrix} & & & 0 \\ & R^{-1}\underline{x}_{N[1,m]} & & \vdots \\ & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix} + f^{d}_{N,m}\, \underline{t}_{m+1}\, \underline{s}^{\dagger}_{m+1} \tag{6.4.26}$$

Substitution of expression (6.4.26) into (6.4.16) then yields

$$P\, \underline{x}_{N[1,m+1]} = P\, \underline{x}_{N[1,m]} + A\, \underline{x}_{N[1,m+1]}\, f^{d}_{N,m}\, \underline{t}_{m+1}\, \underline{s}^{\dagger}_{m+1} \times A^{\dagger} \underline{y}_{N[1,m+1]} \tag{6.4.27}$$

After a simple algebraic manipulation, equation (6.4.27) is compactly expressed as

$$P\, \underline{x}_{N[1,m+1]} = P\, \underline{x}_{N[1,m]} + P\, \underline{d}^{x}_{N,m} \tag{6.4.28a}$$

$$= P\, \underline{x}_{N[1,m]} + (I - P\, \underline{x}_{N[1,m]})\, P\, \underline{d}^{x}_{N,m} \tag{6.4.28b}$$

$$= P\ \underline{x}_{N[1,m]} + P\ \underline{d}^{x}_{N,m}\ (I - P\ \underline{x}_{N[1,m]}) \qquad (6.4.28c)$$

where it is readily shown that $P\ \underline{d}^{x}_{N,m}$ is a projection operator onto the subspace spanned by $\underline{d}^{x}_{N,m}$ along the subspace which is orthogonal to the subspace spanned by $\underline{d}^{y}_{N,m}$ and is defined by

$$P\ \underline{d}^{x}_{N,m} = \frac{1}{f^{d}_{N,m}}\ \underline{d}^{x}_{N,m}\ (\underline{d}^{y}_{N,m})^{\dagger} \qquad (6.4.29)$$

Furthermore, equation (6.4.28c) can be expressed in the form

$$I - P\ \underline{x}_{N[1,m+1]} = (I - P\ \underline{d}^{x}_{N,m})\ (I - P\ \underline{x}_{N[1,m]}) \qquad (6.4.30)$$

Expression (6.4.30) will be used to find the forward error recursion in the next section.

In a similar manner, the following relationship may be also obtained

$$I - P\ \underline{y}_{N[0,m+1]} - (I - P\ \underline{\varepsilon}^{y}_{N,m})\ (I - P\ \underline{y}_{N[1,m]}) \qquad (6.4.31)$$

$$I - P\ \underline{y}_{N[1,m+1]} = (I - P\ \underline{d}^{y}_{N,m})\ (I - P\ \underline{y}_{N[1,m]}) \qquad (6.4.32)$$

where the projection operators $P\ \underline{\varepsilon}^{y}_{N,m}$ and $P\ \underline{d}^{y}_{N,m}$ are defined by

$$P\ \underline{\varepsilon}^{y}_{N,m} = \frac{1}{f^{\varepsilon}_{N,m}{}^{*}}\ \underline{\varepsilon}^{y}_{N,m}\ (\underline{\varepsilon}^{x}_{N,m})^{\dagger} \qquad (6.4.33)$$

$$P\ \underline{d}^{y}_{N,m} = \frac{1}{f^{d}_{N,m}{}^{*}}\ \underline{d}^{y}_{N,m}\ (\underline{d}^{x}_{N,m})^{\dagger} \qquad (6.4.34)$$

Expressions (6.4.31) and (6.4.32) will be used to find the recursion of forward error $\underline{\varepsilon}^{y}_{N,m}$ and backward error $\underline{b}^{y}_{N,m}$.

6.5 Order Update Recursions

In this section, we describe the order update recursive formulas which recursively compute the optimum $m+1^{st}$ order prediction error from the optimum m-th order prediction error. Expressions (6.4.15), (6.4.30), and (6.4.31) and (6.4.32) play a central role in obtaining these order update recursions.

Let us first derive the order update recursion for the forward prediction error vectors. Applying the projection operator (6.4.30) to the column vector $\underline{x}_N$ yields

$$\underline{\varepsilon}^x_{N,m+1} = (I - P\ \underline{d}^x_{N,m})\ \underline{\varepsilon}^x_{N,m} \tag{6.5.1}$$

Substitution of expression (6.4.29) into this relationship then yields

$$\underline{\varepsilon}^x_{N,m+1} = \underline{\varepsilon}^x_{N,m} - \frac{1}{f^d_{N,m}}\ \underline{d}^x_{N,m}\ \ (\underline{d}^y_{N,m})^\dagger\ \underline{\varepsilon}^x_{N,m} \tag{6.5.2}$$

The order update recursion for the N-th component of the forward prediction error vector is found to be

$$\underline{\varepsilon}^x_{N,m+1}(N) = \underline{\varepsilon}^x_{N,m}(N) - \frac{s_{N,m}}{f^r_{N-1,m}}\ \underline{b}^x_{N-1,m}(N-1) \tag{6.5.3}$$

where the partial-correlation coefficients are specified by

$$s_{N,m} = (\underline{d}^y_{N,m})^\dagger\ \underline{\varepsilon}^x_{N,m} = (S^{m+1}\underline{y}_N)^\dagger (I - P\ \underline{x}_{N[1,m]})\ \underline{x}_N \tag{6.5.4}$$

In a similar manner, applying the projection operator (6.4.32) to the column vector $\underline{y}_N$ leads to

$$\underline{\varepsilon}^{y}_{N,m+1}(N) = \underline{\varepsilon}^{y}_{N,m}(N) - \frac{t^{*}_{N,m}}{f^{r}_{N-1,m}{}^{*}} \underline{b}^{y}_{N-1,m}(N-1) \tag{6.5.5}$$

where

$$t^{*}_{N,m} = (\underline{d}^{x}_{N,m})^{\dagger} \underline{\varepsilon}^{y}_{N,m} = (S^{m+1}\underline{x}_{N})^{\dagger} (I - P_{\underline{y}_{N[1,m]}}) \underline{y}_{N} \tag{6.5.6}$$

Next, we will find the order update recursion for the backward prediction error vector. Applying the projection operator (6.4.15) to the column vector $S^{m+1}\underline{x}_{N}$ is found to yield

$$\underline{b}^{x}_{N,m+1} = (I - P_{\underline{\varepsilon}^{x}_{N,m}}) \underline{d}^{x}_{N,m} \tag{6.5.7}$$

Substitution of expression (6.4.14) into this relationship yields

$$\underline{b}^{x}_{N,m+1}(N) = \underline{b}^{x}_{N-1,m}(N-1) - \frac{t_{N,m}}{f^{\varepsilon}_{N,m}} \underline{\varepsilon}^{x}_{N,m}(N) \tag{6.5.8}$$

where the partial correlation coefficient $t_{N,m}$ is specified by

$$t_{N,m} = (\underline{\varepsilon}^{y}_{N,m})^{\dagger} \underline{d}^{x}_{N,m} = \underline{y}^{\dagger}_{N} (I - P_{\underline{x}_{N[1,m]}}) S^{m+1} \underline{x}_{N} \tag{6.5.9}$$

Similarly, applying projection operator (6.4.31) to the column vector $S^{m+1}\underline{y}_{N}$ is found to yield

$$\underline{b}^{y}_{N,m+1}(N) = \underline{b}^{y}_{N-1,m}(N-1) - \frac{s^{*}_{N,m}}{f^{\varepsilon}_{N,m}{}^{*}} \underline{\varepsilon}^{y}_{N,m}(N) \tag{6.5.10}$$

since

$$s^{*}_{N,m} = (\underline{\varepsilon}^{x}_{N,m})^{\dagger} \underline{d}^{y}_{N,m} = \underline{x}^{\dagger}_{N} (I - P_{\underline{y}_{N[1,m]}}) S^{m+1} \underline{y}_{N} \tag{6.5.11}$$

Next, we will derive the recursion for $f^{\varepsilon}_{N,m}$ and $f^{r}_{N,m}$. Manipulation of expressions (6.4.7), (6.4.28c) and (6.4.29) eventually leads to the form

$$f^{\varepsilon}_{N,m+1} = f^{\varepsilon}_{N,m} - \frac{s_{N,m}\, t_{N,m}}{f^{r}_{N-1,m}} \tag{6.5.12}$$

Expressions (6.4.22b), (6.3.4b) and (6.4.13c) yield the recursive formula

$$f^{r}_{N,m+1} = f^{r}_{N-1,m} - \frac{s_{N,m}\, t_{N,m}}{f^{\varepsilon}_{N,m}} \tag{6.5.13}$$

Consequently, expressions (6.5.2), (6.5.5), (6.5.8), (6.5.10), (6.5.12) and (6.5.13) represent the order update recursions.

6.6 Time Update Recursions

As a new element of the time series is observed, the partial reflection coefficients, forward errors, and backward errors may be recursively computed by making use of these values obtained at the last time instant. This being the case, these parameters are said to be "time updated" for each new data point.

The matrix $A\, \underline{x}_{N}[i,m]$ may be expressed in the recursive form

$$\begin{bmatrix} A\, \underline{x}_{N-1}[i,m] \\ \hline 0\ 0\ \cdots\ 0 \end{bmatrix} = A\, \underline{x}_{N}[i,m] - P_N\, A\, \underline{x}_{N}[i,m] \tag{6.6.1}$$

where P_N is the N x N projection matrix given by

$$P_N = \underline{e}_N \underline{e}_N^T \tag{6.6.2}$$

in which $\underline{e}_N$ is an N x 1 unit base vector. The matrix $R\,\underline{x}_{N[i,m]}$ may also be expressed as

$$R\,\underline{x}_{N-1[i,m]} = R\,\underline{x}_{N[i,m]} - A^{\dagger}\underline{y}_{N[i,m]}\,P_N\,A\,\underline{x}_{N[i,m]} \tag{6.6.3}$$

It then follows that the matrix $R^{-1}\underline{x}_{N-1[i,m]}$ is recursively updated by (see Appendix C)

$$R^{-1}\underline{x}_{N-1[i,m]} = R^{-1}\underline{x}_{N[i,m]} + \frac{1}{1 - \gamma_{i,m,N}} R^{-1}\underline{x}_{N[i,m]} \times \left[A^{\dagger}\underline{y}_{N[i,m]}\,P_N\,A\,\underline{x}_{N[i,m]}\,R^{-1}\underline{x}_{N[i,m]}\right] \tag{6.6.4}$$

where $\gamma_{i,m,N}$ is a scalar defined by

$$\gamma_{i,m,N} = \underline{e}_N^T\,A^{\dagger}\underline{y}_{N[i,m]}\,R^{-1}\underline{x}_N\,A\,\underline{x}_{N[i,m]}\,\underline{e}_N \tag{6.6.5}$$

Premultiplying expression (6.6.4) by $(I - P_N)\,A\,\underline{x}_{N[i,m]}$ and then postmultiplying that result by $A^{\dagger}\underline{y}_{N[i,m]}\,(I - P_N)$ leads to the recursive relationship

$$\begin{bmatrix} & & 0 \\ & P\,\underline{x}_{N-1[i,m]} & \vdots \\ & & \vdots \\ 0 & \cdots\cdots & 0 \end{bmatrix} = (I - P_N)\,P\,\underline{x}_{N[i,m]}\,(I - P_N)$$

$$+ \frac{1}{1 - \gamma_{i,m,N}} [I - P_N]\,P\,\underline{x}_{N[i,m]}\,P_N\,P\,\underline{x}_{N[i,m]}[I - P_N] \tag{6.6.6}$$

Since the vectors $\underline{y}_N$ and $\underline{z}_N$ are elements of vector space R^N, the time update recursion is given by

$$\underline{y}_N^{\dagger} (I - P\, \underline{x}_{N[i,m]})\, \underline{z}_N - \underline{y}_{N-1}^{\dagger} (I - P\, \underline{x}_{N-1[i,m]})\, \underline{z}_{N-1}$$

$$= \underline{y}_N^{\dagger}\, \nabla(I - P\, \underline{x}_{N[i,m]})\, \underline{z}_N \qquad (6.6.7)$$

where $\nabla(I - P\, \underline{x}_{N[i,m]})$ designates the time difference of the projection operator defined by

$$\nabla(I - P\, \underline{x}_{N[i,m]}) = P_N - P\, \underline{x}_{N[i,m]} + \begin{bmatrix} & & 0 \\ & & \vdots \\ P\, \underline{x}_{N-1[i,m]} & & \vdots \\ 0 \;\; . \;\; . \;\; . \;\; . \;\; 0 \end{bmatrix} \qquad (6.6.8)$$

Substitution of expression (6.6.6) into this expression yields

$$\nabla(I - P\, \underline{x}_{N[i,m]}) = P_N - P\, \underline{x}_{N[i,m]} + (I - P_N)\, P\, \underline{x}_{N[i,m]}\, (I - P_N)$$

$$+ \frac{1}{1 - \gamma_{i,m,N}} (I - P_N)\, P\, \underline{x}_{N[i,m]}\, P_N\, P\, \underline{x}_{N[i,m]}\, (I - P_N) \qquad (6.6.9)$$

Expression (6.6.9) is straightforwardly carried out by a simple algebraic manipulation and yields (see Appendix C)

$$\nabla(I - P\, \underline{x}_{N[i,m]}) = \frac{1}{1 - \gamma_{i,m,N}} (I - P\, \underline{x}_{N[i,m]})\, P_N\, (I - P\, \underline{x}_{N[i,m]}) \qquad (6.6.10)$$

Expression (6.6.10) is used to find the time update recursion formula. The partial correlation coefficient $s^{x}_{N,m}$ is recursively calculated by

$$s_{N,m} = s_{N-1,m} + \frac{\underline{b}^{y}_{N-1,m}{}^{*}(N-1)\ \underline{\varepsilon}^{x}_{N,m}(N)}{1 - \gamma_{1,m,N}} \tag{6.6.11}$$

In a similar manner, the partial correlation coefficients $t_{N,m}$ is recursively calculated by

$$t_{N,m} = t_{N-1,m} + \frac{\underline{\varepsilon}^{y}_{N,m}{}^{*}(N)\ \underline{b}^{x}_{N-1,m}(N-1)}{1 - \gamma_{1,m,N}} \tag{6.6.12}$$

The time update recursion for forward error is found to be

$$f^{\varepsilon}_{N,m} = f^{\varepsilon}_{N-1,m} + \frac{\underline{\varepsilon}^{y}_{N,m}{}^{*}(N)\ \underline{\varepsilon}^{x}_{N,m}(N)}{1 - \gamma_{1,m,N}} \tag{6.6.13}$$

The backward error is also given by

$$f^{r}_{N,m} = f^{r}_{N-1,m} + \frac{\underline{b}^{y}_{N,m}{}^{*}(N)\ \underline{b}^{x}_{N,m}(N)}{1 - \gamma_{o,m-1,N}} \tag{6.6.14}$$

A recursive formula for auxiliary parameter $\gamma_{1,m,N}$ can be obtained by using relationship (6.4.28c) to yield

$$\gamma_{1,m+1,N} = \gamma_{1,m,N} + \frac{\underline{b}^{x}_{N-1,m}(N-1)\ \underline{b}^{y}_{N-1,m}{}^{*}(N-1)}{f^{r}_{N-1,m}} \tag{6.6.15}$$

Finally, $\gamma_{o,m,N}$ can be computed by using the following relationship

$$\gamma_{o,m+1,N} = \gamma_{1,m,N} + \frac{\underline{\varepsilon}^{x}_{N,m}(N)\ \underline{\varepsilon}^{y}_{N,m}{}^{*}(N)}{f^{\varepsilon}_{N,m}} \tag{6.6.16}$$

which is directly obtained from expression (6.4.13c).

Thus we can use equation (6.6.11) and (6.6.12) to update the partial correlation coefficients. Equations (6.6.13) and (6.6.14) can be used to time update the forward and backward covariance errors $f^{\varepsilon}_{N,m}$ and $f^{r}_{N,m}$. The auxiliary parameters $\gamma_{1,m,N}$ and $\gamma_{o,m,N}$ are recursively computed by expression (6.6.15) and (6.6.16), respectively.

6.7 An Algorithm for Recursive ARMA Spectral Estimation

In this section, we summarize the recursive ARMA spectral estimation algorithm developed in the previous sections. For programming convenience, the following notations shall be used: $\varepsilon^{x}_{n}(m)$, $\varepsilon^{y}_{n}(m)$, $b^{x}_{n}(m)$, $b^{y}_{n}(m)$, $f^{\varepsilon}_{n}(m)$, $f^{r}_{n}(m)$, $s_{n}(m)$, $t_{n}(m)$, $\gamma_{o,n}(m)$ and $\gamma_{1,n}(m)$ in place of $\underline{\varepsilon}^{x}_{N,m}(N)$, $\underline{\varepsilon}^{y}_{N,m}(N)$, $\underline{b}^{x}_{N,m}(N)$, $\underline{b}^{y}_{N,m}(N)$, $f^{\varepsilon}_{N,m}$, $f^{r}_{N,m}$, $s_{N,m}$, $t_{N,m}$, $\gamma_{o,m,N}$ and $\gamma_{1,m,N}$, respectively. At each new data point, the parameters are recursively time updated (see section 6.6) and order updated from m=0 to m=p-1 (see section 6.5). The recursive ARMA spectral estimation algorithm can be presented as follows.

Step 1 Initial Condition (Time Update n=1)

$$\varepsilon^{x}_{1}(0) = \varepsilon^{y}_{1}(0) = b^{x}_{1}(0) = b^{y}_{1}(0) = 0$$

$$f_{1}(0) = f^{r}_{1}(0) = x_{1}y^{*}_{1},\ s_{i}(0) = t_{i}(0) = 0 \text{ for } i = 0, \ldots, p-1$$

Step 2 Initial Condition (Order Update, m=0)

$$\varepsilon^{x}_{n}(0) = b^{x}_{n}(0) = x_{n},\ \varepsilon^{y}_{n}(0) = b^{y}_{n}(0) = y_{n}$$

$$\gamma_{i,n-1}(-1) = \gamma_{i,n}(-1) = \gamma_{i,n}(0) = 0 \quad i = 0,1$$

$$f_n^{\varepsilon}(0) = f_n^{r}(0) = f_{n-1}^{r}(0) + y_n^{*} x_n$$

Step 3 Order Update Recursions
(m = 0, 1, ... , M for M = min (p-1, n-1))

(i) Forward Error

$$\varepsilon_n^{x}(m+1) = \varepsilon_n^{x}(m) - \frac{s_n(m)}{f_{n-1}^{r}(m)} b_{n-1}^{x}(m)$$

$$\varepsilon_n^{y}(m+1) = \varepsilon_n^{y}(m) - \frac{t_n^{*}(m)}{f_{n-1}^{r}(m)^{*}} b_{n-1}^{y}(m)$$

(ii) Backward Error

$$b_n^{x}(m+1) = b_{n-1}^{x}(m) - \frac{t_n(m)}{f_n^{\varepsilon}(m)} \varepsilon_n^{x}(m)$$

$$b_n^{y}(m+1) = b_{n-1}^{y}(m) - \frac{s_n^{*}(m)}{f_n^{\varepsilon}(m)^{*}} \varepsilon_n^{y}(m)$$

(iii) $f_n^{\varepsilon}(m)$, $f_n^{r}(m)$ and $\gamma_{1,n}(m)$

$$f_n^{\varepsilon}(m+1) = f_n^{\varepsilon}(m) - \frac{s_n(m)\, t_n(m)}{f_{n-1}^{r}(m)} \quad \text{if } n \leq p$$

$$f_n^{r}(m+1) = f_{n-1}^{r}(m) - \frac{s_n(m)\, t_n(m)}{f_n^{\varepsilon}(m)} \quad \text{if } n \leq p$$

$$\gamma_{1,n}(m+1) = \gamma_{1,n}(m) + \frac{b_{n-1}^{x}(m)\; b_{n-1}^{y}(m)^{*}}{f_{n-1}^{r}(m)}$$

$$\gamma_{0,n}(m+1) = \gamma_{1,n}(m) + \frac{\varepsilon_n^x(m) \quad \varepsilon_n^y(m)^*}{f_{N,m}^{\varepsilon}}$$

Step 4 Time Update Recursions (m = 0, 1, ... , M)

(i) Partial Correlation Coefficients

$$s_n(m) = s_{n-1}(m) + \frac{b_{n-1}^y(m)^* \varepsilon_n^x(m)}{1 - \gamma_{1,n}(m)}$$

$$t_n(m) = t_{n-1}(m) + \frac{\varepsilon_n^y(m)^* b_{n-1}^x(m)}{1 - \gamma_{1,n}(m)}$$

(ii) $f_{n-1}^{\varepsilon}(m)$ and $f_{n-1}^{r}(m)$

$$f_n^{\varepsilon}(m) = f_{n-1}^{\varepsilon}(m) + \frac{\varepsilon_n^y(m)^* \varepsilon_n^x(m)}{1 - \gamma_{1,n}(m)} \quad \text{if } n > p$$

$$f_n^{r}(m) = f_{n-1}^{r}(m) + \frac{b_n^y(m)^* b_n^x(m)}{1 - \gamma_{0,n}(m)} \quad \text{if } n > p$$

Step 5 Let n = n+1, if $n \neq N$ go to Step 2

Step 6 End of Algorithm

In above N is taken as a time index of a pair of the last observations x_N and y_N.

6.8 Numerical Examples

To test the recursive ARMA spectral estimation algorithm, the time series expressed by (4.5.2a), (4.5.2b) and (4.5.2c) were generated. A program listing of the fast algorithm used in obtaining the denominator coefficients of the ARMA model is illustrated in Appendix D.2. As a first example, 64 data samples were generated according to expressions (4.5.2a), (4.5.2b) and (4.5.2c). These data samples are plotted in Fig. 6.8.1(a). The fast algorithm was then applied to this 64 observations to obtain an ARMA spectral estimate with model order (4,4). The forward error sequence $\underline{\varepsilon}^{x}_{n,4}(n)$ (n = 1, ... , 64) is plotted in Fig. 6.8.1(b). Comparing Figures 6.8.1(a) and 6.8.1(b), the forward error sequence is observed to be more random (uncorrelated) than the given data samples indicating a desired whitening effect. The resultant spectral estimate is shown in Fig. 6.8.1(c). The resolution of the two peaks is evident, however, the estimated level of the first peak is lower than that of second peak. Next, 500 data samples of the same time series expressed were generated. These samples are plotted in Fig. 6.8.2(a). The forward error sequence $\underline{\varepsilon}^{x}_{n,4}(n)$ (n = 1, ... , 500) obtained by the fast algorithm is plotted in Fig. 6.8.2(b). It is observed that the forward error sequence converges in a relatively rapid manner. In Fig. 6.8.2(c), the resultant spectral estimate of model order (4,4) is illustrated. The resolution of the two peaks is again evident. In addition, the height of the two peaks are equal as desired. As these examples illustrate, the fast algorithm maintains a high quality of spectral performance.

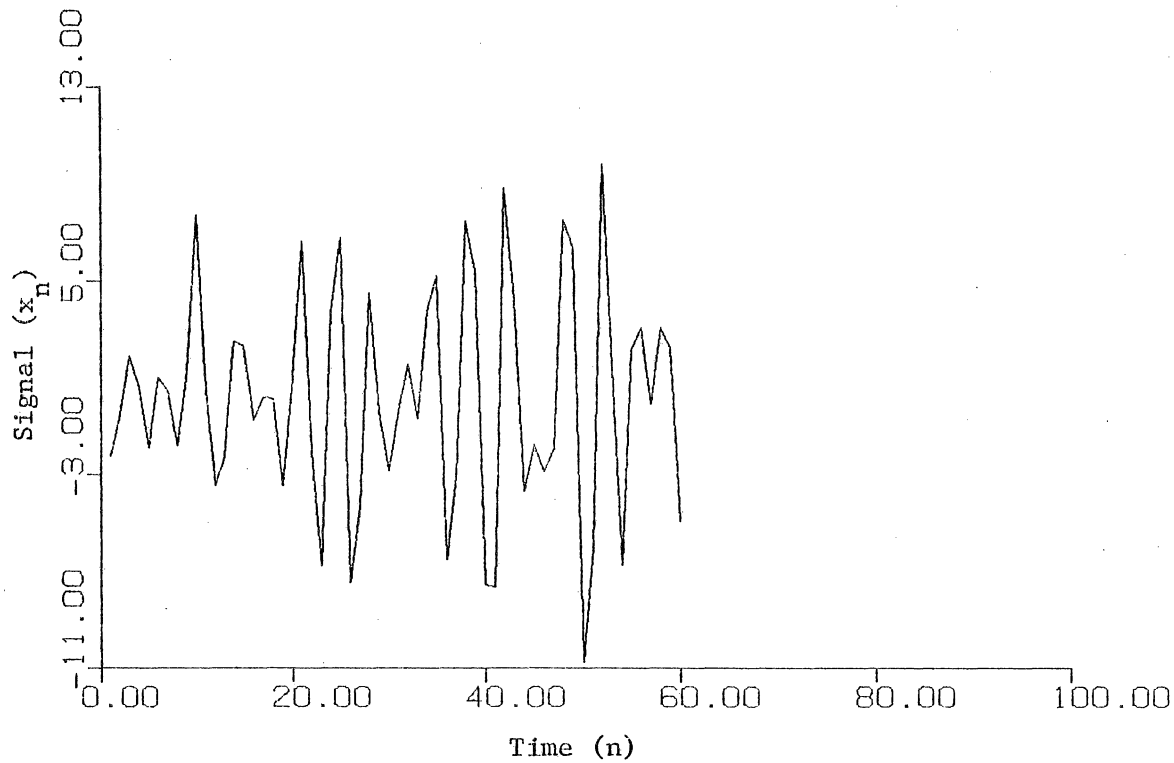


Fig. 6.8.1(a) Signal Samples (N = 64)

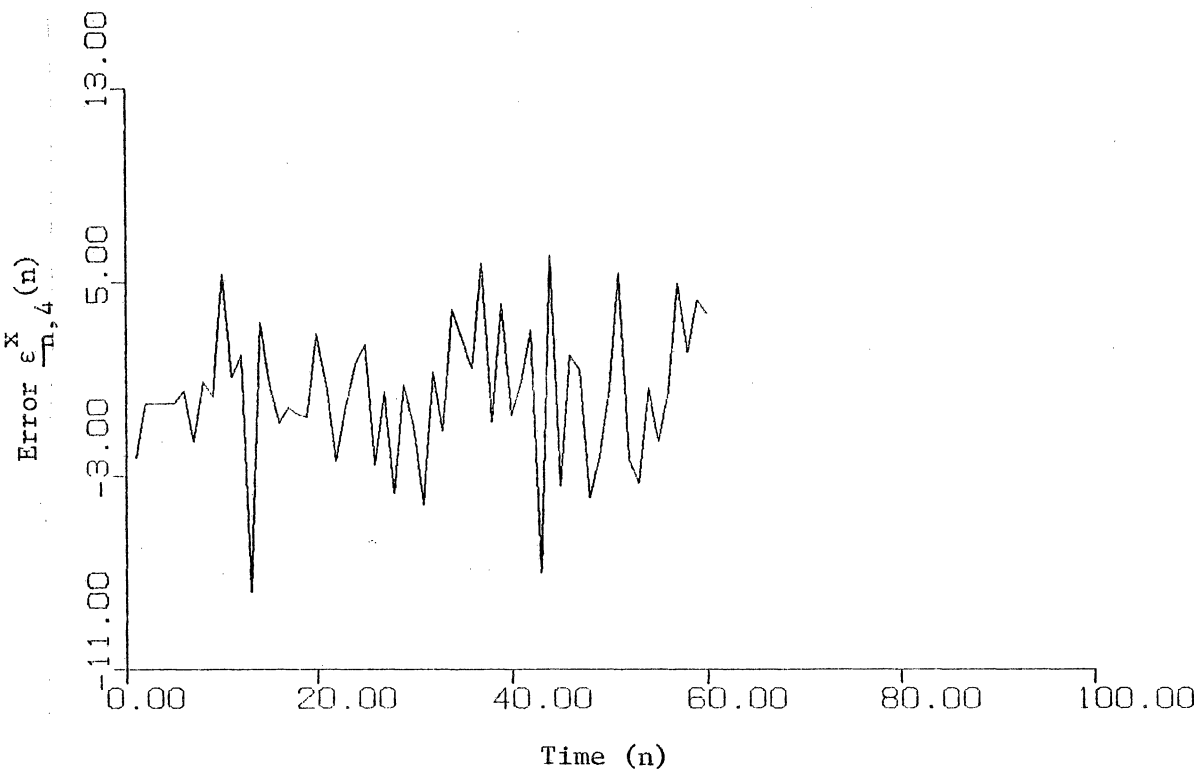


Fig. 6.8.1(b) Forward Error (N = 64)

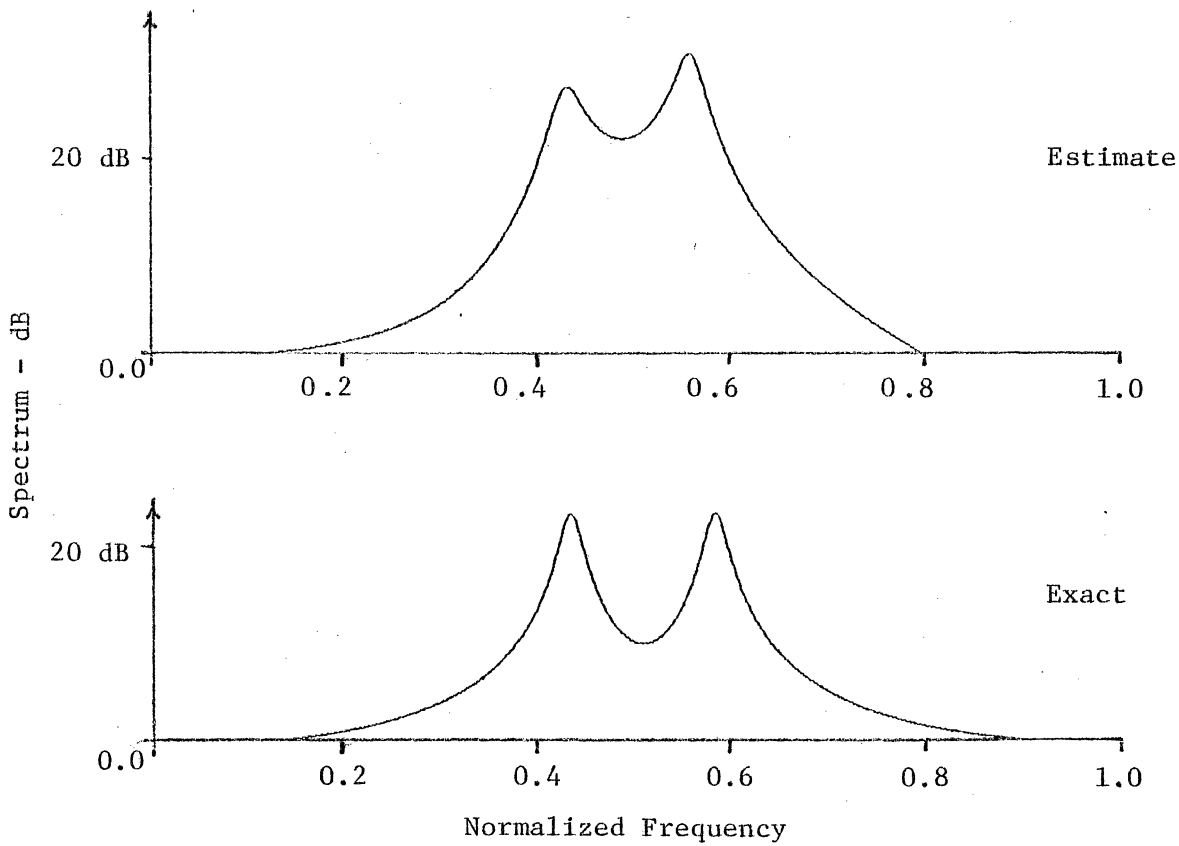


Fig. 6.8.1(c) Spectral Estimate Performance

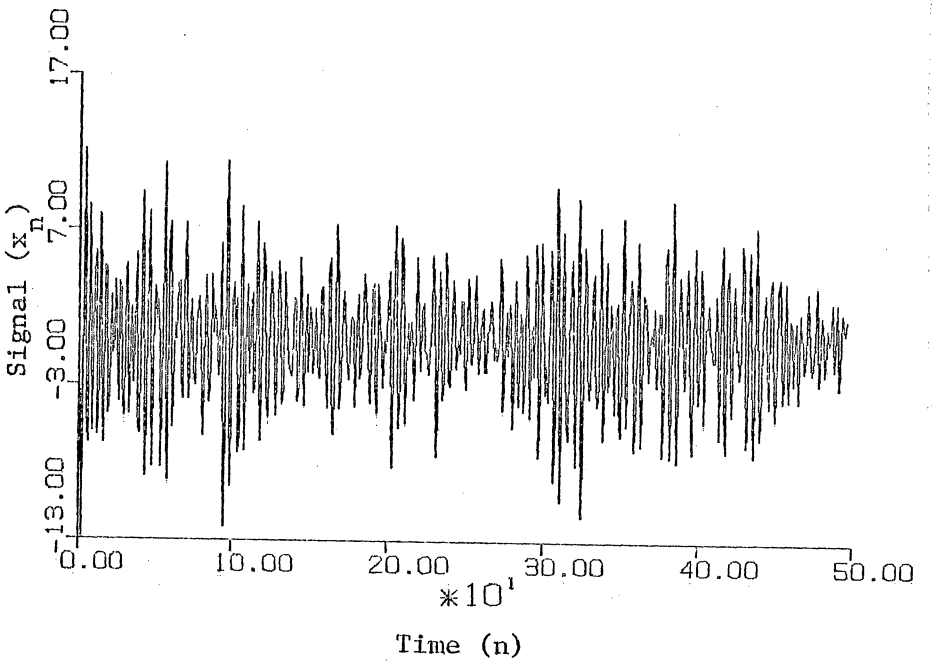


Fig. 6.8.2(a) Signal Samples (n = 500)

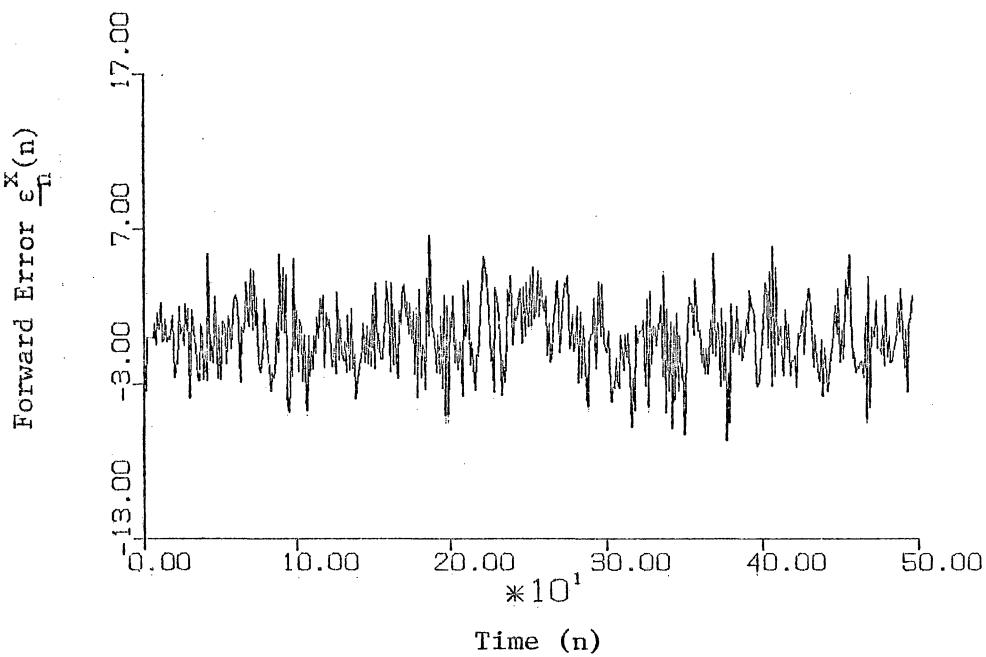


Fig. 6.8.2(b) Forward Error

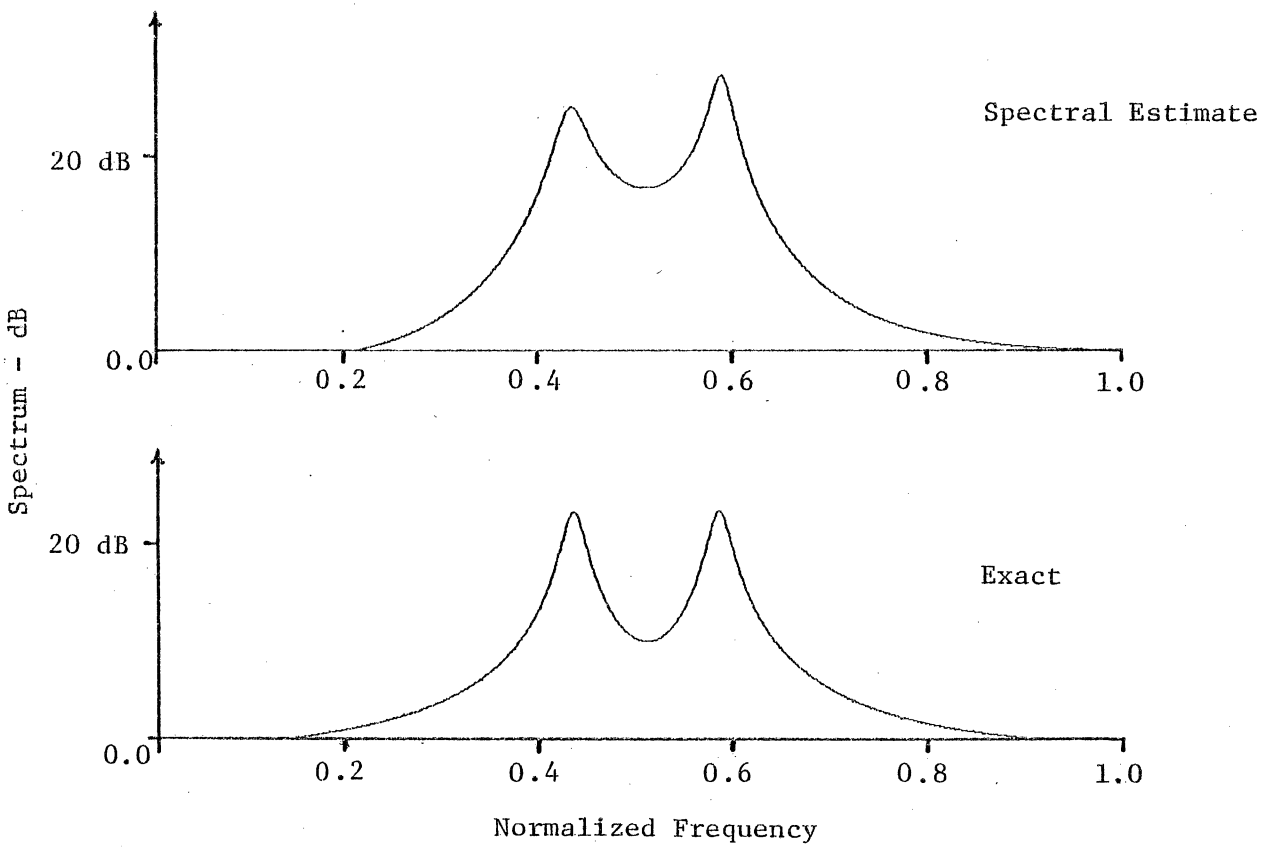


Fig. 6.8.2(c) Spectral Estimate Performance

6.9 Summary

A recursive algorithm has been proved and stated for efficiently updating partial reflection coefficients of an ARMA spectral estimation model. The computational requirement for the order update recursions and time update recursions are $12(M + 1)$ and $10(M + 1)$, respectively, where M is taken to be the minimum of either p-1 or n-1. Numerical examples show that implementation of premodification will result in only a small degradation of spectral performance. If q=0, the ARMA model is converted to the AR model. A recursive AR algorithm can be also developed based on a less general vector space approach discussed in this chapter (see Lee and Morf, 1980).

Chapter 7

CONCLUSION

The development of computationally fast algorithms for high performance ARMA spectral estimation was presented. The required computation for the unmodified method was reduced to $O(4p^2)$ by using a generalized Levinson's approach. Methods of data modifications were applied to reduce the computational complexity. Modifications, referred to as post-modification with $p = q$ and pre- and post-modification, achieved a computational complexity of $O(p \log p)$. A fast recursive algorithm with a computational complexity of $O(p)$ was developed based on the pre-modification method.

The spectral performance of these methods was compared for various numerical examples. Spectral degradation had been expected, because of the restriction $t = p$ and the underlying data modification, however, these numerical examples illustrated only a small degradation in spectral performance. Moreover, the spectral estimation performance of these new methods has been found to be typically far superior to such contemporary approaches as the Box-Jenkins and maximum entropy methods.

Finally, considering the above two aspects, namely, fast computational implementations and high performance spectral estimations, these new methods promise to be primary spectral estimation tools.

REFERENCES

Aho, A. V., J. E. Hopcroft and J. D. Ullman, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, MA, 1974.

Akaike, H., "Statistical Prediction Identification," Ann. Inst. Statist. Math., Vol. 22, pp. 203-217, 1970.

Akaike, H., "Fitting Autoregressive Models for Prediction," Ann. Inst. Statist. Math., Vol. , pp. 243-247, 1969.

Bartlett, M. S., An Introduction to Stochastic Process with Special Reference to Methods and Applications, Cambridge University Press, New York, 1953.

Bitmead, B. and B. Anderson, "Asymptotically Fast Solution of Toeplitz and Related Systems of Linear Equations," Technical Report No. EE7915, July 1979.

Blackman, R. B. and J. W. Tukey, "The Measurement of Power Spectra from the Point of View of Communication Engineering, New York:Dover, 1959.

Bos, A. V. d., "Alternative Interpretation of Maximum Entropy Spectral Analysis," IEEE Transaction on Information Theory, Vol. IT-17, pp. 493-494, July 1971.

Box, G. E. and Jenkins, G. H., Time Series Analysis: Forecasting and Control, San Francisco: Holden Day, 1976, revised edit.

Bruzzone, S. and Kaveh, M., 'On Some Suboptimum ARMA Spectral Estimators,' IEEE Transactions on Acoustic, Speech and Signal Processing, Vol. ASSP-28, Dec. 1980, pp. 753-755.

Burg, J. P., "Maximum Entropy Spectral Analysis," 37th Annual International SEG Meeting, Oklahoma City, 1967.

Burg, John P., "Maximum Entropy Spectral Analysis," Ph.D. Dissertation, Department of Geophysics, Stanford University, Stanford, California, May 1975.

Cadzow, J. A., "ARMA Spectral Estimation: An Iterative Procedure," Proceedings of the RADC Spectrum Estimation Workshop, October 3, 4, & 5, 1979a, pp. 67-79.

Cadzow, J. A., "ARMA Spectral Estimation: An Efficient Closed-Form Procedure," Proceedings of the RADC Spectrum Estimation Workshop, October 3, 4, & 5, 1979b, pp. 81-97.

Cadzow, J. A., "High Performance Spectral Estimation - A New ARMA Method," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-28, No. 5, October, 1980a.

Cadzow, J. A., "ARMA Spectral Estimation: A Model Equation Error Procedure," IEEE 1980 International Conference on Acoustics, Speech and Signal Processing, April 1980b.

Cadzow, J. A. and K. Ogino, "Adaptive ARMA Spectral Estimation," Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing, March 31, 1981, Vol. 2, pp. 475-479.

Cadzow, J. A. and K. Ogino, "Adaptive Rational Spectral Estimation," paper in preparation.

Cooley, J. W. and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," Mathematics of Computation (1965), Vol. 19, No. 90, pp. 297-301.

Friedlandar, Recursive Algorithms for Ladder Forms, Technical Memorandum TM 5334-02, January 1980.

Friedlandar, B., M. Morf, T. Kailath and L. Ljung, "New Inversion Formula for Matrices Classified in Terms of their Distance from Toeplitz Matrices," Linear Algebra and its Application, Vol. 27, pp. 31-60, 1979.

Friedlandar, B., T. Kailath, M. Morf and L. Ljung, "Extended Levinson and Chandrasekhar Equations for General Discrete-Time Linear Estimation Problems," IEEE Transactions on Automatic Control, Vol. AC-23, No. 4, August 1978.

Grenander, U. and M. Rosenblatt, Statistical Analysis of Stationary Time Series, John Wiley & Sons, Inc., New York, 1957.

Gustavson, F. G. and Yun, D. Y. Y., 'Fast Algorithms for Rational Hermite Approximation and Solution of Toeplitz Systems,' IEEE Transactions on Circuits and Systems, Vol. CAS-26, No. 9, September 1979, pp. 750-755.

Gutowski, P. R., Robinson, E. A., Treitel, S., 'Spectral Estimation: Fact or Fiction,' IEEE Transaction on Geoscience Electronics, Vol. GE-16, No. 2, April 1978, pp. 80-84.

Haykin, S., Editor, Nonlinear Methods of Spectral Analysis, Springer Verlag, New York, 1979.

Huffman, S. D. and L. W. Nolte, "Adaptive Linear Estimation Based on Time Domain Orthogonality," Proceeding 1980 International Conference on Acoustic, Speech, and Signal Processing, Denver, CO, April 1980, pp. 453-456.

Jenkins, G. M. and D. G. Watts, Spectral Analysis and Its Applications, Holden-Day, Inc., San Francisco, 1968.

Jones, R. H., "Autoregression Order Selection," Geophysics, Vol. 41, pp. 771-773, Aug. 1976.

Kailath, T., S. Y. Kung and M. Morf, "Displacement Ranks of Matrices and Linear Equations," J. Math. Analysis and Applications, Vol. 68, No. 2, pp. 395-407, April 1979.

Kaveh, M., "High Resolution Spectral Estimation for Noisy Signals," IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. ASSP-27, No. 3, pp. 286-287, June 1979.

Kinkel, J. F., J. Perl, L. Scharf and A. Stubberud, "A Note on Covariance-Invariant Digital Filter Design and Autoregressive Moving Average Spectrum Analysis," IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. ASSP-27, pp. 200-202, April 1979.

Koopmans, L. H., Spectral Analysis of Time Series, Academic Press, New York, 1974.

Lacoss, R. T., "Data Adaptive Spectral Analysis Methods," Geophysics, Vol. 36, pp. 661-675, Aug. 1971.

Lee, D. T. and M. Morf, "Recursive Square-Root Ladder Estimation Algorithms," Proceeding 1980 International Conference on Acoustic, Speech and Signal Processing, Denver, CO, April 1980, pp. 1005-1017.

Levinson, N., "The Wiener RMS (root mean square) Error Criterion in Filter Design and Prediction," J. Math Phys., Vol. 25, pp. 261-278:Jan. 1947.

Luenberger, Optimization by Vector Space Methods, John Wiley & Sons, Inc., 1969.

Luenberger, D., Introduction on Linear and Nonlinear Programming, Addison-Wesley, Reading, Mass., 1973.

Morf, M., "Doubling Algorithms for Toeplitz and Related Equations," Proceeding IEEE 1980 International Conference on Acoustic, Speech and Signal Processing, Denver, CO, April 1980, pp. 954-959.

Morf, M. and T. Kailath, "Square-Root Algorithms for Least-Squares Estimation," IEEE Transactions on Automatic Control, Vol. AC-20, No. 4, August 1975, pp. 487-497.

Morf, M., D. T. Lee, J. R. Nickolls and A. Vieira, "A Classification of Algorithms for ARMA Models and Ladder Realizations," 1977 IEEE International Conference Record on Acoustic, Speech and Signal Processing, May 9-11, 1977, pp. 13-19.

Morf, M. and D. T. Lee, "Recursive Least Squares Ladder Forms for Fast Parameter Tracking," Proceeding 1978 IEEE Conference on Decision and Control, San Diego, CA, pp. 1362-1367, January 12, 1979.

Mullis, C. T. and R. A. Roberts, "The Use of Second-Order Information in the Approximation of Discrete-Time Linear Systems," IEEE Transactions on Acoustic, Speech and Signal Processing, Vol. ASSP-24, pp. 226-238, 1976.

Oppenheim, A. V. and R. W. Schafer, Digital Signal Processing, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

Pao, Y. and D. T. Lee, "Additional Results on the Cadzow ARMA Method for Spectrum Estimation," Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing, March 31, 1981, Vol. 2, pp. 480-483.

Parzen, E., Multiple Time Series Modeling, Multivariate Analysis-II, P. R. Krishnaiah (editor), New York Academic Press, 1969.

Parzen, Emanuel, "Some Recent Advances in Time Series Modeling," IEEE Transactions on Automatic Control, Vol. AC-19, pp. 723-730, December 1974.

Steiglitz, K., "On the Simultaneous Estimation of Poles and Zeroes in Speech Analysis," IEEE Transactions on Acoustics, Speech and Signal Processing, Vol. ASSP-25, No. 3, pp. 229-234, June 1977.

Sullivan, T. M., O. L. Frost and J. R. Treichler, "High Resolution Signal Estimation," ARGO System, Inc. Tech. Rep., June 1978.

Tong, H., "Autoregressive Model Fitting with Noisy Data by Akaikes Information Criterion," IEEE Trans. Inform. Theory, Vol. IT-25, pp. 476-480, July 1975.

Tong, H., "Fitting a Smooth Moving Average to Noisy Data," IEEE Trans. Inform. Theory, Vol. IT-26, pp. 493-496, July 1976.

Tong, H., "More on Autoregressive Model Fitting with Noisy Data by Akaikes Information Criterion," IEEE Trans. Inform. Theory, Vol. IT-23, pp. 409-411, May 1977.

Tretter, S. A., Introduction to Discrete-Time Signal Processing, John Wiley & Sons, Inc., New York, 1976.

Tretter, S. A. and Steiglitz, K., 'Power spectrum identification in terms of rational models,' IEEE Transaction on Automatic Control, Vol. AC-12, pp. 185-188, April 1967.

Ulrych, T. J., "Maximum Entropy Power Spectrum of Long Period Geomagnetic Reversals," Nature, 235, pp. 218-219, 1972.

Ulrych, T. J. and T. N. Bishop, "Maximum Entropy Spectral Analysis and Autoregressive Decomposition," Rev. Geophys. Space Phys., Vol. 13, pp. 183-200, Feb 1975.

Welch, P. D., "The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodgrams," IEEE Transactions on Audio and Electroacoustics, Vol. AU-15, pp. 70-73, June 1967.

Widrow, B., "Adaptive Filters," In Aspects of Network and System Theory, Part IV, R. E. Kalman and N. DeClaris, Editors, Holt, Rhinehart and Winston, New York, 1971.

Widrow, B. and M. E. Hoff, "Adaptive Switching Cricuits," in 1960 WESCON Conv. Rec., Pt. 4, pp. 96-140.

Wiener, N., The Extrapolation, Interpolation and Smoothing of Stationary Time Series, Wiley, New York, 1949.

Appendix A

RECURSIVE AR SPECTRAL ESTIMATION

A.1 Introduction

In many relevant signal processing applications, one seeks to characterize the spectral density of a time series based upon a finite set of time series observations. Without loss of generality, this sample observation set is taken to be the contiguous set of N real valued measurements as given by

$$x(1),\ x(2),\ \ldots\ ,\ x(N) \tag{A.1.1}$$

One of the most widely used spectral estimation models is obtained by postulating the following autoregressive (AR) structure

$$x(n) + a_1 x(n-1) + \ldots + a_m x(n-m) = \varepsilon(n) \tag{A.1.2}$$

in which $\varepsilon(n)$ is a white noise time series with zero mean and variance σ_ε^2. Our object will be that of modeling an underlying time series $\{x(n)\}$ with the AR model structure (A.1.2) in which the a_k coefficients are estimated from the given finite set of observations (A.1.1). This is readily achieved by applying the well known one-step predictor.

An m-th order one-step predictor, by definition, estimates the value of a random time series using a linear combination of the most recent m samples. Namely, the sample x(n) is estimated by means of the relationship

$$\hat{x}(n) = - \sum_{k=1}^{m} a_k \, x(n-k) \tag{A.1.3}$$

The difference between this predicted value and the observed value $x(n)$ over the observation interval is called the prediction error and is specified by

$$e(n) = x(n) - \hat{x}(n) \qquad m < n \leq N \tag{A.1.4}$$

or

$$e(n) = x(n) + \sum_{k=1}^{m} a_k \, x(n-k) \qquad m < n \leq N \tag{A.1.5}$$

Writing these error expressions in matrix form yields

$$\underline{e} = \underline{x} + X\underline{a}$$

where $\underline{a}$, $\underline{e}$, and $\underline{x}$ are m x 1, (N-m) x 1, and (N-m) x 1 column vectors, respectively, given by

$$\underline{a} = [a_1, \ \ldots \ , a_m]^T \tag{A.1.7a}$$

$$\underline{e} = [e(m+1), \ e(m+2), \ \ldots \ , e(N)]^T \tag{A.1.7b}$$

$$\underline{x} = [x(m+1), \ x(m+2), \ \ldots \ , x(N)]^T \tag{A.1.7c}$$

and X is an (N-m) x m matrix specified by

$$X = \begin{bmatrix} x(m) & x(m+1) & \ldots & x(N-1) \\ x(m-1) & x(m) & \ldots & x(N-2) \\ \vdots & \vdots & & \vdots \\ x(1) & x(2) & \ldots & x(N-m) \end{bmatrix}^T \tag{A.1.7d}$$

where the superscript T denotes the transpose operation.

The a_k coefficients are to be now selected so as to cause each of the predictor error terms e(n) to be close to zero. This selection process will give rise to the so-called optimal one-step predictor. To achieve the required objective of setting the e(n) to be near zero, one typically appeals to the least squares method which minimizes a squared error criterion of the form

$$f(\underline{a}) = \underline{e}^T W \underline{e} \tag{A.1.8}$$

where W is an (N-m) x (N-m) nonnegative definite square matrix. The minimization of this quadratic functional with respect to the column vector $\underline{a}$ is straightforwardly carried out and results in

$$X^T W X \underline{a}^o = X^T W \underline{x} \tag{A.1.9}$$

It can be shown that the resulting power spectral density estimate of the time series {x(n)} is then given by (Haykin, 1979)

$$S_X(\omega) = \frac{\sigma_\varepsilon^2}{|1 + a_1^o e^{-j\omega} + a_2^o e^{-2j\omega} + \ldots + a^o e^{-mj\omega}|^2} \tag{A.1.10}$$

where the a_k^o coefficients are obtained upon solving relationship (A.1.9). Generally the solution of relationship (A.1.9) requires on the order of m^3(i.e.$0(m^3)$) number of multiplications and additions if that relationship is directly used. This computational requirement can be excessive in many real time applications. It has been recently shown by Lee and Morf (1980) that this computational requirement can be reduced to 0(m) by slightly reformulating the matrix X and column

vector $\underline{x}$. In many interesting cases, fortunately, the solution to this modified system of equations will be close to that of the desired solution as represented by expression (A.1.9). In this Appendix the method which is identical to the LMS algorithm of Lee and Morf (1980) is presented with more emphasis on insightful development.

This general modification methodology shall herein be referred to as data modification. Applying the specific data modification method referred to as prewindowing, the matrix X is reformulated as the N x m matrix given by

$$X = \begin{bmatrix} 0 & x(1) & x(2) & \cdots & x(m) & \cdots & x(N-1) \\ 0 & 0 & x(1) & \cdots & x(m-1) & \cdots & x(N-2) \\ \vdots & \vdots & 0 & & \vdots & & \vdots \\ \vdots & \vdots & \ddots & & \vdots & & \vdots \\ \vdots & \vdots & \ddots & & \vdots & & \vdots \\ 0 & 0 & 0 \cdots 0 & & x(1) & \cdots & x(N-m) \end{bmatrix}^T \qquad \text{(A.1.11)}$$

while the N x 1 column vector $\underline{x}$ is specified by

$$\underline{x} = [x(1), x(2), \ldots, x(N)]^T \qquad \text{(A.1.12)}$$

If these new entrants are substituted into relationship (A.1.9), an efficient solution procedure for $\underline{a}^o$ is possible. The structure of this reformulated matrix X and the column vector $\underline{x}$ enables us to obtain a recursive least square spectral estimation algorithm which has an excellent convergence behavior and a fast parameter tracking capability relative to the former structure. The development of this algorithm is predicated on the utilization of projection operator theory (Luenberger, 1969). In the sections which follow the necessary

projection operator theory to be used in the algorithm is described.

A.2 Vector Space Formulation

In this section, the given spectral estimation problem will be cast into a convenient vector space setting. It will be assumed that the following observations of the time series $\{x(n)\}$ as specified by

$$x(1),\ x(2),\ \ldots\ ,\ x(N) \tag{A.2.1}$$

are given. This in turn will give rise to the associated column data vector

$$\underline{x}_N = [x(1),\ x(2),\ \ldots\ ,\ x(N)]^T \tag{A.2.2}$$

The vector $\underline{x}_N$ lies in the product space

$$H_N = Rx \quad Rx \quad \ldots \quad xR \quad = R^N \tag{A.2.3}$$

This vector space can be made into an inner product space by defining the following inner product between any two elements $\underline{x}_N$, $\underline{y}_N \in H_N$

$$< \underline{x}_N,\ \underline{y}_N > = \underline{x}_N^T\, \underline{y}_N = \sum_{n=1}^{N} x(n)\ y(n) \tag{A.2.4}$$

The corresponding induced norm of $\underline{x}_N$ is then given by

$$|| \underline{x}_N || = [< \underline{x}_N,\ \underline{x}_N >]^{\frac{1}{2}} = \left[\sum_{n=1}^{N} x^2(n) \right]^{\frac{1}{2}} \tag{A.2.5}$$

We next define the shift matrix S which is represented by the N x N matrix

$$S = \begin{bmatrix} 0 & & & \\ 1 & \ddots & & \\ & \ddots & \ddots & \\ & & 1 & 0 \end{bmatrix} \qquad \text{(A.2.6)}$$

Applying the shift matrix m times to the column vector $\underline{x}_N$ is seen to yield

$$S^m \underline{x}_N = [\underbrace{0, \ldots, 0}_{\text{m zeroes}}, x(1), \ldots, x(N-m-1), x(N-m)]^T \qquad \text{(A.2.7)}$$

We next construct the subspace $M_{\underline{x}_N[i,m]}$ which is spanned by the set of vectors $S^i \underline{x}_N, \ldots, S^m \underline{x}_N$. This subspace will be suggestively denoted by

$$M_{\underline{x}_N[i,m]} = \{S^i \underline{x}_N, S^{i+1} \underline{x}_N, \ldots, S^m \underline{x}_N\} \qquad \text{(A.2.8)}$$

where the first integer index i may take on any value in the set $\{0, 1, \ldots, m\}$. Next, we let $P_{\underline{x}_N[i,m]}$ designate the projection operator onto the subspace $M_{\underline{x}_N[i,m]}$ along the subspace $M^{\perp}_{\underline{x}_N[i,m]}$. This projection operator can be shown to have the form

$$P_{\underline{x}_N[i,m]} \triangleq X_{\underline{x}_N[i,m]} \left[X_{\underline{x}_N}{}^T{}_{i,m} \, X_{\underline{x}_N[i,m]} \right]^{-1} X_{\underline{x}_N[i,m]} \qquad \text{(A.2.9)}$$

where $X_{\underline{x}_N[i,m]}$ is the N x (m-i+1) matrix composed of the following ordered set of column vectors

$$X_{\underline{x}_N[i,m]} = [S^i \underline{x}_N, S^{i+1} \underline{x}_N, \ldots, S^m \underline{x}_N] \qquad \text{(A.2.10)}$$

Similarly, the projection operator on the orthogonal complement of subspace $M\underline{x}_N[i,m]$ is denoted by

$$P^{\perp}\underline{x}_N[i,m] = I - P\underline{x}_N[i,m] \tag{A.2.11}$$

where I is the N x N identity matrix. It then follows that

$$P\underline{x}_N[i,m]\ \underline{y}_N = \underline{y}_N \qquad \text{if } \underline{y}_N \in M\underline{x}_N[i,m] \tag{A.2.12}$$

$$P^{\perp}\underline{x}_N[i,m]\ \underline{y}_N \perp S^k\underline{x}_N \quad i \leq k \leq m, \text{ if } \underline{y}_N \in H_N \tag{A.2.13}$$

Expression (A.2.12) and (A.2.13) specify those properties of the projection operators which will be utilized when developing a recursive least square algorithm in the next section.

A.3 Linear Prediction and Projection Operator

In this section, we will define three methods of linear prediction, namely, forward prediction, backward prediction, and delayed backward prediction. These projection operators will play a central role in the algorithmic solution procedure to be developed.

A.3.1 Forward Prediction

The m-th order forward prediction method is referred to as that specific procedure for estimating the column vector $\underline{x}_N$ by means of a linear combination of the set of m shifted vectors $\{S^1\underline{x}_N, S^2\underline{x}_N, \ldots, S^m\underline{x}_N\}$. It then follows that the m-th order forward prediction estimate of $\underline{x}_N$ is of the form

$$\hat{\underline{x}}_N[1,m] = -\sum_{k=1}^{m} a_k S^k \underline{x}_N \tag{A.3.1}$$

while the associated forward error vector is specified by

$$\underline{\varepsilon}_{N,m} = \underline{x}_N - \hat{\underline{x}}_N[1,m] \tag{A.3.2a}$$

$$= \underline{x}_N + \sum_{k=1}^{m} a_k S^k \underline{x}_N \tag{A.3.2b}$$

Upon examination of the structure of the shifted vector $S^k \underline{x}_N (k = 1, \ldots, m)$, expression (A.3.2b) leads to the aforementioned prewindowing formula where X and $\underline{x}$ are given by (A.1.11) and (A.1.12), respectively.

The problem at hand is to then find the scalar constants $a_1, a_2, \ldots, a_m$ which minimize the squared forward prediction error

$$f(\underline{a}) = \| \underline{x}_N - \hat{\underline{x}}_N[1,m] \|^2 \tag{A.3.3}$$

According to the projection theorem (Luenberger, 1969), $f(\underline{a})$ is minimized when the error vector is orthogonal to each of the one-dimensional subspaces spanned by $S^i \underline{x}_N (i = 1, \ldots, m)$. Thus, we have the orthogonality relationship expressed by

$$(\underline{x}_N - \hat{\underline{x}}_N[1,m]) \perp S^i \underline{x}_N \qquad \text{for } i = 1, 2, \ldots, m \tag{A.3.4}$$

which takes the inner product format

$$< \underline{x}_N - \hat{\underline{x}}_N[1,m], S^i \underline{x}_N > = 0 \qquad \text{for } i = 1, 2, \ldots, m \tag{A.3.5}$$

Substitution of expression (A.3.1) into (A.3.5) yields the set of linear algebraic equations

$$\sum_{k=1}^{m} \langle S^k \underline{x}_N, S^i \underline{x}_N \rangle a_k = -\langle \underline{x}_N, S^i \underline{x}_N \rangle \qquad \text{(A.3.6)}$$

$$\text{for } i = 1, 2, \ldots, m$$

for the optimum set of a_k prediction coefficients. These equations are called the normal equations and can be put into the matrix form

$$\underline{X}_{\underline{x}N[1,m]}^{T} \underline{X}_{\underline{x}N[1,m]} \underline{a} = -\underline{X}_{\underline{x}N[1,m]}^{T} \underline{x}_N \qquad \text{(A.3.7a)}$$

where

$$\underline{X}_{\underline{x}N[1,m]} = [S^1 \underline{x}_N, S^2 \underline{x}_N, \ldots, S^m \underline{x}_N] \qquad \text{(A.3.7b)}$$

$$\underline{a} = [a_1, a_2, \ldots, a_m]^T \qquad \text{(A.3.7c)}$$

Solving equation (A.3.7a) for $\underline{a}$ and substituting this solution into expression (A.3.1) then yields the optimum prediction vector

$$\hat{\underline{x}}_{N[1,m]} = \underline{X}_{\underline{x}N[1,m]} \left[\underline{X}_{\underline{x}N[1,m]}^{T} \underline{X}_{\underline{x}N[1,m]} \right]^{-1} \underline{X}_{\underline{x}N[1,m]}^{T} \underline{x}_N \qquad \text{(A.3.8)}$$

Upon examination of the projection operator (A.2.9) and this expression, $\hat{\underline{x}}_{N[1,m]}$ is seen to be compactly specified by

$$\hat{\underline{x}}_{N[1,m]} = P_{\underline{x}N[1,m]} \underline{x}_N \qquad \text{(A.3.9)}$$

Thus, we see that $\hat{\underline{x}}_{N[1,m]}$ is obtained by projecting $\underline{x}_N$ onto the subspace $M_{\underline{x}N[1,m]}$ and the m-th order forward prediction error vector is obtained by projecting $\underline{x}_N$ onto the orthogonal complement of $M_{\underline{x}N[1,m]}$ in the H_N, that is

$$\underline{\varepsilon}_{N,m} = P^{\perp}_{\underline{x}N[1,m]} \underline{x}_N \qquad \text{(A.3.10)}$$

The corresponding minimum mean squared error is then defined to be

$$f^{\varepsilon}_{N,m} = \underline{\varepsilon}^{T}_{N,m}\,\underline{\varepsilon}_{N,m} = \underline{\varepsilon}^{T}_{N,m}\,\underline{x}_{N}$$

A.3.2 Backward Prediction

The m-th order backward prediction method is that procedure of estimating the m-th shifted column vector $S^{m}\underline{x}_{N}$ by a linear combination of the set of shifted vectors $\{S^{0}\underline{x}_{N}, S^{1}\underline{x}_{N}, \ldots, S^{m-1}\underline{x}_{N}\}$. This backward estimate is then of the form

$$\hat{\underline{x}}_{N}[0,m-1] = -\sum_{k=0}^{m-1} b_{k} S^{k}\underline{x}_{N} \qquad \text{(A.3.12)}$$

and the backward error vector is defined by

$$\underline{b}_{N,m} = S^{m}\underline{x}_{N} - \hat{\underline{x}}_{N}[0,m-1] \qquad \text{(A.3.13)}$$

In the same manner as with forward prediction, by applying the projection theorem it can be shown that the backward estimate is given by

$$\hat{\underline{x}}_{N}[0,m-1] = P_{\underline{x}_{N}[0,m-1]}\, S^{m}\underline{x}_{N} \qquad \text{(A.3.14)}$$

The backward prediction error vector is then found to be

$$\underline{b}_{N,m} = P^{\perp}_{\underline{x}_{N}[0,m-1]}\, S^{m}\underline{x}_{N} \qquad \text{(A.3.15)}$$

and the corresponding minimum mean squared error is obtained by

$$f^{b}_{N,m} = \underline{b}^{T}_{N,m}\,\underline{b}_{N,m} = \underline{b}^{T}_{N,m}\, S^{m}\underline{x}_{N} \qquad \text{(A.3.16)}$$

A.3.3 Delayed Backward Prediction

The m-th order delayed backward prediction method is similarly defined to be that procedure of estimating the column vector $S^{m+1}\underline{x}_N$ by a linear combination of the set of vectors $\{S^1\underline{x}_N, S^2\underline{x}_N, \ldots, S^m\underline{x}_N\}$. It can be shown that the delayed backward estimate is given by

$$\hat{\underline{x}}_N[1,m] = P_{\underline{x}_N[1,m]}\, S^{m+1}\underline{x}_N \tag{A.3.17}$$

and the delayed backward error is obtained by

$$\underline{d}_{N,m} = P^{\perp}_{\underline{x}_N[1,m]}\, S^{m+1}\underline{x}_N \tag{A.3.18}$$

The corresponding minimum mean squared error is measured by

$$f^d_{N,m} = \underline{d}^T_{N,m}\,\underline{d}_{N,m} = \underline{d}^T_{N,m}\, S^{m+1}\underline{x}_N \tag{A.3.19}$$

A little thought will convince oneself that the projection operation $P_{\underline{x}_N[1,m]}$ can be expressed as

$$P_{\underline{x}_N[1,m]} = X_{\underline{x}_N[1,m]} \left[X^T_{\underline{x}_N[1,m]}\, X_{\underline{x}_N[1,m]} \right]^{-1} X^T_{\underline{x}_N[1,m]}$$

$$= \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \hline & & & \\ & X_{\underline{x}_{N-1}[0,m-1]} & & \end{bmatrix} \left[X^T_{\underline{x}_{N-1}[0,m-1]}\, X_{\underline{x}_{N-1}[0,m-1]} \right]^{-1} \begin{bmatrix} 0 & 0 & \cdots & 0 \\ \hline & & & \\ & X_{\underline{x}_{N-1}[0,m-1]} & & \end{bmatrix}^T \tag{A.3.20}$$

The relationship between the backward prediction error and the delayed backward prediction error is then readily found to be

$$\underline{d}_{N,m} = [0 \vdots \underline{b}_{N-1,m}]^T \tag{A.3.21}$$

It then follows that N^{th} delayed prediction error is equal to the $(N-1)^{st}$ backward prediction error

$$\underline{d}_{N,m}(N) = \underline{b}_{N-1,m}\ (N-1) \tag{A.3.22}$$

The relationship of forward, backward, and delayed backward is suggestively depicted in Figure A.1.

A.4 Decomposition of Subspaces

The development of a computational efficient algorithm is dependent on the decomposition of subspaces. Subspaces may be decomposed by appealing to the well known projection theorem (Luenberger, 1969). The formulae obtained in this section will be used for the development of order update recursions in Section A.5.

Since the forward prediction error $\underline{\varepsilon}_{N,m}$ lies in the subspace $M_{\underline{x}_N}[0,m]$ but is orthogonal to $M_{\underline{x}_N}[1,m]$, we can express $M_{\underline{x}_N}[0,m]$ as the direct sum of $M_{\underline{x}_N}[1,m]$ and $\{\underline{\varepsilon}_{N,m}\}$, that is

$$M_{\underline{x}_{N,m}}[0,m] = M_{\underline{x}_N}[1,m] \oplus \underline{\varepsilon}_{N,m} \tag{A.4.1}$$

where $\{\underline{\varepsilon}_{N,m}\}$ denotes the subspace spanned by the forward prediction error vector $\underline{\varepsilon}_{N,m}$. The projection operator on the subspace $\{\underline{\varepsilon}_{N,m}\}$ is defined by

$$P_{\underline{\varepsilon}_{N,m}} = \underline{\varepsilon}_{N,m} (\underline{\varepsilon}^T_{N,m} \underline{\varepsilon}_{N,m})^{-1} \underline{\varepsilon}^T_{N,m} \tag{A.4.2}$$

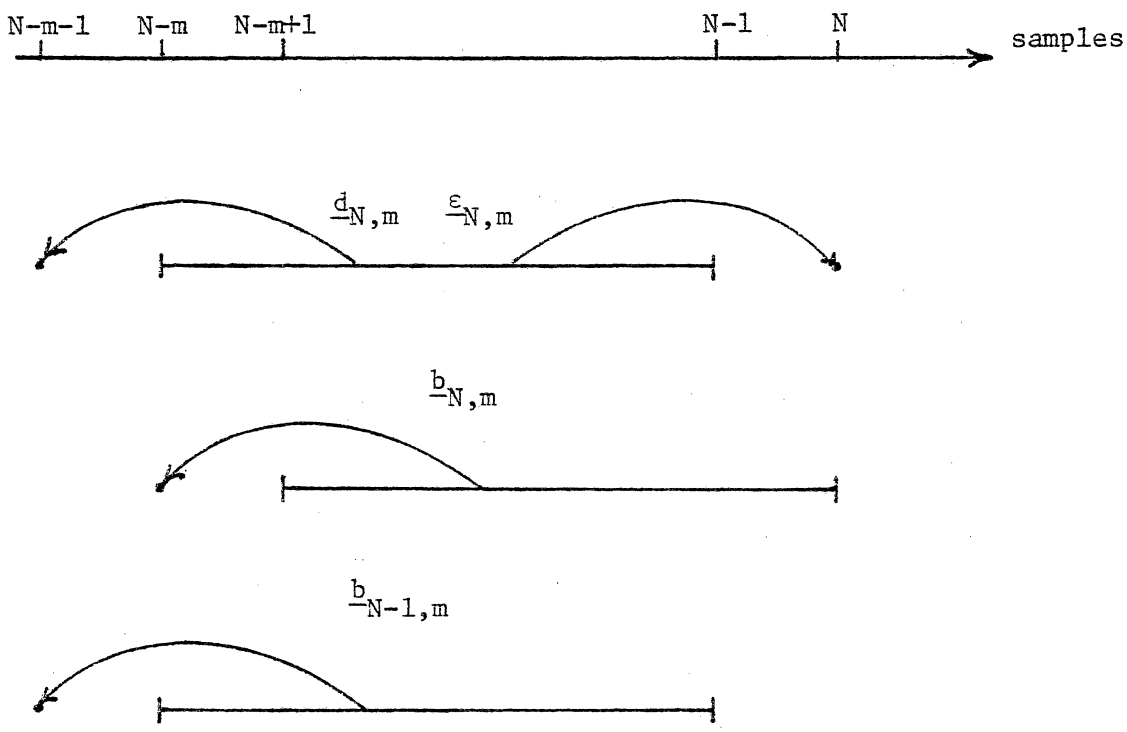


Fig. A.1 Forward, backward and delayed predictions

Relationship (A.4.1) can be readily shown to yield the following decomposition of the projection operator

$$P^{\perp}_{\underline{x}_N[0,m]} = (I - P_{\underline{\varepsilon}_{N,m}}) \, P^{\perp}_{\underline{x}_N[1,m]} \tag{A.4.3}$$

Similarly, since the delayed backward prediction error $\underline{d}_{N,m}$ lies in the subspace $M_{\underline{x}_N[1,m+1]}$ but is orthogonal to $M_{\underline{x}_N[1,m]}$, we obtain

$$M_{\underline{x}_N[1,m+1]} = M_{\underline{x}_N[1,m]} \oplus \{\underline{d}_{N,m}\} \tag{A.4.4}$$

where $\{\underline{d}_{N,m}\}$ denotes the subspace spanned by the backward prediction error vector. The projection operator on the subspace $\{\underline{d}_{N,m}\}$ is defined by

$$P_{\underline{d}_{N,m}} = \underline{d}_{N,m} \, (\underline{d}^T_{N,m} \, \underline{d}_{N,m})^{-1} \, \underline{d}^T_{N,m} \tag{A.4.5}$$

Relationship (A.4.4) is found to yield the following decomposition of the projection operator

$$P^{\perp}_{\underline{x}_N[1,m+1]} = (I - P_{\underline{d}_{N,m}}) \, P^{\perp}_{\underline{x}_N[1,m]} \tag{A.4.6}$$

A.5 Order Update Recursions

In this section, we describe the order update recursive formulae which recursively compute the optimum m+1-st order prediction error from the optimum m-th order prediction error. Expressions (A.4.3) and (A.4.6) play a central role in obtaining these order update recursions.

Let us first derive the order update recursion for the forward prediction error vector. Applying the projection operator (A.4.6) to the column vector $\underline{x}_N$ yields

$$\underline{\varepsilon}_{N,m+1} = (I - P\underline{d}_{N,m})\ \underline{\varepsilon}_{N,m} \tag{A.5.1}$$

Substituting expression (A.4.5) into this relationship then yields

$$\underline{\varepsilon}_{N,m+1} = \underline{\varepsilon}_{N,m} - \underline{d}_{N,m}\ (\underline{d}^T_{N,m}\ \underline{d}_{N,m})^{-1}\ \underline{d}^T_{N,m}\ \underline{\varepsilon}_{N,m} \tag{A.5.2}$$

Recalling expression (A.3.22), the order update recursion for the N-th forward prediction error is found to be

$$\underline{\varepsilon}_{N,m+1}(N) = \underline{\varepsilon}_{N,m}(N) - \Delta_{N,m+1}(f^b_{N-1,m})^{-1}\underline{b}_{N-1,m}(N-1) \tag{A.5.3}$$

where the partial-correlation coefficients are specified by

$$\Delta_{N,m+1} = \underline{d}^T_{m,N}\underline{\varepsilon}_{N,m} = \underline{x}^T_N\ P^{\perp}_{\underline{x}_N[1,m]}\ S^{m+1}\underline{x}_N \tag{A.5.4}$$

Expression (A.5.1) leads to

$$\underline{\varepsilon}^T_{N,m+1}\ \underline{\varepsilon}_{N,m+1} = \underline{\varepsilon}^T_{N,m}\ (I - P\underline{d}_{N,m})\ \underline{\varepsilon}_{N,m} \tag{A.5.5a}$$

The recursion for the forward minimum mean square error is similarly found to be

$$f^{\varepsilon}_{N,m+1} = f^{\varepsilon}_{N,m} - \Delta_{N,m+1}\ (f^b_{N-1,m})^{-1}\ \Delta_{N,m+1} \tag{A.5.5b}$$

Expressions (A.5.3) and (A.5.5b) constitute the order update recursion formulae for the forward prediction.

Next, we will find the order update recursion for the backward prediction error vector. Applying the projection operator (A.4.3) to the column vector $S^{m+1}\underline{x}_N$ is found to yield

$$\underline{b}_{N,m+1} = (I - P\underline{\varepsilon}_{N,m})\,\underline{d}_{N,m} \tag{A.5.6}$$

Substituting expression (A.4.2) into this relationship results in

$$\underline{b}_{N,m+1} = \underline{d}_{N,m} - \underline{\varepsilon}_{N,m}\,(\underline{\varepsilon}^T_{N,m}\,\underline{\varepsilon}_{N,m})^{-1}\,\underline{\varepsilon}^T_{N,m}\,\underline{d}_{N,m} \tag{A.5.7}$$

The order update recursion for the N-th backward prediction error is then specified by

$$\underline{b}_{N,m+1}(N) = \underline{b}_{N-1,m}(N-1) - \Delta_{N,m+1}\,(f^{\varepsilon}_{N,m})^{-1}\underline{\varepsilon}_{N,m}(N) \tag{A.5.8}$$

Expression (A.5.6) leads to

$$\underline{b}^T_{N,m+1}\,\underline{b}_{N,m+1} = \underline{d}^T_{N,m}\,(I - P\underline{\varepsilon}_{N,m})\,\underline{d}_{N,m} \tag{A.5.9}$$

The recursion for $f^b_{N,m}$ is next found to be

$$f^b_{N,m+1} = f^b_{N-1,m} - \Delta_{N,m+1}\,(f^{\varepsilon}_{N,m})^{-1}\,\Delta_{N,m+1} \tag{A.5.10}$$

The order update recursion formulae for the backward prediction are represented by relationships (A.5.8) and (A.5.10).

A.6 Time Update Recursions

As a new element of the time series is observed, the partial-correlation coefficients, forward least square errors, and the backward least square errors can be computed recursively by using the knowledge

of these parameters from the last time instant. This being the case, these parameters are said to be "time updated" for each new data point. These update recursions are obtained by utilizing a method referred to as projection operator decomposition.

For the spectral estimation problem considered here, we decompose the projection operator $P_{\underline{x}_N[i,m]}$ into one that projects on all past observations and another that generates the correction due to a new observation x(N). First, we define the component projection matrix P_N by

$$P_N = e_N e_N^T \tag{A.6.1}$$

where e_N is the N x 1 unit basis vector expressed by

$$e_N = [0, \ldots, 0, 1]^T \tag{A.6.2}$$

Let us define the column vectors

$$\underline{x}_{p_N} = P_N \underline{x}_N = [0, \ldots, 0, x(N)]^T \tag{A.6.3}$$

$$\underline{x}_{p_N}^{\perp} = P_N^{\perp} \underline{x}_N = [x(1), \ldots, x(N-1), 0]^T \tag{A.6.4}$$

Note that $\underline{x}_{p_N}^T \underline{y}_{p_N} = \underline{x}_{p_N}^T \underline{y}_N = \underline{x}_N^T \underline{y}_{p_N}$ and similarly for $\underline{x}_{p_N}^{\perp T} \underline{y}_{p_N}^{\perp}$. The projection of $\underline{x}_N$ on the subspace $M_{\underline{x}_N[i,m]}$ is now decomposed by component projection matrix P_N to obtain

$$P_{\underline{x}_N[i,m]} \underline{x}_N = P_{\underline{x}_N[i,m]} \underline{x}_{p_N} + P_{\underline{x}_N[i,m]} \underline{x}_{p_N}^{\perp} \tag{A.6.5}$$

Multiplication of $(I-P_N)$ and the matrix $\underline{\underline{X}}_{\underline{x}N}[i,m]$ yields the so-called oblique matrix

$$\underline{\underline{C}}_{\underline{x}N}[i,m] = (I - P_N)\ \underline{\underline{X}}_{\underline{x}N}[i,m] \tag{A.6.6}$$

whose last row is the zero row vector. We define the oblique projection operator to be

$$\underline{\underline{Q}}_{\underline{x}N}[i,m] = \underline{\underline{X}}_{\underline{x}N}[i,m] \left[\underline{\underline{C}}^{T}_{\underline{x}N}[i,m]\ \underline{\underline{C}}_{\underline{x}N}[i,m]\right]^{-1} \underline{\underline{C}}^{T}_{\underline{x}N}[i,m] \tag{A.6.7}$$

and its associated orthogonal complement by

$$\underline{\underline{Q}}^{\perp}_{\underline{x}N}[i,m] = I - \underline{\underline{Q}}_{\underline{x}N}[i,m] \tag{A.6.8}$$

Upon inspection of expression (A.6.7), we see that the application of the oblique projection operator to the vector $\underline{x}_N$ implicitly possesses the solution of the prediction coefficients at the N-1st stage.

After simple algebraic manipulation, relationship (A.6.5) can be expressed as

$$\underline{\underline{P}}_{\underline{x}N}[i,m]\ \underline{x}_N = \underline{\underline{Q}}_{\underline{x}N}[i,m]\ \underline{x}_N + \underline{\underline{P}}_{\underline{x}N}[i,m]\ P_N \underline{\underline{Q}}^{\perp}_{\underline{x}N}[i,m]\ \underline{x}_N \tag{A.6.9}$$

The orthogonal complement projection of $\underline{x}_N$ can be expressed as

$$\underline{\underline{P}}^{\perp}_{\underline{x}N}[1,m]\ \underline{x}_N = \underline{x}_N - \underline{\underline{Q}}_{\underline{x}N}[i,m]\ \underline{x}_N - \underline{\underline{P}}_{\underline{x}N}[i,m]\ P_N\ \underline{\underline{Q}}^{\perp}_{\underline{x}N}[i,m]\ \underline{x}_N \tag{A.6.10a}$$

which can be further developed to the form

$$\overset{\perp}{P}\underline{x}_N[i,m]\ \underline{x}_N = \underline{x}_N - {}^{Q}\underline{x}_N[i,m]\ \underline{x}_N - P_N\ \overset{\perp}{{}^{Q}\underline{x}}_N[i,m]\ \underline{x}_N$$

$$+ \overset{\perp}{P}\underline{x}_N[i,m]\ P_N\ \overset{\perp}{{}^{Q}\underline{x}}_N[i,m]\ \underline{x}_N \qquad \text{(A.6.10b)}$$

Considering the relationships (A.6.3), (A.6.4) and (A.6.7), we obtain

$$\overset{\perp}{P}\underline{x}_N[i,m]\ \underline{x}_N = \begin{bmatrix} \overset{\perp}{P}\underline{x}_{N-1}[i,m]\ \underline{x}_{N-1} \\ \hline 0\ 0\ .\ .\ .\ .\ .\ 0 \end{bmatrix} + \overset{\perp}{P}\underline{x}_N[i,m]\ P_N\ \overset{\perp}{{}^{Q}\underline{x}}_N[i,m]\ \underline{x}_N$$

(A.6.11)

Premultiplying $[S^{m+1}\ \underline{x}_N]^T$ on both sides of expression (A.6.11) gives the time update recursions of the partial reflection coefficients

$$\Delta_{m+1,N} = \Delta_{m+1,N-1} + [S^{m+1}\underline{x}_N]^T\ \overset{\perp}{P}\underline{x}_N[1,m]\ P_N\ \overset{\perp}{{}^{Q}\underline{x}}_N[1,m]\ \underline{x}_N$$

(A.6.12)

where i was taken to be 1. Furthermore, operation of the component projection operator P_N on both sides of expression (A.6.10a) yields

$$P_N\ \overset{\perp}{P}\underline{x}_N[i,m]\ \underline{x}_N = P_N\ \overset{\perp}{{}^{Q}\underline{x}}_N[i,m]\ \underline{x}_N - e_N e_N^T\ P\underline{x}_N[i,m]\ e_N e_N^T\ \overset{\perp}{{}^{Q}\underline{x}}_N[i,m]\ \underline{x}_N$$

(A.6.13a)

$$= P_N\ \overset{\perp}{{}^{Q}\underline{x}}_N[i,m]\ \underline{x}_N \left[1 - e_N^T\ P\underline{x}_N[i,m]\ e_N\right]^{-1} \qquad \text{(A.6.13b)}$$

Thus we obtain the relationship

$$P_N\ \overset{\perp}{{}^{Q}\underline{x}}_N[i,m]\ \underline{x}_N = \frac{1}{1 - \gamma_{i,m,N}}\ P_N\ P\underline{x}_N[i,m]\ \underline{x}_N \qquad \text{(A.6.14)}$$

where

$$\gamma_{i,m,N} = e_N^T P_{\underline{x}_N[i,m]} e_N \tag{A.6.15}$$

Directly substituting (A.6.14) into (A.6.12), we see that

$$\Delta_{N,m+1} = \Delta_{N-1,m+1} + \frac{\left[S^{m+1}\underline{x}_N\right]^T P_{\underline{x}_N[1,m]}^{\perp} e_N e_N^T P_{\underline{x}_N[1,m]}^{\perp} \underline{x}_N}{1 - \gamma_{1,m,N}} \tag{A.6.16}$$

which simplifies to the form

$$\Delta_{N,m+1} = \Delta_{N-1,m+1} + \frac{\underline{b}_{N-1,m}(N-1)\underline{\varepsilon}_{N,m}(N)}{1 - \gamma_{1,m,N}} \tag{A.6.17}$$

Similarly, the time-update for $f_{N,m}^{\varepsilon}$ and $f_{N,m}^{r}$ can be obtained as

$$f_{N,m}^{\varepsilon} = f_{N-1,m}^{\varepsilon} + \frac{\underline{\varepsilon}_{N,m}^2(N)}{1 - \gamma_{1,m,N}} \tag{A.6.18}$$

$$f_{N,m}^{r} = f_{N-1,m}^{r} + \frac{\underline{b}_{N,m}^2(N)}{1 - \gamma_{0,m-1,N}} \tag{A.6.19}$$

where

$$\gamma_{0,m-1,N} = e_N^T P_{\underline{x}_N[0,m-1]} e_N \tag{A.6.20}$$

Thus we can use equation (A.6.17) to update the partial reflection coefficients. Equations (A.6.18) and (A.6.19) can be used to update forward and backward prediction errors, respectively.

A.8 Summary

A recursive algorithm has been presented for efficiently obtaining an autoregressive (AR) spectral estimate. To achieve a significant computational improvement, prewindowing was applied, and projection operators were utilized in the vector space setting. Normalizations of the order and time update algorithm yields more computational advantage than the unnormalized method. Interested reader may refer to (Lee and Morf, 1980) and (Friedlandar, 1980).

Appendix B

ADAPTIVE SPECTRAL ESTIMATION

B.1 Introduction

In this chapter, we will discuss two adaptive techniques, namely, the Widrow-Hoff algorithm (Widrow and Hoff, 1960) and the Iterative LMS method. It is well known that the Widrow-Hoff algorithm is a recursive technique which updates parameters with the arrival of each new data sample. At each recursion, parameters are algorithmically selected in a least squares sense. As the number of data samples increases, the model's parameters "may" converge to the least square solution which is also known as the Wiener solution (Wiener, 1949). Primary reason for utilizing the Widrow-Hoff algorithm is computational in nature. As each new data point is obtained, only $0(p)$ computations are required to update the model's parameters.

The Iterative LMS method is a technique which updates the solution for the linear system of equations which approximates the Wiener equations (Wiener, 1949). Although the number of computations for the Iterative LMS method to update parameters at every new data point is $0(p^2)$, the Iterative LMS method gives the exact solution to a given linear system of equations. To compare these two techniques, a number of examples are presented.

B.2 Widrow-Hoff Algorithm

The analysis of an adaptive filter can be developed by considering the linear configuration shown in Fig. B.2.1. An adaptive filter is composed of a tapped delay line, adjustable weights and summers. Delayed signals which are real valued are weighted and summed to form an output signal $\hat{d}(n)$ which designates an estimate for the desired signal $d(n)$. At the n-th observation, a set of delayed signals can be formulated in a vector form

$$\underline{x}_n = [x(n-1),\ x(n-2),\ \ldots\ ,\ x(n-p)]^T \tag{B.2.1}$$

where $\underline{x}_n$ is a pxl column vector. It is also convenient to denote the adjustable weights at the n-th iteration by

$$\underline{h}_n = [h_n(1),\ h_n(2),\ \ldots\ ,\ h_n(p)]^T \tag{B.2.2}$$

where $\underline{h}_n$ is a pxl column vector. The estimate of the value of $d(n)$ based on the vector (B.2.1) will be taken to be the linear combination

$$\begin{aligned}\hat{d}(n) &= \underline{h}_n^T \underline{x}_n \\ &= \sum_{k=1}^{p} h_n(k)\, x(n-k)\end{aligned} \tag{B.2.3}$$

The error between the desired signal and the estimate at the n-th sample is given by

$$\varepsilon(n) = d(n) - \underline{h}_n^T \underline{x}_n \tag{B.2.4}$$

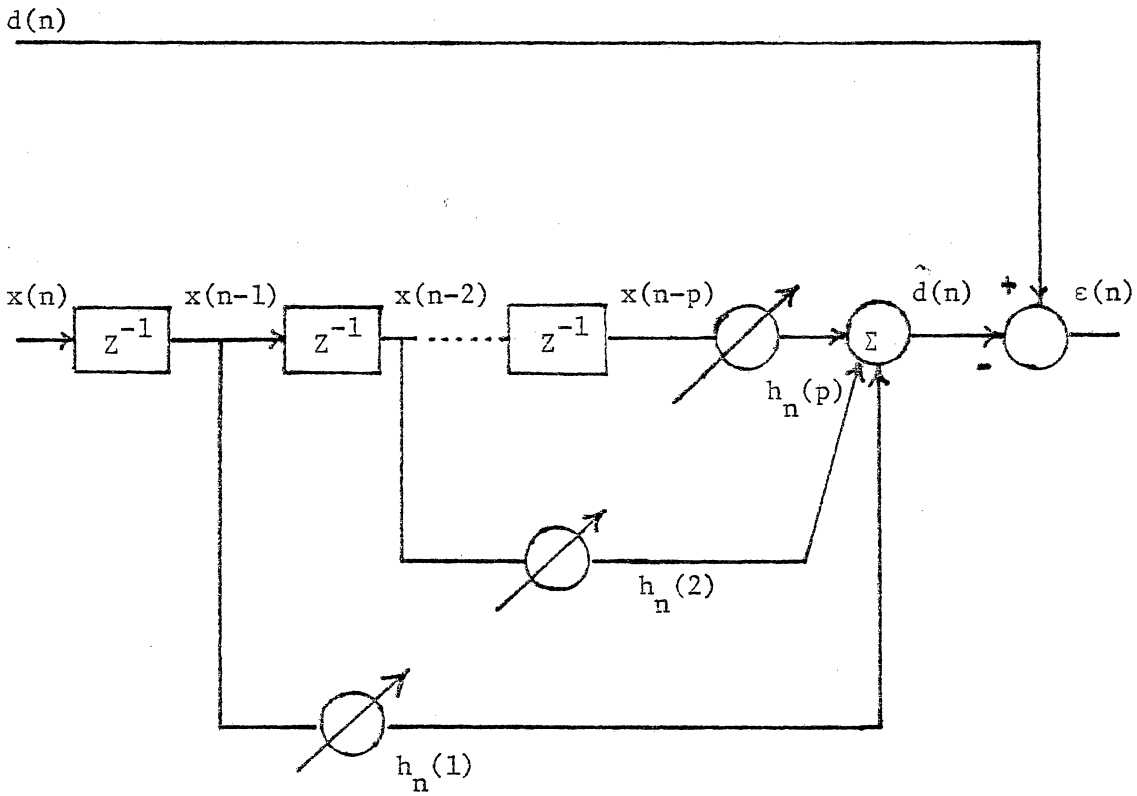


Fig. B.2.1 Adaptive Linear Configuration

The associated mean square error is defined by

$$f(\underline{h}_n) = E[\varepsilon^2(n)] \tag{B.2.5}$$

Substitution of (B.2.4) into (B.2.5) is found to yield

$$f(\underline{h}_n) = \phi_{dd}(0) - 2\, \underline{r}_{dx}^T\, \underline{h}_n + \underline{h}_n^T\, R_x\, \underline{h}_n \tag{B.2.6}$$

where $\phi_{dd}(0)$ is the variance of the desired signal d(n), that is,

$$\phi_{dd}(0) = E[d^2(n)] \tag{B.2.7}$$

while r_{dx} and R_x are the pxl cross correlation vector and the pxp covariance matrix, respectively, defined by

$$r_{dx} = [\phi_{dx}(1),\ \phi_{dx}(2),\ \ldots\ ,\ \phi_{dx}(P)]^T \tag{B.2.8a}$$

and

$$R_x = \begin{bmatrix} \phi_{xx}(0), & \phi_{xx}(1), & \ldots & \phi_{xx}(p-1) \\ \phi_{xx}(1), & \phi_{xx}(0), & \ldots & \phi_{xx}(p-2) \\ \vdots & & & \vdots \\ \phi_{xx}(p-1), & \phi_{xx}(p-2), & \ldots & \phi_{xx}(0) \end{bmatrix} \tag{B.2.8b}$$

in which $\phi_{dx}(i)$ is the cross-correlation sequence between the individual input signal component and the desired signal defined by

$$\phi_{dx}(i) = E[x(n+i)\ d(n)] \tag{B.2.8c}$$

and $\phi_{xx}(i)$ denotes the autocorrelation sequence of the input signal specified by

$$\phi_{xx}(i) = E[x(n + i)\ x(n)] \tag{B.2.8d}$$

It may be observed from expression (B.2.6) that the mean-square error is precisely a second order function of the weights $\underline{h}_n$ and is visualized as a parabolic function of the weight variables. The adaptive process seeks the minimizing weight variable selection by using the well-known method of steepest descent.

In seeking the minimum mean-square error by the method of steepest descent, one first begins with an initial guess of the model's weight parameters. The next estimate is then obtained from that estimate by making a change in the weight vector in the direction of the negative of the gradient vector. The gradient is obtained by differentiating expression (B.2.6) to yield

$$\nabla f(\underline{h}_n) = -2\ \underline{r}_{dx} + 2\ R_x\ \underline{h}_n \tag{B.2.9}$$

If each change in the weight vector is made proportional to the negative of the gradient, the method of steepest descent leads to the following recursive relationship

$$\underline{h}_{n+1} = \underline{h}_n + \mu \nabla f(\underline{h}_n) \tag{B.2.10}$$

For a sufficiently small value of μ, the mean-square error at the $(n + 1)$-st step is approximately found to be

$$f(\underline{h}_{n+1}) \doteq f(\underline{h}_n) - 2\ \mu^2 || \nabla f(\underline{h}_n) ||^2 \tag{B.2.11a}$$

where $||\nabla f(\underline{h}_n)||^2$ is the positive scalar defined by

$$||\nabla f(\underline{h}_n)||^2 = [\nabla f(\underline{h}_n)]^T [\nabla f(\underline{h}_n)] \qquad \text{(B.2.11b)}$$

It may be observed from eq. (B.2.11a) that the mean-square error is reduced with each change of the weight vector. For a proper choice of μ, it has been claimed that this algorithm will converge to an optimum point regardless of the initial weights. (Widrow, 1971)

The method of steepest descent requires the determination of the gradient vector. In practice, the true values of these gradients are seldom available. To overcome this difficulty, the "LMS algorithm" offers a practical procedure for implementing the method of steepest descent. This algorithm uses gradient estimates in place of true gradient values. These estimates may be "noisy" (i.e., contain errors) but the effect of the gradient-measurement errors is observed to be small in many practical applications.

A method of measuring gradients of the mean square error which does not require squaring, averaging or differentiating is now given. The mean square error $f(\underline{h}_n)$ may be represented crudely by the single sample $\varepsilon(n)$, the square of the n-th error value. Then the gradient vector is approximated by

$$\nabla f(\underline{h}_n) \doteq \nabla \varepsilon^2(n) = -2\varepsilon(n)\, \underline{x}_n \qquad \text{(B.2.12)}$$

In order to approximate the gradient vector, the present input-signal $\underline{x}_n$ and its associated scalar error $\varepsilon(n)$ are used. Upon taking an expected value on expression (B.2.12), expression (B.2.9) can be

obtained.

An adaptation cycle will proceed with the arrival of each new input vector. From eqs. (B.2.10) and (B.2.12), the adaptation procedure comprising the LMS algorithm is completely represented by (B.2.13).

$$\varepsilon(n) = d(n) - \underline{h}_n^T \underline{x}_n \tag{B.2.13a}$$

$$\underline{h}_{n+1} = \underline{h}_n - 2\mu\varepsilon(n)\ \underline{x}_n \tag{B.2.13b}$$

Upon examination of expressions (B.2.13), we can see that the computational requirement is O(p). In this algorithm, the selection of μ is also an important factor. If μ is made too small, convergence is slow. On the other hand, if μ is selected to be too large, the adaptive method may not converge. In terms of selecting a best μ, the interested reader may refer to (Widrow, 1971; Luenberger, 1973; Huffman and Nolte, 1980).

B.3 Iterative LMS Method

We will now investigate the problem of how to linearly filter an observed, wide-sense stationary, discrete-time, random time series $\{x(n)\}$. Our primary interest is to best estimate the desired discrete-time random time series $\{d(n)\}$ in the minimum mean square sense. The problem is illustrated in Fig. B.3.1. Our objective is to find the transfer function H(z) that minimizes the mean square error. We assume that the estimate of element d(n) is of the form

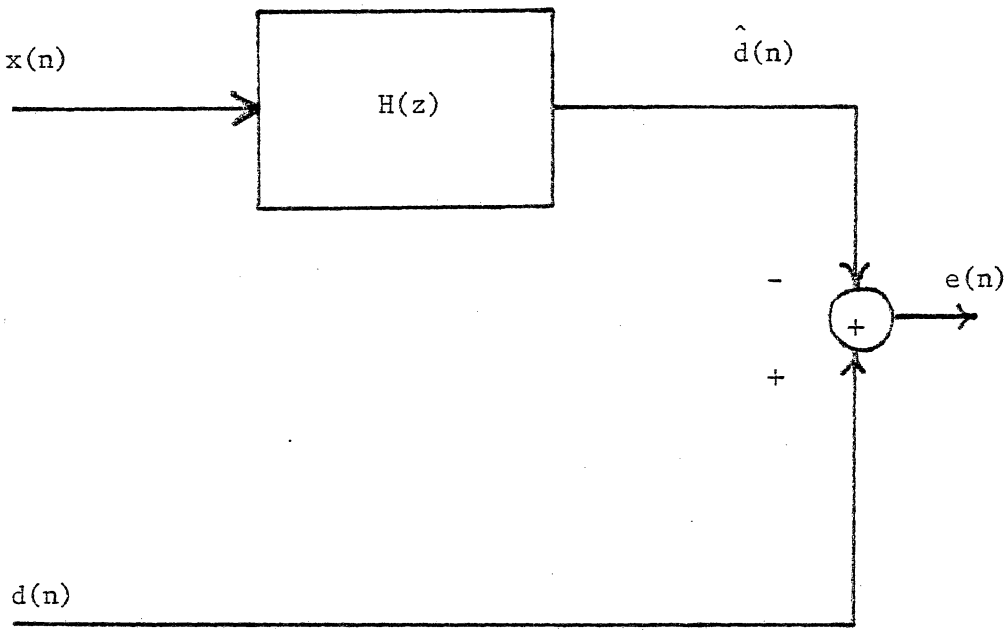


Fig. B.3.1 Pictorial representation of the optimum filtering

$$\hat{d}(n) = \sum_{k=0}^{p-1} h(k)\, x(n-k) \qquad (B.3.1)$$

where h(k) are the filter weight elements. The estimate $\hat{d}(n)$ is then seen to be a linear combination of the most recent p values of the observation signal. The mean square error is found to be a function of filter weights h(k) and is specified by

$$f(\underline{h}) = E[\{d(n) - \hat{d}(n)\}^2] \qquad (B.3.2)$$

where $\underline{h}$ is the pxl column vector defined by

$$\underline{h} = [h(0), h(1), \ldots, h(p-1)]^T \qquad (B.3.3)$$

Substitution of expression (B.3.1) into (B.3.2) and taking the expected value operation yields

$$f(\underline{h}) = r_d(0) - 2\,\underline{h}^T\,\underline{r}_{dx} + \underline{h}^T\,R_x\,\underline{h} \qquad (B.3.4)$$

where $r_d(0) = E[d^2(n)]$, $\underline{r}_{dx}$ is the pxl column vector whose k-th element is given by $E[d(n)\,x(n-k)]$ for $k = 1, 2, \ldots, p$ and R_x is the pxp matrix whose elements are given by $R_x(i,j) = E[x(n-i)\,x(n-j)]$ (see eq. B2.6).

The optimum filter weights vector is readily determined by taking the gradient of quadratic functional (B.3.4) with respect to $\underline{h}$ and setting this gradient equal to the zero vector. This is found to result in the well-known Wiener vector selection (Wiener, 1949).

$$\underline{h}^o = R_x^{-1}\,\underline{r}_{dx} \qquad (B.3.5)$$

Although this approach is indeed attractive and typically results in satisfactory performance, it suffers one serious drawback. Its implementation requires apriori covariance knowledge which is usually lacking in many typical applications.

In order to achieve our object without requiring any statistical information, we introduce an estimation error criterion defined by

$$f_N(\underline{h}) = \sum_{k=p}^{N} [d(k) - \hat{d}(k)]^2 \qquad (B.3.6)$$

It will be beneficial to represent this error criterion in a vector format. Let us define the (N + 1 - p) x 1 estimation error vector

$$\underline{e}_N = \begin{bmatrix} d(p) \\ d(p+1) \\ \cdot \\ \cdot \\ \cdot \\ d(N) \end{bmatrix} - \begin{bmatrix} x(p) & x(p-1) & \cdots & x(1) \\ x(P+1) & x(p) & \cdots & x(2) \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ \cdot & & & \cdot \\ x(N) & x(N-1) & \cdots & x(N+1-p) \end{bmatrix} \begin{bmatrix} h(o) \\ h(1) \\ \cdot \\ \cdot \\ \cdot \\ h(p-1) \end{bmatrix} \qquad (8.3.7)$$

which can be compactly expressed

$$\underline{e}_N = \underline{d}_N - X_N \underline{h} \qquad (B.3.8)$$

Using these expressions, the square error criterion can be represented by

$$f_N(\underline{h}) = (\underline{d}_N - X_N \underline{h})^T (\underline{d}_N - X_N \underline{h}) \qquad (B.3.9)$$

Minimization of the functional (B.3.9) is straightforwardly carried out by setting the gradient $\nabla_{\underline{h}} f_N(\underline{h})$ equal to zero and yields the

following result.

$$\underline{h}_N^o = [X_N^T X_N]^{-1} X_N^T \underline{d}_N \tag{B.3.10}$$

In general applications, the use of this method is not practical since it requires on the order of p^3 multiplications to invert the pxp matrix $[X_N^T X_N]$. We will next discuss a straightforward procedure which reduce this computational complexity.

Upon examination of relationship (B.3.7) and (B.3.8), we can see that when the new data element x(N + 1) is provided, the equation error can be updated by

$$\underline{e}_{N+1} = \underline{d}_{N+1} - X_{N+1} \underline{h} \tag{B.3.11}$$

$$= \begin{bmatrix} \underline{d}_N \\ d(N+1) \end{bmatrix} - \begin{bmatrix} X_N \\ \underline{x}_{N+1}^T \end{bmatrix} \underline{h} \tag{B.3.12}$$

where $\underline{x}_{N+1}$ is the px1 column vector specified by

$$\underline{x}_{N+1} = [x(N+1), x(N), \ldots , x(N-p+2)]^T \tag{B.3.13}$$

It is clear from relationship (B.3.10) that we have to invert the matrix

$$[X_{N+1}^T X_{N+1}] = X_N^T X_N + \underline{x}_{N+1} \underline{x}_{N+1}^T \tag{B.3.14}$$

The following recursive relationship may be used to efficiently update the required matrix inverse

$$[X_{N+1}^T X_{N+1}]^{-1} = [X_N^T X_N]^{-1} - \frac{1}{1 + \underline{x}_{N+1}^T \underline{y}_{N+1}} \underline{y}_{N+1} \underline{y}_{N+1}^T \tag{B.3.15}$$

where

$$\underline{y}_{N+1} = [X_N^T X_N]^{-1} \underline{x}_{N+1} \tag{B.3.16}$$

After a few simple manipulations, the following recursion is obtained

$$\underline{h}_{N+1}^o = \underline{h}_N^o + \frac{d(N+1) - \underline{x}_{N+1}^T \underline{h}_N^o}{1 + \underline{x}_{N+1}^T \underline{y}_{N+1}} \underline{y}_{N+1} \tag{B.3.17}$$

Recursive relationships (B.3.16) and (B.3.17) constitute a more computationally efficient method than the direct approach (B.3.10). It can be shown that the computational complexity is of the order p^2.

B.4 Numerical Examples

In this section, we shall demonstrate the performance of two adaptive methods, namely, the Widrow-Hoff algorithm and the Iterative LMS method. This will be accomplished by investigating the time series whose elements are given by

$$x(n) = \sqrt{20} \sin (0.1 \pi n) + w(n) \tag{B.4.1}$$

where w(n) is a white Gaussian noise with variance one. The normarized Weiner equation error can be defined by

$$\xi(n) = \frac{||R_x \underline{h}_n - \underline{r}_{dx}||}{||\underline{r}_{dx}||} \tag{B.4.2}$$

where R_x is the pxp covariance matrix of the sequence {x(n)} and $\underline{r}_{dx}$ is the px1 cross-correlation vector of the sequences {d(n)} and {x(n)}. The above scalar value $\xi(n)$ yields a normalized measure of how closely the Wiener equations are being approximated. All graphs except Fig. B.4.4 provide the plot of normalized Wiener equation error referring to expression (B.4.2) versus iteration number (i.e., the number of observation data). The desired signal d(n) is specifically chosen to be x(n+1). This yields a problem of predicting one step into the future. Unless specified, the covariance matrix is initialized at 15-th iteration number.

It can be observed from Fig. B.4.1 that the normalized Wiener equation error of the Iterative LMS method converges to approximately zero after 2300 iterations, however, the Widrow-Hoff algorithm with $\mu = 0.001$ fails to converge. In the Widrow-Hoff algorithm, the value of μ was next selected to be .0001 and .01 in Fig. B.4.2 and Fig. B.4.3, respectively. As we can see on Fig. B.4.2, both of the adaptive algorithms converge reasonably close to zero. The Iterative LMS method converges faster than the Widrow-Hoff algorithm. Fig. B.4.3 illustrates an example which shows convergence behavior of the Iterative LMS method and nonconvergence behavior of Widrow-Hoff algorithm. The normalized square error $||h_n - h_0||/||h_0||$ where h_0 is the exact solution of the matrix equation (B.3.5) are displayed in Fig. B.4.4. The convergence behavior of the Iterative LMS and the nonconvergence behavior of the Wiener-Hoff are evident.

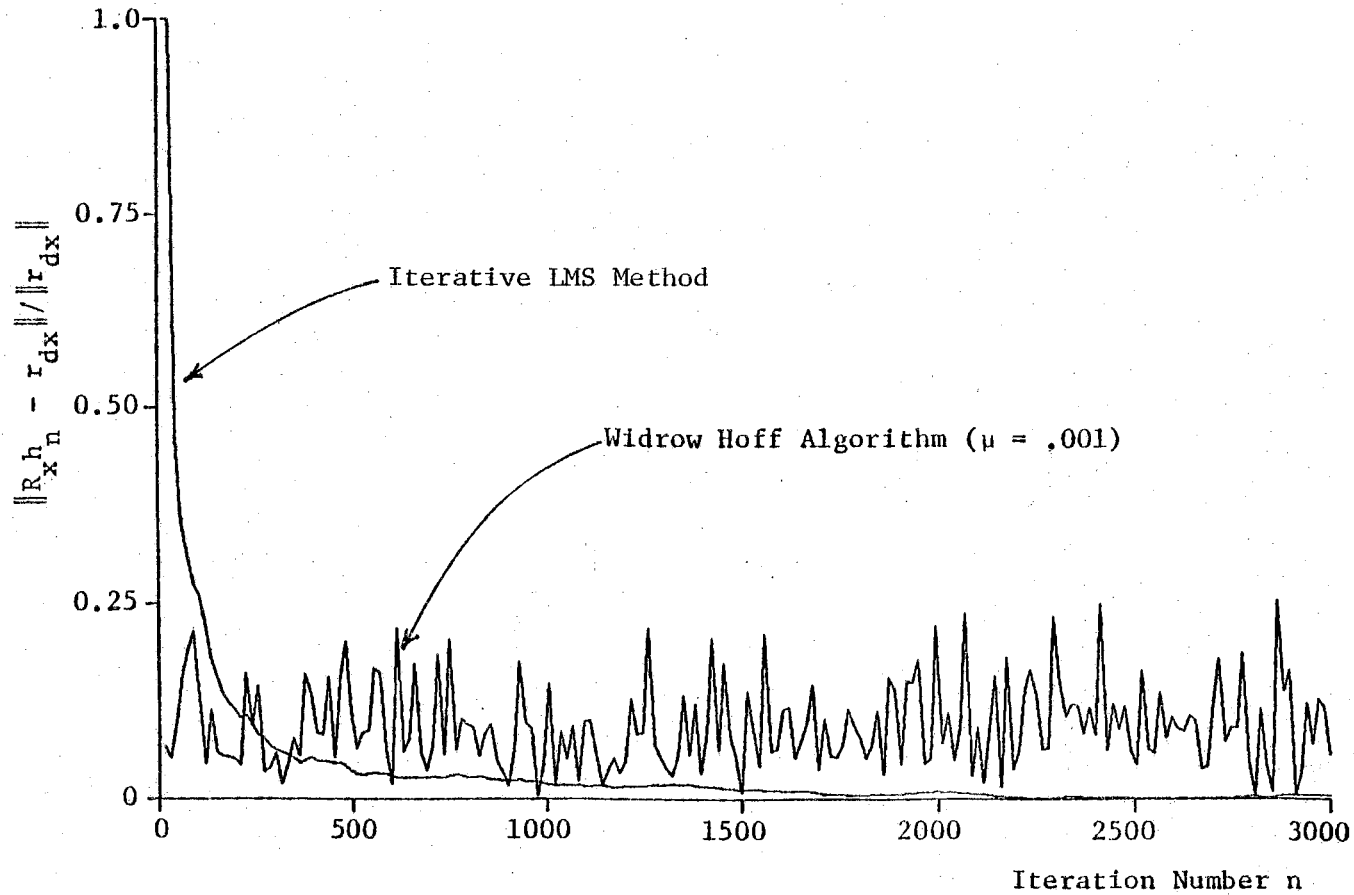


Fig. B.4.1 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 15)

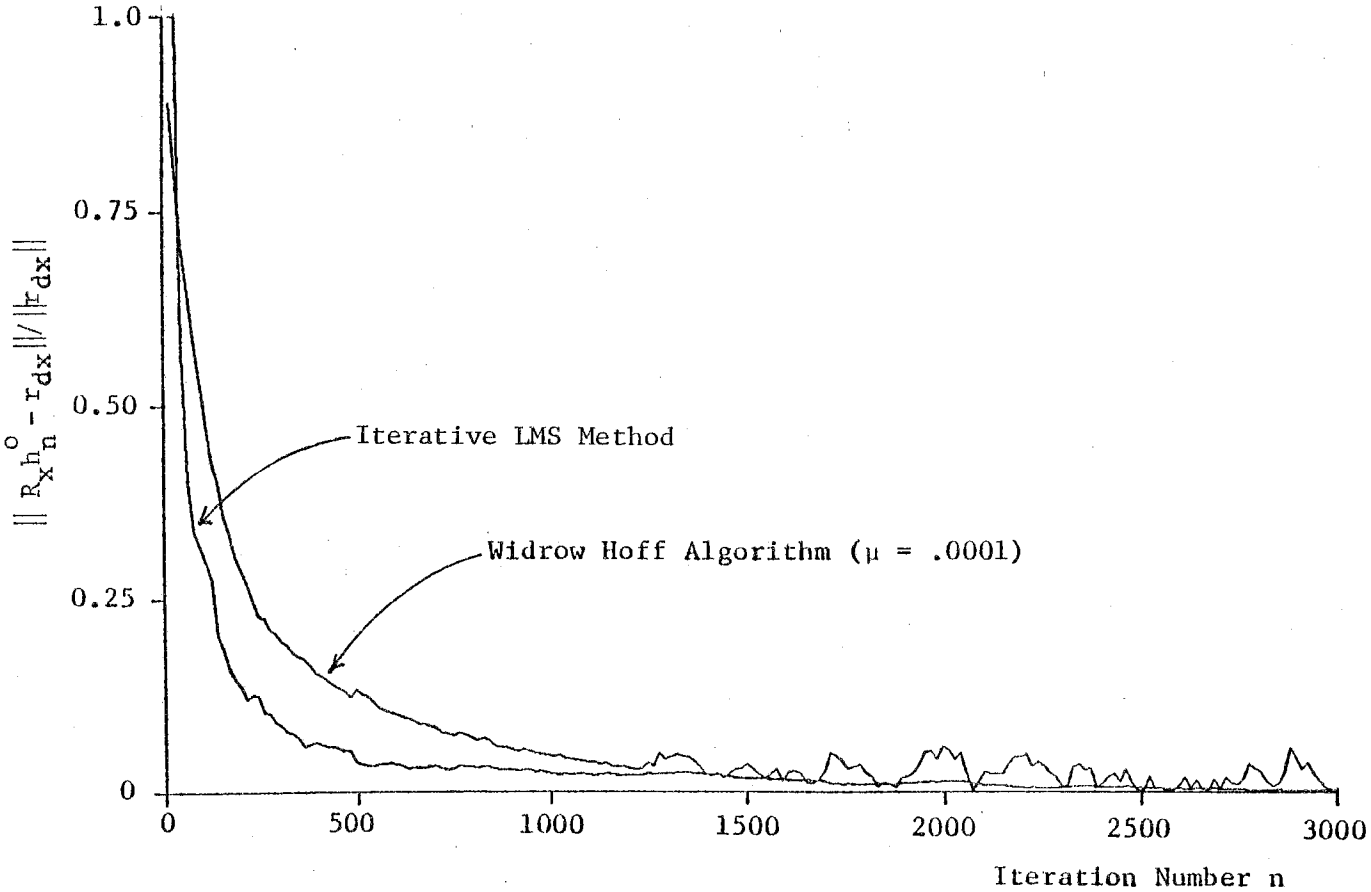


Fig. B.4.2 Normalized Weiner equation Error for Two Adaptive Algorithms (p = 4)

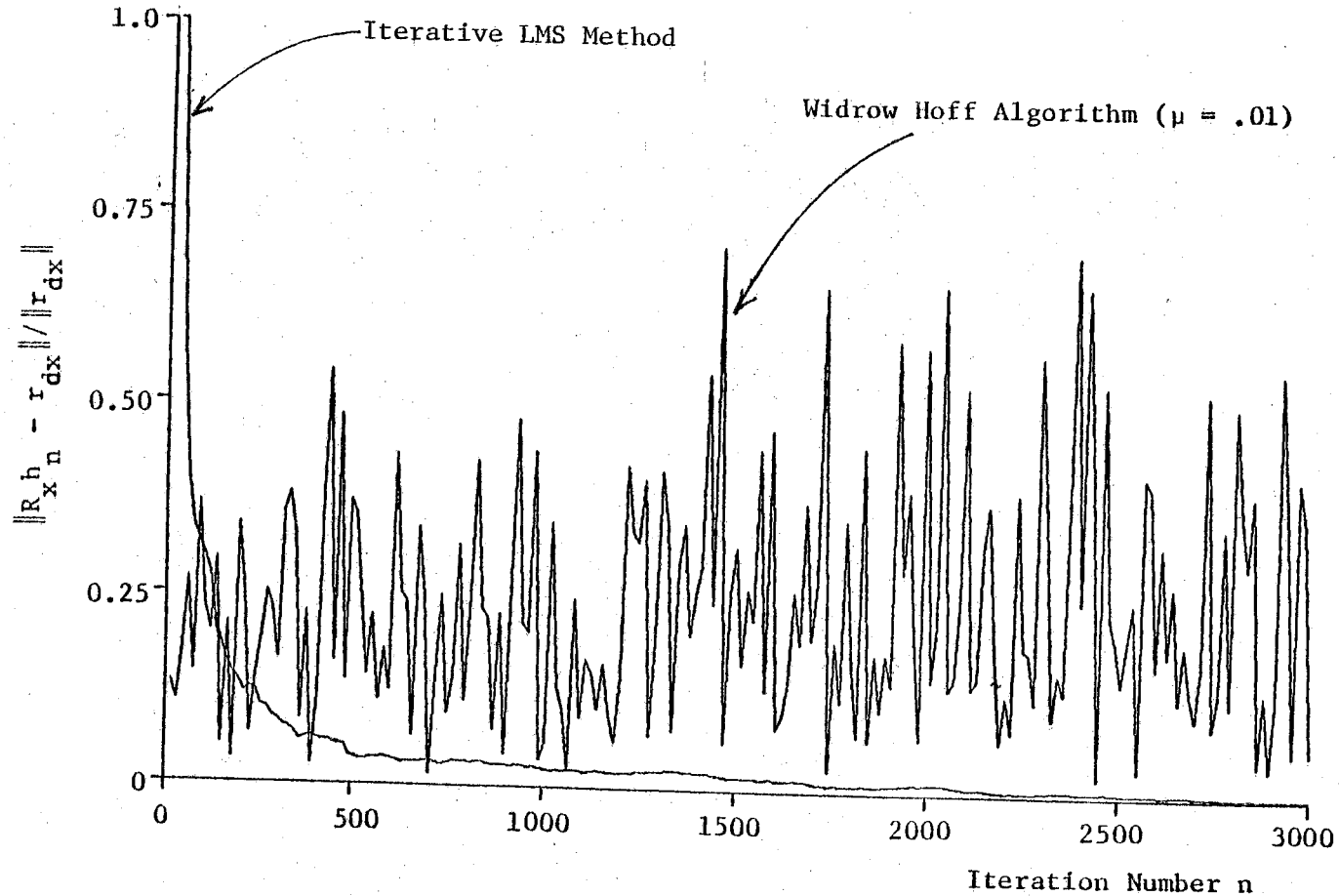


Fig. B.4.3 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 4) for

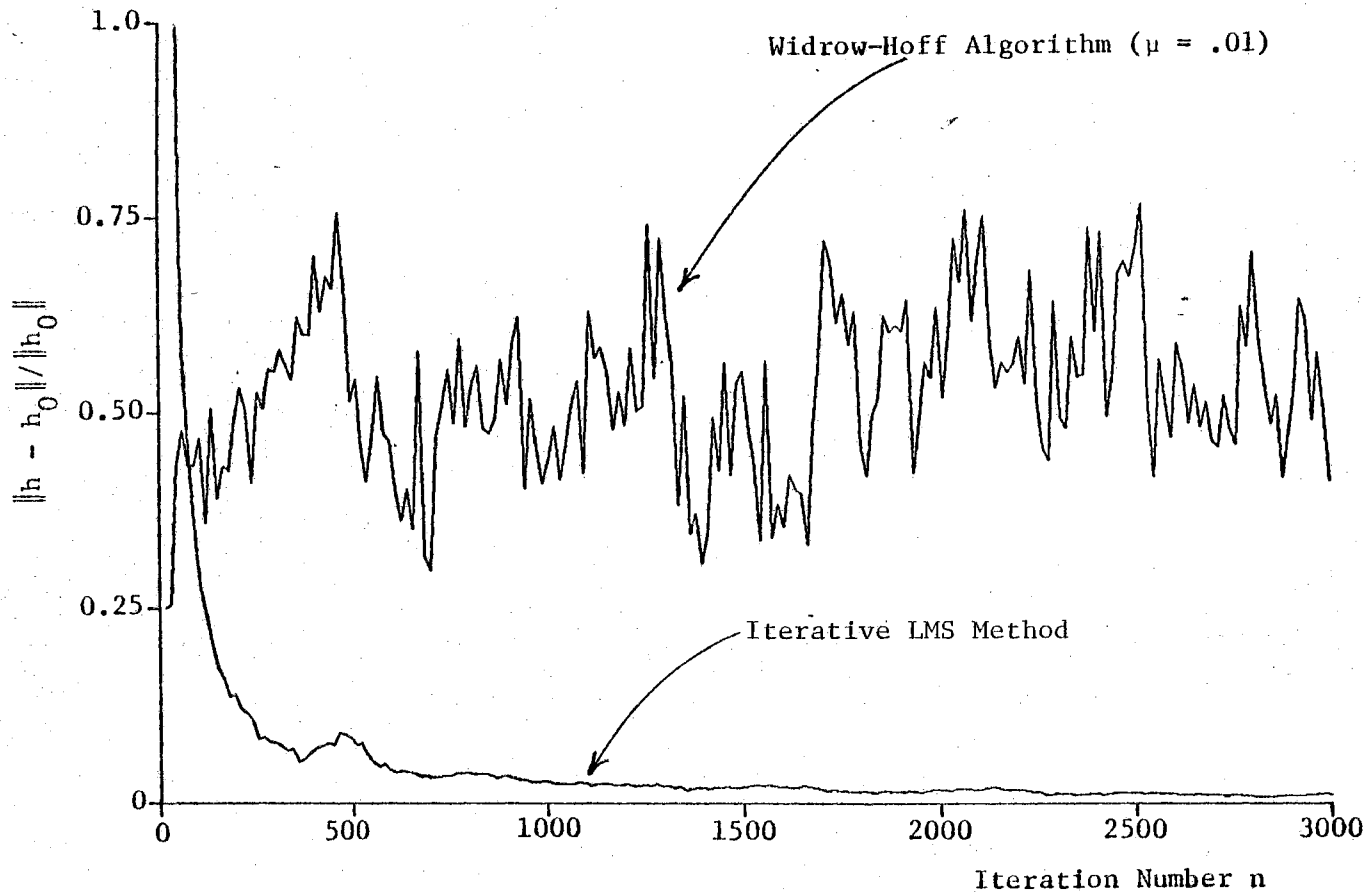


Fig. B.4.4 Normalized Squared Error for Two Adaptive Algorithms (p = 4)

Fig. B.4.5 and Fig. B.4.6 display the method which employs biased estimates for the approximation of covariance matrix elements. Fig. B.4.7 and Fig. B.4.8 display the method which uses unbiased estimates for the approximation of covariance matrix elements. Both the biased and unbiased methods converge to zero, however, the biased method starts with slightly large values of normalized Wiener equation error. Fig. B.4.9 and Fig. B.4.10 show the Iterative LMS method whose initial covariance matrix is the identity matrix. Although the normalized Weiner equation error at the early stage of iteration number are relatively large, this method also converged to zero. Fig. B.4.11 and Fig. B.4.12 display the direct method. Upon examination of Fig. B.4.5 through Fig. B.4.12, the direct method and the method of unbiased estimate are found to be the best, since they started with a smaller normalized error and converged uniformly to zero.

Comparing the Widrow-Hoff algorithm and the Iterative LMS method from the convergence viewpoint, the Iterative LMS method is superior to the Widrow-Hoff algorithm.

B.5 Summary

Two adaptive techniques are compared. From a computational viewpoint, the Widrow-Hoff algorithm is less burdensome than the Iterative LMS method. However, the comparison of Wiener equation errors indicated that the solution from the Iterative LMS method satisfies Wiener equations better than that of the Widrow-Hoff algorithm.

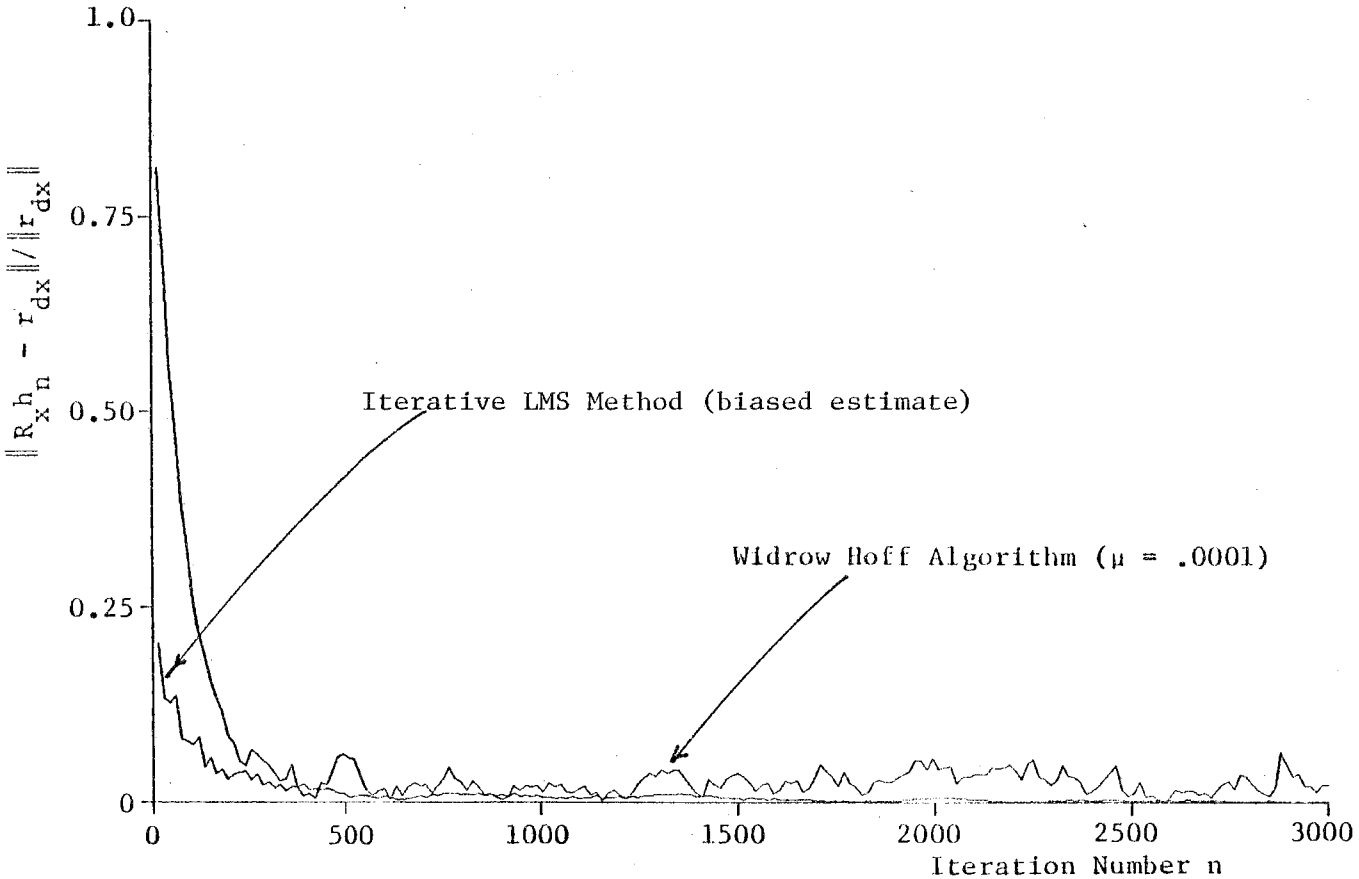


Fig. B.4.5 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 10)

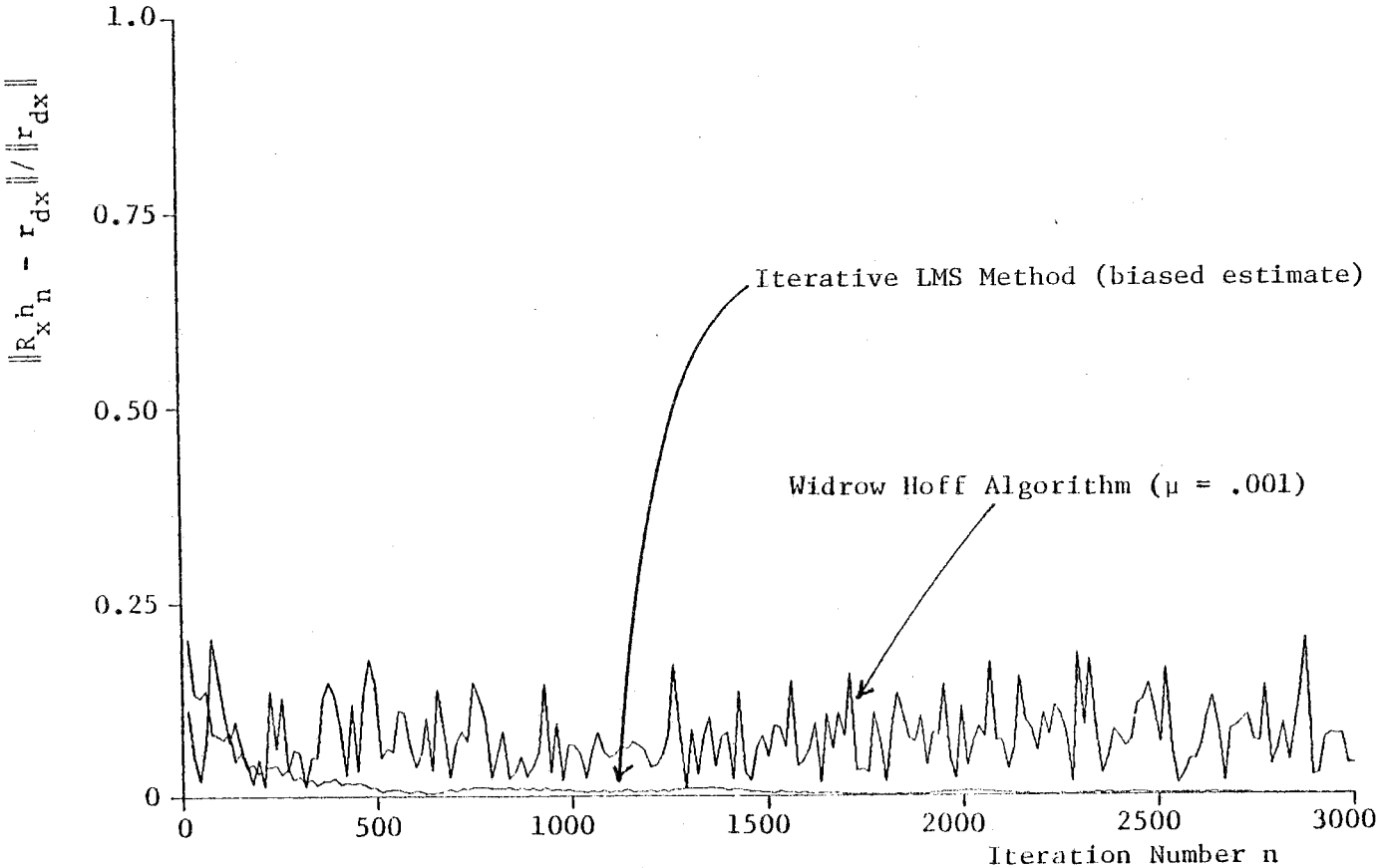


Fig. B.4.6 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 10)

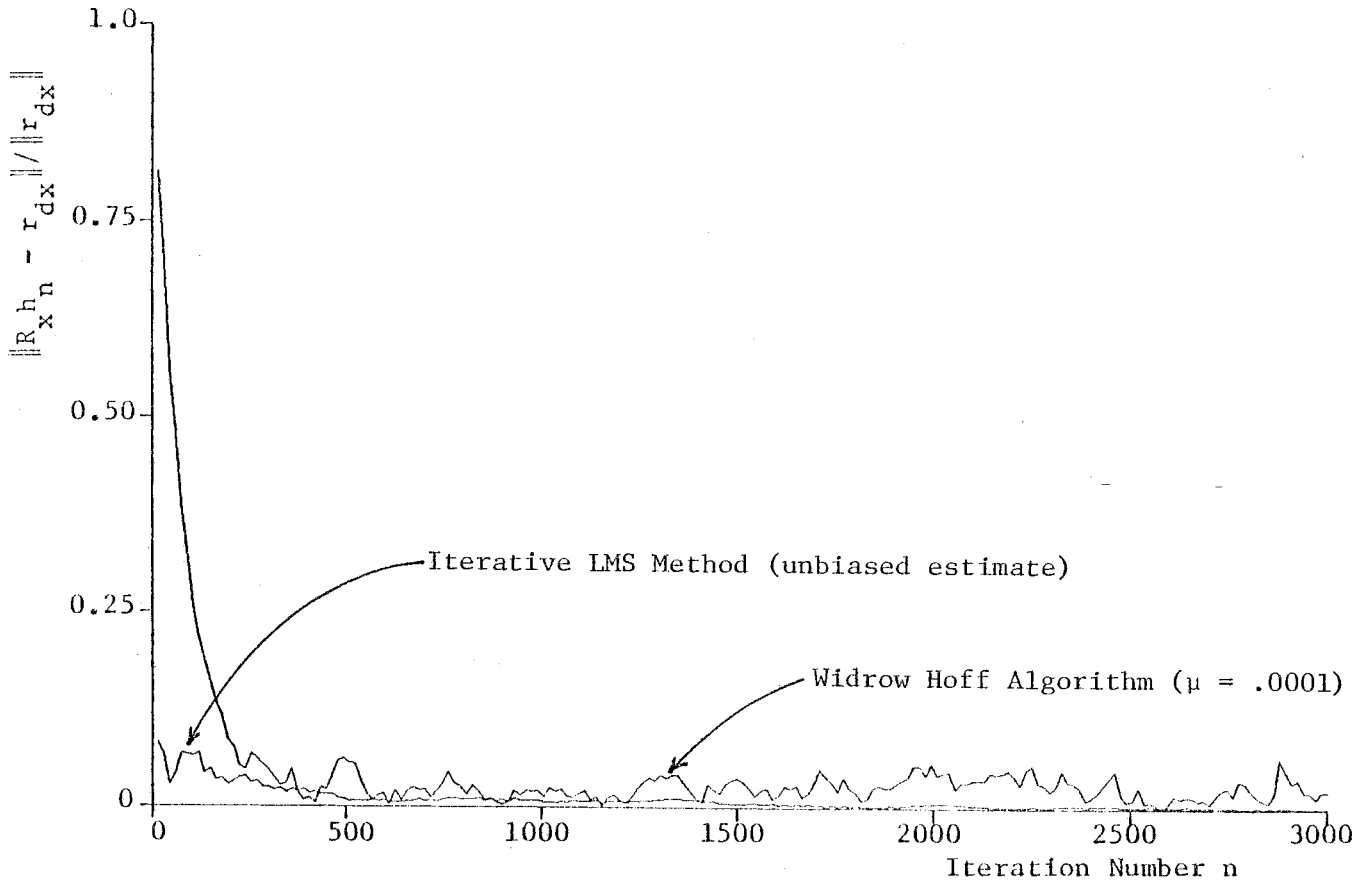


Fig. B.4.7 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 10)

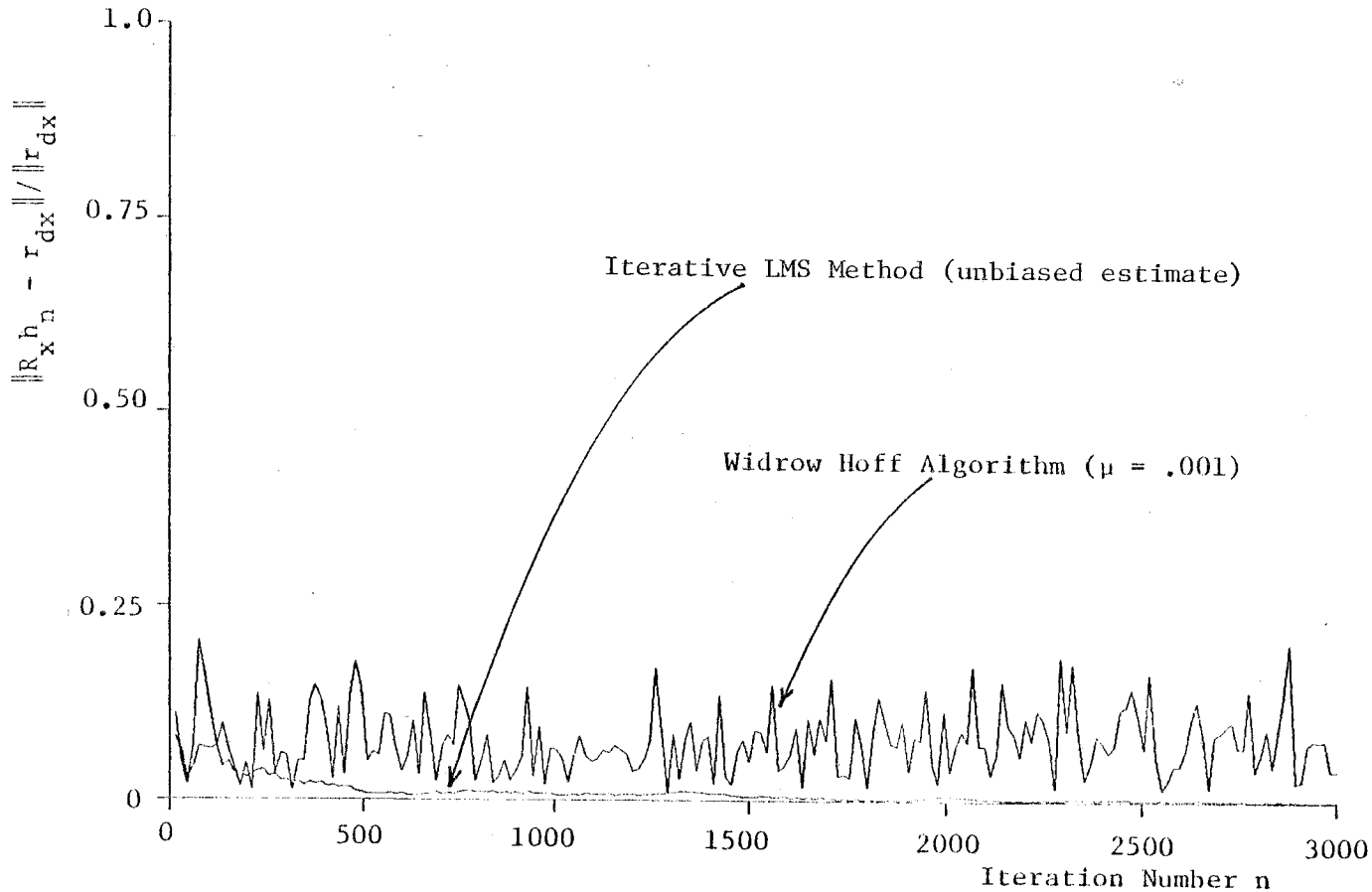


Fig. B.4.8 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 10)

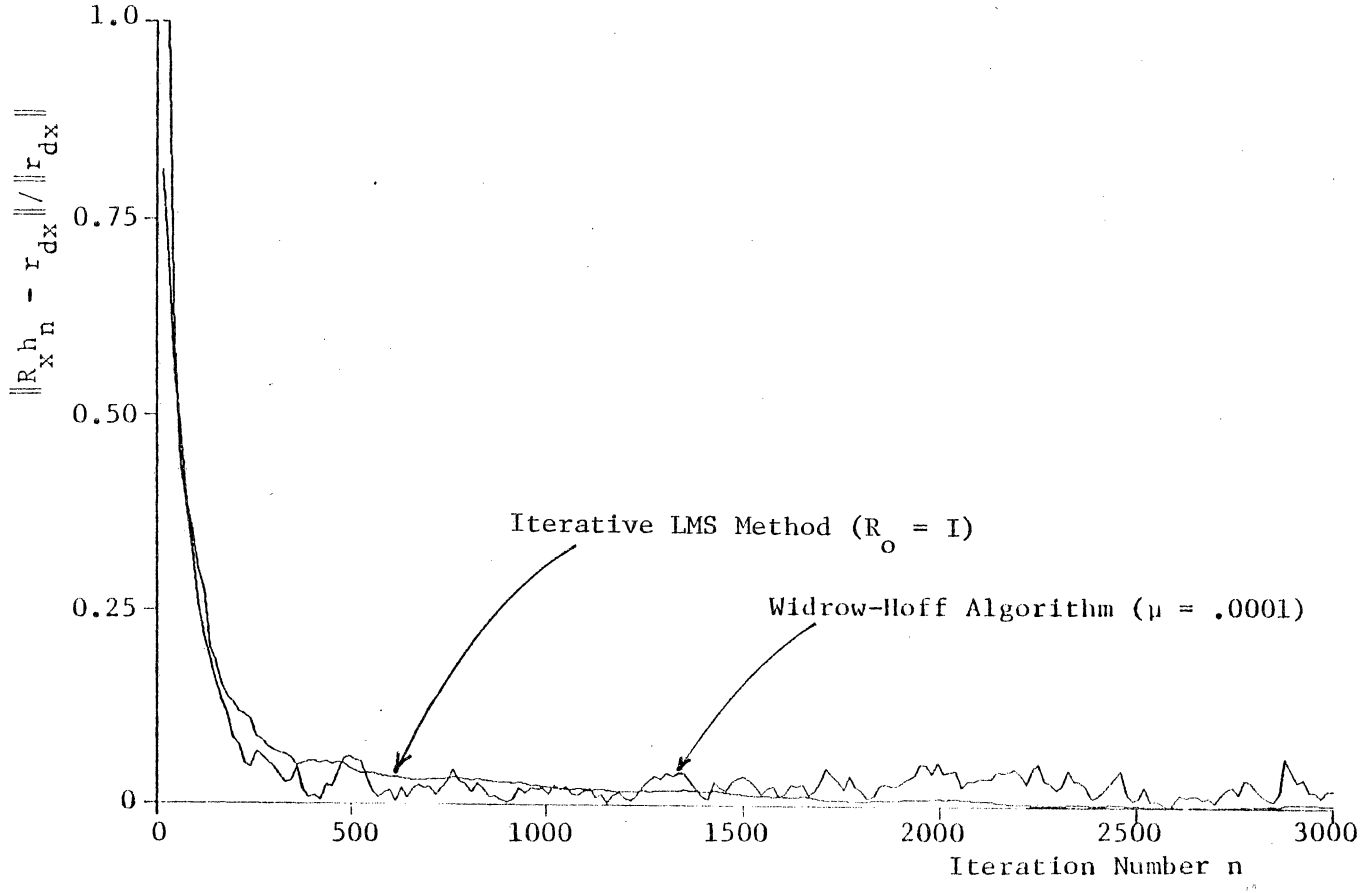


Fig. B.4.9 Normalized Weiner Equation Error for Two adaptive Algorithms (p = 10)

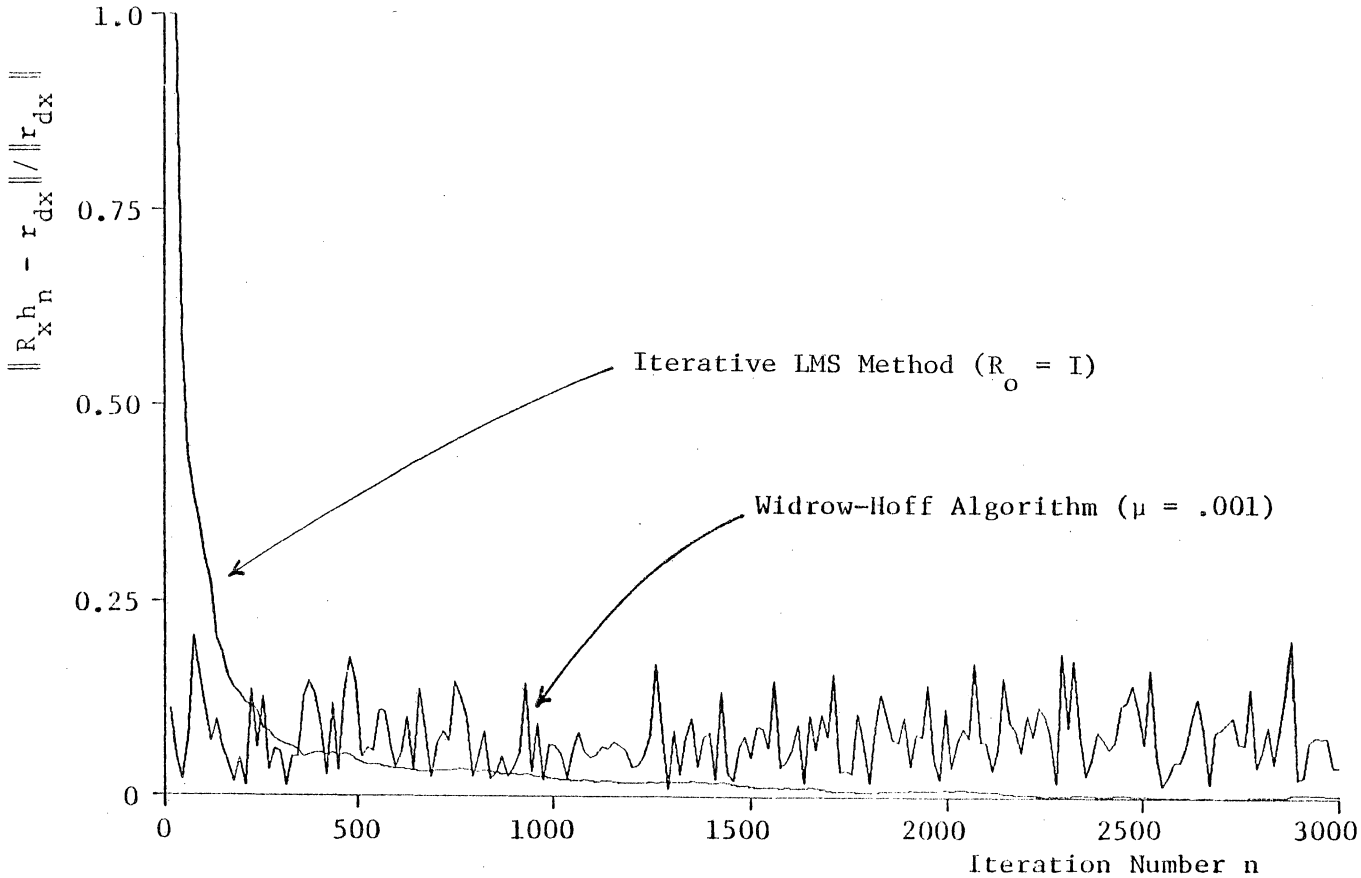


Fig. B.4.10 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 10)

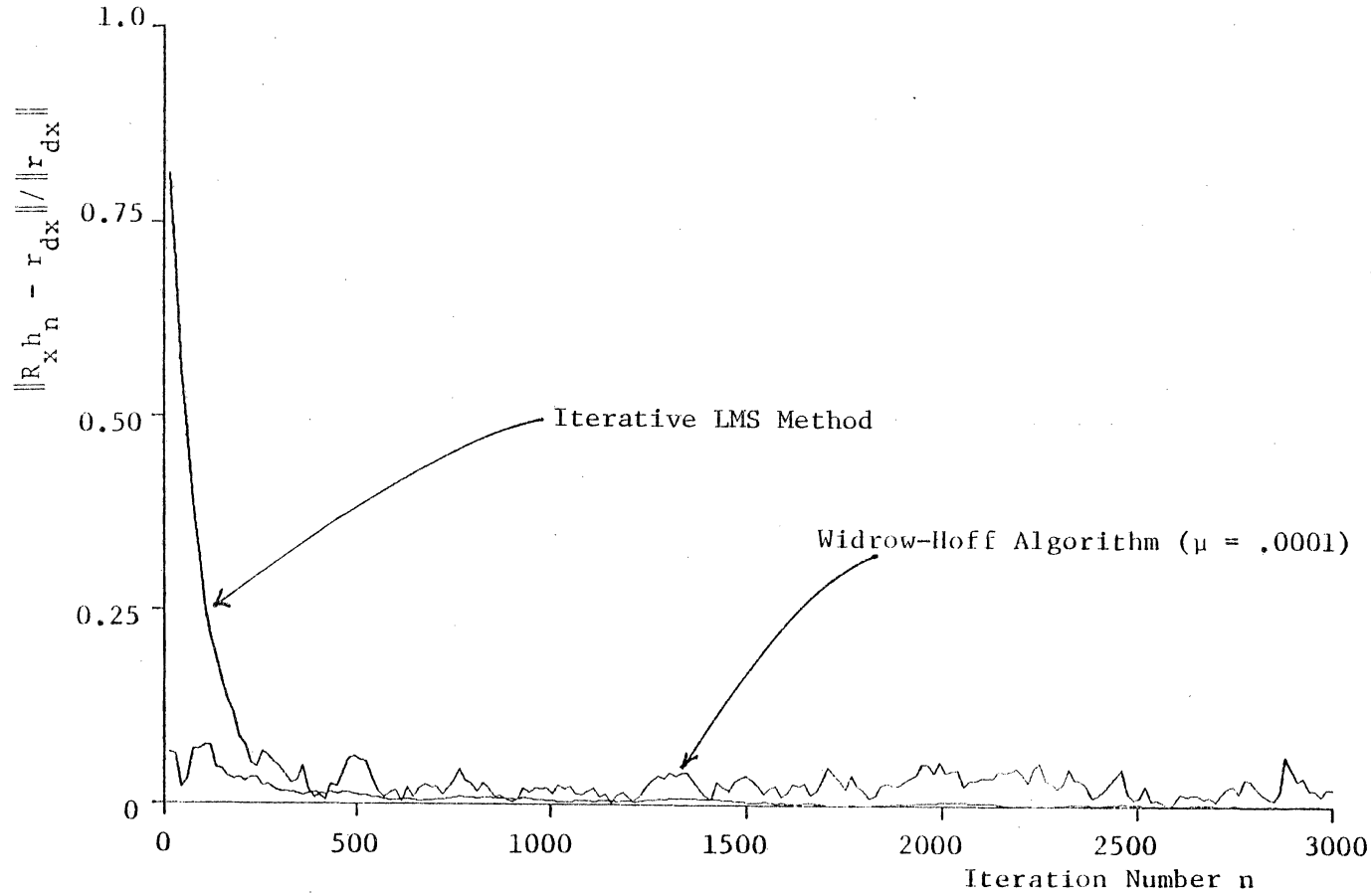


Fig. B.4.11 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 10)

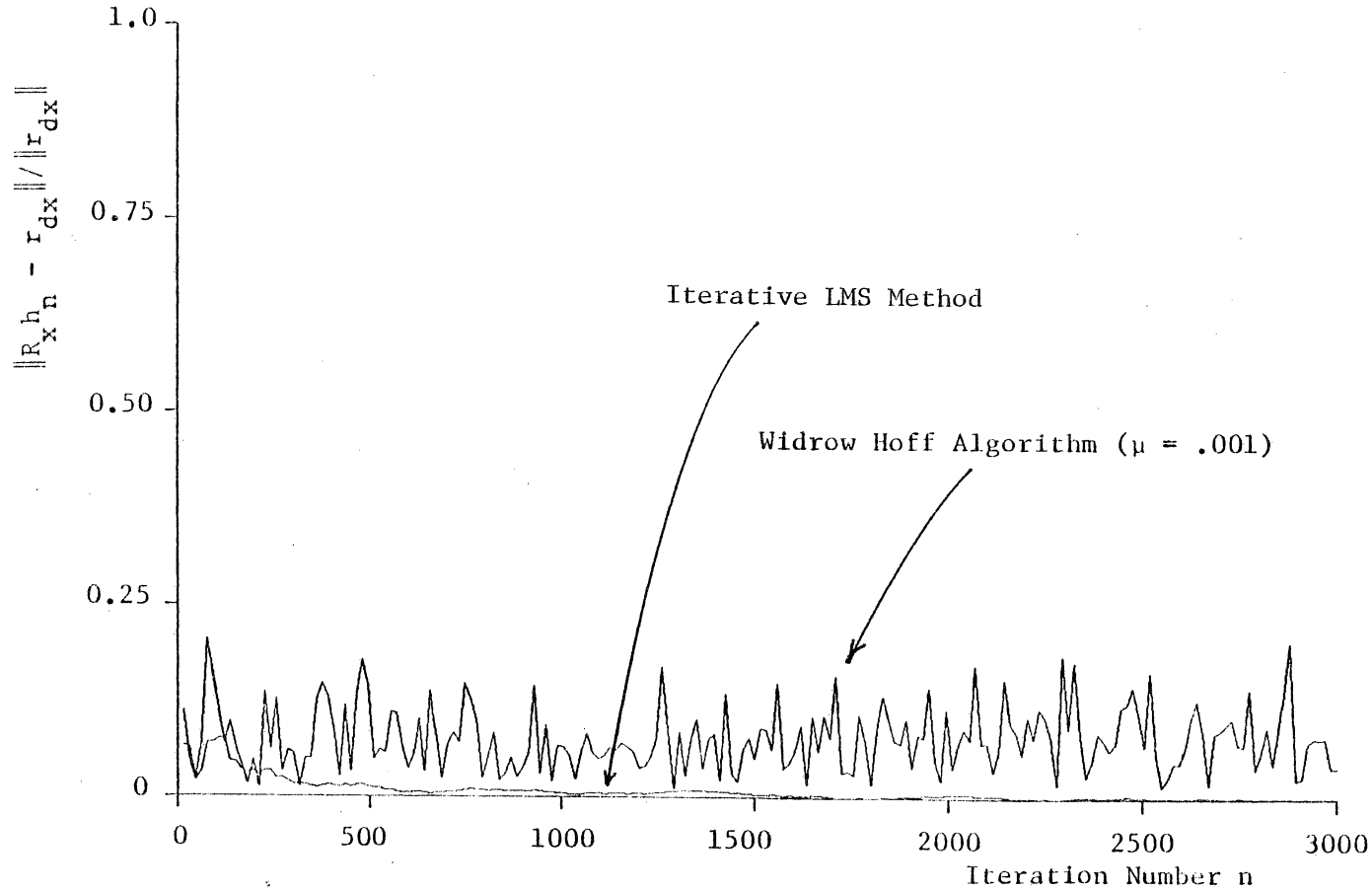


Fig. B.4.12 Normalized Weiner Equation Error for Two Adaptive Algorithms (p = 10)

Appendix C

DERIVATION OF EXPRESSIONS (6.6.4) AND (6.6.10)

In this Appendix, relationships (6.6.4) and (6.6.10) which play a central role for the time update mode are derived.

C.1 Derivation of (6.6.4)

Expression (6.6.3) can be simplified to the form

$$R_{n-1} = R_n - \underline{x}_n \underline{y}_n^{\dagger} \tag{C.1.1a}$$

where the (m-i+1)x1 column vectors $\underline{x}_n$ and $\underline{y}_n$ are defined by

$$\underline{x}_n = A^{\dagger} \underline{y}_{N[i,m]} \underline{e}_N \tag{C.1.1b}$$

$$\underline{y}_n^{\dagger} = \underline{e}_N^T A \underline{x}_{N[i,m]} \tag{C.1.1c}$$

It can be seen that R_{n-1} is expressed as a sum of a nonsingular matrix and a rank 1 matrix. Expression (C.1.1a) can be also expressed as

$$R_{n-1}^{-1} = R_n^{-\frac{1}{2}} [I - \underline{a}\, \underline{b}^{\dagger}]^{-1} R_n^{-\frac{1}{2}} \tag{C.1.2a}$$

where the (m-i+1)x1 column vectors $\underline{a}$ and $\underline{b}$ are defined by

$$\underline{a} = R_n^{-\frac{1}{2}} \underline{x}_n \tag{C.1.2b}$$

$$\underline{b}^{\dagger} = \underline{y}_n^{\dagger} R_n^{-\frac{1}{2}} \tag{C.1.2c}$$

in which $R_n^{-\frac{1}{2}}$ is a NxN matrix which satisfies the relationship $R_n^{-\frac{1}{2}} R_n^{-\frac{1}{2}} = R_n^{-1}$. We will now make use of the following matrix inverse relationship

$$[I - \underline{a}\,\underline{b}^{\dagger}]^{-1} = I + \left(\frac{1}{1 - b^{\dagger} a}\right) a\, b^{\dagger} \qquad \text{(C.1.3)}$$

Substituting (C.1.2b) and (C.1.2c) into (C.1.3) yields

$$[I - \underline{a}\,\underline{b}^{\dagger}]^{-1} = I + \frac{1}{1 - \underline{y}_n^{\dagger} R_n^{-1} \underline{x}_n} R_n^{-\frac{1}{2}} \underline{x}_n \underline{y}_n^{\dagger} R_n^{-\frac{1}{2}} \qquad \text{(C.1.4)}$$

Expression (6.6.4) can be obtained by substituting (C.1.4) into (C.1.2a) along with expressions (C.1.1b) and (C.1.1c).

C.2 Derivation of (6.6.10)

To simplify the complexity of notations, let us define the following compact notations

$$P = P\,\underline{x}_N[i,m] \qquad \text{(C.2.1a)}$$

$$Q = P_N \qquad \text{(C.2.1b)}$$

$$\gamma = \gamma_{i,m,N} = \underline{e}_N^T P\, \underline{e}_N^T \qquad \text{(C.2.1c)}$$

It is readily shown that

$$(1 - \gamma)\, Q = \underline{e}_N (1 - \gamma)\, \underline{e}_N^T = Q\,(I - P)\,Q \qquad \text{(C.2.2a)}$$

$$PQ\ PQ = \gamma PQ \qquad \text{(C.2.2b)}$$

$$QP\ QP = \gamma QP \qquad \text{(C.2.2c)}$$

Equation (6.6.9) may be expressed as

$$\nabla(I - P) = Q - P + (I - Q)\ P(I - Q)$$

$$+ \frac{1}{1 - \gamma}\ (I - Q)\ P\ Q\ P(I - Q) \qquad (C.2.3)$$

Using relationships (C.2.2b) and (C.2.2c), we have

$$\{Q - P + (I - Q)\ P(I - Q)\}\ (1 - \gamma)$$

$$= Q - PQ + \gamma PQ - QP + \gamma QP - \gamma QPQ \qquad (C.2.4)$$

Substitution of (C.2.4) into (C.2.3) yields

$$\nabla(I - P) = \frac{1}{1 - \gamma}\ (I - P)\ Q(I - P) \qquad (C.2.5)$$

Expression (6.6.10) can be obtained by direct substitution of expressions (C.2.1a), (C.2.1b) and (C.2.1c) into (C.2.5).

Appendix D

COMPUTER PROGRAM LISTING

D.1 FORTRAN Program Listing for a Recursive ARMA Spectral Estimation

D.2 FORTRAN Program Listing for Generalized Levinson's Approach of ARMA Model

Appendix D COMPUTER PROGRAM LISTING

D.1 FORTRAN Program Listing for a Recursive ARMA Spectral Estimation

```
C     THIS PROGRAM COMPUTES AUTOREGRESSIVE COEFFICIENTS OF
C     'HIGH PERFORMANCE' ARMA MODEL (REAL DATA, p=q).
C
       DIMENSION X(1024),EXN(30),EXNM1(30),BXN(30),BXNM1(30)
      $ ,FENM1(30),FRN(30),FRNM1(30),SN(30),SNM1(30),TN(30)
      $ ,GAM(30),GAM1(30),EYN(30),EYNM1(30),BYN(30)
      $ ,Y(1024),XA(1),RX(30,30),YX(30),YS(1024,30)
      $ ,WKAREA(30),CM(30),CM1(30),AM(30),AM1(30),BM(30)
      $ ,DM(30),RXX(30,30),FEN(30),TNM1(30),BB(30),BM1(30)
      $ ,XS(1024,30),BYNM1(30)
C
C     N1: TOTAL NUMBER OF OBSERVATION DATA
C     IP: ORDER OF DENOMINATOR COEFFICIENT
C
       N1=64
       IP=4
       NP=N1-IP
       N=N1-1
       IP1=IP+1
       IPM1=IP-1
C
C     GENERATE DATA TO BE MODELED
C
       DSEED=12345
       CALL KAVEH(Y,N1,DSEED)
       DO 25 I=1,NP
    25 X(I)=Y(I+IP)
       WRITE (6,101) (Y(I),I=1,NP)
       WRITE (6,101) (X(I),I=1,NP)
       N1=NP
       N=N1-1
C
C      INITIALIZATION FOR TIME UPDATE
C
       EXNM1(1)=0.
       EYNM1(1)=0.
       BXNM1(1)=0.
       BYNM1(1)=0.
       FENM1(1)=0.0
       FRNM1(1)=0.0
       A12=0.0
       A21=0.0
       A22=0.0
       DO 1 I=1,IP
```

```
      SNM1(I)=0.0
    1 TNM1(I)=0.0
C
C     UPDATE PARAMETERS FROM IT=1 TO IT=N1
C
      DO 2 IT=1,N1
      ITM1=IT-1
      AIT=IT
C     WRITE(6,102) IT
  102 FORMAT (/,3X,'N=',I3)
C
C     INITIALIZATION FOR ORDER UPDATE
C
      EXN(1)=X(IT)
      BXN(1)=X(IT)
      EYN(1)=Y(IT)
      BYN(1)=Y(IT)
      DO 20 I=1,IP1
      SN(I)=0.0
      TN(I)=0.0
      GAM1(I)=0.0
   20 GAM(I)=0.0
C
C     UPDATE FEN(1) AND FRN(1)
C
      FEN(1)=FENM1(1)+X(IT)*Y(IT)
      FRN(1)=FRNM1(1)+X(IT)*Y(IT)
      M=IP
      IF(ITM1.LT.IP) M=ITM1
      M1=M+1
      IF(IT.EQ.1) GO TO 109
C
C     ORDER UPDATE
C
      DO 3 I=1,M
C
C     UPDATE GAM(I+1) AND PARTIAL CORRELATION COEFFICIENT
C     SN(I) AND TN(I)
C
      GAM(I+1)=GAM(I)+BXNM1(I)*BYNM1(I)/FRNM1(I)
      SN(I)=SNM1(I)+BYNM1(I)*EXN(I)/(1.0-GAM(I))
      TN(I)=TNM1(I)+EYN(I)*BXNM1(I)/(1.0-GAM(I))
C
C     UPDATE FORWARD ERRORS EXN(I) AND EYN(I)
C
      EXN(I+1)=EXN(I)-(SN(I)/FRNM1(I))*BXNM1(I)
      EYN(I+1)=EYN(I)-(TN(I)/FRNM1(I))*BYNM1(I)
C
C     UPDATE BACKWARD ERRORS BXN(I) AND BYN(I)
C
```

```
      FEN(I+1)=FEN(I)-SN(I)*TN(I)/FRNM1(I)
      FRN(I+1)=FRNM1(I)-SN(I)*TN(1)/FEN(I)
    3 CONTINUE
  109 IF(IT.EQ.1) M1=1
C
C     PRINT OUT AT EACH NEW DATA POINT
C
      WRITE(6,100)
  100 FORMAT(/,2X, 'EXN(I)',3X,'EYN(I)',3X,'BXN(I)'
     $ ,3X,'BYN(I)',2X,'FEN(I)',2X,'FRN(I)',2X,'SN(I)'
     $ ,2X,'TN(I)',2X,'GAM(I)')
      DO 5 I=1,M1
      WRITE(6,101) EXN(I),EYN(I),BXN(I),BYN(I),FEN(I)
     $ ,FRN(I),SN(I),TN(I),GAM(I)
  101 FORMAT (2X,10F8.3)
      IF(IT.EQ.N1) GO TO 5
C
C     READY FOR NEXT DATA POINT
C
      EXNM1(I)=EXN(I)
      EYNM1(I)=EYN(I)
      BXNM1(I)=BXN(I)
      BYNM1(I)=BYN(I)
      FENM1(I)=FEN(I)
      FRNM1(I)=FRN(I)
      SNM1(I)=SN(I)
      TNM1(I)=TN(I)
      GAM1(I)=GAM(I)
    5 CONTINUE
      IF(IT.EQ.1) GO TO 2
      A12=A12+Y(IT)*X(IT-1)
      A21=A21+Y(IT-1)*X(IT)
      A22=A22+Y(IT-1)*X(IT-1)
    2 CONTINUE
C
C     FIND AUTOREGRESSIVE COEFFICIENTS FROM PARTIAL
C     CORRELATION COEFFICIENTS
C
      A11=FEN(1)
      DET=A11*A22-A21*A12
      CM(1)=(A22*Y(N1)-A12*Y(N1-1))/DET
      CM(2)=(-A21*Y(N1)+A11*Y(N1-1))/DET
      AM(1)=-A21/A22
      BM(1)=-A12/A11
      IF(IP.EQ.1) GO TO 23
      RM=X(N1)*BM(1)+X(N1-1)
      GM=X(N1)*CM(1)+X(N1-1)*CM(2)
      ETM=1.+RM*CM(2)/(1-GM)
      DO 13 IORD=1,IPM1
      IORD1=IORD+1
      IORD2=IORD+2
```

```
C
C     UPDATE AUXILIARY VECTOR DM(I)
C
        DO 14 I=1,IORD
     14 DM(I)=(BM(I)+RM*CM(I)/(1.-GM))/ETM
        TEMP=SN(IORD+1)/FRNM1(IORD+1)
C
C     UPDATE FORWARD VECTOR AM1(I)
C
        DO 15 I=1,IORD
     15 AM1(I)=AM(I)-TEMP*DM(I)
        AM1(IORD+1)=-TEMP
C
C     UPDATE BACKWARD VECTOR BM1(I)
C
        TEMP=TN(IORD+1)/FEN(IORD+1)
        BM1(1)=-TEMP
        DO 16 I=2,IORD1
     16 BM1(I)=DM(I-1)-TEMP*AM(I-1)
C
C     UPDATE AUXILIARY VECTOR CM1(I)
C
        TEMP=BYN(IORD2)/FRN(IORD2)
        DO 17 I=1,IORD1
     17 CM1(I)=CM(I)+TEMP*BM1(I)
        CM1(IORD1+1)=TEMP
        SUM=X(N1-IORD-1)
        SUM1=X(N1)*CM1(1)
        DO 18 I=1,IORD1
        SUM=SUM+X(N1+1-I)*BM1(I)
     18 SUM1=SUM1+X(N1-I)*CM1(I+1)
        RM1=SUM
        GM1=SUM1
        ETM1=1.+(RM1(1.-GM1))*CM1(IORD2)
C
C     SET VECTORS FOR NEXT ITERATION
C
        DO 19 I=1,IORD1
        AM(I)=AM1(I)
        BM(I)=BM1(I)
     19 CM(I)=CM1(I)
        CM(IORD2)=CM1(IORD2)
        RM=RM1
        GM=GM1
        ETM=ETM1
     13 CONTINUE
     23 CONTINUE
C
C     PRINT OUT AUTOREGRESSIVE COEFFICIENT
C
```

```
      WRITE (6,105) (AM(I),I=1,IP)
  105 FORMAT(/,3X,' RECURSIVE SOLUTION = ',//,10F10.5)
      RETURN
      END
```

NOTE: Above program may be applicable to complex data by making following changes

(i) Declare all variables to be complex value except integer variables (i.e. IMPLICIT Statement)

(ii) In DO loop 25, take complex conjugate on the variable Y(I+IP) (i.e. Y(I+IP)=CONJG(Y(I+IP)))

D.2 FORTRAN Program Listing for Generalized Levinson's Approach of ARMA Model

Generalized Levinson's approach discussed in Section 5.3 is programmed for the premodified method.

```
C
C     THIS PROGRAM COMPUTES DENOMINATOR COEFFICIENTS OF
C     ' HIGH PERFORMANCE ' ARMA SPECTRAL ESTIMATION
C     BY GENERALIZED LEVINSON'S APPROACH (REAL DATA, p=q ).
C
        DIMENSION X(64),FEN(30),YV(30),XV(30)
       ,FRN(30),FRNM1(30),SN(30),TN(30)
       ,BMN(10,10),BBMN(10,10)
       ,Y(64),RX(30,30)
       ,WKAREA(30),CM(30),CM1(30),AM(30),AM1(30)
        ,DM(30),RXX(30,30),BM(30),BM1(30)
C
C     N1: NUMBER OF TOTAL OBSERVATION
C
        N1=64
C
C     IP: ORDER OF DENOMINATOR COEFFICIENTS
C
        IP=4
        NP=N1-IP
        N=N1-1
        IP1=IP+1
        IPM1=IP-1
C
C     GENERATE DATA TO BE MODELED
C
        DSEED=12345
        CALL KAVEH(Y,N1,DSEED)
        DO 25 I=1,NP
     25 X(I)=Y(I+IP)
        WRITE(6,101) (Y(I),I=1,NP)
        WRITE(6,101) (X(I),I=1,NP)
    101 FORMAT(2X,10F8.3)
        N1=NP
        N=N1-1
C
C     INITIALIZATION BASED ON THE FIRST TWO DATA SAMPLES
C     X(1) AND Y(1)
C
        DO 40 I=1,IP1
        DO 40 J=1,IP1
```

```
      BBMN(I,J)=0.0
      BMN(I,J)=0.0
   40 RXX(I,J)=0.0
      A11=Y(1)*X(1)+Y(2)*X(2)
      A12=Y(2)*X(1)
      A21=Y(1)*X(2)
      A22=Y(1)*X(1)
      RXX(1,1)=A11
      RXX(1,2)=A12
      RXX(2,1)=A21
      RXX(2,2)=A22
      BMN(1,1)=-A12/A11
      FRNM1(2)=RXX(2,2)+RXX(2,1)*BMN(1,1)
C
C     SOLVE FOR DENOMINATOR COEFFICIENTS (AM(I),I=1,IP)
C     AT EACH NEW DATA POINT FROM IT=3 TO IT=N1
C
      DO 38 IT=3,N1
      IPM1=IP-1
      IF(IT.LE.IP) IPM1=IT-2
C
C     UPDATE ROW VECTORS
C
      DO 37 I=1,IP1
      YV(I)=0.0
      XV(I)=0.0
      IF(I.LE.IT) YV(I)=Y(IT+1-I)
      IF(I.LE.IT) XV(I)=X(IT+1-I)
   37 CONTINUE
      DO 39 I=1,IP1
      JF=I
      IF(I.EQ.1) JF=IP1
      DO 39 J=1,JF
   39 RXX(I,J)=RXX(I,J)+YV(I)*XV(J)
      A11=RXX(1,1)
      A12=RXX(1,2)
      A21=RXX(2,1)
      A22=RXX(2,2)
      AM(1)=-A21/A22
      BM(1)=-A12/A11
      IF(IP.EQ.1) GO TO 23
      DO 13 IORD=1,IPM1
      IORD1=IORD+1
      IORD2=IORD+2
C
C     COMPUTE AUXILIARY PARAMETERS FEN(IORD+1)
C     AND FRN(IORD+1)
C
      SUM=RXX(1,1)
      DO 27 I=1,IORD
```

```
   27 SUM=SUM+RXX(1,I+1)*AM(I)
      FEN(IORD+1)=SUM
      SUM=RXX(IORD1,IORD1)
      DO 28 I=1,IORD
   28 SUM=SUM+RXX(IORD1,I)*BM(I)
      FRN(IORD+1)=SUM
C
C     COMPUTE PARTIAL CORRELATION SN(I)
C
      SUM=RXX(IORD2,1)
      DO 29 I=1,IORD
   29 SUM=SUM+RXX(IORD2,I+1)*AM(I)
      SN(IORD+1)=SUM
      DO 14 I=1,IORD
   14 DM(I)=BMN(I,IORD)
C
C     COMPUTE PARTIAL CORRELATION TN(I)
C
      SUM=RXX(1,IORD2)
      DO 30 I=1,IORD
   30 SUM=SUM+RXX(1,I+1)*DM(I)
      TN(IORD+1)=SUM
C
C     UPDATE VECTOR AM1(I) ; FORWARD SOLUTION
C
      TEMP=SN(IORD+1)/FRNM1(IORD+1)
      DO 15 I=1,IORD
   15 AM1(I)=AM(I)-TEMP*DM(I)
      AM1(IORD+1)=-TEMP
C
C     UPDATE VECTOR BM1(I) ; BACKWARD SOLUTION
C
      TEMP=TN(IORD+1)/FEN(IORD+1)
      BM1(1)=-TEMP
      DO 16 I=2,IORD1
   16 BM1(I)=DM(I-1)-TEMP*AM(I-1)
      SUM=RXX(IORD2,IORD2)
C
C     COMPUTE AUXILIARY PARAMETER FRN(IORD2)
C
      DO 31 I=1,IORD1
   31 SUM=SUM+RXX(IORD2,I)*BM1(I)
      FRN(IORD2)=SUM
C
C     SET FOR NEXT DATA POINT
C
      DO 19 I=1,IORD1
      AM(I)=AM1(I)
      BBMN(I,IORD1)=BM1(I)
   19 BM(I)=BM1(I)
```

```
   13 CONTINUE
      BMN(1,1)=-A12/A11
      DO 43 I=1,IORD1
      DO 43 J=2,IORD1
   43 BMN(I,J)=BBMN(I,J)
      DO 41 I=1,IORD2
   41 FRNM1(I)=FRN(I)
   38 CONTINUE
   23 CONTINUE
C
C    PRINT OUT RESULTED DENOMINATOR COEFFICIENTS
C
      WRITE(6,105) (AM(I),I=1,IP)
  105 FORMAT(/,3X,' GENERALISED LEVINSON SOLUTION = '
     ,//,10F10.5)
      STOP
      END
```

NOTE: Above program may be applicable to complex data by making changes as described in Section D.1 (See Expression (5.3.1d)).

The vita has been removed from the scanned document

Computationally Fast Algorithms for ARMA Spectral Estimation

by

Koji Ogino

(ABSTRACT)

The high performance method for obtaining an ARMA model spectral estimate of a wide-sense stationary time series has been found to provide typically superior performance when compared to such comtemporary approaches as the Box-Jenkins and maximum entropy methods. In this dissertation, fast recursive algorithmic implementations of the high performance method are developed. They are recursive in the sense that as a new element of the time series is observed, the parameters characterizing an ARMA spectral estimate are algorithmically updated. The number of multiplications and additions required at each recursive stage are of the order p with p being the number of denominator coefficients of the ARMA model. Methods of modification of the data are applied to achieve a significant computational improvement. The development is predicated on utilization of various projection operators.